



**DUBLIN INSTITUTE  
of TECHNOLOGY**  
*Institiúid Teicneolaíochta Bhaile Átha Cliath*

# **Interactive Language Learning Tool for children with disabilities**

## **Final Year Project Report**

**DT228**  
**BSc in Computer Science**

**Kieran Hogan**

**Damian Bourke**

School of Computing  
Dublin Institute of Technology

**April 6th 2016**



## Abstract

Many people suffer from language learning disabilities, and can only communicate through the use of symbol systems. This project aims to address this by facilitating the target user who suffers from these disabilities, to understand vocabulary through the use of symbols and animations. This will be done through the creation of a tablet based application, which will offer an interactive learning experience for the user. The target user is very young and suffers from a variety of disabilities that greatly limit her communication and language learning skills. By providing a fun and interactive experience using symbol representation of language, with animations and games, the user will be engaged with a system that can potentially expand their understanding of vocabulary.

There will also be a web interface created for the parent/guardian/therapist of the user, which will offer feedback from the user's progress with the tablet application. This will offer them a level of insight for where the target user may be struggling or excelling in terms of symbol understanding and learning.

## Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

---

Kieran Hogan

4<sup>th</sup> April 2016

## Acknowledgements

I'd like to thank my supervisor Damian Bourke for his continued support and guidance throughout the project lifespan. I would also like to express my gratitude to Orlagh Gregory and her daughter Sophia. Without them the project would never have happened. I would also like to thank John Gilligan, and the other related project students for their time and regular feedback, which has proved invaluable. I would like to express my gratitude to N2y for allowing the use of their SymbolStix symbol set in the project application. Finally I would like to thank my parents for their constant support and patience.

## Table of Contents

Abstract .....	2
Declaration .....	3
Acknowledgements .....	4
Table of Contents .....	5
Table of Figures .....	8
1. Introduction .....	10
1.1. Project Overview .....	10
1.2. Project Objectives .....	10
1.3. Project Challenges .....	11
1.4. Structure of the document .....	11
1.5 GitHub Repository .....	12
2. Research .....	13
2.1. Background Research .....	13
2.1.1. Disabilities .....	13
2.1.2. Language Learning .....	15
2.1.3. Alternative Communication Approaches .....	16
2.2. Existing Solutions .....	19
2.2.1 Users Solution .....	19
2.2.2. Alternative Solutions .....	20
2.2.2.3. Toddler Robot .....	21
2.2.2.4. AtrikPix .....	22
2.3 Other Background Research .....	23
2.3.1. Vocabulary and Core Vocabulary .....	23
2.3.2 Interview/Contact .....	23
2.3.3. Other students doing community projects .....	24
2.3.4. Heuristics and Design principals .....	24
2.4. Technologies .....	25
2.4.1 Mobile Hardware .....	25
2.4.2 Mobile Development Platform .....	27
2.4.3. Web Hosting .....	28
2.4.4. Web Server technology .....	29
2.4.5. Web Application .....	29
2.4.6. Database .....	30

2.4.7. Data Visualisation.....	30
3. Design .....	32
3.1. Approach and Methodology.....	32
3.2.1. User Centred Design.....	32
3.2.2. Scrum.....	33
3.2. Project Analysis.....	34
3.2.1. Android Tablet Application.....	34
3.2.2. Web Application.....	35
3.2.3. Established Requirements.....	35
3.2.4. Unity App Storyboard .....	36
3.3. Project components .....	38
3.2.1. Unity - UI.....	38
3.2.2. Backend Unity and PHP .....	40
3.2.3. Web application - PHP, HTML5, Bootstrap and FusionCharts – UI .....	40
3.2.4. MySQL – database.....	40
3.4. Features and use cases.....	41
3.4.1. Unity App Use Cases.....	41
3.4.2. Web App Use Case.....	43
4. Implementation .....	44
4.1. System Architecture .....	44
4.2. Component Development and problems encountered .....	45
4.2.1. Unity application .....	45
4.2.1.5. Animations.....	64
4.2.2. Web Application.....	65
4.2.3. MySQL database .....	70
4.3. List of classes, scripts and files .....	71
Unity App .....	71
Web App.....	71
4.4. Additional Resources .....	71
5. System Validation.....	72
5.1. Testing.....	72
5.1.1 Unity Application Test Cases .....	72
5.1.2. Web Application Test Cases.....	75
5.1.3. User Testing.....	76
5.2. User Feedback .....	76

5.4 Demonstration .....	77
6. Project Plan .....	78
6.1. Overview .....	78
6.2. Original plan.....	78
6.2.1. Details on original phases.....	79
6.3. Project evolution .....	80
6.4. Project meeting and correspondence.....	81
7. Conclusion .....	83
7.1. Goals and objectives.....	83
To understand the users disabilities and how to reach her. ....	83
To develop a tablet based app for the particular user with her particular disabilities, keeping the specific user driven design at its core. ....	84
To assist the user to learn a chosen symbol set within this application, by an engaging means. ....	84
To capture data from the application and store it. ....	84
To create an interface for Orlagh to access feedback. ....	84
7.2. What I learned .....	84
7.3. Future Work .....	84
7.4 Personal Statement .....	85
8. Bibliography .....	86
Background Research .....	86
Technologies Researched .....	88
Design.....	90
9. Appendix.....	91
Appendix 1 – Custom built symbols .....	91
Appendix 2 – Test Case Template .....	92

## Table of Figures

Figure 1- SymbolStix.....	17
Figure 2 - Widgit symbols .....	17
Figure 3- Low tech board.....	19
Figure 4 - Symbols currently used by Sophia with Orlagh.....	20
Figure 5 - Grid 3 .....	21
Figure 6 - Toddler Robot .....	22
Figure 7- AtrikPix .....	22
Figure 8 - Tablet Market Share .....	25
Figure 9 - Android Nexus 7 .....	26
Figure 10 - Android Studio .....	27
Figure 11 - Unity3D Editor.....	28
Figure 12 - User Centrid Design cycle .....	32
Figure 13- Scrum framework.....	33
Figure 14 –User Requirements Table .....	36
Figure 15 – Tablet App Storyboard .....	38
Figure 16- Fitzgerald Key .....	39
Figure 17 - Prototype menu screen .....	40
Figure 18- Unity App: Sophia’s Use Cases .....	42
Figure 19 - Unity App: Orlagh’s Use Cases .....	43
Figure 20- Web App: Orlagh’s Use Cases.....	43
Figure 21 - System Architectural Model .....	44
Figure 22- Unity App: Main Menu.....	46
Figure 23 - Menu.cs - Scene Management .....	46
Figure 24 - CheckConnection.cs – Start .....	47
Figure 25 - CheckConnection.cs - Coroutine .....	47
Figure 26 - BgButtonSizeController.cs.....	48
Figure 27 - Unity App – Settings.....	49
Figure 28 - Dropdown.cs .....	49
Figure 29 - DropdownController.cs.....	50
Figure 30 - Unity App: Choose level screen.....	50
Figure 31 - Unity App: Library select screen .....	51
Figure 32 - Unity App: Library individual category screen: activities .....	52
Figure 33 - LibAnimation.cs.....	52
Figure 34 - Unity App: Guessing game screen .....	53
Figure 35- Unity App: Game success and failure .....	53
Figure 36 - Guess.cs - Correct/Wrong decider .....	54
Figure 37 - Guess.cs - Correct Answer .....	54
Figure 38 - Guess.cs - game logic.....	55
Figure 39 - Guess.cs - Score upload coroutine .....	55
Figure 40 - SceneMan.cs.....	56
Figure 41 - Unity App: Level 1 – Collect the coins.....	57
Figure 42 - Unity App: Level 2 – Collect sweets .....	57
Figure 43 - Unity App: Level 3 – Hide and seek.....	58
Figure 44 - Unity App: Level 4 – Chasing (being chased) .....	58



Figure 45 - Unity App: Level 5 – Chasing (chasing the cat) .....	59
Figure 46 - CameraController.cs.....	59
Figure 47 - PlayerController.cs - Update function.....	60
Figure 48 - PlayerController.cs – MovePlayer function.....	60
Figure 49 - PlayerController.cs - Button control functions .....	61
Figure 50 - NPC.cs.....	61
Figure 51 - Catcher.cs .....	62
Figure 52 - Chaser.cs .....	62
Figure 53 - Runner.cs.....	63
Figure 54 - Collectibles.cs .....	63
Figure 55 - Points.cs .....	64
Figure 56 - Hide.cs.....	64
Figure 57 - User Character State Machine.....	65
Figure 58 - Web App: Login Screen.....	65
Figure 59 - Web App: Login form .....	66
Figure 60 - Web App: Login form processing .....	66
Figure 61- Web App: Logout.....	66
Figure 62- Web App: Welcome Screen .....	67
Figure 63 - Session monitor.....	67
Figure 64- Graph2.php.....	68
Figure 65 - Web App: Connecting PHP to MySQL, and Disconnecting .....	68
Figure 66 - Web App: Using FusionCharts .....	69
Figure 67- addScore.php.....	70
Figure 68 - user table .....	70
Figure 69 - tracking table.....	70

## 1. Introduction

This chapter outlines the basis of the project and describes the inspiration behind it. It also outlines the main goal of the project, and how predefined objectives will allow this goal to be achieved. This will then be followed by a discussion of the challenges that will likely be faced, and the structure of the document structure.

### 1.1. Project Overview

This project was inspired by one particular individual, who was brought to the attention of DIT through community work. Sophia is only five years old and cannot learn language through conventional reading and writing methods, due to severe language learning disabilities. In October 2015, her mother, Orlagh, came to DIT to speak about her daughter, and to give a further understanding of the difficulties she faces. She has dyslexia, dyspraxia and autism. These all affect her language learning, as well as reducing her motor skills and cognitive ability.

The idea of this particular project was inspired by Orlagh's innate desire to help her child understand vocabulary. Symbol sets are collections of symbols that represent written language and are used in conjunction with communication systems by people with disabilities. The idea here was to develop a tablet based application that helps Sophia to learn a set of these symbols and expand her core vocabulary. However, this must be done in a fun and interactive manner as Sophia is only five years old. Another component of this project would be a web interface that tracks the user's data and outputs this for Orlagh. This would allow her to track her daughter's progress and receive valuable feedback to adjust her therapy.

### 1.2. Project Objectives

The main goal of the project is to facilitate the specific user in learning the meaning of symbols in a fun and interactive manner. Achieving this goal will require thorough research and design stages as building applications for users with disabilities is not something familiar to the developer. When implementing the project solution, user testing will also be a necessity to verify the success of the project. To ensure that this goal is achieved, regular contact will be made with Orlagh, who will be acting user on Sophia's behalf. S.M.A.R.T. goal management was used to establish the following goals. This means they are specific, measurable, attainable, realistic and timely.

The following objectives shall lead to the achievement of the main project goal:

- To understand the users disabilities and how to reach her.
- To develop a tablet based app for the particular user with her particular disabilities, keeping the specific user driven design at its core.
- To assist the user to learn a chosen symbol set within this application by an engaging means.
- To capture data from the application and store it.
- To create an interface for Orlagh to access feedback.

Some of these objectives are easier to accomplish than others, but there will likely be many challenges to face throughout the project life.

### 1.3. Project Challenges

Before beginning this project, it was clear there were many potential pitfalls and challenges. Creating a fun, enjoyable system that's user friendly, while also helping the user learn symbolic vocabulary would be undoubtedly challenging. Catering to this particular user and addressing her needs is the core focus.

It was also a known limitation of this project, that one on one contact with the user would not be possible throughout its lifecycle. Therefore the solution to this was to have the user's mother Orlagh act as the third party user. With this kept in mind, there is also a chance the target user may not get to use the application. However, knowing these limitations, the project is still addressing a gap in the market.

Time pressure is another likely challenge that will be faced throughout the project and will require regular meetings to maintain focus and receive feedback. It is possible that the project will not be fully complete within the deadline.

Since the basis of this project is not something familiar to the developer, it is likely that difficulty may be met when creating an application for any user with disabilities. To combat this, substantial research will be done surrounding the areas of these specific disabilities.

It is also possible that difficulty will be faced when implementing features and connecting project components. These are challenges that will likely be easier to overcome but may still affect the timescale.

### 1.4. Structure of the document

The project has been broken down into multiple sections. The first section covered will discuss all of the research done surrounding the background of the problem area. It will also discuss the research on various technologies that could be used in the project, as well as existing solutions to the problem.

The second section discusses the design of the project. This design of both the application system and more importantly the user interface is a core focal point of the project and requires careful attention.

The third section covers implementation. This includes everything in terms of development and system creation. Developing the tablet application, the backend server and database, and the web interface for our secondary user.

Testing and validation follows this section and is another very important section. Throughout this project, feedback and regular user testing was incredibly valuable in ensuring the design was consistent and attentive to the user's needs.

The fifth section covers the project planning aspects and discusses how the project evolved and took shape over its lifecycle. The project changed drastically over its lifetime and time management lead to deviation from the original plan.

That will bring the report to its conclusion where the success or failure of certain aspects can be examined. This is where the project goal and objectives are revisited to verify their completion. The report begins with the research section.

### 1.5 GitHub Repository

The code throughout the project was stored using GitHub. The repo can be found here:

<https://github.com/kieranhogan13/FinalYearProject>

## 2. Research

Research was a vital factor in carrying out this project. Identifying all of the factors surrounding the problem that this project addresses and researching existing solutions and alternatives was a necessity. This chapter has been broken down into background research and technology. The background section discusses the disabilities that affect the target user, as well as language learning with symbols and alternative learning methodologies.

The technology section discusses the technologies which were evaluated and selected for the project solution. This includes reasons for choosing certain programming languages, platforms and hardware.

### 2.1. Background Research

This section includes the core, non-technology based areas that needed to be researched for the project. The first of these main areas covers the variety of disabilities that affect the target user and how these impact her language skills. Then language learning will be discussed and the various learning techniques that come with this.

#### 2.1.1. Disabilities

There are a huge number of disabilities related to peoples learning abilities. They vary in intensity and severity between individuals, and can have a huge impact on people's lives. Conventional learning methods do not work with these individuals and as a result they require alternative teaching methods.

Many children and adults worldwide suffer from learning disabilities. These learning disabilities are often called learning limitations. In Canada a study was conducted that found *“among children aged 5 to 14, learning limitations (LLs) was the largest disability reported (about 69.3% of the children with disabilities)”* (Statistics Canada, 2006). Language based learning disabilities are a huge part of this, and they can be a complicated and delicate subset of learning disabilities. They affect written and spoken language, and can vary in severity. The disabilities can cause minor impedance of language use, to complete inability to speak or write (Newhall, 2012). Language learning is difficult to address as without being able to read, alternative methods of language representation must be used for language learning to be possible.

The target user of this system has a variety of disabilities. She suffers mainly from Dyslexia, Dyspraxia and Autism. These affect both her speech and her motor skills, and overall reduce her cognitive ability.

##### 2.1.1.1. Dyspraxia

Dyspraxia is a disability that affects mainly balance and coordination, and is summarised as a *“developmental coordination disorder”*. (Dyspraxia Association of Ireland, 2015) It has an adverse effect on a person's general functions, specifically balance, special awareness and behaviour. However it is not directly related to a person's intelligence. It is also more common in children, because many cases can be solved and nurtured with therapy in younger years (Dyspraxia Association of Ireland, 2015).

Children who live with dyspraxia often struggle with language learning and processing. In 2000, the Journal of Clinical and Experimental Neuropsychology published a study of a 5 year old child with dyspraxia. The study was conducted over two years and showed the neural effect of dyspraxia on the brain. It was found from the study that children with dyspraxia struggle with “*production aspects of language*” (Le Normand et al., 2000).

The target user of the project system has severe dyspraxia, and can only speak a couple of words. However, she understands many more words than she can speak. This clearly shows that she would benefit from a system that teaches her a symbol set for use in communication systems.

Typically it is important to treat people who suffer with dyspraxia with patience and care. Children who suffer from dyspraxia will usually be offered therapeutic help with wherever they struggle with the disability. It is important to note that dyspraxia is commonly related to other issues like dyslexia (Patino, 2014).

#### *2.1.1.2. Dyslexia*

Dyslexia is a learning disability that affects people’s ability to read and write. It is a lifelong disorder that can range from mild to quite severe, depending on individual cases. It can affect other areas like rhythm and learning numbers, colours and shapes. It is also commonly diagnosed along with other disabilities like Dyspraxia, and can dramatically affect language learning abilities (Dyslexia Association of Ireland, 2015).

Like dyspraxia, having dyslexia does not mean a person is less intelligent. People who suffer from dyslexia sometimes just require more time to process information if it is written, or may the information to be spoken to them through and audible source. Both dyslexia and dyspraxia are difficult disabilities to live with. However there is refined therapies and tools available to target specific areas to help individuals work around these disabilities. It is hoped that the application system will successfully become one of these tools, and help Sophia’s understanding of vocabulary.

The target user cannot read or write and due to the severity of her dyslexia she may never do so. Therefore helping to learn a symbol set would greatly improve her chances of using other systems that allow her to read, write or speak with symbols, would be a huge benefit to her. The final disability she was most recently diagnosed with was Autism.

#### *2.1.1.3. Autism*

Autism is a “*neuro-developmental disability*”. It is also known as Autism Spectrum Disorders, as it affects patients with a range of problems. These include serious communication difficulties and struggling with social interaction. (Williams, 2010) This spectrum can also extend beyond these issues, as people with Autism are more likely to have other mental health related issues, most commonly ADHD (attention deficit and hyperactivity disorder) and OCD (obsessive compulsive disorder). Therefore autistic individuals need to be diagnosed carefully and accurately. (Autism Speaks Inc, 2012)

When educating students with autism, it is important to interact with them in the correct manner. They can be quite anxious and fearful, so offering them a non-threatening form of

interaction is key. It is also important to keep them engaged by offering them a wide variety of activities. (Irish Society for Autism, 2012)

In order to facilitate Sophia's learning of a symbol set, the core project application will need to offer her an interactive and fun way to learn. This will help engage her in using the symbols and in turn broaden her vocabulary. She could then potentially use these symbols across a variety of systems. The potential for her to become bored with the application is somewhat likely, so keeping her interested is important.

### *2.1.2. Language Learning*

Language learning is the area of how we as humans develop communication skills. It can be summed up as a person's capacity to read, write and speak. From when we are very young, our exposure to spoken word is key to communication. Language learning disabilities are easy to detect early in a child's life. As a result it can be easier to provide treatment and extra assistance for them (Molnar and Sebastian-Galles, 2014).

#### *2.1.2.1. Language learning with Disabilities*

Teaching how to read, write and speak to a five year old child with learning disabilities is drastically different to teaching a normal five year old child. Teaching a normal five year old child usually involves them learning the alphabet and rhymes. At this age they would usually be speaking, and asking questions and have a capability to converse. (ReadingByPhonics, 2015)

However, teaching a five year old with learning disabilities requires a completely different approach. One similarity that is good practice across the board is reading to the child. However, it may be necessary to read a simpler story to a child with learning disabilities. It is also recommended to teach in a fun manner, for instance making a game out of learning. (Johnson, 2015). This will be further discussed in the following sections.

#### *2.1.2.2. Learning with Video Games and Serious Gaming*

Video games are a controversial form of entertainment. They are considered by some to promote violent and abnormal behaviour. However, studies have been conducted which lead us to believe they are beneficial for mental health and coordination. People who suffer from learning disabilities benefit greatly from interactivity and video games. Certain attention must be made to aspects of the game however. It is important for the user to be rewarded for successful completion of levels in the game and for this reward to be worthwhile and enjoyable for the user. It is also important for challenging aspects of a game to meet an appropriate difficulty for the user. This can be established by having the difficulty increase, as the user progresses through the game. (Kulman, Watkins, and Carlson, 2012) This will also contribute greatly to teaching the user symbols and what they represent.

Serious Gaming is a well-established area of utilising games for learning. They offer a motivational and enjoyable interaction for users and are far more engaging than standard teaching practices. (Rooney, 2012) It is clear that implementing a serious game aspect to this project that allows the user to interact would benefit the target user greatly.

### *2.1.2.3. Learning with Animation and Sound*

Animation offers students the opportunity to see a vivid representation of the subject (Bates, 2013). This animated representation of words for example, can closely relate to real world examples and helps the students to understand their meaning. It is also important to acknowledge the importance of sound in learning. From when we are very young children, we learn to associate sounds with words. Although it is less clear to tell where words end and begin in everyday spoken language, hearing words spoken slowly and isolated is integral to learning to speak. The sounds of words and the meanings are vital to language learning. Phonics and phonemes, which are the sounds associated with words, are integral to learning to speak. (Saffran, 2014). The use of animation and sound will drive one of the game aspects of this system and in turn help to meet the objective of teaching the user vocabulary through symbols.

### *2.1.3. Alternative Communication Approaches*

According to the American Speech-Language-Hearing Association, every individual with a disability has communicative rights. These rights were originally established in 1992 by the National Joint Committee for the Communication Needs of Persons with Severe Disabilities. (NJC, 1997) These rights include every person having the right to request and accept the right to refusal and rejection, the right to self-expression, and many others (American Speech-Language-Hearing Association, 2014).

In order to help Sophia to communicate, different approaches must be examined. There are limited options that include symbol sets, symbol systems, Augmentative and Alternative Communication (AAC), low tech boards, and sign language

Enabling these users to communicate as normal can be done so by many means, depending on the disability. Symbol sets are one such successful way.

#### *2.1.3.1. Symbol Sets*

Symbol sets are collections of symbols that offer people with disabilities an alternative to written language. They do so by representing words with familiar and universally understandable images. However they can vary greatly in quality of design and imagery. For the project at hand, use of a reputable and reliable symbol set will be very important.

There are many symbol sets that exist. Some of these are free to use but are of poor quality. Alternatively there is a set called Symbol Stix made by a company called n2y. These require a license to use and have many different price points. The symbols they provide are created specifically with children in mind and are considered one of the best symbol systems available. Their symbols are represented by stick men and are very clear and understandable. (n2y, 2016) They can be seen below in figure 1.



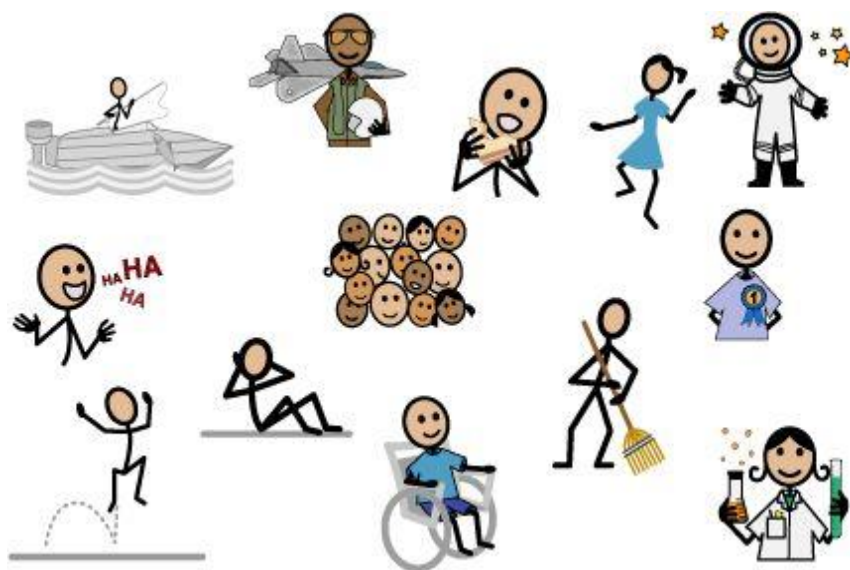


Figure 1- SymbolStix

Widgit Symbols are another set of set of symbols that offer easily understood words. However these are slightly more abstract, and slightly less child friendly as a result. Thus it was concluded from examining them that they could be considered less appropriate for the target user. (Widgit Software, 2015). See Figure 2.



Figure 2 - Widgit symbols

Using the correct symbol set in the application is key to its success. A symbol set must be agreed upon and used across all systems Sophia will be using. It was agreed in a project meeting with Orlagh that SymbolStix would be the most appropriate set to use. In turn all related projects would use this agreed set. This would mean these systems could overlap in usefulness and expand the user's vocabulary. The project supervisors contacted n2y, and were good enough to give each student the use of an academic licence, and access to their web portal for downloading symbols.

### *2.1.3.2. Symbol Systems and AAC*

Concrete symbol systems are systems that implement symbols that accurately and easily represent counterpart reality objects and words. These symbols are typically then used for communication purposes. These symbols can range from everything between actions to objects. Tangible systems are a more detailed area of concrete symbols that look exactly like real world objects that the user would be familiar with. These are better for children with very severe disabilities (Rowland and Schweigert, 2000).

Learning from symbolic objects is a tricky and somewhat complex area. Representing a language as symbols is complex as the symbol itself is an object and its representation is of another object. The most important factor of symbols is that its symbolic representation is understood as universally and easily as possible. From a study documented by Uttal et al., it was found that concrete objects being used by children as symbols helped them to understand their meaning (Uttal and DeLoache, 2006).

Augmentative and Alternative Communication is solely about effective communication. A huge area of AAC involves the use of symbols. Minspeak was one of the first of these, and it did so with sequences of symbols and required no literary skills (Baker, 2009). Important factors in AAC include accessibility, ease of use, and clear audio speech. It is intended that the project application will implement similar rules and principals. (Prentke Romich Company, 2008)

### *2.1.3.3. Low Tech Boards*

Low tech boards are customised alternatives to AAC that can be used by communication impaired individuals. They involve the user pointing or clicking the symbols of the words to communicate (which may not always be possible, depending on the user's particular disabilities). PRC Unity board is a very low tech board that requires very little technical knowhow. It comes in the form of an easily printable PDF (Prentke Romich Company, 2015). LAMP (Language Acquisition through Motor Planning) board is another board similar to the PRC Unity board. This one uses the methodology of LAMP however. This methodology involves the importance of sequencing of symbols to create logical sentences (Prentke Romich Company, 2015). Both of these systems are effective for disabilities, but won't necessarily work for our target user.

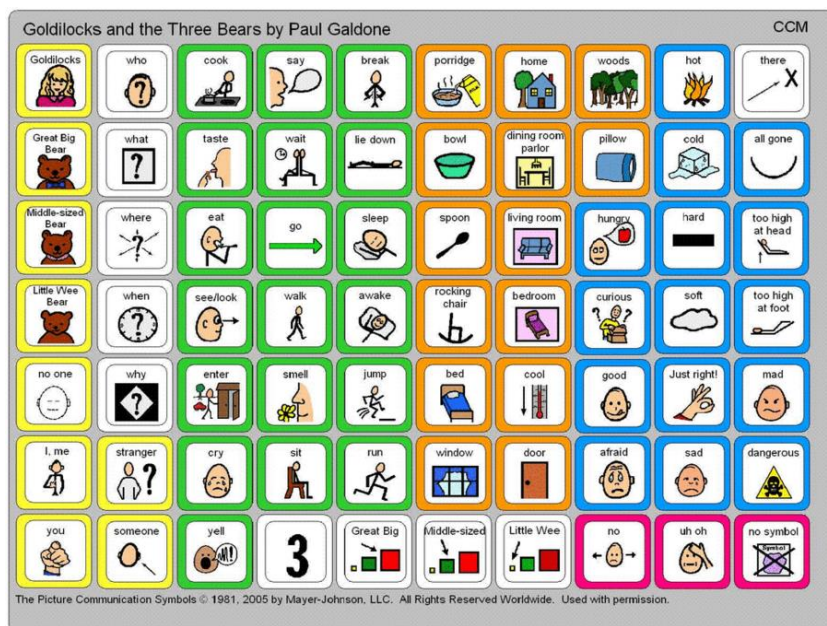


Figure 3- Low tech board

Figure 3 shows an example of a low tech board, specifically for telling a story of the three bears. The grid of 70 symbols seen here is something that could potentially overwhelm a user, therefore much smaller low tech boards will commonly be used. They however are not an ideal solution to the problem at hand.

#### 2.1.3.4. Sign Language

Sign language is another alternative communication system, however in some regards it can be seen as just as difficult to learn as written and spoken language (Foundation for People with Learning Disabilities, 2015). Sign language can sometimes be quite a viable option for language learning, and it can be quite closely related to symbolic systems. Makaton is a sign language style symbolic system developed for people with a variety of disabilities and difficulties (Makaton, 2015). However, due to our user's severe dyspraxia, sign language is not a viable alternative for communication. Her cognitive limitations prevent her from forming shapes with her hands other than pointing. Therefore, other alternative solutions must be examined.

## 2.2. Existing Solutions

Currently, the most common solutions to the problem of understanding the symbols in AAC systems, are achieved manually. These manual means involve tutoring from a therapist or parent with the use of printable activity sheets with symbols on them, like those provided by companies like PRC (Prentke Romich Company, 2015).

### 2.2.1 Users Solution

Orlagh's current solution to teaching her daughter core vocabulary involves the use of a customised symbol set. This symbol set is a hybrid of symbols from SymbolStix (n2y, 2015) and Widgit Symbols (Widgit Software, 2015). This limited set of symbols forms the base of communication vocabulary available to Sophia, however she cannot speak most of these.

However, her understanding of the words is paramount to her using symbol communication systems. Figure 4 shows the table of symbols of her current vocabulary.

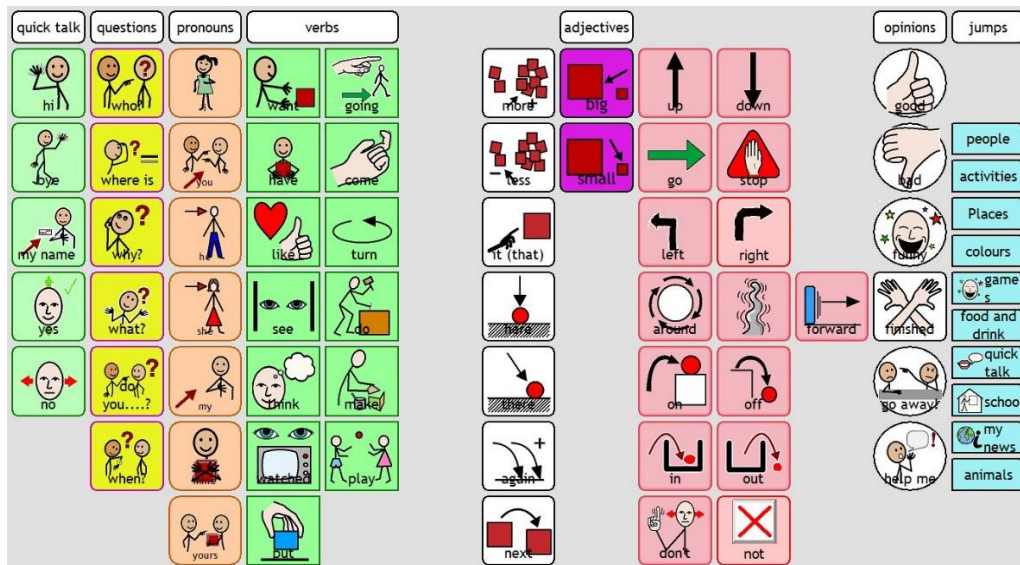


Figure 4 - Symbols currently used by Sophia with Orlagh

Orlagh also provided the work task list she uses to keep track of her progress with Sophia. She manually picks a symbol that represents the word, and arranges them in groups. She then encourages Sophia to use these core words to narrate her activity. She uses these words in different sentences and stories. She tries to encourage Sophia to use these symbols in different scenarios, to show the diverse usage of the words. For instance, she would help Sophia understand that turning can be used in a directional sense while driving per say, but also for turning on a light switch.

Orlagh keeps track of Sophia's attempts, successes, the context of usage and words she prefers to use. This method although long and difficult, is at the heart of the manual method that the project system is intended to automate. By offloading some of the manual elements, Sophia will hopefully be able to learn symbol meanings from the application.

### 2.2.2. Alternative Solutions

Some automated solutions do however exist to the problem this project addresses. These can however be very expensive and problematic. One such company that has addressed the problem successfully is Smartbox, with their system Grid 3.

#### 2.2.2.1. Grid 3

Grid 3 is a software package that enables disabled users to communicate. This software is a tablet based multi-tool for users, which allows them to communicate via symbols, learn with interactivity, and provides great accessibility features. It allows the user to use their desired symbol set, by default being Widgit Symbols. This then allows them to use an AAC communication board, and partake in learning activities using these. (Smartbox, 2016) One of these can be seen in figure 5 below.





Figure 5 - Grid 3

#### 2.2.2.2. *ASC-Inclusion*

ASC-Inclusion is a project being developed in Technische Universität München in Germany. The project is aimed at children with Autism, and is designed to help them understand emotions. They do this by offering interactive games that help with understanding facial expressions and gestures. (Björn Schuller, 2012) This application is close to release and will definitely help patients who suffer from Autism.

#### 2.2.2.3. *Toddler Robot*

On the Android market, there is a free application called Toddler Robot. This application is aimed at very young children, and offers the user animations for a small selection of words, like objects and colours. The symbols the application uses are not universally known but are only relevant to the application. In this sense they are not relevant to any AAC system and Toddler Robot is a local standalone system. The goal of the application is to animate words for the user and explain their meaning. However this system would not exactly complete the task that the project system will be designed to achieve, as it uses crude animations and unclear symbols. The element of having a library of the symbols, words and animations is something that would fit well in this project (Russpuppy, 2015). Screenshots of this app can be seen in Figure 6.



Figure 6 - Toddler Robot

#### 2.2.2.4. AtrikPix

AtrikPix is a symbol based learning app that can be found on the Apple Store. It presents a colourful interface and makes a game of matching symbols to words. This application is more advanced than Toddler Robot however, as it requires a more advanced base level of English. The symbols used here are SymbolStix, which immediately make this app more user friendly than that of Toddler Robot. By using a universal and well known symbol set, the application can lead to the user becoming more comfortable with other symbol based applications that use SymbolStix. (Expressive Solutions, 2013). Figure 7 below shows the interface for AtrikPix.



Figure 7- ArtikPix

From looking at these separate applications, there is a very distinct and universally simple idea among them; using symbols to reach the user. None of the applications address the particular system goal or objective, apart from Grid 3. However Grid 3 is an expensive solution. From studying the manner in which these systems aim to teach users symbol systems, the project system will implement SymbolStix. From the background research conducted, it is clear that the use of animations and serious gaming aspects is a good way to teach a user vocabulary with symbols.

## 2.3 Other Background Research

### 2.3.1. Vocabulary and Core Vocabulary

According to AAClanguagelab.com, vocabulary can be broken into core and fringe vocabulary. Core vocabulary has many characteristics. It is what makes up the majority of a sentence or conversation, and is typically words which are used frequently. They can apply to a broad number of topics and environments, and are typically used repeatedly in conversation. Examples of core vocabulary are universal pronouns like ‘you’ and ‘me’, verbs like ‘do’, ‘like’ and ‘watch’, and question words (Prentke Romich Company, 2015).

Core vocabulary is the most integral part of language vocabulary. The idea of focusing on teaching people with language learning disabilities these core words is that they are “*re-usable*” and “*promote generative language*”. (Vinson, 2015). This means that the student learning these core words is more likely to initiate new conversation, rather than following a previously memorised dialogue (Vinson, 2015).

Other articles and websites have outlined similar and some different common words used and required for communication. These include words like “*on*”, “*off*”, “*go*”, “*stop*” and “*away*” (VanTatenhove, 2005). These are typical words that can be used in a multitude of scenarios.

### 2.3.2 Interview/Contact

Throughout the research phase of this project, there has been direct communication with Orlagh, the parent of a girl who would benefit greatly from the project. She has also studied child mental health and has relevant qualifications regarding the area. Therefore constructive feedback and responses to the ideas has been received, and the project has been tailored and adjusted appropriately. Throughout the course of the project, Orlagh will be consulted with to ensure project focus, and hopefully there will be an opportunity to have her and her daughter test the system.

From contact with Orlagh, she was able to give advice on some features and elements that would be good to implement. These include a strong cause and effect element for the user. This would be implemented by rewarding the user on successful interactivity, with say a round of applause or cheering for example. She also mentioned that audio enforcement of the word in each section would be beneficial, to have the user become familiar with the sound of the word. Orlagh also provided the symbols that her daughter currently uses for AAC systems, which is a perfect starting point for symbols to use in the project system. These were from a combination of SymbolStix (n2y, 2016) and Widgit Symbols (Widgit Software,

2015). She also provided a work list for her core vocabulary, which can be used as a guide for the vocabulary of the proposed implemented learning system.

### 2.3.3. Other students doing community projects

Email correspondence has been made between the students doing other community based projects. Throughout the lifetime of the project, many meetings were had between the students, the project supervisors and Orlagh. These gave the students an opportunity to develop more substantial projects, and to also allow the projects to potentially work together to solve their target problems more effectively.

Due to limitations surrounding the community based projects, Orlagh was our acting user. Orlagh gave substantial feedback and resources, which helped our projects evolve and grow. This will be discussed more in the Design and System Validation sections.

### 2.3.4. Heuristics and Design principals

An important element of this project will be attention to design principals. Design principals are necessary in all software interface designs, however in this system our user is at the core heart of the design. Therefore, Nielsen's Heuristics will be used. There are ten principals of these which are as follows:

- Visibility of system status: Have a responsive system, in this case praising the user for success on goal completion.
- Match between system and the real world: Use symbols and words familiar to the user, which in this case is the chosen symbol set.
- User control and freedom: Allow the user to navigate freely. This will be done using the symbols to represent buttons allowing free navigation for the user.
- Consistency and standards: Consistency in buttons and definitions of symbols leading to where they represent.
- Error prevention: Use effective error messages, which does not really apply to our system. However a check will be made to ensure it connects to the server.
- Recognition rather than recall: User should recognise, should be guided with tutorials and familiar buttons.
- Flexibility and efficiency of use: Having the system customisable in terms of icon size and background colour for instance, will be implemented in settings.
- Aesthetic and minimalist design: Only showing what is needed and relevant, keeping the buttons large and meeting accessibility standards.
- Help users recognize, diagnose, and recover from errors: Help user with error recovery. This does not apply to the system.
- Help and documentation: Provide documentation for system help.

In following these ten heuristics, and tailoring the system around your user, you design a system that is created with your user's ability to use the system as a primary focus. (Nielsen, 1995) Focusing on all ten may be difficult, but will be attempted in designing the system. The use of familiar symbols for the user is key here, for designing minimal easily navigable menus.



Everything in this systems interface is important, from the colour of the symbols being used, to the layout and position of the buttons. All of these will be necessary to ensure the user driven design objective is met. As a result, careful attention will be paid throughout the projects interface design.

## 2.4. Technologies

This area contains all technology based research for the project solution. This ranges from potential platform choices to programming languages and frameworks. The technologies for this system were chosen based on which were most appropriate for the job, but also those which the developer would have confidence in being able to successfully use to complete the project. Those technologies unfamiliar to the developer, that have not used before would require self-teaching, and would need to be familiarised to a sufficient level for project completion.

### 2.4.1 Mobile Hardware

The main application of this project needs to be built for a tablet mobile platform. The system user already has experience using both IOS and Windows platforms, and therefore designing the system for tablet devices is a clear cut choice. This platform will require graphics and animation capabilities for the main application. It will also require the ability to connect to the server layer of the application. There are many options to consider in choosing what platform to use. These include cost effectiveness, difficulty of developing, licensing, etc. Popularity and market share is also an important factor. Below in Figure 8 is a table of statistics that shows how Android has risen to dominate the global tablet market share. (Statsista, 2015) As a result a larger variety of them are now available, bringing cheaper ones to the market, and making them more readily available for users.

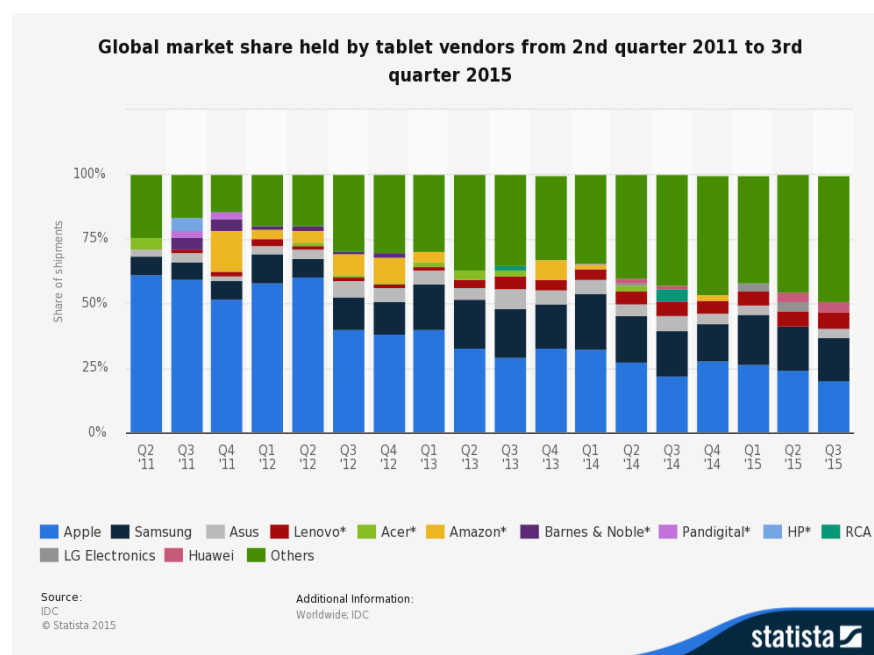


Figure 8 - Tablet Market Share

## *IOS*

Firstly Apple IOS was researched and considered. Many would consider IOS to be the preferential platform to develop tablet based applications for. However, it was found that with IOS a license which costs \$100 is required to publish developed applications (Monus et al., 2015). It must also be noted that access to development on this platform is more difficult than others, as development for IOS is usually done on Macs in XCode. There are alternatives for Windows, but they lack the support and documentation required throughout the project development. Apps for IOS are generally written in Swift which the developer has never used before, and would require additional learning of another OOP language, which is not an established goal of this project.

## *Windows*

With regards developing on the Windows tablet platform, it is clear that the lack of popularity of these devices in comparison to Android and Apple to be a significant factor. Windows were late to entering the mobile platform market in comparison to the others who dominate the market. (Rubens, 2014). As a result, it is more logical to develop the application for a system which is likely to have more potential users.

## *Android*

As seen in Figure 4 above, Android is now the market leader platform for tablets. It dominates the market due to a number of factors, mainly being the cost effectiveness of devices. Google's Android is an open source operating system, meaning any manufacturer can use the operating system on their devices. As a result these devices come in various sizes and price ranges, and are made by a huge range of manufacturers (Ashokan, 2015). These are the main reasons Android has been chosen as the platform to develop on.

It is also important to note that the developer already has access to Android hardware leading to easier and faster development. A second generation Google Nexus 7 tablet device will be used for developing purposes, which is a typical mid-range Android tablet. The project developer also has experience developing on Android from previous college assignments in other modules. In the past the developer has found developing on Android the platform to be difficult but rewarding. It has been established as the appropriate choice for the project system. Below in figure 9 a nexus 7 can be seen.



*Figure 9 - Android Nexus 7*

It is important to respect that although Android was chosen for the project system, an Apple iPad is considered the more suitable physical hardware for children with disabilities. This is

due to Apples attention to design in UI, such as a physical home button. (Taylor, 2013) However, due to the accessibility of the development platform, and the past experience with Android application development, this will be the chosen platform. This justification was deemed acceptable by both the project supervisor and Orlagh.

## 2.4.2 Mobile Development Platform

For creating the animations and gaming elements of the application, a graphics library will need to be used. This will need to be flexible and integrate well with the applications purpose. To accomplish this, either Unity or OpenGL will be used.

### 2.4.2.1. Android and OpenGL

Androids OpenGL ES library allows 2D and 3D graphics to be rendered. This would integrate well with the Android base application, and all the connected layers. It would allow the use of animations and interactive games, but comes at a price of being more complex than Unity3D.

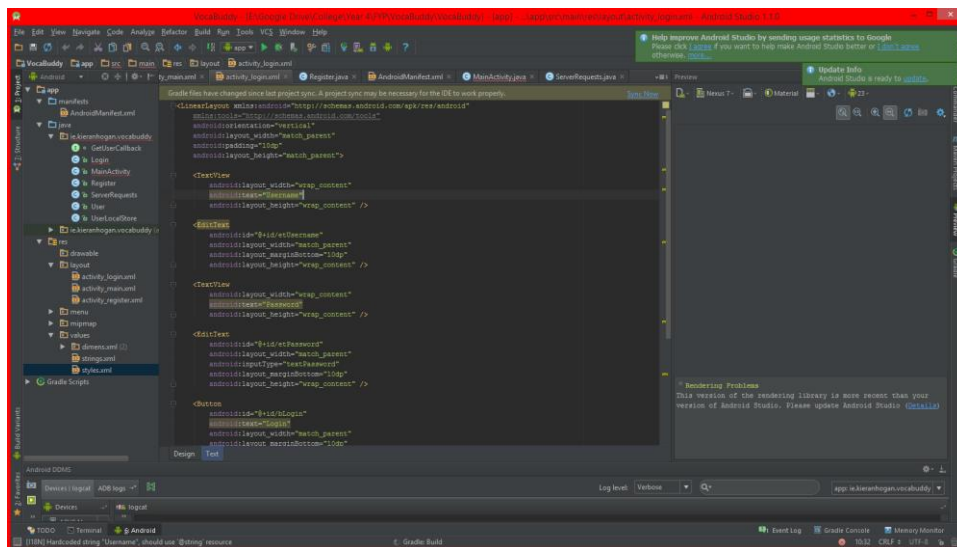


Figure 10 - Android Studio

In figure 10 above, the Android Studio IDE is shown. The project system will likely require rigorous animation and visual manipulation. The developers lack of experience using Open GL with Android, likely means he will seek an alternative. (Android, 2015)

### 2.4.2.2. Unity

Unity engine is a comprehensive game engine that allows game logic to be written in C# scripts or JavaScript. It allows games to be created in a complex 2D or 3D engine, and is virtually limitless when it comes to possibilities. Having used it before, using this engine would be quite easy for the developer. However, it is not apparent how easily Unity would interact with the other architectural layers in the system. The editor can be seen in figure 11.

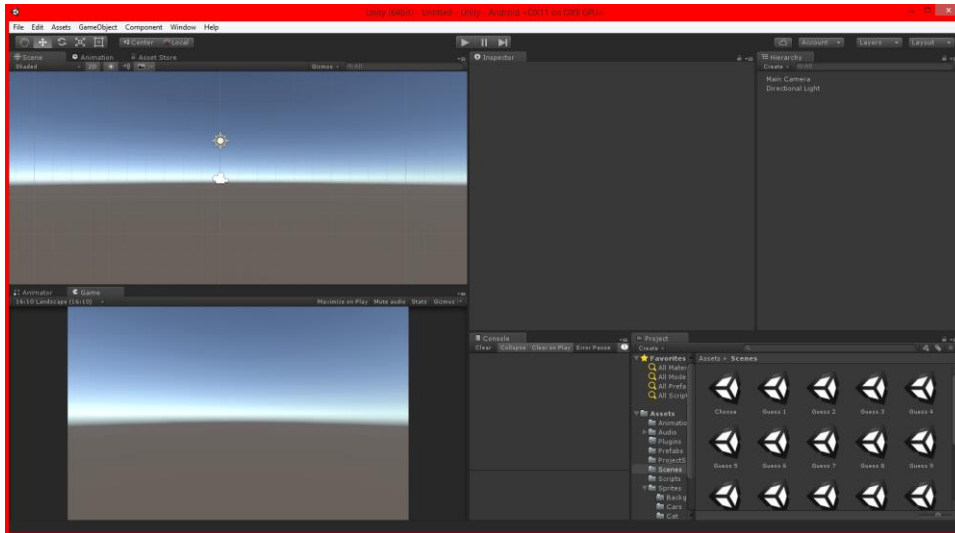


Figure 11 - Unity3D Editor

Another hugely beneficial factor about Unity as a development platform, is that it is multi-platform. This means that the application can be developed and the deployed to a huge variety of devices. (Unity - Game Engine, 2015) This could potentially be very beneficial to future work on this project, as it would allow the application to be brought to IOS. As a result, Unity was chosen as the development environment to create the base application for the project.

### 2.4.3. Web Hosting

For the system a cloud hosting of the server and database will be required. This will be done by creating an Ubuntu virtual machine which runs an Apache server and MySQL database. There are many options out there, but two of the more prominent service providers that stood out are Amazon AWS, and Microsoft Azure. Both have free plans but required careful consideration, as the database storage is an important element in the system (M, 2015).

#### 2.4.3.1. Amazon AWS

Amazon AWS is one of the most popular choices for cloud services in the world. They provide the biggest range of services, from Networking and Storage, to Deployment and Database services. It was found that Amazon offers a much more substantial free hosting plan than other competitors like Google Cloud. This plan is for one year and offers 100GB of database storage and low tier access speeds for free, which is more than enough for the project requirements (Colangelo, 2014). However the Amazon interface can be overwhelming and confusing, as it offers very deep customisability and security features.

#### 2.4.3.2. Microsoft Azure

Microsoft Azure is another competitor in the growing industry of cloud hosting. It offers a number of services from analytics to data storage. For the project, only basic VM hosting is required, and Azure offers very similar prices to Amazon. The Azure interface and virtual machine management is however much more streamlined and user friendly than Amazons. It makes managing your servers and scaling them much easier. Therefore, Microsoft Azure virtual machine hosting is a good option for the project. (Microsoft, 2016)

#### 2.4.4. Web Server technology

For the server side of the application, the Unity application and the web application need to successfully access and connect to the database, where the user's progress data will be stored. This will require the use of a language or technology to handle database requests. For this many languages could be used, but the options have been narrowed down to two of the most popular choices, with these being either Node.js or PHP.

##### 2.4.4.1. Node.js

Node is a Google owned JavaScript library primarily used for handling client side requests for connected databases. Node also pays close attention to the Model View Controller pattern, and allows the user to keep these layers separate on the server. Node is relatively new but is quickly rising in popularity. It has many benefits over other server technologies, and when used correctly can result in a faster more efficient system. (Wayner, InfoWorld, and 12, 2015) Having never used Node.js, and judging by its seemingly steep learning curve, the developer will likely use an alternative like PHP.

##### 2.4.4.2. PHP

PHP is an older language that serves the same purpose as Node.js, to handle the client side requests for a database. It is quite minimal in implementation, and allows developers to quickly code with ease without the need for other tools. It is also quite a good match for SQL languages like MySQL, which will likely be used. PHP has been around for years, and is still quite popular today. It has a large community and many resources available, making it an attractive choice for this project system. (Krill, 2014). The developer has some relevant experience of using PHP on the server layer from second year modules, and as a result of this and other reasons, it will be used. However due to the increasing popularity of Node.js, implementing it as an alternative to PHP will be considered, to see if it will improve the system.

#### 2.4.5. Web Application

Paired with this web server language, an appropriate web application framework or language would also need to be chosen. There are many options out there, including AngularJS.

##### 2.4.5.1 AngularJS

This is another JavaScript based framework has risen in popularity over recent years. It is now considered one of the best to use, as it can achieve everything that jQuery can, and much more. These additional features include RESTful API compatibility and form validation. It also supports the use of MVC (Model View Controller) pattern in development. This allows the presentation, logic and data layers to remain independent of each other, and is considered to be good coding practice. (Wilken, Puigcerber, and Erickson, 2015). Having recently used AngularJS, the developer has found it to be quite a substantial and useful framework. However, it is quite complex to implement and as a result may not be used.

##### 2.4.5.2. PHP, HTML5 and Bootstrap

PHP is a versatile language. As well as using it for the server side of this project, it could easily be used along with HTML5 to create a user friendly web interface. It integrates seamlessly with HTML and can minimize the web applications size greatly compared to alternatives like AngularJS. This was chosen for the project, as it's the simplest and most appropriate solution to use for the web application.

In order to reduce work load on styling the web interface, Bootstrap was also used. Bootstrap utilises HTML, CSS, JavaScript and JQuery to quickly create web interfaces for applications. It is mobile friendly by design, and streamlines creation of web interfaces. (Otto, Thornton, and contributors, 2015)

#### 2.4.6. Database

The project will also require a database to store all of the user's progress information. Before choosing this, the various options were weighed up and considered. Potential hosts and cloud service providers should also be considered, as they may not support the chosen database language.

##### 2.4.6.1. NoSQL

It was first researched whether a relational or NoSQL database would be better suited for the project system. NoSQL generally allows data to be stored in documents, graphs, key values or columnars. They are good for performance and scalability, allowing a business to potentially grow rapidly and more easily. The way in which NoSQL can store its data differently, means it can be quite useful for arbitrary situations. (Coyle et al., 2014). The project system will only require the storage of basic progress data from the application, so many of these NoSQL features would not be utilised. Having no experience with NoSQL until very recently was also an important factor in deciding whether or not to use NoSQL for this project.

##### 2.4.6.2. SQL

Due to past experience with SQL, and the simplicity of the early concept data model, it was clear that SQL would be more than sufficient. For SQL, the many different flavours were researched, including Oracle, Microsoft SQL Server, PostgreSQL and MySQL. Oracle is quite a complex database system, and allows some features that others do not have. MySQL offers a much lighter experience, and is open source (ITX Design, 2015). Due to its popularity and expansive resources and documentation, MySQL would be a good option for the project system. It also naturally compatible with PHP as a server language, which is going to be used in the project system. PHP and MySQL will work well together and provide an appropriate backend for the application system.

#### 2.4.7. Data Visualisation

The supervisor element of the application, will require them being able to analyse the students' progress. For representing the data to the user on the web application, it is likely that a data visualisation framework or tool will be needed. This element of the project is not as urgent as others, and thus it has not been researched it as thoroughly. There are many frameworks and tools available including D3.js, Chart.js Google Charts, FusionCharts and more (Rahman, 2015). The tool required does not need many comprehensive features, just a means of displaying progress of the user.

##### 2.4.7.1. D3.js

D3 is an open source chart, graph and visualisation tool built with JavaScript. It is a very comprehensive data visualisation framework. It allows the developer to attach data to a Document Object Model, which can be used to then generate tables or charts. It allows charts

to be interactive and animated, however it does not come with a chart library. Instead all charts must be generated manually using the tools provided. (Bostock, 2015) This framework would allow the charts to be tailored exactly to a chosen design, but requires more work and time than other charting tools and frameworks.

#### *2.4.7.2. FusionCharts*

FusionCharts is an open source chart framework that integrates well with PHP. It uses JavaScript and the jQuery library to create charts retrieved from SQL queries. It is also a seamless framework that allows charts to be created without much difficulty. Due to these factors FusionCharts will be used for the web application in this project. (FusionCharts, 2015) This chart rendering tool will likely be easier to use than D3 and will provide enough of the required elements for this project. However, if the requirements of the project change, and extra features that D3 provides are required, this will implement that instead.



### 3. Design

Design was a core focal point in the project. When beginning the design phase, it was important to choose an appropriate design methodology.

#### 3.1. Approach and Methodology

During the course of this project, a combination of two methodologies was used. Due to the end user's specific requirements, and the user centric nature of the proposed system, user-centred design was used. This is a design philosophy that puts needs of the end user as the focus. However, aspects of scrum agile methodology was also used. Iterative design, implementation and testing phases were done, and there were periodic meetings with the project supervisor and end user.

##### 3.2.1. User Centred Design

User Centric Design is an approach to software design that involves developing systems tailored to the end user's needs. The systems are designed with the user's capabilities in mind, and are designed to work seamlessly to meet their requirements. The concept of UCD (user centred design) is broken into four stages; research, concept, design, and development. After each development stage a release is possible. This developed product can be critiqued by the user, and the cycle can either restart, or the application can be finalised and released. This cycle can be seen in figure 12 below.

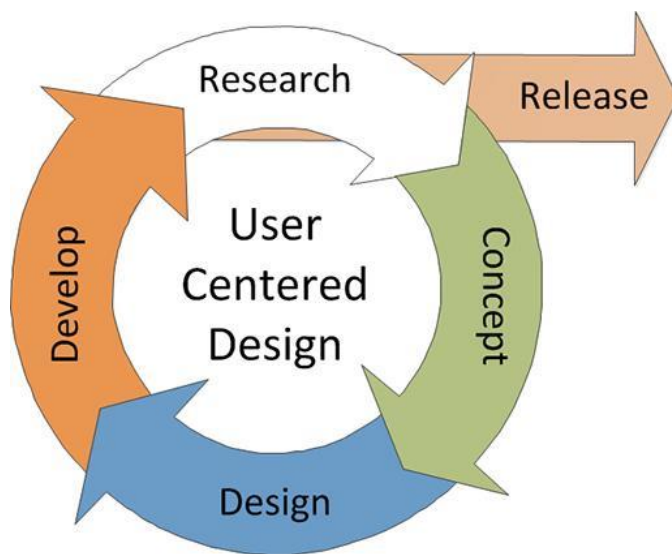


Figure 12 - User Centrid Design cycle

For the project at hand, tailoring the system to be usable by Sophia was a challenging prospect. Receiving user feedback throughout the course of this project allowed the system to iteratively go through the four stages. By iteratively redesigning and developing ideas and core features in the application, a user centric application becomes a reality.

User centric design, can also be implemented alongside other more well-known methodologies. Throughout this project, and adapted scrum project management methodology was also used.



### 3.2.2. Scrum

The Agile project development framework is a framework that focuses on the core of solving a large problem and satisfying the needs of the customer. The Scrum methodology was chosen because of its successful and prevalent use in industry, and also because it focuses attention on the customer and client needs. Agile based methodologies are typically aimed at teams, so may not seem appropriate. However, an adapted approach is being used here.

Scrum involves a team of people working in iterations, where they establish goals and work rapidly to complete these goals. These team members include project managers, developers, data engineers and QA testers. Scrum involves teams being split into roles, and the frequent contact between team-members. However, with this being a solo project, this is where it must be adapted. In this case, there will only be the developer assuming all technical roles, and the project supervisor holding an over watch position on project. The rapid bursts of work are called sprints, and these involve implementing a backlog of tasks quickly and effectively. A representation of scrum can be seen below in figure 13.



Figure 13- Scrum framework

Scrum also relies heavily on regular meetings with the team members. To achieve this, a meeting was organised every week with the project supervisor to discuss progress. As often as possible, meetings with Orlagh were also arranged. However, due to tight schedules and conflicting schedules, less of these were arranged than originally planned.

The adapted version of Scrum being used on this project involved the developer taking on all the roles in the development. The project was done using iterative development of features in small sprints, and repeated analysis of requirements throughout the project lifecycle. This regular revision of requirements was necessary to ensure the final product was catered accurately to the customer's needs. In this case Orlagh will be the customer, who gave feedback on the direction of the project.

Throughout the early stages of the project, and due to its user oriented design, repeated requirements and design phases were conducted. This involved a large research phase, followed by a requirements phase, followed by a high level design phase. However, as a result of the importance of the user driven design elements of this system, the research, design and development phases were done iteratively following UCD. Throughout these

stages, Orlagh was contacted and she gave valuable input into how she felt the system would best function. This was the most efficient way to achieve a usable design in the system.

Throughout the rest of the project, the development was being conducted in short rapid sprints of two weeks. Each of these aimed to implement particular features. To ensure that time was allocated accordingly, priority was given to more important features.

### 3.2. Project Analysis

There were two key deliverables established early on for the practical element of this project. The solution to this problem is broken down into two main client applications. The main user of the system is Sophia. She will be using a Unity tablet application. This allows her to partake in games and activities to improve her vocabulary. Her mother Orlagh, is the secondary user, and will be using the web application. This will allow her to obtain feedback on her daughter's progress. These two applications operate independently but are connected via the same information.

#### 3.2.1. Android Tablet Application

The solution for the addressing the symbol vocabulary learning is to approach helping Sophia in a fun and educational matter. Here the student will be helped by teaching them the meaning of specific symbols, so she can make use of AAC communication systems. The application will use a symbol set, and establish their meanings with animations and interactive game elements. The student will then have the option of selecting to play the vocabulary games, or view the vocabulary library.

The core learning challenge involves the student seeing an animation. The animation represents a key word in the scenario. The scenario is described by a symbol sentence, with there being 2 potential symbols at one part. Completing the challenge involves the user pressing each symbol, and choosing the correct symbol from those two. Every symbol press speaks the word aloud. Upon creating the sentence and selecting the correct symbol, the screen will receive an on screen medal and an applause to let the user know they picked the correct symbol. If they pick the wrong symbol the application will flag this and this will be stored.

For example, there is a character running on the screen. The symbol sentence on screen would be "the dog is \_\_\_\_", and in the position of the blank, two symbols are present, giving the Sophia a choice. These symbols could be running and jumping. In order to complete the challenge, the user needs to press each word and the correct word in the choice slot.

After successfully completing one of these rounds, the user is given an interactive game to play, related to the word challenge created. This interactive game will be a platforming game which would be controlled using left and right direction symbols. Each of these games is also tailored specifically to the end user. The platform games involve the character collecting coins, collecting sweets, playing hide and seek, and chasing.

The user will also have the option of viewing symbols in the library. First they will have to choose a category represented by a symbol. These symbol categories will for instance include, movement, activities and games. Once a category is chosen, they will be able to view animations for all of the words in that category. Both of these app functions are integral to addressing the main objective.

### 3.2.2. Web Application

Another important aspect of the system being developed, is a way for the user's supervisor to view this individual user's progress. The solution for this aspect of the system involves the development of a web interface application, which accesses the user's data. Therefore this system will include a login, using the same credentials as being used on the Android application.

Once the user has been logged in, they have the option to displays graphs and information on where their student needs improving, or where they are excelling. This will help the supervisor to help the user with manual training on specific sets of symbols. Feedback from this system is vital to satisfying one of the other core objectives.

The supervisor will also be able to view additional information on the student, as well as user documentation and help for the Android application. There will be an area to allow the user to configure any potential settings, which are yet to be established and decided.

Another potential aspect of the web application would be to allow the supervisor to upload their own symbols. By doing this, a notification would be sent to allow the admin of the system to create and upload animations for these symbols. In doing this, the symbols would need to be stored online. This is a low priority requirement, and will be implemented if there is time.

### 3.2.3. Established Requirements

Based on the extensive research conducted, an analysis of requirements for the user was vital. The core focus of the project will be tailoring the system to the specific user group. This will involve following strict heuristics and user elements, and paying close attention to the user's needs. In the research stages of this project, elicitation has been completed with a potential client of the software, in order to establish early on in the project stages what the requirements are. Based on these early requirement stages it was clear that there would need to be two main interfaces in the system.

Each of these two application layers have specific requirements that were derived from the analysis. These evolved somewhat over the revisited design phases, and changed slightly from user feedback during implementation. They are ranked in order of priority ranging from high to low, and it's important to note that not all of these we're achieved during the project implementation. The original specified requirements are listed below in the table figure 14.

Name	Description	Priority
Android Unity App		
Vocab Learning Games	This game will challenge the student to choose the correct symbol from a list for a given animation.	HIGH
Interactive Symbol Games	This game will follow the vocab learning, and will allow the student to use the symbols learned to control.	HIGH
Word Library	This will allow the user to view the library of all animations in the games, and will be categorised.	HIGH
Configure settings	Allow configure of any potential system settings.	MEDIUM
Reset progress	Allow the supervisor to reset the students' progress.	LOW

Web App		
User Login	Have the supervisor log in to the system so they can access their web application.	HIGH
Student progress	Show the supervisor the students' progress with graphs and charts, and where they are excelling and struggling.	HIGH
Student info	Allow the supervisor to view any additional information about the student.	MEDIUM
App documentation	Allow the supervisor to view documentation on the apps functionality, and see instructions and help.	MEDIUM
Upload symbols	Allow the user to upload their own symbols to the system, which animations could then be created for.	LOW
General System Reqs.		
Store user login online	Login information stored online in database, universally usable for both the Android and Web apps.	HIGH
Store user tracked information online	Level names, categories, time played and score should be stored in the database.	HIGH
Store symbols online	Links for the symbols should be stored online, allowing updated ones to be pushed by the application.	LOW
Store animations online	Links for the animations should be stored online, allowing updated ones to be pushed by the application.	LOW

Figure 14 –User Requirements Table

### 3.2.4. Unity App Storyboard

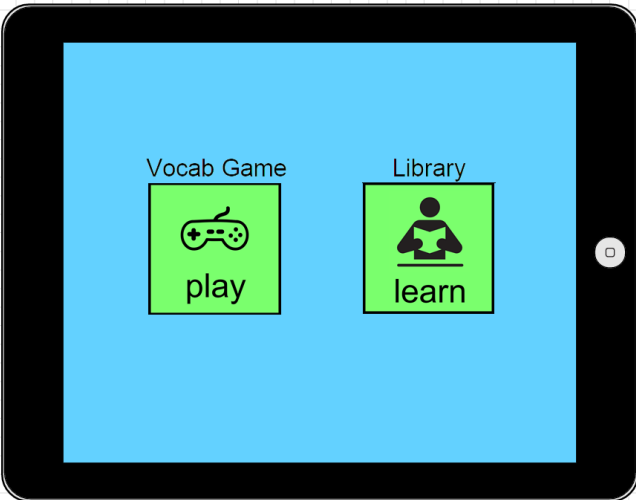
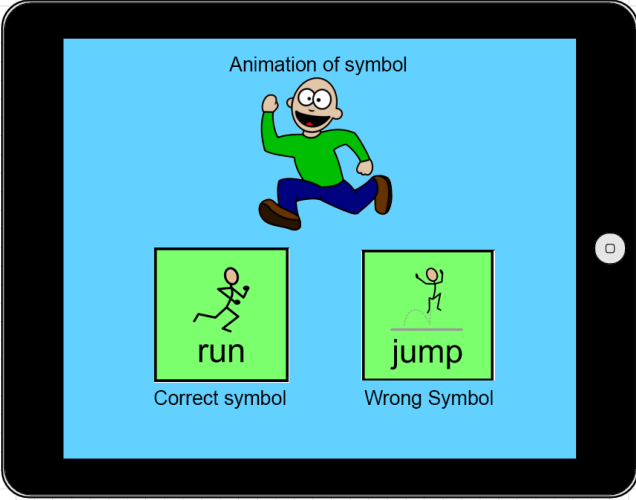
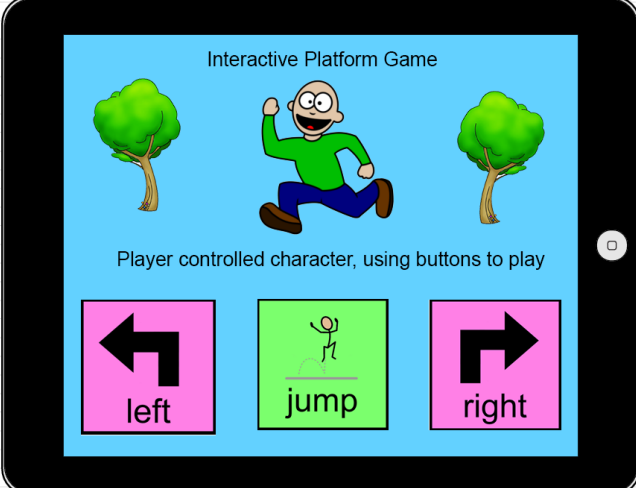
The following storyboard in figure 14 was used early on in design to describe Sophia's step by step interaction with the system. There are two main use scenarios for this user which is represented on these storyboards. These are playing the vocabulary games, and accessing the library of symbols and animations.

A hugely important factor in designing these storyboards is that they have the minimum steps necessary for our particular user to complete the use cases. Therefore, the systems main menu is very simple and minimal in design. The use of bright colours and animations result in the system to be aesthetically pleasing and appealing to the user.

The system also uses symbols for all of the icons. These symbols will come from the user's already existing library of symbols, which they should understand with use. These will be verified and established during test phases.

There is also navigation buttons represented by universally known back and home buttons implemented in the system. These were left out of these storyboards however, as these were very basic demonstrations of the application use. The later prototypes implemented these buttons, as did the finished project. A settings button will also be present throughout the final implementation, which allows certain aspects like icon size to be adjusted.

The following storyboard, figure 15, is a very basic initial demo of the core of the Unity tablet application. This storyboard was used as a base concept for the application, and evolved greatly throughout the iterative design phases. Each screen has a brief explanation.

	<p>A) The user starts the application and is given the choice of going to the vocab game, or the library (after supervisor logs them in, only needs to be done once).</p> <p>or</p> <p>B) The user from the main menu can also click the learn button to access the library.</p> <p>Many changes occurred here, including having level select.</p>
	<p>A) 1. When choosing the vocab game, they are brought to the game screen, this will allow them to play the correct symbol game. Here they must pick the right symbol for the animation.</p> <p>From Orlagh's feedback, she felt having sentence structure here would be beneficial. So the final design used far more symbols to create a full sentence.</p>
	<p>A) 2. Then upon completing each of these rounds, the user will be brought to an interactive game related to the previous animation. These levels have objectives, such as collecting sweets. Upon completion of this level, the app brings the user to another word/animation challenge.</p> <p>More variety between levels was introduced in the final implementation of this application.</p>

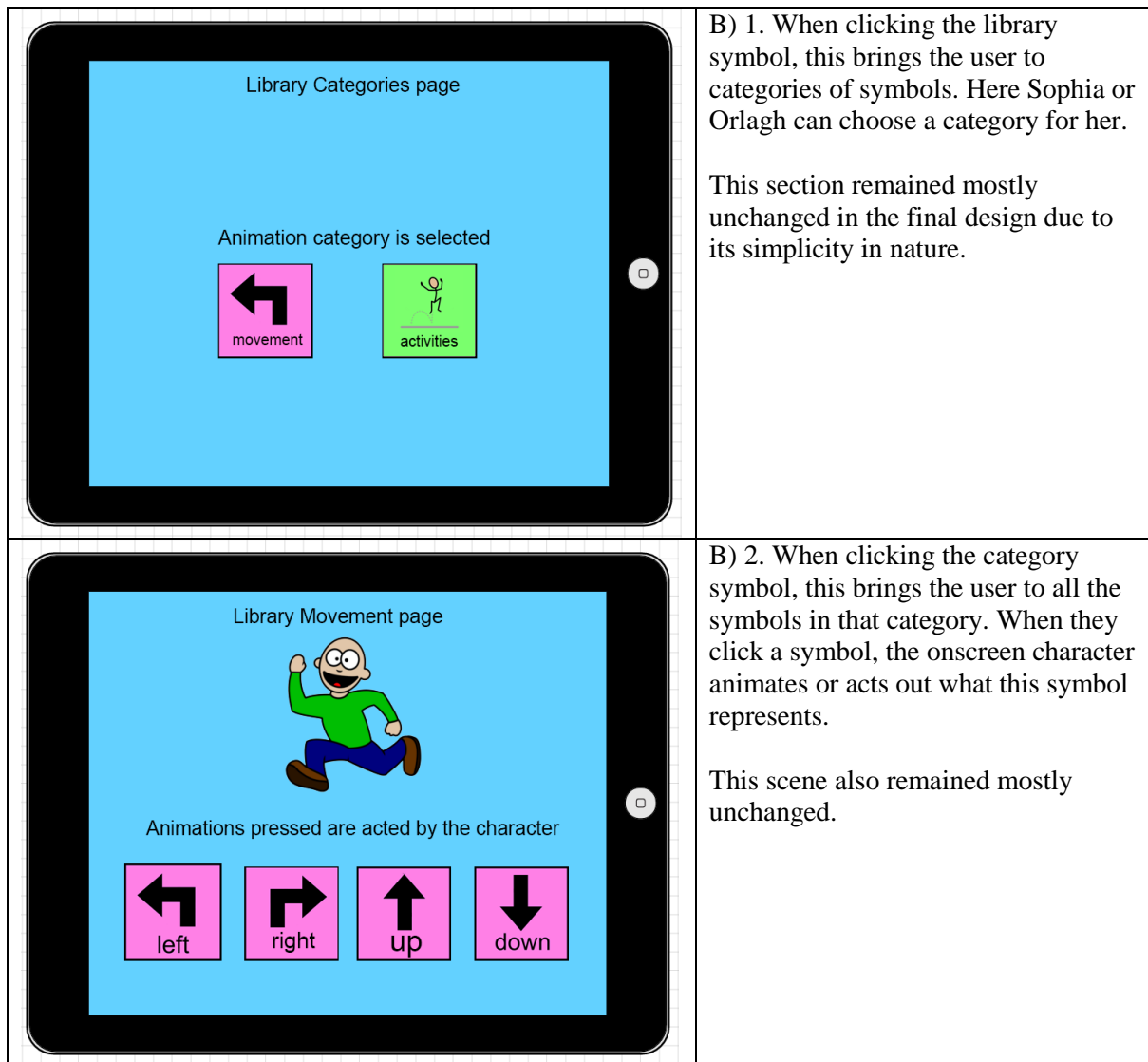


Figure 15 – Tablet App Storyboard

### 3.3. Project components

There are four main component layers in the project system. The Unity user interface, the backend Unity and server connection, the web interface, and the database.

#### 3.2.1. Unity - UI

This layer is designed for Sophia's use, and is tailored to her requirements. This involved designing low fidelity designs like the initial storyboard, and eventually high fidelity interactive designs. When working on this layer, the user centric design (UCD) philosophy was used. Regular feedback from Orlagh helped to iteratively redesign the interface for the target end user.

Various symbol sets were experimented with during this phase, until SymbolStix was chosen as the symbol set to use. These symbols are aesthetically pleasing and easy to understand. The background for these symbols was chosen using the Fitzgerald key. This distinctively separates word groups by colours. Yellow being for people and pronouns, orange for nouns, green for verbs, blue for descriptions, pink for social, and white for miscellaneous. The key is seen below in figure 16.



Figure 16- Fitzgerald Key

Originally the design was very basic, but many aspects were considered inappropriate. Apples UI design guidelines were consulted when deciding on default icon size and layouts. Apple are considered one of the best resources for accessibility design guidelines (Apple, 2016) This modified layout saw the home button being move to the bottom right of the screen, and buttons being more spaced out. The background colour was also carefully chosen to be a calm blue colour.

From UCD, it was decided that customisability would be a valuable feature to include in the system. This involved designing a settings menu where the background colour and button size could be adjusted and tailored for the user. This settings menu was designed to be out of reach of the target user, so requires a hidden button to be pressed in combination with another.

Another major design choice at a point in the project, was to keep a consistency through the different levels and animations. Rather than having different characters animations for each level, a character was created. Initially this was a very basic character sprite (2d character animation), however a website with free resources was found that had many characters. These different characters were examined, and a cat and a dog sprite were found. Feedback from Orlagh lead to choosing the dog sprite for the application. (GameArt2D.com, 2016)

Throughout this application, the dog would be the main character. The dog would be animated doing activities during word challenges, and would also be controlled by Sophia in the more interactive levels. By keeping a consistent character throughout the application, it is hoped that Sophia would develop a bond with the character and enjoy interacting with them.

Feedback was incredibly valuable during the design of this application layer. Below in figure 17 an early prototype of the menu screen can be seen. Here, a definitive symbol set had not been decided. Due to the lack of good free symbol sets, custom ones were created. These can be seen in Appendix 1.



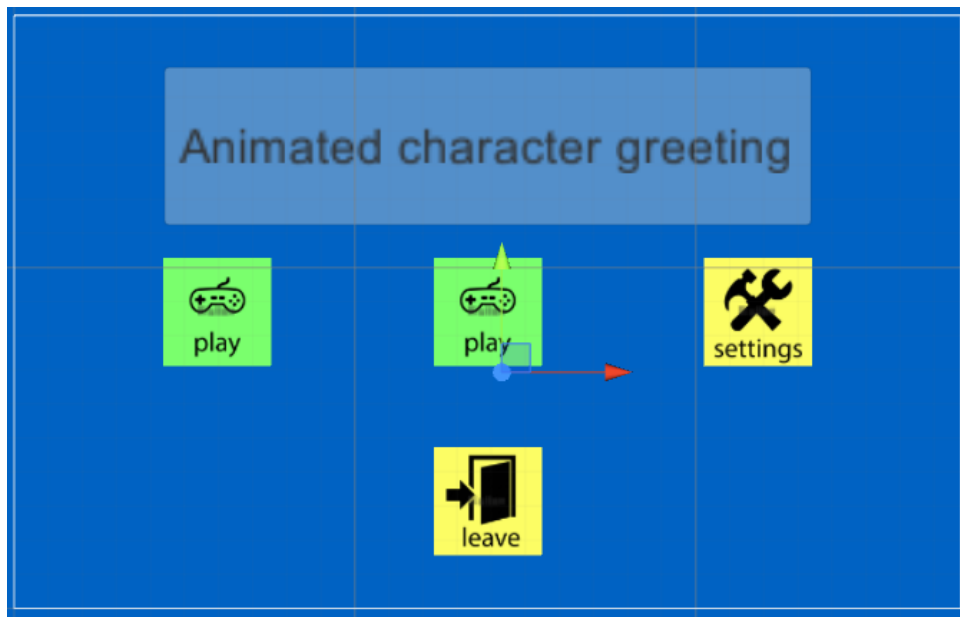


Figure 17 - Prototype menu screen

### 3.2.2. Backend Unity and PHP

This layer involves connecting the unity tablet application to PHP files which control database access. A PHP file is created on the server, which when called by the Unity application, makes a connection. When connected, it accepts progress data through form inputs.

This data is then processed by the PHP files and inserted into the database. Once this data is stored it allows the web interface to access the data and output it to charts. This is the most complex aspect of the project system.

### 3.2.3. Web application - PHP, HTML5, Bootstrap and FusionCharts – UI

The PHP layer controls both database access, and the front end for the web application user, Orlagh. Navigating to the web application brings the user to a login. Here they must enter their credentials. By doing so they are granted access to a very basic bootstrap interface. Here she can cycle between pages which are otherwise blocked off from the user.

The fusion charts PHP library is a JavaScript based library used for implementing charts via PHP. Each of these pages will implement a different chart, to show Orlagh how Sophia is progressing in the application. These charts need not be complicated in design, as they are only representing basic data.

### 3.2.4. MySQL – database

The database layer of the system is very simplistic in design. The data from the Unity application is sent to the PHP server via TCP which then inserts the data into the MySQL database tables. For this database, only two tables exist. The user table and the tacking table.

The user table contains auto incrementing user ids, usernames, and passwords. Due to the nature of the system, only one user will ever be in this table, however more can be added if necessary. Passwords are stored using md5 hashing, so as to ensure an appropriate level of security.



The second table is the tracking table. This is a large table containing all the data from the application. This data consists of the auto incremented tracking id, the level name, the category of the level, the time spent on the level, and the users score. Every round of the app creates a row in this table, which adds another marker to the charts generated for the web user interface.

### 3.4. Features and use cases

The core feature of this system are to have a fun, interactive tablet based application that aids the user in understanding vocabulary using symbols. The second core feature of this system is to have a web application that allows the target users guardian to examine the users progress. These two features can be broken down using the following use cases.

There are two sections to the system use cases, one for each application. Orlagh and Sophia are users for the Unity app, and Orlagh is the only user for the web app.

#### 3.4.1. Unity App Use Cases

##### 3.4.1.1. *Sophia*

Sophia is the primary user in Android tablet system. This system will be launched by her or her mother Orlagh by clicking the app icon, which brings her to the menu. From here an interface using symbols instead of purely word represented buttons will be presented. The user can continue learning by pressing the play button. This will bring Sophia back to the last level she left off in the application. If it is her first time, this will launch from the first level.

From the menu she will also have the option of choosing particular levels which she may particularly enjoy. By clicking the “choose level” button she is brought to a screen where she can pick from list of levels. Within the different levels of the games, tutorials will be present in the form of a “how to” button.

From the menu Sophia can also access the Animation library. By clicking the library button, she is brought to a page of categories in the library. From here, she can pick a category and navigate to a page where she can click on symbols and see the animation which explains the symbol.

If Sophia is confused from the menu itself, there is a “how to” button which shows what each button does. The full use case diagram can be seen below in Figure 18.

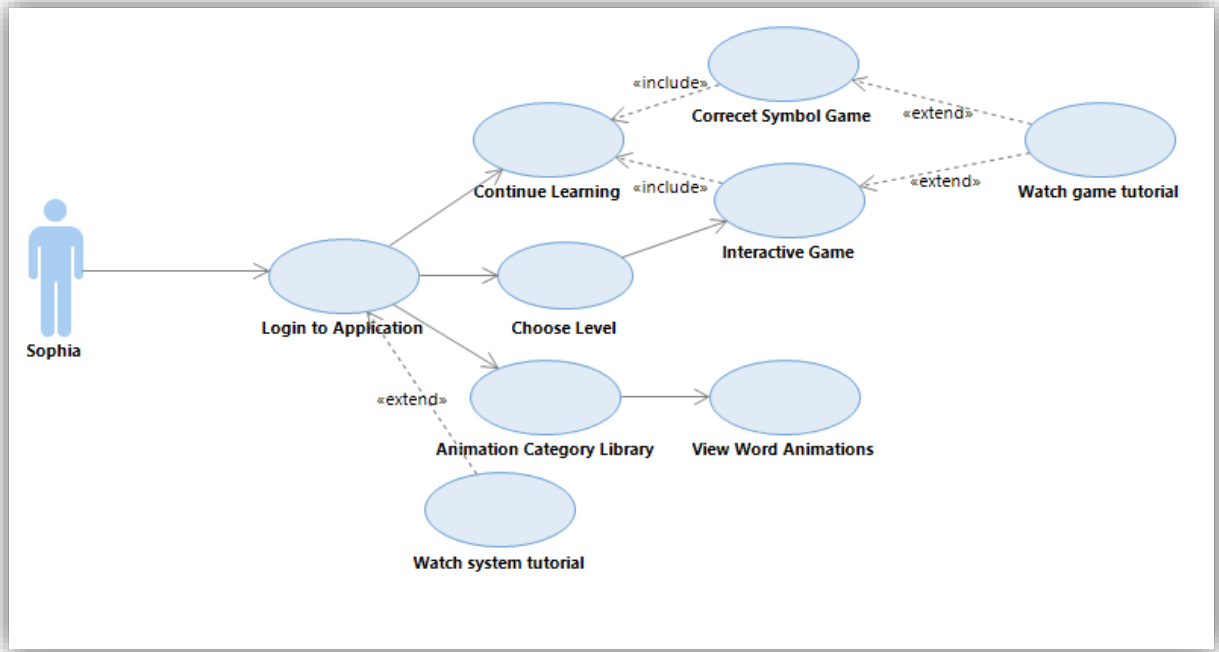


Figure 18- Unity App: Sophia's Use Cases

### 3.4.1.2. Orlagh

Orlagh is the secondary user of the Unity application. Her only use case is for modifying settings in the application to give Sophia the customised experience she may require. Once the application is launched, the menu screen will have a secure settings button in the corner. Once this is pressed, the settings button will appear, and pressing that will open the settings. This minor blockade prevents Sophia from accidentally accessing and changing the settings.

From here Orlagh can change the icon size, the background colour and reset the user's current progress. The full use case diagram can be seen below in Figure 19.

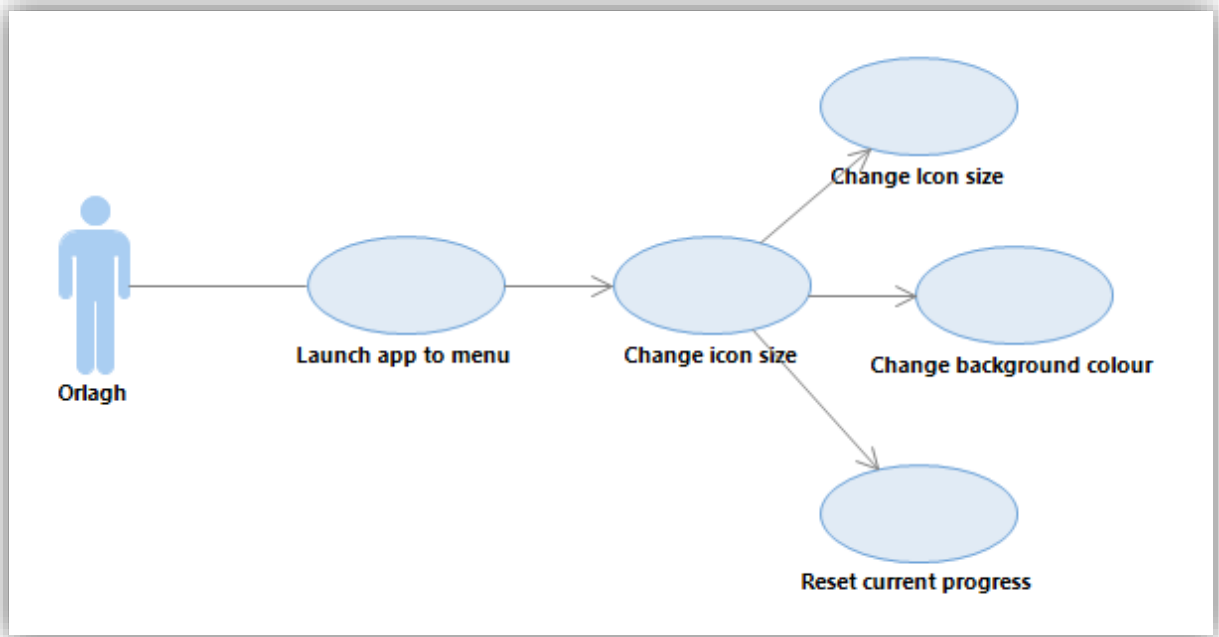


Figure 19 - Unity App: Orlagh's Use Cases

### 3.4.2. Web App Use Case

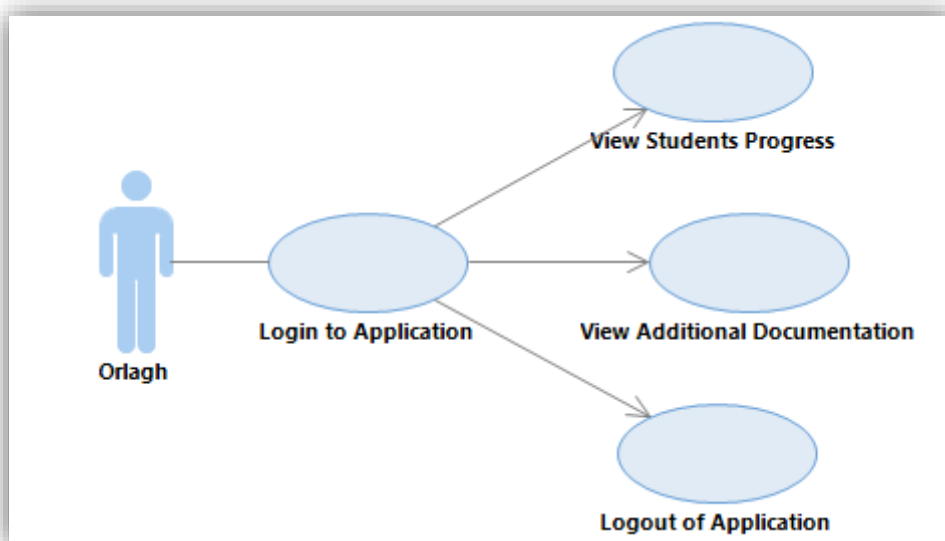


Figure 20- Web App: Orlagh's Use Cases

Orlagh is the only user for this application. She uses this application by navigating to the URL and logging in with her credentials. On successful login, she is brought to a home screen, and from here she can navigate to multiple other pages, where she can view the students' progress. This progress will be displayed as graphs and charts that show where the student is struggling or excelling. This will give Orlagh a good indication of where to manually tutor the Sophia, and give her more targeted help. This can be seen in figure 20.

## 4. Implementation

This stage marked the beginning of high level prototyping and development. Here the high level system architecture is explained. This is followed by a breakdown of each component layer of this architecture, and the code that drives the system and how it works.

### 4.1. System Architecture

Architectural design models are integral to showing how an applications layers are separated and how they connect. For the application system, a very 3 tier model is being used. These three tiers are the client, server and database tiers. This model is a fundamental networking model that has represented applications since the early World Wide Web.

In this system, the tablet application will be used by Sophia. This application is being developed in Unity and deployed to an Android tablet. The Web application will be used by Orlagh. This will be accessible through most popular web browsers, including those on mobile devices. These are considered client tier applications. Sophia's application sends progress data through PHP to be stored on the server. Orlagh's web application will be retrieving data from the database via PHP.

Microsoft Azure will be hosting the server and database layers. The PHP server will be running on a virtual machine Ubuntu server. This will handle incoming client requests from the two client applications. These requests will then be processed and translated into queries for the database.

The database will be a MySQL database running on the same virtual machine as the server layers. This will reduce unnecessary delays when querying the database. This database will store the user's login information, as well as the student user's progress in the application. This system architecture can be examined below in Figure 21.

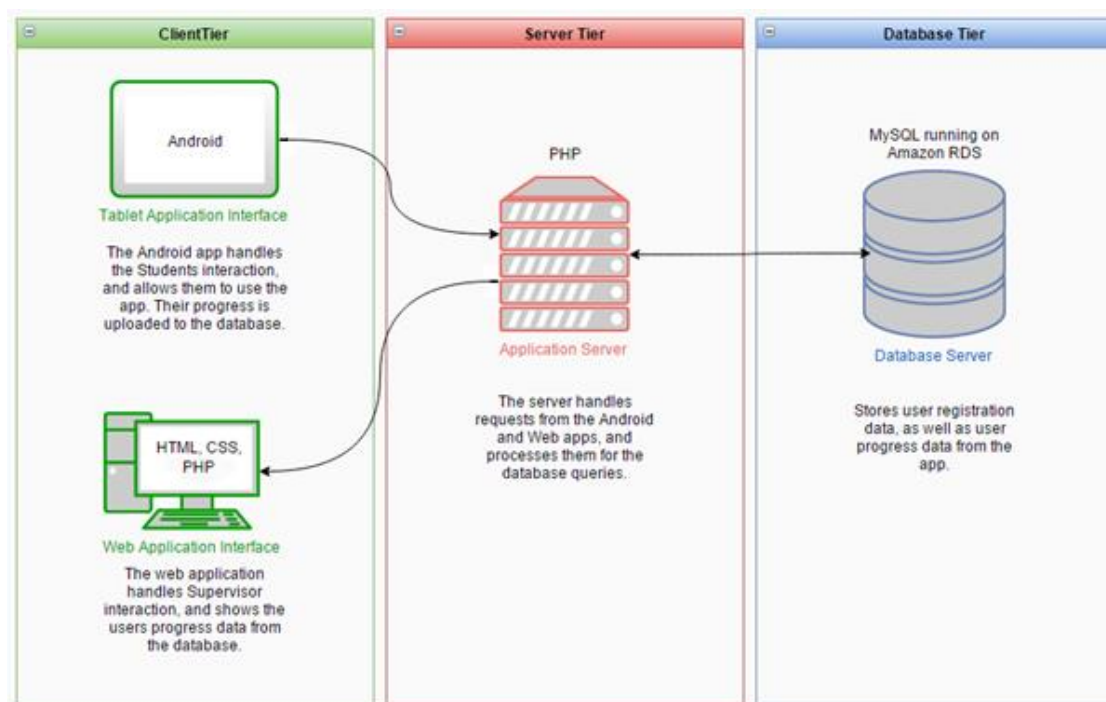


Figure 21 - System Architectural Model

## 4.2. Component Development and problems encountered

Development began with setting up the required development environments for the technologies being used. Firstly, Unity 5 was downloaded, along with the most recent versions of the java JDK and JREs. When installing Unity with the Android build features, the JDK needed to be specified. The workspace was created in a project folder, which was then added to a GitHub repository. This will ensure that the project files are version controlled, and backed up. This same system was set up on the development laptop and desktop, allowing development to be done in the DIT library and at home.

The next important stage that was setting up the server and database on Microsoft Azure. A virtual machine was created here with Ubuntu server 14.04 as the operating system. Ports and permissions were also configured here. Once this was created and set up, Apache, MySQL and phpMyAdmin were all installed.

From here a basic vertical prototype was built, to prove that all layers could connect and communicate. This involved Unity accessing data via PHP hosted on this server. After researching into connecting these two technologies, a vertical prototype was built successfully. Specific design implementation for each component followed this.

### 4.2.1. Unity application

Developing the Unity application was the largest section of the development process for this project. It involved the creation of many scripts and the manipulation of objects and scenes in the Unity editor. It is important to note, that levels are called scenes in the Unity application and may be referred to as such. All of the SymbolStix symbols that were required were downloaded from the n2y website, and background colours were added to these symbols following the Fitzgerald key, which Orlagh recommended. These are the different scenes within the application and how the different scripts and combinations of scripts drive their functionality.

#### 4.2.1.1. Main Menu

The main menu greets the user after the application is opened. This serves as the app's home page. It presents a very user friendly UI, using the symbols as buttons. This is where the user is also first introduced to the in-game character. This character is with the user throughout the entire application, and all of the activities. In this scene, he just simply in an idle animation state. The screen can be seen in figure 22.

From the main menu, there are three main buttons, and two others. The three main buttons are play, choose and learn. The play button continues the application from the current level of the user's local progress. The choose button brings the user to the level selection scene, and the learn symbol brings the user to the library of animations.

There is also two secondary buttons in the scene, a "how to" button that when pressed plays a specified short video short on using the system. This would help the user if they were confused. The last secondary button on the screen is the quit button which closes the application. There is also a settings button in the top right corner. This is a button that when

pressed, causes another button to appear. This is to prevent Sophia from accidentally accessing it.

A connection test is also completed in this scene, and presented on screen in test. This is done to inform Orlagh whether the game tracking will be logged or not.

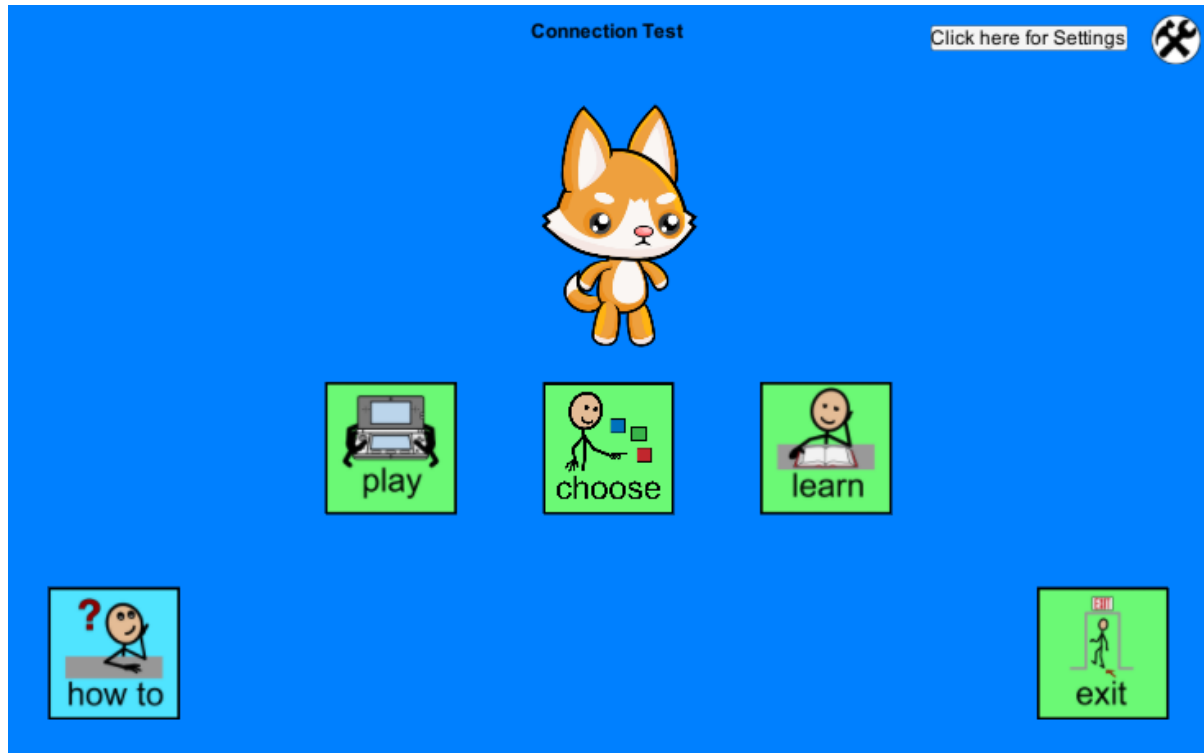


Figure 22- Unity App: Main Menu

### Menu.cs

This class controls the core functionality of the main menu. This includes the buttons which navigate to the various scenes. These buttons are assigned functions. These functions use the SceneManager.LoadScene() function which navigate to specified scenes. For handling the play button, the script checks if there is a current level value in the local data PlayerPrefs file. If there isn't, it assigns the LoadScene() to level 1. See below in figure 23.

```
public void LoadCurrent()
{
    SceneManager.LoadScene(PlayerPrefs.GetInt("CurrentLevel"));
}

public void ExitGame()
{
    Application.Quit();
}

public void LoadHelp()
{
    Handheld.PlayFullScreenMovie("test.mp4", Color.black, FullScreenMovieControlMode.CancelOnInput);
}
```

Figure 23 - Menu.cs - Scene Management

PlayerPrefs is an application file that is capable of storing variables in a data file. This file is persistent and is stored even after the application is closed. The tutorial button calls an android only function to play a video file in full screen. This will only work on the device, and not in the editor while debugging. Some of these functions can be seen above in figure 19.

### CheckConnection.cs

This class does a basic check for a network connection and displays it on the screen. It does this using what is called a coroutine. A coroutine in Unity is a function where the execution does not need to complete to continue through the rest of the application. Instead it allows the use of time elements to dictate pauses.

```
void Start () {  
    connectionStatus = GetComponent<Text>();  
    StartCoroutine(CheckConn());  
}
```

Figure 24 - CheckConnection.cs – Start

Coroutines are initiated in functions using StartCorotuine(name) as seen in figure 24. As seen below in figure 26, we see that the coroutine attempts to ping an IP. While it is doing so it repeatedly checks until a response or definite negative is received.

```
private IEnumerator CheckConn()  
{  
    const float timeout = 10f;  
    float startTime = Time.timeSinceLevelLoad;  
    var ping = new Ping("172.217.18.206");  
  
    while (true)  
    {  
        connectionStatus.text = "Checking network...";  
        if (ping.isDone)  
        {  
            connectionStatus.text = "Network available.";  
            yield break;  
        }  
        if (Time.timeSinceLevelLoad - startTime > timeout)  
        {  
            connectionStatus.text = "No Network available.";  
            yield break;  
        }  
  
        yield return new WaitForEndOfFrame();  
    }  
}
```

Figure 25 - CheckConnection.cs - Coroutine



### BGButtonSizeController.cs

This script is called in every scene. It runs at the beginning of every scene, and pulls the background colour information and button size information from PlayerPrefs. These are stored in the separate R, G and B float variables, and X and Y scale dimensions. It then applies these to the current scene. This is the most efficient way to ensure that the user chosen settings effect all scenes. This can be seen in figure 26 below.

```
void Start () {
    buttons = GameObject.FindGameObjectsWithTag("Button");

    if (PlayerPrefs.HasKey("bgR"))
    {
        Color color0 = new Color(PlayerPrefs.GetFloat("bgR"), PlayerPrefs.GetFloat("bgG"), PlayerPrefs.GetFloat("bgB"), 1);
        cam.backgroundColor = color0;
    }

    if (PlayerPrefs.HasKey("btX"))
    {
        foreach (GameObject button in buttons)
        {
            button.transform.localScale = new Vector3(PlayerPrefs.GetFloat("btX"), PlayerPrefs.GetFloat("btY"), 1);
        }
    }
}
```

Figure 26 - BgButtonSizeController.cs

### SettingsController.cs

This class handles the hidden settings button. In order to access the settings, the user must press the settings icon (hammer and wrench) in the top right corner. When this is pressed, another button appears. Clicking on this second button opens the settings screen via the SceneManager. This was quite simple to implement in code, and required the second button object to be hidden unless it was pressed.

#### 4.2.1.2. Settings

Accessing the settings menu can be done in any scene. It is done using a hidden button. This hidden button doesn't appear until the settings symbol in the top right corner is pressed, essentially requiring two buttons to be pressed. This is to prevent Sophia from accidentally changing settings during use.

In figure 27 below, the fully expanded settings menu can be seen. This settings menu has two dropdowns. One allows the scale of all buttons size, and the other allows background colour to be changed. When these are selected, they are saved to PlayerPrefs variables. All other scenes load these settings from the PlayerPrefs using the BGButtonSizeController.cs.

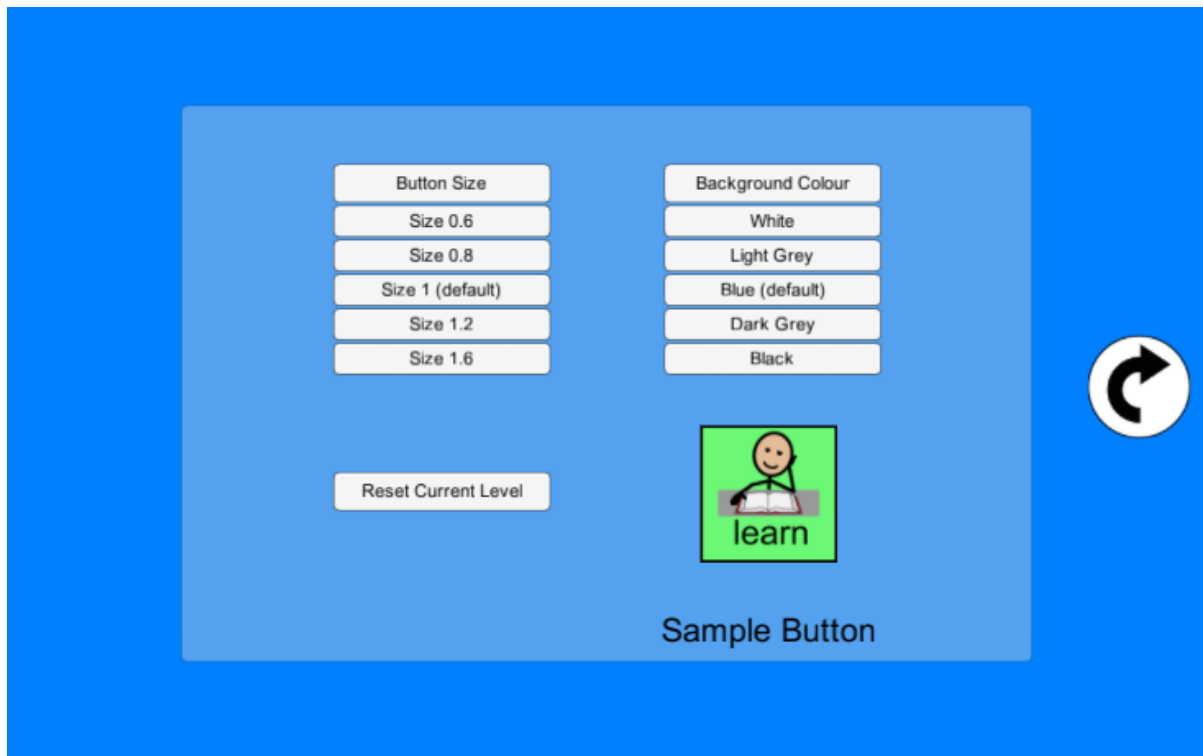


Figure 27 - Unity App – Settings

### Dropdown.cs

This class controls the dropdown menus in the settings screen. It manages the position of buttons from the dropdown controller to the scene, and displays them in a cascading dropdown menu. First a container is created, and all the buttons are placed on top of each other. Then by using a vector and modifying its y scale when hovering over the container, the buttons become revealed and the state is changed to open. This can be seen in Figure 28.

```
void Start()
{
    container = transform.FindChild("Container").GetComponent<RectTransform>();
    isOpen = false;
}

// Update is called once per frame
void Update()
{
    Vector3 scale = container.localScale;
    scale.y = Mathf.Lerp(scale.y, isOpen ? 1 : 0, Time.deltaTime * 12);
    container.localScale = scale;
}

public void OnPointerEnter(PointerEventData eventData)
{
    isOpen = true;
}

public void OnPointerExit(PointerEventData eventData)
{
    isOpen = false;
}
```

Figure 28 - Dropdown.cs

### DropdownController.cs

This script assigns each button in the dropdown menus to the desired settings change in the system. For assigning background colours in Unity camera views, RGB colour assigning is used. Therefore each button in the dropdown assigns an R, G and B value to give the desired background colour for the buttons. Each of these is saved to a variable name in PlayerPrefs. See figure 29 below.

```
public void Drop0()  
{  
    Color color0 = new Color(1, 1, 1, 1);  
    cam.backgroundColor = color0;  
    PlayerPrefs.SetFloat("bgR", 1);  
    PlayerPrefs.SetFloat("bgG", 1);  
    PlayerPrefs.SetFloat("bgB", 1);  
    PlayerPrefs.Save();  
}
```

Figure 29 - DropdownController.cs

#### 4.2.1.3. Choose (level select)

This scene allows the user to select the level to play. Giving the user this opportunity, allows them to easily access their favourite game, without having to play through the entire application. It is a very simple level, and works similarly to the main menu scene, and the library scene. This scene also implements the BGButtonSizeController.cs and SettingsController.cs. The screen can be seen below in figure 30.

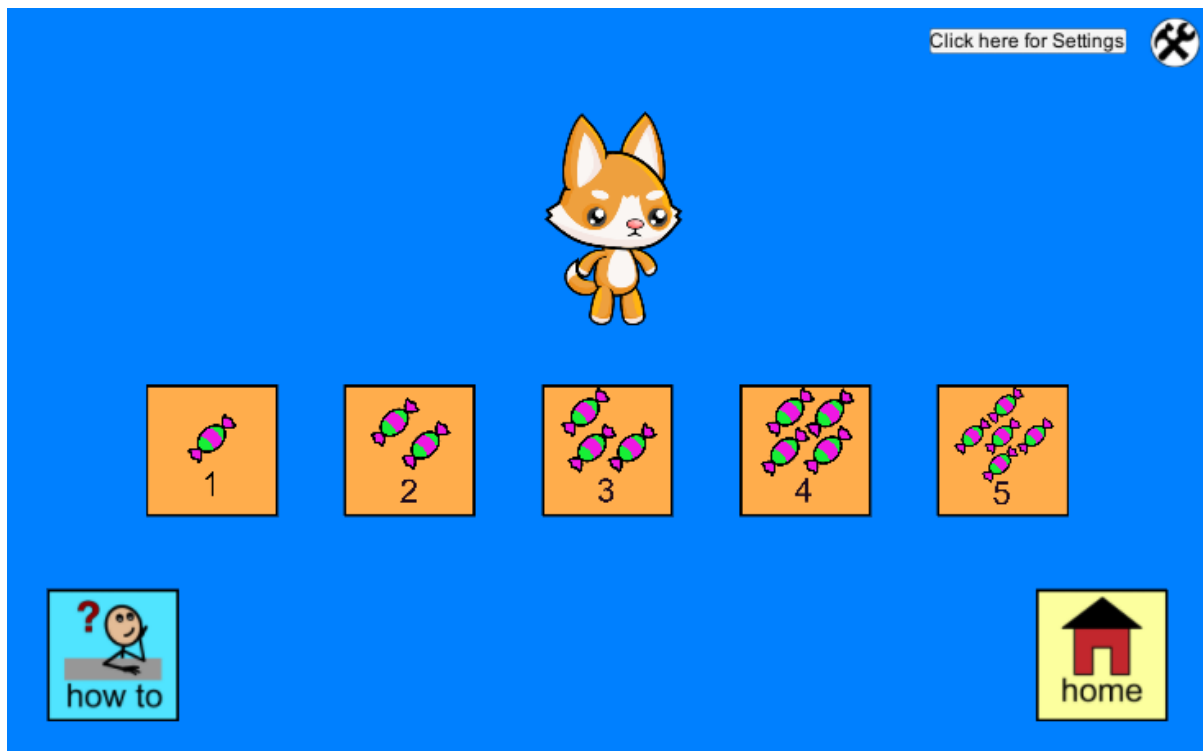


Figure 30 - Unity App: Choose level screen

### Choose.cs

This script governs the basics of this scene. It is a very basic script, and simply controls the buttons using the the SceneManager.LoadScene() function.

#### 4.2.1.6. Library

This library is similar to the level select scene, and main menu scene, in that it is simply a scene that links to other scenes. These scenes give the application logical organisation. This scene utilises Library.cs script, as well as the SettingsController.cs, the BGButtonSizeController.cs. This screen can be seen below in figure 31.

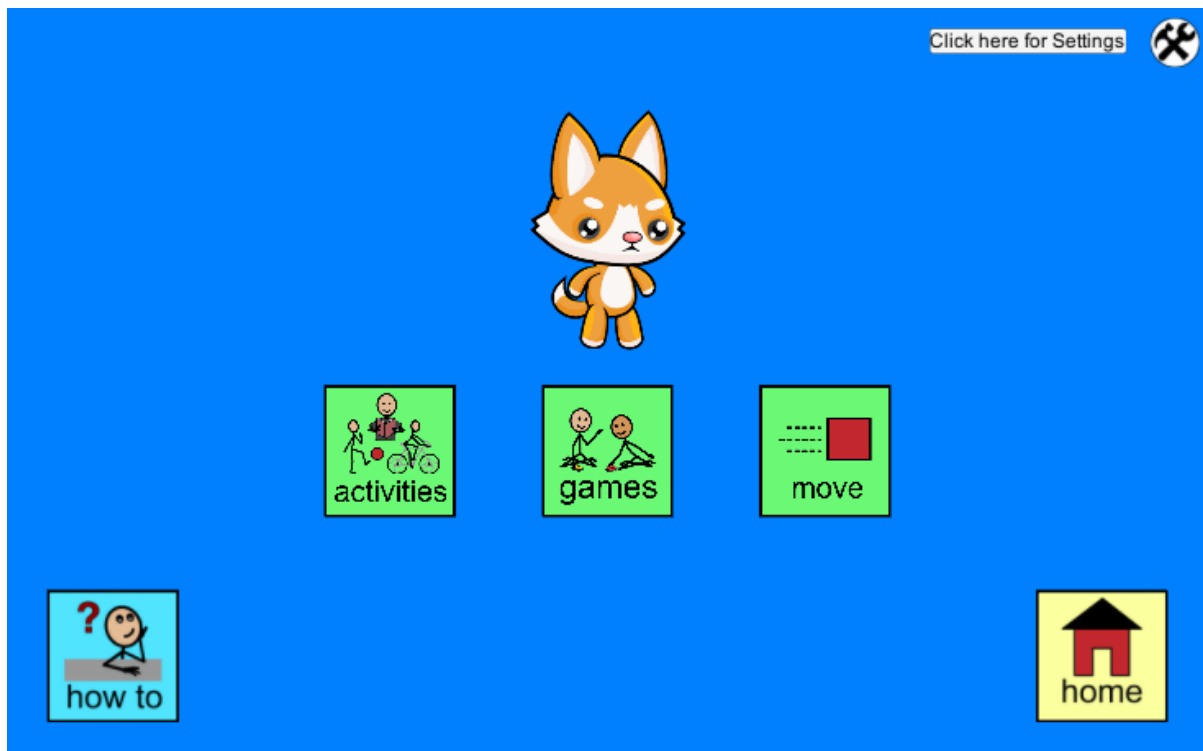


Figure 31 - Unity App: Library select screen

### Library.cs

This script is very similar to the Choose.cs script, but has less buttons. These buttons also bring the user to different scenes. This is all done using the SceneManager and assigning buttons to scene load functions.

#### 4.2.1.7. Library Sections (Activities, Moving, Games)

These sections involve selecting symbols and the user having the opportunity to observe the animation for the symbol. They operate using a single script each, that dictate the animations to be used for each button. This scene also implements the BGButtonSizeController.cs and SettingsController.cs. This screen is represented below in figure 32.

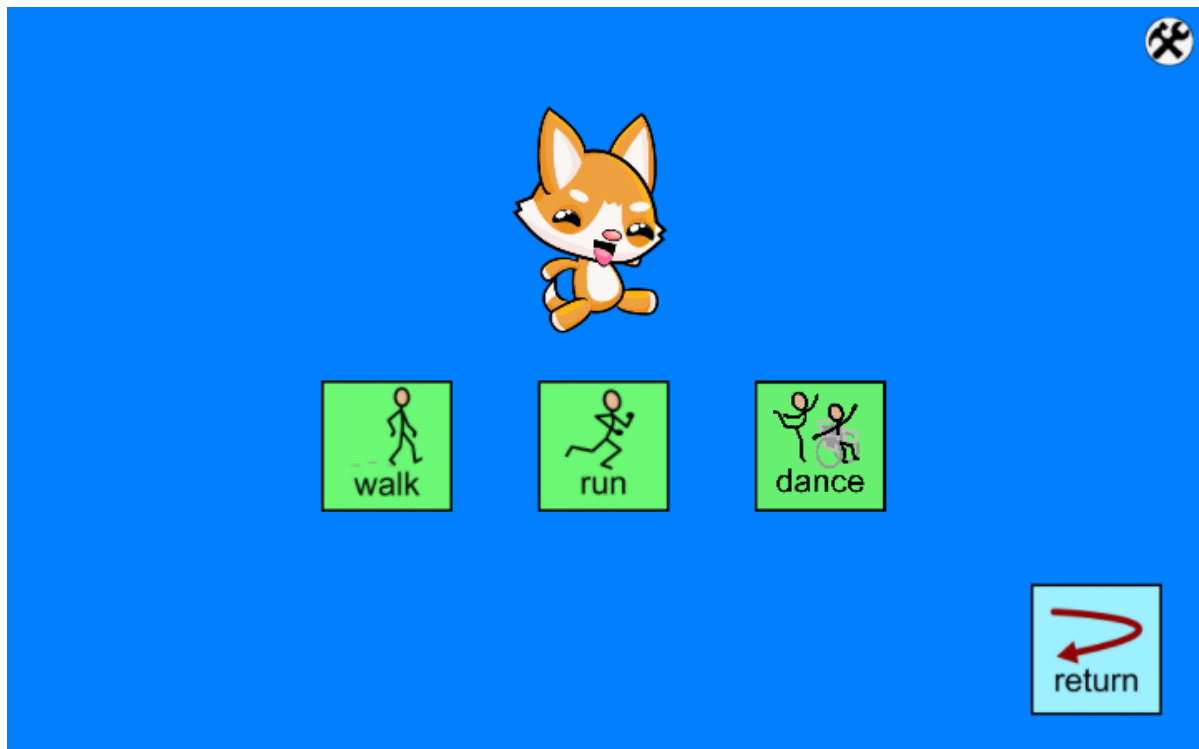


Figure 32 - Unity App: Library individual category screen: activities

### LibAnimation.cs

This script ties methods to each of the buttons that call an animation state to be changed. Each of these states represents the animation of the name of the method. For instance, running is represented by Run(). When the buttons are pressed, the onscreen character will run. This is intended to allow the user to be given an explanation for the symbols. The code can be seen below in figure 33.

```
public void Run()
{
    anim.SetInteger("State", 2);
}

public void Walk()
{
    anim.SetInteger("State", 1);
}

public void Dance()
{
    anim.SetInteger("State", 3);
}
```

Figure 33 - LibAnimation.cs

#### 4.2.1.4. Guess Levels

These levels are the main area in which the user learns vocabulary. All of them implement the SceneMan.cs, BGButtonSizeController.cs and SettingsController.cs, as these are universally required across all levels. The basic logic of the main word game is that each in round a basic single animation is played.

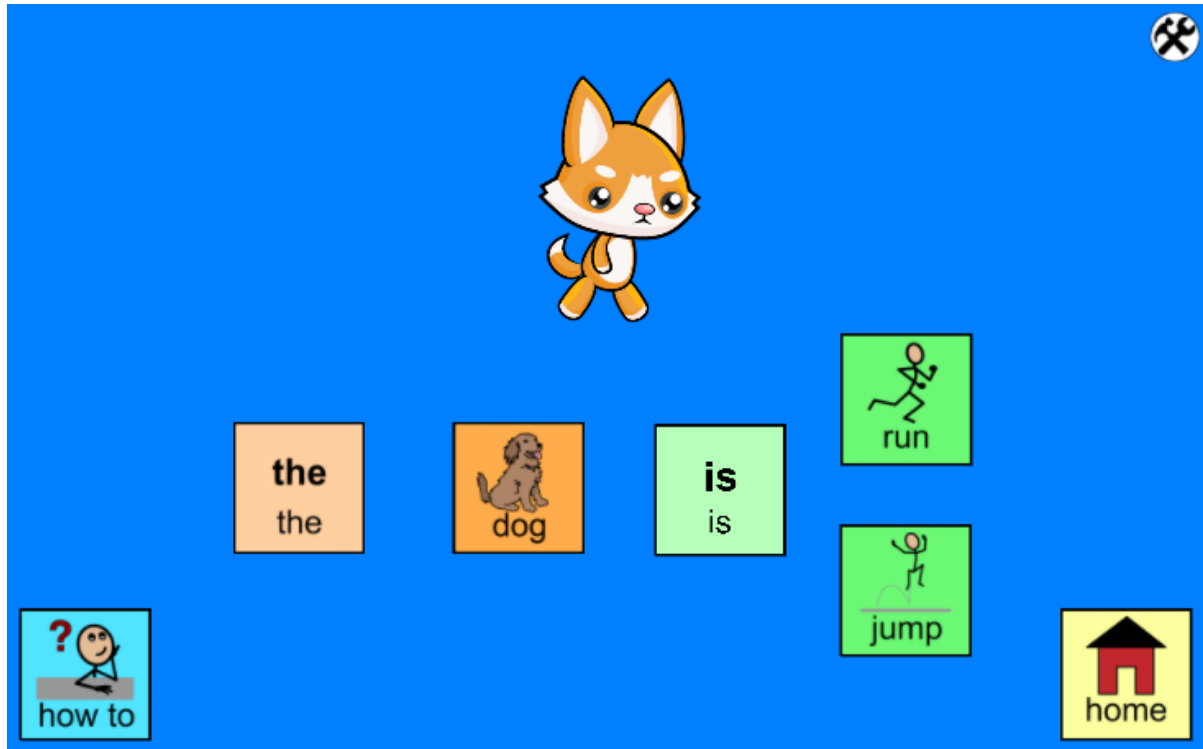


Figure 34 - Unity App: Guessing game screen

The scene also presents a number of symbols that from right to left read a sentence. In figure 34 above we can see the symbols read “the dog is” and then there are two symbols. This gives the user a choice as to what to click. To complete the challenge, the user needs to click the symbols consecutively and create the correct sentence. If they chose the last word incorrectly, it will give them the opportunity to try again. If they are successful, the game will play an applause and award them a medal. This positive feedback helps to encourage the user. These can be seen below in Figure 35.

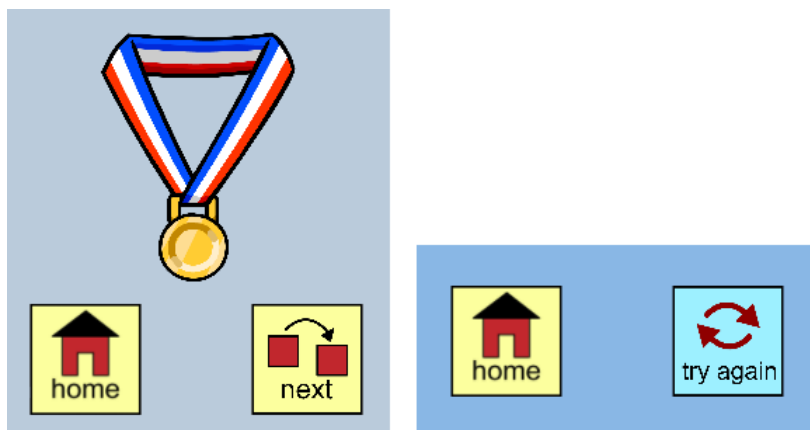


Figure 35- Unity App: Game success and failure

### Guess.cs

This script handles logic for creating the fully complete sentence and guessing the right word. In the update function, two functions are run on conditions. To be given the correct answer response, all of the buttons need to be pressed changing them to true, including the choice button. By choosing incorrectly 3 times, the application will give a response that allows the user to try again. This can be seen below in figure 36.

```
void Update()
{
    if (word1 && word2 && word3 && word4)
    {
        CorrectAnswer();
    }
    if (wrongWordCount >= 3)
    {
        WrongAnswer();
    }
}
```

Figure 36 - Guess.cs - Correct/Wrong decider

The correct answer function adjusts the players current level stored in PlayerPrefs, and assigns these to variables that can then be submitted to the online form in the coroutine function SaveScores(). This can be seen below in figure 37.

```
void CorrectAnswer()
{
    dog.GetComponent<Renderer>().enabled = false;
    MainCanvas.enabled = false;
    WinCanvas.enabled = true;
    PlayerPrefs.SetInt("LevelProgress", 1);
    PlayerPrefs.SetFloat("LeveTime", Time.timeSinceLevelLoad);
    PlayerPrefs.Save();
    if (applause == 0)
    {
        int thisLevel = PlayerPrefs.GetInt("CurrentLevel");
        thisLevel++;
        PlayerPrefs.SetInt("CurrentLevel", thisLevel);
        PlayerPrefs.Save();
        source.PlayOneShot(welldone, 1.0f);
        applause++;
    }

    timeplayed = Time.timeSinceLevelLoad;
    score = 1;

    if (scoreSaved == 0)
    {
        scoreSaved = 1;
        StartCoroutine(SaveScores());
    }
}
```

Figure 37 - Guess.cs - Correct Answer



The logic for handling the correct and false button presses is managed using bools, and can be seen below in figure 38.

```
public void Word1Button()
{
    word1 = true;
    word2 = false;
    word3 = false;
    word4 = false;
    word5 = false;
}
public void Word2Button()
{
    word2 = true;
    word3 = false;
    word4 = false;
    word5 = false;
}
public void Word3Button()
{
    word3 = true;
    word4 = false;
    word5 = false;
}
public void Word4Button()
{
    word4 = true;
}
public void Word5Button()
{
    word5 = true;
    wrongWordCount++;
}
```

Figure 38 - Guess.cs - game logic

The SaveScores() coroutine function is the most complex aspect of this script. This coroutine is called within both the CorrectAnswer() and the WrongAnswer() functions. This ensures that even rounds where the user does not win are logged online to the database. The code for the coroutine can be seen below in figure 39.

```
IEnumerator SaveScores()
{
    WWWForm form = new WWWForm();

    form.AddField("newLevel", levelname);
    form.AddField("newCategory", category);
    form.AddField("newTime", timeplayed.ToString());
    form.AddField("newScore", score);

    WWW webRequest = new WWW(db_url + "save.php", form);

    yield return webRequest;

    string webRequestString = webRequest.text;
    print(webRequestString);
}
```

Figure 39 - Guess.cs - Score upload coroutine

The function works by assigning the variables taken from the application, and inserting them into form fields. This form is then inserted with the URL for the addScores.php file on the server, and creates a web request. This web request is then returned and sent.

### SceneMan.cs

This script handles tracking the current scene the user is on. This is used in every scene to assign the current scene number to PlayerPrefs. This ensures that when at the menu screen, and the user presses the play button, the game will resume where the player last left off. This class also contains methods for navigating to the next level, returning to the home screen, and for relaunching or repeating the current scene. This can be seen below in figure 40.

```
void Start()
{
    PlayerPrefs.SetInt("CurrentLevel", currLevel);
    PlayerPrefs.SetInt("ReturnLevel", retLevel);
    PlayerPrefs.Save();
}

public void ReturnToHome()
{
    SceneManager.LoadScene(0);
}

public void CurrentLevel()
{
    SceneManager.LoadScene(currLevel);
}

public void NextLevel()
{
    if (currLevel <= 21)
    {
        SceneManager.LoadScene(currLevel + 1);
    }
}
```

Figure 40 - SceneMan.cs

#### 4.2.1.5. Play Levels

Each of the different levels implement different scripts and combinations of the scripts created in the application. All of them implement the SceneMan.cs, BGButtonController.cs and the SettingsController.cs, as these are universally required across all levels. These levels have different mechanics, and different aesthetics, to keep the user interested and engaged.

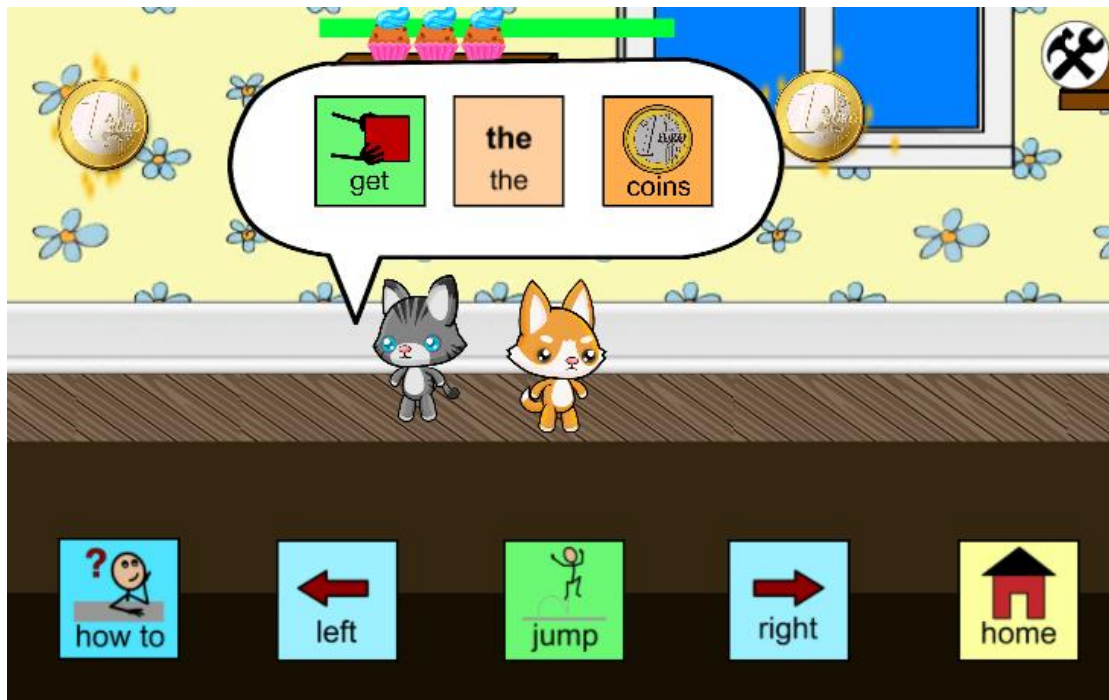


Figure 41 - Unity App: Level 1 – Collect the coins

Above in figure 41, we see level 1. The aim of this level is to collect all of the coins before the time runs out. By collecting all the coins, they win the game.

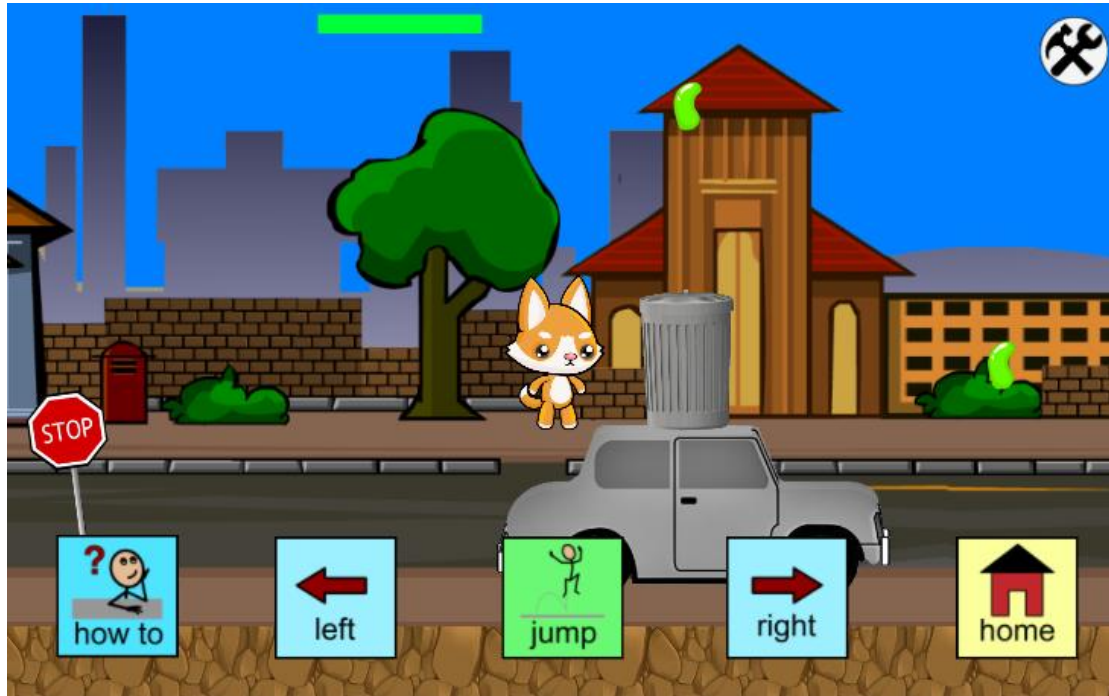


Figure 42 - Unity App: Level 2 – Collect sweets

Above in figure 42, we see level 2. The aim of this level is to collect all of the jelly beans before the time runs out. By collecting all the sweets, they win the game.

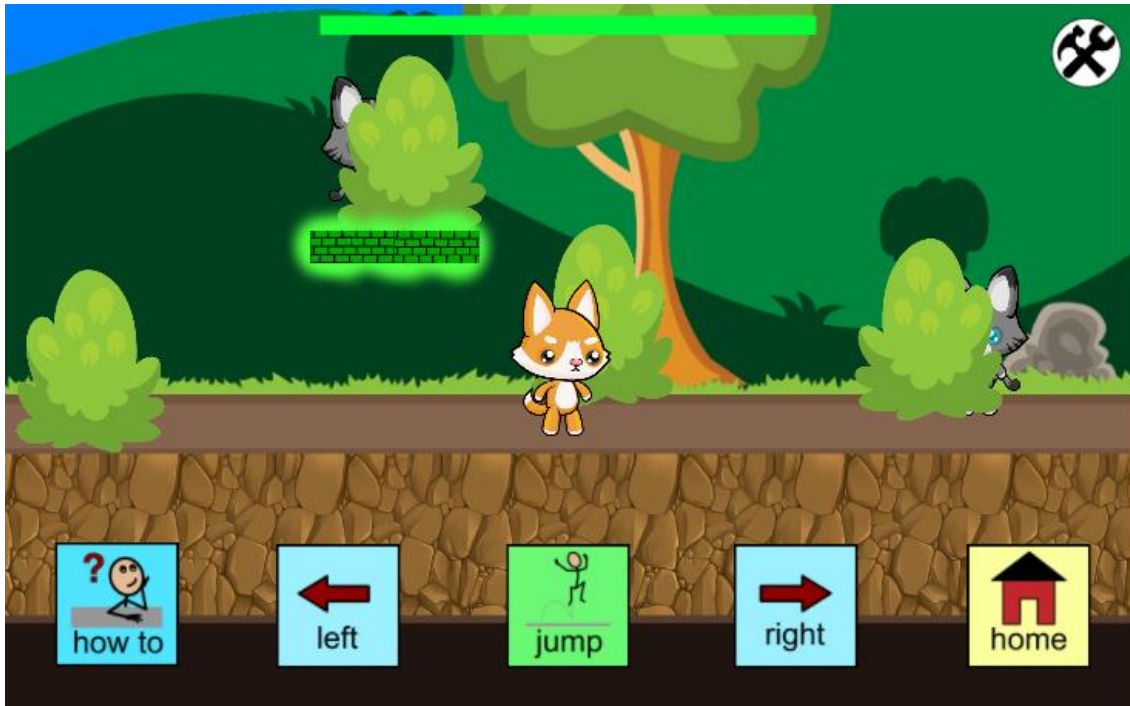


Figure 43 - Unity App: Level 3 – Hide and seek

Above in figure 43, we see level 3. The aim of this level is to find the other characters who are hiding behind some of the bushes. When the user finds them all, they win the game.



Figure 44 - Unity App: Level 4 – Chasing (being chased)

Above in figure 44, we see level 4. The aim of this level is to escape the character chasing the user. If the user doesn't get caught for long enough, they win the game.

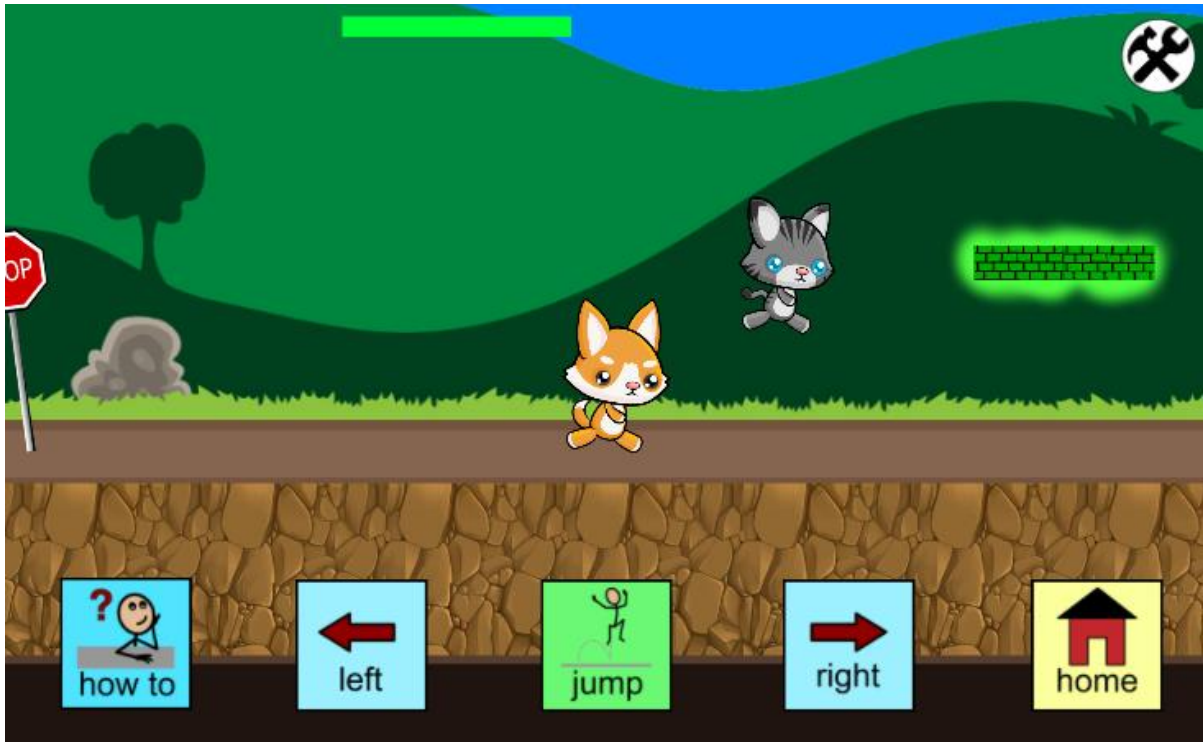


Figure 45 - Unity App: Level 5 – Chasing (chasing the cat)

Above in figure 45, we see level 5. The aim of this level is to catch the other character. When the user catches the other character three times, they win.

### CameraController.cs

This is a very short and basic script and simply gets the camera to track the player. This does so by initially searching for the Dog tag which was assigned to the player in the unity editor. The LateUpdate() function is called after the Update() function, ensuring tracking is done when other processes have completed. This method does a simple transform of the cameras position to the player's position. This can be seen below in figure 46.

```
private Transform target;

void Start ()
{
    target = GameObject.Find("Dog").transform;
}

void LateUpdate ()
{
    transform.position = new Vector3(Mathf.Clamp(target.position.x, xMin, xMax), Mathf.Clamp(target.position.y, yMin, yMax), transform.position.z);
}
```

Figure 46 - CameraController.cs

### PlayerController.cs

This is one of the more complex scripts in the Unity application. This is because of the use of Unity gaming elements, like localScale, Vector3 and velocity. The update function seen in figure 47 below does 2 main things. It moves the player in specified direction at a set speed,



and also rotates the character to face the direction they are moving. This rotation is called on the conditions that if the speed is positive(moving right along x axis) and the character is not facing right, flip the character. It also flips the character if the speed is negative (moving left along x axis) and the character is facingRight.

```
void Update()
{
    MovePlayer(speed);

    if (speed > 0 && !facingRight || speed < 0 && facingRight)
    {
        facingRight = !facingRight;
        Vector3 temp = transform.localScale;
        temp.x *= -1;
        transform.localScale = temp;
    }
}
```

Figure 47 - PlayerController.cs - Update function

The MovePlayer function handles the user's movement. It does this by moving the body of the character game object at a specified velocity. This function also handles the animations of the character. When the user is detected to be jumping, the jumping animation is invoked. When the character is not jumping, the idle animation is used. This can be seen in figure 48.

```
void MovePlayer(float movementSpeed)
{
    body.velocity = new Vector3(speed, body.velocity.y, 0);
    if(movementSpeed < 0 && !Jumping || movementSpeed > 0 && !Jumping)
    {
        anim.SetInteger("State", 2);
    }
    if (movementSpeed == 0 && !Jumping)
    {
        anim.SetInteger("State", 0);
    }
}
```

Figure 48 - PlayerController.cs – MovePlayer function

To control the user, methods needed to be created. These were for left, right, stopping and jumping. The left button makes the player speed a minus, which moves the player left along the x axis. The right button makes the player speed, a positive which moves the player right along the x. The stop button turns the player speed to 0. The jump button only functions if the user isn't already jumping. If they are not, a vector2 force is added moving the character upwards combined with the x direction. Every time the player jumps, the character barks.

Each of these functions is assigned to the relevant button in the Unity editor. Some of the code in this class was also used in non-player controlled characters. The code for this class can be seen below in figure 49.

```

public void Left()
{
    speed = -speedX;
}

public void Right()
{
    speed = speedX;
}

public void Stop()
{
    speed = 0;
}

public void Jump()
{
    if (!Jumping)
    {
        Jumping = true;
        body.AddForce(new Vector2(body.velocity.x, jumpSpeedY));
        anim.SetInteger("State", 3);
        float vol = Random.Range(volLowRange, volHighRange);
        source.PlayOneShot(bark, vol);
    }
}

```

Figure 49 - PlayerController.cs - Button control functions

## NPC.cs

This script is used for handling the collisions of the secondary characters game object. When the player touches this characters collider, a speech bubble object appears with a relevant symbol based message. For instance in the hide and seek game, when the user finds where the secondary character is hiding, the speech bubble appears with the message “you found me”. See figure 50.

```

void OnTriggerEnter2D(Collider2D other)
{
    if (other.gameObject.tag == "Player")
    {
        speechBubble.SetActive(true);
        cat.sortingOrder = 3;
    }
}

void OnTriggerExit2D(Collider2D other)
{
    if (other.gameObject.tag == "Player")
    {
        speechBubble.SetActive(false);
    }
}

```

Figure 50 - NPC.cs

## Catcher.cs

This script is used in the chasing game. It allows the time bar to be reduced when the user is not caught by the secondary character. When the user is caught by secondary characters



collider, the time bar is refilled to 1. When the secondary character catches the user, a cat sound is played. This can be seen in figure 51 below.

```
void Update ()
{
    if (TimeBar.fillAmount < 0.9f)
    {
        meowTime = true;
    }
    if (TimeBar.fillAmount > 0)
    {
        TimeBar.fillAmount -= DecreaseAmount * Time.deltaTime / 10;
    }

    if (TimeBar.fillAmount == 0f)
    {
        print("Game Over");
        this.enabled = false;
    }
}

void OnTriggerEnter2D(Collider2D other)
{
    if (other.gameObject.tag == "Player")
    {
        TimeBar.fillAmount = 1;
        if (meowTime)
        {
            source.PlayOneShot(meow, 1.0f);
            meowTime = false;
        }
    }
}
```

Figure 51 - Catcher.cs

### Chaser.cs

This script controls secondary character in the fourth level of the game. This script changes the location of the secondary character object. The large if statement works the same way it does in the PlayerController.cs. Based on the direction of movement of the object, it ensures the sprite is flipped accordingly, so the character always appears to be moving forward across the 2d plain.

```
void Start()
{
    anim = GetComponent<Animator>();
    speed = .05f;
    facingRight = true;
}

void Update()
{
    if (transform.position.x < target.position.x && !facingRight || transform.position.x > target.position.x && facingRight)
    {
        facingRight = !facingRight;
        Vector3 temp = transform.localScale;
        temp.x *= -1;
        transform.localScale = temp;
    }
    anim.SetInteger("State", 1);
    Chase();
}

void Chase()
{
    transform.position = Vector3.MoveTowards(transform.position, target.position, speed);
    anim.SetInteger("State", 1);
}
```

Figure 52 - Chaser.cs

The Chase() function is called in Update(), so the secondary character is always chasing the user. The Chase() function itself works by transforming the position of this character to the

position of the users character, at a set speed. As the character moves, the run animation is also set. This can be seen in figure 52 above.

### Runner.cs

This script is used in the fifth and final game level. A lot of this script is quite similar to Chaser.cs, however a function called Run() is used here instead. This function instead causes the secondary characters position to move away from the users. This can be seen below in figure 53. The character is also programmed to jump in this level.

```
void Run()
{
    Vector3 moveDirection = transform.position - target.transform.position;
    transform.position = Vector3.MoveTowards(transform.position, moveDirection, speed);
    anim.SetInteger("State", 1);
}
```

Figure 53 - Runner.cs

### Collectibles.cs

This script handles all collectables in the application. In the first two levels, there are collectible objects in the form of coins and jelly beans. Both of these add points to the level score and aid in completion. These are implemented using a 2d box collider, similar to other objects in the application. When the user enters this collider however, the object is destroyed and vanishes from the scene. This can be seen in figure 54.

```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.gameObject.tag == "Player")
    {
        coins.AddPoints(points);
        Destroy(this.gameObject);
    }
}
```

Figure 54 - Collectibles.cs

### Points.cs

This script is used to control the addition of points to the levels that have collectibles. In both of these levels, the user must collect as many of the coins or sweets in the time given, as the time bar decreases. Every time a collectible is picked up, the AddPoints method is called from this scene. When the number of points hits the maximum, or when the time bar is fully depleted, the level finishes. This code is seen below in figure 55.

```

void Update () {
    if (TimeBar.fillAmount > 0)
    {
        TimeBar.fillAmount -= DecreaseAmount = Time.deltaTime / 15;
    }

    if (TimeBar.fillAmount == 0f)
    {
        print("Game Over");
        this.enabled = false;
        gameOver++;
        if (gameOver == 1)
        {
            MainCanvas.enabled = false;
            EndCanvas.enabled = true;
            source2.PlayOneShot(applause, 1.0f);
        }
    }
}

public void AddPoints(int points)
{
    score += points;
    source1.PlayOneShot(collectSound, 1.0f);
}

```

Figure 55 - Points.cs

## Hide.cs

This script is quite simple and handles the logic for the hide and seek in level 3. Success in this level is driven by revealing the 3 hidden characters in the scene, who are hiding behind bushes. When the user is close enough, they become revealed and are risen in the sorting layer of the scene. When all 3 hidden characters are at the sorting layer of 3, the user has won the game. This code is shown below in figure 56.

```

void Start () {
    MainCanvas.enabled = true;
    EndCanvas.enabled = false;
    cat1.GetComponent<SpriteRenderer>();
    cat2.GetComponent<SpriteRenderer>();
    cat3.GetComponent<SpriteRenderer>();
}

// Update is called once per frame
void Update () {

    if (cat1.GetComponent<SpriteRenderer>().sortingOrder == 3 && cat2.GetComponent<SpriteRenderer>().sortingOrder == 3
    {
        Debug.Log("test");
        MainCanvas.enabled = false;
        EndCanvas.enabled = true;
    }
}

```

Figure 56 - Hide.cs

### 4.2.1.5. Animations

Animations of sprites in 2D Unity are handled by utilising different sprite states. These different states can be assigned to specific frames of the sprite animation. When the state is

assigned in code, for instance if the character is jumping, that state is assigned to the DogJump state. Unity gives developers graphical state machine diagrams, which makes state management a breeze. The users state machine diagram can be seen below in figure 57.

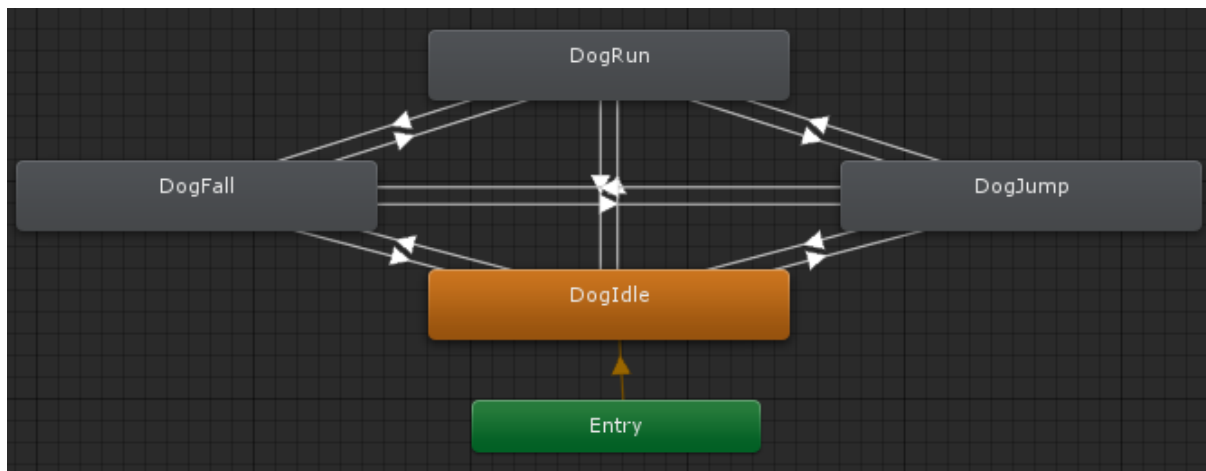


Figure 57 - User Character State Machine

#### 4.2.2. Web Application

Here, the development of the web application is documented. This entire web interface is styled using Bootstrap framework and a Bootstrap template. Both the template and the framework are mobile friendly and allow an aesthetically pleasing web interface to be created with ease.

##### 4.2.2.1. Login

Below in figure 58, the web application login can be seen. This login consists of a basic form that takes in the username and password of the user. The processing of this is driven by the Login.php file.

The image shows a web application login screen. It has a light grey background. At the top center is the title 'Login'. Below the title are two input fields: the first contains the text 'test@test.com' and the second contains four dots '....'. Below these fields is an orange button with the text 'Log in'.

Figure 58 - Web App: Login Screen

##### Login.php

This PHP file handles the user's login. It is the default page of the systems web application. The user must enter their credentials, otherwise they cannot access the rest of the web application.

In the HTML of the file, there is a simple form that takes an email/username and password input. These are then added to a POST request method. This can be seen below in figure 59.

```
<div class="container">
<form class="form-signin" method="post">
<h2 class="form-signin-heading" style="text-align: center;">Login</h2>
<label for="inputEmail" class="sr-only">Email address</label>
<input name="username" type="email" id="inputEmail" class="form-control" placeholder="Email address" required autofocus>
<label for="inputPassword" class="sr-only">Password</label>
<input name="password" type="password" id="inputPassword" class="form-control" placeholder="Password" required>
<button class="btn btn-lg btn-warning btn-block" type="submit">Log in</button>
</form>
</div>
```

Figure 59 - Web App: Login form

The PHP in this file processes the username and password from the POST request method. It then enters these into an SQL query, which is run against the database. If a result is returned, the user is logged in using a session and redirected to the logged in index page. This can be seen in figure 60.

```
session_start();
include("includes/config.php");

if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    $myusername = addslashes($_POST['username']);
    $mypassword = md5(addslashes($_POST['password']));

    $sql = "SELECT userid FROM tbl_users WHERE username='$myusername' and password='$mypassword'";
    $result = mysql_query($sql);
    $count = mysql_num_rows($result);

    if ($count == 1)
    {
        $_SESSION['login_admin']=$myusername;
        header("location: http://fvpc12561353.cloudapp.net/admin/");
    }
}
```

Figure 60 - Web App: Login form processing

## Logout.php

This file is called when the user clicks the logout button. It destroys the users session. See below in figure 61.

```
session_start();

if (session_destroy())
{
    header("Location: index.php");
}
```

Figure 61- Web App: Logout

#### 4.2.2.2. User interface

After logging in the user is brought to a welcome screen. From here they can navigate throughout the web application, and view the various charts. These charts contain the progress information of the tablet application user from the tracking table.

#### Index.php

This document handles all successful logins that are received by the Login.php. It gives the user a home screen with a welcome message. From here the user can navigate to the other document pages. This welcome screen can be seen below in figure 62.

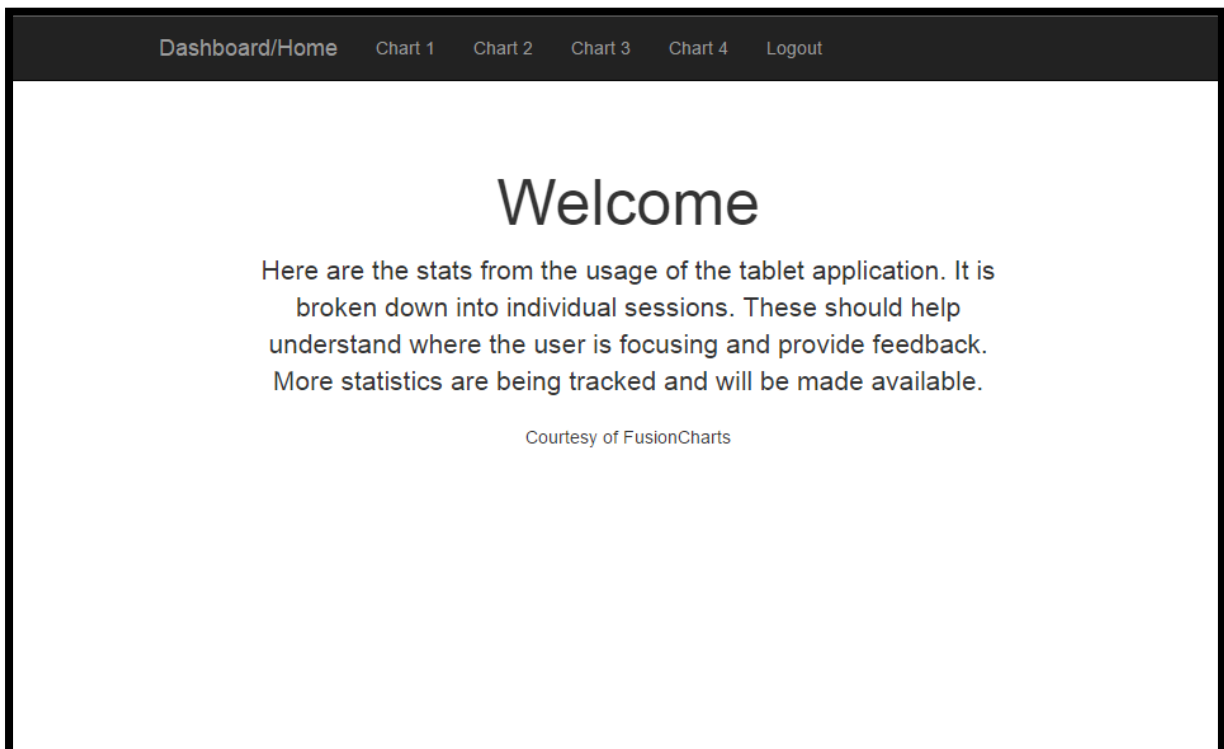


Figure 62- Web App: Welcome Screen

To ensure that users cannot access these pages without logging in, a simple php code snippet is placed at the top of each logged in document. This code verifies that the user has a logged in session. If the user tries to navigate to the logged in pages without logging in, this code will return the user to the log in screen. See code in figure 63 below

```
session_start();
if(empty($_SESSION['login_admin']))
{
    header('Location: http://' . $_SERVER['HTTP_HOST'] . '/login.php');
    exit;
}

include("../includes/fusioncharts.php");
```

Figure 63 - Session monitor

## Graph2.php (1 of 4)

The 4 chart pages operate the same, but create different charts. For the explanation of the implementation here, the Graph2.php file was used. Below in figure 64, we can see the chart appears with data based on the tracked Unity app data.

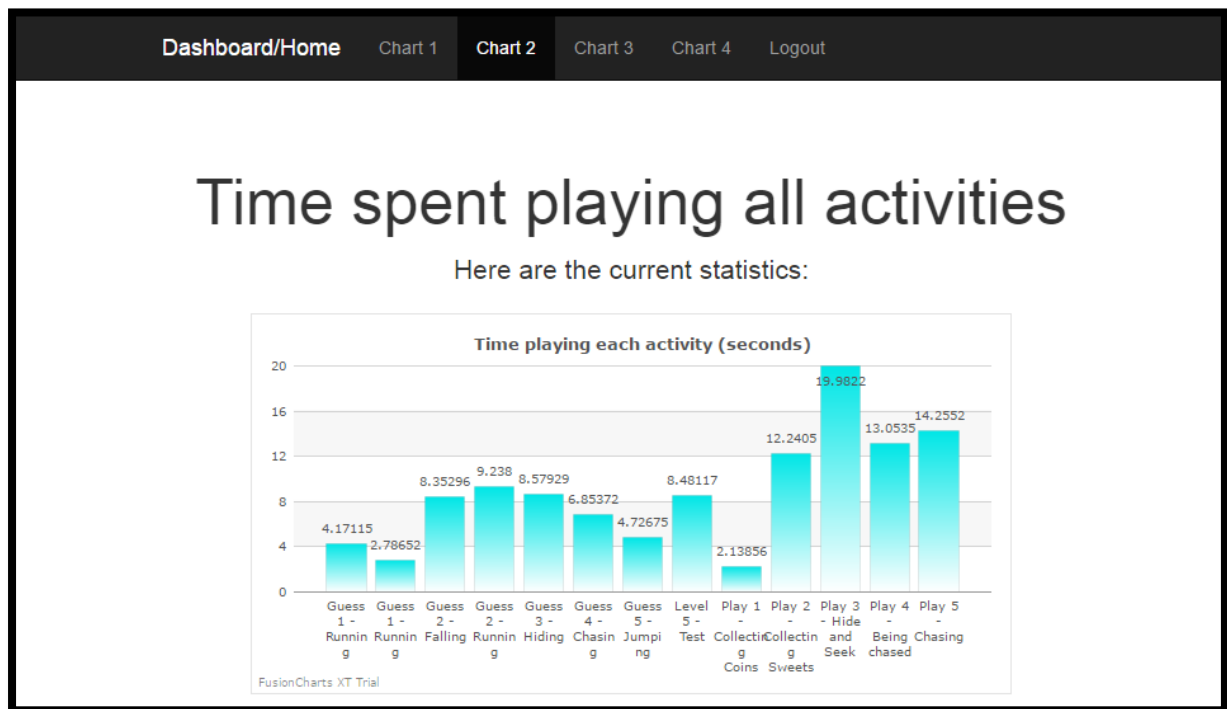


Figure 64- Graph2.php

Each of these graph.php files creates a connection to the database using the set servername, username, password and database name. If a new connection cannot be made, the connection fails. If it is successful, the code progresses to the FusionCharts framework usage. At the end the connection is closed. These functions can be seen below in figure 65.

```
$servername = "localhost";
$username = "root";
$password = "Language2016";
$dbName = "xvr";

$conn = new mysqli($servername, $username, $password, $dbName);

if(!$conn)
{
    die("Connection failed.". mysqli_connect_error());
}
else echo("    Connection success");

$conn->close();
```

Figure 65 - Web App: Connecting PHP to MySQL, and Disconnecting



After the connection is made, the PHP file goes on to first create the query string. This query pulls all the needed data from the table that will be used in the chart. This query is then run. If a result is returned, an array of the specification of the chart is created. Then for every row in the results table, the array applies these custom settings. The data is then encoded to JSON object data. A chart object is then created using the specifications chosen. Finally the chart is rendered. All of this can be seen in figure 66.

```
$strQuery = "SELECT timeplayed, category, levelname FROM tracking WHERE category LIKE '%Game%'";

$result = $dbhandle->query($strQuery) or exit("Error code ({ $dbhandle->errno }): { $dbhandle->error }");

if ($result) {
    $arrData = array(
        "chart" => array(
            (
                "caption" => "Time spent playing game levels (seconds)",
                "paletteColors" => "#e60000",
                "bgColor" => "#ffffff",
                "borderAlpha"=> "20",
                "canvasBorderAlpha"=> "0",
                "usePlotGradientColor"=> "1",
                "plotBorderAlpha"=> "10",
                "showXAxisLine"=> "1",
                "xAxisLineColor" => "#999999",
                "showValues" => "1",
                "divlineColor" => "#999999",
                "divLineIsDashed" => "0",
                "showAlternateHGridColor" => "1"
            )
        );
    $arrData["data"] = array();

    while($row = mysqli_fetch_array($result)) {
        array_push($arrData["data"], array(
            "label" => $row["levelname"],
            "value" => $row["timeplayed"]
        ));
    }

    $jsonEncodedData = json_encode($arrData);

    $columnChart = new FusionCharts("column2D", "myFirstChart" , 600, 300, "chart-1", "json", $jsonEncodedData);

    $columnChart->render();
}
```

Figure 66 - Web App: Using FusionCharts

### addScore.php

This PHP file is used by the Unity application. On every level completion, the Unity application calls the SaveScore coroutine function. This function submits the levelname, category, timeplayed, and score to a WWWform and sends that through a WWWrequest to this file. Here the form information gets processed and assigned to variables.

These variables are then placed in an SQL INSERT query. This query is then submitted to the MySQL database. Successful insertion can is then reflected immediately on the charts in the application, after the next page load. If there is any issue with the score either reaching the

server, or the server being unable to insert the row into the table, then the error is logged. This code is seen below in figure 67.

```
$level = $_POST['newLevel'];
$category = $_POST['newCategory'];
$time = $_POST['newTime'];
$score = $_POST['newScore'];

echo "$level" . " | " . "$category" . " | " . "$time" . " | " . "$score";

$sql = "INSERT INTO tracking (levelname, category, timeplayed, score) VALUES ('$level', '$category', '$time', '$score')";

if ($conn->query($sql) === TRUE)
{
    echo "New record created successfully";
}
else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}
```

Figure 67- addScore.php

#### 4.2.3. MySQL database

The database implementation in this system were quite simple. There are two tables, the user table and the tracking table. The user table contains only one user, with a username and password which is hashed using md5. This is in figure 64.

userid	username	password
1	test@test.com	098f6bcd4621d373cade4e832627b4f6

Figure 68 - user table

In figure 69 we can see the tracking table. This table contains the level name, category, time played and users score. Every time the addScore.php script gets called by the Unity app, a new row is added to this table. As this table grows, as do the charts, displaying more and more data, giving Orlagh a larger spread of feedback.

trackingid	levelname	category	timeplayed	score
32	Guess 1 - Running	Movement	4.17115	0
33	Play 1 - Collecting Coins	Game	2.13856	1
34	Guess 2 - Running	Movement	9.238	1
35	Play 2 - Collecting Sweets	Game	12.2405	1
36	Guess 3 - Hiding	Activities	8.57929	1
37	Play 3 - Hide and Seek	Game	19.9822	1
38	Guess 4 - Chasing	Activities	6.85372	1
39	Play 4 - Being chased	Game	13.0535	1
40	Guess 5 - Jumping	Movement	4.72675	1
41	Play 5 - Chasing	Test	14.2552	1
56	Level 5 - Test	Test	8.48117	1
59	Guess 1 - Running	Moving	2.78652	1
60	Guess 2 - Falling	Moving	8.35296	0

Figure 69 - tracking table

### 4.3. List of classes, scripts and files

#### Unity App

- BGButtonSizeController.cs
- CameraController.cs
- Catcher.cs
- Chaser.cs
- CheckConnection.cs – Using tips found online in forums: <http://forum.unity3d.com/>
- Choose.cs
- Coins.cs
- Collectibles.cs
- Dropdown.cs – Getting the dropdown functioning correctly proved difficult, so a tutorial was used: <https://www.youtube.com/watch?v=nH-ZGodtIQ>
- Ground.cs
- Guess1,2,3,4,5.cs
- Hide.cs
- LibAnimation.cs
- Library.cs
- Menu.cs
- NPC.cs
- PlayerController.cs – Some vector code functionality used from PluralSight Unity2D tutorial: <https://www.pluralsight.com/courses/twod-animated-char-unity-2299>
- Runner.cs
- SceneMan.cs
- Settings.cs
- SettingsController.cs

#### Web App

- Index.html – Bootstrap basic login
- Login.php – Bootstrap basic login
- Logout.php
- Graph1.php – Using fusion charts templates
- Graph2.php
- Graph3.php
- Graph4.php
- addScore.php – Held found a this website: [http://wiki.unity3d.com/index.php?title=Server\\_Side\\_Highscores](http://wiki.unity3d.com/index.php?title=Server_Side_Highscores)

### 4.4. Additional Resources

- Symbolstix symbols used with permission and license (n2y)
- Character sprites were taken from this website: [www.gameart2d.com/freebies.html](http://www.gameart2d.com/freebies.html)
- Background images were retrieved from free online resources and Google images.

## 5. System Validation

The validation of this system was a key factor in measuring its success. This chapter begins with a section on Testing, which covers test cases of the system. This was one of the main ways the system was validated. The section after this discusses the importance of user feedback.

The main validation throughout the course of this project came predominantly from this feedback. This was received from the third party user, Orlagh. Due to limitations, she acted as the target user for the Unity application in place of Sophia. She provided crucial evaluations on the project system in its finishing stages.

### 5.1. Testing

The testing phases of this project involved testing both the Unity and the web applications. Due to the user centric nature of the system, and focus on interface design, black box testing was given massive priority over white box testing. Black box testing involves functionality testing, passing use cases and meeting requirements. These are the primary tests that drove the development of this system. White box testing would involve examining code coverage and designing unit tests.

No white box testing was completed during this project. The majority of the code and scripts in the system would not benefit from unit testing. If there had been more time to do the project, compatibility testing, performance testing and security testing all would have been completed.

The key use cases in the project are outlined in the next sections.

#### 5.1.1 Unity Application Test Cases

Interface and functional testing were the most widely used testing process in the project lifecycle. These involved the testing of core application functions and ensuring the interface was consistent and reliable. Located below are test cases that were completed for the Unity application.

##### *5.1.1.1. User informed of connection status*

###### **Overview**

User should be able to open the application and be told if it's connected to the server.

###### **Steps**

1. The application is launched by pressing the icon.
2. After the load screen, the menu screen will display the connection status.

###### **Result**

Pass: Connection status displayed

##### *5.1.1.2. Adjusting background colour from the settings menu*

###### **Overview**

User should be able to navigate to the settings menu and adjust background colour.

### **Steps**

1. The application is launched by pressing the icon, bringing them to the menu screen.
2. From the menu screen, the settings button is held.
3. The second button that appears is then clicked. This opens the settings menu
4. The background dropdown is clicked, revealing more options.
5. The desired background colour is then chosen.

### **Result**

Pass: The user successfully changed the background colour.

#### *5.1.1.3. Adjusting symbol size from the settings menu*

### **Overview**

User should be able to navigate to the settings menu and adjust icon size.

### **Steps**

1. The application is launched by pressing the icon, bringing them to the menu screen.
2. From the menu screen, the settings button is held.
3. The second button that appears is then clicked. This opens the settings menu
4. The button size scale dropdown is clicked, revealing more options.
5. The desired button scale is then chosen.

### **Result**

Pass: The user successfully changed the button size.

#### *5.1.1.4. User informed of connection status*

### **Overview**

User should be able to open the application and be told if it's connected to the server.

### **Steps**

1. The application is launched by pressing the icon.
2. After the load screen, the menu screen will display the connection status.

### **Result**

Pass: Connection status displayed

#### *5.1.1.5. Playing the game from the first level*

### **Overview**

User should be able to navigate to the main menu, and load the current level.

### **Steps**

1. The application is launched by pressing the icon
2. From the menu screen, the play button is pressed.

### **Result**

Pass: The user was successfully brought to play the first/next level.

#### *5.1.1.6. Entering a chosen level*

### **Overview**

User should be able to navigate to the level selection menu, and choose to play any of the platform levels.

### **Steps**

1. The application is launched by pressing the icon
2. From the menu screen, the choose button is pressed.
3. From here the desired level is chosen and pressed,

### **Result**

Pass: The user successfully reached the selected level.

#### *5.1.1.7. Viewing a library animation*

### **Overview**

User should be able to navigate to the individual library category pages and view animations of symbols.

### **Steps**

1. The application is launched by pressing the icon
2. From the menu screen, the library button is pressed. This opens the library.
3. From here the library category is chosen and pressed. This opens the category.
4. From here presses the symbol of the animation they wish to see.

### **Result**

Pass: The user navigated to the library page, and played an animation.

#### *5.1.1.8. Completing a guess the word level*

### **Overview**

User should be able to navigate to the game section, and complete the word challenge given. This involves an animation and a sentence describing the animation, with a choice of word at one point. To succeed, the full sentence of symbols needs to be pressed.

### **Steps**

1. The application is launched by pressing the icon

2. From the menu screen, the play button is pressed. This opens the next challenge (in this case a guess the word challenge).
3. The first word symbol is pressed.
4. The second word symbol is pressed.
5. The third word symbol is pressed.
6. Then the correct symbol from the choice of two is pressed, thus completing the challenge.

### **Result**

Pass: The user navigated to the game and completed the word challenge.

#### *5.1.1.9. Completing a platform level*

### **Overview**

User should be able to navigate to the game section, and complete the word challenge given. This involves an animation and a sentence describing the animation, with a choice of word at one point. To succeed, the full sentence of symbols needs to be pressed.

### **Steps**

1. The application is launched by pressing the icon
2. From the menu screen, the choose button is pressed.
3. From the choose menu, the first level is selected.
4. In the first level, the left, right, and jump buttons are used to navigate.
5. All of the coins are collected in the level before the time runs out, thus completing the challenge.

### **Result**

Pass: The user navigated to the first platform level, and completed the challenge.

#### *5.1.2. Web Application Test Cases*

##### *5.1.2.1. Logging in*

### **Overview**

User should be able navigate to the URL, enter their credentials, and successfully log in.

### **Steps**

1. Navigated to the web application.
2. Entered the credentials.
3. Clicked the submit button.

### **Result**

Pass: The user was logged in successfully.

#### 5.1.2.1. Accessing/Viewing Charts

##### Overview

User should be able navigate to site, log in and access charts

##### Steps

1. Navigated to the web application.
2. Entered the credentials.
3. Clicked the submit button and logged in.
4. Navigated from welcome screen to Chart screen clicking the tab button.
5. Charts displayed on the screen

##### Result

Pass: The charts displayed correctly.

#### 5.1.3. User Testing

User testing is imperative to the success of a project such as this. It was intended that Orlagh (if possible with the inclusion of Sophia) would complete short assigned test cases. However due to limitations and time constraints, these were not completed. The test case template created can be found in Appendix 2. However Orlagh was able to guide the design of the application and continued design changes were implemented based on her testing feedback.

Throughout the course of the project, email correspondence and meetings with Orlagh provided necessary guidance for UI testing. Initially, some factors were not taken into consideration. It was originally thought that Sophia would struggle with handling button usage and require very tight hand holding. However, Orlagh informed that Sophia was capable of handling many buttons on a screen at one time. This UI testing lead to adjustments early on that impacted the overall design, allowing development to rapidly move forward.

Orlagh was met twice throughout the late stages of the system validation. These system validation meetings were thus considered precious time in the project timeline. Even though Sophia was not testing the system herself, Orlagh was able to do so on her behalf. Orlagh's keen understanding of Sophia's disabilities, ensured that the application was meeting her needs.

Orlagh's testing involved using the buttons to navigate the system, and checking each of the features to ensure Sophia would be able to use them. The first validation meeting was where the majority of feedback was given. From the first meeting to the second, many major changes were made, and verified that they were done correctly by our user.

The second validation meeting involved Orlagh offering further tips and guiding the final finished product with her feedback.

#### 5.2. User Feedback

As mentioned continually throughout the project, user based feedback from Orlagh has been vital to this projects completion. Throughout the design and implementation phases, UCD (user-centred design) iteratively changed application in both aesthetics and features.



This feedback was given both through email correspondence and personal meetings. This will be discussed further in the next section, which discusses the project plan, in section 6.4.2.

#### 5.4 Demonstration

Throughout the implementation chapter, all of the project systems functionality was covered. However, screenshots and explanations may not suffice in exhibiting the functionality of the system applications.

A video demonstrating the project system will be made available on or very shortly after the project deadline. This will demonstrate the use of the application, and will be available on my YouTube channel:

<https://www.youtube.com/user/kiwihogan13>

## 6. Project Plan

This section outlines the project agenda and how the project evolved considerably over its lifetime. Multiple iterations of design, implementation and testing, lead to a constantly changing project. The timeline began in October 2015, and was wrapped up in early April 2016.

### 6.1. Overview

In early October 2015, Orlagh came to DIT and gave a short presentation to students and lecturers, informing them of the opportunity to do a community based project. She spoke about her daughter and the difficulties she faces. After this, ideas were suggested to Orlagh, one of which was the idea for this project.

Throughout the months of October and November, the early research phase was conducted. This involved researching Sophia's disabilities, and the potential solutions to the problem area of the project. Technologies were also researched at this point, and some major technology decisions were made, like deciding to use Android as the tablet platform and PHP for the server side code.

This research was then used to design low fidelity prototypes in late November. These prototypes were used as a proof of concept to show how the tablet application and web application would work. Feedback was given from Orlagh based on design choices and changes she recommended. These duly changes were noted.

In December the interim submissions were completed. This marked the first major checkpoint for the project. It involved the submission of a report and a short presentation of the proposed project system. This was useful, as constructive feedback was given from the project supervisor and audience of the presentation. Shortly after this, exams became a focal point, and the project was temporarily given less priority.

From mid-January onwards, the specific technologies were selected and a vertical prototype was created. This involved connecting the different technology layers, ensuring they worked seamlessly together, without major issues. A demo Unity app was created, which was connected to the PHP server. The PHP was then connected to the MySQL database. The web interface was then created to connect to the same database as the Unity application. This brought the project into February.

In February, the first high fidelity prototypes were created. These were both the Unity Android application and the web interface. In late February, both of these were demoed to Orlagh, in order to obtain feedback and change the applications accordingly. This feedback was vital in ensuring the finished product was user centric in design.

In March the final system was implemented and completed. After this, the majority of the report was documented and written.

### 6.2. Original plan

For the proposed project, an initial Gantt chart was created. This was the original base plan of the project. It outlined the various stages, time management, and date deadlines. The chart can be seen in figure 7 below.

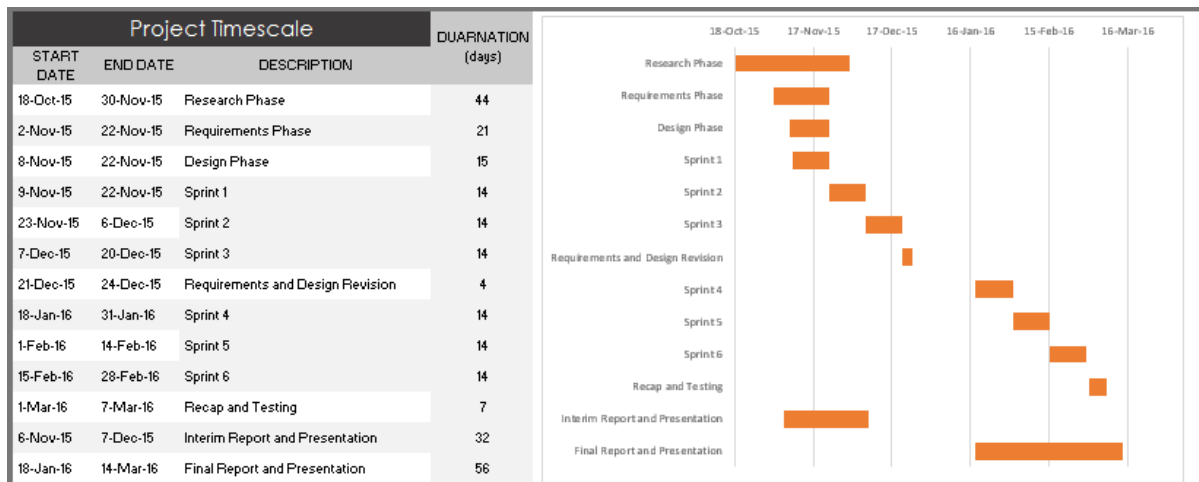


Figure 7- Original Project Timeline Gantt Chart

### 6.2.1. Details on original phases

Research Phase	<ul style="list-style-type: none"> <li>- Researched background of learning disabilities etc.</li> <li>- Investigated current market, similar apps</li> <li>- Technology to use, compare to others, critical analysis</li> <li>- Data analysis tech, what can be used</li> </ul>
Note:	The requirements phase and design phase were first completed in sequence, but were then revised accordingly. This was repeated several times early in the project.
Requirements Phase	<ul style="list-style-type: none"> <li>- User requirements established</li> <li>- Features of the application</li> </ul>
Design Phase	<ul style="list-style-type: none"> <li>- Technology being used, justification of choice</li> <li>- Use case and other UML diagrams</li> <li>- Layers, connectivity, etc.</li> <li>- Design database</li> </ul>
Note:	Each sprint phase involves a brief design phase, development, and ends with some functionality testing. If the last Sprint has any uncompleted tasks, these are done at the start. Time allowance for delays has been incorporated, by setting an early project deadline.
Sprint 1	<ul style="list-style-type: none"> <li>- Android Registration and Login</li> <li>- Android Main Screens</li> <li>- Develop logic of main application, use gifs or static images</li> </ul>
Sprint 2	<ul style="list-style-type: none"> <li>- Database design and look at implementation</li> <li>- Sort out data storage of users details when they register,</li> <li>- Allow users to log in using these stored details in the app and interface</li> </ul>
Sprint 3	<ul style="list-style-type: none"> <li>- Basic Web Interface that allows log in to database</li> <li>- Implement log in and registration on both Android and web application</li> </ul>
Requirements and Revision	<ul style="list-style-type: none"> <li>- Assess correct direction for application by revising requirements</li> <li>- Check for major bugs, test functionality</li> </ul>
Sprint 4	<ul style="list-style-type: none"> <li>- Use animation software to create html5 videos or gifs,</li> <li>- Or use OpenGL and or Unity, or use basic animations based on basic android java code</li> <li>- Create basic and small word library, 30 words</li> </ul>
Sprint 5	<ul style="list-style-type: none"> <li>- Implement game logic and progress storage in database</li> <li>- Implement cosmetic and design features and tidy up app</li> </ul>
Sprint 6	<ul style="list-style-type: none"> <li>- Develop data graphing and charting section</li> <li>- Style and design web interface</li> </ul>

Recap and testing	<ul style="list-style-type: none"> <li>- Recap on objectives and goals, and assess</li> <li>- Check functionality and assess</li> <li>- Search for security flaws and loopholes</li> <li>- Give to user for testing for real feedback</li> </ul>
Interim Report and Presentation	<ul style="list-style-type: none"> <li>- Collect and document Research, Planning, Design</li> <li>- Development so far etc.,</li> <li>- 20ish pages, include diagrams and charts and references</li> <li>- Slideshow Presentation preparation</li> </ul>
Final Report and Presentation	<ul style="list-style-type: none"> <li>- Similar to interim</li> <li>- Contain all information, research, steps (design, research, all of development etc.)</li> </ul>

In order to achieve these goals, deadlines were set in the project, which allowed small contingency periods for catching up. However, many areas of the project changed, and the research phase and iterative design phases were longer than originally planned. As a result, the project evolved from the original strategy.

### 6.3. Project evolution

The initial Gantt chart followed an optimistic and somewhat restrictive plan and timescale. Due to the nature of user centric design, the design phase was not restricted to the period in November. Instead, there were multiple iterative design phases, which happened upon receiving feedback from Orlagh. This lead to a much more customised and user focused design. This was the first major change in the project plan.

From early on, Android had been chosen as the development platform for the application. However, it became clear quite quickly that Unity offered a very useful feature set that would streamline the project development. This was another major change in the software, as it influenced objectives in the initial sprint from the project plan.

Weekly meetings with the project supervisor lead to multiple changes throughout the project timeline. His guidance took precedence over the original plan, as he helped to guide the projects focus correctly. This was a crucial force on the project, as it made potential success more feasible.

Community project group meetings and meetings with Orlagh were other factors that had not been considered in the midst of the project plan. These meetings although time consuming, were hugely beneficial to the project. They did however lead to revisiting design and implementation phases, and even research phases throughout the project. Many of these meetings with Orlagh occurred late in the project timeline. This lead to a shortage of time for implementing some crucial changes in the project system. If the project was to be done again, more regular meetings early on in the project time would be arranged.

The initial Gantt chart also neglected the testing phases. System validation proved to be much more important than previously expected. In order to prove the concept idea behind the project, regular user centric testing had to be completed, both with low fidelity and high fidelity prototypes.

The original project plan also focuses on certain aspects, which throughout the time working on it, were deemed to be very low priority. Features such as good security in the app, and registration systems for the application. These features early on were deemed unnecessary, as the application has only one core user.

In many ways, the project transformed over time, and became much more different than the initial proposal. This shows the true implementation of user centric design methodology, as the system became tailored for the user. In the end, this led to a more substantial and more useful project.

#### 6.4. Project meeting and correspondence

Throughout the course of this project, regular meetings played an important role. This included meeting the project supervisor, meeting Orlagh, and meeting with other students. Some of these meetings were physical, and others took place via video call or email.

##### 6.4.1. Meetings with project supervisor

In total there were twelve one on one meetings with the project supervisor. Six of these took place in the first half of the project, once every week between mid-October and early December. The majority of these meetings involved meeting in person, with one being a video call. These meetings involved developing the idea for the project, and establishing requirements and early design. Everything from the development platform, to the sets of symbols to use in the system were discussed. During the last meetings in this half of the project, the interim report was discussed and guidance was given.

The last six meetings took place in the second half of the project. These took place from early February through to late March. These meetings were crucial as each week, the core development components were covered with the supervisor, and evaluation of implemented design was given. These meetings also covered other core areas of the project. These included receiving feedback from the interim report, and guidance for the final report.

These meetings were vital in ensuring that the project maintained focus. Throughout points in the project, complexity took the back bench. Although this brought about concerns, it was insisted that the core focus of this project is user driven design.

##### 6.4.2. Meetings with Orlagh

The first meeting with Orlagh was an informal presentation she gave in early October. This presentation gave her the opportunity to explain where she felt project applications could be developed to help her daughter. After this meeting, project ideas were proposed via email correspondence, and these were tweaked.

This was followed by a video call on October 5<sup>th</sup>. This video call involved each of the community project students pitching the proposed ideas, and Orlagh gave feedback to each proposal. The idea for this project was approved, with some initial major modifications proposed. These included having more game related elements, and implementation of cause and effect interactivity.

Further communication with Orlagh between November and late January was minimal and done through email. This involved negotiating design and implementation decisions, such as using Android instead of IOS as the base platform. Mock-ups and storyboards were also presented to her during this period. For the month of February there was minimal correspondence with the target user, as much of the implementation was taking place.

On the 8<sup>th</sup> March, the first high fidelity prototype was demoed to Orlagh. This gave her an opportunity to see how the evolved system functioned, including the Unity Android app and the web application. The feedback received at this point was crucial, as it lead to very important modifications. These included button position, level organisation and settings access. It was also suggested that certain games be implemented, such as hide and seek and chasing. These were two valuable ideas, as Orlagh knew Sophia enjoyed these games.

On the 15<sup>th</sup> of March, the last pre submission project meeting with Orlagh took place. This was another crucial meeting in the project period. This involved receiving final feedback for the project system, and making minor tweaks to symbols, lessons etc. Overall, Orlagh was very pleased with the near finished product.

Due to the fact that there may not be another opportunity to meet Orlagh and demonstrate the finished product, a video will be created to show the final implementation.

#### *6.4.3. Meetings with other students*

Throughout the project lifespan, email correspondence and meetings were also organised between the students doing projects for Sophia. These involved the different students pooling valuable resources and sharing information. This information included symbols, user-centred design tips, and references for background and technology research. These meetings although not as vital as meetings with supervisor and Orlagh, were beneficial to the project.

All of these project planning elements, helped to bring the project to a point of success.

## 7. Conclusion

This project began as a simple approach to help one user improve her vocabulary. However, it quickly became much more than this. This was a major learning experience for me, both technically and practically. Each stage of the project was its own learning experience.

In the research stage of the report, it becomes clear that building an application for a user with such debilitating and impeding disabilities, is a considerable challenge. All aspects from the specifics of these disabilities, to how people with them learn differently to students without disabilities. This stage also provided a vital analysis of suitable technologies, and allowed the appropriate ones to be selected for use.

In the design stage, the core focus of the project is addressed. The user centric design of the tablet based application is the most important part of this project. Focusing on the design and ensuring Sophia can in fact use the application, is key. This involved iteratively receiving feedback on mock up designs and storyboards from Orlagh. This was key in maintaining the user focus. This section also allowed each system layer to be designed, in preparation for implementation.

The implementation stage is where the system was brought to life. This was done using the variety of chosen technologies, and implementing the designed system. Many issues were faced throughout this section, but it was in itself a standalone learning experience.

Connecting the different application layers was troublesome, and I learned about many technical limitations of the Unity platform.

System validation was a vital stage in the project. This was driven by feedback received from Orlagh and using this to verifying and ensure the continued success of the project. This section also involved interface testing and functional testing which I did iteratively between development stages. I learned here of the importance of user testing software.

The project plan section discusses how this project evolved throughout its lifecycle. It began as an idea to help Sophia to learn vocabulary. There were initial requirements established, which had varying levels of priority. As time moved quite rapidly, some of the low priority requirements were eliminated, and others that were overlooked were added in.

The finished product of the project combines serious game elements with symbolic language representation for the target user, with a fun and interactive approach.

### 7.1. Goals and objectives

The main goal of the project was to facilitate the specific user in learning the meaning of symbols, in a fun and interactive manner. Achieving this goal required that the following objectives were met.

#### To understand the users disabilities and how to reach her.

This was a difficult objective, and required substantial research and contact with Orlagh. From what I learned about dyslexia, dyspraxia and autism, I concluded that standard teaching practices would not apply to Sophia. I then researched other teaching means like using animation and gamification aspects, and found success there. This objective was completed.

To develop a tablet based app for the particular user with her particular disabilities, keeping the specific user driven design at its core.

This involved creating an application in Unity, which made use of symbols, animations and games. Customisability features were included like icon size and background colour, to ensure it would fit the Sophia's needs. This objective was completed.

To assist the user to learn a chosen symbol set within this application, by an engaging means. In order to complete this objective, serious game theory was used to engage the user. This involved implementing a friendly onscreen character, which would be with the user throughout the application. The application was staged into word games and platform games using symbols from an appropriate symbol set. This symbol set was also selected with care. This objective was completed.

To capture data from the application and store it.

This involved creating a database for storage, and a server which connected to the application. Whenever the user completes a level, data is passed on how long it took and whether they completed it successfully or not. This data is successfully captured and stored, completing this objective.

To create an interface for Orlagh to access feedback.

This objective was easier than other aspects, due to the simplicity of its design. PHP, Bootstrap and HTML was used to create a web interface that the user could log into. FusionCharts was used to generate charts displaying recent user data, to offer feedback to Orlagh. This objective was also achieved.

Having completed these objectives, and receiving good user feedback, the overall goal is in my opinion, achieved successfully.

## 7.2. What I learned

I learned a lot throughout the course of this project. I learned about the struggle that users with learning disabilities face every day. How they require special treatment and attention, and what does and does not work when teaching them vocabulary.

I also learned about the importance of user centric design. It should be the key focal point of application design across the board, with few exceptions. It is definitely important, when designing systems specifically for users with disabilities such as Sophia.

Finally, I learned about the importance of regular meeting with project users and managers. These have major benefits in terms of project management and maintaining direction throughout a project. This direction is key when trying to reach goal completion.

## 7.3. Future Work

In the future, I would like this project to be built upon by other students. Having had more time to work on the project, there are many other features which would have been interesting and beneficial to implement.



Firstly, the application could be targeted to multiple users or a larger user group, as opposed to a single user. This would require more contact with other users, and potentially catering to a broader range of disabilities. For example, colour blindness is a potential additional disability. It would also be interesting to try and include a broader age bracket for this user group. These factors would extend the applications reach of usefulness.

By having more than one target user of the application, connectivity between multiple users in the system would be a possibility. This would allow children to play together, and potentially compete. By being able to play together, that would create a social aspect to the game, which would be very beneficial to users like Sophia.

Another aspect of this application I would like to take further, would be porting the application to IOS from Android. Apple devices are more user friendly and may suit the target user better due to her experience with them. Since the project application was developed in Unity, it can be built to an IOS device quite easily, as long as they have a developer license and access to Apple hardware.

I would also have liked to integrate this project with the other projects being created for Sophia. As it stands, the projects integrate on the basis of using the same symbol set, giving her a good basis for expanding her vocabulary. However, having the different project systems communicate, or even combine via services, would be an interesting avenue to explore.

These are just some of the recommendations and suggestions for how the work on this project can be improved on continued into the future. I feel this project has potential to be carried forward, whether it be by me or a future student.

#### 7.4 Personal Statement

This project brought me completely out of my comfort zone. I came into the project with no previous experience developing for people with disabilities. As a result of the project however, I have developed a far greater understanding of learning disabilities, and the complexity of teaching children using symbol systems.

Many children suffer from learning disabilities, and would benefit from the project system, which has become a strong motivator for me. There are solutions out there, but I feel they require a more fun and engaging approach. This is where I feel my project works well for the particular user.

In reflection, to me this project was humbling experience. It taught me about Sophia's struggles with language learning, and how she requires alternative approaches when it comes to learning. I am glad I was given the opportunity to potentially help her by utilising my skills, and as a result I feel I have grown as an individual.

In one sense, coding took a back seat, and the user became the priority. This is something that I have somewhat overlooked in the four years at DIT. The system may not be as complex as creating web services, or algorithms in data analytics, but it building it required careful design consideration and substantial feedback.

## 8. Bibliography

### Background Research

American Speech-Language-Hearing Association (2014) Communication Bill of Rights. Available at: <https://aaslanguagelab.com/files/communicationbillofrights.pdf> (Accessed: 24 November 2015).

Autism Speaks Inc (2012) Treatment for associated psychiatric conditions. Available at: <https://www.autismspeaks.org/what-autism/treatment/treatment-associated-psychiatric-conditions> (Accessed: 25 March 2016).

Baker, B. R. (2009) Minspeak™ History. Available at: <http://www.minspeak.com/HistoryofMinspeak.php#.VIRNyXbhCUk> (Accessed: 24 November 2015).

Bates, L. (2013) '5 Real Benefits of Using Animation in the Classroom', Tools, 6 December. Available at: <http://www.fractuslearning.com/2013/12/06/animation-in-the-classroom/> (Accessed: 16 November 2015).

Björn Schuller (2012) ASC-Inclusion – Integrated Internet-based Environment for Social Inclusion of Children with Autism Spectrum Conditions (ASC) - Geniiz. Available at: [http://geniiz.com/?page\\_id=240](http://geniiz.com/?page_id=240) (Accessed: 31 March 2016).

Dyslexia Association of Ireland (no date) General Information about Dyslexia. Available at: <http://www.dyslexia.ie/information/general-information-about-dyslexia/definitions/> (Accessed: 21 November 2015).

Dyspraxia Association of Ireland (2015) How to Recognise Dyspraxia | Dyspraxia association of Ireland. Available at: [http://www.dyspraxia.ie/whatisdyspraxia\\_recognise](http://www.dyspraxia.ie/whatisdyspraxia_recognise) (Accessed: 24 November 2015).

Expressive Solutions, E. S. (2013) ArtikPix on the App store. Available at: <https://itunes.apple.com/us/app/artikpix/id383022107?mt=8&ign-mpt=uo%3D8> (Accessed: 24 November 2015).

Foundation for People with Learning Disabilities (no date) Communicating with and for people with learning disabilities. Available at: <http://www.learningdisabilities.org.uk/help-information/learning-disability-a-z/c/communication/> (Accessed: 15 November 2015).

Irish Society for Autism (2012) What is autism. Available at: <http://autism.ie/what-is-autism/> (Accessed: 28 March 2016).

Johnson, D. J. (2015) Helping young children with learning disabilities at home | LD topics. Available at: <http://www.ldonline.org/article/5880/> (Accessed: 7 December 2015).

Kulman, R., Watkins, L. and Carlson, A. (2012) Video games and learning disabilities. Available at: <http://learningworksforkids.com/2012/07/why-video-games-are-good-for-kids-with-learning-disabilities/> (Accessed: 24 November 2015).

Le Normand, M.-T., Vaivre-Douret, L., Payan, C. and Cohen, H. (2000) 'Neuromotor development and language processing in developmental Dyspraxia: A follow-up case study', *Journal of Clinical and Experimental Neuropsychology*, 22(3), pp. 408–417. doi: 10.1076/1380-3395(200006)22:3;1-v;ft408.

Makaton (2015) About Makaton. Available at: <http://www.makaton.org/aboutMakaton/> (Accessed: 1 December 2015).

Molnar, M. and Sebastian-Galles, N. (2014) 'The roots of language learning: Infant language acquisition', *Language Learning*, 64(s2), pp. 1–5. doi: 10.1111/lang.12073.

NJC (1997) National joint committee for the communication needs of persons with severe disabilities (NJC). Available at: <http://www.asha.org/NJC/> (Accessed: 12 March 2016).

Newhall, P. W. (2012) Language-based learning disability: What to know. Available at: <http://www.ldonline.org/article/56113/> (Accessed: 24 November 2015).

Nielsen, J. (1995) 10 Heuristics for user interface design: Article by Jakob Nielsen. Available at: <http://www.nngroup.com/articles/ten-usability-heuristics/> (Accessed: 7 December 2015).

Patino, E. (2014) Understanding Dyspraxia. Available at: <https://www.understood.org/en/learning-attention-issues/child-learning-disabilities/dyspraxia/understanding-dyspraxia> (Accessed: 24 November 2015).

Prentke Romich Company (2008) 'What's Important in AAC?', (January), .

Prentke Romich Company (no date) Core Vocabulary Studies and Core Word Activities. Available at: [http://www.patinsproject.com/trainop\\_files/BethA1.pdf](http://www.patinsproject.com/trainop_files/BethA1.pdf) (Accessed: 24 November 2015).

Prentke Romich Company (no date) Free Resources - Teaching Resources. Available at: <https://aaclanguagelab.com/resources/free> (Accessed: 3 November 2015).

Prentke Romich Company (no date) LAMP Low Tech Board. Available at: <https://aaclanguagelab.com/files/lamplowtechbackup.pdf> (Accessed: 24 November 2015).

Prentke Romich Company (no date) PRC Unity Low Tech Board. Available at: <https://aaclanguagelab.com/files/prcunitylowtechboard.pdf> (Accessed: 24 November 2015).

ReadingByPhonics (no date) How to teach a 5 year old to read, write, and spell. Available at: <http://www.readingbyphonics.com/early-start/5-year-old-read-write-spell.html#.VmTWa3bhCUk> (Accessed: 7 December 2015).

Rooney, P. (2012) 'A theoretical framework for serious game design', *International Journal of Game-Based Learning*, 2(4), pp. 41–60. doi: 10.4018/ijgbl.2012100103.

Rowland, C. and Schweigert, P. (2000) Tangible Symbol Systems. Available at: [https://www.osepideasthatwork.org/toolkit/instpract\\_tan\\_sym.asp](https://www.osepideasthatwork.org/toolkit/instpract_tan_sym.asp) (Accessed: 24 November 2015).

Russpuppy (2015) Toddler robot – Android Apps on Google play. Available at: <https://play.google.com/store/apps/details?id=russh.toddler.robot> (Accessed: 24 November 2015).

Saffran, J. (2014) 'Sounds and meanings working together: Word learning as a collaborative effort', *Language Learning*, 64(s2), pp. 106–120. doi: 10.1111/lang.12057.

Smartbox (2016) Grid 3. Available at: <https://thinksmartbox.com/product/grid-3/> (Accessed: 2 April 2016).

Statistics Canada (2006) 'Participation and Activity Limitation Survey' . .

Taylor, M. (2013) Hardware v. User experience?. Available at: <http://www.onesheep.org/blog/hardware-v-user-experience/> (Accessed: 1 April 2016).

Uttal, D. H. and DeLoache, J. S. (2006) Learning From Symbolic Objects - Observer Vol. 19, No.5. Available at: <http://www.psychologicalscience.org/index.php/publications/observer/2006/may-06/learning-from-symbolic-objects.html> (Accessed: 10 November 2015).

VanTatenhove (2005) Common words - the center for AAC and autism. Available at: <http://www.aacandautism.com/common-words> (Accessed: 29 October 2015).

Vinson, K. (no date) Core Vocabulary. Available at: [http://www4.esc13.net/uploads/low\\_incidence/docs/BTH2013/Friday/Vinson\\_CoreVocabulary.pdf](http://www4.esc13.net/uploads/low_incidence/docs/BTH2013/Friday/Vinson_CoreVocabulary.pdf) (Accessed: 9 November 2015).

Widgit Software (no date) Widgit Symbols. Available at: [https://www.widgit.com/symbols/about\\_symbols/widgit\\_symbols.htm](https://www.widgit.com/symbols/about_symbols/widgit_symbols.htm) (Accessed: 18 November 2015).

Williams, D. (2010) Irish autism action - What is autism? Available at: <http://www.autismireland.ie/about-autism/what-is-autism/> (Accessed: 12 March 2016).

n2y (no date) SymbolStix. Available at: <https://www.n2y.com/products/symbolstix/> (Accessed: 24 November 2015).

## Technologies Researched

Android (no date) OpenGL ES. Available at: <http://developer.android.com/guide/topics/graphics/opengl.html#manifest> (Accessed: 7 December 2015).

Ashokan, A. (2015) Android is a better platform than iOS, know why?. Available at: <http://gadgetsngaming.com/2015/08/04/android-is-a-better-platform-than-ios-know-why/> (Accessed: 1 December 2015).

Bostock, M. (no date) D3.js - data-driven documents. Available at: <http://d3js.org/> (Accessed: 1 December 2015).

Colangelo, A. (2014) 'Google cloud vs AWS: A comparison - cloud academy Blog', Amazon Web Services, 30 October. Available at: <http://cloudacademy.com/blog/google-cloud-vs-aws-a-comparison/> (Accessed: 3 December 2015).

Coyle, P., McNulty, E., HOrlaghh, Poughia, E. and Krishnakumar, A. (2014) SQL vs. NoSQL- what you need to know. Available at: <http://dataconomy.com/sql-vs-nosql-need-know/> (Accessed: 1 December 2015).

FusionCharts (2015) JavaScript charts for web, mobile & Apps. Available at: <http://www.fusioncharts.com/> (Accessed: 15 March 2016).

ITX Design (no date) MySQL vs oracle. Available at: <https://itxdesign.com/mysql-vs-oracle/> (Accessed: 1 December 2015).

Krill, P. (2014) Why developers love and hate PHP. Available at: <http://www.infoworld.com/article/2852329/php/reasons-for-developers-to-love-hate-php.html> (Accessed: 1 December 2015).

Larson, R. (2012) JQuery: The good, the bad & the ugly. Available at: <http://www.webdesignerdepot.com/2012/09/jquery-the-good-the-bad-and-the-ugly/> (Accessed: 1 December 2015).

M, N. (2015) Amazon AWS vs Google cloud platform vs Microsoft azure: Which public cloud is best for you?. Available at: <http://dazeinfo.com/2015/05/22/amazon-aws-google-cloud-microsoft-azure/> (Accessed: 1 December 2015).

Microsoft (2016) What is Azure—the best cloud service from Microsoft | Microsoft azure. Available at: <https://azure.microsoft.com/en-us/overview/what-is-azure/> (Accessed: 1 April 2016).

Monus, A., Rocheleau, J., Ranjit, P. and Agus (no date) Beginner's guide to iOS development: The interface – part I. Available at: <http://www.hongkiat.com/blog/ios-development-guide-part1/> (Accessed: 1 December 2015).

Otto, M., Thornton, J. and contributors, B. (2015) Designed for everyone, everywhere. Available at: <http://getbootstrap.com/> (Accessed: 3 April 2016).

Rahman, S. F. (2015) The 15 best JavaScript charting libraries. Available at: <http://www.sitepoint.com/15-best-javascript-charting-libraries/> (Accessed: 1 December 2015).

Rubens, P. (2014) Is windows 8 development worth the trouble?. Available at: <http://www.cio.com/article/2377123/windows-8/is-windows-8-development-worth-the-trouble-.html> (Accessed: 1 December 2015).

Statsista (2015) Global market share held by tablet vendors from 2nd quarter 2011 to 3rd quarter 2015. Available at: <http://www.statista.com/statistics/276635/market-share-held-by-tablet-vendors/> (Accessed: 1 December 2015).

Wayner, P., InfoWorld, P. W. F. and 12, J. (2015) PHP vs. Node.js: An epic battle for developer mind share. Available at: <http://www.infoworld.com/article/2866712/php/php-vs-node-js-an-epic-battle-for-developer-mind-share.html> (Accessed: 1 December 2015).

Wilken, J., Puigcerber, P. V. and Erickson, K. (no date) JQuery vs. AngularJS: A comparison and migration Walkthrough. Available at: <https://www.airpair.com/angularjs/posts/jquery-angularjs-comparison-migration-walkthrough> (Accessed: 1 December 2015).

Unity - Game Engine (no date) Available at: <https://unity3d.com/unity> (Accessed: 7 December 2015).

### Design




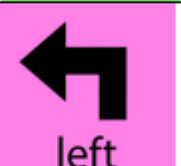

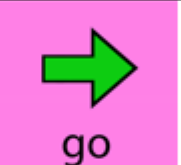












Apple (2016) IOS human interface guidelines: Designing for iOS. Available at: [https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html#//apple\\_ref/doc/uid/TP40006556-CH66-SW1](https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html#//apple_ref/doc/uid/TP40006556-CH66-SW1) (Accessed: 1 April 2016).

GameArt2D.com (2016) Free game assets. Available at: <http://www.gameart2d.com/freebies.html> (Accessed: 7 March 2016).

## 9. Appendix

### Appendix 1 – Custom built symbols

#### Custom built symbols and comparison

Comparisson of own symbols vs Orlaghs			
Orlaghs current set		Custom built	
			
			
			
			
Other symbols created			
			
			
			

## Appendix 2 – Test Case Template

### Test Case Template

<b>Test Case Number:</b>	1
<b>Test Case Name:</b>	Test for 'whatever'
<b>System:</b>	(Your project)
<b>Subsystem:</b>	(the part of the project)
<b>Created by:</b>	(Your name)
<b>Creation date:</b>	__ March 2016
<b>User:</b>	Orlagh (with/without Sophia)
<b>Test date:</b>	__ March 2016

#### Description:

This test case is designed to test the following...

Pre-Conditions:

- User must have....

Step	Action	Desired Response	Pass/Fail	Comment
1				
2				
3				
4				
5				
RESULT				

Post-Conditions:

- The user can now successfully... or something happened
- Expectations of positive result



Suggestions, feedback  
and additional  
comments:

