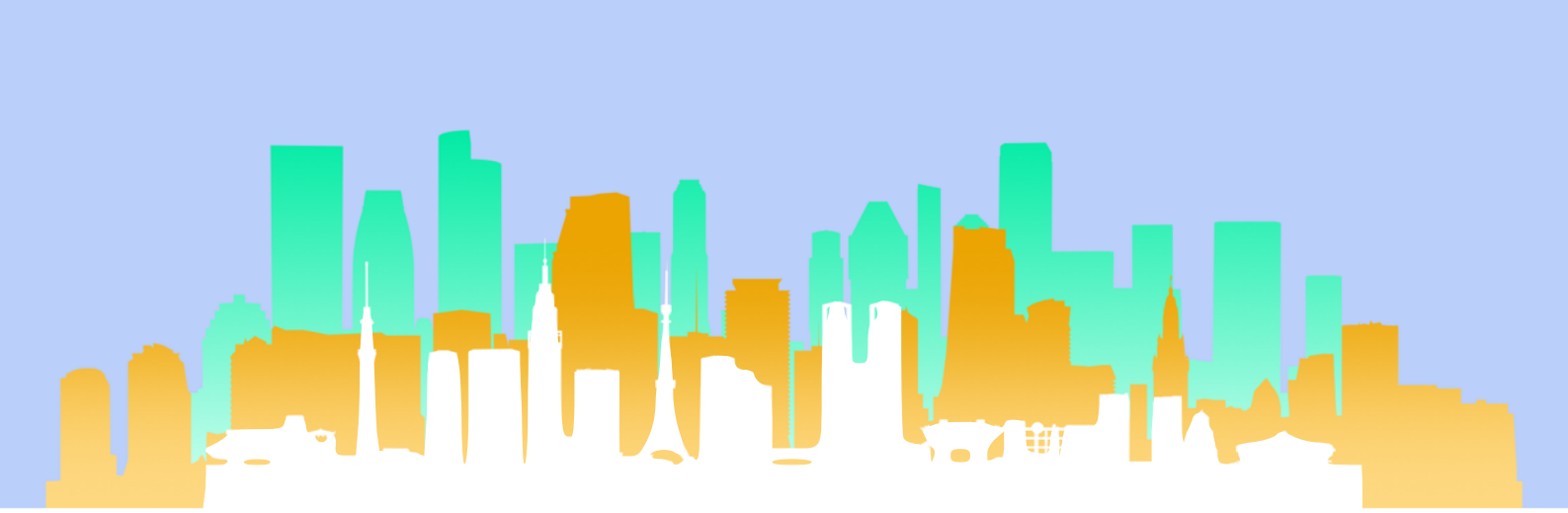


CODEGIRL VS THE HEXABUNNIES

Major Project Report

Course: Interactive Multimedia Design
Year: 4
Sian Finlay (B00612791)
Peer Group: 5; Mentor: Raymond Bond;

ACKNOWLEDGEMENTS

I would like to express thanks to those who help me throughout my final year project…

Raymond Bond - Mentor

I would like to thank Raymond Bond for having faith in my project, even during times of stress, and constantly providing advice on research and the functionality of the project.

George Moore - Module Co-ordinator

I would like to thank George Moore for his guidance lessons on module deliverables and recommendations of best practices and approach.

Peter Nicholl - Course Director

I would like to thank Peter Nicholl for making himself available to give advice and making learning opportunities accessible.

TABLE OF CONTENT

1. INTRODUCTION

1.1. OVERVIEW

To design and develop a web application which teaches young girls the basics of HTML and CSS, and gives an understanding of their uses. To do this through learning puzzles, part of an adventure game. The Adventure game will be set in a world of magical girls similar to; Sailor Moon **(Sailor Moon , n.d.)** and Winx Club **(Winx Club, n.d.)**.

The application is set up like an adventure game to motivate the user throughout the learning tasks. It gives the user a reason to complete the puzzle; talking to other creatures, saving the day and ultimately progressing the story. 'Gamification' **(Gamification, n.d.)** has been used in many other children's, and indeed adult, learning tools. Over the years since video games were first introduced, more and more people of all ages, have been playing games, to the effect that at people often understand the concept of gaming before learning. In a TED talk "Gaming can make a better world" by Jane McGonigal **(Jane McGonigal, Feb 2010)**, she talks about the benefits of learning through games, some of the points she makes are; reduced stress, deep focus and intense concentration, continued use of lessons after playing a game.

Taking examples from undergraduate experiences in Interactive Multimedia Design, the puzzles in the application will follow student learning of HTML and CSS, except worded for children and set up in a gaming format. Puzzles will come in two forms; fill in the missing code and choice based questions. During the beginning of the project experimentation will take place to find the best way to create the "fill in the missing code" puzzles with techniques such as; online text editor, text inputs or drag and drop. This will be laid out like a code editor so the user learns what a real coding environment is like. Following this idea through the story the user will read about hosting, facts about the World Wide Web, etc … in a simple, easy and concise way. This is where the questions will come in, asking the user if they remember what a character said, for example: "the password to the door is Tim Berners Lee".

As the user progresses and completes puzzles, their magical character gets to choose from new outfits. This magical girl is their avatar, their motivation to keep going. Create their avatar, give it a name(username), customise their clothes. This idea is inspired from the success of pet raising games such as Neopets **(Neopets, n.d.)**, and it's a genre that continues to see success in smartphones and tablet games.

1.2. AIM

To create a web application that teaches children aged 8-11 the basics of HTML and CSS, as well as giving them some basic web knowledge. It will do this through a magical girl themed adventure game using puzzles like 'fill in the missing code' and choice based questions. The hope is that it could be used in primary schools and by parents to inspire children to take up careers in web development and design or computer science.

1.3. OBJECTIVES

- Create templates 'Fill in the missing code' and choice based puzzles.
- Plan and create Adventure story, making sure puzzles can fit into the story.
- Plan and write puzzles.
- Plan and Design the illustrations and animations.
- Use Canvas to create user's avatar and allow for customisation.
- Create a login and database to save user information; username, password and avatar options.
- Create shareable awards such as badges for sections completed and/or scoreboard.
- Debug and test prototype.
- User test prototype.
- Refine and optimise application.

1.4. TARGET AUDIENCE

This application is for children aged 8-11. In the Northern Ireland school system they are P5, P6 and P7. At this age children are starting to get into more serious studies; thinking of their grades, what secondary school they want to go to and beginning to experiment with what they want to be in the future. They also begin to build a presence online with social media, "More than half of children use social media by the age of 10"- Daily Mail **(Daily Mail Reporter, 6/2/2014)**. In recent years there has also been a push to get children to start learning code. As the local software technology industry grows, more jobs need filled **(HM Treasury, Department for Education, 4/2/2014)**.

There are several sites online with a focus on teaching women to code, however a lot of these focus on older teenagers and adults, and are only available to those who can travel to boot camps or lectures. Even courses for young girls are limited by travel and money. Few parents are willing to send younger girls to a distant location, at considerable expense, for their first chance to learn computer code. One example of these resources is the Girls Learning to Code program **(Ladies Learning Code, n.d.)** based in Canada. A team of female programmers and developer go to main cities with events such as workshops, camps, after school programs and field trips in order to educate young girls in coding. What they need is access to an online tool they can use wherever they are at school or home with a focus on younger girls. This application will accessible anywhere there is an internet connection, they will learn code with their 'female' magical 'girl' avatar.

1.5. COMPETITION

The Application will compete with other applications such as; eraseallkittens.com **(Drum Roll, n.d.)[Appendix 1.A]**, crunchzilla.com **(Geeky Ventures, 2015)[Appendix 1.B]**, scratch.mit.edu **(Lifelong Kindergarten Group, n.d.)[Appendix 1.C]**, Hopscotch(native application) **(Hopscotch, n.d.)[Appendix 1.D]**, CodeQuest(native application) **(Codarica Inc., n.d.)[Appendix 1.E]** …

Eraseallkittens.com, is a demo online platform game which teaches kids HTML and CSS. It uses an online text editor to help the user complete levels. This application shows the user what the code does on a website and gives audio directions to follow. However while the instructions were great they seemed separate from the game and as a result, they were hard to follow. If the instructions could be displayed in character's speech bubbles, keeping the player's focus on the game, they would be less distracting.

Crunchzilla.com, teaches kids Javascript using it's 'Code Monster'. The application is very simple with a text editor, code display and colourful monster. Apart from the monster the user tends to lose interest in the application due to the lack of challenge. However its simple UI makes it easier to focus on the task.

Scratch.mit.edu and Hopscotch both use the same method of teaching kids programming. Using colourful drag and drops, they get the players used to how programming acts. Both are successful applications however they do not teach the user a specific programming language, only the basic concepts behind them.

CodeQuest, teaches kids HTML and CSS through an adventure game. It's is however a native iPad application and uses games to show sections of code and not actually write it. The application also allows the user to create a simple web page to share, and introduces web jargon like "publish a website".

1.6. WORK UNDERTAKEN

During this project there will be a need to learn new technologies to be used in the application's development. If these technologies do not meet requirements then alternative solutions will have to be implemented. Research will be needed to search for the best technologies and possible alternatives.
These technologies will be used to fulfil the Requirements Specification, defined by user surveys and professional user research already available online.
The application shall then be tested and refined, until completion.

1.7. REPORT OVERVIEW

This report documents the stages taken within the project; from concept, research, testing, design to development of the application. Each stage adds their own challenges to the project, and improves the overall quality of the finished application.

2. CONCEPT DEFINITION AND TESTING

2.1. IDEA GENERATION

The idea for this project was inspired the TED Talks video "Gaming can make a better world" by Jane McGonigal **(Jane McGonigal, Feb 2010)**. The video talks about how to improve everyone's ability to learn, through the use of games. In the video Jane McGonigal talks about how this generation's children have grown up exposed to gaming before they even begin their school years. This fact accompanied by the growing need to get more people, especially females, into the computing industry generated the idea to create an web based game which teaches young girls to HTML and CSS using puzzles.
Before the final idea, there was three initial ideas considered. The third idea was then modified to create final idea:

2.1.1. Dog Training Simulator

A course-based Web Application using Unity3d **(Unity, n.d.)** to teach first time dog owners about dog training through small simulation games. Most owners face the problem of reading through dull dog training books and watching staged TV shows without any way to safely practice the methods shown. This leads to problems in real world training and owners having to spend thousands on professional dog training to undo their mistakes.

The Application aims to use simulation games to teach the user methods of training before getting a dog. These will be separated out into sections such as; introducing dog to home, toilet training, walking, etc… Before beginning a game the user will be given clear and simple instructions to follow during a game, for example a game for "toilet training" could inform the user to click on a dog whilst "peeing" awarding them points for each dog they catch. Using this form of gamification **(Jane McGonigal, Feb 2010)** and repetition in a game will help the user learn the method.

Before starting the Application user will sign up to allow their progress during the course to save, using PHP and MySQL Databases. This progress will be displayed using jQuery and display an inspirational messages as the user proceeds with the course.

This information will also allow the user to discuss and ask questions on a forum with other users after finishing the course for further learning, this can also be used during the course as a helpful guide.

Other such features could include a "success" page which users who have finished the course can upload pictures of their new dog and give feedback to the Application to thank the course or say where improvements could be made. Another feature could add a "useful links" page to other dog training sites or recommended dog equipment.

The Application's training method would be inspired by professional dog trainers such as Cesar Millan and Jan Fennell and would require a qualified dog trainer to certify the techniques being taught.

2.1.2. Trans-Genre

A Recommendation Web Application in which users are shown recommendations of other types of media; Films, TV-series, books, games, poetry, etc … The recommendations are based on tags attached to an item which is Recorded in a MySQL database and using either PHP or Node.js to link other items with similar tags. For example if the user watched “Mad Max: Road Warrior” they will get recommendations from films, books, games… based on the film being in the “post-apocalyptic”, “action” and “racing” genres. A user would not only see the other Mad Max films and similar films such as “Tank Girl” but books such as “The Road” and games like “STALKER”.

On other sites like Amazon.com or Steam, user recommendations are limited to similar medias such as films recommending similar films, games, similar games. Trans-genre will focus on the genre of the item not the type of media, using the type of media to only to sort into lists the recommended books, games, films, etc.

Trans-genre will also be user-populated. Creating an account with the application will allow the user to add new items with genre tags. If the item exists in the database the new tags will be added to the existing item or used to increase the rank of an item’s tags, which will be collected in a variable. This increase in rank will increase the rate of the item’s appearance within that tag. Items may also be rated by the user. Every item added by the user will be displayed on their profile page, like a collection.

On the of other hand, user could maybe vote on an item’s tags to increase its rank. For example, if “Musical” is added to “Mad Max: Road Warrior” they could vote “no” and decrease the occurrence of the film under the “Musical” tag.

Trans-genre will keep a record of items the user has added and display them on the profile page, inspiring the user to watch, play or read more to add to their collection. Some user’s may even acquire legendary status for their recommendations and ability to spot titles in certain genres.

2.1.3. Learn with Robots and Adventure

A Web-based Interactive Adventure game in which kids and young adults learn to code in order to progress the story. The user will have to solve and answer questions on HTML, CSS and possibly more programming languages to continue the game’s story. Each user will get a robot character, which acts as their avatar during the game.

There has been a growing need to get kids and young people to learn to code. However the resources out there are too technical, involve a lot of reading and other applications to practice. Kids are struggling to learn, most give up from boredom. Kids want to learn, and one way is through video gaming. The gamification of learning takes away the stress and boredom that regular conventional learning creates. in addition, learning through games increases the ability to remember what was learnt.

As users progress their robot “levels up” receiving more body parts (HTML) and more design and colour choices (CSS). This robot will be created using Canvas which receives the user’s personal robot choices from a MySQL Database and selects from a library of images and functions. There is also the possibly, if more

programming languages, such as JavaScript, PHP or jQuery, are added; that these could add an animation to the user's robot, give the robot a room, make the robot react to the mouse…

The adventure puzzles will be either/both "fill in the code" or multiple choice. For "fill in the code", it will be similar to Codecademy's online text editor which may use Node.JS or PHP. Another way this could be created is using form inputs. The multiple choice could be stored using JSON, which only lets the user progress with the right answers. JSON could also be used to store the game's story, which will be called upon as they progress.

Other possible features could allow the user to share their progress on social media. Since the game's user will be young, the addition of randomised usernames with a robot theme could be used to protect the user and their data.

2.2. CONTEXTUAL RESEARCH

There is a high demand from businesses, government and high education to employ more people with relevant skills in computer science and ICT across the UK. What is currently taught in schools is not enough, or does not cover the right subjects, to further learning in computer science.

> "Universities want to reverse the decline in applicants for computer science courses. Gaming companies want more programmers. The government wants more high-tech startups. Manufacturers want trainees who can design embedded systems. And head teachers want bigger budgets for even more computer labs. And so on."
>
> Why all our kids should be taught how to code, John Naughton guardian.co.uk, Saturday 31 March 2012 20.15 BST**(John Naughton, 31/3/2012)**

> "Instead of educating children about the most revolutionary technology of their young lifetimes, we have focused on training them to use obsolescent software products."
>
> Why all our kids should be taught how to code, John Naughton guardian.co.uk, Saturday 31 March 2012 20.15 BST**(John Naughton, 31/3/2012)**

However it's not enough to improve education in secondary schools, we need to be teaching computer science at a primary school level. The world has changed a lot in the past decade; social networking is a part of our daily routine, people of all ages have email addresses and smart phones which automatically do tasks which used to take much more time and effort.

> "Starting in primary school, children from all backgrounds and every part of the UK should have the opportunity to: learn some of the key ideas of computer science; understand

computational thinking; learn to program; and have the opportunity to progress to the next level of excellence in these activities."

Why all our kids should be taught how to code, John Naughton guardian.co.uk, Saturday 31 March 2012 20.15 BST[**(John Naughton, 31/3/2012)**

The planned application will solve this problem by providing a tool to learn HTML and CSS through adventure game. Learning code will contribute to the game's story. This is a form of gamification is proven to help people learn and is a tool children are familiar with.

"Games, of course, help put people back in control. Real gameplay is always by definition voluntary; it is always an exercise of our own freedom. Meanwhile, progressing toward goals and getting better at a game instills a sense of power and mastery."

Reality Is Broken, Jane McGonigal (also did a TED Talks about gamification)**(Jane McGonigal, 2011)**

2.2.1. Narrowing the user base

There is an obvious need for more people with computer science and ICT skills, this has to start with children in primary schools. However there is another problem the computer science and ICT faces, and that is the lack of females in the industry. Since 1990/2013 the percentage of women in computer and mathematical occupations has fallen from 35% to 26%. We need to encourage more women to join the industry as programmers and developers. As Shown in **[Figure 2.A]**

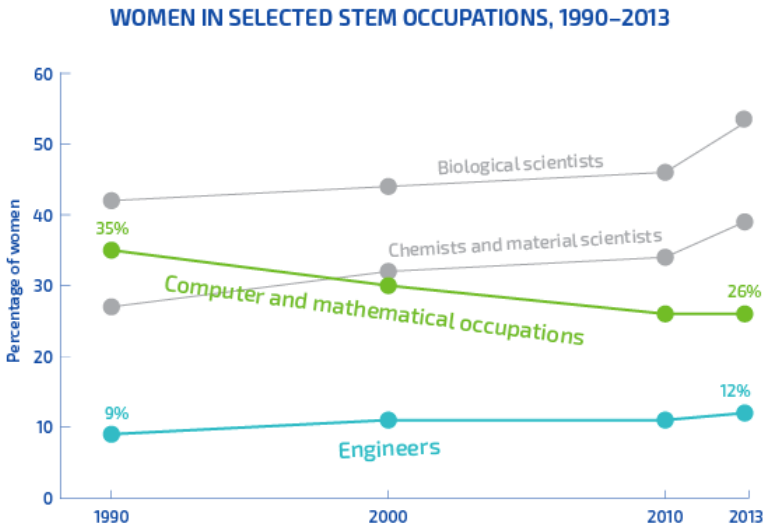


Figure 2.A | Solving the Equation: The Variables for Women's Success in Engineering and Computing, American Association of University Women**(Corbett, Christianne, 2015)**

One of the reason why there is a lack of women is the stereotypes surrounding the industry:

"Computer science and engineering are stereotyped in modern American culture as male oriented fields that involve social isolation, an intense focus on machinery, and inborn brilliance."

Cultural stereotypes as gatekeepers: increasing girls' interest in computer science and engineering by diversifying stereotypes, **(Sapna Cheryan, Allison Master, and Andrew N. Meltzoff, 11/2/2015)**

We need to reverse this stereotype by creating learning tools that are cooperative, noncomplex and friendly which are good introductions to enter to industry before moving on to the more technical learning sites such as http://stackoverflow.com/ which can be imitating to for beginners.
Where this application will differ from other children's code learning tools by focusing on young girls (8-11). In order to do this it was decided to change part of the original concept of using robots as the user's avatar and background story, to Magical girls. Magical girls is a genre in Japan which portrays girls as the heroes of the story, however keeping with feminine theme there is makeup, frilly dresses, pastel colours … For examples of magical girls is; Sailor Moon, CardCaptor Sakura and Princess Tutu. In western media we have had our own version of magical girls; Powerpuff girls, Steven Universe, W.I.T.C.H.
Young girls of primary school age will be encouraged to learn code through gamification with their 'magical 'girl' avatar.

2.3. REQUIREMENT SPECIFICATION

Using data acquired from a survey and statistics from Kidsay Research article **(Renee Weber, 23/1/2015)**, Project's requirements were created using the Volere Requirements Specification Template **(James & Suzanne Robertson, 2016)**. In the survey, parents and relatives of young girls filled out 10 questions related to their children's schoolwork, interest in computing and personal interests. The survey was created on SurveyMonkey (https://www.surveymonkey.co.uk/r/XRKBXLW)**(Sian Finlay, 19/11/2015)**.

Requirements are useful to take into account both the needs of the user and project constraints. Using the Volere Template's to create different types of requirements; functional, non-functional, look and feel… Most of the requirements can be found in a table **[Appendix 2.A]**, here is some examples of the requirements:

2.3.1. Functional Requirement

Requirement#: 1

Description: The game shall contain puzzles and questions.

Rationale: To teach young children how to code HTML and CSS.

Fit Criterion: If the user can solve the puzzles and questions to ultimately learn how to write their own web.

Customer Satisfaction: 5

Customer Dissatisfaction: 5

Priority: 1

2.3.2. Non-Functional Requirement

Requirement#: 19

Description: Product shall be available to all primary schools.

Rationale: Product must work on primary school computer systems.

Fit Criterion: During user testing test the performance and accessibility on their computers.

Customer Satisfaction: 4

Customer Dissatisfaction: 4

Priority: 4

2.4. INITIAL UX DESIGN

For the creation of the initial UX design, techniques such as paper prototyping, wireframes and user personas were used. These do not require any coding.

2.4.1. User Personas

To understand the user better, fictional personas are created based on the information of users. This includes the survey used in the requirements and statistics from the Kidsay article. For the application there is actually two users; the girl and the parent. The young girl uses the site to learn whilst the parent needs access to ensure that the site is safe for their child. As shown below: The young girl **[Figure 2.B]** and The parent **{Figure 2.C]**

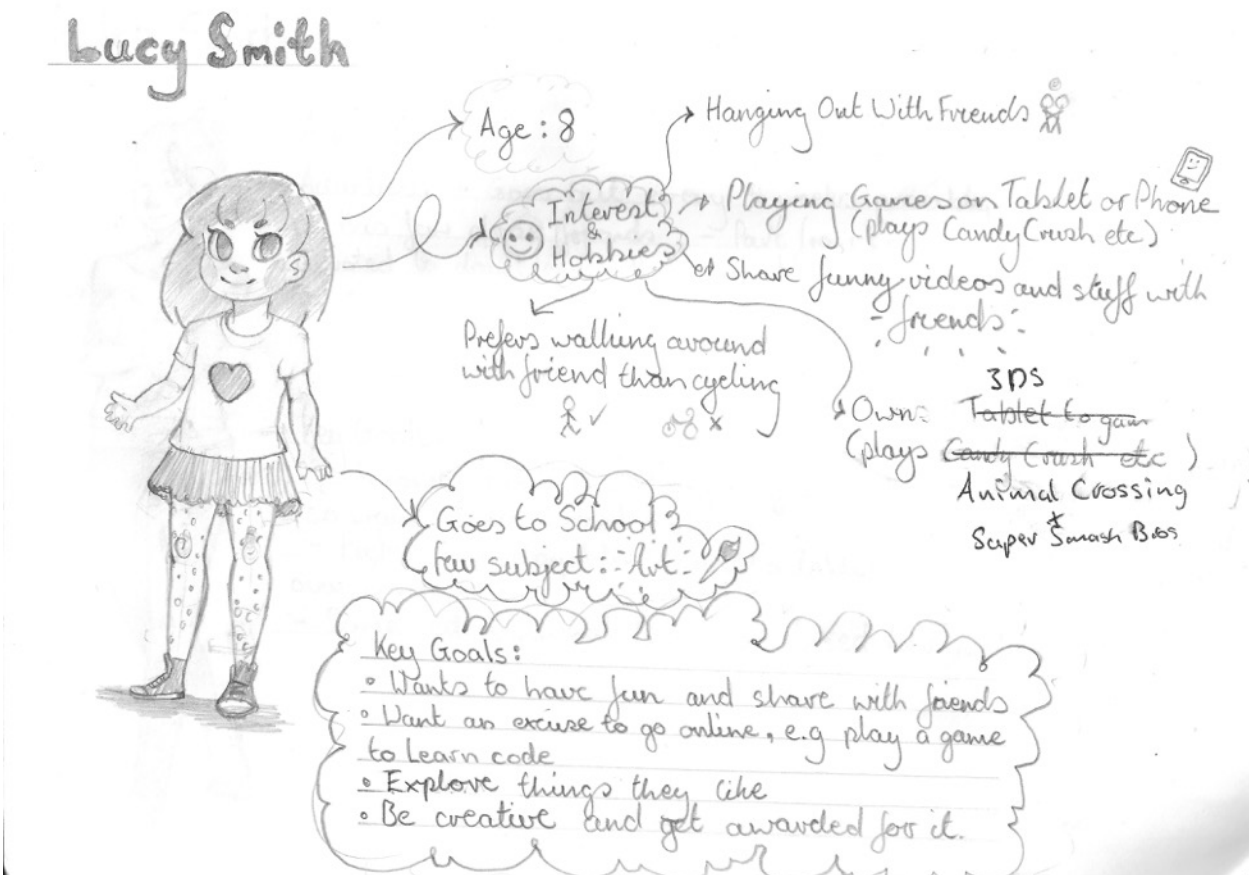


Figure 2.B | The Young Girl

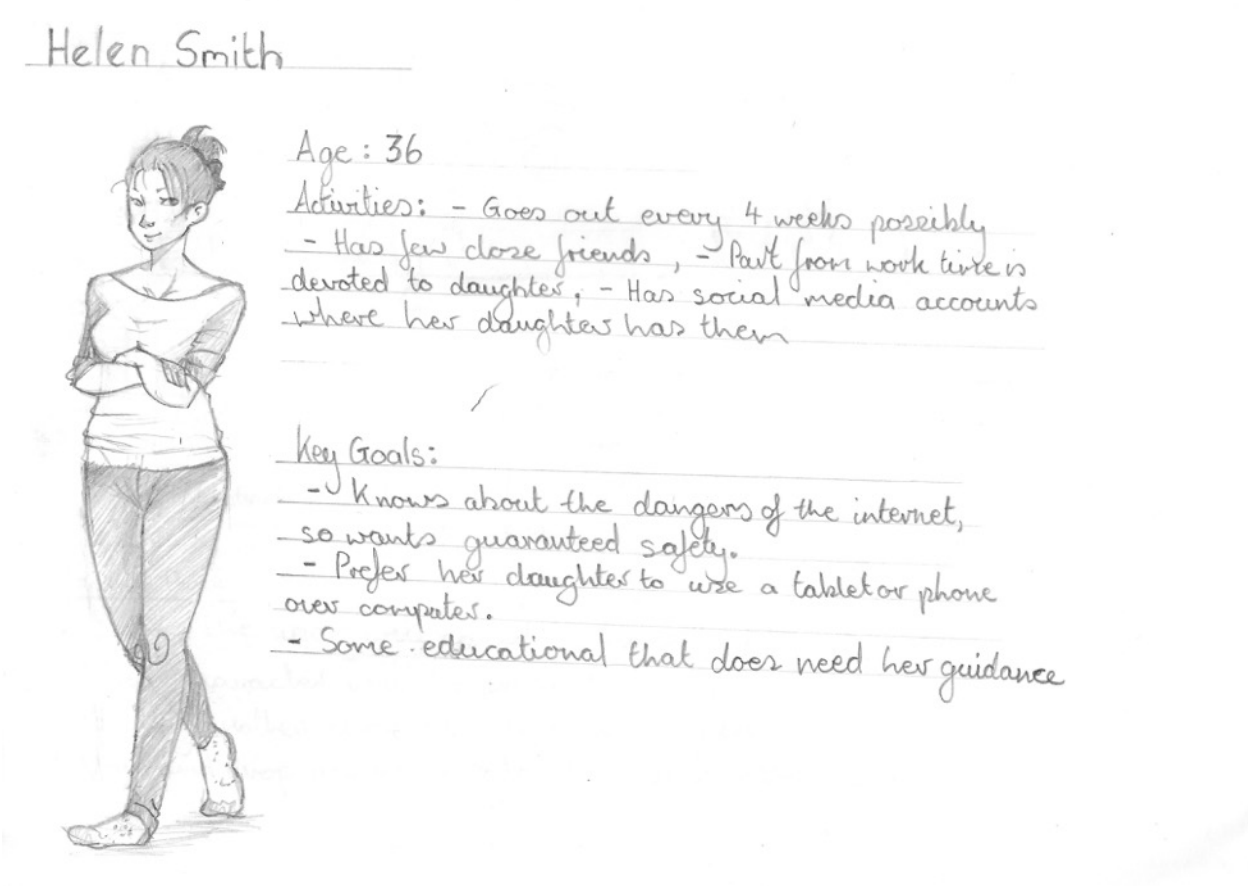


Figure 2.C | The parent

2.4.2. 6-Ups

6-Ups is a good ways of sketching up either a design problem, comparing designs or generating design ideas. This method was used to compare design structures of the Welcome page **[Figure 2.D]**, Main game page **[Figure 2.E]** and Customisation page **[Figure 2.F]** of the application.

Figure 2.D | Welcome page

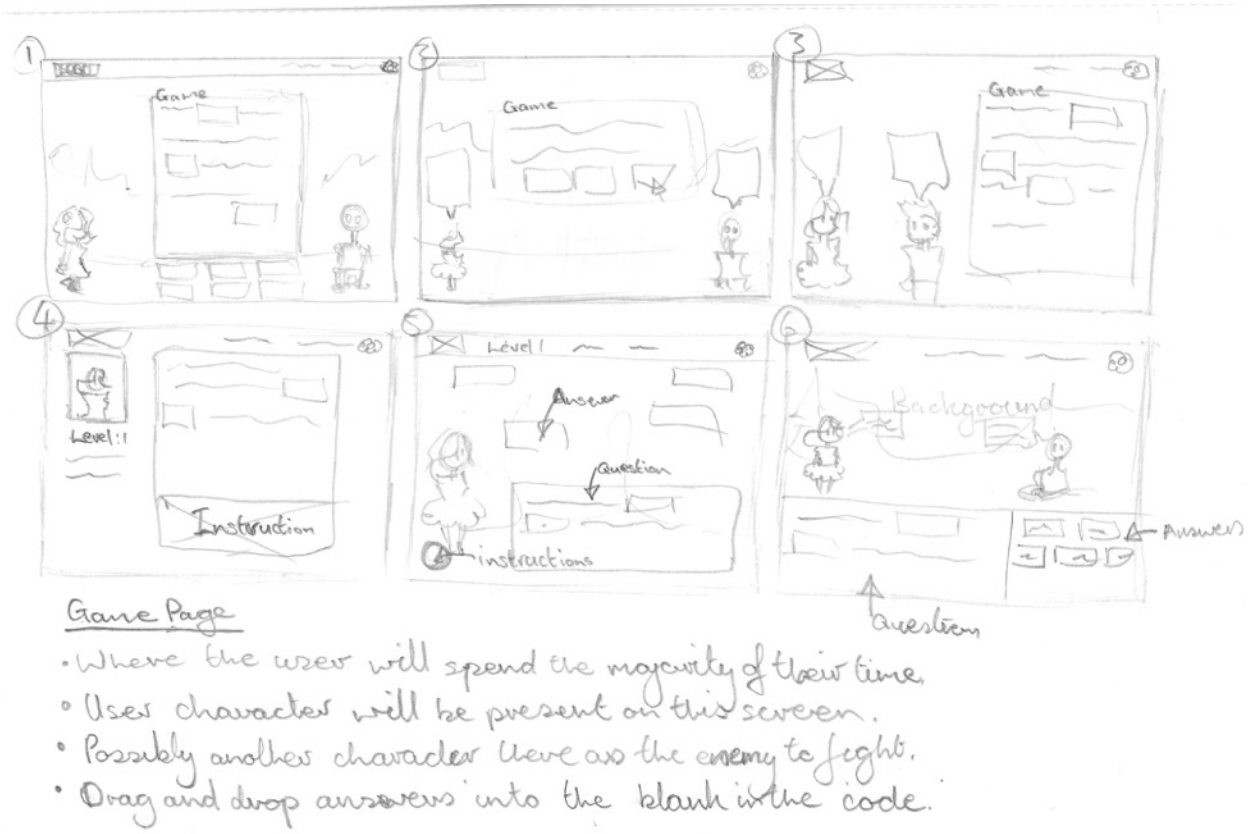


Figure 2.E | Main Game page

Figure 2.F | Customisation page

2.4.2. Wireframes

After choosing the best of the 6-up sketches, wireframes were created using Sketch **(Bohemian Coding, n.d.)**. Wireframes provide better clarity towards the structure of the design, including grid structure, px widths and heights … Welcome page **[Figure 2.G]**, Main game page **[Figure 2.H]** and Customisation page **[Figure 2.I]**

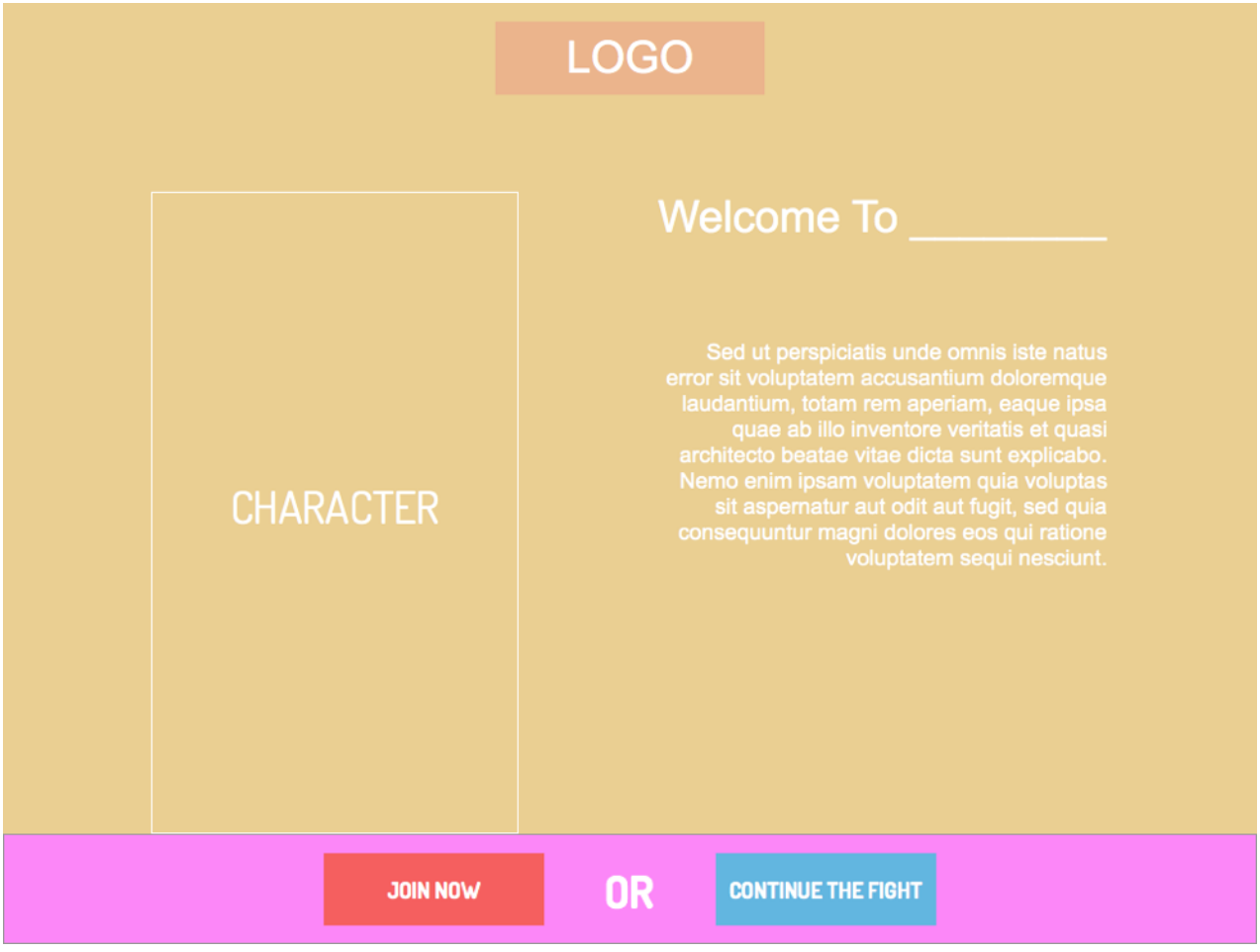


Figure 2.G | Welcome page

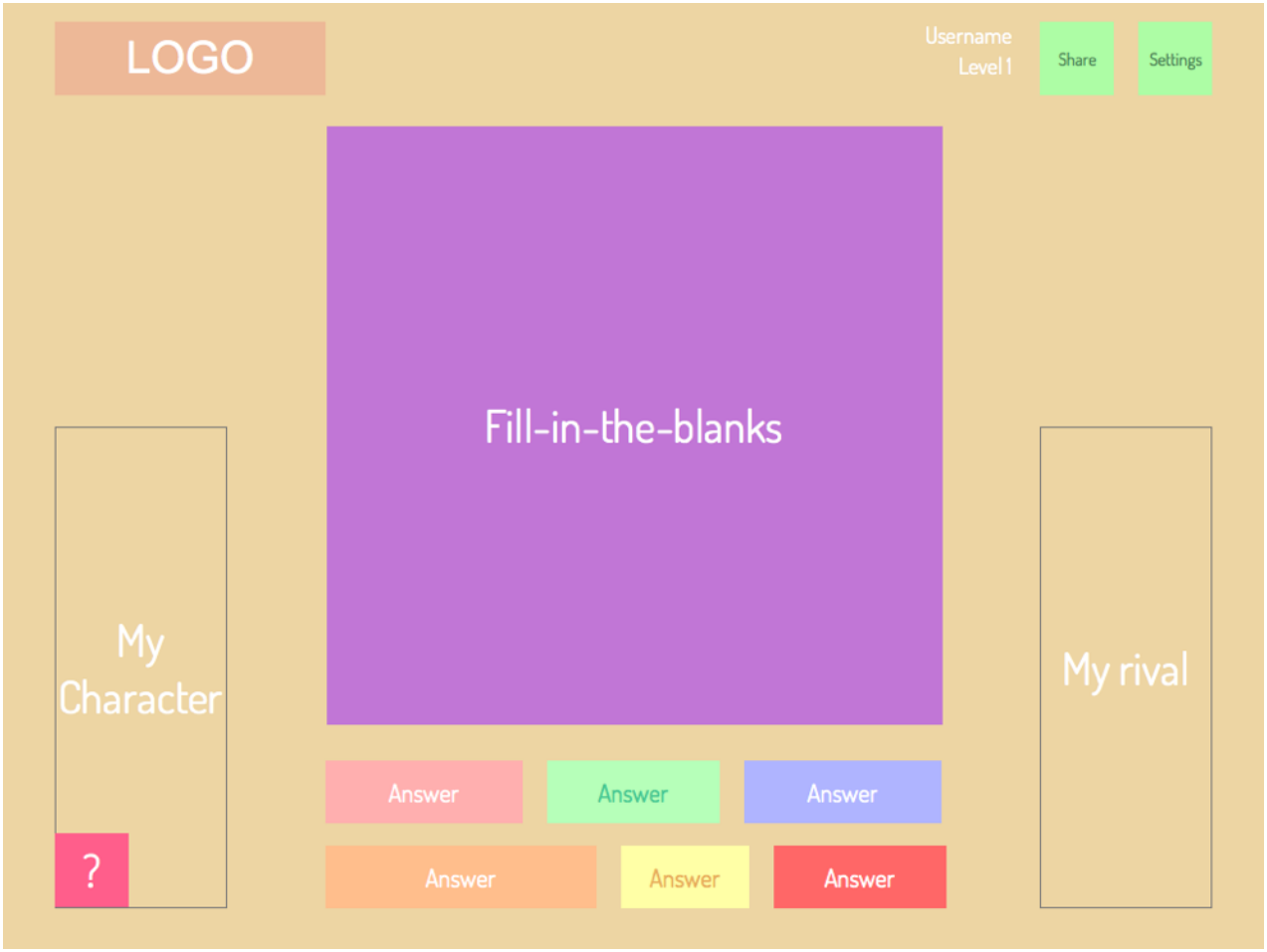


Figure 2.H | Main Game page

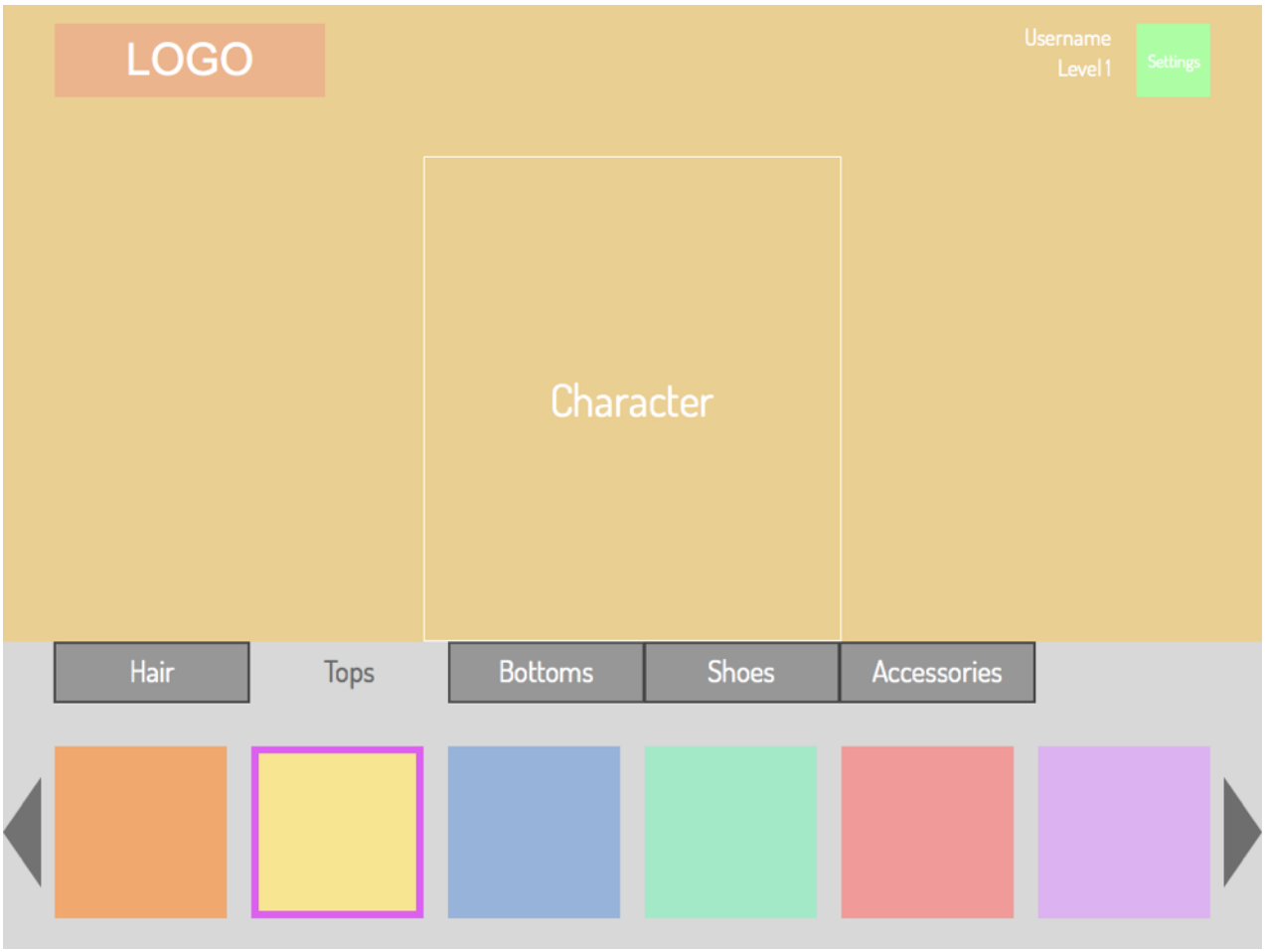


Figure 2.I | Customisation Page

2.5. INITIAL SYSTEM AND TECHNOLOGY REVIEW

Below are the initial system and technology selections for the application. After the feasibility testing these designs and choices will be refined, as several of these designs and technology choices have not been used before and will provide clear solutions to the system design afterwards.

2.5.1. System Design

The system design is used to help understand the structure, identify the elements in the system and their relationships. One way to do this is using the Client-server Model**{Figure 2.J]**. The model sorts the elements by client or server.

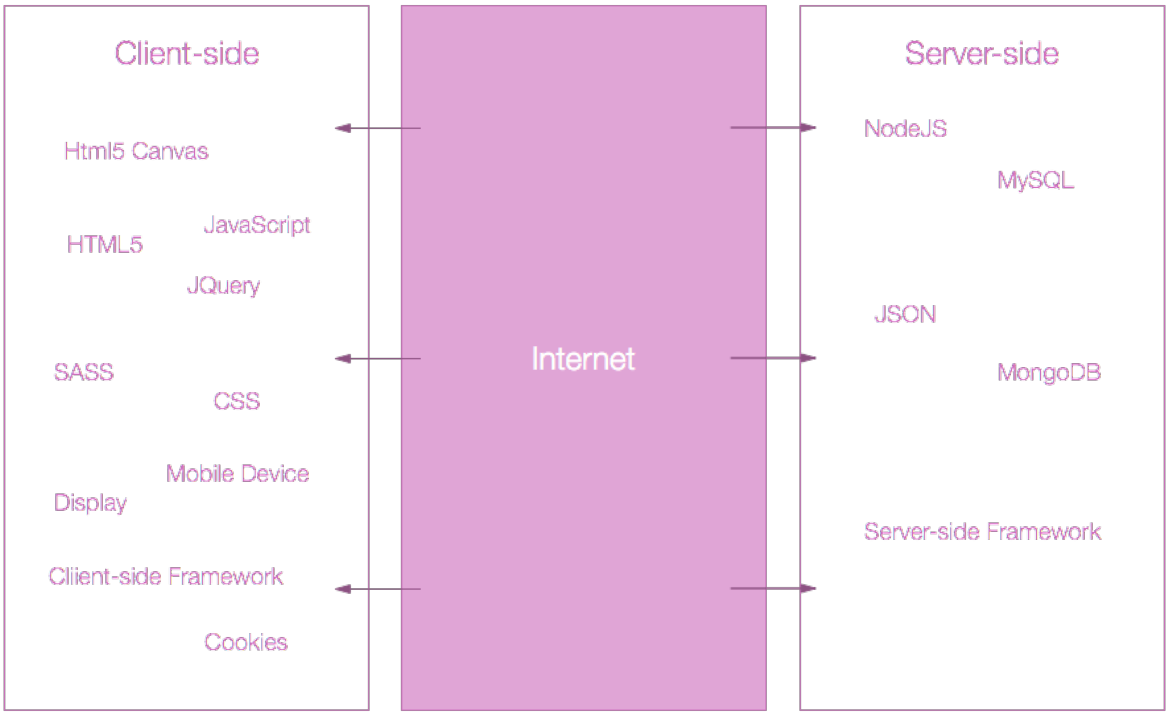


Figure 2.J | Client-server Model

2.5.2. Technology Review

The technologies being reviewed take into consideration the creation of the drag and drop game element, Character creation, login/signup, design and server-side.

EaselJS**(Grant Skinner, n.d.)**

EaselJS is a HTML5 Canvas framework which was built to have some similarities with Adobe Flash's ActionScript, it also makes working with Canvas easier. It's a popular framework, with many tutorials and a large community. It can be used to build a variety of canvas based applications including games, however there are alternatives that focus more on game creations that provide simpler solutions.

HTML Quintus**(cykod, 2012)**

Quintus is an alternative HTML5 Canvas framework built for creating games. The scripting is similar to many other gaming scripts, with a built in modular system to separate gaming functions. However there are few tutorials apart from on the Quintus site, and the community is small, having only a Google+ Community. Quintus focus on standalone Canvas page, it's error handling can get in the way of other scripting languages making it not ideal for this project.

NodeJS**(Joyent, n.d.)**

NodeJS is a server-side JavaScript environment, The use of NodeJS in web development has been on the rise for a few years now, it makes connecting to the client side Javascript much easier. NodeJS also has the benefit of it's package manager, NPM, which packages can simplify many server-side and some client side processes. However the University's server does not support NodeJS therefore using the environment is a big risk. After spending time using the environment however, with its benefits, it is a risk worth taking for the development of the project.

PHP**(Rasmus Lerdorf, 1994)**

PHP is a long standing server-side scripting language with many tutorials and support. The use of the PHP however is declining. With PHP you do not need to download extra packages like NodeJS, PHP controls everything. However more coding is needed with basic PHP to create the features a basic NodeJS has in just a few lines. For this project PHP will be used as a backup server-side script.

SailsJS**(Mike McNeil, 2012)**

SailsJS is a MVC NodeJS framework for build applications, it is good for people who come from a Ruby on Rails or PHP background. SailsJS can support any type of databases; MySQL, PostgreSQL, MongoDB… It comes with several very useful NodeJS packages such as Grunt and Sockets.io. It uses EJS files, which are

JavaScript template files, for front-end development. During the feasibility test both SailsJS and MeteorJS will be look at and decided on.

MeteorJS**(Meteor, n.d.)**

MeteorJS is another NodeJS application framework/platform, which comes with its own package manager which can be used to quickly build many regular features in an application such as Login/signup. MeteorJS's script can be very different from NodeJS standard script which can take a while to get use to, however it's built in features are very powerful. There is no need to link any CSS or Script files to the front-end files as they are added automatically using a few lines of script in the original files. During the feasibility test both SailsJS and MeteorJS will be look at and decided on.

2.6. FEASIBILITY TESTING

(link to prototype: https://scm.ulster.ac.uk/~B00612791/workspace/mp/prototype/prototype_clickheretoberedirect.html)

For the feasibility test of the application, a Risk Analysis was conducted to explore where the high risk areas of the application were. In order to conduct the test, one or more risks from the analysis were taken and developed into a functional prototype. If this prototype was successful then another risk could be tested, if it was unsuccessful then an alternative solution would be developed or the functionality in the risk could be dropped from the application.

The Risk analysis can be found in **[Appendix 2.B]**. The risk analysis uses the traffic light system, meaning that high level risk are marked as "red", medium level is "amber" and low level is "green". A number of these risks were taken into account when developing the functional prototype, these risks being; Drag and Drop, NodeJS with Heroku, NodeJS with either SailsJS or MeteorJS, Social Login and Sign up.

During the development of a functional prototype, the Meteor framework proved to be the best choice for this application. The reasons for this being that the framework automated and simplified a lot of the application processes such as social/regular login and file connections. This would become of greater use as during the development the drag and drop feature proved to be more difficult to produce, and will need additional time to be developed. Lastly, the functional prototype was placed on the Heroku Cloud Platform, using Git to install the platform's toolkit, also update prototype files, and a Meteor Build-pack, this prototype still currently maintained on the platform.

The Meteor test on the prototype followed the tutorial on the meteor website, https://www.meteor.com/tutorials/blaze/creating-an-app

2.7. METHODOLOGY

During the planning stages of this project, research was done to decide on the best methodology to take in order to help with the progression of the project. Listed below are some of the methodologies taken into consideration for this particular project.

2.7.1. Waterfall Model

The Waterfall method follows sequential path, like a waterfall, once one task is completed the end task is started without any review or iteration seen in other methodologies. It is a straightforward method but needs to have well defined requirements set out during the planning stage as there is no going backwards in this method. This was not a suitable method for this project as the ability to modify features based on the outcome of recently developed features is key to maintaining the quality of the application. Example shown in [Figure 2.K]

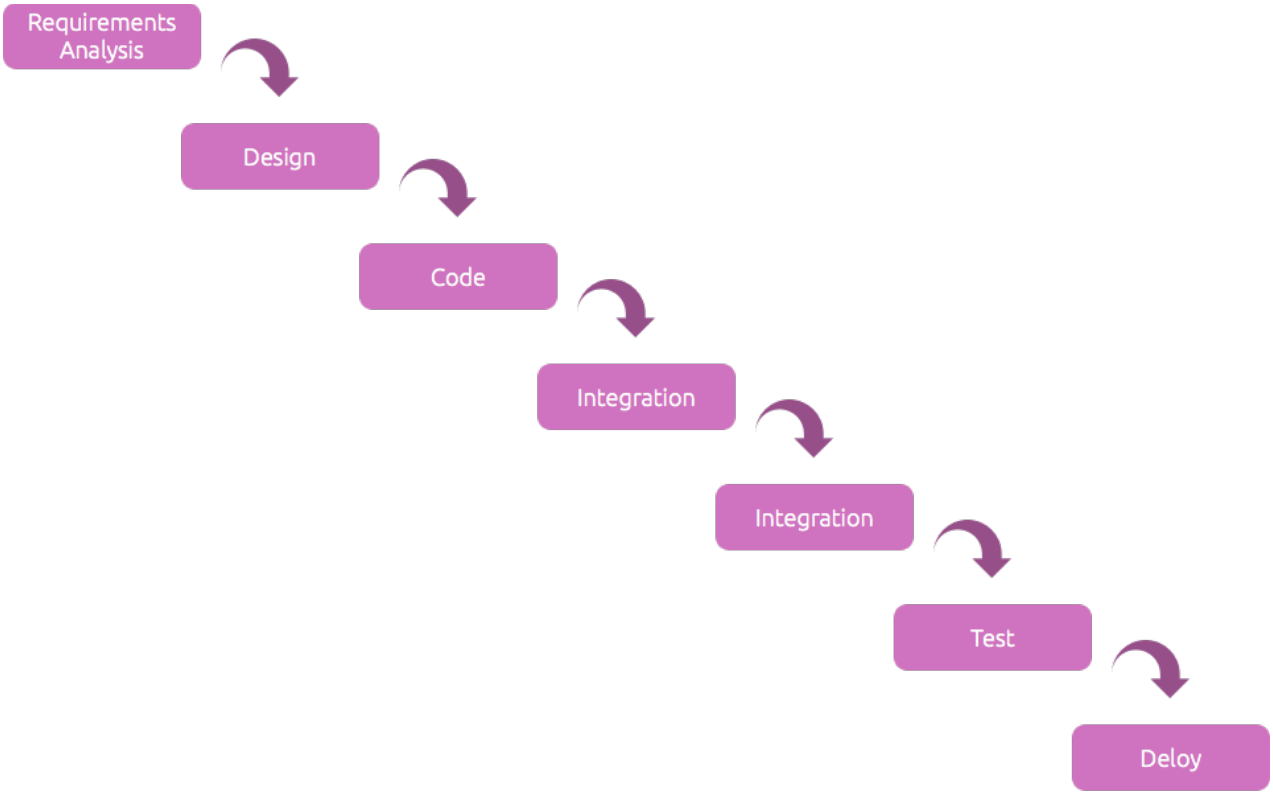


Figure 2.K | Waterfall Methodology

2.7.2. Agile Scrum

Scrum is a framework part of the Agile which is a set of methods based on collaboration between cross-functional teamwork. The Scrum method involves a series of week long “sprints” to developing a product which is improve on each new sprint. After each sprint the team has a meeting to discuss what has been and what can be improve on or added. Since is it a team-based methodology, it is not suitable for this project. It

has the potential to be modified to suit an individual however the time constraint on this project is too short for sprints. Example shown in **[Figure 2.L]**

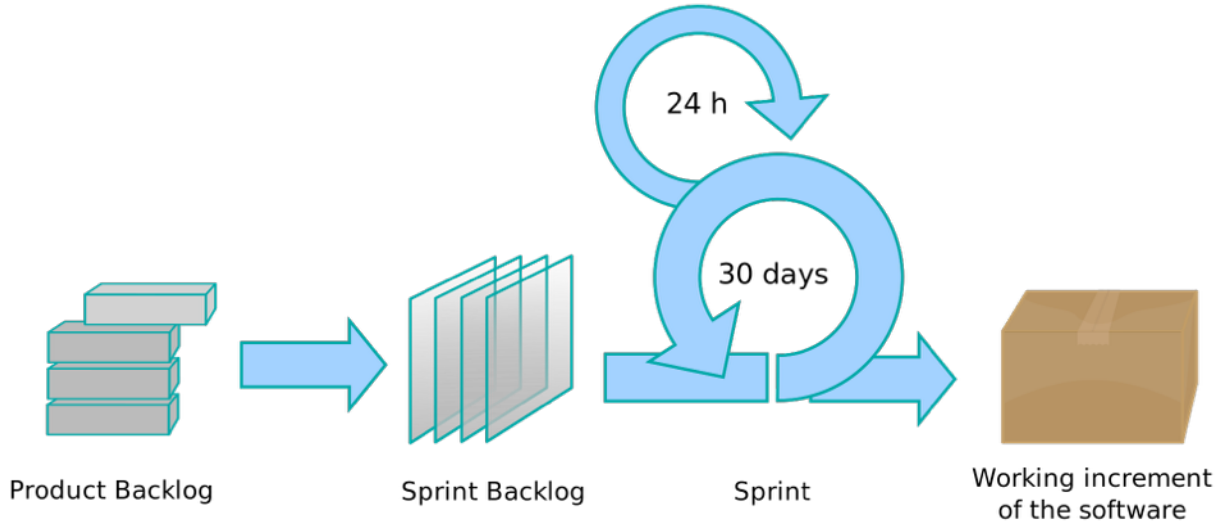


Figure 2.L | Scrum Methodology

2.7.3. Prototyping

Prototyping is a method in which a prototype is developed, tested then refined until there is a desired outcome is reached so a final application can be developed from it. Prototyping is good for project where the requirements are not as defined at the start of a feature needs to set aside time to learn how to build it. This makes it ideal for project where the developer is new to a lot of the features in the application being built. This method does however make the scope of the project hard to control.

The chosen methodology for this project is **Prototyping**. The reasons behind this being that most of the features being developed are new to learn and features near the end of the development are the least important, the core features are required to be developed right at the start of the project. Example shown in **[Figure 2.M]**

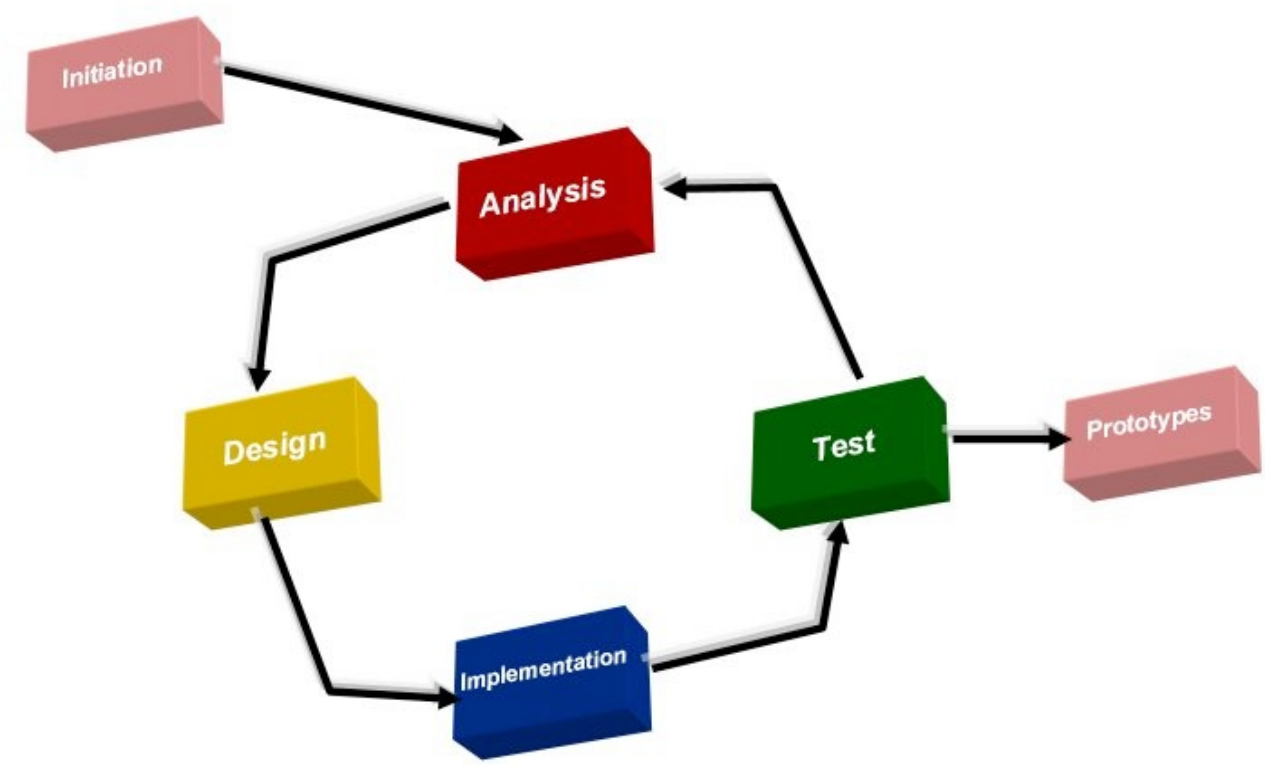


Figure 2.M | Prototype Methodology

3. DESIGN

3.1. UX DESIGN EVOLUTION

During the initial design stages of the project, several wireframes were made for the application layout. However from what follows, many of these layouts were changed in the final designs due to the creation of graphics such as the user's avatar.

The project's avatar, logo, 'HexaBunnies'(part for the game's story) and city style background were created using Affinity Designer **(Serif, n.d.)**, a vector and raster design tool similar to Adobe Illustrator. It was chosen for the project as the user interface made it less complicated to use compare to Adobe Illustrator, there was also the opportunity to incorporate rasters if the vector tools did not suffice.

3.1.1. Branding

The name of the application was decided upon after several concept sketches **{Appendix 3.A]** were made of avatar and game's villain designs. Inspiration was drawn from kids TV shows like Powerpuff girls, Steven Universe, SpongeBob Squarepants and Girl meets World. The wording and style for these TV shows were looked into to discover what made them appealing to the target audience and then used to help form the concepts of sketches of the application logo design. It was from the combination of kids TV shows and HTML

that the name of the application became, ‘CodeGirl vs the Hexabunnies’. One interesting part of the this name is the ‘Hexabunnies’, this also became the inspiration for the game’s villains and was derived from the use of hexadecimal with CSS to display colour.

During the sketches, some colour experiments were made to help towards the final colour scheme. When the the project’s requirements were outlined, it was found that the target audience had no overall preference to colour scheme, so the decision was made to experiment with colours that are usually associated with the target audience, girls aged 8-11. This being bright colours with either pinks or purples. Something that was also taken into consideration during these experiments was to tried pull in images or shapes associated with HTML or any other programming language, the reason for this being that the application must inform the user as to what the application is about through the branding.

As see below **[Figure 3.A]**, the final logo design displays the use of the angled bracket found in HTML elements, this along with the font “Cornerstone”. Cornerstone is a free font made by Zac Freeland **(Zac Freeland, n.d.)**, this font was decided upon experimenting with various other fonts with a similar ‘blocky’ quality. From the previous colour experiments, purple was chosen as the main colour, this was put together with some dark shades and an analogous pink colour, which was slightly altered from it’s original shade to tone down the contrast.

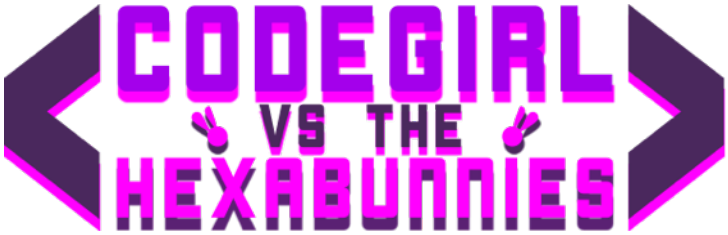


Figure 3.A | Final logo Design

3.1.2. Avatar

The creation of the avatar and the several components that accompany it are essential for attracting the target audience, see first concept **[Figure 3.B]**. The avatar was designed to have a young feminine appearance, as well as having a range of the skin colours. A range of concepts were created for the avatar's style, hair and clothing articles. As the single designer of the project, the styling was limited, to the designer's own personal style. However this appeared to match the project's branding and appealed to the users based on their interest with cartoons.

Figure 3.B | First Avatar Concept

Figure 3.C | Outfit concepts

As a key feature of the avatar, several outfit concepts were made **[Figure 3.C]**, the focus of these outfit designs was to give the user a range of choices to match their own personal preferences and allow the user to project their personality on to the avatar. These were further refined along with the avatar pose for the site. See **[Appendix 3.B]** for further refined outfit designs. Some examples shown in **[Figure 3.D]**

During this refinement stage, hairstyle design were created as well, **[Figure 3.E]**. The designer tried to incorporate a number of styles for the users' natural hair styles. See **[Appendix 3.C]** for more hair styles.

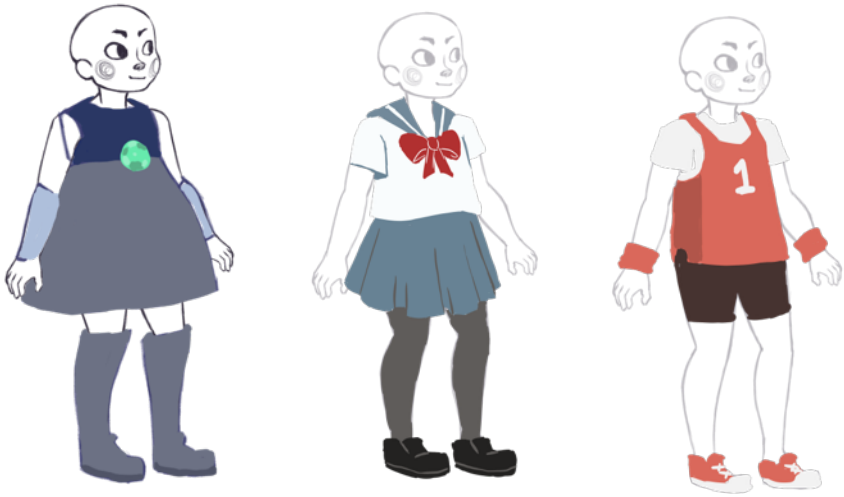


Figure 3.D | Refined Outfit concepts

Figure 3.E | Hair Concepts

The final stage of the avatar's creation was to vectorise the avatar components using Affinity Designer. See **[Figure 3.D]** for example of the finished avatar. The components that were originally sketched up were placed inside as a guidelines. The application pen tool was used to manually trace the shape of the components, these were adjusted using the Node tool until the desired look was created.These were exported as SVG files for use in the project. SVG can be manipulated on a website using CSS and a number of third party APIs such as SnapSVG **(Dmitry Baranovskiy, n.d.)**. During the implementation stage the manipulation of SVG will be attempted to create a way for the user to pick their own hair colour and skin colour, other SVG files will be made for colour options as a backup.

Figure 3.D | Examples of Finished Avatars

3.1.3. HexaBunnies and the Background

The creation of the the HexaBunnies follow the design path as the avatar, with the exception that the design will remain static throughout the application. See **[Appendix 3.D]** for the hexabunnies concepts. The background design was sketch out as a concept, see **[Appendix 3.E]**, however after experimenting to create cityscape in Affinity designer the best result where create using several free city vector available at Vecteezy **(Vecteezy, n.d.)** combining them to create the background. As the colour scheme was created the background colours changes from grey shade to green and orange gradients. See finished HexaBunnies in **[Figure 3.E],** and see finished background in **[Figure 3.F]**.

Figure 3.E | Finished HexaBunnies

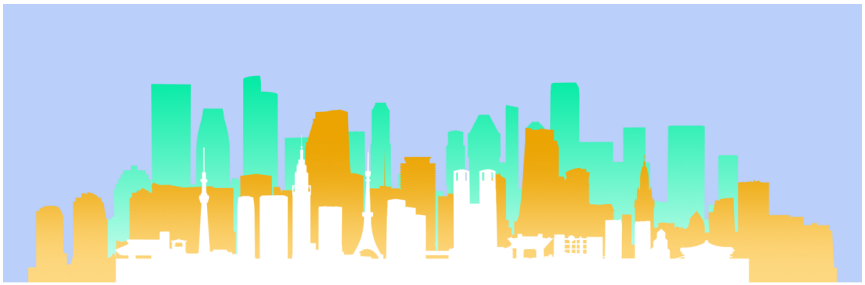


Figure 3.F | Finished Background

3.1.4. User Journey

Before the application's pages could be finalised, how the user would view the application, needed to be considered. A journey map was created to show the pathways the user will follow through the application. See **[Figure 3.G]**.

Beginning at the Home page, if the user visited the application for the first time, they would attempt to signup. Once they have given their details to the signup, they would be allowed access to the site and receive an activation email. Signup would redirect them to the avatar creation page so they can start the game with their personalised avatar before the first level. However if the user has an account with the site already upon logging in they will be redirected to the account page where they can choose from a number of paths such as; level selection, avatar customisation, leaderboard or they can log out.

The Newly signed up user, after customising their avatar, will be directed to the account page and given the same options to choose. If they choose to go to the level selection they will see several levels are available, clicking any of these will begin the game and the level's introduction will appear. The user can choose, after the introduction, to play the level or leave and be brought back to the level selection page. After completing a level

the user will be brought back to the level selection where they can continue to a new level, or select from the menu to be brought to anywhere else on the application.

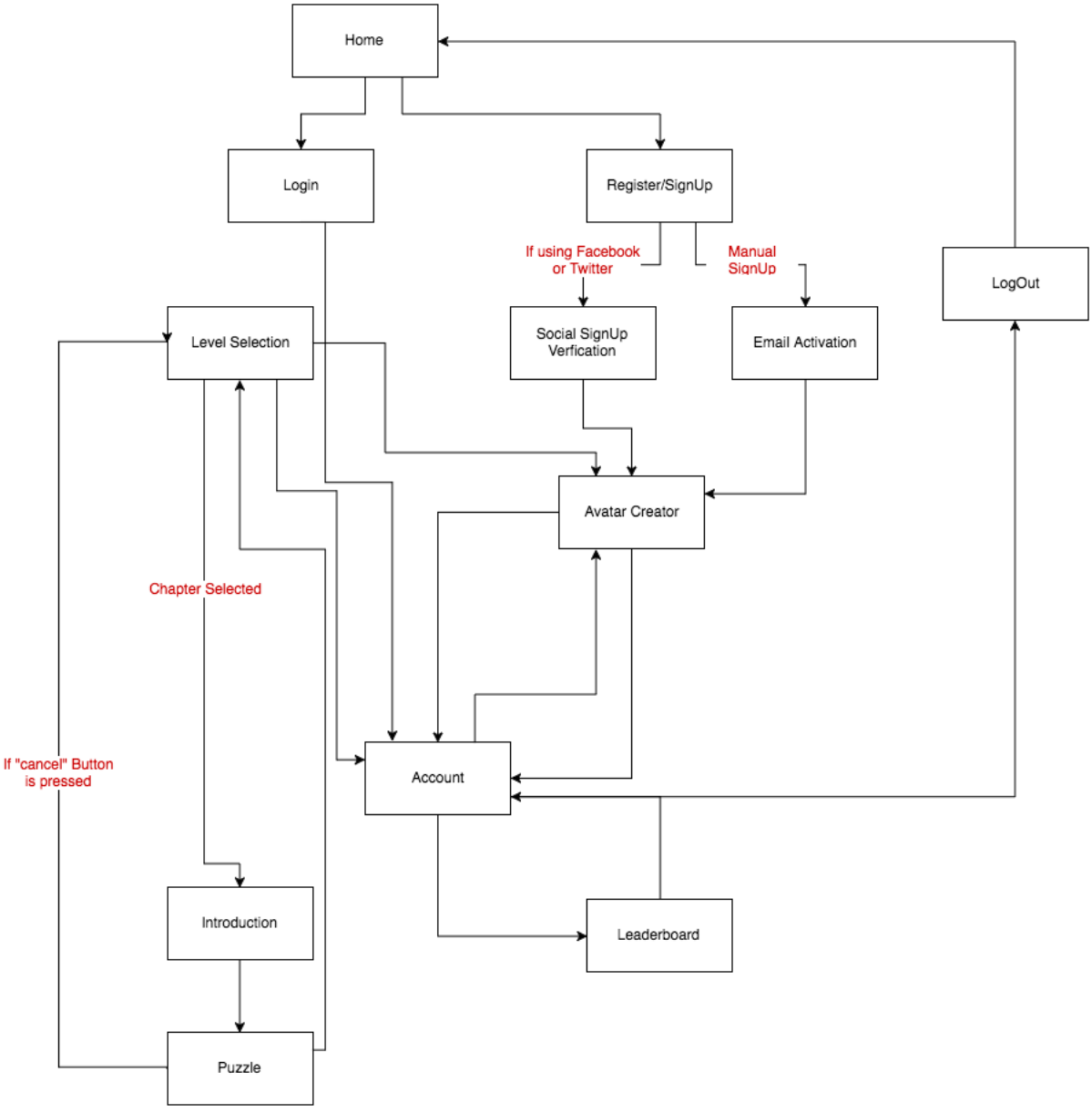


Figure 3.G | User Journey

3.1.5. Style Guide

The application's finalised page layout was designed with Sketch app which was used for the high fidelity wireframes previously. The Sketch app has a number of third party plugins, one of the which being Craft **(INVISION, n.d.)**, Craft helps manage styling items and can even create a style guide from the current design being developed. This was used to create a style guide for the application creating a colour scheme, fonts uses and test styles. See **[Figure 3.H]** for style guide.

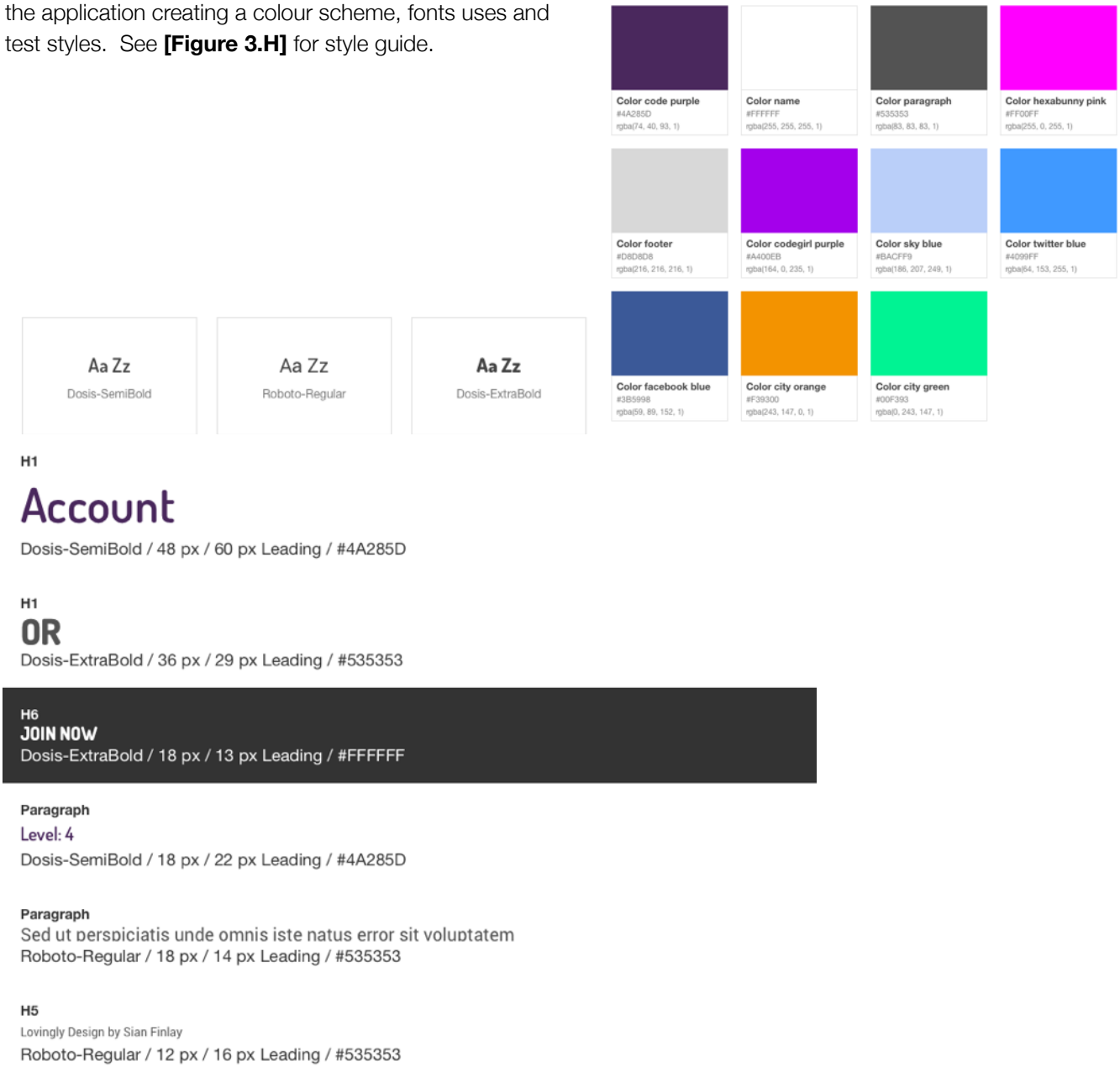


Figure 3.H | Style Guide

3.1.6. Finalised Design

As shown below in **[Figure 3.I]**, all the elements that had been worked on previously were pulled together to create very bright and minimalist design. It was decided in the requirements that the application must work on tablet devices, as this was the most common device used by the target audience. Most of the site can be

Figure 3.I | Finalised Design

viewed without the need to scroll and elements such as buttons were enlarged for touch screens.

3.1.7. Responsive Design

As a key part of the application's design, the layout was reconsidered for smaller screens such as tablet and mobile devices. The key difference between these versions of the application are the removal of access to the game on screens where the level would be unusable. The user's will be informed to return to the account page using a linked button. See **[Figure 3.J]** for responsive design.

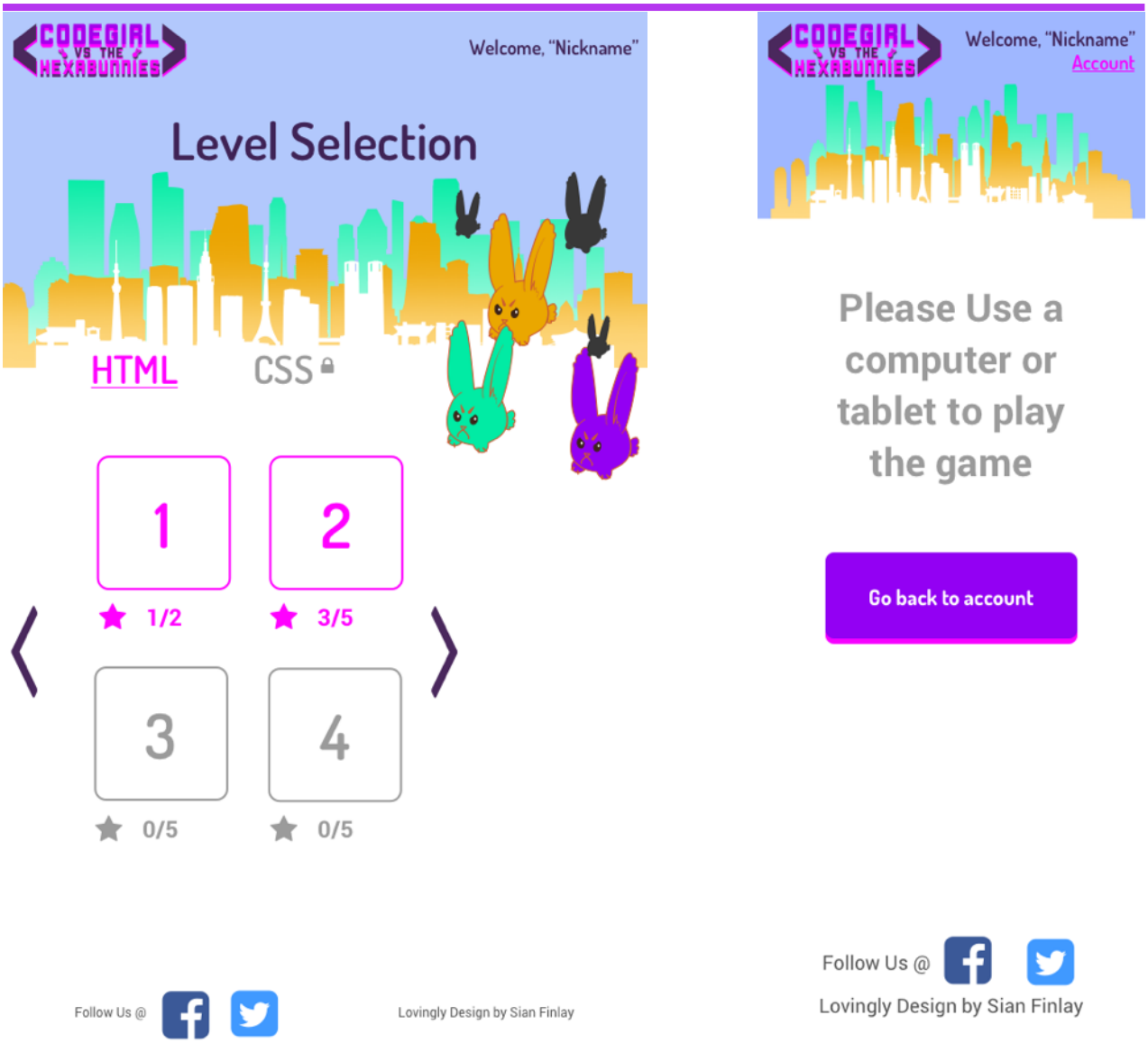


Figure 3.J | Responsive Design

Level Design

Following the work done on the prototype, several level designs using the drag and drop method were made. Each level design consisted of several draggable objects which have matching outlines that work as an area that any draggable object can be placed into. There are also background elements which help the user find the correct areas for the draggable objects to be placed. Some examples of this being a block of text with outlined area wrapped around it as an indicator that the draggable object should be a paragraph element. See **[Figure 3.K].**

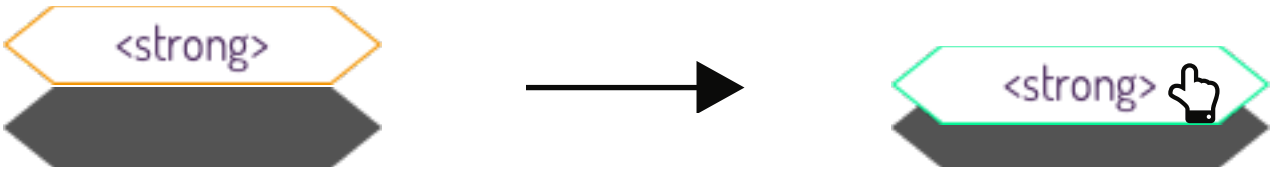


Figure 3.K | Draggable Objects

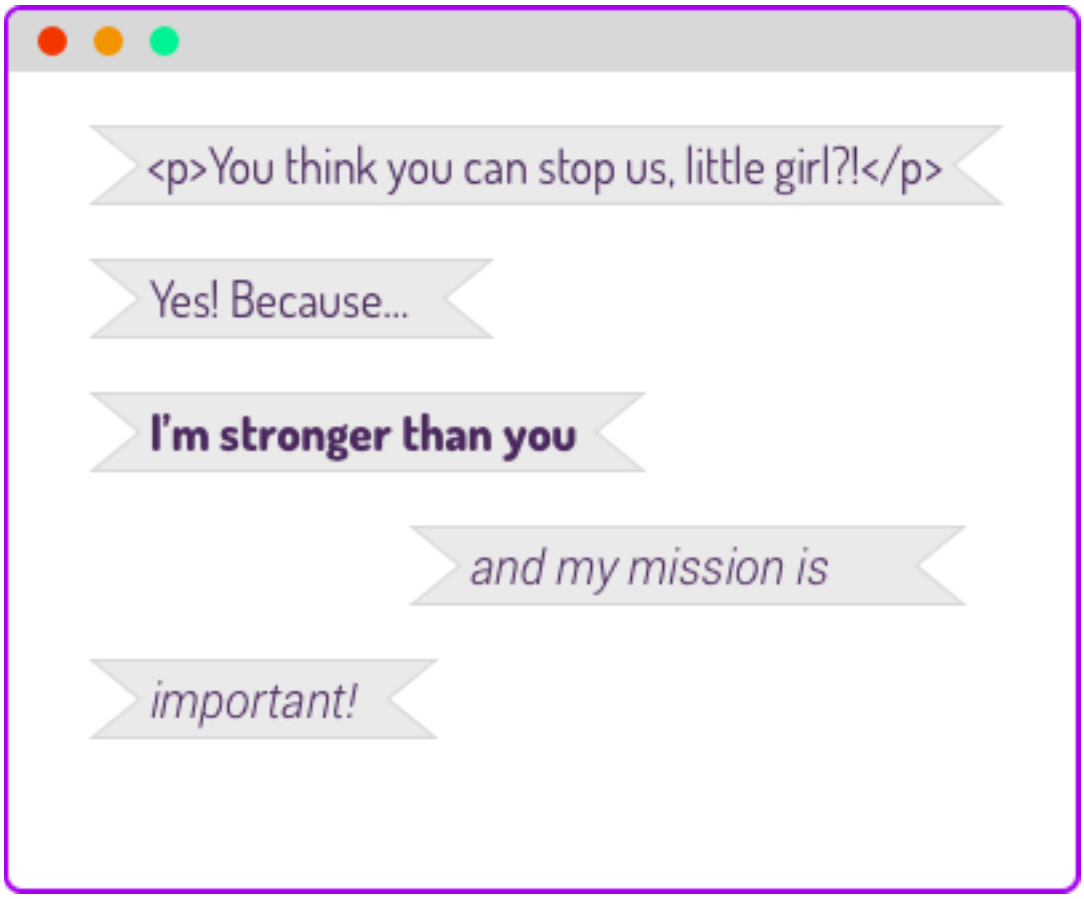


Figure 3.L | Level Design

Each level's background elements are wrapped in a ribbon-like border, these borders match the ends of the draggable objects and outlines, meaning they can be used to make more levels easily in Sketch app. See **[Figure 3.L]**.

3.2. SYSTEM DESIGN

In the initial System design section the project's client and server side technologies were discussed using a client-server model. After the feasibility prototype was created, the client and server Javascript platform Meteor was added to the list of technologies to be used in the project. Meteor uses the MV* web architecture pattern, this mean that it can become any pattern that contains a Model and a View. For this project, meteor was modified to use an MVC pattern with some controllers such as page routing and the use of meteor's own style of Javascript and jQuery event handling. To control the application's page routes a Meteor package called Iron Router will be used, this package will control which templates are used as a part of the application's main layout.

3.2.1. Model-View-Controller Pattern

MVC is the web architecture pattern used to separate information in three parts; the Model, View, Controller. In the Model we find the MongoDB database and database queries. The View is what is seen by the user including HTML CSS and templating. The controller is what the user can interact with such as buttons. Programming languages associated with this are jQuery and canvas. The three parts will interact with each other such as the view displaying updates from the model and the controller send request to the model to update the database… See **[Figure 3.M]**

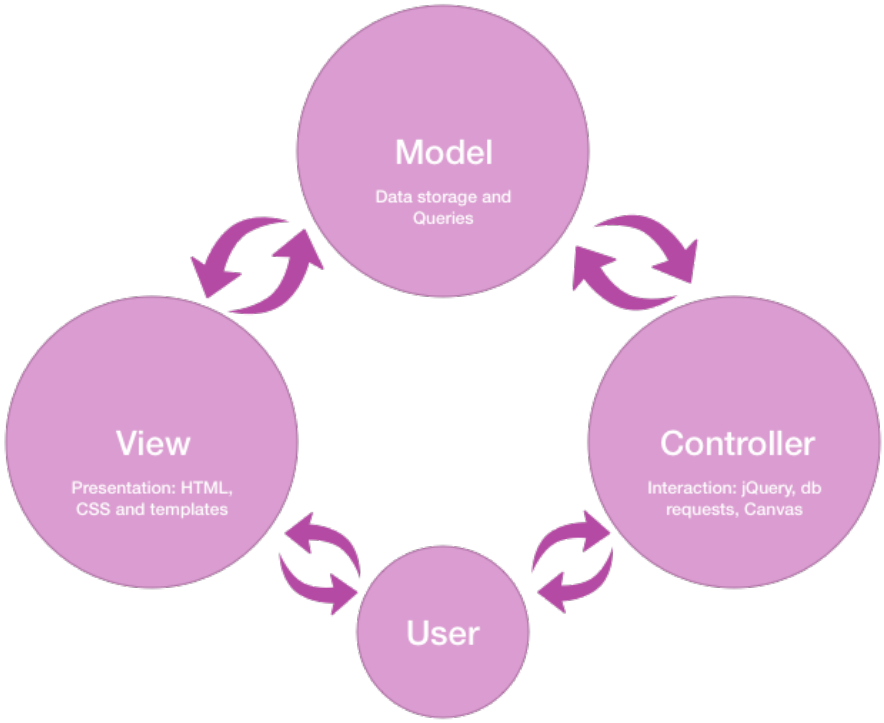


Figure 3.M | MVC Pattern

3.2.2. Database Design

The design for these databases will vary slightly from regular ER diagrams because the Meteor platform uses MongoDB which is a non-SQL database. MongoDb uses JSON to store data. In MongoDB, tables are called collections, rows are called documents which contain field-and-value pairs. Using a JSON structure means MongoDB documents can hold more data by embedding one document inside another. Documents also do not have set fields, so one document can contain a field another does not. However MongoDB collections can not use the JOIN operation that is frequently use with SQL databases to perform queries across multiple

tables, there are workarounds to this in Meteor by writing code that will find collections containing a similar specific field… See **[Figure 3.N]**

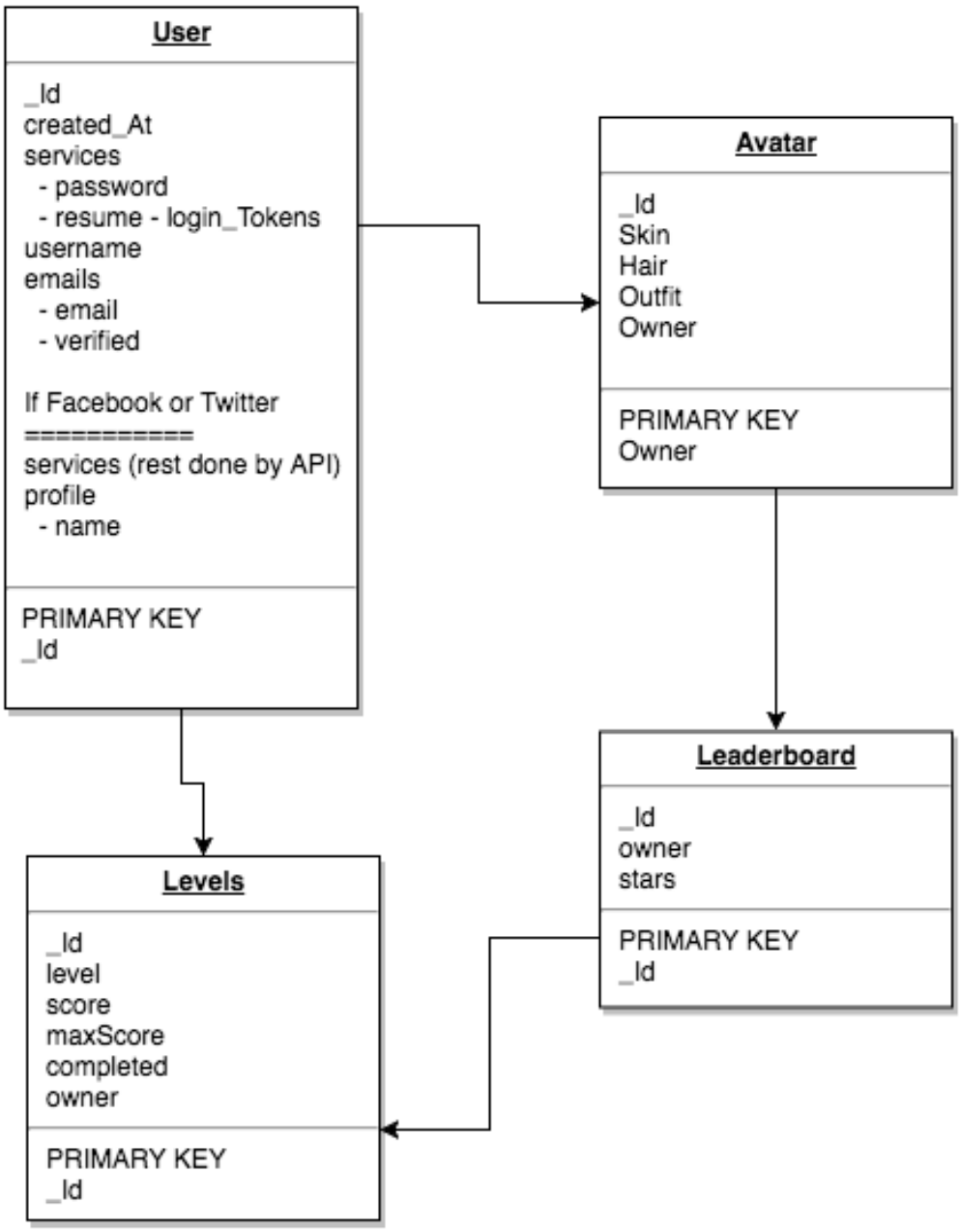


Figure 3.N | Database diagram

In order to create a fake JOIN between the collections, the ‘owner’ field is given to the Avatar, levels and leaderboard upon created, these fields hold the User collection’s ‘_Id’ field link all collections by that specific field.

4. IMPLEMENTATION

4.1. TECHNOLOGY SELECTION

There have been a few change to the technologies since the prototype, better solutions have been found to help develop the application. Others have been added to the lists after better research was done on the technology.

4.1.1. Client-side languages

The client side of the application will use several languages, the structure and styling of the pages will be created using HTML and SCSS. SCSS is a different version of SASS which shares more similarities with CSS, it is much easier to use for developers who are use to the structure of CSS. HTML will be used along with Meteor’s built-in templating system which uses a modified version of Handlebars inside the HTML files. Event handling with be done by jQuery and Meteor’s template event handling. The drag and drop level uses canvas along with a framework, that will be discussed below. Lastly SVG will be used for most of the images, including the avatar’s customisable components. During development, experimentation will be done to see if the SVG file use to display the hair and skin components can be altered and stored in the Avatar collection for greater customisation options.

4.1.2. Client-side Frameworks and Toolsets

KonvaJS **(lavrton, n.d.)**

KonvaJS is a new canvas framework inspired by KineticJS, which is a framework that is no longer maintained. KonvaJS has built in support for mobile devices, such as drag events and both mouse and touch events are handled by the framework. KonvaJS will be replacing EaselJS. The reason for this being one the framework’s demo called “animals on the beach”. This demo has some of the functionality that the project’s drag and drop levels needs, it has a scoring system and images can be loaded and organised using arrays. There is several modifications that will be needed to be made to this demo in order to meet the needs of the project such as fixing the drop and snap areas so that they will take any draggable object, right or wrong answers, and creating a solutions to send the score to the levels collections and be added together with other level’s score for to compare score with other users.

Bourbon **(thoughtbot, n.d.)**

Bourbon is a SASS toolset, it uses SASS mixing and extends abilities to create function that can be used by the developer such as it shade() and tint() functions which are similar to SASS built in lighten() and darken(), but

shade() uses black tones and tint() uses white tones. Bourbon also comes with a handy rem() function is converts any pixel value into rem, instead of the developer having to manually convert pixel number.

Materialize **(Materialize, n.d.)**

Materialize is a CSS and SASS framework, it will be used to handle the application's basic styling such as; grid structure, form styles, button effects. Materialize is also bundled with google's material icons which will be used for usability support for areas such as form inputs and menus. The main reason Materialize was chosen was for its modal function, especially for level introductions where they can be called at the beginning of the level to explain the task to the user, The modal will also be used in the level completion screen, signup and login areas.

Swiper **(iDangero, n.d.)**

Swiper is a 'mobile touch slider' plugin, as the description says it supports touch controls. Swiper has several options available to use such as breakpoints to help make the slider more adaptive to smaller screens. Due to Meteor's templates each slider can be given different options to suit their content.

4.1.3. Server-side language and framework

Thanks to the success of the prototype, the project is able to use NodeJS along with the Meteor Platform on the Heroku servers. The use of the NodeJS, has been on the rise since its creation. Many popular sites have now transferred from their original PHP set up to NodeJS. Using NodeJS and Meteor will future proof the applications, as more advancements in NodeJS emerge, the application can be built upon with this new technology.

There are several benefits to using Meteor such as its built in error log for both server and client side and Meteor's command line which can be used to build the project in the local server and temporary database which can be easily reset with the command 'meteor reset',

4.1.4. Meteor Packages

Meteor, like NodeJS NPM package manager, it has it own package manager called Atmosphere. Meteor uses packages for everything, and expects its developers to do the same, it makes it easy for developers to add new packages to Atmosphere. By adding these packages, developers make these packages available to other developers so instead the same packages being made multiple times. Another benefit of using packages is it makes the application less cluttered, hiding files not created by the developer. All of the above client-side frameworks and tools were added as packages to the application. Listed below are other package use in the applications, including some built-in ones, that come with every new Meteor project. Packages are found in the packages file under the .meteor folder.

Blaze **(Meteor Blaze, n.d.)**

Blaze is Meteor's own template library, similar to "Angular, Backbone, Ember, React, Polymer, or Knockout". Meteor states that it was built because they felt that other templates made user interfaces, "unnecessarily difficult and confusing". Also since it is Meteor's own template, it works well with Meteor's database requests. Blaze uses a modified version of Handlebar called Spacebars, which are embedded into the html to called templates and loop through collection documents.

Tracker **(Meteor Tracker, n.d.)**

Tracker is "Meteor's client-side reactive programming library", it works with Blaze and handles events and other Javascript within the templates. Template.name.events() is where events are held such as click events, Template.name.helper() are normally use called collections and attached them to a Spacebars name, and Template.name.onRendered() works like jQuery's $(document).ready() calling Javascript function when the template has been loaded.

Iron Router **(iron, n.d.)**

Iron Router is a popular routing package for meteor, it works with Meteor templates treating them like routes/ pages in a website. Any template can be used as a route using Iron Router's route function. This package can also declare a template as the main layout for the application and change the title for each page visited.

Accounts **(Accounts-password, n.d.) (Accounts-ui, n.d.) (Accounts-facebook, n.d.) (Accounts-twitter, n.d.)**

There are several packages under the accounts name, these are Meteor's own packages. These packages help build the signup and login. They create the user collection and add many of the necessary fields to the user collection. The account packages used in this project are accounts-password, accounts-ui, accounts-facebook and accounts-twitter. The Password package will encrypt the user password and works with the ui package to deal with the input fields for the email, username… The Facebook and twitter packages add social login to the application, these are called using their associated functions called by the login button. For the social login to work the application must have the right api and secret keys for the application url address.

Service-Configuration **(Service Configuration, n.d.)**

The Service-configuration package works with the social login packages. This package is used to store and deal with the Facebook and twitter keys.

Force-ssl **(Force-ssl, n.d.)**

One of the problems with using the social login packages is they can only work on a secure server, so if a user uses http instead of https they will not be able to login. For applications with signup and login functionality, it is best practice to use a secure server to protect to user's data. Using the Force-ssl package stops the user from using the insecure server by redirecting them to secure server. This package requires no additional coding, once added to a project it will automatically change user accessing http to https.

Minifier **(Standard-minifiers, n.d.)**

Both the js and css version of this package is built into any new project create with Meteor. The benefit of this package is it will only minify files once the project has been deployed for production. This benefits the developer as errors can be dealt with on the local server, using developer tools.

4.1.5. Tools

During this project several tool were used to aid the creation of the application. These tools have various uses throughout the development of the application ranging from error logging and debugging, hosting, coding, testing.

Google Developer tools, Git version control and terminal were used together to find errors on the application and then for debugging. The Developer tools logged errors on the client-side whilst the terminal logged server-side errors, meteor built-in error logging also helped add more meaning to these errors. Using Git as a version control, any error found could be backtracked to a working copy of the site and find what was introduced that caused the error to occur in the first place.

All the application hosting was done through Heroku. Heroku uses a Meteor build-pack to implement Meteor reading the .meteor folder in application to get the packages listed. There is also a plugin called 'mlab' where all the MongoDB collection from the application goes.

The last set of tools is devices used during testing for the application's responsiveness and functionality on the these different devices. The devices used in this project are; Macbook Pro, Nvidia Shield K1, iPad mini, Amazon Fire HD.

4.2. TECHNOLOGY USE

4.2.1. Signup and Login

On the Signup and Login templates there are a set of input fields which take the user's information, as well as the option to either login using Facebook or Twitter.

For a normal Signup using the form fields, the user's information is sent to the 'signup_login.js' file, on the client-side development folder, which use Tracker's form event handling on the Signup template to place the user information into a set of variables… See **[Figure 4.A]**

```
// create user
Template.Signup.events({
    'submit .signup': function (event) {
        event.preventDefault();

        var username = event.target.username.value;
        var email = event.target.email.value;
        var password = event.target.password.value;
```

Figure 4.A | Signup variables

These variables are used to create an array which will be sent to the 'user' collection using the accounts-password and accounts-ui packages. Once this data has been received by the collection the application will begin the process of login the user in and redirecting them to the customisation page to create their avatar. The redirect uses iron-router's go function… See **[Figure 4.B]**

```
        var user = {
            email:email,
            password:password,
            username:username,
            createdAt: new Date()
        };
        Accounts.createUser(user, function(err){
            if(!err) {
                Router.go('/customisation');
            }
        });
```

Figure 4.C | User collection request

Similarity if the user was to login using their email and password. Another Tracker form event takes the user information this time calling the account packages to use the 'loginwithPassword' function… See **[Figure 4.D]**

```
Meteor.loginWithPassword(email, password, function(err){
    if(!err) {
        Router.go('/account');
    }
});
```

Figure 4.D | LoginWithPassword

Both the Facebook and Twitter logins in the signup use the same sets of functions. The differences between those in the signup and the login template is the page which they are redirected… See **[Figure 4.E}** and **[Figure 4.F]**

```
'click .face_btn':function(event){
    event.preventDefault();
    Meteor.loginWithFacebook(function(err){
        if(!err) {
            Router.go('/account');
        }
    });
},
```

Figure 4.E | Route to account

```
'click .twitter_btn':function(event){
    event.preventDefault();

    Meteor.loginWithTwitter(function(err){
        if(!err) {
            Router.go('/customisation');
        }
    });
}
```

Figure 4.F | Route to customisation

A problem arises when there are several ways a user can signup to the application. The accounts packages however has a built-in detector for new user called ‘accounts.onCreateUser()’. With this documents in other collections such as the ‘avatar’ can be initiated for future use. This is function is called on the server-side as it deals with database inputs… See **[Figure 4.G]**

```
Accounts.onCreateUser(function(options, user) {
    Avatar.insert({
        skin: 'skin_1',
        hair: 'hair_1_brown',
        outfit: 'dress_1',
        owner: user._id
    });
```

Figure 4.G | onCreateUser

4.2.2. Customisation

Once the user has signed up they will begin to create their avatar. Customisation items are held within tabs create using the materialize framework, inside these tabs holds a Swiper slider with the items.

In order to collect the correct SVG file for the user avatar, the items in the slider are each linked to an invisible radio button. This is done using jQuery with Tracker to target the template link to each slider.

Since the radio buttons are invisible to the user, when the item is click on it gains the ‘active’ class making it stand out against those where the ‘active’ class has been removed… See **[Figure 4.H]**

```
// change hair being display on avatar, this is not saved yet
$("img.ava_hair").click(function() {
    $('img').removeClass('img_active');
    $(this).addClass('img_active');
    $(this).next().prop("checked", "checked");
    var checked_rad = $('input[name=hair]:checked').val();
    console.log(checked_rad +"radio is checked");

    var hair_src = "/images/dressup/hair/"+ checked_rad +".svg";
    $('img.hair').attr("src", hair_src);
});
```

Figure 4.H | change shown components on Avatar

When the user is satisfied with their choice they will click the 'save' button. Which, like the signup and login, get the values of the selected items and saves them to the user's linked avatar document in the 'avatar' collection. However if the user is only changing one item or presses the save button without any items, their previously saved items will be used and/or only the one item will be changed. In order to do this, hidden inputs were linked to the original selection using Spacebars… See **[Figure 4.I]**

```
"click #saveAvatar": function (event) {
  event.preventDefault();

  var skin = $('input[name=skin]:checked').val();
  var hair = $('input[name=hair]:checked').val();
  var outfit = $('input[name=outfit]:checked').val();

  if(skin == null){
    skin = $('input[name=altskin]').val();
  }
  if(hair == null){
    hair = $('input[name=althair]').val();
  }
  if(outfit == null){
    outfit = $('input[name=altoutfit]').val();
  }
  // Insert avatar changes
  Meteor.call("updateAvatar", skin, hair, outfit);
  Router.go('/account');
}
```

Figure 4.I | Request avatar update

The 'Meteor.call()' is the request to the the server-side which is link to the Meteor method which handles the update query to the collection. The 'owner' field in the 'avatar' collection is what links the avatar to the user, this field contains the user's unique '_Id'… See **[Figure 4.J]**

```
updateAvatar: function (newSkin, newHair, newOutfit) {
    var avatar = Avatar.findOne({
        owner: this.userId
    });

    Avatar.update({
        owner: this.userId
    }, {
        $set: {
            skin: newSkin,
            hair: newHair,
            outfit: newOutfit
        }
    });
}
```

Figure 4.J | Update avatar collection

4.2.3. Levels

The levels were created using Canvas and KonvaJS. Using the KonvaJS 'animals of the beach' demo as a base. The function, called 'snapTo', to allow all the draggable objects to snap to the outlines. The function was created within the loop which rendered the draggable objects to the canvas whilst creating the on drag events. The reason the function was created inside the loop was so every draggable object could be called in the function using the variable which declares the the draggable object is an image. This also took the name of the outline declared in the 'outlines' array. This was then called inside 'dragend' event for each outline in the level… See **[Figure 4.K]**

```
function snapTo(code, outlineSnap) {
    var outline = outlines[outlineSnap + '_black'];
    if(isNearOutline(code, outline)) {
        code.position({
            x : outline.x,
            y : outline.y
        });

        console.log("privKey:"+ privKey);
        console.log("Key:"+ key);
        console.log(code.id());
        codeLayer.draw();
        var testKey = privKey+ "_black";
```

Figure 4.K | snapTo Function

If the Draggable object matches the outline, a point is added to the score which is sent to the 'submit' modal so that the score will be there no matter when the user decides to finish. There is also a win and lose message that will appear depending if the user gets the full score or not.

Similar to the customisation, the levels have a hidden input which is given the value of the user's end score, this has the id of 'newScore'. The document associated with the level in the 'levels' collection has its' 'score' field updated as well as the 'stars' field in the leaderboard collection… See **[Figure 4.L]**

```
// if in right place give 1 point
if(isNearOutline(code, outlines[privKey + '_black'])) {
    if(privKey == code.id()){
        score += 1;
        //update score in sumbit modal
        $('.score').html(score);
        $('#newScore').val(score);
        //change to match score
        if(score >= 2) {
            // text kept for debugging purposes
            var text = '';
            // put max score in modal
            $('.scoreMessage').html("You've learnt you first Element! Keep this up and
                you'll beat those HexaBunnies back to their planet!");
            $('.score').html(score);
            $('#newScore').val(score);
            drawBackground(background, images.background, text);
        }
        // keep track of score, for debugging
        console.log(score);
    }
}
```

Figure 4.L | Score

In order to update the 'leaderboard', we use a Meteor and MongoDB function called 'map' which loops through all of the 'score' linked to the user as the 'owner'. Once all the scores are added together this is used to update stars in the 'leaderboard' collection… See **[Figure 4.M]**

```
Template.levelSelect.helpers({
    levels: function () {
        var stars = 0;
        Levels.find({owner: Meteor.userId()}).map(function(doc) {
          stars += doc.score;
        });
        Meteor.call("updateStars", stars);
        return Levels.find({owner: Meteor.userId()});
    }
  });
```

Figure 4.M | Update stars

4.2.4. Leaderboard

Since MongoDB cannot use joins, an extra function must be called when the 'leaderboard' collection is called to the client-side file. It uses Meteor's transform function to find the 'avatar' document which has an 'owner' field that match the 'leaderboard' 'owner' field. It then returns this as part of the client-side copy of the 'leaderboard' collection… See **[Figure 4.N]**

This new leaderboard collection is called in the leaderboard template using Meteor's helper function. Inside, the leaderboard is called with 'sort' and '-1' this the MongoDB way of sorting document's like SQL queries can do. In this case the leaderboard is being sort from highest to least number of the stars… See **[Figure 4.O]**

```
Leaderboard = new Mongo.Collection("leaderboard",  {
    //create join to avatar collection
    transform: function(doc) {
      doc.avatarObj = Avatar.find({
        owner: doc.owner
      });
      return doc;
    }
  });
```

Figure 4.N | Transform with collection call

```
Template.leaderboard.helpers({
    leaderboard: function () {
        // get leaderboard, and sort from ascending stars amount
        return Leaderboard.find({}, {
          sort: { stars: -1 }
        });
    }
  });
```

Figure 4.O | Sort and display leaderboard

4.3. NOTABLE CHALLENGES

During the development stage of the project some features had to be cut due to time constraints and errors occurring that affected other more important features.

During the development of the avatar customisation, attempts were made to give the user more customisation for the hair and skin colour. One SVG plugins tried was SnapSVG **(Dmitry Baranovskiy, n.d.)**, the plugin allowed SVG files to be loaded in the HTML without copying the whole SVG code into the page, this could then be manipulated with the plugin's JavaScript. The plugin successfully changed the colour of the avatar's skin. However two problems emerged. The first problem was the plugin was having problems loading the entire SVG file; the avatar's shoes would be missing and the highlights in the hair would also be missing. The error would occur frequently at random when the page was reloaded. The second problems was storing the plugin data in the 'avatar' collection. This function was removed due to these problems as the time needed to create a solution would mean losing some the game's functionality yet to be implemented.

At the beginning of level testing, it was discovered that if the user took the draggable object out of the outline the added score would remain. This was fixed by adding a conditional statement to remove the score when the object was not in the outline. However due this the user would also lose score if they continuously dropped the object in the wrong place eventually letting the score drop into negative numbers. In attempts to fix this, another conditions was added so that if the score equals 0 it would not drop anymore, allowing the user to recover their score and still receive the max score. After these fixed this left us with two bugs, one which the users could accidentally receive a lower score than they actually got because they drop the same object twice, the other bug being that if the user drags the object then drops it on outline again they can gain extra points. These bugs could not be fixed due to time constraints… See **[Figure 4.P]**

```
// not in right place take away score
if(!isNearOutline(code, outlines[privKey + '_black'])) {
    codeLayer.draw();
    if(score == 0){
        console.log("no score")
    }
    else {

        score -= 1;
        $('#newScore').val(score);
    }
    console.log(score);
}
```

Figure 4.P | Score Error

4.4. NOTABLE ACHIEVEMENTS

There have been several learning achievements throughout the development of this project such as use of NodeJS and Meteor, creation of game levels, leaderboard and social logins.

Beginning with NodeJS and Meteor, thanks to the use of Heroku as host for the application, the developer was able to gain experiences using the two together and gain the benefits of Meteor's packages and its ability to reactively add changes to the page without any extra development. As one of the newest and most popular server-side languages NodeJS will future proof the developer's skills.

During the levels' creation, a problem emerging early on to add the function that would allow the objects to snap to any outline, the fix for this was discussed early during the technology use section. How this was fixed was through the combination of google's developer tools and asking for advice on StackOverflow **(Stack Overflow, 20/04/2016).**

Another notable achievement was the creation of the social login using the Meteor Facebook and Twitter packages. Having this functionality in the application meant that users could join the site much easier. It was felt that younger users such as the target audience would have more access to these services and appreciate their functionality.

5. TESTING

5.1. APPROACH

Following the Usability planning guide on usability.gov (Usability.gov, n.d.) a user test was created and conducted on 4 members of the University of Ulster LINK gaming society. Due to the age of the target audience these participants were deemed to be the best alternative due their interest in gaming.

As well as the usability test, browser and device testing was done by the developer. Browsers tested on were Chrome, Safari and Firefox. Devices tested on: Macbook Pro, PC, Nvidia Shield K1 (Android), iPad Mini and Amazon Fire HD(Modified Android).

5.2. TEST PROCESS

Following the guide on the usability.gov, the user test was planned out with the following guides:

5.2.1. The Scope

The test will be conducted on the 26-27th April and will be covering the use of the whole application.

5.2.2. Purpose

1. Test the signup and login form and/or social logins.
2. Whether the application appeal to the female candidates.
3. Can the user successfully customise their avatar.
4. Can they navigate through the site without any problems
5. Can they interact with the game's levels on any device
6. Does the user avatar appear on the leaderboard
7. Can they see their stars count on the account page

5.2.3. Schedule and Location

Testing to adapt to the participants availability and to be conducted remotely with Facebook messenger app as the form of communication. The reason the test is to be done remotely, is so that the developer cannot not influence the participants actions or cause distress to them.

5.2.4. Equipment

Personal computers, laptops and tablets devices; all of which are provided by the participants.

5.2.5. Participants

The Number of participants will be 4. They are students aged between 18-24 who are all part of the University of Ulster LINK gaming society.

5.2.6. Scenario

- Create an account with the application
- You will then customise your avatar
- Complete at least one level
- Go to account and find your stars
- Go to the leaderboard and find your avatar
- Then log out
- Complete the following Questionnaire: http://goo.gl/forms/FALWfRMbrE

5.2.7. Developer Browser and Devices Test Results

During the developer tests, a bug was found on the customisation page on the Safari browser and iPad mini. Using Safari's developer tools one of line of code in the customisation JavaScript file was found to be a missing bracket in the Javascript where one of the variables used to collect the input value was missing a ']', this was ignored by other browsers and the two android devices during their tests.

The developer test were also conducted to test how the applications looked on smaller devices, if any components were harder to read or overlapped by others. The tests showed that there was no errors in the look of the applications these smaller devices… See **[Figure 5.A]**

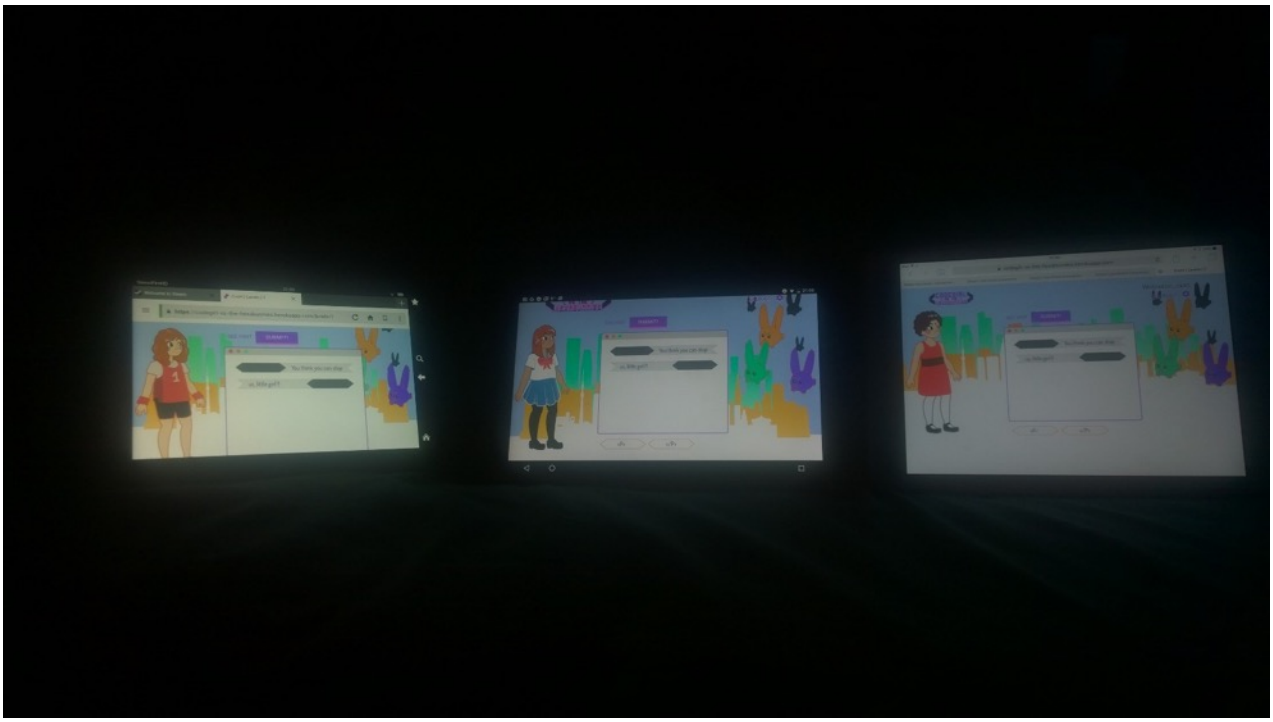


Figure 5.A | Tablet Device testing

5.3. USER SURVEY RESULTS

Due to a time constraint with one of the participants, some of the questions could not be answer by the individual… See Table on **[Figure 5.B]**

Participants > ————— Questions	**1**	**2**	**3**	**4**
Are you male, female or neither?	Male	Female	Male	Female

Participants > ————— Questions	1	2	3	4
What device are you using with the site?		Tablet	Laptop	Laptop
Were you able to signup/login, without any problems?	Yes	Yes	Yes	Yes
Were you able to customise your avatar, without any problems? (did you character changes save etc)		Yes	Yes	Yes
Were the level instructions easy to follow?	Yes	Yes	Yes	Yes
Were you able to navigate the site, without any problems?		Yes	Yes	Yes
Did you encounter any problems on the levels?		I encountered a problem on level two where the / strong tag wasn't sticking to the shareholder however after a refresh it worked.	No comment	Nope
Can you see your star count on the account page?		Yes	Yes	Yes
After completing a level, did you see you score count below the level change on the level selection page?		Yes	Yes	Yes
In the leaderboard was it easy to pick out your avatar against other users' avatar?		Yes	Yes	Yes

Participants > ————— Questions	1	2	3	4
Can you find the log out button?		Yes	Yes	Yes
What do you think of the site?	Solidly built, with clear display for the most part	I thought it was a very interesting way to get younger people especially females to be interested in development as this is still a sector in need of skilled developers for future projects. I thought the design was very inviting and would not put off other people who are not the target audience.	I thought it worked fine, sometimes elements took a while to find.	Definitely hits its target demographic and also good use of humour for audience engagement.
Did you find anything difficult to use when using the site?	Was initially confused about how to move between menus, but figured it out.	It was quite clear, I did try logging back in however I had to do it a second time to allow me to progress.	Nope	Nope
Is there anything you think the site can improve on?	Explore button could be clearer. it doesn't really stand out against the background. Also was able to place multiple answers into 1 slot, so that could be problematic (in heading lesson).	An improvement would be maybe in a different update a pose for the character or to add a password change option in account.	Reconsidering the position/colouring of some of the interactive elements maybe?	-1 for having puppies and not kittens...seriously though it looks good some of the arrows/text can be too close to each other/images.

Figure 5.B | User Test Results

After one the Participants' pointed out that the 'explore' menu was bit small to notice, an adjustment to the size of the text was made from 18px to 24px.

6. EVALUATION

6.1. USER TEST EVALUATION

As shown by the answers given by the participants of the User test, the application is easy to use on both laptop and tablet devices. However the test also showed that participants came across known bugs on the applications such as getting a decreased score and that the lack of validation on the login did not warn the user of a bad email or password used. Also another bug that was not well known appeared, was the fact that the participants could place more than one draggable object at the same time into one outline. Ways this error could be fixed includes increasing the spacing between outline on the level where the bug occurs or developing a function that would remove the outline from play until the current object is removed. On the question about asking the participants to suggest any improvements on the site, most of the participants commented on improving the look of the application such as improved colour scheme and more poses for the avatar.

6.2. PROJECT OUTCOMES

The overall outcome of the project was a fully functional application which fulfilled the many of the requirements and made the application appealing to its target audiences. Some of the main requirements fulfilled were; a drag and drop game, avatar customisation and a story attached to the user's avatar as a magical girl fighting alien bunnies. Some minor requirements had to be left out due to time restrictions and the developers lack of skill. These include making the application playable without graphics and the creation of an admin ability to create more levels. Some requirements which would require testing on the target audience, remained questionable as to whether they have been fulfilled such as keeping the vocabulary of the level introductions at a level that 8-11 year olds could understand.

6.3. METHODOLOGY

The methodology used for this project was Prototyping. This methodology proved to be the best solution as the developer required time to learn the languages and frameworks needed for this project. The use of plugins such as SnapSVG were removed from the application after the customisation prototyping stage showed that the plugin would take up too much time need for other requirements. Another example of where the methodology became useful was during the feasibility prototype which experimentation took place to discover; whether NodeJS could be used in the project, the best framework for the project and testing whether the main game mechanic of drag drop would be produced.

6.4. PLAN

Other than fulfilling the project aim, objective and requirements, this project lacked planning. The project did however follow the hard and soft deadlines given by the project's module. If more planning was put into place, there could have been more time given to refine the application and fulfil more requirements.

7. CONCLUSION

7.1. REPORT SUMMARY

The project set out to build a learning tool for girls aged 8-11, using story and drag and drop games to teach and inform the user about both HTML and other web development languages. An aim and several objectives were defined and the project research was conducted on competitors to find the best solution to the problem. A set of requirements were made after a user survey was conducted including parents and relatives of the target audience.The prototyping methodology was compared to others and finally chosen for the project. After which project risks were defined and a feasibility prototype was developed in order to look at some of the risks. The application's user interface design and system design were worked out before moving on to developing the full application. Once a working application was made, user and developer tests were conducted and evaluated.

7.2. PROJECT REFLECTION

The project proved challenging to work on however it added greatly to the skills of the developer. Due to the poor time management for the project, it caused stress to the developer and several requirements suffered because of this. The project was also interesting to work on, looking at many different programming languages and gave the developer some freedom in the design of the user interface, such as adding their own illustrations as avatar customisation. If there was more time at the end of the project, some of the bugs talked about during the project's notable challenges could have been fixed and validation/feedback could have been added to the signup and login forms for help the user.

7.3. LEARNING OUTCOMES

The developer was able to gain experience in many new languages such as NodeJS and the framework Meteor along with the packages as apart of it. Languages already known by the developer were also improved on such canvas, javascript and jQuery. They also gain knowledge on how to better structure and use web architecture pattern such as MVC.

7.4. FUTURE WORK

There are a number of the future possibilities for this application. One of the most basic ones being the creation of more levels for the user to experience and learn from. There is also the possibility of developing language tracks for different web development languages. The application could also be developed to give the user a reward after each level other than the current score system. Printable achievement could be made, or shareable badges that could only be obtained when the user complete a language track. More avatar outfits could be created that become unlockable after certain number of levels are completed. The colour customisation of the hair and shin could be revised until a working feature is produced. Lastly exploration of the more of Meteor's package could be done to see if anything there could be added.

REFERENCES

1. Sailor Moon , n.d., Sailor Moon - Wikipedia, the free encyclopedia [Sailor Moon], [online]. Available: https://en.wikipedia.org/wiki/Sailor_Moon [08/01/2016].
2. Winx Club, n.d., Winx Club - Wikipedia, the free encyclopedia [Winx Club], [online]. Available: https://en.wikipedia.org/wiki/Winx_Club [08/01/2016].
3. Gamification, n.d., Gamification - Wikipedia, the free encyclopedia [Wikipedia], [online]. Available: https://en.wikipedia.org/wiki/Gamification?oldformat=true [08/01/2016].
4. Jane McGonigal, Feb 2010, Jane McGonigal: Gaming can make a better world | TED Talk | TED.com [TedTalks], [online]. Available: https://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world?nolanguage=en#t-129509 [08/01/2016].
5. Neopets, n.d., Welcome to Neopets! [Welcome to Neopets!], [online]. Available: http://www.neopets.com/ [08/01/2016].
6. Daily Mail Reporter, 6/2/2014, More than half of children use social media by the age of 10: Facebook is most popular site that youngsters join | Daily Mail Online [Home | Daily Mail Online], [online]. Available: http://www.dailymail.co.uk/news/article-2552658/More-half-children-use-social-media-age-10-Facebook-popular-site-youngsters-join.html [08/01/2016].
7. HM Treasury, Department for Education, The Rt Hon Michael Gove MP and The Rt Hon George Osborne MP, 4/2/2014, Year of Code and £500,000 fund to inspire future tech experts launched - News stories - GOV.UK [Gov.uk], [online]. Available: https://www.gov.uk/government/news/year-of-code-and-500000-fund-to-inspire-future-tech-experts-launched [08/01/2016].
8. Ladies Learning Code, n.d., Ladies Learning Code [Home - Ladies Learning Code], [online]. Available: http://ladieslearningcode.com/ [08/01/2016].
9. Drum Roll, n.d., E.A.K. | Erase All Kittens [Erase All Kittens], [online]. Available: https://eraseallkittens.com/ [08/01/2016].
10. Geeky Ventures, 2015, Code Monster from Crunchzilla [Crunchzilla], [online]. Available: http://www.crunchzilla.com/code-monster [08/01/2016].
11. Lifelong Kindergarten Group, n.d., Scratch - Imagine, Program, Share [Scratch], [online]. Available: https://scratch.mit.edu/ [08/01/2016].
12. Hopscotch, n.d., Hopscotch - Make your own game. Learn to code. [Hopscotch], [online]. Available: https://www.gethopscotch.com/ [08/01/2016].
13. Codarica Inc., n.d., CodeQuest - Learn how to Code on a Magical Quest with Games on the App Store [CodeQuest], [online]. Available: https://itunes.apple.com/gb/app/codequest-learn-how-to-code/id919965565?mt=8 [08/01/2016].
14. Unity, n.d., Unity - Game Engine [Unity - Game Engine], [online]. Available: https://unity3d.com/ [08/01/2016].
15. John Naughton, 31/3/2012, Why all our kids should be taught how to code [Fit in IT], [online]. Available: http://fit-in-it.ch.mediapx6.nine.ch/sites/default/files/downloads/Why all our kids should be taught how to code.pdf [08/01/2016].

16. Jane McGonigal, 2011, Reality is Broken [Stanford HCI Group], [online]. Available: http://hci.stanford.edu/courses/cs047n/readings/Reality_is_Broken.pdf [08/01/2016].

17. Corbett, Christianne, 2015, Solving the equation: the variables for women's success in engineering and computing | VITAL Repository 5.4.1 [Home | VITAL Repository 5.4.1], [online]. Available: http://vital.new.voced.edu.au/vital/access/manager/Repository/ngv:68238 [08/01/2016].

18. Sapna Cheryan, Allison Master, and Andrew N. Meltzoff, 11/2/2015, Cultural stereotypes as gatekeepers: increasing girls' interest in computer science and engineering by diversifying stereotypes [National Center for Biotechnology Information], [online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4323745/ [08/01/2016].

19. Renee Weber, – VP Consumer Strategy and Research – The Marketing Store & Terence Burke, VP Qualitative Research – KidSay, 23/1/2015, Digital Tweens: A Peek Into Their World • KidSay.com [KidSay | Full service kids market research & consultancy], [online]. Available: http://kidsay.com/digital-tweens-report/ [08/01/2016].

20. James & Suzanne Robertson, 2016, Requirements Specification Template [Volere Requirements home page], [online]. Available: http://www.volere.co.uk/template.htm [08/01/2016].

21. Sian Finlay, 19/11/2015, survey monkey [survey monkey], [online]. Available: https://www.surveymonkey.com/home/ [08/01/2016].

22. Bohemian Coding, n.d., Sketch - Professional Digital Design for Mac [Sketch], [online]. Available: https://www.sketchapp.com/ [08/01/2016].

23. Grant Skinner, n.d., CreateJS | A suite of JavaScript libraries and tools designed for working with HTML5 [CreateJS | A suite of JavaScript libraries and tools designed for working with HTML5], [online]. Available: http://www.createjs.com/easeljs [08/01/2016].

24. cykod, 2012, Quintus JavaScript HTML5 Game Engine [Quintus JavaScript HTML5 Game Engine], [online]. Available: http://www.html5quintus.com/ [08/01/2016].

25. lavrton, n.d., Konva - JavaScript 2d canvas framework [Konva - JavaScript 2d canvas framework], [online]. Available: http://konvajs.github.io/ [26/04/2016].

26. Joyent, n.d., Node.js [Node.js], [online]. Available: https://nodejs.org/en/ [08/01/2016].

27. Rasmus Lerdorf, 1994, PHP - Wikipedia, the free encyclopedia [Wikipedia, the free encyclopedia], [online]. Available: https://en.wikipedia.org/wiki/PHP [08/01/2016].

28. Mike McNeil, 2012, Sails.js | Realtime MVC Framework for Node.js [Sails.js | Realtime MVC Framework for Node.js], [online]. Available: http://sailsjs.org/ [08/01/2016].

29. Meteor, n.d., Meteor [Meteor], [online]. Available: https://www.meteor.com/ [08/01/2016].

30. Salesforce, n.d., Cloud Application Platform | Heroku [Cloud Application Platform | Heroku], [online]. Available: https://www.heroku.com/ [26/04/2016].

31. Serif, n.d., Affinity Designer - Professional graphic design software for Mac [AFFINITY DESIGNER], [online]. Available: https://affinity.serif.com/en-gb/designer/ [27/04/2016].

32. Zac Freeland, n.d., Behance [Cornerstone - Free Font], [online]. Available: https://www.behance.net/gallery/29835665/Cornerstone-Free-Font [27/04/2016].

33. Dmitry Baranovskiy, n.d., Snap.svg - Home [Snap.svg - Home], [online]. Available: http://snapsvg.io/ [27/04/2016].

34. Vecteezy, n.d., Cityscape Free Vector Art - (2444 Free Downloads) [Download Free Vector Art, Stock Graphics & Images], [online]. Available: http://www.vecteezy.com/free-vector/cityscape [27/04/2016].

35. INVISION, n.d., Craft by InVision LABS [Craft by InVision LABS], [online]. Available: https://www.invisionapp.com/craft [27/04/2016].

36. thoughtbot, n.d., Bourbon - A Lightweight Sass Tool Set [Bourbon - A Lightweight Sass Tool Set], [online]. Available: http://bourbon.io/ [28/04/2016].

37. Materialize, n.d., Documentation - Materialize [Documentation - Materialize], [online]. Available: http://materializecss.com/ [28/04/2016].

38. iDangero, n.d., Swiper - Most Modern Mobile Touch Slider [iDangero.us], [online]. Available: http://idangero.us/swiper/ [28/04/2016].

39. Meteor Blaze, n.d., Meteor [Meteor], [online]. Available: https://www.meteor.com/blaze [28/04/2016].

40. iron, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [iron:router], [online]. Available: https://atmospherejs.com/iron/router [28/04/2016].

41. Meteor Tracker, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/tracker [28/04/2016].

42. Accounts-password, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/accounts-password [28/04/2016].

43. Accounts-ui, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/accounts-ui [28/04/2016].

44. Accounts-facebook, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/accounts-facebook [28/04/2016].

45. Accounts-twitter, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/accounts-twitter [28/04/2016].

46. Service Configuration, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/service-configuration [28/04/2016].

47. Force-ssl, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/force-ssl [28/04/2016].

48. Standard-minifiers, n.d., The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere [Atmosphere], [online]. Available: https://atmospherejs.com/meteor/standard-minifiers [28/04/2016].

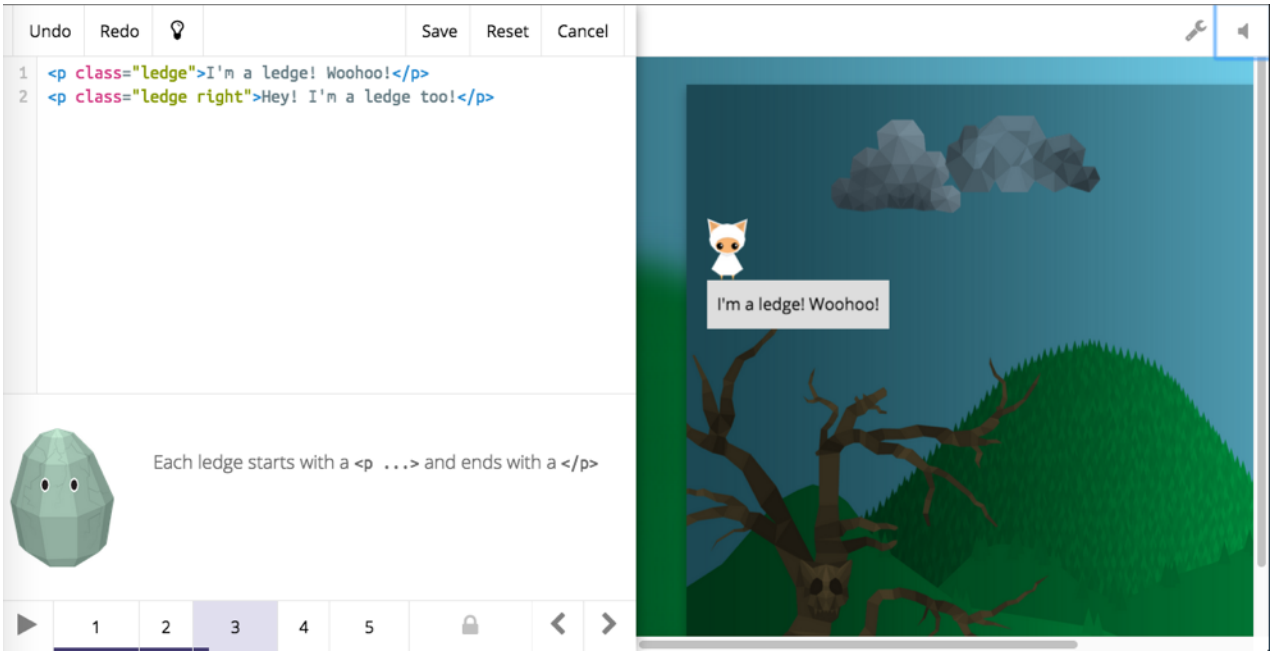
49. Stack Overflow, 20/04/2016, canvas - How to make images snap to any "outline" image (using "animals on the beach" demo) - Stack Overflow [Stack Overflow], [online]. Available: http://stackoverflow.com/questions/36745500/how-to-make-images-snap-to-any-outline-image-using-animals-on-the-beach-dem?

noredirect=1#comment61074609_36745500 [28/04/2016].

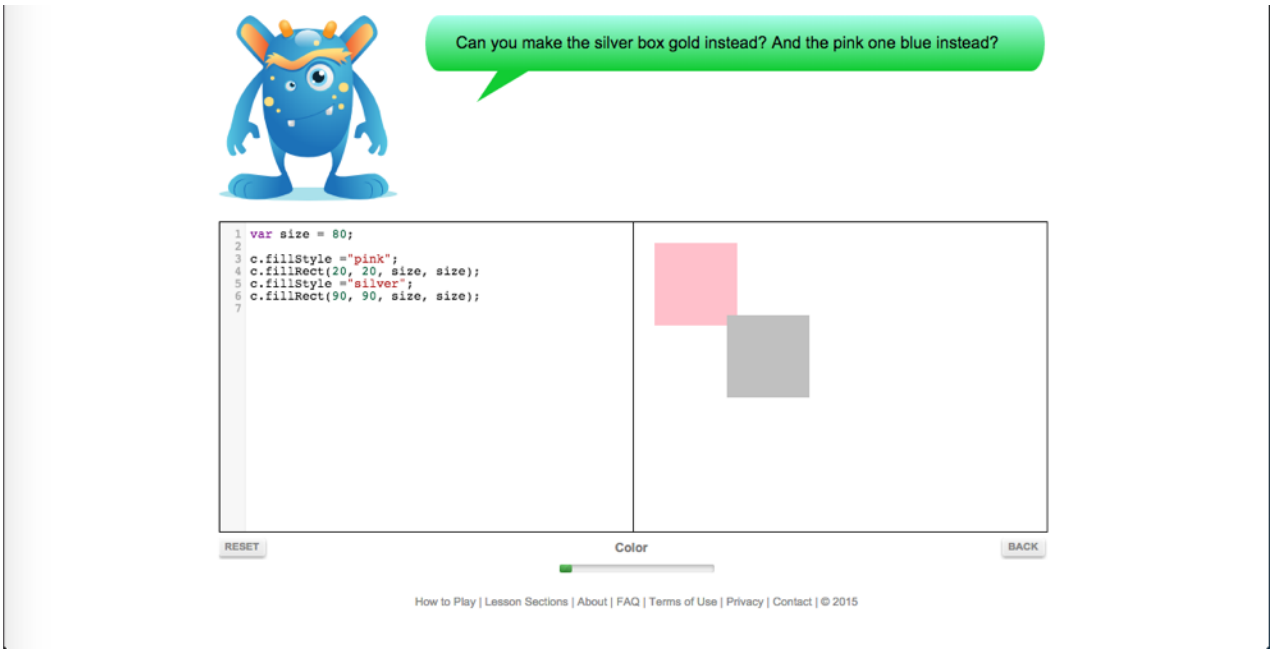
50. Usability.gov, n.d., Planning a Usability Test | Usability.gov [Home | Usability.gov], [online]. Available: http://www.usability.gov/how-to-and-tools/methods/planning-usability-testing.html [28/04/2016].

APPENDICES

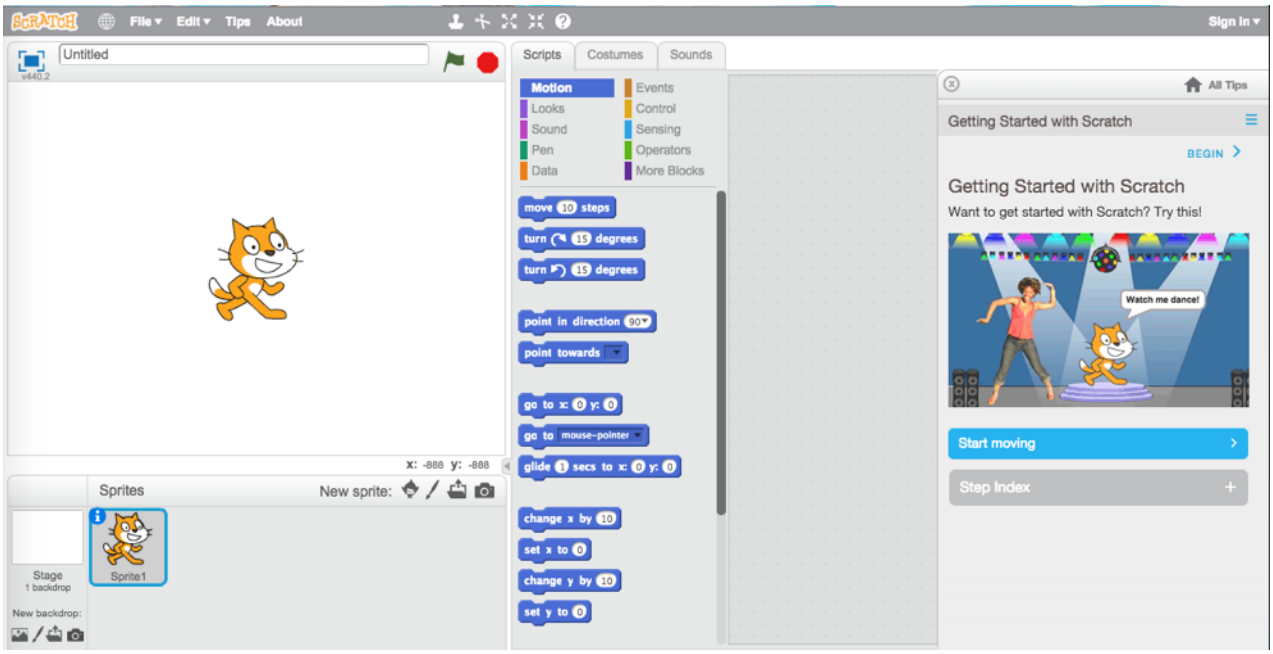
APPENDIX 1.A



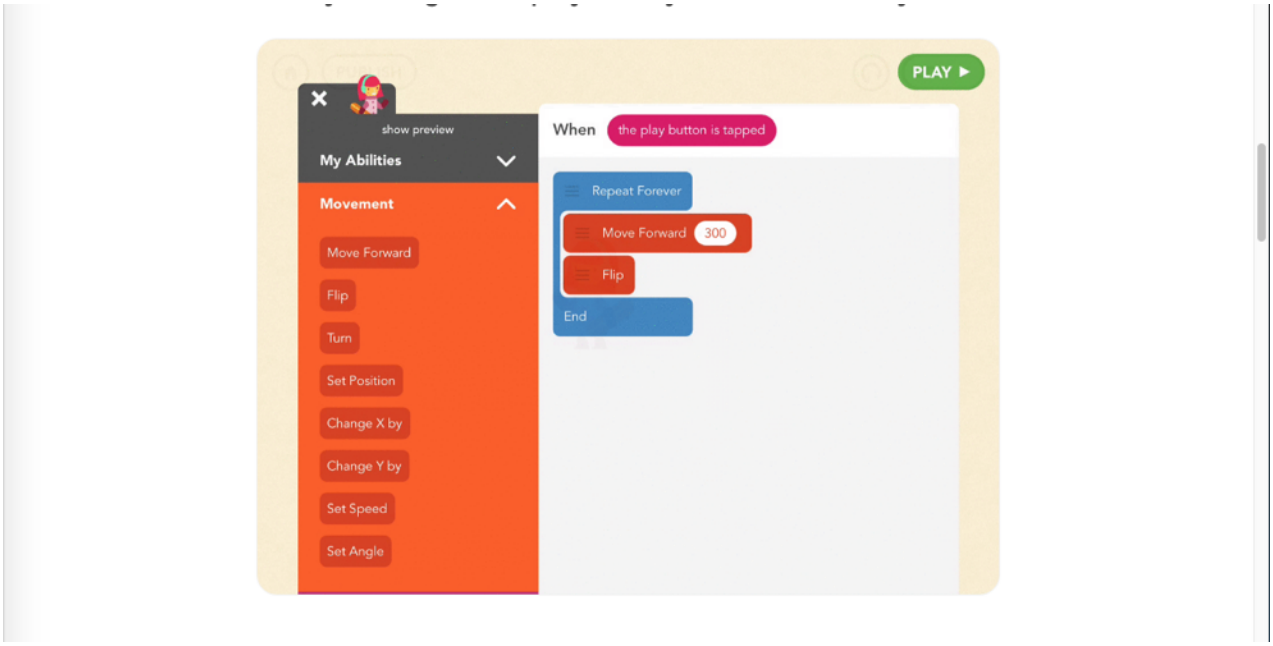
APPENDIX 1.B



APPENDIX 1.C



APPENDIX 1.D



APPENDIX 1.E

APPENDIX 2.A - REQUIREMENTS TABLES

Functional requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
1	The game shall contain puzzles and questions.	To teach young children how to code HTML and CSS	If the user can solve the puzzles and questions to ultimately learn how to write their own web pages.	5	5	1
2	Story attached to the game.	Increase user engagement because they want to continue the story	The user discusses the story's plot after finishing.	5	4	1
3	User can continue their journey.	User is prompted to create an account so their journey can be saved on a database.	The user logins from any device and is brought back to the last puzzle or question they were on.	4	5	1
4	Puzzles are completed by drag and dropping code.	The puzzles help to familiarise the user with chunks of code.	If the user gets the code correct they move on to the next level or are prompted to try again.	5	5	1
5	Multiple Choice Questions are answered using buttons.	User is unable to spell an answer wrong, which is important as the user is young.	If the user presses the correct button they move on to the next level or are prompted to try again.	5	5	1

Non-functional Requirements

Look and Feel

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
6	Product shall be attractive to girls aged 8-11.	To encourage more girls to start learning to code using the product.	A sampling of representative young girls shall, without prompting or enticement, start using the product within four minutes of their first encounter with it.	4	2	1
7	The story shall follow a magical girl, but with reference to girl's journey of growing up.	Popular genre with young girls it also promotes a sense of determination in the audience.	Users will be reminded of some of their favourite media which will entice them to the product.	4	2	2
8	User interface shall use greys and dark colours with few bright tones as directional tools.	Help make the product easy to user friendly. Popular colours with girls in the 'tween' years.	Users are able to use the product without needing to be directed.	4	4	2
9	Full screen backgrounds during levels.	Further engage the user into the story.	User will be emerged in the game.	4	2	2

Usability and Humanity Requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
10	Text shall be kept short.	Young children become distracted easily, so are less likely to read long paragraphs of text.	User continues to play with the product and has better memory of what is happening in the story.	4	2	2
11	Story vocabulary can be understood by younger children.	User's are not struggling to understand the words so the story can continue.	User continues to play with the product and has better memory of what is happening in the story.	4	2	2
12	User will have a customisable magical girl avatar.	User can switch their avatar's components to suit their desire.	User give good feedback on their enjoyment and 'love' of their avatar.	5	2	2
13	User's avatar components will be obtain during puzzles and questions.	User will be prompted to finish puzzles and question in order to customise their avatar.	User get excited at unlocking new avatar components.	5	2	3
14	User can choose a name for their avatar.	User will become attached to their avatar.	User is engage even in the account creation.	4	1	3
15	Puzzles and question are easy to solve.	User will be given hints and guidance during the puzzles and questions.	The success rate should be much higher than the failure rate.	4	4	3
16	Product can be use without graphics.	User will be able to follow the story on a screen reader and puzzles and questions shall also be read out.	If graphic were turn off the product should still be usable.	4	4	4

Performance Requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
17	Product should load under 5 seconds on all devices.	Product component shall be compressed and reduce to increase load times.	All devices should load under 5 seconds.	4	4	4
18	Product shall be available to all primary schools.	Product must work on primary school computer systems.	During user testing test the performance and accessibility on their computers.	4	4	4
19	Product shall load on all touch devices, especially tablets	Young Girls use touch devices more than computers/ laptops	During Devices testing the site should have no errors on touch devices	5	5	2

Operational and Environmental Requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
20	Product shall be accessible on all devices.	Product's CSS will use media queries to control the styling on different devices.	Product should work on all device during testing.	4	3	4
21	Product shall be accessible on tablet as a primary point of access	Touch functions must be working.	User should be able to access the site on a tablet and use the touch functions	4	4	2

Maintainability and Support Requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
22	Admin user shall be able to add new puzzles and questions.	Admin can fill out a form to create the challenges and their answers.	Form should create a new puzzle after the last puzzle or add a new story node.	3	2	5
23	Admin user shall be able to add new graphic to characters and avatar components.	Admin should be able to upload images to database.	New images file will be seen in the database and folders.	3	2	5

Security Requirements

#	Description	Rationale	Fit Criterion	Customer Satisfaction	Customer Dissatisfaction	Priority
24	Bad words should be filter out.	Product code will check itself using a list of bad words for filtering.	If a bad word it add to the product it will be automatically blocked.	3	4	4
25	Users will not be able to interact with other users on the site, other than scoreboard.	Users are very young and interacting with other user they don't know can be very dangerous for them	Product has a one to one interception with the user, there is no messaging system only possibly a way to give feedback to the site.	3	5	3

APPENDIX 2.B - RISK ANALYSIS

Risk	Level	Elimination	Backup
Drag and Drop Canvas game	Red	This risk is the first to be tackled, as it is the application could not function without a proper game mechanic. A prototype will be created, to experiment with canvas frameworks such as easel.js and Quintus to create a drag and drop with response to a specific area.	If this risk can't be eliminated, then the alternative would be to change the main game mechanic to a simple fill in the form, using input fields.
Time required to create character model, backgrounds and custom parts.	Amber	A time limit must be set onto this risk as it is difficult to measure how long this process will take.	If this risk can't be eliminated then the number of graphic for the application must be reduce, such as using one background for whole game, removing custom parts also only one character model remains.

Risk	Level	Elimination	Backup
Node.js with Heroku	Amber	The school's server does not support Node.js, therefore an site like Heroku (Cloud Application Platform) which provide Node.js support and is free to use for some of it's functions. The problem with this method is that if the something goes wrong with the server, I am responsible and may risk losing marks. Additional funding may also be required for custom domain name, To test this a template Node.js application will be published on the Heroku server and functions linked to Node.js will be tested	If this risk does not work, then alternatives such as paying for hosting, or using PHP instead of Node.js will be considered.

Risk	Level	Elimination	Backup
Node.js with Sail.js or Meteor	Amber	As stated with the previous risk, the school server does not support Node.js. This risk is directly linked to the outcome of the previous risk. Sail.js is similar to laravel for PHP, which is provide structure to the application and useful modules. However this is first time I will be using the framework. Meteor is another node framework, must less complicated than sail. This is because it automated a lot of it's process for you, which makes things easier but can be difficult to adjust to. Time must be set aside to tryout the frameworks.	If the framework is too complex or the previous risk determines that Node.js can't be used, then plain Node.js maybe used or PHP with laravel use instead.
Secure login	Amber	The user base is young and vulnerable on the internet. The user's parent must have trust in the security of the site before allowing the user to continue on the site. Time must be invested into researching security methods into the login.	A basic login can be used until a more secure one can be develop in the future.
Social Login and sign up	Green	This risk requires the use of social network APIs which will need time set aside to test.	A basic username and password login/sign up can be develop until time become available to implement this feature.

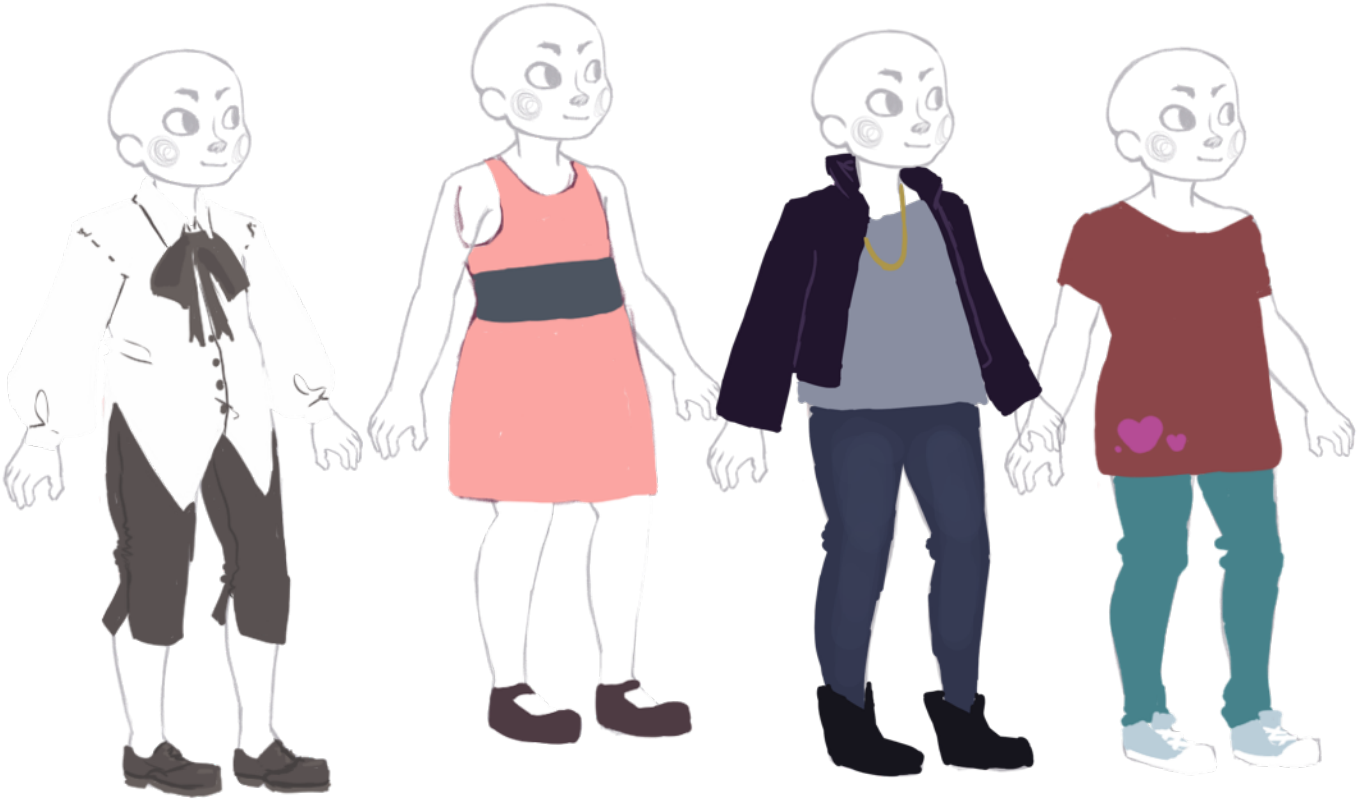
Risk	Level	Elimination	Backup
Character Customisation	Amber	Customisation will personalise the application for the user's experience. This will require further learning into canvas, and a connection to be made to the database to store the customisation options for use in the game.	If I am unable to connect the database to the customisation features, a default character omg will be used for all users.
Character animations	Green	Character preforms animations during the game and when the user beats a level. If the customisation features works a prototype will be made to experiment to see if the animation is best made in canvas or CSS3 animation, or a combination of both. If the customisation feature doesn't work, then the default character animation will be created using sprite sheets with canvas and prototyped.	If none of the character animations work then a static character image shall be used instead.

Risk	Level	Elimination	Backup
Leaderboard	Green	The top 100 user scores in the application will be displayed on a leaderboard. This will require fake user data to be create in order to create a prototype of the connection to the database and to make sure that the score are order by highest to lowest and only 100 user are displayed. Instead of creating 100 faker users, The prototype will be lower to 10 testing the same feature.	If the score are unable to be displayed, the leaderboard will be removed, or an individual user scoreboard will be tested as an alternative.
Scoring system	Amber	Score shall be based on the time it takes the user to complete a puzzle. A timer will be displayed on the interface counting down from 1-5mins (the final time will be determined during testing), the time that is left in the timer will be added to the users score. Prototyping and research will take place in order to find the best solution to get the remaining time added to the score. This score will also have to be saved on to the user's database.	if this risk proves to be too complex then an alternative score system to be test will be counting the mistakes made by the user and subtracting it from a 'perfect' score to equal the final score.

Risk	Level	Elimination	Backup
Game story	Amber	The story is used to increased user interest, create a reason for the user to continue the game. Time will be need to write the story, and research based on the survey done previously for the requirements into the shows the user liked to create a story based on their interests.	If the story take to much time away from the development of the application, a more simple goal based story shall be created, with less content create between puzzles.

APPENDIX 3.A - LOGO CONCEPTS

APPENDIX 3.B - REFINED OUTFITS



APPENDIX 3.C - MORE HAIR CONCEPTS

APPENDIX 3.D - HEXABUNNIES CONCEPTS

APPENDIX 3.E - BACKGROUND CONCEPT