Designing and creating a web application to streamline the character creation and maintenance process of Fantasy Flight Games' Star Wars system

Ryan Day
C1332304

School of Computer Science and Informatics
Cardiff University

Supervisor: Dr Frank Langbein

Acknowledgements

To start this report, I would like to thank my supervisor Dr Frank Langbein for his guidance throughout this project.

I would also like to thank all my family for their support and encouragement that they have given throughout the project. I especially thank my father, Michael Day, who provided a great example while he underwent treatment for and fought off prostate cancer.

Abstract

Throughout this report, I use both the American and English spelling of certain words. Whenever I use American spellings, the word is most likely a term from Fantasy Flight Games which is an America based company.

During the report I refer to players and users. These are generally the same thing but I have tried to use “players” when refer to someone playing in a game and “users” when referring to someone who is using the application.

Table of Contents

Glossary of common terms

Term		Definition
RP	Role-play	When a person assumes the role of a character in a fictional setting
FFG	Fantasy Flight Games [1]	A game company that creates and publishes role-playing games, such as the Star Wars game system used in this project
UI	User Interface	Means by which the user interacts with the application
GM	Gamemaster	Person who acts as organiser for a game and tells a story for the Players to take a role in
Player		Owner of PC
PC	Player Character	Character that is created by a player for a game
NPC	Non-Player Character	Character that is not controlled by a player, normally created by a GM
Exp	Experience Points	Spent by the player to improve their character
EotE	Edge of the Empire [10]	A variation of the FFG Star Wars system focusing on characters similar to Han Solo and Boba Fett
AoR	Age of Rebellion [11]	A variation of the FFG Star Wars system focusing on characters from the Rebellion
FaD	Force and Destiny [12]	A variation of the FFG Star Wars system focusing on characters who can use the Force
XAMPP	XAMPP Apache MariaDB PHP Perl	A webserver package, see section 2.7 on page 12 for more

1 Introduction

In this section I will be giving a little context to the project and why I chose this task. I will also give the scope and direction I took towards the end goal.

1.1 What is RP and Tabletop gaming?

RP, or role-play, is when a person changes their behaviour and actions in order to assume the role of a character in a fictional setting. This is commonly linked to role-playing games where a person creates a character that they take on the role of in order to take part in a story that someone has created. This person is known as a player in the game, while their character is known as the player character (PC). As the name suggests, a PC in a role-playing game takes on a role in the story and generally has some sort of impact on the world. Every action the PC makes changes how the story is developed.

Tabletop role-playing games are a form of role-playing games where the player describes what their character does in each scene through speech. Players are normally free to take whatever action they wish and it is the job of the Gamemaster (GM) to create an interesting story around their actions and location.

1.2 The problem

When playing in a new game, a player's first step (creating a character) can be the most difficult and confusing if they are not familiar with the system they are playing in. When going through the character creation process, they must go through multiple steps which means that they can easily overlook or miss a step if they do not know what they are doing. To overcome this, the GM or someone experienced with the system can guide the player to ensure they complete the whole process.

When it comes to the actual games, it can be confusing when a player wants to make a skill check with their dice as they have to work out how many dice they will be rolling. This can slow down the game and can create confusion if incorrect dice are rolled.

1.3 Aim and goals of project

As a GM myself, I am very familiar with this problem and its impact on games. Since I am most familiar with it, I will be looking solely at the character creation process of Fantasy Flight Games' (FFG) [1] Star Wars game system.

This project's main goal is to create a web application that will aid a player during the character creation process, help guide them when they are lost, and remove the need for another person to be present during this process. It will also be used to maintain these characters afterwards as well after the creation process has been completed. The application will streamline the process while making the options available to players (both experienced and new) clearer and flow more logically.

1.4 Audience

The main target audience of this application are players that are new to FFG's Star Wars game system, while also providing aid to experienced players who know the system already. In its final version, the application will replace the existing physical book version of the process by providing all the information that the book provides.

1.5 Scope of project

In order to create my desired application for this project, I chose five outcomes which I would like to meet. These are:

Design and develop a UI – The UI should be functional and clear to present data and character options to the player with ease.

Present choices to player – The application should retrieve data from the database and present said data to the user in a clear UI.

Create character – The application should allow the user to create a character by choosing from options in each step and save it to the database.

View and maintain character – The application should allow the user to view and make changes to their existing characters that are stored in the database.

Use of character in game – The application should allow the user to make dice rolls, buy equipment and spend Experience Points (Exp) based on their character in the database.

1.6 Approach

I started the project by completing background research on existing solutions to my problem. I also created a brief questionnaire based around the character creation process of the FFG Star Wars game system, which focused on the aspects of their character that players would like to see more often.

To reach my goal of the project, I used a Test driven development approach that allowed me to test the application regularly to ensure each module met its requirements. This was vital to the application as modules each have a set function and purpose and are used in various combinations where they call upon each other, meaning that if one module fails then functionality can be affected.

1.7 Assumptions

Throughout the project, I made very few assumptions. The application I have created is designed to be usable by both experienced and inexperienced players of the game system. I therefore did not make any assumptions about the user's prior knowledge of the game or the character creation process. A key assumption that I have made is that the user will prefer to use a digital tool for character creation as opposed to the more traditional pen and paper method. This is important as it makes my application defunct if the user will not want to use it.

2 Background

In this section I will give background information that ties more closely into my project and the application that I want to create. This includes how the FFG Star Wars system functions which is what I will have to mimic in my application. After this I will be talking about existing solutions that are similar to the application that I intend to create and tools that will help me achieve my target goal.

2.1 The FFG Star Wars game system: Character Creation

To create a character using the FFG Star Wars system, the player must go through 9 steps.

Step 1: Determine Background

In this step the player must choose from a list of backgrounds from which the character may have come from. These can affect the character's story in the game and the way they act in certain situations. Along with the character's morality and motivation, this is used to build the character's personality.

Step 2: Determine Morality

Unique to the FaD system, morality tracks how good or evil a character is. A character's morality can affect other decisions the player makes during character creation, such as the type of character they will play, the starting skills, and gear that they choose.

Step 3: Select a Species

A character's species establishes their initial characteristics of Brawn, Agility, Intellect, Cunning, Willpower and Presence. Secondary characteristics such as Wound threshold and Strain threshold are also determined by the character's species. Each species has unique abilities that separate them from others.

Steps 4 and 5: Select a Career and Specialisation

Steps 4 and 5 are handled together, but represent two distinct choices that shape a character.

The career a character chooses reflects their approach to overcoming challenges and represents a broad archetype of related skills and abilities. Each career can be explored through a range of specialisations that share a common set of skills and focus on similar talents. The choice of career establishes the central focus of the character's training and professional experience. Each career has six associated skills which are noted down as career skills. During character creation, the player may choose four of these skills and gain one rank in each of them. When spending experience points (Step 6), career skills are less expensive to train and improve than non-career skills.

If a career is a broad archetype, then a specialisation is the additional materials added to it. While linked to a careers approach to problems, each specialisation takes a different, more focused approach to a particular aspect of the career. For example, a career may focus on combat as a whole but the specialisations could focus more on close range combat or long range combat. Each specialisation within a career possesses a unique talent tree which characters gain access to when they acquire the specialisation. The character also gains access to four additional career skills. During character creation, the player may choose two of the skills granted by their first specialisation and gain one rank in both of them.

Step 6: Invest Experience Points

The initial Exp available to a character is determined by their choice of species. These points can be used to improve certain aspects of the character i.e. increase characteristics, purchase additional ranks in skills, acquire talents, learn new specialisations, or gain new Force Powers (if they are force sensitive). Players may spend their Exp in any combination of these areas in order to create their unique character. After character creation, during gameplay, the character can gain additional points which they can spend on all the same aspects apart from characteristics.

Step 7: Determine Derived Attributes

Once the player has completed steps 1 to 6, several attributes can be determined. These are Wound threshold, Strain threshold, Defence, and Soak.

Step 8: Determine Motivation

Similar to the Background and Morality options, a character's Motivation is a choice which determines why they have accepted the call to take action and experience adventure. Motivation plays a key role in character progression. If a character acts true to their Motivation, then they may earn additional Exp at the end of the game session.

Step 9: Choose Gear and Equipment

Once the player has made their choices for steps 1 to 8, they can choose the gear that their character will bring into the game. Each PC starts with 500 credits to spend on any gear and weaponry they want. Some GMs will add restrictions to the available gear in order to fit the setting that they will be running their game in.

2.2 The FFG Star Wars game system: The Dice

When a character makes a skill check in the FFG Star Wars system, the dice allow the players to easily see whether they have succeeded or failed. The FFG system uses seven types of dice to accomplish this. Each die has a special function and purpose along with the symbols on them. These dice are split into 3 groups: Positive dice (Ability, Proficiency and Boost), Negative dice (Difficulty, Challenge and Setback) and the Force Die. The symbols shown on each dice show the overall result of the skill check. When looking at the result of the dice pool, which is the combination dice rolled by the user. The symbols can cancel each other out to give the final result. Success cancels Failure and Advantages cancel Threat, and vice versa.

[5] FFG Star Wars dice

The Ability Dice - These form the basis of most dice pools and represent the character's aptitude or skill when making a skill check.

The Proficiency Dice – These represent the combination of the character's ability and training when making a skill check with a skill they are trained in.

The Boost Dice – These are special situational advantages. They are benefits gained through luck, chance, and advantageous actions taken by the character.

The Difficulty Dice – These show the inherent challenge of a particular task a character is trying to overcome. They are the basic dice that are used against characters.

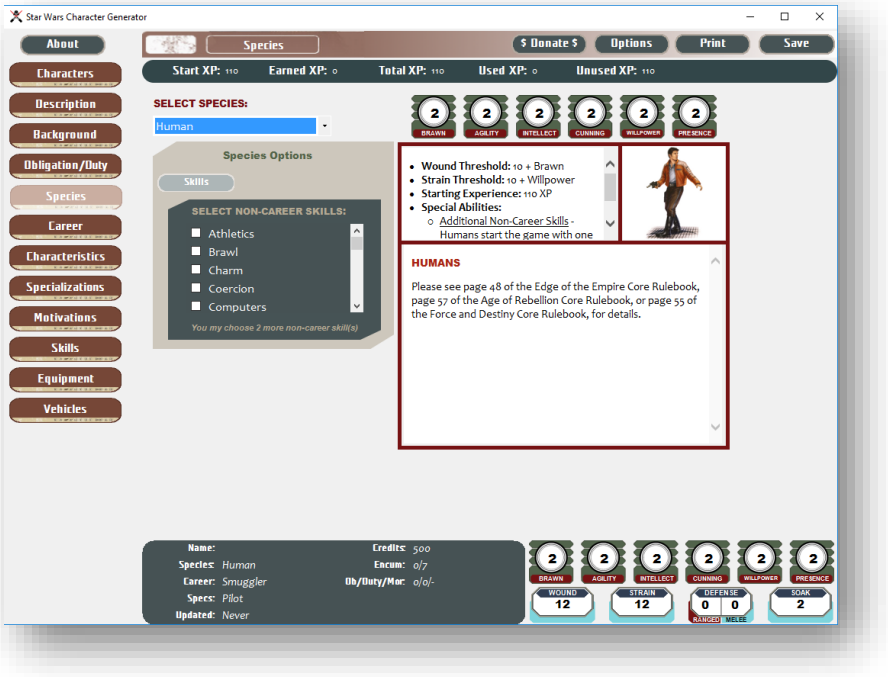
The Challenge Dice – These represent a more difficult version of the Difficulty Die. They may be used in particularly daunting challenges posed by trained or elite enemies.

The Setback Dice – These are the negative version of the Boost Die. Where a Boost Die is from advantageous actions, the Setback Die is from problems or obstacles that block the completion of the task that the character wants to resolve.

2.3 Existing solutions

There have been several similar applications of character creation aids similar to the one I have created. The main three that I have researched into are:

2.3.1 SW:EotE Character Generator by OggDude [2]



This offline application supplies the user with a platform to record their choices during character creation and provides a local database of all the choices the user has available to them, such as gear, careers and talents. It is constantly being updated by the creator whenever new content for the game is

released. The fact that it remains offline allows users to access its features even if they do not have an internet connection, however this means that they cannot access their characters from other devices.

The application does not provide any guidance or further information for creating a character, the user can see their choices but cannot receive any help or information from the application if they need it. When looking at their skills, the user can see the dice they should roll for that skill but the application does not provide a built-in dice roller for this.

OggDude's approach to a character creation tool is essentially what I wanted to improve upon with the guiding aspects of my application.

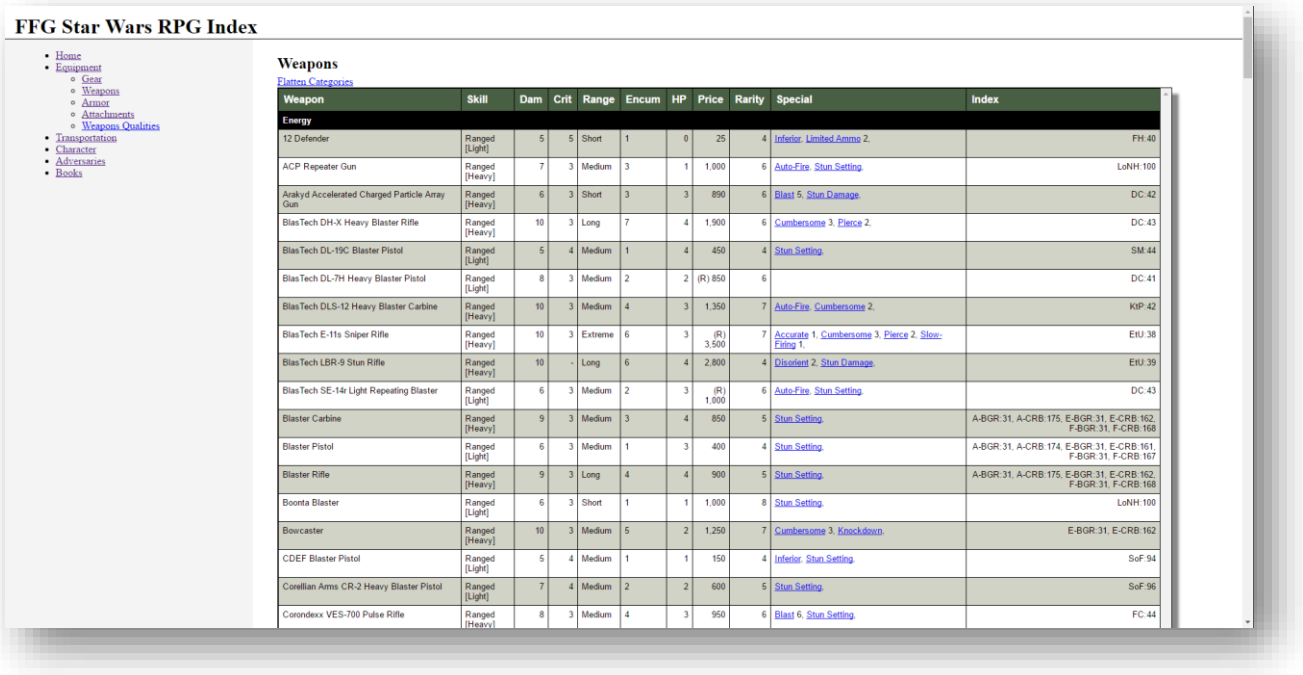
2.3.2 Star Wars: Edge of the Empire (API-compatible) for Roll20 by Steve Day [3]

Roll20 is a virtual tabletop which allows players to communicate, share images and manage games while online. This character creator approach makes use of the Roll20 API to supply its users with a character sheet that they can use to note down the choices that they have made during character creation. As the sheets are hosted on Roll20, the users can save their characters to the web application's cloud storage. These sheets are also interactive which means that they have integrated dice rolls which improves the in-game use of the sheet as users can click a simple button rather that calculate the dice they need to use. In addition to this, the sheet provides small help boxes that the user can look at for brief information on certain aspects of the sheet.

On the downside, the tool still requires the user to have a rulebook in front of them that they can use for guidance as it does not supply the user with their available options, such as talents and careers, or stats for gear. Instead the tool requires the user to input the details of their gear manually which means that custom gear can be created and added to the character. Another disadvantage of this tool which makes it less appealing than other aids is the fact it requires a paid subscription to Roll20 to include the sheet in a game. This can turn users away as there are other free alternatives available.

Steve Day's approach to a character aid holds the interactivity that I aimed to include in my final application, especially the dice rolling component.

2.3.3 FFG Star Wars RPG Index [4]



FFG Star Wars RPG Index

- Home
- Equipment
  - Gear
  - Weapons
  - Armor
  - Attachments
  - Weapons Qualities
- Transportation
- Character
- Adversaries
- Books

Weapons

Flatten Categories

Weapon	Skill	Dam	Crit	Range	Encum	HP	Price	Rarity	Special	Index
Energy										
12 Defender	Ranged [Light]	5	5	Short	1	0	25	4	Inferior, Limited Ammo 2.	FH:40
ACP Repeater Gun	Ranged [Heavy]	7	3	Medium	3	1	1,000	6	Auto-Fire, Stun Setting.	LoNH:100
Arakyd Accelerated Charged Particle Array Gun	Ranged [Heavy]	6	3	Short	3	3	890	6	Blast 5, Stun Damage.	DC:42
BlasTech DH-X Heavy Blaster Rifle	Ranged [Heavy]	10	3	Long	7	4	1,900	6	Cumbersome 3, Pierce 2.	DC:43
BlasTech DL-19C Blaster Pistol	Ranged [Light]	5	4	Medium	1	4	450	4	Stun Setting.	SM:44
BlasTech DL-7H Heavy Blaster Pistol	Ranged [Light]	8	3	Medium	2	2	(R) 850	6		DC:41
BlasTech DLS-12 Heavy Blaster Carbine	Ranged [Heavy]	10	3	Medium	4	3	1,350	7	Auto-Fire, Cumbersome 2.	KtP:42
BlasTech E-11s Sniper Rifle	Ranged [Heavy]	10	3	Extreme	6	3	(R) 3,500	7	Accurate 1, Cumbersome 3, Pierce 2, Slow-Firing 1.	EtU:38
BlasTech LBR-9 Stun Rifle	Ranged [Heavy]	10	-	Long	6	4	2,800	4	Disorient 2, Stun Damage.	EtU:39
BlasTech SE-14r Light Repeating Blaster	Ranged [Light]	6	3	Medium	2	3	(R) 1,000	6	Auto-Fire, Stun Setting.	DC:43
Blaster Carbine	Ranged [Heavy]	9	3	Medium	3	4	850	5	Stun Setting.	A-BGR:31, A-CRB:175, E-BGR:31, E-CRB:162, F-BGR:31, F-CRB:168
Blaster Pistol	Ranged [Light]	6	3	Medium	1	3	400	4	Stun Setting.	A-BGR:31, A-CRB:174, E-BGR:31, E-CRB:161, F-BGR:31, F-CRB:167
Blaster Rifle	Ranged [Heavy]	9	3	Long	4	4	900	5	Stun Setting.	A-BGR:31, A-CRB:175, E-BGR:31, E-CRB:162, F-BGR:31, F-CRB:168
Boonta Blaster	Ranged [Light]	6	3	Short	1	1	1,000	8	Stun Setting.	LoNH:100
Bowcaster	Ranged [Heavy]	10	3	Medium	5	2	1,250	7	Cumbersome 3, Knockdown.	E-BGR:31, E-CRB:162
CDEF Blaster Pistol	Ranged [Light]	5	4	Medium	1	1	150	4	Inferior, Stun Setting.	SoF:94
Corellian Arms CR-2 Heavy Blaster Pistol	Ranged [Light]	7	4	Medium	2	2	600	5	Stun Setting.	SoF:96
Corondexx VES-700 Pulse Rifle	Ranged [Heavy]	8	3	Medium	4	3	950	6	Blast 6, Stun Setting.	FC:44

While not a dedicated character creator in the same sense as the SW:EotE Character Generator by OggDude, this tool is simply an online database of all the gear and vehicles available for purchase by players for their characters. The website also contains NPCs and their stats for GMs to use in their games as quick and easy references.

This idea of a database of all the stats a player would need being easily accessible is the element that I wanted to include in my application.

2.4 Target audience questionnaires

While doing background research relevant to my project, I created questionnaires to be completed by the intended audience of my application. The questions were focused on a player's view of character creation and the difficulties they come across. They also included how the player felt about existing solutions.

From the responses I received, I found that players had a hard time due to the fact that the FFG system uses multiple expansion books that add onto the core rule book. This means that the player has to use multiple sources to create their character rather than have one source for all the information. I also found that the most requested pages are the skills and gear pages. I believe this is due to the fact that players use these the most when in games and need them to make skill checks or when they are in combat and need to be able see their weapons. The respondents also found that existing solutions don't display gear clearly. From the responses I have found that existing solutions give too little space for gear which means that players can't space out their gear and that therefore it is cramped and difficult to read. The last thing I found from the responses was the player need for a "Cheat sheet" for use in game

combat. This would allow the player to see exactly what they can do on their turn of combat and how difficult it would be.

All of these responses were very useful and I believe that I can include them all in my application and there by improve upon existing solutions and meet player needs more closely than these other solutions.

2.5 Bootstrap [6]

When it came to coding the UI, I chose to use the Bootstrap framework. This was because it comes with CSS and JavaScript code built-in which reduced my front end coding significantly. The CSS allows the layout to be dynamically adjusted on the page and dependent on the size of the user's window the page is being displayed in. The default design allowed me to create a clear UI that was easy to navigate, so I decided to stick to the default Bootstrap design as the main focus of the application was not the UI. The framework fits all my needs for the front end code and also came with the additional benefit of the packaged Glyphicons which let me include icons on my pages in order to improve the aesthetics of my pages. The documentation of Bootstrap is also very detailed and shows clearly (with examples) how to include all of the CSS, JavaScript, and Components included in the framework.

2.6 Microsoft Access [9]

In order to store the data for the web application I needed a database management system. I chose to use Microsoft Access for this as it has a clear graphical user interface that display the structure of my database and makes it very easy to input data into the tables within. This is very important to me as someone who has little experience with database management. Microsoft Access allows me to easily navigate around my database and make changes to the structure which is extremely useful when creating a relational database that has table which rely on other each other.

An additional feature that Access includes is the Query wizard. This allows me to create queries from the database with minimal effort. I can simply choose the tables I want and the fields I want to return and Access creates the query. The wizard returns the SQL query and also the fields that would be returned.

The reason that I chose Access is that it is more suitable for applications with a small number of users. However, I feel that using Access wouldn't be appropriate for a larger web application, but I believe that this is not a problem for my project as I will be the only user accessing the database directly.

2.7 XAMPP [8]

For the application to be accessible by multiple users, I decided to use XAMPP for the web server side. XAMPP is a package that contains several pieces of software. For my specific task, I used the Apache web server component. This simply allows me to host my application for multiple users to access and runs my PHP code. This choice of web server came from recommendations by people who have used it in the past and have a lot more experience with web servers than myself.

3 Specification and Design

In this section I will be describing the specification I have chosen for this application. I will also go over the design and will be justify the choices I have made. This includes the database design and the user interface design.

3.1 Use Cases

These are based off the Scope of the project as detailed in section 1.5. The Scope acts as both the goals of the project and the requirements that need to be met. The following use cases have been designed around this Scope.

Use Case: 1	**User sign in**
Description:	The user signs into the system and is set as the current user so that they can view data relevant to them.
Main flow:	User enters their username
	User enters their password
	User clicks the “Sign In” button and are then taken to their home page
	User’s characters and games (if they have existing ones in the database) are displayed

Use Case: 2	**Create a new character**
Description:	The user creates a character that they can access later and make changes.
Main flow:	User clicks on the “New Character” button
	Create character modal window is displayed
	User enters the name of their new character
	User selects the game they will be playing in
	User clicks the “Save” button and are taken to the “Backgrounds” page
	User completes Use Cases 5 to 13

Use Case: 3	**Maintain an existing character**
Description:	The user selects an existing character and can make changes to them.
Main flow:	User clicks on the button in the “Go to…” column of their character table
	User selects the step they wish to change or view from the navbar
	User selects their new choices

Use Case: 4	**Delete an existing character**
Description:	The user can delete characters that they no longer need.
Main flow:	User clicks on the red “Trash” icon next to their character
	Delete character modal window is displayed
	User confirms the character they are deleting by clicking the “Yes” button

Use Case: 5	**Set a character’s background**
Description:	The user sets their active character’s background.
Main flow:	User selects the tab that their desired background is under

	User selects the background they want for their character
	User confirms their choice by clicking the “Save” button

Use Case: 6	Set a character’s morality
Description:	The user sets their active character’s morality.
Main flow:	User clicks “New” button
	New Morality modal window is displayed
	User finds the morality they want and clicks “?” button
	Morality modal window is displayed with details on the selected morality
	User selects the morality they want for their character
	User confirms their choice by clicking the “Save” button

Use Case: 7	Set a character’s species
Description:	The user sets their active character’s species.
Main flow:	User selects their character’s species from the dropdown menu
	User checks the stats of the species they chose
	User confirms their choice by clicking the “Save” button

Use Case: 8	Set a character’s career and specialisation
Description:	The user sets their active character’s career and specialisation.
Main flow:	User selects their character’s career from the dropdown menu
	User selects the 4 career skills they want
	User selects their character’s specialisation from the dropdown menu
	User selects the 2 career skills they want
	User clicks “Talent Trees” button
	User complete Use Case 9
	User clicks “Force Powers” button
	User complete Use Case 10
	User confirms their choices by clicking the “Save” button

Use Case: 9	Set a character’s talents
Description:	The user sets their active character’s talents.
Main flow:	User selects the specialisation talent tree they want to buy from
	User selects the talent they want
	User confirms their choices by clicking the “Save” button

Use Case: 10	Set a character’s force powers
Description:	The user sets their active character’s force powers.
Main flow:	User selects the force power they want to buy from
	User selects the upgrade they want
	User confirms their choices by clicking the “Save” button

Use Case: 11	Set a character's characteristics and skills
Description:	The user sets their active character's characteristics and skills.
Main flow:	User clicks "-" or "+" buttons to increase or decrease their characteristics
	User clicks "-" or "+" buttons to increase or decrease their skills
	User clicks "Roll" button to test their skill
	User confirms their choices by clicking the "Save" button

Use Case: 12	Set a character's motivation
Description:	The user sets their active character's motivation.
Main flow:	User clicks "New" button
	Motivation type modal window is displayed
	User clicks the motivation type they want for their character
	New motivation modal window is displayed
	User finds the motivation they want and clicks "?" button
	Motivation modal window is displayed with details on the selected motivation
	User selects the motivation they want for their character
	User confirms their choice by clicking the "Save" button

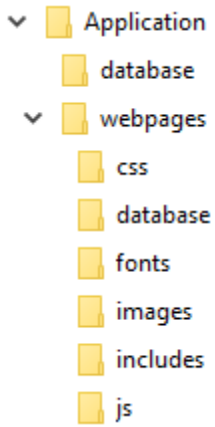
Use Case: 13	Set a character's equipment
Description:	The user sets their active character's equipment.
Main flow:	User clicks "New" button under "Weapons"
	User clicks "Add" button for the weapon they want
	User clicks "New" button under "Armor"
	User clicks "Add" button for the armor they want
	User clicks "New" button under "Gear"
	User clicks "Add" button for the gear they want
	User repeats if they want more equipment
	User confirms their choice by clicking the "Save" button

3.2 Architecture of system

The system is separated into three parts. The first part is the Database. This stores all the data for web application that isn't static on the HTML webpage. The second part is the Middleware. This includes all the SQL queries that select data from the database to be displayed on the webpages and the queries that send data back into the database to be stored. The third part is the Front End Webpage. This contains all the HTML and CSS that configures the layout of the webpage to ensure that the user can clearly see all their options on the webpage and that no information on the webpage is obstructed. Each part interacts with its neighbour(s) to create the web application.

3.2.1 How data flows through the system

In terms of the file structure of the application, I stored the Access database above the webpage root directory. This ensured that users couldn't gain direct access to the database. I then stored the webpages in the root directory. This included all the HTML webpages as PHP files since the pages have PHP that must be run with the HTML.

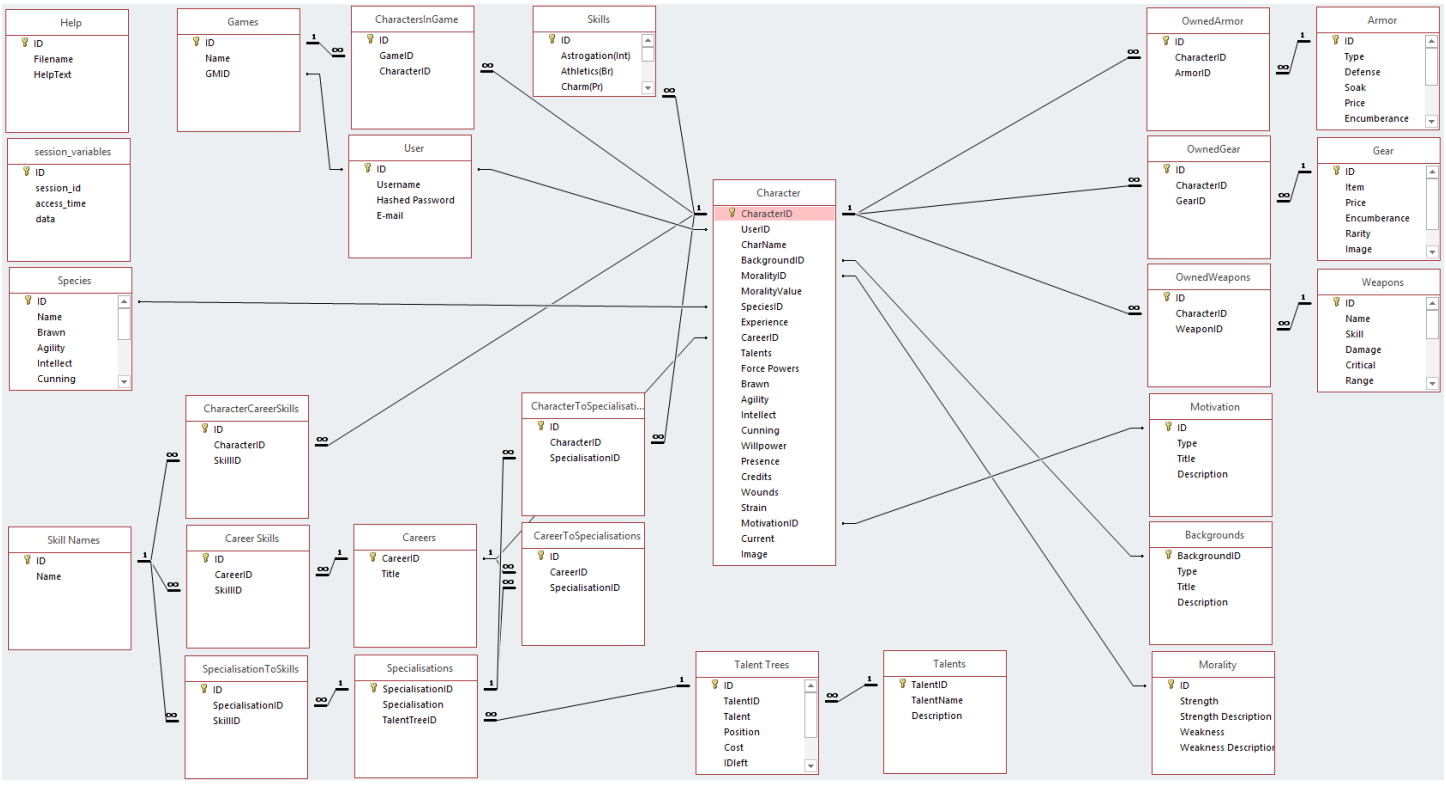


Within the root, I created a database folder which stores all the PHP database functions that contain the SQL queries that are used by the application. These queries return the data in associative arrays which can selectively be accessed by the front end webpage. This allows me to select all the data from a table using a query and then display the data in the format I want on the webpage. The benefits of this is that it separates the SQL from the formatting of the data on the page. The queries are all stored in functions so that they can be called from the webpages to get data from or send data to the database. This is a vital part of the application as users need to be able to create characters and have their character choices saved. Each webpage has a PHP file associated with it which contains all the queries for that page, for example the "Careers.php" page has a "TableCareers.php" file in the database folder with the queries for that page. This kept the folder structure tidy and improves the maintainability as it makes it easy for me to find the queries. In these "Table___.php" files, I gave the table and each field in that table a PHP constant name using the Define function. These constant names were used in all the queries I wrote and meant that I could change the names of fields in the database and I would only have to change the defined constant associated with this field. This saved time as a table or field can be used multiple times and I only have to change the name once in the define statement.

Also in the root I have a folder called "Includes". This contains the header navbar and any other modules that run on multiple webpages. These can then be called by any of the webpages that need to include the code in that file.

3.2.2 Database Relationship Diagram

The database started as simple tables joining to the central character table. But as I started developing sections of the application, I found that I needed to rearrange the structure to normalise the data and enforce integrity between database table fields. Over time the database developed into a fully relational database removing the threat of the duplication of data and ensuring that information is only present in a single place rather than multiple.

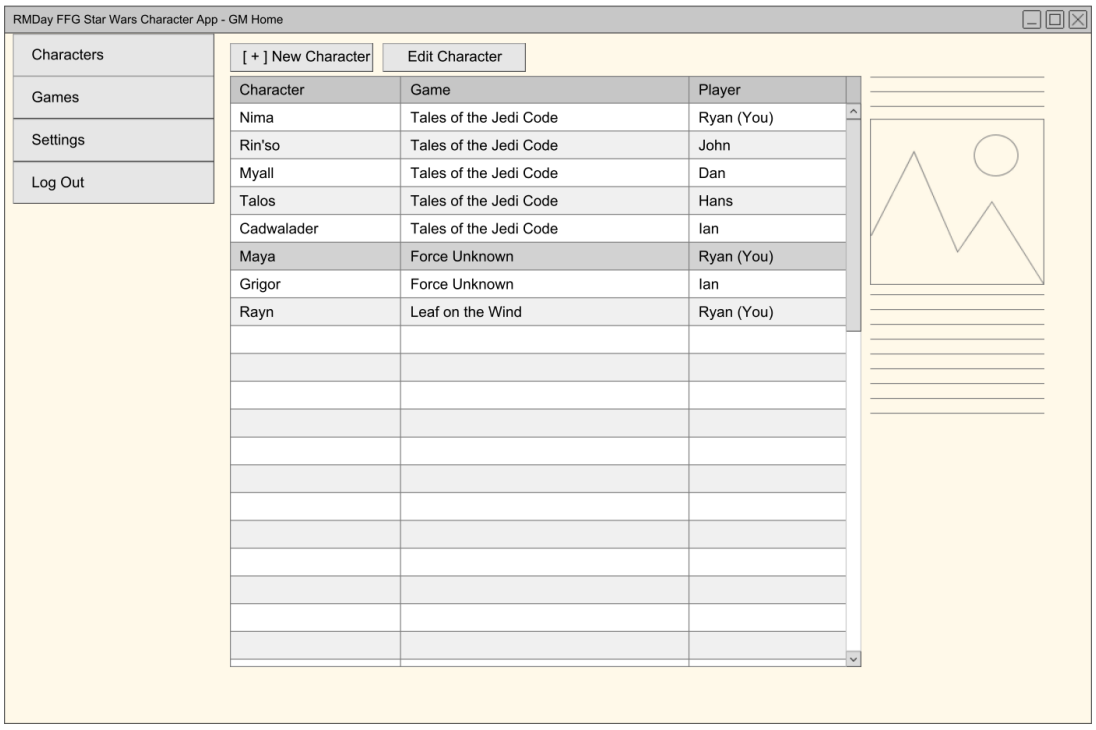


3.2.3 User Interface

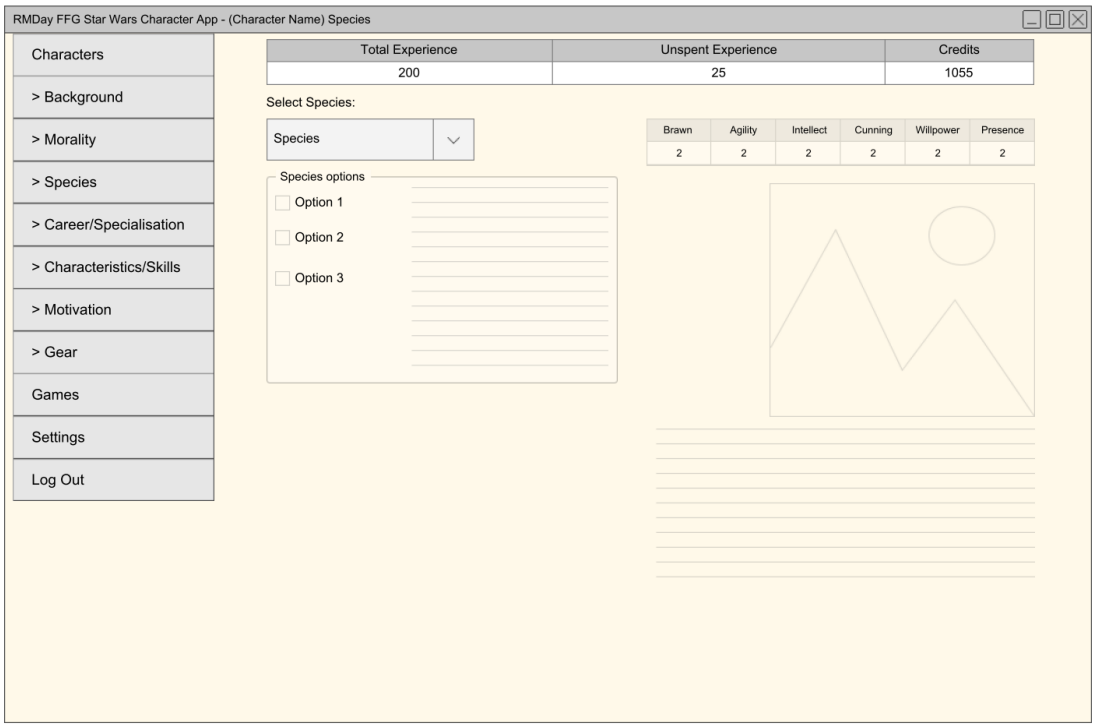
Before beginning my coding of the UI for the web application, I started by creating design mockups. I used the UI prototyping tool ForeUI [7] to arrange the components of my pages. An additional benefit of the tool is that I could include links on each of my mockup pages. This allowed me to simulate the flow of the web application to test how the buttons would work and how the user would move through the pages. The tool had some drawbacks though. I found that as I created more pages and included more components on each page, the tool struggled to keep up and became progressively slower. This inhibited my progress and meant that could only work so far through the designing process before the tool became near to unusable. On top of this, I was under the time constraints of the ForeUI free trial which would severely reduce the functionality of the tool when it came to an end.

When deciding on the layout of the page elements, I took inspiration from OggDude's Character Creator (see section 2.3.1). This was due to the application having a generally good layout which was easy to navigate. This included the navigation bar which allows the user to easily move around their character to see the details they want. I believe that users of digital Character Creator tools (such as OggDude's) would be familiar with this layout, so when they come to use my application they would be comfortable with the navigation of the system.

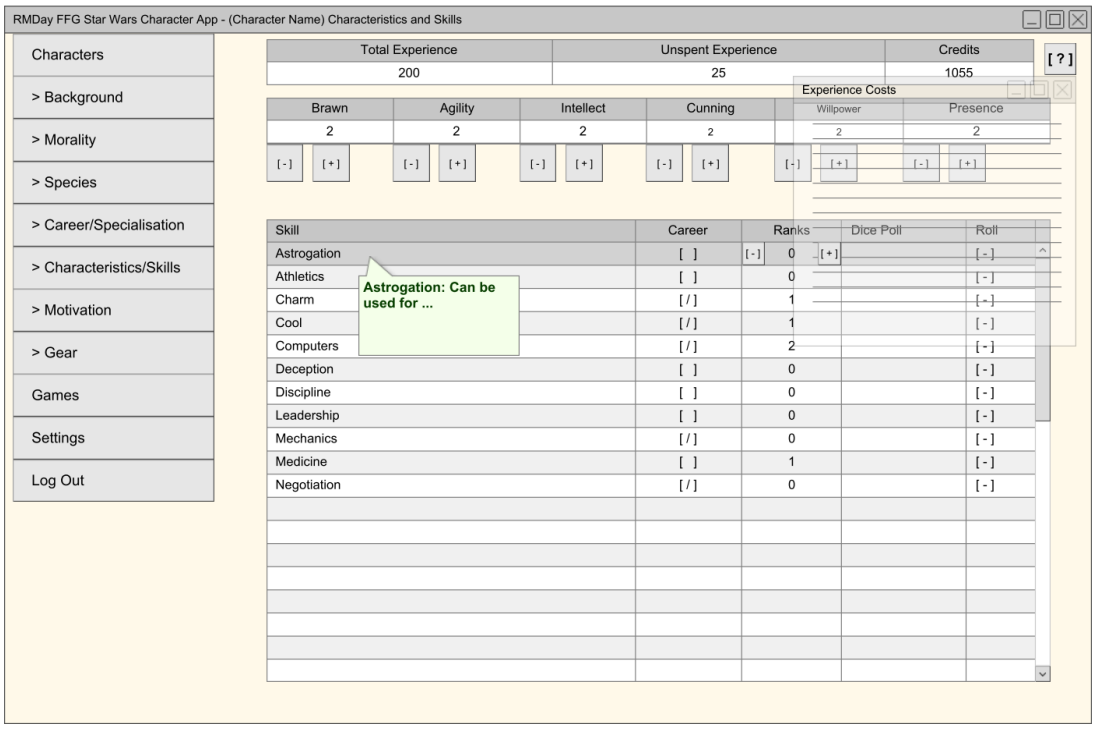
GM Home



Character 3 Species



Character 5 Characteristics/Skills



4 Implementation

In this section I will be detailing key segments of code that I used throughout the implementation of my application. This won't include every single bit of code I used but rather just the key bits that are vital to the system fulfilling the requirements of the project. I will also detail the problems that I had to overcome and slowed my progress.

4.1 Approach

When I came to the implementation of the application, I decided to divide the whole application into three parts. This allowed me to set goals for the project. These were Creating a character, Maintaining a character, and Using a character. Creating a character included retrieving data from the database, displaying it on the webpages and sending the player choices back to the database. Maintaining a character was similar to the latter part of Creating a character but it required the user to be able to view existing/created characters and make changes to them which would be stored in the database. Using a character included the more advanced functionality that would be used when the user is in a game, such as rolling dice in the application.

To save myself work, I used self-documenting code. I used sensible names for variables and functions which achieves this, meaning that when I come back to code that I wrote weeks ago, I understand what the purpose of the code was and what it's function is on the page. This also reduced the need for comments in my code as the function/variable names are descriptive.

4.2 Front end UI

I started with the front end code of the application. This included the HTML, CSS, JavaScript and some PHP to display the information and create a layout that would adjust when displayed on a smaller device or a window. I kept the page structure consistent by placing all the database code at the top of the PHP script before any HTML code is displayed. This code consisted of my Requires (accessing the database, accessing the session variables, and creating the navbar in the header) and any code that retrieves information from the address bar.

I created a file called "header.php" which is used on every page to display the navbar. Not every page has the same links in the header though. In order to get around this, the file compares the current page name with a set name depending on the page and header required. This enables a different header to be displayed on the index page and for the current page to be highlighted in the navbar.

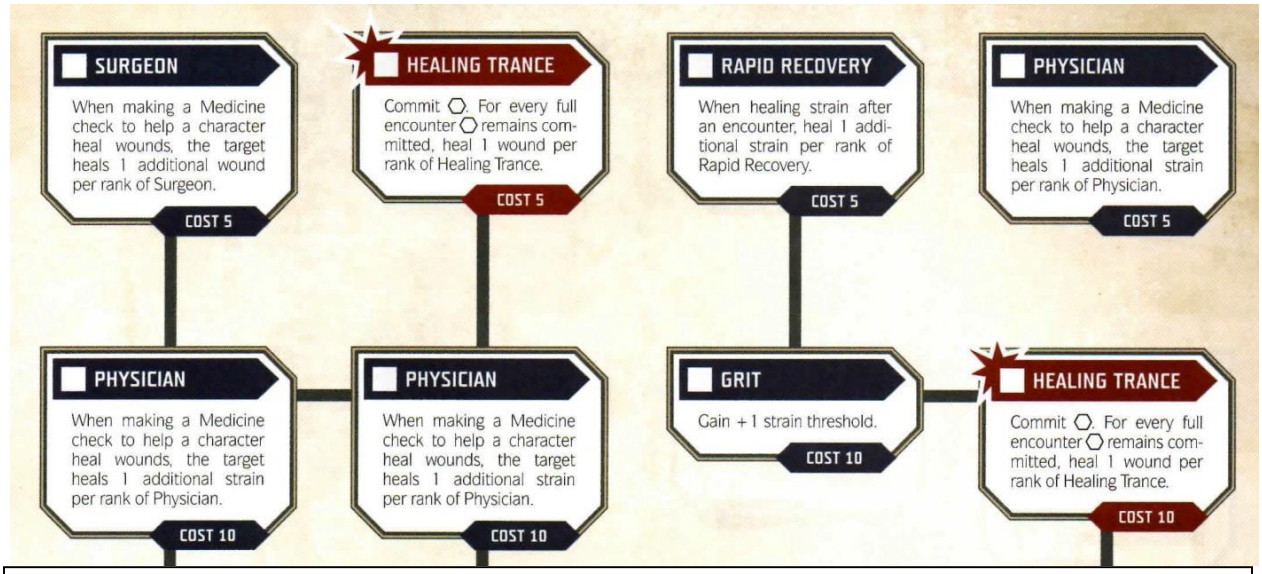
I used HTML Forms to retrieve information from the choices the user makes. Using the "get" method, I was able to get the choice of the user. This was displayed in the URL when the form was submitted. The JavaScript "onclick" and "onchange" functions are used to submit this choice without the user having to click a separate submit button. The page dynamically loads new information according to the user's choice. I used this information with the PHP "filter_input" function to validate the input and set the choice as a variable to be used in the middleware. An example of this occurring is on the "Careers and Specialisations" page. This page initially displays a dropdown menu of all the careers so that the user can pick one. By using "onchange", the choice is submitted and the ID associated with that choice is displayed in the address bar. With this, I used "filter_input" to give the choice a variable name which can be used by the middleware. Once the middleware uses the ID, career skills associated with that career ID are displayed with checkboxes as well another dropdown menu with the career's specialisations. The user can then pick the skills of their choice and select a specialisation. More checkboxes are generated in

the same way as before, but a “Save” button is also displayed. Once clicked, this button submits the form with “onclick” and the variables assigned to the choices are passed to the middleware to be saved to the database. The user is then taken to the next page automatically.

For the Talent Trees and Force powers I used a table to display the trees. I created the table as so:

talent name	talent name	talent name	talent name
talent description ↔	talent description ↔	talent description ↔	talent description
talent cost	talent cost	talent cost	talent cost
↕	↕	↕	↕
talent name	talent name	talent name	talent name
talent description ↔	talent description ↔	talent description ↔	talent description
talent cost	talent cost	talent cost	talent cost
↕	↕	↕	↕

This would be populated with data retrieved by the middleware from the database and would look like this:



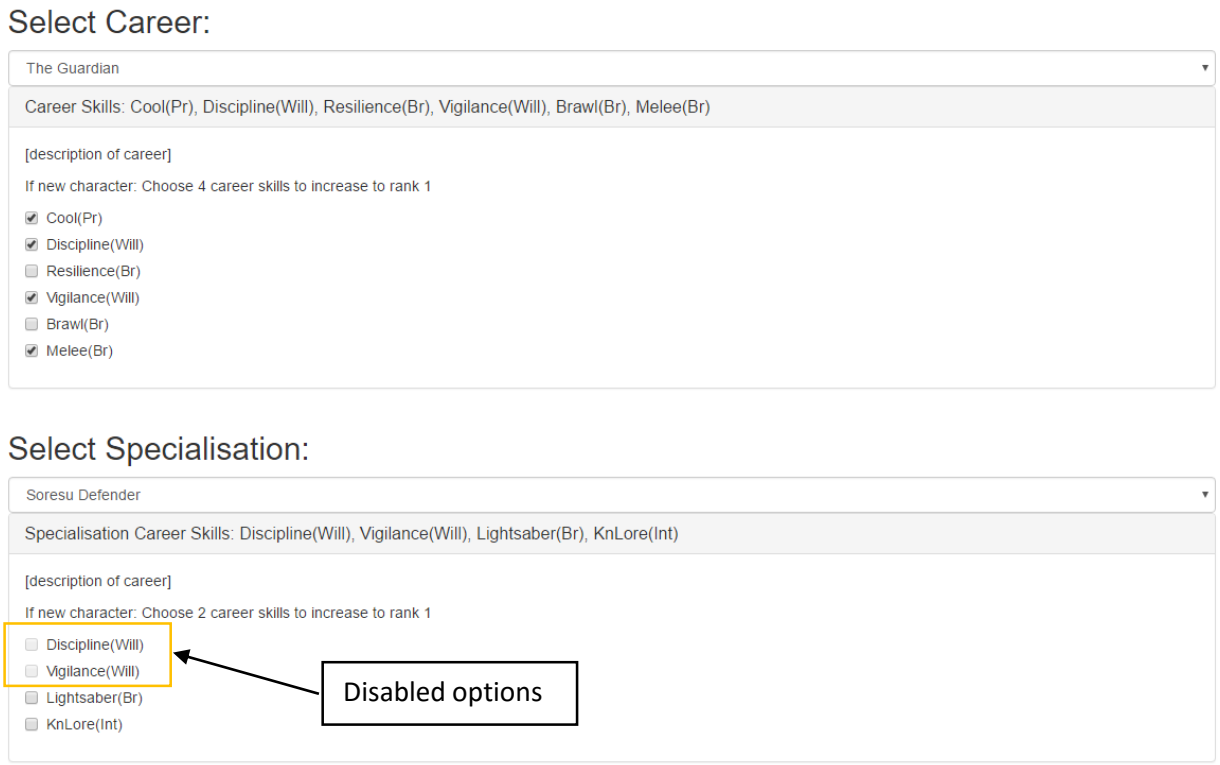
[12] page 69: Consular: Healer Talent Tree

The arrows would be determined by the middleware according the database values (explained in section 8.4).

In order to create a character, the user must set the game they will be playing in. Before they can do this though, the game must be created. Rather than giving all GM’s direct access to the database so that they can add their game, I created a button that triggers a modal with text input fields that allows the GM to write the name of their game and create it.

For some sections of the character creation process, the player is limited in their choices (for example, they cannot have a negative number for the ranks in a skill). This occurs on the Careers page. The player can only pick a skill once out of the choices they have, so if a skill appears in both the Career and Specialisation then the second occurrence of the skill needs to be disabled. By using JavaScript, I was

able to check the selected skills from the Career Skill choices and disable the skills that have already been selected in the specialisation career skills section.



As mentioned before, this limitation also occurs on the Characteristics and Skills page. The player cannot have negative ranks in a Characteristic or Skill. To stop the player getting these, I set a minimum value of 0 in the database for these fields. In the HTML for the "-" button, I included a PHP script that checks the current value. If the value is equal to 0 then the "-" button is disabled, thereby stopping the player from setting the rank to a negative number.

4.3 Back end middleware

The middleware handles the data going to and from the database and acts as the step between the queries and webpages. As mentioned earlier in this report, all queries are located in the database folder. These queries return the data from tables in the database and nearly always return all the in the table. This is returned in an associative array as raw data straight from the database. Therefore, I needed to filter the data and display it in the correct places on the screen. This is done through the use of iterative functions which loop through the rows of the data returned. For example, in the weapons table I used:

```
<?php
        $rows =  getAllWeapons($dbHandle);
        foreach ($rows as $row) {
        ?><tr>
                <td><?php echo $row[WEAPON_NAME];?></td>
                <td><?php echo $row[WEAPON_SKILL];?></td>
                <td><?php echo $row[WEAPON_DMG];?></td>
                <td><?php echo $row[WEAPON_CRIT];?></td>
```

```
…
<?php
        }
?>
```

This lets me display the data I want, in the order I want by looping through the associative array and echoing the field name I want to display.

As the current character being displayed is a crucial part to the data being displayed, this is set as a session variable. At the start of each page where the character is being edited, the session variable is given the variable name “$currentCharacter”. This lets me include it when I call functions so that the character ID can be passed to the query that is retrieving data for that character.

In order to track the currently logged in users, I created a modal on the index page that retrieves the currently active users and displays them. The modal also displays whether they are a GM or a player. I have stored all current users and the characters they are using as session variables. This allows the user to keep editing their character as they move through the character creation process. It also allows me to get the current users and display them in the currently active users modal. I used the PHP time function to check if the user is currently active by seeing when they signed in.

```
<?php
        $activeForSeconds = time() - 900;
        $rows = getAllLoggedInUsersData($dbHandle, array($activeForSeconds));
        if ($rows !== false) {
                foreach($rows as $row) {
                        if(!is_null($row[SESSION_USER_NAME])) {
                                if ($row[SESSION_USER_GM]) {
                                        $userType = "GM";
                                }
                                else {
                                        $userType = "Player";
                                }
                                echo ($row[SESSION_USER_NAME].', '.$userType.'<br>');
                        }
                        else {
                                echo "Unknown User <br>";
                        }
                                        }
                                }
                                else {
                                        echo ("No body is logged in!");
                                }
?>
```

Throughout the implementation stage, I encountered many errors. These would be displayed on the page. In order to keep the UI clear, I created an ErrorLog PHP file that would store the errors in a text file rather than being displayed on the screen. This allows me to look up the errors later for example if a

user is having difficulties with the application, I can look up the error log and see if the database is erroring.

4.4 Database

When I wrote the programming for the database functionality, such as querying the database to perform actions, I used the Open Database Connectivity (ODBC) API to access the database. This API allowed me to write descriptive SQL queries which ODBC then converts to database commands that the Access database can interpret. The ODBC acts as one of the bridges between the middleware and the database by connecting the web application to the database, allowing the queries to function. This API is the means I used to get data to return so that I can display it on the web application. The other bridge between the middleware and the database is the PHP Data Object (PDO) class which allows me to add variables into my queries. For example, when I want to add gear to a character in the database, I used this function and query:

```
function addCharacterGear($_dbHandle, $_params) {
        $sql = "insert into ".OWNED_GEAR_TABLE."
                (".OWNED_GEAR_CHARACTER_ID.", ".OWNED_GEAR_GEAR_ID.")
                values (?,?);";
        …
}
```

The ?s are the variables that PDO allows me to use. These are passed to the function as the "$_params" argument. These are stored as an array which means that when the function is called, it must be in this structure:

```
addCharacterGear($dbHandle, array($currentCharacterID, $gearID));
```

The "$currentCharacterID" and "$gearID" are variables and act as the values that are in the query. So the database receives the query with the data stored in those variables instead of the ? in the function query.

The advantage of using PDO is that inverted commas in strings inputted by the user are escaped, preventing SQL queries from failing due to inverted commas being put of place.

When handling the deletion of characters on the application, I made a simple delete button and confirmation modal. In the database however, this marks the character current field as no. This in turn stops the character from being displayed on the application. This mean that I can retrieve characters that have been "deleted" and I can undo them, restoring them back into the application just by changing a single field.

For the Talent Trees and Force Power Upgrades I stored the data in the following way. I stored the Talents and Upgrades in tables respectively with each talent and upgrade having an ID. I then created a table for the Tree structure. In this table I stored the Talent ID, it's position in the tree from 1-20 (from left to right, up to down in the tree), it's cost, and Booleans for whether there is a talent adjacent to it

(left, right, below and above). This meant that I could create the tree display arrows in the UI according to the Boolean for the talents neighbours.

4.5 Unforeseen problems

Throughout the implementation stage of this project I came across multiple problems that caused me trouble.

TalentID	Position	Cost	IDleft	IDright	IDbelow	IDabove
1	1	5	0	0	1	0
2	2	5	0	0	1	0
3	3	5	0	0	1	0
10	4	5	0	0	0	0
4	5	10	0	1	1	1
1	6	10	1	0	1	1
11	7	10	0	1	0	1
5	8	10	1	0	1	0

The first was in XAMPP. As the application is meant to be usable by multiple users at one time, I needed external devices to access the application on the host computer. At first this would work and I couldn't even get into the XAMPP directory. After researching the issue on the internet, I found that the problem was with the firewall on the host computer. To fix the issue, I made an exception for the HTTP port that I was running XAMPP through and this solved it.

The next issues were with Microsoft Access. Throughout the programming of the queries for the database I came across countless SQL functions that the ODBC Microsoft Access Driver didn't support. This included the Join function which was very important in my relational database. This caused me trouble as I was forced to find alternate methods of retrieving the data I wanted. Once I managed to find a query structure that got me what I wanted, the issue wasn't a problem anymore as I could just use my query structure instead. Another function which wasn't supported was the SQL Describe function. I wanted to use this to retrieve the field names in a table, namely the characteristics fields in the character table, but as the function wasn't supported, I had to find another way. This way was by creating an array myself and filling this array with the field names manually. This fixed the issue but means that the array isn't synchronised with the database, so if I change the field names in the database then I need to manually update the array values.

When creating my queries, they were rarely perfect the first time I ran them, this meant that I saw a lot of the ODBC Microsoft Access Driver errors. This was my next issue. I came across the "COUNT field incorrect… Too few parameters. Expected 2" error message several times which was fairly unhelpful when trying to fix it. I eventually found that this was linked to the database query and meant that the query couldn't find a field that I had mentioned (meaning that I misspelled it) or that the variable "?" in the query wasn't being passed when the function was called. Once I had this idea of what the error meant, I could quite easily solve it as the problem was almost always in the query.

The final issue I had with Access was later in the implementation. When accessing the database, I found that it was giving me errors about an "id" field and stopped me from opening the tables. This confused me at first since the field that I had which were called ID were written in capital letters and the error was mentioning a lower case id. After searching for this issue on the internet I found that it was linked to the database being corrupted. This was obviously a big issue as the database took quite some time for me to

build and populate. I checked a backup database that I had made the day before the incident and it worked fine, so I tried using that in the place on the live database. As soon as the backup was in the system though, it started presenting the same error. After more researching I found that a way around this corruption was to make a new database and export the tables and data from the old one. This seemed to solve the issue but I had to recreate the relationship model.

When I was programming the ErrorLog, I had issues with giving the file a relative location. The ErrorLog file is required in the DatabaseAccess PHP file which runs on every page.

```
require_once('ErrorLog.php');
```

If I tried to store the ErrorLog in a location that wasn't the same as the DatabaseAccess file and give it a relative location, e.g.

```
require_once('..\ErrorLog.php');
```

then the application wouldn't be able to find it and would store the error in a new text file in the XAMPP directory. I wasn't able to fix this issue so I kept the file in the same directory as the DatabaseAccess file as the ErrorLog functions properly there.

5 Results and Evaluation

In this section I will be evaluating my application at its current state through the use of test cases. I will be showing the results of these tests reviewing how well it fits the scope I defined in section 5.5 and the specification use cases from section 7.1. I will also be comparing my application to existing solutions in order to check whether it improves upon them or not.

5.1 Testing

5.1.1 UI

This section contains tests relating to the UI and its functionality.

Test Case ID:	1	**Test Designed date:**	22/04/16
Test Priority:	Medium	**Test Executed date:**	26/04/16
Module Name:	UI		
Test Title:	Consistent Header across all pages		
Description:	Verify that the Header keeps the same styling on all pages		
Pre-Conditions:	None		
Dependencies:	None		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks on “Character” link in Header	Pass	
2	User can click any of the links from the dropdown menu under “Character” and they are taken to the corresponding page	Pass	

Post-Conditions:	User is on page relating to the link they clicked on

Test Case ID:	2	**Test Designed date:**	22/04/16
Test Priority:	Low	**Test Executed date:**	26/04/16
Module Name:	UI		
Test Title:	Mobile device UI layout adjustment		
Description:	Verify that the UI adjusts the page layout when the application is viewed on a mobile device		
Pre-Conditions:	User opens application on mobile device		
Dependencies:	None		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User can view all information on screen	Pass	

Post-Conditions:	No information is being hidden from the User due to screen size

From these test cases I can see that my UI fulfils the requirements that I set and that there are no issues in this area.

5.1.2 System

This section contains tests relating to the system and its functionality.

Test Case ID:	3	**Test Designed date:**	22/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	System		
Test Title:	User signs in		
Description:	User creates a new character by giving it a name and a game		
Pre-Conditions:	User is on the sign in page and exists in database		
Dependencies:	None		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User enters their username	Fail	User currently picks their username from a dropdown menu
2	User enters their password	Fail	No password system is in place yet
3	User clicks “Sign In” button	Pass	Button set current user according to their choice in the dropdown menu

Post-Conditions:	User is signed into system and is now located on their home page with their created characters on screen

From these test cases I can see that the application needs work on the sign in page. The database currently has the structure to store passwords and usernames but the application page for this only uses the username and displays all users in the system.

5.1.3 Creating a character

This section contains tests relating to the Character Creation content and its functionality.

Test Case ID:	4	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Create a new character		
Description:	User creates a new character by giving it a name and a game		
Pre-Conditions:	User is signed in and is on their home page		
Dependencies:	Test Case 3		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks the “New Character” button	Pass	
2	User enters the name of their new character	Pass	
3	User chooses the game they will be playing in	Pass	
4	User clicks “Save” button	Pass	

Post-Conditions:	Character has been created with user’s ID and user is now located on the backgrounds page

Test Case ID:	5	Test Designed date:	23/04/16
Test Priority:	High	Test Executed date:	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character background		
Description:	User chooses their character's background		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		

Step	Test Steps	Pass/Fail	Notes
1	User clicks the tab that contains the background they want	Pass	
2	User selects the background they want	Pass	
3	User clicks "Save" button	Pass	

Post-Conditions:	Character background has been saved in database and user is now located on the morality page

Test Case ID:	6	Test Designed date:	23/04/16
Test Priority:	High	Test Executed date:	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character morality		
Description:	User chooses their character's morality		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		

Step	Test Steps	Pass/Fail	Notes
1	User clicks "New" button	Pass	
2	User selects the morality they want	Pass	
3	User clicks "Save" button	Pass	

Post-Conditions:	Character morality has been saved in database and user is now located on the species page

Test Case ID:	7	Test Designed date:	23/04/16
Test Priority:	High	Test Executed date:	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character species		
Description:	User chooses their character's species		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		

Step	Test Steps	Pass/Fail	Notes
1	User selects the species they want	Pass	
2	User clicks "Save" button	Pass	

Post-Conditions:	Character species has been saved in database and user is now located on the careers and specialisations page

Test Case ID:	8	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character career and specialisation		
Description:	User chooses their character's career and specialisation		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User selects the career they want	Pass	
2	User selects the first set of career skills they want	Pass	
3	User selects the specialisation they want	Pass	
4	User selects the second set of career skills they want	Pass	
5	User clicks "Save" button	Pass	

Post-Conditions:	Character career and specialisation has been saved in database and user is now located on the characteristics and skills page

Test Case ID:	9	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character talents		
Description:	User chooses their character's talents		
Pre-Conditions:	User is signed in, character has been created and a career and specialisation have been chosen		
Dependencies:	Test Case 3, Test Case 4, Test Case 8		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks "Talent Trees" button	Pass	
2	User clicks the Specialisation they want to see the Talent Tree for	Fail	Not implemented
3	User clicks the talent they want to add to their character	Fail	Not implemented
4	User clicks "Save" button	Fail	Not implemented

Post-Conditions:	The chosen talent has been saved in database and user is now located on the careers and specialisations page

Test Case ID:	10	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		

Test Title:	Choose character force powers and upgrades		
Description:	User chooses their character's force powers and upgrades		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4, Test Case 8		
Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks "Force Powers" button	Pass	
2	User clicks the Force Power they want to upgrade	Fail	Not implemented
3	User clicks the upgrade they want to add to their character	Fail	Not implemented
4	User clicks "Save" button	Fail	Not implemented
Post-Conditions:	The chosen force power upgrade has been saved in database and user is now located on the careers and specialisations page		

Test Case ID:	11	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character characteristics and skills		
Description:	User chooses their character's characteristics and skills		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		
Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User uses "+" and "-" buttons to increase the characteristics they want	Pass	
2	User uses "+" and "-" buttons to increase the skills they want	Pass	
3	User clicks "Save" button	Pass	
Post-Conditions:	Character characteristics and skills have been saved in database and user is now located on the motivation page		

Test Case ID:	12	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character motivation		
Description:	User chooses their character's motivation		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		
Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks "New" button	Pass	
2	User clicks the type of motivation they want	Pass	
3	User the selects the motivation that they want	Pass	

4	User clicks “Save” button	Pass	
Post-Conditions:	Character motivation has been saved in database and user is now located on the equipment page		

Test Case ID:	13	**Test Designed date:**	23/04/16
Test Priority:	High	**Test Executed date:**	26/04/16
Module Name:	Creating a character		
Test Title:	Choose character equipment		
Description:	User chooses their character’s equipment		
Pre-Conditions:	User is signed in and character has been created		
Dependencies:	Test Case 3, Test Case 4		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks “New” button under weapons	Pass	
2	User clicks “Add” button for the weapon that they want	Pass	
3	User clicks “New” button under armor	Pass	
4	User clicks “Add” button for the armor that they want	Pass	
5	User clicks “New” button under gear	Pass	
6	User clicks “Add” button for the gear that they want	Pass	
Post-Conditions:	Character equipment has been saved in database		

All Character Creation functionality is implemented with the exception of choosing Talents and Force Powers.

5.1.4 Viewing and maintaining characters

This section contains tests relating to the Character Maintenance content and its functionality. With the way I have written the application, the test cases in this section would be very similar to Test Cases 5 to 13 except with the addition of the character’s data being present on the page when it loads up. Therefore, I have chosen not to show these Test Cases in this report.

Test Case ID:	14	**Test Designed date:**	24/04/16
Test Priority:	Medium	**Test Executed date:**	01/05/16
Module Name:	Viewing and maintaining characters		
Test Title:	View character details		
Description:	User views a character that they’ve already created		
Pre-Conditions:	User is signed in, a character has been created and user is located on their home page		
Dependencies:	Test Case 3, Test Case 4		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks on the name of the character they want to view	Pass	
2	User navigates through their character details via the navbar to see existing character data	Fail	Characteristics/Skills and Gear pages display

			correct data but no other pages do
Post-Conditions:	None		

Test Case ID:	15	**Test Designed date:**	24/04/16
Test Priority:	Medium	**Test Executed date:**	01/05/16
Module Name:	Viewing and maintaining characters		
Test Title:	Maintain character data		
Description:	User maintains a character that they've already created by updating the data		
Pre-Conditions:	User is signed in, a character has been created and user is located on their home page		
Dependencies:	Test Case 3, Test Case 4		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User clicks on the name of the character they want to view	Pass	
2	User navigates to the page they want to update via the navbar (skills in this case)	Pass	
3	User uses "+" and "-" buttons to increase the characteristics they want	Pass	
4	User uses "+" and "-" buttons to increase the skills they want	Pass	
5	User clicks "Save" button	Pass	

Post-Conditions:	Character characteristics and skills have been saved in database and user is now located on the motivation page

5.1.5 Using a character in a game

This section contains tests relating to the Character Use content and its functionality. These relate to functionality that would be used while playing in a game while the previous test cases are for the functionality that is used outside of a game.

Test Case ID:	16	**Test Designed date:**	24/04/16
Test Priority:	Low	**Test Executed date:**	01/05/16
Module Name:	Using a character in a game		
Test Title:	Rolling a skill check		
Description:	User completes a skill check from their character skills		
Pre-Conditions:	User is signed in, a character has been created and user is located on their home page		
Dependencies:	Test Case 3, Test Case 4, Test Case 11		

Step	**Test Steps**	**Pass/Fail**	**Notes**
1	User selects the character they wish to use	Pass	
2	User navigates to skills page by using the navbar	Pass	

3	User finds the skill they want to use and click the “Roll” button adjacent to the skill	Fail	Button exists, but no dice rolling system is implemented
Post-Conditions:	System display the output of the dice roll		

In its current state very little Character Use functionality is implemented but there is a placeholder on the page where the function will go (when it is implemented).

5.2 Evaluation

From these tests I have found that my application doesn’t reach the full potential that I aimed for when I started. In its current state the application can create characters and save them to the database. This excludes the Talent Trees and Force Powers as the functionality to display them on the application is not present.

When it comes to maintaining the characters, the user can change/update their character’s skills, characteristics, and gear but nothing else. I chose to prioritise these pages due to the feedback I received from my questionnaires (section 6.4). The application can be used in games for these pages, but physical dice are required and knowledge of how to build a dice pool is needed to make skill checks. In its current state, the application is an aid for character creation. During a game, I feel that most players would probably use it for the database aspects. The most useful page would be the Equipment page as it stores all the stats and equipment that one player cannot remember by themselves.

I feel that I have not made an improvement over existing solutions. I may have made an application that is better in some aspects (and has the potential to be better in others), but the existing solutions would still be a better choice overall with my application’s current state. This is mostly due to the others having better support and being full versions of my application. If a player was to have a choice over my application and an application like Oggdude’s, I believe the player would choose OggDude’s as it is better in nearly every way. The only advantage my application has is that it is stored on a web server so the player always has access to it and can even open the application on a mobile device. This means that the application would be good in local games. However, in games online, the user has access to OggDude’s application and Roll20 so they would probably use them mainly.

I believe that in time, I will improve the application and create a full version which is better than other solutions that exist. The one thing I need to do this however is time. This would allow me to learn more about the capability of the tools I’m using and therefore allow me to make a much better application.

6 Future Work

The top priority of the project is to complete the goals I set in section 8.1. This includes the Talents and Force Powers to complete the Character Creation part. I believe this would be quite an easy task as the front end code and database structure is complete. This means that the only code needed is the middleware to display the data from the database. For the character maintenance I plan to add an "if" statement at the start of each page for the character. This would check the database for any existing data for the character regarding the current page. If there is existing data for the character, then the data would be displayed, else the page would load with the choices as it does right now. This would be quite simple but time consuming as I would need to do it for every page but the Skills and Characteristics and the Equipment. For the third part, using and character, I intend to create a dice rolling function that would be used in game to make skill checks. This could be a difficult task as the dice pool for the skills is derived from different elements of the character, mainly the characteristic for the skill and the ranks in the skill. I would also need to add functionality for displaying the character's wounds and strain as these will be need when playing in a game. The last functionality required to complete the application is spending Exp. This would be tracked as the user buys skills and talents. This would mean that a user would be restricted from buying these things if they do not have the required Exp. This would also be needed in a similar way for the Equipment section but with credits instead.

At the moment the Equipment section does not support attachments and modifications for weapons and armor, this is something I intend to add. In order to further improve the current state of the Equipment section, I plan to add to a "Create your own Equipment" button and functionality. This would allow users to create Equipment that would be saved to the database. This would then be approved by an admin who would then make it available for all users.

In regards to the sign in page, the current functionality only supports the user logging in by choosing their name and clicking "Set Current User". This is a great security risk as it means that anyone can sign in as any user. I currently have the database set up to store a username and a hashed password for all users, but the password hasn't been included in the current application. As well as this, there is no way to register a new user. These issues are things I plan to fix in the future.

Another issue that needs to be fixed with users is that everyone has the same privileges. This means that no one is an admin other than the person with direct access to the database. This is something I must change.

In order to help new users as they create a character, I created help buttons on every page to provide information about the current page and what the user must do. These currently only display an empty modal with no information. I plan to add a table into the database which stores these help texts, which will be queried by the system and displayed on the corresponding pages.

With the way I having written the code, a user must click a button to save their choices to the database. I believe this was a bad choice since the page reloads on every save. This can be annoying to the user on pages such as the Skills and Characteristics where they have to make multiple choices on that single page. In the future I plan to improve this by having a single button that saves the whole page rather than multiple buttons.

7 Conclusions

In conclusion, I believe I have met most of my goals, but I have failed to meet others. I have successfully created a Character Creation application that can be accessed by multiple users at one time, and all characters created are stored into the database for later use. The application fulfils its goal of having a clear interface for the user and providing a clear path through the creation process. Currently the application is in a state where it can aid a character during a game, but they will also need other physical tools such as dice to play the game fully.

The foundation of the application is in place and the functionality that is missing is more time-consuming than difficult. The database is structured for future additions to the application and is populated with data, the only thing that's missing is the middleware to retrieve this data and send it to the front end UI.

I believe that if I had planned out the project in a more efficient way then I could have achieved more. If I had more time I could have created an application that was all the player needed in a game and they would only need the rulebook if they needed to look up a rule or game mechanic.

8 Reflection on Learning

8.1 Lessons learnt

From this project I have learnt the importance of researching tools and framework that I would be using. This is mostly from the Access database that I used. Even though it was useful in some aspects, such as creating relationship diagrams and creating queries, the software creates too many problems for a larger application. The software is good for my application's current size, but if I were to take it further and create a full website then I would need a proper database server that fully supports SQL commands and can be used with a distributed application over multiple servers rather than being dependent on a single file.

Microsoft Access was good for development. However, for a live application, using a database application such as MySQL would give better performance. The structure of my application makes migration to another database relatively easy as my structure puts all the database queries that might need changing are in one folder.

I also found that, when creating UI design mockups, I should have created low fidelity designs first. Due to the time constraints, I didn't manage to finish all my mockups and was forced to move on to other work due to time passing. Lower fidelity designs would have removed this problem and I could have created high fidelity mockups for selected important pages if I had time.

I also learnt about the importance of judging time and the workload of the project. I believe that my goals were over ambitious which led to me developing a new plan during the implementation stage which gave be more realistic deadlines and stages of completion to aim for. This allowed me to complete the application to different extents and meant that I could easily see what worked according to these deadlines.

8.2 Constraints during development

From the start of this project, there have been multiple constraints which have limited the final outcome. The main constraint was the time to complete the project. This was only a twelve-week period from initially planning the project to creating a final solution for the problem.

The next constraint, somewhat related to time, was that I was the only person developing this application. This meant that I was the only one to provide input (other than verbal advice from my supervisor) into the project which limited the quantity of work that would be produced. By extension, the quality was also limited as the project was restricted to my knowledge and things I could learn during the development of the solution. This slowed the earlier stages as I had to design the whole application myself and predict the workload for the upcoming work. The later progress in the creation of the application was reduced due to the fact that I had to divert time to learning about new areas of coding and how to apply them to my work.

Another constraint was the Access database. This had drawbacks, with the main one being that it doesn't share the full capability of an SQL database. This was due to the Access driver not supporting certain SQL statements, for example: JOIN, DESCRIBE and REPLACE, and meant that I had to develop workarounds for these statements when I wanted to use them. This took time away from other functionality that I had to create.

9 References

[1] Publishing, F.F. (2016) *Fantasy flight games*. Available at: https://www.fantasyflightgames.com/en/index/ (Accessed: 29 April 2016).

[2] OggDude (2013) Emperor Norton. Available at: https://community.fantasyflightgames.com/topic/89135-another-character-generator/ (Accessed: 29 April 2016).

[3] 20, R. (2016) Roll20/roll20-character-sheets. Available at: https://github.com/Roll20/roll20-character-sheets/tree/master/Star%20Wars%20FFG%20API-Compatible (Accessed: 29 April 2016).

[4] FFG Star Wars index (no date) Available at: http://swrpg.viluppo.net/ (Accessed: 29 April 2016).

[5] Publishing, F.F. (2012) At the core. Available at: https://www.fantasyflightgames.com/en/news/2012/11/19/at-the-core/ (Accessed: 29 April 2016).

[6] Otto, M., Thornton, J. and contributors, B. (no date) Designed for everyone, everywhere. Available at: http://getbootstrap.com/ (Accessed: 29 April 2016).

[7] Inc, E.S. (2009) ForeUI. Available at: http://www.foreui.com/ (Accessed: 29 April 2016).

[8] XAMPP Installers and Downloads for Apache friends (no date) Available at: https://www.apachefriends.org/index.html (Accessed: 29 April 2016).

[9] Microsoft (2016) Database software & applications. Available at: https://products.office.com/en-gb/access (Accessed: 29 April 2016).

[10] Games, F.F. (2013) Star Wars: Edge of the empire core Rulebook. United States: Fantasy Flight Publishing, U.S.

[11] Games, F.F. (2014) Star Wars: Age of rebellion RPG core Rulebook. United States: Fantasy Flight Games.

[12] Games, F.F. (2015) Star Wars: Force and destiny RPG core Rulebook (2015) United States: Fantasy Flight Games.