DUBLIN INSTITUTE of TECHNOLOGY
Institiúid Teicneolaíochta Bhaile Átha Cliath

State Street Online RPG/Social Network Game
Ordinary Degree Project Manual

DT249
BSc in Information Systems and Information Technology

Carl MacDiarmada
Dr Aneel Rahim

School of Computing
Dublin Institute of Technology

April 2016

Abstract

The following document is concerned with the development of an Online Multiplayer Roleplaying Game allowing users to create a new persona and build the best empire for themselves. The paper contains details of research that was carried out on the subject, as well as details of the different stages of development ie design, implementation and testing.

Acknowledgements

I would like to thank my supervisor Aneel Rahim and the project co-ordinator Ciaran Cawley for the support and feedback given on this project; its design and its features. They were very understanding with my hectic lifestyle but we made it work.

I would like to thank my friend Jeff Johnson who consulted me about different front end development options and created the beautiful logo of my game.

I would also like to thank Joe Chedburn of Torn City who is the creator and brains behind Torn City who sparked my imagination.

Contents

Table of Contents

Table of figures

1. Introduction

This chapter describes the aim and objectives of the project as well as the various challenges and obstacles encountered by myself. It also contains the outline of this paper’s structure.

1.1. Project Aim and Objectives

The primary aim of my project is to create an online Role-Playing Game (RPG) to offer a social network aspect inside a gaming environment. The user essentially creates a new persona and does what they like in State Street. Playing this proposed game, the player would enter a world populated with different types of money making schemes, missions, crimes, fights and would navigate through a virtual world called State Street. The more friends you make, the more likely you are to succeed. You are able to choose your own path in State Street. Players can choose to be an upstanding citizen by working their way to the top through various legitimate jobs or playing the stock market. Or you can choose the life of crime, as State Street offers various crimes and gang life activities. With such dynamic gameplay features, State Street offers a fun virtual world for everybody’s interest.

Many people are intrigued, if not obsessed, with the thought that they could be a criminal; not working, attacking whoever they want and not having any real life repercussions. As players progress in State Street, in-game actions will start to have consequences. Players will find there are repercussions in the game when they pick on someone bigger and stronger than themselves. It will end up with their player going to hospital. This game would appeal to most people who are long term planners. There is no making a “quick buck” in State Street. Players will need to plan their lifestyle around their goals in the game.

The target audience for this game is anyone above the age of 16, as it deals with some sensitive subjects such as drugs, crime, attacking etc. Anyone who wants to escape the real world for a while and forget about life’s pitfalls can get away with a few keystrokes and build lasting relationships within this game.

There are several objectives to this project, mostly within a personal development capacity as an IT professional:

- To research differences between various DB setups
- To become familiar with objective PHP, CSS and JS
- To develop project and time management skills.
- To do research on other online RPGs and their ‘hook’

1.2. Challenges and Obstacles

Due to my limited knowledge, from college projects, on objective PHP and creating databases, it was quite difficult planning out the right database structure to use, and which functions to use in PHP/MySQL for securing the game so that no one could inject SQL into my queries and cheat on the game.

I also work full time and go to college 3 nights a week, so finding the time to do all of the work was challenging, especially as this affected my ability to meet with my project supervisor.

1.3. Outline of this Project Manual

The next section, **Section 2,** of this document is concerned with the project plan and shows the timing and duration of the various stages of the project.

This is followed by **Section 3** that describes the research carried out on other online RPG’s, PHP, DB structures and security.

The purpose of **Section 4** is to discuss the design and implementation of different software system components that were developed as part of this project.

This is followed by **Section 5**, which is concerned with the testing process of the above components.

The final section, **Section 6**, is just a conclusion to this document which contains some additional information referenced from other parts of the document.

2. Project Timeline

The timelines for the project have been very tight so there was not much room for changing the initial design and requirements or introducing any major enhancements. It was also difficult to finish off the design and look of the website which will be further developed during the summer for its release to the public.

Figure 1: This illustrates how the project was divided into various different stages and when each of those stages were completed. Certain tasks overlapped because the project consists of multiple software components and unit testing which were carried out together. I also had to continue with these tasks with the development process to detect any software bugs as early as possible. It’s also worthy to note that the research and prototype was an ongoing stage as extensive and ongoing research was needed to develop various parts of the storyline.

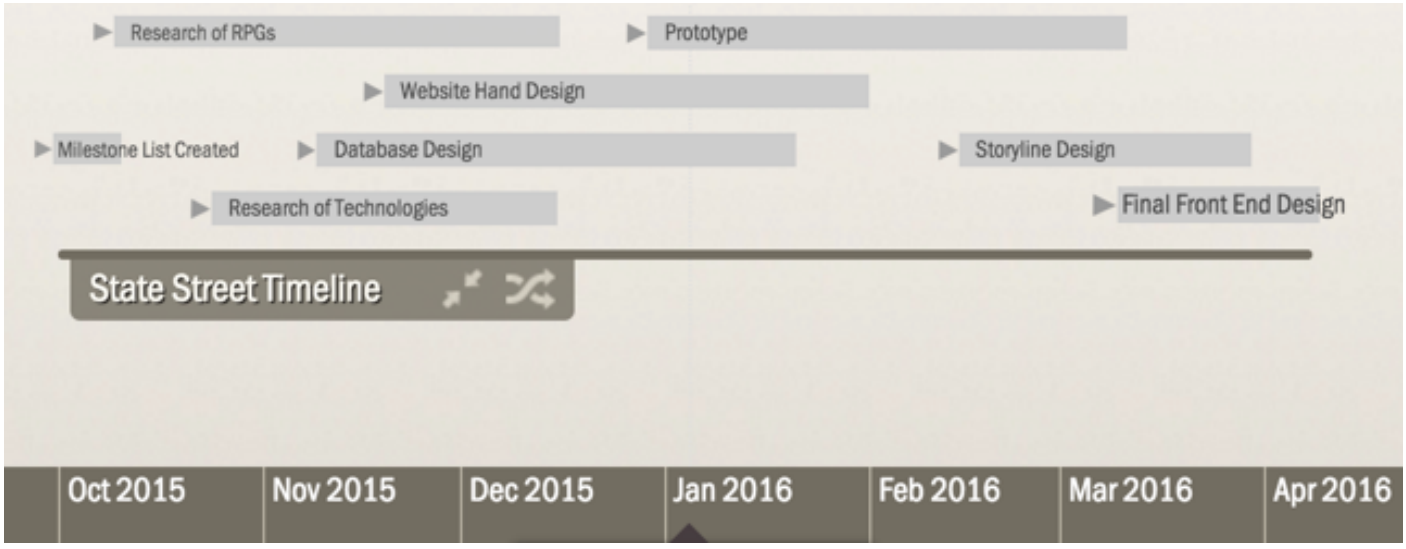


Figure 1 Project Plan chart

3. Research

This chapter outlines the research carried out as part of the project and how it has affected the decisions around target environment and implementation approach.

3.1. Massive Multiplayer Online Roleplaying Games (MMORPG)

Anyone who plays online games knows there are many Massive Multiplayer Online Roleplaying games. The most popular MMORPG is ‘Torn’®, previously known as Torn City®, which is the game that gave the inspiration to create another similar one. Many of the functions of this game are based on it and were added them into State Street with a few upgrades and changes. On speaking with many of the staff members and admins of Torn about their design structure, a great insight was given into how the game is such a success and how it retains its users.

The figure below shows the user stats of Torn® (TornCity 2006)

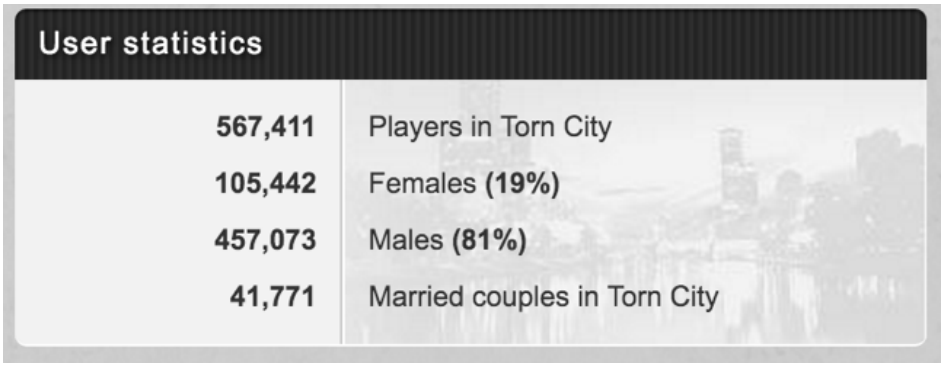


Figure 2 User Statistics of Torn®

As you can see from Figure 2, which is available in the game, there is a substantially bigger percentage of male players. There are 567,411 players signed up to the game, but a majority of those players wouldn’t play regularly. Figure 7 below shows how many players are active in this game.

The biggest questions that came to mind after seeing these statistics were:

Why is it so heavily used by males?

On researching the game further, it was apparent that the game was aimed towards males. A few women that play the game were asked the following questions.

What attracts you to this game?

Are there parts of this game that you feel oppress women?

Have you ever felt like the game demeans women?

Do you have any ideas on how to make the game more appealing to women?

3.1.1 Torn Survey
Role of Women

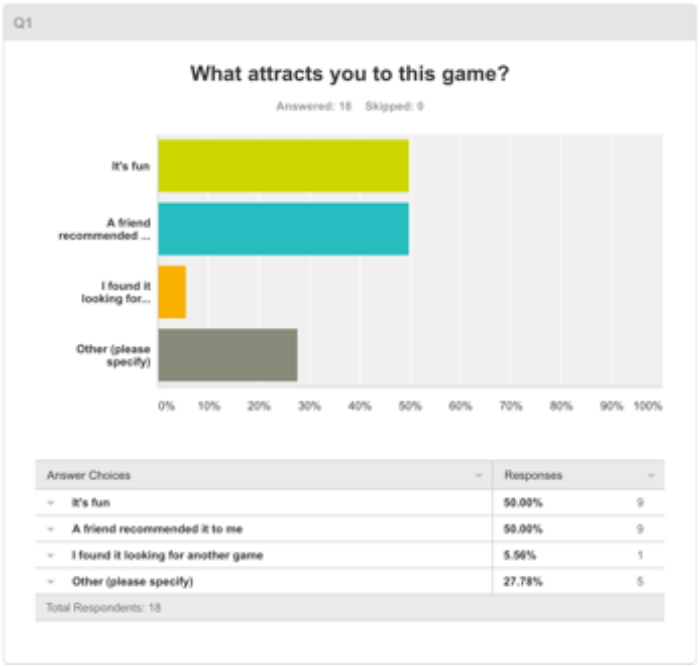


Figure 3 Question 1 of Survey

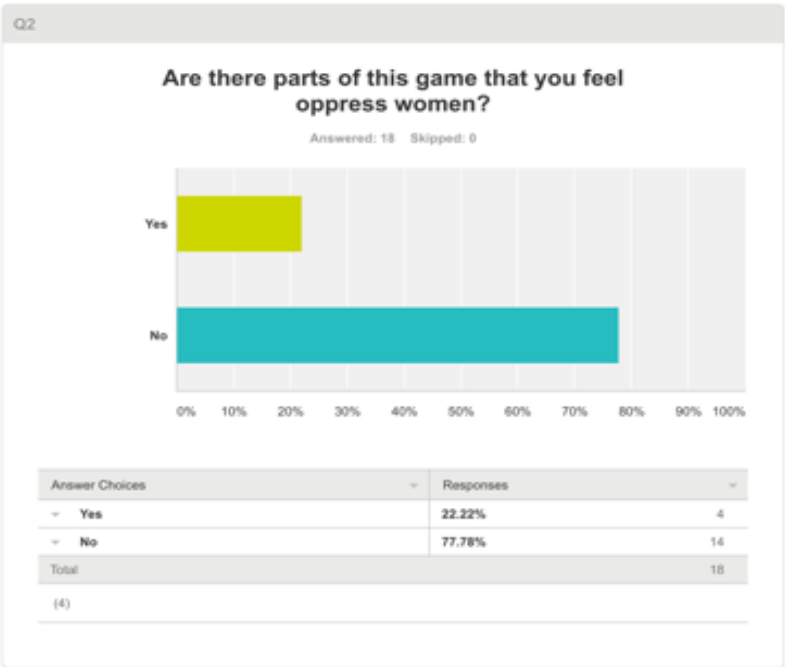


Figure 4 Question 2 of Survey

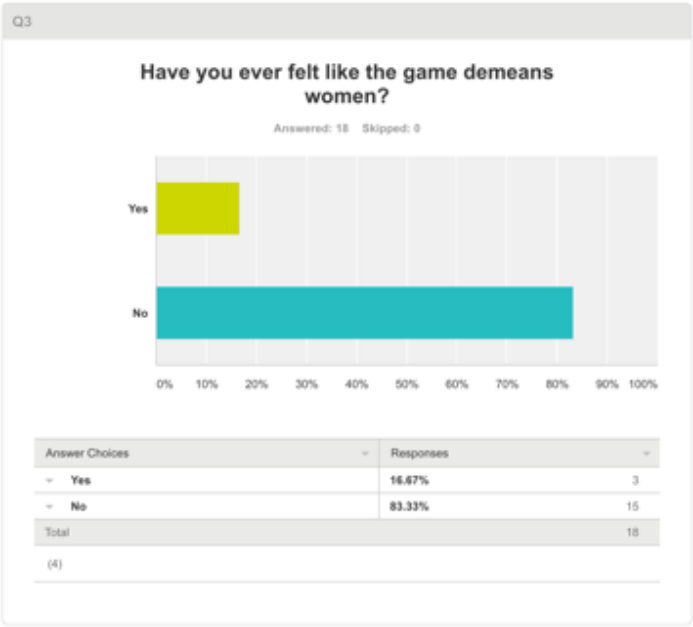


Figure 5 Question 3 of Survey

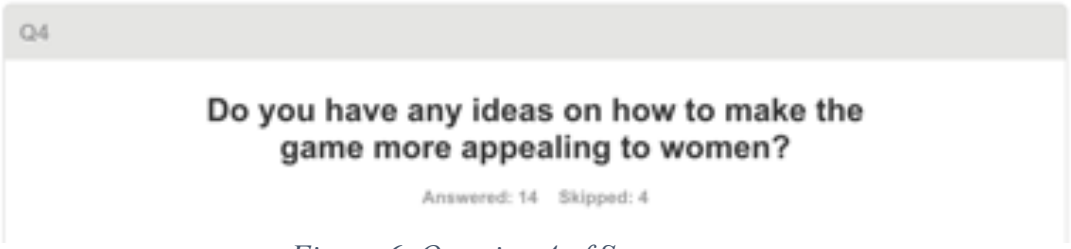


Figure 6 Question 4 of Survey

Anon Answer: “It's not really a problem with the game itself. There's a problem with some players’ attitudes to women, but there are people like that in the real world too. I think purposely including anything to make it "more appealing" for women will just make the player community laugh it off, as well as just being quite patronizing.”

Anon Answer: “Its already very appealing to women”

Anon Answer: “Most women have different values from men. There are some like me, who grew up as a Tom-Boy and didn't hang out with other females often. I am assuming many of the women of torn fall into this category. I don't believe this sort of game will ever be appealing to most women. There is nothing wrong with that. The idea that something is male or female can be destructive, but to do the opposite and say that there are no "male" or "female" type things is just as destructive. We are, and should be, different. There are a few men who dig getting pedicures and manicures. For the most part, the nail salon is a woman's domain. It's not that men and woman can't both benefit from keeping their cuticles moisturized, it's just more women value it. So no, I have no ideas on how to make the game more appealing to women.”

3.1.2 Torn User Activity

*Figure 7 User Acitiy Stats of Torn ® **(TornCity 2006)***

These statistics change on a daily basis as it only shows up to the last 24 hours. I find it quite impressive that after 11 years being online, it can still retain a 16,000 active player base. They also do not advertise on their site to gain revenue.

3.2. MC Codes – RPG Gamescript Engine (MCCodes 2005)

The research into developing a game engine for this text-based RPG, lead to MC Codes. MC Codes was developed over 10 years ago and actually sold licences for people to make their very own text based game, which cost in the region of $200. Torn® was originally developed on MC Codes, but they have since upgraded all of their code and systems and is now a standalone script. I downloaded the free version of MC Codes and this was the starting point for my game. The whole concept of MC Codes was to give people the opportunity to just change the look and names of certain parts on the GameScript. It wasn’t supposed to be edited in source code or have features added. It was more so intended for the average person who had an idea for a game but had little to no technical background.

For what it was and did, it was quite advanced for its age, and it was impressive to look at. The closest thing to compare it to is Zork, but with many live players, and you will have to shape your own story. It was definitely the best source of research, and point of contact to go to when issues arose in my code, which were plentiful.

In a technology obsessed world, you can imagine that the code, over 10 years old, was quite outdated. Even though this was the case, it gave an extensive insight in terms of a starting point and figuring out how to store all the data in a database, what tables to use and how to link them, as their systems have been tried and tested.

They had updated some of their files in 2012. This was because their site got taken down and their files subsequently removed. They had to re-publish them on their site, which is why they are dated 2012 on their website.

3.3. Software and Hardware Requirements

3.3.1. Development Environment

The software system developed as part of this project consists of several components, which are described in much more detail in sections 4.4 - 4.6.

To develop and test this online RPG game, you need to have a program to write your code(*Sublime Text 2, Notepad*++), a server, a database, HTML, CSS, JavaScript, PHP and SQL syntax. You can easily use the program *XAMPP* which will locally host your website. It will read all PHP code and you can setup your own database. It is a great tool that can easily change code and the user is then able to view the edited code on their site instantly.

To develop graphics *Photoshop*®; a standalone program by Adobe was used. The appearance of photographs were able to be manipulated for the game. It also allows the user to create their own images.

XAMPP and *Sublime Text 2* do not require a lot in terms of processor or memory amount – practically any of the modern computers are able to run those for development purposes.

Subsequently, *Photoshop*® has a higher minimum requirement. It is very heavy on graphic processing and it is recommended to have 8GB of RAM and a 2GB standalone graphics card, where possible.

3.3.2. Production Environment

One of the main things of any online content, whether it be a social network, content sharing, game, or website, is that you need is reliability. Speed is the second most important aspect. Trying to figure out whether an own server or an external server would most benefit the online RPG was quite interesting and gave some good insight into what businesses should weigh up in terms of setting up some online content for their consumers.

3.3.3. UML in Game Development (Sneftel 2003)

After extensive research in software engineering transferring the skills taught in UMLwas an exciting venture, but after researching UML in gaming development, it was apparent that it is not widely used. “It's no secret that development practices in the games industry are usually execrable. The culture encourages and rewards cowboy programmers, and views "crunch time" as inevitable. It's no wonder that structured techniques such as UML enables are ignored.”(Sneftel, gamedev.net moderator) After seeing this, a bit more research was undertaken and it makes sense that UML is difficult to use for game development, not by its own fault, but because UML needs to be adapted as it was developed around 30 years ago.(There are roots of UML in the late 80s). Gaming development has become more and more complex also. UML tends to favor a sequence-based, as well as a transaction-based paradigm, which is a square hole that most games are not easily hammered into.

3.3.4. Database Research

In researching the final database schema that was decided upon to implement, there were many suggestion that were available online, some more interesting and efficient than others; all of which could have implemented the required result. The first stop was to a website called dbforums (DBForums 2011) where someone had already asked about a database design for an online RPG. There were a few answers posted to this question that were critiquing his database structure. This critique was used to ensure the same mistakes weren't used. For example, for the user's safety, the database would have to be secure. The number one thing people would be afraid of getting hacked is their password. From research, it was possible to use an md5 encryption with PHP and encrypt every user's password with this.

The way this was coded meant when any POST or GET function was used to enter, amend or authenticate a users password, the code would read md5('{$_POST['password']}'), and thus encrypted/decrypted the variable.

Figure 8 below demonstrates how the passwords are stored into the database, and gives complete confidence that the database administrator nor any hackers can view that password.

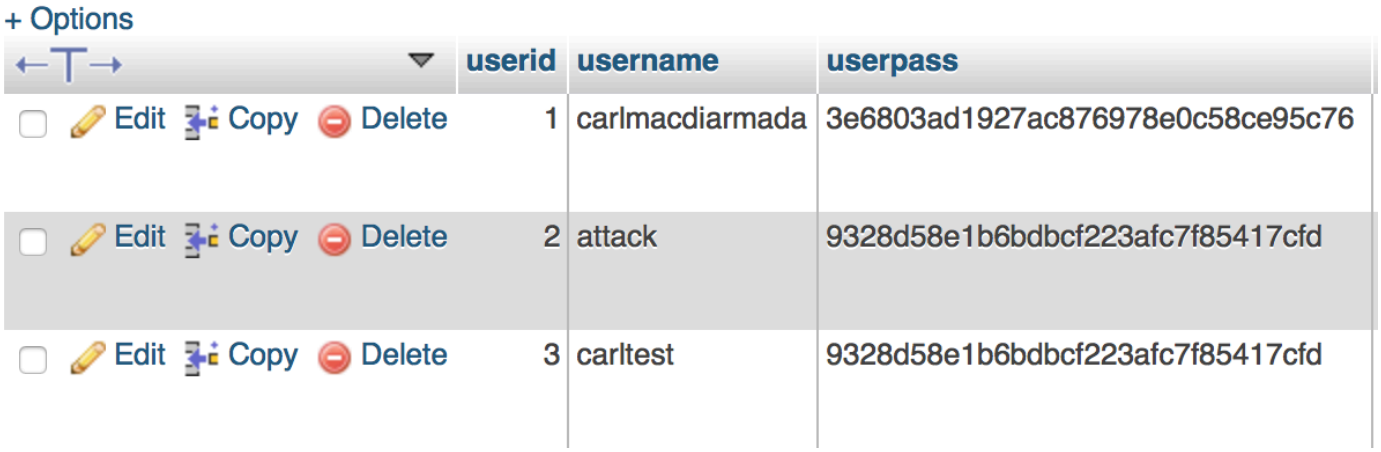


Figure 8 Database Password Security

In the end, the database was based on MCCode’s (MCCodes 2005) database schema as they were the closest to what was trying to be created. It was great to have that as a quick reference when any obstacles were faced.

The schema will be discussed further, along with showing the class diagrams in section 4.

Self-Hosting vs External Hosting (Creative 2014)

Self-Hosting Pros:

- You control your own hardware. The majority of hosting services only give you 3 choices : “*good, better, best”* with different prices. You could get any type of hardware for all three and you would never know.

- Easy access to and you have control over the hardware. You can change or upgrade your hardware whenever you want without anyone questioning why or telling you not to. You can add extra storage as you wish, which would be a lot cheaper than paying extra per year/ per month to a provider. If you find your processor too slow you can change it or add extra RAM without much cost.

- Total control over content and software. If you want to change server software, you can whenever you want. If you want to give 3rd parties root access to the server (for whatever reason that may be) you can.

Self-Hosting Cons

- Dealing with **Internet Service Providers**(ISPs). Many ISPs explicitly block web hosting in their contracts, or they have clauses in their contracts to raise prices for customers who run servers.

- IP Address issues: most people would be provided with a ‘*Dynamic IP address’* which means that the IP address of the server would change every few days, or even every few hours. A ‘*static IP address’* is the most advisable one to have, which as the name suggests, remains the same.

- **Security:** Your home network, generally, isn’t the most secure network in the world. Unless you’re using a physically separate network for hosting, your personal computers would be on the same network as the server. This leaves you open to hacking and your personal information would be at risk.

- **Downtime:** With an external host, you are assured that if something goes wrong, someone will be there to fix it. When you host it yourself, it’s up to you to fix it personally 24/7.

For an online RPG, it would be best to use a 3rd party hosting service as you can upscale or downscale easily. You wouldn’t have to worry too much about maintenance and repair if it fails whilst unattended. The user base would be happier knowing that it is under constant supervision and they can play without hesitation.

For the likes of creating a personal blog to update your relatives on what you have been doing all year, or a personal website that has a form for your DVDs and who you loan them to, may better benefit from a self-hosted server because it is such a small scale investment for a small scale project.

4. Design and Development

This chapter will be concerned with the design and development process of the project in question. It will contain a detailed description of each component of the system, together with the list of functional and non-functional requirements.

4.1. Functional Requirements

This section contains list of requirements for the software system, together with Use Case diagram. Details of how some of those requirements are satisfied are contained in latter sections of the manual.

4.1.1. List of Requirements

Following list gives high level overview of what the user is able to achieve using my online RPG

- Create your own personal character
- Use your *energy* to train your character stats
- Use *guts* on crimes
- Earn money through a job
- Attack other users
- Send messages to other users
- Join or create a *clique*

There are 3 basic ‘users’ but you cannot simply *select* these playing styles. You are the playing style.

The Brawler: The most feared in town. You’ve put that work in getting your stats up slowly and now it’s time to reap the benefits. You can go around attacking players who mocked your strong etiquette and dedication to making your stats better. When you defeat your enemies you have the option to mug them, take some of their hard-earned (or dirty) cash for yourself.

The Criminal: You see this person in jail a little too often. If you don’t see them burning down warehouses for a cut of insurance, you’ll be sure to find them in the black market selling the guns and drugs that “were obtained in an illegal fashion”

The CEO: You’ve come a long way from being a Burger Flipper, but you finally have your own franchise built from the ground up and a solid skyscraper on State Street with your name on it. You know how difficult it can be, scraping the bottom of the barrel, but you appreciate all the scars you’ve acquired getting to the top, now it’s time to sit back, relax and let the money roll in.

4.1.2. Use Case Diagram

**

Figure 8 is a Use Case diagram that illustrates the system capabilities.

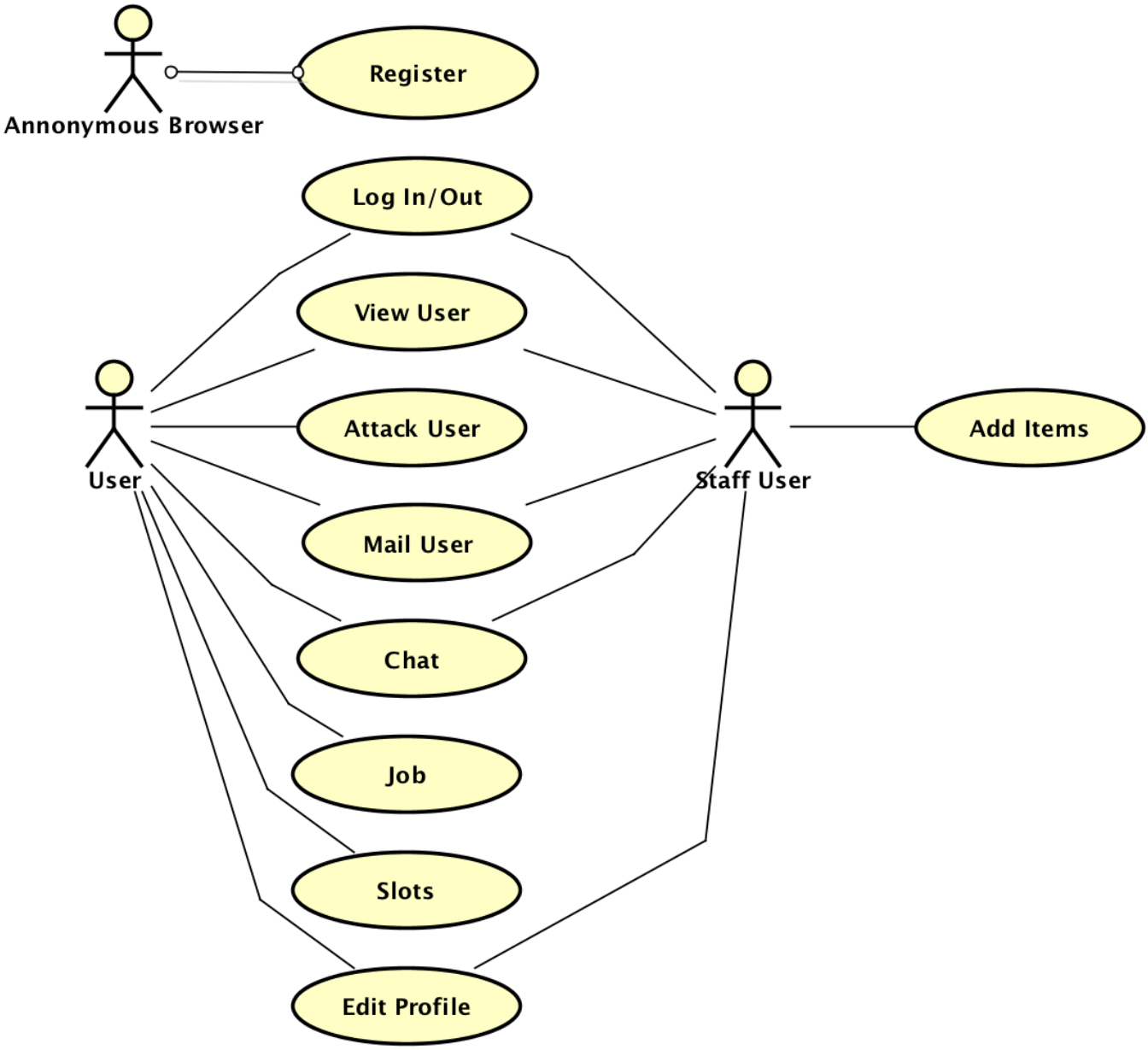


Figure 9 Use Case Diagram

4.2. Non-Functional Requirements

The following list outlines non-functional requirements that need to be satisfied by the Database Creator system:

- Performance – response times should be kept to minimum in all components of the system. Heavy processing should be avoided in the front end to allow all devices and browsers to run it and to keep requirements to a minimum.
- Security – all data has to be securely transmitted between different parts of the system. Any SQL injections or cross site scripting attacks need to be prevented and sessions must be managed in a secure manner.
- Scalability and Extensibility – system should be developed in such a way that would allow to easily add new functionalities and reuse existing code.
- Logging – all errors and/or exceptions should be recorded in the log files on server side for investigation purposes.

4.3. Project Structure Overview

An online RPG consists of several components that are integrated with each other. Figure 5 shows those components and links them all together.

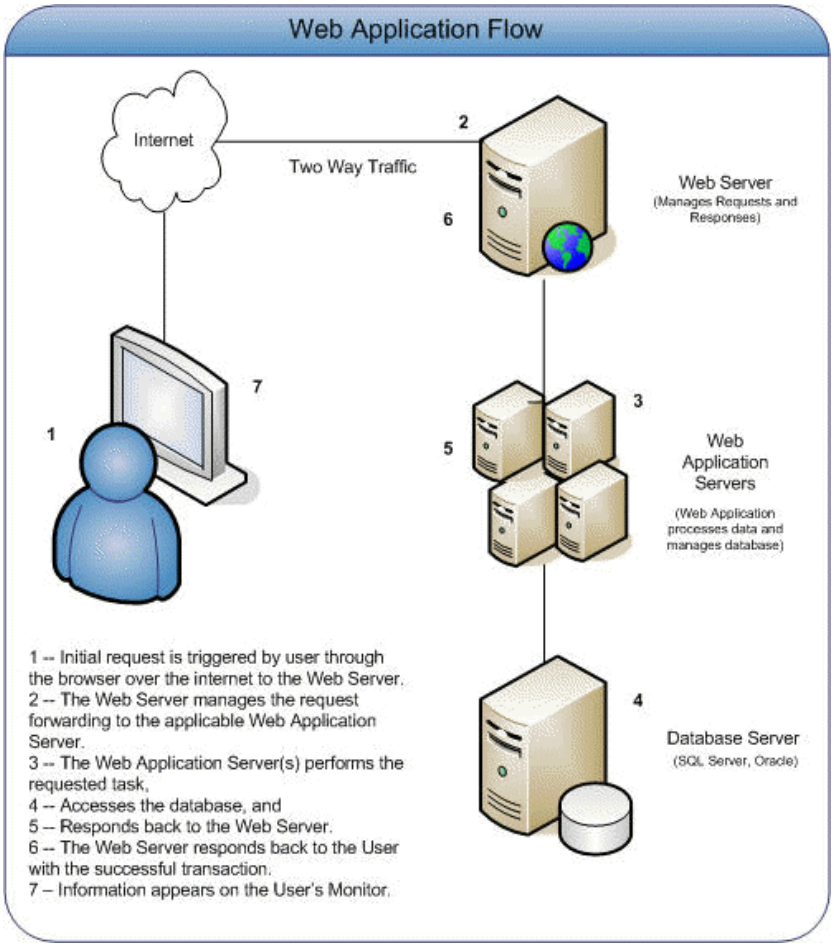


Figure 10 Web Applications

The central part of any online RPG is the game engine. This is what everyone connects to, sees and uses. The particular game engine used , like many other similar RPGs, is inspired by the MCCodes game engine.

From the user’s perspective, the main parts of the RPG would be the web design and user’s device. They wouldn’t be concerned with the database directly, but they do use it in nearly everything they do in the RPG, whether they know it or not.

One could argue that the database aspect of the online RPG is the ‘game engine’ but it is also much more than that. It is used to store all of the user’s data and when queried can display the necessary values.

The following sections describe the design for each of the components in much more detail.

4.4. Database

4.4.1. Environment Setup

A MySQL database environment is very easy to setup. The XAMPP default environment setup was used ,which has both a code and visual setup, so you could either type in code eg

INSERT INTO ‘table’ (‘attr 1’ , ‘attr 2’, attr 3’) VALUES (1, 2, 3);

Or could use the visual setup to create a table and add in attributes and values without coding.

4.4.2. Class Structures

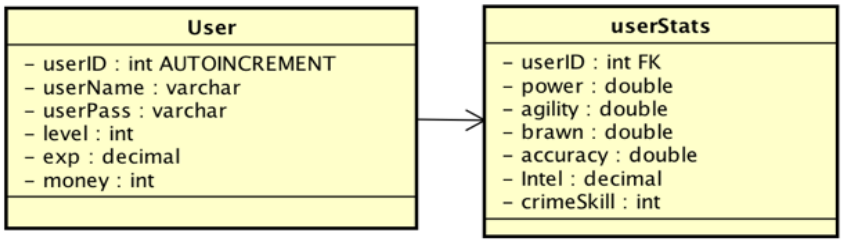


Figure 11 Basic Class Diagram

This is a very low level class diagram based on the actual classes *‘users’* and ‘*userstats’*. The actual class diagram, originally created as the *‘users’* table is quite large and could not be used. This shows the relationship between *users* and *userstats* using the ‘*userID’* attribute so the stat attributes were not needed to be inputted into in users table. This is called a dependency. Dependencies are when one of the classes depend on the other at some point. In my games case, the *users* table depends on the *userstats* for quite a few parts eg; attacking, displaying the stats on the homepage, job.

4.4.3 Database Joins

Database joins are a very important part of a database, especially in an online RPG. You can store information in two separate tables and join both of them to display something of importance to the user.

Figure 12 shows an example of the code that used for the job section of the game.

```
{
$q=$db->query("SELECT j.*,jr.* FROM jobs j LEFT JOIN jobranks jr ON j.jFIRST = jr.jrID WHERE j.jID={$_GET['interview']}");
$r=$db->fetch_row($q);
if($ir['strength'] >= $r['jrSTRN'] && $ir['labour'] >= $r['jrLABOURN'] && $ir['IQ'] >= $r['jrIQN'])
{
$db->query("UPDATE users SET job={$_GET['interview']},jobrank={$r['jrID']} WHERE userid=$userid;");
print "<font color=green><b>

<div id='mainOutput' style='text-align: center; color: green;  width: 600px; border: 1px solid #222222; height: 70px;
                margin: 0 auto 10px; clear: both; position: relative; left: -20px; padding: 8px'>

Congratulations you got the job!</div></div><br/>
<a href='job.php'><u>Go To Your Job</u></a>";
}
```

Figure 12 Join Code from job.php

	jID	jNAME	jFIRST	jDESC	jOWNER
Edit Copy Delete	4	Government	16	Got a great idea and think you can get people to l...	Public voters
Edit Copy Delete	3	Police Department	11	Do you dream of changing the world? Start here and...	Deputy Dickerson
Edit Copy Delete	2	Burger Queen	6	A whopper place to eat, and get worked off your fe...	Mr Thompson
Edit Copy Delete	1	Hospital	1	While the demands and stresses of hospital work ar...	Dr. Johnson

Figure 13 Jobs Table

	jrID	jrNAME	jrJOB	jrPAY
Edit Copy Delete	1	Receptionist	1	200
Edit Copy Delete	2	Student Intern	1	500
Edit Copy Delete	3	Nurse	1	1000
Edit Copy Delete	4	Specialist	1	2000
Edit Copy Delete	5	Surgeon	1	3500
Edit Copy Delete	6	Stock Taker	2	50
Edit Copy Delete	7	Burger Flipper	2	200
Edit Copy Delete	8	Cashier	2	500
Edit Copy Delete	9	Shift Manager	2	1000
Edit Copy Delete	10	General Manager	2	2000
Edit Copy Delete	11	Traffic Officer	3	150
Edit Copy Delete	12	Sargeant	3	400
Edit Copy Delete	13	Lieutenant	3	850
Edit Copy Delete	14	Deputy	3	1700
Edit Copy Delete	15	Sheriff	3	3000
Edit Copy Delete	16	Member of Parliament	4	250
Edit Copy Delete	17	Mayor	4	700
Edit Copy Delete	18	Prime Minister	4	1750
Edit Copy Delete	19	Vice-President	4	4000
Edit Copy Delete	20	President	4	10000

Figure 14 Job Ranks Table

The first part ***(figure 12)*** with the variable $q is the join part. It brings together two parts from the *jobs* and *jobranks* tables. In the jobs table ***(figure 13)*** there is an attribute jFirst which is joined with the jobrank table ***(figure 14)*** attribute jrID. So the **Hospitals** first rank would be *jrid 1:* Receptionist, the **Burger Queen** first rank would be *jrid 6:* stock taker, and so on.

4.4.4 Database Access Sequence Diagram

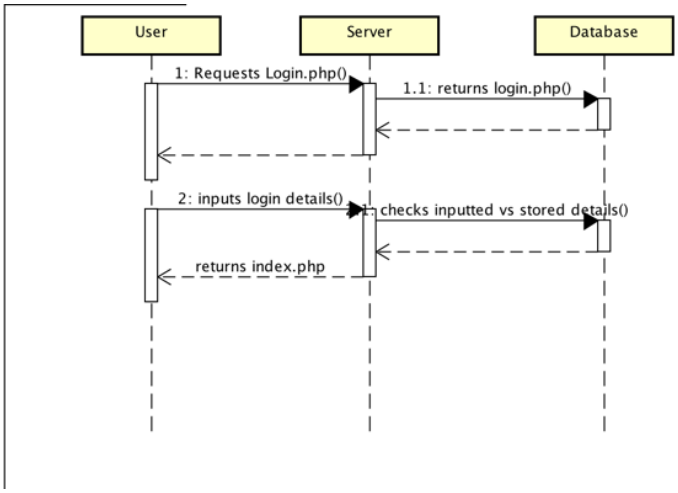


Figure 15 Sequence Diagram for User Access Database

Whenever a user goes to the homepage of the game, index.php it will automatically request the login.php page when there is no active session. The server will then communicate with the database and return to the login.php page.

The user will then input their login details and submit them to the server. It will check the details against what is stored in the database.

4.4.5 Why MySQL?

MySQL was initially released just over 20 years ago, in 1995, by the Oracle Corporation. It has been one of the biggest and most widely used open sourced *relational database management system* ***(RDBMS)***. There are quite a few different RDBMS that could have been used, but as MySQL was already used in the studies, it seemed like the best one to use. A lot of research into MySQL was done and there was an update to MySQLi, which is known as MySQL improved. There were quite a lot of updates to MySQL with this update. Most notably:

- An object orientated interface
- Support for prepared statements
- Support for multiple statements
- Support for transactions
- Enhanced debugging capabilities
- Embedded server support

Even though these updates weren’t initially needed for the game, it did make sense to use MySQLi as a lot of MySQL functions became depreciated in 2012 and a lot of security features were introduced in MySQLi to support my own security measures, like anti SQL injection.

4.5 HTML and PHP

4.5.1. Background of HTML and PHP

HTML 2.0 was brought out in 1995, and by 1997 we were already up to HTML 4.0, and that was the sweet spot. It took 7 years before the internet, and the technology using the internet, required more. Thus HTML 5.0 was released, with up to date features that you used to need JavaScript to do. HTML in this game is used to display any directions to users, prompts and various layout options (in conjunction with CSS). PHP is then used to communicate with the backend of the game, the GameEngine/Database.

Any website, including the game can be broken into 3 layers: Presentation, Controller, Transport. These lahyers are discussed further below.

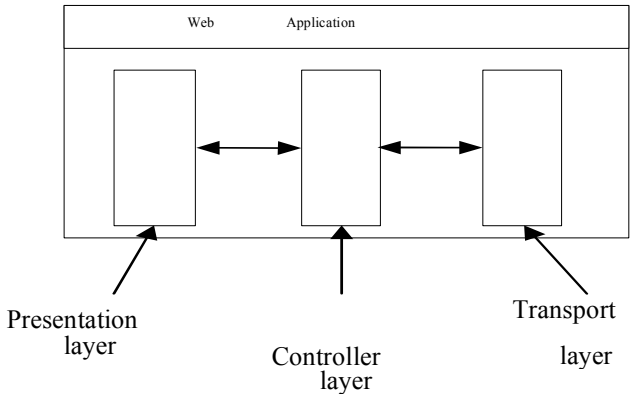


Figure 16 Website Structure Overview

4.5.2 Presentation Layer

The presentation layer is responsible for rendering web pages in the user’s internet browsers. It consists of both server side and client side technologies to provide a fully interactive interface to the user.

Server side of this layer includes CSS pages, CSS is short for Cascading Style Sheets. CSS is an industry standard presentation layer that is extremely powerful when used correctly. With CSS you can tag certain parts of your HTML pages and reference them in the CSS page to format them. Therefore you can have account details as a class, and you can have them white with a black background. If you ever want to change the layout of the account details, you don’t have to change your code in the HTML pages, simply; with CSS you just change the .css file and it changes the layout of your whole site. The main pages you can see this on is my login page and Index page, although it is at work throughout the whole website.

Login page is really just a form to allow users to enter their credentials and send to the server to be authenticated. If they match what is stored on the MySQL database that was mentioned in previous section, users are brought to the index page where they will instantly see their data, not someone else’s.

Index page allows user to access the information from the database and it is displayed to them, from the index page they can then choose what to do in State Street. Without CSS this information may not format correctly, or may not display properly so it is great to have that for the user.

4.5.3 Controller Layer

Controller layer is concerned with handling and dispatching HTTP requests. Based on the URL, it directs users to appropriate pages in presentation layer and updates relevant models.

The controller in this case is the web server as hyperlink references would be embedded in the HTML code, and that would send the request to the server to retrieve that file, and display it with the presentation layer. This would use the customer’s bandwidth as they would have to download the requested page, and any images belonging to it for the presentation layer.

4.5.4 Transport Layer

The transport layer is responsible for the communication between the web server and the database, in which PHP and MySQL statements are used, that aren’t visible to the users. This gives the user no control over what they see, and limits the likelihood of SQL Injection, and database hacking.

The user cannot see this process, and can’t see the functions that communicate with the database. It’s also important not to put your database configuration details in your pages, and to use a different file to connect to the database and to store your configuration details, for security purposes.

4.6 Component Breakdown

In this section, the main components of the website and how they work will be outlined. There were a few fundamental components to the website, that, without them, this game wouldn’t work.

4.6.1 Cron Jobs

Crons jobs are time-based **jobs** scheduled in Unix-like computer operating systems. People who set up and maintain software environments use **cron** to schedule **jobs** (commands or shell scripts) to run periodically at fixed times, dates, or intervals. This is used to give the users a % of their energy, will and guts every 5 minutes, as long as their max amount allowed hasn’t been reached. As you can see from the below script, each query will update the users table and set either energy or will +a certain amount. These crons are essential to the development of the game.

```
$db->query($query4);
//energy , will update
$query="UPDATE users SET energy=energy+(maxenergy/(20)) WHERE energy<maxenergy";
$query2="UPDATE users SET energy=maxenergy WHERE energy>maxenergy";
$query3="UPDATE users SET will=will+10 WHERE will<maxwill";
$query4="UPDATE users SET will=maxwill WHERE will>maxwill";

$db->query($query);
$db->query($query2);
$db->query($query3);
$db->query($query4);
```

Figure 17 5 Min Cron php file

4.6.2 Main Menu

Figure 18 Navi Menu from my mainmenu.php

The navigation bar is one of the most used parts of this game. Users use the navigation bar to go anywhere in the game and when they get a mail or event, it will notify them and become bold until they read the event.

The way this was implemented was to put it into a separate file and reference that file in each subsequent file. This decreased the need for writing unnecessary code and the names of the links could be edited game-wide from one file, rather than going through them manually.

The menu also changes depending on where the user is, for example in the hospital, home would be replaced with just a hospital view.

4.6.3 Anti SQL Injection

mysql_real_escape_string is a library provided by the MySQL library, which automatically prepends backslashes to the following characters: *\x00*, *\n*, *\r*, \, ', " and *\x1a*. This function must always be called before you send a query to MySQL. Although this looks extremely simple, it was probably one of the parts I had the most difficulty with. I had a test for SQL injection for my form posts, and no matter what I did I couldn’t figure out why it wasn’t running the anti injection and I could inject SQL into my queries. The two lines $_POST and $_GET were all I needed in the end.

```
function anti_inject($campo)
{
    foreach($campo as $key => $val)
    {
        $val = mysql_real_escape_string($val);
        // store it back into the array
        $campo[$key] = $val;
    }
    return $campo; //Returns the the var clean


//the next two lines make sure all post and get vars are filtered through this functior
$_POST = anti_inject($_POST);
$_GET = anti_inject($_GET);

}
```

Figure 19 anti inject function in mysql_db class

4.6.4 Email Validation

To ensure users put in a valid email (for future developments like email verification, user verification, newsletters, etc) regular expression function was created to run it through on the signup page. Below is the code used for it, with comments of what everything does.

```
function valid_email($email) {
  // First, we check that there's one @ symbol, and that the lengths are right
  if (!ereg("^[^@]{1,64}@[^@]{1,255}$", $email)) {
    // Email invalid because wrong number of characters in one section, or wrong number of @ symbols.
    return false;
  }
  // Split it into sections to make life easier
  $email_array = explode("@", $email);
  $local_array = explode(".", $email_array[0]);
  for ($i = 0; $i < sizeof($local_array); $i++) {
    if (!ereg("^(([A-Za-z0-9!#$%&#038;'*+/=?^_`{|}~-][A-Za-z0-9!#$%&#038;'*+/=?^_`{|}~\.-]{0,63})|(\"[^(\\|\")]{0,62}\"))$", $local_array[$i])) {
      return false;
    }
  }
  if (!ereg("^\[?[0-9\.]+\]?$", $email_array[1])) { // Check if domain is IP. If not, it should be valid domain name
    $domain_array = explode(".", $email_array[1]);
    if (sizeof($domain_array) < 2) {
        return false; // Not enough parts to domain
    }
    for ($i = 0; $i < sizeof($domain_array); $i++) {
      if (!ereg("^(([A-Za-z0-9][A-Za-z0-9-]{0,61}[A-Za-z0-9])|([A-Za-z0-9]+))$", $domain_array[$i])) {
        return false;
      }
    }
  }
  return true;
}
```

Figure 20 Email Validation/Reg Expression code

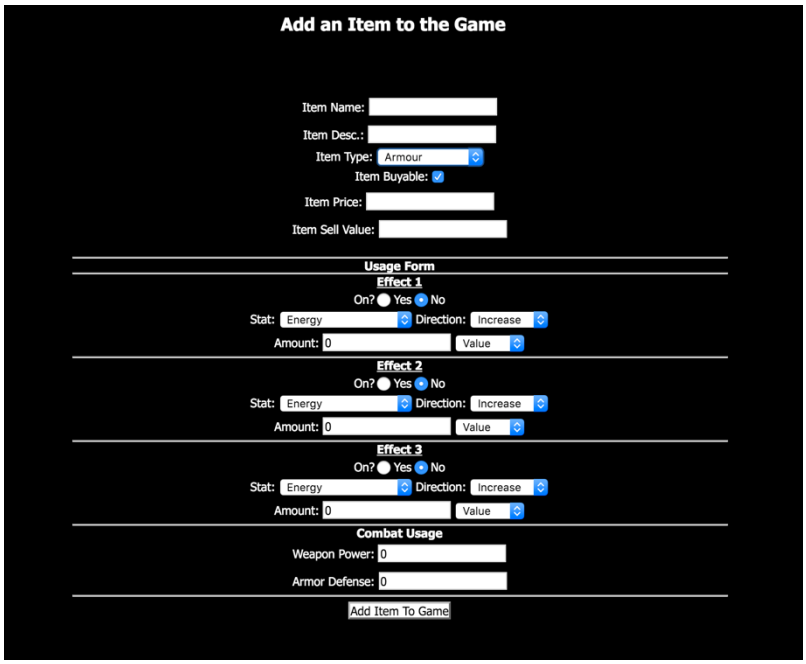
4.6.5 Scalability and Extensibility

While creating this particular game, it dawned on me that scalability and extensibility would be needed to be able to expand more and more as my user base grew, and began understanding the game more and completing the various challenges of the game.

This led me to thinking about a way of adding in extra features, items, property, crimes etc without the need to re write my whole site, or the components associated with the parts that needed to be updated or expanded upon.

There, the decision was made to use a *staff panel* to help me with this, as administrators of the game can assign multiple users as a staff and give them certain authority/permissions to update some parts of the game. Below will show the use of my function of adding an item as an example.

Figure 21 Add Item Form



This is the form used for adding the items, it is a much easier way to input the data into my database instead of having to edit the database directly. It also means a panel of staff could be given this permission and they can add items into the game that would benefit the gameplay.

Below you can see the code used when this form is submitted, as you can see there is error checking at the very beginning to ensure the user is above level 2, which is an admin, essential fields and inputted, and when they are, it goes on to post all the inputted data into the items table.

```
function new_item_submit()
{
global $db,$ir,$c,$h;
if($ir['user_level'] > 2)
{
die("403");
}
if(!isset($_POST['itmname']) || !isset($_POST['itmdesc']) || !isset($_POST['itmtype'])  || !isset($_POST['itmbuyprice']) || !isset($_POST['itmsellprice']))
{
print "You missed one or more of the fields. Please go back and try again.<br />
<a href='staff_items.php?action=newitem'>&gt; Back</a>";
$h->endpage();
exit;
}
$itmname=$db->escape($_POST['itmname']);
$itmdesc=$db->escape($_POST['itmdesc']);
$weapon=abs((int) $_POST['weapon']);
$armor=abs((int) $_POST['armor']);
if($_POST['itmbuyable'] == 'on') { $itmbuy=1; } else { $itmbuy=0; }
$efx1=$db->escape(serialize(array("stat" => $_POST['effect1stat'], "dir" => $_POST['effect1dir'], "inc_type" => $_POST['effect1type'], "inc_amount" => abs((int) $_POST['effect1amount']))));
$efx2=$db->escape(serialize(array("stat" => $_POST['effect2stat'], "dir" => $_POST['effect2dir'], "inc_type" => $_POST['effect2type'], "inc_amount" => abs((int) $_POST['effect2amount']))));
$efx3=$db->escape(serialize(array("stat" => $_POST['effect3stat'], "dir" => $_POST['effect3dir'], "inc_type" => $_POST['effect3type'], "inc_amount" => abs((int) $_POST['effect3amount']))));
$m=$db->query("INSERT INTO items VALUES('',{$_POST['itmtype']},'$itmname','$itmdesc',{$_POST['itmbuyprice']},{$_POST['itmsellprice']},$itmbuy, '{$_POST['effect1on']}',
 '$efx1', '{$_POST['effect2on']}', '$efx2', '{$_POST['effect3on']}', '$efx3', $weapon, $armor)");
print "The {$_POST['itmname']} Item was added to the game.";
stafflog_add("Created item {$_POST['itmname']}");

}
```

Figure 22 Add Item Function code

5. Testing

5.1. Game Testing

The only way to test a game is to play it. A few users, one admin user with staff administrators and various different stated players were created to test out the attack features and jobs. Send mails to different users via the mail function, try crimes with different crime skill. Try out the crons regeneration system and the storyline.

User action	**Expected result**	**Actual Result**
Access login.php page	Login.php page appears prompting for username and password.	As expected (pass)
User enters username and password and clicks login.	Index.php page is returned for that user	As expected (pass)
User clicks on Inventory	New screen Your Inventory is displayed	As expected (pass)
User clicks equip on a certain weapon	New screen “What would you like to equip it as” is displayed	As expected (pass)
User selects either primary or secondary weapon	“Equipped item successfully” is displayed	As expected (pass)
User clicks on the ‘back’ button to go back to their inventory	User is brought back to their inventory	As expected (pass)
User clicks on ‘Job’	User is brought to their job page, if no job then to the job select page	As expected (pass)
Assume user has no job: click on ‘Apply for Job’ on a job that has one or more red requirement.	Page displays ‘ Sorry you do not qualify for our requirements’ and user doesn’t get the job	As expected (pass)
User clicks on job with all green requirements.	New screen is displayed with ‘Congratulations, you’ve got the job’	As expected (pass)
User clicks ‘Go to Job’	User is brought to the job screen.	As expected (pass)
User clicks on Explore	User is brought to the city main screen. It shows all the links around the city including shops, their job, the fitness centre etc.	As expected (pass)
User clicks on Fitness Centre	Displays users current stats, and allows them to train while they have energy.	As expected (pass)
User clicks train while no energy	Display you have no energy	As expected (pass)
User clicks train while they have energy	Algorithm for gym works and displays how much they have gained.	As expected (pass)

These tests were very easy to do, but they mustn’t be overlooked as they gave a great insight into the fact that if the layouts work on every page, the algorithm in place are working correctly, the math is correct and there aren’t any bugs.

5.2. Other Testing

Apart from system testing above, there’s not much testing one can do on this. I did however test the validity of the database data by editing it directly in the database. This showed me instantly that my game was constantly getting the most up to date data, and that there was no redundancy.

5.3. Multi Device Testing

To ensure that my game worked on a wide variety of browsers and computers I used as many different machines as I could. For the purpose of this project, I have used 6 different devices to perform the testing. This list included:
13” Macbook Pro, Mac OS X 10.11.2 (El Captain) , Google Chrome, Safari.
15” Macbook Pro, Mac OS X 10.9.0 (Mavericks) , Google Chrome, Safari.
23” HP Monitor, Acer i7 (Windows 10) , Google Chrome, Microsoft Edge.
15” Lenovo Yoga 2 (Windows 8.1), Internet Explorer, Mozilla Firefox
13” Dell XPS (Windows 7), Mozilla Firefox, Opera
Effort was made to make sure that broad spectrum of device sizes and OS versions are covered. I tried to cover as many browsers too, and as you can see from my figure below, I took the 5 most popular browsers (w3schools 2016) and tested on them.

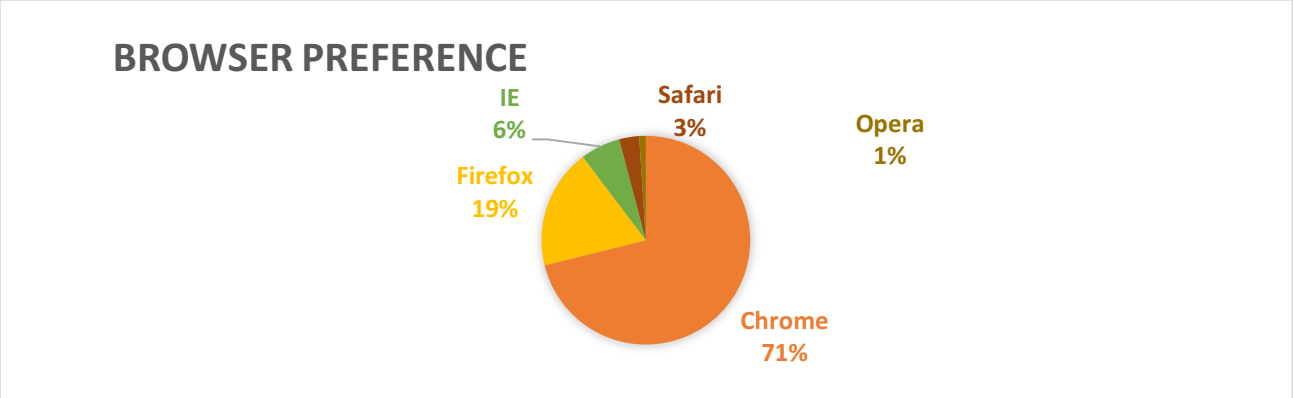


Figure 23 Browser Use Statistics

5.4. Known Issues and Design Flaws

During system and integration testing phase, there were a number of issues encountered, most of which have been resolved. However, some minor bugs still exist in the application that would need to be addressed in next application release. They are contained in the table below and were found too late in the process to be fixed without a risk of introducing new problems into the application.

Problem	Effect on the user	Solution
User doesn’t get logged out automatically, if they have cookies enabled	If the user leaves the tab open, they won’t have to re login, leaving a security gap.	It is caused because the session isn’t terminated after a certain amount of time, I will fix this before deploying it to the public.
Estate.php not working	Not showing different properties the user can buy	I did not have the time to find the right solution, part of the solution is my images were incorrect

6. Conclusion

The aim of the project was to create an online Role-Playing Game (RPG) to offer a social network aspect inside a gaming environment. I did create an online game, that allows users to send messages to each other, create their own, fake persona that they could either commit crimes, work in a job or attack and mug people to gain money. Although the game could be considered as an Alpha/Beta version of a fully functioning, sellable game, I do believe that its main purpose has been fulfilled.

The most challenging part was to get everything together in such a short period of time. The research performed around MonoGames was the most helpful thing as there is not much documentation on what I have done. It also gave me a good structure on how to code my game. Creating the different algorithms for level experience and crime experience was also quite challenging for game development.

Overall, I believe that this game would appeal to my target audience, and any gaming fanatic, especially when given more time to add new features to the product. There is a lot of room for improvement and enhancements, some of which are listed below:

- Create a possible Android/iOS application for the consumer to use on the go
- Develop the game storyline more
- Create ‘starter missions’ to help new players to navigate the game
- Create a ‘referral’ system for players so I can expand the player database

I have enjoyed working on this project, and despite few setbacks I think it allowed me to gain invaluable experience in technologies that were new to me, and also hands on experience with what I studied in my course, and I feel that should be very helpful in my professional career in the future.

7. References

Works Cited

Creative, Muran. *Pros and Cons of using a self hosted WP site.* 29 September 2014. http://murnancreative.com/pros-cons-self-hosted-wordpress-site/ (accessed February 14, 2016).
DBForums. *Starting a browser-based RPG, database design.* 08 06 2011. http://www.dbforums.com/showthread.php?1667179-Starting-a-browser-based-RPG-database-design (accessed November 28, 2015).
MCCodes. *McCodes.* 4 January 2005. http://mccodes.com/ (accessed December 10, 2015).
Sneftel, GameDev. *Game Dev.* 19 November 2003. http://www.gamedev.net/topic/192120-uml-for-games/ (accessed January 19, 2016).
TornCity. *Torn City.* 1 October 2006. www.torn.com (accessed February 5, 2016).
w3schools. *Browser Stats.* 1 March 2016. http://www.w3schools.com/browsers/browsers_stats.asp (accessed March 28, 2016).