Declaration

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

Stephen Bannable 7th April 2014
Acknowledgements

I would like to thank my supervisor Declan O’Sullivan for supporting and guiding me through this process. I would like to dedicate this project to my late grandfather, William Bannable, who was always proud of my academic achievements. I would also like to thank my friends and family for all your help and support.
Abstract

This report aims to outline the approach taken in implementing an application capable of matching participants in a Trinity College grinds programme. While institutions such as colleges and hospitals use much the same technologies for different aspects of their respective fields, it is believed that this is the first application to make use of these technologies for this kind of programme. It will simplify the work of the Trinity College Dublin Voluntary Tuition committee and aid the hundreds of tutors and tutees alike who participate in the Trinity College Dublin Voluntary Tuition Programme every year.
# Table of Contents

**DECLARATION** 2

**ACKNOWLEDGEMENTS** 3

**ABSTRACT** 4

**CHAPTER 1 INTRODUCTION** 8

1.1 **Aim** 9
1.2 **Motivation** 9
1.3 **Scope of the Project** 11

**CHAPTER 2 SYSTEM OVERVIEW** 12

2.1 **Customer Requirements** 12
2.2 **Non-Functional Requirements** 12
2.3 **Functional Requirements** 13
2.4 **Use Cases for the VTP** 14
   2.4.1 *Use case for the school pupil* 15
   2.4.2 *Use case for the Trinity Tutor* 15
   2.4.3 *Use cases for administrator* 16
2.5 **Chapter Summary** 17

**CHAPTER 3 SYSTEM DESIGN** 18

3.1 **Introduction** 18
3.2 **Architecture** 18
3.3 **Design of matching system** 19
   3.3.1 *Gale-Shapley Algorithm* 19
   3.3.2 *Applications that utilise the Gale-Shapley Model* 21
   3.3.3 *How the Gale-Shapley algorithm can be utilised in this project* 24
3.4 **Design of user interface** 26
3.5 **Database Design** 29
Chapter 1 Introduction

The purpose of tutoring is to help students help themselves and to assist or guide them to the point at which they become independent, successful learners. Tutoring benefits both the tutor and the tutee.

Benefits for the tutee include:

- Provides personalised attention
- Improves grades
- Increases knowledge and understanding of subjects
- Increases motivation to succeed
- Provides intensive practice
- Allows progress at own pace
- Leads to better use of study time
- Improves self-esteem and confidence
- Encourages higher levels of learning
- Encourages self-directed learning
- Reduces competition
- Provides praise, feedback, and encouragement
- Provides review of skills not mastered but no longer taught

Benefits for the tutor include:

- A sense of pride and accomplishment for having helped someone else
- Increased academic mastery (especially in cross age and peer tutor situations)
- Increased self esteem and confidence
- Enhanced sense of connection to their community
- Valuable career related experience

The Trinity College Voluntary Tuition Programme (VTP) is a free tuition service run and organised by Trinity College students. The VTP as a whole is a partnership between Trinity students, parents and volunteers from the local communities of Pearse Street and Ringsend. Every year the VTP committee members match Trinity
students, graduates and staff-members as tutors with children and teenagers studying in nearby schools. Each pair spends usually one hour a week in one of three centres, getting to know each other and working together on homework and other educational activities. The three centres used by the VTP are Ringsend Technical Institute in Ringsend, St. Andrew’s Resource Centre in Pearse Street and Goldsmith Hall in Pearse Street.

1.1 Aim

The aim of this project is to create an online matching system for VTP. Previously the matching system consisted of the VTP committee members collecting hundreds of registration forms from school pupils and tutors. The VTP committee would then have to manually pore over and analyse all of the registration forms and find a match between a school pupil and tutor.

This new system will pair together a school pupil and tutor in compliance with the criteria of:

- Venue selected
- Time selected
- Subjects selected

The idea of the matching system is to find tutors who are perfectly suited to a certain tutee and vice versa. When a pair is matched both parties receive a confirmation email and the details about their tutor/tutee.

1.2 Motivation

Having used the service back in 2005, I had a respectable knowledge of how the VTP service runs. The system runs as follows:

Upon arrival at the grinds students are asked what subjects they would like help with and other basic details such as name and year of study. Tutors are asked questions that are similar to the ones asked to the tutee, including what subjects
they would like to help students with. The coordinators of the grinds programme then try to match tutees with specific tutors to suit the preferences collected.

Through informal interviews conducted during the planning of this proposed idea, one has found that this system has not changed and is still in use. From past experience and the conversations, the general consensus is that the service is slow and out-dated. Tutors and tutees waste too much time being paired together at the centres. This time could be used for tuition and is instead wasted by attempting to tediously match pairs together.

I could scarcely believe that no one had attempted to improve the VTP service in all its longstanding years. However, this year there were slight improvements to the service. The new chairperson, Gareth Walsh, had created an online survey where tutees and tutors could enter their preferences online. This survey was only applicable to the tutors and not the school pupils. The survey was imperceptibly successful. It succeeded in simply reducing the number of registration forms collected from tutors and placed all preferences into a huge survey. However, after collecting this large amount of data it was still up to members of the VTP council to sift through all the participants and find suitable tutors for the tutees. Although this is an improvement the process is still quite slow.

“Although basic utilization of IT in the form of computers, e-mail, Internet access, and Web sites has grown relatively rapidly, there are nevertheless several critical deficiencies in the typical nonprofit’s employment of IT to help fulfill the organizational mission” – (Hackler & Saxton, 2007)

One of the main motivations for this application is my chance to re-pay the VTP for the help they had given me when I needed it during secondary school. Where other companies could spend money on web applications such as the one proposed, VTP is a non-profit organisation and can only rely on contributions such as this project. VTP is a superb organisation and deserves a superb website.
1.3 Scope of the project

In the project an online application called “The tutor matching system” was developed and evaluated. The application has three types of users: school pupils, Trinity College tutors, and administrators. School pupils will register with their information, the key aspects being their availability and subject choices. Similarly, Trinity College tutors register with their information, the key aspects being their availability and subject choices. An administrator can view the confirmed matches and he/she can also view the pending matches. A “verify” button will be provided for the administrator to confirm pending matches. The author will be designing a set of dynamic PHP [1] pages with JavaScript [2] to make user interaction pleasant and simple. The system will have a register student page, register tutor page and an administration page. The matching system will be deployed in the backend of the system.
Chapter 2 System Overview

This chapter provides an overview of the system. The overview identifies and examines the customer’s requirements, the functional and non-functional requirements, and the uses cases of the system.

2.1 Customer Requirements

The client for this project was the VTP committee. At each meeting the VTP committee was represented by their chairperson Gareth Walsh. The author met with Gareth a number of times over the course of the project to discuss various aspects of the website.

The first meeting between the author and Gareth took place on 22nd November 2013. In this meeting the author outlined the ideas for the website as presented in the original proposal submitted to the project supervisor. The author and Gareth worked through all the ideas put forth and decided on a structure for the website. In the next section the author will evaluation these customer requirements and present them as non-functional and functional requirements. Uses cases, with the aid of a UML use case diagram, of all the users will help give an overview of the system as a whole.

2.2 Non-Functional Requirements

This section discusses the non-functional requirements for the tutor matching system.

SECURITY

- Access Control: It must be secure. No one but the administrator should be able to access the matches in the admin page.

- SQL injection prevention: Every query in the backend of the system, that
is somehow dynamically generated, needs to be analysed for SQL injection. Every dynamic part of each query will be analysed to see where it come from. If it comes from anywhere that can be affected by user input then it needs to be properly treated.

**PERFORMANCE**

- Every part of the system needs to work in real time. Tutor and school pupil registration forms and the matching process needs to perform in real time.

**USABILITY**

- The system is designed in a user friendly way.

### 2.3 Functional Requirements

This section discusses the functional requirements for VTP by presenting UML use cases for the system.

- A student can register
- A tutor can register
- There will be automatic “perfect” matches that will confirmed routinely
- A set of pending matches if the match is not “perfect”
- An admin page giving the administrator an overview of the confirmed and pending matches

- When a pair is matched both will receive emails containing all the all relevant information of the match including subject, venue, and contact details of the school pupil/ tutor

- A limit of 20 tutors and school pupils per venue
• When school pupil of tutor selects “primary school” he/she should not be allowed to choose subjects.

2.4 Use Cases for the VTP

This section discusses the Functional requirements for tutor matching system by presenting UML use cases for the system, as overviewed in figure 2.1.

![UML use case diagram for tutor matching system](image)

Figure 2.1 UML use case diagram for tutor matching system
### 2.4.1 Use case for the school pupil
#### Register school pupil

<table>
<thead>
<tr>
<th>Trigger</th>
<th>The school pupil clicks the “Sign Up” button</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precondition</td>
<td>The school pupil is on the registration page</td>
</tr>
</tbody>
</table>
| Basic Path       | • The school pupil enters their name, school, email, year of study, the venue they wish to attend, 2 subjects that they would like help with in order of preference  
|                  | • The system checks that all the information is correct brings school pupil to the confirmation page |
| Post Condition   | Information stored in the database and school pupil is potentially matched |

### 2.4.2 Use case for the Trinity Tutor
#### Register Trinity Tutor

<table>
<thead>
<tr>
<th>Trigger</th>
<th>The Trinity tutor clicks the “Sign Up” button</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precondition</td>
<td>The Trinity tutor is on the registration page</td>
</tr>
<tr>
<td>Basic Path</td>
<td>• The Trinity tutor enters their name, email, year of study of the school pupil they would like to help, the venue they wish to attend, 2 subjects they would like to help a school pupil with</td>
</tr>
</tbody>
</table>
in order of preference
  • The system checks that all the information is correct brings Trinity tutor to the confirmation page

| Post Condition | Information stored in the database and Trinity tutor is potentially matched |

### 2.4.3 Use cases for administrator

<table>
<thead>
<tr>
<th>Trigger</th>
<th>Log in to admin page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Precondition</strong></td>
<td>Administrator is on the admin page</td>
</tr>
<tr>
<td><strong>Basic Path</strong></td>
<td>Administrator can view confirmed matches</td>
</tr>
<tr>
<td><strong>Post Condition</strong></td>
<td>Tutor is satisfied with confirmed match and it’s storage in the database</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trigger</th>
<th>Log in to admin page, and clicks “verify” button</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Precondition</strong></td>
<td>Administrator is on the admin page</td>
</tr>
<tr>
<td><strong>Basic Path</strong></td>
<td>Administrator can confirm pending matches</td>
</tr>
<tr>
<td><strong>Post Condition</strong></td>
<td>Tutor clicks “verify” button of pending match, tutor and school pupil receive emails and database is updated accordingly</td>
</tr>
</tbody>
</table>
2.5 Chapter Summary

The author began this project with a meeting with the client to acquire the customer requirements. After these requirements were outlined, the author separated them into two sections.
Chapter 3 System Design

3.1 Introduction

In the next section the author presents the three-tier architecture of the tutor matching system. The algorithm utilised in the execution of the matching system is explored and analysed. The author outlines the UI mock-ups that were used as aids for the design of the user-friendly website. Finally the database design is presented with a view of the tables contained in the database, and a view of the database schema.

3.2 Architecture

Figure 2.1 is the system architecture for the tutor matching system. The system architecture is divided into 3 subsystems – the presentation layer/user interface, the middle layer and the back-end. The presentation layer is used for displaying user interface of the data, dealing with user interaction and communicating with the middle layer. The middle layer deals with all queries. The middle layer analyses the request and then queries the database for the relevant data. Once the data is returned from the database, the middle layer processes it and returns it in the minimal and ideal format for consumption in the user interface layer.
3.3 Design of matching system

3.3.1 Gale-Shapley Algorithm

The matching system was designed using the Gale-Shapley algorithm. In the next section the author will introduce the Gale-Shapley algorithm and how it is utilised in modern applications. The author will specify how the algorithm was applied to this project.

The notion of stability is a central concept in cooperative game theory which seeks to determine how any constellation of rational individuals might cooperatively choose an allocation. The primary architect of this branch of game theory was Lloyd Shapley, who developed its main concepts in the 1950s and 1960s. In 1962, Shapley applied the idea of stability to a special case. In a short paper, joint with David Gale, he examined the case of pairwise matching: how individuals can be paired up when they all have different views regarding who would be the best match.

The underlying idea of the Gale-Shapley algorithm is to have one set of people (men, or women) do all the proposing of matches, starting with their most preferred partners and working their way toward less preferred ones, while the other set accepts or rejects matches, working their way from less preferred to more preferred partners. See figure 4.2 for outline of the Gale-Shapley model.
More specifically, a man begins by proposing a match with his most preferred partner. If this woman is not yet matched, then she provisionally accepts and the pair is placed into the matching. On the other hand, if she is already matched, then she does one of two things: (i) if the man is lower ranked on her preference list than her current partner, then she rebuffs the proposal, otherwise (ii) she accepts the proposal and rejects her current partner. If the man's proposal is unsuccessful, he repeats the process, moving down his preference list. Once a man is matched, we move to the next unmatched man and start at the next woman down in his list. We repeat this process until all men are matched. A catchy way to remember the procedure is that “men propose, women dispose.” See figure 2.2 for outline of the Gale-Shapley model.

Figure 2.2 Gale-Shapley Algorithm

Without yet going into the data structures for storing and managing the preference lists, the pseudo code for the Gale-Shapley algorithm is straightforward. See figure 2.3 for pseudo-code of the Gale-Shapley algorithm
This algorithm is guaranteed to halt and upon termination, it returns a stable matching. Additionally, the order in which we consider unmatched men has no impact on the stable matching that is found. And finally, this version of the Gale-Shapley algorithm, in which men do all the proposing, is called “man optimal” because when the algorithm halts, men are matched with their highest-ranked woman, conditioned on the matching being stable. (If the women did all the proposing, then it would be the woman-optimal version of the algorithm.)

3.3.2 Applications that utilise the Gale-Shapley Model

The following section will outline applications which currently use the

- Matching doctors and hospitals

In the U.S., students who graduate from medical school are typically employed as interns at hospitals, where they comprise a significant part of the labor force. In the early 1900s, this market was largely decentralised. During the 1940s, competition for scarce medical students forced hospitals to offer internships increasingly early. Matches were made before the students could produce evidence of their qualifications, and even before they knew which branch of medicine they would like to practice. When an offer was rejected, it was often too late to make offers to other candidates. A market ridden with such problems does not produce stable matches, because not enough offers can be made in time to ensure mutually beneficial trades. In order to make more offers quickly,
hospitals imposed strict deadlines for responding to offers. This, in turn, forced students to make early decisions without knowing what other opportunities would become available later on.

In response to these problems, a centralised system, called the National Resident Matching Program (NRMP), was introduced in the early 1950s. In a paper from 1984, Alvin Roth studied the algorithm used by this system and discovered that it was closely related to the Gale-Shapley algorithm. He then hypothesised that the fundamental reason for the success of the NRMP was that it produced stable matches.

_matching students and high schools_

Up until 2003, applicants to New York City public high schools were asked to rank their five most preferred choices, after which these preference lists were sent to the schools. The schools then decided which students to admit, reject, or place on waiting lists. The process was repeated in two more rounds, and students who had not been assigned to any school after the third round were allocated through an administrative process. However, this did not provide the applicants with enough opportunities to list their preferences, and the schools did not have enough opportunities to make offers. As a result, about 30,000 students per year ended up at schools they had not listed. Moreover, the process gave rise to misrepresentation of preferences. Since schools were more likely to admit students who ranked them as their first choice, students unlikely to be admitted to their favourite school found it in their best interest to list a more realistic option as their first choice, while applicants who simply reported their true preferences suffered unnecessarily poor outcomes. In 2003, Roth and his colleagues helped redesign this admissions process, based on an applicant-proposing version of the Gale-Shapley algorithm. The new algorithm proved to be successful, with a 90% reduction in the number of students assigned to schools for which they had expressed no preference.

_CAO and the college acceptance_

From the information published at the website of the Central Applications Office it seems that the college-proposing Gale-Shapley algorithm is used with some
special features. One is that students can apply for ‘level 8’ and 'level 7/6' courses simultaneously, and these applications are processed separately, so a student may receive more than one offer at a time. There are deadlines for accepting offers and if offers are rejected then further offers are made by the higher education institutions.

Spanish and Turkish college applications

The Spanish and Turkish higher education matching schemes are all based on a centralised scoring system. Students are assigned a score with regard to each programme they applied to; these scores are coming mainly from their grades and entrance exams. The scores of a student may differ at two programmes, since when calculating the score of a student for a particular programme only those subjects are considered which are relevant for that programme. The solution of the admission processes is represented by the so-called score-limits, which are referred to as ‘base scores’ in Turkey and ‘cut-off marks’ in Spain. The score-limit of a programme means the lowest score that allows a student to be admitted to that programme. The score-limits together with the preferences of the students naturally induce a matching, where each student is admitted to the first place on his/her list where he/she achieved the score-limit.

Matching kidneys and patients

The matching settings described so far involve two sides that both make active decisions. Some real-world situations are one-sided. A practical example is the matching of kidneys and other human organs to patients in need of a transplant. How can this be accomplished in an efficient way?

This problem was studied by Shapley and his colleagues, again in the abstract and based on the notion of stability. The proposed algorithm is in fact very simple. It is based on an initial allocation of objects and subsequent swapping. A challenge in the case of human organs is that some kidney-patient pairs may not be compatible and that complex multilateral swaps may be quite time consuming. Again, a combination of theory and experimental work has been used to compare different versions of top trading. As a result, increasingly complex chains of kidney donations are now adopted in a number of U.S. states.


3.3.3 How the Gale-Shapley algorithm can be utilised in this project

The author created a system that utilised the matching system of the Gale-Shapley algorithm to create the tutor matching system.

The tutor matching system works in the following way:

- When a tutor registers, he/she provides the following information:
  - Name
  - Email
  - Study year of school pupil he/she wants to help
  - The time and venue that best suits them
  - 2 subjects

The key items for the matching process are underlined.

- When a school pupil registers, he/she provides the following information:
  - Name
  - Email
  - Year in school
  - The time and venue that best suits them
  - 2 subjects

The key items for the matching process are underlined.

All of the above information provided by the tutors and school pupils is stored in the tables in the database. An algorithm will then check the key items of the tutor and school pupil. If these items all match then both the tutor and school pupil are matched.

Unfortunately we will not always get a perfect match, as seen in the Gale-Shapley algorithm. This is why school pupils and tutors are asked to enter two subjects.

The author created a ranking system that will make sure every person is properly matched. The ranking system works as follows:
S = school pupil
T = tutor

The criteria of $s$.study_year = $t$.study_year AND $s$.availability = $t$.availability must always be met before the rest of the queries can execute.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(t$.subject$_1 = s$.subject$_1$ OR $t$.subject$_1$ = $s$.subject$_2$) AND $(t$.subject$_2 = s$.subject$_2$ OR $t$.subject$_2 = s$.subject$_1$)</td>
<td>Perfect Match</td>
</tr>
<tr>
<td>$t$.subject$_1 = s$.subject$_1$ AND $t$.subject$_2$ != $s$.subject$_2$</td>
<td>6</td>
</tr>
<tr>
<td>$t$.subject$_1$ != $s$.subject$_1$ AND $t$.subject$_2$ = $s$.subject$_2$</td>
<td>6</td>
</tr>
<tr>
<td>$t$.subject$_1$ != $s$.subject$_1$ AND $t$.subject$_2$ = $s$.subject$_1$</td>
<td>4</td>
</tr>
<tr>
<td>$t$.subject$_1$ != $s$.subject$_1$ AND $t$.subject$_1$ = $s$.subject$_2$</td>
<td>4</td>
</tr>
</tbody>
</table>

If the top rank is met by both a tutor and school pupil then they are automatically matched and are both sent an email confirming their match.

However if a pair receives a rank 6 or 4 then their details are passed into the “admin” page and put in a table. In this table, the administrator will be able to view all the possible matches for all of the school pupils and tutors who have a rank 6 or 4. If the administrator wants to confirm a match a button will be selected that will confirm the match and update the tables in the database.
3.4 Design of user interface

For the user interface design the author used wireframes to illustrate the presentation and format of the application. A website wireframe, also known as a screen blueprint, is a visual guide that represents the skeletal framework of a website. This is more desirable and time-efficient then building a prototype for the client to give feedback on. Wireframes were carried out using Creately mock-up pages. This is a graphical user interface builder application which gives the ability to portray the page layout and arrangement of the website’s content, including interface elements and helping establish functionality, and the relationships between different screen templates of a website. Once a satisfactory user interface was achieved, work began on the implementation of the design to construct the application. Figures 2.4 to 2.9 provide the illustrations of the final mock-ups that were accepted by the client.

Figure 2.4 User interface mock-up: School Pupil Registration form
Figure 2.5 User interface mock-up: Trinity Tutor Registration form

Figure 2.6 User interface mock-up: Home page
Figure 2.7 User interface mock-up: About Us page

Figure 2.8 User interface mock-up: Contact Us page
3.5 Database Design

All data used in the tutor matching system is stored, retrieved and updated from a database. The database was set up using MySQL to handle all query executions.

The database for the tutor matching system consists of the following tables:

<table>
<thead>
<tr>
<th>Table</th>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>(school_id)</td>
</tr>
<tr>
<td></td>
<td>name_school</td>
</tr>
<tr>
<td></td>
<td>school_type</td>
</tr>
<tr>
<td></td>
<td>capacity</td>
</tr>
<tr>
<td>subject</td>
<td>(subject_id)</td>
</tr>
<tr>
<td></td>
<td>subject_name</td>
</tr>
<tr>
<td>students</td>
<td>(student_id)</td>
</tr>
<tr>
<td>Student/Teacher</td>
<td>student_first_name</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Tutor</td>
<td>tutor_id</td>
</tr>
</tbody>
</table>

The database schema for the tutor matching system is seen in figure 2.10.
3.6 Chapter Summary

In this chapter the author has explored the three-tier architecture utilised in this system. The reader was presented with examples of the Gale-Shapley algorithm and how it was utilised in the matching system. UI mock-ups have allowed the reader to see what the design of the user interface will look like when the project is complete. Finally, the author portrayed the tables of the database and the database schema.
Chapter 4 Implementation

The work carried out in order to ensure the objectives of the project were met is outlined in this chapter. The author begins the chapter by introducing the technologies utilised in the implementation of the project. After this the reader is introduced to the construction and web navigation sections of the chapter, where the author outlines how construction was started how the UI mock-ups from Chapter 3 were implemented. The author then outlines the construction of registration forms and the matching system using snippets of code from the relevant areas of the project. The chapter is concluded with an implementation schedule and screen shots of the tutor matching system website.

4.1 Technologies

This section discusses the technical environment and the software that was used to create this new system. It was fully implemented in Text Wrangler and the database is stored in “phpMyAdmin” on MySQL using the XAMPP webserver.

4.1.1 Software Interfaces

For the purpose of creating and designing this web-based application the XAMPP software tool was used. XAMPP was used because it is very useful as a development tool, for testing purposes on a local computer without access to the Internet. The web development application used to create the tutor matching system is Text Wrangler.

Software that was required in developing the tutor matching system is shown in the table below:

<table>
<thead>
<tr>
<th>Software</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAMPP</td>
<td>Web Server</td>
</tr>
<tr>
<td>Text Wrangler</td>
<td>Develop the system</td>
</tr>
</tbody>
</table>
4.1.2 Programming Languages

The programming languages used for the purpose of constructing this project are HTML [3], CSS [4] with some JavaScript support for the frontend and MySQL and PHP[5] for the backend. CSS was essential for the development of this system, as it must be user friendly and have a good design.

Programming languages that were required in developing the tutor matching system is shown in the table below:

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>Frontend</td>
</tr>
<tr>
<td>CSS</td>
<td>Frontend</td>
</tr>
<tr>
<td>JavaScript</td>
<td>Frontend</td>
</tr>
<tr>
<td>PHP</td>
<td>Backend</td>
</tr>
<tr>
<td>MySQL</td>
<td>Backend</td>
</tr>
</tbody>
</table>

4.3 Web navigation

One of the main requirements for VTP was the website acted as an advertisement for the VTP. Hence, a user can navigate through the website, reading information about the VTP, its services, location, venues, session times. This was all implemented by creating a “header.php” within a “div” with a list of these options. Each of these options were provided with different page links, and they all include the common component “header.php” so that the users had the option to navigate through each page (See figure 4.1). The content of the frontend included forms, tables and information (created using HTML and CSS) that is retrieved from or stored in the database through the backend activities. To provide a user friendly and interactive user interface, the bootstrap [6] framework was used. It
assisted in creating fashionable icons, forms, tables and placeholders for the frontend design.

4.4 Registration forms

The following section outlines how the registration forms were implemented. This section is broken into three parts: 4.4.1 static form fields, 4.4.2 dynamic form fields and 4.4.3 processing of data from the form.

4.4.1 Static form field

The registration forms for both the school pupil and tutor were quite similar. The only element that was different in both forms was “school”; school pupils had to provide their school name. The first three elements were one-line input fields. These fields were:

- First Name
- Surname
- Email
Figure 4.2 is an example of one of these one-line input fields:

```html
<label for="student_first_name">First Name</label><br>
<input type='text' name='student_first_name' />
```

Figure 4.2: one-line input field example

The next four items were the difficult part of the registration form. These four parts are:

- Study year
- Availability
- Subject 1
- Subject 2

### 4.4.2 Dynamic form fields

To meet the condition of the non-functional requirement "Usability" (see chapter 2 section 2.2) the author created a simple selection process for the user to avoid errors and confusion. This process was achieved using dynamically updated drop down lists (DDL) for both “availability” of the user and “subject” choice of the user.

#### Availability

The first DDL, “Study Year” or “Year of study that you wish to tutor in”, had three selections:

1. Senior Cycle (4th year and above)
2. Junior Cycle (3rd year and below)
3. Primary School
These three selections were hard-coded into the website. The author hard-coded these selections because they will not change over time. The next DDL was “availability”.

An entity exists in the table “schools”, which stores the currently configured availabilities of venues. Upon loading the form, a PHP script retrieves the data from the form and generates a JSON object with the schools types along with each “school type’s” availabilities, which are then passed through to the front-end.

Using JavaScript, the front-end catches an “onChange” event on the “year of study” DDL. This then determines the selected value and with the data provided in the JSON object, populates the availabilities DDL for the particular school type, which in this case will be “0” (Primary School) or “1” (Junior Cycle or Senior Cycle).

Doing this ensures that the user can only select valid and available inputs, working towards better usability of the system.

Figures 4.3 and 4.4 outline the code that initialises the DDL for availability:

Figure 4.3 is the file “functions.php”. This code contains a function labelled “genSchoolJSON”. This function creates the arrays for the “school type” (primary or senior cycle or junior cycle).
<?php
    include "config.php";

    function genSchoolJSON($subject, $avail) {
        $qry = mysql_query("SELECT school_id, school_type, name_school FROM school LEFT OUTER JOIN ( SELECT COUNT( * ) AS curr, ",$avail, " FROM ",$subject, " GROUP BY ",$avail," ) AS cap ON school_id = ",$avail," WHERE (curr < (capacity/2) OR curr IS NULL) and capacity > 0");

        $result = array(0 => array(), 1 => array());

        while($res = mysql_fetch_array($qry)){
            array_push($result[$res["school_type"]], array("id" => $res["school_id"], "name" => $res["name_school"]));
        }

        return json_encode($result);
    }
?>

Figure 4.3 functions.php

Figure 4.4 is a picture of the file ""scripts.js". This JavaScript file utilises the arrays created in "functions.php". The function in this JavaScript file is labeled "handleTypeSelect". This function checks the entry in the “year of study/year of study you wish to tutor in” and dynamically updates the other DDLs

```javascript
function handleTypeSelect(src, avail_id) {
    var val = $(src).val();
    var availSelect = $('#'+avail_id);
    var typeIndex = 2;
    if(val.toLowerCase() === "primary school"){
        typeIndex = 0;
        $('#subjects').hide();
    } else if(val.toLowerCase() === "junior cycle" || val.toLowerCase() === "senior cycle") {
        typeIndex = 1;
        $('#subjects').show();
    } else {
        return;
    }
    availSelect.html"");
    availSelect.append($("<option value="null" selected>--Select One--</option>");
    for(var i = 0; i < schools[typeIndex].length; i++) {
        var elem = $("<option value=""+schools[typeIndex][i].id">"+schools[typeIndex][i].name"></option>");
        availSelect.append(elem);
    }
}
```

Figure 4.4: “handleTypeSelect” function
These two functions were imported are then utilised in both the school pupil and tutor registration pages (see figure 4.5 and 4.6).

```php
var schools = <?php echo genSchoolJSON("students", "student_availability"); ?>;
```

Figure 4.5: genSchoolJSON

```html
<label for="year_of_study">Year of study</label>
<select name="study_year" id="study_year" onchange="handleTypeSelect(this, 'student_availability')">
   <option value="null" selected>---Select One---</option>
   <option value="Senior Cycle">Senior Cycle</option>
   <option value="Junior Cycle">Junior Cycle</option>
   <option value="Primary School">Primary School</option>
</select>
```

Figure 4.6: handleTypeSelect

### Subjects

Pulling data from the table “subjects” table using a SELECT statement and a while loop populated both subject choice DDLs. Figure 4.7 is an example of how the subjects section of the DDL works:

```php
<?php
   $query = "SELECT DISTINCT subject_name FROM subject";
   $result = mysql_query($query);
   if($result)
   {
      while($row = mysql_fetch_array($result))
      {
         echo "<option value="$row['subject_name']."">$row["subject_name"]</option>";
      }
   }
</select>
```

Figure 4.7 subjects DDL
As stated in the functional requirements, no subject preference is required for primary school selection. To handle this requirement the author inserted a small piece of code into the “handleTypeSelect” function. By utilising the indexes of the arrays created in the file “functions.php”, the author could stop the user from using the subject DDLs if they selected “primary school”. Figure 4.8 is the code used to handle this condition of this functional requirement.

```javascript
if(val.toLowerCase() === "primary school"){
    typeIndex = 0;
    $('#subjects').hide();
} else if(val.toLowerCase() === "junior cycle" || val.toLowerCase() === "senior cycle") {
    typeIndex = 1;
    $('#subjects').show();
```

Figure 4.8 subjects DDL for primary school

**Capacity**

The author included in the project a piece of code that would determine the capacity of the room. If the capacity was above or equal to 40 individuals then the venue selection would become unavailable in the DDL. It is worth noting that capacity is configurable and does not necessarily have to be 40. This capacity feature was included in the “functions.php” file (see figure 4.9).

```sql
WHERE (curr < (capacity/2) OR curr IS NULL) and capacity > 0"
```

Figure 4.9 capacity feature in “functions.php”

**4.4.3 Processing of data from the form**

Once the user hits the “sign up” button, the data is sent to the relevant PHP script for processing. This happens by wrapping the input fields with a form element, which uses http post method to send the data to the script.
Figure 4.10 is an example of how the post method is used to send data to the script:

```php
@student_first_name = mysql_real_escape_string($_POST['student_first_name']);
@student_surname = mysql_real_escape_string($_POST['student_surname']);
```

Figure 4.10: post method is used to send data to the script

The “my_sql_real_escape_string” part of this piece of code ensures that SQL injection is prevented by checking the user input when it is submitted. PHP provides this function for escaping such potentially dangerous inputs. The use of this function satisfies the second condition of the non-functional “Security” requirement (see chapter 2 section 2.2). These treated values can then be inserted into the relevant tables within the database with the following piece of code (figure 4.11):

```sql
$query2 = "insert into students (student_first_name) values ('$student_first_name')";
```

Figure 4.11: insertion of data into the database

The matching process described in the design section (see chapter 3, section 3.3.3) is performed at this stage. The implementation of the matching process will be described in the next section.

### 4.5 Matching system

The following section describes how the matching process was implemented. The matching process was implemented following the guidelines of the design chapter (see chapter 3, section 3.3.3).

When both a tutor and school pupil are matched, whether it is through an automatic match or pending match, both the student and the tutor tables are updated to reference to each other. This links the entries to show that they are matched together.
4.5.1 Perfect Match/Confirmed Match

The “perfect match” criteria check is performed in the registration script. The author has produced an algorithm that attempts to match the users automatically based on predefined “perfect match” criteria.

As stated in chapter 3 section 3.3.3, the criteria for a perfect match is if both the tutor and school pupil select the same:

- "Year of study", or “Year of study you wish to tutor in” in the case of the tutor
- Availability
- Subject 1
- Subject 2

A MySQL “SELECT” statement selects the criteria needed for a perfect match in the student/school pupil database and compares this criteria with the data inputted by the tutor.

Figure 4.12 is the code produced for this “perfect match”:

```
"select *
from tutors INNER JOIN school ON tutor_availability = school_id
where tutor_availability = '$student_availability'
AND study_year = '$study_year'
AND (tutor_subject_1 = '$student_subject_1'
OR tutor_subject_1 = '$student_subject_2')
AND (tutor_subject_2 = '$student_subject_2'
OR tutor_subject_2 = '$student_subject_1')
AND (tutors.student_id is null);
```

Figure 4.12 code for the “perfect match”
4.5.2 Imperfect match

The “imperfect match” criteria check is performed in the admin script. As stated in chapter 3 section 3.3.3, the author produced an algorithm that attempts to match users according to a certain “rank”. These users have not been “perfectly matched” and remain unmatched in the database.

There are two possible imperfect ranks:

1. **Rank score of 4**

As stated in chapter 3 section 3.3.3, the criteria for an imperfect match with a rank score of 4 is if both the tutor and school pupil select the following:

- The same “Year of study”, or “Year of study you wish to tutor in” in the case of the tutor
- The same “availability"

AND

- tutor subject 1 does not equal school pupil subject 1 AND tutor subject 2 equals school pupil subject 1

OR

- tutor subject 1 does not equal school pupil subject_1 AND tutor subject 1 equals school pupil subject 2
Figure 4.13 is the code produced for this “imperfect match” with a rank of 4:

```sql
SELECT students.student_id as s_id, tutors.tutor_id as t_id, 4 as rank,
  school.name_school as venue, students.student_subject_1 as ss1, students.student_subject_2 as ss2, tutors.tutor_subject_1 as ts1,
  tutors.tutor_subject_2 as ts2 FROM students, tutors INNER JOIN school on tutors.tutor_availability = school.school_id WHERE
  ((student_subject_1 != tutor_subject_1 AND student_subject_1 = tutor_subject_2) OR
   (student_subject_1 != tutor_subject_1 AND student_subject_2 = tutor_subject_1)) AND
   students.tutor_id IS NULL AND tutors.student_id is null ORDER BY rank DESC);
```

Figure 4.13 imperfect match rank of 4

2. Rank score of 6

As stated in chapter 3 section 3.3.3, the criteria for an imperfect match with a rank score of 6 is if both the tutor and school pupil select the following:

- The same “Year of study”, or “Year of study you wish to tutor in” in the case of the tutor
- The same “availability”

AND

- tutor subject 1 equals school pupil subject 1 AND tutor subject 2 does not equal school pupil subject 2

OR

- tutor subject 1 does not equal school subject 1 AND tutor subject 2 equals school pupil subject 2
Figure 4.14 is the code produced for this “imperfect match” with a rank of 6:

```
$spendingQuery = mysql_query("SELECT students.student_id as s_id, tutors.tutor_id as t_id, 6 as rank,
    school.name_school as venue, students.student_subject_1 as ss1, students.student_subject_2 as ss2,
    tutors.tutor_subject_1 as ts1, tutors.tutor_subject_2 as ts2 FROM students, tutors INNER JOIN
    school on tutors.tutor_availability = school.school_id WHERE
    ((student_subject_1 = tutor_subject_1 AND student_subject_2 != tutor_subject_2) OR
    (student_subject_1 != tutor_subject_1 AND student_subject_2 = tutor_subject_2)) AND
    students.tutor_id IS NULL AND tutors.student_id is null
```

Figure 4.14: imperfect match rank of 6

Analysing the two queries one found that both are very similar. To avoid having to use two queries and perform programmatic merging and sorting of the data the author used the SQL UNION functionality to combine them into a single query. The results of which can then be ordered using the SQL ORDER BY functionality.

4.5 Admin page

This page was designed differently than the other pages in the project. HTML and CSS was included. However the “header.php” common component was not included in this page. Instead the author created a simple table which would display the results of the confirmed and pending matches. This table contained both the confirmed and pending matches. The information for the confirmed matches table was obtained through a MySQL SELECT statement. The information for the pending matches table was obtained through the select SELECT statement form section 4.5.2, where both of the ranks are combined using SQL UNION. A while statement and a “my_sql_fetch array” statement obtained the data from the database and displayed it in both tables. A code snippet for both the confirmed and pending matches is contained in figures 4.15 and 4.16. This information was then displayed in each of the respective tables. A screen shot of these tables can be seen in figure 4.17
Figure 4.15 Confirmed Matches

```
<table class="w95p">
  <tr>
    <td><strong>Student Name</strong></td>
    <td><strong>Tutor Name</strong></td>
    <td><strong>Venue</strong></td>
    <td><strong>Subject 1</strong></td>
    <td><strong>Subject 2</strong></td>
  </tr>
  <?php
  while($res = mysql_fetch_array($confirmQry)) {
    echo '<tr>
          <td>'.htmlspecialchars($res['fname']).'</td>
          <td>'.htmlspecialchars($res['ssname']).'</td>
          <td>'.htmlspecialchars($res['tfname']).'</td>
          <td>'.htmlspecialchars($res['tsname']).'</td>
          <td>'.htmlspecialchars($res['ssname']).'</td>
        </tr>
    }
  ?>
</table>
```

Figure 4.16 pending matches

```
<table class="w95p">
  <tr>
    <td><strong>Student Subject 1</strong></td>
    <td><strong>Student Subject 2</strong></td>
    <td><strong>Tutor Subject 1</strong></td>
    <td><strong>Tutor Subject 2</strong></td>
    <td><strong>Verify</strong></td>
  </tr>
  <?php
  while($res = mysql_fetch_array($pendingQry)) {
    echo '<tr>
          <td>'.htmlspecialchars($res['venue']).'</td>
          <td>'.htmlspecialchars($res['ssname']).'</td>
          <td>'.htmlspecialchars($res['tssname']).'</td>
          <td>'.htmlspecialchars($res['tsname']).'</td>
          <td><a href="?act=verify&sid='.htmlspecialchars($res['sid']).'&tid='.htmlspecialchars($res['t_id']).'">Verify</a></td>
        </tr>
    }
  ?>
</table>
```
The author implemented a verify button for the administrator to confirm a pending match. When the verify button is selected it uses http get parameters to send the match to be verified into a processing PHP script. The verify button can be seen in figure 4.17. When the verify button is pressed it updates the pair as confirmed matches. This page is password protected to meet the requirement of “Security” (see chapter 2 section 2.2).

4.6 Confirmation emails

Confirmation emails of matches between pairs were sent using the “sendmail” function. All emails contained information retrieved from the database. Figure 4.18 is one of the emails used in the system. An example of one of the “confirmed match” emails is contained in figure 4.19.

```php
$msgStudent = 'Dear, 

You have been matched with the tutor: ' . $row['tutor_first_name'] . ' ' . $row['tutor_last_name'] . '. For the subjects: ' . $student_subject_1 . ' and ' . $student_subject_2 . '. The venue is: ' . $row['name_school'] . ' 

mail($student_user_email, 'Match', $msgStudent);
```

Figure 4.18 Email for confirmation of match
4.7 Implementation Schedule

The table below sets out the implementation schedule that was prepared. This acted as a guideline in implementing the project. It was particularly useful in organising the work involved into a hierarchy of tasks.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Start</th>
<th>Finish</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Name)</td>
<td>(Date)</td>
<td>(Date)</td>
<td>(In days)</td>
</tr>
<tr>
<td>System Requirement Specification</td>
<td></td>
<td></td>
<td>7 days</td>
</tr>
<tr>
<td>Design Process</td>
<td>28th Nov. 2013</td>
<td>20th Dec. 2014</td>
<td>22 days</td>
</tr>
<tr>
<td>System Implementation:</td>
<td></td>
<td></td>
<td>60 days</td>
</tr>
<tr>
<td>• Frontend website</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Register function</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Matching process</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Admin page and table
• Password security
• Email set up

| System Testing | 3rd May 2014 | 2nd April 2014 | 31 days |

4.8 Screen shots of tutor matching system website

Figure 4.20 Student Registration page

Figure 4.21 Tutor Registration page
Figure 4.22 Contact Us page

Figure 4.23 About Us page

Figure 4.23 Home Page
4.9 Chapter Summary

In this chapter the author has presented a complete overview of how the project was implemented. The programming languages and software interfaces utilised in this project were outlined. Construction of the user interface, registration forms, the matching system, the admin page, and the confirmation emails was analysed and demonstrated. The chapter was concluded with an implementation schedule and screen shots of the finished website.
Chapter 5 Evaluation

5.1 Introduction

To evaluate the application, the author surveyed tutors and school pupils that are currently participating in the VTP and members of the VTP committee. Overall 80 participants were surveyed for this evaluation. The author, with permission from the VTP committee, attended two sessions. The first session was in St. Andrew’s Resource Centre on a Monday evening at 6:30 pm; primary school students were surveyed during this session. The second session was in Goldsmith Hall on a Wednesday evening at 7pm; secondary school students were surveyed during this session. The process was simple, the author used give pairs of school pupil and tutor combinations and asked them to complete the “School Pupil” and “Tutor” registration pages respectively. They were then provided with a questionnaire sheet to answer, which they answered alone. The author left the room for their testing to ensure the respondents could be as honest as possible. The information sheet provided to participants is included in appendix 1.

5.2 The Questionnaire

The questionnaire was created the “System Usability Scale”. The System Usability Scale (SUS) provides a “quick and dirty”, reliable tool for measuring the usability. It consists of a 10-item questionnaire with five response options for respondents; from strongly agree to strongly disagree (see figure 4.16). Originally created by John Brooke in 1986, it allows you to evaluate a wide variety of products and services, including hardware, software, mobile devices, websites and applications.

SUS has become an industry standard, with references in over 1300 articles and publications. The noted benefits of using SUS include that it:

- It is a very easy scale to administer to participants
- Can be used on small sample sizes with reliable results
As stated previously, the SUS is a 10-item questionnaire with 5 response options. The 10 questions are:

1. I think that I would like to use this system frequently.

2. I found the system unnecessarily complex.

3. I thought the system was easy to use.

4. I think that I would need the support of a technical person to be able to use this system.

5. I found the various functions in this system were well integrated.

6. I thought there was too much inconsistency in this system.

7. I would imagine that most people would learn to use this system very quickly.

8. I found the system very cumbersome to use.

9. I felt very confident using the system.

10. I needed to learn a lot of things before I could get going with this system.

<table>
<thead>
<tr>
<th>Strongly Disagree</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.16 contains the response format used in the SUS.

To obtain an overall result from the questionnaire the author has to add up all the participant’s scores for each question and multiply the total by 2.5. Multiplying by 2.5 converts the original scores of 0-40 to 0-100.
5.3 The Results

This results section will be broken down into three parts:

1. Primary school results: This will include all of the results from the questionnaires gathered at the first session attended
2. Secondary school students: This will include all of the results from the questionnaires gathered at the second session attended
3. Committee members: This will include feedback from the client

1. Primary School results
Before beginning this section the author must point out that under the recommendations of the VTP the primary school pupils did not participate in this survey; instead a parent/guardian was a participant on the pupil’s behalf. The VTP committee are hoping to encourage parents of primary school pupils to use the website to register their child.

For the purpose of the getting a broad set of results the author surveyed 20 tutors and 20 parents/guardians of school pupils. The number of participants was also significant. There was a limit of 40 people per room. The author surveyed a full room in order to determine that the “capacity” feature (chapter 2 section 2.3 and chapter 4 section 4.4.2) of the website worked sufficiently.

The results of the survey are presented as bar charts and table form in figures 4.2 – 4.5
Figure 4.2 SUS scores for parents/guardians of primary school pupils

The blue bars represent the SUS score attained from each participant, ranging from 0-100. The red line represents the average score of the 20 participants. The y-axis represents the participants and the x-axis represents the SUS score.
<table>
<thead>
<tr>
<th>Participants</th>
<th>SUS score for parent/guardian of primary school pupils</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>52.5</td>
</tr>
<tr>
<td>P2</td>
<td>57.5</td>
</tr>
<tr>
<td>P3</td>
<td>52.5</td>
</tr>
<tr>
<td>P4</td>
<td>55</td>
</tr>
<tr>
<td>P5</td>
<td>57.5</td>
</tr>
<tr>
<td>P6</td>
<td>70</td>
</tr>
<tr>
<td>P7</td>
<td>72.5</td>
</tr>
<tr>
<td>P8</td>
<td>75</td>
</tr>
<tr>
<td>P9</td>
<td>77.5</td>
</tr>
<tr>
<td>P10</td>
<td>80</td>
</tr>
<tr>
<td>P11</td>
<td>82.5</td>
</tr>
<tr>
<td>P12</td>
<td>82.5</td>
</tr>
<tr>
<td>P13</td>
<td>87.5</td>
</tr>
<tr>
<td>P14</td>
<td>77.5</td>
</tr>
<tr>
<td>P15</td>
<td>82.5</td>
</tr>
<tr>
<td>P16</td>
<td>70</td>
</tr>
<tr>
<td>P17</td>
<td>80</td>
</tr>
<tr>
<td>P18</td>
<td>75</td>
</tr>
<tr>
<td>P19</td>
<td>72.5</td>
</tr>
<tr>
<td>P20</td>
<td>70</td>
</tr>
</tbody>
</table>

Figure 4.3 The SUS scores for parents/guardians of primary school pupils in table form

**Analysis:** The average score for this section was 71.5. The author was extremely pleased with this result. The first 5 participants had no smart phone. The author had to use his own phone to show these participants how the system worked. Both participants admitted they were not comfortable using smart phones or emails, hence the slight dip in the scores at the beginning as can be seen in figure 4.2. Participants 6-20 were all above 70, with the highest score of 87.5 attained from participant 13. Participant 13 was really pleased with how the system worked, and was extremely interested in the matching process. Overall the author was happy with the results.
Figure 4.4 SUS scores for tutors participating in primary school grinds

The blue bars represent the SUS score attained from each participant, ranging from 0-100. The red line represents the average score of the 20 participants. The y-axis represents the participants and the x-axis represents the SUS score.
Figure 4.5 The SUS scores for parents/guardians of primary school pupils in table form

**Analysis:** The average score for this section was 79.25. Tutor feedback was generally quite positive. All of the tutors said they really liked the matching system. Almost all of the tutors stated that they liked how they would now have direct contact with the parent/guardian of their pupil. Previously, tutors would show up at the primary school grinds without knowing if their pupil was going to attend. However now that they have direct contact they can find out beforehand if their pupil is attending. The first tutor gave the lowest score; the author was unable to find out more details about this tutor as they did not give much feedback after they had completed the questionnaire.

The “capacity” feature of the system worked perfectly during the testing of this session. At the end of the session, after all participants had entered their details into the system, the following change occurred in the user interface (see figures 4.4 and 4.5).
As can be seen from figures 4.5 and 4.6 the “capacity” feature of the system worked perfectly. Prior to the survey there was 40 free places in the session. After all the participants had entered their data there was no places left and therefore the selection of this session now became unavailable.

2. Secondary School results
Using the same method employed during the first session, the author surveyed 20 tutors and 20 secondary school students. As before, the author surveyed a full room in order to determine that the “capacity” feature of the website worked sufficiently.

The results of the survey are presented as bar charts and table form in figures 4.6 – 4.10
Figure 4.6 SUS scores for secondary school pupils participating in the grinds

The blue bars represent the SUS score attained from each participant, ranging from 0-100. The red line represents the average score of the 20 participants. The y-axis represents the participants and the x-axis represents the SUS score.
### Analysis:
The average score for this section was 81.25. The lowest score was a 72.5; this student had no smart phone, making the impact of his evaluation slightly skewed. The author was extremely pleased with this result. Overall the response was extremely positive. A large majority of the pupils enquired about how the system worked and how the general matching process worked. The students seemed very impressed with how the whole system worked, as can be viewed in the results provided. Participant 3 was particularly impressed with the system. This participant had used the old system in his Junior Cycle and hated the old matching method. His enthusiasm in the system is reflected in his score.
Figure 4.8 SUS scores for tutors participating in the secondary school grinds

The blue bars represent the SUS score attained from each participant, ranging from 0-100. The red line represents the average score of the 20 participants. The y-axis represents the participants and the x-axis represents the SUS score.
Figure 4.9 SUS scores for tutors participating in the secondary school grinds in table form

**Analysis**: The average score for this section was 81. Like the tutor survey for the primary school grinds, the feedback was really positive. Tutors seemed to really like this system. A lot of the tutors complained about the system that was currently being used. They claimed that it took the VTP committee too long to sort through the matches and find a pair. The tutors claimed that this process hindered the amount of help they could give students during the year. The tutors believed that the new system would create a more efficient matching process and would allow them to start working earlier with their student. Although it looks like there are some dips in the graph, the reader must take into account that all the scores were quite high, with the lowest score being 75. The highest score of 90 was achieved in this survey.

Once again, “capacity” feature of the system worked perfectly during the testing of this session. At the end of the session, after all participants had entered their
details into the system, the following change occurred in the user interface (see figures 4.8 and 4.9).

![Figure 4.10 DDL before the secondary school survey began](image)

Figure 4.10 DDL before the secondary school survey began

![Figure 4.11 DDL when the secondary school survey ended](image)

Figure 4.11 DDL when the secondary school survey ended

As can be seen from figures 4.8 and 4.9 the “capacity” feature of the system worked perfectly. Prior to the survey there was 40 free places in the session. After all the participants had entered their data there was no places left and therefore the selection of this session now became unavailable.

During the process of surveying both of the above sessions, the practicality of the “confirmed matches” and “pending matches” section in the system can to the fore. There was a large majority of “confirmed matches” during the course of both surveys. When a “pending match” was placed into the database the author immediately confirmed the match and the system worked perfectly.
It would have been ideal for a system to work in the exact way that the author wanted, whereby tutors and students have multiple pending matches according to their rank of match. Unfortunately due to time constraints the author had to match both pairs instantaneously so that the users could see receive their email confirming their match and fill out the questionnaire.

However the author did test this system with a number of fellow students after the evaluation. Figure 4.10 demonstrates how the ranking system works in the “admin” page.

As can be seen in figure 4.10. Tutors and pupils alike have a large set of matches. The matches are ordered in terms of rank. As seen in figures 4.11 and 4.12 when a “pending match” is verified both the tutor and the pupil are placed into the “confirmed matches” table, and their details are deleted from the “pending matches” table. Overall the “admin” page was a success.

3. Committee members
For the final part of the evaluation of the system, the author requested feedback from the client. Gareth and Aisling have guided the author in the design of this system. It was their requirements which lead to the creation of this new system. Therefore their feedback is the most important of all:

“In previous years, when matching in VTP, we’d be going through hundreds of sheets of application forms, for both tutors and kids, to find a tutor who was capable to teach the kid’s preferred subjects, and for a time slot that would suit both etc.

This involved weeks of work at the start of the year, which delayed the start date for tuition, and generally was very time-consuming for the students who volunteer on the committee.

Thanks to Stephen and his new, innovative website, matching tutors and kids will become a much more efficient and much quicker process in the
future. Stephen listened carefully to our brief, that we required a complex system whereby subject choices, subject levels, school years and days of the week were all variables which needed to be taken into account when making matches. Stephen’s nuanced and sophisticated system is a fantastic resource for the programme, allowing us to take on more children and get the tuition started earlier than ever. The impact the new website will have on the programme cannot be understated, it will revolutionise the VTP” – (Aisling Kelleher Chairperson VTP 2014-2015, Gareth Walsh VTP Chairperson 2013-2014).

5.4 Analysis of results

Based on the results of the evaluation process it can be deducted that the application worked appropriately and efficiently. The combined score of all of the SUS scores returned an average of 78.25 (71.5 + 79.25 + 81.25 + 81 = 313 /4). According to the guidelines of the SUS a score above a 68 would be considered above average and anything below 68 is below average. With this in mind the author is extremely pleased with the results

The functionality and operations returned positive results and the interaction between the database and the user interface was reacting very well. After the completion of the implementation and testing processes, the client’s specifications and requirements where reviewed in a meeting with the client. This was done to contrast the results and observations of the completed system with the objective of the project. Also during this meeting the author and the client produced a few future additions to the website for possible implementation after the project. These additions will be explored in the next chapter. Overall the client was satisfied with the end result and the project was a success.
5.5 Chapter Summary

In this chapter the author has outlined the way in the system was evaluated. This evaluation included questionnaire responses from 80 participants. The results of this evaluation were presented in chart and table form, and were fully analysed.
Chapter 6 Conclusions

6.1 Difficulties faced

There were a number of difficulties faced during the process of construction and implementation of this project.

There were numerous challenges faced when converting a paper-based system into an efficient electronic database. The VTP tutoring matching system was developed to manage the matching process of the committee, therefore it was essential that communication was clear and consistent, and requirements were fully understood and agreed upon. Meetings were held quite regularly between the author and Gareth Walsh. The author had to meet regularly with the client to identify the requirements, break down the design, and illustrate mock-ups to make sure that I fully comprehend the requirements and that the client was fully aware of what was to be expected. One of the major challenges occurred near the end of the project, the committee stated that they wanted a number of changes in the website. These changes were created due to new committee members being elected in the VTP. Another challenge identified while developing the project was with regards room constraints. The author did not initially take into account that the capacity of rooms would be a factor in the project; indeed Gareth did not even identify this problem until half way through the construction of the project. The committee had set a limit of 20 people per room per night. This was however quite a simple problem to overcome.

6.2 Further work

Although deemed an overall success, the application was not without its flaws. It was important to note that this system was the first prototype to be presented to the client and not the final version. Due to the time constraints of this project there were a few issues that required more work to be completed. This version was delivered to the client for feedback, so that improvements and changes can
be made where necessary and appropriately. The main issues and additional work required to make the system fully comprehensive are highlighted in this section:

Art and drama club

In the initial requirements document for the project Gareth had outlined that the committee had wanted an extra page on the website for the “art and drama club”. This was a club that had been running in the VTP for the last number of years. During April 2014 the VTP committee had decided not to continue with this club. However, just before this report was published the new VTP chairperson, Aisling, had contacted the author to arrange for the club to be set up again and a new page to be added to the website to accommodate this. This page will be added in due course.

Adding Ringsend Technical College to the selection of venues

The VTP also runs a service for primary school students in Ringsend. This service takes place in Ringsend Technical Institute. There are two sessions in the Ringsend service. Both sessions take place on Tuesday evenings; the first session is 6pm-7pm and the second session follows directly after at 7pm-8pm. Altogether the Ringsend sessions accommodate over 40 primary school pupils.

Originally the author included this service in the venue selection options on the website. However when the author was showing the client an early prototype of the website, the client identified the inclusion of the Ringsend service as a problem. The reason this was a problem, according to the client, was because in previous years the VTP had found it quite difficult to get Trinity students to go down to Ringsend Technical Institute as it is quite a long journey compared to the distance of the journeys to the other centres. The client stated that in previous years the committee would receive only three of four applications of
tutors who would attend the service in Ringsend. In most cases the VTP committee would have to contact certain tutors and ask them directly if they were willing to attend the Ringsend service.

For future work the author would like to include the Ringsend service in the choices of venue. With the use of the dynamically updated drop down lists options for venues will eventually decrease. Users using the site at a late stage of registration (usually registration takes place in September/October) may only be left with the choice of Ringsend and will therefore be forced to choose one of the Ringsend sessions.

**Tutor accessibility**

The author would like to add an authentication feature to the tutor registration page. This would stop people from outside Trinity College entering their details into the database and potentially causing a problem for the VTP committee.

**School Pupil accessibility**

Although the VTP is a non-profit organization, a small fee is accepted from school pupils. This small payment stops pupils from attending who are not that serious about partaking in the programme. The author would like to include a payment section on the school pupil registration form. This would pupils who are not serious about attending form entering their details into the database and potentially causing a problem for the VTP committee.

**6.3 Conclusion**

The purpose of the project was to create an online school pupil and tutor matching system for the VTP. This system would address the sluggish manual labour aspect of the existing matching method. The website would also provide school pupils and tutors with an easy-to-use online registration forms and information about the VTP.

Although there are other websites which provide access to grinds, this website is quite unique as it matches school pupils and tutors together through there
preference of venue, time and subjects. The automatic matching system is used to match the school pupil and tutor at an earlier stage so that a personal relationship can be built between the pair. This leaves less work for the VTP, who will no longer have to arrange the set up for school pupils and tutors.

In conclusion, this project demonstrates that the tutor matching system is on the right path to becoming a feasible replacement for the existing manual system previously employed by the VTP.
References


[4].W3Schools (2014). CSS Tutorial. Available at:

[5].W3Schools (2014). SQL Tutorial. Available at:
http://www.w3schools.com/sql/default.asp?PHPSESSID=300ae3404d5fa2612f238abeebb8869c [Accessed 20th January 2014].

[6]. Twitter Bootstrap (2014). Twitter Bootstrap. Available at:
Appendix

1. Questionnaire

**Tutor matching system**

The following is a questionnaire for the "tutor matching system".

The VTP are creating a new website in which school pupils and tutors are matched online. When a match is found both the student and the tutor will receive a confirmation email about their match.

The new "tutor matching system" is currently in its evaluation phase. Your contribution to this evaluation would be greatly appreciated.

Assuming that you have now used the system, please answer each of the ten questions. The responses are marked from 1-5 (1 being "strongly disagree" and 5 being "strongly agree"). If you have any questions please do not hesitate to ask the demonstrator.
### System Usability Scale

<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly disagree</th>
<th>Strongly agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. I think that I would like to use this system frequently</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. I found the system unnecessarily complex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. I thought the system was easy to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. I think that I would need the support of a technical person to be able to use this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. I found the various functions in this system were well integrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. I thought there was too much inconsistency in this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7. I would imagine that most people would learn to use this system very quickly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. I found the system very cumbersome to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9. I felt very confident using the system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>