Management Science and Information Systems Studies

Final Year Project Report

Archives Database for D.U. Players

Deirdre Van Wolvelaere March 2015
D.U. PLAYERS

Archives Database

20th March 2015

Prepared By: Deirdre Van Wolvelaere
Supervisor: Dr. Aideen Keaney
I declare that the work described in this dissertation has been carried out in full compliance with the ethical research requirements of the School of Computer Science and Statistics.

Signed: ___________________________

Deirdre Van Wolvelaere
20/03/2015
ABSTRACT

The aim of this project was to design and develop a prototype archive database system with a web interface for Dublin University Players (hereafter referred to as Players), the drama society within Trinity College Dublin. The prototype system was created for the use of the committee members of Players and the public. The system's public area allows users to view the events, festivals, productions and awards of Players. The system's password protected committee area allows users to view reports and insert, edit, delete and view records of events, festivals, productions, awards, members and committees. The system consists of a web front end, developed using a combination of PHP, CSS, HTML, JQuery and MySQL.
Dublin University Players (hereafter referred to as Players) is the drama society within Trinity College Dublin. The main client contact throughout the project was Donal McKeating, the chairperson of Players.

The project was successful in fulfilling the terms of reference and the client has given positive feedback about the system.

The purpose of this project was to develop an information system with a web interface to archive all activities within Players. Some areas of the system were to be available to the public and some areas were to be only available to committee members.

Over the course of the project I developed proficiency in a number of software languages, which, initially, were more difficult than expected. I furthered my knowledge of software development far beyond what I expected. PHP, MySQL, JQuery, HTML and CSS were used to build the application.

As the system is a prototype, the user interface has not been put on the internet. It has, however, been demonstrated to the client. The next phase for the client will be to put the system online. This will be discussed further in the report.

I would like to give a special thanks to both past and present committee members at Players, their input and assistance has been of huge benefit to this project.

Finally, I would like to thank my project supervisor, Dr. Aideen Keaney. Her guidance, advice and feedback were invaluable throughout the course of the project.
D.U. PLAYERS
Archives Database

March 2015

TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>NO.</th>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>INTRODUCTION AND SUMMARY</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Client Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Project Background</td>
<td>1</td>
</tr>
<tr>
<td>1.3</td>
<td>Objectives</td>
<td>2</td>
</tr>
<tr>
<td>1.4.</td>
<td>Terms of Reference</td>
<td>2</td>
</tr>
<tr>
<td>1.5.</td>
<td>Summary of remaining chapters</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>SYSTEM OVERVIEW</td>
<td>3</td>
</tr>
<tr>
<td>2.1.</td>
<td>System Objectives</td>
<td>3</td>
</tr>
<tr>
<td>2.2.</td>
<td>Technical Environment</td>
<td>3</td>
</tr>
<tr>
<td>2.3.</td>
<td>System Overview Diagram</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>DESCRIPTION OF WORK DONE</td>
<td>7</td>
</tr>
<tr>
<td>3.1.</td>
<td>Requirements Phase</td>
<td>7</td>
</tr>
<tr>
<td>3.2.</td>
<td>Design Phase</td>
<td>9</td>
</tr>
<tr>
<td>3.3.</td>
<td>Development Phase</td>
<td>16</td>
</tr>
<tr>
<td>3.4.</td>
<td>Testing</td>
<td>23</td>
</tr>
<tr>
<td>3.5</td>
<td>Difficulties Encountered</td>
<td>23</td>
</tr>
<tr>
<td>3.6.</td>
<td>Points Regarding Further Attention</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>CONCLUSIONS AND RECOMMENDATIONS</td>
<td>25</td>
</tr>
<tr>
<td>4.1.</td>
<td>Conclusions</td>
<td>25</td>
</tr>
<tr>
<td>4.2.</td>
<td>Recommendations</td>
<td>25</td>
</tr>
</tbody>
</table>
## APPENDICES

<table>
<thead>
<tr>
<th>NO.</th>
<th>CONTENT</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>Project Outline</td>
<td>A.1</td>
</tr>
<tr>
<td>B.</td>
<td>Interim Report</td>
<td>B.1</td>
</tr>
<tr>
<td>C.</td>
<td>Technical Manual</td>
<td>C.1</td>
</tr>
<tr>
<td>D.</td>
<td>Design Documentation</td>
<td>D.1</td>
</tr>
<tr>
<td>E.</td>
<td>Screenshots</td>
<td>E.1</td>
</tr>
<tr>
<td>F.</td>
<td>Sample Source Code</td>
<td>F.1</td>
</tr>
</tbody>
</table>

References
1. **INTRODUCTION AND SUMMARY**

This chapter describes the client, the project background, the project objectives, and the terms of reference. This chapter will also outline the remaining chapters of the report.

1.1. **Client Background**

Dublin University Players (hereafter referred to as Players) is the drama society within Trinity College Dublin. It is responsible for a student run theatre which stages two separate productions each day; a lunchtime show and an evening show. Weekly social events are also held for society members. Currently, there are approximately 2,500 members, however this number changes annually. Players often launch and coordinate festivals, both in the summer and in term time. These festivals vary in theme and form, for example the Dublin Shakespeare Festival, or the Trinity Fringe Festival. Players also participate in the Irish Student Drama Association (ISDA) awards, where productions from throughout year are recreated to compete in a national intervarsity competition.

Players organise a large amount of events, productions and members, but currently there is no digital system to catalogue all these events and their participants. There is currently a website for Players (www.duplayers.com). The website has basic contact information for the society and basic information about previous productions. However, many productions are missing from the website and what information is posted about productions is limited.

Players is run by a committee of 12 students. This committee changes annually in February.

1.2. **Project Background**

Players has a “Front of House, Office and Archive Manager”, who is obliged to keep archives of the events of Players. However, the current method of archiving does not incorporate the full scope of information available. Currently, the archive consists only of production posters, with no information catalogued other than the production name and the director. This information is saved in a separate hard copy folder for each year, which is stored in the Players office. There are no digital copies saved. This method of archiving was introduced three years ago. Before that, some material was saved, but there was no structure in place, therefore many articles are missing.

There is also archive material stored in the Trinity College Dublin Archives, within the Trinity Library. This material consists of articles from 1932 (the founding of Players) up to the 1990s. However, the material saved is inconsistent and sporadic; there is no structure to what has been catalogued.

Players believes that this paper based system is outdated, and does not adequately catalogue the society. Players would like a digital system to archive all events, productions and members. This system should be easily accessible to the public and to Players members. The information available to the public and members will be events, productions and festivals. The committee will only be able to access information regarding members and
past committees. It should be easily managed by the committee. This digital system will allow a clear cataloguing structure for all future archiving of the society. This will allow the society to have more information about its past.

1.3. Objectives

The objectives of the system are to create a clear, easy to use database that is publicly available, while allowing specific committee members access to add and alter information. In order to allow public access, the database will be delivered in the form of a web application. The web application will have a separate area for committee members, which will allow them to add, delete and alter information.

A further aim of this database is to allow Players to query the database, for example, if they wish to evaluate who were the most active society members within specific years. This will give Players a greater understanding of their member’s activity and subsequently give them information about alumni. This alumni information will be invaluable in the future, as it will allow Players to remain in contact with alumni.

1.4. Terms of Reference

- To develop and design a prototype web application that provides:
  - A database which will store all members, shows, events and archive material.
  - Box office functionality that is specific to the needs of D.U. Players
  - To create separate areas for committee members and visitors within the web application.
- Research and compare box office system alternatives.
- Research and compare archive system alternatives.

1.5. Summary of Remaining Chapters

This report will outline the design and development of the database and web application. It will also explore possible extensions that could be applied to the proposed system. Below are the summaries of the remaining chapters:

Chapter 2: Presents an overview of the current system, including system objectives, the technical environment and a system overview diagram.

Chapter 3: Contains a description of the work done over the course of the project.

Chapter 4: Outlines the conclusions and recommendations of the project.
2. SYSTEM OVERVIEW

This chapter provides an overview of the objectives of the system, the software used to design and implement the system and contains diagrams of the proposed system.

2.1. System Objectives

The purpose of this system, as outlined in the terms of reference, is to create a prototype information system, which will catalogue all members, events, festivals, productions, committees and awards. The system will provide a clear and simple user interface to display, edit, delete and add catalogue information. Currently, none of this information is saved digitally. Each production has a poster, the production’s name and director’s name stored in a hard copy folder. That is the extent of the archiving system at present.

The new system to be created will enable the client to catalogue all of Players activities. It will allow the public to access information on festivals, productions, events and awards. It will have a password and username protected committee area. Within the password protected area, committee members will be able to access all information on members and committees. Committee members will be able to add new records to the database through forms. There will also be the ability to import member data from a CSV file. This is because Players signs up approximately 2,000 new members each year, so entering in a new member one by one is implausible. Committee members will also be able to edit and delete records. Certain reports will be incorporated into the system, such as who the most active members of the society are. These reports will be created by querying the database.

The system will be created with a PHP and MySQL database back-end and a web interface HTML front end. The technical environment of the system will be discussed further in the next section.

2.2. Technical Environment

This section outlines the technical environment the system was developed in. The system is made up of a technical side composing a web front end and a website back end. The majority of the system is made up of a series of PHP and HTML pages, which make up the user interface. The back end is comprised of PHP and a MySQL database. The application runs locally on a free and open source cross platform web server solution called MAMP. MAMP consists of Mac OS X, Apache, MySQL and PHP. It is ideal for building web applications, such as this one. A diagram of the components within a MAMP server can be seen in Figure 2.2.1.
Mac OS X is a popular operating system.

Apache (Apache HTTP server) is a web server.

MySQL (My Structured Query Language) is a relational database management system. It is most commonly used in small to medium scale single-server deployments, making it ideal for this system.

PHP (Hypertext Pre-Processor) is a server-side website development language. It is a simple programming language for building dynamic web pages.

It was decided to develop the system in the MAMP environment because its components are open source. Open source software is available for anyone to download and have free access to the software’s source code (Lakhani, Von Hippel, 2003). MAMP is specifically for the Mac OS X operating system, which is the operating system used to create this system. However, there are also similar solutions available for the Windows operating system, XAMPP, and the Linux operating system, LAMP. The system can be run using these packages as well, which allows the system to be used on any major operating system. This is another reason for using the MAMP environment.

A variety of languages were used to create this system, PHP, MySQL, HTML, CSS and JQuery. PHP was the main language used for development. PHP is a server-side language. All the results of PHP processing is generated at the server and then sent to the browser, without the PHP code itself (Welling, Thompson, 2008). This means that PHP code can not be seen or accessed by the end user. If a PHP page’s source code is inspected only the HTML output would be shown. PHP’s main use in this system is its ability to connect to a database through MySQL. Using PHP, the end user can view and edit the contents of the database at the web user interface.

MySQL is a relational database system that is used to store information. SQL (Structured Query Language) is used for managing the data held in a MySQL server (Welling, Thompson, 2008). SQL statements are used to perform tasks such as insert, delete and update on databases. SQL statements are also used to query the database and subsequently create reports.
Dreamweaver CS5 was used to write the HTML, PHP, CSS and JQuery code and it was run on a Mac OS X laptop. MySQL Workbench was used to test queries and create the MySQL database.

2.3. **System Overview Diagram**

Figure 2.3.1. displays the System Overview Diagram of the system. This diagram describes the layout of the system and maps how the user will use it. For each of the entities within Players (events, festivals, members, productions, committees and awards), there are functions to view, edit, insert, delete and search records. Each member, production, committee and festival record has its own individual record page, with further details on the record. It is also possible to import member data from a CSV file.

Parts of the system are password protected. Only committee members will be given usernames and passwords. Figure 2.3.1. displays the separation between public access and committee access. Once a committee member is logged into the system they remain logged in until they log out.
FIGURE 2.3.1. - System Overview Diagram
3. DESCRIPTION OF WORK DONE

This chapter contains a description of the work done throughout the course of the project. The system was developed using an incremental model. A diagram of the processes of this model can be seen in Figure 3.0.1. The incremental model is an iterative process, which means some stages were repeated as modifications were made to the system. It consists of several stages: requirement, design, development, testing and implementation (Galvan-Lopez, 2013).

![Incremental Design Methodology](image)

**FIGURE 3.0.1. - Incremental Design Methodology**

3.1 Requirements Phase

This first phase was critical to the design process as it was where the needs of the client were clearly defined. There were many client meetings, which concluded with the terms of reference, (see Section 1.4). Following the definition of the terms of reference, other committee members were met with, such as the treasurer and the front of house, office and archive manager. As the organisation is a student society, people are elected to each position yearly. Therefore, past committee members were also contacted and met with to gain a better perspective of the requirements. Currently there is minimal information catalogued by the organisation. The information saved consists of a poster of each production accompanied with the name and the director of the production. This information is not saved digitally.

The client’s main requirement was to create a prototype information system to catalogue all members, events, festivals, committees, awards and productions. The system was to be in part accessible to the public, providing them with information on productions, events, awards and festivals. There would be a separate committee area to access information on members and committees. Within the committee area, users would also be able to add, edit and delete
information. It was also required that information on members could be uploaded from a spreadsheet, as well as through a form. This was because approximately 2,000 new members join Players every year, so adding every new member in through a form would be impractical.

When initially deciding upon the terms of reference, the possibility of integrating a box office system into the archiving system was discussed. It was initially thought that this would be possible, as it was believed that the box office requirements of the client were minimal. However, upon further discussing the box office integration with the client it was discovered that the box office requirements were more complicated than initially thought. The requirements include:

- Creating tickets that are compatible with current thermal ticket printer.
- Ability to manage multiple shows at once.
- Ability to set multiple different prices.
- Save all transaction data.
- Allow for cash only transactions.

It would not have been possible to integrate a box office system that met these requirements into the archive system created for Players. Because of this, a variety of different, off-the-shelf, options were looked into. Table 3.1.1. shows the researched box office systems, and whether they met the requirements.

<table>
<thead>
<tr>
<th>TABLE 3.1.1. – Comparison of Box Office Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Little Box Office</strong></td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>Vendini</td>
</tr>
<tr>
<td>Savoy Systems</td>
</tr>
<tr>
<td>Little Box Office</td>
</tr>
</tbody>
</table>

Little Box Office met some of the requirements, however it was not compatible with the current printer and only allowed for one show to be sold at a time. This was not appropriate for Players as Players has two shows each day. This means they require the ability to sell more than one type of ticket at once. Using Little Box Office, it was also necessary to print tickets out using a A4 paper printer. This was not appropriate for Players as they required tickets to be printed out on their thermal printer. Little Box Office charges an annual fee of £200. Although it is the cheapest of the box office systems, because it did not meet all of the requirements it was not appropriate for the client.

Vendini met all of the requirements. Vendini does not have an annual fee, although it does have ongoing fees. For each ticket sold, a certain percentage is paid to Vending. As it met
all the requirements, Vendini was presented to the client. However the client was not comfortable with ongoing-transaction based fees. The client preferred an annual fixed fee.

Spektrix also met all the requirements. Similarly to Vendini, however, the pricing model was not appropriate for the client. Spektrix charges an annual fee, but the lowest pay tier began at £10,000. This price range was far above the client's means.

Savoy Systems is the system currently used by the client. However, the way the system is currently used, it does not incorporate all of its available features, for example the ability to create different pricing options for each show. Including the additional features, all the requirements set out by the client. It charges an annual fee of £350. As this system meets the requirements, and the client is comfortable with the price of the system, it has been suggested to the client to continue using Savoy Systems.

The terms of reference also specify that off-the-shelf archiving systems should be reviewed. However, no archiving systems could be found, therefore no review occurred. As there were no archiving systems found it was necessary to develop an archiving system from the ground up.

3.2. Design Phase

The next stage in the process was the design phase. The two areas of the system that required design were the user interface and the database.

Database Design

The database was designed to be in the third normal form (3NF). For a database to be in the 3NF, non-primary key data must be “mutually independent” and “irreducibly dependent on the primary key” (Date, 2003: 358). This allows the database to be efficiently organised by reducing redundant data and having logical data dependencies.

To begin designing the database, all the entities required in the database were noted. The entities were productions, members, committees, awards, festivals and events. These entities became the main tables within the database. Next, the data fields saved in each table were decided, for example, within the event table the following data fields were saved: the event name, the date, the week, the year, the blurb for the event and the entertainment officer that organised the event.

This was problematic for two tables; the committee table and the shows table. A committee within Players does not always follow the same structure, it can have anywhere between two and fourteen members, and each member may have a unique title. Similarly, a show does not have a set amount of cast or crew, and each member of the production will have a unique role. To circumvent this problem bridge tables were created.

Bridge tables are tables that allow many-to-many relationships within a database. Relational Database Management Systems (RDBMS) do not allow for many to many relationships
between two tables. Therefore, bridge tables must be used to create many to many relationships between data. This is done by having two tables, each with a one to many relationship with the bridge table. The bridge table, therefore, will have two foreign keys. An example of this can be seen in Figure 3.2.1.

Once the tables and the data within the tables were decided upon, each table (except for the bridge tables) were allocated a primary key. A primary key is a unique key belonging to a table. For all relevant tables the primary key was an ID number. Foreign keys were also decided upon. A foreign key is used to indicate that data within one table relates to the primary key of another table (Date, 2003). For example, in this project, the entertainment officer, within the events table, was a foreign key that pointed to the primary key of member ID, within the members table.

After the primary and foreign keys were set, the relationships between the tables were defined. One-to-one, one-to-many, and many-to-many relationships were used, as mentioned previously. Subsequently, an entity relationship (ER) diagram was created. ER diagrams show the tables and the relationships within a database. These diagrams were useful as they gave a clear picture of the interactions between the various tables. The ER diagram of the final database can be seen in Appendix D, Figure D.2.1.

![FIGURE 3.2.1. – Example of Bridge Table Relationships](image-url)
User Interface Design:

The user interface was designed for ease of use. As the committee changes yearly, it was necessary to design a user interface that was intuitive and required no training. The client made it clear that a clear display of information was the most important aspect of the user interface. A website template and general layout was agreed upon with the client. An example of one of the web pages can be seen in Figure 3.2.2. The system overview diagram (see Section 2.3, page 6) shows the layout of the system.

One of the requirements of the system was that certain sections were open to the public and certain sections were only open to committee members. This was done by adding a log-in for committee members. Each committee member has their own username and password. A committee member can log in at any time, as there is link to the log-in page in the footer of every page. When a member of the public attempts to access a committee-only area, they are asked to log in or return to the previous page.
Another requirement of the system was the clear display of information. For each of the entities that are catalogued (members, festivals, awards, productions, committees, and events) there is a separate report, which shows all of the records of that entity. These pages are created by querying the database. For example, Figure 3.2.3. shows the “all members” web page.

FIGURE 3.2.3. – Screenshot of All Members Page
In order for the website to be easily navigated three features were added:

- Having menu and submenu bars which appeared on every page. Entity menu bars, which are specific to the entity that is being viewed were also added. This allows the user to easily move throughout the website. These can be seen in Figure 3.2.4.

![Figure 3.2.4. – Screenshot and Explanation of Menu Bars](image)

- Having a search bar for each entity. The user is able to search for specific records of festivals, events, members, productions, committees and awards. An example of one of these search bars can be seen in Figure 3.2.5.

![Figure 3.2.5. – Screenshot of Event Search Bar](image)

- Within the pages that show all records of members, productions, committees and festivals, having a link for each record that leads to a page with further information on the record. An example of the links can be seen in Figure 3.2.6 and an example of an individual record page can be seen in Figure 3.2.7.
FIGURE 3.2.6. – Screenshot of All Members page
Dreamweaver was used to create the template page of the website, through HTML. CSS (Cascading Style Sheets) was used to determine many of the visual aspects of the website, such as font and background colour. CSS works by creating rules that determine how the content of different elements within a page should appear (Duckett, 2010). One CSS file was created which is used for all of the web pages. This means changing the CSS file changes the look of all of the web pages. This makes it easy for the Players to change the look and feel of the entire site by just changing a single file.
3.3. **Development Phase**

Once the design phase was complete, development of both the website and the database could begin. This section outlines this process.

**Database Development**

Once the database was designed, it was created using MySQL Workbench. The database consists of eight tables, committees, shows, members, festivals, awards, events, and two bridge tables, committee roles and member roles. These tables can be seen in the Appendix. The three most important tables are the shows table, the members table and the committee table. They are the most important because all relationships are tracked back to one of their primary keys.

The bridge tables, committee roles and member roles, were created to link committees to members and shows to members, respectively. When these tables were created, additional data, such as the name of their role in the show or committee, was added. In the bridge tables, the foreign key relationships which referred to the member table were created with cascade delete. This meant that if a member was deleted from the database, so would the record of them being in any committee or show. This was done to reduce redundant data within the database.

After the database was created, records were created in Excel to fill the tables. This was done to test the display of the records in the web pages. The Excel files created were then converted into CSV files. VB code was used to do this to ensure the file’s format was correct for importing into the MySQL database.

**Web Interface Development**

The development of the web interface for the system was done using Dreamweaver CS5. Initially, only the “design” view of Dreamweaver was used, to create the template. In the “design” view of Dreamweaver, no code is seen, and the web page layout is created by the user dragging and dropping various objects onto the page, with Dreamweaver creating the HTML code behind it. However, when this template was tested in multiple browsers, it was found that the web page did not look as planned. Subsequently, all web pages were coded manually.

Once the template for the website was created, it was necessary to connect to the MySQL database. This was done using PHP. The connection was then tested, checking that information from the database could be displayed in the webpage, and that information entered into forms in the webpage would be entered into the database. After this connection was tested the development of the website could begin. Below is a list of features developed.
Displaying Data

As specified by the client, an important aspect of this system is the clear display of data. Reports were created to display all the records of the main entities, all festivals, all events, all members, all committees, all productions and all awards. As some entities, such as members, will grow to include over 10,000 records, the reports are designed to only display 100 records at a time. Individual pages were then developed for all records of festivals, members, committees and productions. These individual pages have further information included on them. For example, the individual member page has all the committees and shows they had been a part of, along with their accompanying roles. These individual pages can be accessed through the link provided on the webpage, which shows all of the records. An example of this can be seen in Figure 3.2.6 (page 15) and Figure 3.2.7 (page 16).

Adding Data

The next feature added was the ability to add data to the database. Forms were created for all tables. The forms were designed to minimise data entry errors. Drop down lists were created for many forms. These act as controls, so the user cannot input incorrect data. For example, in the add award form, in the award name field, only names of existing awards are available in the drop down list. This can be seen in Figure 3.3.1. Many of these drop down lists query the database for their options. For example, in the events form, when selecting the entertainment officer only members who have held the position of entertainment officer appear. This can be seen in Figure 3.3.2. Drop down lists that query the database are used for any field that relate to a foreign key, so that these relationships do not become compromised. This means that members must be added to the members table before they can be included in other records.
It was necessary to use JQuery to create two of the forms, the “add new committee” form and the “add new show” form. For example, when adding a new show to the system, it is unknown how many members are involved in a production. Therefore, it not possible to have a form with a fixed amount of fields. Users must be able to add more fields. This was done by adding a button which added one new field into the form every time it was clicked. JQuery was used to achieve this. It was not possible to use PHP, as PHP is a server side language. This means that information would have to be sent back to the server to add more fields. This would detract from the user’s experience. JQuery was, therefore used instead. JQuery is a Javascript library designed to make scripting easier. Javascript is a programming language that executes code on the client’s side (York, 2009). This means the code is executed in the browser, which means that when the user requires another text field, reloading the page is not necessary. This can be seen in Figure 3.3.3. and Figure 3.3.4.
Searching Data

The search function was then added. A separate search bar was created for each main entity, festivals, events, members, committees, productions and awards. The value inputted into the search bar is compared with all data within the specific entity and then results are returned in the same layout as the “all” page of that entity. For example, if “b” is inputted in the productions search bar, then it would return all production records which contained a “b” in any of the columns. This result can be seen in Figure 3.3.5. This feature allows the user to find specific information easily, and also allows the user to filter data viewed as they see fit, whether it is by year, term, etc.
Editing and Deleting Data

The ability to edit and delete data was then added to the system. Edit and delete functions are available for each main entity: festivals, events, members, committees, productions and awards. An example of how editing occurs can be seen in Figure 3.3.6. Deleting some records can cause records in other tables to be affected. This is because some relationships within the database were created with cascade delete. For example, if you delete a member from the database, all record of that member in shows or committees will also be deleted.
Creating Separate Areas

The website was then divided into the public and committee sections. The committee section requires a password and username that is saved within the username table of the database. Each committee member is provided with their own username and password. These passwords and usernames have no relationship with records in the member table. This is because the committee changes annually. Once logged in, the user is sent to the committee area (see Figure 3.3.7). There, the user has the ability to add, edit and view all data. They are also able to view and interact with the “Member Activity” report. This page is discussed in detail below. Users without usernames and passwords are able to view any pages that display festival, production, event and award data. They are not able to view member or committee data and they are not able to edit, add or delete any data. The system overview diagram (see Figure 2.3.1, Section 2.3, page 6) shows exactly which pages are accessible to the public and which pages are password protected committee areas.

FIGURE 3.3.7. – Screenshot of Committee Area
Importing from CSV

As described in the requirements phase, it was necessary to add in the ability to import data from a CSV file into the members table. This feature was added into the system next. When the CSV file is uploaded, the email address of each new record is checked against the email addresses of already existing member records. If the email address already exists, the record will not be added. This prevents duplicate data from entering the system.

Information on Activity of Members

One further feature was added to the system. This was a page, which displayed the most active members in the society. The activity of the members was calculated using a scoring system. The scoring system was created through consultation with the client. For each activity the member was involved in, they would get a certain amount of points. For each production they were involved with they would receive one point, for each committee they were on, they would receive three points, and if they were a chairperson of a committee they would receive four points. The members who received the most points were deemed to be the most “active” within the society. Filters were added to the page so that the activity of members in a specific year, or in a specific period of years could be displayed. An example of one of these reports can be seen in Figure 3.3.8.

![Most Active Members Report](image)

FIGURE 3.3.8. – Screenshot of Active Member Report
3.4. Testing

As the Incremental Model was used, the system was tested throughout the development. Testing occurred after each feature was completed to ensure the work done was robust. This involved entering trial data into the database, viewing and editing the data, and deleting trials, to ensure that each function was operating as expected. Once all development was complete a third party was asked to test the whole system to ensure that each feature was operating as expected.

During development and test, when errors occurred, debugging was necessary. Depending on the language, different debugging techniques were used. All JQuery code used was tested first using JSFiddle, which is an online code editor. All MySQL code was tested first in MySQL Workbench. Testing this code before it was implemented meant that it was far easier to pinpoint coding issues.

3.5. Difficulties Encountered

During the development of the prototype system some difficulties were encountered.

The first difficulty encountered was Dreamweaver’s “Design View”. When creating a website in Dreamweaver, you are able to view how the page will appear in a browser. The first few pages made were created in this mode exclusively. However, when the pages were later tested in browsers, they did not appear in the same manner as that displayed in Dreamweaver. This resulted in all subsequent pages being coded manually. This slowed down the process as the languages used were new and unfamiliar.

The second difficulty encountered was creating the upload CSV feature. Within the system, there is a feature to upload member data into the database from a CSV file. This was difficult to achieve as it was necessary to ensure that no duplicate records could be uploaded into the database. However, it was possible once the database was changed slightly. The email address of each member now acts as a unique key. This means that it is not possible to have more than one record with the same email address, within the member’s table.
3.6. Points Regarding Further Attention

The next step to fully implement the system is to make the website available on the internet. There are many web hosting companies that support the technologies used in this system. A web hosting service would allow Players to make the website accessible to the World Wide Web.

The Web Hosting Companies reviewed were the following:

- Hosting Ireland
- GoDaddy
- Blacknight Solutions
- Amazon Web Services

Hosting Ireland has provided domain name registration and hosting services. For €39.95 per year the client will get 10GB disk space, 100GB of bandwidth and a MySQL database.

GoDaddy is an American company that offers a variety of hosting plans and domain name registration. For €6.14 a month, €73.68 annually, the package comes with unlimited bandwidth, 100GB Disk Space and 1 GB of database storage. This package is more than enough for the system created.

Blacknight Solutions is an Irish based web hosting company. It provides hosting plans and domain name registration. An annual fee of €49.95 provides the client with 10GB disk space, 200GB monthly transfer, and a MySQL database. This package is suitable for a system of this scale.

Amazon Web Services provides hosting services only. It has a free tier that offers 15GB of bandwidth, 5GB disk space and 20GB database storage. This tier is free for the first 12 months.

It is suggested that the client uses Amazon Web Services for hosting and one of the other company for domain name registration. This will allow Players to test the system online free of charge.

Once the website is connected to the internet, Players should begin using the system. It is suggested that Players sets up times throughout the year where the system will be updated. It is suggested that new productions and events are updated weekly (as there is a new event and two new shows every week). Festivals and committees should be added to the system whenever new festivals or committees occur within Players. Members should be added to the system annually, after Fresher’s week (the time where the majority of members sign up). Awards should also be added to the system annually, after the Irish Student Drama Association awards. It is also suggested that the Front of House, Office and Archives manager begins entering any records they can from the archives.
4. CONCLUSIONS AND RECOMMENDATIONS

4.1. Conclusions

This project involved designing and developing a prototype information system for D.U. Players to catalogue all of its activities; events, members, festivals, productions, awards and committees. The finished system met all the client’s requirements. The current archiving system is minimal, and not digital, so this new system will be invaluable.

The system is intuitive and simple to use. After demonstrating the system to the client, the need for a user manual was discussed. The client expressed that no user manual would be necessary. However, a technical manual has been created.

The final system was developed using five programming languages; PHP, CSS, JQuery, MySQL and HTML. The system has been tested extensively to ensure that users can not make mistakes when using the system. However, the system has only been tested in a development environment so, if the system is connected to the internet, it is recommended that the client thoroughly tests the system before allowing access to the public. This would include having multiple users on the system at a time.

4.2. Recommendations

The system was developed in a short amount of time, as such there are additional features that could be added to the system, such as the one discussed below. There are also many actions to correctly implement the system within the organisation.

An additional feature that could be added to the system are allowing pdf uploads into the database. This would allow posters of events, festivals, and productions to be catalogued. It would also allow scripts from new writing productions to be archived.

As mentioned previously (see Section 3.5), this system is a prototype and has not been made available on the internet. The next step in implementing this system is putting the user interface online, using one of the hosting services mentioned in Section 3.5. Once the system has been put online, it is suggested that it is tested thoroughly before allowing access to the public. Following security measures should also be looked into, such as encryption, so that all information sent over the internet, such as passwords, is safe. Additionally, a back-up system should be created so that in the event of a crash or a fault occurring, all information would be saved.

Ideally, hands on training would occur with all committee members. This would allow them to ask questions and iron out any issues they may have with the user interface. Further training would be specifically given to the front of house, office and archive manager. It would be necessary to ensure that they understand the system completely, as it would specifically be their responsibility to maintain the records stored in the system.
A PROJECT OUTLINE

Client: Dublin University Players
Project: D.U. Players archives database
Location: D.U. Players, Samuel Beckett Centre, Trinity College Dublin
Client Contact: Donal McKeating mckeatd@tcd.ie

Client Background

D.U. Players is the drama society within Trinity College. It is responsible for a completely student run theatre which stages a lunchtime show and an evening show every day during term time. D.U. Players also hosts weekly social events for society members and has approximately 2500 members every year.

Project Background

D.U. Players has existed for 80 years and as such there is an extensive amount of archive material. Some of this archive material is kept within the Trinity Library, the rest of the archive material is kept on the D.U. Players premises. This material is not properly organised and has not been digitised.

D.U. Players use a box office system, which acts as a point-of-sale for tickets. The system is quite outdated, complicated to use and only has a limited amount of features.

Client Requirement

The construction of a prototype of a web based application which will act as both a database for archive material and a box office system. The application will have a box office system which allows for each show to have general information about the show stored within a database while also keeping track of ticket sales for each show. The application will allow for various analytics features, such as box office reports and financial reports. The application will also allow old show archive material to be stored within the database.

What is involved for the student?

There will be research into what systems are already available on the market. The application itself will consist of a front-end user interface and a back-end database that stores the required information. A combination of programming languages will most likely be used to create the application, including HTML, PHP, CSS, Javascript and MySQL.
B INTERIM REPORT

Project: Design and development of web application for D.U. Players
Client: D.U. Players
Student: Deirdre Van Wolvelaere
Supervisor: Aideen Keaney

Review of Background and Work to Date

D.U. Players is the drama society within Trinity College. It is responsible for a student run theatre which stages a lunchtime show and an evening show every day during term time. D.U. Players also hosts weekly social events for society members and has approximately 2500 members every year. At present there is no system in place to store and archive all this information. D.U. Players also pays an annual fee for a complex box office system that is difficult to use.

To date, I have met with a variety of people to discuss the limitations of the current system, and to discuss implementation of a new system. I have also begun investigating box office systems that are available off the shelf.

Terms of Reference

• To develop and design a prototype web application that provides:
  o A database which will store all members, shows, events and archive material.
  o Box office functionality that is specific to the needs of D.U. Players
  o To create separate areas for committee members and visitors within the web application.
• Research and compare box office system alternatives.
• Research and compare archiving systems alternatives.

Further Work

Until Christmas break, I will continue researching box office alternatives, I will begin researching archiving system alternatives and will begin designing the back-end database. During Christmas break, I will finish the design of the application. At the beginning of Hilary term I will begin work on building the web application using a combination of programming languages, such as HTML, PHP, CSS, Javascript and MySQL.

Conclusions

From meetings with the client so far I have found that there are many improvements that can be made to the way D.U. Players operates. These improvements can be made through digitising their archiving system. As D.U. Players is such a unique theatre and society, in the way it operates, there is nothing currently available off the shelf to cater to its needs. The web application I will build will allow D.U. Players to archive all its activities while providing a box office, which is a daily necessity within D.U. Players.
The Technical Manual for this system is attached as a separate document. It gives instructions on how to navigate the user interface and describes how to set up the system for initial deployment.
D DESIGN DOCUMENTATION

D.1. Design Methodology

This system was created using the incremental method. Software methodologies provide structure during the development process. Five fundamental phases were involved in the systems development life cycle of the project:

- Requirements Analysis
- Design
- Development
- Implementation
- Testing

As the system progressed, adjustments were made, repeating the design and development stages in the next iteration.

![Incremental Model Diagram](image)

Figure D.1.1. – Incremental Model Diagram

The incremental model was chosen as it allows the system to be tested while features are still being developed. This means that if features require changes, it is possible to return to the design phase.
## D.2. Database Tables

### TABLE D.2.1 – Design for Committees Table

<table>
<thead>
<tr>
<th>Table Name:</th>
<th>Committees</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Description:</td>
<td>Contains information on committees</td>
</tr>
</tbody>
</table>

### Related Tables

<table>
<thead>
<tr>
<th>Table Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Festivals</td>
</tr>
<tr>
<td>Committee Members</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommitteeID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required</td>
<td>Festivals.FestivalID, CommitteeMembers.CMCommitteeID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Primary Key</td>
<td>Unique, Auto-Incrementing</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>VARCHAR(45)</td>
<td>Name of Committee</td>
<td>Must be less than 45 characters</td>
<td></td>
</tr>
<tr>
<td>Year</td>
<td>YEAR(4)</td>
<td>Year of Committee</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table Name: Committee Members

### Brief Description:
Bridge table between Committees and Members. Also stores what role the member had on a committee.

### Related Tables:
- Members
- Committees

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMMemberID</td>
<td>INT(11)</td>
<td>Member ID</td>
<td>Required</td>
<td>Members.MemberID</td>
</tr>
<tr>
<td>CMCommitteeID</td>
<td>INT(11)</td>
<td>Committee ID</td>
<td>Required</td>
<td>Committees.CommitteeID</td>
</tr>
<tr>
<td>CommitteeRole</td>
<td>VARCHAR(45)</td>
<td>Year of Committee</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>Chairperson</td>
<td>INT(1)</td>
<td>If 1 – Member was Chairperson of Committee</td>
<td>Required</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE D.2.3 – Design for Events Table

<table>
<thead>
<tr>
<th>Table Name:</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Description:</td>
<td>Information on Events</td>
</tr>
<tr>
<td>Related Tables</td>
<td>Members</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
<tbody>
<tr>
<td>EventID</td>
<td>INT(11)</td>
<td>ID Number Primary Key</td>
<td>Required Unique Auto-Increment</td>
<td></td>
</tr>
<tr>
<td>EventName</td>
<td>VARCHAR(45)</td>
<td>Event Name</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>EventYear</td>
<td>YEAR(4)</td>
<td>Event Year</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>EventWeek</td>
<td>INT(11)</td>
<td>Event Week</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EventDate</td>
<td>DATE</td>
<td>Event Date</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EventBlurb</td>
<td>VARCHAR(500)</td>
<td>Blurb about event</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EntsOfficer</td>
<td>INT(11)</td>
<td>ID Number of Entertainment Officer Foreign Key</td>
<td>Members.MemberID</td>
<td></td>
</tr>
</tbody>
</table>
TABLE D.2.4 – Design for Festivals Table

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
<tbody>
<tr>
<td>FestivalID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>FestName</td>
<td>VARCHAR(45)</td>
<td>Festival Name</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>FestYear</td>
<td>YEAR(4)</td>
<td>Festival Year</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>FestDate</td>
<td>VARCHAR(45)</td>
<td>Festival Dates</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FestBlurb</td>
<td>VARCHAR(500)</td>
<td>Blurb about Festival</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FestCommitteeID</td>
<td>INT(11)</td>
<td>ID Number of Committee</td>
<td>Foreign Key</td>
<td>Committees.CommitteeID</td>
</tr>
<tr>
<td>Data Name</td>
<td>Data Type</td>
<td>Description</td>
<td>Validation Rule</td>
<td>Linked To</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------------------------------</td>
<td>----------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>AwardID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required, Unique, Auto-Increment</td>
<td></td>
</tr>
<tr>
<td>AwardName</td>
<td>VARCHAR(45)</td>
<td>Award Name</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>AwardYear</td>
<td>YEAR(4)</td>
<td>Award Year</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>Host</td>
<td>VARCHAR(45)</td>
<td>Host of ISDA Awards</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISDAShowID</td>
<td>INT(11)</td>
<td>ID Number of Show</td>
<td>Required</td>
<td>Shows.ShowID</td>
</tr>
<tr>
<td>ISDAMemberID</td>
<td>INT(11)</td>
<td>ID Number of Member</td>
<td>Required</td>
<td>Members.MemberID</td>
</tr>
<tr>
<td>NomWin</td>
<td>VARCHAR(45)</td>
<td>Whether award was a nomination or a win</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TABLE D.2.6 – Design for Shows Table

**APPLICATION DESIGN WORKSHEET TABLES**

<table>
<thead>
<tr>
<th>Table Name:</th>
<th>Shows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Description:</td>
<td>Information on Shows</td>
</tr>
</tbody>
</table>
| Related Tables | ISDAAwards  
MemberRoles |

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
</table>
| ShowID    | INT(11)   | ID Number   | Required, Unique, Auto-Increment | ISDAAwards.ISDAShowID  
MemberRoles.MRShowID |
| ShowName  | VARCHA R(45) | Show Name   | Required        |           |
| ShowYear  | YEAR(4)   | Show Year   |                 |           |
| ShowDate  | VARCHA R(45) | Show Dates  |                 |           |
| ShowWeek  | INT(11)   | Show Week   |                 |           |
| ShowTerm  | VARCHA R(45) | Show Term   |                 |           |
| Location  | VARCHA R(45) | Location   | Default = ‘Players Theatre’ |           |
| Playwright | VARCHA R(45) | Playwright  |                 |           |
| Show Type | VARCHA R(45) | Type of show – Lunch, evening, festival, etc. | | |
| ISDA      | TINYINT(1) | If 1 – Show was an ISDA production | Default = ‘0’ | |
# TABLE D.2.7 – Design for Members Table

## APPLICATION DESIGN WORKSHEET TABLES

<table>
<thead>
<tr>
<th>Table Name:</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Description:</td>
<td>Information on Members</td>
</tr>
<tr>
<td>Related Tables</td>
<td>CommitteeMembers, MemberRoles, ISDAAwards, Events</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Data Type</th>
<th>Description</th>
<th>Validation Rule</th>
<th>Linked To</th>
</tr>
</thead>
<tbody>
<tr>
<td>MemberID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required, Unique, Auto-Increment</td>
<td>ISDAAwards.ISDAMemberID, MemberRoles.MRMemberID, CommitteeMembers.CMMemberID, Events.EntsOfficer</td>
</tr>
<tr>
<td>FirstName</td>
<td>VARCHAR(45)</td>
<td>First Name</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>SecondName</td>
<td>VARCHAR(45)</td>
<td>Second Name</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>MemberYear</td>
<td>YEAR(4)</td>
<td>Year Member Joined</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Email</td>
<td>VARCHAR(45)</td>
<td>Email</td>
<td>Unique, Required</td>
<td></td>
</tr>
<tr>
<td>Course</td>
<td>VARCHAR(45)</td>
<td>College Course</td>
<td>Default = 'Drama'</td>
<td></td>
</tr>
<tr>
<td>Graduated</td>
<td>VARCHAR(45)</td>
<td>Whether member has graduated or not</td>
<td>Default = 'No'</td>
<td></td>
</tr>
<tr>
<td>Lifetime</td>
<td>VARCHAR(45)</td>
<td>Whether member has received lifetime membership or not</td>
<td>Default = 'No'</td>
<td></td>
</tr>
</tbody>
</table>
TABLE D.2.8 – Design for Member Roles Table

<table>
<thead>
<tr>
<th><strong>APPLICATION DESIGN WORKSHEET TABLES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table Name:</strong></td>
</tr>
<tr>
<td><strong>Brief Description:</strong></td>
</tr>
</tbody>
</table>
| **Related Tables** | Members  
| | Shows |

<table>
<thead>
<tr>
<th><strong>Data Name</strong></th>
<th><strong>Data Type</strong></th>
<th><strong>Description</strong></th>
<th><strong>Validation Rule</strong></th>
<th><strong>Linked To</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>MRMemberID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required</td>
<td>Members.MemberID</td>
</tr>
<tr>
<td>MRShowID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required</td>
<td>Shows.ShowID</td>
</tr>
<tr>
<td>MemberRole</td>
<td>VARCHAR(45)</td>
<td>Role in Show</td>
<td>Required</td>
<td></td>
</tr>
</tbody>
</table>
| Acting        | TINYINT(1)   | If 0 – Production Role  
If 1 – Acting Role | Required            |               |

TABLE D.2.8 – Design for Users Table

<table>
<thead>
<tr>
<th><strong>APPLICATION DESIGN WORKSHEET TABLES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table Name:</strong></td>
</tr>
<tr>
<td><strong>Brief Description:</strong></td>
</tr>
<tr>
<td><strong>Related Tables</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Data Name</strong></th>
<th><strong>Data Type</strong></th>
<th><strong>Description</strong></th>
<th><strong>Validation Rule</strong></th>
<th><strong>Linked To</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>UserID</td>
<td>INT(11)</td>
<td>ID Number</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>Username</td>
<td>VARCHAR(45)</td>
<td>Username</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td>VARCHAR(45)</td>
<td>Password</td>
<td>Required</td>
<td></td>
</tr>
</tbody>
</table>
D.2. Entity Relationship Diagram

FIGURE D.2.1. – Database Entity Relationship Diagram
FIGURE E.1 – All Members Page

<table>
<thead>
<tr>
<th>Name</th>
<th>Year Joined</th>
<th>Email</th>
<th>Course</th>
<th>Graduated</th>
<th>Lifetime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adam Theakre</td>
<td>2012</td>
<td><a href="mailto:vvv@gmail.com">vvv@gmail.com</a></td>
<td>Science</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Alison Hennessy</td>
<td>2013</td>
<td><a href="mailto:hnh@gmail.com">hnh@gmail.com</a></td>
<td>Nanoscience</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Aoife Leonard</td>
<td>2012</td>
<td><a href="mailto:hii@gmail.com">hii@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Aoife Meaney</td>
<td>2012</td>
<td><a href="mailto:lli@gmail.com">lli@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Alan O’Grady</td>
<td>2014</td>
<td><a href="mailto:jjj@gmail.com">jjj@gmail.com</a></td>
<td>Music</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Ben Ferris</td>
<td>2013</td>
<td><a href="mailto:vjv@gmail.com">vjv@gmail.com</a></td>
<td>History</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Benedict Fadale</td>
<td>2014</td>
<td><a href="mailto:faa@gmail.com">faa@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Brian McMahon-Gallagher</td>
<td>2013</td>
<td><a href="mailto:sas@gmail.com">sas@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Brian Fallon</td>
<td>2012</td>
<td><a href="mailto:uu@gmail.com">uu@gmail.com</a></td>
<td>Engineering</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Brian Donnelly</td>
<td>2012</td>
<td><a href="mailto:lli@gmail.com">lli@gmail.com</a></td>
<td>European Studies</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Cameron Naicaly</td>
<td>2009</td>
<td><a href="mailto:rrr@gmail.com">rrr@gmail.com</a></td>
<td>Music</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Charli Mathews</td>
<td>2011</td>
<td><a href="mailto:asdf@gmail.com">asdf@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Clare Fleming</td>
<td>2011</td>
<td><a href="mailto:dfds@gmail.com">dfds@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Claire O'Reilly</td>
<td>2011</td>
<td><a href="mailto:fji@gmail.com">fji@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Dara C'Ocraire</td>
<td>2014</td>
<td><a href="mailto:uu@gmail.com">uu@gmail.com</a></td>
<td>MSiSS</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>David Roan-Nolan</td>
<td>2014</td>
<td><a href="mailto:jjj@gmail.com">jjj@gmail.com</a></td>
<td>History</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deirdre Van Wijvelaere</td>
<td>2011</td>
<td><a href="mailto:Dxx44@gmail.com">Dxx44@gmail.com</a></td>
<td>MSiSS</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Donal McGinty</td>
<td>2011</td>
<td><a href="mailto:sjj@gmail.com">sjj@gmail.com</a></td>
<td>Economics</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Ellen Kirk</td>
<td>2012</td>
<td><a href="mailto:zll@gmail.com">zll@gmail.com</a></td>
<td>Russian</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Eileen Gorman</td>
<td>2012</td>
<td><a href="mailto:wti@gmail.com">wti@gmail.com</a></td>
<td>Drama</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
FIGURE E.2 – Individual Members Page
FIGURE E.3 – Add Member Page

FIGURE E.4 – All Festivals Page
FIGURE E.5 – Edit Member Page
FIGURE E.6 – Individual Festival Page

- **ISDA Festival**
  - **Year:** 2015
  - **Dates:** 08.04.2015 - 15.04.2015
  - **Committee:** ISDA
  - **Information:** D.U. Players hosted the Irish Student Drama A

**Committee Roles**

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Michael Browne</td>
<td>Chairperson</td>
</tr>
<tr>
<td>Brian Fagan</td>
<td>Festival Coordinator</td>
</tr>
<tr>
<td>Trish Taylor</td>
<td>Productions Officer</td>
</tr>
<tr>
<td>Cara O’Carrol</td>
<td>Technical Manager</td>
</tr>
<tr>
<td>Ben Kelli</td>
<td>Publicity and Communications Officer</td>
</tr>
<tr>
<td>Eoin Xerrien</td>
<td>University Liaison</td>
</tr>
<tr>
<td>Jade Conn</td>
<td>Fringe Co-ordinator</td>
</tr>
<tr>
<td>Mary Lamond</td>
<td>Secretary</td>
</tr>
<tr>
<td>Marie Walsh</td>
<td>Treasurer</td>
</tr>
<tr>
<td>Elaine O’Byrne</td>
<td>Volunteer Co-ordinator</td>
</tr>
<tr>
<td>Matt Annidge</td>
<td>Entertainment Officer</td>
</tr>
</tbody>
</table>

FIGURE E.7 – Add Festival Page

**Add a New Festival:**

* Indicates required field

- **Festival Name:**
- **Year:**
- **Dates:**
- **Sub:**
- **Festival Committee:**

Add Festival: "Search Festivals..."
FIGURE E.8 – All Awards Page

FIGURE E.9 – Add Awards Page
FIGURE E.10 – Add Committee Page

FIGURE E.11 – Add Show Page
### Lunch

<table>
<thead>
<tr>
<th>Date Range</th>
<th>09.03.2015 - 14.03.2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week</td>
<td>9</td>
</tr>
<tr>
<td>Term</td>
<td>History</td>
</tr>
<tr>
<td>Year</td>
<td>2015</td>
</tr>
<tr>
<td>Location</td>
<td>Player Theatre</td>
</tr>
<tr>
<td>Playwright</td>
<td>Steven Burcott</td>
</tr>
<tr>
<td>Type</td>
<td>Lunch</td>
</tr>
<tr>
<td>IESDA Show</td>
<td>No</td>
</tr>
</tbody>
</table>

### Production Crew

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
<th>Cast or Crew</th>
</tr>
</thead>
<tbody>
<tr>
<td>Claire O'Reilly</td>
<td>Director</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Tim Touey</td>
<td>Lighting Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Johann Fitzpatrick</td>
<td>Set Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Brian Donnelly</td>
<td>Sound Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Matt Armstrong</td>
<td>Man</td>
<td>Cast</td>
</tr>
<tr>
<td>Maggie Craig</td>
<td>Woman</td>
<td>Cast</td>
</tr>
</tbody>
</table>
D.U. Players

shows

Add Show

FIGURE E.13 – Edit Show Page

D.U. Players

Home

Contact Us

About Us

Archive

Committee Area

shows

Festivals

Add Show

All Shows

Search Shows

FIGURE E.14 – All Shows Page

You are logged in as user: ents

Log Out

You are logged in as user: ents

Log Out

FIGURE E.13 – Edit Show Page

Lunch

Name: Lunch

Date: 09.03.2019 - 14.03.2011

Week: 9

Term: Hilary

Year: 2015

Location: Player Theatre

Playwright: Steven Bursik

Type: Lunch

ISDA Show: No

Production Crew

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
<th>Cast or Crew</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clare O’Reily</td>
<td>Director</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Tilly Taylor</td>
<td>Lighting Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Johann Fitzpatrick</td>
<td>Set Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Brian Donnelly</td>
<td>Sound Design</td>
<td>Production Crew</td>
</tr>
<tr>
<td>Matt Armbragg</td>
<td>Man</td>
<td>Cast</td>
</tr>
<tr>
<td>Maggie Crane</td>
<td>Wombat</td>
<td>Cast</td>
</tr>
</tbody>
</table>

FIGURE E.14 – All Shows Page
Import Members from File

Upload new member data by browsing to file and clicking on Upload

File chosen must be a .csv file

File name to import: Choose File. No file chosen

FIGURE E.15 – Import CSV Page

Most Active Members Report

Showing Active Members for Year: 2012

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Member Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Michael Stone</td>
<td>12</td>
</tr>
<tr>
<td>Tilly Taylor</td>
<td>9</td>
</tr>
<tr>
<td>Brian Connolly</td>
<td>8</td>
</tr>
<tr>
<td>Brian Fallon</td>
<td>8</td>
</tr>
<tr>
<td>Mary Jamison</td>
<td>6</td>
</tr>
<tr>
<td>Alan Kas</td>
<td>4</td>
</tr>
<tr>
<td>Ada Leander</td>
<td>3</td>
</tr>
<tr>
<td>Ever Heslop</td>
<td>3</td>
</tr>
<tr>
<td>Elias Germain</td>
<td>3</td>
</tr>
<tr>
<td>Megan Carey</td>
<td>1</td>
</tr>
<tr>
<td>Matt Cooke</td>
<td>1</td>
</tr>
<tr>
<td>Griffin Manchester</td>
<td>1</td>
</tr>
<tr>
<td>Delia O’Fallona</td>
<td>1</td>
</tr>
<tr>
<td>Mary O’Gallahan</td>
<td>1</td>
</tr>
<tr>
<td>Alan Edgar</td>
<td>1</td>
</tr>
<tr>
<td>Adam Chalks</td>
<td>0</td>
</tr>
</tbody>
</table>

FIGURE E.16 – Most Active Members Report Page
SAMPLE SOURCE CODE

Due to the large amount of source code written for the application the source code can be found on the CD attached to this document. However there are two samples of code shown here.

Source code for "allmembers.php":

```php
<?php
session_start();

if(!isset($_SESSION["Username"])) {
    header('Location: Login.php');
}
?>
```

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Untitled Document</title>
<link href="Template.css" rel="stylesheet" type="text/css" />
<!--[if lte IE 7]> 
<style>
.content { margin-right: -1px; } /* this 1px negative margin can be placed on any of the columns in this layout with the same corrective effect. */ ul.nav a { zoom: 1; } /* the zoom property gives IE the hasLayout trigger it needs to correct extra whitespace between the links */
</style>
<![endif]-->
</head>

```php
$username="root";
$password="root";
$database="Players";

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or die("Unable to select database");

$resultsPerPage = 20;

if (!isset($_POST['page'])) {
    $Page = 0;
} else

if ($Page['page'] == 'test') {
```
$Page = 0;

} else {
    $Page = ($_POST['page']) + 1;
}

$currentOffset = $resultsPerPage * $Page;

$query = "SELECT * FROM Members ORDER BY FirstName LIMIT $resultsPerPage OFFSET $currentOffset";

$result = mysql_query($query);

$num = mysql_numrows($result);

?>

<body>
<div class="container">
    <div class="header">
        <div class="HeaderLogo" id="HeaderLogo">
            &nbsp;
            D.U. <p>
            Players</p>
        </div>
        <div class = "headermenu">
            <ul class="nav">
                <li><a href="home.php">Home</a></li>
                <li><a href="contactus.php">Contact Us</a></li>
                <li><a href="aboutus.php">About Us</a></li>
                <li><a href="archive.php">Archive</a></li>
                <li><a href="committee.php">Committee Area</a></li>
            </ul>
        </div>
    </div>
    <div class="content">
        <div id="Submenu">
            <ul class="nav2">
                <li><a href="allshows.php">Shows</a></li>
                <li><a href="allevents.php">Events</a></li>
                <li><a href="allmembers.php">Members</a></li>
                <li><a href="allfestivals.php">Festivals</a></li>
                <li><a href="allawards.php">Awards</a></li>
                <li><a href="allcommittees.php">Committees</a></li>
            </ul>
        </div>
    </div>
</div>
</body>
<?
echo "<div class="subsubmeny">

<ul class="nav">
<li><a href="addmember.php">Add Member</a></li>
<li><a href="allmembers.php">All Members</a></li>
<li><a href="importmembers.php">Import Member List</a></li>
</ul>
</div>

<div class="datatables">

<title>
<h2>Showing $resultsPerPage Records</h2>
</title>

<form action='searchname.php' method='post'>
<input type='submit' name='searchmember' value='go' style='float: right'/>
<div style='overflow: hidden; padding-right: .5em;'>
<input type='text' name='searchmember' placeholder='Search Members' style='width: 150;' />
</div>
</form>

<p></p>

<form action='allmembers.php' method='post'>

</form>

<p></p>

</div>

<?
if($Page==0){
$Previouspage = -1;
}else{
$Previouspage = $Page-2;
}

echo"<input id='Index' type='hidden' value='$Previouspage' name='page'>";

echo "<input type='Submit' value='View previous $resultsPerPage Records'>";

echo "<form action='allmembers.php' method='post'>";

<p></p>

<?

?>
<table>
<thead>
<tr>
<th>Name</th>
<th>Year Joined</th>
<th>Email</th>
<th>Course</th>
<th>Graduated</th>
<th>Lifetime</th>
</tr>
</thead>
</table>

```php
<?php
$i=0;
while ($i < $num) {

$MemberID=mysql_result($result,$i,"MemberID");
$Name1=mysql_result($result,$i,"FirstName");
$Name2=mysql_result($result,$i,"SecondName");
$Year=mysql_result($result,$i,"MemberYear");
$Email=mysql_result($result,$i,"Email");
$Course=mysql_result($result,$i,"Course");
$Graduated=mysql_result($result,$i,"Graduated");
$Lifetime=mysql_result($result,$i,"Lifetime");

$Name= $Name1 . ' ' . $Name2;
echo "<tr>";
echo "<td><a href ='indmember.php?MemberID=$MemberID'>
$Name</a></td>";
echo "<td>$Year</td>";
echo "<td>$Email</td>";
echo "<td>$Course</td>";
echo "<td>$Graduated</td>";
echo "<td>$Lifetime</td>";
echo "</tr>";

$i++;
}
?>
</table>
<?php
if(isset($_SESSION['Username'])) {
    echo "<p><a href='logout.php'>Log Out</a></p>";
    $temp = $_SESSION['Username'];
    echo "<p>You are logged in as user: $temp</p>";
} else{
    echo "<a href='Login.php'>Log In</a>";
}
?>
</div> <!-- end .footer -->
<!-- end .container -->
</div>
</body>
</html>
Source code for "searchawards.php":

```php
session_start();
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Untitled Document</title>
<link href="Template.css" rel="stylesheet" type="text/css" /></head>
<?php
$username="root";
$password="root";
$database="Players";

$Searchname = $_POST['searchmember'];

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or die( "Unable to select database");
$query1="SELECT ISDAAwards.*, Members.MemberID, Members.FirstName, Members.SecondName, Shows.ShowID, Shows.ShowName FROM ISDAAwards, Members, Shows WHERE Members.MemberID = ISDAAwards.ISDAMemberID AND Shows.ShowID = ISDAAwards.ISDAShowID AND(ISDAAwards.AwardID LIKE '%{$Searchname}%' OR ISDAAwards.AwardName LIKE '%{$Searchname}%' OR ISDAAwards.AwardYear LIKE '%{$Searchname}%' OR ISDAAwards.Host LIKE '%{$Searchname}' OR ISDAAwards.NomWin LIKE '%{$Searchname}%' OR Shows.ShowName LIKE '%{$Searchname}' OR Members.SecondName LIKE '%{$Searchname}%' OR Members.FirstName LIKE '%{$Searchname}%' )";

$results=mysql_query($query1);
```
$num=mysql_numrows($results);

mysql_close();

?>

<body>

<div class="container">
  <div class="header">
    <div class="HeaderLogo id=HeaderLogo">
      <p>&nbsp;</p>
      <p>D.U. </p>
      <p>Players</p>
    </div>
    <div class = "headermenu">
      <ul class="nav">
        <li><a href="home.php">Home</a></li>
        <li><a href="contactus.php">Contact Us</a></li>
        <li><a href="aboutus.php">About Us</a></li>
        <li><a href="archive.php">Archive</a></li>
        <li><a href="committee.php">Committee Area</a></li>
      </ul>
    </div>
  </div>
  <div class="content">
    <div id="Submenu">
      <ul class="nav2">
        <li><a href="allshows.php">Shows</a></li>
        <li><a href="allevents.php">Events</a></li>
        <li><a href="allmembers.php">Members</a></li>
        <li><a href="allfestivals.php">Festivals</a></li>
        <li><a href="allawards.php">Awards</a></li>
        <li><a href="allcommittees.php">Committees</a></li>
      </ul>
    </div>
    <div class="subsubmeny">
      <ul class="nav">
        <li><a href="addshow.php">Add Award</a></li>
        <li><a href="#news">All Awards</a></li>
        <form action='searchawards.php' method='post'>
          <input type='submit' name='searchmember' value='go' style='float: right' />
        </form>
      </ul>
    </div>
  </div>
</div>
<?php
if($num == 0)
{
    echo "<h1>No Results Found For:</h1>";
    echo "<h1>$Searchname</h1>";
}
else{
    $i=0;
    while ($i < $num) {
        $MemberID=mysql_result($results,$i,"MemberID");
        $Name=mysql_result($results,$i,"AwardName");
        $Year=mysql_result($results,$i,"AwardYear");
        $Host=mysql_result($results,$i,"Host");
        $Show=mysql_result($results,$i,"ShowName");
        $MemberName1=mysql_result($results,$i,"FirstName");
        $MemberName2=mysql_result($results,$i,"SecondName");
        $NomWin=mysql_result($results,$i,"NomWin");

        $MemberName= $MemberName1 . ' ' . $MemberName2;
        echo "<tr>
        echo "<td>$Name</td>
        echo "<td>$Year</td>
        echo "<td>$Host</td>
        echo "<td>$Show</td>
        echo "<td>$NomWin</td>
        echo "</tr>
    }
}
?>
echo "<td><a href='indmmember.php?MemberID=$MemberID'>$MemberName</td>";
echo "<td>$NomWin</td>";
</tr>
$i++;
}}
?>
</table>

</div> <!--end div#datatables-->

<p>&nbsp;</p>
<p>&nbsp;</p>
<!-- end .content --></div>
<div class="footer">
<?php
if(isset($_SESSION['Username'])) {
  echo "<p><a href='logout.php'>Log Out</a></p>
  $temp = $_SESSION['Username'];
  echo "<p>You are logged in as user: $temp</p>";
} else{
  echo "<a href='Login.php'>Log In</a>";
}
?>
<!-- end .footer --></div>
<!-- end .container --></div>
</body>
</html>
REFERENCES


• Blackknight Hosting:
  o https://www.blacknight.com/shared-hosting.html

• GoDaddy Hosting:
  o https://ie.godaddy.com/hosting/web-hosting.aspx

• Amazong Web Services:
  o http://aws.amazon.com/free/

• Hosting Ireland:
  o https://www.hostingireland.ie/cheap-web-hosting.php

• The Little Box Office:
  o https://thelittleboxoffice.com/

• Vendini:
  o http://www.vendini.com/ticketing/box-office-ticketing

• Spektrix:
  o https://www.spektrix.com/

• Savoy Systems:
  o http://savoysystems.co.uk/SavoySystems.dll/Home
TRINITY COLLEGE DUBLIN
Management Science and Information Systems Studies
Project Report

D.U. PLAYERS

Archives Database

Technical Manual
<table>
<thead>
<tr>
<th>NO.</th>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Overview</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>About</td>
<td>1</td>
</tr>
<tr>
<td>2.</td>
<td>Technical Environment</td>
<td>2</td>
</tr>
<tr>
<td>3.</td>
<td>Deployment</td>
<td>3</td>
</tr>
<tr>
<td>3.1</td>
<td>Database Set-Up</td>
<td>3</td>
</tr>
<tr>
<td>3.2</td>
<td>PHP Set-Up</td>
<td>5</td>
</tr>
<tr>
<td>4.</td>
<td>Aesthetics</td>
<td>6</td>
</tr>
<tr>
<td>5.</td>
<td>System Back Up</td>
<td>7</td>
</tr>
</tbody>
</table>
1. **OVERVIEW**

1.1. **Introduction**

This technical manual gives instructions for the initial set up of the system.

1.2. **About**

This application was produced by Deirdre Van Wolvelaere for Dublin University Players.
2. TECHNICAL ENVIRONMENT

Most of the system was developed using Dreamweaver CS4.

This application is made up of four different languages:

- PHP, for building the web pages.
- HTML, for building the web pages.
- SQL, for querying the MySQL database.
- CSS, for the aesthetics of the user interface.
- JQuery, for improving the user experience of some forms.
3. **DEPLOYMENT:**

In order for the client to run the system correctly the server needs to be set up correctly. A MAMP (Mac OS X, Apache, MySQL, PHP) software bundle is what was used in this project for developing the system. Using this software bundle requires the operating system, Mac OS X. If that is not the operating system used by the client, a different software bundle must be used. If using a Windows operating system, XAMPP must be downloaded. If using a Linux operating system, LAMP must be downloaded. Each of these software bundles consist of the following:

- Apache is the web server.
- MySQL is a relational database management system.
- PHP is a server-side website development language.
- PhpMyAdmin is intended to handle the administration of MySQL with the use of a web browser.

3.1 **Database Set-Up**

PhpMyAdmin must then be used to create a database called “Players”. Once the database is created, it is possible to upload the file “PlayersImport.sql” into the database. Within the “Players” database, select the “Import” option at the top of the page. Once at the import page, it is possible to upload file. This file will create all the tables within the database and populate the tables with existing records.

Alternatively, it is possible to set up the database by adding the following tables to the database. It is important the exact name below is used, otherwise a large amount of PHP alterations would be necessary.

**Table Name: Committees**

<table>
<thead>
<tr>
<th>Data Name:</th>
<th>CommitteeID</th>
<th>CommitteeName</th>
<th>CommitteeYear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type:</td>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>INT(4)</td>
</tr>
<tr>
<td>AUTO INCREMENT</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table Name: CommiteeMembers**

<table>
<thead>
<tr>
<th>CMCommitteeID</th>
<th>CMMemberID</th>
<th>CommitteeRole</th>
<th>Chairperson</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>INT(1)</td>
</tr>
</tbody>
</table>
### Table Name: Events

<table>
<thead>
<tr>
<th>EventID</th>
<th>EventName</th>
<th>EventYear</th>
<th>EventWeek</th>
<th>EventDate</th>
<th>EventBlurb</th>
<th>EntsOfficer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>year(4)</td>
<td>INT(1)</td>
<td>Varchar(45)</td>
<td>Varchar(500)</td>
<td>Int(11)</td>
</tr>
</tbody>
</table>

### Table Name: Festivals

<table>
<thead>
<tr>
<th>FestivalID</th>
<th>FestName</th>
<th>FestYear</th>
<th>FestDate</th>
<th>FestBlurb</th>
<th>FestCommitteeID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>year(4)</td>
<td>Varchar(45)</td>
<td>Varchar(500)</td>
<td>Int(11)</td>
</tr>
</tbody>
</table>

### Table Name: ISDAAwards

<table>
<thead>
<tr>
<th>AwardID</th>
<th>AwardName</th>
<th>AwardYear</th>
<th>Host</th>
<th>ISDAShowID</th>
<th>ISDAMemberID</th>
<th>NomWin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>year(4)</td>
<td>Varchar(45)</td>
<td>Varchar(500)</td>
<td>INT(11)</td>
<td>Varchar(45)</td>
</tr>
</tbody>
</table>

### Table Name: MemberRoles

<table>
<thead>
<tr>
<th>MRMemberID</th>
<th>MRShowID</th>
<th>MemberRole</th>
<th>Acting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>Int(11)</td>
<td>Varchar(45)</td>
<td>tinyint(1)</td>
</tr>
</tbody>
</table>
Table Name: Shows

<table>
<thead>
<tr>
<th>ShowID</th>
<th>ShowName</th>
<th>ShowDate</th>
<th>ShowWeek</th>
<th>ShowTerm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
</tr>
<tr>
<td>AUTO_INCREMENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table below is Shows table continued

<table>
<thead>
<tr>
<th>ShowYear</th>
<th>Showtype</th>
<th>Location</th>
<th>Playwright</th>
<th>ISDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>YEAR(4)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>TINYINT(1)</td>
</tr>
</tbody>
</table>

Table Name: Members

<table>
<thead>
<tr>
<th>MemberID</th>
<th>FirstName</th>
<th>SecondName</th>
<th>MemberYear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>YEAR(4)</td>
</tr>
<tr>
<td>AUTO_INCREMENT</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table below is Members table continued

<table>
<thead>
<tr>
<th>Email</th>
<th>Course</th>
<th>Graduated</th>
<th>Lifetime</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
</tr>
<tr>
<td>UNIQUE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table Name: Users

<table>
<thead>
<tr>
<th>UserID</th>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int(11)</td>
<td>VARCHAR(45)</td>
<td>VARCHAR(45)</td>
</tr>
<tr>
<td>AUTO_INCREMENT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.2. PHP Set-Up

Select all the files and place all of them on the server. This can be done by opening MAMP (or LAMP or XAMPP) and placing all the files into the 'htdocs' folder.
4. AESTHETICS

CSS (Cascading Style Sheets) was used to create the site template. All the pages of the site are linked to one external style sheet, “Template.css”. If any alterations need to be made to the style and aesthetic of the website, then the changes only need to be made once, within the “Template.css” file.

For example the following code controls the default aesthetics of each page. In order to change the background colour of each page, it is only necessary to change the attribute “background-color” below.

```html
body {
    margin: 0;
    padding: 0;
    color: #000;
    font-family: Arial, Helvetica, sans-serif;
    font-size: 100%;
    line-height: 1.4;
    background-color: #D2EDFF;
}
```
5. **SYSTEM BACK UP**

Backing up the system from phpMyAdmin is quick and easy. The following steps are necessary:

- Open phpMyAdmin
- Open the ‘Players’ database
- Select ‘Export’
- Select ‘Custom’ under ‘Export Method’
- Select which tables are to be backed up, or select the whole database.
- Select ‘Save Output to a File’ under ‘Output’
- Choose the format for the output from the list provided. Some notes on formats:
  - CSV for MS Excel is the most recognised format.
  - SQL allows you to import data into another database
- Press the ‘Go’ button