**CHAPTER THREE**

**SYSTEM DESIGN**

* 1. **INTRODUCTION**

This chapter describes in detail the system design methodology. It focuses on the system structure and interactions. The proposed system is a reference tracking system. It is created to be a desktop application or deployed on the web and is aimed at providing application processing services for all academy staff and research globally. This chapter begins by examining the Systems Requirement Specification (SRS) document which is majorly focused on only the functional requirements to be provided by the system. It proceeds to the system design which consists of the logic design. The logic design consists of various user interfaces and the chapter also explains the system design using UML diagrams.

* 1. **SYSTEM REQUIREMENT SPECIFICATION**

The system requirement specification is a structured document that collects information which encompasses the requirements of a system. This section would focus mainly on the functional requirements of the proposed system and these include:

1. The system should be able to validate all user input and respond to exceptions appropriately.
2. The system should enforce the policy of non-multiple users of an account using standard authentication processes.
3. The system should allow users to create and maintain staff details and also be able to delete staff from accessing the system.
4. The system should be able to track insecure penetrations and prevent unauthorized intrusions.
5. The system should also allow users to maintain an online profile.
   1. **SYSTEM DESIGN**

This section explains the design methodology, data and modules for the proposed system. The system design incorporates both UML diagrams and user interface designs.

* + 1. **LOGICAL DESIGN**

The logical design of the system is concerned with the underlying logic of the proposed system which would be abstracted from the various interfaces of the system. The interfaces discussed would be the input design and output design

* + 1. **INPUT DESIGN**

This section includes the various input design interfaces in the system. The input design interfaces to be considered would be the login form interface, Add Staff form, Edit staff form interfaces and reference form model.

**LOGIN WINDOW**

**Password:**

**User Name:**

**User Login**

**Login Now**

**Figure 3.1: Login Interface**

The Login interface allows authorize personnel to login into the system to make change and obtain appropriate results. The actual form contains more detailed information as seen in the application.

ADD USER

|  |
| --- |
| Add staff Detail |
| User Name  Password  fname  lname   |  | | --- | | REGISTER NOW |   level: |

* + 1. **OUTPUT DESIGN**

This section describes the various output of the system to the user. The format of output for the system is majorly text. The output that would be discussed would be the

Edit User

|  |
| --- |
| Edit Staff Detail |
| User Name  Password  fname  lname   |  | | --- | | UPDATE NOW |   level: |

View Staff Detail(Output window)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Username | Fname | Lname | Level |
|  |  |  |  |  |

* 1. **USE CASE DIAGRAM**

The use case diagram is a UML diagram that shows the users of a system and the various interactions that exists between the user and the system.

CASE DIAGRAM

LOGIN

SELECT REFERENCING STYLE

ENTER SOURCE DETAILS

INITIATE AUTOMATION

USER

LOGOUT

LOGIN

ADD USER

(SYSTEM USER)

EDIT/DELETE

USER (SYSTEM USER)

LOGOUT

ADMIN

**The Use Case Diagram**

The above use case shows the interaction between the user and the application. Discuss is the Admin and User interactions.

* 1. **ACTIVITY DIAGRAM**

Activity diagrams are representative of step wise activities and actions in a system. They describe the operational step-by-step work flow of components in a system.

Display UI

Enter command

Show registration form

Enter registration of data

Store data in database

Grant Access

Show Reference Menu

Select Reference style

Initiate command

View User Details

Delete user

Display results

DELETE USER

[Reference]

[Register]

* 1. **STRUCTURE OF DATABASE DESIGN**

The proposed system makes use of a relational database to store and maintain records. This database will consist of three (3) relational tables discussed below:

**TABLE**

**USER TABLE**

|  |  |  |
| --- | --- | --- |
| **FIELD** | **DATA TYPE** | **DESCRIPTION** |
| ID | INT (II)  (auto-increment) | Unique ID primary key Auto-increment |
| username | varchar(200) | Username of the user |
| password | Varchar (50) | Password of the user |
| Fname | Varchar (50) | Firstname of the user |
| Lname | Varchar (50) | Last name of the user |
| Level | Varchar (50) | Access level of the user |
|  |  |  |

* 1. **FLOW CHART**

End

User deleted

Yes No

Delete user

Display all user reference

Display result

Enter user detail

Add user

Is login?

Display result

Perform reference operation

Display all reference style

Userin

Admin

NO

Enter login details

Start

**CHAPTER FOUR**

**SYSTEM IMPLEMENTATION**

* 1. **INTRODUCTION**

This chapter focuses on the implementation of the system. The features of the implementation languages used in this research- PHP and MYSQL will be discussed extensively. The system testing strategies, the target computer requirements as well as the software maintenance issues that would arise in the system would be discussed also.

* 1. **FEATURES OF IMPLEMENTATION LANGUAGES**

The programming languages used in the implementation of this project are PHP (Hypertext Preprocessor) and MYSQL programming languages. PHP is a general purpose server side scripting language originally designed for web development to produce dynamic web pages. It has also evolved to include a command line interface capability and can be used in stand-alone graphical applications.

The following features make PHP a preferred implementation language for this project:

1. PHP has its root in C and C++. PHP syntax is most similar to C and C++ language syntax, so programmers find it easy to learn and manipulate.
2. PHP can run on both UNIX and windows. Hence it is compatible across various operating systems.
3. PHP has powerful output buffering that further increases over the output flow. PHP internally rearranges the buffer so that the header comes before the content.
4. PHP is platform independent: this is because it is parsed by the web browser hence compatibility issues do not arise when code written in PHP is ported to a different platform.
5. PHP can be used with a large number of relational database management systems, runs on all of the most popular web servers and is available to many different operating systems.
6. PHP is fully an object oriented programming language and its platform independence and speed on LINUX servers help to build large and complex web applications.
7. PHP has also attracted the development of many frameworks that provide building blocks and design structure to promote Rapid Application Development (RAD). Some of these include cake PHP, code igniter, Yii framework and Zend framework.
8. PHP IDS add security to any PHP application to defend against intrusion. PHPIDS detects cross-site scripting (XSS), SQL injection, header injection, directory traversal, remote file execution, local file execution and Denial of Service (DOS).

MYSQL is a relational database management system written in C and C++, that runs as a server providing multi user access to a number of databases. MYSQL is used basically to create a relational database structure on a server in order to store data or automate procedures. The following features make MYSQL also a preferred implementation language in this research:

1. MYSQL is written in C and C++ and tested with a broad range of different compilers. It also functions on different platforms.
2. It uses multi-layered server design with independent modules.
3. It is designed to be fully multi-threaded using kernel threads to easily use multiple CPUs if they are available.
4. It is a server/client system. The database server (MYSQL) and the arbitrary many clients (application programs) which communicates with the server to query data and save changes.
5. MYSQL is designed to make it relatively easy to add other storage engines. This is useful if you want to provide an SQL interface for an in-house database.
6. It provides transactional and non-transactional storage engines, uses very fast B-tree disk tables with index compression and a fast thread-base memory allocation system.
7. It executes very fast joins using an optimized nested loop join; implements in-memory hash tables which are used as temporary tables.
8. It implements SQL functions using a highly optimized class library that should be as fast as possible.
9. It provides the server as a separate program for use in a client/server networked environment and as a library that can be embedded (linked) into stand-alone applications. Such applications can be used in isolation or in environments where no network is available.
   1. **SYSTEM TESTING STRATEGIES**

This section is concerned with testing and debugging of the programs and general processes involved in achieving the objectives of the system requirement. System testing is conducted on a complete integrated system to evaluate the system’s compliance with its specified requirements. System testing falls within the scope of black box testing and as such should require no knowledge of the inner design of the code or logic. During system testing, the focus is on the software design, behavior and even the believed expectations of the customer. So we can also refer to the system testing phase as investigatory testing phase of the software development life cycle. The system testing strategies used in this system include the unit test and integration test.

* + 1. **UNIT TEST**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code and determine whether it behaves exactly as it is expected to behave. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use.

The most common approach to unit testing requires drivers and stubs to be written. The driver simulates a calling unit and the stub simulates a called unit. The investment of developer time in this activity sometimes results in demoting unit testing to a lower level of priority and that is almost always a mistake. Even though the drivers and stubs cost time money, unit testing provides some undeniable advantages. It allows for automation of the testing process, reduces difficulties in discovering errors contained in complex pieces of the application. During the unit testing of the application, errors uncovered by the researcher were rectified and the result was satisfactory.

* + 1. **INTEGRATION TESTING**

Integration testing is a logical extension of unit testing. In its simplest form, the units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program. The idea is to test combination of pieces and eventually expand the process to test your modules with those of other groups. Integration testing can be done in a variety of ways which include top-down approach, bottom-up approach and the umbrella approach.

In the integration testing of the software, satisfactory results were obtained from the test using the bottom-up approach.

* 1. **TARGET COMPUTER SYSTEM REQUIREMENTS**

This section considers the requirements that must be met by the target system to enable the developed software application function as required. The target computer system requirement will be discussed in the area of software and hardware requirements.

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| Operating system | Windows 2000, XP, Vista |
| Memory | 128MB or higher |
| Database | MySQL 5 |
| Web server | WAMP server |

**Table 4.1: software requirement for target computer system**

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| RAM | 256MB of RAM |
| Hard disk | 10GB of hard disk space |
| Processor | 333Hz or higher |

**Table 4.2: hardware requirements for target computer system**

* 1. **SOFTWARE MAINTENANCE ISSUES**

This section focuses on software maintenance issues. Software maintenance is the modification of a software product after delivery to correct faults, improve performance or other product attributes or to adapt the product to a new or changing environment. It also serves as an opportunity to improve the performance o the software to suit the needs of the users if it becomes necessary for the user requirements to be improved upon or changed.

Maintenance would be seen in three areas in this research; corrective maintenance, preventive maintenance and adaptive maintenance.

* + 1. **CORRECTIVE MAINTENANCE**

Corrective maintenance is a maintenance task performed to identify, isolate and rectify a fault so that the failed system can be restored to an operational condition within the tolerances or limits established for in-service operations. Necessary corrections in the form of removal, modification or addition of program modules should be permitted by the software to allow for optimal use of the application.

* + 1. **PREVENTIVE MAINTENANCE**

This is a schedule o planned maintenance actions aimed at the prevention of breakdowns and failures. The primary goal of preventive maintenance is to prevent the failure of software before it actually occurs. It is designed to preserve and enhance software reliability by replacing error-prone components before they actually fail. Recent technological advances in tools for inspection and diagnosis have enabled more accurate and effective software maintenance. Measures like regular diagnosis, database backups, creating system mirrors preserve the integrity of information stored in the application. If these are strictly followed, limited instances of such occurrences would be noticed in the use of the software application.

* + 1. **ADAPTIVE MAINTENANCE**

This involves enhancing the system by adding features, capabilities and functions in response to new technology, upgrades, new requirements or new problems. Since the environment in which the application would be running is dynamic, it should be made to suit whatever requirements that may change in the long run.

**CHAPTER FIVE**

**5.0 Introduction**

This chapter focuses on summary, conclusion and recommendations.

Here, the entire summary of the research from the problem stage to the implementation stage, the relevant conclusion and recommendations are discussed.

**5.1 Constraints of the Study**

The problems encountered during the course of carrying out this research project include:

1. **Time**: Time for the research project was too short coupled with researcher’s academic time table.

2. **Fund**: There was limited fund to take care of the research properly in terms of transportation and other expenses especially when visiting attraction sites.

3. **Research Materials:** Lack of access to research materials on the topic in the school library and even public libraries were also major constraint in the cause of this project.

4. **Reference System:** Access to existing referencing system pose a great challenge as most online materials are has no software develop on this topic except a written documentation on this domain.

**5.2 Summary**

Implementation of an online and offline referencing system has been created.

The existing method of reading reference material on its book or online has been eliminated. Where all the referencing types has been full automated by the system. The new system would be very easy to use because of its accuracy and reliability. Information about each referencing style and type and its examples are recorded for easy understanding.

**5.3 CONCLUSION**

Referencing is a standardized method of formatting the information sources you have used in your assignments or written work (usq, 2016). Any given referencing style serves two purposes:

1. acknowledges the source
2. allows the reader to trace the source.

Referencing system is a system that’s takes care of every reference style by providing the user with a form to add his/her source details, the system automates it and display his reference using the standard format.

**5.4 Recommendations**

Having designed, tested and implemented the new system, the following must be put in place to fully achieve the objective of which the software is designed.

1. **Maintenance:** The system needs to be maintained. This implies that any fault detected should be reported to the programmer for correction at any point in time.
2. **Internet Connection:** The system needs to be connected to the internet before the user can access the features of the software online else it can be deployed ofline using a remote server(local server: WAMP, XAMP, MANP e.t.c).
3. **Research:** More research should be conducted on the topic to assess it effectively.

**Reference**

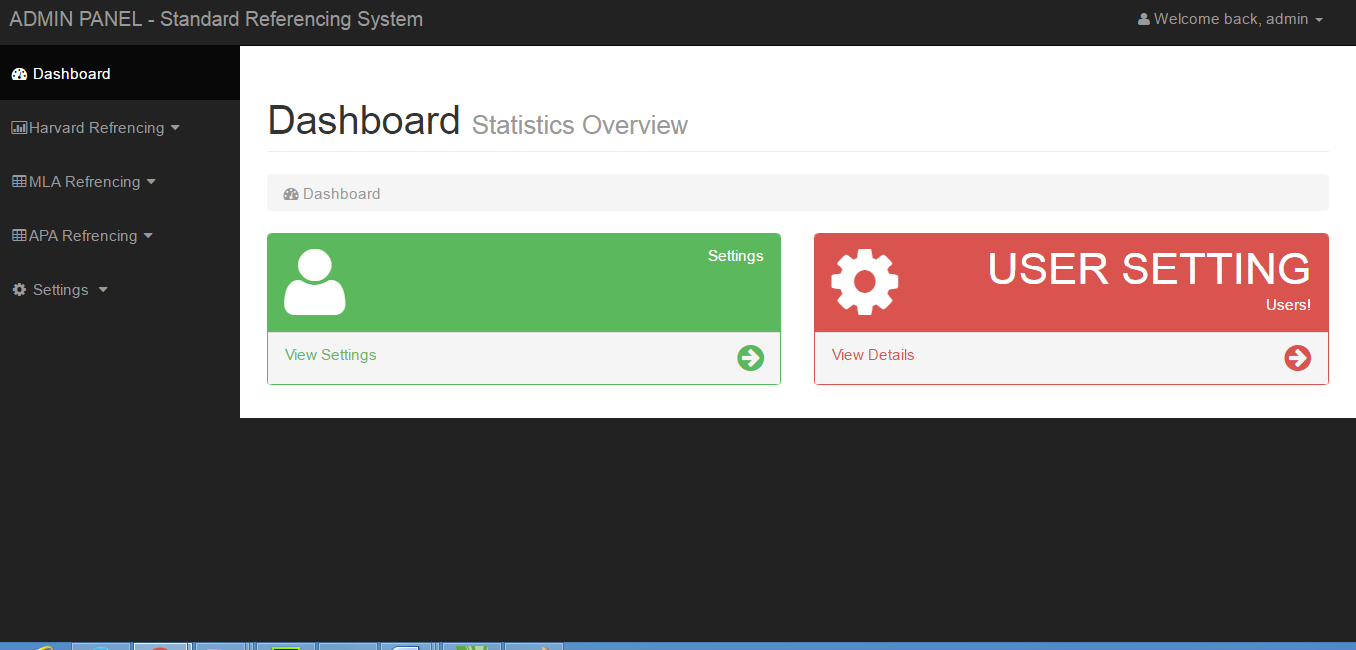
University of Southern Queensland

https://www.usq.edu.au/library/referencing/what-is-referencing

**APPENDIX**

**APPENDIX A: WINDOW SCREEN**

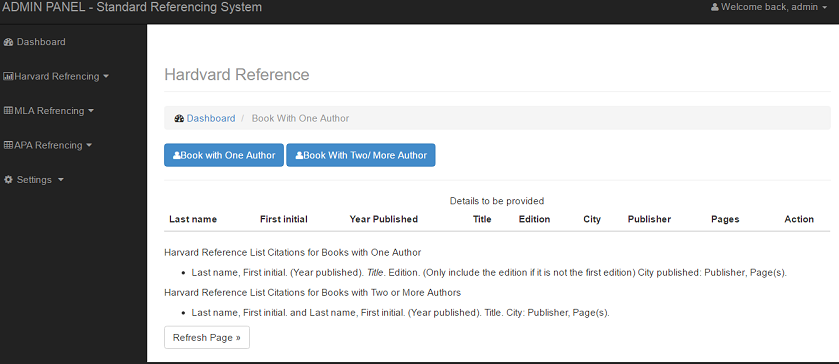
**Dash board window**

****

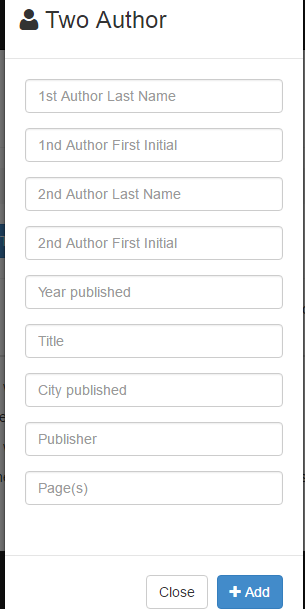
**Login window**

****

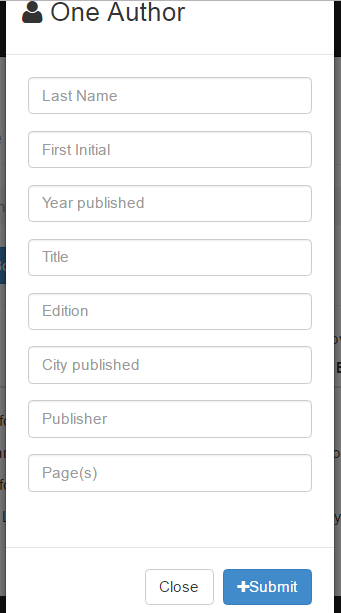
**HARVARD REFERENCE MODEL FOR for Author’s**

****

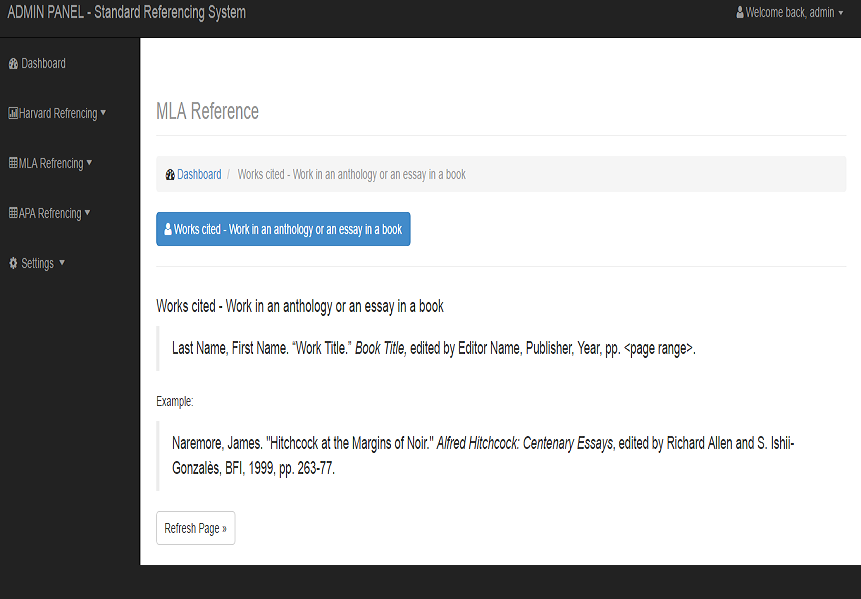
**HARVARD REFERENCE MODEL FOR AURTHOR TWO**

****

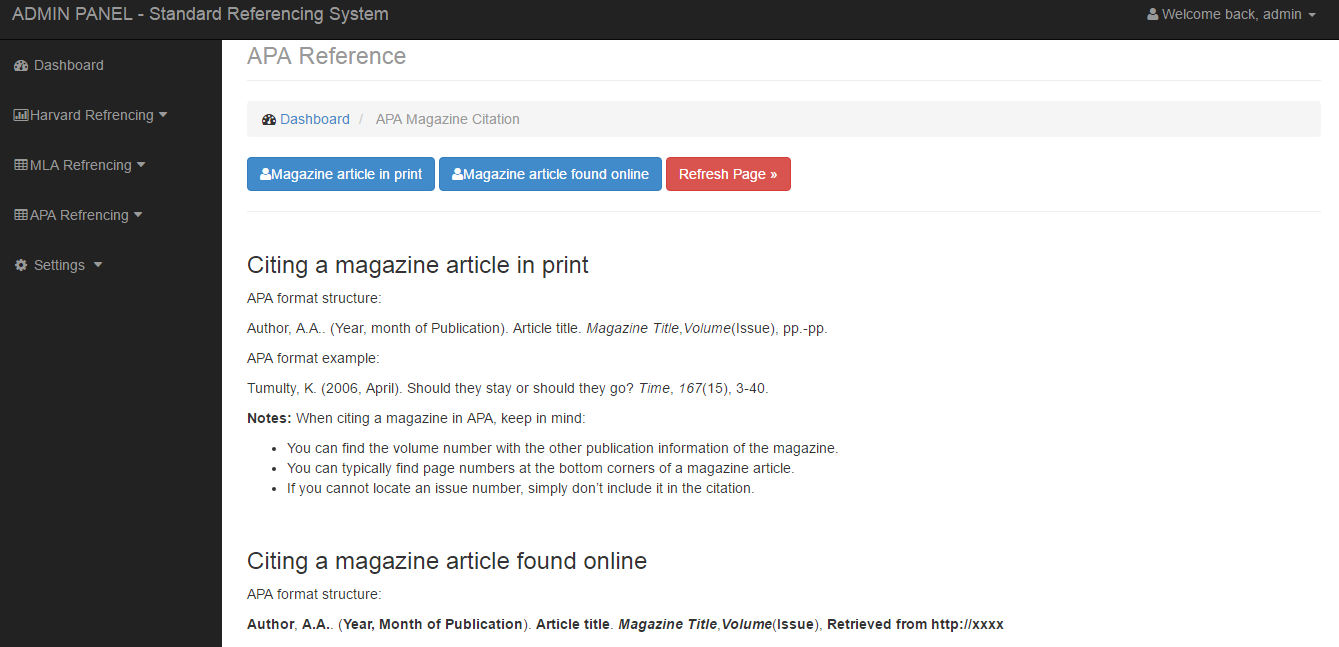
**HARVARD REFERENCE MODEL FOR AUTHOR ONE**

****

**MLA REFERENCE MODEL FOR ANTHOLOGY**

****

**APA MAGAZINE**

****