WXES 3182

Personal Library System

Mohamed Irfan bin Jan Mohamed

WET 990203

Supervisor : Mr. Ang Tan Fong

Moderator : En Mohamad Nizam Hj Ayub

Abstract

According to the thesis title – Personal Library System is a web – based library system that will enable necessary and relevant data to be stores online and be accessible at all times.

By using this system users will be able to search for necessary information on lecturers and members of the faculty at ease. This will not only increase afficiency but will also indirectly create an inventory of all the skills and know how available.

Lecturers will be able to include personal details about themselves and an inventory of books, online journals, software and other misceallanous resources accessible to students. This system will also record students who have borrowed their resources making it easier to keep track of such occurances.

This system will be developed with ASP.NET. The operating system to be used is Windows 2000 Professional. Microsoft SQL Server 2000 shall be used as the database management system and Internet Information Server(IIS) shall act as webserver. Other development tools include Microsoft Visual Studio.NET Architect Edition and Editplus.

With the implementation of the Personal Library System our faculty will hopefully be even more user – friendly.

Acknowledgements

I would like to express my deepest gratitude for the contributions of several people whohave helped me to complete this project.

Firstly I would like to convey my gratitude to my advisor Mr Ang Tan Fong for giving me valuable assistance in overcoming the initial and later problems I faced throughout the process of completing this project.

Secondly I would like to thank my moderator En. Mohamad Nizam Hj Ayub who took time of from his busy schedule to moderate my VIVA and project report.

Last but not the least I would like to thank my family and friends who were always there to support me in my difficulties and often assisting me in my research.

Table of Contents

ABSTRACT ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES xi

CHAPTER 1 : INTRODUCTION

1.1 OVERVIEW 1

1.2 PROBLEM DEFINITION 3

1.3 PROJECT OBJECTIVE 3

1.4 PROJECT SCOPE 4

1.4.1 Project Content 4

1.4.2 Project features 5

1.4.3 Targeted Audience 5

1.5 EXPECTED OUTCOME 6

1.6 PROJECT LIMITATION 6

1.7 PROJECT SCHEDULE 7

CHAPTER 2 : REVIEW OF LITERATURE

2.1 INTRODUCTION TO PERSONAL LIBRARY SYSTEMS 9

2.1.1 What is a Personal Library System 9

2.2 Analytical Studies 10

2.2.1 Case Study 1:Biotech Personal Library Homepage 10

2.2.2 Case Study 2: Digital Library Project, University of California, Berkeley 12

2.2.3 Case Study 3: Brarydog.Net 14

2.3 Software Architecture 16

2.3.1 Mainframe Architecture 16

2.3.2 Client-Server Architecture 16

2.3.3 One -Tier Architecture 19

2.3.4 Two – Tier Architecture 19

2.3.5 Three – Tier Architecture 21

2.4 Security Technology 24

2.4.1 Secure Sockets Layer(SSL) 24

2.5 Web Server 27

2.5.1 Apache 28

2.5.2 Internet Information Server 5.0 31

2.6 Operating system 32

2.6.1 Windows 98 33

2.6.2 Windows 2000 34

2.6.3 UNIX 34

2.6.4 Linux 36

2.7 Database Server **37**

2.7.1 MySQL 37

2.7.2 Microsoft SQL Server 38

2.8 Data Access Technology 40

2.8.1 Open Database Connectivity (ODBC) 40

2.8.2 ActiveX Data Objects (ADO) 42

2.8.3 OLE DB 43

2.9 Language 44

2.9.1 ASP 44

2.9.2 JSP 45

2.9.3 PHP 47

2.9.4 ASP.NET 47

CHAPTER 3 : METHODOLOGY

3.1 Project Development Life Cycle 49

3.2 Software Process Model 49

3.3 Justification of proposed Methodology 52

CHAPTER 4 : SYSTEM ANALYSIS

4.1 Requirement Analysis 53

4.1.1 Functional Requirements 54

4.1.2 Non - Functional Requirements 55

4.1.3 Techniques Used to Define Requirements 57

4.2 Chosen Platform, Web Server, Database, Language and Tools 58

4.2.1 Choice of Software Architecture 58

4.2.2 Choice of Development Platform 58

4.2.3 Choice of Database Management System 59

4.2.4 Choice of Data Access Technology 60

4.2.5 Choice of Web Development Server 61

4.2.6 Choice of Web Development Tools 61

4.3 System Requirements

CHAPTER 5 : SYSTEM DESIGN

5.1 Overview 63

5.2 System Architecture Design 64

5.3 System Functionality Design 65

5.3.1 System Structure 65

5.3.2 Data Flow Diagram 66

5.4 Database Design 68

5.5 User Interface Design 70

Chapter 6: System Development and Implementation

Introduction 72

6.1 Database Development 73

6.1.1 Database Design 73

6.1.2 Building of Queries 74

6.2 Module Development and Implementation 77

6.2.1 Building The Database Connection 78

6.2.2 Testing out the SQL query in and ASP.NET environment 78

6.2.3 Building The Data Manipulation and Control Module 79

6.2.5 Building The User Access Levels 84

6.2.6 Building The User Access Levels Search Module 86

6.2.7 Building The Resources Module 88

Chapter 7: System Testing

7.1 Introduction to Testing Software 91

7.2 What is testing 91

7.3 Types of testing 91

7.3.1 Component Testing 91

7.3.2 Integration Testing 92

7.3.3 Defect Testing 93

7.3.4 Black Box Testing 94

7.3.5 White Box Testing 95

7.4 Unit Testing 96

7.4.1 Database Queries Testing 96

7.5 Integration Testing 97

7.6 System Testing

Chapter 8: Discussion

Introduction 99

8.1 Problems and Solutions 100

8.1.1 Problems faced with the development tools 100

8.1.2 Problems and Advantages with ASP.Net 101

8.2 Strengths and Weaknesses of the Systems 103

8.3 Improvements and Suggestions 104

8.4 Conclusion 104

Bibliography

User Manual 106

Module Source Codes 112

Reference 72

LIST OF FIGURES

Figure 1.1: Project Schedule 8

Figure 2.1: Biotech Personal Library Home Page 10

Figure 2.2: University of California, Berkeley Digital Library Project 12

Figure 2.3: BraryDog.Net Personal Homepage 14

Figure 2.4: Two Tier Client Server Architecture Design 20

Figure 2.5: Three- Tier Architecture 24

Figure 2.6: SSL runs above TCP/IP and below high-level application protocols 25

Figure 2.7: Structure of Operating System 32

Figure 3.1: Evolutionary Model 51

Figure 5.1: Thin Client Model 64

Figure 5.2 :Context Diagram 66

Figure 5.3 :Database Design 68

Figure 6.1 :Database Diagram 73

Figure 6.2: Design of tblMaster 74

Figure 6.3: Example of sql query built using View option in SQL Server 2000 75

Figure 6.4: Edittable datagrid for updating the membership information 80

Figure 6.3: Login Page (newlogin.aspx) 85

Figure 6.4: Search Page (selmedia.aspx) 87

Figure 6.5: The form for adding new books (newbook.aspx) 89

Figure 6.6: The add books page with the datagrid at the bottom (newbook.aspx) 90

Figure 7.1: Testing Phases 92

Figure 7.2: Top-Down Integration 93

Figure 7.3: Bottom-Up Integration 93

Figure 7.4: The Defect Testing Process 94

Figure 7.5: The 'Black Box' Testing Process 94

Figure 7.6: The 'White Box' Testing Process 95

Figure 7.7: The Integration Testing Process 97

CHAPTER 1 : INTRODUCTION

Chapter 1: Introduction

1.1 Overview

A library is defined as a place in which books, recording, films and other forms of media are kept. Nowadays most libraries also use an Online Paging Access Control(OPAC) to keep an inventory of all its media. This system would also enable users to search for the available media, the status and the location.

In a way a personal library system has similar concepts to an OPAC. This system would hold data of all personnel or staff in a certain organisation and details of these person(s). In addition the system can also hold information useful to the users like the career paths of the personnel, available media, experience and other miscellaneous information.

A personal library system is actually the most basic idea in implementating database solutiuons. Many people are utilising similar ideas and solutions without realising it. A good example would be the address book on a Personal Digital Assistant (PDA). Here this system is able to store necessary contact information of person(s) and provide links to appointments made with them.

Several Multi National Corporations like Sclumberger, Shell and Petronas have been profiting directly or indirectly from such systems in terms of increasing efficiency, reducing redundancy of raw data. Starting with a basic database of employees these

systems have expanded to becoming separate or integrated systems holding compiled data of the companies.

In a way a Personal Library System is not a significant asset as it does not directly generate income for a company. However it does increase efficiency and reduce cost. This will indirectly contribute to the profits of a company. This system can also be expanded to becoming an inventory of all equipment in an organisation making it easier to keep record of the usage of such equipment and the locations at any point of time.

Basically the personal library system I intend to develop will hold library information on all staff enabling easy searches with the help of a search option. This system will enable users to obtain necessary information of the available resources on campus without the hassle of having to go through other people or the risk of handling third hand information.

At the same time the people in an organisation will also have a record of equipment others have borrowed from them. This will also make it easier when performing ISO auditing.

In a way the most significant progress as a result of the implementation of this system is that it reduces repetition of data entry. A personal library system will hold all necessary data, convenient users and contribute towards a paperless office.

1.2 Problem Definition

The traditional/manual practise for a personal library has a number of weaknesses. Some of the major disadvantages of the present system are stated below:

- **Time Consuming**

  The traditional system consists of physical files where information of staff or personel are kept in various offices. In a faculty each lecturer would have their own website but not necessarily their own personal library. Having this system would reduce time in searching for resources.

- **Inconsistency of Data**

  A number of academic staff have developed their own website allowing students to download notes, get links to various websites and find information about their lecturers. However their is no consistency or basic guidelines for the information that should be included on the website. This has resulted in differing information on these websites. While some lecturers provide complete information about themselves others merely upload lecture notes. Students have difficulty finding information about their lecturers especially when enquiring about their expertise in supervising final year projects.

1.3 Project Objectives

The main purpose of this project is to solve the problems of the current system(s). The core objectives of the project are:

- Reduce time involved in searching for information avalaible resources
- Build an information centre that will complement the existing personal websites by providing links to them and information not contained in them.
- To provide a user – friendly database system enabling lecturers to construct their very own personal library.
- To develop a system that will integrate all future and existing systems if not by integration the database at least by linking them.
- Create a powerful and stable system utilising database solutions which will either directly or indirectly increase efficiency
- Build a system that will hold a record of necessary information making is easier to perform ISO auditing.

1.4 Project Scope

1.4.1 Project Content

The scope of this project has been limited to a much smaller scale. Unlike the existing systems that often is implemented via wide area networks the system I intend to design is scaled down to a single faculty. The faculty I shall be using for the implementation of this system is the Faculty of Computer Science and Information Technology, University Malaya. The contents of the system should include:

- General information about all faculty staff
- Research interest of faculty staff
- Available resources of faculty members that students have access to
- Ongoing research being done on faculty and members of the faculty involved in it
- Career paths of academic staff and future plans
- A transaction section that keeps information of resources that have been borrowed and the person(s) concerned
- Necessary security and authentication when inserting, deleting and updating database.

1.4.2 Project Features

The following are a list of features that will be given emphasis in this project:

- Backend functionalities and database systems
- The implementation of a relational database on a database server platform
- Complete search options for querying the database
- Multiple access levels consisting of administrator, lecturers, students and outsiders
- Online user interface for editing and manipulating data
- Administration and maintenance page for the purpose of housekeeping in the system by authorized person(s).

1.5 Expected Outcome

The expected outcome of this project are listed below:

- The design of the system has to be simple and user – friendly. The system will be easy to use as it is menu driven. Users should be able to operate the even without any training or online help.
- The data entry will be checked by the system on the client side and should there be any errors the system will report it to the user by sending a user – friendly message.
- The system will generate accurate and easy to read reports of with certain extra functions for users at different access levels. Example: mid-level users would be able to manipulate personal data through the user interface.
- In relation to the statement above the completed system will have a number of access levels catering for outsiders, students of the faculty and lecturers.
- All web pages will have a standard and dynamic graphical user interface that will have the same interface across multiple browser display.
- The database cam be easily expanded if the capabilities and functionalities of the system increase in the future.

1.6 Project Limitations

The limitations of this project basically surround the scope of the system. The obvious limitations will only be apparent once the system has been used for a period of time as

user requirements would be different then. The limitations of the project are listed below:

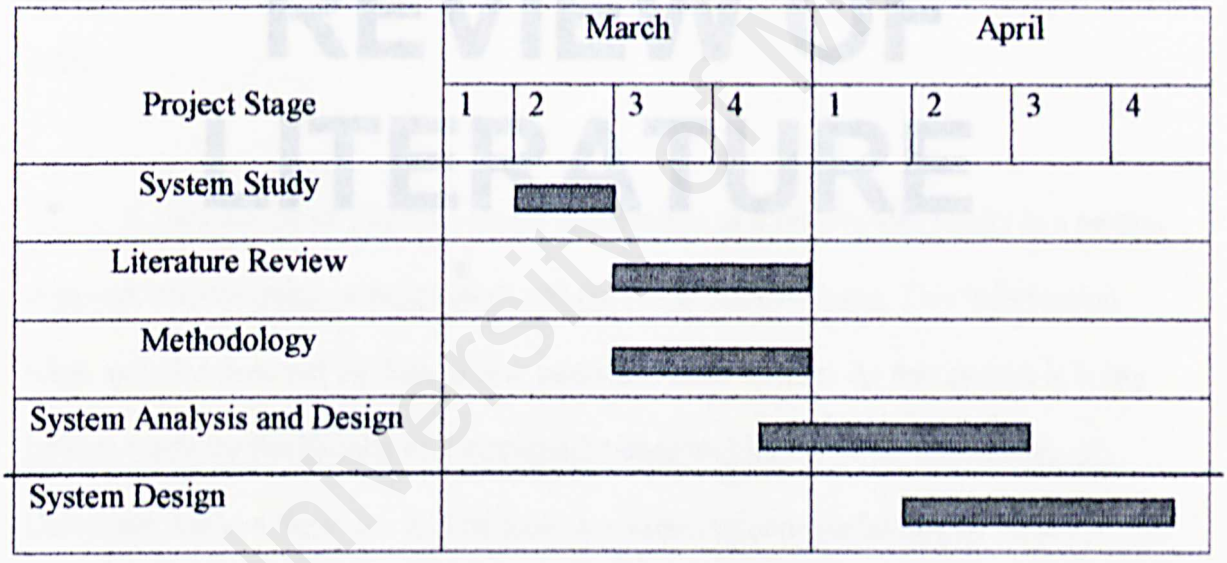
- The main limitation for this project is that it is an online system and therefore dependant on the availability of internet access. Without access to the world wide web users would no be able to access the system. However should the system be access regularly from the faculty computer labs the, certain contents of the system would be stored in the server cache enabling users limited access to the system.
- The response time of the system would depend on the efficiency of the webhost and the internet service provider.
- As this system is dependant on a database it would need expansion and changes in the future.This system will serve its purpose for now taking into consideration the present technology. Should technology change some parts of the system might become redundant

1.7 Project Schedule

In purpose of achieving the objectives of the system, project milestones have to be drawn. These milestones will assist in allocating the time for each stage in developing the system. This will lead to preparing a set of guidelines in developing the system. A project must be properly managed as it may involved extensive effort. Good project management is necessary in coordinating the aspects of a project. Hence the project can be completed despite the constraints and obstacles defined. As the time span of this project is limited certain aspects have to be considered. Among them are:

- Defining the goals of the project
- Defining, identifying and allocation of resources
- Establish a project schedule and a work breakdown
- Trace and monitor progress to ensure milestones are accomplished on time
- Document the development process of the project

A project schedule has been drawn for a graphical view of the work break down and project schedule. The project schedule of the Personal Library System is shown in Figure 1.1.



CHAPTER 2 : REVIEW OF LITERATURE

Chapter 2: Review of Literature

2.1 Introduction to Personal Library Systems

2.1.1 What is a Personal Library System

A Library is basically a place that stores all types of media accessible by members of the library. A personal library in a way is similar to that. Instead of storing media it stores information in a database, again accessible to members or users with accounts to the system. In a way this system has certain similarities to an Online Paging Access System (OPAC).

A personal library system stores information of a person who works in a certain organisation, company or department and the resources they have. This information often includes personal biodata, career paths and other details. As this system is being custom made for the Faculty of Computer Science and Information Technology of University Malaya the focus will be more on resources and availability of these resources.

The information available would be more towards academic achievements and research. Other valid information to students of the faculty such as subjects taught and thesis titles offered would also be available. This would make it easier for students to find information about their lecturers.

2.2 Analytical Studies

2.2.1 Case Study 1: Case Study 1:Biotech Personal Library Homepage

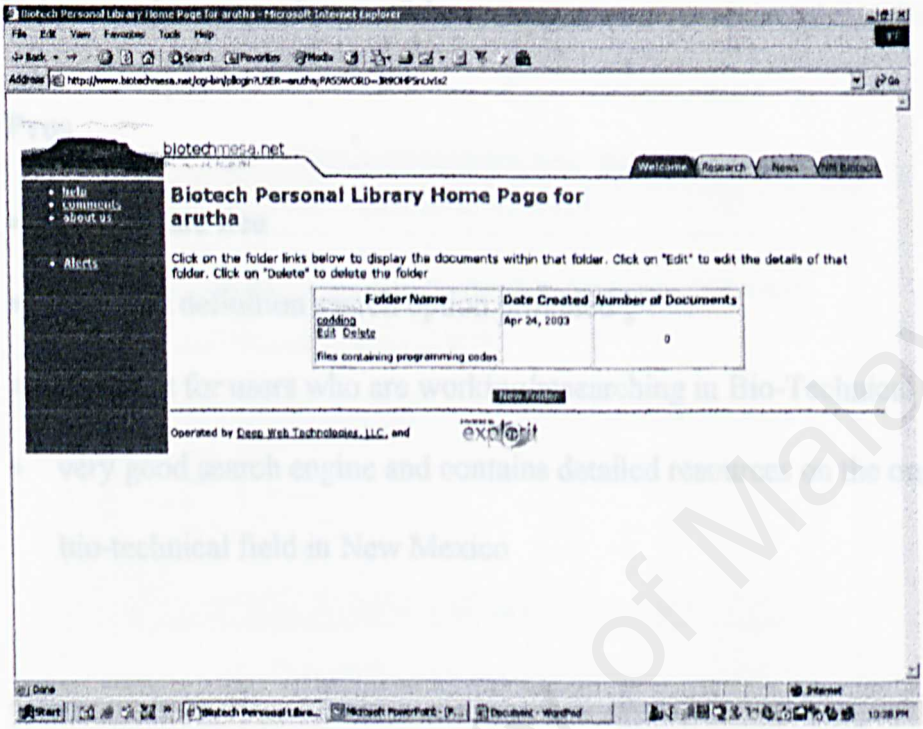


Figure 2.1: Biotech Personal Library Home Page

Biotechmesa.net (http://www.biotechmesa.net/) is THE site for the biotechnology community in New Mexico. Here biotech and biomedical researchers can search a wide array of research resources and information technology databases at the same time. Users can search biotechnology dictionary, glossary, and thesaurus at the same time. [28]

Biotech Personal Library Homepage is a system that creates a compilation the best online references that has relevance to the bio-technical industry. The personal library is actually part of BiotechMesa.net. This website in a way creates an online

presence for other people in the bio-technical industry to refer to. Members of the website can use the personal library services to compile information about their research and also find resources that might assist them. Here users can store and categorise documents of interest retrieved through Biotechmesa.net.

Below are the pros and cons of the system:

Pros

- services are free
- term and definition search option provided
- excellent for users who are working/researching in Bio-Technical field
- very good search engine and contains detailed resources on the ongoing work in the bio-technical field in New Mexico

Cons

- Resources are limited to search results retrieved from search option provided by website.
- Search options are limited to bio-tech websites or searches that are linked to the website.

2.2.2 Case Study 2: Digital Library Project, University of California, Berkeley

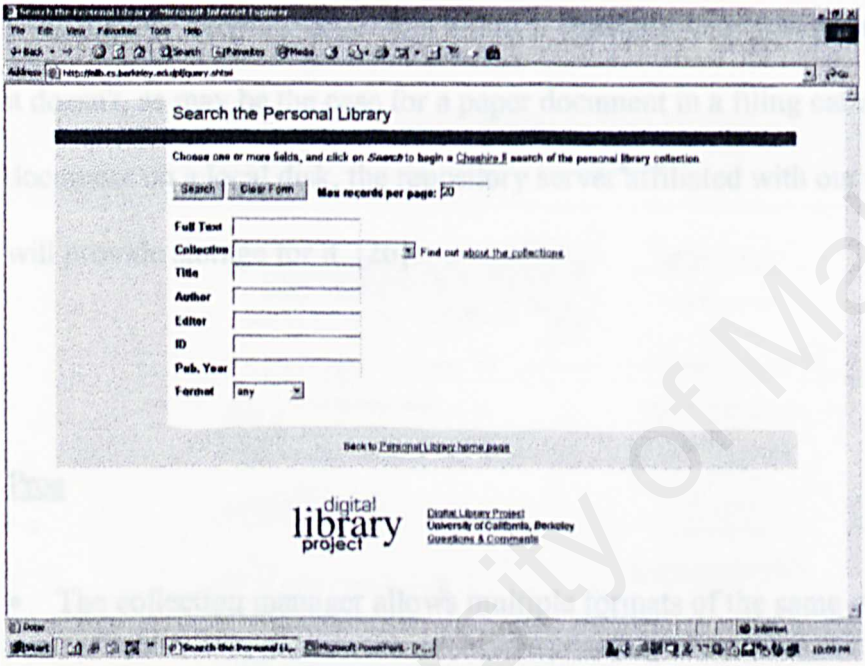


Figure 2.2: University of California, Berkeley Digital Library Project

The Digital Library Project is a set of services that allows users to store, organize and maintain network-accessible distributed document resources. Examples of such resources are on-line readers for courses, collections of documents related to some topic, collections of personal documents, technical report series, and collections of documents maintained by a working group.

The primary services provided by the PL are collection management service and repository service. In our terminology, a collection is a set of documents, which in turn may comprise one or more (potentially distributed) resources (general, a fixation of that document in a given format). The collection manager lets users create, populate, maintain and search collections, among other things. Of course, the resources comprising documents in a collection have to live somewhere. If a resource already has a satisfactory network-accessible home, a collection may just point to that. However, if it doesn't, as may be the case for a paper document in a filing cabinet or an electronic document on a local disk, the repository server affiliated with our collection manager will provide storage for it. [26]

Pros

- The collection manager allows multiple formats of the same work to be organized together.
- The collection manager caches resources in each collection
- Items in each collection are indexed, both by metadata and full-text
- The collection manager, in conjunction with the repository manager, allows users to easily include scanned image documents and born-digital documents in the same collection
- Accessible to collections can be restricted to a group, facilitating collaborate among individuals across different administrative domains
- Material requiring some protection can easily be made available only via a password

Cons

- Modifications to entries are not immediately updated automatically instead updated manually when the system administrator re-indexes the collections

2.2.3 Case Study 3: Brarydog.Net

Figure 2.3: BraryDog.Net Personal Homepage

BraryDog.Net is a students homework and help companion. It is tailor maid for students in the Charlotte & Mecklenburg County and the users of the libraries in the county.

BraryDog provides a system allowing students to browse the collection of these libraries for videos, books, cds, and other resources available at the libraries. [27]

Pros:

- Easy to access and easy to use
- Available access to several types of information sites, e-books etc.
- Excellent search results when searching for relevant subjects
- Provides online services for members of the Public Library of Charlotte & Mecklenburg County

Cons:

- User unable to revamp personal library to suit self instead must use tools and looks provided by website
- All options are what is provided by the website, users can't add their own options to personal site
- Resources are limited to what is offered by the website
- Not all resources are free, certain resources are limited and require payment

2.3 Software Architecture

There are a few types of software architecture available nowadays. Among them are mainframe architecture, client – server architecture, one – tier architecture and three – tier architecture.

2.3.1 Mainframe Architecture

Mainframe architecture is a client/server architecture. With mainframe software architectures intelligence is contained within the central host computer. Users interact with a host through a terminal that captures keystrokes and sends information to the host. User interaction is can be done using either PCs or UNIX workstations.

Among the limitations of this architecture is that it does not easily support Graphical User Interfaces (GUI) or access to multiple databases from geographically dispersed sites. In the last few years Mainframe has found a new use as a server in distributed client/server architectures. [1]

2.3.2 Client-Server Architecture

Client: The client part of a client-server architecture. Typically, a client is an application that runs on a personal computer or workstation and relies on a server to perform some operations. For example, an e-mail client is an application that enables you to send and receive e-mail. [2]

Server: A computer or device on a network that manages network resources. For example, a file server is a computer and storage device dedicated to storing files. Any user on the network can store files on the server. A print server is a computer that

manages one or more printers, and a network server is a computer that manages network traffic. A database server is a computer system that processes database queries.

Servers are often dedicated, meaning that they perform no other tasks besides their server tasks. On multiprocessing operating systems, however, a single computer can execute several programs at once. A server in this case could refer to the program that is managing resources rather than the entire computer. [3]

Client/Server Architecture

The term client/server was first used in the 1980s in reference to personal computers (PCs) on a network. The actual client/server model started gaining acceptance in the late 1980s. The client/server software architecture is a versatile, message-based and modular infrastructure that is intended to improve usability, flexibility, interoperability, and scalability as compared to centralized, mainframe, time sharing computing. [4]

Client/Server is a computational architecture that involves client processes requesting services from server processes. In general client/server maintains a distinction between processes and network services. Usually the client and the server are 2 separate devices, each customized for the necessary given tasks. For example a Web server would contain large amount of storage space, whereas the client(s) would feature necessary features to

support the graphical user interface of the browser such as high end graphic memory and high resolution screen displays.

The client/server architecture reduced network traffic by providing a query response rather than total file transfer. It improves multi-user updating through a GUI front end to a shared database. In client/server architectures, Remote Procedure Calls (RPCs) or standard query language (SQL) statements are typically used to communicate between the client and server. [1]

RPC is a client/server infrastructure that increases the interoperability, portability and flexibility of an application by allowing the application to be distributed over multiple heterogeneous platforms. It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces--function calls are the programmer's interface when using RPC .

The concept of RPC has been discussed in literature as far back as 1976, with full-scale implementations appearing in the late 1970s and early 1980s .

2.3.3 One -Tier Architecture

A One-tier application is simply a program that doesn't need to access the network while running. Most simple desktop applications like word processors or compilers, fall into this category.

Among the advantage of this architecture is simplicity. One-tier applications don't need to handle any network protocols, so their code is simpler. Such code also benefits from being part of an independent operation. It needs neither guarantee synchronization with data at other machines nor locations does it need exception handling routines to deal with network failures, inaccurate data from the server, or a server running different protocols or systems.

Moreover, one-tier applications can have a major performance advantage: the users' requests don't need to cross the network, queue at the server, then return to the client. This added effect of not weighing down your network with extra traffic and does not burden servers with extra tasks.[5]

2.3.4 Two – Tier Architecture

Two tier software architectures were developed in the 1980s from the file server software architecture design. The two tier architecture is intended to improve usability by supporting a forms-based, user-friendly interface. The two tier architecture improves scalability by accommodating up to 100 users (file server architectures only accommodate a dozen users), and improves flexibility by allowing data to be shared, usually within a homogeneous environment. The two tier architecture requires minimal

operator intervention, and is frequently used in non-complex, non-time critical information processing systems.

Two tier architectures consist of three components distributed in two layers: client (requester of services) and server (provider of services). The three components are

1. User System Interface (such as session, text input, dialog, and display management services)
2. Processing Management (such as process development, process enactment, process monitoring, and process resource services)
3. Database Management (such as data and file services)

The two tier design allocates the user system interface exclusively to the client. It places database management on the server and splits the processing management between client and server, creating two layers. Figure 2.4 depicts the two tier software architecture.[6]

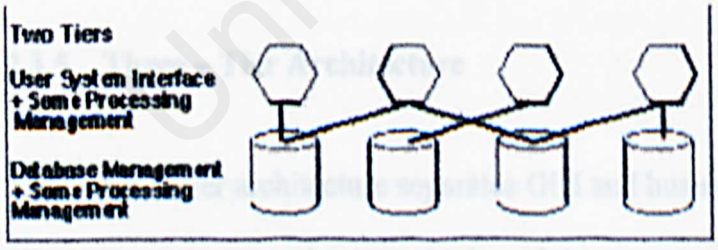


Figure 2.4: Two Tier Client Server Architecture Design

With two tier client/server architectures, the user system interface is usually located in the user's desktop environment and the database management services are usually in a

server that is a more powerful machine that services many clients. Processing management is split between the user system interface environment and the database management server environment. The database management server provides stored procedures and triggers. There are a number of software vendors that provide tools to simplify development of applications for the two tier client/server architecture.

In general, the user system interface client invokes services from the database management server. In many two tier designs, most of the application portion of processing is in the client environment. The database management server usually provides the portion of the processing related to accessing data (often implemented in store procedures). Clients commonly communicate with the server through SQL statements or a call-level interface. It should be noted that connectivity between tiers can be dynamically changed depending upon the user's request for data and services. [6]

2.3.5 Three – Tier Architecture

Just as a two-tier architecture separates GUI and business logic, a three-tier architecture allows you to separate business logic and data access. You can also provide highly optimized data indices and retrieval methods, and provide for replication, backup, redundancy, and load-balancing procedures specific to your data's needs. Separating code into client and server code increases the scalability of your application; so does placing data on a dedicated process, host, or series of hosts.

(Currently, SQL RDBMSs, like those from Oracle and Sybase, vastly outnumber other database types. You may have heard the names of some of these other types -- OODBs (object-oriented databases), ORDBs (object-relational databases), and embedded databases -- thrown around as buzzwords, but these are still exotic species, rarely encountered in the real world.)

The general procedure for using a database is to design a schema that describes your data, and queries that store and retrieve that data. There is one shortcoming to this approach: you need to learn a whole new programming language! SQL is not Java, and with the added effort of designing and implementing a new schema, translation code to go from one schema to the other, and queries executed by your program, your development time can greatly increase. When you throw stored procedures and the hiring of a full-time DBA into the mix, the decision to go with a database can seem unnatural. I've heard people estimate that as much as 70 percent of a given project's programming and debugging is spent in the object-relational mapping code.

Granted, there are many cases in which using a database is necessary, and reduces an application's time to market. But there are also many cases in which you can store a small amount of data in a simple local file instead of a relational database. A simple rule of thumb: if you only need to store data, and can get away with retrieving files by name, use a filesystem. If you also need to search through those data, then use a database -- especially if those searches are based on varying criteria.

One reason you can benefit from using a database -- besides the improvement in concurrency, access speed, and reliability -- is that multiple applications (or services) can access the same data. This benefit is on the border between three-tier and n-tier applications.

A word on stored procedures: they are evil. Stored procedures are essentially little programs that run inside the database. Since they are close to the data, they can perform manipulations (sorting, filtering, transforming, etc.) that would be prohibitively expensive to perform on the server. Some operations require that stored procedures or triggers be efficient. However, you can easily misuse them. It is tempting to put business logic inside stored procedures. All this constitutes a disruption of the three-tiered structure: instead of GUI, logic, and storage being neatly separated, you now have logic intermingling with storage, and logic on multiple tiers within the architecture -- causing potential headaches down the road if that logic has to change. Furthermore, the stored-procedure logic is written in a different language than the application code, and is subject to a different revision-control mechanism (if there is one, which there often isn't) and a different debugger mechanism (generally text-based and rudimentary, if it exists). This makes stored-procedure code much more difficult to develop and debug.

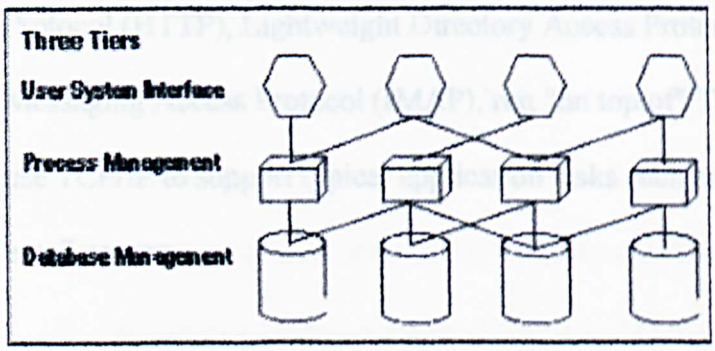


Figure 2.5: Three- Tier Architecture

2.4 Security Technology

Security in an important part of developing a system. Without a good security system, a website would be exposed to all kinds of security threats including hacking and cracking. As a result users would loose faith in the website resulting in lower hit counts.

2.4.1 Secure Sockets Layer(SSL)

Originally developed by Netscape, SSL has been universally accepted on the World Wide Web for authenticated and encrypted communication between clients and servers.

The Transmission Control Protocol/Internet Protocol (TCP/IP) governs the transport and routing of data over the Internet. Other protocols, such as the HyperText Transport

Protocol (HTTP), Lightweight Directory Access Protocol (LDAP), or Internet Messaging Access Protocol (IMAP), run "on top of" TCP/IP in the sense that they all use TCP/IP to support typical application tasks such as displaying web pages or running email servers.

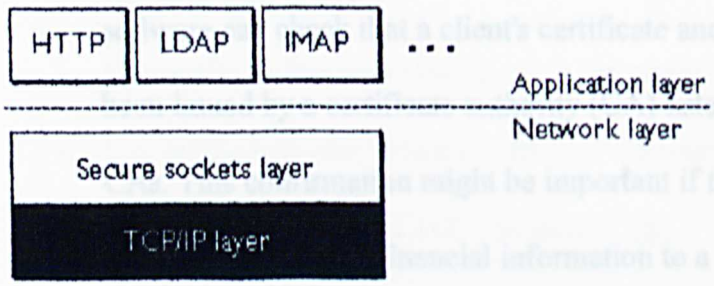


Figure 2.6: SSL runs above TCP/IP and below high-level application protocols

The SSL protocol runs above TCP/IP and below higher-level protocols such as HTTP or IMAP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to an SSL-enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

These capabilities address fundamental concerns about communication over the Internet and other TCP/IP networks:

- **SSL server authentication** allows a user to confirm a server's identity. SSL-enabled client software can use standard techniques of public-key cryptography

to check that a server's certificate and public ID are valid and have been issued by a certificate authority (CA) listed in the client's list of trusted CAs. This confirmation might be important if the user, for example, is sending a credit card number over the network and wants to check the receiving server's identity.

- **SSL client authentication** allows a server to confirm a user's identity. Using the same techniques as those used for server authentication, SSL-enabled server software can check that a client's certificate and public ID are valid and have been issued by a certificate authority (CA) listed in the server's list of trusted CAs. This confirmation might be important if the server, for example, is a bank sending confidential financial information to a customer and wants to check the recipient's identity.
- **An encrypted SSL connection** requires all information sent between a client and a server to be encrypted by the sending software and decrypted by the receiving software, thus providing a high degree of confidentiality. Confidentiality is important for both parties to any private transaction. In addition, all data sent over an encrypted SSL connection is protected with a mechanism for detecting tampering--that is, for automatically determining whether the data has been altered in transit.

The SSL protocol includes two sub-protocols: the SSL record protocol and the SSL handshake protocol. The SSL record protocol defines the format used to transmit data. The SSL handshake protocol involves using the SSL record protocol to exchange a

series of messages between an SSL-enabled server and an SSL-enabled client when they first establish an SSL connection. This exchange of messages is designed to facilitate the following actions:

- Authenticate the server to the client.
- Allow the client and server to select the cryptographic algorithms, or ciphers, that they both support.
- Optionally authenticate the client to the server.
- Use public-key encryption techniques to generate shared secrets.
- Establish an encrypted SSL connection. [8]

2.5 Web Server

AS web servers have increasingly become feature sets bundled with an operating system, the choice of operating system will ultimately depend on the selected platform. The universal compliancy of TCP/IP networking we can use cross platforms and servers although this might not be the best solution. The platform chosen for development should be one that is commonly used by the intended users and on we are familiar with.

The most common security measures supported by web servers is authentication, where the user will be required to provide authentication in order to access the system. Some servers go a step further implementing restricted access by IP addressors' host name. Other security measures include encryption, key servers and Virtual Private Network(VPN) access.

Web Servers use Secure Sockets Layer to support encryption than can be protected against unwanted access. All products handle security admirable well except for Apache, whose public-domain version does not support SSL. Today, some of the popular web servers are Apache, Microsoft Exchange Server, Lotus Domino and several others.

2.5.1 Apache

Apache remains the king of Web servers despite intense efforts by Microsoft and Netscape to gain dominance in the market. In fact, the latest Netcraft surveys indicate that the freeware Apache is widening its lead over the rest of the field. Apache users have come to rely on the server's rock-solid reliability, outstanding performance, and rich set of features. As its two closest competitors have found out, a brand name alone does not necessarily equate to a loyal customer following (Netscape SuiteSpot), nor does a plump pocketbook ensure market share (Microsoft IIS).

The keys to Apache's attractiveness and popularity lie instead in the qualities listed above and its extensibility, its freely distributed source code, and active user support for the server. And version 1.3.0, now in official release, is already being touted as the most stable and fastest version of Apache ever. When coupled with the fact that the server will now run on Windows NT and 95/98, Apache appears poised to make inroads on Microsoft's sacred soil as well.

Based originally on NCSA's freely available HTTPd server, Apache's features and strengths are too numerous to list. And if more than half of the Internet's Web sites use

Apache, the server must be doing something right, right? Among the most notable features are its cross-platform support, protocol support (HTTP/1.1), modularity (API), security, logging, and overall performance and robustness. Apache runs on Windows (95/98/NT), OS/2, and all the major variants of Unix. The server is fully compliant with HTTP/1.1 and supports API and ISAPI (NT). Apache distributes a core set of modules that handle everything from user authentication and cookies to typo correction in URLs. There are many other tried and true custom modules readily available as well.

Apache's overall security, performance, and robustness are unquestionable -- many of the most accessed sites in the world run Apache or Apache derivatives. Public distribution of the source code results in patches for the software being distributed quickly, and allowing public scrutiny helps ensure that security holes in the software are promptly caught and reported. As a result, Apache's large user base has allowed its developers to create a package that is extremely stable and secure and one that is also able to compete more effectively with commercial packages in terms of both raw speed and integrated features.

Despite all of its strengths, Apache certainly isn't for everybody. Setup and maintenance of the server are accomplished via command-line scripting tools. Unlike most popular commercial servers, Apache offers neither browser-based maintenance capabilities nor any GUI configuration/administration tools. This is an advantage for some developers, but for others it can translate into higher deployment and maintenance costs, especially if the site's administrators are unfamiliar with the fundamentals of the server. The lack of visuals, wizards, and/or browser-based administration tools may be enough to turn some users away. Furthermore, Apache's "user-driven" technical support via newsgroups may

not get the job done for more than a few developers. There are, however, several companies that do provide full commercial support -- for a price, of course.

The atypical development and marketing style of the Apache server have not precluded it from becoming the most popular world wide Web server on the Internet today. Apache's robust design and extensibility, coupled with its freeware status and the availability of its source code to the public, make Apache a good choice for enterprise-level Web sites and for individuals and workgroups that use UNIX or a combination of UNIX and NT platforms. While Netscape and Microsoft sustain their search for a chink in Apache's armor, the most popular server on the 'net continues to show that it can withstand the competition's best efforts and still reign supreme as the champion of Web servers.

Pros: 7 Price (freeware), 7 Performance and robustness, 7 Rock-solid reliability, 7 Security, 7 Support for the HTTP 1.1 protocol, 7 Extensibility, 7 Quick tech support via Usenet newsgroup, 7 Streamlined interface

Cons: 7 No Mac version available, 7 NT version is in its infancy (still lacks a number of the UNIX versions' performance enhancements), 7 Interface lacks wizards and graphical administration tools for facilitating configuration and administration tasks, 7 More extensive technical support requires the purchase of a third-party support contract [9]

Internet Information Server (IIS) v5.0

Microsoft's IIS is a gret Web server, which comes exclusively as part of the Windows 2000 Server operating system, contains many new features along with performance and reliability enhancements.

IIS v5.0 is good as both a first-time Web server for those familiar and comfortable with Windows operating systems, and a high-end server for hosting providers and large corporate installations. It handles the basics well and is better integrated in Windows than previous versions. IIS v5.0 also comes with performance and feature enhancements that will be attractive for mission-critical tasks.

The ideal computer to run IIS on is at least a 200 MHz Pentium with 128 MB of RAM. Organizations should plan on doubling the RAM and CPU speed if they intend to run Advanced Server's clustering, SQL or Transaction services on the same machine as the Web server. As with previous versions, IIS runs only on server editions of Windows 2000: Organizations planning to use Windows 2000 Professional, should get the stripped-down Peer Web Services version.[10]

2.6 Operating system

Operating System

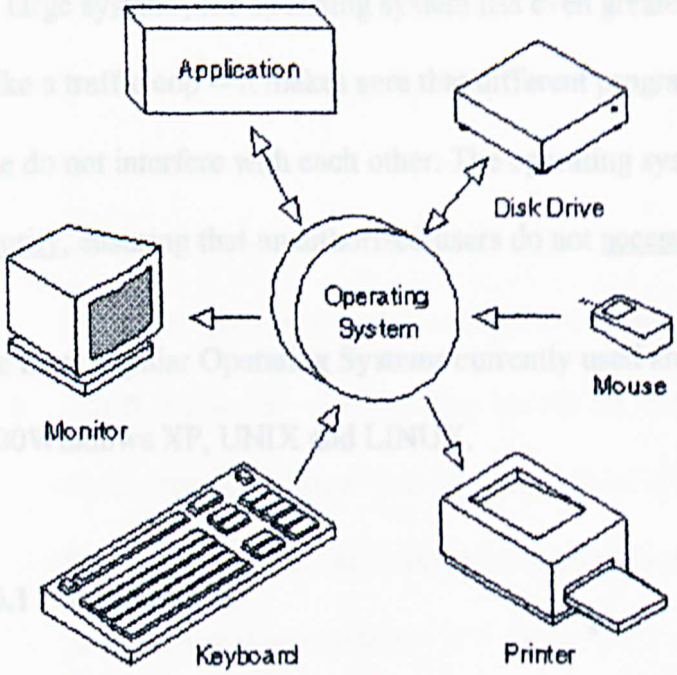


Figure 2.7: Structure of Operating System

Operating systems perform basic tasks, such as recognizing input from the keyboard, sending output to the display screen, keeping track of files and directories on the disk, and controlling peripheral devices such as disk drives and printers. [12]

Operating systems is essentially the body of the computer. Every general-purpose computers require some type of operating system that tells the computer how to operate and how to utilize other software and or hardware that is installed onto the computer.[13]

For large systems, the operating system has even greater responsibilities and powers. It is like a traffic cop -- it makes sure that different programs and users running at the same time do not interfere with each other. The operating system is also responsible for security, ensuring that unauthorized users do not access the system. [12]

The most popular Operating Systems currently used are Windows 98 SE, Windows 2000Windows XP, UNIX and LINUX.

2.6.1 Windows 98

Windows 98 offers support for a number of new technologies, including FAT32, AGP, MMX, USB, DVD, and ACPI. Its most visible feature, though, is the Active Desktop, which integrates the Web browser (Internet Explorer) with the operating system. From the user's point of view, there is no difference between accessing a document residing locally on the user's hard disk or on a Web server halfway around the world. [14]

2.6.2 Windows 2000

Windows 2000 is a product of Microsoft's Windows line of operating systems. Often described as the more stable among others the NT based technology has made it a rather preferred OS for developers working with Microsoft products. There are 4 version of Windows 2000:

- Professional – an operating system for business notebooks and desktops. It is used to run software applications, connect to the internet and intranet, access files, printers and other network resources.
- Server – both a web server and an office server. Windows 2000 Server lets users build Web applications and connect to the internet.
- Advanced Server – an operating system for line-of-business applications and e – commerce. It contains all the functionalities of the standard Windows 2000 Server, plus additional features for applications that require higher levels of scalability and availability.
- Data Center Server – developed to work in high-traffic computer networks, it is designed for enterprises that need reliable high-end drivers and software. It supports up to 32- way SMP and up to 64 GB of physical memory.

2.6.3 UNIX

The UNIX* operating system was designed to let a number of programmers access the computer at the same time and share its resources.

The operating system coordinates the use of the computer's resources, allowing one person, for example, to run a spell check program while another creates a document, lets another edit a document while another creates graphics, and lets another user format a document -- all at the same time, with each user oblivious to the activities of the others.

The operating system controls all of the commands from all of the keyboards and all of the data being generated, and permits each user to believe he or she is the only person working on the computer.

This real-time sharing of resources make UNIX one of the most powerful operating systems ever.

Although UNIX was developed by programmers for programmers, it provides an environment so powerful and flexible that it is found in businesses, sciences, academia, and industry. Many telecommunications switches and transmission systems also are controlled by administration and maintenance systems based on UNIX.

While initially designed for medium-sized minicomputers, the operating system was soon moved to larger, more powerful mainframe computers. As personal computers grew in popularity, versions of UNIX found their way into these boxes, and a number of companies produce UNIX-based machines for the scientific and programming communities. [15]

The features that made UNIX a hit from the start are:

- Multitasking capability
- Multiuser capability
- Portability
- UNIX programs
- Library of application software

Due to its portability, flexibility and power, UNIX has become the leading operating system for workstations. Historically, it has been less popular in the personal computer market, but the emergence of a LINUX has contributed to revitalising UNIX across all platforms.

2.6.4 Linux

Linux is a free (GPL Licensed), from scratch operating system based heavily on the POSIX and UNIX API's. It supports both 32 and 64 bit hardware and provides a stable multiuser internet ready operating system.

Linux itself is not Unix, although many people call it that and you would be very hard pushed to tell the difference. This is because the Unix trademark is specific to systems that meet a complex set of X/Open standards and has a cost. Some Linux vendors however are working on "Unix" branding.

Linux uses internet and industry standard components and protocols giving a system with complete network integration. The operating system can act as a server for most major file serving protocols, and provide all the major internet applications. The X window system provides a networked and platform independant graphical interface that (unlike proprietary user interfaces) allows one desktop to access applications running on multiple machines across local and wide area networks.

Linux is normally obtained as a "distribution". This is a combination of the Linux operating system kernel and other tools, utilities and applications. Some of these are available for free over the internet, and others on CD-ROM. Because Linux itself is free software that can be freely copied, many distributions are available both over the internet and sold on CD-ROM with added convenience and support. [16]

2.7 Database Server

Modern Day websites seem to be relying more and morea on complex database systems. These systems store all critical data, and allow for easy maintenance. Among the database servers available are Oracle, MySQL and Microsoft SQL Server.

2.7.1 MySQL

MySQL is a small, compact database server ideal for small - and not so small - applications. In addition to supporting standard SQL (ANSI), it compiles on a number of platforms and has multithreading abilities on Unix servers, which make for great performance. For non-Unix people, MySQL can be run as a service on Windows NT and as a normal process in Windows 95/98 machines.

In addition to being free (MySQL does have some licensing restrictions though), the PHP-MySQL combination is also cross-platform, which means you can develop in

Windows and serve on a Unix platform. Also, PHP can be run as an external CGI process, a stand-alone script interpreter, or an embedded Apache module. [18]

Some of the features of MySQL include:

- Handles large databases, in the area of 50,000,000 + records
- No memory leaks. Tested with a commercial memory leakage detector (purify).
- A privilege and password system which is very flexible and secure which allows host-based verification. Passwords are secure since all password traffic when connecting to a server is encrypted.

2.7.2 Microsoft SQL Server

Microsoft SQL Server 2000 Windows® CE Edition (SQL Server CE) version 2.0 is the compact database for rapidly developing applications that extend enterprise data management capabilities to mobile devices. SQL Server CE is a powerful tool that makes it easy to develop mobile applications by supporting familiar Structured Query Language (SQL) syntax and providing a development model and API consistent with SQL Server.

The SQL Server CE engine exposes an essential set of relational database features, such as an optimizing query processor and support for transactions and assorted data types, while maintaining a compact footprint that preserves precious system resources. Remote data access and merge replication ensure that data from SQL Server databases is

delivered reliably, can be manipulated offline, and can be synchronized later to the server, making SQL Server CE ideal for mobile and wireless environments.

SQL Server CE 2.0 is designed to integrate with the Microsoft .NET Compact Framework* by means of Microsoft Visual Studio® .NET, simplifying database application development for smart devices. Using the new SQL Server CE data provider to manage code by means of the Common Language Runtime, mobile application developers can build highly extensible applications with offline data management capability for disconnected scenarios.

SQL Server CE extends the frontier of data management by delivering:

- A familiar database platform for rapid development. The SQL Server family provides data management support and programmability across the enterprise from the largest servers to desktop workstations. SQL Server CE provides robust data management capabilities on mobile devices. By exposing a programming and operational model consistent with the rest of the SQL Server family, SQL Server CE ensures that organizations can easily integrate with existing systems and take advantage of existing development skills.
- A compact yet capable relational database. Though devices are advancing rapidly, system resources such as available memory are often scarce, so it is critical that a relational database system be as compact as possible while still exposing essential functionality. SQL Server CE has a small memory footprint, delivering all of its functionality in approximately 1 megabyte (MB). Performance is enhanced with an optimizing query processor. A range of data

types is supported to ensure flexibility, and 128-bit encryption is provided on the device for database file security.

- Flexible data access. SQL Server CE enables straightforward, efficient access to enterprise data whether a device is always connected or intermittently connected to the computer running SQL Server. Remote data access exposes data in SQL Server 6.5, SQL Server 7.0, and SQL Server 2000 databases through remote execution of Transact-SQL statements and the ability to pull record sets to the client device for updating. When used with SQL Server 2000, SQL Server CE provides extended capabilities for synchronization through merge replication. Both of these data access technologies take advantage of Internet standards, including HTTP Secure Sockets Layer (SSL) encryption, through integration with Internet Information Services (IIS). This approach ensures data can be accessed reliably and flexibly, even through firewalls.

2.8 Data Access Technology

2.8.1 Open Database Connectivity (ODBC)

ODBC stands for Open Database Connectivity, and is an open standard API for accessing databases. ODBC can access information in a variety of different databases by converting the request to a form of SQL that the respective databases can understand.

ODBC is an open standard application programming interface (API) for accessing a database. By using ODBC statements in a program, you can access files in a number of

different databases, including Access, dBase, DB2, Excel, and Text. In addition to the ODBC software, a separate module or driver is needed for each database to be accessed. The main proponent and supplier of ODBC programming support is Microsoft.

ODBC is based on and closely aligned with The Open Group standard Structured Query Language (SQL) Call-Level Interface. It allows programs to use SQL requests that will access databases without having to know the proprietary interfaces to the databases. ODBC handles the SQL request and converts it into a request the individual database system understands.

ODBC was created by the SQL Access Group and first released in September, 1992. Although Microsoft Windows was the first to provide an ODBC product, versions now exist for UNIX, OS/2, and Macintosh platforms as well.

In the newer distributed object architecture called Common Object Request Broker Architecture (CORBA), the Persistent Object Service (POS) is a superset of both the Call-Level Interface and ODBC. When writing programs in the Java language and using the Java Database Connectivity (JDBC) application program interface, you can use a product that includes a JDBC-ODBC "bridge" program to reach ODBC-accessible databases. [20]

2.8.2 ActiveX Data Objects (ADO)

ActiveX Data Objects (ADO) is an application program interface from Microsoft that lets a programmer writing Windows applications get access to a relational or non-relational database from both Microsoft and other database providers. For example, if you wanted to write a program that would provide users of your Web site with data from an IBM DB2 database or an Oracle database, you could include ADO program statements in an HTML file that you then identified as an Active Server Page. Then, when a user requested the page from the Web site, the page sent back would include appropriate data from a database, obtained using ADO code.

Like Microsoft's other system interfaces, ADO is an object-oriented programming interface. It is also part of an overall data access strategy from Microsoft called Universal Data Access. Microsoft says that rather than trying to build a universal database as IBM and Oracle have suggested, finding a way to provide universal access to various kinds of existing and future databases is a more practical solution. In order for this to work, Microsoft and other database companies provide a "bridge" program between the database and Microsoft's OLE DB, the low-level interface to databases.

OLE DB is the underlying system service that a programmer using ADO is actually using. A feature of ADO, Remote Data Service, supports "data-aware" ActiveX controls in Web pages and efficient client-side caches. As part of ActiveX, ADO is also part of Microsoft's overall Component Object Model (COM), its component-oriented framework for putting programs together.

ADO evolved from an earlier Microsoft data interface, Remote Data Objects (RDO). RDO works with Microsoft's ODBC to access relational databases, but not nonrelational databases such as IBM's ISAM and VSAM. [23]

2.8.3 OLE DB

OLE DB is Microsoft's strategic low-level application program interface (API) for access to different data sources. OLE DB includes not only the Structured Query Language (SQL) capabilities of the Microsoft-sponsored standard data interface Open Database Connectivity (ODBC) but also includes access to data other than SQL data. As a design from Microsoft's Component Object Model (COM), OLE DB is a set of methods (in earlier days, these might have been called routines) for reading and writing data. The objects in OLE DB consist mainly of a data source object, a session object, a command object, and a rowset object. An application using OLE DB would use this request sequence:

- Initialize OLE.
- Connect to a data source.
- Issue a command.
- Process the results.
- Release the data source object and uninitialize OLE.

OLE once stood for "Object Link Embedding" and "DB" for database. However, Microsoft no longer ascribes these meanings to the letters "OLE" and "DB." [24]

2.9 Language

2.9.1 ASP

An Active Server Page (ASP) is an HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server, usually tailoring a page for the user. Typically, the script in the Web page at the server uses input received as the result of the user's request for the page to access data from a database and then builds or customizes the page on the fly before sending it to the requestor.

ASP is a feature of the Microsoft Internet Information Server (IIS), but, since the server-side script is just building a regular HTML page, it can be delivered to almost any browser. You can create an ASP file by including a script written in VBScript or JScript in an HTML file or by using ActiveX Data Objects (ADOs) program statements in the HTML file. You name the HTML file with the ".asp" file suffix. Microsoft recommends the use of the server-side ASP rather than a client-side script, where there is actually a choice, because the server-side script will result in an easily displayable HTML page.

Client-side scripts (for example, with JavaScript) may not work as intended on older browsers. [23]

For Web service applications, Microsoft provides a new version of ASP support called ASP.NET.

2.9.2 JSP

JavaServer Pages technology allows web developers and designers to rapidly develop and easily maintain, information-rich, dynamic web pages that leverage existing business systems. As part of the Java family, JSP technology enables rapid development of web-based applications that are platform independent. JavaServer Pages technology separates the user interface from content generation enabling designers to change the overall page layout without altering the underlying dynamic content.

JavaServer Pages technology uses XML-like tags that encapsulate the logic that generates the content for the page. Additionally, the application logic can reside in server-based resources (such as JavaBeans component architecture) that the page accesses with these tags. Any and all formatting (HTML or XML) tags are passed directly back to the response page. By separating the page logic from its design and display and supporting a reusable component-based design, JSP technology makes it faster and easier than ever to build Web-based applications.

JavaServer Pages technology is an extension of the Java Servlet technology. Servlets are platform-independent, 100% pure Java server-side modules that fit seamlessly into a Web server framework and can be used to extend the capabilities of a Web server with minimal overhead, maintenance, and support. Unlike other scripting languages, servlets involve no platform-specific consideration or modifications; they are Java application components that are downloaded, on demand, to the part of the system that needs them. Together, JSP technology and servlets provide an attractive alternative to other types of dynamic Web scripting/programming that offers platform independence, enhanced performance, separation of logic from display, ease of administration, extensibility into the enterprise and most importantly, ease of use.

Today, servlets are a popular choice for building interactive Web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually a component of Web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun ONE Web Server, Sun ONE Application Server, and others.

The JSP specification is the product of industry-wide collaboration with industry leaders in the enterprise software and tools markets, led by Sun Microsystems. Sun has made the JSP specification freely available to the development community, with the goal that every Web server and application server will support the JSP interface. JSP pages share the "Write Once, Run Anywhere" characteristics of Java technology. JSP technology is a key component in the Java 2 Platform, Enterprise Edition, Sun's highly scalable architecture for enterprise applications.[21]

2.9.3 PHP

In Web programming, PHP is a script language and interpreter that is freely available and used primarily on Linux Web servers. PHP, originally derived from Personal Home Page Tools, now stands for PHP: Hypertext Preprocessor, which the PHP FAQ describes as a "recursive acronym."

PHP is an alternative to Microsoft's Active Server Page (ASP) technology. As with ASP, the PHP script is embedded within a Web page along with its HTML. Before the page is sent to a user that has requested it, the Web server calls PHP to interpret and perform the operations called for in the PHP script.

An HTML page that includes a PHP script is typically given a file name suffix of ".php" ".php3," or ".phtml". Like ASP, PHP can be thought of as "dynamic HTML pages," since content will vary based on the results of interpreting the script.

PHP is free and offered under an open source license.

2.9.4 ASP.NET

ASP.NET (originally called ASP+) is the next generation of Microsoft's Active Server Page (ASP), a feature of their Internet Information Server (IIS). Both ASP and ASP.NET allow a Web site builder to dynamically build Web pages on the fly by inserting queries to a relational database in the Web page. ASP.NET is different than its predecessor in two major ways: it supports code written in compiled languages such as

Visual Basic, C++, C#, and Perl, and it features server controls that can separate the code from the content, allowing WYSIWYG editing of pages. Although ASP.NET is not backwards compatible with ASP, it is able to run side by side with ASP applications. ASP.NET files can be recognized by their .aspx extension. [25]

In a way, ASP.Net is a much more stable language as compared to ASP as unlike ASP (which is an interpretation language) it is compiled. In a way you could say it uses the Java concept of 'compile once run anywhere'. ASP on the other hand is interpreted every time it is refreshed by the browser. This increases the probabilities for errors despite the fact the ASP page might no have had any errors earlier.

ASP.Net supports several languages and is mostly used with VB.NET and C#. It also support JavaScript. The downside is it does not support VBScript so ASP developers might encounter slight difficulties when changing over to ASP.Net.

CHAPTER 3 : METHODOLOGY

Chapter 3: Methodology

3.1 Project Development Life Cycle

In order to ensure a systematic and organized approach is deployed in developing a system, it is necessary to follow a sequence of steps. This sequence of steps accommodate a complete set of tasks generally referred to as a process. This process contains a series of steps involving activities, constraints, and resources that contribute to producing the intended output for the user. Thus, this process often includes a series of tools and techniques.

When it comes to product development, the process is referred to as a life cycle. Therefore the development process of the Personal Library System is defined as a Development Life Cycle, as it describes the development stages of the system from its conceptualization to its implementation, to its use and maintenance.

3.2 Software Process Model

The software process model used to develop this system is based on the Evolutionary Model as shown in Figure 3.1. Evolutionary development is based on the idea of developing an initial implementation, exposing this to user comment and refining this through many version until the adequate system has been developed. Rather than have

separate specifications, development and validation activities, these are carried out concurrently with rapid feedback across these activities.

There are 2 types of evolutionary development:

- Exploratory Development – where the objective of the system is to work with the customer to explore their requirements and deliver a final system. The development starts with parts of the system which are understood. The system evolves by adding new features as they are proposed by the customer. (In the context of this system as the users are the lecturers of FCSIT, the system wil be subjected to continuous evaluation by the project supervisor.)
- Throw-away Prototyping – this type concentrates on experimenting with the parts of the system that are poorly understood by the customer. As the Personal Library Project is not all that complicated this will not be the feasible type.

Each part of the model is essential in developing the system. The most obvious processes though is the System Design, Prototype Development, Prototype Evaluation and Prototype Enhancement. Based on the initial system design a prototype of the system is developed. Once this is complete the working prototype will be evaluated and tested for errors and other faults.

Based on the evaluation, changes or a complete new design will be implemented and the prototype will go through the former processes again. These steps are repeated until and adequate system is developed.

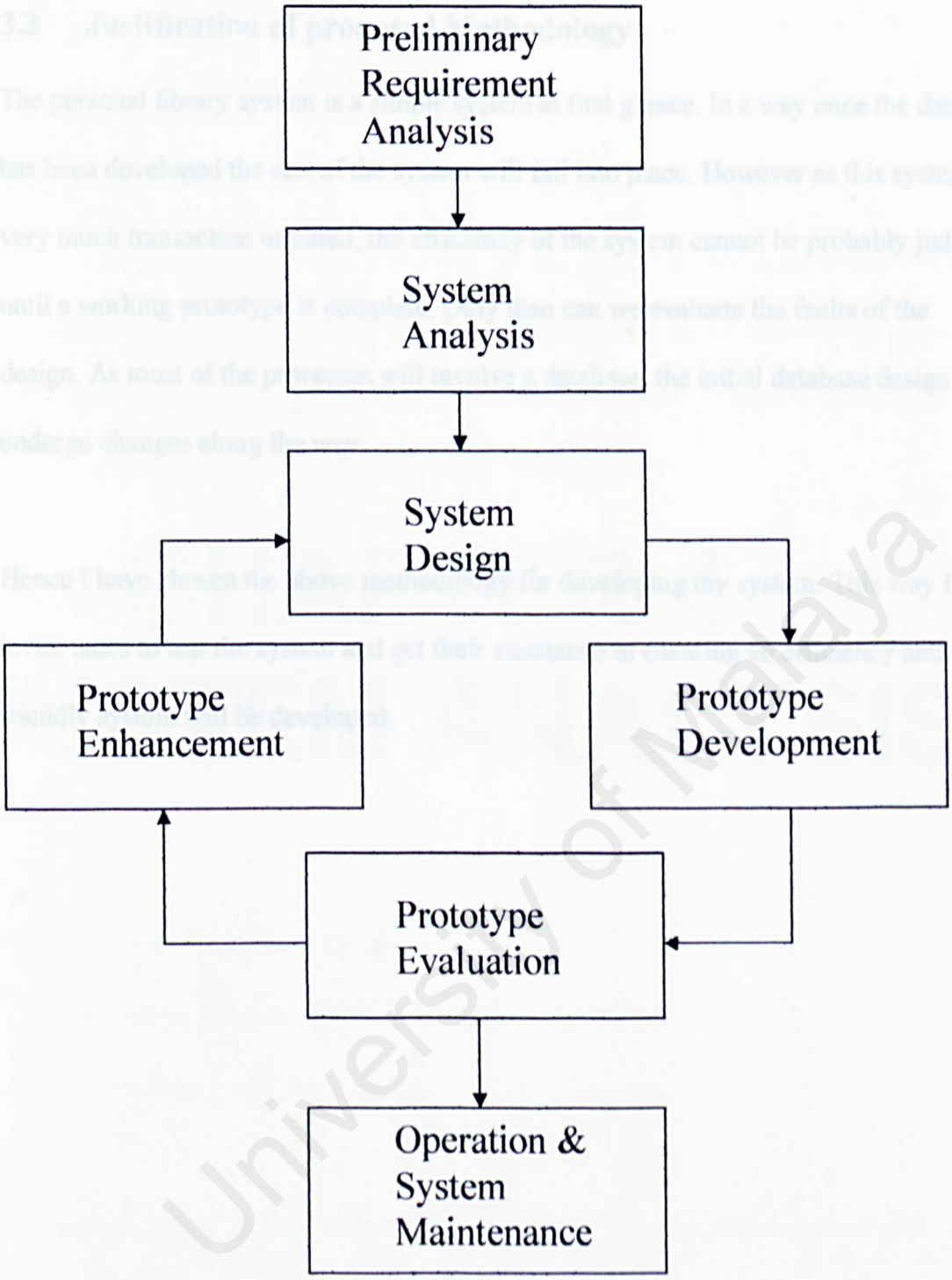


Figure 3.1: Evolutionary Model

3.3 Justification of proposed Methodology

The personal library system is a simple system at first glance. In a way once the database has been developed the rest of the system will fall into place. However as this system is very much transaction oriented, the efficiency of the system cannot be probably judged until a working prototype is complete. Only then can we evaluate the faults of the design. As most of the processes will involve a database, the initial database design will undergo changes along the way.

Hence I have chosen the above methodology for developing my system. This way I can invite users to test the system and get their assistance in ensuring an efficiency and user-friendly system will be developed.

CHAPTER 4 : SYSTEM ANALYSIS

Chapter 4: System Requirements Analysis

4.1 Requirement Analysis

The requirement analysis is often the most important part of developing a foundation for a system. Analyzing the requirement of the systems based on the user requirements and other observations will help the developer define the functional requirements of the system. This will indirectly contribute to the system design.

The requirements define the 'What' of the system and design identifies the 'How' of the system. This enables the developer to acquire an explanation of the system and its logic. In a way the Requirement Analysis represents an understanding between the customer and the developer.

The requirement analysis covers 2 parts:

- Functional Requirements
- Non- Functional Requirements

Other requirements include the necessary hardware, software, development tools, web server and database server.

4.1.1 Functional Requirements

Functional requirements are the developers interpretation of functions or parts of the system based on the user requirements. More often than not users don't really understand the functional requirements.

Functional Requirement Modules:

1. Data Manipulation and Control

   Data owned by users can be editted, deleted and manipulated by the user. However users can only do so to data owned by them. The ownership of data is defined is the Access Levels Module

2. User Access Levels

   There are a number of access levels that define the level of access to the data. The lowest access levels are the students and outsiders. These users can browse through the data available. The next level are the lecturers who can manipulate data owned by them. As this is a personal library system, these users have exclusive access to the system and may remove users , data or limit options of users as they see fit provided it's owned by them.

3. Search Module

   This module is the most important module from the users' point of view. The search option will enable users to search for relevant data and references from

the data contained within the database. However the search results will be limited to the data contained within the database.

4. Resources Module

   This module is basically is the data entry module for resources. There are 3 general groups that have been indetified. They are media, equipment and books. Users may define the status of the resource as to wether it can be borrowed others or not.

5. Transaction Module

   As a follow up of the previous module, lecturers can insert transaction information about the borrowed resources using this module. Information such as name of person, contacts, date borrowed and returned date will be stored enabling the lecturer to keep a records of all transactions.

4.1.2 Non-Functional Requirements

The non-functional requirements are a description of features, characteristics and attributes of the system including its constraints and limitations. Non-functional requirements are defined as the constraints under which the system must operate and the standard that should be met by the delivered product. These requirements are very subjective but are as essential as the functional requirements. The non-functional requirements are listed below.

Usability

- The layout of the system should be constant to ensure the system is user – friendly.
- Appropriate error messages will be used in the event of errors.
- Users should get an instant view of data entered into database to ensure accuracy.
- The system should avoid from including jargon that might confuse the users. Should the jargon be used necessary explanation should be provided.
- A standard side menu for easy maneuvering throughout system should be included.
- Certain buttons like exit and reset should be avalaible at all forms.
- As this is an online system it should be designed as such that it will be viewaqble through all commonly used browsers.

Reliability

They system should be reliable and stable to ensure ease of use. The software used and the web server should be stable and not cause any unpleasant down time pf the overall environment. Should the system be unstable users will avoid using the system.

Security

All administrators will be provided with necessary access to the system. This access will require authentication ensuring that the data is safe and not accessible by other users. Users without authentication will only be able to browse the system.

Performance

The system will be stable and able to accommodate the usage. By deploying server-based database systems and using stable database connection the system should ensure that the performance is at par with the users needs.

Scalability

The system should be designed allowing for future upgrades and integration. Here the platforms chosen also lay a role in ensuring that the chosen platforms, data servers and language are up to date with the current trends. This will enable future upgrades at a lower cost.

4.1.3 Techniques Used to Define Requirements

- **Internet Research**

  The internet was the main source of information for me. Using search engines such as Google, Yahoo and Alltheweb I browse the website searching for similar digital libraries. Through these search results I found my 3 case studies and a lot of other information that helped me in defining the requirements of the system.

- **Discussion with Lecturers and Peers**

  As the target users of the system are the lecturers and peers of my faculty they played a major role in defining the requirements. Discussions followed by me explaining the system to them assisted me in realizing the final requirements of the system.

- **Observation**

  Through many observations of existing online database systems and designs I was able to identify the necessary modules required by the system. These observations assisted me in identifying the limitations and ensuring the functionalities of the system were as necessary.

4.2 Chosen Platform, Web Server, Database, Language and Tools

The choices of the products below were based on the Literature Review done in Chapter 2: Review of Literature. Below are the choices made and brief justification on the reasons for these choices.

4.2.1 Choice of Software Architecture

The chosen architecture for this system is the Client-Server. As this is web based system this model is ideal and the simplest. The client will basically send requests to the server and the server will respond to the requests as it sees fit. This system will also reduce network traffic by providing a query response rather than a whole file transfer. As this systems mostly involves querying the database this architecture is ideal.

4.2.2 Choice of Development Platform

Windows 2000 professional is my choice for the operating system. This operating system is among the most efficient of all Widows operating systems and fully supports IIS. It also supports most database servers. Through the Internet Service Manager I would be able to perform meticulous testing of the system ensuring it is indeed efficient.

Windows 2000 also supports the .NET framework and most development tool and application software.

4.2.3 Choice of Database Management System

The chosen database for the development of this system is Microsoft SQL Server 2000. This database management system is server based and is among the better online database systems around. It also runs very well with Windows 2000 Professional making it the natural choice.

Some advantages of Microsoft SQL Server 2000 are:

- Convenient construction of relational database with a proper view of the actual design.
- Query analyzer which enables developers to test out their SQL statements before implementing them
- With the use of its components like View option, stored procedures can be created
- Supports OLAP
- Able to support multiple users and multiple transactions
- Very stable as compared to several other database management systems that often decline in performance after 1000 entries.

4.2.4 Choice of Data Access Technology

ActiveX Data Objects (ADO) have become the latest in data access technology taking over it's predecessor Remote Data Objects (RDO). As this technology is supported by most web servers and is quite efficient it is the chosen technology for database connection. With the use of OLEDB (also known as DSN-less connection or Connection String) database connections will be stable and fast. It is also supported by most web servers as most of the commonly used web development languages.

4.2.5 Choice of Web Development Server

The choice of web development server is Microsoft Internet Information Server 5.0(IIS 5.0). This web server is fully supported by Windows 2000 professional and is in fact and add-on component of Windows 2000. The IIS 5.0 is an upgraded version of IIS 4.0 that was designed for Windows 98. It has several upgrades and new components such as the Internet Services Manager. As this web server works very well with the chosen operating system it became the natural choice.

Despite Apache being the King of the Web Servers IIS 5.0 has been rated as efficient as Apache as fully supports the .NET Framework. As .NET is possibly the future tool to be used for web development IIS 5.0 is a much better choice compared to Apache.

4.2.6 Choice of Web Development Tools

In order to ensure the scalability of the system it is important to develop the system using the latest tools. Therefore the chosen development tool is ASP.Net. This language which is part of the .NET framework is an upgrade of Active Server Pages 3.0(ASP) and has overcome many of the faults of ASP. It also supports several languages and scripting like VB.Net, C#, JavaScript and JScript. The software used to write the code is Microsoft Vidual Studio.Net Architect Edition.

4.3 System Requirements

4.3.1 Server hardware Requirements

The minimum hardware requirements of the server:

- Pentium 166 MHz processor
- 64MB RAM
- Network interface card(NIC) and network connection with recommended bandwidth at 10Mbps or more
- Other standard computer periphals

4.3.2 Server Software Requirements

The server machine requires at least the following software in order to host and run the system:

- Operating System: Microsoft Windows 2000 Professional

- Web Server: Microsoft Internet Information Server 5.0
- Database: Microsoft SQL Server 7.0
- Components that support the .NET framework

4.3.3 Client Hardware Requirements

The client hardware requirements are quite minimal as long as it has the necessary specifications that support internet access. The minimum hardware requirements are:

- 486 Processor (Although Pentium class processors are preferred)
- 16 MB RAM
- Network connection either through existing Local Area Networks or dial-up.

4.3.4 Client Software Requirements

The client requirements are:

- Operating system that supports internet browsers such as Windows 95/98/NT/2000, UNIX/Linux.
- Web browser: Internet Explorer 5.0 or other web browsers of similar capabilities.

CHAPTER 5 : SYSTEM DESIGN

Chapter 5: System Design

5.1 Overview

The system design is an important stage of the system development. Here the requirements and specifications are translated into system characteristics to meet the user requirements. It is a creative process of transforming problems into solutions. The system design includes complete descriptions of the functions and user interaction involved. Although the system design describes only the appearance of the system, this stage is important in determining the success of a software project.

In this phase the topics that will be covered are:

- System Architecture Design
- System Functionality design
- Database Design
- User Interface Design

5.2 System Architecture Design

The system architecture of the Personal Library System is chosen based on the scope and complexity of this project. In general this system functions using Client – Server architecture. To be more specific the system implements the Thin Client model where the server is responsible for data management. The software on the client implements the application logic and the interactions with the system users.

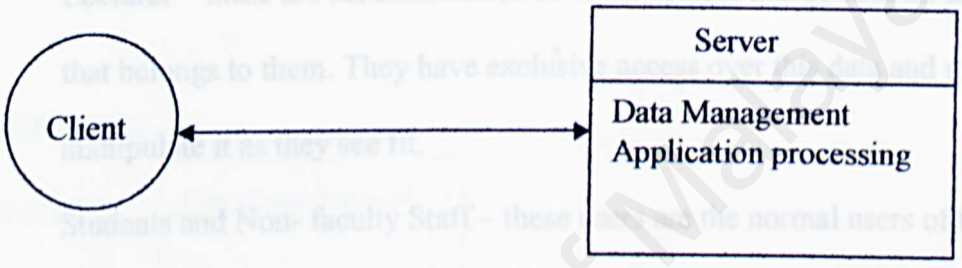


Figure 5.1: Thin Client Model

Basically the system is centralised and completely dependant on the server. All processing is centralised making the control of the system easier. This is why it is necessary that the database management system is server based in order to support the system.

5.3 System Functionality Design

In this stage I would like to translate the functional requirements on the systems into diagrams making it easier to identify the users and understand the general flow of the system. The system functionality design transforms all requirements into an organised picture of the system functionalities and data flow diagrams.

5.3.1 System Structure

The general structure of the system is relatively simple. There are only 2 users defined:

1. Lecturer – these are the administrators of the system and control all the data that belongs to them. They have exclusive access over this data and may manipulate it as they see fit.
2. Students and Non- faculty Staff – these users are the normal users of the system. They are only allowed to browse the system. Should they wish to borrow any of the resources listed on the system they would have to approach the owner of the resource directly.

As the data flow diagram section explains the system flow in detail I will not dwell any further in this section.

5.3.2 Data Flow Diagram

In the process design, the Personal Library system is designed based on the data flow diagram(DFD). DFDs are a graphical method used to depict the flow of data through a system. It gives and overview of the systems input/output and show the flow of data from the source entities to the processes and from the processes to the destination entities.

Below is the context diagram that is used to show the overall flow of the system. The boxes with a shadow are the users and the box in the middle with the circular edges is the Personal Library System. (Figure 5.2)

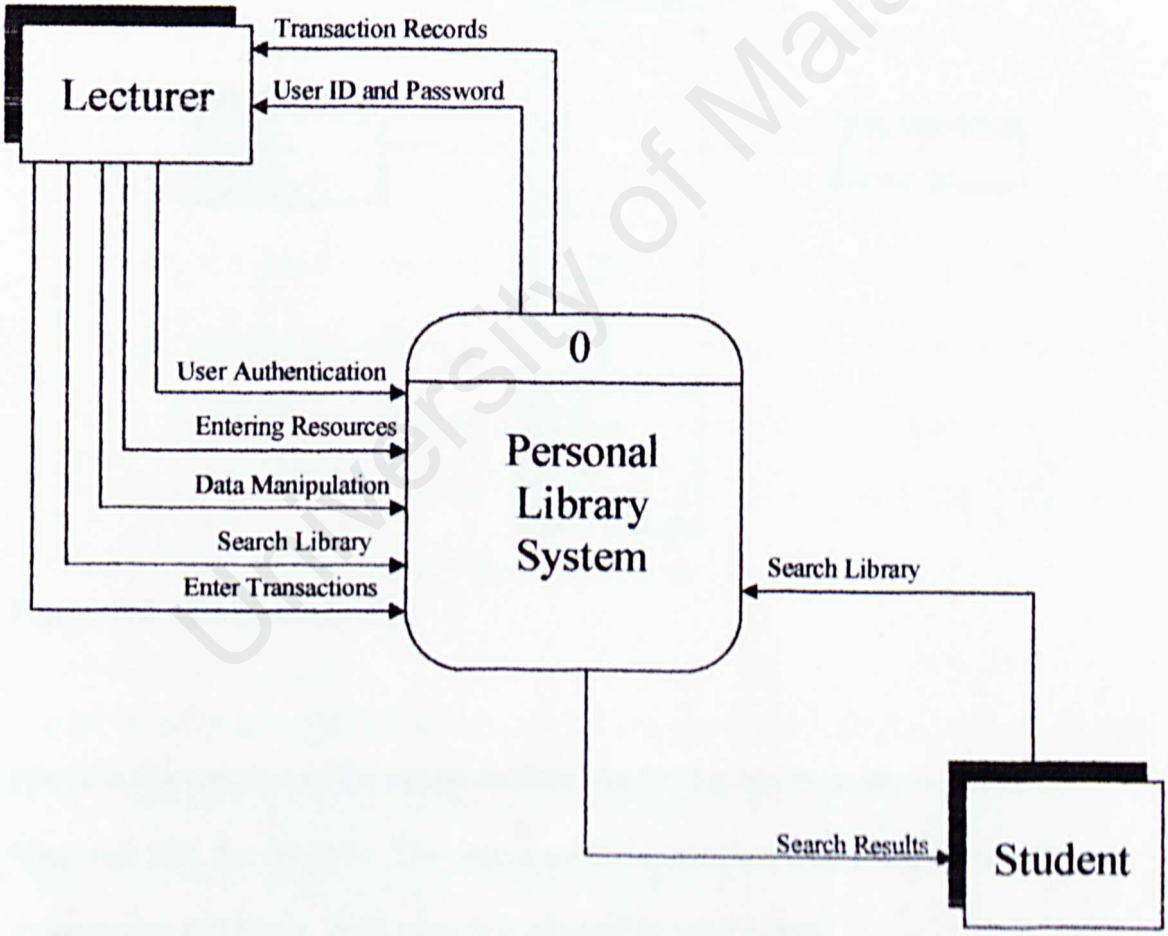


Figure 5.2 :Context Diagram

Here we can see the implementation of the functional requirements. From here we can see how simple the system design is. This simplicity greatly reduces the possibility for errors and misjudgements. The arrows in the diagram shows the flow of data to and from the system. For example when searching the library, data is sent to the system.

Though the system looks simple from here, the database design is not as such. In order to ensure the simplicity of the system the database design has to be accurate and fully support the dynamicity of the design.

5.4 Database Design

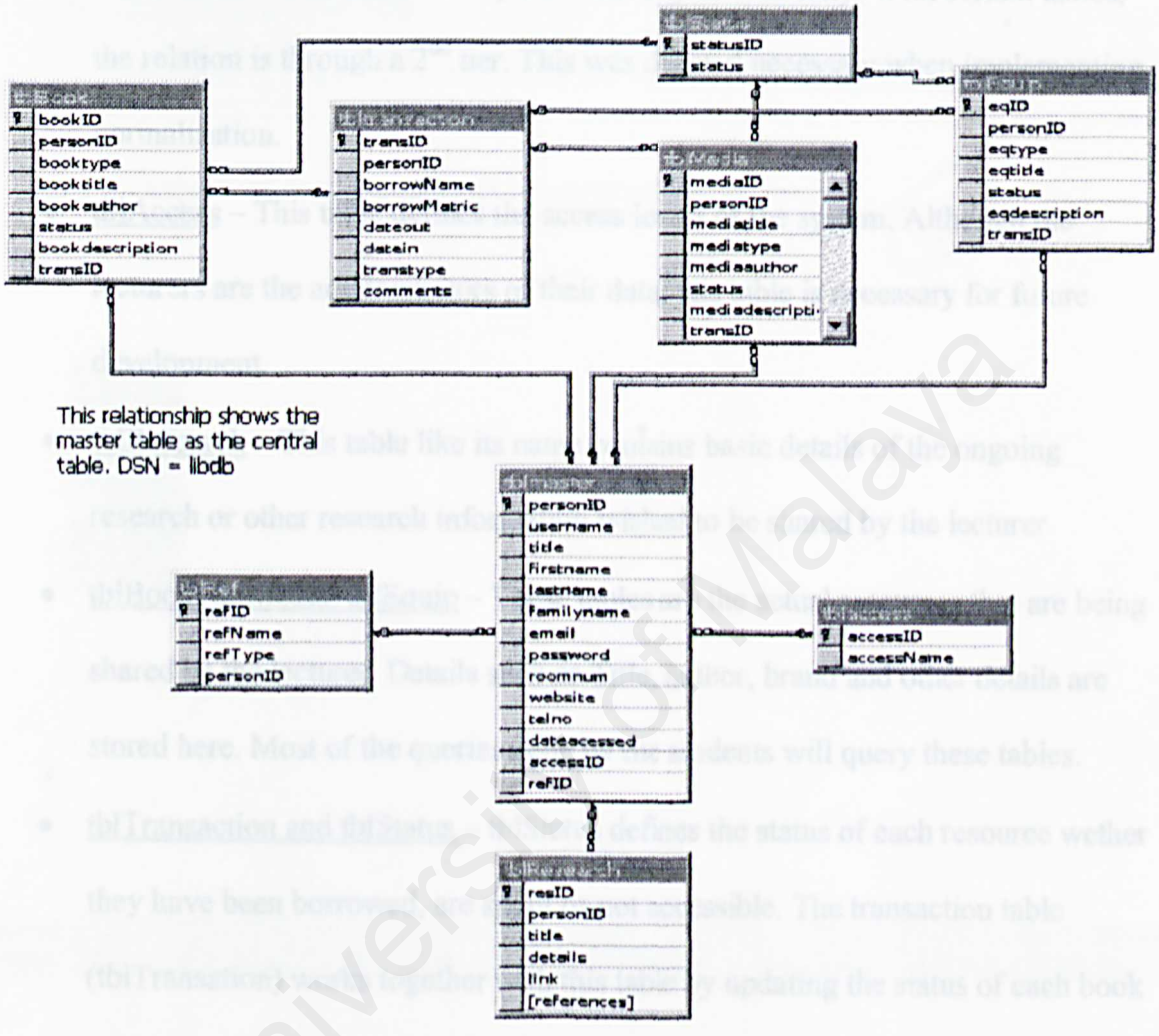


Figure 5. 3 :Database Design

Above is a screenshot of the actual database design that has been implemented in Microsoft SQL Server 2000. The design used is centralised where nearly everything is connected to tblMaster. Each table is explained in detail below.

- tblMaster – This table contains the basic information of the lecturers including the foreign keys of the other tables. The primary key here is personID. This table has a relation with almost every table directly or indirectly. With certain tables, the relation is through a 2nd tier. This was deemed necessary when implementing normalisation.
- tblAccess – This table defines the access levels of the system. Although the lecturers are the administrators of their data, this table is necessary for future development.
- tblResearch – This table like its name contains basic details of the ongoing research or other research information wished to be shared by the lecturer
- tblBook, tblMedia, tblEquip – These tables are the actual resources that are being shared be the lecturer. Details such as Title, author, brand and other details are stored here. Most of the queries done by the students will query these tables.
- tblTransaction and tblStatus – tblStatus defines the status of each resource wether they have been borrowed, are spoilt or not accessible. The transaction table (tblTransation) works together with this table by updating the status of each book every time it is borrowed. Hence the relationship.

Each table has its primary key as shown in the diagram and this often acts as the foreign key in a relationship. As the database design shows the actual implementation of the design I will not be defining details of each field in the table.

5.5 User Interface Design

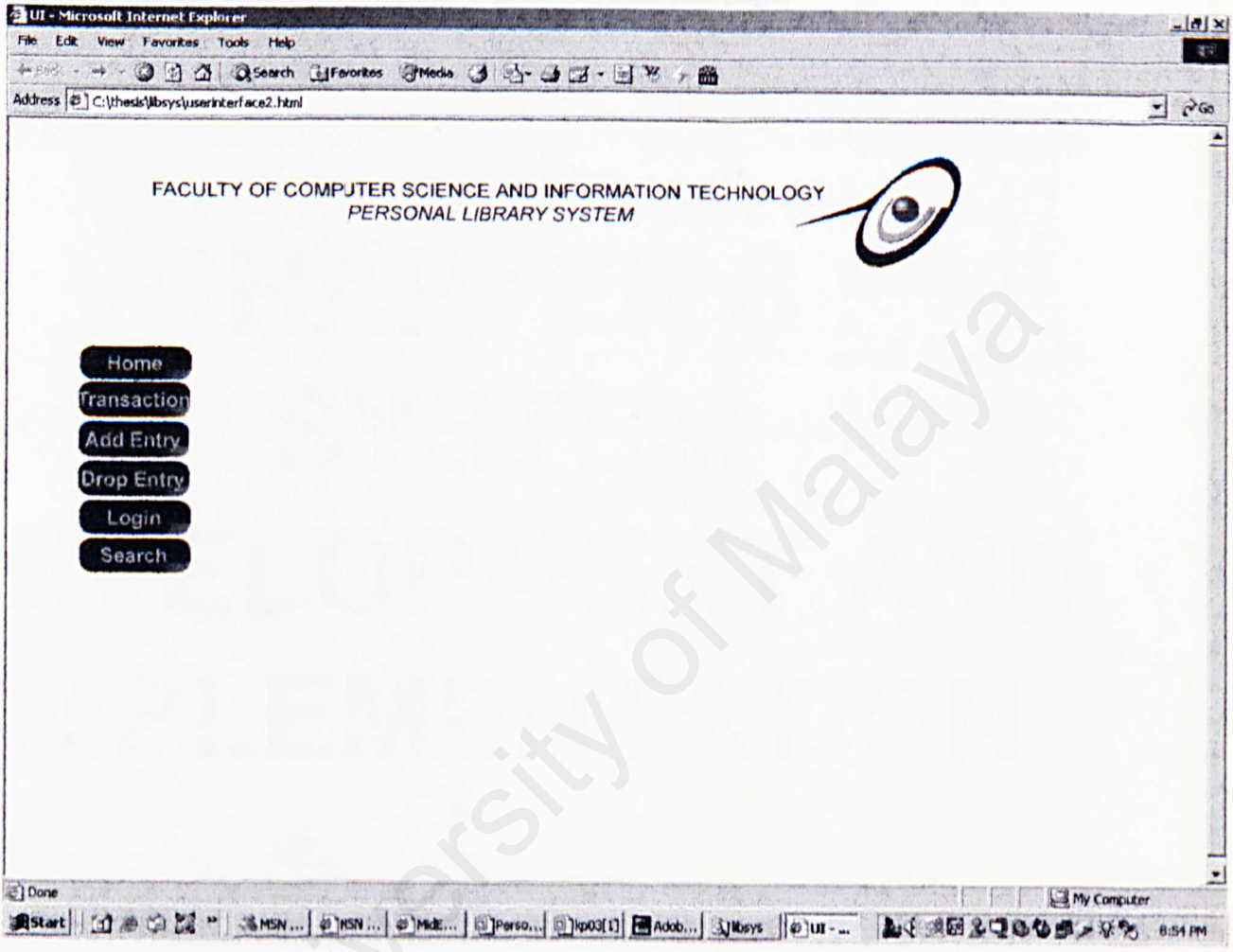


Figure 5.4: User Interface

Figure 5.4 show a screenshot of the user interface. The interface is very basic as more weight is put on the back-end running of the system. Basically the side menu will be the navigator for the users enabling them to browse the website. With the use on encryption, Administrators will be able to access the authentication page to edit their data. I have

applied minimal frills to the design to ensure quick loading and also used GIF images to reduce the file size. The user interface was developed using Adobe Photoshop 6.0 and sliced to smaller files then compiled together as a HTML page.

CHAPTER 6 : SYSTEM DEVELOPMENT AND IMPLEMENTATION

Chapter 6: System Development and Implementation

Introduction

The system development of this project was more like a top down approach where I started with the database design and development then the processing and user interface. The downside of this approach is that I often had to make changes to my database design, field properties and relationships to make it suitable with the processing. In a way this is the main reason I chose the Evolutionary Model. This methodology caters for such development methods and provides room for error and correcting of mistakes.

The first part of the development was implementing the database design. This was followed by building SQL queries using the VIEW option in SQL Server 2000. Once this was complete I moved on to developing the modules as mentioned in chapter 4. There are 5 modules all together :

1. Data Manipulation and Control
2. User Access Levels
3. Search Module
4. Resources Module
5. Transaction Module

6.1 Database Development

6.1.1 Database Design

The database development was based on the database design presented in chapter 5. The database consists of one main foreign key used for most queries that is 'personID' from 'tblMaster'. Each table has its own primary key and is connected to other tables using either a primary key or a foreign key.

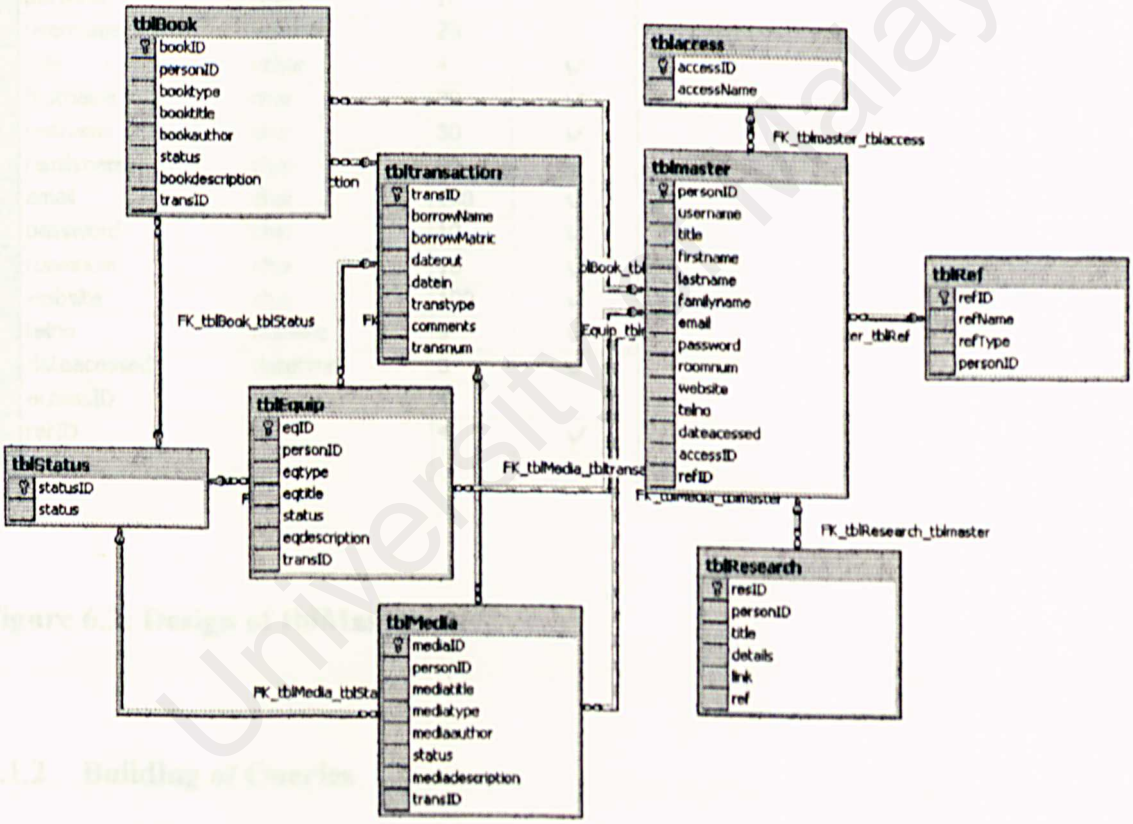


Figure 6.1: Database Diagram

Based on the diagram above, we can summarise that there are 2 central tables, tblMaster and tblTransaction. The tables which hold most information and would be most queried

are tblBook, tblEquip and tblMedia. Looking at this the 2 redundant tables, tblRef and tblResearch are more for future expansion. The status table (tblStatus) basically classifies the resource under certain classes. The transaction table holds information of people users have lent their resources to. The design of this database is basically to ensure that the system can act as dynamically as possible plus it has made a lot of allowance for future upgrades.

	Column Name	Data Type	Length	Allow Nulls
🔑	personID	char	10	
	username	varchar	20	
	title	nchar	4	✓
	firstname	char	30	✓
	lastname	char	30	✓
	familyname	char	30	✓
	email	char	100	✓
	password	char	10	✓
	roomnum	char	10	✓
	website	char	100	✓
	telno	numeric	9	✓
	dateacessed	datetime	8	✓
	accessID	int	4	
	refID	int	4	✓

Figure 6.2: Design of tblMaster

6.1.2 Building of Queries

The queries for the whole system were mostly built using the SQL query builders available in SQL Server 2000. Among the options used were SQL Query Builder and View. These queries are rather accurate hence reducing probability in errors due to sql queries. This system made the building of joints very convenient. This was quite an

essential as every single query using a 'Select' statement required a joint. For example when performing a search the query used is :

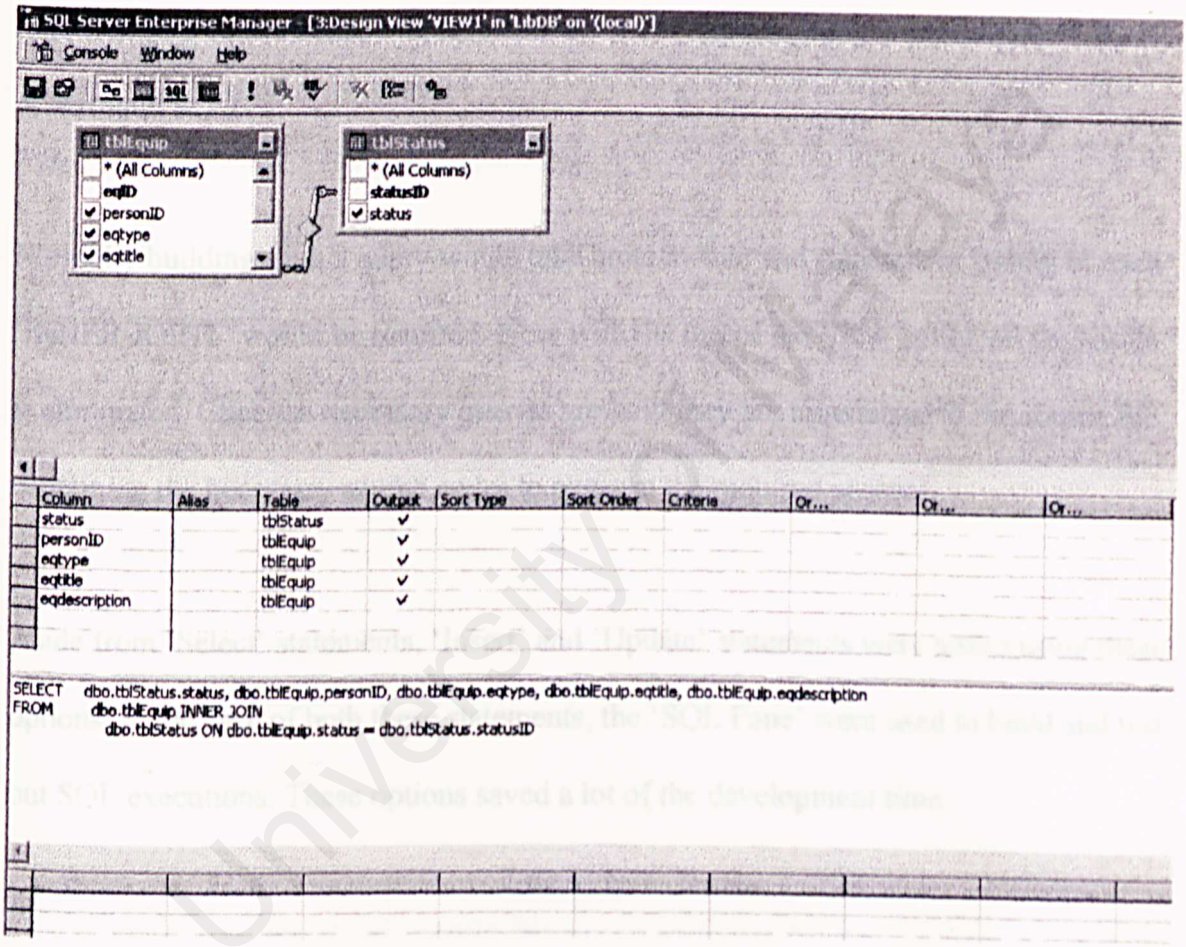


Figure 6.3: Example of sql query built using View option in SQL Server 2000

This was quite an essential as every single query using a 'Select' statement required a joint. For example when performing a search the query used is :

strsql = "SELECT dbo.tblEquip.eqtype, dbo.tblEquip.eqtitle, dbo.tblEquip.status, dbo.tblEquip.eqdescription, dbo.tblStatus.status AS Expr1, dbo.tblMaster.firstname, "

```
strsql &= "dbo.tblMaster.lastname, dbo.tblMaster.familyname "
strsql &= "FROM dbo.tblStatus INNER JOIN "
strsql &= "dbo.tblEquip ON dbo.tblStatus.statusID = dbo.tblEquip.status INNER JOIN "
strsql &= "dbo.tblMaster ON dbo.tblEquip.personID = dbo.tblMaster.personID "
strsql &= "WHERE (dbo.tblEquip.eqtype LIKE '" & textinput & "') OR
(dbo.tblEquip.eqtitle LIKE '" & textinput & "') OR  (dbo.tblEquip.eqdescription LIKE '"
& textinput & "') "
```

Normally building such a query would take quite awhile and substantive testing of each 'INNER JOINT' would be required. Now with the use of the View option all the hassle is eliminated. Once the necessary queries are built they are transferred to the source file containing the necessary source codes to perform the required process.

Aside from 'Select' statements, 'Insert' and 'Update' statements were tested using these options. In the case of both these statements, the 'SQL Pane' were used to build and test out SQL executions. These options saved a lot of the development time.

6.2 Module Development and Implementation

6.2.1 Building The Database Connection

ASP.Net provides to general methods of creating the database connection, OLEDB Connection and SQL Connection. The SQL connection string is specifically for SQL Server 7.0 and above. This connection, imbedded in the public class file makes it accessible by any page in a project file. Example of the connection string is as below:

"Data Source=ANX815\S2DB;Initial Catalog=LibDB;User ID=sa1;Password=s22003"

This string is used by an SQL Connection object making the actual string look something like this:

```
Dim objconn As New SqlConnection
Objconn.SqlConnection = "Data Source=ANX815\S2DB;Initial Catalog=LibDB;User
ID=roddey;Password=roddey"
```

To make this connection string a public object, I included this string in the Web.config after the <configuration> tag:

```
<appSettings>
 <add key="connectionstring" value="server=RODDEY; database=LibDB;      uid=sa1;
                                    pwd=s22003"/>
</appSettings>
```

Then by declaring the connection string object at the public class of each source file I was able to use it. This made it convenient when it came to changing the location of the application as only the connection string in Web.config had to be changed.

The public class declaration:

```
Dim objconn As New

SqlConnection(ConfigurationSettings.AppSettings("ConnectionString"))
```

6.2.1 Testing out the SQL query in and ASP.NET environment

This part of the development process was the most difficult in the whole development process. Each reference book provided different methods which often conflicted with one another. Finally I found to efficient ways of executing SQL statement. They were the SqlDataAdapter method and the SQL Command method. The SQL Command method is very similar to using a command object using the ActiveX object in classic ASP. The SQL Adapter method is similar to using the Connection object with the ActiveX object.

For most of the project I used the SqlDataAdapter method as I found this object much more stable compared to the command object. An example of the SqlDataAdapter method is below:

```
Dim strsql As String

strsql = "SELECT * FROM tblBook"

Dim sdalib As New SqlDataAdapter(strsql, objconn)

objconn.Open()
```

```
sdalib.SelectCommand.CommandType = CommandType.Text
dslib.Clear()
sdalib.Fill(dslib, "tblBook")
dvlib = dslib.Tables(0).DefaultView
objconn.Close()

grid1.DataSource = dvlib
grid1.DataBind()
```

As the connection object has been already declared earlier as objconn, it did not need to be declared again. Instead the SqlDataAdapter object just had to call it and refer it to the SQL string object – 'strsql'. Two other objects that had been declared as public objects were the dataview (dvlib) and dataset (dslib). The datagrid (grid1) had already been declared in the aspx file and merely called as a public object here. The convenience of developing in a Visual Studio.Net environment is that all objects once declared are easily called in any macros or functions. This made it very convenient for a person new to Dot Net programming to learn how to go about developing in it.

6.2.2 Building The Data Manipulation and Control Module

This was probably the most challenging module to build. The displaying of data was very easy with the use of the datagrid. However building and editable datagrid is another matter all together. There are 2 cases where editable datagrids had to be built. The first case was updating user information. This was quite simple as the query only involved a single table.

The first challenge was creating a template datagrid. This was a rather simple task but attempting to query more than one table proved to be very difficult. The process involved a number of sub processes and functions.

1. Preparing the datagrid for editing by first loading the information into the datagrid.
2. Creating the necessary private Sub for the edit, update and cancel buttons. These buttons would then call the relevant function to execute the processes necessary.
3. Creating the update command event. This involved retrieving the changes made by the user, updating the database then displaying the new query results.

Below is a screenshot of part of the editable datagrid:

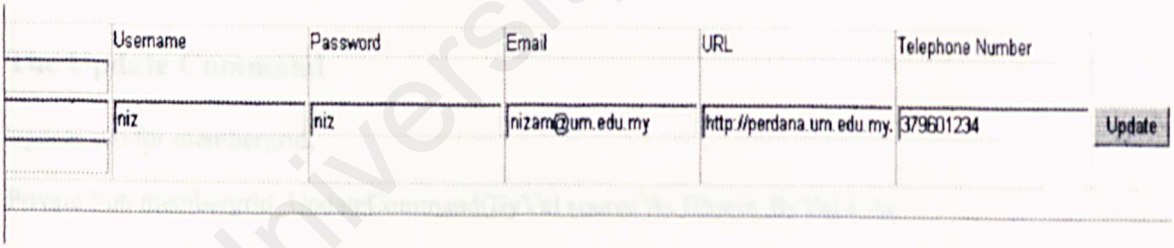


Figure 6.4: Edittable datagrid for updating the membership information.

The sourcecodes involved is as below:

The Edit Command

```
Private Sub membergrid_EditCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles membergrid.EditCommand
```

```
    membergrid.EditItemIndex = e.Item.ItemIndex
    Dim dsmem As DataSet = GetDataSet()
    BindGrid(dsmem)
End Sub
```

The Cancel Command

```
'cancel sub for membergrid
Private Sub membergrid_CancelCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles membergrid.CancelCommand

    membergrid.EditItemIndex = -1
    Dim dsmem As DataSet = GetDataSet()
    BindGrid(dsmem)

End Sub
```

The Update Command

```
'update sub for membergrid
Private Sub membergrid_UpdateCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles membergrid.UpdateCommand

    'variables to hold values
    Dim newFirstName, newLastName, newFamilyName As String
    Dim newUsername, newPassword, newEmail, newWebsite As String
    Dim newtelno As String

            newFirstName = CType(e.Item.FindControl("txtfirstname"),  TextBox).Text
        newLastName = CType(e.Item.FindControl("txtlastname"),          TextBox).Text
```

```
        newFamilyName = CType(e.Item.FindControl("txtfamilyname"), TextBox).Text

    newUsername = CType(e.Item.Cells(2).Controls(0), TextBox).Text
    newPassword = CType(e.Item.Cells(3).Controls(0), TextBox).Text
    newEmail = CType(e.Item.Cells(4).Controls(0), TextBox).Text
    newWebsite = CType(e.Item.Cells(5).Controls(0), TextBox).Text
    newtelno = CType(e.Item.Cells(6).Controls(0), TextBox).Text

    Dim personID As String
    personID = e.Item.Cells(0).Text
    Dim sdamember As New SqlClient.SqlDataAdapter
    Dim commember As New SqlClient.SqlCommand
    Dim update As String

        update = "UPDATE tblMaster SET username='" & newUsername & "',firstname='" &
        newFirstName & "', "
        update &= "lastname='" & newLastName & "',familyname='" & newFamilyName &
        "',email='" & newEmail & "', "
        update &= "password='" & newPassword & "', website='" & newWebsite & "',telno='" &
        newtelno & "'"
    update &= "WHERE personID='" & personID & "'"

    objconn.Open()
    commember.Connection = objconn
    commember.CommandText = update
    commember.CommandType = CommandType.Text
```

```
commember.ExecuteNonQuery()
objconn.Close()

End Sub
```

In this case the command object was used to execute the sql statement. In order for the datagrid to go back to it's original form after updating the changes, the grid needs to be rebound. This is done using the line below:

```
grid1.EditItemIndex = -1
Dim ds As DataSet = GetDataSet()
BindGrid(ds)
```

The command for cancelling the edit mode is:

```
grid1.EditItemIndex = -1
```

And the other 2 lines rebinds the grid using the BindGrid() sub and updates the select statement using GetDataSet() function. A function is used here as it needs to return a value.

Below is the BindGrid sub:

```
Public Sub BindGrid(ByVal ds As DataSet)
    membergrid.DataSource = ds.Tables("tblMaster")
    Me.DataBind()
End Sub
```

Below is the BindGrid sub:

```
Public Function GetDataSet() As DataSet
    Dim str As String
    Dim dsmem As New DataSet
    str = "Select * FROM tblMaster where personID='" & Session("personID") & "'"

    Dim sdamember As New SqlDataAdapter(str, objconn)
    objconn.Open()

    sdamember.SelectCommand.CommandType = CommandType.Text
    dsmem.Clear()
    sdamember.Fill(dsmem, "tblMaster")
```

```
    objconn.Close()
    Return dsmem

End Function
```

6.2.3 Building The User Access Levels

There a basically only 2 user access levels taken into consideration in this application, that is lecturers and the students and outsiders. For this a login session is created for lecturers to be able to manipulate their data. A session object is created and passed on from page to page till another user logs into the system or the page is closed. As the security levels hare are quite low, no cookies are implemented.

The login module basically performs a check using the 'Select' statement to check it the user exists and then verifies it by checking the password as well. Should either password or login name be non-existent, an error message will be generated at the login page.

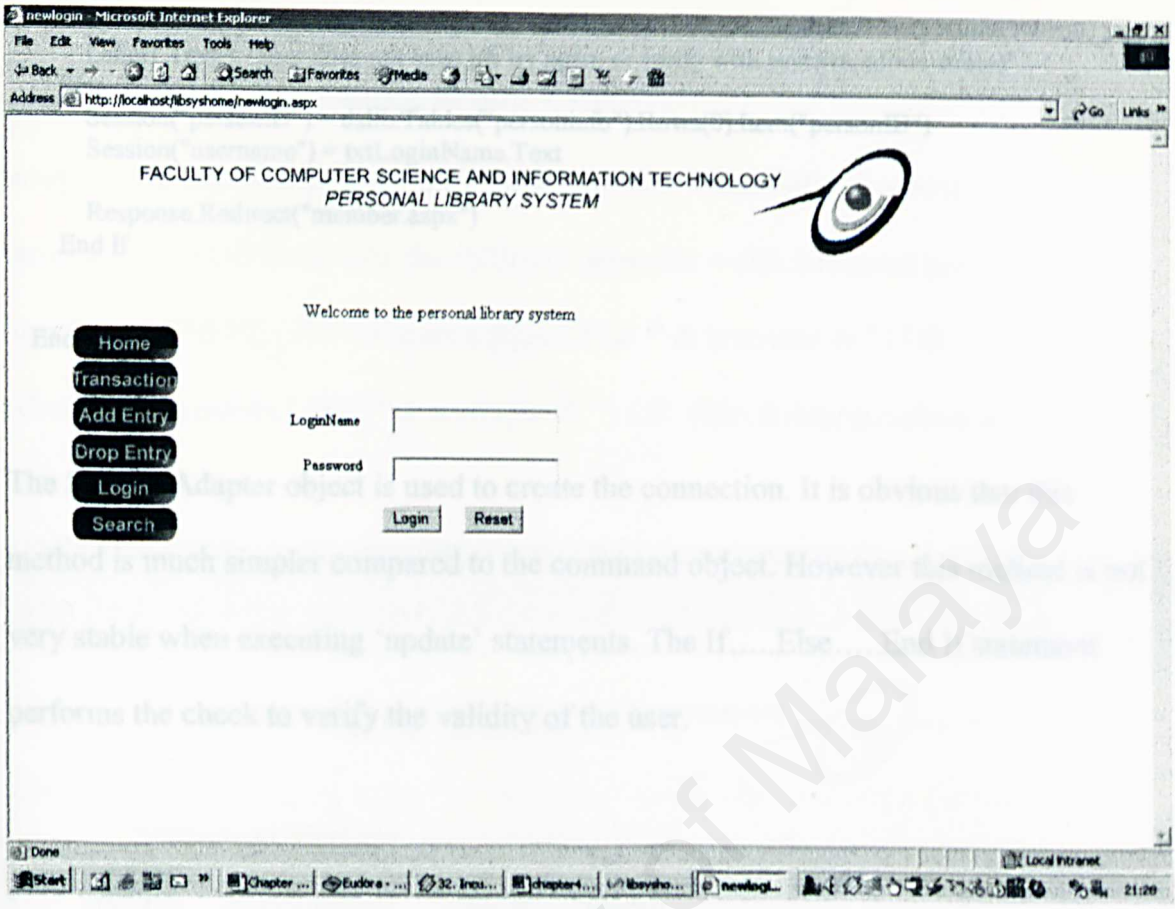


Figure 6.3: Login Page (newlogin.aspx)

Below is the code that performs the login upon the click of the login button.

```
Private Sub btnLogin_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
btnLogin.Click

    Dim password As String
    Dim user As String
    user = txtLoginName.Text.Trim
    password = txtPassword.Text.Trim

    Dim str As String
    str = "Select personID from tblMaster where username='" & user & "' AND password='" & password
& "'"

    Dim sdalib As New SqlDataAdapter(str, objconn)

    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()

    sdalib.Fill(dslib, "personinfo")
```

```
    If dslib.Tables("personinfo").Rows.Count = 0 Then
        Label1.Text = "user does not exist pls try again or verify with systems administrator"
    Else
        Session("personID") = dslib.Tables("personinfo").Rows(0).Item("personID")
        Session("username") = txtLoginName.Text

        Response.Redirect("member.aspx")
    End If


End Sub
```

The SqlDataAdapter object is used to create the connection. It is obvious that this method is much simpler compared to the command object. However this method is not very stable when executing 'update' statements. The If…..Else…..End If statement performs the check to verify the validity of the user.

6.2.4 Building The User Access Levels Search Module

This module is basically custom made for the students. It basically performs simple searches through the database to see if the database contains the required resource. To make the development of this module simpler, I divided the search to the specific types of resources. Using the 'LIKE' option in a 'Select' statement the search is reasonably effective. Example of the SQL statement:

```
strsql = "SELECT dbo.tblEquip.eqtype, dbo.tblEquip.eqtitle, dbo.tblEquip.status,
dbo.tblEquip.eqdescription, dbo.tblStatus.status AS Expr1, dbo.tblMaster.firstname, "
```

```
strsql &= "dbo.tblMaster.lastname, dbo.tblMaster.familyname "
strsql &= "FROM dbo.tblStatus INNER JOIN "
strsql &= "dbo.tblEquip ON dbo.tblStatus.statusID = dbo.tblEquip.status INNER JOIN "
strsql &= "dbo.tblMaster ON dbo.tblEquip.personID = dbo.tblMaster.personID "
strsql &= "WHERE (dbo.tblEquip.eqtype LIKE '" & textinput & "') OR
(dbo.tblEquip.eqtitle LIKE '" & textinput & "') OR  (dbo.tblEquip.eqdescription LIKE '"
& textinput & "') "
```

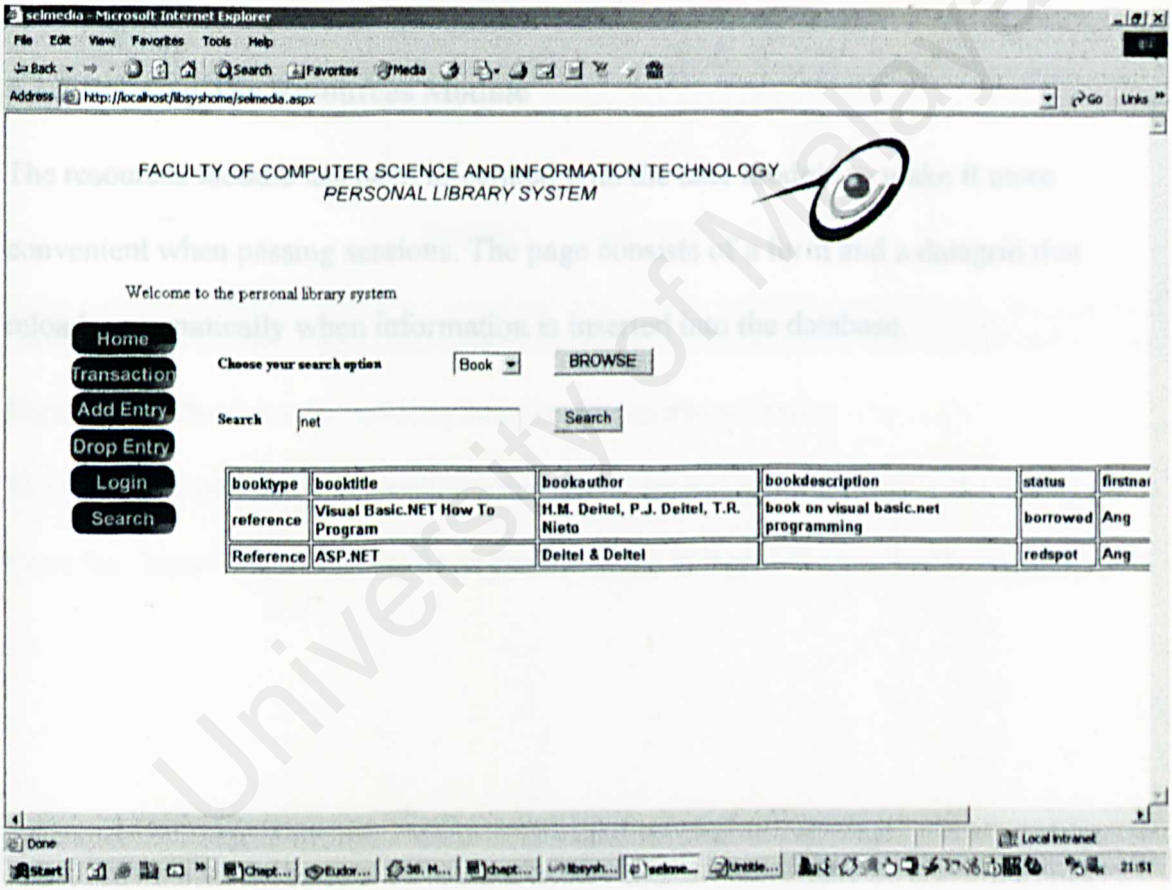


Figure 6.4: Search Page (selmedia.aspx)

The Sql statement basically queries the respective fields to see any similarity between the users search and the database. The query is executed using a simple 'onclick' sub.

Below is the onclick sub:

```
Private Sub btnsearch_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
btnsearch.Click

    If ddloption.SelectedItem.Value = 1 Then
      bindBooksearch()
    ElseIf ddloption.SelectedItem.Value = 2 Then
      bindmediasearch()
    Else
      bindEquipsearch()
    End If

  End Sub
```

6.2.5 Building The Resources Module

The resources module has been integrated with the user module to make it more convenient when passing sessions. The page consists of a form and a datagrid that reloads automatically when information is inserted into the database.

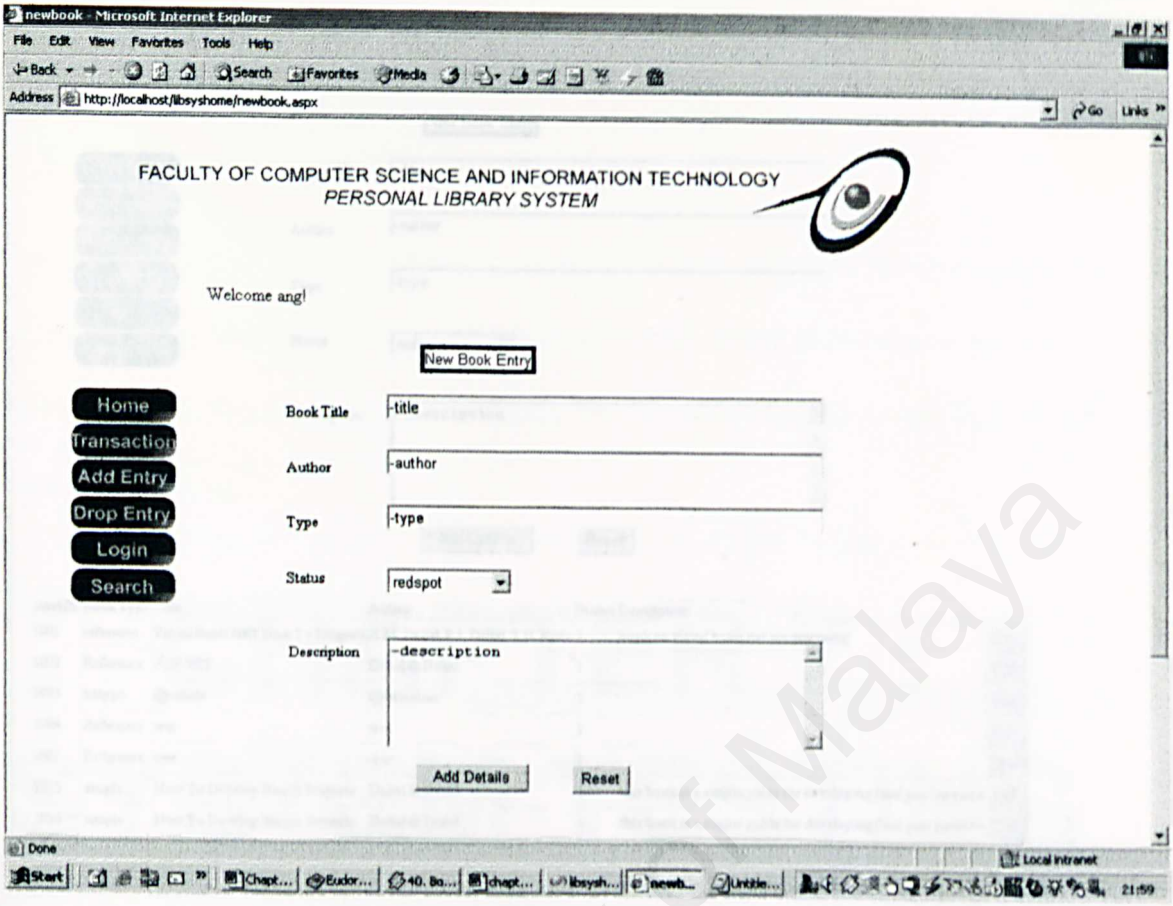


Figure 6.5: The form for adding new books (newbook.aspx)

The form executes with the onclick sub that inserts the information into the database.

Once the 'Insert' statement has been executed, the datagrid automatically updates itself.

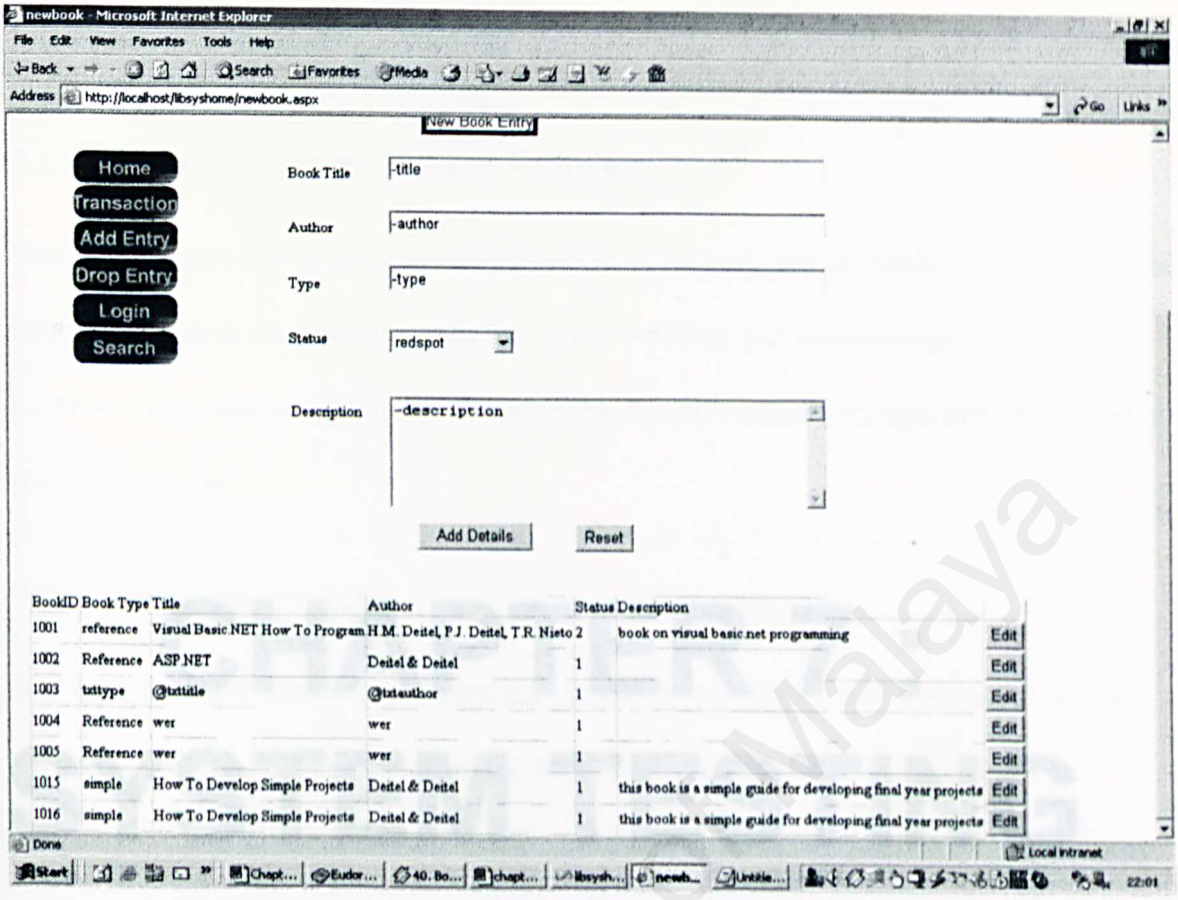


Figure 6.6: The add books page with the datagrid at the bottom (newbook.aspx)

The same method is used for all resources. The reset button us useful when executing multiple entries.

CHAPTER 7 : SYSTEM TESTING

Chapter 7: System Testing

7.1 Introduction to Testing Software

Testing Software is a most essential part of developing a system. Many companies spend more time testing their system as to the actual building and implementation. This part involves many aspects including checking the overall system for bugs, defects and other possible mishaps.

7.2 What is testing

Software testing is the process of testing programs to establish the presence of system defects. There are certain objectives necessary in achieving this. They are:

- To understand testing techniques that are geared to discover program faults
- To understand specific approaches to object-oriented testing
- To understand the principles of CASE tool support for testing

7.3 Types of testing

There are several types of testing. Each uniquely focuses on different parts of the whole testing process.

7.3.1 Component Testing

The first part of testing is component testing. This is where each module and component is tested for bugs and such. During this process dumb data might be inserted into certain

components to see how the component functions on its own. The steps in component testing are:

- Testing of individual program components
- Testing the output of each component

7.3.2 Integration Testing

Once the components have been tested another, integration testing has to be performed. This involves testing of groups of components integrated to create a system or sub-system. Tests are based on a system specification.

The testing phases are as per the diagram below:

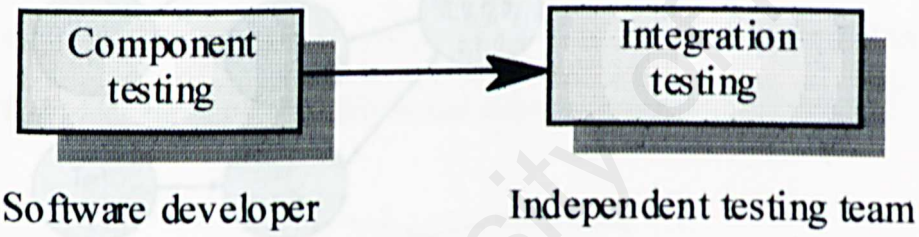


Figure 7.1: Testing Phases

Integration testing should be black-box testing with tests derived from the specification. The main difficulty is localising errors. Incremental integration testing reduces this problem. There are a number of approaches to integration testing. They are:

Top-down testing

Starts with high-level systems and integrate from the top-down replacing individual components by stubs where appropriate.

Figure 7.2: Top-Down Integration

Bottom-up testing

Process of integrating individual components in levels until the complete system is created.

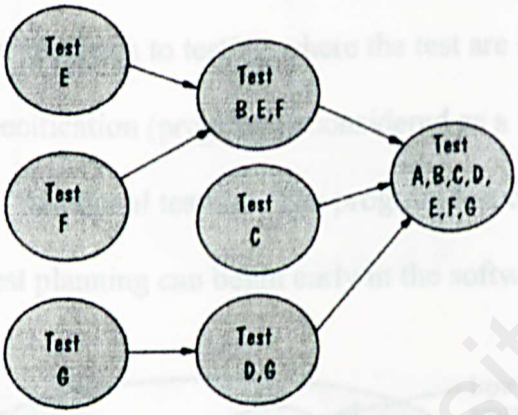


Figure 7.3: Bottom-Up Integration

7.3.3 Defect Testing

Once this is complete the system has to be tested for defects. The goal of defect testing is to discover defects in programs. A successful defect test is a test which causes a program to behave in an anomalous way. Tests show the presence not the absence of defects. A better picture of the defect testing idea is pictured in the figure below:

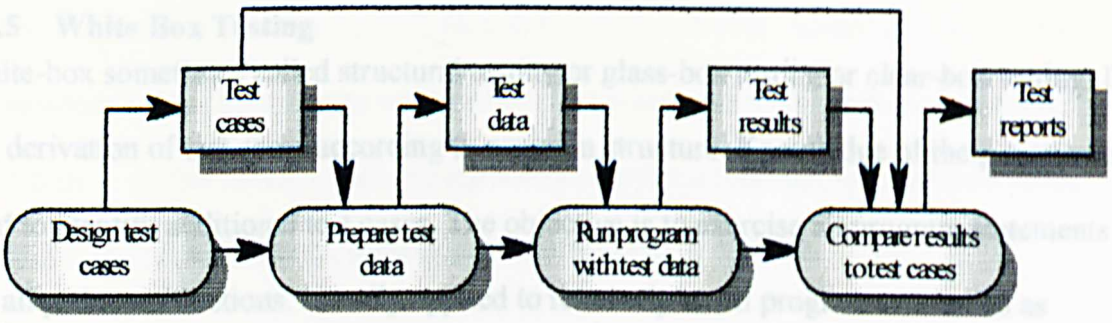


Figure 7.4: The Defect Testing Process

7.3.4 Black Box Testing

An approach to testing where the test are derived from the program or component specification (program is considered as a 'black-box'). Black Box Testing is also known as 'functional testing'. The program test cases are based on the system specifications. Test planning can begin early in the software process.

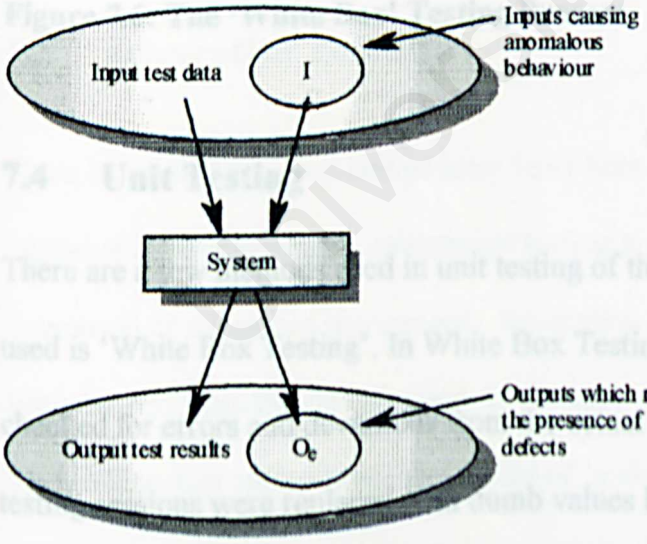


Figure 7.5: The 'Black Box' Testing Process

7.3.5 White Box Testing

White-box sometimes called structural testing or glass-box testing or clear-box testing. It is a derivation of test cases according to program structure. Knowledge of the program is used to identify additional test cases. The objective is to exercise all program statements not all path combinations. Usually applied to relatively small program units such as subroutines or operations associated with an object.

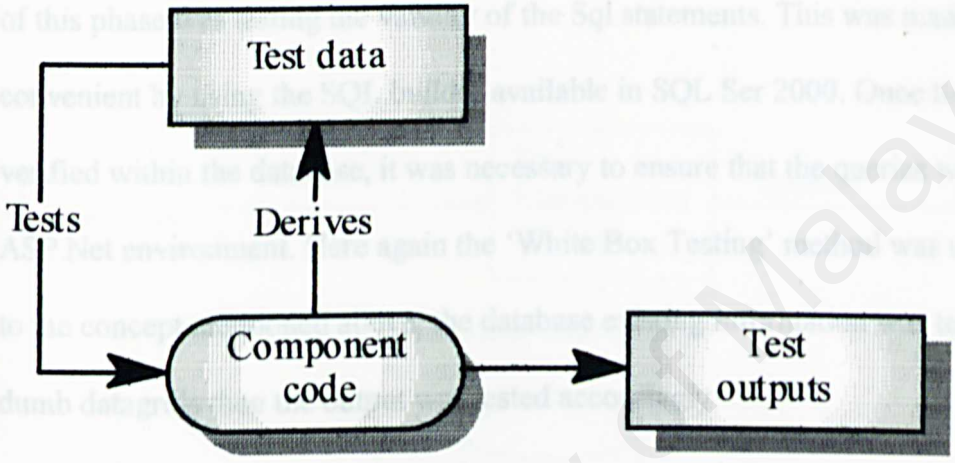


Figure 7.6: The 'White Box' Testing Process

7.4 Unit Testing

There are a few methods used in unit testing of the system. However the main method used is 'White Box Testing'. In White Box Testing, the internal logic of each module is checked for errors and deviations from the actual systems specifications. To enable such testing sessions were replaced with dumb values before integrating it with the remainder of the system. This enables the testing of output. With reference to Figure 7.6 (The 'White Box' Testing Process), the initial test is to test data, and then to test output. For testing the data, dumb data grids are created, then displayed to ensure correct input is

received. If the input data is limited, then this is replaced with 'labels' that are loaded by the necessary subs. This was an essential method when testing the authentications in the login pages. In this manner both the input and output are thoroughly tested before the module is integrated with the remainder of the system.

7.4.1 Database Queries Testing

The most important component testing was the testing of the SQL queries. The first part of this phase was testing the validity of the Sql statements. This was made very convenient by using the SQL builder available in SQL Ser 2000. Once the queries were verified within the database, it was necessary to ensure that the queries worked in a ASP.Net environment. Here again the 'White Box Testing' method was used. Similarly to the concept mentioned above, the database existing information was tested using dumb datagrids then the output was tested accordingly.

7.5 Integration Testing

Once individual program components have been tested, they must be integrated to create partial or complete system. Integration testing tests complete systems or subsystems composed of integrated components. Integration testing should be black-box testing with tests derived from the specification. In the case of integration testing, I did use the 'black box' testing approach where the concept of 'auditing around the system' was used. The main modules that required integration testing where the:

1. Data Manipulation and Control
2. User Access Levels

3. Resources Module

These modules were interlinked with each other at either one point or another. Basically the testing process starts and the user 'login' page. Then the user is sent to the data page where data manipulation may be performed. This page is linked to the resources module where additions can be made to the

database.

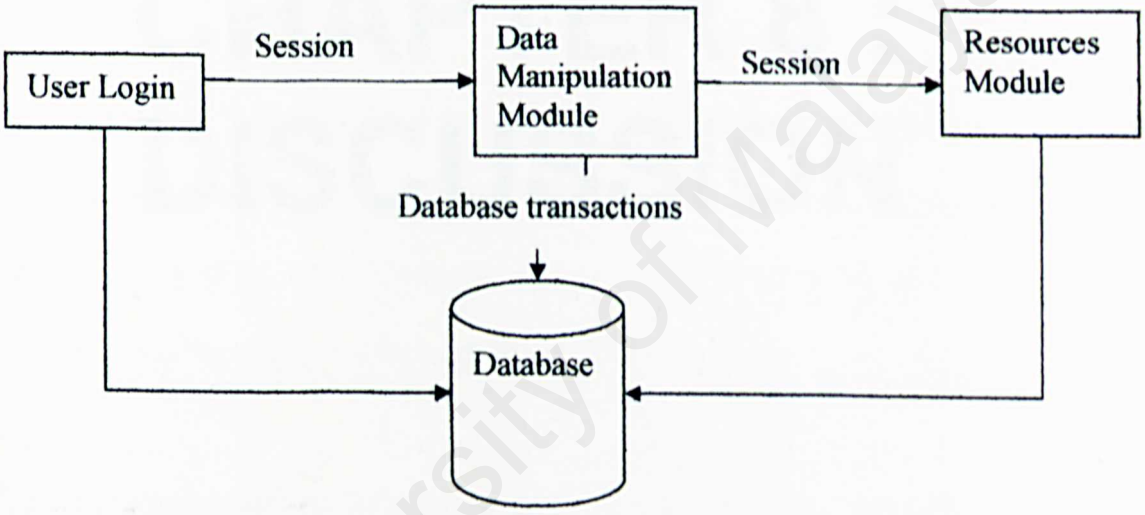


Figure 7.7: The Integration Testing Process

7.6 System Testing

The system testing involved the full integration of the system. The main concept used in this phase of testing is Defect Testing. Here the system undergoes constant scrutiny where any possible defects as a whole is singled out and corrected. This involved linking, database transactions and sessions. As the system is quite simple the main

system testing is basically similar to the integration testing. A lot of redesigning was done in the process of system testing to ensure free flow of information throughout the system.

CHAPTER 8 : DISCUSSION

Chapter 8: Discussion

Introduction

The Personal Library System is basically a very simple system. It mainly consists of a search module, a data manipulation module and a transaction module. Developing in a ASP.Net environment posed as quite a challenge for me as I had not done development in this area before. I faced many problems which resulted in a very simple system that did not fulfil all the systems specification set in the initial proposal.

There were many initial problems faced until I finally managed to understand the concept behind the .NET framework. A lot of time was wasted chasing dead ends as I found different solutions for similar problems. As ASP.Net is a very new language many others developing in this environment faced similar problems.

The final system might not be a very good system, nevertheless it is simple, and has a lot of room for upgrades.

8.1 Problems and Solutions

Many problems were faced during the duration of developing this system. Problems faced are probably similar to the ones faced by other students developing in a new environment for the first time. The process of familiarising with a new environment proved challenging enough. This was coupled with the common flaw among Microsoft products – the necessity to update and upgrade constantly to ensure problem free development.

8.1.1 Problems faced with the development tools

A lot of service packs were required in order to deploy .Net systems in a Windows 2000 environment. Often the necessary service packs downloadable from the internet. However the file sizes were quite big ranging from 50 to 170 megabytes. The initial .Net SDK 1.0 had to be upgraded to .NET SDK 1.1 and this again required certain service packs and server patches.

A lot of time was wasted trying to understand the source of an error, later on discovering it was due to a missing upgrade.

The second problem faced was in the actual development. Visual Studio.Net is a very user-friendly development tool. The only fault of it is that many of the drag and drop options are not stable hence resulting in errors when debugging. Among the problematic areas are the establishment of a SQLDataConnection, SQLDataAdapter and a

SQLDataCommand. These objects when declared using the available wizard options are unstable and static. For dynamic deployment these objects must be declared in the source file (.aspx.vb file) manually.

There were also a lot of inadequacies in terms of objects for convenient programming control. For example: DataSourceControl. This object is to assist in convenient database transactions. However only the latest versions of Visual Studio.NET supported this object. The other versions did not support it as the necessary dll and class file was not contained within the class library.

8.1.2 Problems and Advantages with ASP.Net

Despite the fact that ASP.Net is merely an upgrade from classic ASP, there are a lot of differences between the two. The first thing is that the fact ASP.Net is a proper compilation language that shares the concept of Java as in compile once, run anywhere. As it is a new language, there were limited resources. Unlike ASP were source codes for almost anything is available on the internet, this is no the case.

Techniques and concepts available in tutorials are rather limited. Finding tutorials and books that had good techniques on querying databases were very difficult. The main problem faced here was the need to upgrade all source codes from OLEDB connections to SQL connections. Finding syntax for a good working SQL connections was another problem altogether.

Many websites such as 123ASPX.com and Planet Source Code contained tutorials for certain parts of the language but not proper complete processes. A lot of time was spent conducting 'trial and error' experiments before finally understanding the general concept behind ASP.Net.

However once these general concepts were grasped, the programming process was made quite easy. ASP.Net contains a lot of class objects that reduced development time and paved a way for short and accurate programming. Once the project file is built, the system runs very well, running more and more efficiently at every runtime.

Deploying the concepts of templates are also very easy. Unlike classic ASP where XML or CSS is required, here templates can be designed within the '.aspx' page. For example below is the code designing and editable datagrid:

```
<asp:DataGrid id="grid1" style="Z-INDEX: 124; LEFT: 18px; POSITION: absolute;
TOP: 621px" runat="server" AutoGenerateColumns="False">
<Columns>
        <asp:BoundColumn DataField="bookID" HeaderText="BookID"
ReadOnly="True" />
        <asp:BoundColumn DataField="booktype" HeaderText="Book Type" />
        <asp:BoundColumn DataField="booktitle" HeaderText="Title" />
        <asp:BoundColumn DataField="bookauthor" HeaderText="Author" />
        <asp:BoundColumn DataField="status" HeaderText="Status" />
        <asp:BoundColumn DataField="bookdescription" HeaderText="Description" />
```

<asp:EditCommandColumn EditText="Edit" CancelText="Cancel"

UpdateText="Update" ButtonType="PushButton" />

</Columns>

</asp:DataGrid>

This template can be generated in a ASP.Net page without the necessity of including any CSS or XML files.

8.2 Strengths and Weaknesses of the Systems

The personal library system has many weaknesses and a few strengths. The main plus point is the database design which was done in a very dynamic manner. The design allows a lot of room for further expansion.

Among the weaknesses of the system are:

1) Poor security - the authentication is performed by a session and does not contain any cookies or server runtimes. This would compromise the security of users

2) Unattractive User Interface – the user interface was designed with complete simplicity. An inexperienced ser might find this annoying as there are no guides or messages in guiding the user on how to use the system.

3) Incomplete transaction module – the transaction module was not successfully built due to time and knowledge constraints. As a result users cannot record all transactions made.

4) Absence of 'Delete' option – Users are unable to delete entries that have been made.

8.3 Improvements and Suggestions

The main improvement I would like to suggest for my system is a complete upgrade of the user access level, inclusive of regular and power users. This would ensure proper security. There should also be a proper system of recording error logs in the occurrence of outside abuse. This would enable proper systems audit and upgrades.

Another suggestion would be a delete option, allowing users to delete data owned by them. The necessary SQL statement would look something like:

Strsql = "DELETE FROM tblBook where bookID='" & bookID & " ' "

Where the bookID string would be the selected object for delete.

8.4 Conclusion

The personal library system developed leaves a lot to be desired. The system here is merely an initial prototype. As the system has been built using the .NET framework, a lot of area for expansion has already been given. The upgrades to the system would be more in the form of additions using ASP.Net. An upgrade to a new language or database management system would not be required for at least a few years.

Although .NET might be a difficult area to develop in, deploying online applications in this area is certainly the way to go. No doubt I would have been able to build a much more advanced system using classic ASP but this system would need to be upgraded to ASP.Net sooner or later. The options and areas of ASP.Net with the use of VB.Net and C# are quite immense. I certainly shall spend a lot more time exploring this new tool.

Despite the fact my application was developed using Visual Basic.NET (VB.Net),I think a more efficient manner of developing with ASP.Net in the future would be C#. C# has already become the preferred language for development and there are more resources for developing with C# as compared to VB.Net. My recommendations to future new developers is to attempt to explore C# before deciding on the language. Although C++ might be complicated and C# is based on C++, it is a much more versatile language and is able to develop many more sophisticated applications as compare to VB.Net.

BIBLIOGRAPHY

Bibliography

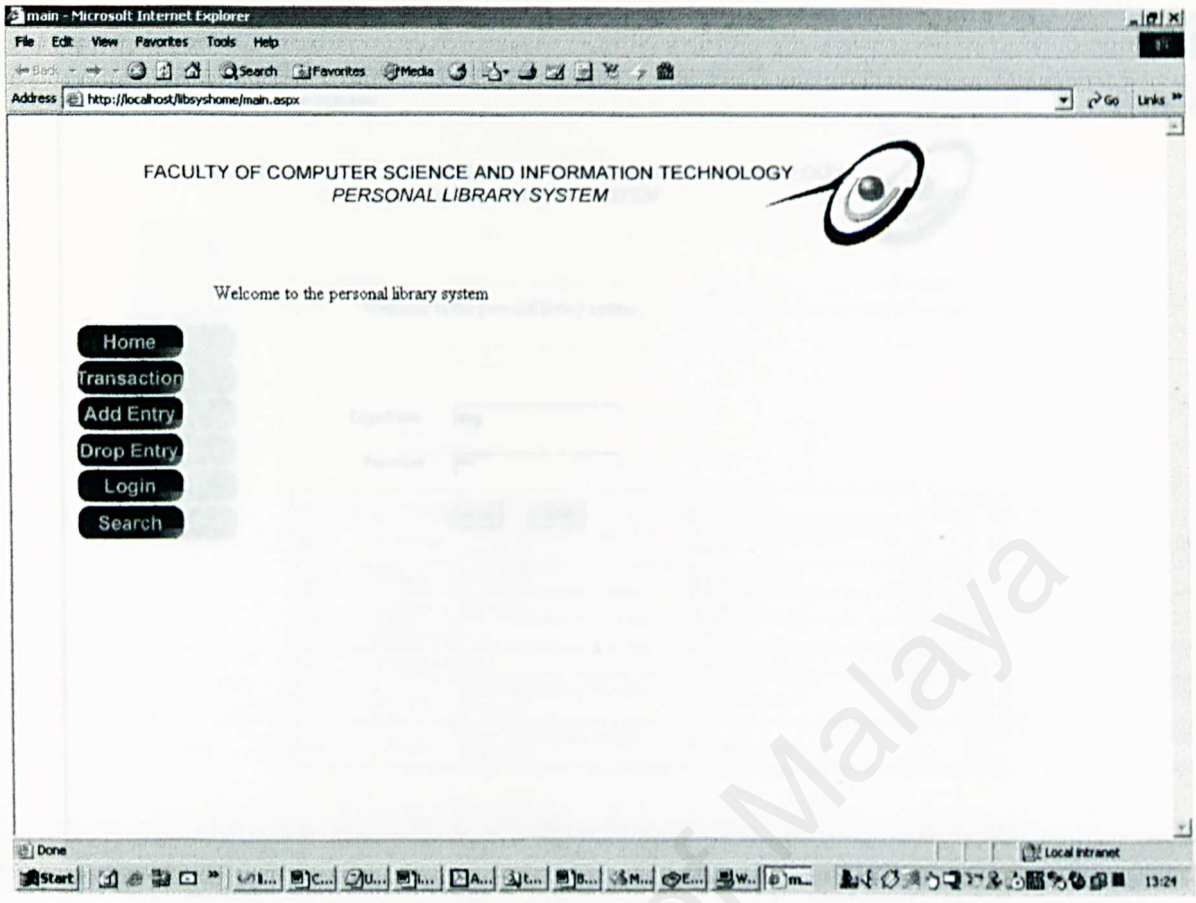
User Manual

Before uploading this application to a web host, necessary changes have to be made to the database connection. This database connection string in contained within the 'web.config' file. The string below is shown below is the database connection string used:

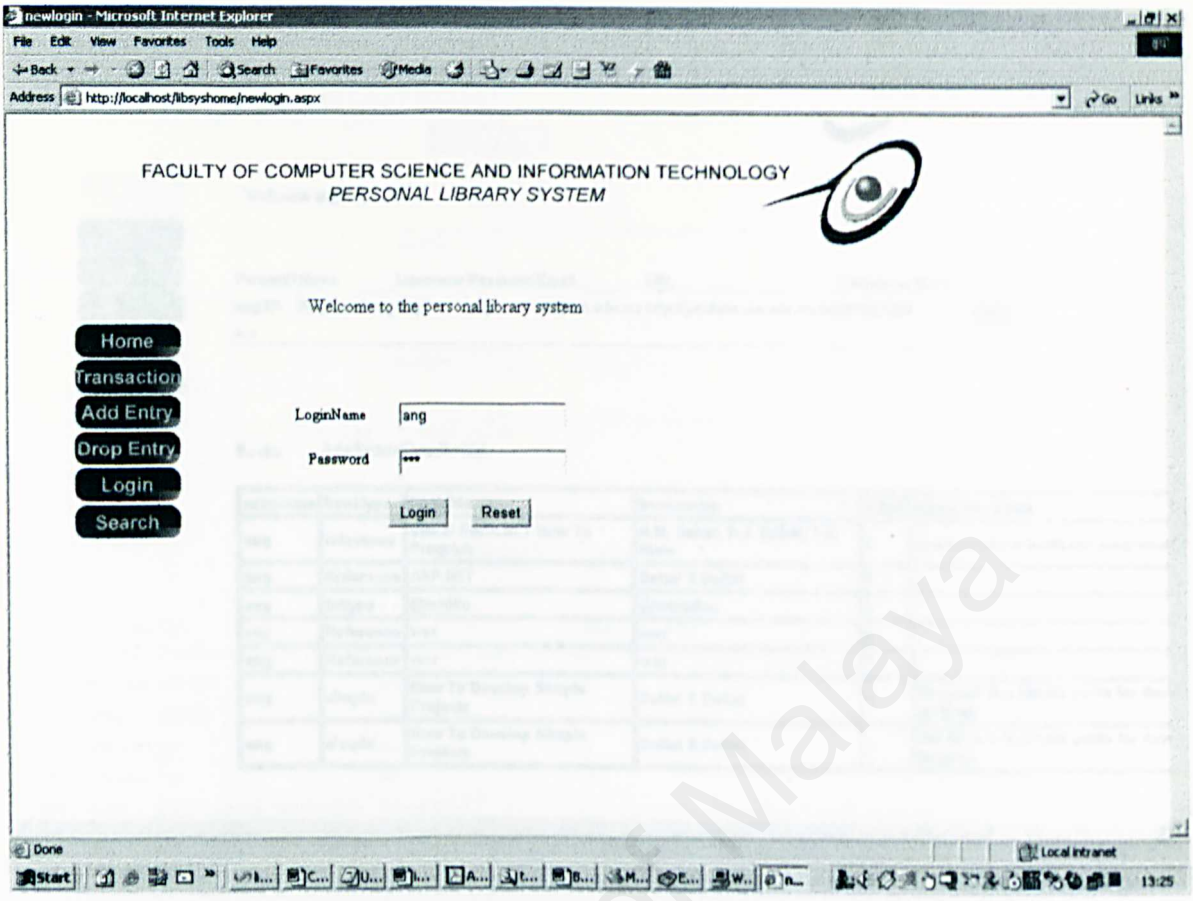
<add key="connectionstring" value="Data Source=ANX815\S2DB;Initial Catalog=LibDB;User ID=sa1;Password=s22003"/>

The necessary changes would be the Data Source which would have to be according to the name of the SQL Server being used. The second change would be the 'User ID' and 'Password'. These two options would be based on the information provided by the web host.

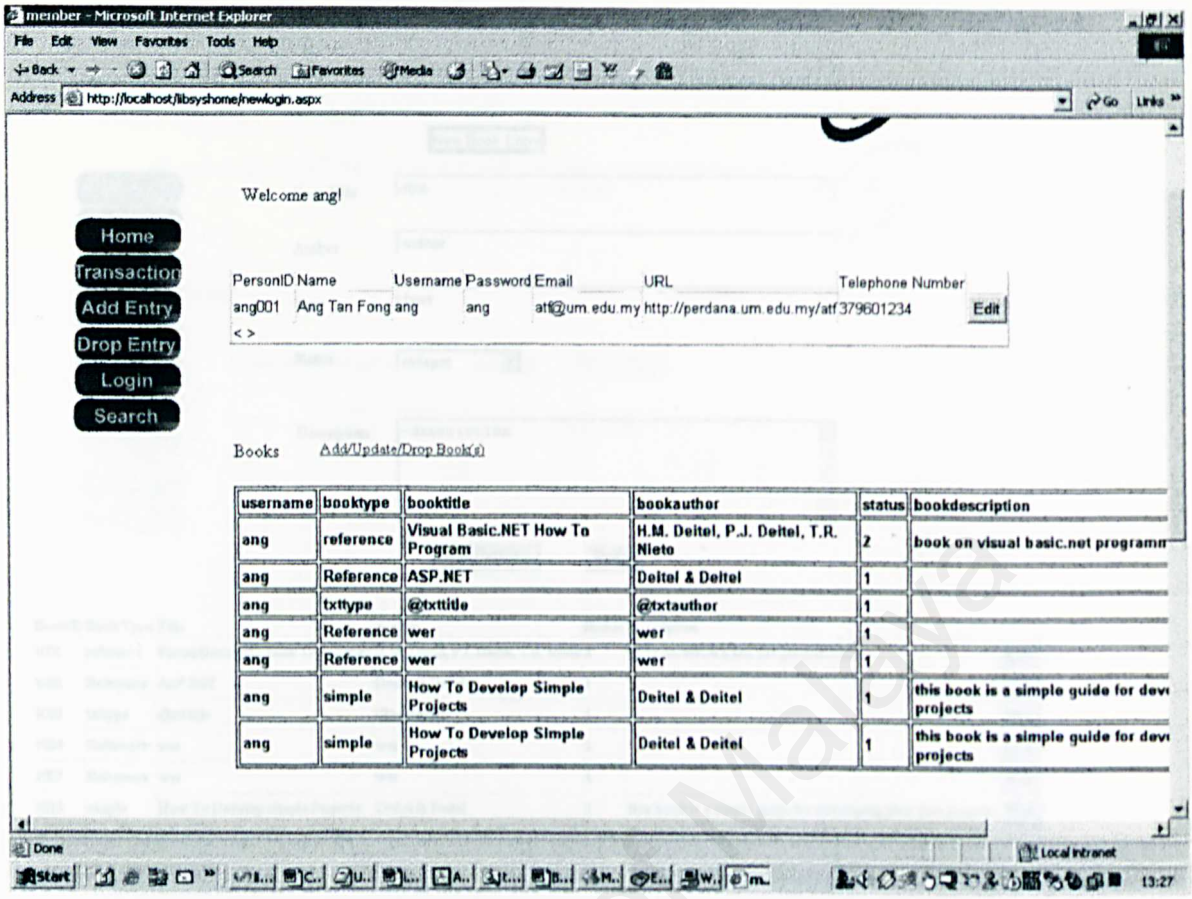
The respective usernames and passwords for lecturers would be provided by the systems administrator. Once this is done the user may change the password by following the steps below:



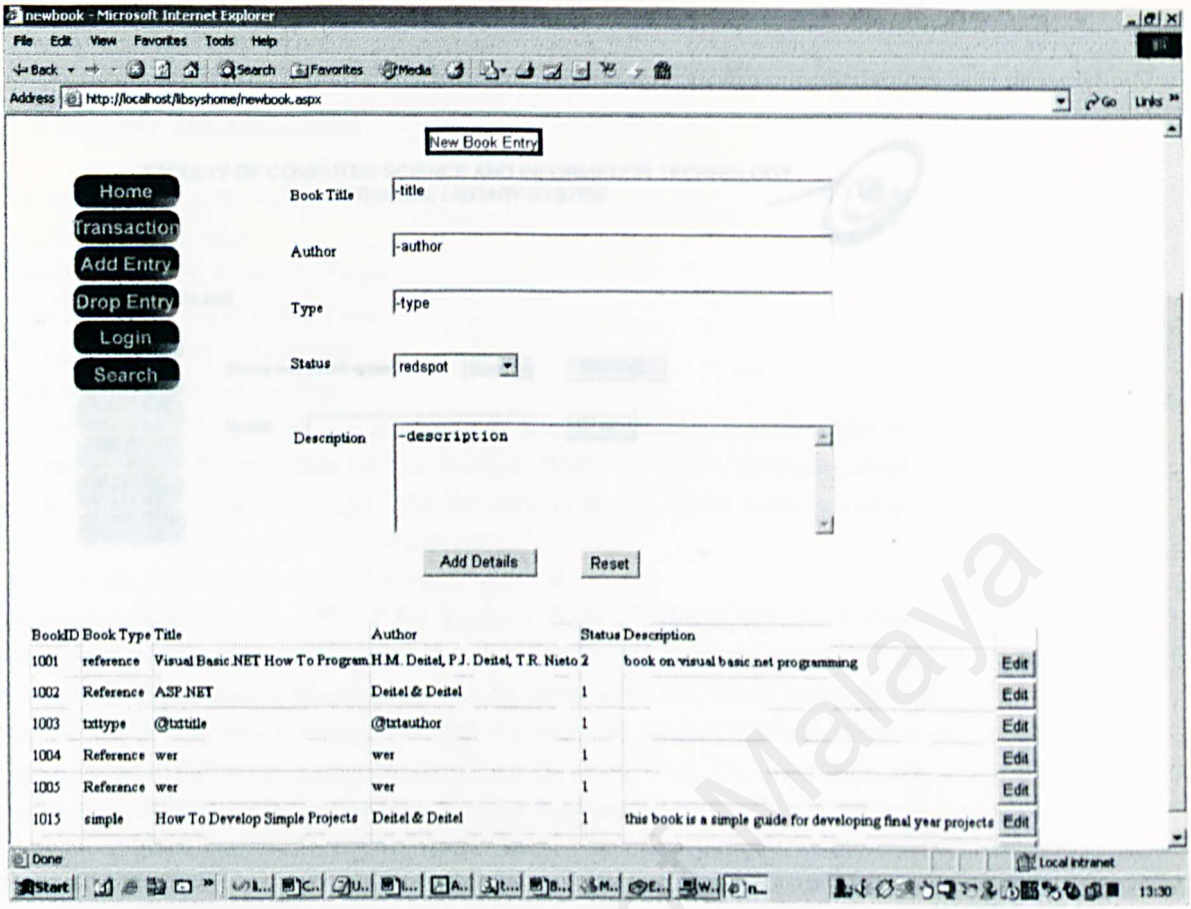
1. At the main page, click on the 'Login' button.



2. Key in the username and password as given by the systems administrator. The click on the login button. Should there be an error, a error message will appear.



3. At the first datagrid, there is all the user information that can be edited. Click the edit button, edit necessary details then click the update button. The information would be automatically updated.
4. For adding resources, click on the link above the relevant datagrid. You will be directed to a page containing a form and a datagrid below.

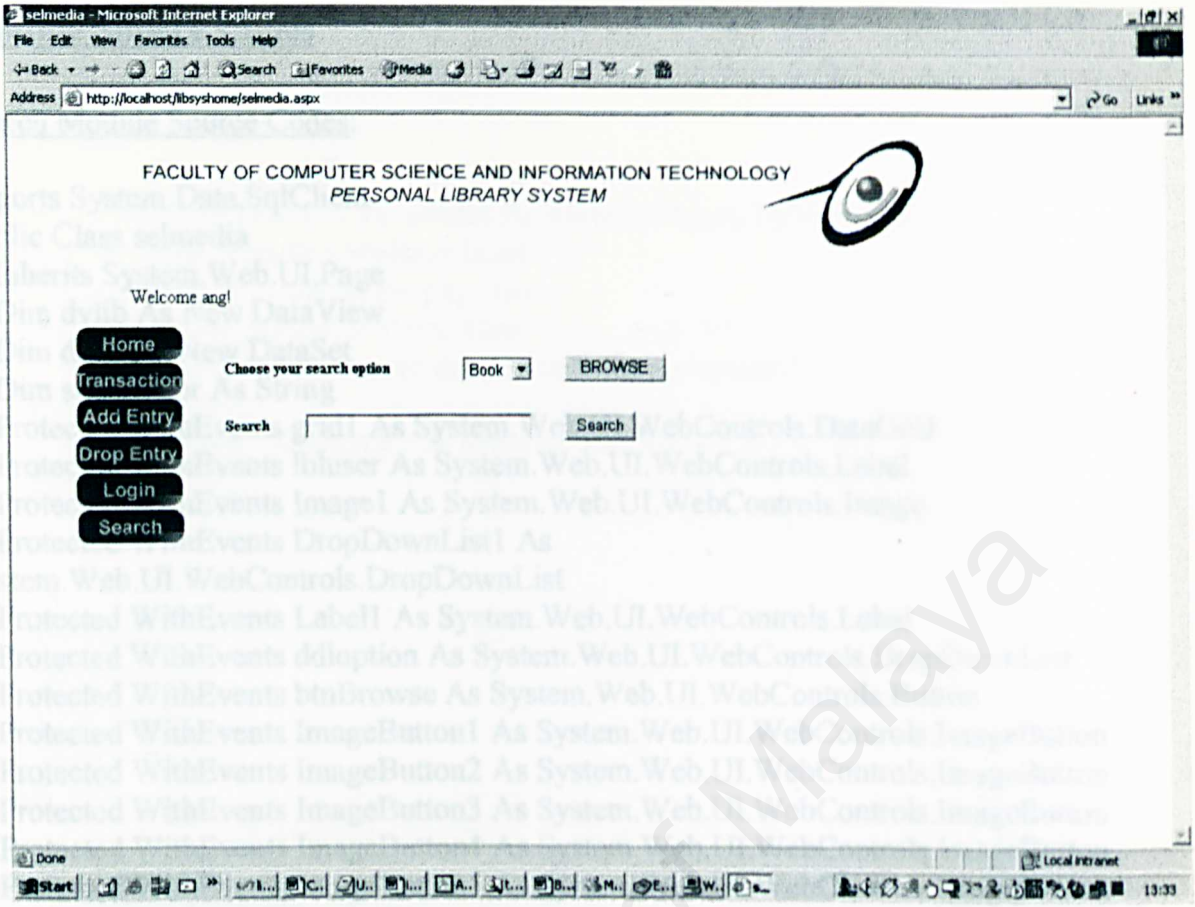


5. Fill in the form, click Add Details and the information will be updated accordingly.

6. For editing the resource information, click the edit button and follow step 3.

Searching the database

1. For searching or browsing the database click on the 'Search' button. The page as shown below will be seen.



2. Choose the search category, key in the keyword into the text box then click the 'search' button on the right of the textbox. The results will be displayed on a datagrid.
3. Should you wish to browse the database, just choose the search option, the click on the 'BROWSE' button. This will display all the data contained within the database for the particular option.

Module Source Codes

Search Module Source Codes:

```
Imports System.Data.SqlClient
Public Class selmedia
    Inherits System.Web.UI.Page
    Dim dvlib As New DataView
    Dim dslib As New DataSet
    Dim strsortexpr As String
    Protected WithEvents grid1 As System.Web.UI.WebControls.DataGrid
    Protected WithEvents lbluser As System.Web.UI.WebControls.Label
    Protected WithEvents Image1 As System.Web.UI.WebControls.Image
    Protected WithEvents DropDownList1 As
System.Web.UI.WebControls.DropDownList
    Protected WithEvents Label1 As System.Web.UI.WebControls.Label
    Protected WithEvents ddloption As System.Web.UI.WebControls.DropDownList
    Protected WithEvents btnBrowse As System.Web.UI.WebControls.Button
    Protected WithEvents ImageButton1 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton2 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton3 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton4 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton5 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton6 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton7 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents Label2 As System.Web.UI.WebControls.Label
    Protected WithEvents btnsearch As System.Web.UI.WebControls.Button
    Protected WithEvents txtsearch As System.Web.UI.WebControls.TextBox
    Dim objconn As New
SqlConnection(ConfigurationSettings.AppSettings("ConnectionString"))

#Region " Web Form Designer Generated Code "

    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()

    End Sub

    'NOTE: The following placeholder declaration is required by the Web Form Designer.
    'Do not delete or move it.
    Private designerPlaceholderDeclaration As System.Object

    Private Sub Page_Init(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Init
        'CODEGEN: This method call is required by the Web Form Designer
        'Do not modify it using the code editor.
```

```
    InitializeComponent()
  End Sub

#End Region

  Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    'Put user code to initialize the page here
    If Len(Session("personID")) = 0 Then
      lbluser.Text = "Welcome to the personal library system"
    Else
      lbluser.Text = "Welcome " & Session("username") & "!"
    End If



    'bindgrid()
    'bindmedia()
    'bindEquip()
    'bind()
  End Sub

  Private Sub bindgrid()
    Dim strsql As String
    'strsql = "SELECT * FROM tblBook"
    strsql = "SELECT dbo.tblBook.booktype, dbo.tblBook.booktitle,
dbo.tblBook.bookauthor, dbo.tblBook.bookdescription, "
    strsql &= "dbo.tblMaster.firstname, dbo.tblMaster.lastname,
dbo.tblMaster.familyname, dbo.tblStatus.status "
    strsql &= "FROM dbo.tblBook INNER JOIN dbo.tblMaster ON
dbo.tblBook.personID = dbo.tblMaster.personID INNER JOIN "
    strsql &= "dbo.tblStatus ON dbo.tblBook.status = dbo.tblStatus.statusID"



    Dim sdalib As New SqlDataAdapter(strsql, objconn)

    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "tblBook")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    grid1.DataSource = dvlib
```

```
    grid1.DataBind()
End Sub

Sub bindmedia()
    Dim strsql As String
    'strsql = "SELECT * FROM tblMedia"

    strsql = "SELECT dbo.tblMedia.mediatitle, dbo.tblMedia.mediatype,
dbo.tblMedia.mediaauthor, dbo.tblMedia.mediadescription, dbo.tblStatus.status,
dbo.tblMaster.firstname, dbo.tblMaster.lastname, dbo.tblMaster.familyname "
    strsql &= "FROM dbo.tblStatus INNER JOIN dbo.tblMedia ON
dbo.tblStatus.statusID = dbo.tblMedia.status INNER JOIN "
    strsql &= "dbo.tblMaster ON dbo.tblMedia.personID = dbo.tblMaster.personID "

    Dim sdalib As New SqlDataAdapter(strsql, objconn)


    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "tblMedia")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    grid1.DataSource = dvlib
    grid1.DataBind()
End Sub

Sub bindEquip()
    Dim strsql As String
    'strsql = "SELECT * FROM tblEquip"

    strsql = "SELECT dbo.tblEquip.eqtype, dbo.tblEquip.eqtitle,
dbo.tblEquip.eqdescription, dbo.tblStatus.status, dbo.tblMaster.firstname,
dbo.tblMaster.lastname, "
    strsql &= "dbo.tblMaster.familyname FROM dbo.tblStatus INNER JOIN "
    strsql &= "dbo.tblEquip ON dbo.tblStatus.statusID = dbo.tblEquip.status INNER
JOIN "
    strsql &= "dbo.tblMaster ON dbo.tblEquip.personID = dbo.tblMaster.personID "

    Dim sdalib As New SqlDataAdapter(strsql, objconn)


    objconn.Open()
```

```
    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "tblEquip")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    grid1.DataSource = dvlib
    grid1.DataBind()
  End Sub

  Sub bindEquipsearch()
     Dim strsql As String
     Dim textinput As String
     'textinput = txtsearch.Text.Trim
     textinput = "%" & txtsearch.Text.Trim & "%"



     strsql = "SELECT dbo.tblEquip.eqtype, dbo.tblEquip.eqtitle, dbo.tblEquip.status,
dbo.tblEquip.eqdescription, dbo.tblStatus.status AS Expr1, dbo.tblMaster.firstname, "
     strsql &= "dbo.tblMaster.lastname, dbo.tblMaster.familyname "
     strsql &= "FROM dbo.tblStatus INNER JOIN "
     strsql &= "dbo.tblEquip ON dbo.tblStatus.statusID = dbo.tblEquip.status INNER
 JOIN "
     strsql &= "dbo.tblMaster ON dbo.tblEquip.personID = dbo.tblMaster.personID "
     strsql &= "WHERE (dbo.tblEquip.eqtype LIKE '" & textinput & "') OR
 (dbo.tblEquip.eqtitle LIKE '" & textinput & "') OR  (dbo.tblEquip.eqdescription LIKE '"
 & textinput & "') "

     Dim sdalib As New SqlDataAdapter(strsql, objconn)


     objconn.Open()

     sdalib.SelectCommand.CommandType = CommandType.Text
     dslib.Clear()
     sdalib.Fill(dslib, "dbo.tblEquip")
     dvlib = dslib.Tables(0).DefaultView
     objconn.Close()

     grid1.DataSource = dvlib
     grid1.DataBind()
   End Sub

   'book search
```

```
Sub bindBooksearch()
    Dim strsql As String
    Dim textinput As String
    textinput = "%" & txtsearch.Text.Trim & "%"

    strsql = "SELECT dbo.tblBook.booktype, dbo.tblBook.booktitle,
dbo.tblBook.bookauthor, dbo.tblBook.bookdescription, dbo.tblStatus.status,
dbo.tblMaster.firstname, "
    strsql &= "dbo.tblMaster.lastname, dbo.tblMaster.familyname "
    strsql &= "FROM dbo.tblStatus INNER JOIN "
    strsql &= " dbo.tblBook ON dbo.tblStatus.statusID = dbo.tblBook.status INNER
JOIN "
    strsql &= "dbo.tblMaster ON dbo.tblBook.personID = dbo.tblMaster.personID "
    strsql &= "WHERE (dbo.tblBook.booktype LIKE '" & textinput & "') OR
(dbo.tblBook.booktitle LIKE '" & textinput & "') OR  (dbo.tblBook.bookauthor LIKE '"
& textinput & "') OR  (dbo.tblBook.bookdescription LIKE '" & textinput & "') "

    Dim sdalib As New SqlDataAdapter(strsql, objconn)


    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "dbo.tblBook")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    grid1.DataSource = dvlib
    grid1.DataBind()
End Sub

Sub bindmediasearch()
    Dim strsql As String
    Dim textinput As String
    textinput = "%" & txtsearch.Text.Trim & "%"

    strsql = "SELECT dbo.tblMedia.mediatitle, dbo.tblMedia.mediatype,
dbo.tblMedia.mediaauthor, dbo.tblMedia.mediadescription, dbo.tblStatus.status, "
    strsql &= "dbo.tblMaster.firstname, dbo.tblMaster.lastname,
dbo.tblMaster.familyname "
    strsql &= "FROM dbo.tblStatus INNER JOIN "
    strsql &= "dbo.tblMedia ON dbo.tblStatus.statusID = dbo.tblMedia.status INNER
JOIN "
    strsql &= "dbo.tblMaster ON dbo.tblMedia.personID = dbo.tblMaster.personID "
    strsql &= "WHERE (dbo.tblMedia.mediatitle LIKE '" & textinput & "') OR
(dbo.tblMedia.mediaauthor LIKE '" & textinput & "') OR
```

```
(dbo.tblMedia.mediadescription LIKE '" & textinput & "') OR (dbo.tblMedia.mediatype
LIKE '" & textinput & "')"
        Dim sdalib As New SqlDataAdapter(strsql, objconn)

        objconn.Open()

        sdalib.SelectCommand.CommandType = CommandType.Text
        dslib.Clear()
        sdalib.Fill(dslib, "dbo.tblMedia")
        dvlib = dslib.Tables(0).DefaultView
        objconn.Close()

        grid1.DataSource = dvlib
        grid1.DataBind()

    End Sub

    'browse option returns all results in query
    Private Sub btnBrowse_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnBrowse.Click
        If ddloption.SelectedItem.Value = 1 Then
            bindgrid()
        ElseIf ddloption.SelectedItem.Value = 2 Then
            bindmedia()
        Else
            bindEquip()
        End If
    End Sub


    'search option returns queried results
    Private Sub btnsearch_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnsearch.Click
        If ddloption.SelectedItem.Value = 1 Then
            bindBooksearch()
        ElseIf ddloption.SelectedItem.Value = 2 Then
            bindmediasearch()
        Else
            bindEquipsearch()
        End If

    End Sub
```

```
    Private Sub ImageButton6_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton6.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton1_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton1.Click
        Response.Redirect("main.aspx")
    End Sub

    Private Sub ImageButton3_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton3.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton4_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton4.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton7_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton7.Click
        Response.Redirect("selmedia.aspx")
    End Sub

End Class
```

The Data Manipulation Source Codes

```
Imports System.Data.SqlClient
Public Class member
    Inherits System.Web.UI.Page
    Dim dvlib As New DataView
    Dim dslib As New DataSet
    Dim strsortexpr As String
    Protected WithEvents lbluser As System.Web.UI.WebControls.Label
    Protected WithEvents DataGrid4 As System.Web.UI.WebControls.DataGrid
    Protected WithEvents membergrid As System.Web.UI.WebControls.DataGrid
```

```
    Protected WithEvents newbook As System.Web.UI.WebControls.LinkButton
    Protected WithEvents ImageButton1 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton2 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton3 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton4 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton5 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton6 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton7 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents newequip As System.Web.UI.WebControls.LinkButton
    Protected WithEvents newmedia As System.Web.UI.WebControls.LinkButton
    Protected WithEvents Form1 As System.Web.UI.HtmlControls.HtmlForm
    Dim objconn As New
SqlConnection(ConfigurationSettings.AppSettings("ConnectionString"))

#Region " Web Form Designer Generated Code "

    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()

    End Sub
    Protected WithEvents Image1 As System.Web.UI.WebControls.Image
    Protected WithEvents DataGrid2 As System.Web.UI.WebControls.DataGrid
    Protected WithEvents DataGrid3 As System.Web.UI.WebControls.DataGrid
    Protected WithEvents Label2 As System.Web.UI.WebControls.Label
    Protected WithEvents Label3 As System.Web.UI.WebControls.Label
    Protected WithEvents Label4 As System.Web.UI.WebControls.Label
    Protected WithEvents DataGrid1 As System.Web.UI.WebControls.DataGrid

    'NOTE: The following placeholder declaration is required by the Web Form Designer.
    'Do not delete or move it.
    Private designerPlaceholderDeclaration As System.Object

    Private Sub Page_Init(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Init
      'CODEGEN: This method call is required by the Web Form Designer
      'Do not modify it using the code editor.
      InitializeComponent()
    End Sub

#End Region

    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
      'Put user code to initialize the page here
      'Session("personID")
      'Session("username")
      Dim name As String
```

```
    name = Session("username")
    'name = "ang001"
    Session("username") = name
    If Len(Session("personID")) = 0 Then
      lbluser.Text = "Welcome to the personal library system"
    Else
      lbluser.Text = "Welcome " & Session("username") & "!"
    End If

    If Me.IsPostBack = False Then
      Dim dsmsmber As DataSet = GetDataSet()
      BindGrid(dsmsmber)
    End If


    bindbook()
    bindmedia()
    bindequip()
  End Sub

  Public Function GetDataSet() As DataSet
    Dim str As String
    Dim dsmem As New DataSet
    str = "Select * FROM tblMaster where personID='" & Session("personID") & "'"

    Dim sdamember As New SqlDataAdapter(str, objconn)
    objconn.Open()

    sdamember.SelectCommand.CommandType = CommandType.Text
    dsmem.Clear()
    sdamember.Fill(dsmem, "tblMaster")
    objconn.Close()
    Return dsmem

  End Function

  Public Sub BindGrid(ByVal ds As DataSet)
    membergrid.DataSource = ds.Tables("tblMaster")
    Me.DataBind()
  End Sub

  Sub bindbook()
    Dim strsql As String

    strsql = "SELECT dbo.tblMaster.username, dbo.tblBook.booktype,
dbo.tblBook.booktitle, dbo.tblBook.bookauthor, dbo.tblBook.status,
dbo.tblBook.bookdescription, dbo.tblBook.transID FROM dbo.tblBook INNER JOIN
```

```
dbo.tblMaster ON dbo.tblBook.personID = dbo.tblMaster.personID WHERE
dbo.tblBook.personID ='" & Session("personID") & "'"

    Dim sdalib As New SqlDataAdapter(strsql, objconn)

    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "tblMaster")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    DataGrid1.DataSource = dvlib
    DataGrid1.DataBind()

    Response.Write("<br>")
    Response.Write("<br>")

  End Sub

  Sub bindmedia()
    Dim dvmed As New DataView
    Dim strsql As String
    Dim dsmed As New DataSet
    'strsql = "SELECT dbo.tblMedia.mediatitle, dbo.tblMedia.mediatype,
dbo.tblMedia.mediaauthor, dbo.tblMedia.mediadescription, dbo.tblMedia.transID,
dbo.tblStatus.status AS Expr1 FROM dbo.tblMaster INNER JOIN dbo.tblMedia ON
dbo.tblMaster.personID = dbo.tblMedia.personID INNER JOIN dbo.tblStatus ON
dbo.tblMedia.status = dbo.tblStatus.statusID WHERE dbo.tblMedia.personID ='" &
Session("user") & "'"
    strsql = " SELECT dbo.tblMedia.mediatitle, dbo.tblMedia.mediatype,
dbo.tblMedia.mediaauthor, dbo.tblMedia.mediadescription, dbo.tblStatus.status "
    strsql &= "FROM dbo.tblMedia INNER JOIN "
    strsql &= " dbo.tblStatus ON dbo.tblMedia.status = dbo.tblStatus.statusID "
    strsql &= "WHERE dbo.tblMedia.personID = '" & Session("personID") & "'"
    Dim sdalib As New SqlDataAdapter(strsql, objconn)

    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dsmed.Clear()
    sdalib.Fill(dsmed, "dbo.tblMedia")
    dvmed = dsmed.Tables(0).DefaultView
    objconn.Close()
```

```
        DataGrid2.DataSource = dvmed
        DataGrid2.DataBind()
    End Sub

    Sub bindequip()
        Dim strsql As String
        Dim dveq As New DataView
        Dim dseq As New DataSet
        'strsql = "SELECT dbo.tblMaster.username, dbo.tblEquip.eqtype,
dbo.tblEquip.eqtitle, dbo.tblEquip.status, dbo.tblEquip.eqdescription,
dbo.tblEquip.transID FROM dbo.tblMaster INNER JOIN  dbo.tblEquip ON
dbo.tblMaster.personID = dbo.tblEquip.personID WHERE dbo.tblMaster.username='"
& Session("user") & "'"

        strsql = "SELECT dbo.tblEquip.eqtype, dbo.tblEquip.eqtitle,
dbo.tblEquip.eqdescription,dbo.tblStatus.status "
        strsql &= "FROM dbo.tblEquip INNER JOIN "
        strsql &= "dbo.tblStatus ON dbo.tblEquip.status = dbo.tblStatus.statusID WHERE
dbo.tblEquip.personID='" & Session("personID") & "'"

        Dim sdalib As New SqlDataAdapter(strsql, objconn)

        objconn.Open()

        sdalib.SelectCommand.CommandType = CommandType.Text
        dseq.Clear()
        sdalib.Fill(dseq, "dbo.tblEquip")
        dveq = dseq.Tables(0).DefaultView
        objconn.Close()

        DataGrid3.DataSource = dveq
        DataGrid3.DataBind()
    End Sub

    'edit sub for membergrid
    Private Sub membergrid_EditCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
membergrid.EditCommand
        membergrid.EditItemIndex = e.Item.ItemIndex
        Dim dsmem As DataSet = GetDataSet()
        BindGrid(dsmem)
    End Sub

    'cancel sub for membergrid
    Private Sub membergrid_CancelCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
membergrid.CancelCommand
```

```
        membergrid.EditItemIndex = -1
        Dim dsmem As DataSet = GetDataSet()
        BindGrid(dsmem)
    End Sub

    'update sub for membergrid
    Private Sub membergrid_UpdateCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
membergrid.UpdateCommand
        'variables to hold values
        Dim newFirstName, newLastName, newFamilyName As String
        Dim newUsername, newPassword, newEmail, newWebsite As String
        Dim newtelno As String

        'newFirstName, newLastName, newFamilyName,newUsername, newPassword,
newEmail,newWebsite,newtelno
        'txtfirstname,txtlastname,txtfamilyname,
        newFirstName = CType(e.Item.FindControl("txtfirstname"), TextBox).Text
        newLastName = CType(e.Item.FindControl("txtlastname"), TextBox).Text
        newFamilyName = CType(e.Item.FindControl("txtfamilyname"), TextBox).Text

        'newUsername, newPassword, newEmail,newWebsite,newtelno
        newUsername = CType(e.Item.Cells(2).Controls(0), TextBox).Text
        newPassword = CType(e.Item.Cells(3).Controls(0), TextBox).Text
        newEmail = CType(e.Item.Cells(4).Controls(0), TextBox).Text
        newWebsite = CType(e.Item.Cells(5).Controls(0), TextBox).Text
        newtelno = CType(e.Item.Cells(6).Controls(0), TextBox).Text

        Dim personID As String
        personID = e.Item.Cells(0).Text
        Dim sdamember As New SqlClient.SqlDataAdapter
        Dim commember As New SqlClient.SqlCommand
        Dim update As String
        update = "UPDATE tblMaster SET username='" & newUsername & "',firstname='"
& newFirstName & "', "
        update &= "lastname='" & newLastName & "',familyname='" & newFamilyName
& "',email='" & newEmail & "', "
        update &= "password='" & newPassword & "', website='" & newWebsite &
"',telno=" & newtelno & ""
        update &= "WHERE personID='" & personID & "'"

        objconn.Open()

        commember.Connection = objconn
        commember.CommandText = update
        commember.CommandType = CommandType.Text
```

```
    commember.ExecuteNonQuery()
    objconn.Close()

    'cancel edit mode
    membergrid.EditItemIndex = -1

    Dim ds As DataSet = GetDataSet()
    BindGrid(ds)

  End Sub


  Private Sub newbook_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles newbook.Click
    Response.Redirect("newbook.aspx")
  End Sub

  Private Sub newmedia_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles newmedia.Click
    Response.Redirect("newmedia.aspx")
  End Sub

  Private Sub newequip_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles newequip.Click
    Response.Redirect("newequip.aspx")
  End Sub


  Private Sub ImageButton6_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton6.Click
    Response.Redirect("newlogin.aspx")
  End Sub

  Private Sub ImageButton1_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton1.Click
    Response.Redirect("main.aspx")
  End Sub

  Private Sub ImageButton3_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton3.Click
    Response.Redirect("newlogin.aspx")
  End Sub

  Private Sub ImageButton4_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton4.Click
    Response.Redirect("newlogin.aspx")
```

```
    End Sub

    Private Sub ImageButton7_Click(ByVal sender As System.Object, ByVal e As
 System.Web.UI.ImageClickEventArgs) Handles ImageButton7.Click
        Response.Redirect("selmedia.aspx")
    End Sub

End Class
```

Example of the Resources Module Source Codes

```
Imports System.Data.SqlClient
Public Class newbook
    Inherits System.Web.UI.Page
    Dim objconn As New
SqlConnection(ConfigurationSettings.AppSettings("ConnectionString"))
    Dim dvlib As New DataView
    Protected WithEvents grid1 As System.Web.UI.WebControls.DataGrid
    Protected WithEvents ImageButton1 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton2 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton3 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton4 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton5 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton6 As System.Web.UI.WebControls.ImageButton
    Protected WithEvents ImageButton7 As System.Web.UI.WebControls.ImageButton
    Dim dslib As New DataSet

#Region " Web Form Designer Generated Code "

    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()

    End Sub
    Protected WithEvents Label1 As System.Web.UI.WebControls.Label
    Protected WithEvents Label2 As System.Web.UI.WebControls.Label
    Protected WithEvents Label3 As System.Web.UI.WebControls.Label
    Protected WithEvents txttitle As System.Web.UI.WebControls.TextBox
    Protected WithEvents txtauthor As System.Web.UI.WebControls.TextBox
    Protected WithEvents txttype As System.Web.UI.WebControls.TextBox
    Protected WithEvents btninsert As System.Web.UI.WebControls.Button
    Protected WithEvents rvtxttitle As
System.Web.UI.WebControls.RequiredFieldValidator
    Protected WithEvents rvtxtauthor As
System.Web.UI.WebControls.RequiredFieldValidator
```

```
    Protected WithEvents rvtxttype As
System.Web.UI.WebControls.RequiredFieldValidator
    Protected WithEvents Image1 As System.Web.UI.WebControls.Image
    Protected WithEvents Label4 As System.Web.UI.WebControls.Label
    Protected WithEvents Label5 As System.Web.UI.WebControls.Label
    Protected WithEvents DropDownList1 As
System.Web.UI.WebControls.DropDownList
    Protected WithEvents lbluser As System.Web.UI.WebControls.Label
    Protected WithEvents txtDesc As System.Web.UI.WebControls.TextBox
    Protected WithEvents Label6 As System.Web.UI.WebControls.Label

    'NOTE: The following placeholder declaration is required by the Web Form Designer.
    'Do not delete or move it.
    Private designerPlaceholderDeclaration As System.Object

    Private Sub Page_Init(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Init
        'CODEGEN: This method call is required by the Web Form Designer
        'Do not modify it using the code editor.
        InitializeComponent()
    End Sub

#End Region

    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'Put user code to initialize the page here
        If Len(Session("personID")) = 0 Then
            lbluser.Text = "Welcome to the personal library system"
        Else
            lbluser.Text = "Welcome " & Session("username") & "!"
        End If

        If Me.IsPostBack = False Then
            Dim dsbook As DataSet = GetDataSet()
            BindGrid(dsbook)
        End If

        bind()
        'bindgrid1()
    End Sub


    Public Function GetDataSet() As DataSet
        Dim str As String
        Dim dsbook As New DataSet
        str = "Select * FROM tblBook where personID='" & Session("personID") & "'"
```

```
    'str = "SELECT bookID, booktype, booktitle, bookauthor, status, bookdescription"
    'str &= "FROM dbo.tblBook"
    'str &= "WHERE personID ='" & Session("personID") & "'"

    Dim sdamember As New SqlDataAdapter(str, objconn)
    objconn.Open()

    sdamember.SelectCommand.CommandType = CommandType.Text
    dsbook.Clear()
    sdamember.Fill(dsbook, "tblBook")
    objconn.Close()
    Return dsbook

  End Function

  'sub that binds dataset to gtrid
  Public Sub BindGrid(ByVal ds As DataSet)
     grid1.DataSource = ds.Tables("tblBook")
     Me.DataBind()
  End Sub


   Private Sub btninsert_Click(ByVal sender As Object, ByVal e As System.EventArgs)
Handles btninsert.Click
      Dim statusID As String
      statusID = DropDownList1.SelectedItem.Value
      Dim str As String
      Dim sqldtadp As SqlClient.SqlDataAdapter
      Dim InsCom As New SqlClient.SqlCommand
      'create insert statement
      str = "INSERT INTO tblBook (personID, booktype, booktitle, bookauthor, status,
 bookdescription)"
      str &= " Values('" & Session("personID") & "','" & txttype.Text & "','" &
 txttitle.Text & "','" & txtauthor.Text & "','" & statusID & "','" & txtDesc.Text & "')"


      objconn.Open()

      InsCom.Connection = objconn
      InsCom.CommandText = str
      InsCom.CommandType = CommandType.Text

      InsCom.ExecuteNonQuery()
      objconn.Close()

    End Sub
```

```
'this was a dumb function to test out the sql
Sub bindgrid1()
    Dim strsql As String
    strsql = "Select * FROM tblMaster where personID='" & Session("personID") & "'"
    'strsql = "SELECT dbo.tblBook.bookID AS Expr1, dbo.tblBook.booktype AS
Expr2, dbo.tblBook.booktitle AS Expr3, dbo.tblBook.bookauthor AS Expr4,"
    'strsql &= "dbo.tblBook.bookdescription AS Expr5, dbo.tblStatus.status"
    'strsql &= "FROM dbo.tblBook INNER JOIN dbo.tblStatus ON dbo.tblBook.status
= dbo.tblStatus.statusID"
    'strsql &= "WHERE dbo.tblBook.personID ='" & Session("personID") & "'"

    Dim sdalib As New SqlDataAdapter(strsql, objconn)

    objconn.Open()

    sdalib.SelectCommand.CommandType = CommandType.Text
    dslib.Clear()
    sdalib.Fill(dslib, "tblBook")
    dvlib = dslib.Tables(0).DefaultView
    objconn.Close()

    grid1.DataSource = dvlib
    grid1.DataBind()
End Sub

'the dropdown box for status
Private Sub bind()
    Dim str As String
    Dim dtstat As New DataTable
    Dim dsstat As New DataSet

    str = "SELECT status,statusID FROM tblStatus ORDER BY statusID"

    Dim adpstat As New SqlDataAdapter(str, objconn)

    objconn.Open()
    adpstat.SelectCommand.CommandType = CommandType.Text
    dsstat.Clear()
    adpstat.Fill(dsstat, "tblStatus")
    objconn.Close()

    DropDownList1.DataSource = dsstat
    DropDownList1.DataValueField = "statusID"
    DropDownList1.DataTextField = "status"
    DropDownList1.DataBind()
```

```
End Sub

Private Sub grid1_CancelCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
grid1.CancelCommand
    grid1.EditItemIndex = -1
    Dim dsmem As DataSet = GetDataSet()
    BindGrid(dsmem)
End Sub

Private Sub grid1_EditCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
grid1.EditCommand
    grid1.EditItemIndex = e.Item.ItemIndex
    Dim dsmem As DataSet = GetDataSet()
    BindGrid(dsmem)
End Sub

Private Sub grid1_UpdateCommand(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridCommandEventArgs) Handles
grid1.UpdateCommand
    Dim bookID, booktype, booktitle As String
    Dim bookauthor, bookdescription As String
    Dim status As String

    'bookID,booktype,booktitle,bookauthor,status,bookdescription
    'newFirstName = CType(e.Item.FindControl("txtfirstname"), TextBox).Text
    status = CType(e.Item.Cells(4).Controls(0), TextBox).Text
    booktype = CType(e.Item.Cells(1).Controls(0), TextBox).Text
    booktitle = CType(e.Item.Cells(2).Controls(0), TextBox).Text
    bookauthor = CType(e.Item.Cells(3).Controls(0), TextBox).Text
    bookdescription = CType(e.Item.Cells(5).Controls(0), TextBox).Text

    bookID = e.Item.Cells(0).Text
    Dim sdamember As New SqlClient.SqlDataAdapter
    Dim commember As New SqlClient.SqlCommand
    Dim update As String
    'bookID,booktype,booktitle,bookauthor,status,bookdescription
    update = "UPDATE tblBook SET booktype='" & booktype & "',booktitle='" &
booktitle & "', "
    update &= "bookauthor='" & bookauthor & "',status=" & status &
",bookdescription='" & bookdescription & "' "
    update &= "WHERE bookID=" & bookID & ""
```

```
        objconn.Open()
        commember.Connection = objconn
        commember.CommandText = update
        commember.CommandType = CommandType.Text

        commember.ExecuteNonQuery()
        objconn.Close()

        grid1.EditItemIndex = -1

        Dim ds As DataSet = GetDataSet()
        BindGrid(ds)
    End Sub

    Private Sub ImageButton6_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton6.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton1_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton1.Click
        Response.Redirect("main.aspx")
    End Sub

    Private Sub ImageButton3_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton3.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton4_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton4.Click
        Response.Redirect("newlogin.aspx")
    End Sub

    Private Sub ImageButton7_Click(ByVal sender As System.Object, ByVal e As
System.Web.UI.ImageClickEventArgs) Handles ImageButton7.Click
        Response.Redirect("selmedia.aspx")
    End Sub
End Class
```

REFERENCES

References

[1] Client Server Software Architectures – An Overview

http://www.sei.cmu.edu/str/descriptions/clientserver_body.html

[2] Client

http://www.webopedia.com/TERM/C/client.html

[3] Server

http://www.webopedia.com/TERM/C/server.html

[4] Purpose and Origin of Client/Server

http://www.sei.cmu.edu/str/descriptions/clientserver.html#777375

[5] One - Tier Architecture

http://www.javaworld.com/javaworld/jw-01-2000/jw-01-ssj-tiers-p2.html

[6] Two – Tier Architecture

http://www.sei.cmu.edu/str/descriptions/clientserver.html#777375

[7] Three – Tier Architecture

http://www.javaworld.com/javaworld/jw-01-2000/jw-01-ssj-tiers-p3.html

[8] Introduction to SSL

http://developer.netscape.com/docs/manuals/security/sslin/contents.htm

[9] Web Servers – Apache

http://www.serverwatch.com/stypes/servers/index.php/15877

[10] Web Servers – Microsoft Internet Information Server

http://www.serverwatch.com/stypes/servers/article.php/15835_1434241

[11] Web Servers – Lotus Domino

[12] Operating System

http://www.pcwebopedia.com/TERM/o/operating_system.html

[13] Operating System Types

http://www.computerhope.com/os.htm

[14] Windows 98

http://www.pcwebopedia.com/TERM/W/Windows_98.html

[15] An overview of the UNIX Operating System

http://www.bell-labs.com/history/unix/tutorial.html

[16] An Introduction to Linux

http://www.linux.org.uk/WhatIs.html

[17] Oracle Technology Network

http://www.otn.oracle.com

[18] PHP/MySQL Tutorial

http://www.hotwired.lycos.com/webmonkey/99/21/index2a.html

[19] Microsoft SQL Server - Windows® CE Edition

http://www.microsoft.com/sql/ce/productinfo/overview.asp

[20] Open Database Connectivity

http://searchvb.techtarget.com/sDefinition/0,,sid8_gci214133,00.html

[21] Java Server Pages

http://java.sun.com/products/jsp/

[22] Active Server Pages

http://searchwin2000.techtarget.com/sDefinition/0,,sid1_gci213787,00.html

[23] ActiveX Data Objects

http://searchwin2000.techtarget.com/sDefinition/0,,sid1_gci213761,00.html

[24] OLE DB

http://searchdatabase.techtarget.com/sDefinition/0,,sid13_gci214419,00.html

[25] ASP.NET

http://searchdatabase.techtarget.com/sDefinition/0,,sid13_gci509342,00.html

[26] University of California, Berkeley Digital Library Project

http://elib.cs.berkeley.edu/pl/about.html

[27] Brarydog.Net

http://www.brarydog.net/

[28] BiotechMesa.Net Personal Library

http://www.biotechmesa.net/plmain.shtml

[29] 123 ASPX

http://www.123aspx.com

[30] Planet Source Code

http://planetsourcecode.com

[31] Microsoft ASP.Net

http://asp.net/Default.aspx?tabindex=0&tabid=1