Faculty of Computer Science and Information Technology

UNIVERSITY MALAYA

WXES3182 Projek Ilmiah Tahap Akhir II

Panel Doctor System (PDS)

For Employee Medical Information System

Prepared by:

TEH BOON HENG

WEK990003

(SOFTWARE ENGINEERING)

Supervised by

Prof. Madya Dr. Lee Sai Peck

Session 2001/2002

ABSTRACT

Panel Doctor System (**PDS**) is a distributed system developed for the companies to keep track and control the company expenses on the employees' medical expenses.

Most of the companies have their own medical consultants that provide medical treatment to the employees of the company. It is one of the welfare that a company had provided to protect their employees. Hence, every year a company will allocate a budget for the medical expenses that spend on their employees. Unfortunately, many companies don't have a system to help them keep track of these expenses. The filing records are not arranged and the management find very difficult to do decision making on the aspect of medical expenses budget.

PDS, a distributed system is a solution to all the problem that exists above. The **PDS** have functionalities on keep all the records of the employees, company approved Hospital, clinics information and panel doctors information in the company database that been create specialize to meet the company needs. Other tools that are available include tools to manage the amount of user, and to change password. In addition, the **PDS** helps the top management to do the decision making with several reports generated by the system. The reports of **PDS** are generate by the system using the data in the database and have been analyze accurately and correctly by the system.

Basically, the **PDS** comprises 8 modules and each module has it's own functionalities. There are 2 categories of users in **PDS** which is normal users (mostly use by the data entry clerks) and super users (administrator). Conceptually, some functionality will be hide for normal user view. This is to protect the confidential information in the database.

ETS is developed using Visual Basic technologies running on Microsoft Windows NT Server 4.0 platform-utilizing database created and stored using SQL Server 6.0.

ACKNOWLEDGEMENTS

Firstly, I would like to express my heartiest gratitude to my supervisor, Prof madya.Dr. Lee Sai Peck for her invaluable advice, guidance and professional supervision and not forgetting his willingness in sharing his knowledge and thoughts with me throughout the course of this project.

Then, I would like to express my thankfulness to my project moderator, Cik Norazlina Khamis for attending my presentation and making it a success.

I am very grateful because a found many colleagues which also giving me support when I need them. Thanks for their valuable ideas and guidance. Their encouragement had contributed to the success and completion of this project.

Last but not the least, I would like to dedicate a special thanks to my family and all my beloved for their wonderful understanding that sits besides me.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 PROJECT INTRODUCTION

In the organization, it is a need to audit all the expenses that involve the employee of the company. These include the salary of the employee, the charge on the meeting, employee medial charges and others.

This project is called the Panel Doctor System (**PDS**) for Management of Employee's Medical Reports and Billing. The project is a distributed computing system. Where it involved the basic 2 tier software Server and Client relationship architecture.

The employee medical management is a need because many of the company expenses were use under these employee medical funds. This is why a tracking on it has to be done. This project will involve the object oriented programming language, which is Visual Basic and the support of the SQL database. The expecting duration to conclude the application is 3 months.

1.2 PROJET OBJECTIVE

The fundamental objective of this project is to develop a system to enable an organization to keep track and update their information on employees' medial record and the expenses spend on panel doctor billing. This is not just any means of information system but it also meets the objective stated:

i. Helping to examine the quality of the expenses in the organization by building chart, graph and others analysis forms of materials.

ii. Enhanced the quality of administrating by putting all the information in a well organized database.

iii. To create a better environment for company to update their employee medical record and doing data syntheses.

In summary, this project will ease a company process on tracking employee's health condition and company expenses on employees medical billing.

1.3 PROJECT SCOPE

The scope of the project means the coverage of the project as in what is the limitation of the application and what are the boundaries set upon the application.

1. The application will involve 2 parties, which is normal user and administrator. The limitation o the application is that normal user such as department clerk can only accessed to specific part of the application whereas for super user such as administrator, he will view through all modules in the application.

2. To achieve the goal on security, an administrator has the right to approve a new user's registration or reject it because of unforeseen problems.

3. Since the application did not set a quota for maximum users, it is unlimited to users as long as they are registered with the system after approve by administrator.

]Below are stated the scopes; boundaries of what can be done and what is not in 2 viewpoints namely the administrator (both super and normal administrator) and the user.

Administrator Section

In this section, administrator can perform the following functions:

i) Add, update, delete and view employee's medical records.

ii) Add, update, delete and view panel doctor billing record.

iii) Add, update, delete, view and search panel doctor data.

iv) Add, update, delete, view and search employees data.

v) View reports of annual, monthly, weekly, and daily panel doctor billing.

vi) View employee's medical billing report on various queries as generate by the system.

vii) To approve or reject new registration.

viii) To add a new administrator or new user.

ix) To delete a new administrator or existing user.

User Section

For users (Department Clerks), the following functions were allowed:

i) Add, and update employee's medical records.

ii) Add, and update company approved panel doctor billings.

iii) Add, and update panel doctor information.

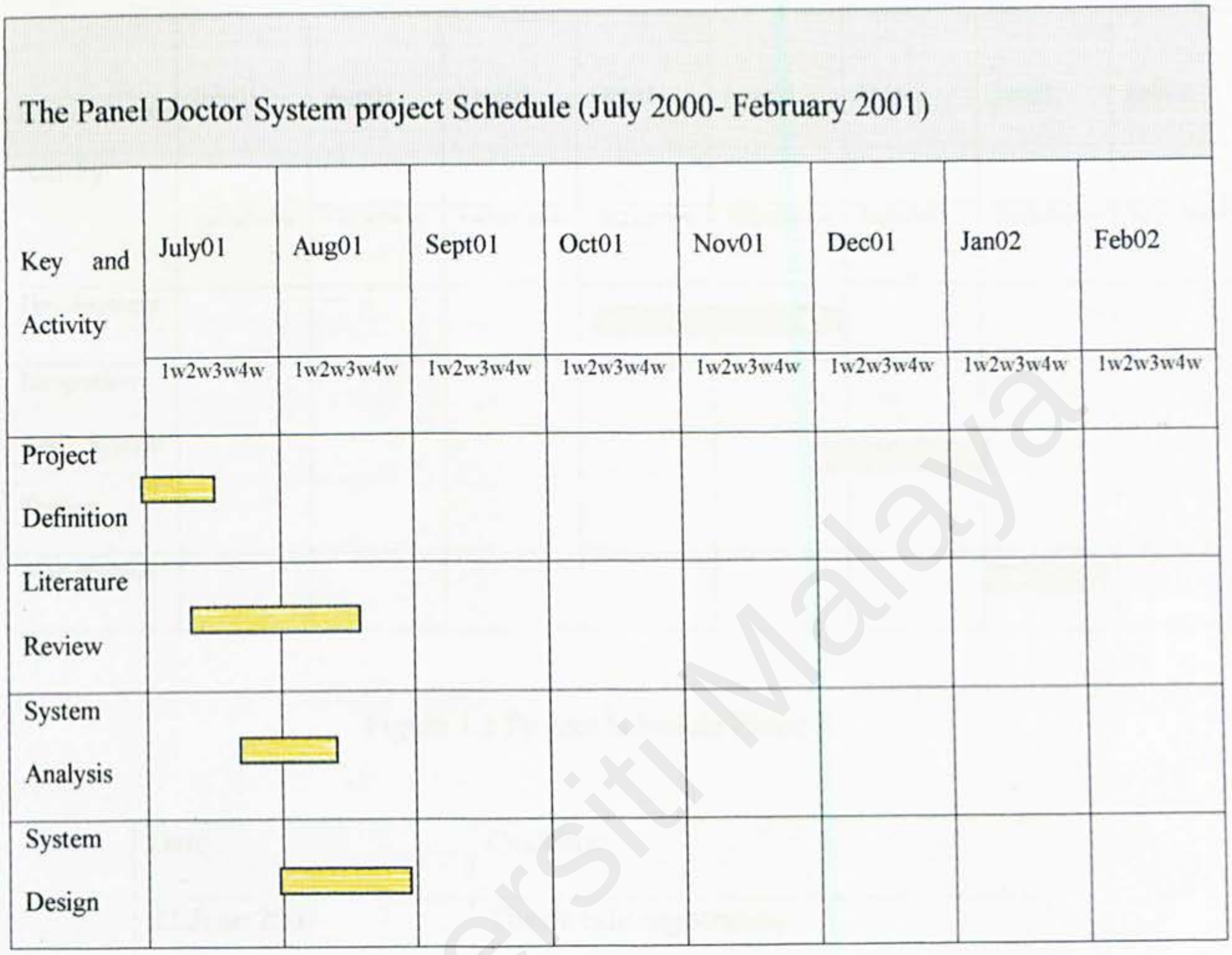
iv) Add, and update company approved hospital information.

v) To maintain data integrity.

1.4 PROJECT SCHEDULE

This project will be separate to two phrases, which is the system proposal and the software product. In the early 4 month, it is define as the period on studying the software module and preparing the documentation on the software and will be present to the supervisor as assign. And the following 4-month will be the real time project development period. This duration of time are assign to doing the coding of the software according to the modules which had been done in the early 4 month.

Despite of it, there will be allocate a period of time to do the debugging and real time testing on the program in the laboratory. And the time is set in the end of the 4-month system development period.



The Panel Doctor System project Schedule (July 2000- February 2001)

Key and Activity	July01	Aug01	Sept01	Oct01	Nov01	Dec01	Jan02	Feb02
	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w
Project Definition	▨							
Literature Review	▨	▨						
System Analysis	▨	▨						
System Design		▨						

Figure 1.1 Project Schedule Phase I

The Panel Doctor System project Schedule (July 2000- February 2001)								
Key and Activity	July01	Aug01	Sept01	Oct01	Nov01	Dec01	Jan02	Feb02
	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w	1w2w3w4w
Development				▨	▨			
Integration And System Testing					▨	▨		
Maintenance							▨	

Figure 1.2 Project Schedule Phase II

Date	Occasion
22 June 2001	Thesis title registration
8 August 2001	VIVA seminar by Mr. C.S. Woo
16 August 2001	VIVA presentation
30 August 2001	Submission of thesis report

Figure 1.3 Date To remember

1.5 PROJECT ORGANIZATION

The text of this report is organized into four chapters:

Chapter 1 Introduction aims to give user a brief and overview picture of the panel doctor information system and the **PDS** application in this project that will be developed. Besides, user will also get to know the objective and the scope of the project. User may know approximately when the project started and is going to be complete through the project schedule.

Chapter 2 Literature review gives brief explanations to each research area such as the field of software development, explanation on the theoretical background of the software development, the various operating system and the client/server architecture.

Chapter 3 System Analysis mainly describes the functional requirements and non-functional requirements for this system.

Chapter 4 System Design focus on the design issues involved in developing the project, such as System Functionality Design, Algorithm Design, and Database Design.

1.6 EXPECTED OUTCOME

Expectation of **PDS** should be made after consider the technologies available as well as the timeline of the project. What is expect here with the outcomes of the **PDS** are shown as below:

1. Complete **PDS** Application Modules

All the application Modules in PDS should be well develop with bugs free and should be the guide for future module development. Besides, the modules will have their own project documentation, system requirement specification, user manual and also system documentation.

2. Complete **PDS** core system

All the application modules should be able to play their role correctly and accurately. System Analysis, System design, rules and documentation standard should be well defined. The system and unit testing shall not emerge any unexpected errors.

3. Reliability of the system

The system shall be reliable to the user. The results generated by analyze the internal data should be accurate and always keep the data consistent.

4. Ease the burden of the users.

PDS shall be able to help the administrator of a company to analyze the data and provide sufficient reports. This will ease the burden of the administrator to analyze the data manually. Besides, clear and well display reports help the administrator to do the decision-making more effectively.

5. Private Data Protected

There is a security features to keep the private data protected. The private data shall be encrypting in the database and no one except the authorized user can view the content of the data.

6. Utilities

Utilities such as change password, search data shall be exist in the **PDS**. This features help to enhance the usability of the application.

7. User Friendly Environment

The whole Application shall be user-friendly to the user because not all the users are computer-educated. Icons shall be added to help the user to use the function in the applications more effectively.

8. Foundation for Future Growth

Well implement foundation architecture of **PDS** and the final implementation will enable future enhancement. This will include addition of modules as well as add-on new functionality.

CHAPTER 2: LITERATURE REVIEW

Before the development process started, researches have been carried out to do a comparison review. The literature review process helps to analyze the strengths and weaknesses of all the information that were found.

The literature survey can be categorized into 4 sections, which are:

i. Development Operating System

ii. Client/Server Architecture

iii. Development Server

iv. Development Software

2.1 DEVELOPMENT OPERATING SYSTEM

2.1.1 Microsoft Windows NT Server 4.0

Windows NT server 4.0 was known as network operating system because it posses multi purpose servers. Windows NT Server well known as more stabile operating system compares to others Microsoft windows operating system. Besides that, it is more powerful where it builds application faster and easier than the others do.

Windows NT Server 4.0 can be evolve into a very powerful environment in building application of all kind when it integrated with other Windows NT related product such as Windows NT Option Pack and Back office family. On the other hand, this particular operating system is easy to use where it Graphical User Interface is user friendly with many icons which is easy to navigate between applications.

Another criterion of Windows NT Server 4.0 is that it is high security protected. Windows NT has been approved as being C2-compliant where it security features includes object protection on a user and group based, and enable to audit security-related events. Lastly, Windows NT Server 4.0 is more reliable by comparing with Windows 95 and Windows 98. This is because Windows NT uses separate memory apace for different 16 bit applications where it will not affect other applications if one application fails.

2.2 CLIENT/SERVER ARCHITECTURE

2.2.1 Client/Server Fundamentals

2.2.1.1 Definitions

Client/Server model is a concept for describing communications between computing processes that are classified as service consumers (clients) and service providers (servers). The basic features of a C/S model are:

1. Clients and servers are functional modules with well-defined interfaces (i.e., they hide internal information). The functions performed by a client and a server can be implemented by a set of software modules, hardware components, or a combination thereof.

2. Each client/server relationship is established between two functional modules when one module (client) initiates a service request and the other (server) chooses to respond to the service request.

3. Information exchange between clients and servers is strictly through messages (i.e., no information is exchange through global variables).

4. Clients and servers typically reside on separate machines connected through a network. Conceptually, clients and servers may run on the same machine or on separate machines.

Client/server applications, an area of vital importance to us, employ the C/S model to deliver business aware functionality. C/S applications provide a powerful and flexible mechanism for organizations to design applications to fit the business needs.

2.2.1.2 Client/Server – A Special Case of Distributed Computing

Conceptually, client/server model is a special case of distributed-computing model. A Distributed Computing System (DCS) is a collection if autonomous computers interconnected through a communication network to achieve business function. Technically, the computers do not share main memory so that the information can not be transferred through global variables. The information (knowledge) between the computers is exchanged only through messages over a network.

The restriction of no shared memory and information exchange through messages is of key importance because it distinguishes between DCS and shared memory multiprocessor computing systems.. The definition requires that the computers have to work together and cooperate with each other to satisfy enterprise needs

Distributed computing can be achieved through one or more of the following:

1 File transfer model

2 Client/server model

File transfer model is one of the oldest models to achieve distributed computing at a very minimal level. Basically, programs at different computers communicate with each other by using file transfer. Although this is a very old and extremely limited model of distributed computing, it is still used to support loosely coupled distributed computers.

For example, media clips, news items, and portions of corporate databases are typically exchanged between remote computers through file transfers; and e-mail is used frequently to exchange files through embedding and attachments.

The C/S model is state of the market and state of the practice for distributed computing at the time of this writing. C/S model, as stated previously, allows application processes at different sites to interactively exchange messages and is thus a significant improvement over the file transfer model. Initial versions of C/S model utilized the remote procedure call paradigm that extends the scope of a local procedure call. At present, the C/S model is increasingly utilizing the distributed objects paradigm that extends the scope of local object paradigm

2.2.1.3 Client/Server Architectures

Client/server architecture provides the fundamental framework that allows many technologies to plug in for the applications of 1990s and beyond. Clients and servers typically communicate with each other by using one of the following paradigms:

Remote Procedure Call (RPC). In this paradigm, the client process invokes a remotely located procedure (a server process), the remote procedure executes and sends the response back to the client. The remote procedure can be simple or complex. Each request/response of an RPC is treated as a separate unit of work, thus each request must carry enough information needed by the server process. RPCs are supported widely at present.

Remote Data Access (RDA). This paradigm allows client programs and/or end-user tools to issue ad hoc queries, usually SQL, against remotely located databases. The key technical difference between RDA and RPC is that in a RDA the size of the result is not known because the result of an SQL query could be one row or thousands of rows. Database vendors heavily support RDA.

Queued Massage Processing (QMP). In this paradigm, the client message is stored in a queue and the server works on it when free. The server stores ("put") the response in another queue and the client actively retrieves ("get") the responses from this queue. This model, used in many transactions processing systems, allows the clients to asynchronously send requests to the server. Once a request is queued, the request is processed even if the sender is disconnected (intentionally or due to a failure). QMP support is becoming commonly available.

Initial implementations of client/server architecture were based on the "two-tiered" architectures. Although a given C/S application can be architects in any of this configuration, the remote data and distributed program configurations are used heavily at present. The remote data configuration at present is very popular for departmental applications and is heavily supported by numerous database vendors. Most data warehouses also use a remote data configuration because the data warehouse tools can reside on user workstations and issue remote SQL calls to the data warehouse. However, the distributed program configuration is very useful for enterprise wide applications, because the application programs on both sides can exchange information through messages.

2.3 DEVELOPMENT SERVER

2.3.1 Relational Database Management System (RDBMS)

RDBMS is a standard or Client-server database computing. It is based on the relational model that originated in papers published by Dr. E.F.Codd in 1969. In a RDBMS, data is organized in a row/column manner and is stored in a table. The number o RDBMS vendor has increased over the years as Client-Server has grown in popularity. Following is a brief summary of popular RDBMS vendors.

2.3.1.1 Oracle 8.0

Oracle server is a multi-user relational database management system (DBMS) that runs on numerous operating systems. Oracle 8.0, the worlds most powerful object relational database is the heart of the open, standards-based Network Computing Architecture. Network Computing Architecture allows IT organizations to spend less time struggling with interoperability issues and more time focusing on deploying solutions.

Standard-based network architecture makes it possible to introduce objects into mainstream enterprise environments. Oracle 8.0,s development environment allows users to ease into object-relational functionality while providing the industrial strength properties required by network-based applications.

Oracle 8.0 delivers all the major platform require by network based architecture which involve multiple hardware and software platforms, this include UNIX and NT. Oracle 8.0 data management, security, reliability, and ease of use, is unique designed to meet the demands of the network era.

On top of the positive feature of Oracle, it is known with it features such as security and compatibility. In the view of security, Oracle has many level of security and has seamless integration with Windows NT, therefore provides security, a web application environment. As for compatibility, Oracle compatible with many operating system and it has it's own product line such as Portal-to-go, JDeveloper and others.

2.3.1.2 Microsoft's SQL Server

SQL Server product was originally developed by Sybase in the mid-1980s. In 1988, Sybase released SQL Server for OS/2 with its partner, Microsoft. While Sybase and Microsoft ended their partnership in 1994, Microsoft's SQL Server has grown to be a huge success in the RDBMS market. Microsoft has been successful in combining performance, support for multiple platforms, and ease of use.

There are several advantages of Microsoft's SQL Server. Ease-of-use is accomplished through SQL Server's graphical management tools. The SQL Server allows two billion tables within each of 32,767 databases to be defined. It allows user to define up to 250 columns for each table. SQL Server allows users to combine columns from as many as 16 tables in a single query.

Structured Query Language (SQL), the query language developed by IBM in the 1970s, has become the de facto standard database query language or relational databases. The dialect o /SQL that user use with SQL Server is the Transact-SQL, which Microsoft implement as a core component o SQL Server.

Support for multiple platforms is accomplished through Microsoft's NT operating system, which runs on the Intel, RISC, and other chip sets. However the SQL Server database must be installed on the Windows NT platform.

2.4 DEVELOPMENT SOFTWARE

2.4.1 Java

Java is known as a very powerful programming language developed by Sun Corporation in 1991. It was originally designed to run on small electronic appliances such as set-top boxes (connecting TV to a digital video server) or PDA's (personal digital assistants or hand-held computers).With the development of the Web, Sun began to look at possibilities for integrating Java (then called "Oak") into Internet-delivered applications. "HotJava" was developed in 1994 as a sophisticated Web browser (which run in Unix computers) capable of running small programs called applets. Applets are written in Java and can be sent over the Web like HTML files to the browser, where they run in the browser window, just like HTML documents.

Java has the potential to run not just on desktop computers but on a variety of other microprocessor-driven devices. Java OS, for example, is a version of Java developed to run on PDA's. Java, in comparison to other programming languages, is small, efficient, and versatile. Its versatility lies not only in the great variety of programs it can be used to create, but also in the fact that it is easily transportable from one environment or operating system to another. Java programs need only be compiled once for all operating systems. This is because for each platform a specific Java interpreter (called a "virtual machine") runs the Java program. In other words, there is currently a kind of translator which sits between Java programs and computer's operating system.

With the "write once, run everywhere" nature of Java. Microsoft's was tying its implementation of Java tightly to the Windows operating system. The idea is to optimize Java performance and features for Windows users. Another versatile feature of Java is the fact that it can be used on any platform to create programs that are platform-independent. There is Java development tools available for Unix, Windows and Macintosh.

2.4.2 Visual Basic

VISUAL BASIC is a high level programming language evolved from the earlier DOS version called BASIC. BASIC means **B**eginners' **A**ll-purpose **S**ymbolic **I**nstruction **C**ode. It is a fairly easy programming language to learn. The codes look a bit like English Language. Different software companies produced different version of BASIC, such as Microsoft QBASIC, QUICKBASIC, GWBASIC, IBM BASICA and so on.

VISUAL BASIC is a VISUAL and events driven Programming Language. These are the main divergence from the old BASIC. In BASIC, programming is done in a text-only environment and the program is executed sequentially. In VISUAL BASIC, programming is done in a graphical environment. Because users may click on a certain object randomly, so each object has to be programmed independently to be able to response to those actions (events). Therefore, a VISUAL BASIC Program is made up of many subprograms, each has its own program codes, and each can be executed independently and at the same time each can be linked together in one way or another.

2.4.3 Power Builder

PowerBuilder is a graphical application development environment. It can easily develop powerful graphical applications that access databases. PowerBuilder provides all the tools need to build industrial-strength applications, such as order entry, accounting, and manufacturing systems. The user interface of a PowerBuilder application consists of menus and windows that users interact with. PowerBuilder applications include all the standard window controls such as buttons, dropdown, checkboxes, listboxes, and edit boxes as well as special PowerBuilder controls that make applications easy to develop and easy to use.

PowerBuilder applications are event-driven, what is meant was users control what happens by the actions they take. For example, when a user clicks a button, chooses an item from a menu, or enters data into an edit box, one or more events are triggered. Users write scripts that specify the processing that should happen when events are triggered. For example, buttons have a Clicked event. Users can write a script for a button's Clicked event that specifies what happens whenever the user clicks the button. Similarly, edit boxes have a Modified event, which is triggered each time the user changes a value in the box.

PowerBuilder use Power Script as it PowerBuilder language and user write script using it. Scripts consist of Power Script commands, functions, and statements that perform processing in response to an event. For example, the script for a button's Clicked event might retrieve and display information from the database; the script for an edit box's Modified event might evaluate the data and perform processing based on the data. The execution of an event script can also cause other events to be triggered. For example, the script for a Clicked event in a button might open another window, which triggers the Open event in that window. Power Script provides a rich assortment of built-in functions that enable to act upon the various components of their application. For example, there is a function to open a window, a function to close a window, a function to enable a button, a function to retrieve data, a function to update the database, and so on. In addition, users can build their own functions to define processing unique to their application.

PowerBuilder is an object-oriented programming. Each menu or window user create with PowerBuilder is a self-contained module called an object. The basic building blocks of a PowerBuilder application are the objects create by user. Each object contains the particular characteristics and behaviors (properties, events, and functions) that are appropriate to it. By taking advantage of object-oriented programming techniques such as encapsulation, inheritance, and polymorphism, user can get the most out of each object they create, making their work more reusable, extensible, and powerful.

PowerBuilder supports cross-platform development and deployment. For example, user can develop an application using PowerBuilder under Windows and deploy the very same application without changes on the Macintosh. Or vice versa. It even enable users to have a cross-platform team of developers, some using Windows and some using the Macintosh, developing the same application at the same time. They can freely share PowerBuilder objects used in the application, because the objects are the same across the different computing platforms that PowerBuilder supports.

PowerBuilder provides easy access to corporate information stored in a wide variety of databases. By using PowerBuilder, user access databases in either of the following ways:

i) By using the Powersoft ODBC interface and,

ii) By using one of the Powersoft database interfaces that provide a direct database connection

The Powersoft ODBC interface allows users to access the database by using Open Database Connectivity (ODBC), Microsoft's standard for database connectivity. When user use the ODBC interface, they define an ODBC data source that consists of the data they want to access and its associated DBMS or file manager, operating system, and, if present, network software that accesses the DBMS. The data source stores and manages the data on behalf of the application. User can access an ODBC data source that resides locally on their computer or remotely on a network server. For example, they can access a Sybase SQL Anywhere database created on a remote server by installing the SQL Anywhere ODBC driver and defining the ODBC data source.

A Powersoft database interface is a native (direct) connection to a database. PowerBuilder does not go through ODBC to access a database through a Powersoft database interface. For example, if users have the appropriate SQL Server software installed, they can access a SQL Server database through the Powersoft SQL Server interface. Each Powersoft database interface has its own interface DLL that communicates with the specified database. When user uses a Powersoft database interface, the interface DLL connects to the database through the database vendor's application programming interface (API).

CHAPTER 3:SYSTEM REQUIREMENT

3.1 PROJECT DEFINITION

PDS is defined as a client/server application where it can be accessed by logging into this program. It is not a web-based application which it is restricted to internal users in an organization. The administrator of the project will be the staff in the administration department. The users of **PDS** could be those who were getting approval by the administrator to logging to the system (Mostly the data key-in clerk).

The functionalities of the **PDS** will be explained in further details under functional requirements and non-functional requirements.

3.2 OVERVIEW OF PDS ARCHITECTURE

Figure 3-1 shows the overview of **PDS** architecture to be built after the feasibility study and shows the relationship between the services.

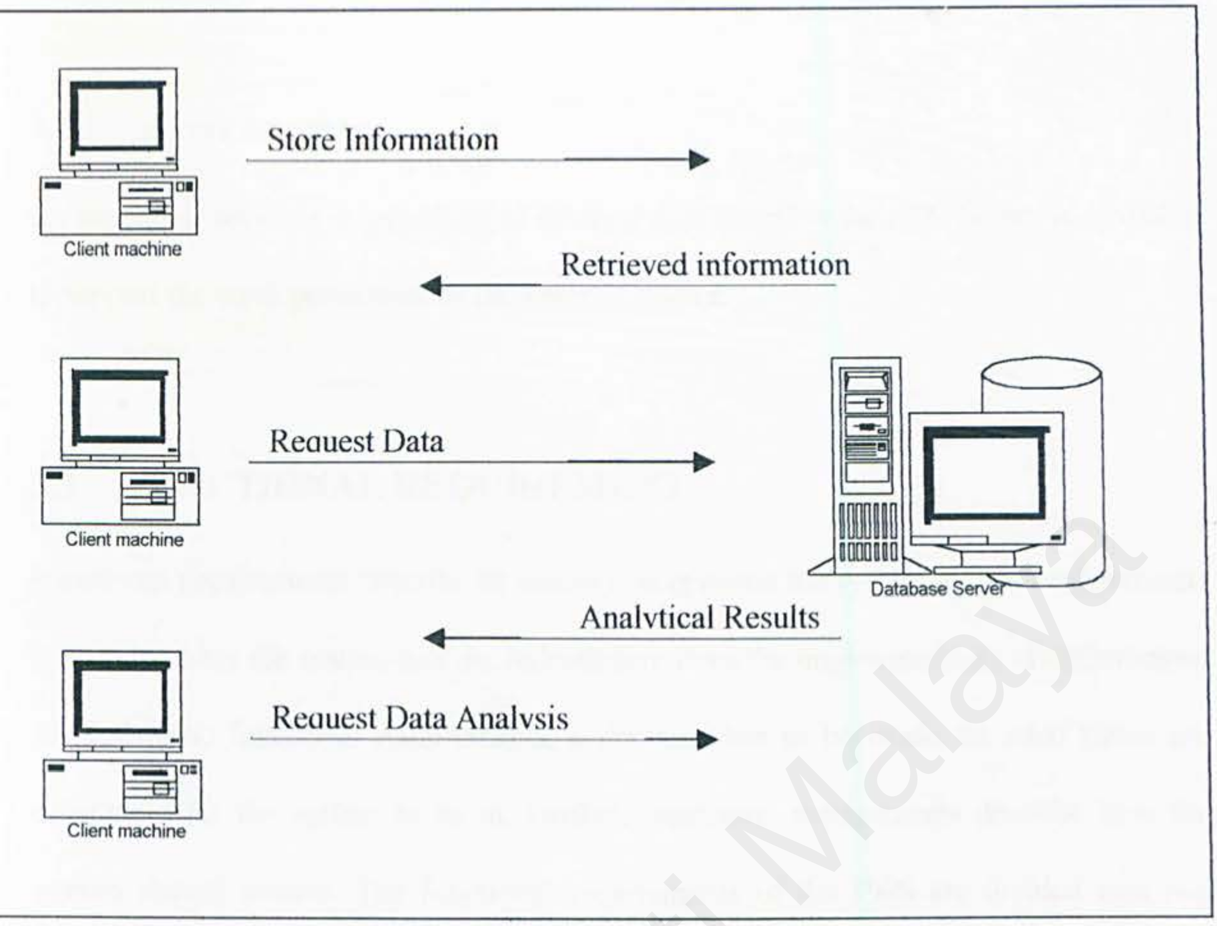


Figure 3.1 Overview Architecture of the **PDS**

PDS is designed to leverage the traditional client/server architecture and extend it Components were built into each tier to fulfill its role and then tied together to form a final solution.

3.2.1 Client Services

At the client services level, user may input all the data in the application and also require information from the server. User may perform all the task in the client side and store the result in the server.

3.2.2 Server Services

On the server services, a repository of relevant data stored in the SQL Server is available to support the work performed by the analysis engine.

3.3 FUNCTIONAL REQUIREMENT

Functional requirements describe an interaction between the system and its environment. It explains what the system will do, independent from the implementation of the solution. To determine functional requirements, a decision has to be made on what states are acceptable for the system to be in. Further, functional requirements describe how the system should behave. The functional requirements of the **PDS** are divided into two sections because there are two types of users. The first section is the user environment and the second section will be the administrator environment.

3.3.1 User Environment

i) Log In/Log out

The **PDS** provides a common login or users to access modules in the system. To ensure all the modules share the same user authentication method, it is necessary to define the function alls that can be used to encrypt the password, and to authenticate the user. The function is called whenever user access to a page. The function will check whether the user is authorized to use **PDS** application and also the right access of the modules

ii) Insertion/ update of employee's record

This module allow authentic user to key-in employee record. This will include:

- Name
- Age
- Sex
- Race
- NRIC Number
- License Number
- Nationality
- Status
- Address
- Telephone Number
- Fax Number
- Contact Person
- Dependent Name
- Relationship Between Employee and Dependent
- Dependent Age
- Dependent Sex
- Email Address
- Position
- Salary
- Account Number

- EPF Number
- SOSSO Number
- Medical Status

The user in this part just has the authorization on editing the information. They are not allowed to modify the information in the system.

iii) Insertion/update of panel doctor's record

This module allow user to key-in company approved panel doctor and hospital record and store in company database. This information include:

- Clinic Name
- Clinic Doctor
- Clinic Address
- Contact Number
- Fax Number
- Email Address
- Contact Person
- Clinic Account Number

iv) Employee's Medical Billing

In this module, user may key-in employee's medical billing from company's panel doctor. Whenever an employee or his/her dependent went to company-approved clinic, the clinic will be printout a receipt that stated information on particulars person along with the medical report and the billing. User at this stage will enter all this information in this module. This will include:

- Employee or Dependent Name.
- Medical Billing
- Types of Treatment
- Relevant Clinic/Hospital

v) Leave Requirement

This is the module where user type-in employees leave application whenever employees need a MC that approved by panel doctor. Therefore, this module will perform function that will check the company's leave timetable and require validation from the administrator.

vi) Maintenance

This module provide function change password for security purposes. User has to key in the old password correctly in order for him/her to make changes. This is a step to ensure that only the valid user is making changes.

3.3.2 Administrator Environment

i) Log In/Log out

The login/logout features exist in administrator module as the user module does. This module ensures that administrator with authentication to login to this high security section.

ii) Insertion/Remove and Modified Employee's Record

Similar with the user module, administrator has the rights to remove and delete particular employee record with reason. Besides, they may also modify the record. These functions are visible only for administrator to keep the consistently of the information.

iii) Insert/remove and Modified Panel Doctor's Record

Other than employees' records, administrator may also capable to add, remove and modify existing panel doctor information under acceptable reason.

iv) Leave Requirement

Administrator is the key decision to employee's leave application. This module allow administrator to view all the application been made along with the reason. Then, administrator will decide whether the application is approve by placing a mark in this module.

v) Employee Medical Billing

This module is also visible to administrator where administrator was given authorized to delete and modify the record.

vi) Add/Remove User

This module allows administrator to give authentication to particular employee to be able to access the **PDS.** On the other hand, administrator can also remove and delete an employee from the system. After the deletion, this employee is no longer attaching to **PDS**.

vii) Report Viewing

The system will be able to generate reports required by the administrator and company management. The administrators can search information by keywords and order by field of choice. They can inquiry the system to view or print the following reports:

- Employees Listing report
- Clinic/Hospital Listing report
- Panel Doctor Listing report
- Employee Medical Billing report
- Employees Medical Billing report
- Clinic/Hospital billing report
- Monthly employees medical expenses report
- Quarter-yearly employees medical expenses report

- Yearly employees medical expenses report

viii) Maintenance

Besides normal user, administrator may also change their password anytime to ensure their password security. User has to key in the old password correctly in order for him/her to make changes. This will ensure that only the valid user is making changes.

3.4 NON-FUNCTIONAL REQUIREMENT

3.4.1 Reliability

A system is said to have reliability if it does not produce dangerous or costly failures when it is used in a reasonable manner, a manner that a typical user expects is normal. The system would be able to identify the correct login to give access to the correct users. For example, the normal user may only see the module that authorized to them, this means that they cannot see module like add/remove users when they access the **PDS**. The system would have to ensure that all the data inside the system is secure and strictly confidential.

3.4.2 Security

The system must have sufficient security measures to ensure the program runs smoothly without corrupted. Therefore, only administrators are authorized to control users of the system and normal users would not have access to other employee's record. Some of security measures include:

i) Only the administrator will be able to authorize new user.

ii) Each user will need a login userID and password to access the **PDS**.

iii) After administrator have authorized the new user. The user was asked to change his/her password on the first time he/she access to the system.

iv) Password in the database will be encrypted.

3.4.3 Usability

The system must be easy to use and user friendly to support users that are not very computer literate. The system will apply the use of suitable and meaningful icons to help users to use the system with more confidence. Besides, confirmation message for any non-trivial process such as updating or deleting a record should be displayed. Confirmation messages should also be displayed after adding, updating and deleting a record.

The system provides an effective error handling and validation procedure. It displays an error message or warning if an error occurs, such as invalid password/user ID, invalid data, etc.

3.4.4 Response Time and availability

The response time to retrieve the information such as employee's record can be considered within a reasonable interval time. It means that all desirable information should be available to users at any point of time. The requirement for up-to-date or timely information is also important.

3.4.5 Manageability

The system is easy to manage for administrator because their jobs are made easy by just click on the icon to perform task and need no filling the blank manually. For example, if an administrator wants to delete an user, he/she just click on the user name list in multi-listbox and just press delete key on the keyboard or click on the delete icon.

3.5 PROTOTYPING APPROACH

In **PDS**, prototyping approach is used. Prototyping is an information-gathering technique to modify system with minimum expense and disruption because changes are made from time to time to improve the system. Prototyping stress the importance of the potential users and it seeks user suggestion, innovation, reaction, and revision plan in order to make improvements to the prototype even in the development stage o the project.

Prototyping is consider because of the following advantages:

i) Scrapping – Users and analysts may find undesirable module appear in the system and this module may be scrapped off. This can saves time on developing the unwanted module right from the start rather than waiting until the end of the project.

ii) Dynamic and early changes – A successful system requires frequent user feedback to help modify the system and making it more responsive to the user's actual needs and expectations from the system.

3.6 TOOLS CHOSEN

Based on the literature review in Chapter 2. The **PDS** will be implemented using the following tools:

3.6.1 Microsoft Windows NT Server 4.0

Windows NT Server is the most complete platform available for building and hosting Web-based applications and also distributed computing system. Besides, it also has the easiest server operating system available. User will experience far less downtime thanks to its reliability and easy management. Windows NT Server 4.0 was designed to help developers build and deploy business applications faster than ever before.

3.6.2 Microsoft Visual Basic 6.0

Visual Basic 6.0 is the most productive tool for creating high-performance enterprise and standalone applications as well as distributed computing system. Integrated Visual Database Tools and a RAD environment promote productivity while native code compilation provides fast applications. Microsoft Visual Basic 6.0 includes some of the features as shown below:

- ADO (ActiveX Data Objects)

  Visual Basic 6.0 introduces ADO as the powerful new standard for data access. Included OLE DB drivers include SQL Server™ 6.5+, Oracle 7.3.3+, Microsoft Access, ODBC, and SNA Server.

- Native Code Compiler

  Create applications, and both client and server-side components that are optimized for throughput by the world-class Visual C++ 6.0 optimized native-code compiler.

- Visual Basic WebClass Designer

  Create server-side applications and components that are easily accessible from any Web browser on any platform.

- Integrated enterprise visual database tools

  Visual Basic 6.0 provides a complete set of tools for integrating databases with any application. Database features include design tools for creating and modifying SQL Server 6.5, Oracle 7.3.3 or above, and also AS/400 databases.

- Data environment designer

  Visually create reusable record set command objects with drag-and-drop functionality.

- Automatic data binding

  Virtually no code is needed to bind controls to data sources. Setting just two properties in the Property window connects the control to any data source.

3.6.3 SQL Server 6.5

Microsoft SQL Server 6.5 is a scalable, high performance database management system designed for Windows NT-based systems. To meet the requirements of distributed client-server computing, SQL Server 6.5 is tightly integrated with the Microsoft BackOffice family of servers to enable organizations to improve decision-making and streamline business processes. Microsoft SQL Server 6.5 is actually the best Database for Window NT Server. It is the solution to complex business problems on the Windows NT platform.

Despite of that, Microsoft SQL Server 6.5 lowers the cost and complexity of distributed computing by reduced complexity for users, administrators and developers

3.7 RUN TIME REQUIREMENT

The run time environment of the system consists of hardware and software configurations.

3.7.1 Server Hardware Requirements

Server computer requirements are:

i) A server with at least Pentium II 500MHz processor with at least 64 Mb RAM.

ii) Network Interface Card (NIC) and network connection with recommended bandwidth at 10Mbps or more

iii) Others standard computer peripherals

3.7.2 Server Software Requirements

To host and run the system, the server computer needs to have various supporting software installed.

i) Windows NT Server 4.0 as Operating System

ii) SQL Server 6.5 as database

iii) Microsoft Visual Basic 6.0

iv) ODBC driver

3.7.3 Client Hardware Requirements

For the client side, hardware requirements are quite minimal. The recommended configurations are:

i) At least 32 Megabytes of RAM

ii) Network Interface Card (NIC) and network connection with recommended bandwidth at 10Mbps or more

3.7.4 Client Software Requirements

The client software requirements were recommended to be:

i) Installed with Windows 95/98 and,

ii) Microsoft Visual Basic 6.0

CHAPTER 4: SYSTEM DESIGN

4.1 PROJECT METHODOLOGY

The Waterfall-Review Model

PDS is developed based on the "Waterfall-Review" Model. "Waterfall-Review" model is an enhanced model based on standard "Waterfall" model where every process end with review before proceeds to the next phase. The basic "Waterfall" model is shown below:

φ Project Definition

Project definition process describes how things are done in tradition, the weaknesses and drawbacks of the traditional approach. Then the project will be proposed in non-technical manner.

φ Literature review

Literature review involves the steps of finding, summarization, analysis and synthesis o case study, related literature work of other researchers and technology to be used.

φ System Analysis

The system's services constraints and goals are established by consultation with system users. The functional requirement and non-functional requirement will be list out in the manner that is understandable by both users and development staff.

φ System Design

System design process established the overall system architecture. In this phase, project requirements were partitions to either hardware or software systems. The software design will be the representation of the software system functions in a form that may be transformed into executable programs.

φ System Development and Unit Testing

This stage will be the phase where the software design has been realized as set of program units. Developer shall start create the application module by module and checking the codes. Unit testing will then be done to verify that each unit meets its specification.

φ System Testing and Integration

The modules that developed from the early stage will be integrated together individually in this stage. Besides, overall testing of the entire system will also been conducted during this phase to ensure that all the functions developed are working after the entire module have been combined.

φ Maintenance

This is the phase where vendors/developers correcting the errors that were not discovered in earlier stages of application life cycle. Thus, enhance the system's services and improving the implementation of system units.

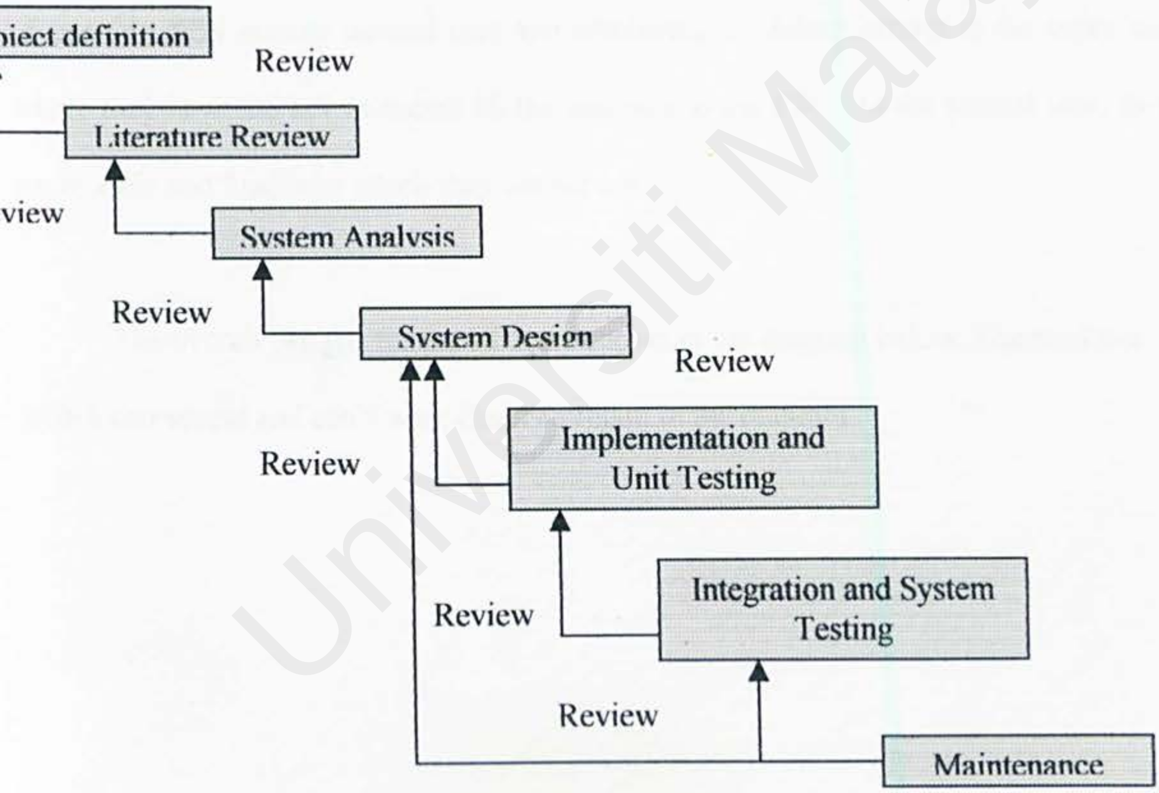


Figure 4.1 "Waterfall-Review' Model

The purpose of choosing the "Waterfall-Review " model is that the model supports iterative design. Very phases of this model end with a review process, this helps the developer of the project control the quality of the project.

4.2 APPLICATION DESIGN

PDS has 8 main modules as what had been described in the functional requirement but not all the modules in the application are visible to the user. Basically, there are 2 levels of users in **PDS** namely normal user and administrator. Administrator is the super user where they have the key to access all the modules in the **PDS**. As for normal user, there are module and functions which they cannot apply.

The overall designs for the PDS are shown in the diagram below. The modules which can access and can't were clearly explain in the diagram.

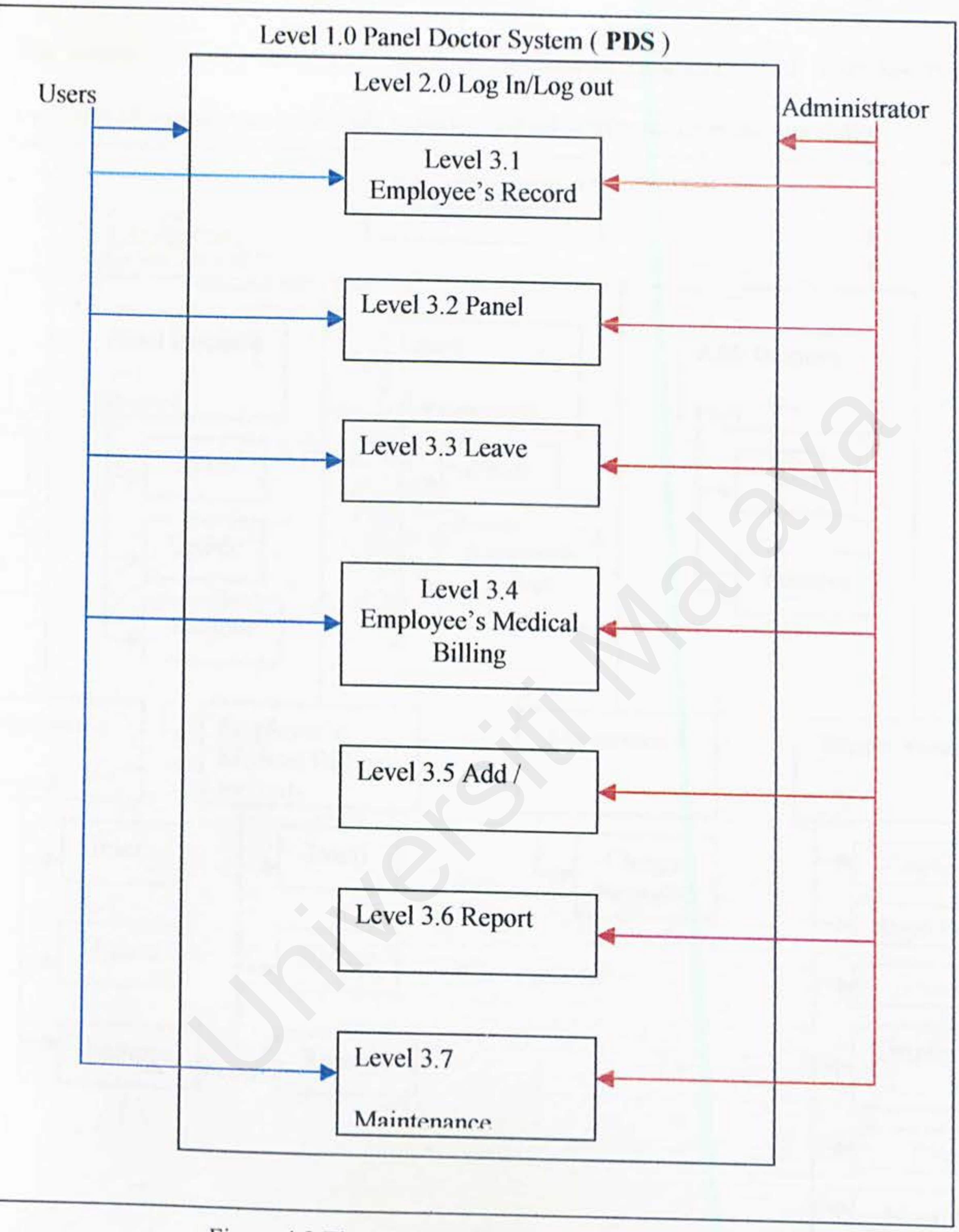


Figure 4.2 The **PDS** Application Design Overview

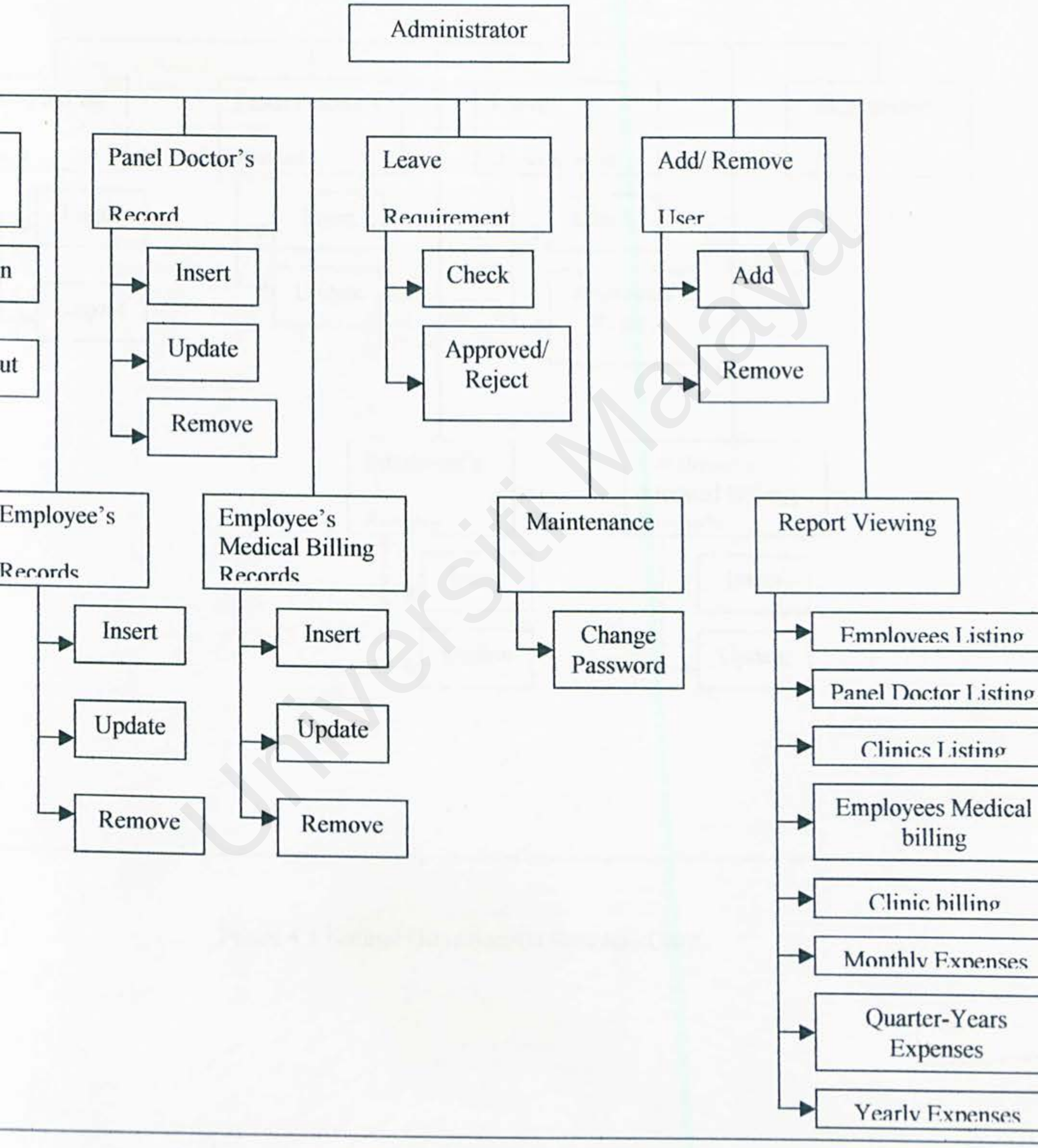
Legend:

→ Administrator accessible modules

→ Normal User accessible modules

4.3 SYSTEM STRUCTURE CHART

The system structure developed based on the system functionality. It describes the overview of normal user accessible modules and administrator level user modules.



4.3 SYSTEM STRUCTURE CHART

The system structure developed based on the system functionality. It describes the overview of normal user accessible modules and administrator level user modules.

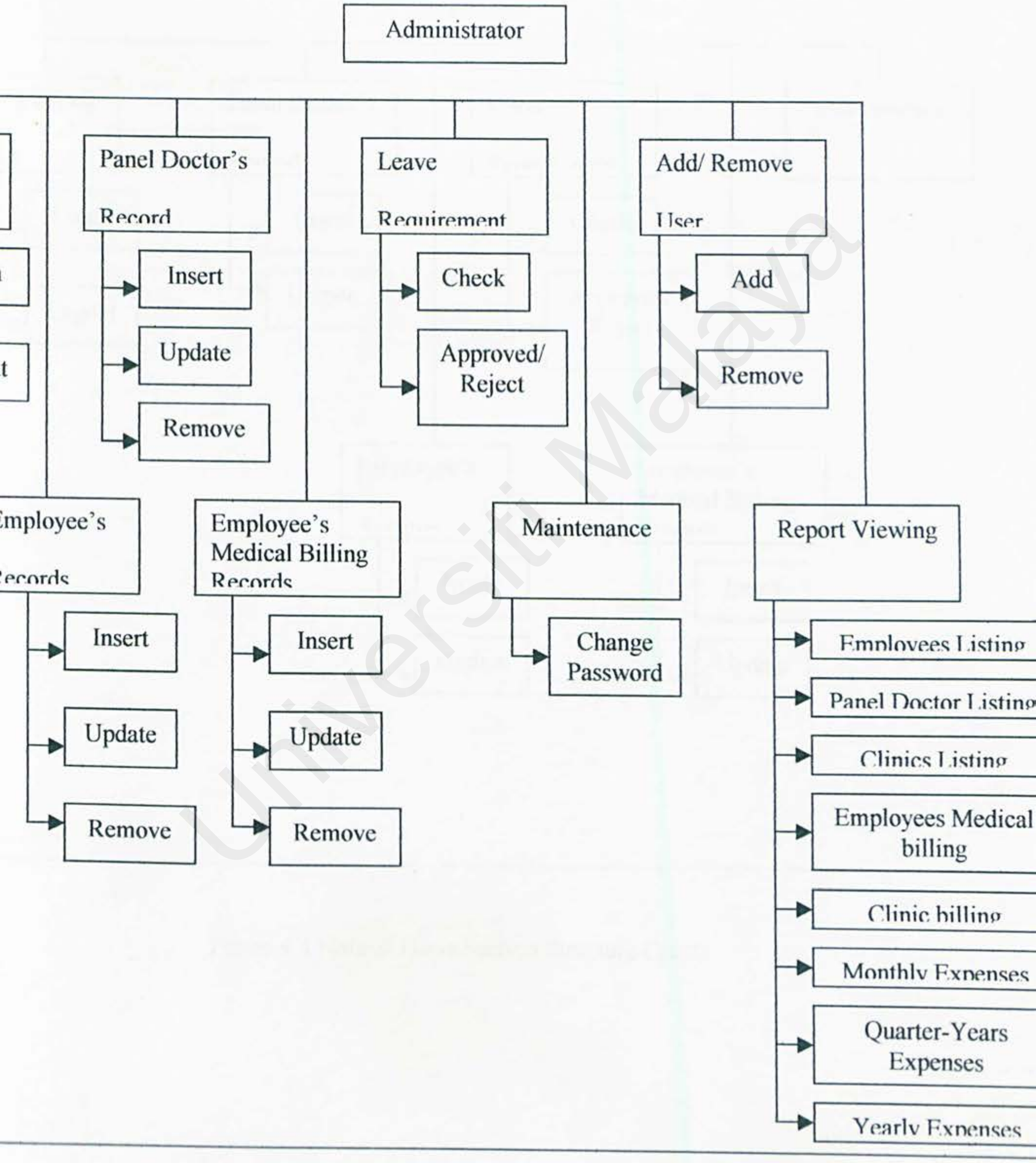


Figure 4.3 Administrator Section Structure Chart

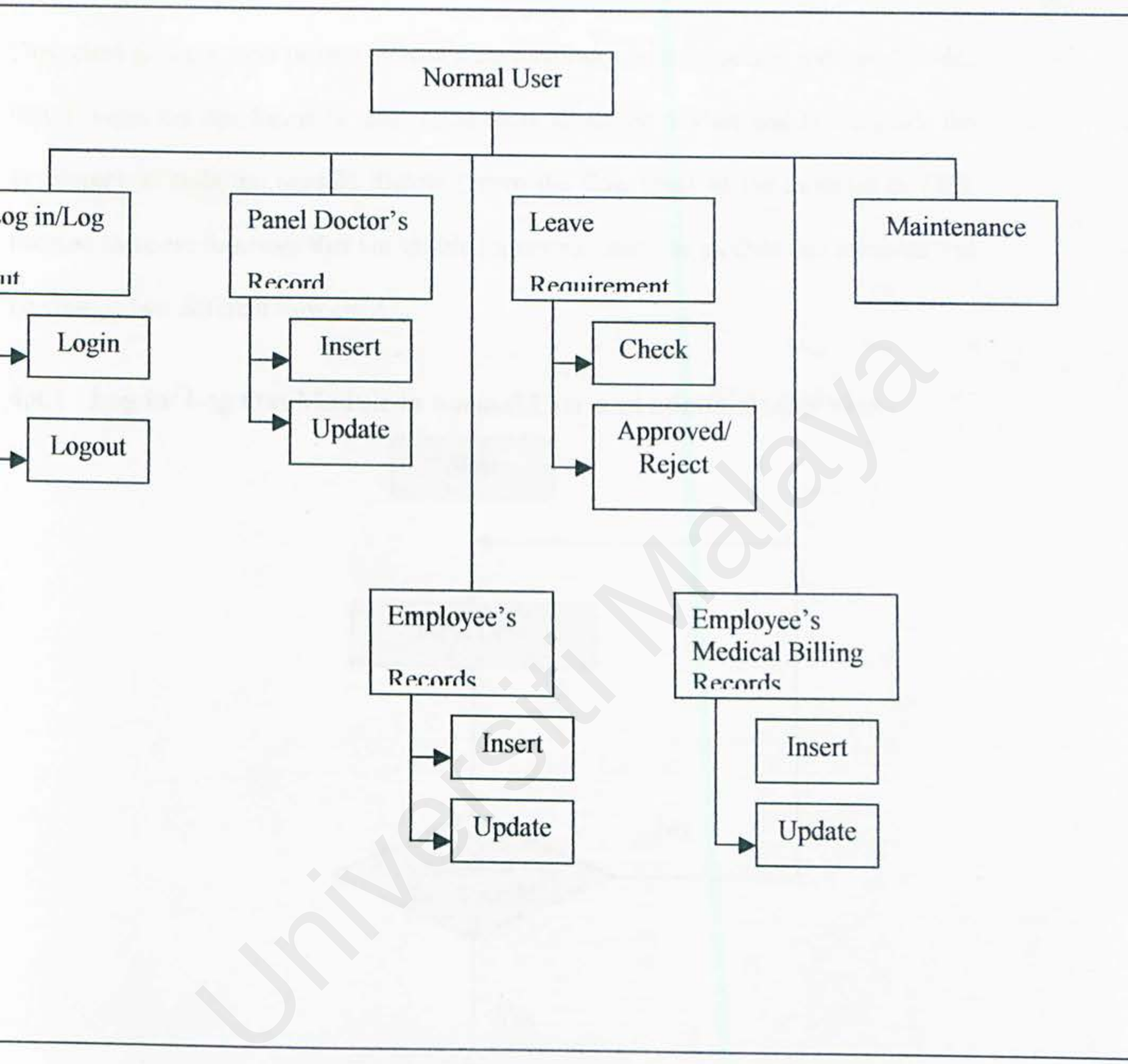


Figure 4.4 Normal Users Section Structure Chart

4.4 MODULES FLOW CHART

Flow chart giving a clear picture on how's the function runs in a specific module. Besides that, it helps the developers to understand more about the module and hence guide the developers to code the module. Below shown the flow chart of the modules in **PDS**, because there are functions that not visible for normal user, the module that involved will be view as two different flow chart:

4.4.1 Log in/ Log Out Module in normal User and administrator view

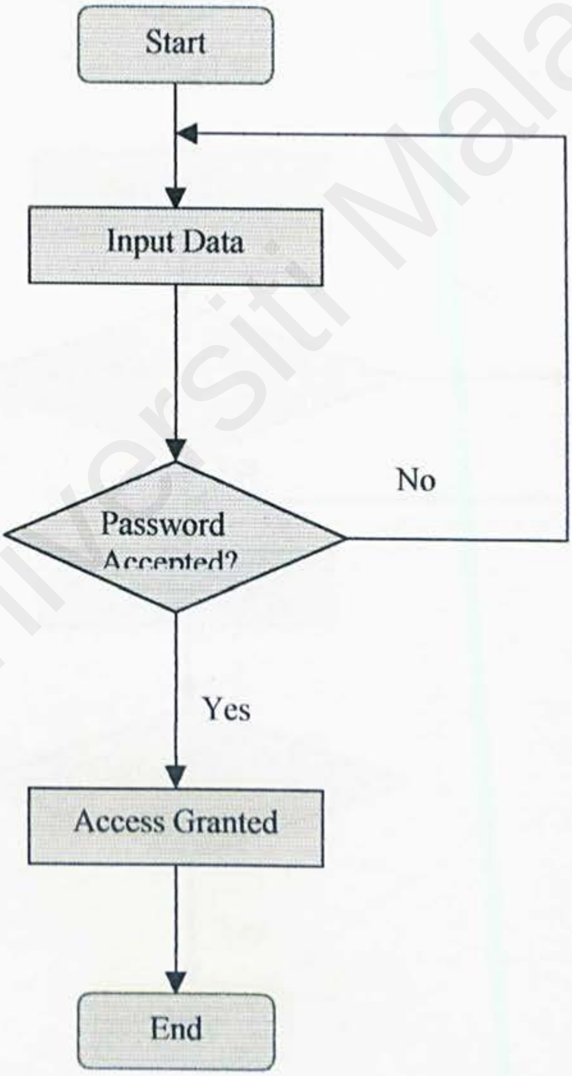


Figure 4.5 Login/Logout Module Flow Chart

4.4.2 Employee Records Entry Module in normal user view

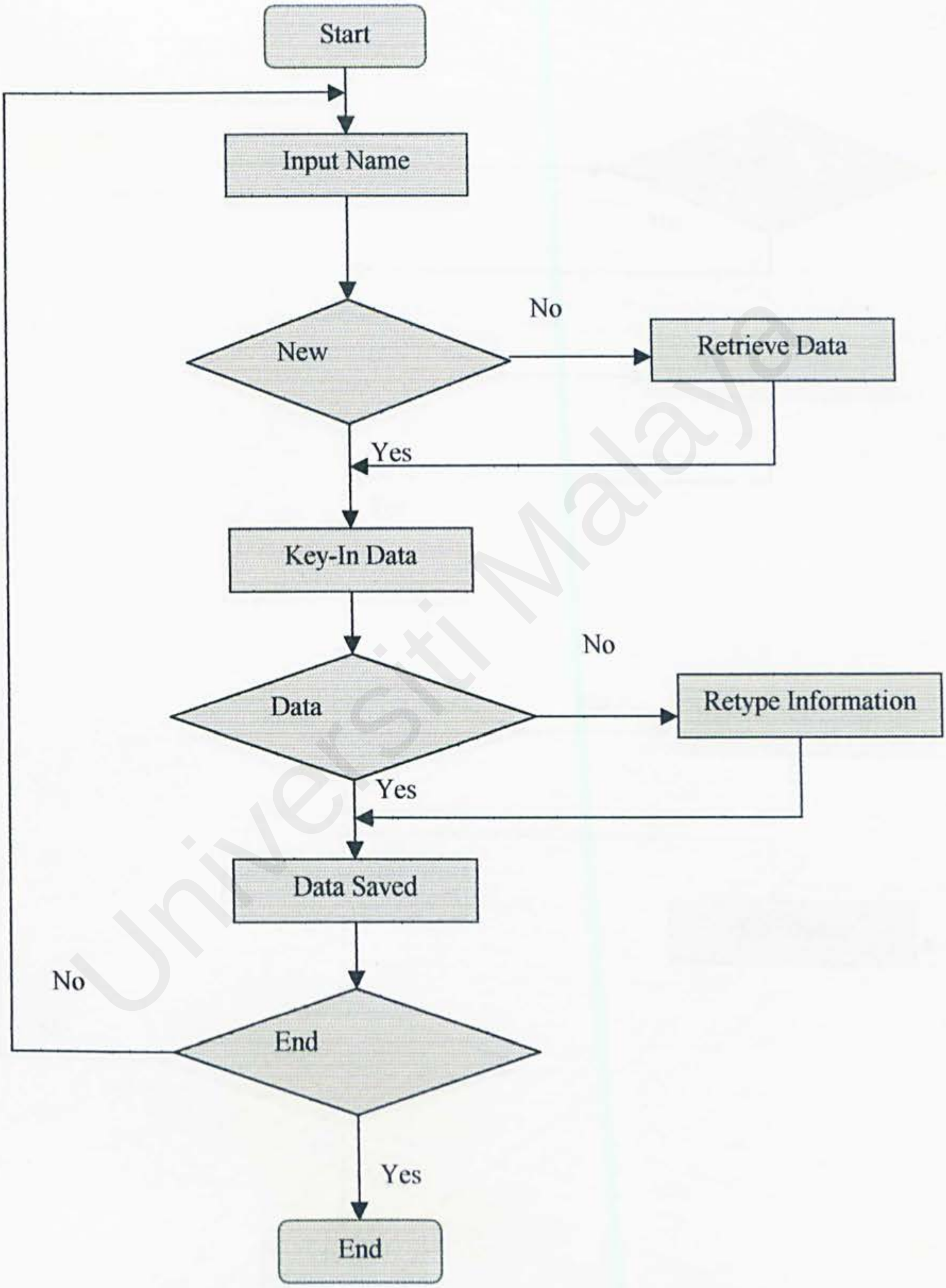


Figure 4.6 Employee Records Entry Module Flow Chart (Normal User)

4.4.3 Employee Records Entry Module in Administrator view

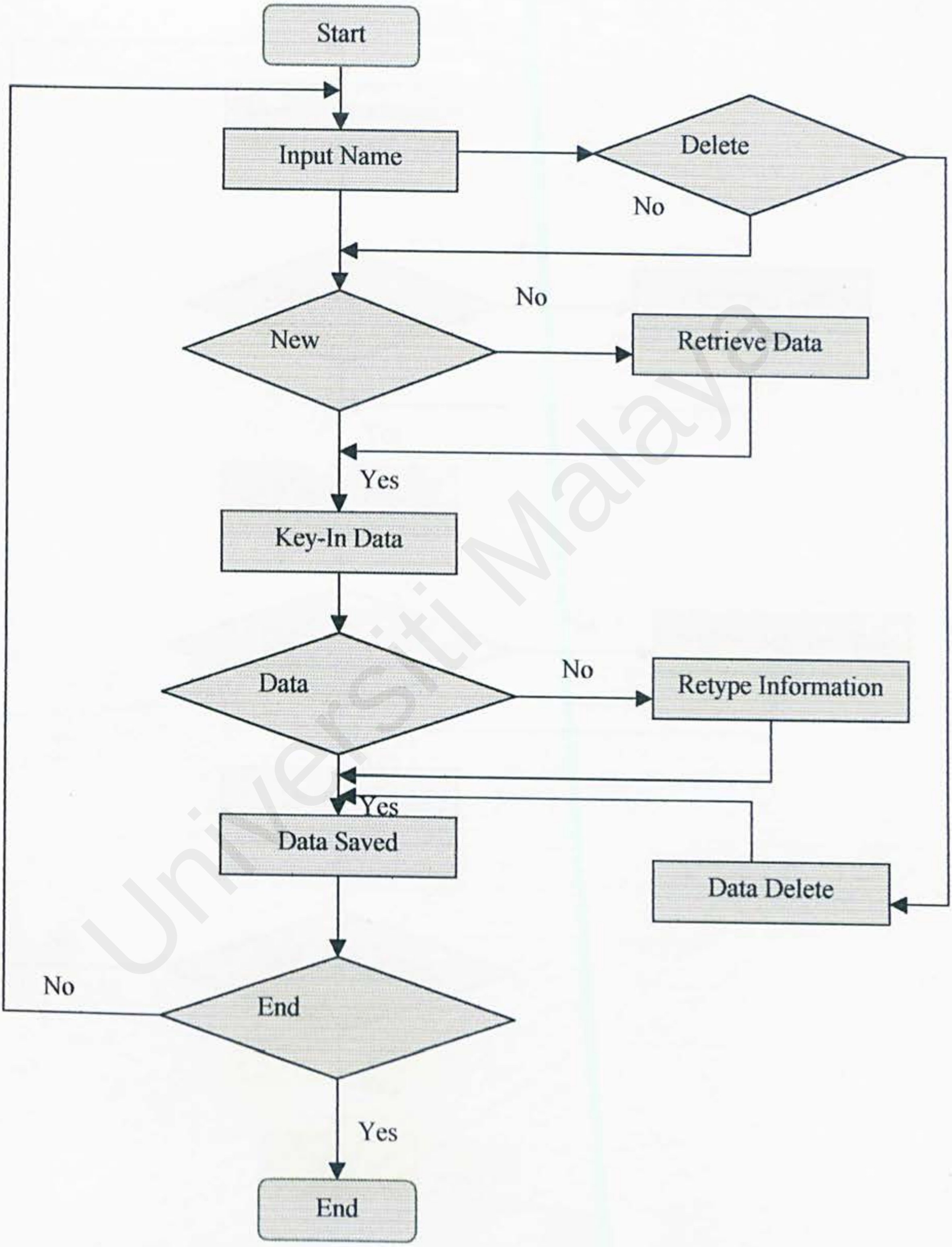


Figure 4.7 Employee Records Entry Module Flow Chart (Administrator)

4.4.4 Panel Doctor Records Entry Module in normal user view

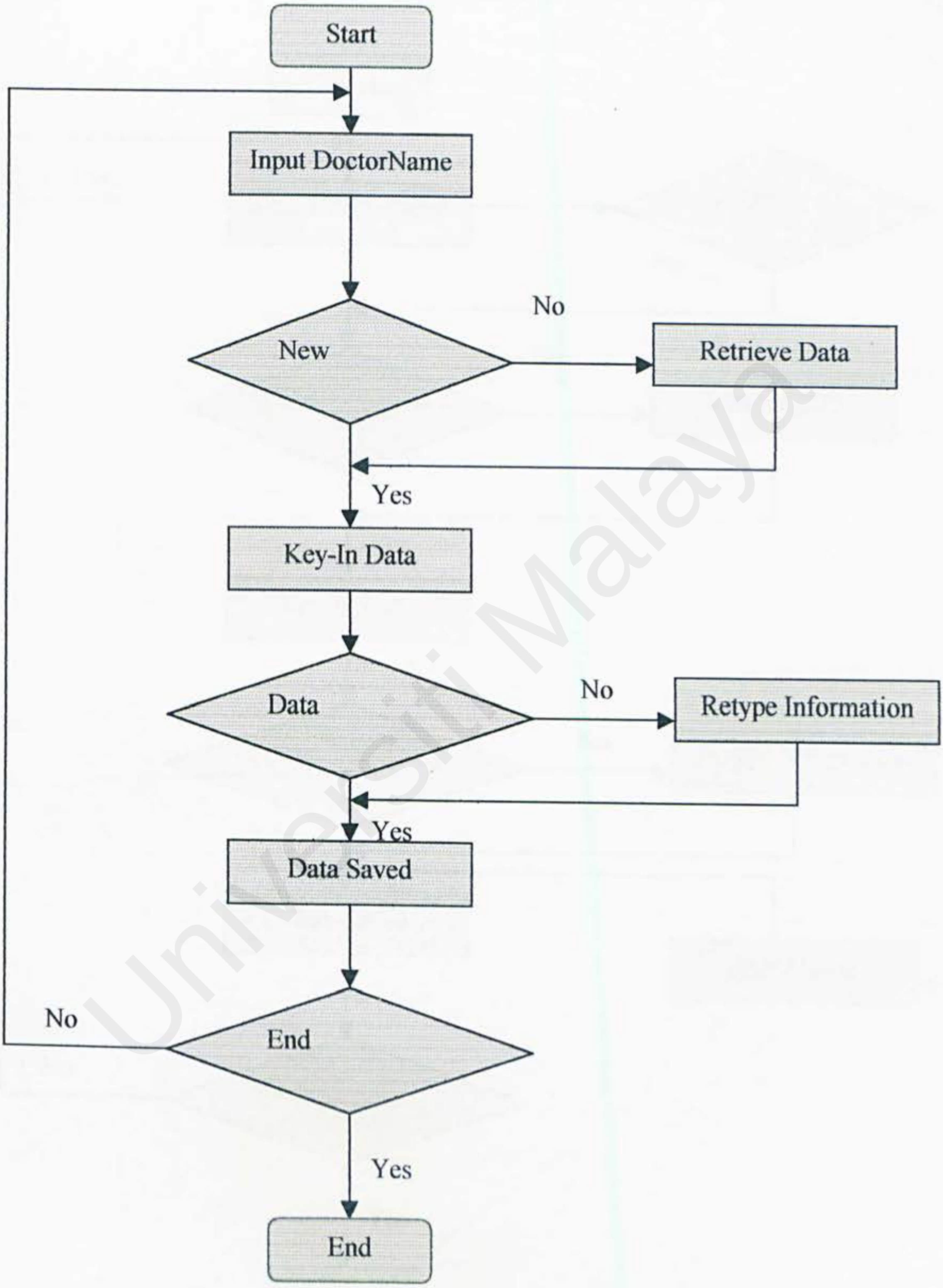


Figure 4.8 Panel Doctor Records Entry Module Flow Chart (Normal User)

4.4.5 Panel Doctor Records Entry Module in Administrator view

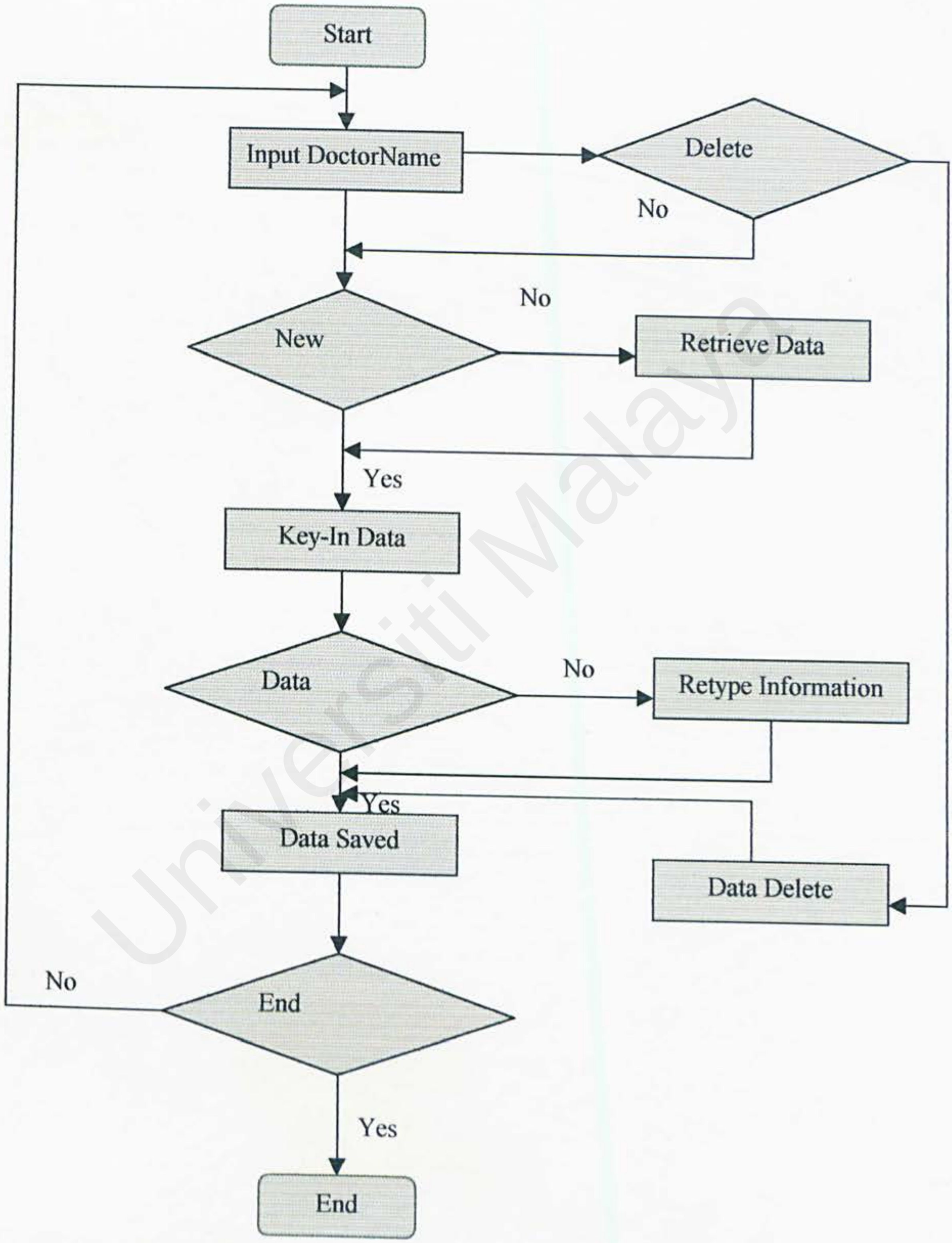


Figure 4.9 Panel Doctor Records Entry Module Flow Chart (Administrator)

4.4.6 Medical Billing Records Entry Module in normal user view

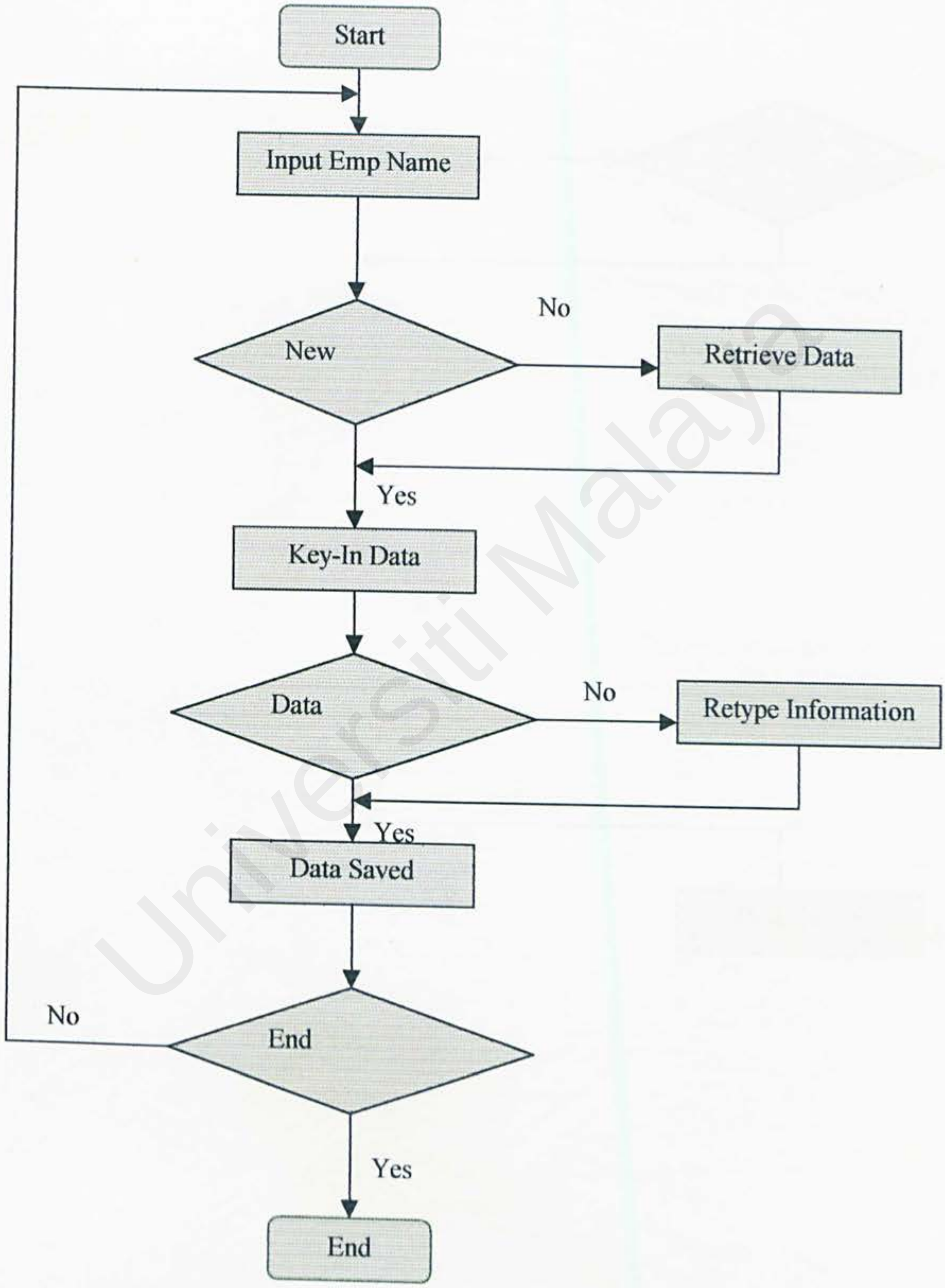


Figure 4.10 Medical Billing Records Entry Module Flow Chart (Normal User)

4.4.7 Medical Billing Records Entry Module in Administrator view

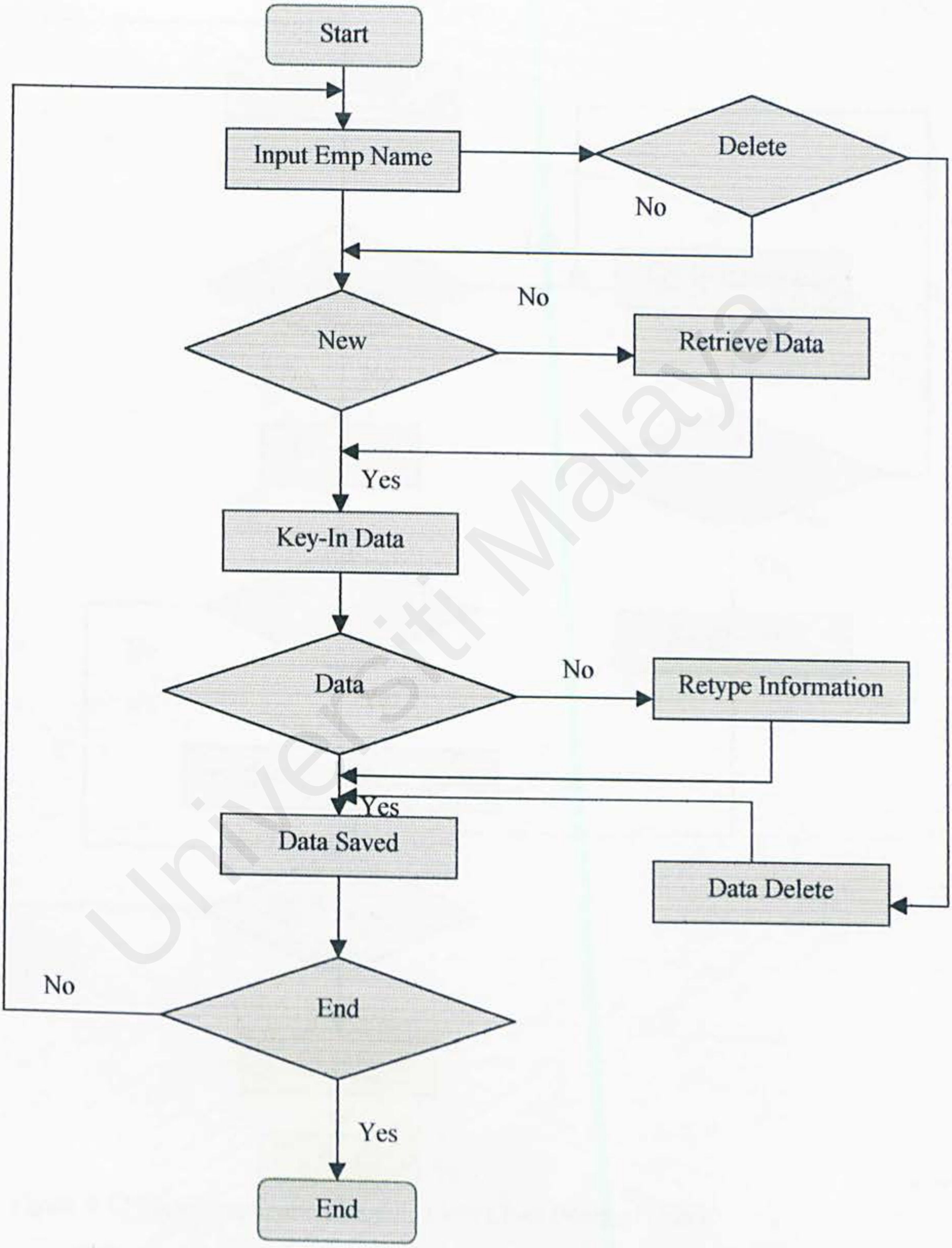


Figure 4.11 Medical Billing Records Entry Module Flow Chart (Administrator)

4.4.8 Leave Application Module in normal user view

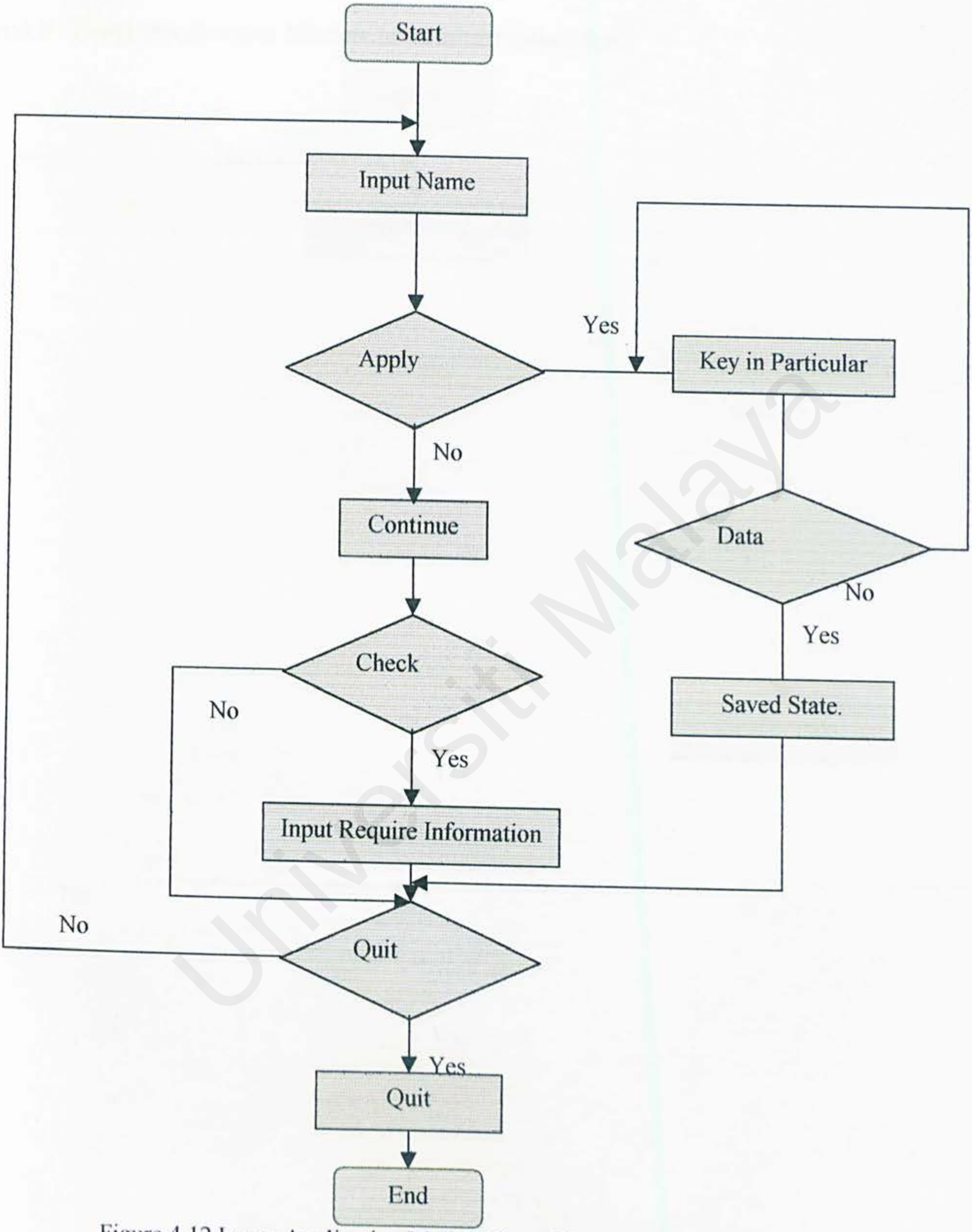


Figure 4.12 Leave Application Module Flow Chart (Normal User)

4.4.9 Leave Application Module in administrator view

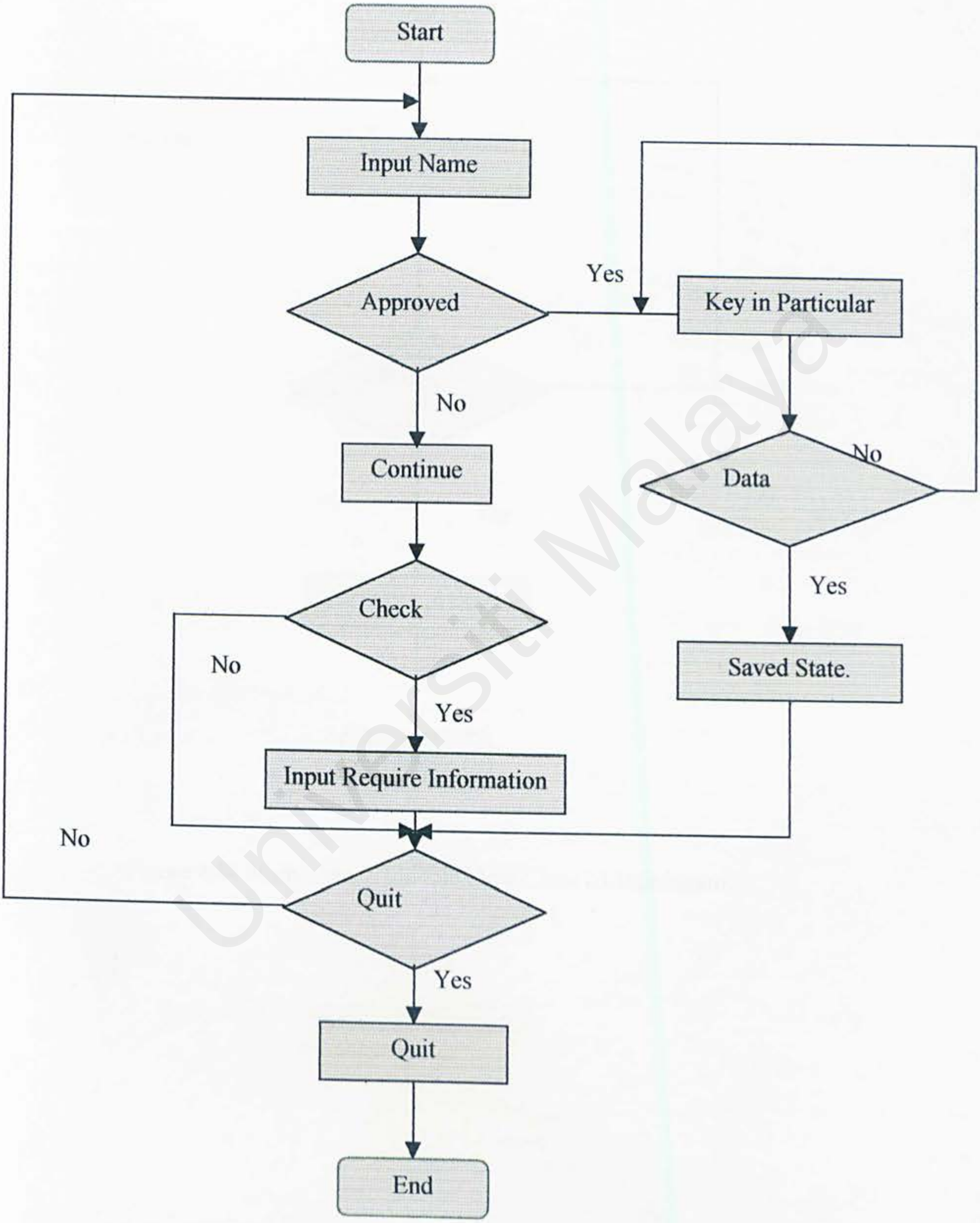


Figure 4.13 Leave Application Module Flow Chart (Administrator)

4.4.10 Maintenance Module in administrator view

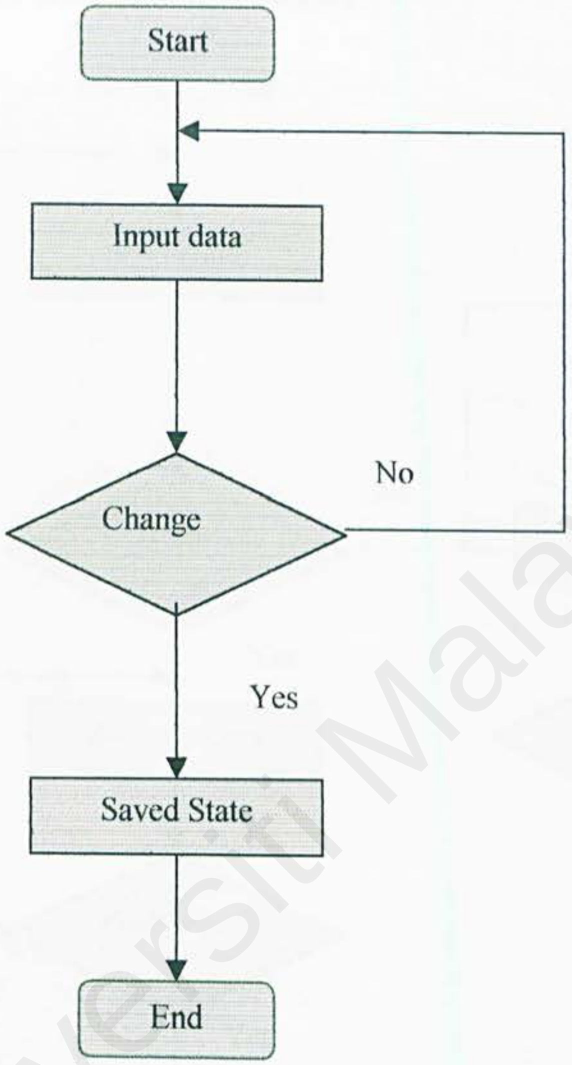


Figure 4.14 Maintenance Module Flow Chart (Administrator)

4.4.11 Add/remove User Module in administrator view

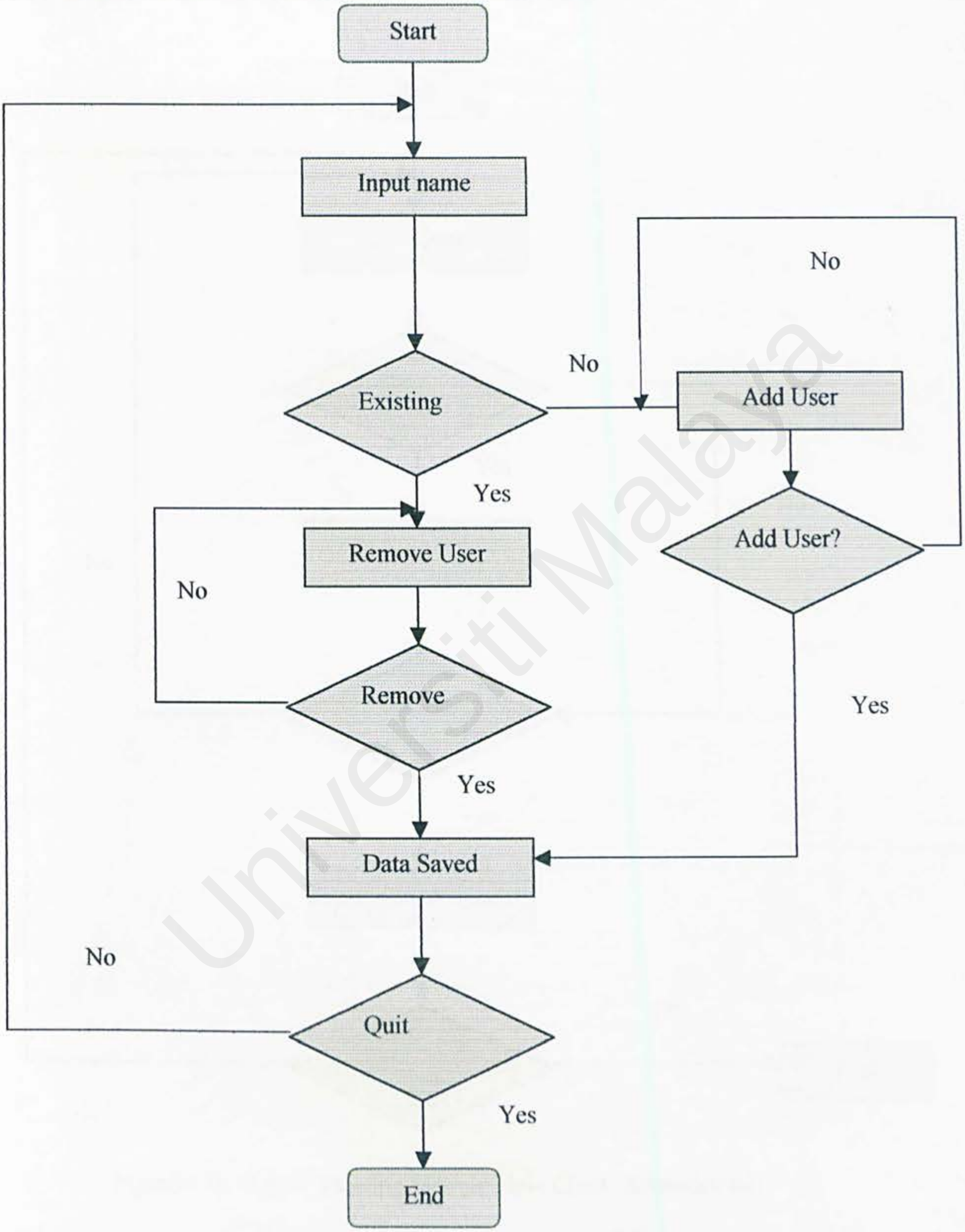


Figure 4.15 Maintenance Module Flow Chart (Administrator)

4.4.12 Report Viewing Module in administrator view

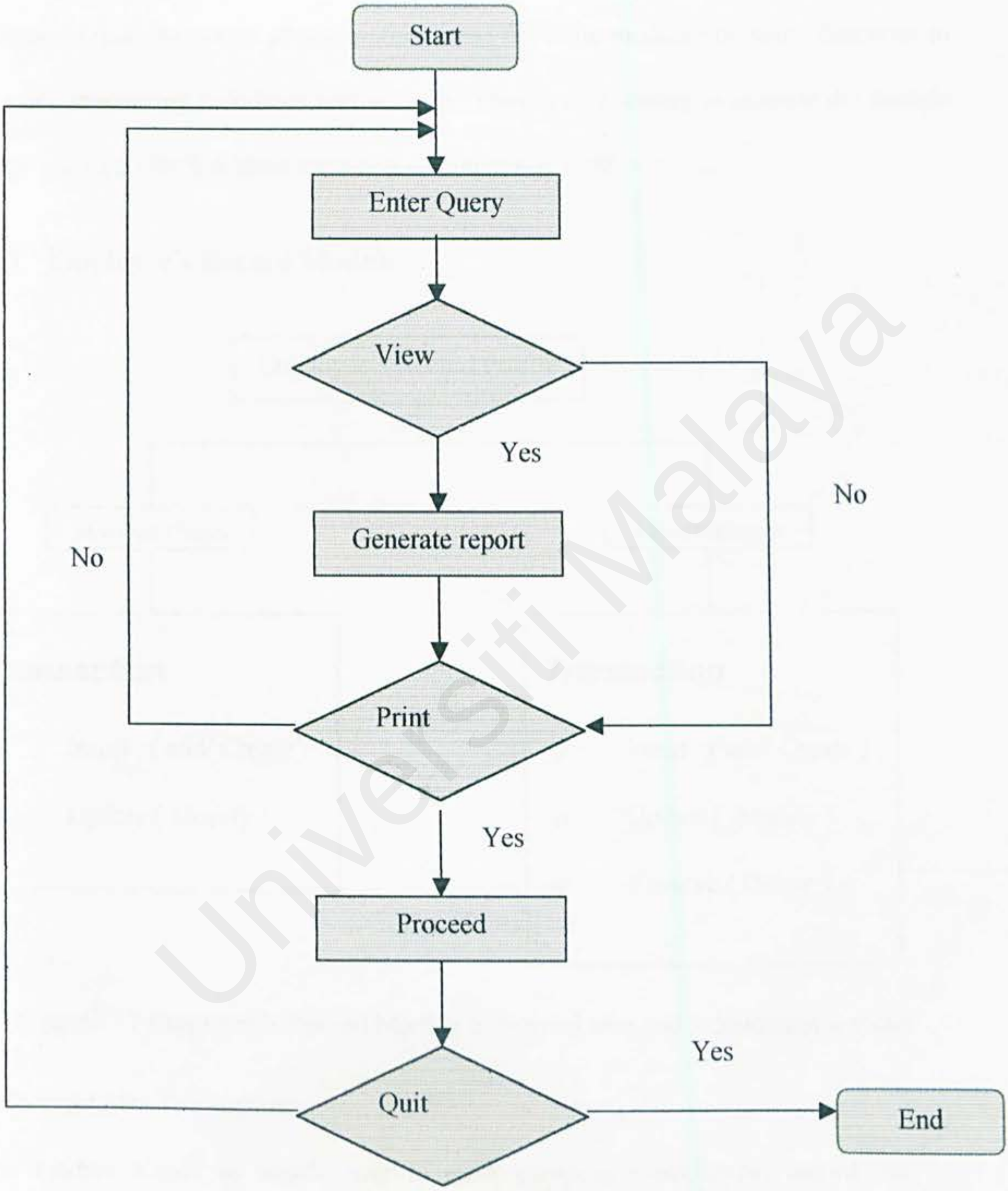


Figure 4.16 Report Viewing Module Flow Chart (Administrator)

4.5 MODULES DESCRIPTION

Because normal users and administrator shared the same module and some functions in the same module are hide from normal users. Thus, it is necessary to separate the module in two sections which is users view and administrator view.

4.5.1 Employee's Record Module

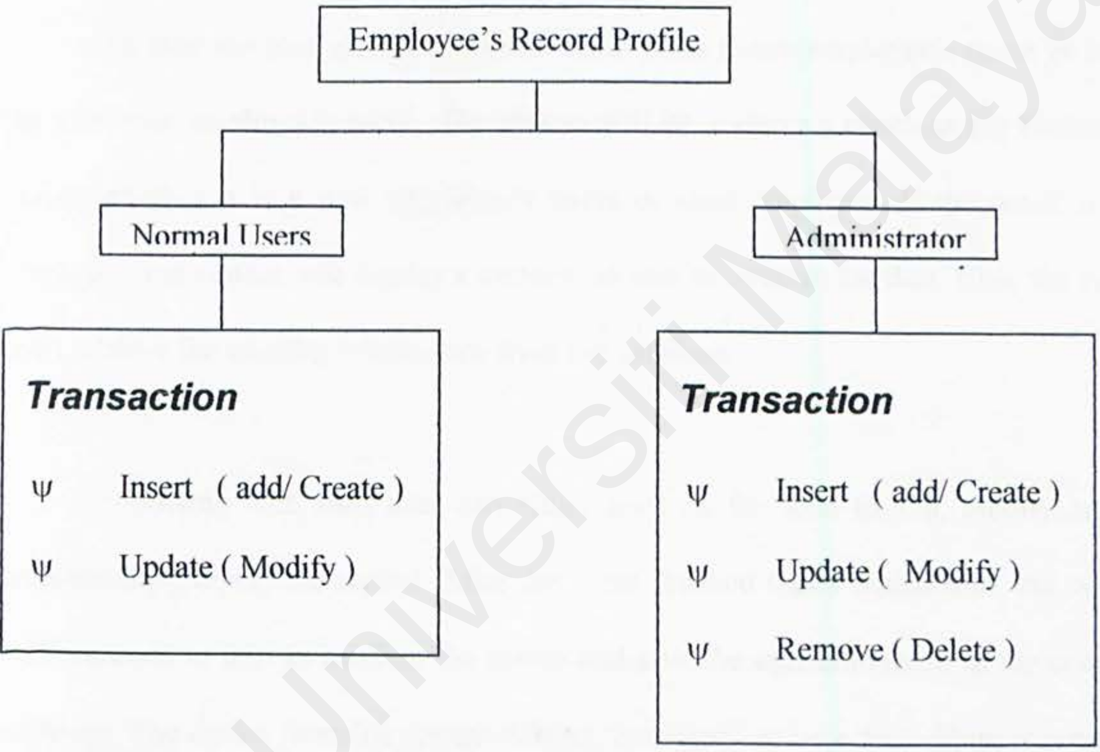


Figure 4.17 Employee's Record Module in Normal user and Administrator views

a) Normal Users Transaction

This module stands to enable user to enter company's employees record into the database. Therefore, the transaction which assign for normal user will be insert employee's information, and update employee's information. In this module, normal users are not allowed to delete the employee's record. The delete function is hide from normal user view.

b) Administrator Transaction

Administrator inherits all the transaction that can be perform by normal user and had been enable the delete record function. The authorized administrators are visible to the delete record feature because they are reliable employees in the company.

To start the transaction, users can either enter a new employee's name or key in the existence employee's name. The system will be perform a check in the backend to define whether it is a new employee's name or exist employee. If the result is new employee, the system will display a message to user to confirm the data. Else, the system shall retrieve the existing information from the database.

Following with that, user can either click on the icon to add, modify and for administrator, delete the record. After the users finished theirs transaction, the systems will response to user to confirm the action and save the updated record in the company database. The option from the system will be "accepted" or 'rejected'. Thus, it is not late for user to change their setting if they find there is an error performs by them.

4.5.2 Panel Doctor's Record Module

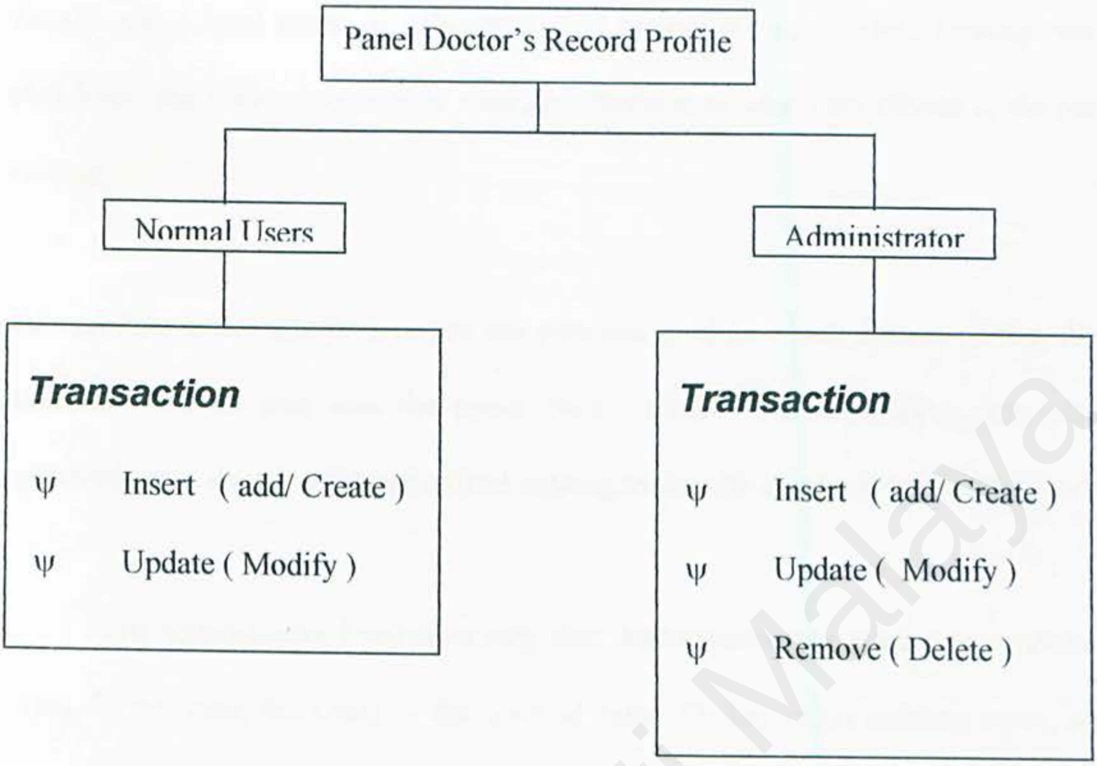


Figure 4.18 Panel Doctor's Record Module in Normal user and Administrator views

a) Normal User Transaction

The module enable users to key in company approved panel doctor information and edit the existing records. In this case, normal users are not allowed to delete the records in the company database.

b) Administrator Transaction

Administrator level users are allowed to add in new records, update existing data, and also delete the existing records in company database as what they can do in the previous module.

This module is actually behave like the previous module which discuss earlier, the only different between that was the panel doctor record module involve only company approved panel doctor information and nothing to do with employee's record.

The administrator level users may start the transaction by enter a new panel doctor name. In this case, the clinic or the hospital name. Or key in the existing name, and the system shall retrieve the existing information from the database. This module can be discuss by using the scenario below:

Scenario 1:

A new user was asked to key in new panel doctor information in the company database. The user received the hardcopy and access to the PDS. But he/she gets confusing when they came to the scene where it require panel doctor name. He/she looks at the hardcopy and found many panel doctor's name in the paper. Suddenly he/she was lost and don't know which name should be enter. She/he wondering to key in either one or creates many record?

Scenario above can be happen to anyone who's using the **PDS** for the first time. And as the answer to that question is:

i. For the first entry, key in the clinic or particular hospital name. If it is a new name to the system, it will pop up a message to confirm the date. Else, the system will perform check at the company database and retrieve related data.

ii. Then, in the specific area in the application, there will be a place for the user to key in panel doctor's name and their particular. And the user may key in as many panel doctor name as the clinic have.

4.5.3 Leave Application Module

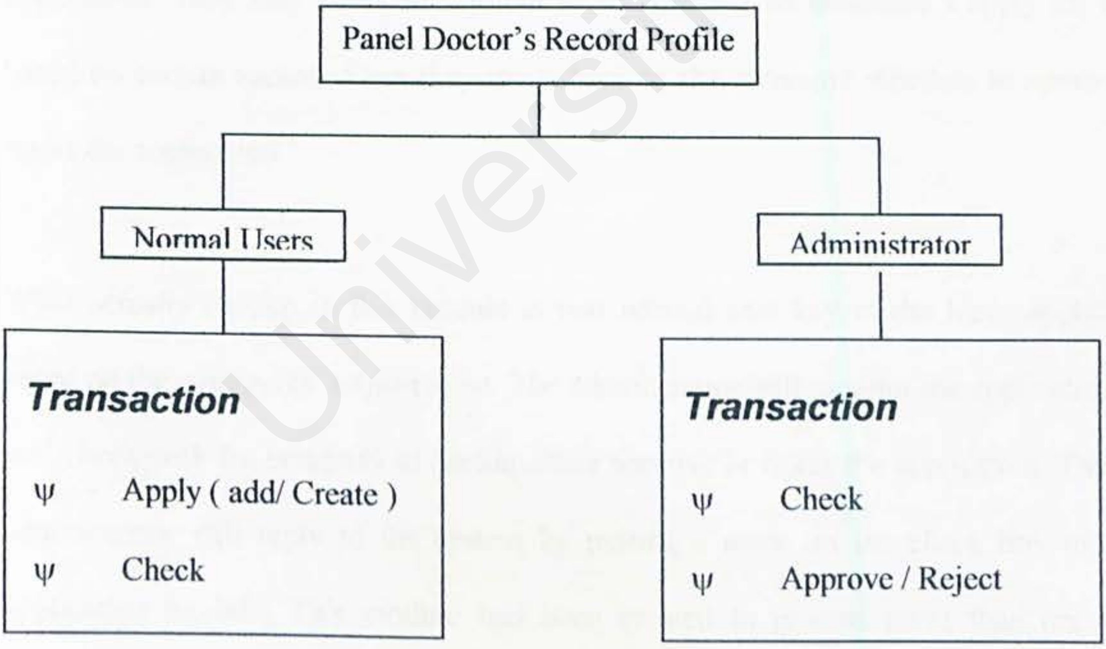


Figure 4.19 Leave Application Module in Normal user and Administrator views

4.5.4 Employee's Medical Billing Module

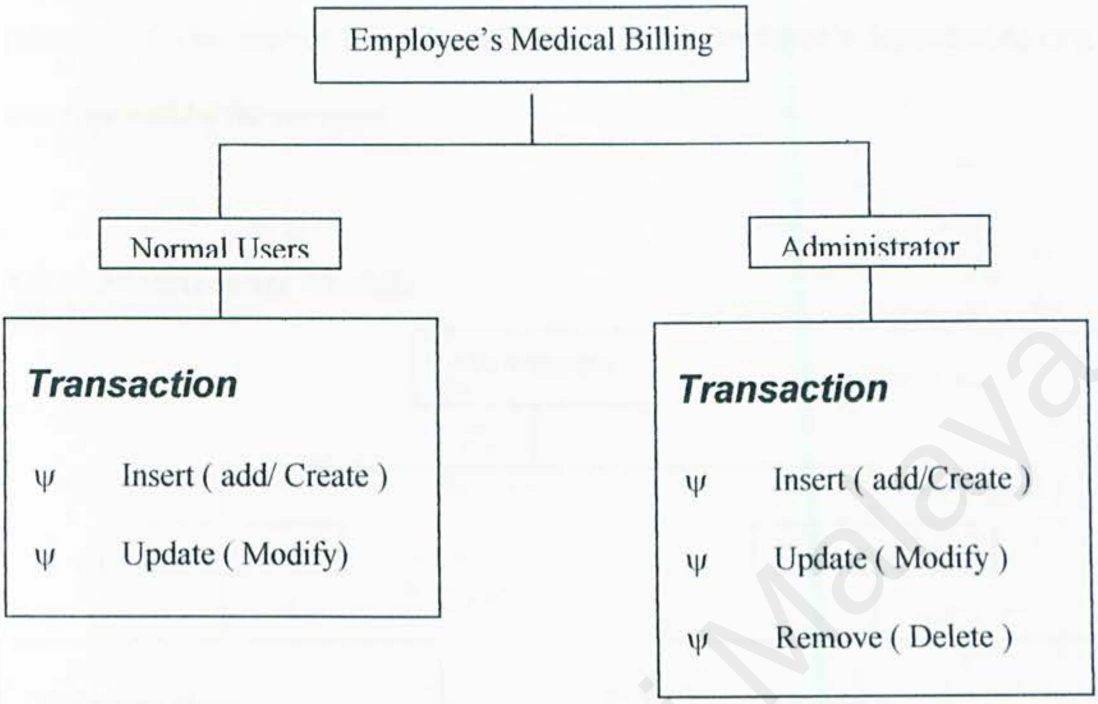


Figure 4.20 Employee's Medical Billing Module in Normal user and Administrator views

a) Normal User Transaction

This module require information on employee's medical billing from company's panel doctor. User input the information (in this case, employee's medical billing) in the system, and update the information whenever changes been made.

b) Administrator Transaction

As what the previous modules do, administrator level users are authorized to delete the records besides adding and modify it.

This module keep track of employee's medical billing records from time to time. It will keep the information on when the employees when for health check, the billing each time employees goes to the panel doctor, employees medical status, and others related information. This module even prepared a column for employee's dependent as long as it was approved by the company.

4.5.5 Maintenance Module

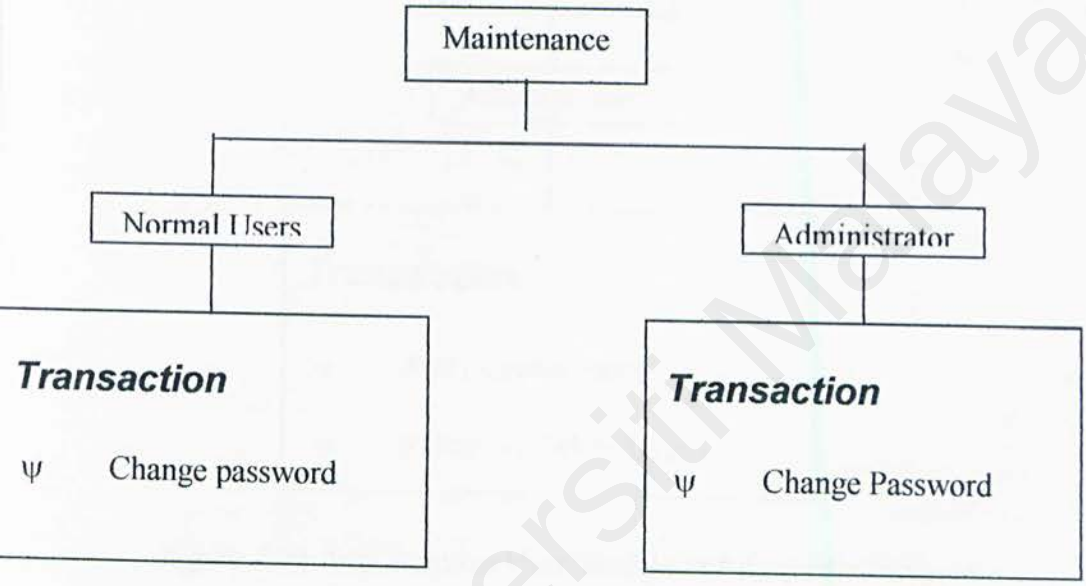


Figure 4.21 Maintenance Module in Normal user and Administrator views

.) Normal users and Administrator view

'his is a module which provide security function where users may change their old assword whenever they felt that their password is no more secure. This is the only nodule where normal user and administrator share the same view and function.

To change their password, the users will be asked to enter their user id along with the old password. The process is to verify the correct user to change their password. After the user key in the new password, a message will be pop up to confirm the changes. This enables the user to undo the changes if they had performed error accidentally.

4.5.6 Add/Remove User Module

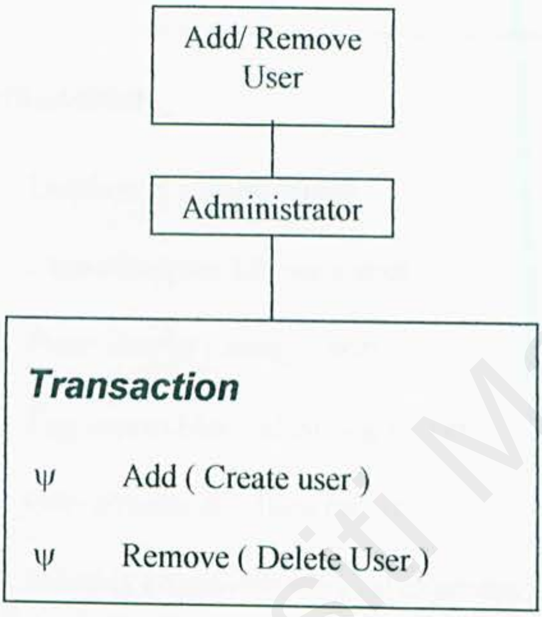


Figure 4.22 Add/ Remove User Module in Administrator views

a) Administrator View

This module allowed the administrator to coordinate the system by control the amount of users who can access to the **PDS**. The transaction in this module include add new user, and delete existing user from the **PDS**. The **PDS** did not provide function to check password if users forget their password because it will not be necessary since the amount of users will be no more than 100. As the result, administrator will delete those account which the users already forgot their password and create a new account for it.

4.5.7 Report Viewing Module

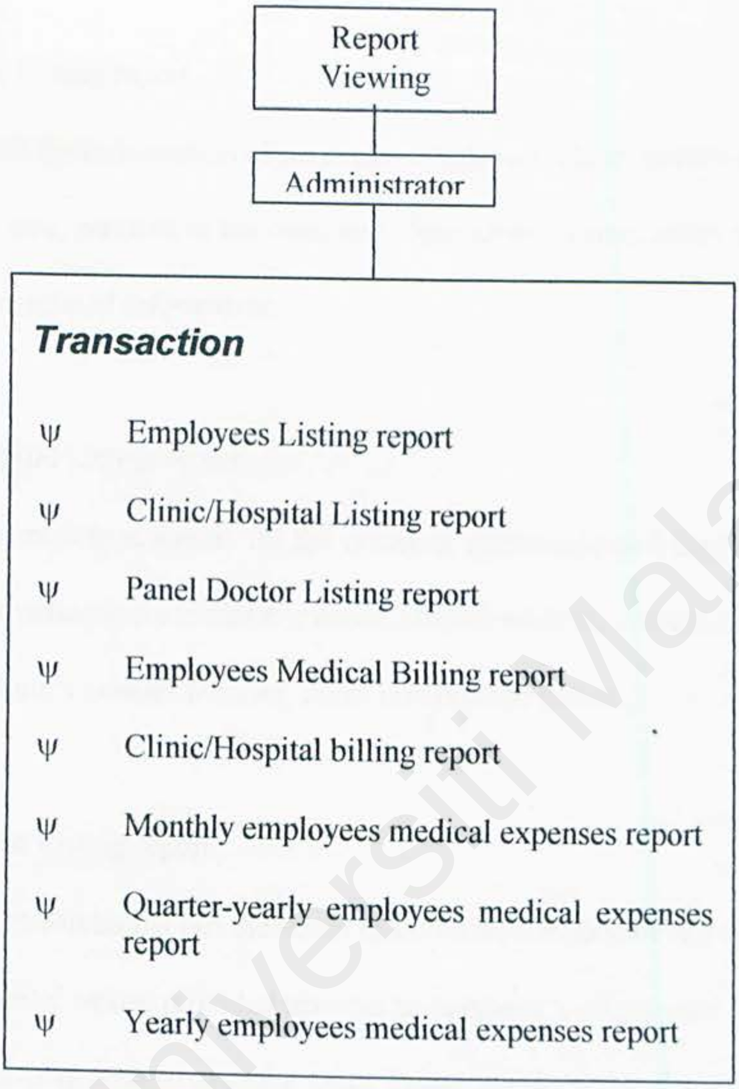

Report Viewing
Administrator
Transaction
ψ Employees Listing report
ψ Clinic/Hospital Listing report
ψ Panel Doctor Listing report
ψ Employees Medical Billing report
ψ Clinic/Hospital billing report
ψ Monthly employees medical expenses report
ψ Quarter-yearly employees medical expenses report
ψ Yearly employees medical expenses report

Figure 4.23 Add/ Remove User Module in Administrator views

a) Administrator View

Report viewing is another module which is visible only for administrator level users. This module comprises only vary types of generated reports by the system. The reports might be those had been analyze, or those just been categorize together. This can be describe as below:

Ω Employees Listing report

- List out all the information of particular employee. These include employee's name, age, sex, race, position in the company, dependent's name, salary in the company and others related information.

Ω Clinic/Hospital Listing report

- Provide complete summary for the company approved panel doctor and hospital.
- The information include clinic's name, clinic's address, amount of doctor in the clinic, clinic's contact number, faxes number and others.

Ω Panel Doctor Listing report

- This report will be list out the detail of clinic's panel doctor and the doctors from hospital whose provided services to company's employees. The report listing will be display according to the users. Either it will list by clinic, list by name or others sequences.

Ω Employee Medical Billing report

- This report generates overview records on the employees medical expenses. Inside the report there are information on employee's medical status, frequencies of particular employee visit the panel doctor, consultant's fees on each visit, and even medicine billing on the employees. Not forget that this report also include the employee's dependent who going to the panel doctor.

Ω Clinic/Hospital billing report

- The report is displaying information on clinical/hospital billing. What this means is that it will printout the accumulated amount of money which spend on each clinic/hospital.

Ω Monthly employees medical expenses report

- This report lists out the total expenses on company's employees medical billing in a month selected by the users. It will analyze the data in the database and display the information in the form of chart and data depends on the demands of the users.

Ω Quarter-yearly employees medical expenses report

- This is the report which generates the employees medical expenses accumulated in quarter-years time. It wills response according to the requirement of users. Where the data will be display in the form of chart or data type.

Ω Yearly employees medical expenses report

- This report collected all the information and generates final reports which conclude the company expenses on employees medical treatments in the whole year.

4.6 ENTITY RELATIONSHIP DIAGRAM

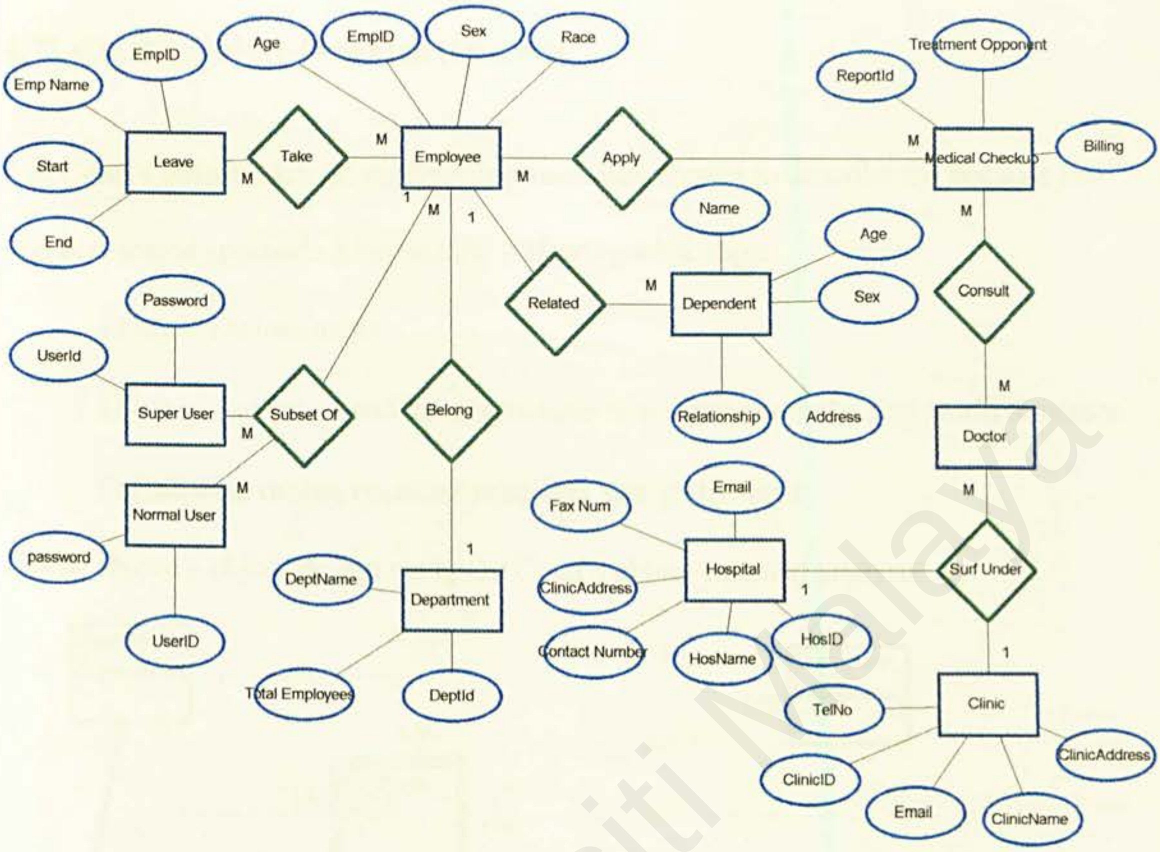


Figure 4.24 Entity Relationship Diagram of **PDS**

4.7 OBJECT-ORIENTED PROBLEM ANALYSIS

4.71 Coad's Object-Oriented Analysis

The Coad's object oriented analysis approach was chosen to describe the because the object oriented approach inherited the following advantages:

Ω More Maintainable

Ω Less error prone and have structures that mimic with the real world structure.

Ω Database design becomes relatively straightforward.

Below shown a object design using the Coad's object-oriented analysis:

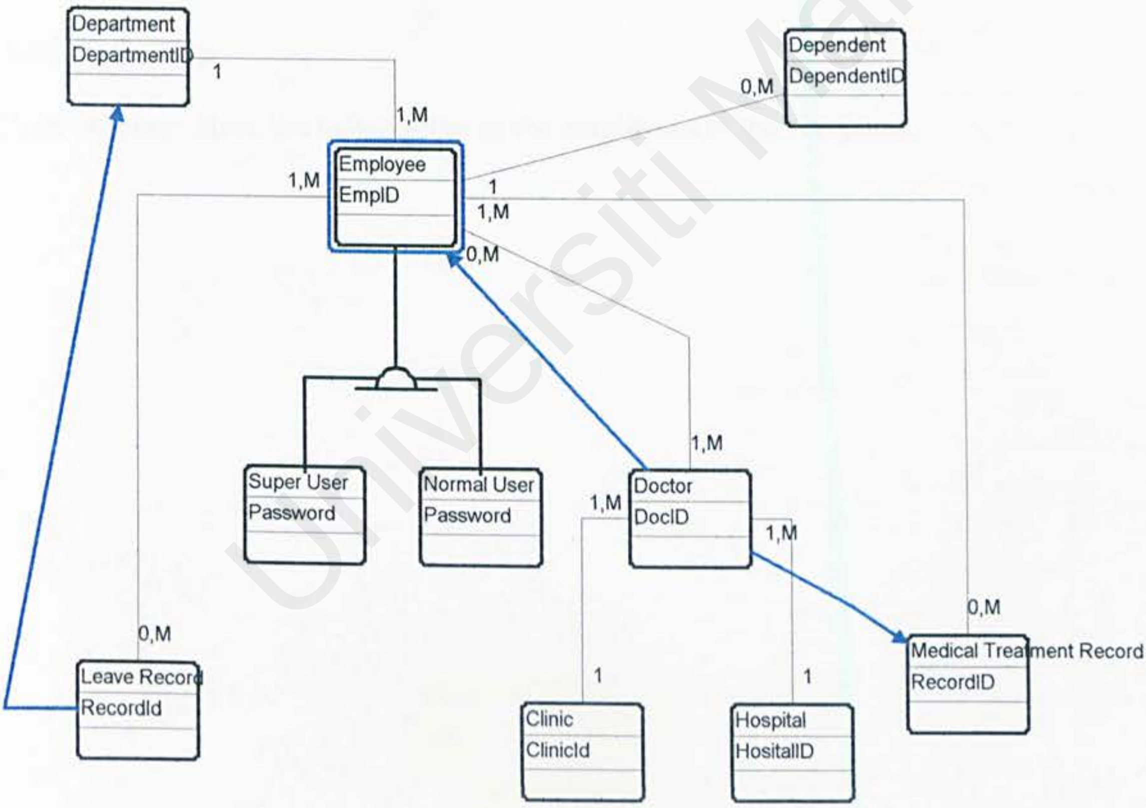


Figure 4.25 Complete OOA model of the **PDS**

4.8 DATABASE DESIGN

The **PDS** database was developed based on standard relational database model. In a relational database concept, table was defined as a collection of unique instances that include similar data. In addition, the normalization technique was used to that helps reduces data redundancies exist in the database, and helps to eliminate data anomalies result from those redundancies.

The database designs of PDS consists of nine tables, they were shown as below:

Table: pds_emp

Role-playing: Store the information of the employees in the company.

No	Field Name	Data Type	Value Range	Description
1.	*emp_id	Varchar (10)	Free Text	Employee ID
2.	Emp_name	Varchar (50)	Free Text	Employee Name
3.	Emp_age	Int	1-150	Employee Age
4.	Emp_sex	Char(1)	User Select	Employee Sex
5.	Emp_race	Varchar(10)	Free Text	Employee Race
6.	Emp_nric	Varchar(20)	Free Text	Employee IC Number
7.	Emp_license	Varchar(20)	Free Text	Employee License Number
8.	Emp_nation	Varchar(20)	Free Text	Employee Nationality
9.	Emp_status	Char(2)	User Select	Employee Status
10.	Emp_address	Varchar(100)	Free Text	Employee Current Address
11.	Emp_tel	Varchar(20)	Free Text	Employee telephone number
12.	Emp_fax	Varchar(20)	Free Text	Employee Fax Nimber
13.	Emp_email	Varchar(40)	Free Text	Employee Email Address
14.	Emp_position	Varchar(20)	Free Text	Employee Position
15.	Emp_med_stat	Varchar(10)	Free Text	Employee Medical Status
16.	Emp_account_num	Varchar(30)	Free Text	Employee Account Number
17.	Emp_salary	Currency	Free Text	Employee Salary in the Company

Table: pds_depd

Role-playing: Store the information of the employee's dependents.

No	Field Name	Data Type	Value Range	Description
1.	*depd_id	Varchar (10)	Free Text	Dependent ID
2.	Dept_name	Varchar(50)	Free Text	Dependent Name
3.	Dept_age	Int	1-150	Dependent Age
4.	Dept_sex	Char(2)	User Select	Dependent Sex
5.	Dept_relationship	Varchar(20)	Free Text	Dependent relationship with employee

Table: pds_spr_user

Role-Playing: store information of super user of the **PDS**

No	Field Name	Data Type	Value Range	Description
1.	*user_id	Varchar (10)	Free Text	User ID
2.	User_name	Varchar (30)	Free Text	User Name
3.	User_psw	Varchar (12)	Free Text	User password
4.	User_flag	Boolean	System Generated	Signal showing the state of the module.

Table: pds_nrm_user

Role-Playing: store information of normal user of the **PDS**

No	Field Name	Data Type	Value Range	Description
1.	*user_id	Varchar (10)	Free Text	User ID
2.	User_name	Varchar (30)	Free Text	User Name
3.	User_psw	Varchar (12)	Free Text	User password
4.	User_flag	Boolean	System Generated	Signal showing the state of the module.

Table: pds_clinic

Role-Playing: Storing Company approved clinics information.

No	Field Name	Data Type	Value Range	Description
1.	*clinic_id	Varchar (10)	Free Text	Clinic ID
2.	clinic_name	Varchar(50)	Free Text	Clinic Name
3.	clinic_address	Varchar(50)	Free Text	Clinic Address
4.	clinic_tel	Varchar (15)	Free Text	Clinic Contact Number
5.	clinic_fax	Varchar(15)	Free Text	Clinic Fax Number
6.	clinic_email	Varchar(50)	Free Text	Clinic email Address
7.	clinic_doc	Int	1-50	Total numbers of Doctor in the Clinic
8.	clinic_flags	Boolean	System Generated	Store the state of the module.

Table: pds_hospital

Role-Playing: Storing Company approved hospital information.

No	Field Name	Data Type	Value Range	Description
1.	*hospital_id	Varchar (10)	Free Text	Hospital ID
2.	hosital_name	Varchar(50)	Free Text	Hospital Name
3.	hospital_address	Varchar(50)	Free Text	Hospital Address
4.	hospital_tel	Varchar (15)	Free Text	Hospital Contact Number
5.	hospital_fax	Varchar(15)	Free Text	Hospital Fax Number
6.	hosital_email	Varchar(50)	Free Text	Hospital email Address
7.	hospital_doc	Int	1-50	Total numbers of Doctor in the Hospital
8.	hos_flags	Boolean	System Generated	Store the state of the module.

Table: pds_doctor

Role-Playing: Storing Company approved panel doctor information.

No	Field Name	Data Type	Value Range	Description
1.	*doc_id	Varchar (10)	Free Text	Doctor ID
2.	doc_name	Varchar(50)	Free Text	Doctor Name
3.	doc_address	Varchar(50)	Free Text	Doctor Current Address
4.	doc_tel	Varchar (15)	Free Text	Doctor Contact Number
5.	doc_serv	Varchar(50)	Free Text	Clinic/hospital that currently serve to..
6.	doc_email	Varchar(50)	Free Text	Doctor email Address
7.	doc_flags	Boolean	System Generated	Store the state of the module.

Table: pds_med_treat_rec

Role-Playing: Storing information on employees medical treatment records.

No	Field Name	Data Type	Value Range	Description
1.	*med_rec_id	Varchar (10)	Free Text	Medical Record Number/ ID
2.	med_emp_name	Varchar(30)	Free Text	Employee Name
3.	med_treat	Varchar(50)	Free Text	Medical Treatment
4.	med_visit_date	Date	yyyy-mm-dd	Employee treatment date
5.	med_clinic_visit	Varchar(50)	Free Text	Clinic Visit
6.	med_doc_const	Varchar(30)	Free Text	Doctor Consult
7.	med_bill	Currency	Free Text	Employees Medical Billing

Table: pds_leave_rec

Role-Playing: Store employees leave records.

No	Field Name	Data Type	Value Range	Description
1.	*leave_id	Varchar (10)	Free Text	Leave Record Number/ ID
2.	leave_emp_name	Varchar(30)	Free Text	Employee Name
3.	leave_reason	Varchar(50)	Free Text	Employees leave reason
4.	leave_apply_date_start	Date	yyyy-mm-dd	Leave apply Starting Date
5.	leave_apply_date_end	Date	yyyy-mm-dd	Leave apply End Date
6.	leave_appve_date_start	Date	yyyy-mm-dd	Leave Approved Starting Date
7.	leave_appve_date_end	Date	yyyy-mm-dd	Leave Approved End Date

CHAPTER 5:

SYSTEM IMPLEMENTATION

This section will be discussed about the some of the modifications to the previous design due to the limitation of the programming language used. Here it will show the coding method in detail.

5.1 DEVELOPMENT ENVIRONMENT

The implementation of the **PDS** consists of hardware and software requirement as shown below:

5.1.1 Hardware Requirement

The following hardware specific ations have been used to develop **Employee's Panel Doctor System**:

- Pentium III 700 MHz CPU
- 256 MB RAM
- 512K Pipeline Burst Cache
- 52X CD-ROM Drive
- 1.44 MB Floppy Drive

- 10.5 GB Hard Disk

5.1.2 Software Requirement

The following software specifications have been used to develop **ETS**:

i) Microsoft Windows NT Server 4.0 with NT Option Pack 4.0

It is act as the operating system for the whole program.

ii) Microsoft Visual Basic 6.0 Enterprise Edition

Required in the development phase. It is a main programming language used to implement the whole system.

iii) ODBC 32-bit Driver

Used to Connecting database between Visual Basic and the database. (SQL Server 6.0)

iv) Adobe Photoshop 6.0

Used as Image design and manipulation tolls in designing the user interface.

v) MS SQL Server 6.0

Act as the system database to store the information insert by the PDS system.

5.1.2.1 Software Tools for Report Writing and Design

Microsoft Word 2000 is used to write the report and MetaEdit has been chosen to draw the DFD, Structure Chart and System Model.

5.1.2.2 Software Tools for Database

MSSQL Server 6.0 was chosen due to it capability to handle large amount of data such as company information system. Besides, it's having a strong database security features that can avoid unauthorized access into the database that used to store confidential data. The database is accessed using ADODB methods.

5.2 MODULE IMPLEMENTATION

Generally, there are two levels of users that can access to the **PDS**. They are the Administrator and the Normal User. Administrator are those who having authority to access the entire module in the **PDS** whereas the Normal Users are having part of the responsibility on the system. As the results, there will be no addition module for the normal users.

The **PDS** was developed using the Microsoft Visual Basic 6.0 that is an Event-Driven programming language, which it means that it giving response based on the action did by the user. Thus, it will involve many functions depends on what users do. The following will be discussed some important functions implement in each modules, purpose of the method, parameters passed, and verifications run by the Visual Basic.

5.2.1 PDS logon Module

- Sub Form_Load()
  - Purpose: get Connected To the database and perform User Checking
  - Parameters: None
  - Verifications:
    - The Database is connected

- Private Sub GetLoginEmp()
  - Purpose: Check the user password and verify the user while login.
  - Parameters: UserId, UserPsw
  - Verifications:
    - The User ID is in the right format

- Private Sub GetLogoutEmp()
  - Purpose: Check the user password and verify the user while logout.
  - Parameters: UserId, UserPsw
  - Verifications:

        - The User ID is in the right format

- Private Sub SetLogin()
    - Purpose: Set User Flag As "On Use" when authorized user login.
    - Parameters: None
    - Verifications:
        - The User ID Is authorized
        - The User Password is verified.

- Private Sub SetLogout()
    - Purpose: Set User Flag As "Non Use" when authorized user Logout from the **PDS**.
    - Parameters: None
    - Verifications:
        - The User ID Is authorized
        - The User Password is verified.

5.2.2 MDI Main program Platform

- Private Sub GetUserState()
    - Purpose: Check database to check the status of the user. Whether he / she is an administrator level user or normal user.
    - Parameters: User login Flag

- Verifications:
  - The User ID Is authorized and already login
  - The User Password is verified.

5.2.3 PDS Main Menu Module

- Private Sub GetUser()
  - Purpose: Check database to check the status of the user. Whether he / she is an administrator level user or normal user.
  - Parameters: User login Flag
  - Verifications:
    - The User ID Is authorized and already login
    - The User Password is verified.

5.2.4 Employee's Record Module

- Public Sub TotalEmpRecord()
  - Purpose: Get the value from GetEmpRecord() and return the Employees' ID increment by one.
  - Parameters: None
  - Verifications:

        - None

- Private Sub GetEmpRecord()
    - Purpose: Calculate the total employees' record in the database.
    - Parameters: None
    - Verifications:
        - None

- Private Sub UpdateEmpRecord()
    - Purpose: Updating the total employee's record in the database after inserting the record in the **PDS**.
    - Parameters: None
    - Verifications:
        - The Insert Button had been activate
        - The record had been added.

- Private Sub GetEmpInfo()
    - Purpose: Get and Retrieve all the information from the database base on the user id that user key in.
    - Parameters: User ID
    - Verifications:
        - The employees' record exists in the database.
        - The employees' records in the database are complete.

- Private Sub LoadEmpRetrieve()
  - Purpose: Load and display the employees' information based to the command by the user.
  - Parameters: User ID
  - Verifications:
    - The employees' record exists in the database.
    - The employees' records in the database are complete

- Private Sub LoadEmpAddRetrieve()
  - Purpose: Load and display the employees' Address information based to the command by the user.
  - Parameters: User ID
  - Verifications:
    - The employees' address record exists in the database.
    - The employees' records in the database are complete

- Private Sub CheckDependRec()
  - Purpose: Check the database whether the dependents records exist in the database. (Single record or Double Record)
  - Parameters: Employees' ID
  - Verifications:
    - The employees' Id is verified.

- Private Sub RetrieveDepartment()
  - Purpose: Retrieve the existing department name in the database into the selection column to enable user selecting the correct department.
  - Parameters: None
  - Verifications:
    - The department record in the database is not NULL.

5.2.5 Panel Doctor Record Module

- Private Sub Form_Load()
  - Purpose: Connecting to the database and perform checking on the user status. If normal user status detected. Hide unauthorized module.
  - Parameters: None
  - Verifications:
    - The database connection is valid.

- Private Sub GetUserState()
  - Purpose: Check database to check the status of the user. Whether he / she is an administrator level user or normal user.
  - Parameters: User login Flag
  - Verifications:
    - The User ID Is authorized and already login

        - The User Password is verified.

- Public Sub TotalDocRecord()
    - Purpose: Get the value from GetDocRecord() and return the Doctor's ID increment by one.
    - Parameters: None
    - Verifications:
        - None

- Private Sub GetDocRecord()
    - Purpose: Calculate the total doctor's record in the database.
    - Parameters: None
    - Verifications:
        - None

- Private Sub GetClinicRecord()
    - Purpose: Get the clinic record from the database based n the clinic name.
    - Parameters: Clinic Name
    - Verifications:
        - Clinic name is a valid name.

- Private Sub LoadClinic()
    - Purpose: Get the clinic record from GetClinicRecord() and displays the information in the form.
    - Parameters: Clinic Name
    - Verifications:
        - Clinic name is a valid name.
        - GetClinicRecord() had capture the data from the database.

- Private Sub GetDocInfo()
    - Purpose: Get the doctor record from the database.
    - Parameters: Doctor ID
    - Verifications:
        - Doctor ID is a valid name.

- Private Sub LoadDocRetrieve()
    - Purpose: Get the doctor record from GetDocInfo() and displays the information in the form.
    - Parameters: Doctor ID
    - Verifications:
        - Doctor ID is valid.
        - GetDocInfo() had capture the data from the database.

- Private Sub LoadDocAddRetrieve()
    - Purpose: Get the doctor address record from GetDocInfo() and displays the information in the form.
    - Parameters: Doctor ID
    - Verifications:
        - Doctor ID is valid.
        - GetDocInfo() had capture the data from the database.

- Private Sub RetrieveClinic()
    - Purpose: Get the Clinic name and hospital name from the database and retrieve the name to the selection combo box.
    - Parameters: None
    - Verifications:
        - Clinic information exists in the database.
        - Hospitals information exists in the database.

- Private Sub GetMedCenterData()
    - Purpose: Get the Clinic information and hospital information from the database and retrieve the information.
    - Parameters: Clinic Name, hospital Name
    - Verifications:
        - Clinic information exists in the database.

  - Hospitals information exists in the database.

5.2.6 Employee's Medical Billing Module

- Private Sub Form_Load()
  - Purpose: Get Connecting to the database, perform User status check, Get Medical Record ID, and call function to retrieve all the information needed for interface selection.
  - Parameters: None
  - Verifications:
    - Connection to the database is valid.

- Public Sub TotalMedRecord()
  - Purpose: Get the value from GetMedRecord () and return the Medical Record ID increment by one.
  - Parameters: None
  - Verifications:
    - None

- Private Sub GetMedRecord()
  - Purpose: Calculate the total medicals record in the database.
  - Parameters: None
  - Verifications:

        - None

- Private Sub UpdateMedRecord()
    - Purpose: Updating the total medicals record in the database.
    - Parameters: None
    - Verifications:
        - None

- Private Sub RetrieveDoctor()
    - Purpose: Get existing Doctor Name from the database and display it on the selection box.
    - Parameters: None
    - Verifications:
        - The doctor records exist.

- Private Sub RetrieveEmpId()
    - Purpose: Get existing Employees ID from the database and display it on the selection box.
    - Parameters: None
    - Verifications:
        - The Employees records exist.

- Private Sub GetEmpDetail()
    - Purpose: Get existing Employees information based on the employee ID from the database and display it on the selection box.
    - Parameters: Employee's ID
    - Verifications:
        - The Employees records exist.

- Private Sub GetDepartment()
    - Purpose: Get existing Department name from the database and display it on the selection box.
    - Parameters: None
    - Verifications:
        - The Department records exist.

- Private Sub GetEmpName(ByVal temp As String)
    - Purpose: Get employee name from the database base on the Emp ID insert by the user and display it on the selection box.
    - Parameters: Employees ID
    - Verifications:
        - The Employees records exist

5.2.7 Leave Requirement Module

- Private Sub Form_Load()
  - Purpose: Get Connecting to the database, perform User status check, Get Total Leave Record..
  - Parameters: None
  - Verifications:
    - Connection to the database is valid.

- Private Sub GetUserState()
  - Purpose: Check database to check the status of the user. Whether he / she is an administrator level user or normal user.
  - Parameters: User login Flag
  - Verifications:
    - The User ID Is authorized and already login
    - The User Password is verified.

- Public Sub TotalLeaveRecord()
  - Purpose: Get the value from GetLeaveRecord () and return the Leave Record ID increment by one.
  - Parameters: None
  - Verifications:
    - None

- Private Sub GetLeaveRecord()
  - Purpose: Calculate the total leave record in the database.
  - Parameters: None
  - Verifications:
    - None

- Private Sub UpdateLeaveRecord()
  - Purpose: Updating the total leave record in the database.
  - Parameters: None
  - Verifications:
    - None

- Private Sub RetrieveRecord()
  - Purpose: Search the leave record from the database based on the leave record ID.
  - Parameters: Leave ID
  - Verifications:
    - Check the validation of the database.

- Private Sub LoadRetrieve(strEmpId As String, strDesc As String, llSet As Long)
  - Purpose: Search the leave record from the database based on the leave record ID and retrieve the information found.

- Parameters: Leave ID
- Verifications:
    - Verify the data consistency from the database..

5.2.8 Maintenance Module

- Private Sub ChangePsw()
    - Purpose: Change user password and replace old password.
    - Parameters: User ID, User Password
    - Verifications:
        - Check the existing of the User ID from the database.

- Private Sub RetrieveUserId()
    - Purpose: Retrieve existing User ID and other information to check the validation of the data.
    - Parameters: User ID, User Password
    - Verifications:
        - Check the existing of the User ID from the database.

5.2.9 Add Remove User Module

- Private Sub GetEmp()
  - Purpose: Get the Existing Employees ID from the emp_record database to find the valid employees ID.
  - Parameters: None
  - Verifications:
    - Check the existing of the Employees ID from the database.

- Private Sub RetrieveUserId()
  - Purpose: Get the Existing User ID from the User_record database and retrieve the information to be remove..
  - Parameters: None
  - Verifications:
    - Check the existing of the Users ID from the database.

- Private Sub GetEmpName()
  - Purpose: Get the Employee's detail from the database and display it hence check the user ID with the existing Emp ID.
  - Parameters: None
  - Verifications:
    - Check the existing of the Users ID from the database.
    - Check the existing of the Employees ID from the database.

5.2.10 Report Viewing Module

- Private Sub Form_Load()
    - Purpose: Call function to retrieve all the selections of employees ID, hospital name, clinic name, dependent ID, medical record ID.
    - Parameters: None
    - Verifications:
        - Connection to the database is valid.

- Private Sub RetrieveDepartment()
    - Purpose: Retrieve the existing department name in the database into the selection column to enable user selecting the correct department.
    - Parameters: None
    - Verifications:
        - The department record in the database is not NULL.

- Private Sub RetrieveEmp ()
    - Purpose: Get existing Employees ID from the database and display the employee ID on the selection box.
    - Parameters: None
    - Verifications:
        - The Employees records exist.

- Private Sub RetrieveClinic()
    - Purpose: Get the Clinic name and hospital name from the database and retrieve the name to the selection combo box.
    - Parameters: None
    - Verifications:
        - Clinic information exists in the database.

- Private Sub RetrieveHospital()
    - Purpose: Get the Hospital id and hospital name from the database and retrieve the name to the selection combo box.
    - Parameters: None
    - Verifications:
        - Hospitals information exists in the database.

- Private Sub RetrieveDoctor()
    - Purpose: Get existing Doctor Name from the database and display it on the selection box.
    - Parameters: None
    - Verifications:
        - The doctor records exist.

- Private Sub LoadYear()
    - Purpose: Calculate the year between 1900 until 2100 and allow the value to be selected in the cmbobox.
    - Parameters: None
    - Verifications:
        - None.

- Private Sub GetMonthFr()
    - Purpose: Calculate the month and change the data into database which can be read by the SQL server.
    - Parameters: month insert by user
    - Verifications:
        - input are valid from the selection box.

- Private Sub GetMonthTo()
    - Purpose: Calculate the month and change the data into database which can be read by the SQL server.
    - Parameters: month insert by user
    - Verifications:
        - input are valid from the selection box.

5.3 CLASS MODULE IMPLEMENTATION

5.3.1 Class Module cls_dbconnect

This class is to connect the system to the database. Function include in this class is:

- Public Function GetConnect() As Connection
    - Purpose: Perform connection to the database
    - Parameters: None
    - Verifications:
        - Database exist

5.3.2 Class Module cls_encrypt

This class is to encrypt the user password and return a dummy password storing into the database.

- Public Function EncryptPsw(ByVal strPsw As String) As String
    - Purpose: Change the user password and encrypts the password
    - Parameters: User Password
    - Verifications:
        - None

5.3.3 Class Module cls_docrec

This class is to perform record insertion into the database by calling the stored procedure at the SQL Server 7.0

- Public Function AddDocRec()
  - Purpose: Add the doctor record into the database
  - Parameters: objconn, strDocId, strDocName, strRace, strSex, IntAge, strNation, strDesc, strMedServ, strError
  - Verifications:
    - All data insert are valid.

- Public Function AddDocAddrRec()
  - Purpose: Add the doctor address record into the database
  - Parameters: objconn, strDocId, stradd, strstate, strHPNo, strPNo, strFax, strEmail, strError
  - Verifications:
    - All data insert are valid.

- Public Function UpdateCliAddrRec()
  - Purpose: Update Clinic address record into the database
  - Parameters: objconn, strCliId, strCliAdd, strsNum, strFax, strEmail, strError,
  - Verifications:

        - All data insert are valid.

- Public Function UpdateDoc()
    - Purpose: Update doctor record into the database
    - Parameters: objconn, strDocld, strDocName, strsDocRace, strSex, IntDocAge,strDocNation, strDesc, strServ , strError,
    - Verifications:
        - All data insert are valid.

- Public Function DeleteDoc()
    - Purpose: Delete doctor record inside the database
    - Parameters: objconn, strDocld, strError,
    - Verifications:
        - Valid data retrieve.
        - Data exist in the database.

- Public Function DeleteDocAdd()
    - Purpose: Delete doctor address record inside the database
    - Parameters: objconn, strDocld, strError,
    - Verifications:
        - Valid data retrieve.
        - Data exist in the database.

5.3.4 Class Module cls_emp

The class is taking responsibility in inserting user's record into the database.

- Public Function GetEmp(ByRef m_objConn As ADODB.Connection)
    - Purpose: get user detail from the database
    - Parameters: objconn
    - Verifications:
        - Valid user ID gets.
        - Data exist in the database.

- Public Function AddEmp()
    - Purpose: Insert user detail into the database
    - Parameters: objconn, strEmpID, strEmpName, strPsw, strUserFlag, strUserState, strError,
    - Verifications:
        - Valid user ID gets.
        - All insertion data are valid.

- Public Function DeleteUser()
    - Purpose: Delete user detail from the database
    - Parameters: objconn, strEmpID, strError,
    - Verifications:
        - Valid user ID gets.

5.3.5 Class Module cls_empinfo

This class module are perform tasks as insert the employees record into the database, updating the record into the databsase and delete record from the database.

- Public Function AddEmpRec()
  - Purpose: Insert employees detail into the database
  - Parameters: objconn, strEmpID, strEmpName, strRace, strNation, strSex, IntAge, strStatus, datedob, strNric, strPassport, strdeartment, strPosition, strAccount, douSalary, strDesc, strError
  - Verifications:
    - Valid employees ID gets.
    - All insertion data are valid.

- Public Function UpdateEmp()
  - Purpose: Update employees detail into the database
  - Parameters: objconn, strEmpID, strEmpName, strRace, strNation, strSex, IntAge, strStatus, datedob, strNric, strPassport, strdeartment, strPosition, strAccount, douSalary, strDesc, strError
  - Verifications:
    - Valid employees ID gets.
    - All insertion data are valid.

- Public Function DeleteEmp()
    - Purpose: Delete employees detail from the database
    - Parameters: objconn, strEmpID, strError,
    - Verifications:
        - Valid employees ID gets.

5.3.6 Class Module cls_leave

cls_leave was used by system to control all the operation of the Leave Requirement Module.

- Public Function AddLeaveRec()
    - Purpose: Insert new leave record into the database
    - Parameters: objconn, strLeaveId, strEmpID, DtStart, DtEnd, strDesc, InSet, strError
    - Verifications:
        - Valid Leave ID gets.
        - All insertion data are valid.

- Public Function UpdateRec()
    - Purpose: Update existing leave record into the database
    - Parameters: objconn, strLeaveId, strEmpID, DtStart, DtEnd, strDesc, InSet, strError
    - Verifications:

        - Valid Leave ID gets.
        - All insertion data are valid.

- Public Function DeleteRec()
    - Purpose: Delete leave record from the database
    - Parameters: objconn, strLeaveld, strError,
    - Verifications:
        - Valid leave ID gets.

- Public Function CheckRec()
    - Purpose: Check the validation of the data require by the user
    - Parameters: objconn, strLeaveld, strError,
    - Verifications:
        - Valid leave ID gets.

5.3.7 Class Module cls_login

cls_loogin is the Microsoft Visual Basic Class Module Created to handle all the login operation in the **PDS.**

- Public Function GetLogin()
    - Purpose: Check the user information while they were login.
    - Parameters: objconn, strUserId, strUserPsw,
    - Verifications:
        - UserId format check.

- Public Function GetState()
    - Purpose: Get the User State from the database to check the user level.
    - Parameters: objconn
    - Verifications:
        - Existing UserId.

- Public Function GetUser()
    - Purpose: Return the flag to verify whether the user are online.
    - Parameters: objconn, strTag
    - Verifications:
        - Existing UserId.

- Public Function UpdatePsw()
    - Purpose: Change User password and update into the database.
    - Parameters: objconn, strUserId, strnewPsw, strError
    - Verifications:
        - Existing UserId.

- Public Function UpdateUser()
    - Purpose: Update the user Tag when user logout from the system. .
    - Parameters: objconn, strUserId, strTag, strError
    - Verifications:

- Existing UserId.

5.3.8 Class Module cls_medrec

- Public Function AddMedRec()
    - Purpose: Insert new medical record into the database
    - Parameters: objconn, strMedId, strVisitDate, strCliVisit, strDogChg, strMedTreat, strMedBill, strEmpId, strEmDepd, strEmpDepart, strError
    - Verifications:
        - Valid medical ID gets.
        - All insertion data are valid.

- Public Function UpdateMedBill()
    - Purpose: Insert new medical record into the database
    - Parameters: objconn, strMedId, strVisitDate, strCliVisit, strDogChg, strMedTreat, strMedBill, strEmpId, strEmDepd, strEmpDepart, strError
    - Verifications:
        - Valid medical ID gets.
        - All insertion data are valid.

- Public Function DeleteMedRec()
    - Purpose: Delete selecting medical record from the database.
    - Parameters: objconn, strMedId, strError

  - Verifications:
    - Valid medical ID gets.

5.3.9 Class Module cls_report

The following functions are the function to call out all the clinic and hospital information from the database to be list out on the report

- Public Function GetAllClinicInfo()
  - Purpose: Select all existing clinic name from the database to be display in the report as selection item.
  - Parameters: objconn
  - Verifications:
    - Existing of the clinic record in the database.

- Public Function GetAllHospitalInfo()
  - Purpose: Select all existing hospital name from the database to be display in the report as selection item.
  - Parameters: objconn
  - Verifications:
    - Existing of the clinic record in the database.

5.3.10 Class Module cls_decrypt

This class is to decrypt the dummy password and return a user password. (Created but not practiced in the system because is not necessary)

- Public Function DecryptPsw(ByVal strPsw As String) As String
    - Purpose: Change the user password and encrypts the password
    - Parameters: User Password
    - Verifications:
        - None

5.4 DOCUMENTATION

Internal documentation in the source code was used to provide information that identifies the program, describe its data structures, algorithms and control flow. The header comment block approach is used to provide above-mentioned information.

CHAPTER 6: SYSTEM TESTING

The Software System testing and debugging are critical elements in a system development. It is aim to discover bugs and defects that present in the system. A System testing will only be recognized as successful only if errors are detected. Generally, system testing is focusing on:

- Demonstrate that the software have being implement and has meet the software requirement specification.
- Shows that errors handling has being done.
- Reveal different classes of errors and introduce the solution for each error.
- Report on any bugs that can not be resolve. In this case, the programming language bugs.

For the PDS system, the integration and system testing has been carried out to test for multiple errors based on certain test cases. The PDS has being tested based on the following generic characteristic:

- The process starts from the main login module and go through module by module toward the integration of the entire system.
- Testing and debugging have been separated and treat as two different activities.

- Several testing technique have been used at different points in time.

6.1 PDS TESTING TECHNIQUE

The Employee's Panel Doctor System PDS have been going through two different testing techniques which is Grass Box testing and Black Box testing.

6.1.1 Grass Box Testing

Grass Box testing, as it's name called. Is a software testing technique that is done focusing on the software functionality without go through it's code. It is a test case design method that uses the control structure of the procedural design to derive test cases. Here, the PDS has being test by normal user without training to trace and check for the error that occurs.

The user are those who has no ideas on what this system is, they are just guide by the developers on how to use each module and what each modules do. The tester will enter anything they decide without concerning the sequence of each module. This process is to check whether the module has sequence limitation, which means that it must follow certain process before continue to another module. In **PDS** for example, the user will navigate through doctor record module, employee's record module, maintenance module as they like without any sequence. In this case, this will also check the data validation of the system. It assures the internal data structures validity.

Besides, by using the White Box testing, event bugs are also tested that usually occurs when the users are using the system without knowing the detail of the system. It will guarantee that all The PDS module have been exercised at lease once and all the logical exercise decisions in the PDS have been went through.

With White Box Testing, it will ensure internal operations of the system perform according to specifications and all internal components have been adequately exercised.

6.1.2 Black Box Testing

Black Box Testing are technique that exercised by software tester focus on the functional requirement of the software. Here, the software tester will navigate through all the code of the system row by row and performing data debugging in detail. This is done enables Software engineers to derive set of input condition that will fully exercise all functional requirements for the **PDS**.

Black box testing attempt to find errors on the data validation in the system.. Besides, it wills also checkout the all the logical decision in detail. The tester are debugging based on the flow of the system and the data Flow Diagram to check the data integrity.

Black Box testing is generally focus on source code of the system. As the result, tester will more concentrate to check the missing functions or invalid function that occurs in the system. Besides, redundant functions were list out and will be removing from the system to reduce the complexity of the code. Errors in data structures are always check because it is a fatal error seems PDS is a company information system that held important and private information that cannot occurs data inconsistency.

Black Box testing is exercise by the end of the testing phase because it requires a complete **PDS** system source code to be walkthrough.

6.2 SOFTWARE TESTING STRATEGIES

The Employee's Panel Doctor System Testing process were been done based on the standard software testing strategies which can be derive as :

1. Unit Testing
2. Integration Testing and,
3. System Testing

6.2.1 Unit Testing

Unit testing focuses on the most basic are focus on the smallest unit of software design - the module. It is done to verify that the elementary elements of the product run correctly on the host computer, independently of their target environment. For the **PDS** System, unit testing are being done start from the login module. The source will be check and debug row by row in detail.

On the Unit Testing Phase, it is attempt to identify such items as infinite loops, paths through the code which should be allowed, but which cannot be executed, and unreachable code. **PDS** was tested in an ordered way by following unit testing, integration testing and functioning testing. A unit was tested one by one and isolated with other units or modules. The unit was tested by different input and the result is observed. If the result is incorrect, the codes within the unit have to be checked against the algorithmic fault. The testing and coding was repeated again and again until the expected result obtained. For example, the **Add Remove User** module is a unit that only performs the function to add a user into the system on behalf of the administrator. This unit will perform validation on the input and produce a result to acknowledge the administrator whether the insertion success or not.

Besides ensuring the function of a unit was performed correctly, the codes inside a unit were reviewed in order to spot any faults, documentation, to check efficiency on data usage, and the unit's interface. To review the code, Code walkthrough technique has been adopted because this approach is simple and easy. This will done by a group of people where they presenting his or her codes and accompanying documentation to other members.

6.2.2 Integration Testing

In short, Integration Testing is a process to verify that all the elements that compose the product can run together. This testing will ensure that interface such as modules calling sequence in **PDS** is arranged correctly.

There are two approaches for exercising the integration testing. It is call incremental integration strategy, the bottom-up integration and regression testing approach. And for the **PDS**, bottom-up testing has need used. The main reason that this approach being chosen was because simple to implement. If any integration failure occurs, the integrated system could be split to its original modules, then repeat the unit testing again.

To shows step by step how **Administrator** modules being integrated. First step it will performed unit testing, then second step merged related modules and perform testing again, the third step merged all the integrated modules in step 2 into a larger integrated system, and run testing again. This will show at the chart below:

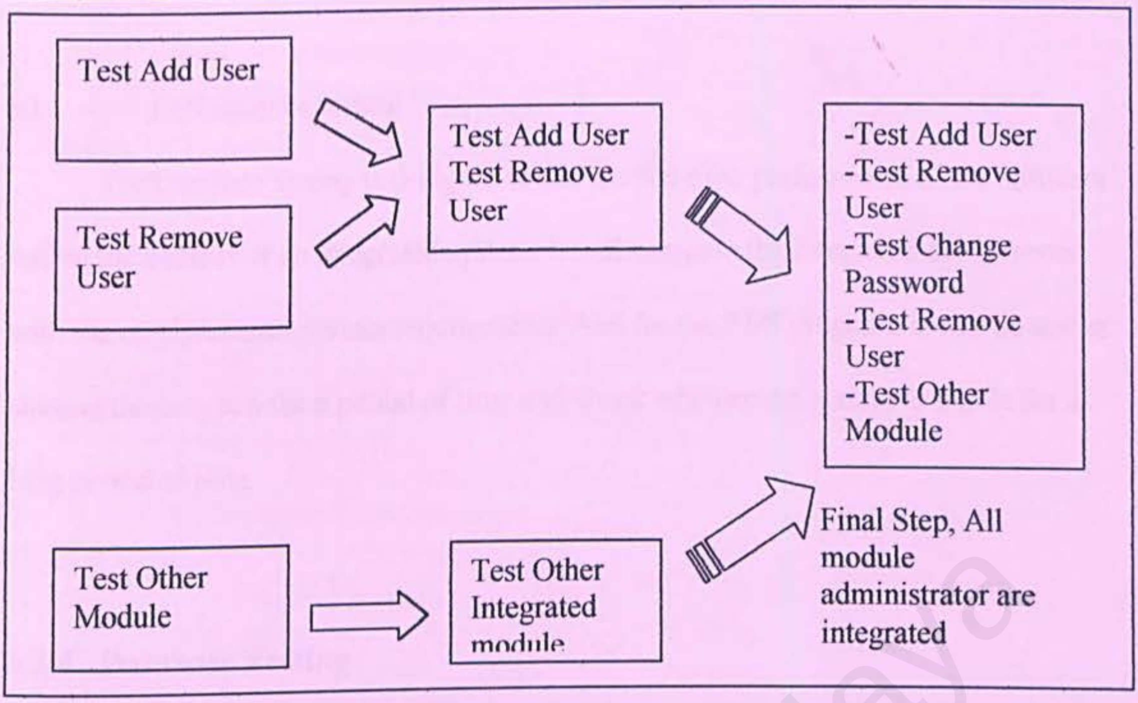


Figure 6.1 Integration Testing on Administrator Modules

6.2.3 System Testing

System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system. For the **PDS**, the primary purpose of this testing is to verify that all system elements have been properly integrated and perform the allocated functions as specified in the requirements.

System testing can be performing in the series of process as shown below:

i) Function testing

The previous tests concentrated on components and their interactions. Now the **PDS** integrated system has to run through functioning testing to make sure the system fulfill functional requirements. At this point, **PDS** was tested by login different user, input valid and invalid values, and observe the messages responded by the system.

ii) Performance testing

Performance testing is designed to test the run-time performance of the software within the context of an integrated system. It will compare the integrated components with the nonfunctional system requirements. And for the **PDS** program, it will be test as starting the program for a period of time and check whether the system is stabile for a long period of time.

6.2.4 Database Testing

To finalize the testing process, the data consistency in the database will be check. In the case, the accuracy and integrity of data stored by the server is tested. Transactions posted by **PDS** are examined to ensure that data are properly stored, updated, and retrieved.

CHAPTER 7: SYSTEM EVALUATION

Employee Panel Doctor System **(PDS)** is a client server application developed to provide a company with computer-based employee's medical information system that can help them to store employee particulars, company panel doctor detail, employees medical expenses record and generate company monthly medical expenses report, as well as employees medical expenses report.

PDS can be categorized as pure information system where it doesn't contain of complex internal calculation functions, complex logic transactions as well as computer-based decision making functions. The management and the decision making are all depend on the administrator of the system and the top level management group who use the system.

7.1 SYSTEM STRENGHT

7.1.1 General Solution

The **PDS** is a useful system providing for company to store their medical expenses and their employee's information. It integrated a general employee's information system and medical expenses system in a single program. This system can be easily adapt to changes and can be apply to other organization because most of the company system on storing their employees' data are the same.

The **PDS** are ready to be customized when needed if the company policies are different. Multiple functions can be developed to provide greater flexibility to the company when needed.

7.1.2 User Friendliness

PDS is ready with user-friendly interface and functions. It has a standard and systematic Software system design such as command buttons, text boxes, drop-down menus, etc. Users, who are experienced in Computer applications, can easily use this system. It is no other different comparing with standard information system application.

Besides, **PDS** provides sufficient instructions and guidance on the interface to enable users to use the system effectively and with ease. Because the user of the **PDS** are all the office workers such as clerk, the **PDS** interface and button are more text than icon because they know text command well compares with picture icon that sometimes create confusions to the users.

This system was developed to be robust to handle any invalid input into the system. Error messages will be displayed to guide user whenever an invalid input is encountered.

7.1.3 Access Level

In the **PDS**, the normal users are all having limitation on accessing to certain module that is just visible for the administrator status users of the system. They are not allowed to access protected module that is having private and confidential material of the company.

This means that as there are two levels of users in this system, each level of user has the different access right. The access level details have been discussed in the earlier section at **System Design.**

7.1.4 Main Index

Every module inside the **PDS** system is concluding inside a main index. Main index acts as a central for controlling all the modules.

7.1.5 System Security

Security issues are always being taken into consideration for the administrator section so as to avoid unauthorized users from breaking into the system, harming the system and accessing the confidential information of the company. This is done through the implementation of a login procedure before a user can gain access into the system.

7.1.6 System Transparency

System transparency refers to the condition where users do not need to know how the system is structured, where the database resides, its database management system or any details related to the system built. This is to ensure that it does not confuse users in retrieving information.

7.1.7 Database Security

Database security emerges as a very important issue because all the information is stored in the database. Hence there is a main database users created in SQL Server. Each type of users has different privileges on the database, this is to avoid that an authorized user retrieves and updates data which he or she not suppose to.

7.1.8 Report

As the **PDS** provide information to the company, there are prepared with seven reports generated by **PDS** for report keeping and checking for error.

7.2 LIMITATION OF PDS

Refer to the time constraint and the constraints of the programming language itself, there are some limitations in **PDS**. These include:

7.2.1 Convert Currency Data Type into the Database

Because of the limitation of the data type in Visual Basic and SQL Server, the money value in the program cannot be inserting in the correct format into the database. Which means that all the value inserting into the database are all in the non floating point value As the data being retrieve in the report, the money value will be shown integer format.

7.2.2 Screen Resolution

The **PDS** system is developed in 1024 x 768 resolution environment. As the result, the system interface might happen to be change when it was developed in 800 x 600 resolution environment. Anyway, the system functions work well in any resolution.

7.2.3 The Information Cannot Be Shown in Data Grid Format

Report Viewing Module in the **PDS** does not allowed user to view the record enters by the user in data grid format which will list out the entire information database. The users are only view the record in one by one format.

7.3 FUTURE ENHANCEMENT

Due to the ability of the system to be customize to support other organization in the future, further development and many new ideas have come. Owing to time constraints, not all of these ideas could be incorporated into the system. But the ideas and consideration in future had been listed down as below:

7.3.1 Extend To Other Information

The **PDS** provide medical record information for the company. Thus it can be extend to have another module such as company income record as well.

7.3.2 Enhance the Leave Application Module to Be Access by Company Employees

Currently, the Leave Requirement module in **PDS** is just accessible by authorized user. It is planning to be access by employees to perform computer-based leave application.

7.3.3 Provide Information on the Record Enter in overall Format

There is planning to have an option to be able to view the content of overall record without going into the report viewing module.

7.3.4 Provide More Statistic for Decision Making

The **PDS** will be more complete when the system has added with Chart, Graph and others information statistic diagram to enable Top Management to view the medical record in the company clearer.

7.4 OBJECTIVE ACHIEVED

The **PDS** system has completed it objective as the following:

7.4.1 Developed a Local Client-Server Application

PDS is a client server application implement to meet the requirement of company to store the information needed.

7.4.2 Managing all Resources and Processes

PDS is being developed to simplify the process of managing the company employees' medical record and information.

7.4.3 Generate Reliable Report

PDS was successfully to retrieve the information store in the database by the system and generate require report.

7.4.4 Secure System

PDS provide secure system where it blocked unauthorized user to access the system. On the other hand, all the password key in into the database has been encrypt to avoid hacker who steel login information from the database.

7.4.5 Customize Flexibility

The Employees Panel Doctor System is flexible to be customized to meet any other specification.

7.5 CONCLUSION

With the best understanding and knowledge of the system, the system can be said to be successfully achieved its objectives in meeting the functional and non-functional requirements specified during the initial stages of development.

PDS is a system that will ease the job of the administrator and enable company top management people to perform decision making with the support of the information generated by the system. Thus, it is a valuable product for company as it will save a lot of time in the consideration of medical expenses. Besides, the system also show its capability to store the company information in a more systematic form.

USER MANUAL

SERVER HARDWARE REQUIREMENT

For the Server machine, PDS is recommended to be setup with the computer with the support of Intel Pentium III 700 MHz and at least 256MB RAM. The Server is also required a IDE disk space of at least 3GB. The others setting will be same with the client machine setting.

CLIENT HARDWARE REQUIREMENT

PDS is recommended to be installed on the machine with the Intel Pentium II 500 MHz or above and with the support of 64MB RAM. The hard disk capacity recommended is 8.0GB. For the other accessories such as Display card, keyboard and mouse are all based on normal PC set configuration. Besides, network connection are also required in this system to be able the client to connect to the Server Database.

SERVER SOFTWARE REQUIREMENT

Server machine need to be run with the Windows NT 4.0 Server or the Windows 2000 Advanced Server. Besides, it will be prepared with the MS SQL Server 6.0 installed.

CLIENT SOFTWARE REQUIREMENT

Client machine will need a windows 9x operating system such as windows 98(recommended).

ACCESING THE PDS

To run the **PDS** system, there should be a default account which is administrator account created in the database. The database Server should be created with user login account to be able to access to the database. Then, the main system shall be installed at the client machine while the server machine is setting up with the database. Before starting the system, the vendors should be called to be setting and configure the connection between the client and the server as well as the database setting.

After all the configuration had finished, the users can start the system by:

- Ω Clicking to the **PDS** icon at the Program files folder on the start menu bar.
- Ω Getting Login to the **PDS** by entering the default User ID and the default password.

TWO LEVELS OF USERS IN THE PDS

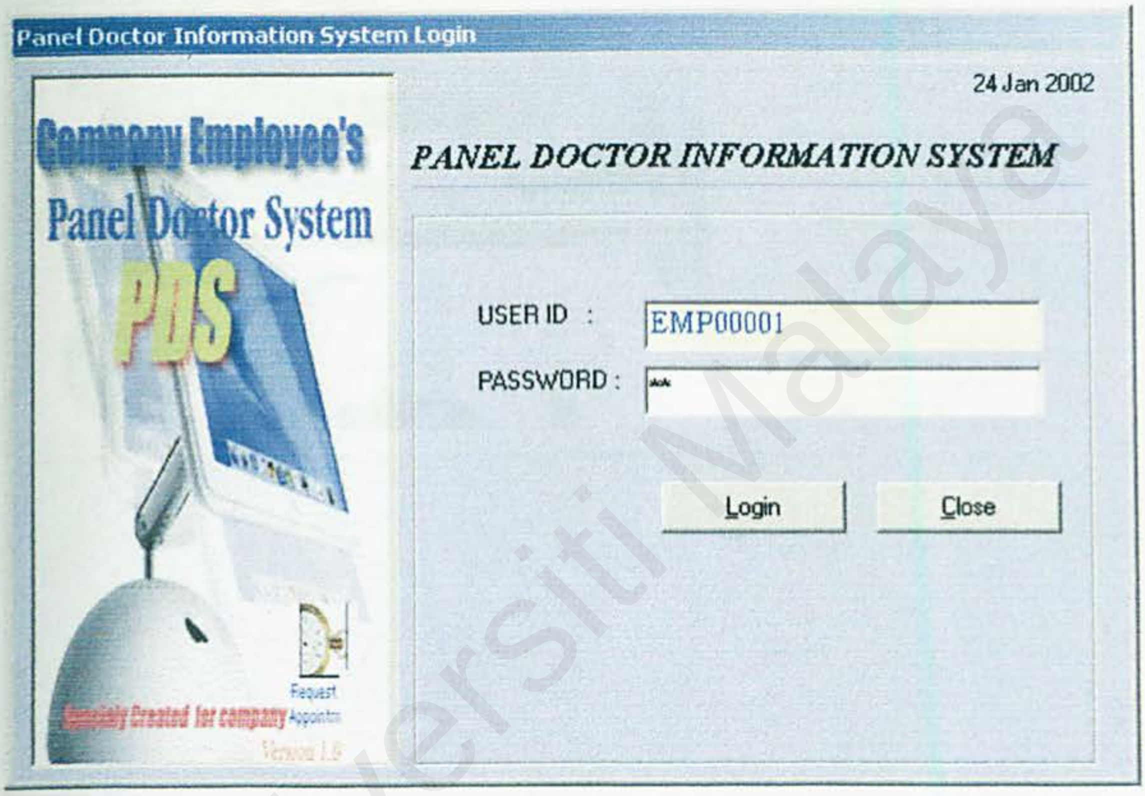
Because there are two different levels of users in the **PDS,** thus he user manual will be divide into two part which is:

- Administrators Manual
- Normal Users Manual

SECTION 1: PDS SYSTEM ADMINISTRATOR MANUAL

1.1 Main Login

To Start Using the system, user has to get Log In from the interface below.



Panel Doctor Information System Login

24 Jan 2002

PANEL DOCTOR INFORMATION SYSTEM

USER ID : EMP00001

PASSWORD : **

Figure 8.1 Panel Doctor System Main Login

To Start Login:

Ω Enter the User ID.

Ω Type in the User Password.

Ω Press "Login ", to get connected with the database and login.

Ω Press "Close", the pop up menu will be shown to ask user whether she/ he want to cancel the process.

1.2 PDS Main Program Menu

After user login successfully at the previous module, they will go to the page as shown below:

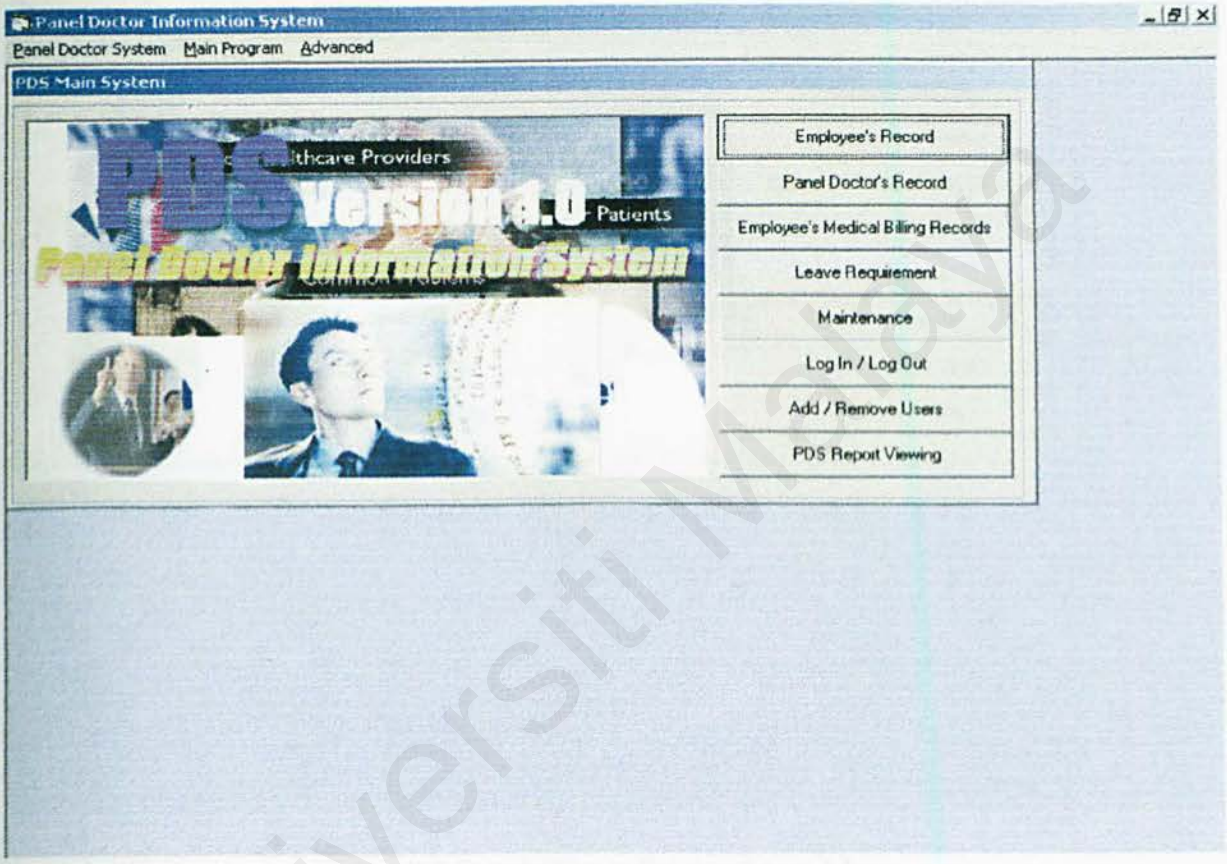


Figure 8.2 Panel Doctor System Main Menu

Select The Option:

Above is the Employees Panel Doctor System Main Menu. User may select their option at this page by simple click on the text button as shown.

For Example:

Ω When user click on the Employees Record, the Employees Record processing page will be pop up on the screen.

Ω When user wants to change to other transaction, they may simple click on the other button available on the main menu.

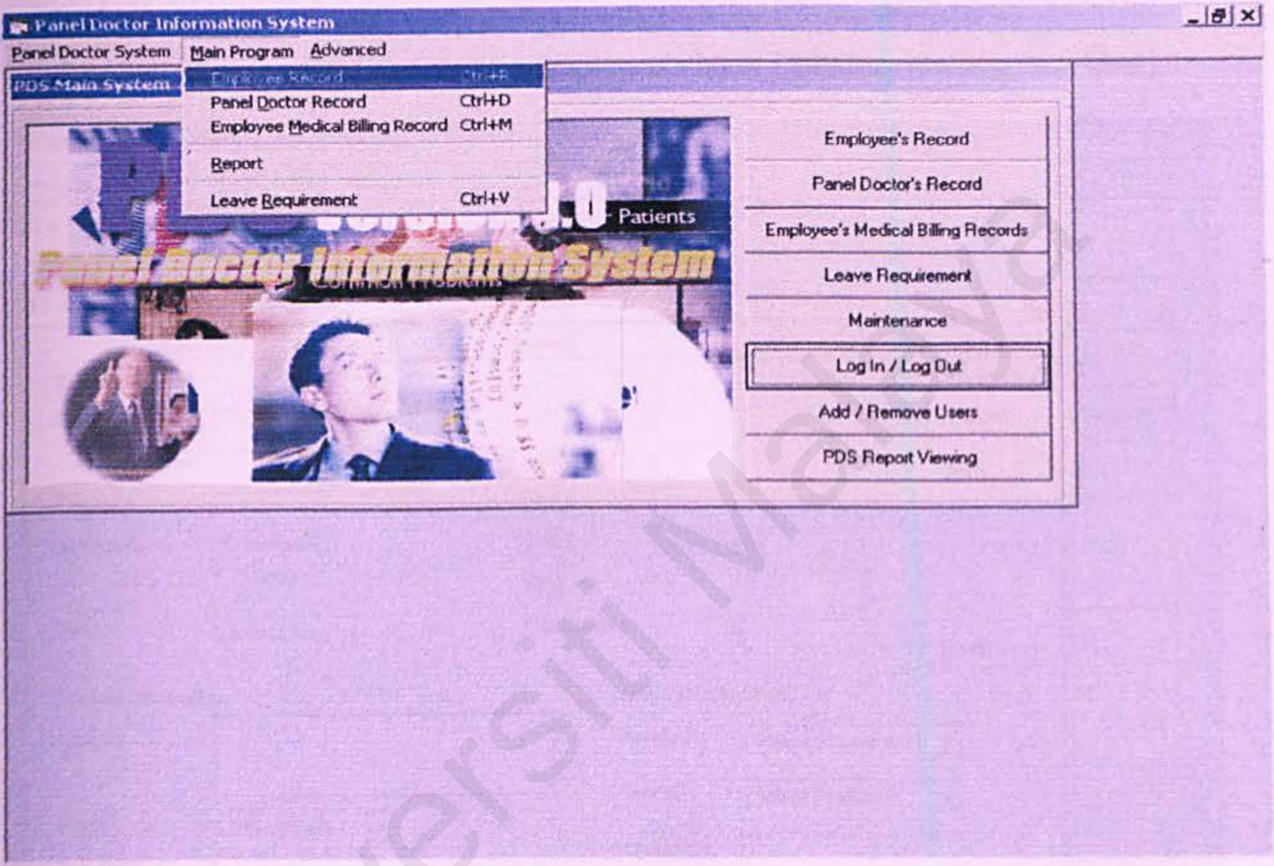


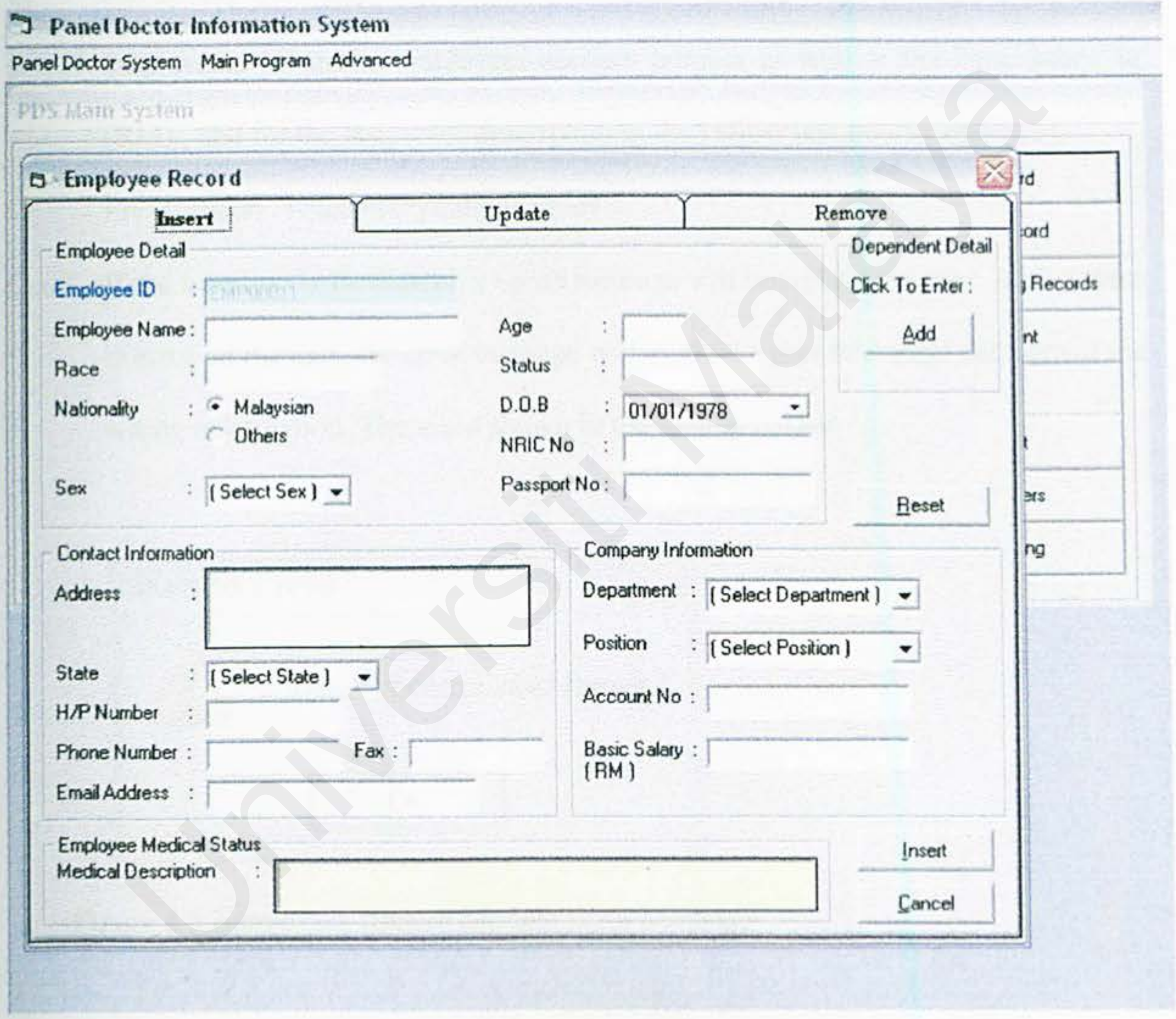
Figure 8.3 Panel Doctor System Main Menu

Ω Beside on the main menu, user may able to select the module from the menu bar on the top border of the system as above.

Ω There are shortcut key prepared for the advanced computer user where they will find faster and easier to press the short cut key. The short cut keys in the PDS are building due to the consideration of ease to remember. For example. To call out the Panel Doctor Record page. User just has to remember the "Doctor" and press Ctrl+D.

1.3 PDS Employees Record

Employees Record is a module for the user to insert a new employees record, update existing employee's record and remove employee's logo(Only for Administrator User).



Employee Detail

Employee ID : EMP00011

Employee Name :

Race :

Nationality : ● Malaysian ○ Others

Sex : (Select Sex)

Age :

Status :

D.O.B : 01/01/1978

NRIC No :

Passport No :

Dependent Detail

Click To Enter :

Contact Information

Address :

State : (Select State)

H/P Number :

Phone Number : Fax :

Email Address :

Company Information

Department : (Select Department)

Position : (Select Position)

Account No :

Basic Salary : (RM)

Employee Medical Status

Medical Description :

Figure 8.4 Panel Doctor System Employee Record module

To insert new Employees record:

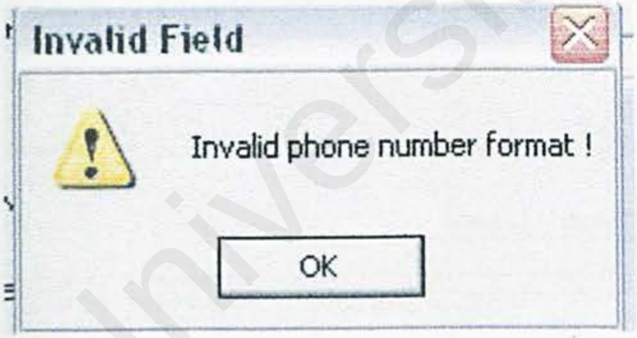
Ω Enter the employee's name, race, select the sex, enter age, status, date of birth, New IC No, passport number if exist.

Ω Enter the detail in the contact information section, key in the address, state, hand phone number, phone number, fax number (optional) email address (optional).

Ω Fill in the Company information's below, select department, select position of employees, fill in the employees account number as well as the basic salary in (RM)., and for the last, enter description in the yellow box below(optional).

Ω Press "Insert" when everything is ready.

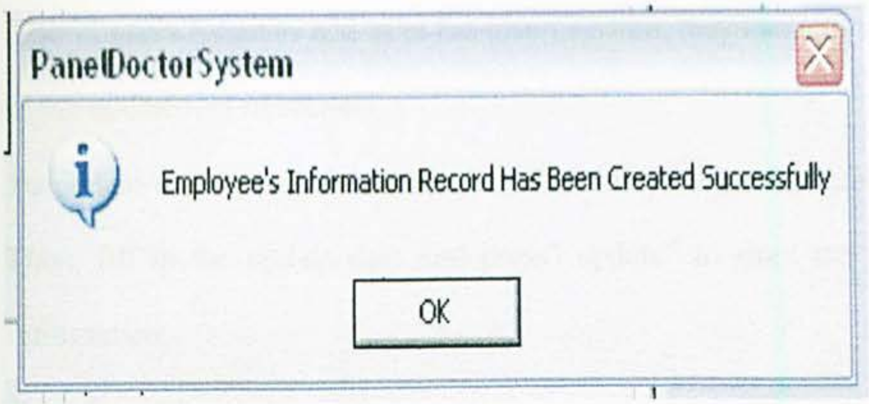
Ω If the insertion is successful, a op up message will be inform the user. And if there is error on the data, the error message will guide the user to change and correct the wrong information. These are shown in the picture below:



Invalid Field

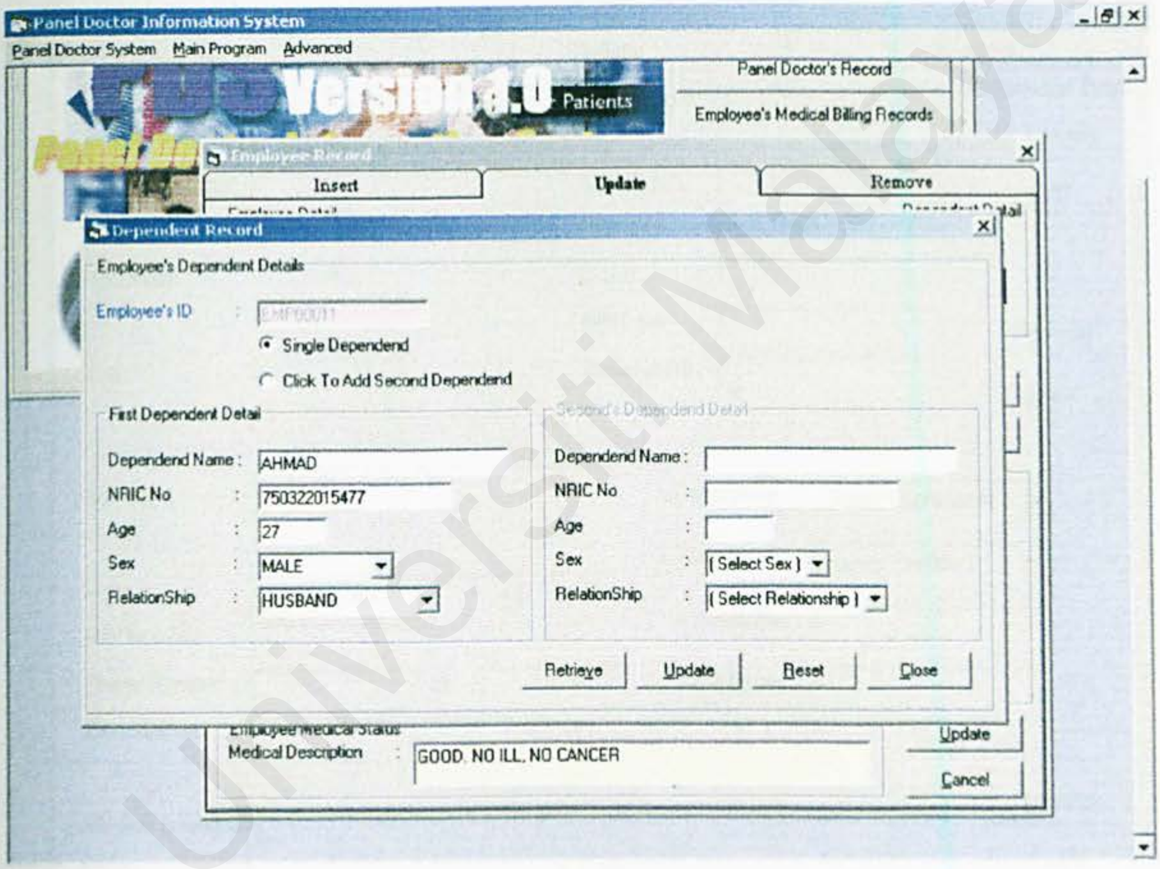
Invalid phone number format !

Figure 8.5 Employees Record Module Error Message



Employee's Information Record Has Been Created Successfully

Figure 8.6 Employees Record Module Insert Success Message



Employee's Dependent Details

Employee's ID : EMP00011

● Single Dependend

○ Click To Add Second Dependend

First Dependent Detail		Second's Dependend Detail	
Dependend Name :	AHMAD	Dependend Name :	
NRIC No :	750322015477	NRIC No :	
Age :	27	Age :	
Sex :	MALE	Sex :	[Select Sex]
RelationShip :	HUSBAND	RelationShip :	[Select Relationship]

Employee Medical Status

Medical Description : GOOD, NO ILL, NO CANCER

Figure 8.7 Panel Doctor System Dependent's insertion records module

Ω After the employees record, user has to insert the dependents record based on the employees ID.

Ω Else if, users forget to add in at the insert section, they can still update the record in the update record section.

Ω To update the dependent record, press "retrieve" to get the information retrieve. Then, fill in the update data and press" update" to store the employees latest information.

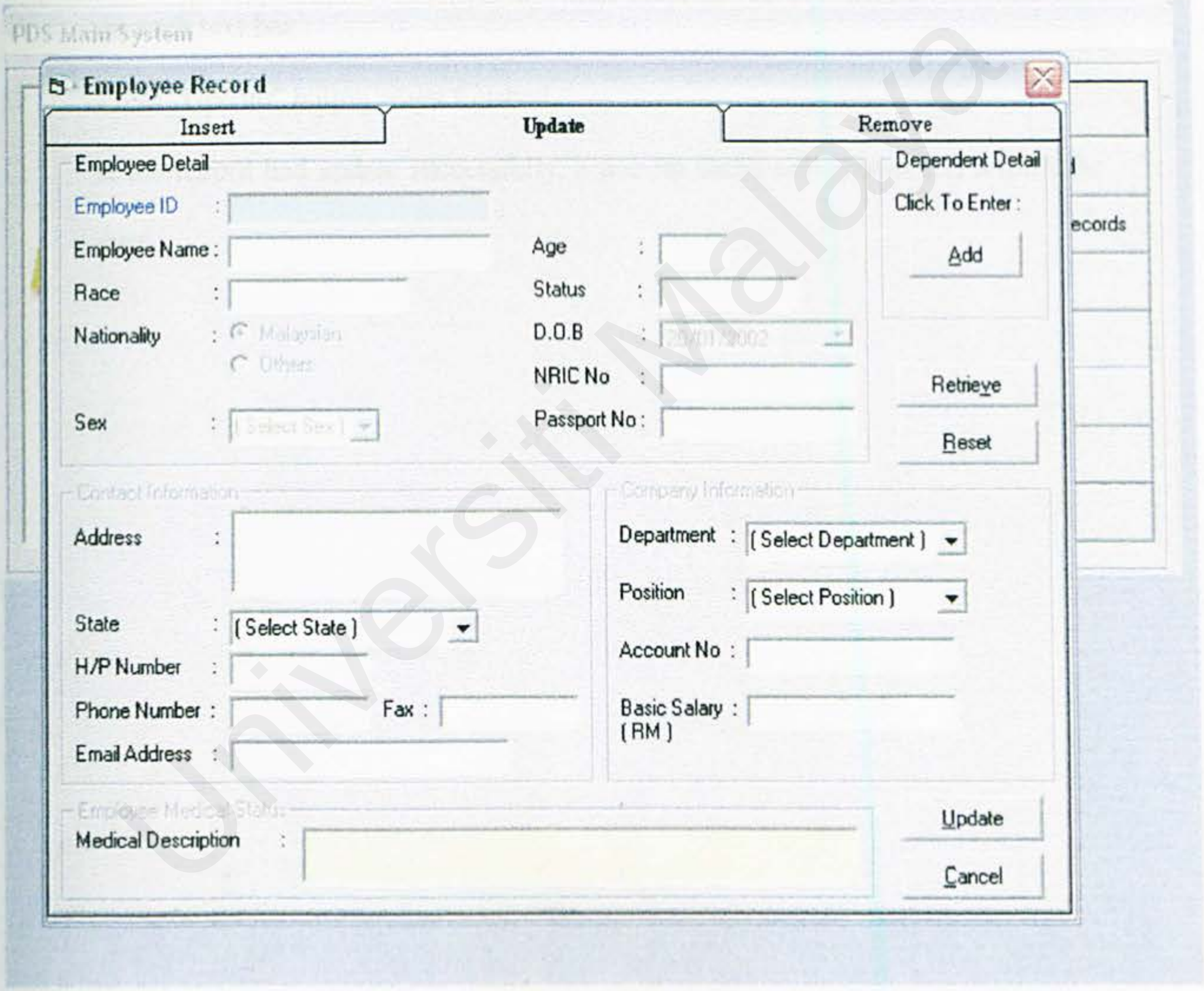


Figure 8.8 Employees Record Module update Module

To update a record:

Ω Enter the existing employees ID.

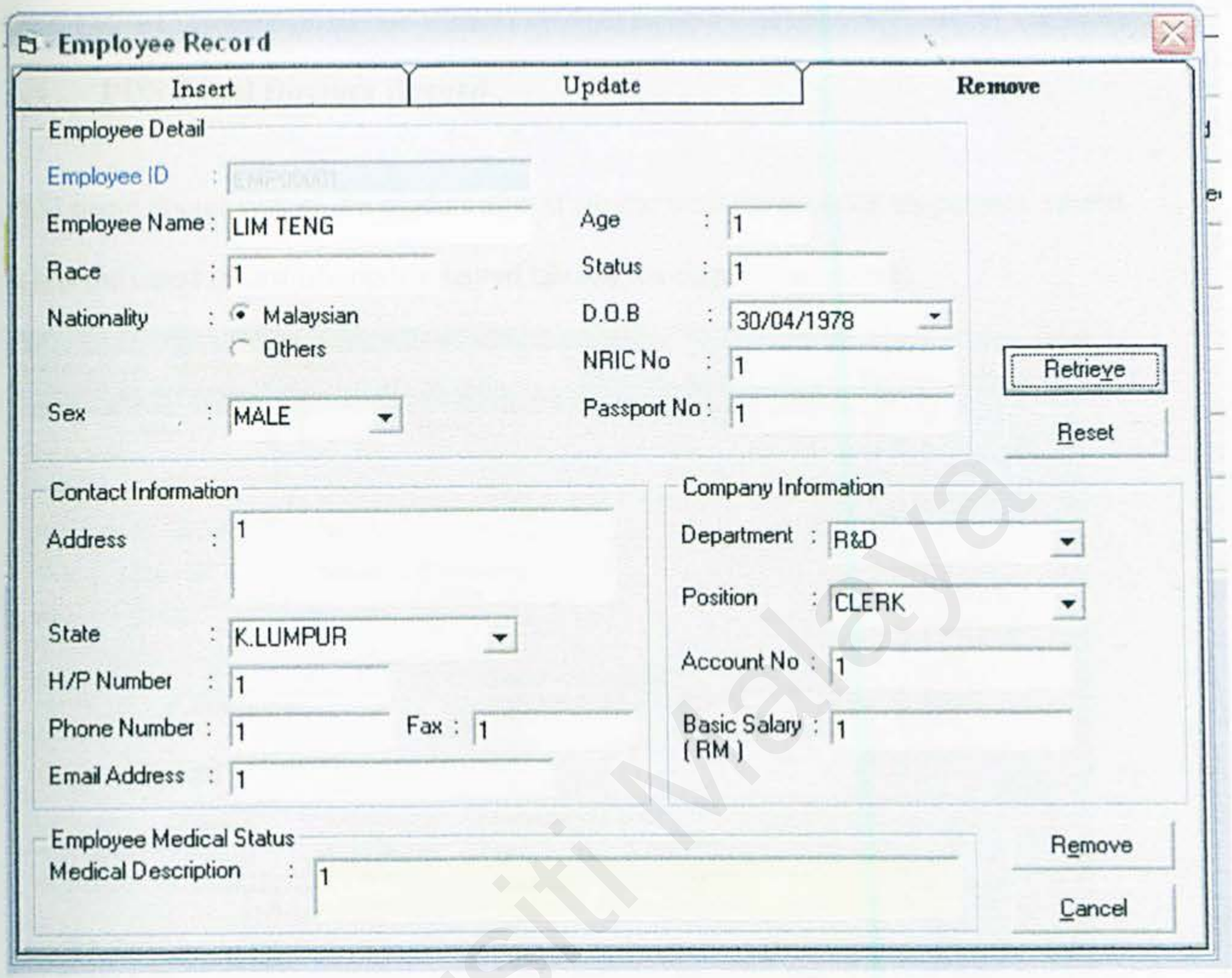
Ω Press the "Retrieve" button to get the information retrieve.

Ω If the record not found, try to key in another Employees ID.

Ω After the wanted data retrieved, user may update the record by fill in the new data on each text box.

Ω Press "Update" button when finished.

Ω If the record had update successfully, a pop up menu will shown and inform the user.



Employee Detail

Employee ID : EMP0001

Employee Name : LIM TENG

Race : 1

Nationality : ● Malaysian ○ Others

Sex : MALE

Age : 1

Status : 1

D.O.B : 30/04/1978

NRIC No : 1

Passport No : 1

Contact Information

Address : 1

State : K.LUMPUR

H/P Number : 1

Phone Number : 1 Fax : 1

Email Address : 1

Company Information

Department : R&D

Position : CLERK

Account No : 1

Basic Salary (RM) : 1

Employee Medical Status

Medical Description : 1

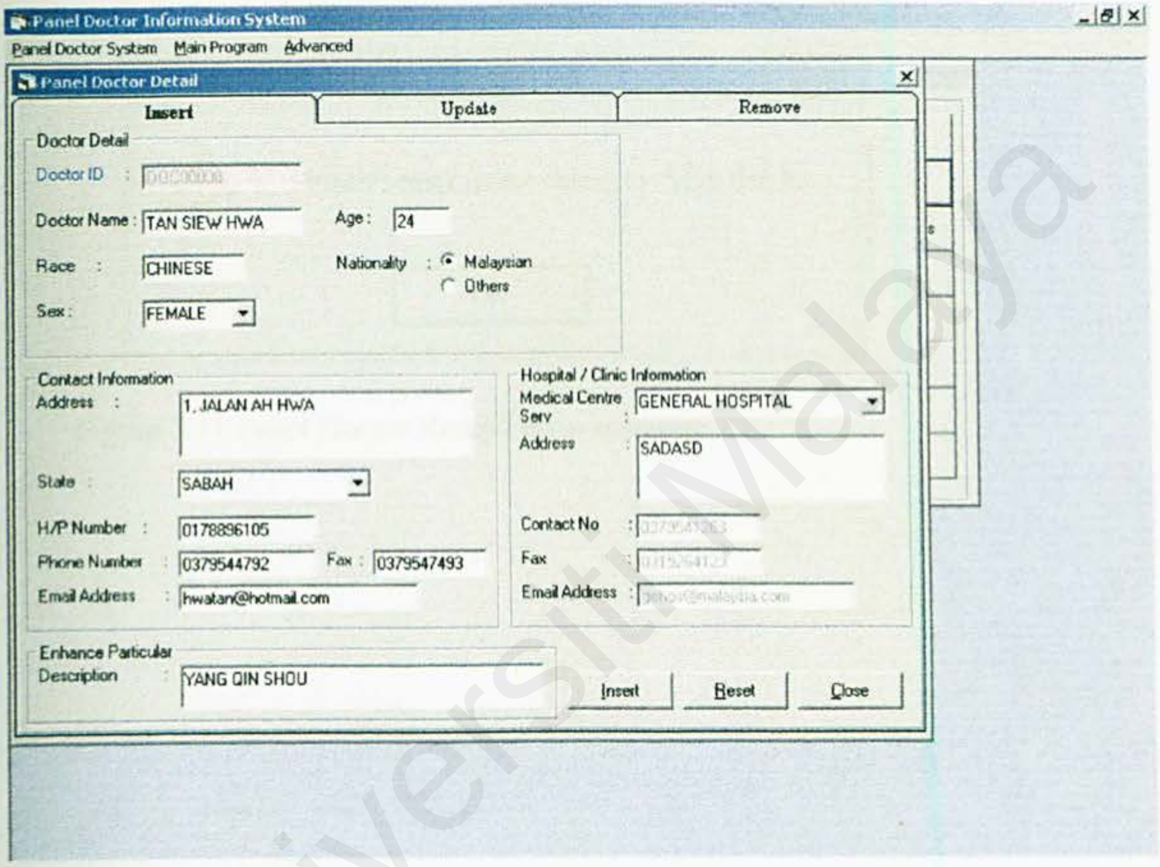
Figure 8.9 Employees Record Module Remove Module

To remove a record from the database:

- Ω Key in a employees ID in the Employees ID column.
- Ω Press" Retrieve" to retrieve particular record.
- Ω Press"Remove" to delete the record from the database.
- Ω When an employees record deleted, the dependents record attach to it will also be deleted.

1.4 PDS Panel Doctors Record

PDS panel doctor system is a module almost similar with the previous employees' record. It has the insert record tab, update record tab and the remove record tab.



Panel Doctor Information System

Panel Doctor System | Main Program | Advanced

Panel Doctor Detail

Insert | Update | Remove

Doctor Detail

Doctor ID : DOC00008

Doctor Name : TAN SIEW HWA

Age : 24

Race : CHINESE

Nationality : ● Malaysian ○ Others

Sex : FEMALE

Contact Information

Address : 1, JALAN AH HWA

State : SABAH

H/P Number : 0178896105

Phone Number : 0379544792

Fax : 0379547493

Email Address : hwatan@hotmail.com

Hospital / Clinic Information

Medical Centre Serv : GENERAL HOSPITAL

Address : SADASD

Contact No : [illegible]

Fax : [illegible]

Email Address : [illegible]

Enhance Particular

Description : YANG QIN SHOU

Figure 8.10 Panel Doctor Record insertions Module

To Insert the Doctor record:

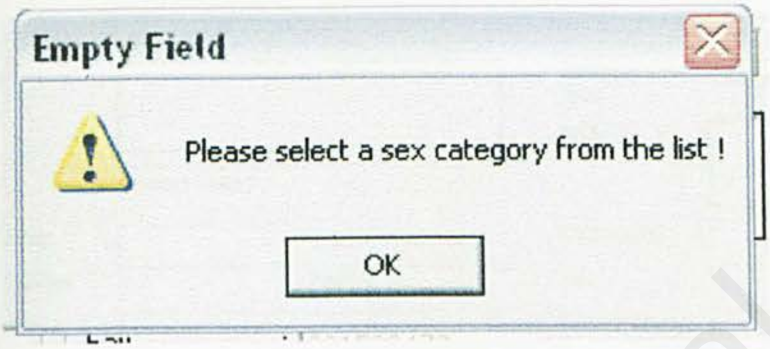
Ω Enter the doctor's name, race, select the sex, enter age, status,

Ω Enter the detail in the contact information section, key in the address, state, hand phone number, phone number, fax number (optional) email address (optional).

Ω Fill in the Medical center information's below, select hospital / clinic,

Ω Press "Insert" when everything is ready.

Ω If the insertion is successful, a op up message will be inform the user. And if there is error on the data, the error message will guide the user to change and correct the wrong information as the insert employees data do. These are shown in the picture below:



Empty Field

Please select a sex category from the list !

Figure 8.11 Panel Doctor Record error message

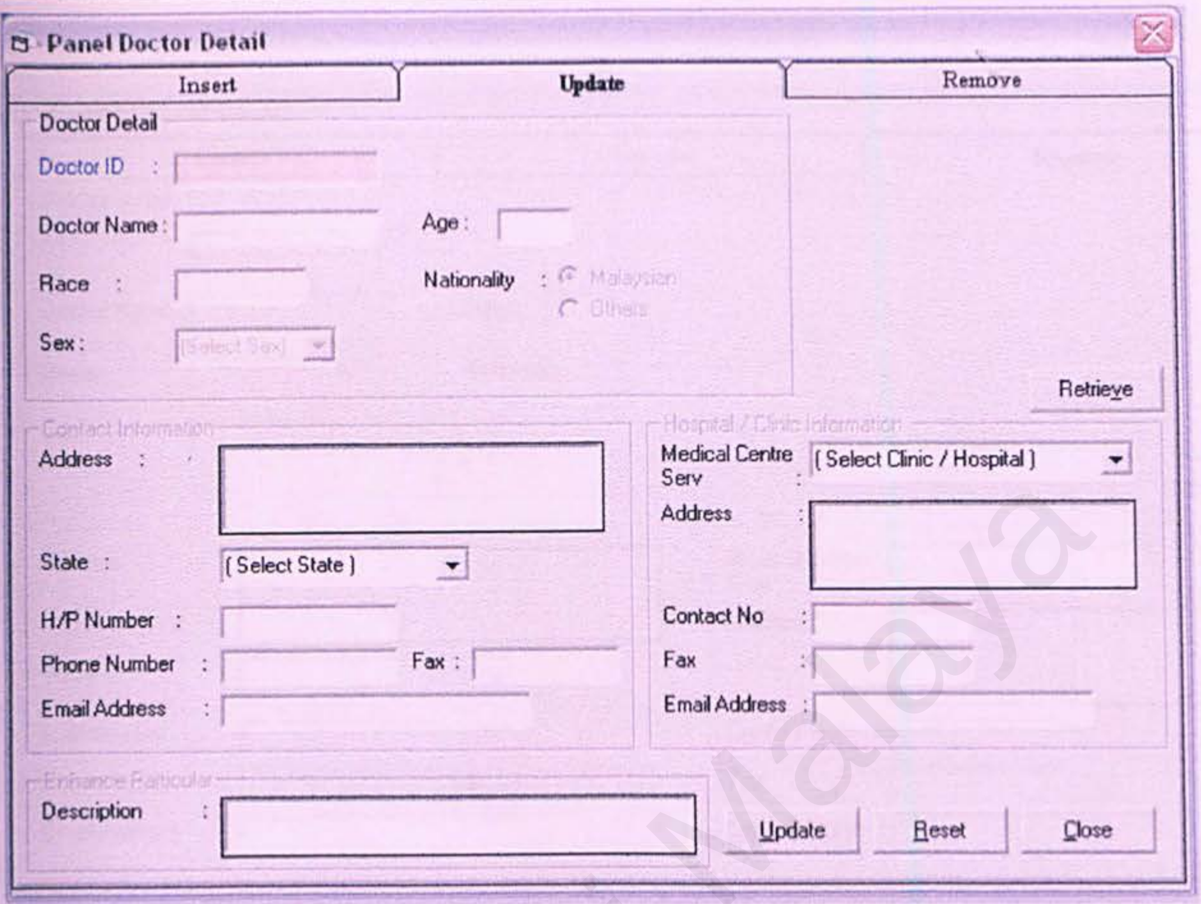


Figure 8.12 Panel Doctor Record Update module

To update the doctor record here:

- Ω Enter the existing Doctor's ID.
- Ω Press the "Retrieve" button to get the information retrieve.
- Ω If the record not found, try to key in another Doctor's ID.
- Ω After the wanted data retrieved, user may update the record by fill in the new data on each text box.
- Ω Press "Update" button when finished.
- Ω If the record had update successfully, a pop up menu will shown and inform the user.

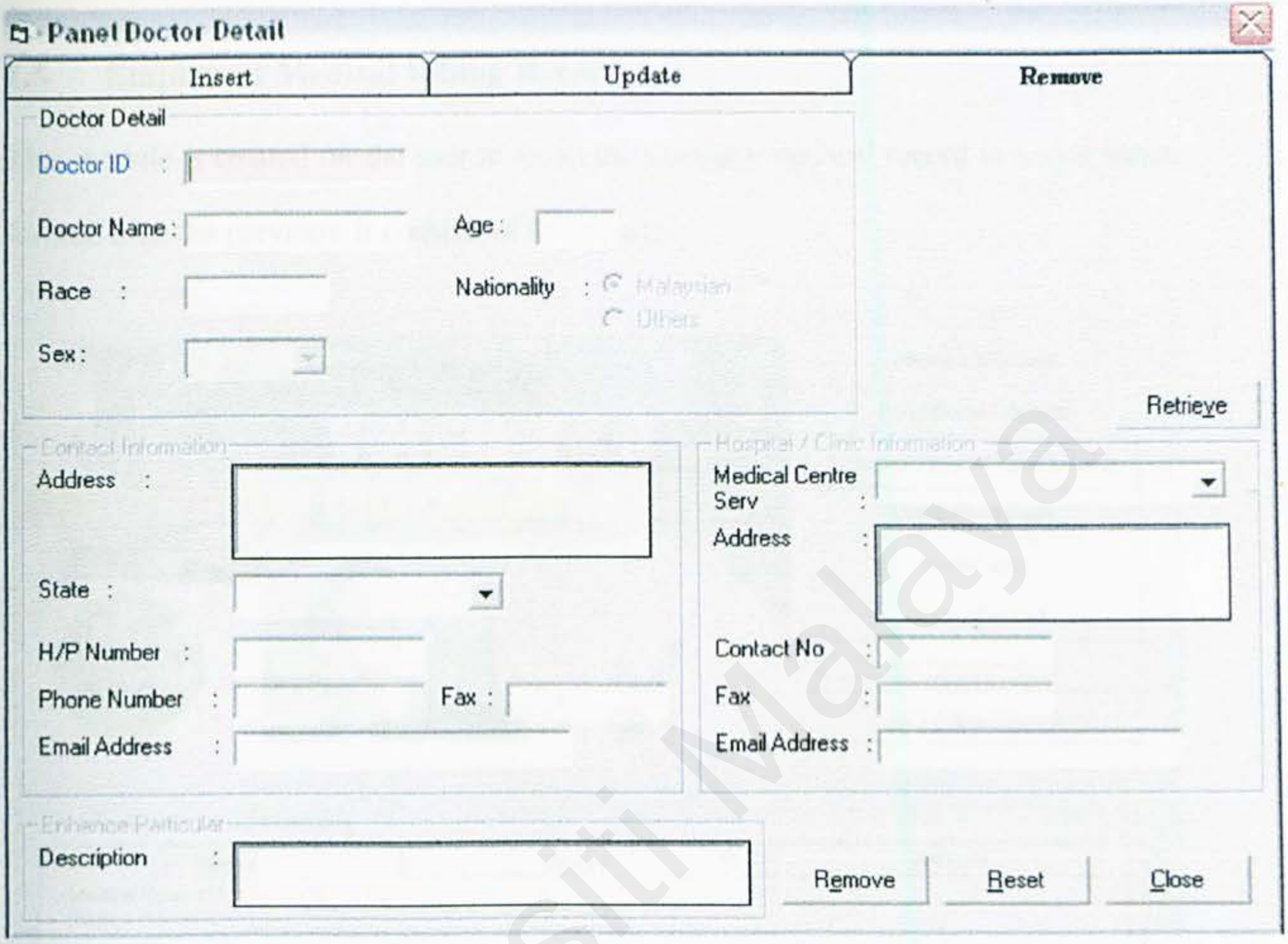


Figure 8.13 Panel Doctor Record Remove module

To delete particular doctor record from the database:

Ω Key in a doctor ID in the doctor ID column.

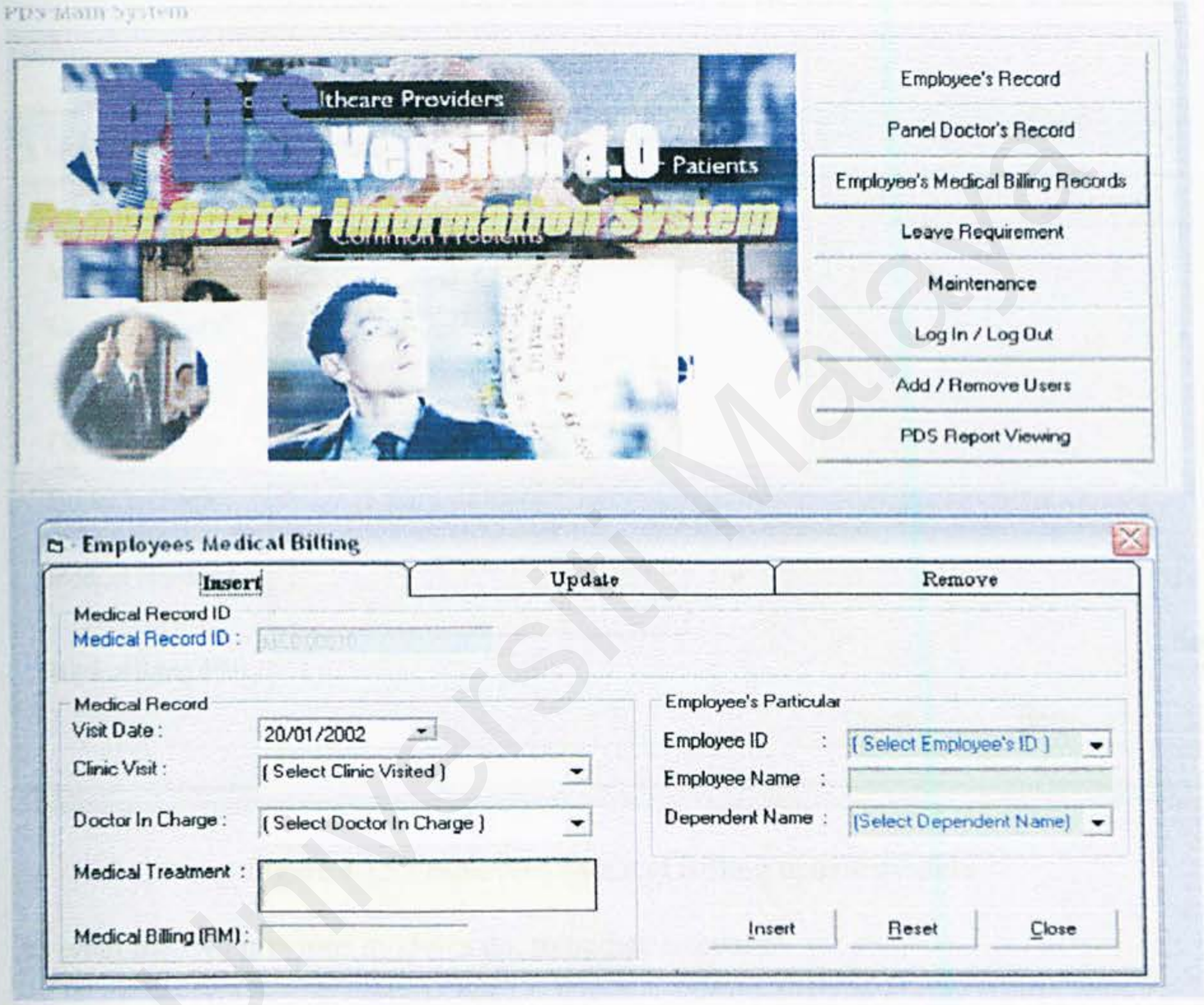
Ω Press" Retrieve" to retrieve particular record.

Ω Press"Remove" to delete the record from the database.

Ω If the deletion is successful, a message will be pop up and inform the user.

1.5 Employees Medical Billing Record

This module is created for the user to insert the company medical record in a summarize format. Like the previous. It consists of three parts.



Medical Record ID
Medical Record ID : MED00010

Medical Record
Visit Date : 20/01/2002
Clinic Visit : (Select Clinic Visited)
Doctor In Charge : (Select Doctor In Charge)
Medical Treatment :
Medical Billing (RM) :

Employee's Particular
Employee ID : (Select Employee's ID)
Employee Name :
Dependent Name : (Select Dependent Name)

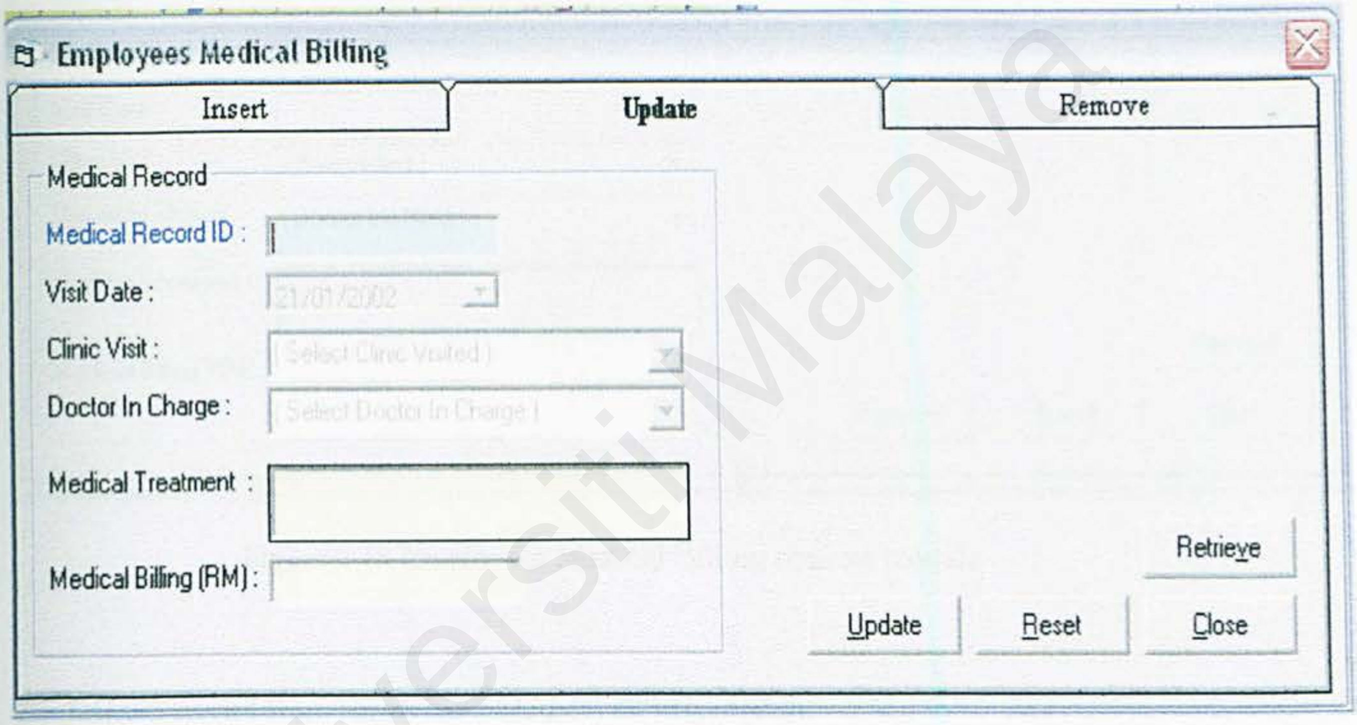
Figure 8.14 Employees Medical Billing Insert module

To insert a new medical record:

Ω Enter the Visit date, clinic visit to (selection), doctor in charge(selection), enter Medical treatment(optional), medical billing in RM format, Ω Select the employees id who responsible to the record.

Ω Press "Insert" when everything is ready.

Ω If the insertion is successful, a op up message will be inform the user. And if there is error on the data, the error message will guide the user to change and correct the wrong information.



Employees Medical Billing

Medical Record

Medical Record ID :

Visit Date : 21/01/2002

Clinic Visit : (Select Clinic Visited)

Doctor In Charge : (Select Doctor In Charge)

Medical Treatment :

Medical Billing (RM) :

Figure 8.15 Employees Medical Billing update module

As what the two previous modules do, to update a record:

Ω Enter the existing medical ID.

Ω Press the "Retrieve" button to get the information retrieve.

Ω If the record not found, try to key in another Medical ID.

Ω After the wanted data retrieved, user may update the record by fill in the new data on each text box.

Ω Press "Update" button when finished.

Ω If the record had update successfully, a pop up menu will shown and inform the user.

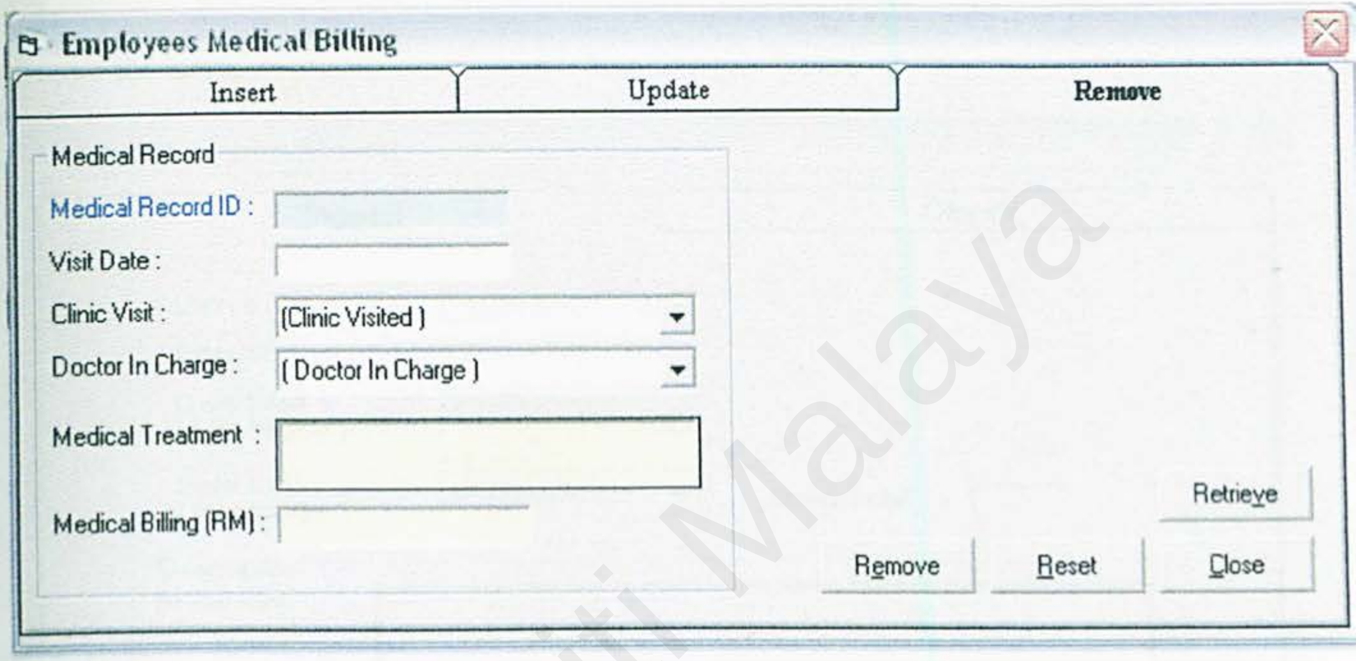


Figure 8.16 Employees Medical Billing remove module

To delete a record from Employees Medical Billing module

Ω Key in a medical record ID in the Medical ID column.

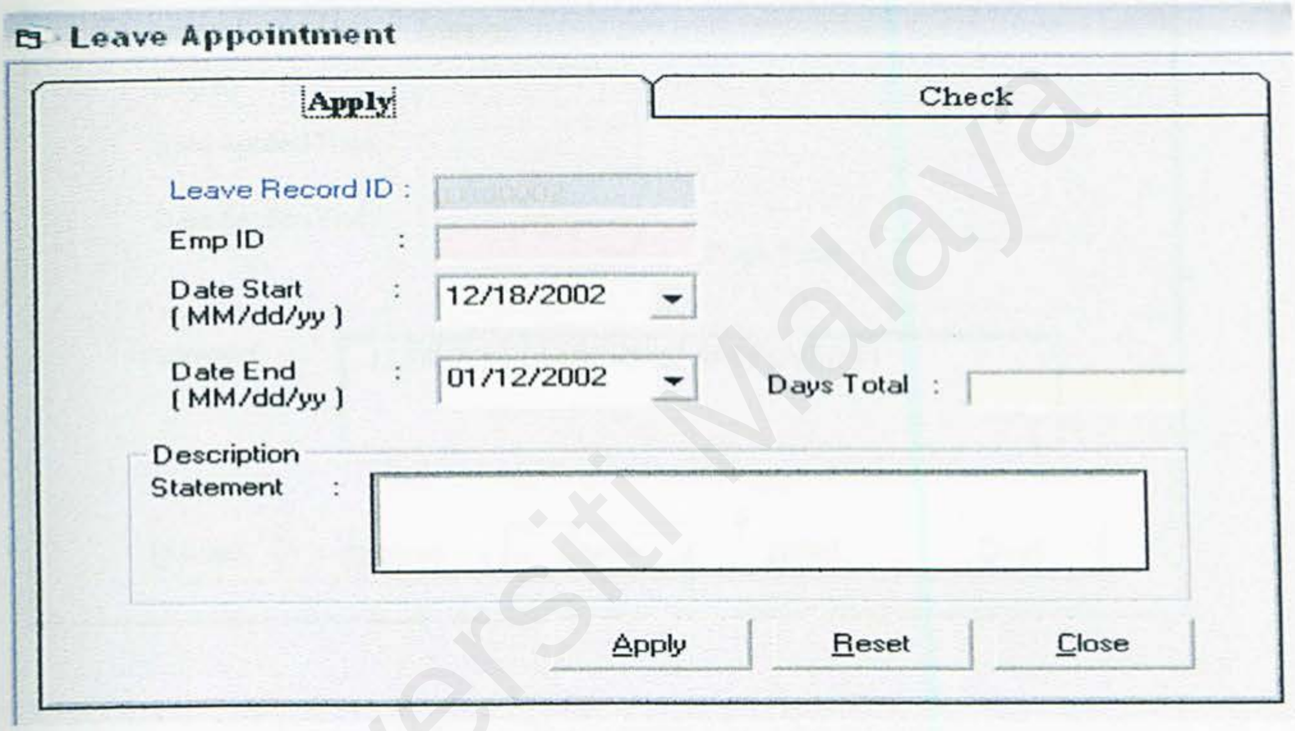
Ω Press" Retrieve" to retrieve particular record.

Ω Press" Remove" to delete the record from the database.

Ω f the deletion is successful, a message will be pop up and inform the user.

1.6 Leave Requirement Record

In leave requirement, user will key in requirement form whether it is approve or not into the database and the administrator user can reject the leave, approved the leave and discard the leave.



Leave Record ID :

Emp ID :

Date Start (MM/dd/yy) : 12/18/2002

Date End (MM/dd/yy) : 01/12/2002

Days Total :

Description Statement :

Figure 8.17 Leave Requirement Apply modules

To start the leave requirement application:

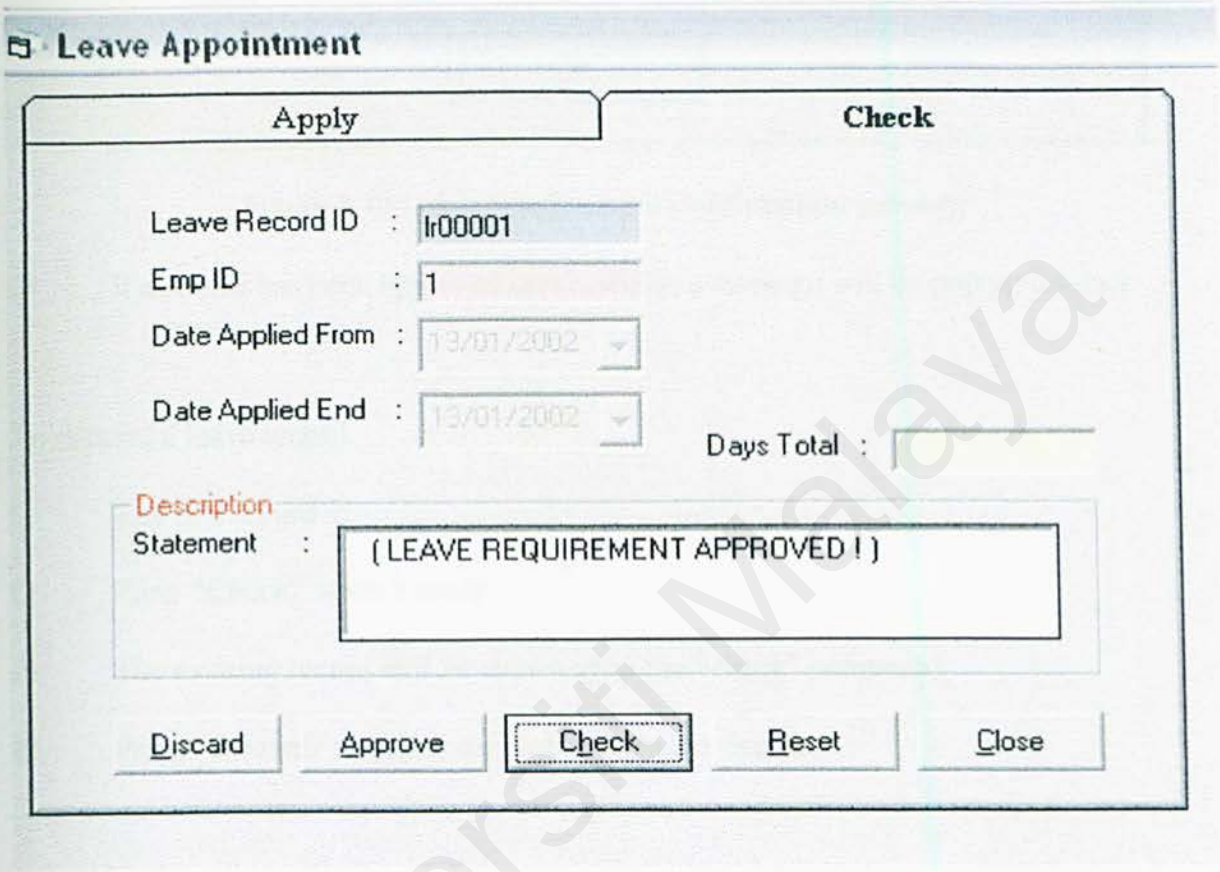
Ω Enter the employee id who is taking the leave requirement form.

Ω Fill in the date start of the leave application and the end date.

Ω Enter description if any.

Ω If everything is ready, Press" Apply" key to insert the record into the database.

The leave appointment check module is set to let the administrator level user to approve the leave application and to discard the application.



Leave Record ID : lr00001

Emp ID : 1

Date Applied From : 13/01/2002

Date Applied End : 13/01/2002

Days Total :

Description

Statement : (LEAVE REQUIREMENT APPROVED !)

Figure 8.18 Leave Requirement Checking modules

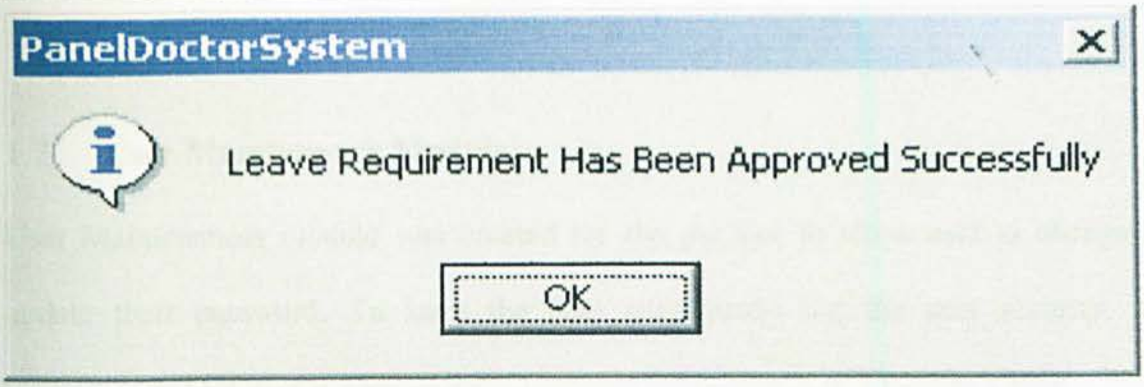
To approve a leave record:

Ω Key in a record id.

Ω Press "Check" when's ready.

Ω The existing record will be shown after the "check" command.

Ω Press "Approve" to approve a leave record



Leave Requirement Has Been Approved Successfully

Figure 8.19 Leave Requirement Confirmation message

Ω If a record has been approved successfully, a message will be pop up for user.

To discard a leave record:

Ω Key in a record id.

Ω Press "Check" when's ready.

Ω The existing record will be shown after the "check" command.

Ω Press" Discard" to delete the record from the database.

1.7 User Maintenance Module

User Maintenances module was created for the purpose to allow user to change and update their password. To keep the data consistency and the user security, it is recommended for user to change their password every month or 3 months a time.

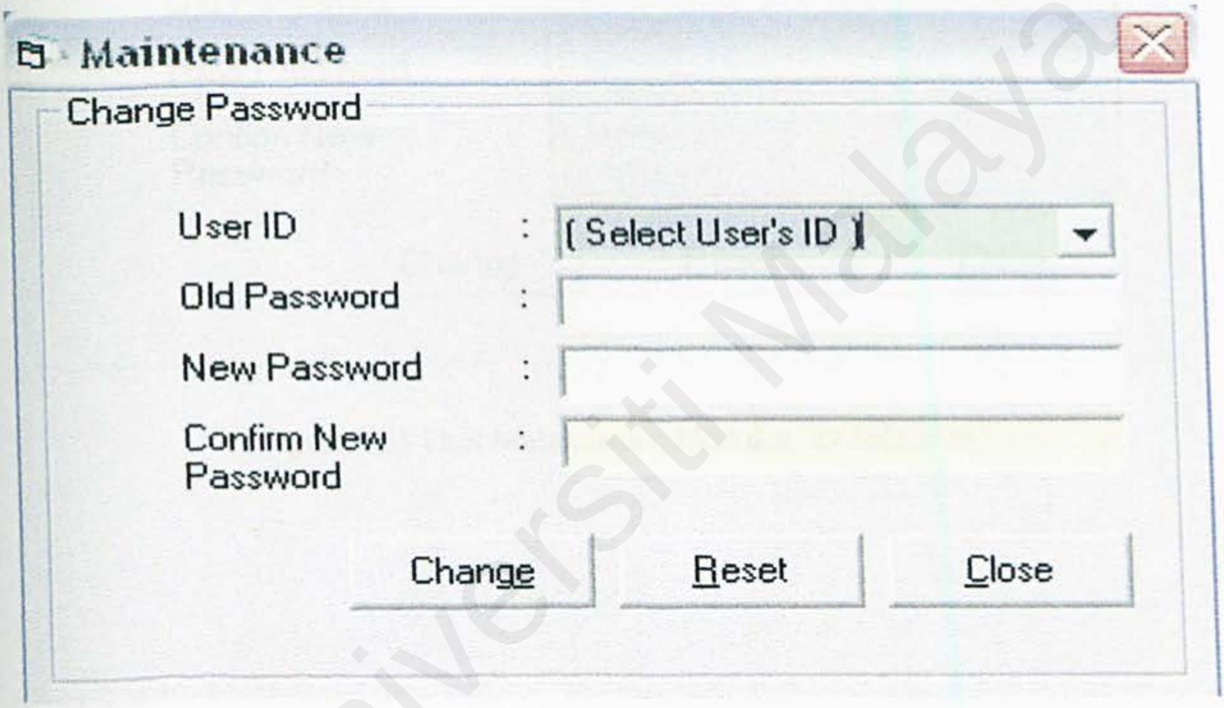


Figure 8.20 User Maintenance Modules

To change User Password:

Ω Select the existing User ID from the User ID selection box.(as shown in Figure 8.19 User Maintenance Modules, selection ID)

Ω Key in the old password.

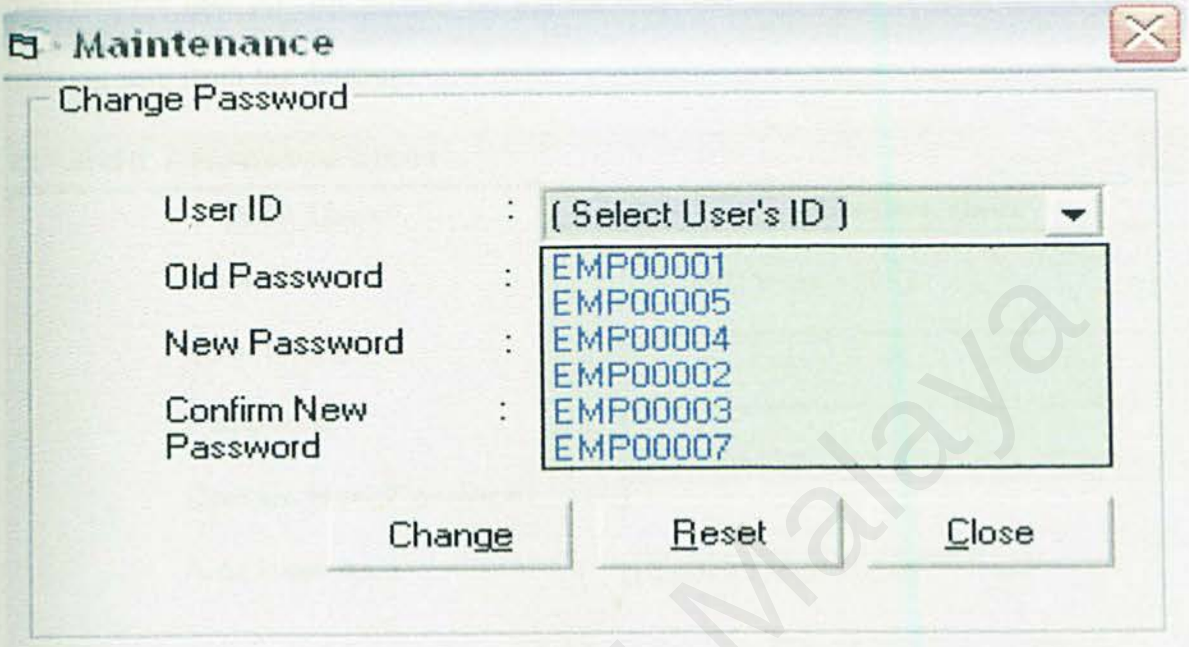
Ω Fill in the user new password.

Ω Confirm the new password by entering the password again.

Ω Press" Change" to change user password.

Ω Press" Reset" to reset the record.

Ω Press" Close" to close the module and back to the main menu.



Maintenance

Change Password

User ID : (Select User's ID)

Old Password :

New Password :

Confirm New Password :

Figure 8.21 User Maintenance Modules, ID Selection

1.8 Add Remove User Module

Add remove user provide service for adding new user into the user database or remove existing user from the database

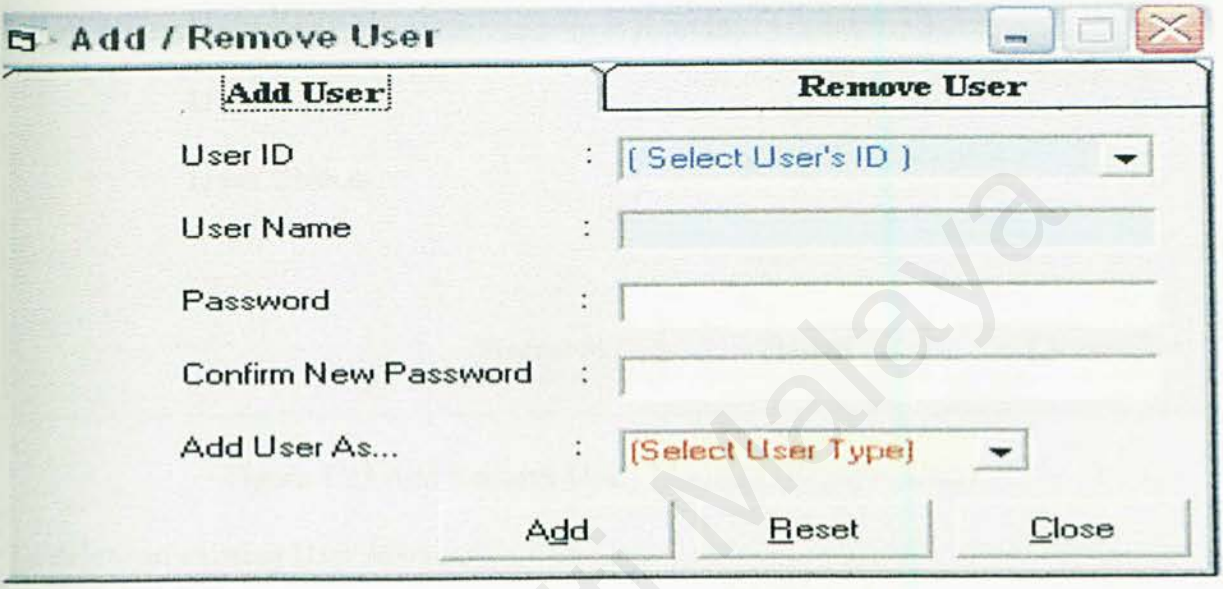
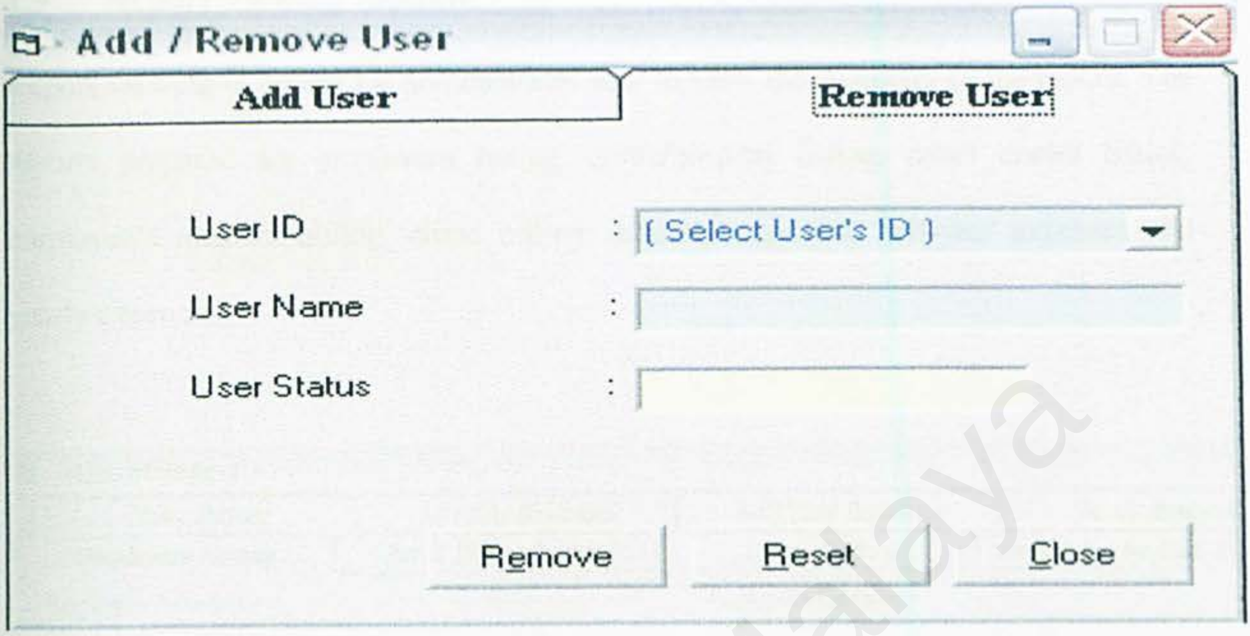


Figure 8.22 Add Remove Users Modules

To Add a New User:

- Ω Select User ID from the Selection box
- Ω Type in the user name.
- Ω Type in the user password.
- Ω Confirm the new password by retype it again in the prepared textbox.
- Ω Press" Add" to add the new account.
- Ω Press" Reset" to reset the record.
- Ω Press" Close" to close the module and back to the main module.



User ID : (Select User's ID)

User Name :

User Status :

Figure 8.23 Add Remove Users Modules (Remove User)

To delete an existing User Account:

Ω Select User ID from the Selection box

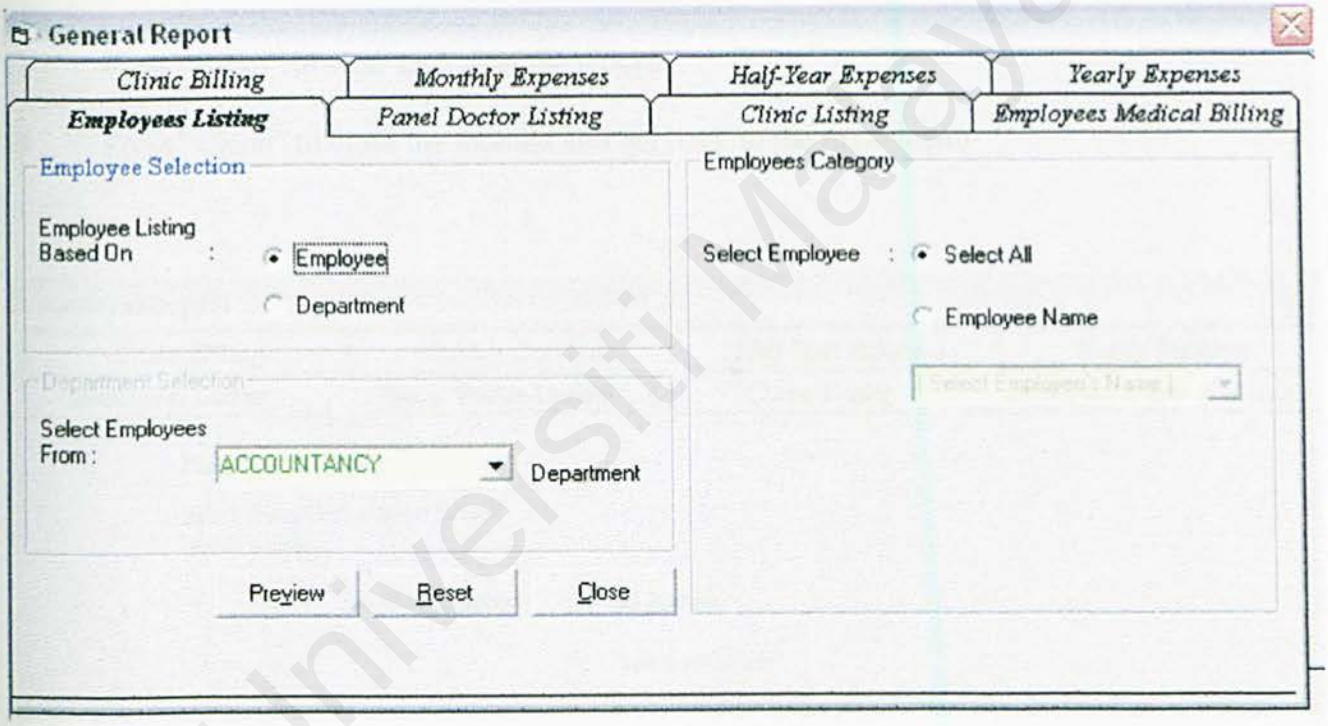
Ω Press" Remove" to delete the account.

Ω Press" Reset" to reset the record.

Ω Press" Close" to close the module and back to the main module.

1.9 Report Viewing Module

Report viewing is design for administrator user to view the summary of the record. The reports prepared are employees listing, clinic/hospital listing, panel doctor listing, employee's medical billing, clinic billing, monthly expenses, half-year expenses and yearly expenses.



General Report

Employee Selection

Employee Listing Based On : ● Employee ○ Department

Department Selection

Select Employees From : ACCOUNTANCY Department

Employees Category

Select Employee : ● Select All ○ Employee Name

[Select Employee's Name]

Figure 8.24 Report Viewing, Employee Listing

To View the employees listing report:

Ω Click on the Employees option button.

Ω Select employees, either select all or select single employee listing

Ω To select only one employee, click on the Employee Name option button and select employee's name from the list box.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu.

To view employees listing report based on department:

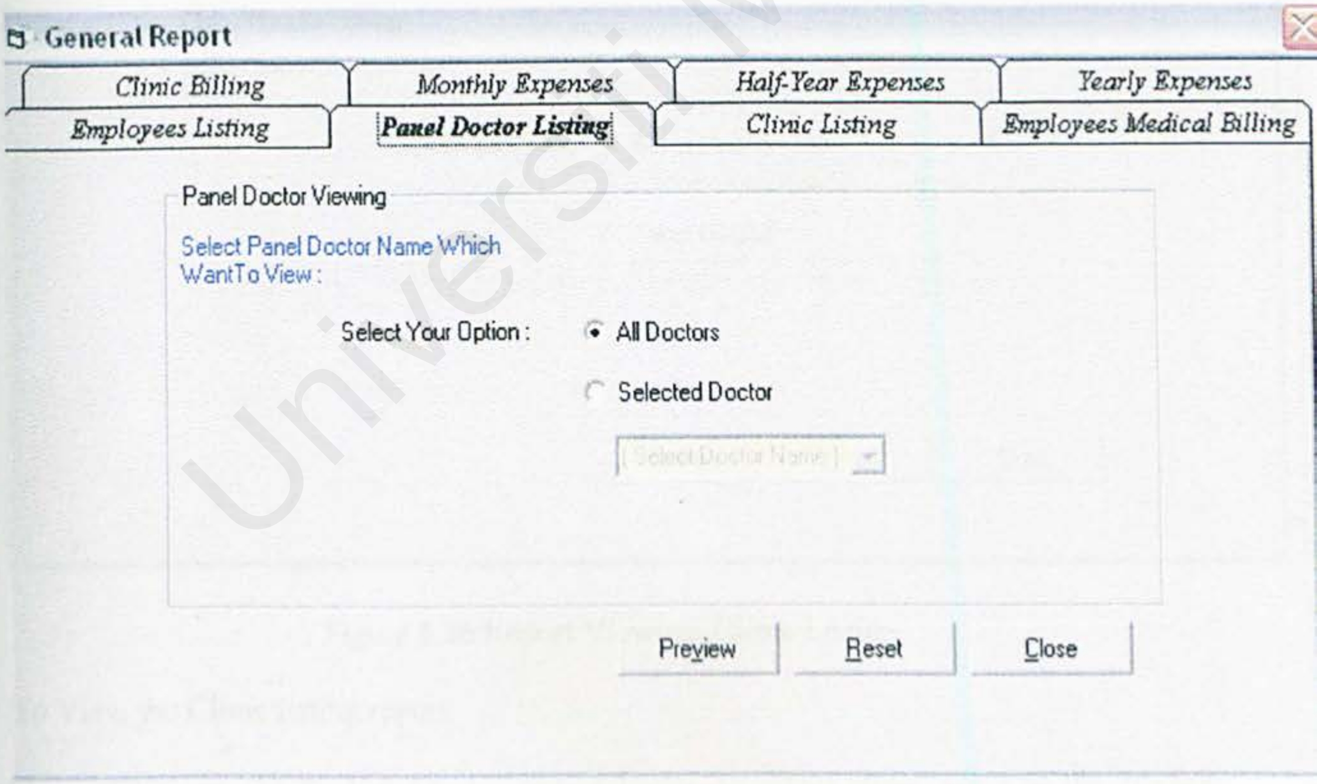
Ω Click on the Department option button.

Ω Select department from the list box.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu



Panel Doctor Viewing

Select Panel Doctor Name Which WantTo View :

Select Your Option : ● All Doctors

○ Selected Doctor

[Select Doctor Name]

Figure 8.25 Report Viewing, Panel Doctor Listing

To View the panel Doctor Listing report:

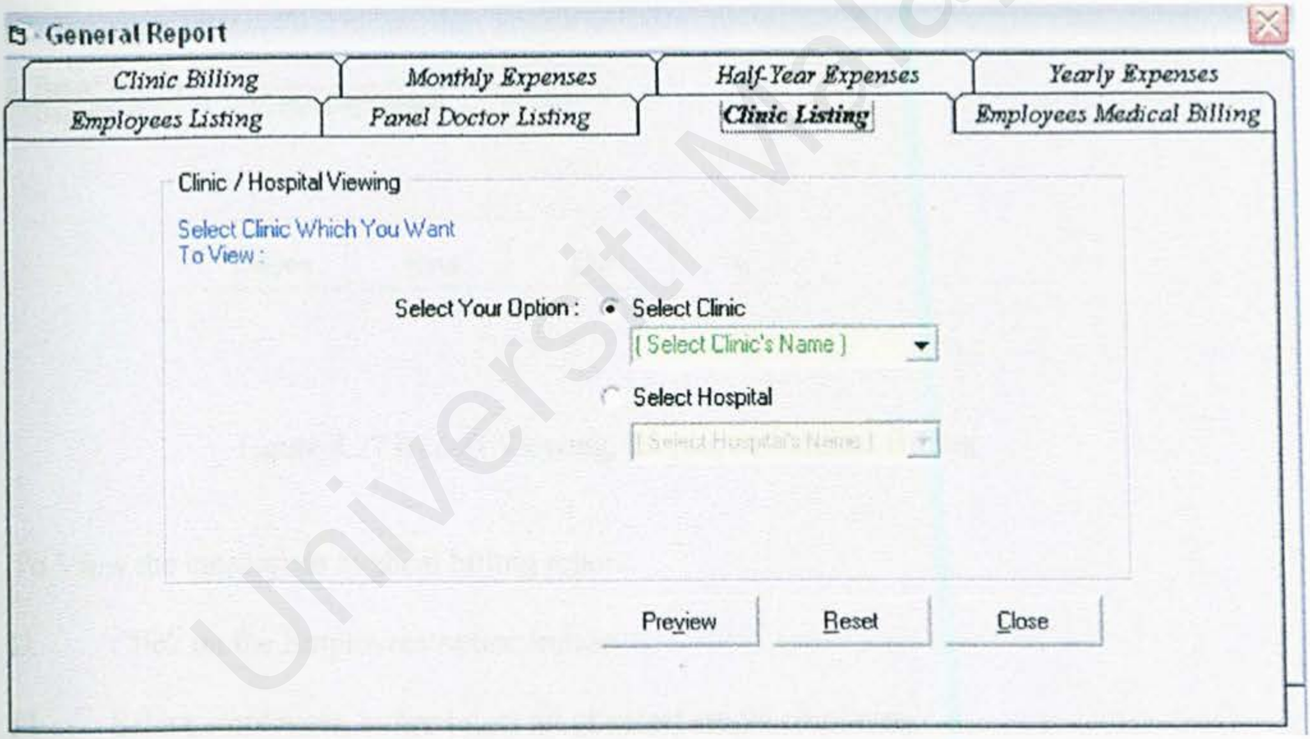
Ω Select the select all option button to view all doctor record or selected doctor to view selected doctor record.

Ω Select doctor's name from the list box.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu



General Report

Clinic / Hospital Viewing

Select Clinic Which You Want To View :

Select Your Option : ● Select Clinic

[Select Clinic's Name]

○ Select Hospital

[Select Hospital's Name]

Figure 8.26 Report Viewing, Clinic Listing

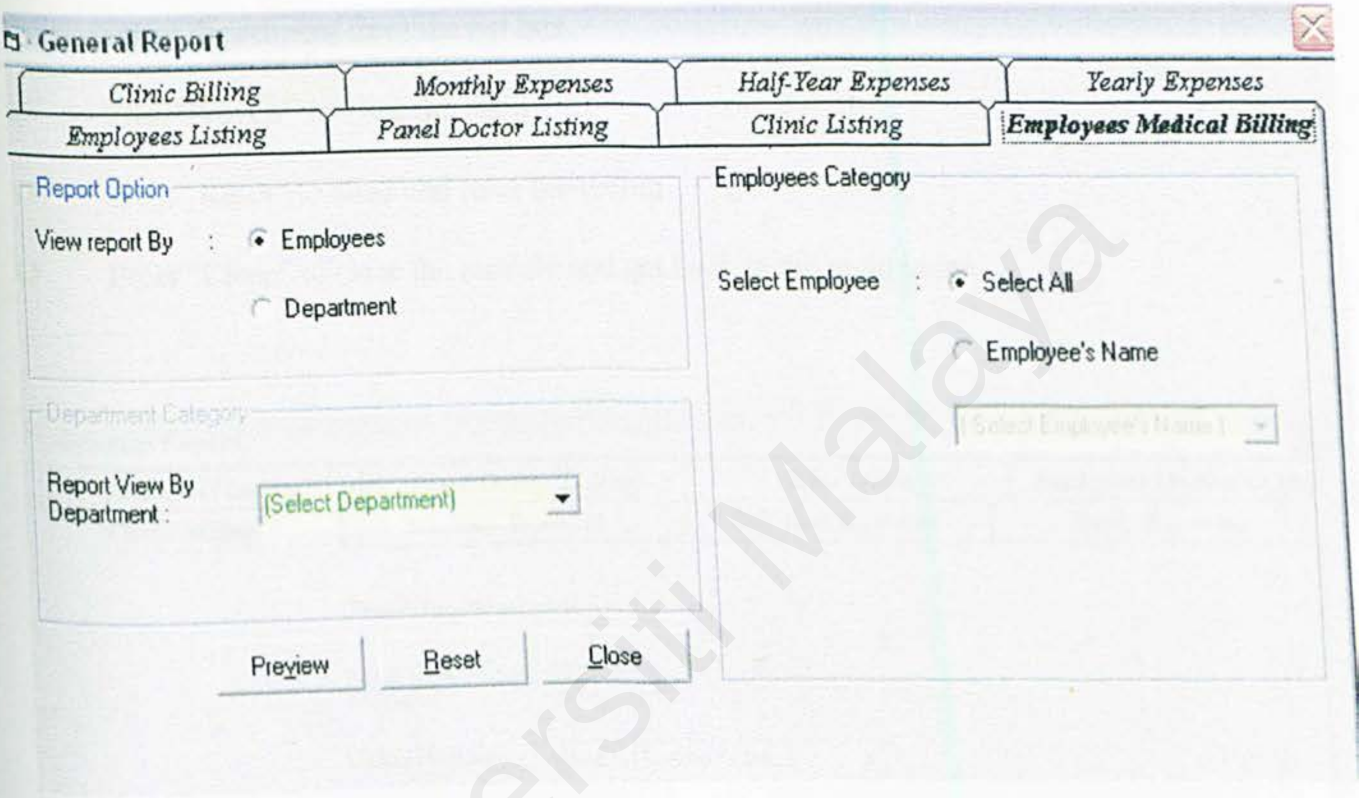
To View the Clinic listing report:

Ω Click on the "Select Clinic" option button to select clinic name or the "Select Hospital" option button to select hospital name..

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu



Report Option

View report By : ● Employees ○ Department

Department Category

Report View By Department : (Select Department)

Employees Category

Select Employee : ● Select All ○ Employee's Name

(Select Employee's Name)

Figure 8.27 Report Viewing, Employee Medical Billing

To View the employees medical billing report:

Ω Click on the Employees option button.

Ω Select employees, either select all or select single employee

Ω To select only one employee, click on the Employee Name option button and select employee's name from the list box.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu

To view employee's medical billing report based on department:

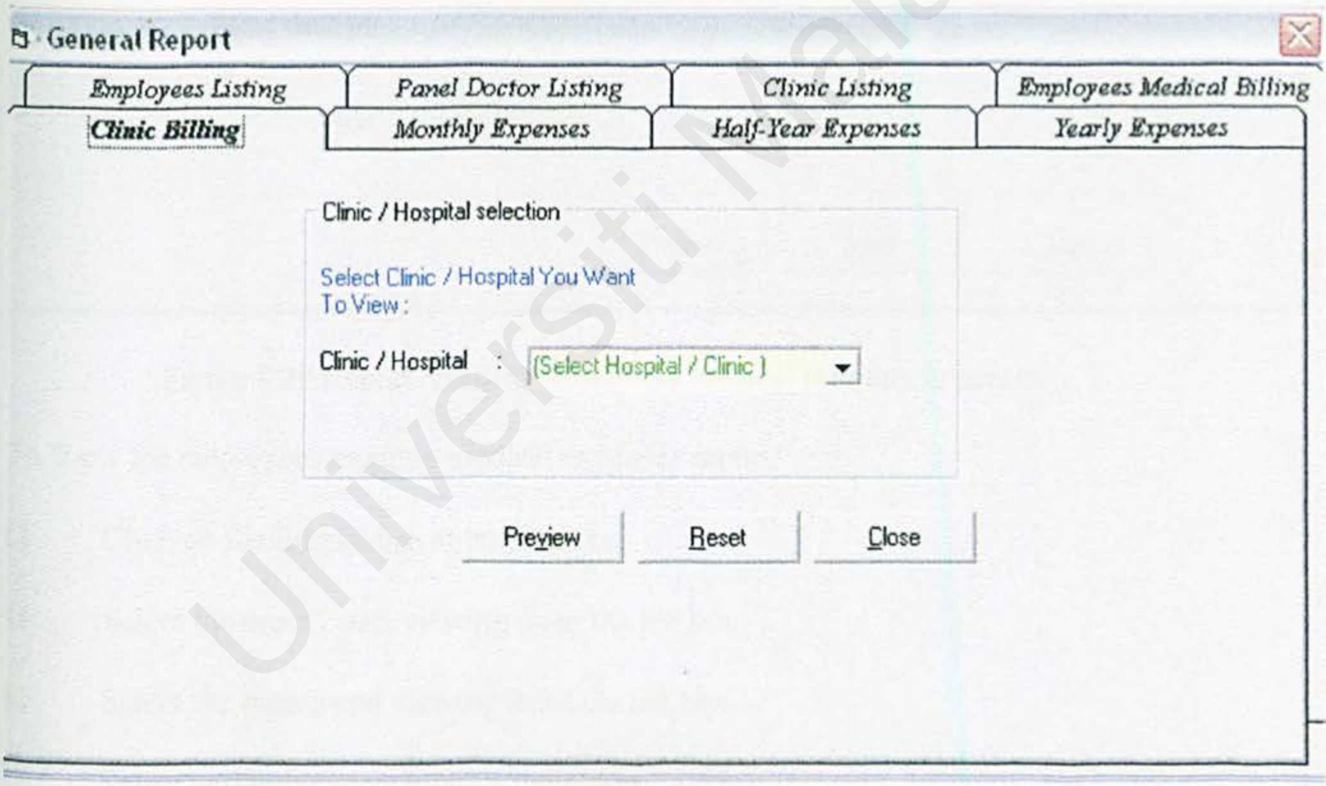
Ω Click on the Department option button.

Ω Select department from the list box.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu



Clinic / Hospital selection

Select Clinic / Hospital You Want To View :

Clinic / Hospital : (Select Hospital / Clinic)

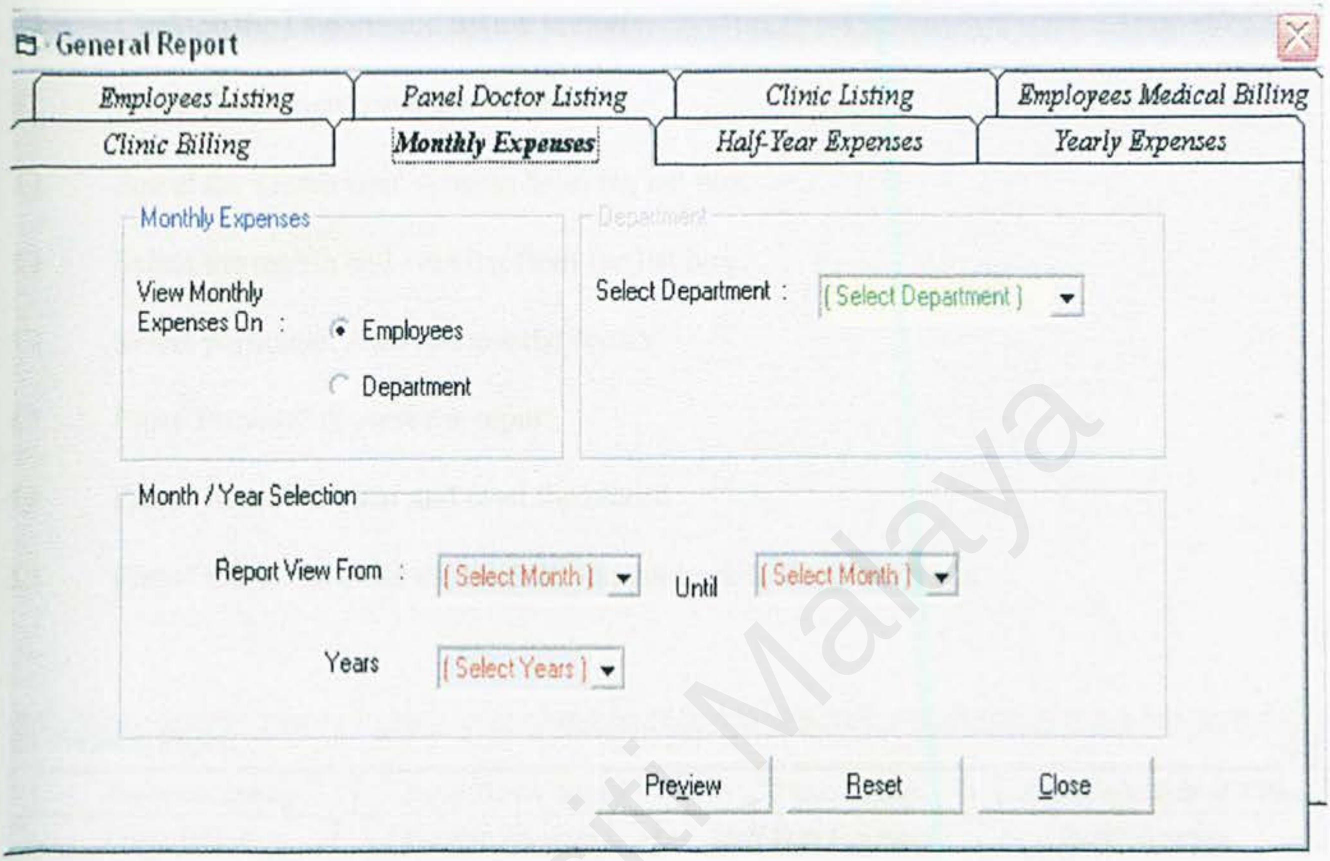
Figure 8.28 Report Viewing, Clinic Billing

To View the Clinic billing report:

Ω Click on the list box to select clinic name or hospital name.

Ω Press"Prevew" to view the report.

- Ω Press" Reset" to clear and reset the record.
- Ω Press" Close" to close the module and get back to the main menu



View Monthly Expenses On : ● Employees ○ Department

Select Department : (Select Department)

Report View From : (Select Month) Until (Select Month)

Years (Select Years)

Figure 8.29 Report Viewing, Company medical monthly expenses

To View the employees monthly medical expenses report:

- Ω Click on the Employees option button.
- Ω Select the month start viewing from the list box.
- Ω Select the month end viewing from the list box
- Ω Select particular years to view the report.
- Ω Press"Prevew" to view the report.
- Ω Press" Reset" to clear and reset the record.
- Ω Press" Close" to close the module and get back to the main menu.

To view employee's monthly medical expenses based on department:

Ω Click on the Department option button.

Ω Select department from the list box.

Ω Select the month start viewing from the list box.

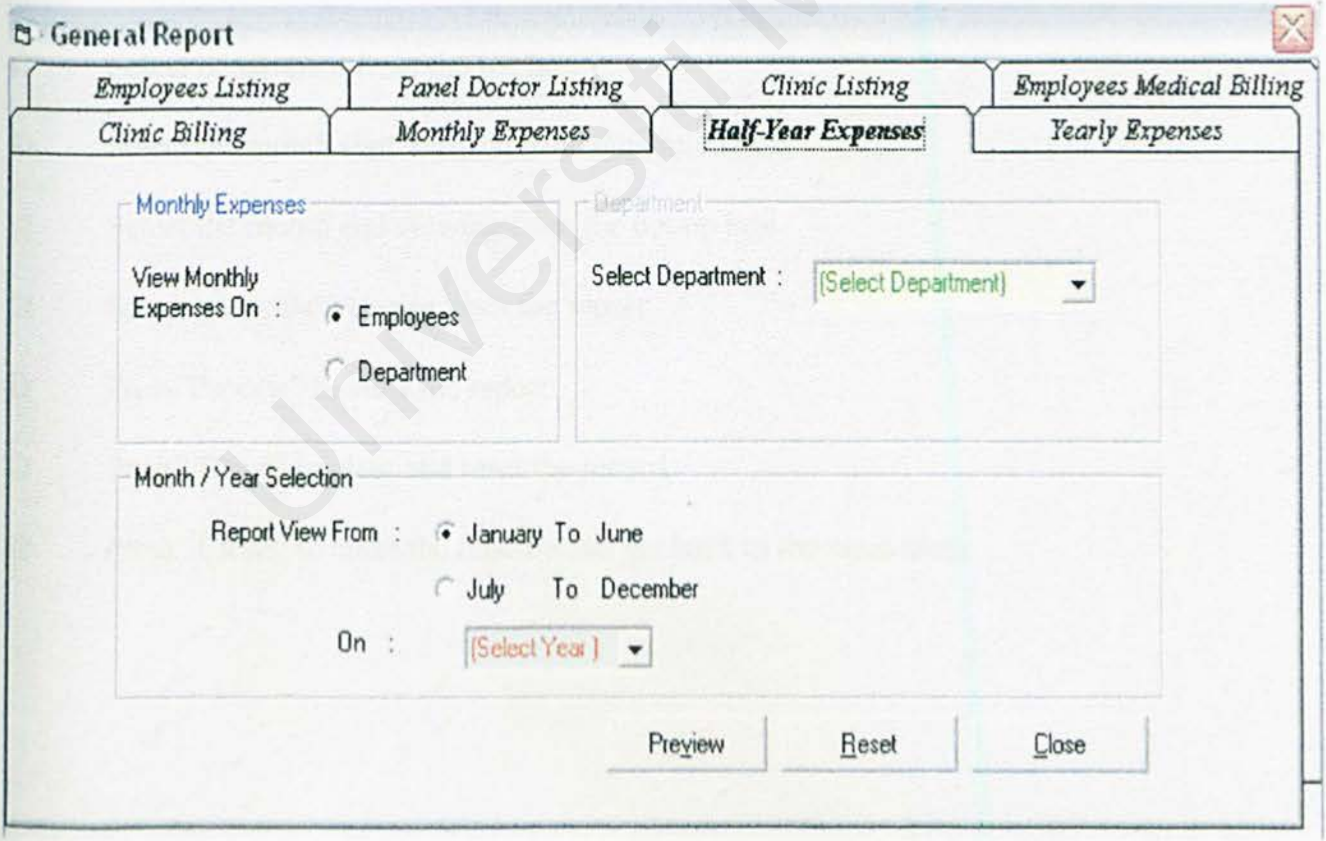
Ω Select the month end viewing from the list box

Ω Select particular years to view the report.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record

Ω Press" Close" to close the module and get back to the main menu



General Report

Tabs: Employees Listing | Panel Doctor Listing | Clinic Listing | Employees Medical Billing | Clinic Billing | Monthly Expenses | **Half-Year Expenses** | Yearly Expenses

Monthly Expenses

View Monthly Expenses On : ● Employees ○ Department

Department

Select Department : (Select Department)

Month / Year Selection

Report View From : ● January To June ○ July To December

On : (Select Year)

Figure 8.30 Report Viewing, Company medical half Year expenses

To View the employees monthly medical half year expenses report:

Ω Click on the Employees option button.

Ω Select the month start viewing from the option box

Ω Select the month end viewing from the option box

Ω Select particular years to view the report.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu.

To view employee's monthly medical half year expenses based on department:

Ω Click on the Department option button.

Ω Select department from the list box.

Ω Select the month start viewing from the option box.

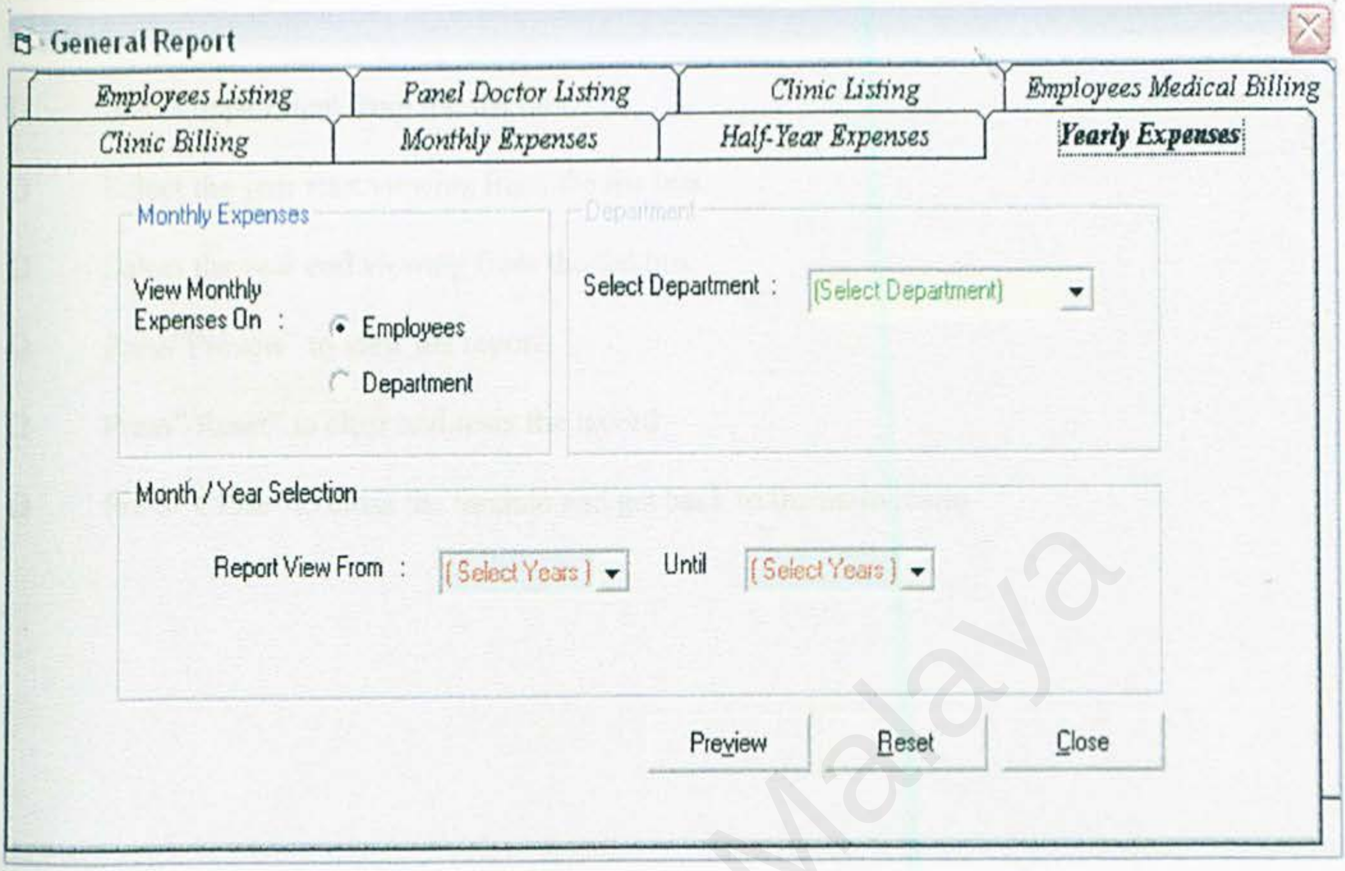
Ω Select the month end viewing from the option box

Ω Select particular years to view the report.

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record

Ω Press" Close" to close the module and get back to the main menu



View Monthly Expenses On : ● Employees ○ Department

Select Department : (Select Department)

Report View From : (Select Years) Until (Select Years)

Figure 8.31 Report Viewing, Company medical monthly Yearly expenses

To View the employees monthly medical yearly expenses report:

Ω Click on the Employees option button.

Ω Select the year start viewing from the list box.

Ω Select the year end viewing from the list box

Ω Press"Prevew" to view the report.

Ω Press" Reset" to clear and reset the record.

Ω Press" Close" to close the module and get back to the main menu.

To view employee's monthly medical yearly expenses based on department:

Ω Click on the Department option button.

Ω Select department from the list box.

Ω Select the year start viewing from the list box.

Ω Select the year end viewing from the list box

Ω Press"Prevew" to view the report.

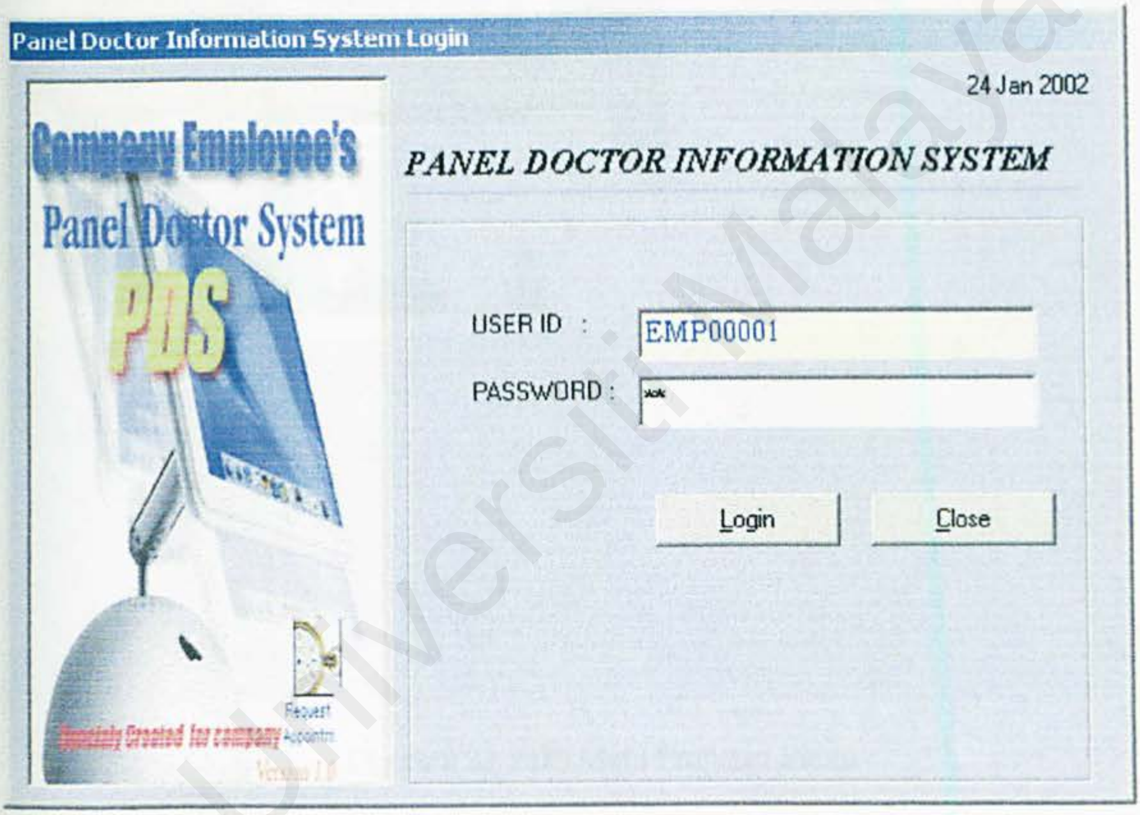
Ω Press" Reset" to clear and reset the record

Ω Press" Close" to close the module and get back to the main menu

SECTION 2: PDS SYSTEM NORMAL USER MANUAL

Basically, Normal user shares the same module with the Administrator. The only different is for Normal User module, there are some module were hide and not visible for them.

2.1 Main Login


Panel Doctor Information System Login
24 Jan 2002
Company Employee's
Panel Doctor System
PDS
Request Appointm
Specially Created for company
Version 1.0
PANEL DOCTOR INFORMATION SYSTEM
Login
Close

USER ID : EMP00001

PASSWORD : **

Figure 8.32 PDS Login Interface

To Login:

Ω (Same as the administrator module procedure)

2.2 PDS Main Program Menu

As the interface below shown, there are 2 modules which already hide for the normal user module.

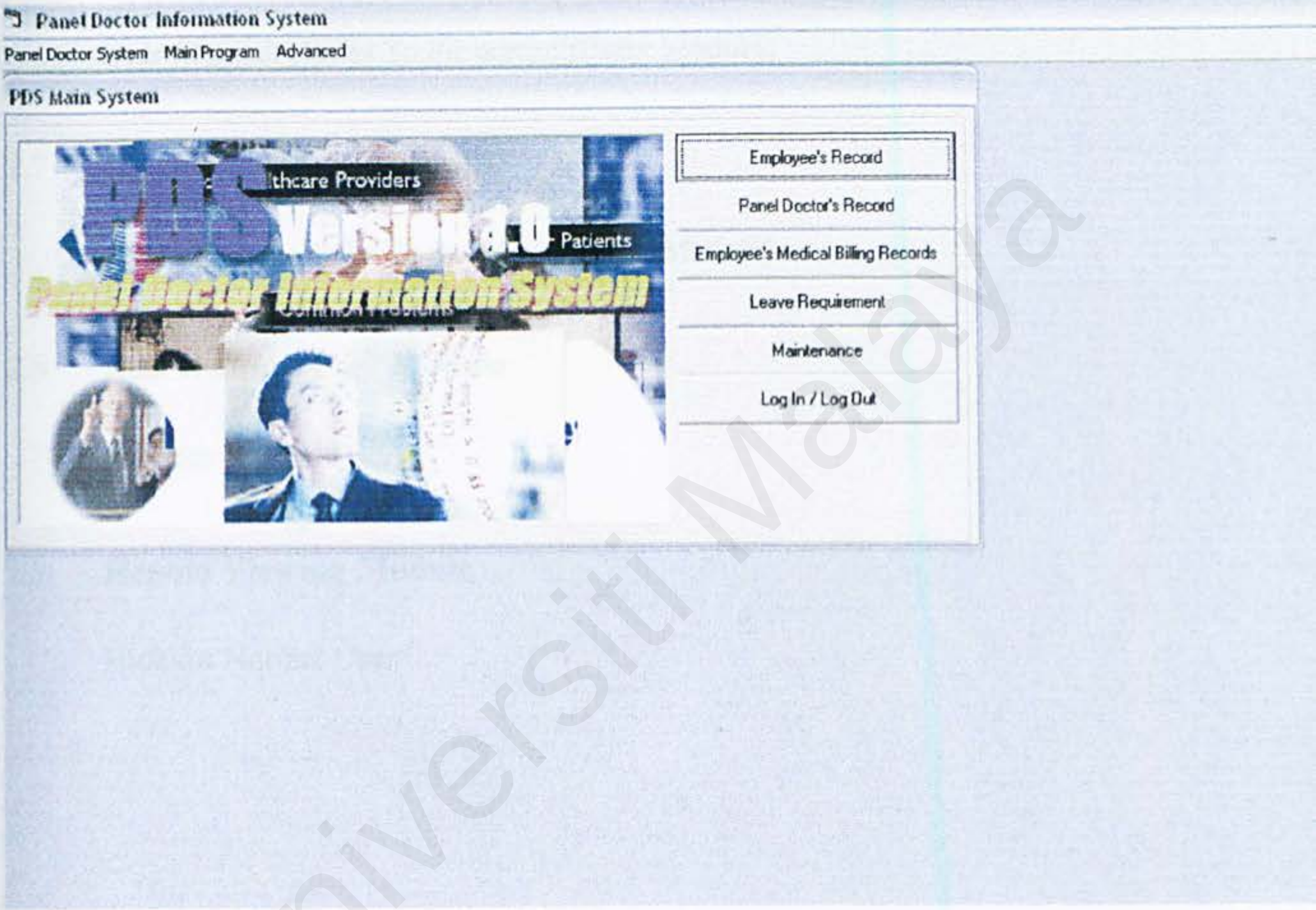


Figure 8.33 **PDS** Main Program Menu

2.3 PDS Employees Record

User Manual: (Refer To the administrator Module)

2.4 PDS Panel Doctors Record

User Manual: (Refer To the administrator Module)

2.5 Employees Medical Billing Record

User Manual: (Refer To the administrator Module)

2.6 Leave Requirement Record

User Manual: (Refer To the administrator Module)

2.7 User Maintenance Module

User Manual: (Refer To the administrator Module)

2.8 Add Remove User Module

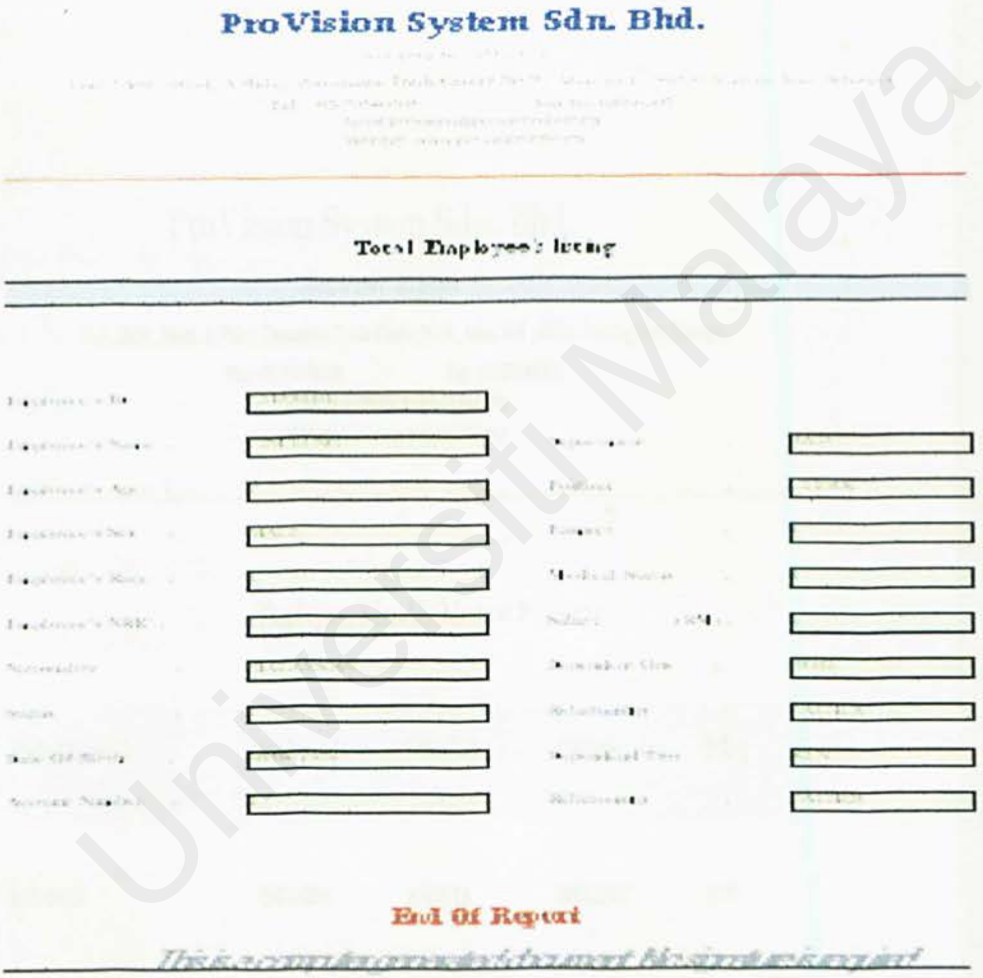
Hide for Normal User

2.9 Report Viewing Module

Hide for Normal User

APPENDIX:

Example Report from the system:



ProVision System Sdn. Bhd.

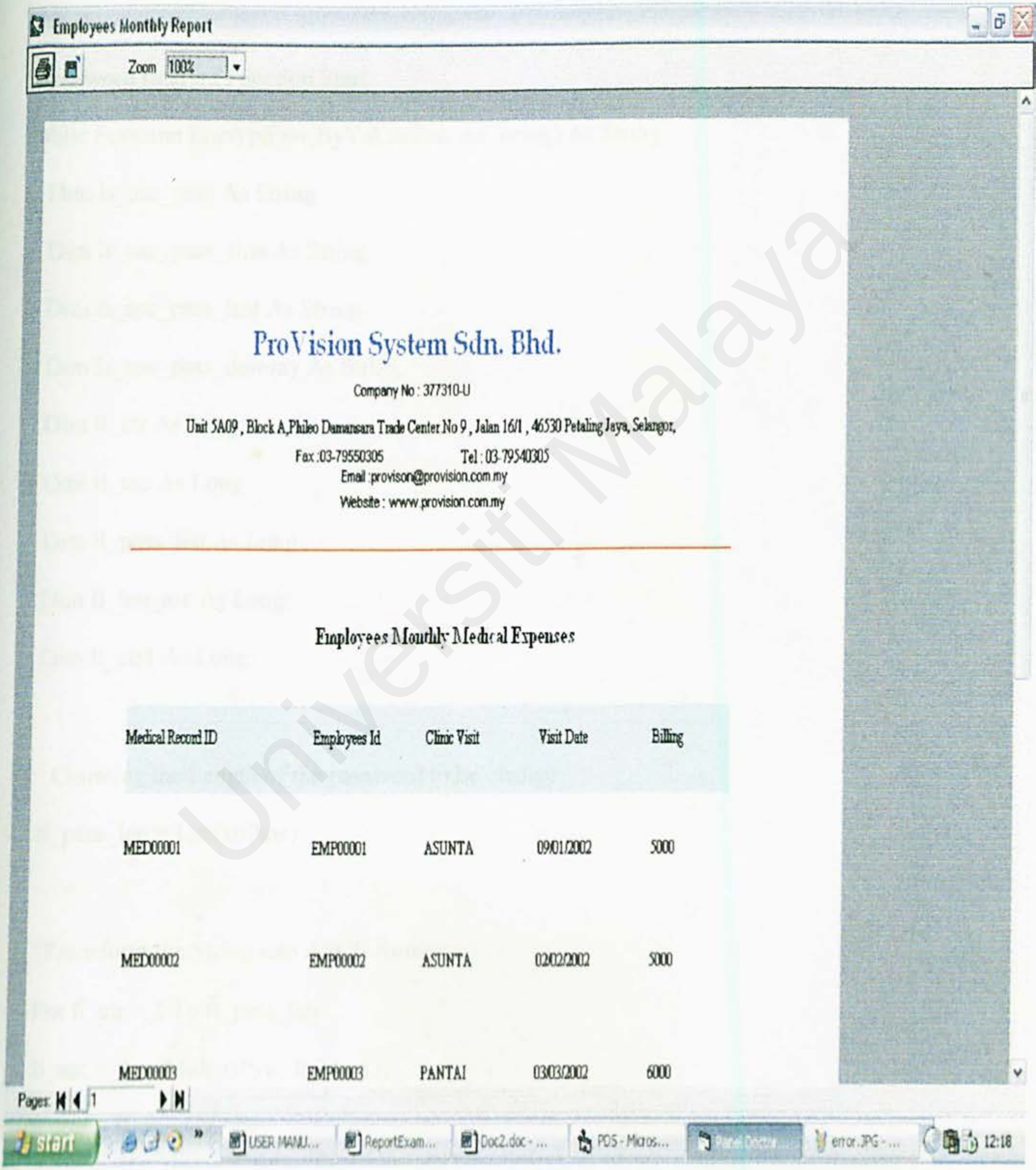
Total Employee's listing

Employee's ID	[illegible]		
Employee's Name	[illegible]	Department	[illegible]
Employee's Age		Position	[illegible]
Employee's Sex	[illegible]	[illegible]	
Employee's Race		Medical Status	
Employee's NRIC		Salary (RM)	
Nationality	[illegible]	Dependent Name	[illegible]
Status		Relationship	[illegible]
Date Of Birth	[illegible]	[illegible]	[illegible]
Account Number		[illegible]	[illegible]

End Of Report

This is a computer generated document. No signature is required

Appendix 2: Example Report 2:



ProVision System Sdn. Bhd.

Company No : 377310-U

Unit 5A09 , Block A,Phileo Damansara Trade Center No 9 , Jalan 16/1 , 46530 Petaling Jaya, Selangor,

Fax :03-79550305 Tel : 03-79540305

Email :provison@provision.com.my

Website : www.provision.com.my

Employees Monthly Medical Expenses

Medical Record ID	Employees Id	Clinic Visit	Visit Date	Billing
MED00001	EMP00001	ASUNTA	09/01/2002	5000
MED00002	EMP00002	ASUNTA	02/02/2002	5000
MED00003	EMP00003	PANTAI	03/03/2002	6000

Appendix 3: Microsoft Visual Basic Source code example from the **PDS** (Password Encryption)

```
‘ Password Encrypt Function Start

Public Function EncryptPsw(ByVal strPsw As String) As String

    Dim ls_asc_pass As String

    Dim ls_asc_pass_first As String

    Dim ls_asc_pass_last As String

    Dim ls_asc_pass_dummy As String

    Dim ll_ctr As Long

    Dim ll_asc As Long

    Dim ll_pass_len As Long

    Dim ll_len_asc As Long

    Dim ll_ctr1 As Long


    ‘ Counting the Length of the password to be change

    ll_pass_len = Len(strPsw)


    ‘Transform the String into ASCII format

    For ll_ctr = 1 To ll_pass_len

    ll_asc = Asc(Mid(strPsw, ll_ctr, 1))

    ll_asc = ll_asc + 7
```

```
If Len(CStr(ll_asc)) < 3 Then
    ls_asc_pass = ls_asc_pass + CStr(ll_asc) + "*"
Else
    ls_asc_pass = ls_asc_pass + CStr(ll_asc)
End If


Next
ll_len_asc = Len(ls_asc_pass)
ls_asc_pass_first = ""
ls_asc_pass_last = ""


ll_ctr1 = (ll_len_asc / 2)
For ll_ctr = 1 To (ll_len_asc / 2)
ls_asc_pass_first = ls_asc_pass_first + Mid(ls_asc_pass, ll_ctr1, 1)
ll_ctr1 = ll_ctr1 - 1
Next


ll_ctr1 = ll_len_asc
For ll_ctr = ((ll_len_asc / 2) + 1) To ll_len_asc
ls_asc_pass_last = ls_asc_pass_last + Mid(ls_asc_pass, ll_ctr1, 1)
ll_ctr1 = ll_ctr1 - 1
Next
```

```
    ls_asc_pass_dummy = ls_asc_pass_first + ls_asc_pass_last

    EncryptPsw = ls_asc_pass_dummy

End Function
```

REFERENCE:

1. Kroenke David M., "Database Processing Fundamentals, Design and Implementation", Prentice Hall International Inc., 1998

2. Kenneth E. Kendall & Julie E. Kendall, "Systems Analysis And Design, 4th Edition", Prentice-Hall Inc., 1998.

3. Davis Alan M. "Software Requirements, Objects, Functions and States", Prentice Hall International Inc.

4. "System Analysis and Design Methods" McGraw-Hill Irwin, 2000

5. "JAVA" retrieved from World Wide Web: http://java.sun.com

6. "Windows Information" retrieved from World Wide Web: http://www.microsoft.com

7. Jguru Web Site: http://www.jguru.com/.

8. Java World Web Site: http://www.javaworld.com/

9. "Faculty Resource System (FRS) and Industry Training Management System (ITMS)", Thesis title from Tan Ching Ching, UM, 1999/2000

10. "Web-Based University Examination Timetable System (ETS)", Thesis title from Loo Chia Ling, UM, 1998/1999 "WAP Messagging Application", Thesis title from Yew Hooi Phaik, UM, 2000/2001.