Developing a System for Inward Hospital Patient Management

By

Dimuth Roshan Sepalage
99/A/As/007

Dissertation submitted in partial fulfillment of the requirements for the degree of Bachelor of Science

In

Physical Sciences

Faculty of Applied Sciences

Sabaragamuwa University of Sri Lanka

Buttala

DECLARATION

"I certify that this dissertation dose not incorporate without acknowledgment any material previously submitted for degree or diploma in any university and to the best of my knowledge and belief it dose not contain any material previously published or written or orally communicated by another person except where due reference is made in the text"

Dimuth Roshan Sepalage.
99/A/AS/007

"To the best of my knowledge the above particulars are correct"

External Supervisor

T.Venugopal,
IDSystems (Pvt.) Ltd.
454/12A, Piachaud Gardens,
Kandy.

Internal Supervisor

Mr. Yves Noel
Visiting Specialist on IT,
Department of Physical Sciences,
Faculty of Applied Science,
Sabaragamuwa University of Sri Lanka.

Head of the Department

Dr(Mrs).N.Wickramaratne
Head/Dept Physical Sciences,
Faculty of Applied Science,
Sabaragamuwa University of Sri Lanka.

ACKNOWLEDGEMENTS

First and foremost I wish to express my deepest gratitude to my internal supervisor, Mr.Yves Noel, Visiting Specialist on IT, Department of Physical Sciences, Faculty of Applied Science, University of Sabaragamuwa for his assistance, encouragement, guidance and his valuable time to make this study a success.

I express my sincere gratitude to my external supervisor T.Venugopal, Network/Software Engineer, IDSystems (Pvt.) Ltd. Who kindly offered me industrial placement with all the facilities. And my special thanks for Elankayer Sithirasenan (Faculty of Engineering, University of Peradeniya), Director IDSystems (Pvt.) Ltd.

My deepest gratitude is expressed to Dr.D.B.M.Wickramaratne, Dean, Faculty of Applied Science, University of Sabaragamuwa and Dr (Mrs).N.Wickramaratne Head, Department of Physical Sciences, Faculty of Applied Science, University of Sabaragamuwa for guiding me towards a successful completion.

I am heavily indebted to Ms. Dilini Gunasekara, Director Singapore Informatics in Kandy Branch, who gave me invaluable support throughout my training period. I would also like to extend my thanks to all the staff members including R.Prakache and Deveaka Randeniya who were always willing to give their support and assistance to me throughout the training period.

Finally, I express my heart felt gratitude towards the lecturers for their co-operation throughout my study and my colleagues for their invaluable help and guidance given to me at all times.

ABSTRACT

This document is submitted to present the detailed description on the project, which I underwent at the work site of IDSystems (Pvt) Ltd in Kandy.I was trained under this project as a trainee developer for 5 months in partial fulfillment of the requirements for the degree of Bachelor of Science in Physical Sciences, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka. This project is a team effort and I took part in this project from the designing of the technical specifications. At the time of preparing this document the project was completed only up to the system testing stage. The rest of the stages were documented according to the way it is expected to be completed.

The project objective is to develop a user-friendly Hospital Inward Management system for the private channeling hospitals based on the functional specifications given by the Suwasewane Hospital in Kandy. To achieve this objective, Classical Life Cycle/ Waterfall model was used and the below given steps were followed.

- Defining the requirements of the system and get the approval for it from the Hospital.
- Designing the technical specifications (Table definitions, Table normalization)
- Develop the software

Hospital weather private or governments do not maintain patients' records in a systematic way. Mostly this is done manually and many situations these records are either misplaced or lost.

The purpose of this project is to develop a computer software t keep track of patients in hospital and to maintain these records for further reference. Also the System provides inward patient billing, receipting and channeling. This system is making for any hospital within changing simple operation and to use the system. This system is developed keeping in mind the further trends. It provides facilities for E-comers and Intranet access.

The front-end of the software is developed using HTML and Java scripts. The back-end used PHP, MySQL is used to maintain and administer the database. Finally the development is done on the Linux platform.

TABLE OF CONTENTS

I

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1 IDSystems (Pvt) Ltd.

1.1.1 Background

IDSystems (Pvt) Ltd is a professional organisation which provides an array of services including networking for business organisations, selling of personal computers, system development. The firm stated its business in 1993 and by today has been involved in many large projects and is one of the big names in Kandy.

The founder of this prestige organisation who is currently the owner is Mr. E.Sithirasenan an expect in the field of computer software development.

1.1.2 Services

IDSystems (Pvt) Ltd is of the view to provide complete solutions to the customer satisfying his needs. The services that this organisation provides range from software development to selling of personal computers.

One major area that this company does its business is assembly of personal computers. It has a very good market share in the Kandy region. Further business is expanded to such extent that it repairs computers and sells computer accessories.

IDSystems (Pvt) Ltd is one of the few Internet Service Providers in Kandy. It is the market leader as an ISP.

Another service is web designing which is the order of the day. There is a huge demand for web designing in the country as it heads towards meeting the IT demands. So **IDSystems (Pvt) Ltd** caters to the requirements of the customers by being innovative.

This organisation provides networking solutions as well.

Mainly as community welfare service it coordinates lectures on subjects like Linux Administration, Networking, E-Channelling etc.

IDSystems (Pvt) Ltd is one of the best organisations in the country specialising in Linux.

1.2 Project

1.2.1 Project description

The project is called Inward Patient Management System for a hospital. The software used to develop this is MySQL and PHP in Linux platform. This purpose of this project is to develop computer software to keep track of patients in hospital and to maintain these reports for further references. Also the system provides inward patient billing, receipting and channelling. This enables effective management.

This system is developed to be suitable for any hospital if necessary by doing only minor changes.

1.2.2 Objectives

The following have been identified as main objective of this project.

1. To provide a **user-friendly** solution to a Hospital where users of the system are not very much computer literate.
2. To develop the application with **minimum resources** and in a **minimum time frame**.

1.2.3 My Role in the Project

When I joined the project it was already started and data collection part was being done. I joined in to a team of 3.

My major involvements were data base creation, designing of user interfaces using PHP and data Flow diagrams. The creation of databases were almost handed all handled by me.

But before getting into the project full time I was asked to be thorough in MySQL and PHP by referring books and getting expertise help. As I was involved in the project I got the opportunity to get the knowledge of other areas about the project such as networking. Further I did my best inside the team and participated to the project as much as I could.

2. Linux Overview

2.1 What is an Operating System?

Operating system is a program or set of programs driving the raw hardware of computers, which manages the resources of the computer, is accordance with certain objectives providing higher layers of software with simplified computer.
In the past, almost all operating system were writing in a low level language. Currently, many operating systems are partly or complete written in a high level language.
All operating system have two major objectives. These are to make it possible for the resources of computer to be used efficiently, and to conceal the difficulties of dealing directly with the hardware of the computer.
The function of operating system may be loosely classified as time allocation, resources control, input/output control, error handling and protection, operator interface and accounting.
Some of most popular operating systems are:-

- Linux
- MSDOS
- MSWINDOWS
- NOVEL
- WINDOWS-NT

(Pitts David, Ball Bill Read Hat Linux, 1999)

2.2 About Linux/Unix Operating System

2.2.1 What is Linux?

Linux is an operating system that can be downloaded free and "belongs" to an entire community of developers, not one corporate entity. In other words, anyone from professional software developers to hobbyist computer hackers can access and make changes to the Linux kernel all the information about Linux is open and available to everyone. That's why Linux is known as "open source" or "free software," because there is nothing secret about this system. This freedom also allows companies to sell and distribute Linux on CD-ROM or by other means, although those companies must keep their code open to the public.
With more and more people looking for an alternative to Windows, Linux has recently grown in popularity and is quickly becoming a favorite among major corporations and curious desktop users. Not only does it give users a choice of operating systems, it also proves itself valuable with its power, flexibility, and reliability.

2.2.2 How did Linux get started?

The concept of open source programming has been around for many years its roots stem from universities that needed to be able to share information as well as allow students and

developers to adapt programs to meet their needs. In 1984, Richard Stallman, a researcher at the MIT AI Lab, started a project he called GNU to counter the fast-moving trend toward proprietary, fee-based software. Stallman, who remains an open advocate of open source, believes that making source code available to anyone who wants it is integral to furthering computer science and innovation.
This concept served as the basis of Linux development, the brainchild of Linus Torvalds. When Torvalds began developing Linux in 1991, he was a student at the University of Helsinki and originally targeted Linux at the Intel 386 (although it is now one of the most widely ported operating systems available for PCs). Torvalds wanted to write a new version of UNIX, so he and a group of programmers combined talents and created a core operating system called Linux. Linux almost had same Unix Like feature for e.g.

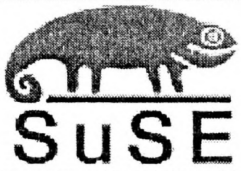
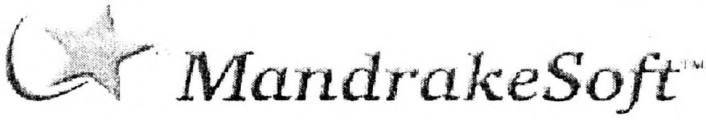
- Like Unix, Linux is also written in C.
- Like Unix, Linux is also the Multi-user/Multitasking/32 or 64 bit Network OS.
- Like Unix, Linux is rich in Development/Programming environment.
- Like Unix, Linux runs on different hardware platform; for e.g.
    - Intel x86 processor (Celeron/PII/PIII/PIV/Old-Pentiums/80386/80486)
    - Macintosh PC's
    - Cyrix processor
    - AMD processor
    - Sun Microsystems Sparc processor
    - Alpha Processor (Compaq)

It comes with a variety of s/w like send mail apache, Pine, netscape communicator, gnome etc.

So you can use Linux for:

- Personal Work
- Web Server
- Software Development Workstation
- Workgroup Server
- In Data Center for various server activities such as FTP, Telnet, SSH, Web, Mail, Proxy, Proxy
- Cache Appliance etc

(Pitts David, Ball Bill Read Hat Linux, 1999)

Linux distributions.	Website/Logo
Red Hat Linux: http://www.redhat.com/	
SuSE Linux: http://www.suse.com/	
Mandrake Linux: http://www.mandrakesoft.com/	
Caldera Linux: http://www.calderasystems.com/	
Debian GNU/Linux: http://www.debian.org/	
Slackware Linux: http://www.slackware.com/	

(www.Howto.com)

2.2.3 Hardware Requirements

To Install the Linux OS the following h/w are suggested:

- Intel x86 or Pentium II/333 Mhz
- 32 MB RAM
- 2 GB HDD of either IDE OR SCSI TYPE.
- 1.44 MB HDD
- Network Interface Card (for networking)
- Mouse
- SVGA Color Monitor

2.2.4 Important Parts of the kernel

The Linux kernel consists of several important parts:

- process management
- memory management
- hardware device drivers
- file system drivers
- Network management, and various other bits and pieces.

2.2.5 Major services in a UNIX system

- init

The single most important service in a UNIX system is provided by init. init is started as the first process of every UNIX system, as the last thing the kernel does when it boots. When init starts, it continues the boot process by doing various startup chores (checking and mounting file systems, starting daemons, etc).

- Logins from terminals

Logins from terminals (via serial lines) and the console (when not running X) are provided by the getty program. init starts a separate instance of getty for each terminal.

- Syslog

The kernel and many *system programs* produce error, warning, and other messages. It is often important that these messages can be viewed later, even much later, so they should be written to a file. The program doing this is **syslog.**

- cron and at

Each user can have a crontab file, where she lists the commands she wishes to execute and the times they should be executed. The **cron** daemon takes care of starting the commands when specified.

The **at** service is similar to **cron,** but it is once only: the command is executed at the given time, but it is not repeated.

- Graphical user interface

The graphical environment primarily used with Linux is called the X Window System (X for short). X also does not implement a user interface; it only implements a window system, i.e., tools with which a graphical user interface can be implemented.

Some popular window managers are: fvwm, icewm, blackbox and windowmaker. There are also two popular desktop managers, KDE and Gnome.

- Networking

UNIX operating systems have many networking features. Most basic services (filesystems, printing, backups, etc) can be done over the network. This can make system administration easier, since it allows centralised administration, while still reaping in the benefits of microcomputing and distributed computing, such as lower costs and better fault tolerance.

- Network logins

Network logins work a little differently than normal logins. There is a separate physical serial line for each terminal via which it is possible to log in. For each person logging in via the network, there is a separate virtual network connection, and there can be any number of these.

- Network file systems

With a network file system any file operations done by a program on one machine are sent over the network to another computer. This fools the program to think that all the files on the other computer are actually on the computer the program is running on. This makes information sharing extremely simple, since it requires no modifications to programs.

- Mail

The mail system consists of many programs. The delivery of mail to local or remote mailboxes is done by one program (the *mail transfer agent* (MTA), e.g., **sendmail** or **smail**), while the programs users use are many and varied *(mail user agent* (MUA), e.g., **pine**, **mutt** or **elm**). The mailboxes are usually stored in /var/spool/mail.

- Printing

The print queue software also *spools* the printouts on disk, i.e., the text is kept in a file while the job is in the queue. This allows an application program to spit out the print jobs quickly to the print queue software; the application does not have to wait until the job is actually printed to continue.

- The filesystem layout

The filesystem is divided into many parts; usually along the lines of a root filesystem with /bin, /lib, /etc, /dev, and a few others; a /usr filesystem with programs and unchanging data; a /var filesystem with changing data (such as log files); and a /home filesystem for everyone's personal files. Depending on the hardware configuration and the decisions of the system administrator, the division can be different; it can even be all in one filesystem.

(www.Redhat.com)

2.3 Frequently Used Linux Commands

2.3.1 Basic Linux Commands

All of UNIX is case sensitive. A command with even a single letter's capitalization altered is considered to be a completely different command. The same goes for files, directories, configuration file formats, and the syntax of all native programming languages.

- **ls**

  List the information about the FILEs.

ls -a

The -a option means to list *all* files as well as hidden files.

Ls -l

Which lists the contents in *long* format?

- **cp**

  The command cp stands for *copy*. It duplicates one or more files. The format is

cp <file> <newfile>

- **cd**

  The cd command is used to take you to different directories.

- **mkdir**

  Creates a new directory.

Eg:

mkdir foo

cd foo

- **pwd**

The command pwd stands for *present working directory* (also called the *current directory*) and tells what directory you are currently in.

- **rm**

To remove (i.e., erase or delete) a file, use the command rm <filename>.

- **rmdir**

To remove a directory, use the command rmdir <dir>.

Note that you cannot remove a directory unless it is empty. To remove a directory as well as any contents it might contain, use the command rm -R <dir>.

- **mv**

The mv command is used to move files and directories. It really just renames a file to a different directory.

Eg: mv apple.txt foo/orange.txt

- **man**

The command man <command> displays help on a particular topic and stands for *manual*.

- **cat**

cat <filename> [<filename> ...]

Writes the contents of all the files listed to the screen. cat can join a lot of files together with cat <filename> <filename> ... > <newfile>. The file <newfile> will be an end-on-end *concatenation* of all the files specified.

- **clear**

Erases all the text in the current terminal.

- **df**

Stands for *disk free* and tells you how much free space is left on your system.

- **du**

du <directory>

Stands for *disk usage* and prints out the amount of space occupied by a directory.

- **head**

head [-n <lines>] <filename>

Prints the first <lines> lines of a file or 10 lines if the -n option is not given

- **wc**

wc [-c] [-w] [-l] <filename>

Counts the number of bytes (with -c for character), or words (with -w), or lines (with -l) in a file.

- **whoam**

Prints your login name.

- **grep**

grep [options] <pattern> <filename>

grep -n <pattern> <filename>

shows the line number in the file where the word was found.

grep -<num> <pattern> <filename>

prints out <num> of the lines that came before and after each of the lines in which the word was found.

grep -A <num> <pattern> <filename>

prints out <num> of the lines that came After each of the lines in which the word was found.

grep -B <num> <pattern> <filename>

prints out <num> of the lines that came Before each of the

lines in which the word was found.

- **find**

the find command is used to search for files.

Eg: find ./ -name foo*.c -print

grep -<num> <pattern> <filename>

prints out <num> of the lines that came before and after each of the lines in which the word was found.

grep -A <num> <pattern> <filename>

prints out <num> of the lines that came After each of the lines in which the word was found.

grep -B <num> <pattern> <filename>

prints out <num> of the lines that came Before each of the lines in which the word was found.

- **find**

  the find command is used to search for files.

  Eg: find ./ -name foo*.c -print

- **chmod**

chmod command is used to change the permissions of a file. It's usually used as follows:

chmod [-R] [u|g|o|a][+|-][r|w|x|s|t] <file>

For example,

chmod u+x myfile

adds execute permissions for the user of myfile.

- **date**

  Prints out the current date and time.

- **ps**

  List Running Processes.

- **stat**

To get a complete listing of the file's permissions.

- **free**

Prints out available free memory. You will notice two listings: swap space and physical memory.

- **less**

With less, you can use the arrow keys to page up and down through the file.

- **tar**

When many files are packed together into one, this packed file is called an *archive*. Usually archives have the extension .tar, which stands for *tape archive*.

To *create* an archive of a directory, use the tar command:

tar -c -f <filename> <directory>

Create a directory with a few files in it, and run the tar command to back it up. A file of <filename> will be created. You can also use the *verify* option (see the man page) of the tar command to check the integrity of <filename>. Now remove the directory, and then restore it with the *extract* option of the tar command:

tar -x -f <filename>

- **vi**

To edit a text file.

- **exit**

To logout from the current login.

2.3.2 Advanced Linux Commands

Device Files

The MAKEDEV Script

Most device files will already be created and will be there ready to use after you install your Linux system. If by some chance you need to create one which is not provided then you should first try to use the **MAKEDEV** script.

```
# /dev/MAKEDEV -v ttyS0
create ttyS0 c 4 64 root:dialout 0660
```

This will create the device file /dev/ttyS0 with major node 4 and minor node 64 as a character device with access permissions 0660 with owner root and group dialout. ttyS0 is a serial port. The major and minor node numbers are numbers understood by the kernel. The kernel refers to hardware devices as numbers, this would be very difficult for us to remember, so we use filenames. Access permissions of 0660 means read and write permission for the owner (root in this case) and read and write permission for members of the group (dialout in this case) with no access for anyone else.

The mknod command

MAKEDEV is the preferred way of creating device files which are not present. However sometimes the **MAKEDEV** script will not know about the device file you wish to create. This is where the **mknod** command comes in. In order to use **mknod** you need to know the major and minor node numbers for the device you wish to create.

The devices.txt file in the kernel source documentation is the canonical source of this information.

To take an example, let us suppose that our version of the **MAKEDEV** script does not know how to create the /dev/ttyS0 device file. We need to use **mknod** to create it. We know from looking at the devices.txt file that it should be a character device with major number 4 and minor number 64. So we now know all we need to create the file.

 # mknod /dev/ttyS0 c 4 64
 # chown root.dialout /dev/ttyS0
 # chmod 0644 /dev/ttyS0
 # ls -l /dev/ttyS0
crw-rw---- 1 root dialout 4, 64 Oct 23 18:23 /dev/ttyS0

Formatting

Formatting is the process of writing marks on the magnetic media that are used to mark tracks and sectors.

Floppies are formatted with **fdformat**. The floppy device file to use is given as the parameter. For example, the following command would format a high density, 3.5 inch floppy in the first floppy drive:

$ fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.

The **badblocks** command can be used to search any disk or partition for bad blocks (including a floppy). It does not format the disk, so it can be used to check even existing filesystems. The example below checks a 3.5 inch floppy with two bad blocks.

$ badblocks /dev/fd0H1440 1440

Filesystems

A *filesystem* is the methods and data structures that an operating system uses to keep track of files on a disk or partition; that is, the way the files are organized on the disk.

Most UNIX filesystem types have a similar general structure, although the exact details vary quite a bit. The central concepts are *superblock*, *inode*, *data block*, *directory block*, and *indirection block*.

The superblock contains information about the filesystem as a whole, such as its size (the exact information here depends on the filesystem). An inode contains all information about a file, except its name.

The name is stored in the directory, together with the number of the inode. A directory entry consists of a filename and the number of the inode which represents the file. The inode contains the numbers of several data blocks, which are used to store the data in the file.

Linux supports several types of filesystems. As of this writing the most important ones are:

minix

The oldest, presumed to be the most reliable, but quite limited in features (some time stamps are missing, at most 30 character filenames) and restricted in capabilities (at most 64 MB per filesystem).

xia

A modified version of the minix filesystem that lifts the limits on the filenames and filesystem sizes, but does not otherwise introduce new features. It is not very popular, but is reported to work very well.

ext2

The most featured of the native Linux filesystems, currently also the most popular one. It is designed to be easily upwards compatible, so that new versions of the filesystem code do not require re-making the existing filesystems.

In addition, support for several foreign filesystem exists, to make it easier to exchange files with other operating systems.

dos, vfat, iso9660, nfs, smbfs

Creating a file system

Filesystems are created, i.e., initialised, with the **mkfs** command. There is actually a separate program for each filesystem type. **mkfs** is just a front end that runs the appropriate program depending on the desired filesystem type. The type is selected with the -t fstype option.

The programs called by **mkfs** have slightly different command line interfaces. The common and most important options are summarized below; see the manual pages for more.

-t fstype
Select the type of the filesystem.

-c
Search for bad blocks and initialise the bad block list accordingly.

-l filename
Read the initial bad block list from the name file.

To create an ext2 filesystem on a floppy, one would give the following commands:

```
$ fdformat -n /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity
1440 kB.
Formatting ... done
$ badblocks /dev/fd0H1440 1440 >bad-blocks
$ mkfs -t ext2 -l bad-blocks /dev/fd0H1440
```

```
mke2fs 0.5a, 5-Apr-94 for EXT2 FS 0.5, 94/03/10
360 inodes, 1440 blocks
72 blocks (5.00%) reserved for the super user
First data block=1
Block size=1024 (log=0)
Fragment size=1024 (log=0)
1 block group
8192 blocks per group, 8192 fragments per group
360 inodes per group
Writing inode tables: done
Writing superblocks and filesystem accounting information
done
$
```

2.3.3 Mounting and unmounting

Before one can use a filesystem, it has to be *mounte*d. The operating system then does various bookkeeping things to make sure that everything works. Since all files in UNIX are in a single directory tree, the mount operation will make it look like the contents of the new filesystem are the contents of an existing sub-directory in some already mounted filesystem.

The mounts could be done as in the following example:

```
$ mount /dev/hda2 /home
$ mount /dev/hda3 /usr
$
```

The **mount** command takes two arguments. The first one is the device file corresponding to the disk or
partition containing the filesystem. The second one is the directory below which it will be mounted.

Linux supports many filesystem types. **mount** tries to guess the type of the filesystem. You can also use the -t fstype option to specify the type directly; this is sometimes

necessary, since the heuristics **mount** uses do not always work. For example, to mount an MS-DOS floppy, you could use the following command:

$ mount -t msdos /dev/fd0 /floppy

When a filesystem no longer needs to be mounted, it can be unmounted with **umount**. **umount** takes one argument: either the device file or the mount point.

$ umount /dev/hda2

$ umount /usr

2.3.4 Checking filesystem integrity with fsck

Most systems are setup to run **fsck** automatically at boot time, so that any errors are detected (and hopefully corrected) before the system is used.

fsck must only be run on unmounted filesystems, never on mounted filesystems (with the exception of the read-only root during startup). This is because it accesses the raw disk, and can therefore modify the filesystem without the operating system realizing it. There *will* be trouble, if the operating system is confused.

It can be a good idea to periodically check for bad blocks. This is done with the **badblocks** command. It outputs a list of the numbers of all bad blocks it can find. This list can be fed to **fsck** to be recorded in the filesystem data structures so that the operating system won't try to use the bad blocks for storing data.

The following example will show how this could be done.

$ badblocks /dev/fd0H1440 1440 >bad-blocks

$ fsck -t ext2 -l bad-blocks /dev/fd0H1440

Parallelising fsck version 0.5a (5-Apr-94)

e2fsck 0.5a, 5-Apr-94 for EXT2 FS 0.5, 94/03/10

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Check reference counts.

Pass 5: Checking group summary information.

/dev/fd0H1440: ***** FILE SYSTEM WAS MODIFIED *****

/dev/fd0H1440: 11/360 files, 63/1440 blocks

2.3.5 Fighting fragmentation

When a file is written to disk, it can't always be written in consecutive blocks. A file that is not stored in consecutive blocks is *fragmented*.

It takes longer to read a fragmented file, since the disk's read-write head will have to move more. It is desirable to avoid fragmentation, although it is less of a problem in a system with a good buffer cache with read-ahead.

Ext2 effectively always allocates the free block that is nearest to other blocks in a file. For ext2, it is therefore seldom necessary to worry about fragmentation. There is a program for defragmenting an ext2 filesystem called, strangely enough, **defrag**

2.3.6 Boots And Shutdowns

During bootstrapping, the computer first loads a small piece of code called the *bootstrap loader*, which in turn loads and starts the operating system. The bootstrap loader is usually stored in a fixed location on a hard disk or a floppy.

The boot process

The boot sector contains a small program (small enough to fit into one sector) whose responsibility is to read the actual operating system from the disk and start it. On a Linux boot floppy, there is no filesystem, the kernel is just stored in consecutive sectors, since this simplifies the boot process. It is possible, however, to boot from a floppy with a filesystem, by using LILO, the LInux LOader.

When booting with LILO, it will normally go right ahead and read in and boot the default kernel. It is also possible to configure LILO to be able to boot one of several kernels, or even other operating systems than Linux, and it is possible for the user to choose which kernel or operating system is to be booted at boot time.

LILO can be configured so that if one holds down the **alt**, **shift**, or **ctrl** key at boot time (when LILO is loaded), LILO will ask what is to be booted and not boot the default right away. Alternatively, LILO can be configured so that it will always ask, with an optional timeout that will cause the default kernel to be booted.

With LILO, it is also possible to give a *kernel command line argument*, after the name of the kernel or operating system

2.3.7 More about shutdowns

It is important to follow the correct procedures when you shut down a Linux system. If you fail do so, your filesystems probably will become trashed and the files probably will become scrambled. This is because Linux has a disk cache that won't write things to disk at once, but only at intervals. This greatly improves performance but also means that if you just turn off the power at a whim the cache may hold a lot of data and
that what is on the disk may not be a fully working filesystem (because only some things have been written to the disk).

Another reason against just flipping the power switch is that in a multi-tasking system there can be lots of things going on in the background, and shutting the power can be quite disastrous. By using the proper shutdown sequence, you ensure that all background processes can save their data.
If your system has many users, use the command **shutdown -h +time message**, where time is the time in minutes until the system is halted, and message is a short explanation of why the
system is shutting down.

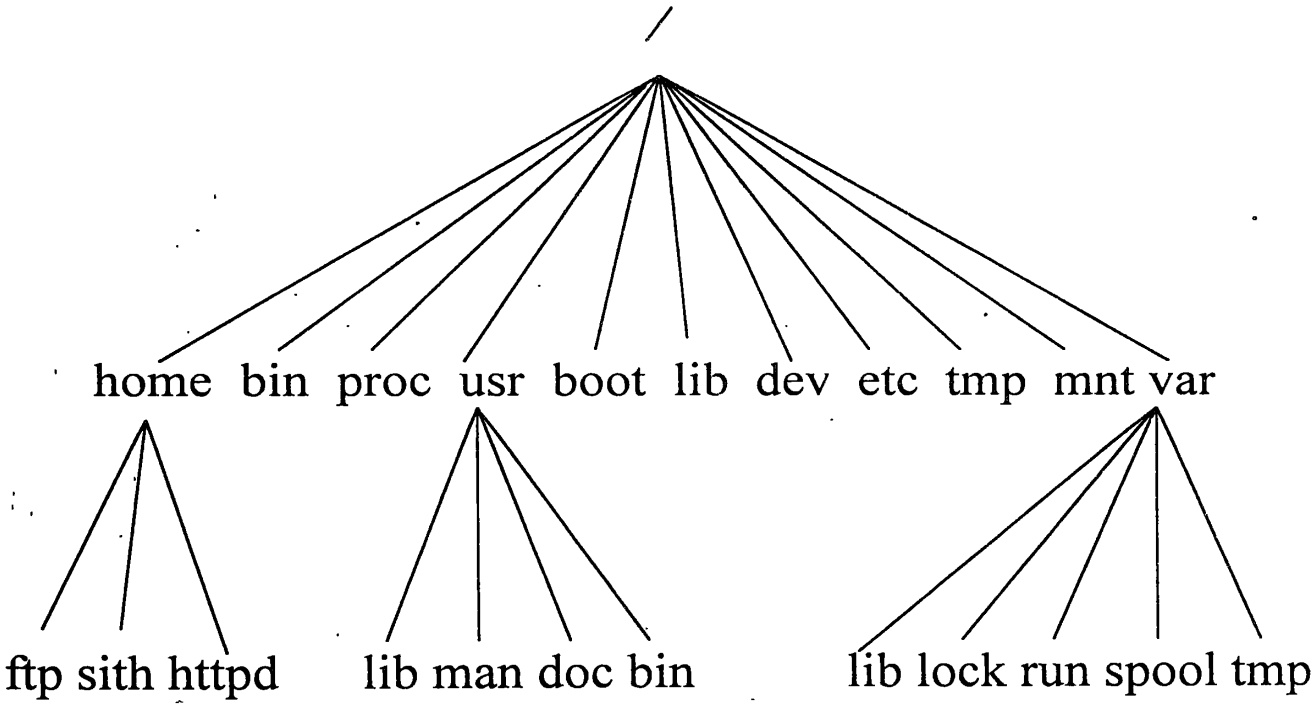
```
# shutdown -h +10 'We will install a newdisk. System should
> be back on-line in three hours.'
#
```

This will warn everybody that the system will shut down in ten minutes, and that they'd better get lost or lose data.

When the real shutting down starts after any delays, all filesystems (except the root one) are unmounted, user processes (if anybody is still logged in) are killed, daemons are shut down, all filesystem are unmounted, and generally everything settles down.

2.3.8 Overview of the Directory Tree

- The full directory tree is intended to be breakable into smaller parts, each capable of being on its own disk or partition.
- The major parts are the root (/), /usr, /var, and /home filesystems. Each part has a different purpose. The directory tree has been designed so that it works well in a network of Linux machines which may share some parts of the filesystems over a read-only device (e.g., a CD-ROM), or over the network with NFS.



- The root filesystem is specific for each machine (it is generally stored on a local disk, although it could be a ramdisk or network drive as well) and contains the

files that are necessary for booting the system up, and to bring it up to such a state that the other filesystems may be mounted.

- The /usr filesystem contains all commands, libraries, manual pages, and other unchanging files needed during normal operation. No files in /usr should be specific for any given machine, nor should they be modified during normal use.

- The /var filesystem contains files that change, such as spool directories (for mail, news, printers, etc), log files, formatted manual pages, and temporary files. Traditionally everything in /var has been somewhere below /usr, but that made it impossible to mount /usr read-only.
- The /home filesystem contains the users' home directories, i.e., all the real data on the system. Separating home directories to their own directory tree or filesystem makes backups easier; the other parts often do not have to be backed up, or at least not as often as they seldom change. A big /home might have to be broken across several filesystems, which requires adding an extra naming level below /home, for example /home/students and /home/staff.

(Pitts David, Ball Bill Read Hat Linux,1999)

2.4 Vi states

Vi has 3 modes:

1. ***command mode*** - Normal and initial state; others return here (use **ESC** to abort a partially typed command)
2. ***input mode*** - entered by specific commands **a i A I o O c C s S R** and ended by **ESC** or abnormally with interrupt
3. ***line mode*** - i.e. waiting for input after a **:** , **/** , **?** or a **!** command (end with **CR**, abort with **CTRL-c**). **CTRL** is the control key: **CTRL-c** means "control c"

2.4.1 Shell Commands

1. **TERM=** *code* Puts a code name for your terminal into the variable **TERM**
2. **export TERM** Conveys the value of **TERM** (the terminal code) to any UNIX system program that is terminal dependant.
3. **tput init** Initializes the terminal so that it will function properly with various UNIX system programs.

4. **vi** *filename* Accesses the **vi** screen editor so that you can edit a specified file.
5. **vi** *file1 file2 file3* Enters three files into the **vi** buffer to be edited. Those files are *file1, file2,* and *file3*.
6. **view** *file* Invoke vi editor on *file* in read-only mode
7. **vi -R** *file* Invoke vi editor on *file* in read-only mode
8. **vi -r** *file* Recover *file* and recent edits after system crash

2.4.2 Interrupting, canceling

1. **ESC** end insert or incomplete command
2. **CTRL-? CTRL** is the control key: **CTRL-?** means "control ?" delete or rubout interrupts
3. **CTRL-l** reprint/refresh screen if CTRL-? scrambles it

2.4.3 File Manipulation

1. **ZZ** Save the file and exit vi
2. **:wq** Save the file and exit vi
3. **:w** Write the current file
4. **:w!** Force write the current file, if file is read-only
5. **:w** *name* Write to file *name*
6. **:q** Exit from vi
7. **:q!** Force exit from vi (discarding changes)
8. **:e name** Edit file *name*
9. **:e!** reedit, discard changes
10. **:e + name** edit file *name*, starting at end
11. **:e +** n edit starting at line *n*
12. **:e #** edit alternate file
13. **:n** edit next file in *arglist*
14. **:args** list files in current filelist
15. **:rew** rewind current filelist and edit first file
16. **:n args** specify new arglist
17. **:f** show current file and line
18. **CTRL-G** synonym for :f , show current file and line
19. **:ta tag** to tag file entry *tag*
20. **CTRL-]** :ta, following word is tag

2.4.4 Corrections during insert

1. **CTRL-h** Erase last character
2. **CTRL-w** Erase last word
3. **erase** Press DELETE key, same as CTRL-h
4. **kill** Your kill key, erase input this line
5. \ Escapes CTRL-h, DELETE and kill
6. **ESC** Ends insertion, back to command
7. **CTRL-?** Interrupt, terminates insert

8. **CTRL-d** Backtab over *autoindent*
9. **CTRL-v** Quote non-printing character

2.4.5 Delete

1. **x** Delete the character under the cursor
2. **X** Delete the charater before the cursor
3. **D** Delete to the end of line
4. **d^** Delete back to start of line
5. **dd** Delete the current line
6. **ndd** Delete n lines starting with the current one
7. **dnw** Delete n words starting from cursor

2.4.6 Insert, change

1. **i** Enter input mode inserting before the cursor
2. **I** Enter input mode inserting before the first non-blank character
3. **a** Enter input mode inserting after the cursor
4. **A** Enter input mode inserting after the end of the line
5. **o** Open a new line below current line and enter input mode
6. **O** Open a new line above current line and enter input mode
7. **r** Replace the character under the cursor (does NOT enter input mode)
8. **R** Enter input mode replacing characters
9. **C** shift-c. Change rest of line
10. **D** shift-d. Delete rest of line
11. **s** Substitute chars
12. **S** Substitute lines
13. **J** Join lines
14. **J** Join lines

2.4.7 Copy and Paste

The "yank buffer" is filled by *EVERY* delete command, or explicitely by **Y** and **yy**.

1. **Y** Copy the current line to the yank buffer
2. **n**yy Copy n lines starting from the current to the yank buffer
3. **p** Paste the yank buffer after the cursor (or below the current line)
4. **P** Paste the yank buffer before the cursor (or above the current line)
5. "x**p** Put from buffer x
6. "x**y** Yank to buffer x
7. "x**d** Delete into buffer x

2.4.8 Operators (use double to affect lines)

1. **d** delete
2. **c** change

3. < left shift
4. > right shift
5. ! filter through command
6. = indent for LISP
7. y yank text to buffer

(www.Redhat.com)

2.5 Why Linux use in this project?

It is our convenient. I believe Linux is the high security and less bug compare with Microsoft Windows. The files management of Linux is higher than other operating systems. It is fast working in sell mode comparing other operating system. Using Linux server we can connect other dummy terminals using telnet it is very convenient.

3. Learning MySQL

3.1 What is a database system?

A database is a collection of interrelated data items that can be processed by one or more application systems. A database system is comprised of a database, general-purpose software called the database management system (DBMS) that manipulate the database and appropriate hardware and personal.

(Sommerville Ion, Software Engineering, 1996)

3.2 What is My SQL?

3.2.1 Introduction to MySQL

MySQL is a database management system.

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL. Since computers are very good at handling large amounts of data, database management plays a central role in computing, as stand-alone utilities, or as parts of other applications.

MySQL is a relational database management system.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The tables are linked by defined relations making it possible to combine data from several tables on request. The SQL part of MySQL stands for "Structured Query Language" - the most common standardized language used to access databases.

MySQL is Open Source Software.

Open Source means that it is possible for anyone to use and modify. Anybody can download MySQL from the Internet and use it without paying anything. Anybody so inclined can study the source code and change it to fit their needs. MySQL uses the GPL (GNU General Public License) http://www.gnu.org, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL into a commercial application you can buy a commercially licensed version from us.

Why use MySQL?

MySQL is very fast, reliable, and easy to use. If that is what you are looking for, you should give it a try. MySQL also has a very practical set of features developed in very close cooperation with our users. MySQL was originally developed to handle very large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Though under constant

development, MySQL today offers a rich and very useful set of functions. The connectivity, speed, and security make MySQL highly suited for accessing databases on the Internet.

The technical features of MySQL

MySQL is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and several programming interfaces. We also provide MySQL as a multi-threaded library which you can link into your application to get a smaller, faster, easier to manage product.

MySQL has a lot of contributed software available.

It is very likely that you will find that your favorite application or language already supports MySQL.

The official way to pronounce MySQL is ``My Ess Que Ell" (not MY-SEQUEL). But we try to avoid correcting people who say MY-SEQUEL.

The database has become an integral part of almost every human's life. Without it, many things we do would become very tedious, perhaps impossible tasks.

Banks, universities, and libraries are three examples of organizations that depend heavily on some sort of database system. On the Internet, search engines, online shopping, and even the website naming convention (http://www...) would be impossible without the use of a database. A database that is implemented and interfaced on a computer is often termed a database server.

- One of the fastest SQL (Structured Query Language) database servers currently on the market is the MySQL server.
- The capabilities range across a number of topics, including the following:
  - Ability to handle an unlimited number of simultaneous users.
  - Capacity to handle 50,000,000+ records.
  - Very fast command execution, perhaps the fastest to be found on the market.
  - Easy and efficient user privilege system

(DuBois Paul MySQL, First Indian Edition 2000)

3.2.2 MySQL Commands

- **Connect to MySQL.**

  This involves the insertion of the hostname, username and password given specifically for MySQL use.

Syntax: mysql h hostname -u username -p

or

mysql -u username --password

The user will then be prompted for a password If that works, you should see some introductory information followed by a mysql> prompt:

Shell> mysql -h host -u user -p

Enter password: ********

Welcome to the MySQL monitor. Commands end with;

or \g.

Your MySQL connection id is 459 to server version:

3.22.20a-log

Type 'help' for help.

Mysql>

The prompt tells you that mysql is ready for you to enter commands.

- **Creating and Selecting a Database**
  - Mysql> CREATE DATABASE database name
  - Your database needs to be created only once, but you must select it for use each time you begin a mysql session. You can do this by issuing a USE statement.

Mysql> USE database base

Database changed

- **Creating a Table**

```
Mysql>  CREATE  TABLE  test  (
>  name  VARCHAR  (15),
>  email  VARCHAR  (25),
>  phone_number  INT,
>  ID  INT  NOT  NULL  AUTO_INCREMENT,
> PRIMARY KEY (ID) );
```

Query OK, 0 rows affected (0.10 sec)

Mysql>

- **Describe Command**

To verify that your table was created the way you expected,

Use a DESCRIBE statement:

Mysql> DESCRIBE test;

- **Insertion of data into the table**

Mysql> INSERT INTO test (name, email,

Mysql> phone_number) VALUES

Mysql> ('Bunny', 'carrots@devshed.com',

Mysql> 5554321) ;

Query OK, 1 row affected (0.02 sec)

Mysql>

- **Select Command**

SELECT what_to_select FROM which table WHERE

conditions_to_satisfy

Mysql> SELECT * FROM test

Mysql> WHERE name = 'Bugs';

- **Delete Command**

Mysql> DELETE FROM test

Mysql> WHERE name = 'Bugs Bunny';

- **Update Command**

Mysql> UPDATE test SET name = 'Duck'

Mysql> WHERE name = 'Bunny

- **Alter Command**

Rename the table

Mysql> ALTER table test RENAME mytest;

Add a column

Mysql> ALTER table mytest ADD birthday DATE;

Modify a column

Mysql> ALTER table mytest CHANGE

Mysql> name newname VARCHAR (25);

Delete a column

Mysql> ALTER table mytest DROP newname

(www.Mysql.com)

3.3 Why MySQL use in this project?

MySQL is a very fast, multithreaded, multi-user and robust SQL (Structured Query Language) database server.MySQL is a database management system.MySQL is relational database management system. It means the relational database store data in a separate table rather than putting all data in one big store room. This adds speed and flexibility.MySQL is open source software. It means it's possible for anyone to use and modify. The connectivity, speed and security make MySQL highly suited for accessing database on the internet.MySQL handle a data limiting. There for it's called lightweight database. But that must more than enough our requirements.MySQL is support very accurate under Linux platform.

4. Learning PHP

4.1 What is a scripting language?

Scripting language was created by Netscape. The idea was to make web pages more dynamical to getting more interactive with the people purpose programming language to control the behaviour of software objects.
Many web sites employ simple multimedia provided by Java, such as animator class.
Scripting means have less vocabulary than normal programming language.
E.g.: for scripting languages:

- JavaScript
- PHP
- Perl
- ActiveX

(Musciano Chuck & Kennedy Bill, HTML & XHTML)

4.2 Introduction to PHP

4.2.1 What is PHP?

PHP (recursive acronym for "PHP: Hypertext Preprocessor") is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.

Example. 1 An introductory example

```
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
        echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

Notice how this is different from a script written in other languages like Perl or C -- instead of writing a program with lots of commands to output HTML, you write an HTML script with some embedded code to do something (in this case, output some text). The PHP code is enclosed in special start and end tags that allow you to jump into and out of "PHP mode".

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server? If you were to have a script similar to the above on your server, the client would receive the results of running that script, with no way of determining what the

underlying code may be. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

(www.PHP.com)

4.2.2 Why we are use PHP

Working under the Linux platform, we can work with Python, Pearl, Java but PHP is simplest and powerful language.PHP is widely used as open source general purpose scripting language. That is specially suited for web development and can be embedded into HTML.Using HTML we can not create a dynamic pages, but using PHP can create dynamic pages.PHP is server-side scripting, so we can do anything any other CGI program can do. Such as collect from data, generate dynamic page content, send and receive cookies.PHP is also command line scripting and writing client-side GUI applications.
PHP is support for wide range of databases.
Such as,

- Adabas D
- Dbase
- Oracle
- Ingres
- MySQL
- Informix
- ODBC
- Velocis
- Unixdbm

(Castagnetto Jesus, Rawat Harish, Schumann Sascha,Scollo Chris & Veliath Deepak Professional PHP Programming)

4.2.3 Basic syntax of PHP

Escaping from HTML

When PHP parses a file, it simply passes the text of the file through until it encounters one of the special tags which tell it to start interpreting the text as PHP code. The parser then executes all the code it finds, up until it runs into a PHP closing tag, which tells the parser to just start passing the text through again. This is the mechanism which allows you to embed PHP code inside HTML: everything outside the PHP tags is left utterly alone, while everything inside is parsed as code.

There are four sets of tags which can be used to denote blocks of PHP code. Of these, only two (<?php. . .?> and <script language="php">. . .</script>) are always available; the others can be turned on or off from the php.ini configuration file. While the short-form tags and ASP-style tags may be convenient, they are not as portable as the longer versions. Also, if you intend to embed PHP code in XML or XHTML, you will need to use the <?php. . .?> form to conform to the XML.

The tags supported by PHP are:

Example 4.2.1: Ways of escaping from HTML

```
1. <? echo ("this is the simplest, an SGML processing instruction\n"); ?>
   <?= expression ?> This is a shortcut for "<? echo expression ?>"

2. <?php echo("if you want to serve XHTML or XML documents, do like
this\n"); ?>

3. <script language="php">
      echo ("some editors (like FrontPage) don't
          like.processing instructions");
   </script>

4. <% echo ("You may optionally use ASP-style tags"); %>
   <%= $variable; # This is a shortcut for "<% echo . . ." %>
```

The first way is only available if short tags have been enabled. This can be done via the **short_tags()** function (PHP 3 only), by enabling the short_open_tag configuration setting in the PHP config file, or by compiling PHP with the --enable-short-tags option to **configure**.

Again, the second way is the generally preferred method, as it allows for the the use of PHP in XML-conformant code such as XHTML.

The fourth way is only available if ASP-style tags have been enabled using the asp_tags configuration setting.

Note: Support for ASP-style tags was added in 3.0.4.

The closing tag for the block will include the immediately trailing newline if one is present. Also, the closing tag automatically implies a semicolon; you do not need to have a semicolon terminating the last line of a PHP block.

PHP allows you to use structures like this:

Example 4.2.2: Advanced escaping

```
<?php
if ($expression) {
   ?>
   <strong>This is true.</strong>
   <?php
} else {
   ?>
   <strong>This is false.</strong>
   <?php
}
?>
```

This works as expected, because when PHP hits the ?> closing tags, it simply starts outputting whatever it finds until it hits another opening tag. The example given here is contrived, of course, but for outputting large blocks of text, dropping out of PHP parsing mode is generally more efficient than sending all of the text through **echo()** or **print()** or some such.

(www.PHP.com)

4.2.4 Instruction separation

Instructions are separated the same as in C or Perl - terminate each statement with a semicolon.

The closing tag (?>) also implies the end of the statement, so the following are equivalent:

```
<?php
    echo "This is a test";
?>

<?php echo "This is a test" ?>
```

4.2.5 Comments

PHP supports 'C', 'C++' and UNIX shell-style comments.

For example:

```
<?php
```

```
    echo "This is a test"; // This is a one-line c++ style comment
    /* This is a multi line comment
       yet another line of comment */
    echo "This is yet another test";
    echo "One Final Test"; # This is shell-style style comment
?>
```

The "one-line" comment styles actually only comment to the end of the line or the current block of PHP code, whichever comes first.

```
<h1>This is an <?php # echo "simple";?> example.</h1>
<p>The header above will say 'This is an example'.
```

You should be careful not to nest 'C' style comments, which can happen when commenting out large blocks.

```
<?php
 /*
   echo "This is a test"; /* This comment will cause a problem */
 */
?>
```

(www.PHP.com)

4.3 Why PHP use in this project?

Working under the Linux platform, we can work with Python, Perl, Java but PHP is simplest and powerful language.PHP is widely used as open source general purpose scripting language. That is specially suited for web development and can be embedded into HTML.Using HTML we can not create a dynamic pages. but using PHP can create dynamic pages.PHP is server-side scripting, so we can do anything any other CGI program can do. Such as collect from data, generate dynamic page content, send and receive cookies.PHP is also command line scripting and writing client-side GUI applications.
PHP is support for wide range of databases.PHP is support also MySQL very compatibly.

5. Inward Patient Management System for Hospitals

5.1 Project Plan

5.1.1 System Development Life Cycle

The tasks involved in the development of the Hospital In ward patient management System (HWM) are given in the diagram below. It defines the steps involved in the project, and IDSystems, deliverables and approval points during the project life cycle.
Each of the tasks is described in details in the following chapters.

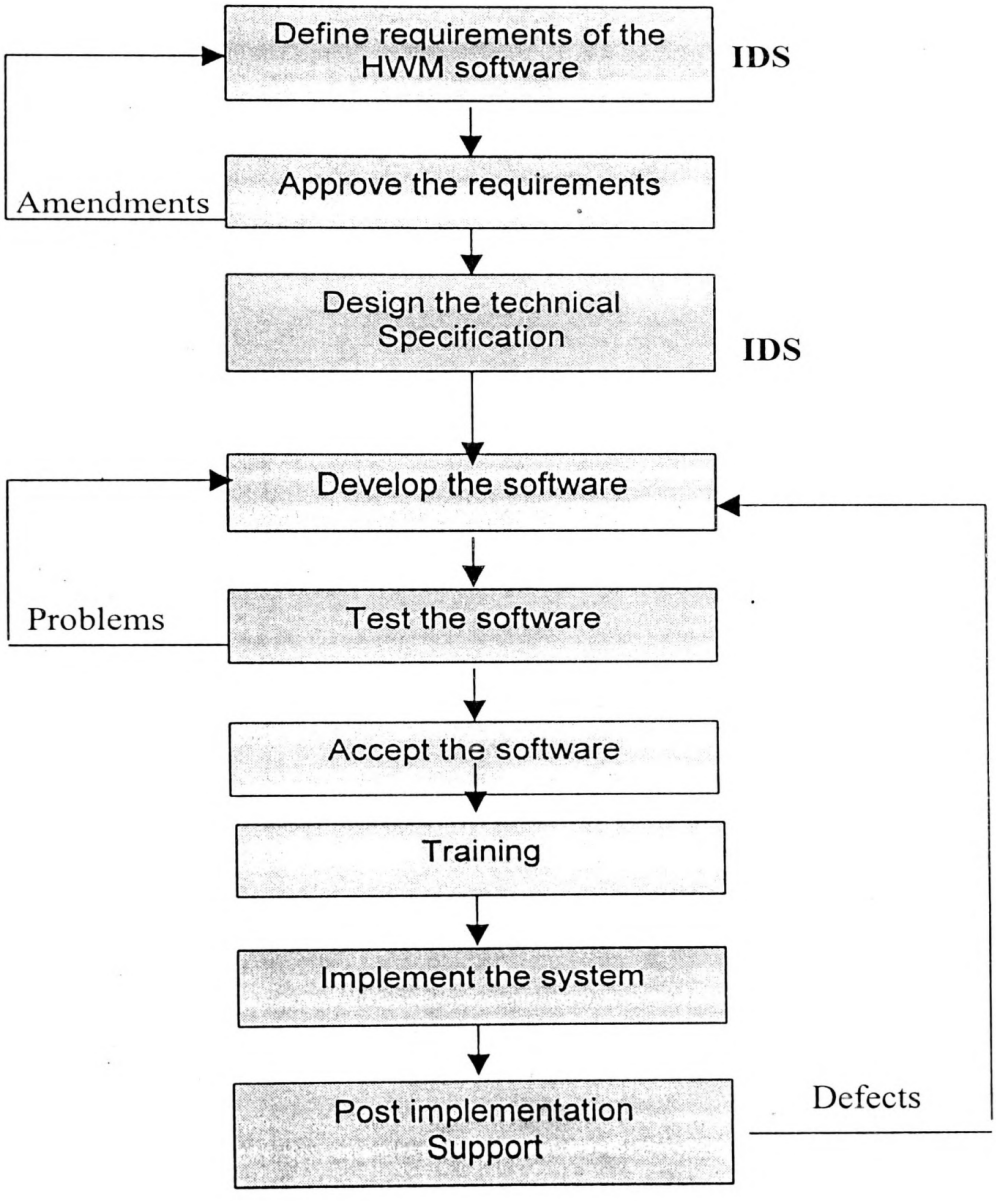


Figure 5.1 **HWM system development life cycle**

In according to my project life cycle:-

Project plan: Requirements have to be defined of HWM software. In this step we establishment of cost and time estimates and resource requirements for each phase of the project. We do preliminary analysis after that detail analysis. Then we give the detail report.

Product specification: Approve the requirements. Under this step. It describes how the product should behave from the user point. We approve the detail report and sign the agreements.

Architectural designing: We design the technical specification. In this step we essentially design the top level moulder design with preliminary versions of the interface plane, user guide, test plane.

Detail design: Under this step we develop the software. Continuing the top-down design, but likely with individual teams working strictly in their own areas except for period of integration tests and meeting.

Implementation and Testing: Under this we test the software. We put into user and implementation of the design, debugging and testing from the component level through the system level.

Qualification testing and final documentation releases: Under this step we accept the software and writing a final documentation. In meanwhile we training the users and we try to more the system user friendly one.

Maintenance: Implement the system and post implementation support under this step we connection and enhancement and adaptations the system.

(Sommerville Ion, Software Engineering, 1996)

5.1.2 Resource Allocation

The project team consist of the following personals

- Project manager/ System Designer
- Technical Designer
- Senior programmer
- Three trainee programmers
- Crystal Report trainee
- Quality Assurance Team

5.1.3 Time Allocation

Time allocation for each task of the system was estimated in '*Man Days*' which implies the number of days needed for an individual to complete a certain task.

Task	Man Days
Project Planning	8
Functional Design	5
Technical Design	2
Program Design	5
Test Plane creation	5
Database creation	2
Menu Structure creation	2
Develop and unit Test	77
User Documentation	6
Acceptance Test	2
Bug Fixing	6
Final release	1
Training and User Support	24
	145

Table 5.1 **Time Estimates for HWM System**

5.2 Design a System

5.2.1 Problem with the Existing System

Following are some of the main problems with the existing system.

- There is no proper way of patient information collecting.
- Difficult to save pass patient records.
- Difficult to prepare final settlement bill.
- Difficult to mange financial documents.

5.2.2 New application requirement

Using Linux server and making appropriate network system. This power full server to handle the data base system smoothly.

The entire patients are listed within the Patient_id.

5.2.3 Proposed Architecture

- The technology solution is specified in 4 sections.
- Network and communication infrastructure.
- System Software, consisting of the operating system.
- Computer Hardware.
- Database system.

5.2.4 Scope

The project is to study the hospital inward patient management and identify the existing drawbacks, identify the new requirements and re-engineer and come up with the new proposal for a new system.

Designing the new database structure using Mysql and the network layout come under the scope of this project. There for a network solution to inter connect of the computers with the other components would be proposed. And an overall databases structure would be proposed as well.

5.2.5 Computer Hardware

Major requirements for Linux server. Other client servers can be connecting using Tel-net.

5.2.6 Data base

The database will be MySql, becauseit'sveryfast, multithreaded, multi-userandrobut. SQL (Structured Query Language) database Server, MySql is data base management system. MySql is a relational database management system. It means the relational database store in a separate table rather than putting all data in one big storeroom. This adds speed and flexibility. MySql is open source software. It's mean possible for anyone to use and modify. The connectivity, speed and security make MySql highly suited for accessing databases on the Internet. MySql handling a data limiting. They're for it's called lightweight database. But that limits is more than our requirements.

5.2.7 Main system processing

Main system processing:-

- Channeling
- Ward management
- Accounting
- Other patient test

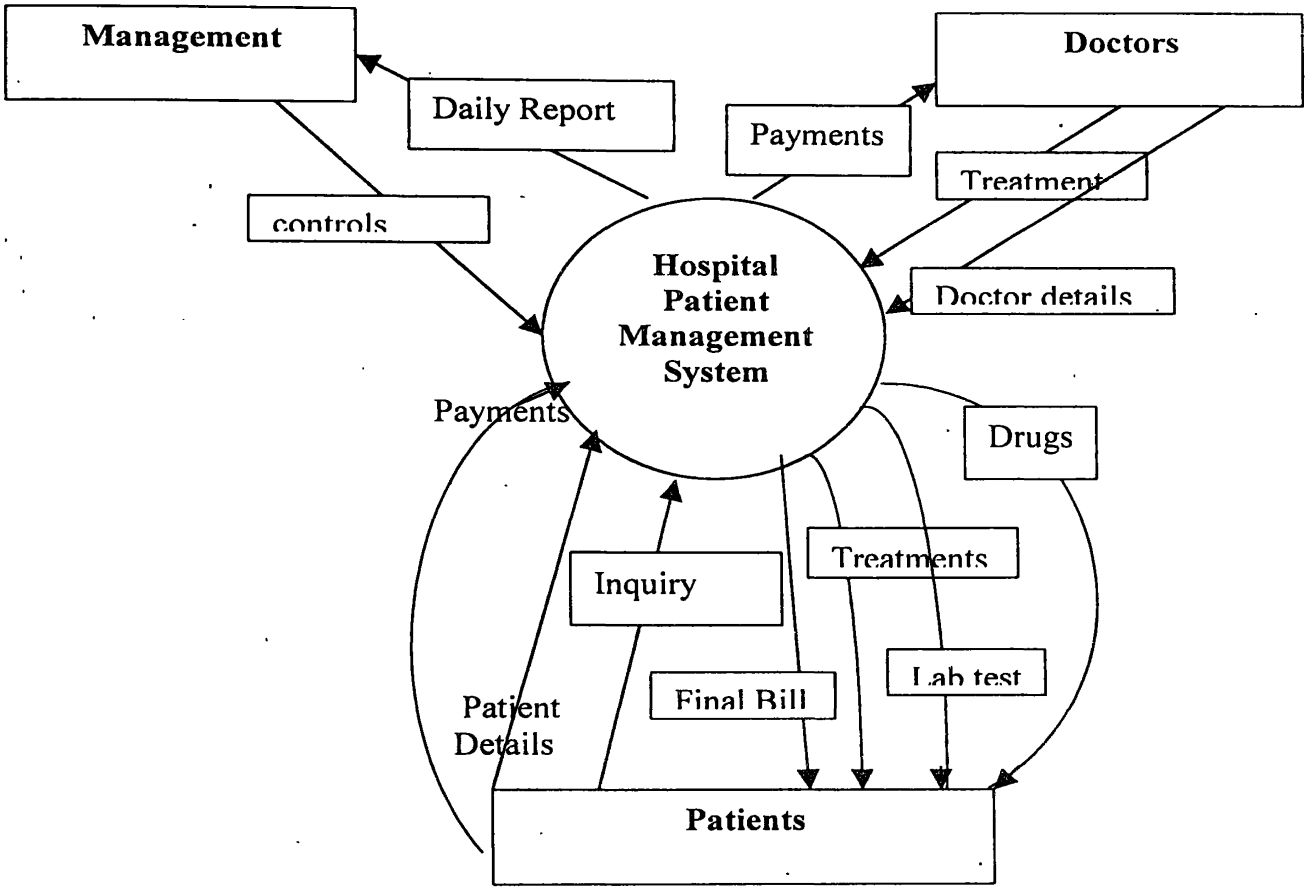
Pseudo code for laboratory process:

Begin
Get ward massage or receipt
Release test report
Send charges report to account section
End.

Pseudo code for Pharmacy process:

```
Begin
Apply "purchase order "
Receive drugs
Return defective or expire drugs
If (O.P.D patient)
 Get receipt
 Supply drugs
 Send bill for account section
Else (external patient)
 Get receipt
 Supply drugs
End.
```

5.2.8 Drawing a Data flow diagram



5.2.8.1: Context diagram for Inward Patient Management System

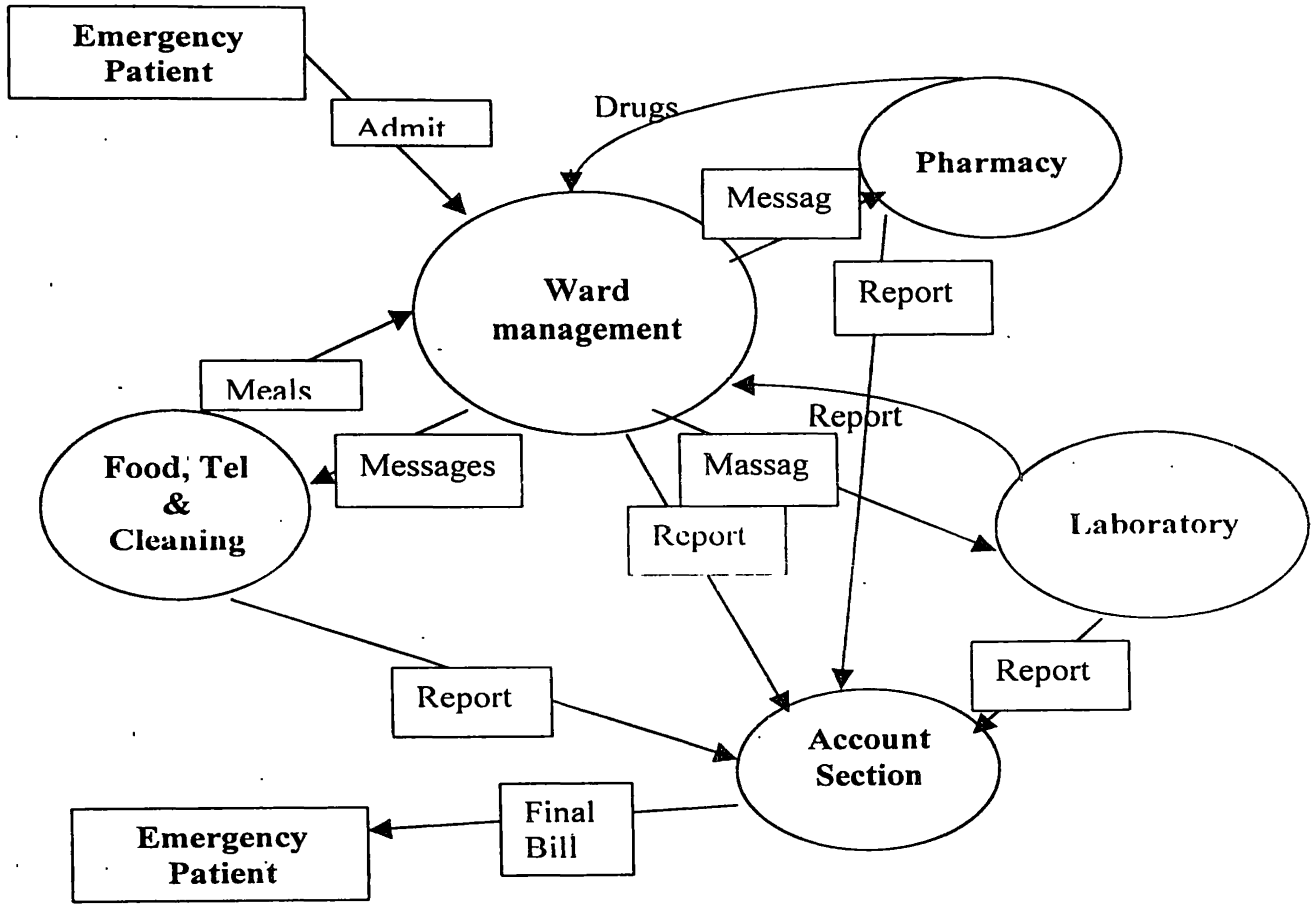
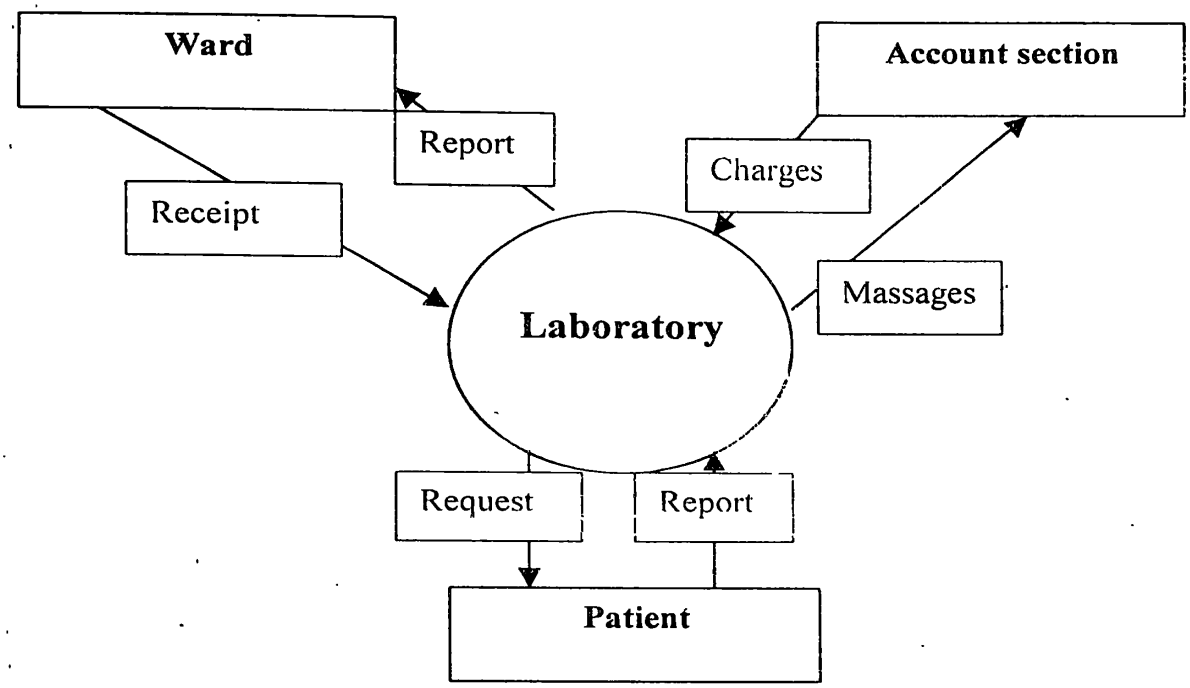
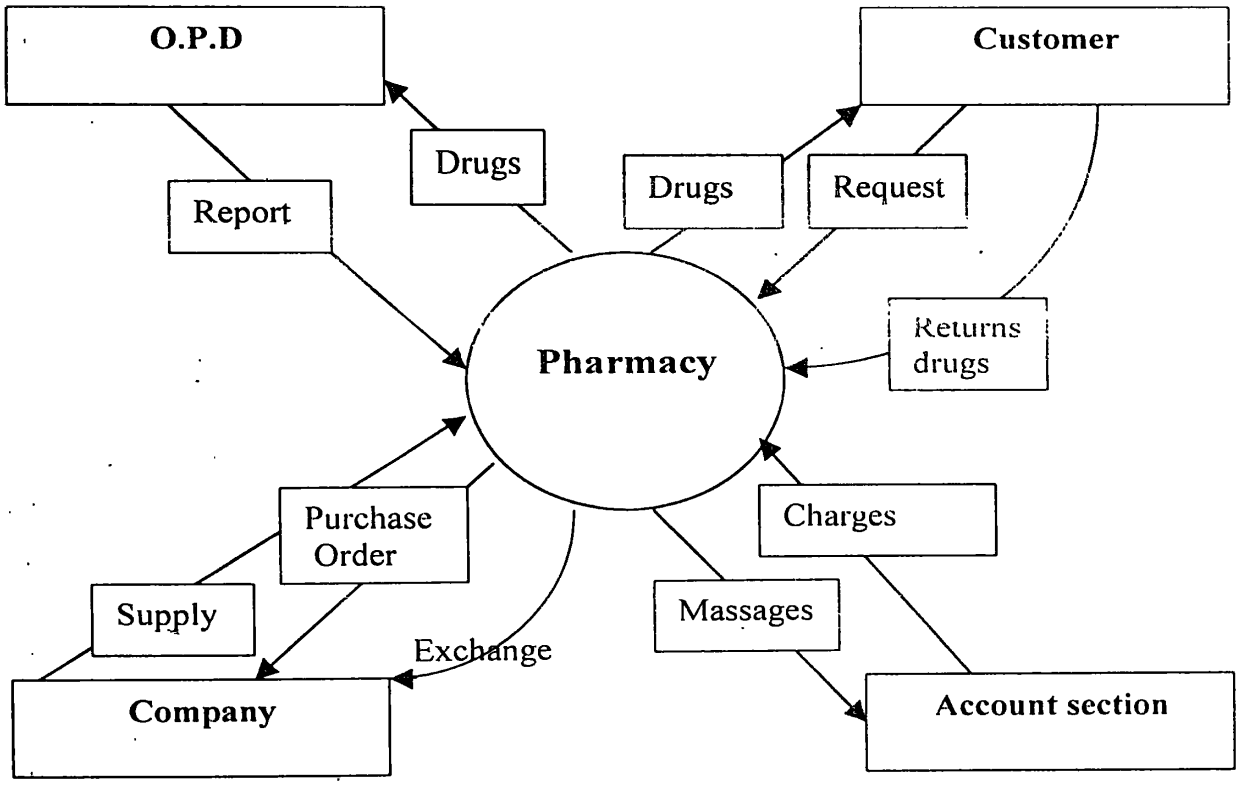


Figure 5.2.8.2: Level 0 DFD for "Management ward process"



5.2.8.3: Top level DFD for Laboratory process



5.2.8.4: Top level DFD for Pharmacy process

5.2.9 Design databases using MySql

Appendix-A:

5.2.10 Design User interfaces using PHP

Appendix-B:

4. Discussion

The project was scheduled to be completed from 145 man-days. But it was delayed. The main reason for the delay was to analyse the problem and getting data from a hospital. Here the procedure of collecting data was one by meeting all of the key persons who's information was vital in doing this work. It has been problem meeting all of them. Therefore data collection could not because of one time. But delay was not because the staffs were so busy. Further the requirements of the hospital staff for which the system was developed changed day by day affecting the team designing and analysing the problem.

Personally I also had difficult times in my part of the project. Since I was not very much familiar with advanced features of the software I used, I had to refer to the manuals regularly. For example, I had to get the number of days a patient spent in the ward where there was a conversion to be done from minutes to seconds. In doing this I had to refer PHP manual and MySQL manual several Times. One day forgets my password in Linux machine, and then I refer the Red-hat-manual and solve the problem.

In overall I gathered lots of knowledge by being involved in this project.

- The practical aspect of designing DFD models and table structures
- How to use the main functions of MySQL
- Designing user interfaces with PHP.
- The practical problems arise in the software development process and how to handle them.
- And the common things that we need to know about the working environment and the software development field.
- Assembling of computer, network cabling and installation.

Although I got involved in many aspects of the project I couldn't get involved with testing part and collection of data including user requirements. Other than the project I had little opportunity in getting involved with other managerial aspects of the firm.

At the time of my departure the project was at the testing stage. Where unit testing id begin done after finishing preliminary testing. There for customer response is getting to be known. My thought on the system is that it would be a convenient, accurate and user friendly one once it is implemented.

This system can be integrated with e-commerce and e-channelling can be incorporated into this.

I feel proud and happy to be involved in such an advanced project. This environment, where I worked in, provided me the confidence of facing problems and pressure situations very well.

5. Reference

- Boumphrey Frank, Greer, Dave Cassandra Raggett, Raggett Jenny, Schnitzenbaumer Sebastian & Wugofski Ted ,**Beginning XHTML**,1th ed. Published by Wrox pressLtd.
- Castagnetto Jesus, Rawat Harish, Schumann Sascha,Scollo Chris & Veliath Deepak **Professional PHP Programming**. Published by Wrox Press Ltd.
- DuBois Paul **MySQL**, First Indian Edition 2000."General Mysql Use", Copyright 1999 by New Riders Publishing.
- **HTML & XHTML** (The Definitive Guide) Musciano Chuck & Kennedy Bill.
- www.Howto.com
- www.Mysql.com
- www.PHP.com
- Pitts David, Ball Bill **Read Hat Linux**, 3rd ed. P 17-35. Copyright 1999 by Sams Publishing.
- www.Redhat.com
- Sommerville Ion (Lancaster University) **Software Engineering**, 5th ed.Software Design. Copyright 1996 by Addison-Wesly.

Appendix-A:

Field	Type	Null	Key	Default
ID	Int(20)	PRI	Null	Auto increment
Age	Int(5)		NULL	
Address	varchar(20)	Yes	NULL	
Sex	enum('F','M')	Yes	NULL	
PhoneNo	Int(10)	Yes	NULL	
EntryDate	Date		0000-00-00	
WardNo	Int(10)	Yes	NULL	
Status	Enum('Mr','Miss','Mrs')	Yes	NULL	
FirstName	varchar(20)			
LastName	Varchar(20)			
Name	Varchar(40)			
EntryTime	Time		00:00:00	
BHT No	Int(10)	Yes	NULL	
RoomNo	Int(10)	Yes	NULL	
BedNo	Int(10)	Yes	NULL	
Consultant	varchar(30)	Yes	NULL	

Table 5.2.9.1: To keep patient information

Field	Type	Null	Key	Default
Bgroup	Varchar(10)			
density	Varchar(10)	Yes	NULL	
Volume	varchar(10)	Yes	NULL	
Date	Date		0000-00-00	
PaID	Int(20)		0	
ID	Int(20)	PRI	NULL	Auto increment

Table5.2.9.2: To keep patient Blood information

Field	Type	Null	Key	Default
TestNo	Int(10)	Yes	NULL	
Amount	Float(10,2)	Yes	NULL	
PaID	Int(11)	Yes	0	
ID	Int(20)	PRI	NULL	Auto increment

Table 5.2.9.3: To keep patient Test Charges information

Field	Type	Null	Key	Default
PaID	Int(20)	Yes	NULL	
ID	Int(20)	PRI	NULL	Auto increment
DischargeDate	Date	Yes	NULL	
DischaregeTime	Time	Yes	NULL	
RentDays	Int(10)	Yes	NULL	
RoomCharges	Int(20)	Yes	NULL	

Table5.2.9.4: To keep patient Final data

Field	Type	Null	Key	Default
PaID	Int(20)		0	
ID	Int(20)	PRI	NULL	Auto increment
Date	Date		0000-00-00	
Sugar	Varchar(11)	Yes	NULL	
Cells	Varchar(15)	Yes	NULL	
Water	Varchar(10)	Yes	NULL	

Table 4.2.9.5: To keep patient Urine test data

Field	Type	Null	Key	Default
TestNo	Int(10)	Yes	NULL	
TestName	Varchar(20)	Yes	NULL	
Active	Enum('yes','no')			Yes
amountT	Float(10,2)	Yes	NULL	

Table 5.2.9.6: To keep patient Test type

Field	Type	Null	Key	Default
PaID	Int(20)		0	
ID	Int(20)	PRI	NULL	Auto increment
Date	Date		0000-00-00	
Others	Varchar(10)	Yes	NULL	
Homonetype	Varchar(15)	Yes	NULL	

Table 5.2.9.7: To keep patient Hormones test information

Appendix-B:

This is the first screen of HWPM (hospital ward patients' management system):

Figure: 5.2.10.1 User Interface 1-Screen

This is the second screen to insert patients:

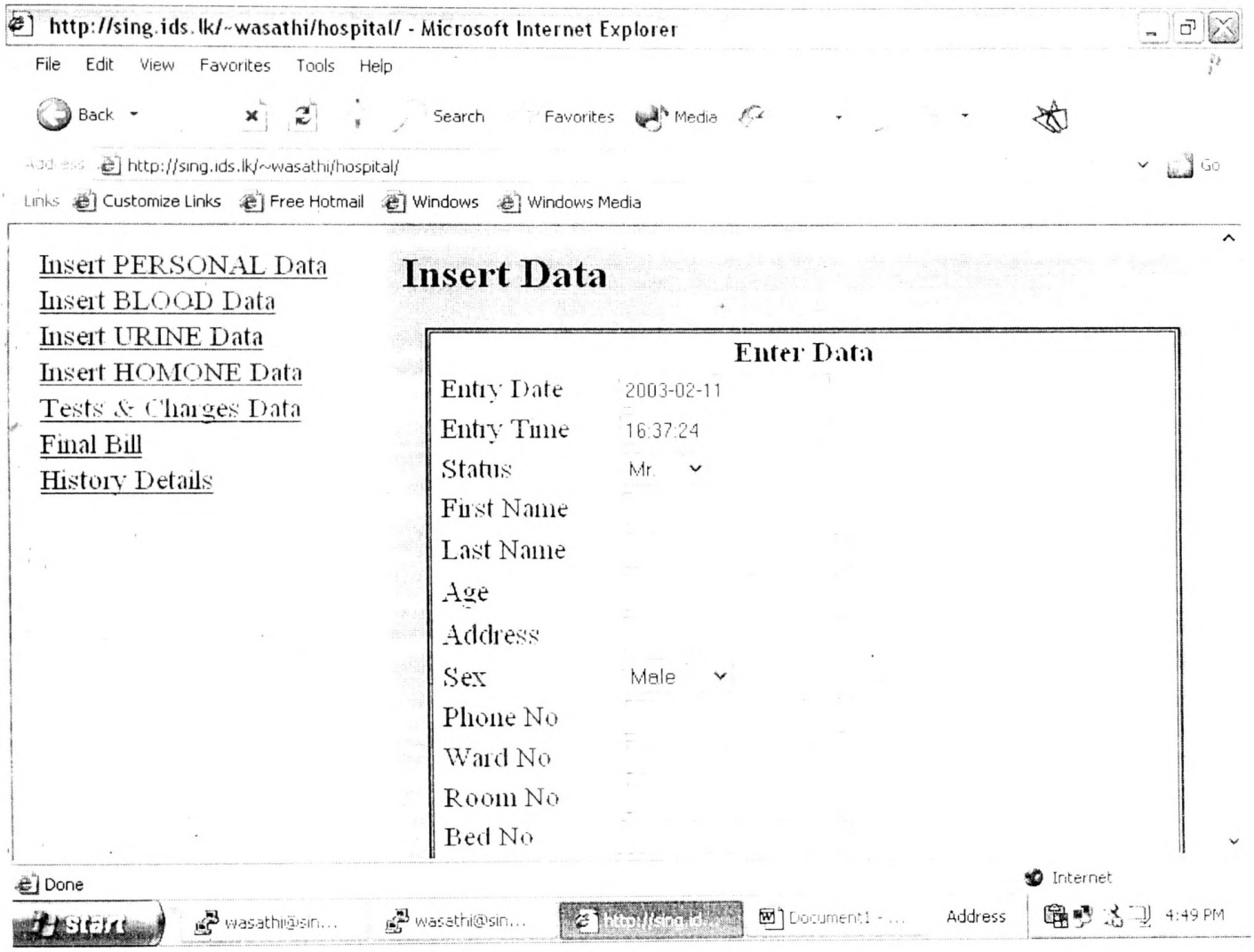


Figure: 5.2.10.2 User Interface 2-Screen

This is the third screen of insert blood data:

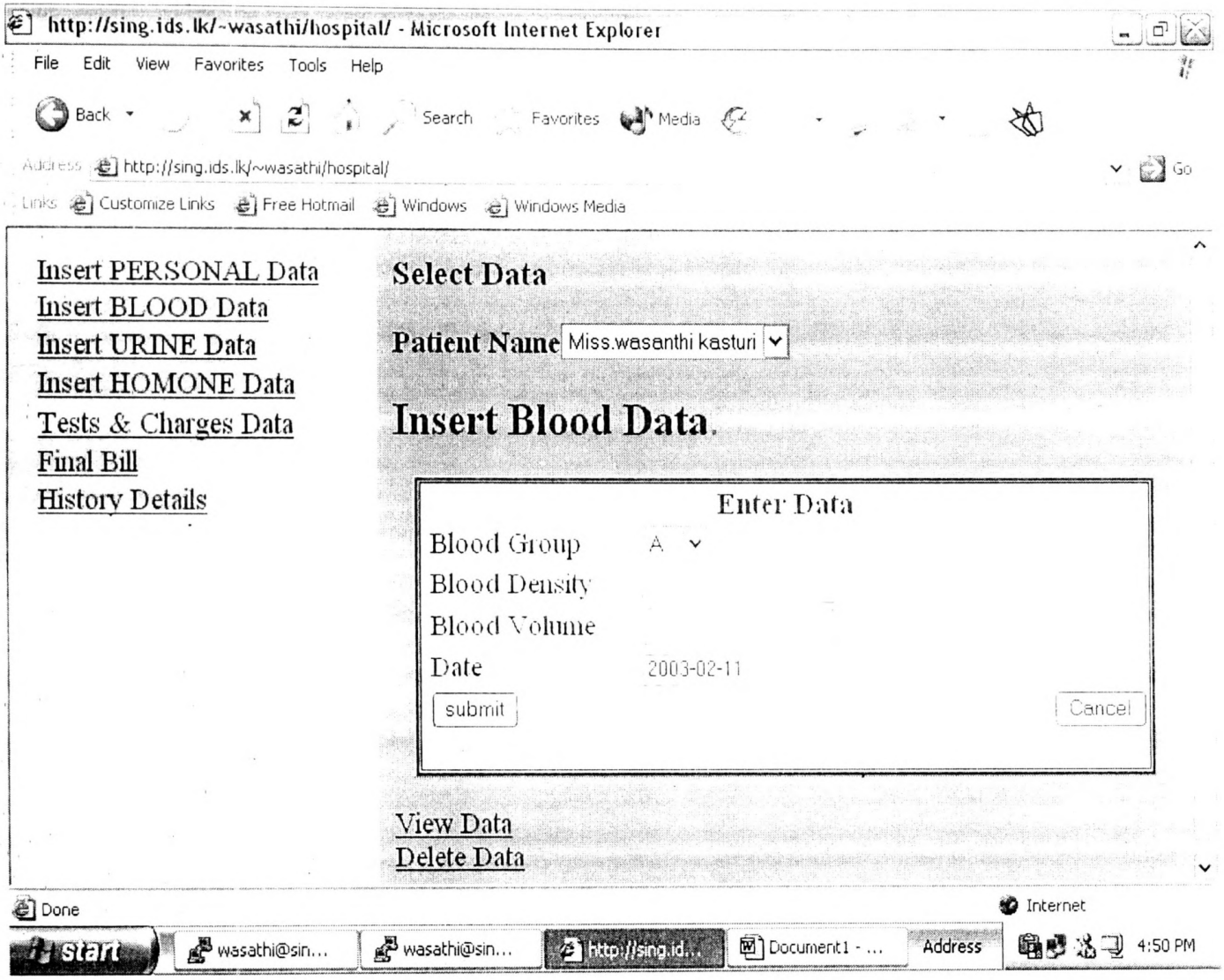


Figure: 5.2.10.3 User Interface 3-Screen

This is the forth screen for insert urine data:

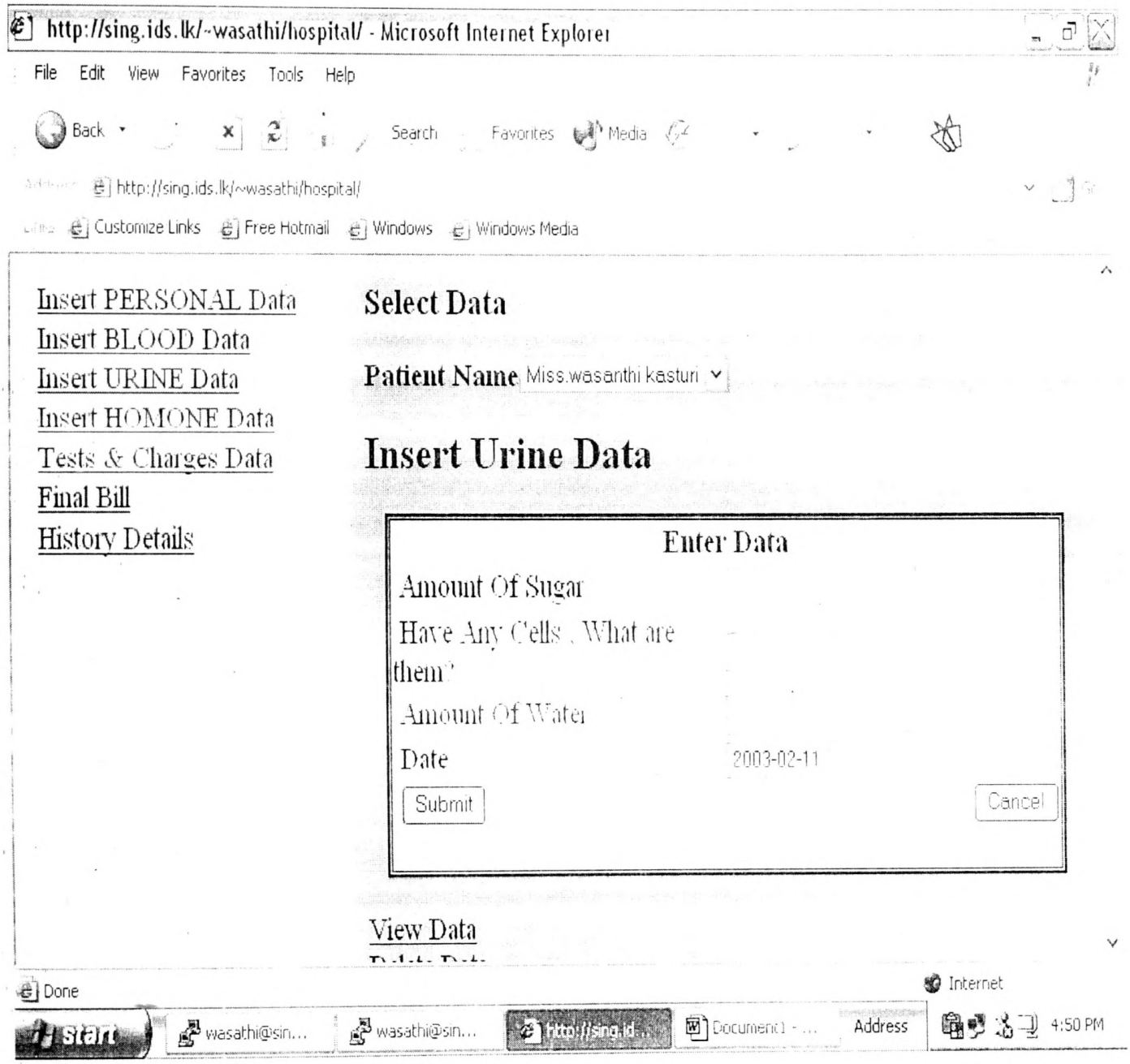


Figure: 5.2.10.4 User Interface 4 -Screen

This is the fifth screen for insert hormones data:

Figure: 5.2.10.5 User Interface 5-Screen

This is the sixth screen for insert hormones data:

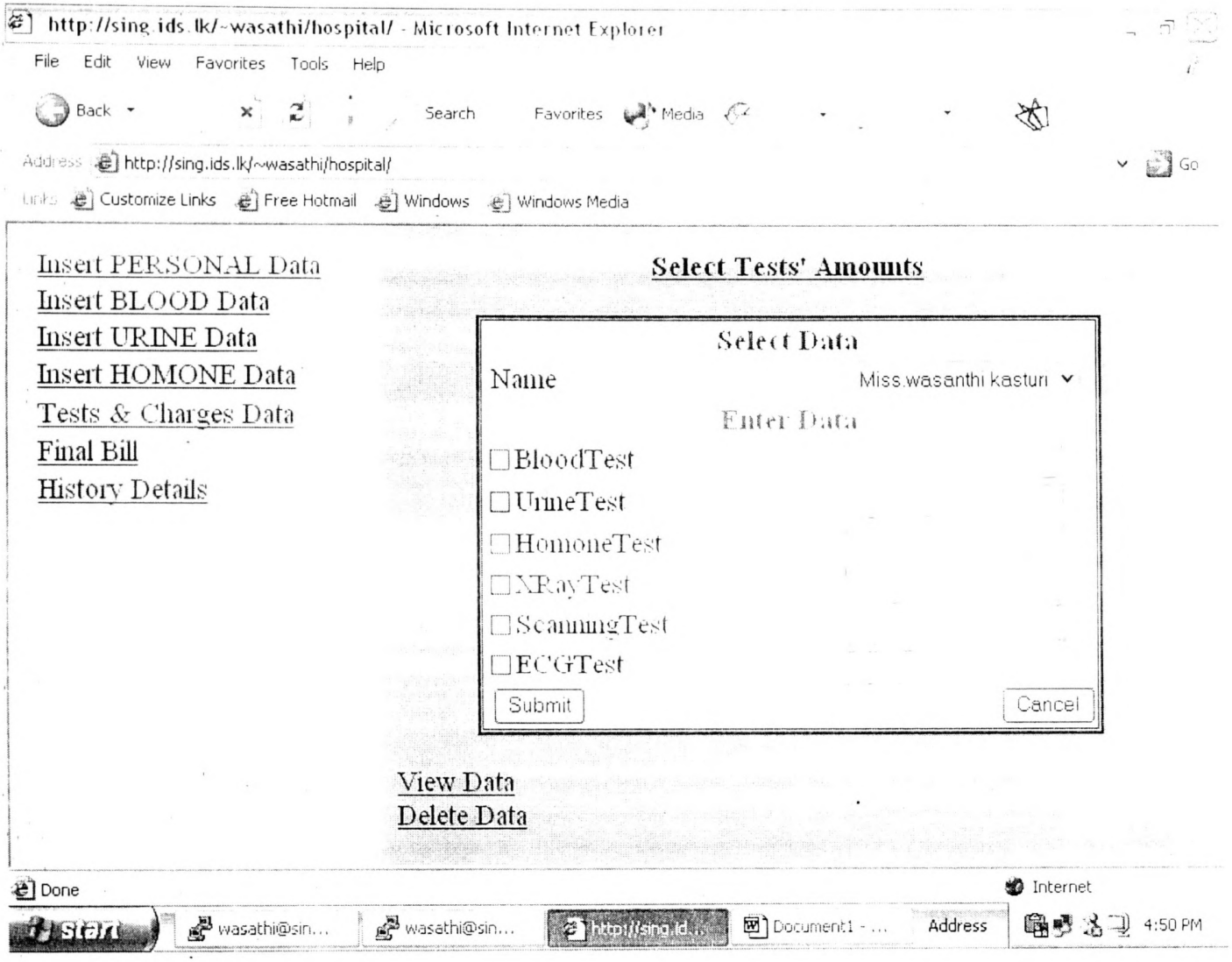


Figure: 5.2.10.6 User Interface 6-Screen

This is the seventh screen for insert hormones data:

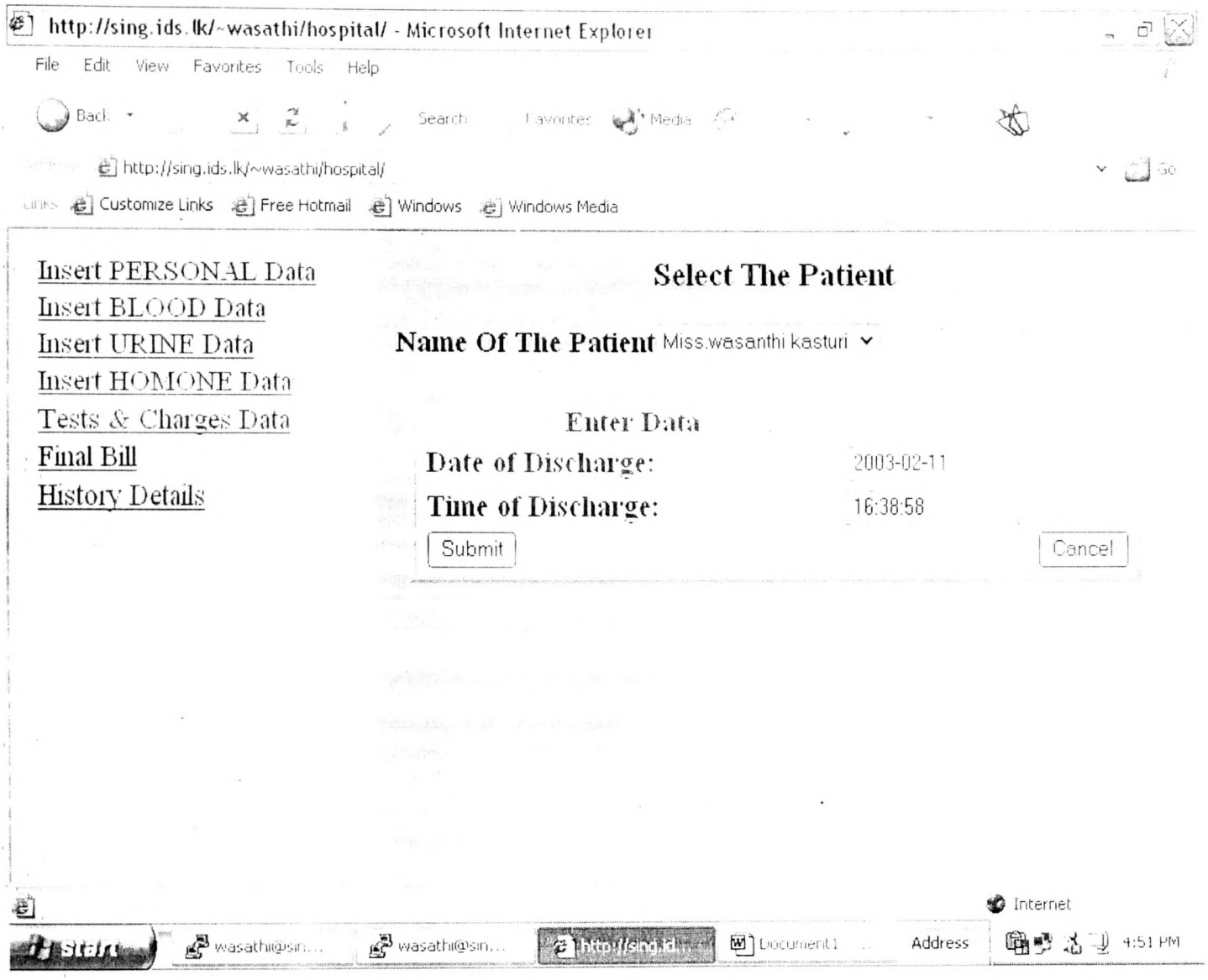


Figure: 5.2.10.7 User Interface 7-Screen

This is the eighth screen for insert hormones data:

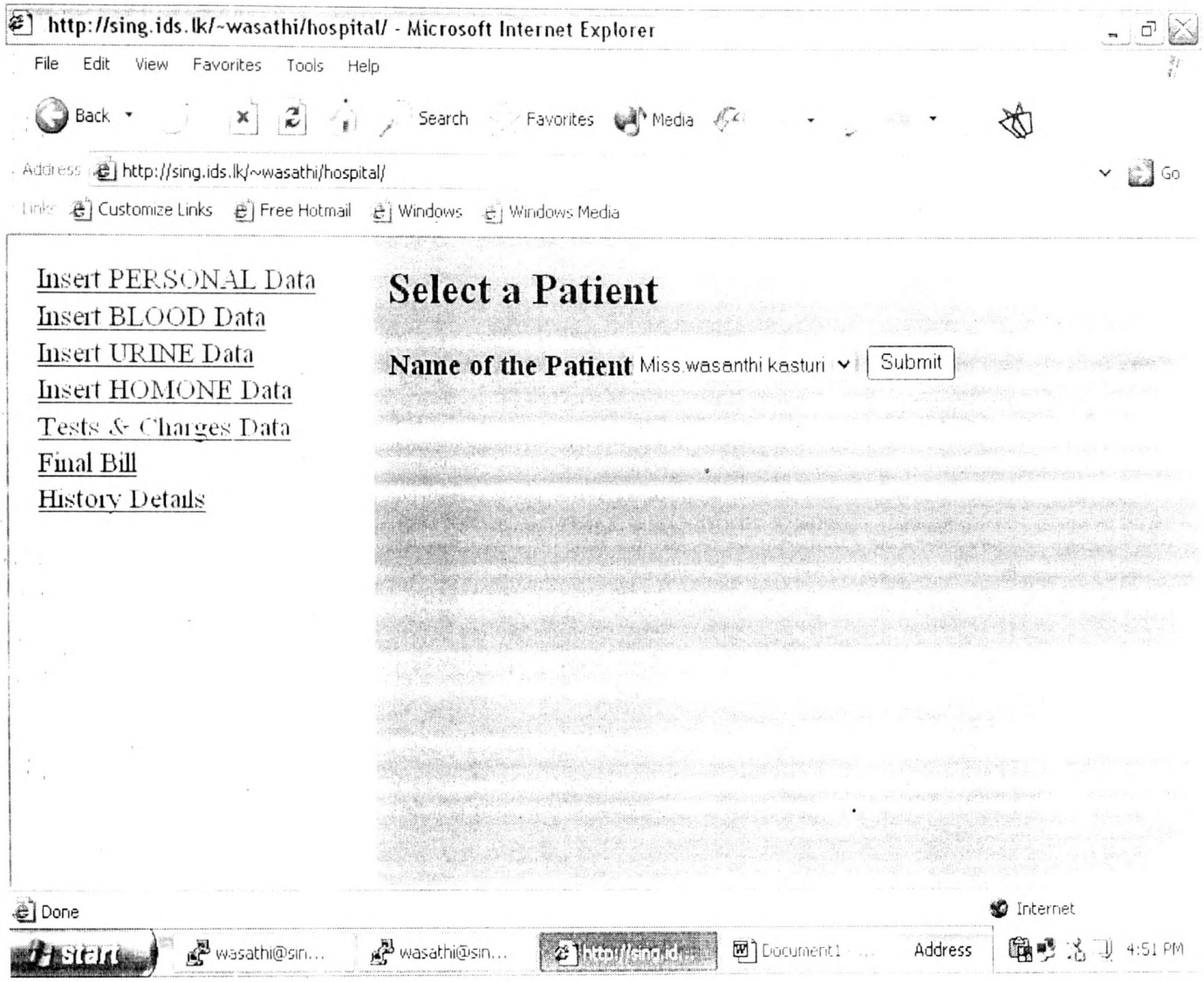


Figure: 5.2.10.8 User Interface 8-Screen

This is the ninth screen for insert hormones data:

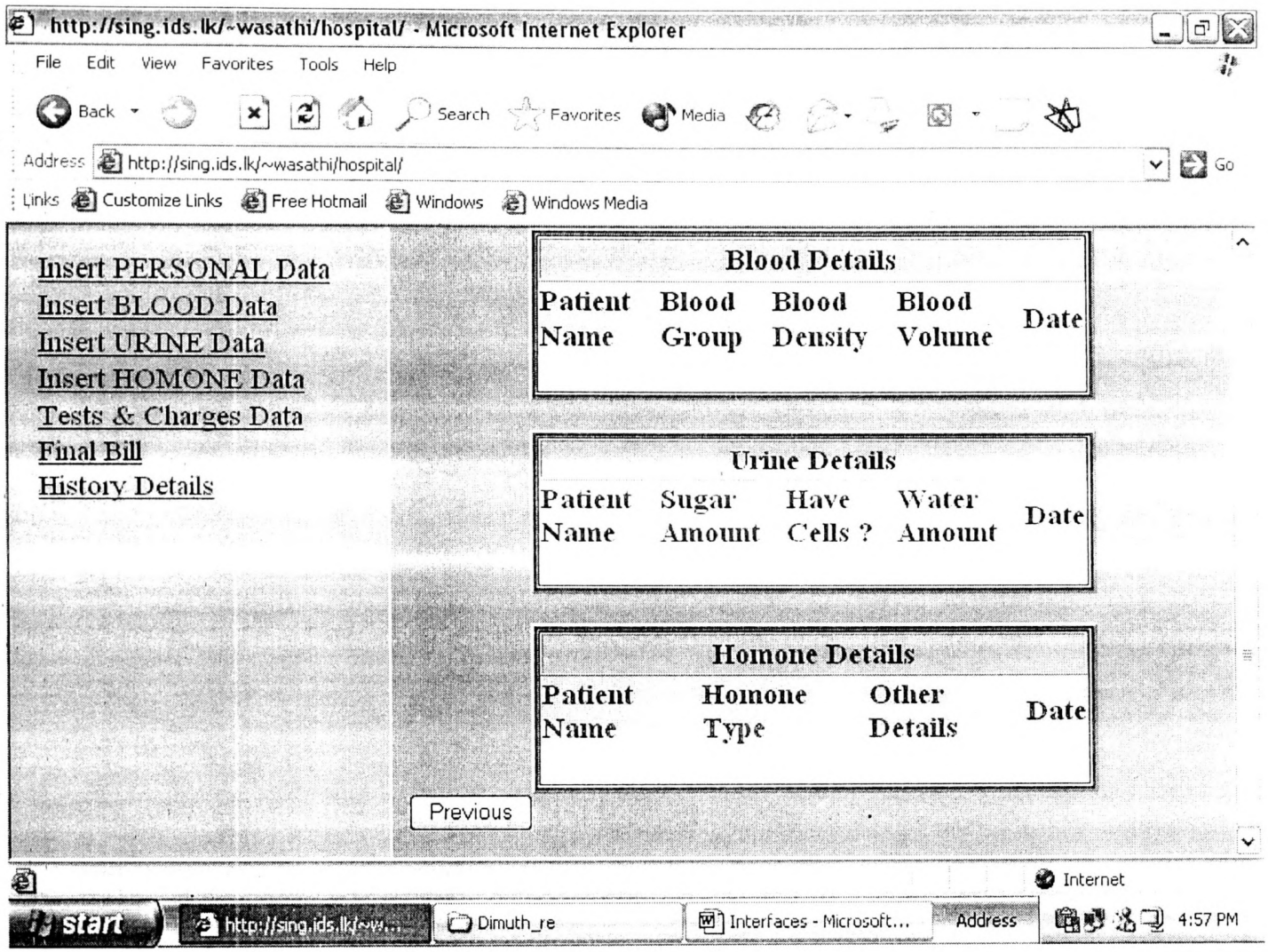


Figure: 5.2.10.9 User Interface 9-Screen

This is the tenth screen for patient final bill:

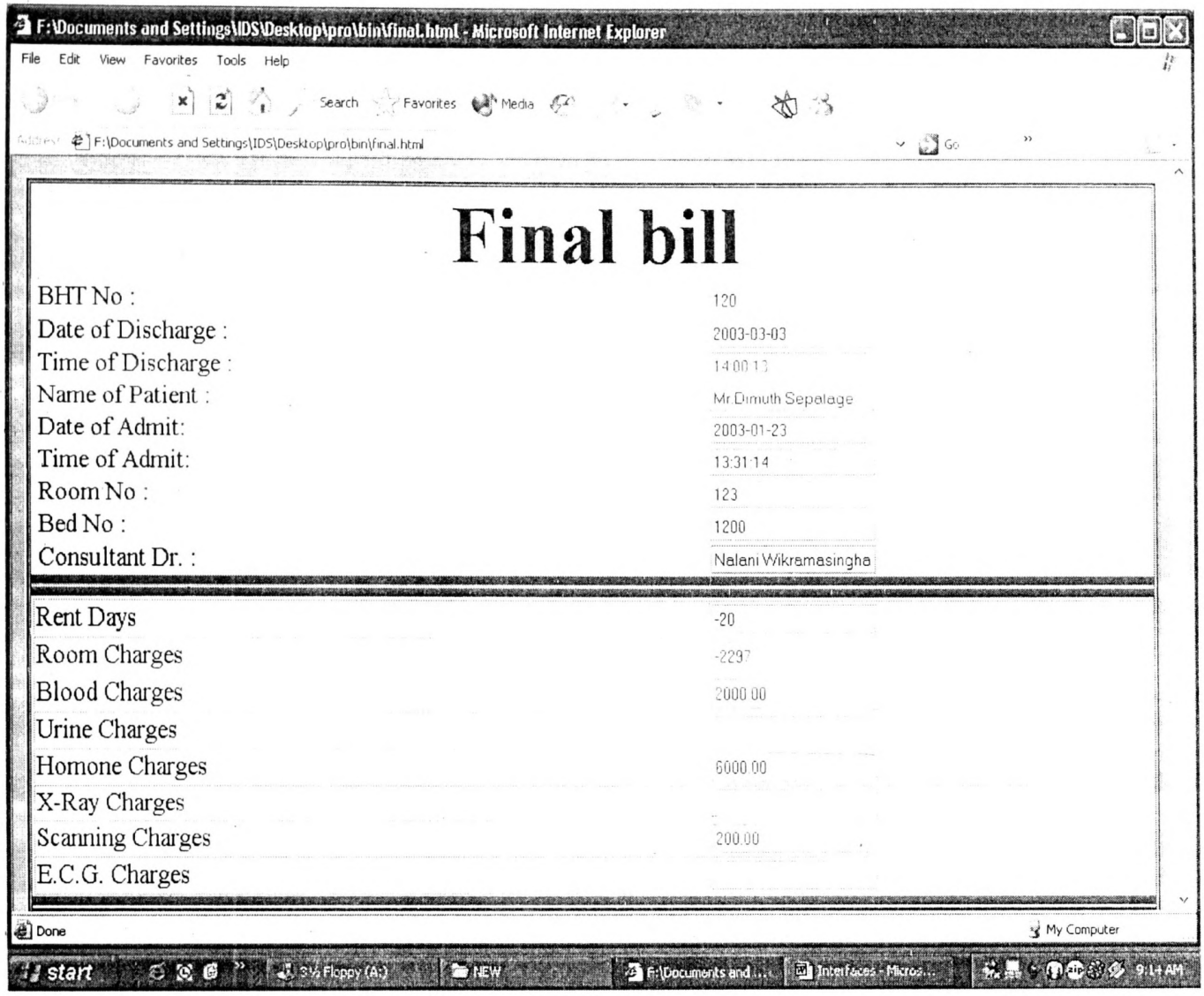


Figure: 5.2.10.10 User Interface 10-Screen

National Digitization Project

National Science Foundation

Institute : Sabaragamuwa University of Sri Lanka

1. Place of Scanning : Sabaragamuwa University of Sri Lanka, Belihuloya

2. Date Scanned : 2017-09-25

3. Name of Digitizing Company : Sanje (Private) Ltd, No 435/16, Kottawa Rd, Hokandara North, Arangala, Hokandara

4. Scanning Officer

Name : B.A.C. Gadarawan

Signature : .....

Certification of Scanning

I hereby certify that the scanning of this document was carried out under my supervision, according to the norms and standards of digital scanning accurately, also keeping with the originality of the original document to be accepted in a court of law.

Certifying Officer

Designation : Librarian

Name : T. N. Neighsoorei

Signature : .....

Mrs. T.N. NEIGHSOOREI
(MSSc.PGD.ASLA,BA)
Librarian
Sabaragamuwa University of Sri Lanka
P.O.Box 02 Belihuloya,Sri Lanka
Tele:0094 45 2280045
Fax:0094 45 2280045

Date : 2017-09-25

"This document/publication was digitized under National Digitization Project of the National Science Foundation, Sri Lanka"