CHAPTER 1
INTRODUCTION

Medical Record library (MRL)is an advanced system that can be used in hospitals for management of patient data and records in a systematic and secure manner which is also easily accessible during emergency and protected from vulnerabilities. MRL can be considered as the 'Brain of the Hospital' since it will contain all records of patients utilizing health services in the hospital. All relevant information's will be included which can be reused or rechecked during occurrence of any kind of disputes and legal necessities. MRL repository will contain details like patient information, doctors, information, Diagnosis information and cure advised, tests conducted, results obtained and many more necessary and important information about the Patient from the time of consultation till discharge

The MRL system access will be limited to authorized personals and officials for security and safety reasons. The doctors and related authorized staff will be registered separately in the MRL system as employees and a username and password will be provided to them through which they can access patient and treatment info for recheck and also if patient history is necessary for treatment for same or different disease diagnosis. Even though the access is made available editing of patient details or treatment records are not possible by the doctors or other staff of the hospital. A formal request should be submitted through proper administrative channels in the hospital stating the reason for changing the records for doing that since these details are considered to be very important and should not be leaked to public.

MRL system makes the functioning of the hospital easier than normal mode of using hard copies and manual work. The time of receiving necessary data, submitting it and processing it and many other activities will be faster. Patients are also benefited from the MRL system since they get timely service from the hospitals. Taking an appointment with doctor to discharge of the patient will be available as record in the MRL making it suitable for billing, transfer and also readmission if necessary. Also, major advantages like collecting necessary documents like birth certificate, death certificate and editing information will be done easier than before. With suitable administrative permissions and approvals from authorities a global information exchange program can be conducted to access patient information during referral to other hospitals or other doctors for faster actions for emergency cases. Rare and critical disease information's and detailing can be utilized for research and

development which will be an added advantage to the whole Medical field. So MRL can play a key role in modern medical field and hospital services

1.1 STATEMENT OF THE PROBLEM

The MRL means Medical Recode Library. MRL is a web application. The patient data and recode in a secure manner. It can easily accessible during the emergency and protected from vulnerabilities.

MRL admin is handling the control of the system.

Superintendent is the main person in the project. They will approval request and give the certificates to need person. It mostly by any higher official of the hospital management.

Librarian is responsible for providing details form the need patient. The middle bone of the MRL system. They request the Superintendent for any change the certificates.

User is checking the all details of patient. It can communicate other modules. It can update the details of patient.

Counter it helps to maintain the details and register that details and verify the information of patient.

MRL has five modules

1. MRL Admin
2. Superintendent
3. Librarian
4. User
5. counter

CHAPTER 2

SYSTEM ANALYSIS

After analysing the requirements of the task to be performed, the next step is to analyse the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

This document plays a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

Spiral model was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The steps for Spiral Model can be generalized as follows:

1. The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.

2. A preliminary design is created for the new system.

3. A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.

4. A second prototype is evolved by a fourfold procedure:

    1. Evaluating the first prototype in terms of its strengths, weakness, and risks.

    2. Defining the requirements of the second prototype.

    3. Planning a designing the second prototype.

    4. Constructing and testing the second prototype.

5. At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer's judgment, result in a less-than-satisfactory final product.

6. The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.

7. The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.

8. The final system is constructed, based on the refined prototype.

9. The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time. In order to develop a new system, a detailed study of the existing system called the System Analysis is necessary. Analysis is the first step in the system development life cycle.

Identifying the need for a new information system and launching an investigation for the required system which best satisfy the exact requirement of the user is the first part of the development activities of a system.

2.1 PRESENT SYSTEM

It is a manual system. But it has a limit and time consuming system.no communication between doctors. When a patient goes to other hospital, they need know all about them in first to end. Some time without knowing history the patient came to death.

2.2 LIMITATIONS OF PRESENT SYSTEM

1. No better coordination
2. Lack of flexibility
3. Delay in actions
4. Lack of communications
5. Difficult to search etc.

2.3 PROPOSED SYSTEM

In this project we introduce a socially relevant system called MEDICAL RECODE LIBRARY (MRL). Anybody can register in MRL and the admin will verify and forward the report of their profile

2.4 ADVANTAGES OF PROPOSED SYSTEM

1. Easy to Access
2. User-friendly
3. Reliable and coordinative
4. Easy to handle
5. Time delay can be avoided
6. Paper works can be reduced in a hospital patient details management process

2.5 FEASIBILITY STUDY

The development of a computer-based system or product is more likely plagued by a resources and difficult delivery data. It is both necessary and prudent to evaluate the feasibility of a project at a time months and years of effort, thousands or millions and untold professional embarrassment can be averted if an ill-convinced system is recognized early in the definition phase.

All the projects are feasible given unlimited resources and infinite time. Unfortunately, the development of a system is more likely affected by scarcity of resources an*d difficult delivery dates. It is both necessary and prudent to evaluate the feasibility of the project in the earliest possible time. Feasibility and risk analysis is related in many ways. If projects risk is great, the feasibility of producing quality software is reduced. Feasibility study is divided into several functions.

During product engineering, we consider following types of feasibility:

2.5.1 Technical Feasibility

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn't want to try to put Star Trek's transporters in their building currently; this project is not technically feasible.

2.5.2 Economic Feasibility

This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility helping decision-makers determine the positive economic benefits to the organization that the proposed project will provide.

2.5.3 Operational Feasibility

This assessment involves undertaking a study to analyse and determine whether and how well the organization's needs can be met by completing the project. Operational feasibility studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SOFTWARE REQUIREMENTS

Operating System	:	Windows 98/XP/NT
Frontend	:	ASP.NET, C#
Back end	:	SQL SEWVER

3.2 HARDWARE REQUIREMENTS

Platform	:	Pentium 4
CPU Speed	:	750MHz
RAM	:	256MB
Monitor	:	17”color
Keyboard	:	Standard Keyboard
Mouse	:	Standard two button or higher

CHAPTER 4

SYSTEM DESIGN

4.1 INPUT DESIGN

Input design is the process of converting user-oriented inputs to computer-based format. It also includes determining the record media, method of input, speed of capture, and entry into system. Input design is the process of converting a user – oriented description of the inputs to a computer-based business system into a programmer-oriented specification. The goal of designing input data is to make data entry as easy, logical and free from errors as possible. Input design is the part of the overall system design, which requires carefully attention. If the data going into the system is incorrect, then the processing and output will magnify these errors. The proposed system satisfies the following input design objectives.

1. cost effective method of input
2. The highest possible level of accuracy.
3. The input is acceptable to and understood by the users staff.

4.2 OUTPUT DESIGN

The normal procedure is to design the outputs in detail first and then to work back to the inputs. The output can be in the form of operational documents, lengthy reports. The input records have to be validated, edited, organized and accepted by the system before being processed to produce the outputs.

Output Objectives:

The output from an information system should accomplish one or more of the following objectives:

1. Convey information about past activities, current status or projections in future.
2. Signal important events, opportunities, problems or warnings.
3. Trigger an action.
4. Confirm the action

4.3 DATA FLOW DIAGRAM

Data Flow Diagram (DFD) is an important tool used by system analyst. DFD provide an overview of what data a system would process, what transformation of data are done, what files are used and where the results flow. The graphical representation of the system makes it a good communication tool between the user and the analyst.

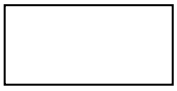
Analysis model help us to understand the relationship between different components in the design. Analysis model shows the user clearly how a system will function. This is the first technical representation of the system.

The analysis modelling must achieve three primary objectives.

1. To establish a basis for creation of software design.
2. To describe what the user requires.
3. To define set of requirements that can be validated once the software us build.

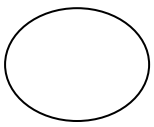
A data flow diagram is a graphical technique that depicts information flow and transforms that are applied as data move from input to output. The DFD is used to represent increasing information flow and functional details. A level 0 DFD also called fundamental system model represents the entire software elements as single bubble with input and output indicated by incoming and outgoing arrow respectively.

4.3.1 Components of Data Flow Diagram



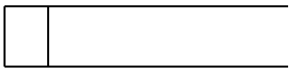
Entities

External entities represent the sources of data that enter the system or the recipients of data that leave the system. They can be duplicated, one or more times on the diagram to avoid line crossing.



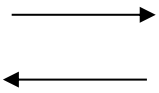
Process

Processes represent activities in which data is manipulated by being stored or retrieved or transformed in some way. A circle represents it. The process will show the data transformation or change. Data coming into a process must be "worked-on" or transformed in some way.



Data Stores

Data stores represent stores of data within the system. Data stores may be long-term files such as sales ledgers, or may be short-term accumulations: For example, batches of documents that are waiting to be processed. Each data store should be given a reference followed by an arbitrary number.



Data Flow

A data flow shows the flow of information from its source to its destination. A line represents a data flow, with arrowheads showing the direction of flow. Information always flows to or from a process and may be written, verbal or electronic. Each data flow may be referenced by the processes or data stores at its head and tail, or by a description of its contents.

4.3.2 Context Level Diagram

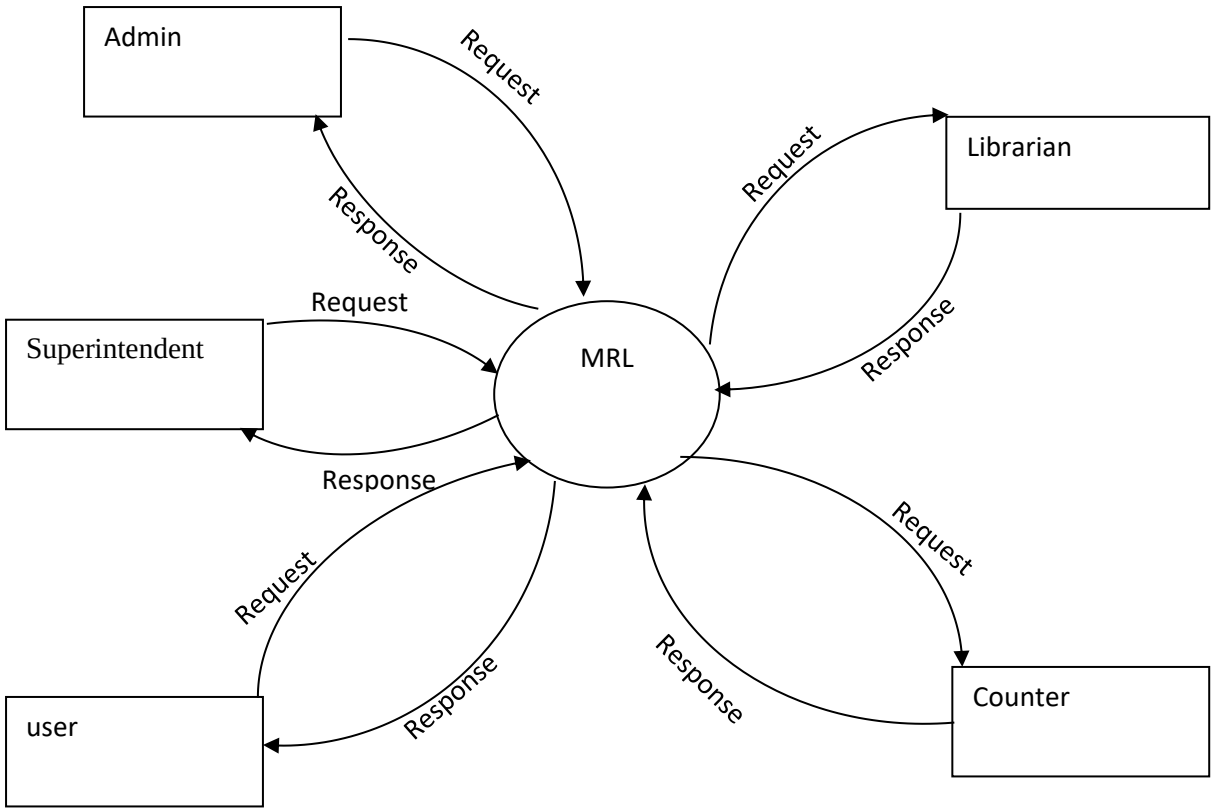


Figure 4.3.2 Context Level Diagrams

4.3.3 Level 1 DFD

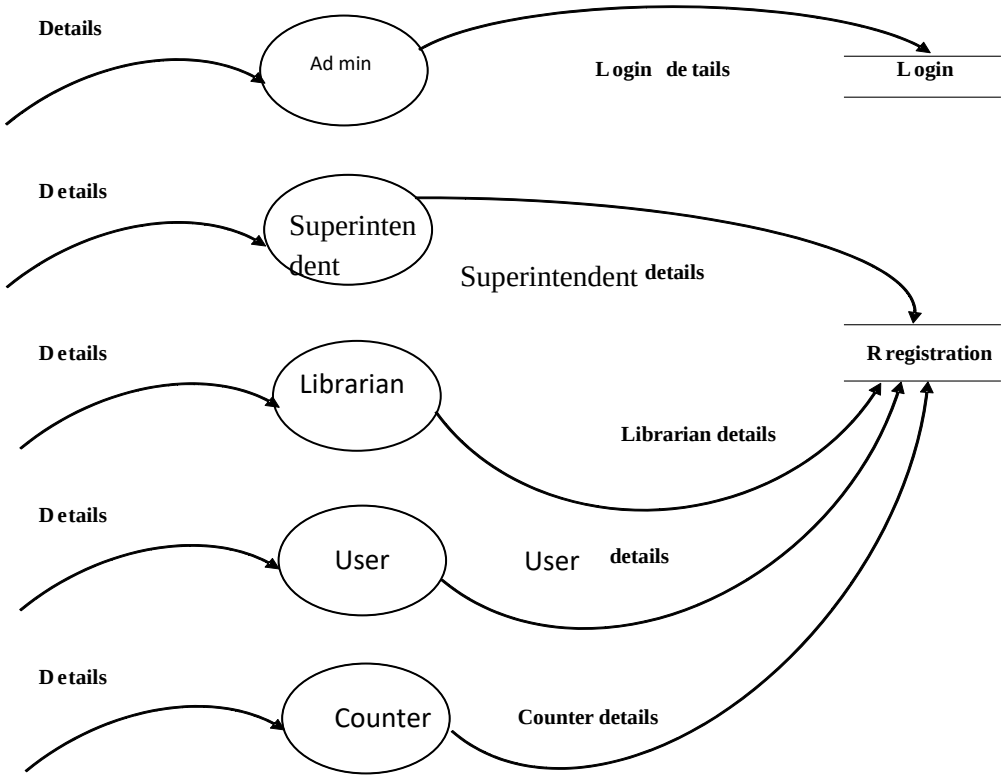


Figure 4.3.3 Level 1 DFD

4.3.4 Level 2 DFD

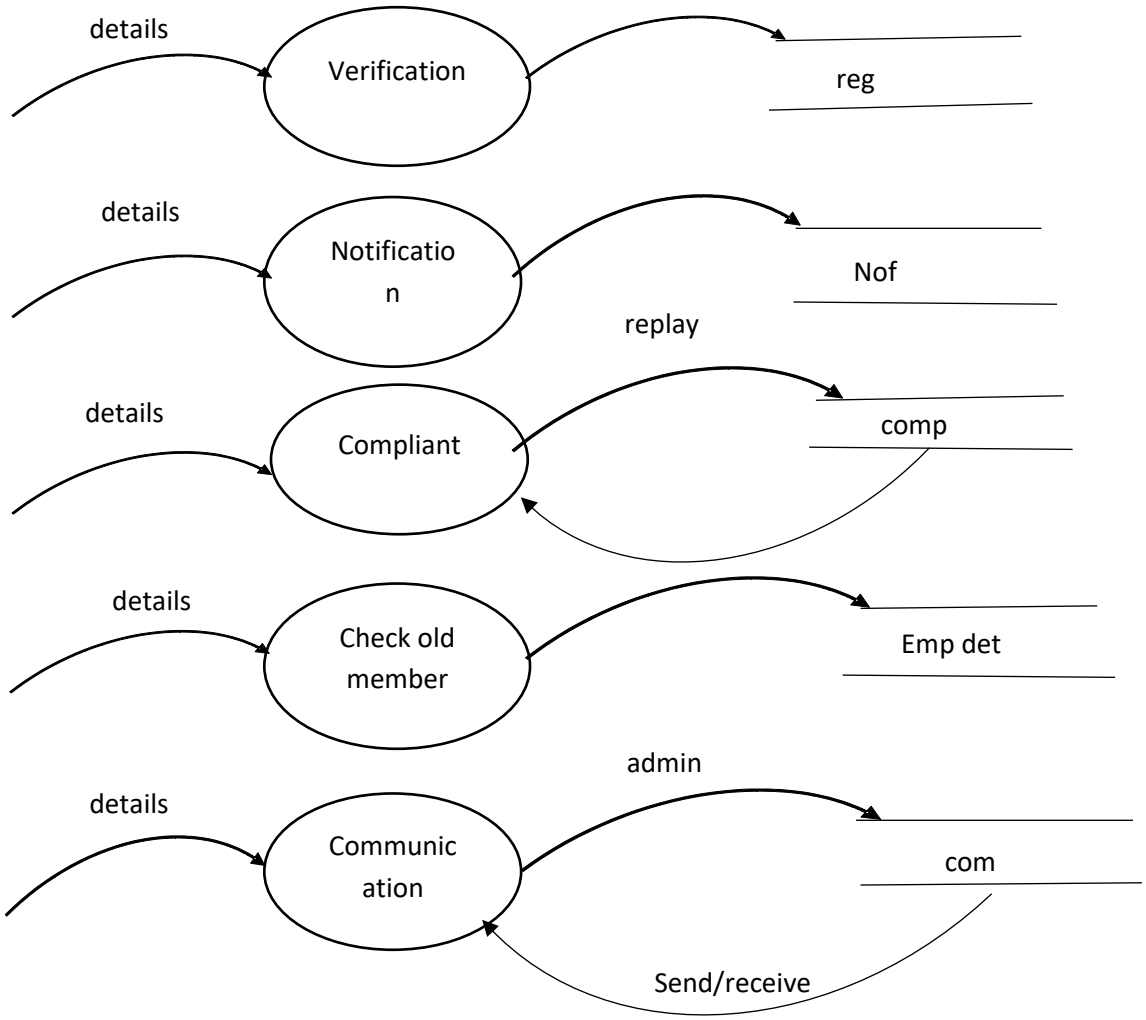


Figure 4.3.4 Level 2 DFD for Admin

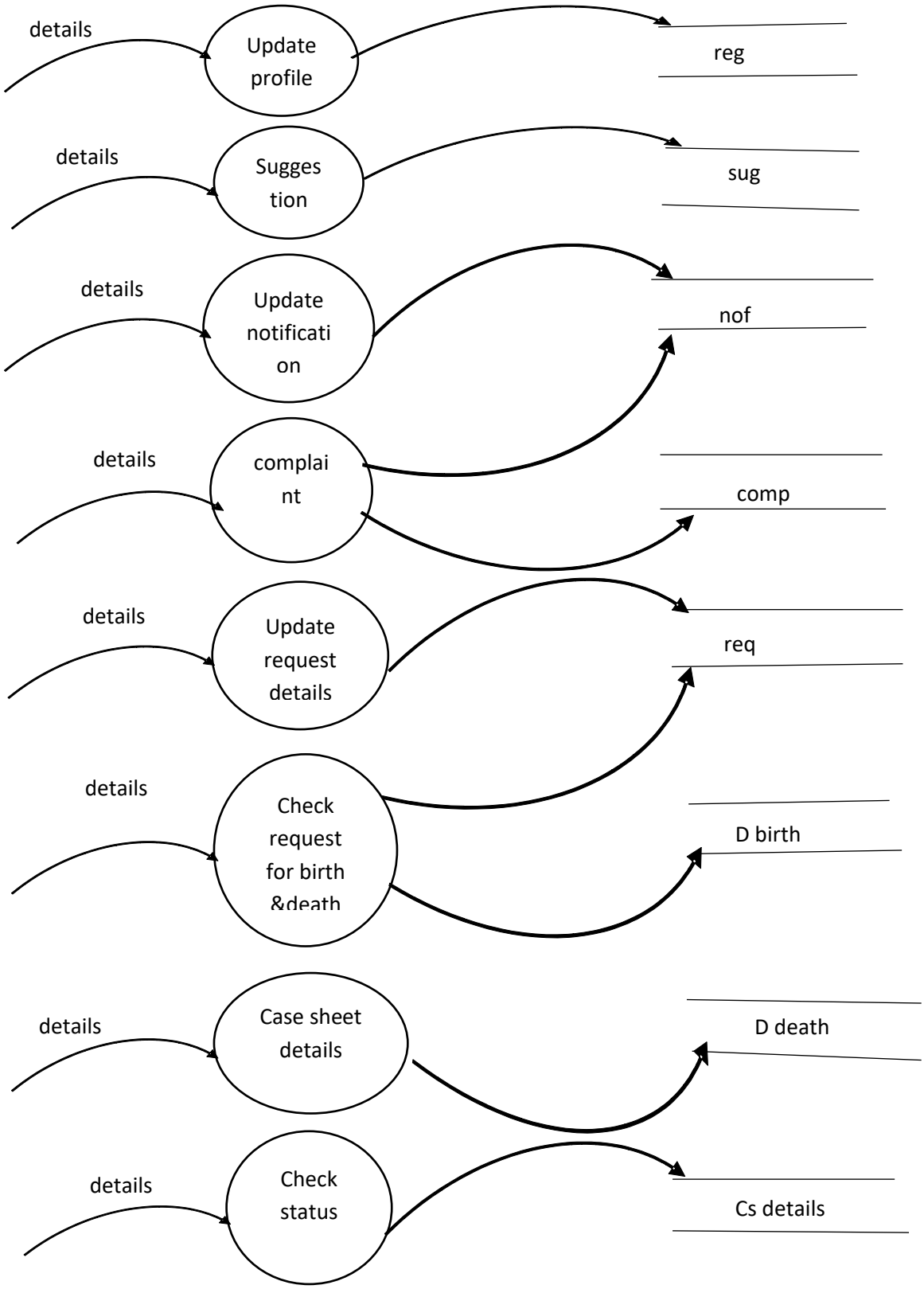


Figure 4.3.5 Level 2 DFD of Superintendent

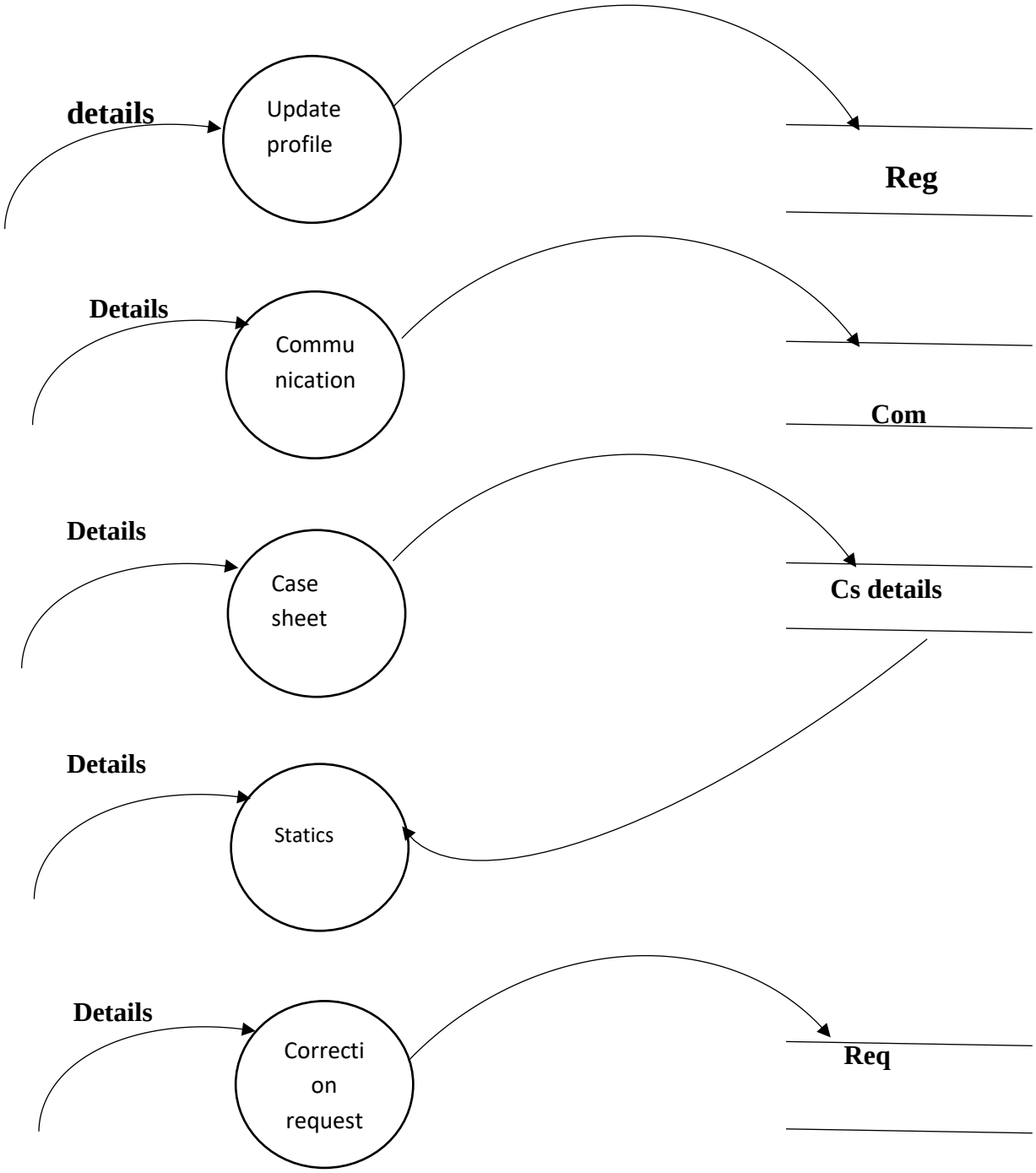


Figure 4.3.6 Level 2 of Librarian

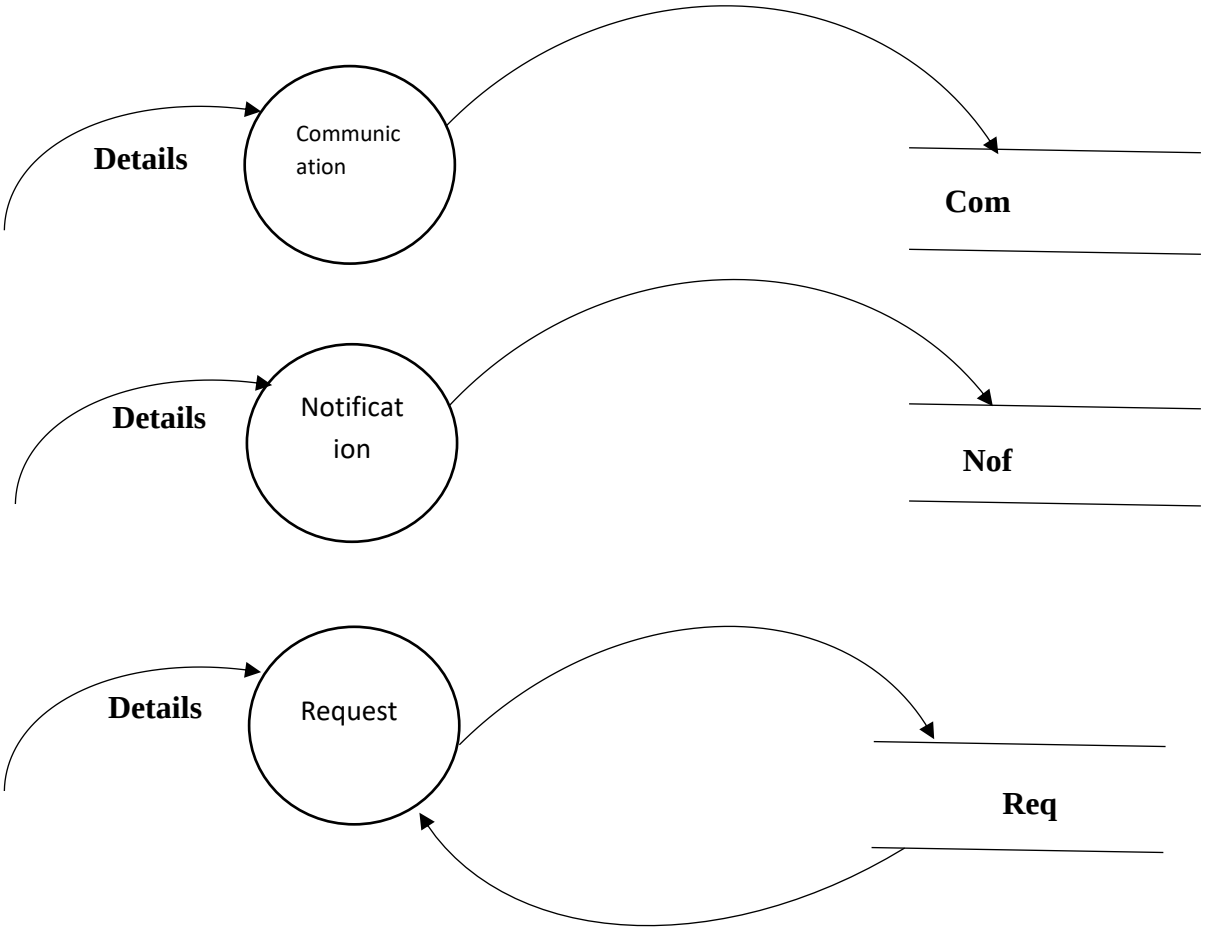


Figure 4.3.7 Level 2 DFD for User

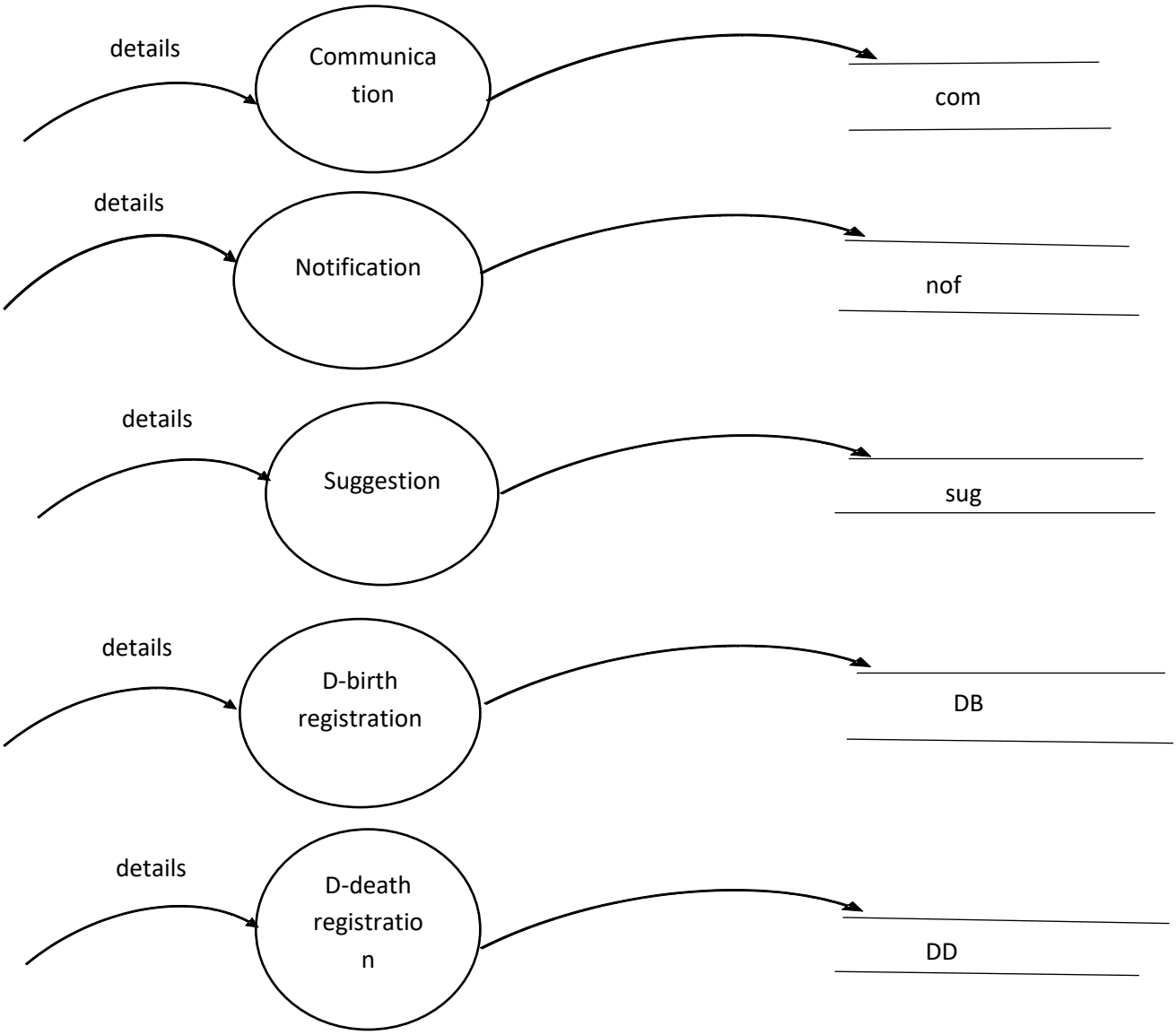


Figure 4.3.8 Level 2 DFD For Counter

4.4 ER DIAGRAM

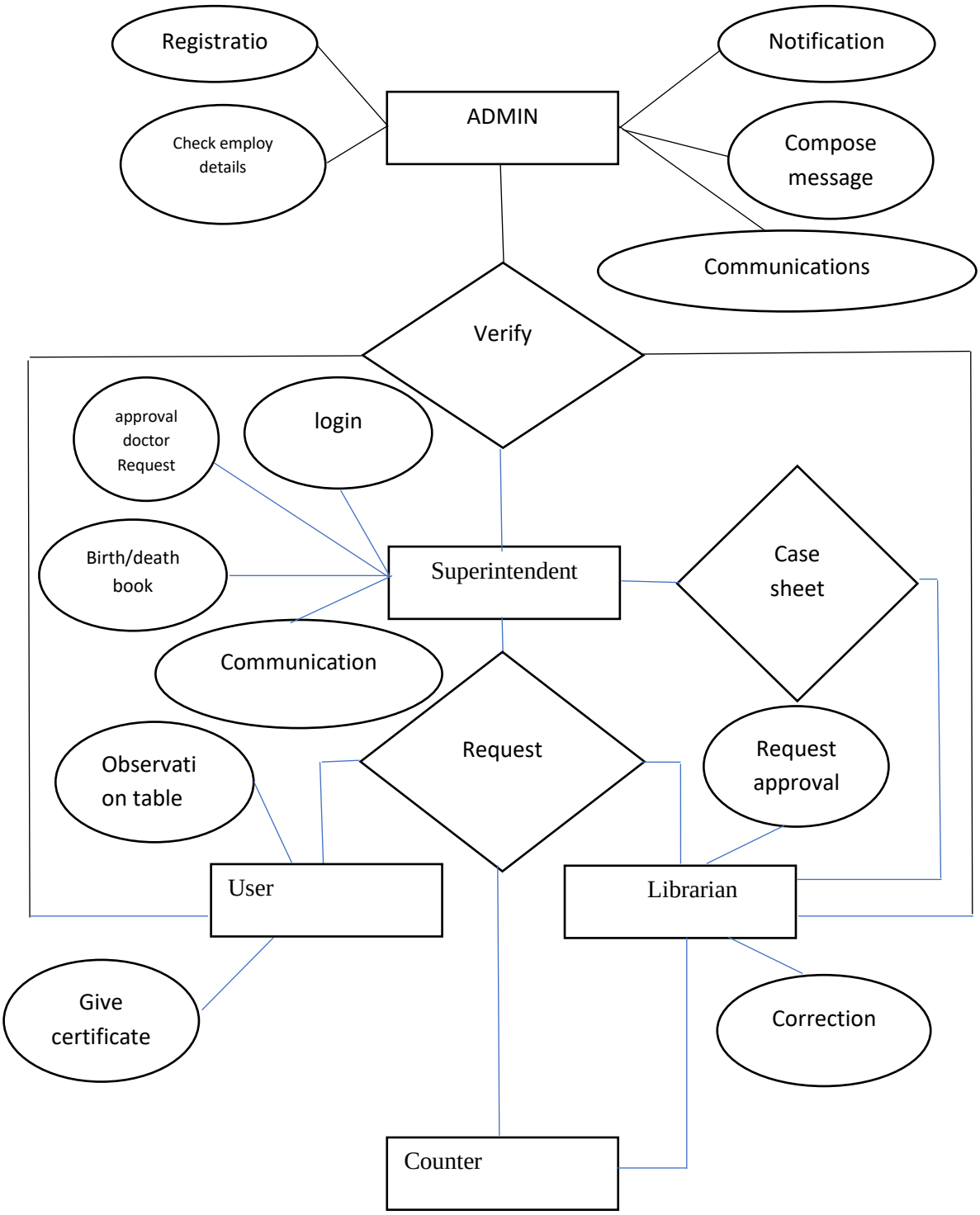


Figure 4.3.1 Level ER diagram

4.5 DATABASE DESIGN

Table 4.5.1: *login*

Field	Type	Size	Constraints
Username	Varchar	20	not null
Password	Varchar	20	not null
user type	Varchar	20	not null

Table 4.5.2*: Registration*

Field	Type	Size	Constraints
Sl.No	Numeric	20	primary key
Name	Varchar	50	Not null
Username	Varchar	50	Not null
Gender	Varchar	10	Not null
Address	Varchar	50	Not null
Place	Varchar	50	Not null
Post	Varchar	50	Not null
Pin code	Numeric	10	Not null
District	Varchar	50	Not null
State	Varchar	20	Not null
Mobile no	Numeric	12	Not null
Email	Varchar	50	Not null
Reg.No	Numeric	20	Not null
DOB	Varchar	20	Not null
Password	Varchar	20	Not null
DOJ	Varchar	18	Not null

Table 4.5.3: *Request*

Field	Type	Size	Constraints
Req_id	Numeric	20	Primary key
Req_type	Varchar	50	Not null
Req_form	Varchar	50	Not null
Req_to	Varchar	50	Not null
Req-time	Numeric	10	Not null
Req_date	Varchar	50	Not null
Req-sub	Varchar	50	Not null
Remark	Varchar	50	Not null

Table 4.5.4: *Feed back*

Field	Type	Size	Constraints
FB_id	Numeric	20	Primary key
User_id	Numeric	10	Foreign key
FB_type	Varchar	20	Not null
Date	Varchar	10	Not null
Time	Varchar	10	Not null
Feedback	Varchar	100	Not null

Table 4.5.5: D_ *Death*

Field	Type	Size	Constraints
Ip.no	Numeric	20	Primary key
Time	Varchar	25	Not null
Date	Varchar	20	Not null
Name	Varchar	20	Not null
Gender	Varchar	20	Not null
Diseases	Varchar	50	Not null
DOA	Numeric	20	Not null
Address	Varchar	30	Not null
Phone no	Numeric	10	Not null
Remark	Numeric	50	Not null

Table 4.5.6: *CS_Details*

Field	Type	Size	Constraints
Ip.no	Numeric	20	Primary key
Name	varchar	20	Not null
Gender	Varchar	10	Not null
DOB	Varchar	50	Not null
DOA	Varchar	50	Not null
DOG/DOD	Varchar	50	Not null
Address	Varchar	50	Not null
Phone no	Numeric	10	Not null
Diseases	Varchar	50	Not null
Current status	Varchar	50	Not null
Scanned details	Varchar	50	Not null
Doctor name	varchar	20	Not null
MRL_code	Numeric	20	Not null

Table 4.5.7**:** *Emp_details*

Field	Type	Size	Constraints
Emp_id	Numeric	20	Primary key
Sl.no	Numeric	10	Not null
Name	Varchar	50	Not null
Destination	Varchar	20	Not null
DOJ	Varchar	50	Not null
DOR	Varchar	50	Not null
Description	Varchar	50	Not null

Table 4.5.8: *Communication*

Field	Type	Size	Constraints
C_id	Numeric	20	Primary kay
Sender type	Varchar	30	Not null
Sender name	Varchar	20	Not null
R_type	Varchar	20	Not null
R_name	Varchar	20	Not null
Sub	Varchar	50	Not null
Msg	Varchar	100	Not null
Send date	Numeric	10	Not null
Replay	Varchar	100	Not null

Table 4.5.9: *D_brith*

Field	Type	Size	Constraints
Ip_no	Numeric	20	Primary key
Time	Varchar	10	Not null
DOB	Varchar	20	Not null
Mother's name	Varchar	20	Not null
Father's name	Varchar	20	Not null
Name	Varchar	20	Not null
Gender	Varchar	10	Not null
Address	Varchar	100	Not null
Phone no	Numeric	10	Not null

4.6 NORMALIZATION

Normalization is the process of reorganizing data in a database so that it meets two basic requirements: (1) There is no redundancy of data (all data is stored in only one place), and (2) data dependencies are logical (all related data items are stored together). Normalization is important for many reasons, but chiefly because it allows databases to take up as little disk space as possible, resulting in increased performance. Normalization is also known as data normalization. The three main types of normalization are listed below. Note: "NF" refers to "normal form."

1. 1NF-First normal form
2. 2NF-Second normal form
3. 3NF-Third normal form

4.6.1 First Normal Form

First normal form (1NF) sets the fundamental rules for database normalization and relates to a single table within a relational database system. Normalization follows three basic steps, each building on the last. The first of these is the first normal form.
The first normal form states that:

1. Every column in the table must be unique
2. Separate tables must be created for each set of related data
3. Each table must be identified with a unique column or concatenated columns called the primary key
4. No rows may be duplicated
5. No columns may be duplicated
6. No row/column intersections contain a null value
7. No row/column intersections contain multivalued fields

4.6.2 Second Normal Form

Second normal form (2NF) is the second step in normalizing a database. 2NF builds on the first normal form (1NF).

Normalization is the process of organizing data in a database so that it meets two basic requirements:

1. There is no redundancy of data (all data is stored in only one place).
2. Data dependencies are logical (all related data items are stored together).

A 1NF table is in 2NF form if and only if all of its non-prime attributes are functionally dependent on the whole of every candidate key.

4.7 DESIGN OF EACH SUBSYSTEM (MODULAR DIAGRAM)

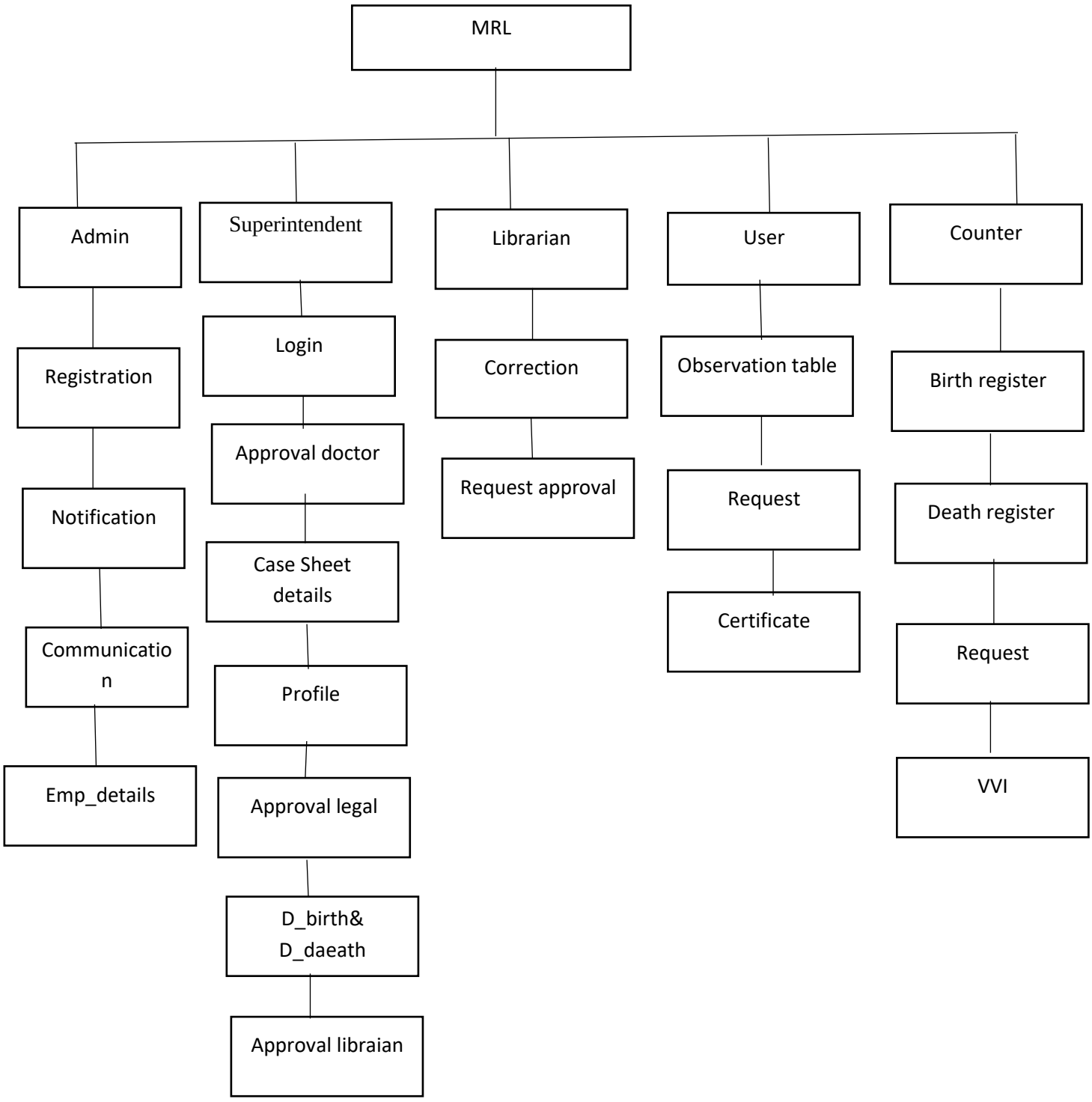


Figure 4.7.1 Design of Each Sub System

4.8 UML DIAGRAMS

The UML is one of the most existing tools in the world of system developed today. The UML enables system builders to create blue prints that capture their visions in a standard and easy way to understand and communicate them to others. UML has got various components like Use Case Diagram, Class Diagram, and Sequential Diagram. There is no class diagram in our project.

4.8.1 Use case diagram for admin

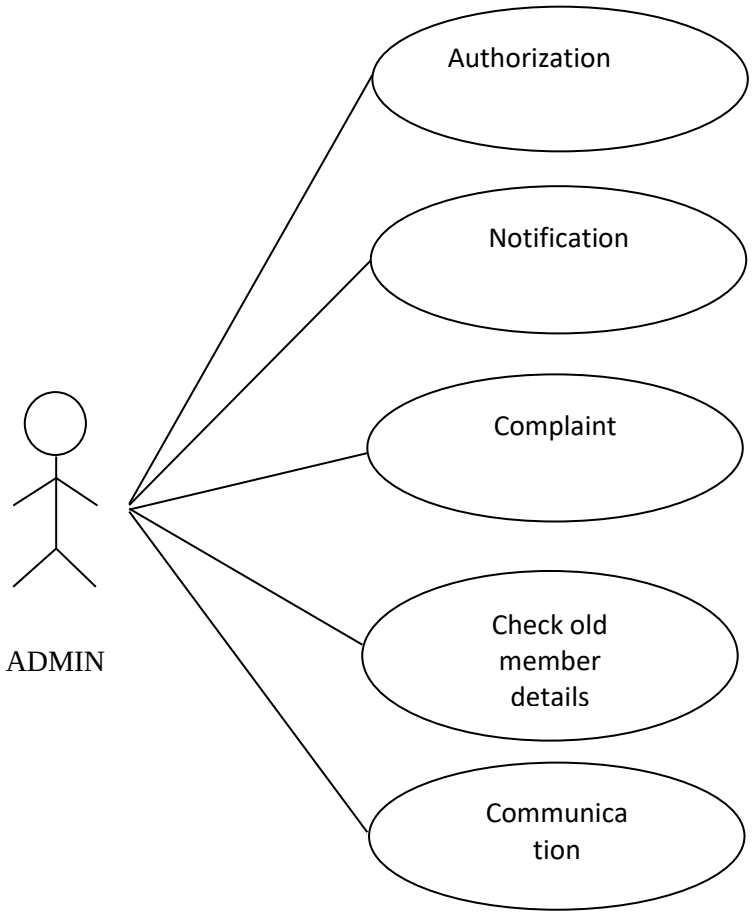


Figure 4.8.1.1 Use Case Diagram for Admin

4.8.1 Use case diagram for Superintendent

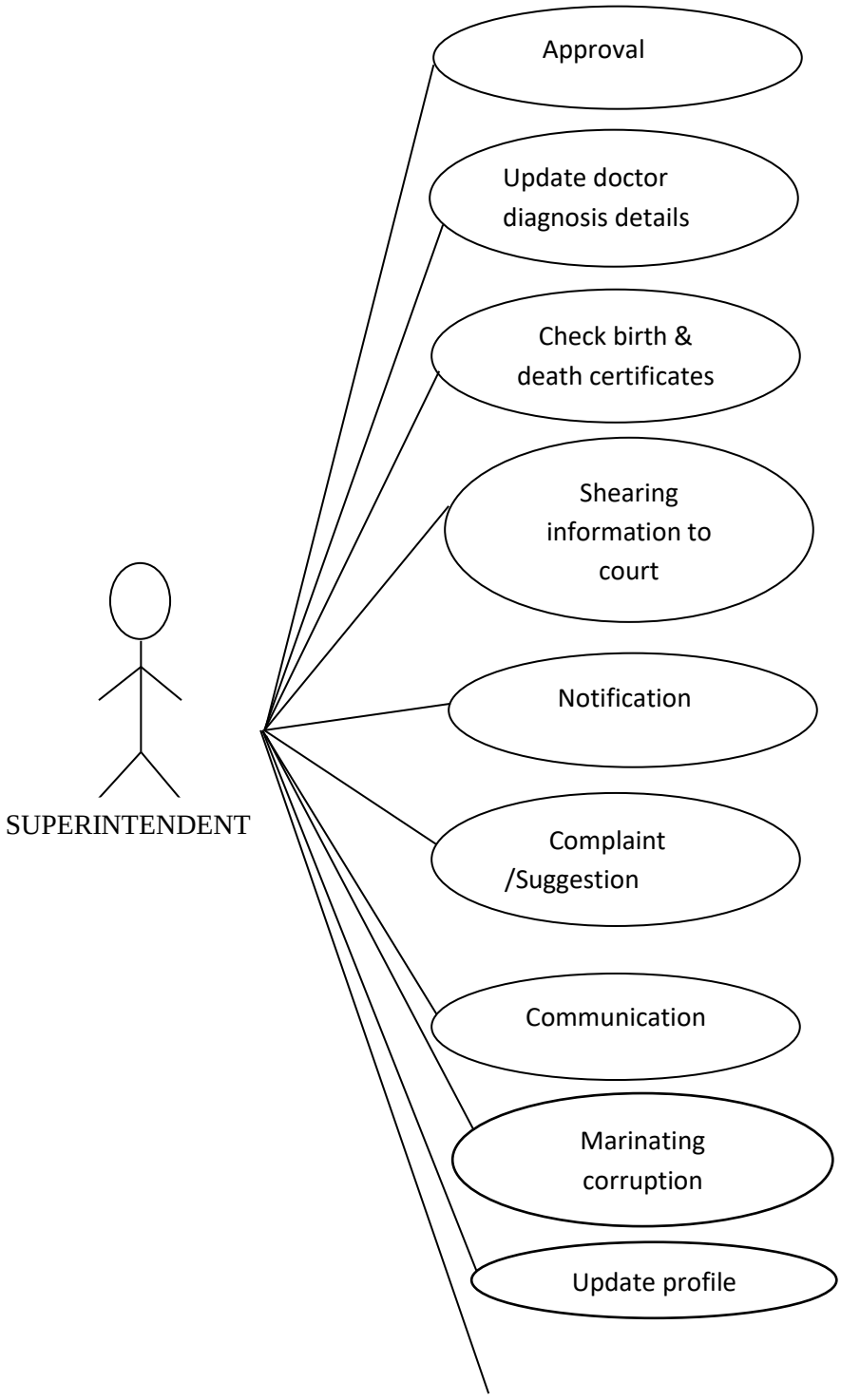


Figure 4.8.1.2 Use Case Diagram for Superintendent

4.8.1 Use case diagram for Librarian

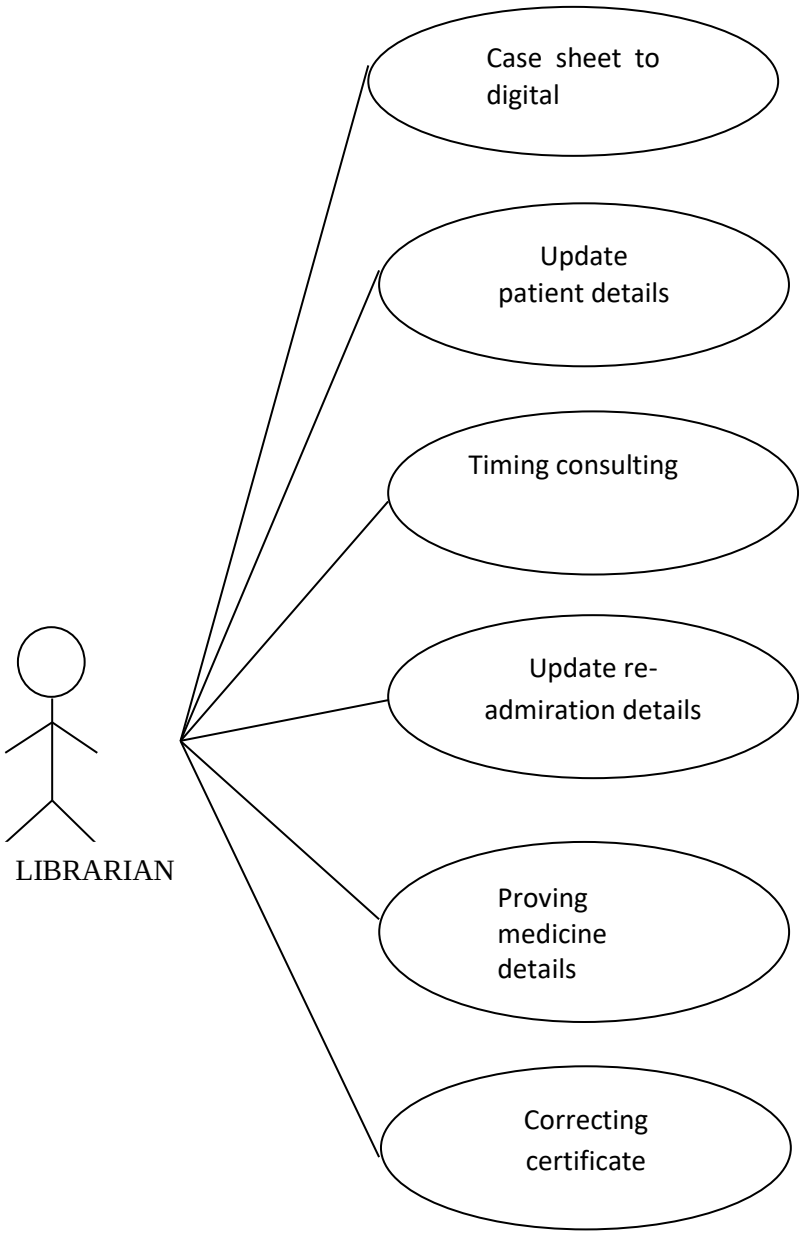


Figure 4.8.1.3 Use Case Diagram for Librarian

4.8.1 Use case diagram for user

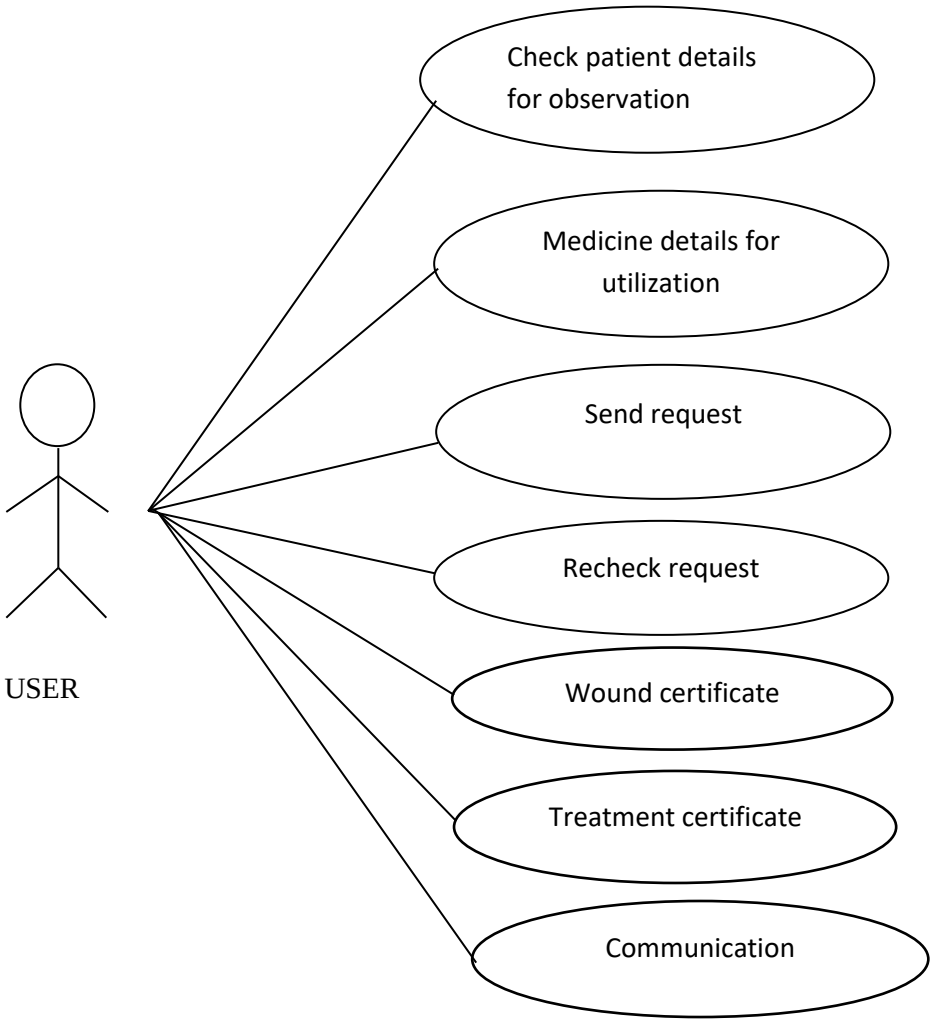


Figure 4.8.1.4Use Case Diagram for User

4.8.1Use case diagram for Counter

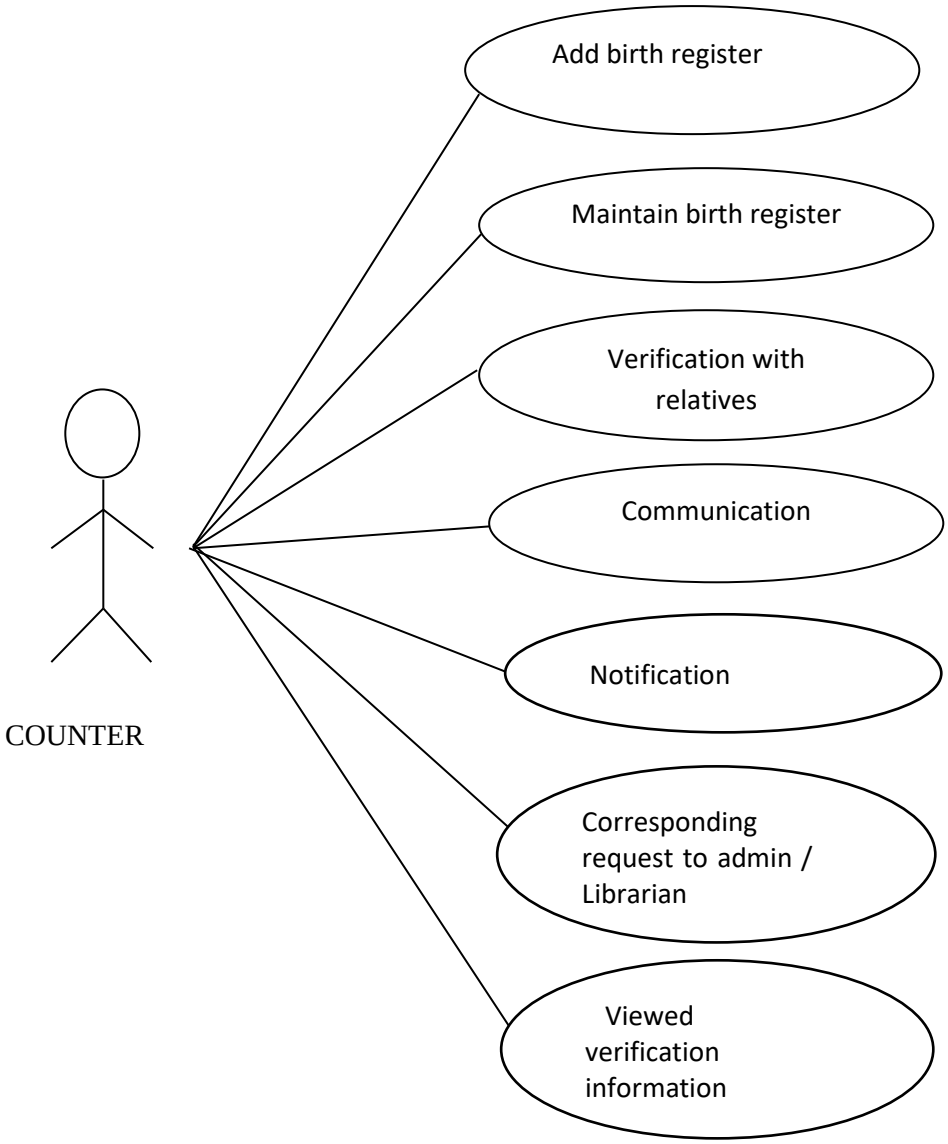


Figure 4.8.1.5 Use Case Diagram for Counter

4.7.2 Class Diagram

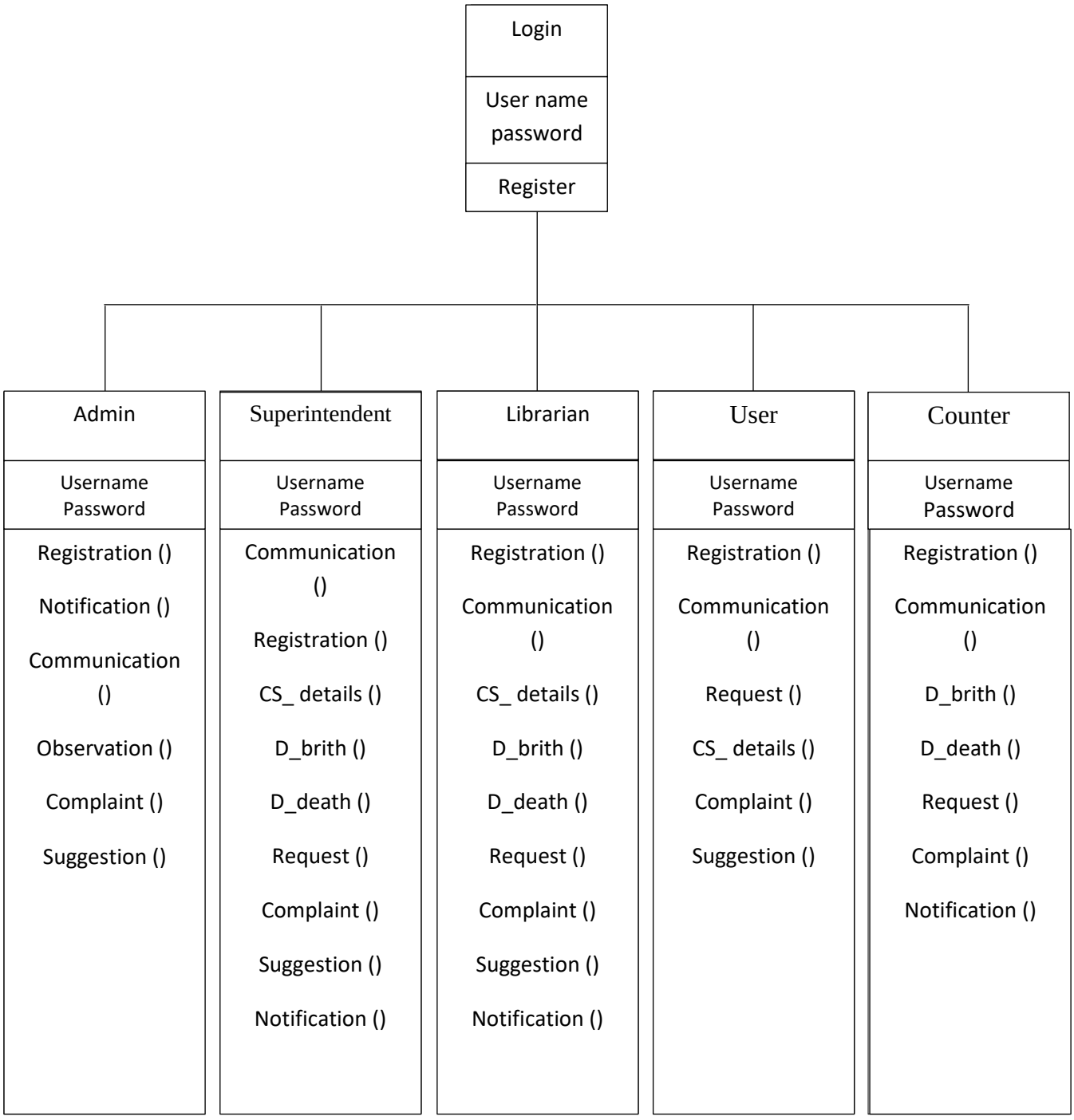


Figure 4.7.2.1 Class Diagram

4.8.3 Sequence Diagram

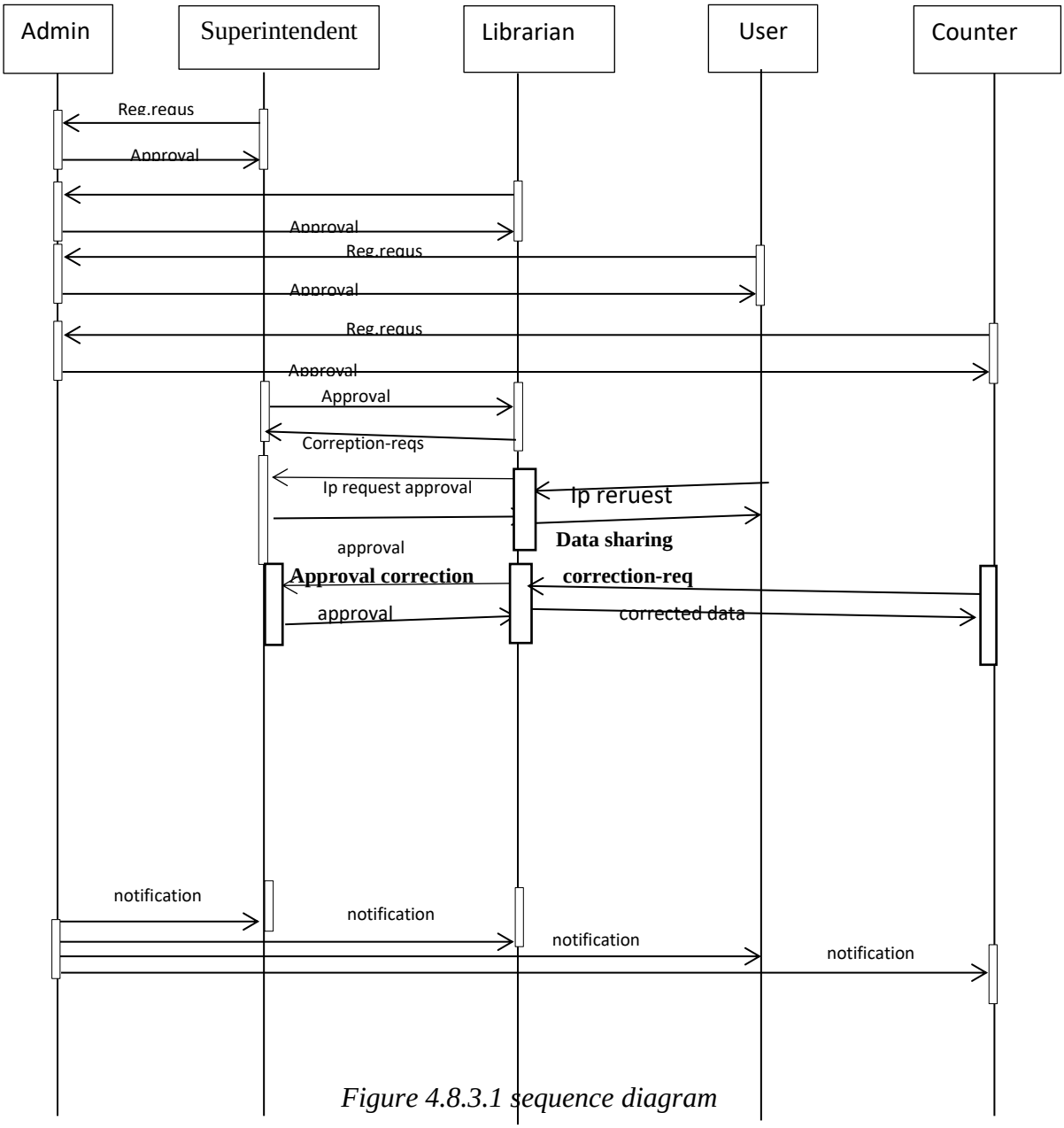


Figure 4.8.3.1 sequence diagram

CHAPTER 5

CODING

5.1 FEATURES OF LANGUAGE

Code editor

Like any other IDE, it includes a code editor that supports syntax highlighting and code completion using IntelliSense for not only variables, functions and methods but also language constructs like loops and queries. IntelliSense is supported for the included languages, as well as for XML and for Cascading Style Sheets and JavaScript when developing web sites and web applications. Autocomplete suggestions are popped up in a modeless list box, overlaid on top of the code editor. In Visual Studio 2008 onwards, it can be made temporarily semi-transparent to see the code obstructed by it. The code editor is used for all supported languages. The Visual Studio code editor also supports setting bookmarks in code for quick navigation. Other navigational aids include collapsing code blocks and incremental search, in addition to normal text search and regex search. The code editor also includes a multi-item clipboard and a task list. The code editor supports code snippets, which are saved templates for repetitive code and can be inserted into code and customized for the project being worked on. A management tool for code snippets is built in as well. These tools are surfaced as floating windows which can be set to automatically hide when unused or docked to the side of the screen.

Debugger

Visual Studio includes a debugger that works both as a source-level debugger and as a machine-level debugger. It works with both managed code as well as native code and can be used for debugging applications written in any language supported by Visual Studio. In addition, it can also attach to running processes and monitor and debug those processes. If

source code for the running process is available, it displays the code as it is being run. If source code is not available, it can show the disassembly. The Visual Studio debugger can also create memory dumps as well as load them later for debugging. Multi-threaded programs are also supported. The debugger can be configured to be launched when an application running outside the Visual Studio environment crashes.

ASP.NET

ASP.NET is an open source server-side Web application framework designed for Web development to produce dynamic Web pages. It was developed by Microsoft to allow programmers to build dynamic web sites, web applications and web services. It was first released in January 2002 with version 1.0 of the .NET Framework, and is the successor to Microsoft's Active Server Pages (ASP) technology. ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET language. The ASP.NET SOAP extension framework allows ASP.NET components to process SOAP messages.ASP.NET is in the process of being re-implemented as a modern and modular web framework, together with other frameworks like Entity Framework. The new framework will make use of the new open-source .NET Compiler Platform (Code-name "Roslyn") and be cross platform. The project is called "ASP.NET vNext"..NET Framework (pronounced dot net) is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library known as Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs written for .NET Framework execute in a software environment (as contrasted to hardware environment), known as Common Language Runtime (CLR), and an application virtual machine that provides services such as security, memory management, and exception handling. FCL and CLR together constitute .NET Framework.

C#

C# can be written with any text editor, like Windows Notepad, and then compiled with the C# Command line compiler, csc.exe, which comes with the .NET framework. However, most people prefer to use an IDE (Integrated Development Environment), and Microsoft offers several options for this. Their flagship is Visual Studio, which can be used to work on every possible aspect of the .NET framework. This product is very advanced, and comes in several editions. Visual Studio is not exactly cheap, and might even be too advanced for hobby programmers. With .NET framework 2.0, Microsoft introduced the so-called Express versions, targeted at hobby programmers and people wanting to try .NET, and they continued this tradition with the later release of .NET 3.0 and 3.5. The Express versions only work for one language, like C# or VB.NET, and miss some of the really advanced features of Visual Studio. However, they are free and will work just fine for learning the languages, which is why we will use it for this tutorial.

C# is a modern, general-purpose, object-oriented programming language developed by Microsoft and approved by Ecma and ISO. C# was developed by Anders Hejlsberg and his team during the development of .Net Framework. C# is designed for Common Language Infrastructure (CLI), which consists of the executable code and runtime environment that allows use of various high-level languages to be used on different computer platforms and architectures.

SQL Server

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet). There are at least a dozen different editions of Microsoft SQL Server aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing

applications with many concurrent users. Its primary query languages are T-SQL and ANSI SQL.

Data storage

Data storage is a database, which is a collection of tables with typed columns. SQL Server supports different data types, including primary types such as Integer, Float, Decimal, Char (including character strings), Varchar (variable length character strings), binary (for unstructured blobs of data), Text (for textual data) among others. The rounding of floats to integers uses either Symmetric Arithmetic Rounding or Symmetric Round Down (Fix) depending on arguments: SELECT Round(2.5, 0) gives 3.
Microsoft SQL Server also allows user-defined composite types (UDTs) to be defined and used. It also makes server statistics available as virtual tables and views (called Dynamic Management Views or DMVs). In addition to tables, a database can also contain other objects including views, stored procedures, indexes and constraints, along with a transaction log. A SQL Server database can contain a maximum of 231 objects, and can span multiple OS-level files with a maximum file size of 260 bytes (1 exabyte).[43] The data in the database are stored in primary data files with an extension .mdf. Secondary data files, identified with a .ndf extension, are used to store optional metadata. Log files are identified with the .ldf extension.

Buffer management

SQL Server buffers pages in RAM to minimize disc I/O. Any 8 KB page can be buffered in-memory, and the set of all pages currently buffered is called the buffer cache. The amount of memory available to SQL Server decides how many pages will be cached in memory. The buffer cache is managed by the Buffer Manager. Either reading from or writing to any page copies it to the buffer cache. Subsequent reads or writes are redirected to the in-memory copy, rather than the on-disc version. The page is updated on the disc by the Buffer Manager only if the in-memory cache has not been referenced for some time. While writing pages back to disc, asynchronous I/O is used whereby the I/O operation is

done in a background thread so that other operations do not have to wait for the I/O operation to complete. Each page is written along with its checksum when it is written. When reading the page back, its checksum is computed again and matched with the stored version to ensure the page has not been damaged or tampered with in the meantime.

Data retrieval

The main mode of retrieving data from an SQL Server database is querying for it. The query is expressed using a variant of SQL called T-SQL, a dialect Microsoft SQL Server shares with Sybase SQL Server due to its legacy. The query declaratively specifies what is to be retrieved. It is processed by the query processor, which figures out the sequence of steps that will be necessary to retrieve the requested data. The sequence of actions necessary to execute a query is called a query plan. There might be multiple ways to process the same query. For example, for a query that contains a joint statement and a select statement, executing join on both the tables and then executing select on the results would give the same result as selecting from each table and then executing the join, but result in different execution plans. In such case, SQL Server chooses the plan that is expected to yield the results in the shortest possible time. This is called query optimization and is performed by the query processor itself.

Database Design

A database consists of a collection of interrelated data. A database provides an environment that is both convenient and efficient for people to store and retrieve information. Database, any collection of data, or information that is specially organized for rapid search and retrieval by a computer. Databases are structured to facilitate the storage operations. Databases can be stored on magnetic disk or tape, optical disk. A database consists of a file or a set of files. The information in these files may be broken down into records, each of which consists of one or more fields. Fields are the basic units of data storage, and each field typically contains information pertaining to one aspect or

commands, users can rapidly search, rearrange, group, and select the fields in many records to retrieve or create reports on particular aggregates of data.

5.2 FUNCTIONAL DESCRIPTION

Login

This process is used to identify the type of user requesting to sign in and then redirect the user to his/her corresponding home page. The user interface for this process prompts the user to enter his/her username and password. The method reads the username and password, checks if the username and password belong to a registered member of the system or not and takes action accordingly. If the username and password do not match, the function displays a message saying, “Invalid Username and Password”.

MRL Admin Module

1. Registration
2. notification
3. Communication
4. Check Emp-details

Superintendent Module

1. Login
2. Approval-doctor
3. Check case sheet details
4. Update profile
5. D-birth certificate
6. D-death certificate

Librarian Module

1. Check corruption
2. Request-approval
3. Communicate to librarian

User Module

1. Check observation table
2. Request to librarian
3. Give certificates
4. Communication

Counter Module

1. Birth registration
2. Death registration
3. Request
4. View verified information

5.3 SYSTEM CODING

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;


public partial class HoReg : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    int rid;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
    }
    protected void Button2_Click(object sender, EventArgs e)
    {

    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        Label1.Visible = true;
        Label2.Visible = true;
        TextBox2.Visible = true;
        TextBox3.Visible = true;

        con.Open();
        cmd = new SqlCommand("select uid from reg order by uid DESC", con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            rid = Convert.ToInt32(rdr["uid"]);
            rid = rid + 1;
        }
        else
        {
            rid = 1;
        }
        cmd.Dispose();
        con.Close();
        if (DropDownList1.SelectedItem.Text == "Doctor")
        {
```

```
            Label3.Text = "";
            Label1.Visible = true;
            Label2.Visible = true;
            TextBox2.Visible = true;
            TextBox3.Visible = true;
            rfdep.Visible = true;
            rfdesi.Visible = true;
            con.Open();
            d1 = DateTime.Now.ToString("dd-MM-yyyy");
            cmd = new SqlCommand("insert into reg values(" + rid + ",'" +
TextBox1.Text + "',null,'" + DropDownList1.SelectedItem.Text + "','" +
TextBox3.Text + "','" + TextBox2.Text + "','" + TextBox4.Text + "'," +
TextBox5.Text + ",'" + TextBox6.Text + "',0)", con);
            cmd.ExecuteNonQuery();
            con.Close();
            TextBox1.Text = "";
            TextBox2.Text = "";
            TextBox3.Text = "";
            TextBox4.Text = "";
            TextBox5.Text = "";
            TextBox6.Text = "";

            Label3.Text = "Succesfully Added Your Information.. Pls Note Your
Verification ID  :    " + rid;
        }
        else if (DropDownList1.SelectedItem.Text == "Nurse")
        {
            Label3.Text = "";
            Label1.Visible = true;
            Label2.Visible = true;
            TextBox2.Visible = true;
            TextBox3.Visible = true;
            rfdep.Visible = true;
            rfdesi.Visible = true;
            con.Open();
            d1 = DateTime.Now.ToString("dd-MM-yyyy");
            cmd = new SqlCommand("insert into reg values(" + rid + ",'" +
TextBox1.Text + "',null,'" + DropDownList1.SelectedItem.Text + "','" +
TextBox3.Text + "','" + TextBox2.Text + "','" + TextBox4.Text + "'," +
TextBox5.Text + ",'" + TextBox6.Text + "',0)", con);
            cmd.ExecuteNonQuery();
            con.Close();
            TextBox1.Text = "";
            TextBox2.Text = "";
            TextBox3.Text = "";
            TextBox4.Text = "";
            TextBox5.Text = "";
            TextBox6.Text = "";
            Label3.Text = "Succesfully Added Your Information.. Pls Note Your
Verification ID  :    " + rid;
        }

        else if (DropDownList1.SelectedItem.Text == "Parent")
        {
            Label3.Text = "";
            Label1.Visible = false;
            Label2.Visible = false;
            TextBox2.Visible = false;
```

```
            TextBox3.Visible = false;
            rfdep.Visible = false;
            rfdesi.Visible = false;
            con.Open();
            d1 = DateTime.Now.ToString("dd-MM-yyyy");
            cmd = new SqlCommand("insert into reg values(" + rid + ",'" +
TextBox1.Text + "',null,'" + DropDownList1.SelectedItem.Text + "',null,null,'" +
TextBox4.Text + "'," + TextBox5.Text + ",'" + TextBox6.Text + "',0)", con);
            cmd.ExecuteNonQuery();
            con.Close();
            TextBox1.Text = "";

            TextBox4.Text = "";
            TextBox5.Text = "";
            TextBox6.Text = "";
            Label3.Text = "Succesfully Added Your Information.. Pls Note Your
Verification ID  :    " + rid;
        }

        else if (DropDownList1.SelectedItem.Text == "Patient")
        {
            Label3.Text = "";
            Label1.Visible = false;
            Label2.Visible = false;
            TextBox2.Visible = false;
            TextBox3.Visible = false;
            rfdep.Visible = false;
            rfdesi.Visible = false;
            con.Open();
            d1 = DateTime.Now.ToString("dd-MM-yyyy");
            cmd = new SqlCommand("insert into reg values(" + rid + ",'" +
TextBox1.Text + "',null,'" + DropDownList1.SelectedItem.Text + "',null,null,'" +
TextBox4.Text + "'," + TextBox5.Text + ",'" + TextBox6.Text + "',0)", con);
            cmd.ExecuteNonQuery();
            con.Close();
            TextBox1.Text = "";

            TextBox4.Text = "";
            TextBox5.Text = "";
            TextBox6.Text = "";
            Label3.Text = "Succesfully Added Your Information.. Pls Verify after 2
hrs with your code  :    " + rid;
        }

    }
    protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
    {
        if (DropDownList1.SelectedItem.Text == "Doctor")
        {

            Label1.Visible = true;
            Label2.Visible = true;
            TextBox2.Visible = true;
            TextBox3.Visible = true;
            rfdep.Visible = true;
            rfdesi.Visible = true;
        }
        else if (DropDownList1.SelectedItem.Text == "Nurse")
```

```
        {
            Label1.Visible = true;
            Label2.Visible = true;
            TextBox2.Visible = true;
            TextBox3.Visible = true;
            rfdep.Visible = true;
            rfdesi.Visible = true;
        }
        else if (DropDownList1.SelectedItem.Text == "Parent")
        {
            Label1.Visible = false;
            Label2.Visible = false;
            TextBox2.Visible = false;
            TextBox3.Visible = false;
            rfdep.Visible = false;
            rfdesi.Visible = false;
        }
        else if (DropDownList1.SelectedItem.Text == "Patient")
        {
            Label1.Visible = false;
            Label2.Visible = false;
            TextBox2.Visible = false;
            TextBox3.Visible = false;
            rfdep.Visible = false;
            rfdesi.Visible = false;
        }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class HoLogin : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    SqlDataReader rdr;
    string strl;
    string utyp;
    protected void Page_Load(object sender, EventArgs e)
    {
        strl = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strl);
        TextBox2.Focus();
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
         string un = TextBox2.Text;
        string pw = TextBox3.Text;

         con.Open();
```

```
            cmd = new SqlCommand("SELECT * From Login where username='" + un + "'
and password='"+pw+"'", con);
            rdr = cmd.ExecuteReader();
            if(rdr.Read())
            {
                utyp = rdr["usertype"].ToString();
            }
            con.Close();
            Session["x"] = un;
            Session["utyp"] = utyp;
            if (utyp == "Doctor")
            {
                Response.Redirect("DoctorHome.aspx");

            }
            else if (utyp=="Nurse")
            {
                Response.Redirect("NurseHome.aspx");
            }
            else if (utyp == "Parent")
            {
                Response.Redirect("ParentsHome.aspx");
            }
            else if (utyp=="Patient")
            {
                Response.Redirect("PatientsHome.aspx");
            }
            else if (utyp=="Admin")
            {
                Response.Redirect("AdminHome.aspx");
            }
            else
            {
                Label3.Text="Invalid User Name or Password";
            }


    }

}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class HoNoti1 : System.Web.UI.Page
{
    SqlConnection con;
    SqlDataAdapter da;
    DataSet ds;
    String strcon;
    protected void Page_Load(object sender, EventArgs e)
    {
```

```
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);



        con.Open();
        da = new SqlDataAdapter("select * from notification where status=0", con);
        ds = new DataSet();
        da.Fill(ds, "notification");
        GridView1.DataSource = ds;
        GridView1.DataBind();
        con.Close();
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

public partial class HoNoti2 : System.Web.UI.Page
{
    SqlConnection con;
    SqlCommand comm;
    SqlDataReader reader;
    String strcon;
    int id;
    protected void Page_Load(object sender, EventArgs e)
    {
        id = Convert.ToInt32(Request.QueryString.ToString());
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);

        con.Open();
        if (!Page.IsPostBack)
        {


            comm = new SqlCommand("select * from notification where nid=" + id +
"", con);
            reader = comm.ExecuteReader();
            if (reader.Read() == true)
            {
                tn.Text = reader["ntype"].ToString();
                da.Text = reader["date"].ToString();
                no.Text = reader["notification"].ToString();




            }

        }
```

```
        con.Close();
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        Response.Redirect("HoNoti1.aspx");
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class ParComplaint : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader reader;
    int rid,id;
    string uname, utype;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);


        uname = Session["x"].ToString();
        //Label2.Text = uname.ToString();
        string utype = Session["utyp"].ToString();
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        d1 = DateTime.Now.ToString("dd-MM-yyyy");

        con.Open();
        cmd = new SqlCommand("select cid from complaint order by cid Desc", con);
        reader = cmd.ExecuteReader();
        if (reader.Read() == true)
        {
            id = Convert.ToInt32(reader["cid"]);
            id = id + 1;
        }
        else
        {
            id = 1;
        }
        cmd.Dispose();
        con.Close();


        con.Open();
        cmd = new SqlCommand("INSERT INTO complaint values(" + id + ",'" + uname +
"','" + ct.Text + "','" + co.Text + "','" + d1 + "',null,0   )", con);
```

```
        cmd.ExecuteNonQuery();
        if (cmd != null)
        {
            ct.Text = "";
            co.Text = "";
       }

        con.Close();
        Label3.Text = "Complaint updated :   Your Complaint ID is   :  "+ id;



    }
}
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;

public partial class ParCommSend : System.Web.UI.Page
{
    String str, r, un, y, utyp, d1;
    SqlCommand cmd;
    SqlConnection con;
    int comid;
    String d;
    SqlDataReader rdr;
    ArrayList vg = new ArrayList();
    protected void Page_Load(object sender, EventArgs e)
    {
        y = Session["x"].ToString();
        // Label2.Text = y.ToString();
        utyp = Session["utyp"].ToString();
        // y = "procoo";
        str = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(str);
        //d = Convert.ToString(DateTime.Now.ToString("dd/MM/yyyy"));
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        con.Open();
        cmd = new SqlCommand("select cid from communication order by cid DESC",
con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            comid = Convert.ToInt32(rdr["cid"]);
            comid = comid + 1;
        }
        else
        {
```

```
            comid = 1;
        }
        cmd.Dispose();
        con.Close();

        con.Open();
        d1 = DateTime.Now.ToString("dd/MM/yyyy");
        cmd = new SqlCommand("insert into communication values(" + comid + ",'" +
utyp + "','" + y + "','" + DropDownList1.SelectedItem.Text + "','" +
DropDownList2.SelectedItem.Text + "','" + d1 + "','" + TextBox1.Text + "','" +
TextBox2.Text + "',null,null,0)", con);
        // cmd.ExecuteNonQuery();
        int i = cmd.ExecuteNonQuery();
        if (i > 0)
        {
            Page.RegisterStartupScript("UserMsg", "<script>alert('Message Send
SuccessFully !');</script>");
        }
        // Label1.Text = " Message Sent  ";
        TextBox1.Text = "";
        TextBox2.Text = "";
        cmd.Dispose();
        con.Close();
    }
    protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
    {
        con.Open();
        cmd = new SqlCommand("select username from login where usertype='" +
DropDownList1.SelectedItem.Text + "'", con);
        rdr = cmd.ExecuteReader();
        while (rdr.Read() == true)
        {
            un = rdr["username"].ToString();
            vg.Add(un);
            DropDownList2.DataSource = vg;
            DropDownList2.DataBind();
        }
        cmd.Dispose();
        rdr.Dispose();
        con.Close();
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class ParCommR1 : System.Web.UI.Page
{
    SqlConnection con;
    SqlDataAdapter da;
    DataSet ds;
    String strcon;
```

```
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        string uname = Session["x"].ToString();
        string s1 = "Parent";
        // Label1.Text = uname.ToString();
        string utyp = Session["utyp"].ToString();
        con.Open();
        da = new SqlDataAdapter("select * from communication where sendertype<>'"
+ s1 + "' and receivername='" + uname + "' and status=0", con);
        ds = new DataSet();
        da.Fill(ds, "communication");
        GridView1.DataSource = ds;
        GridView1.DataBind();
        con.Close();
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

public partial class ParCommR2 : System.Web.UI.Page
{
    SqlConnection con;
    SqlCommand comm;
    SqlDataReader reader;
    String strcon;
    int id;
    protected void Page_Load(object sender, EventArgs e)
    {
        id = Convert.ToInt32(Request.QueryString.ToString());
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        string uname = Session["x"].ToString();
        //Label2.Text = uname.ToString();
        string utyp = Session["utyp"].ToString();
        con.Open();
        if (!Page.IsPostBack)
        {


            comm = new SqlCommand("select * from communication where cid=" + id +
"", con);
            reader = comm.ExecuteReader();
            if (reader.Read() == true)
            {
                st.Text = reader["sendertype"].ToString();
                sn.Text = reader["sendername"].ToString();
                s.Text = reader["subject"].ToString();
                msg.Text = reader["message"].ToString();
                sdate.Text = reader["sdate"].ToString();
```

```
                reply.Text = reader["reply"].ToString();


            }

        }

        con.Close();
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Label7.Visible = true;
        TextBox1.Visible = true;
        Button2.Visible = true;
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        string d1 = DateTime.Now.ToString("dd/MM/yyyy");

        con.Open();
        comm = new SqlCommand("update communication set status=1,reply='" +
TextBox1.Text + "',rdate='" + d1 + "'  where cid=" + id + "", con);
        comm.ExecuteNonQuery();
        con.Close();
        Response.Redirect("ParCommR1.aspx");
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class PaComplaintUpdatesV : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader reader;
    int rid, id;
    string uname, utype;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);


        uname = Session["x"].ToString();
        //Label2.Text = uname.ToString();
        string utype = Session["utyp"].ToString();
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        con.Open();
```

```
        cmd = new SqlCommand("select * from complaint where status=1 and
cid='"+un.Text+"'", con);
        reader = cmd.ExecuteReader();
        if (reader.Read() == true)
        {
            ct.Text = Convert.ToString(reader["ctype"]);
            ds.Text = Convert.ToString(reader["date"]);
            co.Text = Convert.ToString(reader["complaint"]);
            cr.Text = Convert.ToString(reader["updation"]);

        }
        else
        {
            Label1.Text = "Invalid Number!! Pls Try Again!!";
        }

        con.Close();

    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class ParAppointmentFixing : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    SqlDataAdapter da;
    DataSet ds;
    int rid, id;
    string ut1, ut2, ti,st, sn;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        ut1 = "Patient";
        ut2 = "Doctor";
        sn = Session["x"].ToString();
        st = Session["utyp"].ToString();

        if (!Page.IsPostBack)
        {
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut1 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            pid.DataSource = ds;
            pid.DataTextField = "name";
```

```
            pid.DataValueField = "uid";
            pid.DataBind();
            con.Close();

            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut2 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            did.DataSource = ds;
            did.DataTextField = "name";
            did.DataValueField = "uid";
            did.DataBind();
            con.Close();
        }
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        d1 = DateTime.Now.ToString("dd-MM-yyyy");


        con.Open();
        cmd = new SqlCommand("select apid from appointment order by apid Desc",
con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            id = Convert.ToInt32(rdr["apid"]);
            id = id + 1;
        }
        else
        {
            id = 1;
        }
        cmd.Dispose();
        con.Close();

        con.Open();
        d1 = DateTime.Now.ToString("dd-MM-yyyy");
        cmd = new SqlCommand("insert into appointment values(" + id + ", '" + st +
"','" + sn + "','" + pid.SelectedValue + "','" + pid.SelectedItem.Text + "','" +
did.SelectedValue + "','" + did.SelectedItem.Text + "','" + de.Text + "','" +
doa.Text + "','" + toa.Text + "','" + re.Text + "',null,null,null,0)", con);
        cmd.ExecuteNonQuery();
        con.Close();
        re.Text = "";
        de.Text = "";
        doa.Text = "";
        toa.Text = "";


        Label3.Text = "Appointment Request Send Successfully!!!  Your Tracking ID
is    :  " + id;

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
```

```
        re.Text = "";
        de.Text = "";
        doa.Text = "";
        toa.Text = "";
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class PatViewAppointment : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    SqlDataAdapter da;
    DataSet ds;
    int rid, id;
    string ut1, ut2, ti, st, sn;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        ut1 = "Patient";
        ut2 = "Doctor";
        sn = Session["x"].ToString();
        st = Session["utyp"].ToString();
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        con.Open();
        cmd = new SqlCommand("select * from appointment where apid=" + ap.Text + "
and sendername='" + sn + "'", con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            con.Close();
            Label3.Visible = false;
            GridView1.Visible = true;
            con.Open();


            da = new SqlDataAdapter("select sendername, doctorid, allowdate,
allowtime, department, statusupdate from appointment where apid=" + ap.Text + "
and sendername='" + sn + "'", con);
            ds = new DataSet();
            da.Fill(ds, "appointment");
            GridView1.DataSource = ds;
            GridView1.DataBind();
            con.Close();
```

```
        }
        else
        {
            Label3.Visible = true;
            GridView1.Visible = false;
            Label3.Text = "Invalid Appointmnent Number! Pls Try Again!!";
        }

    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;

public partial class ParAppointmentFixing : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    SqlDataAdapter da;
    DataSet ds;
    int rid, id;
    string ut1, ut2, ti,st, sn;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        ut1 = "Patient";
        ut2 = "Doctor";
        sn = Session["x"].ToString();
        st = Session["utyp"].ToString();

        if (!Page.IsPostBack)
        {
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut1 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            pid.DataSource = ds;
            pid.DataTextField = "name";
            pid.DataValueField = "uid";
            pid.DataBind();
            con.Close();

            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut2 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            did.DataSource = ds;
```

```
            did.DataTextField = "name";
            did.DataValueField = "uid";
            did.DataBind();
            con.Close();
        }
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        d1 = DateTime.Now.ToString("dd-MM-yyyy");


        con.Open();
        cmd = new SqlCommand("select apid from appointment order by apid Desc",
con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            id = Convert.ToInt32(rdr["apid"]);
            id = id + 1;
        }
        else
        {
            id = 1;
        }
        cmd.Dispose();
        con.Close();

        con.Open();
        d1 = DateTime.Now.ToString("dd-MM-yyyy");
        cmd = new SqlCommand("insert into appointment values(" + id + ", '" + st +
"','" + sn + "','" + pid.SelectedValue + "','" + pid.SelectedItem.Text + "','" +
did.SelectedValue + "','" + did.SelectedItem.Text + "','" + de.Text + "','" +
doa.Text + "','" + toa.Text + "','" + re.Text + "',null,null,null,0)", con);
        cmd.ExecuteNonQuery();
        con.Close();
        re.Text = "";
        de.Text = "";
        doa.Text = "";
        toa.Text = "";



        Label3.Text = "Appointment Request Send Successfully!!!  Your Tracking ID
is    :  " + id;

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        re.Text = "";
        de.Text = "";
        doa.Text = "";
        toa.Text = "";
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;


public partial class NuEmergencyAlert : System.Web.UI.Page
{
    SqlCommand cmd;
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    SqlDataAdapter da;
    DataSet ds;
    int rid, id;
    string ut1, ut2, ti, st, sn;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        ut1 = "Patient";
        ut2 = "Doctor";
        sn = Session["x"].ToString();
        st = Session["utyp"].ToString();

        if (!Page.IsPostBack)
        {
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut1 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            pid.DataSource = ds;
            pid.DataTextField = "name";
            pid.DataValueField = "uid";
            pid.DataBind();
            con.Close();
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut2 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            did.DataSource = ds;
            did.DataTextField = "name";
            did.DataValueField = "uid";
            did.DataBind();
            con.Close();
        }
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        d1 = DateTime.Now.ToString("dd-MM-yyyy");


        con.Open();
        cmd = new SqlCommand("select eid from emergency order by eid Desc", con);
        rdr = cmd.ExecuteReader();
```

```
        if (rdr.Read() == true)
        {
            id = Convert.ToInt32(rdr["eid"]);
            id = id + 1;
        }
        else
        {
            id = 1;
        }
        cmd.Dispose();
        con.Close();

        con.Open();
        d1 = DateTime.Now.ToString("dd-MM-yyyy");
        cmd = new SqlCommand("insert into emergency values(" + id + ",'" +
pid.SelectedValue + "','" + pid.SelectedItem.Text + "','" + pch.Text + "','" +
did.SelectedItem.Text + "','" + sn + "','" + description.Text + "','" +
remark.Text + "','" + time1.Text + "','" + date1.Text + "','" + etd.Text + "',0)",
con);
        cmd.ExecuteNonQuery();
        con.Close();


        description.Text = "";
        remark.Text = "";
        time1.Text = "";
        date1.Text = "";
        etd.Text = "";
        pch.Text = "";



        Label3.Text = "Updated Emergency Alert !!";
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        description.Text = "";
        remark.Text = "";
        time1.Text = "";
        date1.Text = "";
        etd.Text = "";
        pch.Text = "";
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;


public partial class NuCurrentUpdates_Remark : System.Web.UI.Page
{
    SqlCommand cmd;
```

```
    SqlConnection con;
    string strcon, d1;
    SqlDataReader rdr;
    SqlDataAdapter da;
    DataSet ds;
    int rid, id;
    string ut1, ut2, ti, st, sn;
    protected void Page_Load(object sender, EventArgs e)
    {
        strcon = ConfigurationManager.ConnectionStrings["w"].ConnectionString;
        con = new SqlConnection(strcon);
        ut1 = "Patient";
        ut2 = "Doctor";
        sn = Session["x"].ToString();
        st = Session["utyp"].ToString();

        if (!Page.IsPostBack)
        {
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut1 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            pid.DataSource = ds;
            pid.DataTextField = "name";
            pid.DataValueField = "uid";
            pid.DataBind();
            con.Close();
            con.Open();
            da = new SqlDataAdapter("select * from reg where usertype='" + ut2 +
"' and status=1", con);
            ds = new DataSet();
            da.Fill(ds, "reg");
            did.DataSource = ds;
            did.DataTextField = "name";
            did.DataValueField = "uid";
            did.DataBind();
            con.Close();
        }
    }
    protected void Button2_Click1(object sender, EventArgs e)
    {
        d1 = DateTime.Now.ToString("dd-MM-yyyy");


        con.Open();
        cmd = new SqlCommand("select uid from remark_currentupdates order by uid
Desc", con);
        rdr = cmd.ExecuteReader();
        if (rdr.Read() == true)
        {
            id = Convert.ToInt32(rdr["uid"]);
            id = id + 1;
        }
        else
        {
            id = 1;
        }
```

```
        cmd.Dispose();
        con.Close();

        con.Open();
        d1 = DateTime.Now.ToString("dd-MM-yyyy");
        cmd = new SqlCommand("insert into remark_currentupdates values(" + id +
","'"+sn+"','" + pid.SelectedValue + "','" + pid.SelectedItem.Text + "','" +
did.SelectedValue + "','" + did.SelectedItem.Text + "','" + re.Text + "','" +
rem.Text + "',0)", con);
        cmd.ExecuteNonQuery();
        con.Close();


        re.Text = "";
        rem.Text = "";



        Label3.Text = "Updated!!";
    }
protected void  Button1_Click(object sender, EventArgs e)
{
     re.Text = "";
        rem.Text = "";
}
}
```

CHAPTER 6

TESTING

Software testing is the process used to help identify the correctness, completeness, security and quality of developed computer software. Testing is vital to the success of the system. System Testing makes logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. System Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The user tests the developed system and changes are made according to their needs. The testing phase involves the testing of developed system using various kinds of data. There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following rote procedure. One definition of testing is "the process of questioning a product in order to evaluate it", where the "questions" are things the tester tries to do with the product, and the product answers with its behaviour in reaction to the probing of the tester. The quality of the application can, and normally does, vary widely from system to system but some of the common quality attributes include reliability, stability, portability, maintainability and usability

6.1 LEVELS OF TESTING

6.1.1 Unit testing

Unit testing focuses verification effort on the smallest unit of software design that is the module. The Unit Testing is always White Box oriented and the step can be conducted in parallel from modules. Boundary conditions are tested to ensure that the module operates properly. In white box testing the test developer has access to the source code and can write code that links into the libraries, which are linked into the target software. This is typical of unit tests, which only test parts of a software system. They ensure that components used

in the construction are functional and robust to some degree. Unit testing is an equivalent to the coding step. After the source code level code has been developed, reviewed and verified for correct syntax, unit test case design begins since a module is not a stand-alone program. In most applications a driver program is nothing more than a main program that accepts test case data. Passes such data to the module to be tested, and prints the relevant result.

6.1.2 Integration testing

Integration testing (sometimes called Integration and testing and abbreviated I&T) is the phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing .Integration testing is a systematic technique for constructing, while at the same time conducting test to uncover errors associated with interfacing. The purpose of Integration testing is to verify functional, performance and reliability requirements placed on major design items. All test cases are constructed to test that all components within assemblages interact correctly the objective is to take unit tested modules and built a program structure that has been dictated by design. The steps involved in this testing include:

1. The main control module is used as a test driver and status are substituted for all modules directly subordinate to the main control module.
2. Depending on the integration approach selected that is depth or breadth first, subordinate stubs are replaced one at time eight action modules.
3. Test is conducted as each module is integrated.
4. On the completion of each are of tests, another stub is replaced with the real module.
5. The low-level modules are combined into clusters hat performed a specific software sub function.
6. A driver that is the control program for testing is written to coordinate test case input and output.
7. The cluster is tested.

6.1.3 Validation testing

Validation Testing, carried out by QA professionals, is to determine if the system complies with the requirements and performs functions for which it is intended and meets the organization's goals and user needs. This kind of testing is very important, as well as verification testing. Validation is done at the end of the development process and takes place after verification is completed. Thus, to ensure customer satisfaction, developers apply validation testing. Its goal is to validate and be confident about the product or system and that it fulfills the requirements given by the customer. The acceptance of the software from the end customer is also its part. When software is tested, the motive is to check the quality regarding the found defects and bugs. When defects and bugs are detected, developers fix them. After that, the software is checked again to make sure no bugs are left. In that way, the software product's quality scales up. The aim of software testing is to measure the quality of software in terms of a number of defects found in it, the number of tests run and the system covered by the tests. When bugs or defects are found with the help of testing, the bugs are logged and the development team fixes them. Once the bugs are fixed, testing is carried out again to ensure that they are indeed fixed and no new defects have been introduced in the software.

6.1.4 Black box testing

Black Box Testing is also known as functional testing. A software testing technique is internal workings of the item being tested are not known by the tester. In a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. Black box testing is testing that occurs from the viewpoint of an end user. Black box tests find bugs such as incorrect functions, interface problems, and database errors. Black box techniques focus on information domain of the software deriving test cases by partitioning input and output in a manner that provides through test coverage. The test cases are developed for each condition or combination of conditions and submitted for processing. By examining the results, the performance of the program according to its specified requirements can be determined.

6.1.5 White box testing

White box testing strategy deals with the internal logic and structure of the code. White box testing is also called as glass, structural, open box or clears box testing. The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc. The system has been tested by providing variety of inputs to ensure that all the statements are executed at least once and that too in the expected manner. All topic and transaction path from origin to destination was tested to identify and correct the possible error.

6.1.6 Output Testing

No system could be useful if it doesn't produce the required output in the specific format. The output interfaces was tested to ensure if the system provides correct, accurate output in the specified format. The output of online medicine donation system is tested by combining the entire system and it produces the desired result.

6.1.7 User Acceptance Testing

The online medicine donation system is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing. The testing of the software began along with the coding. The unit testing was done for each module in the software. For various inputs such that each line of code is executed at least once. After all modules were coded, integration testing was carried out. Some minor errors were found in the earlier stage an each of them was corrected. In the implementation of the user interface part no major errors were noted. After the software was completely developed the testing was done. The output was correct and accurate during the time of documentation. After that no errors were reported.

CHAPTER 7
IMPLIMENTATION

7.1 IMPLEMENTATION OF THE PROPOSED SYSTEM

Implementation is the process of converting a new or revised system design into operation. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system. It must therefore be carefully planned and controlled. Apart from planning the two major tasks of preparing for implementation are education and training of users and testing of the system. Education of users should really take place much earlier in the project .i.e. when they are involved in the investigation and design work. Training has been given to the staff regarding the new system. Once staff has been trained, the system can be tested. Implementation is the stage of project where the theoretical design is turned into working system.

Implementation is the final and important phase. It is the phase where theoretical design is turned into working system, which works for user in the most effective manner. It involves careful planning investigation of the present system and the constrained involve, user training, system testing and successful running of developed proposed system. The Implementation process begins with preparing a plan for the implementation of the system.

7.2 INSTALLATION PROCEDURE

Implementation of software refers to the final installation of the package in its real implementation of software refers to the final installation of the packages in its real environment, to the satisfaction of the internal users and the operation of the system. In many organization someone who will not be operating it, will commission the software development project .the people who are not sure that the software is meant to make their job easier. In the initial stage they doubt software but we have to ensure that the resistance does not built up as one has to make sure that the active user must be aware of the benefits

of using the system proper guidance be imparted to the user so that he is comfortable in using the application. Their confidence in the software is built up

Before going ahead and viewing the system, the must know that for viewing the result, the server program should be running in the server, the actual processes will not take place.

The task of installing a new system has degree of difficulties. Since the hard ware is already running, only new software system has to be installed, in this case there arise some issues. They are:

1. Is there are a big enough?
2. Is the layout appropriate?
3. Is the environment appropriate?
4. Does it require an additional hardware?
5. Is this site secure?

After solving these issues the ONLINE MEDICINE DONATION SYSYTEM is installed. The interface between all the sub systems is checked. All errors are corrected before acceptance test. After this entire task is performed the ONLINE MEDICINE DONATION SYSTEM is provided to work. To install the system, the primary need is web based environments without which the system will not have a proper utilization. The following components components are needed at the server end,

1. ASP.NET
2. MySQL
3. The following components are needed at the client end,
4. Web browser
5. Internet connection

CHAPTER 8

SECURITY BACKUP AND RECOVERY MECHANISAM

Security is an important consideration in this system. The first step in securing this system is deciding where you need security and what it needs to protect.

Security Concepts:

1. Authentication
2. Authorization

Authentication is a process in which the credentials provided are compared to those on file in a database of authorized users' information on a local operating system or within an authentication server. If the credentials match, the process is completed and the user is granted authorization for access.

The process of an administrator granting rights and the process of checking user account permissions for access to resources are both referred to as authorization. The privileges and preferences granted for the authorized account depend on the user's permissions, which are either stored locally or on the authentication server.

8.1 ONLINE HELP

A link will be provided to contact the developer by the users if any inconvenience faced during the usage of this system. The developer then will respond to the queries by the user immediately.

8.2 USER MANUALS

User Manual is a technical communication document intended to give assistance to people using a particular system. It is usually written by a technical writer, although user guides are written by programmers, product or project managers, or other technical staff, particularly in smaller companies.

CHAPTER 9

CONCLUSION

The Adulteration Detecting System eliminates most of the drawbacks of existing system. The system is more flexible and can be modified easily whenever needed. It is very much easy to understand. The project Adulteration Detecting System after being tested and found to be achieving what is meant for. The system is found to be 100% error free and ready for implementation. The system provides high security. The maintenance of the system is less when compared with the existing system.

CHAPTER 10

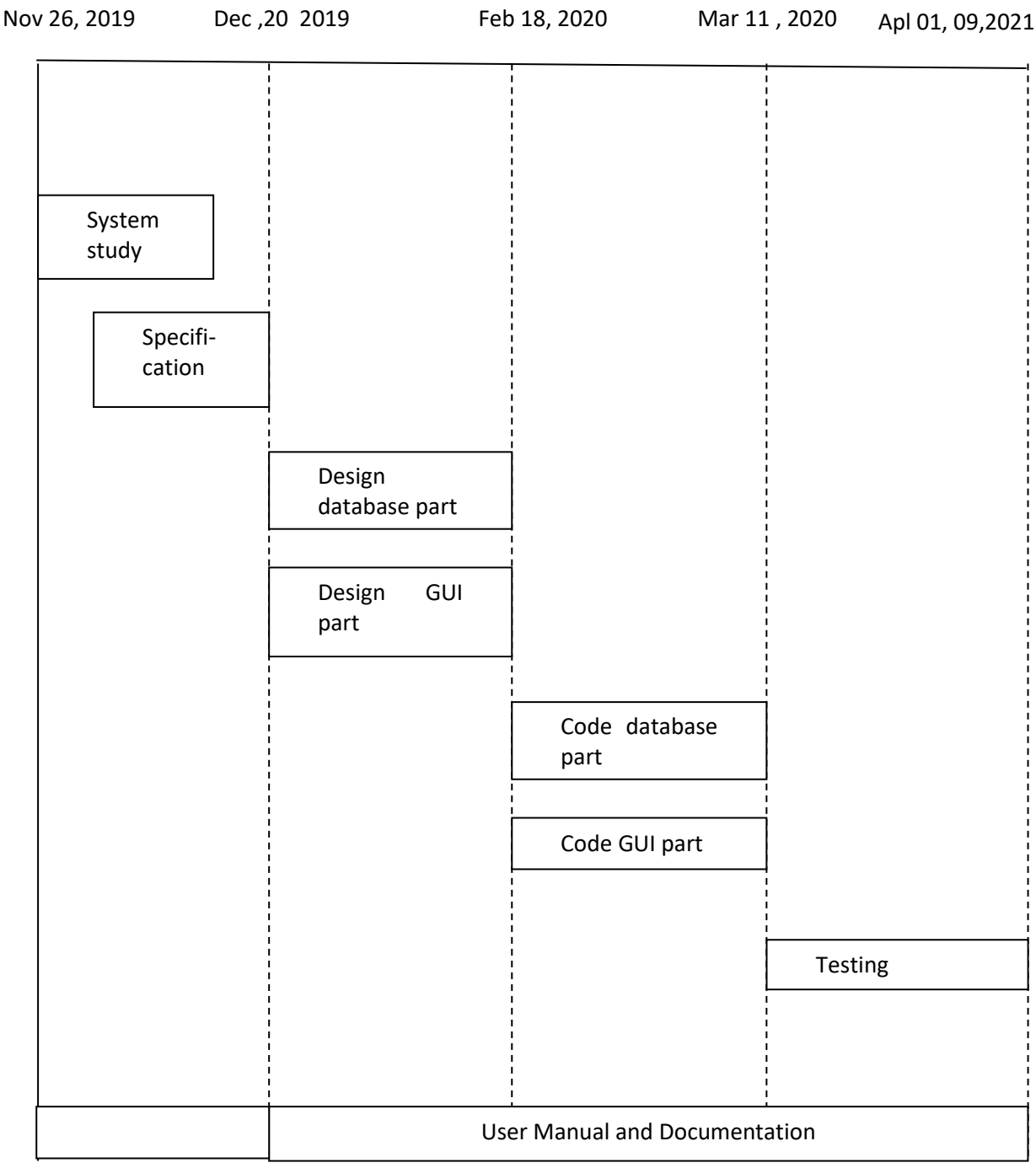
FUTURE ENHANCEMENT

1. Access to the system and database as per user identification. Thus maximum security ensured.
2. Increased reliability and integrity of data.
3. User-friendly and flexible in all aspects.
4. Data entry and updating is quite easy.
5. Effective table manipulation as facilitated by the rich MySQL.
6. Increased customer satisfaction.
7. Good validation checking.
8. Facilitates registration cancelling.
9. Reduce complexity in data entry.
10. Easy maintenance is providing.

CHAPTER 11

APPENDIX

11.1 GANTT CHART



11.2 SCREEN SHORTS

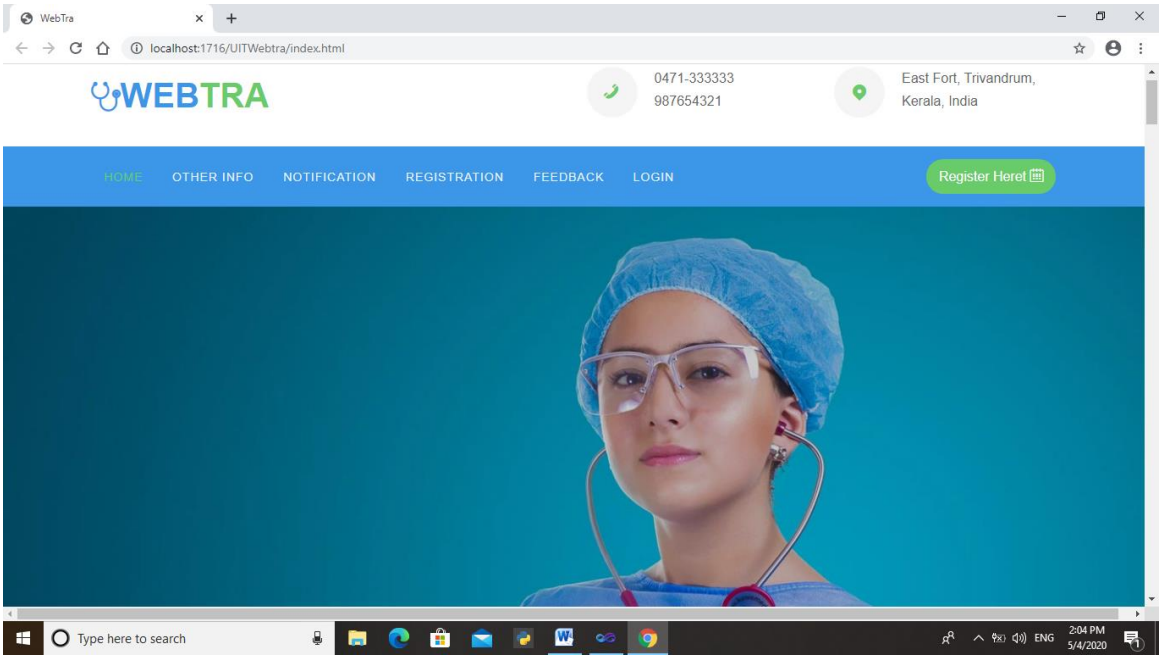


Figure A:1 Home page

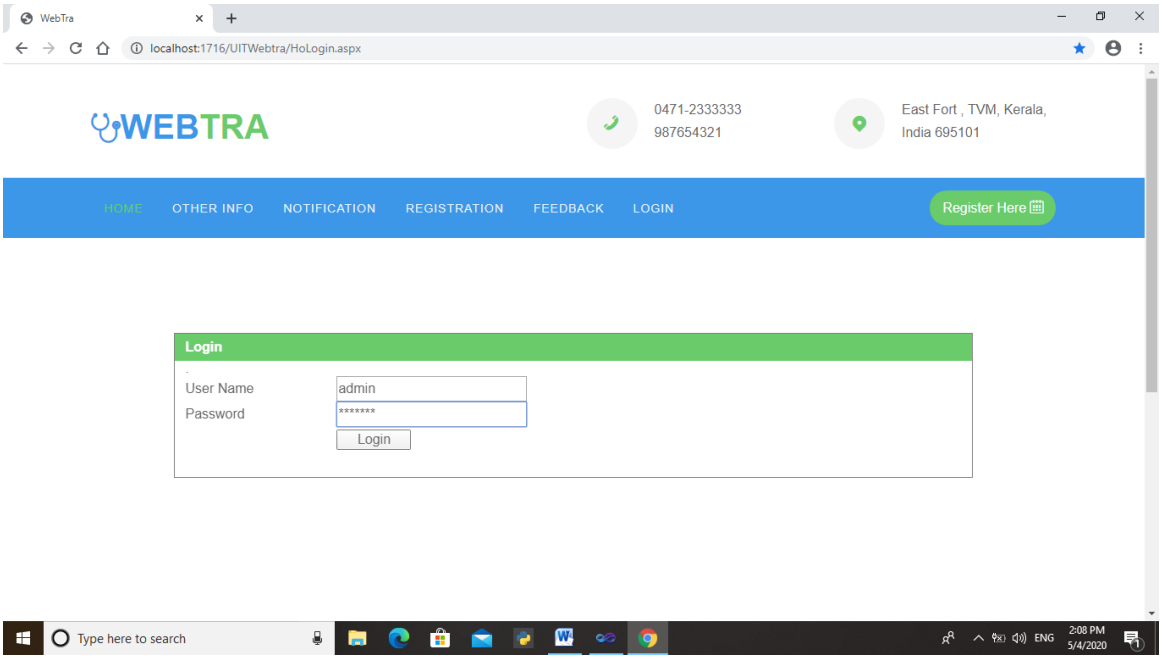


Figure A: 2 login page

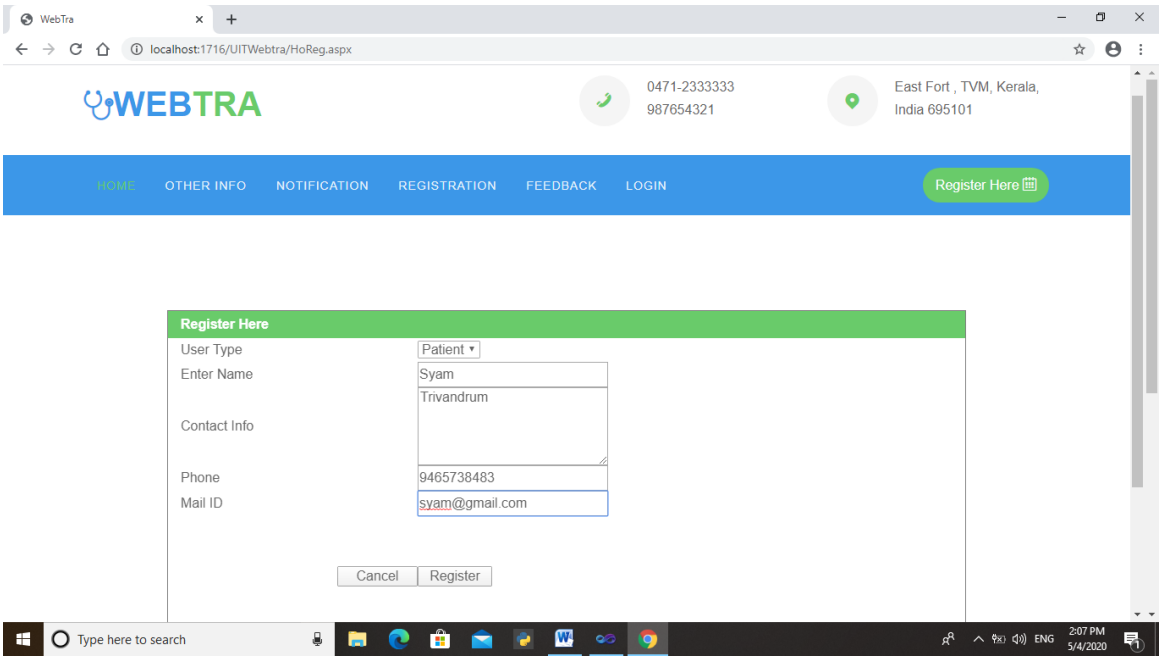


Figure A: 3 Registration page

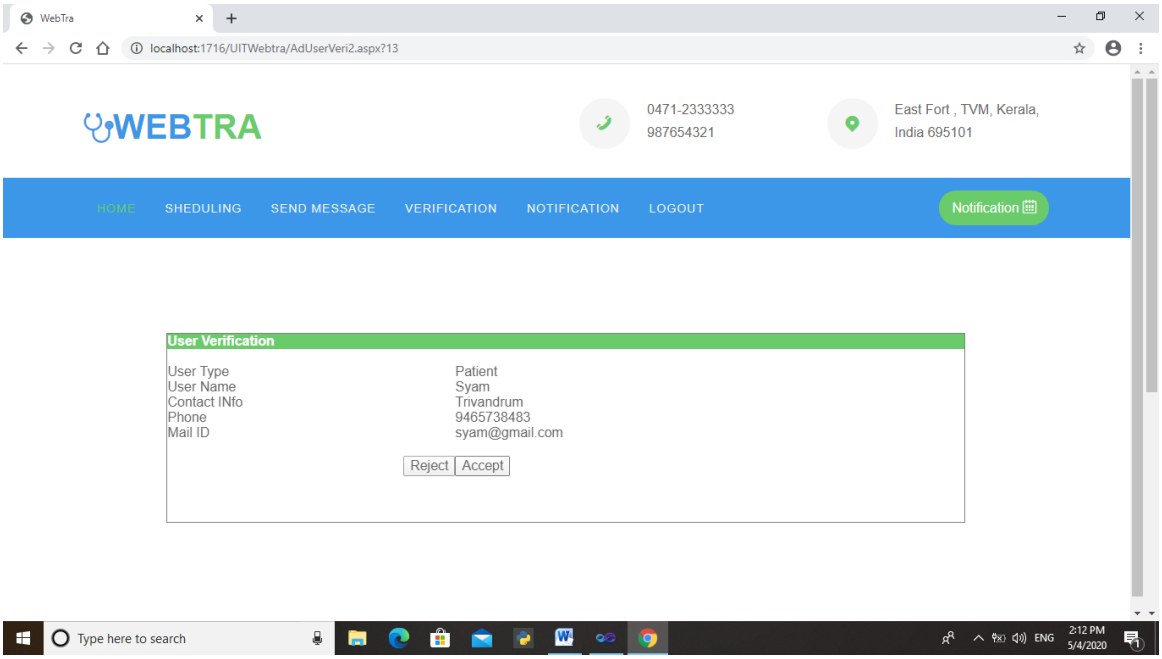


Figure A: 4 Verification page

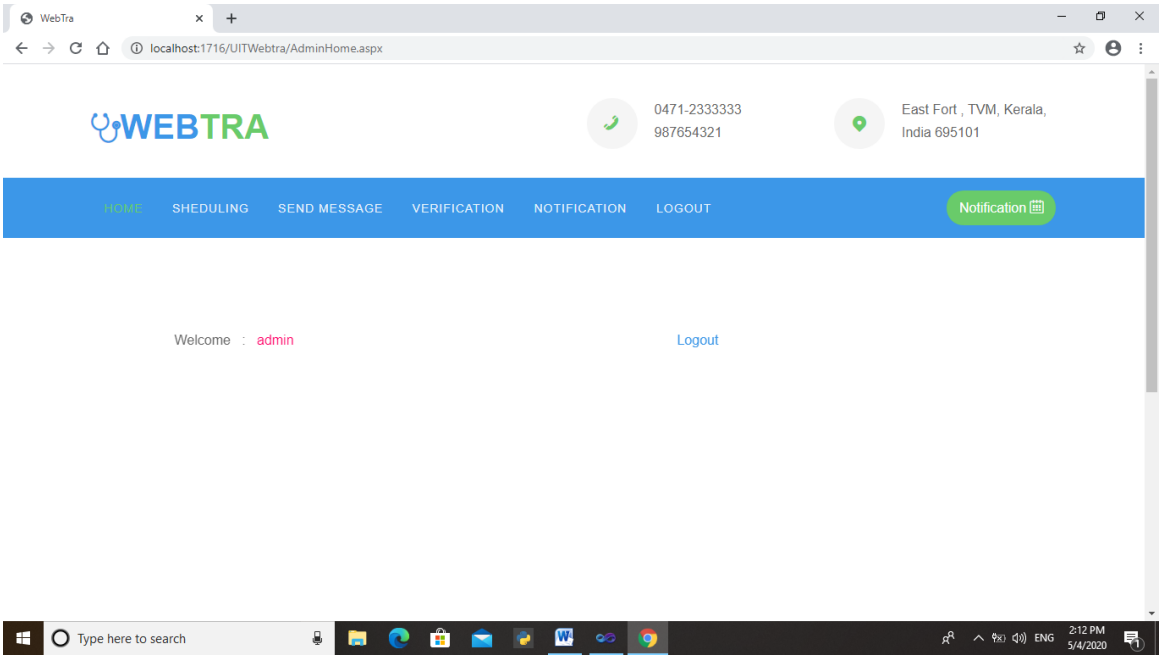


Figure A: 5 Admin page

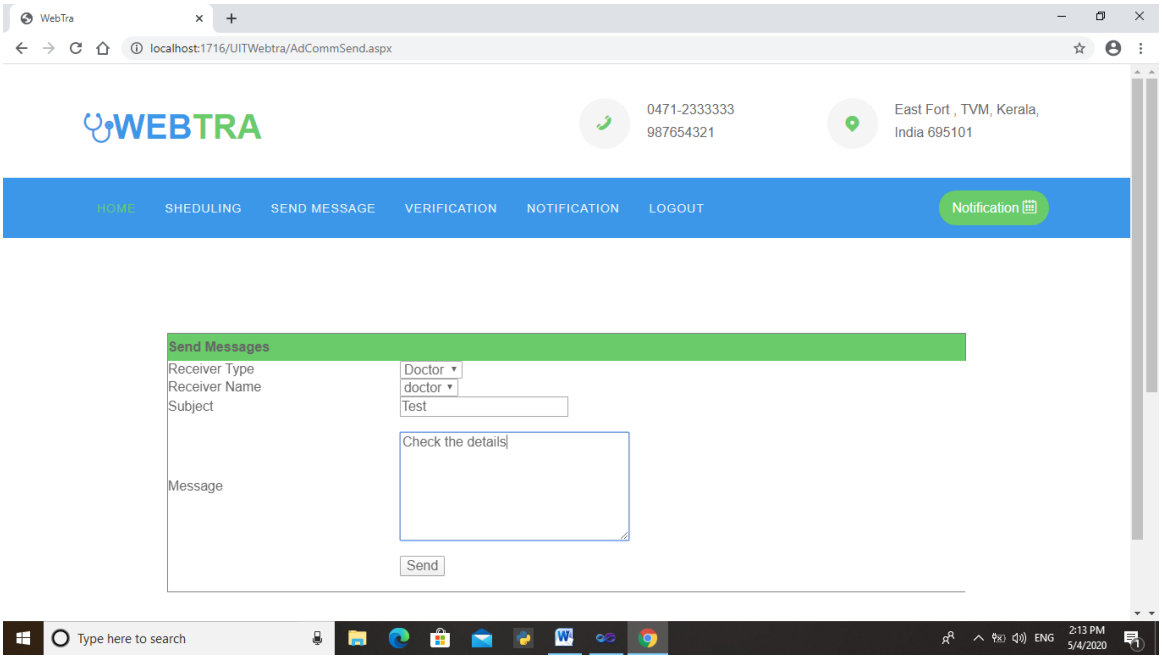


Figure A: 6 Send Message page

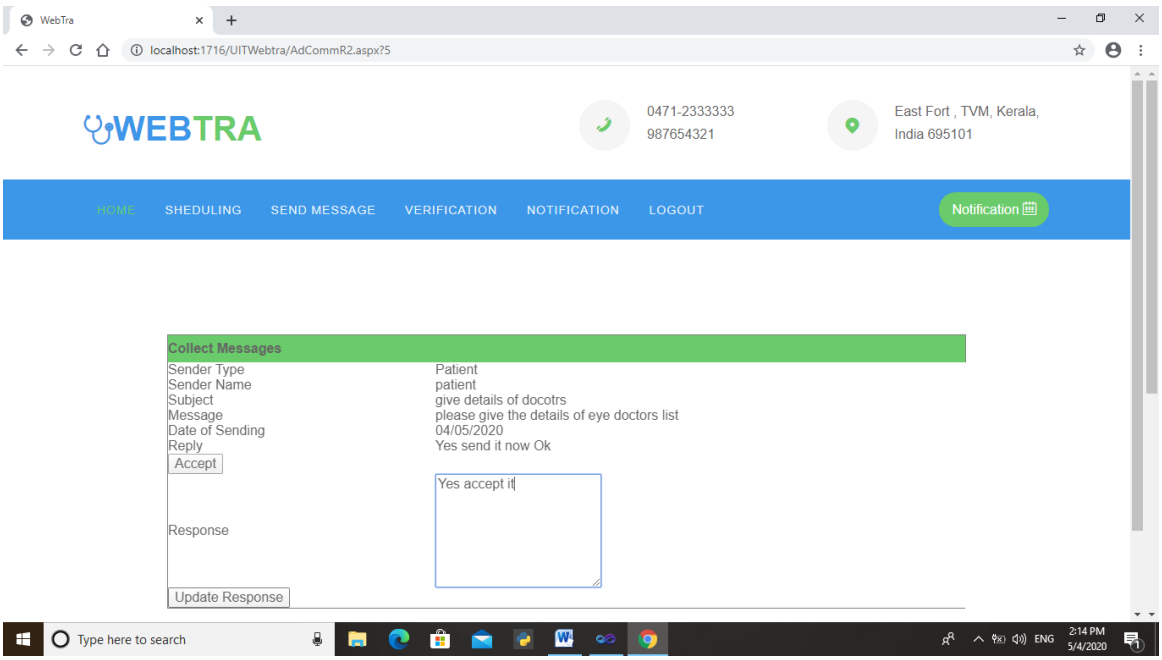


Figure A: 8 COMMENTS

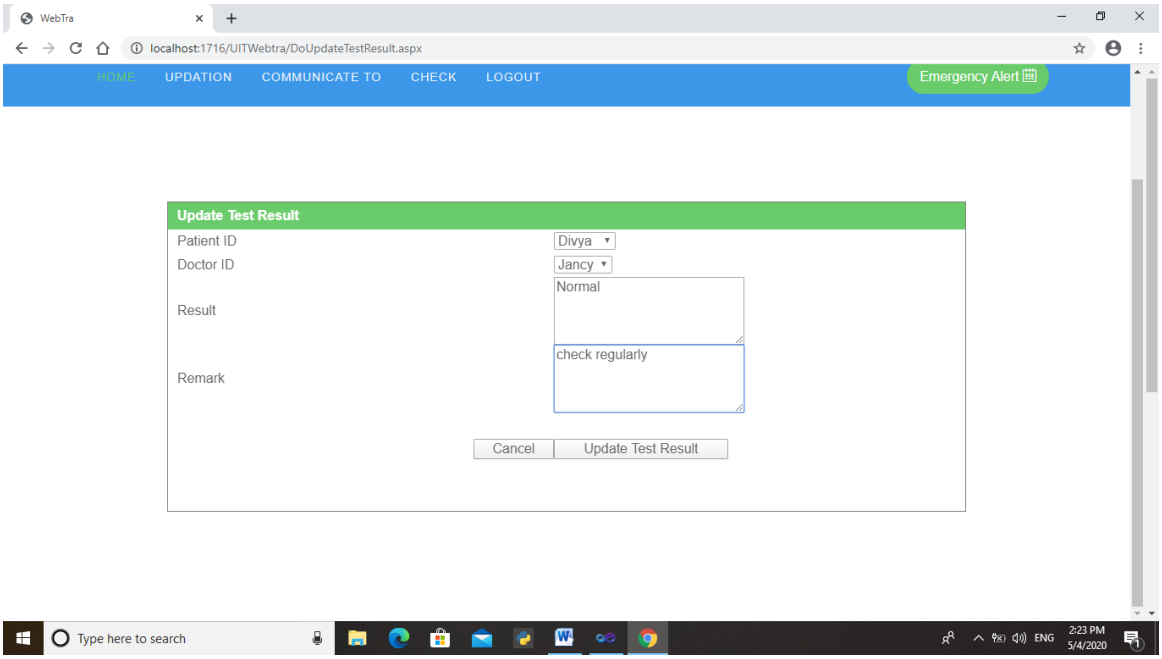


Figure A: 9 UPDATE TEST RESULT pages

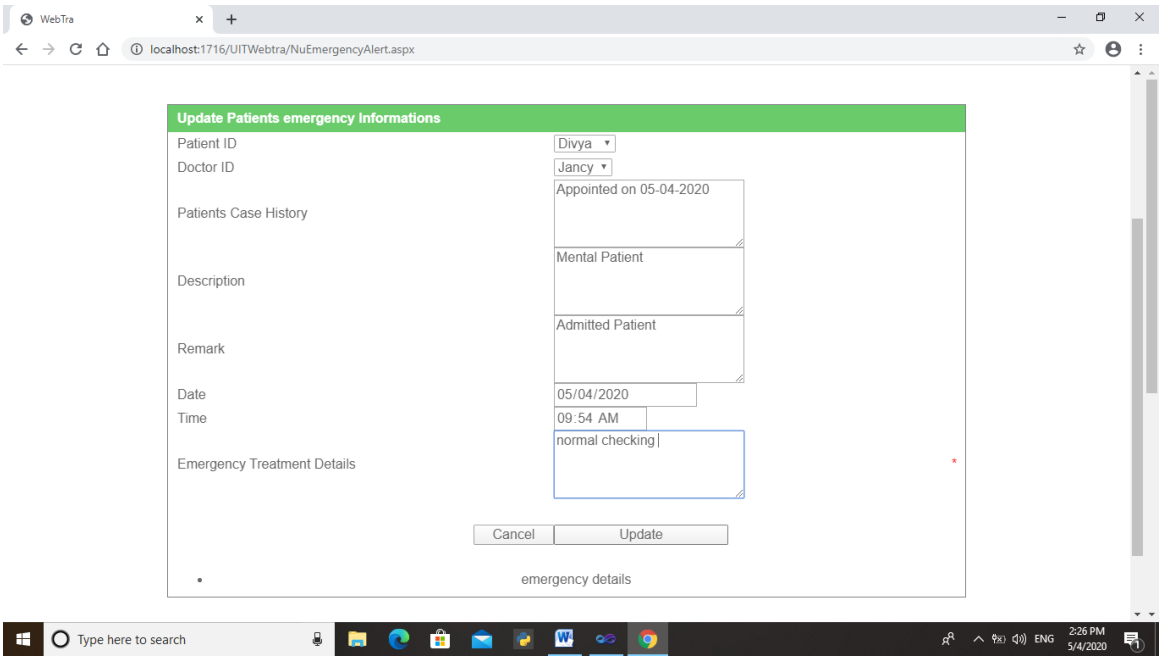


Figure A: 10 UPDATE PATIENTS EMERGENCY INFORMATIO

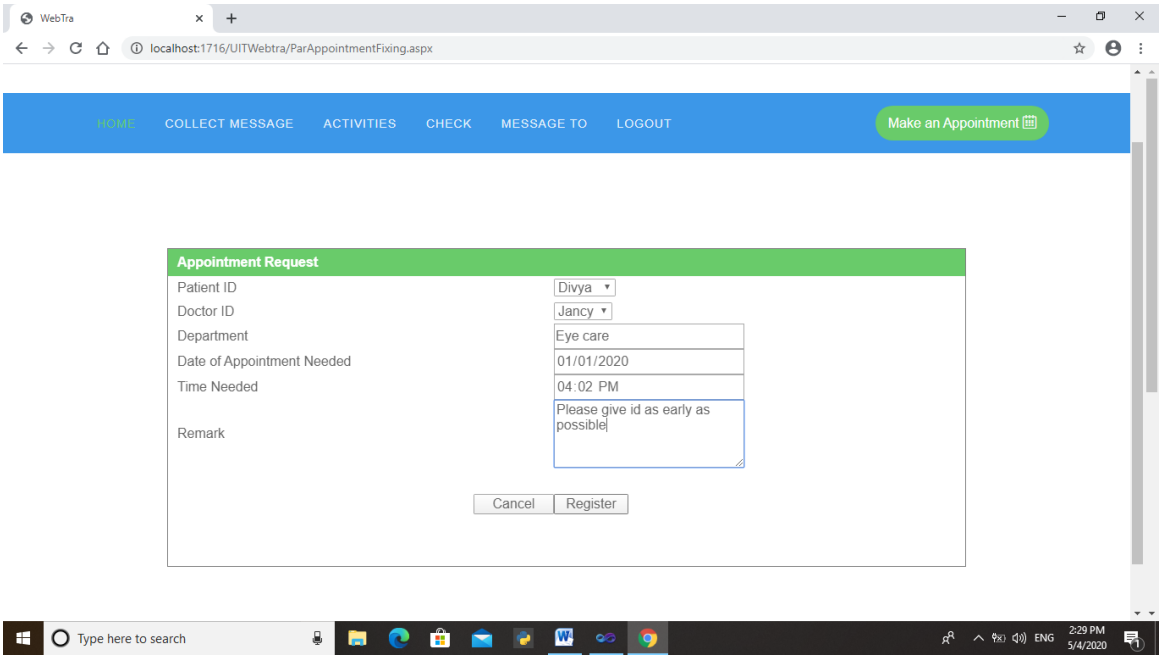


Figure A: 11 APPONITMENT REQUEST

CHAPTER 12

BIBILIOGRAPHY

1. Fundamentals of Software Engineering, Rajib Mall, third edition, Eastern Economy Edition (EEE)

2. Roger S Pressman, Software Engineering, A practitioner approach, Tara Mchraw Hill

3. .NET Book Zero by Charles Petzold

4. System Analysis and Design-Elias M.Award

5. Fundamentals of Database by Elmasi and Nacathe

6. Structured Analysis and System Specification, DeMarco, Yourdon press, Newyork, 1978.

List of web addresses visited

1. www.wikipedia.com
2. www.codeproject.com