"DEVELOPING AN EFFICIENT ONLINE QUERY PROCESSING SYSTEM"

Submitted in partial fulfillment for the Degree of B. Tech

in

Computer Science Engineering

under the Supervision of

Dr. PRADEEP KUMAR GUPTA

By

MANSI SAHI(091225)

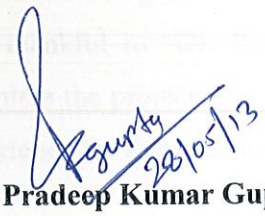
JIVESH SHARMA(091227)

Session 2012-2013

Jaypee University of Information Technology
Waknaghat, Solan – 173234, Himachal Pradesh

CERTIFICATE

This is to certify that the work entitled "DEVELOPING AN EFFICIENT QUERY PROCESSING SYSTEM" is the original work being submitted by Mansi Sahi (091225), Jivesh Sharma(091227)" in partial fulfillment for the Degree of B.Tech from Jaypee university of information and technology, Waknaghat. To the best of my knowledge and belief, these candidates have fulfilled the requirements of the rules and regulations relating to the degree in B.Tech of Jaypee University of information and technology, Waknaghat. The matter embodied in this project has not been submitted for the award of any degree of any university



28/05/13

(Dr. Pradeep Kumar Gupta)

Supervisor

ACKNOWLEDGEMENT

The satisfaction that accomplishes the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

The acknowledgement transcends the reality of formality when we'd like to express deep gratitude and respect, so all those people behind the screen who guided, inspired and helped us for the completion of our project work.

First and foremost our sincere thanks to Prof. Dr. Satya Prakash Ghrera (Head of Dept. of CSE, JUIT), Dr. Nitin (Project coordinator and assistant Lecturer, JUIT) for his permission and grants to undertake the project and also for his precious guidance. We are also thankful to "Dr. Pradeep kumar Gupta (Senior Lecturer, JUIT)", to guide us to complete the project. We would also extend my gratitude to all our faculty members and my friends for their support and cooperation, which help us to carry out this final report.

We also thank them for their enthusiasm, wide support, co-operation and valuable they gave us in order to go ahead with the project successfully.

Jivesh.

Jivesh Sharma(091227)

Mansi Sahi

Mansi Sahi(091225)

TABLE OF CONTENTS

List of Figures

FIGURE **PAGE NO.**

List of Definitions, Acronyms and Abbreviations.

Terms	Definitions
User	User of the system
Guest	User of the system
Admin	The administrator and the moderator of the system.
HTTP	Hypertext Transfer Protocol is a transaction oriented client/server protocol between web browser & a Web Server
HTML	Hyper text markup language
SDD	Software Design Document
ODMCS	Online Document Modification, Conversion and Sharing Find doctor
JSP	Java Server Pages
FTP	File Transfer Protocol
PHP	Hypertext Preprocessor
FAQ	Frequently Asked Questions
Css	cascade style sheet
www	World Wide Web
DSS	Decision Support System
UI	User Interface
ODBC	Open Database Connectivity
DLL	Dynamic Link Library
IIS	Internet Information System
OLE DB	Object Linking and Embedding Database
WBIS	Web Based Information System
SDLC	Software development life cycle
WISDM	Web Information System Development Methodolgy
UML	Unified Modeling Language
ASP	Active Server Pages
SDM	System Development Method
SQL	Structured Query Language
DBMS	Database Management System
.aspx	Filename extension for ASP.net Files
.cs	Visual C#(Sharp) Source code file
HCI	Human Computer Interface

List of Tables

Chapter 1

Introduction of the project: It covers an overview of the project and includes the Problem statements, project objectives, scope of the project, project methodology and expected outcomes of the project.

Chapter 2

Literature review: This chapter consist of literature review and background material regarding the overall aspects of Find Doctor. The first section began with a definition of Find Doctor followed by their history.

Chapter 3

Requirement of the project: This chapter explains the requirements and specifications of the project.

Chapter 4

Software development: This chapter explains the technology used for the software development, the tools used, and the methodology of the project

Chapter 5

Implementation of design: This chapter discusses implementation of design of the project, including the requirements, interface, database, etc. Screen shots of various web modules and database have been shown in this chapter.

Chapter 6

Conclusion: This chapter presents the conclusion of the whole study. It presents a short summary of the outcomes of the project, the strength and limitation of developed system and there commendation for future research and development.

CHAPTER 1 - INTRODUCTION

1.1 Introduction

There are hundreds of websites which will give all the information you need about a doctor to choose from many. For example if you have eye problem such as refractive error of eye , you will need an ophthalmologist or eye doctor, to get yourself treated, conservatively or surgical correction of your vision problem. Out of hundreds of websites which are providing service of finding a good doctor online, you need to choose a good website to find a doctor of your required quality and experience and qualification. Many of the websites makes it simple to find quality doctor as these websites are there to provide quality service in finding a doctor online as well as providing healthcare service to the needy patients. But is it really so?

Let's check out. The results of doctors and health experts lists is random, and anonymous opinions don't truly measure quality of a good physician according to specific case of the patient. So we decided to make a project that would make doctor search concise and easy and according to the disease/need of the patients.

This web application is developed to provide the facilities to various users. In this developed application there are three main users, which have the different privileges. These users can perform the various operations through this application. In this application the provided facilities according to the user type are as follows:

- **Administrator** has the following facilities-
  - Administrator can add new doctor\patient.
  - Administrator can provide username and password to doctor to change their accounts.
  - Administrator can delete the existing doctor's account.
  - Administrator can edit the existing doctor's account.

- **Doctor** has the following facilities-
  - Doctor can edit its account.
  - Doctor can upload the documents.
  - Doctor can view their profile.

- **User** has the following facilities-
  - User can view the particular doctor's profile.
  - User can view the particular doctors list.
  - User can download the files from the respective doctor's profile.
  - Users contact with doctors by provided their contacts.
  - Users can rate the doctor.

1.2 Motivation

Web portal:- A web portal is a web site that brings information from various sources in a unified way. Usually, each information source gets its dedicated area on the page for displaying information, often , the user can configure which ones to display.
Web portals providing doctor information are Just dial, Ask doctor, Aol, Webmd etc.
They provide information about worldwide doctors along with location oriented search, specialty oriented search, name oriented search etc.
But they don't provide Problem oriented search or query based search. This motivated us to make a project which combines all the features of the existing web portals which provide information about doctors and add to it the feature of problem or query related search.

1.3 Problem Statement

To create an online application which provides and maintains, details of different health professionals, case studies of practitioners, search by location, search by name, search by specialty, search by disease.

Find Doctor has become more important than ever because of the need to get access to find information and to get services of the doctors, through the Internet. Find Doctor helps to come out with the problem of searching doctors through various website for various diseases but this project has been developed as solution to the problems in a concatenated form.

The problems to address in this project would be on how to improve services of the doctor by using the internet. Some issues pertinent to this project to consider include:

Most patient do not have enough basic knowledge on how to select a specialized doctors available according to the categories of their problems .

User expectations in terms of treatment services specifications provided by the doctor in the mean time as provided by other doctors.

Often user cannot find the service as expected from the available doctor on this project
Many new doctors filling their details, who become member of Find Doctor fill some false detail because they do not have to submit their proof identification on this project.
When people have technical questions related to a health issue, professionals holds a way is not provided in easy and efficient way. When a situation involves more personal issues of how to cope with a health issue or get quick relief, then user faces lot problems involved in the project.

1.4 Project Objectives

This project aims to develop a website that will allow the user to find for suitable doctor on the basis of their specialty, sub specialty experience and location from various location within seconds. This project saves our time and help to get the services on time as required by the patients relevant data and information.

The objectives of this project are:

I. To design, implement and develop a knowledge management system.

II. To identify the needs of the user which will be incorporated in a formatted way through knowledge management system with updated information.

III. To understand the meaning, features and categories of Find Doctors as a knowledge management system.

IV. To upload doctors detail in a relevant way that allows user to get services as per required or whenever looking for details of doctor.

V. This Online Query Processing System is able to search for doctor various skills and experiences of the doctor.

The above objectives cover the overall study of the Find Doctor, and the significance of the objectives is to meet the user and doctors need for a system that contains all specific details of doctors availability and meet the users relevant needs.

This web application is developed to provide the facilities to various users all over the world. In this developed application there are three main categories on the basis of which user can search for doctor and they are specialty, sub specialty and experience. It is the project which also provide the different ways to find the doctor according to the various skills and experiences of the doctor.

These Administrator also get the privilege to add themselves to this web application and perform the various operations as per the user requirement through this application. In this application it also provide the facility to search for doctor within specific range as required by the user.

Find Doctor has following facilities:

- New Doctors detail can be added to the existing list.
- User can search for doctor within the required range by specifying the range for which the user want to search for doctor.
- Find Doctor also provide username and password to members on registration.
- Categories and sub categories of the doctor can also be added to the record.
- User can search for specialist doctor on the basis of the disease.
- Gives quick and efficient solution to the problems.

1.5 Organization of the project report:

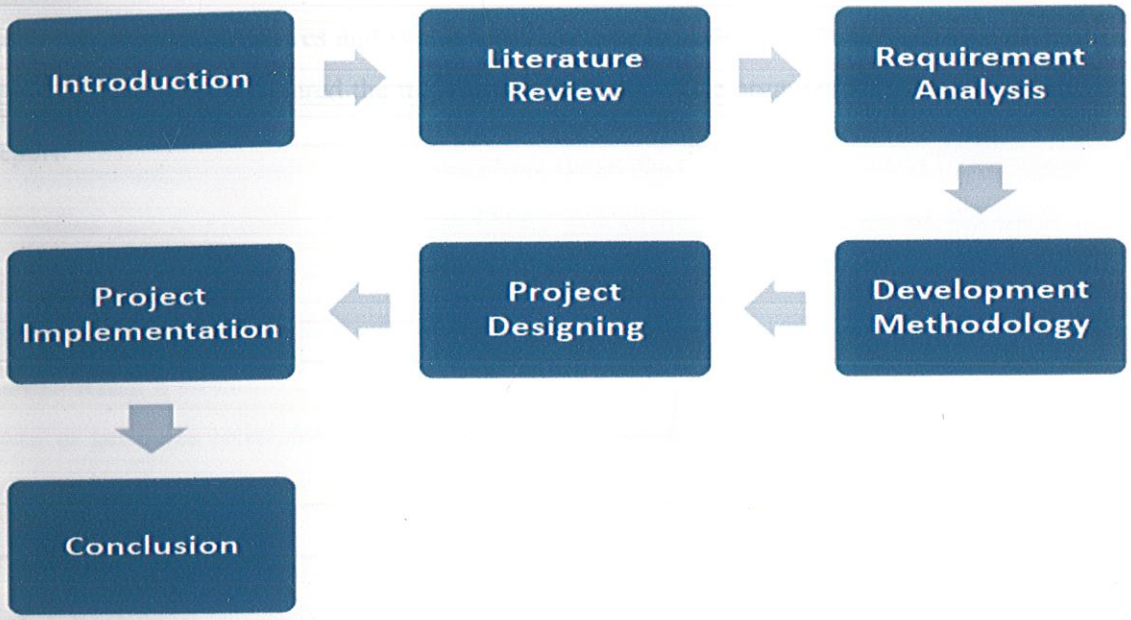


Fig 1.1:Organisation of project report

Introduction of the project: It covers an overview of the project and includes the Problem statements, project objectives, scope of the project, project methodology and expected outcomes of the project.

Literature review: This phase consist of literature review and background material regarding the overall aspects of Find Doctor.

Requirement of the project: This phase explains the requirements and specifications of the project.

Software development: This phase explains the technology used for the software development, the tools used, and the methodology of the project.

Implementation of design: This phase discusses implementation of design of the project, including the requirements, interface, database, etc.

Conclusion: This chapter presents the conclusion of the whole study. It presents a short summary of the outcomes of the project, the strength and limitation of developed system and there commendation for future research and development.

1.6 Conclusion

This chapter gives a background of the issues pertinent to the project. It presents the problems which find doctor aspirants and doctor regarding access to the required information. The objectives and scope of the project are also presented in this chapter. The chapter has also covered the methodology used and the organization of the project report.

CHAPTER 2 - LITERATURE REVIEW

2.1 Introduction

The World Wide Web (WWW) has greatly impacted many aspects of our lives, such as in communication, business and education, these days (Yang et al., 2005).In the age of online communication, the Online Query processing System is one of the most important project as need of today.

When people have technical questions related to a health issue, professionals hold away. When a situation involves more personal issues of how to cope with a health issue or get quick relief, We can find for doctor in a filtered way either by the diseases category by the experience or by the skills in a systematic way.

2.1.1 Problem Statements

Online Query processing System has become more important than ever because of the need to get access to find information and to get services of the doctors, through the Internet. Online Query processing System helps to come out with the problem of searching doctors through various website for various diseases but this project has been developed as solution to the problems in a concatenated form.

The problems to address in this project would be on how to improve services of the doctor by using the internet. Some issues pertinent to this project to consider include:

- To create an online application which provides and maintains:
    - Details of different health professionals.
    - Case studies of practitioners.
    - Search by location.
    - Search by specialty.

- Search by disease.

2.1.2 Project objectives

This project aims to develop a website that will allow the user to find for suitable doctor on the basis of their specialty, sub specialty and experience from various location within seconds. This project saves our time and help to get the services on time as required by the patient relevant data and information.

The objectives of this project are:

- To design, implement and develop a knowledge management system.
- To identify the needs of the user which will be incorporated in a formatted way through knowledge management system with updated information.
- To understand the meaning, features and categories of Online Query processing System as a knowledge management system.
- To upload doctors detail in a relevant way that allows user to get services as per required or whenever looking for details of doctor.
- This Online Query processing System is able to search for doctor various skills and experiences of the doctor.

The above objectives cover the overall study of the Online Query processing System, and the significance of the objectives is to meet the user and doctors need for a system that contains all specific details of doctors availability and meet the users relevant needs.

This web application is developed to provide the facilities to various users all over the world. In this developed application there are three main categories on the basis of which user can search for doctor and there specialty, sub specialty and experience. It is the project which also provide the different ways to find the doctor according to the various skills and experiences of the doctor.

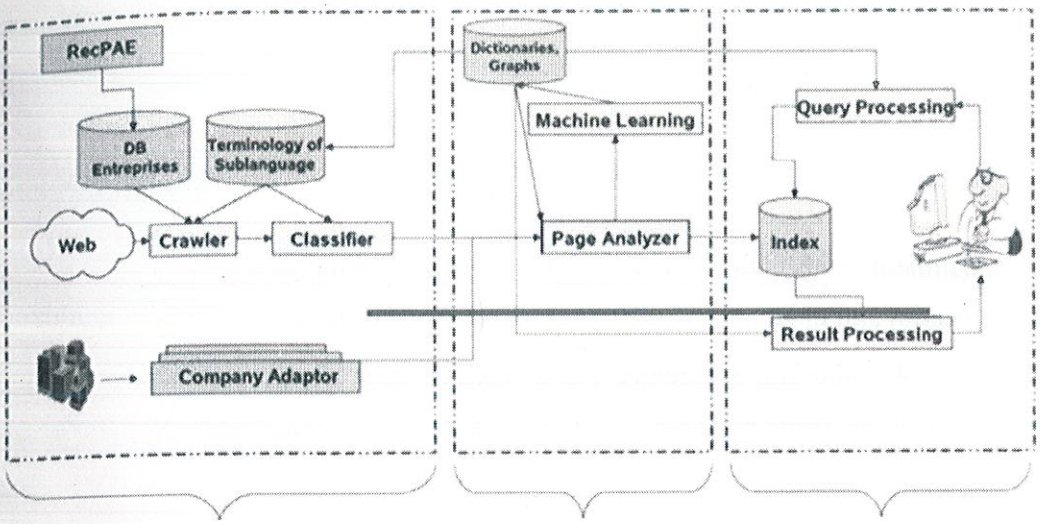


Fig 2.1: Online Query Processing System Architecture

2.2 Find Doctor Searching Management System

Online Query processing System management system help people to search for doctor by following different steps. It contains searching on the basis of the specialty of the doctors that are categorized according to the Sub-Specialty and experience of the doctors.

2.2.1 Specialty

Online Query processing System management helps doctors specialty by which the user can get the suitable doctor on their needs by creating account on Online Query processing System. Different doctor are easily available on the web all together and also doctor are specialist in various diseases

2.2.2 Sub Specialty

Online Query processing System management helps doctors specialty by which the user can get the suitable doctor on their needs by creating account on Online Query processing System. Different doctor are easily available on the web all together and also doctor are specialist in various diseases.

This project saves our time and help to get the services on time as required by the patient relevant data and information.

2.2.3 Experience

The most commonly-researched specific diseases or conditions; treatments or procedures; and doctors or other health professionals are available in this project. New doctors can feed their details such as their skills, experience and other details about them on Online Query processing System and can made themselves available for other.

People turn to different sources for different kinds of information about any of the problem they face so, Online Query processing System is best and easy service provider for various problems. Online Query processing System is a project helps user too search for any of the best doctor by their specialty, sub specialty and experience from various location within seconds. This project saves our time and help to get the services on time as required by the patient. This project is also useful for adding new doctors to the existing list. It also specifies the range within which we can find the doctor according to the various skills and experiences of the doctor.

2.3 Searching Find Doctor

Find Doctor is a project that helps user too search for any of the best doctor by their specialty, sub specialty and experience from various location within seconds. This project saves our time and help to get the services on time as required by the patient. This project is also useful for adding new doctors to the existing list. It also specifies the range within which we can find the doctor according to the various skills and experiences of the doctor.

When people have technical questions related to a health issue, professionals hold sway. When a situation involves more personal issues of how to cope with a health issue or get quick relief, We can find for doctor in a filtered way either by the diseases category , by the experience or by the skills in a systematic way.

2.3.1 Search By Specialty

This project aims to develop a website that will allow the user to find for suitable doctor on the basis of their specialty, sub specialty and experience from various location within seconds. This project saves our time and help to get the services on time as required by the patient relevant data and information.

To design, implement and develop a knowledge management system. To identify the needs of the user which will be incorporated in a formatted way through knowledge management system with updated information.

To understand the meaning, features and categories of Online Query processing System as a knowledge management system.

2.3.2 Search By Experience

This Find Doctor is able to search for doctor various skills and experiences of the doctor.

To upload doctors detail in a relevant way that allows user to get services as per required or whenever looking for details of doctor. The above objectives cover the overall study of the Find Doctor, and the significance of the objectives is to meet the user and doctors need for a system that contains all specific details of doctors availability and meet the users relevant needs.

This web application is developed to provide the facilities to various users all over the world. In this developed application there are three main categories on the basis of which user can search for doctor and they are specialty, sub specialty and experience. It is the project which also provide the different ways to find the doctor according to the various skills and experiences of the doctor.

2.3.3 Search By Location

These doctors also get the privilege to add themselves to this web application and perform the various operations as per the user requirement through this application. In this application also provide the facility to search for doctor within specific range as required by the user. Find Doctor has facilities as New Doctors detail can be added to the existing list. User can search for doctor within the required range by specifying the

range or which the user want to search for doctor. Find Doctor also provide username and password to members on registration. Categories and sub categories of the doctor can also be added to the record. User can search for specialist doctor on the basis of the disease. Gives quick and efficient solution to the problems.

2.4 What is Online Query processing System?

Online Query processing System is a project that understands the health of peoples and helps to connect to various doctors within specified range. What Online Query processing System do impacts both patient and doctor nearby. So Online Query processing System is dedicated for delivering better care to members by providing greater value to members and helping improve the health of our communities.

The most commonly-researched specific diseases or conditions; treatments or procedures. This project saves our time and help to get the services on time as required by the patient. This project is also useful for adding new doctors to the existing list. These doctors also get the privilege to add themselves to this web application and perform the various operations as per the user requirement through this application.

In this application also provide the facility to search for doctor within specific range as required by the user. Often user cannot find the service as expected from the available doctor on this project. Many new doctors filling their details, who become member of Online Query processing System fill some false detail because they do not have to submit their proof identification on this project.

It is the project which also provide the different ways to find the doctor according to the various skills and experiences of the doctor. These doctors also get the privilege to add themselves to this web application and perform the various operations as per the user requirement through this application.

2.5 Importance of Online Query Processing System

In the age of technology, the Internet has become the main source of information for the user. Good hospitals are available at various places in our society but Online Query processing System is a web facility by which we can easily search for doctor for quick result.

This websites or project provide a search engine to access information about doctors on user request. The reason why more peoples throughout the countries is that they want quick result to fulfill the requirement without wasting time as it is time saver. It is useful for doctors that they can upload their details on net. These methods result in great saving in costs. It is also stated that the use of this web technology gives higher speed of procurement but reduced transaction cost.

Today, the Internet is used for a large number of business transactions. People find the Internet to be an effective communication tool. An online availability of doctors through Online Query processing System has the following advantages: user can identify a large number of

eligible doctors and get their information easily. It means their experience and their skills. Internet provides user way to get a good doctor at specified range and especially, those who fulfill the requirements. With online Online Query processing System, people can access information from anywhere in the world.

2.6 Features of Online Query Processing System Management System

- New Doctors detail can be added to the existing list.
- User can search for doctor within the required range by specifying the range for which the user want to search for doctor.
- Online Query processing System also provide username and password to members on registration.
- Categories and sub categories of the doctor can also be added to the record.
- User can search for specialist doctor on the basis of the disease.
- Gives quick and efficient solution to the problems.

CHAPTER 3 - REQUIREMENT ANALYSIS

3.1 Project Requirements

Following is a list of functionalities of the system. More functionality that you find appropriate can be added to this list. And, in places where the description of functionality is not adequate, you can make appropriate assumptions and proceed.The Doctors can add their detail to the existing list of doctor.The requirements are as follows:

1. Generic:

a. Login to the system through the first page of the application consisting of two logins that is user login and admin login.
b. Create an account by passing user name and password.
c. Button to login after filling required data.

2. Requirements:

a. Doctors should be experienced.
b. Categories and sub categories should be mentioned.
c. Location should be filled for easy searching.
d. Disease name should be mentioned.
e. Range of search for doctor should be mentioned.

3. Administrator:

a. Should be able to add new doctor details.
b. Detail should be available for each account holder.
c. Should be able to search on basis of range specification.
d. Should be able to view the details by each user.
c. Should be able to be visible by experienced based.

4. Other details:

a. Records can be searched on basis of Search By Specialty of the doctors.

b. Records can be searched on basis of Search By Experience of the doctors.

c. Records can be searched on basis of Search By Location of the doctors.

d. Records of the doctor should be unique and cannot be changed later on.

e. Records once filled by doctors cannot be changed later on.

3.2 System Requirements

The design of the Online Query processing System takes into consideration the requirements identified from the research on internet and from the literature review. The successful running of any project primarily depends upon hardware and software used in its compilation. The hardware used in the machine should be such that it supports the software that is to be mounted for assembling the project. This project deals with the hardware and software, which is available readily and easy on each and every machine given to the user.

3.2.1 Hardware Requirements

Number	Description	Alternatives(If available)
1.	300 MHz Pentium Processor	Not-Applicable
2.	256 MB RAM	
3.	40 GB hard disk	
4.	1.44 MB Floppy Drive	
5.	CD-ROM Drive	
6.	16-Bit color display, 640x480 resolution	

Hardware Specifications:

Processor:

Minimum: 300 MHz Pentium Processor

The processor does all the processing for the contents of the program and when the clock speed increases so does the processing speed.

Memory Requirements:

Minimum: 256 MB RAM

The memory selection is done and preferred for higher memory because the program before running is flushed into memory buffer of computer, then it is executed. More the memory, more will be the speed and hence less time for execution.

Display Card:

Higher the memory display card better is the resolution and better is the experience of using the application.

3.2.2 Software Requirements

Number	Description	Alternatives(If available)
1.	Microsoft Visual Studio 2010(Developing	Visual Basic 6.0
2.	Tool) Microsoft SQL Server 2008	Oracle 8, SQL Server 2000
3.	Windows XP	Alternatively, Windows 98
4.	C#(Front end)	with IIS Server Installed.

Software Specifications:

Software plays an important role in any project development. One should understand that which software he/she should use to develop the project. Window XP was used as the operating system.

The application has been developed using:

- Front End: C#
- Back End: Microsoft SQL Server
- Picture Tool: Photoshop

3.3 System Design and Requirements

3.3.1 The tools used in developing the system

The choice of tools to use to develop a system is critical, as this will eventually influence the quality and efficiency of the system. Hence, it is important for a programmer to select suitable tools for designing and developing the system. In this project, ASP.NET was selected for web programming and SQL server 2008 for designing the database. ASP.NET was developed by Microsoft for building dynamic websites, web applications, and web services. Version 1.0- was published in 2002 to overcome some limitations of the Active Server Pages (ASP) technology, and it became an improved replacement for the ASP framework. ASP.NET is a web application framework that provides for the creation of server-side applications based on HTML, XML and SOAP. The Visual Studio software was used to develop ASP.NET. It is web application software to design the user interface (UI) very easily. Visual Studio 2010 was used to develop the Online Query processing System for this project. The web application of ASP.Net involves the state of application, session and the use of cookies. ASP.Net and PHP are well-known for web programming in the market. However, both software have almost similar functions and also in their implementation, but they have some differences, also. For example, PHP supports only C and C++ style scripting languages with older ASP style markup as its coding language, while ASP.Net supports more than 25 languages, and two of them – C# and Visual basic.Net - are the most commonly used. C# was selected for developing the Online Query processing System in this project. PHP is associated with most commonly-used operating systems such as Windows, Mac, Solaris and Linux, but

ASP.Net supports Windows 2000, Windows XP and Windows Server 2003. C#, ASP.Net is selected for this system.

The database is developed using SQL Server 2008. This version of SQL Server was released in 2008 as the successor of SQL Server 2000. It fulfills the need for data management and it increases performance and application programmability. SQL Server 2008 presents the SQL Native Client, which involves an updated SQL Open Database Connectivity (ODBC) drivers and SQL OLE database (OLEDB) is provided with the network libraries in a single dynamic-link library (DLL). It supports the association with the Microsoft.NET framework that covers the latest ADO.NET version. Databases are connected to the web Online Query processing System with an ODBC on the web server. The web server which is selected for this find doctor is Internet Information Service (IIS), which is a collection of Internet-based services such as Hypertext transfer protocol or file transfer protocol that is integrated with Microsoft Windows NT, and Windows Server 2000 operating systems.

3.3.2 Product perspective

This software is totally self contained and works relatively as efficient as other packages related to the subject. It provides simple database rather than complex ones for high requirements and it provides good and easy graphical user interface to both new, naive as well as experienced users of the computers.

3.3.3 System Interfaces :

The software provides good graphical interface for the front end of the database and a good informative interface for the rear end

Hardware Interface:

The system should have these hardware requirements:

- The processor should be at least Pentium 3 or above
- The processor speed should be greater than 400Mhz
- The video device should support graphics
- Ram should be or greater than 120 mb

Software Interfaces :

The software requires the support of the following softwares for the database and other requirements

- Visual Studio editor for web interface(C#)
- Microsoft SQL Server 2008 for database
- Server(windows 2000,apache or…)

Communication Interfaces :

Local intranet and internet protocols.

Supports all HTTPS,SMTPS and POP3 services

3.3.3 User characteristics

- No pre knowledge of html
- No pre knowledge of database management
- Should be familiar with internet
- Should know English
- Should be able to use and do according to the graphical user interface

3.3.4 Assumptions and Dependencies

The product assumes that the users don't opt for the same product number simultaneously. Cannot support multiple user interfaces.

3.3.5 Performance Requirements

Performance requirements are:

1. Good working pc with all the requirements as stated in the hardware interfaces
2. Works for medium size information databases
3. Should not be overloaded

3.3.6 Future Developments

- multiple user interface
- support for large database
- Oracle for database

CHAPTER 4 - SYSTEM DEVELOPMENT METHODOLOGY

4.1 Introduction

Online Query processing System requirements, interface, etc. This chapter discusses the system development methodologies which are suitable for this project. The purpose of this project is to develop a web to Online Query processing System. A web Online Query processing System is a gateway to online information and services that provide users with: a search function, community features, personal applications, and a way for communication between the user and doctor. This web Online Query processing System to be developed is intended to share information with find doctor, and to assist find doctor in their find doctor search by establishing links between them and related problems can be solved. This means that this knowledge management system, should also serve as online recruitment system. Today, most Online Query processing System add career links to their find doctors to allow find doctor seekers to obtain information on users and career prospects in the industry. The web find doctor can guide the find doctor aspirants to search their related find doctor opportunities available, and the type of skill sets needed by the industry. The different methodologies used for developing information systems. He identified levels, sub-levels and the technologies used in each level. He recommended that there should be planning management, evaluation and control at the different levels and sub-levels. Before the web based technology was available, old information systems were developed using computer–based systems. The main issue at that time was that the organizations were willing to spend large amount of money for information system development to stay competitive. Today, web technology and the Internet have changed the traditional way of developing information systems to become what is called Web-based Information Systems (WBIS). Online Query processing System can be a type of Web-based Information Systems.

4.2 System Development Methodology

System development methodology is the process or framework involved in planning, designing, and implementing an information system. Different approaches have evolved over the years, each with its own features. Generally, the system development methodology also involves the techniques, tools and documentation that are used by system analysts, developers, designers and users to develop and implement an information system, successfully. It is very important for developers to consider the time-frame and quality assurance aspects of the system when choosing a system development methodology. Traditional system development methodologies can be classified into 5 main categories:

☐ Waterfall

☐ Prototyping

☐ Incremental

☐ Spiral

☐ Rapid Application Development (RAD)

4.3 Traditional system development Methodology Used

Spiral Model

The Spiral Life Cycle Model is a type of iterative software development model which is Generally implemented in high risk projects. It was first proposed by Boehm. In this system development method, it clubs the features of both, waterfall model and prototype model. In Spiral model we can arrange all the activities in the form of a spiral. Each loop in a spiral represents a development phase. Each loop has four sections or quadrants:

- . To determine the objectives, alternatives and constraints. We try to understand the product objectives, alternatives in design and constraints imposed because of cost, technology, schedule, etc.

- Risk analysis and evaluation of alternatives. Here we try to find which other approaches can be implemented in order to fulfill the identified constraints. Operational and technical issues are addressed here. Risk mitigation is in focus in this phase. And evaluation of all these factors determines future action.
- Execution of that phase of development. In this phase we develop the planned product. Testing is also done. In order to do development, waterfall or incremental approach can be implemented.
- Planning the next phase. Here we review the progress and judge it considering all parameters. Issues which need to be resolved are identified in this phase and necessary steps are taken. Subsequent loops of spiral model involve similar phases. Analysis and engineering efforts are applied in this model. Large, expensive or complicated projects use this type of life cycle. If at any point of time one feels the risk involved in the project is a lot more than anticipated, one can abort it. Reviews at different phases can be done by an in-house person or by an external client. Spiral model is also called meta model.

Spiral model is also called as meta-model because in a way it comprises of other models of SDLC. Both waterfall and prototype models are used in it. Here we do software development systematically over the loops (adhering to waterfall approach) and at the same time we make a prototype and show it to user after completion of various phases (just in case of prototype model). This way we are able to reduce risks as well as follow systematic approach. The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model (or spiral development), it is a systems development method (SDM) used in information technology (IT). This model of development combines the features of the prototyping and the waterfall model.

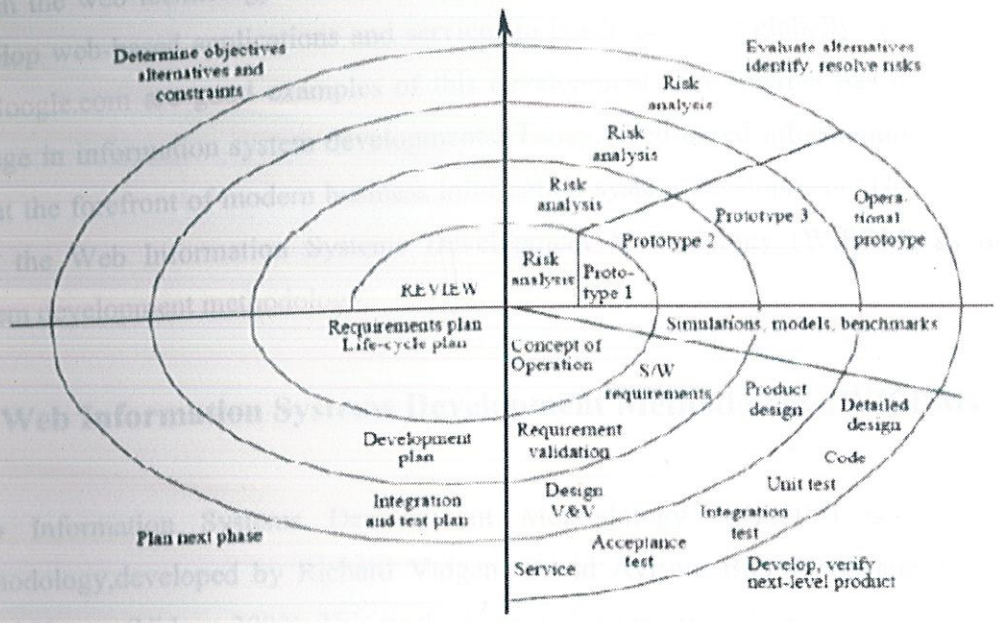


Figure 4.1: Spiral Life Cycle Model

Advantages of Spiral Model

- Spiral Life Cycle Model is one of the most flexible SDLC models in place. Development phases can be determined by the project manager, according to the complexity of the project.
- Project monitoring is very easy and effective. Each phase, as well as each loop, requires a review from concerned people. This makes the model more transparent.
- Risk management is one of the in-built features of the model, which makes it extra attractive compared to other models.
- Changes can be introduced later in the life cycle as well. And coping with these changes isn't a very big headache for the project manager.
- Project estimates in terms of schedule, cost etc become more and more realistic as the project moves forward and loops in spiral get completed.
- It is suitable for high risk projects, where business needs may be unstable.
- A highly customized product can be developed using this.

When the web technology became available, organizations and companies started to develop web-based applications and services to reach out more globally. Yahoo.com or Google.com are good examples of this development. The internet has led to the change in information system developments. Today, Web-based information systems are at the forefront of modern business information system development. This project uses the Web Information Systems Development Methodology (WISDM) as the system development methodology.

4.4 Web Information Systems Development Methodology (WISDM)

Web Information Systems Development Methodology (WISDM) is an ISD methodology,developed by Richard Vidgen, David Avison, Bob Wood and Trevor Wood-Harper (Vidgen 2002). This method adapted the traditional system development methods, web development technology and the hypermedia development methodology. Hypermedia is a mix of rich texts, graphics, audio and video, and uses hyperlink to link to other pages and sections of an application. The main framework of WISDM is extracted from Multitier. Multitier is a methodology with user participative approach that includes many stakeholders like computer experts who are responsible for developing the system and users who are using the system. Therefore, Multitier focuses on both the human and technical aspects of Information System. The framework of the WISDM that helps in the development of a Web-based Information Systems considers two aspects: one relating to the organizations, people and technology; the other relating to the analysis and design.

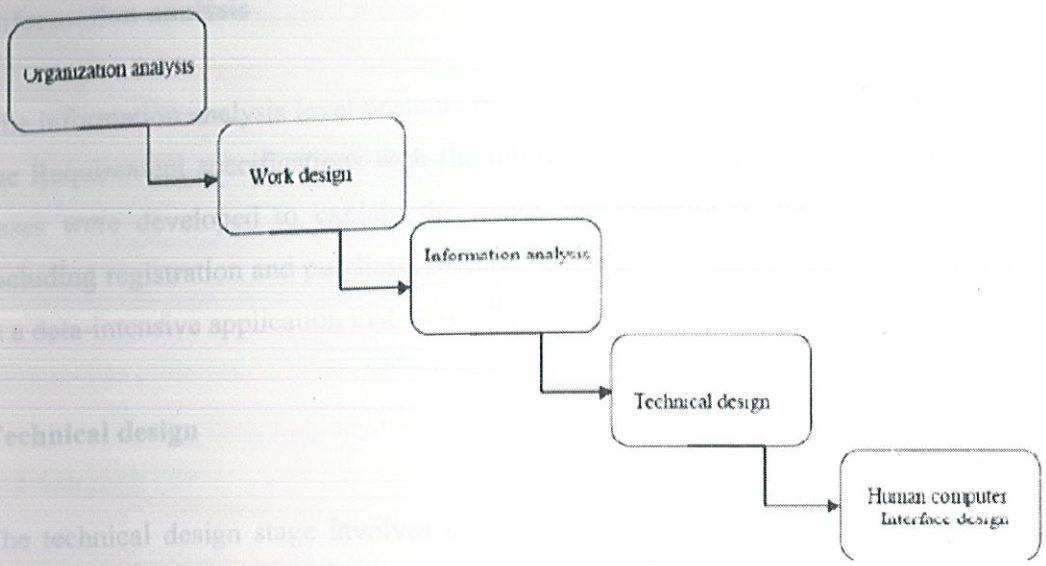


Figure 4.2: Stages in WISDM

Organization analysis

The first stage, organization analysis originated of building an e-commerce strategy and Conducting a market survey. The e-commerce survey is focused on aligning the development project. The market survey concerned with customers and involved a postal and telephone survey to identify attitudes to the Internet, e-commerce, and confirm their project information requirements. Organization analysis, involves the creation of values.

Work design

In work design stage, the programmer plans an approach to socio-technical design. It is concerned with achieving a suitable match between find doctor satisfactions – Online Query processing System's expectations and the Online Query processing System requirements as defined by management –and the efficiency objectives of the organization.

Information analysis

The information analysis level presents the needs of the users; the third stage involves the Requirement specifications with the unified modeling language, UML. UML use cases were developed to explain the major functionality of the proposed system, including registration and purchase, research queries, and maintenance. Given that this is a data-intensive application that would be implemented around a relational database.

Technical design

The technical design stage involves the design of the software model. The physical requirements of the implementation were clear; a database was needed together with some technology to link the database to the web.

Human computer interface (HCI) design

In the human computer interface (HCI) design stage involves the development of the user interface. HCI design is located in an area overlapping software design and work design as it needs to draw on both. (Vidgen, 2002).

4.5 Conclusion

Web Information Systems Development Methodology (WISDM) is selected as system Development methodology for this project. WISDM is a new information system development that mixes the traditional methods with the web development technology.

CHAPTER 5 – SYSTEM DESIGN

5.1 Introduction

Based on information from the literature review, as well as the identified Online Query processing System aspirants' requirements, a new web Online Query processing System was proposed. This project is helpful for user finding the good doctors quickly.

5.2 Objectives

Its objective cover the overall study of the Online Query processing System, and the significance of the objectives is to meet the user and doctors need for a system that contains all specific details of doctors availability and meet the users relevant needs.

This web application is developed to provide the facilities to various users all over the world. In this developed application there are three main categories on the basis of which user can search for doctor and they are specialty, sub specialty and experience. It is the project which also provide the different ways to find the doctor according to the various skills and experiences of the doctor.

These doctors also get the privilege to add themselves to this web application and perform the various operations as per the user requirement through this application. In this application also provide the facility to search for doctor within specific range as required by the user.

5.3 Scope

The main scope of this chapter concerns the selection of tools used and the requirements.When people have technical questions related to a health issue, professionals hold sway. When a situation involves more personal issues of how to cope with a health issue or get quick relief, We can find for doctor in a filtered way either by the diseases category , by the experience or by the skills in a systematic way.

5.4 Architecture Design:

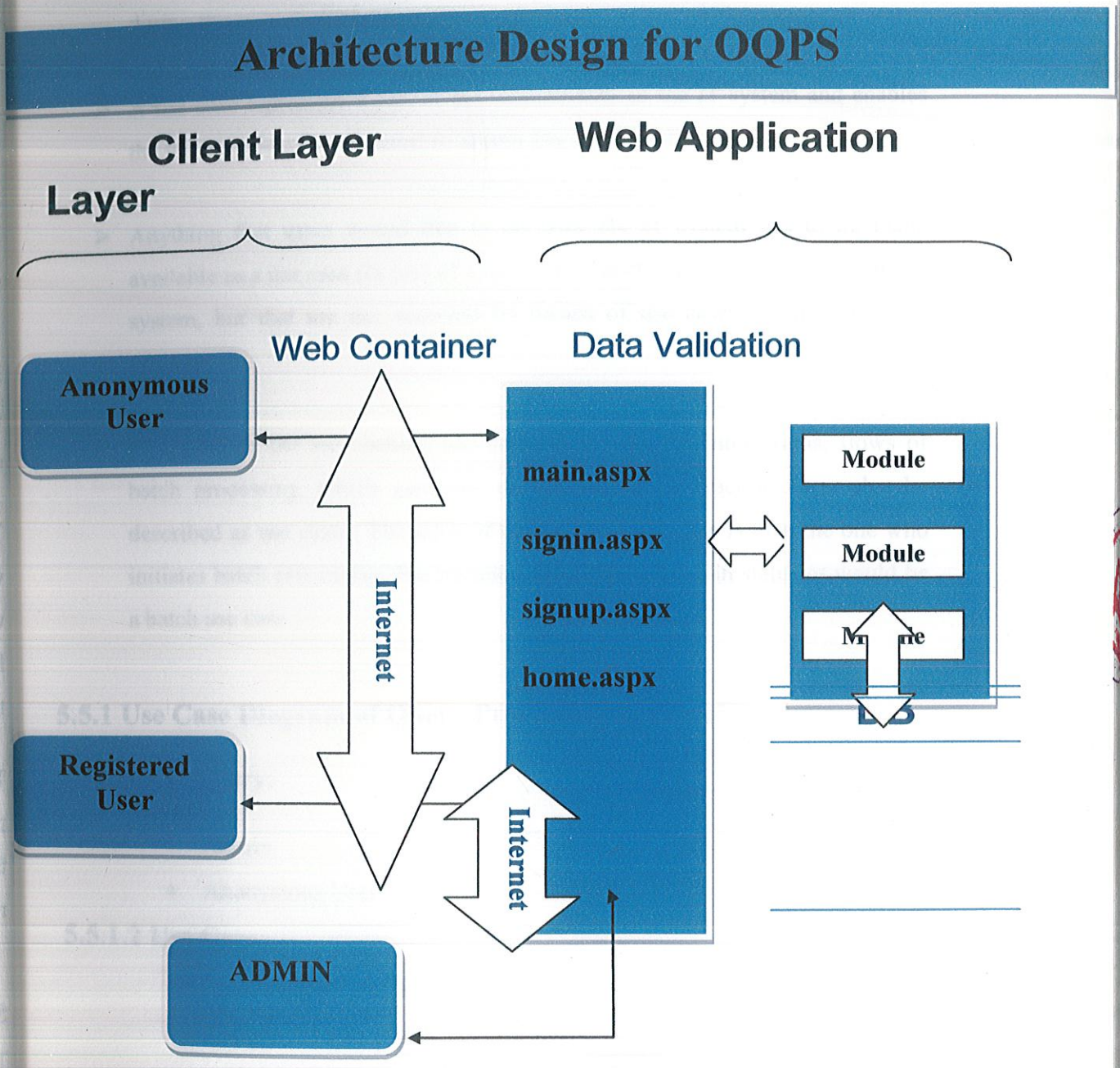


Fig 5.1: Architecture of system

5.5 Use Case Diagrams:

- Use cases describe the interactions that take place between actors and system during the execution of processes:

- A use case represents a part of the functionality of the IT system and enables the user (modeled as an actor) to access this functionality.

- Anything that users would like to do with the IT system has to be made available as a use case (or part of a use case). Functionalities that exist in the IT system, but that are not accessed by means of use cases, are not available to users.

- Even though the idea behind use cases is to describe interactions, flows of batch processing, which generally do not include interactions, can also be described as use cases. The actor of such a batch use case is then the one who initiates batch processing. For instance, generating check-in statistics would be a batch use case.

5.5.1 Use Case Diagram of Query Processing System:

5.5.1.1 Actors:

- Admin
- Anonymous User

5.5.1.2 Use Case:

- View Doctors profile
- Manage Doctors profile
- Add Case Study
- Database Management
- Search/Rate doctor
- Feedback

In software and systems engineering, a **use case** (a case in the use of a system) is a list of steps, typically defining interactions between a role (known in UML as an "actor") and a system, to achieve a goal. The actor can be a human or an external system.

USE CASE DIAGRAM

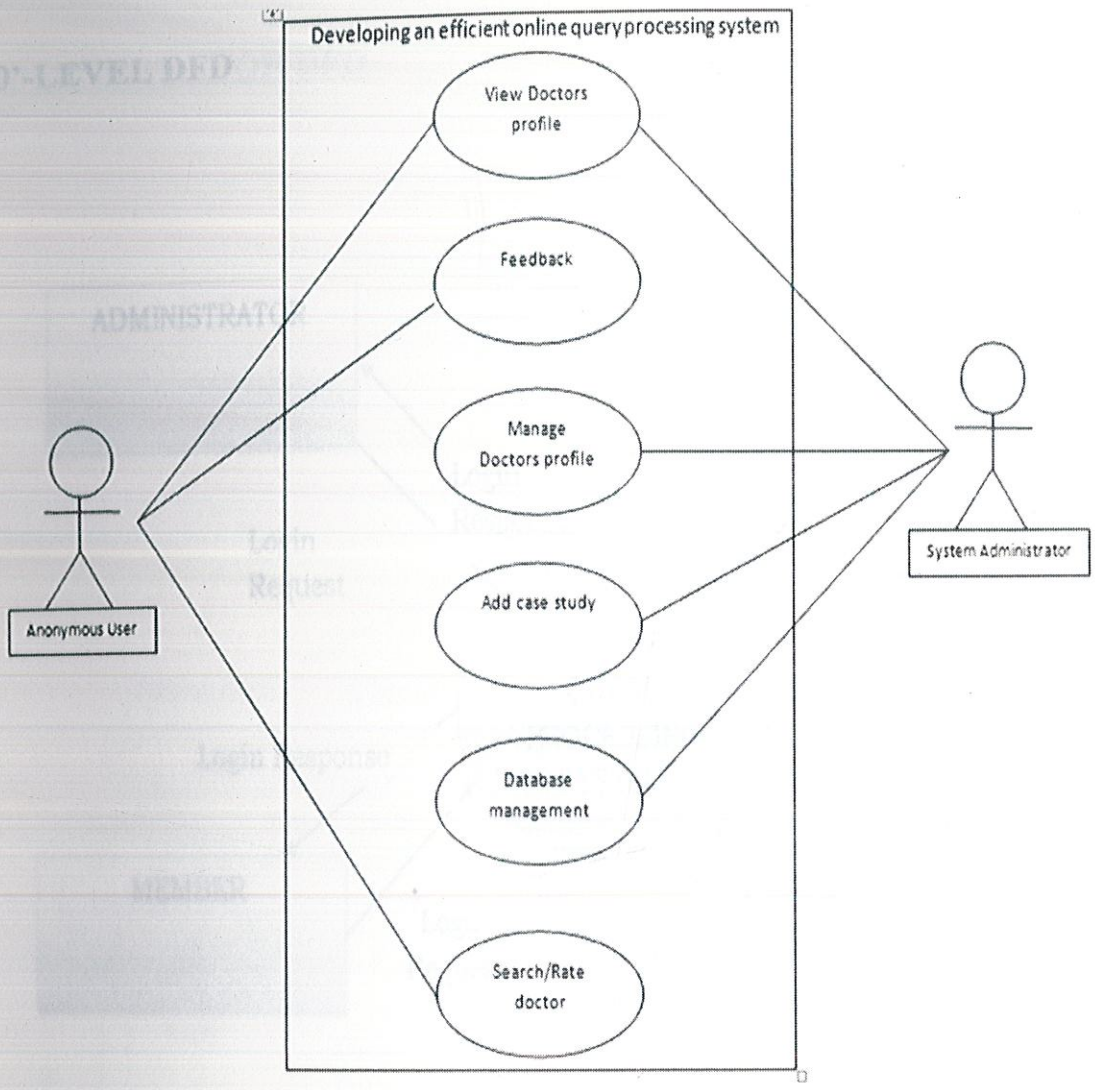


Fig 5.2:Use case diagram of DOQPS

5.6 DFD (Data Flow Diagram)

'0'-LEVEL DFD

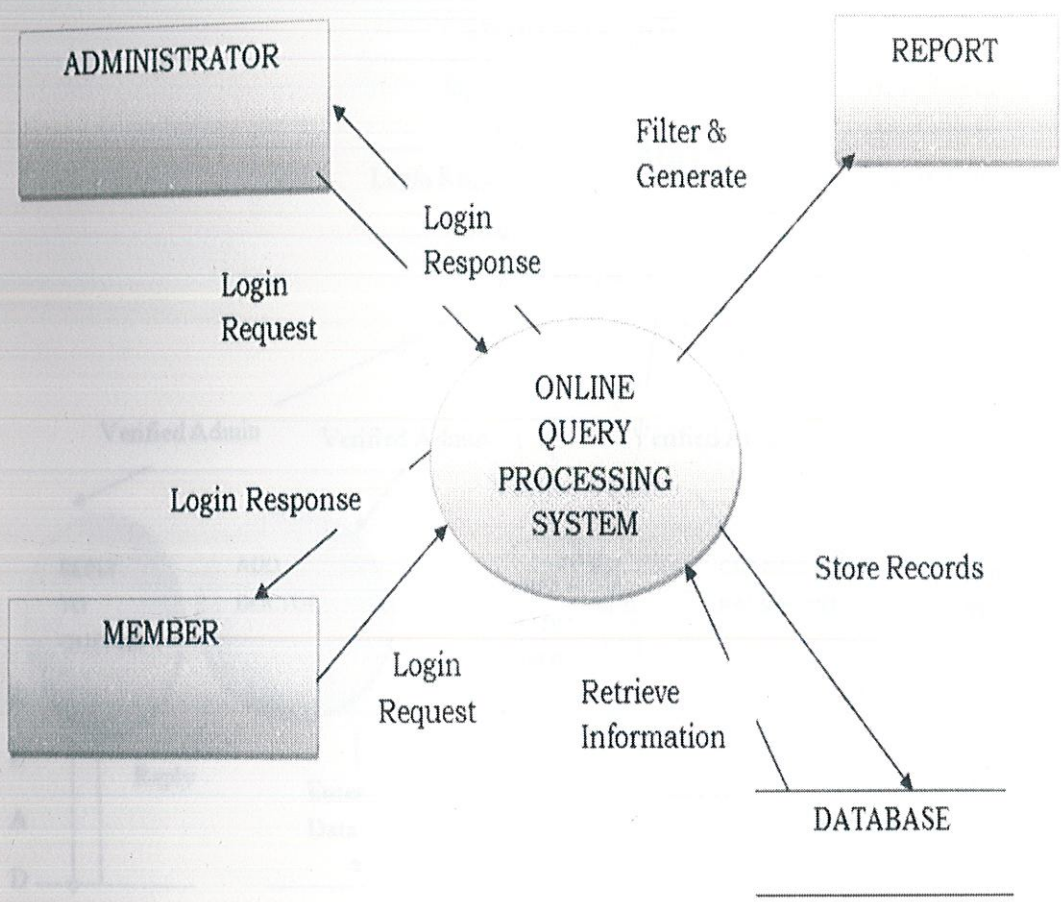


Fig 5.3 : 0-level DFD

LEVEL '1' DFD

ADMINISTRATOR DFD

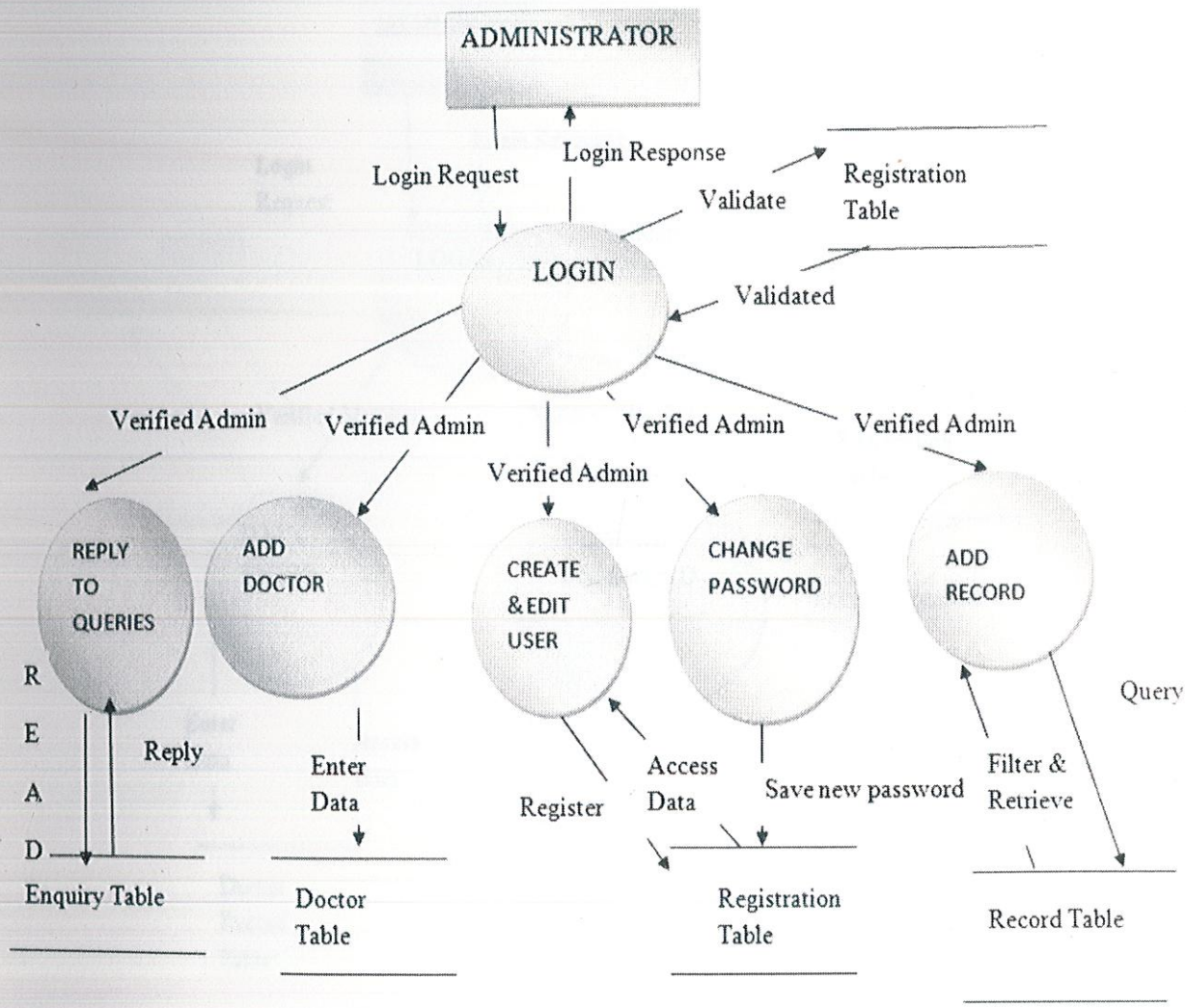


Fig 5.4: DFD of administrator

MEMBER DFD

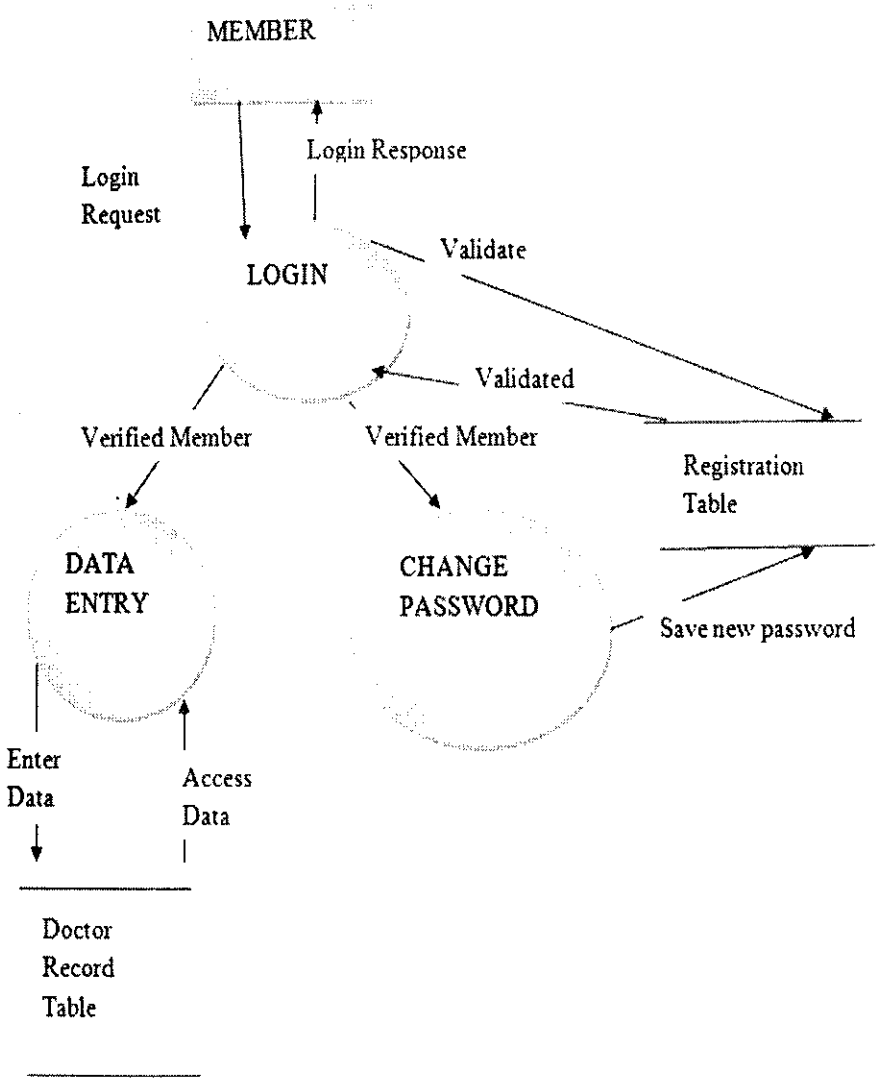


Fig 5.5 : DFD of member

REGISTER MEMBER DFD

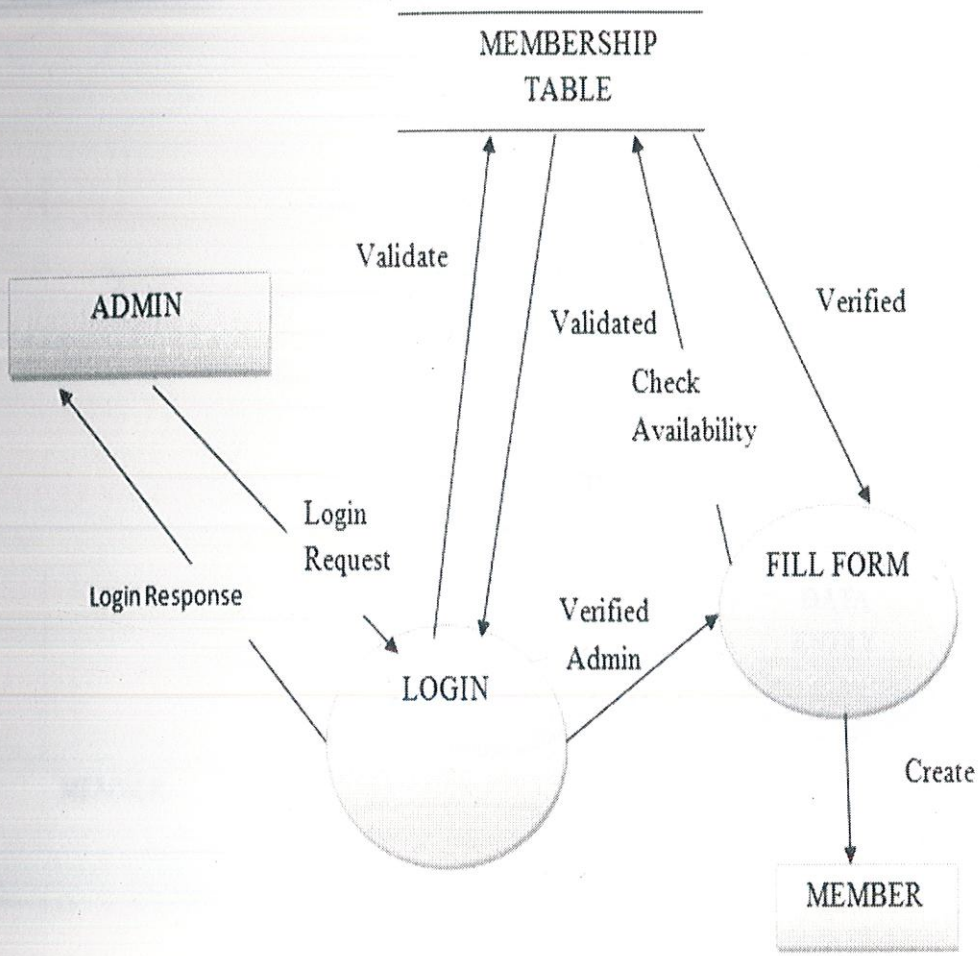


Fig 5.6: DFD of registered member

DATA ENTRY DFD:

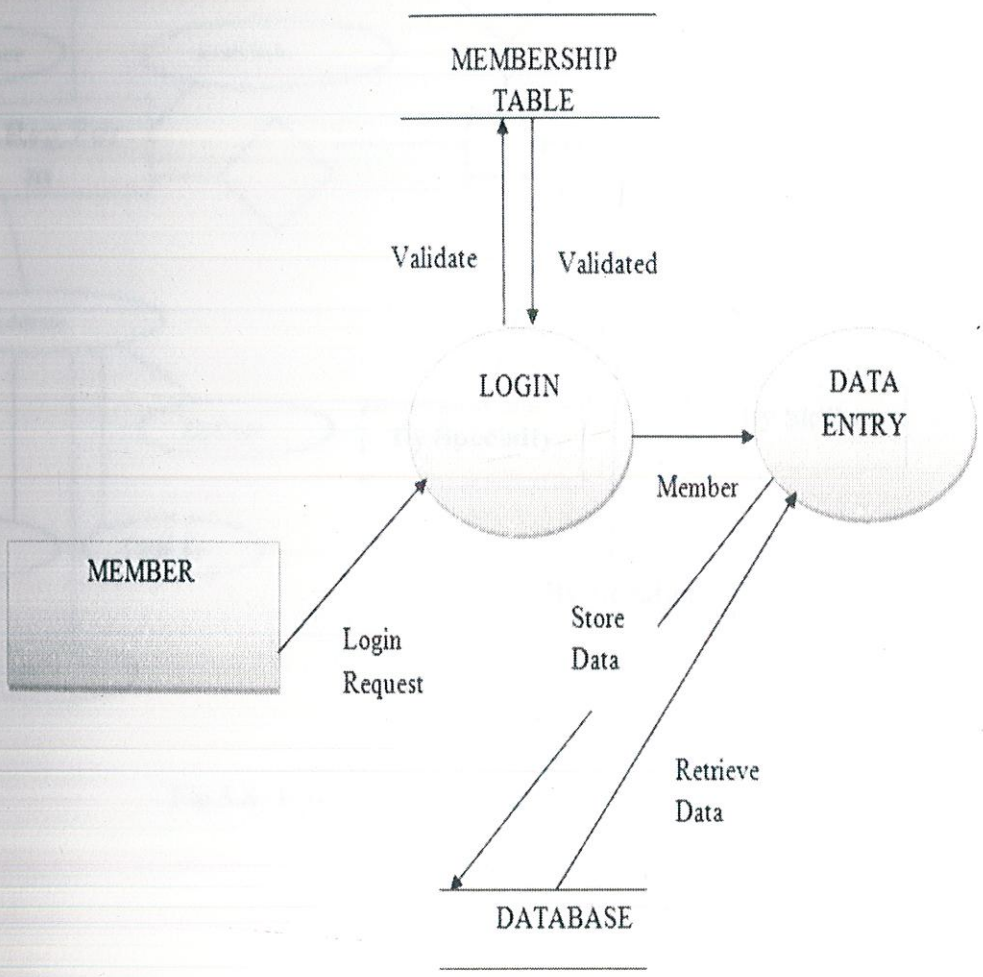


Fig 5.7: DFD of data entery

5.7 E-R Diagrams

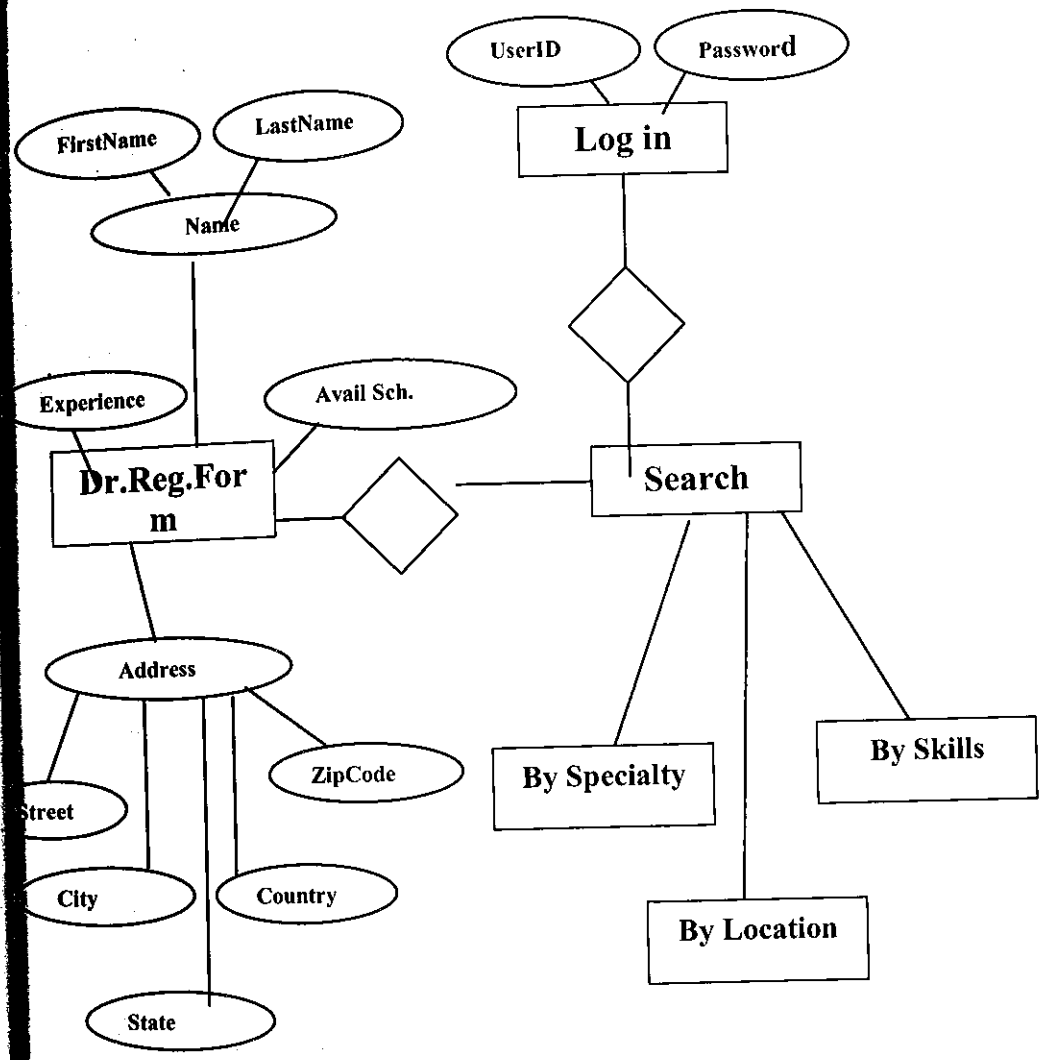


Fig 5.8: E-R Digram of DOQPS

5.8 FLOW CHART

INFORMATION FLOW DIAGRAM

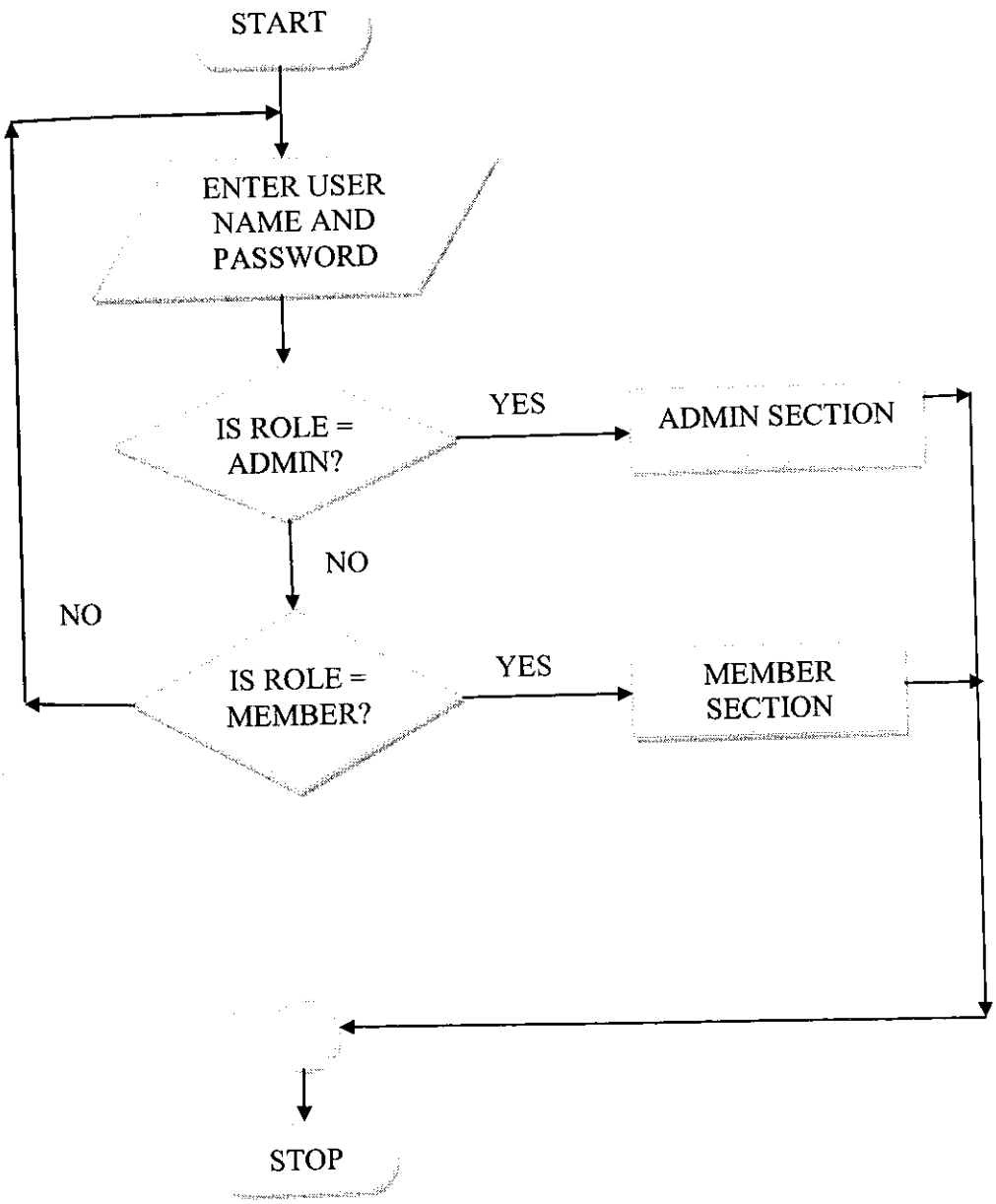
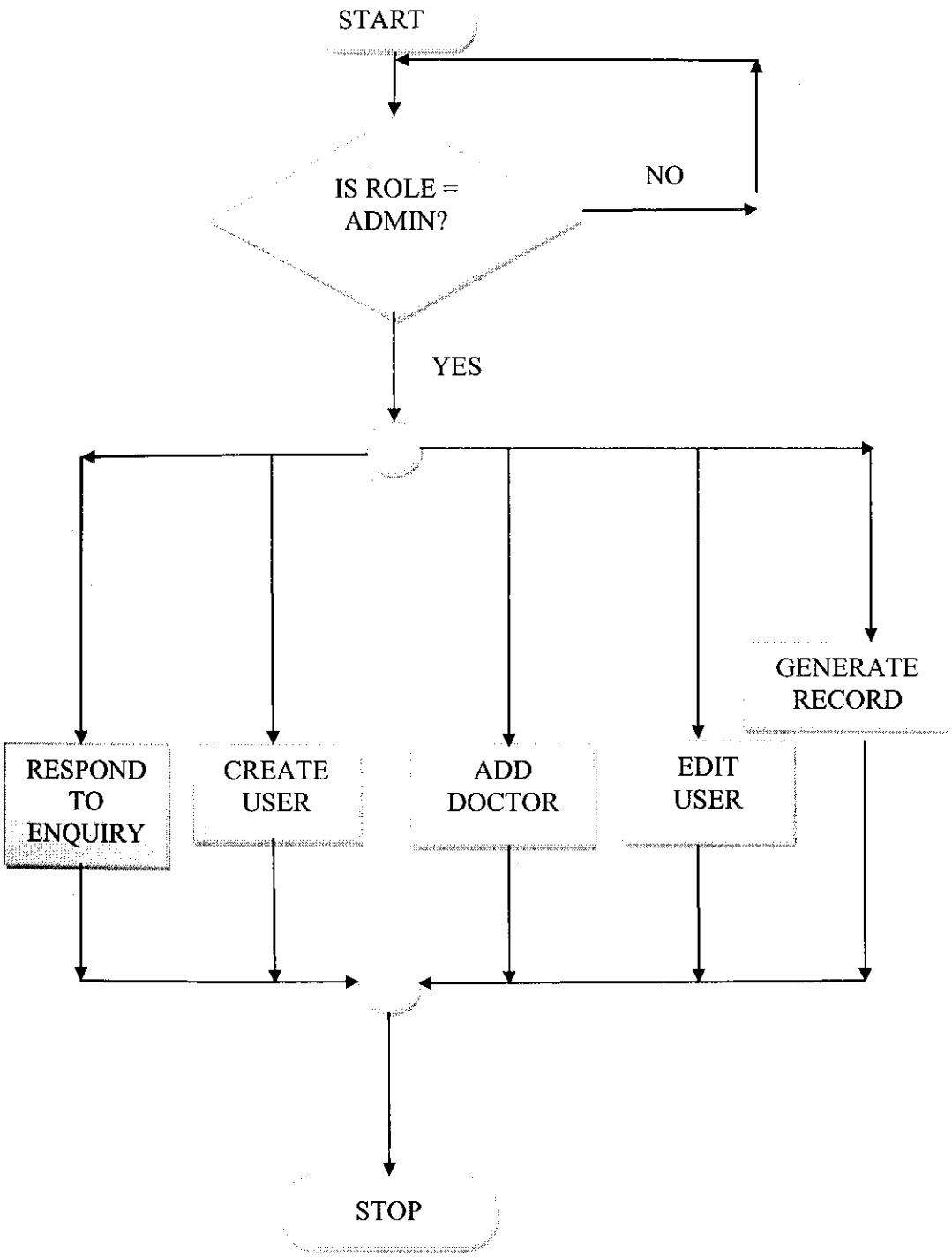


Fig 5.9 Main flow chart

ADMIN SECTION



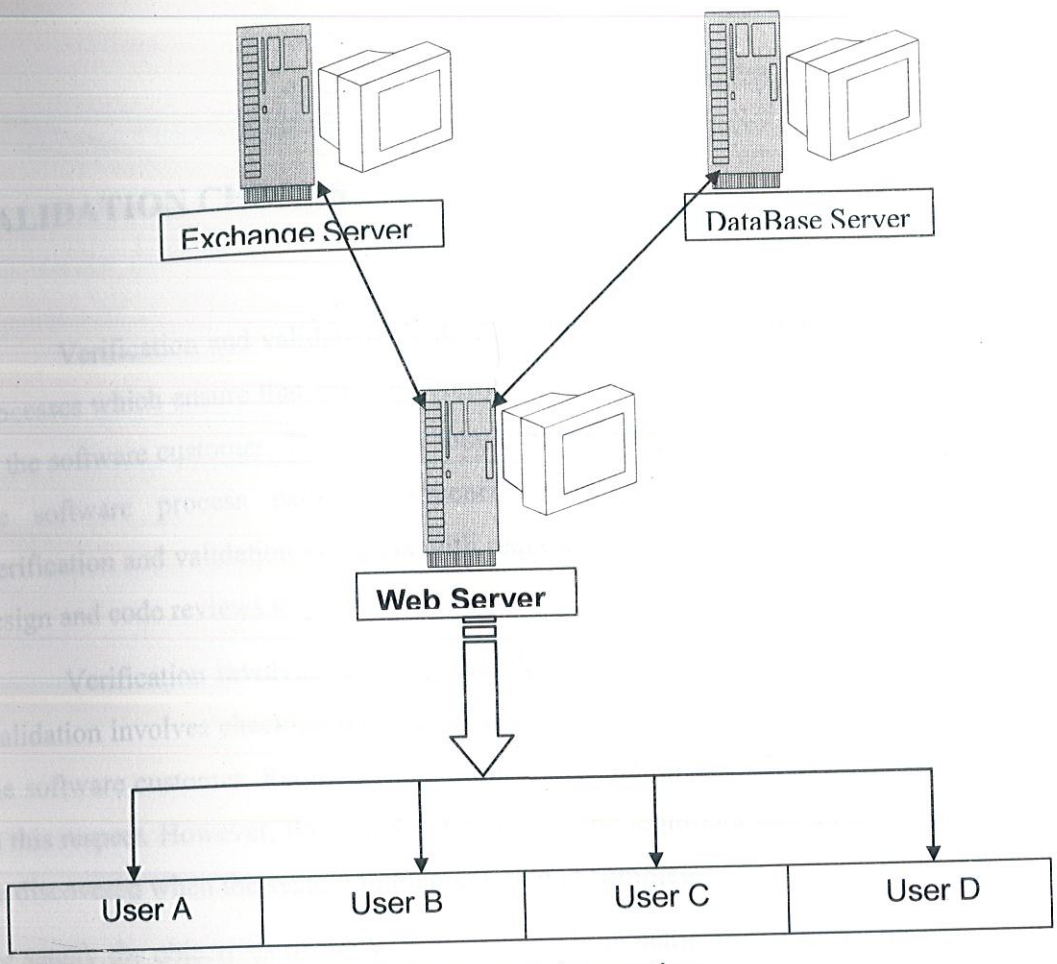


Fig 5.12 : System interaction

PROCESS LOGIC

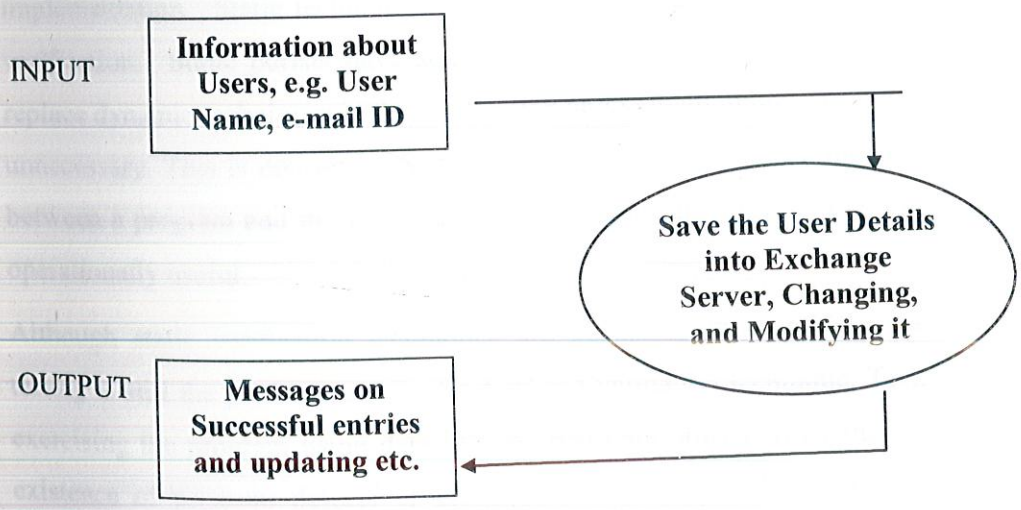


Fig 5.13:Digram of process logic

VALIDATION CHECKS

Verification and validation (V & V) is the generic name given to the checking processes which ensure that software conforms to its specification and meets the need of the software customer. The system should be verified and validated at each stage of the software process using documents produced during the previous stage. Verification and validation i.e. starts with requirements reviews and continues through design and code reviews to product testing.

Verification involves checking that the program conforms to its specification. Validation involves checking that the program implemented meets the expectations of the software customer. Requirements validation techniques, such as prototyping, help in this respect. However, flaws and deficiency in the requirements can sometimes only be discovered when the system implementation is complete.

To satisfy the objectives of the V & V process, both static and dynamic techniques of system checking and analysis should be used. Static techniques are concerned with the analysis and checking of system representations such as the requirements document. Design diagram and the program source code. They may be applied at all stages of the process through structured reviews. Dynamic techniques or test involve exercising and implementation. Static techniques include program inspections, analysis and formal verification. Some purists have suggested that these techniques should completely replace dynamic techniques in the verification and validation process and that testing is unnecessary. This is nonsense. Static techniques can only check the correspondence between a program and its specification; they cannot demonstrate that the software is operationally useful.

Although static verification techniques are becoming more widely used, program testing is still the predominant verification and validation technique. Testing involves exercising the program using data like the real data processed by the program. The existence of program defects or inadequacies is inferred from unexpected system output.

5.9 System Functionality

This section presents the functions of the Web Online Query processing System , together with an explanation on how the find doctors user and doctor use this find doctor .

In designing a system, the functional requirements reflect a set of inputs, the actions and the outputs of the system. They define the reactions of the designed system in different situation. The developed find doctor which also acts as a find doctor also to help fresh graduates and final year find doctor seekers to search for users has three sections- Admin, non-registered users and registered users. The functional requirements of the proposed web find doctor are explained below:

Admin section

This involves all the management functions in order to control and manage those aspects of the web find doctor . Such as doctor and find doctor management, news, about us, settings, files upload, contact information, security, received message settings, users and advertising.

The General Users

General users are the students and users who are already who use the standard services in the find doctor . They can view information about the find doctor categories, find doctor advertisements, requirements, qualification.

The Non-registered Users

When a user wants to be a member of the find doctor as a userseeker, he or she should register with the find doctor as a new userseeker .This sub-module works in the same way for the companies or organizations, which might wish to upload information and their find doctor vacancies for the find doctor. They register in the system as a new employer.

☐ The Registered Users

When users register themselves as a userseeker or an employer, they can access their personal data and the control panel. The registered users can login and log out from the

find doctor and can access the relevant information based on their membership type. When a user signs in as a user and access the control panel, he can have access to special facilities which the general users are deprived:

- Security management: Users can change their old password and replace it with a new one.

- Profile management: The registered users are able to view their profile information and edit it, if necessary.

- Resume management: The find doctor create their resume to indicate their skills, abilities, and qualifications to the doctor , they can read and edit the resume and upload it as a Microsoft Word or text file.

- Upload Details: The find doctor can upload their CV as a text or document file to show it to the potential doctor .

- Search User: The search option is one of the important features available on the find doctor control panel. From the control panel, the users can search for find doctor which have been uploaded into the find doctor perform advanced search by entering more detailed information for search such as the qualifications (degree), find doctor type, skills, level of experience, part-time or full- time find doctor preference.

- Apply user: When the users find a suitable find doctor vacancy, they can apply for it online, and this application will be sent to the company that advertised the vacancy.

- Log out: Users can exit from the find doctor by using their control panel log out page and return to the main page of the find doctor . Doctor who are registered with the find doctor also have their control panel. The security, management profile, search and log out functions are similar to that for the find doctor, Function except for the following differences:

- Find doctor offer management: The registered doctor can create the find doctor advertisements for the find doctor here and also can edit them.

- Review applications: Doctor can view the applications sent online by the find doctor and appoint them to a specific find doctor position.

5.9.1 The Non-functional Requirements

The non-functional requirements include the services constraints and the quality of the system. Quality can be defined as the characteristics of the system which affect the level of users' satisfaction with the system such as quality attributes, quality goals and quality of service requirements. Constraints are restrictions to the services or functions of the system such as the choice limitation during the development process. The non-functional requirements for current web find doctor are as follows:

Availability: The amount of time available for some activities such as database upgrade and backup.

Reliability: It focuses on maintaining the performance of the system. An unreliable system is more prone to failure. It is important to choose the correct parameters to evaluate a web find doctor to ensure its reliability.

Robustness: A robust system is able to solve errors and tolerate invalid data, software defects, and unexpected operating conditions. The find doctor should be robust enough to face expected and unexpected failures.

5.10 SYSTEM ANALYSIS

System Analysis refers to the process of examining a situation with the intent of improving it through better procedures and methods. System design is the process of planning a new system to either replace or complement an existing system. But before any planning is done, the old system must be thoroughly understood and the requirements determined. System Analysis is therefore, the process of gathering and interpreting facts, diagnosis problems and using the information to re-comment improvements in the system. Or in other words, System Analysis means a detailed

explanation or description. Before computerizing a system under consideration, it has to be analyzed. We need to study how it functions currently, what are the problems, and what are the requirements that the proposed system should meet.

The main components of making software are:

- System and software requirements analysis
- Design and implementation of software
- Ensuring, verifying and maintaining software integrity

System analysis is an activity that encompasses most of the tasks that are collectively called Computer System Engineering. Confusion sometimes occurs because the term is often used in context that all dues It only to Software requirement analysis activities, but system analysis focuses on all the system elements- not just software.

System analysis is conducted with the following objectives in mind:

- Identify the customer's need
- Evaluate the system concept for feasibility
- Perform economic and technical analysis
- Allocate functions to hardware, software, database and other system elements
- Establish cost and schedule constraints
- Create a system definition that forms the foundation for all the subsequent engineering work.

System Analysis is consisting of two main works i.e. Identify the need and Preliminary Investigation.

5.11 DATABASE DESIGN

Database design is one of the main tasks in the design of an information system. Database design involves the logical design of the database which is used to store data and manage the data for retrieval. Data integrity, data availability, efficient updating and retrieval are the main factors which should be considered in designing the database.

SQL Server 2008 is used as the database management system (DBMS) for the proposed web find doctor . SQL Server 2008 has been one of the exciting releases of SQL Server for many years because of new and applicable features. SQL Server 2008 has the features which are big improvements from the earlier SQL Server 2000. Microsoft SQL Server 2008 has been revised to meet the new data management requirements and to improve performance programmability.

Each component in SQL Server 2008 has a unique architecture and life cycle. SQL Server 2008 was selected for the database infrastructure because of the following features:

Easy to Install and Manage: SQL Server 2008 can be downloaded very fast and setting up the user interface (UI) is simple.Rich Database functionality: SQL Server 2008 has the ability to store procedures, Extended indexes, support SQL transactions and good report the generation facility.

Deep Integration with Visual Studio 2010.Robust Security: Security is one of the prominent features of SQL Server 2008. It Implements security by using default settings and supports active directory.

1. USER TABLE

COLUMN NAME	DATA TYPE	SIZE	KEY
Fn	Varchar	50	
Ln	Varchar	50	
Un	Varchar	50	
Pass	Varchar	50	
Gen	Char	1	
Email	Varchar	50	

Table 5.1: User Table

2. DOCTOR SPECIALTY TABLE

COLUMN NAME	DATA TYPE	SIZE	KEY
Docspccod	Int		
Docspecdoccod	Int		
Docspcsubspccod	Int		

Table 5.2: Doctor speciality Table

3. SUB-SPECIALTY CODE

COLUMN NAME	DATA TYPE	SIZE	KEY
Docspccod	Int		
Docspcnam	Varchar	50	
Docspcspccod	Int		

Table 5.3: Sub-speciality Table

4. DOCTOR TABLE

COLUMN NAME	DATA TYPE	SIZE	KEY
Doccod	Int		Primary
docfstnam	Varchar	50	
doclstnam	Varchar	50	
Docgnd	Varchar	50	
Docqal	Varchar	200	
Docprf	Varchar	500	
docavlsch	Varchar	500	
docstreet	Varchar	50	
Doccity	Varchar	50	
docstate	Varchar	50	
doccountry	Varchar	50	
doczipcode	Varchar	100	
Ddfrom	Int		
Ddto	Int		
dochiglig	Varchar	500	
Docexp	Int		
docspec	Varchar	50	
docsubspec	Varchar	50	
Docpic	Varchar	50	

Table 5.4: Doctor Details Table

CHAPTER 6 - ALGORITHMS.

This chapter of the project report includes the implementation and development of the part of project that has been completed till May, 2013.

6.1 Advanced Pattern Matching.(String Searching)

- The previous slide is not a great example of what is meant by "String Searching." Nor is it meant to ridicule people without eyes....

- The object of string searching is to find the location of a specific text pattern within a larger body of text (e.g., a sentence, a paragraph, a book, etc.).

- As with most algorithms, the main considerations for string searching are speed and efficiency.

  There are a number of string searching algorithms in existence today, but the one we shall review is Brute Force.

6.1.1 Brute force algorithm:

- The Brute Force algorithm compares the pattern to the text, one character at a time, until unmatching characters are found:

  *TW*O ROADS DIVERGED IN A YELLOW WOOD *R*OADS
  T*W*O ROADS DIVERGED IN A YELLOW WOOD *R*OADS
  TW*O* ROADS DIVERGED IN A YELLOW WOOD *R*OADS
  TWO ROADS DIVERGED IN A YELLOW WOOD *R*OADS
  TWO ***ROADS*** DIVERGED IN A YELLOW WOOD
  ROADS

1. Compared characters are italicized.

2. Correct matches are in boldface type.

The algorithm can be designed to stop on either the first occurrence of the pattern, or upon reaching the end of the text.

BRUTE FORCE PSEUDO CODE

Here's the pseudo-code

do
if (text letter == pattern letter) compare next letter of pattern to next letter of text
else
move pattern down text by one letter **while** (entire pattern found or end of text)

```
tetththeheehthtehthethththehehtht
the
tetththeheehthtehthethththehehtht
  the
tetththeheehthtehthethththehehtht
    the
tetththeheehthtehthethththehehtht
      the
tetththeheehthtehthethththehehtht
        the
tetththeheehthtehthethththehehtht
          the
```

3. Given a pattern M characters in length, and a text N characters in length...

4. **Worst case**: compares pattern to each substring of text of length M. For example, M=5.

5. *AAAAA*AAAAAAAAAAAAAAAAAAAAH

AAAAH **5 comparisons made**

4. *AAAAA*AAAAAAAAAAAAAAAAAAAAH

AAAAH **5 comparisons made**

5. *AAAAA*AAAAAAAAAAAAAAAAAAAAH

AAAAH **5 comparisons made**

6. *AAAAA*AAAAAAAAAAAAAAAAAAAAH

AAAAH **5 comparisons made**

7. *AAAAA*AAAAAAAAAAAAAAAAAAAAH

AAAAH **5 comparisons made**

....

N) AAAAAAAAAAAAAAAAAAAAAAAA*AAAAH*

5 comparisons made *AAAAH*

- Total number of comparisons: M (N-M+1)
- Worst case time complexity: O(MN)

6.2 ADVANCE SEARCH (A Fast Multiple String-Pattern Matching Algorithm)

Text Partitioning Algorithm

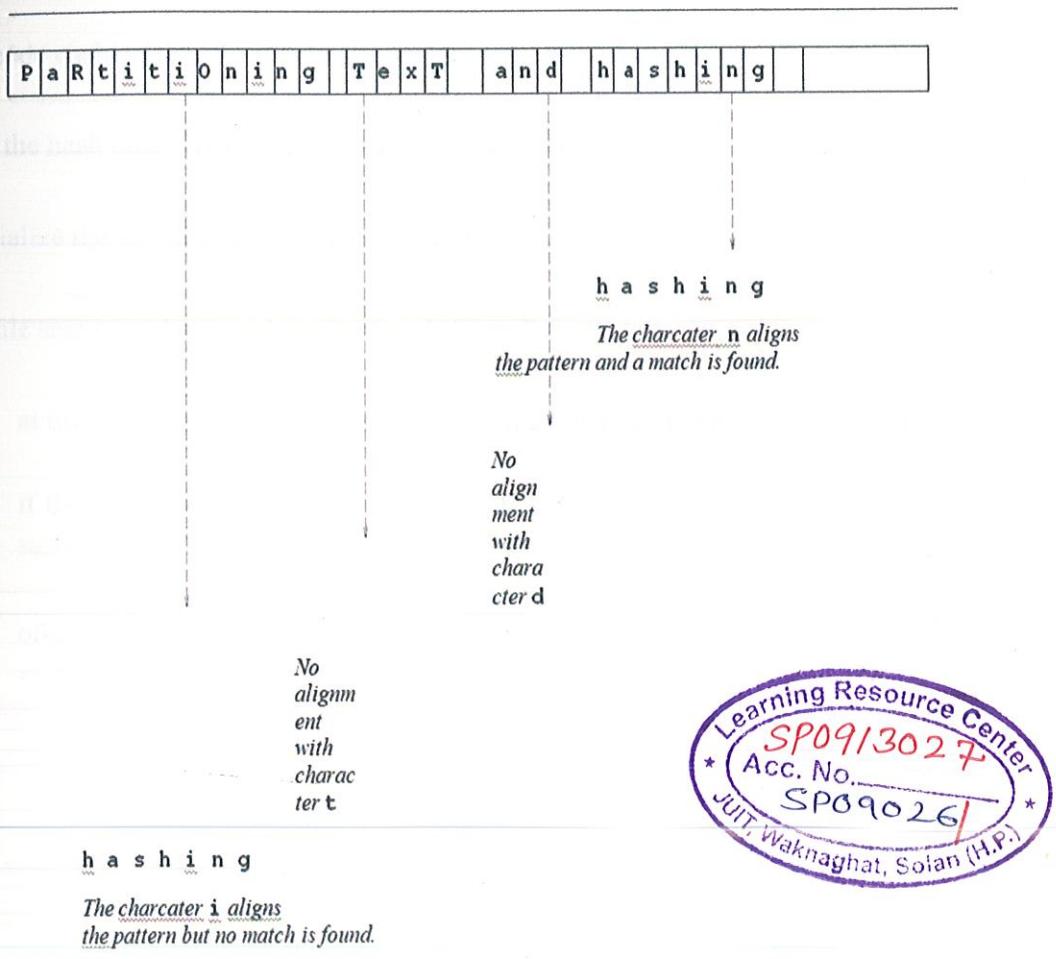


Fig 6.1 :Text partioning algorithm

6.2.1 FINDING MULTIPLE PATTERNS

A separate pattern mask, $PMASK_i$, is needed for each pattern P_i. Then, perform the pattern testing procedure for each pattern at every text position. Expensive for many input patterns

To avoid unnecessary pattern occurrence testings, we use a hashing scheme:

- Prepare a hash table of size H bits, and
- a hash mask, HMASK whose lower H bits are 1's and others are 0's.

MULTIPLE STRING – Pattern Matching Algorithm

- Scan the input patterns to determine the number of bits, E, for each character encoding and de ne the encoding function Encode
- Encode each pattern P_i and set the associated mask $PMASK_i$.
- Set the hash mask HMASK according to the hash table size.
- Initialize the text scanning variable T to 0.
- While scanning the text character by shifting T E bits left,
  1. at the hash entry position computed by logically ANDing T and HMASK,
  2. if the hash entry at the position is empty, skip the pattern testing procedure and scan the next text character.
  3. otherwise, perform the pattern testing procedure for all patterns at the hash entry.

STRING MATCHING ALGORITHM

```
struct hash_entry {PAT p; PATMASK pmask; struct hash_entry * next}; struct
hash_entry HTBL[HASH_SZ]; /* HASH_SZ = exp(2,H) */
PAT                     HMASK;          /* a mask for hashing */

for (i=1; i<=n; i++) insert_pattern_into_hash_table(P[i]);


T = encode_ncharacters(text,S);

i = S+1;
while (i <= Tlen) {
        if (HTBL[T&HMASK] != NULL) {
                candidate =
                HTBL[T&HMASK];
                while(candidate) {
                        if (!(candidate->p ^ (T & candidate->pmask)))
                                report_pattern_match(candidate);
                        candidate = candidate->next;
                }
        }
        T = T<<E |
        ENCODE(text[i]); i ++;
}
```

CONCLUSION

In developing Find doctor, care was taken to incorporate all the good features considered in the earlier chapters, and the users' requirements identified from the survey. The system, however, can be further enhanced by adding new features

.

It should be noted; however, that the Find Doctor has addressed the following issues:

- Achieved the Project objectives.
- It provides online recruitment through the knowledge management system for the users.
- Provides enough information to helps to choose the right information for their diseases and find good doctors.
- Promotes a new relationship between the users and the doctors.

Although the Project objectives have been achieved, the find doctor can be further improved in future, by adding new features and services such as chat facility or reporting facility. Most of the users claimed that the new doctor should be user-friendly, should have high performance.

It provides information for users who need help about selected problem either minor or major, and for senior users who are interested to know about the details of different doctors can easily search for it. In this way, the find doctor has established link between the user and doctors.

BIBLIOGRAPHY

1. Unleashed ASP.NET Copyrights © 2004 By SAMS' publication, Techmedia, New Delhi.
2. Introduction To ASP.NET By A. Russell Jones copyright © 2000 SyBEX Inc. USA, Bible Publications New Delhi.
3. Greg Perry, ASP.Net in 21 Days, © Copyright 1998 by sams publications, Techmedia, New Delhi - 110002.
4. Professional ASP.NET, by Richard Anderson ©copyright 2004 wrox press, USA.
5. Introduction To EXCHANGE SERVER 2000, Copyrights © 1999 By BPB' publication, Techmedia.
6. Mastering SQL Server 2000 © Copyright Tata InfoTech Ltd. New Delhi- 110048
7. For POP3 Protocol –S/W 30 Days Trial (aspnetPOP3) provided By WWW.aspnetpop3.com (website).
8. For IMAP4 Protocol –S/W 30 Days Trial (aspnetIMAP4) provided by By WWW.aspnetpop3.com (website)
9. For EMAIL (SMTP) Protocol –S/W 30 Days Trial (aspnetEMAIL) Provided By WWW.aspnetpop3.com (website)
10. Protocol used to interact with Active Directory Provided by Adnan Syad Website is WWW.codeproject.com
11. Protocol used to interact with Active Directory Provided by Dll on Ex Ser. Help Site Is WWW.Microsoft.com

Others:-

1. Alavi, M. & Leidner, D. 1999. Knowledge management systems: issues, challenges, And benifits.

2. Ang, Z., Cai, S., Zhou, Z. & Zhou, N. 2005. Development and validation of an Instrument to measure user perceived service quality of information presenting web Find Doctors.

3. Benbya, H., Belbaly, N., Passiante, G. & GallipololiI, V. 2004. CorporateFind Doctor: tool for knowledge management synchronization. International Journal of Information Management, 24,

4. Bsiri, S., Geierhos, M. & Ringlstetter, C. 2008. Structuring User search via local grammars. Advancesin Natural Language Processing and Applications, 201.

5. CORTES, U., SANCHEZ-MARRE, M. & SANGUEESA, R. 2001. Knowledge management inenvironmental decision support systems. Ai Communications, 14, 3-12.

6. Doyle, A. 2008. Internet Your Way to a New User: How to Really Find a User Online, Happy about.

7. Maddison, R. N., Baker, G. J., Bhabuta, L., Fitzgerald, G., Hindle, K., Song, J. H. T., Stokes, N. and

8. Wood, J. R. G. (1983) Information Systems methodologies, Wiley Heyden, on behalf of british.

9. Mochol, M., Wache, H. & Nixon, L. 2007. Improving the accuracy of User search with semantic techniques. In, 2007. Springer, 301-313.

10. Robins, D., Sochats, K. 2000. Web Portals: History and Direction [online].

11. Straub, D., Boudreau, M. & Gefen, D. 2004. Validation guidelines for IS positivist research.

12. Communications of the Association for Information Systems, 13, 380-427.

13. Sulaiman, N. & Burke, M. 2009. A case analysis of knowledge sharing implementation and User

14. Searching in Malaysia. International Journal of Information Management.

15. Telang, R. & Mukhopadhyay, T.2005. Drivers of Web portal use. Electronic Commerce research and applications, 4, 49-65.

16. Vidgen, R. 2002. Constructing a web information system development methodology. Information Systems Journal, 12, 247-261.

17. Weber, A. & Mahringer, H. 2008. Choice and success of User search methods. Empirical Economics, 35, 153-178.

18. Wege, C., 2002. Portal Server Technology. IEEE Internet Computing [online], 6 (3), 73-77.

19. Whitten, J., Bentley, L. & Dittman, K. 2004. Fundamentals of Systems Analysis and Design Methods. The McGraw-Hill companies, Inc.

WEB SITES

1. www.msdn.microsoft.com

2. www.databasejournal.com

3. www.wikipedia.com

4. www.C#Forum.com

5. en.csharp-online.net

APPENDIX A: SOURCE CODE

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

namespace nsvitals
{
    //Connection Class\\
    public abstract class clscon
    {
        protected SqlConnection con = new SqlConnection();
        public clscon()
        {
            con.ConnectionString =
ConfigurationManager.ConnectionStrings["cn"].ConnectionString;
            {
                con.Open();
            }
        }
    }
    /// <summary>
    /// //interfaces\\
    /// </summary>
    public interface intspc
{
    Int32 p_spccod
    {
        get;
        set;
    }
    string p_spcdsc
    {
        get;
        set;
    }

      }
    public interface intsubspc
    {
        Int32 p_subspccod
        {
            get;
            set;
        }
        string p_subspcnam
        {
            get;
            set;
        }
        Int32 p_subspcspccod
        {
            get;
            set;
        }
```

```
}
    public interface intdoc
    {
        Int32 p_doccod
        {
            get;
            set;
        }
        string p_docfstnam
        {
            get;
            set;
        }
        string p_doclstnam
        {
            get;
            set;
        }
        string p_docqal
        {
            get;
            set;
        }
        string p_docprf
        {
            get;
            set;
        }
        string p_docavlsch
        {
            get;
            set;
        }
        string p_doczipcod
        {
            get;
            set;
        }
        string p_docpic
        {
            get;
            set;
        }
       char p_docgnd
       {
           get;
           set;
       }
        string p_dochighlig
        {
            get;
            set;
        }
    }
    public interface intdocspc
    {
        Int32 p_docspccod
        {
            get;
            set;
        }
        Int32 p_docspcdoccod
```

```
        {
            get;
            set;
        }
        Int32 p_docspcsubspccod
        {
            get;
            set;
        }
    }
    public interface intrat
    {
        Int32 p_ratcod
        {
            get;
            set;
        }
        Int32 p_ratdoccod
        {
            get;
            set;
        }
        DateTime p_ratdate
        {
            get;
            set;
        }
        float p_ratscr
        {
            get;
            set;
        }
        string p_ratcom
        {
            get;
            set;
        }
    }
    /// <summary>
    /// //Property Classes\\
    /// </summary>
    public class clsspcprp:intspc
    {
        private Int32 spccod;
        private string spcdsc;
        public int  p_spccod
    {
        get
      {
            return spccod;
      }
        set
      {
            spccod = value;
      }
}

public string  p_spcdsc
{
        get
      {
            return spcdsc;
```

```
        }
          set
        {
             spcdsc = value;
        }
}
}
    public class clssubspcprp:intsubspc
    {
        private Int32 subspccod,subspcspccod;
        private string subspcnam;
public int  p_subspccod
{
          get
        {
             return subspccod;
        }
          set
        {
             subspccod = value;
        }
}

public string  p_subspcnam
{
          get
        {
              return subspcnam;
    }
          set
        {
             subspcnam = value;
        }
}

public int  p_subspcspccod
{
          get
        {
             return subspcspccod;
        }
          set
        {
             subspcspccod = value;
        }
}
}
    public class clsdocprp:intdoc
    {
        private Int32 doccod;
        private string
docfstnam,doclstnam,docgal,docprf,docavlsch,doczipcod,docpic,dochighlig;
        private char docgnd;
public int  p_doccod
{
          get
        {
             return doccod;
        }
          set
        {
             doccod = value;
```

```
        }
    }

    public string  p_docfstnam
    {
            get
        {
                return docfstnam;
        }
            set
        {
                docfstnam = value;
        }
    }

    public string  p_doclstnam
    {
            get
        {
                return doclstnam;
        }
            set
        {
                doclstnam = value;
        }
    }

    public string  p_docqal
    {
            get
        {
                return docqal;
        }
            set
        {
                docqal = value;
        }
    }

    public string  p_docprf
    {
            get
        {
                return docprf;
        }
            set
        {
                docprf = value;
        }
    }

    public string  p_docavlsch
    {
            get
        {
                return docavlsch;
        }
            set
        {
                docavlsch = value;
        }
    }
```

```
public string  p_doczipcod
{
        get
      {
            return doczipcod;
      }
        set
      {
            doczipcod = value;
      }
}

public string  p_docpic
{
        get
      {
            return docpic;
      }
        set
      {
            docpic = value;

      }
}

public char  p_docgnd
{
        get
      {
            return docgnd;
      }
        set
      {
            docgnd = value;
      }
}

public string  p_dochighlig
{
        get
      {
            return dochighlig;
      }
        set
      {
            dochighlig = value;
      }
}
}
    public class clsdocspcprp:intdocspc
    {
        private Int32 docspccod,docspcdoccod,docspcsubspccod;

public int  p_docspccod
{
        get
      {
            return docspccod;
      }
        set
      {
```

```
            docspccod = value;
        }
}

public int  p_docspcdoccod
{
        get
        {
            return docspcdoccod;
        }
        set
        {
            docspcdoccod = value;
        }
}

public int  p_docspcsubspccod
{
        get
        {
            return docspcsubspccod;
        }
        set
        {
            docspcsubspccod = value;
        }
}
}
    public class clsratprp:intrat
    {
        private Int32 ratcod,ratdoccod;
        private string ratcom;
        private DateTime ratdate;
        private float ratscr;
public int  p_ratcod
{
        get
        {
            return ratcod;
        }
        set
        {
            ratcod = value;
        }
}

public int  p_ratdoccod
{
        get
        {
            return ratdoccod;
        }
        set
        {
            ratdoccod = value;
        }
}

public DateTime  p_ratdate
{
        get
        {
```

```
            return ratdate;
        }
          set
        {
              ratdate = value;
        }
}

public float  p_ratscr
{
          get
        {
              return ratscr;
        }
          set
        {
              ratscr = value;
        }
}

public string  p_ratcom
{
          get
        {
              return ratcom;

        }
          set
        {
              ratcom = value;

        }
}
}
    /// <summary>
    /// //Main Classes\\
    /// </summary>
     public class clsspc:clscon
             {
                 public void Save_Rec(clsspcprp p)
                 {
                     if (con.State == ConnectionState.Closed)
                     {
                         con.Open();
                     }
                     SqlCommand cmd = new SqlCommand("insspc",con);
                     cmd.CommandType = CommandType.StoredProcedure;
                     //cmd.Parameters.Add("@spccod",SqlDbType.Int).Value =
p.p_spccod;

cmd.Parameters.Add("@spcdsc",SqlDbType.VarChar,50).Value = p.p_spcdsc;
                     cmd.ExecuteNonQuery();
                     cmd.Dispose();
                     con.Close();
                 }


                 public void Update_Rec(clsspcprp p)
                 {
                     if (con.State == ConnectionState.Closed)
                     {
                         con.Open();
```

```
                    }
                    SqlCommand cmd = new SqlCommand("upspc",con);
                    cmd.CommandType = CommandType.StoredProcedure;
                    cmd.Parameters.Add("@spccod",SqlDbType.Int).Value =
p.p_spccod;

cmd.Parameters.Add("@spcdsc",SqlDbType.VarChar,50).Value = p.p_spcdsc;
                    cmd.ExecuteNonQuery();
                    cmd.Dispose();
                    con.Close();
                }

                 public void Delete_Rec(clsspcprp p)
                {
                    if (con.State == ConnectionState.Closed)
                    {
                        con.Open();
                    }
                     SqlCommand cmd = new SqlCommand("delspc",con);
                     cmd.CommandType = CommandType.StoredProcedure;
                     cmd.Parameters.Add("@spccod",SqlDbType.Int).Value =
p.p_spccod;
                     cmd.ExecuteNonQuery();
                     cmd.Dispose();
                     con.Close();
                }
          public List<clsspcprp>Disp_Rec()
          {

          if (con.State == ConnectionState.Closed)
                    {
                        con.Open();
                    }
                     SqlCommand cmd = new SqlCommand("dispspc",con);
                     cmd.CommandType = CommandType.StoredProcedure;
              SqlDataReader dr = cmd.ExecuteReader();
              List<clsspcprp>obj = new List<clsspcprp>();
              while (dr.Read())
              {
                  clsspcprp k= new clsspcprp();
                  k.p_spccod = Convert.ToInt32(dr[0]);
                  k.p_spcdsc = dr[1].ToString();
                  obj.Add(k);
              }
              dr.Close();
              cmd.Dispose();
              con.Close();
              return obj;
          }
                 public List<clsspcprp>Find_Rec(Int32 spccod)
                 {
                     if (con.State == ConnectionState.Closed)
                    {
                        con.Open();
                    }
                     SqlCommand cmd = new SqlCommand("findspc",con);
                     cmd.CommandType = CommandType.StoredProcedure;
                     cmd.Parameters.Add("@spccod",SqlDbType.Int).Value
=spccod;
                    SqlDataReader dr = cmd.ExecuteReader();
                    List<clsspcprp>obj = new List<clsspcprp>();
                    if(dr.HasRows)
```

```
                    {
                        dr.Read();
                        clsspcprp k = new clsspcprp();
                        k.p_spccod = Convert.ToInt32(dr[0]);
                        k.p_spcdsc = dr[1].ToString();
                        obj.Add(k);
                    }
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    return obj;
            }
        }
    public class clssubspc : clscon
    {
        public void Save_Rec(clssubspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("inssubspc", con);
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.Add("@subspcnam", SqlDbType.VarChar, 50).Value
= p.p_subspcnam;
            cmd.Parameters.Add("@subspcspccod", SqlDbType.Int).Value =
p.p_subspcspccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }

        public void Update_Rec(clssubspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("upsubspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@subspccod", SqlDbType.Int).Value =
p.p_subspccod;
            cmd.Parameters.Add("@subspcnam", SqlDbType.VarChar, 50).Value
= p.p_subspcnam;
            cmd.Parameters.Add("@subspcspccod", SqlDbType.Int).Value =
p.p_subspcspccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }

        public void Delete_Rec(clssubspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("delsubspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@subspccod", SqlDbType.Int).Value =
p.p_subspccod;
```

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

        public List<clssubspcprp> Disp_Rec(Int32 spccod)
        {

            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("dispsubspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@spccod", SqlDbType.Int).Value =
                spccod;

            SqlDataReader dr = cmd.ExecuteReader();
            List<clssubspcprp> obj = new List<clssubspcprp>();
            while (dr.Read())
            {
                clssubspcprp k = new clssubspcprp();
                k.p_subspccod = Convert.ToInt32(dr[0]);
                k.p_subspcnam = dr[1].ToString();
                k.p_subspcspccod = Convert.ToInt32(dr[2]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
        public List<clssubspcprp> Find_Rec(Int32 subspccod)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("findsubspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@subspccod", SqlDbType.Int).Value =
subspccod;
            SqlDataReader dr = cmd.ExecuteReader();
            List<clssubspcprp> obj = new List<clssubspcprp>();
            if (dr.HasRows)
            {
                dr.Read();
                clssubspcprp k = new clssubspcprp();
                k.p_subspccod = Convert.ToInt32(dr[0]);
                k.p_subspcnam = dr[1].ToString();
                k.p_subspcspccod = Convert.ToInt32(dr[2]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
    }
    public class clsdoc : clscon
    {
        public DataSet srcdocbyspc(Int32 subspccod)
```

```
        {
            SqlDataAdapter adp = new SqlDataAdapter("srcdocbyspc", con);
            adp.SelectCommand.CommandType = CommandType.StoredProcedure;
            adp.SelectCommand.Parameters.Add("@subspccod",
SqlDbType.Int).Value = subspccod;
            DataSet ds = new DataSet();
            adp.Fill(ds);
            return ds;
        }
        public DataSet srcdocbyspcloc(Int32 subspccod, String zipcod)
        {
            SqlDataAdapter adp = new SqlDataAdapter("srcdocbyspcloc",
con);
            adp.SelectCommand.CommandType = CommandType.StoredProcedure;
            adp.SelectCommand.Parameters.Add("@subspccod",
SqlDbType.Int).Value = subspccod;
            adp.SelectCommand.Parameters.Add("@zipcod",
SqlDbType.VarChar, 50).Value = zipcod;
            DataSet ds = new DataSet();
            adp.Fill(ds);
            return ds;
        }


        public Int32 Save_Rec(clsdocprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("instbdoc", con);
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.Add("@docfstnam", SqlDbType.VarChar, 50).Value
= p.p_docfstnam;
            cmd.Parameters.Add("@doclstnam", SqlDbType.VarChar, 50).Value
= p.p_doclstnam;
            cmd.Parameters.Add("@docqal", SqlDbType.VarChar, 200).Value =
p.p_docqal;
            cmd.Parameters.Add("@docprf", SqlDbType.VarChar, 500).Value =
p.p_docprf;
            cmd.Parameters.Add("@docavlsch", SqlDbType.VarChar,
500).Value = p.p_docavlsch;
            cmd.Parameters.Add("@doczipcod", SqlDbType.VarChar,
100).Value = p.p_doczipcod;
            cmd.Parameters.Add("@docpic", SqlDbType.VarChar, 50).Value =
p.p_docpic;
            cmd.Parameters.Add("@docgnd", SqlDbType.Char,1).Value =
p.p_docgnd;
            cmd.Parameters.Add("@dochiglig", SqlDbType.VarChar,
500).Value = p.p_dochighlig;

            cmd.Parameters.Add("@r", SqlDbType.Int).Direction =
ParameterDirection.ReturnValue;
            cmd.ExecuteNonQuery();
            Int32 a = Convert.ToInt32(cmd.Parameters["@r"].Value);
            cmd.Dispose();
            con.Close();
            return a;
        }
```

```
        public void Update_Rec(clsdocprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("uptbdoc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@doccod", SqlDbType.Int).Value =
p.p_doccod;
            cmd.Parameters.Add("@docfstnam", SqlDbType.VarChar, 50).Value
= p.p_docfstnam;
            cmd.Parameters.Add("@doclstnam", SqlDbType.VarChar, 50).Value
= p.p_doclstnam;
            cmd.Parameters.Add("@docqal", SqlDbType.VarChar, 200).Value =
p.p_docqal;
            cmd.Parameters.Add("@docprf", SqlDbType.VarChar, 500).Value =
p.p_docprf;
            cmd.Parameters.Add("@docavlsch", SqlDbType.VarChar,
500).Value = p.p_docavlsch;
            cmd.Parameters.Add("@doczipcod", SqlDbType.VarChar,
100).Value = p.p_doczipcod;
            cmd.Parameters.Add("@docpic", SqlDbType.VarChar, 50).Value =
p.p_docpic;
            cmd.Parameters.Add("@docgnd", SqlDbType.Char, 1).Value =
p.p_docgnd;
            cmd.Parameters.Add("@dochiglig", SqlDbType.VarChar,
500).Value = p.p_dochighlig;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }

        public void Delete_Rec(clsdocprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("deltbdoc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@doccod", SqlDbType.Int).Value =
p.p_doccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public List<clsdocprp> Disp_Rec()
        {

            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("distbdoc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            SqlDataReader dr = cmd.ExecuteReader();
            List<clsdocprp> obj = new List<clsdocprp>();
            while (dr.Read())
            {
                clsdocprp k = new clsdocprp();
                k.p_doccod = Convert.ToInt32(dr[0]);
```

```
                k.p_docfstnam = dr[1].ToString();
                k.p_doclstnam = dr[2].ToString();
                k.p_docqal = dr[3].ToString();
                k.p_docprf = dr[4].ToString();
                k.p_docavlsch = dr[5].ToString();
                k.p_doczipcod = dr[6].ToString();
                k.p_docpic = dr[7].ToString();
                k.p_docgnd = Convert.ToChar(dr[8]);
                k.p_dochighlig = dr[9].ToString();
                //k.p_subspcspccod = Convert.ToInt32(dr[2]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
        public List<clsdocprp> Find_Rec(Int32 doccod)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("findtbdoc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@doccod", SqlDbType.Int).Value = doccod;
            SqlDataReader dr = cmd.ExecuteReader();
            List<clsdocprp> obj = new List<clsdocprp>();
            if (dr.HasRows)
            {
                dr.Read();
                clsdocprp k = new clsdocprp();
                k.p_doccod = Convert.ToInt32(dr[0]);
                k.p_docfstnam = dr[1].ToString();
                k.p_doclstnam = dr[2].ToString();
                k.p_docqal = dr[3].ToString();
                k.p_docprf = dr[4].ToString();
                k.p_docavlsch = dr[5].ToString();
                k.p_doczipcod = dr[6].ToString();
                k.p_docpic = dr[7].ToString();
                k.p_docgnd = Convert.ToChar(dr[8]);
                k.p_dochighlig = dr[9].ToString();
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
    }
    public class clsdocspc : clscon
    {
        public DataSet disdocspc(Int32 doccod)
        {
            SqlDataAdapter adp = new SqlDataAdapter("disdocspc", con);
            adp.SelectCommand.CommandType = CommandType.StoredProcedure;
            adp.SelectCommand.Parameters.Add("@doccod",
SqlDbType.Int).Value = doccod;
            DataSet ds = new DataSet();
            adp.Fill(ds);
            return ds;
        }
```

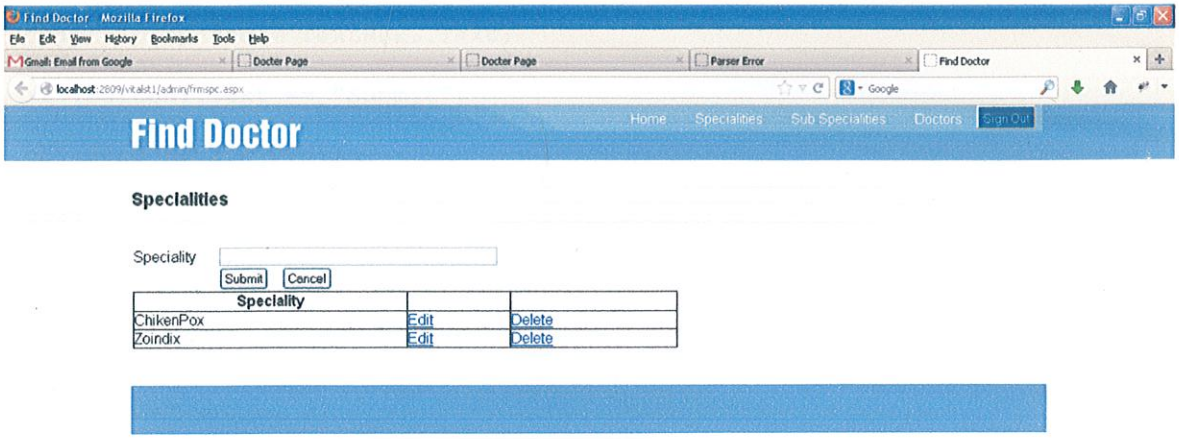
```
        public void Save_Rec(clsdocspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("insdocspc", con);
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.Add("@docspcdoccod", SqlDbType.Int).Value =
p.p_docspcdoccod;
            cmd.Parameters.Add("@docspcsubspccod", SqlDbType.Int).Value =
p.p_docspcsubspccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }


        public void Update_Rec(clsdocspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("updocspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@docspccod", SqlDbType.Int).Value =
p.p_docspccod;
            cmd.Parameters.Add("@docspcdoccod", SqlDbType.Int).Value =
p.p_docspcdoccod;
            cmd.Parameters.Add("@docspcsubspccod", SqlDbType.Int).Value =
p.p_docspcsubspccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }

        public void Delete_Rec(clsdocspcprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("deldocspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@docspccod", SqlDbType.Int).Value =
p.p_docspccod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public List<clsdocspcprp> Disp_Rec()
        {

            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("disdocspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
```

```
            SqlDataReader dr = cmd.ExecuteReader();
            List<clsdocspcprp> obj = new List<clsdocspcprp>();
            while (dr.Read())
            {
                clsdocspcprp k = new clsdocspcprp();
                k.p_docspccod = Convert.ToInt32(dr[0]);
                k.p_docspcdoccod = Convert.ToInt32(dr[1]);
                k.p_docspcsubspccod = Convert.ToInt32(dr[2]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
        public List<clsdocspcprp> Find_Rec(Int32 docspccod)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("finddocspc", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@docspccod", SqlDbType.Int).Value =
docspccod;
            SqlDataReader dr = cmd.ExecuteReader();
            List<clsdocspcprp> obj = new List<clsdocspcprp>();
            if (dr.HasRows)
            {
                dr.Read();
                clsdocspcprp k = new clsdocspcprp();
                k.p_docspccod = Convert.ToInt32(dr[0]);
                k.p_docspcdoccod = Convert.ToInt32(dr[1]);
                k.p_docspcsubspccod = Convert.ToInt32(dr[2]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
    }
    public class clsrat : clscon
    {
        public void Save_Rec(clsratprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("insrat", con);
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.Add("@ratdoccod", SqlDbType.Int).Value =
p.p_ratdoccod;
            cmd.Parameters.Add("@ratdat", SqlDbType.DateTime).Value =
p.p_ratdate;
            cmd.Parameters.Add("@ratscr", SqlDbType.Float).Value =
p.p_ratscr;
            cmd.Parameters.Add("@ratcom", SqlDbType.VarChar,500).Value =
p.p_ratcom;
            cmd.ExecuteNonQuery();
```

```
            cmd.Dispose();
            con.Close();
        }
        public void Update_Rec(clsratprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("uprat", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@ratcod", SqlDbType.Int).Value =
p.p_ratcod;
            cmd.Parameters.Add("@ratdoccod", SqlDbType.Int).Value =
p.p_ratdoccod;
            cmd.Parameters.Add("@ratdat", SqlDbType.DateTime).Value =
p.p_ratdate;
            cmd.Parameters.Add("@ratscr", SqlDbType.Float).Value =
p.p_ratscr;
            cmd.Parameters.Add("@ratcom", SqlDbType.VarChar, 500).Value =
p.p_ratcom;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void Delete_Rec(clsratprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("delrat", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@ratcod", SqlDbType.Int).Value =
p.p_ratcod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }

        public List<clsratprp> Disp_Rec(Int32 doccod)
        {

            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("disprat", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@doccod", SqlDbType.Int).Value = doccod;

            SqlDataReader dr = cmd.ExecuteReader();
            List<clsratprp> obj = new List<clsratprp>();
            while (dr.Read())
            {
                clsratprp k = new clsratprp();
                k.p_ratcod = Convert.ToInt32(dr[0]);
                k.p_ratdoccod = Convert.ToInt32(dr[1]);
                k.p_ratdate = Convert.ToDateTime(dr[2]);
                k.p_ratscr = Convert.ToSingle(dr[3]);
                k.p_ratcom = dr[4].ToString();
                obj.Add(k);
```

```
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clsratprp> Find_Rec(Int32 ratcod)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("findrat", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@ratcod", SqlDbType.Int).Value = ratcod;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsratprp> obj = new List<clsratprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clsratprp k = new clsratprp();
            k.p_ratcod = Convert.ToInt32(dr[0]);
            k.p_ratdoccod = Convert.ToInt32(dr[1]);
            k.p_ratdate = Convert.ToDateTime(dr[2]);
            k.p_ratscr = Convert.ToSingle(dr[3]);
            k.p_ratcom = dr[4].ToString();
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
}
```

APPENDIX B- Specialties(frmspc)



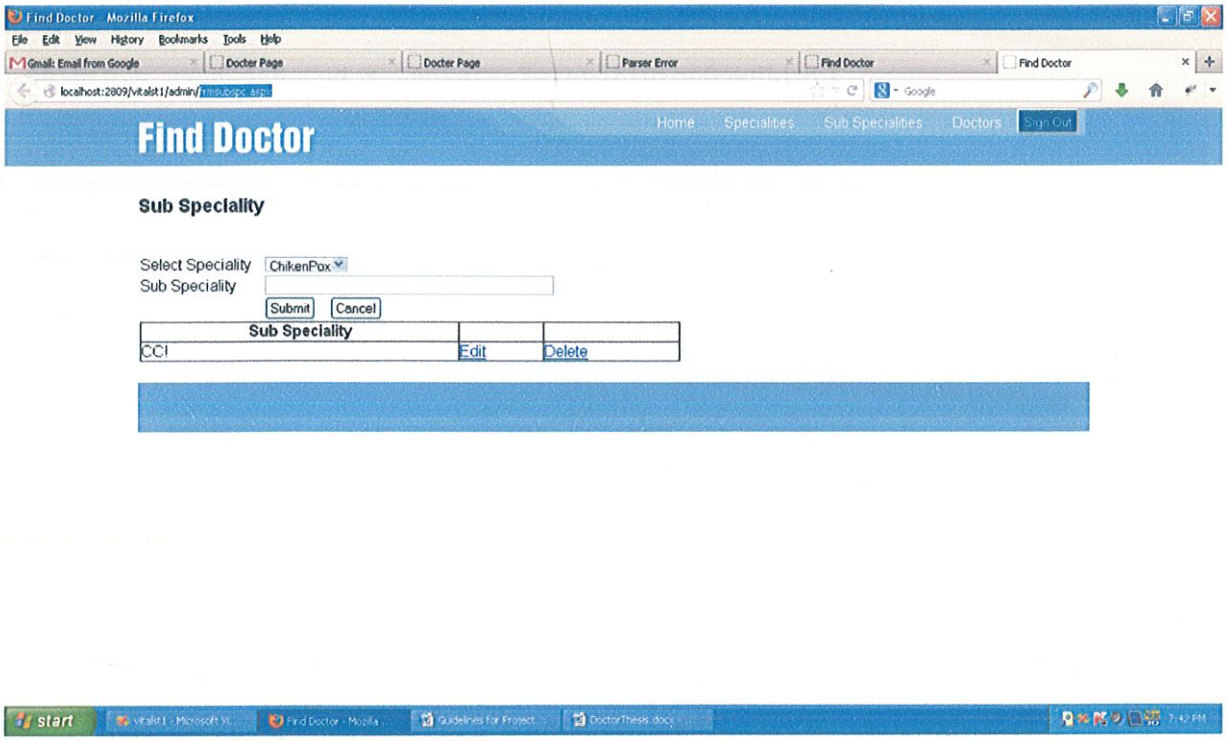
frmspc.aspx.cs

```
using System;

using System.Collections.Generic;

//using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;


public partial class admin_Default : System.Web.UI.Page

{

protected void Page_Load(object sender, EventArgs e)

{


}

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

```
{
nsvitals.clsspc obj = new nsvitals.clsspc();
nsvitals.clsspcprp objprp = new nsvitals.clsspcprp();
objprp.p_spcdsc = TextBox1.Text;
if (Button1.Text == "Submit")
obj.Save_Rec(objprp);
else
{



}
TextBox1.Text = String.Empty;
GridView1.DataBind();
}
protected void GridView1_RowDeleting(object sender,
GridViewDeleteEventArgs e)
{
nsvitals.clsspc obj = new nsvitals.clsspc();
nsvitals.clsspcprp objprp = new nsvitals.clsspcprp();
objprp.p_spccod = Convert.ToInt32(GridView1.DataKeys
[e.RowIndex][0]);
obj.Delete_Rec(objprp);
GridView1.DataBind();
e.Cancel = true;
}
protected void GridView1_RowUpdating(object sender,
GridViewUpdateEventArgs e)
{
nsvitals.clsspc obj = new nsvitals.clsspc();
nsvitals.clsspcprp objprp = new nsvitals.clsspcprp();
objprp.p_spccod = Convert.ToInt32(GridView1.DataKeys
[e.RowIndex][0]);
```

```
objprp.p_spcdsc = e.NewValues[0].ToString();
obj.Update_Rec(objprp);
GridView1.EditIndex = -1;
GridView1.DataBind();
e.Cancel = true;
}
protected void Button2_Click(object sender, EventArgs e)
{
TextBox1.Text = String.Empty;
}
}
```

APPENDIX C - Sub Specialty (frmsubspc.aspx)



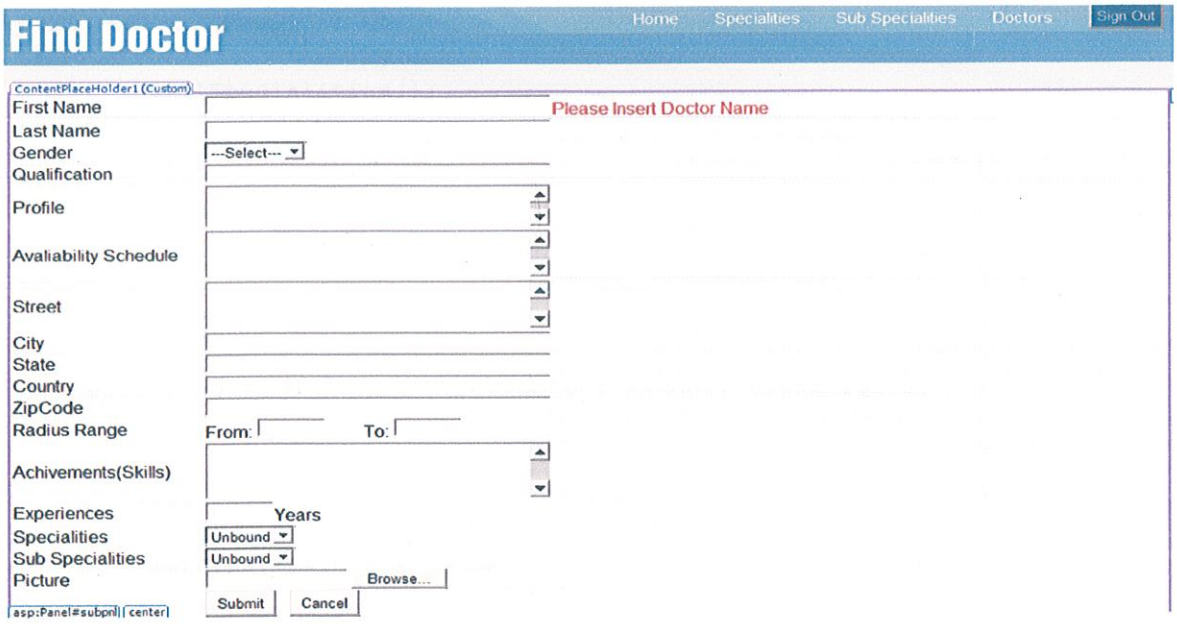
frmsubspc.aspx.cs

```
using System;

using System.Collections.Generic;

//using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;


public partial class admin_Default : System.Web.UI.Page

{

protected void Page_Load(object sender, EventArgs e)

{


}
```

```
protected void Button1_Click(object sender, EventArgs e)
{
nsvitals.clssubspc obj = new nsvitals.clssubspc();
nsvitals.clssubspcprp objprp = new nsvitals.clssubspcprp();
objprp.p_subspcnam = TextBox1.Text;
objprp.p_subspcspccod = Convert.ToInt32(DropDownList1
.SelectedValue);
if (Button1.Text == "Submit")
obj.Save_Rec(objprp);
else
{
objprp.p_subspccod = Convert.ToInt32(ViewState["cod"]);
obj.Update_Rec(objprp);
Button1.Text = "Submit";
}
TextBox1.Text = String.Empty;
GridView1.DataBind();
}
protected void GridView1_RowDeleting(object sender,
GridViewDeleteEventArgs e)
{
nsvitals.clssubspc obj = new nsvitals.clssubspc();
nsvitals.clssubspcprp objprp = new nsvitals.clssubspcprp();
objprp.p_subspccod = Convert.ToInt32(GridView1.DataKeys[e.RowIndex][0]);
obj.Delete_Rec(objprp);
GridView1.DataBind();
e.Cancel = true;
}
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs
e)
{
nsvitals.clssubspc obj = new nsvitals.clssubspc();
```

```
List<nsvitals.clssubspcprp> k = obj.Find_Rec(Convert.ToInt32(
GridView1.DataKeys[e.NewEditIndex][0]));
if (k.Count > 0)
{
TextBox1.Text = k[0].p_subspcnam;
Button1.Text = "Update";
ViewState["cod"] = k[0].p_subspccod;
}
e.Cancel = true;
}
}
```

APPENDIX D –Doctor details (frmdocdetails)



docdetails.aspx.cs

```
using System;

using System.Collections;

using System.Configuration;

using System.Data;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Data.SqlClient;


public partial class admin_docdetail : System.Web.UI.Page

{

  SqlConnection con = new
SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings
["cn"].ToString());

    protected void Page_Load(object sender, EventArgs e)
```

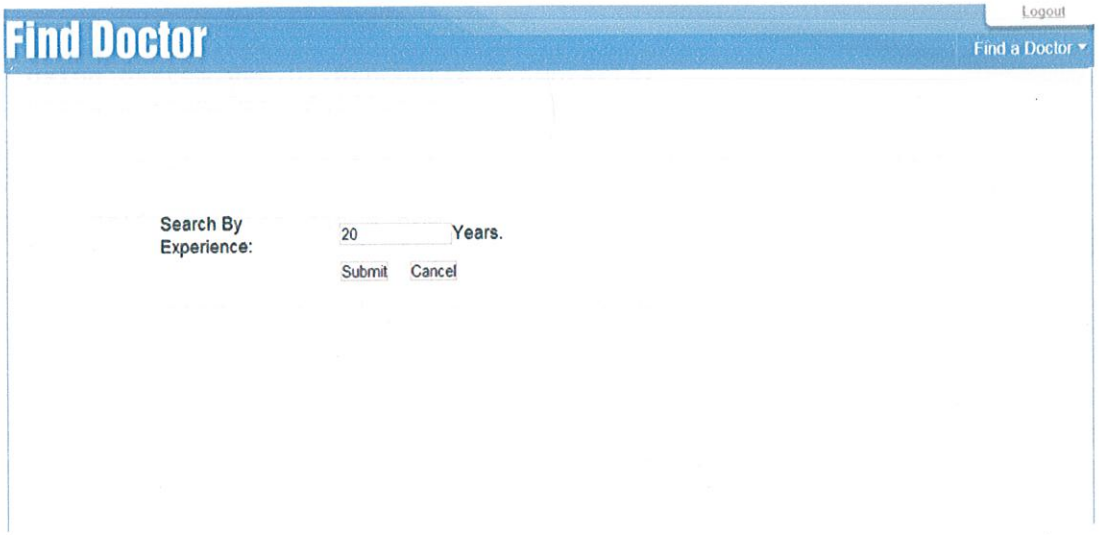
```
{
    if(!Page.IsPostBack)
    {
    fillSpecialities();
       // fillZipcode();
    }
}

protected void submit_Click(object sender, EventArgs e)
{
    addrec();
    mainpnl.Visible = false;
    subpnl.Visible = true;
}
protected void addrec()
{    con.Open();
    string path=Request.PhysicalApplicationPath +"/admin/Images/"+
FileUpload1.FileName ;
FileUpload1.SaveAs(path);
    string insert = "insert into
tbdoc(docfstnam,doclstnam,docgnd,docqal,docprf,docavlsch,docstreet,doccit
y,docstate,doccountry,doczipcod,ddfrom,ddto,dochiglig,docexp,docspec,docs
ubspec,docpic)values('" + txtfstnam.Text + "','" + txtlstnam.Text + "','"
+ ddgen.SelectedValue + "','" + txtqal.Text + "','" + txtprf.Text + "','"
+ txtavlsch.Text + "','" + txstreet.Text  + "','"+ txtcity.Text +"','"+
txtstate.Text +"','"+ txtcountry.Text+"','" + txtzipcode.Text + "','" +
txtfrom.Text + "','" + txtto.Text + "','" + txtach.Text + "','" +
txtexperince.Text + "','" + ddspecil.SelectedItem.Text  + "','" +
ddSubspecil.SelectedItem.Text  + "','" + FileUpload1.FileName + "')";
    //Response.Write(insert);
    //Response.End();
    SqlCommand cmd = new SqlCommand(insert,con);
    cmd.ExecuteNonQuery();
    con.Close();
}
```

```
    protected void fillSpecialities()
    {   con.Open();
        string select ="select * from tbspc";
        SqlCommand cmd = new SqlCommand(select, con);
        SqlDataReader dr = cmd.ExecuteReader();
        ddspecil.Items.Add(new ListItem("----Select----", "0"));
        if (dr.HasRows)
        {  while (dr.Read())
            {
                ddspecil.Items.Add(new ListItem(dr["spcdsc"].ToString(),
dr["spccod"].ToString()));
            }
        }
        dr.Close();
        con.Close();
        con.Close();
    }
    // protected void fillZipcode()
    //{
    //   con.Open();
    //    string select ="select distinct zipcode from  zipcodes ";
    //    SqlCommand cmd = new SqlCommand(select, con);
    //    SqlDataReader dr = cmd.ExecuteReader();
    //    ddZipCode.Items.Add(new ListItem("----Select----", "0" ));
    //    if (dr.HasRows)
    //    {
    //        while (dr.Read())
    //        {
    //            ddZipCode.Items.Add(new
ListItem(dr["zipcode"].ToString(), dr["zipcode"].ToString()));
    //        }
    //    }
```

```
     //    dr.Close();
 //       con.Close();
  //     con.Close();
    //}
 protected void ddspecil_SelectedIndexChanged(object sender, EventArgs e)
{
    fillSubCat();
    }
    protected void fillSubCat()
    {
        con.Open();
        string select = " select * from tbsubspc where subspcspccod =" +
ddspecil.SelectedValue;
        //Response.Write(select);
        //Response.End();
        SqlDataAdapter adp = new SqlDataAdapter(select, con);
        DataSet ds = new DataSet();
        adp.Fill(ds);
        if (ds.Tables[0].Rows.Count > 0)
        {
           for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
            {   ddSubspecil.DataSource = ds;
                ddSubspecil.DataTextField = "subspcnam";
                ddSubspecil.DataValueField = "subspccod";
                ddSubspecil.DataBind();
            }
        }
        con.Close();
    }
protected void Button2_Click(object sender, EventArgs e)
    {
    }
```

APPENDIX E -Search by Experience

input = 20 years



Output in descending order

Name: ori verma Experience: 15 years
Qualification: md Gender: Male
Schedule: 9-5 Skills: ach
Address: mall,shimla,hp,india,171001

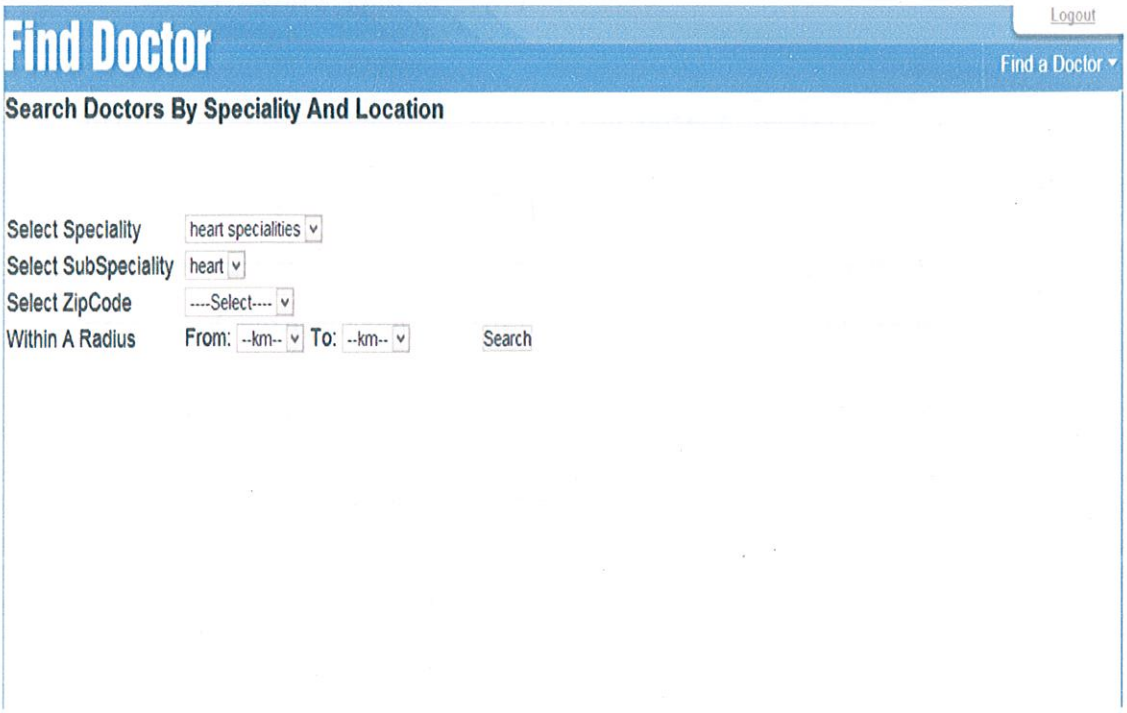
Name: sanjay sharma Experience: 12 years
Qualification: md Gender: Male
Schedule: 9 - 6 Skills: ach
Address: sim,sanjauli,hp,india,171005

Name: nitin kumar Experience: 7 years
Qualification: heart specialist Gender: Male
Schedule: 10 - 6 Skills: ach
Address: rd,totu,hp,india,171111

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

public partial class searchExperience : System.Web.UI.Page
{
    SqlConnection con = new
SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings
["cn"].ToString());
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            fillSearchgrid();
        }

    }
    protected void GridView1_RowDeleting(object sender,
GridViewDeleteEventArgs e)
    {

    }
    protected void GridView1_RowEditing(object sender,
GridViewEditEventArgs e)
    {

    }

    protected void fillSearchgrid()
    {
        con.Open();
        string select = " select * from tbdoc  where docexp=" +
Request.QueryString["docexp"] + "";

        //Response.Write(select);
        //Response.End();
        SqlDataAdapter adp = new SqlDataAdapter(select, con);
        DataSet ds = new DataSet();
        adp.Fill(ds);
        if (ds.Tables[0].Rows.Count > 0)
        {

            for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
            {

                GridView1.DataSource = ds;
                GridView1.DataBind();
            }

        }
        else
        {
```

```
            string select1 = " select * from tbdoc  where docexp <" +
Convert.ToInt32(Request.QueryString["docexp"]) + "  order by docexp desc
";
            //Response.Write(select1);
            //Response.End();
            SqlDataAdapter adp1 = new SqlDataAdapter(select1, con);
            DataSet ds1 = new DataSet();
            adp1.Fill(ds1);
            if (ds1.Tables[0].Rows.Count > 0)
            {

                for (int i = 0; i < ds1.Tables[0].Rows.Count; i++)
                {

                    GridView1.DataSource = ds1;
                    GridView1.DataBind();
                }

            }
        }
        con.Close();
        //pnlmain.Visible = true;
        //pnlsub.Visible = true;
    }
}
```

APPENDIX F- Search by zip code




```
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;

public partial class admin_Default : System.Web.UI.Page
{
    SqlConnection con = new
SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings
["cn"].ToString());
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {

            fillZipcode();
        }
    }
    protected void LinkButton1_Click(object sender, EventArgs e)
    {
        Response.Redirect("frmnewdoc.aspx");
    }
    protected void GridView1_RowDeleting(object sender,
GridViewDeleteEventArgs e)
    {
        //Int32 a = Convert.ToInt32(GridView1.DataKeys[e.RowIndex][0]);
```

```
        //nsvitals.clsdoc obj = new nsvitals.clsdoc();
        //nsvitals.clsdocprp objprp = new nsvitals.clsdocprp();
        //objprp.p_doccod = a;
        //obj.Delete_Rec(objprp);
        //GridView1.DataBind();
        //e.Cancel = true;
    }
    protected void GridView1_RowEditing(object sender,
GridViewEditEventArgs e)
    {
        //Int32 a =
Convert.ToInt32(GridView1.DataKeys[e.NewEditIndex][0]);
        //Response.Redirect("frmnewdoc.aspx?dcod=" + a.ToString());
    }
    protected void Button1_Click(object sender, EventArgs e)
    {

        Response.Redirect("searchLocation.aspx?docspec='" +
DropDownList1.SelectedItem.Text + "'&docsubspec='" +
DropDownList2.SelectedItem.Text + "' &doczipcod='" + ddZipCode.Text +
"'&ddfrom='" + ddfrom.Text + "' &ddto='" + ddto.Text + "' ");

    }
    protected void fillZipcode()
    {
        con.Open();
        string select = "select distinct doczipcod from tbdoc ";
        SqlCommand cmd = new SqlCommand(select, con);
        SqlDataReader dr = cmd.ExecuteReader();
        ddZipCode.Items.Add(new ListItem("----Select----", "0"));
        if (dr.HasRows)
        {
            while (dr.Read())
            {
                ddZipCode.Items.Add(new
ListItem(dr["doczipcod"].ToString(), dr["doczipcod"].ToString()));
            }
        }
        dr.Close();
        con.Close();


    }

    protected void fillSearchgrid()
    {
        con.Open();
        string select = " select * from tbdoc where docspec='" +
DropDownList1.SelectedItem.Text + "' and docsubspec='" +
DropDownList2.SelectedItem.Text + "' and doczipcod='" + ddZipCode.Text +
"' and ddfrom ='" + ddfrom.Text + "' and ddto='" + ddto.Text + "'  ";

        SqlDataAdapter adp = new SqlDataAdapter(select, con);
        DataSet ds = new DataSet();
        adp.Fill(ds);
        if (ds.Tables[0].Rows.Count > 0)
        {

            for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
            {

                GridView1.DataSource = ds;
```

```
                GridView1.DataBind();
            }

        }
        else
        {

            GridView1.DataSource = ds;
        }
        con.Close();
        pnlmain.Visible = true;
        pnlsub.Visible = true ;
    }
}
```