Design and Development of Online Doctor Appointment System

Presented By:
Md. Maharaz Hossain
Reg. Number: WUB 03/13/28/772
Roll: 772
Batch: 28 B CSE

Sharief Mohammad Zamshed
Reg. Number: WUB 03/13/28/764
Roll 764
Batch: 28 B CSE

Supervisor:
Md. Mahin
Lecturer
Department of Computer Science & Engineering
B.Sc in CSE

Approval

We Here by submit this Project Report for Examination with the approval of the project supervisor.

Signature……………………..…….……………..……….. Date………………..…………………....…..

Kazi Hassan Robin
Asstt. Professor & Head
Department of Computer Science & Engineering
M.Sc. in IT, University of East London, United Kingdom
Member, British Computer Society (MBCS)

DEDICATION

We wish to dedication this entire project report to our beloved mothers and fathers for their tireless support they accorded us ever since we were children.

ACKNOWLEDGEMENT

First and for most, we would like to express our sincere thanks to the almighty ALLAH for the gift of life, wisdom understanding **he has given to us**, a reason for our existence and to our families for the love and support they had been provided throughout our life.

We also thank the staff of Doctor Attendants at **Farazy Hospital Ltd.** for cooperation during our System Study and Analysis stage they had been particularly helpful in providing the necessary data about the manuals patient record management system.

Special thanks go to **Md. Shahazalal Shahin** for initiating the ideas for our research topic hence establishing a framework for the project proposal. He was very co-operative to us .

Md. Mahin whom we regard as our parent and supervisor, we thank him for the expertise and intelligence he has displayed whole supervising this project. We believe this good work in a result of his good guidance and cooperation.

We cannot forget our friends in the faculty of Computer Science & Engineering for the academic interactions and Ideas.

Lastly, we would like to convey our gratitude to the Department of Head Kazi Hassan Robin in our Faculty for the good of during the 4 month class period of our course Coordinator and All CSE Faculty Members.

May the good lord bless them and keep them safe. We love you all.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABUSTRACT

Online Doctor appointment System in hospital today necessitate a competent administration when handling patients, generating reports for cashier, patient details which serves as a key factor for the flow of business transactions in Farazy Hospital Ltd. Unfortunately the current Record Management System Leads to misplacement of during details, Patient details and doctor record of reports and insecurity to records. This research project is aimed at computerizing all the records about Patients, Hospital and Doctors. In order to achieve this goal, a through System Study and investigation was carried out and data was collected and analyzed about the current system using document and data flow diagrams. The concept of report production has been computerized hence, no more delay in report generation to the Hospital, Patient and Doctors. Errors made on hand held calculators and dealt out completely. The method used to develop the system include iterative water full model approach, dataflow, logical and entity relationship diagram where used to design the system and finally the language used SQL Server 2008, C# & ASP.Net.

LIST OF ABBREVIATIONS

Terminology	**Meaning**
ODAS	Online Doctor Appointment System
ERD	Entity Relationship Diagram
DBD	Database Diagram
DFD	Data Flow Diagram
IT	Information Technology
C#	C Sharp
ASP	Active Server Page
HTTP	Hyper Text Transfer Protocol
SQL	Structure Query Language
DBMS	Database Management System
GUI	Graphical User Interface
IDE	Integrated Development Environment

CHAPTER 1
INTRODUCTION

1.0 Introduction

Aim of this project is to create doctor patient handling management system that will help doctors in their work and will also help patients to book doctor appointments and view medical progress. The system allows doctors to manage their booking slots online. Patients are allowed to book empty slots online and those slots are reserved in their name. The system manages the appointment data for multiple doctors of various date and times. Each time a user visits a doctor his/her medical entry is stored in the database by doctor. Next time a user logs in he may view his/her entire medical history as and when needed. At the same time a doctor may view patient's previews medical history while the patient visits him. The system also consists of Blood donor module. This module allows for Blood donation registration as well as Blood group search. The module is designed to help urgent Blood requirements through easy/instant searches.

1.2 Objectives

- To Create Web based online Doctor Appointment management system.
- To manage all patients related information.
- To provide emergency information in critical situation.

1.3 Justification of study

We implement this system for better user experience. This system is very easy to access. Also for establish real time communication, using modern and updated technology. So, user can see the update without reload or refresh. This system will compatible with user device such as pc, laptop, tab & smart phone. So user can easily access the system anytime anywhere. This system is very simple & user friendly so, any user can use this system easily.

1.4 Scope of Study

Scope of the project is very broad in terms of other online doctor appointment portal.
Few of them are:

- Patient and doctor get SMS from the system.
- There is huge collection of doctor information.
- Anyone can get Blood on time.

CHAPTER 2
LITERATURE REVIEW

2.1 Waiting Time

Waiting time simply means a period of time which one must wait in order for a specific action to occur, after that action is requested or mandated (Fernandes et al., 1994). Patients' waiting time has been defined as "the length of time from when the patient entered the outpatient clinic to the time the patient actually received his or her prescription" (Jamaiah, 2003). It is defined as the total time from registration until consultation with a doctor. There were two waiting times, the first is time taken to see a physician and the second is time to obtain medicine (Suriani, 2003). This paper deals with the waiting time to see physicians. Long waiting times are a serious problem for patients using urban health centres in developing countries (Bachmann, 1998). A block appointment system was introduced and evaluated in a large South African health centre. Waiting times of all patients were measured over one-week period before and after the implementation of appointments. Focus groups and individual interviews were conducted with staff and patients. After introducing appointments, patients with acute and chronic illnesses and having appointments had significantly shorter waits time than similar patients without appointments (Mahomed, 1998). Appointments had no benefits for patients not seeing doctors or collecting repeat medication. There was, however, an overall increase in patients' waiting times after introducing the system, mainly due to one typical day in the follow-up study. Focus groups and interviews revealed that staff were skeptical at baseline but at follow-up were positive about the system. Patients were enthusiastic about the appointment system at all stages. The study shows that block appointments can reduce patients' waiting times for acute patients, but may not be suitable for all patients. Staff and patients had different views, which converged with experience of the new system (Mahomed, 1998).

2.2 Patients' Appointment System

A patient appointment system or appointment schedule for health care center started long time ago (Harper, 2003). Management of patients' appointments has earlier works and has developed simplified queuing models and fairly static scheduling conditions. Another attempt was made to calculate the waiting time between patient and doctor using the mathematical queuing models to minimize waiting time (Gamlin, 2003). However; traditionally the appointment system has considered that the doctor time is more important than patient time (Wijewickrama, 2005). So an appointment system was designed to minimize the doctor idle time but current designing of an appointment system is based on decisive factors with respect to both the patient and doctor (Takakuwa, 2005). The patient appointment system has complex structures because it represents the patient appointment time in the healthcare center and controls the patient waiting time based on the type and the period of patient appointment (Gamlin, 2003). Moreover, a patient appointment system is International Journal of Computer Science & Information Technology (IJCSIT) Vol 6, No 4, August 2014 62 meant for: managing doctor's time, reducing patient's waiting time, reducing doctor's idle time, reducing nurse's idle time, and improving the quality of service in the health care (Harper, 2003).

2.2.1 Appointment Delay

Past research shows that the longer the appointment delay which is defined as the time between the day a patient requests an appointment and her actual appointment date, the higher the chances that he/she will cancel or not show up (Gallucci et al. 2005). This suggests an obvious way of minimizing no-shows and cancellations: this is done by asking the patients to come right away or make appointment requests on the day they want to be seen (Murray, 2000). This is called an open access (OA) or advanced access policy (Tantau, 2000), and of late it has become a popular paradigm in practice and the subject of active research. Several authors report on their experiences in implementing OA, both positive and negative (Dixon et al. 2006). Some practitioners strongly advocate OA (Murray and Tantau 2000), and there are some who are strongly against it (Lamb, 2002).

2.2.2 Managing Patients' Appointment system

According to Dexter (1999), managing patient appointment system is a computer application used to manage and reduce the patient waiting time in the health care center. Some health care centers do not use any appointment system. So it has a longer average patients' waiting time than the health care center that adopts the patients' appointment system. While patients can wait for more than one hour to be attended to by a physician in a health care center, they also can feel that they are being disregarded and treated unfairly. So when patients are given the time of appointment in a health care centre, they can evaluate the quality of service in the centre (Dexter, 1999). Hence, developing patients' appointment process for health care centres necessitates the use of a sophisticated queuing model that captures much of the real system's features (saving time, reducing idle time, etc). Therefore the appointment schedule represents the real situation in the health care centre faced by patient appointment schedulers (Rohleder, 2002). On the other hand, the standard practice for scheduling and processing patient appointments are based on the nature of treatments of the patients and that better approaches more sensitive to patient needs are desirable (Klassen, 2002).

2.3 Online Booking System

An online system is also known as a web based system. A web is made up of page that is commonly known as web page or web site, and a web site is a computer program that runs a web server that provides access to a group of related web pages (Alex, 2000). A system is a set of independent components working together to achieve a common objective. Therefore a web based system is a system that is accessible over the internet by a user in order to achieve a particular task for a given purpose. The Internet is a system that is use to connect computers and computer networks. It helps to link millions of computer networks all over the world and it allows the users to get information stored on other computers from a long distance (James, 1999). According to Chua (2010) the public demand for better healthcare system and the alarming number of missed appointments have forced the healthcare sector to recognize how they deliver care services. With the advance of IT technology today and seen healthcare system as a critical system, appointment booking system lies at the intersection of delivering efficient, dependable and timely access to health services. The conventional way of appointment booking is via fax, phone or email. But with the growing internet penetration, healthcare industry is moving towards the use of an online appointment booking system. A web-based appointment system is used in Taiwan; everyone is required to enroll in the national health insurance program. When one needs health service, he shows his health insurance card to doctors in an hospital to start with. There are several ways of making an appointment. A person can either go to the hospital directly for consultation day by day or make an appointment from home through phone call or email if his condition is not emergent (Gruca, 2004). The Internet provides a wide range of technologies that enable hospitals to communicate with their patients. Recently, as the prevalence of Internet increasing, many hospitals initiated the website appointment system. Electronic patient-provider communication promises to improve efficiency and effectiveness of clinical care (Wakefield, 2004)

2.4 Existing Hospital Appointment Schemes

One application developed to manage patients' appointment scheduling has used exponential enter arrival times. This model assumes that the exponential enter arrival times could not be directly validated by date, and it is limited due to the nature of the appointment scheduling (Rohleder, 2002). Since appointments are scheduled in the future, the exact model of call arrivals will only have limited impact on measures related to the time between the call and the appointment time. For this reason, the challenge for making appointment system is designing a suitable system based on the health care procedure environment (Klassen, 2002). Hence, the appointment provider in the health care center can schedule a patient into an appropriate time slot on a given day. Klassen (2004) developed another method for managing patients' appointment using multiple schedule appointment in multiple period environments. Patients can call for any appointment time but if the period time is full, they should replace the appointment to another time. Moreover, various combinations for multi appointment and double booking are measured and recommended for different operational use depending on the health care environment because the varying appointment request has little effect on appointment system performance, especially maintaining acceptable performance, except when the system has the overloaded option (Rohleder, 2004). Many studies about patients' appointment have found that there are rules or policies for scheduling appointment system such as no scheduling for more than 20 or 30 clients and the best schedule is to place two patients in the first appointment and spread the rest consistently over a period based on average service times (Klassen, 2004). On the other hand, a patient can call for an appointment without knowledge of the type of appointment and appointment queue number and the patient is not aware whether the appointment is variable or not. Sometimes the exact duration for each patient can be known but at other times this is unknown (Rohleder, 2004). Another system developed by Mustafa, (2004) allows a registered patient, having user name and password, to access and explore the list of physicians alphabetically and select a physician whose email contact and profile are also provided. A patient can also view the physician working calendar to find out his/her working and non working day to make an appointment. When the patient selects, view calendar the patient can then choose any valid day in any month to make an appointment (Mustafa, 2004). After that, the patient will receive an e-mail from the system to confirm the appointment time or to inform the patient that the selected time is already taken by another patient or blocked by the physician. In general, the patient appointment system provides all the choices and the capabilities to the patients, such as selecting a physician, selecting the time of appointment, and allows them to access the health care system day or night and schedule their own appointments using the Internet without spending time holding for a nurse or having lengthy phone calls. Wijewickrama and Takakuwa (2005) opine that the health care operating time (due time) is from 8:30 am to 5:30 pm during the week days. Throughout this period, four types of patients arrive to have a consultation appointment in the health care center-appointed patients, same day appointment patients (walk-ins), patients who come for a medical test and new patients (Wijewickrama, 2005). Patients who have appointments are given priority over those who walkin for consultation. Consequently, these latter patients have to wait a long time in the waiting room to meet a doctor even if the consultation time only last few minutes (Takakuwa, 2005). Porta-Sales et al. (2005) developed another system. The main concept of the system is contacting, screening and scheduling appointment with the health care center initially by an expert nurse and the patient initiating contacting with the health care center using the telephone. Moreover, the health care center can be accessible from different places. So there should be PC resources and PC consultations to be accessed from different sources, from other hospitals, from general practitioners, or even from the patients themselves. Porta-Sales et al. (2005) studied 534 patients for a period exceeding one year. After the first visit, 195 patients did not return for the second scheduled appointment and 203 patients had progressed on to the third scheduled visit. The main reason given for the scheduled visits was admission into the health care; the median time-lapse between the first and second visit was 21 days, between the second and third was 27.5 days and between the first and third was 48 days. Comparing patients, who did not attend the three consecutive visits with those who did, indicated that the former had (at the first visit) a lower performance status. Su et al. (2003) studied in a private hospital which has several clinics. For each clinic, the average patient load is 20 per consultation section (morning or afternoon) and the health care system adopts both a patient appointment model and patient registration model. The system allows patients to have self-selected specific physicians for consultation and registration (Shih, 2003). The management appointment system studied by Su and Shih, (2003) is based on the first 20 reserved for scheduled patients, after that, only seven are offered for scheduling. Odd numbers after 20 are left for walk-ins. The arrival time of the first patient is assumed to be the same as the clinic starting time. The scheduled patients are assigned based on 3- main intervals and are also informed about their appointed arrival times (Su, 2003). If the scheduled patient does not appear on time, the next available patient receives consultation immediately. The management operating philosophy of services here is based on "first in, first seen" to limit patient waiting time. Therefore, patients can walk-in to see a physician, when patient shows up at the appointed time (Shih, 2003). Some of the existing appointment booking system have some limitations and the system developed in this research eradicate the limitations of the existing system in confirming patient medical appointment by sending an email to the patient if the appointment have being confirmed or not. It will also enable the patients to view and monitor their medical records online.

CHAPTER 3
METHODOLOGY

3.0 Methodology

➢ Waterfall Model

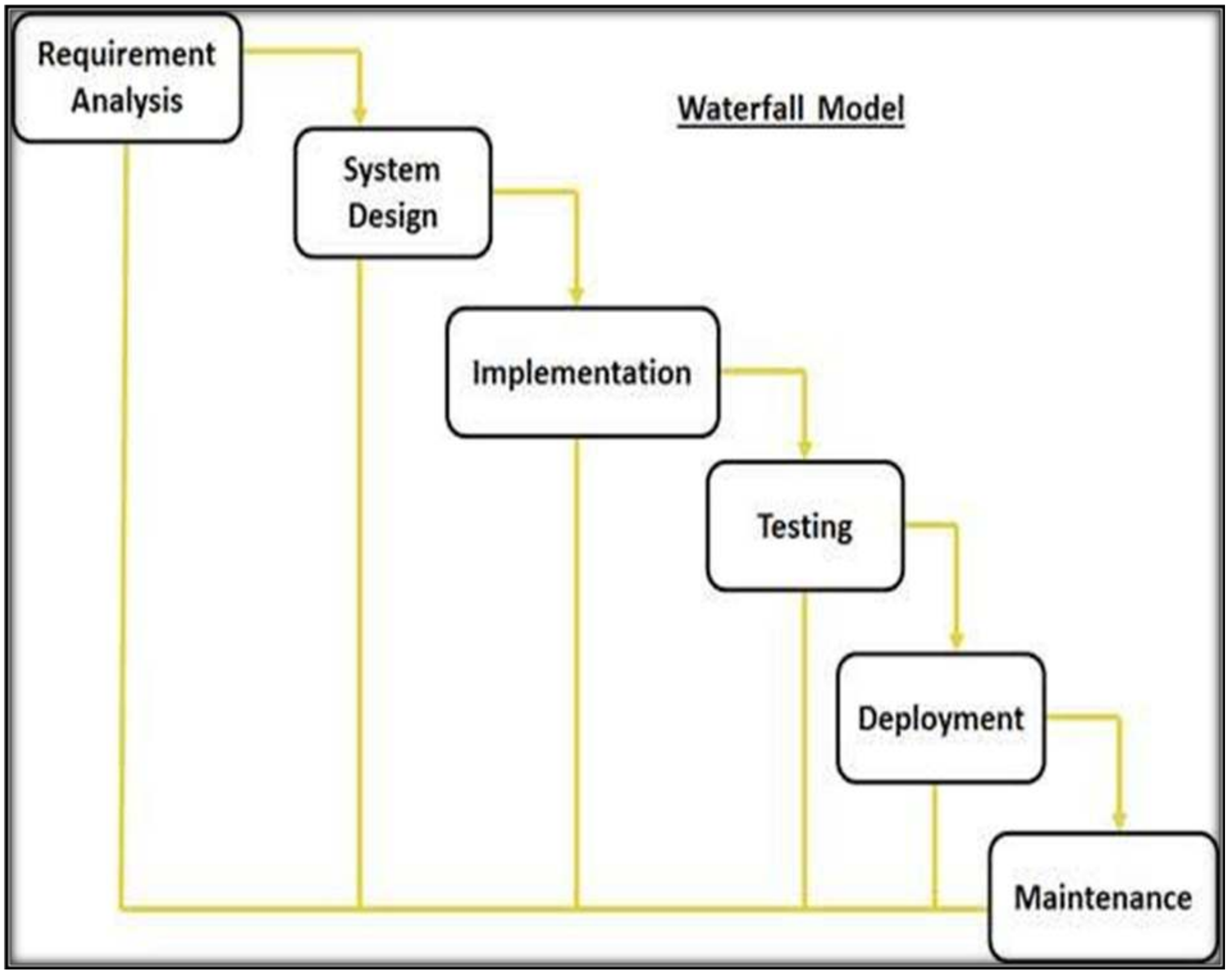


Fig. 3.0 Methodology

3.2 Justification of Methodology

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- The project is short.
- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model . each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Easy to arrange tasks.

1.3 Description of Methodology

The sequential phases in Waterfall model are:

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system:** Once the functional and nonfunctional testing is done, the product is deployed in the customer environment or released into the market.

- **Maintenance:** There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

CHAPTER 4
ANALYSIS, DESIGN AND DEEVELOPMENT

4.0. Introduction

The chapter describes the system study, analysis, design strengths and weaknesses of the current system, Contest level diagrams, Entity Relationship Diagram, Architectural design.

4.1 System Study

The study was carried out at Patient, Doctors and Hospital the main purposed of the study was to find out how the process of recording patient's data is carried out. The system that is currently being used in Patient, Doctor and Hospital is entirety manuals. When a patient requests all the information is recorded manually from the appointment then the system are very lazy and more hesitation from the real information, doctor availability and proper time maintenance of the doctor appointment system.

4.2 System Analysis

During the system study phase, requirements of Online Doctor Appointment System (ODAS) were categorized into user requirements, system and hardware requirements.

4.2.1 Existing Online Doctor Appointment Systems

Refer to the literature review, observation, interviews and questionnaires as explained in chapter three it should be noted that at Hospital doctor maintenance we were able to analyze existing systems as discussed below.

The current system was manual where data is written on different papers and transferred to the different departments, human errors were vulnerable since it was paper based and retrieval of files was time consuming as they had to manually locate patient some of which were even lost and thus finding such information was hard. Per the statistics carried 90% of the users were not contented with the system reason that is was not secure in terms of security and storage as it was prone to damages like loss of important information, worn out papers, out break of lire, The speed of recording and retrieval Patients information was average yet 10% were some ok with the system reason that the paper work can used for future reference.

The users recommended that the proposed system should be user friendly, multipurpose enough to handle a number of users at a go, could generate feedback when request is submitted and a use of passwords which could deny access to unauthorized users of system which ensured security. Context diagrams, Data flow diagrams and Entity Relationship Diagram (ERD) where used in the analysis and design of the system.

4.2.2 Requirements Specifications

After analyzing the data collected, we formulated a number of requirements namely user requirement, system hardware software attribute. These were grouped as user, functional, non-functional and systems requirements.

4.2.3 User Requirement

During data collection, the we investigated and found out how the current system operates, not only that but also tried out which problems are faced and how best they can be settled. The users described some of the basic requirements of the system this includes Search for Patients, Register Patient, Update record, Doctor information record, view doctor availability record and view all types of reports.

4.2.4 Functional and Non Functional Requirements

The following is the desired functionality of the new system.

Accept of submissions in form of raw patients; perform analysis of financial to authenticate the users of the system.

And non functional requirement include the following

The system must verify the validate all user input ant user must be notified in case of errors detected in the database, the system should allow room for expansion.

4.2.5 System Requirement

This section describes the hardware components and software requirements needed for effective and efficient running of the system

Table: 4.0 Hardware Requirements

SL	**Hardware**	**Minimum System Requirement**
01	Processor	2.4 GHz Processor speed
02	Memory	2 GB RA
03	Disk Space	500 GB

Table: 4.1 Software Requirements

SL	**Software**	**Minimum System Requirement**
01	Operating System	Windows Server 2008,Windows7
02	Database Management System	Microsoft SQL Server 2014
03	Runtime Environment	Visual Studio 2008 Team System

The table above shows software requirements recommended to enable the system to run as required for using Online Doctor Appointment System (ODAS).

4.3. System Design

After interpretation of the data, tables were drawn and process of data determined to guide the researcher of the implementation stage of the project. The tools, which were employed during this methodology stage, where mainly tables, Data Flow Diagrams and Entity Relationship Diagrams. The design ensures that only allows authorized users to access the systems information.

4.1.3 Logical Model

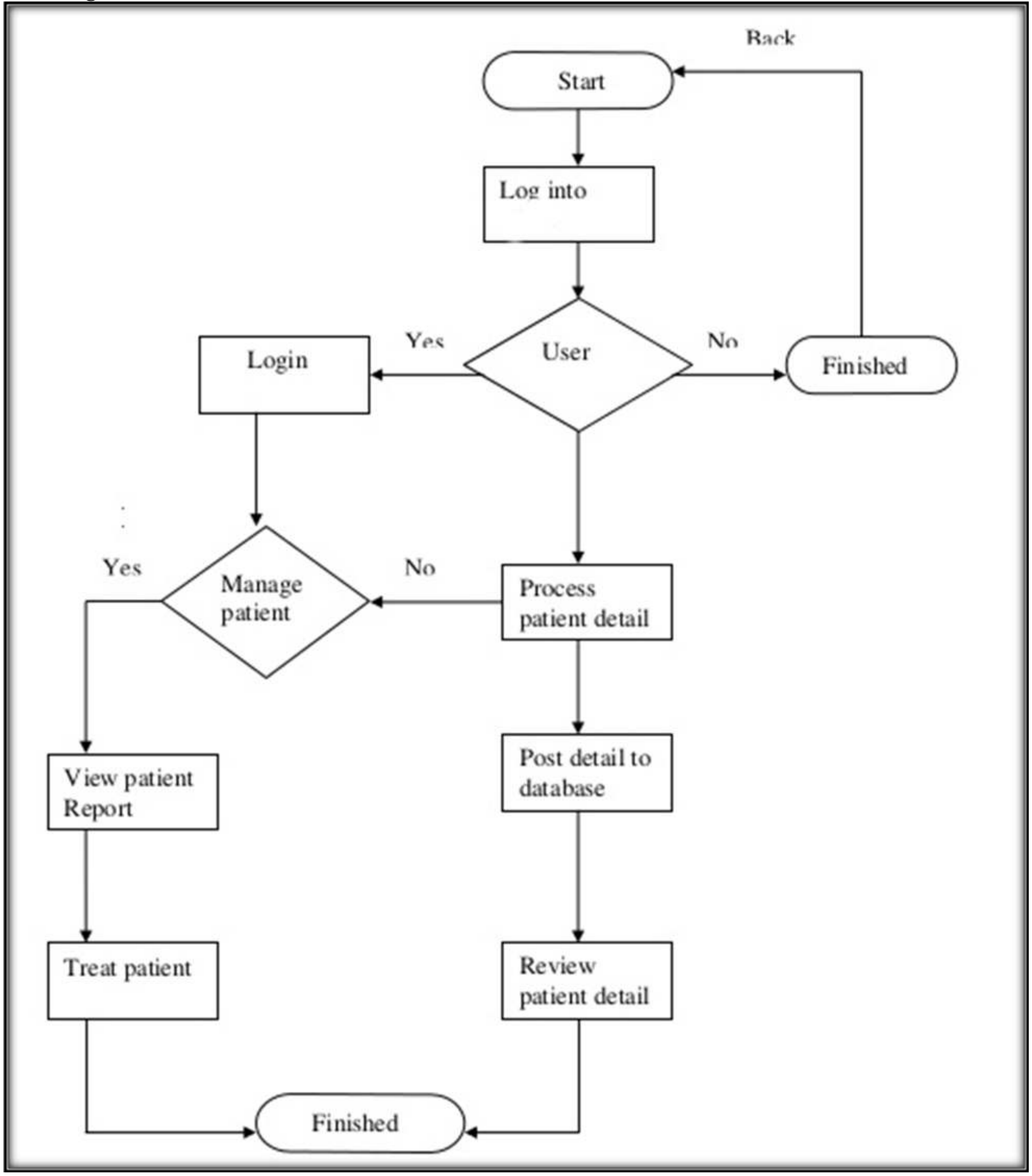
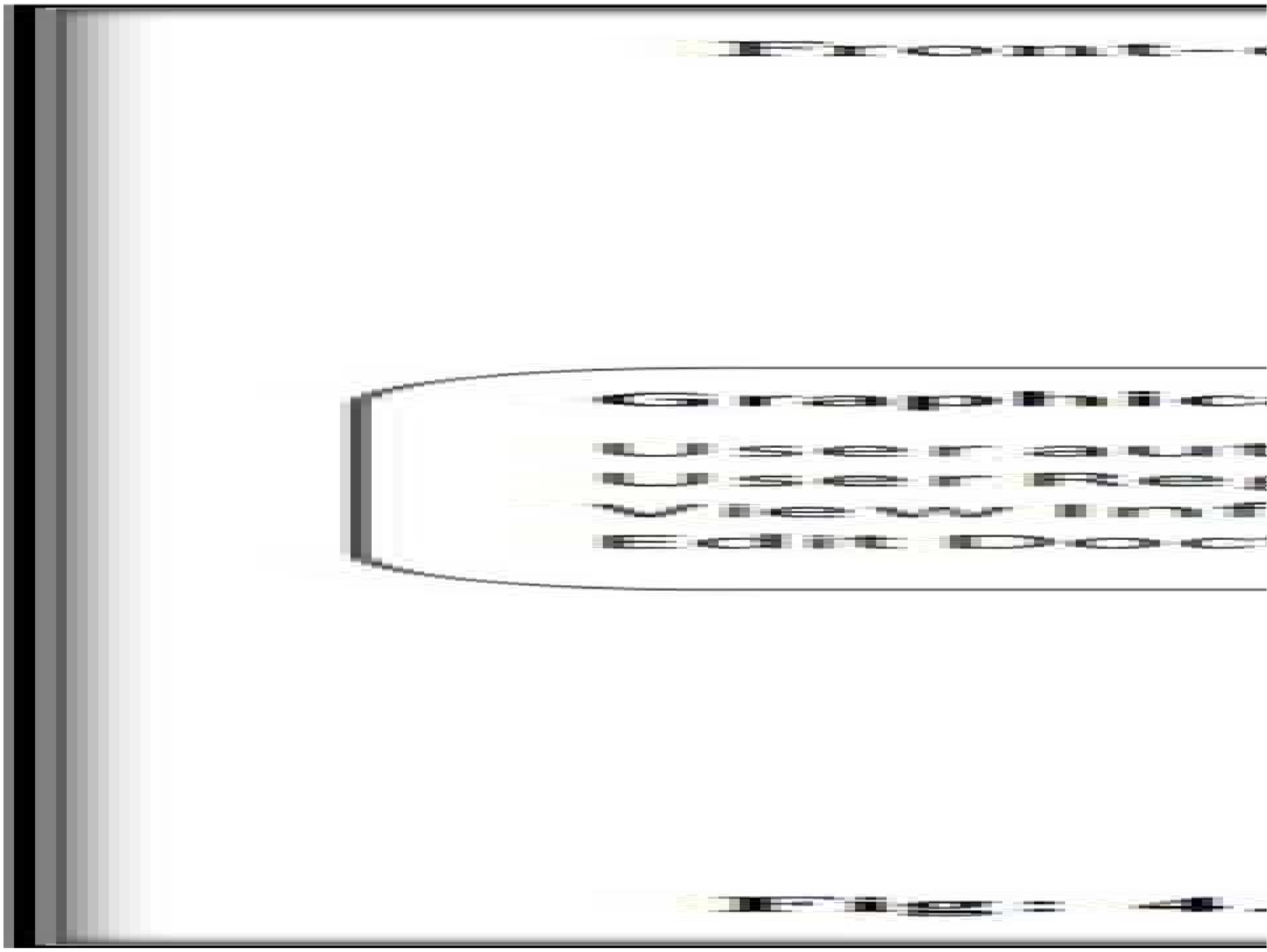


Fig: 4.1.3 Logical Model

4.3.2 System Architecture

This gives a high level view of the new system with the main components of the system and the service they provide and how they communicate. The system is implemented using a three-tier architecture that comprises of our interface, process management and DBMS as illustrated bellow.



4.3.3 Entity Relationship (E-R) Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. An entity relationship diagram is a means of visualizing how the information a system produces is related.

Entity

Which are represented by rectangle. An entity is an object or concept that has its existence in the real world. It includes all those things about which data is collected. A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.

Attributes

Which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.

An Entity Set

It is a set of entities of the same type that share the same properties, or attributes.

Process

A process shows a transformation or manipulation of data flows within the system.

Actions

Which are represented by diamond shapes, show how two entities share information in the database.

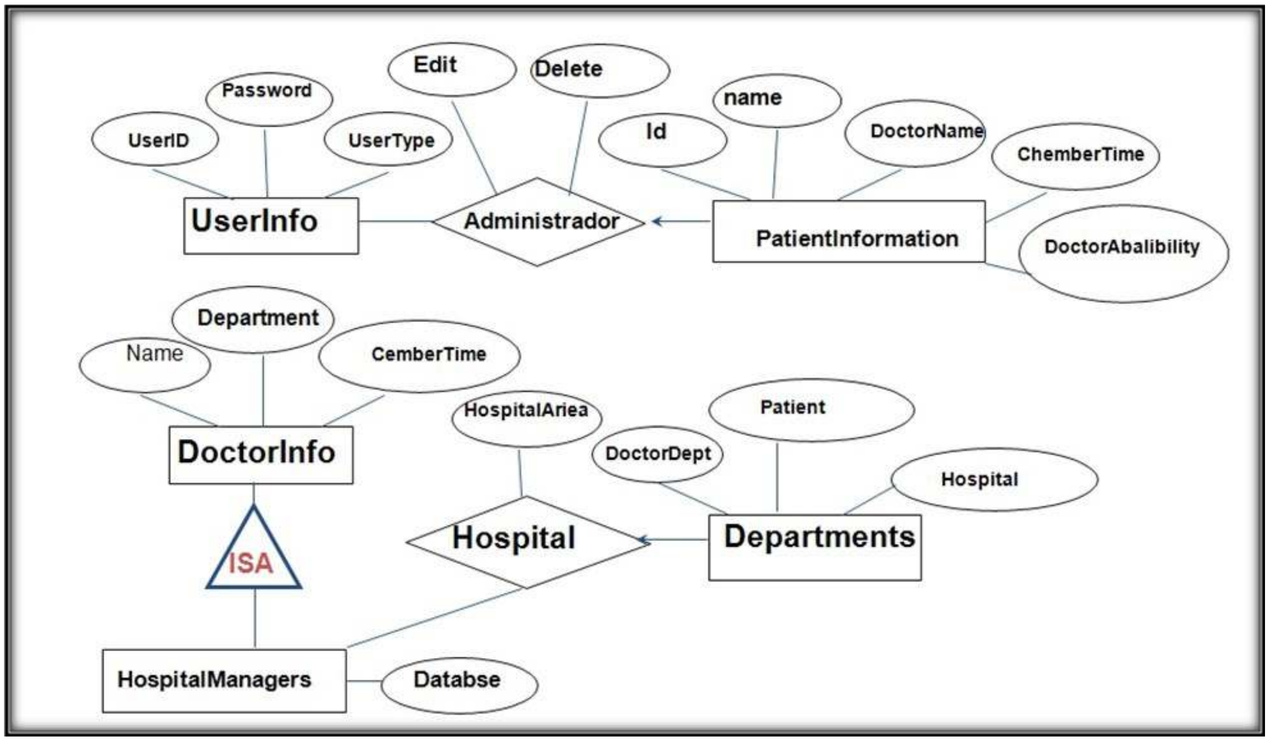


Fig: 4.3.3 Entity Relationship Diagram

4.3.4 Data Flow Diagram

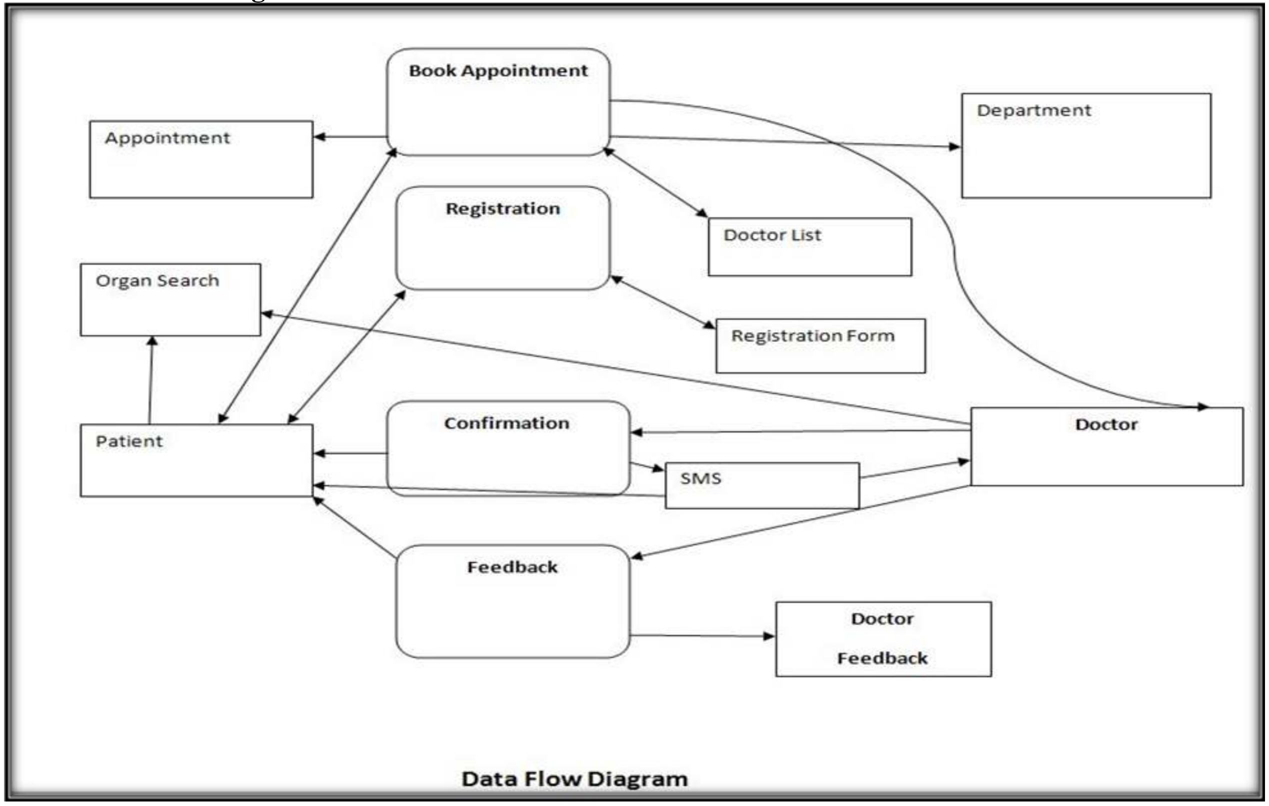
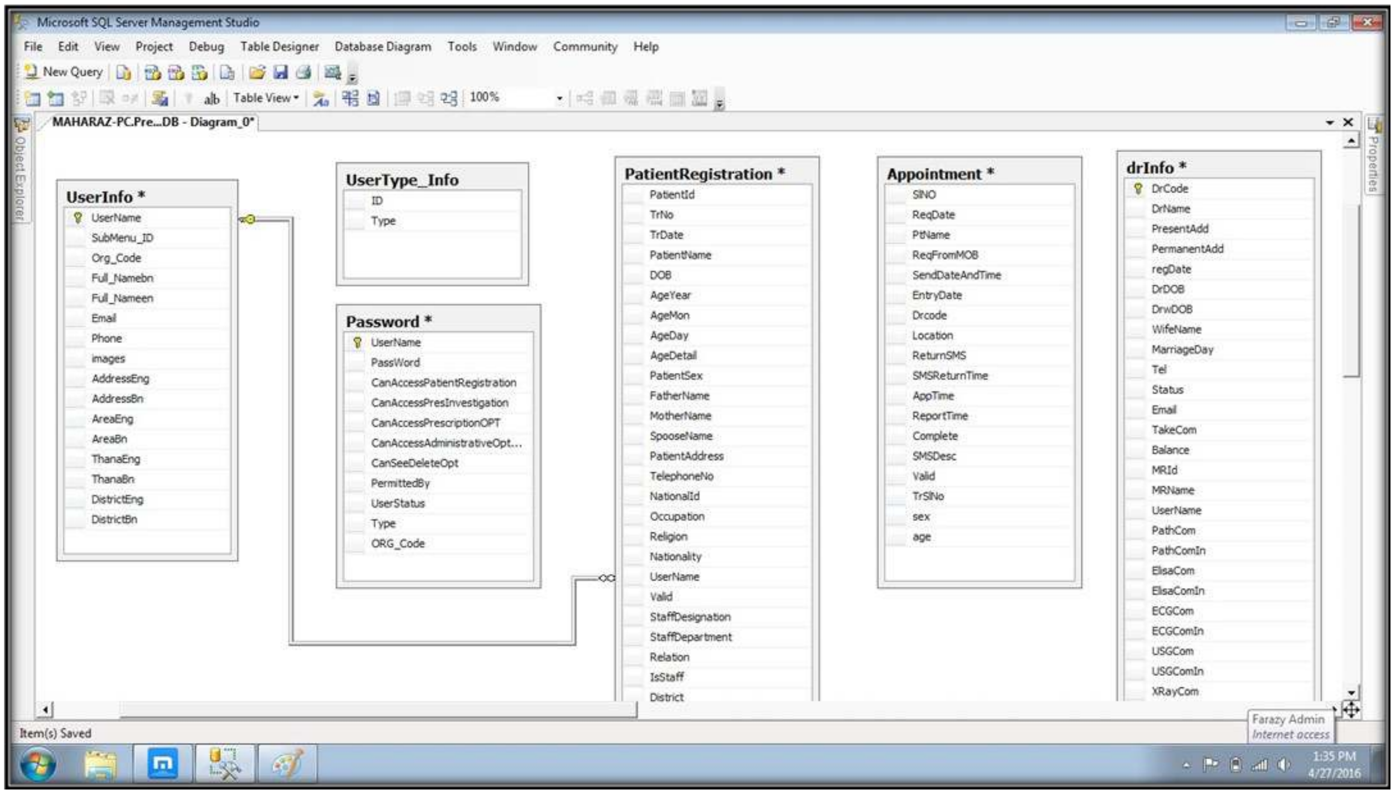


Fig. 4.3.4 Data Flow Diagram


Microsoft SQL Server Management Studio
File Edit View Project Debug Table Designer Database Diagram Tools Window Community Help
New Query
Table View
100%
MAHARAZ-PC.Pre...DB - Diagram_0*
Object Explorer
Properties
Item(s) Saved
Farazy Admin
Internet access
1:35 PM
4/27/2016

UserInfo *
- UserName
- SubMenu_ID
- Org_Code
- Full_Namebn
- Full_Nameen
- Email
- Phone
- images
- AddressEng
- AddressBn
- AreaEng
- AreaBn
- ThanaEng
- ThanaBn
- DistrictEng
- DistrictBn

UserType_Info
- ID
- Type

Password *
- UserName
- PassWord
- CanAccessPatientRegistration
- CanAccessPresInvestigation
- CanAccessPrescriptionOPT
- CanAccessAdministrativeOpt...
- CanSeeDeleteOpt
- PermittedBy
- UserStatus
- Type
- ORG_Code

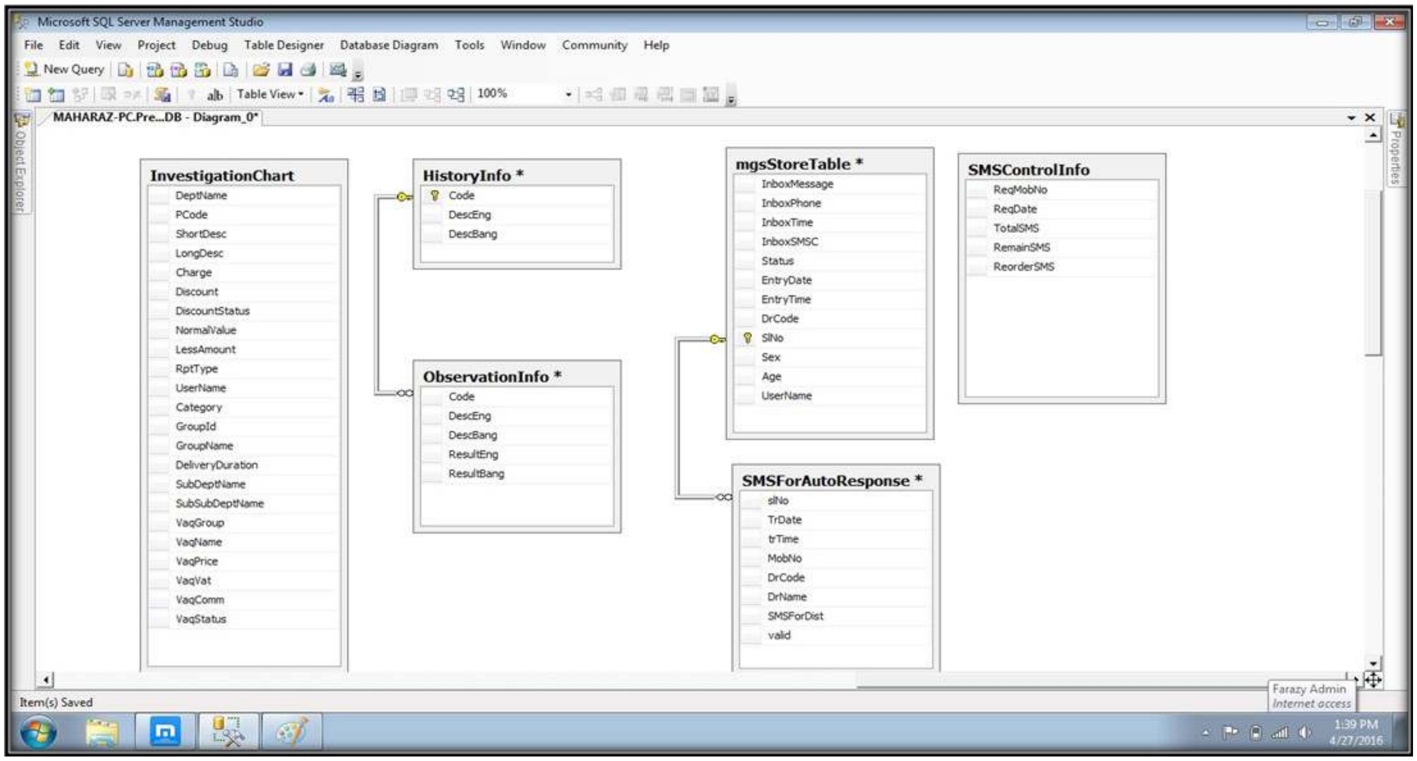
PatientRegistration *
- PatientId
- TrNo
- TrDate
- PatientName
- DOB
- AgeYear
- AgeMon
- AgeDay
- AgeDetail
- PatientSex
- FatherName
- MotherName
- SpooseName
- PatientAddress
- TelephoneNo
- NationalId
- Occupation
- Religion
- Nationality
- UserName
- Valid
- StaffDesignation
- StaffDepartment
- Relation
- IsStaff
- District

Appointment *
- SlNO
- ReqDate
- PtName
- ReqFromMOB
- SendDateAndTime
- EntryDate
- Drcode
- Location
- ReturnSMS
- SMSReturnTime
- AppTime
- ReportTime
- Complete
- SMSDesc
- Valid
- TrSlNo
- sex
- age

drInfo *
- DrCode
- DrName
- PresentAdd
- PermanentAdd
- regDate
- DrDOB
- DrwDOB
- WifeName
- MarriageDay
- Tel
- Status
- Email
- TakeCom
- Balance
- MRId
- MRName
- UserName
- PathCom
- PathComIn
- ElisaCom
- ElisaComIn
- ECGCom
- ECGComIn
- USGCom
- USGComIn
- XRayCom

4.4 Database Diagram

Fig: 4.4 Database Diagram I


Microsoft SQL Server Management Studio
File Edit View Project Debug Table Designer Database Diagram Tools Window Community Help
New Query
Table View
100%
MAHARAZ-PC.Pre...DB - Diagram_0*
Object Explorer
Properties
Item(s) Saved
Farazy Admin
Internet access
1:39 PM
4/27/2016

InvestigationChart
DeptName
PCode
ShortDesc
LongDesc
Charge
Discount
DiscountStatus
NormalValue
LessAmount
RptType
UserName
Category
GroupId
GroupName
DeliveryDuration
SubDeptName
SubSubDeptName
VaqGroup
VaqName
VaqPrice
VaqVat
VaqComm
VaqStatus

HistoryInfo *
Code
DescEng
DescBang

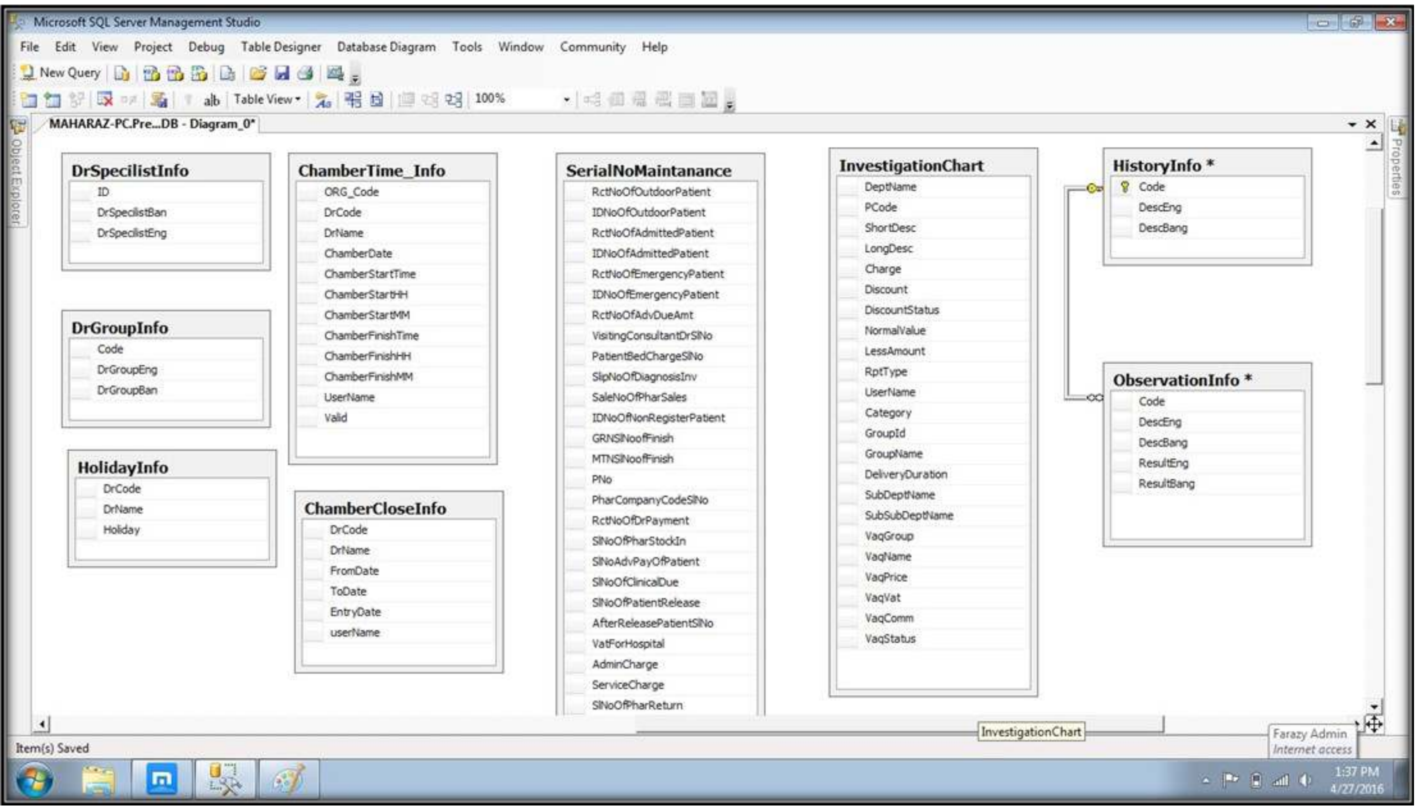
ObservationInfo *
Code
DescEng
DescBang
ResultEng
ResultBang

mgsStoreTable *
InboxMessage
InboxPhone
InboxTime
InboxSMSC
Status
EntryDate
EntryTime
DrCode
SlNo
Sex
Age
UserName

SMSControlInfo
ReqMobNo
ReqDate
TotalSMS
RemainSMS
ReorderSMS

SMSForAutoResponse *
slNo
TrDate
trTime
MobNo
DrCode
DrName
SMSForDist
valid

Fig: 4.4 Database Diagram II


Microsoft SQL Server Management Studio
File Edit View Project Debug Table Designer Database Diagram Tools Window Community Help
New Query
Table View
100%
MAHARAZ-PC.Pre...DB - Diagram_0*
Object Explorer
Properties
InvestigationChart
Item(s) Saved
Farazy Admin
Internet access
1:37 PM
4/27/2016

DrSpecilistInfo
ID
DrSpecilistBan
DrSpecilistEng

DrGroupInfo
Code
DrGroupEng
DrGroupBan

HolidayInfo
DrCode
DrName
Holiday

ChamberTime_Info
ORG_Code
DrCode
DrName
ChamberDate
ChamberStartTime
ChamberStartHH
ChamberStartMM
ChamberFinishTime
ChamberFinishHH
ChamberFinishMM
UserName
Valid

ChamberCloseInfo
DrCode
DrName
FromDate
ToDate
EntryDate
userName

SerialNoMaintanance
RctNoOfOutdoorPatient
IDNoOfOutdoorPatient
RctNoOfAdmittedPatient
IDNoOfAdmittedPatient
RctNoOfEmergencyPatient
IDNoOfEmergencyPatient
RctNoOfAdvDueAmt
VisitingConsultantDrSlNo
PatientBedChargeSlNo
SlipNoOfDiagnosisInv
SaleNoOfPharSales
IDNoOfNonRegisterPatient
GRNSlNoofFinish
MTNSlNoofFinish
PNo
PharCompanyCodeSlNo
RctNoOfDrPayment
SlNoOfPharStockIn
SlNoAdvPayOfPatient
SlNoOfClinicalDue
SlNoOfPatientRelease
AfterReleasePatientSlNo
VatForHospital
AdminCharge
ServiceCharge
SlNoOfPharReturn

InvestigationChart
DeptName
PCode
ShortDesc
LongDesc
Charge
Discount
DiscountStatus
NormalValue
LessAmount
RptType
UserName
Category
GroupId
GroupName
DeliveryDuration
SubDeptName
SubSubDeptName
VaqGroup
VaqName
VaqPrice
VaqVat
VaqComm
VaqStatus

HistoryInfo *
Code
DescEng
DescBang

ObservationInfo *
Code
DescEng
DescBang
ResultEng
ResultBang

Fig: 4.4 Database Diagram III

4.5 System Development

System development we are used the tools visual studio dot net 2008, C # language, ASP dot net language and data store we are database used to Microsoft SQL server database 2014.

4.5.1 Introducing Visual Studio .NET 2008

Visual Studio .NET is a complete set of development tools for building ASP Web applications, XML Web services, desktop applications, and mobile applications. Visual Basic .NET, Visual C++ .NET, Visual C# .NET, and Visual J# .NET all use the same integrated development environment (IDE), which allows them to share tools and facilitates in the creation of mixed-language solutions. In addition, these languages leverage the functionality of the .NET Framework, which provides access to key technologies that simplify the development of ASP Web applications and XML Web services.

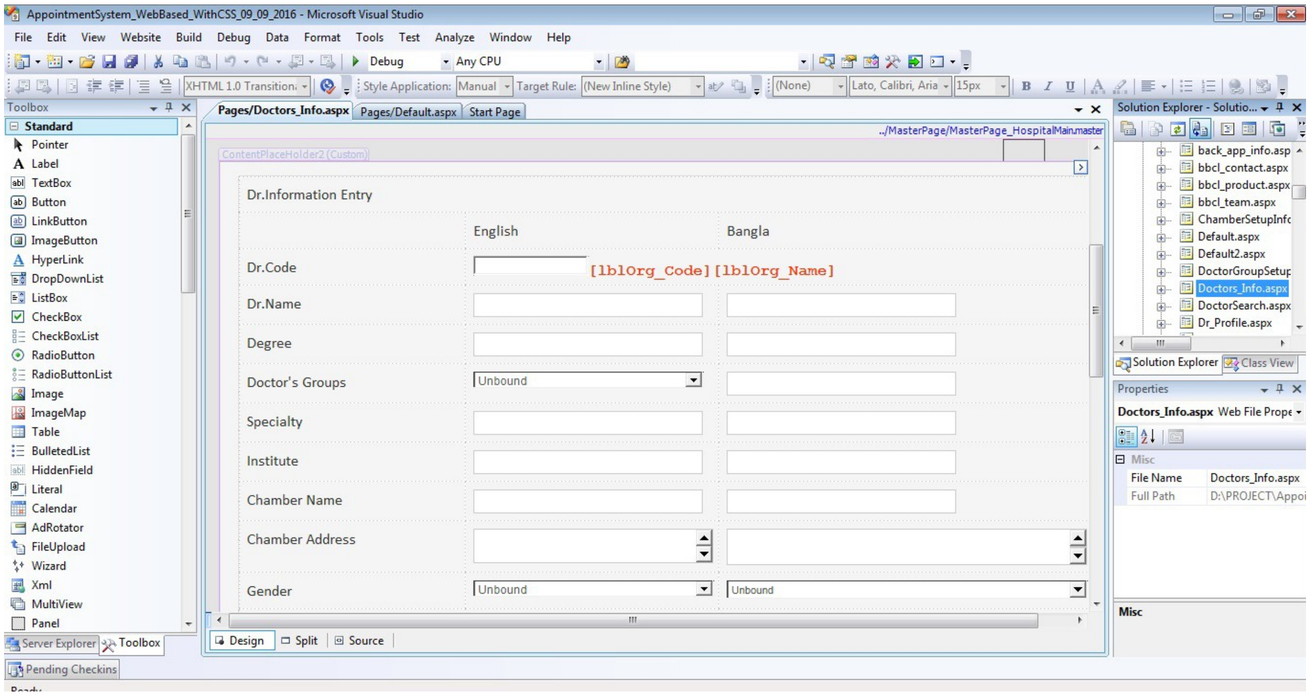


Fig: 4.5.1 Introducing Visual Studio .NET 2008

4.5.2 Introduction to the C# Language

C# is an elegant and type-safe object-oriented language that enables developers to build a variety of secure and robust applications that run on the .NET Framework. You can use C# to create Windows client applications, XML Web services, distributed components, client-server applications, database applications, and much, much more. Visual C# provides an advanced code editor, convenient user interface designers, integrated debugger, and many other tools to make it easier to develop applications based on the C# language and the .NET Framework.

C# syntax is highly expressive, yet it is also simple and easy to learn. The curly-brace syntax of C# will be instantly recognizable to anyone familiar with C, C++ or Java. Developers who know any of these languages are typically able to begin to work productively in C# within a very short time. C# syntax simplifies many of the complexities of C++ and provides powerful features such as nullable value types, enumerations, delegates, lambda expressions and direct memory access, which are not found in Java. C# supports generic methods and types, which provide increased type safety and performance, and iterators, which enable implementers of collection classes to define custom iteration behaviors that are simple to use by client code. Language-Integrated Query (LINQ) expressions make the strongly-typed query a first-class language construct.

As an object-oriented language, C# supports the concepts of encapsulation, inheritance, and polymorphism. All variables and methods, including the Main method, the application's entry point, are encapsulated within class definitions. A class may inherit directly from one parent class, but it may implement any number of interfaces. Methods that override virtual methods in a parent class require the **override** keyword as a way to avoid accidental redefinition. In C#, a struct is like a lightweight class; it is a stack-allocated type that can implement interfaces but does not support inheritance.

In addition to these basic object-oriented principles, C# makes it easy to develop software components through several innovative language constructs, including the following:

- Encapsulated method signatures called *delegates*, which enable type-safe event notifications.
- Properties, which serve as accessors for private member variables.
- Attributes, which provide declarative metadata about types at run time.
- Inline XML documentation comments.
- Language-Integrated Query (LINQ) which provides built-in query capabilities across a variety of data sources.

If you have to interact with other Windows software such as COM objects or native Win32 DLLs, you can do this in C# through a process called "Interop." Interop enables C# programs to do almost anything that a native C++ application can do. C# even supports pointers and the concept of "unsafe" code for those cases in which direct memory access is absolutely critical.

The C# build process is simple compared to C and C++ and more flexible than in Java. There are no separate header files, and no requirement that methods and types be declared in a particular order. A C# source file may define any number of classes, structs, interfaces, and events.

The following are additional C# resources:

- For a good general introduction to the language, see Chapter 1 of the C# Language Specification.
- For detailed information about specific aspects of the C# language, see the C# Reference.
- For more information about LINQ, see LINQ (Language-Integrated Query).
- To find the latest articles and resources from the Visual C# team, see the Visual C# Developer Center.

.NET Framework Platform Architecture

C# programs run on the .NET Framework, an integral component of Windows that includes a virtual execution system called the common language runtime (CLR) and a unified set of class libraries. The CLR is the commercial implementation by Microsoft of the common language infrastructure (CLI), an international standard that is the basis for creating execution and development environments in which languages and libraries work together seamlessly.

Source code written in C# is compiled into an intermediate language (IL) that conforms to the CLI specification. The IL code and resources, such as bitmaps and strings, are stored on disk in an executable file called an assembly, typically with an extension of .exe or .dll. An assembly contains a manifest that provides information about the assembly's types, version, culture, and security requirements.
When the C# program is executed, the assembly is loaded into the CLR, which might take various actions based on the information in the manifest. Then, if the security requirements are met, the CLR performs just in time (JIT) compilation to convert the IL code to native machine instructions. The CLR also provides other services related to automatic garbage collection, exception handling, and resource management. Code that is executed by the CLR is sometimes referred to as "managed code," in contrast to "unmanaged code" which is compiled into native machine language that targets a specific system. The following diagram illustrates the compile-time and run-time relationships of C# source code files, the .NET Framework class libraries, assemblies, and the CLR.

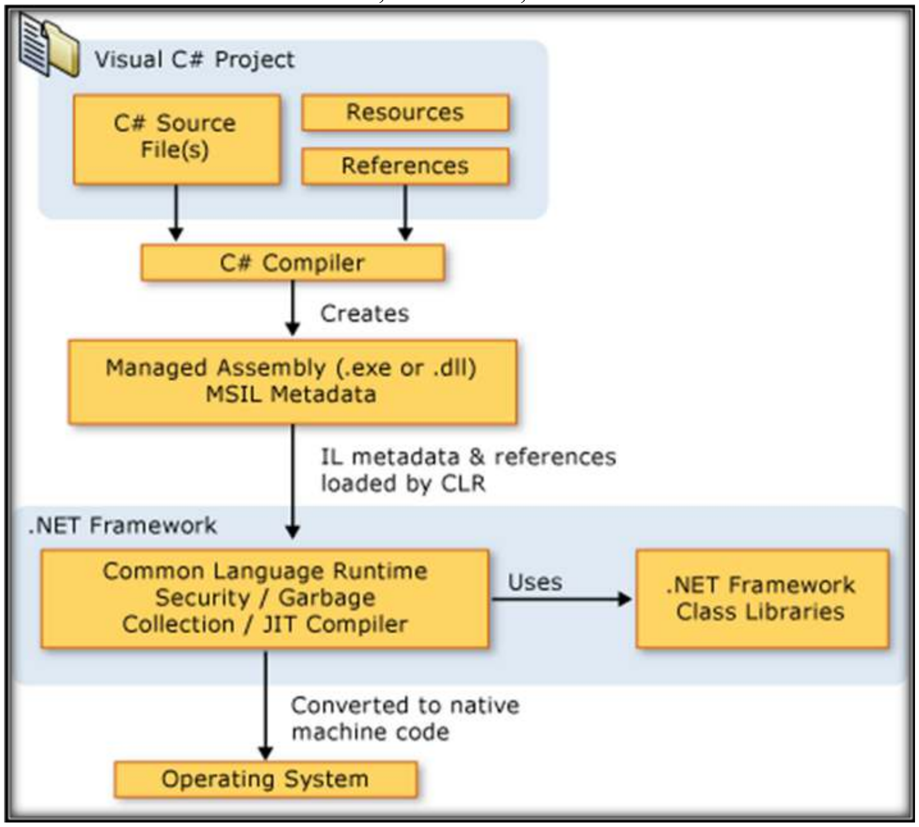

Visual C# Project
C# Source File(s)
Resources
References
C# Compiler
Creates
Managed Assembly (.exe or .dll) MSIL Metadata
IL metadata & references loaded by CLR
.NET Framework
Common Language Runtime Security / Garbage Collection / JIT Compiler
Uses
.NET Framework Class Libraries
Converted to native machine code
Operating System

Fig: 4.5.2 Introduction to the C# Language

Language interoperability is a key feature of the .NET Framework. Because the IL code produced by the C# compiler conforms to the Common Type Specification (CTS), IL code generated from C# can interact with code that was generated from the .NET versions of Visual Basic, Visual C++, or any of more than 20 other CTS-compliant languages. A single assembly may contain multiple modules written in different .NET languages, and the types can reference each other just as if they were written in the same language.
In addition to the run time services, the .NET Framework also includes an extensive library of over 4000 classes organized into namespaces that provide a wide variety of useful functionality for everything from file input and output to string manipulation to XML parsing, to Windows Forms controls. The typical C# application uses the .NET Framework class library extensively to handle common "plumbing" chores.

4.5.3 Introduction to ASP.NET

ASP.NET is a web development platform, which provides a programming model, a comprehensive software infrastructure and various services required to build up robust web applications for PC, as well as mobile devices.

ASP.NET works on top of the HTTP protocol, and uses the HTTP commands and policies to set a browser-to-server bilateral communication and cooperation.

ASP.NET is a part of Microsoft .Net platform. ASP.NET applications are compiled codes, written using the extensible and reusable components or objects present in .Net framework. These codes can use the entire hierarchy of classes in .Net framework.

The ASP.NET application codes can be written in any of the following languages:

- C#
- Visual Basic.Net
- Jscript
- J#

ASP.NET is used to produce interactive, data-driven web applications over the internet. It consists of a large number of controls such as text boxes, buttons, and labels for assembling, configuring, and manipulating code to create HTML pages.

4.5.4 Introduction to SQL Server Management Studio

SQL Server Management Studio (SSMS) is the main administration console for SQL Server.

SSMS enables you to create database objects (such as databases, tables, stored procedures, views etc), view the data within your database, configure user accounts, perform backups, replication, transfer data between databases, and more.

SQL Server Management Studio is a graphical user interface; so many tasks are "point and click". It is also the interface that enables you to run SQL scripts, so there are also tasks that require programming/scripting. However, many tasks can be performed either via GUI or SQL script, so it's your choice which one you use. For example, you can create a database using the GUI or by running a SQL script. Having said that, you still need the GUI in order to run the script.

Connect to Server

When you open up SQL Server Management Studio, you will be prompted to connect to SQL Server with a log in screen that looks like the following screenshot. You can either keep the default authentication settings or change them.

Fig: 4.5.4 Connect to Server

Here's what SQL Server Management Studio looks like once you've connected (and opened a new query):

Query Interface

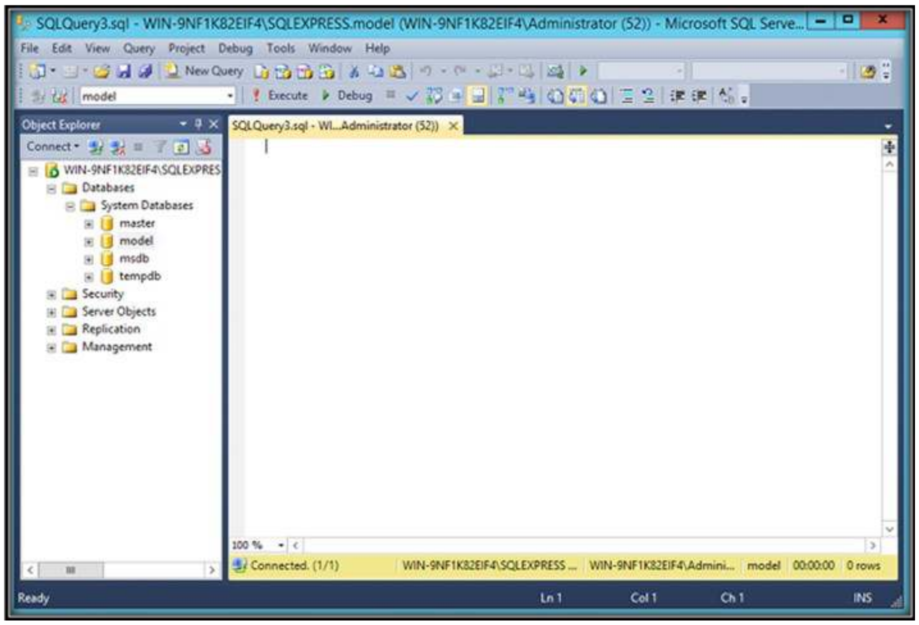

SQLQuery3.sql - WIN-9NF1K82EIF4\SQLEXPRESS.model (WIN-9NF1K82EIF4\Administrator (52)) - Microsoft SQL Serve...
File Edit View Query Project Debug Tools Window Help
New Query
model
Execute
Debug
Object Explorer
Connect
WIN-9NF1K82EIF4\SQLEXPRES
Databases
System Databases
master
model
msdb
tempdb
Security
Server Objects
Replication
Management
SQLQuery3.sql - WI...Administrator (52))
100 %
Connected. (1/1)
WIN-9NF1K82EIF4\SQLEXPRESS ...
WIN-9NF1K82EIF4\Admini...
model
00:00:00
0 rows
Ready
Ln 1
Col 1
Ch 1
INS

Fig: 4.5.4 Query Interface

The left pane contains the Object Explorer. The Object Explorer provides navigation to databases, server objects (such as triggers), log files, and more.

The right pane will change depending on what task you're performing. For example if you're modifying a table, you might see the table design and properties in the right pane. In this screenshot I have opened a blank query by clicking the New Query button. Many database tasks can be performed either via this window (i.e. progamatically), or via a GUI equivalent (i.e. "point and click").

You can use SQL Server Management Studio to create as many databases as you like. You can also connect to as many databases on as many servers as you like. These all appear in the Object Explorer. So you could run a query on your development environment, then switch to your test or production environment and run a query there. Because of this, you need to be careful that you don't accidentally run a script against the wrong server.

Most of the tasks performed with SQL Server Management Studio are initiated either from the top menu, or by right-clicking on an icon/object.

CHAPTER 5
RESULT

5.5.1 System Interfaces
All System interfaces in this chapter were created in ASP dot net and C# coder while creating ASP and Visual Studio Dot Net 2008 Integrated Development Environment (IDE) interfaces.

5.5.2 Login Form for the Different Users

Only authorized user with the right user name and password has right to access the services to particular department as like Patients, Doctors and Hospital he or she intern to view. When wrong user name and password is used the System rejects access to the services.

5.5.3 System Administration Home Page

The system administrator can add, edit system user and has access to view the services offered by the different for easer tracking in cases of mismanagement in the Hospital.

5.5.4 Patient Appointment Home Page

This page is patient can access the appointment and view the doctor available time and can known the time to views the doctor actual time. Then the system is patient not a more time queue.

5.5.5 Doctor Registration Page

This page is Doctor Registration to the system. Doctor Name, Designation, Department, Available Time, Chamber Time and more information registration this pages.

CHAPTER 6
CONCLUSION

6.0 Introduction

This Chapter describes discuss the objectives of the system stipulated in earlier chapters, limitation of the system conclusion and recommendation of the System.

6.1 Summary

As discussed in the previous chapters the main problem that we addressed was dealing with patient, doctor and hospital document. It is the above situation that above us to techniques of developing this Online Doctor Appointment System to be used the Patient, Doctor and Hospital to enable them to handle details on policies efficiently and effectively. The Project has implemented most of the objectives stipulated in earlier chapter. The Online Doctor Appointment System offers a number of benefits to the user and can capture data, store, view, add and delete the records entered the data cal also be posted information to the database.

Problems Encountered during Data Collection: sensitive information released to us, few projects and books written about Online Doctor Appointment System.

Problem Encountered during System Design: Limited time to finish up to work, but we are finally finished to our system design the projects.

6.2 Limitations:

This section describes those services that are not provided by the system and those included the following. It is always connect to the Internet connection. It Support to Microsoft Windows Platform and not support to others platform as like MAC or Linux.

6.3 Conclusions

The core reason for the establishment of computerizing Online Doctor Appointment System is to enable the hospital administrators in a convenient, fair and timely manner. Therefore the IT used should support the core objective of the system if it is to remain relevant to the hospital. A lot still needs to be done in the IT department in order to make available technology effective. This may involve training of the hospital staffs on how to enter data in the right and relevant data in the system and the management to keep updating the hardware and software requirements of the system. IT and computer systems need to be kept being upgraded as more and more IT facilities software are introduced in to days IT market. The researcher acknowledges the fact this system does not handle all patient doctor and hospital. The researcher therefore suggests that for further research, the following can be researched on. The most cost effective way of handling all Hospital Patient management system process.

6.4 Recommendations

Thanking of all the members of hospital staff in the hospital to get accustomed to the system will be a priority. This being a new system the computerized Online Doctor Appointment System will replace their jobs.

REFERENCES

CHAPTER 7
APPENDIX I

WORLD UNIVERSITY
O F B A N G L A D E S H

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Questionnaire

The purpose of the questionnaire is to identify and specify functional requirements of our proposed Online Doctor Appointment System (ODAS) to be used by Patient, Doctor & Hospital.

By

GROUP No. CSE 28 (B) D

MAY, 2016

APPENDIX II
The Organizational Structure

This section describes the flow of powers of delegation in the hospital, patient & doctor.

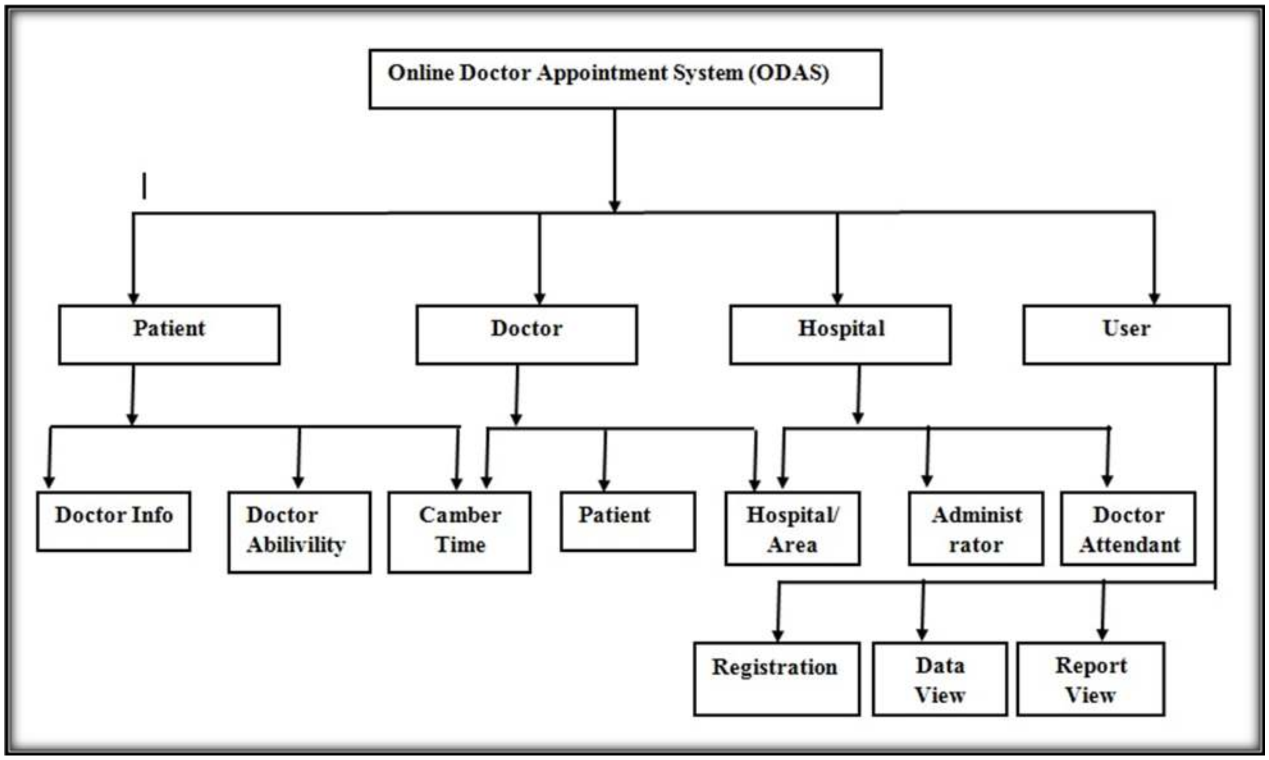


Fig: The Organizational Structure

APPENDIX III

Source Code of the System

Login Code

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage/main_master_page.master"
AutoEventWireup="true" CodeFile="login_page.aspx.cs"
Inherits="Pages_login_page" Title="LOGIN" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
    <div class="login_page">
            <div class="container">
                  <div class="login_panel col-md-4 col-sm-4">
                        <div class="login_header">
                              <div class="login">
                                    <div class="table_top_login"
colspan="4"><span>LOG IN</span></div>
                              </div>
                        </div>
                        <div class="login_area row">
                              <div class="form-group">
                                    <div class="col-md-4 col-sm-4 login_col"
>User Name</div>
                                    <div class="col-md-8 col-sm-8 login_col" >
                                          <asp:TextBox placeholder="User Name"
ID="txtUserName" class="form-control" runat="server"
required></asp:TextBox></div>
                              </div>
                        </div>
                        <div class="login_area row">
```

```
                                <div class="form-group" >
                                      <div class="col-md-4 col-sm-4 
login_col">Password</div>
                                      <div class="col-md-8 col-sm-8 login_col">
                                            <asp:TextBox placeholder="Password" 
ID="txtPassWord" class="form-control" runat="server" TextMode="Password" 
required>
                                        </asp:TextBox>
                                    </div>
                                </div>
                          </div>
                          <div class="login_btn">
                                <div>
                                      <asp:Button ID="BtnLogin" class="btn btn-
md btn-success login_button" 
                              runat="server" Text ="LogIn" 
onclick="BtnLogin_Click1" >
                                      </asp:Button>
                                </div>
                          </div>
                          <div class="regi_frorm">
                              <a 
href="register_appoinment.aspx">Registration</a>

                          </div>
                    </div>
              </div>
        </div>

</asp:Content>
```

C Sharp Code

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

public partial class Pages_login_page : System.Web.UI.Page
{
    BasicDM cn = new BasicDM();
    DataSet ds;
    String lcStrSql, lcType;
    int i;
    protected void Page_Load(object sender, EventArgs e)
    {
        try
        {
            //txtUserName.Text = "UserName";
            //txtPassWord.Text = "Password";
            Session["Org_Code"] = Request.QueryString["PicId"].ToString();


        }
        catch
        {

            Session["Org_Code"] = "";
        }
    }
    protected void BtnLogin_Click(object sender, EventArgs e)
    {
        //aply sql injection like 1 or 1='1' ---

        ds = new DataSet();
        lcStrSql = "Select * From Password Where UserName='" +
cn.ProperCase(txtUserName.Text) + "' AND Password='" +
cn.ProperCase(txtPassWord.Text) + "'";
        ds = cn.Search(lcStrSql);
        if (ds.Tables[0].Rows.Count > 0)
        {
            Session["UserName"] = txtUserName.Text;
            lcType = ds.Tables[0].Rows[0]["Type"].ToString();
            Session["Type"] = lcType;
            Session["SessionID"] = GenerateSessionID("1");

            switch (lcType)
            {
                case "Organization":
                    Session["Org_Code"] = ds.Tables[0].Rows[0]
["Org_Code"].ToString();
                    Session["Org_Name"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Org_Name");
                    Session["Org_Address"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Address");

                    Response.Redirect("Index_HospitalMain.aspx", false);

                    break;
                case "Doctor":

                    Session["Org_Code"] = ds.Tables[0].Rows[0]
["Org_Code"].ToString();
                    Session["Org_Name"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Org_Name");
                    Session["Org_Address"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Address");
```

```
                    Response.Redirect("Index_HospitalMain.aspx", false);

                    break;

                case "User":

                    Session["Org_Code"] = ds.Tables[0].Rows[0]
["Org_Code"].ToString();
                    Session["Org_Name"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Org_Name");
                    Session["Org_Address"] =
cn.ReturnFiledValue("Organization_Info", "Org_Code='" +
Session["Org_Code"].ToString() + "'", "Address");
                    Response.Redirect("Index_HospitalMain.aspx", false);

                    break;

                default:
                    break;
            }

            //Response.Redirect("")
        }
        else
        {
            Response.Write("<script language=javascript>alert('Invalid
UserName/Password');</script>");
            //Response.Redirect("Index.aspx",false);
        }

    }
```

Patient Appointment

ASP Code

```
<%@ Page Language="C#"
MasterPageFile="~/MasterPage/MasterPage_HospitalMain.master"
AutoEventWireup="true" CodeFile="Appoinment.aspx.cs"
Inherits="Pages_Appoinment" Title="Appoinment List" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder2"
Runat="Server">
<div class="row">
      <div class="col-md-10 col-md-offset-1 backend_app">
            <fieldset>
              <!-- Form Name -->
              <legend class="dr_app_head">Doctor Appointment
Information</legend>
              <!-- Text input-->
             <div class="form-group">
                  <div class="col-sm-6 control-label">
                        <label class="col-sm-4 " for="textinput">Dr
Code</label>

                        <div class="col-sm-8">
```

```
                              <asp:TextBox ID="txtDrCode" runat="server"
placeholder="Dr Code" class="form-control" AutoPostBack="True"
                        ontextchanged="txtDrCode_TextChanged"></asp:TextBox>
                        </div>
                  </div>

                  <div class="col-sm-6 control-label">
                        <label class="col-sm-4 " for="textinput">Dr
Name</label>
                        <div class="col-sm-8">
                              <asp:TextBox ID="txtDrName" runat="server"
placeholder="Dr Name" class="form-control"></asp:TextBox>
                        </div>
                  </div>
            </div>

            <div class="form-group">
                  <div class="col-sm-6 control-label">
                        <label class="col-sm-4 " for="textinput">Date
From</label>
                        <div class="col-sm-8">
                              <asp:TextBox ID="dtDateFrom" class="form-control
datepicker" runat="server" onclick="ds_sh(this);"></asp:TextBox>
                        </div>
                  </div>

                  <div class="col-sm-6 control-label">
                        <label class="col-sm-4 " for="textinput">Date
To</label>
                        <div class="col-sm-8">
                          <asp:TextBox ID="dtDateTo" class="form-control
datepicker" runat="server" onclick="ds_sh(this);"></asp:TextBox>
                        </div>
                  </div>
            </div>

              <div class="form-group">
                  <div class="col-sm-offset-2 col-sm-6">
                    <div class="pull-right">
                        <asp:Button ID="BtnShow" runat="server"
onclick="BtnShow_Click" class="btn btn-primary" Text="Show" />
                    </div>
                  </div>
              </div>
            </fieldset>
            <div class= "container_Appointment">
            <%=getAppoinmentList() %>
        </div>
            <table cellpadding="0" cellspacing="0" border="0"  style="margin-
top:5px">
                  <tr>
                        <td class="leave_app">Weekly Holiday</td>
                        <td>&nbsp&nbsp&nbsp</td>
                        <td style="background-
color:#60BB46">&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp</td>
                        <td class="leave_app">Leave</td>
                        <td>&nbsp&nbsp&nbsp</td>
                        <td style="background-
color:#F28085">&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp&nbsp</td>
                  </tr>
             </table>
      </div><!-- /.col-lg-12 -->
</div>
</asp:Content>
```

C#Code

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
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.Xml.Linq;
using CrystalDecisions.CrystalReports.Engine;

public partial class Pages_Appoinment : System.Web.UI.Page
{

    BasicDM cn = new BasicDM();
    int i;
    DataSet ds = new DataSet();
    String lcStrSql, lcTitle, rptPath, lcTrSlNo;
    DataRow dr;
    DataTable dt;

    ReportDocument rptdoc = new ReportDocument();
    ReportModule rm = new ReportModule();
    DS_Preview DSRP;
    protected void Page_Load(object sender, EventArgs e)
    {
        try
        {
            if (Session["SessionID"] == null)
            {
                Response.Redirect("Index.aspx", true);
            }
            if (IsPostBack == false)
            {
                txtDrCode.Text = Session["UserName"].ToString();
                dtDateFrom.Text = cn.ConvertDateToSQLFormat(DateTime.Now);
                dtDateTo.Text = cn.ConvertDateToSQLFormat(DateTime.Now);


                txtDrName.Text = cn.ReturnFiledValue("SignatureDrInfo",
"SignaturedrCode='" + cn.ProperCase(txtDrCode.Text) + "'", "SignatureDrName");
                txtDrName.Enabled = false;
                txtDrCode.Enabled = false;

                if (Request.QueryString["PicId"] !=null)
                {
                    lcTrSlNo = Request.QueryString["PicId"].ToString();
                    fncPrintPreview(lcTrSlNo);
                }
            }

        }
        catch (Exception ex)
        {
            Response.Write(ex.Message);
        }
    }
    protected void BtnShow_Click(object sender, EventArgs e)
    {
        getAppoinmentList();
    }
    public string getAppoinmentList()
    {

        int lnTotalCol,j;
        String
htmlStr,lcAppoinment,lcDayName,lcBgColor,lc2ndBgColor,lcTrSlNo;
        htmlStr = "";
        i = 1;
        DateTime d1, d2,d3,d4;
        d1 = Convert.ToDateTime(dtDateFrom.Text);
        d2 = Convert.ToDateTime(dtDateTo.Text);
        lnTotalCol = Convert.ToInt32((d2 - d1).TotalDays)+1;



        htmlStr += "<table valign=top  cellspacing=0 cellpadding=0  border=0
style=background-color:#94CDE9;border-color:Blue; >";
```

```
        ds = new DataSet();

        for (j = 0; j <= 1; j++)
        {
            htmlStr += "<tr>";

            for (i = 1; i <= lnTotalCol; i++)
            {

                lcBgColor = "#F28085";
                lc2ndBgColor = "";
                        d3 = d1.AddDays(i - 1);
                       // lcDayNo =d3.Date.Day.ToString();
                        lcDayName= GetDayName(d3.DayOfWeek.ToString());
                        if (cn.fncSeekRecordNew("HolidayInfo", "DrCode='" +
txtDrCode.Text + "' AND Holiday='" + lcDayName + "'") == true)
                        {
                            lcBgColor = "Green";
                            lc2ndBgColor = "#60BB46";
                        }

                        if (j == 0)
                        {
                            htmlStr += "<td  align=left class=table_top
width=200px  height=20px >" + cn.GetCustomDateFormat(d3) + "</td>";
                        }
                        else
                        {
                            if (cn.fncSeekRecordNew("ChamberCloseInfo",
"FromDate = '" + cn.ConvertDateToSQLFormat(d3) + "' AND DrCode = '" +
txtDrCode.Text + "'") == true)
                            {
                                //htmlStr += "<td style=background-
color:#7B98E1 width=180px  height=20px>" + "Holy Day" + "</td>";
                                //htmlStr += "<td style=background-
color:#FF0000 valign=top width=180px  height=20px>" + " " + "";
                                htmlStr += "<td style=background-color:" +
lcBgColor + "  valign=top width=250px  height=20px >" + " " + "";
                                lcStrSql = "SELECT (ReqFromMOB + '  ' +
RIGHT('000000'+CONVERT(nvarchar(20), Slno),4)+' ' + PtName+' '+AppTime) AS
Appointment,Complete,TrSlNo ";
                                lcStrSql += "FROM Appointment ";
                                lcStrSql += "WHERE reqDate ='" +
cn.GetCustomDateFormat(d3) + "' AND DrCode ='" + txtDrCode.Text + "' AND
Valid=1 AND slNo <> 0 Order By SlNo";
                                ds = cn.Search(lcStrSql);
                                if (ds.Tables[0].Rows.Count > 0)
                                {
                                    foreach (DataRow dr in ds.Tables[0].Rows)
                                    {
                                        htmlStr += "<table valign=top
cellspacing=4 cellpadding=0 width:200px border=1 style=background-
color:White;border-color:White;>";
                                        lcAppoinment =dr[0].ToString();
//ds.Tables[0].Rows[i]["Appointment"].ToString();
```

```
                                        lcTrSlNo = dr[2].ToString();
//ds.Tables[0].Rows[i]["TrSlNo"].ToString();
                                        htmlStr += "<td class=table_td_Dr
style= font-size:11px width:200px><a style=text-decoration:None;padding:0px;
href=" + "Appoinment.aspx" + "?PicId=" + lcTrSlNo + ">" + lcAppoinment +
"</a></td>";

                                        htmlStr += "</table>";
                                    }
                                    htmlStr += "</td>";
                                }

                            }
                            else
                            {
                                //htmlStr += "<td valign=top style=width:180px
height:20px  valign=top background-color:" + lcBgColor + " >" + "-";
                                htmlStr += "<td style=background-color:" +
lc2ndBgColor + "  valign=top width=180px  height=20px>" + " " + "";

                                lcStrSql = "SELECT (ReqFromMOB + '  ' +
RIGHT('000000'+CONVERT(nvarchar(20), Slno),4)+' ' + PtName+' '+AppTime) AS
Appointment,Complete,TrSlNo ";
                                lcStrSql += "FROM Appointment ";
                                lcStrSql += "WHERE reqDate ='" +
cn.GetCustomDateFormat(d3) + "' AND DrCode ='" + txtDrCode.Text + "' AND
Valid=1 AND SlNo <> 0 Order By SlNo";
                                ds = cn.Search(lcStrSql);
                                if (ds.Tables[0].Rows.Count > 0)
                                {
                                    //htmlStr += "<table valign=top
cellspacing=4 cellpadding=0 class=shadowBG border=3>";
                                    foreach (DataRow dr in ds.Tables[0].Rows)
                                    {
                                        htmlStr += "<table valign=top
cellspacing=4 cellpadding=0 width:200px border=1 style=background-
color:White>";
                                        lcAppoinment = dr[0].ToString();
//ds.Tables[0].Rows[i]["Appointment"].ToString();
                                        lcTrSlNo = dr[2].ToString();
//ds.Tables[0].Rows[i]["TrSlNo"].ToString();
                                        //htmlStr += "<tr><td
class=table_td_Dr style= font-size:11px width:200px>" + lcAppoinment +
"</td></tr>";
                                        htmlStr += "<td class=table_td_Dr
style= font-size:11px width:200px><a style=text-decoration:None;padding:0px;
href=" + "Appoinment.aspx" + "?PicId=" + lcTrSlNo + ">" + lcAppoinment +
"</a></td>";

                                        htmlStr += "</table>";
                                    }
                                    // htmlStr += "</table>";
                                    htmlStr += "</td>";
                                }
                            }
                        }
```

```
                    }

                      htmlStr += "</tr>";
                  }
                    htmlStr += "</table>";


        return htmlStr;
}


    protected void txtDrCode_TextChanged(object sender, EventArgs e)
    {
        try
        {
            txtDrName.Text = cn.ReturnFiledValue("SignatureDrInfo",
"SignatureDrCode='" + txtDrCode.Text + "'", "SignatureDrName");
        }
        catch (Exception ex)
        {
            Response.Write(ex.Message);
        }
    }

    public string GetDayName(String lcDayNo)
    {
        String lcDayName;
        switch (lcDayNo)
        {
            case "1":
                lcDayNo = "Saturday";
                break;

            case "2":
                lcDayNo = "Sunday";
                break;

            case "3":
                lcDayNo = "Monday";
                break;
            case "4":
                lcDayNo = "TuesDay";
                break;
            case "5":
                lcDayNo = "Wednesday";
                break;
            case "6":
                lcDayNo = "Thursday";
                break;
            case "7":
                lcDayNo = "Friday";
                break;
            default:
                break;
        }
        return lcDayNo;
    }

    private void fncPrintPreview(String lcSlipNo)
    {
        try
        {

            String lcPatientName, lcAge, lcSex, LcMobileNo, lcDrCode, 
lcDrDetailsENG, lcDrDetailsBan;
            lcPatientName = "";
            lcAge = "";
            lcSex = "";
            lcDrCode = "";
            LcMobileNo = "";

            ds = new DataSet();
            lcStrSql = "Select * From Appointment Where TrSlNo='" + lcSlipNo + 
"'";
            ds = cn.Search(lcStrSql);
            if (ds.Tables[0].Rows.Count > 0)
            {
                lcPatientName = ds.Tables[0].Rows[0]["PtName"].ToString();
                lcAge = ds.Tables[0].Rows[0]["Age"].ToString();
                lcSex = ds.Tables[0].Rows[0]["Sex"].ToString();
                LcMobileNo = ds.Tables[0].Rows[0]["ReqFromMob"].ToString();
                lcDrCode = ds.Tables[0].Rows[0]["DrCode"].ToString();
            }

            rm = new ReportModule();
            DSRP = new DS_Preview();
            rptdoc = new ReportDocument();
            //dt=(DataTable)Session["GV1"];

            lcTitle = "";

            dt = new DataTable("DT_Prescription");


            dt.Columns.Add("DrCode");
            dt.Columns.Add("DrNameENG");
            dt.Columns.Add("DesignationENG");
            dt.Columns.Add("SpecialityENG");
            dt.Columns.Add("ChamberENG");
            dt.Columns.Add("DegreeENG");
            dt.Columns.Add("ChamberAddressENG");
            dt.Columns.Add("InstituteEN");
            dt.Columns.Add("DrNameBAN");
            dt.Columns.Add("DegreeBAN");
            dt.Columns.Add("SpecialityBAN");
            dt.Columns.Add("InstituteBAN");

            dt.Columns.Add("ChamberNameBAN");
            dt.Columns.Add("ChamberAddressBAN");
            dt.Columns.Add("ChamberTimeBAN");
            dt.Columns.Add("PTName");
            dt.Columns.Add("Age");
            dt.Columns.Add("Sex");
            dt.Columns.Add("MobileNo");
            dt.Columns.Add("DetailsENG");
            dt.Columns.Add("DetailsBAN");
            lcStrSql = "Select * From SignatureDrInfo Where Org_code='" +
Session["Org_Code"].ToString() + "' AND SignatureDrCode='" + lcDrCode + "'";
            ds = cn.Search(lcStrSql);
            if (ds.Tables[0].Rows.Count > 0)
            {

                dr = DSRP.Tables["DT_Prescription"].NewRow();
                dr["DrCode"] = lcDrCode;
                dr["DrNameENG"] = ds.Tables[0].Rows[0]
["SignatureDrName"].ToString();
                dr["DesignationENG"] = ds.Tables[0].Rows[0]
["Designation"].ToString();
                dr["SpecialityENG"] = ds.Tables[0].Rows[0]
["Specility"].ToString();
                dr["ChamberENG"] = ds.Tables[0].Rows[0]["Chamber"].ToString();
                dr["DegreeENG"] = ds.Tables[0].Rows[0]["Degree"].ToString();
                dr["ChamberAddressENG"] = ds.Tables[0].Rows[0]
["ChamberAddressEng"].ToString();

                dr["InstituteEN"] = ds.Tables[0].Rows[0]
["InstituteEN"].ToString();
                dr["DrNameBAN"] = ds.Tables[0].Rows[0]
["DrNameBAN"].ToString();
                dr["DegreeBAN"] = ds.Tables[0].Rows[0]
["DegreeBAN"].ToString();
                dr["SpecialityBAN"] = ds.Tables[0].Rows[0]
["SpecialityBAN"].ToString();
                dr["InstituteBAN"] = ds.Tables[0].Rows[0]
["InstituteBAN"].ToString();
                dr["ChamberNameBAN"] = ds.Tables[0].Rows[0]
["ChamberNameBAN"].ToString();
                dr["ChamberAddressBAN"] = ds.Tables[0].Rows[0]
["ChamberAddressBAN"].ToString();

                dr["ChamberTimeBAN"] = ds.Tables[0].Rows[0]
["ChamberTimeBAN"].ToString();
                dr["PTName"] = lcPatientName;
                dr["Age"] = lcAge;
                dr["Sex"] = lcSex;
                dr["MobileNo"] = LcMobileNo;
                lcDrDetailsENG = dr["DegreeENG"].ToString() +
System.Environment.NewLine + dr["SpecialityENG"].ToString() +
System.Environment.NewLine + dr["InstituteEN"].ToString() +
System.Environment.NewLine + dr["ChamberAddressENG"].ToString();
                lcDrDetailsBan = dr["DegreeBan"].ToString() +
System.Environment.NewLine + dr["SpecialityBAN"].ToString() +
System.Environment.NewLine + dr["InstituteBan"].ToString() +
System.Environment.NewLine + dr["ChamberAddressBan"].ToString();
```

```
                dr["DetailsEng"] = lcDrDetailsENG;
                dr["DetailsBAN"] = lcDrDetailsBan;
                // dt.Rows.Add(dr);
                DSRP.Tables["DT_Prescription"].Rows.Add(dr);

            }




            rptPath = Server.MapPath("~/Reports/Prescription_Print.rpt");
            rptdoc.Load(rptPath);

            rptdoc.SetDataSource(DSRP);
            Hashtable hParameter = new Hashtable();

            rptdoc.SetParameterValue(0, Session["Org_Name"].ToString());
            rptdoc.SetParameterValue(1, Session["Org_Address"].ToString());
            rptdoc.SetParameterValue(2, "");
            rptdoc.SetParameterValue(3, lcTitle);

            Session.Add(SessionInfo.REPORT_FILE, rptdoc);
            Session.Add(SessionInfo.REPORT_PARAM, hParameter);
            ClientScriptManager cScript = Page.ClientScript;
            Type cType = this.GetType();
            //Response.Redirect("Appoinment.aspx");
            string script = @"<script language =
javascript>window.open('FrmRptViewer.aspx')</script>";
            if (!cScript.IsClientScriptBlockRegistered(cType, "ViewReport"))
                cScript.RegisterClientScriptBlock(cType, "ViewReport",
script);


        }
        catch (Exception ex)
        {
            Response.Write(ex.Message);
        }
    }

}
```

SMS Sent Conformation Code

ASP. Net

```
<%@ Page Language="C#"
MasterPageFile="~/MasterPage/MasterPage_HospitalMain.master"
AutoEventWireup="true" CodeFile="frmDrLateSMS.aspx.cs"
Inherits="Pages_frmDrLateSMS" Title="Untitled Page" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
```

```
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder2"
Runat="Server">
<table style="width:600px" class="lefttd">
<tr>
<td style="width:150px">
    Mobile No:
</td>
<td>
    <asp:TextBox ID="txtMobileNo" runat="server"></asp:TextBox>
</td>
<td>
</td>
<td>
</td>
</tr>
<tr>
<td style="height: 18px">
Dr Code:
    </td>
<td style="height: 18px">
    <asp:TextBox ID="txtDrCode" runat="server"></asp:TextBox>
    </td>
<td style="height: 18px">
    </td>
<td style="height: 18px">
    </td>
</tr>
<tr>
<td>
    Message Details:</td>
<td>
    <asp:TextBox ID="txtSMSDetails" runat="server" TextMode="MultiLine"
        Width="250px"></asp:TextBox>
    </td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
</tr>
<tr>
<td>
    <asp:Button ID="BtnSave" runat="server" onclick="BtnSave_Click"
Text="Save"
        Width="150px" />
    </td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
</tr>
<tr>
<td>
    &nbsp;</td>
```

```
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
</tr>
<tr>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
<td>
    &nbsp;</td>
</tr>
</table>

</asp:Content>
```

C Sharp

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
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.Xml.Linq;

public partial class Pages_frmDrLateSMS : System.Web.UI.Page
{
    BasicDM cn = new BasicDM();
    String lcStrSql;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (IsPostBack == false)
        {
            if (Session["SessionID"] == null)
```

```
                {
                    Response.Redirect("Index.aspx", false);
                }
            }
        }
        protected void BtnSave_Click(object sender, EventArgs e)
        {
            try
            {

                lcStrSql = "Insert INto TableName (DrCode,MobileNo,SMS,UserName)
";
                lcStrSql += " Values('" + txtDrCode.Text + "','" +
txtMobileNo.Text + "','" + txtSMSDetails.Text + "','"+
Session["UserName"].ToString() +"')";
                cn.SDE(lcStrSql, "");
                Response.Write("<script language =javascript>alert('Successfully
Save..');</script>");


            }
            catch (Exception ex)
            {
                Response.Write(ex.Message);
            }

        }
    }
```