**HOSPITAL MANAGEMENT SYSTEM**

**A Project work submitted to the**

College Name:-

**Submitted by**

**Guided by**

**Teacher Name :-**

**ACKNOWLEDGEMENT**

**CONTENTS**

**S.No Contents**

1.

Introduction

 Synopsis

2.

Project Objective

3.

System Specification

 System Configuration

 Software Specification

4.

System Analysis

 Existing System

 Proposed System

5.

System Design

 Input Design

 Output Design

6.

Database Design

7.

Dataflow Diagram

8.

System Testing

9.

Sample Codings

10.

Screenshots

**INTRODUCTION**

**SYNOPSIS**

This project will automate the daily operations of LIFE LINE hospital.

The project keeps track of the staff and patient (in-patient, out-patient) details. It

also takes care of the ward, medical, invoice and the doctor’s appointment

details. The system generates the daily ward availability, the status of the

operation theatres and ICU.

HOSPITAL MANAGEMENT is an integrated Hospital Information

System, which addresses all the major functional areas of multi-specialty

hospitals. The HOSPITAL MANAGEMENT enables better patient care, patient

safety, patient confidentiality, efficiency, reduced costs and better management

information system. It provides easy access to critical information thus enabling

the management to take better decisions on time.

This project deals with processing of each and every department in the

hospital. This project sincerely aims to reduce the manual processing of each

department.

The Scope of the project takes care of the details of each and every

department. These details gives the doctor, staffs, specialists and patient details

including their salary, attendance , doctor’s appointments and the billing

system. The details of Doctor and staff help the hospital to maintain the record

of every person. Their attendance details help them to know about their attentive

presence while salary is calculated. The billing system provides an efficient way

for calculating bill details of the patients.

**PROJECT OBJECTIVE**

 To computerize all details regarding patient details & hospital details.

 To automate the process of ward entries.

 To maintain records effectively.

 To manage current status of staff and doctor availablity.

 The project has information regarding the inpatient details, outpatient details, Billing details and Ambulance details.

**This project includes modules such as**

1) Admission

 Inpatient

 Outpatient

2) Doctor Details

 Experience

 Personnel Details

 Attendance

 On Duty

 Fees

3) diagnosis

 Inpatient

 Outpatient

**MODULES REQUIRED**

***Admission : -***

This module records basic patient related information, which is collected

when the patient visits the hospital for the first time. Each patient is allocated

a unique patient identification numbers also known as Hospital No.

***Patient Details:***

It keeps track of all details about both in-patient and out-patient.

Patient id, patient name, address, admitted date, doctor name, room no are

entered in a form and stored for future reference. Also particular patient details

can be viewed in the table using a separate form with a attribute patient id.

***Outpatient:***

This module manages activities related to patient who visits the Hospital

Resident Doctor or Consultant Doctor for Medical Consultations, diagnosis and

treatment.

***Inpatient:***

Admission request will be made here. Request for admission is made

before patient admitting the hospital.

***Doctor Details:***

It keeps track of all details about doctors and staffs of the hospital. staffs,

Doctors, Nurses name, staff id, address, qualification, cell no, e-mail are entered and stored in a separate form. Individual staff details can be viewed in the table

using a separate form with a attribute Staff id

**SYSTEM SPECIFICATION**

**SYSTEM CONFIGURATION**

**HARDWARE REQUIREMENTS**

PROCESSOR :INTEL PENTIUM 4 (OR)HIGHER

RAM :512 MB & ABOVE

HARD DISK DRIVE : 500 MB FREE SPACE OR ABOVE

PRINTER : INK-JET PRINTER

PEN DRIVE : 512MB.

**SOFTWARE REQUIREMENTS**

PROGRAMMING LANGUAGE : .net (C#)

BACKEND :MS SQL SERVER-2019

FRONT END : C#

OPERATING SYSTEM : WINDOWS XP & HIGHER VERSION

**SOFTWARE SPECIFICATION**

**VISUAL STUDIO . NET**

Visual Studio .NET is a complete set of development tools for building

 ASP Web applications

 XML Web services

 desktop applications

 mobile applications

Visual Basic .NET, Visual C++ .NET, and Visual C# .NET all use the

same integrated development environment (IDE), which allows them to share

tools and facilitates in the creation of mixed-language solutions.

* **Desktop applications**

## What is C#?

C# is pronounced "C-Sharp".

It is an object-oriented programming language created by Microsoft that runs on the .NET Framework.

C# has roots from the C family, and the language is close to other popular languages like [C++](https://www.w3schools.com/cpp/default.asp) and [Java](https://www.w3schools.com/java/default.asp).

The first version was released in year 2002. The latest version, ****C# 8****, was released in September 2019.

C# is used for:

* Mobile applications
* Desktop applications
* Web applications
* Web services
* Web sites
* Games
* VR
* Database applications
* And much, much more!

## Why Use C#?

* It is one of the most popular programming language in the world
* It is easy to learn and simple to use
* It has a huge community support
* C# is an object oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs.
* As C# is close to C, [C++](https://www.w3schools.com/cpp/default.asp) and [Java](https://www.w3schools.com/java/default.asp), it makes it easy for programmers to switch to C# or vice versa

C# could theoretically be compiled to machine code, but in real life, it's always used in combination with the .NET framework. Therefore, applications written in C#, requires the .NET framework to be installed on the computer running the application. While the .NET framework makes it possible to use a wide range of languages, C# is sometimes referred to as THE .NET language, perhaps because it was designed together with the framework.

C# is an Object Oriented language and does not offer global variables or functions. Everything is wrapped in classes, even simple types like int and string, which inherit from the System.Object class.

C# is a modern, general-purpose programming language that can be used to perform a wide range of tasks and objectives that span over a variety of professions. C# is primarily used on the Windows .NET framework, although it can be applied to an open source platform. This highly versatile programming language is an object-oriented programming language (OOP)—which isn’t very common—and fairly new to the game, yet already a reliable [crowd pleaser](https://pypl.github.io/PYPL.html).

When compared to long-standing languages like Python and PHP, C# is a young addition to the programming family at nearly twenty years old. The language was developed in the year 2000 by Microsoft’s Anders Hejlsberg, a Danish software engineer with a history for popular creations. Anders has taken part in the creation of a handful of dependable programming tools and languages, including Microsoft’s TypeScript and Delphi, a suitable replacement for Turbo Pascal.

* **Windows Forms :-**

## **Introduction**

Windows Forms is a UI framework for building Windows desktop apps. It provides one of the most productive ways to create desktop apps based on the visual designer provided in Visual Studio. Functionality such as drag-and-drop placement of visual controls makes it easy to build desktop apps.

With Windows Forms, you develop graphically rich apps that are easy to deploy, update, and work while offline or while connected to the internet. Windows Forms apps can access the local hardware and file system of the computer where the app is running.

To learn how to create a Windows Forms app, see [Tutorial: Create a new WinForms app (Windows Forms .NET)](https://docs.microsoft.com/en-us/dotnet/desktop/winforms/get-started/create-app-visual-studio?view=netdesktop-5.0).

Windows Forms is the new platform for Microsoft Windows application

development, based on the .NET Framework. This framework provides a clear,

object-oriented, extensible set of classes that enables you to develop rich

Windows applications. Additionally, Windows Forms can act as the local user

interface in a multi-tier distributed solution

The Windows Forms development platform supports a broad set of app development features, including controls, graphics, data binding, and user input. Windows Forms features a drag-and-drop visual designer in Visual Studio to easily create Windows Forms apps.

There are two implementations of Windows Forms:

1. The open-source implementation hosted on [GitHub](https://github.com/dotnet/winforms).
2. This version runs on .NET 5 and .NET Core 3.1. The Windows Forms Visual Designer requires, at a minimum, [Visual Studio 2019 version 16.8 Preview](https://visualstudio.microsoft.com/downloads/?utm_medium=microsoft&utm_source=docs.microsoft.com&utm_campaign=inline+link&utm_content=download+vs2019+desktopguide+winforms).
3. The .NET Framework 4 implementation that's supported by Visual Studio 2019 and Visual Studio 2017.
4. .NET Framework 4 is a Windows-only version of .NET and is considered a Windows Operating System component. This version of Windows Forms is distributed with .NET Framework.
5. This Desktop Guide is written for Windows Forms on .NET 5. For more information about the .NET Framework version of Windows Forms, see [Windows Forms for .NET Framework](https://docs.microsoft.com/en-us/dotnet/desktop/winforms/?view=netframeworkdesktop-4.8&preserve-view=true).

**The .NET Framework**

**Overview**

The .NET Framework is a new computing platform that simplifies

application development in the highly distributed environment of the Internet.

The .NET Framework is designed to fulfill the following objectives:

To provide a consistent object-oriented programming environment

whether object code is stored and executed locally, executed locally but

Internet-distributed, or executed remotely.

To provide a code-execution environment that minimizes software

deployment and versioning conflicts.

To provide a code-execution environment that guarantees safe execution

of code, including code created by an unknown or semi-trusted third

party.

To provide a code-execution environment that eliminates the performance

problems of scripted or interpreted environments.

To make the developer experience consistent across widely varying types

of applications, such as Windows-based applications and Web-based

applications.

To build all communication on industry standards to ensure that code

based on the .NET Framework can integrate with any other code.The .NET Framework is a multi-language environment for building,

deploying, and running XML Web services and applications.

 **Common Language Runtime :-**

Despite its name, the runtime actually

has a role in both a component's runtime and development time

experiences. While the component is running, the runtime is responsible

for managing memory allocation, starting up and stopping threads and

processes, and enforcing security policy, as well as satisfying any

dependencies that the component might have on other components. At

development time, the runtime's role changes slightly; because it

automates so much (for example, memory management), the runtime

makes the developer's experience very simple, especially when compared

to COM as it is today. In particular, features such as reflection

dramatically reduce the amount of code a developer must write in order to

turn business logic into a reusable component.

 **Unified programming classes :-**

The framework provides developers

with a unified, object-oriented, hierarchical, and extensible set of class

libraries (APIs). Currently, C++ developers use the Microsoft Foundation

Classes and Java developers use the Windows Foundation Classes. The

framework unifies these disparate models and gives Visual Basic and

JScript programmer’s access to class libraries as well. By creating a

common set of APIs across all programming languages, the common

language runtime enables cross-language inheritance, error handling, and

debugging. All programming languages, from JScript to C++, have

similar access to the framework and developers are free to choose the

language that they want to use.

 **ASP.NET : -**

ASP.NET builds on the programming classes of the .NET

Framework, providing a Web application model with a set of controls and

infrastructure that make it simple to build ASP Web applications.

ASP.NET includes a set of controls that encapsulate common HTML userinterface elements, such as text boxes and drop-down menus. These

controls run on the Web server, however, and push their user interface as

HTML to the browser. On the server, the controls expose an object-

oriented programming model that brings the richness of object-oriented

programming to the Web developer. ASP.NET also provides

infrastructure services, such as session state management and process

recycling that further reduce the amount of code a developer must write

and increase application reliability. In addition, ASP.NET uses these

same concepts to enable developers to deliver software as a service.

**Debugging**

Visual Studio .NET provides a single integrated debugger for all Visual

Studio languages, including Visual Basic and C#. A new, unified interface

combines features of the Visual C++ and Visual Basic 6.0 debuggers, as well as

many new features.

 **Cross-Language Debugging :-**

Using the integrated debugger, you can debug projects that are part of the

same solution but are written in different languages. For example, you can

debug a solution that contains a project consisting of a Visual Basic or Visual

C# user interface application and a Visual C++ server application, and you can

step back and forth between these projects, for example from Managed

Extensions for C++ to Visual Basic and back to Managed Extensions for C++.

 **Attaching to a Running Program**

You can attach the debugger to a program that is already running and

debug the program. Attaching to a running program works the same way

whether the program is running on a host machine or a remote machine. The

program does not need to be launched in the Visual Studio .NET integrated

development environment (IDE).

 **Remote Debugging :-**

You can attach to and debug a process that is running on a different

computer from the one on which you are running Visual Studio. For example, if

you are debugging a Windows Forms client application and an XML Web

service it uses, you can run Visual Studio on the client and then attach to the

server that is providing the XML Web service. You typically perform this kind

of debugging for Web Forms projects or ASP.NET Web Service projects.

 **Debugging Multithreaded Applications**

New for Visual Basic users is the ability to write and debug multithreaded

applications. The Threads window can be used to view the threads that are

running and to switch context.

 **Debugging Multiple Programs**

You can debug multiple programs by attaching to running programs or by

launching multiple programs from the Visual Studio IDE.

 **Debugging ASP.NET Web Applications**

Configuring debugging for ASP.NET Web applications has been

significantly improved. In most cases you can simply choose Start from the

Debug menu and the debugger will automatically attach to the ASP.NET

worker process for debugging Web Forms, even if the ASP.NET worker process

is running on a different server. Security configuration has been made easier by

the addition of a debugger users group.

 **.NET Framework Classes for Debugging and Code Tracing**

You can include instrumentation code in your Microsoft .NET

application. Several .NET Framework classes are available for use both in

**ADO (Activex Data Objects).Net**

Most Visual Basic and Visual C# applications revolve around reading and

updating information in databases. To allow data integration in distributed,

scalable applications, Visual Studio .NET provides support for a new generation

of data access technology: ADO.NET.

**Data Access with ADO.NET**

As you develop applications using ADO.NET, you will have different

requirements for working with data. In some cases, you might simply want todisplay data on a form. In other cases, you might need to devise a way to share

information with another company.

No matter what you do with data, there are certain fundamental concepts

that you should understand about the data approach in ADO.NET. You might

never need to know some of the details of data handling — for example, you

might never need to directly edit an XML file containing data — but it is very

useful to understand the data architecture in ADO.NET, what the major data

components are, and how the pieces fit together.

**New in ADO.Net**

**ADO.NET Does Not Depend On Continuously Live Connections**

In traditional client/server applications, components establish a connection to

a database and keep it open while the application is running. For a variety of

reasons, this approach is impractical in many applications:



Open database connections take up valuable system resources. In most

cases, databases can maintain only a small number of concurrent

connections. The overhead of maintaining these connections detracts

from overall application performance.

Similarly, applications that require an open database connection are

extremely difficult to scale up. An application that does not scale up well

might perform acceptably with four users but will likely not do so with

hundreds. ASP.NET Web applications in particular need to be easily

scalable, because traffic to a Web site can go up by orders of magnitude

in a very short period.

In ASP.NET Web applications, the components are inherently

disconnected from each other. The browser requests a page from the

server; when the server has finished processing and sending the page, it

has no further connection with the browser until the next request. Underthese circumstances, maintaining open connections to a database is not

viable, because there is no way to know whether the data consumer (the

client) requires further data access.

A model based on always-connected data can make it difficult and

impractical to exchange data across application and organizational

boundaries using a connected architecture. If two components need to

share the same data, either have to be connected, or a way must be

devised for the components to pass data back and forth.

For all these reasons, data access with ADO.NET is designed around an

architecture that uses connections sparingly. Applications are connected to the

database only long enough to fetch or update the data. Because the database is

not holding on to connections that are largely idle, it can service many more

users.

**Benefits of ADO.NET**

**Interoperability**

ADO.NET applications can take advantage of the flexibility and broad

acceptance of XML. Because XML is the format for transmitting datasets across

the network, any component that can read the XML format can process data. In

fact, the receiving component need not be an ADO.NET component at all: The

transmitting component can simply transmit the dataset to its destination

without regard to how the receiving component is implemented. The destination

component might be a Visual Studio application or any other application

implemented with any tool whatsoever. The only requirement is that the

receiving component be able to read XML. As an industry standard, XML was

designed with exactly this kind of interoperability in mind.

**Maintainability**

In the life of a deployed system, modest changes are possible, but

substantial, architectural changes are rarely attempted because they are so

difficult. That is unfortunate, because in a natural course of events, such

substantial changes can become necessary. For example, as a deployed

application becomes popular with users, the increased performance load might

require architectural changes. As the performance load on a deployed

application server grows, system resources can become scarce and response

time or throughput can suffer. Faced with this problem, software architects can

choose to divide the server's business-logic processing and user-interface

processing onto separate tiers on separate machines. In effect, the application

server tier is replaced with two tiers, alleviating the shortage of system

resources.

**Programmability**

ADO.NET data components in Visual Studio encapsulate data access

functionality in various ways that help you program more quickly and with

fewer mistakes. For example, data commands abstract the task of building and

executing SQL statements or stored procedures.

**Performance**

For disconnected applications, ADO.NET datasets offer performance

advantages over ADO disconnected recordsets. When using COM marshalling

to transmit a disconnected recordset among tiers, a significant processing cost

can result from converting the values in the recordset to data types recognized

by COM. In ADO.NET, such data-type conversion is not necessary.

**Scalability**

Because the Web can vastly increase the demands on your data,

scalability has become critical. Internet applications have a limitless supply of

potential users. Although an application might serve a dozen users well, it might

not serve hundreds —or hundreds of thousands — equally well. An application

that consumes resources such as database locks and database connections will

not serve high numbers of users well, because the user demand for those limited

resources will eventually exceed their supply.

**Comparison of ADO.NET and ADO**

You can understand the features of ADO.NET by comparing them to

particular features of ActiveX Data Objects (ADO).

**In-memory Representations of Data**

In ADO, the in-memory representation of data is the recordset. In

ADO.NET, it is the dataset. There are important differences between them.

**Number of Tables**

A recordset looks like a single table. If a recordset is to contain data from

multiple database tables, it must use a JOIN query, which assembles the data

from the various database tables into a single result table.

In contrast, a dataset is a collection of one or more tables. The tables

within a dataset are called data tables; specifically, they are DataTable objects.

If a dataset contains data from multiple database tables, it will typically contain

multiple **DataTable** objects. That is, each **DataTable** object typically

corresponds to a single database table or view. In this way, a dataset can mimic

the structure of the underlying database.A dataset usually also contains relationships. A relationship within a

dataset is analogous to a foreign-key relationship in a database —that is, it

associates rows of the tables with each other. For example, if a dataset contains

a table about investors and another table about each investor's stock purchases,

it could also contain a relationship connecting each row of the investor table

with the corresponding rows of the purchase table.

Because the dataset can hold multiple, separate tables and maintain

information about relationships between them, it can hold much richer data

structures than a recordset, including self-relating tables and tables with many

to-many relationships.

**Data Navigation and Cursors**

In ADO you scan sequentially through the rows of the recordset using the

ADO **MoveNext** method. In ADO.NET, rows are represented as collections, so

you can loop through a table as you would through any collection, or access

particular rows via ordinal or primary key index. **DataRelation** objects maintain

information about master and detail records and provide a method that allows

you to get records related to the one you are working with. For example, starting

from the row of the Investor table for "Nate Sun," you can navigate to the set of

rows of the Purchase table describing his purchases.

A *cursor* is a database element that controls record navigation, the ability

to update data, and the visibility of changes made to the database by other users.

ADO.NET does not have an inherent cursor object, but instead includes data

classes that provide the functionality of a traditional cursor. For example, the

functionality of a forward-only, read-only cursor is available in the ADO.NET

**DataReader** object. For more information about cursor functionality, see Data

Access Technologies.**Minimized Open Connections**

In ADO.NET you open connections only long enough to perform a

database operation, such as a Select or Update. You can read rows into a dataset

and then work with them without staying connected to the data source. In ADO

the recordset can provide disconnected access, but ADO is designed primarily

for connected access.

There is one significant difference between disconnected processing in

ADO and ADO.NET. In ADO you communicate with the database by making

calls to an OLE DB provider. In ADO.NET you communicate with the database

through a data adapter (an OleDbDataAdapter or SqlDataAdapter object), which

makes calls to an OLE DB provider or the APIs provided by the underlying data

source. The important difference is that in ADO.NET the data adapter allows

you to control how the changes to the dataset are transmitted to the database —

by optimizing for performance, performing data validation checks, or adding

any other extra processing.

**Sharing Data Between Applications**

Transmitting an ADO.NET dataset between applications is much easier

than transmitting an ADO disconnected recordset. To transmit an ADO

disconnected recordset from one component to another, you use COM

marshalling. To transmit data in ADO.NET, you use a dataset, which can

transmit an XML stream.

The transmission of XML files offers the following advantages over

COM marshalling:**Richer data types**

COM marshalling provides a limited set of data types — those defined by

the COM standard. Because the transmission of datasets in ADO.NET is based

on an XML format, there is no restriction on data types. Thus, the components

sharing the dataset can use whatever rich set of data types they would ordinarily

use.

**Performance**

Transmitting a large ADO recordset or a large ADO.NET dataset can

consume network resources; as the amount of data grows, the stress placed on

the network also rises. Both ADO and ADO.NET let you minimize which data

is transmitted. But ADO.NET offers another performance advantage, in that

ADO.NET does not require data-type conversions. ADO, which requires COM

marshalling to transmit records sets among components, does require that ADO

data types be converted to COM data types.

**Penetrating Firewalls**

A firewall can interfere with two components trying to transmit

disconnected ADO recordsets. Remember, firewalls are typically configured to

allow HTML text to pass, but to prevent system-level requests (such as COM

marshalling) from passing.

Because components exchange ADO.NET datasets using XML, firewalls

can allow datasets to pass.

**SQL SERVER**

**SQL SERVER:**

SQL server is a client/server relational database management system

(RDBMS) that uses transact-SQL to send request between a client and SQL

server.

**Client/server Architecture**

SQL server is designed to be a client/server system. Client/server systems

are constructed so that the database can reside on a central computer, know as a

server, and be shared among several users. When users want to access the date

in SQL server, they run an application on their local computer, know as a client

that connects over a network to the server running SQL server.

SQL server can work with thousands of client applications

simultaneously. The server has features to prevent the logical problems that

occur if a user tries to read or modify data currently being used by others.

While SQL server is designed to work as a server in a client/server

network. It also capable of working as a stand-alone database directly on the

client. The scalability and ease-of-use features of SQL server allow it to work

efficiently on a client without consuming too many resources. SQL server

efficiently allocates the available resources, such as memory, network

bandwidth, and disk I/O, among the multiple users.

 **Additional facility Like TIMESTAMP and TIMESTAMP with TIMEZONE for storing Time.**

 **Flexibility in intervals setting**

 Code’s 12 rules are satisfied. That is,

 Data Representation

 Rule of guaranteed access

 Proper treatment of null value

 Security

 Versioning

 Physical Independence

 Logical Independence

 Integrity constraint independence

 View Updating

 Data Description

 Comprehensive data sub language

 Insert and update rule

**SYSTEM ANALYSIS**

**System Analysis**

System analysis is the first and foremost step performed in developing the

software to solve a particular problem. In the analysis part, a software developer

examines the requirements. Carrying out preliminary investigation identifies

these requirements

Analysis consists of two sub phases

 Planning

 Requirement definition

During planning phase, cost estimates and work schedules will be planned.

Requirement definition is a specification that describes the processing

environment, the required software functions, performance constraints (size,

speed, machine configuration) and exception handling.

**EXISTING SYSTEM**

The existing system uses manual transaction processing.

**Drawbacks**

 Large amount of clerical time is required.

 The record maintainability is difficult.

 Accessibility of accurate information from the past record is difficult.

 There is always delay in information search and retrieval. It requires

many people to carry out a single problem.

 Lot of human resources is required.

 Data reliability and maintainability is difficult.

 Lot amount of records need much place to save.

 The paper works have to be taken care.

 The patient entry form may miss

 The doctor appointment cannot be maintained in properly

 The patient fix the appointment to consulting with doctor such things may

misplaced

 The patients records verification is too complicated

 Compare with the patient and doctor record verification and validation is

too complicated

 Cannot be maintaining manual record for long time

**PROPOSED SYSTEM**

The proposed system has been designed to overcome all the drawbacks

found in the existing system. The new system has been proposed to use

C# as front end and SQL SERVER as backend.The proposed system has enhanced features, which was not found in

the existing system. The salient features are

 Security for the data is done easily.

 Validation is done to enter correct data.

 Memory consumption is very less and the processing speed

is fast.

 Data reports are presented in a neat format.

 It is apt for this modern world.

It is easy to combine the database of other software and to view the

**SYSTEM DESIGN**

**THE SYSTEM DESIGN PROCESS:**

System design develops the architectural detail required to build a

system or product.

**The system design process encompasses the following activities:**

• Partition the analysis model into subsystems.

• Identify concurrency that is dictated by the problem.

• Allocate subsystems to processors and tasks.

• Develop a design for the user interface.

• Choose a basic strategy for implementing data management.

• Identify global resources and the control mechanisms required to access them.

• Design an appropriate control mechanism for the system, including task

management.

• Consider how boundary conditions should be handled.

• Review and consider trade-offs.

**Input design**

Input design is a part of overall system design, requires the very

careful analysis of the input data items. The goal of the input design is to make

the data entry easier, logical and free from errors. The user controls input data.

The commonly used input, output devices are mouse, keyboard

and the visual display unit. The well designed, well organized screen formats

are used to acquire the inputs. The data accepted is stored on database file.

Our system is classified into subsystem such as

 patient Admission

 Doctor Details

 Diagnosis

 Consultation Details

**Output Design**

Output is the most important and direct source of information the user.

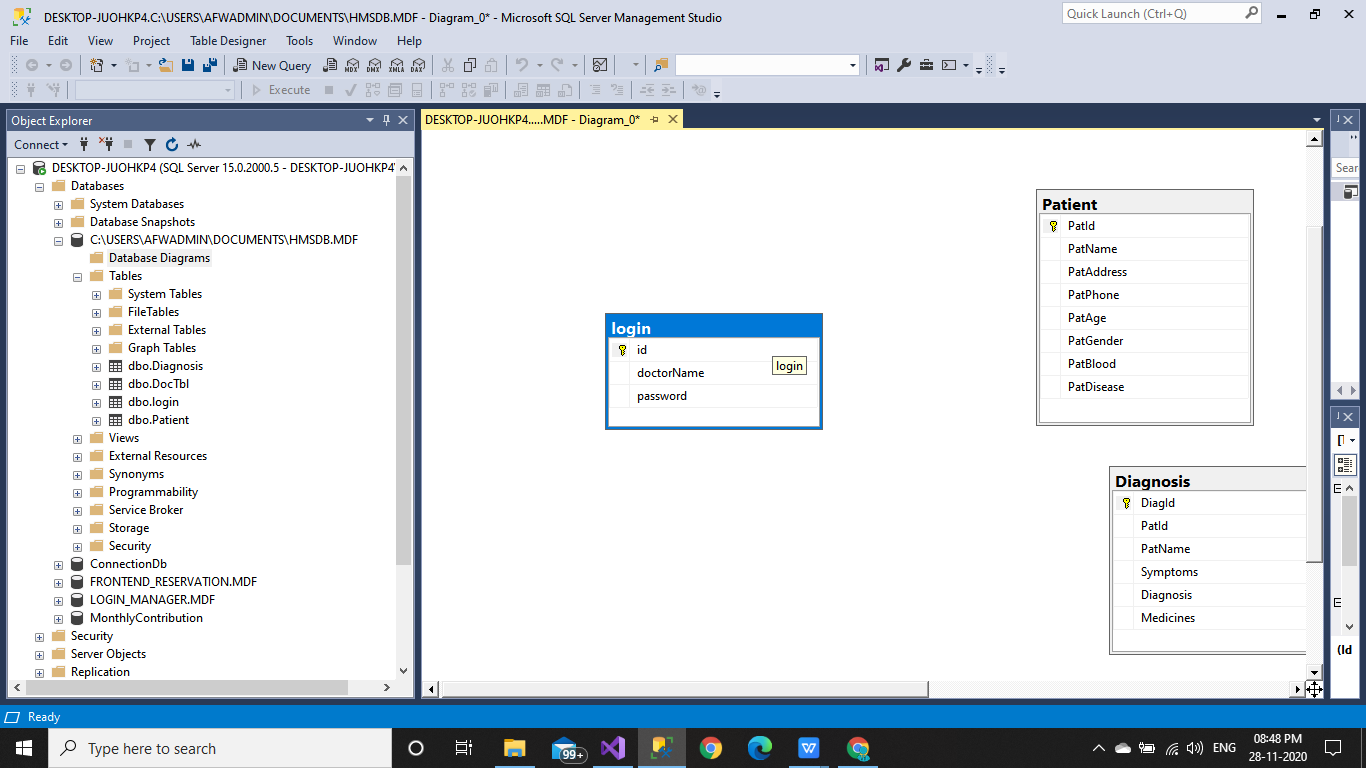
Efficient & intelligent output design improves the system relationships

with the users and helps in decision-making. The output is collected in order to

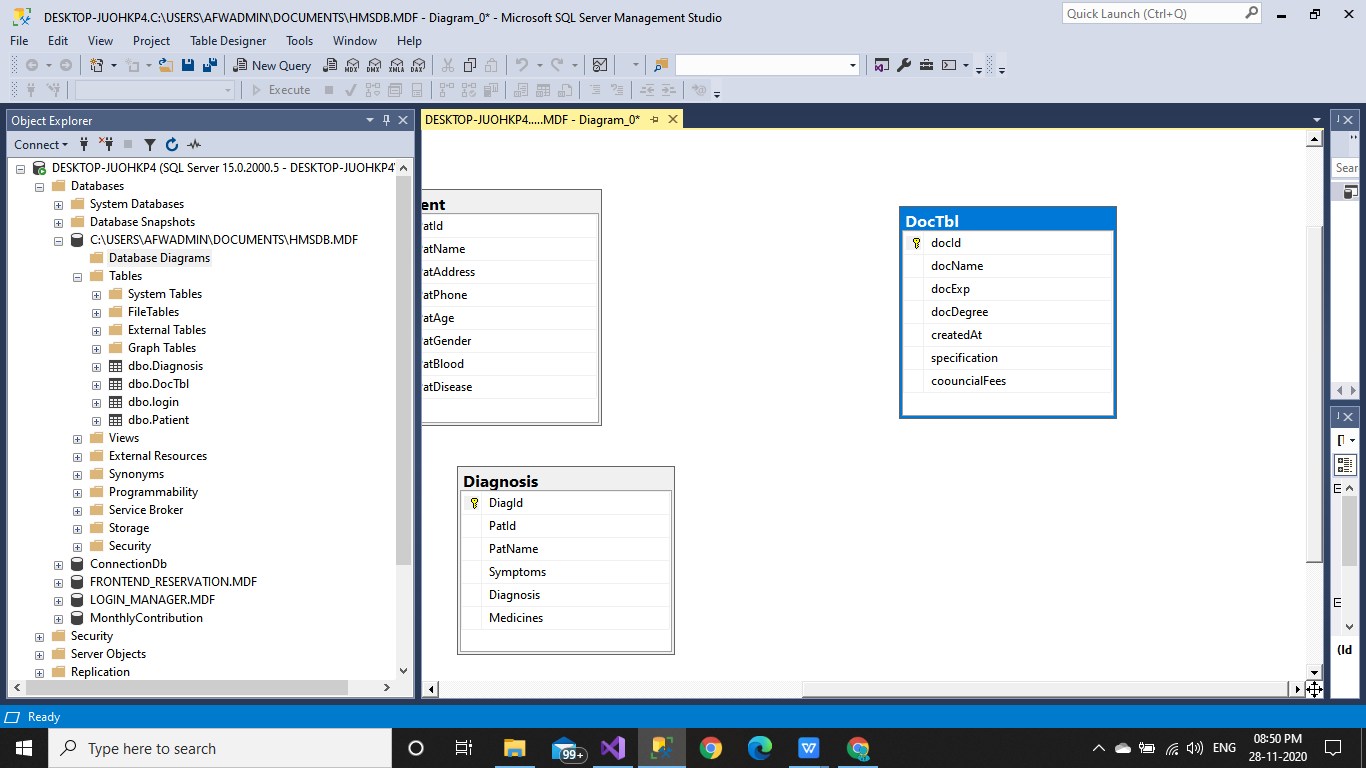
help the user to make a wise decision

**DATABASE DESIGN**

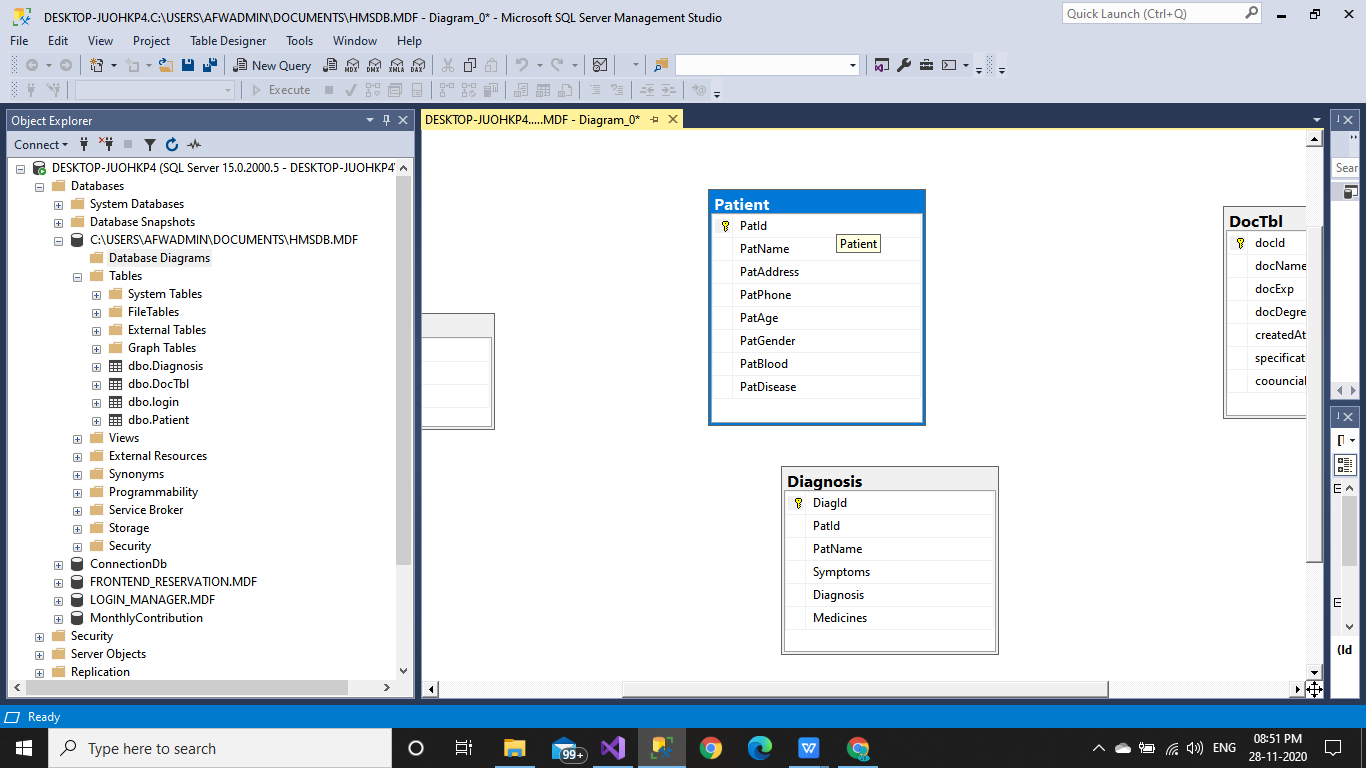
**Login Table :-**



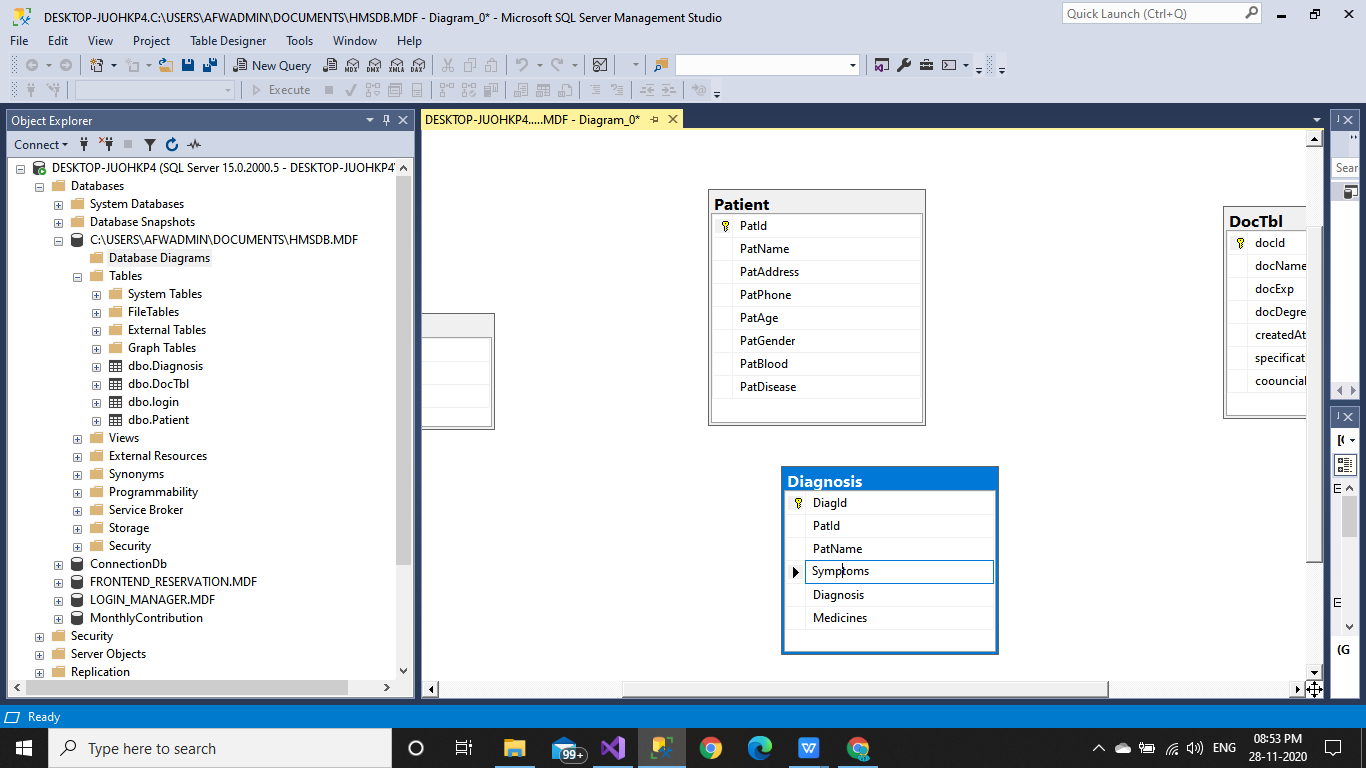
Doctor Table :-



Patient Table :-



Diagnosis Table :-



**DATA FLOW DIAGRAM**

**DATA FLOW DIAGRAM**

Data Flow Diagram (DFD) is a design tool constructed to show how data

within the system. It is designed from the data which is collected during data

collection phase. DFD is otherwise called as “Bubble Chart”.

There are five symbol used in DFD. They are Rectangle, Open

Rectangle, Circle, arrow, small circle. Each one has its own meaning.

= Source or Destination

= Data flow

= Process

= Data Storage

= Control Flow

database

Check User

Login

Home

Diagnosis

Doctor

Patient

Name

Experience

Specification

Degree

Fees

Symptoms

Diagnosis

Medicines

Patient Name

Name

Address

Phone No.

Age

Gender

Blood group

Major disease

Hospital Management System Database

**SYSTEM TESTING**

**System Testing**

System Testing is an important stage in any system development life

cycle. Testing is a process of executing a program with the intention of finding

errors. The importance of software testing and its implications with respect to

software quality cannot be overemphasized. Software testing is a critical

element of software quality assurance and represents the ultimate review of

specification, design and coding. A good test case is one that has a high

probability of finding a yet undiscovered error.

Testing is the set of activities that can be planned in advance and

conducted systematically. Different test conditions should be thoroughly

checked and the bugs detected should be fixed. The testing strategies formed by

the user are performed to prove that the software is free and clear from errors.

To do this, there are many ways of testing the system’s reliability, completeness

and maintainability.

**Unit Testing:**

In the unit testing the analyst tests the program making up a system. The

software units in a system are the modules and routines that are assembled and

integrated to perform a specific function. In a large system, many modules on

different levels are needed.

Unit testing can be performed from the bottom up starting with the

smallest and lowest level modules and proceeding one at a time. For each

module in a bottom-up testing, a short program executes the module and

provides the needed data.**Integration Testing:**

Integration testing is a systematic technique for constructing the program

structure while conducting test to uncover errors associate with interfacing.

Objectives are used to take unit test modules and built program structure that

has been directed by design.

The integration testing is performed for this Hospital Management

System when all the modules where to make it a complete system. After

integration the project works successfully.

**Validation Testing:**

Validation testing can be defined in many ways, but a simple definition is

that can be reasonably expected by the customer. After validation test has been

conducted, one of two possible conditions exists.

 The functions or performance characteristics confirm

to specification and are accepted.

 A deviation from specification is uncovered and a

deficiency list is created.

Proposed system under consideration has been tested by using validation

testing and found to be working satisfactorily.

For example, in this project validation testing is performed against

inpatient search module. This module is tested with the following valid and

invalid inputs for the field patientname.**White Box Testing**

White box testing, sometimes called glass-box testing is a test case design

method that uses the control structure of the procedural design to derive test

cases. Using white box testing methods, the software engineer can derive test

cases that

 Guarantee that all independent paths with in a module have been

exercised at least once.

 Exercise all logical decisions on their true and false sides.

 Execute all loops at their boundaries and with in their operational

bounds and

 Exercise internal data structure to assure their validity.

For example in this project white box testing is performed against

inpatient module. Without entering text if we apply it displays the message

“First add record then save it” else it should be saved.

**Black Box Testing**

This method treats the coded module as a black box. The module runs

with inputs that are likely to cause errors. Then the output is checked to see if

any error occurred. This method cannot be used to test all errors, because some

errors may depend on the code or algorithm used to implement the module.**System Implementation**

Implementation is the process of having system personal check out and

provides new equipments into use, train the users to install a new application

and construct any files of data needed to use it. There are three types of

implementation.

 Implementation of computer system to replace a manual system.

The problems encountered are covering files, training users,

creating accurate files and verifying print outs for integrity.

 Implementation of a new computer system to replace an existing

one. This is usually difficult conversion. If not properly planned,

there can be many problems. So large computer systems may take

as long as a year to convert.

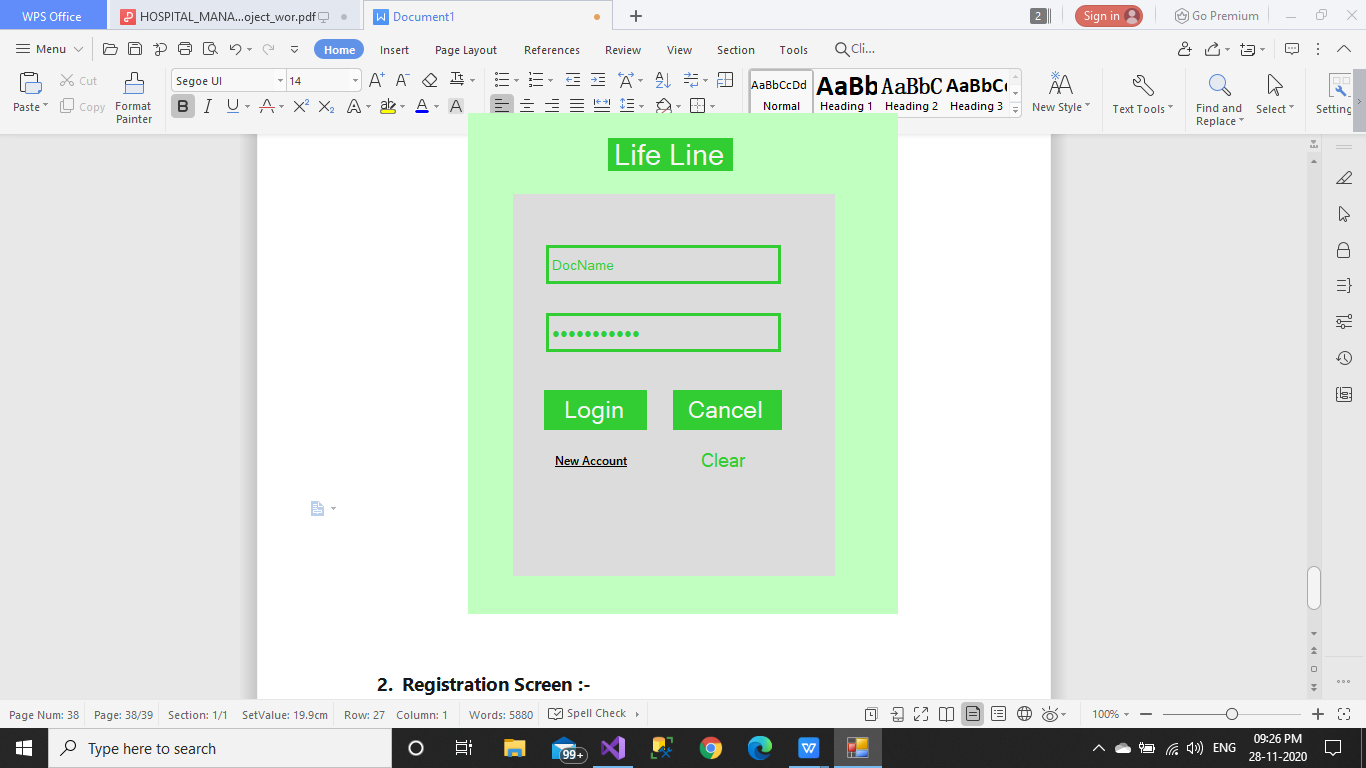
 Implementation of a modified application to replace the existing

one using the same computer. This type of conversion is relatively

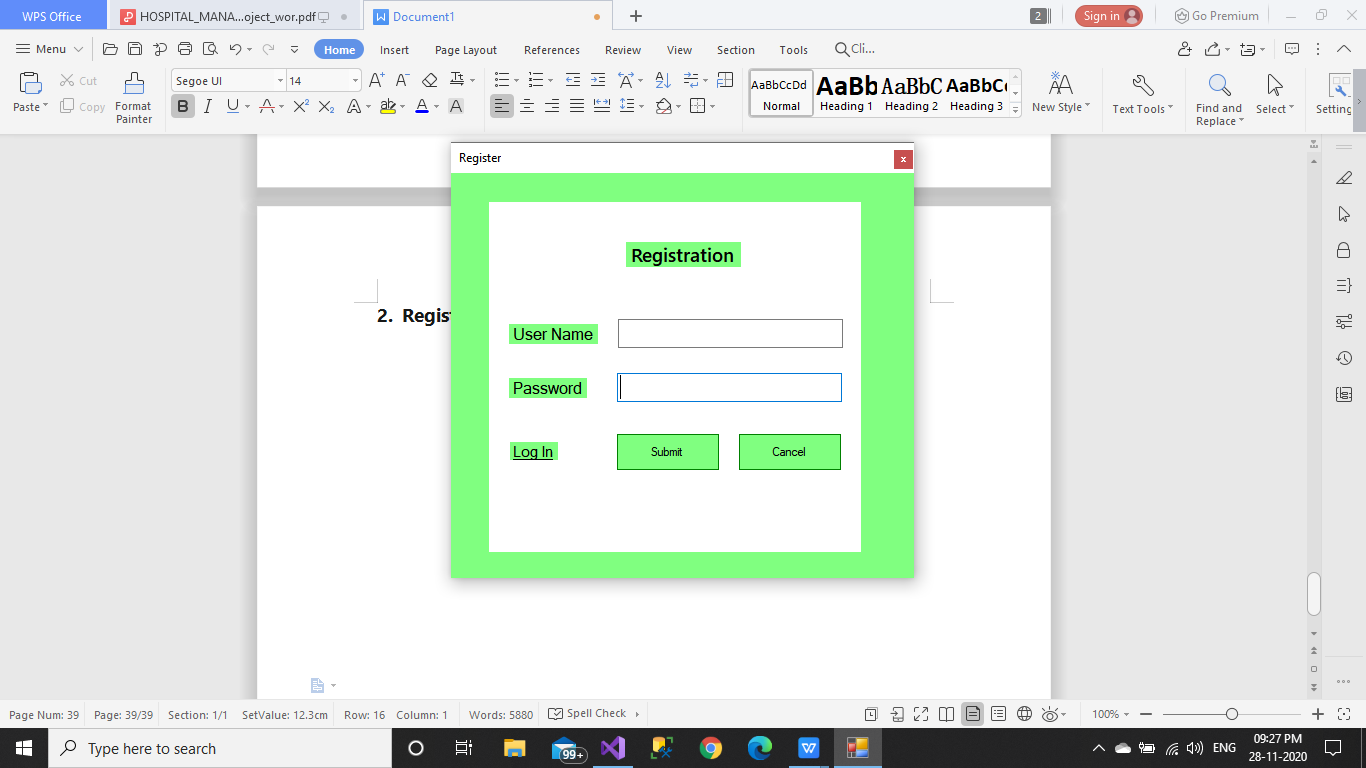
easy to handle, usually there are no major changes in the file.

**Screen Shoot :-**

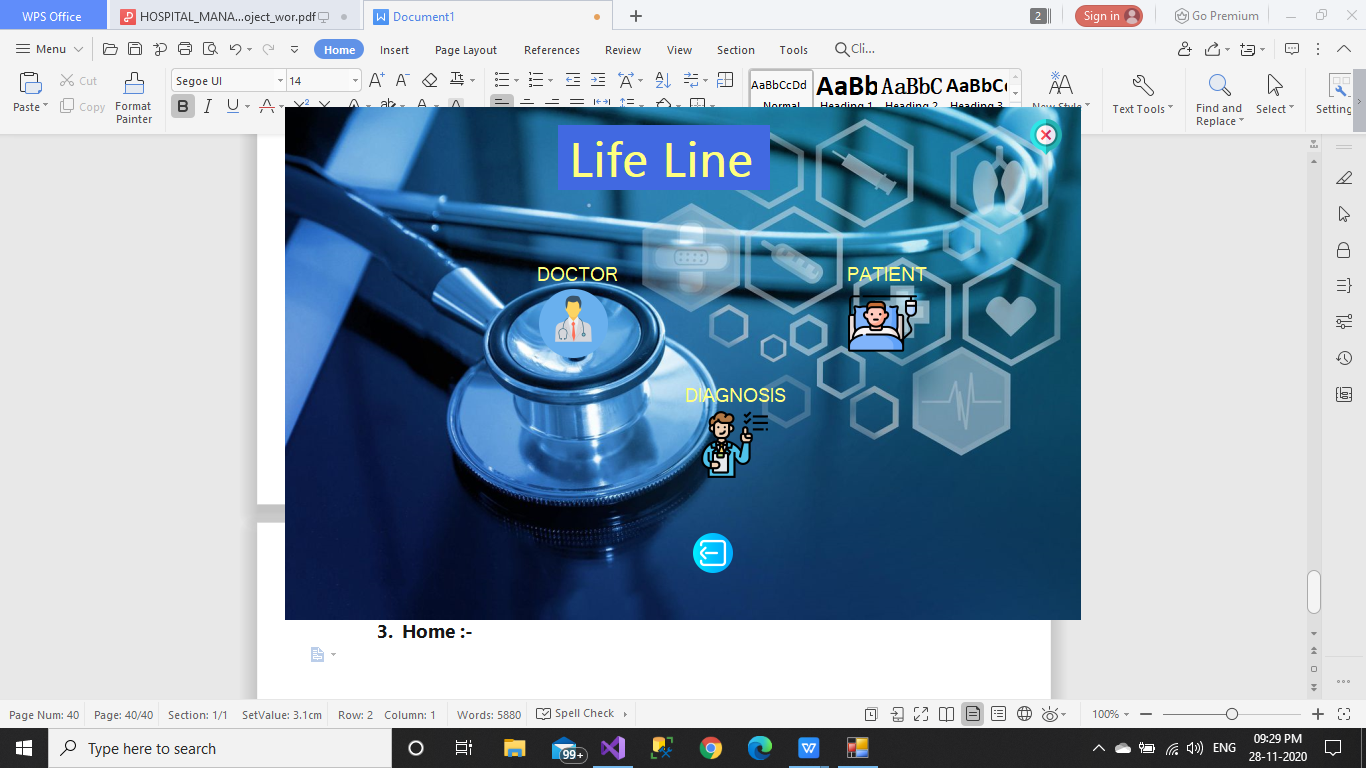
1. **Login Screen :-**



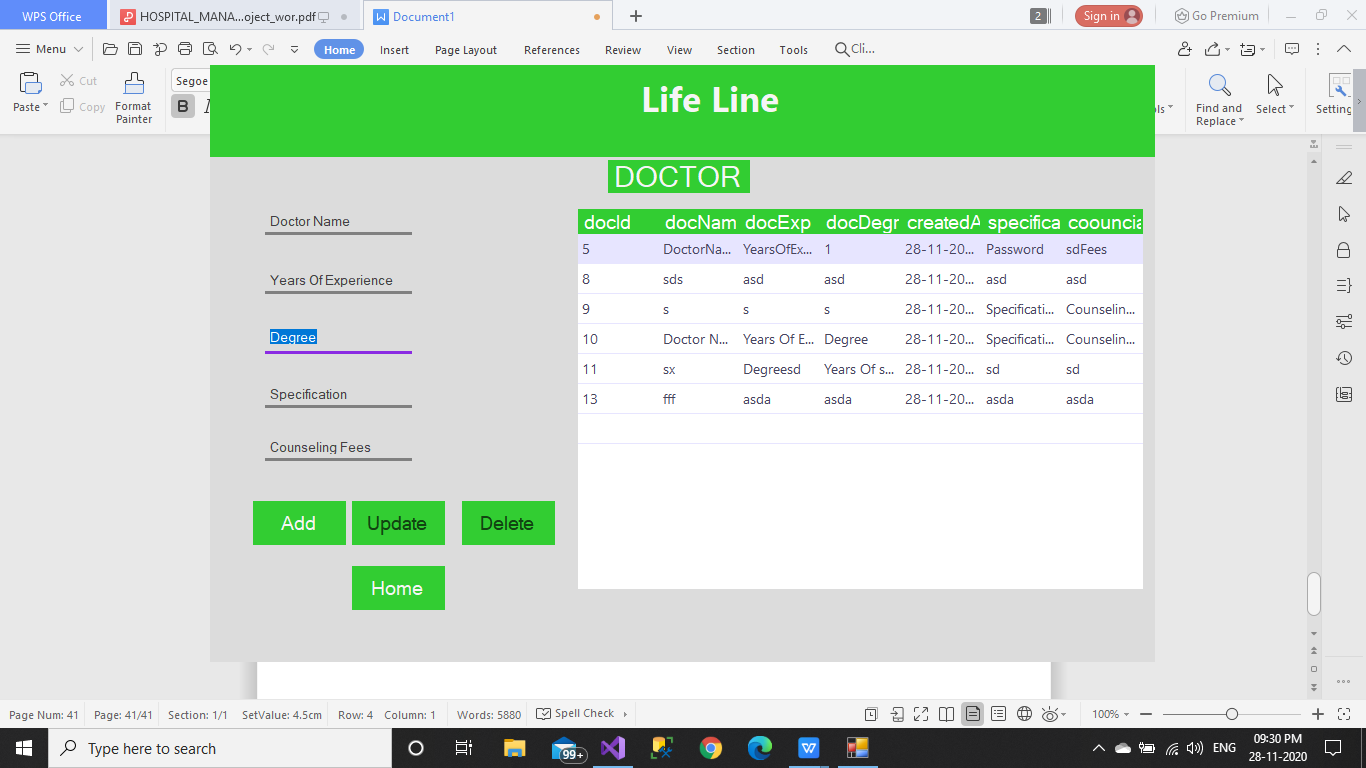
1. **Registration Screen :-**



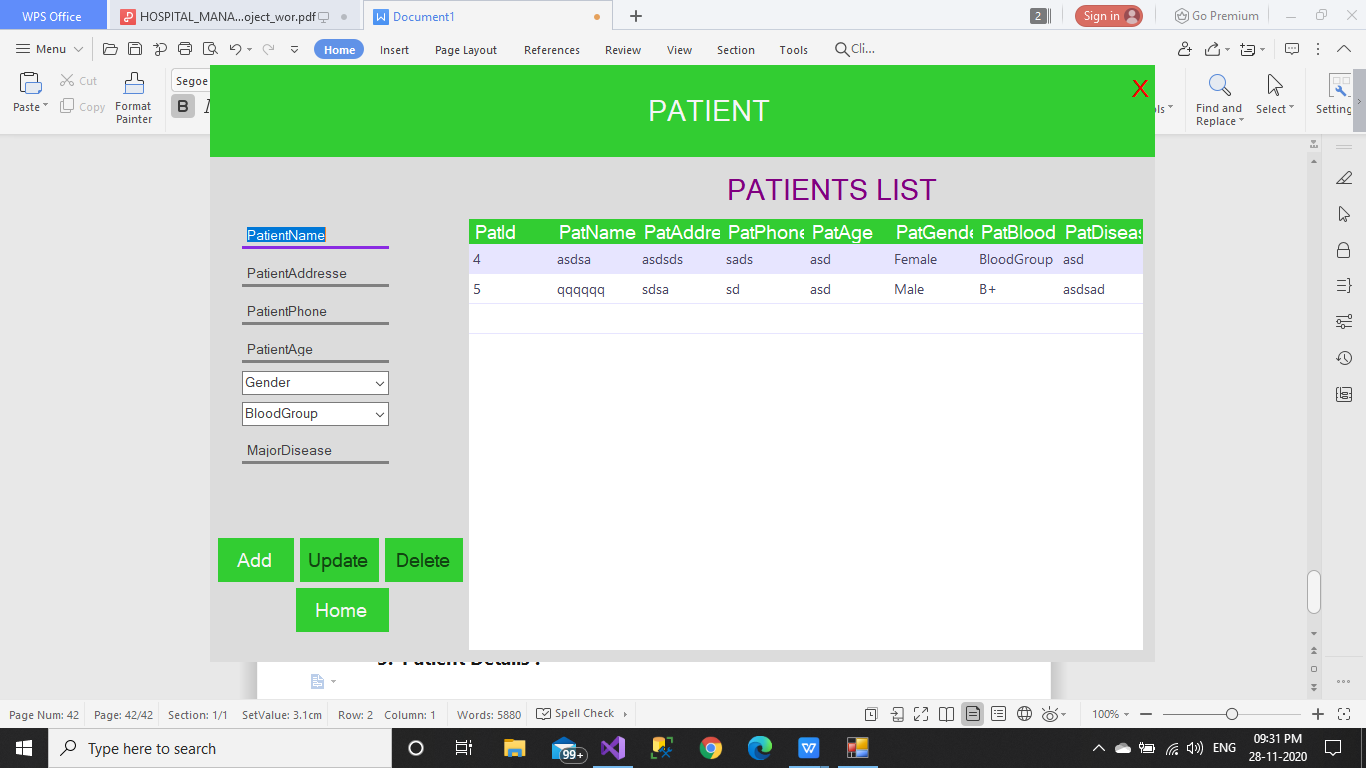
1. **Home :-**



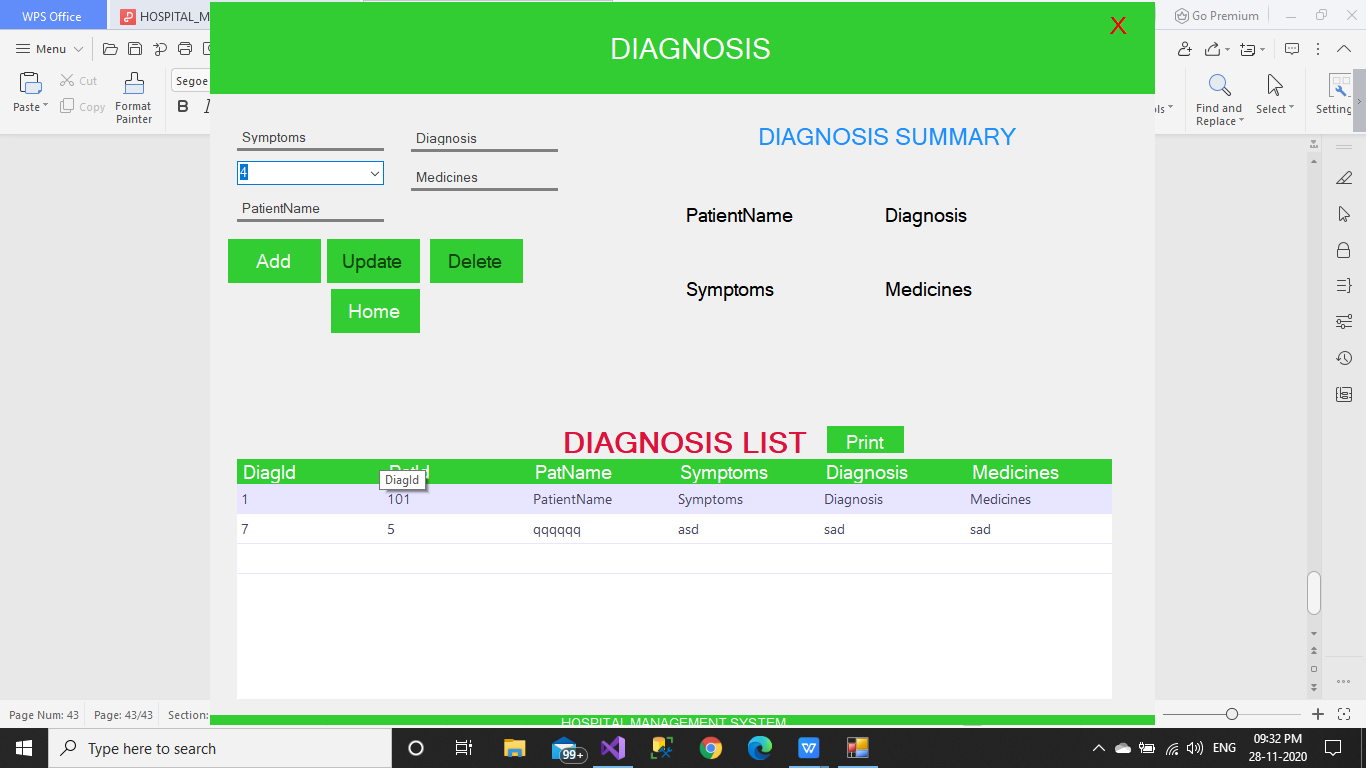
1. **Doctor Details :-**



1. **Patient Details :-**



1. **Diagnosis : -**



1. **Splash :-**

