Proposal of Hospital Management System

Avishek Sah, Raj Oli, Abhay Pratap Sah

Bsc. (Hons.) Computing, Softwarica College of IT and E-commerce, Coventry University

ST4008CEM Computing Activity LED Learning Project

Giriraj Rawat

July 28 ,2022

Table Of Contents

Introduction

The hospital management system project is a computer system that allows the effective management of healthcare information and the task completion of healthcare personnel. Clinical, financial, and laboratory elements of healthcare are all governed by the data in this hospital management system project report. It includes a pdf report and supporting materials that outline the entire hospital management system project.

The project Hospital Management System comprises automated billing in the labs and pharmacy as well as patient registration and data storage. Every patient can receive a unique ID from the software, which also automatically stores staff and patient information. It has a search feature so you can see how each room is doing right now. Using the ID, a user can look up a doctor's availability and a patient's information.

With the use of a login and password, one can access the Hospital Management System. A receptionist or an administrator can access it. They alone are able to add data to the database. The information is simple to retrieve. The user experience is excellent. The information is secure for personal use only.

Fig 1: Hospital Management System

Aim

A hospital management system is a piece of software created to oversee all aspects of a hospital, including the provision of services and related financial, administrative, and medical management. The Hospital Management System is strong, adaptable, and simple to use.

A comprehensive variety of hospital management and administrative processes are covered by the Hospital Management System, which is developed for multispecialty facilities. It is a fully integrated end-to-end hospital management system that offers pertinent data throughout the hospital to assist efficient decision-making for vital financial accounting, hospital administration, and patient care in a smooth flow.

Clinical process analysis and activity-based pricing are two areas where the Hospital Management System, a software product package, is intended to improve the quality and management of hospital management.

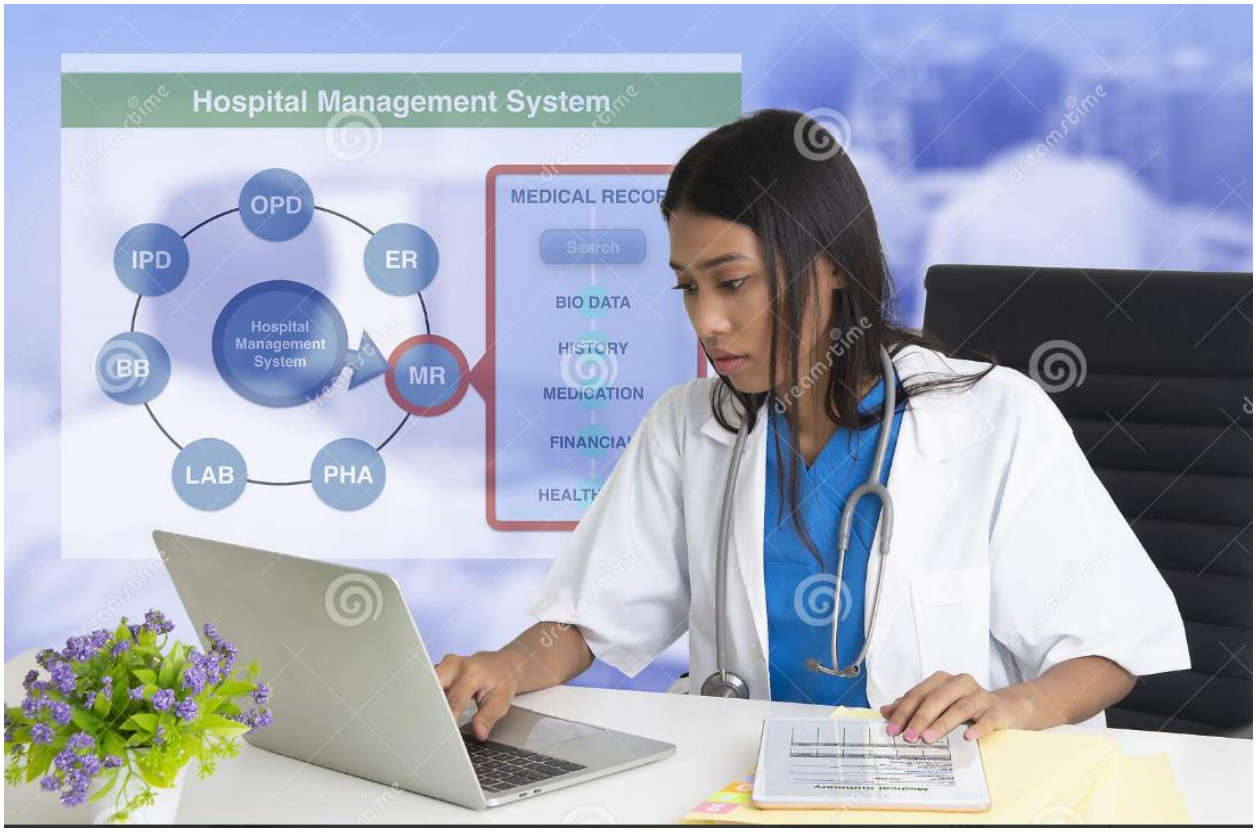


Fig 2: Aim for Hotel Management System

Objectives

A hospital management system's primary goals are to: Create a system for better patient care. cut back on hospital overhead. Giving management MIS (Management Information System) reports as needed would help them make better decisions.

Keeping tabs on the daily operations and records of the hospital's patients, doctors, nurses, ward boys, and other staff members who are essential to the facility's performance. However, it is extremely time-consuming and error-prone to keep track of all the actions and their records on paper. Seeing the ongoing rise in population and the number of hospital visitors is another indication of how inefficient and time-consuming the process is. All of these records must be kept, which is extremely unreliable, inefficient, and error-prone. Maintaining these records on paper is also not economically or technically possible. Keeping the manual system's operation as our project's foundation. The "Administration support system for medical institutions" is an automated version of the manual system that we have created.

Data flow diagram of Hospital Management System

The most abstract representation of a system's data flow is a context diagram. The entire system is depicted as a single bubble. The several external entities that the system interacts with are all depicted, as are the data flows that take place between the system and the external entities. The name context diagram is justified because it depicts the environment in which the system will operate, including the users and other external entities that will interact with the system.

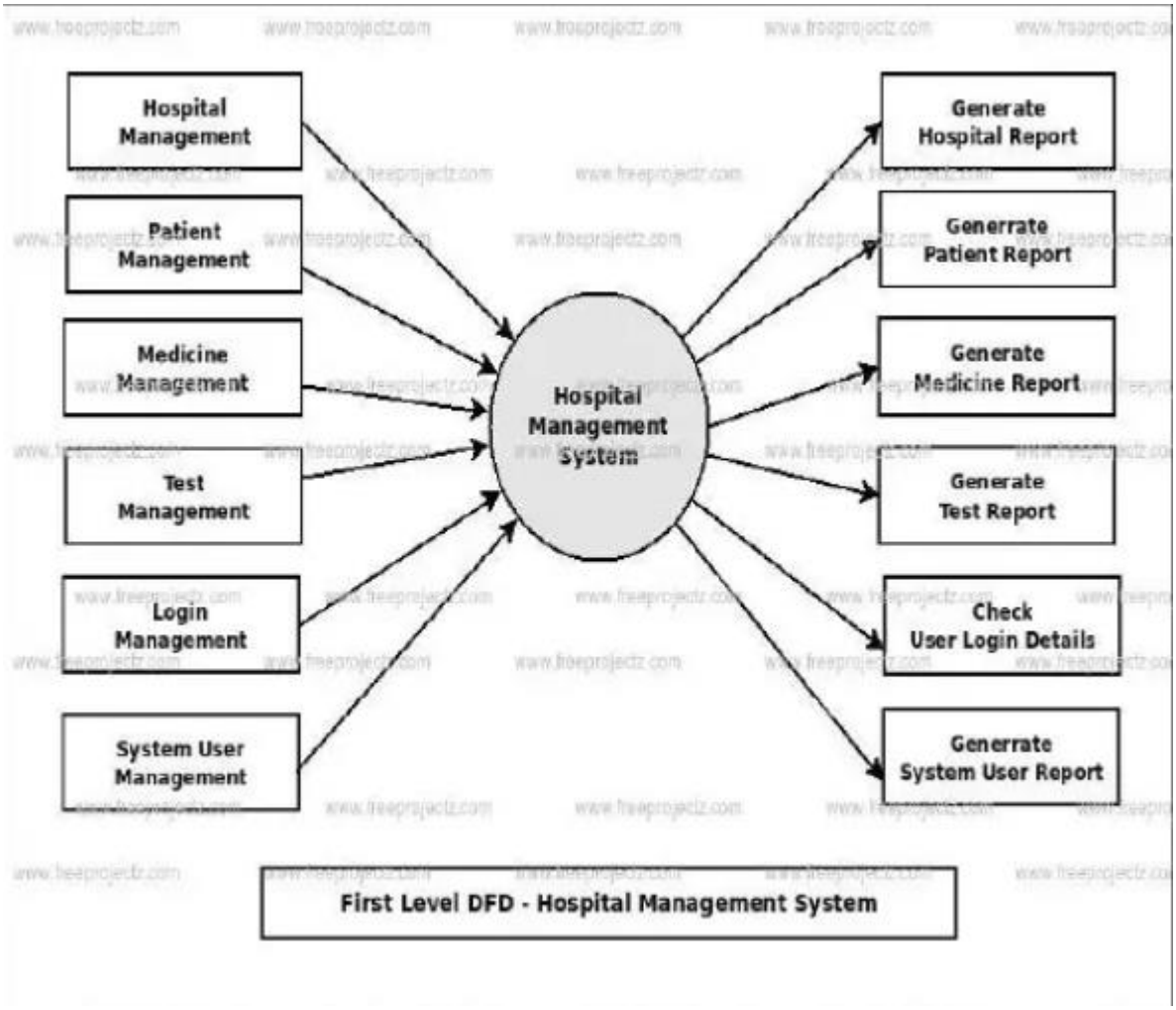


Fig: 3 Data flow diagram of Hospital Management System

Problem statement

I chose to work on this project since Hospital is connected to everyday life and habits of regular people. The manual handling of the record takes a long time and is very error-prone. The goal of this project is to automate or put online routine tasks such room duties, new patient admission, patient discharge, assigning a doctor, and billing calculations. I did my best to simplify the challenging process.

System for managing hospitals using a menu-oriented interface, structured and modular techniques, and simplicity as much as possible. I made an attempt to create the software such that users would not encounter any problems using it and that future extension would be simple and straightforward.

Risk Management

Software for Risk Management: A proactive strategy for reducing project-related uncertainty and potential loss is risk management. Product size, business impact, customer-related, process, technology, development environment, staffing (size and experience),

schedule, and cost are a few types of risk. Project risk management is a practice that includes procedures, techniques, and tools.

Identification of risks is a methodical endeavor to identify dangers to the project strategy. We may take the first step toward avoiding risks when it's possible and regulating them when it's required by identifying known and foreseeable dangers. We divided the risks into various categories in order to accomplish the risk identification, including:1.

1.Task Risk

2.Technical Danger

3. Enterprise Risk

4. Known Risk

5.Probabilistic

6.Unpredictable

Features

HL-7(Health Level-7) interfaces across all modules that facilitate standards conformance.

Flow of the financial data with centralized information to the payer modules.

Data for the management reviews displays graphically by executive information system (EIS).

The billing, insurance processing modules effectively process clinical data for efficient payout for health encounters.

Functional requirements

Two sections make up this system interface.

1.interface for administrators.

2.People interact.

Administrator Interface

A post may be deleted by an administrator.

User accounts can be confirmed by the administrator.

User Interface

Without creating an account, users can peruse all advertising.

Create an account before posting an advertisement3.

A user's personal account can be updated or edited.

Login and logout procedures.

Methodology

This chapter offers a thorough explanation of the issue, examines how it is perceived by various parties, and then offers a conceptual approach to problem solving. It also discusses possible solutions and the value of problem solving before moving on to a theoretical program design for problem solving and technology. This problem's solution was studied. 2.2 Problem definitions a theoretical investigation into the issue and potential

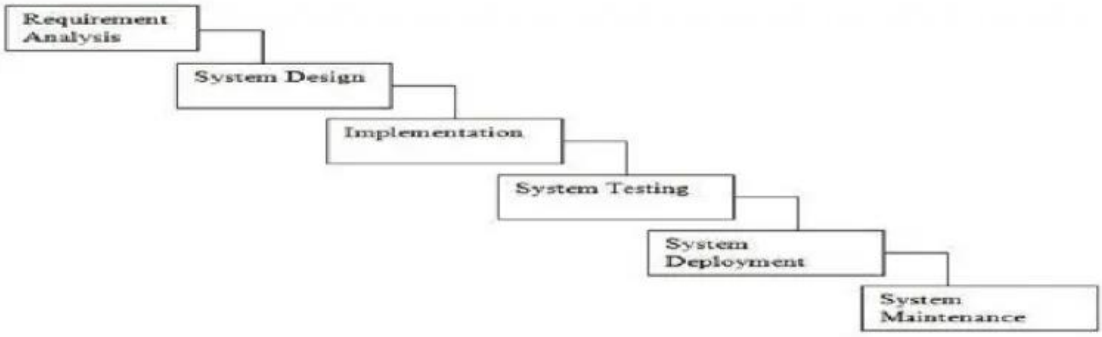


Fig. 2.1: Waterfall model

Tools and Technology

An evaluation of the hardware that is already in place and the hardware that is still required for the system's full introduction should be done. Users frequently complain that information systems (IS) are not user-friendly and lack straightforward data input, which is one of the main difficulties with IS. The method used to enter data into a system reveals the practitioner's practice style. Therefore, it is necessary for the data structure and interface design to match. It also depends on the technology being used, which is another problem. The introduction of such a system presents challenges in terms of flexibility and adaptation. Finding the appropriate language for input is another issue.

Fig:4 Tools and technology for Hospital Management System

Project Plan

Project management, which involves using timetables like to plan and then report progress within the project environment, includes project planning. In the beginning, the project's scope is established, and the best ways to finish the project are chosen. The times required to accomplish the various jobs are then enumerated and organized into a work breakdown structure after this stage. An activity network diagram is used to define the logical relationships between tasks and to identify the critical path.

```
                if Admonitions == "1":  # Admin mode --> Patients Management
                    print ("----------------------------------------")
                    print ("|To add new patient Enter 1      |")
                    print ("|To display patient Enter 2      |")
                    print ("|To delete patient data Enter 3      |")
                    print ("|To edit patient data Enter 4        |")
                    print ("|To Back enter B                 |")
                    print ("----------------------------------------")
                    Admin_choice = input("Enter your choice : ")
                    Admin_choice = Admin_choice.upper()

                    if Admin_choice == "1":  # Admin mode --> Pateints
Management --> Enter new patient data
                        try:  # To avoid non integer input
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID in Pateints_DataBase:  # if Admin
entered used ID
                                patient_ID = int(input("This ID is
unavailable, please try another ID : "))
                            Department = input("Enter patient
department                : ")
                            DoctorName = input("Enter name of doctor following
the case : ")
                            Name = input("Enter patient
name                      : ")
                            Age = input("Enter patient
age                       : ")
                            Gender = input("Enter patient
gender                    : ")
                            Address = input("Enter patient
address                   : ")
                            RoomNumber = input("Enter patient room
number                : ")
                            Pateints_DataBase[patient_ID] = [Department,
DoctorName, Name, Age, Gender, Address,
                                                             RoomNumber]
```

```
                            print("----------------------Patient added
successfully----------------------")
                        except:
                            print("Patient ID should be an integer number")

                    elif Admin_choice == "2":  # Admin mode --> Pateints
Management --> Display patient data
                        try:  # To avoid non integer input
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID not in Pateints_DataBase:
                                patient_ID = int(input("Incorrect ID, Please
Enter patient ID : "))
                            print("\npatient name        : ",
Pateints_DataBase[patient_ID][2])
                            print("patient age         : ",
Pateints_DataBase[patient_ID][3])
                            print("patient gender      : ",
Pateints_DataBase[patient_ID][4])
                            print("patient address     : ",
Pateints_DataBase[patient_ID][5])
                            print("patient room number : ",
Pateints_DataBase[patient_ID][6])
                            print("patient is in " +
Pateints_DataBase[patient_ID][0] + " department")
                            print("patient is followed by doctor : " +
Pateints_DataBase[patient_ID][1])
                        except:
                            print("Patient ID should be an integer number")

                    elif Admin_choice == "3":  # Admin mode --> Pateints
Management --> Delete patient data
                        try:  # To avoid non integer input
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID not in Pateints_DataBase:
                                patient_ID = int(input("Incorrect ID, Please
Enter patient ID : "))
                            Pateints_DataBase.pop(patient_ID)
                            print("----------------------Patient data deleted
successfully----------------------")
                        except:
                            print("Patient ID should be an integer number")

                    elif Admin_choice == "4":  # Admin mode --> Pateints
Management --> Edit patient data
                        try:  # To avoid non integer input
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID not in Pateints_DataBase:
```

```
                                patient_ID = int(input("Incorrect ID, Please
Enter patient ID : "))
                            while True:
                                print("----------------------------------------
---")
                                print("|To Edit pateint Department Enter 1
:    |")
                                print("|To Edit Doctor following case Enter 2
: |")
                                print("|To Edit pateint Name Enter 3
:          |")
                                print("|To Edit pateint Age Enter 4
:           |")
                                print("|To Edit pateint Gender Enter 5
:        |")
                                print("|To Edit pateint Address Enter 6
:       |")
                                print("|To Edit pateint RoomNumber Enter 7
:    |")
                                print("|To be Back Enter
B                      |")
                                print("----------------------------------------
--")
                                Admin_choice = input("Enter your choice : ")
                                Admin_choice = Admin_choice.upper()
                                if Admin_choice == "1":
                                    Pateints_DataBase[patient_ID][0] =
input("\nEnter patient department : ")
                                    print(
                                        "----------------------Patient
Department edited successfully----------------------")

                                elif Admin_choice == "2":
                                    Pateints_DataBase[patient_ID][1] =
input("\nEnter Doctor followuing case : ")
                                    print(
                                        "----------------------Doctor
followuing case edited successfully----------------------")

                                elif Admin_choice == "3":
                                    Pateints_DataBase[patient_ID][2] =
input("\nEnter patient name : ")
                                    print(
                                        "----------------------Patient name
edited successfully----------------------")

                                elif Admin_choice == "4":
```

```
                                    Pateints_DataBase[patient_ID][3] =
input("\nEnter patient Age : ")
                                    print("----------------------Patient age
edited successfully----------------------")

                                elif Admin_choice == "5":
                                    Pateints_DataBase[patient_ID][4] =
input("\nEnter patient gender : ")
                                    print(
                                        "----------------------Patient address
gender successfully----------------------")

                                elif Admin_choice == "6":
                                    Pateints_DataBase[patient_ID][5] =
input("\nEnter patient address : ")
                                    print(
                                        "----------------------Patient address
edited successfully----------------------")

                                elif Admin_choice == "7":
                                    Pateints_DataBase[patient_ID][6] =
input("\nEnter patient RoomNumber : ")
                                    print(
                                        "----------------------Patient
RoomNumber edited successfully----------------------")

                                elif Admin_choice == "B":
                                    break

                                else:
                                    print("Please Enter a correct choice")
                            except:
                                print("Patient ID should be an integer number")

                        elif Admin_choice == "B":  # Admin mode --> Pateints
Management --> Back
                            break

                        else:
                            print("Please enter a correct choice\n")

                    elif AdminOptions == "2":  # Admin mode --> Doctors Management
                        print("----------------------------------------")
                        print("|To add new doctor Enter 1             |")
                        print("|To display doctor Enter 2             |")
                        print("|To delete doctor data Enter 3         |")
                        print("|To edit doctor data Enter 4           |")
                        print("|To be back enter B                    |")
```

```
                    print("----------------------------------------")
                    Admin_choice = input("Enter your choice : ")
                    Admin_choice = Admin_choice.upper()

                    if Admin_choice == "1":  # Admin mode --> Doctors
Management --> Enter new doctor data
                        try:  # To avoid non integer input
                            Doctor_ID = int(input("Enter doctor ID : "))
                            while Doctor_ID in Doctors_DataBase:  # if Admin
entered used ID
                                Doctor_ID = int(input("This ID is unavailable,
please try another ID : "))

                            Department = input("Enter Doctor department : ")
                            Name = input("Enter Doctor name       : ")
                            Address = input("Enter Doctor address    : ")
                            Doctors_DataBase[Doctor_ID] = [[Department, Name,
Address]]
                            print("----------------------Doctor added
successfully----------------------")
                        except:
                            print("Doctor ID should be an integer number")

                    elif Admin_choice == "2":  # Admin mode --> Doctors
Management --> Display doctor data
                        try:  # To avoid non integer input
                            Doctor_ID = int(input("Enter doctor ID : "))
                            while Doctor_ID not in Doctors_DataBase:
                                Doctor_ID = int(input("Incorrect ID, Please
Enter doctor ID : "))
                            print("Doctor name    : ",
Doctors_DataBase[Doctor_ID][0][1])
                            print("Doctor address : ",
Doctors_DataBase[Doctor_ID][0][2])
                            print("Doctor is in " +
Doctors_DataBase[Doctor_ID][0][0] + " department")
                        except:
                            print("Doctor ID should be an integer number")

                    elif Admin_choice == "3":  # Admin mode --> Doctors
Management --> Delete doctor data
                        try:  # To avoid non integer input
                            Doctor_ID = int(input("Enter doctor ID : "))
                            while Doctor_ID not in Doctors_DataBase:
                                Doctor_ID = int(input("Incorrect ID, Please
Enter doctor ID : "))
                            Doctors_DataBase.pop(Doctor_ID)
```

```
                            print("/----------------------Doctor data deleted
successfully----------------------/")
                        except:
                            print("Doctor ID should be an integer number")

                    elif Admin_choice == "4":  # Admin mode --> Doctors
Management --> Edit Doctor data
                        try:  # To avoid non integer input
                            Doctor_ID = input("Enter doctor ID : ")
                            while Doctor_ID not in Doctors_DataBase:
                                Doctor_ID = int(input("Incorrect ID, Please
Enter doctor ID : "))
                            print("----------------------------------------")
                            print("|To Edit doctor's department Enter 1    |")
                            print("|To Edit doctor's name Enter 2          |")
                            print("|To Edit doctor's address Enter 3       |")
                            print("To be Back Enter B                      |")
                            print("----------------------------------------")
                            Admin_choice = input("Enter your choice : ")
                            Admin_choice = Admin_choice.upper()
                            if Admin_choice == "1":
                                Doctors_DataBase[Doctor_ID][0][0] =
input("Enter Doctor's Department : ")
                                print(
                                    "/----------------------Doctor's
department edited successfully----------------------/")

                            elif Admin_choice == "2":
                                Doctors_DataBase[Doctor_ID][0][1] =
input("Enter Doctor's Name : ")
                                print("----------------------Doctor's name
edited successfully----------------------")

                            elif Admin_choice == "3":
                                Doctors_DataBase[Doctor_ID][0][2] =
input("Enter Doctor's Address : ")
                                print(
                                    "----------------------Doctor's address
edited successfully----------------------")

                            elif Admin_choice == "B":
                                break

                            else:
                                print("\nPlease enter a correct choice\n")

                        except:
                            print("Doctor ID should be an integer number")
```

```
                    elif Admin_choice == "B":  # Back
                        break

                    else:
                        print("\nPlease enter a correct choice\n")

                elif AdminOptions == "3":  # Admin mode --> Appointment
Management
                    print("----------------------------------------")
                    print("|To book an appointment Enter 1         |")
                    print("|To edit an appointment Enter 2         |")
                    print("|To cancel an appointment Enter 3       |")
                    print("|To be back enter B                     |")
                    print("----------------------------------------")
                    Admin_choice = input("Enter your choice : ")
                    Admin_choice = Admin_choice.upper()
                    if Admin_choice == "1":  # Admin mode --> Appointment
Management --> Book an appointment
                        try:  # To avoid non integer input
                            Doctor_ID = int(input("Enter the ID of doctor :
"))
                            while Doctor_ID not in Doctors_DataBase:
                                Doctor_ID = int(input("Doctor ID incorrect,
Please enter a correct doctor ID : "))
                            print("----------------------------------------
--------------")
                            print("|For book an appointment for an exist
patient Enter 1|\n|"
                                  "For book an appointment for a new patient
Enter 2|\n|"
                                  "To be Back Enter B|")
                            print("----------------------------------------
--------------")
                            Admin_choice = input("Enter your choice : ")
                            Admin_choice = Admin_choice.upper()
                            if Admin_choice == "1":
                                patient_ID = int(input("Enter patient ID : "))
                                while patient_ID not in Pateints_DataBase:  #
if Admin entered incorrect ID
                                    patient_ID = int(input("Incorrect ID,
please Enter a correct patient ID : "))


                            elif Admin_choice == "2":
                                patient_ID = int(input("Enter patient ID : "))
                                while patient_ID in Pateints_DataBase:  # if
Admin entered used ID
```

```
                                    patient_ID = int(input("This ID is
unavailable, please try another ID : "))
                                Department = Doctors_DataBase[Doctor_ID][0][0]
                                DoctorName = Doctors_DataBase[Doctor_ID][0][1]
                                Name = input("Enter patient name    : ")
                                Age = input("Enter patient age     : ")
                                Gender = input("Enter patient gender  : ")
                                Address = input("Enter patient address : ")
                                RoomNumber = ""
                                Pateints_DataBase[patient_ID] = [Department,
DoctorName, Name, Age, Gender, Address,
                                                                 RoomNumber]

                            elif Admin_choice == "B":
                                break

                            Session_Start = input("Session starts at : ")
                            while Session_Start[:2] == "11" or
Session_Start[:2] == "12":
                                Session_Start = input("Appointments should be
between 01:00PM to 10:00PM,"
                                                      "Please enter a time
between working hours : ")

                            for i in Doctors_DataBase[Doctor_ID]:
                                if type(i[0]) != str:
                                    while Session_Start >= i[1] and
Session_Start < i[2]:
                                        Session_Start = input("This
appointment is already booked,"
                                                              "Please Enter an
other time for start of session : ")
                            Session_End = input("Session ends at : ")

                            New_Appointment = list()
                            New_Appointment.append(patient_ID)
                            New_Appointment.append(Session_Start)
                            New_Appointment.append(Session_End)
                            Doctors_DataBase[Doctor_ID].append(New_Appointment
)
                            print("/----------------------Appointment booked
successfully----------------------/")
                        except:
                            print("Doctor ID should be an integer number")

                    elif Admin_choice == "2":  # Admin mode --> Appointment
Management --> Edit an appointment
                        try:  # To avoid non integer input
```

```
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID not in Pateints_DataBase:
                                patient_ID = int(input("Incorrect Id, Please
Enter correct patient ID : "))
                            try:  # To avoid no return function
                                AppointmentIndex, PairKey =
AppointmentIndexInDoctorsDataBase(patient_ID)
                                Session_Start = input("Please enter the new
start time : ")
                                while Session_Start[:2] == "11" or
Session_Start[:2] == "12":
                                    Session_Start = input("Appointments should
be between 01:00PM to 10:00PM,"
                                                          "Please enter a time
between working hours : ")

                                for i in Doctors_DataBase[Doctor_ID]:
                                    if type(i[0]) != str:
                                        while Session_Start >= i[1] and
Session_Start < i[2]:
                                            Session_Start = input("This
appointment is already booked, "
                                                                  "Please
Enter an other time for start of session : ")
                                Session_End = input("Please enter the new end
time : ")
                                Doctors_DataBase[PairKey][AppointmentIndex] =
[patient_ID, Session_Start, Session_End]
                                print("/----------------------appointment
edited successfully----------------------/")
                            except:
                                print("No Appointment for this patient")
                        except:
                            print("Doctor ID should be an integer number")

                    elif Admin_choice == "3":  # Admin mode --> Appointment
Management --> Cancel an appointment
                        try:  # To avoid non integer input
                            patient_ID = int(input("Enter patient ID : "))
                            while patient_ID not in Pateints_DataBase:
                                patient_ID = int(input("Invorrect ID, Enter
patient ID : "))
                            try:
                                AppointmentIndex, PairKey =
AppointmentIndexInDoctorsDataBase(patient_ID)
                                Doctors_DataBase[PairKey].pop(AppointmentIndex
)
```

```
                                print("/----------------------appointment
canceled successfully----------------------/")
                            except:
                                print("No Appointment for this patient")
                        except:  # To avoid no return function
                            print("Patient ID should be an integer number")

                    elif Admin_choice == "B":  # Back
                        break

                    else:
                        print("please enter a correct choice")

                elif AdminOptions == "B":  # Back
                    break

                else:
                    print("Please enter a correct option")


            elif Password != "1234":
                if tries < 2:
                    Password = input("Password incorrect, please try again :
")
                    tries += 1
                else:
                    print("Incorrect password, no more tries")
                    tries_flag = "Close the program"
                    break

            Write_Hospital_Excel_Sheet.Write_Patients_DataBase(Pateints_DataBa
se)
            Write_Hospital_Excel_Sheet.Write_Doctors_DataBase(Doctors_DataBase
)


    elif Admin_user_mode == "2":  # User mode
        print("*****\n|         Welcome to user mode         |\n*****")
        while True:
            print("\n----------------------------------------")
            print("|To view hospital's departments Enter 1 |")
            print("|To view hospital's docotrs Enter 2     |")
            print("|To view patients' residents Enter 3    |")
            print("|To view patient's details Enter 4      |")
            print("|To view doctor's appointments Enter 5  |")
            print("|To be Back Enter B                     |")
            print("----------------------------------------")
            UserOptions = input("Enter your choice : ")
            UserOptions = UserOptions.upper()
```

```
            if UserOptions == "1":  # User mode --> view hospital's
departments
                print("Hospital's departments :")
                for i in Doctors_DataBase:
                    print(" " + Doctors_DataBase[i][0][0])

            elif UserOptions == "2":  # User mode --> view hospital's Doctors
                print("Hospital's doctors :")
                for i in Doctors_DataBase:
                    print(
                        "   " + Doctors_DataBase[i][0][1] + " in " +
Doctors_DataBase[i][0][0] + " department, from " +
                        Doctors_DataBase[i][0][2])

            elif UserOptions == "3":  # User mode --> view patients' residents
                for i in Pateints_DataBase:
                    print(" Patient : " + Pateints_DataBase[i][2] + " in " +
Pateints_DataBase[i][
                        0] + " department and followed by " +
Pateints_DataBase[i][1] + ", age : " +
                          Pateints_DataBase[i][3] + ", from : " +
Pateints_DataBase[i][5] + ", RoomNumber : " +
                          Pateints_DataBase[i][6])

            elif UserOptions == "4":  # User mode --> view patient's details
                try:  # To avoid non integer input
                    patient_ID = int(input("Enter patient's ID : "))
                    while patient_ID not in Pateints_DataBase:
                        patient_ID = int(input("Incorrect Id, Please enter
patient ID : "))
                    print(" patient name        : ",
Pateints_DataBase[patient_ID][2])
                    print(" patient age         : ",
Pateints_DataBase[patient_ID][3])
                    print(" patient gender      : ",
Pateints_DataBase[patient_ID][4])
                    print(" patient address     : ",
Pateints_DataBase[patient_ID][5])
                    print(" patient room number : ",
Pateints_DataBase[patient_ID][6])
                    print(" patient is in " + Pateints_DataBase[patient_ID][0]
+ " department")
                    print(" patient is followed by doctor : " + Pateints
DataBase[patient_ID][1])
                except:
                    print("Patient ID should be an integer number")
```

```
            elif UserOptions == "5":  # User mode --> view doctor's
appointments
                try:  # To avoid non integer input
                    Doctor_ID = int(input("Enter doctor's ID : "))
                    while Doctor_ID not in Doctors_DataBase:
                        Doctor_ID = int (input ("Incorrect Id, Please enter
doctor ID : "))
                    print (Doctors_DataBase[Doctor_ID][0][1] + " has
appointments :")
                    for i in Doctors_DataBase[Doctor_ID]:
                        if type(i[0]) == str:
                            continue
                        else:
                            print (" from: " + i[1] + "    to : " + i[2])
                except:
                    print ("Doctor ID should be an integer number")

            elif UserOptions == "B":  # Back
                break

            else:
                print ("Please Enter a correct choice")


    else:
        print ("Please choice just 1 or 2")
    finally:
        print ("hospital management system")
```

Scope

Hospitals and healthcare facilities have changed for the better. Healthcare administrations are using IT technologies to improve management and patient care on their hospital campuses. Examine some of the applications for managing hospitals' key attributes.

After installing hospital software, routine tasks like controlling admissions, keeping an eye on the blood bank, and registering patients may be completed more quickly and accurately. The hospital management software's modules are simple to use and provide easy access. It offers a standard, user-friendly interface with many modules. The utilization of the hospital management system can be maximized by the officials by incorporating these modules into their procedures without any problem.

Fig: 5 scopes of Hospital Management System

Testing

Testing for Integration

Before, during, and after the integration of a new module into the main software package, integration testing is performed. Every module of the code must be tested individually for this. A software application may have numerous modules, which are frequently written by a number of different programmers. Testing each module's impact on the overall program model is essential. Integration testing shows that the project is operating successfully.

Functional testing

Functional completeness testing is another name for functional testing. The goal of functional testing is to identify any potential gaps in functionality. During functional testing, testers may compile a list of additional features that a product could have to enhance it.

Software and Hardware Testing

Hardware and software testing is referred to as "HW/SW Testing" by IBM. During system testing, this is the time when the tester focuses on the interactions between the hardware and software.

```
from tkinter import *
from tkinter import ttk
import sqlite3
from tkinter import messagebox


window = Tk()
window.title("Hospital Management System")
window.geometry("1540x800+0+0")

lbltitle = Label(window, bd=20, relief=RIDGE, text="HOSPITAL MANAGEMENT
SYSTEM",
                 fg="red", bg="white", font=("times new roman", 50, "bold"))
lbltitle.pack(side=TOP, fill=X)

# data frame
Dataframe = Frame(window, bd=20, relief=RIDGE)
Dataframe.place(x=0, y=130, width=1530, height=400)

DataframeLeft = LabelFrame(Dataframe, bd=10, relief=RIDGE, padx=10,
                           font=("times new roman", 12, "bold"), text="Patient
Information")
DataframeLeft.place(x=0, y=5, width=980, height=350)

DataframeRight = LabelFrame(Dataframe, bd=10, relief=RIDGE, padx=10,
                            font=("times new roman", 12, "bold"),
text="Prescription")
DataframeRight.place(x=990, y=5, width=460, height=350)


# **********************button frame*********************

Buttonframe = Frame(window, bd=20, relief=RIDGE)
Buttonframe.place(x=0, y=530, width=1530, height=70)

# ****************Details frame***********

Detailsframe = Frame(window, bd=20, relief=RIDGE)
Detailsframe.place(x=0, y=600, width=1530, height=190)

# ************************DataframeLeft*************
lblNameTablet = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Names of Tablet", padx=2, pady=6)
lblNameTablet.grid(row=0, column=0, sticky=W)


comNameTablet = ttk.Combobox(DataframeLeft,  state="readonly", font=("arial",
12, "bold"),
                             width=33)
```

```
comNameTablet['value'] = (
    "Corona Vaccine", "Acetaminophen", "Adderall", "Amlodipine", "Ativan")
comNameTablet.current(0)
comNameTablet.grid(row=0, column=1)

lblref = Label(DataframeLeft, font=("arial", 12, "bold"),
               text="Reference No:", padx=2)
lblref.grid(row=1, column=0, sticky=W)
txtref = Entry(DataframeLeft, font=("arial", 13, "bold"),
               width=35)
txtref.grid(row=1, column=1)

lblDose = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Dose:", padx=2, pady=4)
lblDose.grid(row=2, column=0, sticky=W)
txtDose = Entry(DataframeLeft, font=("arial", 13, "bold"),
                width=35)
txtDose.grid(row=2, column=1)


lblNooftablets = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="No of Tablets", padx=2, pady=6)
lblNooftablets.grid(row=3, column=0, sticky=W)
txtNooftablets = Entry(DataframeLeft, font=(
    "arial", 13, "bold"),  width=35)
txtNooftablets.grid(row=3, column=1)

lblLot = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Lot", padx=2, pady=6)
lblLot.grid(row=4, column=0, sticky=W)
txtLot = Entry(DataframeLeft, font=("arial", 13, "bold"),
               width=35)
txtLot.grid(row=4, column=1)

lblissueDate = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Issue Date", padx=2, pady=6)
lblissueDate.grid(row=5, column=0, sticky=W)
txtissueDate = Entry(DataframeLeft, font=(
    "arial", 13, "bold"), width=35)
txtissueDate.grid(row=5, column=1)

lblExpDate = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="ExpDate", padx=2, pady=6)
lblExpDate.grid(row=6, column=0, sticky=W)
txtExpDate = Entry(DataframeLeft, font=(
    "arial", 13, "bold"), width=35)
txtExpDate.grid(row=6, column=1)
```

```
lblDailyDose = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="DailyDose", padx=2, pady=4)
lblDailyDose.grid(row=7, column=0, sticky=W)
txtDailyDose = Entry(DataframeLeft, font=(
    "arial", 13, "bold"), width=35)
txtDailyDose.grid(row=7, column=1)


lblSideEffect = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="SideEffect", padx=2, pady=6)
lblSideEffect.grid(row=8, column=0, sticky=W)
txtSideEffect = Entry(DataframeLeft, font=(
    "arial", 13, "bold"), width=35)
txtSideEffect.grid(row=8, column=1)


lblFurtherinfo = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Futherinformation", padx=2)
lblFurtherinfo.grid(row=0, column=2, sticky=W)
txtFurtherinfo = Entry(DataframeLeft, font=(
    "arial", 13, "bold"),  width=35)
txtFurtherinfo.grid(row=0, column=3)

lblBloodPressure = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Blood Pressure", padx=2, pady=6)
lblBloodPressure.grid(row=1, column=2, sticky=W)
txtBloodPressure = Entry(DataframeLeft, font=(
    "arial", 13, "bold"),  width=35)
txtBloodPressure.grid(row=1, column=3)


lblStorage = Label(DataframeLeft, font=("arial", 12, "bold"),
                   text="Storage Advice:", padx=2, pady=6)
lblStorage.grid(row=2, column=2, sticky=W)
txtStorage = Entry(DataframeLeft, font=(
    "arial", 12, "bold"),  width=35)
txtStorage.grid(row=2, column=3)


lblMedicine = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Medication", padx=2, pady=6)
lblMedicine.grid(row=3, column=2, sticky=W)
txtMedicine = Entry(DataframeLeft, font=(
    "arial", 12, "bold"), width=35)
txtMedicine.grid(row=3, column=3, sticky=W)


lblPatientId = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="PatientId", padx=2, pady=6)
```

```
lblPatientId.grid(row=4, column=2, sticky=W)
txtPatientId = Entry(DataframeLeft, font=(
    "arial", 12, "bold"), width=35)
txtPatientId.grid(row=4, column=3)


lblNhsNumber = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="NHS Number", padx=2, pady=6)
lblNhsNumber.grid(row=5, column=2, sticky=W)
txtNhsNumber = Entry(DataframeLeft, font=(
    "arial", 12, "bold"),  width=35)
txtNhsNumber.grid(row=5, column=3)


lblPatientname = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Patient Name", padx=2, pady=6)
lblPatientname.grid(row=6, column=2, sticky=W)
txtPatientname = Entry(DataframeLeft, font=(
    "arial", 12, "bold"),  width=35)
txtPatientname.grid(row=6, column=3)


lblDateOfBirth = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Date Of Birth", padx=2, pady=6)
lblDateOfBirth.grid(row=7, column=2, sticky=W)
txtDateOfBirth = Entry(DataframeLeft, font=(
    "arial", 12, "bold"),  width=35)
txtDateOfBirth.grid(row=7, column=3)


lblPatientAddress = Label(DataframeLeft, font=(
    "arial", 12, "bold"), text="Patient Address", padx=2, pady=6)
lblPatientAddress.grid(row=8, column=2, sticky=W)
txtPatientAddress = Entry(DataframeLeft, font=(
    "arial", 12, "bold"),  width=35)
txtPatientAddress.grid(row=8, column=3)


#
#============================    (DataframeRight)   ============================
============#

txtPrescription = Text(DataframeRight, font=(
    "arial", 12, "bold"), width=45, height=16, padx=2, pady=6)
txtPrescription.grid(row=0, column=0)


#================    (
Buttons)    ============================================================#
```

```
btnPrescription = Button(Buttonframe,  text="Prescription", bg="green",
fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnPrescription.grid(row=0, column=0)

btnPrescriptionData = Button(Buttonframe, text="Prescription Data",
bg="green", fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnPrescriptionData.grid(row=0, column=1)

btnUpdate = Button(Buttonframe, text="Update", bg="green", fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnUpdate.grid(row=0, column=2)

btnDelete = Button(Buttonframe,  text="Delete", bg="green", fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnDelete.grid(row=0, column=3)

btnClear = Button(Buttonframe,  text="Clear", bg="green", fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnClear.grid(row=0, column=4)

btnExit = Button(Buttonframe,  text="Exit", bg="green", fg="white", font=(
    "arial", 12, "bold"), width=23, height=16, padx=2, pady=6)
btnExit.grid(row=0, column=5)


scroll_x = ttk.Scrollbar(Detailsframe, orient=HORIZONTAL)
scroll_y = ttk.Scrollbar(Detailsframe, orient=VERTICAL)

hospital_table = ttk.Treeview(Detailsframe, column=("nameoftable", "ref",
"dose", "nooftablets", "lot", "issuedate", "expdate",
                                                    "dailydose", "storage",
"nhsnumber", "pname", "dob", "address"), xscrollcommand=scroll_x.set,
yscrollcommand=scroll_y.set)

scroll_x.pack(side=BOTTOM, fill=X)
scroll_y.pack(side=RIGHT, fill=Y)

scroll_x = ttk.Scrollbar(command=hospital_table.xview)
scroll_y = ttk.Scrollbar(command=hospital_table.yview)


hospital_table.heading("nameoftable", text="Name of Table")
hospital_table.heading("ref", text="Reference No. ")
hospital_table.heading("dose", text="Dose")
hospital_table.heading("nooftablets", text="No Of Tablets")
hospital_table.heading("lot", text="Lot")
```

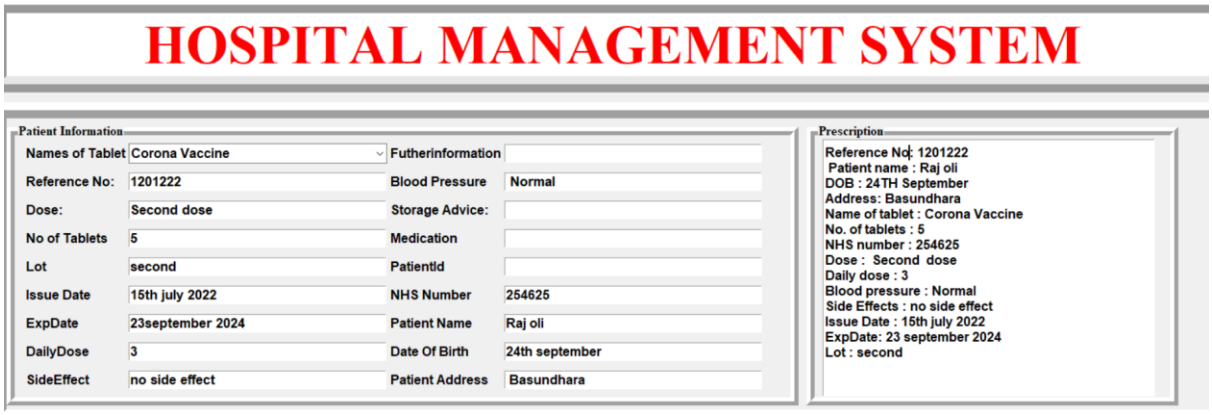
```
hospital_table.heading("issuedate", text="Issue Date")
hospital_table.heading("expdate", text="Exp Date")
hospital_table.heading("dailydose", text="Daily Dose")
hospital_table.heading("storage", text="Storage")
hospital_table.heading("nhsnumber", text="NHS Number")
hospital_table.heading("pname", text="Patient Name")
hospital_table.heading("dob", text="DOB")
hospital_table.heading("address", text="Address")

hospital_table["show"] = "headings"
hospital_table.column("nameoftable", width=100)
hospital_table.column("ref", width=100)
hospital_table.column("dose", width=100)
hospital_table.column("nooftablets", width=100)
hospital_table.column("lot", width=100)
hospital_table.column("issuedate", width=100)
hospital_table.column("expdate", width=100)
hospital_table.column("dailydose", width=100)
hospital_table.column("storage", width=100)
hospital_table.column("nhsnumber", width=100)
hospital_table.column("pname", width=100)
hospital_table.column("dob", width=100)
hospital_table.column("address", width=100)

hospital_table.pack(fill=BOTH, expand=1)




window.mainloop()
```



Conclusion

This project work has informed us. a) We now have a better understanding of how the HOSPITAL operates. This circumstance is representative of the real world. c) We now have a better knowledge of database design. this is due to the fact that database designing must be carefully followed in order to provide final results. c) A strong sense of time management is developed by scheduling a project and sticking to the timetable. d) A strong sense of teamwork and enhanced self-assurance in managing real-world projects have emerged. e) Initially, there were validation issues, but after conversations, we were able to implement validations.

References

Keywords used in Hospital Management System:

http://www.pharmatips.in/Articles/Hospital-Pharmacy/Hospital-Management-System.aspx

Scope Of Hospital Management System:

https://existek.com/blog/hospital-managment-system/

Tools and Technologies for hotel Management :

https://www.google.com/search?q=hospital+management+system+project+image+for+tools+AND+TECHNOLOGIES&sxsrf=ALiCzsb4d_CoCfggg8mLi18bvWaer757WQ:1659000953932&tbm=is

Classic waterfall Model:

https://www.google.com/search?q=hospital+management+system+project+image+for+tools+AND+TECHNOLOGIES&sxsrf=ALiCzsb4d_CoCfggg8mLi18bvWaer757WQ:1659000953932&tbm=is4Q

Individual report (Raj Oli)

Oracle Corporation creates, distributes, and supports MySQL. MySQL is a server-based database system that is used on the internet. Both small and large applications can benefit from MySQL. It is really quick, dependable, and simple to use. It supports conventional SQL. It is possible to compile MySQL on a variety of platforms. Tables are used by MySQL to store data. A table is a group of associated data, which is made up of rows and columns. Using databases to store information is beneficial. categorically.