

# **SANKETIKA VIDYA PARISHAD**

## **ENGINEERING COLLEGE**

**DEPARTMENT OF MASTER OF COMPUTER APPLICATION**



This is to certify that the course based project entitled “**HOSPITAL MANAGEMENT SYSTEM**” is a bona fide work done by **I.SAIKIRAN** (321232920019) in a particular fulfilment of the requirement for the award of degree “**MASTER OF COMPUTER APPLICATION** “ during the academic year **2021-2023**

# **SANKETIKA VIDYA PARISHAD**

## **ENGINEERING COLLEGE**

### **DEPARTMENT OF MASTER OF COMPUTER APPLICATION**



## **DECLARATION**

We here by declare that the project-based lab report entitled "**HOSPITAL MANAGEMENT SYSTEM**" has been prepared by us in a particular fulfilment of the requirement for the award of degree "**MASTER OF COMPUTER APPLICATION**" during the academic year **2021-2023**.

We also declare that this project - Based lab report is our own effort and it has not been submitted to any other university for the award of degree.

NAME OF THE STUDENT	REGISTRATION NUMBERS
I SAIKIRAN	321232920019
P.KANAKA SIVAJI	321232920044
M.SNEHITHA	321232920040
L.PREMA LATA	321232920030
Y.VASUDHA	321232920064

## **ABSTRACT**

The purpose of the project entitled as "Hospital Management System" is to computerize the Front Office Management of Hospital to develop software which is user friendly simple, fast, and cost-effective. It deals with the collection of patient's information like add patient, update patient, delete patient, search patient, view patient diagnosis, etc. Traditionally, it was done manually. The main function of the system is register and store patient details and doctor details and retrieve these details as and when required, and also to manipulate these details meaningfully. The Hospital Management System can be entered using a username and password. It is accessible by an Admin, Doctor & Receptionist. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast.

Government of India has still aimed at providing medical facilities by establishing hospital. The basic working of various hospitals in India is still on paper as compared to hospitals in European countries where computers have been put in to assist the hospital personals their work. The concept of automation of the administration and management of hospital is now being implemented in India also, with large hospitals like APPOLO and AIIMS in Delhi, ESCORTS in Chennai, having automated their existing system.

Computers are not only used to increase the efficiency in all fields ranging from fixing the appointment with the Doctor to keeping the record of the Patient.

## **ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. I owe a great many thanks to great many people who assisted me during and till the end of the project.

At the onset with great solemnity and sincerity ,I offer my profuse thanks to my project guide **P.T.S PRIYA , Assistant Professor** ,Department of MCA for guiding me all through my project work, giving right direction and shape to my learning .I am deeply motivated by her valuable guidance and kind cooperation throughout the making of the project.

I Express my profound gratitude to **DR.K.N.S LAKSHMI, Professor, Head of the Department**, Master of Computer Applications ,for giving me continuous inspiration and facilities to do this project work.

I Express my gratitude to all TEACHING STAFF and friends who supported me in preparing this project report.

**I SAIKIRAN**

<b><u>S.NO</u></b>	<b><u>TITLE</u></b>	<b><u>PAGE NO</u></b>
	DECLARATION	1
	ABSTRACT	2
	ACKNOWLEDGEMENT	3
1.	INTRODUCTION	7
1.1	Database Environment System	9
1.2	Advantages of Using DBMS Approach	10
1.3	Architecture of Database	11
2	INTRODUCTION TO PROJECT	13
2.1	Brief Description	13
2.2	Scope	14
2.3	Table Description	15
2.3.1	Patient Info	15
2.3.2	Doctor's Info	15
2.3.3	Test Info	16
2.3.4	Pharmacy Info	16
2.3.5	Employee Info	16
2.3.6	Check Info	17
2.3.7	Cafeteria Info	17
2.4	Triggers	18
3	DESIGN	20
3.1	E-R Diagram	20
3.2	Normalization	21

3.2.1	First Normal Form(1NF)	21
3.2.2	Second Normal Form(2NF)	21
3.2.3	Third Normal Form(3NF)	21
3.3	Schema Diagram	22
4.	HARDWARE AND SOFTWARE	23
4.1	Hardware Requirements	23
4.2	Software requirements	23
4.2.1	Server side	23
4.2.2	Client side	23
4.2.1.1	PHP	23
4.2.1.2	Web Server: Apache	24
4.2.2.1	HTML	24
4.2.2.2	JavaScript24	24
5	COMMANDS	25
5.1	DDL	25
5.2	DML	26
5.3	DQL	28
6	VIEWS	29
	CONCLUSION	30
	APPENDIX 'A' – CODE SNIPPETS	
A.1	Database Connection	31
A.2	Insert Query	31
A.3	Select Query	31
A.4	Update Query	32

## **APPENDIX 'B' – SCREENSHOTS**

<b>B.1</b>	<b>Home Page</b>	<b>33</b>
<b>B.2</b>	<b>PATIENT REG. FORM</b>	<b>33</b>
<b>B.3</b>	<b>ADMIN LOGIN PAGE</b>	<b>34</b>
<b>B.4</b>	<b>PATIENT ADMISSION</b>	<b>34</b>
<b>B.5</b>	<b>PATIENT RECORD SEARCH</b>	<b>35</b>
<b>B.6</b>	<b>PATIENT BILL</b>	<b>35</b>
<b>B.7</b>	<b>PATIENT REPORTS</b>	<b>36</b>
<b>B.8</b>	<b>PATIENT BILLS</b>	<b>36</b>

# **1.INTRODUCTION TO DATABASE**

A database is a collection of inter-related data which helps in the efficient retrieval, insertion, and deletion of data from the database and organizes the data in the form of tables, views, schemas, reports, etc. For Example, a university database organizes the data about students, faculty, admin staff, etc. which helps in the efficient retrieval, insertion, and deletion of data from it. DDL is the short name for Data Definition Language, which deals with database schemas and descriptions, of how the data should reside in the database.

- **CREATE:** to create a database and its objects like (table, index, views, store procedure, function, and triggers)
- **ALTER:** alters the structure of the existing database
- **DROP:** delete objects from the database
- **TRUNCATE:** remove all records from a table, including all spaces allocated for the records are removed
- **COMMENT:** add comments to the data dictionary
- **RENAME:** rename an object

DML is the short name for Data Manipulation Language which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

- **SELECT:** retrieve data from a database
- **INSERT:** insert data into a table
- **UPDATE:** updates existing data within a table
- **DELETE:** Delete all records from a database table
- **MERGE:** UPSERT operation (insert or update)
- **CALL:** call a PL/SQL or Java subprogram
- **EXPLAIN PLAN:** interpretation of the data access path
- **LOCK TABLE:** concurrency Control

DCL is short for Data Control Language which acts as an access specifier to the database.(basically to grant and revoke permissions to users in the database



- **GRANT:** grant permissions to the user for running DML(SELECT, INSERT, DELETE,...) commands on the table
- **REVOKE:** revoke permissions to the user for running DML(SELECT, INSERT, DELETE,...) command on the specified table

### **Database Management System:**

The software which is used to manage databases is called Database Management System (DBMS). For Example, MySQL, Oracle, etc. are popular commercial DBMS used in different applications. DBMS allows users the following tasks:

#### ➤ **Data Definition:**

It helps in the creation, modification, and removal of definitions that define the organization of data in the database.

Data Updation: It helps in the insertion, modification, and deletion of the actual data in the database.

#### ➤ **Data Retrieval:**

It helps in the retrieval of data from the database which can be used by applications for various purposes.

#### ➤ **User Administration:**

It helps in registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information corrupted by unexpected failure.

### **Paradigm Shift from File System to DBMS**

File System manages data using files on a hard disk. Users are allowed to create, delete, and update the files according to their requirements. Let us consider the example of file-based University Management System. Data of students is available to their respective Departments, Academics Section, Result Section, Accounts Section, Hostel Office, etc. Some of the data is common for all sections like Roll No, Name, Father Name, Address, and Phone number of students but some data is available to a particular section only like Hostel allotment number which is a part of the hostel office. Let us discuss the issues with this system:

#### ➤ **Redundancy of data:**

Data is said to be redundant if the same data is copied at many places. If a student wants to change their Phone number, he or has to get it updated in

various sections. Similarly, old records must be deleted from all sections representing that student.

➤ **Inconsistency of Data:**

Data is said to be inconsistent if multiple copies of the same data do not match each other. If the Phone number is different in Accounts Section and Academics Section, it will be inconsistent. Inconsistency may be because of typing errors or not updating all copies of the same data.

➤ **Difficult Data Access:**

A user should know the exact location of the file to access data, so the process is very cumbersome and tedious. If the user wants to search the student hostel allotment number of a student from 10000 unsorted students' records, how difficult it can be.

➤ **Unauthorized Access:**

File Systems may lead to unauthorized access to data. If a student gets access to a file having his marks, he can change it in an unauthorized way.

➤ **No Concurrent Access:**

The access of the same data by multiple users at the same time is known as concurrency. The file system does not allow concurrency as data can be accessed by only one user at a time.

➤ **No Backup and Recovery:**

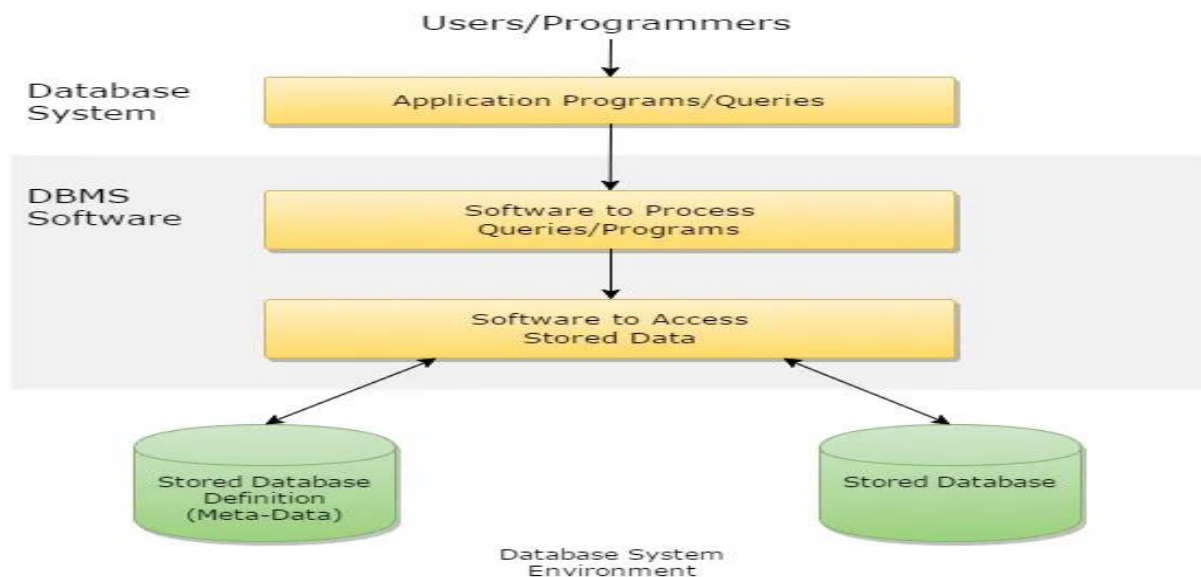
The file system does not incorporate any backup and recovery of data if a file is lost or corrupted.

## **1.1 DATABASE ENVIRONMENT SYSTEM :**

One of the primary aims of a database is to supply users with an abstract view of data, hiding a certain element of how data is stored and manipulated. Therefore, the starting point for the design of a database should be an abstract and general description of the information needs of the organization that is to be represented in the database. And hence you will require an environment to store data and make it work as a database. In this chapter, you will learn about the database environment and its architecture.

environment is a collective system of components that comprise and regulates the group of data, management, and use of data, which consist of software, hardware, people, techniques of handling database, and the data also.

Here, the hardware in a database environment means the computers and computer peripherals that are being used to manage a database, and the software means the whole thing right from the operating system (OS) to the application programs that include database management software like M.S. Access or SQL Server. Again the people in a database environment include those people who administrate and use the system. The techniques are the rules, concepts, and instructions given to both the people and the software along with the data with the group of facts and information positioned within the database environment.



## 1.2 ADVANTAGES OF USING DBMS APPROCH

Some of them are given as follows below

### 1 Better Data Transferring:

Database management creates a place where users have an advantage of more and better-managed data. Thus making it possible for end-users to have a quick look and to respond fast to any changes made in their environment.

### 2 Better decision making:

Due to DBMS now we have Better managed data and Improved data access because of which we can generate better quality information hence on this basis better decisions can be made. Better Data quality improves accuracy,

validity, and time it takes to read data. DBMS does not guarantee data quality, it provides a framework to make it easy to improve data quality.

### **3 Better data integration:**

Due to the Database Management System we have an access to well managed and synchronized form of data thus it makes data handling very easy and gives an integrated view of how a particular organization is working and also helps to keep a track of how one segment of the company affects another segment.

### **4 Faster data Access:**

The Database management system (DBMS) helps to produce quick answers to database queries thus making data access faster and more accurate. For example, to read or update the data. For example, end-users, when dealing with large amounts of sale data, will have enhanced access to the data, enabling a faster sales cycle. Some queries may be like:

- What is the increase in sales in the last three months?
- What is the bonus given to each of the salespeople in the last five months?
- How many customers have a credit score of 850 or more?

### **5 Increased end-user productivity:**

The data which is available with the help of a combination of tools that transform data into useful information, helps end-users to make quick, informative, and better decisions that can make difference between success and failure in the global economy.

### **6 Simple:**

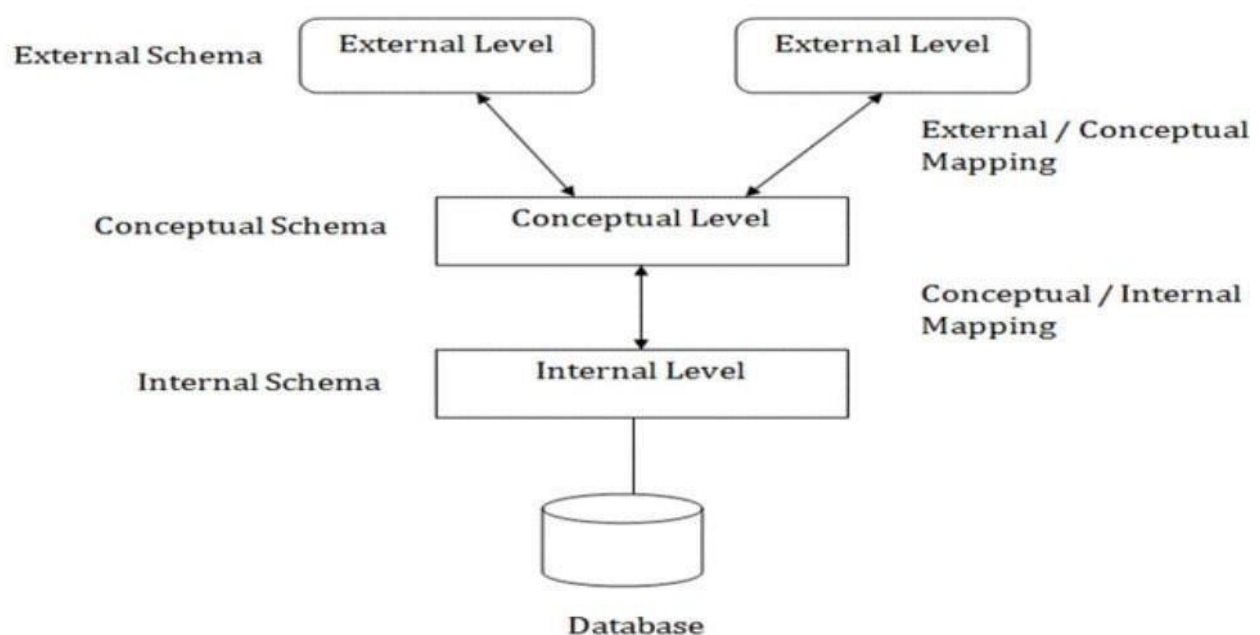
Database management system (DBMS) gives a simple and clear logical view of data. Many operations like insertion, deletion, or creation of files or data are easy to implement.

## **1.3 ARCHITECTURE OF DBMS**

### **The Three-Schema Architecture**

The goal of the three-schema architecture illustrated in the figure is to separate the user application from the physical database. In this architecture, schemas can be defined at the following three levels:

- The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.
- The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.
- The external or view level includes a number of external schemas or user views. Each external schema describes the part of a database that a particular user group is interested in and hides the rest of the database from that user group.



**Fig 1.2: Architecture of DBMS**

## **2. INTRODUCTION TO PROJECT**

### **HOSPITAL MANAGEMENT SYSTEM**

#### **2.1 Brief description**

- The mini-project entitled "Hospital Management System " is developed as apart of the Second semester DBMS laboratory, for the partial fulfillment of the requirement for the BE( Information Science) course.
- Our website has various kinds of information that helps regarding booking Appointment to the Doctor's
- Patients will be able to search the doctor's availability ,the exact solutions ,the availability and treatment of the disease and they can also buying the pharmacy by using the prescription and after buying the medicines if the patient must to check it how to use the medicine then they can easily how to know use medicine in doctor's

#### **Objectives**

The objective of the online Doctor's Dairy&Billing System Project is to design software to fully automate the process of issuing online appointment.

That is:-

- 1.To create a database of the Hospital
- 2.To search the doctor's arrival and register patient for booking the OP for doctor's,pharmacy
- 3.To check the availability of the OP
- 4.To calculate fare
- 5.To book the OP pass

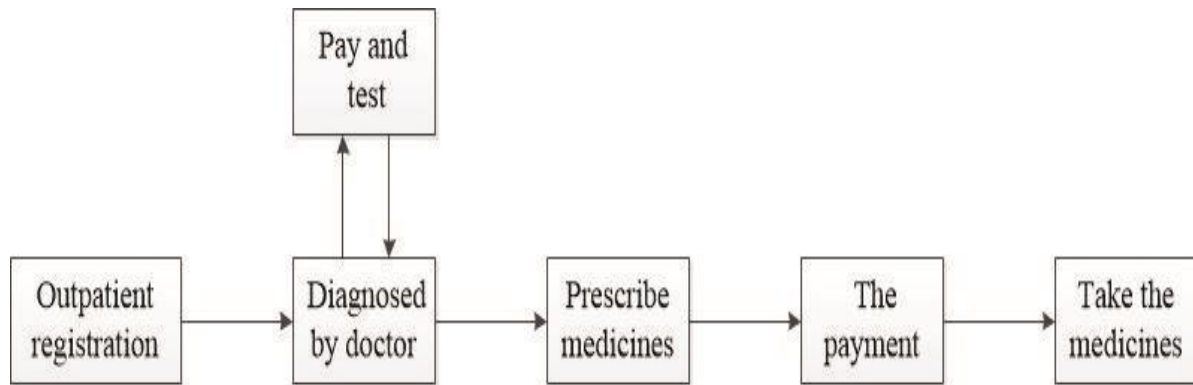
6.To cancel the OP if necessary

## 2.2 Scope

- Patient registration
- Appointment scheduling
- Billing and payments(cash,CC,insurance)
- Security of the whole system
- Pharmaceutical drugs/equipment
- Staff management(work roster,availability,scheduling,etc)
- Management functions (report generations,accounting,etc)
- System administration
- Resource allocation(booking rooms,operation theatres,etc)
- Comprehensive database
- Web interface(proposed for future)

### Analysis:

We can take the outpatient service as an example, analyzed the flow based on UML model. The outpatient management includes some subsystems, such as outpatient registration subsystem, outpatient pharmacy subsystem and outpatient fee subsystem (Lee et al., 2013). Different subsystems have different functions. The registration subsystem (Lee et al., 2013) is related with registration process, registration number, information of the registration, providing expert information and so on. The outpatient pharmacy subsystem is related with medical supplies, medical information, etc. The outpatient fee subsystem is mainly responsible for integrating the price, confirming the charges, entering the related requisition, providing invoices and receipts and so on.



The diagram of patients' registration processes.

## 2.3 Table description

### 2.3.1 PATIENT INFO:

PATIENT INFO table has attributes patient no,name,age,sex,address and deposite currency,reg date,dis date,remarks,etc shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	<u>patientNo</u>	int(10)	PRIMARY KEY
2	PatientName	VARCHAR(20)	REFERENCES employee (empID)
3	Age	int(5)	-
4	Gender	VARCHAR(6)	-
5	Address	text	-
6	Deposite currency	int(10)	NOT NULL
7	regDate	DATE	-
8	disDate	DATE	-
9	Remarks	VARCHAR(25)	-
10	empID	int(6)	NOT NULL

TABLE A:STRUCTURE OF PATIENT INFO

### 2.3.2 DOCTOR INFO:

DOCTOR INFO has attributes docID,docName,Address,Contact,Faculty shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	docID	int(5)	PRIMARY KEY
2	docName	VARCHAR(20)	NOT NULL
3	Address	VARCHAR(50)	-
4	Contact	int(10)	CHECK(contact> 10)
5	Faculty	VARCHAR(100)	-

TABLE B:STRUCTURE OF DOCTOR INFO



### 2.3.3 TEST INFO:

TEST INFO table has attributes patientNo,Testdate,Testhead,Amount,Remarks shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	patientNo	int(6)	FORIGEN KEY(patientNo) REFERENCES Patient (patientNo) NOT NULL
2	Testdate	date	-
3	Testhead	VARCHAR(100)	-
4	Amount	currency	-
5	Remarks	text	-

**TABLE C:STRUCTURE OF TEST INFO**

### 2.3.4 PHARMACY INFO:

PHARMACY INFO TABLE has attributes PatientNo,Buydate,Particulers,Rate,Quantity,etc shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	PatientNo	int(5)	REFERENCES patient(patientNo)
2	Buydate	date	NOT NULL
3	Particulars	VARCHAR(100)	-
4	Rate	currency	-
5	Quantity	int(30)	-

**TABLE D:STRUCTURE OF PHARMACY INFO**

### 2.3.5 EMPLOYEE INFO:

EMPLOYEE TABLE has Attributes empID,empName,CounterNo,etc shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	empID	int(6)	PRIMARY KEY
2	empName	VARCHAR(20)	NOT NULL
3	CounterNo	int(5)	PRIMARY KEY

**TABLE E:STRUCTURE OF EMPLOYEE INFO**

### 2.3.6 CHECK INFO:

CHECK TABLE has Attributes DocID,patientNo,checkDate,fee,remarks,etc shown in table below

S.NO	NAME	TYPE	CONSTRAINT
1	docID	int(7)	FOREIGN KEY (docID) REFERENCES doctor (docID),NOT NULL
2	patientNo	int(6)	FOREIGN KEY (patientNo) REFERENCES patient,NOT NULL
3	checkDate	Date	-
4	Fee	currency	-
5	reamarks	text	-

**TABLE F:STRUCTURE OF CHECK INFO**

### 2.3.7 CAFETERIA INFO:

CAFETERIA TABLE has Attributes patientNo,orderdate,particulars,rate,qty,et as shown below table

S.NO	NAME	TYPE	CONSTRAINTS
1	patientNo	int(5)	FOREIGN KEY (patientNo) REFERENCES patient(patientNo)
2	orderdate	Date	NOT NULL
3	particulars	VARCHAR(100)	-
4	rate	currency	-
5	qty	int(10)	-

**TABLE G:STRUCTURE OF CAFETERIA INFO**

## 2.5 TRIGGERS:

A trigger is a special type of stored procedure that automatically executes when a event occurs in the database server.

```

BEGIN
EXECUTE IMMEDIATE 'DROP SEQUENCE
SEQ':EXCEPTION
WHEN OTHERS THEN IF SQLCODE!= -2289 THEN
RAISE; END IF;
END;
CREATE SEQUENCE BILL_NO_SEQ
MINVALUE 1
MAXVALUE 100000
START WITH 1
INCREMENT BY 1;
CREATE OR REPLACE TRIGGER BILL_TRG BEFORE INSERT ON BILL
FOR EACH ROW
DECLARE DOC_ID VARCHAR(10 CHAR);
BEGIN
IF NEW.BILL_NO IS NULL THEN
SELECT BILL_NO_SEQ.NEXTVAL INTO :NEW.BILL_NO FROM DUAL;
END IF;
IF NEW.PAT_NAME IS NULL THEN SELECT PAT_NAME INTO NEW. PAT NAME
FROM
PATIENT WHERE PAT_ID=: NEW.PAT_ID;
END IF;
IF NEW.PAT GENDER IS NULL THEN
SELECT PAT_GENDER INTO NEW. PAT_GENDER FROM PATIENT WHERE
PAT_ID=:NEW.PAT_ID;
END IF;
IF NEW.PAT_ADDRESS IS NULL THEN SELECT ADDRESS INTO NEW.
PAT_ADDRESS
FROM PATIENT WHERE PAT_ID=: NEW. PAT_ID;
END IF; IF NEW.DOC_NAME IS NULL THEN
SELECT PAT_DOC_CODE INTO DOC_ID FROM
BEGIN
EXECUTE IMMEDIATE 'DROP SEQUENCE

```

```
SEQ':  
EXCEPTION  
WHEN OTHERS THEN IF SQLCODE!= -2289 THEN  
RAISE; END IF;  
END;  
CREATE SEQUENCE BILL_NO_SEQ  
MINVALUE 1  
MAXVALUE 100000  
START WITH 1  
INCREMENT BY 1;  
CREATE OR REPLACE TRIGGER BILL_TRG BEFORE INSERT ON BILL  
FOR EACH ROW  
DECLARE DOC_ID VARCHAR(10 CHAR);  
BEGIN  
IF NEW.BILL_NO IS NULL THEN  
SELECT BILL_NO_SEQ.NEXTVAL INTO :  
NEW.BILL_NO FROM DUAL;  
END IF;  
IF NEW.PAT_NAME IS NULL THEN SELECT PAT_NAME INTO NEW. PAT NAME  
FROM  
PATIENT WHERE PAT_ID=: NEW.PAT_ID;  
END IF;  
IF NEW.PAT GENDER IS NULL THEN  
SELECT PAT_GENDER INTO NEW. PAT_GENDER FROM PATIENT WHERE  
PAT_ID=:NEW.PAT_ID;  
END IF;  
IF NEW.PAT_ADDRESS IS NULL THEN SELECT ADDRESS INTO NEW.  
PAT_ADDRESS  
FROM PATIENT WHERE PAT_ID=: NEW. PAT_ID;
```

## 3 DESIGN

### 3.1 ER-Diagram:

An entity–relationship model describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types and specifies relationships that can exist between instances of those entity types.

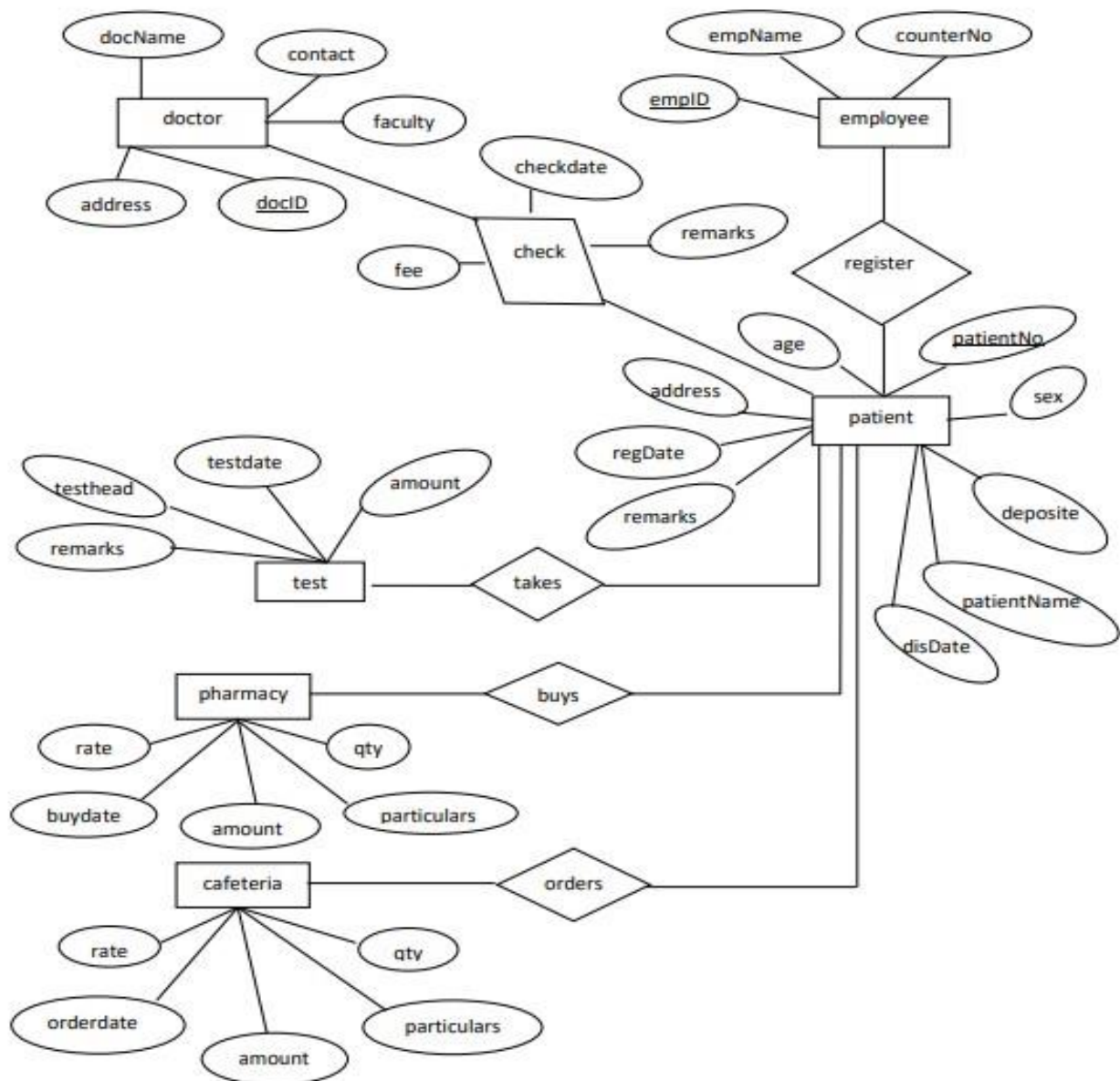


Figure 3.1 E-R DIAGRAM

This ER Diagram gives a brief idea about the relations existing between the tables and tells about the primary and the foreign keys being used in this database

## **3.2 NORMALIZATION:**

### **3.2.1 First Normal Form (1NF):**

If a relation contain composite or multi-valued attribute, it violates first normal form or a relation is in first normal form if it does not contain any composite or multi-valued attribute. A relation is in first normal form if every attribute in that relation is singled valued attribute.

### **3.2.2 Second Normal Form (2NF):**

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has **No Partial Dependency**, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

**Partial Dependency** – If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.

### **3.2.3 Third Normal Form (3NF):**

A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.

A relation is in 3NF if at least one of the following condition holds in every non-trivial function dependency  $X \rightarrow Y$

1. X is a super key.
2. Y is a prime attribute (each element of Y is part of some candidate key).

**Transitive dependency** – If  $A \rightarrow B$  and  $B \rightarrow C$  are two FDs then  $A \rightarrow C$  is called transitive dependency.

3.3 SCHEMA DIAGRAM:

A database schema can be represented in a visual diagram, which shows the database object and their relationship which represents the logical view of the database and how the relationships among them are represented.

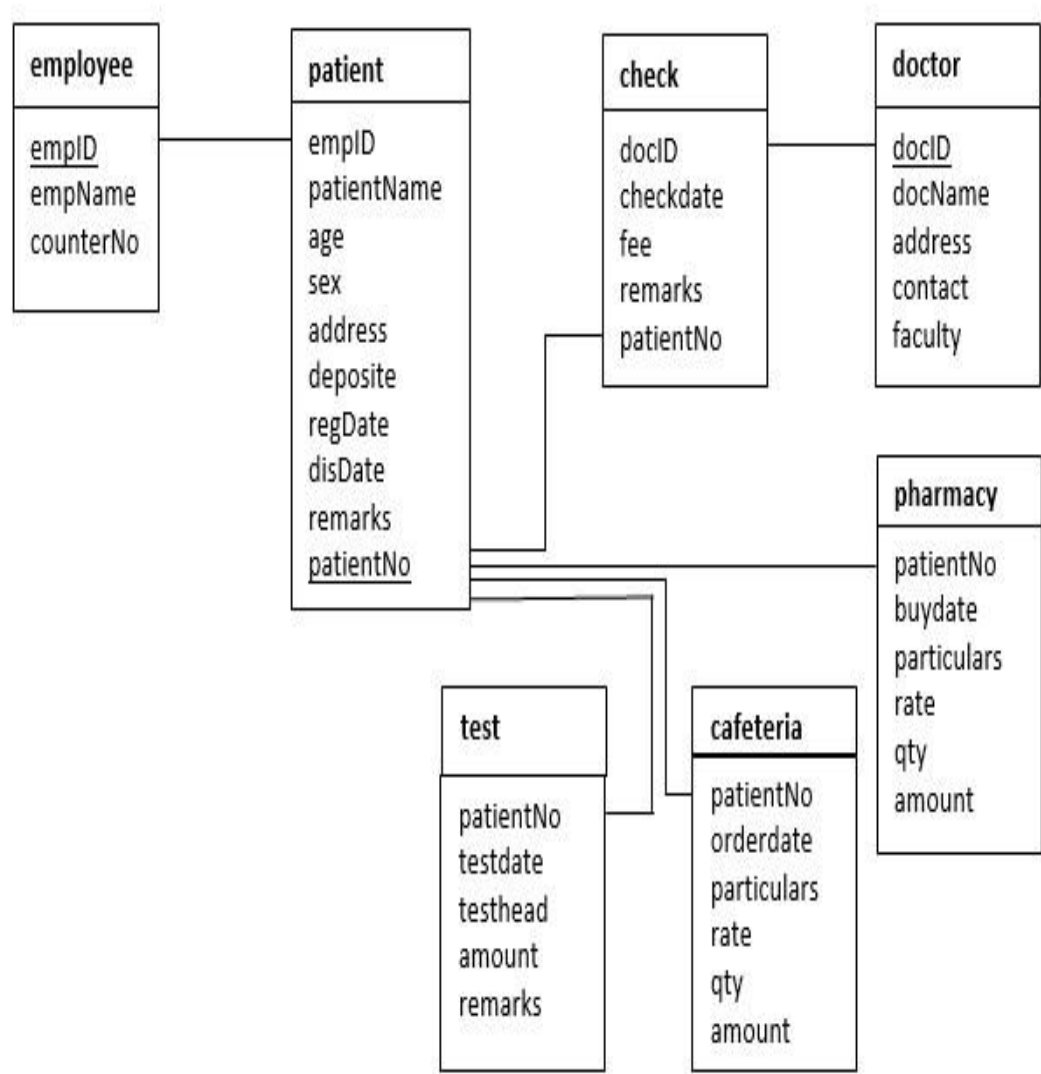


Figure 3.2 SCHEMA DIAGRAM

This Schema Diagram in Fig 3.2 represents different tables used

## 4 HARDWARE AND SOFTWARE REQUIREMENTS

### 4.1 Hardware requirement:

- A desktop or laptop with a proper internet connection
- 2 500GB or 60GB of the hard disk
- 3.4GB 2GB of the RAM
- 4 Windows 7 or 8 or 10 Operating system

### 4.2 Software requirements:

#### 4.2.1 Server side

1. Programming language: PHP 5.6.31
2. Web Server: Apache 2.4.27
3. Database: SQL 5.7.19



#### 4.2.2 Client side

1. Programming language: JAVASCRIPT, HTML, CS
2. OS: windows7/8/10
3. MYSQL server

##### 4.2.1.1 PHP



PHP is a server-side scripting language designed primarily for web development but also used as a general programming language. PHP code may be embedded into HTML or HTML5 markup or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated webpage.



#### 4.2.1.2 WEB SERVER: APACHE



Apache is the most widely used web server software. Developed and maintained by Apache Software Foundation, Apache is open-source software available for free. It runs on 67% of all web servers in the world. It is fast, reliable, and secure. It can be highly customized to meet the needs of many different environments by using extensions and modules. Most WordPress hosting providers use Apache as their web server software. However, WordPress can run on other web server software as well.

#### 4.2.2.1 HTML



HTML is an acronym that stands for HyperText Markup Language. HyperText: HyperText simply means "Text within Text". A text has a link within it, is a hypertext. Every time you click on a word that brings you to a new webpage, you have clicked on a hypertext.

Markup language: A markup language is a programming language that is used to make text more interactive and dynamic. It can turn a text into images, tables, links, etc. An HTML document is made of many HTML tags and each HTML tag contains different content

#### 4.2.2.2 JAVASCRIPT



Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is

## 5 COMMANDS

### **5.1 DDL COMMANDS:**

#### **For table "employee"**

```
CREATE TABLE employee
(
    empID int NOT NULL,empName varchar(100) NOT NULL,counterNo int NOT NULL,
    constraint pk_empID PRIMARY KEY (empID)
)
```

#### **For table "patient"**

```
CREATE TABLE patient
(
    patientNo int NOT NULL,PatientName varchar(100) NOT NULL,age int,sex char,address
    varchar(100),deposite currency NOT NULL,regDate date,disDate date,remarks text,empID
    int NOT NULL,CONSTRAINT pk_patientNo PRIMARYKEY (patientNo),CONSTRAINT
    fk_empID FOREIGNKEY (empID) REFERENCES employee (empID)
)
```

#### **For table "doctor"**

```
CREATE TABLE doctor
(
    docID int NOT NULL,docName varchar(100) NOT NULL,address varchar(100),contact
    int(10),faculty varchar(100),CONSTRAINT pk_docID PRIMARY KEY (docID)
)
```

#### **For table "check"**

```
CREATE TABLE check
(
    docID int NOT NULL,patientNo int NOT NULL,checkDate date,fee currency,remarks text,
    CONSTRAINT fk_docID FOREIGN KEY (docID) REFERENCES doctor (docID),CONSTRAINT
    fk1_patientNo FOREIGN KEY (patientNo) REFERENCES patient (patientNO)
)
```

)

### **For table "pharmacy"**

CREATE TABLE pharmacy

(

patientNo int NOT NULL ,buydate date NOT NULL,particulars varchar(100),rate currency,  
qty int,amount currency,CONSTRAINT ph\_patientNo FOREIGN KEY (patientNo)

REFERENCES patient (patientNo)

)

### **For table "cafeteria"**

CREATE TABLE cafeteria

(

patientNo int NOT NULL,orderdate date NOT NULL,particulars varchar(100),rate currency,  
qty int,amount currency,CONSTRAINT caf\_patientNo FOREIGN KEY (patientNo)

REFERENCES patient (patientNo),

)

### **For table "test"**

CREATE TABLE test

(

patientNo int NOT NULL,testdate date,testhead varchar(100),amount currency,remarks  
text,CONSTRAINT test\_patientNo FOREIGNKEY (patientNo) REFERENCES patient

(patientNo)

)

## **5.2 DML COMMANDS**

### **INSERT:**

#### **Employee:**

INSERT INTO employee (

(empID int(4)NOT NULL,empName varchar(100) NOT NULL,counterNo int NOT NULL)

VALUES(544,vikram,7));

```
INSERT INTO employee (  
(empID int(4)NOT NULL,empName varchar(100) NOT NULL,counterNo int NOT NULL)  
VALUES(602,sanju,8));
```

```
INSERT INTO employee (  
(empID int(4)NOT NULL,empName varchar(100) NOT NULL,counterNo int NOT NULL)  
VALUES(478,vatsav,9));
```

```
INSERT INTO employee (  
(empID int(4)NOT NULL,empName varchar(100) NOT NULL,counterNo int NOT NULL)  
VALUES(428,naveen,12));
```

### **Patient:**

```
INSERT TABLE patient (  
(patientNo int(4) NOT NULL,PatientName varchar(100) NOT NULL,age int,sex char,address  
varchar(100),deposit currency NOT NULL,regDate date,disDate date,remarks text,empID  
int NOT NULL)  
VALUES(23,rajiv,34,male,hyderabad,15000,10-3-2021,17-4-2021,428));
```

```
INSERT TABLE patient (  
(patientNo int(4) NOT NULL,PatientName varchar(100) NOT NULL,age int,sex char,address  
varchar(100),deposit currency NOT NULL,regDate date,disDate date,remarks text,empID  
int NOT NULL)  
VALUES(26,sahasra,29,female,guntur,4500,14-6-2022,22-7-2022,478));
```

```
INSERT TABLE patient (  
(patientNo int(4) NOT NULL,PatientName varchar(100) NOT NULL,age int,sex char,address  
varchar(100),deposit currency NOT NULL,regDate date,disDate date,remarks text,empID  
int NOT NULL)  
VALUES(32,ramesh,34,male,delhi,5000,22-2-2022,21-3-2022,544));
```

### **Doctor:**

```
CREATE TABLE doctor(  
(docID int NOT NULL,docName varchar(100) NOT NULL,address varchar(100),contact  
int(10),faculty varchar(100)  
VALUES (567,Rohit,Mumbai,9875743150,ENT));
```

```
CREATE TABLE doctor(  
(docID int NOT NULL,docName varchar(100) NOT NULL,address varchar(100),contact  
int(10),faculty varchar(100)  
VALUES (567,Rohit,Mumbai,9875743150,ENT));
```

```
(docID int NOT NULL,docName varchar(100) NOT NULL,address varchar(100),contact  
int(10),faculty varchar(100)  
VALUES (428,Narendra,delhi,787633150,Orthopedic));
```

### **UPDATE:**

#### **EMPLOYEE:**

```
UPDATE employee SET empName='rahul' WHERE counterNo=7;
```

```
UPDATE employee SET counterNo='15' WHERE empID=428;
```

#### **PATIENT:**

```
UPDATE patient SET regDate='3-6-2022' WHERE
```

#### **DOCTOR:**

```
UPDATE doctor SET address='canada' WHERE docID=428;
```

## **5.3 DQL COMMANDS**

### **SELECT:**

#### **PATEINT:**

```
SELECT *FROM patient;
```

#### **DOCTOR:**

```
SELECT *FROM doctor;
```

#### **EMPLOYEE:**

```
SELECT *FROM employee;
```

## 6 VIEWS

```
CREATE VIEW Employee_View AS  
SELECT empID,empName,counterNo  
FROM employee Report;
```

S.NO	NAME	TYPE
1	empID	Int()
2	empName	VARCHAR()
3	counterNo	Int()

```
SELECT *FROM Employee_View
```

s.no	empID	empName	counterNo
1	602	sanju	8
2	544	Rahul	7
3	428	naveen	15

## **CONCLUSION**

The project is focused towards creating an integrated billing system for hospital's patient. It is partially used in different hospitals right at now, but the combined / integrating database system has not be fully developed. It is not imaginary project and holds real value if developed sincerely. In our country where database system is at it's infant phase, projects as such is a good sign of striving to move forward. The use of database at hospitals, especially in the government hospitals will surely make it more efficient and minimizes the errors that occur from traditional papering system. It is also a research tool for researchers, and it also helps to identify the subtle health epidemics.

## APPENDIX 'A'- CODE SNIPPETS

### A.1 DATABASE CONNECTION:

The connect() / mysqli\_connect() function opens a new connection to the MySQL server with the following syntax :

mysqli\_connect(host, username, password, dbname, port, socket)

```
mysqli_select_db($conn,"$db_name") or die("cannot select db");
```

FIG A.1 DATABASE CONNECTION

### A.2 INSERT QUERY:

This query is used to insert a booking

```
$sql="INSERT INTO $tbl_name(pfname, plname, pid, doj, DOB, age, gender, address, occupation, disease)
VALUES ('$fname', '$plname', '$pid', '$doj', '$dob', '$age', '$gender', '$address', '$occupation', '$disease')";
$result=$conn->query($sql);
echo "$sql<br>";
if(!$result) die ($conn->error);
```

FIG A.2 INSERT QUERY

### A.3 SELECT QUERY:

In this query, all the details are fetched using SELECT\* command

```
$pname=$_SESSION['name'];
$pid=$_GET['pid'];
$doj=$_GET['doj'];
$DOB=$_GET['DOB'];
$age=$_GET['age1'];
$gender=$_GET['gender1'];
$address=$_GET['address'];
$occupation=$_GET['occuption'];
$disease=$_GET['disease'];

echo " , , $.pid , , $.pname , , $.age , , $.gender , , $.seat , , ";
$sql1="*SELECT*.$seat."from seats_availability where patient_No='".$pno,'"and doj='".$doj.'"";
$result1=$conn->query($sql1);
```

FIG A.3 SELECT QUERY



## A.4 UPDATE QUERY

Here the update query is called to update the user of an already existing based on its name and category id, address etc respectively.

```
$sql="UPDATE $tbl_name SET l_name='$lname',email='$mail',gender='$gender',marital='$marital',dob='$dob',mobile='$mobile',ques='$ques',WHERE  
f_name='$fname';  
$result=mysql_query($conn,$sql);  
  
$SESSION['error'];
```

FIG A.4 UPDATE QUERY

## APPENDIX 'B'- SCREENSHOTS

### B.1 HOME PAGE:

This is the first window when the application is executed as shown in Fig B.1

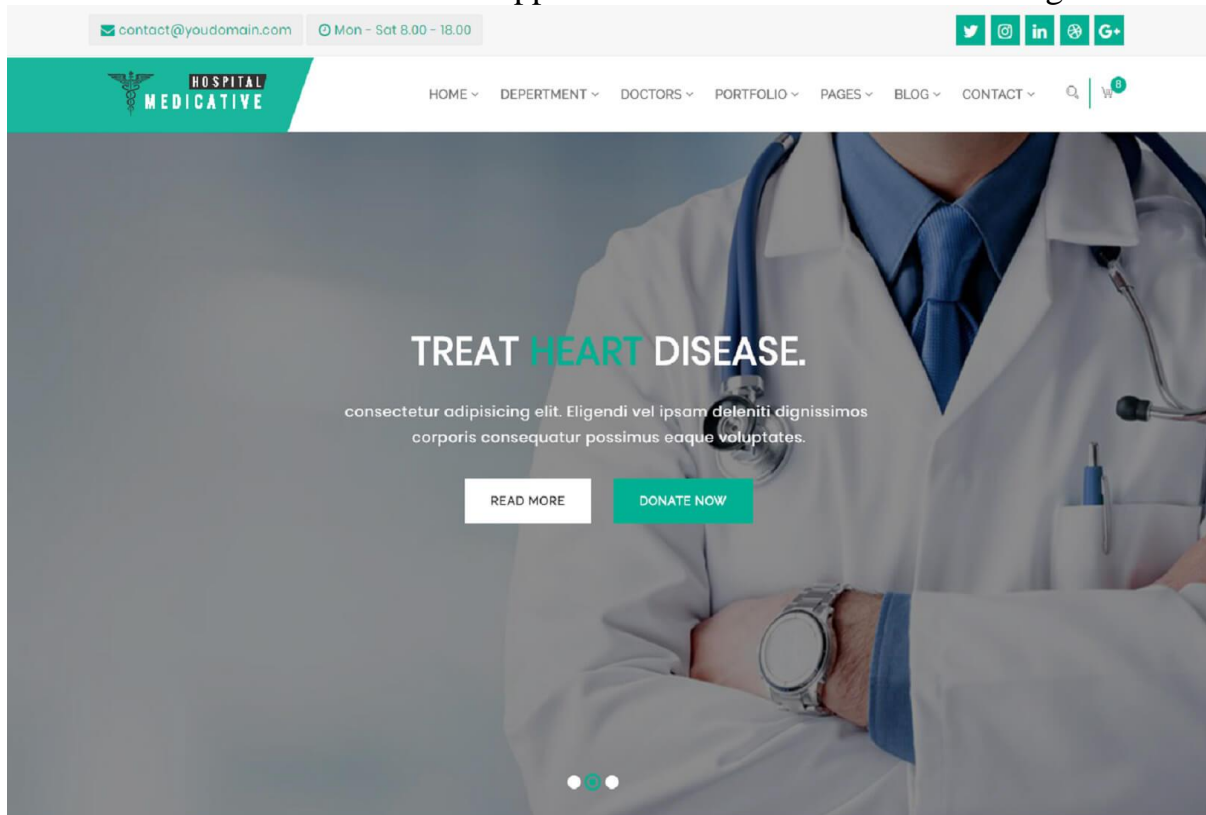


FIG B.1 HOME PAGE

### B.2 PATIENT REGISTRATION FORM:

The page allows patient to appointment to doctor in hospital database as shown in Fig B.2

The screenshot shows a form titled 'OP Registration Form' overlaid on a background image of a doctor's hands using a stethoscope. The form is divided into two main sections: 'Personal Details' and 'Emergency Contact Details'. The 'Personal Details' section includes fields for First Name, Last Name, Email, Phone Number, DOB, Gender, Height (Inches), Weight (Pounds), Marital Status, and Address. The 'Emergency Contact Details' section includes fields for First Name, Last Name, Relationship, and Phone Number. Below these sections, there is a section for 'Taking Any Medications Currently' with radio buttons for 'Yes' and 'No'. If 'Yes' is selected, there is a text area for 'If you are taking please list them here...'. The form is styled with a dark background and white text.

### B.3 ADMIN LOGIN PAGE:

This page allows admin to login and make changes to data base as shown in Fig B.3

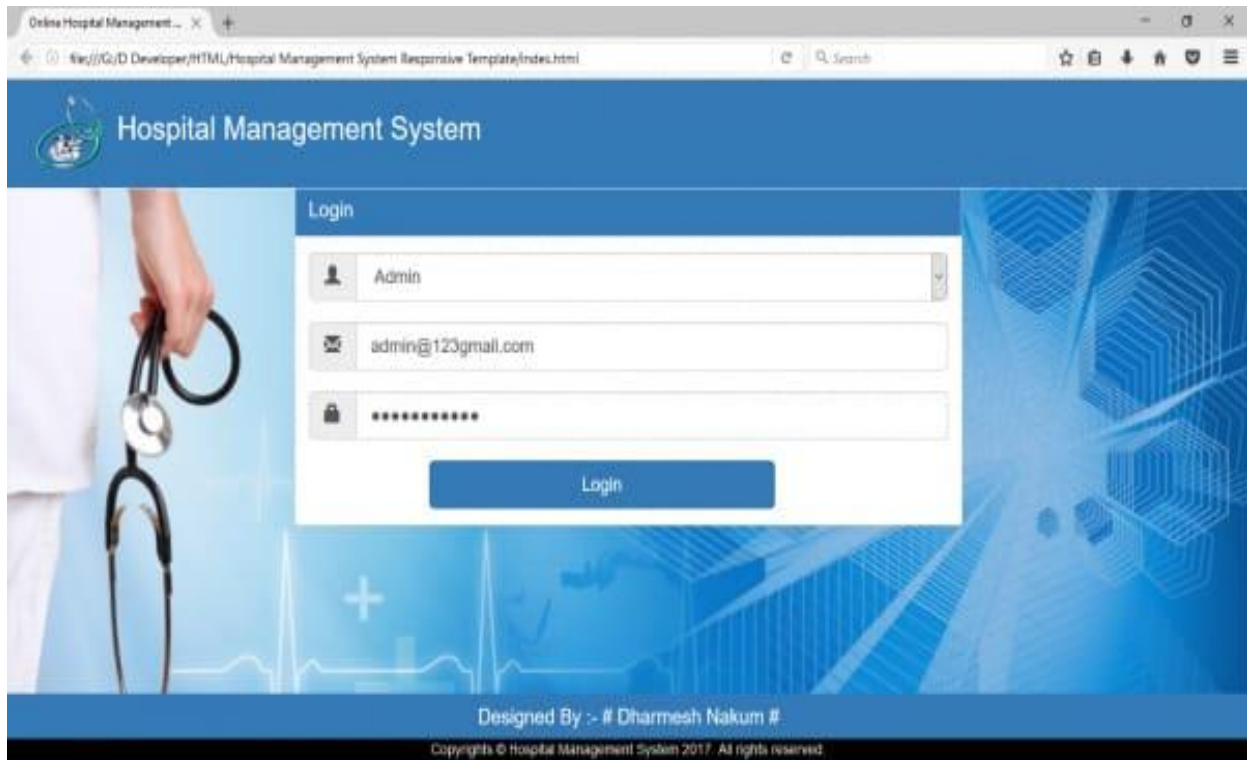


FIG B.3 ADMIN LOGIN PAGE

### B.4 PATIENT ADMISSION:



## B.5 PATIENT RECORD SEARCH:

iptrecord

Search: Aru Benjamin Chandru

midno	name	age	addr	dob	gender	state	district	concession
8866	Arul Benjamin Ch...	21	18A, Samathana...	5/23/1989	Male	Tamilnadu	Madurai	200

Windows taskbar: COLOR PRINTS, WindowsAppli..., COLOR PRINT..., WindowsAppli..., inpatient, iptrecord, 6:06 PM

## B.6 PATIENT BILL:

INPATIENT BILLING

### HOSPITAL MANAGEMENT

GENERAL BILL

BILL NO: 7

MID NO: 1234 PATIENT NAME: S. Thiaga Ram

DATE OF ADMIT: 23-Mar-2018 DATE OF DISCHARGE: 25-Apr-2018

ROOM RENT: 450

LAB TEST FEES: 50

CONSULTATION FEES: 150

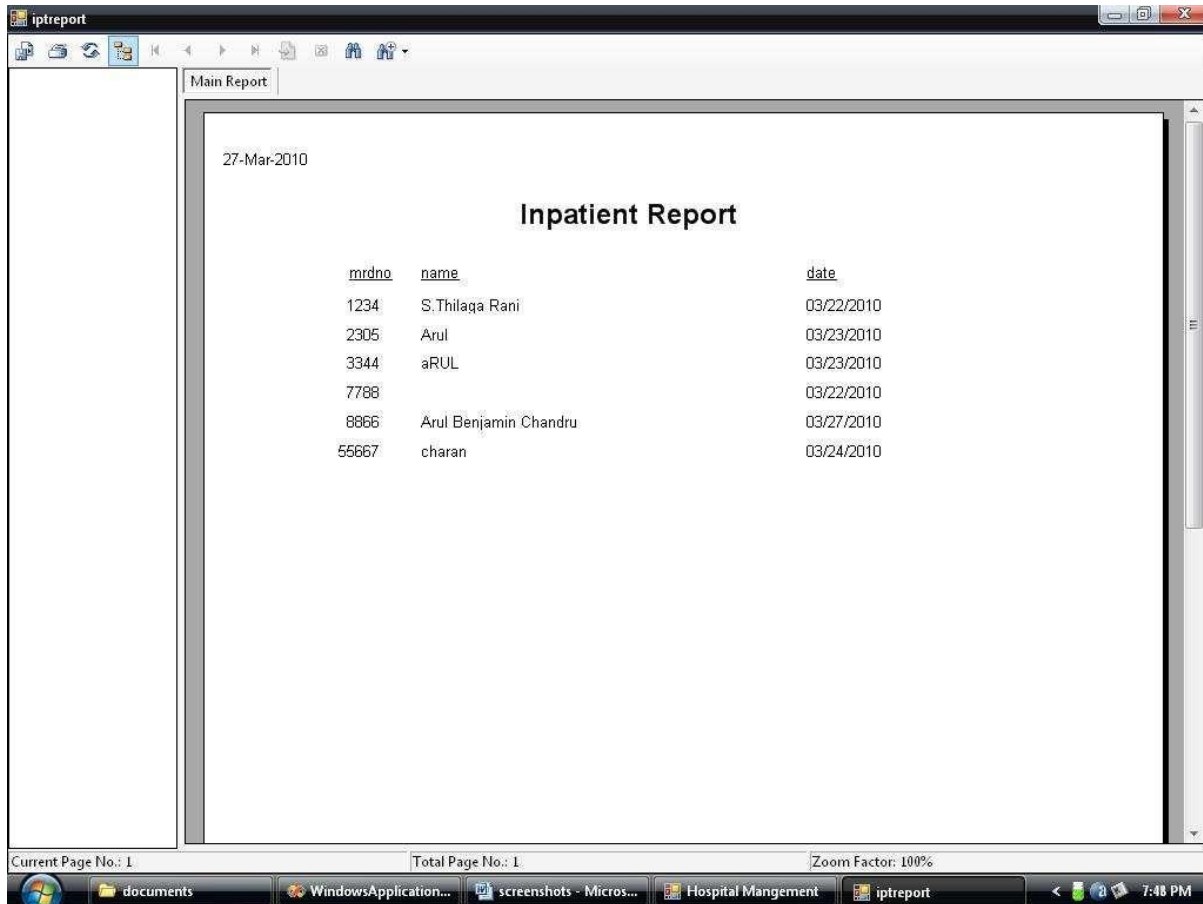
NET PAYMENT: 650

Hospital Management  
Bill Saved to Database  
OK

SAVE BILL View Bills EXIT

Windows taskbar: 100mbits - Moz..., WindowsAppli..., INPATIENT BILLING, 7:18 PM

## B.7 PATIENT REPORTS:



27-Mar-2010

### Inpatient Report

<u>mrdno</u>	<u>name</u>	<u>date</u>
1234	S.Thilaga Rani	03/22/2010
2305	Arul	03/23/2010
3344	aRUL	03/23/2010
7788		03/22/2010
8866	Arul Benjamin Chandru	03/27/2010
55667	charan	03/24/2010

Current Page No.: 1      Total Page No.: 1      Zoom Factor: 100%

## B.8 PATIENT BILLS:



3/27/2010

### INPATIENT BILLS

<u>bno</u>	<u>date</u>	<u>name</u>	<u>amount</u>
1	26-Mar-2010	S.Thilaga Rani	55.00
2	26-Mar-2010	Arul	66.00
3	26-Mar-2010	S.Thilaga Rani	666.00
0	26-Mar-2010	S.Thilaga Rani	66.00
6	27-Mar-2010	S.Thilaga Rani	1,850.00
4	26-Mar-2010	S.Thilaga Rani	1,321.00
5	26-Mar-2010	Arul	1,021.00

55.00  
121.00  
787.00  
853.00  
2,703.00  
4,024.00  
5,045.00