CLINIC DATABASE MANAGEMENT SYSTEM

Project report submitted

As the requirement for the course of

'Database Management and Information System'

By

Vinayak Mohite	Ankit Yadav
170004040	170001007

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

(SEMESTER III – ACADEMIC YEAR 2018-2019)

CONTENTS

CERTIFICATE

It is certified that the work contained in the project report titled "Clinic Database Management System" by "Vinayak Mohite (170004040)" and "Ankit Yadav (170001007)" has been carried out under my/our supervision.

Signature of Supervisor(s)

Computer Science and Engineering

I.I.T. INDORE

Semester III Academic Year (2018-19)

DECLARATION

We declare that this written submission represent our ideas in our own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name: Vinayak Mohite

Roll No: 170004040

Date: 19/11/2018

Name: Ankit Yadav

Roll No: 170001007

Date: 19/11/2018

ACKNOWLEDGEMENT

We wish to acknowledge to excellent support that we have received to complete this project. We would like to thank our instructor, Professor Aruna Tiwari for your labour of love as you toil to bequeath us knowledge and introduce us into the field of Database Management and Information System.

Our heartfelt gratitude goes to google.com which gave answers to every possible question.

INTRODUCTION

PROBLEM STATEMENT

The medical record system is a database management system that uses database technology to construct, maintain and manipulate various kinds of data about a person's medical history and care across time. The DBMS can track and update all the information of registered patients in the medical Centre during a particular time span.

Medical records(Visit Records) are created when a patient receive treatment from a health professional. Records may include the patients:

- ✓ personal information
- ✓ medical history
- ✓ tests prescribed

The medical record serves a variety of purposes and is essential to the proper functioning of the medical practice—especially in today's complicated health care environment. The medical record is a key instrument used in planning, evaluating, and coordinating patient care in both the inpatient and the outpatient settings. The content of the medical record is essential for patient care, accreditation (if applicable to the practitioner), and reimbursement purposes.

The medical record administrator may be a clinician, billing manager, coder, or anyone assigned the responsibility in the medical office. The medical record administrator is responsible for filing patient information in the medical record. He or she is also responsible for knowing legal requirements pertaining to privacy and confidentiality of the patient.

First, patient visits the OPD (outpatient department) where his/her data (Reason for Visit) is recorded and then someone from Medical Office appoints them a staff and some tests are prescribed.

Then, prescribed tests are to be taken. (X-ray, CT scan, USG scan)

OPD time is specified.

Patients are to be seen on first come first serve basis. However, out of turn consultation may be provided in case of emergency or to senior citizens.

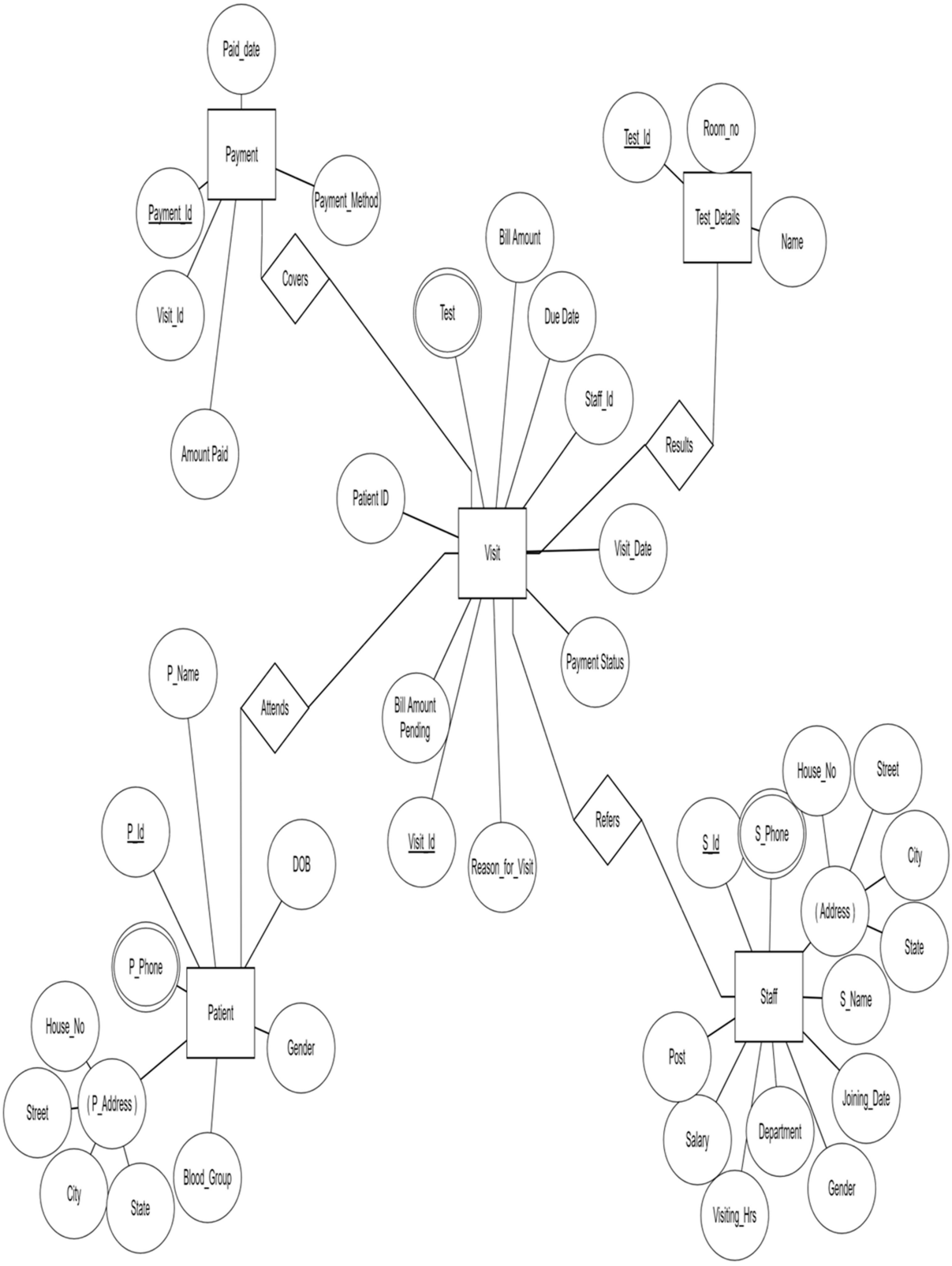
BUSINESS RULE

The business rule for a medical record database:

- ✓ A patient medical record is registered on the first visit to the doctor.
- ✓ A patient can make many appointments with one or more doctors.
- ✓ A doctor can accept appointments with many patients.
- ✓ However, each appointment is made with only one doctor, and each appointment references a single patient
- ✓ If an appointment yields a visit with a doctor, the visit yields a diagnosis when appropriate treatment.
- ✓ Each visit creates a bill. Each patient visit is billed by one doctor, and each doctor can bill many patients.
- ✓ Each bill must be paid in the specified time duration. However, a bill may be paid in many installments.

CONCEPTUAL SCHEMA

ER DIAGRAM



Entity	Attributes
Patient	P_Id, P_Name, DOB, Gender, Blood_Group, P_Phone, P_Address (House_No, Street, City, State)
Staff	S_Id, S_Name, S_Phone, Joining_date, Gender, Department, Visiting_Hrs, Salary, Post, Address (House_No, Street, City, State)
Visit	Visit_Id, Staff_Id, Visit_Date, Reason_For_Visit, Bill_amount_pending, Payment _status, Bill _amount, Due_date, Test_id_1, Test_id_2, Test_id_3, Test_Id_4, Patient_Id
Test_Details	Test_Id, Room_No, Name, Test_cost
Payment	Payment_Id, Visit_Id, Paid_Date, Payment_Method, Amount Paid

RELATIONAL MODEL

FUNCTIONAL DEPENDENCIES

Relation	Function Dependencies
Patient	P_Id →P_Name, DOB, Gender, Blood_Group, P_Phone, P_Address (House_No, Street, City, State)
Staff	S_Id → S_Name, S_Phone, Joining_date, Gender, Department, Visiting_Hrs, Salary, Post, Address (House_No, Street, City, State)
Visit	Visit_Id → Staff_Id, Visit_Date, Reason_For_Visit, Bill_amount_pending, Payment _status, Bill _amount, Due_date, Test_id_1, Test_id_2, Test_id_3, Test_Id_4, Patient_Id
Test_Details	Test_Id → Room_No, Name
Payment	Payment_Id → Visit_Id, Paid_Date, Payment_Method, Amount Paid

RELATIONS

Based on the functional dependencies above, we normalized all relations and converted the ER model into a relational model:

- ✓ **Patient (**P_Id, P_Name, DOB, Gender, Blood_Group, P_Phone, P_Address (House_No, Street, City, State))
- ✓ **Staff (**S_Id, S_Name, S_Phone, Joining_date, Gender, Department, Visiting_Hrs, Salary, Post, Address (House_No, Street, City, State))
- ✓ **Visit** (Visit_Id, Staff_Id, Visit_Date, Reason_For_Visit, Bill_amount_pending, Payment _status, Bill _amount, Due_date, Test_id_1, Test_id_2, Test_id_3, Test_Id_4, Patient_Id)
- ✓ **Test_Details** (Test_Id , Room_No, Name)
- ✓ **Payment** (Payment_Id, Visit_Id, Paid_Date, Payment_Method, Amount Paid)

KEYS DEFINITION

CANDIDATE KEYS DEFINITION

Relations	Candidate Keys (Primary Keys)
Patient	P_Id
Staff	S_Id
Visit	Visit_Id
Test_details	Test_Id
Payment	Payment_Id

DOMAIN KEY NORMAL FROM DEFINITION

Relations	Constraints
Patient	**Key:** (primary) P_Id **Domain:** P_id: int(5) primary key auto_increment, P_name: varchar(25) not null, DOB: date, Gender: enum('M','F'), P_phone_1: int check(P_phone_1 % 10000000 <> 0), P_phone_2: int check(P_phone_2 % 10000000 <> 0), Blood_group: enum('B+', 'B-', 'O+', 'O-', 'A+', 'A-', 'AB+', 'AB-'), Add_house_no: varchar(5), Add_street varchar(25), Add_city varchar(25), Add_state varchar(25)
Staff	**Key:** (primary) S_Id **Domain:** S_id: int(5) primary key auto_increment, S_name: varchar(25) not null, Joining_date: date, Gender: enum('M', 'F'), S_phone_1: int check(P_phone_1 % 10000000 <> 0), S_phone_2: int check(P_phone_2 % 10000000 <> 0), Department: varchar(25), Salary: int, Post: varchar(25), Visiting_hrs: varchar(15), Add_house_no: varchar(5), Add_street: varchar(25), Add_city: varchar(25), Add_state: varchar(25)

Visit	**Key:** (primary) Visit_Id **Domain:** Visit_id: int(5) primary key auto_increment, S_id: int(5), Visit_date: date, Reason_for_visit: varchar(50), P_id: int(5), test_1: int(5), test_2: int(5), test_3: int(5), test_4: int(5), bill_amount: int, Due_date: date, Payment_status: enum('completed', 'pending') default 'pending', Amount_pending: int
Test_details	**Key:** (primary) Test_Id **Domain:** Test_id: int(5) primary key auto_increment, Room_no: varchar(5), Name: varchar(25), test_cost: int
Payment	**Key:** (primary) Payment_Id **Domain:** Payment_id: int(5) primary key auto_increment, Visit_id: int(5), Paid_date: date, Payment_method: varchar(25), amt_paid: int

POPULATING THE TABLES

1. PATIENT

Payment_id	Visit_id	Paid_date	Payment_method	amt_paid
22	11	2018-11-16	Cash	700
23	11	2018-11-16	Cash	700
24	12	2018-11-16	Cash	600
25	14	2018-11-17	Debit Card	600
NULL	NULL	NULL	NULL	NULL

2. STAFF

S_id	S_name	Joining_date	Gender	S_phone_1	S_phone_2	Department	Salary	Post	Visiting_hrs	Add_house_no	Add_street	Add_city	Add_state
3	Harish	2016-10-04	M	948564754	783522747	Mental Disorder	40000	Psychiatrist	07:00 - 13:30	H-10	Govt Street	Bhopal	Madhya Pradesh
4	Ramesh	2018-10-01	M	973762273	793762576	Surgery	30000	Physician	10:00 - 16:00	J-89	H-Block	Devas	Madhya Pradesh
5	Prerna	2015-04-14	F	966472473	894323627	Children's Department	35000	Pediatrician	09:00 - 16:00	B-90	Nagar Manmar Road	Amarnagar	Maharashtra
6	Vijay	2017-09-10	M	954365783	832568743	OPD	10000	Ward Boy	08:00 - 18:00	C-1	Khandwa Colony	Indore	Madhya Pradesh
7	Sarita	2016-09-26	F	923564554	893574548	Homeopathy	40000	Doctor	10:00 - 16:00	K-9	Taj Road	Agra	Uttar Pradesh
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3. VISIT

Visit_id	S_id	Visit_date	Reason_for_visit	P_id	test_1	test_2	test_3	test_4	bill_amount	Due_date	Payment_status	Amount_pending
13	6	2018-11-16	Loose Motion	10	3	4	6	6	600	2019-01-02	completed	0
15	7	2018-11-16	Back pain	12	2	3	4	5	750	2019-01-16	completed	0
16	5	2018-11-16	Neck Pain	12	2	3	4	5	750	2018-12-21	pending	400
17	3	2018-11-21	Headache	14	4	5	6	6	150	2018-12-19	pending	150
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

4. TEST_DETAILS

Test_id	Room_no	Name	test_cost
2	101	X-Ray	100
3	102	CT-scan	500
4	201	Influenza A	100
5	202	Influenza B	50
6	Null	Null	0
NULL	NULL	NULL	NULL

5. PAYMENT

	Payment_id	Visit_id	Paid_date	Payment_method	amt_paid
▶	26	13	2018-11-21	Cash	500
	27	16	2018-11-21	Debit Card	350
	28	15	2018-11-21	Debit Card	750
	29	13	2018-11-21	Cash	100
*	NULL	NULL	NULL	NULL	NULL

QUERIES

1.

SHOWING THE NUMBER OF VISITS MADE BY PARTICULAR PATIENT

```
1  select P_id, P_name, count(Visit_id) as 'No. of Visits' from Patient
2  join Visit using(P_id)
3  group by P_id;
4
5
6
7
8
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

P_id	P_name	No. of Visits
10	Gaurav	1
12	Asha	2
14	Rama	1

2.

GETTING ALL THE PAYMENTS BY PARTICULAR PATIENTS

```
1  select P_id, P_name, Payment_id, amt_paid from Payment
2  join (Visit join Patient using(P_id)) using(Visit_id)
3  order by P_id;
4
5
6
7
8
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

P_id	P_name	Payment_id	amt_paid
10	Gaurav	26	500
10	Gaurav	29	100
12	Asha	27	350
12	Asha	28	750

3.

GETTING ALL THE DETAILS OF PARTICULAR PATIENT

```
1  select P_id, P_name, Visit_id, bill_amount, (bill_amount - Amount_pending) as 'Amount Paid', Payment_status from Payment
2  join (Visit join Patient using(P_id)) using(Visit_id)
3  group by Visit_id;
4
5
6
7
8
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

P_id	P_name	Visit_id	bill_amount	Amount Paid	Payment_status
10	Gaurav	13	600	600	completed
12	Asha	16	750	350	pending
12	Asha	15	750	750	completed

SOFTWARE SPECIFICATIONS

HTML

HTML is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start-tag, and the second tag is the end tag (they are also called opening tags and closing tags). Though not always necessary, it is best practice to append a slash to tags which are not paired with a closing tag.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags but uses the tags to interpret the content of the page. HTML describes the structure of a website semantically along with cues for presentation, making it a mark-up language rather than a programming language.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behaviour of HTML web pages.

MYSQL

MySQL is developed, distributed, and supported by Oracle Corporation. MySQL is a database system used on the web it runs on a server. MySQL is ideal for both small and large applications. It is very fast, reliable, and easy to use. It supports standard SQL. MySQL can be compiled on a number of platforms.

The data in MySQL is stored in tables. A table is a collection of related data, and it consists of columns and rows. Databases are useful when storing information categorically.

WHY USE MySQL:

- ✓ Leading open source RDBMS
- ✓ Ease of use - No frills
- ✓ Fast
- ✓ Robust
- ✓ Security
- ✓ Multiple OS support
- ✓ Free
- ✓ Technical support
- ✓ Support large databaseup to 50 million rows, file size limit up to 8 Million TB

FLASK IN PYTHON

Flask is a micro framework for python which is based on **Werkzeug** and **Jinja 2.**

In simple words, a framework is anything that helps you complete your work faster. For example, if I were to tell you to create a python server that could host a website; you would be able to do it in just under 120 lines of python code using several libraries for tasks such as, mapping a function to a route(URL), writing socket functions for protocols and finally displaying that one webpage. You will have to repeat all of this for every webpage in the website. Not to forget the overhead of actually interpreting and typing all the lines of code. A framework abstract all the above lower level tasks and helps you focus on the actual application. Clearly, Flask doesn't restrict itself to being a backend service, but instead stages, hosts and handles your complete web application

Why Flask:

There are several python web frameworks out there; there is the infamous Django, bottle, tornado and then there is Flask. What separates Flask from all the others is:

- ✓ It's simple to learn and extensively scalable.
- ✓ Since it isn't scaffolded, you can do whatever you want and however, you want to. No restrictions on the application architecture nor on the Data abstraction layers. This is exactly why not including the batteries makes the application even more utilitarian and functional.
- ✓ A growing community which is providing neat solutions to several problems being faced by developers.

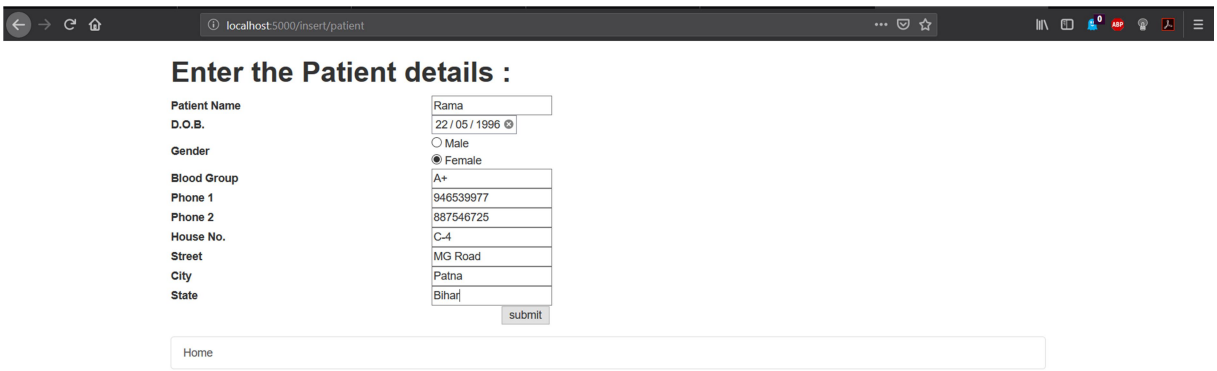
The major advantage of Flask is that you are in control of anything and everything. Knowing this, you can set out to write an application which is limited by the power of python itself. Though it hasn't received its 1.0 update yet, several companies have successfully deployed it to production level and are still using it. Learning Flask, lets you understand the internal mechanics of other frameworks and gives you a good picture of how things work, on the inside.

SCREENSHOTS

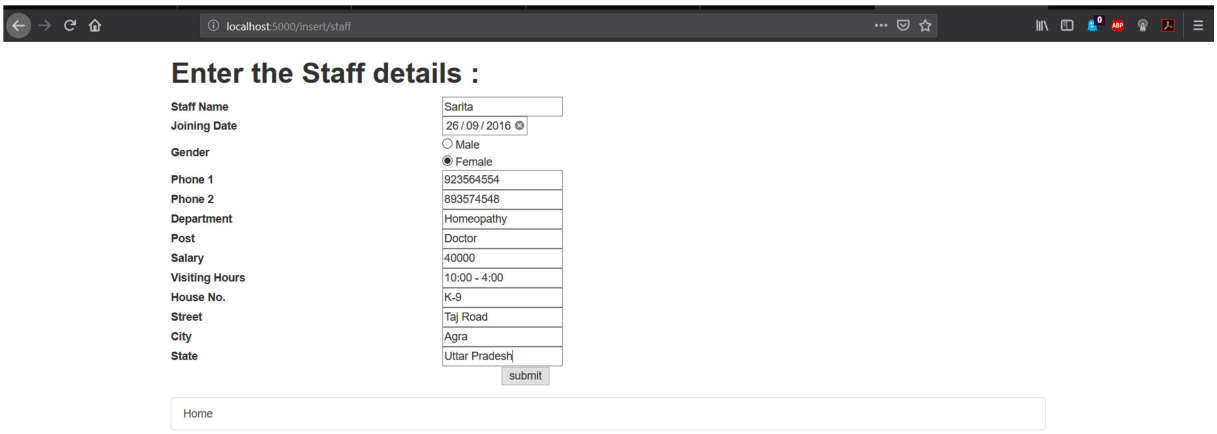
ADMIN PANEL

INSERT PAGE

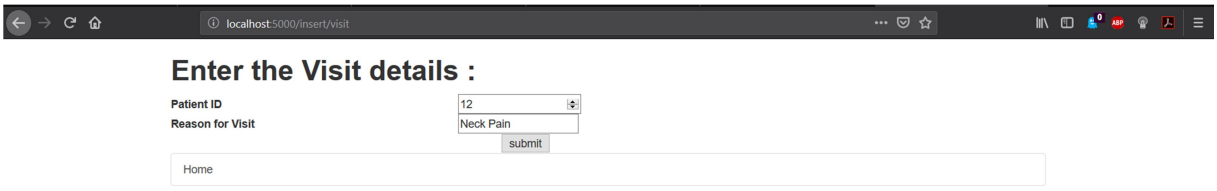
ADDING A NEW PATIENT DETAILS



ADDING A NEW STAFF DETAILS

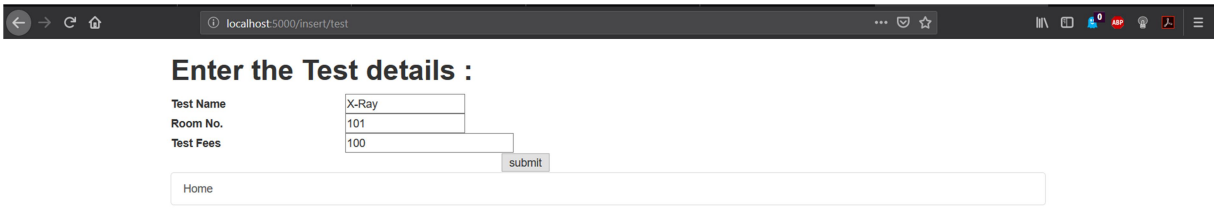


ADDING VISIT DETAILS

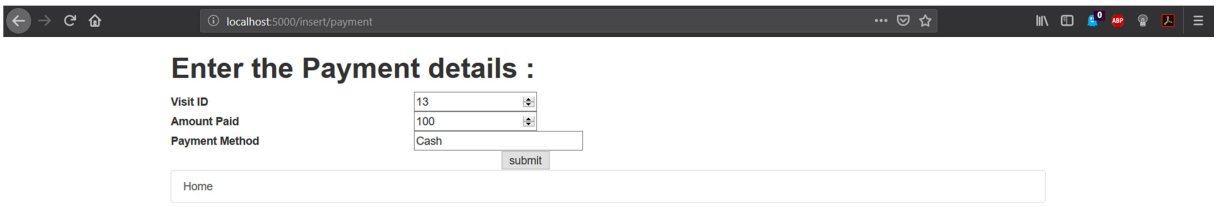


EDITING THE VISIT DETAILS

ADDING A NEW TEST DETAILS

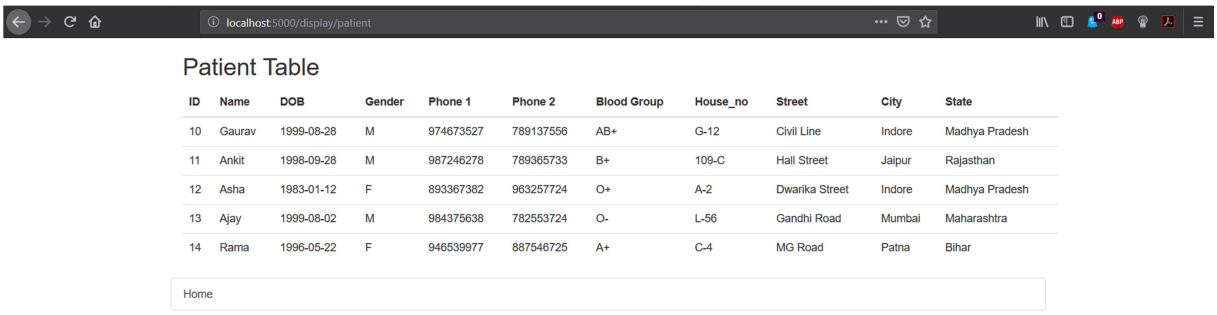


ADDING A NEW PAYMENT DETAILS

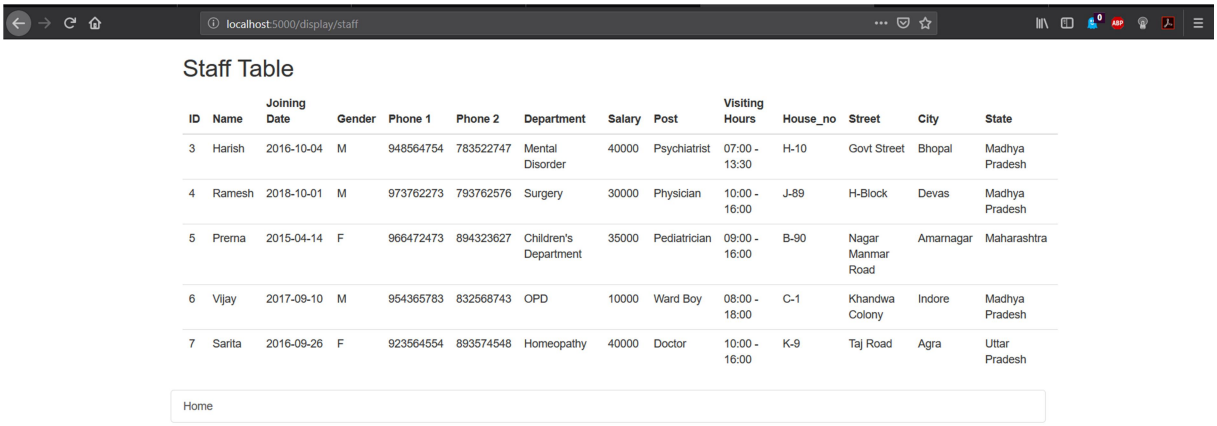


DISPLAY PAGE

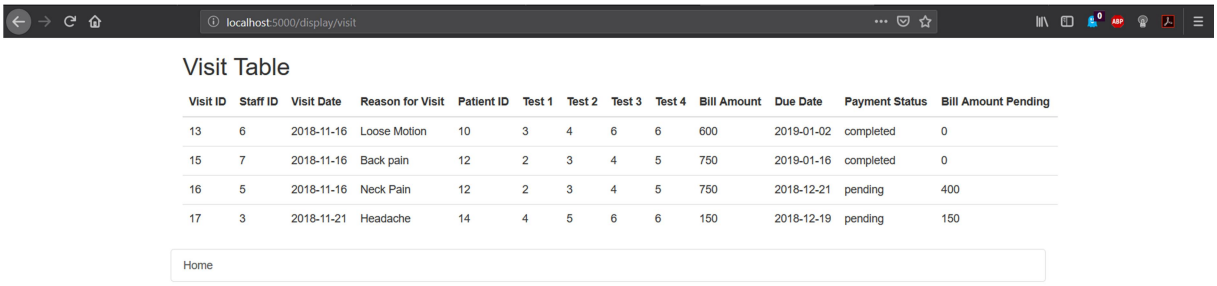
PATIENT DETAILS



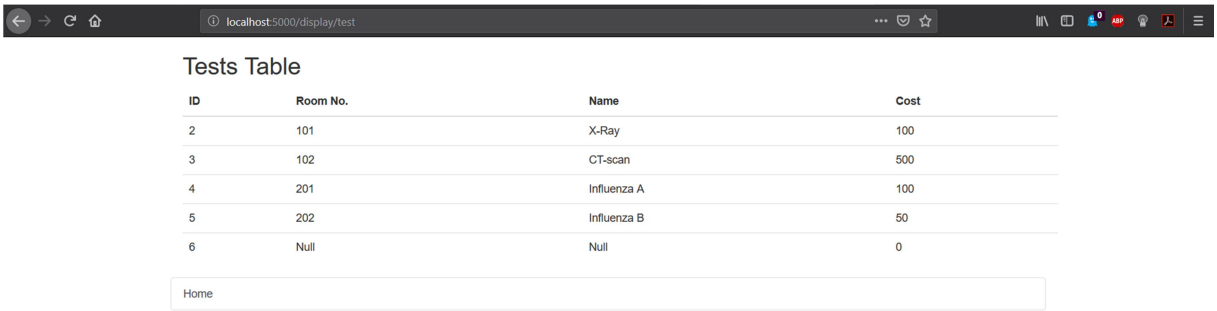
STAFF DETAILS



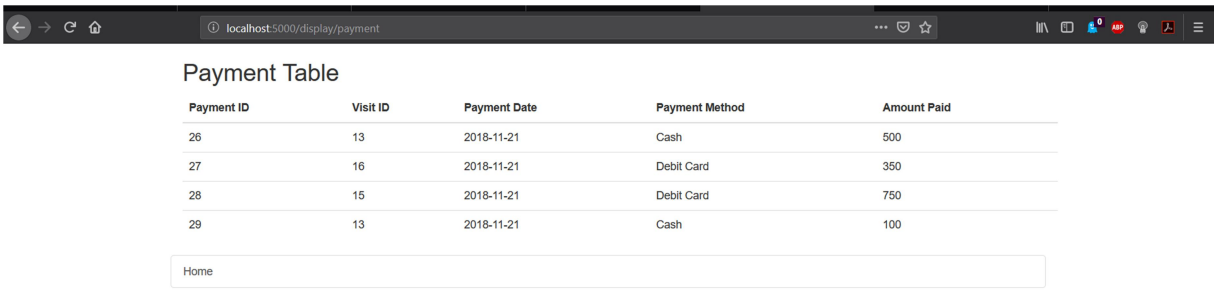
VISIT DETAILS



TEST DETAILS

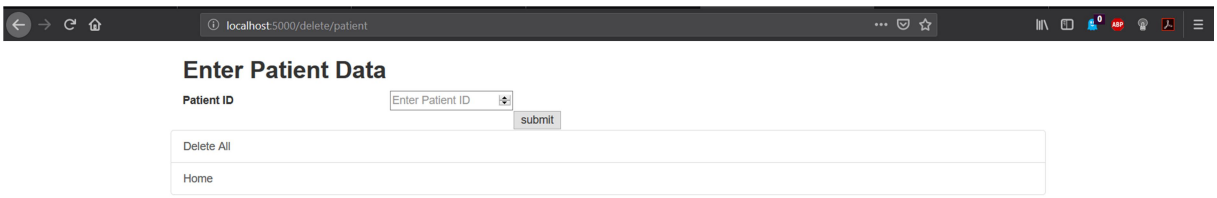


PAYMENT DETAILS

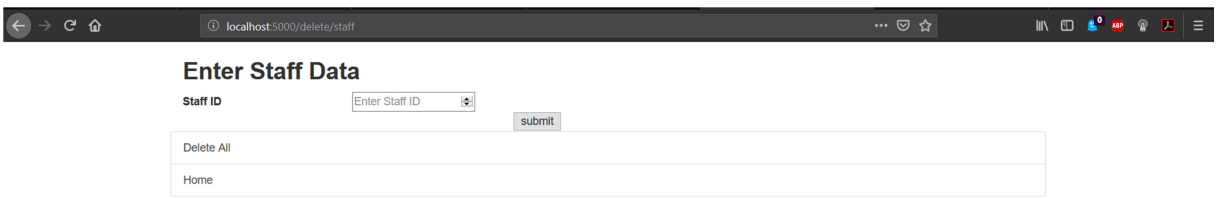


DELETION PAGE

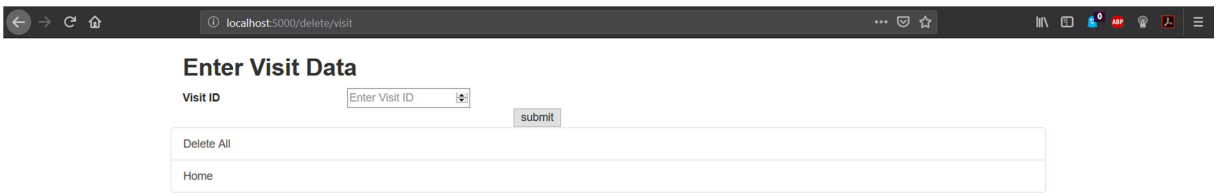
PATIENT DELETION



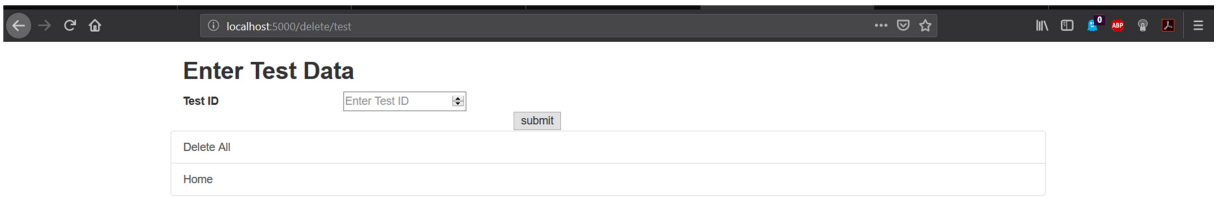
STAFF DELETION



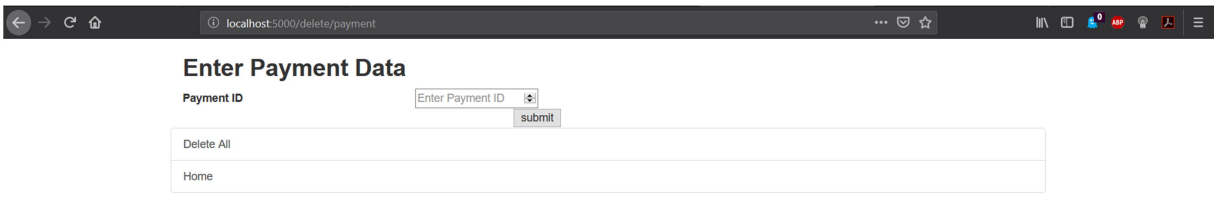
VISIT DELETION



TEST DELETION

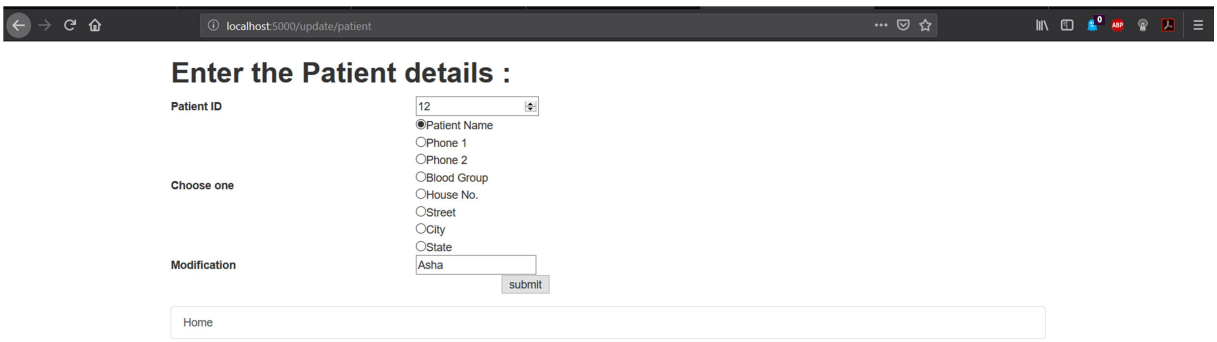


PAYMENT DELETION

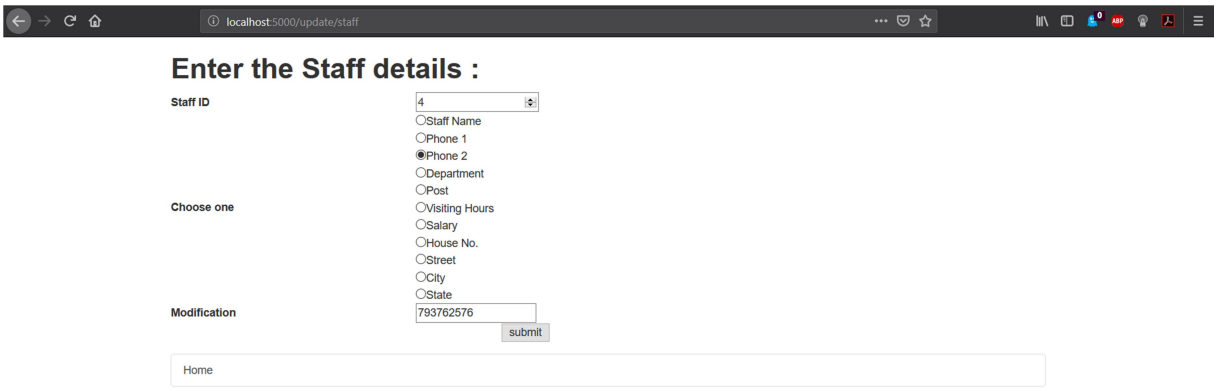


UPDATE PAGE

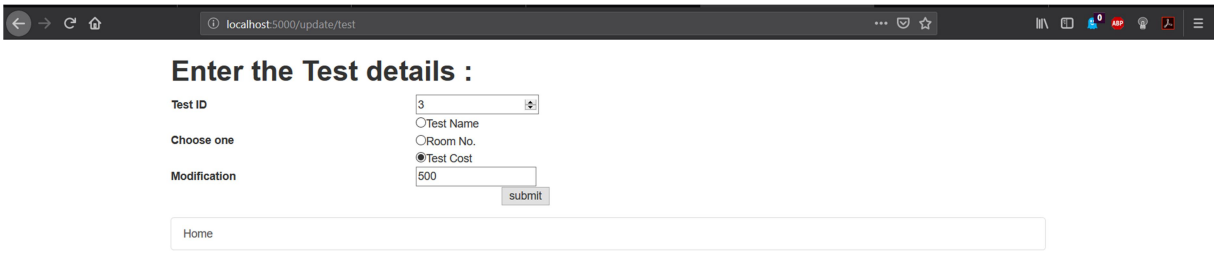
EDITING PATIENT DETAILS



EDITING STAFF DETAILS

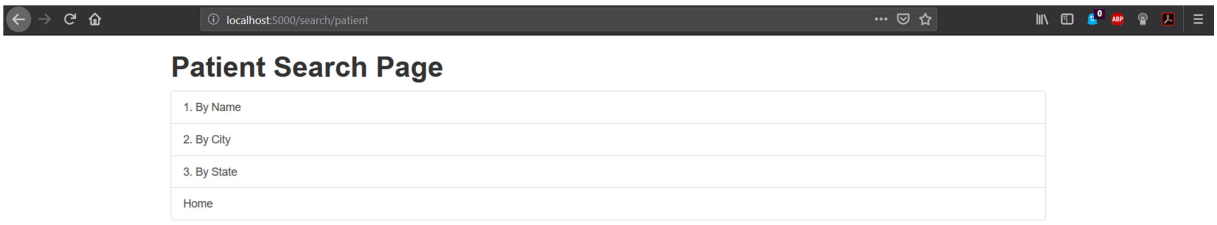


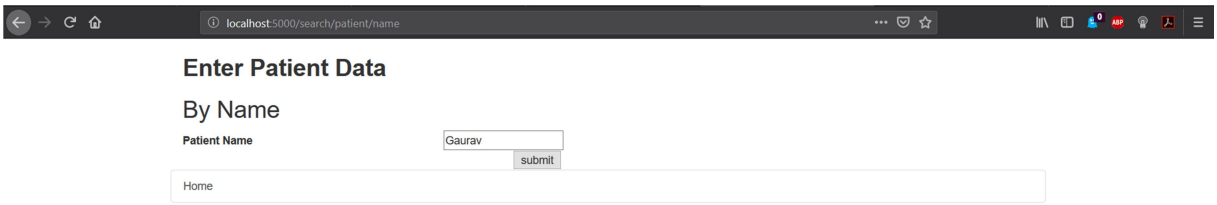
EDITING TEST DETAILS

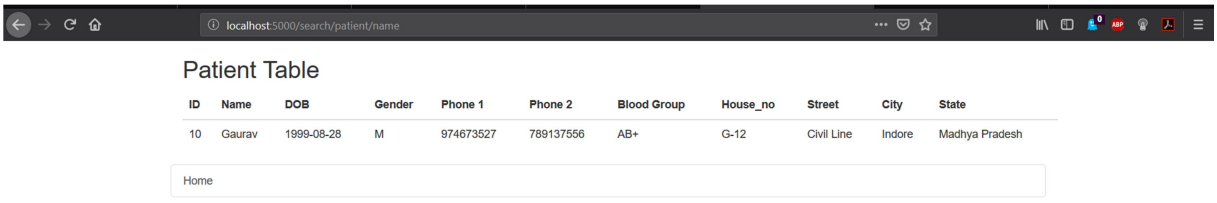


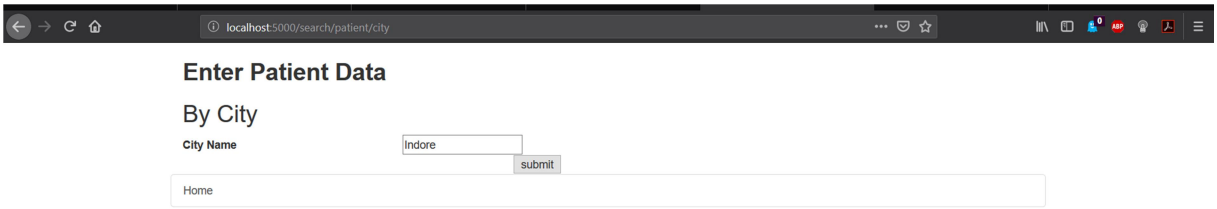
SEARCH PAGE

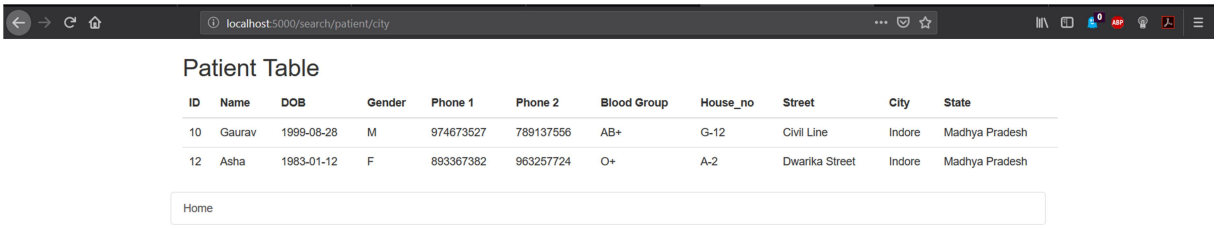
SEARCHING PATIENT DETAILS

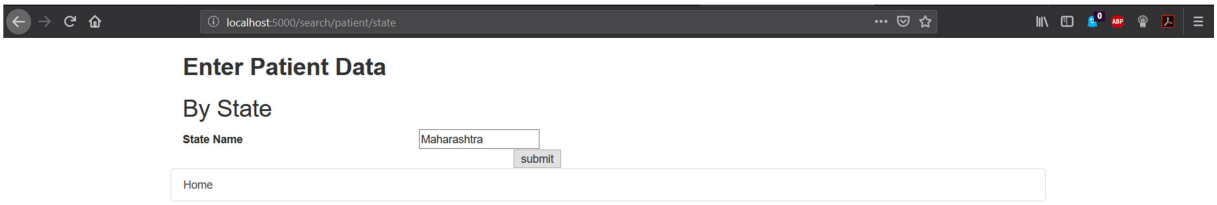


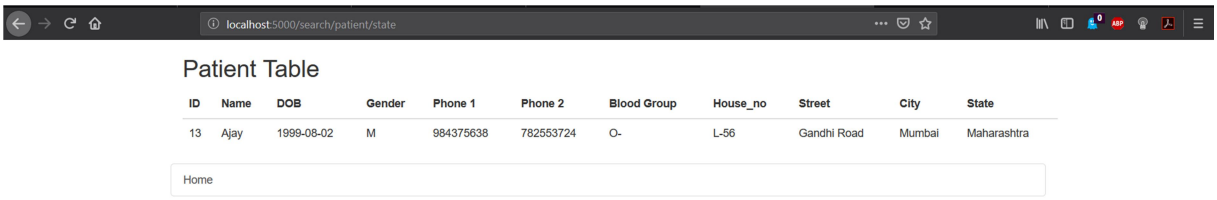




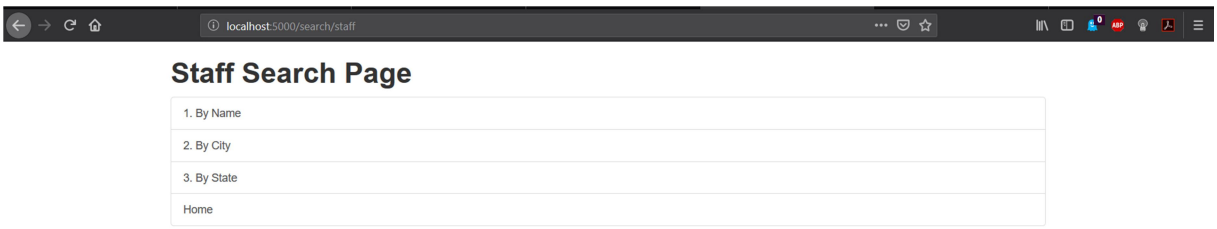




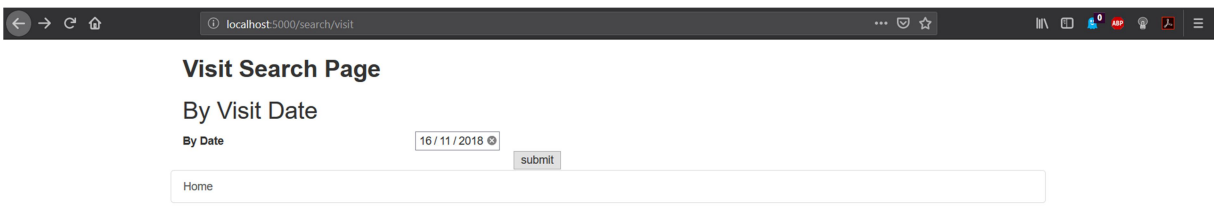


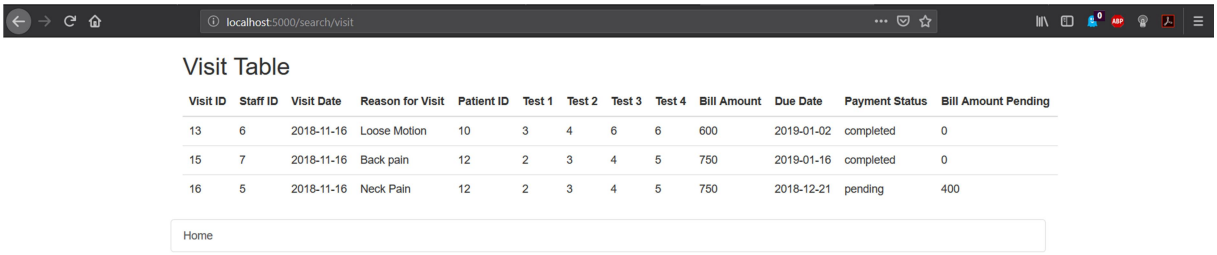


SEARCHING STAFF DETAILS

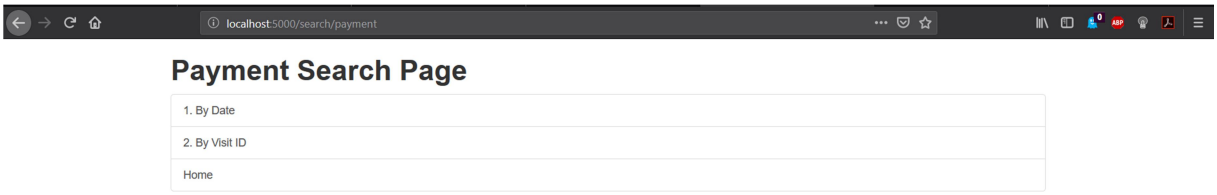


SEARCHING VISIT DETAILS





SEARCHING PAYMENT DETAILS



CONCLUSION

Since we are entering details of the patients electronically in the" Hospital Management System", data will be secured. Using this application we can retrieve the patient's history with a single click. Thus processing information will be faster. It guarantees the accurate maintenance of Patient details. It easily reduces the bookkeeping task and thus reduces the human effort and increases accuracy speed.

REFERENCES

- ✓ Lecture notes
- ✓ http://www.wikipedia.com
- ✓ https://books.google.co.in/
- ✓ Database System Concepts by A. Silberschatz, H. Korth and S. Sudarshan
- ✓ http://flask.pocoo.org/
- ✓ https://www.academia.edu