Database Management System Mini Project Report

Hospital Management System

By

Mr.Tejas Janjire

(20CO042)

Ms.Sanika Jawanjal

(20CO044)

MsAtharva Kharat

(20CO056)

Mr.Sanket Kewale

(20CO054)

Under the Guidance of

Prof.V.V.Navale

Prof.A.M.Jagtap

Department of Computer Engineering

All India Shri Shivaji Memorial Society's College of Engineering, Pune-1

AISSMS
COLLEGE OF ENGINEERING
ज्ञानम् सकलजनहिताय
Accredited by NAAC with "A+" Grade

Department of Computer Engineering

C E R T I F I C A T E

This is to certify that:
Ms. Sanika Jawanjal(20CO044)

Ms. Atharva Kharat(20CO056)

Mr. Sanket Kewale(20CO054)

Mr. Tejas Janjire(20CO042)

The third year, Branch **COMPUTER-I** has completed the report under Database Management System (310246) on the topic Project entitled ***"Hospital Management System"*** under my supervision, in the partial fulfilment of Third Year Bachelor of Engineering (Choice Based Credit System) (2019 Course) of Savitribai Phule University of Pune.

Date

4/11/22

Place:

Pune

Prof.V.V.Navale	**Dr. S. V. Athawale**	**Dr. D. S. Bormane**
Prof.A.M.Jagtap		
GUIDE	HOD	PRINCIPAL

ACKNOWLEDGEMENT

Apart from the efforts of the team, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. The completion of any inter-disciplinary project depends upon cooperation, coordination and combined efforts of several sources of knowledge. We are eternally grateful to our teacher **Prof. V.V.navale ma'am and Prof. A.M.Jagtap sir** for her even willingness to give us valuable advice and direction under which we executed this project. Her constant guidance and willingness to share her vast knowledge made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks.

We would like to extend thanks to our respected Head of the Department, **Dr. S.V. Athawale sir** for allowing us to use the facilities available. We would like to thank other faculty members also.

Tejas Janjire

Sanika Jawanjal

Atharvaa Kharat

Sanket Kewale

ABSTRACT

Our project **Hospital Management system** includes registration of patients, storing their details into the system, and also booking their appointments with doctors.
Our software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. Users can search for the availability of a doctor and the details of a patient using the id. The Hospital Management
System can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data is well protected for personal use and makes the data processing very fast.

It has mainly two modules. One is at Administration Level and other one is of user
I.e. of patients and doctors. The Application maintains authentication in order to access the application. Administrator tasks include managing doctors information, patient's information. To achieve this aim a database was designed one for the patient and other for the doctors which the admin can access. The complaints which are given by the user will be referred to by authorities.
The Patient modules include checking appointments, prescription. User can also pay
doctor's Fee online

TABLE OF CONTENTS

CONTENT	PAGE NO.
1.INTRODUCTION a.Problem Statement b.Aim c.Objective d.Overview e. Benefits	
2.REQUIREMENTS SPECIFICATIONS a.Product Perspective b.H/W Requirements c.S/W Requirements	
3.DETAILED DESIGN a. Entity Relationship Diagram b.Relational Schema c.Description of tables d.Overall Description	
4.SNAPSHOTS OF WEBSITE	
5.SYSTEM TESTING	
6.CONCLUSION	
7.REFERENCES	

INTRODUCTION

Problem Statement:

In this busy world we don't have the time to wait in infamously long hospital queues. The problem is, queuing at hospital is often managed manually by administrative staff, then take a token there and then wait for our turn then ask for the doctor and the most frustrating thing - we went there by travelling a long distance and then we come to know the doctor is on leave or the doctor can't take appointments.

This will help us overcome all these problems because now patients can book their appointments at home, they can check whether the doctor they want to meet is available or not. Doctors can also confirm or decline appointments, this helps both patient and the doctor because if the doctor declines' appointment then patient will know this in advance and patient will visit hospital only when the doctor confirms' the appointment this will save time and money of the patient.Patients can also pay the doctor's consultant fee online to save their time.

This is essential for all healthcare establishments, be it hospitals, nursing homes, health clinics, rehabilitation centres, dispensaries, or clinics. The main goal is to computerise all the details regarding the patient and the hospital. The installation of this healthcare software results in improvement in administrative functions and hence better patient care, which is the prime focus of any healthcare unit.

Aim:

This software will help the company to be more efficient in registration of their patients and manage appointments, records of patients. It enables doctors and admin to view and modify appointments schedules if required. The purpose of this project is to computerise all details regarding patient details and hospital details.

Objective:

The system will be used as the application that serves hospitals, clinics, dispensaries or other health institutions. The intention of the system is to increase the number of patients that can be treated and managed properly.

Overview:

Our application contains two modules – the admin module and the user module. Our application will not only help the admin to preview the monthly and/or yearly data but it will also allow them to edit, add or update records.

The software will also help the admin to monitor the transactions made by the patients and generate confirmations for the same. The admin will be able to manage and update information about doctors.

The user module can be accessed by both the doctors and the patients. The doctor can confirm and/or cancel appointments. The doctors can even add prescriptions for their patients using our application. The patients will be able to apply for the appointment and make transactions for the same, and can even cancel appointments with the doctors. They can track details about the previous transactions made by them.

Benefits of implementing a hospital management system:

- **Appointment booking**
  - Helps patients cut the long queue and saves their time
  - Is equipped with features like automated email and text message reminders

- **Role-Based Access Control**
  - Allows employees to access only the necessary information to effectively perform their job duties
  - Increases data security and integrity

- **Overall cost reduction**
  - Cuts down paper costs as all the data are computerised
  - No separate costs for setting up physical servers

- **Data security**
  - Helps to keep patients records private
  - Restricts access through role-based access control

- **Revenue management**
  - Makes daily auditing simple
  - Helps with statistics and other financial aspects.

- **Data accuracy**
  - Removes human errors
  - Alerts when there’s a shortage of stock

REQUIREMENTS SPECIFICATION

Product Perspective:

This Hospital Patient Info Management System is a self-contained system that manages activities of the hospital.

Due to improperly managed details, the medical centre faces quite a lot of difficulties in accessing past data as well as managing present data. The fully functional automated hospital management system which will be developed through this project will eliminate the disadvantages caused by the manual system by improving the reliability, efficiency and performance. The usage of a database to store patient, employee, stock details etc. will accommodate easy access, retrieval, and search and manipulation of data.

The access limitations provided through access privilege levels will enhance the security of the system. The system will facilitate concurrent access and convenient management of activities of the medical centre.

H/W Requirements:

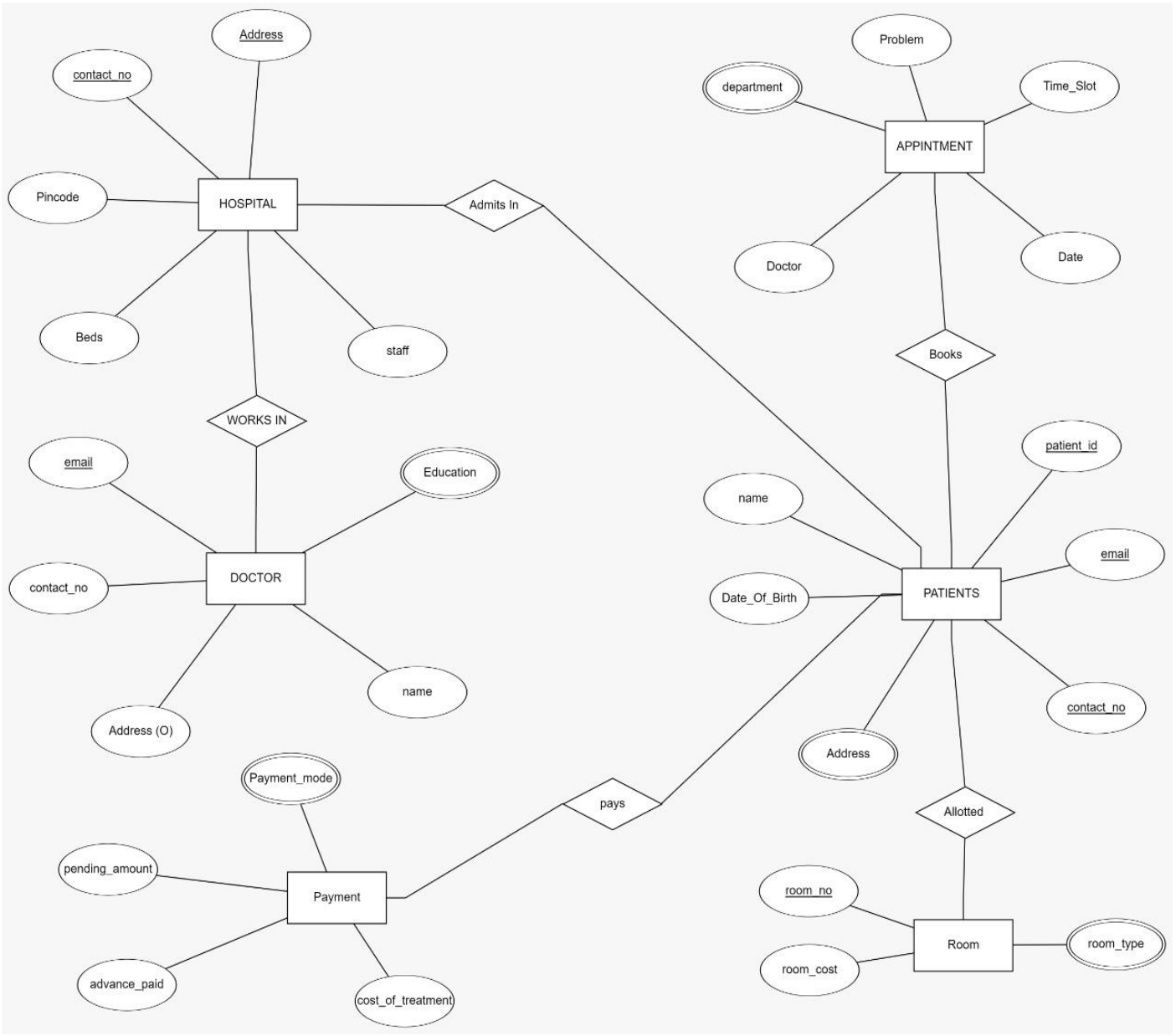
- Processor - Pentium iv and above
- Ram - 256 MB and above
- Hard disk - 1GB or above

S/W Requirements:

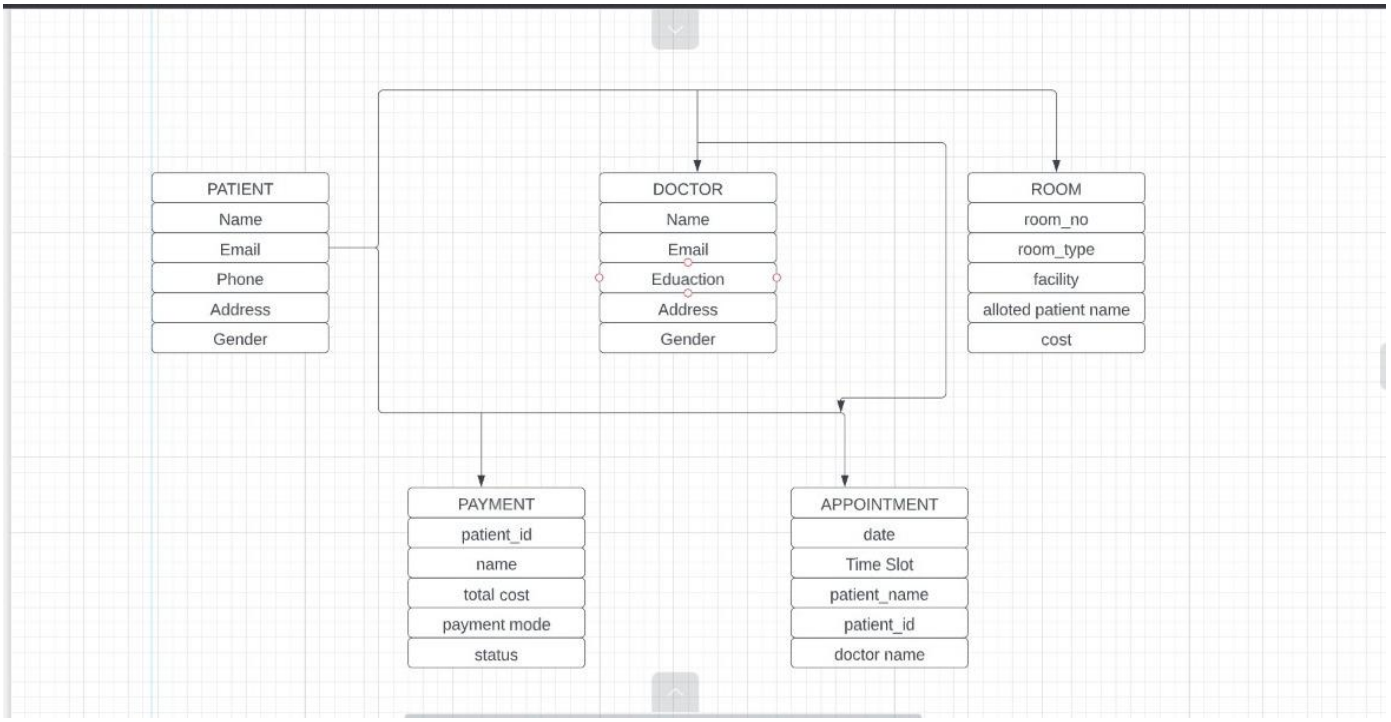
- MySQL
- HTML
- JavaScript
- CSS
- DJANGO

DETAILED DESIGN

Entity Relationship Diagram:

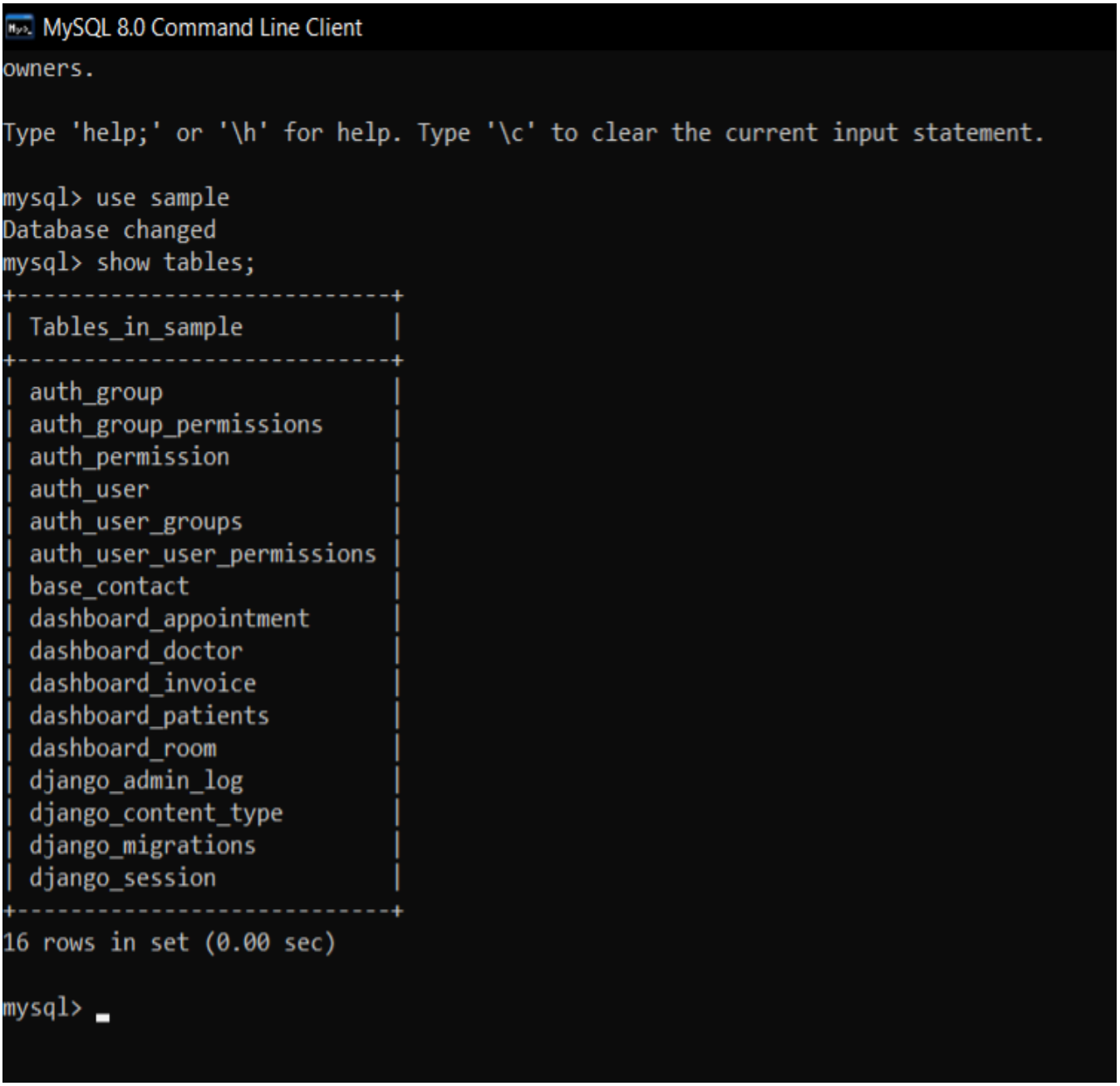


Relational Schema:



Description of Tables:

```
mysql> desc dashboard_appointment;
+------------------+--------------+------+-----+---------+----------------+
| Field            | Type         | Null | Key | Default | Extra          |
+------------------+--------------+------+-----+---------+----------------+
| id               | bigint       | NO   | PRI | NULL    | auto_increment |
| Patient_id       | varchar(100) | NO   |     | NULL    |                |
| department       | varchar(100) | NO   |     | NULL    |                |
| doctor_name      | varchar(100) | NO   |     | NULL    |                |
| appointment_date | date         | NO   |     | NULL    |                |
| time_slot        | varchar(70)  | NO   |     | NULL    |                |
| problem          | longtext     | NO   |     | NULL    |                |
| patient_name     | varchar(100) | NO   |     | NULL    |                |
+------------------+--------------+------+-----+---------+----------------+
8 rows in set (0.04 sec)
```


MySQL 8.0 Command Line Client
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> use sample
Database changed
mysql> show tables;
+----------------------------+
| Tables_in_sample |
+----------------------------+
| auth_group |
| auth_group_permissions |
| auth_permission |
| auth_user |
| auth_user_groups |
| auth_user_user_permissions |
| base_contact |
| dashboard_appointment |
| dashboard_doctor |
| dashboard_invoice |
| dashboard_patients |
| dashboard_room |
| django_admin_log |
| django_content_type |
| django_migrations |
| django_session |
+----------------------------+
16 rows in set (0.00 sec)
mysql>

```
mysql> desc  dashboard_invoice;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| id              | bigint      | NO   | PRI | NULL    | auto_increment |
| Patient_id      | varchar(100)| NO   |     | NULL    |                |
| patient_name    | varchar(100)| NO   |     | NULL    |                |
| department      | varchar(100)| NO   |     | NULL    |                |
| doctor_name     | varchar(100)| NO   |     | NULL    |                |
| addmission_date | date        | NO   |     | NULL    |                |
| discharge_date  | date        | NO   |     | NULL    |                |
| cost            | varchar(100)| NO   |     | NULL    |                |
| advance         | varchar(100)| NO   |     | NULL    |                |
| pending         | varchar(100)| NO   |     | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
10 rows in set (0.00 sec)
```

```
mysql> desc  dashboard_doctor;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| id            | bigint      | NO   | PRI | NULL    | auto_increment |
| dr_Name       | varchar(100)| NO   |     | NULL    |                |
| date_of_birth | date        | NO   |     | NULL    |                |
| age           | int         | NO   |     | NULL    |                |
| phone         | varchar(70) | NO   |     | NULL    |                |
| email         | varchar(100)| NO   |     | NULL    |                |
| gender        | varchar(50) | NO   |     | NULL    |                |
| address       | longtext    | NO   |     | NULL    |                |
| education     | varchar(100)| NO   |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
9 rows in set (0.00 sec)
```

```
mysql> desc dashboard_patients;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| id            | bigint       | NO   | PRI | NULL    | auto_increment |
| patient_Name  | varchar(100) | NO   |     | NULL    |                |
| dr_Name       | varchar(100) | NO   |     | NULL    |                |
| date_of_birth | date         | NO   |     | NULL    |                |
| age           | int          | NO   |     | NULL    |                |
| phone         | varchar(70)  | NO   |     | NULL    |                |
| email         | varchar(100) | NO   |     | NULL    |                |
| gender        | varchar(50)  | NO   |     | NULL    |                |
| address       | longtext     | NO   |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
9 rows in set (0.00 sec)
```

```
mysql> desc  dashboard_room;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| id             | bigint       | NO   | PRI | NULL    | auto_increment |
| Patient_id     | varchar(100) | NO   |     | NULL    |                |
| patient_name   | varchar(100) | NO   |     | NULL    |                |
| department     | varchar(100) | NO   |     | NULL    |                |
| doctor_name    | varchar(100) | NO   |     | NULL    |                |
| admission_date | date         | NO   |     | NULL    |                |
| discharge      | date         | NO   |     | NULL    |                |
| room_no        | varchar(100) | NO   |     | NULL    |                |
| room_type      | varchar(100) | NO   |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
9 rows in set (0.00 sec)
```

Overall Description:

(1) PATIENT:

* Registration:

Description - The new patient can register themselves and add their details like name, age , gender, blood group etc. The patient entry will be made in the hms database.

Pre-Condition – The patient must be a new patient, If necessary fields left by the user then prompt the user to fill the necessary fields.

Main Flow of Events:

1. Patient selects the sign up in the login module.
2. A registration form get displayed
3. Patient fills the required details.

Post Condition - Patient record is added to hms database.

* Updation:

Description-The patient should be enabled to update his/her details and the changes should reflect in the system database.

Pre-condition – The patient must be a registered patient, The patient cannot update details after treatment starts.

Main flow of events:

1. Patient logs in to the system.
2. Patient view his record
3. Patient selects update details.
4. Now the patient may change the necessary fields.
5. Pop of update details.

Post Condition - The record of the patient is updated in the system database.

*Appointment:

Description - It shows users a list of available doctors, timings, dates and enables patients to select the most suitable appointment date and doctor. The patient may also cancel the appointment.

Pre-Condition- The patient must be a registered patient, Patient can fix only one appointment for a particular department.

Main Flow of Events:

1. Patient first logs in to the system.
2. View his/her record.
3. Create a new appointment or cancel the appointment..

Post-Condition - patient details are displayed and a new appointment is fixed or an existing appointment is cancelled. The hms database is updated.

*Payment:

Description– It enables users to pay the consultant fee of Doctor online.

Pre-Condition- The patient must be a registered patient, If Patient don't wants to pay online he/she can pay by cash also.

Main Flow of Events:

1. Patient first logs in to the system.
2. View his/her record.
3. Appointment confirmed by the Doctor then goes for Payment.

Post-Condition – A Receipt will be displayed. The hms database is updated

(2)DOCTOR

Description- The doctor views the patient record/ updates his details and adds a description of the treatment given to the patient.

Pre-condition– The doctor must be a registered doctor, System does not allow the doctor to modify the qualification, hospital managed details.

Main Flow of Events:

1.Doctor logs in to the system.
2.Doctors may select a patient.
3.Patient record is displayed with treatment history.
4.Doctors add descriptions of patient treatment.
5.Doctor may select appointment details
6.Appointment Requests are displayed with a schedule.
7.Doctors confirm or cancel appointments.

Post-Condition – The patient and doctor ‘s database are updated.

(3)ADMIN

Description - The admin adds a doctor, updates doctor details and verifies payment and generates Bill/Receipt for the same.

Main Flow of Events:

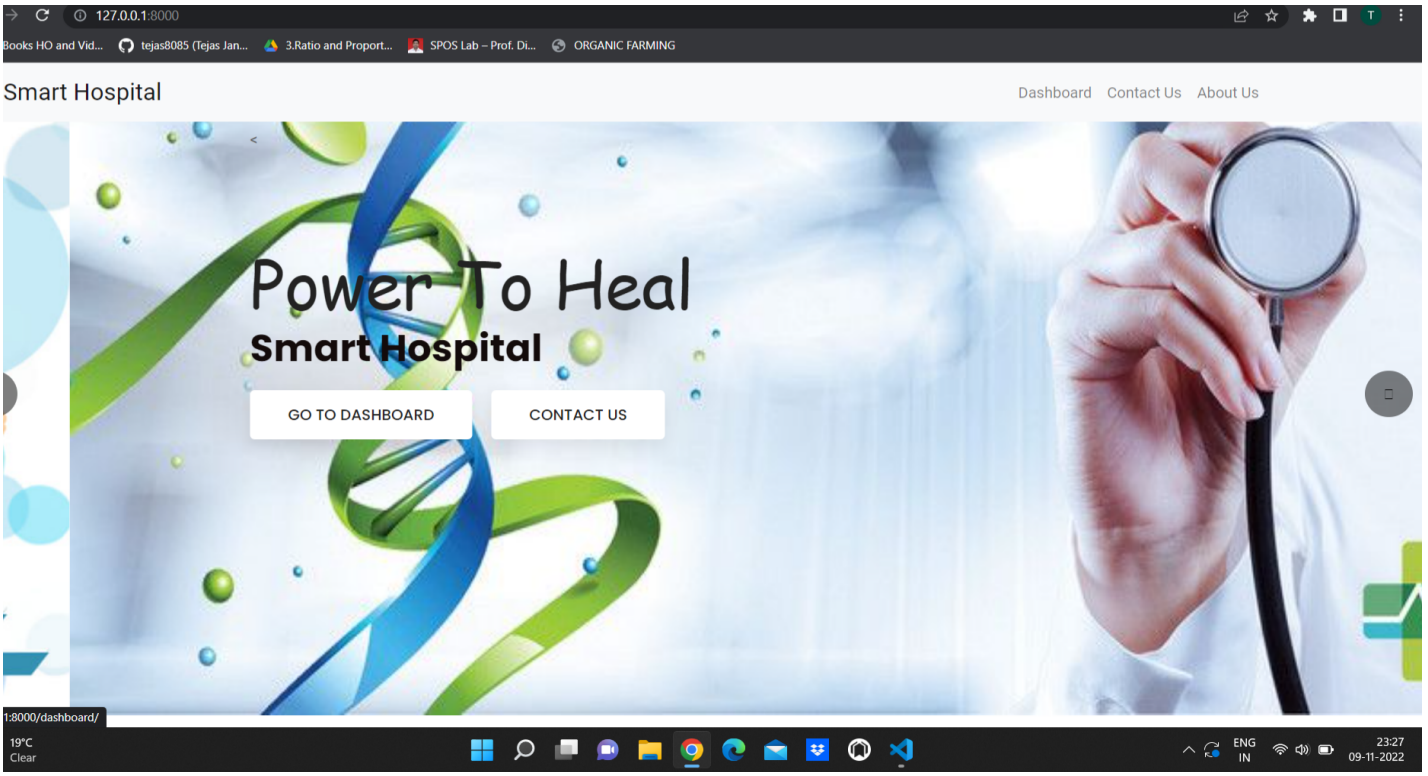
1. Admin logs in the system.
2. Admin may add a new doctor.
   2.1 admin fills the doctor’s details.
3. Admin view Doctor record.
   3.1 Admin enters the doctor and is in the system.
   3.2 Doctor details are displayed, Admin can update details.
4. Admin Verify the payment submitted by the Patient.
   4.1 Generate Bill/Receipt and confirmation message for the same.

Pre-condition - Admin must first log in with his/her credentials.

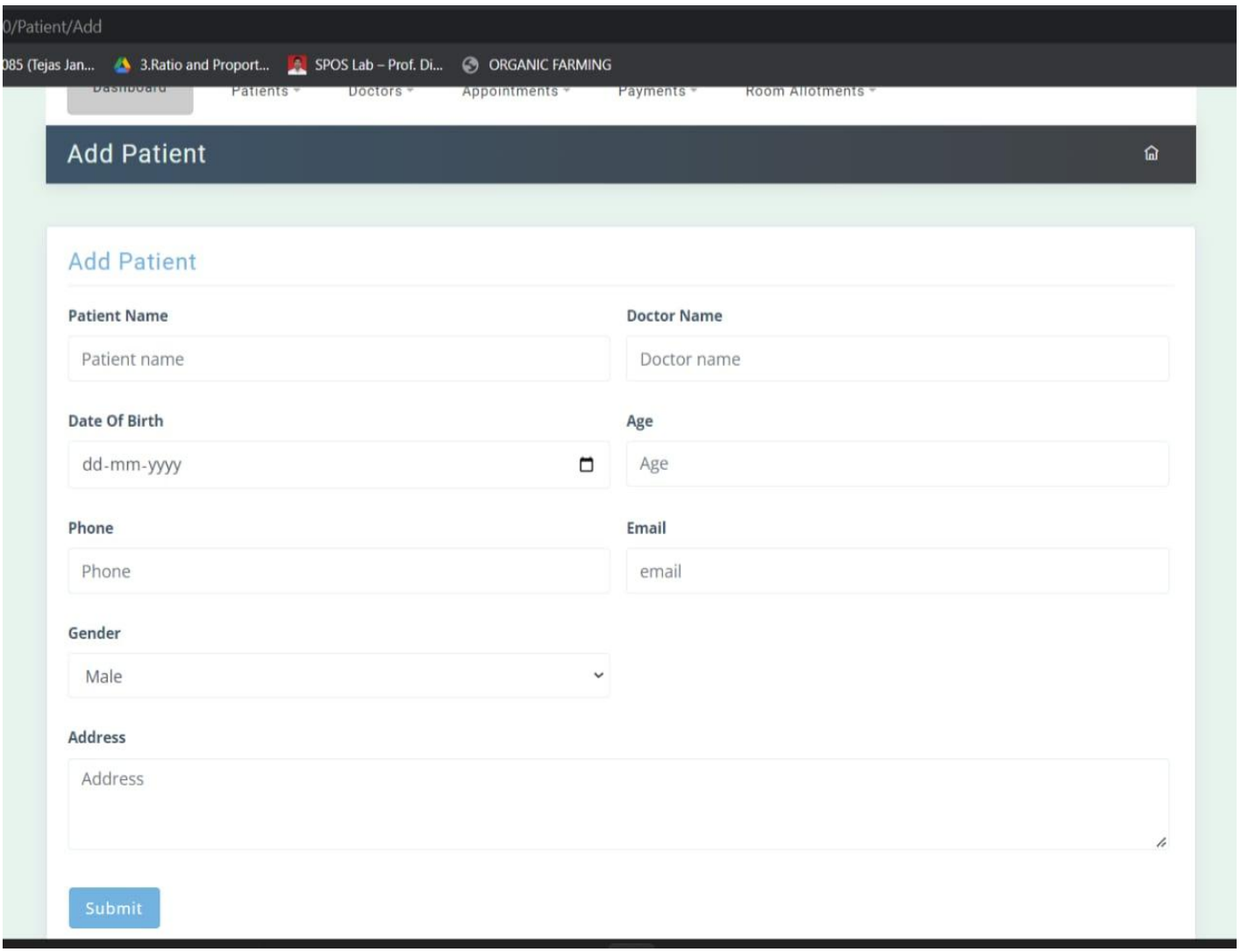
Post-condition- The system database is updated

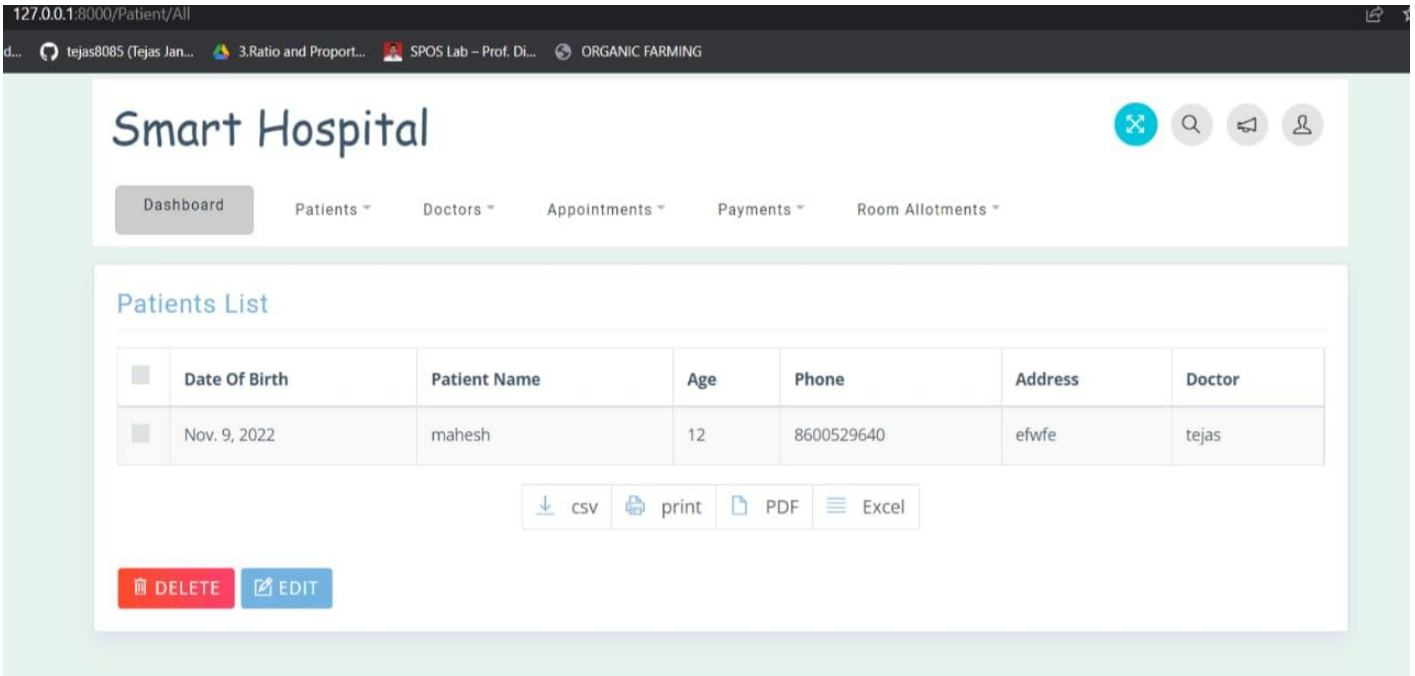
SNAPSHOTS OF WEBSITE

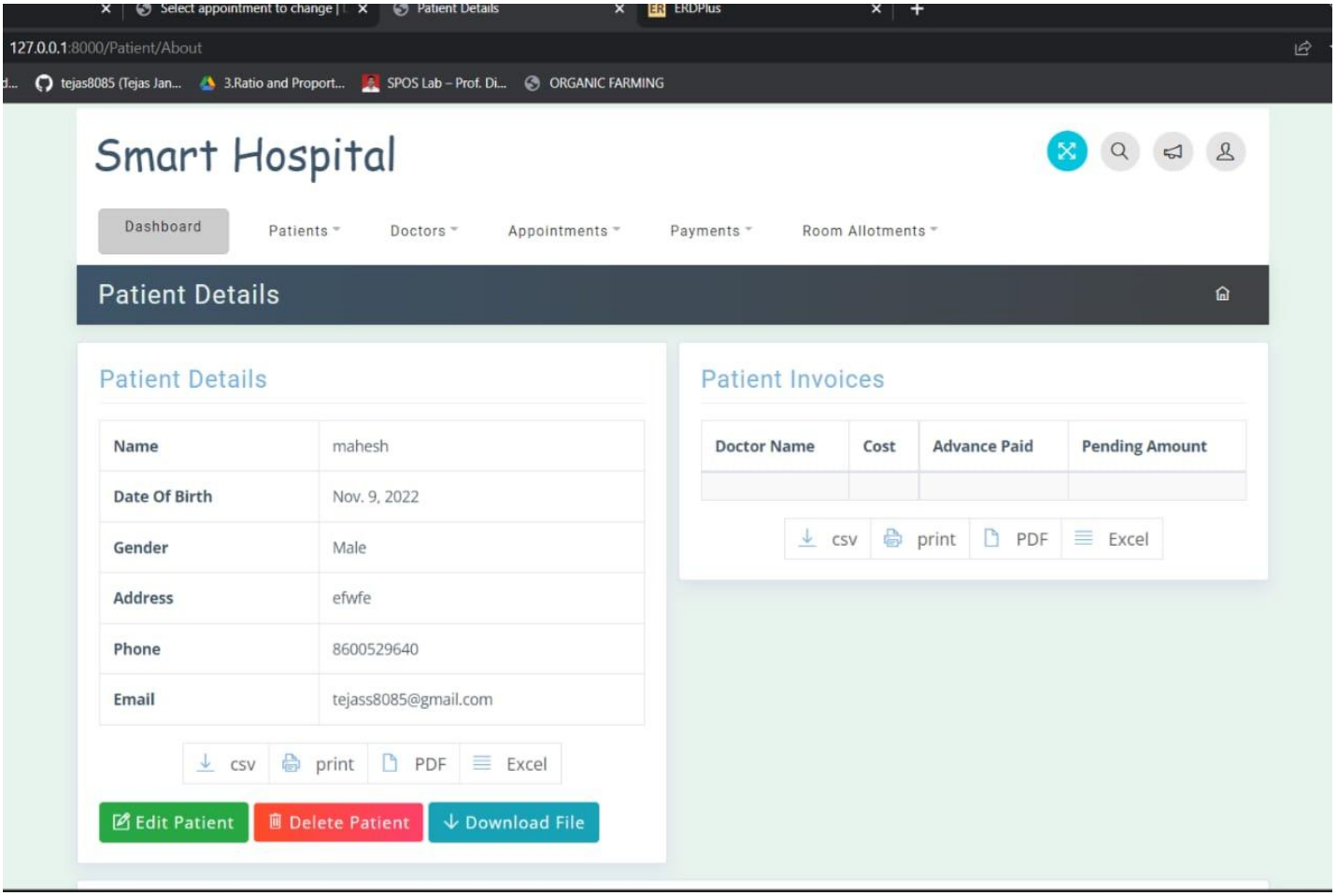
1.Home page:



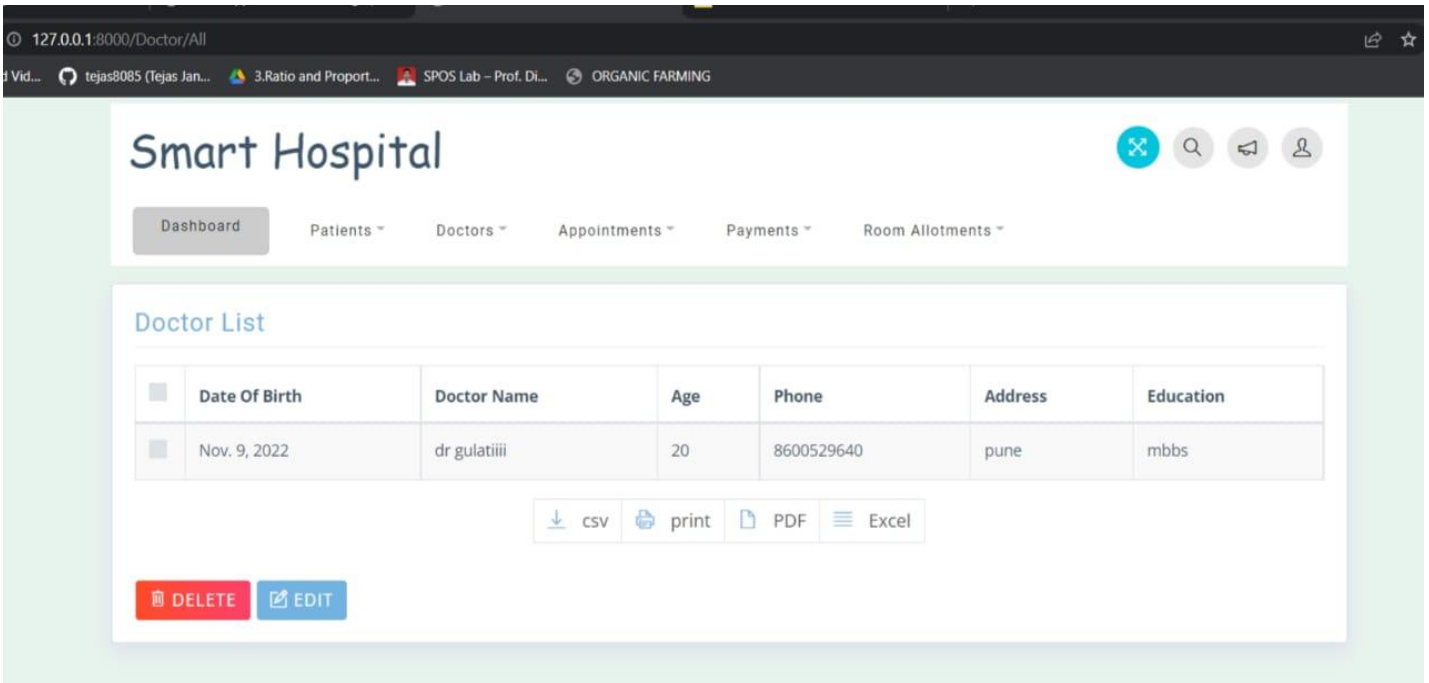
2.Patients details:



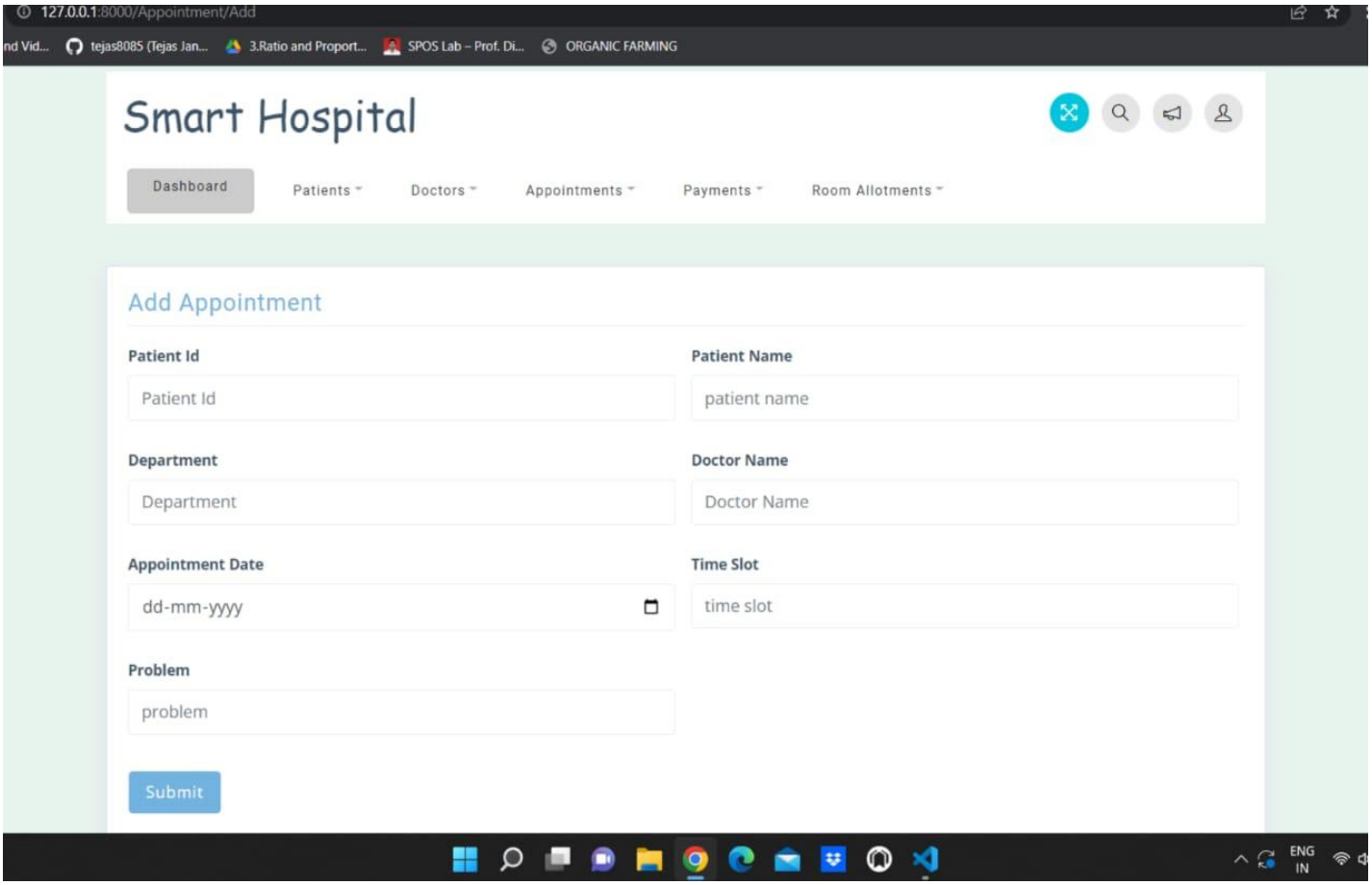


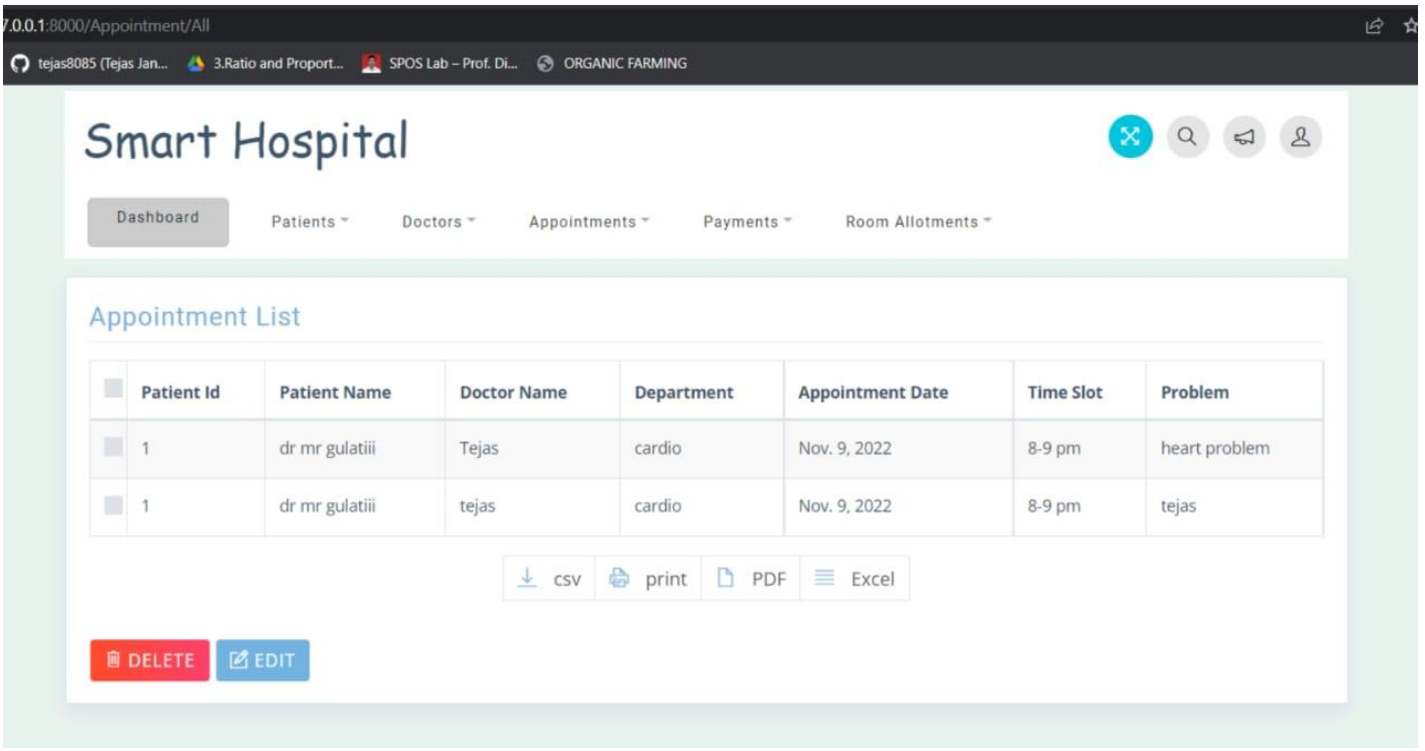


3.Doctors Details:

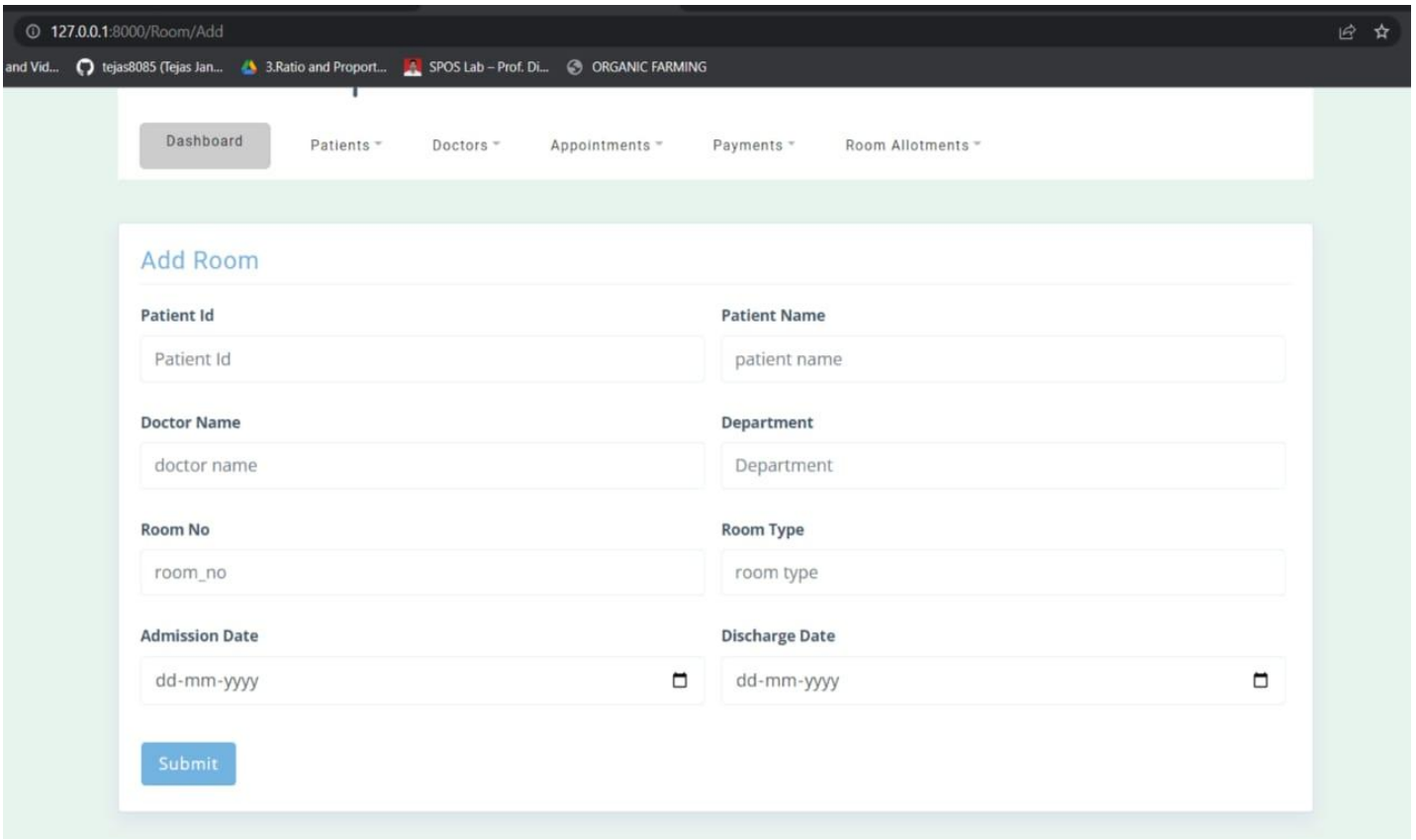


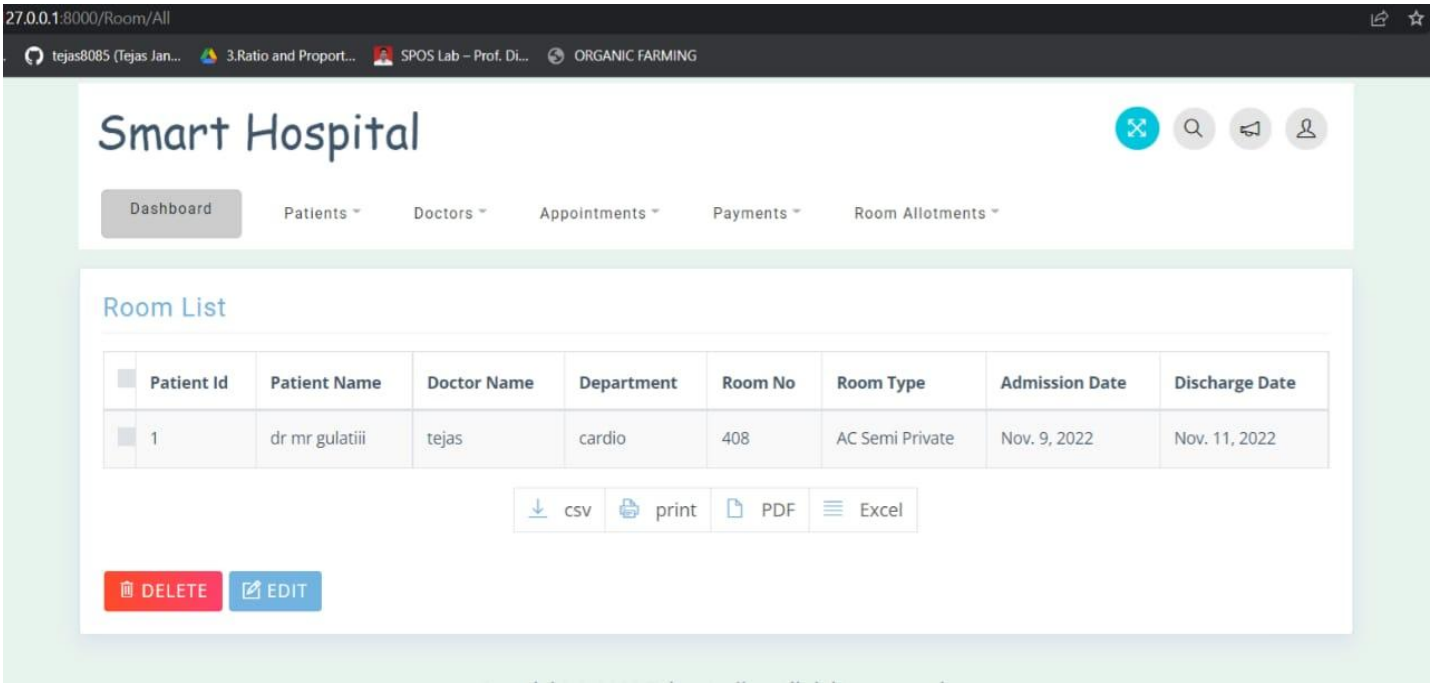
4.Booking an appointment:



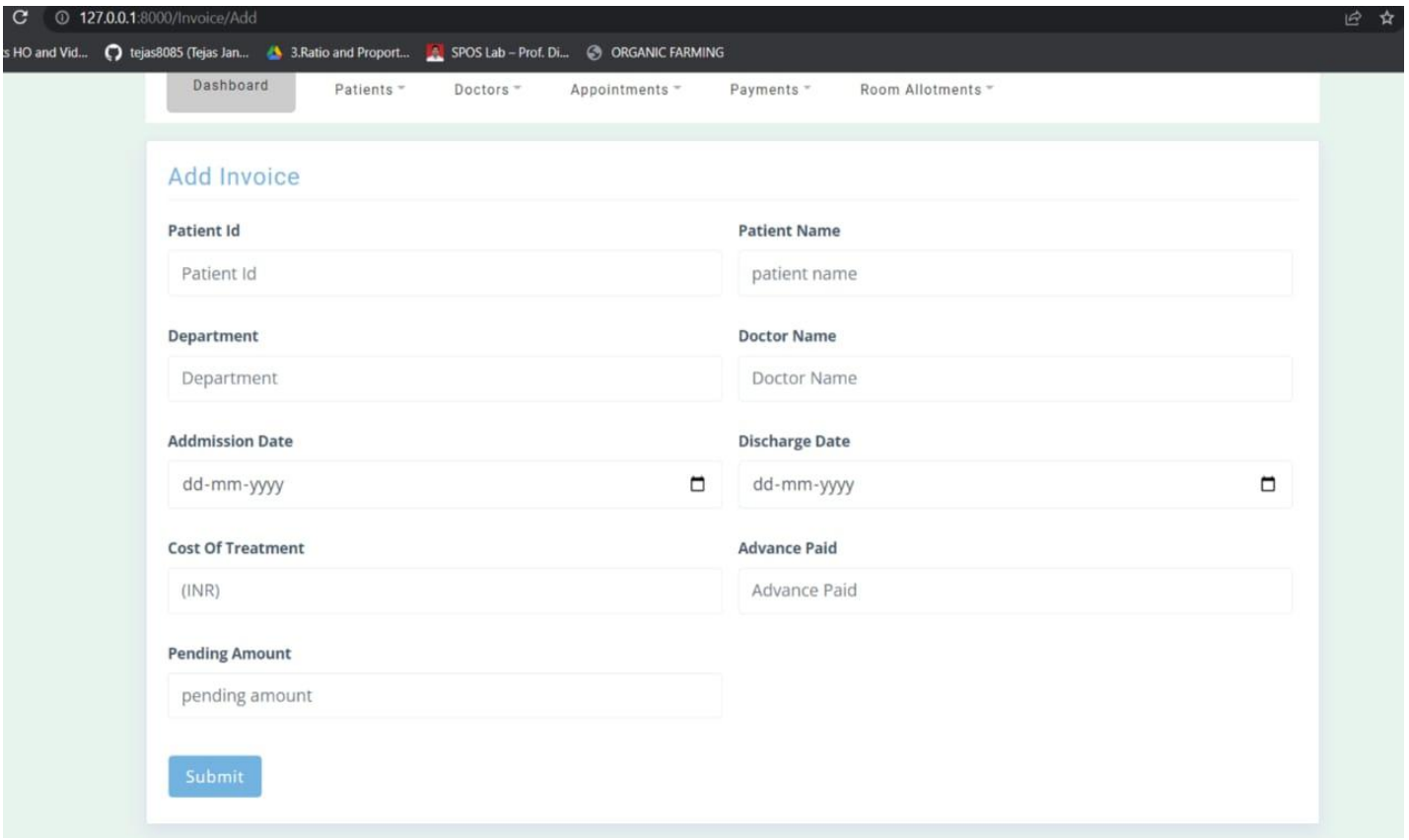


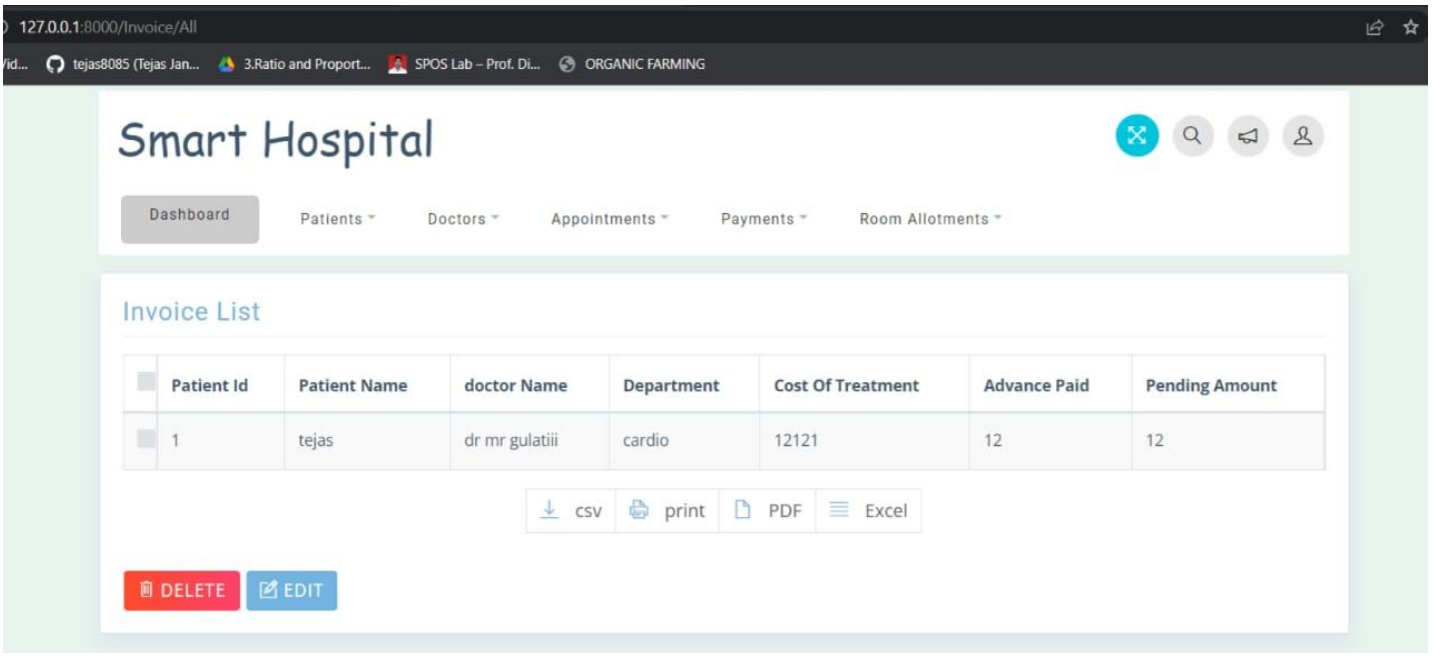
5.Rooms allocated:





6.Invoices:


127.0.0.1:8000/Invoice/Add
tejas8085 (Tejas Jan...
3.Ratio and Proport...
SPOS Lab – Prof. Di...
ORGANIC FARMING
Dashboard
Patients
Doctors
Appointments
Payments
Room Allotments
Add Invoice
Patient Id
Patient Name
Patient Id
patient name
Department
Doctor Name
Department
Doctor Name
Addmission Date
Discharge Date
dd-mm-yyyy
dd-mm-yyyy
Cost Of Treatment
Advance Paid
(INR)
Advance Paid
Pending Amount
pending amount
Submit


127.0.0.1:8000/Invoice/All
tejas8085 (Tejas Jan...
3.Ratio and Proport...
SPOS Lab – Prof. Di...
ORGANIC FARMING
Smart Hospital
Dashboard
Patients
Doctors
Appointments
Payments
Room Allotments
Invoice List
Patient Id
Patient Name
doctor Name
Department
Cost Of Treatment
Advance Paid
Pending Amount
1
tejas
dr mr gulatiii
cardio
12121
12
12
csv
print
PDF
Excel
DELETE
EDIT

SYSTEM TESTING

System testing is normally carried out in a planned manner according to the system test plan document. The system test plan identifies all testing-related activities that must be performed, specifies the schedule of testing, and allocates resources. It also lists all the test cases and the expected outputs for each test case. Here the modules are integrated in a planned manner.

Functional testing:

Functional testing refers to tests that verify a specific action or function of the code. These are usually found in the code requirements documentation, although some development methodologies work from use cases or user stories. Functional tests tend to answer the question of "can the user do this" or "does this particular feature work". Some examples of functional testing done in our project:

By checking all the login modules, it is ensured that only registered users can access all the facilities.
All the agents are verified by their IEC (Import Export Code)

Structural testing:

Structural testing is also called White box testing. This means a testing technique whereby explicit knowledge of the internal workings of the item being tested is used to select the test data. White box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the tester knows what the program is supposed to do. He or she can then see if the program diverges from its intended goal. White box testing does not account for errors caused by omission, and all visible code must also be readable.

System testing:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated “software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limiting type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing is an investigatory testing phase, where the focus is to have almost a destructive attitude

Test cases:

A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not.

- Unit Test Cases: The software is divided into different components and unit testing is performed on each of these modules. This section is repeated for all components.
- Integration Test cases: Integration testing is a part of stress testing which involves integrating the components to create a system or subsystem. It may involve testing an increment to be delivered to the customer. In integration testing, the test team has access to the system source code. The system is tested as components are integrated.
- Validation Test cases: This testing is done to see whether the integrated software is valid according to the user needs.

CONCLUSION

Working on the project was an excellent experience. It helped us to understand the importance of planning, designing and implementation so far we have learnt in our theory books. It helped us unleash our creativity while working in a team. It also realised the importance of team working, communication as a part of this project.

The project was successfully completed after a lot of effort and work hours. This project underwent number of compiling, debugging, removing errors, making it big free, adding more facilities in Hospital Management System and interactivity making it more reliable and useful

REFERENCES

www.google.com

www.wikipedia.com

http://www.github.com/

https://www.python.org/

https://www.djangoproject.com/download/

https://stackoverflow.com/

https://getbootstrap.com/