Project Team No	**03**
Project Title	**HOSPITAL MANAGEMENT SYSTEM**

Table of Contents

1.INTRODUCTION

1.1 ABSTRACT

Hospital Management System is an organized computerized system designed and programmed to deal with day to day operations and management of the hospital activities. The program can look after inpatients, outpatients, status illness, invoice. It also maintains hospital information such as doctors and department administering. The major problem for the patient nowadays to get report after consultation , many hospital managing reports in their system but it's not available to the patient when he / she is outside. In this project we are going to provide the extra facility to store the report in the database and make available from anywhere in the world.

1.2 INTRODUCTION:

The project Hospital Management system includes registration of patients, storing their details into the system, and also computerized invoice. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. User can search 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. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast. Hospital Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals. Hospital Management System is designed for multispeciality hospitals, to cover a wide range of hospital administration and management processes. It is an integrated end-to end Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow. Hospital Management System is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the hospital helps you manage your processes.

1.3 Overview

1.3.1 Objective

Hospital are the essential part of our lives, providing best medical facilities to people suffering from various ailments, which may be due to change in climatic conditions, increased work-load, emotional trauma stress etc. It is necessary for the hospitals to keep track of its day-to-day activities & records of its patients, doctors, nurses, ward boys and other staff personals that keep the hospital running smoothly & successfully. But keeping track of all the activities and their records on paper is very cumbersome and error prone. It also is very inefficient and a time-consuming process Observing the continuous increase in population and number of people visiting the hospital. Recording and maintaining all these records is highly unreliable, inefficient and error-prone. It is also not economically & technically feasible to maintain these records on paper. Thus keeping the working of the manual system as the basis of our project. We have developed an automated version of the manual system, named as “Administration support system for medical institutions”. The main aim of our project is to provide a paper-less hospital up to 90%. It also aims at providing low-cost reliable automation of the existing systems. The system also provides excellent security of data at every level of user-system interaction and also provides robust & reliable storage and backup facilities

1.3.2 Scope of the Project:-

1) Information about Patients is done by just writing the Patients name, age and gender. Whenever the Patient comes up his information is stored freshly.
2) Bills are generated by recording price for each facility provided to Patient on a separate sheet and at last they all are summed up.
3) Diagnosis information to patients is generally recorded on the document, which contains Patient information. It is destroyed after some time period to decrease the paper load in the office.
4) Immunization records of children are maintained in pre-formatted sheets, which are kept in a file.
5) Information about various diseases is not kept as any document. Doctors themselves do this job by remembering various medicines.

All this work is done manually by the admin and other operational staff and lot of papers are needed to be handled and taken care of. Doctors have to remember various medicines available for diagnosis and sometimes miss better alternatives as they can’t remember them at that time.

2 SYSTEM OVERVIEW

2.1 List of Modules for Hospital Management System :

- **Patient Management:** This module covers from the process of intake until discharge of an account of the patient’s engagement with the health-care team. Communication, empathy, examination, evaluation, diagnosis are all part of the process.
- **Doctor Management:** The management of the physicians would be included in creating this system. Through this process, the admin will have the information and transactions made by the doctors with the patients.
- **Appointment Management:** This process is a tool that helps hospital admin manage their appointments. Internet booking is one of the tools available in an appointment management solution. Booking with a mobile app.
- **Invoice Management :** invoice management module id meant to assist the admin in the payment management process. This will help the hospital with the full payment processing and accounts payable process.
- **Discharge Management:** Discharge Management is that to helps admin and doctor to easy way to get discharge patient.

These modules must be present in creating the Hospital Management to satisfy the needs in managing Hospital transactions. Through this, the management and monitoring of patients would be much easier for both hospital admin and physicians.

2.2 REQUIREMENT SPECIFICATION

2.2.1 INTRODUCTION:

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as(computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a biggerpart in driving upgrades to existing computer systems than technological advancements.

2.2.2 HARDWARE REQUIREMENTS:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

HARDWARE REQUIREMENTS FOR PRESENT PROJECT:

PROCESSOR	:	Intel dual Core ,i3
RAM	:	1 GB
HARD DISK	:	80 GB

2.2.3 SOFTWARE REQUIREMENTS:

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:

OPERATING SYSTEM	:	Windows 10
FRONT END	:	Html, css, java script.(Angular)
SERVER SIDE SCRIPT	:	Apache tomcat 8
DATABASE	:	Mysql

2.3 AUTHENTICATION AND AUTHORIZATION:

Authentication:

Every time we signed up, we likely been asked to create a username and password. Because this is such a common process now, it's become almost second-nature for some users to set up their accounts without much thought about the credentials they choose. And unfortunately, there's a lot at stake if a user chooses weak credentials.

Authentication for User name and password:

When a user first signs up for website, they're asked to choose a username and password to identify themselves. In an ideal world, the user would always pick a strong and unique password so that it's harder for an attacker to guess.

Password Rules:

When it comes to password safety, the longer and more complex the password is, the better. We think its good practice to enforce certain minimum requirements when asking users to create a new password. Of course, you have to find a balance between these requirements and user experience. If you make the sign-up process too tedious, you could be driving users away. To enforce password strength, you should define a set of rules that a password must satisfy and then enforce these with form validation.

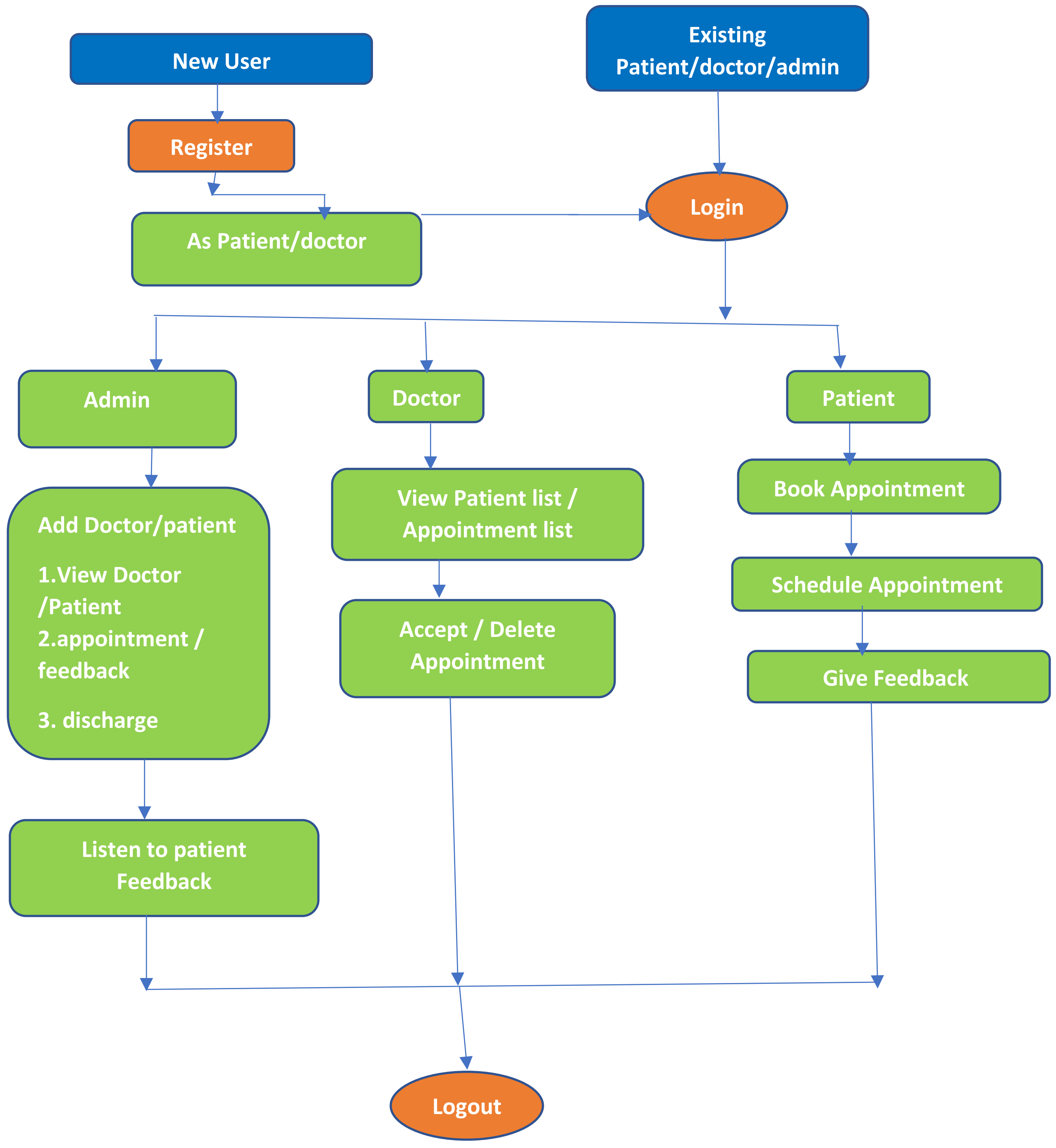
Example password strength rules:

- Minimum of 8 characters
- At least one uppercase letter
- At least one number
- At least one special character

2.3.1 Authorization:

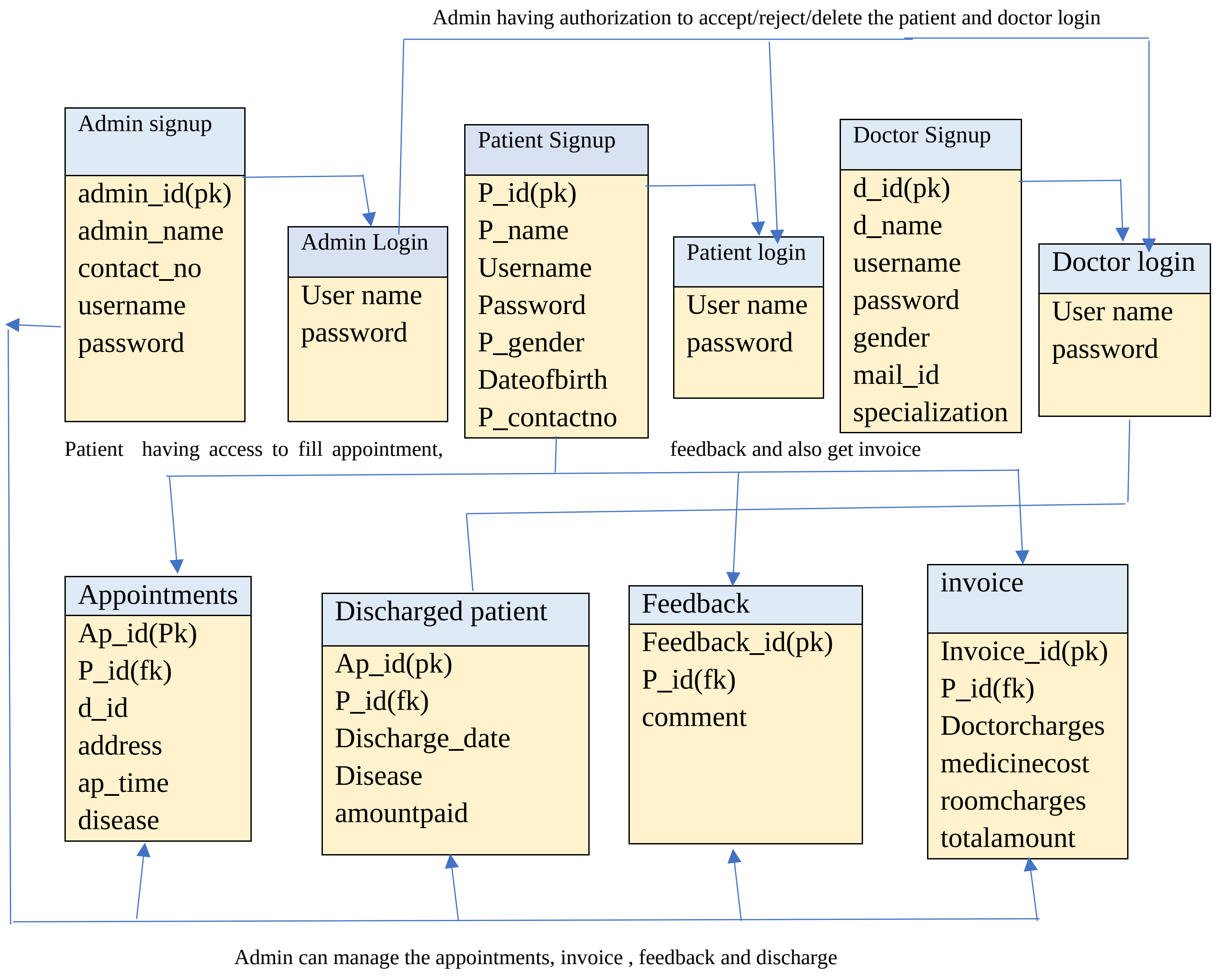
For Authorization, think of owning a hospital page. When admin log in, since he own the page. Hence, admin can post content on page, modify and even approve content from doctors and patient and others of your page that are not administrators. When users who are not administrators try to approve other people's content, they'll find out they can't. This is because they don't have the right to do that; only admins can.

2.4 Schematic Diagram



2.5 Hospital Management System Database:

The Hospital management system database design is a database design use for managing hospital functions and events. It enables the admin to register a patient for the hospital, stores their disease details into the database. Any of the staff members, doctor & admin is able to add, view, edit, update or delete data.



Patient signup table:

Column	Data Type	Length	Nulls
admin_ID	**number**		
Admin_name	**varchar**	**20**	
Contact number	**varchar**	**10**	
username	**varchar**	**20**	
password	**varchar**	**20**	

Doctor signup table:

Column Name	Data Type	Length	Nulls
d_ID	**number**	**30**	**-**
d_Name	**varchar**	**30**	**-**
username	**varchar**	**30**	**-**
password	**varchar**	**30**	**-**
gender	**varchar**	**30**	**-**
mail_id	**varchar**	**30**	
specialization	**varchar**	**30**	

Patient login:

Column Name	Data Type	Length	Nulls
username	varchar	20	-
password	varchar	20	-

Doctor login:

Column Name	Data Type	Length	Nulls
username	varchar	20	-
password	varchar	20	-

Admin login:

Column Name	Data Type	Length	Nulls
username	varchar	20	-
password	varchar	20	-

Appointment Table:

Column Name	Data Type	Length	Null
p_id	Number	20	
<u>ap_id</u>	Number	20	
d_id	Number	20	
address	Varchar	50	
ap_date	Date	20	
disease	varchar	30	

Feedback table:

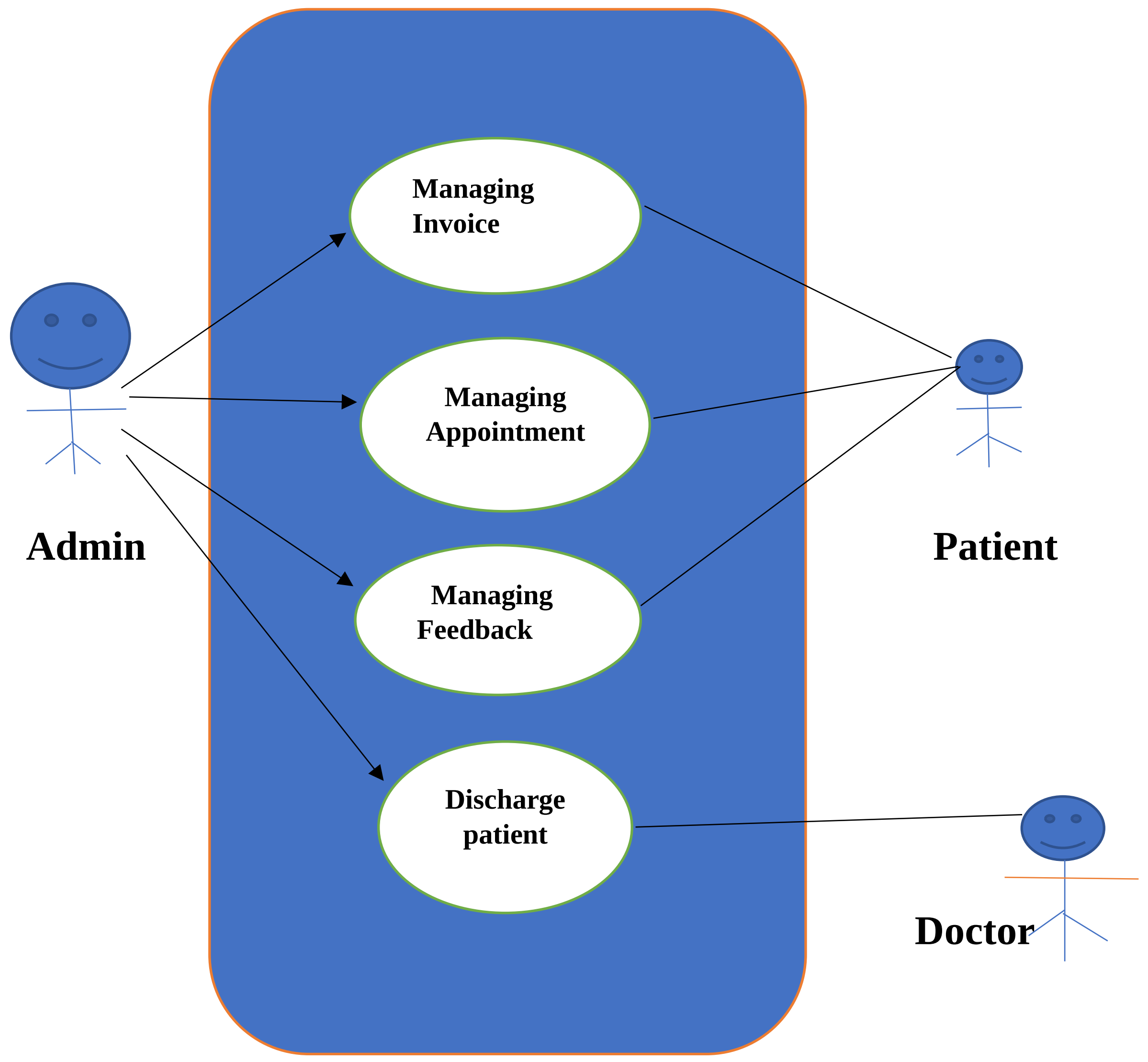
Column Name	Data Type	Length	Nulls
P_ID	number	20	
Feedback_id	Varchar	20	
comment	Varchar	100	

invoice table:

Column Name	Data Type	Length	Nulls
<u>Invoice_id</u>	varchar	30	-
P_id	varchar	30	-
Doctor charges	varchar	30	-
Medicine cost	varchar	30	-
Roomcharges	varchar	30	-
Totalamount	varchar	30	-

case diagram:

case diagram is a visual representation of how a user might interact with a program. A use case diagram depicts the system’s numerous use cases and different sorts of users. The circles or ellipses are used to depict the use cases.


Managing Invoice
Managing Appointment
Managing Feedback
Discharge patient
Admin
Patient
Doctor

3. Sub-System Details

The Hospital Management System is defined, where in all users need to login successfully before performing any of their respective operations. Below tables that provides functionality descriptions for each type of user . Against each requirement, indicative data is listed in column 'Data to include'.

3.1 Admin

The admin as a user is defined to perform below listed operations after once admin login.

Objects	Operations	Data to include	Remarks
User	register	Admin id,admin name, contact no,username,password	
Doctor list	view	Doctor id, doctor name	
Appointments	Approve/reject	Appointment id, address, date, status, time, doctor id, patient id	
Patients List	view	Patient id, patient name	
Invoice	generate	Invoice id, doctor charges, medicine cost, patient id, room charges, total amount	

3.2 Doctor

The doctor as a user is defined to perform below listed operations after once doctor login.

Objects	Operations	Data to include	Remarks
user	Register	Doctor id, doctor name, username, password, gender, mail id, specialization	
Patients Lists	view	Patient id, patient name	
Appointments	view	Patient id, disease, appointment time	
Discharge Patients		Patient id, patient name	

3.3 Patient

The Patient as a user is defined to perform below listed operations after once Patient login.

Objects	Operations	Data to include	Remarks
User	Register	Patient id, dob,contactno,gender,patient name, password	
Appointment	Register	Appointment id, address, date, status, time, doctor id, patient id	
invoice	View	Invoice id, doctor charges, medicine cost, patient id, room charges, total amount	

4 Data Organization

This section explains the data storage requirements of the Product Order Entry System and indicative data description along with suggested table (database) structure. The following section explains few of the tables (fields) with description. However, in similar approach need to be considered for all other tables.

4.1 Admin Registration table

Field Name	**Description**
Admin_id	Admin id is auto generated after registration and it is used as Login ID.
Admin_name	Name of the admin
Contact_no	10 digits mobile number
Username	Name of the admin
Password	User unique password

4.2 Patient Registration table

Field Name	**Description**
Patient_id	Patient id is auto generated after registration and it is used as Login ID.
Patient_name	Name of the Patient
Patient_gender	Give gender weather Male\Female
Username	Name of the Patient
Password	User unique password
DateOfBirth	Birth date of the patient
Patient contact number	10 digits patient mobile number

4.3 Doctor Registration table

Field Name	**Description**
Doctor_id	Doctor id is auto generated after registration and it is used as Login ID.
Doctor_name	Name of the doctor
Contact_no	10 digits mobile number
Username	Name of the Doctor
Password	User unique password
Gender	Mention whether doctor male\female
Mail_id	Doctor Email id
Specialization	Doctor Specialization or expert

4.4 Appointment table

Field Name	**Description**
Appointment_id	Appointment id is auto generated after appointment filled successfully.
Patient_id	Id of patient
Doctor_id	Id of the doctor to whom you were referred
Address	Complete Address of the Patient
Appointment time	At what time the patient want to schedule the appointment
disease	Mention the type of disease

4.5 Invoice table

Field Name	**Description**
Invoice_id	Once after the paying bill started it creates invoice id
Patient_id	Id of patient
Doctor_charges	Charges to the doctor how much to be paid
Medicine cost	Complete cost of the medicine
Room charges	If patient uses the room, the cost of the room for all the days from the time of hosptialistion
Total Amount	Total amount to be paid

4.6 Feedback table

Field Name	Description
Patient_id	Id of patient
comments	Feedback to be given In 50 words

4.7 Discharge Table

Field Name	Description
Appointment_id	Appointment id which was given
Patient_id	Id of patient
Discharge_date	The date of discharge
Disease	Patient Disease
Amount paid	Clearance of bill amount

5 REST APIs to be Built.

Create following REST resources which are required in the application,

1. Creating Admin Entity: Create Spring Boot with Microservices Application with Spring Data JPA

Technology stack:

· Spring Boot

· Spring REST

· Spring Data JPA

Here will have multiple layers into the application:

1. Create an Entity: Admin

2. Create a AdminRepository interface and will make use of Spring Data JPA

a) Will have findByAdminme method.

b) Add the Admin details

3. Create a Admin Service class and will expose all these services.

4. Finally, create a AdminRestController will have the following Uri's:

GET---http://localhost:9080/api/v1/admin

http://localhost:9080/api/v1/adminsignup/{adminsignupId}

http://localhost:9080/api/v1/adminlogin

http://localhost:9080/api/v1/appointments

http://localhost:9080/api/v1/appointments/12

http://localhost:9080/api/v1/dischargepatientdetails

http://localhost:9080/api/v1/dischargedpatient/22

http://localhost:9080/api/v1/doctors

http://localhost:9080/api/v1/doctors/45

http://localhost:9080/api/v1/doctorlogin

http://localhost:8080/api/v1/feedbacks

http://localhost:8080/api/v1/feedbacks/1

http://localhost:9080/api/v1/invoice

http://localhost:9080/api/v1/invoice/3

http://localhost:8080/api/v1/patients

http://localhost:8080/api/v1/patient/23

POST---http://localhost:9080/api/v1/admin

http://localhost:9080/api/v1/appointments

http://localhost:9080/api/v1/patientdischargedetails

http://localhost:9080/api/v1/doctors

http://localhost:9080/api/v1/feedbacks

http://localhost:9080/api/v1/invoice

http://localhost:8080/api/v1/patient

PUT---http://localhost:9080/api/v1/admin/2

http://localhost:9080/api/v1/appointments/12

http://localhost:9080/api/v1/dischargedpatient/22

http://localhost:9080/api/v1/doctors/2

http://localhost:9080/api/v1/feedbacks/3

http://localhost:9080/api/v1/invoice/4

http://localhost:8080/api/v1/patient/89

DELETE---http://localhost:9080/api/v1/admin/1

http://localhost:9080/api/v1/appointments/12

http://localhost:9080/api/v1/dischargedpatient/1

http://localhost:9080/api/v1/doctors/1

http://localhost:8080/api/v1/feedbacks/1

http://localhost:9080/api/v1/invoice/4

http://localhost:8080/api/v1/patient/6

2. Creating Appointment Entity:

Creating Appointment Entity:

Build a RESTful resource for Appointment manipulations, where CRUD operations to be carried out. Here will have multiple layers into the application:

1. Create an Entity: Appointment

2. Create a AppointmentRepository interface and will make use of Spring Data JPA

a) Will have findByAppointmentName method.

b) Add the Appointment details method.

c) Will have deleteAppointmentById method.

d) Will have findAllAppointments method.

3. Create a AppointmentService class and will expose all these services.

4. Finally, create a AppointmentRestController will have the following Uri's:

Appoitment: getall :http://localhost:8088/api/v1/appointments

get:http://localhost:8088/api/v1/appointments/{id}

save[post]:http://localhost:8088/api/v1/appointments

delete[delete]:http://localhost:8088/api/v1/appointments/{id}

update[put]: http://localhost:8088/api/v1/appointments/{id}

3. Creating Feedback Entity:

Build a RESTful resource for Feedback, Here will have multiple layers into the application:

1. Create an Entity: Feedback

2. Create a FeedbackRepository interface and will make use of Spring Data JPA

a) Will have findByFeedback method.

b) Add the Feedback details method.

c) Will have deleteFeedbackById method.

d) Will have findAllFeedback method.

3. Create a FeedbackService class and will expose all these services.

4. Finally, create a FeedbackRestController will have the following Uri's:

getall :http://localhost:8088/api/v1/feedbacks

get:http://localhost:8088/api/v1/feedbacks/{id}

save[post]:http://localhost:8088/api/v1/feedbacks

delete[delete]:http://localhost:8088/api/v1/feedbacks/{id}

update[put]: http://localhost:8088/api/v1/feedbacks/{id}

4. Creating Patient Entity:

Build a RESTful resource for Patient manipulations, Here will have multiple layers into the application:

1. Create an Entity: Patient

2. Create a PatientRepository interface and will make use of Spring Data JPA

a) Will have findByPatientName method.

b) Add the Patient details method.

c) Will have deletePatientById method.

d) Will have findAllPatient method.

3. Create a PatientService class and will expose all these services.

4. Finally, create a PatientRestController will have the following Uri’s:

getall :http://localhost:8088/api/v1/patients

get:http://localhost:8088/api/v1/patient/{id}

save[post]:http://localhost:8088/api/v1/patient

delete[delete]:http://localhost:8088/api/v1/patient/{id}

update[put]: http://localhost:8088/api/v1/patient/{id}

5. Creating Doctor Entity:

Build a RESTful resource for Doctor manipulations, Here will have multiple layers into the application:

1. Create an Entity: Doctor

2. Create a DoctorRepository interface and will make use of Spring Data JPA

a) Will have findByDoctorName method.

b) Add the Doctor details method.

c) Will have deleteDoctorById method.

d) Will have findAllDoctor method.

3. Create a DoctorService class and will expose all these services.

4. Finally, create a DoctorRestController will have the following Uri's:

getall :http://localhost:8088/api/v1/doctors

get:http://localhost:8088/api/v1/doctors/{id}

save[post]:http://localhost:8088/api/v1/doctors

delete[delete]:http://localhost:8088/api/v1/doctors/{id}

update[put]: http://localhost:8088/api/v1/doctors/{id}

6. Assumptions :

●Each user(Admin,doctor,patient) must have a valid user id and password

●Server must be running for the system to function

●Users must log in to the system to access any record.

●Only the Administrator can delete records.

Only admin can approve appointment, discharge, generate invoice.

7. Advantages:

1. Digital Medical Records
2. Less time consuming
3. Insurance claim functions
4. Patient self-service
5. Financial control and taxpaying
6. Staff interactions

Digital medical records :

The hospital database includes all the necessary patient data. The disease history, test results, prescribed treatment can be accessed by doctors without much delay in order to make an accurate diagnosis and monitor the patient's health. It enables lower risks of mistakes.

Less time consuming:

As the services and interactions are improved in all possible ways, everything is being planned with greater precision. It saves the time of all the system users and provides them with up-to-date information.

Insurance claims processing:

Integration with health insurance services improves the experience of the patients and brings benefits to the institution. It allows you to be innovative and helps both the patient and hospital to handle many aspects of the insurance process successfully.

Patient self-service:

Patients have their own system accounts where the list of various actions can be performed. They are able to make online requests or reservation, receive the test results, receive the consultation of the medical specialists and many more.

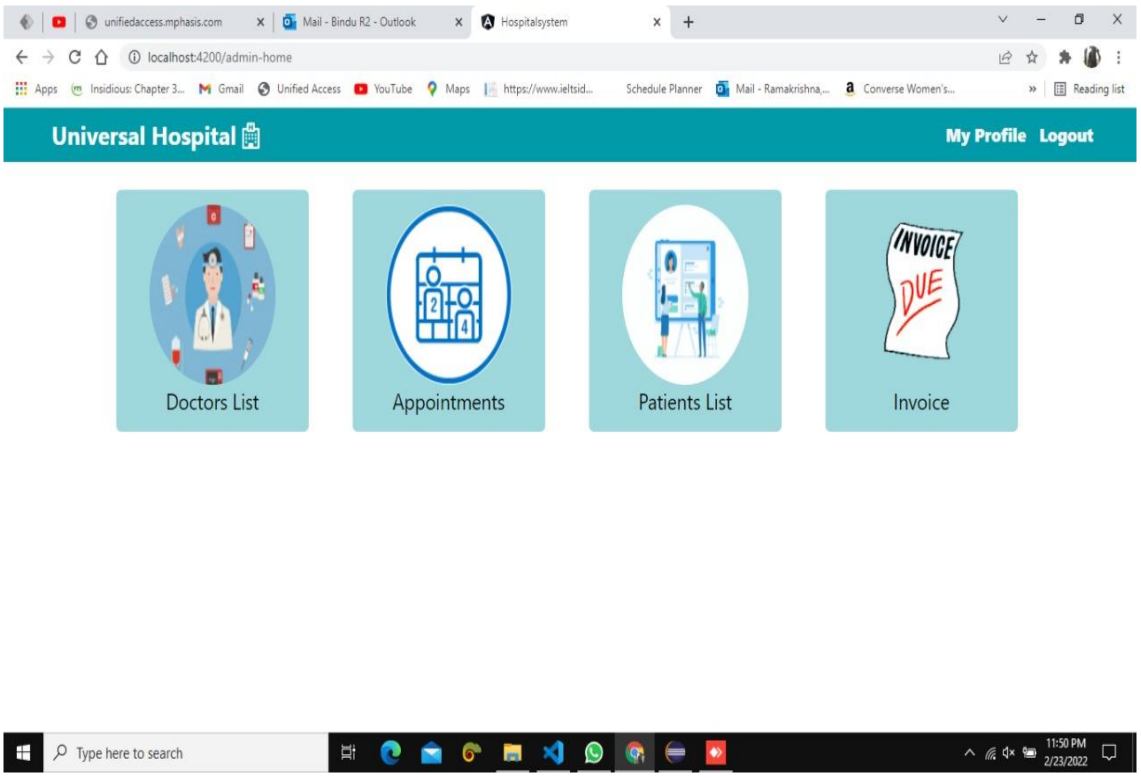
Financial control and tax planning:

The management has the ability to monitor different financial operations including expenses, profits, and losses, paying bills and taxes, in and outpatient billing. The financial awareness helps to analyze business prospects quite clear and move in the right direction

8 Output Screenshots Of HMS

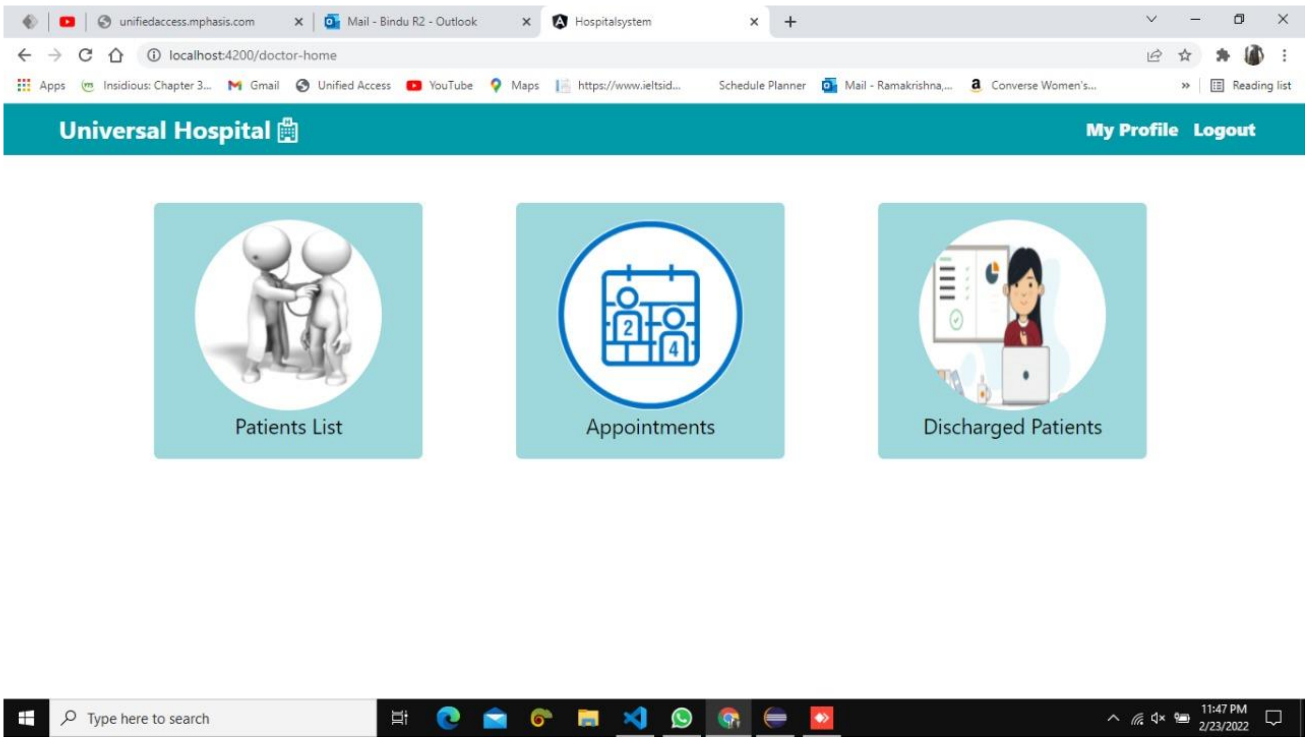
Admin: home

1. Once of after successful login admin will get homepage
2. Where we can view Doctor list, Appointment, Patient list, invoice modules.



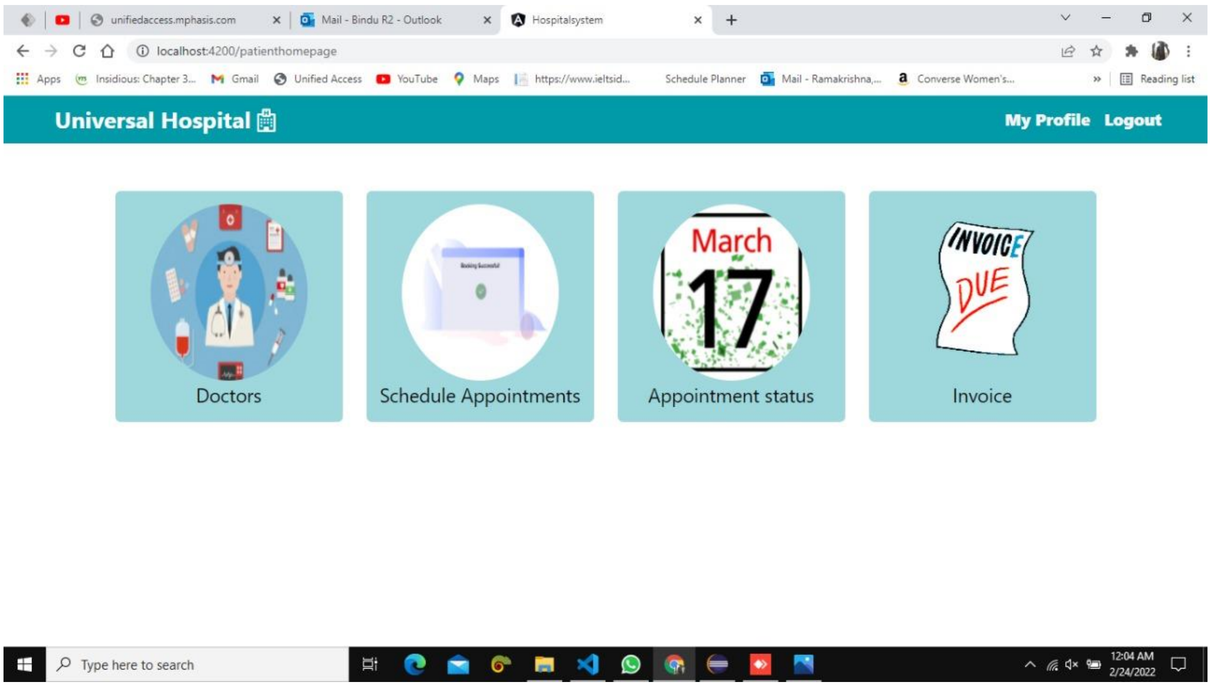
Doctor :home

1. Once of after successful login doctor will get homepage
2. Where we can view Patient list, Appointments, Discharged patients.



Patient: home

1. Once of after successful login Patient will get homepage
2. Where we can view Doctors, schedule Appointments, Appointment Status, invoice.



Conclusion:

Hospital management systems allows us the ability to optimize and digitize all the processes within the institution, which will help to improve customer service, reduce process costs, streamline the search of medical records, bills, patients, doctors, etc.; thus, having a database of each module implemented. HMS will contributed enormously to better health and influenced the lives and well-being of billions of humans.