CS 353

Database Systems

Design Report

Hospital Management Database System

Group 27

Osama Tanveer	21801147	Section 1
Adeem Adil Khatri	21801174	Section 2
Mohammad Elham Amin	21701543	Section 1
Mohammed Sameer Yaseen	21801331	Section 1

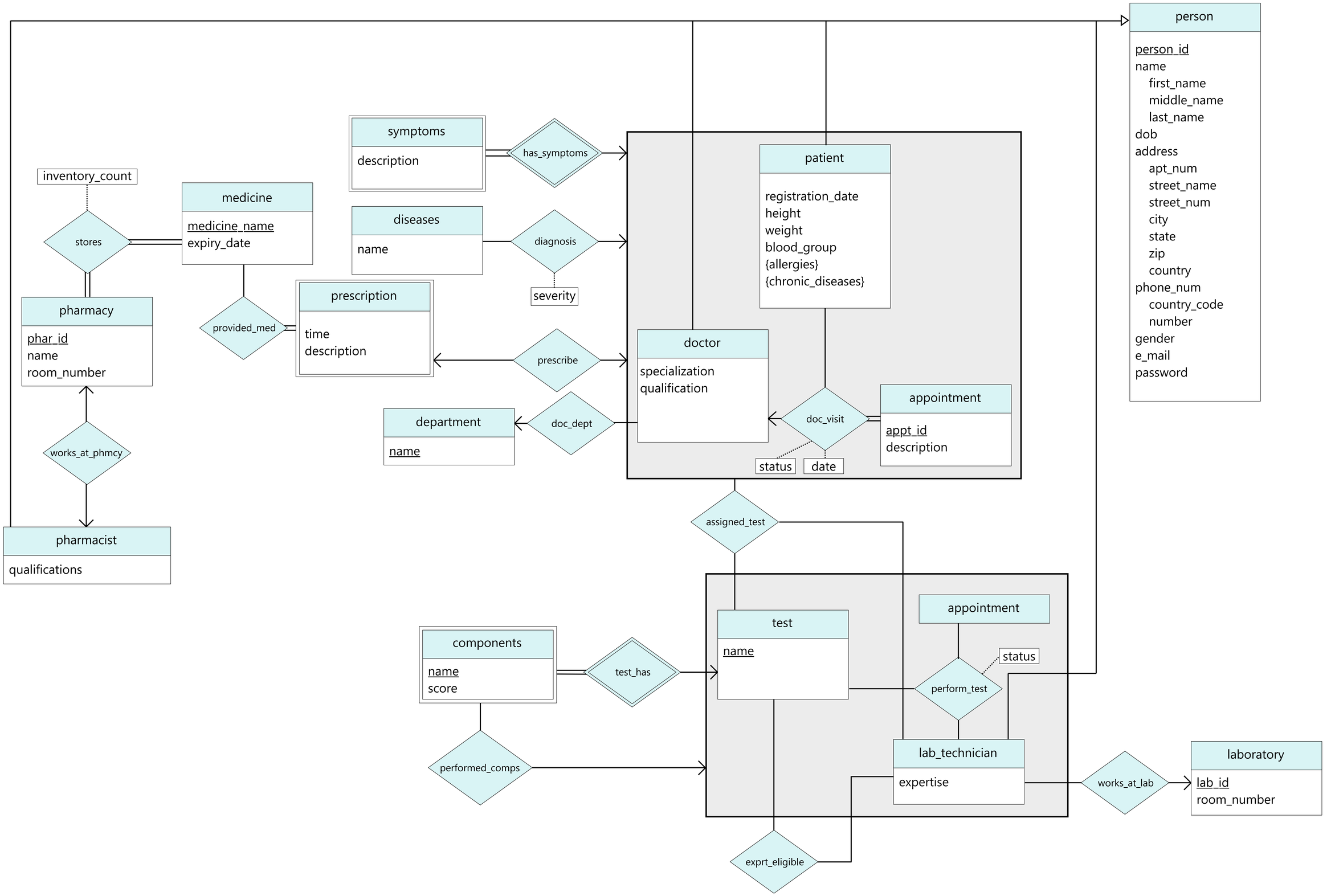


Figure 1: The revised Entity-Relation (E-R) Model

Relations Schemas

1. person

Model:
person(person_id, first_name, middle_name, last_name, dob, apt_num, street_name, street_num, city, state, zip, country, country_code, number, gender, e_mail, password)

Candidate Keys:
{(person_id)}

Primary Key:
{person_id}

Foreign Keys:
None.

Table Declaration:

```
CREATE TABLE person(
        person_id VARCHAR(50),
        first_name VARCHAR(50) NOT NULL,
        middle_name VARCHAR(50),
        last_name VARCHAR(50) NOT NULL,
        dob DATE,
        apt_num INT,
        street_name VARCHAR(100),
        street_num INT,
        city VARCHAR(25),
        state VARCHAR(25),
        zip INT,
        country VARCHAR(25),
        country_code  VARCHAR(5),
        number VARCHAR(20),
        gender VARCHAR(10),
        e_mail VARCHAR(50) NOT NULL,
        password VARCHAR(100) NOT NULL,
        PRIMARY KEY (person_id)
);
```

2. pharmacist

Model:
pharmacist(ph_id, qualifications)

Candidate Keys:
{(ph_id)}

Primary Key:
{ph_id}

Foreign Keys:
ph_id references person.peson_id

Table Declaration:

```
CREATE TABLE pharmacist(
        ph_id VARCHAR(50),
        qualifications VARCHAR(100) NOT NULL,
        PRIMARY KEY(ph_id),
        FOREIGN KEY (ph_id) REFERENCES person(person_id)
);
```

3. patient

Model:
patient(pid, height, weight, blood_group, registration_date);

Candidate Keys:
{(pid)}

Primary Key:
{pid}

Foreign Keys:
pid references person.person_id

Table Declaration:

```
CREATE TABLE patient(
        pid VARCHAR(50),
        height NUMERIC(5, 2),
        weight NUMERIC(5, 2),
        blood_group CHAR(3),
        registration_date date NOT NULL,
```

```
    PRIMARY KEY (pid),
    FOREIGN KEY (pid) REFERENCES person(id)
);
```

4. patient_allergies

Model:
patient_allergies(pid, allergy);

Candidate Keys:
{(pid, allergy)}

Primary Key:
{pid, allergy}

Foreign Keys:
pid references patient.pid

Table Declaration:

```
CREATE TABLE patient_allergies(
    pid VARCHAR(50),
    allergy VARCHAR(100) NOT NULL,
    PRIMARY KEY(pid),
    FOREIGN KEY (pid) REFERENCES patient(pid)
);
```

5. patient_chronic_diseases

Model:
patient_chronic_disease(pid, disease);

Candidate Keys:
{(pid, disease)}

Primary Key:
{pid, disease}

Foreign Keys:
pid references patient.pid

Table Declaration:

```
CREATE TABLE patient_chronic_disease(
    pid VARCHAR(50),
    disease VARCHAR(100) NOT NULL,
    PRIMARY KEY(pid),
```

```
        FOREIGN KEY (pid) REFERENCES patient(pid)
);
```

6. doctor

Model:
doctor(d_id, dept_name, specialization, qualification);

Candidate Keys:
{(d_id)}

Primary Key:
{d_id}

Foreign Keys:
d_id references person.person_id

Table Declaration:

```
CREATE TABLE doctor(
        d_id VARCHAR(50),
        dept_name VARCHAR(50) NOT NULL,
        specialization VARCHAR(50) NOT NULL,
        qualification VARCHAR(200) NOT NULL,
        PRIMARY KEY(d_id),
        FOREIGN KEY (d_id) REFERENCES person(person_id),
        FOREIGN KEY (dept_name) REFERENCES department(name)
);
```

7. lab_technician

Model:
lab_technician(lt_id, expertise);

Candidate Keys:
{(lt_id)}

Primary Key:
{lt_id}

Foreign Keys:
lt_id references person.person_id

Table Declaration:

```
CREATE TABLE lab_technician(
        lt_id VARCHAR(50),
```

```
        expertise VARCHAR(0) NOT NULL,
        PRIMARY KEY(lt_id),
        FOREIGN KEY (lt_id) REFERENCES person(person_id)
);
```

8. pharmacy

Model:
pharmacy(phar_id, name, room_number)

Candidate Keys:
{(phar_id)}

Primary Key:
{phar_id}

Foreign Keys:
None

Table Declaration:

```
CREATE TABLE pharmacy(
        phar_id VARCHAR(50) NOT NULL,
        name VARCHAR(50) NOT NULL,
        room_number INT,
        PRIMARY KEY(phar_id),
);
```

9. works_at_pharmacy

Model:
works_at_phar(phar_id, ph_id);

Candidate Keys:
{(phar_id, ph_id)}

Primary Key:
{phar_id, ph_id}

Foreign Keys:
ph_id references person.person_id
phar_id references pharmacy.phar_id

Table Declaration:

```
CREATE TABLE works_at_phar(
```

```
    ph_id VARCHAR(50) NOT NULL,
    phar_id VARCHAR(50),
    PRIMARY KEY(phar_id, ph_id),
    FOREIGN KEY (phar_id) REFERENCES pharmacy(phar_id),
    FOREIGN KEY (ph_id) REFERENCES person(person_id)
);
```

10. medicine

Model:
medicine(med_name, expiray_date)

Candidate Keys:
{(med_name)}

Primary Key:
{med_name}

Foreign Keys:
None.

Table Declaration:

```
CREATE TABLE medicine(
    med_name VARCHAR(50),
    expiry_date DATE,
    PRIMARY KEY(med_name),
);
```

11. stored_medicine

Model:
stored_medicine(med_name, phar_id, inventory_count)

Candidate Keys:
{ (med_name, phar_id)}

Primary Key:
{med_name, phar_id}

Foreign Keys:
med_name references medicine.med_name
phar_id references pharmacy.phar_id

Table Declaration:

```
CREATE TABLE stored_medicine(
        med_name VARCHAR(50),
        phar_id INT NOT NULL,
        inventory_count INT,
        PRIMARY KEY(med_name, phar_id),
        FOREIGN KEY (med_name) REFERENCES medicine(med_name),
        FOREIGN KEY (phar_id) REFERENCES pharmacy(phar_id)
);
```

12. presc_medicine

Model:
presc_medicine(med_name, time, presc_details)

Candidate Keys:
{ (med_name, presc_details)}

Primary Key:
{med_name, presc_deatils}

Foreign Keys:
med_name references medicine.med_name
presc_details references prescription.description

Table Declaration:

```
CREATE TABLE presc_medicine(
        med_name VARCHAR(50),
        presc_detals VARCHAR(50),
        PRIMARY KEY (med_name, presc_Details),
        FOREIGN KEY (med_name) REFERENCES medicine(med_name),
        FOREIGN KEY (presc_details) REFERENCES prescription(description)
);
```

13. appointment

Model:
appointment(appt_id, description);

Candidate Keys:
{(appt_id)}

Primary Key:

appt_id

Foreign Keys:

Table Declaration:

```
CREATE TABLE appointment(
        appt_id VARCHAR(50),
        description VARCHAR(200),
        PRIMARY KEY (appt_id)
);
```

14. doc_visit

Model:
doc_visit(d_id, p_id, appt_id, date, status);

Candidate Keys:
{(appt_id)}

Primary Key:
{appt_id}

Foreign Keys:
d_id references doctor.d_id
p_id references patient.p_id
appt_id references appointment.appt_id

Table Declaration:

```
CREATE TABLE doc_visit(
        d_id VARCHAR(50),
        p_id VARCHAR(50),
        appt_id VARCHAR(50),
        date DATE,
        status VARCHAR(20),
        PRIMARY KEY (appt_id),
        FOREIGN KEY (d_id) REFERENCES doctor(d_id),
        FOREIGN KEY (p_id) REFERENCES patient(p_id),
        FOREIGN KEY (appt_id) REFERENCES appointment(appt_id)
);
```

15. works_at_lab

Model:
works_at_lab(lt_id, lab_id);

Candidate Keys:
{(lt_id)}

Primary Key:
lt_id

Foreign Keys:
lt_id references person.person_id
lab_id references person.lab_id

Table Declaration:
```
CREATE TABLE works_at_lab(
        lab_id VARCHAR(50),
        lt_id VARCHAR(50) NOT NULL,
        PRIMARY KEY(lab_id),
        FOREIGN KEY (lab_id) REFERENCES person(lab_id),
        FOREIGN KEY (lt_id) REFERENCES person(person_id)
);
```

16. laboratory

Model:
laboratory(lab_id, room_number);

Candidate Keys:
{(lab_id)}

Primary Key:
lab_id

Foreign Keys:
None.

Table Declaration:
```
CREATE TABLE laboratory(
        lab_id VARCHAR(50),
        room_number INT NOT NULL,
        PRIMARY KEY(lab_id)
);
```

17. test

Model:
test(name, expertise_required);

Candidate Keys:
{(name)}

Primary Key:
name

Foreign Keys:
None.

Table Declaration:
```
CREATE TABLE test(
        name VARCHAR(100),
        expertise_required VARCHAR(20),
        PRIMARY KEY(name)
);
```

18. components

Model:
components(t_name, c_name, score);

Candidate Keys:
{(t_name, c_name)}

Primary Key:
{t_name, c_name}

Foreign Keys:
t_name references test.name

Table Declaration:
```
CREATE TABLE components(
        t_name VARCHAR(100),
        c_name VARCHAR(100),
        score INT NOT NULL,
        PRIMARY KEY(t_name, c_name),
        FOREIGN KEY t_name REFERENCES test(name)
);
```

19. assigned_test

Model:
assigned_test(appt_id, test_name, lt_id);

Candidate Keys:

{(appt_id, test_name, lt_id)}

Primary Key:
{appt_id, test_name, lt_id}

Foreign Keys:
appt_id references appointment.appt_id
lt_id references lab_technician.lt_id
test_name references test.name

Table Declaration:

```
CREATE TABLE appointment(
        lt_id VARCHAR(50),
        appt_id VARCHAR(50),
        test_name VARCHAR(100),
        PRIMARY KEY (lt_id, appt_id, test_name),
        FOREIGN KEY appt_id REFERENCES appointment(appt_id),
        FOREIGN KEY lt_id REFERENCES lab_technician(lt_id),
        FOREIGN KEY test_name REFERENCES test(name)
);
```

20. perform_test

Model:
perform_test(appt_id, t_name, lt_id, status);

Candidate Keys:
{(appt_id, t_name, lt_id)}

Primary Key:
{appt_id, t_name, lt_id}

Foreign Keys:
p_id references appointment.apt_id
t_name references test.name
lt_id references lab_technician.ld_id

Table Declaration:

```
CREATE TABLE appointment(
        appt_id VARCHAR(50),
        t_name VARCHAR(50),
        lt_id VARCHAR(50),
        status VARCHAR(20),
        PRIMARY KEY (p_id, t_name, lt_id),
        FOREIGN KEY p_id REFERENCES appointment(appt_id),
```

```
        FOREIGN KEY t_name REFERENCES test(name),
        FOREIGN KEY lt_id REFERENCES lab_technician(ld_id)
);
```

21. performed_comps

Model:
performed_comps(t_name, c_name, lt_id, appt_id);

Candidate Keys:
{(t_name, c_name, lt_id, appt_id)}

Primary Key:
{t_name, c_name, lt_id, appt_id}

Foreign Keys:
t_name references test.name
c_name references component.name
lt_id references lab_technician.id
appt_id references appointment.appt_id

Table Declaration:

```
CREATE TABLE performed_comps(
        t_name VARCHAR(100),
        c_name VARCHAR(100),
        lt_id VARCHAR(50),
        appt_id VARCHAR(50),
        PRIMARY KEY(t_name, c_name, lt_id, appt_id),
        FOREIGN KEY t_name REFERENCES test(name),
        FOREIGN KEY c_name REFERENCES component(name),
        FOREIGN KEY lt_id REFERENCES lab_technician(id),
        FOREIGN KEY appt_id REFERENCES appointment(appt_id)
);
```

22. department

Model:
department(name);

Candidate Keys:
{(name)}

Primary Key:
name

Foreign Keys:
None.

Table Declaration:
```
CREATE TABLE department(
        name VARCHAR(50),
        PRIMARY KEY(name)
);
```

23. doc_dept

Model:
doc_dept(name, doc_id);

Candidate Keys:
{(name)}

Primary Key:
name

Foreign Keys:
doc_id references person.person_id
name references department.name

Table Declaration:
```
CREATE TABLE doc_dept(
        Name VARCHAR(50),
        Doc_id VARCHAR(50) NOT NULL,
        PRIMARY KEY(name),
        FOREIGN KEY (name) REFERENCES department(name),
        FOREIGN KEY (doc_id) REFERENCES person(person_id)
);
```

24. symptoms

Model:
symptoms(symp_desc, appt_id);

Candidate Keys:
{(symp_desc, appt_id)}

Primary Key:
{symp_desc, appt_id}

Foreign Keys:
appt_id references appointment.appt_id

Table Declaration:
```
CREATE TABLE symptoms(
        symp_desc VARCHAR(100),
        appt_id VARCHAR(50),
        PRIMARY KEY(symp_desc, appt_id),
        FOREIGN KEY appt_id REFERENCES appointment(appt_id)
);
```

25. diagnosed_disease

Model:
diagnosed_disease(dis_name, appt_id, severity)

Candidate Keys:
{ (dis_name, appt_id)}

Primary Key:
{dis_name, appt_id}

Foreign Keys:
appt_id references appointment.appt_id
dis_name references disease.dis_name

Table Declaration:
```
CREATE TABLE diagnosed_disease(
        dis_name VARCHAR(50),
        appt_id INT NOT NULL,
        severity INT,
        PRIMARY KEY(dis_name, appt_id),
        FOREIGN KEY (dis_name) REFERENCES disease(dis_name),
        FOREIGN KEY (appt_id) REFERENCES appointment(appt_id)
);
```

26. disease

Model:
disease(name)

Candidate Keys:

{(name)}

Primary Key:
{name}

Foreign Keys:
None

Table Declaration:
```
CREATE TABLE disease(
        name VARCHAR(50) NOT NULL,
        PRIMARY KEY(name),
);
```

27. prescription

Model:
prescription(time, description, appt_id)

Candidate Keys:
{(appt_id)}

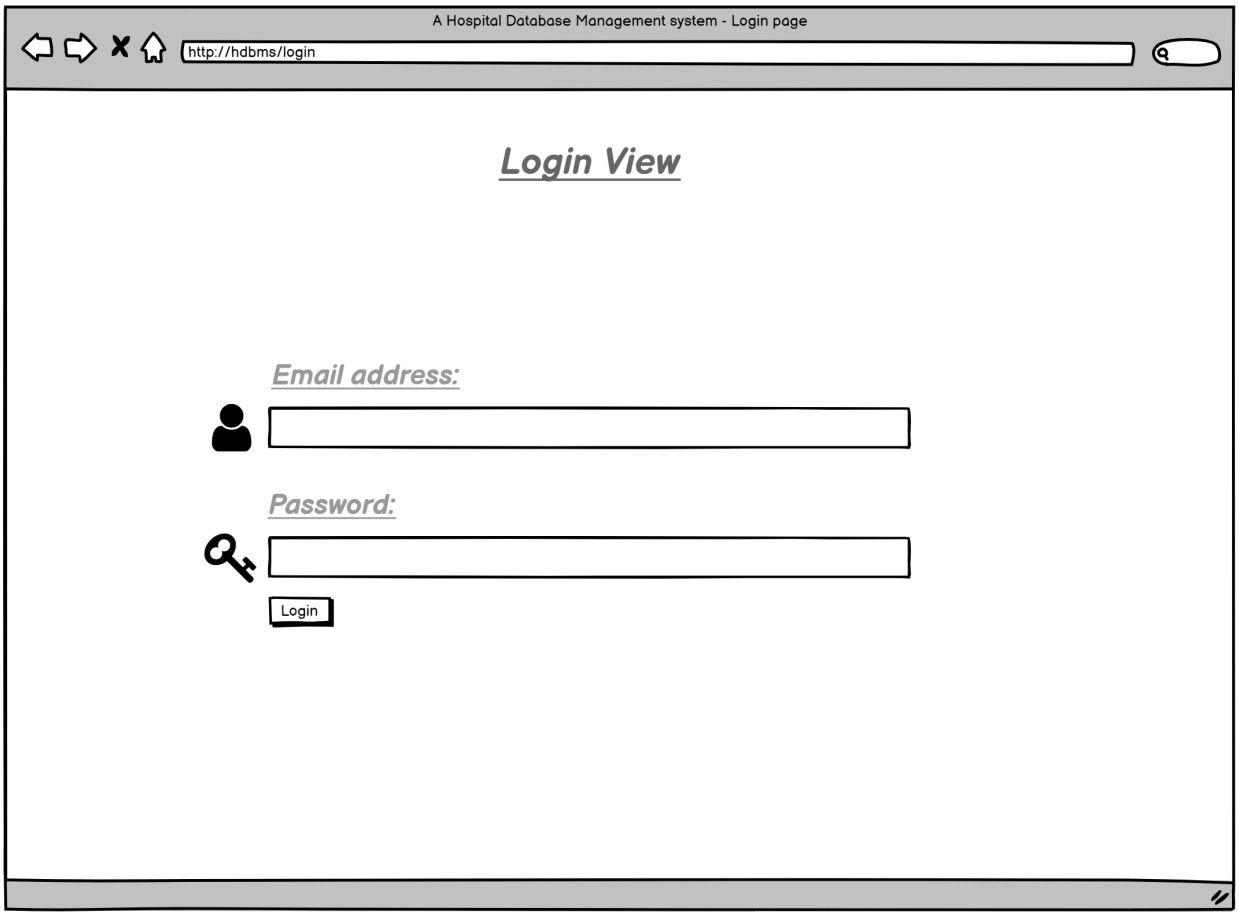
Primary Key:
{appt_id}

Foreign Keys:
appt_id references appointment.appt_id

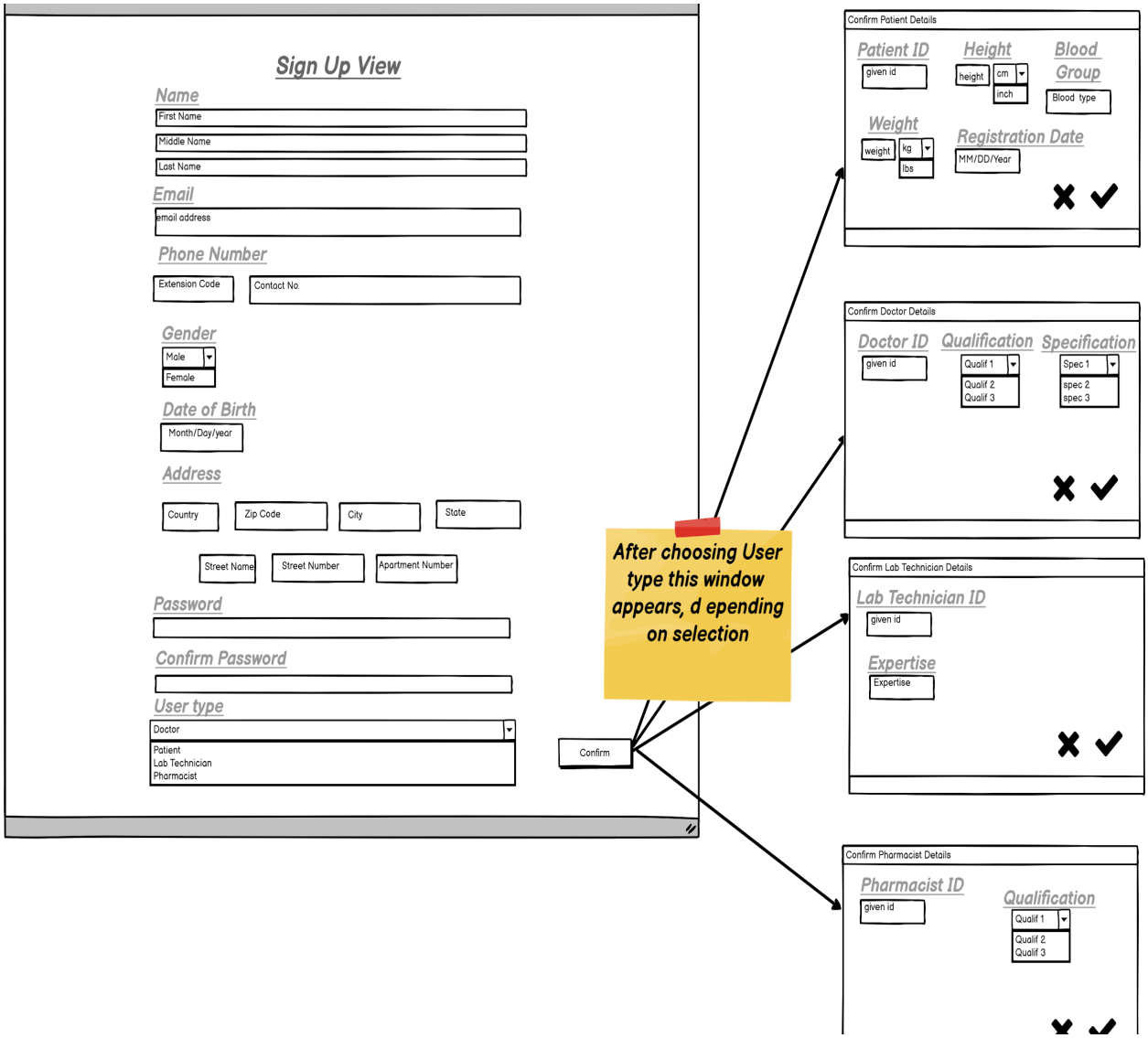
Table Declaration:
```
CREATE TABLE prescription(
        time DATE
        description VARCHAR(100) NOT NULL,
        appt_id VARCHAR(50) NOT NULL,
        PRIMARY KEY(apt_id),
        FOREIGN KEY (appt_id) REFERENCES appointment(appt_id)
);
```

User Interfaces

1. **Login/Sign UP**
    1.1. Login

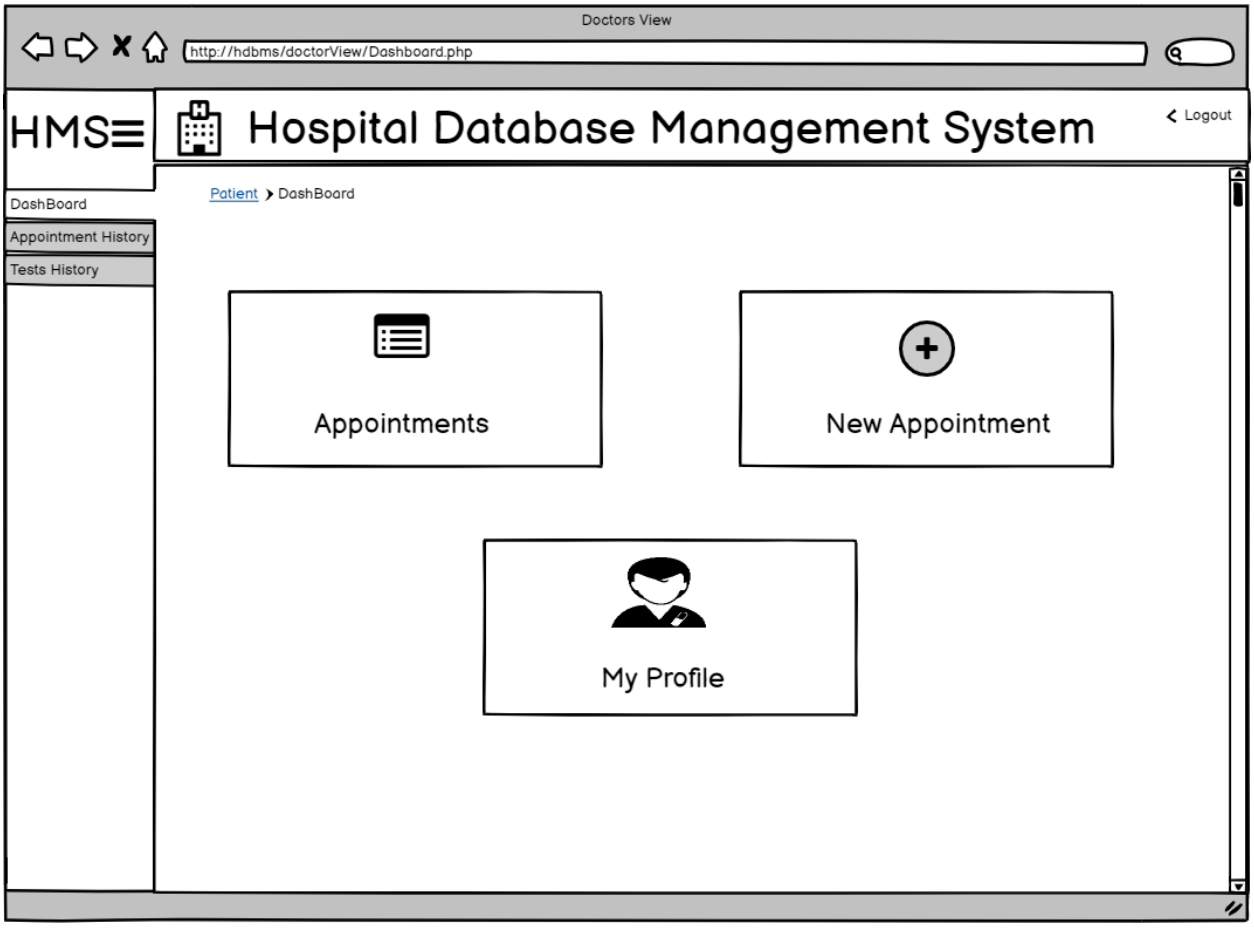


1.2. Sign Up

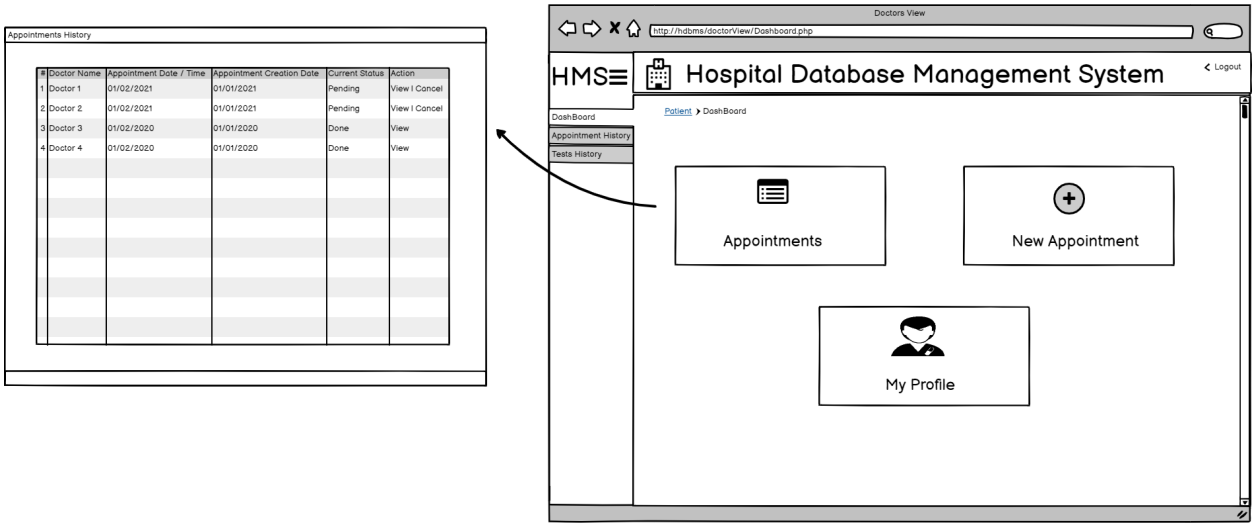

Sign Up View
Name
First Name
Middle Name
Last Name
Email
email address
Phone Number
Extension Code
Contact No.
Gender
Male
Female
Date of Birth
Month/Day/year
Address
Country
Zip Code
City
State
Street Name
Street Number
Apartment Number
Password
Confirm Password
User type
Doctor
Patient
Lab Technician
Pharmacist
Confirm
After choosing User type this window appears, d epending on selection
Confirm Patient Details
Patient ID
given id
Height
height
cm
inch
Blood Group
Blood type
Weight
weight
kg
lbs
Registration Date
MM/DD/Year
Confirm Doctor Details
Doctor ID
given id
Qualification
Qualif 1
Qualif 2
Qualif 3
Specification
Spec 1
spec 2
spec 3
Confirm Lab Technician Details
Lab Technician ID
given id
Expertise
Expertise
Confirm Pharmacist Details
Pharmacist ID
given id
Qualification
Qualif 1
Qualif 2
Qualif 3

2. Patients View

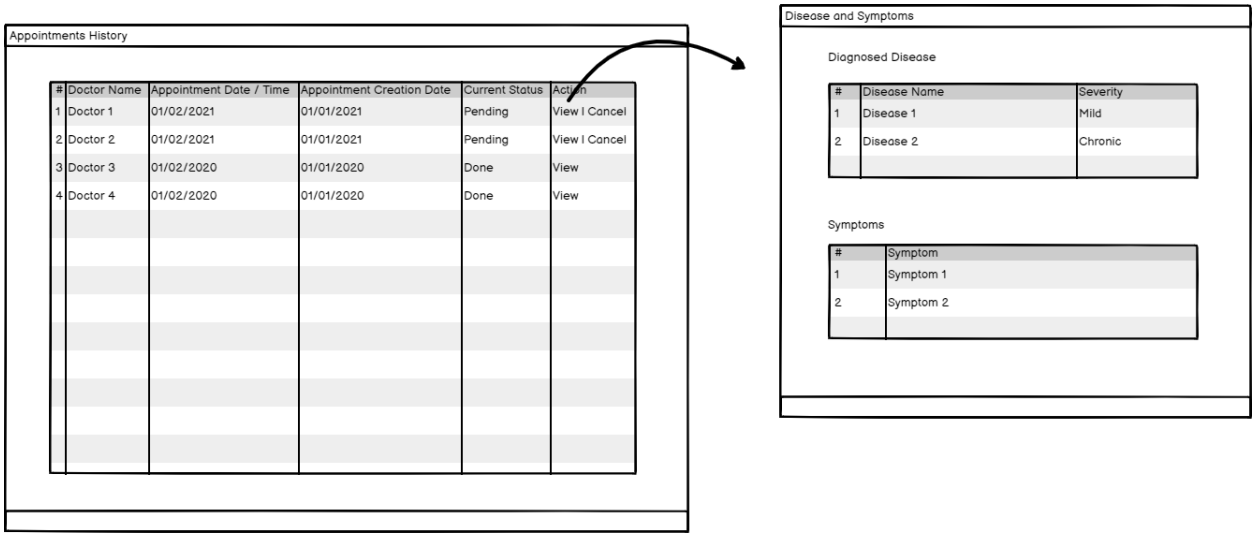
2.1. Dashboard



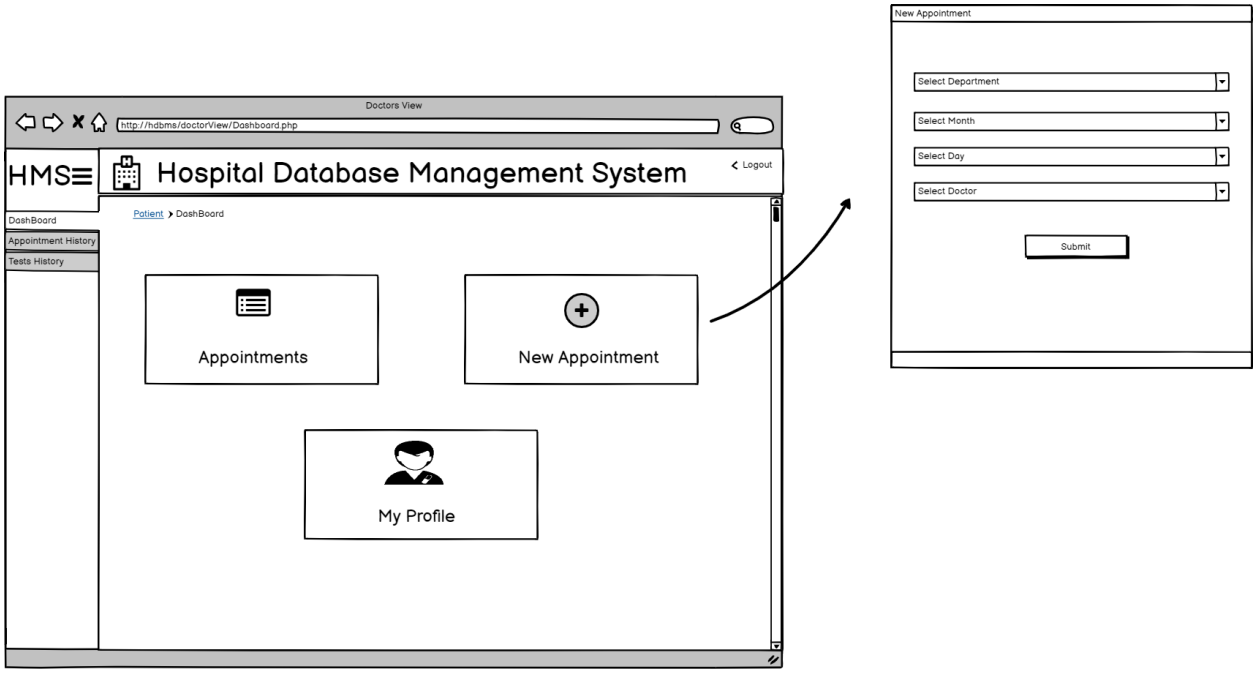
2.2. Appointments



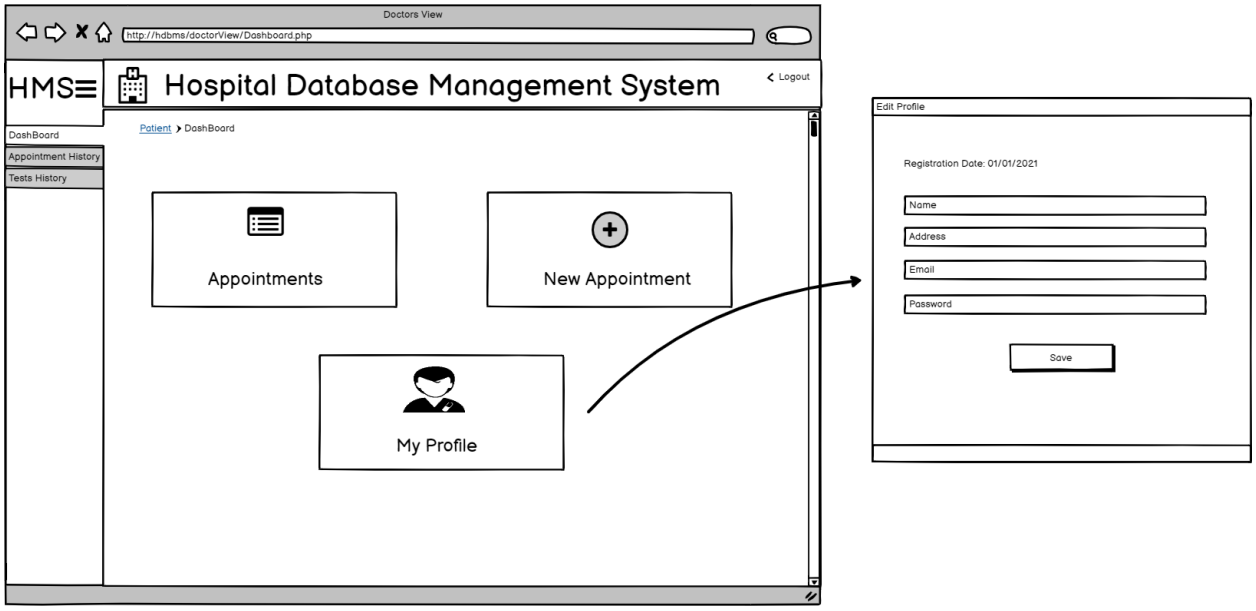
2.3. Diagnosis and Symptoms



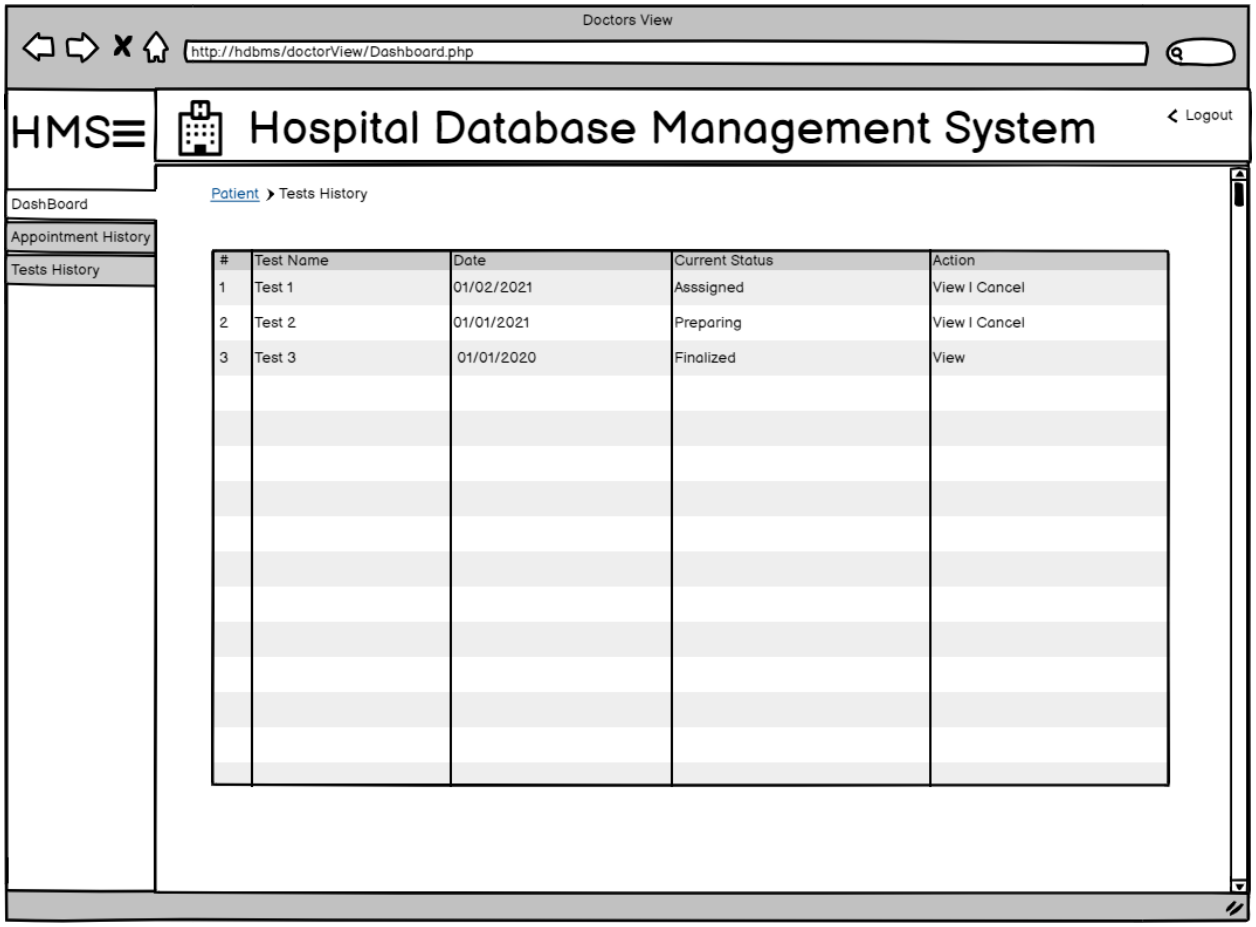
2.4. New Appointment



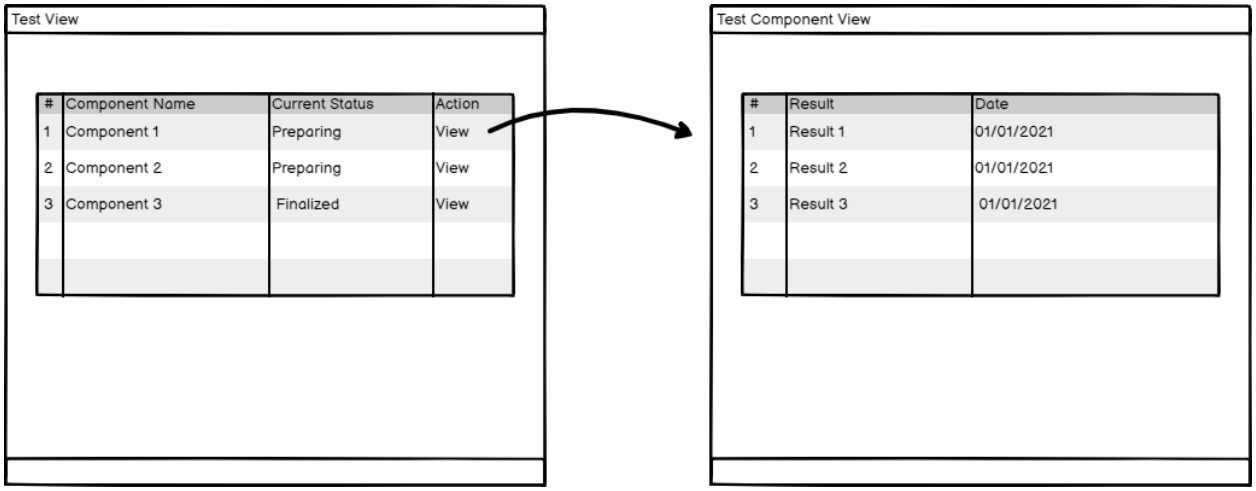
2.5. Profile



2.6. Test Results



2.7. Test Component Results



SQL Queries

Login

```
SELECT *
FROM person
WHERE email=@email
        AND password=@password;
```

Signup

```
INSERT INTO person
VALUES (@person_id, @first_name, @middle_name,  @last_name, @dob, @apt_num,
        @street_name, @street_num, @city, @state, @zip, @country, @country_code,
        @number, @gender, @email, @password
);

INSERT INTO doctor
VALUES (@person_id, @specialization, @qualification);

INSERT INTO patient
VALUES (@person_id, @height, @weight, @blood_group, @registration_date);

INSERT INTO lab_technician
VALUES (@person_id, @expertise);

INSERT INTO pharmacist
VALUES (@person_id, @qualifications);
```

Get free doctors for a date in a department

```
SELECT *
FROM doctor
WHERE doctor.dept_name=@dept_name
        AND d_id NOT IN (
        SELECT d_id
        FROM doc_visit
        WHERE date=@date AND status='SCHEDULED'
);
```

Get disease diagnosed for an appointment

```
SELECT dis_name
FROM diagnosed_disease
WHERE appt_id=@appt_id;
```

Get symptoms shared for a particular disease

```
SELECT symp_desc
FROM symptoms
WHERE appt_id=@appt_id;
```

Make an appointment

```
INSERT INTO appointment
VALUES(@appt_id, @description);

INSERT INTO doc_visit
VALUES(@d_id, @p_id, @appt_id, @date, 'SCHEDULED');
```

Cancel an appointment

```
UPDATE appointment
SET status='CANCELLED'
WHERE appt_id=@apptt_id;
```

Show Patient Appointment History

```
SELECT *
FROM doc_visit NATURAL JOIN appointment
WHERE p_id=@p_id
ORDER BY date DESC;
```

Show All Tests History Sorted by Date

```
SELECT *
FROM perform_test
WHERE appt_id IN (
        SELECT appt_id
        FROM doc_visit
        WHERE p_id=@p_id
);
```

Show components for a test

```
SELECT *
FROM performed_comps
WHERE t_name=@t_name
        AND appt_id=@appt_id;
```

Show previous history for a component

```
SELECT *
FROM performed_comps
WHERE c_name=@c_name AND performed_comps.appt_id IN (
        SELECT perform_test.appt_id AS appt_id
        FROM perform_test
```

```
        WHERE perform_test.appt_id IN (
                SELECT appt_id
                FROM doc_visit
                WHERE p_id=@p_id
                        AND status='DONE'
        )
);
```

Additional Functional Requirements

The additional functional requirements added are the prescription and pharmacy entities. The prescription is given out to a patient and they will be recorded in the database in order to get the patient history. The pharmacy entity is added in order to keep track of the medicines available in the hospital. This entity represents the pharmacies located in the hospital. Moreover, the inventory of these pharmacies is also recorded in a relation. This inventory will be used to keep track of all the requirements that are available in a particular pharmacy.

Website

Project Document Webpage
https://cs353-group-27.github.io/