Bilkent University
Department of Computer Engineering

CS353 Term Project

Final Report

Hospital Database Management System
Section - 3
Group - 29

Project Group Members

1.	Oğuzhan Angın	21501910
2.	Mehmet Alperen Yalçın	21502273
3.	Ahmet Furkan Ahi	21501903

Supervisor: Arif Usta

Table of Contents

1. DESCRIPTION OF THE PROGRAM

The hospital database system is a web-based application that can be used synchronously by patients, doctors, laboratories and pharmacists in the hospital. The system contains much information belonging to users or used/processed by users, such as diseases, symptoms, tests and medicines, and presents them to the application user by filtering them with appropriate filters. It is explained below what a doctor, patient and laboratory can do, respectively.

First of all, there are many doctors from many departments in the hospital, and each doctor has its schedule. According to this schedule, they arrange the dates when patients can get an appointment and they check the appointments that have already been made by looking at this schedule. In addition, through the program, they can reactivate the canceled days and learn the details of the patients who made an appointment from this page. When looking at the patient details, they reach the patient's symptoms, diagnoses, information about the patient's disease and the tests requested from the patient. In addition, they can add/remove existing information and request extra tests from here.

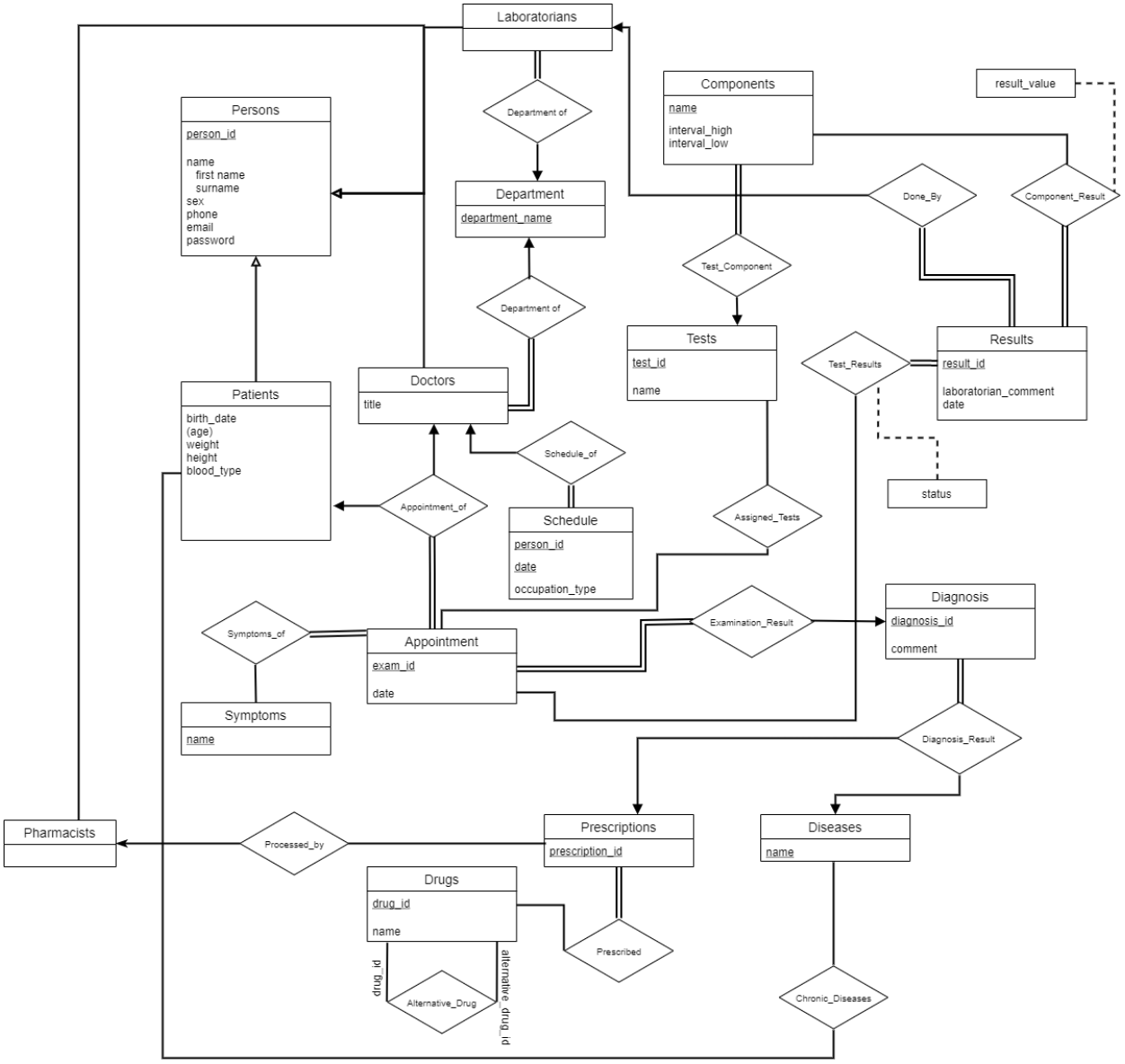
When we come to the patient part of the program, the first thing the patient will do should be to make an appointment. With the help of filters, the patient can make an appointment from the department they want and (if applicable) at any time. The patient can check all the appointments he has made through the website. In addition, they cannot make an appointment for the same day from the same department. If the patient wishes, he/she can cancel some of the symptoms he/she shows by clicking the details on the appointments page.

Laborants, on the other hand, can see all tests to be done/performed from a screen where all tests are collected in a pool. The tests to be performed are randomly assigned to any laboratory. Laborants can comment on the tests and check the result status of the test.

Pharmacists are staff members responsible for supplying patients with medicines prescribed by doctors. They can also offer patients an alternative to a drug.

In summary, the Hospital database system is a control program where many potential events that may occur in a hospital are created, processed and stored. It keeps up-to-date and informs patients, doctors and other staff such as laboratory professionals and pharmacists.

2. FINAL ER DIAGRAM



3. RELATIONAL SCHEMAS

3.1 Persons

Relational Model
Persons(person_id, first_name, last_name, sex, phone, email, password)

3.2 Patients

Relational Model
patients(birth_date, weight, height, blood_type)

FOREIGN KEY (person_id) references persons (person_id)

3.3 Doctors

Relational Model
doctors(title)

FOREIGN KEY (person_id) references persons (person_id)

3.4 Laboratorians

Relational Model
laboratorians(person_id)

FOREIGN KEY (person_id) references persons (person_id)

3.5 Pharmacist

Relational Model
pharmacist(person_id)

FOREIGN KEY (person_id) references persons (person_id)

3.6 Appointment

Relational Model
appointment(exam_id, date)

3.7 Symptoms

Relational Model
symptoms(name)

3.8 Schedule

Relational Model
schedule(person_id, date, occupation_type)

FOREIGN KEY (person_id) references persons (person_id)

3.9 Department

Relational Model
department(department_name)

3.10 Drugs

Relational Model
drugs(drug_id, name)

3.11 Prescriptions

Relational Model
prescriptions(prescription_id)

3.12 Components

Relational Model
components(name, interval_high, interval_low)

3.13 Tests

Relational Model
tests(test_id, name)

3.14 Results

Relational Model
results(result_id, laboratorian_comment, date)

3.15 Diagnosis

Relational Model
diagnosis(diagnosis_id, comment)

3.16 Diseases

Relational Model
diseases(disease_id, name)

3.17 symptoms_of

Relational Model
symptoms_of(exam_id, name)

FOREIGN KEY (exam_id) references appointment (exam_id)
FOREIGN KEY (name) references symptoms (name)

3.18 appointment_of

Relational Model
appointment_of(exam_id, patient_id, doctor_id)

FOREIGN KEY (doctor_id) references doctors(person_id)
FOREIGN KEY (patient_id) references patients(person_id)
FOREIGN KEY (exam_id) references appointment(exam_id)

3.19 processed_by

Relational Model
procesesed_by(prescription_id, person_id))

FOREIGN KEY (person_id) references pharmacists(person_id)
FOREIGN KEY (prescription_id) references prescriptions(prescription_id)

3.20 department_of

Relational Model
department_of(person_id, department_name)

FOREIGN KEY (person_id) references doctors(person_id)
FOREIGN KEY (department_name) references department(department_name)

3.21 alternative_drug

Relational Model
alternative_drug(drug_id, alternative_drug_id)

FOREIGN KEY (alternative_drug_id) references drugs(drug_id)
FOREIGN KEY (drug_id) references drugs(drug_id)

3.22 prescribed

Relational Model
prescribed(prescription_id, drug_id)

FOREIGN KEY (drug_id) references drugs(drug_id)
FOREIGN KEY (prescription_id) references prescriptions(prescription_id)

3.23 test_component

Relational Model
test_component(test_id, name)

FOREIGN KEY (name) references components(name)
FOREIGN KEY (test_id) references tests(test_id)

3.24 assigned_tests

Relational Model
test_component(test_id, exam_id)

FOREIGN KEY (exam_id) references appointment(exam_id)
FOREIGN KEY (test_id) references tests(test_id)

3.25 examination_result

Relational Model
examination_result(diagnosis_id, exam_id)

FOREIGN KEY (exam_id) references appointment(exam_id)
FOREIGN KEY (diagnosis_id) references diagnosis(diagnosis_id)

3.26 done_by

Relational Model
done_by(result_id, person_id)

FOREIGN KEY (person_id) references laboratorians(person_id)
FOREIGN KEY (result_id) references results(result_id)

3.27 component_result

Relational Model
component_result(result_id, name, result_value)

FOREIGN KEY (name) references components(name)
FOREIGN KEY (result_id) references results(result_id)

3.28 test_result

Relational Model
test_result(result_id, exam_id, status)

FOREIGN KEY (exam_id) references appointment(exam_id)
FOREIGN KEY (result_id) references results(result_id)

3.29 diagnosis_result

Relational Model
diagnosis_result(diagnosis_id, name, prescription_id)

FOREIGN KEY (prescription_id) references prescriptions(prescription_id)
FOREIGN KEY (diagnosis_id) references diagnosis(diagnosis_id)
FOREIGN KEY (name) references diseases(name)

3.30 chronic_diseases

Relational Model
chronic_diseases(name, person_id)

FOREIGN KEY (name) references diseases(name)
FOREIGN KEY (person_id) references patients(person_id)

4. IMPLEMENTATION DETAILS

Our hospital database system consists of 2 parts: the database part and the website part. MySQL is preferred in the database part. Java project has been used in database creation and organization and in processing SQL queries. While creating the database, care was taken to enter data parallel to real life. Although it is possible to get data from another source, it is deemed appropriate to enter data manually in this version of the project.
The technologies we use in the creation and processing of the user interface and website functions are PHP, HTML, CSS and Javascript.

- HTML and CSS have been used to create and design the front-end of the user interface; CSS has been used to add some styles to the page and for similar purposes to HTML.
- In addition, in some parts, Bootstrap was used while creating web pages.
- PHP is used to perform system operations. Thus, integration between database and website has been established. In addition, PHP has been used intimately with HTML, enabling us to fulfill the page actions.
- Javascript, on the other hand, is a language that we use in matters such as the input management of some parts of the pages. JQuery, which is a part of Javascript, has been used for similar purposes.

5. USER'S MANUAL

5.1 Register Page

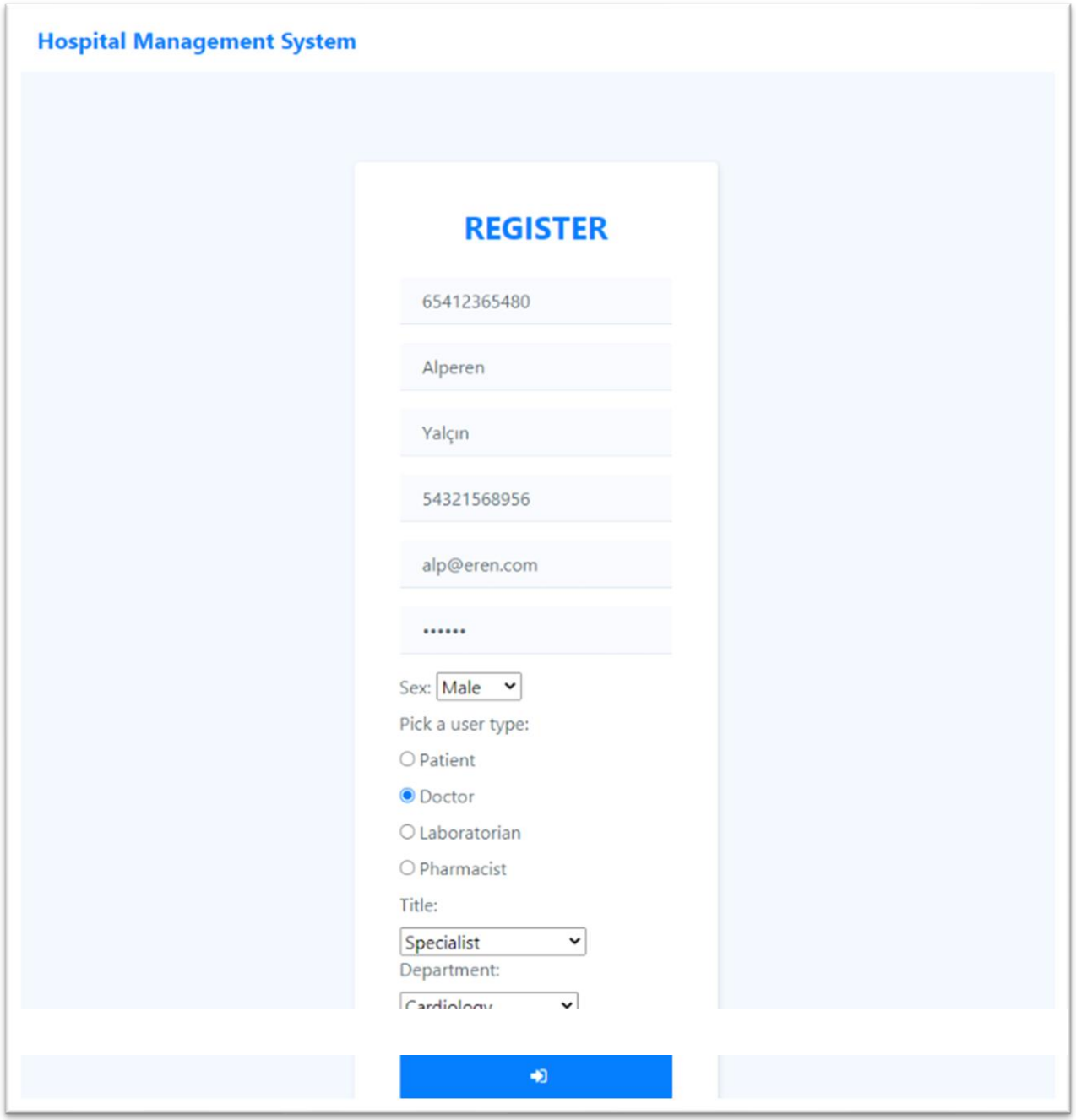


Figure 1: Sign up screen for the unregistered users

On this page, the user enters the information required to register into the system. The information entered according to the user type varies in addition to the fixed information. For example, if a patient is registered, some basic medical information of the patient is recorded in the system (height, weight, blood type, date of birth etc.). If a doctor is registered, registration is made according to the doctor's field of expertise. In addition, the specified fields must be filled; otherwise, the system will issue a warning. When the necessary information is filled in, the system automatically directs you to the login page.

5.2 Login Page

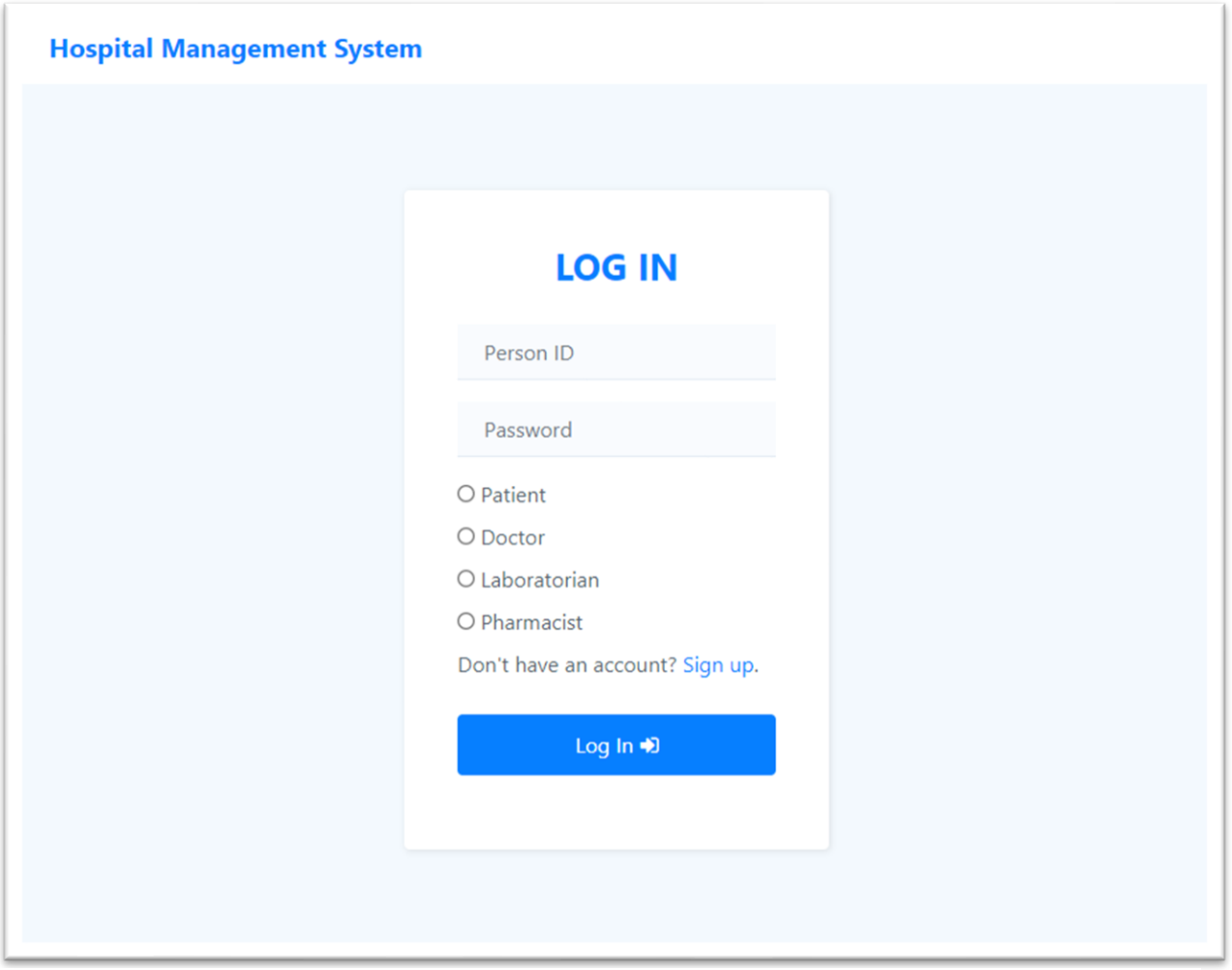


Figure 2:: Home Screen for login the system

This page is a simple login page. Users can log into the system when they enter their id and password and select the user type. It is mandatory to choose a user type. If there any unmatched input, the system gives an error again.

5.3 Doctors' Page

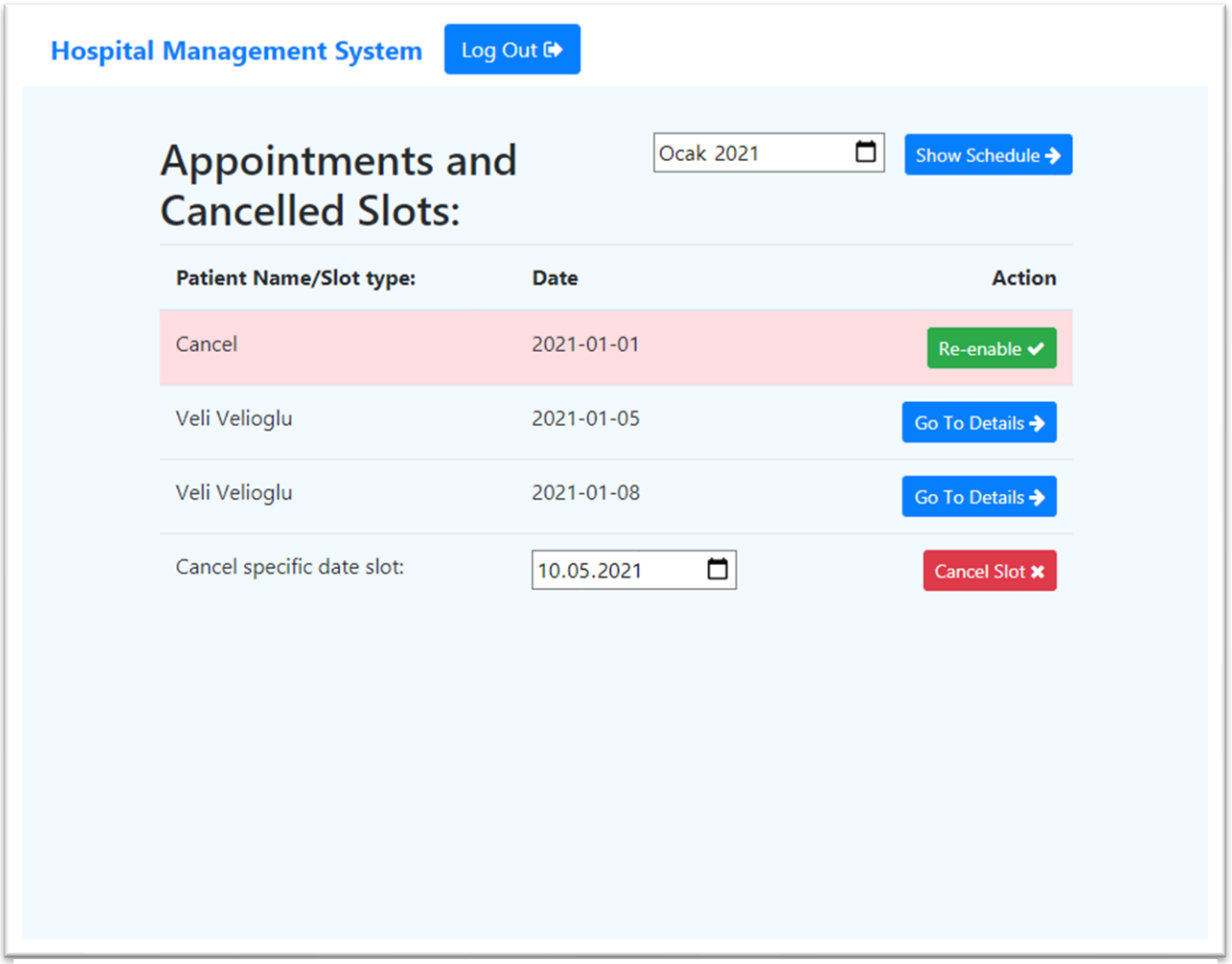


Figure 3: A doctor's schedule

This page shows the chart of the month chosen by a doctor who entered the system. A doctor can check his schedule by choosing the month he wishes and cancel any day as he wishes, or make the day he canceled available for an appointment again. If a patient can make an appointment for that day before the cancellation, the doctor cannot cancel that day and (unless the patient cancels) he will have to work that day.

In addition, the doctor can take a look at the details of the patient, the appointment owner, on this page. They can look at the patient's symptoms, diagnose/cancel them, review their tests, and request new tests from the laboratory.

Lastly, the back and logout buttons on the page allow direct access to the related pages.

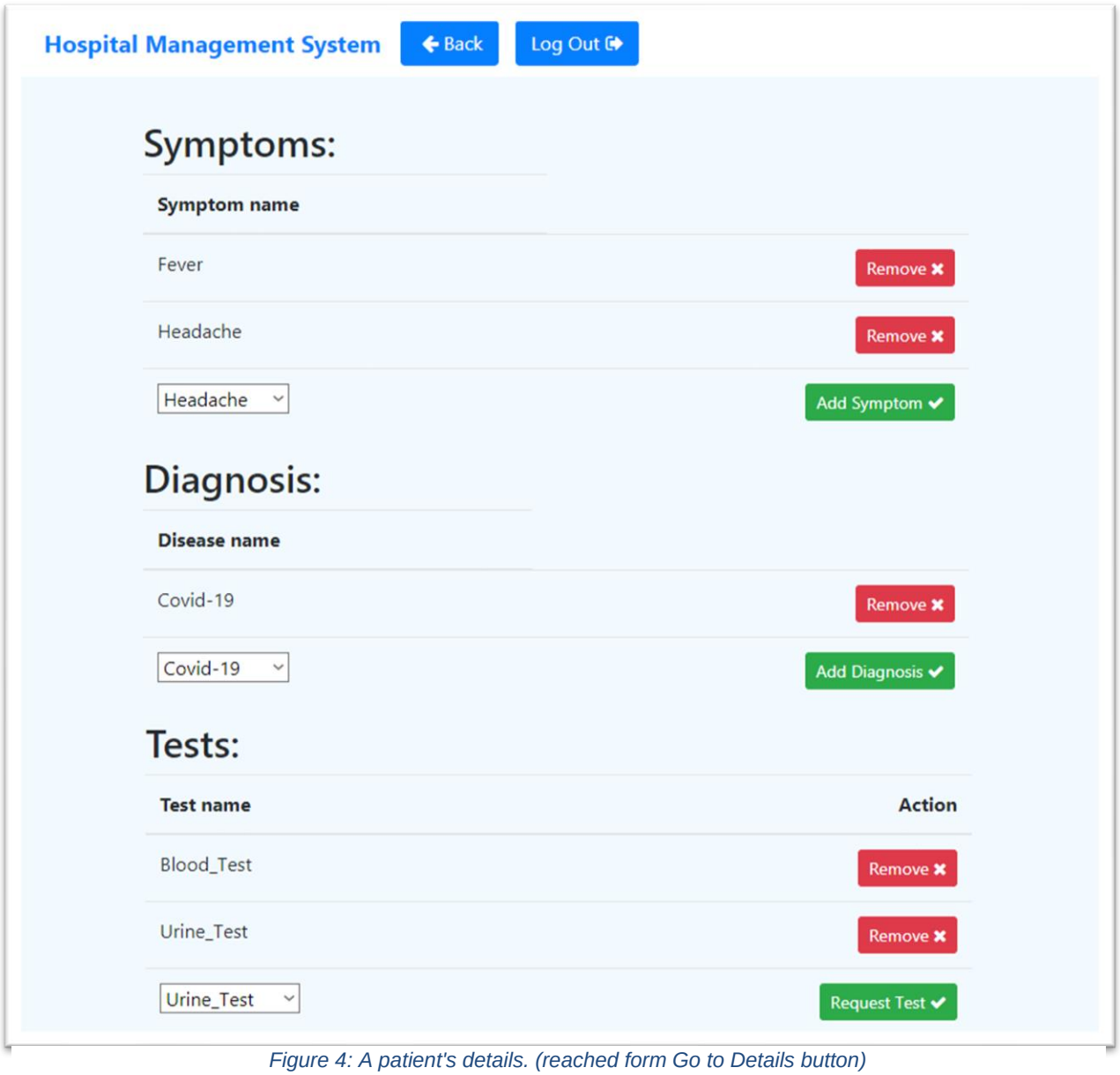


Figure 4: A patient's details. (reached form Go to Details button)

The detail page and related buttons are described above in Figure 4.

5.4 Patients' Page

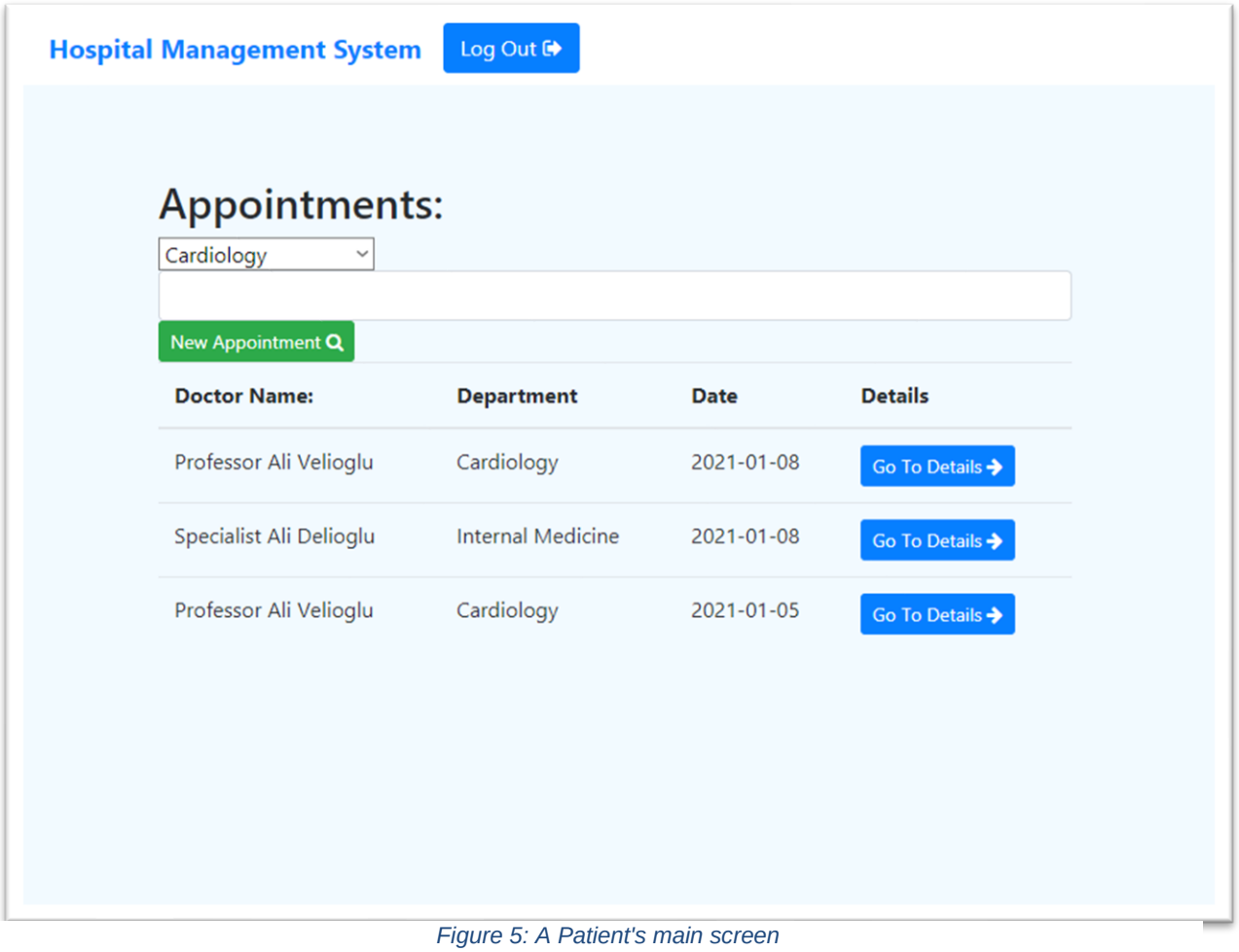


Figure 5: A Patient's main screen

The screen the patient encounters when he enters the system is as in Figure 5. Currently, his appointments are listed by showing doctor names, departments and dates. The patient can create new appointments by choosing a department and date according to his request. However, they are not allowed to create another appointment for the same day from the same department.

From this screen, the patient can go to the detail screen with his/her information. Similar to what the doctor sees, it can examine its symptoms, diagnoses and tests, and add/remove his/her symptoms accordingly at will.

5.5 Laboratorians' Page

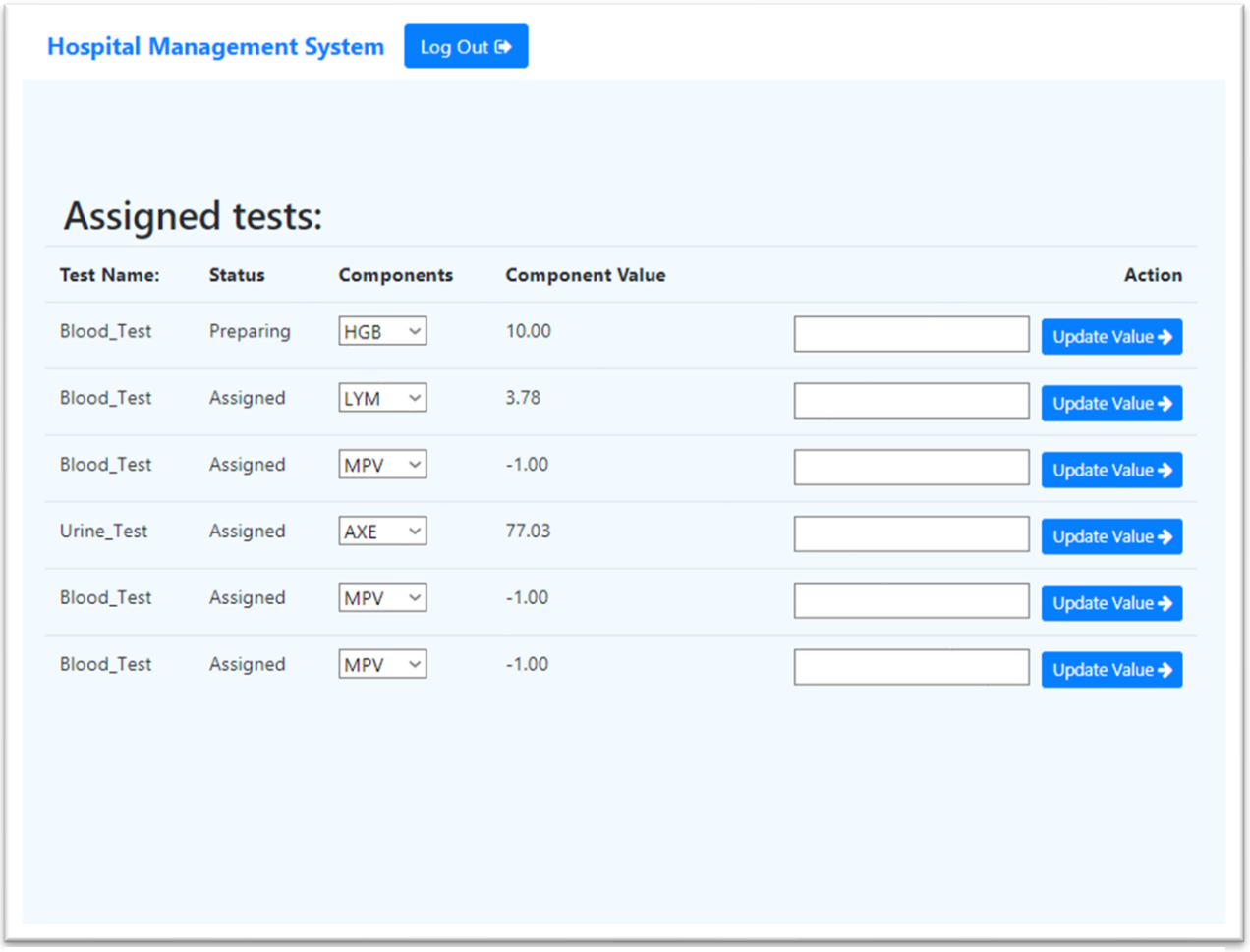


Figure 6: A Laboratorian's page

On this screen, the laboratory can view all the tests that have been created together with their status and update their values. If no value has been entered yet, it is displayed as -1.00.

6. ADVANCED DATABASE COMPONENTS

6.1 Views

Reaching the patient age by looking at the birth_date:

CREATE VIEW patient_age as SELECT person_id, TIMESTAMPDIFF (YEAR, birth_date, CURDATE()) as age FROM patients

Showing the history of the appointments of a patient:

CREATE VIEW appointment_history as SELECT D.title, P.first_name, P.last_name, DE.department_name, A.date, A.exam_id FROM persons P, appointment A, doctors D, department_of DE WHERE A.exam_id IN (SELECT exam_id FROM appointment_of WHERE patient_id = '20000000000' AND doctor_id = P.person_id AND doctor_id = D.person_id AND doctor_id = DE.person_id) AND A.date < CURDATE() ORDER BY A.date DESC

6.2 Reports

Working days in a month of a doctor:

SELECT person_id, COUNT(date) as NumberOfWorkedDay from schedule WHERE person_id = '10000000000' and MONTH(date) = '1' and occupation_type = 'Appointment'

6.3 Triggers

When the doctor requests a test, a record is created in assigned_test. Then this test needs to be created in some other tables, too, such as result, test_result, component_result, done_by. We used a trigger for this purpose.

Similarly, we use a similar approach when we are removing a test and its component.

7. WEBSITE (Github)

- https://github.com/alperenya/CS-353-Project-Group-29