USE CASES AND QUERIES.

USE CASE 1: Log in.

Actor: sorting medic

Goal/Name: log in

Description: sorting medic type its information, as user and password

Precondition: the sorting medic must have an account

Standard Scenario:

1. The sorting medic type its user and its password

Postcondition: the sorting medic has access to the app

SELECT username, password FROM sortingMedic WHERE id = '?'

USE CASE 2: Select a patient.

Actor: sorting medic

Goal/Name: select a patient

Description: search for the data of the patient

Precondition: the sorting medic must have logged in and the patient must be in the database

Standard Scenario:

1. Input the name of the patient into the search bar
2. The app shows a list of patients and asks for the id
3. Input the id of the patient we want to select

Postcondition: two options show up, "show data" "update data"

SELECT p.id, p.name, p.surname, p.dob, h.id, h.name, h.location, i.id, i.condition, hi.severity

FROM patient AS p JOIN has_illness AS hi ON hi.patient_ID = p.ID

JOIN illness AS i ON i.ID = hi.illness_ID

JOIN hospital AS h ON h.ID = p.Hospital_ID

WHERE p.name = 'Bob'

USE CASE 3: Select a doctor.

Actor: sorting medic

Goal/Name: select a doctor

Description: search for the data of the doctor

Precondition: the sorting medic must have logged in and the doctor must be in the database

Standard Scenario:

1. Input the name and surname of the doctor into the search bar

2. The app shows a list of doctors and asks for the id

3. Input the id of the doctor we want to select

Postcondition: three options show up: “show data”, “update data” and “delete doctor”

SELECT d.name, d.surname, d.dob, d.speciality, d.salary, i.condition, h.name, h.location

FROM doctor AS p JOIN doctor_treats AS dt ON dt.doctor_ID = d.ID

JOIN illness AS i ON i.ID = dt.illness_ID

JOIN hospital AS h ON h.ID=p.Hospital_ID

WHERE d.name= 'Homero' AND d.surname= 'Ketchup’

USE CASE 4: Register a patient.

Actor: sorting medic

Goal/Name: register a new patient

Description: input the data of the patient

Precondition: the sorting doctor must have logged in

Standard Scenario:

1. Sorting medic writes patient data

2. App shows a list of illness to choose from

3. Sorting medic selects illness

4. Sorting medic add the severity of this illness

Postcondition: patient data added

INSERT INTO patient (name, surname, dob, photo)

VALUES (‘Bob’, ’Simpson’,’1980-05-21’ , ‘Bob.png’)

SELECT * FROM illness

SELECT * FROM illness WHERE id = (SELECT id FROM illness WHERE name = ‘Acoustic Neuroma’)

SELECT id FROM patient WHERE name = ‘Bob’

SELECT id FROM illness WHERE name = ‘Acoustic Neuroma’

INSERT INTO has_illness (illness_ID, patient_ID, severity)

VALUES (‘?’, ‘?’, ‘98%’)

USE CASE 5: Update patient.

Actor: sorting medic

Goal/Name: update patient

Description: replace the old data of the patient for new one

Precondition: sorting medic must have logged in and patient must have been selected

Standard Scenario:

1. App shows patient selected menu
2. Sorting medic selects update option
3. Sorting medic changes patient data or presses enter if there is a field we don't want changed.
4. App asks if we want to change the illness severity
5. If yes app shows the illness the patient has and asks for the illness we want updated
6. Sorting medic enters new severity, if he wants to

Postcondition: patient data updated

SELECT name FROM patient WHERE id = '?'

UPDATE patient

SET name = '?', surname = '?', photo = '?' WHERE name = '?'

SELECT * FROM illness WHERE id = '?'

SELECT * FROM hasIllness WHERE patient_ID = '?' AND illness_ID = '?'

#if sorting medic wants to change the severity of the illness, the database will do the next steps

UPDATE has_illness

SET severity = '?' WHERE patient_ID = '?' AND illness_ID = '?'

USE CASE 6: Update doctor.

Actor: sorting medic

Goal/Name: update patient

Description: replace the data of the doctor

Precondition: the sorting medic must have logged in and the doctor must have been selected

Standard Scenario:

1. App shows doctor data
2. Sorting medic changes doctor data or presses enter if there is a field we don't want changed.

Postcondition: data of the doctor updated

UPDATE doctor

SET name = '?', surname = '?', salary = '?'

WHERE id = (SELECT id FROM doctors WHERE name = '?' AND surname = '?')

USE CASE 7: Delete doctor.

Actor: sorting medic

Goal/Name: delete a doctor

Description: delete the doctor selected

Precondition: the sorting medic must have logged in and the doctor must have been selected

Standard Scenario:

1. The app ask the user to put the name and the surname of the doctor

2. The user select the option to delete the doctor

3. The doctor that was selected at the beginning is deleted

3. The app shows a message that let us know the doctor was deleted

Postcondition: the doctor has been deleted

DELETE FROM doctors WHERE id = (SELECT id FROM doctor WHERE name = '?' AND surname = '?')

USE CASE 8: Start search for hospital.

Actor: sorting medic

Goal/Name: search a hospital for the patient

Description: once the patient data is on the app, the sorting medic selects the option to search for a hospital according to the patient data (illness) and the availability of the hospital

Precondition: the sorting medic must have logged in and the patient must have been selected

Standard Scenario:

1. App shows list of illness of the patient and asks for the id of the illness we want to treat

2. Sorting medic inputs the id of the illness

3. App checks doctors that treats that illness

4. App checks mahines that treat that illness

5. App matches the machine and doctor to search for a hospital that has both

6. App checks for the availability of the hospital

7. App shows a hospital with the conditions required

Postcondition: the app has assigned a hospital to the patient

SELECT * FROM illness

SELECT id, name, location FROM hospital

WHERE id = (SELECT hospital_ID FROM doctor WHERE id = (SELECT doctor_ID FROM illness WHERE id = ANY (SELECT illness_ID FROM has_illness WHERE patient_ID = (SELECT id FROM patient WHERE name= '?' AND surname = '?'))))

AND id = (SELECT hospital_ID FROM machine WHERE id = (SELECT machine_ID FROM treats WHERE illness_ID = ANY (SELECT illness_ID FROM has_illness WHERE patient_ID= (SELECT id FROM patient WHERE name= '?' AND surname = '?'))))

USE CASE 9: Show Hospitals.

Actor: sorting medic

Goal/Name: show a list of hospitals

Description: show the list of hospitals inside the database

Precondition: the sorting medic must have logged in and the doctor must have been selected

Standard Scenario:

1.The sorting medic selects the option to show hospitals

2.The app gets the list of hospitals registered and prints it

Postcondition: the list of hospitals is shown

SELECT * FROM hospital

USE CASE 10: Create patient XML.

Actor: sorting medic

Goal/Name: shows the xml of the patient

Description: create the xml of the patient and shows it

Precondition: the sorting medic must have logged in

Standard Scenario:

1. The sorting medic selects the option to creat a patient xml
2. App generates a patient xml

Postcondition: the patient xml is created and shown

USE CASE 11: Create hospital XML.

Actor: sorting medic

Goal/Name: show the xml of the hospital

Description: create the xml of the hospital and shows it

Precondition: the sorting medic must have logged in

Standard Scenario:

1. The sorting medic selects the option to creat a patient xml
2. App generates a hospital xml

Postcondition: the hospital xml is created and shown

USE CASE 12: Create doctor XML.

Actor: sorting medic

Goal/Name: show the xml of the doctor

Description: create the xml of the doctor and shows it

Precondition: the sorting medic must have logged in

Standard Scenario:

1. The sorting medic selects the option to create a doctor xml
2. App generates a doctor xml

Postcondition: the doctor xml is created and shown

USE CASE 13: Create machine XML.

Actor: sorting medic

Goal/Name: show the xml of the machine

Description: create the xml of the machine and shows it

Precondition: the sorting medic must have logged in

Standard Scenario:

1. The sorting medic selects the option to create an xml
2. Then, the sorting medic selects the option to create a machine xml
3. App generates a machine xml

Postcondition: the machine xml is created and shown

USE CASE 14: Create illness XML.

Actor: sorting medic

Goal/Name: show the xml of the illness

Description: create the xml of the illness and shows it

Precondition: the sorting medic must have logged in

Standard Scenario:

1. The sorting medic selects the option to create a xml

2. Then, the sorting medic selects the option to create a illness xml
3. App generates a illness xml

Postcondition: the illness xml is created and shown

USE CASE 15: Show patient data.

Actor sorting medic

Goal/Name: show data of patient

Description: show the information of the patient selected

Precondition: sorting medic must have logged in and patient must have been selected

Standard Scenario:

1. App shows patient selected menu

2. Sorting medic choose the option to show data option, and put the name of the patient

3. App gets and prints the information of the patient and shows the name, surname, dob, photo (if the patient has it), hospital and illness that the patient is treated for

Postcondition: patient data shows up on the screen

SELECT p.id, p.name, p.surname, p.dob, h.id, h.name, h.location, i.id, i.condition, hi.severity

FROM patient AS p JOIN has_illness AS hi ON hi.patient_ID = p.ID

JOIN illness AS i ON i.ID = hi.illness_ID

JOIN hospital AS h ON h.ID = p.Hospital_ID

WHERE p.name = 'Bob'

USE CASE 16: Show doctor data.

Actor: sorting medic

Goal/Name: show data of doctor

Description: show the information of the doctor selected

Precondition: sorting medic must have logged in and a doctor must have been selected

Standard Scenario:

1. App shows doctor selected menu

2. Sorting medic selects show data option

3. App gets and prints the information of the doctor. Also, the app prints the information of the hospital where the doctor works at, and the illness that the doctor treats.

Postcondition: doctor data shows up on the screen

SELECT d.name, d.surname, d.dob, d.speciality, d.salary, i.condition, h.name, h.location

FROM doctor AS p JOIN doctor_treats AS dt ON dt.doctor_ID = d.ID

JOIN illness AS i ON i.ID = dt.illness_ID

JOIN hospital AS h ON h.ID = p.Hospital_ID

WHERE d.name= 'Homero' AND d.surname = 'Ketchup'

USE CASE 17: Look for illness.

Actor: sorting medic

Goal/Name: look for an illness

Description: assigns an illness to the patient

Precondition: sorting medic must have logged in and patient must have been selected

Standard Scenario:

1. App shows patient selected menu
2. Sorting medic selects assign illness option
3. App shows a list of illnesses to choose from
4. Sorting medic inputs the id and the severity of the illness the patient has

Postcondition: illness added to the patient

INSERT INTO has_illness (illness_ID, patient_ID, severity)

VALUES ('?', '?', '?')

USE CASE 18: Update illness severity.

Actor: sorting medic

Goal/Name: update the severity of an illness

Description: update the severity of one of the illnesses the patient has

Precondition: sorting medic must have logged in and patient must have been selected

Standard Scenario:

1. App shows patient selected menu
2. Sorting medic selects update state of an illness
3. App shows a list of the illness the patient has
4. Sorting medic inputs the id of the illness and then enters the new severity

Postcondition: illness updated

UPDATE has_illness

SET severity = '?' WHERE patient_ID = '?' AND illness_ID = '?'