Software Ontwerp
Project Assignment - Iteration I

For the course *Software Ontwerp*, you will develop a *Hospital System*, which supports the examination and treatment of patients.

In Section 1, we explain how the project is organized, discuss the quality requirements for the software you will develop and the report you will write. In Section 2, we explain the problem domain of the application, followed by the use cases in Section 3.

1 General Information

In this section, we explain how the project is organized, what is expected of the software you will develop and the report you will write.

1.1 Team Work

For this project, you will work in groups of four. Each group is assigned an advisor from the educational staff. If you have any questions regarding the project, you can contact your advisor and schedule a meeting. **When you come to the meeting, you are expected to prepare specific questions and have sufficient design documentation available**. It is your own responsibility to organize meetings with your advisor and we advise to do this regularly. Experience from previous years shows that groups that regularly meet with their advisors produce a higher quality design.

If there are problems within the group, you should immediately notify your advisor. Do not wait until right before the deadline or the exam!

1.2 The Software

We expect you to use the development process and the techniques that are taught in this course.

When designing and implementing your system, you should use a defensive programming style. This means that the *user* of the public interface of a class cannot bring the objects of that class, or objects of connected classes, in an inconsistent state.

You are also required to provide extensive class and method documentation, as taught in previous courses.

You are expected to have a public API towards your user interface, which exposes the functionality of the system. For this API, you should generate a separate documentation page (only including the public operations). Naturally, this public API is ideal for creating system-wide test cases.

You do not have to implement a distributed system. You also do not have to implement networking, use a database, or allow multiple simultaneous logins.

1.3 User Interface

You are strongly recommended to create a text-based user interface, which is considerably simpler to implement than a graphical user interface. The type and looks of the user interface do not determine the grades of your project, only the design and implementation do. Additionally, the amount of time allocated for each iteration does not include the workload for a graphical user interface.

1.4 The Report

You will write a report that explains the reasoning behind the design of your system. This means that you will document **the decisions that you have made** which have led to the design of the system. The size of the report is **limited to 25 pages**. You have to include at least the following items (but the structure of your document does not have to be a copy of this list):

- a table of contents
- an introduction that explains the structure of the report
- an overview of the system and its subsystems, along with how you assigned different responsibilities
- a discussion of explicit applications of the GRASP patterns and how they have influenced the design. You can for instance include alternative designs and their trade-offs that have been discussed within the group.
- a discussion of potential extension scenarios, especially for inheritance hierarchies (e.g. how does the design support an additional subclass?)
- an explanation of your testing approach.
- information about how you managed the project. Include at least an overview of the tasks performed by each group member, and an estimate of the time invested in those tasks. You should also provide an estimate for the total time that each group member invested in the project.
- a conclusion in which you describe the project iteration: problems, interesting experiences,...

The report should at least explain how the main set of system operations is designed within your system. Sufficiently use **correct UML diagrams** to illustrate and support design decisions and discussion. Be careful to avoid a step-by-step narration of UML diagrams in your text!

1.5 What You Should Hand In

You hand in 1 physical printout of your report, clearly mentioning the course name, Professor Tom Holvoet's name, the name of your advisor and your group number.

Additionally, you hand in an electronic ZIP-archive via Toledo. **The archive contains the items below and follows the structure defined below**:

- `groupXX` (where XX is your group number (e.g. 01, 12, ...))
  - `design`: a folder containing **all** your design diagrams as image files (PDF, PNG, JPG, ...), including those not used in the report
  - `doc`: a folder containing two versions of your Javadoc documentation
    - `api`: a folder containing the rendered documentation of your public API, which exposes the system's functionality
    - `system`: a folder containing the rendered documentation of your entire system, including internal operations
  - `src`: a folder containing your source code
  - `report.pdf`: a PDF version of your report you handed in
  - `system.jar`: an executable JAR file of your system

When including your source code into the archive, **make sure you use the `svn export` command**, which removes unnecessary repository folders from the source tree.

Make sure you choose relevant file names for your analysis and design diagrams (e.g. `SSDsomeOperation.png`). You do **not** have to include the project file of your UML tool, only the exported diagrams.

We should be able to start your system by executing the JAR file with the following command: `java -jar system.jar`.

1.6 Peer/Self-assessment

You are asked to perform a peer/self-assessment within your team. The goal is that each team member evaluates both the other team members and himself/herself. These assessments are an incentive for you to critically reflect upon the contribution each member (including you) has delivered, as well as on the quality of the result.

Below are a number of criteria you can use to perform this assessment (these are also used by us to evaluate your work). The evaluation report consists of indicating, for each of the criteria, the *advantages* and *disadvantages* of your project solution for this second iteration, and to give a *rating* according to the following scale: poor/below average/adequate/above average/excellent. With respect to the "collaboration" criterium, you will evaluate each of your team members and yourself. The total length of your evaluation report is between 1/2 and 1 full page.

Please be fair and to the point. Your team members will not have access to your evaluation report. If the reports reveal significant problems, the project advisor may discuss these issues with you and/or your team. Please note that your score for this course will be based on the quality of the work that has been delivered, and not on how you are rated by your other team members.

The criteria to be use in your evaluation report:

- Quality of UML (clarity, correctness, ...)
- Quality of the code (correctness, defensive programming, documentation,...)
- Quality of the report (clarity, structure, completeness)
- Collaboration (teamwork, communication, commitment)

Submit your peer/self-assessment by mail to Prof. Tom Holvoet and your project advisor, using the following subject:

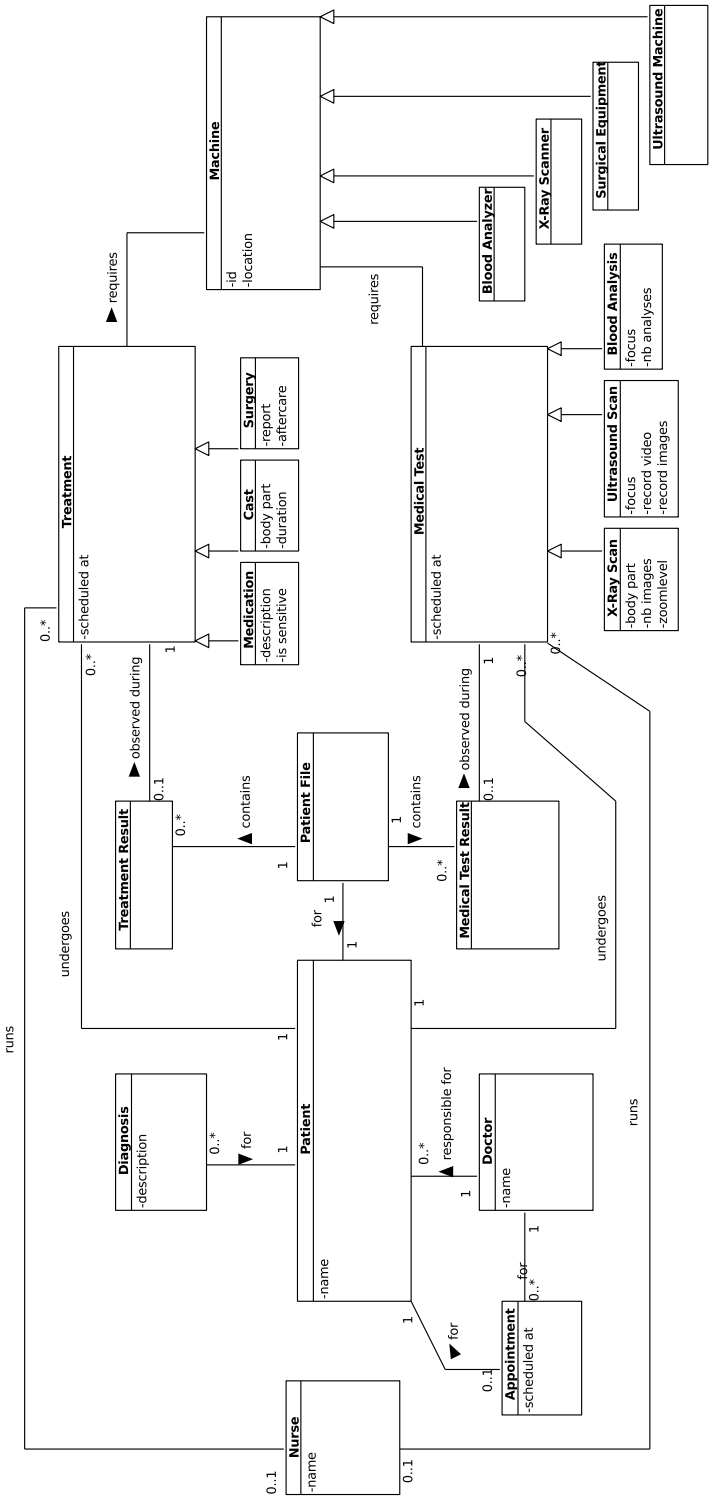
[SWOP] peer-/self-assessment of group $groupnumber$ by $firstname$ $lastname$

1.7 Deadline

- The deadline for handing in the physical report and ZIP-archive on Toledo is **November 3th, 10:00 AM**.
- The deadline for submitting your peer/self-assessment is **November 3th, 20:00**.

2 Problem Domain

2.1 Domain Model



2.2 Terminology

Doctor: a doctor works at the hospital and is known by his/her name. He is responsible for examining and diagnosing patients when they have an appointment. A doctor orders medical tests, prescribes treatments and is responsible for reviewing medical test and treatment results.

Machine: every machine in the hospital has a unique identifier and a location. Several types of machines are available. Currently, the available machine types are: X-Ray Scanner, Blood Analyzer, Ultrasound Machine and Surgical Equipment.

Nurse: a nurse works at the hospital and is known by her name. She is responsible for registering patients and carries out scheduled medical tests and treatments.

Patient: a patient comes to the hospital with specific complaints, and is known by his/her name.

Appointment: when registered at the hospital, a patient will have an appointment with a doctor, who will follow the case of the patient. An appointment is scheduled at a certain moment in time, when the doctor is available. An appointment takes 30 minutes to complete.

Medical Test: a medical test can be used to gather information about the patient's condition. This can be useful both during the diagnostic stage as well as during the treatment stage, e.g. to follow up on certain critical values. A medical test is always carried out by a nurse at the scheduled time. Several types of medical tests are available, each with their own properties:

- **X-Ray Scan:** X-ray scans reveal information about the skeleton. Properties of an X-Ray scan: body part (text), number of needed images (positive integer), level of zoom (range between 1-3). An X-ray scan requires the use of an X-ray machine and takes 15 minutes to complete.
- **Ultrasound Scan:** an ultrasound scan reveals information about the inner structure of the body. Properties of an ultrasound are: focus (text), record video (boolean), record images (boolean). An ultrasound scan requires the use of an ultrasound machine and takes 30 minutes to complete.
- **Blood Analysis:** a blood analysis reveals statistics about the patient's blood work. Properties of a blood analysis are: focus (text), number of analyses (positive integer). A blood analysis requires the use of the blood analyzer machine and takes 45 minutes to complete.

Medical Test Result: a medical test result contains information about a completed medical test, to be consulted by doctors. Each type of medical test has specific parameters that need to be present in the result:

- **X-Ray Scan:** abnormalities (text), number of images taken (positive integer)
- **Ultrasound Scan:** scan information (text), nature of scanned mass (benign, unknown or malignant)
- **Blood Analysis:** amount of blood withdrawn (positive integer), red cell count (positive integer), white cell count (positive integer), platelet count (positive integer)

Diagnosis: a diagnosis reflects the analysis made by a doctor about the complaints and symptoms of a patient. Based on a diagnosis, a doctor will decide on the correct course of treatment. In complex cases, a doctor can make a diagnosis and propose a treatment, but request a second opinion from a colleague. The diagnosis is not considered valid unless the other doctor approves with the findings in the diagnosis.

Treatment: a treatment can be ordered by the doctor based on the diagnosis and should lead to the cure of the patient. Treatments are carried out by nurses. There are several types of treatment available within the hospital, each with their own properties:

- **Medication:** a prescription of medication has the following properties: description (text), sensitive (boolean). Non-sensitive treatment using medication takes 10 minutes, sensitive medication takes 20 minutes.

Cast: a cast involves putting plaster on a body part. The properties for a cast are: body part (text), duration in days (positive integer). Creating a cast takes 120 minutes to complete.

Surgery: a surgery is an invasive treatment, to be used with care. Its properties are: description (text). A surgery requires the use of surgical equipment and takes 180 minutes to complete.

Treatment Result: a treatment result reports on the results of applying the prescribed treatment. Each type of treatment has different result properties:

Medication: abnormal reaction (boolean), report (text)

Cast: report (text)

Surgery: report (text), special aftercare (text)

Patient File: a patient file is a file containing the paperwork for a patient. It contains the information obtained about the patient during his/her stay at the hospital, including medical test results and treatment results.

2.3 Additional Domain Information

Within the system, there are several "business rules" that need to be enforced. These rules help to avoid medical mistakes or severe consequences. The rules that need to be enforced are:

- Treatments linked to a diagnosis that requires a second opinion can only be carried out after the diagnosis has received an approving second opinion
- A patient can never be subjected to more than 10 X-ray images per year. When this limit is reached, additional images will have to be scheduled in such a way that this constraint is not violated without waiting longer than necessary.

When a medical test or treatment is scheduled, a nurse needs to be assigned. Within the hospital, a number of nurses are available. Nurses' working hours are from 8-17h, every day of the week. The system automatically assigns the new task to the nurse that has the earliest available time span to complete the task.

A hospital is run by the *hospital administrator*, who is responsible for managing personnel and equipment. Within the system, the hospital administrator always exists (and is thus able to populate the system with data).

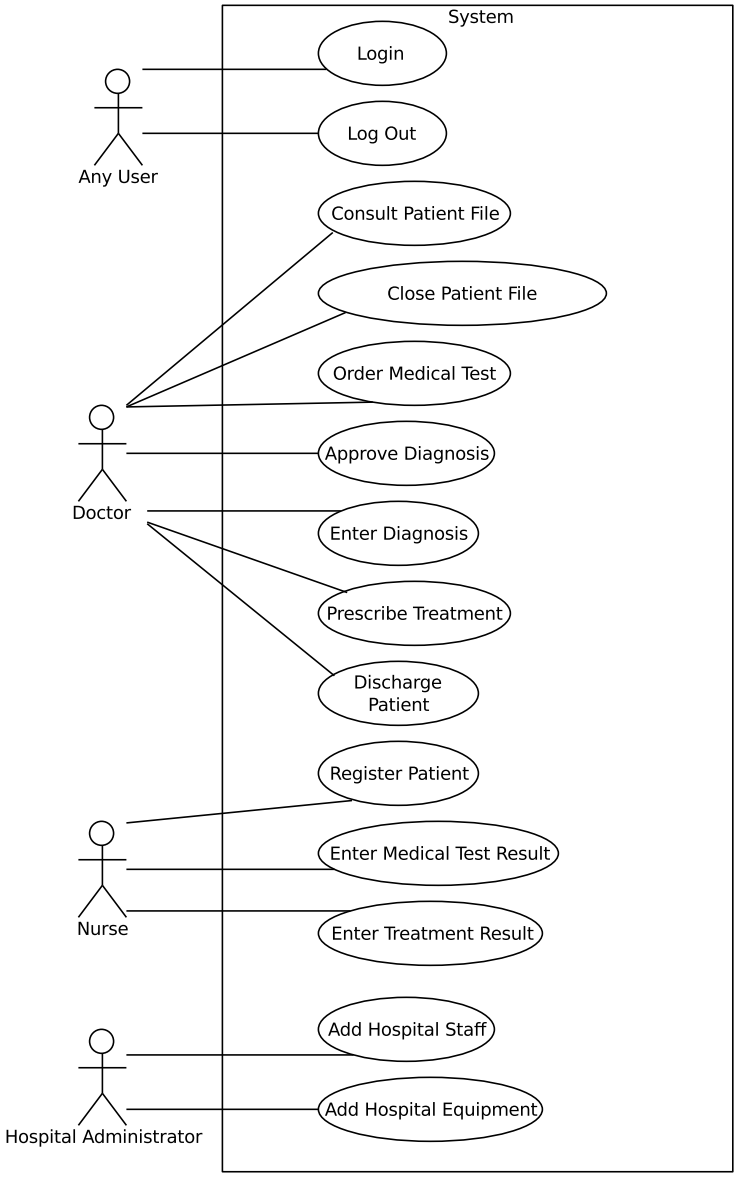
To simplify the implementation of the system, we assume to be in a time-perfect world, where scheduled tasks take exactly as long as the allocated time span, and never start late. Scheduled tasks are also never canceled, so once a task is scheduled, the allocated resources in that time span become unavailable. Unless a task can be scheduled back-to-back with another task (e.g. one task ends at 08:15, the other starts at 08:15), it should start at a full hour (e.g. 08:00 or 09:00).

You do not have to simulate advancing time within your system, but you do have to implement scheduling correctly. In order to obtain deterministic behavior, your system has to assume that the **time at startup is November 8th, 2011, 08:00**.

3 Use Cases

The use cases describe how the actors interact with the system.

3.1 Use Case Diagram



3.2 Use Case Descriptions

3.2.1 Use Case: Login

Primary Actor: Any User

Basic Flow:

1. The user indicates s/he wants to authenticate him/herself to the system
2. The system presents a list of available users and their roles

3. The user selects his/her account from the list
4. The system considers the selected user as the current user, until the user logs out

3.2.2 Use Case: Register Patient

Primary Actor: Nurse

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a nurse

Basic Flow:

1. The nurse indicates s/he wants to register a patient who arrives at the check-in desk
2. The system presents a list of previously registered patients
3. The nurse selects the correct patient from the list
4. The system registers the check-in of the patient
5. The nurse selects the doctor that will see the patient
6. The system creates an appointment for the patient with the doctor, in the next available time slot, at least one hour from the current time[1]
7. The system displays the newly created appointment to the nurse

Alternate Flow:

3. (a) The patient does not occur in the list of previously registered patients (i.e. the patient has not visited the hospital before)
    1. The nurse signals that the patient is new and needs to be created
    2. The system requests the necessary patient details
    3. The nurse enters the requested patient details
    4. The system registers the new patient
    5. *The use case continues with step 4*

3.2.3 Use Case: Consult Patient File

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Basic Flow:

1. The doctor indicates s/he wants to consult a patient file
2. The system presents the list of patients who are not discharged
3. The doctor selects a patient from the list
4. The system registers the doctor is currently consulting the patient file of the selected patient. If the doctor has already opened another patient file, this previously opened patient file is then closed
5. The system displays a list of all the medical test and treatment results of the selected patient (if any)
6. The doctor can review any of these results in detail

[1]The current time is hard coded as the startup time, as explained earlier

3.2.4 Use Case: Close Patient File

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The doctor has opened a patient file (see use case "Consult Patient File")

Basic Flow:

1. The doctor indicates s/he wants to close the patient file s/he has opened in the use case "Consult Patient File"
2. The system registers the doctor has closed the patient file

3.2.5 Use Case: Order Medical Test

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The doctor has opened a patient file (see use case "Consult Patient File")

Precondition: The patient whose file the doctor is working on is not yet discharged

Basic Flow:

1. The doctor indicates s/he wants to order a medical test for the patient whose file s/he is working on
2. The system presents the list of available medical tests
3. The doctor selects the appropriate test from the list
4. The system requests detailed input for the specific kind of test[2]
5. The doctor enters the requested information
6. The system stores this information and schedules the requested test in the next time span when all resources for the test are available, at least one hour from the current time[3]
7. The system displays the scheduled medical test and its details to the doctor

3.2.6 Use Case: Enter Diagnosis

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The doctor has opened a patient file (see use case "Consult Patient File")

Precondition: The patient whose file the doctor is working on is not yet discharged

Basic Flow:

1. The doctor indicates s/he wants to enter the diagnosis for the patient whose file s/he is working on
2. The system presents an input form for the diagnosis
3. The doctor enters the diagnosis details

[2]The properties for each test are discussed in the domain model

[3]The current time is hard coded as the startup time, as explained earlier

4. The system registers the diagnosis

Alternate Flow:

3. (a) The doctor indicates s/he wants to request a second opinion about the diagnosis
    1. The system presents a list of other doctors
    2. The doctor selects the doctor that needs to give a second opinion
    3. The doctor enters the diagnosis details
    4. The system registers the diagnosis and the request for a second opinion
    5. *The use case ends here*

3.2.7 Use Case: Prescribe Treatment

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The doctor has opened a patient file (see use case "Consult Patient File")

Precondition: The patient whose file the doctor is working on is not yet discharged

Precondition: A diagnosis has been entered (see use case "Enter Diagnosis") for the patient whose file the doctor is working on

Basic Flow:

1. The doctor indicates s/he wants to prescribe a treatment for the patient whose file s/he is working on
2. The system presents a list of available treatments
3. The doctor selects the appropriate treatment
4. The system requests detailed input for the specific kind of treatment[4]
5. The doctor enters the requested information
6. The system stores this information and schedules the requested treatment in the next time span when all resources for the treatment are available, at least one hour from the current time[5]
7. The system displays the scheduled treatment and its details to the doctor

Alternate Flow:

6. (a) The treatment belongs to a diagnosis which requires a second opinion, but has not yet been reviewed
    1. The system stores the treatment, but does not yet schedule it
    2. *The use case ends here*

3.2.8 Use Case: Approve Diagnosis

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The patient whose file the doctor is working on is not yet discharged

Basic Flow:

[4]The properties for each treatment are discussed in the domain model

[5]The current time is hard coded as the startup time, as explained earlier

1. The doctor indicates s/he wants to review a diagnosis
2. The system displays the list of diagnoses which require a second opinion from the current doctor
3. The doctor selects the appropriate diagnosis from the list
4. The system displays the details of the diagnosis and offers the doctor to review the patient's previous test/treatment results
5. The doctor approves the diagnosis that has been determined by his/her colleague
6. The system stores the decision and schedules the treatment associated with the diagnosis
7. The system displays the scheduled treatment and its details to the doctor

Alternate Flow:

5. (a) The doctor does not approve the diagnosis of his/her colleague
    1. The doctor signals his/her disagreement to the system
    2. The system marks the original diagnosis as invalid (the associated treatment is discarded and not scheduled)
    3. The system asks the doctor to enter a new diagnosis: *include use case "Enter Diagnosis"*, but automatically requires the doctor of the original diagnosis to give his/her second opinion about the new diagnosis

3.2.9 Use Case: Enter Medical Test Result

Primary Actor: Nurse

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a nurse

Basic Flow:

1. A nurse indicates s/he wants to report on a scheduled medical test that has been completed
2. The system displays a list of unfinished[6] medical tests assigned to the current nurse
3. The nurse selects the appropriate medical test
4. The system requests the necessary test information
5. The nurse enters the information
6. The system registers the results

3.2.10 Use Case: Enter Treatment Result

Primary Actor: Nurse

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a nurse

Basic Flow:

1. A nurse indicates s/he wants to report on a scheduled treatment that has been completed
2. The system displays a list of unfinished[7] treatments assigned to the current nurse
3. The nurse selects the appropriate treatment
4. The system requests the necessary test information
5. The nurse enters the information
6. The system registers the results

[6] Unfinished tests are tests for which no test result has been entered yet
[7] Unfinished treatments are treatments for which no treatment result has been entered yet

3.2.11 Use Case: Discharge Patient

Primary Actor: Doctor

Precondition: The use case "Login" has been successfully completed and the user is authenticated as a doctor

Precondition: The doctor has opened a patient file (see use case "Consult Patient File")

Precondition: The patient whose file the doctor is working on is not yet discharged

Basic Flow:

1. A doctor indicates s/he wants to discharge the currently selected patient
2. The system marks the patient as discharged

Alternate Flow:

2. (a) The patient has a diagnosis that has not been cleared by a doctor (i.e. a diagnosis which has no treatments, or which was not explicitly marked as successfully treated)
    1. The system signals an error
    2. *The use case ends here*

3.2.12 Use Case: Add Hospital Staff

Primary Actor: Hospital Administrator

Precondition: The use case "Login" has been successfully completed and the user is authenticated as the hospital administrator

Basic Flow:

1. The hospital administrator indicates s/he wants to add a new staff member
2. The system presents a list of staff member types within the hospital (i.e. nurse or doctor)
3. The hospital administrator selects the appropriate type of staff member
4. The system requests the necessary staff member details
5. The hospital administrator provides the requested information
6. The system creates the new staff member

Alternate Flow:

6. (a) The hospital administrator has entered a staff member's name that already exists within the system
    1. The system signals an error

3.2.13 Use Case: Add Hospital Equipment

Primary Actor: Hospital Administrator

Precondition: The use case "Login" has been successfully completed and the user is authenticated as the hospital administrator

Basic Flow:

1. The hospital administrator indicates s/he wants to add a new machine to the inventory
2. The system presents a list of machine types within the hospital
3. The hospital administrator selects the appropriate type of machine
4. The system requests the necessary machine details

5. The hospital administrator provides the requested information
6. The system creates the new machine

Alternate Flow:

6. (a) The hospital administrator has entered a machine identifier that already exists within the system
    1. The system signals an error

3.2.14 Use Case: Log Out

Primary Actor: Any User

Precondition: The use case "Login" has been successfully completed

Basic Flow:

1. The user indicates s/he wants to log out.
2. The system registers the user is logged out.

Good luck!
The SWOP Team members