

# Mental Hospital – Software

## Requirements & Implementation Specifications

### User Requirements

The system is a comprehensive medical platform designed to simplify the workflow of employees and designed to have access to several types of useful information about patients or rooms in one place. The platform aims to provide users with a seamless and efficient experience when staff members need to use it as a reference or to manage patient treatment.

Each room has a variety of equipment in it and has a different capacity of possible patient placement, ranging from 1 (in severe cases) to 5. Also, the system should store a history of patients' assignment to certain rooms (date of being placed in there and date of discharge).

When it comes to employees, there are nurses and therapists. The system stores their name, surname, date of birth and address, same goes to patients. For every employee there is information about the bonus that they have on top of their salary, overtime they worked per month and dates of being hired or the end of their employment. Moreover, staff members have a basic salary that is a minimum threshold (6000 zł for nurses and 10000 zł for therapists) that they must be paid regardless of their work efficiency, whereas for therapists their qualification should be provided. The final salary is made up of basic salary + bonus + estimations for the overtime( $\text{overtimePerMonth} * \text{overtimePaidPerHour}$ ), where  $\text{overtimePaidPerHour}$  is 50 and 70 for nurses and therapists respectively.

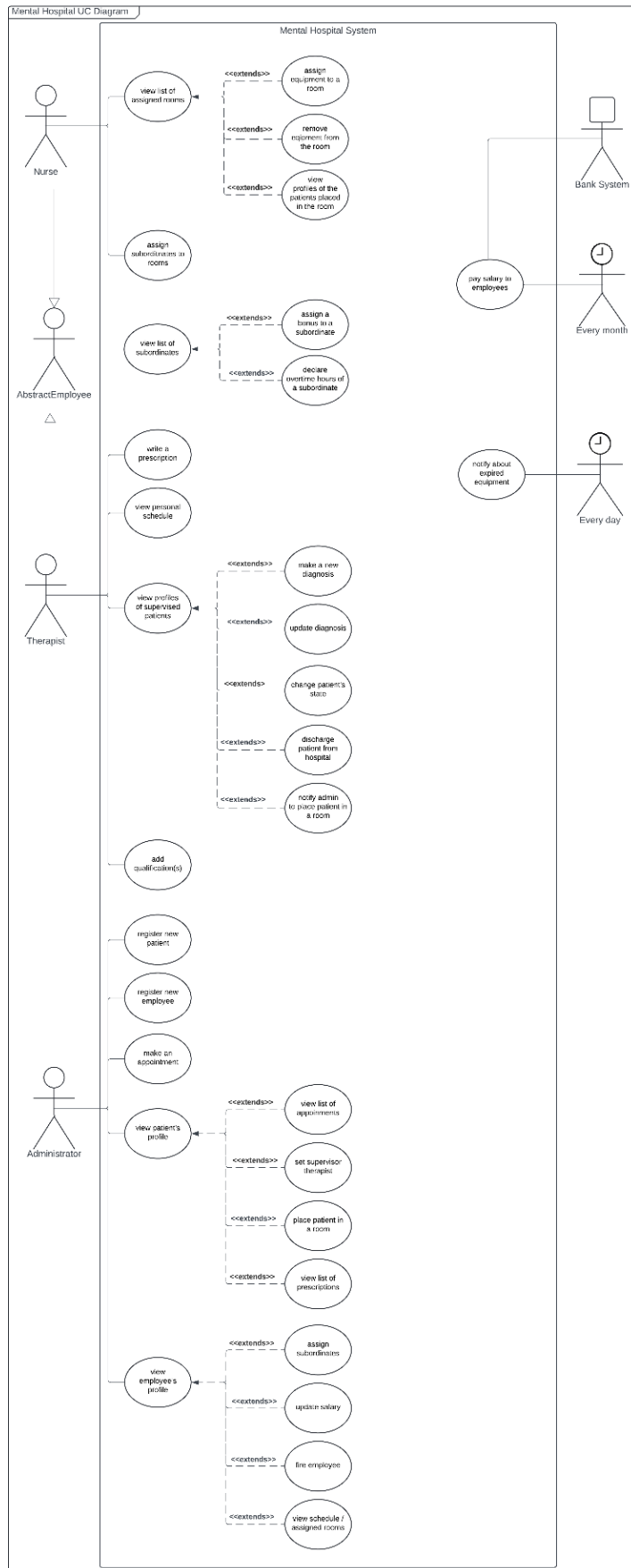
Each employee can be supervised by the other employee. Supervisors may have multiple subordinates, but it is crucial that nurses cannot supervise therapists. Also, after enough experience a nurse can become a therapist. Additionally, multiple rooms can be assigned to multiple nurses to be governed by, meanwhile for therapists there are appointments stored. Every single appointment goes with its description (how the patient felt before the visit, main conclusions and treatment decisions...) and the date of its conducting. For effectiveness of appointments there is anamnesis of every patient, which makes it easier for a therapist to construct a suitable treatment.

Every patient has a list of their diagnoses, each having its name and description, and date of death is also stored if this occurs. Furthermore, the date of being diagnosed and (if that is the case) being healed is stored to maintain a history of diseases for each patient. Diseases can be classified as severe or light, depending on the disease itself and also depending on their nature

anxiety/mood/psychotic (these are the main ones from a well-known plethora of mental illnesses that we take care of, but in the future the list in the system will be broaden). In some cases, patients may have multiple types of nature combined. For nature divisions there should be different types of data stored: anxiety – list of triggers, mood – list of psychedelics consumed, psychotic – list of hallucinations that the patient has. As an additional measure on severe cases, staff members have information on the level of danger of such patients (low, medium and high) and also if physical restraint is required.

Needless to say, that consultations with therapists are not enough to cure the disease in many cases. For such medical conditions therapists can make multiple prescriptions during appointments. Each prescription has the name of the drug, quantity that the patient can purchase, dosage needed and the overall description of how to take the medicine. We want to archive all prescriptions that can be searched by a unique identifier.

# Use case diagram



# Use case scenario: “Make a new Diagnosis”

## Make diagnosis UC scenario

Actor: Therapist

Purpose and context: Therapist adds diagnosis for the patient in the system

Assumptions: 1-patient already exists in system  
2-Appointment already took place and info is available  
3-Type of the given disease is already implemented in the system

Pre-conditions: Therapist chose patient or registered new one, and selected it for adding a diagnosis

Initiating business event: Click "Add New Diagnosis" button

Basic flow of events:

- 1-Therapist clicks "Add New Diagnosis" button
- 2-System displays form for adding new diagnosis
- 3-Therapist enters all initial data (name, description, date)
- 4-Therapist selects Severe type of the disease
- 5-System displays Severe specific fields (level of danger, restrictions)
- 6-Therapist fills fields for Severe type
- 7-System displays selection of diseases nature type (anxiety, mood, psychotic)
- 8-Therapist chooses Anxiety
- 9-System display field for triggers input
- 10-Therapist fills in triggers
- 11-System displays ability to add another nature for the disease (mixed-nature disease)
- 12-Therapist selects not to add another nature
- 13-Therapist clicks submit
- 14-System validates input
- 15-System displays result on the screen and adds record to the database

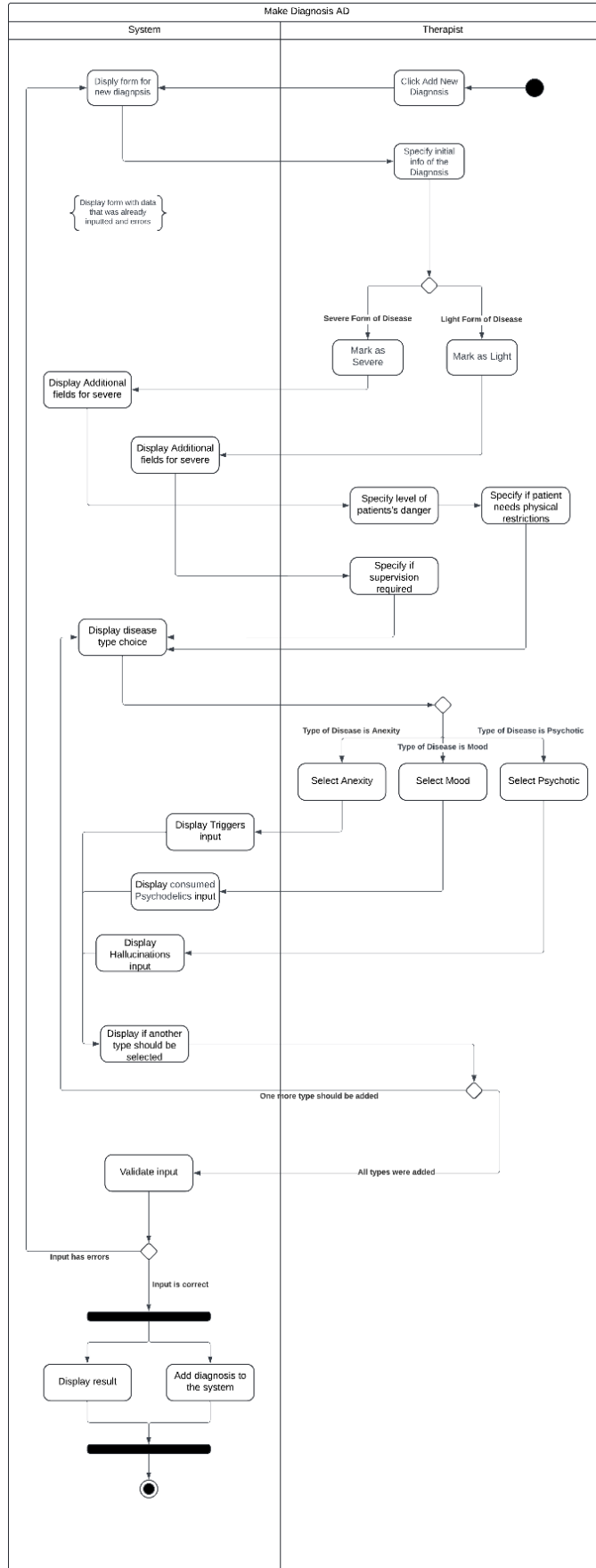
Alternative flow of events:

- **Disease with mixed nature type:**
  - 12a1-Therapist adds Psychotic type of disease
  - 12a2-System display additional fields for Psychotic disease
  - 12a3-Therapist fills additional fields for Psychotic disease
  - 12a4-System displays ability to add another nature for the disease
  - 12a5-Therapist selects not to add another nature
  - 12a6-continue to step 13
- **Input did not pass validation:**
  - 15a1-System displays form with the previous data and highlighted errors in it
  - 15a2 -Therapist fixes errors where needed
  - 15a3-continue to step 14
- **Therapist chose to create Light Disease**
  - 4a1-Therapist selects Light type of the disease
  - 4a2-System displays Light specific fields (is supervision required)
  - 4a3-Therapist fills fields for Light type
  - 4a4-continue to step 7
- **Therapist chose to create Mood disease**
  - 8a1-Therapist chooses Mood
  - 8a2-System display field for consumed psychedelics input
  - 8a3-Therapist fills in consumed psychedelics
  - 8a4-continue to step 11
- **Therapist chose to create Psychotic disease**
  - 8b1-Therapist chooses Psychotic
  - 8b2-System display field for hallucinations input
  - 8b3-Therapist fills in hallucinations
  - 8b4-continue to step 11

Post conditions:

Diagnosis is added to the patient (associated with patient)

# Activity Diagram: “Make a new Diagnosis”



# Use case scenario: “Assign equipment to a room”

## Assign equipment to a room UC scenario

Actor: Nurse

Purpose and context: A nurse wants to add a currently available equipment to one of the rooms they are assigned to

Assumptions:

- 1 - all employees immediately record their actions in the system, so there are no errors and all the equipment displayed on the list of available equipment is physically available and is not assigned to any other room
- 2 - while the nurse is assigning new equipment to a room it is unavailable to other employees on their list of available equipments

Initiating business event: The nurse chose "assign equipment" option on the main page

Basic flow of events:

- 1 - The nurse clicks "assign equipment" button
- 2 - The system displays a list of currently available equipment
- 3 - The nurse chooses the required equipment by clicking on it
- 4 - The system displays a list of rooms the nurse is assigned to
- 5 - The nurse chooses a room from the list
- 6 - The system assigns equipment to the room in the database and displays a message that equipment was assigned successfully

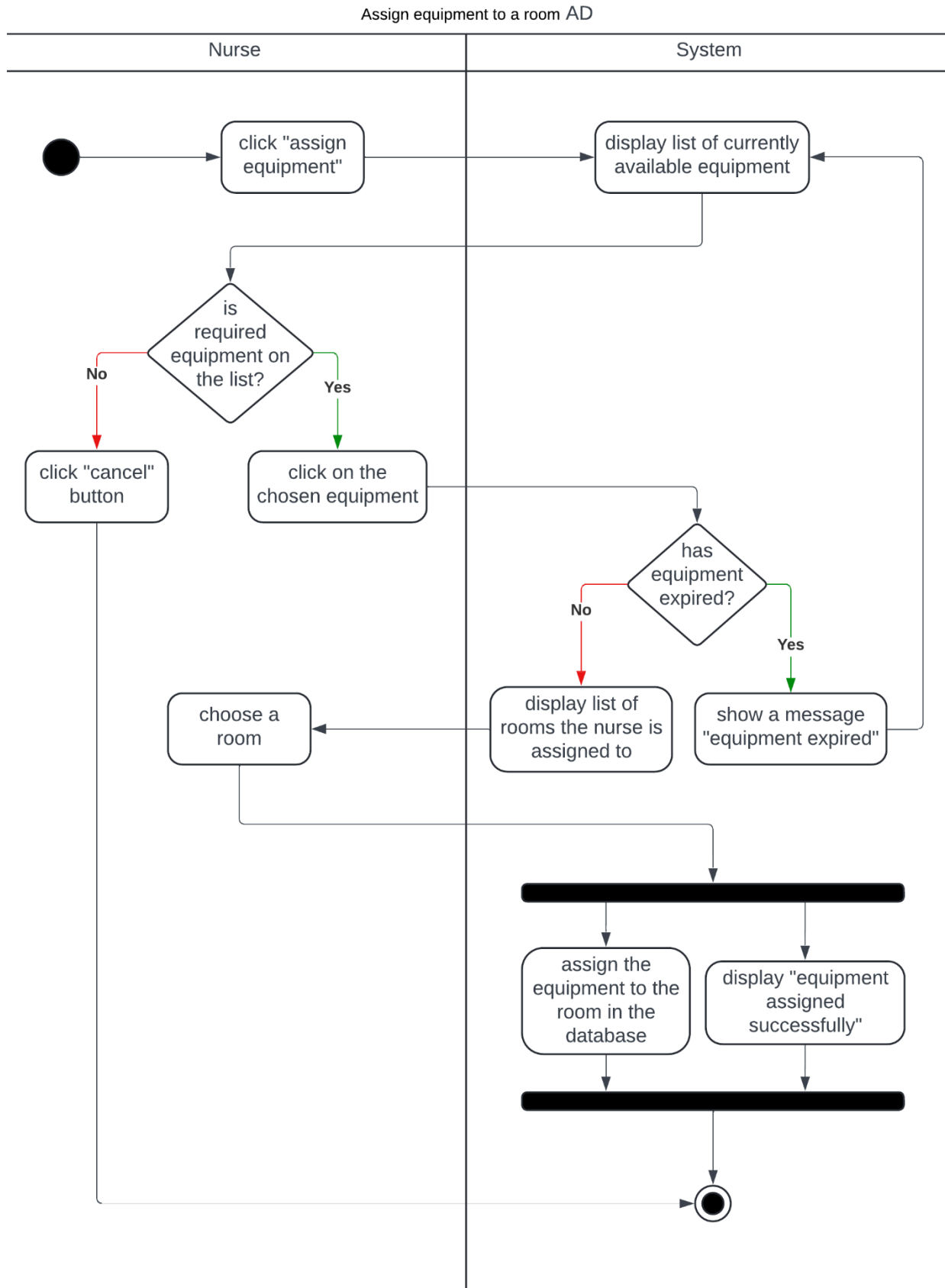
Alternative flow of events:

- **The required equipment is not on the list of available ones:**
  - 2a1 - The nurse clicks "cancel" button
- **The chosen equipment has expired:**
  - 5a1 - The system shows a message that equipment has expired
  - 5a2 - Continue to step 2

Post conditions:

- **Basic:** The equipment was assigned to the room in the database, information about successful assignment displayed to the user
- **Required equipment currently unavailable:** No changes were made in the system
- **The equipment has expired:** No changes were made in the system

# Activity Diagram: "Assign equipment to a room"



## Use case scenario: “Place patient in a room”

### Place patient in a room UC scenario

Actor: Admin

Purpose and context: Admin places the selected user into a room

Assumptions: 1- room has enough capacity left 2-patient is already registered in a system 3- room is already registered in a system 4 - patient is not currently placed in a room

Pre-conditions: Admin opened patient's profile

Initiating business event: Click "Place patient in a room" button

Basic flow of events:

- 1-Admin clicks "Place patient in a room" button
- 2-System displays a list of rooms available
- 3-Admin selects the room
- 4-Admin confirms the choice
- 5-System adds patient to the room
- 6-System returns to the patient's profile page

Alternative flow of events:

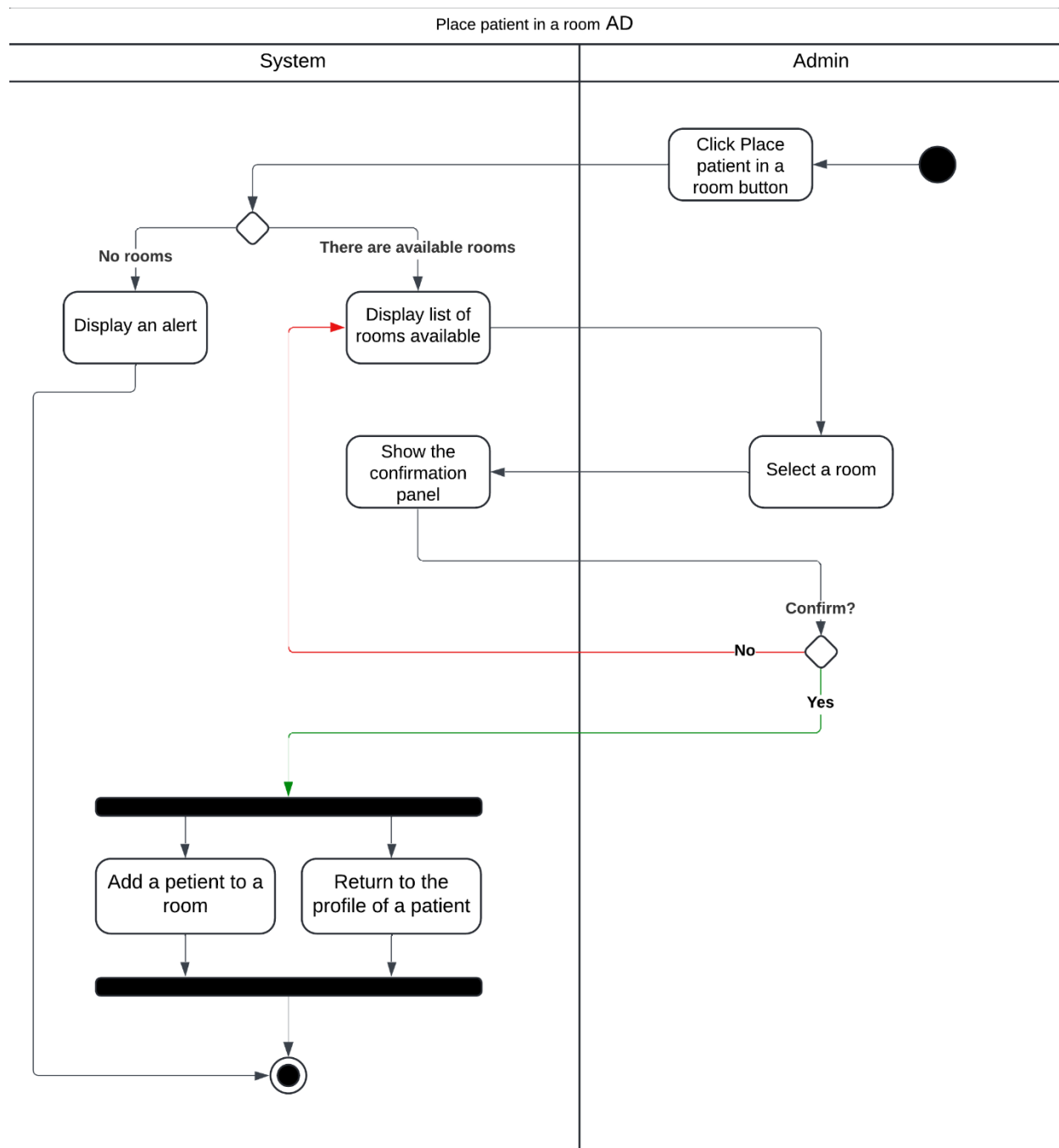
- **No available rooms:**
  - 2a1-System displays an alert
- **Admin did not confirm the choice:**
  - 5a1-continue to step 2

Post conditions:

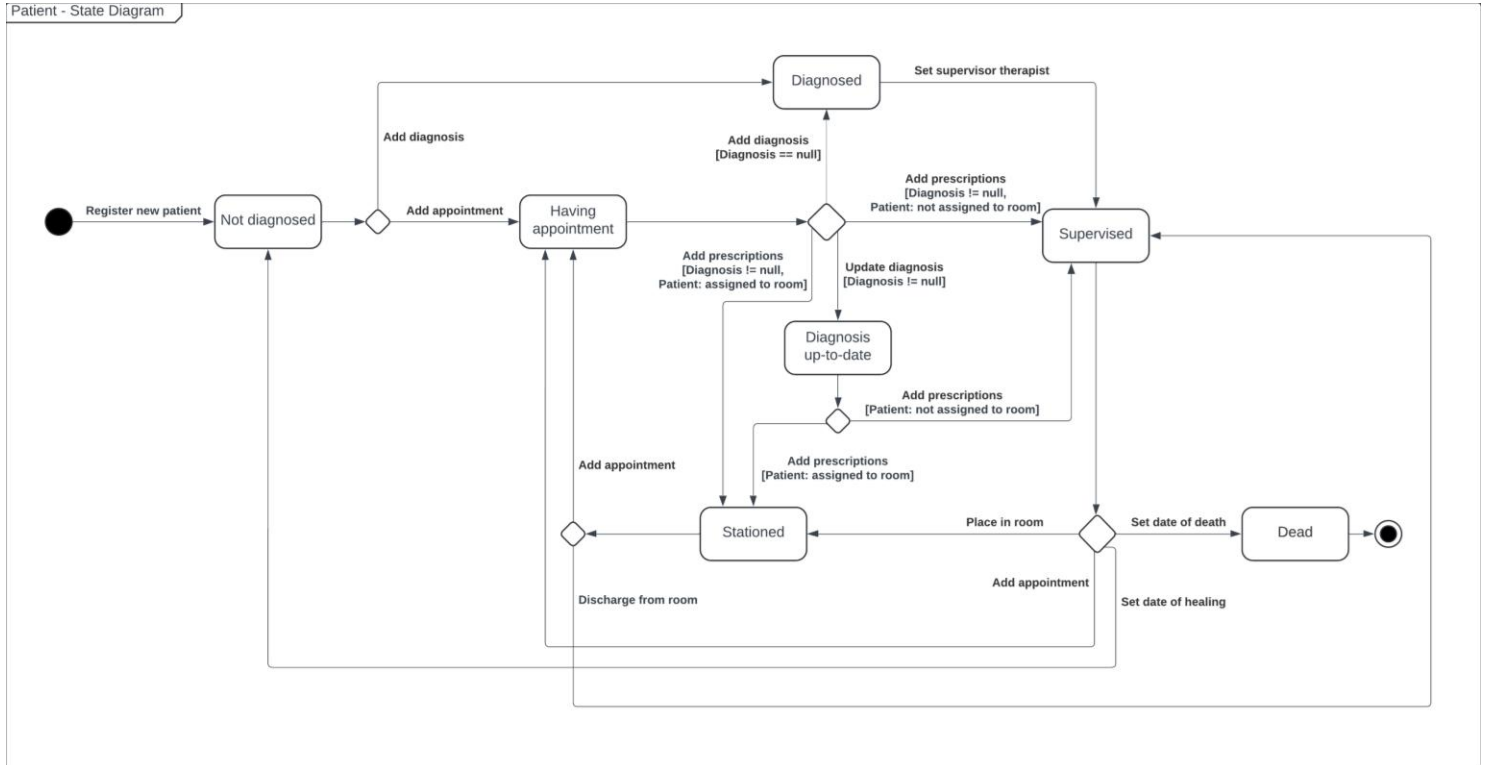
Patient is added to the room



# Activity Diagram: “Place patient in a room”



# State Diagram



## State Diagram – Design Description

In accordance with the state diagram of Patient Objects depicted above, a new instance of Patient with a status “**Not diagnosed**” is created afterwards the call of the RegisterNewPatient() method, which is implemented in PersonFactory class. In the following stage there are two possible outcomes:

- 1) “Diagnosed” - In case when diagnosis is already known
- 2) “Having appointment” - Supposing a professional examination is needed

When it comes to the first one, Patient is assigned to a Therapist and becomes “Supervised”, whereas from the second position options vary:

- 1) “Diagnosed” - Which leads to the flow described before (only when diagnosis is null)
- 2) “Supervised” - Same position of a Patient as described previously, yet only accessible when conditions of not null diagnosis and patient not being assigned to a room are met
- 3) “Diagnosis up-to-date” - Possible when diagnosis is not null, meaning the Patient has been diagnosed before
- 4) “Stationed” - Also only possible when the Patient is diagnosed, but requires them to be placed in a room

Room allocation is checked with the help ViewListOfPatients() method of Room class.

From the “Diagnosis up-to-date” state the Patient can go to(dependent on room allocation):

- 1) “Stationed” - In case when assigned to a room
- 2) “Supervised” - In case when not

Referring to “Stationed” state there are such possibilities:

- 1) “Having appointment” - Planned visit to renew medicine or update the diagnosis
- 2) “Supervised” - If room placement is no longer necessary or in-between the change

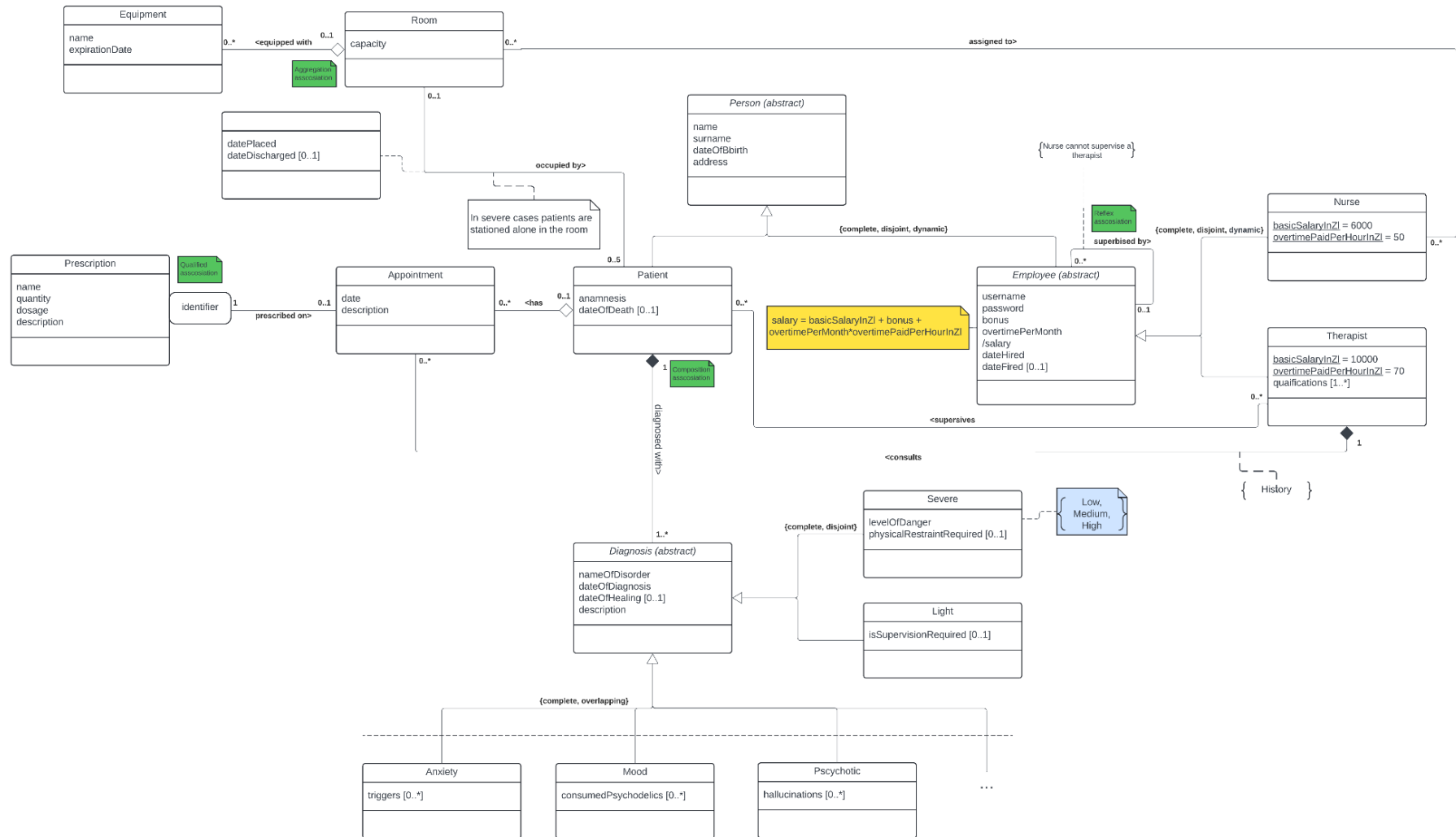
When the Patient is in “Supervised” state they can move on to:

- 1) “Stationed” - Reserved for cases when room placement is vital
- 2) “Having appointment” - Planned visit to renew medicine or update the diagnosis
- 3) “Not diagnosed” - In case the treatment succeeded, and the Patient is healed

The only possibility for the Patient to have the state ended is through Supervised. After the occurrence of this accident the system sets the date of death to the Patient and removes them from the supervised patients from a corresponding therapist.

# Class diagram – Analytical

Mental hospital Analytical Diagram



## Analytical Class diagram - Design Description

The above analytical class diagram visualizes the mental hospital system. The system allows hospital's employees to access the needed information about patients, their diagnoses, rooms they are assigned to, and equipment available in the rooms.

Employees can monitor all the information regarding the rooms. Each room has a variety of equipment in it and has a different capacity for possible patient placement.

When it comes to employees, there are nurses and therapists. The system stores their data, the same goes for patients. Employees' specific information includes: dates of hiring and dismissal, the bonus money they will receive in addition to their salary, and number of hours they worked overtime this month and amount of money each hour of overtime will cost (50 for nurses and 70 for therapists). Moreover, every employee has a basic salary that is a minimum threshold for their salary (6000 zl for nurses and 10000 zl for therapists), to which all additional bonuses and overtime are being added. Information about the therapist's qualification should be provided. The final salary is calculated according to the following formula:  $\text{basic salary} + \text{bonus} + \text{overtime} (\text{overtimePerMonth} * \text{overtimePaidPerHour})$ . Each employee can be supervised by the other employee or/and be a supervisor, but it is crucial that nurses cannot supervise therapists. Supervisors may have multiple subordinates. Nurse can become a therapist if they acquire enough experience (qualifications must be provided).

Therapists can make multiple prescriptions during appointments. Each prescription has the name of the drug, quantity that the patient can purchase, dosage needed and the overall description of how to take the medicine. All prescriptions are archived and can be found by a unique identifier.

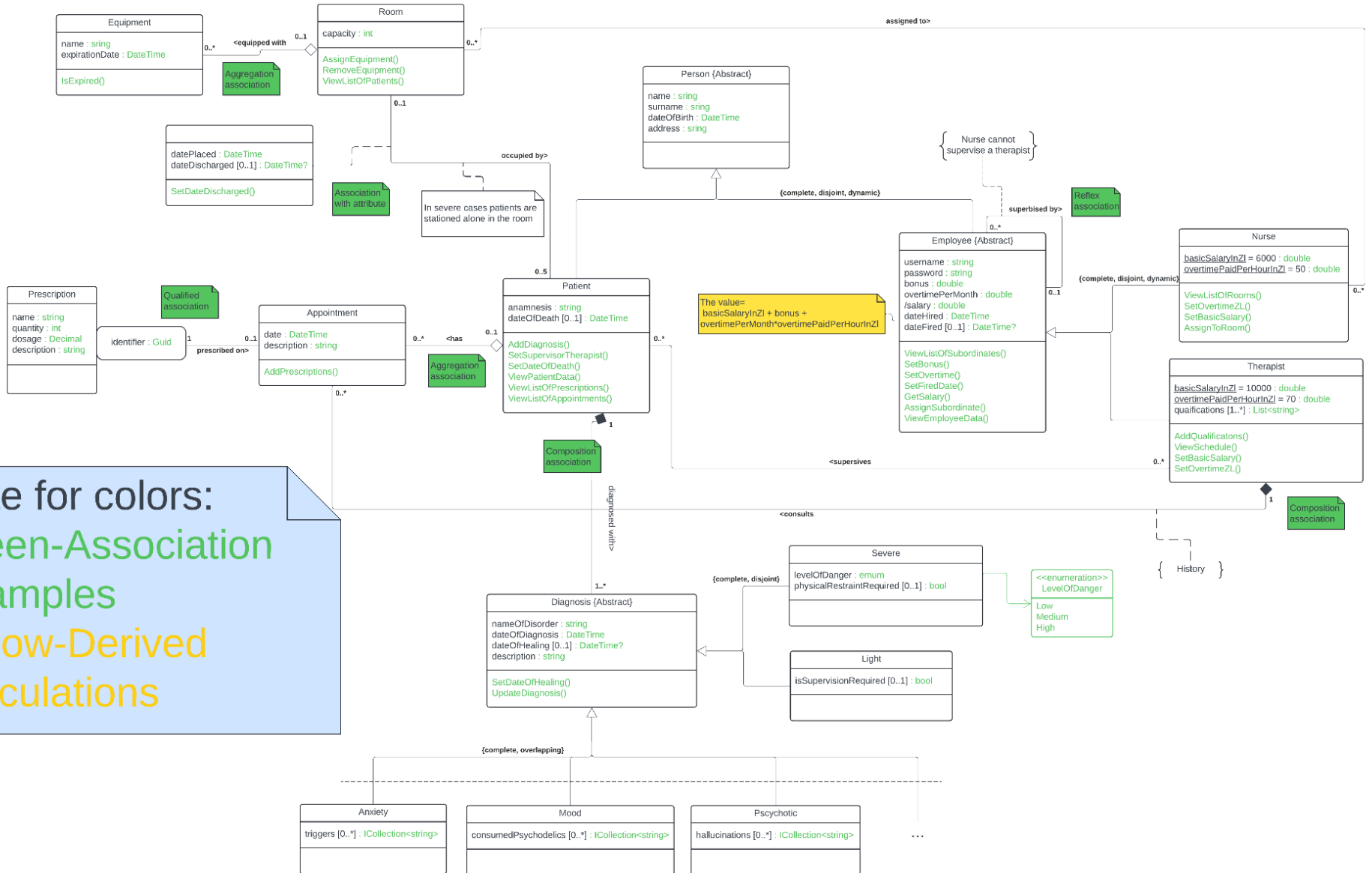
A number of rooms can be assigned to multiple nurses to be managed by, meanwhile therapists conduct appointments, which should be stored in the system. Every single appointment goes with its description (how the patient felt before the visit, main conclusions and treatment decisions...) and the date it took place.

Every patient has anamnesis, a list of their diagnoses, and date of death if this occurs. Furthermore, the date of being diagnosed and (if that is the case) being healed is stored to maintain a history of diseases for each patient.

Diseases can be classified as severe or light, depending on the disease itself, its nature (anxiety, mood, psychotic, and the list will be broadened). In some cases, patients may have diseases of multiple types combined. For nature divisions there should be different types of data stored: anxiety – list of triggers, mood – list of psychedelics consumed, psychotic – list of hallucinations that the patient has. As an additional measure on severe cases, staff members have information on the level of danger of such patients (low, medium and high) and if physical restraint is required.

# Design diagram – Initial version

Mental hospital Class Diagram Design



## Design Decisions

Notwithstanding the fact that particular aspects of UML are not present in C#, the above diagram represents vital workarounds to implement core functionality of the system. The solution utilizes DI which allows efficient object usage.

In the project implementation there are 4 different types of classes: Models, Factories, Validators, Storages, and one additional Service class which main functionality is managing serialization.

**Models** represent classes from the class diagram, whereas corresponding associations are present in the form of a single object, generic `AssociationCollection<T>` or `AssociationDictionary<T>` fields, where T represents a Model. Fields of the generic classes include `List<Guid>` or `Keys ICollection` and `List<T>` or `Values ICollection` as inner fields respectively, as well as methods for their management, including `Add()` to add new associations, `Remove()` to remove associations, and `RestoreObjects()` method for restoring associations after serialization.

**Factories** are classes where Model classes are being created, validated and added to the appropriate Storage.

**Validator** classes ensure that data provided by the user is suitable for the given type of object, along with clear error messages regarding every specific field where something was input wrong.

**Storage** is a generic class (meaning that it is a template for different types, that reduces the amount of code, as there is no need for writing the same code, but for diverse attribute types), in which generic attributes represent the type of objects that are being stored inside. Moreover, Storage is able to add new objects, delete objects from the storage, ensuring that this object will be deleted from all related objects as well, serializing and deserializing along with restoring all of the connection with other objects, finding objects from the storage by any predicate; all object specific manipulations are implemented via `StorageActions` and association handled via `AssociationCollection` or `AssociationDictionary`.

## Attribute Validations

Attribute level validation was implemented using `FluentValidation`, a .NET library for building validation rules. In factory, when creating a new instance of a class, resulting object's attributes are always checked according to the rules specified in a corresponding Validator class. Each Validator class inherits from `AbstractValidator<T>`, where T is the type of the class that will be validated. Inheritor classes Validators include rules from the base class, as well as additional rules specific to this particular inheritor.

## Optional attributes

In the project, implementation optional attributes are represented by nullable objects. In the class diagram, every instance of optional attributes is indicated with `[0..1]` notation, in the code implementation of the project dates, such as date of death, date of healing, etc., are denoted as

DateTime? class, implying that date can be set to null. In C#, DateTime cannot be null, meaning that null values are set to DateTime.MinValue (DateTime.MinValue equals 01.01.0001).

## Multi-value attributes

Multi-value attributes in C# are classes that implement IEnumerable or/and ICollection interfaces. IEnumerable interface represents the set of objects that can be iterated/looped through, ICollection has the same functionality but with some additional features. In classes, which can be found in Models directory, multi-value attributes are declared as IEnumerable or ICollection interfaces, to provide flexibility in the further implementation, as these attributes are initialized as List or another class implementing IEnumerable or/and ICollection interfaces.

## Derived attributes

Being reliant on automation of the relevant data retrieval, derived attributes are depicted throughout the project with the help of mechanism embedment directly into a RecalculateSalary() method. Not only is it implemented in a neat way, but it also enables recalculation, which is done by aggregating the segmental attributes on demand.

## Tagged value

In the project, tagged values are implemented via enums. Custom enumeration class LevelOfDanger with 3 options (Low, Medium, High) was created, which is represented on the design class diagram as well.

## General association implementation

Associations represent the relationships between different entities in the project. In order to handle different types of associations (one-to-one, one-to-many, many-to-many) they are implemented using navigation and collection properties.

**Navigation properties** are stored as an object and an id (ids are stored separately to avoid infinite recursion during serialization). If an association represented by a single object is modified, a reversed association is established via the setter property.

**Collection properties** are implemented via generic AssociationCollection<T> class, with inner List<Guid> for saving ids and List<T> storing objects of type T, and AssociationDictionary<T>, representing a dictionary for a qualified association where Keys ICollection stores ids and Values stores objects. The given classes implement RestoreObjects() method of IAssociationCollection interface, as well as the main Add() and Remove() methods of ICollection<T> and IDictionary<Guid,T> interfaces respectively to ensure that each association is bidirectional and updates to one side of the relationship are reflected in the other.

### Add Method:

The Add() method is used to establish a new association. This method is implemented in both AssociationCollection<T> and AssociationDictionary<T> and accepts an object of a generic type T,



saves its id to the inner field `List<Guid>` or `Keys ICollection` and the object to `List<T>` or `Values ICollection` respectively, then uses reflection to add the reverse connection.

During reflection the method looks through attribute properties of a “parent” object - which stores the `AssociationCollection<T>` or `AssociationDictionary<T>` on which the `Add()` method was invoked initially – and searches for either a single object, `AssociationCollection<S>` or `AssociationDictionary<S>` of the same type with the “parent” object. If it was found, the reflection is used to get respective method and add/set the object to that collection/dictionary/single object, ensuring the bidirectional association.

#### Remove Method:

The `Remove()` method is used to detach an association. This method is implemented in both `AssociationCollection<T>` and `AssociationDictionary<T>` and accepts an object of a generic type `T`, removes its id from `List<Guid>` or `Keys ICollection` respectively, uses reflection to detach the reverse connection that could be either a single object, `AssociationCollection<S>` or `AssociationDictionary<S>`, and finally removes the object from `List<T>` or `Values ICollection` respectively.

#### Change/Modify Method:

The modification of an association consists of first removing the old one and adding a new one. It involves calling `Remove()` method to remove the old association and `Add()` method to add a new one, both methods ensuring the reverse connections are updated respectively.

#### Restore Objects Method:

The `RestoreObjects()` method is defined in `IAssociationCollection` interface and is implemented by both `AssociationCollection<T>` and `AssociationDictionary<T>`. The method uses `List<Guid>` or `Keys ICollection` to fill the `List<T>` or `Values ICollection` respectively with the objects from a `Storage<T>` after serialization.

## Composition

Composition represents a "part-of" relationship where the existence and lifespan of a contained entity are tightly bound to the containing entity. In the project, an example of qualified association is between `Diagnosis` and `Patient` entities, where a diagnosis will be deleted along with a patient.

The composition is ensured via creation of a custom `CompositionAttribute` class used to annotate either `AssociationCollection<T>` or `AssociationDictionary<T>` in a `Model`. When `Remove()` method is called the association is checked for the above annotation and if present the child entity is deleted alongside removing the association.

## Qualified association

Qualified association involves a key or a qualifier to uniquely identify the associated entity. In the project the qualified association is created between Prescription and Appointment entities, where Prescription can be referred from Appointment by a qualifier of type Guid.

To implement this type of association was created a generic class `AssociationDictionary<T>` that implements `IDictionary<Guid,T>` and `IAssociationCollection` interfaces and stores Keys and Values `ICollection`s as its inner fields. The class implements basic `Add()` and `Remove()` methods which use reflection to ensure bidirectional update of an association.

## Association class/attribute

Association classes are used when the relationship itself has attributes. In the project, the example of the given association is between Patient and Room entities having an additional `RoomPatient` class implemented.

The additional class stores association attributes as well as saves the association itself. In order to change/modify the association the instance of `RoomPatient` must be removed from the `Storage<RoomPatient>` and a new instance added instead.

## Reflex Association

Reflex association occurs when an entity is associated with itself. In the project the reflex association is present in `Employee` entity. The given association is handled following general association implementation without any additional measures required.

## Association validation

Validation ensures the integrity of associations like preventing duplicate associations or infinite loops caused by bidirectional dependencies.

To ensure no duplicate associations made, `Add()` method checks if an Id or a Key is present for `AssociationCollection<T>` or `AssociationDictionary<T>` classes respectively, and if found, the method is returned immediately without adding association again.

To prevent infinite recursion loops both `Add()` and `Remove()` method check if an Id or a Key was already added/removed for `AssociationCollection<T>` or `AssociationDictionary<T>` classes respectively, and if found, the method is returned immediately without adding/removing association again. In case an association represented by a single object is added a new one by assigning an object or removed by setting it to null, a reversed association is updated respectively via the setter property.

Additionally, to avoid infinite recursion during serialization all the associations are represented in two ways: a single object association stores both object and id fields, as well as `AssociationCollection<T>` and `AssociationDictionary<T>` store collections of objects and ids separately. The associations represented by objects are marked with `JsonIgnore` annotation, hence only ids representations of associations are serialized. This way there is no possibility of recursive

serialization to occur, and all the associations are restored during deserialization based on associations saved with the ids in RestoreObjects() method of a IAssociationCollection interface.

## Dynamic Analysis

After performing dynamic analysis of the above-presented diagrams, the need for expansion of design class diagram was indicated. New methods for accommodation of the requirements and correct behavior of the system need to be added. All introduced changes can be seen in the below presented final design class diagram.

- **Dynamic Analysis on “Place patient in a room” use case**

Methods that were added:

- ViewPatientData() was added in Patient to display all information available for a particular patient in the section “Info” (base) of the Patient Profile page.
- ViewRoomData() was added in Room so that admins can see whether the selected room really fits for the needs of the patient.
- PlacePatient() was added in Room in order to associate patient with the chosen by the admin room.

- **Dynamic Analysis on “Make diagnosis” use case**

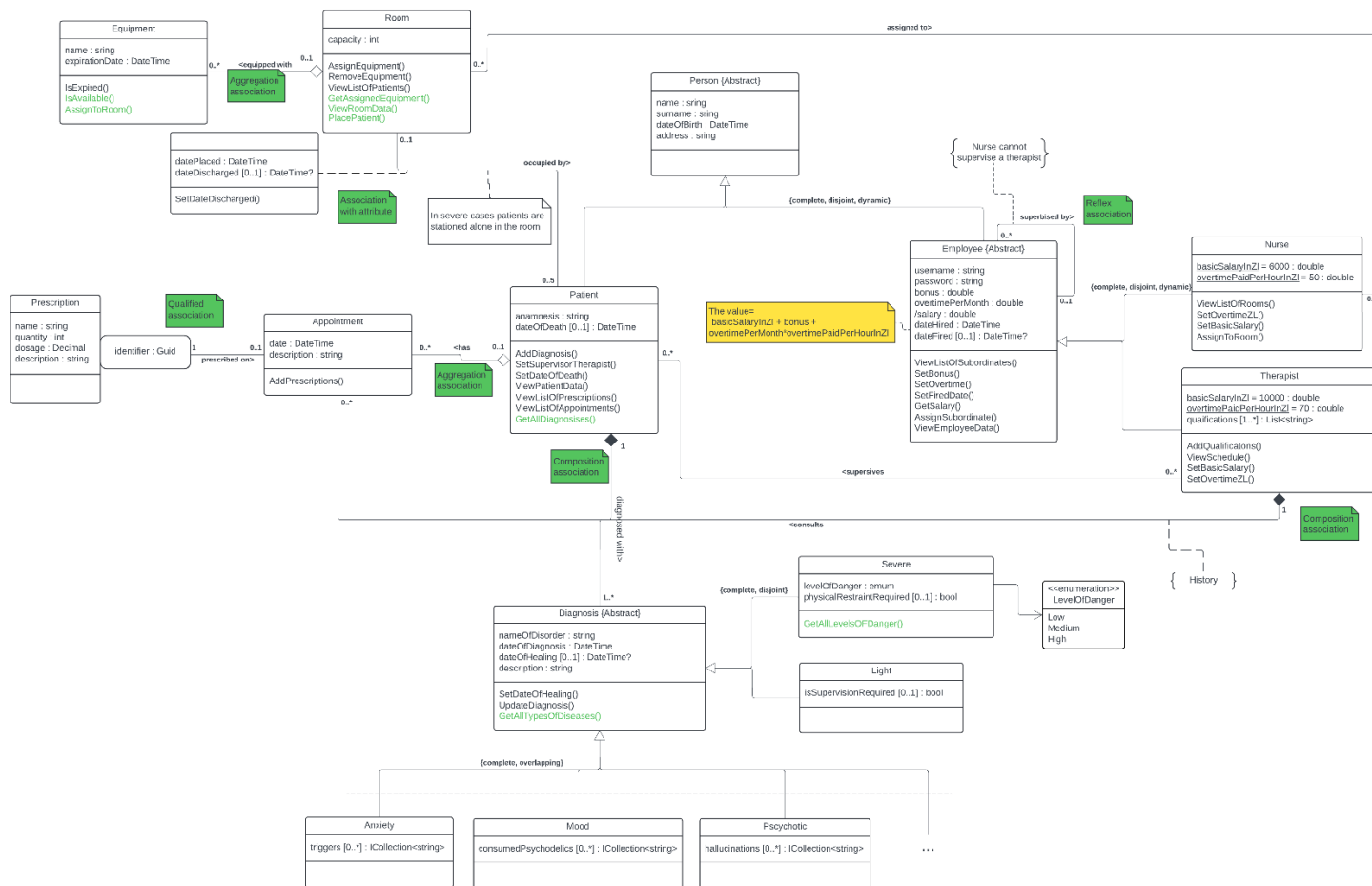
Methods that were added:

- GetAllDiagnoses() was added in Patient to correctly display the page with the list of all diagnoses related to the specific patient, and ability to add new.
- GetAllLevelsOfDanger() was added in Severe to provide information about levels of danger stored in our system to display the dropdown list with selection for the easier use when creating a new diagnosis.
- GetAllTypesOfDiseases() was added in Diagnosis to provide information about different types of conditions natures stored in our system to display the dropdown list with selection for the easier use and representation of available options when creating a new diagnosis.

- **Dynamic Analysis on “Assign equipment to a room” use case**

Methods that were added:

- GetAssignedEquipment() was added in Room to display the initial page for the use case with the list of currently assigned equipment to the specifically chosen room.
- IsAvailable() was added in Equipment for the system to be capable to check if the equipment date has passed and should it allow the user to assign it to a room or show an error in a negative case.
- AssignToRoom() was added in Equipment to assign the searched equipment to the chosen by the nurse room and finalize the use case process.



## GUI Design discussion

Throughout the next part the main goal is to show a GUI mockup, mentioning the reasoning behind its implementation and design choices. Constructing GUI in Marvelapp in compliance with Dynamic analysis, which stated crucial interface parts and features to support the run of use cases, several additional measures were taken in order to provide neat and smooth user experience.

Being accomplished with key objective of guaranteeing a vivid picture of the design logic and architecture, this mockup lacks an implementation of a specific set of features illustrated due to the fact that not every single one is a vital component of a particular use case.

- Place patient in a room

Info Anamnesis Appointments Prescriptions Diagnoses Placement history

### Patient

Name: patient name

Surname: patient surname

Date of birth: 01/01/2025

Address: SomeStreet 1,1

Room: not assigned Assign

Therapist: Abobo Abaaba

Patients

Appointments

Employees

Rooms

### Place patient in a room (Step 1)

The page presented above depicts an admin panel after opening patient's profile. On every page of the panel itself, sections with patients, employees, appointments and rooms can be found. In case of the patient's tab, there are modules with related information, anamnesis, appointments, prescriptions, diagnosis and placement history. The opened patient panel lists basic information about the patient (Name, Surname, Date of birth, Address, room the patient was placed in and therapist's name and surname. Additionally, if the patient is dead, Date of death will be shown. Apart from that, as current patient is not placed in any room, there is a button "Assign", which initiates the action. Apart from that, as current patient is not placed in any room, there is a button "Assign", which initiates the action. Once the admin clicks on the button, a new page is loaded with a list of available rooms to choose from.

## Patient

Available rooms

▼

Option 1

Option 2

Option 3

Option 4

Select

Patients

Appointments

Employees

Rooms

Place patient in a room (Step 2)

After the list is rendered by the system, admin can select only one of the rooms mentioned for the future patient placement.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

# Patient

Available rooms

▼

Option 1

Option 2

Option 3

Option 4

Select

Patients

Appointments

Employees

Rooms

Place patient in a room (Step 3)

Admin selects the room to place a patient in which is followed by clicking the “Select” button. Having done this, the system redirects to a confirmation page.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

# Patient

Are you sure you want to place  
a patient in this room?

Room Number: 2

Capacity: 5

Number of patients: 3

Equipment: ECT, TMS, MRI, CT, EEG

Cancel

Confirm

Patients

Appointments

Employees

Rooms

Place patient in a room (Step 4 and 5)



The page mentions all of the necessary information about the selected room, which makes it easier to refine the choice. Having checked the data shown, admin can proceed to executing the procedure by clicking “Confirm” button.

Info	Anamnesis	Appointments	Prescriptions	Diagnoses	Placement history
------	-----------	--------------	---------------	-----------	-------------------

## Patient

**Name:** patient name

**Surname:** patient surname

**Date of birth:** 01/01/2025

**Address:** SomeStreet 1,1

**Date of death:** 02/01/2025

**Room:** N 2 Discharge


**Therapist:** Abobo Abaaba

Patients
Appointments
Employees
Rooms

Place patient in a room (Step 6)

The final stage is returning to base of patient’s profile, however, this time showing the room assigned with an option to discharge a patient from the room in a form of “Discharge” button.

Info	Anamnesis	Appointments	Prescriptions	Diagnoses	Placement history
------	-----------	--------------	---------------	-----------	-------------------

There are no available rooms. 

## Patient

**Name:** patient name

**Surname:** patient surname

**Date of birth:** 01/01/2025

**Address:** SomeStreet 1,1

**Room:** not assigned Assign

**Therapist:** Abobo Abaaba

Patients
Appointments
Employees
Rooms

Place patient in a room (Step 2a1)

When there are no available rooms for disposal, the system throws an alert mentioning this fact.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

## Patient

Available rooms

▼

Option 1

Option 2

Option 3

Option 4

Select

Patients

Appointments

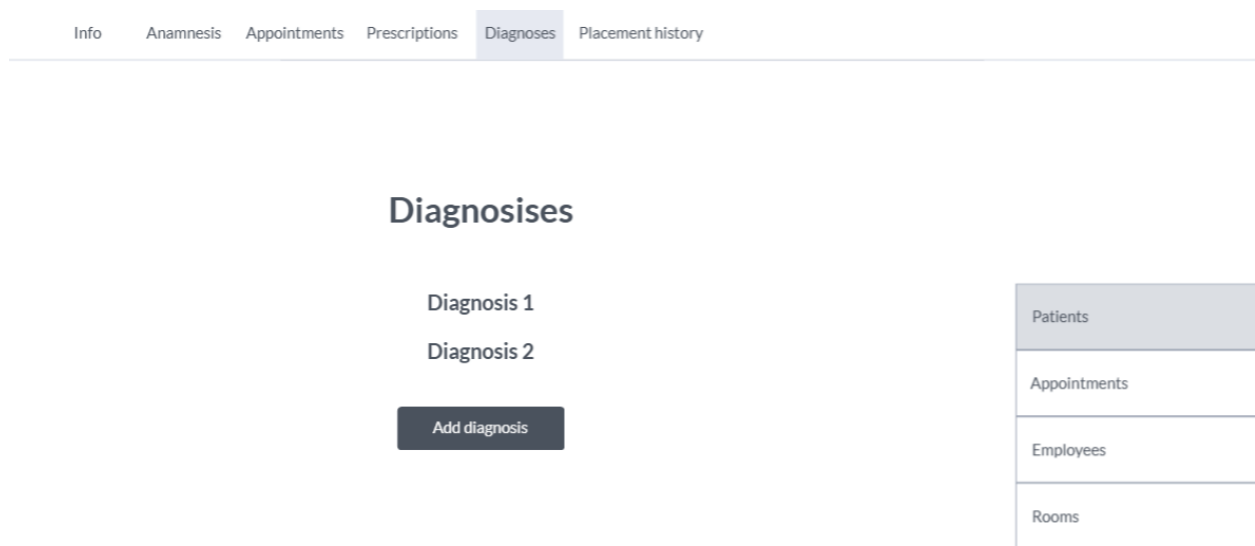
Employees

Rooms

Place patient in a room (Step 5a1)

Referring to the situation when admin changes their mind and pressed “Cancel” button, the system returns to step 2.

- Make a diagnosis



#### Make diagnosis (Step 1)

This is the start of “make a diagnosis” use case. It is located on the Diagnoses tab on patient’s page, where the list of all diagnoses is. The start of the use case is initialized by clicking the add diagnosis button, that is placed under the list of patient’s diagnoses. After that the “Add new diagnosis” tab appears.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

## Add new Diagnosis

Name of disorder:

Date diagnosed:

Date of healing:

Description:

Short description

Form of disease:

Light

Severe

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 2)

The tab for therapist to fill in basic information about the new diagnosis is displayed.

## Add new Diagnosis

Name of disorder:	<input type="text" value="Anexity"/>
Date diagnosed:	<input type="text" value="05/07/2002"/>
Date of healing:	<input type="text" value="-"/>
Description:	<div> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tincidunt congue ligula in rutrum. Morbi nec lacus condimentum, hendrerit mi eu, feugiat.</div>
Form of disease:	<input type="radio"/> Light <input checked="" type="radio"/> Severe
<div>ResetNext</div>	

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 3 and 4)

Therapist fills in all the displayed field with the appropriate information about the patients condition and selects severe form of condition.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

## Add new Diagnosis

Fill out additional fields

Level of danger:

Low

▼

Is restriction required? : ☒ Yes ☐ No

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 5 and 6)

If therapist chose to create severe disease, the following additional field would appear: level of danger field represented by the dropdown list with 3 options (low, medium, high) and requirement of the patient's physical restriction, which is represented by the two options yes or no.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

## Select type of Disease

Type of disease :

Anxiety

▼

Anxiety

Mood

Psychotic

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 7 and 8)

Page with selection of the diseases type is displayed. The selection is made with the dropdown list with all possible options. The ability to add another type for the diseases of the mixed nature will be possible later.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

Input triggers

Triggers :

Loud noises, cars, fire

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 9 and 10)

If therapist selects anxiety as the desired type of disease, the field for the input for things that trigger patient will be displayed.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

Do you want to choose additional type?

Type of disease : ☐ Yes ☒ No

Reset

Submit

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 11,12 and 13)

Page with option to add another nature for the creation of mixed-nature diseases is displayed. Therapist opts not to add additional types for the disease being created and clicks submit button.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

Diagnosis successfully added.

Patients

Appointments

Employees

Rooms

Make diagnosis (Step 14 and 15)

After provided information passes the system's validation, a new diagnosis is added to the system and associated with the patient. A page with the message about the successfully added diagnosis is displayed.



Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

### Add new Diagnosis

Fill out additional fields

Is supervision required? : ☒ Yes ☐ No

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis – Alternative: Create Light diagnosis (Step 4a1 and 4a2)

In case therapist selects light form of disease, the field for need of supervision will be displayed.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

### Input consumed psychedelics

Consumed psychedelics:

Asasa, hddh, hkok

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis – Alternative: Mood nature was chosen (Step 8a1 and 8a2)

In case therapists chose to create a disease of a mood type, the page with fields specific to the mood diseases is displayed. Then the therapist provides the list of psychedelics consumed by the patient in the according field.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

Input hallucinations

Hallucinations:

aliens, cars, people, cats, faces

Reset

Next

Patients

Appointments

Employees

Rooms

Make diagnosis – Alternative: Psychotic nature was chosen (Step 8b1 and 8b2)

A page with the input field for the patient’s hallucinations is displayed for therapist to fill.

Info

Anamnesis

Appointments

Prescriptions

Diagnoses

Placement history

Select type of Disease

Type of disease :

Anxiety

Anxiety

Mood

Psychotic

Reset

Patients

Appointments

Employees

Rooms

Make diagnosis – Alternative: Add another nature for mixed-nature diseases (Step 12a1)

If therapist decides to add another type for the conditions with mixed type, the same page with the selection of the disease type is displayed, and flow of the events returns to step 13.

Info	Anamnesis	Appointments	Prescriptions	Diagnoses	Placement history
------	-----------	--------------	---------------	-----------	-------------------

**Data was entered incorrectly,  
please try again.**

Patients
Appointments
Employees
Rooms

Make diagnosis – Alternative: Data did not pass validation (Step 15a1)

If the data provided by the therapist did not pass the validation, a page with the message about it is displayed and therapist is returned to the start of the creation process to fix errors.

- Assign equipment to a room

## Room N220

### Equipment List:

Assign equipment

Equipment ID	Name	Expiration date	
72982882	Computerised EEG	01.06.2025	<button>Remove</button>
73877382	ECG monitor	01.09.2026	<button>Remove</button>

Rooms

Subordinates

### Assign equipment to a room (Step 1)

The screen above illustrates the page with the table of the equipment currently kept in a chosen by a nurse room from the list of the rooms assigned to the nurse. The page displays the room number for which the information is shown, data about each equipment assigned to it (EquipmentId, Name, ExpirationDate) and a “Remove” button for each. There is also a menu on the right, with “Rooms” highlighted, as a currently opened tab, and there is an “Assign equipment” button on the top of the table. The use case is initialized to the nurse clicking on the “Assign equipment” button which lead to the next page.

Cancel

## Available quipment

Search

Equipment ID	Name
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor

Rooms

Subordinates

Cancel

## Available quipment

Pulse ECT Machin

Equipment ID	Name
82928791	Pulse ECT Machin
73877382	Pulse ECT Machin 2.0
82928739	Pulse ECT Machin

Rooms

Subordinates

### Assign equipment to a room (Step 2)

The following page is displayed after the nurse pressed “Assign equipment” button. It displays to the nurse the list of all the equipment that is available i.e. that is not assigned to any room in the moment. The table displays Id and a Name of each equipment with the search bar above the table and a “Cancel” button. The user can either find an equipment manually using the scroll bar or searched it in the search bar and then press on it.

Cancel

## Available quipment

🔍 Search

Equipment ID	Name
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor

Rooms

Subordinates

Cancel

## Available quipment

🔍 Pulse ECT Machin

Equipment ID	Name
82928791	Pulse ECT Machin
73877382	Pulse ECT Machin 2.0
82928739	Pulse ECT Machin

Rooms

Subordinates

### Assign equipment to a room (Step 3)

The page above illustrates the view when the nurse presses on an equipment from the list of available ones. After clicking on the record in the table, the system checks the equipment expiration date and redirects the user to the next page.

Room to assign the equipment to:

Room N
220
229A
302-2B
306B
313

Rooms

Subordinates

#### Assign equipment to a room (Step 4)

After choosing an equipment, the following screen is displayed to continue the process of assigning a new equipment to the room. It features the same panel on the right, a table with the list of room numbers to which the nurse is currently assigned to and a search bar.

Room to assign the equipment to:

Room N
220
229A
302-2B
306B
313

Rooms

Subordinates

#### Assign equipment to a room (Step 5)

The page above illustrates the view when the nurse presses on a room from the list of the ones they are assigned to. After clicking on the record in the table, the user will be redirected to the next page.

New equipment added successfully.



## Room N220

### Equipment List:

Assign equipment

Equipment ID	Name	Expiration date	
72982882	Computerised EEG	01.06.2025	<button>Remove</button>
73877382	ECG monitor	01.09.2026	<button>Remove</button>
82928791	Pulse ECT Machin	31.03.2025	<button>Remove</button>

Rooms

Subordinates

### Assign equipment to a room (Step 6)

The final stage of the process, after the equipment was successfully assigned to the chosen room the system displays the initial page with a message that says “New equipment added successfully”. The equipment table for the given room was updated and now displays the newly added equipment along with the previous ones.



## Room N220

### Equipment List:

Assign equipment

Equipment ID	Name	Expiration date	
72982882	Computerised EEG	01.06.2025	<button>Remove</button>
73877382	ECG monitor	01.09.2026	<button>Remove</button>

Rooms

Subordinates

### Assign equipment to a room - Alternative (Step 2a1)

In the alternative case where the nurse pressed the “Cancel” button on the previous page, they will be taken back to the initial page with the list of equipment for the chosen room.

Chosen equipment has expired. Please choose another one.



Cancel

### Available quipment

Search

Equipment ID	Name
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor
82928791	Pulse ECT Machin
73877382	ECG monitor

Rooms

Subordinates

### Assign equipment to a room – Alternative (Step 5a1 and 5a2)

The alternative page represents the case where the system check finds out the equipment has expired. The screen shows the same page the user has been on with the list of available equipment along with an appropriate error message at the top of the page saying “Chosen equipment has expired. Please choose another one”.