**User stories**

1. As a manager I want to be able to add an employee to the system based on their specialty so that they can handle their tasks.
2. As a manager I want to search for employees so that I can keep track of them.
3. As a manager I want to be able to remove employees.
4. As a manager I want to be able to edit existing employees.
5. As a secretary I want to store patient personal information so that I can book a medical appointment for them.
6. As a secretary I want to be able to search for patient information so that I can obtain relevant information about a certain patient.
7. As a secretary I want to be able to manage an appointment so that I can organize patient meetings.
8. As a secretary I want to be able to view patients medical records so that their condition can be monitored.
9. As a secretary I want to view patients medicine so that I can send request for renew.
10. As a doctor I want to add a patient’s medical condition so that I can form a medical case.
11. As a doctor I want to prescribe medicine that can be stored in the patient’s personal record so that I can handle the data
12. As a doctor I want to renew medicine prescription that can be stored in the patient’s personal record so that I can satisfy his/her requests.
13. As a doctor I want to approve medicine prescription request that comes from the secretary so that patients can get what they requested.

**Requirements**

**Functional requirements**

1. The system must allow the manager to be able to add new employees based on category. (Done Scenario)
2. The system must allow the manager to be able to remove employees. (Done Scenario)
3. The system must allow the manager to be able to edit existing employees. (Done Scenario)
4. The system must allow the manager to be able to search for certain employees. (Done Scenario)
5. The system must allow the secretary to be able to add new patient’s information. (Done Scenario)
6. The system must allow the secretary to be able to remove patient’s information. (Done Scenario)
7. The system must allow the secretary to be able to edit existing patient’s information. (Done Scenario)
8. The system must allow the secretary to be able to search for patient’s information. (Done Scenario)
9. The system must allow the secretary to be able to add an appointment. (Done Scenario)
10. The system must allow the secretary to be able to remove an appointment. (Done Scenario)
11. The system must allow the secretary to be able to edit an appointment.(Done Scenario)
12. The system must allow the secretary to be able to get a list of appointments. (Done Scenario)
13. The system must allow the secretary to be able to send a request in order to get the medicine renewed by the doctor. (Done Scenario)
14. The system must allow the doctor to be able to add a patient’s medical case. (Done Scenario)
15. The system must allow the doctor to be able to add a medical prescription to the patient’s medical record.(Done Scenario)
16. The system must allow the doctor to be able to renew medicine prescription for the patient. (Done Scenario)
17. The system must allow the doctor to be able to approve the requested medicine that comes from the secretary. (Done Scenario)
18. Add Medicine
19. Remove Medicine
20. Edit Medicine

**Nonfunctional requirements**

1. The system must be developed in java.
2. The usability of the system must be tested by end users.
3. The system must store information in a database.
4. The system must handle multiple requests at the same time.
5. The system must

**Scenarios**

1. **Add employee scenario**
2. Manager selects employee type.
3. Manager provides name and relevant information for an employee.
4. System validates employee data by confirming the saving of the entered information.

**Classes;**

* **System**
* **Manager**
* **Employee**
* **Type**

Methods:

* Select
* Validate
* Save
* Provide
* Confirming

**B. Search employee scenario:**

1. Manager enters employee name in the system.
2. System returns searched employee.

**Classes**

-Manager

- Employee

- System

- Database

Methods

* Search.

**C. Edit employee scenario**

1. Manager selects an employee.
2. Manager updates employee information.
3. System saves the updated information.

**Classes:**

-Manager

- Employee

- System

- Database

Methods;

* Update.

**D. Remove employee scenario**

Manager selects an employee and removes the selected item.

System confirms remove has been successful.

**Classes:**

* Manager
* Employee

- System

- Database

Methods:

* Remove.

**E. Add patient scenario**

1. Secretary adds patient information and the system validates the information.
2. System saves the information.

**Classes:**

* **Patient.**
* **Secretary.**
* **System**
* **Database**

Methods:

* Add
* Validate
* Save

**F. Edit patient scenario**

1. Secretary selects an patient.
2. Secretary updates patient information.
3. System saves the updated information.

**Classes**

* **Secretary**
* **Patient**
* **System**

Methods

* Select
* Save
* Update

**G. Remove patient scenario**

1. Secretary selects a patient and removes patient.
2. System confirms data has been removed.

**Classes:**

* **Secretary**
* **Patient**
* **System**
* **Database**

Methods

* Select
* Remove
* Confirm

**H. Search patient scenario: Oskars**

Secretary enters patient's name in the system.

System returns searched patient.

**Classes**

- Secretary

- Patient

- System

- Database

Methods

* Search.

**I. Remove an appointment scenario: Oskars**

1. The secretary finds patient.
2. The system will return a list of patients appointments.
3. The secretary picks the desired appointment to remove.
4. The system removes appointment.

**Classes**

- Secretary

- Patient

- System

- Database

- Appointment

Methods

* Search.
* Remove
* Save

**I. Get a list of appointments scenario: Oskars**

1. The secretary enters patient name or/ and doctor name or/ and a period of time.
2. The system will return a list of appointments based on the search criteria.

**Classes**

- Secretary

- Patient

- Doctor

- System

- Database

- Appointment

Methods

* Search.

**J. Add a medical case scenario: Oskars**

1. Doctor selects patient.
2. The doctor edit’s medical case
3. Doctor updates medical case.

**Classes**

- Doctor

- Patient

- System

- Database

- Medical case

Methods

* Search.
* Save/Update

K. **Add an appointment (Nadeem)**

1. Secretary search patient.
2. Secretary register reason.
3. Secretary search doctors with available dates.
4. Secretary select doctor.
5. System save the information.

**Classes**

- Secretary

- Patient

- Doctor

- System

- Date

- Database

Methods

* Search
* Register
* Select
* Save

L. **Edit an appointment(Nadeem)**

1. Secretary search patient.
2. Secretary select appointment.
3. Secretary edit appointment
4. System saves the updated information.

**Classes**

- Secretary

- Patient

- System

- Date

- Database

-Appointment

Methods

* Search
* Edit
* Select
* Save

**M. Renew Medicine (Nadeem)**

1. Doctor search patient.
2. Doctor select medicines.
3. Doctor renew medicine.
4. System saves the updated information.

**Classes**

- Doctor

- Patient

-Medicine

- System

- Database

Methods

* Search
* Register
* Select
* Renew
* Save

**N. Request Medicine Renew (Nadeem)**

1. Secretary search patient.
2. Secretary select medicines.
3. Secretary send request.
4. System update information.

**Classes**

- Secretary

- Patient

-Medicine

- System

- Database

Methods

* Search
* Select
* Send
* Update

**O. Approve Medicine Request (Nadeem)**

1. Doctor search patient.
2. Doctor select medicines.
3. Doctor renew medicine.
4. System saves the updated information.

**Classes**

- Doctor

- Patient

-Medicine

- System

- Database

Methods

* Search
* Register
* Select
* Renew
* Save

**P. Add Medical Prescription (Alex)**

1. **Doctor selects the patient.**
2. **System returns patient list.**
3. **Doctor chooses the corresponding medicine and adds it.**
4. **System saves changes.**

**Classes**

* **Doctor**
* **Patient**
* **System**
* **Database**
* **Medicine**

Methods

* Select
* Add
* Return

**Q. Renew medical prescription (Alex)**

1. Doctor searches for patient
2. System returns patients information
3. Doctor selects and edits the existing medical prescription.
4. System saves changes.

**Classes**

* **Doctor**
* **Patient**
* **System**
* **Database**
* **Medicine**

Methods

* Select
* Edit
* Return

**R Approve renew medica prescription (Alex)**

1. **System returns renew medicine requests.**
2. **Doctor selects one request at a time**
3. **System returns a list of medicine for the given patient**
4. **Doctor approves or denies request.**
5. **System saves changes**

**Classes**

* **System**
* **Doctor**
* **Patient**
* **List of medicine**

**Methods**

* **Return**
* **Select**
* **Approve**
* **Deny**