

Object Oriented Software Engineering CA Part 1
March 2023

Miguel Angel Vinas

Student Number: x22116133

INDEX

- 1. Briefly describe your system. Identify and describe the actors.**
- 2. Construct a use case model.**
- 3. Use Case description.**
- 4. Conceptual class diagram.**
- 5. System sequence diagram.**
- 6. Contract**
- 7. Communication diagram.**
- 8. Glossary.**

Name: Miguel Angel Vinas

Student Number: x22116133

Business Domain: A hospital or driving licence renewal

Business Domain Chosen: Hospital

Application: I want to build a system that creates a rota for doctors to know when they are scheduled to work. It will be managed by one person, the Rota administrator. There should be an administrator (actor), a doctor who looks at the software (actor), the database (actor) I think.

1. Briefly describe your system. Identify and describe the actors.

I want to build an application / system that creates a rota schedule for doctors, so any doctor within a given department in one hospital can know when they are scheduled to work, who they are working with, how many hours they are working in any given day and can download the schedule month by month.

For the purpose of this project, we will have the following Primary users / actors:

A) **One Anaesthetic Doctor**. This user will be the primary user of the system, will be accessing the system and will perform different tasks within it.

B) **One Rota Administrator**. This user will be able to create and update the schedules, answer different questions for the Anaesthetic Doctor and manage the whole system.

For the purpose of this project, we will have the following Secondary users / actors:

C) **One Database** where the data is stored and that will be accessed by the server when there is a request to retrieve some data from both, the Anaesthetic Doctor and the Rota Administrator.

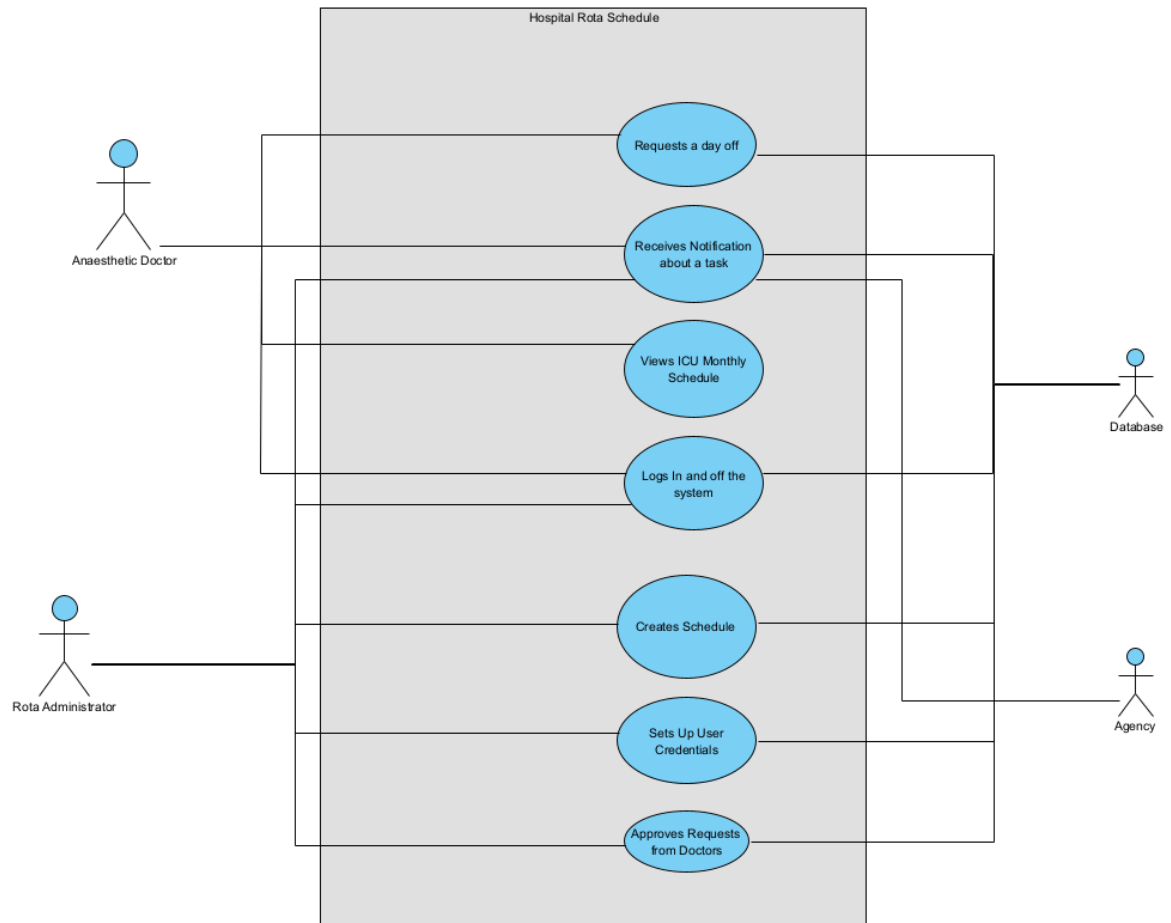
D) **One Agency** that receives a request from the system to hire somebody for the days that the Anaesthetic Doctor is off once the Rota Administrator has approved the time off.

2. Construct a use case model

Primary actors are on the left hand side of the system.

Secondary actors are on the right hand side of the system.

uc [Use Case CA Project – Miguel Angel Vinas]



3. Describe 1 use case. Alternate flow or exceptional flow must be included in the use case. Explain why you chose this use case. Pick the primary use case. Keep it as simple as possible and clear to the reader. Read again about extended use case. Use of the arrows should be clear.

Why did I pick this Use Case: I picked this use case because as the husband of a doctor who works in a really busy ICU in Ireland I am always listening to her telling me that the systems that they use in order to create the rota schedule and request holidays / time off are extremely cumbersome and very non user friendly.

I want to believe that something as simple as "Request Holidays" from a system should be very straight forward and I feel that I could create a system that handles that request easily and with no friction for the user.

Use Case Name: Request Time Off

1. Purpose

The purpose of this use case is for the Anaesthetic Doctor to look at the monthly schedule, request a week off from that monthly schedule and receive confirmation that the request has been approved.

2. Scope

This use case starts when as an Anaesthetic Doctor working in a busy ICU (Intensive Care Unit) department I want to request a week off so that I can go on holidays and relax and this use case ends when I have checked the Request Status and received confirmation that the request has been approved.

3. Actors

In this use case we will have four actors.

A) Anaesthetic Doctor (Primary Actor 1): This actor will be the primary user of the system, will be accessing the system, will look at the ICU monthly schedule, will request a week off from that monthly schedule and will receive confirmation that the request has been approved.

B) Rota Administrator (Primary Actor 2): This actor will be able to create and update the schedules, answer requests from the Anaesthetic Doctor, manage the whole system and set up credentials for different Doctors.

C) Database (Secondary Actor 1): This actor will be a secondary actor, its purpose is to be able to receive, send and hold data and interact with the first two primary actors.

D) Agency (Secondary Actor 2): This actor will be a secondary actor. It will receive a notification from the System to hire an Anaesthetic Doctor to cover the days off once the Rota Administrator has approved the request for Time Off and has approved the hire of a substitute.

4: Description

In this use case the Anaesthetic Doctor will be requesting a week off from the monthly schedule so she can go on holidays.

The Rota Administrator manages the schedule and answers to the requests of the Anaesthetic Doctor.

The Database is the place where the data for the monthly schedule and the request is stored.

The Agency will receive a notification to hire an Anaesthetic Doctor to cover the time off if the Rota Administrator approves the hire.

5: Flow Description

5A) Pre – Conditions:

The system must be working and functional. The Anaesthetic Doctor has to have a set of valid credentials in order to log in on the system, look at the ICU monthly schedule, request a week off and check on the request status.

The Rota Administrator has to have a set of valid credentials in order to log in on the system, access the current ICU monthly schedule and receive and respond the request from the Anaesthetic Doctor.

The ICU Monthly Schedule must be active and updated.

5B) Activation

The use case starts when the Anaesthetic Doctor logs in on the system with a valid set of credentials.

5C) Main Flow

1. The system presents the Anaesthetic Doctor with a main page where the Anaesthetic Doctor can click on "Request Time Off".

2. Anaesthetic Doctor clicks on "Request Time Off".
3. System shows the ICU monthly schedule together with a way to select the days she wants off from that month.
4. Anaesthetic Doctor chooses the time off from the ICU monthly schedule.
5. Anaesthetic Doctor sends the request for time off.
6. System shows confirmation to the Anaesthetic Doctor that the request has been submitted and the status of it.
7. System sends a notification to the Rota Administrator about the request for time off.
8. Rota Administrator receives the notification in the system regarding the request.
9. Rota Administrator clicks on the request to review it.
10. System opens a page showing the request and the ICU Monthly Schedule.
11. Rota Administrator reviews the request, checks the ICU Monthly Schedule and makes sure that the request can be confirmed.
12. Rota Administrator approves the request.
13. System sends an Approval notification to the Anaesthetic Doctor.
14. System asks Rota Administrator if a substitute Anaesthetic Doctor needs to be hired.
15. Rota Administrator denies the request to hire a substitute.
16. System shows confirmation of the action taken to the Rota Administrator.
16. Anaesthetic Doctor receives notification and checks task status.

5D) Alternative Flow

- 15A) Rota Administrator approves the request to hire a substitute.
- 15B) System sends a notification to Agency with information about the rank, qualification, dates and rates for the hire of a substitute doctor.
- 15C) System shows confirmation of the action taken to the Rota Administrator.
- 15C) Agency confirms the request and hires a substitute doctor.

5E) Exceptional Flow

- 4A) Anaesthetic doctor chooses days that are not available in the ICU Monthly Schedule.
- 4B) System shows a Special Request page.
- 4C) Anaesthetic Doctor confirms the submission for the Special Request.

5F) Termination

The use case finishes when the Anaesthetic Doctor gets a confirmation from the system that the request for her holidays has been approved.

5G) Post – Conditions

The System sends an email to Anaesthetic Doctor with the times and dates she has off.

6: Views

Anaesthetic Doctor views credentials page, main page, Request Time Off page, ICU Monthly Schedule page and Status of the Request page.

Rota Administrator views credentials page, main page, ICU Monthly Schedule page, Requests page, Status of the Requests page.

Agency views Requests page and Hire Substitute Doctor page.

7. Special Requirements

The system must be able to have multiple users logged in at the same time, the system must be

accessible for users with disabilities, the system could have a timed auto log off feature. The system must provide feedback to the user at every step of the process.

8. Interfaces

The system's interface must be created with the user in mind, it must be easy to use, clear and have as less pain points and friction as possible.

The Anaesthetic Doctor will need an interface to log in on the system, another interface as a main page, another interface to look at the ICU Monthly Schedule and another interface to check the Request Status.

The Rota Administrator needs an interface to log in on the system, another interface to allow for looking and replying at requests and another interface to check the ICU Monthly Schedule.

The Agency needs an interface to allow for looking and replying at requests.

All interfaces must be accessible for users with disabilities.

9: System Characteristics / Performance

System must be efficient, with no bugs, easy to use, with up to date coding, performance and security characteristics.

When transferring data from System to Agency the data needs to be encrypted on both ends with a strong encrypted key that needs to be shared securely.

10. Implementation Requirements

The System must have a database to store the data, needs to be stored in a secure server.

11. Technical Specifications

The system must be developed in a language that allows for easy changes and implementations from user feedback.

It needs to handle a large dataset, needs to be a stand alone application working with PC / Apple Computer users, needs to have a lightweight web application and a mobile application compatible with Android and iOS.

4. Create a conceptual class diagram of the chosen use case. The conceptual class diagram should demonstrate at least three of the following: (1) attributes, (2) relationships, (3) navigability, (4) association, (5) multiplicity, and (6) composition. And then explain the reasons as to why you have chosen the different relations.

The reason as to why I have chosen the different relations are as follow:

- <<Doctor>> is a parent and abstract class and Anaesthetic Doctor and Substitute Doctor inherit the attributes of the <<Doctor>> class as they are children of the <<Doctor>> class.

<<Doctor>> has a method that is private so no children can access to it.

There can be 1 to many doctors in the System.

There can be 0 to many Anaesthetic Doctors in the System.

There can be only from 0 to 2 Substitute Doctors at any given time.

- A Substitute Doctor has a relation with Agency but if there is no Substitute Doctor in the system the Agency still exists.

There is 1 agency that can have 0 to 2 Substitute doctors as there can be 0 to 2 Substitute Doctors in the System at any given time.

- 1 Anaesthetic Doctor is associated with Requests for Time Off because the Anaesthetic Doctor can submit 1 to many requests. The reason being for 1 to many is that if the Anaesthetic Doctor submits 0 requests for time off then there is no request for time off.

- The Anaesthetic Doctor can check 1 to 12 ICU Monthly Schedule so it is associated with it.

- <<Administrator>> is a parent and abstract class and have a child class called Rota Administrator that inherits all the attributes and methods but one.

There can be 1 to many Administrators in the System.

- <<Administrator>> is associated with <<Schedule>> because it creates the <<Schedule>> for the System.

- Rota Administrator is associated with the Substitute Doctor because it hires it.

It is also associated with Agency since the Rota Administrator has to approve the hire of the Substitute Doctor.

1 Rota Administrator creates 1 ICU Schedule.

1 Rota Administrator checks the submitted Request for Time Off from either the <<Doctor>> or the Anaesthetic Doctor.

- <<Schedule>> is a parent and abstract class and has one child called ICU Schedule and ICU Schedule has a sub child called ICU Monthly Schedule.

<<Schedule>> is associated with and accesses the Database where the data is stored and backed up. ICU Monthly Schedule has a composition relationship with ICU Schedule as it can't exist without ICU Schedule.

ICU Monthly Schedule has a composition relationship with <<Schedule>> as it can't exist without the parent class.

There can be 1 to many <<Schedules>>, there is only 1 ICU Schedule in the System and there is 1 to 12 ICU Monthly Schedule in the System as there are 12 months in the year.


```

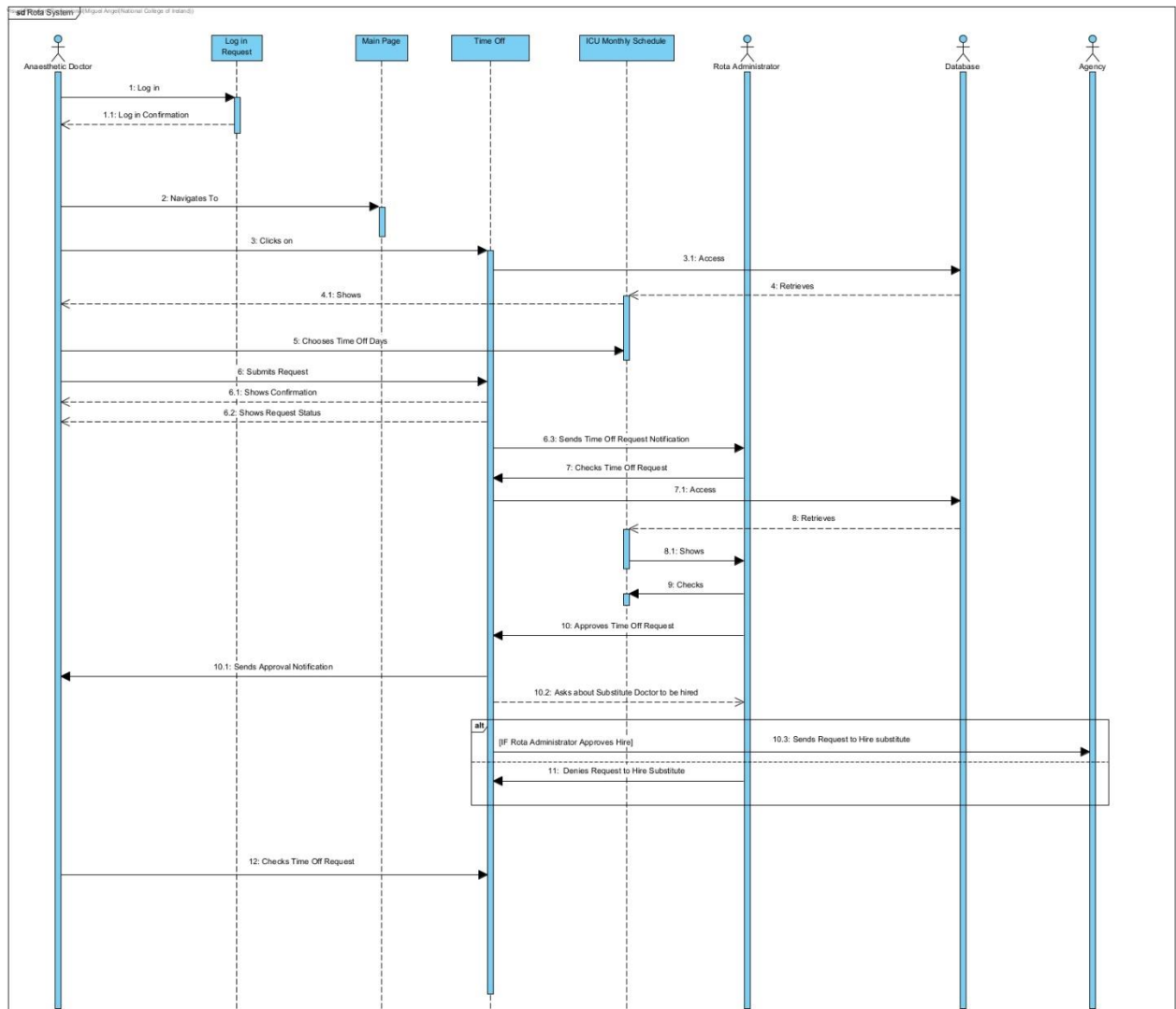
classDiagram
    class Doctor {
        +name: string
        +id: string
        +age: int
        +manager: string
        +address: string
        +generalMedicalPractitionerNumber: int
        +email: string
        +phoneNumber: int
        +dodded: int
        +hasAccessTheticDoctor()
        +verifyLogin() boolean
        +requestsTimeOff()
    }
    class AnaestheticDoctor {
        +generalMedicalPractitionerNumber: int
        +accessCUMonthlySchedule()
    }
    class SubstituteDoctor {
        +generalMedicalPractitionerNumber: int
        +agency: string
    }
    class RotaAdministrator {
        +checksRequestTimeOff()
        +approvesTimeOff()
        +savesTimeOff()
        +accessCUMonthlySchedule()
        +increasesCUMonthlySchedule()
        +modifiesCUMonthlySchedule()
        +approvesHireSubstituteDoctor()
        +increasesHireSubstituteDoctor()
        +accessCUSchedule()
        +modifiesCUSchedule()
        +increasesCUSchedule()
    }
    class Agency {
        +hiresSubstituteDoctor()
        +paysSubstituteDoctor()
    }
    class Schedule {
        +month: string
        +day: int
        +year: int
        +hour: int
        +minutes: int
        +department
    }
    class ICUSchedule {
        +populatesCUMonthlySchedule()
    }
    class RequestForTimeOff {
        +month: string
        +day: int
        +year: int
        +hour: int
        +minutes: int
        +department
        +dodded: int
        +int requested
        +showsConfirmation()
        +showsTaskStatus()
        +createsRequest()
    }
    class Database {
        +storesSchedule()
        +schedulesSchedule()
    }

    Doctor "1..*" -- "1" AnaestheticDoctor
    Doctor "1..*" -- "1" SubstituteDoctor
    Doctor "1..*" -- "1..*" RotaAdministrator
    Doctor "1..*" -- "1..*" Agency
    Doctor "1..*" -- "1..*" Schedule
    Doctor "1..*" -- "1..*" ICUSchedule
    Doctor "1..*" -- "1..*" RequestForTimeOff
    AnaestheticDoctor "1" -- "1" SubstituteDoctor
    SubstituteDoctor "0..2" -- "0..2" RotaAdministrator
    RotaAdministrator "1" -- "1" Agency
    RotaAdministrator "1" -- "1" Schedule
    RotaAdministrator "1" -- "1" ICUSchedule
    RotaAdministrator "1" -- "1..12" RequestForTimeOff
    Agency "1" -- "1..12" RequestForTimeOff
    Schedule "1..*" -- "1..2" ICUSchedule
    Database "1" -- "2" Schedule
  
```

The Rota Administrator denies the request from the System. However, if the Rota Administrator had approved the request from the System, the System would have sent a "Request to Hire Substitute" to the Agency, with the needed details for the Agency to find a substitute Anaesthetic Doctor.

The use cases finishes when the Anaesthetic Doctor checks the time off Request and sees that has been approved.

I placed the Rota Administrator on the right side of the screen because I feel that visually it is less cluttered. I could have placed it on the left side of the screen since it is a Primary Actor.



System Sequence Diagram Number 2

The use case starts when the Anaesthetic Doctor logs in with valid credentials and as the system validates the login the Anaesthetic Doctor navigates to a main page.

The Anaesthetic Doctor "Requests Time Off" and the system accesses the Database in order to retrieve the ICU Monthly Schedule needed for the request.

The System shows the ICU Monthly Schedule to the Anaesthetic Doctor and the Anaesthetic Doctor chooses the days off and submits the request for time off.

The System shows the Anaesthetic Doctor with a confirmation of the task done and shows the Request Status.

The System sends a "Time Off Request Notification" to the Rota Administrator so the Rota Administrator can review it and respond.

The Rota Administrator then checks the "Time Off Request" from the Anaesthetic Doctor, the System accesses the Database and retrieves the ICU Monthly Schedule and shows both, the Time Off Request and the ICU Monthly Schedule to the Rota Administrator.

The Rota administrator checks the ICU Monthly Schedule, sees that everything is ok and "Approves" the "Request for Time Off".

As the Rota Administrator approves the request for time off, the System sends an approval notification to the Anaesthetic Doctor regarding the Request for time off.

At the same time, the System asks the Rota Administrator if he / she wants to hire a substitute Doctor to cover the days off.

The Rota Administrator denies the request from the System and the system confirms the action taken (Not to hire a substitute doctor in this particular case). However, if the Rota Administrator had approved the request from the System, the System would have sent a "Request to Hire Substitute" to the Agency, with the needed details for the Agency to find a substitute Anaesthetic Doctor, the System would have shown a confirmation of the action taken to the Rota Administrator and the Agency would have sent a confirmation of the Request being acknowledged.

The use cases finishes when the Anaesthetic Doctor checks the time off Request and sees that has been approved.

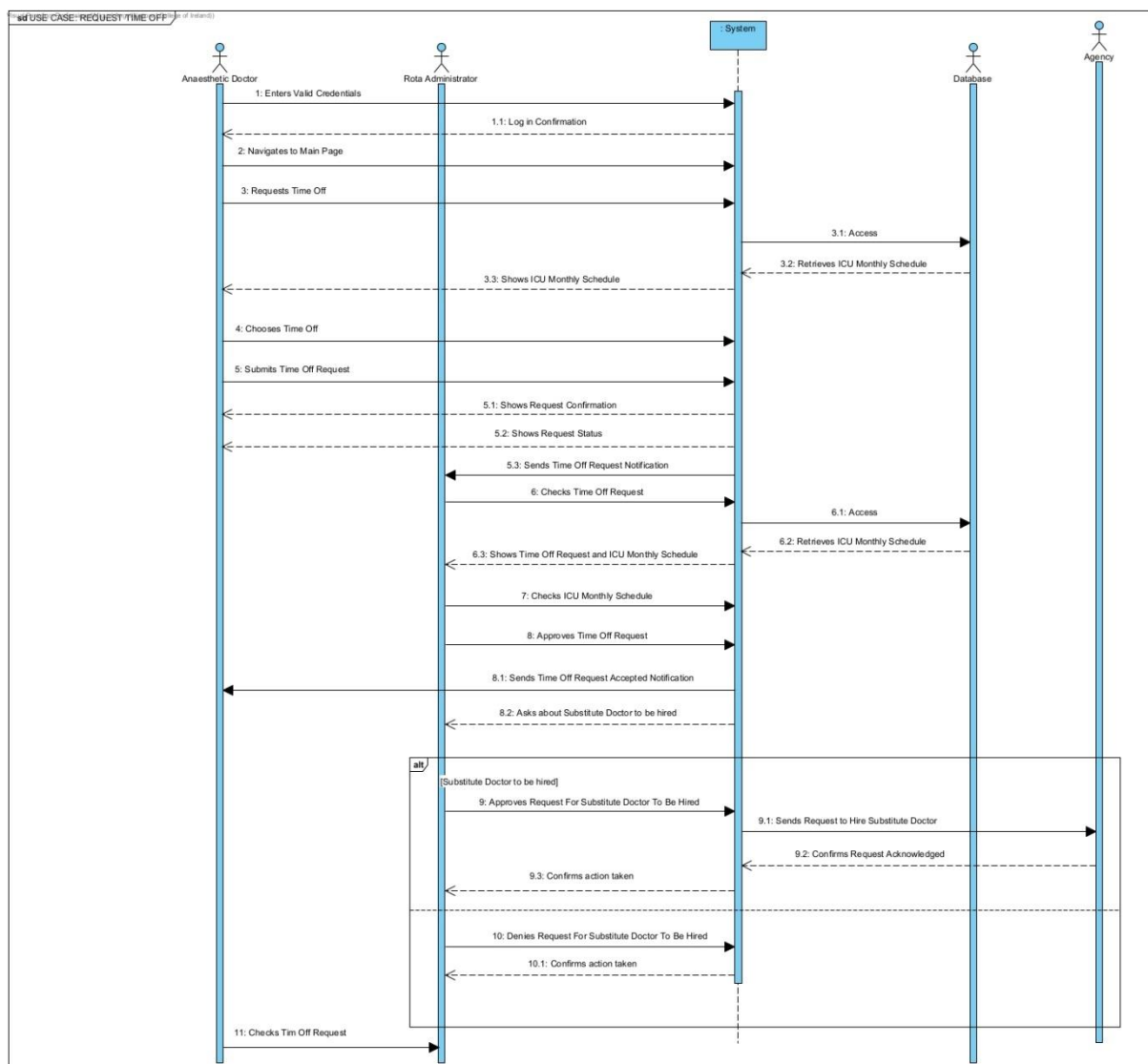
I found this System Sequence Diagram where I used the System as a Black Box easier to draw, easier to conceptualize and easier to read than the **System Sequence Diagram Number 1**.

It allowed me to think about the actions that the Actors and the System perform without inferring a design in them, as an example in the System Sequence Diagram 1 I have a "Clicks on", however, using the System as a black box made me just create actions like "Requests Time Off", which are abstract and noncommittal.

It also helped me realize that the System wasn't being proactively giving feedback to one of the actors, the Rota Administrator, when performing some tasks. The Agency wasn't giving feedback either when a message was sent to it so I updated that in the System Sequence Diagram 2 and also in the Use Case!.

I didn't update it in the System Sequence Diagram 1 so you can take a look at the iteration and how using the System as a black box has helped.

In this case I placed the Rota Administrator on the left hand side because it was visually easier to follow and also because it is a Primary Actor.



6. Develop one contract for the distinct system operations in the sequence diagram.

I have chosen to develop a contract in the operation "Choose Time Off"

Operation:

- chooseTimeOff (doctorID: String, selectStartDate: StartDate, selectEndDate: EndDate)

Responsibilities:

- Choose the start date and the end date of the time off requested, associate it with the doctorID and add it to the RequestTimeOff

Cross References:

- Use Cases: Request Time Off

Preconditions:

- Anaesthetic Doctor must be logged in and active.
- Anaesthetic Doctor must have requested time off.
- ICU Monthly Schedule must be up to date.
- System must be working.

Postconditions:

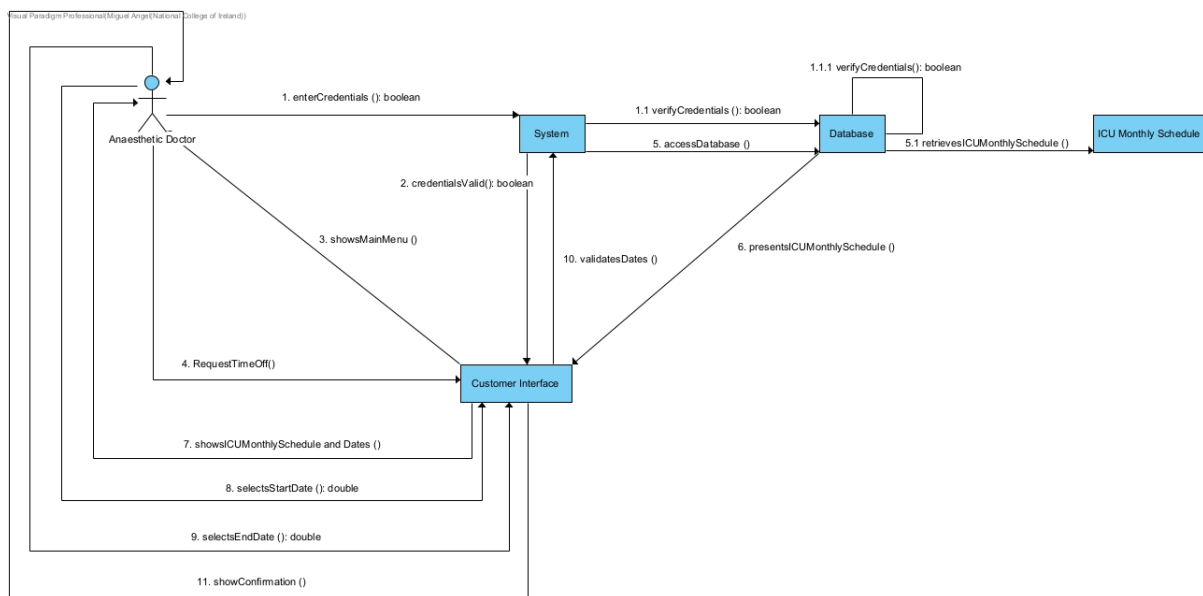
- A timeOffRequested(tor) was created.
- tor was associated with the current Request Time Off.
- tor.startdate was set to selectStartDate.
- tor.enddate was set to selectEndDate.
- tor was associated with the current RequestTimeOff based on a doctorID match.

7. Draw a communication diagram based on the above contract. The communication diagram should clearly demonstrate the use of named design patterns.

Task: Choose Time Off

- Step 1: Anaesthetic Doctor logs in on the system.
- Step 2: System validates the credentials.
- Step 3: Anaesthetic Doctor Requests Time Off.
- Step 4: System access Database.
- Step 5: Database retrieves ICU Monthly Schedule.
- Step 6: Database shows ICU Monthly Schedule to Anaesthetic Doctor.
- Step 7: Anaesthetic Doctor chooses start date and end date.
- Step 8: Anaesthetic Doctor submits request.
- Step 9: System shows confirmation of task.
- Step 10: System shows Request Status.

I had a problem with the associations, I couldn't place the arrows for the messages on top so I decided to just create normal associations pointing towards the direction of the message in the System.



8. Create a glossary with more than four items that lists and defines all the project related terms that require clarification.

Glossary

Dataset: A collection of data. A file that contains one or more records.

Record: Basic unit information used by a program. Any named group of records is called a data set.

Lightweight: In computing, a computer program that is designed to have a small memory footprint and low CPU usage, overall a low usage of system resources.

Encrypted: The process that transforms data to obscure its meaning, in other words, turn the data into something that appears random and nonsensical.

Encrypted Key: A value that is applied to the data to produce encrypted data or to decrypt the data in a receiving message.

Android: A mobile operating system designed primarily for touchscreen mobile devices such as smartphones and tablets.

iOS: A mobile operating system designed for use with Apple's multitouch devices.

PC: A personal computer designed for individual use.

Apple Computer: A personal computer made by Apple designed for individual use that uses Apple's computer operating system.

Accessible: A measure of how useable a computer system is to all people, including those with disabilities or impairments. It concerns both, software and hardware and how they are configured in order to enable a disabled or impaired person to use that computer system successfully.

Interface: A program that allows a user to interact with a computer system in person or over a network.

Pain Points: Problems that occur at the different levels of the customer experience with a computer system.