



MedTech
Mediterranean
Institute of Technology

Junior Project

-Software Engineering-

Author(s):

Najla SEGHAIER

Mariem BOUSAADIA

Jihene CHOUIREF

Mohamed Amine MHIRI

Mohamed Aziz ROUROU

Mohamed Aziz TRABELSI

Tunisian Online Medical Assistance (TOMA)

Supervisor:

Dr. Asma AMDOUNI

Academic Year

2020 - 2021

Declaration of Authorship

This report has been prepared on the basis of our own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Signed:

Date:

"Developments in medical technology have long been confined to procedural or pharmaceutical advances, while neglecting a most basic and essential component of medicine: patient information management."

John Doolittle

MEDTECH-SMU

Abstract

Software Engineering

Junior Project

Tunisian Online Medical Assistance (TOMA)

by Mariem BOUSSAADIA

Jihene CHOUIREF

Mohamed Amine MHIRI

Mohamed Aziz ROUROU

Najla SEGHAIER

Mohamed Aziz TRABELSI

This report describes the conception and development of a medical assistance platform to connect patients and physicians. The present work is part of the ISS 396 project course. This project has a goal to create a website with the ability to advise patients, manage appointments, and ease teleconsultations. To realize this work, "Angular" was used for the front-end, "Node.js" and "Express" for the back-end and "MongoDB" for the database. We have used the SCRUM framework to manage the project during the working cycle.

Keywords: *Medical Assistance, MongoDB, Express, Angular, Node.js, SCRUM*

Acknowledgements

We are using this opportunity to express our gratitude to Dr. Asma Amdouni and all those who provided us with their help for their invaluable constructive criticism and advice during the project work.

We are sincerely grateful to them for sharing their knowledge views on a number of issues related to the project.

Contents

Declaration of Authorship	iii
Abstract	vii
Acknowledgements	ix
General Introduction	xvi
1 Preliminary Study	1
1.1 Introduction.....	1
1.2 Project Presentation.....	1
1.2.1 Project Context	1
1.2.2 Interest of Application	1
1.3 Analysis of the Existing	1
1.4 Proposed Solution	3
1.5 Objectives	3
1.6 Conclusion	3
2 Requirement Specification	5
2.1 Introduction.....	5
2.2 Modelling Language and Working Method.....	5
2.2.1 Modeling Group.....	5
2.2.2 Project Management	5
2.2.3 Development Methodologies.....	6
2.2.4 Comparative Study Between Methods.....	6
2.2.5 Development Methodologies.....	6
2.2.6 Scrum Framework.....	8
2.2.7 Roles and Responsibilities	8
2.2.8 Scrum Artifacts.....	9
2.2.9 Scrum events.....	9
2.3 Technological Study.....	10
2.3.1 Hardware Architecture	10
2.3.2 Software Environment.....	11
2.3.3 Architecture Environment	13
2.4 Requirement Analysis	14
2.4.1 Functional Requirements.....	14
2.4.2 Non-functional Requirements.....	15
2.5 Identifying Actors	15
2.6 Product Backlog.....	15
2.7 General Use Case Diagram.....	16
2.8 Conclusion	16
2.9 References	16
3 Sprint 1	19

3.1	Introduction	19
3.2	Organization.....	19
3.3	Sprint Backlog.....	19
3.4	Analysis	20
3.4.1	Use Case Diagram.....	20
3.4.2	Textual Description.....	20
3.5	Design	22
3.5.1	Sequence Diagram	22
3.5.2	Class Diagram	22
3.6	Implementation.....	23
3.7	Conclusion	24
4	Sprint 2	31
4.1	Introduction	31
4.2	Organization.....	31
4.3	Sprint Backlog.....	31
4.4	Analysis	31
4.4.1	Use Case Diagram.....	31
4.4.2	Textual Description.....	31
4.5	Design	34
4.5.1	Sequence Diagram	34
5	Sprint 3	37
5.1	Introduction	37
5.2	Organization.....	37
5.3	Sprint Backlog.....	38
5.4	Analysis	38
5.4.1	Use Case Diagram.....	38
5.4.2	Textual Description.....	39
5.5	Design	40
5.5.1	Sequence Diagram	40
5.5.2	Class Diagram	40
5.6	Conclusion	40
	Bibliography	43

List of Figures

2.1	Project Division using Scrum	8
2.2	Scrum Project Sprints.....	8
2.3	Scrum Roles	9
2.4	User Story Construction.....	9
2.5	Scrum Artifacts	10
2.6	First Laptop used for Development and Testing.....	10
2.7	Second Laptop used for Development only	11
2.8	Visual Studio Code Logo	12
2.9	MySQL Logo	12
2.10	Node.js Logo	12
2.11	Angular Logo	13
2.12	Bootstrap Logo.....	13
2.13	Postman Logo	14
2.14	Client-side Application.....	14
2.15	Genral Use Cas Diagram.....	18
3.1	Diagram showing the Organization of the First Sprint.....	19
3.2	Use Case Diagram of the First Sprint.....	20
3.3	Sequence Diagram showing how a visitor signs in and then logs in as a user.....	22
3.4	Sequence Diagram showing how a doctor accesses the patients list.....	23
3.5	Class Diagram of the First Sprint	24
3.6	Create Account Step 1	25
3.7	Create Account Step 2	25
3.8	Sign in.....	26
3.9	Filter All Doctors	26
3.10	Filter My Doctors.....	27
3.11	Schedule Appointment Request.....	27
3.12	Update Profile Information	28
3.13	Doctor's Request Notification	28
3.14	Doctor's Appointment List.....	29
3.15	Doctor's Patient List.....	29
3.16	Doctor's Archived Patient List	30
3.17	Patient Requests Status.....	30
3.18	Patient Appointment List.....	30
4.1	Diagram showing the Organization of the Second Sprint.....	31
4.2	Use Case Diagram of the Second Sprint	33
4.3	Sequence Diagram showing how a patient schedules an appointment .	34
4.4	Sequence Diagram showing how a doctor validates the appointment .	35
5.1	Diagram showing the Organization of the Third Sprint.....	37
5.2	Use Case Diagram of the Third Sprint	38

5.3	Sequence Diagram showing how a patient rates a doctor.....	41
5.4	Class Diagram of the Third Sprint.....	41

List of Tables

1.1	Comparative Existing Solutions Feature Study	2
2.1	Comparative Study of Methods	7
2.2	The Primary Actors	16
2.3	The Secondary Actors	16
2.4	The Product Backlog	17
3.1	Sprint 1 Backlog showing the tasks, the complexity and the estimation of the duration of each task.....	20
4.1	Sprint 2 Backlog showing the tasks, the complexity and the estimation of the duration of each task.....	32
5.1	Sprint 3 Backlog showing the tasks, the complexity and the estimation of the duration of each task.....	38

General Introduction

Nowadays, every citizen carries a critical responsibility in helping to contain the spread of the Covid-19 virus. Therefore, everyone is trying their best to limit close contact with others especially in closed or crowded spaces. This issue is even more pressing when it comes to visiting the doctor's office, where the risk of exposure is much higher.

Besides, even before the spread of the virus, taking an appointment could be complicated, time-consuming, and almost a struggle due to the many constraints that are out of the patients' control. This is because in Tunisia we are still relying on the traditional ways to book appointments, which are mainly, via phone calls during the working hours only, or going to the doctor's office in person. In this context, the medical field in Tunisia needs to catch up with the modern age of digitalization especially under the current circumstances caused by the Covid-19 pandemic. Patients need to have the possibility to manage their own medical follow up schedule and to consult doctors of their choosing online whenever possible.

Moreover, it is very frequent for patients to visit their doctor's office just to ask simple follow-up questions or deliver required documents such as scans and previous medical reports hence we need to think of a more adequate way of interaction between both parties and this is where digitalization comes in handy.

It is in that spirit that we came up with our project proposal Tunisian Online Medical Assistance in an attempt to limit risks, wasted time and energy, and more importantly, to create a bridge of guidance between patients and their doctors.

This report is a synthesis of all the work we have done in this perspective. It begins with the first chapter, Preliminary Study, which deals with the presentation of the project, the analysis of the existing similar platforms and their flaws as well as the solution for them and our project's main objectives. Subsequently, the second chapter, Requirements Specification, will be devoted to explain the working methods, compare them and explain why we choose our method for this work. In the rest of this chapter, we will go through the main concepts in this field and display diagrams of the general use cases for our web application and the identification of the actors of the platform. The final chapters include the composition of each sprint and their implementation.

Chapter 1

Preliminary Study

1.1 Introduction

In this chapter, we set the the project in its context. At first, we start by presenting the project purpose, then we show the existing similar project and their flaws. Finally, we present the proposed solutions for these problems and the main objectives of our project.

1.2 Project Presentation

1.2.1 Project Context

Our project consists of designing and implementing a user-friendly platform for digitalizing the interactions between doctors and their patients. The platform is a web-based application that enables the doctors to connect to their patients and take advantage of several services such as:

- Appointment Management
- Chat
- Online consultations

This work is part of the Junior project at the Mediterranean Institute of Technology.

1.2.2 Interest of Application

One of the interests of our solution is to help the medical field in Tunisia catch up with the modern age of digitalization especially under the current circumstances caused by the Covid-19 pandemic.

Another important aspect is the ease of use and gain in time for the patients by giving them the possibility of managing their own medical follow up schedule, thus simplifying the assistants' daily tasks. From the doctor's point of view, this solution will provide them with a larger market that can cover the whole country instead of their local city.

1.3 Analysis of the Existing

In this section of the report, we analyze existing platforms that are similar to our project in order to acquire a thorough knowledge of the prevailing systems in use. The analysis of the existing allows us to study their assets and shortcomings, to prevent committing common mistakes and to determine the aspects we need to include

TABLE 1.1: Comparative Existing Solutions Feature Study.

Features	Med.tn	Telemedecine.tn
Medical Directories	+	+
Filters	+	+
Scheduling Appointments	+	+
Profile Descriptions	+	+
Tags	+	-
Static Working Hours	+	+
Online Consultations	-	+
Responsive Working Hours	-	-
Forum	+	+
Videos or Articles	+	+
Mandatory Registration	-	+
Online Payment	-	+
Doctor Recommendations	-	-
Chatbot	-	-

in our project and the possible improvements we can introduce to our ideas. Table 1.1 summarizes the following comparison and discussion.

In the available Tunisian online medical platforms such as “med.tn”, the user can find different medical directories such as pharmacies, laboratories and medications. They can look for a doctor or a pharmacy by specifying the name, the specialty, the location. They can filter the physicians by gender and spoken languages or go manually through the entire medical directory. Each physician has a profile that generally includes a biography highlighting their experience or keywords that will help the user’s query alongside their working schedule. They can also post informative videos and articles. The patient can take an appointment online by choosing a date and time as they provide their name and mobile number. If the chosen slot is available, a code is sent to them to validate their appointment.

Aside the diversity of the proposed services that may confuse the user. Such a system displays mainly two disadvantages. On the one hand, the dilemma a person might face when searching for a physician as they are likely unaware of their medical issue. Each individual should receive guidance to help them narrow down their choices of specialists. On the other hand, the randomness of the appointments, the patient cannot truly see the availability of their doctor who in return cannot provide their patients with an ideal meeting time. It is important to grant both parties an efficient and updated scheduling.

The other most important medical website in Tunisia is “telemedicine.tn” which is a virtual clinic that enables the patients and the doctors to meet remotely. This constitutes an interesting aspect particularly under health crisis circumstances (covid 19+) which prevent people to move freely.

Nonetheless, the mandatory registration for simply asking a question or looking for a doctor to take an appointment is not necessary. The patient has the right to access to the information about the involved doctors before deciding to join the platform and supply their personal data.

Forums can be found on both Tunisian medical websites, where patients can interact with their doctors or among themselves. However, many questions are left unread and unanswered. The downside of this feature is the absence of an entity

that quickly assists all the users.

1.4 Proposed Solution

To remedy the problems mentioned above, the team will aim to use optimized tools and adapt to facilitate the user's mission and providing rich and useful features. Among those we have:

- **Online Appointment Scheduling and Management of Several Timetables.**
The proposition consists in creating some luxuries such as the possibility of taking part of the doctor assistant's role.
- **Management of Several Patients and Doctors.**
The proposition consists in creating patient lists for the doctors to store their medical information and providing a doctors dashboard for the patients according to their recommendations.
- **Live chat.**
The proposition consists in providing an instantaneous textual interaction between the patient and the doctor.
- **Online Consultation.**
The proposition consists in a web-based application able to serve users -mainly patients- when in need with the help of medical knowledge implemented in partnership with medical doctors.
- **Online Payment.**
The proposition is valid in case the meeting is online , the user will be directed to the online payment page where he/she could perform the transaction safely.

1.5 Objectives

The ultimate objective of the project is to simulate a whole patient-Physician interaction offering a platform able to chat with the user, giving him the possibility to schedule his appointments online and eliminating the burden of even leaving his house for a consultation.

All in all, we can come out with those main points that our application aims to satisfy:

- Optimize schedules management and planning of appointments taking into account various constraints (simultaneous appointments, alternation of physical and virtual consultation, etc.)
- Provide an easy to use application able to offer the patient more control on their health situation.

1.6 Conclusion

In this chapter, we put the attention on how and why our platform should enhance the patient-doctor interaction and teleconsultations, which will be primarily thanks to a better way of taking appointments and even choosing which doctor to see via a recommendation system.

Chapter 2

Requirement Specification

2.1 Introduction

In this chapter, we set the requirements of our customers for this project. At first, we start by presenting the project's modelling language and working method, then we define the actors involved in this project. Finally, we will present the product backlog and the general use case diagram of our project.

2.2 Modelling Language and Working Method

2.2.1 Modeling Group

The Unified Modeling Language (UML) is a modeling language that we can use to represent the needs of the project. UML (Unified Modeling Language) summarizes and visualizes object-oriented programming systems. The modeling language is therefore a practical tool for developers. On the one hand, it allows you to create clear plans for software projects. On the other hand, it also makes it possible to present complex software systems in a way that is simple and understandable even to people outside the field.

2.2.2 Project Management

The development of software for an improved business process, the construction of a building or bridge, the relief effort after a natural disaster, the expansion of sales into a new geographic market — all are projects.

And all must be expertly managed to deliver the on-time, on-budget results, learning and integration that organizations need.

Project management, then, is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements.

Project management processes fall into five groups:

- Initiating
- Planning
- Executing
- Monitoring and Controlling
- Closing

2.2.3 Development Methodologies

The development methodology chosen by the team, and applied to developing the product, has an impact on the goals of the project, especially when changes to the methodology occur during or between product releases. Methodologies most often change in response to a project failure, inability to quickly respond to new requirements, quality issues with past releases, or frustration with the existing development workflow by the engineering team.

- Agile development methodology:

Teams use the agile development methodology to minimize risk (such as bugs, cost overruns, and changing requirements) when adding new functionality. In all agile methods, teams develop the software in iterations that contain mini-increments of the new functionality.

Pros: The primary benefit of agile software development is that it allows software to be released in iterations. Iterative releases improve efficiency by allowing teams to find and fix defects and align expectation early on. They also allow users to realize software benefits earlier.

Cons: Agile development methods rely on real-time communication, so new users often lack the documentation they need to get up to speed. They require a huge time commitment from users and are labor intensive because developers must fully complete each feature within each iteration for user approval.

- Unified Software Development Software:

The Unified Software Development Process or Unified Process (UP) is an iterative and incremental software development process framework. We can apply this process on different type of software systems. It introduces a new standard for creating today's software that will certainly be useful for any software developer or manager who is acquainted with UML.

2.2.4 Comparative Study Between Methods

We have to verify that our methodologies suit our project and meet our needs . In order, to come up with the best solution to our customer's need and to have a very quick deliverable.

To succeed doing that we have to go through all the methodologies. In fact , a comparative study between the methodologies such as shown in Table 2.1 is mandatory in order to choose the best methodology that suits our project .

2.2.5 Development Methodologies

Each one of the above mentioned methodologies focuses on a well-defined phase of the project but the best methodology that suits our conditions and our project is 'Scrum' for these reasons:

- The task progress is made for short development time.
- The "scrum" will fit better a reduced and limited number team.

TABLE 2.1: Comparative Study of Methods

Method and Description	Advantages	Disadvantages
DSDM(dynamic system development method) is an Agile method that focuses on the full project life-cycle. One of a number of agile methods DSDM's goal is to deliver projects on time and on budget while at the same time is flexible enough to accommodate change in requirements.	High user involvement. Basic Functionalities are delivered faster and even more at frequent intervals. Projects are delivered on time and on budget. Provides access by developers to end users.	Not suitable for small organizations or one time projects. As it is a newer model in comparison to other traditional model such as the waterfall therefor it is not as common and easy to understand. Because of its strictness and eight principles, DSDM can be restrictive and difficult to work with compared to other agile development software methods.
XP(Extreme programming) shares all Agile principles including strong customer involvement in the software development process, good communication inside of the teams, and iterative cycles of development.	The main advantage is that it allows software development companies to save costs and time required for project realization. Simplicity is one more advantage of Extreme Programming projects. The whole process in XP is visible and accountable.	Some specialists say that Extreme Programming is focused on the code rather than on design. That may be a problem because good design is extremely important for software applications.
Kanban is a method applied across all fields of work to help teams drive down costs and become more efficient by visualizing and improving workflows.	Better visibility. Improved efficiency. Increased Productivity . Preventing team overburden.	It cannot be used as an independent tool. It is not a methodology that could be applied solely rather it can be merged with other processes and systems of a company like JIT, make to order and scrum etc. making these systems more visible.
Scrum in Agile software development is often perceived as a methodology but it is rather a framework for managing a process.	It creates a system of transparency. It offers motivation on multiple levels. It provides continuous feedback.	It does not care about the final project deadline. It requires a team environment. It requires experience.

- The customer is involved in the development of the application.

2.2.6 Scrum Framework

With Scrum, the project is broken up in 4 small pieces as described in Figure 2.1.

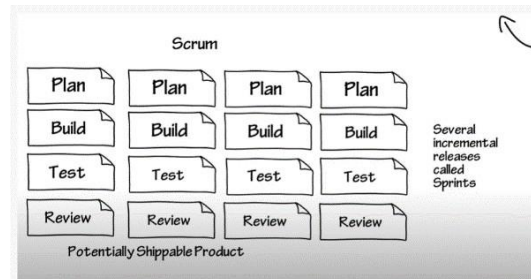


FIGURE 2.1: Project Division using Scrum

The Scrum model suggests that projects progress via a series of sprints. In keeping with an agile methodology, One sprint takes from 1 to 3 weeks. We keep repeating this sprint until your project is completely complete as explained in Figure 2.2.

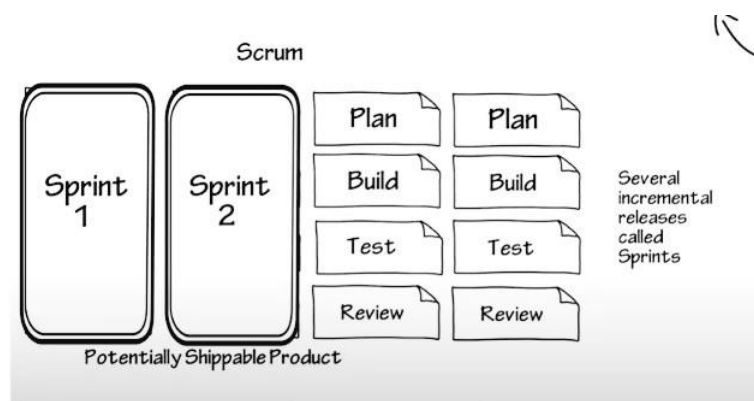


FIGURE 2.2: Scrum Project Sprints

2.2.7 Roles and Responsibilities

In Scrum there are 3 key roles -see Figure 2.3 needed to the framework to work well:

1. The product owner: This is the person responsible for finding the features needed in the product.
2. Scrum Master: A servant leader to the team , responsible for protecting the team and the process and keeping things going.
3. Team: It can be made of developers , testers , writers and everyone that helps involving the product , the team works to make the product done.

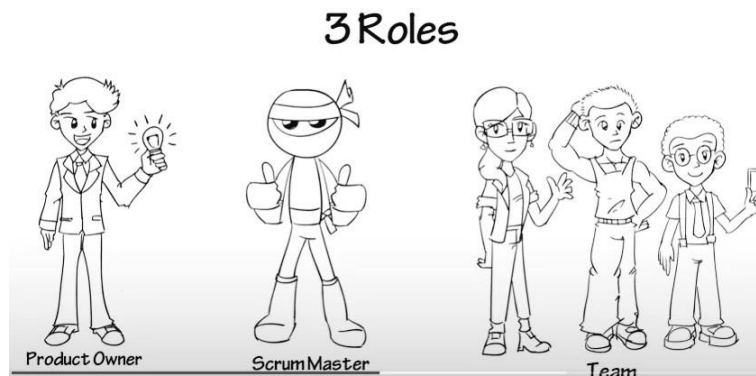


FIGURE 2.3: Scrum Roles

2.2.8 Scrum Artifacts

In scrum , there are 3 artifacts such as displayed in Figure 2.4:

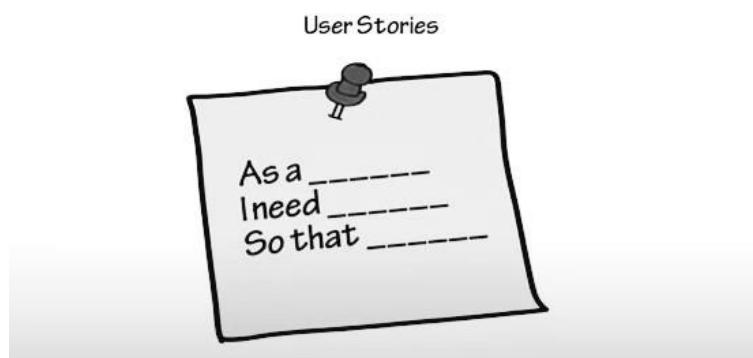


FIGURE 2.4: User Story Construction

1. **Product Backlog** : A prioritized list of features.
2. **Sprint Backlog** : A list of tasks identified by the Scrum team to be completed during the Scrum sprint and identifies the tasks necessary to complete each user story which are a way of describing a feature set (see Figure 2.5).
3. **Burndown Chart** : A graphic representation of how quickly the team is working through a customer's user stories. It shows the total effort against the amount of work for each iteration.

2.2.9 Scrum events

Scrum includes 4 events:

1. **Sprint planning** : An event that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. It is done in collaboration with the whole scrum team.
2. **Daily scrum** : A meeting where the teams discuss what they have achieved and plan what is to be achieved the following 24 hours.

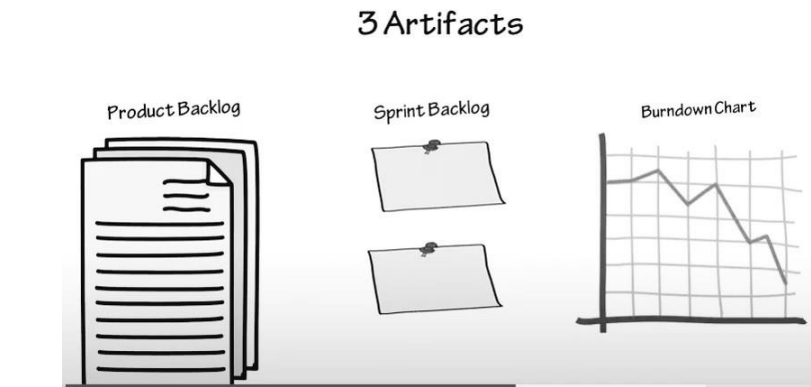


FIGURE 2.5: Scrum Artifacts

3. Sprint review: An inspection of the Burndown Chart and adapt the Product Backlog if needed.
4. Sprint retrospective: An opportunity for the Scrum Team to question itself and establish a plan for improvements to be applied during the next Sprint.

2.3 Technological Study

2.3.1 Hardware Architecture

Our website will be developed using two laptops and will be tested on one laptop with the following specifications:

Laptop name: Lenovo Idea pad gaming 3i 15 (See Figure 2.6)

Operating system: Windows 10 Home 64



FIGURE 2.6: First Laptop used for Development and Testing

Processors: 10th Gen Intel Core i7-10300H (4C / 8T, 2.5 / 4.5GHz, 8MB)

Memory: 16GB

Graphics: NVIDIA GeForce GTX 1650 4GB GDDR6

The other laptop that will be used for development has these specifications:

Laptop name: HP pavillon (see Figure 2.7)

Operating system: Windows 8.1 Professional

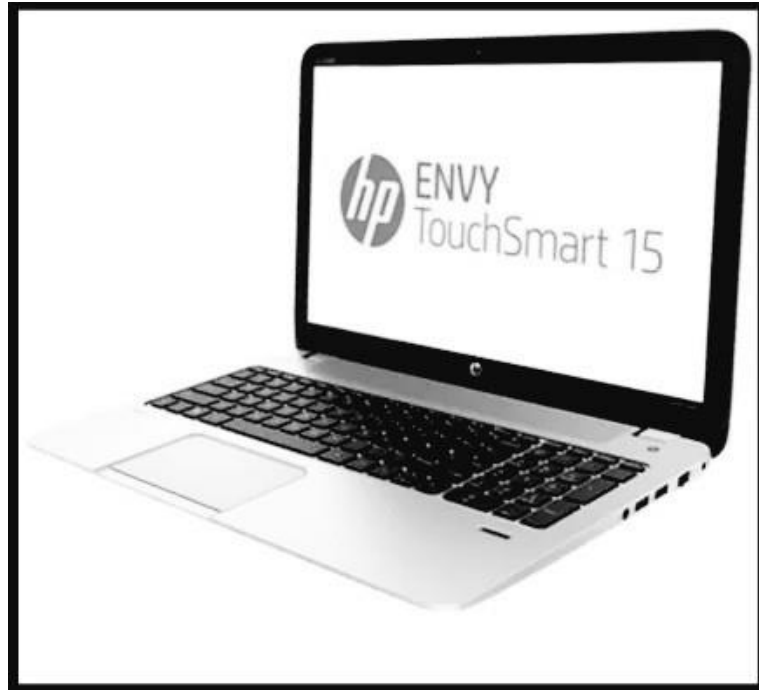


FIGURE 2.7: Second Laptop used for Development only

Processors: 4th Gen Intel Core i7-4510U CPU @2.00GHz-2.60GHz

Memory: 8GB

2.3.2 Software Environment

In this section , we are going to list all the Software that we used to develop the website.

Visual Studio Code (Figure ??) is a free source-code editor made by Microsoft for Windows, Linux and macOS. It is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE. It has a rich system of extensions (such as C++, C, Java, Python, PHP...).

MySQL (Figure 2.9) is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL runs on virtually all platforms, including Linux, UNIX and Windows. Although it can be used in a wide range of applications, MySQL is most often associated with web applications and online publishing. MySQL was originally developed to handle large databases quickly. Although MySQL is typically installed on only one machine, it is able to send the database to multiple locations, as users are able to access it via different MySQL client interfaces. These interfaces send SQL statements to the server and then display the results.

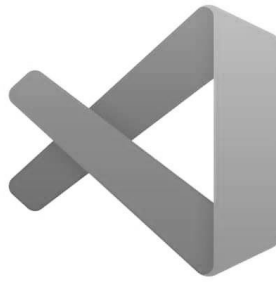


FIGURE 2.8: Visual Studio Code Logo



FIGURE 2.9: MySQL Logo

Node.js (Figure 2.10) is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.



FIGURE 2.10: Node.js Logo

Angular (Figure 2.11) is Google's JavaScript (TypeScript-based) open-source front-end web application framework. It is designed specifically for creating dynamic web applications. With this framework, you can develop front-end based applications without having to use other plug-ins or frameworks. Angular is used to develop state-of-the-art client-side applications, especially Single-Page applications. Pure JavaScript is used to write this Angular framework. Various platforms like mobile, web and desktop natives are supported by Angular.

Bootstrap (Figure 2.12) is a potent front-end framework used to create modern websites and web apps. It's open-source and free to use, yet features numerous



FIGURE 2.11: Angular Logo

HTML and CSS templates for UI interface elements such as buttons and forms. Bootstrap also supports JavaScript extensions. Where it can be used together with modern JavaScript web mobile frameworks like Angular.



FIGURE 2.12: Bootstrap Logo

Postman (Figure 2.15) is an interactive and automatic tool for verifying the APIs of your project. It is a Google Chrome app for interacting with HTTP APIs. It presents you with a friendly GUI for constructing requests and reading responses. It works on the backend, and makes sure that each API is working as intended.

2.3.3 Architecture Environment

The client-side part of our application (Figure 2.14), will be developed using Angular where it is going to handle using their modules and injectors the front-end part. Our app contains several pages, thanks to Angular, we will manage to have these pages connected in a dynamic way.

For the server-side part, our application will use Node.js as a back-end framework, where it builds a HTTP server using HTTP models and be able to connect our data base to the front-side following a path. It manages also all the requests of the client and ensure the dynamicity of the program.



FIGURE 2.13: Postman Logo

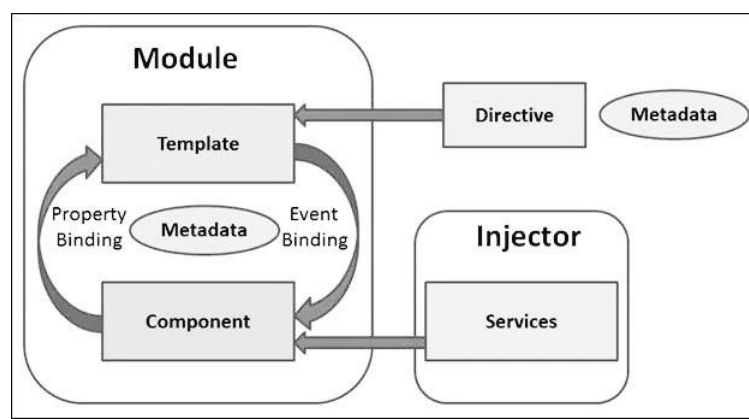


FIGURE 2.14: Client-side Application

2.4 Requirement Analysis

In this section, we will focus on the customer requirements that we are going to work on during the project. During this phase, we will go into determining the functional and the non-functional requirements that need to be implemented into the system in order to end up with a product backlog.

2.4.1 Functional Requirements

In this sub-section, we will go into functional requirements analysis that will explain what has to be done and will identify the necessary tasks that must be accomplished. Our project has basically six main points as listed below.

- **Access the platform**
Our application must allow the user to log in to the platform using his account, be able to visualize his various options and go through the several features that the platform offers.
- **Schedule appointments**
As our application provides to our user (as a patient) the comfort of picking a time slot and asking for an appointment, it should appear on the schedule of the doctor in order to be confirmed. This operation requires synchronization between the scheduling demands to achieve the highest performance.

- **Manage Profiles Patient lists**
The user has the ability to modify his personal information and even delete his account. On the other hand, the doctor can visualize his patient lists, manage the current ones and archive their related information.
- **Live chat receive notifications**
This feature will allow the patient to have a chat with his doctor on his situation and the doctor may hand subscriptions through the chat. All users can be notified, if preferred, by their upcoming events and reminded if they have forums to attend.
- **Online consultation**
The appointment patient-doctor can be set online and the concerned users will be notified before the event. This includes an online payment option.
- **Recommendation System**
Once the contact between the patient and the doctor is done, the patient should have the freedom to evaluate the doctor and recommend him or not to the other users in the community.

2.4.2 Non-functional Requirements

The non-functional requirements are important as the functional requirements, after all, they have a big impact on the final product quality. And they can be so decisive on its success or failure. And for that, we need to give big attention on them. The next list represents the main points that our product must provide and respects.

- **Security**
Since our platform contains personal information about the patient's health, we must respect and protect all the gathered data, and ensure confidentiality.
- **Usability**
Our application should be user-friendly and should guide the user, no matter his technological background, through a smooth user experience.
- **Reliability**
Our application must provide an experience without bugs and errors to our users, and must hold all the operating correctly.
- **Scalability**
The application should make it easy to add new features at the lowest cost. For that, the system needs to follow a clear architecture so it will be easy.

2.5 Identifying Actors

An actor in software engineering is a role played by the user or any other system that interacts with the program and exchange information. The primary and secondary actors are listed respectively in Table 2.2 and Table 2.3 with their descriptions.

2.6 Product Backlog

The Tables 2.4 represents the product backlog of our project.

TABLE 2.2: The Primary Actors

Actors	Descriptions
The patient	The patient is able to take an appointment with doctor and chooses whether the consultation will be online or in the doctor's office . The patient can have a video chat with the doctor.
The doctor	The doctor Can choose whether he/she accepts the consultation or not . The doctor can manage his/her schedule.

TABLE 2.3: The Secondary Actors

Actors	Descriptions
API server	The server responsible for defining the request-response message , which is exposed via the web.
The recommendations system	It is responsible for analyzing the patients' patterns of the patient and recommending them to the right doctor.

In our project, Najla Seghaier will be the product owner, Dr. Asma Amdouni will be the scrum master, and Jihene Chouiref, Mohamed Amine Mhiri, Mohamed Aziz Rourou and Mohamed aziz Trabelsi, will form the scrum team.

2.7 General Use Case Diagram

In this subsection, we will describe our system with a use case diagram as shown in Figure ??.

2.8 Conclusion

In this chapter, we put the attention on the method and methodology used. We presented our product backlog and our general use case diagram which will help us understand what has to be done in each sprint.

2.9 References

Most of the information in this chapter is not our own work. All merits go to the persons and websites cited between parentheses (VisualParadigm, n.d. PMI, n.d. Synopsys, n.d. Kruchten, 2004; Sabbir M Saleh, 2017; Scrum, n.d. VisualStudioCode.com, n.d. Node.js, n.d. MySQL, n.d. Angular.io, n.d. Postman.com, n.d.) . For more details please refer to the bibliography at the end of the report.

TABLE 2.4: The Product Backlog

User story	Product backlog item	Sprint	Priority	Estimation
As a user, I'm able to create an account	Platform access	1	1	4 days
As a user, I'm able to connect to my account				2 days
As a user, I'm able to modify my account	Account management	1	2	3 days
As a user, I'm able to delete my account				3 days
As a doctor, I'm able to manage current patients	Patient list management	1	3	4 days
As a doctor, I'm able to manage archive of patients				3 days
As a patient, I'm able to take an appointment	Appointment scheduling	2	4	3 days
As a doctor, I'm able to validate an appointment				2 days
As a patient, I'm able to cancel an appointment				2 days
As a patient, I'm able to chat with my doctor	Live chat, Forum & Notifications	2	5	3 days
As a doctor, I'm able to chat with my patient				1 day
As a patient, I'm able to participate in a forum				2 days
As a user, I'm able to receive a notification				2 days
As a patient, I'm able to consult doctor online	Online consultations	3	6	7 days
As a doctor I'm able to generate prescriptions online				4 days
As a patient I'm able to evaluate doctor	Doctor recommendation system	3	7	3 days
As a patient, I'm able to recommend doctor				3 days
As a patient, I'm able to get diagnostic				3 days
As a patient, I'm able to get recommendations of doctors suitable to described state				3 days

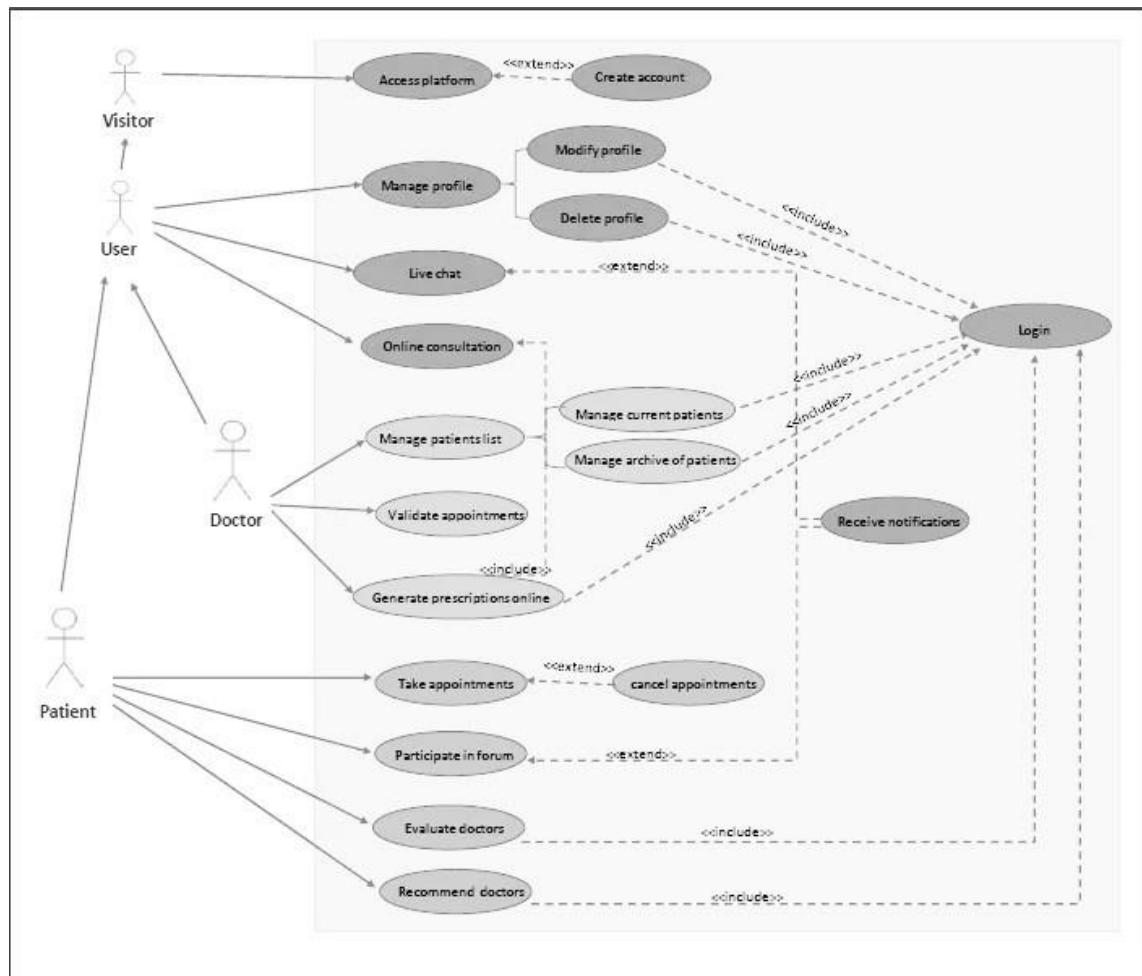


FIGURE 2.15: Genral Use Cas Diagram

Chapter 3

Sprint 1

3.1 Introduction

In this section, we introduce the first sprint of our project. First, we present the sprint's organization and its backlog. Then, we analyse all our user requirements for this sprint and explain a few scenarios for a number of those requirements using Use Case, Sequence and Class Diagrams. To conclude this chapter, we display some screenshots and illustrated demonstrations of the actual implementation of this sprint.

3.2 Organization

See Figure 3.1.

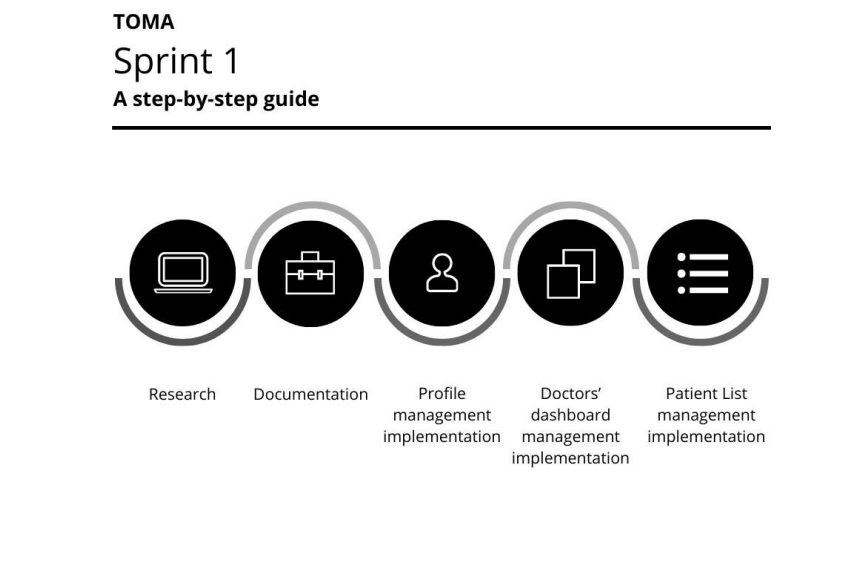


FIGURE 3.1: Diagram showing the Organization of the First Sprint

3.3 Sprint Backlog

See Table 3.1.

TABLE 3.1: Sprint 1 Backlog showing the tasks, the complexity and the estimation of the duration of each task

	Tasks	Complexity	Estimation
SPRINT 1	create backend user & models	M	3 days
	Create user & Authentication API's (Full CRUD) and controllers	H	3 days
	create user's interaces using Angular	H	3 days
	Link Angular Intefraces to Node Js	H	3 days
	develop functions such as get so that the user can access his/her info	M	1.5 day
	Give the Users the ability to Update and Delete their info	M	1 day
	create the models of patients	M	1 day
	Stores list patients in a mongoDB database		0.5 day
	develop functions such as get to access the patients' info	M	1 day
	develop functions such as archive , restore and add to manage the patient list	M	1.5 day
	link the functions to the client side server	H	1.5 day
	Total		20 days

3.4 Analysis

3.4.1 Use Case Diagram

See Figure 3.2.

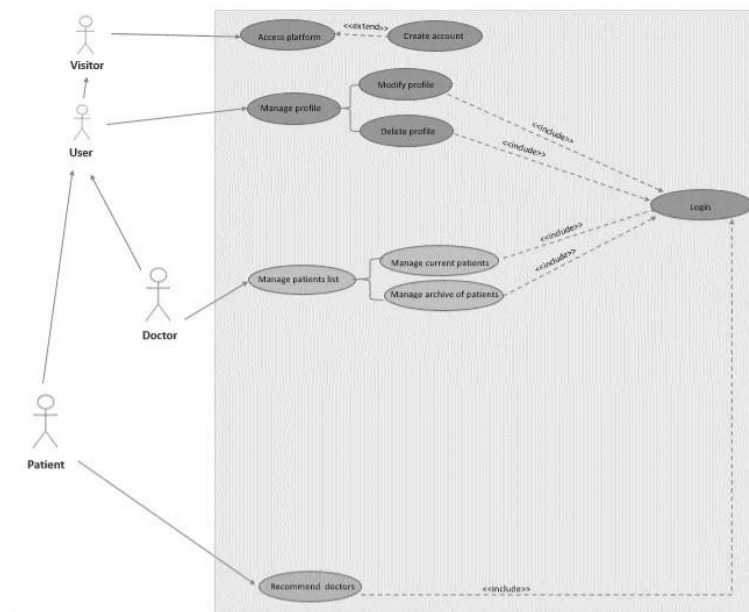


FIGURE 3.2: Use Case Diagram of the First Sprint

3.4.2 Textual Description

Use case scenario 1:

Req1: The patient shall be able to access the platform[High priority].

Req2: The patient shall be able to manage their account [High priority].

Precondition:

- The platform is functional

- The patient has successfully accessed the sign up/login page

Post-condition:

- The patient has signed up and is logged in

Normal scenario:

1. The system loads the page with the sign up/login page
2. The patient views the sign up/login page
3. The patient enters their email and password
4. The patient validates their credentials by clicking on sign up
5. The system verifies the validity of the credentials
6. The system shows the new profile of the patient
7. The patient is logged in

Alternative scenario:

- The patient's credentials contain errors or already existing
- The system displays the errors and asks the patient to
- re-enter the data The scenario starts at step 3

Use case scenario 2:

Req1: The doctor shall be able to upload and edit the patients' list[High priority].

Precondition:

- The doctor has successfully accessed their profile

Post-condition:

- The added/edited patients appear in a list

Normal scenario:

1. The system loads the page with the patients' list
2. The doctor views the patients' descriptions
 - (a) The doctor clicks on "create new description"
 - (b) The doctor adds the patient name
 - (c) The doctor writes the description
 - (d) The doctor clicks on save

- The doctor clicks on “edit”
- The doctor enters modifications to the description
- The doctor clicks on “save changes”

Alternative scenario:

- The doctor does not click on save
- The doctor is redirected to the patient list
- The scenario starts at step 1

3.5 Design

3.5.1 Sequence Diagram

See Figures 3.3 and 3.4.

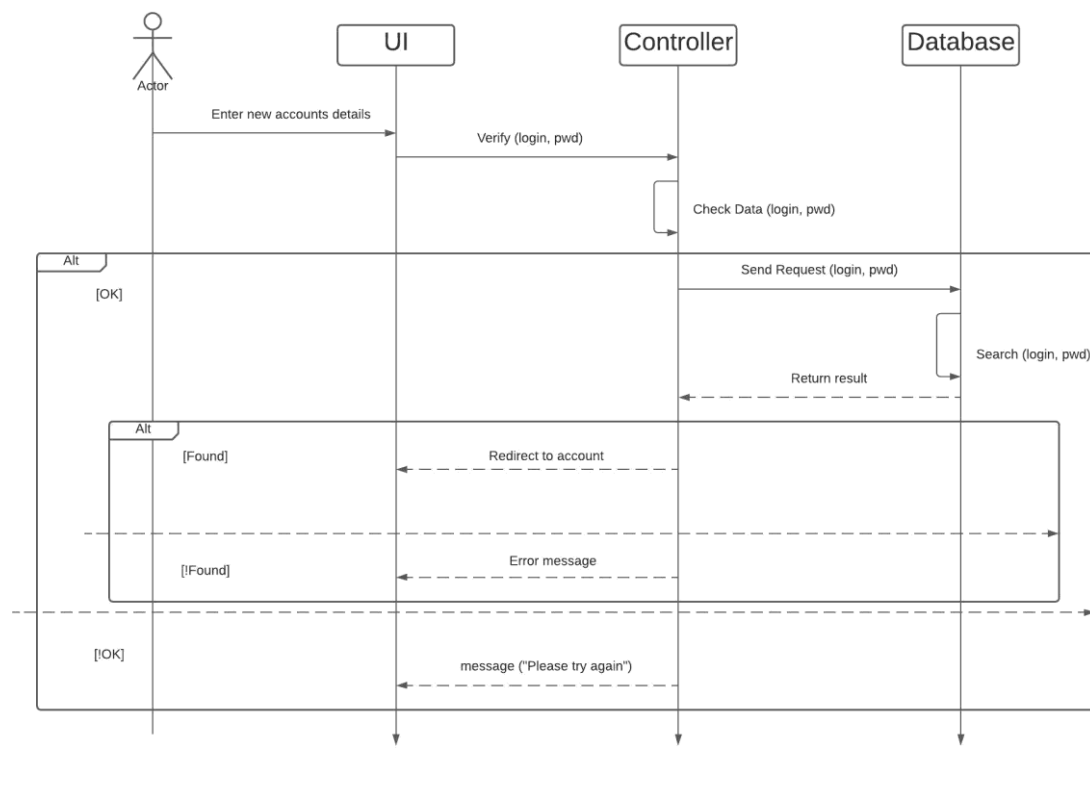


FIGURE 3.3: Sequence Diagram showing how a visitor signs in and then logs in as a user

3.5.2 Class Diagram

See Figure 3.5.

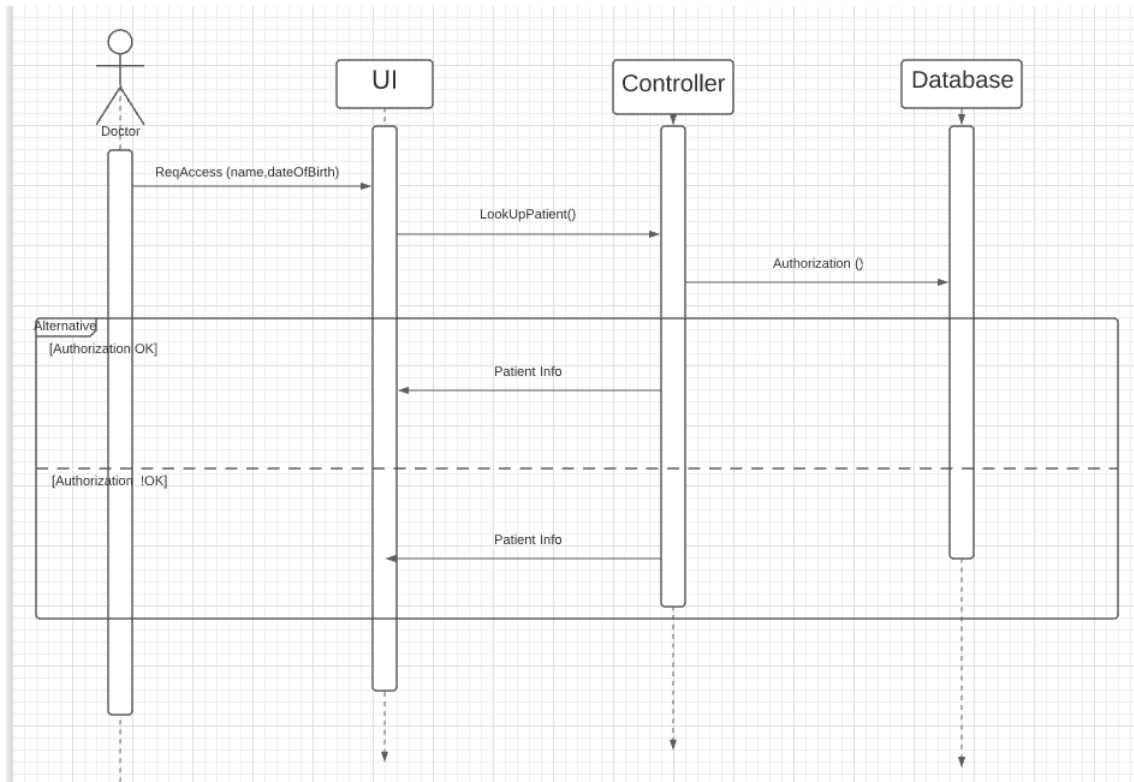


FIGURE 3.4: Sequence Diagram showing how a doctor accesses the patients list

3.6 Implementation

The first page that appears to our visitors whether they are patients or doctors is the Create Account page. In the following scenario we chose to assume for a start that our user is a patient. The patient starts by entering her basic credentials such as an email and a password (Figure 3.6) then he/she enters their other credentials at a second step (Figure 3.7). From there after an email confirmation, the patient can sign in to their account (Figure 3.8). Now she can check the doctors dashboard where all the doctors are listed (Figure 3.9). From there she can filter the doctors according to the choices she enters in the left filter bar. Assuming here that the patient already connected with a few doctors, she can now filter her doctors (Figure 3.10). We here suppose that the patient took an appointment with one of her doctors, a small window pops up for her to enter the date and the reason why she is taking the appointment (Figure 3.11). Now that the request for the appointment has been sent, the patient decides to update her account information (Figure 3.12). Meanwhile, a doctor who is also connected to his account receives the request from our patient (Figure 3.13). He has the choice to refuse or accept it. He goes to the dashboard to see all his appointments where he finds only our patients appointment (Figure 3.14). Once he accepts the request, our patient is added to his patients' list (Figure 3.15). Later on, the doctor checks the archived patients' list (Figure 3.16). At the same time our patient refreshes her page and finds her accepted request alongside her other pending and refused requests (Figure 3.17). Before logging out, she checks again on her doctors dashboard for her pending requests (Figure 3.18).

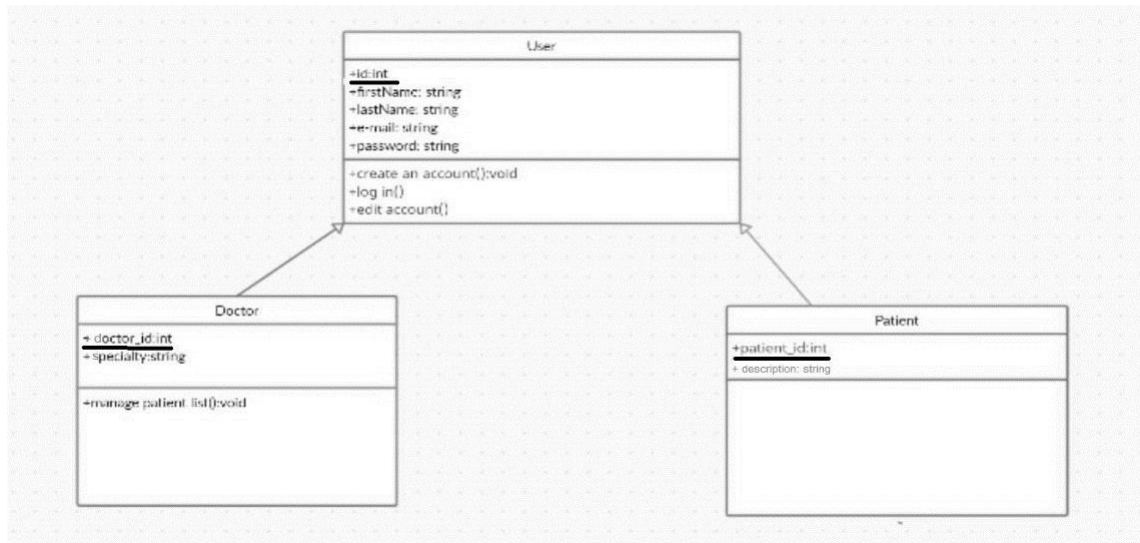


FIGURE 3.5: Class Diagram of the First Sprint

3.7 Conclusion

As shown in this chapter, the requirements have been met as planned. We provided an interface where the users can create and access their profiles. The patients can now access and filter the dashboard for doctors and the doctors can manage their patients' list. In the next chapter, we will focus on the appointment scheduling.

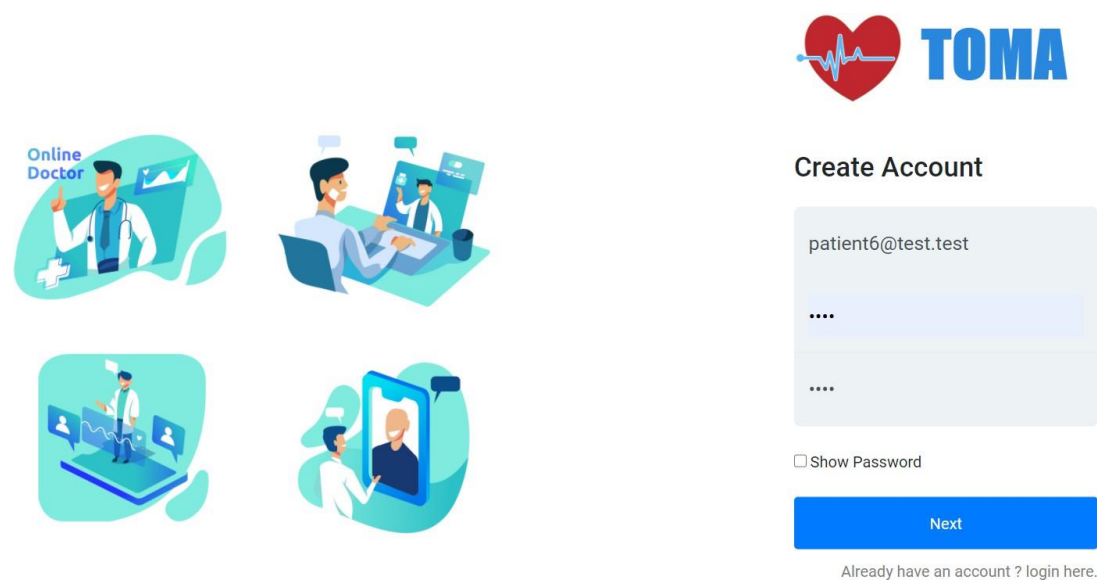


FIGURE 3.6: Create Account Step 1

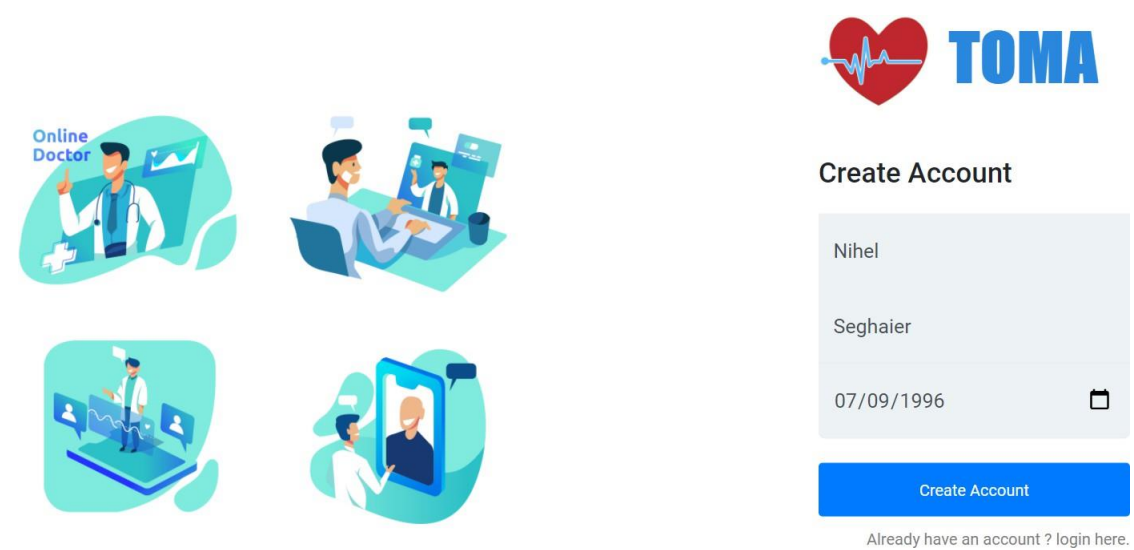


FIGURE 3.7: Create Account Step 2

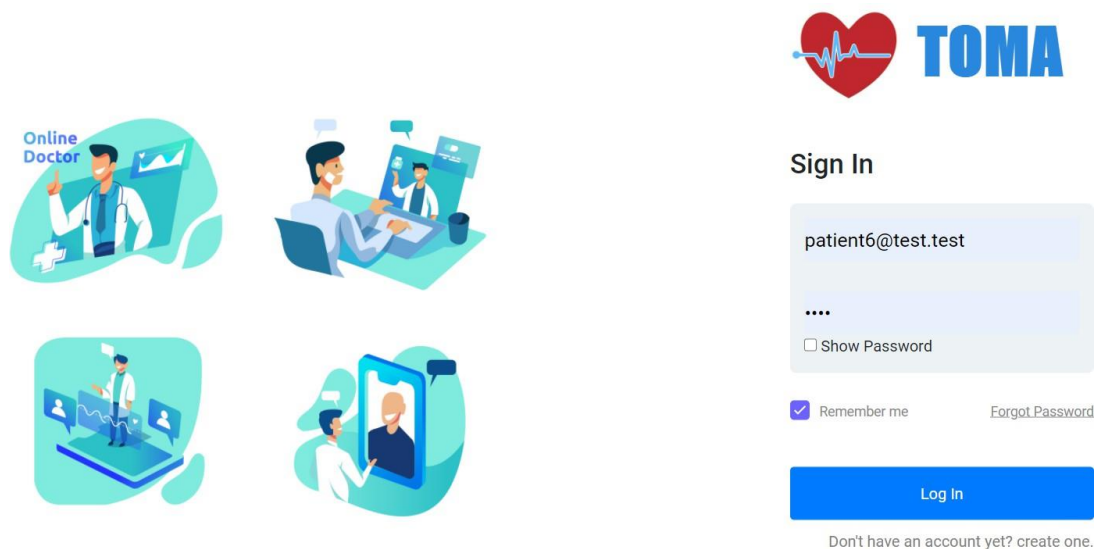


FIGURE 3.8: Sign in

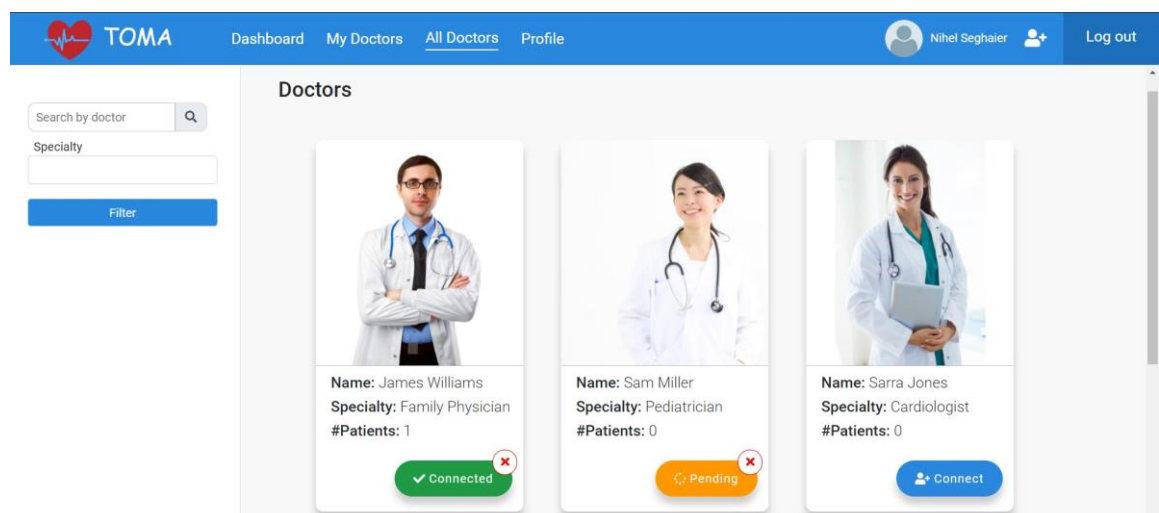


FIGURE 3.9: Filter All Doctors

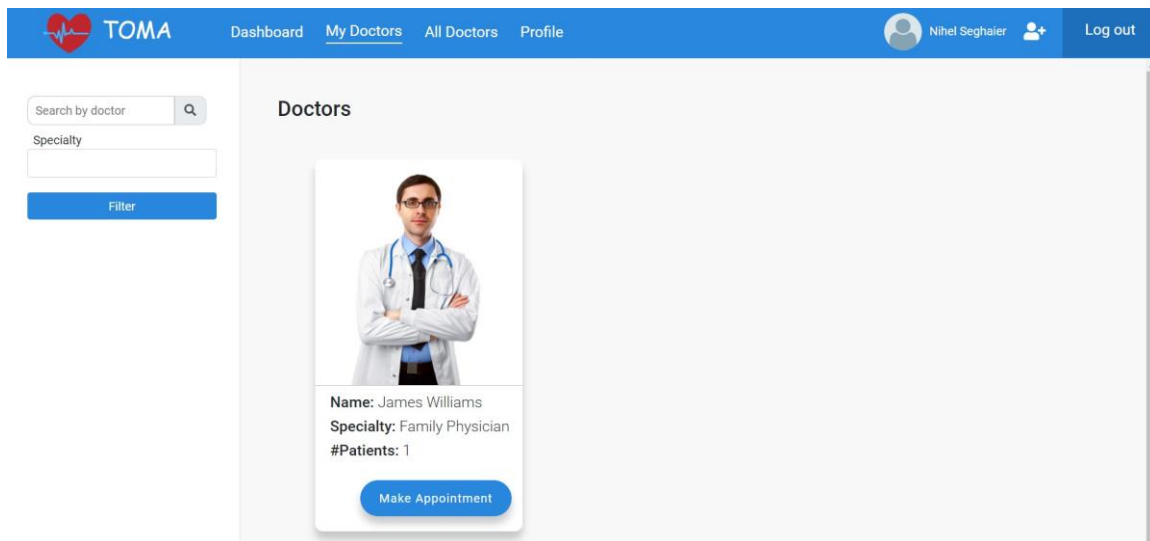


FIGURE 3.10: Filter My Doctors

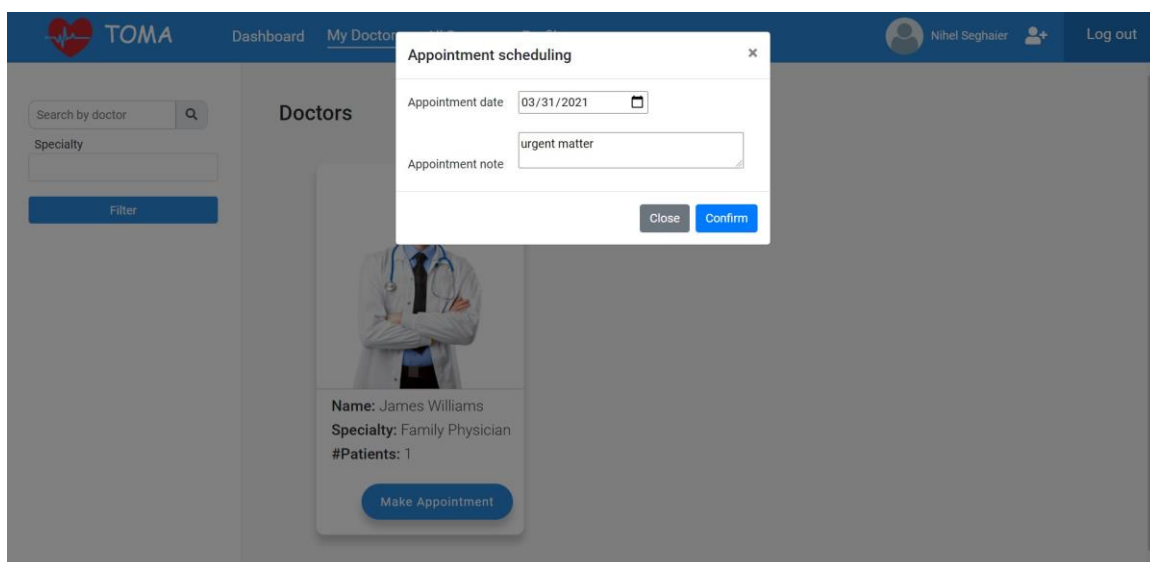
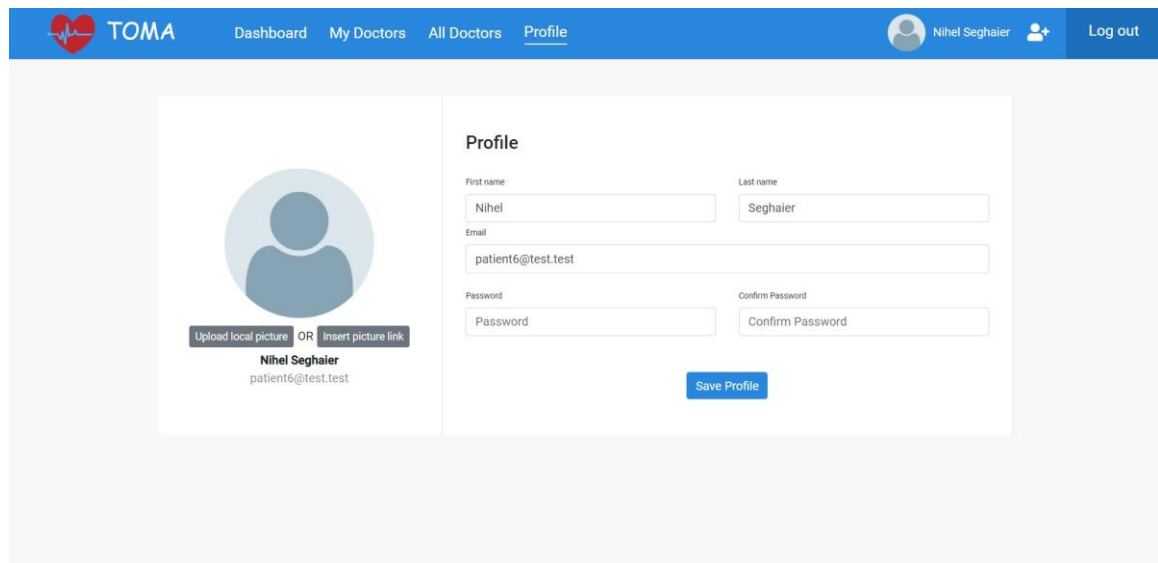
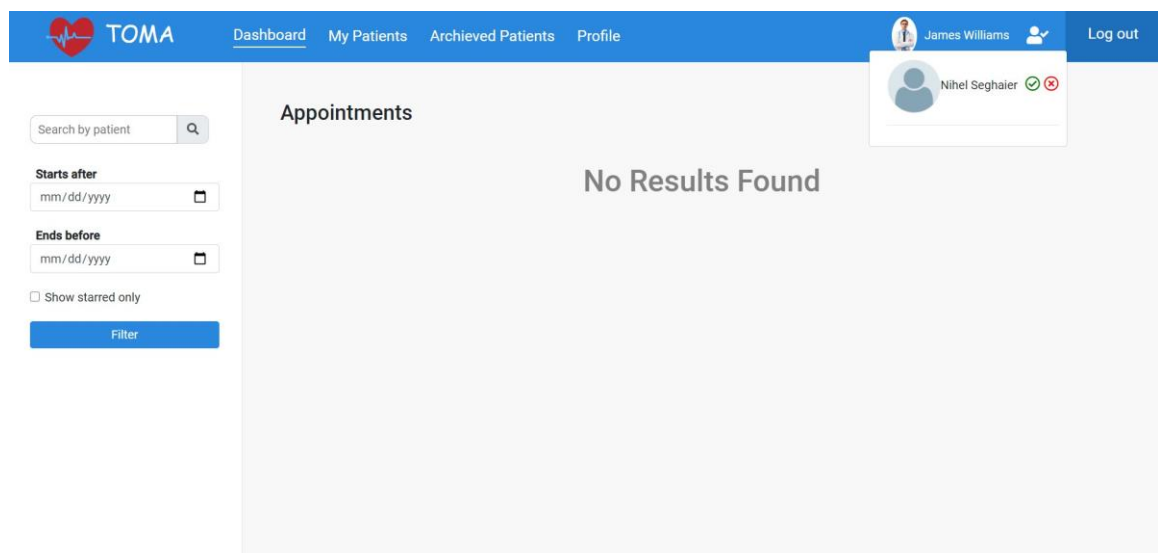


FIGURE 3.11: Schedule Appointment Request



The screenshot shows the TOMA application's profile update interface. The top navigation bar is blue with the TOMA logo, links for Dashboard, My Doctors, All Doctors, and Profile, and a user profile for Nihel Seghaier with a Log out button. The main content area is divided into two sections. On the left, there is a circular profile picture placeholder with the text "Upload local picture OR Insert picture link" and the user's name "Nihel Seghaier" and email "patient6@test.test". On the right, the "Profile" section contains form fields for First name (Nihel), Last name (Seghaier), Email (patient6@test.test), Password, and Confirm Password. A "Save Profile" button is at the bottom right.

FIGURE 3.12: Update Profile Information



The screenshot shows the TOMA application's doctor's request notification page. The top navigation bar is blue with the TOMA logo, links for Dashboard, My Patients, Archived Patients, and Profile, and a user profile for James Williams with a Log out button. A dropdown menu is open, showing the profile of Nihel Seghaier with a green checkmark and a red X icon. The main content area is titled "Appointments" and displays "No Results Found". On the left, there is a search bar labeled "Search by patient" and two date pickers labeled "Starts after" and "Ends before", both with the format "mm/dd/yyyy". A checkbox labeled "Show starred only" and a "Filter" button are also present.

FIGURE 3.13: Doctor's Request Notification

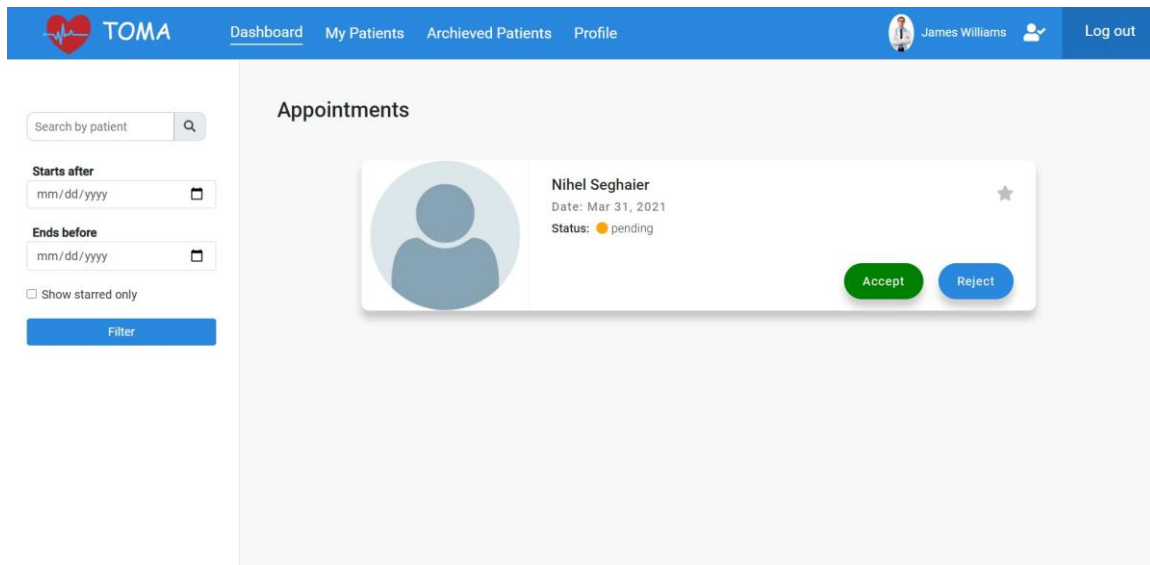


FIGURE 3.14: Doctor's Appointment List

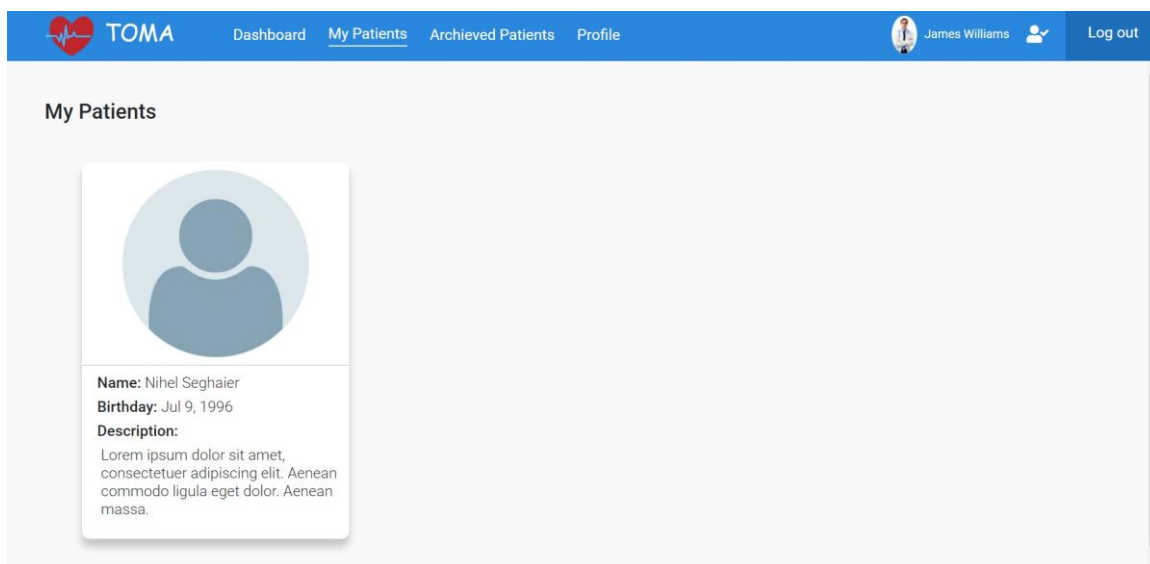


FIGURE 3.15: Doctor's Patient List

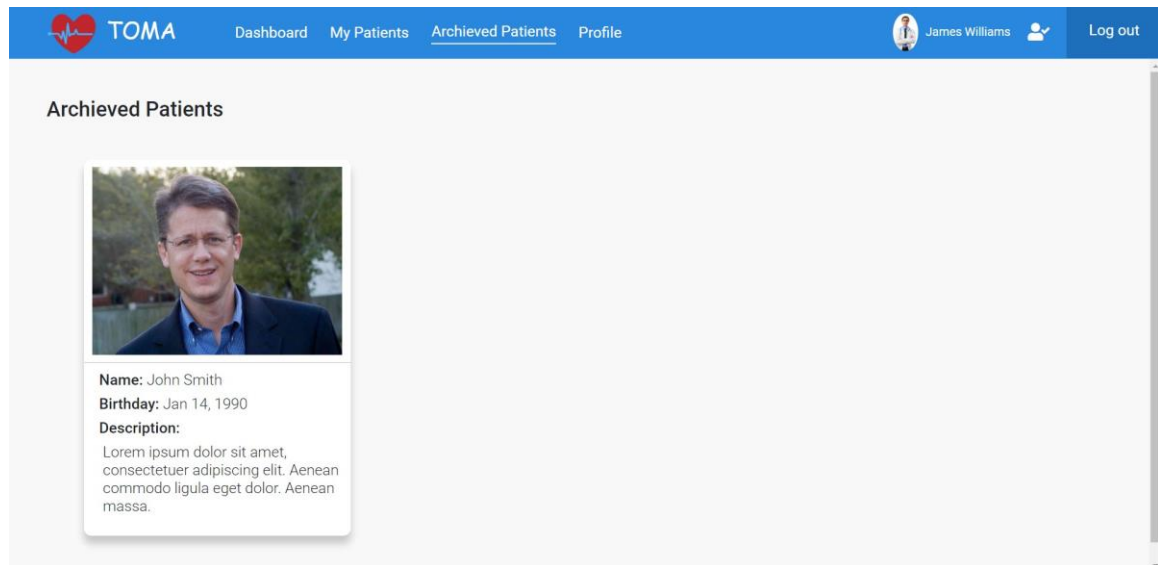


FIGURE 3.16: Doctor's Archived Patient List

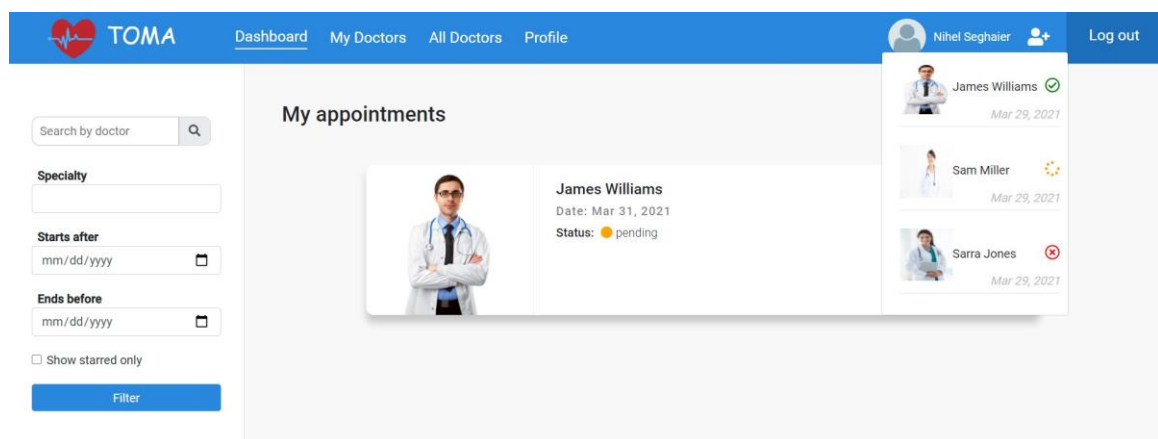


FIGURE 3.17: Patient Requests Status

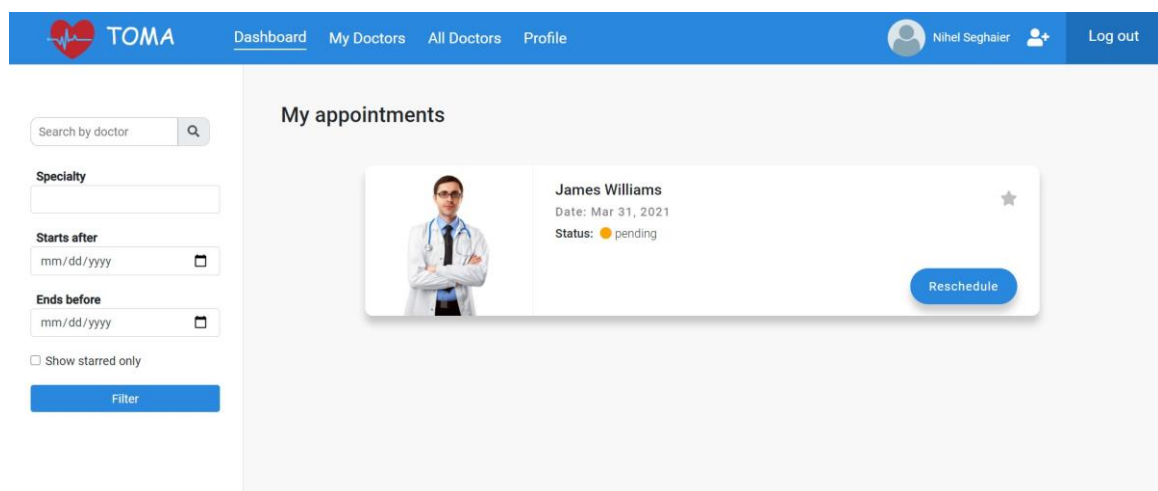


FIGURE 3.18: Patient Appointment List

Chapter 4

Sprint 2

4.1 Introduction

In this section, we introduce the second sprint of our project. First, we present the sprint's organization and its backlog. Then, we analyse all our user requirements for this release and explain a few scenarios for a number of those requirements using Use Case, Sequence and Class Diagrams. To conclude this chapter, we display some screenshots and illustrated demonstrations of the actual implementation of this sprint.

4.2 Organization

See Figure 4.1.

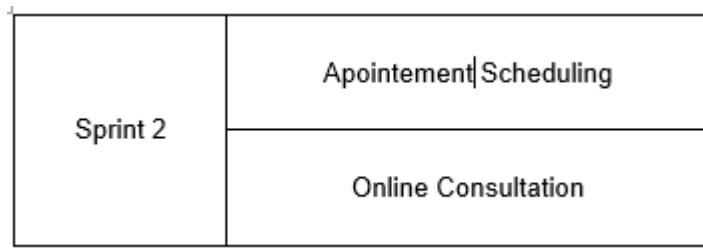


FIGURE 4.1: Diagram showing the Organization of the Second Sprint

4.3 Sprint Backlog

See Table 4.1

4.4 Analysis

4.4.1 Use Case Diagram

See Figure 4.2

4.4.2 Textual Description

Use case scenario 1:

Req1: The patient shall be able to access the platform[High priority].

TABLE 4.1: Sprint 2 Backlog showing the tasks, the complexity and the estimation of the duration of each task

Topics	Tasks	Complexity	Sprint	Estimation
Appointment Scheduling	-create appointment models	M	Sprint 2	3
	-loading calendar appointments from a database	L		1
	-storing appointments in a mongoDB database	M		2
	-create a function where you can delete an appointment	H		6
Online Consultation	-implement the streaming by using Apple HTTP Live Streaming (HLS) with Video on Demand (VOD) using a desktop application and IIS.	H		8
			Total	20

Req2: The patient shall be able to manage their account [High priority].

Precondition:

- The platform is functional
- The patient has successfully accessed the sign up/login page

Post-condition:

- The patient has signed up and is logged in

Normal scenario:

1. The system loads the page with the sign up/login page
2. The patient views the sign up/login page
3. The patient enters their email and password
4. The patient validates their credentials by clicking on sign up
5. The system verifies the validity of the credentials
6. The system shows the new profile of the patient
7. The patient is logged in

Alternative scenario:

- The patient's credentials contain errors or already existing
- The system displays the errors and asks the patient to
- re-enter the data The scenario starts at step 3

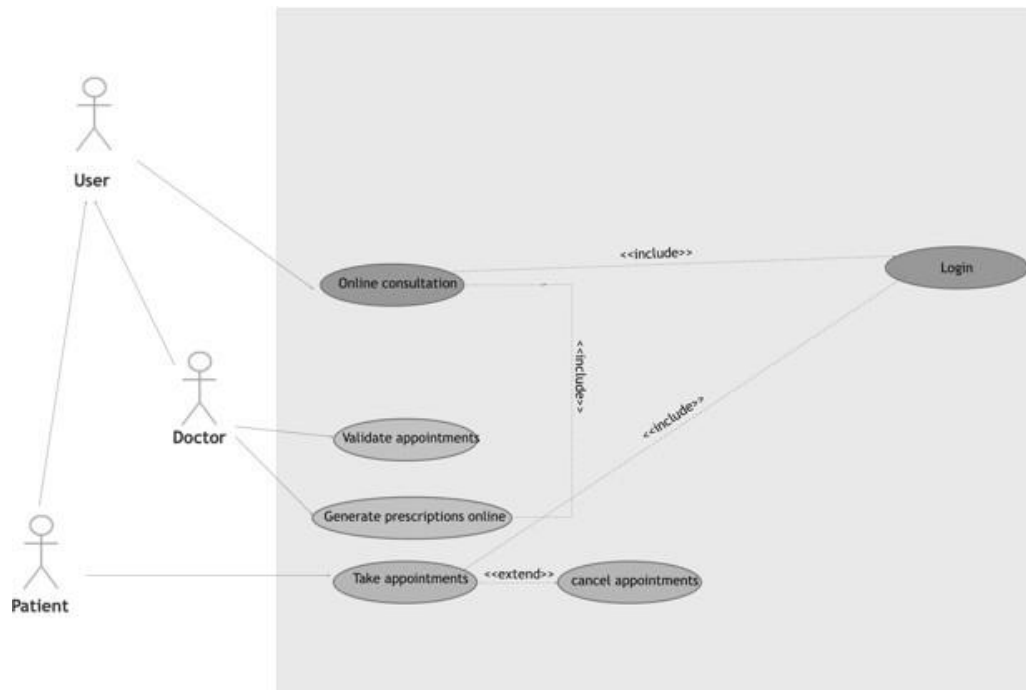


FIGURE 4.2: Use Case Diagram of the Second Sprint

Use case scenario 2:

Req1: The doctor shall be able to pick an appointment [High priority]. Req2: The doctor and the patient shall receive a message telling them if the appointment is validated successfully [High priority].

Precondition:

- The database contains booked appointments for the doctors.
- The doctors has successfully accessed their profile.

Post-condition:

- The appointment have been validated successfully

Normal scenario:

1. The system loads the page in which the appointments are validated
2. The doctor select an appointment
3. The doctor select to validate it
4. The system check the availability
5. The system displays a message telling the appointment is validated successfully

Alternative scenario:

- There is a constraint that blocks the appointment from being validated.
- The system will display a message telling the doctor and the patient that the appointment can't be validated and why.

4.5 Design

4.5.1 Sequence Diagram

See Figures 4.3 and 4.4.

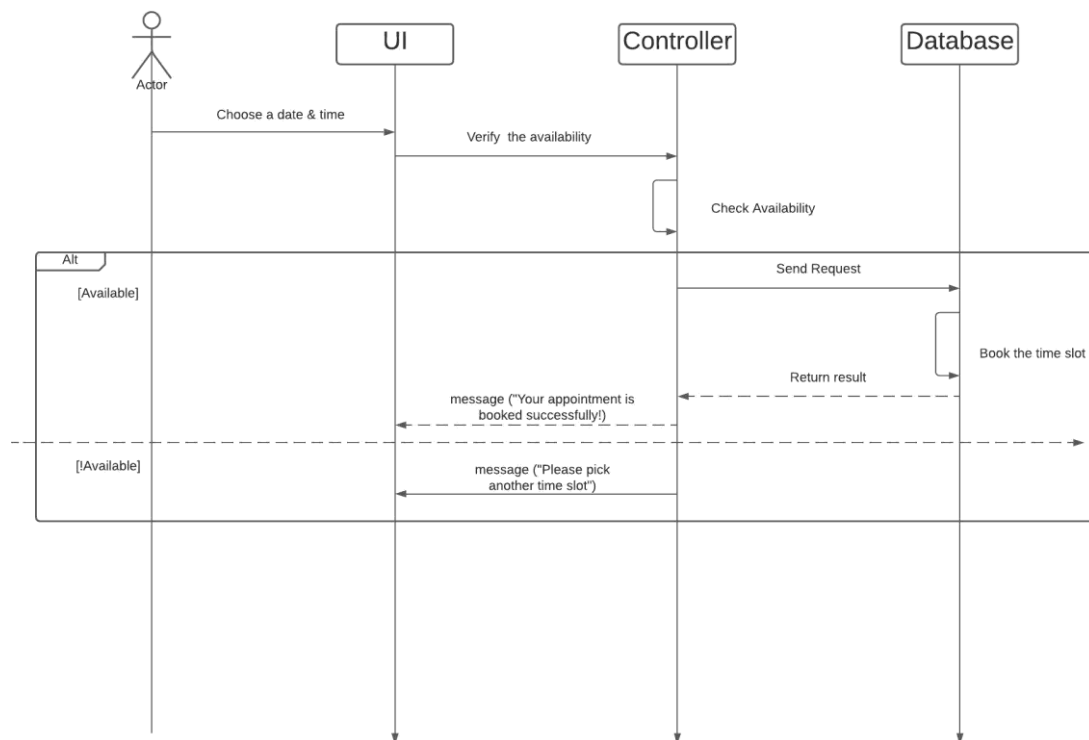


FIGURE 4.3: Sequence Diagram showing how a patient schedules an appointment

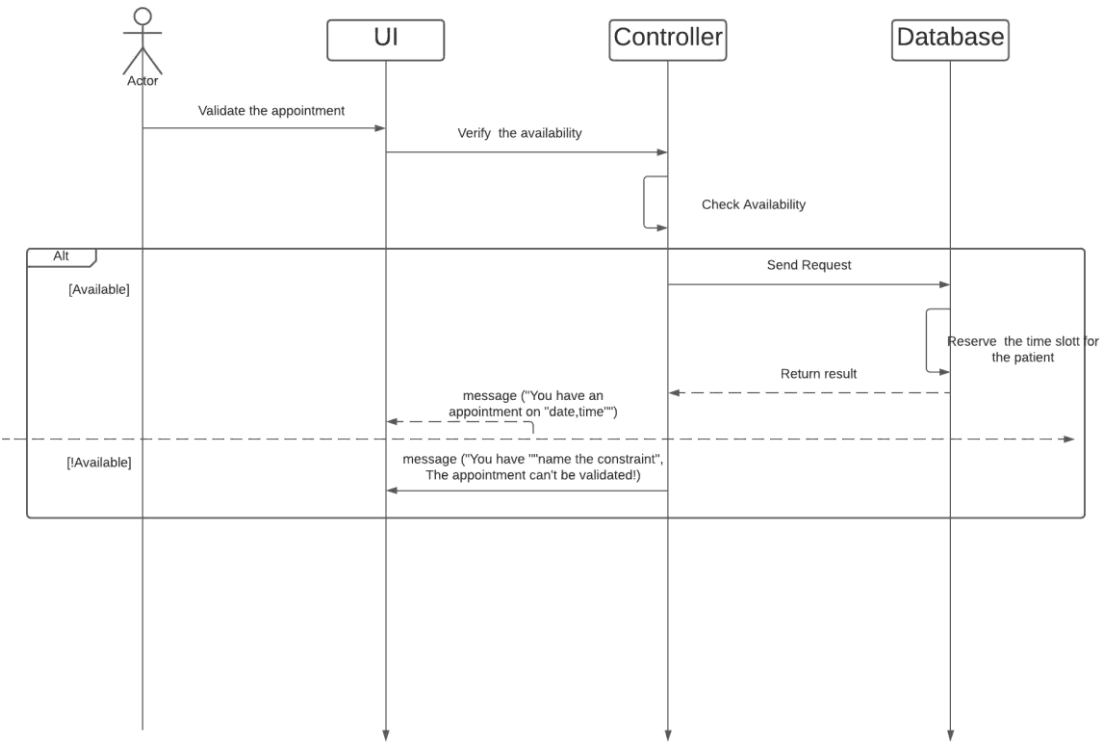


FIGURE 4.4: Sequence Diagram showing how a doctor validates the appointment

Chapter 5

Sprint 3

5.1 Introduction

In this section, we introduce the third sprint of our project. First, we present the sprint's organization and its backlog. Then, we analyse all our user requirements for this sprint and explain a few scenarios for a number of those requirements using Use Case, Sequence and Class Diagrams.

5.2 Organization

See Figure 5.1

TOMA Sprint 3 A step-by-step guide

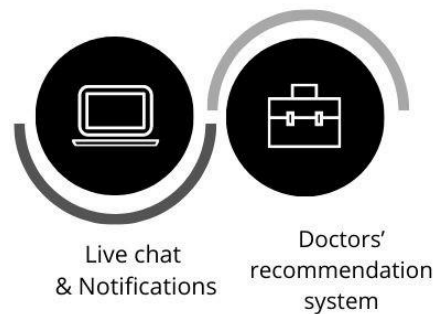


FIGURE 5.1: Diagram showing the Organization of the Third Sprint

TABLE 5.1: Sprint 3 Backlog showing the tasks, the complexity and the estimation of the duration of each task

		Tasks	Complexity	Estimation
Sprint 3	live chat & notifications	Setting up the server	H	14 days
		building the chat interface	M	7 days
		create an angular interface where patients and doctor can interact	M	7 days
		store the comments in the database	M	5 days
		send notifications to the user through cellphone number	M	7 days
		gives the ability to the patients to star the doctors	M	5 days
	doctor recommendation system	store the doctor's profile in the DB	M	5 days

5.3 Sprint Backlog

See Table5.1.

5.4 Analysis

5.4.1 Use Case Diagram

See Figure 5.2.

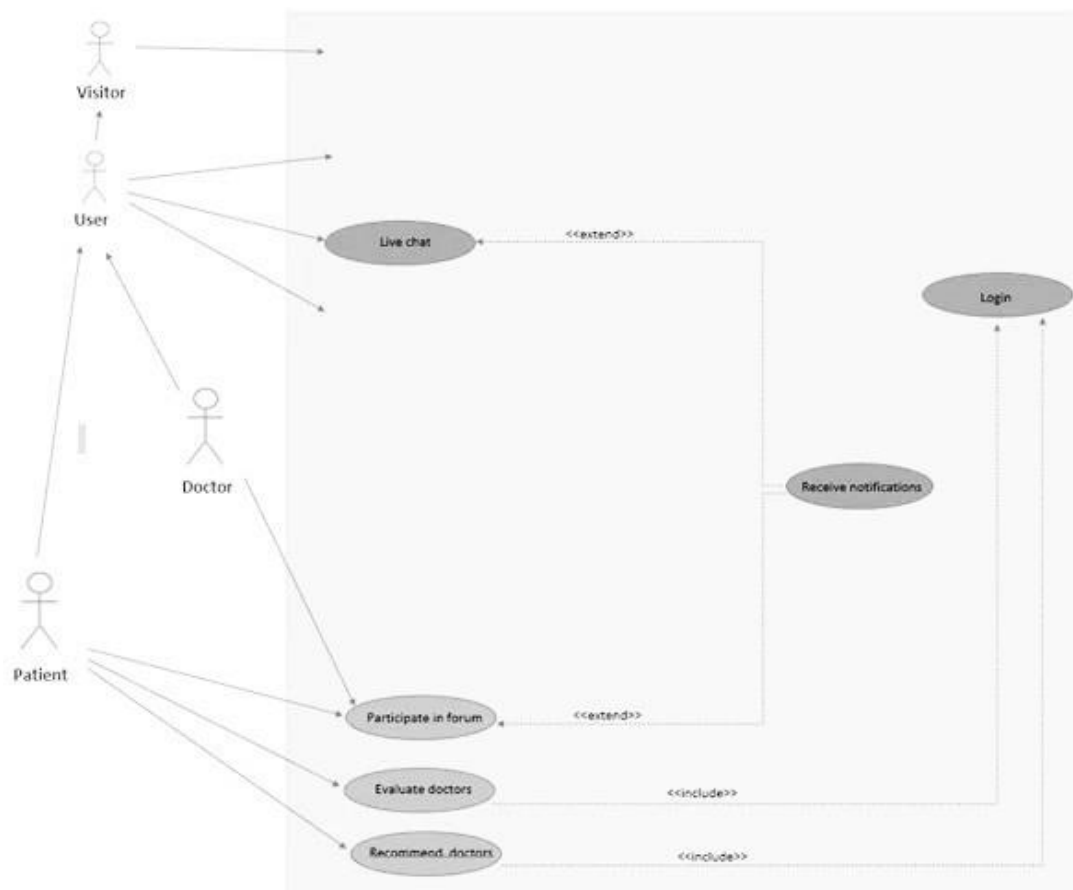


FIGURE 5.2: Use Case Diagram of the Third Sprint

5.4.2 Textual Description

Use case scenario 1:

Req1: The patient shall be able to evaluate the doctor
[Medium priority].

Req2: The patient shall be able to recommend doctors
[Medium priority].

Precondition:

- The patient completed at least one consultation
- The patient has successfully completed the chat or consultation with the doctor up/login page

Post-condition:

- The system stores the score of the evaluation(stars

Normal scenario:

1. The patient completes the consultation
2. The system asks the patient to rate the doctor
3. The patient rates the doctor on a scale of 5
4. The patient validates the rating

Alternative scenario:

1. The patient searches for the doctor profile existing
2. The patient views the doctor profile
3. The patient clicks the rate button
4. The patient validates the rating

Use case scenario 2:

Req1:The doctor shall be able to chat with his patients
[Low priority].

Precondition:

- The doctor has accepted the request of the patient

Post-condition:

- the discussion has ended successfully

Normal scenario:

1. The doctor accepts the request of the patient
2. The doctor views the chat page
3. The doctor views the patient 's name and messages
4. The doctor text back the patient
5. The doctor sends images and voice messages

Alternative scenario:

- The doctor searches for the profile of the patient
- The doctor clicks on the message icon in the patient 's profile
- The doctor write the message to the patient
- scenario restarts from step 4

5.5 Design

5.5.1 Sequence Diagram

See Figure 5.3.

5.5.2 Class Diagram

See Figure 5.4.

5.6 Conclusion

As shown in this chapter, the requirements have been met as planned. We provided an interface where patients and doctors can interact together. In addition patients can evaluate the doctors using the doctor recommendation system based on stars.

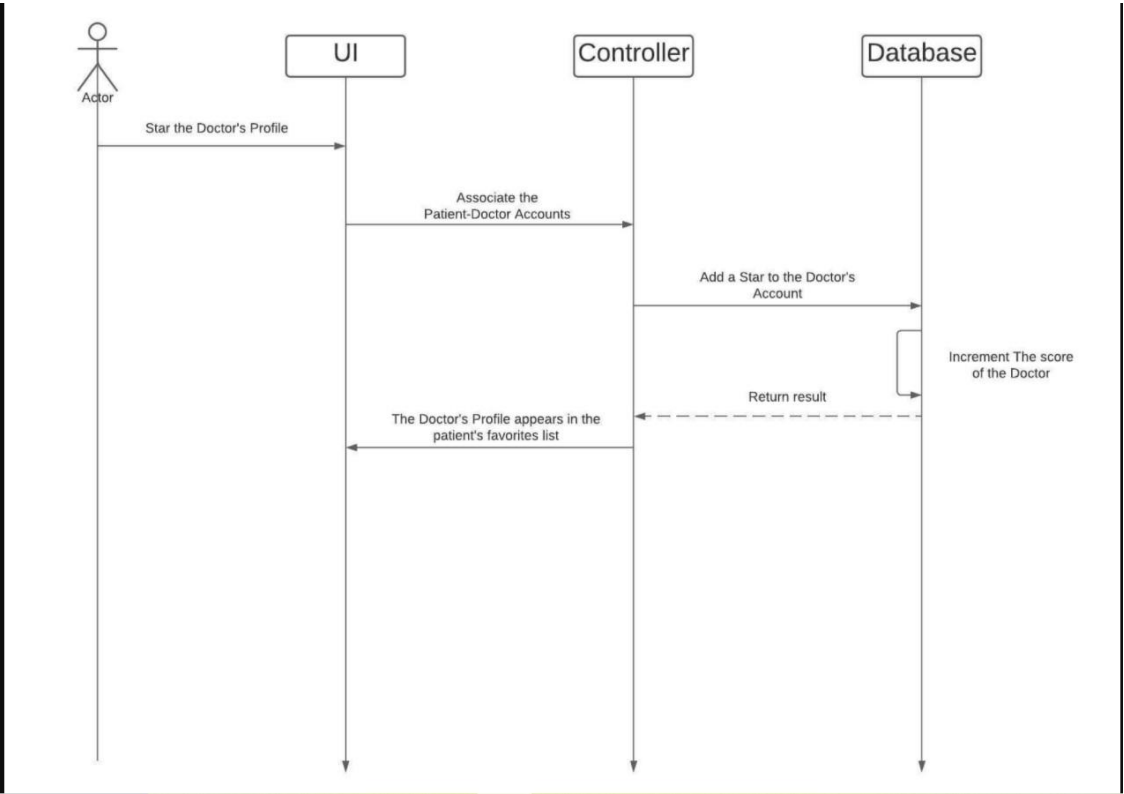


FIGURE 5.3: Sequence Diagram showing how a patient rates a doctor

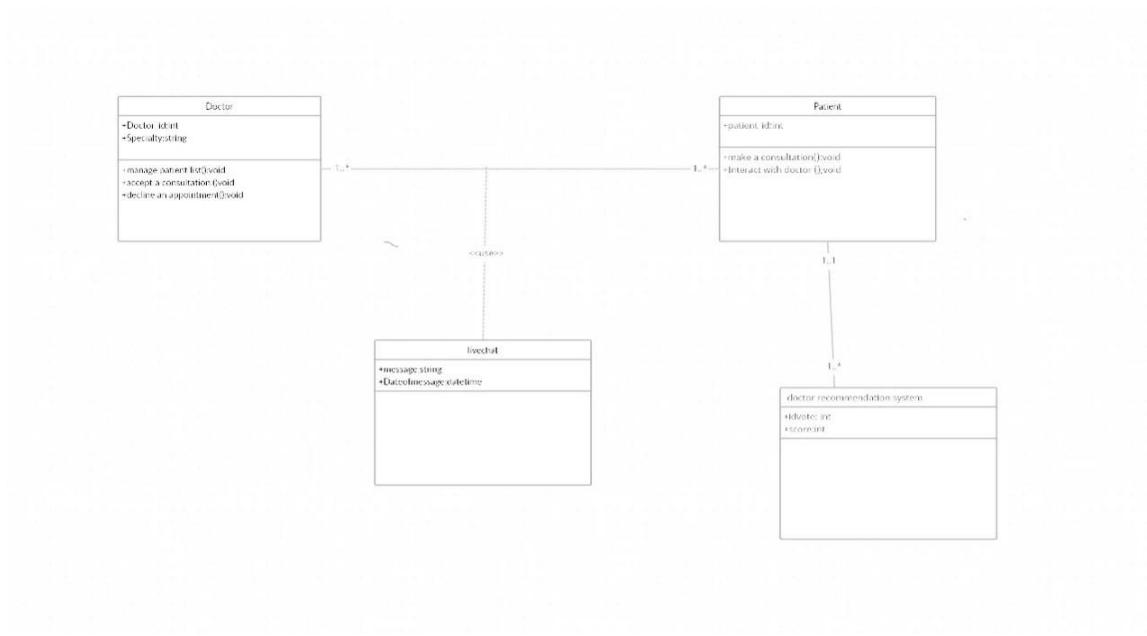


FIGURE 5.4: Class Diagram of the Third Sprint

Bibliography

- Angular.io (n.d.). *angular*. URL: <https://angular.io/>.
- Kruchten, Philippe (2004). *The Rational Unified Process: An Introduction(3rd Ed.)* ISBN: ISBN 0-321-19770-4.
- MySQL (n.d.). *mySQL*. URL: <https://www.mysql.com/>.
- Node.js (n.d.). *node.js*. URL: <https://nodejs.org/en/>.
- PMI (n.d.). *what is project management?* URL: <https://www.pmi.org/about/learn-about-pmi/what-is-project-management>.
- Postman.com (n.d.). *postman*. URL: <https://www.postman.com/>.
- Sabbir M Saleh, M Ashikur Rahman (2017). "Comparative Study on the Software Methodologies for Effective Software Development". In:
- Scrum (n.d.). *what is scrum?* URL: <https://www.scrum.org/resources/what-is-scrum>.
- Synopsys (n.d.). *Top 4 software methodologies*. URL: <https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/>.
- VisualParadigm (n.d.). *What is Unified Modeling Language (UML)?* URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>.
- VisualStudioCode.com (n.d.). *visual studio*. URL: <https://code.visualstudio.com/docs>.