

Democratic and Popular Algerian Republic
Ministry of Higher Education and Scientific Research

Université de BLIDA 1
Faculté des Sciences
Département d'Informatique



FINAL REPORT

Option: Systèmes Informatiques

Web Platform for Medical Services

By:

- Zentici youcef
- Bouchelaghem mohamed
- Chaouch mahdia

Supervised by:

FZ. Zahra

Presented on: may 2025

In front of jury composed of :

Ouahrani leila

Promotion : 2024/2025

Acknowledgement

First of all, we thank God for giving us the strength and knowledge to complete this project.

We would also like to express our sincere gratitude to Miss Zahra for dedicating her time and sharing her knowledge with us.

We are deeply grateful to our families and friends for their unwavering support and encouragement, which provided us with the strength and motivation to complete this work.

Our gratitude extends to the Faculty of Science for the funding opportunity to undertake us
at the Department of Computer Science, University of Saad Dahleb
BLIDA.

Finally, we dedicate this work to the beloved grandmother of our teammate, who has sadly passed away. Her love, wisdom, and support have left a lasting impact, and we honor her memory with this project. May she rest in peace.

Abstract

In this fast-paced world, where access to timely healthcare is crucial, time is incredibly valuable. Our first priority should be to spend it on meaningful activities, not just waiting in long queues for doctor appointments at crowded hospitals.

This is where our **hospital management system** and **online medical booking platform** come in. They are designed for hospitals to manage **medical services** such as **doctor consultations**, **lab test scheduling**, **radiology imaging**, and **surgical operations**.

For patients, our **digital healthcare solution** makes it easy to **book appointments online**, avoid unnecessary delays, and eliminate the need for long, costly travel to access essential care.

Keywords: Hospital management, Online booking, Medical services, Mobile application, Radiology, Telemedicine.

Résumé

Dans ce monde où tout va très vite, le **temps** est une ressource précieuse et inestimable. Notre priorité devrait être de le consacrer à des activités significatives, et non à **attendre notre tour pour une consultation chez le médecin**.

De plus, de nombreux patients doivent parcourir de longues distances pour accéder à des **services de santé**, ce qui peut être fatigant, chronophage et parfois coûteux.

C'est précisément là qu'interviennent notre **application mobile** et notre **site web**. Conçus pour les **hôpitaux**, ils permettent de gérer les **consultations médicales**, les **examens de laboratoire**, les **tests de radiologie** et les **interventions chirurgicales**.

Pour les patients, notre **plateforme de santé en ligne** facilite l'accès à ces services et permet d'éviter les **délais inutiles** et les **déplacements longs**.

Mots-clés: Gestion hospitalière, Réservation en ligne, Services médicaux, Application mobile

Radiologie, Télémédecine

ملخص:

مورداً ثميناً لا يُقدَّر بثمن. يجب أن تكون أولويتنا هي تخصيصه للأنشطة ذات المعنى، في هذا العالم السريع، يُعتبر الوقت وليس فقط للانتظار في الموعد الطبي.

علاوة على ذلك، يضطر العديد من المرضى إلى السفر لمسافات طويلة للوصول إلى الخدمات الصحية، مما قد يكون مرهقاً، ويستغرق وقتاً، وأحياناً يكون مكلفاً.

تم تصميمهما خصيصاً لإدارة الاستشارات الطبية، وموقعنا المخصص للمستشفيات وهنا تأتي أهمية تطبيقنا الإلكتروني والتحليل المخبرية، والفحوصات الإشعاعية، والإجراءات الجراحية.

ومشقة الوصول إلى هذه الخدمات وتقلل من التأخيرات غير الضرورية بالنسبة للمرضى، تسهل منصتنا الصحية الرقمية التنقل

لكلمات المفتاحية: إدارة المستشفيات، الحجز عبر الإنترنت، الخدمات الطبية، التطبيق المحمول، الأشعة، الطب عن بُعد

Table of Contents

General Introduction	1
1.Problematic	1
2. objectives	1
3.Organization of manuscript	1
Chapter 1: Analysis and Conception	2
1.Introduction	2
2. Software Development Process	2
4.Use case diagram	3
4.1 Identification of Actors	3
4.2Identification of use cases:	4
5. Sequence Diagram	6
5.1 Managing a Referral Letter	7
5.2Ticket Request	8
5.3. Upload Result	10
3. Class Diagram	10
6. From Class Diagram to NoSQL database:	11
7. Software Architectural Design Pattern	12
7.1 MVC (Model-View-Controller)	12
7.2 How MVC Was Used in Our Project	12
Chapter 2: Implementation	13
Introduction	13
2.Tools used	13

3.Project Presentation	15
3.1Manage Staff page	15
3.2 Provider Signup Page	16
3.3 Manage Prescription page	17
3.4Manage Results page	18
3.5Referral Submission for Surgery Request page	19
4. Study of Existence	21
4.1Market study	21
4.2Target Market Analysis	21
4.3Analysis of the Offer and Competition	22
4.Economic and Environmental Analysis	23
Conclusion	24
General Conclusion and Perspectives	25
References	25

List of Figures

Figure 1 : Use cases diagram	6
Figure 2 : Sequence diagram for the use case « Manage referral letter »	8
Figure 3 : Sequence diagram of “Ticket Request” use case.	9
Figure 4 : sequence diagram for “upload result” use case.	10
Figure 5 : Class diagram	11
Figure 6 : Manage Staff interface.	16
Figure 7 : Provider Sign up interface	17
Figure 8 : Manage Prescription interface	18
Figure 9 : Manage Results interface	18
Figure 10 : Referral Submission for Surgery Request	20
Figure 11 : Selection of the Medical Facility	21
Figure 12 : Request Confirmation	21

General Introduction

The biggest leap in the history of technology and the healthcare industry in Algeria is the development of online medical services. Creating a platform for consultations, laboratory and radiology tests, and surgical consultations is expected to improve patient accessibility, efficiency, and overall experience.

1. Problematic

How can digital health platforms, by leveraging a wide range of modern technologies—such as mobile applications, online booking systems, telemedicine tools, and secure data exchange protocols—help reduce patient waiting times, minimize unnecessary travel, and improve the accessibility, efficiency, and overall quality of healthcare services, while ensuring strict compliance with data security and privacy regulations?

2. objectives

Here are several reasons why online consultations and test bookings are so important:

- **Availability:** the patients can book consultations and tests at any time, regardless of office hours. This is particularly important for urgent medical advice or non-emergency concerns.
- **Gain Time:** You can manage your time and do other things instead of just staying in the waiting room for a 10-minute consultation.
- **Easy Test Bookings:** Patients can schedule lab tests or radiology appointments at their convenience, reducing the likelihood of missed appointments and streamlining healthcare processes.
- **Data Security and Privacy:** With the proper safeguards, online medical consultations booking and test booking platforms can ensure that patient information is secure, private, and in compliance with health data regulations such as HIPAA or GDPR.

3. Organization of manuscript

Our dissertation consists of four sections, organized as follows:

- First, we conducted a market study on the healthcare industry and its technological aspects.
- Second, we discuss the design and modeling of our work using the UML language.
- Before concluding, we describe our website as well as the development tools used.

Chapter 1: Analysis and Conception

1. Introduction

The main goal of a market study is to identify and understand the requirements of a project or an application. It starts with a detailed identification of each requirement. The main objective of this analysis is to ensure that the solutions proposed are in line with expectations, and that they are realistic and make the best use of available resources and efforts to ensure effective implementation.

2. Software Development Process

The software development process represents the set of essential activities required to create a software product, such as requirements analysis, design, development, testing, deployment, and maintenance [1].

To structure these stages, various development models have been designed, each adapted to specific project contexts and constraints [1], [2].

In the context of this work, we have chosen to adopt the **waterfall model**, a sequential and linear model in which each phase must be fully completed before moving on to the next. [2]

3. Adopted Methodology: Bottom-Up Approach

In the design phase of our medical application, we adopted the Bottom-Up development approach. This method consists of designing and developing basic components first, then assembling and integrating them to form more complex modules, leading to the full architecture of the application [1].

This methodology offers several advantages [1], [2]:

- It allows the project to be divided into small, independent parts, making development more manageable.
- Each component can be developed and tested separately, which simplifies the detection and correction of errors.
- It encourages component reuse, a key aspect of modular and scalable systems.
- It promotes a step-by-step understanding of the system by constructing it piece by piece.

For example, In the authentication functionality, we first designed independent functions such as registration, login, and password reset before integrating them into a complete authentication module.

This approach is well-suited to our project because it offers significant flexibility and ensures a solid foundation for assembling various medical functionalities (appointment booking, results management, prescription sending, etc.).

4. Use case diagram

Use case diagrams describe the interactions between the system and its actors. They show what the system is expected to do and how the actors interact with it, without specifying how the system works internally. These diagrams are commonly used during the requirements analysis phase to give a clear, functional overview of the system. [3]

4.1 Identification of Actors

admin:

it is the person who manage service providers requests to create an account and join our platform.

Private Doctor Manager:

It is the person who have the ability to manage the account of every agent in the cabinet.

Hospital/Clinic Manager:

It is the person who have the ability to manage the account of every agent in the Hospital/Clinic and overseeing the services provided there.

Department Chef:

it is the person responsible for handling the referral letters sent by clients.

Agent:

This person is responsible for managing the consultation tickets.

Laboratory agent:

This person is responsible for managing the laboratory appointment tickets, handling prescription letters sent by clients, and delivering the results of laboratory tests.

Radio agent:

This person is responsible for managing the radiology appointment tickets, handling prescription letters sent by clients, and delivering the results of radiology tests.

Client:

The client (patient) is the person who have the ability to request tickets for consultations, radiology tests, and laboratory tests. They will also be able to receive their results, request an operation consultation, and submit a referral letter for review.

4.2 Identification of use cases:

Actors	Uses cas
Admin	Authentication Manage accounts
Private Doctor Manager	Authentication Manage stuff
Hospital/Clinic Manager	Authentication Manage service Manage stuff
Department Chef	Manage referral letters Accept Reject Send notification about appointment Send a reason of rejection Accept Reject Send notification about appointment Send a reason of rejection Authentication
Agent	Authentication Manage ticket
Labo/radio agent	Manage ticket Manage prescriptions Accept Reject Send notification about appointment Send a reason of rejection

	Authentication
Client	Authentication Request consultation Request a radio test Get result Request operation consultation

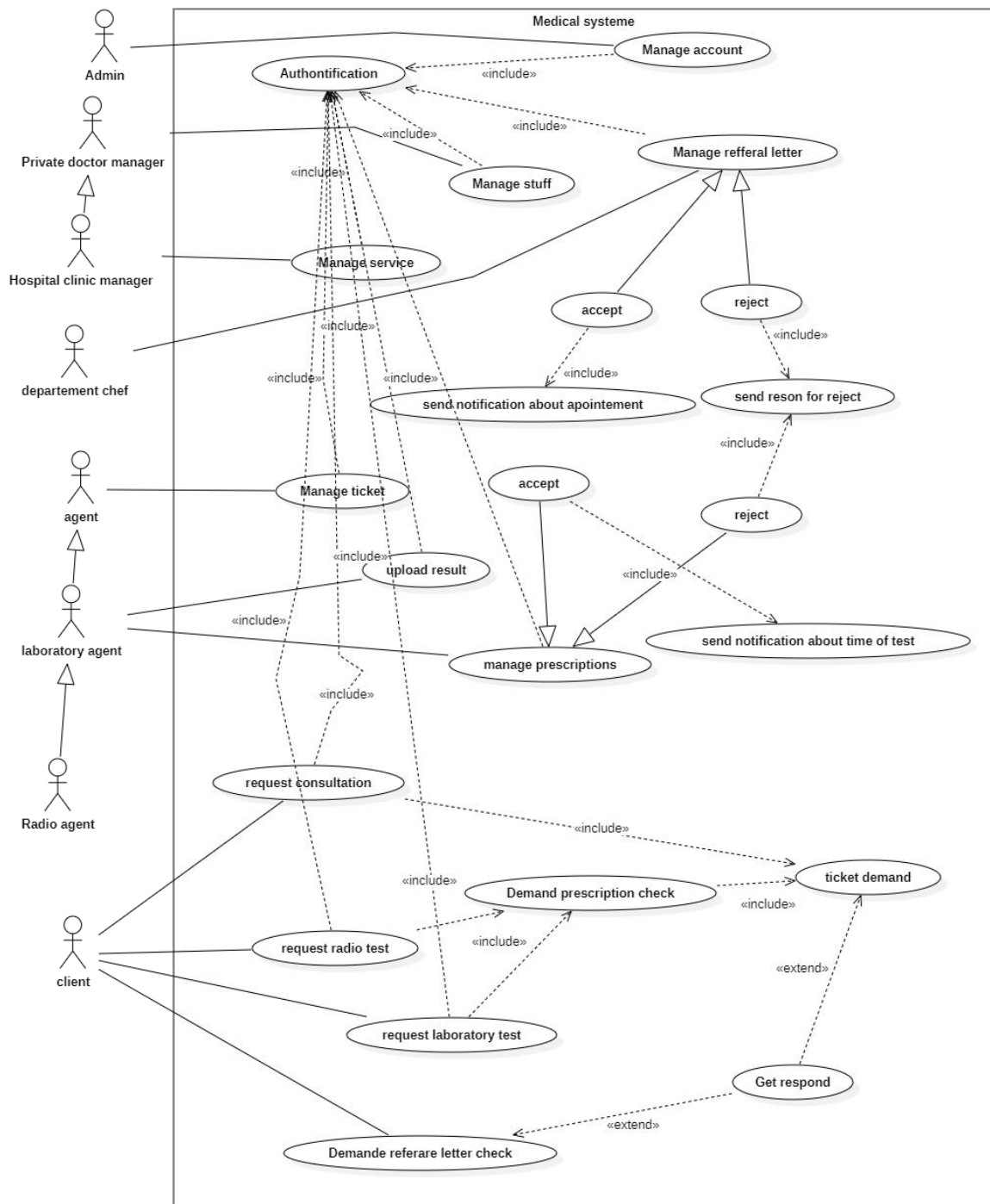


Figure 1 : Use cases diagram

5. Sequence Diagram

A sequence diagram is a type of UML (Unified Modeling Language) diagram that illustrates the interaction between different objects in a system, arranged in chronological order based on the sequence of messages exchanged. It shows how objects interact with each other through messages such as method calls, responses, and data exchanges during a specific scenario, thereby highlighting the temporal flow of events [3].

In this section, we present the main sequence diagrams used to model the interactions between the various actors and components of the system. These diagrams are especially useful for detailing the dynamic behavior of use cases and helping developers understand how functionalities unfold over time.

5.1 Managing a Referral Letter

The process of managing referral letters by the department chief begins with authentication in the system. Once verified, the system grants them access. The department chief can then view the list of received referral letters, select one, and read its content. After reviewing and understanding the letter, the chief decides whether to accept or reject the patient.

- If accepted, they propose a date for the consultation appointment and enter it into the system.
- If rejected, they must provide a justification for the rejection and send it to the patient.

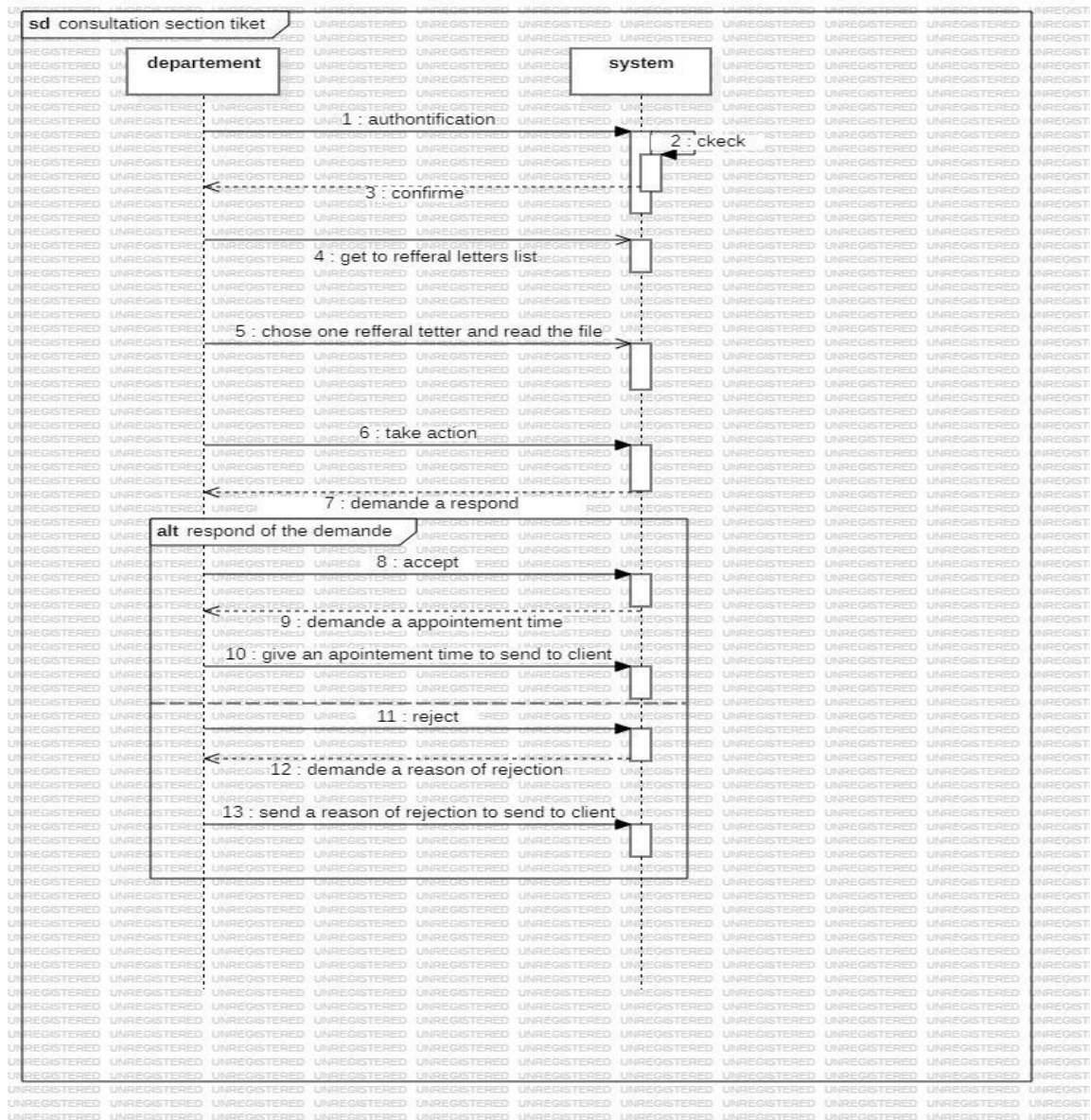


Figure 2 :Sequence diagram for the use case « Manage referral letter »

5.2 Ticket Request

The client authenticates in the system, which verifies their information and confirms their identity. Once authenticated, the client accesses the consultation section, where the system presents them with various actions. The client then chooses a service, and the system prompts them to select the type of establishment from three options: hospital, clinic, or private practice. Next, the client must choose a wilaya. The system checks if establishments matching the selected type and offering the chosen service are available in the specified wilaya. If such establishments are found, a list is displayed for the client to choose from, after which a consultation ticket is generated. If no matching establishments are found, an error message is displayed.

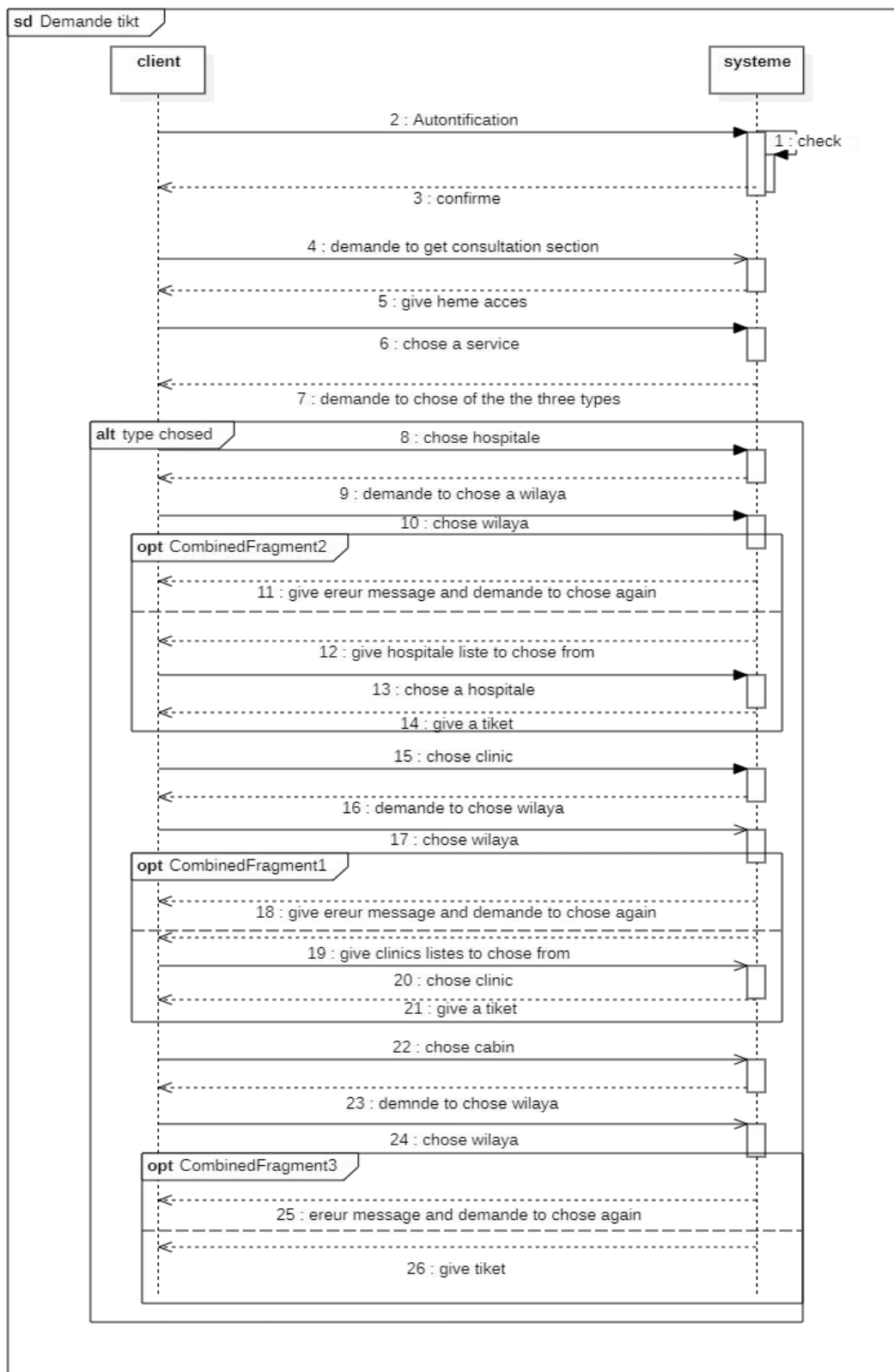


Figure 3 :Sequence diagram of “Ticket Request” use case.

5.3. Upload Result

In this scenario, the laboratory agent begins by accessing the system's authentication interface. They enter their credentials (username and password), and the system proceeds to verify the provided information. If the information is correct, access is granted to the agent. The agent can then upload the client's test results through the designated interface. Once the results are saved, the system automatically sends a notification to the client to inform them that their results are available. The client, in turn, accesses the authentication interface, enters their credentials, and the system once again verifies the information. If valid, access is granted, allowing the client to securely view their results from their personal space.

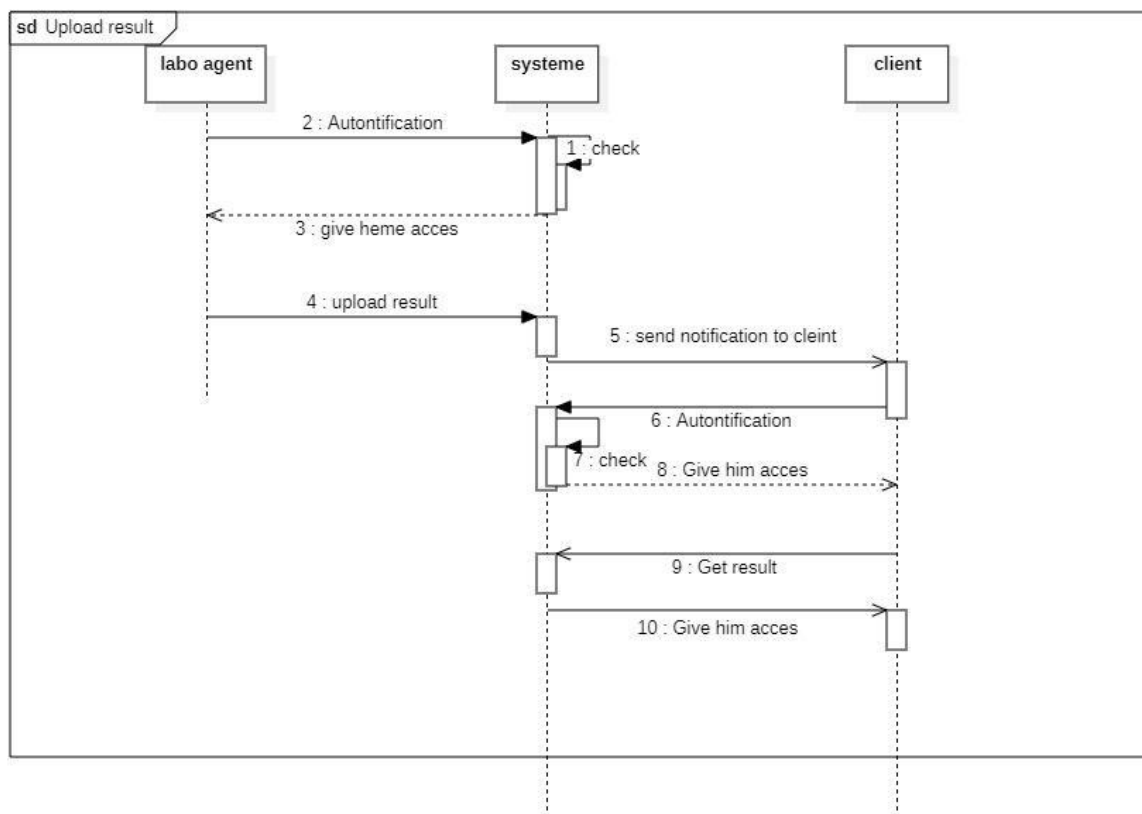


Figure 4 :sequence diagram for “upload result” use case.

3. Class Diagram

The class diagram represents the static structure of the system by defining the objects that compose it, the relationships between these objects, as well as the attributes and operations specific to each class. This model provides the essential foundation for understanding changes and interactions within the system. Indeed, changes or interactions can only occur if there are concrete elements to be modified or made to interact.[3]

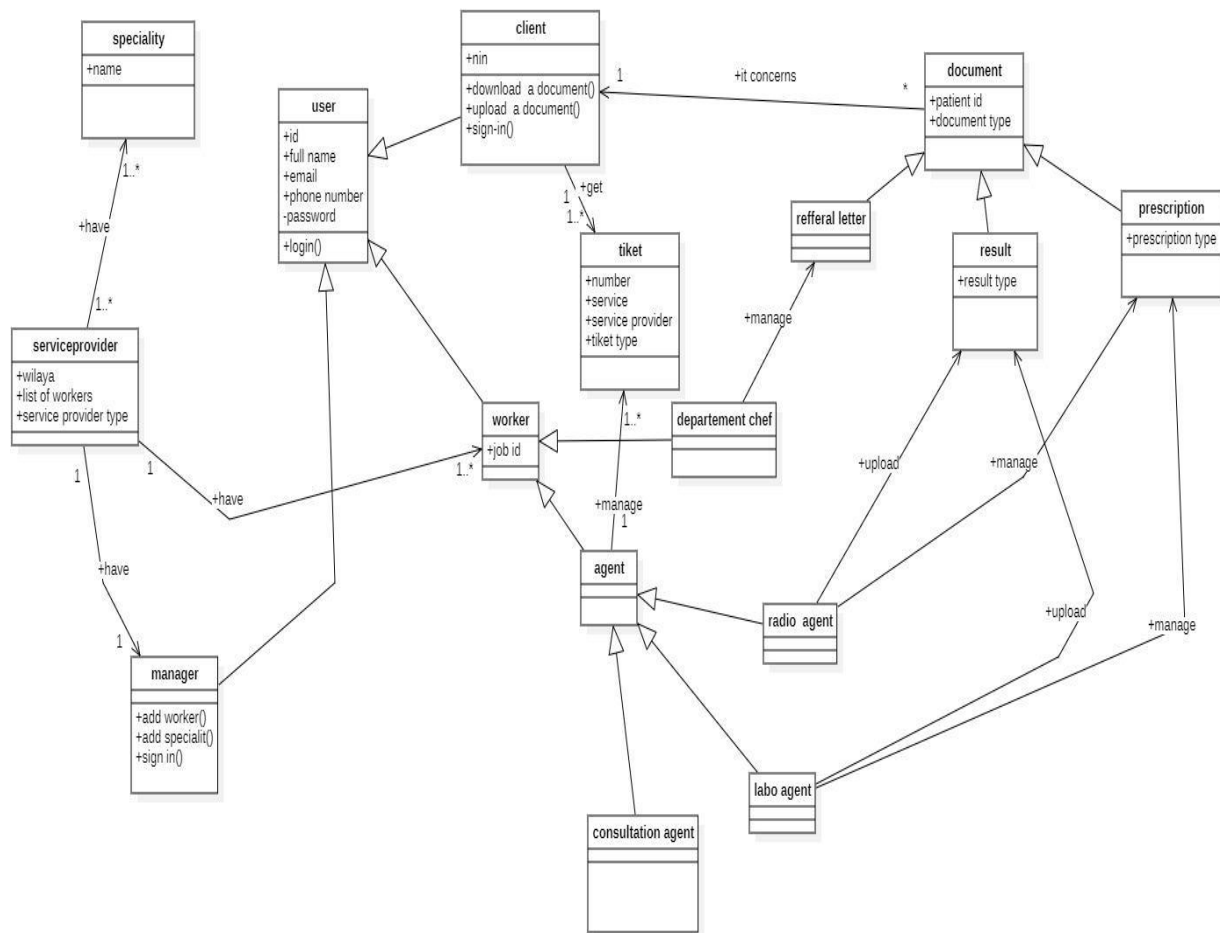


Figure 5 :Class diagram

6. From Class Diagram to NoSQL database:

Identify Classes as Collections

- ✓ Each main class becomes a collection (e.g., user, client, ticket, etc.) [4].

Attributes Become Fields

- ✓ All attributes inside classes are fields in the document [4].

Model Relationships

- ✓ 1:1 or 1:* → Reference (store _id of related document) [5].
- ✓ 1:* or : with low cardinality → Embedded document (nest the object). [5]
- : with high cardinality → Use intermediate collections or arrays of references. [5]

Inheritance:

- Use a type field in a base collection or separate collections for each subclass. [4]

Methods :

- Ignore methods in the schema, but implement them in the application logic.[5]

7. Software Architectural Design Pattern

A Software Architectural Design Pattern is a general, reusable solution to a common structural or interaction problem in software architecture. It defines how software components should be organized and how they should interact with one another to achieve specific qualities such as scalability, maintainability, and flexibility. Unlike broader architectural styles, a design pattern typically focuses on internal structure and communication between components within a module or layer [5]. In our project, we adopted the Model-View-Controller (MVC) architectural design pattern, which we will define in detail in the following section [6].

7.1 MVC (Model-View-Controller)

Model-View-Controller (MVC) is a software architectural design pattern that divides an application into three main components:

- **Model:** Manages the data, logic, and rules of the application. It represents the business layer.
- **View:** Handles the display of information (the user interface). It presents data from the model to the user.
- **Controller:** Responds to user input, manipulates the model, and updates the view accordingly.

This separation of concerns increases modularity, making applications easier to maintain, scale, and test [6], [7].

7.2 How MVC Was Used in Our Project

In our project, we applied the MVC pattern in the following way:

- **Model:** We used the classes from our UML diagram (such as User, Client, Ticket, Document, etc.) to design the MongoDB data models, which represent the business logic and data.
- **View:** The front-end interfaces we developed (like the ticket system dashboard, document upload forms, etc.) represent the View part of the pattern.
- **Controller:** The logic responsible for handling requests (such as creating or managing tickets, uploading documents, or assigning roles) was implemented in the Controller layer to manage communication between the View and the Model.

By using MVC, we ensured a clean separation of concerns, allowing us to:

- Build and test each component individually,
- Maintain and update specific parts of the application without affecting the whole system,
- Collaborate effectively, as responsibilities were clearly defined.

Chapter 2: Implementation

Introduction

This section aims to present the main steps necessary for constructing an interface. It focuses on the essential stages of building a successful user interface. To do this, it is necessary to use an appropriate set of tools, software, and programming languages to carry out various technical tasks. These tasks include creating a database to organize information, setting up APIs to ensure communication between the back-end and the front-end, and configuring the servers required for the application to function properly. Finally, screenshots are taken to illustrate the final result of the web application as it will appear to the user.

1.Tools used

HTML

HTML (HyperText Markup Language) is a standard markup language used to structure the content of web pages. It allows developers to define elements such as headings, paragraphs, images, or videos using tags and attributes. Web browsers interpret this code to display pages in a readable and interactive way for the user [8].

CSS

CSS (feuilles de style en cascade) is a computer language designed to format and style web pages written in HTML or XML. This language relies on files called style sheets, which use rules to control fonts, colors, margins, and other aspects of page elements [9].

React

React is a JavaScript library that allows you to build user interfaces from components. These components can then be combined to create screens, pages, and even complete applications [10].

Node.js

is an open-source, cross-platform JavaScript runtime environment that allows developers to create network applications, servers, and command-line tools. Based on Google Chrome's V8 JavaScript engine, Node.js is particularly well-suited for applications that require fast, non-blocking execution, such as real-time applications, chat systems, and APIs[11].

Express.js

Express.js is a minimalist and flexible framework for Node.js, designed to simplify the development of web and mobile applications. It provides a robust set of features to handle HTTP requests, route management, cookie management, and middleware integration. Express allows for the creation of dynamic web servers and RESTful APIs in a quick and structured manner. This framework is often used to build large-scale web applications, as it enables the development of efficient and well-structured applications while minimizing the necessary code[11].

StarUML

Staruml is a software modeling tool used primarily in analysis and design, to create simple models and facilitate project management[12].

Visual Studio Code (VS Code)

is a lightweight, powerful, and open-source code editor developed by Microsoft. It supports multiple programming languages such as JavaScript, Python, C++, and many others, through its plugin system. It is designed to be highly customizable and offers features such as code intelligence (IntelliSense), integrated debugging, project management, and an integrated terminal. VS Code is popular for its intuitive user interface and extensions, which allow it to be adapted to a wide variety of development need.[13].

GitHub

GitHub is a website and service used by developers to store and manage their code, as well as track and control changes to it[14].

MongoDB

MongoDB is a NoSQL database that stores data as BSON documents. It offers a scalable and extensible structure, making it particularly suitable for modern applications requiring efficient, flexible, and scalable data management[15].

API

An API is a set of definitions and protocols that allows the connection between the front-end and the back-end, facilitating the exchange of data between the user interface and the server. I can use this in my program[16].

3.Project Presentation

In this section, we present several screenshots showcasing the main interfaces of our project. These interfaces demonstrate how users interact with the system, emphasizing ease of use and smooth navigation within the platform designed for managing medical services.

3.1 Manage Staff page

This page allows the manager to add a new nurse to the establishment. It contains a structured form in which the manager can enter essential information about the nurse.

The form includes the following fields:

- Job ID
- Full name
- Contact details (phone number, email, etc.)
- Specialty or department (e.g., general care, emergency, pediatrics, etc.)
- Other relevant data for their integration into the system.

Once the information is entered and the form is validated, the nurse is added to the database and appears in the list of nursing staff.

This feature simplifies the management of nursing resources and helps maintain an up-to-date medical staff hierarchy in a clear and efficient manner.

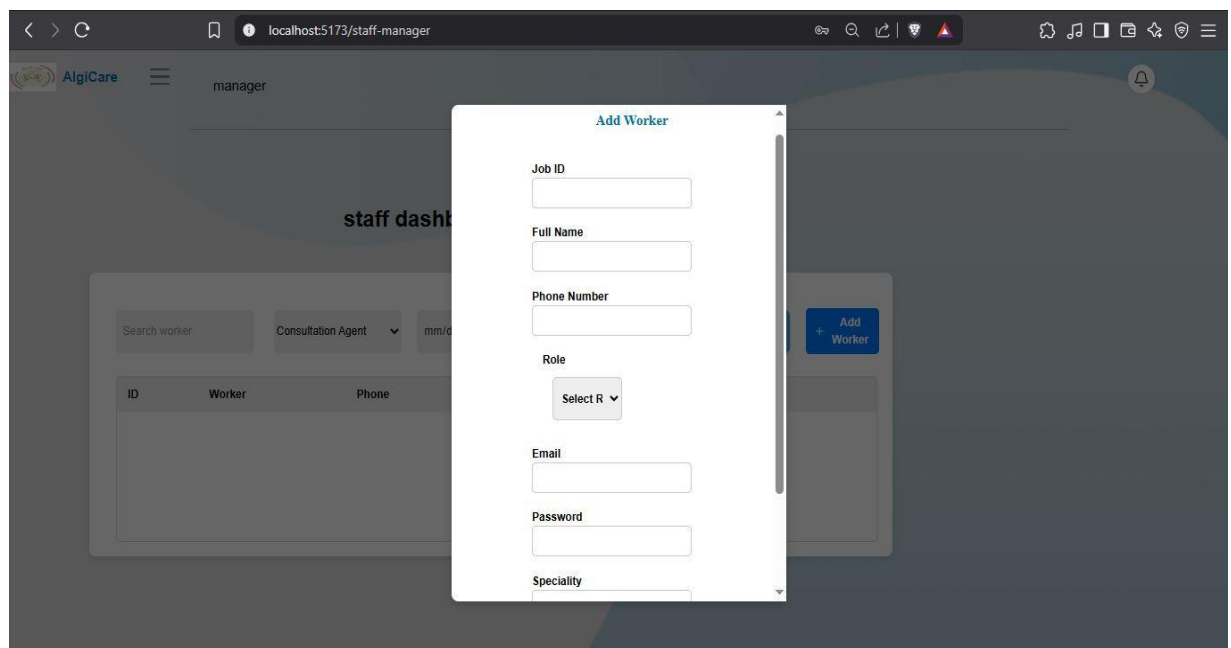
The image shows a web browser window with the URL 'localhost:5173/staff-manager'. The page title is 'manager'. A modal form titled 'Add Worker' is open in the center. The form contains the following fields: 'Job ID' (text input), 'Full Name' (text input), 'Phone Number' (text input), 'Role' (dropdown menu with 'Select R' visible), 'Email' (text input), 'Password' (text input), and 'Speciality' (dropdown menu). In the background, a 'staff dash' is visible with a search bar, a 'Consultation Agent' dropdown, and a table with columns 'ID', 'Worker', and 'Phone'. A blue '+ Add Worker' button is also visible on the right side of the background interface.

Figure 6 :Manage Staff interface.

3.2 Provider Signup Page

This page allows healthcare providers (such as hospitals, clinics, or private medical practices) to register on the platform. When filling out the form, the user must provide several pieces of information, including:

- Wilaya
- Director ID
- Type of structure (hospital, clinic, or private practice)
- Contact details

Depending on the type of structure selected:

- If the provider is a hospital or clinic, an additional field appears to select the available medical specialties.
- If it is a private practice, the specialty field does not appear, as services are generally limited to a single practitioner.

Once the form is validated, the registration request is automatically submitted to the platform administrator, who must approve or reject the request. The provider will only appear on the platform after this validation.

Figure 7 :Provider Sign up interface

3.3 Manage Prescription page

When an agent (from the laboratory or radiology department) accesses the page, they can view a list of pending requests awaiting processing.

For each request, the agent has two options:

- **Accept the request**
→ In this case, they are prompted to provide an appointment date for the patient.
- **Reject the request**
→ In this case, they must specify a reason for the rejection.

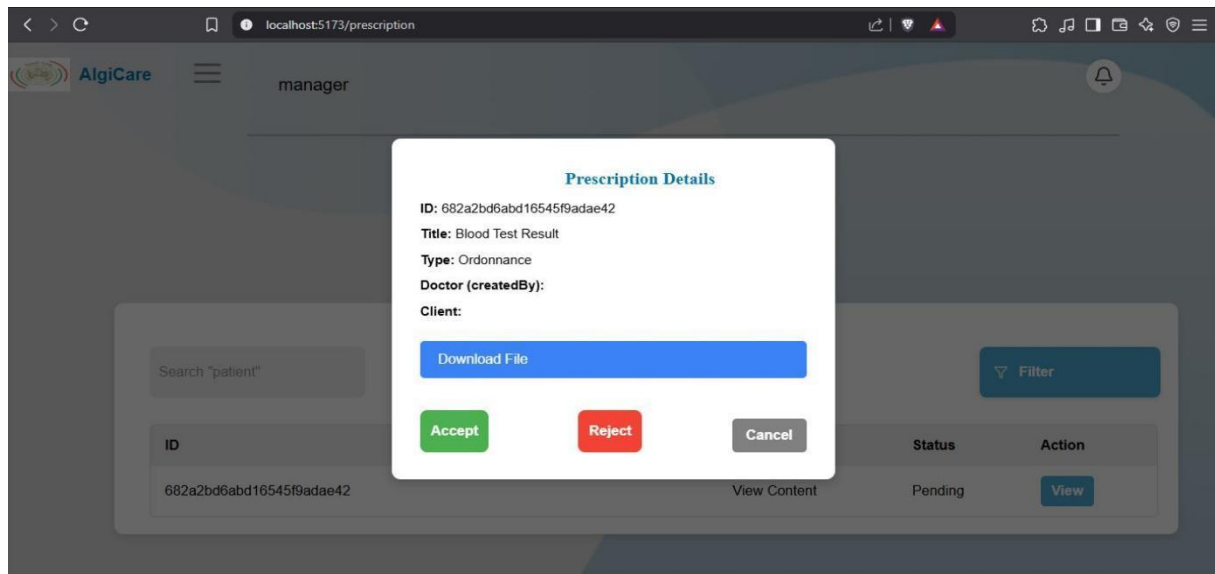


Figure 8 :Manage Prescription interface

3.4Manage Results page

When an agent accesses the page, they first see a summary table of previously uploaded results.

If the agent wishes to add a new result, they simply need to click on the "Add Result" button. A form will then appear, allowing them to:

1. Fill in the patient's ID and the agent's ID;
2. Upload the examination or analysis result file intended to be sent to the patient;
3. Confirm the submission, so that the document is correctly saved and added to the patient's medical record.

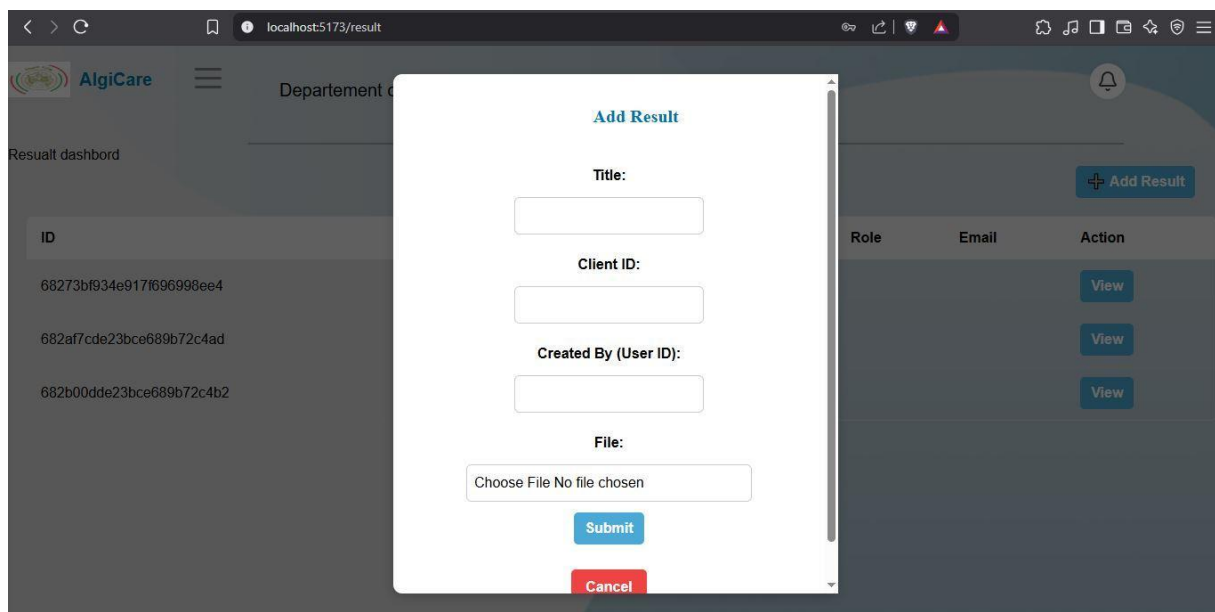


Figure 9 :Manage Results interface

3.5Referral Submission for Surgery Request page

This interface allows patients to submit a medical appointment request by following these steps:

1. **Specialty and Referral Letter:**

The patient first selects the desired medical specialty (e.g., cardiology, orthopedics...) and uploads their referral letter or medical file.

2. **Location and Type of Facility:**

Next, the patient chooses the *wilaya* (region) and the type of medical facility: hospital, clinic, or private practice.

3. **Search and Selection of Facility:**


By clicking the “Search” button, a list of facilities matching the selected criteria is displayed. The patient can then choose the one that suits them best.


4. **Request Confirmation:**


When the patient clicks “Book Appointment,” a confirmation message appears, indicating that the referral letter has been successfully sent to the selected facility. The medical center will review the request and follow up accordingly.


This process ensures a clear and efficient management of medical requests while facilitating the transmission of required documents.


Choose a speciality for operation


Pediatrics



Neurology


Ophthalmology



Dentistry


Endocrinology

Upload referral letter


Click to upload
Upload your operation prescription
Supported formats: PDF

Uploaded Files (1)


 referral letter.pdf


View


Choose a State

Oran

Choose Type


Clinic


Hospital


Cabinet

Search

Figure 10 :Referral Submission for Surgery Request

19

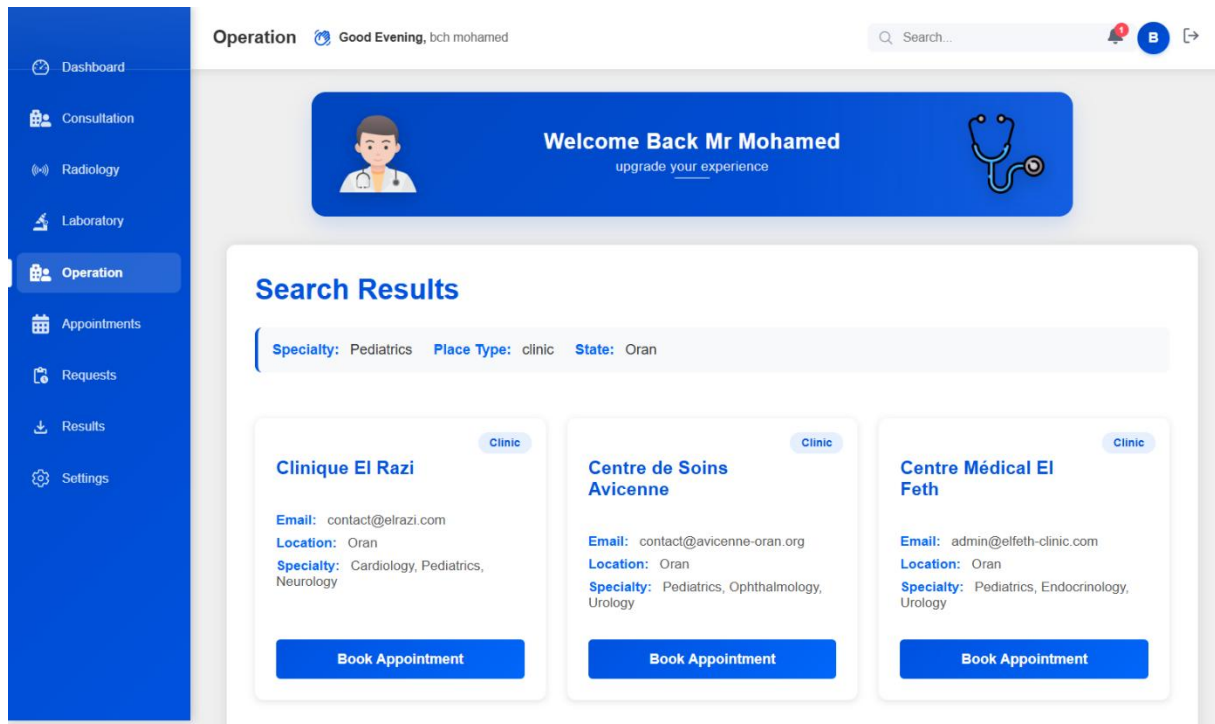


Figure 11 :Selection of the Medical Facility

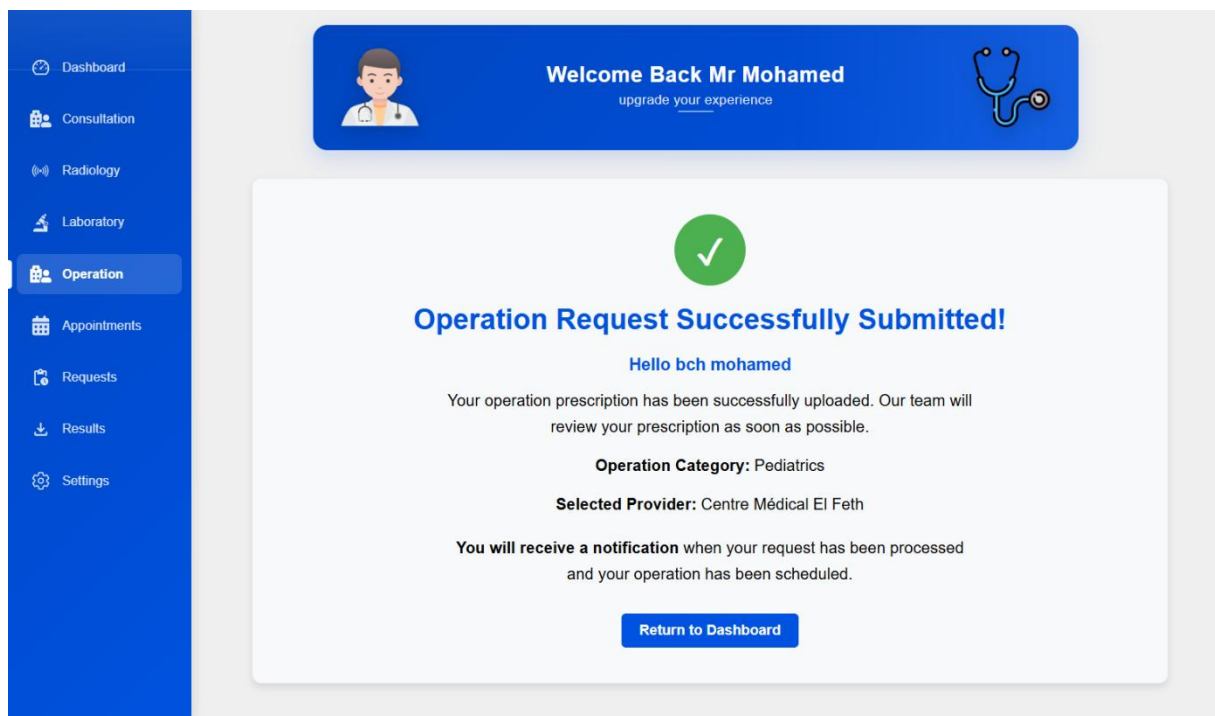


Figure 12 :Request Confirmation

4. Study of Existence

4.1Market study

4.1.1 Introduction

Before launching this website, it is necessary to analyze the market and identify similar services to ensure the success of our project in the future. This project aims to conduct a market study for an application that allows clients to book consultation appointments online, request operations, and schedule appointments for radiology and laboratory tests.

4.2Target Market Analysis

4.2.1Definition

Our targeted market of our project is to understand the patients that search for reduce the waiting time and take appointments easy and fast .and also for medical service workers and search for ameliorate the gestion of appointments and their consultation

We have identified three main users in our website:

The patient: search fast ways for taking appoint ents

Workers of the medical service: search for ameliorate the gestion of consultation appointment.

The minister: manage service provider account demand and that help him to manage the medical system

4.2.2Market Segmentation

We have divided our market based on three criteria:

Demographic criteria: our project is directed to for all ages and all genders.

Geographic criteria: our project target all Algerians.

Psychographic criteria: Our project is designed for individuals seeking convenient and efficient medical consultation and testing services."

4.2.3User needs and expectations

The market study has validated the requirements of the application's potential users:

1-Fast and easy access to the platform.

2-Secure data base.

3-User-friendly interface.

4-Convenient appointment booking

4.2.4Market trends

The transformation of healthcare to a digital system has increased the chances of our application succeeding

4.3Analysis of the Offer and Competition

4.3.1Identification of Main Competitors

The digital healthcare market includes several important players that offer services similar to those of **Algicare**:

1 Altibbi (<https://www.altibbi.com>)

Provides online medical consultations via video calls and a smart medical chatbot. This platform is widely used in the Arab world and connects patients with certified doctors [17].

2 Doctolib (Algeria) – via [SantéUp Home Care](#)

Offers home consultations provided by general practitioners, specialists, and nurses approved by the Ministry of Health [18].

3 MyQuest – by Quest Diagnostics (<https://myquest.questdiagnostics.com>)

1. An American platform that allows patients to securely access their lab test results online, supporting confidentiality and digital patient engagement [19].

4.3.2 Comparative Feature Analysis

The table below compares the main features of **Algicare** with those of its competitors

Feature	Our website Algicare	<u>MyQuest</u>	Altibbi	Doctolib
Free	✓	✗	✗	✗
Lab Appointment	✓	✗	✗	✗
Radiology Appointment	✓	✗	✗	✗
Consultation Appointment	✓	✗	✗	✓
Surgical Appointment	✓	✗	✗	✗
Lab Results Access	✓		✗	✗
Radiology Results Access	✓	✗	✗	✗
Online consultation	✗	✗	✓	✗
Operation Results Access	✓	✗	✗	✗

This comparison shows that **Algicare** provides a more **comprehensive and integrated offering**, combining functionalities that are spread across multiple services in other platforms.

4.Economic and Environmental Analysis

4.1Economic Analysis

The **Algicare** platform is built using **open-source technologies**, including:

- **React** for the user interface
- **Node.js** and **Express** for backend development,
- **MongoDB** for the database system.

These technology choices allow:

- A significant **reduction in development and maintenance costs**,
- Better **scalability and flexibility** of the platform,
- Enhanced **management of medical records and appointment systems**, which helps reduce administrative workload in clinics and improves overall healthcare service efficiency [20].

4.2.Environmental Analysis

The digitalization of medical services such as test results, prescriptions, and appointments has several environmental benefits:

- **Reduction in paper usage** in clinics and hospitals,
- **Fewer in-person visits**, thanks to online access to information,
- A decrease in **carbon emissions** associated with unnecessary travel and physical documentation.

In addition, adopting **eco-friendly (green) hosting** — such as cloud data centers powered by renewable energy — could enhance the platform’s positive environmental impact [21][22].

Conclusion

Algicare stands out from its competitors by offering a broader range of medical services in a single, free-to-use platform. Its use of cost-effective open-source technologies and commitment to digital sustainability make it a strong contender in the evolving healthcare sector. With features designed to optimize patient experience and reduce environmental impact, it addresses current demands in both healthcare and eco-responsibility.

General Conclusion and Perspectives

As part of our project, we designed and developed a medical website aimed at facilitating and modernizing healthcare services. This system is built on a comprehensive web architecture, featuring a user interface developed with HTML, CSS, and React, and a robust backend using Node.js, Express.js, and MongoDB, with API management ensuring smooth communication between the different application layers.

The needs analysis, modeled through a use case diagram, helped us better understand the interactions between the system's main stakeholders—patients, medical staff, laboratories, department heads, and managers—as well as the essential functionalities such as account management, authentication, appointment scheduling, referral letter management, medical test requests and processing, and result delivery.

The main objective of this system is to improve access to healthcare, streamline communication among the various stakeholders, and reduce processing times. The methodology we adopted—combining a waterfall approach with a bottom-up design—enabled a structured progression of the project, ensuring better control over each development phase.

Looking ahead, several perspectives for improvement and expansion can be considered. Developing a mobile application would greatly enhance accessibility for both patients and medical staff. Integrating artificial intelligence could assist in prioritizing urgent cases or recommending tests based on symptoms. Adding real-time communication features, such as live chat or video consultations, could promote the adoption of telemedicine. From a technical standpoint, migrating to a cloud infrastructure would improve scalability and data availability. Lastly, future updates will incorporate user feedback to enhance the user experience and ensure full compliance with evolving data protection regulations in the healthcare sector.

References

- [1] R. S. Pressman and B. R. Maxim, **Software Engineering: A Practitioner's Approach**, 8th ed., New York: McGraw-Hill, 2014.
- [2] I. Sommerville, **Software Engineering**, 10th ed., Boston: Pearson, 2015.
- [3] G. Booch, J. Rumbaugh, and I. Jacobson, **The Unified Modeling Language User Guide**, 2nd ed., Addison-Wesley, 2005.
- [4] K. Chodorow, *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*, 2nd ed. Beijing: O'Reilly Media, 2013.
- [5] MongoDB Manual, "Data Modeling," [Online]. Available: <https://docs.mongodb.com/manual/core/data-modeling/>. [Accessed: 21-May-2025].
- [6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Reading, MA: Addison-Wesley, 1994.
- [7] F. Buschmann, R. Meunier, H. Rohnert, and P. Sommerlad, *Pattern-Oriented Software Architecture: A System of Patterns*, Chichester, UK: Wiley, 1996.
- [8] J. Keith, *HTML5 for Web Designers*, A Book Apart, 2010.
- [9] CSS, *AT Internet Glossary*, [Online]. Available: <https://www.atinternet.com/glossaire/css/>. [Accessed: 21-May-2025].
- [10] React, *React Documentation*, [Online]. Available: <https://fr.react.dev/>. [Accessed: 21-May-2025].
- [11] Express, *Express Documentation*, [Online]. Available: <https://expressjs.com/>. [Accessed: 21-May-2025].
- [12] StarUML Documentation, [Online]. Available: <https://docs.staruml.io>. [Accessed: 21-May-2025].
- [13] Visual Studio Code, Microsoft, [Online]. Available: <https://code.visualstudio.com/>. [Accessed: 21-May-2025].
- [14] Kinsta, *Base de connaissances GitHub*, [Online]. Available: <https://kinsta.com/fr/base-de-connaissances/base-de-connaissances-github/>. [Accessed: 21-May-2025].
- [15] MongoDB, **Documents — MongoDB Manual**, [Online]. Available: <https://www.mongodb.com/docs/manual/core/document>. [Accessed: 21-May-2025].
- [16] MDN Web Docs, "API - Glossary," Mozilla, [Online]. Available: <https://developer.mozilla.org/fr/docs/Glossary/API>. [Accessed: May 22, 2025].
- [17] Altibbi, "Altibbi - Medical consultations and information," [Online]. Available: <https://www.altibbi.com>. [Accessed: May 21, 2025].

- [18] SantéUp Soins à Domicile, “Doctolib Algeria – Health services at home,” [Online]. Available: <https://www.santeup-soinadomicile.com>. [Accessed: May 21, 2025].
- [19] Quest Diagnostics, “MyQuest: Secure access to lab results,” [Online]. Available: <https://myquest.questdiagnostics.com>. [Accessed: May 21, 2025].
- [20] J. Brown and L. Keller, “Open-source technologies in healthcare: Cost efficiency and scalability,” *International Journal of Health Informatics*, vol. 15, no. 2, pp. 88–97, 2022.
- [21] World Health Organization, “Digital health and the environment: A sustainable approach,” WHO Report, Geneva, 2021. [Online]. Available: <https://www.who.int/publications/i/item/9789240020924>
- [22] European Commission, “Green Data Centres and Cloud Sustainability,” *Digital Strategy Report*, 2023. [Online]. Available: <https://digital-strategy.ec.europa.eu>