الجمهورية الجزائرية الديمقراطية الشعبية
The People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Resea
جامعة محمد بوضياف بالمسيلة
University Mohamed Boudiaf of M'sila

كلية الرياضيات والإعلام الآلي
Faculty of Mathematics and Informatics

قسم الإعلام الآلي
Department of Computer Science

Domain: Mathematics and Computer Science

Thesis Presented to Fulfill the Partial Requirement
for Master's Degree in Computer Science

Specialty: information system and software engineering

Prepared By: Ben Dahmane Dhiya El Haq , Kouriche Oussama

Supervised By:

Debbi Hichem

ENTITLED

A Web-based Platform for Healthcare with Different Services

Jury Members

Mokhtari Rabeh	President
Debbi Hichem	Supervisor
Chalabi NourElhouda	Examiner

Academic Year 2023/2024

Acknowledgment

First and foremost, heartfelt gratitude and praises go to the Almighty Allah who guided me through and through.

We would like to thank our supervisor teacher, Pr. Hichem Debbi, your mentorship, feedback, and dedication to my growth have been instrumental. Thank you for steering me in the right direction.

We would also like to thank all the Jury Members, who have agreed to review this work.

To everyone who contributed, whether directly or indirectly, your assistance mattered. From late-night brainstorming sessions to technical troubleshooting, your collective efforts made this journey possible.

Thanks

Dedication

*I dedicate this work to my dear **grandmother** who resides forever in my heart. Your wisdom, warmth, and unwavering love have shaped my journey. Though you're no longer with us, your spirit continues to guide me.*

*And to my beloved **mother, Amer Ouali Halima** and to my **father, Madjid***

*To my dear brothers **Ahmed** and my little sister **Maria***

*And to my cousin **Ikram***

And to all those are precious parts of our lives

Ben Dahmane Dhiya El Haq

*I dedicate this humble work to my parents, my beloved mother **khmisa** and my dear father, **Mahfoud** and to my seven brothers, **Abd al Latif**, **Nour El Din**, **Hamida**, **Hanan**, **Adel**, **Marwa**, **Amira** and to the children, **Adam Arwa Israa***

Kouriche Oussama

Table of contents:

List Of Figures

ABBREVIATION LIST

CMS: Cabin Management System

SaaS: Software as a Service

EHR: Electronic Health Records

HMS: Healthcare Management Systems

EMRs: Electronic Medical Records

PHRs: Personal Health Records

PBMR: Paper-Based Medical Records

DCMS: Doctor Cabin Management Systems

CM : Cabin Management

RCM : revenue cycle management

SSO: Single Sign-On

CASB: Cloud Access Security Brokers

CSB: Cloud Services Broker

CRM: Customer Relationship Management

ERP: Enterprise Resource Planning

AaaS: Applications as a service

UML: Unified Modeling Language

IT: information technology

IaaS: infrastructure as a service

PaaS: platform as a service

B2B: Business to Business

B2C: Business to Consumer

AWS: Amazon Web Services

SSL: Secure Sockets Layer

XSS: Cross Site Scripting

CSRF: Cross Site Request Forgery

XSRF: Cross-site Request Forgery

HTML: Hypertext Markup Language

CSS: Cascading Style Sheets

AJAX: Asynchronous JavaScript and XML

JSON: JavaScript Object Notation

ORM: Object-relational mapper

PHP: Hypertext Preprocessor

GUIs: Graphical User Interface

General Introduction

The healthcare industry has undergone a significant digital transformation in recent years, driven by the ever-increasing demand for efficient and accessible healthcare services. With the proliferation of cloud computing and the advent of Software-as-a-Service (SaaS) models, healthcare providers have been presented with an opportunity to streamline their operations, enhance collaboration, and improve patient care.

One of the critical aspects of healthcare management is the effective organization and coordination of healthcare cabin operations, encompassing various processes such as appointment scheduling, patient record management, billing, and reporting. Traditionally, these tasks have been carried out through manual or fragmented systems, leading to inefficiencies, data lose, and potential errors.

The primary objective of this work is to design and implement a robust and secure SaaS platform, utilizing multi-tenancy technology, to create a comprehensive cabin management system. This platform, built on a multi-tenant architecture, aims to enable the creation of multiple virtual healthcare cabins, each tailored to the specific needs of individual healthcare providers or clinics. By embracing the principles of multi-tenancy, the platform can isolate data and configurations for each tenant while sharing the same codebase and infrastructure, resulting in efficient resource utilization and simplified maintenance.

To address the challenges faced by healthcare providers in managing cabin operations, we have developed an innovative Software-as-a-Service (SaaS) platform that leverages the latest cloud computing technologies and multi-tenancy architecture. This platform offers a scalable, cost-effective renting service for healthcare cabin management systems, enabling providers to access tailored solutions without the complexities of maintaining traditional systems.

CHAPTER 1

HEALTHCARE MANAGEMENT SYSTEMS

Introduction:

Paper-based record keeping significantly diminished the efficiency of healthcare providers. The manual process of updating and maintaining physical files consumed valuable time and resources. Accessing comprehensive patient profiles has been always a complex and time consuming task, leading to potential gaps in information. The risk of losing or misplacing crucial medical data is generally large. As healthcare systems grew more complex, the limitations of paper records became clearly limited. This outdated approach hindered prompt, coordinated, and well-informed treatment delivery. The need for a digital transformation became evident to overcome these inefficiencies.

As healthcare delivery systems evolved, the adoption of electronic forms of record-keeping emerged as a transformative solution. Electronic Health Records (EHR) and Healthcare Management Systems (HMS) have served as powerful tools to facilitate the task for healthcare professionals, fostering improved efficiency and effectiveness in care delivery processes. By digitizing patient data and streamlining information flow, these systems have led to a new era of accountability and coordination within the healthcare sector.

Developing robust and effective HMS has become a paramount concern for governments and healthcare decision-makers alike. The spotlight on the management aspects of these systems underscores their pivotal role in facilitating patient treatment and care.

In this chapter, we will provide a background on the concept of Healthcare Management Systems and their impact on individuals. We will explore HMS strategies, their importance, and highlight successful implementation examples from our country as well as others. Additionally, we will

examine what constitutes HMS software, the key functions it performs, and how it empowers the healthcare industry to be more effective and patient centric.

1. Healthcare Management Systems:

1.1 Definition:

Basically, a healthcare management system is a cloud-based web medical management framework that registers and incorporates the data from every single department to make all the internal activities automated.

A healthcare management system software seeks to cover a wide range of tasks: from streamlining the workflow for medics to providing superior patient experience and everything in between, such as downsizing administrative costs and reducing errors.

HMS consolidates clinicians, patients, medical supplies, laboratory results data, insurance, medical bills, doctor appointments, and reports.

Imagine a healthcare system without an accurate medical facility management apparatus: it doesn't efficiently retain decent staff, provides low-grade patient care, and isn't even profitable enough. To be more specific, it is necessary to determine the types of healthcare management systems we will illustrate them as following.

1.2 Healthcare Management Systems Type:

1.2.1 Paper-based Medical Records (PBMR):

Before the advent of electronic systems, healthcare facilities relied on paper-based medical records to document and store patient information. These records typically included handwritten notes, test results, prescriptions, and other relevant medical data.

There are reasons why paper medical records were persistent for several decades. We cite here some advantages:

- Familiarity: Healthcare professionals have been using paper records for decades, and the transition to a new system can be challenging.

- Ease of use: Paper records can be accessed and updated quickly without the need for specialized training or technology.
- No dependency on technology: Paper records do not rely on computers, software, or internet connectivity.

However, with paper medical records, we still have many open problems. We cite here some challenges and drawbacks:

- Limited accessibility: Paper records are typically stored in a single location, making it difficult for multiple healthcare providers to access them simultaneously.
- Potential for loss or damage: Physical records can be misplaced, damaged, or destroyed due to natural disasters or human error.
- Inefficient data management: Retrieving and analyzing data from paper records is time-consuming and prone to errors.
- Lack of interoperability: Paper records cannot be easily shared or integrated with other healthcare systems or facilities.
- Storage requirements: Physical storage space is needed to store large volumes of paper records, which can be costly and space intensive.

1.2.2 Electronic Health Record (EHRs):

EHRs are digital versions of patient medical records that store and manage patient information electronically. They are designed to replace paper-based records and provide a more comprehensive, integrated, and accessible approach to healthcare documentation. EHRs typically include the following components:

- Electronic Medical Records (EMRs): EMRs are the digital equivalent of paper-based medical records, containing a patient's medical history, diagnoses, treatments, and test results.
- Personal Health Records (PHRs): PHRs allow patients to access and manage their own health information, promoting patient engagement and self-care.
- Practice Management: EHRs often include functionality for scheduling appointments, billing, and other administrative tasks.

- Clinical Decision Support: EHRs can provide decision support tools, such as alerts for potential drug interactions or evidence-based treatment guidelines, to assist healthcare providers in making informed decisions.

There are reasons why EHR takes over PBMR We cite here some advantages:

- Improved accessibility: EHRs can be accessed by authorized healthcare providers from multiple locations, facilitating better coordination of care.
- Enhanced data management: EHRs allow for efficient storage, retrieval, and analysis of patient data, enabling better decision-making and quality improvement efforts.
- Increased interoperability: EHRs can be integrated with other healthcare systems, enabling seamless data sharing and communication.
- Reduced risk of errors: EHRs can help minimize transcription errors, illegible handwriting, and other potential sources of mistakes.
- Patient engagement: EHRs can provide patients with access to their medical records, promoting better understanding and engagement in their care.

However, with Electronic Health Record we still have many challenges. We cite here some challenges and drawbacks:

- High implementation costs: Acquiring and implementing an EHR system can be expensive, particularly for smaller healthcare organizations.
- Training requirements: Healthcare professionals and staff need to be trained on how to use the EHR system effectively, which can be time-consuming and costly.
- Data privacy and security concerns: EHRs store sensitive patient information, which requires robust cybersecurity measures and compliance with data protection regulations.
- Interoperability issues: Despite efforts to standardize data formats, interoperability between different EHR systems and healthcare facilities can still be a challenge.
- Resistance to change: Some healthcare professionals may be reluctant to adopt new technologies and workflows, which can hinder the successful implementation of EHRs.
- Potential for system downtime: EHRs rely on technology, and any system failures or outages can disrupt healthcare operations and access to patient data.

1.3 Doctor Cabin Management System (DCMS):

Doctor cabin management is a software product suite designed to improve the quality and management of Cabin Management (CM) in the areas of clinical process analysis and activity-based costing, CM enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the cabin helps you manage your processes.

1.3.1 Key Components of Doctor Cabin Management:

- Patient Registration and Appointment Scheduling Module: The Registration module is an integrated patient management system, which captures complete and relevant patient information. The system automates the patient administration functions to have better and efficient patient care process
- Streaming patient Data: Enabling a fast data stream between different doctors within a healthcare facility ensures seamless collaboration and care coordination. Rapid access to up-to-date patient information empowers physicians to make informed decisions, ultimately enhancing the quality of care provided.
- Optimized Doctor Workflow: Efficient scheduling allows doctors to focus on patient care and get Quick access to patient records aids decision-making.

1.4 Healthcare Management Systems for the patient:

In cabin management systems, prioritizing the patient experience is paramount. Ensuring efficient queue management, minimizing wait times, and maintaining a comfortable environment within the consultation rooms contribute to enhanced patient satisfaction. Integrating patient education tools and secure data access further elevates the overall quality of care delivery.

- Enhanced Patient Experience: An effective doctor cabin management system optimizes efficiency by streamlining workflows, reducing patient waiting times through seamless appointment scheduling that allows convenient online bookings, and providing real-time

updates to ensure patients stay informed about their appointments and any potential changes.

- Improved Records Management: Patient data is securely stored in a structured format and can quickly access and update patient information in real time.
- Transparency and Communication: empower patients by allowing them to track their appointments, prescriptions, and treatment plans, facilitating efficient communication between patients and doctors, and ensuring patients feel well-informed and actively involved in their own care journey.

2. Healthcare Management Systems in the word

In this section, we will set here the most well-known Healthcare Management Systems in the world and in Algeria. We will reveal some of the advantages and limitations.

2.1 Athenahealth:

Athenahealth is a start-up founded in 1997 [1].Withe the evolution of cloud base it's the leading provider of cloud-based healthcare software and services for medical practices and health systems. In the first three quarters of 2022 saw athenahealth signing up 2,200 new users [2]. The key value proposition of athenahealth is providing an integrated, cloud-based solution that combines clinical, revenue cycle, patient engagement, and care coordination capabilities on one platform

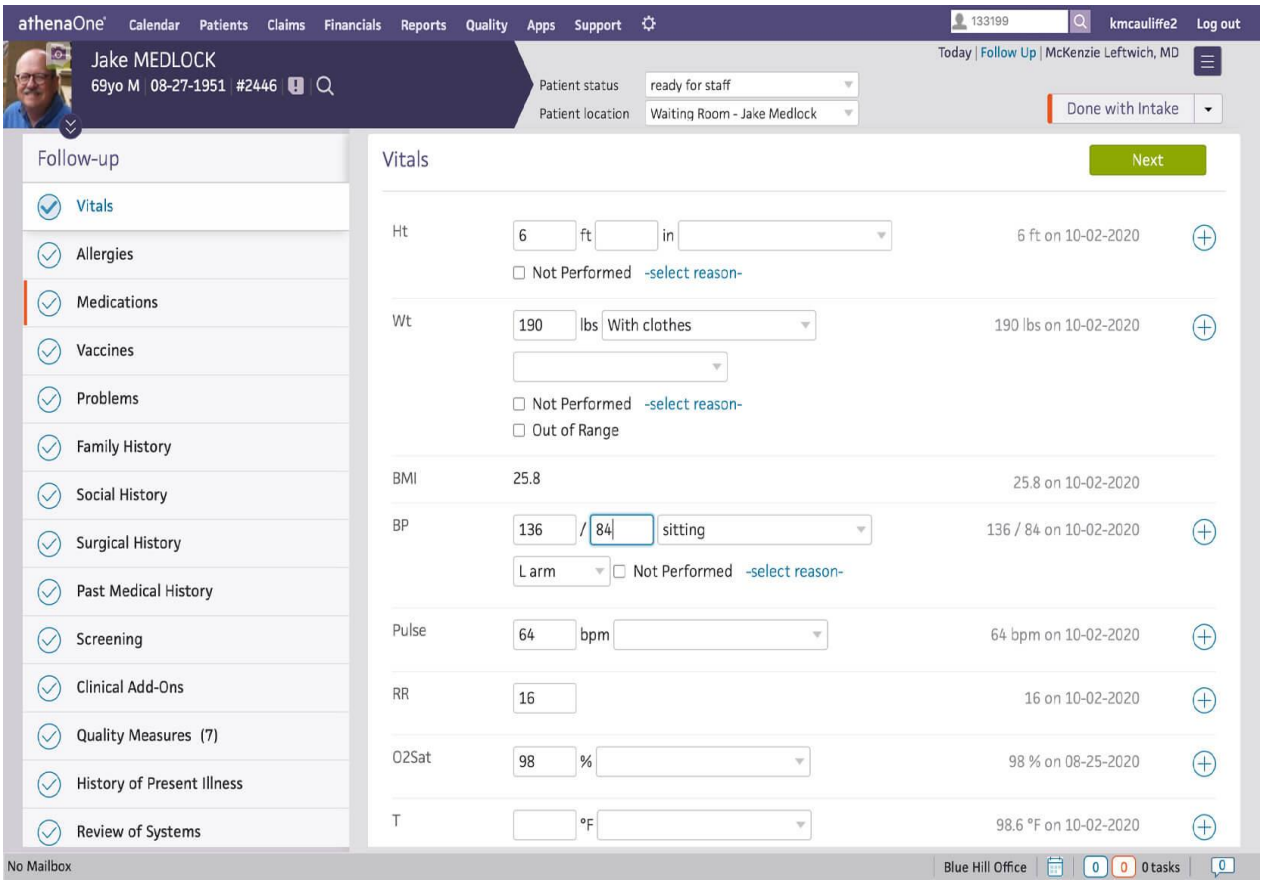


Figure 1: athenahealth Dashboard

Advantages:

- Offers a comprehensive suite of services in one platform
- Real-time access to patient data
- Wide range of third-party integrations
- Cloud-based, accessible from anywhere
- Regular software updates and maintenance included
- Comprehensive suite of integrated solutions

Disadvantages :

- Can be expensive for smaller practices
- Limited customization options
- Third-Party Software Integrations with Fees, while athenahealth integrates with third-party applications, some of these integrations may come with additional costs.
- All staff using the software must be trained

- Users need to understand the application in detail to get the maximum benefit.

2.2 ZOCDOC:

Zocdoc, the healthcare marketplace that makes it easy for patients to find and book in-person or virtual care across over 250 specialties and more than 18,000 insurance plans, today introduced Guided Search. This new search experience intuitively provides patients with a more tailored set of results based on their unique care needs so they can select the right provider and book with greater confidence [3].

Advantages:

- ZOCDOC offers a wide range of tools, including patient management, electronic health records (EHR),
- Psychiatry providers available to prescribe mental health medication
- Zocdoc doctor reviews available to help you choose your psychiatrist or therapist [4]
- Cloud-based, accessible from anywhere
- Regular software updates and maintenance included

Disadvantages :

- No guarantee of an in-network provider match when using insurance
- Works only in some countries

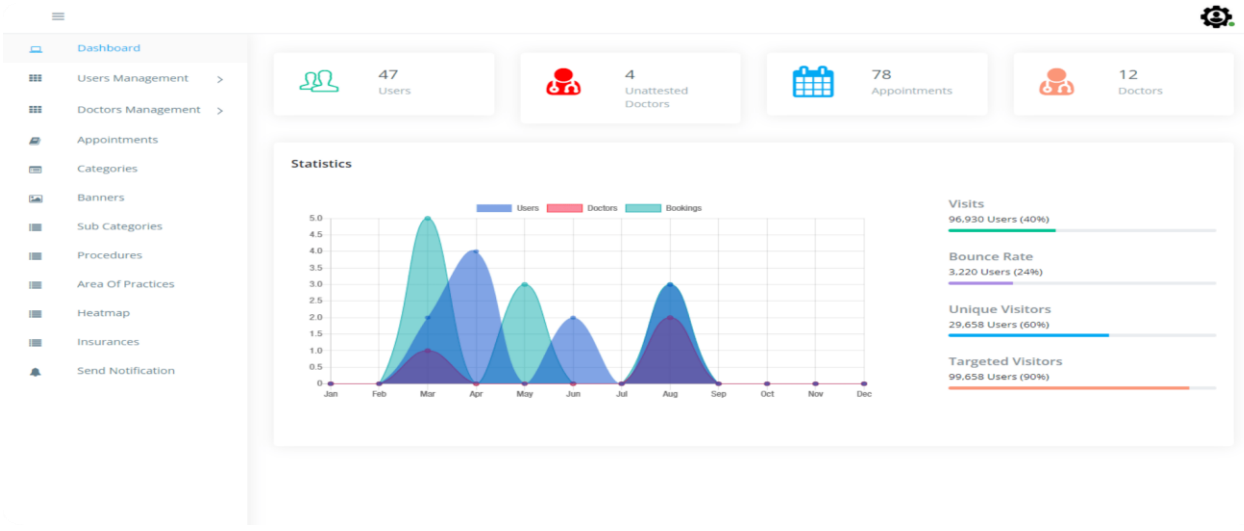


Figure 2: ZOCDOC Dashboard

3. Healthcare Management Systems in Algeria

3.1 CabiSante:

"CabiSanté is designed to help you manage all the activity of your medical practice, from the waiting room to payments and accounting with complete simplicity and efficiency" [5]

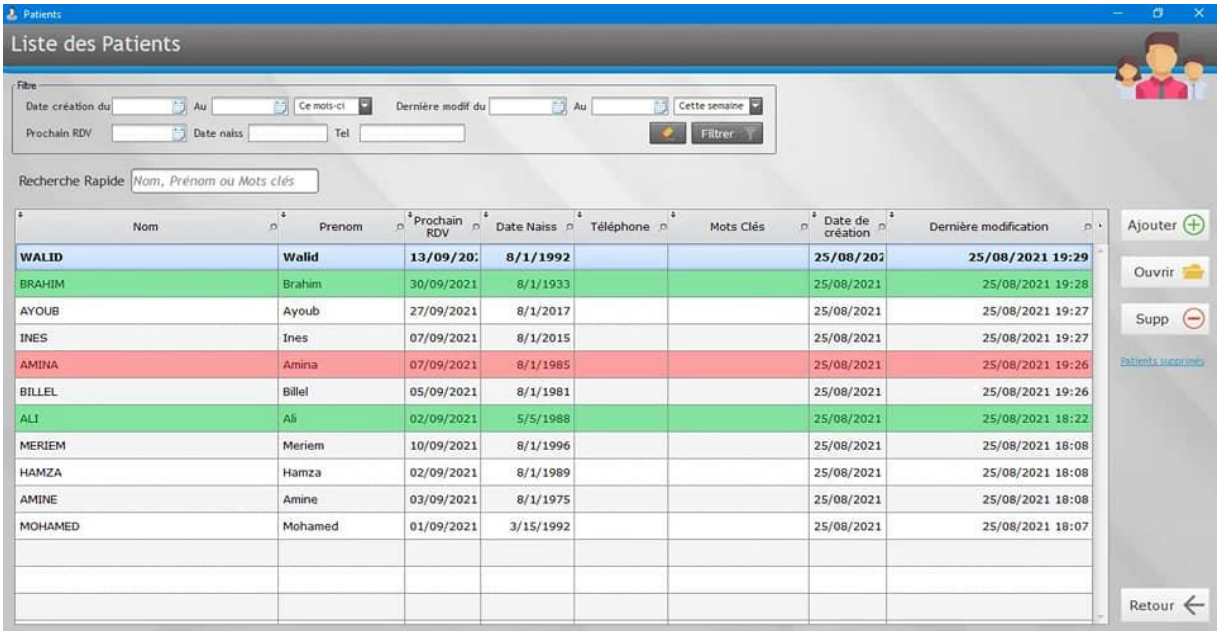


Figure 3:CabiSante Dashboard

Advantages:

- CabiSanté boasts a user-friendly interface with an intuitive design. It allows users to quickly familiarize themselves with its most interesting and attractive features.
- You do not need prior experience in using hospital systems a little training is enough for the new joiners to understand the software
- It is suitable both for small and big hospitals.
- Customer support is extremely good.

Disadvantages :

- It can run only on one computer because it is a desktop application and it works offline
- For every new client, it is necessary to install the application on the client compute
- There is no backup in case of losing data.

3.2 ETabib:

An Algerian e-health platform that bridges the gap between qualified doctors and citizens across the country, its lain focus is for patients since they offer Telemedicine and Online Consultations. [6]

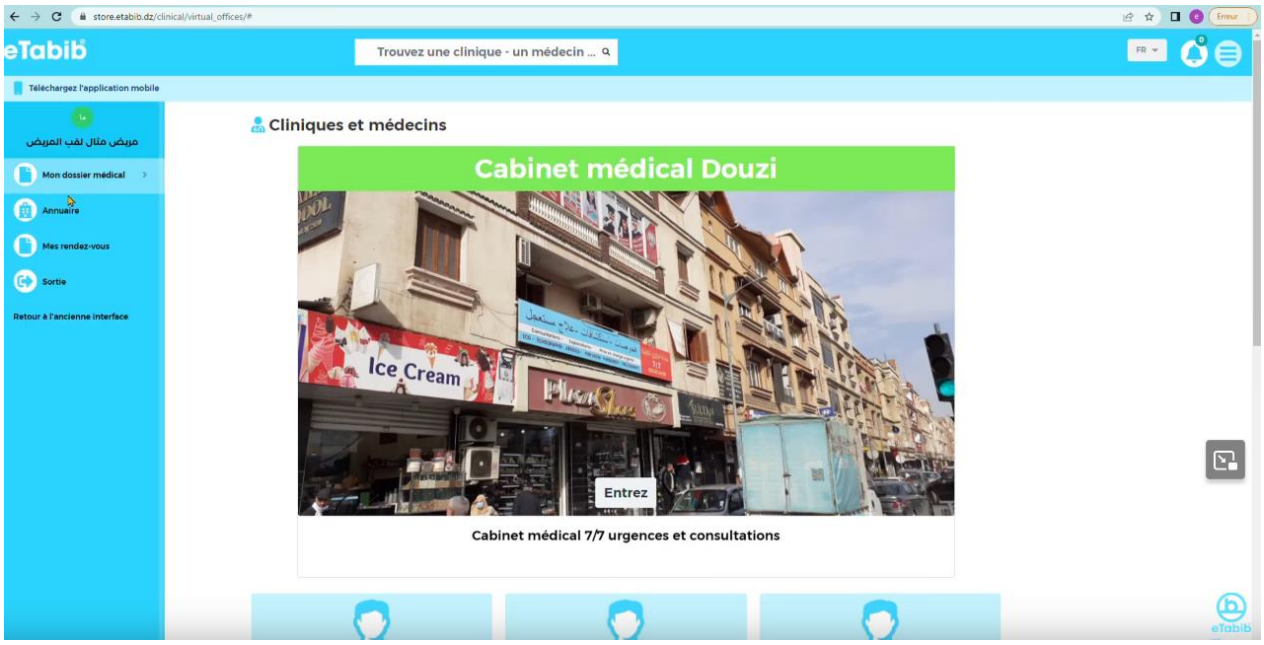


Figure 4: eTabib patient Dashboard

Advantages:

- search for doctors, book appointments online.
- access medical advice.
- telemedicine consultations with doctors.
- provides a secure platform for users to store their medical records**.**
- They have an android app.

Disadvantages :

- Slow internet connection it can be a problem and affecting the solution speed.
- A very complicated interface.
- A very limited features for doctor's side.

- Doctors not always available for the telemedicine feature.

Conclusion:

In this chapter, we have introduced Healthcare Management Systems with its types, Doctor Cabin Management System and its impact on patients' records management. And we talked about the role of DCMS in patient side. Also, we have presented some of the most known HMS in the world, and the one known in Algeria. Taking into account all the mentioned details, so we say that the health care management System is very important in the medical section, and every doctor who has not kept pace with the digitization of the health sector should resort to this solution that makes it easier for him and his patients

CHAPTER 2

ANALYSIS AND DESIGN

Introduction:

This chapter presents the practical part of this project, the analysis and design phase that will be done through diagrams. To lead to a design and analysis we have chosen the UML, which intended to facilitate the design of the documents necessary for the development of object-oriented software, as a standard for modeling software architecture.

1. Application's Analysis and Design:

Based on our study and analysis of the case study design, we provide the following Unified Modeling Language (UML) models to describe formally our proposed architecture.

1.1 Analysis by Use Case diagram:

The global use case diagram represents the different functions of our application and the needs and requirements of the various actors who will interact within the system.

1.1.1 General Use Case Diagram:

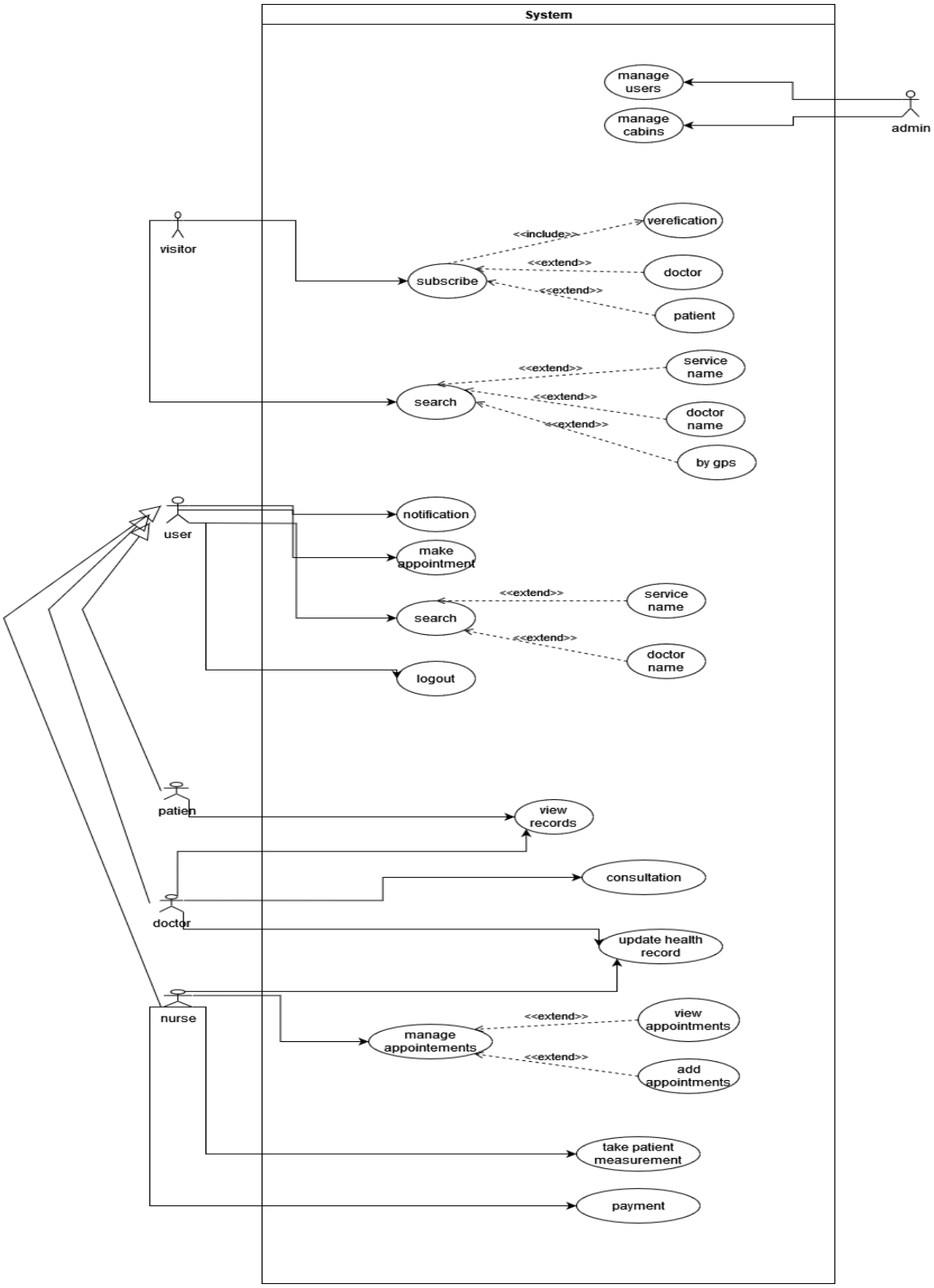


Figure 5: General use case diagramme

Next, we will detail in the above use case.

1.1.2 Visitor use case diagram:

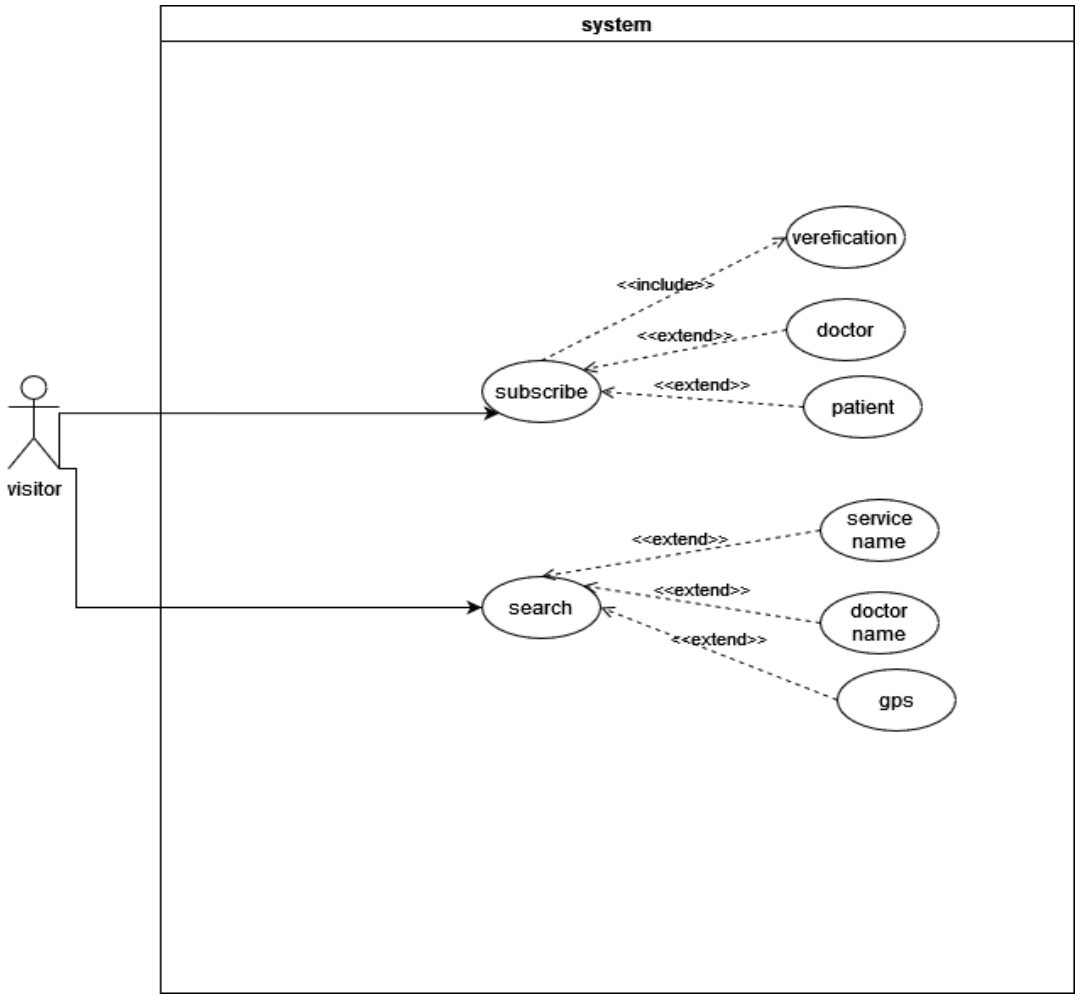


Figure 6:Visitor use case Diagram

- **The actors:**
  - **The Visitor:** an unregistered user that can view what we offer or either register as a patient or a cabin
- **Use cases**
  - **Subscribe as a cabin(doctor):** request of subscription for a new cabin by providing personal information and Property documents
  - **Subscribe as a patient:** register by filling all the information requested
  - **Search:** Search for a cabin by name, service or GPS

1.1.3 User use case diagram:

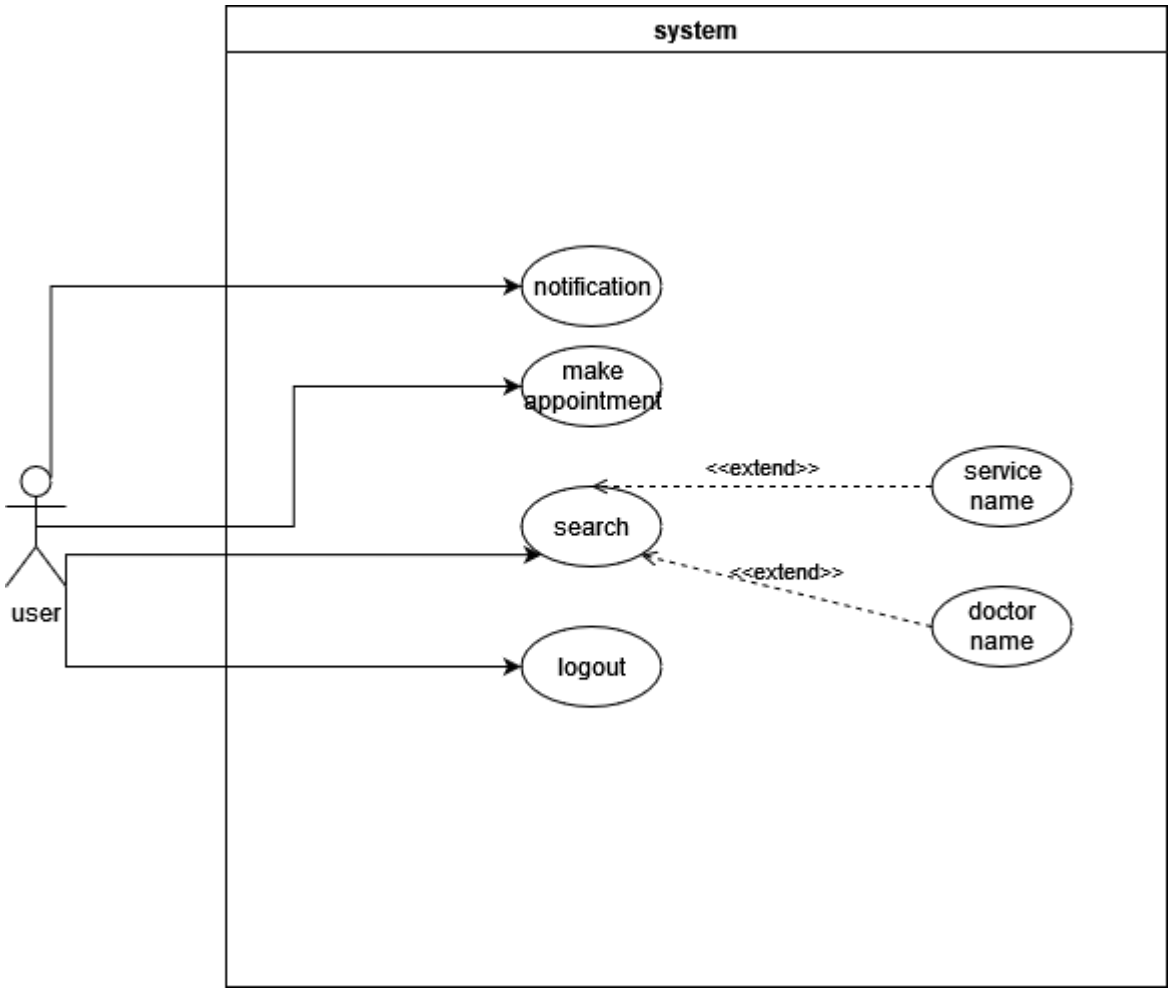


Figure 7: User use case Diagram

- **The actors:**
    - **The user:** any connected account called user on general, also any user can receive a notification, search for a cabin about Doctors or Services also can logout
- **Use cases**
    - **notification:** notifications are sent by the application of specific information
    - **Search:** Search cabin profile about the doctors or services
    - **Make appointment**: set an appointment by providing personal information and email address

- **Logout:** Logout from the website

1.1.4 Patient use case diagram:

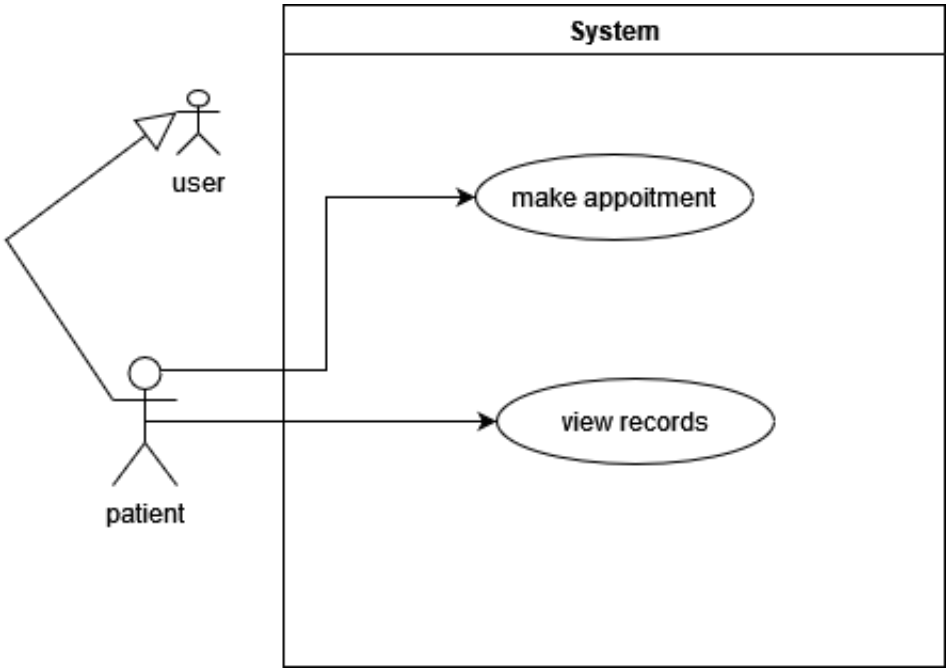


Figure 8: Patient use case diagram

- **The actors:**
  - **The user:** a registered user that has an account
  - **The Patient:** a user with type patient can consult its profile, view its history, and make appointments with its registered information
- **Use cases**
  - **Make appointment:** set an appointment by select the date and the doctor
  - **View records:** review personal history.

1.1.5 Doctor use case diagram:

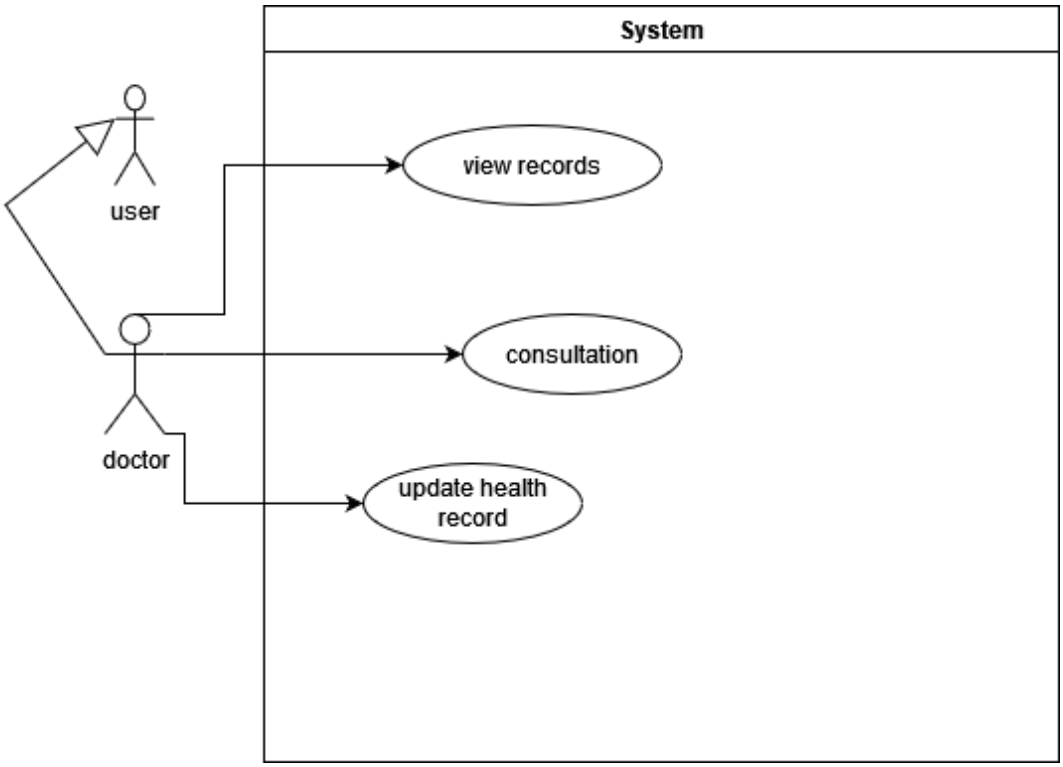


Figure 9: Doctor use case diagram

- **The actors:**
    - **The user:** a registered user that has an account
    - **The Doctor:** his responsibility is about consulting the patients, set electronic reports
- **Use cases**
    - **consultation:** making consultation to the patient
    - **View records:** review personal history.
    - **Update health record:** means he can edit previous consultations

1.1.6 Nurse use case diagram:

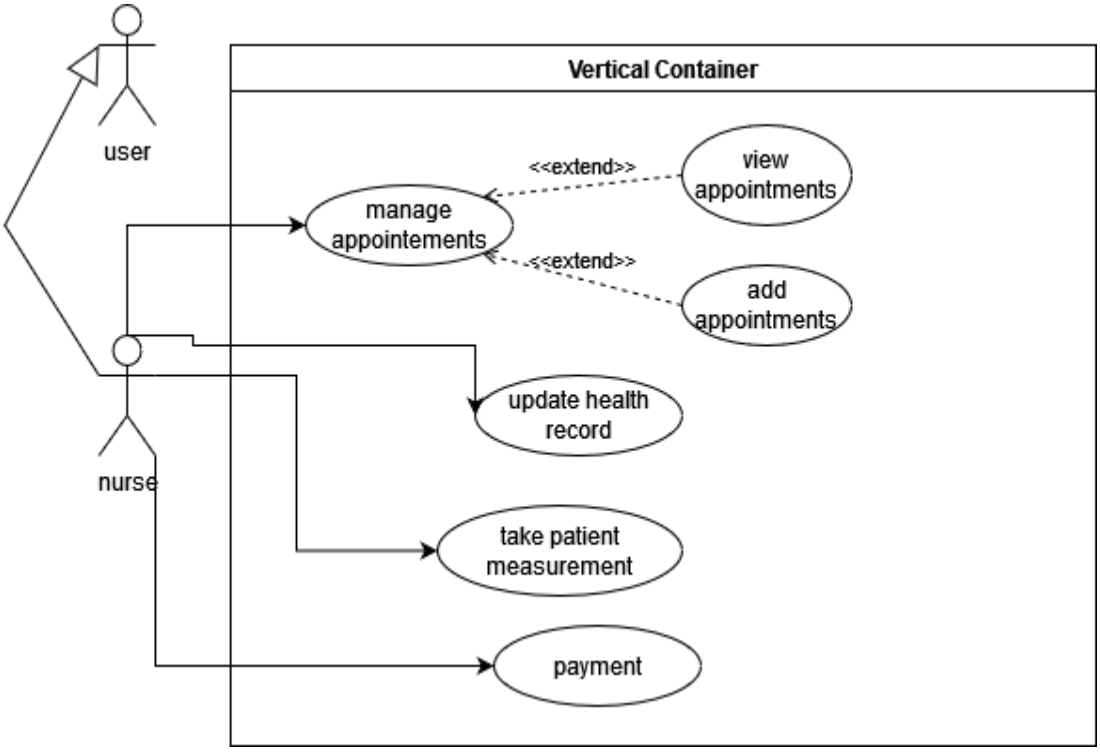


Figure 10: Nurse use case diagram

- **The actors:**
    - **The user:** a registered user that has an account
    - **The nurse:** have the responsibility editing the consultation date if it was containing a fault and take the cost after the patient finish his consultation
- **Use cases**
    - **Update health record:** if there is an error in the consultation the nurse can make small changes
    - **Take patient measurement:** check blood pressure, height, wight and save it
    - **Payment:** after the doctor done with consultation the nurse takes the cost from the patient
    - **Manage appointment:** he/she can check the appointments available or add new appointments and send the patients to doctor

1.1.7 Admin use case Diagram :

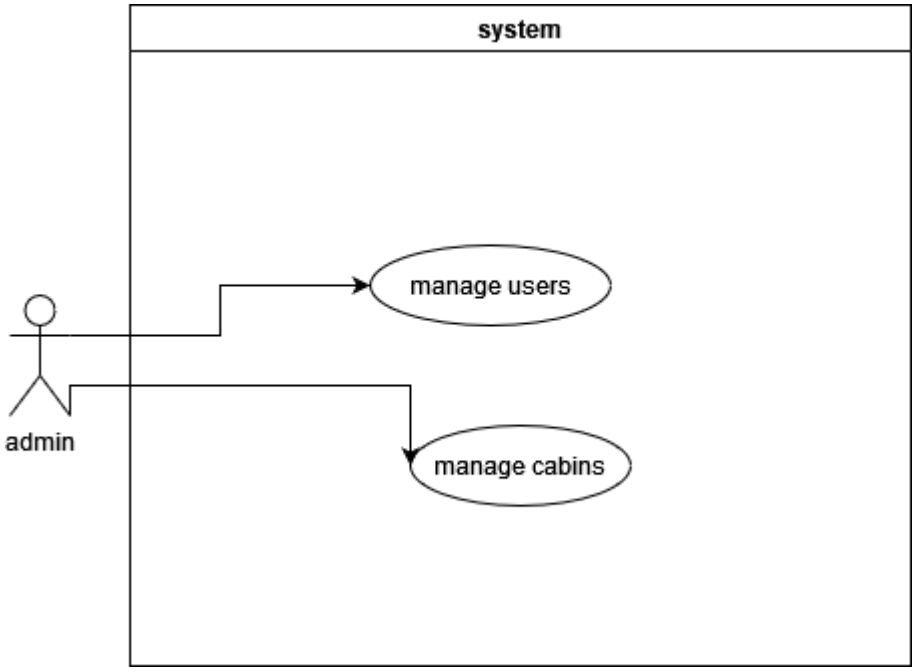


Figure 11: Admin use case diagram

- **The actors:**
    - **The admin:** the one who can handle all what's going on the web site he creates the cabins and manage all the users
- **Use cases**
    - **Manage users:** he creates the cabins and have access to all the user's information
    - **Manage cabins:** means he create a cabin service after confirming the subscription and can add features deepens on the demand of the doctors

1.2 Application Design by static diagram:

The class diagram represents the main structure of our application system at the level of classes, shows their relationships, associations, generalizations, and dependencies.

1.2.1 General class diagram:

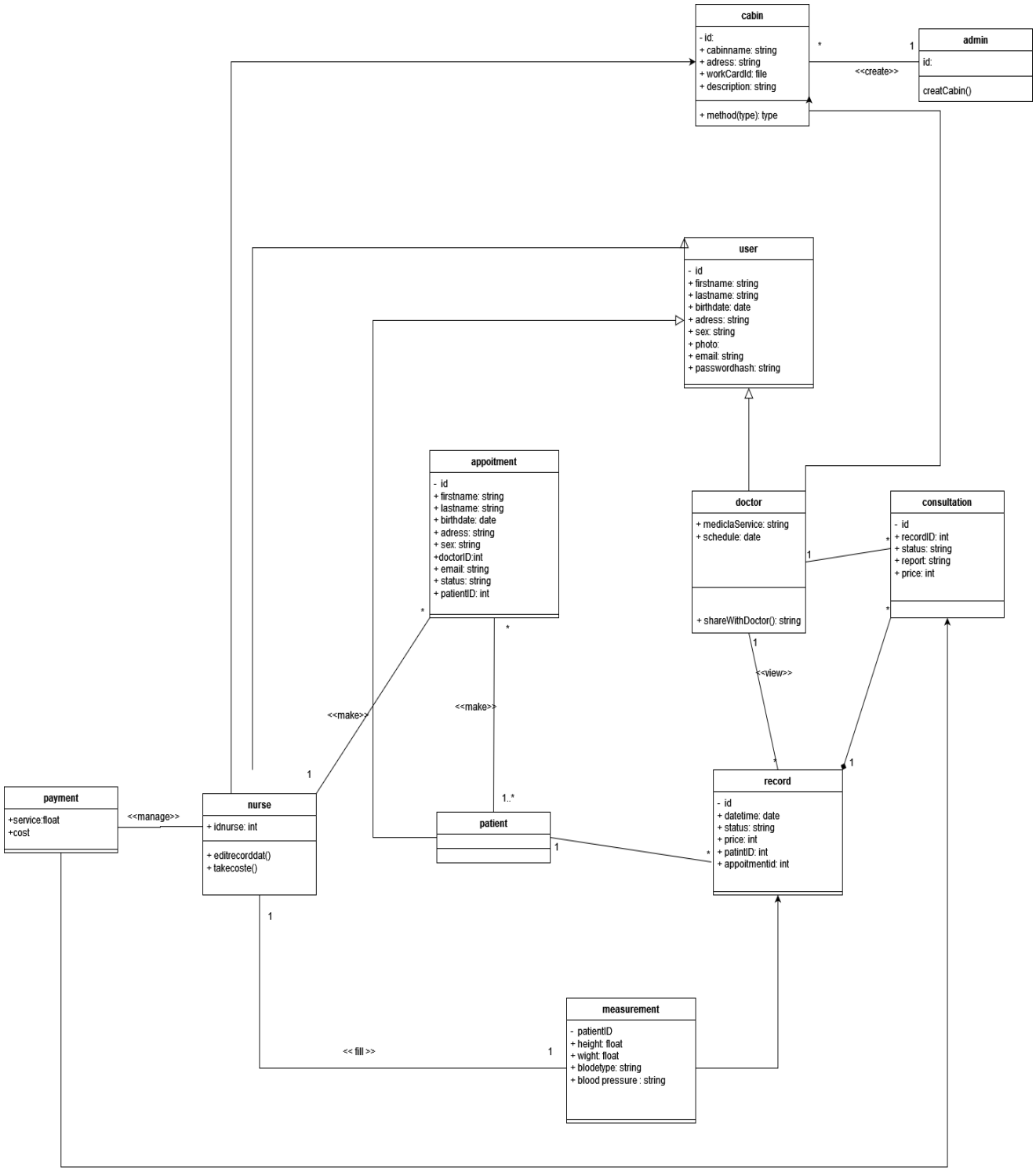


Figure 12: General class Diagram

1.2.2 Patient class diagram:

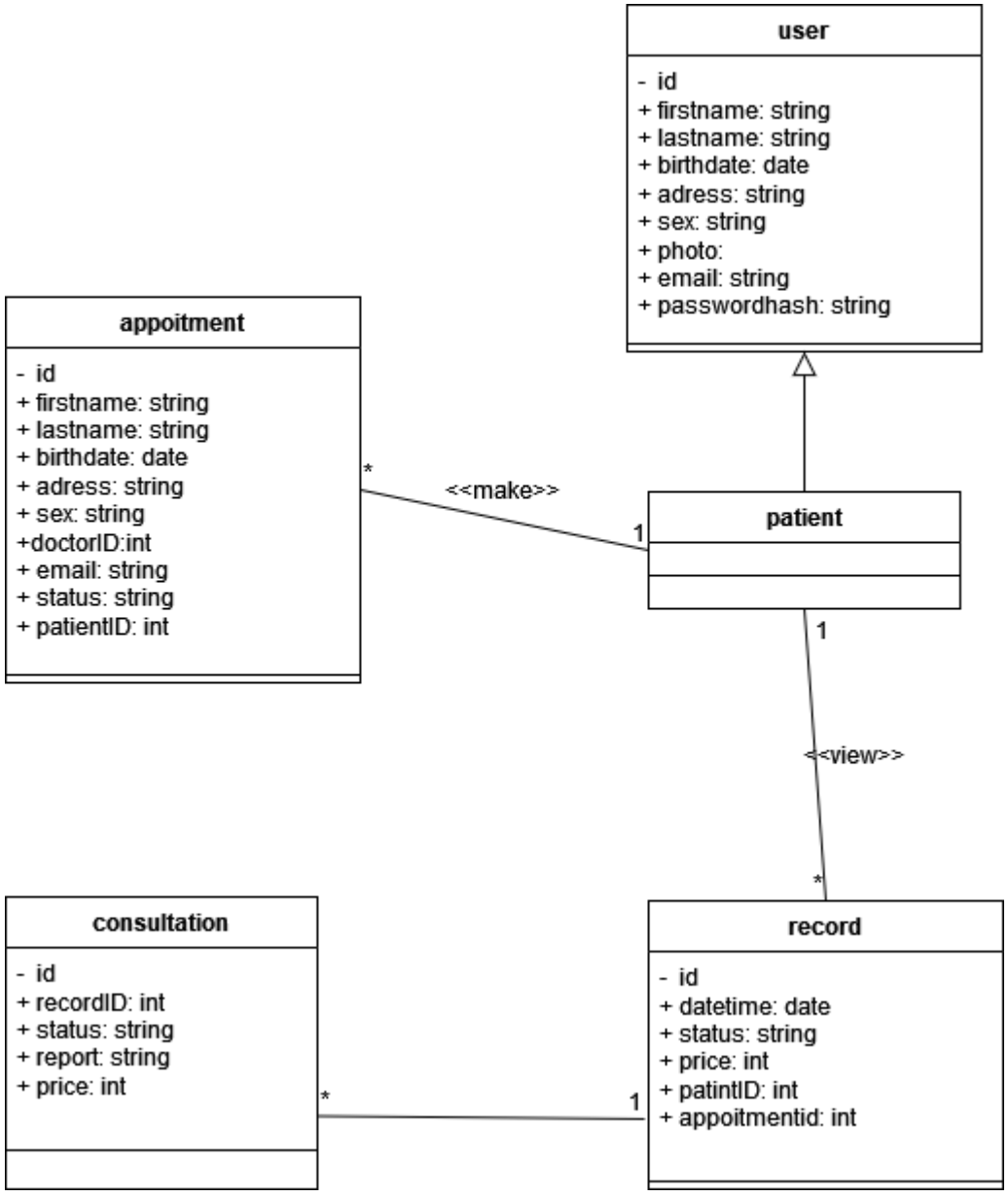


Figure 13: Patient class Diagram

1.2.3 Doctor class diagram:

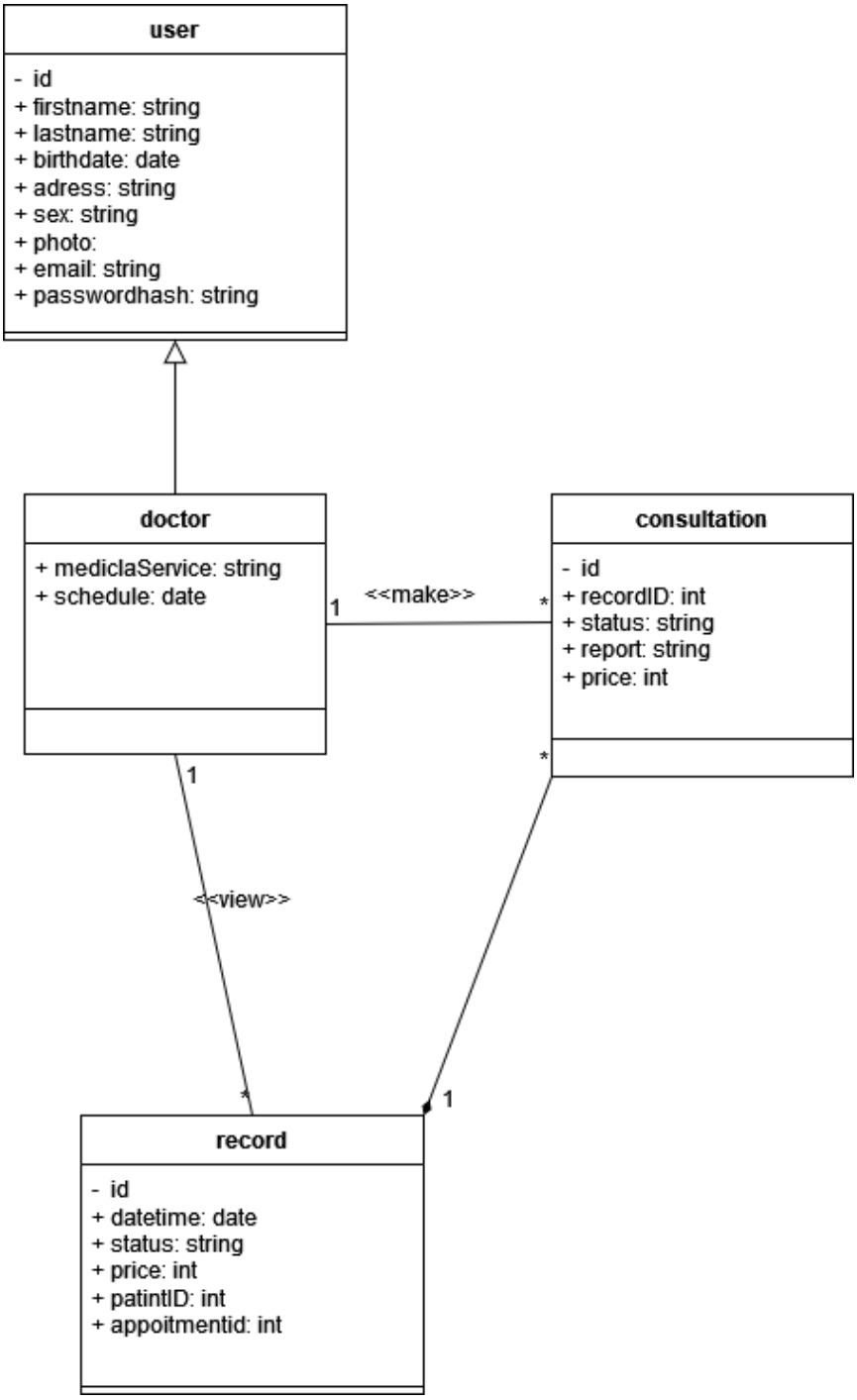


Figure 14: Doctor class Diagram

1.2.4 Nurse class diagram:

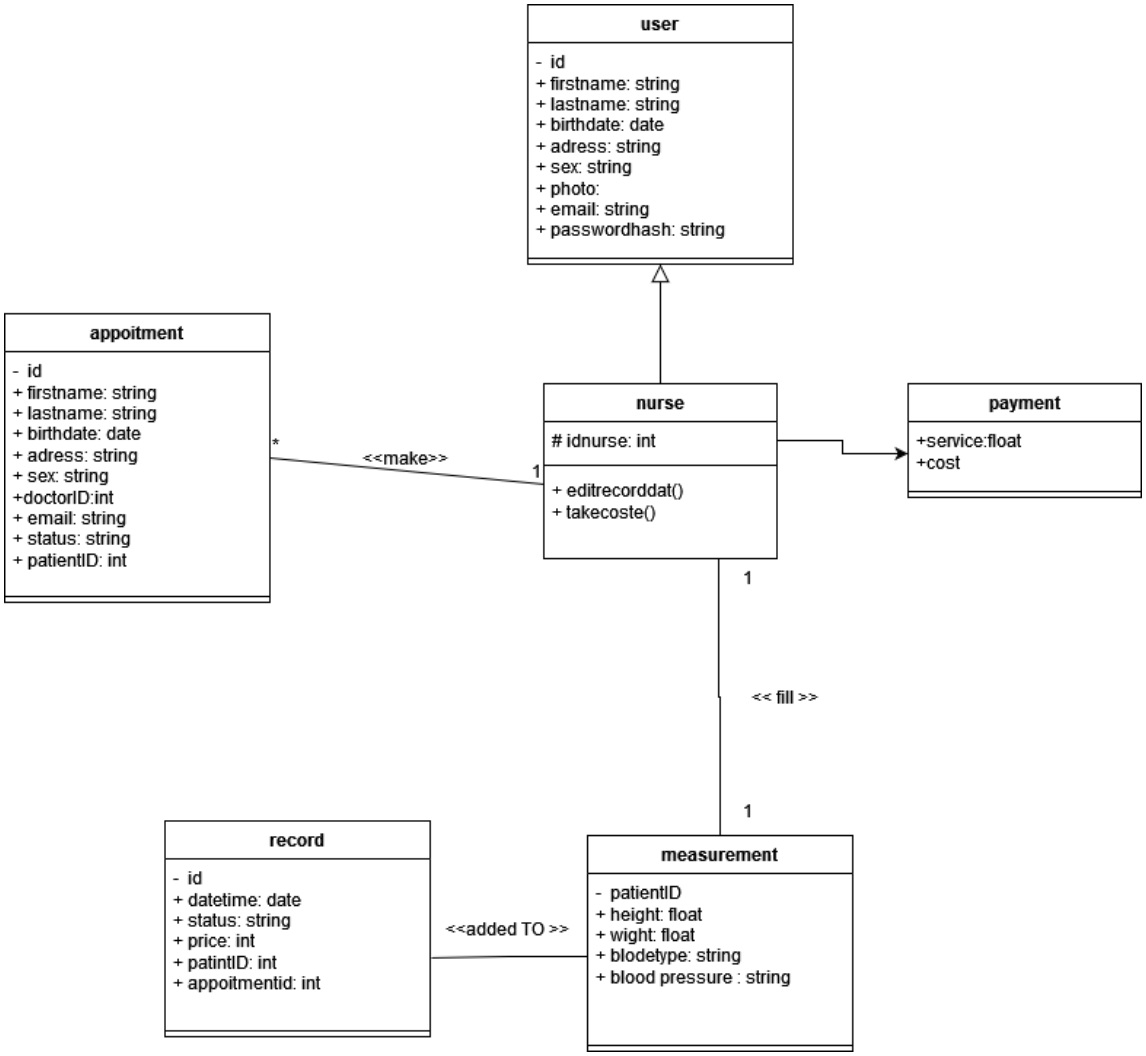


Figure 15: Nurse class Diagram

1.2.5 Admin class diagram:

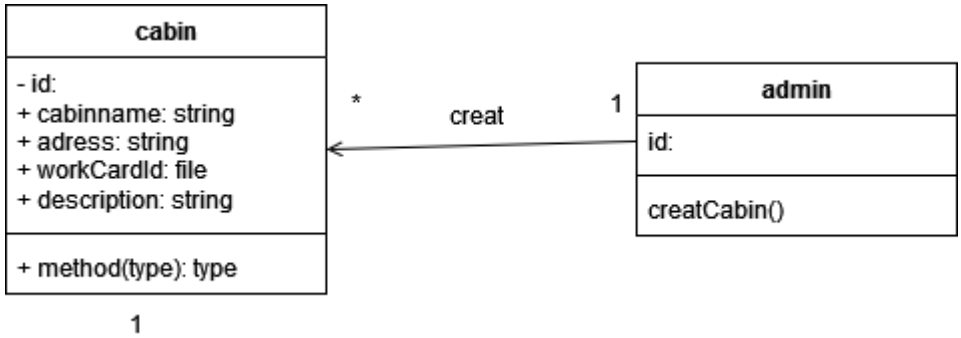


Figure 16: Admin class Diagram

1.3 Application's Design by Sequence diagrams:

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration.

1.3.1 sequence diagram for patient registration:

Name: Registering

Objective: User register as a patient

Actors: User

Pre-condition: User must comply with the registration conditions

Nominal scenario:

1- user brows to the patient registration page

2- user insert all the information needed and register

3- the system checks if the user exists or not, create new user

Post-condition:

- Successful: the creation of a User (account), updating the database.
- Failure: return to the registration page.

Extensions:

- Required fields not filled: the system asks to fill the empty fields.
- Invalid input, an error message displays.

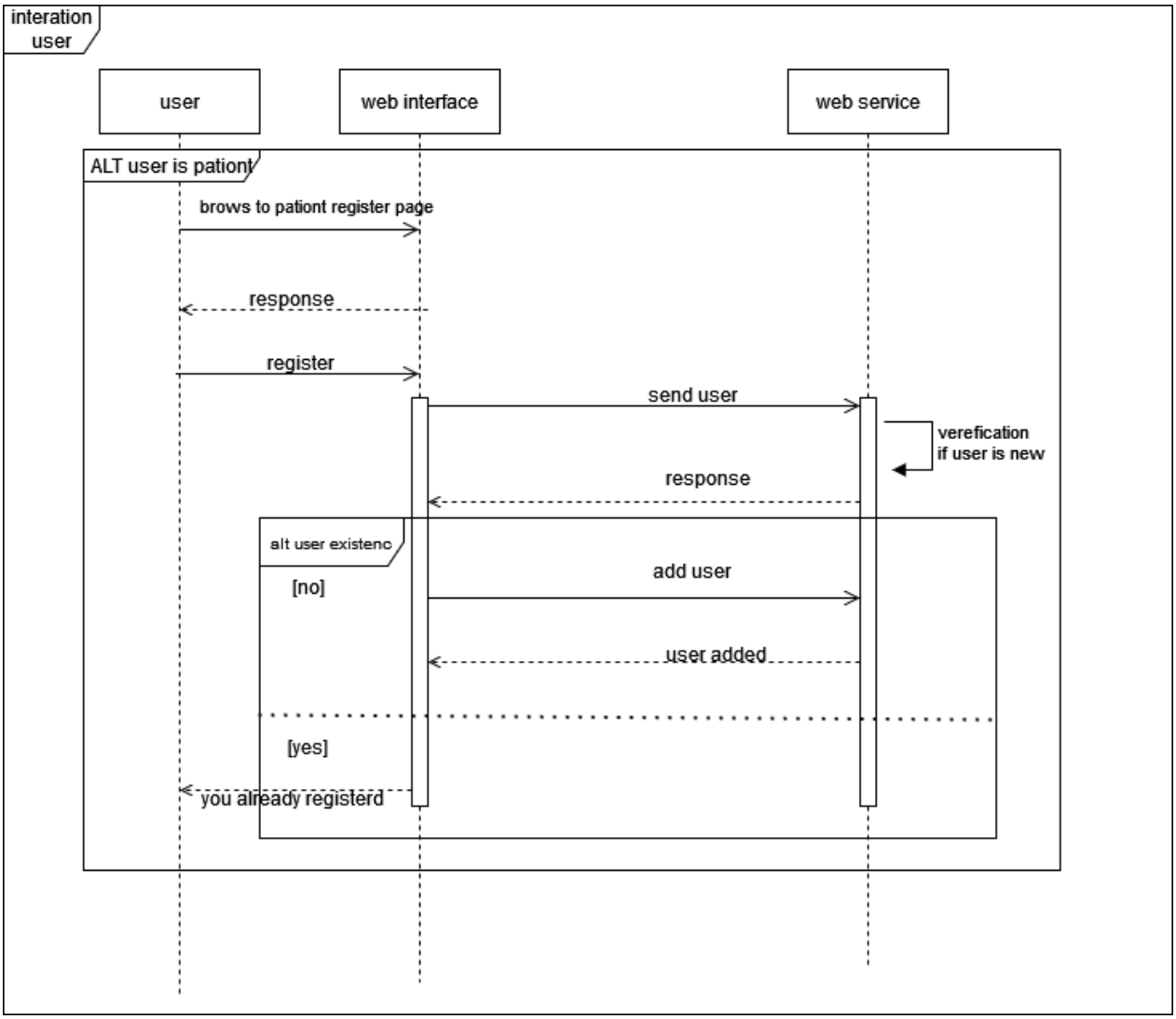


Figure 17: Sequence Diagram for Patient Registration

1.3.2 sequence diagram for cabin registration:

Name: Registering

Objective: User register as a cabin

Actors: User, Admin

Pre-condition: User must comply with the cabin registration conditions

Nominal scenario:

- user brows to the cabin registration page

- user insert all the information needed and register
- the admin check if all the necessary files are valid, create cabin

Post-condition:

- Successful: the creation of a cabin and send an email with all the informations , updating the database.
- Failure: return to the registration page.

Extensions:

- Required fields not filled: the system asks to fill the empty fields.
- Invalid input, an error message displays.

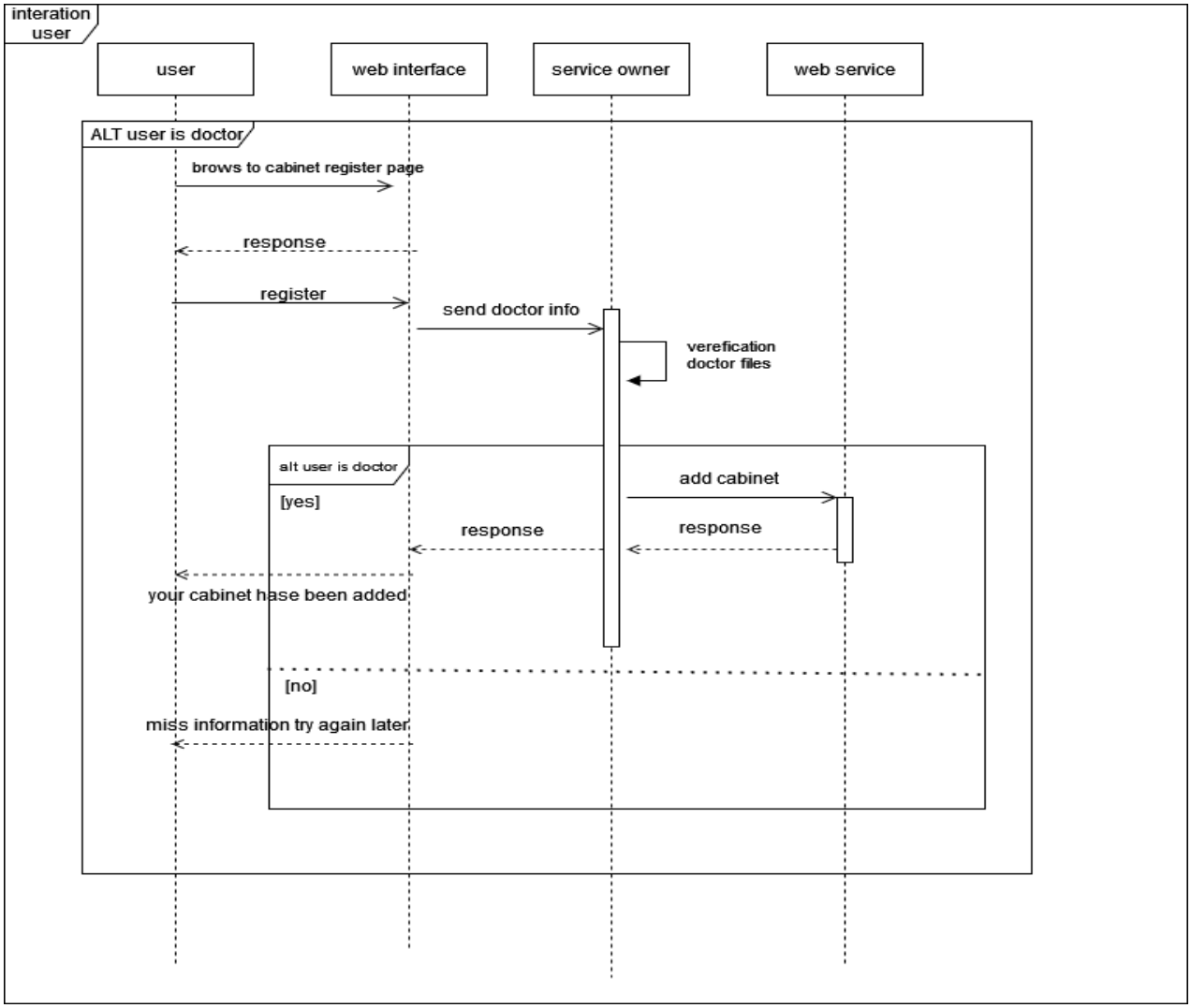


Figure 18: Sequence Diagram for Cabin Registration

1.3.3 sequence diagram for appointment:

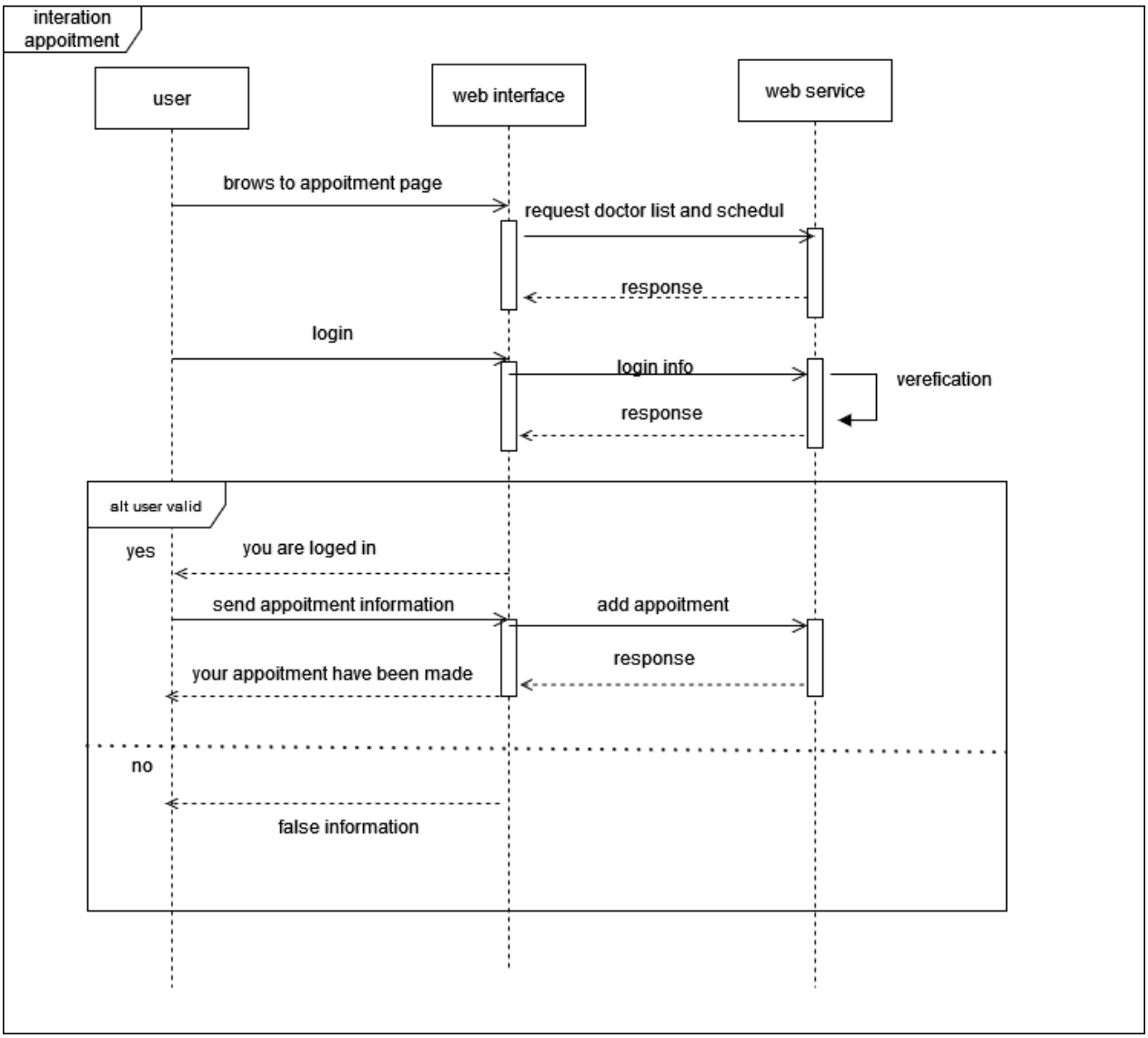


Figure 19: Sequence Diagram for Appointment

Name: appointment

Objective: User make an appointment

Actors: User

Pre-condition: User must LOGED IN

Nominal scenario:

- user brows to the appointment page

- The system displays the list of available doctors
- user insert all the information needed to make an appointment

Post-condition:

- Successful: the appointment has been made.
- Failure: return to the appointment page.

Extensions:

- Required fields not filled: the system asks to fill the empty fields.
- Invalid input, an error message displays.

1.4 Application's Behavior by Activity Diagrams:

1.4.1 Signup of Cabin by Activity Diagram:

To sign up as a cabin, in the application, the client must enter his true personal information and the papers that represent his work, then if the application administrator accepts the request, the cabin profile will be created and an acceptance email will be sent to the client.

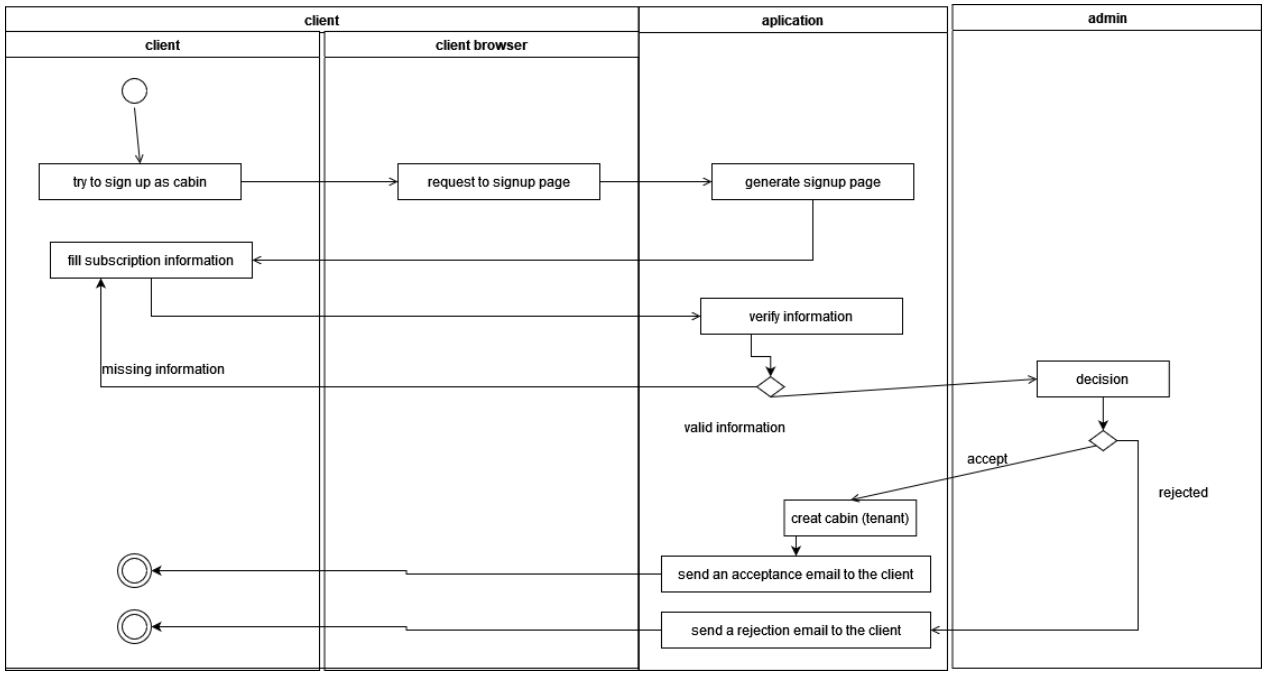


Figure 20: Activity Diagram of Cabin Subscription

1.4.2 Consultation Activity diagram:

After the doctor sees the appointment that made by the client e goes to the consultation page and examines the patient and creates an electronic prescription or send the patient to the nurse to take patient measurement ... before. After that, the doctor saves the consultation and the application notifies the nurse to take the cost.

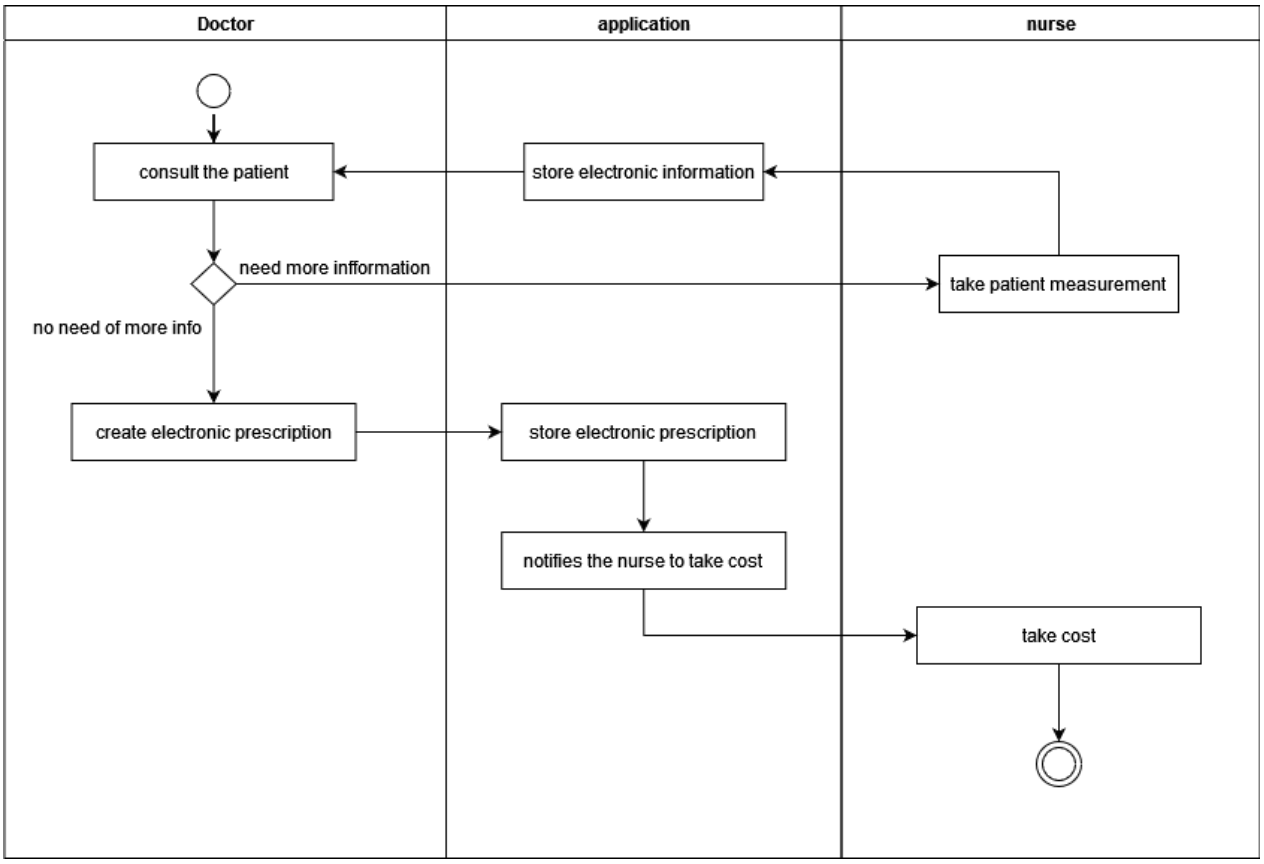


Figure 21: Activity Diagram of Consultation

1.5 Deployment Diagram:

- **User client:** the resource requester by web browser either by a computer operating system or by the mobile operating system.
- **A web server:** accepts and fulfills requests from clients for static content (i.e., HTML pages, files, images, and videos) from a website. Web servers handle HTTP requests and responses only.

- **An application server:** exposes business logic to the clients, which generates dynamic content
- **Database server:** used to provide database services like storing, processing and securing data

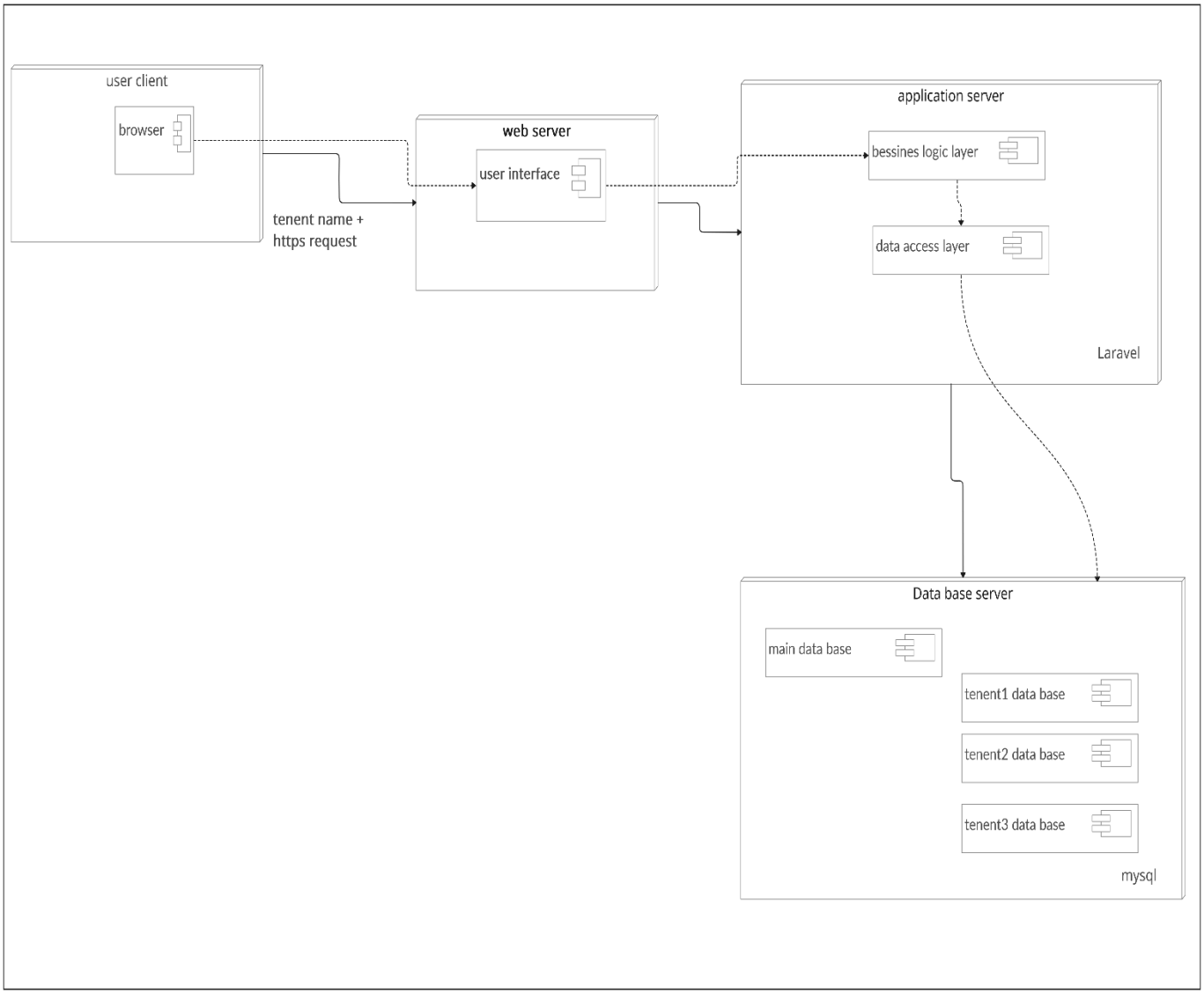


Figure 22: Deployment Diagram

1.6 Application's Design by Entity Relationship diagrams:

An entity relationship diagram (ERD) is a popular type of database diagram that clearly displays system entities and their internal relationships.

1.6.1 Entity Relationship Diagram for Cabin Relations:

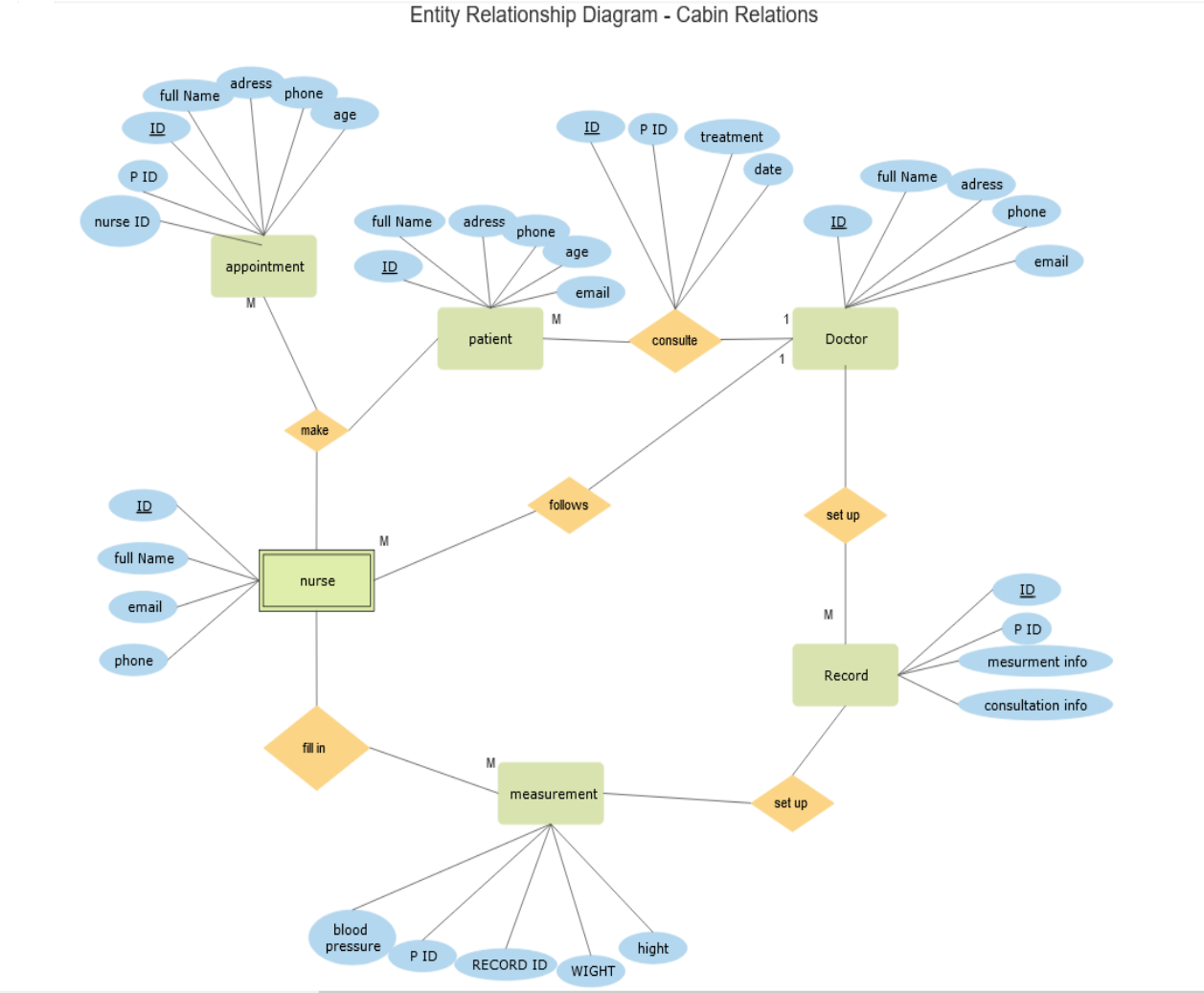


Figure 23: Entity Relationship Diagram for Cabin Relations

1.6.2 Entity Relationship Diagram Cabin Service applying:

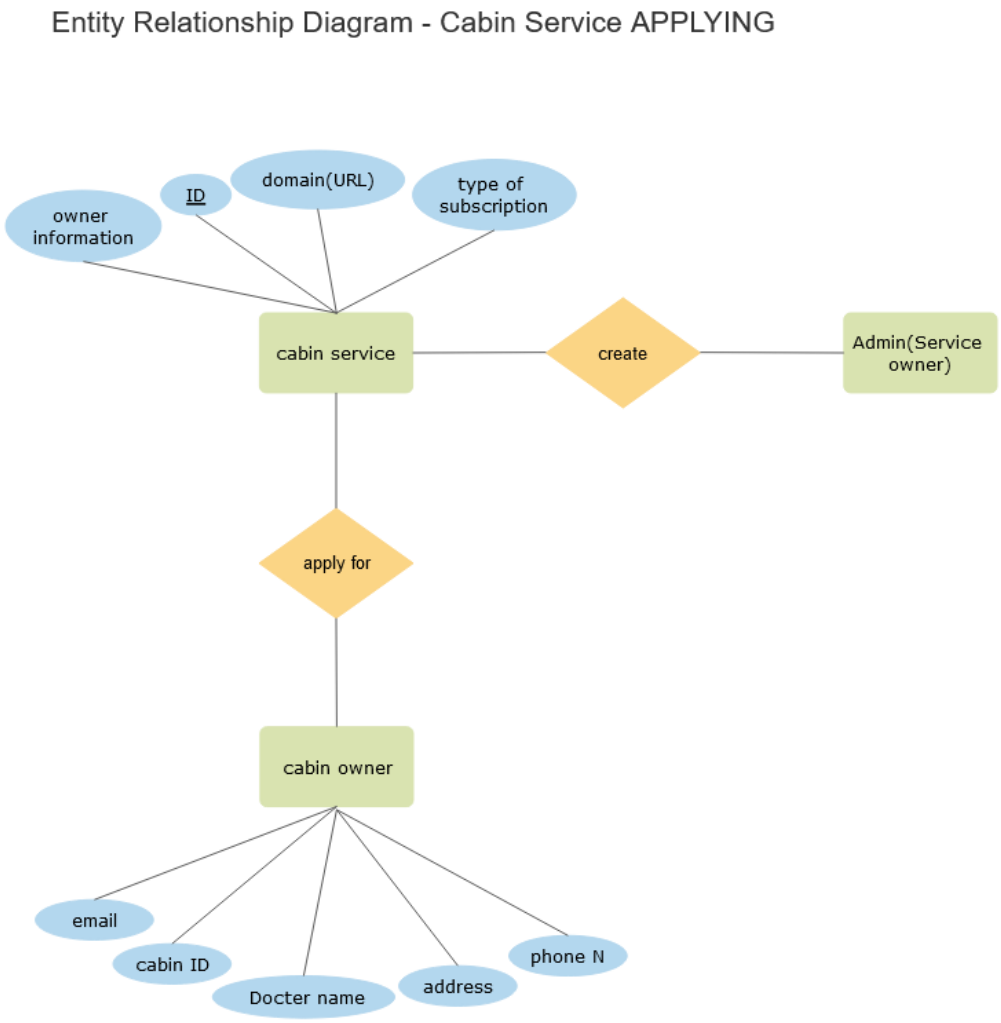


Figure 24: Entity Relationship Diagram Cabin Service applying

Conclusion:

This chapter presented a comprehensive set of diagrams, including use case, class, sequence, entity-relationship, deployment, and activity diagrams, to illustrate the design and architecture of the proposed system. These diagrams were created using the StarUML modeling tool and collectively provide a thorough understanding of the system's functionalities, static structure, dynamic behavior, data model, physical architecture, and business processes. By employing these industry-standard diagramming techniques with StarUML, the system's design is effectively communicated, facilitating its development, maintenance, and documentation throughout the project lifecycle.

CHAPTER 3

IMPLEMENTATION AND REALIZATION

Introduction:

This chapter explains the transformation of designs and specifications into a fully operational website. It describes the proposed system and the development environment, including the programming languages, frameworks, and tools employed for the realization of the project. Additionally, it provides an overview of the work accomplished, supported by screenshots that illustrate the various components and functionalities of the website. The chapter serves as a comprehensive guide to the implementation and realization phase.

1. Proposed System (web application as services):

1.1 Applications As a Service:

Applications as a service (AaaS) refers to the delivery of computer software applications as a service via the Internet. This type of software is also referred to as SaaS (Software as a Service), SaaS is one of three main categories of cloud computing, alongside infrastructure as a service (IaaS) and platform as a service (PaaS). A range of Definition information technology (IT) professionals, business users and personal users use SaaS applications. Products range from personal entertainment, such as Netflix, to advanced IT tools. Unlike IaaS and PaaS, SaaS products are frequently marketed to both Business to Business (B2B) and Business to Consumer (B2C) users. The SaaS vendor can either host the application on its own cloud infrastructure or use a cloud service provider like Amazon Web Services (AWS), Google Cloud, IBM Cloud, or Microsoft Azure. Users can access SaaS applications from any device with an internet connection. Most of its applications run directly in web browsers, while some may have dedicated mobile or tablet apps. And the main part is SaaS applications follow a **multi-tenant architecture**, meaning a single instance of the application serves all customers.

1.1.1 The key characteristics of AaaS are:

- Automated Provisioning: The users should be able to access the SaaS applications on the fly, which means the process of provisioning the users with the services needs to be automated. SaaS applications are typically used by B2B/B2C customers and this requirement demands creating companies/users just by invoking web services and provide the access credentials. Most of the SaaS applications provide this critical feature and a great example would be CREST API from Microsoft. Cloud Services Broker (CSB) platforms can automate this procedure to provide access to SaaS applications on demand basis. Another important characteristic is the de-provisioning ability - remove the access from the user/organizations whenever the customer decides not to use the Software as a Service applications. A good example for this is Salesforce, used by sales folks to manage the sales related operations. Typically, Salesforce tenant gets created for an organization with unique identification by invoking APIs of Salesforce. Another set of APIs are called to create users under the tenant and the access credentials are shared to user. Also delete API is called for when an organization decides to discontinue the application.
- Single Sign-On (SSO) is a mechanism that allows users to authenticate once and gain access to multiple applications or systems without having to log in separately for each one. It is particularly beneficial for enterprise organizations that use various Software as a Service (SaaS) applications
- Accessibility: AaaS applications are accessible from anywhere with an internet connection, enabling remote access and collaboration among distributed teams or mobile users.
- Data Security: Ensuring that the data/business information is protected from corruption and unauthorized access is very important in today's world. Since the Software as a Service applications are designed to be shared by different tenants, it becomes extremely important to know how well the data is secured. Certain types of data must be enabled with encrypted storage for a particular tenant and the same should not be accessible to another tenant. So, having a good Key Management Framework or ability to integrate/interface with external Key Management Frameworks becomes essential part of SaaS applications. Also integration with CASB (Cloud Access Security Brokers) system will increase the

confidence with respect to data security. A very strong Role Based Access Controls need to be ensured in order to protect the data.

- Automatic Updates: The service provider is responsible for maintaining, updating, and patching the applications. Users always have access to the latest version of the software without the need for manual upgrades or installations.

1.1.2 Examples of AaaS offerings include:

- Office Productivity Suites: Microsoft Office 365, Google Workspace (formerly G Suite).
- Customer Relationship Management (CRM): Salesforce, Zoho CRM.
- Enterprise Resource Planning (ERP): NetSuite, Oracle Cloud ERP.
- Business Intelligence and Analytics: Tableau Online, Microsoft Power BI
- Human Resource Management (HRM): Workday, BambooHR

1.2 Multi-Tenancy:

Multi-tenancy is an architecture in which a single instance of a software application serves multiple customers. Each customer is called a tenant. Tenants can be given the ability to customize some parts of the application, such as the color of the user interface or business rules, but they can't customize the application's code.

Multi-tenancy is a specific characteristic of cloud applications that can change the underlying economics of applications through sharing infrastructure, platform and services.it allows each cloud application or “tenant”, each with their own customers, processes and data, to obtain a single application instance [7].

There are two main types of multi tenancy data base and these two types are:

1.2.1 Single Database in Multi-Tenancy:

All tenants’ data is stored within a single shared database and each tenant’s data is logically separated using a tenant identifier. This approach minimizes hardware and backup costs because it serves multiple tenants using one database

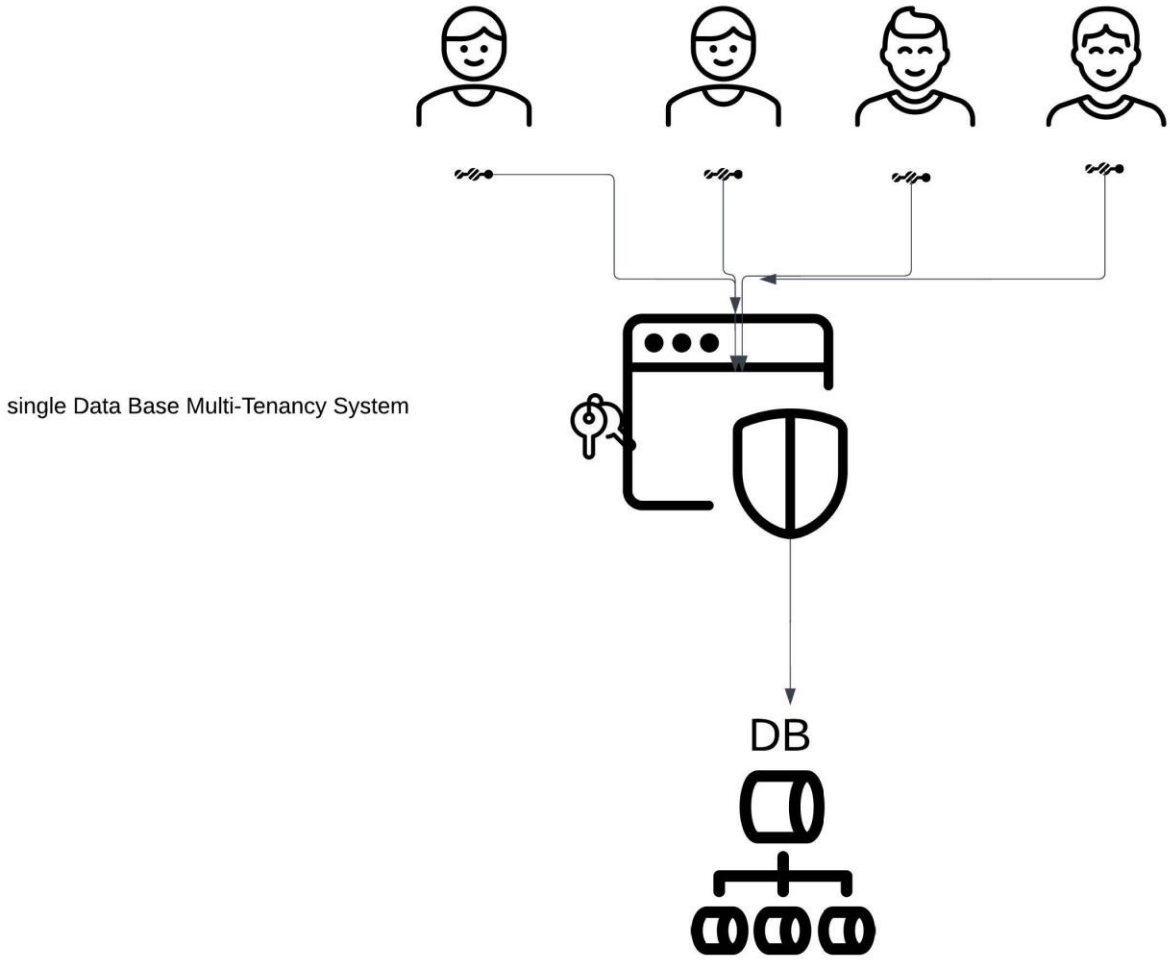


Figure 25: Single Data Base Multi-Tenancy System

1.2.2 Multiple Databases in Multi-Tenancy:

Every tenant will have their own database, and the application will have one common database to store tenant-specific information. Multiple databases provide stronger security and isolation, reducing the risk of data breaches and unauthorized access among tenants. This is the type we are going to use in our web application

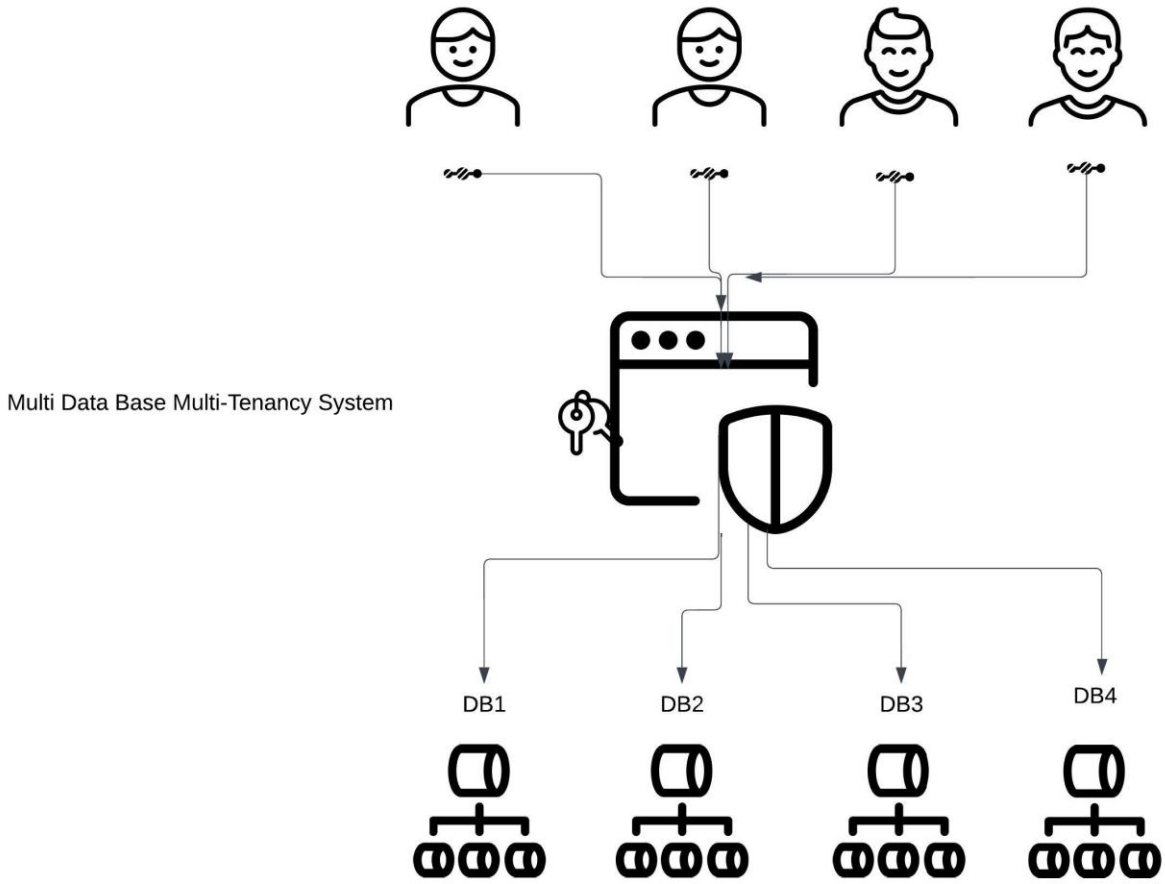


Figure 26: Multi Data Base Multi-Tenancy System

1.2.3 Benefits of using multi-tenant architecture:

- Cost Efficiency: Multi-tenancy allows for efficient resource sharing and utilization. A single instance of the application serves multiple tenants, reducing the need for redundant hardware, software, and maintenance costs. This leads to lower operational costs for the service provider, which can be passed on to customers in the form of lower pricing.
- Scalability: With multi-tenancy, the application can scale more easily to accommodate a growing number of tenants. Additional resources can be allocated to the shared instance as needed, without the need to provision separate instances for each new tenant. This scalability helps service providers cater to fluctuating demand and rapid growth.
- Ease of Maintenance and Updates: Since there is a single codebase serving all tenants, updates, bug fixes, and new feature releases can be rolled out simultaneously to all tenants.

This simplifies maintenance efforts and ensures that all tenants benefit from the latest improvements and security patches without the need for individual deployments.

- Faster Deployment and Onboarding: New tenants can be quickly onboarded and provisioned with the existing application instance, eliminating the need for separate installations or deployments. This accelerates time-to-market and reduces the overhead associated with setting up new instances for each tenant.
- Centralized Management: Multi-tenant architecture allows for centralized management and monitoring of the application, its resources, and tenant configurations. This simplifies administration, monitoring, and troubleshooting efforts, as everything is managed from a single control plane.
- Increased Collaboration and Data Sharing: In some multi-tenant scenarios, tenants may benefit from controlled data sharing or collaboration features within the application. This can foster a community-like ecosystem and enable cross-tenant interactions or data exchange when desired.
- Easier Security: Having separate data bases means lowering the chance of hacking full data base and taking all the data, Despite the daunting complexity of cloud environments, major cloud services are generally pretty secure, and for customers it can be as easy as activating the cloud provider's own security controls and tools

2. Presentation of the System:

2.1 Presentation of web pages:

Home page:

The home page is the first page the visitor sees when he joins the website and it's for doctors and patients, the doctors can see all the offers we have than they can apply for an offer, the patient can register for free. In the home page the visitor can see the rating of the doctors and their feedback on the platform and can search for doctors

Figure 27: Home page

Cabin offers page:

The offers page contains all the offers available for creating the cabin service and the doctor have to choose an offer then he going to be directed to an applying page

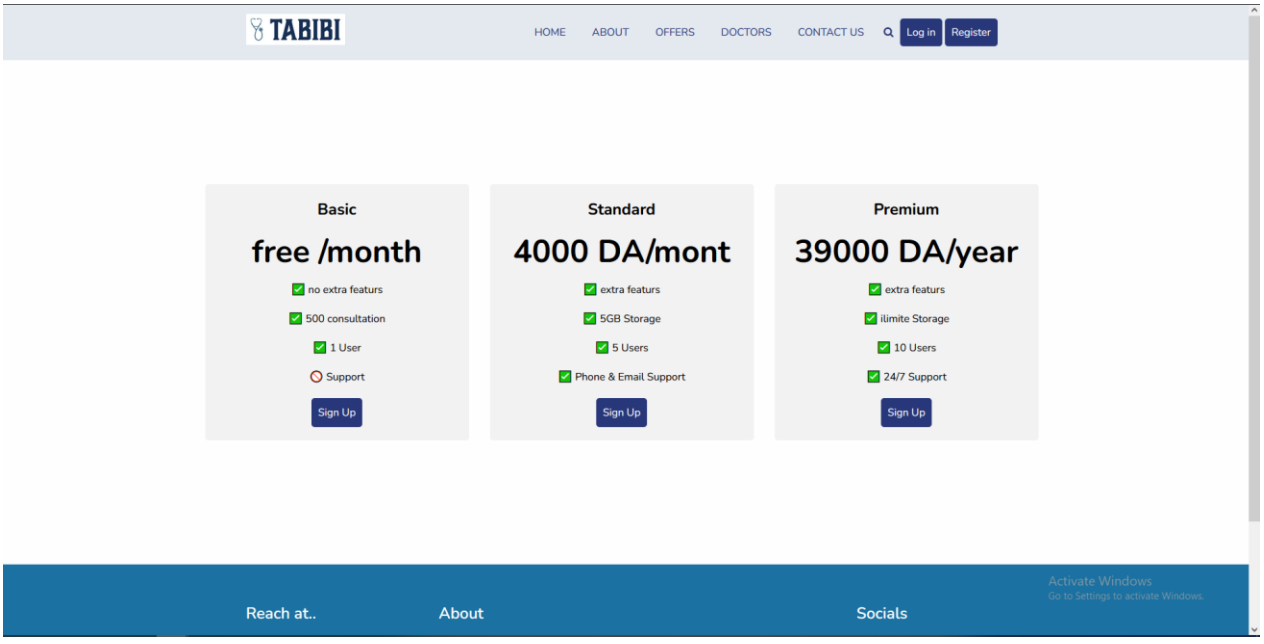


Figure 28: Cabin offers page

Applying for cabin:

In the applying page the doctor fills cabin name and his full name email phone number and identity documents

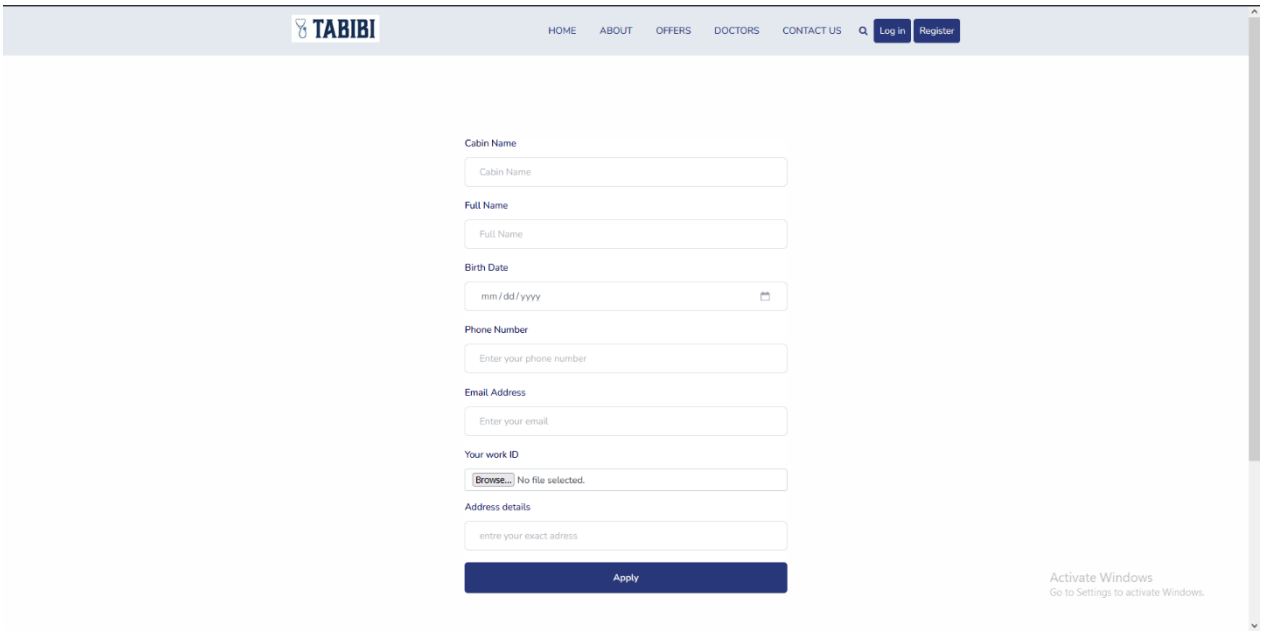


Figure 29: Applying for cabin page

Doctors in our platform:

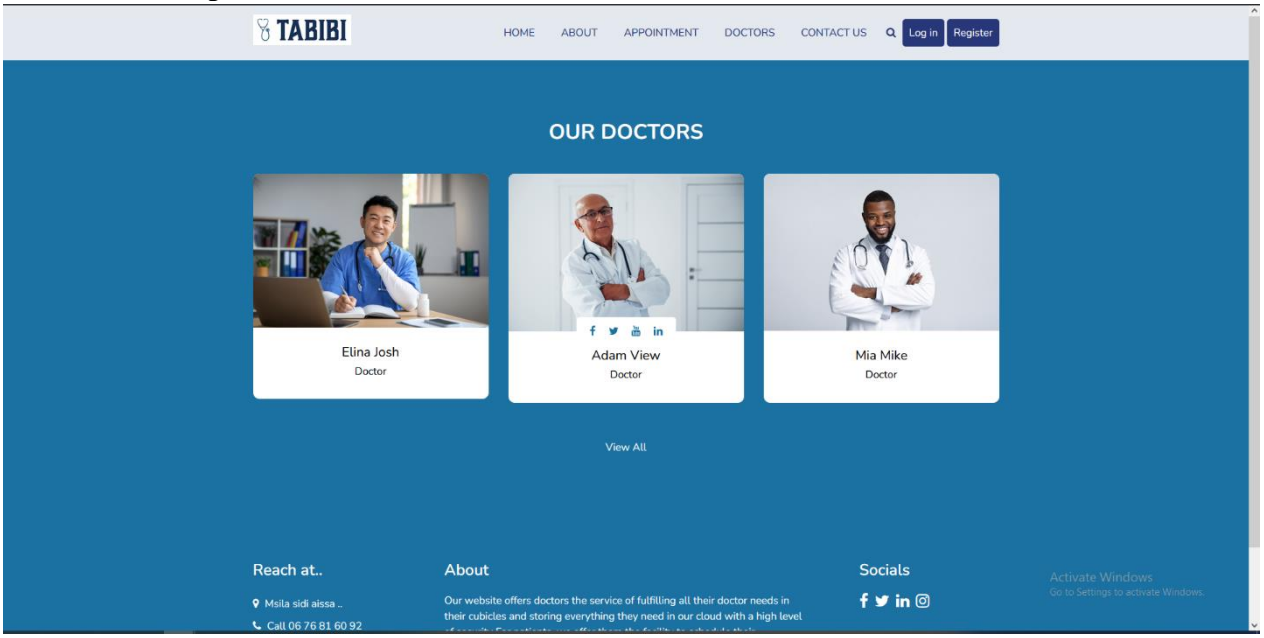


Figure 30: Doctors

About:

In about us page we have an overview of what we offer in our platform and some reviews of the doctors that uses our platform

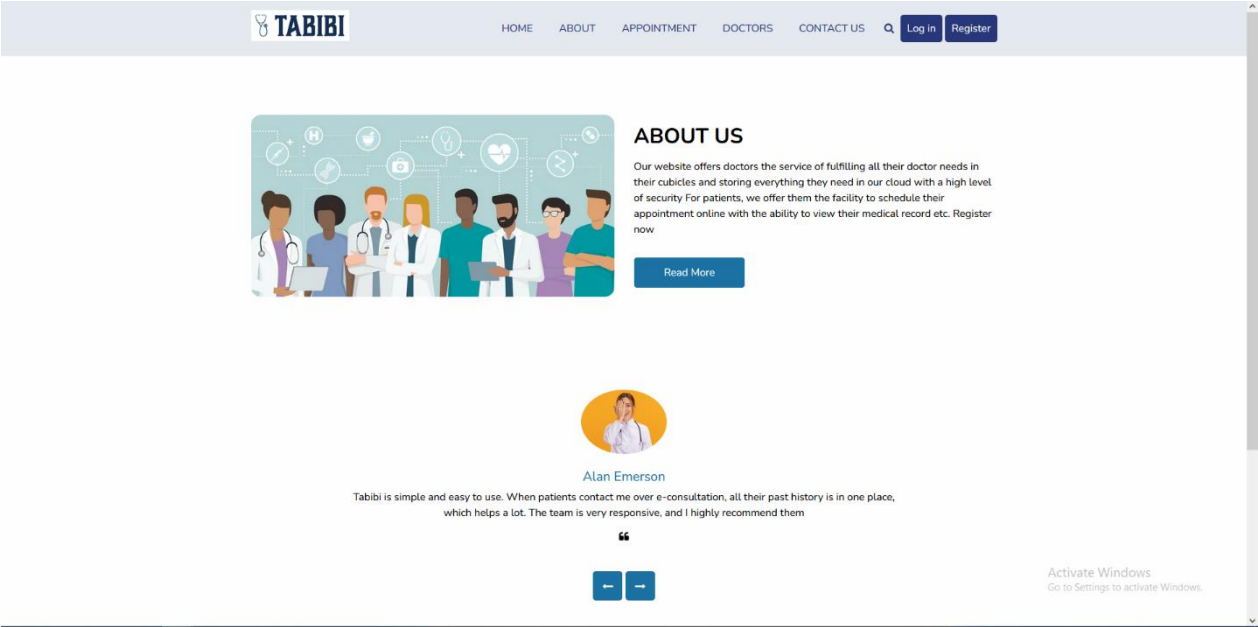


Figure 31: About web page

Login:

In the login page the design is the same the patient will login by pressing the login button in any page of the website but the Doctors will login through the Sub Domain after receiving the confirmation email:

Figure 32: Login page

Owner of platform:

Owner of platform Dashboard:

In the dashboard of the owner of platform (Admin) he can see how many Cabins in the work and how many patients registered in the platform the sales, how many visitors visited the website and a cloud usage overview

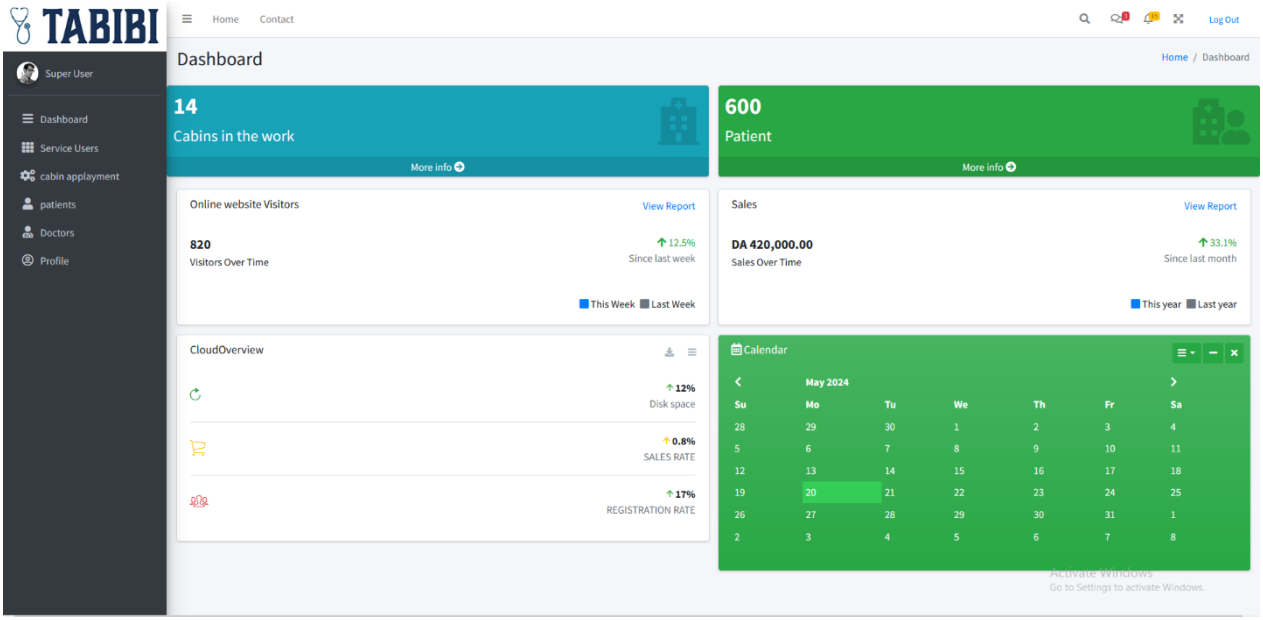


Figure 33: Admin Dashboard

Service Users:

In service users page the admin can see all the cabins that are in work with their domain for fast access and can delete a cabin service or add new cabin service

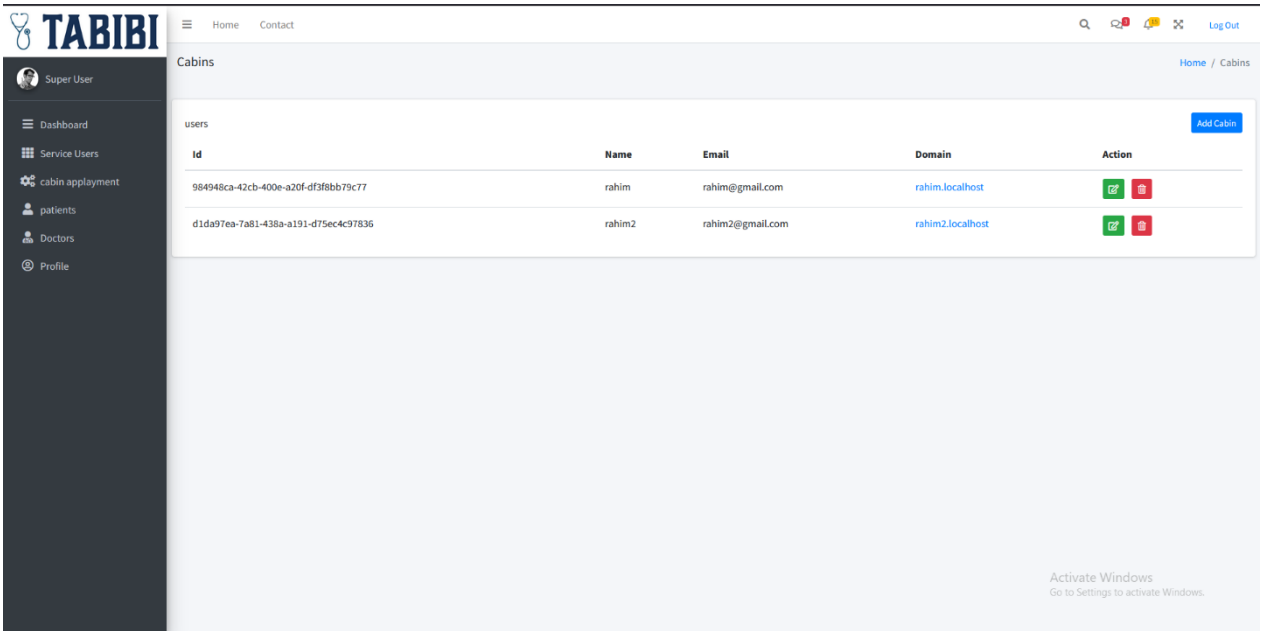


Figure 34: Service users (cabins) page

Cabin Applicants for the service dashboard:

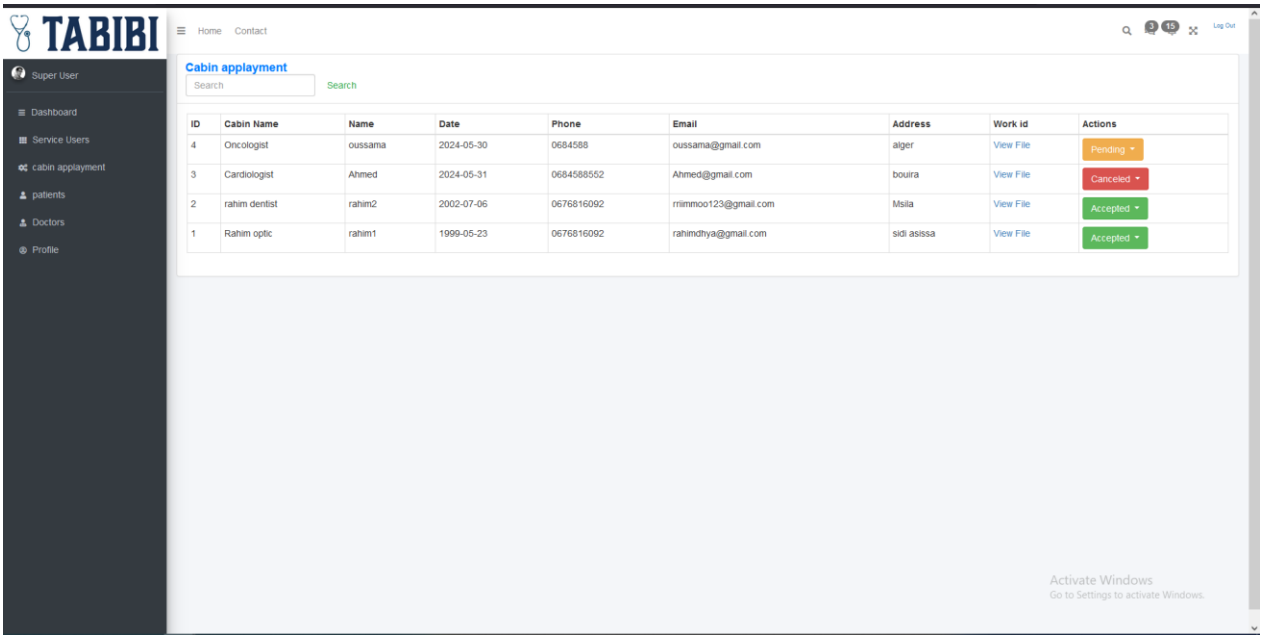


Figure 35: who applied for cabin dashboard

Patient list:

In patient page list the admin can see all the patients registered in the platform and can see the details of any one:

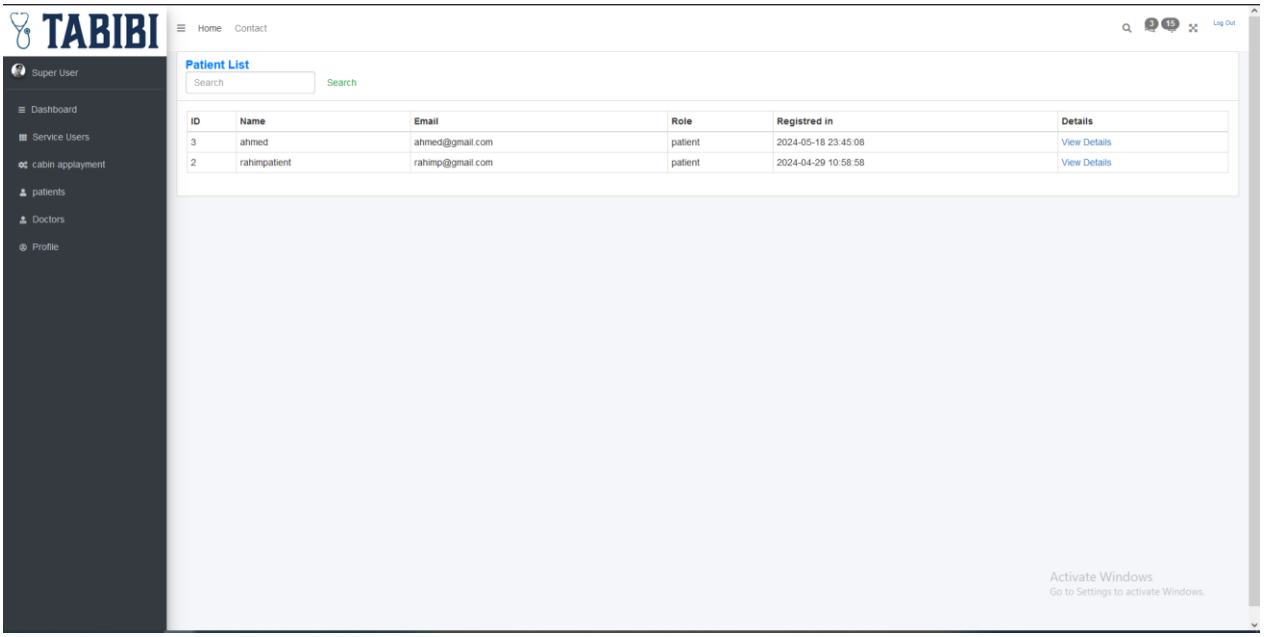


Figure 36: patient List page

Doctor List:

In this page where the admin sees all the doctors that accepted an are using the platform

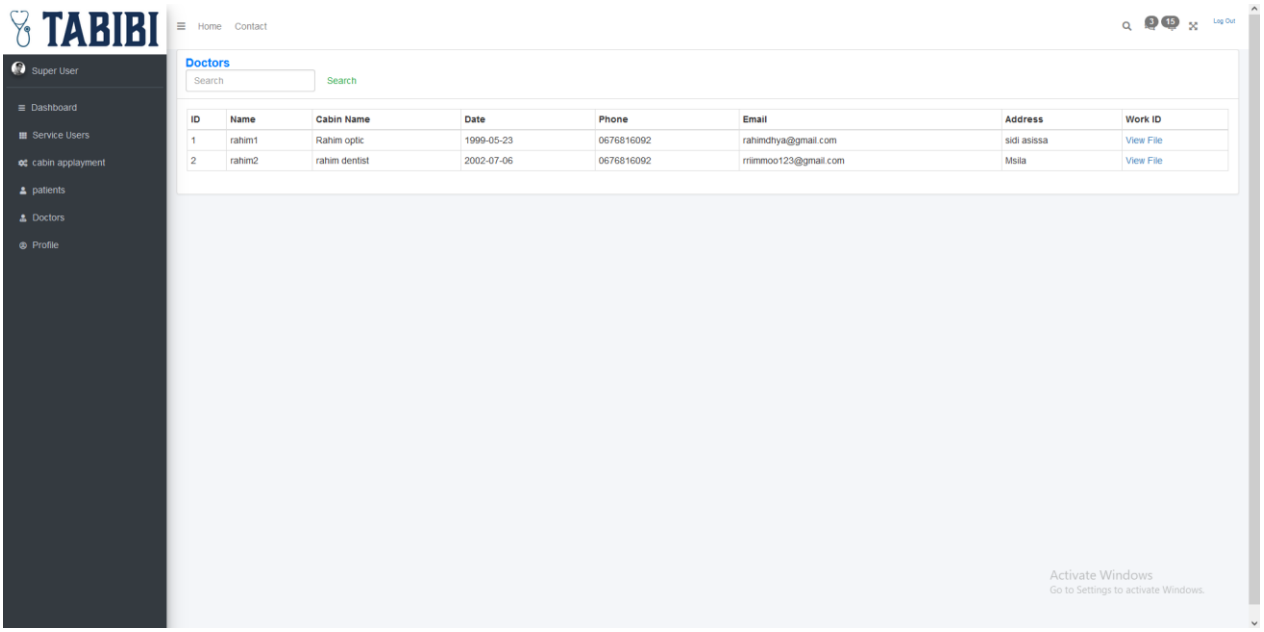


Figure 37: ADMIN Doctor List Dashboard

Patient Pages:

Make appointment for patient page:

In this page the user has to be logged in to see the page and then he can make appointment for him or for someone under his account

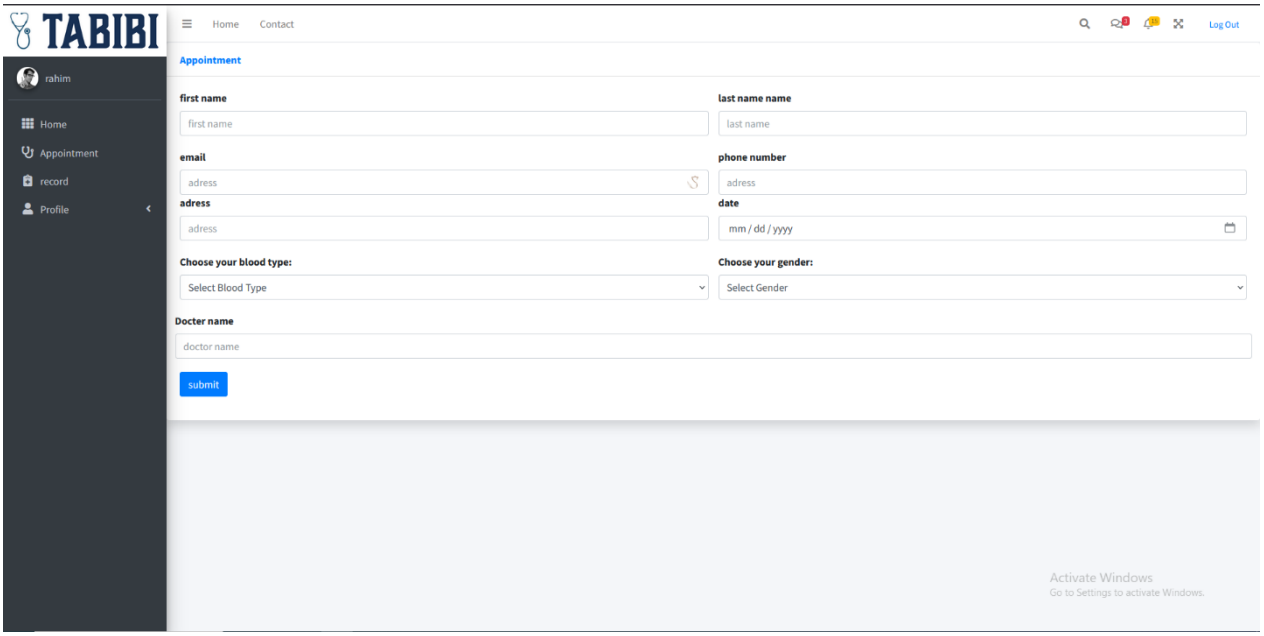


Figure 38: patient making appointement page

Patient consultation record page :

In this page the patient can see all his records from all the doctors he consulted at their cabin

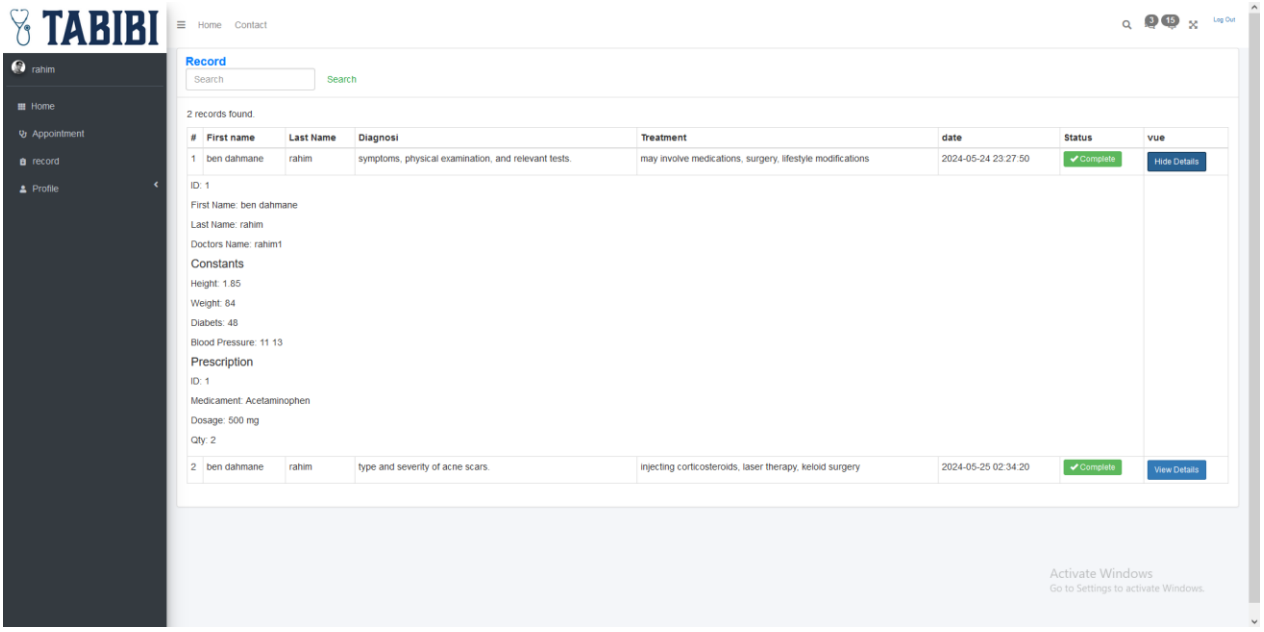


Figure 39: patient health record page

Doctor pages:

Waiting room:

In this page the doctor sees all the patient that the nurse approved that they are here and waiting for consultation and this page can also the nurse sees it and can edit the measurement for the patient

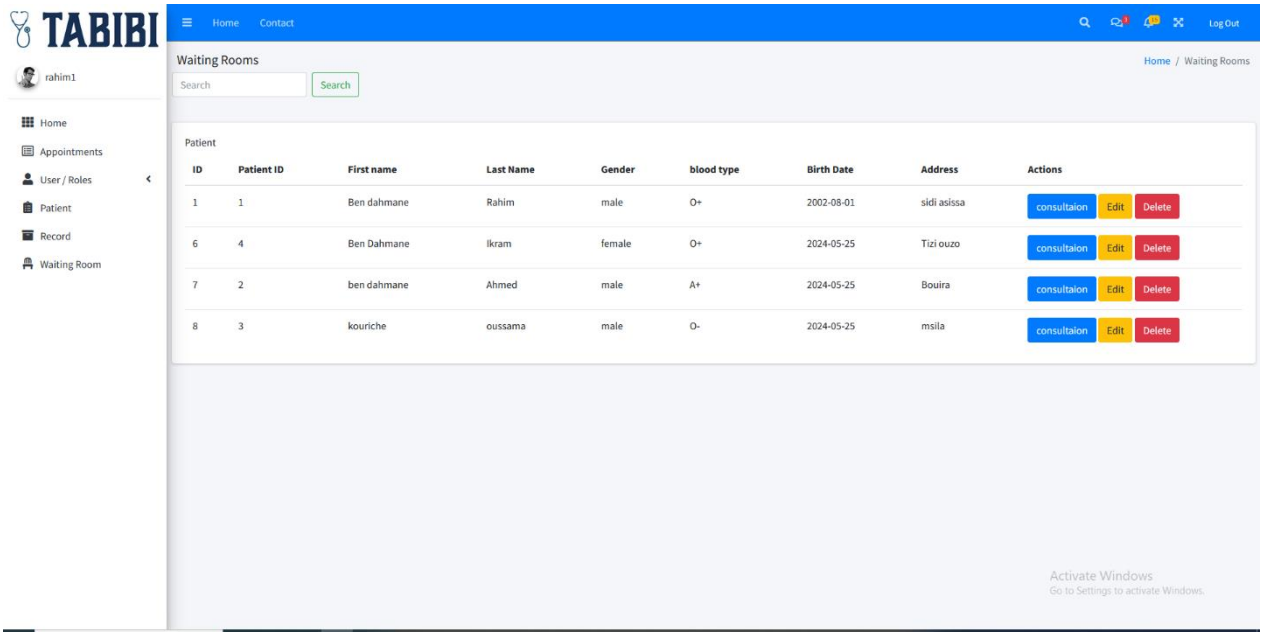


Figure 40: doctors waiting room page

Consultation page:

In this page where the doctor fills the diagnosis and Treatment

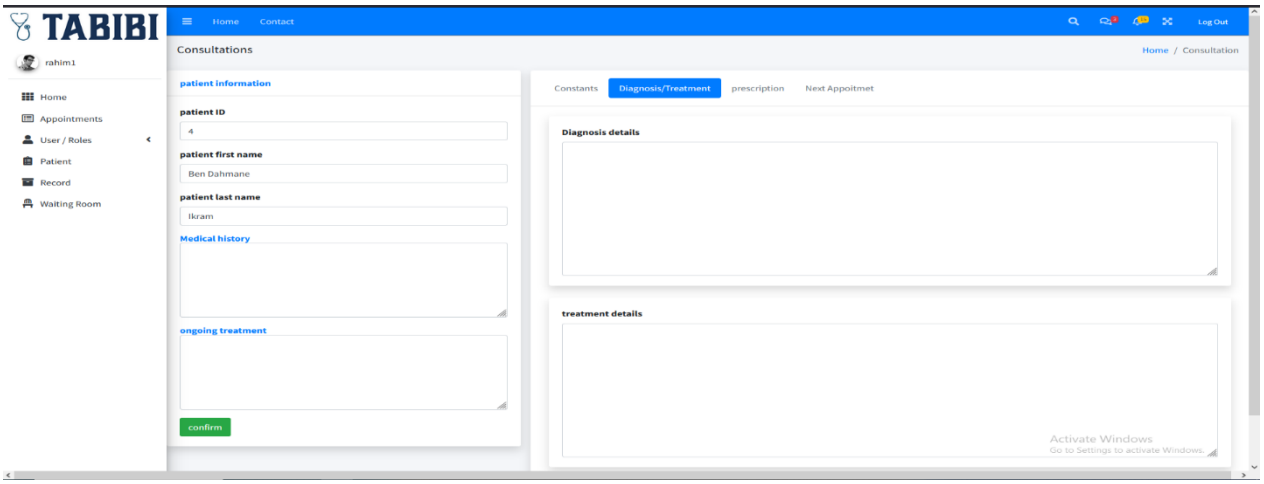


Figure 41: Consultation page (diagnoses and treatment)

Here where the doctor fills the medicaments and their dosage and quantity and everything in the page with autocomplete

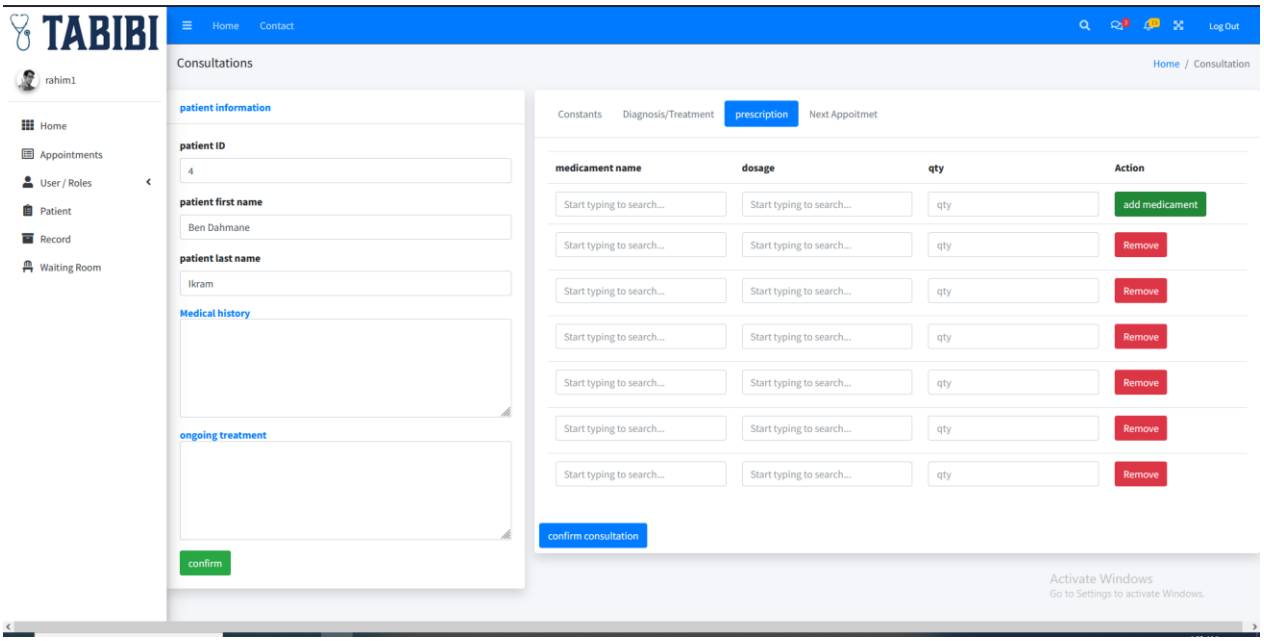


Figure 42: Consultation page (prescription)

Record page :

In this page where the doctor sees all the consultation, he did for all the patient.

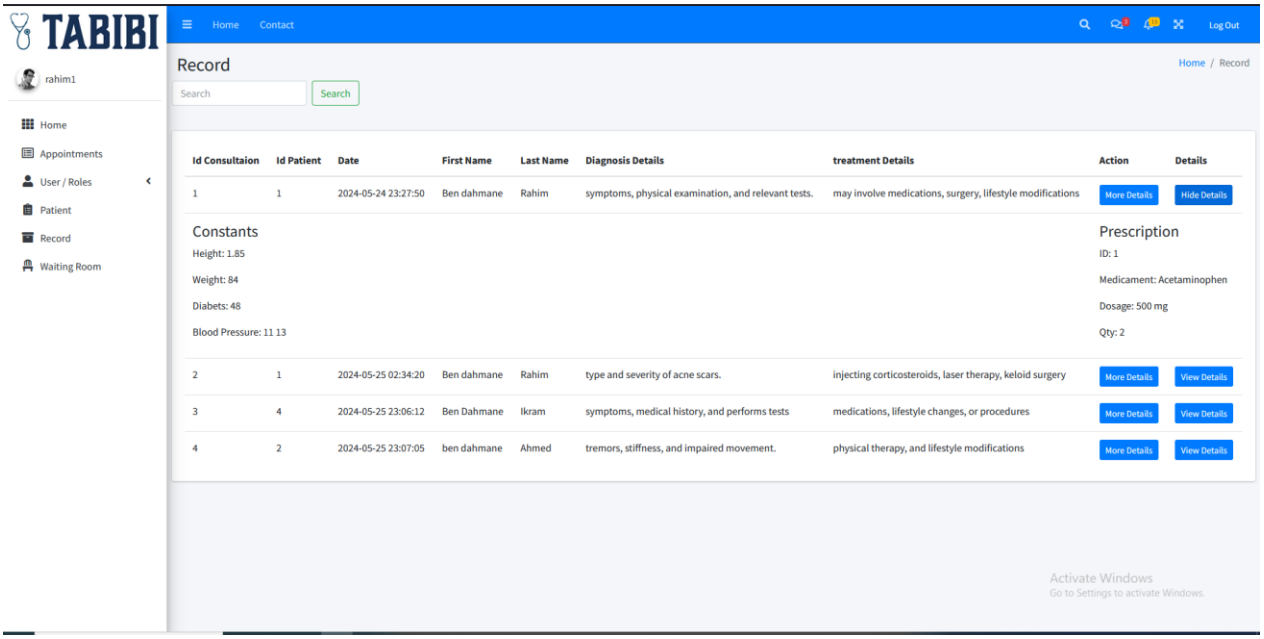


Figure 43: Doctor consultation Record

Nurse pages:

Appointment List:

In this page the nurse sees all the appointment that the users that already have an account and made an appointment online and then can send them to the waiting room

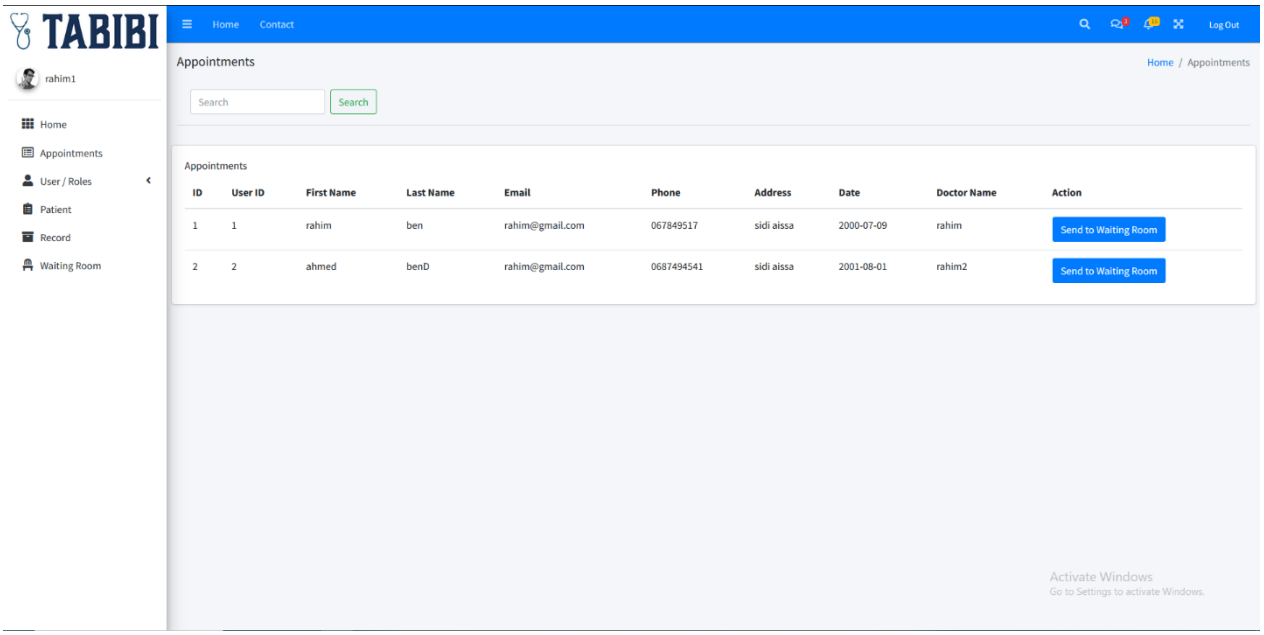


Figure 44: List of appointments

Patient that didn't make an appointment online page:

In this page the nurse will have to add the patients that didn't make an appointment online so it add all the information of them then send them to waiting room

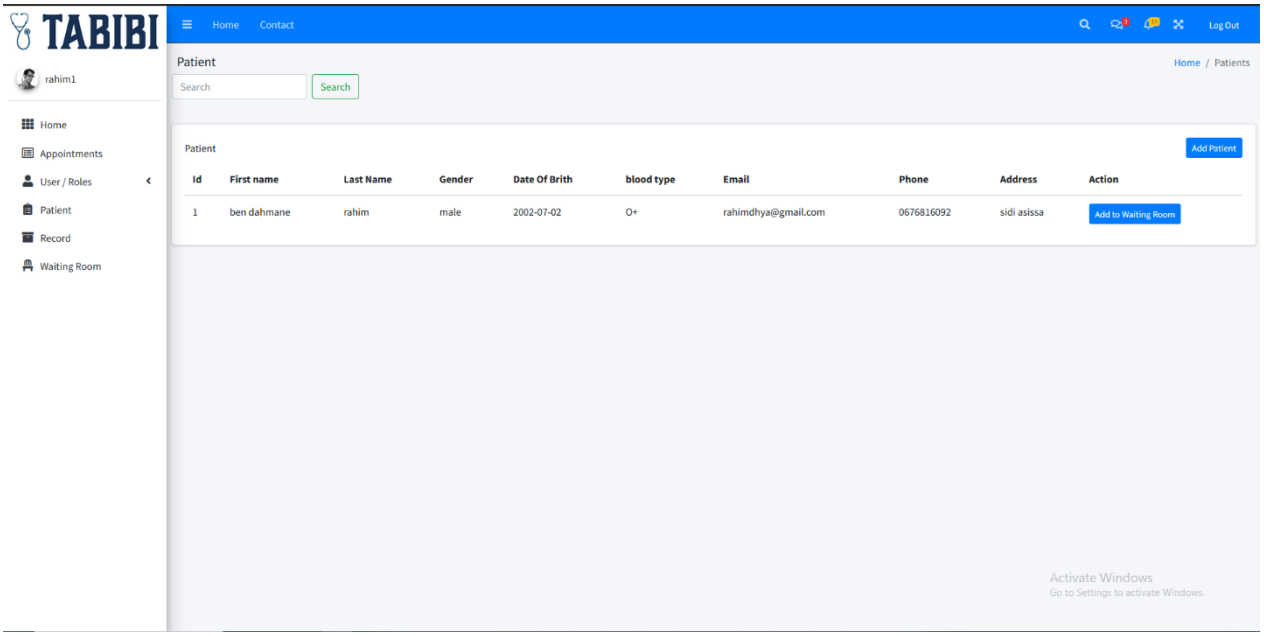


Figure 45: page for patient that don't have an account

Here where the nurse adding the patient that don't have an account

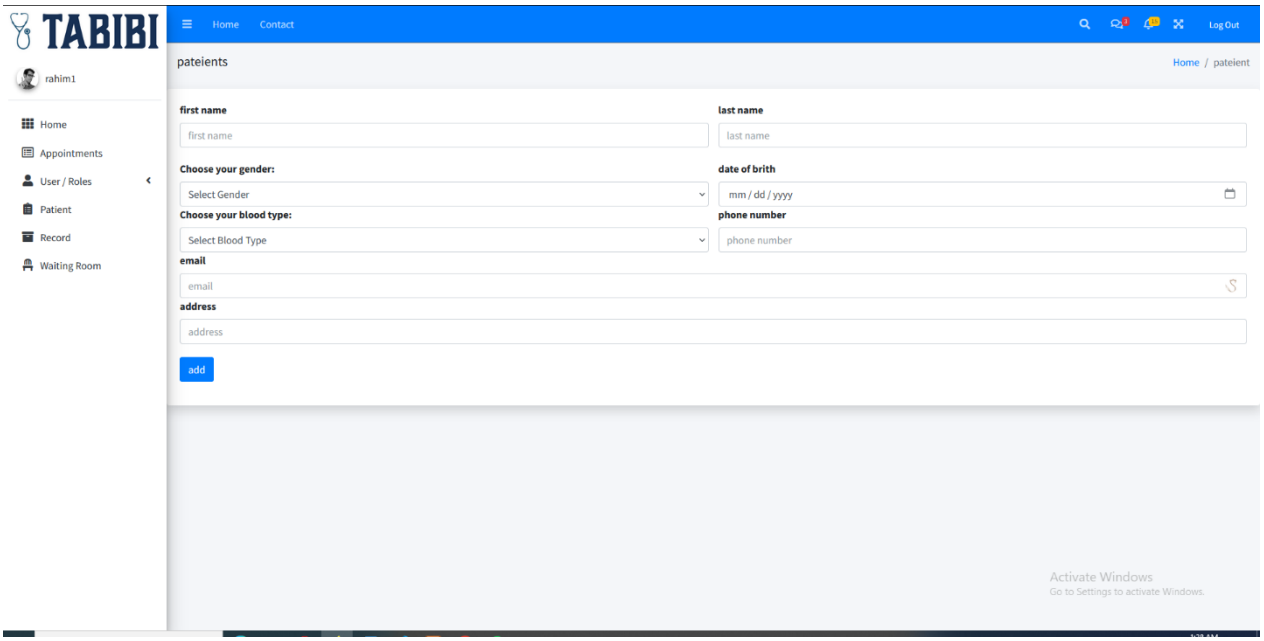


Figure 46: page for adding patient that don't have an account

List of the users in the cabin

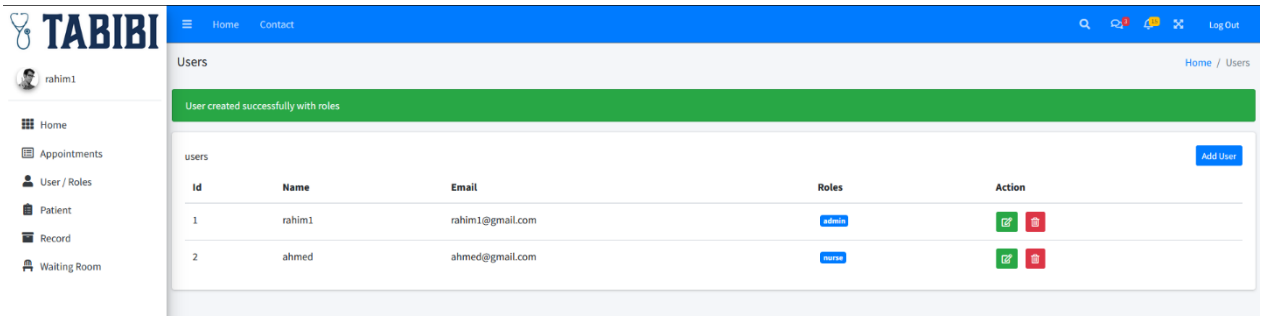


Figure 47: List of the users in the cabin page

The admin of the cabin adds nurse account page

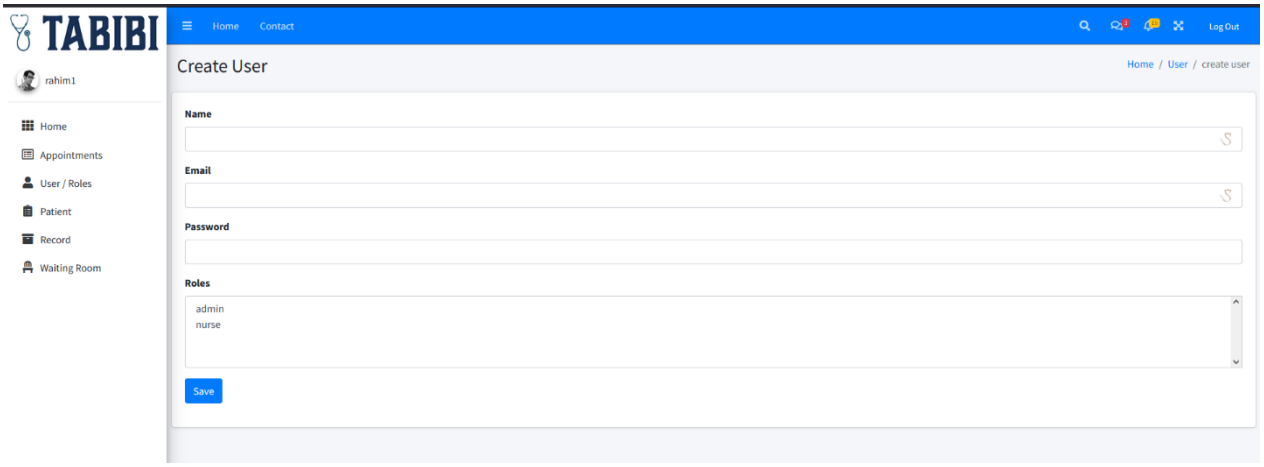
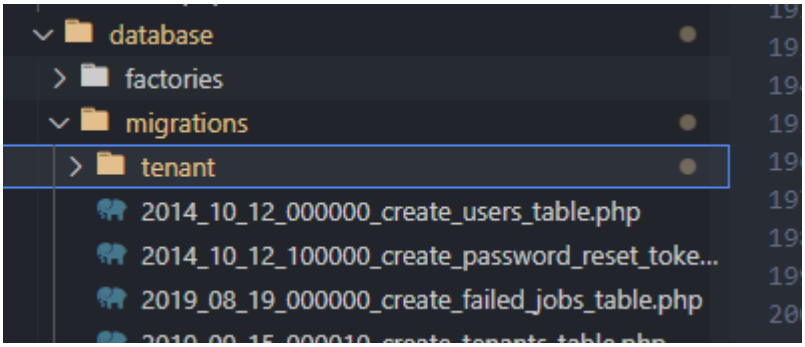


Figure 48: adding nurse account page

2.2 Multi-Tenancy System:

2.2.1 Data base for each tenant:

So, we know that each tenant has his own data base and its automatically created, we have to defined the tables of the tenants to make the migration so any table we wanted to be in the tenant data base we make it in file called tenant inside the migrations and we defined the tables we want in that file



Thene to make the automatic migration command we do this in the config tenancy

```
184     /**
185      * Parameters used by the tenants:migrate command.
186      */
187     'migration_parameters' => [
188         '--force' => true, // This needs to be true to run migrations in production.
189         '--path' => [database_path('migrations/tenant')],
190         '--realpath' => true,
191     ],
192
```

2.2.2 Tenant creation:

Each tenant will have his own name and email and sub domain and a password, so after the admin fills this information, the tenant will be created and will have his unique id that's by an id generator of tenancy then an email will be sent to the cabin manager with all the information.

```php
/**
 * Store a newly created resource in storage.
 */
public function store(Request $request)
{
    // validation
    $validatedData = $request->validate([
        'name'        => 'required|string|max:255',
        'email'       => 'required|email|max:255',
        'domain_name' => 'required|string|max:255|unique:domains,domain',
        'password'    => ['required', Rules\Password::defaults()]
    ]);

    $tenant = Tenant::create($validatedData);

    $tenant->domains()->create([
        'domain' => $validatedData['domain_name'] . '.' . config('app.domain')
    ]);

    return redirect()->route('tenants.index');
}
```

Figure 49: Tenant Creation

2.2.3 Tenant Routs:

Here we have all the routs we used in our platform:

```php
22
23 > Route::middleware([···
27 > ])->group(function () {···
55   });
56
```

Figure 50: Tenant Routs

2.3 SaaS Cloud Used:

We are using Aiven Cloud to host our data bases so we got the free test and they gave us a space in their cloud:

Figure 51: Aiven cloud service used in our platform

2.3.1 connection information:

The server hosted in Germany and we are using MySQL and as we see in the picture the server requires Secure Sockets Layer (SSL) mode with a CA certificate

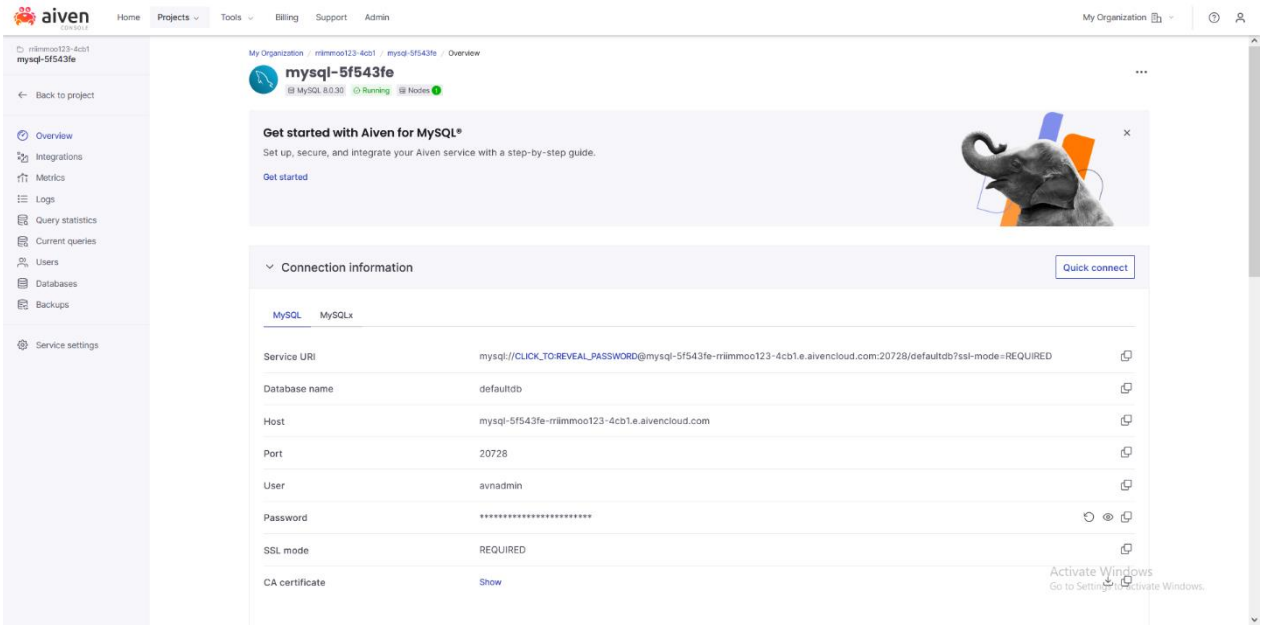


Figure 52: Aiven cloud connection information

2.3.2 Servers Overview:

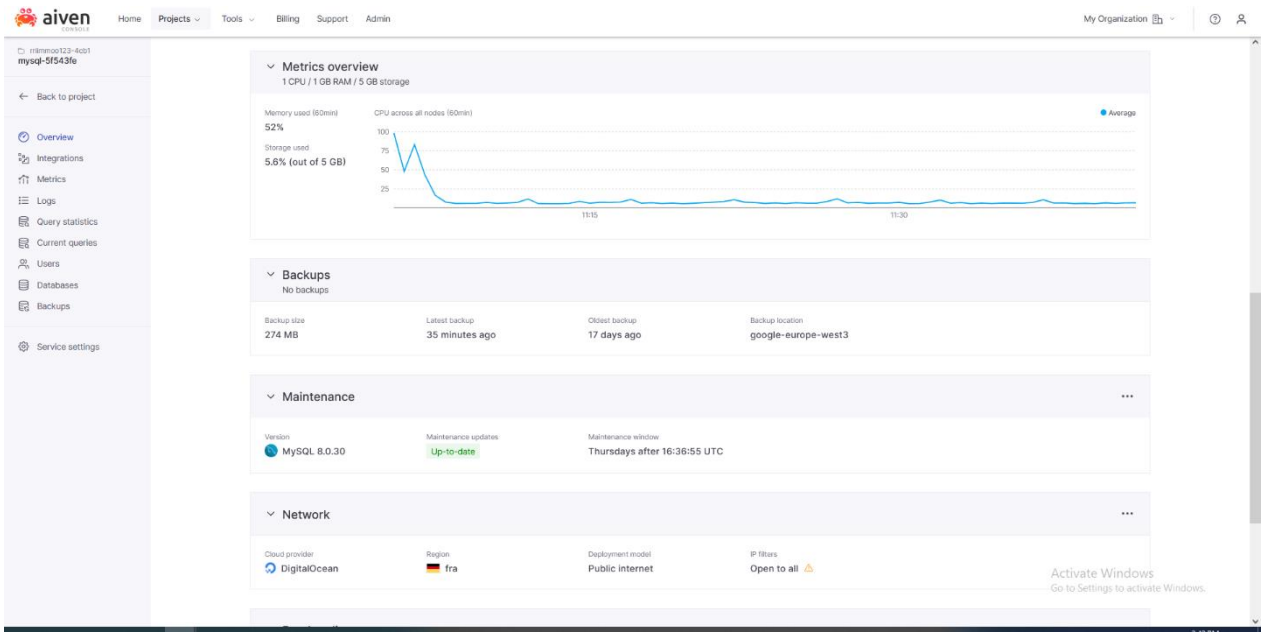


Figure 53: Cloud Server Overview

2.3.3 Cloud data bases:

First after the migration we have only the main data base called(laravel_multi_tenant):

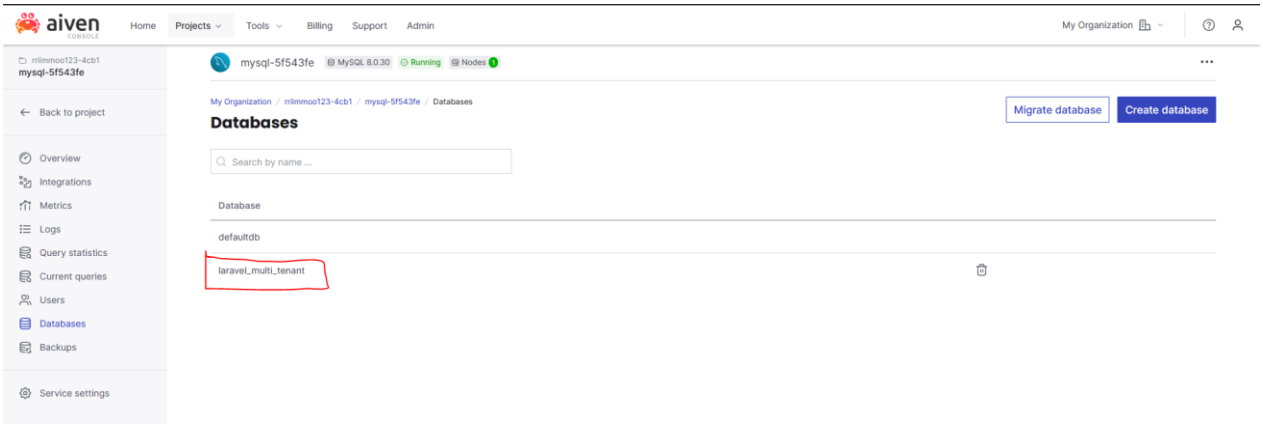
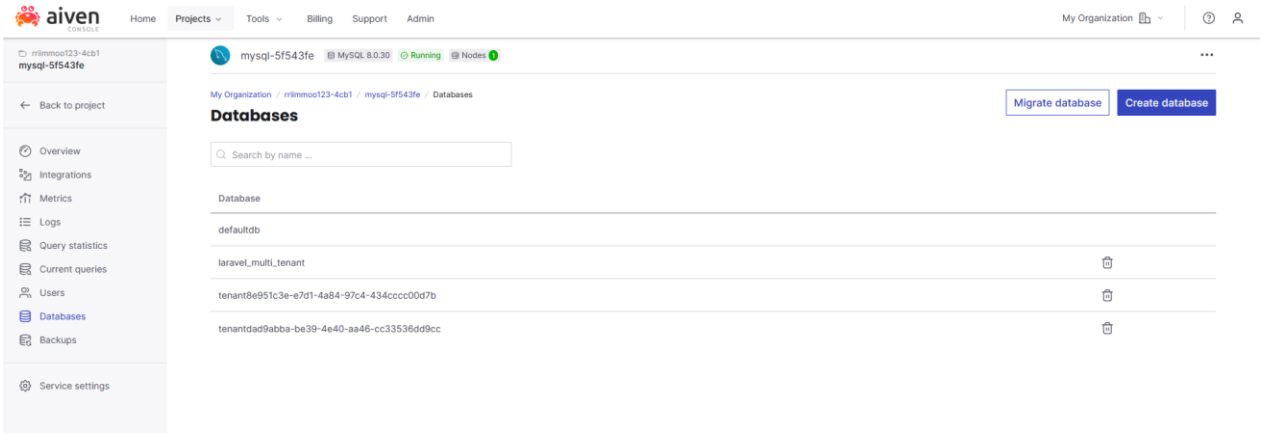


Figure 54: Cloud Data Base

Now after creating 2 cabins (tenants) we will see that each tenant has his new data bases added



2.3.4 Displaying database:

To see and display all the tables of data base we need to use MySQL Workbench and connect to the MySQL server that we created in aiven and we get like this:

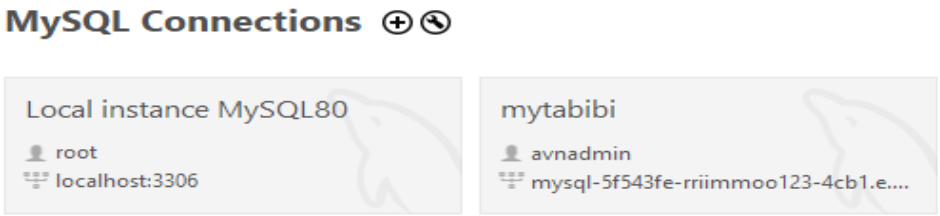


Figure 55: MySQL Workbench Dashboard

And after connecting we can see all the databases we have and all their tables

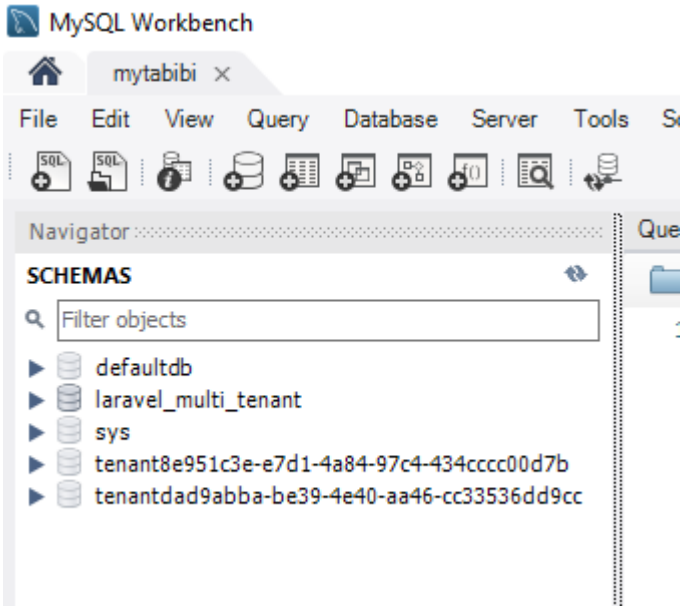


Figure 56: MySQL Workbench to display data bases

2.4 Security

2.4.1 Authentication:

Authentication is the process of determining whether someone or something is who or what they say they are. Authentication technology provides access control for systems by checking to see if a user's credentials match the credentials in a database of authorized users or a data authentication server. In doing this, authentication ensures that systems, processes and enterprise information are secure.

```php
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\View\View;

class AuthenticatedSessionController extends Controller
{···
}
```

```php
App\Http\Controllers\Controller
<?php
class Controller extends Controller
{
    use AuthorizesRequests,
        ValidatesRequests;
}
```

Figure 57: Authentication

2.4.2 Authorization:

Determine if the user is authorized to make the request.

Here is an example:

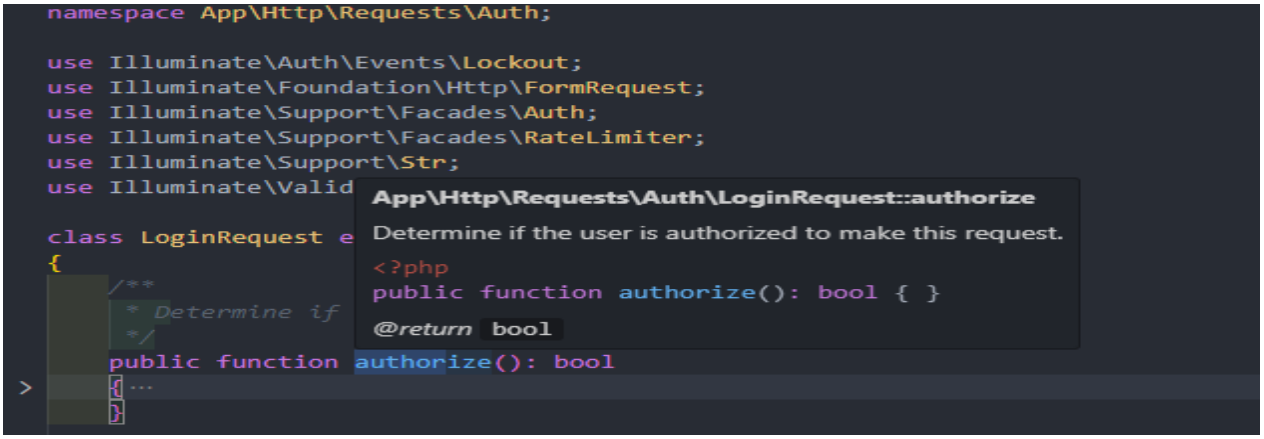


Figure 58: Authorization

2.4.3 Anti XSS (Cross Site Scripting):

In an XSS attack, attackers look for vulnerabilities on a website that will let them inject malicious scripts into the website. So, we made a middleware that will block users to post scripts and HTML code by default.

Figure 59: Anti XSS

2.4.4 CSRF (Cross Site Request Forgery):

Thankfully, Laravel makes it easy to protect your application from cross-site request forgery (CSRF) attacks. Laravel automatically generates a CSRF "token" for each active user session managed by the application. This token is used to verify that the authenticated user is the person actually making the requests to the application. Since this token is stored in the user's session and changes each time the session is regenerated, a malicious application is unable to access it.so anytime we define a "POST", "PUT", "PATCH", or "DELETE" HTML form in our application, we just include a hidden CSRF _token field in the form so that the CSRF protection middleware can validate the request.

The middleware:

```php
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;

class VerifyCsrfToken extends Middleware
{…
}

```

Figure 60: CSRF Middleware

Example of how we implement it:

```blade
<form method="POST" action="/profile">
    @csrf

    <!-- Equivalent to... -->
    <input type="hidden" name="_token" value="{{ csrf_token() }}" />
</form>
```

2.4.5 secure Session and XSRF-TOKEN cookies:

Session cookies contain data about a user’s session, including the user’s ID and the time of the last request. These cookies are essential because HTTP applications are stateless; they do not store information from previous visits to the site. The XSRF-TOKEN is passed along with each request

to counter Cross-site Request Forgery (XSRF) attacks. To prevent cookie stealing you must allow HTTPS requests only.

```
160     /*
161     |--------------------------------------------------------------------------
162     | HTTPS Only Cookies
163     |--------------------------------------------------------------------------
164     |
165     | By setting this option to true, session cookies will only be sent back
166     | to the server if the browser has a HTTPS connection. This will keep
167     | the cookie from being sent to you when it can't be done securely.
168     |
169     */
170
171     'secure' => env('SESSION_SECURE_COOKIE'),
172
```

Figure 61 Configure HTTPS Protocol

```
<?php

namespace App\Http\Middleware;

use Illuminate\Cookie\Middleware\EncryptCookies as Middleware;

class EncryptCookies extends Middleware
{
}
```

Figure 62: Cookie middleware

2.4.6 Preventing Open Redirection Attacks:

Removing the user's ability to change the value of the URL is possibly the most secure way to fix this. In this case, we simply remove the variable redirection and always send the user to his dashboard page after logging in. So, your confirm Login would then be updated to something like this:

```php
/**
 * Handle an incoming authentication request.
 */
public function store(LoginRequest $request): RedirectResponse
{
    $request->authenticate();

    $request->session()->regenerate();

    if ($request->user()->role == '1'){
        return redirect('user/dashboard2');

    }
```

Figure 63: redirect function to prevent redirection attacks

2.4.7 Blocking Brute Force Attacks:

A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works.

To prevent this type of attack the user account locked after a specific number of logins attempts which are 5 attempts if it passes, he will get 1 min lockout

```php
55      /**
56       * Ensure the login request is not rate limited.
57       *
58       * @throws \Illuminate\Validation\ValidationException
59       */
60      public function ensureIsNotRateLimited(): void
61      {
62          if (! RateLimiter::tooManyAttempts($this->throttleKey(), 5)) {
63              return;
64          }
65
66          event(new Lockout($this));
67
68          $seconds = RateLimiter::availableIn($this->throttleKey());
69
70          throw ValidationException::withMessages([
71              'email' => trans('auth.throttle', [
72                  'seconds' => $seconds,
73                  'minutes' => ceil($seconds / 60),
74              ]),
75          ]);
76      }
```

Figure 64: Blocking Brute Force Attacks

3. Presentation of the Development Tools:

3.1 Visual Studio Code:

"Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET)." [8]

3.2 Aiven Data Base:

"Aiven is a versatile platform empowering you with AI-driven workload optimization and control over your data. Deploy widely adopted technologies across multiple clouds with just a few clicks to stream, store, and serve your data." [9]

3.3 MySQL Workbench:

"MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more. MySQL Workbench is available on Windows, Linux and Mac OS X. [10]

3.4 Laravel:

"Laravel is a comprehensive PHP web application framework known for its expressive syntax, robust features, and developer-friendly environment. It simplifies web development by offering modular architecture, built-in tools, and extensive functionality for both simple and complex web applications." [11]

3.5 MVC (Model-view-controller):

"MVC is a design pattern used to decouple user-interface (view), data (model), and application logic (controller). This pattern helps to achieve separation of concerns.

Using the MVC pattern for websites, requests are routed to a Controller that is responsible for working with the Model to perform actions and/or retrieve data. The Controller chooses the View to display and provides it with the Model. The View renders the final page, based on the data in the Model." [12]

3.6 Laravel Breeze:

"Laravel Breeze is a minimal, simple implementation of all of Laravel's authentication features, including login, registration, password reset, email verification, and password confirmation. In addition, Breeze includes a simple "profile" page where the user may update their name, email address, and password." [13]

3.7 Tenancy for Laravel:

"A flexible multi-tenancy package for Laravel. Single & multi-database tenancy, automatic & manual mode, event-based architecture. Integrates perfectly with other packages." [14]

3.8 HTML:

"HTML (Hypertext Markup Language) is a text-based approach to describing how content contained within an HTML file is structured. This markup tells a web browser how to display text, images and other forms of multimedia on a webpage." [15]

3.9 CSS:

"Cascading Style Sheets, most of the time abbreviated as CSS, is a stylesheet language used to describe the presentation of a document written in HTML or XML (including various XML languages like SVG or XHTML). CSS describes how the structured element must be rendered on screen, on paper, in speech, or on other media" [16]

3.10 Bootstrap:

"It is a front-end framework used for easier and faster web development. It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many others. It can also use JavaScript plug-ins. It facilitates you to create responsive designs." [17]

3.11 JavaScript:

"JavaScript is a programming language enabling developers to interact with the functionality provided by web browsers. More specifically, JavaScript is a scripting language, which means (a) traditionally, JavaScript source code is interpreted at runtime and not pre-compiled into byte code and (b) practically, its main purpose is to modify the behavior of another application typically written in a different programming language, in which it is interpreted and run in real time." [18]

3.12 jQuery:

"jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript." [19]

3.13 AJAX:

"Asynchronous JavaScript and XML (AJAX) is a combination of web application development technologies that make web applications more responsive to user interaction. Whenever your users interact with a web application, such as when they click buttons or checkmark boxes, the browser exchanges data with the remote server. Data exchange can cause pages to reload and interrupt the user experience. With AJAX, web applications can send and receive data in the background so that only small portions of the page refresh as required." [20]

3.14 JSON:

"JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language." [21]

3.15 Mailtrap:

"Mailtrap is a platform designed to help businesses improve how they test, send, and control emails. Use it to build better marketing campaigns, inspect and debug messages before a campaign begins, and ensure better deliverability through features like a dedicated IP and suppression list segmentation." [22]

3.16 StarUML:

"StarUML is an open-source project to develop fast, flexible, extensible, featureful, and freely-available UML/MDA platform running on Win32 platform. StarUML modeling application licensed under a modified version of GNU GPL. The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software.

StarUML is based on UML version 1.4 and provides UML version 2.0 notations and eleven different types of diagrams" [23]

3.17 Eloquent:

"Laravel includes Eloquent, an object-relational mapper (ORM) that makes it enjoyable to interact with your database. When using Eloquent, each database table has a corresponding "Model" that is used to interact with that table. In addition to retrieving records from the database table, Eloquent models allow you to insert, update, and delete records from the table as well." [24]

3.18 Blade:

"Blade is the simple, yet powerful templating engine that is included with Laravel. Unlike some PHP templating engines, Blade does not restrict you from using plain PHP code in your templates. In fact, all Blade templates are compiled into plain PHP code and cached until they are modified, meaning Blade adds essentially zero overhead to your application. Blade template files use the.blade.php file extension" [25]

3.19 PHP:

Hypertext Preprocessor (PHP) is an open-source server-side scripting language that many developers use for web development. It is also a general-purpose language that you can use to make lots of projects, including Graphical User Interfaces (GUIs).

Conclusion:

The proposed healthcare cabin management system offers a comprehensive solution built on a Software-as-a-Service (SaaS) model and multi-tenancy architecture. It provides a wide range of functionalities, from appointment scheduling to electronic medical records management, catering to the needs of healthcare providers and patients. Robust security measures, including access controls and data encryption, ensure the utmost privacy and confidentiality of sensitive healthcare data. Developed using modern tools and technologies, the system is designed for scalability and adaptability to evolving healthcare needs

General Conclusion:

Throughout this study, we explored the landscape of healthcare management systems, by presenting different types, including doctor cabin management systems and patient-centric systems. We then examined their promising adoption and current state in Algeria. The second chapter focused on meticulously analyzing requirements and designing a comprehensive solution using various diagrams and modeling techniques to capture essential entities and relationships, laying a solid foundation for development. In the third chapter, we implemented and realized the proposed healthcare management system, leveraging cutting-edge technologies such as, Software-as-a-Service (SaaS), and multi-tenancy architecture to develop a robust and scalable platform tailored to the needs of healthcare providers and patients.

The developed system streamlines operations, enhances efficiency, and ensures the utmost security and privacy of sensitive medical data. Through user-friendly interfaces and real-time notifications, healthcare professionals can seamlessly manage patient records, scheduling, and billing, while patients can access their information and communicate with their providers effectively.

By integrating advanced technologies like SQL Server and Laravel Eloquent, which is an object-relational mapper (ORM) that simplifies database interactions, we have achieved observable improvements in database management speed and performance, even with a complex data structure as well as complex queries.

Our efforts have also resulted in the successful development of a cloud-based, Software-as-a-Service (SaaS) platform for clinic management systems, featuring a multi-tenancy architecture that allows for customization to meet the unique needs of each client (cabin).

While we have achieved our initial goals, this study serves as a foundation for further advancements and innovations in the field of healthcare management systems. We anticipate that future research and development efforts will continue to push the boundaries, incorporating emerging technologies and design principles to further enhance the quality of healthcare services and patient experiences. Potential areas for expansion could include developing a mobile application, integrating laboratory management systems, or incorporating pharmacy management capabilities into the platform.

Bibliography

[1] . B. Jonathan, "HOW I DID IT," *Harvard Business Review,* pp. 39-42, December 2015.

[2] "healthdatamanagement," 17 mars 2023. [Online]. Available: https://www.healthdatamanagement.com/articles/sticking-to-its-knitting-athenahealth-aims-for-consistency.

[3] I. Zocdoc, "pr newswire," 13 february 2024. [Online]. Available: https://www.prnewswire.com/news-releases/zocdoc-introduces-guided-search-helping-patients-more-confidently-connect-with-the-right-providers-302060150.html.

[4] M. R. b. H. S. L. Vanessa Ling, "everydayhealth," 17 august 2023. [Online]. Available: https://www.everydayhealth.com/emotional-health/zocdoc-review/.

[5] "cabisante," Logiciel de gestion de clinique et cabinet médical, march 2023. [Online]. Available: https://cabisante.com/.

[6] "eTabib," 1 april 2024. [Online]. Available: https://etabib.dz/en/find-doctor-and-medical-advice-online/.

[7] J. a. B. I. a. Z. J. a. Z. L.-J. Fiaidhi, "Enforcing Multitenancy for Cloud Computing Environments," *IT professional,* vol. 14, p. 1, 2012.

[8] v. c. team, "Visual Studio Code," [Online]. Available: https://code.visualstudio.com/docs. [Accessed 2 may 2024].

[9] aiven, "aiven," [Online]. Available: https://aiven.io/. [Accessed 12 may 2024].

[10] mysql, "mysql," [Online]. Available: https://www.mysql.com/products/workbench/. [Accessed 12 may 2024].

[11] C. Ribeiro, "sitePoint," 31 August 2023. [Online]. Available: https://www.sitepoint.com/laravel-introduction/.

[12] A. M. Pattern, "Microsoft," [Online]. Available: https://dotnet.microsoft.com/en-us/apps/aspnet/mvc. [Accessed 3 may 2024].

[13] L. Breeze, "Laravel," [Online]. Available: https://laravel.com/docs/11.x/starter-kits. [Accessed 03 may 2024].

[14] "tenancyforlaravel," [Online]. Available: https://tenancyforlaravel.com/. [Accessed 5 may 2024].

[15] B. Lutkevich, "theserverside," [Online]. Available: https://www.theserverside.com/definition/HTML-Hypertext-Markup-Language?fbclid=IwAR3sqhDz9NJEOVaStv38pzNiuyFazJWlsAbwmfTkb9cMhqHge9EDDDb-3SU. [Accessed 3 may 2024].

[16] bhagyamtiwari23, "MDN," 23 September 2025. [Online]. Available: https://web.archive.org/web/20150929020648/https:/developer.mozilla.org/en-US/docs/Web/CSS. [Accessed 5 may 2024].

[17] S. S. G. PROF PRATIBHA ADKAR, "A Review Paper on Bootstrap Framework," *IRE Journals,* vol. 2, no. 10, p. 349, 2019.

[18] K. Theisen, "Programming languages in chemistry:a review of HTML5/JavaScript," *Journal of Cheminformatics,* no. 11, p. 1, 2019.

[19] "JQuery," [Online]. Available: https://jquery.com/. [Accessed 5 may 2024].

[20] "aws amazon," [Online]. Available: https://aws.amazon.com/what-is/ajax/. [Accessed 5 may 2024].

[21] "json," [Online]. Available: https://www.json.org/json-en.html. [Accessed 5 may 2024].

[22] T. Eggspert, "crazyegg," 20 june 2023. [Online]. Available: https://www.crazyegg.com/blog/mailtrap-review/. [Accessed 4 may 2024].

[23] S. O. B. Noor Azian Mohamad Ali, "ANALYSIS BETWEEN ARGO UML AND STAR UML," *INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINERING AND MANAGEMENT V,* vol. 1, no. 1, p. 21, 2015.

[24] Laravel, "Laravel," [Online]. Available: https://laravel.com/docs/11.x/eloquent. [Accessed 26 May 2024].

[25] Laravel, "laravel," [Online]. Available: https://laravel.com/docs/11.x/blade. [Accessed 28 may 2024].

ABSTRACT

Healthcare management systems revolutionize patient care by replacing paper charts with digital records, allowing doctors instant access to comprehensive patient histories. These platforms also empower patients, enabling them to schedule appointments online and review their medical data securely.

In the rapidly evolving technological landscape, new architectures and approaches have emerged to address the limitations of traditional systems. Multi-tenancy architectures have become a necessity, providing added architectural value for companies with virtualized systems (Cloud, Fog, Edge, etc.) and a valuable skill set for engineers and developers.

To address the ambiguity surrounding this architectural trend, our contribution to this work (focused on application rather than research) is based on the use of the highest degree of multi-tenancy architecture, "unique database," when designing and building a Cloud Software-as-a-Service (SaaS) application. This approach aims to demystify the conceptual and programming complexities associated with this architectural trend, leveraging a SaaS application tailored for healthcare establishments that supports the demands of its tenants.

To realize this distinguished multi-tenant SaaS application, the Laravel Framework has been targeted in this thesis. By adopting a multi-tenancy architecture with a unique database, the proposed healthcare cabin management platform offers a scalable and efficient solution for healthcare providers.

Key words: SaaS ; Multi-Tenant ; Cloud ; Healthcare

ملخص

تحول نظام إدارة الرعاية الصحية إلى ثورة في رعاية المرضى باستبدال الرسوم البيانية الورقية بسجلات رقمية، مما يتيح للأطباء إمكانية الوصول الفوري إلى تاريخ المرضى الشامل. وتمكن هذه البرامج أيضا المرضى، وتمكنهم من تحديد مواعيدهم على الإنترنت واستعراض بياناتهم الطبية بأمان

في ظل التطور التكنولوجي المتسارع، ظهرت هياكل بنائية جديدة ونهج جديدة لمعالجة نقاط الضعف في الأنظمة التقليدية وقد أصبحت هياكل البنية التحتية متعددة المستأجرين ضرورة، حيث تضيف قيمة هيكلية للشركات ذات الأنظمة الافتراضية ومجموعة مهارات قيمة للمهندسين والمطورين

لمعالجة الغموض المحيط بهذا الاتجاه المعماري، تتمحور مساهمتنا في هذا العمل حول استخدام أعلى درجة من هيكلية متعددة المستأجرين "قاعدة بيانات فريدة" عند تصميم وبناء تطبيق للبرمجيات كخدمة سحابية. يهدف هذا النهج إلى تبسيط

التعقيدات المفاهيمية والبرمجية المرتبطة بهذا الاتجاه المعماري، من خلال تطبيق برمجيات كخدمة سحابية مصمم لمرافق الرعاية الصحية يلبي متطلبات مستأجريه.

لتحقيق هذا التطبيق متعدد المستأجرين والمميز للبرمجيات كخدمة سحابية، تم استهداف إطار عمل لارافيل في هذه الأطروحة. من خلال اعتماد هيكلية متعددة المستأجرين ذات قاعدة بيانات فريدة، يوفر منصة إدارة العيادات الصحية المقترحة حلاً قابلاً للتوسع وفعالاً لمقدمي الرعاية الصحية. توفر المنصة تبسيطًا للعمليات، وتعزيزًا للكفاءة، وضمانًا لخصوصية وأمن البيانات الطبية الحساسة.

الكلمات المفتاحية: ساس , متعدد المستأجرين , الرعاية الصحية , السحابة

Résumé

Les systèmes de gestion des soins de santé révolutionnent les soins aux patients en remplaçant les dossiers papier par des dossiers numériques, permettant aux médecins d'accéder instantanément aux antécédents complets des patients. Ces plateformes permettent également aux patients de prendre rendez-vous en ligne et de consulter leurs données médicales en toute sécurité.

Dans un paysage technologique en évolution rapide, de nouvelles architectures et approches ont émergé pour remédier aux limites des systèmes traditionnels. Les architectures multi-tenant sont devenues une nécessité, apportant une valeur architecturale ajoutée aux entreprises disposant de systèmes virtualisés (Cloud, Fog, Edge, etc.) et un ensemble de compétences précieuses pour les ingénieurs et les développeurs.

Pour lever l'ambiguïté entourant cette tendance architecturale, notre contribution à ce travail (axé sur l'application plutôt que sur la recherche) repose sur l'utilisation du plus haut degré d'architecture multi-tenant, « base de données unique », lors de la conception et de la construction d'un logiciel Cloud. Application en tant que service (SaaS). Cette approche vise à démystifier les complexités conceptuelles et de programmation associées à cette tendance architecturale, en s'appuyant sur une application SaaS adaptée aux établissements de santé qui répond aux demandes de ses locataires.

Pour réaliser cette application SaaS multi-tenant distinguée, le Framework Laravel a été ciblé dans cette thèse. En adoptant une architecture multi-tenant avec une base de données unique, la plateforme de gestion de cabines de soins de santé proposée offre une solution évolutive et efficace pour les prestataires de soins de santé.

Mots clés : SaaS ; Multi-locataire ; Soins de santé ; cloud

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY MOHAMED BOUDIAF- M'SILA

1985

جامعة محمد بوضياف - المسيلة

Université Mohamed Boudiaf - M'sila

TRADE NAME:

TABIBI

BRAND IMAGE :

Subject:

A Web-based Platform for Healthcare with Different Services

A project to obtain a certificate for a start-up institution within the framework of Ministerial Resolution 1275

Information Card:

About the supervision team and the work team

Supervision team:

Main supervisor 1: Pr. Hichem Debbi Specialization: computer science

work team:

Student: Ben Dahmane Dhiya El Haq Specialization: SIGL College: MI

Student: Kouriche Oussama Specialization: SIGL College: MI

1. **Project idea (proposed solution):**
    - ✓ Our field of activity is modern applications
    - ✓ The project idea began by noticing that all patients need to facilitate their services towards doctors, from booking appointments without hassle to the doctor's clinic and solving the problem of losing medical documents after the examination. After this observation, we moved to the side of the doctors, who in turn do not have a platform or network linking them to their patients. Sometimes the patient travels to a cabin hundreds of kilometers away, and then either finds the doctor out of service or cannot find an appointment. This is because doctors do not resort to a program to solve these problems due to the high cost of this service and its ineffectiveness.
    - ✓ We will produce a service platform
    - ✓ This is done by making Software as a Service and use the latest cloud technology
    - ✓ As a prototype, we are the ones who will accomplish it and it will be available on the web and as a phone application
2. **Suggested values:**
    - ✓ Provide the service for a very nominal amount instead of what is available in the market, relying on advanced technology
    - ✓ Make information available from anywhere using the latest technology Storage
    - ✓ Facilitating the exchange of patient information between doctors
    - ✓ Book an appointment with the doctor online
    - ✓ Solving the problems of losing medical files on the part of the doctor or the patient

3. **work team:**

    The project team consists of the following:
    - ✓ Student 1: Ben Dahmane Dhiya El Haq Specialization: SIGL
    - ✓ Student 1: Kouriche oussama Specialization: SIGL
    - ✓ The role of student 1 is to run the project and research markets and marketing and the latest technologies that will be relied upon, in addition to the applied needs of the project.
    - ✓ The role of student 2 is applying the ideas and information provided by Student 1 programmatically
    - ✓ The team is always in contact using Discord

4. **Project goals:**

   We seek to become the only company that provides Healthcare Management System by using the latest technologies that no one is using it perfectly, and in the long term we want take the agreement to take all the healthcare sector in Algeria in the digitalization area and this is what the country are looking for

5. **A timetable for achieving the project:**

			Months						
			1	2	3	4	5	6	7
1		**Searching patent databases and collecting the necessary information for the project**	✖	✖	✖				
2		**Initiate the programming process and complete a prototype**			✖	✖			
3		**Testing the prototype**				✖	✖	✖	
4		**Upload the project online to open the public experience**					✖	✖	
5		**Registering a patent in order to obtain a filing number and industrial protection**							✖

6. **View the market sector:**
   - ✓ Potential market: Everyone who can and wants to facilitate his work by booking medical appointments or doctor running his cabin
   - ✓ Target market: We seek to provide this platform to clinic owners and everyone in the health sector, that is, to digitize the sector.
   - ✓ This category was chosen because of the need for a simple, effective and inexpensive platform for this sector.
   - ✓ There is a project to conclude contracts with doctors who use the platform, and the goal is to make doctors as partners and not as customers.

7. **Measuring the intensity of competition:**

Our competitors in the Algerian market are few or even no one uses advanced technologies to meet the needs of customers in Algeria. And they are in order:

- ✓ CabiSante – etabib -……
- ✓ Among their major strengths is immorality and good use of old technology, with no competitor using modern technology
- ✓ Among their weaknesses is their use of classic technology and their failure to keep up with the modern generation / the ineffectiveness of the products, which led to the loss of many customers and their unwillingness to rely on healthcare management system software.

8. **Costs and burdens:**

- ✓ **Fixed costs:** Equipment costs (computers), platform design, platform hosting costs, domain costs, workers' wages, workplace with transportation vehicles, and Internet costs.
- ✓ **Variable costs:** Costs of renting storage space in the cloud. Advertising Coste

9. **Business number:**

	N	N+1	N+2	N+3	N+4
Subscriptions to the platform	100	140	190	250	300
Subscription amount (DA)	39000	39000	39000	39000	39000
Annual total (Da)	3900000	5460000	7410000	9750000	11700000

10. **Experimental prototype:**

Prototype of our platform**:** **https://youtu.be/uP_LnJ6udYY**

مخطط نموذج العمل التجاريّ

الشراكات الرئيسيَّة
Key Partners

-cloud owner
-host service owner
-Doctor
-financial institutions
-Governmental entities
-Algeria post
-Algeria Telecom

الأنشطة الرئيسيَّة
Key Activities

-Preparing the platform
-preparing the cloude
-sales and marketing
-platform licensing

الموارد الرئيسيَّة
Key Resources

-Materialism:cloude/PCs/
-Human: programmers/marketers
-Finance: advertisement

القيم المقترحة
Value Proposition

-Providing a platform rental service that meets the cabin's needs instead of purchasing a one-time program.
-checking any information from any place just by useing internet.
-Facilitating the exchange of patient information between doctors.
-Possibility of booking appointments online.

العلاقات مع العملاء
Customer Relationships

-Direct communication with a doctor.
-For patients, it is through the doctor.
-Offering benefits to continuing customers.

القنوات
Channels

-direct communication
-Advertisements
-social media sites
-Advertisements within our platform

شرائح العملاء
Customer Segments

- Doctors
- patient
-medicament companies that want do an advertisement for the patients

هيكل التكاليف
Cost Structure

-Equipment costs (computers)
-the design
-cloud renting
-hosting costs
-Staff costs
-Advertising costs, internet, workplace, car

مصادر الإيرادات
Revenue Streams

-Via subscription fees
-Through advertisements on the platform