



CLINICAL APPOINTMENT MANAGEMENT AND DOCTOR CORRESPONDENCE APPLICATION

Date: 1st June 2022

Supervisor:

Professor Ahsan Shah

Group Members:

Syed Muhammad Ashhar Shah	2020478
Khizar Ali Shah	2020196
Muhammad Omer	2020355
Harris Abdullah	2020159



List Of Contents

1. Introduction	Page 4
1.1 Purpose	Page 4
1.2 Product Scope	Page 4
2. Overview	
2.1 The Overall Description	Page 5
2.2 Product Perspective	Page 5
2.3 Product Functions	Page 5
2.4 User Characteristics	Page 6
2.5 Constraints	Page 6
2.6 Assumptions and Dependencies	Page 7
3. State of the Art	Page 7
4. User/System Requirements	Page 8
4.1 External Interface Requirements	Page 8
4.1.1 User Interfaces	Page 8
4.1.2 Hardware Interfaces	Page 9
4.1.3 Software Interfaces	Page 9
4.1.4 Communication Interfaces	Page 10
5. Functional Requirements	Page 10
5.1 Functional Requirements with Traceability Information	Page 10
6. Non-Functional Requirements & Software System Attributes	Page 13
6.1 Performance Requirements	Page 13
7. Project Design/Architecture	Page 14
7.1 Use-Case View	Page 14
7.2 Logical View	Page 15
7.3 Process View	Page 15
7.4 Development View	Page 17
7.3 Physical View	Page 17

**List of Figures:**

1. State-Of-The-Art Application	Page 7
2. Use-Case Diagram	Page 14
3. UML Class Diagram	Page 15
4. Doctor Sequence Diagram	Page 16
5. Patient Sequence Diagram	Page 16
6. Component Diagram	Page 17
7. Deployment Diagram	Page 18

Appendix: Page 19

1. User Interfaces	Page 19
2. Entity-Relationship (ER) Model	Page 24



1. Introduction

The project described in this document is a web-based medical care system which deals in connecting patients with doctors around the globe so that they can carry out medical treatment from the comfort of their home. Not only this but the patient can also check for a disease they may have contracted from just a click of a button due to the symptom checker ability we provide for them.

1.1 Purpose

The world is changing, due to the pandemic many people have figured out the convenience of online transactions and jobs. Many people now have an understanding that they do not have to leave the comfort of their homes to buy an item and even for their jobs. So, bringing one more thing under the umbrella will only be more convenient for people. We propose an application which can be used to get online doctor appointments and a hub for patients to correspond with medical experts from the comfort of their homes. Not only will this be useful for users, but it will also be very useful for doctors to treat patients with not-so serious ailments and make an extra buck on the side. With this, if the world does have to shut down again for whatever reason, doctors will not be at a loss of patients as people cannot leave their homes. Also, as it was seen in the recent pandemic, many people with minor symptoms or ailments other than Covid 19 were not even allowed into the hospitals. This application will also help treat people with minor symptoms and ailments which do not warrant a costly and time-taking trip to the hospital. Another plus point of this application is for people who are afraid to go to the hospital as many people are in countries like Pakistan. They can correspond with experts through our app and get a better perspective on their condition.

1.2 Product Scope

The users for this application are patients who need to seek medical advice from doctors through a remote platform. This project is targeted the users who can not leave their home due to any issues or circumstances and need to seek medical help from a professional or they live in a place where they don't have access to such help. This project deals with the medical industry and can be beneficial as it can help provide medical aid across the globe by connecting doctors and patients together and help them communicate with each other to resolve the issue the patients might be facing. Our product provides a live chat service as a platform for communication between doctors and patients and gives the patient full control over the appointment timings that suit them so they don't have to go out of their way to make time for the appointment, they can just select a slot they are most comfortable



with. We focus to deliver an easy to use medical assistance platform for users around the world.

2. Overview

2.1 The Overall Description

The service we offer is an appointment booking and symptom checking application. In this we have two types of users, doctors, and patients. Both users must create an account on the system using the Sign-Up functionality. After signing up they can use the Login functionality to login to their account to access the respective dashboards. Furthermore, a doctors can interact with a patient through the chat functionality we have developed which is further generalized into chatting either through the messaging app or through a live calling session. The users will be able to book an appointment with the doctor online for a live chatting session in which they can explain the symptoms for a full diagnosis of the medical issue they may have developed. A user can also check for the issue they may be facing by the symptom checking application we have integrated into our system as well.

2.2 Product Perspective

There are two types of stake holders involved in our project, patients, and doctors. Both stakeholders have various requirements that need to be met by the product. The main requirement of the developed application is the ease of use for everyone to seek expert medical consultation. The application gives our patients the flexibility of getting medical assistance from the comfort of their home hence the system completes the requirements laid down by the end users and leads to a high level of customer satisfaction.

2.3 Product Functions

Our product provides multiple functions for our end users. We provide a login and signup function in which a user first creates an account on our platform using his desired credentials. There is also the function for checking the any disease that the user might have developed by exchanging information about the symptoms with the system which would process and find the disease. We also provide a live chat bot which is accessible by both the doctors and patients as a ground for communication. Furthermore, there is also an appointment booking feature in which a patient can schedule and appointment after which the desired medical professional can respond.



2.4 User Characteristics

Our user would be divided into two types, one would be a doctor and the other would be a patient. The patient would be the one who would use the website and the doctor would be a respondent who would assist the patient through the application. Our patient user would be anyone who doesn't have access to a local hospital due to any reason whatsoever and needs medical assistance. The patient would most likely be from a backwards area where there is not a lot of medical personnel. He would possess enough basic computer skills so that he knows how to carry out the functionalities we designed. The patients who will be using the website will be over the age of 18 if we have a patient younger then he shall be allowed to book an appointment through a parent or a legal guardian. The user would also have a computer with basic hardware like a microphone and web camera for the appointment meeting to take place. They would also need a stable internet access to perform operations on our application.

Our other user would be a doctor who would need to possess sufficient medical knowledge on his area of expertise. He would be required to have a verified doctor's license to carry out any medical practice. He would also be required to have any experience prior to working with us as we would need trained medical staff to act as the respondents. He would also need to possess sufficient computer skills to use the application, the doctor would also need to fulfill the hardware requirements such as he would need to have a microphone and a web camera for the appointment as well as a stable internet connection.

2.5 Constraints

The sign-up and sign-in systems are limited because they do not gather a lot of data from the user. If they can gather much data, then the application can be made more productive. Moreover, users do not have an easy way of sign-up other than filling the form, like providing fingerprint or face or a Gmail account.

Another constraint is that there is no notification system. Either party such as patient or doctor, may miss the appointment time and date and realizing it later. The disease checker is constrained in such a way that it contains very few symptom and disease patterns in it, which may not be able to fulfil the need to every patient.

The chat is not fully developed and there is a lag in it due to slower queries and database access. Database is not live yet, which is a big hurdle because two users cannot login on different computers, each having local database tables, and use the app in the manner it is supposed to be used. Each will be updating their own tables and other will not see changes in their local php-mysql-database.



Disease checker does not require doctor's permission. This is a constraint because it might not give correct information all the time, which may mislead the patient. Patient can access it without doctor's prescription, which is a problem.

2.6 Assumptions and Dependencies

The following assumptions are made when developing the system:

- The user of the application already has a fully functional front camera installed for the purpose of video calling between doctor and patient.
- The user has a stable and fast internet connection for clear and undistorted video and voice calling.
- The user has a fully functional microphone either separately or one integrated into his device for the purpose of voice calling between the doctor and the patient.

The following dependencies are present in the application developed:

- Being a data driven model the application is dependent on data for it to perform the functionalities. As seen from the symptom checking functionality it depends on data, the more the data the more accurate the diagnosis. Hence data is a big dependency in our system.

3. State of the Art

There is already a state-of-the-art application like the one mentioned in this report which goes by the name oladoc. Figure 1 attached below shows the main page for the application.

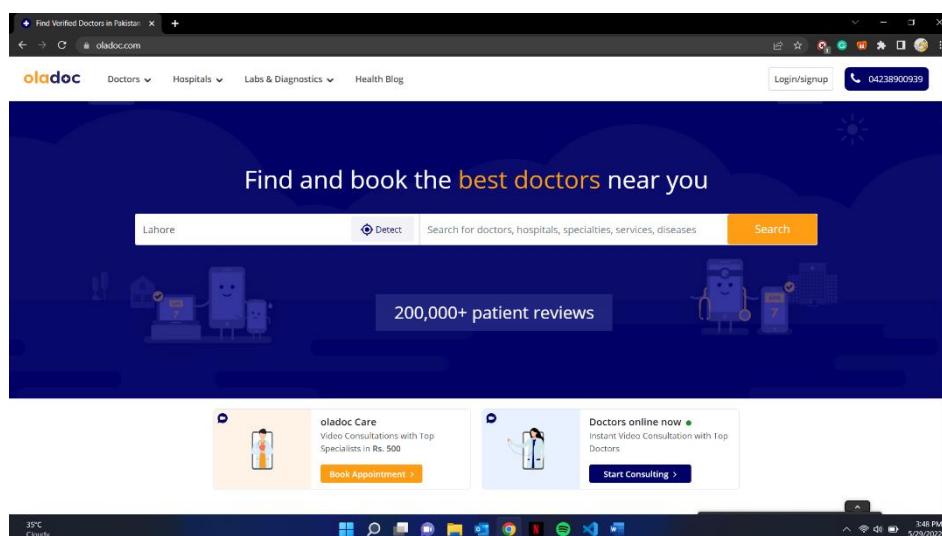


Fig.1 OlaDoc Application Homepage



This application has various doctors and surgeons which you can filter out based on their profession in the medical industry. You can also check for available hospitals in major cities of Pakistan to find out which one you would want to visit for any medical assistance you might need. Furthermore, there is also the feature to locate labs near you where you might go to perform any tests you may want to get done. This application also provides an appointment booking system for the user in which they can filter out from a list of qualified doctors on the basis of experience, fee, rating, and area of expertise. There is also a whole section of the application dedicated to a health blog that has various articles on various trending topics in the health industry of Pakistan.

While this is a good application and a possible competitor of ours. However, this application does not provide the flexibility to check for problems using the developing symptoms before consulting a specialist related to the issue diagnosed by our system which saves the time for the patient along with any fee they may have to pay for the diagnosis of the problem at hand.

4. User/System Requirements

4.1 External Interface Requirements

4.1.1 User Interfaces

We developed the interface keeping in mind the medical theme of our application hence a clean and minimalistic interface. The users of the system have multiple interfaces below is a list of the user interfaces along with the functionalities listed in each of the given interface.

Interface	User(s)	Functionalities and Requirements
Doctor Login	Doctor	The user would enter the credentials he entered during the initial signup. If he entered the wrong credentials he would be prompted with an error.
Doctor Sign-Up	Doctor	The user would create an account using a unique id and password which would then be saved into the database.
Patient Login	Patient	The user would enter the credentials he entered during the initial signup. If he entered the wrong credentials he would be prompted with an error.
Patient Sign-Up	Patient	The user would create an account using a unique id and password which would then be saved into the database.
Patient Dashboard	Patient	Here we have a list of doctors which are available in our system along with some of the features defined for the patient.



Doctor Dashboard	Doctor	Here we have a list of patients that have booked an appointment with the doctor along with some features defined for the doctor.
Doctor Chat Box	Doctor	In this interface we provide a chat box where a doctor can send messages to his patient along with an notification option for unread messages.
Patient Chat Box	Patient	In this interface we provide a chat box where a patient can send messages to his respective doctor along with a notification option for any unread messages.
Appointment Form	Patient	The following interface provides an appointment booking form for the patient to fill, with fields to be filled like Date, Time, Note and Phone No.
Symptom Form	Patient	The patient enters the symptom for which related diseases are displayed.

At the end of the report, you can find images of the respective interfaces mentioned so you can get a better understanding of the User Interfaces of the designed application.

4.1.2 Hardware Interfaces

All server-side components would be executed on the server-class computers as all the queries would be processed at the server side. Along with that all client related components would run at the client side on his own machine like a personal computer class some of these components include the chatting feature which would be processed at the client side. On the hardware side the users would be using any camera and microphone attached to the system for the chatting functionality.

4.1.3 Software Interfaces

The application was developed using languages like HTML5, CSS and PHP hence the on the software side the user would need any web browser that has support for the given technologies. Our database would be maintained through MySQL and MariaDB using the administrative tool phpMyAdmin. The application would use these interfaces to communicate with the web browser on how to display the data and from where it would retrieve the desired data.

4.1.4 Communication Interfaces

The application designed would need connection to the internet through wifi or an ethernet cable as when the server is hosted, we will need it to access the database server. We also need an internet connection to run our functionality of chatting with the user.



5. Functional Requirements

Functional requirements are the features of the developed applications or functions that must be implemented by the application developers to enable the users of the application to accomplish their tasks. Our application has various functional requirements for the both the users of our application that are listed below.

5.1 Functional Requirements with Traceability Information

6. Requirement ID	1		Requirement Type		Functional		Use Case #		1
Status	New	Yes	Agreed-to	-	Baselined	-	Rejected	-	
Parent Requirement #	None								
Description	The patient and doctor both have to sign-up before entering the application. Each patient and doctor is identified by a unique id and a non-unique username. They can provide extra information such as profile picture, symptoms (for patients only). If during sign-up, the provided data matches any data before, then that user cannot sign-up, because the users must be unique. This requirement checks for validity if the account is new or not. Moreover, it gives an error in case password is not meeting the conditions.								
Rationale	The rationale is that without sign-up, users will not be able to communicate with each other properly. Patients and Doctors will be mixed and other requirements such as login and appointment will not be fulfilled. Therefore, sign-up is mandatory.								
Source					Source Document		-		
Acceptance/Fit Criteria	User provides a username and a password. Password must be of at least 6 digits. Usernames may not be unique.								
Dependencies	None								
Priority	Essential	Yes	Conditional	-	Optional	-			
Change History	None								

Requirement ID	2	Requirement Type				Functional	Use Case #		1
Status	New	Yes	Agreed-to	-	Baselined	-	Rejected	-	
Parent Requirement #	1								
Description	When the parent requirement of sign-up is complete, the user's provided data, which uniquely identifies that user, goes to the database. Then the user can sign-in anytime. The sign-in process allows the user, patient, or doctor, to log into the application and start using it.								



Rationale	Even if sign-up is done, users can only enter their data which stores in the database. Sign-up alone does not allow them to enter the application. Hence login is the bridge between the system and the sign-up and only those users and enter the system who have already signed up.						
Source				Source Document	-		
Acceptance/Fit Criteria	User provides a username and a password. It must match their username and password as they have used it in the sign-up requirement. If either does not match, then user will not be able to login.						
Dependencies	1 (Sign-up). Only signed up users can sign-in/login						
Priority	Essential	Yes	Conditional	-	Optional	-	
Change History	None						

Requirement ID	3			Requirement Type	Functional			Use Case #	1
Status	New	Yes	Agreed-to	-	Baselined	-	Rejected	-	
Parent Requirement #	1, 2								
Description	Patient can make an appointment for a doctor. They select their desired doctor first and then fill the form for appointment. In the form, patient provides the date, time, and their phone number. They must write a message to the doctor explaining their condition so that the doctor knows about the patient's symptoms and information beforehand.								
Rationale	This is the core part of our application. There are two types of users, doctors, and patients. Our application makes them communicate with each other but not in a freeway, as in social media. But in a very restricted way. Patient cannot talk to a doctor without appointment. Same is the case with the doctor. Without an appointment system, patients and doctors will be able to communicate with each other freely as in social media, which must be avoided here. Doctors are busy and they must only respond to patients who requested an appointment. Similarly, patients time is also important; which they set in the form.								
Source	None				Source Document	-			
Acceptance/Fit Criteria	Patient fills a form for making the appointment. (S)He provides the data, time, message, and phone-number.								
Dependencies	It depends on who the user is, if it is a patient then this requirement is present. Doctors do not make appointments. They can respond to requirements.								
Priority	Essential		Conditional	Yes	Optional	-			
Change History	None								



Requirement ID	4			Requirement Type		Functional		Use Case #		1
Status	New	Yes	Agreed-to	-	Baselined	-	Rejected	-		
Parent Requirement #	1, 2									
Description	Patients can search for the disease they have by providing the symptoms. This requirement does not need doctors' permission; it may be implemented in one of the next versions that patient could not check this without proper permission. Currently however, this requirement is not dependent or related to the appointment.									
Rationale	Patients comes to the application to know what disease(s) they have. This way, they can easily know what type to doctor they should get appointment form. Otherwise, patient will not know whether to take appointment from a cardiac specialist or a general practitioner, given that they have some pain in the chest along with some other symptoms.									
Source	None				Source Document		-			
Acceptance/Fit Criteria	Patients provides his/her symptoms. A list of diseases gets shown that match these symptoms. They are reduced as much as possible.									
Dependencies	It depends on who the user is, if it is a patient then this requirement is present. Doctors do not need this. Patients need this for referring to the proper doctor for appointment.									
Priority	Essential	-	Conditional	-	Optional	Yes				
Change History	None									

Requirement ID	5		Requirement Type		Functional		Use Case #		1
Status	New	Yes	Agreed-to	-	Baselined	-	Rejected	-	
Parent Requirement #	1, 2, 3								
Description	Patients and doctors can chat with each other. This is mimicking the actual appointment when patient goes to the doctor. Patient can describe condition and doctor can prescribe medicine, etc.								
Rationale	After making an appointment, patient can chat with the doctor at the specified time. Without the functionality of chat, there is no point in making an appointment.								
Source	None				Source Document		-		



Acceptance/Fit Criteria	Does not require any criteria. Patient and doctor can just go to their respective chats and start chatting. They can only chat if appointment is made and it is the appointed time.					
Dependencies	Chat depends on the appointment. The chatting between patients and doctors is blocked unless there exists an appointment.					
Priority	<i>Essential</i>	-	<i>Conditional</i>	-	<i>Optional</i>	Yes
Change History	None					

6. Non-Functional Requirements and Software System Attributes

A non-functional requirement specifies the criteria that can be used to judge the operation of a system, rather than specific behavior of a system. It can be measured. Well-defined non-functional requirements are relatively easy to test. Hence, they can be used to evaluate a project's success. Some of the non-functional requirements are listed below for the developed application.

6.1 Performance Requirements

- *Stream latency*: It is the delay between your camera capturing an event and the event being displayed to viewers. It depends on the internet connectivity of the users of our application and differ from user to user, the better the internet connectivity the less the latency. Some users may experience zero to no latency while others may not.
- *Scalability*: Our system is limited to only work as a service to provide a basis of communication between a patient and a doctor along with the ease of symptom diagnosis however the system has a lot of potential for scalability by adding new features such as searching for hospitals in a vicinity or a feedback system for the doctor.
- *Accuracy*: Our symptom diagnosis depends on data for accuracy. The more data we insert the more accurate the diagnosis it can perform. Hence for a large data set our application is fairly accurate.
- *Security*: Our application offers a security protection layer through our login and signup feature where a combination of a password and a username is required to access a specific user's information.
- *Flexibility*: A flexible software architecture can adapt to changes in both environment and usability requirements without encompassing structural changes. Our application can be used in various environments like phones, laptops and desktops using just a web browser and an internet connection.



7. Project Design/Architecture

A mandatory 4+1 architecture view model is discussed which provides a clear understanding of the system, its components and how they communicate together for the application to perform the requirements set by the stakeholders for the application.

7.1 Use-Case View

The description of an architecture is illustrated using a small set of use cases, or scenarios, this is also known as the fifth view in the 4+1 architecture. Below in Figure 2 we have attached the use case for the clinical appointment management and doctor correspondence application we have built, this use case can be used to identify architectural elements and to illustrate and validate the architecture design. In our use case we have two types of end users, doctors, and patients. The patient acts as an actor and the doctor act as a respondent. Both share the common functionality of Sign-Up and Login. A patient can make an appointment with the doctor. Both also share the same functionality of chatting which a generalized form of the two different types listed in the diagram.

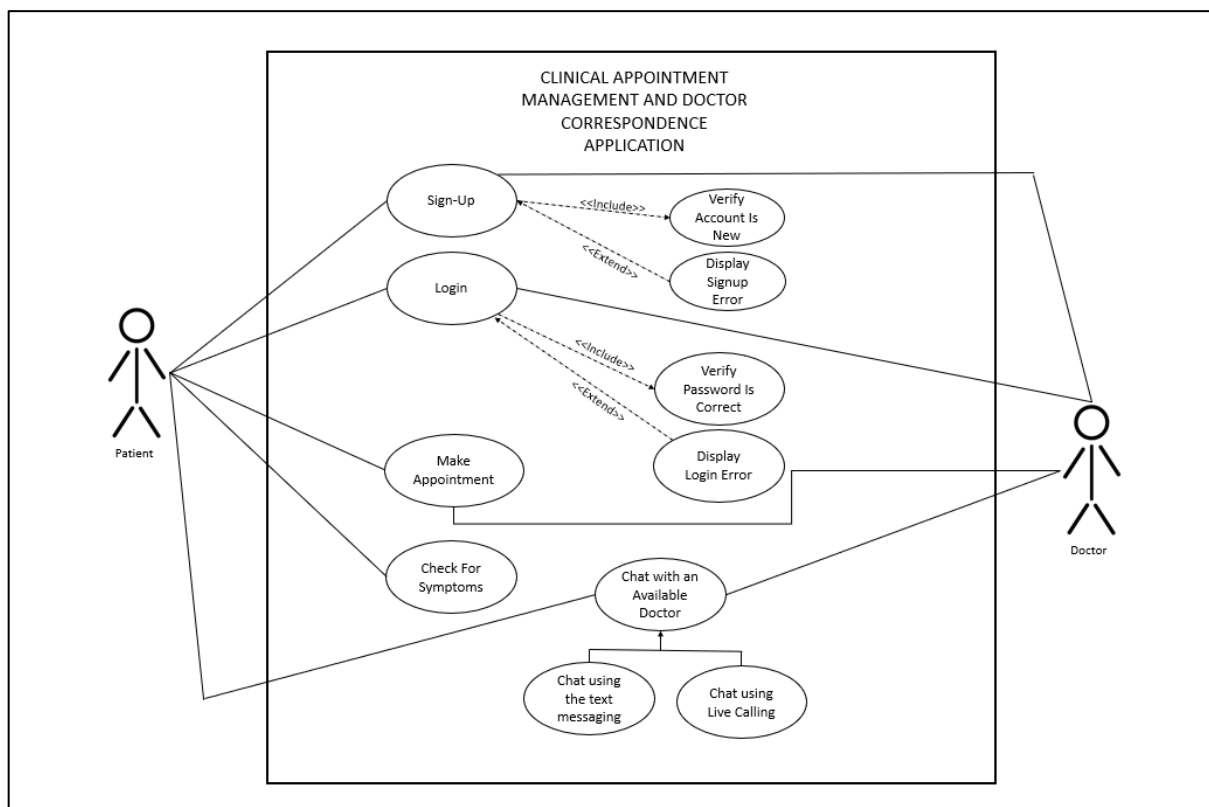


Fig.2 System Use-Case Diagram



7.2 Logical View

This view is concerned with the functionality that the system provides to end-users. We have used a UML Class Diagram shown below in Figure 3 to represent the Logical View of the system we have developed. In the diagram below our classes correspond to the tables we have in our database along with the data types of the various data our tables are populated with also we provide the functions in the class blocks that represent what the data is to be used for by the application.

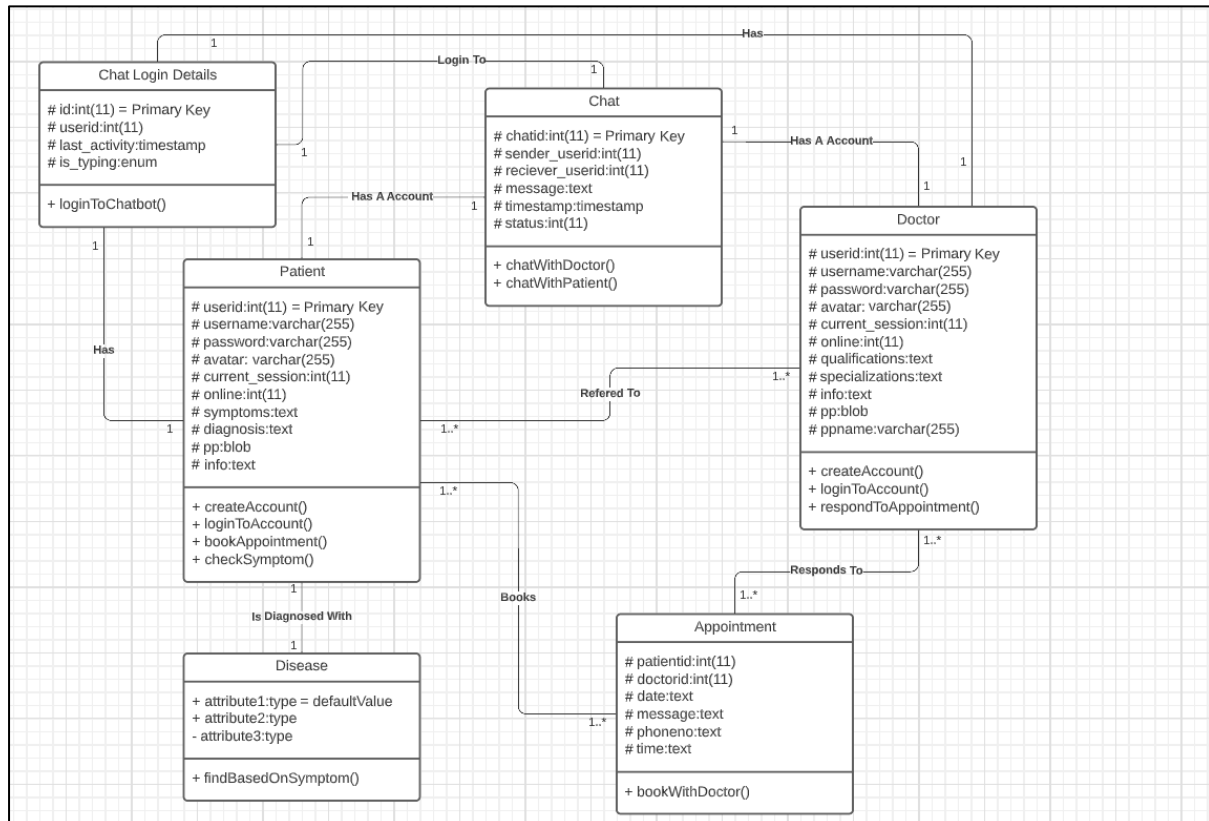


Fig.3 UML Class Diagram for the Application

7.3 Process view

The process view deals with the dynamic aspects of the system this view explains the system processes and how they communicate, with each other. The focus of this view is on the run time behavior of the system and how it follows a process to do a certain task that is performed by the user. We will use two types of sequence diagrams to describe this view as we have two types of application users each with a different set of requirements. In Figure 4 we have the sequence diagram concerning a doctor along with the sequence of processes executed. In Figure 5 we have the sequence diagram associated with a patient and the processes he is to execute.

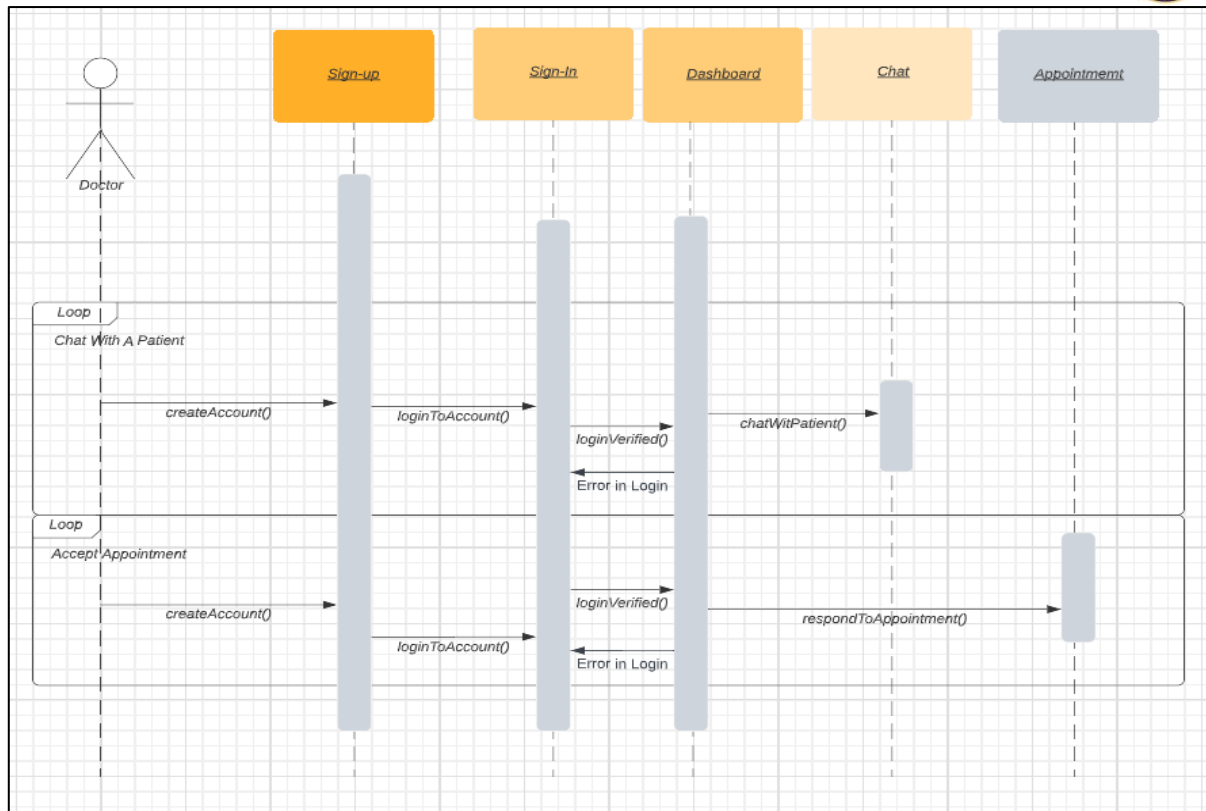


Fig.4 Doctor Sequence Diagram

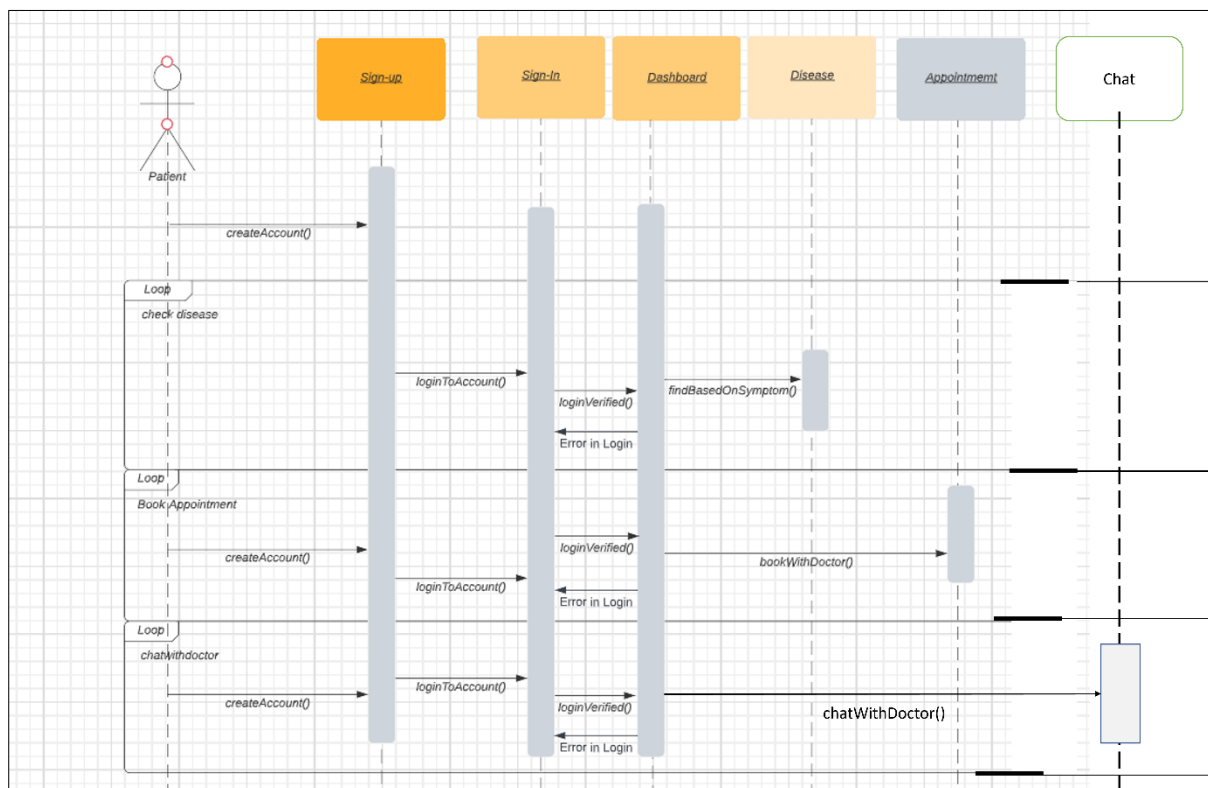


Fig.5 Patient Sequence Diagram



7.4 Development View

The development view illustrates a system from a programmer's perspective and is concerned with software management. This view is also known as the implementation view. To represent this view, we use a component diagram shown in Figure 6 that describes the organization of the physical components in a system.

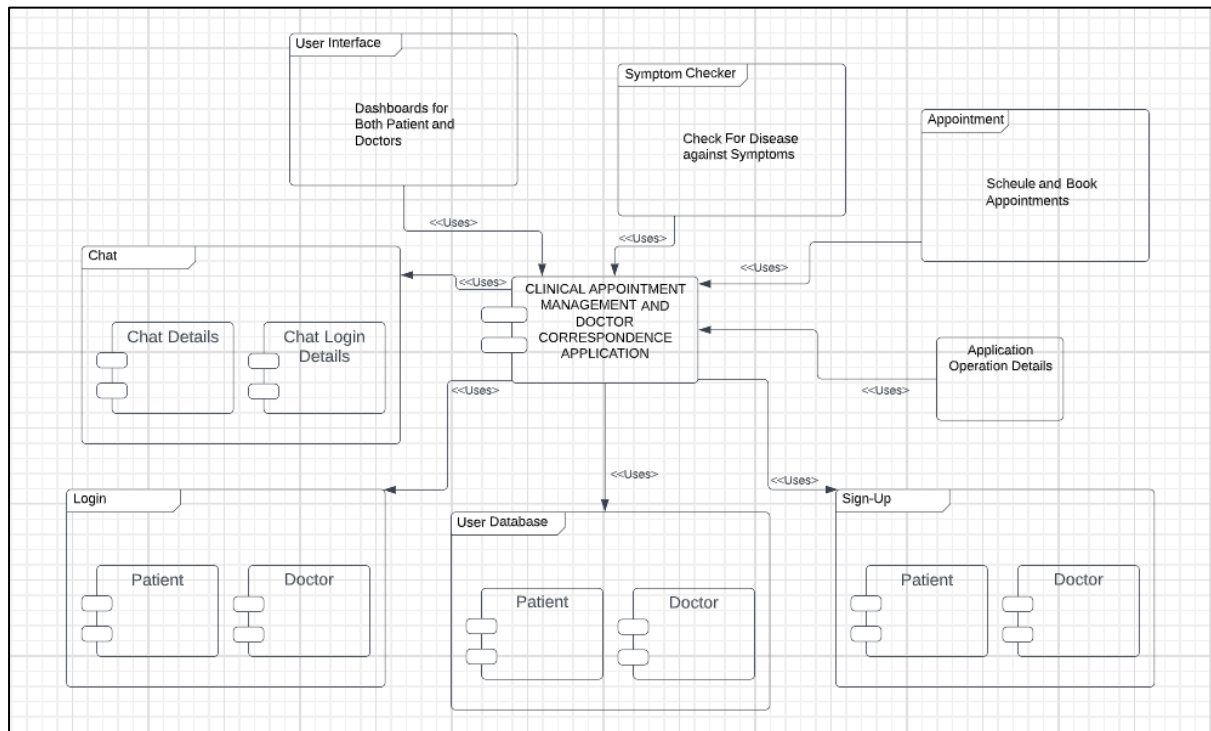


Fig.6 Component Diagram of the System

7.5 Physical View

The physical view depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer and the physical connections between these components. This view is also referred to as the deployment view and is represented through a deployment diagram. A deployment diagram is a type of a UML diagram that shows the execution architecture of a system. It includes nodes such as hardware or software execution environments, as well as the middleware connecting them. The figure below shows the deployment diagram for the developed application.

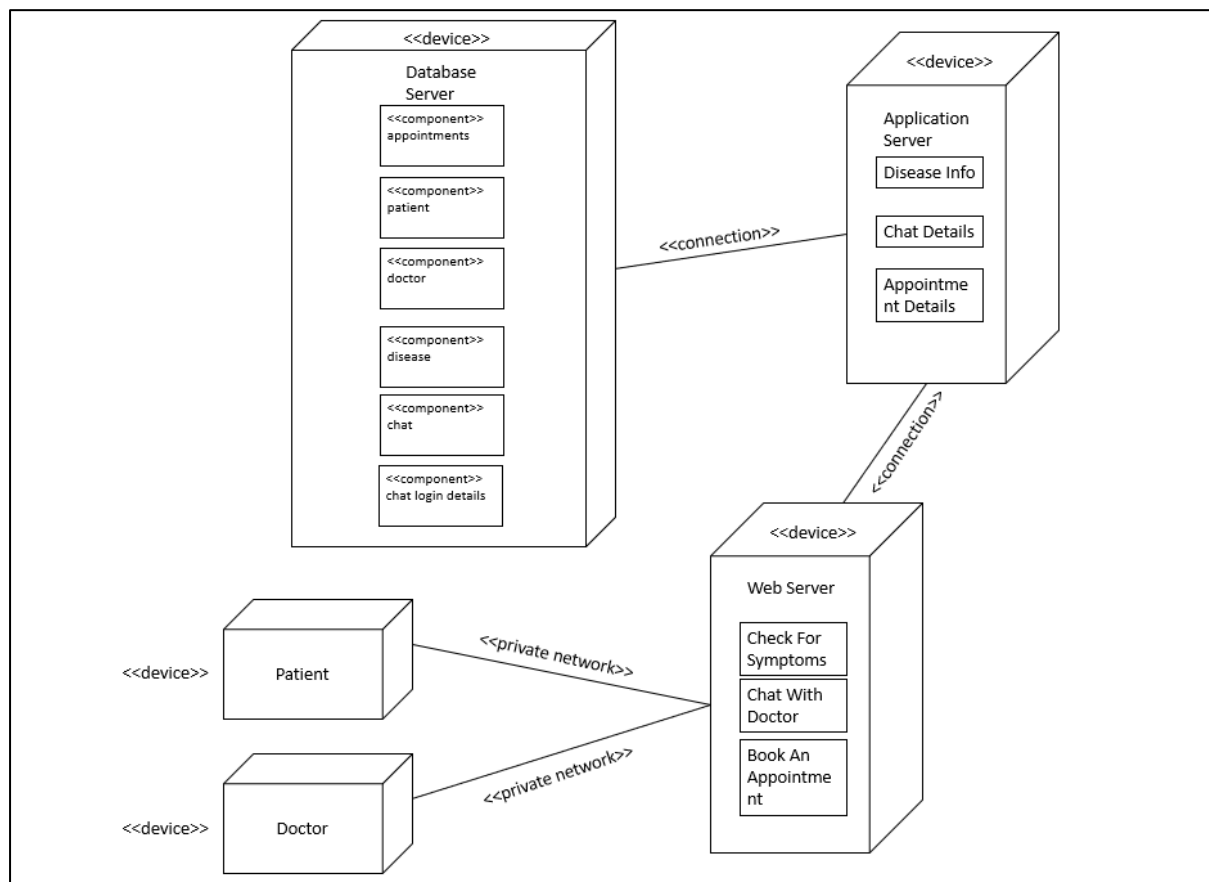


Fig.7 Deployment Diagram for the Application

APPENDIX

1. User Interfaces:

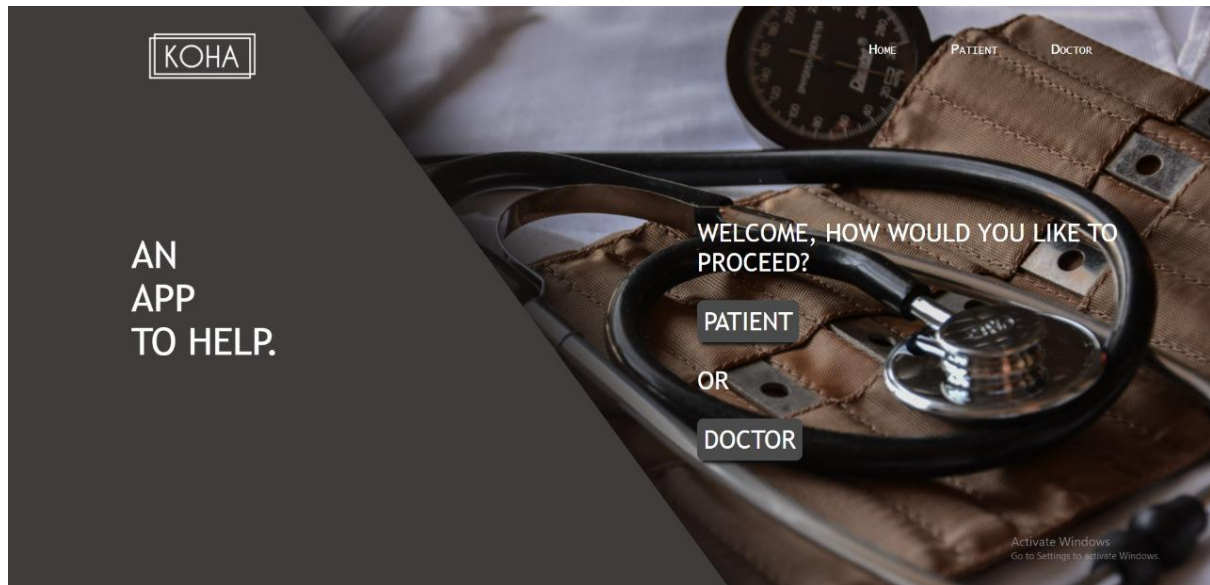


Fig.8.1 Homepage

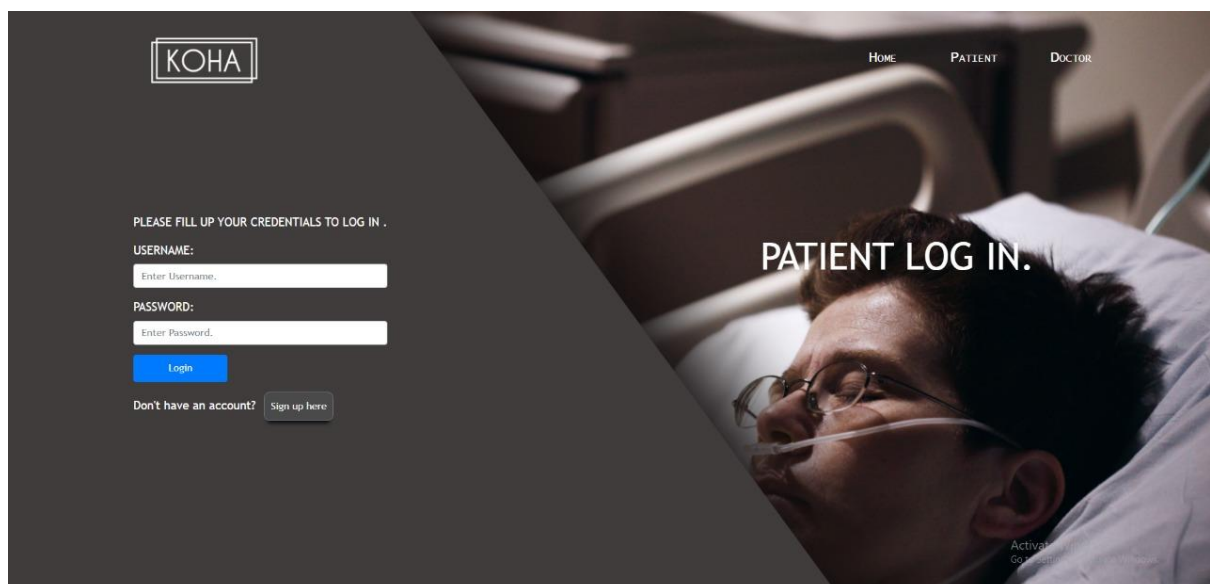
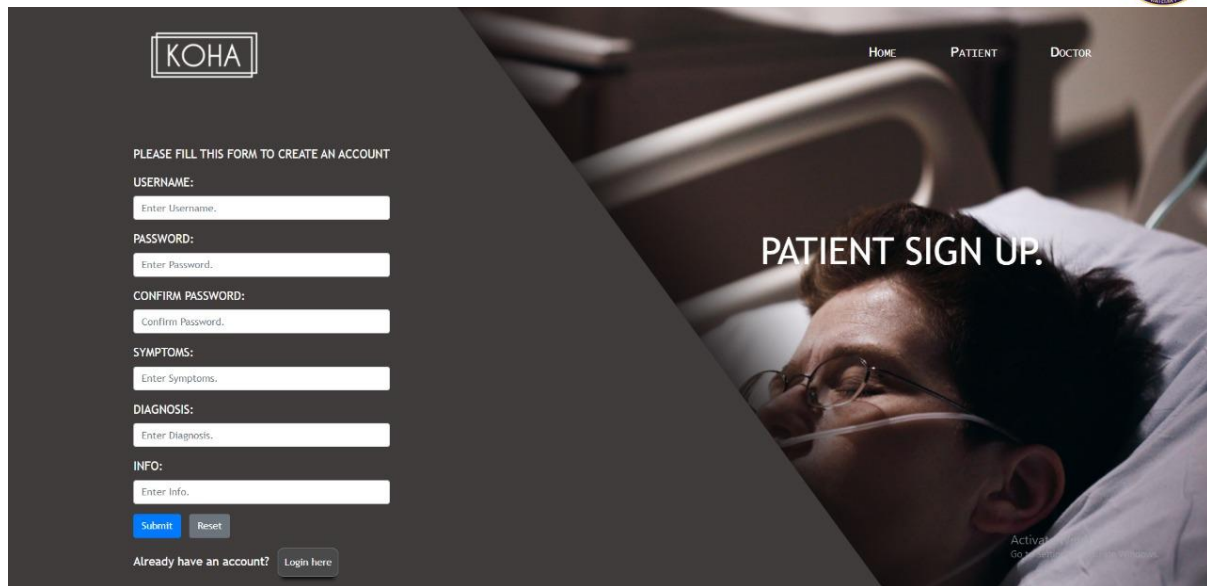


Fig.8.2 Patient Log-in Page

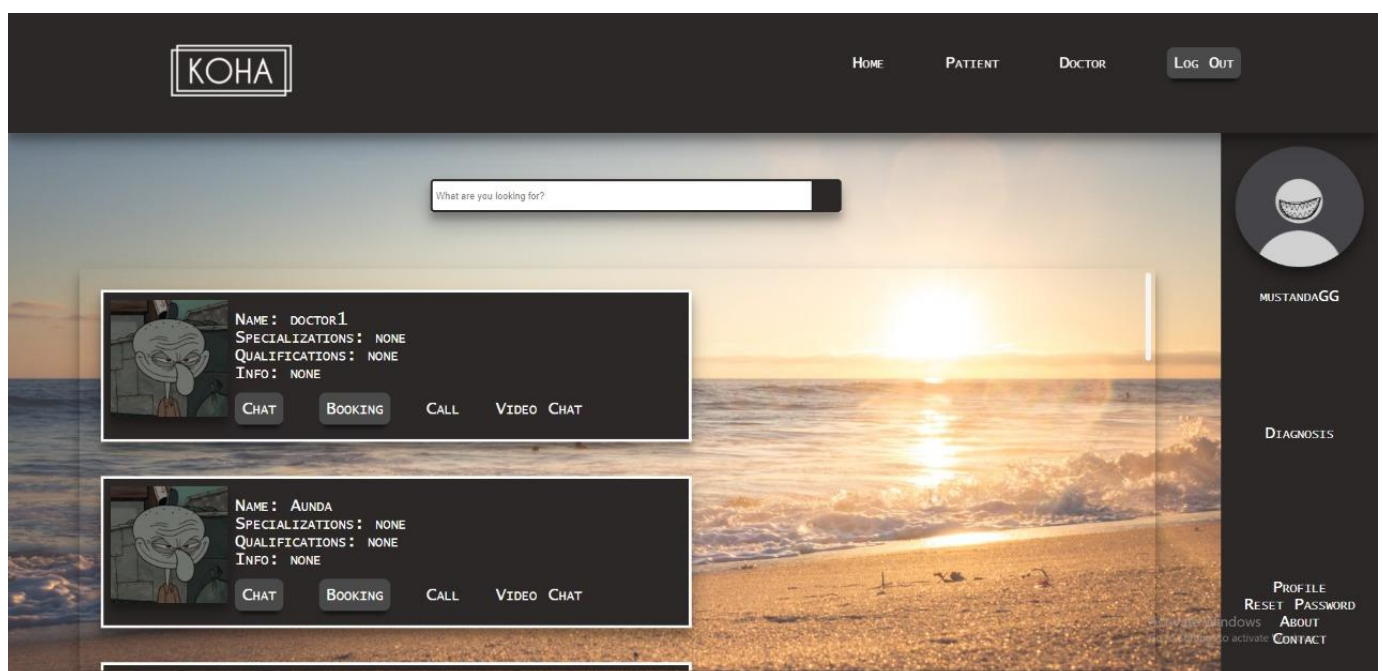


The Patient Sign-Up Page features a dark background with a large image of a patient in a hospital bed on the right. The left side contains a registration form with the following fields:

- USERNAME: Enter Username.
- PASSWORD: Enter Password.
- CONFIRM PASSWORD: Confirm Password.
- SYMPTOMS: Enter Symptoms.
- DIAGNOSIS: Enter Diagnosis.
- INFO: Enter Info.

Buttons for 'Submit' and 'Reset' are located below the INFO field. A link 'Already have an account? Login here' is at the bottom left. The text 'PATIENT SIGN UP.' is overlaid on the right image. A small 'Activate your account' message is visible in the bottom right corner of the image area.

Fig.8.3 Patient Sign-Up Page



The Patient Dashboard has a dark header with the KOHA logo and navigation links: HOME, PATIENT, DOCTOR, and a LOG OUT button. A search bar is centered below the header. The main content area features a large background image of a sunset over the ocean. On the left, there are two doctor profiles, each with a cartoon avatar and the following details:

- DOCTOR 1:** NAME: DOCTOR 1, SPECIALIZATIONS: NONE, QUALIFICATIONS: NONE, INFO: NONE. Buttons: CHAT, BOOKING, CALL, VIDEO CHAT.
- AUNDA:** NAME: AUNDA, SPECIALIZATIONS: NONE, QUALIFICATIONS: NONE, INFO: NONE. Buttons: CHAT, BOOKING, CALL, VIDEO CHAT.

On the right side, there is a user profile section with a circular avatar, the name MUSTANDA GG, and a list of links: DIAGNOSIS, PROFILE, RESET, PASSWORD, ABOUT, and CONTACT.

Fig.8.4 Patient Dashboard



Fill the Following Form to Book an Appointment.

Date:

Select a time:

Note for the Doctor:

Enter Phone No:

Activate Windows
Go to Settings to activate Windows.

Fig.8.5 Application Form

KOHA

HOME PATIENT DOCTOR

PLEASE FILL THIS FORM TO CREATE AN ACCOUNT

USERNAME:

PASSWORD:

CONFIRM PASSWORD:

QUALIFICATIONS:

SPECIALIZATIONS:

INFO:

Already have an account?

DOCTOR SIGN UP.

Activate Windows
Go to Settings to activate Windows.

Fig.8.6 Doctor Sign-Up Page

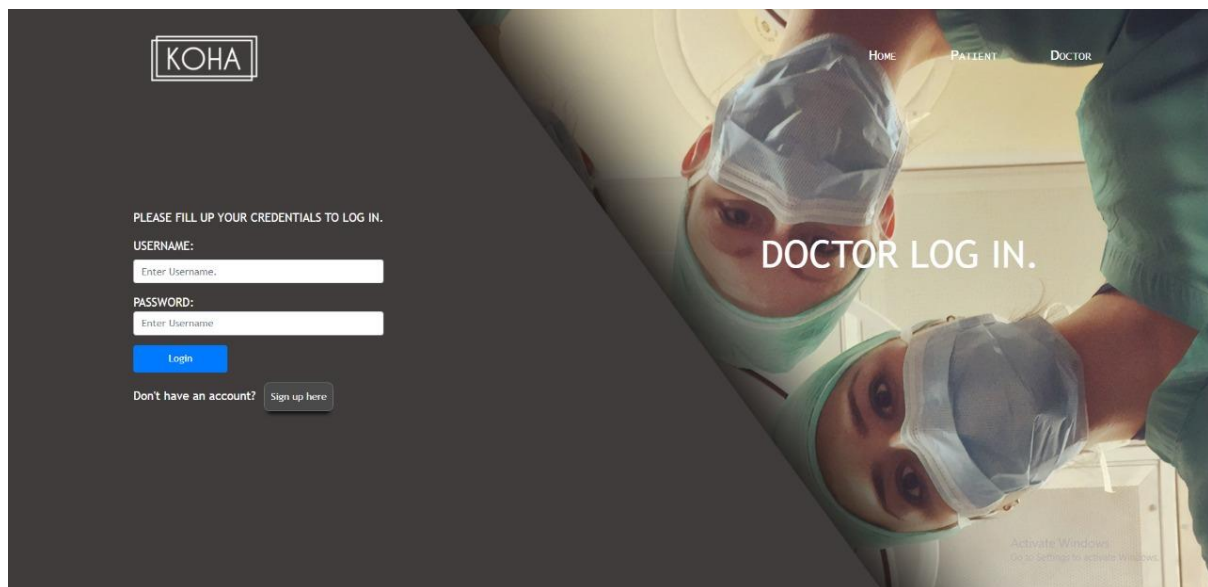


Fig.8.7 Doctor Log-In Page

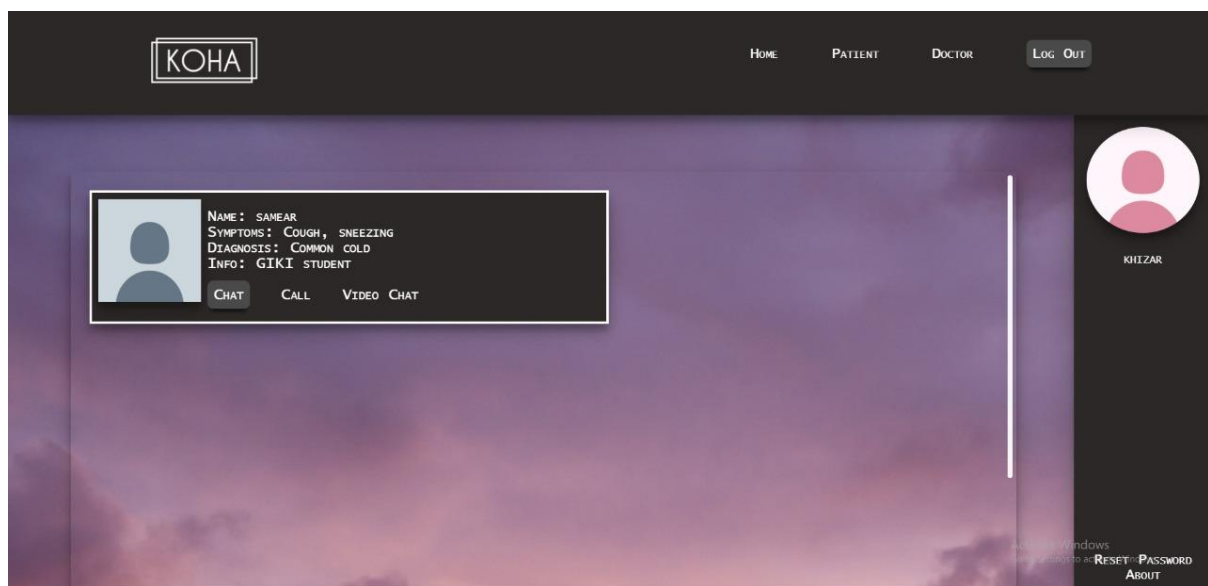


Fig.8.8 Doctor Dashboard

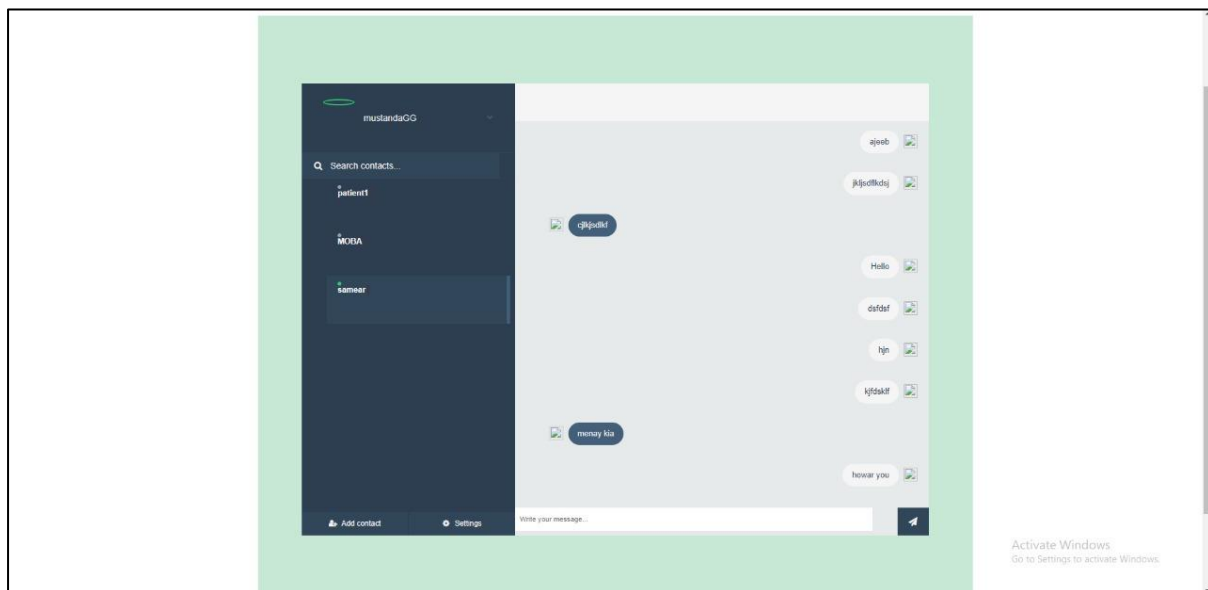


Fig.8.9 Chat Interface



Fig.8.10 Symptom Checker UI

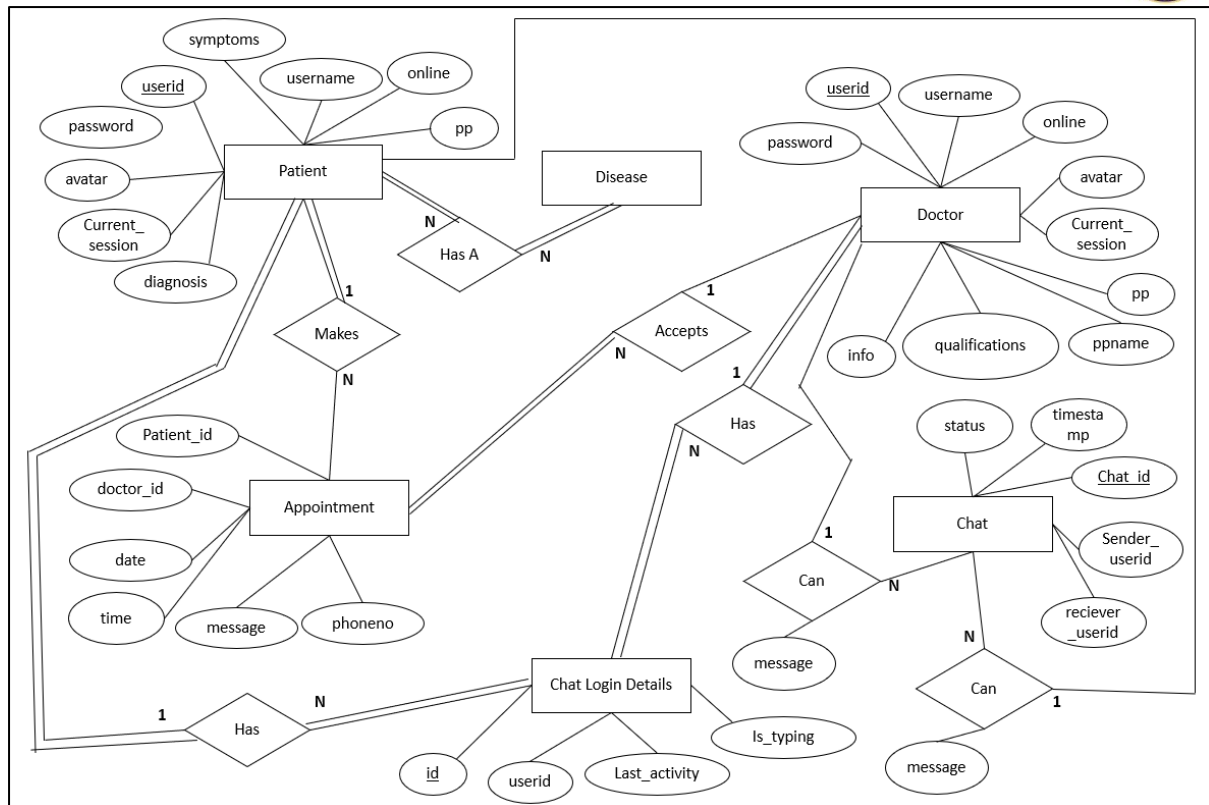


Fig.9 Entity Relationship Diagram