



**FINAL SEMESTER ASSESSMENT  
(FSA) B.TECH. (CSE)  
VI SEMESTER**

**UE18CS355 – OBJECT ORIENTED ANALYSIS AND  
DESIGN WITH SOFTWARE ENGINEERING  
LABORATORY**

**PROJECT REPORT**

**ON**

**MEDIXO - Online doctor appointment system**

**SUBMITTED BY**

**Prathik Bafna  
Sandeep Bhat  
Supreet Ronad**

**PES2201800058  
PES2201800632  
PES2201800705**

**JANUARY – MAY 2021  
DEPARTMENT OF COMPUTER SCIENCE &  
ENGINEERING,  
EC CAMPUS,  
BENGALURU – 560100, KARNATAKA,  
INDIA**

<b>TABLE OF CONTENTS</b>		
<b>Sl.No</b>	<b>TOPIC</b>	<b>PAGE No</b>
	ABSTRACT	2
<b>1.</b>	SOFTWARE REQUIREMENTS SPECIFICATION	3
<b>2.</b>	PROJECT PLAN	18
<b>3.</b>	DESIGN DIAGRAMS	24
<b>4.</b>	MODULE DESCRIPTION	30
<b>5.</b>	TEST CASES	31
<b>6.</b>	SCREENSHOTS OF OUTPUT	37

## **Abstract**

The proposed project is a smart appointment booking system that provides patients or any user an easy way of booking a doctor's appointment online. This is a web based application that overcomes the issue of managing and booking appointments faced by users. The system is intended to be used in medical clinics in order to help improve appointment services and to save time and money.

The task sometimes becomes very tedious for the compounder or doctor himself in manually allocating appointments for the users as per their availability. Hence Doctor Appointment Reservation System is a self-contained system that allows patients to book appointments and doctors to manage appointments. This project offers an effective solution where users can view various booking slots available and select the preferred date and time.

Doctors can confirm the appointment booked by patients and patients can view the status. This system also allows users to cancel their booking anytime. The application uses php and html as a front-end and sql database as the back-end.

# 1. Software Requirement Specification

## Introduction

### Purpose

The purpose of this document is to highlight the functional and nonfunctional requirements of a medical consultation booking application and to provide a detailed description of the system that is to be developed. This document includes a thorough study on various requirements fulfilled by the application, use cases, various constraints on the implementation, and other aspects involved in the project. This document serves as a brief to outline our project and as a foundation for further developments in the future.

### Intended Audience

The primary audience of this document is the project development team that is involved in making the consultation booking application. The secondary audience of this document are the stakeholders that the project/application is intended for - doctors and patients: - to help patients in booking an appointment with a doctor and to make this process as smooth and efficient as possible and to help doctors in providing a platform for access to patients and to act as a mechanism to keep track of appointments that patients have scheduled with him/her.

### Product Scope

The objective of this application is to provide an integrated platform for doctors to interact with patients - In a conventional scenario, patients obtain the phone number of a clinic/hospital usually recommended to them by one of their colleagues, peers or family members and then proceed to set an appointment with a doctor with a receptionist or nurse acting as the mediator between the doctor and patient to set the appointment.

However, with an application like the one that this project aims to build, the process of browsing for a doctor becomes more accessible as the patient can now choose from several doctors of various specializations, the appointment booking process doesn't need a mediator to be facilitated.

This platform also helps doctors by providing them with a better reach especially doctors who're looking to establish themselves - they simply have to register themselves on the platform with the help of an administrator and the patients can then look them up.

### References

- IEEE 830 Template
- *Practo: Healthcare Done Right.*
- *Online Appointment Management System*

# Overall Description

## Product Perspective

This Doctor Appointment Reservation System is a self-contained system that allows patients to book appointments and doctors to manage appointments. Various stakeholders are involved in this system. The system is intended to be used in medical clinics in order to help improve appointment services. There already exists a wide variety of similar products on the market. Most of these commercial applications are intended to be very general and try to cover any possible business. Although these products exist, Medical clinics still exhaust a great deal of time and money scheduling their appointments. This is because Medical clinics have very specific needs that differ from other businesses. our system is a web-based application that is tailored to a Medical clinic's needs.

## Product Functions

This product aims to make appointments easily. Users should be able to search for a Doctor either by name or Hospital and be provided with a list of possible appointments. Patients can choose a suitable appointment after which Doctors can either confirm or reschedule the appointment. Both the doctors and patients should be able to check their schedules whenever they choose to do so. Admins must have access to verify accounts as well as ensure that appointments run smoothly.

## User Classes and Characteristics

Broadly classified into Users and admins there are three types of User classes we can make a note of. Users are further classified into Patients and Doctors.

**Patients:** These are the people who want to make the appointment. They are perceived to have almost no experience with the app and must therefore be provided with an easy to use interface

- Attributes - Name, Age, Address, Medical History, Email, Phone No
- Book Appointment - The patient shall be able to book an appointment with any doctor. There are some types of appointments that can only be made by a doctor, e.g. a surgery.
- Cancel an Appointment - The patient can cancel his appointment, as long as he cancels it 60 hours in advance.
- Create and Edit Personal Account - The patient can register and will be able to modify his/her personal information such as his/her address, telephone number, etc.
- View Appointments/Schedule - The patient will be able to consult all of his or her active appointments.

**Doctors:** These are the specialists for whom appointments are being booked. They are perceived to have some experience with the app but are still provided with a user-friendly interface.

- Attributes - Name, Specialisation, Field, Age, Phone No, Email

- **Confirm Patient Appointments** - The doctor can either confirm patient appointments if possible or they may need to cancel or reschedule patient appointments.
- **View Appointments** - A doctor may need to identify all of the appointments that have been scheduled for a specific day or week.
- **Schedule Special Appointments** - When surgery is needed, an administrator must schedule the appointment on the doctor's behalf. The patient will not have control over the time and date.

**Administrator** - They are responsible for maintaining and overseeing the database of the system including making sure of no overlapping of appointments. They are expected to be technically skilled enough to operate the backend of the software.

- **Management of Catalogues** - The administrator will verify doctors who register, add department information, add types of appointments, and set time limits for each type of appointment.
- **Create Doctors Accounts** - Each time a new doctor joins the clinic, the administrator needs to add the doctor to the network.
- **Delete Doctors Accounts** - If a doctor has to unregister, the administrator will delete this doctor's information and re-assign all of his/her appointments to another doctor or cancel according to availability.

## **Operating Environment**

1. Operating System: Windows, Linux, macOS
2. Web Browser: Google Chrome, Mozilla Firefox

## **Design and Implementation Constraints**

1. **User Interface Constraints** - The user should be familiar with basic browser navigation and be able to understand the functionalities provided by the system.
2. **Hardware Constraints** - The system must be user-friendly supporting JavaScript applications. The system should work on any internet browser with GUI whether the underlying Operating System is Windows, Linux/Unix, or Macintosh. Only the old obsolete command-line web browsers won't be able to connect to the system.
3. **Software Constraints** - The system requires to be run on Firefox 4 and above, Google Chrome 10 and above.
4. **Data Management Constraints** - Storage is limited and will have to accommodate no of users.
5. **Operational Constraints** - The system will be constrained by the max limit of users and OS.
6. **Memory Constraints** - The system is supported by any web browser by and can be accessed by any computer with internet access. The following is needed for the architecture of our web browser: Pentium 4 processor, 256 MB ram, 250 GB HDD

## **Assumptions and Dependencies**

- It is assumed that the Patients will be well versed in handling an online interface.
- It is assumed that the Doctors will be well versed in handling an online interface.
- The only language used in the app is English, hence it is assumed that the user knows English

## **External Interface Requirements**

### **User Interfaces**

#### **Patient Interface**

1. On opening the website, the patient will be taken to the home screen where the user can choose whether to Log In/Sign Up or browse through the medical information provided.
2. A user not logged in does not have the ability to search for doctors, hospitals.
3. Searching for Doctors can be done by Name, Specialisation, or even time of appointment. Hospitals and Pharmacies can be looked up by Name or Location. Medicine can be looked up using Name or Illness it cures.
4. Each user has a profile page that contains all the info required for creating a basic account. Users can choose to either save their personal record of medical history or refill the required for each appointment they make.
5. When the users have found a suitable Doctor to make an appointment for, they will be alerted to log in if not done already before moving ahead.
6. The user can choose whether or not they would like to book an online appointment or an offline one, and the date and time must be confirmed.
7. Once the appointment is made, users can track their upcoming appointments on their dashboard/schedule. Completed appointments can be looked up if desired.

#### **Doctor Interface**

1. As a doctor, the signup page will acquire info such as specialization, dates available, timings one sits for, and offline or online preference.
2. Doctors can confirm whether or not an appointment would be possible at the requested time or whether it has to be rescheduled.
3. The dashboard would list the patients who have requested an appointment with all the necessary details.

### **Software Interfaces**

1. The User Interface of this application will be built using the Django web framework in Python
2. The database used is SQL/SQLite.
3. The data items that need to be sent are booked appointments and confirmations which will be transferred to and from the system.

## Communications Interfaces

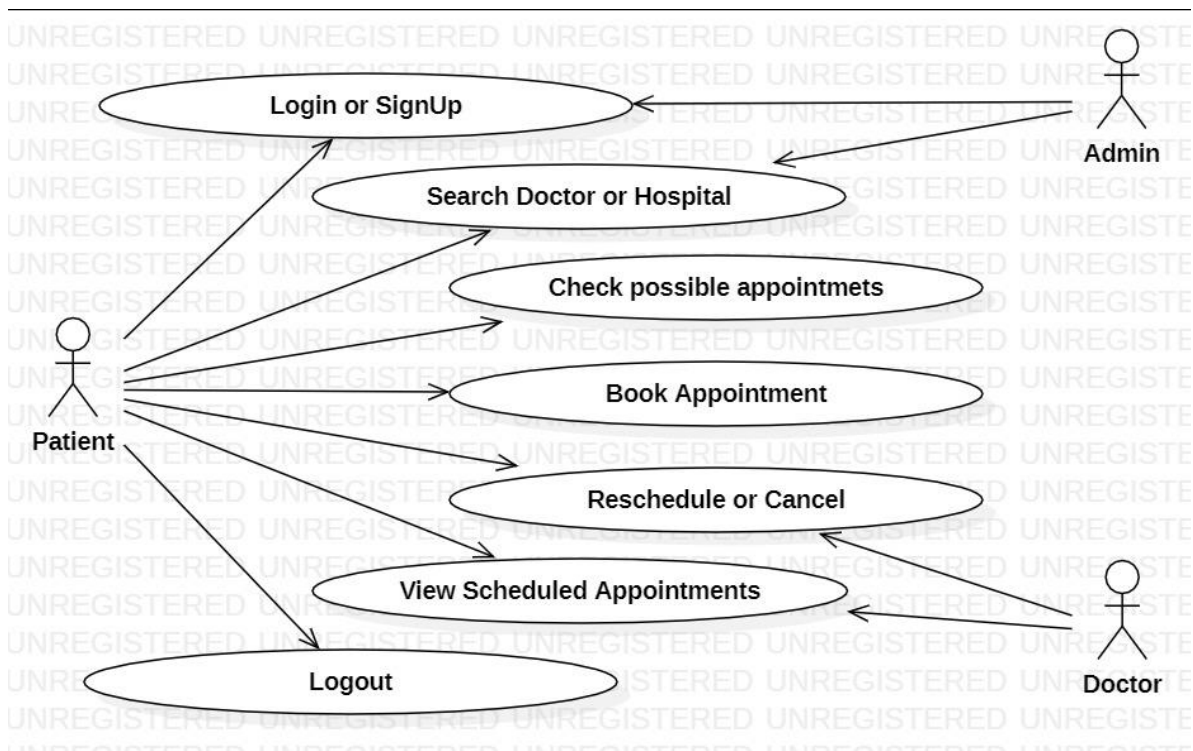
1. The client and the server will communicate through web browsers.
2. Communication standards to be used would be HTTP would be used as communication standards.
3. The message formatting to be used is JSON.

## Analysis Models

### Actors

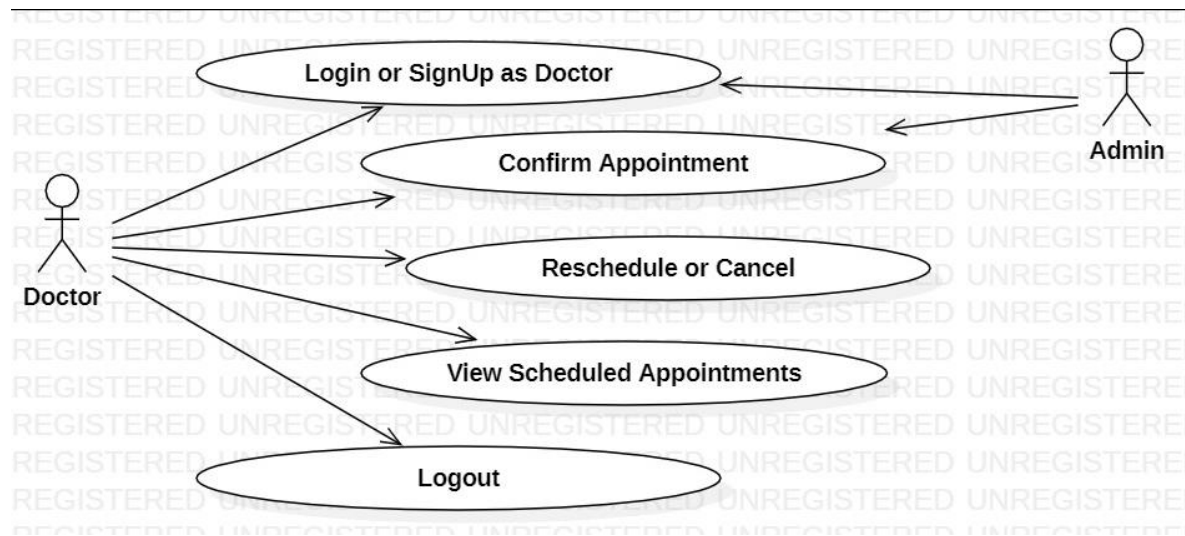
- **Patient:** The patients can browse through the different options of Doctors, Hospitals, and Pharmacies.
- **Doctor:** The doctors can accept or reschedule appointments that are assigned to them.
- **Admin:** The admin shall be able to manage users of any type and account.

### Patient Use Case Diagrams



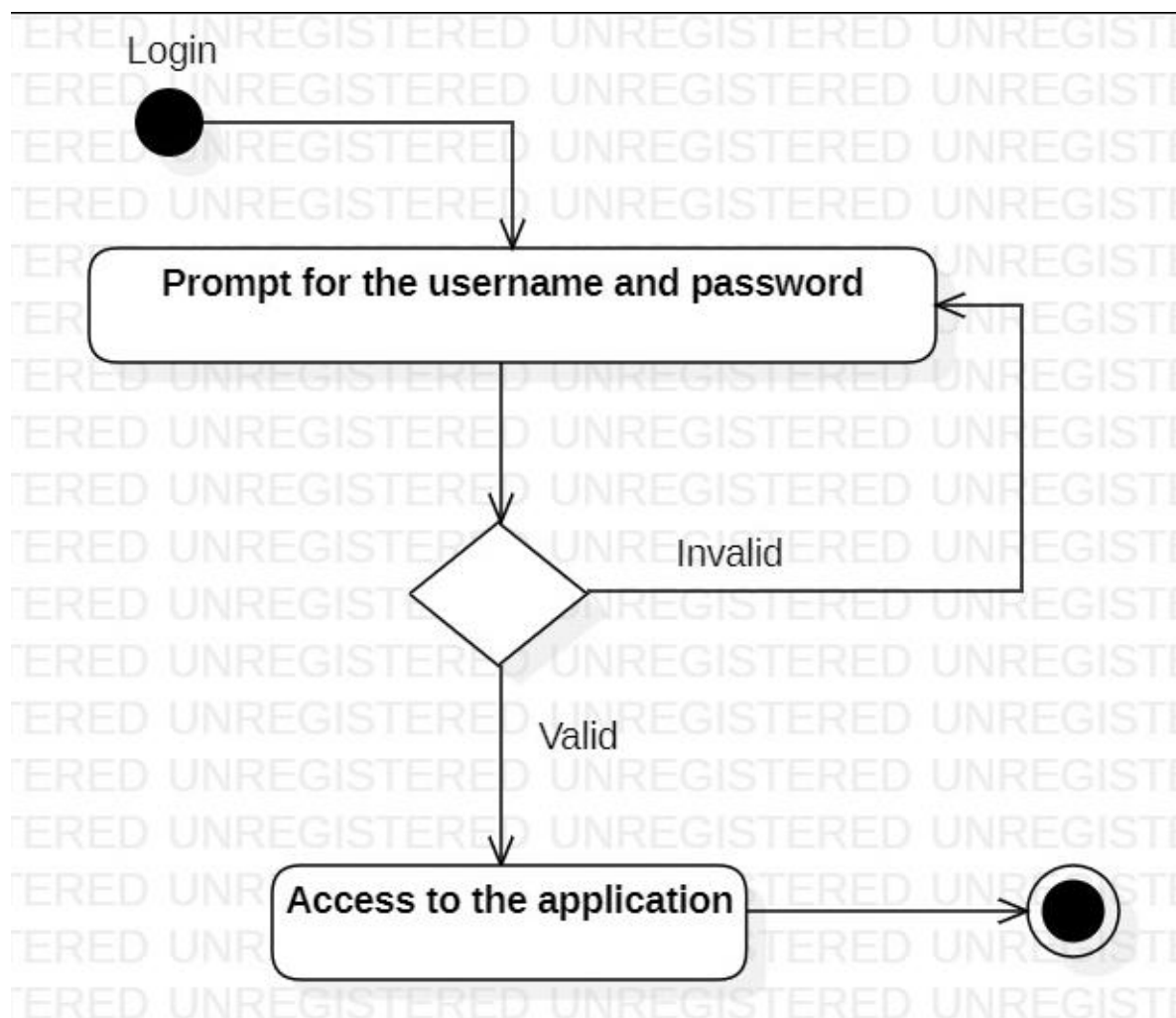


## Doctors Use Case Diagrams

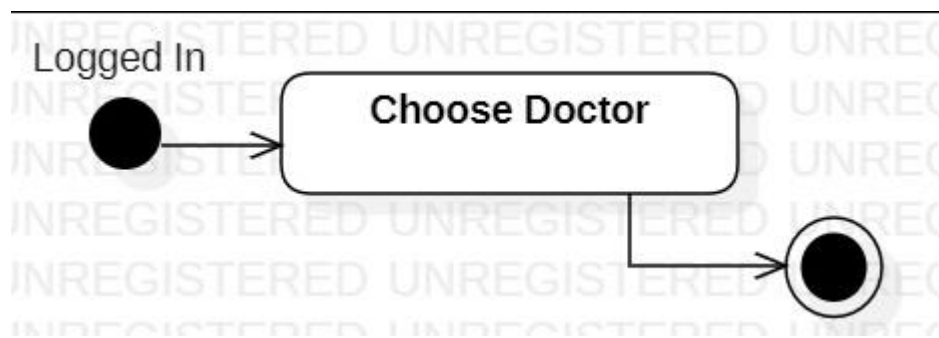


## Use Case Descriptions

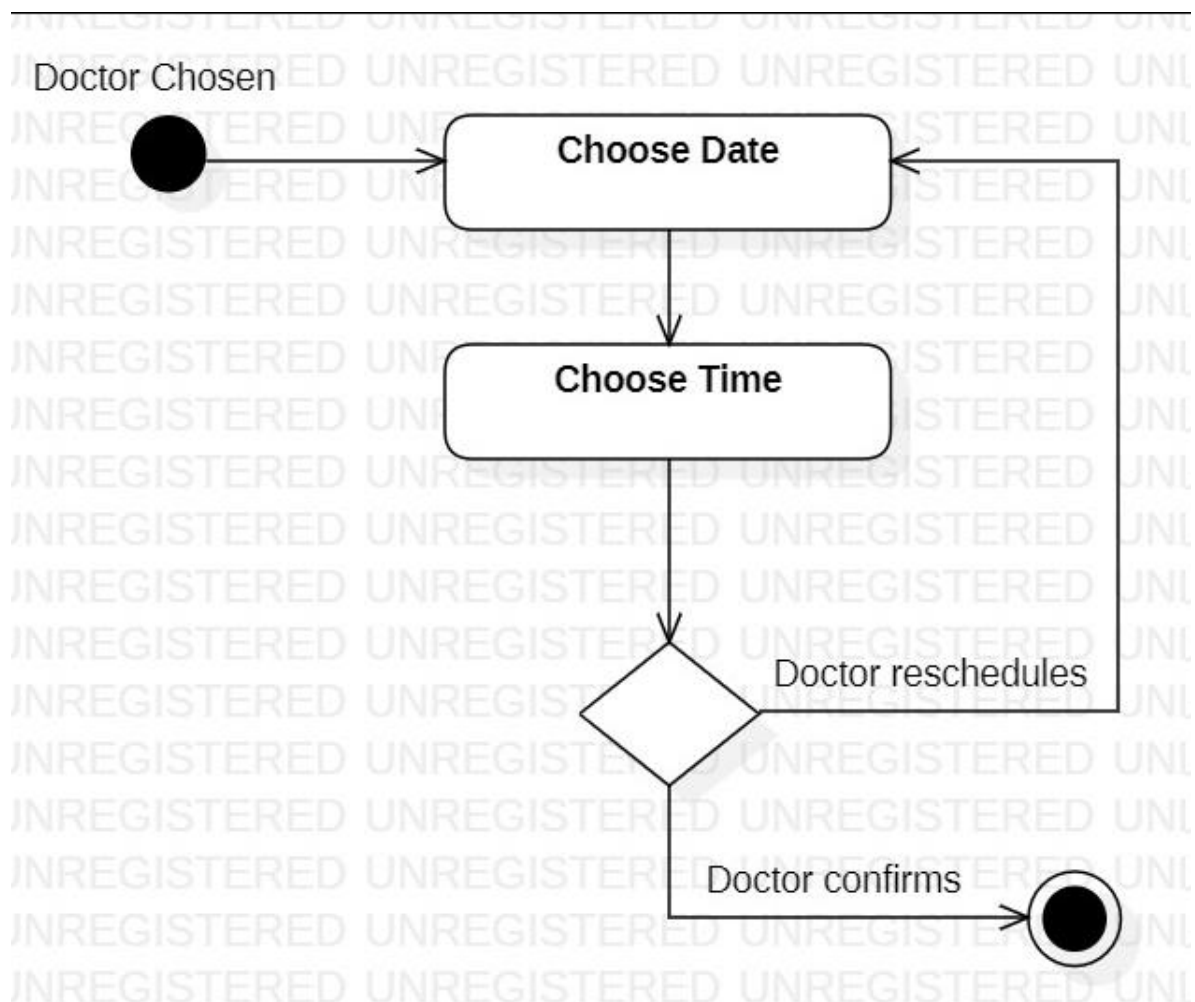
1. **Login:** The user can login to the application with a valid username and password.



2. **Search Doctor:** The patient can choose any Doctor registered on the Medixo app.



3. **Book Appointment:** The patient has to choose a Date and Time to set up an appointment, giving the necessary details.
4. **Confirm Appointment:** The doctor must confirm or reschedule appointments according to their needs.



5. **Logout:** All users must log out of the application when done.

# System Features

## Authentication

### Description and Priority

The system should enable patients and doctors to register and later log in as their respective roles.

**Priority:** High

### Stimulus/Response Sequence

- They shall enter their username and password.
- In the case of a patient, collect user information (Name, Date of birth, Address, Password, etc.).
- In the case of doctor collect (Names, Department, Hospital, Telephone, etc.).
- The information given shall be verified as valid.
- If registering, add the user to the database.
- If Login access shall be granted/denied.

### Functional Requirements

**REQ 1:** The system should enable patients and doctors to log in after authenticating their passwords and email ids.

**REQ 2:** The system should enable patients and doctors to register to ask for information about each respective role.

## Logging Out

### Description and Priority

The system should enable patients and doctors to log out successfully once they have finished their work.

**Priority:** Medium

### Stimulus/Response Sequence

- Log the user out when the user clicks on the log out button.
- End session time once logged out.

### Functional Requirements

**REQ 3:** The system should enable patients and doctors to log out successfully ending session time.

## **Search Requirements**

### **Description and Priority**

Patients must be able to search for Doctors by Name, Hospital or Specialisation

**Priority:** High

### **Stimulus/Response Sequence**

- The Patient enters the name or specialization or ID of the doctor.
- The system displays all doctors that fit the patient's criteria.
- The system shall display the doctor's available

### **Functional Requirements**

**REQ 4:** The system should allow patients to search for available doctors of a particular field with an easy-to-use search bar interface.

## **Availability**

### **Description and Priority**

Doctors must be able to enter in their available time appointments.

**Priority:** High

### **Stimulus/Response Sequence**

- The doctor will enter the time he'll be available.
- This information is saved in the database and showed when necessary

### **Functional Requirements**

**REQ 5:** The system should allow Doctors to set the days and times they are available to sit for appointments.

## **Booking Appointments**

### **Description and Priority**

Patients can choose a time and date for appointments which must be either confirmed or rescheduled by Doctors.

**Priority:** High

### **Stimulus/Response Sequence**

- The system shall check if the patient is logged in or not.
- The patient shall select the preferred time among the available slots of the particular doctor to be booked.
- The system shall generate a unique booking ref for each appointment.
- The Doctor will either confirm the appointment or request to reschedule.
- Reservations shall be modified as if required.
- The system shall make the necessary updates after changes have been made.

### **Functional Requirements**

REQ 6: The application should allow Patients to book an appointment from among the available slots of the doctor chosen.

REQ 7: The application should allow doctors to confirm the appointments and add them to the schedules of both parties involved. Else they should allow the doctor to request for rescheduling.

REQ 8: The application should allow patients to modify or cancel their bookings without having to re-enter all the patient's information.

### **Modify Profiles**

#### **Description and Priority**

Patients and Doctors can view and modify their profiles.

**Priority:** Medium

#### **Stimulus/Response Sequence**

- Users enter new or modified information
- This information then replaces the old information in the database.
- Patients can choose to add medical history at any time.

### **Functional Requirements**

**REQ 9:** The system should allow Patients and Doctors to view and modify their profile at any moment they choose to.

### **Manage Users and Appointments**

#### **Description and Priority**

Admins should be able to login and manage Users and Appointments

**Priority:** High

#### **Stimulus/Response Sequence**

- The admin shall enter their username and password.
- The information given shall be valid.
- Administrators access databases and add new customers.
- Administrators delete any user due to some rules from the database.
- Administrators change patient's or doctor's information.
- Administrators change patient's or doctor's information like last name, email, password, department etc.
- Administrators can add, delete or modify scheduled appointments to the database
- Administrators make sure there are no conflicting appointments.

### Functional Requirements

**REQ 10:** The system should enable administrators to log in with the required authentication.

**REQ 11:** The system should allow the administrator to manage the database, i.e. manage and verify Users, as well as maintain a conflict-free schedule.

## Other Non-functional Requirements

### Performance Requirements

ID	REQUIREMENT
PER1	The Server should be able to service multiple terminal connections simultaneously. At least 100 internet users should be able to use the server without any connection delay issues
PER2	The server must run error free while operating with a huge set of data.
PER3	For a specific time on a specific date, all users have the same priority to reserve that slot. Users are served in a First-Come First-served fashion.
PER4	The software should be able to process more that 90% of the user queries (like reserving an appointment, modifying an appointment, etc.) in a fraction of 1 second.

### Safety Requirements

ID	REQUIREMENT
S1	All system data must be backed up every 24 hours and the backup copies stored in a secure location which is not in the same building as the system
S2	The system's backup plan should support the protection of the DB data and assure the functioning of the system after recovery of a crash

## Security Requirements

ID	Requirement
SEC1	All external communications between the system's data server and clients must be encrypted.
SEC2	User authentication is needed to use the system. Users should not be able to create or delete reservations for other users. Thus, a track of each user's reservations is needed in order to apply these constraints.
SEC3	Personal information collected upon registration should not be revealed for the sake of privacy. A user needs only to know whether a certain time slot is reserved or not, without the need to reveal the patient's name
SEC4	The access permissions for system data may only be changed by the system's data administrator.

## Software Quality Attributes

**Usability:** The System's interface should be user-friendly;

- All web pages used by the users should be consistent with standardized colours and fonts.
- In case of errors, help links or instructions should be provided for correct use.
- An HTML online documentation should be available and accessible to the users on all pages and steps of system usage.

**Reliability:** The system should ensure that the user actions are performed correctly:

- If the user entered invalid data, the system should be able to recover and react robustly with the error, by sending a message back to the user
- The System should be set up on a dedicated computer machine for s 24-hours a day and 7-days a week availability.
- The system's backup plan should support the protection of the DB data and assure the functioning of the system after recovery of a crash

### Portability

- Because the appointment scheduler is a web-based system, it can be accessed by users having any operating system with GUI internet browser installed; thus ensuring high portability and minimum system requirements.
- 100% of the client code is host independent.
- The PHP language used at the hosting machine is system independent.

### Correctness

- All algorithms implemented in the system should be correct; meaning they should perform as required. The testing phase ensures correctness of the software by trying all possible cases and matching their output with the documentation.

## Interoperability

- Since the system relies on the web agents (browsers) in its interoperability and all web agents interoperate with other software packages, the system can work with other software. For example, it can provide an Acrobat Reader version of the documentation.

## Maintainability

- The client-side scripting of the system can be easily fixed and enhanced due to simplicity. The server-side scripting is well maintainable as well as all models and code is preceded by related design and this requirements document

## Business Rules

1. Payment confirmation is given as per consultation fee of the specific doctor but payment to be done physically upon appointment.
2. Qualifications of a doctor are assessed when a doctor registers on the platform to ensure legitimate doctors are registered with the platform.

## Other Requirements

### Performance Requirements

As the medical field is fraught with emergencies and patients that require assistance immediately it is necessary that the system handle at least 200 concurrent users including the doctors. The system performance can never go slack as the appointments will be unpredictable.

### Logical database requirements

The database should be reliable and well designed as all appointments must be scheduled with no errors or overlaps. The database will be accessed concurrently and must be kept consistent throughout.

## Appendix A: Glossary

1. **Patient:** A user who wishes to book an appointment for medical consultation.
2. **Doctor:** Professional users with a valid background to give medical advice and prescribe medicine.
3. **Admin:** Person who handles verification of account as well as data to ensure no overlapping of appointments.
4. **Hospital:** Hospitals which have been affiliated with Medixo or have Doctors who have registered with the app.
5. **Appointment:** A booked time slot and date corresponding to both the Doctor and Patient which has to be unique with respect to Time, Date and Doctor.
6. **Schedule:** Each user has a unique list of upcoming non conflicting appointments.
7. **Specialisation:** The Doctors' field of medicine in which they are legally allowed to treat patients, e.g.: Paediatrics, Dermatology etc.
8. **Timestamp:** Date and Time including seconds.
9. **Search Doctor:** Search for doctors either by Name or Hospital or Specialisation.



10. **View available Appointments:** View timestamps in which the chosen Doctor is available for appointments.
11. **Confirm Appointments:** Doctors confirm each requested appointment at their convenience.
12. **Reschedule Appointments:** Change a booked appointment timestamp to a more feasible time for both parties.
13. **Cancel Appointments:** If necessary, appointments can be cancelled due to various reasons and schedules must be updated accordingly.
14. **JSON:** JavaScript Object Notation
15. **HTTP:** Hypertext Transfer Protocol

## Appendix B: Field Layouts

### Patient Details

FIELD	LENGTH	DATA TYPE	DESCRIPTION	MANDATORY
Patient_Id	16	Integer	Identifier of the patient	YES
Name	50	String	Name of the patient	YES
Age	3	Integer	Age of the patient	YES
Address	200	String	Address of the patient	YES
Username	20	String	Unique username to log in	YES
Password	8	String	Password authentication for	YES

### Doctor Details

FIELD	LENGTH	DATA TYPE	DESCRIPTION	MANDATORY
Doctor_Id	16	Integer	Identifier of the doctor	YES
Name	50	String	Name of the doctor	YES
Age	3	Integer	Age of the doctor	NO
Address	200	String	Address of the doctor	NO

Username	20	String	Unique username to log in	YES
Password	8	String	Password authentication for	YES
Specialization	50	String	Field of medicine	YES

### Appointment Details

FIELD	LENGTH	DATA TYPE	DESCRIPTION	MANDATORY
Doctor_Id	16	Integer	Identifier of the doctor	YES
Patient_Id	16	Integer	Identifier of the patient	YES
Date	200	Date	Date of appointment	YES
Time	200	Timestamp	Time of appointment	YES

## 2. Project Plan

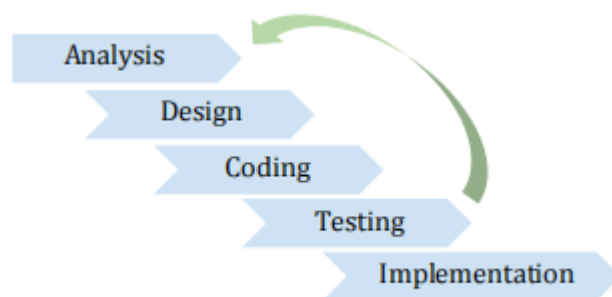
### Development Cycle

#### Method Chosen

During a system development life cycle, two main factors are considered: to emphasize the process and the quality of the software and process itself. Since software development is a complex field that contains countless variables impacting the system, any developer would always seek an organized structure that could be used as a base for developing a system. All software systems are imperfect because they cannot be built with mathematical or physical certainty. Thus, system development methods are introduced to provide the developer a baseline of processes and sequences to follow while building a system.

Different software developing methods have different characteristics of processes to reach the completion of a system. Based on the research carried out and analysis of system requirements for the proposed Online Doctor Appointment System, we have chosen the Iterative Model to go about it:

The Iterative model does not need the full list of requirements before the project starts. The development process may start with the requirements to the functional part, which can be expanded later. The process is repetitive, allowing to make new versions of the product for every cycle. Every iteration includes the development of a separate component of the system, and after that, this component is added to the functional units developed earlier.



#### Why was it chosen?

The iterative model is particularly popular when the requirements to the final product are not strictly predefined, it can be applied to large-scale projects or when the main task is predefined, but the details may advance with time.

- Some functions are quickly developed at the beginning of the development lifecycle.
- Paralleled development can be applied.
- The shorter the iteration is, the easier the testing and debugging stages are.
- It is easier to control the risks as high-risk tasks are completed first.
- Problems and risks defined within one iteration can be prevented in the next sprints.
- Flexibility and readiness to the changes in the requirements.

## Software Tools

### Planning Tools

**Gantt Chart** : A Gantt chart is a type of bar chart that illustrates a project schedule. It shows the dependency relationships between activities and the current schedule status. This chart lists the tasks to be performed on the vertical axis, and time intervals on the horizontal axis.

**WPS tool** : Help people read, edit and transfer PDF files online easily.

**LibreOffice** : LibreOffice is a free and open-source office productivity software suite, a project of The Document Foundation. The LibreOffice suite consists of programs for word processing, creating and editing of spreadsheets, slideshows, diagrams and drawings, working with databases, and composing mathematical formulas (suitable for ubuntu os).

**Google Docs** : Google Docs is a word processor included as part of the free, web-based Google Docs Editors suite offered by Google. With Google Docs, you can create and edit text documents right in your web browser—no special software is required. Multiple people can work at the same time, you can see people's changes as they make them, and every change .

### Design Tools

**Star UML** : A sophisticated software modeler for agile and concise modeling. StarUML is built as a modular and open tool. It provides frameworks for extending the functionality of the tool. It is designed to allow access to all functions of the model/meta-model and tool through COM Automation, and it provides extension of menu and option items.

**Creately** : Visual software to draw and collaborate on ideas, concepts and processes. Use it as a chart and diagram maker/collaboration tool/visual space. For creating graphs and charts. We also used this for drawing UML /use case diagrams.

### Version Control

**Git**: Git tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to. Git also makes collaboration easier, allowing changes by multiple people to all to be merged into one source.

### Development Tools

**Stack Overflow**: Stack Overflow is the largest online community for programmers, which is visited by many developers every month. This is a place where they learn, share knowledge, and advance their careers.

**GitHub**: GitHub is a website for developers and programmers to collaboratively work on code. The primary benefit of GitHub is its version control system, which allows for seamless collaboration without compromising the integrity of the original project. The projects on GitHub are examples of open-source software.

## Bug Tracking Tools

**Marker:** We can convert screenshots from any website into a powerful bug report directly into your existing tools.

## Testing Tools

**Selenium:** It is used to perform web application testing across various browsers and platforms like Windows, Mac, and Linux using languages like Java, PHP, C#, python,

## Project Deliverables

### Project Management

- Project Charter: Reuse.

The project is very similar to Practo, an online appointment booking application and hence the project charter can be reused.

- Project Planning: Build.

As it will be easier to follow a self-made plan we have decided to build the project planning document.

### Requirements Gathering

- Product/Software Requirements: Build.

The Software Tools and other functionalities should be decided according to the discretion of the developer and hence we chose to build it.

- Customer Requirements: Reuse.

Apps such as Practo have already detailed out a list of Customer Requirements which will be easy to follow

### Design and Analysis

- User Interface Design: Build.

The design of the frontend is always the most unique of all applications and thus we chose to rebuild it in our own image.

- Database Design: Build.

Aside from the common functionalities in every similar app, it is better to design one's own database to make sure the schema fits our vision.

### Site Software Development

- Frontend: Build.

The frontend shall be made according to user interface design.

- Backend: Build.

The backend of the system shall be built keeping in mind the schema.

### Testing and Production

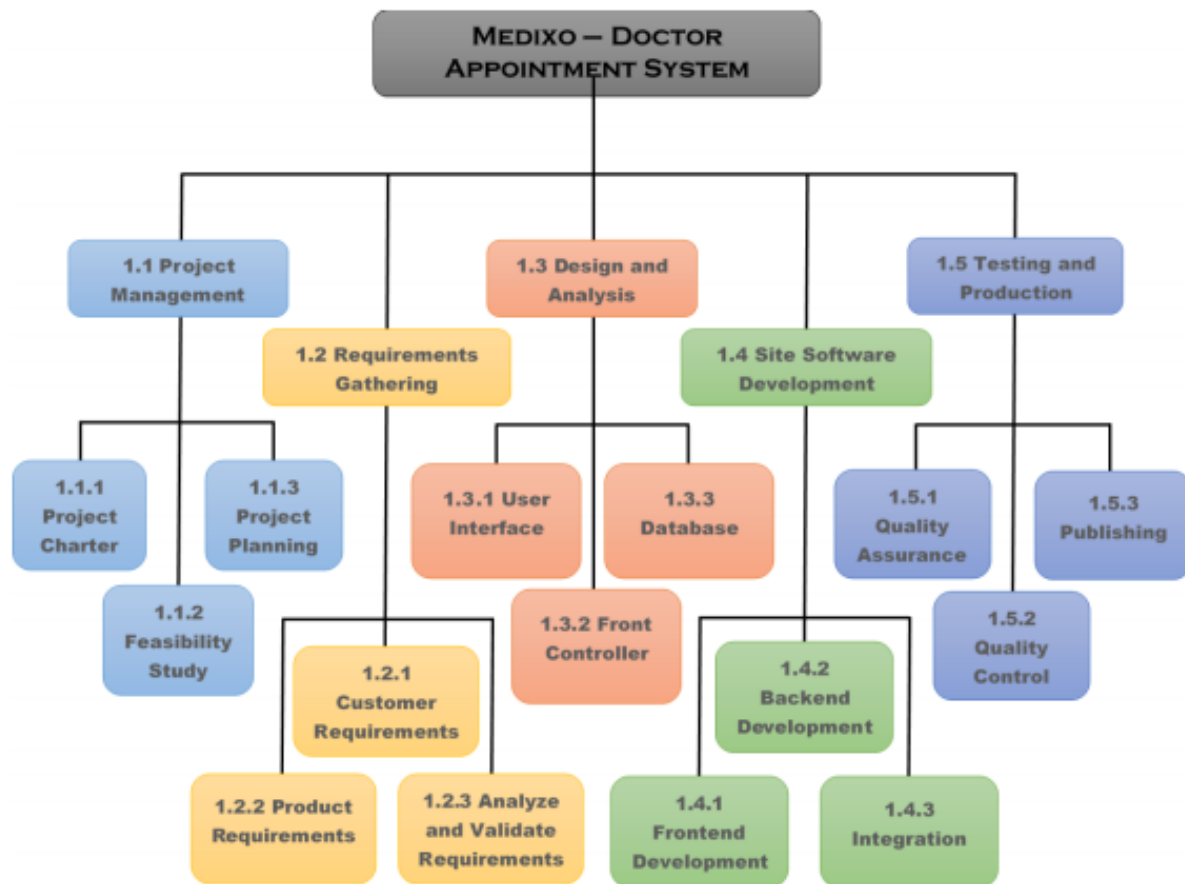
- QA/QC Testing: Reuse.

Software tools that are already available will be used to test the product's functionality.

- User Acceptance: Build.

User Acceptance testing will be done based on a sample of users, Any changes required thus will be considered and implemented from their responses.

## Work Breakdown Structure



## Estimate of effort

**COCOMO** is one of the most generally used software estimation models in the world. The key parameters which define the quality of any software products, which are also an outcome of the Cocomo are primarily Effort & Schedule:

- **Effort:** Amount of labor that will be required to complete a task. It is measured in person months units.
- **Schedule:** Simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put in. It is measured in the units of time such as weeks, months.

Under the COCOMO model, we shall proceed with the organic type since the project deals with developing a well-understood application program, the size of the development team is reasonably small, and the team members are relatively experienced in developing similar methods of projects.

Formulas and Constants:

$E= a(KLOC)^b$					
$time = c(Effort)^d$	Software Projects	a	b	c	d
$Personrequired = Effort/time$	Organic	2.4	1.05	2.5	0.38

KLOC is the estimated size of the software product indicated in Kilo Lines of Code, the effort is measured in Person-Months and The development time is measured in months.

These formulas are used as such in the Basic Model calculations, as not much consideration of different factors such as reliability, expertise, and other cost drivers which is difficult to determination at this stage is taken into account, henceforth the estimate is rough.

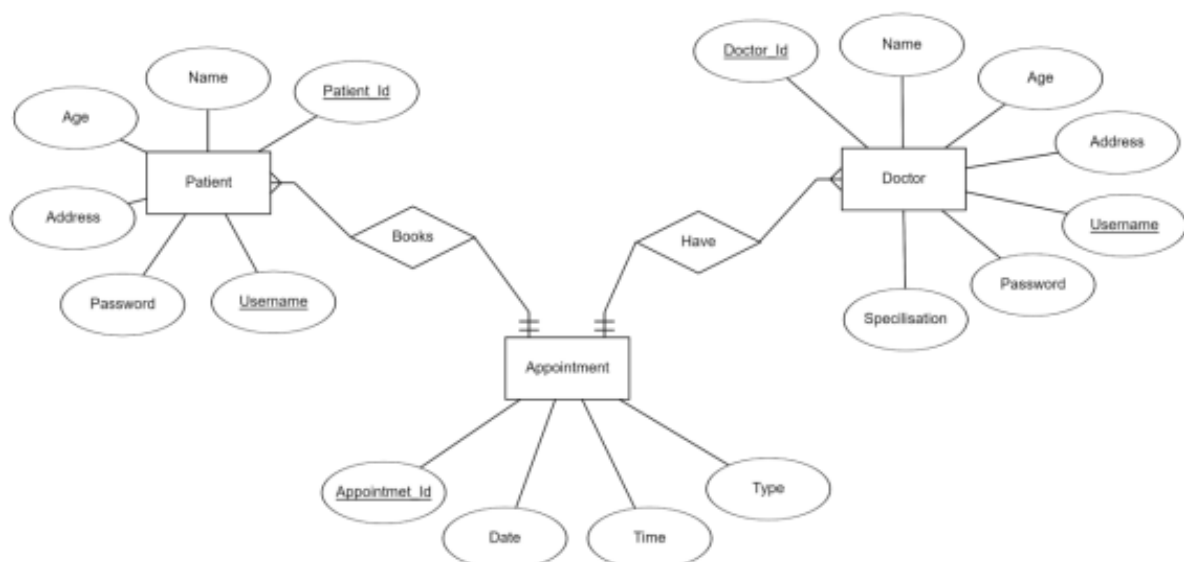
The KLOC value chosen for our project is 6.5

Average effort,  $E = 2.4 \times (6.5)^{1.05} = 17.11$  person-months

Development Time =  $2.5 \times (17.11)^{0.38} = 7.35$  months

Average persons required =  $[17.11 / 7.35]$  approximated = 2 persons

## ER Diagram



There are 3 entities in the project, namely

**a. Patient:** The patient/user table possesses details such as age, address, a unique patient ID, username, and password.

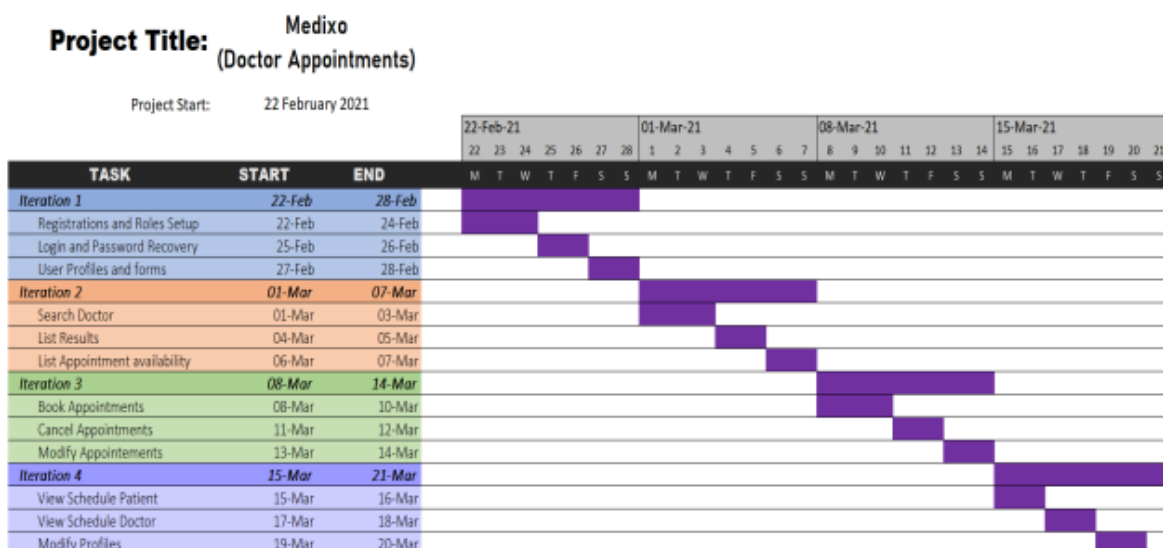
**b. Doctor:** The doctor/user table has details of the doctor like age, username, and specialization.

**c. Appointment:** The appointment table consists of the time, date, and type of every appointment

## Tentative Deadlines

We have decided to break down the development process into 4 iterations or sprints. Each sprint is tentatively a week-long and focuses on a particular functional module of the web application.

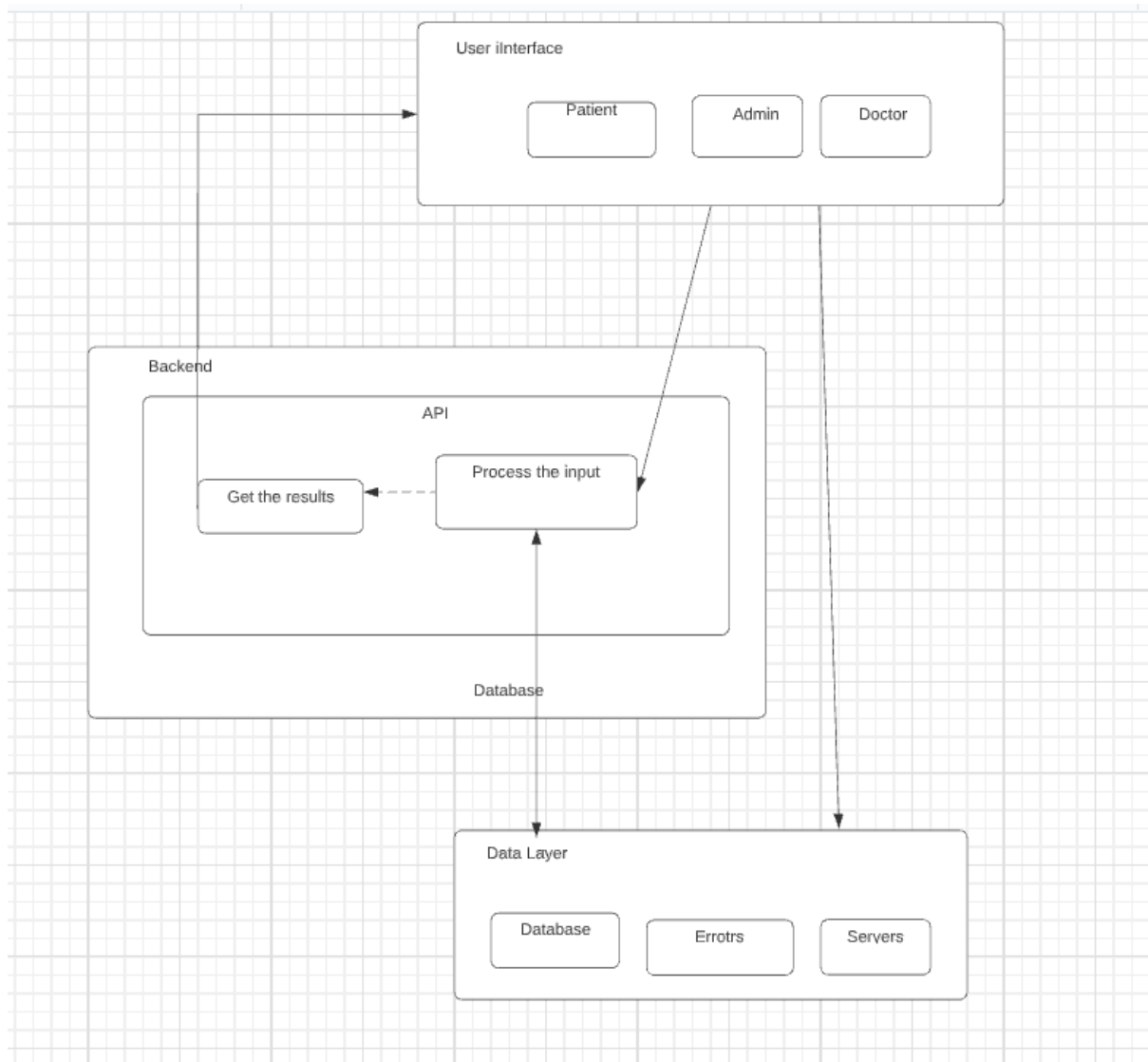
Before every sprint begins, we plan the next step in development to decide which functionalities have more priority. Those that have a high cost(time) are the complex functions that will require more effort to develop. At the end of each sprint, we review the iteration we focused on and test it before attaching it to the final product.





### 3. Design Diagrams

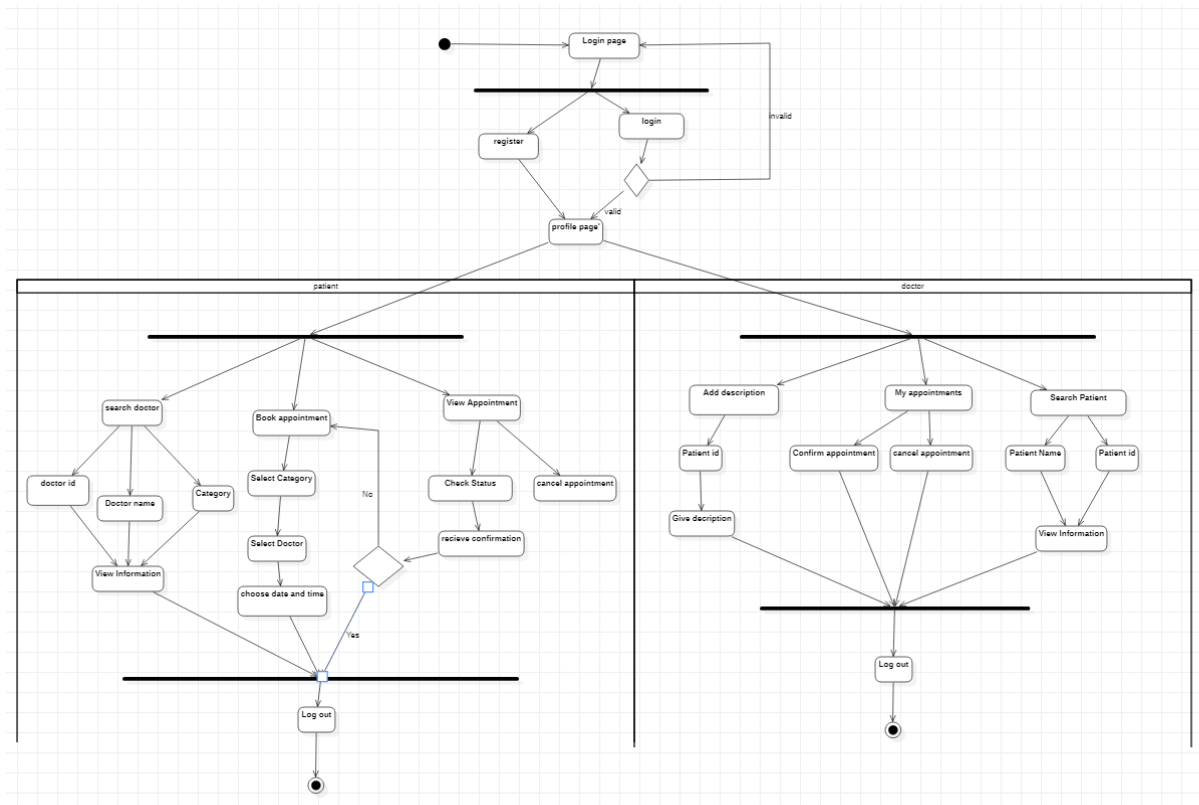
#### Architecture Diagram



- This is a Three-tier architecture that organizes applications into three logical and physical computing tiers:
  - The presentation tier/ user interface layer contains all of the classes responsible for presenting the UI to the end-user or sending the response back to the client
  - The application tier, where data is processed; contains all the logic that is required by the application to meet its functional requirements

- and the data tier, where the data associated with the application is stored and managed.

## Activity Diagram

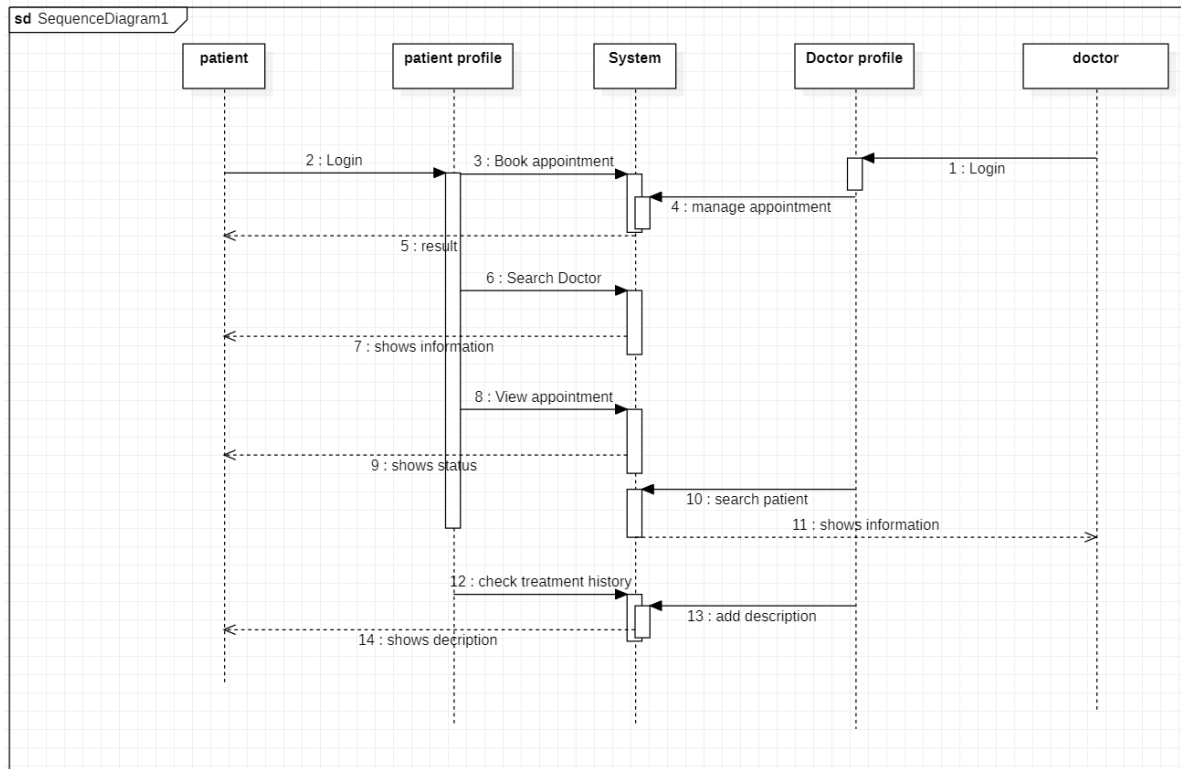


User logs in with credentials. The profile page is divided into 2 parts according to type of user. If the user is a patient then he can do concurrent actions like search doctor, Book appointment, View appointment. The patient can search the doctor with help of doctorid, doctor name and category to see the information. The patient can book appointment by selecting category, doctor pertaining to that and data and time.

After booking an appointment the patient can check the status of appointment whether it is cancelled or confirmed by the doctor. If the appointment was cancelled then book the appointment again. If the appointment was confirmed then go to the doctor.

If the user type is doctor then he can do concurrent actions like adding description by selecting patient id, managing appointments or searching patients by name and id. Doctor can manage his appointments by either cancelling or confirming appointments.

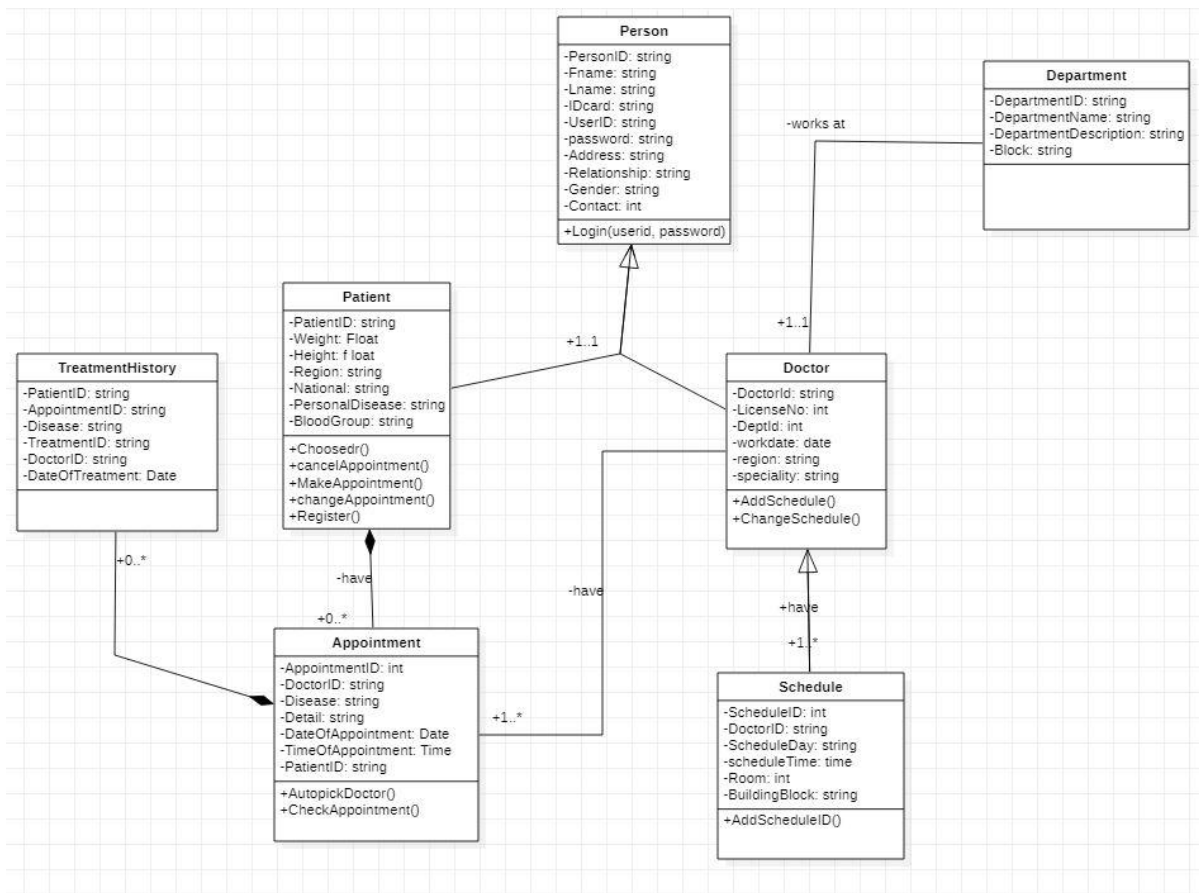
## Sequence Diagram



The sequence diagram contains participants/objects they are Patient , patient profile, System, Doctor profile, doctor. Each object is represented as a lifeline. Patients and doctors will login to their respective profiles. Message indicated by horizontal arrow to another object with message name. In the patient profile the patient can send a message as a book appointment to the system for booking appointments. The doctor can manage appointments by either cancelling or confirming it. The patient can also view the appointment and search doctors in the system.

The doctor can also search the patient and give a description to the patient which can be seen by the patient through check treatment history.

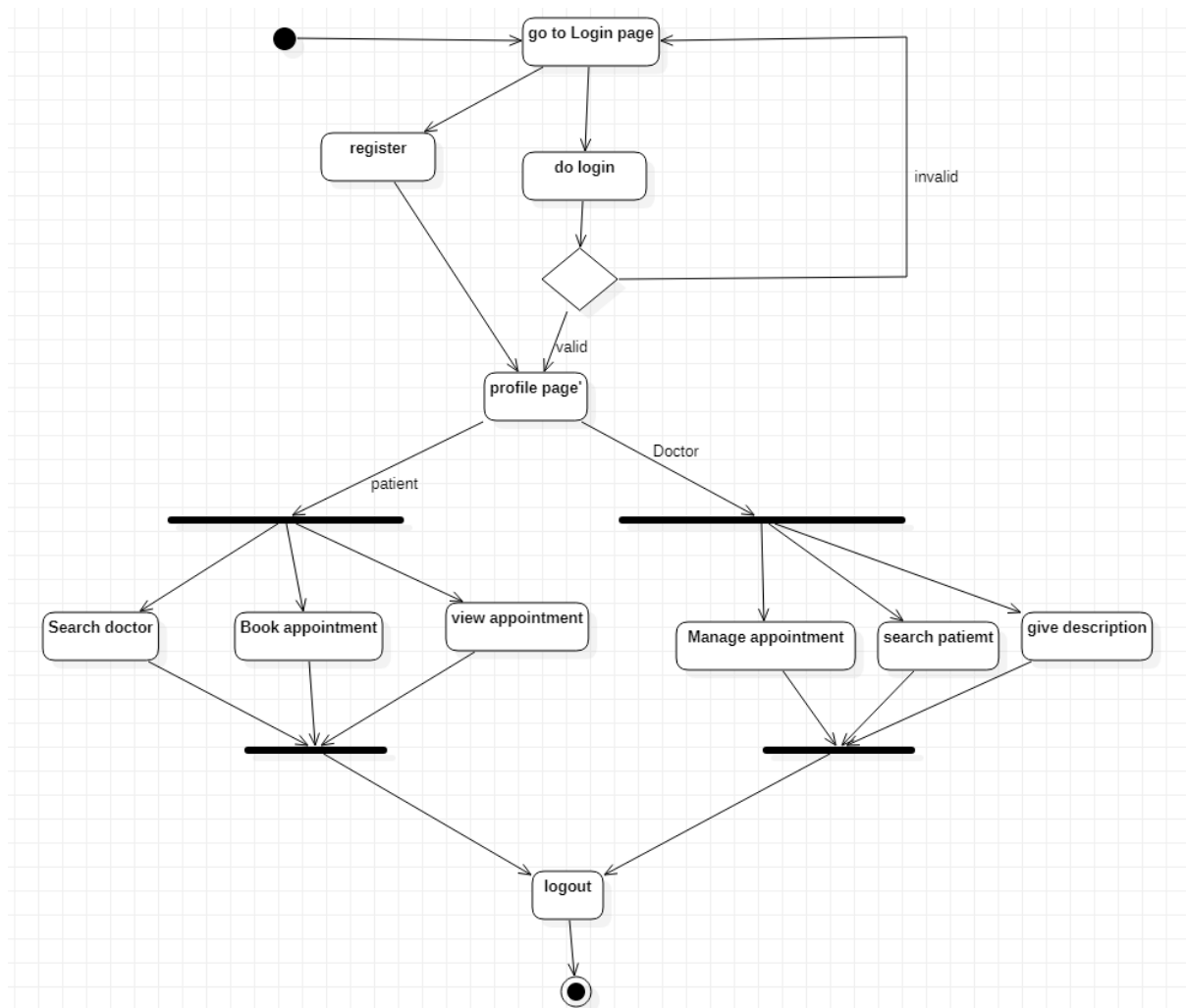
## Class Diagram



The **class diagram** is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application.

The class is represented with class name at the top, attributes in the middle, operations at the bottom. The various classes used are Person, Department, Patient, Treatment History, Schedule, Appointment, Doctor. The association and multiplicity rules used are self explanatory.

## State Diagram

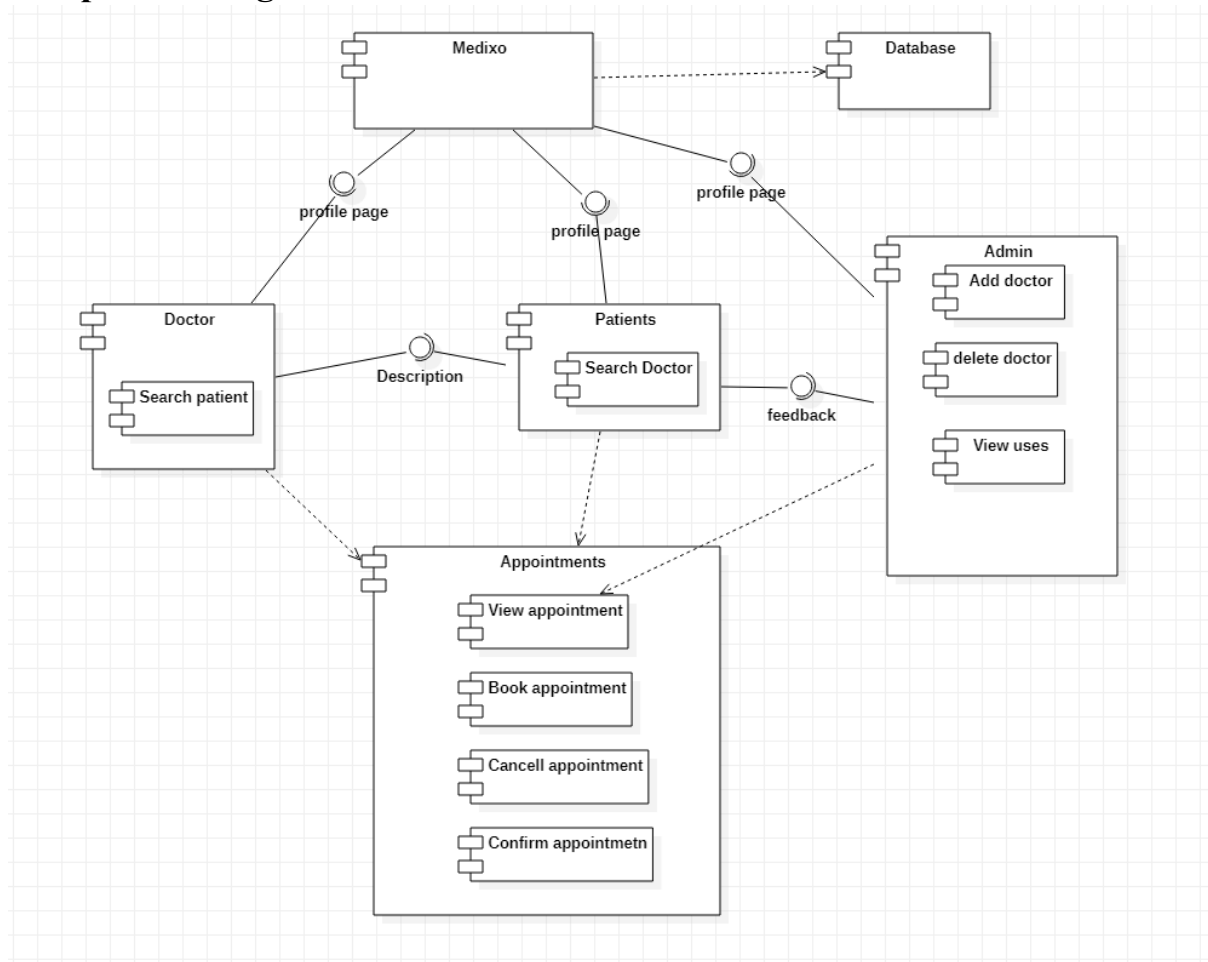


State diagram is representation of potential states of the objects and the transitions among the those states

The state of an object is a condition or a situation during the life of an object during which it satisfies some condition, performs some activity or waits for an event. When an event occurs some activity will take place based on the current

state of the object. The event is represented as an arrow which moves from one state to another.

## Component Diagram



The UML component diagram shows how a software system will be composed of a set of deployable components dynamic-link library files, executable files, or web services—that interact through well-defined interfaces and which have their internal details hidden. Components implementation details are hidden in the diagram. A component diagram shows the internal parts, connectors, and ports that implement a component.

## 4. Module description

### Patient

**Book appointment :** The patient will be able to book an appointment with any doctor based on category by selecting appropriate date and time.

**Cancel appointment :** The patient can cancel his appointment and can rebook the appointment.

**Search Doctor:** The patient can search the doctor based on doctor id , doctor name and doctor category to see the information .

**View appointment:** The patient can view all the appointments made by him to the specified doctor.He can also get to see the status of the appointment whether it is confirmed or cancelled.

**Treatment history:** The patient can see the treatment history which contains a description given by the doctor.

**Feedback:** The patient can give feedback to the doctor based on treatment given by the doctor.

### Doctor

**View appointments :** Doctors can view appointments requested by patients to them. Doctors can either confirm or cancel the appointments.

**View Patient's history :** Doctors can view the prescription he/she had given to any particular patient.

**Add prescription :** Doctor can add prescription to any particular patient using his/her ID.

### Admin

**Add doctor :** If a doctor wants to register, he will contact the admin who can add doctors by mentioning his/her name, an unique ID, address, specialization, email and phone number.

**Delete doctor :** Only the admin can delete existing doctors by using his/her ID.

**View appointments :** Admin can also view the appointments requested by the patients along with their status.



**View feedbacks :** Admin can view the feedback given by each patient.

## 5. Test Cases

### Management of Appointment

Test CaseID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-01	Booking appointment	To test the booking of appointment	Patients must be logged in and have access to browsers.	1.Log into patient's account 2.Navigate to book appointment page 3.Fill the required details 4.click "submit" button.	1.Dr-ID 2.Appointment-Id 3.Date 4.Time	Appointment data should be stored in DB and confirmation status is set to pending	Appointment data should be stored in DB.	Pass
UT-02	Viewing appointments on patient side	To test the patient's functionality by Viewing appointment	Patients must be logged in and have access to browsers.	1.Log into patient's account 2.Navigate to View appointment page	NIL	All the appointments scheduled by the patient if any.	All the appointments scheduled by the patient.	Pass
UT-03	Cancelling Appointment by patient	To test cancellation of appointment on patient side	Patients must be logged in and have a booking scheduled.	1.Log into patient's account 2.Navigate to View appointments page 3.Click on the 'x' button next to the appointment you want to cancel.	NIL	The appointment must be cancelled and be removed from the appointments displayed.	Appointment was cancelled and it was removed from the appointments list.	Pass
UT-04	Viewing Appointments on the Doctor's side	To test if the appointments have been stored in the database and visible on the doctors side.	Doctors must have access to browsers and must be logged in.	1.Log into the Doctor's account 2.Navigate to view appointments	NIL	All the appointments for the current doctor must be visible	All the appointments scheduled for the current doctor were displayed	Pass



UT-05	Cancelling Appointment by Doctor	To test cancellation of appointment on Doctor side	Patients must be logged in and have a booking schedule d.	1.Log into Doctor's account 2.Navigate to View appointments page 3.Click on the 'x' button next to the appointment you want to cancel.	NIL	The appointment must be cancelled and should show the message as cancelled	The appointment must be cancelled and should show the message as cancelled	Pass
UT-06	Confirmation of Appointments by the doctor	Doctor can confirm the appointment request according to his/her schedule	Doctor must be logged in and have appointment requests	1.Log into Doctor's account 2.Navigate to 'My appointments' page 3.Click on the ' button(  or  ) next to the appointment you want to approve or deny	NIL	The appointment must be confirmed and should show the message as confirmed	The appointment must be confirmed and should show the message as confirmed	Pass
UT-07	Reflection of appointments on the other side	To test whether the modification of data reflects in other side	Patient /Doctor must be logged in	1.Do task like book appointment, cancel ,confirm appointment etc .2. Log out from account 3. Check the reflection in other side	NIL	On clicking cancel or confirm button the change in data should reflect in patient side	On clicking cancel or confirm button the change in data should reflect in patient side	pass
UT-08	Updation of appointment status	To check whether the operations between patient and doctors are updated	Patients must be logged in and have a booking schedule	1.Log in with credentials 2.Go to view appointment page 3.check the status of appointment	NIL	On approve or deny button click, the status on patient side gets updated From pending to confirm or cancelled respectively	On approve or deny button click, the status on patient side gets updated From pending to confirm or cancelled respectively	pass

## Managing Users

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-01	View doctor/patient	To test admin functionality	To show the info ,patient and doctor should register	1.Go to view doctor/view patient page 2.See the details	NIL	Information in the database about patient/Doctor	It shows the information about patient/Doctor	Pass
UT-02	Doctor login	To test the login functionality of doctor	Doctor should register before login	1: Navigate to Doctor's login page	Doctor id password	Login should be successful ,going to profile page	Login was successful ,going to profile page	Pass

				2: Enter Doctor id and Password 3: Click login				
UT-03	View profile by doctor	To test whether doctor login is successful	Doctor should login before visiting profile	1: Navigate to <a href="http://www.medixo.com">http://www.medixo.com</a> 2. Login with credentials	Doctor id Name Address number Email id specialisation	Result should be the profile page of respective doctors	It was redirected to the profile page of respective doctors	Pass
UT-04	Give prescription by doctor	To test the interaction between doctor and patient	Doctor should know the patient id	1.Go to add prescription page 2. Give the patient id 3. Add description	Patient id Name of the patient	Prescription should be visible	Prescription was visible in treatment history section	Pass
UT-05	Log out by doctor	exit from profile page	Doctor should login before logout	1.Navigate to <a href="http://www.medixo.com">www.medixo.com</a> 2.Login with credentials 2.Log out	NIL	Should exit from the profile page	It successfully logged out	pass
UT-06	Search patient by doctor	To test whether the patient info is visible to doctor	Doctor should login and Patient details must be present	1.Go to search patient page 2.Search patient by patient id/patient name	Patient id Patient name	Show patient information	Patient information Was shown	pass
UT-07	Patient registration	To test the register functionality	Access to Chrome Browser	1: Navigate to the Login page and click on register. 2: Enter the details 3: Click Submit	User name: Password : Contact Address Email id Blood type	Login should be successful	Login was successful	pass
UT-08	Patient login	To test the login functionality	Patient should register before login	1: Navigate to Login page 2: Enter user id and Password 3: Click login	patient id password	Login should be successful ,going to profile page	Login was successful ,going to profile page	pass
UT-09	View profile by patient	To test Information retrieval for viewing profile of user.	Patient should login before visiting profile	1: Navigate to <a href="http://www.medixo.com">http://www.medixo.com</a> 2. Login with credentials	patient id Name Address number Email id Blood type	Result should be the profile page of respective patients	the profile page of respective patients	pass

UT-1 0	Patient Sending feedback	To test the interaction between patients and doctors	Patient login should be before sending feedback	1.Go to profile page 2.click on send feedback 3. Write feedback and click send	Patient id Patient name	Result should be visible to admin	Feedback by the patients was visible to admin	Pass
UT-1 1	Admin login	To test the login functionality	NIL	1: Navigate to <a href="http://www.medixo.com">http://www.medixo.com</a> 2: Enter admin id and Password 3: Click login	admin id password	Login should be successful ,going to profile page	Login was successful ,redirecting to profile page	pass
UT-1 2	Adding doctors by admin	To test admin functionality	Admin Should know doctors info	1.go to add doctor page 2. Enter the doctor info 3.click on add doctor	Doctor id - name -Address -password -category -email id	should result in adding information in doctor table in database	should result in adding information in doctor table in database	pass
UT-1 3	Deleting Doctor by admin	To test admin functionality	Admin should know doctor id	1.go to delete doctor page 2. Enter the doctor id 3.click on delete doctor	Doctor id	should result in deleting information in doctor table in database	Deleting information in doctor table in database	pass

## Recommend Doctors

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-01	Search Doctor based on ID	The user inputs the doctor ID and clicks the search button which shows the doctor details with that dr ID	Patient must be logged in and enter a valid DR ID	1.Log into patient's account 2.Navigate to search doctors page 3.Enter the Doctor ID 4.click submit	1.Dr-ID	The Doctors information with the entered Dr ID should be retrieved,if doctor with that ID does not exist a message should be displayed to the user	Information of the Doctor is retrieved for appropriate doctor Id and for inappropriate ID error message is shown	Pass

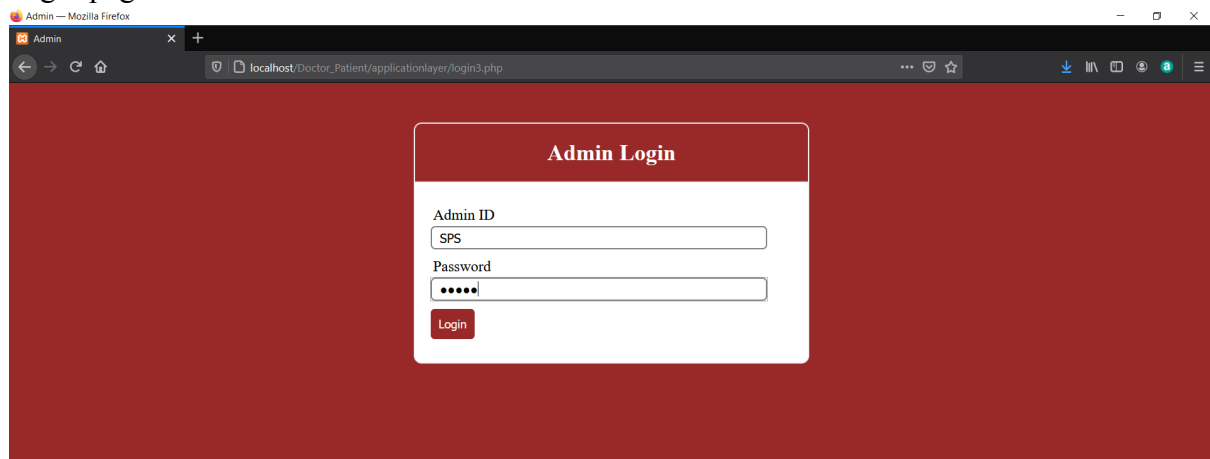
UT-02	Search Doctor based on Name	The user should inputs the doctor name and clicks the search button which shows the doctor details with that Name	Patient must be logged in and enter a Doctor Name which needs to be searched	1.Log into patient's account 2.Navigate to search doctors page 3.Enter the Doctor Name 4.click submit	1.Doctor Name	The Doctor details with the entered Dr name should be retrieved,if doctor with that name does not exist a message should be displayed to the user	Information of the Doctor is retrieved for appropriate doctor name and for inappropriate name error message is shown	Pass
UT-03	Search Doctor based on department /category	The user enters the department name and clicks the search button which shows the doctor details belonging to that department	Patient must be logged in and enter a Department which needs to be searched	1.Log into patient's account 2.Navigate to search doctors page 3.Enter the Department 4.click submit	1.Department	The Doctor details with the entered Dr name should be retrieved,if doctor with that name does not exist a message should be displayed to the user	Information of the Doctor is retrieved for appropriate doctor name and for inappropriate department error message is shown	Pass
UT-04	Fetching list of appropriate Doctors	To test the functionality of search doctor	Doctor must register with account and patient must search doctor based on id,name and category	1.Log into a patient's account. 2.Navigate to search doctors page 3.Search doctors 4.click submit	Doctors id,Doctor name, Doctor category	The query should fetch the appropriate doctors from the database	The query fetched the appropriate doctors from the database	pass
UT-05	Displaying the list of appropriate doctors while searching	To test the functionality of search doctor	Doctor must register with account and patient must search doctor based on id,name and category	1.Log into a patient's account. 2.Navigate to search doctors page 3.Search doctors 4.click submit	Doctors id,Doctor name, Doctor category	The query should display the appropriate doctors to patient	The query displayed the appropriate doctors from the database	pass

UT-06	Recommend the doctor name based on category while booking appointment	To help the user to know doctor name while booking	Doctor must register with account and patient must book appointment based on category	1.Log into a patient's account. 2.Navigate to book appointment page 3.Search category to book appointment 4.click submit 5.Check whether doctor is available	Doctor name, Doctor category	The query will display the appropriate doctors to patient	The query should display the appropriate doctors to patient	pass
UT-07	Redirection to search doctor page	On the button click the website should redirect to the search doctor page	Patient must be logged in and click on search doctor	1.Log into patient's account 2.Navigate to search doctors page	NIL	The action should successfully redirect to search doctor page	On button click the website was redirected to search doctor page	pass
UT-08	Recommend the doctor id based on category while booking appointment	To help the user while booking if 2 doctors have same name	Doctor must register with account and patient must book appointment based on category	1.Log into a patient's account. 2.Navigate to book appointment page 3.Search category to book appointment 4.click submit 5.Check whether doctor is available	Doctor id, Doctor category	The query will display the appropriate doctors to patient	The query should display the appropriate doctors to patient	pass
UT-09	Recommend the multiple doctors along with name and id based on category while booking appointment	To help the user to choose doctor while booking	Doctor must register with account and patient must book appointment based on category	1.Log into a patient's account. 2.Navigate to book appointment page 3.Search category to book appointment 4.click submit 5.Check whether doctor is available	Doctor id, Doctor category	The query will display the list of appropriate doctors to patient	The query should display the list of appropriate doctors to patient	pass

## 6. Screenshots of Outputs

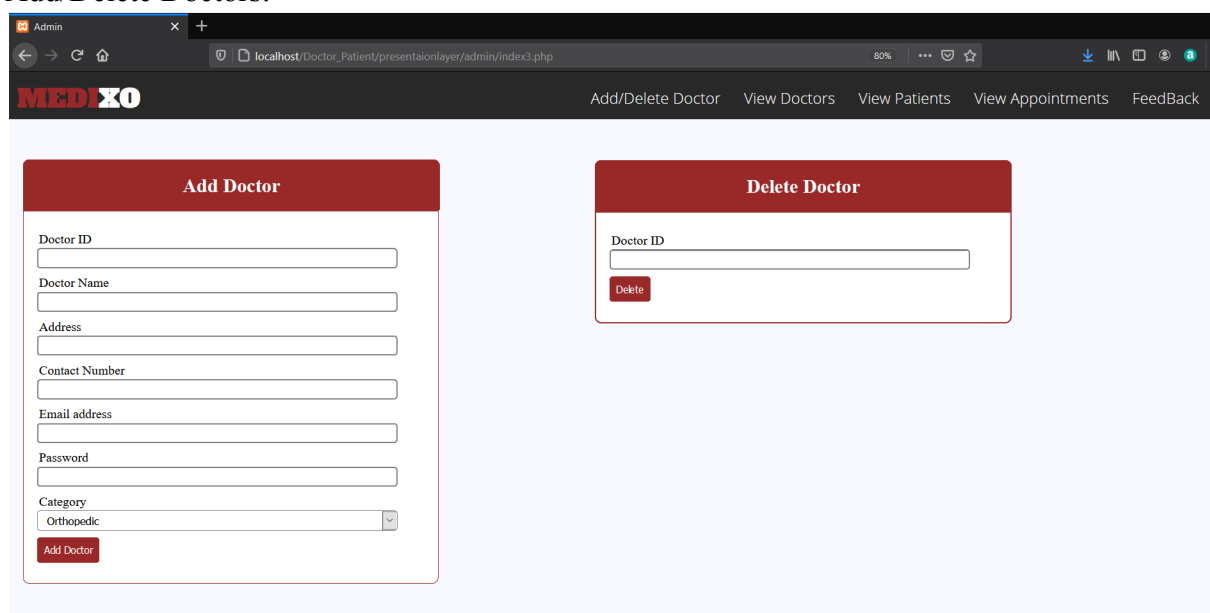
### Admin

Login page:



A screenshot of a web browser window showing the Admin Login page. The browser's address bar displays 'localhost/Doctor\_Patient/applicationlayer/login3.php'. The page has a dark red background. In the center, there is a white rectangular box with a red header titled 'Admin Login'. Inside this box, there are two input fields: 'Admin ID' with the text 'SPS' and 'Password' with masked characters '•••••'. Below these fields is a red 'Login' button.

Add/Delete Doctors:



A screenshot of a web browser window showing the 'Add/Delete Doctors' page. The browser's address bar displays 'localhost/Doctor\_Patient/presentationlayer/admin/index3.php'. The page has a dark red header with the 'MEDIXO' logo on the left and navigation links 'Add/Delete Doctor', 'View Doctors', 'View Patients', 'View Appointments', and 'FeedBack' on the right. The main content area has a light blue background and contains two white rectangular boxes with red headers. The left box is titled 'Add Doctor' and contains several input fields: 'Doctor ID', 'Doctor Name', 'Address', 'Contact Number', 'Email address', 'Password', and a 'Category' dropdown menu currently set to 'Orthopedic'. At the bottom of this box is a red 'Add Doctor' button. The right box is titled 'Delete Doctor' and contains a single 'Doctor ID' input field and a red 'Delete' button.

## View Doctors:

MEDI <sup>XO</sup>						
Add/Delete Doctor   View Doctors   View Patients   View Appointments   FeedBack   Logout						
Doctors Information						
Doctor ID	Doctor Name	Email	Address	Contact Number	Category	
1	Supreeth Ronad	suppi@gmail.com	Pes University	9148489627	Orthopedic	
2	Gourav	gjpujari@gmail.com	PES	9876543210	Cardiologist	
3	Anusha	anusha08@gmail.com	MS Ramaiha	9696878745	Genral Surgeon	
4	Praveen	pavi@gmail.com	Manipal	8855888558	Dentist	
5	Naveen	navi@gmail.com	SJC	7676367979	Orthopedic	
6	Raghuveer	raghu@gmail.com	People tree	9620000033	Dentist	
7	vishal	vishal@gmail.com	SJIT	9845300700	Genral Surgeon	
8	viyaan	viyaan@gmail.com	PES	8787878787	Genral Surgeon	

## View Patients:

MEDI <sup>XO</sup>						
Add/Delete Doctor   View Doctors   View Patients   View Appointments   FeedBack   Logout						
Patients Information						
Patient ID	Patient Name	Address	Contact Number	Email	Blood Group	
1	Prathik	Pes University	9148489627	prathikbafna0@gmail.com	b-	
2	sandeep	pes	9383838383	qbhejkd@gmail.com	a+	
5	sandy	PES	1234567890	sandy@gmail.com	a+	
10	abc	abc	1234567890	abc@gmail.com	o+	
18	qwerty	qwerty	1234567890	qwerty	b-	
19	Shashank	xyz	8888881234	shashi@gmail.com	b-	

## View Appointments:

MEDI <sup>XO</sup>						
Add/Delete Doctor   View Doctors   View Patients   View Appointments   FeedBack   Logout						
Appointments						
Appointments ID	Doctor ID	Patient ID	Date	Time	Status	
12	1	1	2022-05-03	22:45:00	Confirmed	
17	1	10	2022-05-04	23:11:00	Cancelled	
21	1	1	2021-04-27	23:30:00	Cancelled	
23	1	1	2022-05-10	00:00:00	Cancelled	
24	1	1	2021-04-26	13:25:00	Confirmed	
25	1	1	2021-11-21	12:30:00	Confirmed	
28	2	2	2021-05-01	10:00:00	Cancelled	
42	3	1	2020-10-17	01:51:00	Confirmed	
43	8	1	2022-05-21	20:30:00	Confirmed	
46	1	1	2050-05-20	20:00:00	Cancelled	

## View Feedback:

MEDI <sup>XO</sup>			<a href="#">Add/Delete Doctor</a>	<a href="#">View Doctors</a>	<a href="#">View Patients</a>	<a href="#">View Appointments</a>	<a href="#">FeedBack</a>	<a href="#">Logout</a>
Patients FeedBack								
Patient ID	Patient Name	FeedBack						
1	Prathik	DR Abc is good						
1	Prathik	Dr viyaan is good						
1	Prathik	qq						

## Patient

### Patient Login:

**Patient Login**

User ID  
1

Password  
\*\*\*\*\*

[Login](#)

Not a member? [Sign Up](#)

### Profile page:

**MEDI<sup>XO</sup>**

[Profile](#) [Book Appointment](#) [View Appointment](#) [Search Doctor](#) [Logout](#)

**My Information**

**Prathik #1**  
prathikbafna0@gmail.com | 9148489627

**Pes University**

Blood Type :  
**b-**

[My Treatment History](#)

[Send Feedback](#)

### Book Appointment:



Patient x +

localhost/Doctor\_Patient/presentationlayer/patient/book.php 80%

**MEDIXO** Profile Book Appointment View Appointment Search Doctor Logout

### Book Appointment

**Category**

Orthopedic

Search

**Doctor Name**

Praveen #4

**Date**

14 / 04 / 2021

**Time**

11:45 \_am

BOOK

## View Appointments:

Patient x +

localhost/Doctor\_Patient/presentationlayer/patient/view.php 80%

**MEDIXO** Profile Book Appointment View Appointment Search Doctor Logout

### MyAppointment

Appt ID	DATE	TIME	Doctor Name	Contact	Category	Confirmation
12	2022-05-03	22:45	Supreeth Ronad	9148489627	Orthopedic	Confirmed
21	2021-04-27	23:30	Supreeth Ronad	9148489627	Orthopedic	Cancelled
23	2022-05-10	00:00	Supreeth Ronad	9148489627	Orthopedic	Cancelled
24	2021-04-26	13:25	Supreeth Ronad	9148489627	Orthopedic	Confirmed
25	2021-11-21	12:30	Supreeth Ronad	9148489627	Orthopedic	Confirmed
42	2020-10-17	01:51	Anusha	9696878745	Genral Surgeon	Confirmed
43	2022-05-21	20:30	viyaan	8787878787	Genral Surgeon	Confirmed
46	2050-05-20	20:00	Supreeth Ronad	9148489627	Orthopedic	Cancelled

## Search/recommend Doctors:

Patient x +

localhost/Doctor\_Patient/presentationlayer/patient/searchdoctor.php 80%

**MEDIXO** MyInfo Book Appointment View Appointment Logout

Search By:

- Doctor ID
- Doctor Name
- Category

Search

Doctor ID	Doctor Name	Address	Contact Number	Category
7	vishal	SSIT	9845300700	Genral Surgeon

# Doctor

## Doctor Login:

Doctor Login

Doctor ID  
1

Password  
\*\*\*\*\*

Login

## Profile Page:

MED XO

Profile My Appointments Search Patient Add Description Logout

My Information

Supreeth Ronad #1  
9148489627

Pes University

Specialization :  
**Orthopedic**

## My appointments:

Appt ID	DATE	TIME	PatientID	Name	Address	Email	Contact	Cancel
12	2022-05-03	22:45	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Confirmed
17	2022-05-04	23:11	10	abc	abc	abc@gmail.com	1234567890	Cancelled
21	2021-04-27	23:30	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Cancelled
23	2022-05-10	00:00	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Cancelled
24	2021-04-26	13:25	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Confirmed
25	2021-11-21	12:30	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Confirmed
46	2050-05-20	20:00	1	Prathik	Pes University	prathikbafna0@gmail.com	9148489627	Cancelled

## Search Patient:

The screenshot shows a web browser window with the URL `localhost/Doctor_Patient/presentationlayer/doctor/searchpatient.php`. The page features a navigation bar with the 'MEDIXO' logo and links for 'Profile', 'My Appointments', 'Search Patient', 'Add Description', and 'Logout'. The main content area is divided into two sections. The top section, titled 'Search By:', contains a form with two input fields: '\*Patient ID' and '\*Patient Name', and a 'Search' button. The bottom section, titled 'Patient Information', displays a table with the following data:

PatientID	Name	Address	Contact Number	Email	BloodGroup
1	Prathik	Pes University	9148489627	prathikbafna0@gmail.com	b-

Below the 'Patient Information' table is another section titled 'Treatment History', which displays a table with the following data:

PatientID	PatientName	Treatment	Doctor's Note
1	Prathik	covid-19	Re-check up after 2 weeks
1	Prathik	cccc	qqqqqqqqqqqqqqqqqq
1	Prathik	qqqqqqqqweretyu	hxfogjvmhj,.

## Write Prescription:

The screenshot shows a web browser window with the URL `localhost/Doctor_Patient/presentationlayer/doctor/add.php`. The page features a navigation bar with the 'MEDIXO' logo and links for 'Profile', 'My Appointments', 'Search Patient', 'Add Prescription', and 'Logout'. The main content area contains a form for writing a prescription. The form includes the following fields and buttons:

- PatientID**: A text input field with a 'Search' button below it.
- Patient ID**: A text input field containing the value '2'.
- Name**: A text input field containing the value 'sandeep'.
- Treatment**: A text input field containing the value 'covid-19'.
- Note**: A text input field containing the value 'Isolate for 21 days and regular O2 check'.
- Add**: A button at the bottom of the form.