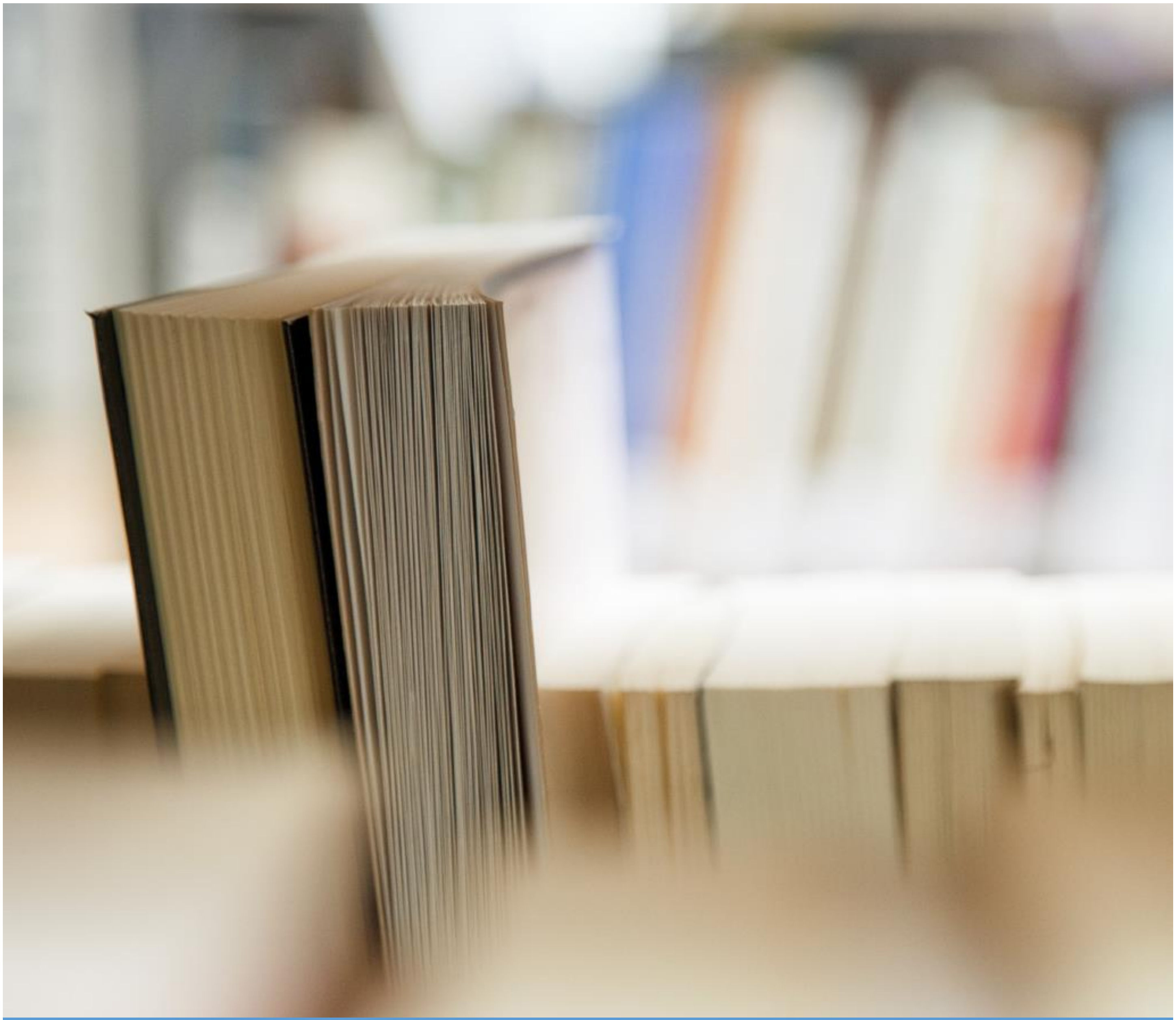


Smart Health

shuvo 10/29/17 Software Project Lab-3

Software Requirement Specification, Analysis, Design and Test Plan

Smart Health Project

Software project lab-3

Supervisor

Nadia Nahar
Lecturer
Institute of Information Technology
University of Dhaka

Submitted by

Shubho Chandro Roy
BSSE-0619

Submission Date

21-12-2017

Institute of Information Technology
University of Dhaka

Letter of Transmittal

BSSE 4th Year Program Committee
Institute of Information Technology,
University of Dhaka
December 14, 2017

Sir,

I have prepared the enclosed report with Software Requirements Specification, Architectural Design, Component Design, MOC Design and Test Plan of Smart Health Project. This report details the requirements I gathered for the project.

The purpose of this document is creating SRS, design report and testing report for the project I am doing for my Software Project Lab-III. This report includes the details of each steps I followed to collect the requirements.

Sincerely Yours,
Shubho Chandro Roy (BSSE0619)
4th Year, 8th Semester, 6th Batch
Institute of Information Technology
University of Dhaka

Document Authentication

This project document has been approved by the following persons.

Prepared by

Shubho Chandro Roy
BSSE0619

Approved by

Nadia Nahar
Lecturer, IIT
University of Dhaka

Acknowledgement

At first I am expressing my gratitude to almighty GOD.

I would like to thanks Nadia Nahar mam, Lecturer, Institute of Information Technology (IIT), University of Dhaka. She has been helped me a lot from the start of this project and till now. For that, I have completed the Smart Health project. I am expressing my thankfulness to her.

I would like to thanks BSSE 4th year program committee for accepting the idea of this project and have given me an opportunity to do this kind of project as my SPL-III project. I am expressing my appreciativeness to them.

Executive Summary

Smart Health project is a web based system, where patient can easily find their expected doctor. They can take service without any cost and contact with their expected doctor via messaging and video calling. Patients are also able to take recommendation hospital name according their symptoms.

Table of Contents

Chapter 1
Introduction

1.1 Purpose

This document is the Software Requirements Specification (SRS) for Smart Health project. It contains detailed functional, non-functional, and support requirements and establishes a requirements baseline for development of the system. The requirements contained in the SRS are independent, uniquely numbered, and organized by topic. The SRS serves as the official means of communicating user requirements to the developer and provides a common reference point for both the developer team and stakeholder community.

1.2 Intended Audience

This SRS is intended for several audiences including the customer, as well as the project managers, designers, development and testers.

- The User will use this SRS to verify that the developer team has created a system that is acceptable to the user.
- I will use this SRS as a basis for developing the system’s functionality. I will link the requirements defined in this SRS to ensure that I have created software that will fulfill all of the user’s documented requirements.
- The testers will use this SRS to derive test plans and test cases for each documented requirement. When portions of the software are being completed, the testers will run their tests on the software to ensure that the software fulfills the requirements documented on the SRS. The testers will again run their tests on the entire system when it is completed and more use of that all requirements documented in the SRS have been fulfilled.

Chapter 2
Inception

2.1 Introduction

Inception is the beginning phase of requirements engineering. It defines how does a software project get started and how is the scope and nature of the problem to be solved. The goal of the inception phase is to identify concurrence needs and conflict requirements among the stakeholders of a software project. To establish the groundwork I have worked with the Factors related to the inception phases:

- Identifying Stakeholders
- Asking the First Questions
- Recognizing Multiple Viewpoints
- Working Towards Collaboration

2.1.1 Identifying Stakeholders

Stakeholder refers to any person or group who will be affected by the system directly or indirectly. Stakeholders include end-users who interact with the system and everyone else in an organization that may be affected by its installation. To identify the stakeholders, I find some question:

- Who is paying for the project?
- Who will be using the project outcomes?
- Who gets to make the decisions about the project (if this is different from the money source)?
- Who has resources I need to get the project done?
- Whose work will my project affect? (During the project and also once the project is completed)

Stakeholder

1. Doctor: Doctor can login into the system via email id and password. He/she can provide service to the patient according the patients' problem.
2. Patient: Patient can login into the system via email id and password. He/she can find doctor, contact with them through messaging and if necessary
3. Supervisor: He/she plays a vital role for this project. Every steps and features of this project has been verified by his/her.
4. Developer: A group of developer or a single person may develop this system.
5. Tester: A group of tester or a single person may test this project.

2.1.2 Asking the First Questions

I set my first set of context-free questions focusing on the users and other stakeholders, overall project goals and benefits. The questions are mentioned above. This questions helped me to identify all stakeholders' measurable benefit of the successful implementation and possible alternatives to custom software development. Next set of question helped us to gain a better understanding of problem and allows the customer to voice his or her perception about the solution. The final set of question focused on the effectiveness of the communication activity itself.

2.1.3 Recognizing Multiple Viewpoints

I collect this viewpoint by discussing with the supervisor, doctors, patients, medical center and Teachers of Institute of Information Technology, University of Dhaka. Further I also collect some viewpoints from some non-registered people as guest.

1. Doctors: Create own profile and give services to patients.
2. Patients:
    - Contact with doctors via messaging.
    - Contact with doctors via video calling.
    - Get recommended hospital name nearby his/her location according to his/her disease.

2.1.4 Working towards Collaboration:

Every stakeholder has their own requirements. I have followed following steps to merge these requirements:

- Identify the common and conflicting requirements
- Categorize the requirements
- Take priority points for each requirement from stakeholders and on the basis of this voting prioritize the requirements
- Make final decision about the requirements

2.1.5 Conclusion

Inception phase helped me to establish basic understanding about Smart Health project for doctor and patient and identify the people who want to give and take treatment via online, define the nature of the Smart Health project and establish a preliminary communication with our stakeholders.

Chapter 3
Elicitation

3.1 Introduction

Elicitation is a task that helps the traveler to define what is required. To complete the elicitation step I face many problems like problems of scope, problems of volatility and problems of understanding. However, this is not an easy task. To help overcome these problems, I have worked with the Eliciting requirements activity in an organized and systematic manner.

3.2 Eliciting Requirements

Unlike inception where Q&A (Question and Answer) approach is used, elicitation makes use of a requirements elicitation format that combines the elements of problem solving, elaboration, negotiation, and specification. It requires the cooperation of a group of end-users and developers to elicit requirements. To elicit requirements, I completed following four works.

- Collaborative Requirements Gathering
- Quality Function Deployment
- Usage Scenarios
- Elicitation work products

3.3 Collaborative Requirements Gathering

Many different approaches to collaborative requirements gathering have been proposed. Each makes use of a slightly different scenario.

3.4 Quality Function Deployment

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software .It concentrates on maximizing customer satisfaction from the Software engineering process .With respect to my project the following requirements are identified by a QFD.

3.4.1 Normal Requirements

Normal requirements consist of objectives and goals that are stated during the meeting with the customers. Normal requirements of my project are:

- Accessible via the Internet.

- Allow interface to check for valid access and decision for valid system administrator and user.
- Doctor create his/her own profile for that patient can easily find his/her expected doctor.
- Patient sends message to the doctors
- Patient can direct communicate with doctors via video calling.

3.4.2 Expected Requirements

- Restrict access to functionality of the system based upon roles.
- Allow valid system administrator to login and logout to the system.

3.4.3 Exciting Requirements

- Get recommended medical center nearby patients' location and disease and other information.

3.5 Usage Scenario

Smart Health is an online platform for patient where they can easily find their expected doctor and contact with them through messenger or video calling. At first, every user must be registered into the system. For registration, they submit their first and last name, username, email id, password and select user role. Based on the user role some additional fields are appeared. Such as, for doctor, designation, college/university name, hospital name, specialty and for patient, age, height, weight, blood pressure, blood group and address. According the user role, user get a profile which they can update.

After login, patient can get more option such as messaging, video calling and recommendation. According to patients' problem, they can find their expected doctor and contact with doctor through messaging and video calling. If the doctor is unable to understand patients' problem in messaging session, they make video call for direct communication. Patients also able to get recommended hospital name nearby their location and according their given information which they give in authentication session.

Chapter 4
Scenario-Based Model

4.1 Introduction

In this model the system is described from the user's point of view. As this is the first model, it serves as input for creation of other modeling elements. Basically scenario based modelling evolves basic use case, use case diagrams (activity and Swim lane diagram).

4.2 Use Case Scenario

In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language as an actor) and a system to achieve a goal. The actor can be a human or other external system. The first step in writing a use case is to define that set of "actors" that will be involved in the story. Actors are the different people that use the system or product within the context of the function and behavior that is to be described. Actors represent the roles that people play as the system operators. Every user has one or more goals when using system.

Primary Actor: Primary actors interact directly to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software. Here doctors and patients are primary actor.

Secondary Actor: Secondary actors support the system so that primary actors can do their work. They either produce or consume information.

4.3 Use Case Diagram

Use case diagrams give the non-technical view of overall system.

4.3.1 System Description

After analyzing the user scenario, I found two actors who will directly communicate through the system as a system operator. I shall elaborate use case scenario to use case diagram, description, activity diagram & swim lane diagram. Here is the use case diagram of level-0 for Smart Health in Figure-1:

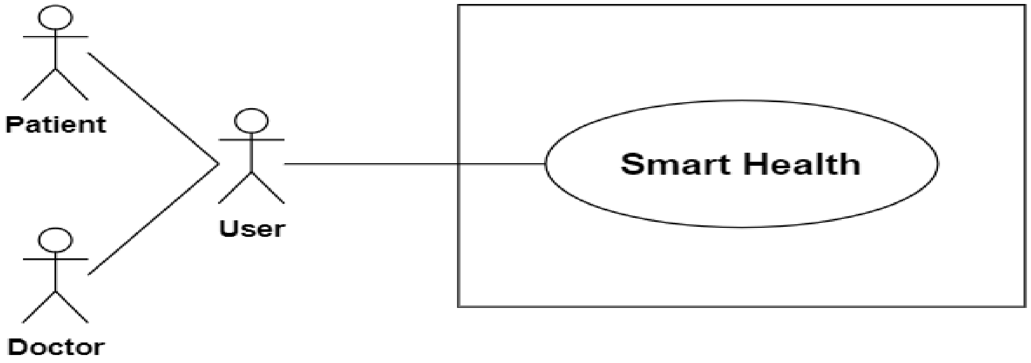


Figure 1: Use case diagram for level-0

4.3.2 Use Case Diagram for Smart Health level-1 Description

Primary actor : User
Secondary actor : System
Goal in context : To operate the application
Scenario : The actors of this system have to play different actions and system will reply according to these actions-
Action 1 : Enter Registration form.
Reply 1 : Please fill up the required filled.
Action 2 : Enter the information.
Reply 2 : Registration successful.
Action 3 : Enters username and password.
Reply 3 : Successfully login and shows more option for interact with system.
Action 4 : Click on profile.
Reply 4 : Shows user profile.
Action 5 : Enter information for update user profile
Reply 5 : User profile updated.
Action 6 : Click on Find Doctor
Reply 6 : Shows active doctor list
Action 7 : Click on message
Reply 7 : Shows chat box for messaging
Action 8 : Click on video call
Reply 8 : Video calling window appear for video conference
Action 9 : Click on recommendation
Reply 9 : Recommendation page appeared
Action 10 : Enter disease name for recommendation and search
Reply 10 : Recommended hospital name

Exception : No Exceptions (if system works correctly by fulfilling
requirements)
Exceptions (if) : User is not authenticated.
: System error

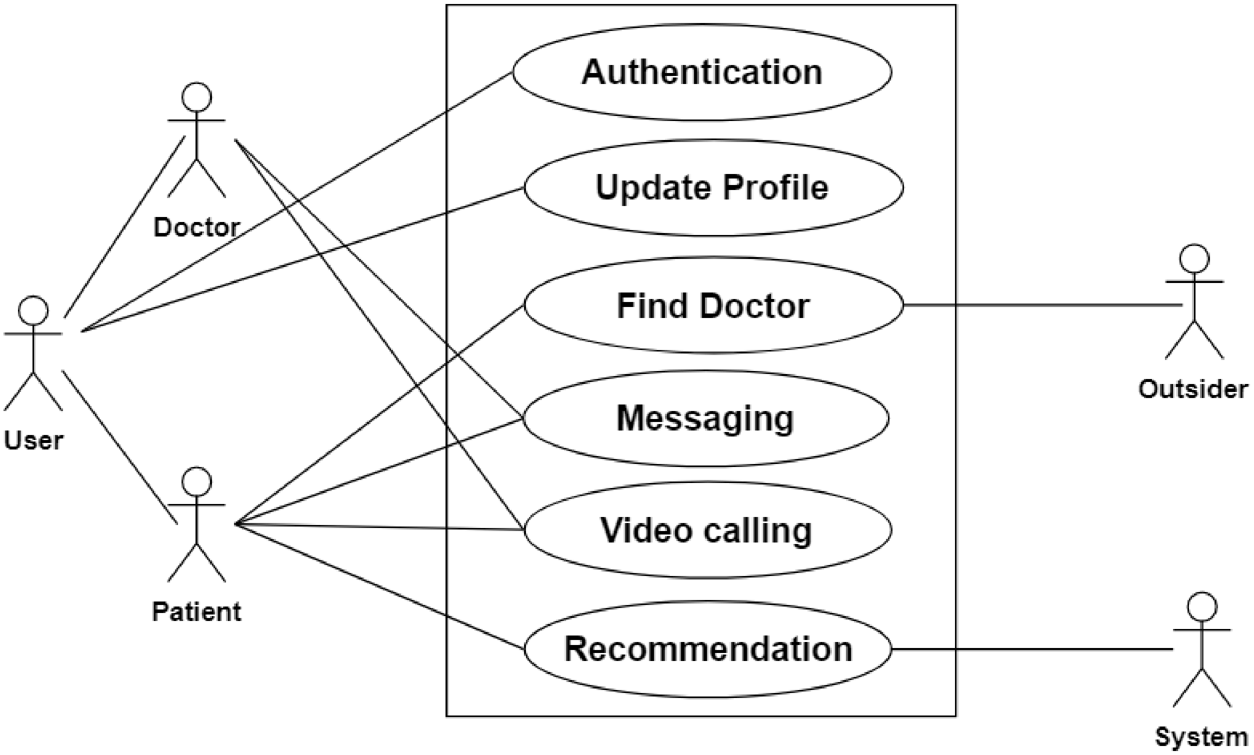


Figure 2: Use case diagram for level-1

4.3.2.1 Use Case for Authentication

I define authentication system into four subsystem. The use case for authentication is given below:-

Action 1	:	Enter required information
Reply 1	:	Successfully registered.
Action 2	:	Enters Email id and password.
Reply 2	:	Successfully logged in.
Action 3	:	Enter information for update profile.
Reply 3	:	Profile is updated.
Action 5	:	Click on log out
Reply 4	:	Logged out.

Exception: No Exceptions

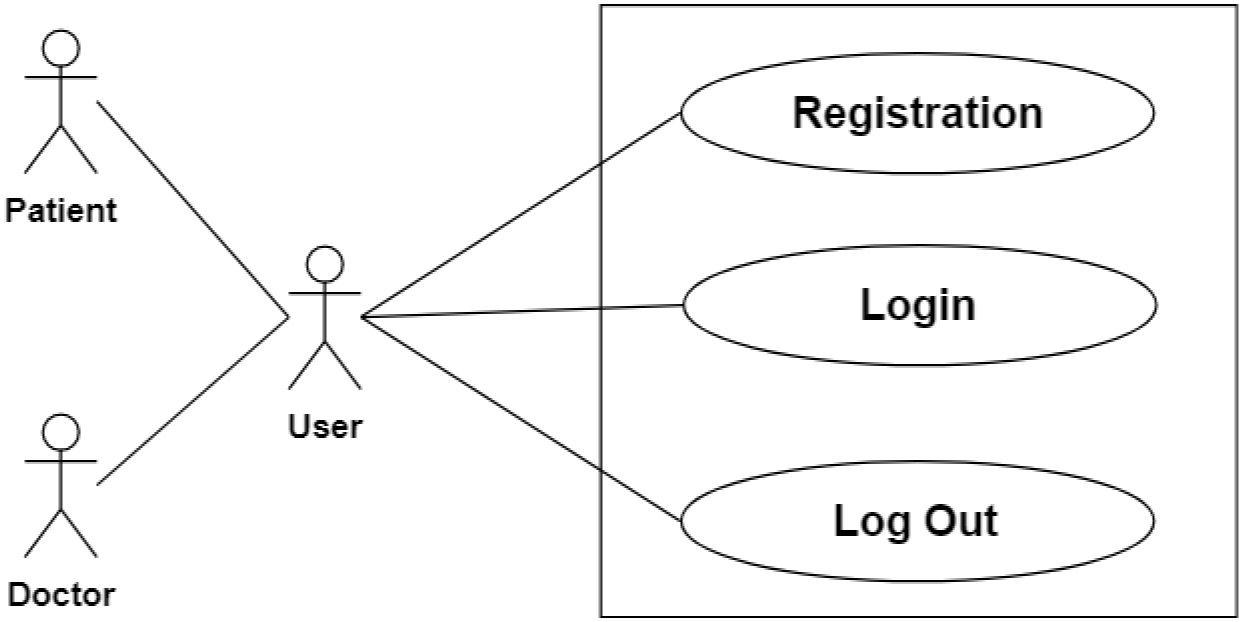


Figure 3: Use case for authentication level-1.1

4.3.2.2 Use Case for Registration

Use case: Registration

Primary Actor	:	User (Doctor, Patient)
Secondary Actor	:	System
Goal in Context	:	To enter the system for User.
Priority	:	Essential, must be implemented.

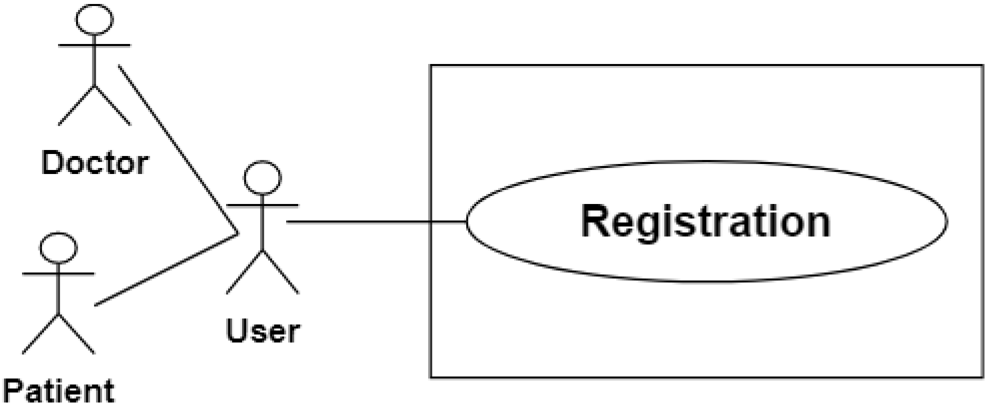


Figure 4: Use case for registration level-1.1.1

Activity diagram of Registration In is given below in figure in Figure-5

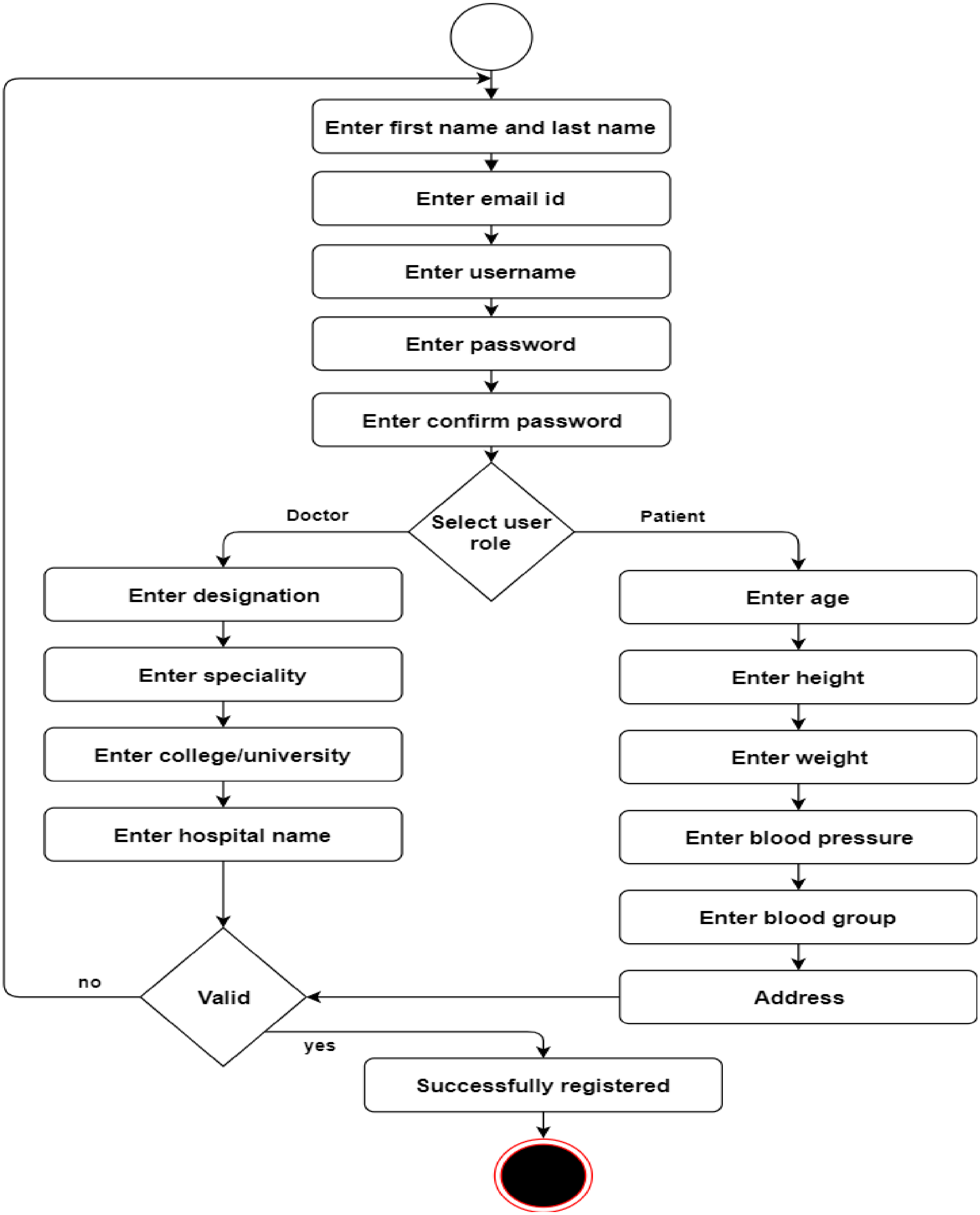


Figure 5: Activity diagram for registration

Activity Diagram (Registration): Doctor and patient must be completed registration activity for getting more option to perform. Without registration, a patient is not able to contact with doctor and cannot use the recommendation activity. If a doctor do not want to registered, he/she are not allowed to provide any online activity. So for registration, user must enter first name, last name, email id, username, password, confirm password and additional fields.

Swim Lane diagram of registration is given below in Figure-6

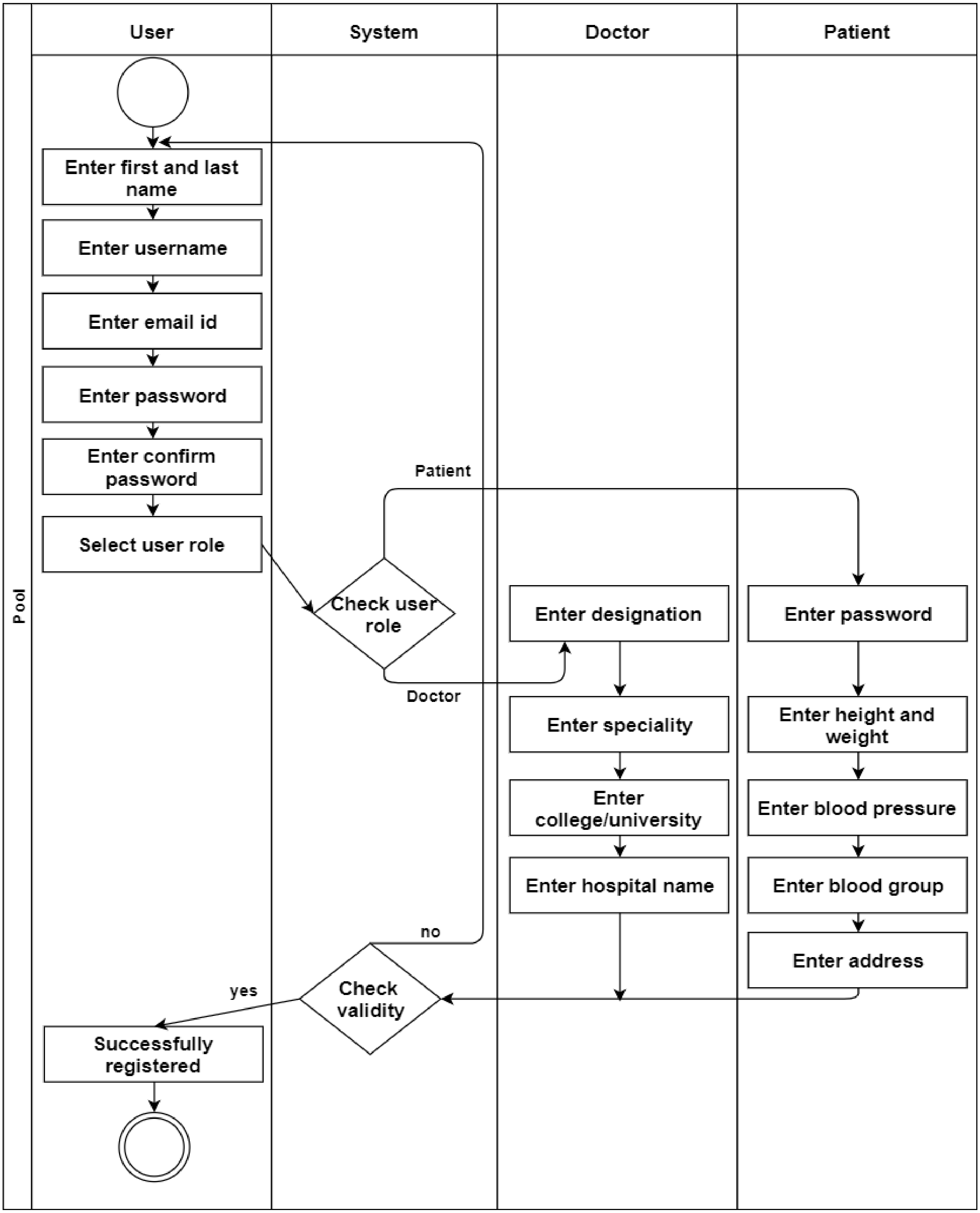


Figure 6: Swim lane diagram for registration

4.3.2.2 Use Case for Login

Use case: Login

Primary Actor	:	User (Doctor, patient)
Secondary Actor	:	System
Goal in Context	:	To entry the system.
Priority	:	Essential, must be implemented.

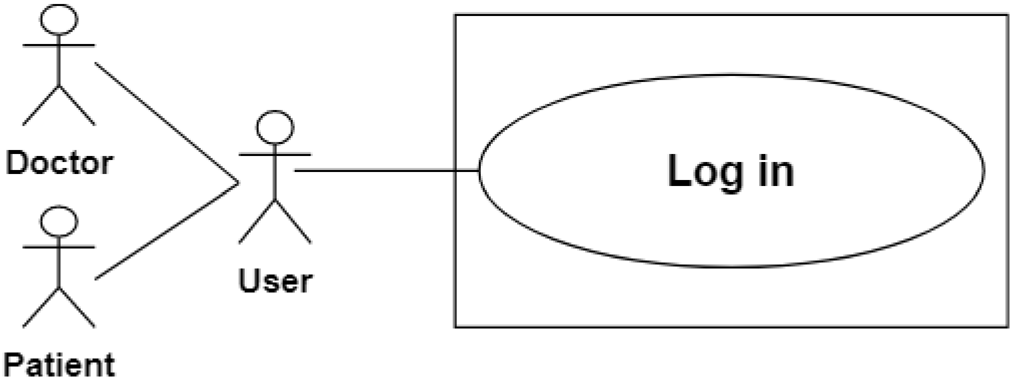


Figure 7: Use case for login level-1.1.2

Activity diagram of Log in is given below in figure in Figure-8

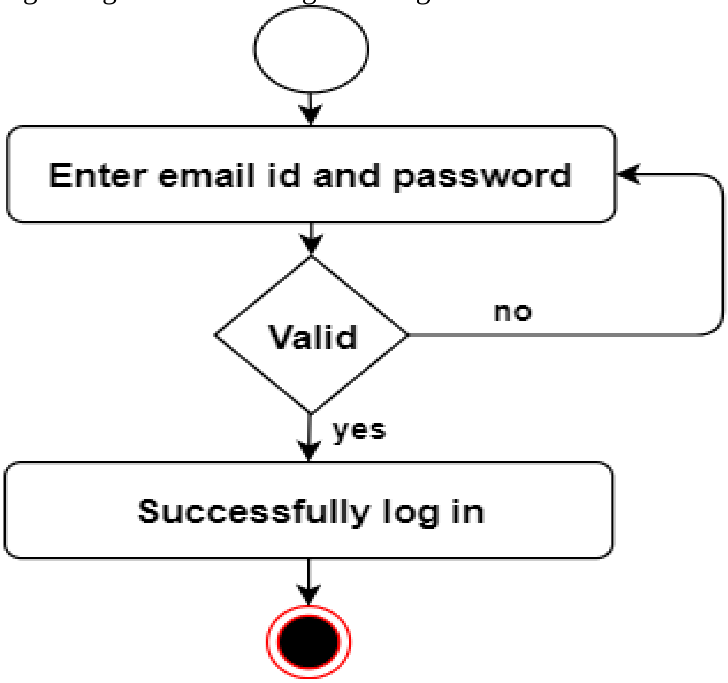


Figure 8: Activity diagram for login

Activity Diagram (Log in): For completing Log in activity, user need to provide valid email id and password

Swim lane diagram for login is given below in figure-9

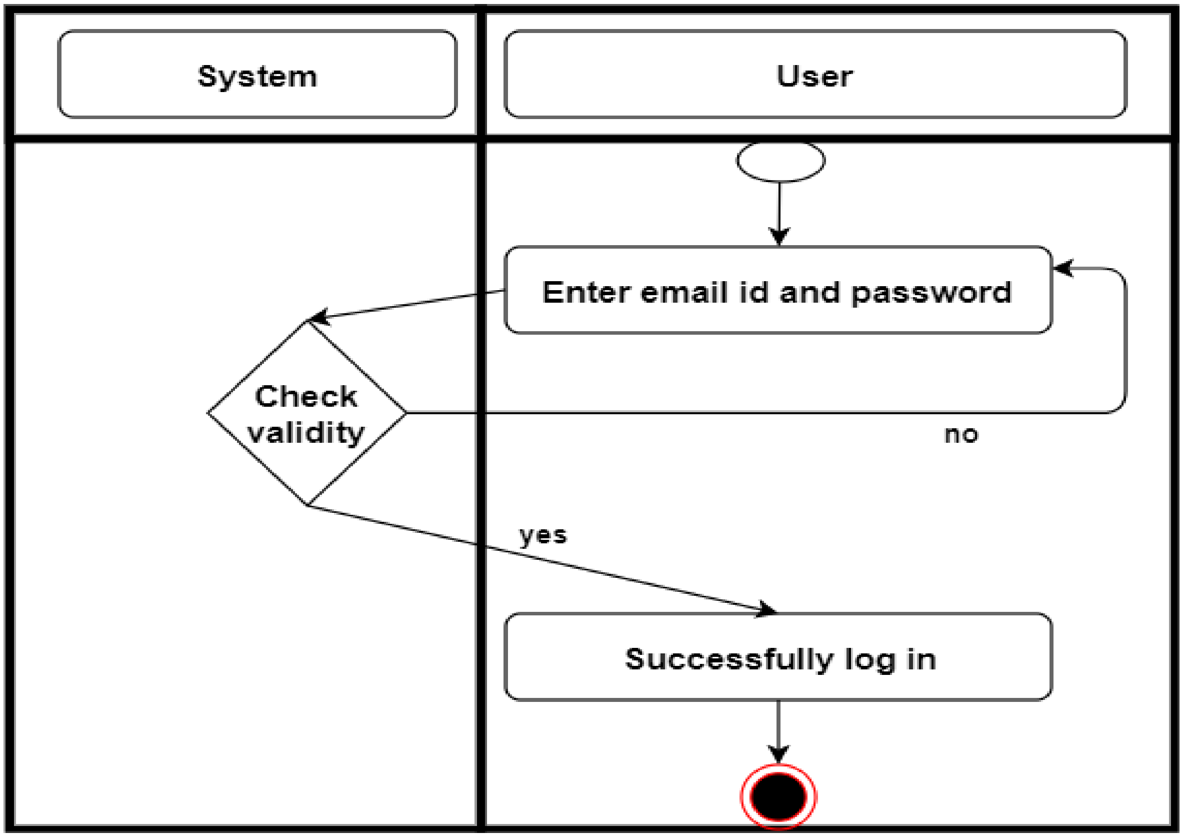


Figure 9: Swim lane diagram for login

4.3.3 Use Case for Update Profile

Use case: Update profile

Primary Actor	:	User (Doctor, Patient)
Secondary Actor	:	System
Goal in Context	:	To entry the system.
Priority	:	Essential, must be implemented.

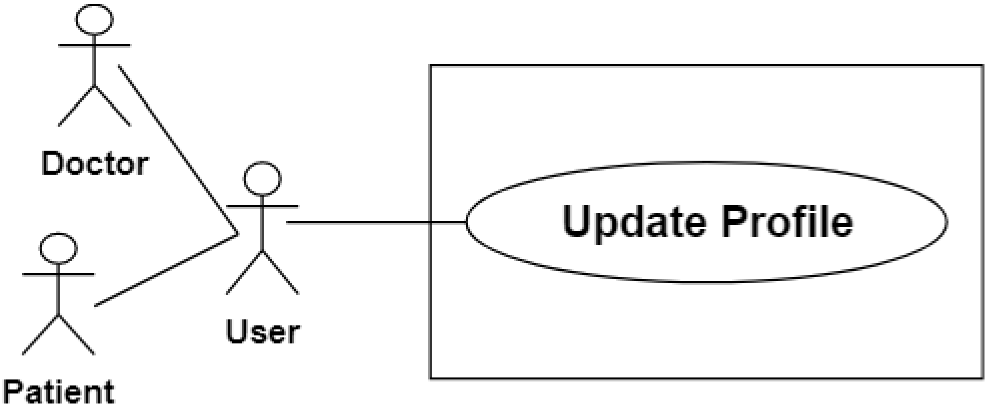


Figure 10: Use case for update profile level-1.1.3

Activity diagram of update profile is given below in figure-11

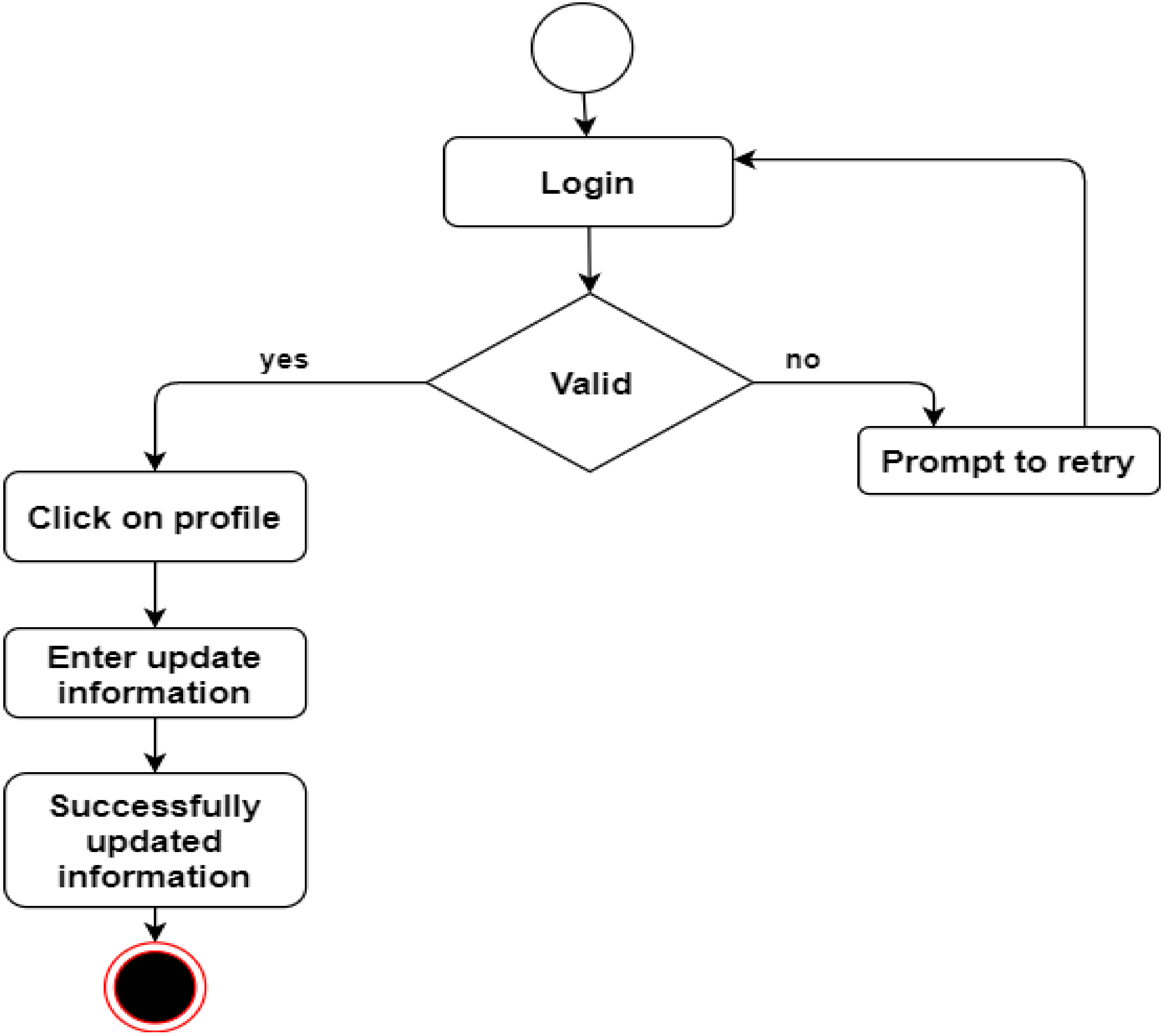


Figure 11: Activity diagram for update profile

Activity diagram (Update profile): User can update their profile. For updating their profile they must be logged into the system. After login they can update their profile information like password change, name change or anything they want to update.

Swim lame for update profile is given in figure 12

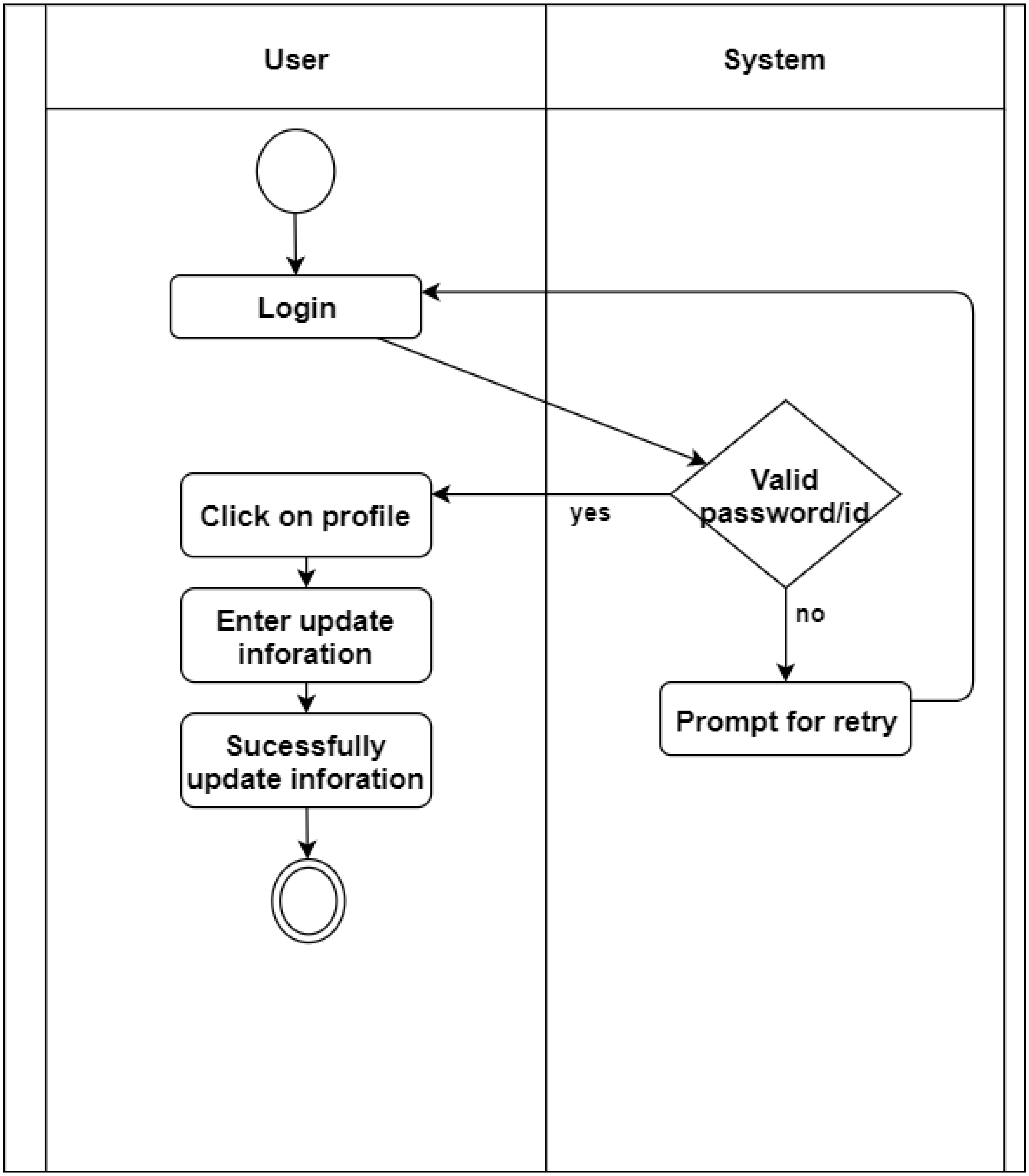


Figure 12: Swim lane diagram for update profile

4.3.4 Use Case for Find Doctor

Use case: Find Doctor

Primary Actor	:	Patient
Secondary Actor	:	System
Goal in Context	:	To find doctor list.
Priority	:	Essential, must be implemented.

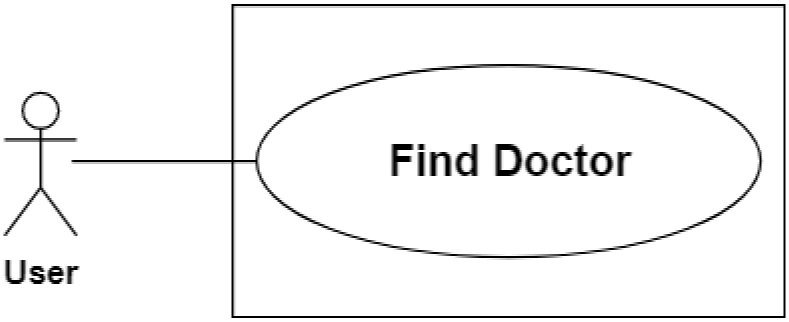


Figure 13: Use case for find doctor

Activity diagram for find doctor is given below in figure-14

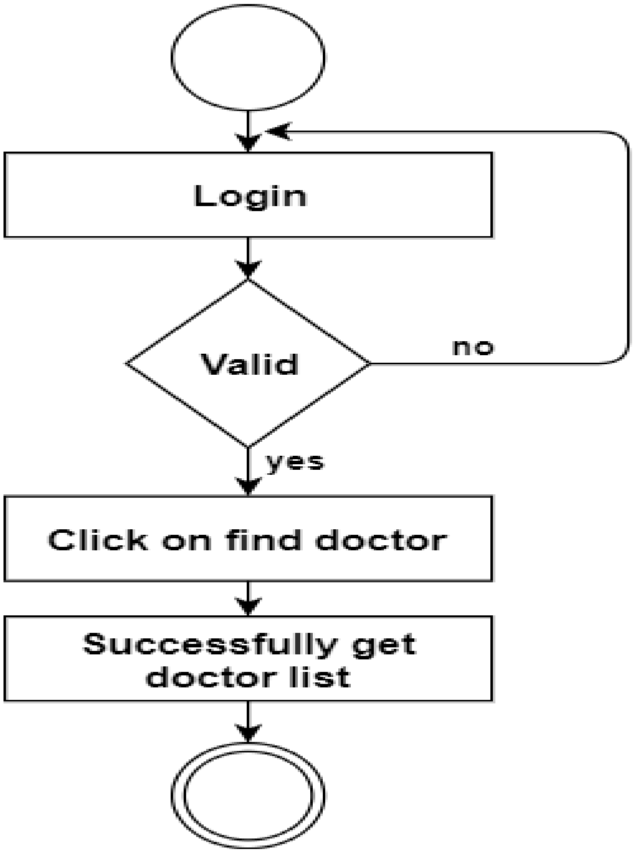


Figure 14: Activity diagram for find doctor

Activity diagram (Find Doctor): Patient firstly login into the system and then click on find doctor for getting doctor list.

Swim lane diagram for find doctor is given in figure-15

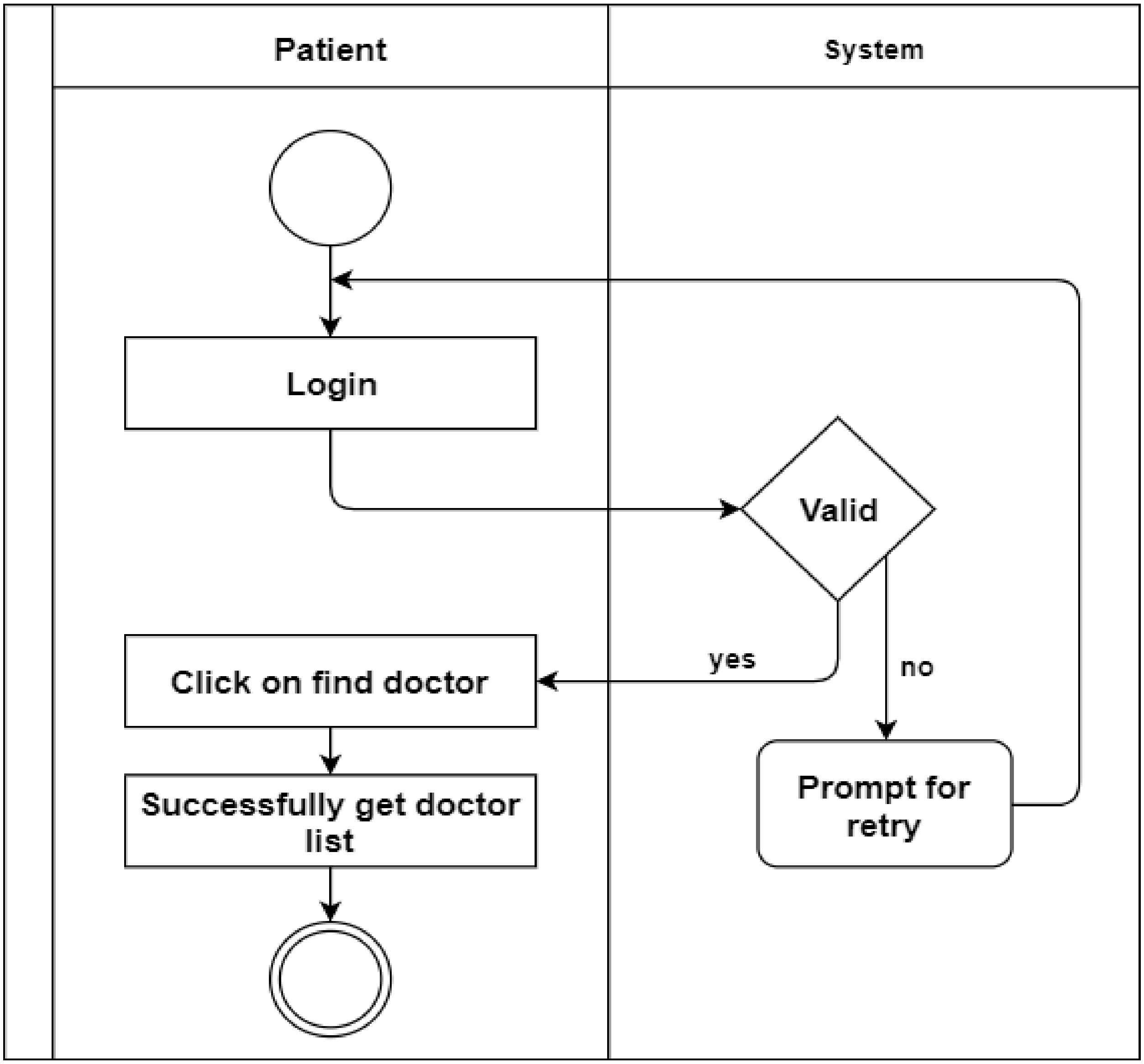


Figure 15: Swim lane diagram for find doctor

4.3.5 Use Case for Messaging

Use case: Messaging

Primary Actor	:	Patient, Doctor
Secondary Actor	:	System
Goal in Context	:	For communication between doctor and patient.
Priority	:	Essential, must be implemented.

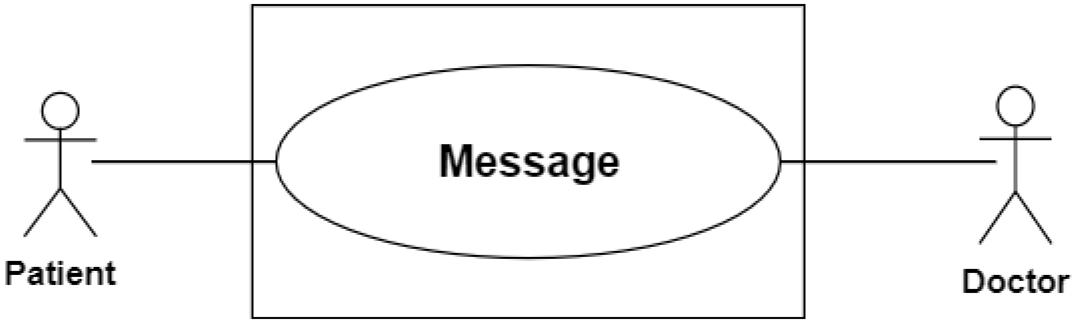


Figure 16: Use case for message

Activity diagram for message is given in figure-17

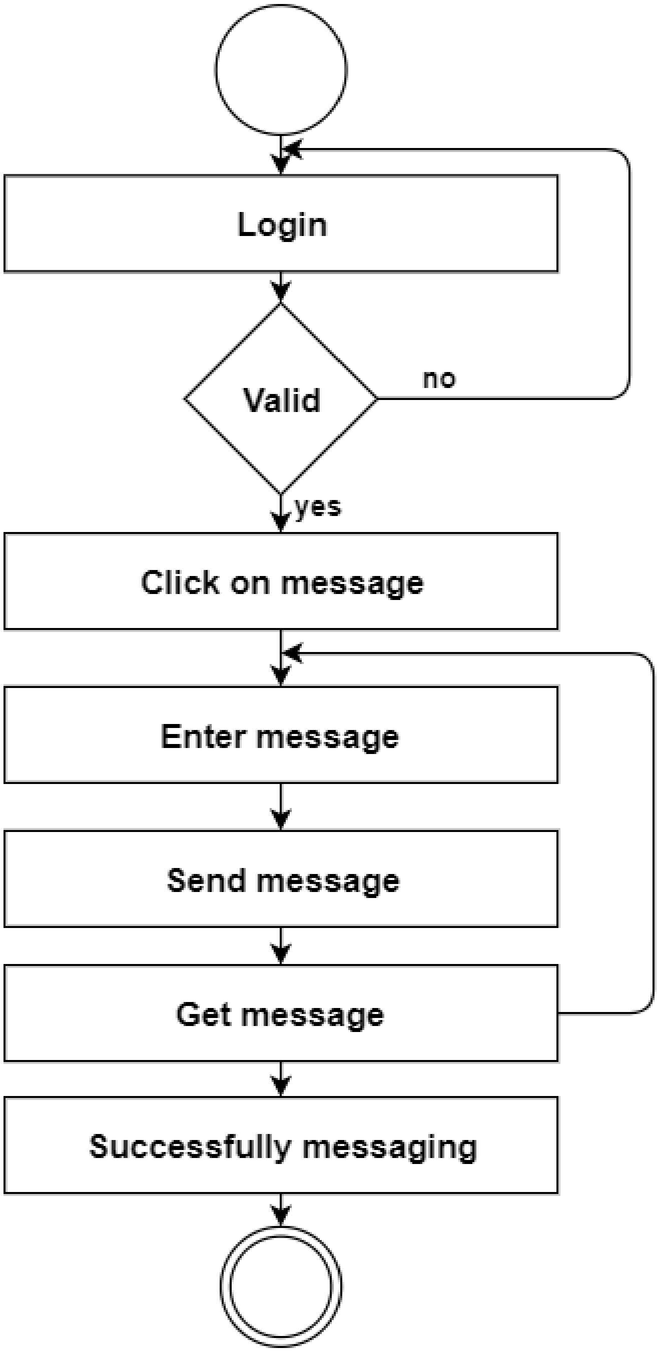


Figure 17: Activity diagram for message

Activity diagram (Messaging): Primarily patient can communicate with a doctor via messaging system.

Swim lane diagram for Messaging is in figure-18

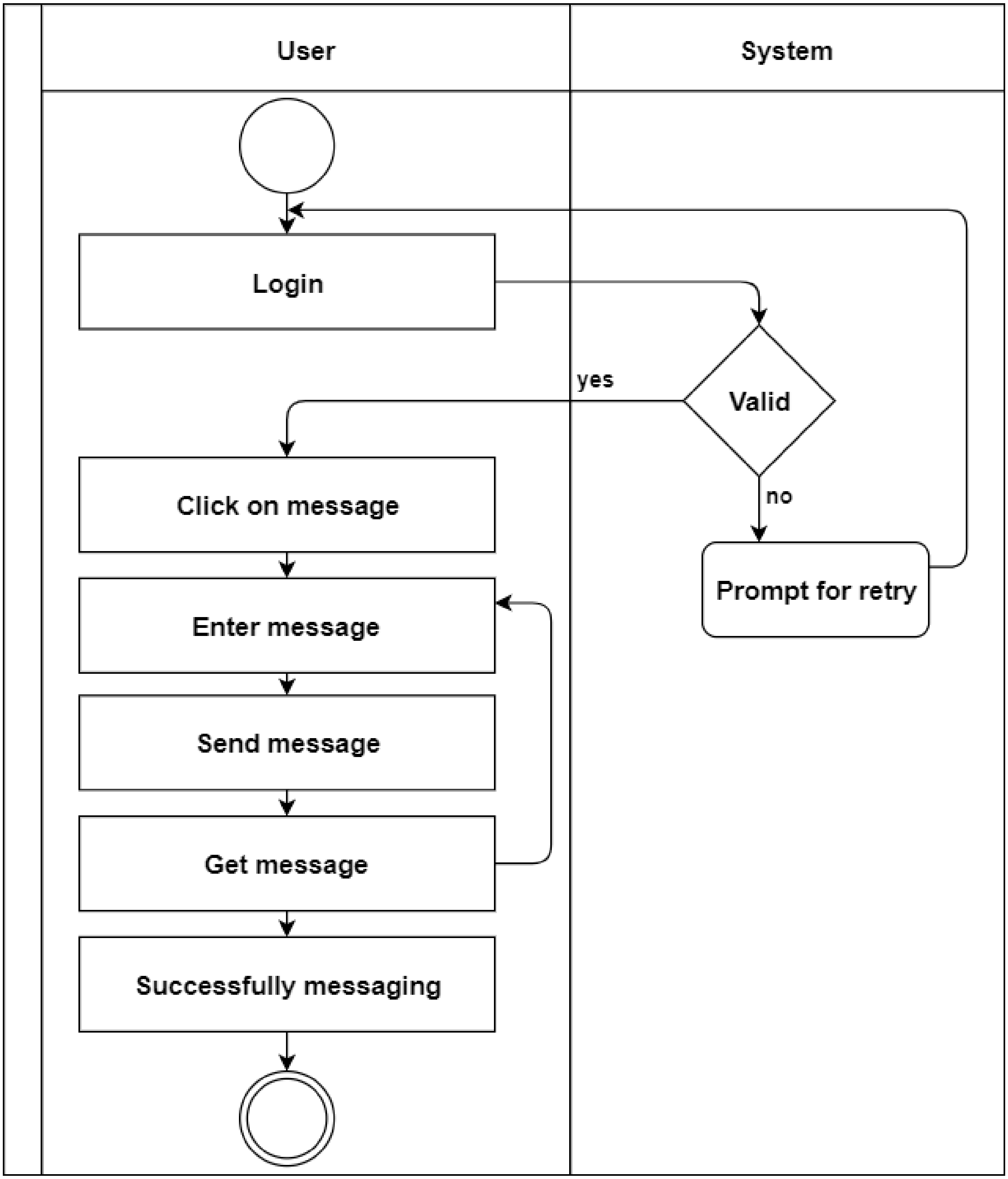


Figure 18: Swim lane diagram for messaging

4.3.6 Use Case for Video Calling

Use case: Video calling

Primary Actor	:	Patient, Doctor
Secondary Actor	:	System
Goal in Context	:	For communication between doctor and patient.
Priority	:	Essential, must be implemented.

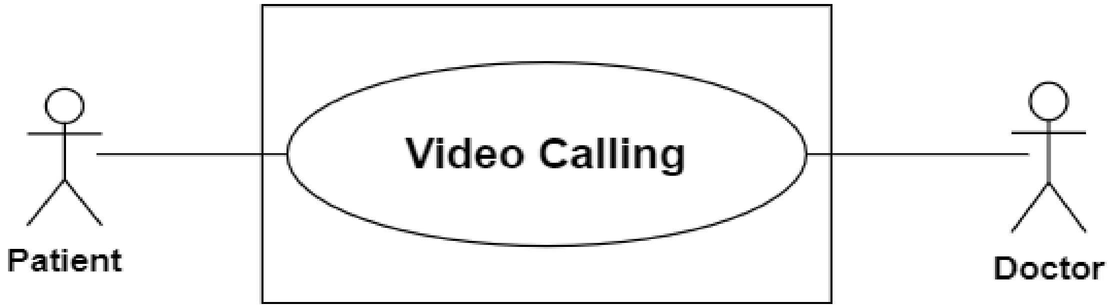


Figure 19: Use case for video calling

Activity diagram for video calling is given in figure-20

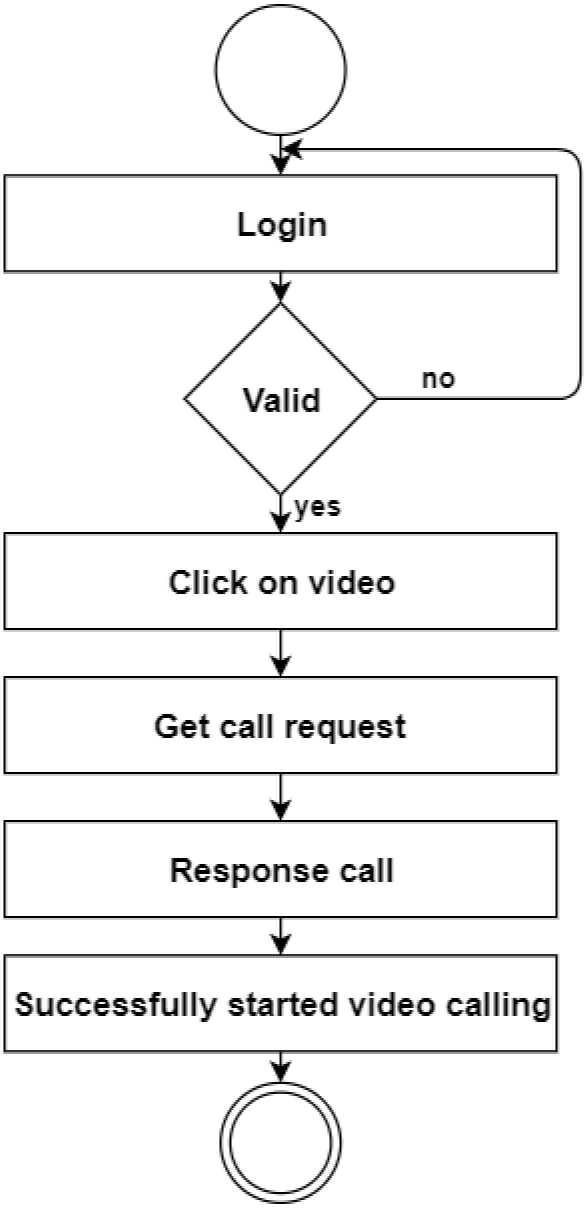


Figure 20: Activity diagram for video calling

Activity diagram (Video calling): Primarily patient can communicate with a doctor via video calling. Here doctor see the patient and suggest proper medicine. For video calling, patient enter his/her username.

Swim lane diagram for video calling is given in figure-21

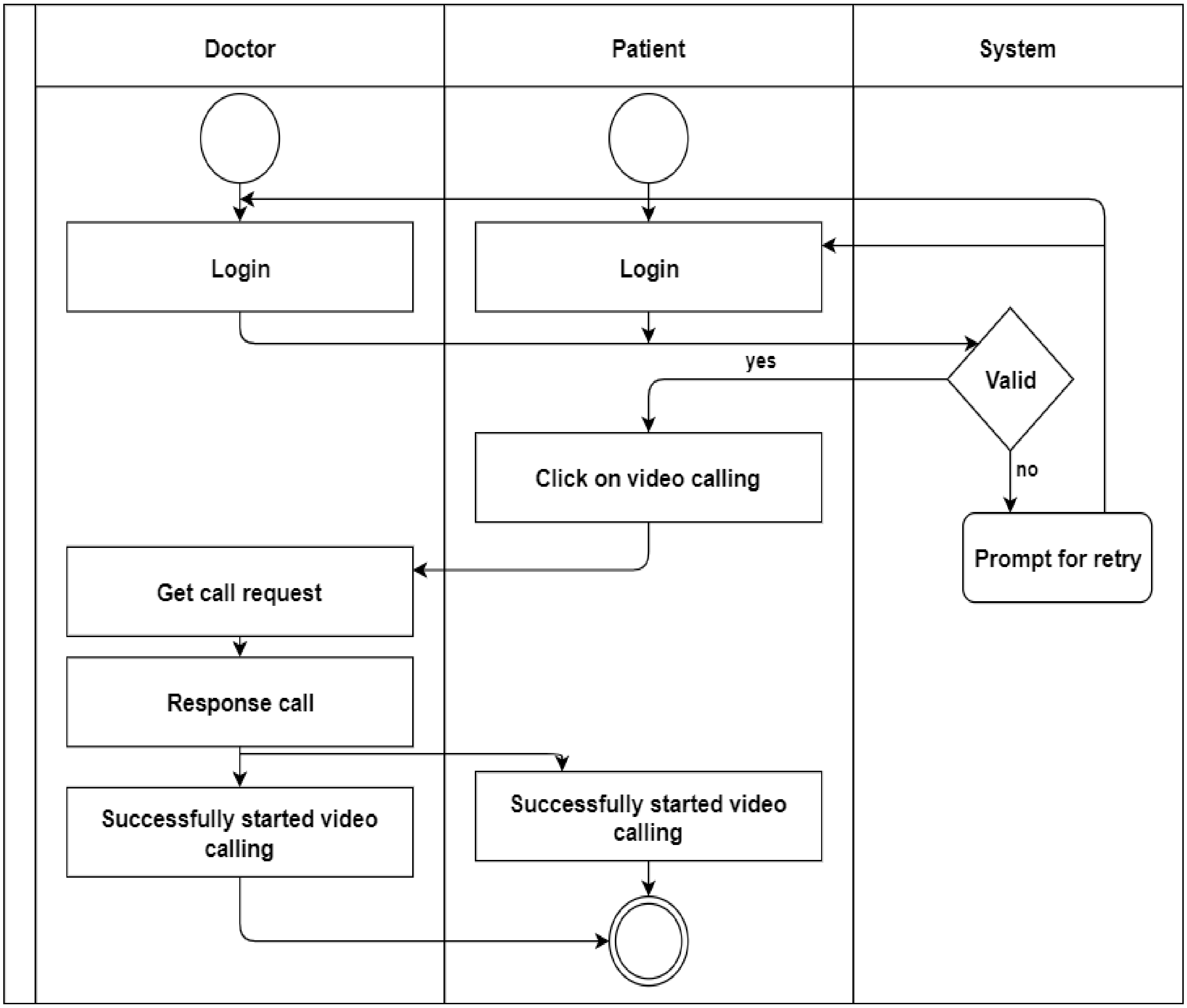


Figure 21: Swim lane diagram for video calling

4.3.7 Use Case for Recommendation

Use case: Recommendation

Primary Actor	:	Patient
Secondary Actor	:	System
Goal in Context	:	For recommendation
Priority	:	Essential, must be implemented.

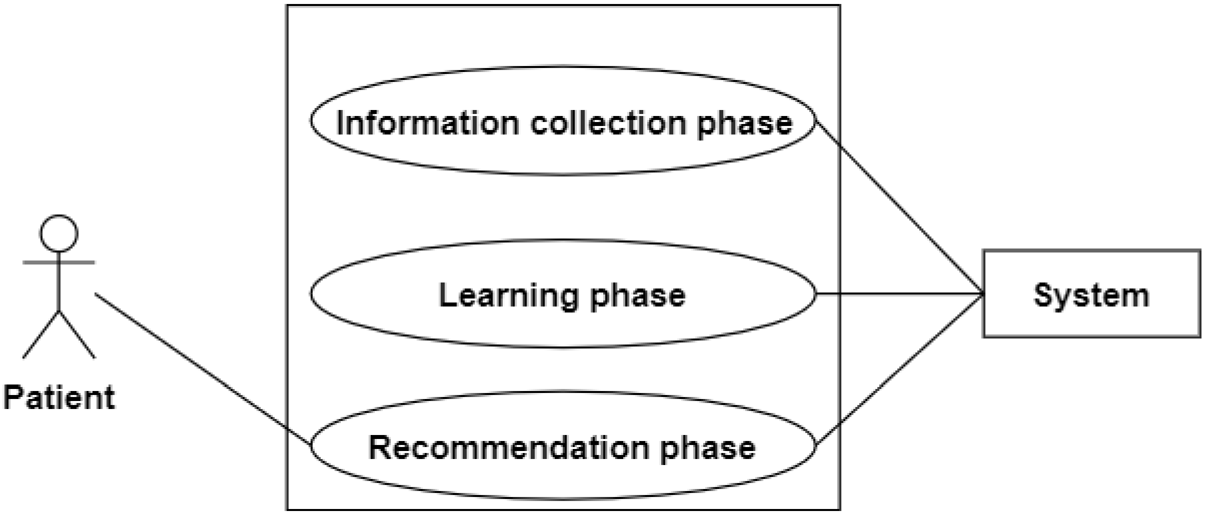


Figure 22: Use case for recommendation

Activity diagram of information collection phase for patient is in figure-23

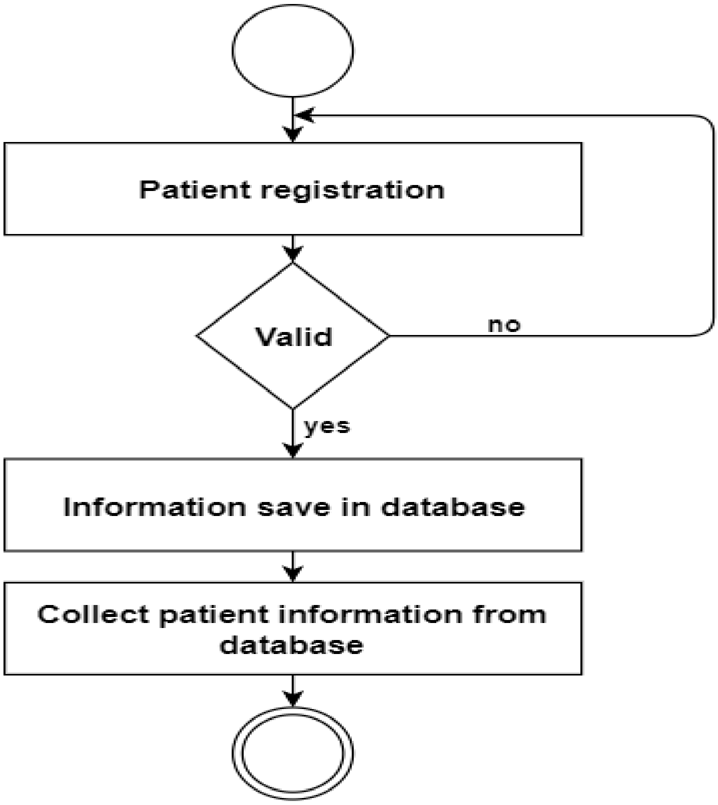

Patient registration
Valid
no
yes
Information save in database
Collect patient information from database

Figure 23: Activity diagram of information collection phase for patient

Activity diagram (Information collection phase): Information collect from patient registration.

Activity diagram of information collection phase for hospital is in figure-24

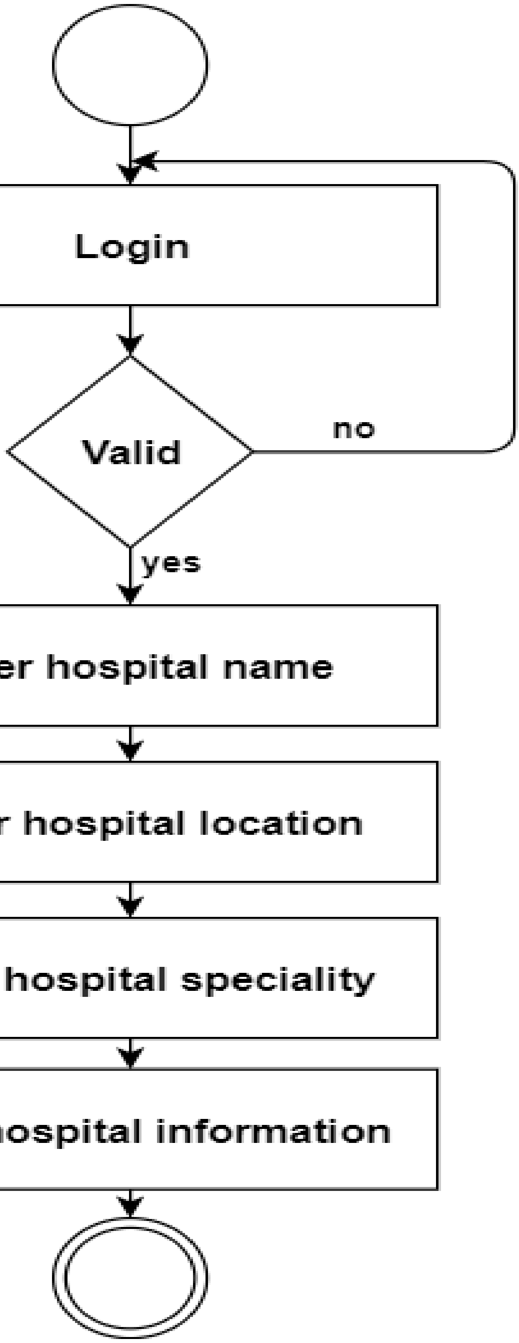


Figure 24: Activity diagram of information collection phase for hospital

Activity diagram (Information collection phase for hospital): Admin can enter hospital information after login into the system.

Swim lane diagram of information collection phase for patient is given below

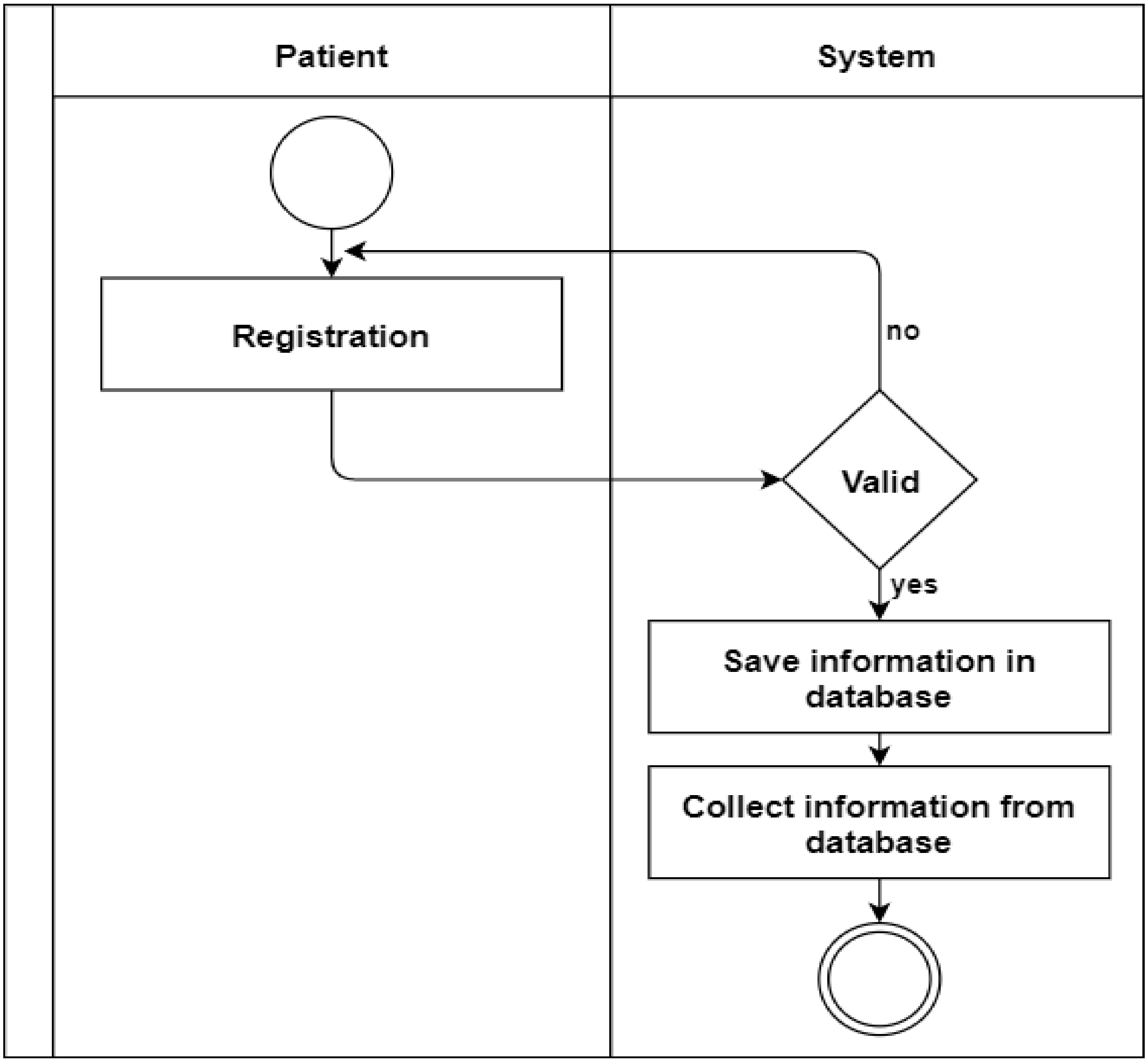

Patient
System
Registration
no
Valid
yes
Save information in database
Collect information from database

Figure 25: Information collection (patient) swim lane diagram

Swim lane diagram of information collection phase for hospital is given below

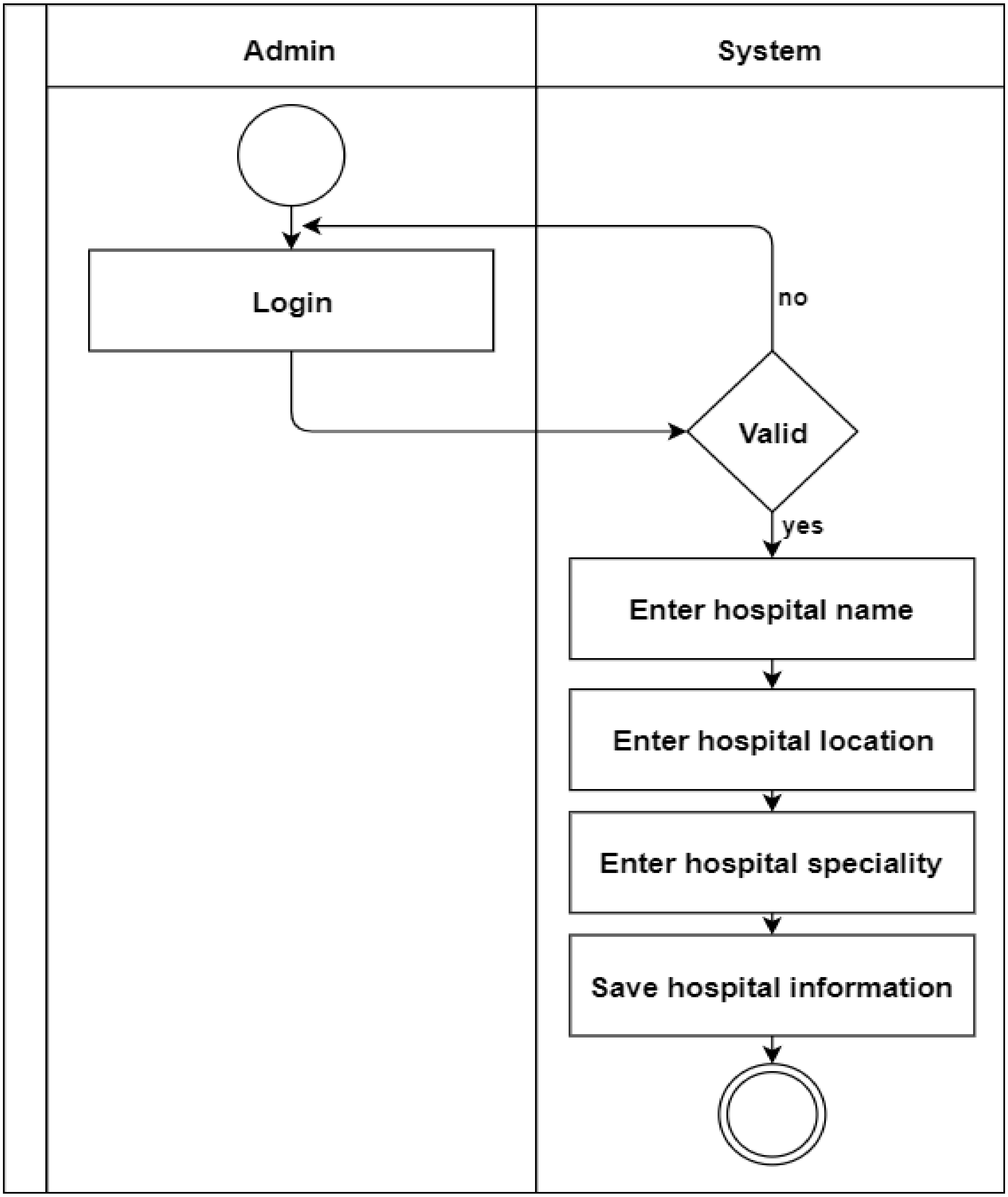


Figure 26: Information collection (hospital) swim lane diagram

Activity diagram for learning phase

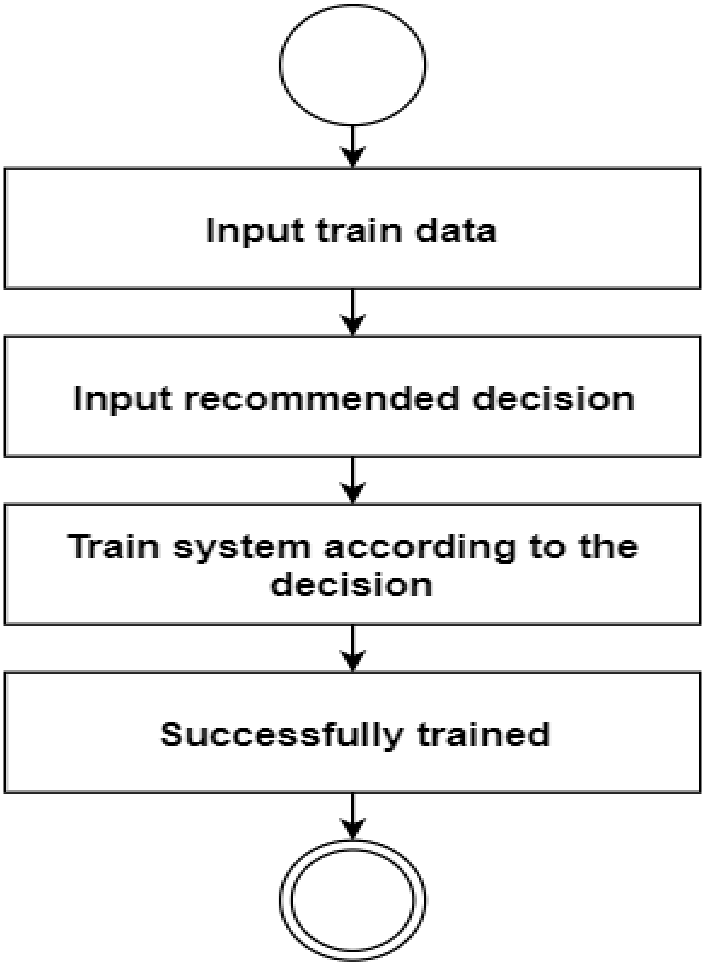

Input train data
Input recommended decision
Train system according to the decision
Successfully trained

Figure 27: Activity diagram for learning phase

Activity diagram for recommendation phase

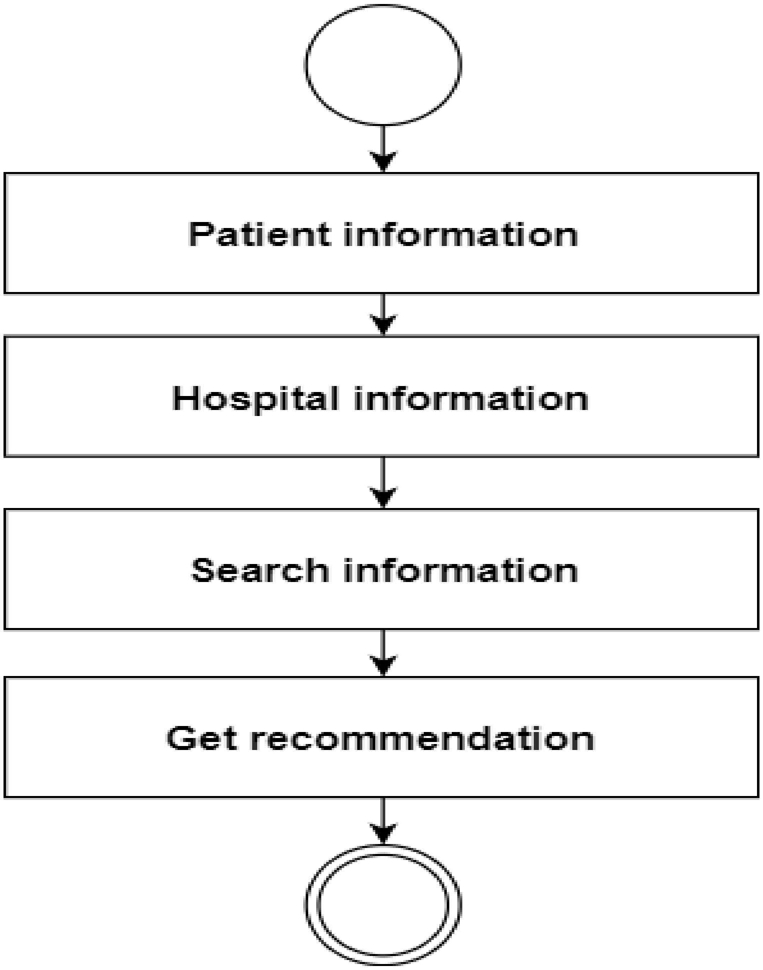


Figure 28: Activity diagram for recommendation phase

Swim lane diagram for recommendation

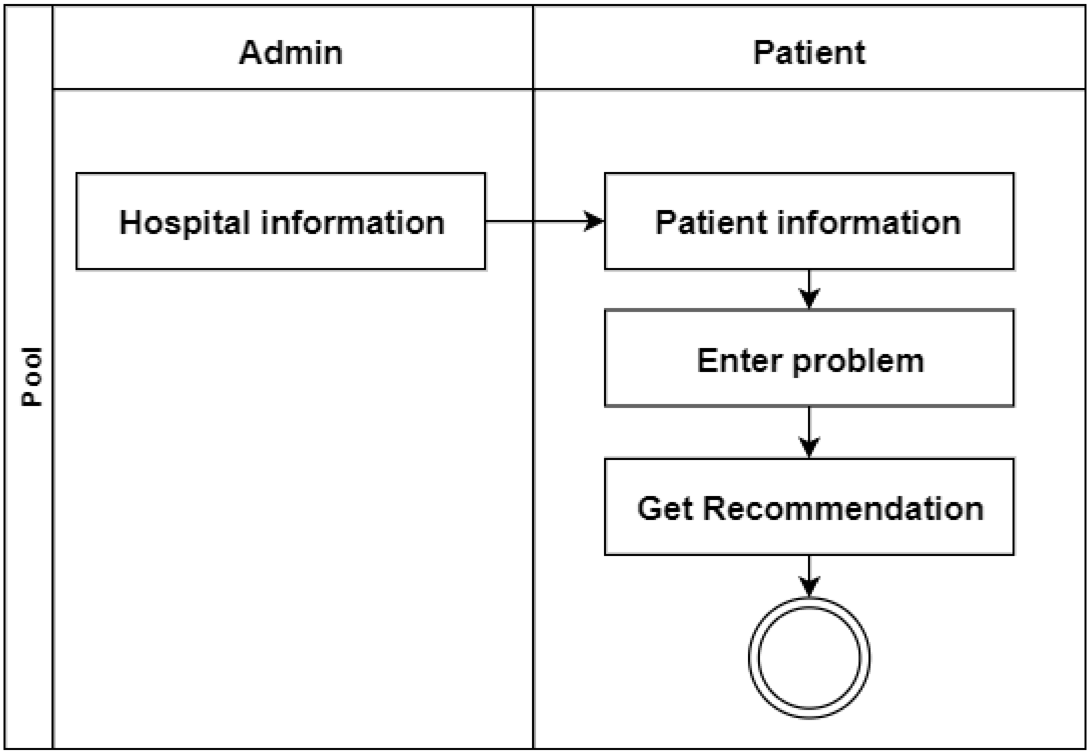


Figure 29: Swim lane diagram for recommendation

Chapter 5
Data Model

In this chapter I will discuss about the data models of Smart Health system.

5.1 Data Modeling Concept

If software requirements include the need to create, extend, or interface with a database or if complex data structures must be constructed and manipulated, a software team may choose to create a data model as part of overall requirements modeling.

5.2 Data Objects

A data object is representation of composite information that must be understood by software. Here, composite information means that has a number of different properties or attributes. A data object can be an external entity, a thing, an occurrence, a role, an organizational unit, a place or a structure.

Identify Data Objects

Nouns having attributes are selected as data object. Those who doesn’t have any attributes have covered under the data objects.

Data Object: User
Attributes:

- User id
- Password
- First Name
- Last Name
- Email id
- User role
- Picture
- Gender

Data Object: Role
Attributes

- Role id
- Role name

Data Object: Doctor
Attributes

- College/University
- Designation
- Speciality

- Hospital id
- Role id
- User id

Data Object: Patient
Attributes

- Age
- Height
- Weight
- Blood pressure
- Blood group
- Location
- Name of disease
- User id
- Role id

Data Object: Messaging
Attributes

- Patient id
- Doctor id
- Connection id

Data Object: Video calling
Attributes

- Patient id
- Doctor id
- Connection id

Data Object: Hospital
Attributes

- Hospital id
- Hospital name
- Location
- Speciality
- Description
- Doctor id

5.3 Entity Relationship Diagram

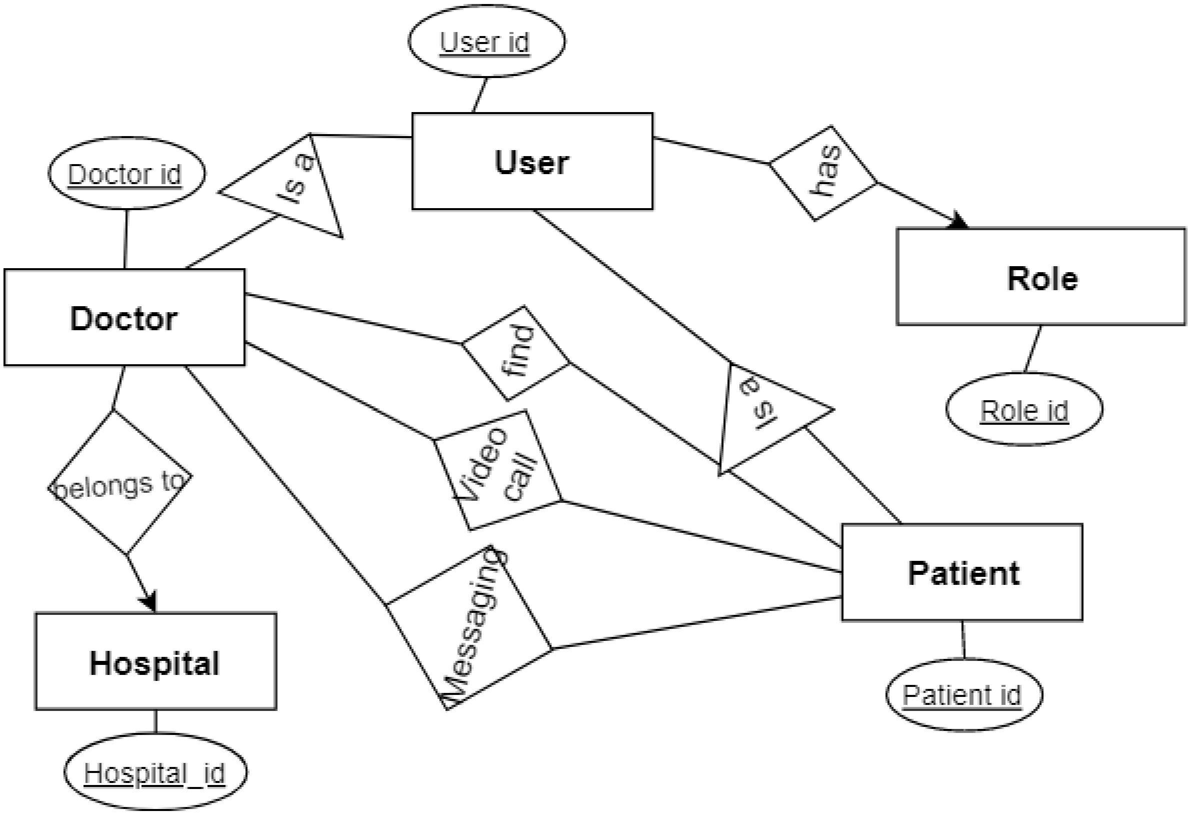


Figure 30: Entity diagram

Chapter 6
Class Based Model

6.1 Introduction

Class-based modeling is the operations that will be applied to the objects, relationships between the objects and the collaborations that occur between the classes that are defined.

6.2 General Classification

To identify potential classes, I have first find out all the nouns which are in solution space. Then analyses the nouns to find out if they are in following zero or more criteria:

- External entity
- Thing
- Occurrence
- Role
- Organizational unit
- Place
- Structure

G.C	Noun
External entity	First name, last name, username, password, gender, email id, designation, college/university name, specialty, age, height, weight, blood pressure, blood group, address
Thing	Recommendation system
Occurrence	Authentication, messaging, video calling, recommendation
Role	Doctor, patient
Organizational unit	Doctor, patient
place	-
Structure	-

6.3 Selection Criteria

There have six characteristics that are given below:

- Retained information
- Needed Services
- Multiple Attributes
- Common Attributes

- Common operations
- Essential Requirements

Table 3: Potential Class Identification

Noun	Problem/Solution space	Selection criteria	General criteria
Authentication	s	Accepted	1,2,3,4,5
Doctor	s	Accepted	1,2,3,4,5,6
Messaging	s	Accepted	1,2,3,4,5,6
Video calling	s	Accepted	1,2,3,4,5,6
Patient	s	Accepted	1,2,3,4,5,6
Recommendation	s	Accepted	1,2,3,4,5,6
User	s	Accepted	1,2,3,4,5,6
Role	s	Accepted	1,2,3,4,5,6

6.4 Attributes Section

Here I find attributes for selected classes.

Table 4: Selected Classes and their attributes:

Selected class	Attributes
Authentication	User id, first name, last name, email id, password, gender, role id, designation, college/university name, specialty, age, height, weight, blood pressure, blood group, address, hospital name
User	User id, first name, last name, email id, password, gender, role id
Doctor	designation, college/university name, hospital name, specialty
Patient	age, height, weight, blood pressure, blood group, address
User role	Role id, role name
Messaging	Message id, connection id, user id
Video calling	Connection id, user id
Hospital	Hospital id, name, location, specialty, description

6.5 Method Selection

Table 4: Selected method

Selected class	Method
Authentication	Signup(), Login(), Logout(), Update_Profile()
User	Get(), Set()
Doctor	Get(), Set(), GetMessage(), GetVideoCall()
Patient	GetDoctorList(), GetRecommendation(), SendMessage(), SendVideoCall(0
User role	Get(), Set(), SendRole()
Messaging	GetUserId(), SetConnection()
Video calling	GetUserId(), SetConnection()
Hospital	SendHospitalList()

6.6 Class Responsibility

Class responsibility is shown below

Table: Class responsibility

Class	Responsibility
User	Entering the system
Doctor	Creating profile
Patient	Messaging and video calling with doctor, take recommended information
Message	Make a bridge between patient and doctor via message chatting
Video conference	Make a bridge between patient and doctor via video calling
Hospital	Sending hospital list for recommendation
Database	Storing and retrieving information

6.6 Class Responsibility Collaboration (CRC)

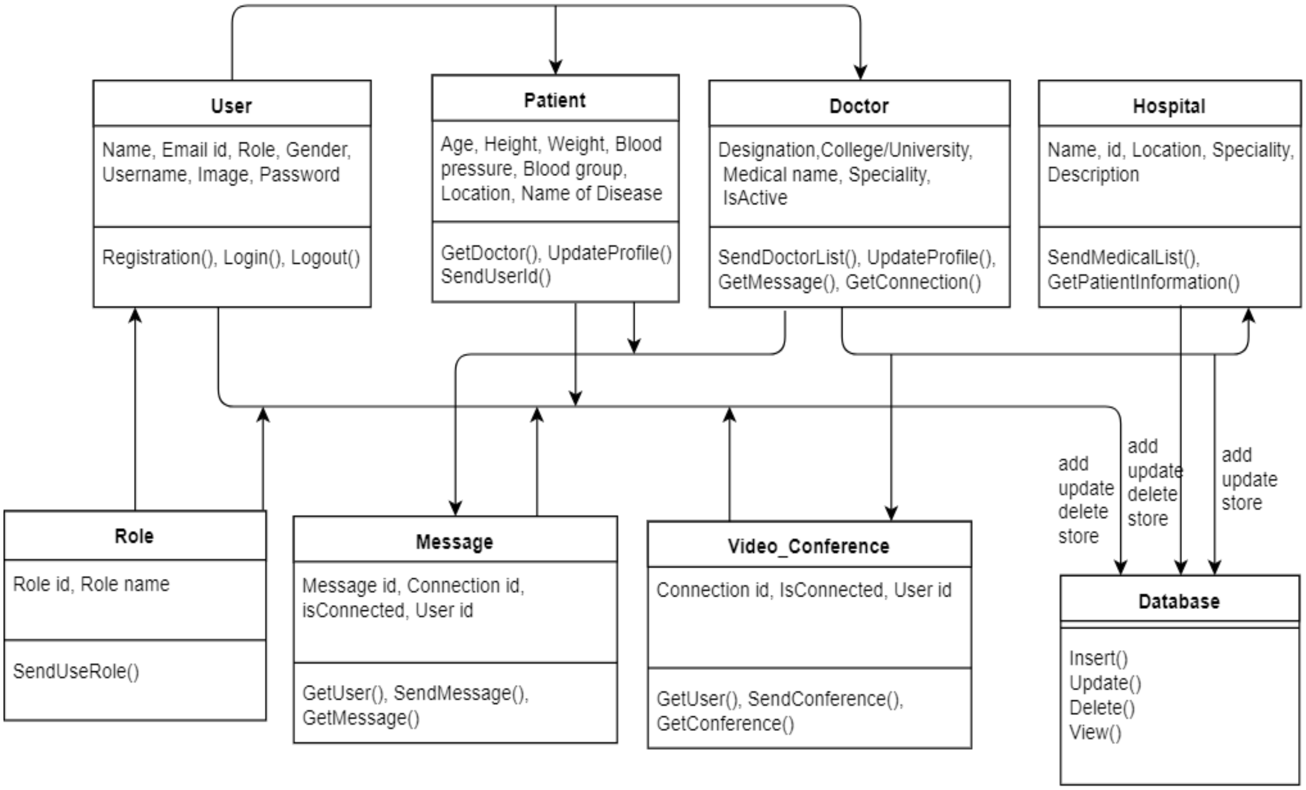

User
Name, Email id, Role, Gender, Username, Image, Password
Registration(), Login(), Logout()
Patient
Age, Height, Weight, Blood pressure, Blood group, Location, Name of Disease
GetDoctor(), UpdateProfile() SendUserId()
Doctor
Designation,College/University, Medical name, Speciality, IsActive
SendDoctorList(), UpdateProfile(), GetMessage(), GetConnection()
Hospital
Name, id, Location, Speciality, Description
SendMedicalList(), GetPatientInformation()
Role
Role id, Role name
SendUseRole()
Message
Message id, Connection id, isConnected, User id
GetUser(), SendMessage(), GetMessage()
Video_Conference
Connection id, IsConnected, User id
GetUser(), SendConference(), GetConference()
add update delete store
add update delete store
add update store
Database
Insert() Update() Delete() View()

Figure 31: Class diagram

Chapter 7
Flow-Oriented Model

7.1 Introduction

Although data flow-oriented modeling is perceived as an outdated technique by some software Engineers, it continues to be one of the most widely used requirements analysis notations in use Today.

7.2 Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) takes an input-process-output view of a system. Data objects flow into the software, are transformed by processing elements and resultant data objects flow out of the software. Data objects are represented by labeled arrows and transformations are represented by circles.

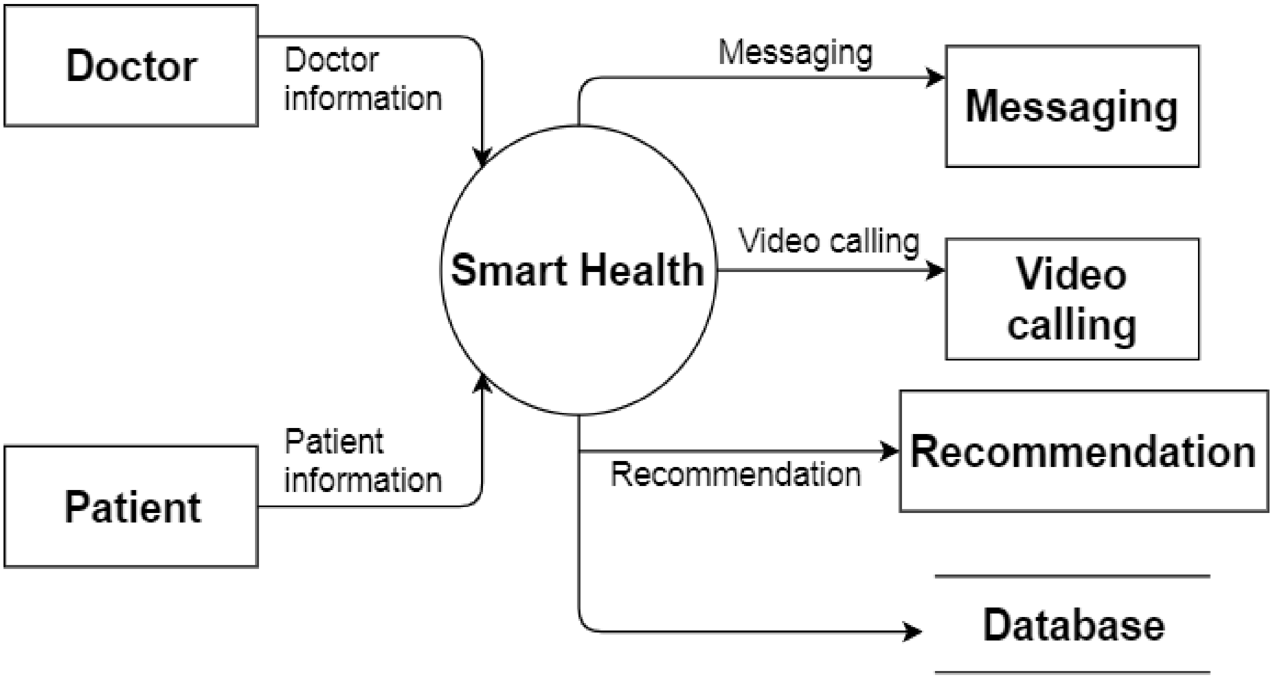


Figure 32: Data flow diagram level-1

Chapter 8
Behavioral Model

8.1 Introduction

The behavioral model indicates how software will respond to external events or stimuli. To create the model, some steps should be followed. These steps are given below:

- Evaluate all use cases to fully understand the sequence of interaction within the system.
- Identify events that drive the interaction sequence and understand how these events relate to specific objects.
- Create a sequence for each use case.
- Build a state diagram for the system.
- Review the behavioral model to verify accuracy and consistency.

8.2 Identifying Events with the Use Case

Use case for Smart Health authentication.

User uses the keypad to key in a six digit password. The password is compared with the valid password stored in the system. If the password is incorrect, it shows an error message and user may try again for additional input. If the password is correct, user can access more options.

8.3 State Transition

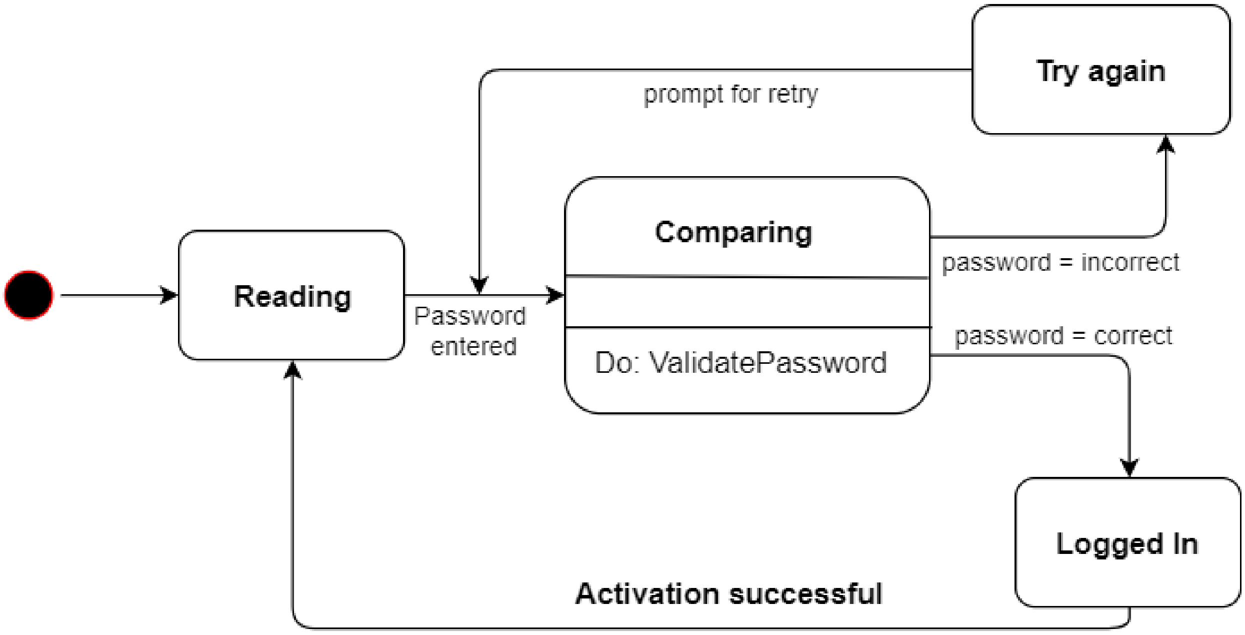


Figure 33: State transition

Messaging and video calling

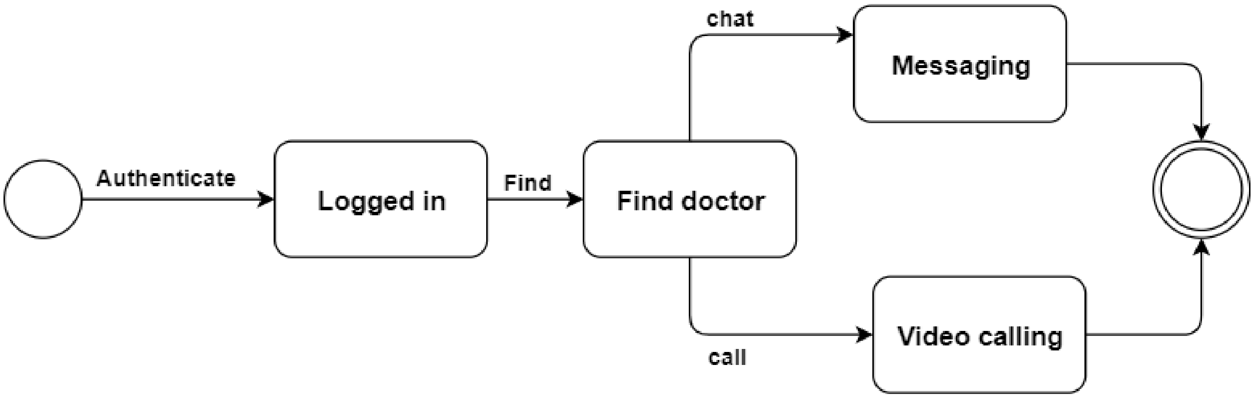


Figure 34: State transition for messaging and video calling

Recommendation

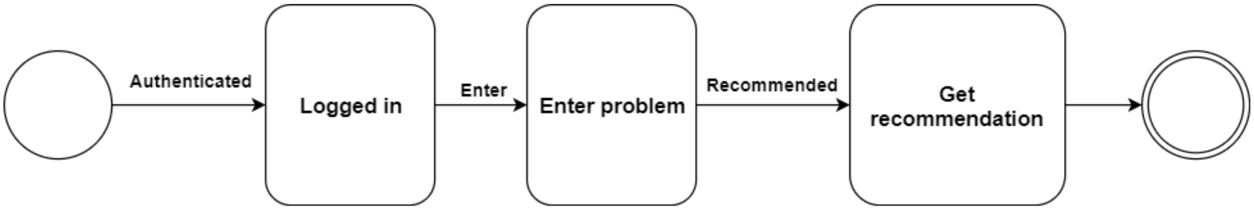


Figure 35: State transition for recommendation

8.4 Sequence Diagram

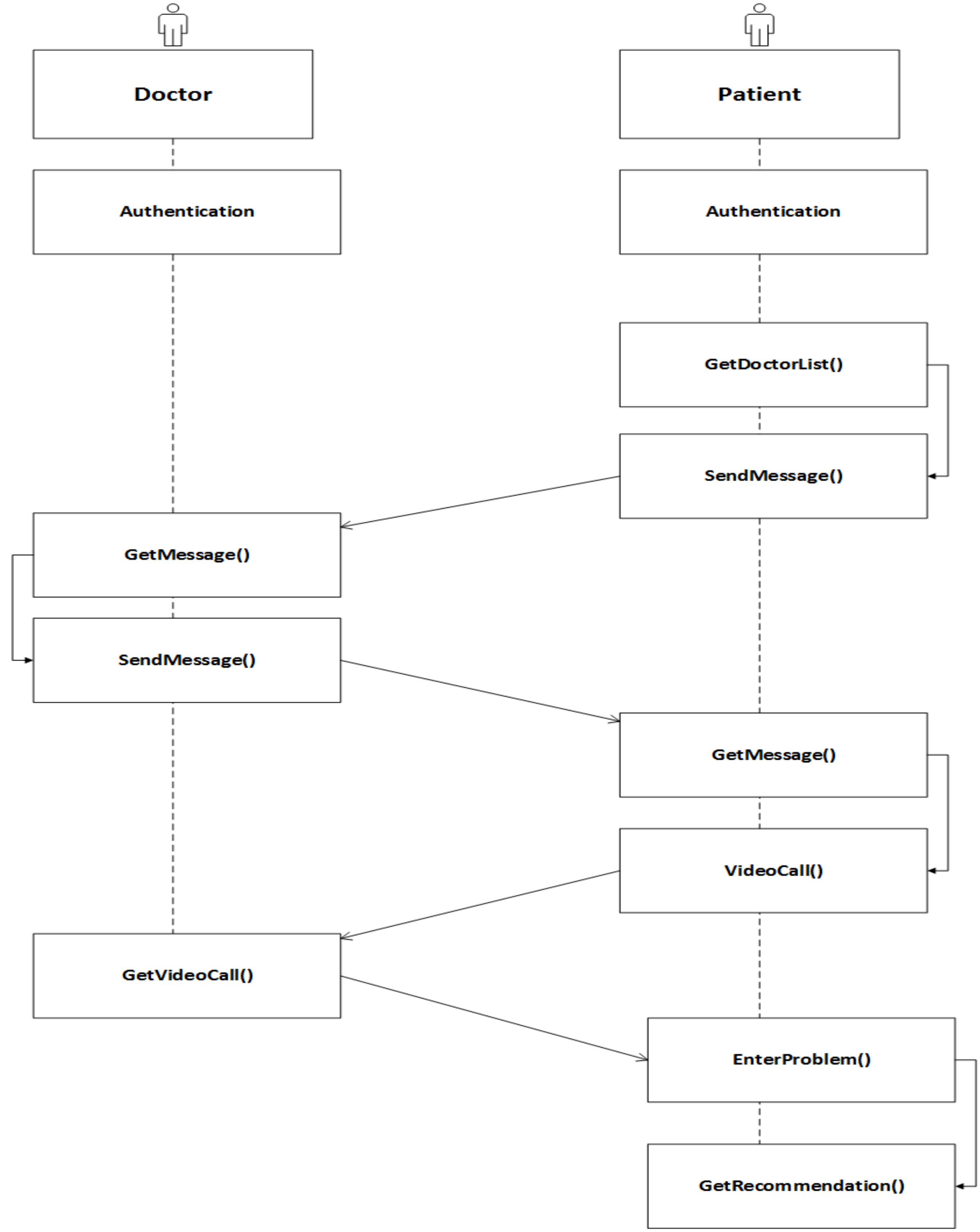


Figure 36: Sequence diagram

Chapter 9
Architectural Design

9.1 Architectural Design for OOP

9.1.1 Representing the system in context

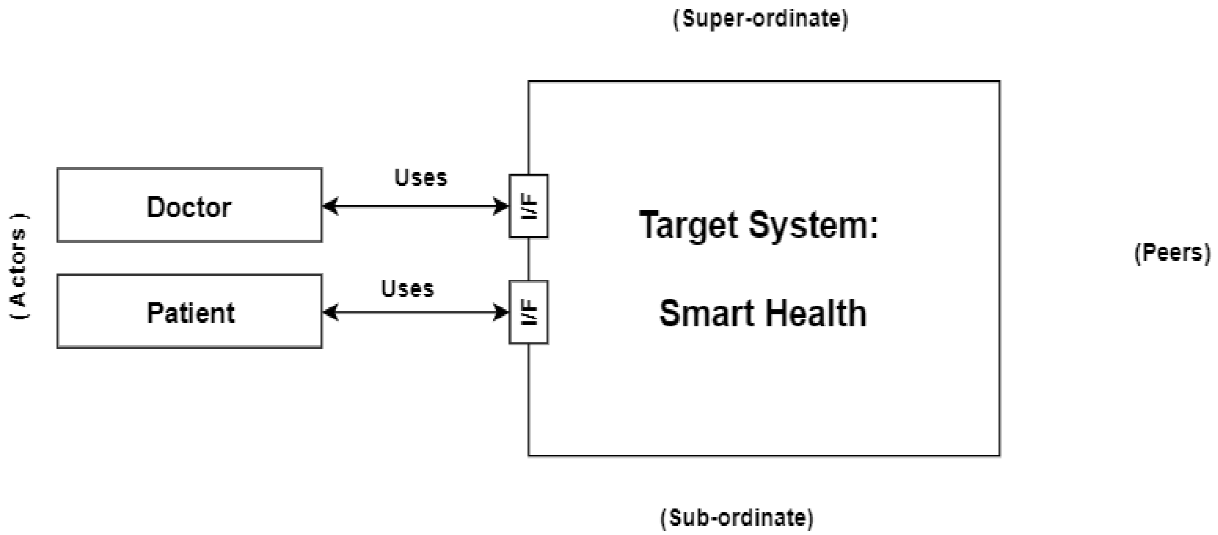


Figure 9.1.1: Representing the system in context

9.1.2 Define Archetypes

1. Authentication
2. Find doctor
3. Messaging
4. Video calling
5. Recommendation

9.1.3 Refining architecture into components

Components

1. Doctor details, Patient details
2. Doctors’ details
3. Message details
4. Video conferencing
5. Patient details, Recommended hospital

Classes:

1. Doctor, Patient, DAL
2. DAL, Doctor
3. Patient, Doctor, Message, DAL
4. Patient, Doctor, Video conference, DAL
5. Hospital, Patient, DAL

9.1.4 Describing Instantiation of the system

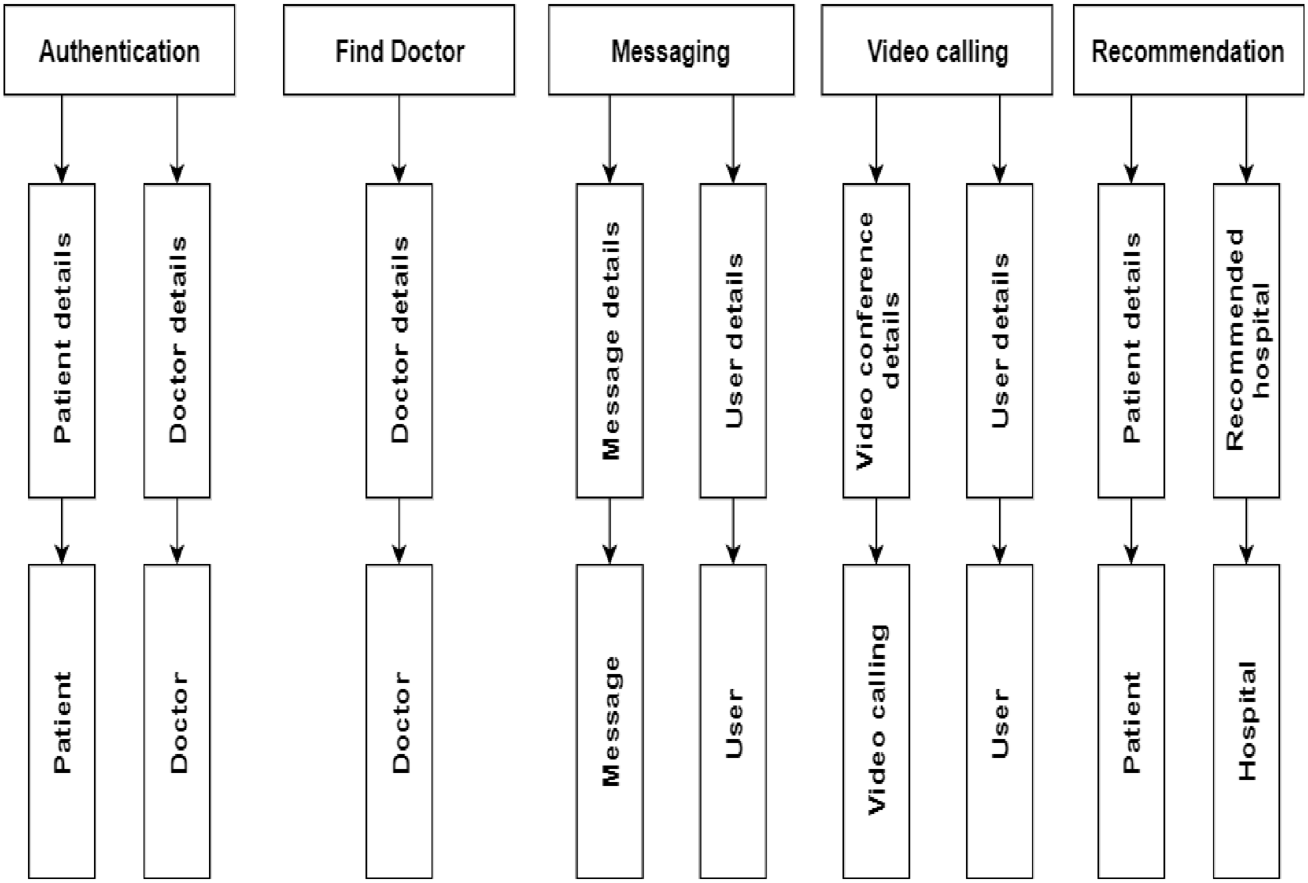


Figure 9.1.4: Refining archetype into component & classes

9.1.5 Mapping Requirements into Software Architecture

As this system built in OOP concept so mapping requirement into software architecture is not needed here.

Chapter 10
Component Design

10.1 Analysis Class

This is the first step of object-oriented approach in component level design. In this section, all design classes that correspond to the problem domain as defined in the analysis model and architectural model are identified. The analysis classes for this project are:

1. User
2. Doctor
3. Patient
4. Messaging

Elaborate class

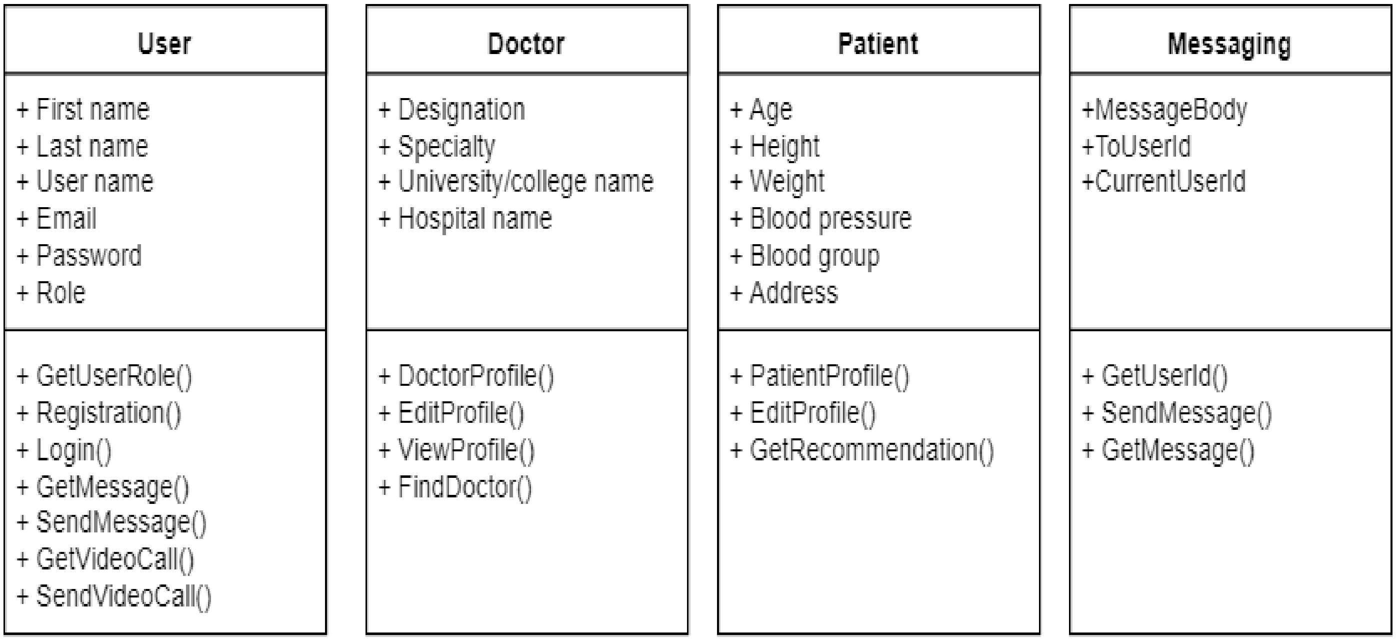


Figure 10.1: Elaborate classes

Analyzing class and design component

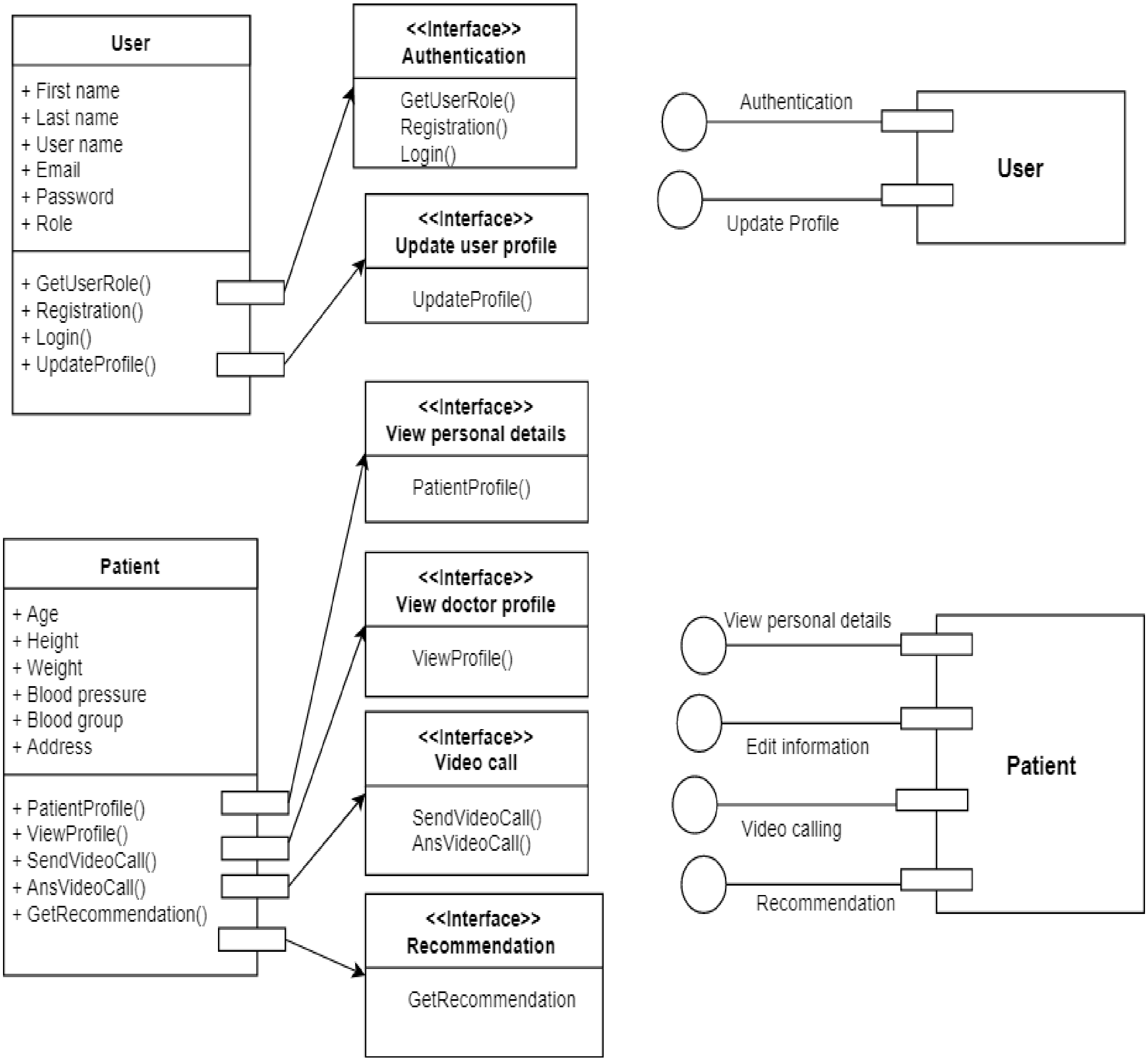


Figure 10.2(A): Analyzing class and design component of user and patient classe

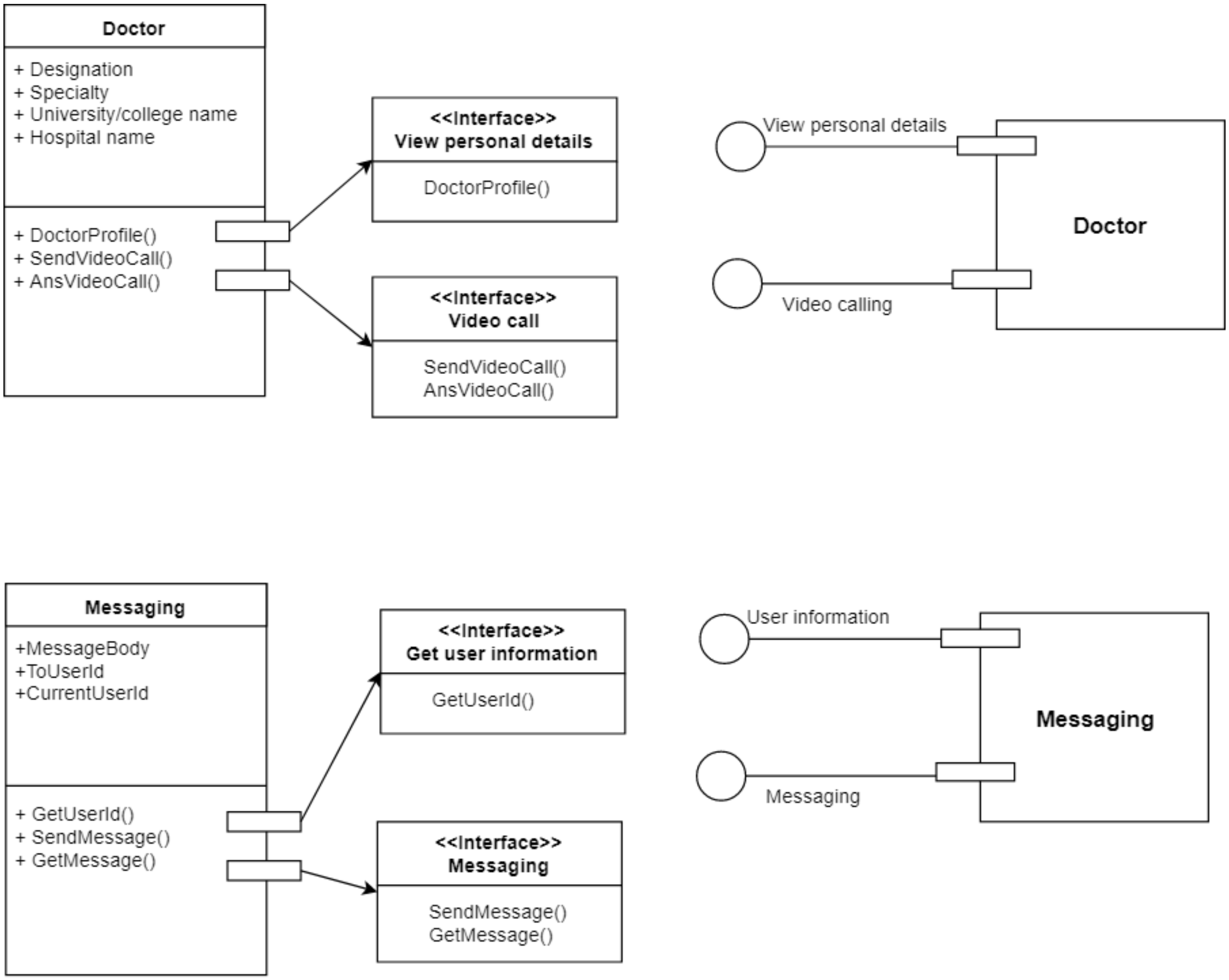


Figure 10.2(B): Analyzing class and design component of doctor and message class

10.2 Infrastructure Class

In this section, all design classes that correspond to the infrastructure domain are identified and listed. These classes are usually not present in the analysis or architectural models. These classes include GUI components, operating system components, data management components, networking components, etc. For Smart Health project, the infrastructure classes are listed below:

1. Video calling
2. Recommendation

10.3 Collaboration Details

In this section, collaboration between classes will be designed. The designed diagram is given below:

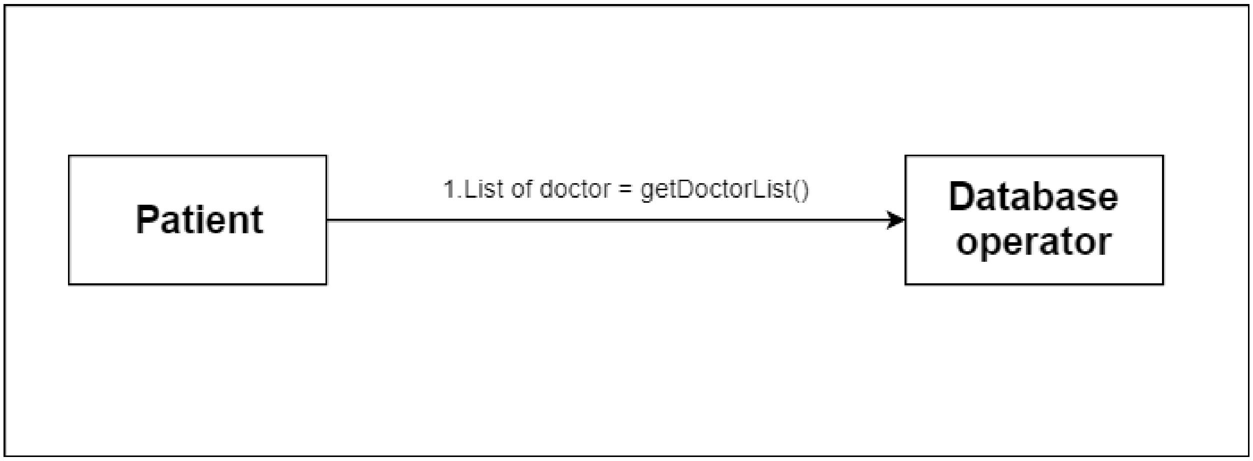


Figure 10.3: Collaborative design for student class

10.4 Appropriate Interfaces

There is no necessity to divide the classes in subclasses as they exhibit sufficient cohesion. So there is no need to define appropriate interfaces.

10.5 Elaborate Attributes

Elaborate attributes for System **User** Class:

First-name: string = “name1” {contains name value – Shuvo}

Last-name: string = “name2” {contains name value – Roy}

Email: email = “null” {Contains value – shuvo@gmail.com}

Role: string = “null” {Patient, Doctor}

Username: string = “username” {All Characters}

Password: password= “password” {All Characters}

Elaborated attributes for **Patient** Class:

Age: integer = “null” {contains value – 0, 1 …….. }

Height: Decimal = “null” {contains value – 6.5 …….}

Weight: Decimal = “null” {contains value – 60.5 …….}

Blood Pressure: string = “null” {contains value – Normal, High ……..}

Blood group: string= “null” {contains value – 0-, A+ ………}

Address: string = “null” {contains value – any address}

Elaborated attributes for **Doctor** Class:

Designation: string = “null” {contains value – MBBS …….. }

Specialty: string= “null” {contains value – Kidney, High …….}

Hospital name: string= “null” {contains value – Dhaka medical………..}

10.6 Describe Processing Flow

The processing flow of the verification system is described below:

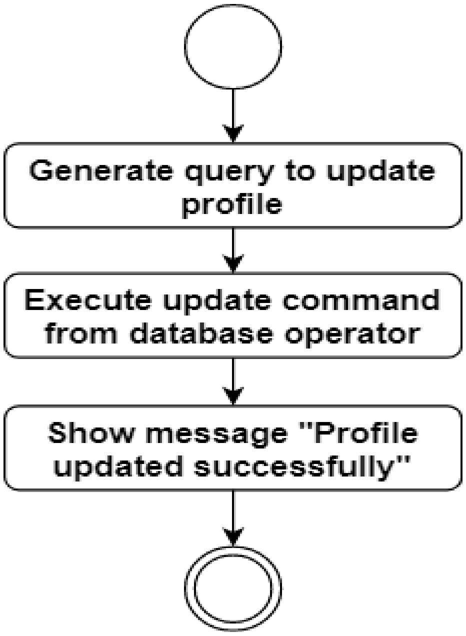

Generate query to update profile
Execute update command from database operator
Show message "Profile updated successfully"

Figure 10.6(A): Update profile

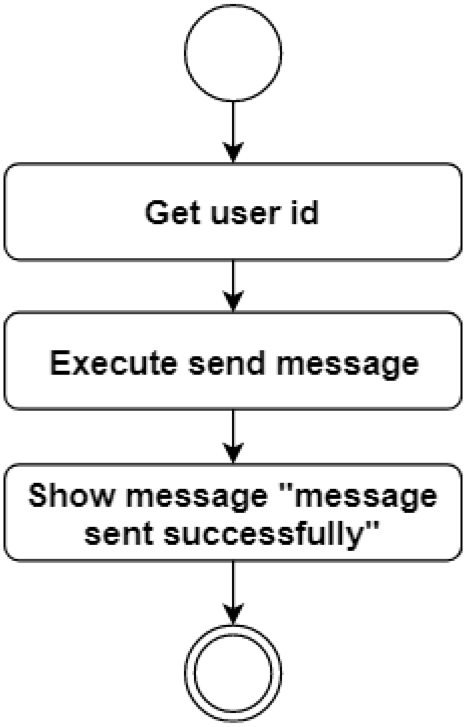


Figure 10.6(B): Send message

10.7 Persistent Data

Persistent data source: Database

Classes to manage data source: Database classes for each entity

10.8 Elaborative Deployment

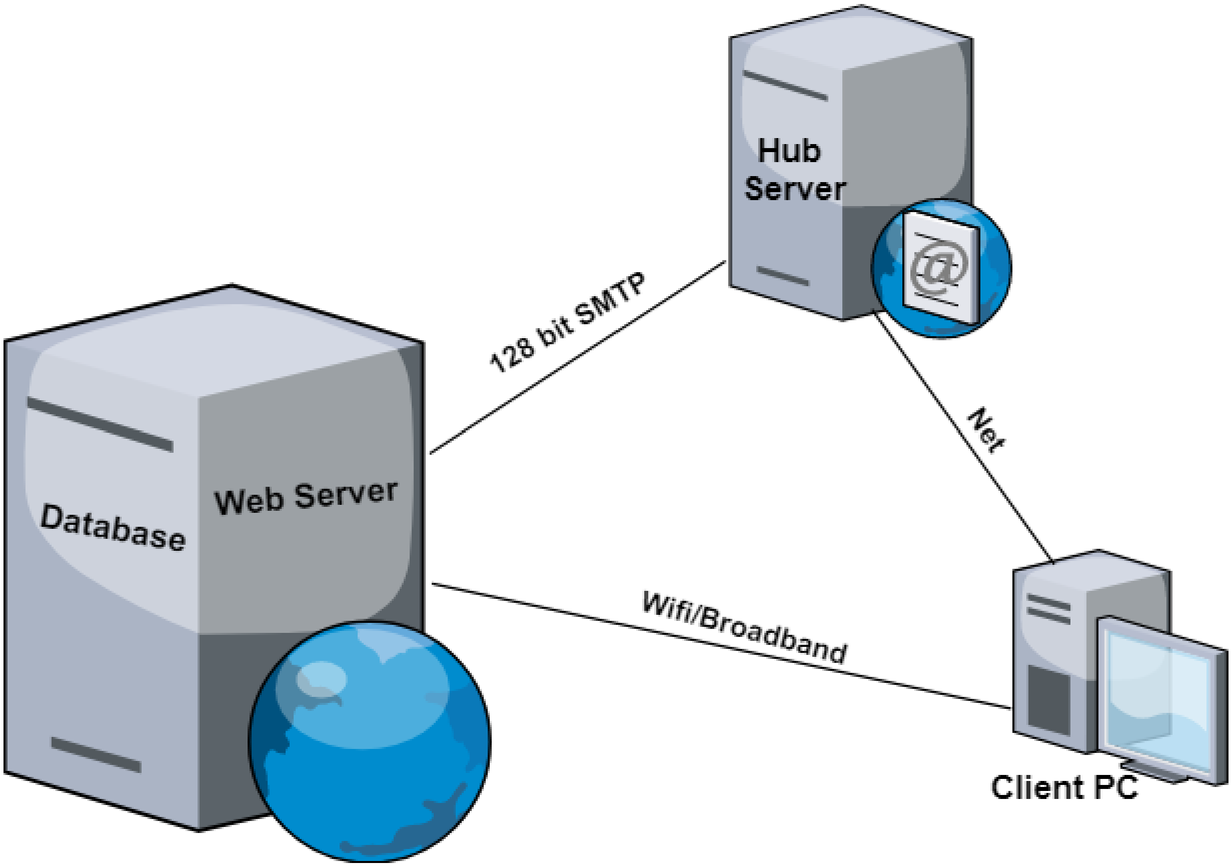


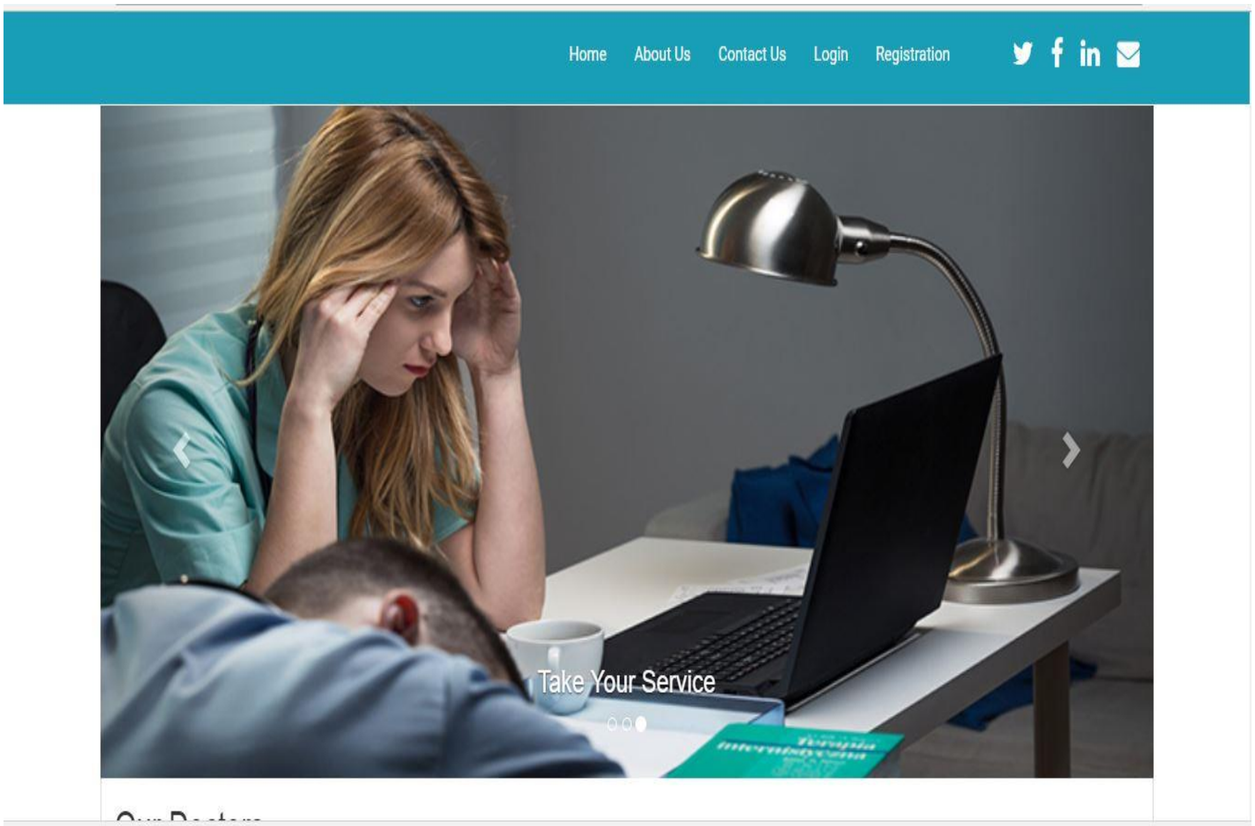
Figure 10.8: Elaborative Deployment of Smart Health

Chapter 11
Interface Design

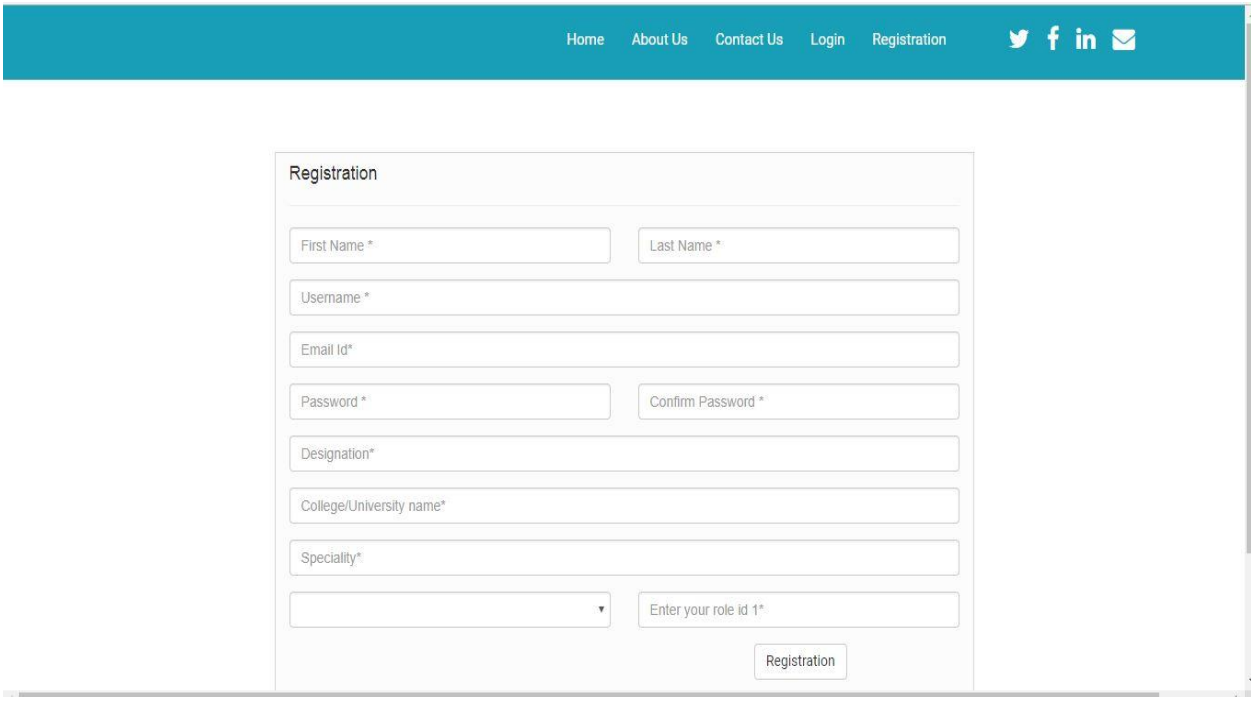
11.1 Introduction

User interface design (UI) or user interface engineering is the design of user interfaces for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing the user experience. User interface design creates an effective communication medium between a human and a computer. Following a set of interface design principles, design identifies interface objects and actions and then creates a screen layout that forms the basis for a user interface prototype. When we run the application, the following window will appear:

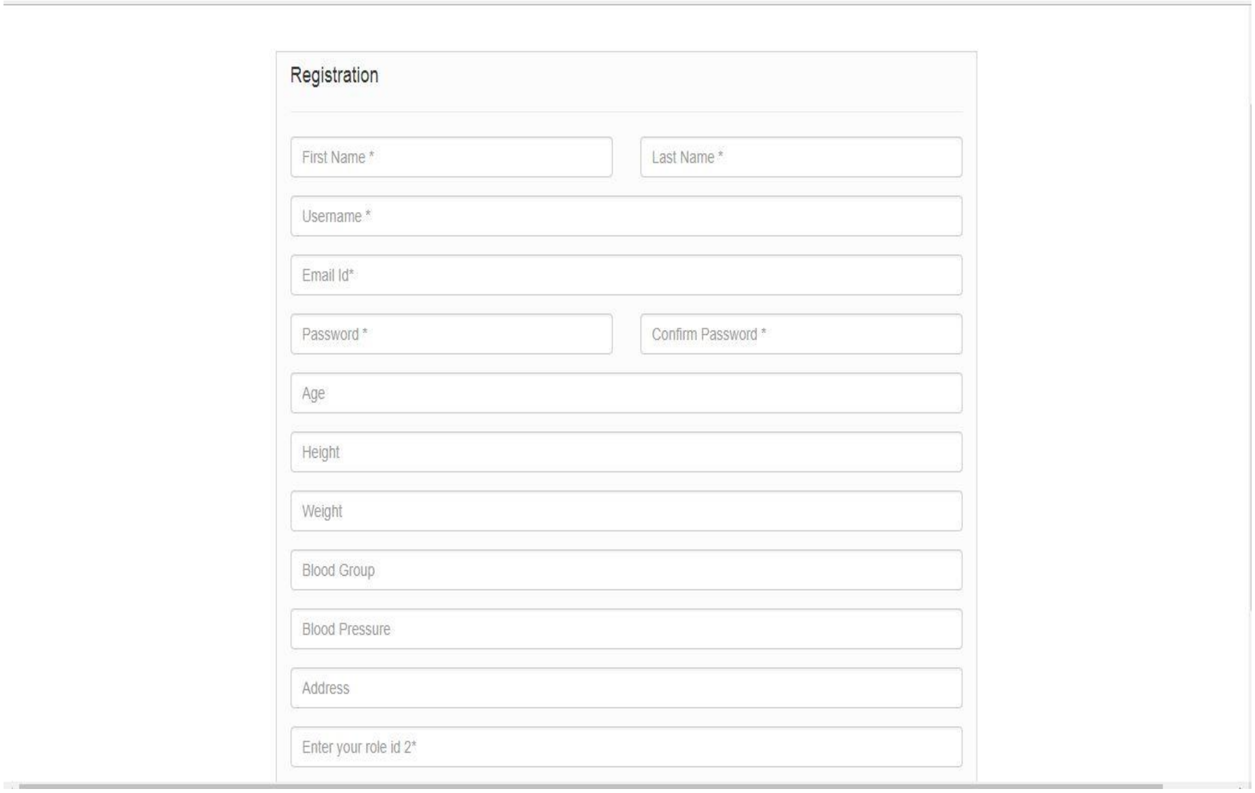
First appearance of smart health webpage. User just get a page for registration and login. Without registration or login, user can’t do anything.



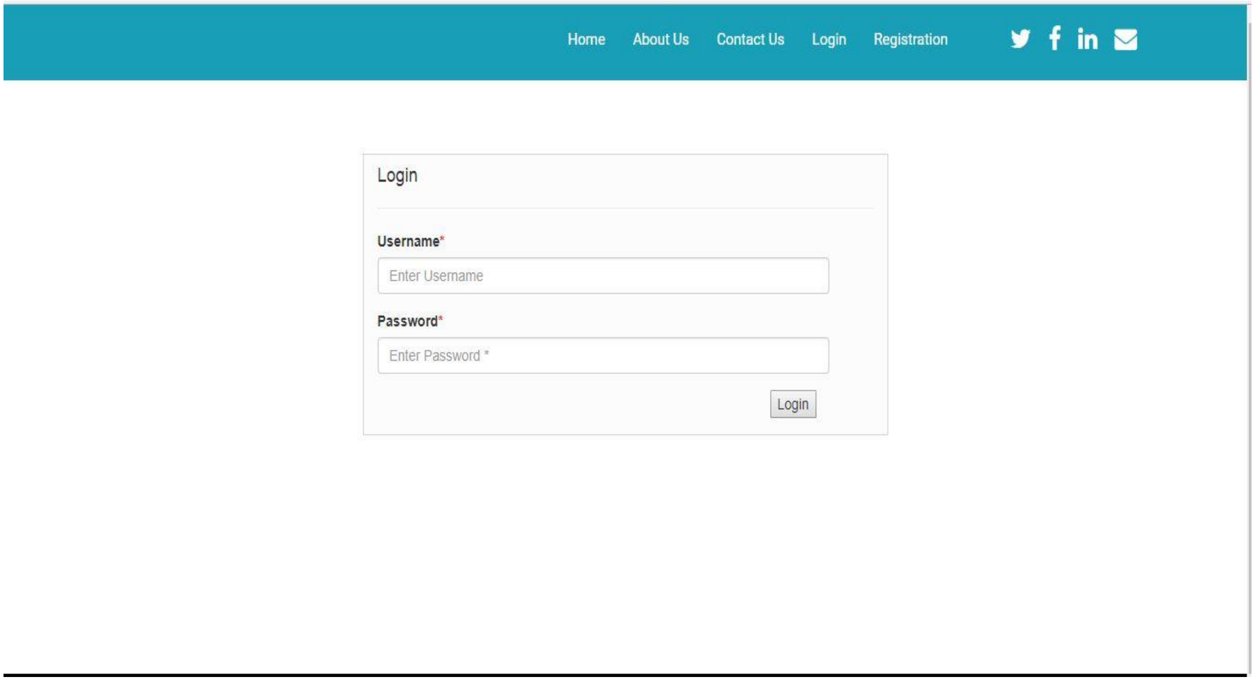
Registration page of doctor



Registration page of patient



Login page for user



Login

Username*

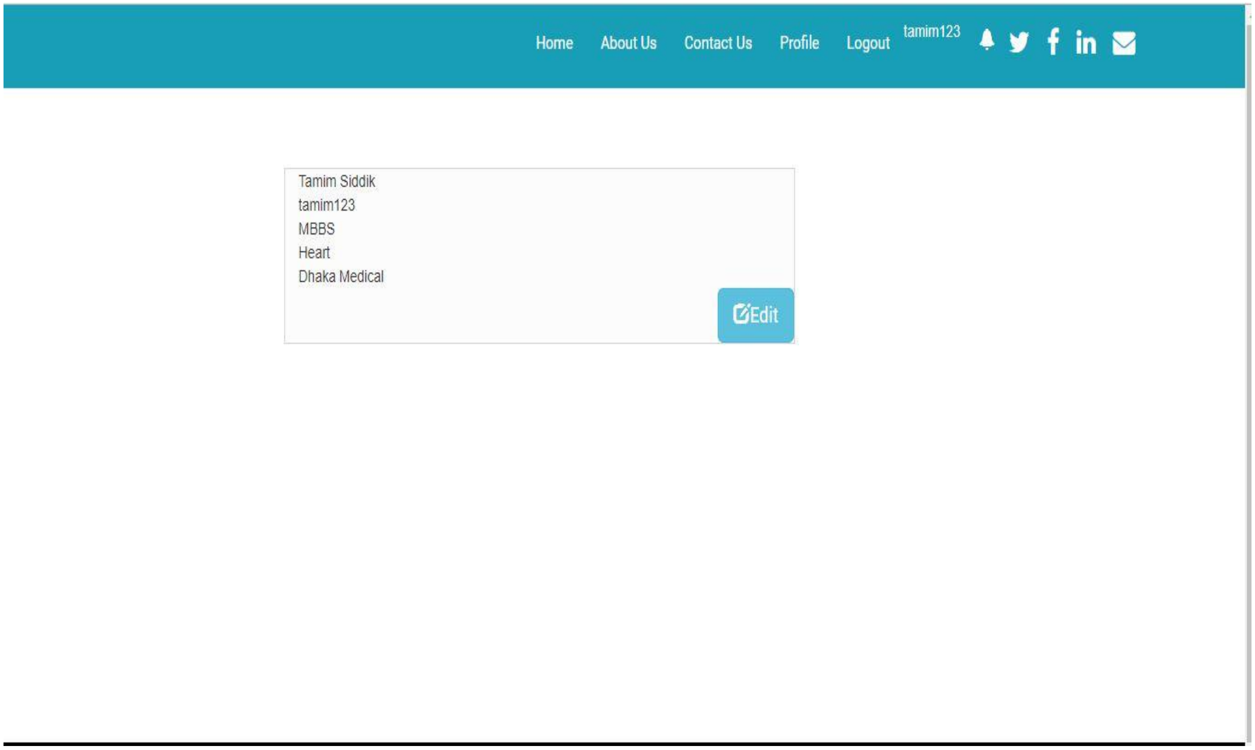
Enter Username

Password*

Enter Password *

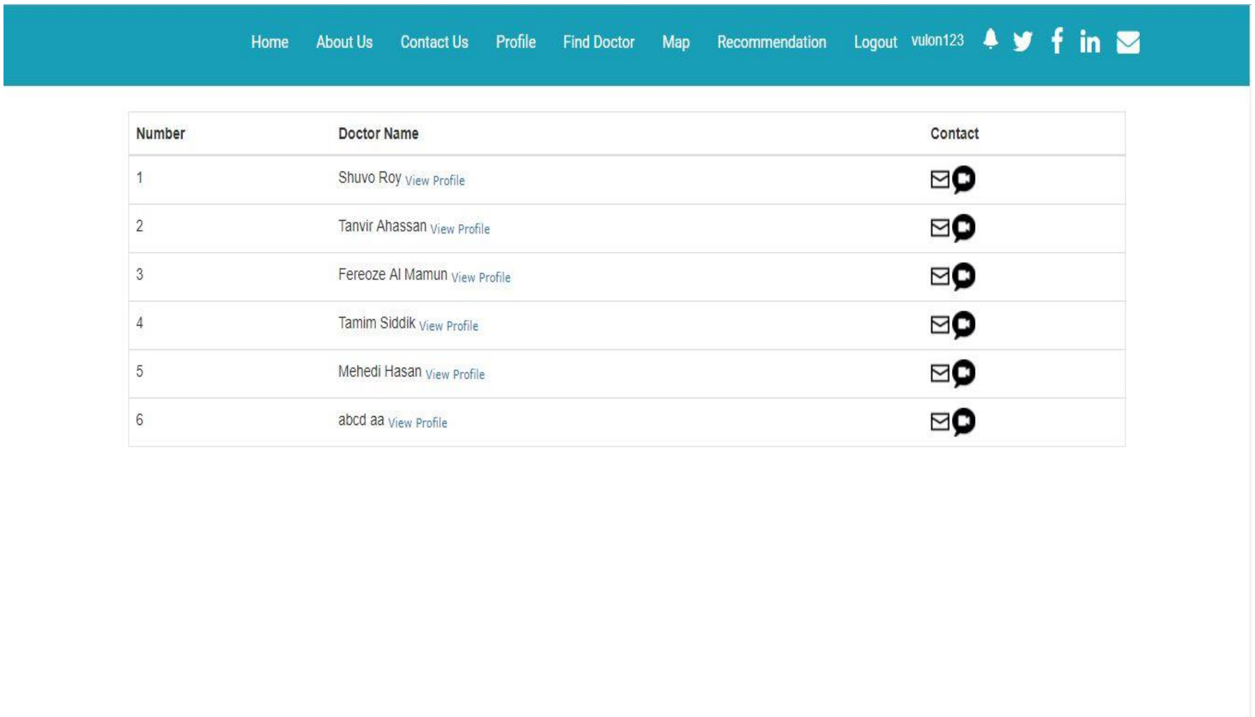
Login

Doctor profile and doctor site page



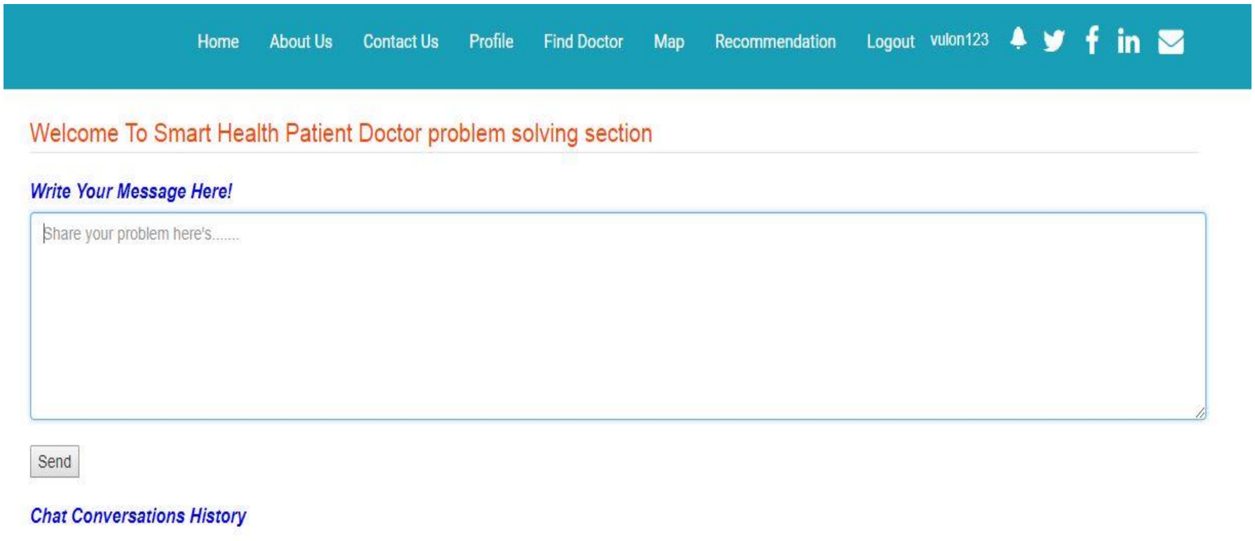
Tamim Siddik
tamim123
MBBS
Heart
Dhaka Medical

Patient site webpage and find doctor page



Number	Doctor Name	Contact
1	Shuvo Roy View Profile	✉ 💬
2	Tanvir Ahassan View Profile	✉ 💬
3	Fereoze Al Mamun View Profile	✉ 💬
4	Tamim Siddik View Profile	✉ 💬
5	Mehedi Hasan View Profile	✉ 💬
6	abcd aa View Profile	✉ 💬

Sending message page



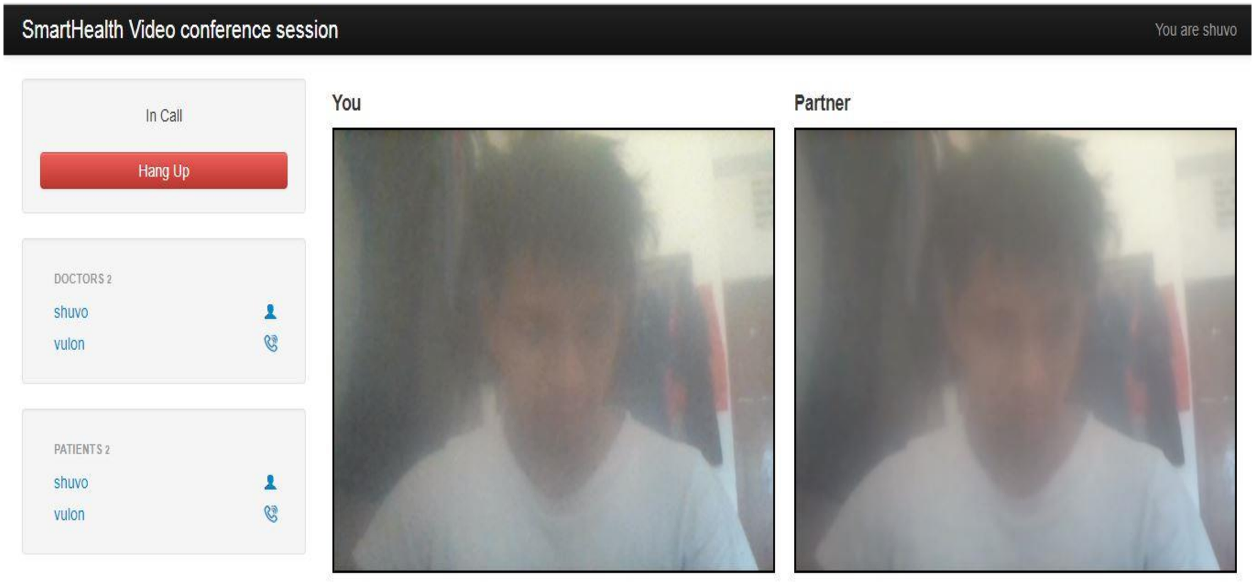
Welcome To Smart Health Patient Doctor problem solving section

Write Your Message Here!

Share your problem here's.......

Chat Conversations History

Video calling page



Chapter 12
Testing

Version	Date	Document Name
1.0	29-10-2017	TEST PLAN OUTLINE

12.1 Test plan identifier

Master test plan id for Smart Health project TP_1.0.

12.2 Introduction

This test plan describes the testing approach and overall framework that will drive the testing of the Smart Health version 1.0. The test plan document is created during the planning phase of the project.

12.2.1 Summary of Items and Features to be tested

Web application version is to be tested for Smart Health project. In this application, it has some features that to be nested, these are: user authentication, user profile, find doctor, messaging, video calling and recommendation system.

12.2.2 Requirement and History of Items

The requirement of items that will be tested collect from the software requirement and specification (SRS) and architectural design primarily.

12.2.3 High-level Description of Testing Goals

Testing makes a software bug free and more secure. After release a software product, there have a lot of bug which may not solve during the development phase. These bugs are solved after testing phase. So after development, testing is required for developing a better software product.

12.2.4 Reference Document

- Software requirement and specification
- Architecture design
- Component design
- Interface design

12.3 Test-Items to be tested

- Web application of Smart Health project.

12.4 Features to be tested

- User authentication
- User profile
- Find doctor
- Messaging
- Video calling
- Recommendation

12.5 Features not to be tested

All feature of this system must be tested.

12.6 Approach

- I will use an online bug tracking tool called BUG track for posting bug.
- Before starting posting bug, training for using this tool will be available.
- I will test the functionality of the application as I will cover the black box testing for my project.
- I will test all modules in Google Chrome and Mozilla Firefox browser only.
- One level of regression testing will be possible for the project.
- Integration testing will be performed.

12.7 Item Pass/Fail Criteria

Stipulating the criteria that I will use to determine whether each test item of my project has passed or failed. The planning criteria gives the framework for how the project will be assessed and under what circumstances it will be released. At unit test level all test cases of the corresponding modules of Smart Health project will be tested with the specified percentage of number of minor defects. Although it is dependent on the number and strictness of the defects which may cause a failure or system crashes.

12.8 Suspension Criteria and Resumption Requirements

Suspension criteria will be used when it is needed to suspend all or a percentage of the testing activities when the testing has no value and the build is not working properly which is overall a wasting of resources. On the other hand, resumption criteria specify when testing can resume after it has been suspended. These will be applied in such situations.

Suspension when,

- Unavailability of external dependent systems during execution.
- A defect is introduced that cannot allow any further testing.

Resumption when,

- When the external dependent systems become available again.
- When a fix is successfully implement.

12.9 Test Deliverables

I will provide some deliverables after testing the application at the end of the project. These are given below:

- Test plan document
- Test cases
- Error log
- Problem report and corrective option

12.10 Environmental Needs

- Test data which will be provided is an environmental need as I need data to test. Except provided data I cannot perform any phase of testing.
- I will perform a manual unit test, one phase of regression test on each module. After testing these phases I will start integration test for the whole application.

12.11 Staffing and Training Needs

In my testing phase, previously mention that I will use BUG track as bug reporting software. But I don't know anything about this tool. So I will take a short training on this tool before starting testing.

12.12 Responsibilities

Responsibility for different functionality of the project-

Number	**Functionality**	**Responsible person**	**Roll**
01	Authentication	Shubho Chandro Roy	BSSE0619
02	User profile	Shubho Chandro Roy	BSSE0619
03	Messaging	Shubho Chandro Roy	BSSE0619
04	Video calling	Shubho Chandro Roy	BSSE0619
05	Recommendation	Shubho Chandro Roy	BSSE0619

12.13 Risks and Contingencies

- After development, testing will start. If development finish lately, there is a risk of late delivery of the software.
- In every software project there is a chance of changing of modification on the requirement. If this happens I may need to redesign or modify my plan and test cases.
- There is a risk of getting original data to test.

Requirements definition is completed. If the requirements change further, the following actions will be taken:

- As I am promised to deliver my product in due time, if requirement changes I will reschedule my working period and enhance my working time to complete my testing.

- I may reduce the number of performed test.
- Number of acceptable defects may be increased. These defects will be resolved in further release.
- The scope of the plan may be changed.

12.14 Testing Cost

As this is an academic project, so no need to estimate any testing cost.

12.15 Approval

Name	Role	Approver/Reviewer	Approval/Review Date
Nadia Nahar	Project Supervisor		

12.16 Test Cases

Test case: 1

Test case name: Registration

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on “Registration”	Registration page should appeared	
2	Verify that user select user role	User should get Doctor or patient as user role	
3	Verify that user click on “Next” button	Registration page should appeared based on user role	
4	Verify that user give his/her first name, last name, user name	User can’t registration without username	
5	Verify that user give valid email id	Email id contains “@___.com” extension	
6	Verify that user give invalid email id	An error “Ask for valid email id” message show	
7	Verify that user create password and confirm password	User should give two matched password and confirm password	
8	Verify that user create password and confirm password with two unmatched number	An error “Ask for valid confirm password” message show	

Registration as "Doctor"

9	Verify that user give designation, specialty, college/university name	Doctor can't registration except designation, specialty, college/university name	
10	Verify that doctor select hospital	Hospital list will be appeared	

Registration as "Patient"

10	Verify that user give age, height, weight, blood group, blood pressure, address	Patient can't registration without age, height, weight, blood pressure, address	

11	Verify that clicking on "Registration" button	"Registration successful" message show	

Test case: 2

Test case name: Login

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on "Login"	Login page should appeared	
2	Verify that user give valid username and password and click on "Login" button	Successfully login into the system	
3	Verify that User give null value of username and password and click on "Login" button	An error message show	
4	Verify that user give invalid username and password	"Ask for valid username and password" message show	

Test case: 3

Test case name: Logout

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on "Logout"	successfully logout	

Test case: 4

Test case name: Update profile

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on “Update Profile”	Update profile page should appeared	
2	Verify that user give update information for update profile	“Update profile successful” message show	

Test case: 5

Test case name: View profile

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on “View Profile”	Successfully appeared doctor’s profile page	

Test case: 6

Test case name: Messaging

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on “message” icon	Message page should required	
2	Verify that user enter message and clicking on “Send” button	“Sent message successfully” message show	
3	Verify that user get message notification on “Notification” icon	User should get message notification on “Notification” icon	
4	Verify that user click on “Notification” icon	Notification message should appeared	
5	Verify that user click on message	Send message page should appeared	

Test case: 7

Test case name: Video calling

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that clicking on “video conference” icon	Video calling page should appeared	
2	Verify that user call for video calling	User should get video call	
3	Verify that user click on “Answer Call” icon	Video calling started successfully	

Test case: 8

Test case name: Recommendation

Test Steps	Description	Expected Result	Pass/Fail
1	Verify that user click on recommendation	Recommendation page should appeared successfully	
2	Verify that user enter problem for recommendation	User should get recommended hospital name	

Chapter 13
Reference

1. Pressman, Roger S. Software Engineering: A Practitioner's Approach (7th ed.). Boston, Mass: McGraw-Hill. ISBN 0-07-285318-2
2. Database System Concepts, 5th Ed. ©Silberschatz, Korth and Sudarshan
3. Sommerville, I. Software Engineering, 7th ed. Harlow, UK: Addison Wesley, 2006
4. http://ecomputernotes.com/software-engineering/component-level-design
5. http://ecomputernotes.com/software-engineering/architecturaldesign
6. http://www.fit.vutbr.cz/study/courses/ITS/public/ieee829.html