**HARAMAYA UNIVERSITY**

**

**College of Computing and Informatics**

**Department of Information Systems**

**Project Report**

**Fayyaan Faaya Online Medical Consultationand Subscription System**

**By**

**Group Members**

1. Gadisa Teka (1639/10)
2. Habtemu Yetayo (1642/10)
3. Jibril Gela (1648/10)
4. Sena Kenea (1662/10)

Haramaya, Oromia

2021 E.C

**HARAMAYA UNIVERSITY**

***College of Computing and Informatics***

**Department of Information Systems**

**CERTIFICATE**

This is to certify that the work embodies in this project entitled **“**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**”** being submitted by following students for partial fulfillment of the requirement to “Haramaya University” during the academic year 2020-21 is a record of bonafide piece of work, carried out by them under my supervision and guidance in the **“**Department of Information Systems**”, College of Computing and Informatics.**

1. **Gadisa Teka (1639/10)**
2. **Habtemu Yetayo (1642/10)**
3. **Jibril Gela (1648/10)**
4. **Sena Kenea (1662/10)**

**ADVISER BY: CO ADVISER BY:**

**Mr. Beyene Bedasa Mr. Tariku D.**

(Lecture….) (Lecture….)

**HEAD OF DEPARTMENT**

**Mr. Beyene Bedasa**

(Lecture….)

**HARAMAYA UNIVERSITY**

***College of Computing and Informatics***

**Department of Information Systems**

**CERTIFICATE OF APPROVAL**

The project report entitled **“\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**”being submitted by following students has been examined by me/us and is hereby approved for project work which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project report only for the purpose for which it has been submitted.

1. **Gadisa Teka (1639/10)**
2. **Habtemu Yetayo (1642/10)**
3. **Jibril Gela (1648/10)**
4. **Sena Kenea (1662/10)**

**APPROVED & ADVISOR/ CO ADVISER BY:**

**Mr. Beyena Bedasa Mr. Tariku D.**

(Lecture….) (Lecture….)

**Acknowledgement**

First of all, we would like to praise almighty God who helps usto accomplish this documentation successfully. Next, my deepest gratitude goes to my advisor **Mr. Beyena Bedasa** for his excellent advice and active guidance throughout this project. And we also would like to express our gratitude towards our coordinator **Mr. Abduraman Redi** and department staff and classmates for their kind cooperation and encouragement which help us in completion of our project.

We will fail in our duty if we are not thanking Haramaya university department of Information Systems for their acquiescence of laboratory class, free internet service that make us to accomplish our project. Lastly Special thanks to the people who provided insightful comments and questions that led to significant improvements in this document.

**Table of Contents**

[**List of Figure** IV](#_Toc80392463)

[**List of Tables** V](#_Toc80392464)

[**Lists of Abbreviations** VI](#_Toc80392465)

[**Abstract** VII](#_Toc80392466)

[**CHAPTER ONE** 1](#_Toc80392467)

[**1.** **Introduction** 1](#_Toc80392468)

[**1.1.** **Background of Fayyaan Faaya Online Medical Consultation and Subscription System** 1](#_Toc80392469)

[**1.2.** **Statement of the Project Problem** 2](#_Toc80392470)

[**1.3.** **Objectives of the Project** 2](#_Toc80392471)

[1.3.1. General Objective 2](#_Toc80392472)

[**1.3.2.** **Specific Objective** 2](#_Toc80392473)

[**1.4.** **Methodology of the Project** 2](#_Toc80392474)

[1.4.1. Fact Finding Techniques 2](#_Toc80392475)

[1.4.2. System Conceptualization and Formalization Techniques 3](#_Toc80392476)

[**1.5.** **Scope of the Project** 3](#_Toc80392477)

[**1.6.** **Significance of the Project** 4](#_Toc80392478)

[**1.7.** **Feasibility Assessment** 4](#_Toc80392479)

[1.7.1. Economic Feasibility 4](#_Toc80392480)

[1.7.2. Technical Feasibility 5](#_Toc80392481)

[1.7.3. Operational Feasibility 5](#_Toc80392482)

[1.7.4. Schedule Feasibility 6](#_Toc80392483)

[**1.8.** **Communication Plan** 7](#_Toc80392484)

[1.8.1. Team Configuration and Management 7](#_Toc80392485)

[1.8.2. Communication Plan 7](#_Toc80392486)

[**CHAPTER TWO** 8](#_Toc80392487)

[**2.** **Modeling and Prototype** 8](#_Toc80392488)

[**2.1.** **Overview of Existing System** 8](#_Toc80392489)

[**2.2.** **Overview of Proposed System** 8](#_Toc80392490)

[**2.3.** **System Requirement Specification** 8](#_Toc80392491)

[2.3.1. Functional Requirements 9](#_Toc80392492)

[2.3.2. Non-Functional Requirements 9](#_Toc80392493)

[**2.4.** **Class Responsibility Collaboration (CRC)** 10](#_Toc80392494)

[**2.5.** **Use Case Modeling** 13](#_Toc80392495)

[2.5.1. Essential Use Case Modeling 13](#_Toc80392496)

[2.5.2. Essential Use Case Description 14](#_Toc80392497)

[2.5.3. System Use Case Diagram 17](#_Toc80392498)

[2.5.4. System Use Case Description 18](#_Toc80392499)

[2.5.5. Features 23](#_Toc80392500)

[**2.6.** **User Interface Prototype** 23](#_Toc80392501)

[**2.7.** **User-Interface Flow Diagramming** 24](#_Toc80392502)

[**2.8.** **Supplementary Specification** 25](#_Toc80392503)

[2.8.1. Business Rule 25](#_Toc80392504)

[2.8.2. Constraints 26](#_Toc80392505)

[**Chapter Three** 27](#_Toc80392506)

[**3.** **System Design Document** 27](#_Toc80392507)

[**3.1.** **Layered Class Type Architecture** 27](#_Toc80392508)

[**3.2.** **Class Diagram Modeling** 28](#_Toc80392509)

[**3.3.** **Sequence Diagram** 29](#_Toc80392510)

[**3.4.** **Activity Diagram** 32](#_Toc80392511)

[**3.5.** **User Interface Design** 35](#_Toc80392512)

[3.5.1. Form Diagram 35](#_Toc80392513)

[3.5.2. Dialogue and Interface Design 36](#_Toc80392514)

[**3.6.** **State Chart Diagram** 38](#_Toc80392515)

[**3.7.** **Object Diagram** 40](#_Toc80392516)

[**3.8.** **Persistent Modeling** 41](#_Toc80392517)

[**3.9.** **Component Diagram** 49](#_Toc80392518)

[**3.10.** **Deployment Diagram** 50](#_Toc80392519)

[**References** 51](#_Toc80392520)

# **List of Figure**

[Figure 1. Gantt chart that showing schedule feasibility 6](#_Toc80392161)

[Figure 2. CRC diagram for actor class 10](#_Toc80392162)

[Figure 3. CRC diagram for User Interface class 11](#_Toc80392163)

[Figure 4. CRC diagram for business class 12](#_Toc80392164)

[Figure 5. Essential Use Case Diagram 13](#_Toc80392165)

[Figure 6. System Use Case Diagram 17](#_Toc80392166)

[Figure 7. Traditional User Interface Prototyping 23](#_Toc80392167)

[Figure 8. Proposed System User Interface Flow Diagram 24](#_Toc80392168)

[Figure 9. Layered Class Type Architecture 27](#_Toc80392169)

[Figure 10. Class Diagram of Fayyaan Faaya OMCSS 28](#_Toc80392170)

[Figure 11. Sequence Diagram for System Users Login expect Information Seeker. 29](#_Toc80392171)

[Figure 12. Sequence Diagram for Getting Consultation 30](#_Toc80392172)

[Figure 13. Sequence Diagram for Assigning Doctor 31](#_Toc80392173)

[Figure 14. Activity Diagram for Creating Account 32](#_Toc80392174)

[Figure 15. Activity Diagram for Users Login expect for Information Seeker 32](#_Toc80392175)

[Figure 17. Activity Diagram for Checking Symptom 33](#_Toc80392176)

[Figure 18. Activity Diagram for Getting Consultation 33](#_Toc80392177)

[Figure 19. Activity Diagram for Adding Health Info 34](#_Toc80392178)

[Figure 20. User Interface Design for Home Page 35](#_Toc80392179)

[Figure 21. Report Design for System Admin Registration Page 36](#_Toc80392180)

[Figure 22. Dialogue and Interface Design for Login Page 37](#_Toc80392181)

[Figure 23. Dialogue and Interface Design for Searching 37](#_Toc80392182)

[Figure 24. State Chart for Login 38](#_Toc80392183)

[Figure 25. State Chart Diagram for Checking Symptom 38](#_Toc80392184)

[Figure 26. State Chart Diagram for Getting Consultation 39](#_Toc80392185)

[Figure 27. State Chart Diagram for Activating Account 39](#_Toc80392186)

[Figure 28. Object Diagram of Fayyaan Faaya OMCSS 40](#_Toc80392187)

[Figure 29. Persistent Diagram Fayyaan Faaya OMCSS 48](#_Toc80392188)

[Figure 30. Component Diagram Fayyaan Faaya OMCSS 49](#_Toc80392189)

[Figure 31. Deployment Diagram for Fayyaan Faaya OMCSS 50](#_Toc80392190)

# **List of Tables**

[Table 1. Table showing tangible benefits of a system 5](#_Toc80392027)

[Table 2. Description of role for team member 7](#_Toc80392028)

[Table 3. Essential Use Case Description for Getting Consultation 14](#_Toc80392029)

[Table 4. Essential Use Case Description for Checking Symptom 14](#_Toc80392030)

[Table 5. Essential Use Case Description for Giving Consultation 15](#_Toc80392031)

[Table 6. Essential Use Case for Adding Health Info 15](#_Toc80392032)

[Table 7. Essential Use Case Description for Assigning Doctor 16](#_Toc80392033)

[Table 8. System Use Case Description for Users Login 18](#_Toc80392034)

[Table 9. System Use Case Description for Getting Consultation 19](#_Toc80392035)

[Table 10. System Use Case Description for Finding Care Services 20](#_Toc80392036)

[Table 11. System Use Case Description for Adding Health Info 21](#_Toc80392037)

[Table 12. System Use Case Description for Activating Account 21](#_Toc80392038)

[Table 13. System Use Case Description for Assigning Doctor 22](#_Toc80392039)

[Table 14. User Account Database Table 41](#_Toc80392040)

[Table 15. Doctor Account Database Table 42](#_Toc80392041)

[Table 16. System Admin Account Database Table 42](#_Toc80392042)

[Table 17. Patient Account Database Table 43](#_Toc80392043)

[Table 18. Subscription Database Table 43](#_Toc80392044)

[Table 19. Consultation Database Table 44](#_Toc80392045)

[Table 20. Care Services Database Table 44](#_Toc80392046)

[Table 21. Feedback Database Table 44](#_Toc80392047)

# **Lists of Abbreviations**

BR = Business Rule

CRC = Class Responsible Collaboration

CSS = Cascading Style Sheet

EUS = Essential Use Case

HTML = Hypertext Markup Language

OMCSS = Online Medical Consultation and Subscription

OO = Object Oriented

PostgreSQL = Postgres Structured Query Language

SD =Sequence diagram

SRS = System Requirement Specification

SUC = System Use Case

UI = User Interface

UML = Unified Modeling Language

# **Abstract**

Having a better health is a key to an individual happiness and a country’s development. Hence, health is not merely the absence of disease, but it’s the attainment of state of physical, mental, emotional and social wellbeing. Healthcare is the diagnosis, treatment, and prevention of disease, illness, injury, and other physical and mental impairments in human beings.

The study so called **Fayyaan Faaya Online Medical Consultation and Subscription System** is web-based application that focuses on healthcare services and subscription system. The main purpose of the study was to develop a web-based healthcare system that allows our communities to have access of healthcare related information through internet access without physically visiting the healthcare centers.

The system is aims to design and implement an expertise system which will allow patients to have an access of healthcare related information, online consultation service either via chatbot or video technology with doctor, and medical diagnosis. It is also going to designed in a way of enabling patient to have find care-find doctor service via subscription medicine.

We develop the behavioral nature of our system using UML behavioral diagram such as sequence diagram, activity diagram and state chart diagram. And the static nature of the system by using the static UML diagram such as class diagram, object diagram, collaboration and deployment diagram.

# **CHAPTER ONE**

# **Introduction**

## **Background of Fayyaan Faaya Online Medical Consultation and Subscription System**

Having a better health is a key to an individual happiness and a country’s development. Hence, health is not merely the absence of disease, but it’s the attainment of state of physical, mental, emotional and social wellbeing. And so, healthcare is the diagnosis, treatment, and prevention of disease, illness, injury, and other physical and mental impairments in human beings. [1]

In Ethiopia healthcare service have very limited automation system with respect to providing healthcare related information and consultation service for communities. Most of the information those related to healthcare services are carried out in manual way. The manual way of providing healthcare and consultation service was time and resource consuming, and it requires physically being at healthcare centers. Due to this reason, there must be a need of an automate and expertise healthcare system which beneficiary for patients, doctors and healthcare providers.

That is the reason why we are initiated to develop the system so called Fayyaan Faaya Online Medical Consultation and Subscription System to allow our communities to have access of healthcare related information through internet access without physically visiting the healthcare centers.

Our system aims to design and implement an expertise system which will allow patients to have an access of healthcare related information, online consultation service either via chatbot or video technology, and medical diagnosis. It is also going to designed in a way of enabling patient to have find care-find doctor service via subscription medicine.

Generally, Fayyaan Faaya Online Medical Consultation and Subscription System will be developed to make both patients and doctors benefit able in accordance of healthcare services.

## **Statement of the Project Problem**

As the world we are living in is becoming to virtual world, different country community services are also becoming too virtual including public healthcare institutions. But, in Ethiopia many public services including healthcare services are operate through manual system in order to provide all of its services. Due to lack of automated and expertise healthcare system, our country healthcare system is currently facing the following problems: -

* There is no healthcare related information provider system.
* There is no online medical consultation, basic medical diagnosis and treatment system.
* In addition to the above problems, there is no way in which the doctors get payment from consulting patient via online rather working at healthcare centers.

## **Objectives of the Project**

### General Objective

The main objective of our project is to develop automated medical consultation and subscription system.

## **Specific Objective**

The specific objectives of the project are: -

* To allow communities to access healthcare related information via online.
* To develop and facilitate questionnaire based online medical consultations.
* To develop online symptom checker system where the patient enters their symptoms.
* To develop find doctor service in which patients are consulted by doctor via online video using video technologies apps.
* To inform the newly emerging diseases with its prevention and recovery methods.

## **Methodology of the Project**

The methodology that we used for identification of problems and understanding the existing system workflow involves: -

### Fact Finding Techniques

We, as a group use and follow a series of information acquisition methodologies by becoming a Knowledge Engineers for collecting precise and reliable information from different source of information. The following knowledge acquiring techniques are used for fact finding.

1. **Interview: -** to acquire information for construction of knowledge base and to draw conclusion from inference engine about how medical consultation system was applied with respect to information about diseases and its treatment, we interviewed and make phone calls to some domain experts/physician of healthcare institutions. The physicians that we interviewed are also gave us some books of medicals and assist us to different websites for further information.
2. **Direct Observation: -** we also observed that by existing at healthcare centers in which many patients are get serious conditions that may lead them to death due to lack of medical information, especially those come from remote areas.
3. **Document Review:** to get more information about the different diseases, and its treatment methods, we have reviewed different websites. The symptoms of the disease are collected from kaggle.com and other different health related websites with .csv extension. This csv file contain 4921 rows of record of the patients with their symptoms (132 types of different symptoms) and their corresponding disease(41 class of general disease) .
4. **Training Model using Supervised Learning**

Supervised learning is machine learning which works on classification, which means there is known label that you want the system to generate. Hence, classification is the problem identification to which set of categories anew observation is belongs. Based on the training set of data containing observation. There are different classifier algorithms; from these we use two classifier algorithms which are Navie Bayes with Multinomial Navie Bayes algorithm based on Baye’s theorem for predicting disease and Decision Tree for chatbot and disease prediction based on conditions.

**Navie Bayes: -** is simple probabilistic classifier based on applying Baye’s theorem with strong(navie) independence assumption between the features.

**P(D|S) = P(S|D) P(D)**

**P(S)**

**Decision Tree :** - is graphical representation of all possible solution to decision.

It works based on some conditions.

### System Conceptualization and Formalization Techniques

An initial discussion between the knowledge engineer and the domain expert helped to clarify the scope of the system, its basic features in more detail, and to identify a sub-problem that could be implemented relatively quickly, in order to obtain feedback from the expert as the feasibility and desirability of the system. [2]

## **Scope of the Project**

As we are developing a system, we identified the scope of our project that determines what will be included or not included in, based on the triplet project condition (i.e., scope, time and cost).

**Scope In: -** The system that will go to develop should include the following activities:

* Everyone has the access to healthcare related information and symptom checking service
* Patients are able create account to get medical consultation either via healthcare Chabot or video technology and medical diagnosis
* Patients are able to get find care-find doctor service through subscription
* Administrator are able to create an account and assign doctor
* Doctor are able give find care-find doctor service for his/her subscribers
* Doctor are able to add new healthcare related information
* Administrator are able to post news and manage all accounts

**Scope Out**

* The system is not expected to order drugs and medicine for patients
* The system is not going to in consist face to face medical consultation
* We are not going to develop the complete payment system and Banking system.
* The system did not support all local languages like Amharic, Tigrigna, Harari and etc.it only works in English and rarely in Afaan Oromo.
* The system is not expected to make medical appointment for patient.

## **Significance of the Project**

Fayyaan Faaya Online Medical Consultation and Subscription system will play a crucial role in giving reliable and timely healthcare service communities including for both doctor and patients.

The significance of the system includes: -

**For everyone: -**

* Provide healthcare related information on healthcare topics
* Allow them to check symptoms of their disease

**For Patients**

* Enable them to check their disease symptom
* Give brief clue of health topics
* Enable them to get Find Care - Find Doctor service

**For Doctors**

* Enable them to get income from medical subscription.
* Reduce manual work overload of staff during consultation

## **Feasibility Assessment**

Feasibility assessment is a mechanism to determine whether the project is feasible to be developed and implemented or not, by considering various factor such as economic, technical, operational, schedule feasibilities.[2]

### Economic Feasibility

The system to be developed is economically feasible and the benefit is outweighing the cost because it highly uses materials from internet. Generally, the system that we develop is brought a number of tangible and intangible benefits.

**Tangible benefits: -**

Table 1. Table showing tangible benefits of a system

|  |  |  |
| --- | --- | --- |
| **Description** | **Quantity** | **Cost in ETB (birr)** |
| Paper | 120 | 80 |
| Pen/pencil | 4 | 40 |
| Flash Disk | 2 | 500 |
| Web and database server | **\_** | Free |
| MS word | **\_** | Free |

**Intangible benefits: -**

* Increase in satisfaction for both patients and doctors
* Give better and efficient services
* Faster response for each activity
* Provide efficient information processing

Implementing the system on both computer and mobile devices is easy and does not require more resources, because the system is compatible with current devices requirements. So that the new system will not cost the healthcare provider and system user for further maintenances and resources.

### Technical Feasibility

While developing this system, the tools that we use are computer, networking devices, database server, programming language (i.e. Java script, HTML, Bootstrap, CSS and Jquery for Frontend development and Django(Python based web framework), and PostgreSQL database with Pgadmin tool for backend development. In today’s world, these products are easily available and afforded to users. So, this can lead us to say the proposed system is technical feasible.

### Operational Feasibility

Our proposed system is trying to solve and overcome the problems those are outlined in the statement of the problem in order to achieve specified objectives. Hence, when the developed system is user friendly, and interactive with the environment, it will not have any difficulties to operate the system. So, the system is operationally feasible.

### Schedule Feasibility

In schedule feasibility the major activities are described with time-frame and the description chart is drawn by using Gantt chart. We hope that we have finalized our system before the end of submission date.

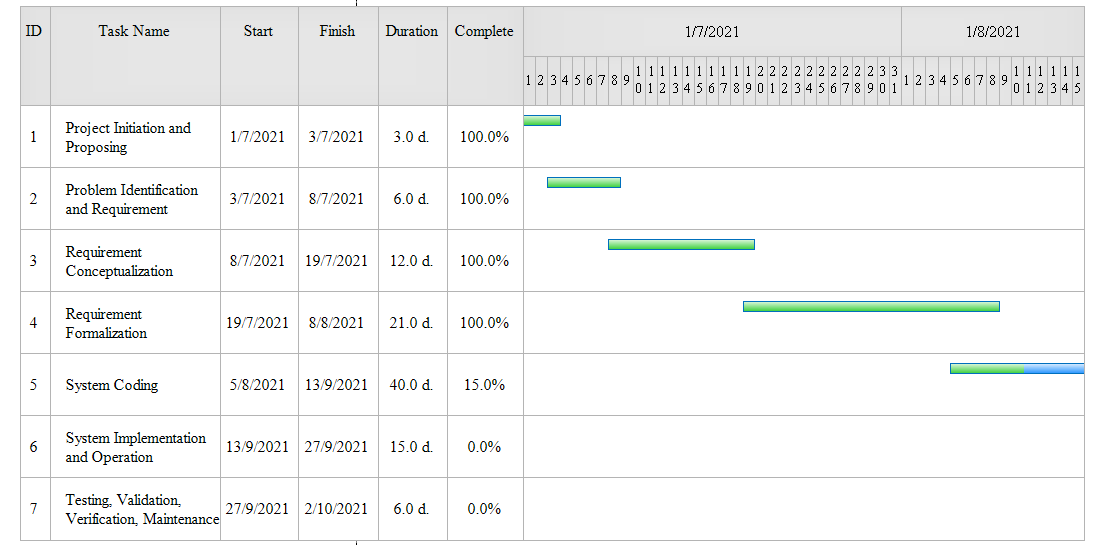


Figure 1. Gantt chart that showing schedule feasibility

## **Communication Plan**

### Team Configuration and Management

As a team, we all are playing our own roles in effective and efficient manner to complete the proposed project timely. Here is the table which shows roles of our team members:

Table 2. Description of role for team member

|  |  |
| --- | --- |
| **NAME** | **ROLE** |
| Gadisa Teka | Project manager/Team Leader |
| Habtemu Yetayo | Problem Identifier and Data gather |
| Sena Kenea |
| Jibril Gela | System Formalizer (Designer) |
| Gadisa Teka |
| Sena Kenea | System conceptualization/Analyst |
| Habtemu Yetayo |
| Gadisa Teka |
| Jibril Gela | Web Content Designer and Coding |
| Gadisa Teka |
| All team members | Implementation, Testing, Validation, Verification, and Maintenance |

### Communication Plan

As we are working on our project, we are using the following techniques in order to meet one another and make discussion on project progresses. We communicate through mobile phone, Face book, face-to-face communication, telegram and also some other social media techniques to communicate each other. Also, we meet to our advisor to get guidance and discuss about overall activities of our project via face-to-face communication, email, and mobile phone.

* 1. Limitation
* The system requires network connection.
* The Person who accesses the web system should have basic knowledge of computer.
* In our system Chatbot is work only command line interface ( this happen due short age of time.)
* The system is basically applicable only in English language and rarely in Afaan Oromoo

# **CHAPTER TWO**

# **Modeling and Prototype**

## **Overview of Existing System**

The current Ethiopian health policy has been in existence since 1993. “It emphasizes the importance of achieving access to a basic package of quality primary health care services by all segments of the population, using the decentralized state of governance. The health policy stipulates that the health services should include preventive, primitive, and curative components**”**[3]

In the existing healthcare system, it’s not possible to access healthcare related information, get consultation and check symptom without physically existing at hospitals/healthcare centers. It’s going on through the ways of delivering healthcare service by directly available at hospitals. So, the existing healthcare service was consuming time, resource and requires physically available at hospitals.

## **Overview of Proposed System**

The proposed system mainly focuses on providing healthcare related information and facilitating medical consultation via online. It also allows medical diagnosis, medical consultation, and medical subscription for find care-find doctor service.

In addition to these, the proposed system enables and increases the public access medical information, the demand of healthcare for IT solution, and it can enhance the experience of healthcare for patients and support general practice in utilizing time and money, improving both access and sustainability.

## **System Requirement Specification**

System requirements specification of this document is fully describing what the expected behavior of our system is. It states the functional and non-functional requirements as well as capabilities our system that needs to provide, and as well as the constraints that it must respect. It also provides the basis for all subsequent of project conceptualization, formalization, coding, implementation, and testing.

### Functional Requirements

Functional requirements of our system are defining the fundamental actions of system that must be performed. It describes functionality or services of system that should be provided, how the system should react to particular inputs, and how the system should behave in particular situations.

* The system provides healthcare related or medical information and symptom checker service to everyone who access it
* It facilitates an organized storage of medical information, which is used to provide medical information
* Patients are able to access medical information, get medical diagnosis, get text-based consultation, and view news
* Patients are able to create an account
* They are able to get find care-find doctor service, video consultation
* They are able get find care find doctor service by communicating with Doctors
* Administrator should able to create and manage account and all over system activities
* Administrator are able to manage feedback and post news related to health services
* Doctors are able provide consultation and find care-find doctor services

### Non-Functional Requirements

As the name indicates, non-functional requirements are not directly concerned with the system services that delivered to its users. They are related to emergent system properties such as reliability, response time, and store occupancy.

**Usability:** the system is easy to use for both users and all service providers with some basic knowledge of computer skill and training.

**Security:** the system can secure its users information.

**User Interface:** the user can interact with the system due to system’s user interface simplicity and fully responsiveness to communicate with its users.

**Reliability:** the system is trustworthy to be used by information seekers, patients, administrator, and doctors consistently.

**Performance:** the system can perform and execute basic commands of users without needs of much effort.

**Availability:** Users have access to the system at any time whenever they have internet access.

**Maintainability:** the system is easily modifiable due to its methodology is made by agile methodology that allows continuous modification of the system.

**Error Handling:** the system can handle error by displaying error message whenever users enter incorrect information to the system.

## **Class Responsibility Collaboration (CRC)**

Class responsibility collaboration diagram (CRC) shows the classes of the system, their interrelationships, and the operations and attributes of the classes.[3]

* **Actor Classes: -** the actors those appears in use case

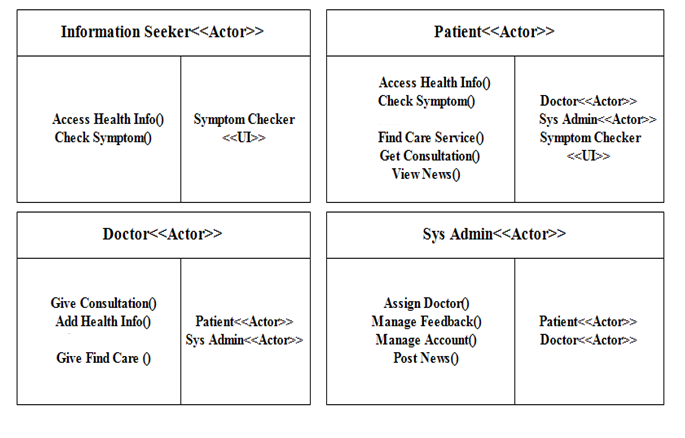


Figure 2. CRC diagram for actor class

* **UI Classes: -** screens, homepages, menus, reports, and sometimes database.

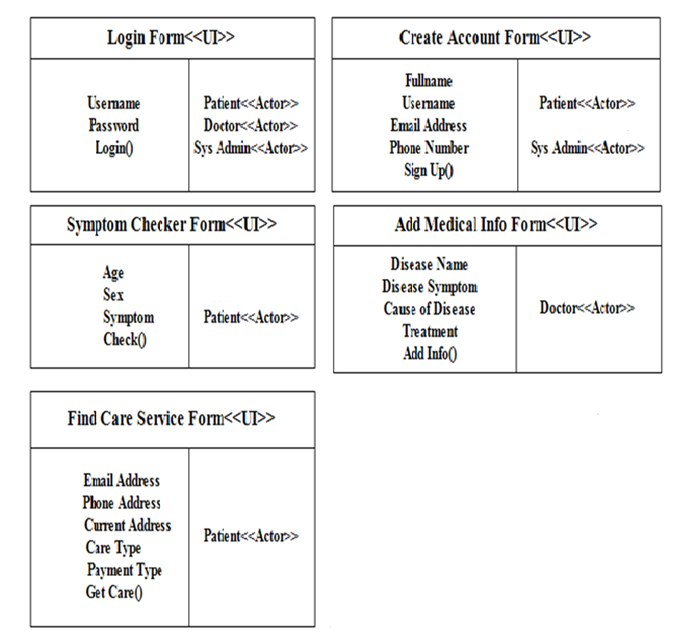


Figure 3. CRC diagram for User Interface class

* **Business Classes: -** places, things, concepts, and events that describe what the business is all about.

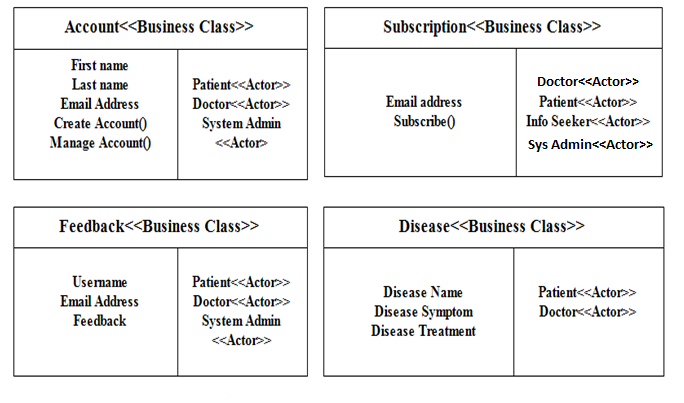
****

Figure 4. CRC diagram for business class

## **Use Case Modeling**

Use case diagrams are used to depict graphically the interactions between the system and system users. In other words, they graphically describe who will use the system and in what ways the user, expect to textually describe the sequence of steps of each interaction.

### Essential Use Case Modeling

Essential use case modelling is a simplified abstract, generalized use case that captures theintentions of the users in a technology and implementation independent manner.

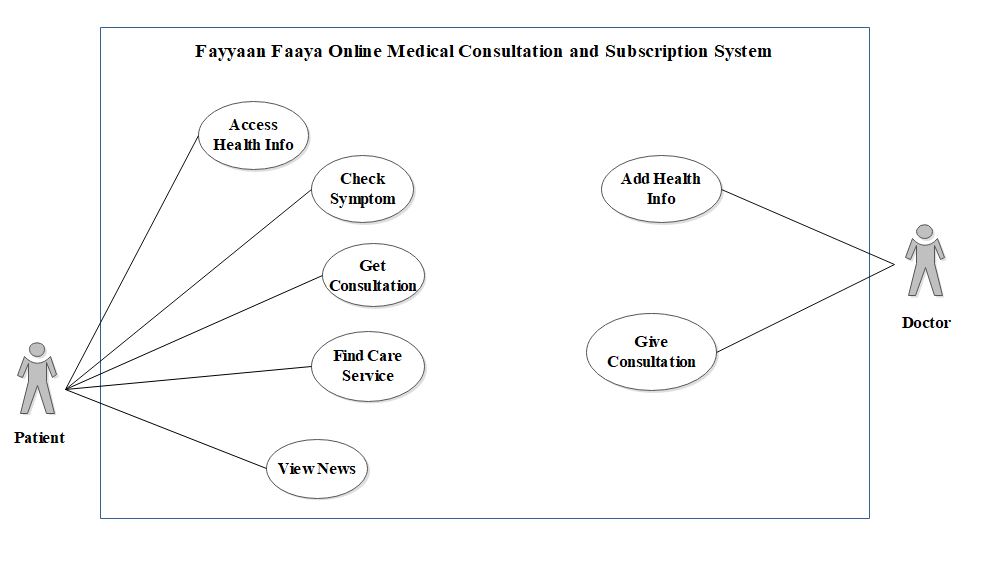
****

Figure 5. Essential Use Case Diagram

### Essential Use Case Description

Table 3. Essential Use Case Description for Getting Consultation

|  |  |
| --- | --- |
| **Use Case ID** | EUC-01 |
| **Use Case Name** | Get Consultation |
| **Actor** | Patient |
| **Description** | Patient got consultation service for their conditions |
| **Preconditions** | The doctor must available for consultation application |
| **Basic Course of Action** | 1. Patient fills consultation service form 2. Doctor check the application, and ask for payment if it’s the application is payment required consultation 3. Doctor give consultation 4. Use case end |
| **Post Conditions** | Patient got consultation service |

Table 4. Essential Use Case Description for Checking Symptom

|  |  |
| --- | --- |
| **Use Case ID** | EUC-02 |
| **Use Case Name** | Check Symptom |
| **Actor** | Information Seeker, Patient |
| **Description** | Patient check symptom for his/her disease condition |
| **Preconditions** | Patient must enter his/her symptom |
| **Basic Course of Action** | 1. Information Seeker/Patient enter his/her symptom 2. The disease condition and treatment are generated conditions based on the entered symptom 3. Use case end |
| **Post Conditions** | Symptom checked |

Table 5. Essential Use Case Description for Giving Consultation

|  |  |
| --- | --- |
| **Use Case ID** | EUC-03 |
| **Use Case Name** | Give Consultation |
| **Actor** | Doctor |
| **Description** | Doctor give consultation service for his/her patients |
| **Preconditions** | The patient must apply for consultation service |
| **Basic Course of Action** | 1. Patient applies for consultation 2. Admin approve the consultation application and assign Doctor 3. Doctor give consultation service 4. Use case end |
| **Post Conditions** | Consultation service is given |

Table 6. Essential Use Case for Adding Health Info

|  |  |
| --- | --- |
| **Use Case ID** | EUC-04 |
| **Use Case Name** | Add Health Info |
| **Actor** | Doctor |
| **Description** | Doctor add health-related information |
| **Preconditions** | The admin must approve the added information |
| **Basic Course of Action** | 1. Doctor add health-related information 2. Admin approve the information 3. Use case end |
| **Post Conditions** | Health-related information added |

Table 7. Essential Use Case Description for Assigning Doctor

|  |  |
| --- | --- |
| **Use Case ID** | EUC-05 |
| **Use Case Name** | Assign Doctor |
| **Actor** | Admin |
| **Description** | Admin add and assign doctor for service |
| **Preconditions** | The Doctor must apply for service |
| **Basic Course of Action** | 1. Doctor applies for service 2. Admin approve and assign the Doctor 3. Use case end |
| **Post Conditions** | Doctor Assigned |

### System Use Case Diagram

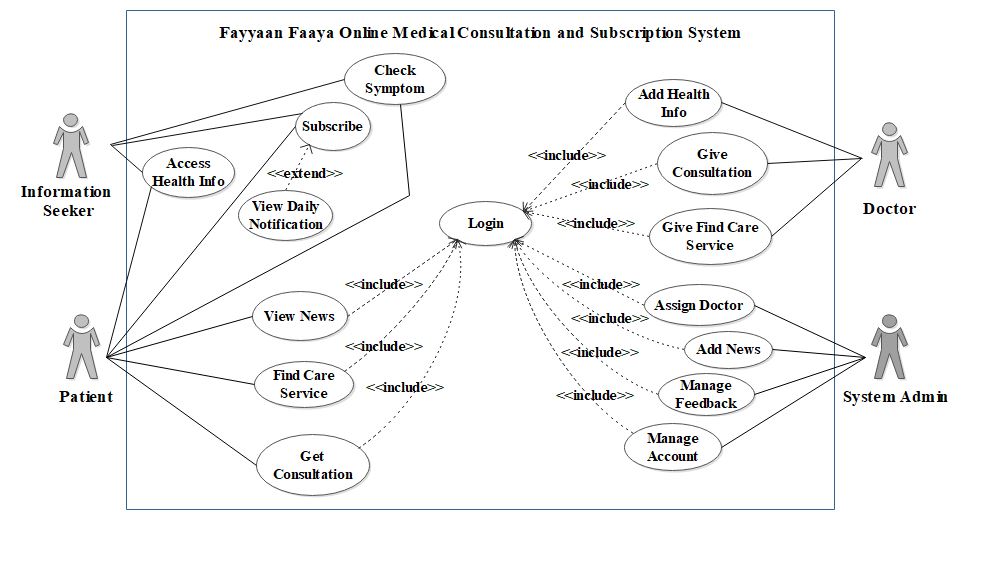
****

Figure 6. System Use Case Diagram

### System Use Case Description

Table 8. System Use Case Description for Users Login

|  |  |
| --- | --- |
| **Use Case ID** | SUC-01 |
| **Use Case Name** | Login |
| **Actor** | Patient, Doctor, Admin |
| **Description** | Users of the system must login to the system to perform an action |
| **Preconditions** | Users must have an account of the system |
| **Basic Course of Action** | 1. Users open the web browser 2. The system display: -    1. Home page with login button 3. Users click on the login button 4. System display login form 5. Users fill the login form correctly 6. Users click login button 7. System display: -    1. Homepage with personal feds for patient    2. Doctor dashboard for doctor    3. Admin dashboard for admin 8. Use case end |
| **Post Conditions** | Users logged into the system |
| **Alternative Course of Action** | A5: System verifies if the usersinsert username and password correctly, if not   1. System display error message 2. Use case direct: -    1. To return to step 5 or    2. To reset password form 3. Use case end |

Table 9. System Use Case Description for Getting Consultation

|  |  |
| --- | --- |
| **Use Case ID** | SUC-02 |
| **Use Case Name** | Get Consultation |
| **Actor** | Patient |
| **Description** | Patient get consultation |
| **Preconditions** | Patient must login to the system |
| **Basic Course of Action** | 1. Patient login to the system 2. Patient click on get consultation button 3. System displays two consultation option buttons either for free or paid    1. If patient click on free consultation: -       1. System display text-based consultation form       2. Patient enter his/her concern    2. Else if patient click on paid consultation: -       1. System display payment form       2. Patient fill the form       3. Patient pay his/her payment via online       4. System displays paid consultation service form with patient chooses option       5. Patient click options 4. The patient get consultation 5. Use case end. |
| **Post Conditions** | Patient get consultation service |
| **Alternative Course of Action** | A1: System verifies if the patient inserts username and password correctly, if not   1. System display error message 2. Use case return to step 1   A3.2. System checks if the patient is made payment for consultation, if not   1. System display “please first pay for service” text 2. Use case return to step 3 3. Use case end. |

Table 10. System Use Case Description for Finding Care Services

|  |  |
| --- | --- |
| **Use Case ID** | SUC-03 |
| **Use Case Name** | Find Care Service |
| **Actor** | Patient |
| **Description** | Patient allowed to have a find care medical subscription |
| **Preconditions** | Patient must login to the system. |
| **Basic Course of Action** | 1. Patient login to the system 2. Patient click on Find Care – Find Doctor button 3. System displays find care form 4. The patient fills all needed information 5. Patient choose his/her payment rate schedule 6. System displays all available find care services 7. Patient choose one or more find care service form listed 8. System provides the patient find care service schedule 9. Use case end. |
| **Post Conditions** | Patient get find care services |
| **Alternative Course of Action** | A1: System verifies if the patient inserts correct username and password, if not   1. System display error message 2. Use case return to step 1   A4. System verifies if the patient fills find care form correctly, if not   1. System display error message 2. Use case stays on find care service 3. Use case end |

Table 11. System Use Case Description for Adding Health Info

|  |  |
| --- | --- |
| **Use Case ID** | SUC-04 |
| **Use Case Name** | Add Health Info |
| **Actor** | Doctor |
| **Description** | Doctor add health-related information to database |
| **Preconditions** | Doctor must login to system |
| **Basic Course of Action** | 1. Doctor login to the system 2. System displays Doctor Dashboard 3. Doctor click on add health info 4. System display note area 5. Doctor fill health related information to knowledge 6. Use case end |
| **Post Conditions** | Information added to knowledge base |
| **Alternative Course of Action** | A1: System verifies if the doctor inserts username and password correctly, if not   1. System display error message 2. Use case return to step 1 3. Use case end |

Table 12. System Use Case Description for Activating Account

|  |  |
| --- | --- |
| **Use Case ID** | SUC-05 |
| **Use Case Name** | Activate Account |
| **Actor** | Admin |
| **Description** | System Admin Activate Users Account |
| **Preconditions** | Users must be deactivated before |
| **Basic Course of Action** | 1. Admin login to its system 2. System display admin dashboard 3. Admin click on manage account button 4. System display list of previously activated and deactivated users 5. Admin click on activate button respective to specific user 6. System display success message 7. Use case end |
| **Post Conditions** | User is activated |
| **Alternative Course of Action** | A1: System verifies if the admin inserts username and password correctly, if not   1. System display error message 2. Use case return to step 1 3. Use case end |

Table 13. System Use Case Description for Assigning Doctor

|  |  |
| --- | --- |
| **Use Case ID** | SUC-06 |
| **Use Case Name** | Assign Doctor |
| **Actor** | Admin |
| **Description** | System Admin assign doctor for consultation and preventive care service |
| **Preconditions** | Doctor should be available to be assigned |
| **Basic Course of Action** | 1. Admin of the system login to his account 2. System display Admin dashboard 3. Admin click assign doctor button 4. System display doctor assigning page 5. Admin has to write doctor name and id 6. Admin click assign doctor button 7. Use case end |
| **Post Conditions** | Doctor assigned |
| **Alternative Course of Action** | A5: System verifies if the admin inserts matching doctor name and id correctly, if not   1. System display error message 2. Use case return to step 5 3. Use case end |

### Features

The system has the following features: -

**Easily Accessible: -**the system is easy to access for all users.

**Interactivity: -**The system has responsive and organized interface for interactivity.

**Security:** - the system is able to keep patient information securely without third party involvement.

**Communication: -**Patient and assigned doctor can communicate with each other without any problem

## **User Interface Prototype**

The user interface prototype is an iterative analysis technique in which users are actively involved in the mocking-up of the UI for a system. User interface prototype is a low fidelity model or prototype of the user interface for system that represents the general ideas behind the user interface in a technology-independent manner, but not the exact details. [4]

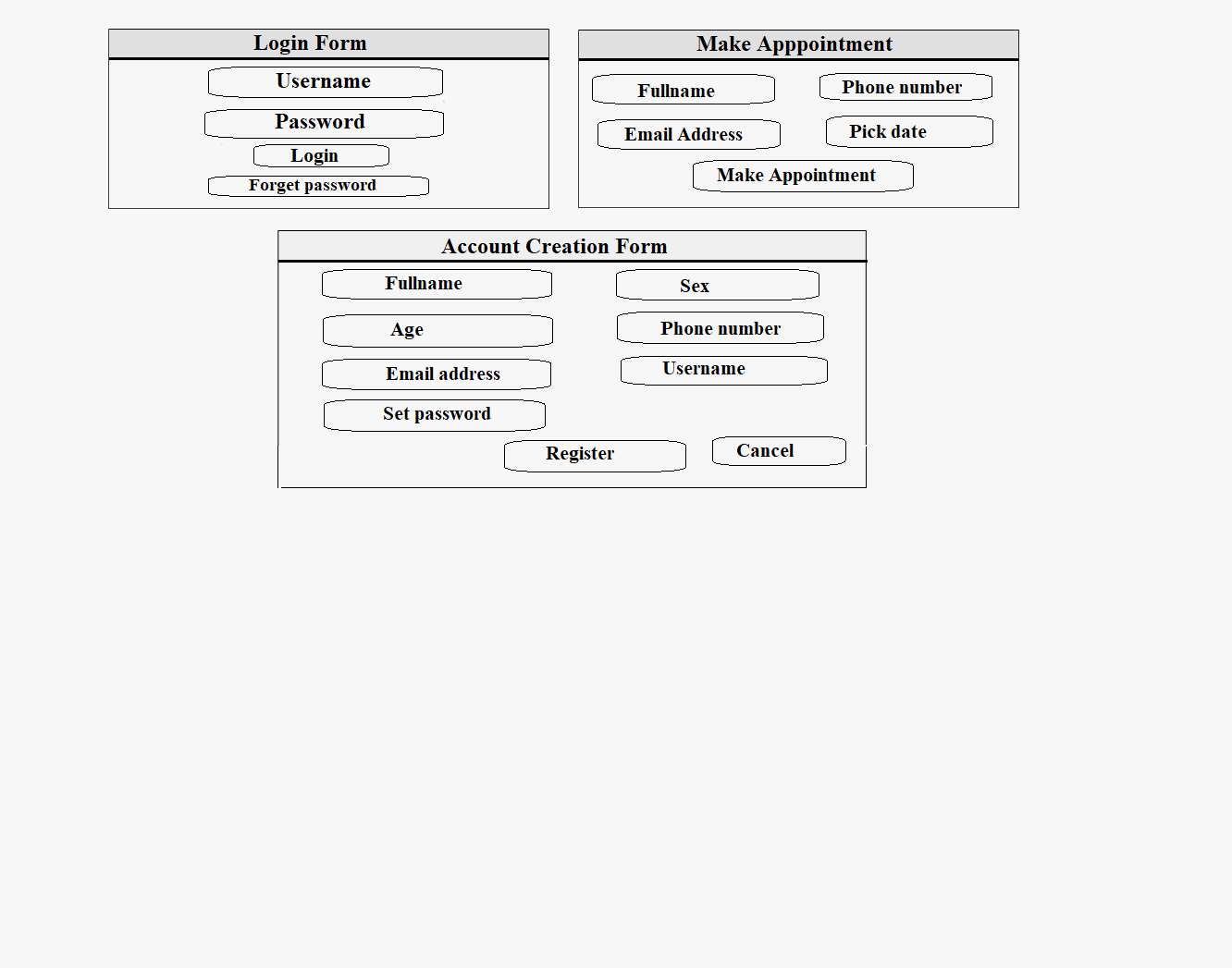
****

Figure 7. Traditional User Interface Prototyping

## **User-Interface Flow Diagramming**

UI prototypes are an excellent means of exploring your user interface, but unfortunately, it’s easy to quickly become bogged down in the details of the UI and not see the bigger picture. The boxes represent major user interface elements, modeled as you would instances and the arrows represent the possible flow between them, modeled as you would transition in activity diagrams.[5]

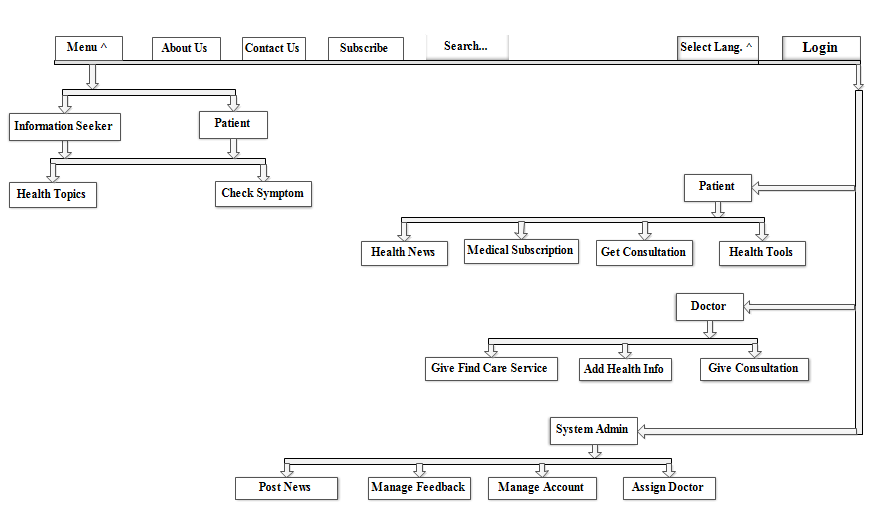


Figure 8. Proposed System User Interface Flow Diagram

## **Supplementary Specification**

### Business Rule

Business rule of proposed system

**Identifier:** BR#01

**Name:** Login to the system

**Description:** Users of the system such as doctor, and admin must login the system in order to perform healthcare activities.

**Identifier:** BR#02

**Name:** Validating users’ information

**Description:** The system validates the users to use the system if users’ information matches with system database information and knowledge base.

**Identifier:** BR#03

**Name:** System accessing

**Description:** Once the user’s login to the system, they can access system resource except for some exceptional reason.

**Identifier:** BR#04

**Name:** Consulting patient

**Description:** The system by itself provides medical consultation via Chabot and assigned doctor can also give medical consultations service.

**Identifier:** BR#05

**Name:** checking symptom

**Description:** The system check user/patient symptom based on entered symptoms if it’s matched with knowledge base.

**Identifier:** BR#06

**Name:** Making medical appointment date

**Description:** Patient made medical appointment and has to available on appointment date according to doctor schedule.

**Identifier:** BR#07

**Name:** Assigning doctor

**Description:** The admin of system assigns registered doctor for service according to their department.

**Identifier:** BR#08

**Name:** Patient create an account

**Description:** Patient can able to create an account in order to get more services of healthcare system

**Identifier:** BR#09

**Name:** Manage users’ account

**Description:** Admin can manage users account form that stated in the database

### Constraints

During the development of our project there are some constraints those face and challenge us. Among these the most common problem that constrains our project is the shortage of time that caused due to recently happen COVID-19 pandemic disease.

# **Chapter Three**

# **System Design Document**

## **Layered Class Type Architecture**

We originally use the term "class type" because we first started with this approach using object oriented (OO) modeling.

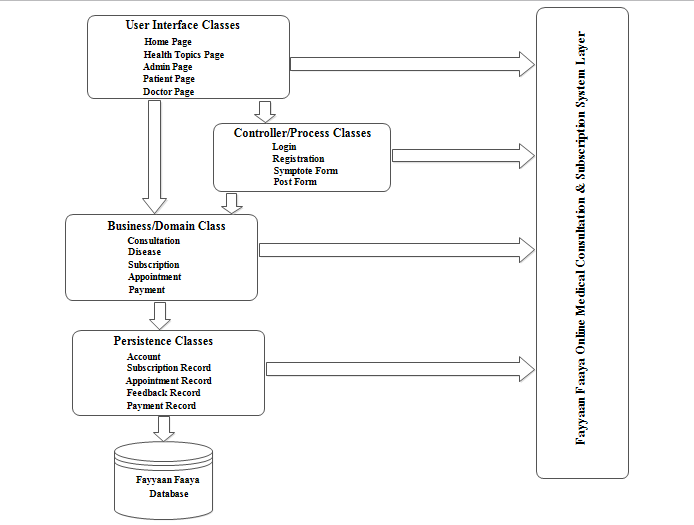


Figure 9. Layered Class Type Architecture

## **Class Diagram Modeling**

A class diagram is structural diagram which is used to describe the relationship between class of a system and their relationship. It shows a set of classes, interfaces, and collaborations with their relationships.

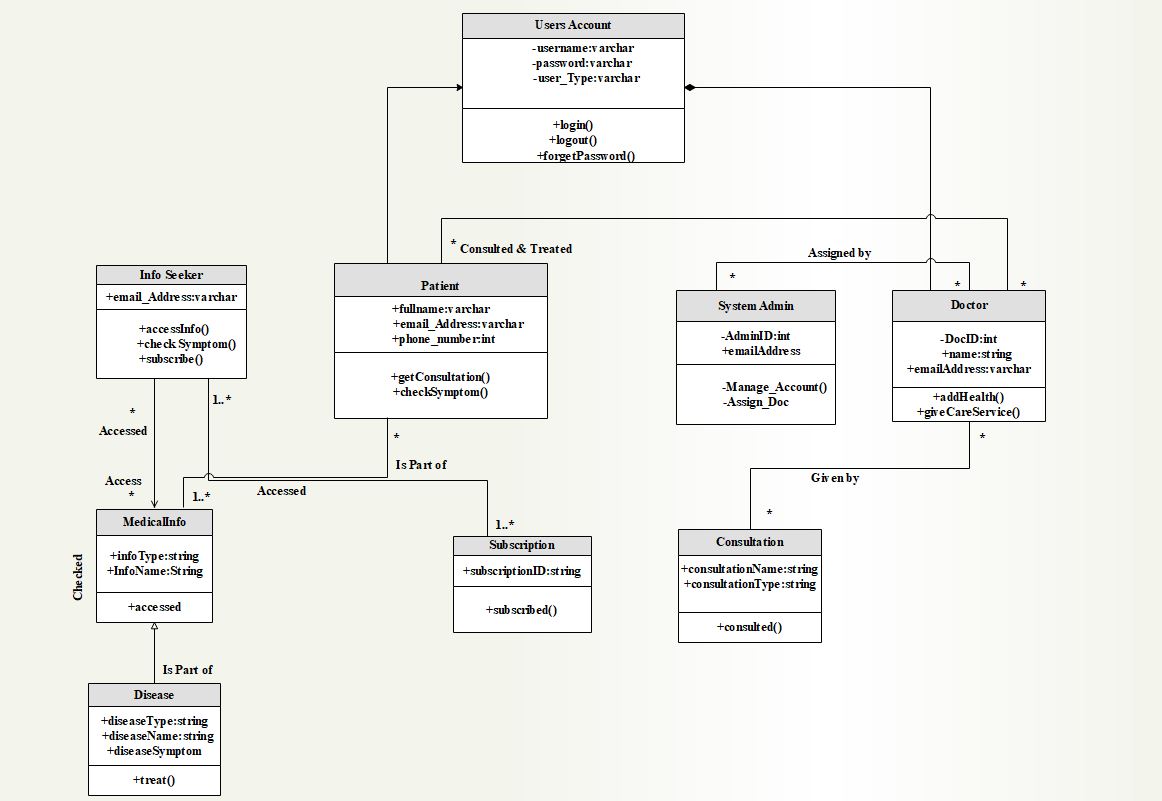


Figure 10. Class Diagram of Fayyaan Faaya OMCSS

## **Sequence Diagram**

Sequence diagrams are used to show the flow of logic within our system in a visual manner, enabling us to document and validate logic, and are commonly used for both analysis and design purpose.

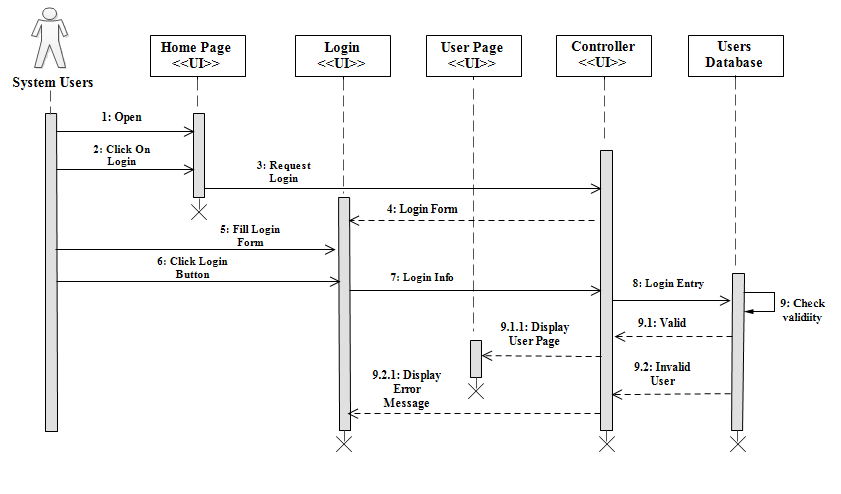


Figure 11. Sequence Diagram for System Users Login expect Information Seeker.

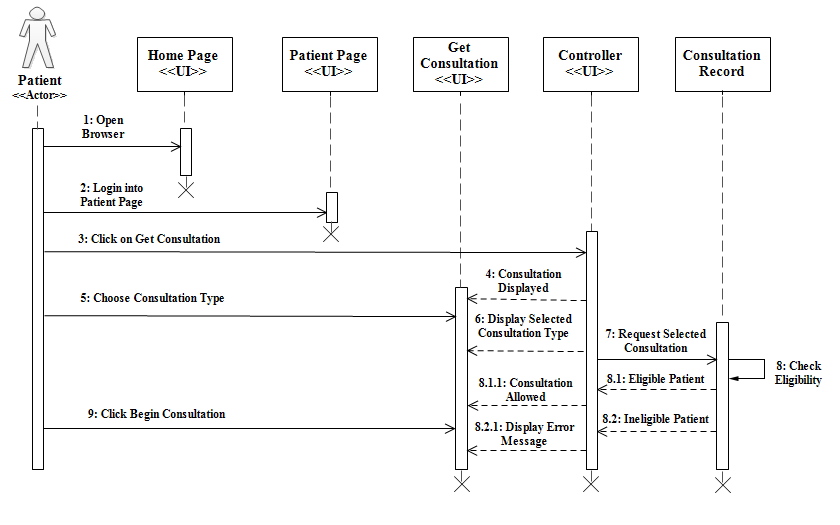


Figure 12. Sequence Diagram for Getting Consultation

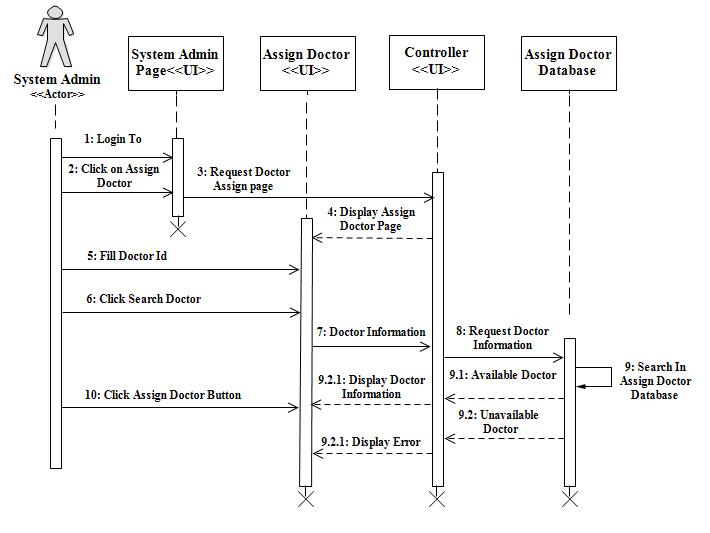
****

Figure 13. Sequence Diagram for Assigning Doctor

## **Activity Diagram**

Activity diagram is an advanced version of flow charts that modeling the flow from one activity to another activity.

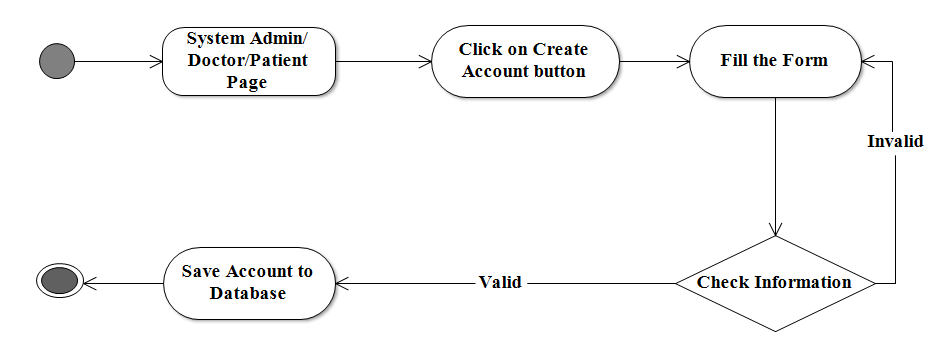
****

Figure 14. Activity Diagram for Creating Account

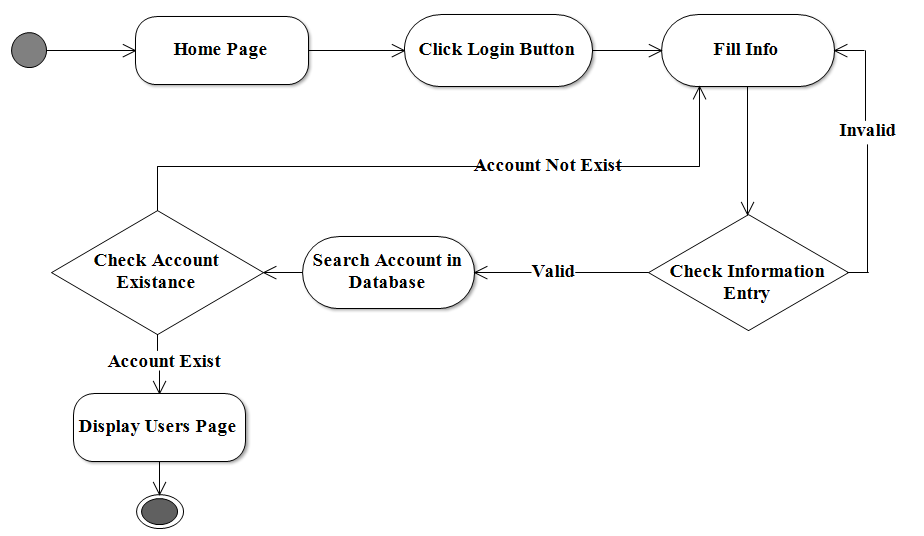
****

Figure 15. Activity Diagram for Users Login expect for Information Seeker

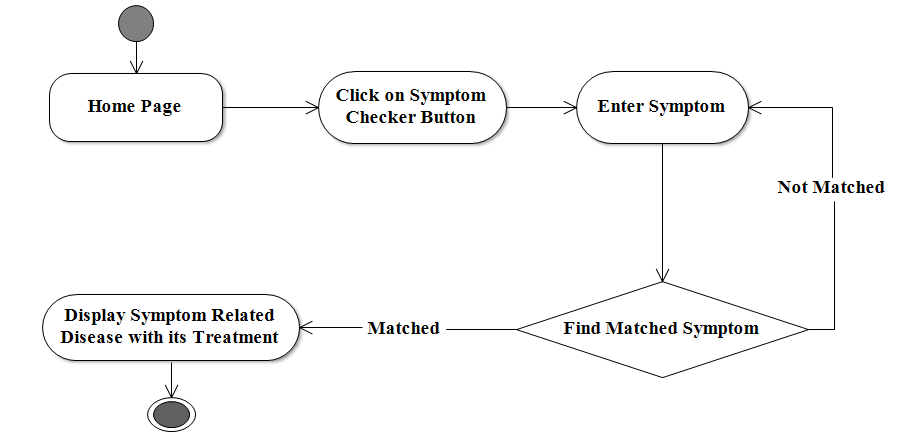
****

Figure 16. Activity Diagram for Checking Symptom

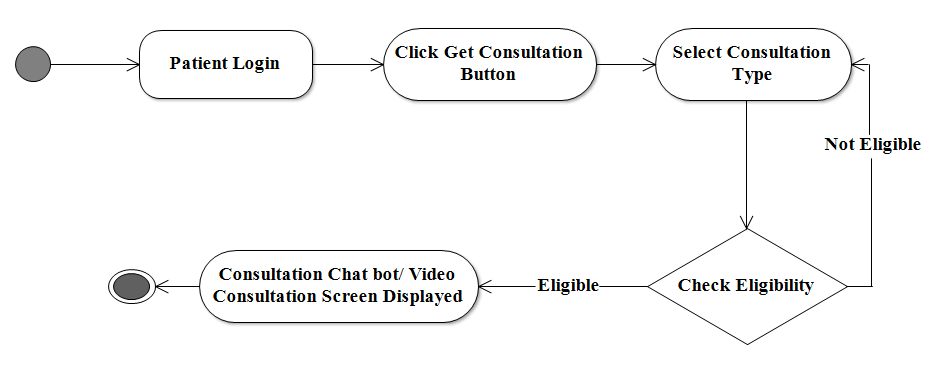
****

Figure 17. Activity Diagram for Getting Consultation

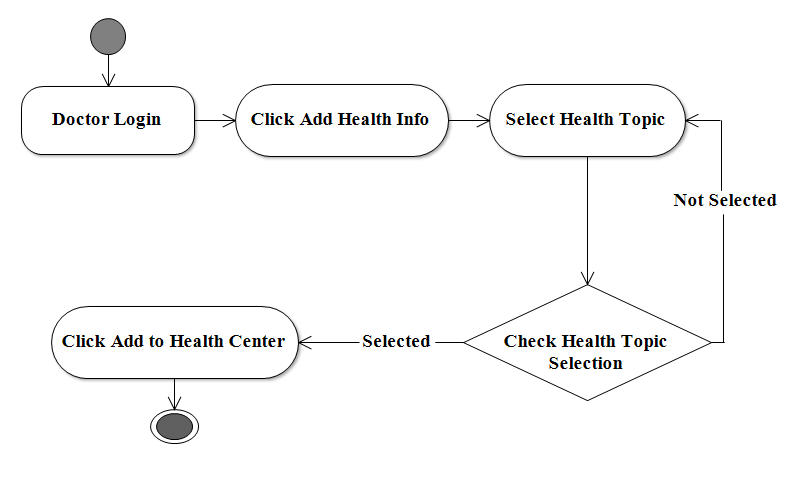
****

Figure 18. Activity Diagram for Adding Health Info

## **User Interface Design**

### Form Diagram

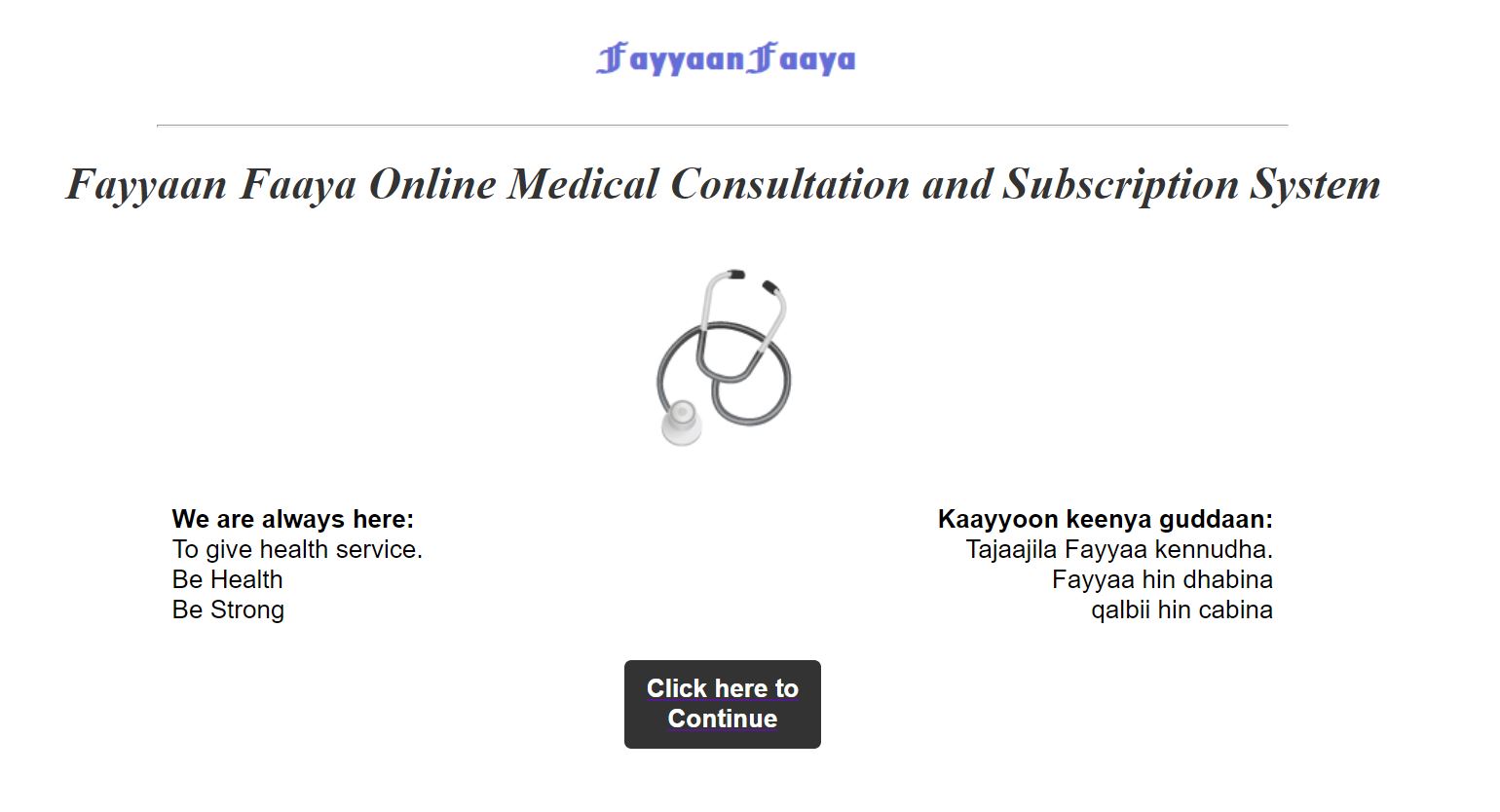
****

Figure 19. User Interface Design for Home Page

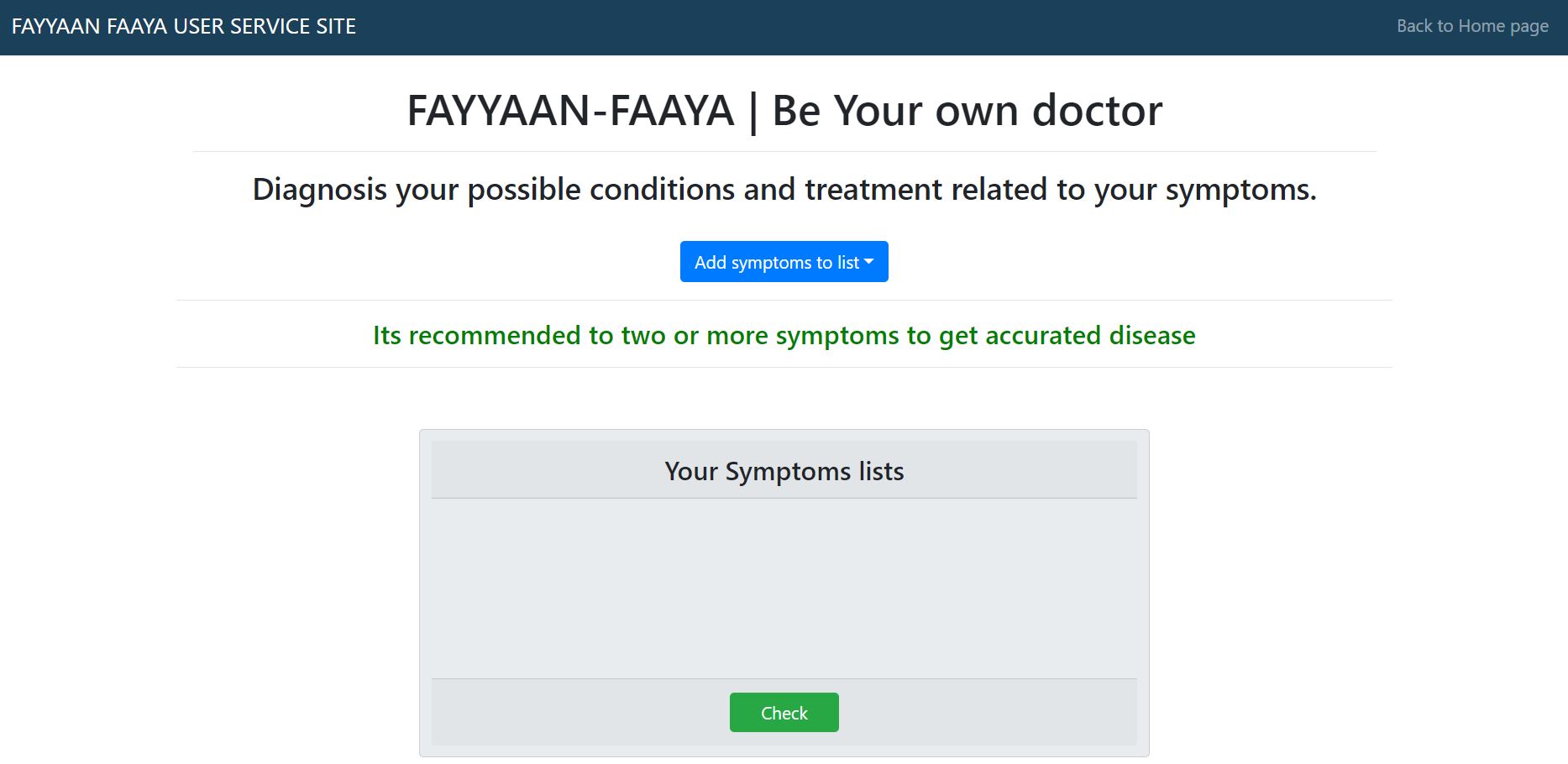


Figure 20. User Interface Design for Checking Symptom

### Dialogue and Interface Design

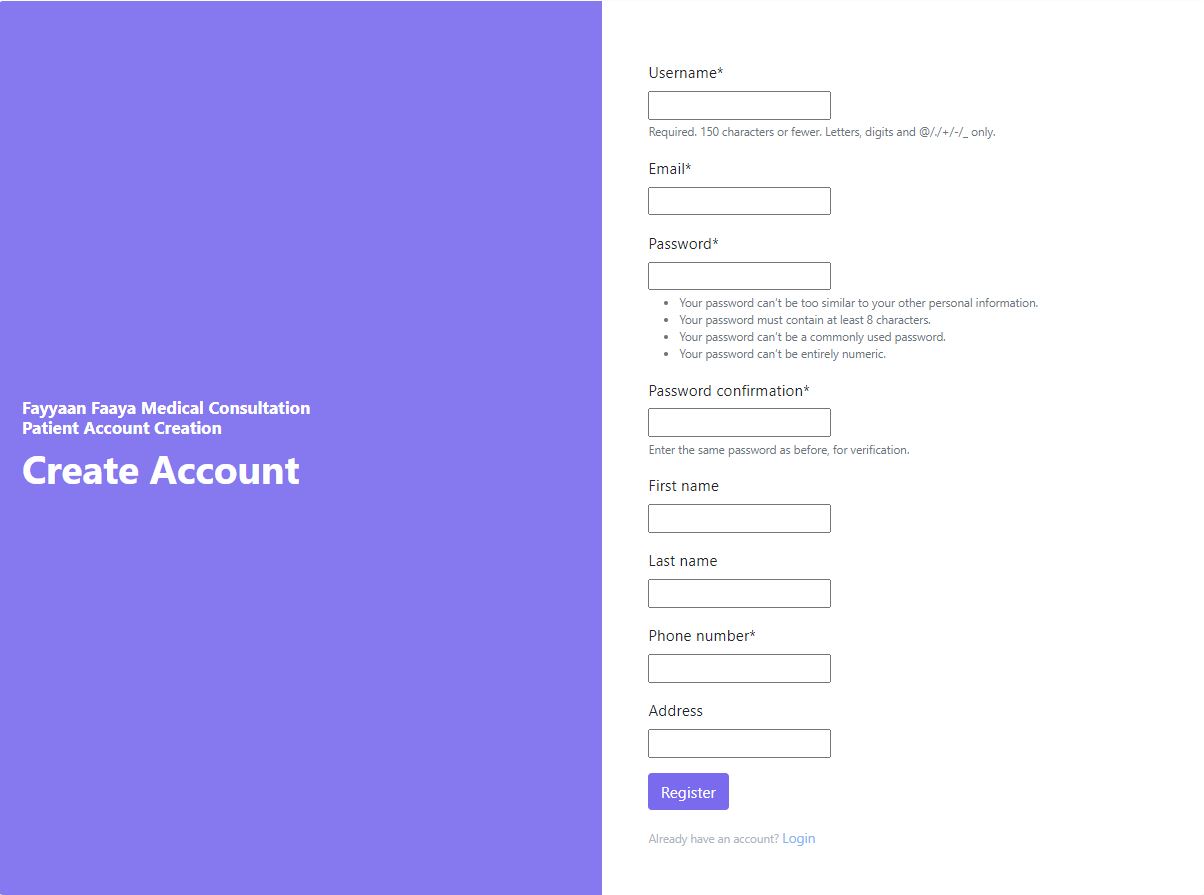


Figure 21. Report Design for System Patient Registration Page

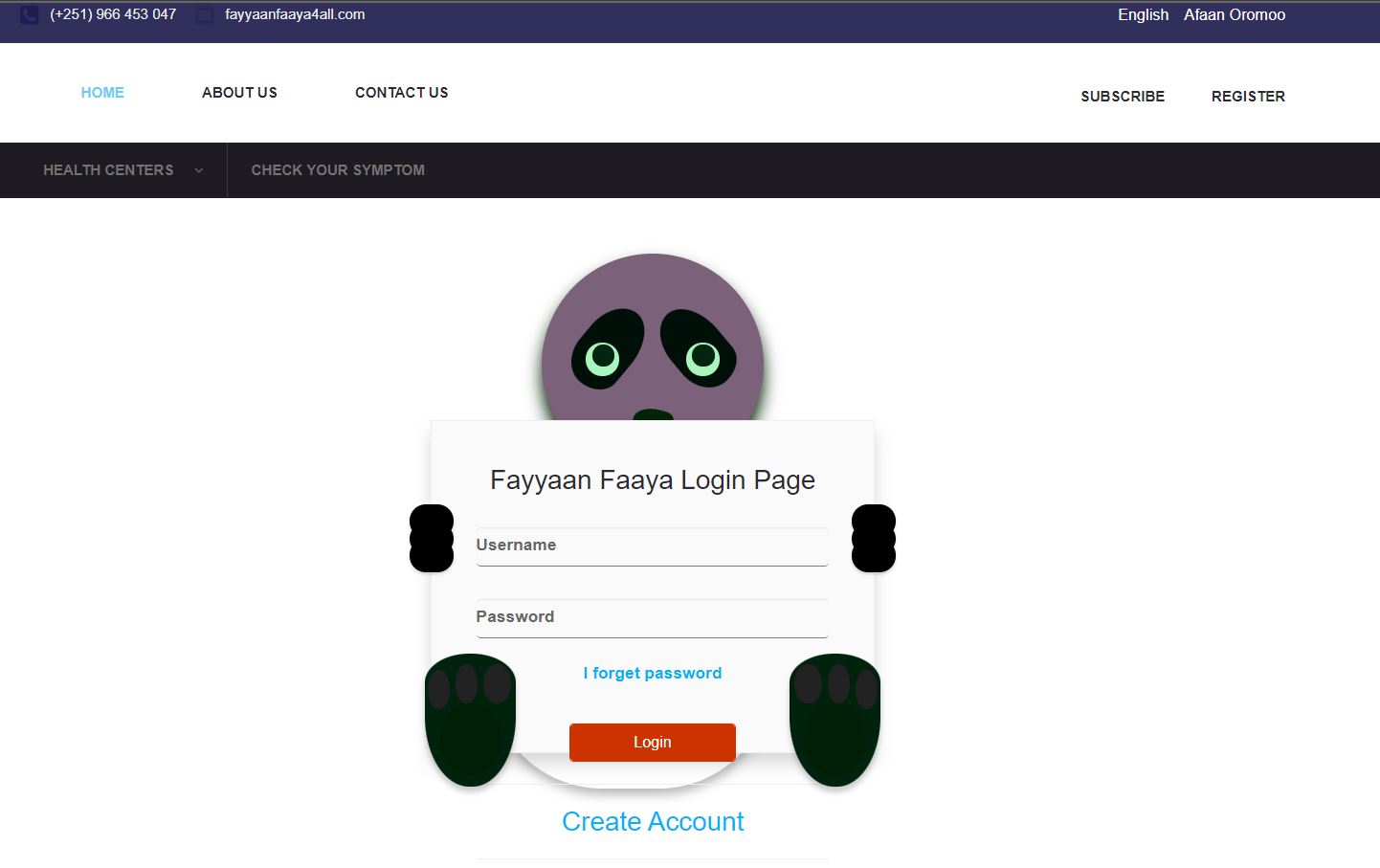
****

Figure 22. Dialogue and Interface Design for Login Page

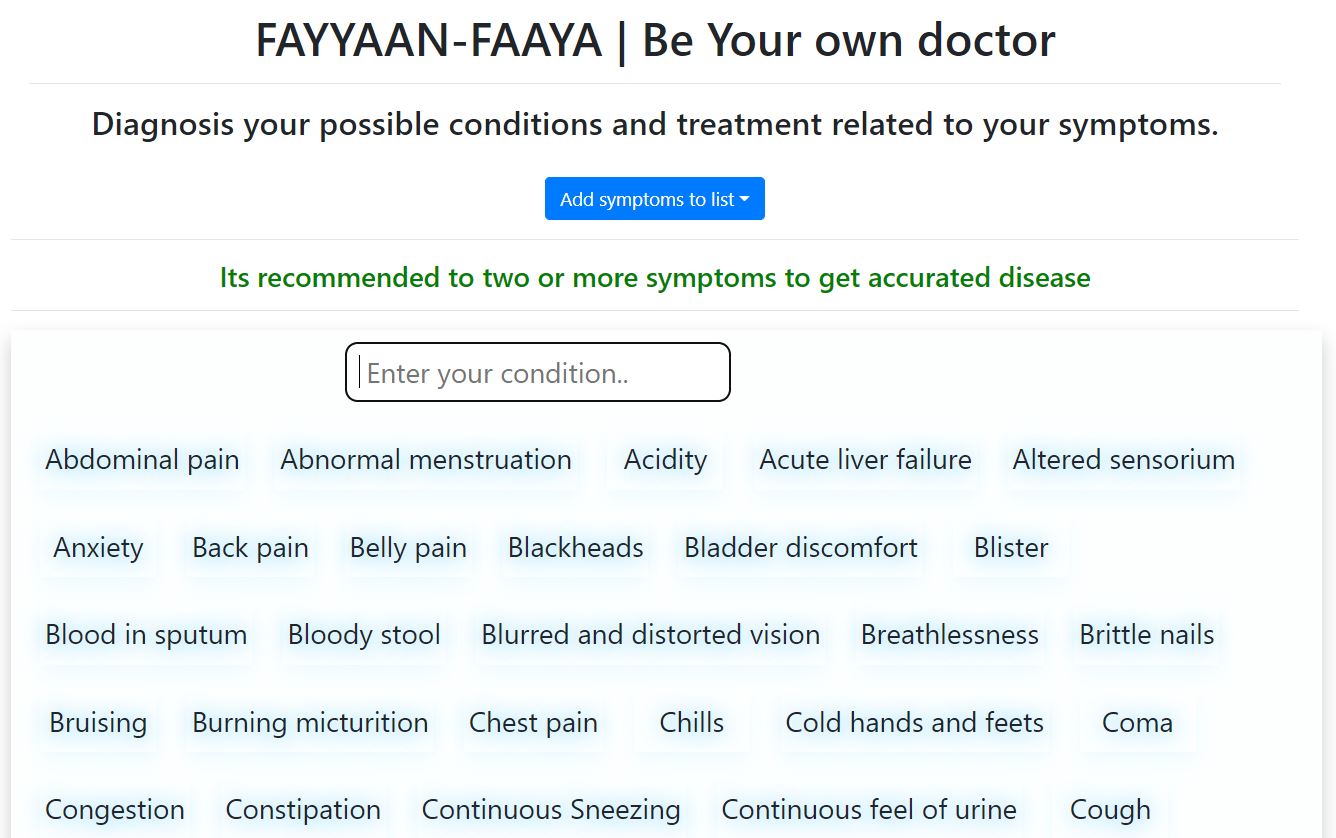
****

Figure 23. Dialogue and Interface Design for Searching and Entering Symptom

## **State Chart Diagram**

A state chart diagram shows a state machine. It’s useful for modelling the lifetime of an object and shows flow of control from state to state.

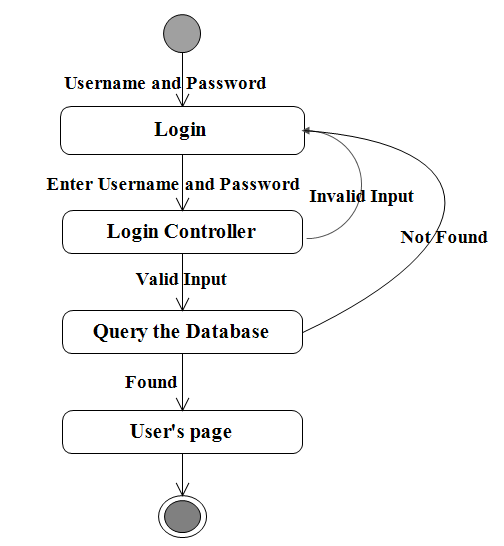
****

Figure 24. State Chart for Login

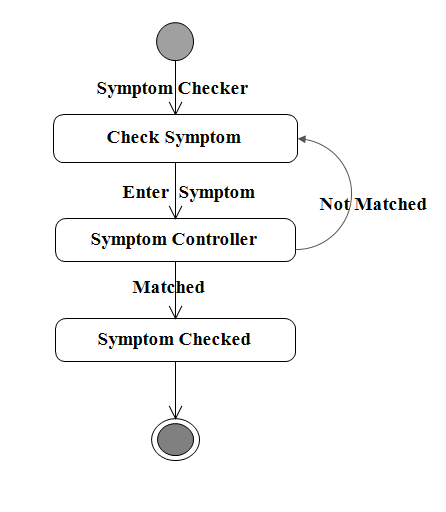


Figure 25. State Chart Diagram for Checking Symptom

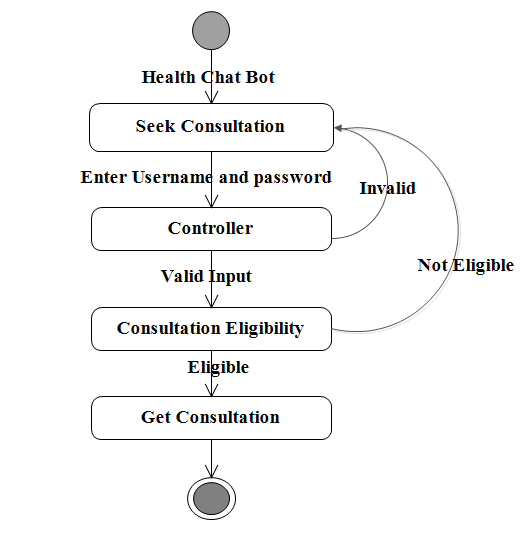
****

Figure 26. State Chart Diagram for Getting Consultation

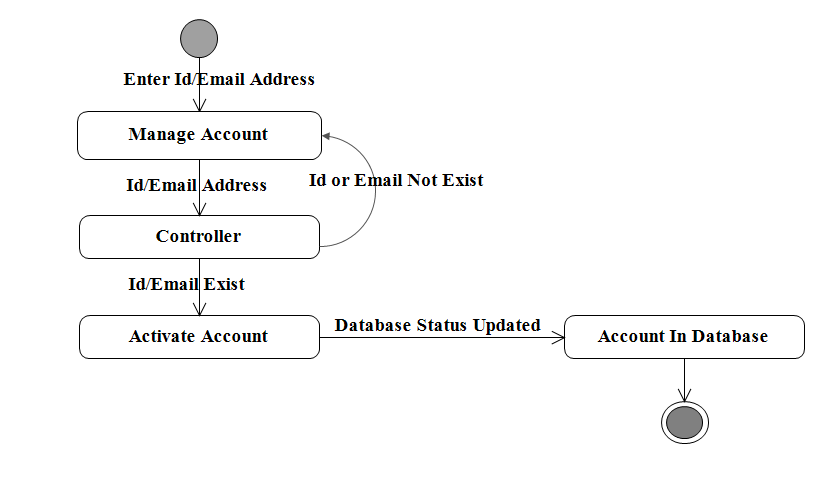
****

Figure 27. State Chart Diagram for Activating Account

## **Object Diagram**

An object diagram models is a group of objects and their links at a point of time. It shows the instances of the things in a class diagram. It is the static part of an interaction diagram.

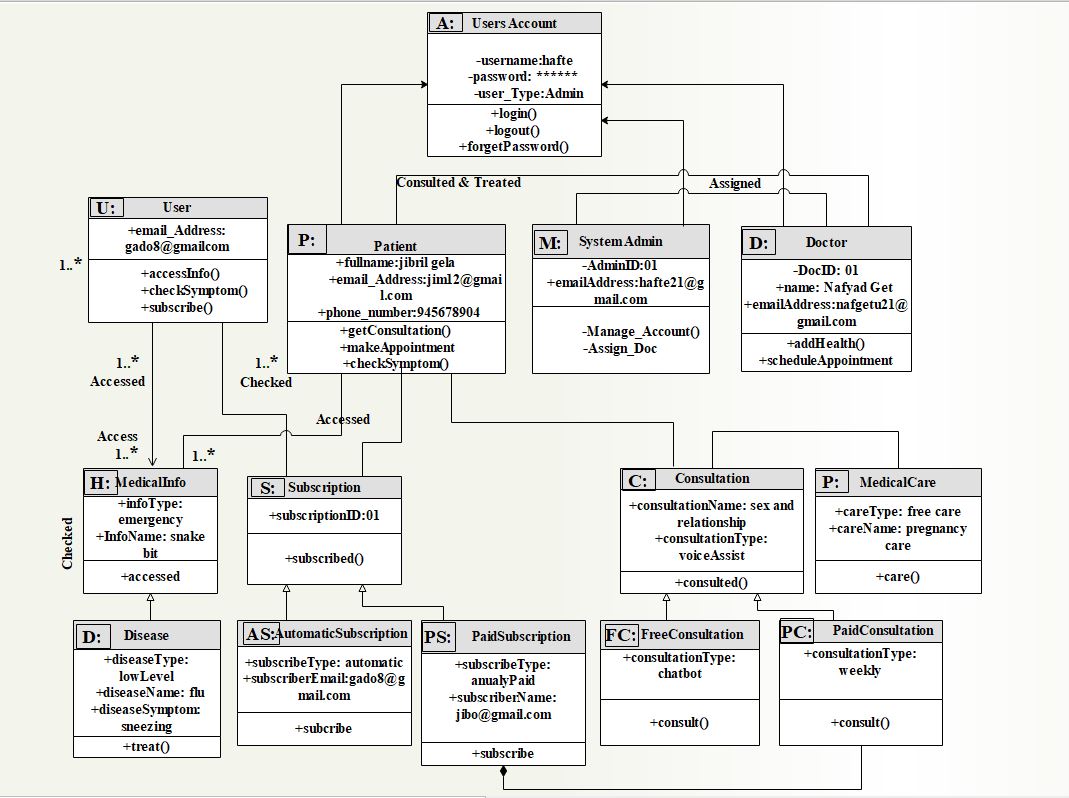


Figure 28. Object Diagram of Fayyaan Faaya OMCS

## **Persistent Modeling**

Persistent modeling is logical data model that contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which can then be used to create a database.

**Database**: **FayyaanFaayaDB**

**Tables:**

* Accounts\_user
* Account\_systemadmin
* Accounts\_doctor
* Accounts\_patient
* Accounts\_user\_user\_permissions
* HealthForAll\_subscribe
* HealthForAll\_disease
* HealthForAll\_disease\_adder\_doctor
* HealthForAll\_consultation
* HealthForAll\_News
* HealthForAll\_feedback
* HealthForAll\_feedback\_recieved
* auth\_group
* auth\_group\_permission
* auth\_permission
* Session

Table 14. Accounts\_user Database Table

|  |  |  |
| --- | --- | --- |
| **Accounts\_user** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
| **FK** | User\_id | Bigint |
|  | First\_name | varchar**(1**50**)** |
|  | Last\_name | varchar(150) |
| **UNIQUE** | email | varchar(254) |
| **UNIQUE** | username | varchar(150) |
|  | password | varchar(128) |
|  | last\_login | Date |
|  | is\_superuser | Boolean |
|  | is\_patient | Boolean |
|  | is\_doctor | Boolean |
|  | is\_staff | Boolean |
|  | is\_active | Boolean |
|  | date\_joined | Date |

Table 15. Accounts\_doctor Database Table

|  |  |  |
| --- | --- | --- |
| **Accounts\_doctor** | | |
|  | **Column** | **Type** |
| **PK** | user\_id | int(10) |
| **FK** | doctor\_id | Bigint |
| **UNIQUE** | phone\_number | varchar(12) |
|  | specialization | varchar(100) |
|  | address | varchar(250) |
|  | Social\_media | varchar(200) |

Table 16. Accounts\_systemadmin Database Table

|  |  |  |
| --- | --- | --- |
| **Accounts\_systemadmin** | | |
|  | **Column** | **Type** |
| **PK** | user\_id | Int(10) |

Table 17. Accounts\_patient Database Table

|  |  |  |
| --- | --- | --- |
| **Accounts\_patient** | | |
|  | **Column** | **Type** |
| **PK** | user\_id | Int(10) |
| **Unique** | phone\_number | varchar(12) |
|  | address | varchar(250) |

Table 18. HealthForAll\_subscribe Database Table

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_subscribe** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
| **index** | subscribe\_id | Bigint |
| **UNIQUE** | subscriber | varchar(254) |
|  | Subscribe\_date | Date |

Table 19. HealthForAll\_consultation Database Table

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_consultation** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
|  | Consultation\_date | Date |
|  | status | varchar(20) |
| **Index** | doctor\_id | Bigint |
| **Index** | patient\_id | Bigint |

Table 20. HealthForAll\_disease Database Table

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_disease** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
|  | diseasename | varchar(200) |
|  | diseasedescription | text |
|  | diseasesymptom | text |
|  | diseasetreatment | varchar(250) |

Table 21. HealthForAll\_feedback Database Table

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_feedback** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
|  | fullname | varchar(100) |
|  | email | varchar(200) |
|  | message | text |

Table 22. HealthForAll\_news Database table

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_news** | | |
|  | **Column** | **Type** |
| **PK** | id | int(10) |
|  | title | varchar(100) |
|  | content | text |
| **Index** | adder\_id | bigint |

Table 23. Django\_session database table

|  |  |  |
| --- | --- | --- |
| **Django\_session** | | |
|  | **Column** | **Type** |
| **PK** | session\_key | varchar(40) |
|  | session\_data | text |
|  | expire\_date | date |

* + 1. **Normalized Physical Database Model**

**1rst Normal Form (Cell Atomicity):** - Its used to remove repeating groups from the table and create atomicity.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Accounts\_user (1rst Normal)** | | | | | | | |
| **id** | **first\_name** | **middle\_name** | **username** | **password** | **email** | **is\_superuser** | **last\_login** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Accounts\_doctor** | | | | |
| **user\_id** | **specilization** | **phone\_number** | **address** | **social\_media** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Doctor\_AddressInfo(newly introduced table)** | | | |
| **user\_id** | **phone\_umber** | **address** | **social\_media** |

|  |  |  |
| --- | --- | --- |
| **Accounts\_subscribe (1rst Normal Form)** | | |
| **id** | **subscriber** | **subDate** |

|  |  |  |
| --- | --- | --- |
| **Accounts\_patient** | | |
| user\_id | **phone\_number** | **address** |

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_consultation (1rst Normal Form)** | | |
| **id** | **date** | **status** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CareServices (1rst Normal Form)** | | | | |
| **careID** | **emailAddress** | **payrate** | **careType** | **payType** |

|  |  |  |  |
| --- | --- | --- | --- |
| **HealthForAll\_feedback (1rst Normal Form)** | | | |
| **id** | **name** | **email** | **message** |

**2nd Normalized Form (1st NF+ Full-dependency on Primary key): -** The table must be in 1rst Normal Form and should not contain partial dependency or full depend on primary key.

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_feedback\_recieved(1rst and 2nd Normal)** | | |
| **id** | **feedback\_id** | **user\_id** |

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_diseaseinfo\_adder\_doctor(1rst and 2nd Normal Form)** | | |
| **id** | **diseaseinf\_id** | **doctor\_id** |

|  |  |  |
| --- | --- | --- |
| **HealthForAll\_subscriber\_viewer(1rst Normal Form)** | | |
| **id** | **subscribe\_id** | **user\_id** |

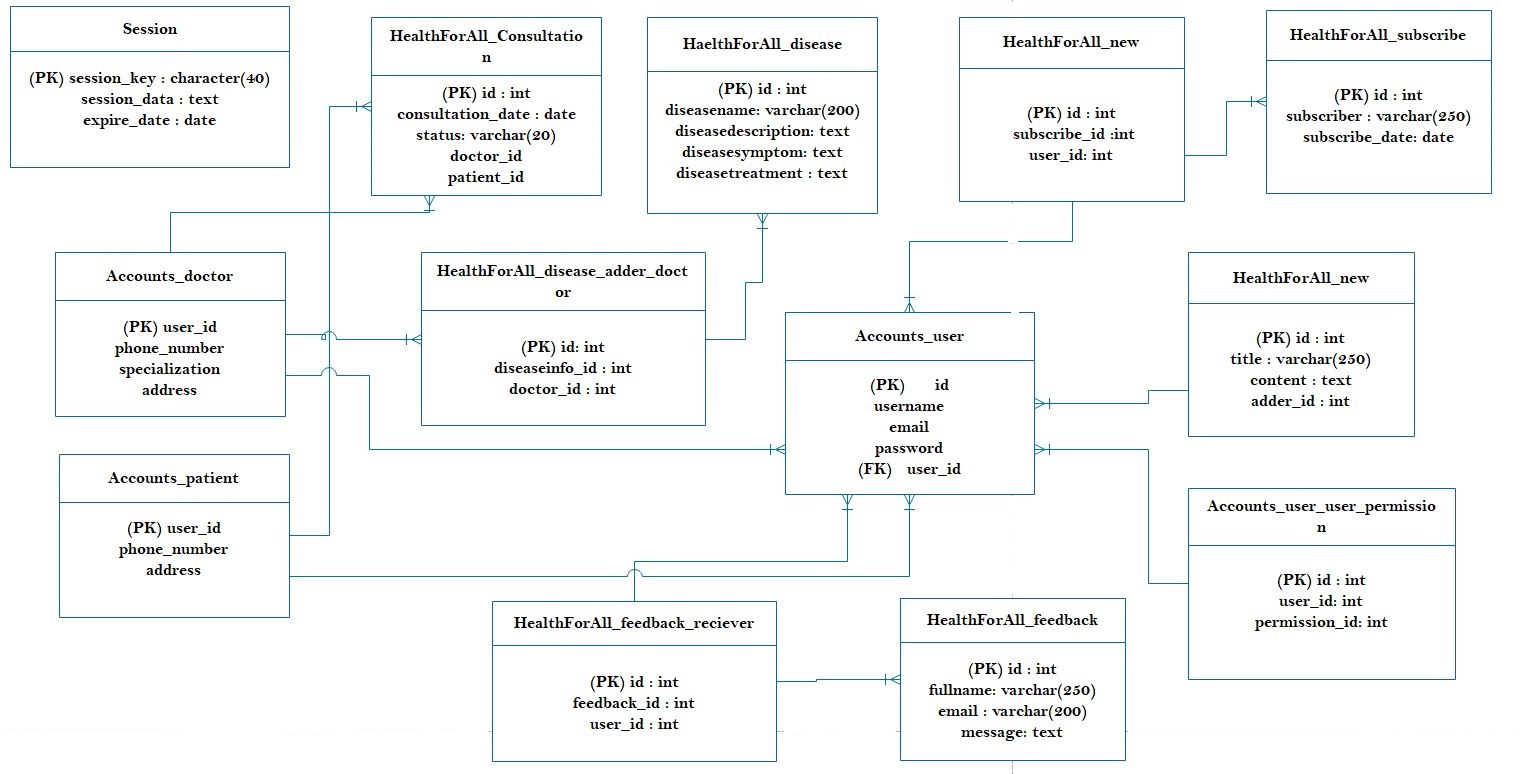
****

Figure 29. Persistent Diagram Fayyaan Faaya OMCSS

## **Component Diagram**

Component diagram show how Components are combined together to form alarger component and or software system.

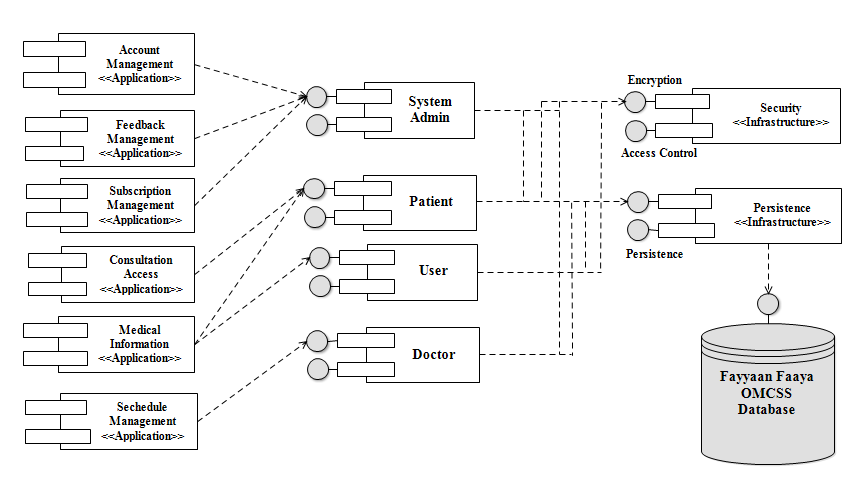


Figure 30. Component Diagram Fayyaan Faaya OMCSS

## **Deployment Diagram**

Deployment diagrams are show the hardware for our system, the software installed on that hardware, and the middleware used to connect the disparate machines to one another.

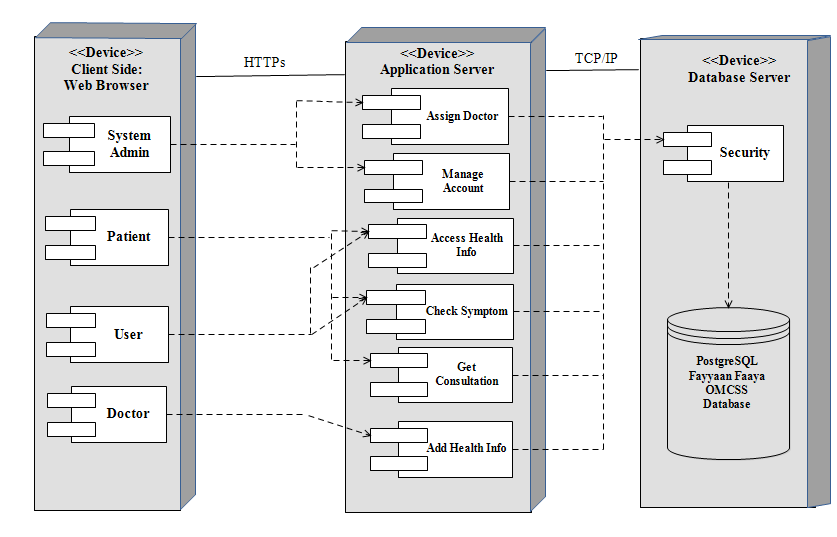


Figure 31. Deployment Diagram for Fayyaan Faaya OMCSS

# Chapter Four

# **IMPLEMENTATION AND TESTING**

# **Report on Programming/Coding**

The system is developed with the help of appropriate tools that been suggested and are loaded in to the server. The model is trained is based on machine learning models and we use Naive Bayes (with Multinomial NB model) and Decision Tree. Then the system is tested with appropriate data inputs and trained datasets to check the successfulness of the system. This being carried out by inputting data that are of rare to be inputted.

**Sample code for Training models using Decision Tree**

from django.shortcuts import render

from .models import Chatbot

import pandas as pd

import pyttsx3

from sklearn import preprocessing

from sklearn.tree import DecisionTreeClassifier,\_tree

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import cross\_val\_score

from sklearn.svm import SVC

import csv

import warnings

warnings.filterwarnings("ignore", category=DeprecationWarning)

training = pd.read\_csv('Training.csv')

testing= pd.read\_csv('Testing.csv')

cols= training.columns

cols= cols[:-1]

x = training[cols]

y = training['prognosis']

y1= y

reduced\_data = training.groupby(training['prognosis']).max()

#mapping strings values to numbers

le = preprocessing.LabelEncoder()

le.fit(y)

y = le.transform(y)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.33, random\_state=42)

testx    = testing[cols]

testy    = testing['prognosis']

testy    = le.transform(testy)

clf1  = DecisionTreeClassifier()

clf = clf1.fit(x\_train,y\_train)

scores = cross\_val\_score(clf, x\_test, y\_test, cv=3)

#print (scores.mean())

model=SVC()  # Support Vector Classifier

model.fit(x\_train,y\_train)

**Sample code for creating database table using django models**

from django.db import models

from django.contrib.auth.models import AbstractUser

from django.db import models

from phonenumber\_field.modelfields import PhoneNumberField

class User(AbstractUser):

    is\_patient = models.BooleanField(default=False)

    is\_doctor = models.BooleanField(default=False)

class SystemAdmin(models.Model):

   user = models.OneToOneField(User, on\_delete=models.CASCADE,verbose\_name = "related to User", primary\_key=True, related\_name='admin')

class Patient(models.Model):

    user = models.OneToOneField(User, on\_delete=models.CASCADE,verbose\_name = "related to User", primary\_key=True, related\_name='patient')

    phone\_number = models.CharField(max\_length=12,null=False, blank=False, unique=True)

    address = models.CharField(max\_length=250)

    def \_\_str\_\_(self):

        return self.user.username

class Doctor(models.Model):

    department = (

        ("Doctor Allergist","Allergist"),("Doctor of Cardiologist","Cardiologist"),("Doctor of Dermatologist","Dermatologist"),("Doctor of Gastroenterologist","Gastroenterologist"),

        ("Doctor of Neurologist","Neurologist"),("Doctor of Orthopedist","Orthopedist"),("Doctor of Rheumatologist","Rheumatologist"),("Doctor of Urologist","Urologist"),("Other","Other")

    )

    social\_media = (

        ("Facebook","Facebook"),("Whatsapp","Whatsapp"),("Viber","Viber"),("Telegram","Telegram"),("Other","Other")

    )

    user = models.OneToOneField(User,on\_delete=models.CASCADE, verbose\_name = "related to User", related\_name="doctor", primary\_key=True)

    phone\_number = models.CharField(max\_length=12,null=False, blank=False, unique=True)

    specialization = models.CharField(max\_length=100,choices = department)

    address = models.CharField(max\_length=250)

    social\_media = models.CharField(max\_length=200,choices = social\_media)

    def \_\_str\_\_(self):

        return self.user.username

class News(models.Model):

    adder = models.ForeignKey(User,on\_delete=models.CASCADE, related\_name="Doc")

    title = models.CharField(max\_length=200)

    content = models.TextField()

class DiseaseInfo(models.Model):

    adder\_doctor = models.ManyToManyField(Doctor,primary\_key=False, related\_name="adder")

    diseasename = models.CharField(max\_length = 200)

    diseasedescription = models.TextField()

    diseasesymptom =models.TextField()

    diseasetreatment = models.CharField(max\_length=250)

class Disease(models.Model):

    diseasename = models.CharField(max\_length = 200)

    diseasedescription = models.TextField()

    diseasesymptom =models.TextField()

    diseasetreatment = models.CharField(max\_length=250)

class Consultation(models.Model):

    patient = models.ForeignKey(Patient ,null=True, on\_delete=models.SET\_NULL)

    doctor = models.ForeignKey(Doctor ,null=True, on\_delete=models.SET\_NULL)

    consultation\_date = models.DateField()

    status = models.CharField(max\_length = 20)

class subscribe(models.Model):

    viewer = models.ManyToManyField(User,primary\_key=False,related\_name="viewer")

    subscriber = models.EmailField(max\_length=250)

    subscribe\_date = models.DateField("Subscribe date",auto\_now= True)

class Feedback(models.Model):

    recieved = models.ManyToManyField(User,primary\_key=False)

    fullname = models.CharField(max\_length=100)

    email = models.EmailField(max\_length=200)

    message = models.TextField()

    def \_\_str\_\_(self):

        return "Message from " + self.fullname + ' - ' + self.email

**Sample code for both urls and views**

**urls.py**

from django.urls import path

from . import views

urlpatterns = [

    path('',views.welcome,name='welcome'),

    path('index/', views.index, name='index'),

    path('contact/', views.contact, name='contact'),

    path('about/', views.about, name='about'),

    path('subscriber/', views.subscriber, name='subscriber'),

    path('news/', views.news, name='news'),

    path('checkdisease/', views.checkdisease, name='checkdisease'),

    path('diseaseinfo/',views.diseaseinfo, name='diseaseinfo'),

    path('consult\_option/', views.consult\_option, name='consult\_option'),

    path('doctor\_profile/', views.doctor\_profile, name='doctor\_profile'),

    path('notfound/', views.notfound, name='notfound'),

    # Urls of disease information

    path('cancer/', views.cancer, name='cancer'),

]

**views.py**

def welcome(request):

    return render(request,'English/base/welcome.html')

def index(request):

    services1 = Our\_Service()

    services1.id = 0

    services1.service\_name = 'Get common Information about disease'

    services2 = Our\_Service()

    services2.id = 1

    services2.service\_name = 'Check Symptom of your condition'

    services3 = Our\_Service()

    services3.id = 2

    services3.service\_name = 'Get Consultation for your Conditions'

    services4 = Our\_Service()

    services4.id = 3

    services4.service\_name = 'Get Care Service for your special condition'

    services = [services1,services2,services3,services4]

    return render(request,'English/base/index.html',{'services':services})

def contact(request):

   if request.method == 'POST':

      fullname = request.POST['cf\_name']

      email = request.POST['cf\_email']

      message = request.POST['cf\_message']

      if len(fullname)<2 or len(email)<3 or len(message)<4:

          messages.error(request, "Please fill the form correctly")

      else:

         contact=Feedback(fullname=fullname, email=email, message=message)

         contact.save()

         messages.success(request, "Your message has been successfully sent")

   return render(request,'English/base/contact.html')

def about(request):

    return render(request, 'English/base/about.html')

def news(request):

    return render(request, 'English/base/news.html')

def doctor\_profile(request):

    information = Doctor.objects.all()

    return render(request,'English/doctor/doctor\_profile.html', {'information': information })

def notfound(request):

    return render(request, 'English/base/404.html')

**Sample code for login**

**views.py**

def login\_view(request):

    if request.user.is\_authenticated and request.user.is\_patient:

        return redirect('index')

    if request.user.is\_authenticated and request.user.is\_doctor:

        return redirect('dashboard')

    else:

       if request.method=="POST":

        username=request.POST.get('username')

        password=request.POST.get('password')

        user=authenticate(request,username=username ,password=password)

        if user is not None:

            if user.is\_patient:

                login(request,user)

                return redirect('index')

            elif user.is\_doctor:

                login(request,user)

                return redirect('dashboard')

            elif user.is\_superuser:

                return redirect('admin:index')

        else:

            messages.error(request,"Please enter correct username & password")

       return render(request, 'English/account/login.html')

**login.html**

{% extends 'English/base/header.html'%}

{% load static %}

{% block content %}

<!DOCTYPE html>

<html lang="en" >

<head>

  <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, user-scale=yes">

  <title>Fayyaan Faaya | Login</title>

  <link rel="stylesheet" href="{% static 'account/login/css/style.css'%}">

</head>

<body>

<div class="panda">

 <!-- <div class="ear"></div> -->

  <div class="face">

    <div class="eye-shade"></div>

    <div class="eye-white">

      <div class="eye-ball"></div>

    </div>

    <div class="eye-shade rgt"></div>

    <div class="eye-white rgt">

      <div class="eye-ball"></div>

    </div>

    <div class="nose"></div>

    <div class="mouth"></div>

  </div>

  <div class="body"> </div>

  <div class="foot">

    <div class="finger"></div>

  </div>

  <div class="foot rgt">

    <div class="finger"></div>

  </div>

</div>

<form action="" method="POST" novalidate>

  {% csrf\_token %}

  {% for message in messages %}

    <h5 style="color:red;"> {{ message }} </h5>

    <p class="alert">{{ message }}</p>

  {% endfor %}

  <div class="hand"></div>

  <div class="hand rgt"></div>

  <h3>Fayyaan Faaya Login Page</h3>

  <br>

  <div class="form-group">

    <input type="text" name="username" required="required" class="form-control" />

    <label class="form-label">Username</label>

  </div>

  <div class="form-group">

    <input id="password" type="password" name="password" required="required" class="form-control" autocomplete=""/>

    <label class="form-label">Password</label>

    <br>

    <b><p><a href="{% url 'reset\_password'%}">I forget password</a></p></b>

    <button class="btn">Login </button>

    <hr>

    <h3><a href="{% url 'patient\_signup'%}">Create Account</h3></a>

    <hr>

  </div>

</form>

<h3><a href="{% url 'index'%}">Back To Home Page </a></h3>

  <script src='https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>

  <script  src="{% static 'account/login/js/script.js'%}"></script>

</body>

</html>

{% endblock %}

**Sample code for subscribing**

**views.py**

def subscriber(request):

    if request.method == "POST":

        subscriber = request.POST["email"]

        if subscribe.objects.filter(subscriber=subscriber).exists():

            messages.error(request,"You already subscribe to our website! Thank for being our member!")

        else:

            send\_mail(

            'Fayyaan Faaya Medical Consultation',

            'Hello, Thank you for being our subscriber. We will notify you daily news.',

            'fayyaanfaaya@gmail.com',

            [subscriber],

            fail\_silently=False,

            )

            done = subscribe(subscriber=subscriber)

            messages.info(request, "You subscribe to our website!. Thank you for being our subscriber!")

            done.save()

            return redirect('index')

    return render(request,'English/account/subscriber.html')

**Subscriber.html**

{% extends 'english/base/header.html'%}

{% load static %}

{% block content %}

<link rel="stylesheet" href="{% static 'subscribe/auto.css'%}">

<!-- content -->

  <!-- popup -->

  <div class="popScroll">

    <div class="popup">

      <span class="ribbon top-left ribbon-primary">

        <small>Hello!</small>

      </span>

      <br/>

      <h2>Be Fayyaan Faaya Subscriber</h2>

      <div class="subscribe-widget">

        <br/>

        <!-- form -->

        <form id="subscribe-form" method="POST">

          {% csrf\_token %}

<input type="email" name="email" placeholder="Your Email Please" class="email-form" required>

          <button type="submit" class="button">Subscribe Now</button>

        </form>

            {% for mesage in messages %}

             <p style="color:red;"> {{ message }} </p>

            {% endfor %}

      </div>

      <hr>

      <p>You can close the subscription form</p>

      <div id="option">

        <a href="{% url 'index'%}" id="home" class="boxi">Back to Home</a>

        <em>or</em>

        <a href="{% url 'welcome'%}" id="close" class="boxi closei">Close</a>

        <p class="adstext">If you subscribe to our website you get the following service</p>

           <pre><p>

              1. Fayyaan Faaya Health Infos<br>

              2. Daily Fayyaan Faaya Notifications<br>

              3. Daily Fayyaan Faaya News and Update

            </p></pre>

      </div>

    </div>

  </div>

  <script src="{% static 'subscribe/auto.js'%}"></script>

  {% endblock %}

# **Testing**

We have tried to test individual pages as well as the whole system. This process involves any activity aimed at evaluating an attribute of pages or capacity of application and determines that it meets its intended requirements or objectives. In short, it is the execution of the system to see its capability and effectiveness. Hence, our main goal is finding errors (if any) and correcting them.

The overall test plan is not just a single document but a collection of documents. Each of the component documents represents a complete test plan for part of the system or for particular type of test. After the integration of modules, the whole system was tested.

## **Unit Testing**

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually.

Unit testing focuses efforts on the smallest unit of software design. This is known as module testing. The modules are tested separately. The test is carried out during programming stage itself. In this step, each module is found to be working satisfactory as regards to the expected output from the module.

## **Integration and System testing**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. In this testing part, all the modules will be combined together and tested it for its fitness with each other and with the systems functionality.

In this testing part, all the modules will be combined together and tested it for its fitness with each other and with the systems functionality. If error occurs in combining them, the module with problem will be identified and recombined.

The objectives of system testing are: To verify that the system components perform effectively

* To perform inter-system test
* To demonstrate that the system performs both functionally and operationally as specified
* To perform appropriate types of tests relating to transaction flow, installation, reliability we have tested the overall system performance by acting like a user of the system. And we found some defects then we enhance the system.

# **Conclusion**

Fayyaan Faaya Online Medical Consultation and Subscription System is a project developed to solve problems related to health which enable the access of healthcare related information, and medical diagnosis.

The project has two phase; the first phase of the project focus on introducing the system to be developed, which includes statement of the problems, objective of the project, methodology, scope, significant, feasibility assessment, functional and non-functional requirement and use-case to analyze and design the system using object oriented methodology.

The second phase of the system is focused on programming and testing which includes coding, report on coding, sample code, testing, client feedback and user manual which give guidance to system user, on how to use the system correctly. We, the developer of this system believe that, this system will help our community all aspects of providing health related information

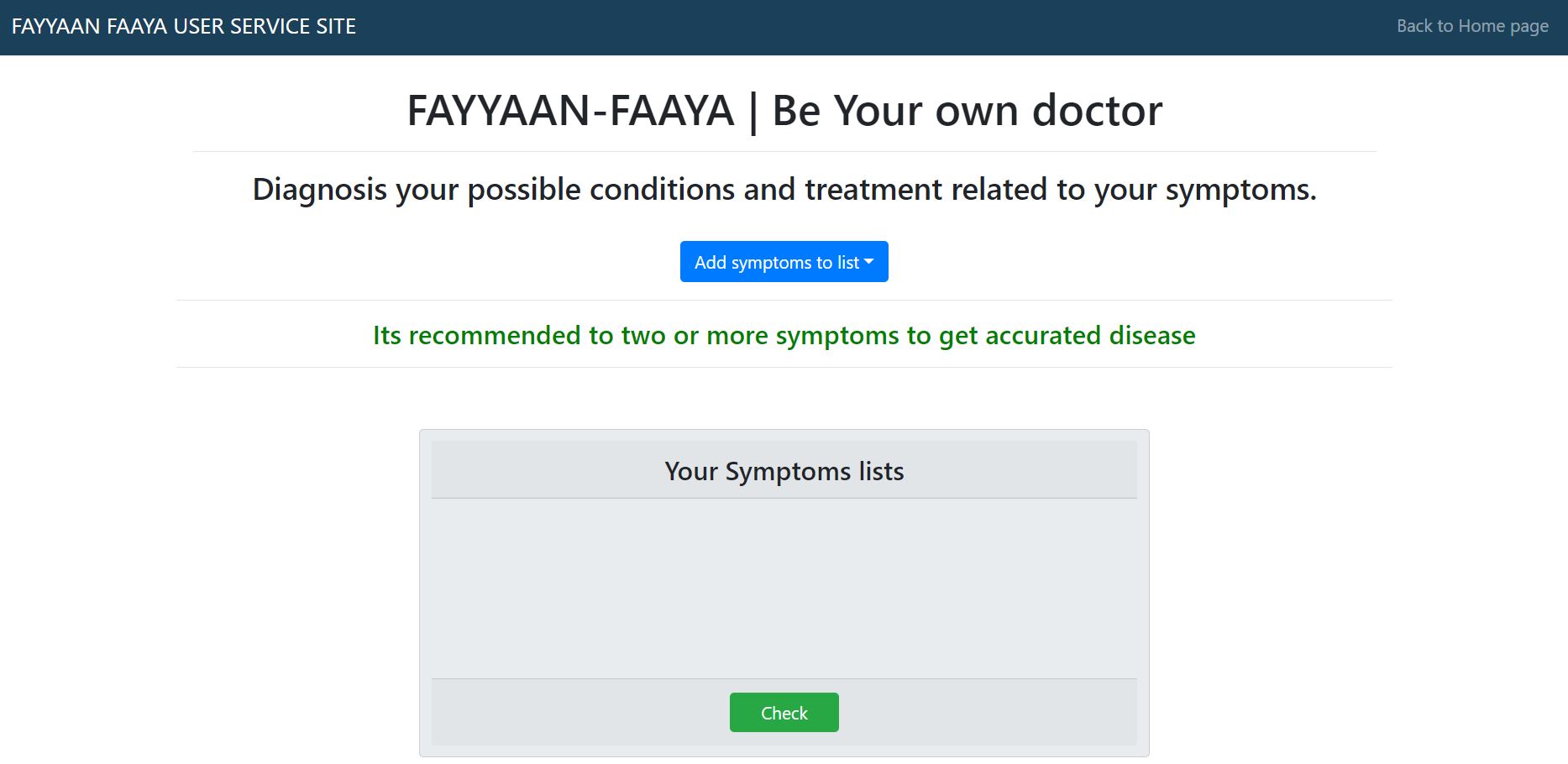
# **Client Feedbacks**

The clients responded, the system is good but there is something to add to this system, according to those comments we try to add so many things to the system as soon as possible and also, we try our best to keep what the client perceived as good from the system.

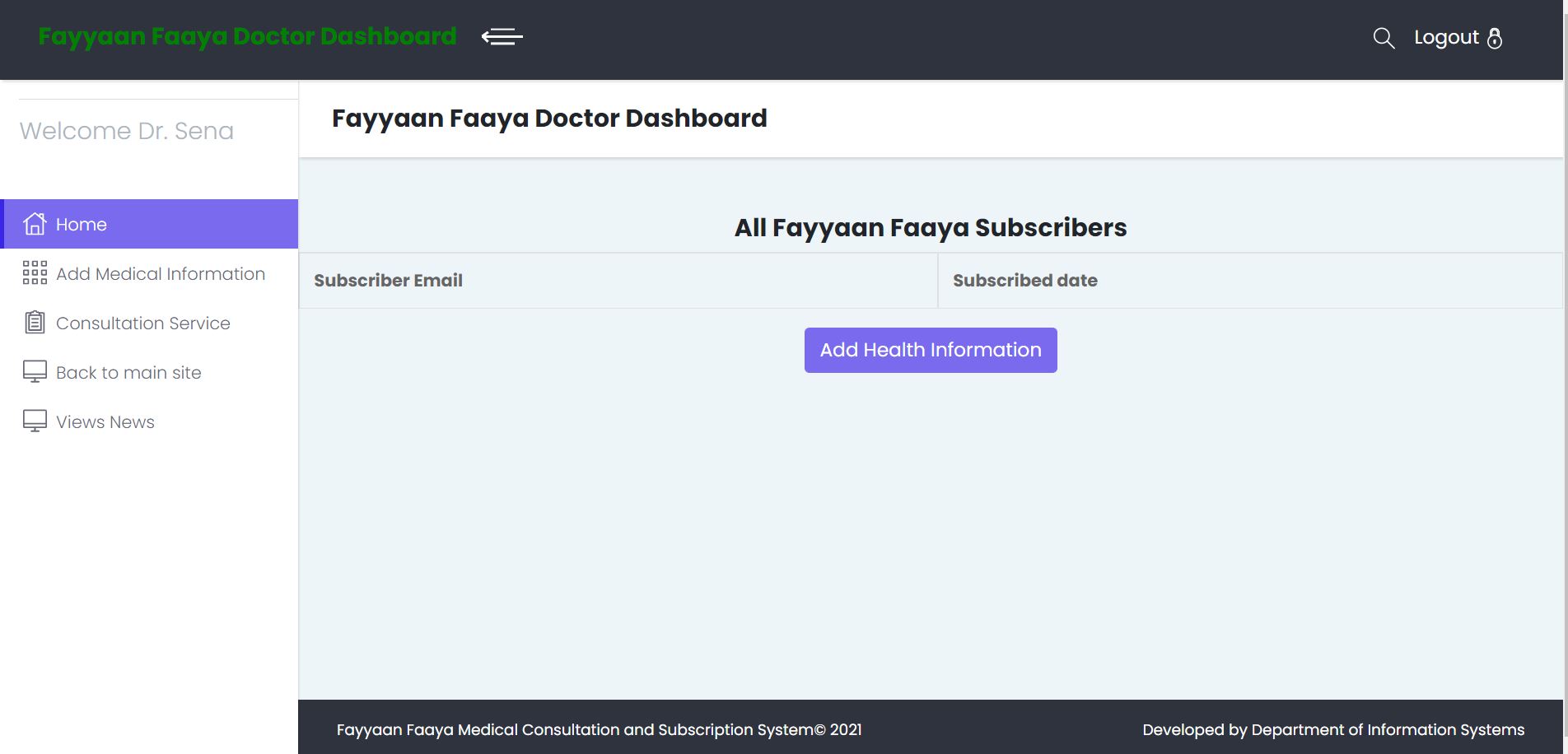
Most of them are say that your system requires time and you should have to work hard for future to develop the project until the production.

# **User Manual**

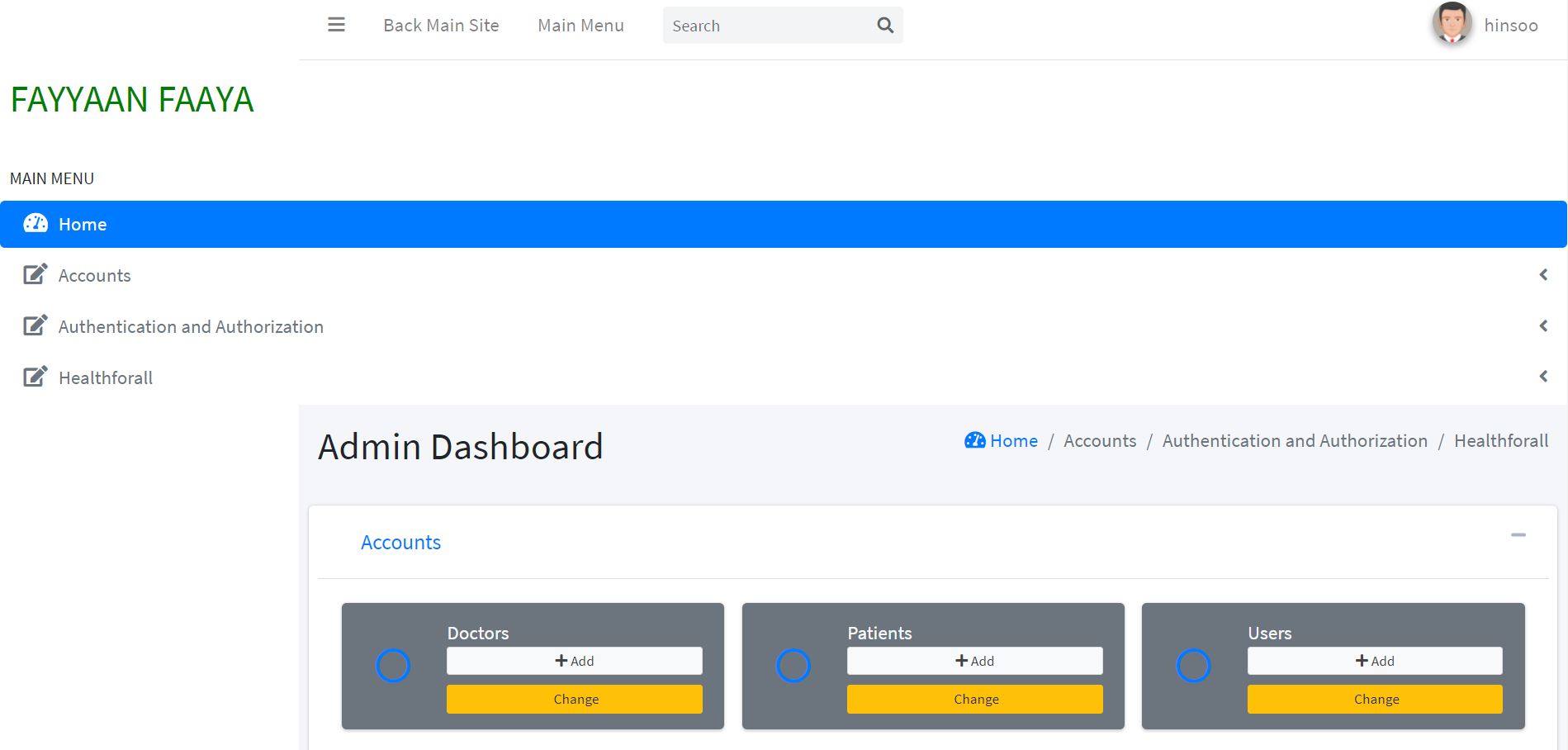
Since the system is web based, everything important for the user will be explained and implemented, while giving short training when the system is deployed.

****

1. **User manual symptom checker**

****

1. **User manual for Doctor**

****

1. **User manual for System Admin**

# **References**

|  |  |
| --- | --- |
| [1] | B. Rotimi-Williams, "Development of a Secured Web-Based," ADVANCE ACADEMIC , Akure, Nigeria, March, 2016. |
| [2] | Mohan Tanniru & Ritu Agarwal, "Systems development life-cycle for expert systems," ResearchGate, Arizona, 24 February 2018. |
| [3] | investopedia.com, "feasibility-study.asp.," Investopedia, 22 7 2021. [Online]. Available: https://www.investopedia.com/terms/f/feasibility-study.asp.. [Accessed 22 7 2021]. |
| [4] | cnhde.ei, "health\_policy.html," cnhde.ei.columbia, 17 7 2021. [Online]. Available: http://cnhde.ei.columbia.edu/healthsystem/health\_policy.html. [Accessed 17 7 2021]. |
| [5] | elementool, "index.php/2012/04/05/learn-how-to-create-srs-documents-in10-seconds/," elementool.com, 22 7 22. [Online]. Available: https://www.elementool.com/blog/index.php/2012/04/05/learn-how-to-create-srs-documents-in10-seconds/. [Accessed 22 7 2021]. |
| [6] | N. Ceta, "All You Need to Know About UML Diagrams," Tallyfy, Inc.," 12 March 2015. [Online]. Available: https://tallyfy.com/uml-diagram/.. [Accessed 22 7 2021]. |
| [7] | S. W. Ambler, " User Interface Prototyping Tips and Techniques," Ambysoft Inc.," 15 January 2014. [Online]. Available: http://www.ambysoft.com/essays/userInterfacePrototyping.html.. [Accessed 20 7 2021]. |