Find-A-Doctor: An Online Platform

By

Shamima Akter
152-35-1181

Supervisor

Ms. Nusrat Jahan

Lecturer (Senior Scale)

A project submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Department of Software Engineering
DAFFODIL INTERNATIONAL UNIVERSITY

Spring – 2019

APPROVAL

This project entitled on "Find-A-Doctor(An Online Platform)" submitted by **ShamimaAkter**bearing **ID:152-35-1181** to the Department of Software Engineering of Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

_____

Dr. Touhid Bhuiyan
Professor & Head
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

_____ Internal Examiner

_____ Internal Examiner

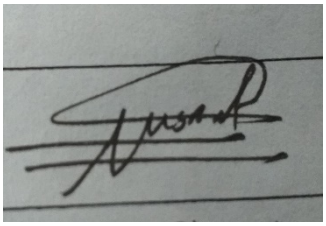
_____ External Examiner

Declaration

I hereby declare that I have taken this project under the supervision of **Ms.Nusrat Jahan**, Senior Lecturer, Department of Software Engineering, Daffodil International University. I also declare that neither this report nor any part of this has been submitted elsewhere for award of any degree.

ShamimaAkter
ID: 152-35-1181
Batch: 17th
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

Certified By:



Ms.Nusrat Jahan
Senior Lecturer
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

ACKNOWLEDGEMENT

At first, I am grateful to The Almighty Allah for making me eligible to complete this project. Then I would like to thank my supervisor Senior Lecturer, **Ms.Nusrat Jahan** Department of Software Engineering. We are extremely grateful and indebted to her expert, sincere and valuable guidance and encouragement extended to us.

We wish to express our sincere thanks to **Dr. Touhid Bhuiyan**, Professor & Head of Software Engineering department for his constant encouragement.

Last but not least, we would like to thank our parents, for their unconditional support, love and without this we would not have come this far.

Above all, we would like to thank to The Almighty Allah for giving me strength to complete this project.

A project submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering.

Table of Contents

Chapter 1 1
Introduction 1
1.1. Project Overview 1
1.2. Project Purpose 1
1.2.1. Project Background 1
1.2.2. Beneficiaries and Benefits 1
1.2.3. Project Goals 2
1.3. Stakeholders 2
1.4. Proposed System Model 2
1.4.1. Agile-Model 2
1.4.2. How we used Agile 3
1.5. Project Schedule 4
1.5.1. Gantt Chart 4
1.5.2. WBS Planning for Development Phase 4
Chapter 2 5
Software Requirements Specification 6
2.1. Requirement Specification 6
2.2. Data Requirements 7
2.3. Performance Requirements 7
2.3.1. Speed and Latency Requirements 7
2.3.2. Precision and Accuracy Requirements 7
2.3.3. Capacity Requirements 8
2.4. Dependability Requirements 8
2.4.1. Reliability Requirements 8
2.4.2. Availability Requirements 8
2.4.3. Robustness or Fault-Tolerance Requirement 9
2.5. Maintainability and Supportability Requirements 9
2.5.1. Maintenance Requirements 9
2.5.2. Supportability Requirements 9
2.5.3. Adaptability Requirements 9
2.6. Security Requirements 10
2.6.1. Access Requirements 10
2.6.1. Integrity Requirements 10
2.6.2. Privacy Requirements 10
2.7. Usability and Human-Interaction Requirements 11
2.8.1. Appearance Requirements 12
2.8.2. Style Requirements 12

Chapter 3 13
System Analysis 13
3.1. Use Case 14
3.2. Use Case Description 15
3.2.1. Authenticate User 16
3.2.2. Create Appointment 17
3.3. Activity Diagram 19
3.3.1.Activity Diagram For Patient. 20
3.3.2.Activity Diagram For Doctor 21
3.4.Sequence Diagram 22
Chapter4 23
System Design Specification 23
4.1. Class Diagram 24
4.2. User Interface 25
4.2.1. Homepage 25
4.2.2. Patient Details 25
4.2.3. Book An Appointment 26
4.2.4. Doctor List 26
4.3 Development Tools & Technology 26
Chapter 5 27
System Testing 28
5.1. System Testing 28
5.1.1. Features To be Tested 29
5.1.2. Features not To be Tested 29
5.2. Testing Straegies 29
5.2.1. Login 29
5.2.2. Manage Appointment 30
Chapter 6 31
Project Summary 31
6.1. GitHub Link 32
6.2. Limitations 32
6.3. Obstacles & Achievement 32
6.4.Future Scope 32
6.3. Reference 32

Chapter 1

Introduction

1.1 Project Overview

This project is focused on how to Find-A-Doctor: An online platform. To ensure easier, reliable and faster service.

1.2 Project Purpose

Find-A-Doctor is a system of organizing the medical procedures easily. It provides the scope for patient and doctormanaging appointment. Users want an environment without any troubles for medical activities. They can register themselves easily.

1.2.1 Project Background

Find-A-Doctor is a system that provides successful help to manage the medical doctor easier for the user. In our society, every family haveat least one sick person. Almost in every three months, we need to deal appointments with consultants for various kind of diseases. For this system we can easily find a specialist doctor.

1.2.2 Beneficiaries and Benefits

This project is mainly beneficiaries for patient and doctor. The following benefits will be provided by this system-

- Patient can
  - ➢ Conduct appointment easily
  - ➢ Manage Doctor easily

- Doctor
  - ➢ Manage appointments
  - ➢ Manage all patient under his surveillance

1.2.3 Project Goals

The basic functionality of this system is making the process user friendly for doctor and patient. It helps users to control easily the medical activities like-

- Find Doctor
- Create Appointment

The main purpose of this project is to reduce time and make a reliable system. By using this system, user can find a doctor easily.

1.3 Stakeholders

There are two types of stakeholders in this system

- Patient
- Doctor
- Admin

1.4 Proposed System Model

Proposed system means explaining what developers are going to do this project. What is the project about and what is new in the project other than existing things. And how they are going to do this.

1.4.1 Agile-Model

Our proposed system model is agile model which is an incremental process of software development. Each iteration lasts one to three weeks on average. Engineering actions are carried out by cross functional teams. In software development the term ‘agile’ means the ability to respond to changes-changes from requirements, technology and people.

1.4.2 How we used Agile

1. Ideate a plan from real life concerning issues.
2. Gathering and changing userrequirements are embraced for the user’s competitive advantage.Face-to-face communication is the best way to transfer information to and from a team.
3. Concentrate on delivering working software frequently.
4. Projects must be based on people who are motivated. Give them the proper environment and the support that they need.
5. Self-organized teams usually create the best designs.Constant attention to technical excellence and good design will enhance agility.
6. Simplicity is considered to be the art of maximizing the work that is not done, and it is essential.
7. At regular intervals, the team putted their concentration on how to become more effective, and they willadjust their behavior accordingly.
8. Regular monitoring personnel were attentive to test the system.

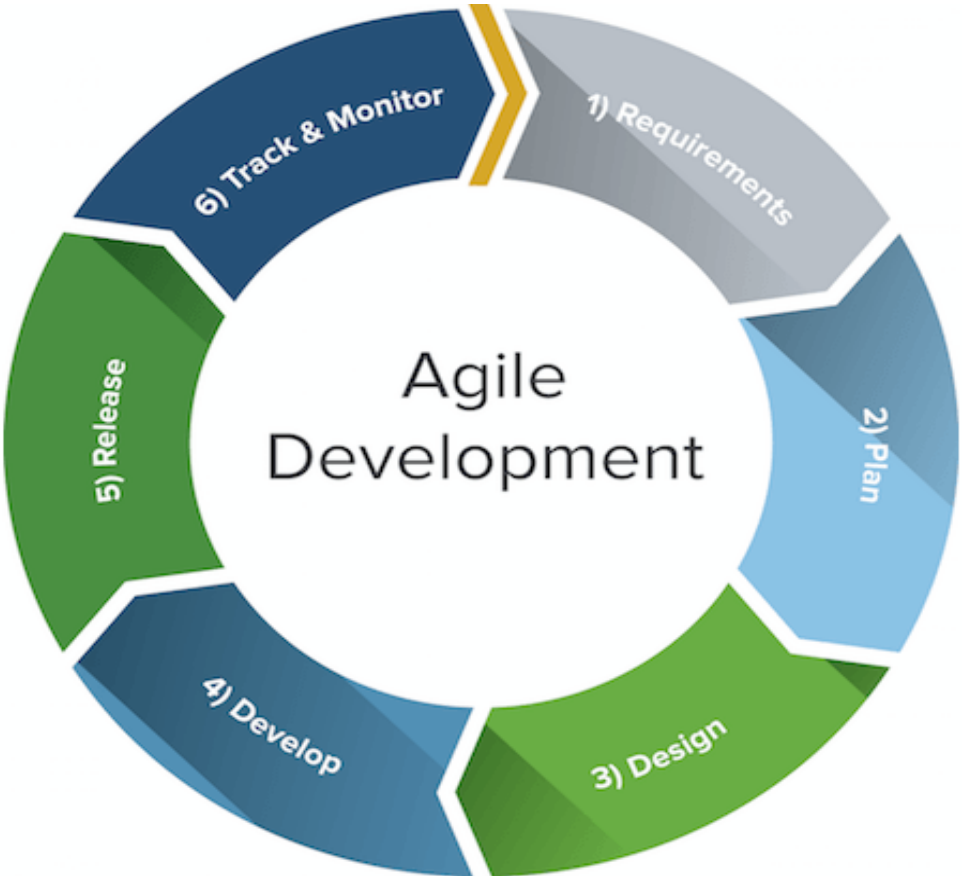


Figure 1.1: Proposed System Model (Block Diagram)

1.5 Project Schedule: In Project management a schedule is a listing of project's milestones, activities, and deliverables, usually with intended start and finish dates. A schedule is commonly used in the project planning and project portfolio management parts of project management.

1.5.1 Gantt Chart

A Gantt chart is a series of horizontal lines shows the amount of work done or production completed in certain periods in relation to the amount planned for those periods [2].

Table 1.1: Gantt Chart

Weeks Works	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Analysis Phase																		
Feasibility Study																		
Project Proposal																		
Project UI																		
Mid-Term Defense																		
Implementation of the Project																		
Testing																		
Documentation of the Project																		
Final Defense																		

1.5.2 Release plan

Release	**Version**	**Date**
1st Release	Beta version 1.0.0	25/11/2018
2nd Release	Beta version 2.0.0	10/12/2018
3rd Release	Version 3.0.0	25/12/2018
4th Release	Version 4.0.0	11/01/2019

Chapter 2

Software Requirement Specification

2.1 Requirement Specification

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform.
Requirement prioritization is used in software product management for determining which candidate requirements are high, medium and low of a software function moduleshould be included in a certain release. Requirements are also prioritized to minimize risk during development so that the most important or high risk requirements are implemented first.

Table 2.1: Functional Requirements

FRQ_ID	FRQ_Name	Description	Priority
FRQ01	Registration	Patient can registration by using this System.	High
FRQ02	Registration	Doctor can registration by using this System.	High
FRQ03	Search Doctor	Patient can search doctor	High
FRQ04	Manage Appointment	Doctor and patient can manage appointment	Low

2.2 Data Requirement:

Table 2.2: Data Requirements

No	Description	Priority
DR01	Patient name, age, weight, contact, address.	High
DR02	Doctor name, designation, specialty, address.	High

2.3 Performance Requirements

2.3.1 Speed and Latency Requirements

Table 2.3: Speed and Latency Requirements

No	Description	Priority
SLR01	When system requires the code for registration it should send verification code to entered number.	High
SLR02	When patient send appointment request to the system, Firebase's database should send info to server in real-time	High
SLR03	When doctor accept appointment request, Firebase's database should add info to server in real-time	High

2.3.2 Precision and Accuracy Requirements

Table 2.4: Precision and Accuracy Requirements

No	Description	Priority
SLR01	The input data should be accurate when Patient or Doctor provide data to the system.	High
SLR02	All data should be in place accurately where it is associated	Low

2.3.3. Capacity Requirements

Table 2.5: Capacity Requirements

No	Description	Priority
CR01	The mobile application size must able to load at hosting site.	Medium
CR02	The Firebase database size must be able to store the system data.	Low
CR03	System should support 100k user at the beginning version.	Medium
CR04	System should support 1000 request per second.	Medium

2.4 Dependability Requirements

2.4.1 Reliability Requirements

Table 2.6: Reliability Requirements

No	Description	Priority
RR01	All confidential data must have to be encrypted.	Medium
RR02	All data should collect from users by permission and by accepting privacy policy	Low
RR03	No one can use user's data for any other purpose except system needs.	Low

2.4.2 Availability Requirements

Table 2.7: Availability Requirements

No	Description	Priority
AR01	The system should work 24 hours a day.	Medium
AR02	The system should provide the desired data to the user in time.	Low

2.4.3 Robustness or Fault-Tolerance Requirement

Table 2.8: Robustness or Fault Tolerance Requirements

No	Description	Priority
FTR01	If the system has been crashed, it should not be more than an hour.	Low

2.5 Maintainability and Supportability Requirements

2.5.1 Maintenance Requirements

Table 2.9: Maintenance Requirements

No	Description	Priority
MR01	The system maintenance should be quick.	Low

2.5.2 Supportability Requirements

Table 2.10: Supportability Requirements

No	Description	Priority
SR01	The system should support latest Android Version and Firebase version.	Low

2.5.3 Adaptability Requirements

Table 2.11: Adaptability Requirements

No	Description	Priority
AR01	The system should adapt all upgrading version and time.	Low
AR02	New version of system should support latest Firebase modules.	Low

2.6 Security Requirements

2.6.1 Access Requirements

Table 2.12: Access Requirements

No	Description	Priority
AR01	User's accesshave to be limited with their use case boundaries.	Low
AR02	Users need to be authorized first to access data.	Medium
AR03	Only Administrative authority will be able to enter the system to make maintenance.	Low
AR04	User's boundaries should be within the system	Medium

2.6.2 Integrity Requirements

Table 2.13: Integrity Requirements

No	Description	Priority
IR01	Only authorized user can update data with their respective accessibility and authorization.	Medium
IR02	Only administrative authority can access and update or delete user's information.	Medium

2.6.3 Privacy Requirements

Table 2.14: Privacy Requirements

No	Description	Priority
PR01	The user data should not contain any private issues.	Medium
PR02	All the confidential data should be encrypted.	Low

2.7 Usability and Human-Interaction Requirements

Table 2.15: Usability Requirements

No	Description	Priority
UR01	Patient request for appointment and Doctor can approve the appointment requests.	Medium
UR02	Doctor can save prescriptions and test document details using OCR and can those for next instructions.	Medium
UR03	Patient can take medicine and observe medicine quantities.	Medium
UR04	Patient can get daily instruction such as appointments, consuming medicine alarm notifications.	Medium

2.8 Look and field Requirements

2.8.1 Appearance Requirements

Table 2.16: Appearance Requirements

No	Description	Priority
AR01	The user interface must be attractive.	High
AR02	The user interface must be user friendly.	Medium
AR03	The user interface must be user interactive with user experiences.	High

2.8.2. Style Requirements

Table 2.17: Style Requirements

No	Description	Priority
SR01	The interface color should be material.	Low

Chapter 3

System Analysis

3.1 Use case

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. It is a sequence between system and user.Use case diagrams are employed in UML(Unified Modeling Language), a standard notation for the modeling of real-world objects and systems.

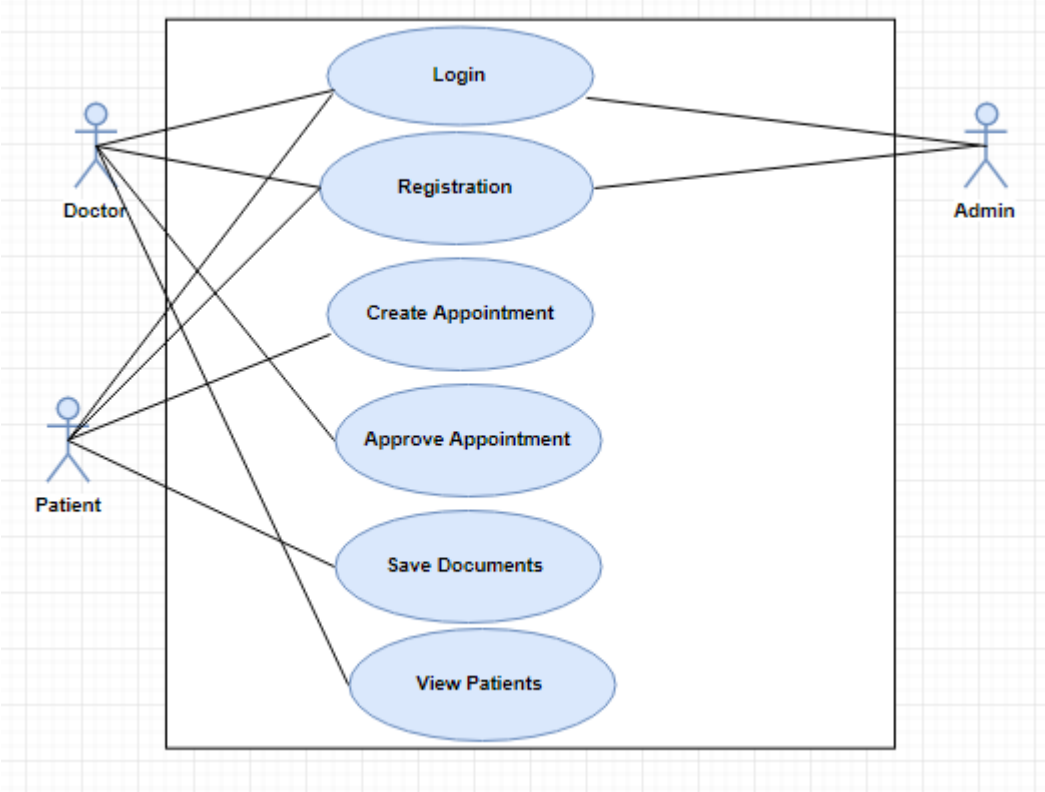


Fig 3.1: Use case Diagram

3.2 Use Case Description:

3.2.1 Authenticate user: Authentication is a process that's allow a device to verify the identity of someone who connect to the system.

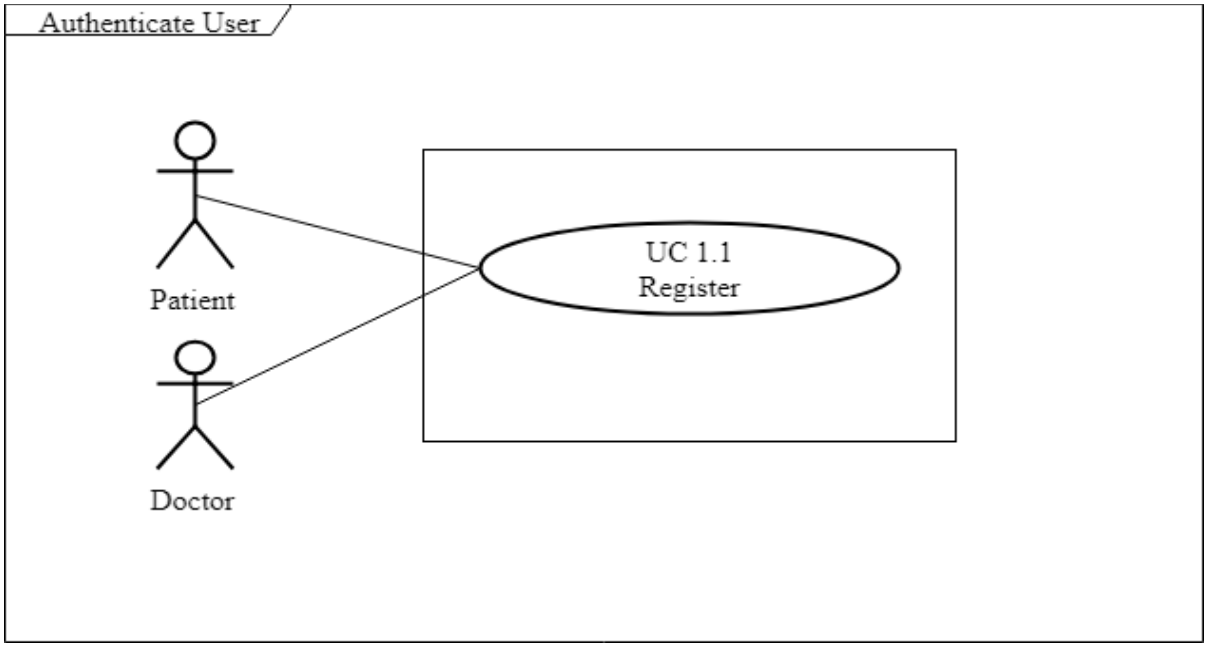


Fig 3.2.1: Authenticate User

Table:3.1: Authenticate User

Use Case Name	**Authenticate User**
Scenario	User can register
Actor	1.Patient 2.Doctor
Brief Description	Patient can search doctor, hospital. area

3.2.2 Create Appointment: Patient can create an appointment. Patient can login this system then create an appointment. Select doctor, area & hospital.

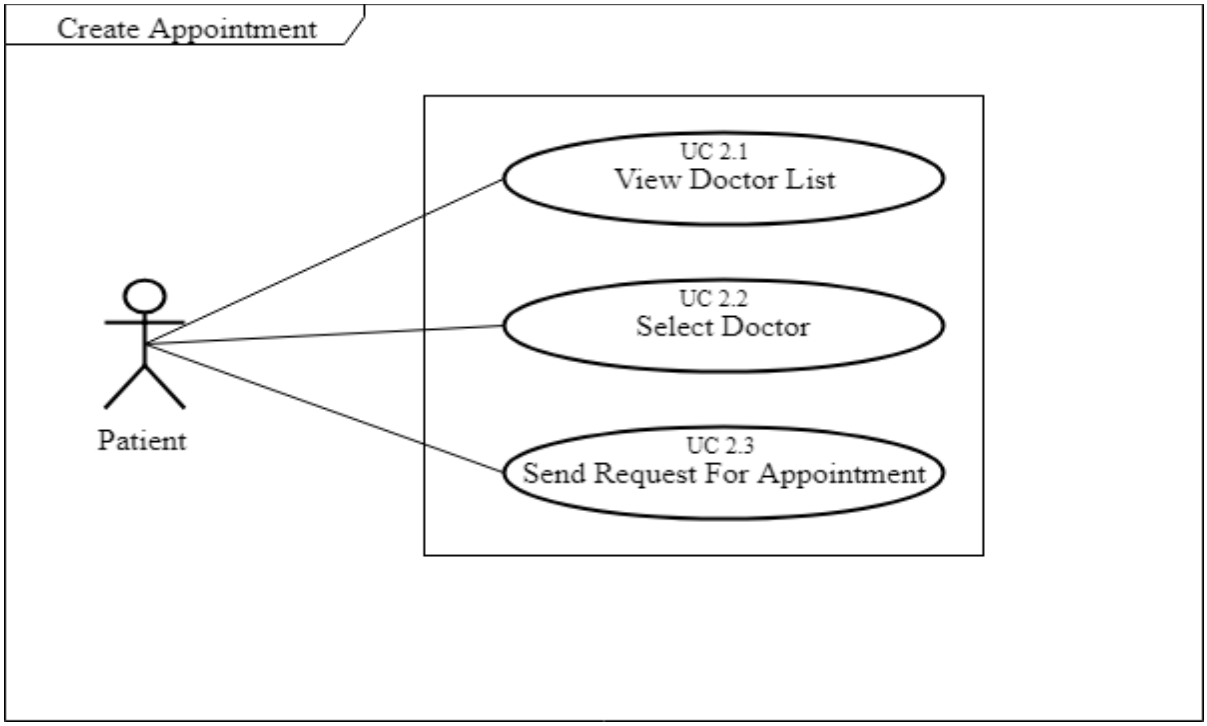


Fig3.2.2 : Create Appointment

Table 3.2: Create Appointment

Use Case	Create Appointment
Scenario	Patient can creates appointment
Main Success Scenario	1. Patient can search doctor. 2. Patient can search hospital.
Actor	Patient

3.2.3 Approve Appointment: Doctor can approve appointment**.** Doctor see the patient details. When doctor see the patient request for an appointment he approve the request. User can view doctor list.

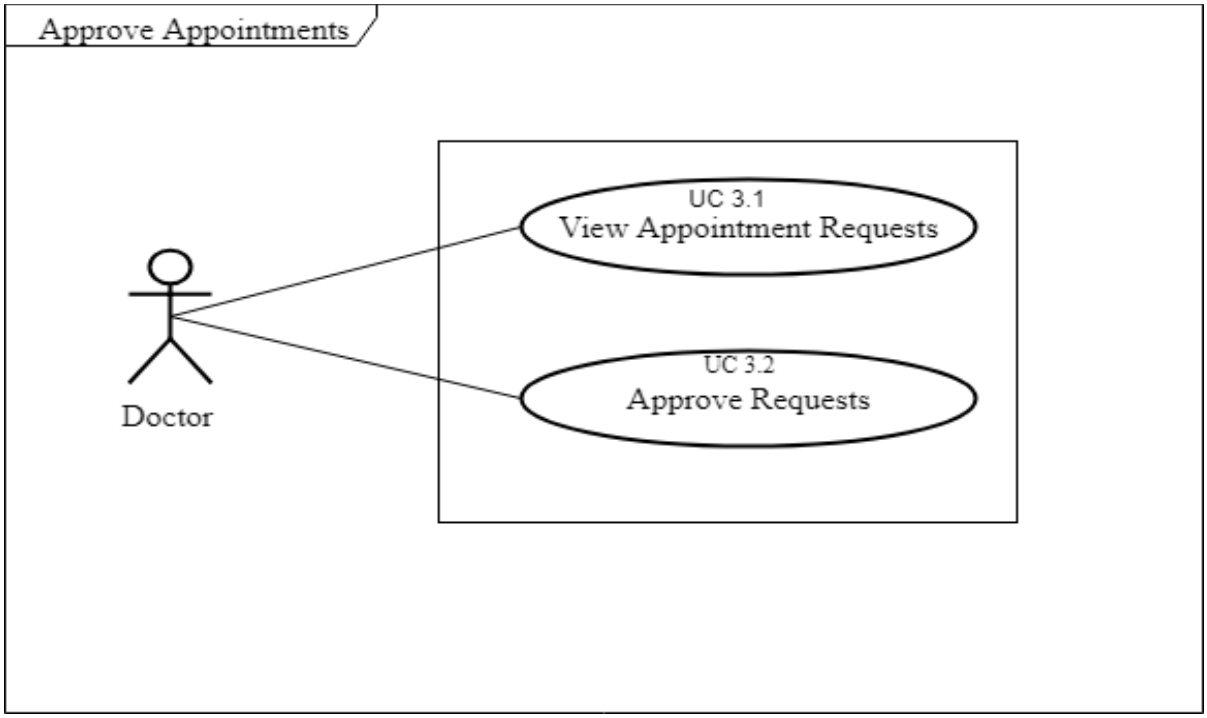


Fig 3.2.3:Approve Appointment

Table 3.3: Approve Appointments

Use Case Name	Approve Appointments
Scenario	Doctor can approve Appointment
Brief Description	User can view doctor list and select a Doctor and can send an appointment request.
Actor	● Doctor
Precondition	Doctor has to register to the system.
Post condition	Doctor must choose this method.
Main Success Scenario	● Doctor can view appointment requests. ● Doctor can approve appointment.
Scenario Extensions	● If doctor isn't registered, they cannot send request. ● Proper registration procedure must needed either they can approve a request.

3.2.4 Save Documents: Doctor can save documents patient can see the document.Doctor & patient can view documents.

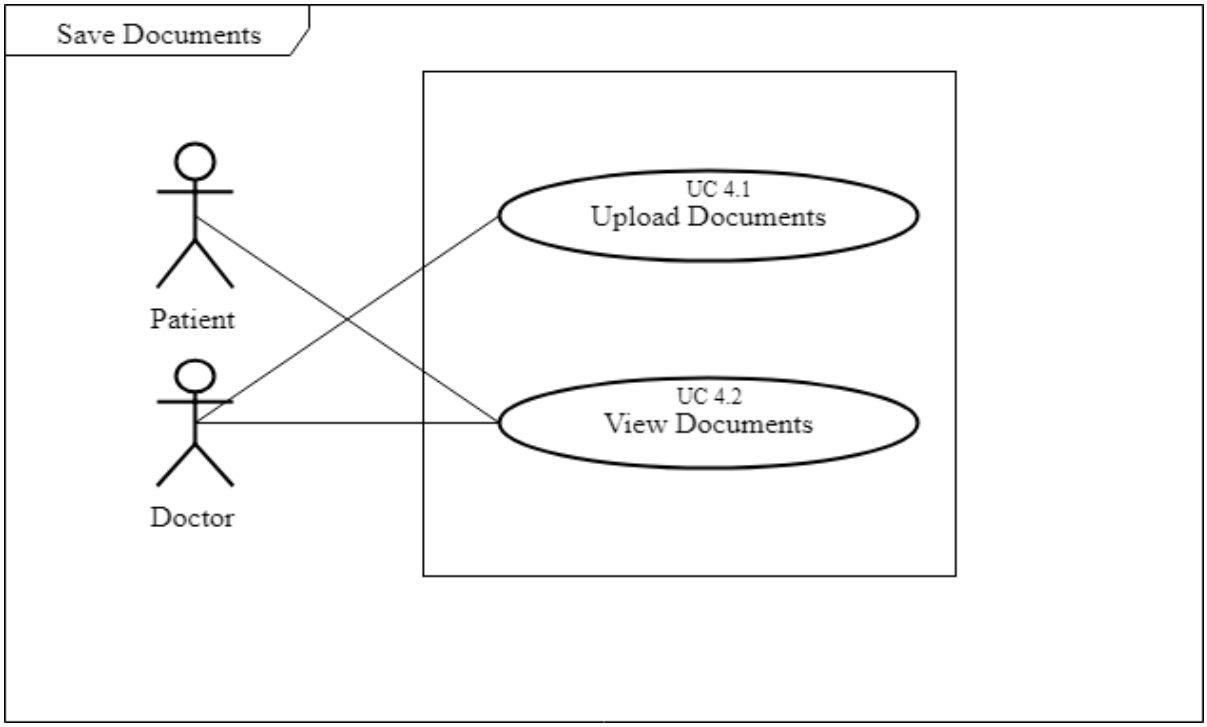


Fig 3.2.4: Save Documents

Table 3.4: Save Documents

Use Case Name	Save Documents
Scenario	Doctor can save documents and also patient can view documents.
Brief Description	Doctor can save documents, Doctor and Patient can view those documents for prescription.
Actor	● Patient ● Doctor
Precondition	Doctor has to save documents to the system.
Post condition	Doctor must choose this method.
Main Success Scenario	● Doctor can save documents. ● Patient can view those documents.
Scenario Extensions	● If patient didn't save documents, doctor cannot view documents.

3.3 Activity Diagram:

3.3.1 Activity Diagram for patient: Patient can registration can login this system.Patient can view doctor list. Select doctor, hospital & area.

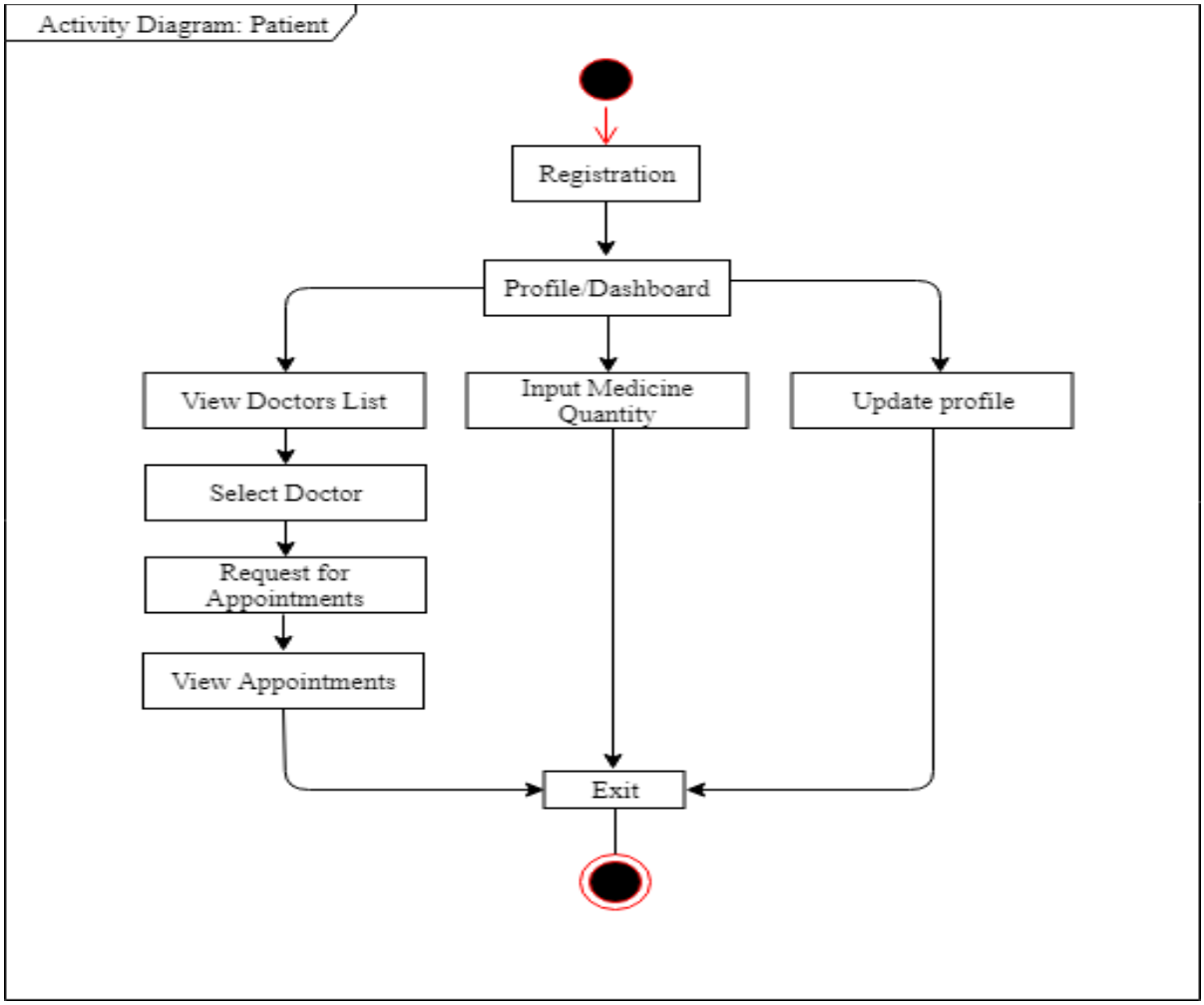


Fig 3.2.1: Activity Diagram for patient

3.3.2 Activity Diagram For Doctor: Doctor also can registration than he can login this system. He can view appointment list.

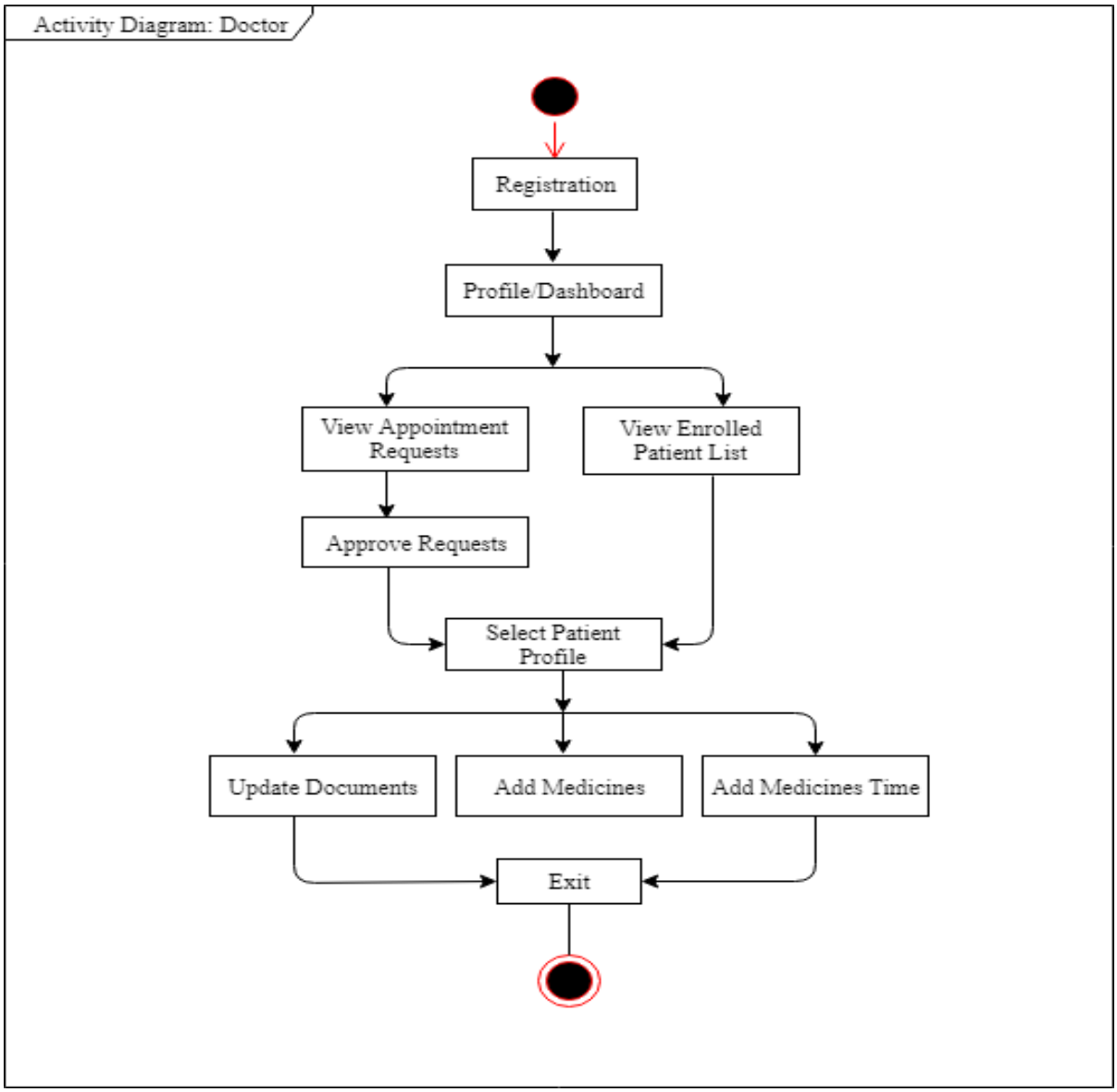


Fig 3.2.2: Activity Diagram For Doctor

3.4. Sequence Diagram:

UML Sequence diagrams are interaction diagrams that detail how operation are carried out.

3.1.1. System sequence diagram: Authenticated User:Doctor

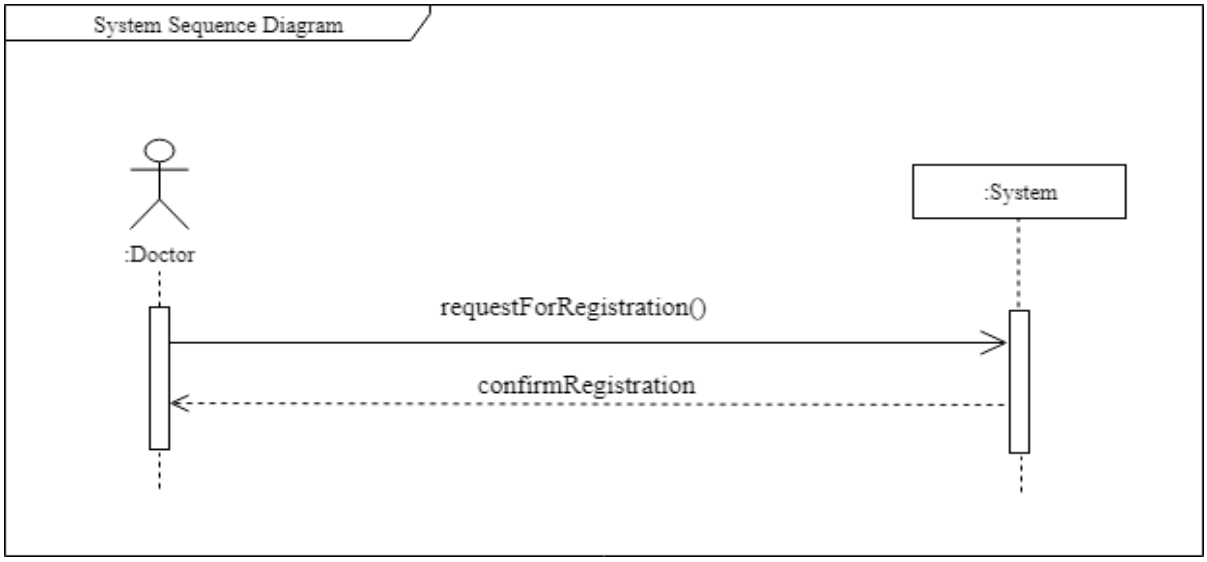


Fig 3.1.1: Authenticated User (Doctor)

3.1.2.System sequence diagram: Create Appointment

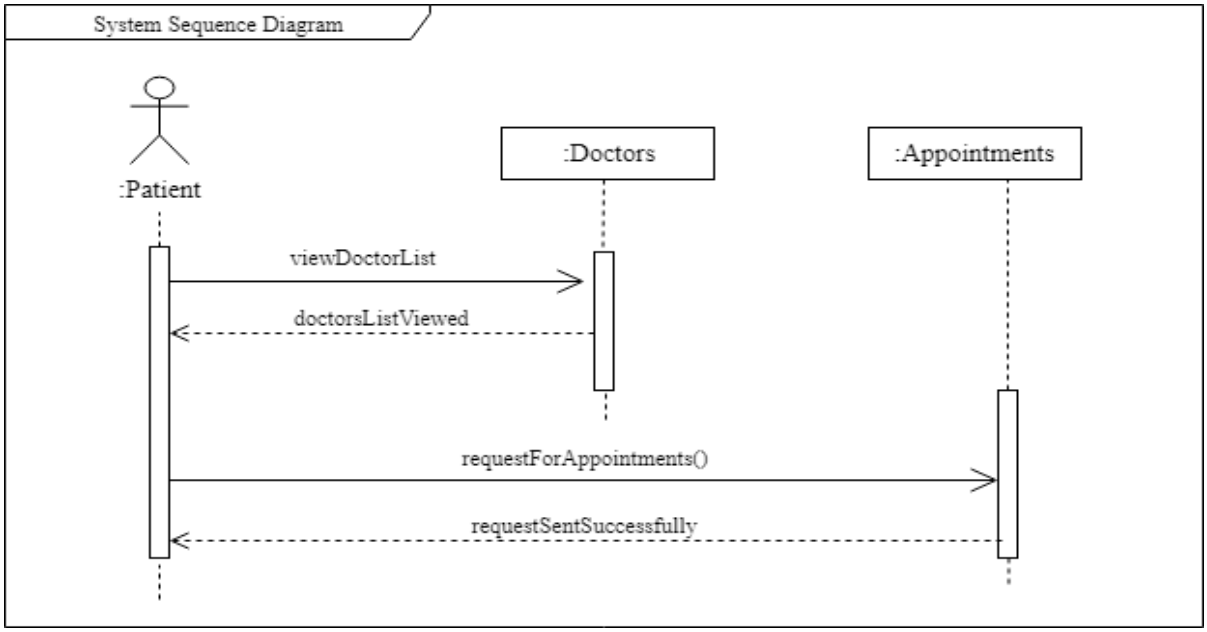


Fig 3.1.2: Create Appointment

3.1.3.System sequence diagram: Approve Appointment

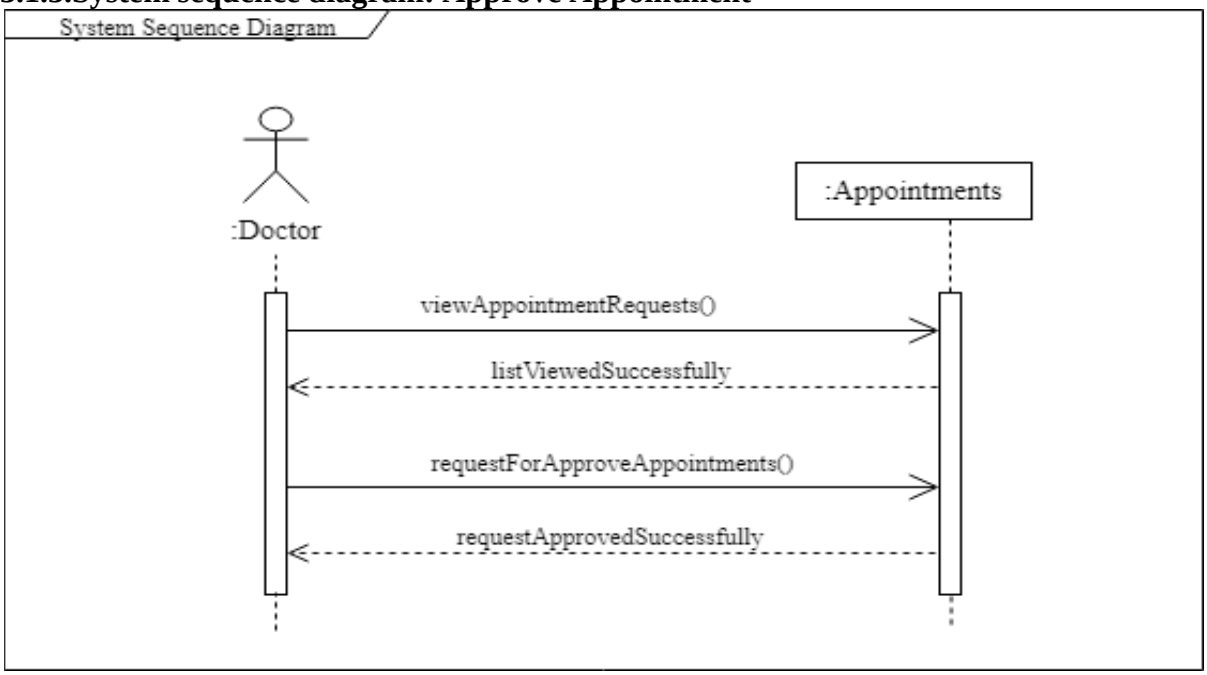


Fig 3.1.3:Approve Appointment

Chapter 4

System Design Specification

4.1. Class Diagram:

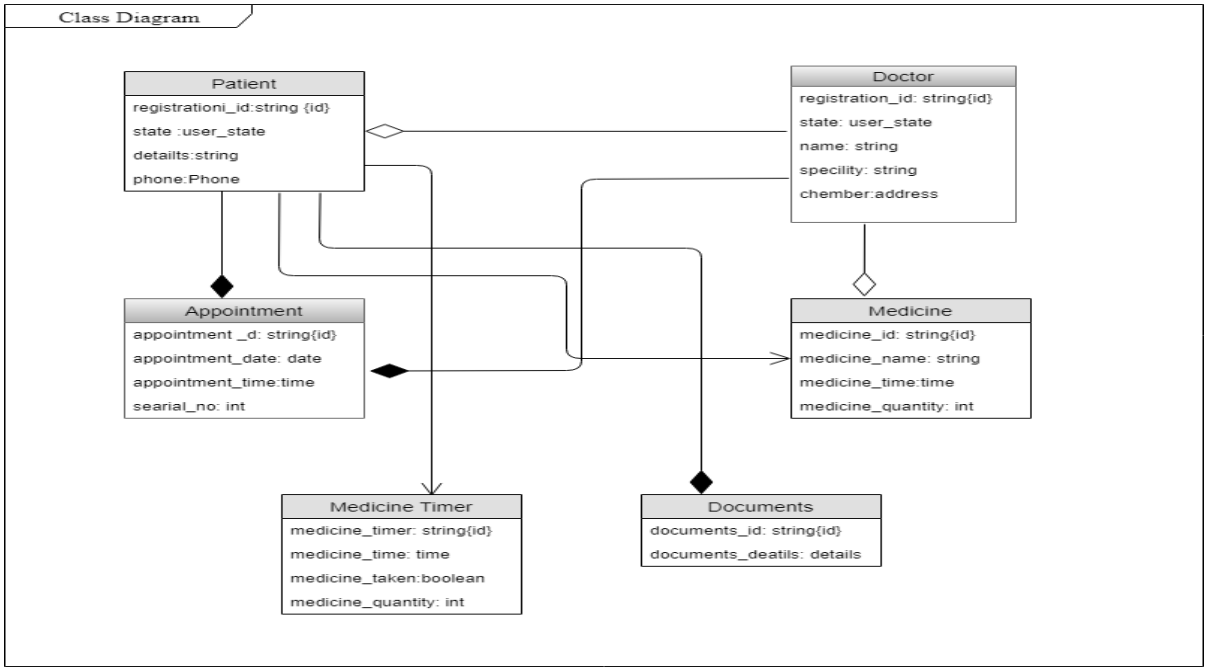
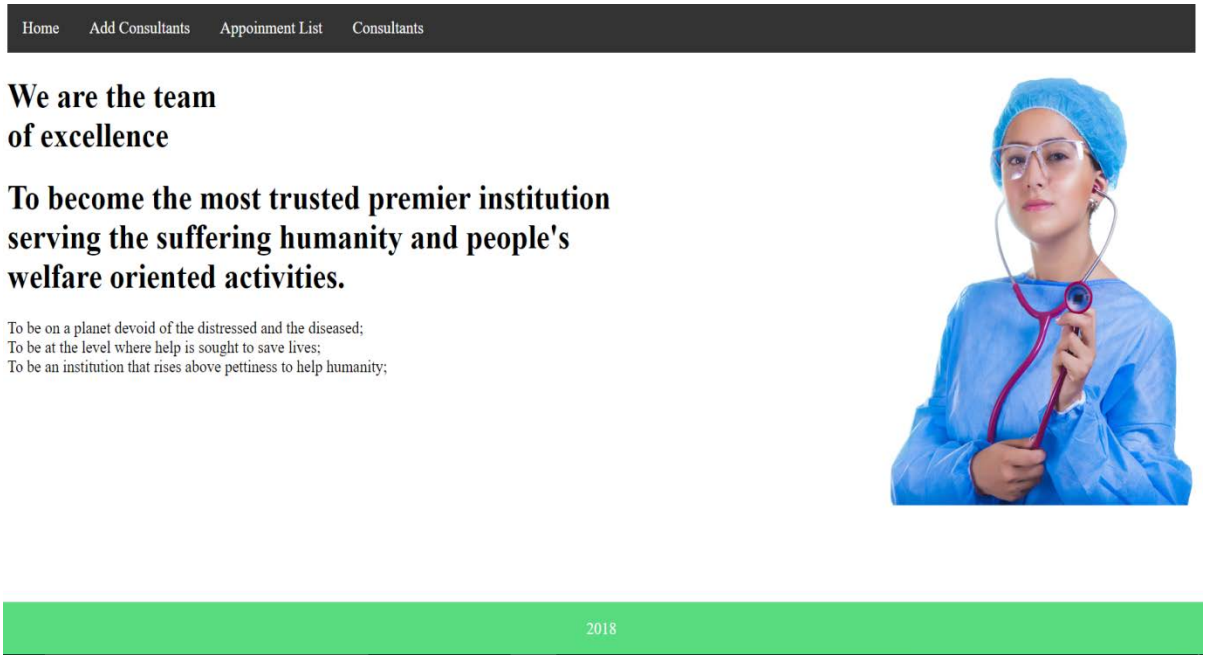


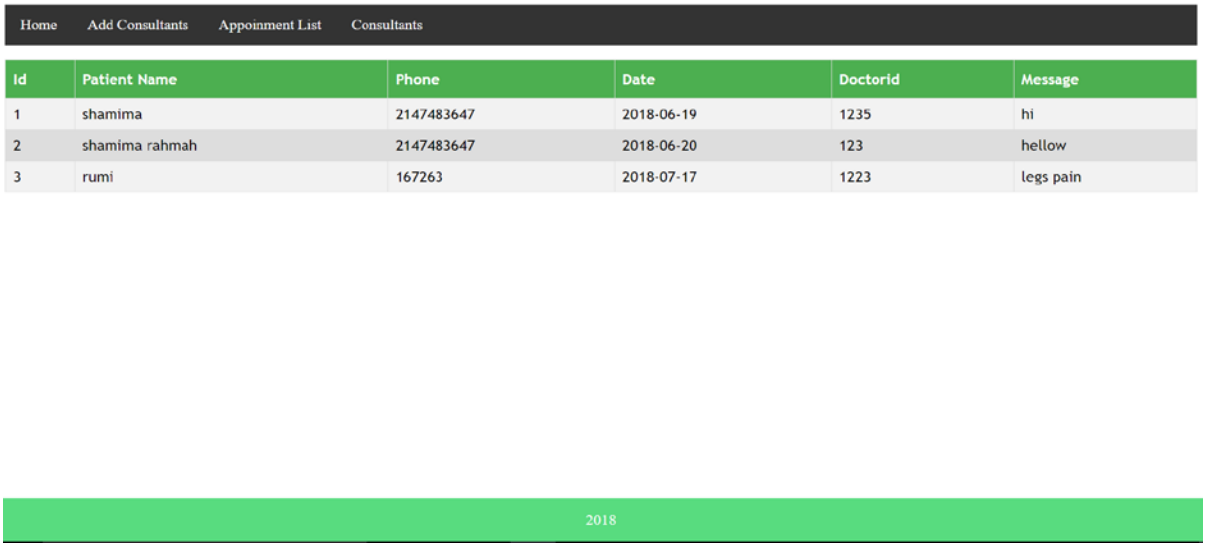
Fig 4.1: Class Diagram

4.2User Interface:

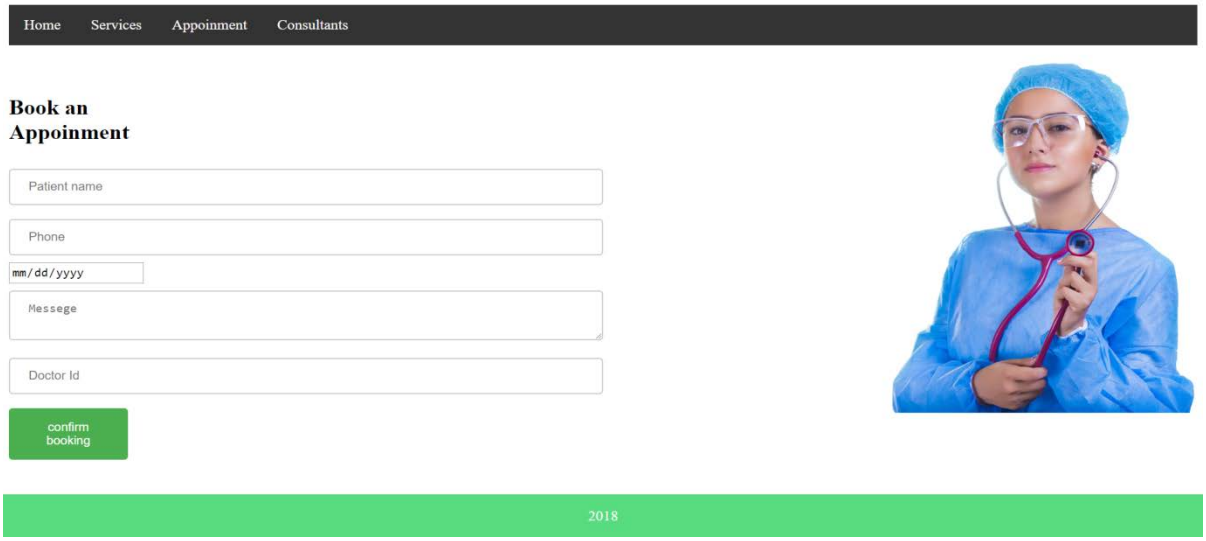
4.2.1 Homepage: WhenAdmin, Doctor, Patient login than they can view this homepage.



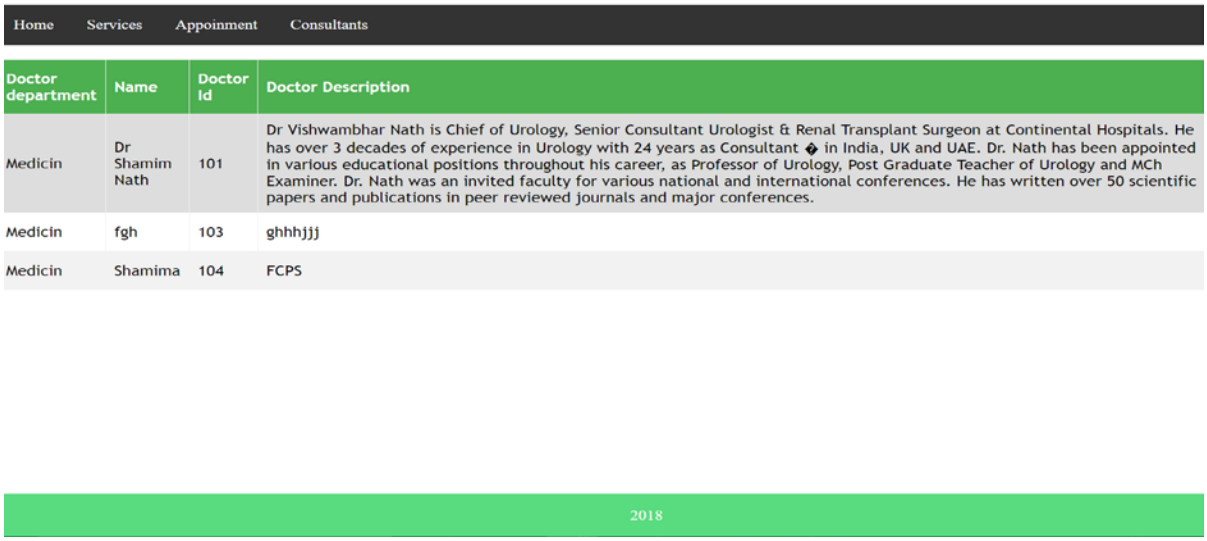
4.2.2 Patient Details: When doctor login he can see this list.



4.2.3 Book An Appointment: First of all Patient login this system than they can book an appointment.



4.2.4 Doctor List: Patient can view doctor list. They can choose doctor.



4.3.Development Tools & Technologies

4.3.1. User Interface Technologies:

- Bootstrap
- PHP
- Google Chrome
- XAMPP

Chapter 5

SystemTesting

5.1 System Testing: Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

5.1.1 Features to be tested

- Registration
- Manage Appointment
- Doctor's Details

5.1.2 Features not to be tested

- Login
- Patient Details
- Services

5.2 Testing Strategies:

5.2.1 Login

Test Case: TC01	Test Case Name: Testing the login panel.
System: Find-A-Doctor	Requirements ID: FRQ_01 (Login)
Designed by: ShamimaAkter	Design Date: 31.03.19
Executed by: ShamimaAkter	Execution Date: 31.03.19
Short Description: This field will handle the login functionality of the system.	
Precondition: Open the application.	

Table 5.1:Test case of login

Steps	Action	Input	Result	Pass/Fail
01	Enter all information in login page	Id, password	Sign up	pass
02	Valid id & invalid password	Id, password	Sign up	Fail

5.2.2 Manage Appointment

Test Case: TC02	Test Case Name: Manage Appointment
System: Find-A-Doctor	Requirements ID: FRQ_02 (Manage Appointment)
Designed by: ShamimaAkter	Design Date: .03.04.19
Executed by: ShamimaAkter	Execution Date: 03.04.19
Short Description: This field will handle the login functionality of the system.	
Precondition: login the application.	

Table 5.2: Test Case of Manage Appointments

Steps	Action	Input	Action Result	Expected System Response	Pass/Fail
01	Select doctor from doctor list and send a request.	Select doctor list	Doctor selected and send appointment request.	Send an appointment request.	Pass
02	Approve appointment.	Click approve	Not added in enrolled list.	Added in enrolled list.	Fail
03	Approve appointment.	Click approve	Added in enrolled list.	Added in enrolled list.	Pass

Chapter 6

Projectsummery

6.1. GitHub link:https://github.com/SAktersonia/Appointment

6.2. Limitations:

- ❖ This system has only few functionality and scope for international level
- ❖ This system can't show real time .

6.3. Obstacles & Achievements:

Obstacles:

- Learning new technology and environment
- Limited time and budget

Achievements:

- Learnt new technology
- Successfully built a project

6.4. Future Scope:

- Real time Communication
- Notification
- Payment gateway

References:

- (1998). W3School. Retrieved 05-03-2019, from https://www.w3schools.com/
- John resig. (2006). Jquery. Retrieved 05-04-2019, from http://jquery.com/
- Chad hurley, stevechen, jawed karim. (2005). YouTube. Retrieved 05-02-2019, from https://www.youtube.com/
- Mark otto, jacobthornton. (2011). Bootstrap. Retrieved 05-04-2019, from https://getbootstrap.com/
- Jeff Atwood, Joel Spolsky. (2008). Stackoverflow. Retrieved 05-04-2019, from https://stackoverflow.com/