



SOEN-6471 ADVANCED SOFTWARE ARCHITECTURES(SUMMER 2020)

iCARE

DELIVERABLE 1 (D1)

Submitted By: (Team D)

Niralkumar Lad(40080612)
Moshood Kolawole (40102157)
Ahmad Memari (40088010)
Xiaofeng Liu (40126183)
Anusha Keralapura Thandavamurthy (40102962)

Submitted To:

Dr. Pankaj Kamthan

GitHub – [https : //github.com/Anushakt/SOEN₆₄₇₁softwareArchitecture](https://github.com/Anushakt/SOEN6471softwareArchitecture)

TABLE OF CONTENT

PART 1: VISION	1
Project Overview	1
Project Background	1
Business Goal	1
PART 2: STAKEHOLDERS AND CONCERNS	1
Mind Map	2
List of Stakeholders	2
Concern	3
PART 3: PROBLEM	3
Functional Requirements	4
Non-Functional Requirements	5
PART 4: VIEWPOINTS AND VIEW	6
Logical View	7
Implementation View	8
Process View	8
Deployment View	12
Use Case View	13
PART 5: ARCHITECTURAL DECISIONS	14
Architectural Decision 1: Use the MVC architecture	15
Architectural Decision 2: Architectural tactics	16
Architectural Decision 3: Principles	17
Architectural Decision 4: Styles	17
PART 6: IMPLEMENTATION	19
REFERENCES	29

List of Figures

Mind Map	2
4+1 View	7
Package Diagram	7
Component Diagram	8
Class Diagram	9
Communication Diagram for Patient	10
Communication Diagram for Doctor	10
Communication Diagram for Admin	10
User Login Sequence Diagram	11
Appointment Booking Sequence Diagram	11
Sign Up Sequence Diagram	12
View Doctor Details Sequence Diagram	12
Deployment Diagram	13
Use Case Diagram	14
Activity Diagram	14
MVC Pattern for iCARE	16
Layered Architecture Style	18
Homepage	21
Patient Sign Up	22
Patient Dashboard	23
Book Appointment	24
Doctor Profile	25
Doctor Dashboard	26
Staff Registration	27
Schedule Appointment	28
Admin Dashboard	29

PART 1: VISION

Project Overview

The iCare project is going to be solely implemented for patients, doctors and admin. In detail, the main function of the system is to register and select a doctor to make the appointment completely online. Patient will be able to access their appointment history and make/reschedule an appointment specific to their problem through the web app.

Project Background

With the COVID-19 pandemic in 2020, Health Informatics is becoming more important than ever. Online Appointment System can greatly improve patients' experience as well as reducing the workload on doctors and medical system.

Business Goal

To design and develop an Online Booking Appointment for iCare which allows a potential patient to book an appointment with the doctor directly through the website. The system will:

1. Be accessible on any mobile device with an internet connection within Canada.
2. Reduce the workload on medical system staffs greatly.
3. Visualize the appointment schedule so that it is easy to manage appointments and make the workflow less cluttered.
4. Offer patients greater choice and convenience, and remove the opportunity for double-bookings.
5. Design and implement the appointment list so that it is easy to manage appointments and make the workflow less cluttered.
6. Improve data capturing and reporting.

PART 2: STAKEHOLDERS AND CONCERNS

Mind Map:



Mind Map

List of Stakeholders:

1. Software Provider Organization:

(a) Owner.

(b) Employees:

- Developers.
- Maintainers.
- Clerks.

2. Medical Organisation:

(a) Medical Staff:

- Doctors.
- Nurse.
- Lab Operators.

(b) Administrative Staff:

- Managers.
- Clerks.

3. Customer:
 - (a) Patient.
 - (b) Patient next of kin.
 - (c) Patient assistant.
4. Black Hat Hacker.
5. Competitor.

Concerns:

1. **Patient:** They want to book appointments with doctor before visiting the clinic.
2. **Doctor:** They want to view and follow-up appointments with their patients, create and update their patient's records from anywhere easily.
3. **Receptionist/Administrator:** They want to register new patients and make changes to patients' information from anywhere.

PART 3: PROBLEM

Functional Requirements:

1. When the user, i.e. patient, doctor or admin, will try to login with the username and password, the system will match the credentials with those in the database and grant access to the system and display the dashboard if the credentials are valid and if the credentials are invalid, an appropriate message will be displayed.
2. When the patient have logged in and when the patients click on the 'Doctor Profile' button on dashboard to view doctor's profile, the system will show a list of all profile of the doctors available in the database. Here, the patient can click on the 'Filter' button to view the doctor's profile based on doctor's field of expertise as well as name.
3. While on dashboard, when the patients click on 'Book Appointment' button, the system will show a list of available date and time-slots of the available doctors.
4. Further, to book an appointment, the patient shall have to select a time-slot for a particular doctor and confirm the appointment to which the system will prompt the user to again confirm the appointment to avoid any confusion or mistake.
5. When the patient successfully book an appointment with a doctor, an email notification as well as a notification within the system will be send to the doctor and the patient.
6. When patient will be logged in and when they are on the dashboard, patient will be able to view the timeline which will depict written prescription by the doctor, the medical reports as well as the treatment plan.
7. While the patient have logged in and when the patient clicks on 'My Profile' button, they will be able to see their respective profile and can update their profile, if necessary.
8. When the doctors have logged in and they click on 'Patient Profile' button on the dashboard, doctor will be able to view patient's profile who are consulting them.
9. While viewing patients' profile, if the doctor select any particular patient's profile then the doctor will be able to enter new details about that specific patient's health if there is none or the doctor will be able to update the details. To save the details, the doctor will have to confirm the details when prompted.
10. When the doctor logged in and when they click on 'View Appointments' button on the dashboard, doctor will be able to view all the appointments of the patients who are consulting them in Chronological order.

11. When the doctor is logged in and on the view appointment page, they can select and reschedule any appointment and that patient will be notified about the event.
12. When the patient have logged in and when they click on 'View Appointments' button on the dashboard, they will be able to view upcoming appointments.
13. The patient will also be able to reschedule an appointment by clicking on the 'Reschedule Appointment' on the view appointment page but only with staff's or doctor's approval and the patient will receive the notification.
14. When a user, i.e. patients accessing the system for the first time, then they will have to click on 'Sign Up' to create an account before they are able to access the system facilities but the staff will be registered into the system by the admin.
15. When a user are on sign up page, they will have to fill up the mandatory details and click on 'Sign Up' button to continue using the system.

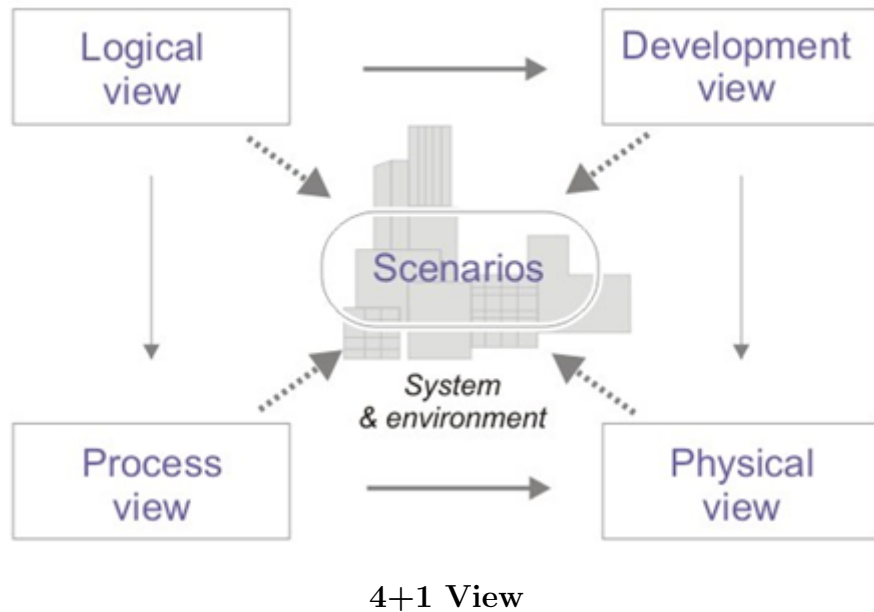
Non-Functional Requirements:

1. Maintainability:
 - (a) A Database server to store and manage the data.
 - (b) Backing up the database to a backup server at every midnight to restore the data in an event of server failure.
 - (c) Logging every event to keep a track in text form.
 - (d) Managing timestamp every time the data is changed.
2. Privacy:
 - (a) Only authorized users can log in/sign up to the system.
 - (b) Patients data can only be viewed by concerned doctors.
 - (c) Only admin will have access to give authorization for the staff.
 - (d) Patient's personal and medical data is not shared with anyone.
 - (e) Any change in the patient's profile should be notified.
 - (f) Privacy of communication channels, input interfaces, and secure storage of sensitive data.
3. Security:
 - (a) Use encryption to avoid bots from booking.
 - (b) Register with Social Security Card check.
 - (c) Ensure the appointment can only be viewed and modified by the corresponding patient and doctor.
 - (d) Set expiration time when users no longer use the website.

4. Usability:

- (a) Giving a positive experience to the user.
- (b) User should be guided for all process to complete all tasks required by the user.
- (c) Minimize the need to have a tutorial to use the system.
- (d) Easy to remember, rapid recovery from error and increasing productivity and profitability.
- (e) Converting the maximum number of visitors of the system to clients.
- (f) Different sign up/login for different categories of users.
- (g) Sitemap to help user find the related information easily.

PART 4: VIEWPOINTS AND VIEW

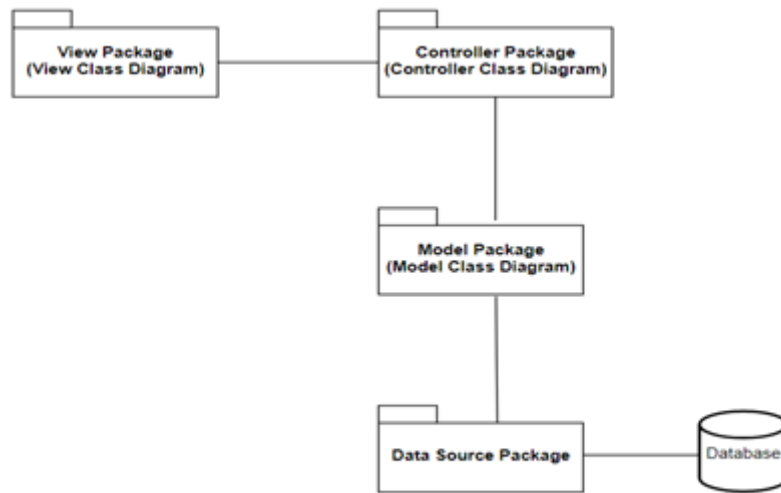


Logical View:

The logical view is concerned with the functionality that the system provides to end-users. UML Diagrams used to represent the logical view include Class diagram, and interaction diagrams (communication diagrams, or sequence diagrams). Icare website allows the end user to book an appointment with doctors and share their health concerns with them. Another functionality is that the doctor can accept/reject the appointment and give suggestions to the patient by updating their profile.

Stakeholders: Designers.

Representation: Package Diagram.



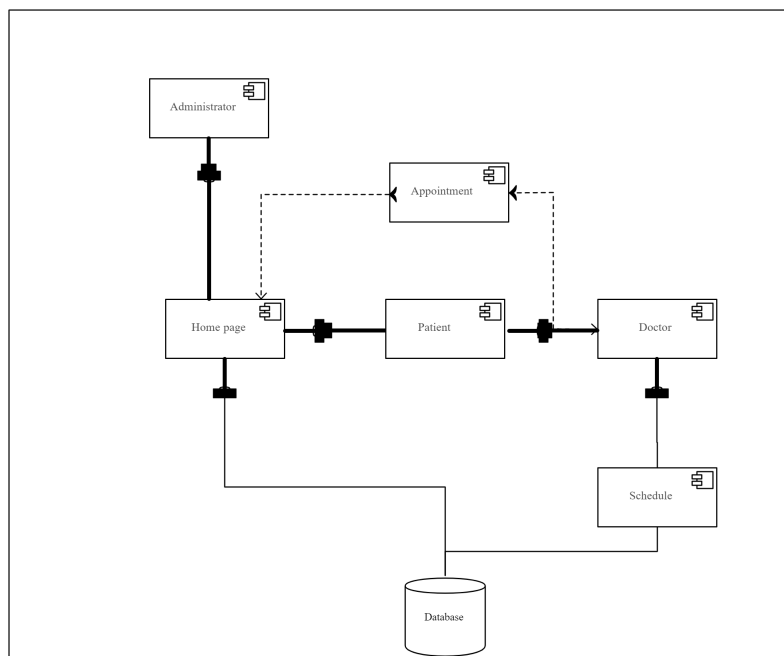
Package Diagram

Development View or Implementation View:

The development view illustrates a system from a programmer's perspective and is concerned with software management. This view is also known as the implementation view. It uses the UML Component diagram to describe system components.

Stakeholders: Programmers.

Representation: Package diagram, Component diagram



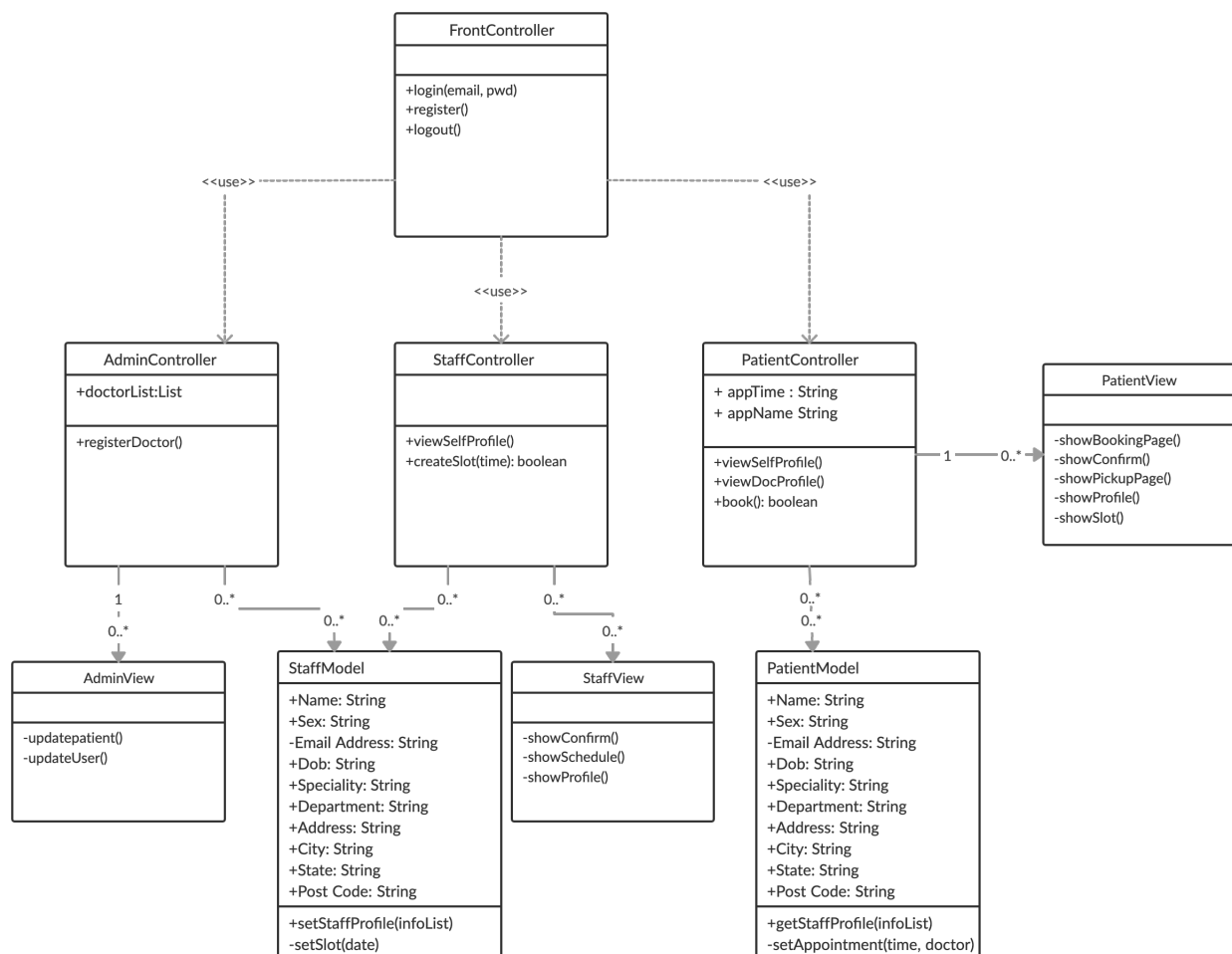
Component Diagram

Process view:

The process view deals with the dynamic aspects of the system, explains the system processes and how they communicate, and focuses on the run-time behaviour of the system. The process view addresses concurrency, distribution, integrators, performance, and scalability and so on.

Stakeholders: Integrators

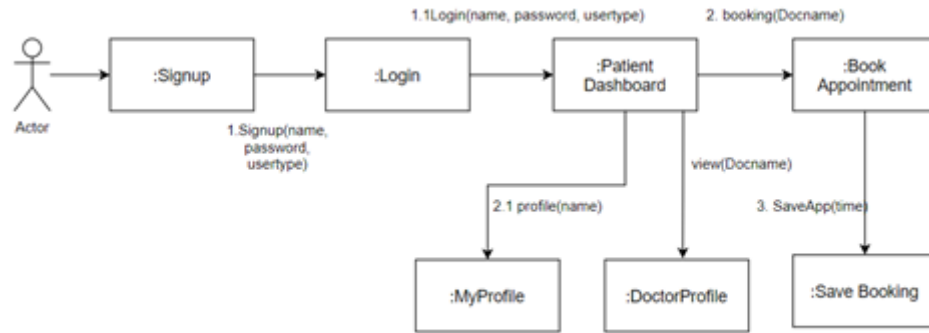
Representation: Class diagram and Interaction Diagram (Communication Diagram or Sequence Diagram)



Class Diagram

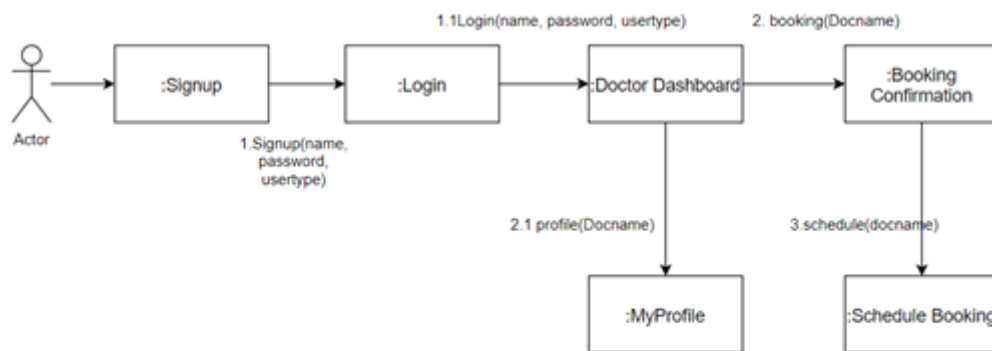
The parts for Communication diagram are divided into 3 phases:

1. **Patient Functionalities:** Book an appointment with a doctor.



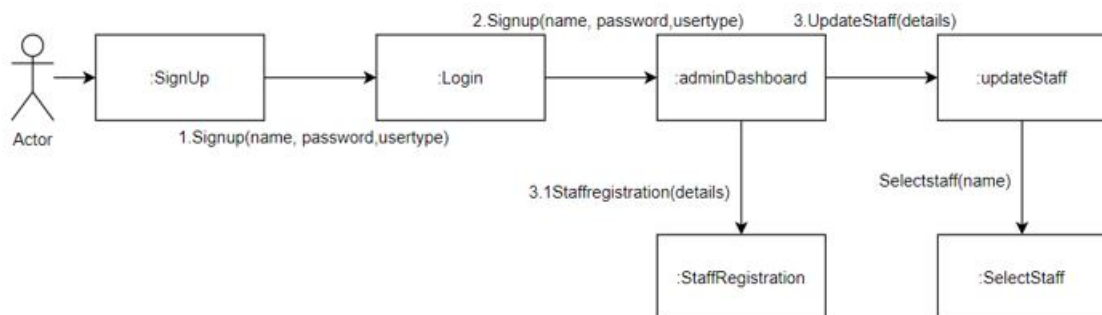
Communication Diagram for Patient functionality

2. **Doctor Functionalities:** Schedule appointment and update patients' profile.

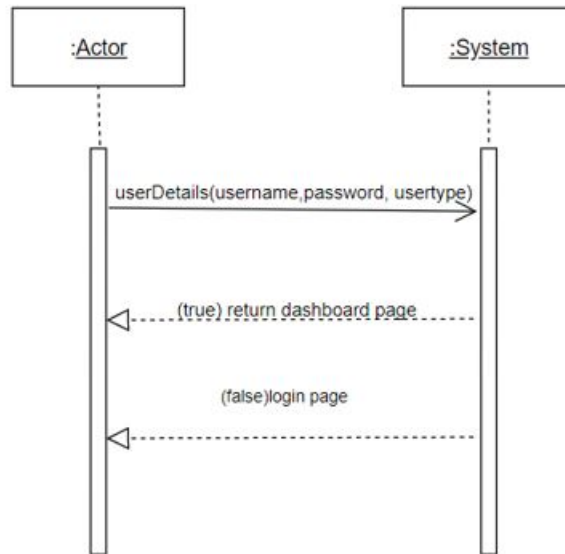


Communication Diagram for Doctors functionality

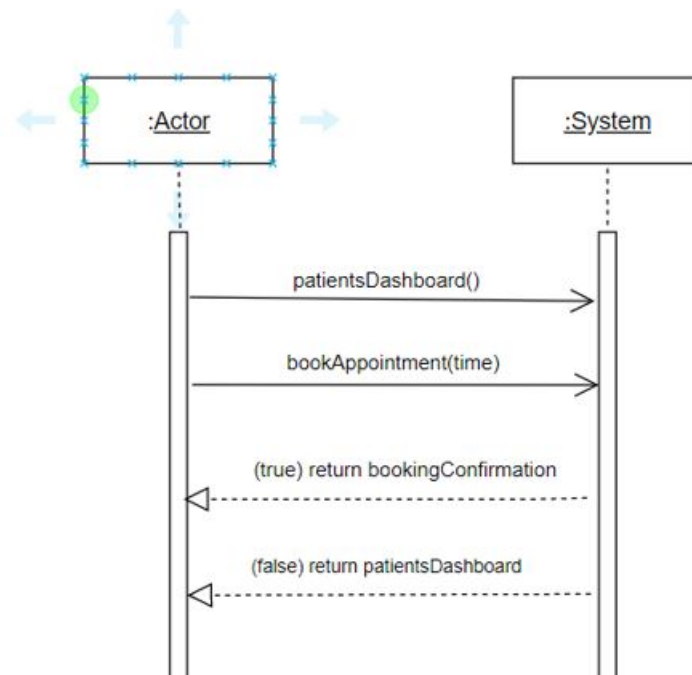
3. **Admin Functionalities:** To add and update user details.



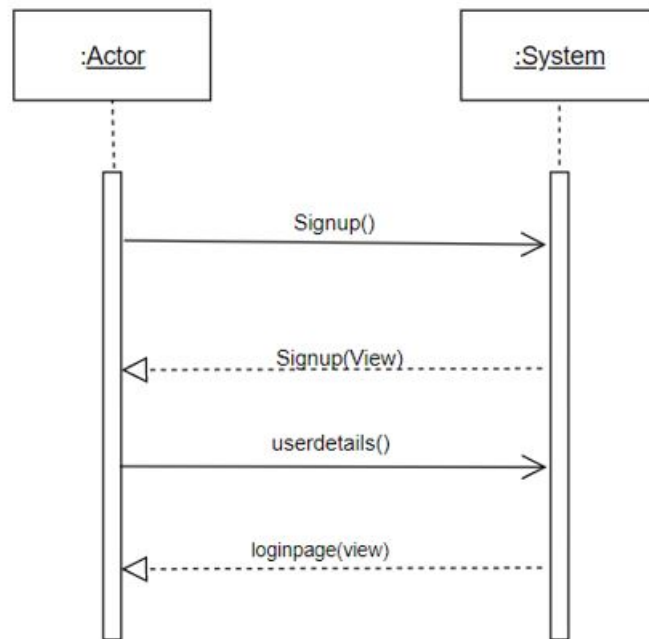
Communication Diagram for admin functionality



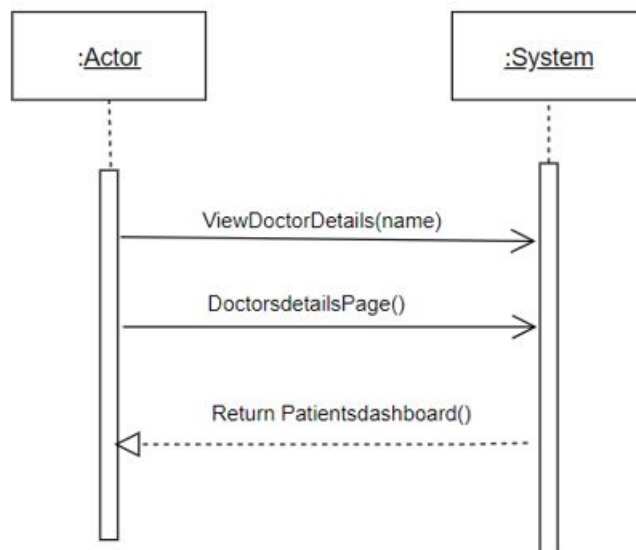
System Sequence Diagrams for User Login



System Sequence Diagrams for Appointment Booking



System Sequence Diagrams for Sign Up



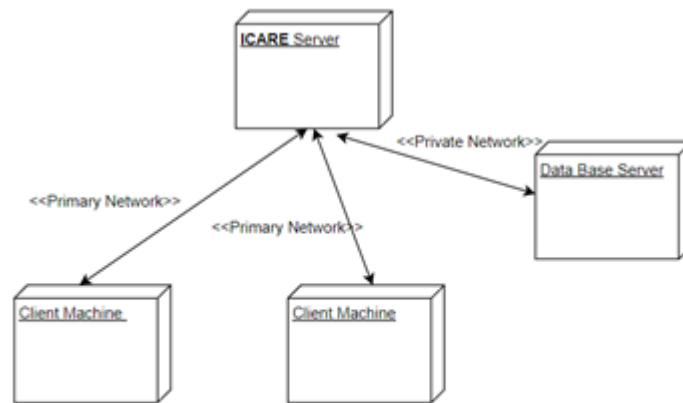
System Sequence Diagrams for View Doctors Details

Physical or Deployment View:

The physical view depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer, as well as the physical connections between these components.

Stakeholders: Deployment managers and System Engineers.

Representation: Deployment diagram.



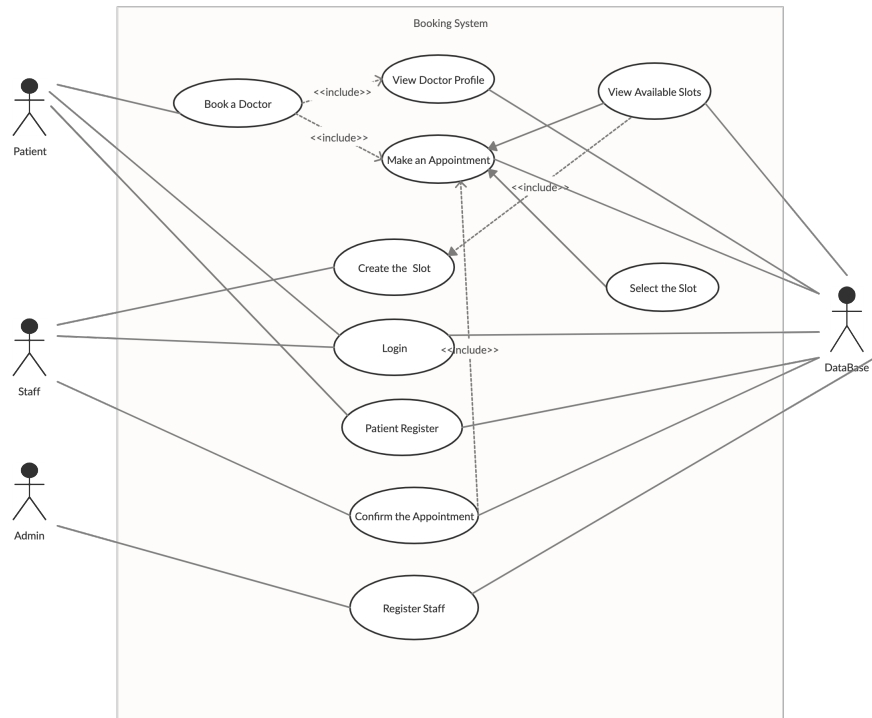
Deployment diagram

Use Case View or Scenario View:

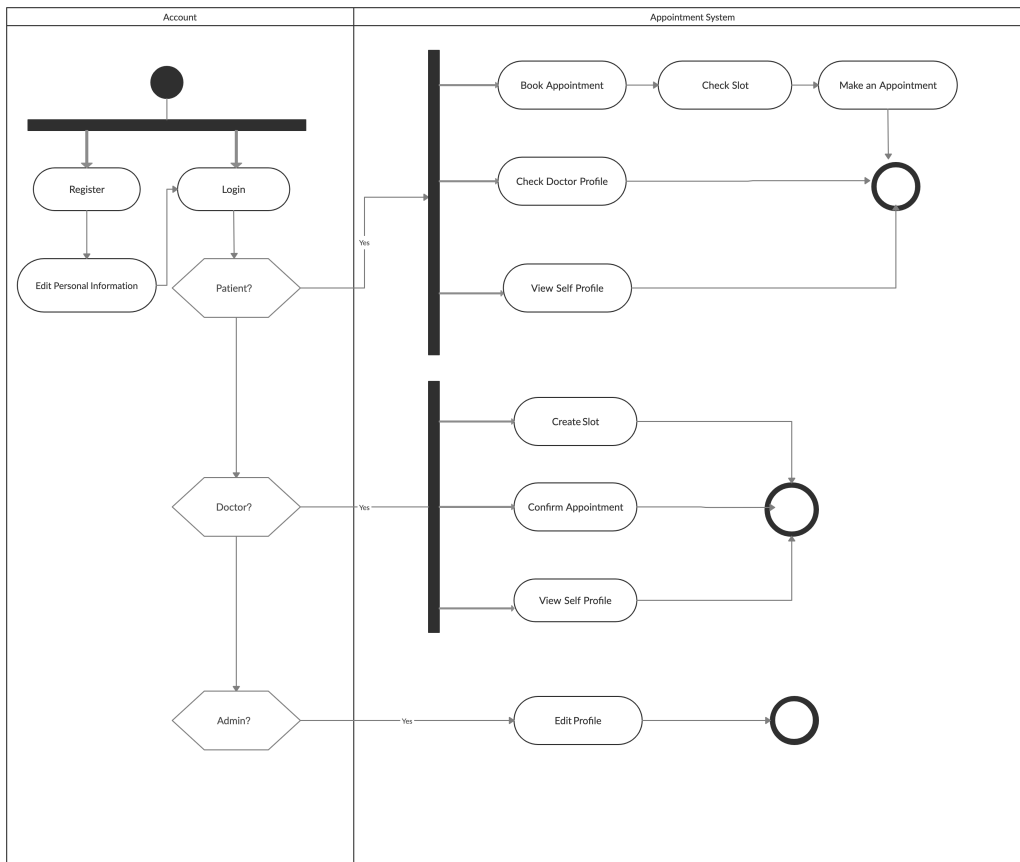
The description of an architecture is illustrated using a small set of use cases, or scenarios which become a fifth view. The scenarios describe sequences of interactions between objects, and between processes. They are used to identify architectural elements and to illustrate and validate the architecture design. They also serve as a starting point for tests of an architecture prototype.

Stakeholders: All the stakeholders of the system, including the end-users.

Representation: Use-Case diagram, Activity diagram.



Use Case Diagram



Activity Diagram

PART 5: ARCHITECTURAL DECISIONS

Architectural Decision 1: Use the MVC architecture

The PHP development model mixes the code of data access, the processing of business logic, and web presentation layer together, as a result, it brought about many problems in the web applications[cite]. In ICARE project, there are three types of users, namely patient, staff and administrator, and different users play different roles in the system. So it is essential to divide these users into different models.

1. Decision:

We will organize ICARE with the MVC pattern.

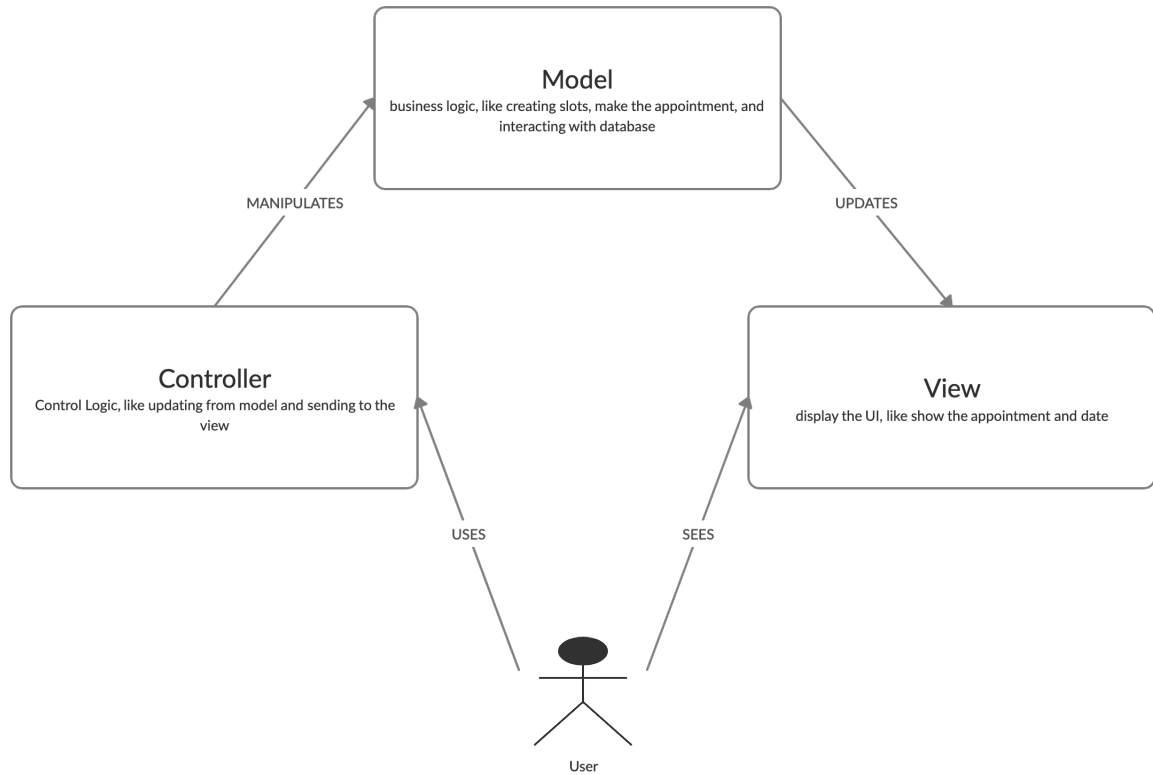
(a) Rationale:

The knowledge acquired in industrial and academic experiences [02] [03] shows that using patterns (specifically, architectural patterns) in systems development allows (1) discerning the domain where the system is being developed, (2) satisfying systemic properties (“quality attributes” or QAs [04]), and (3) creating large-scale reuse design techniques (frameworks [05]). In ICARE, initially, the user interact with the login in page and then the system show corresponding page according to the role of the user. So it is clear that the front page is a controller and view will get the data from model to show in the website. The MVC pattern is abstracted into three parts as model, view, and controller. Model interact with database systems (or simply data storage systems) and responds to request of data (usually from view). Controller responds to events (click of mouse, keyboard stroke etc.) and informs model and/or view to make changes accordingly [06]. The view is the application interface shown or seen by the clients and interacts with the clients. A view will be loaded in the browser by the controller according to the client’s request [07].

2. Status:

Accepted

3. Consequences:



MVC Pattern for ICARE

ICARE is to follow the MVC pattern. The front controller will response to the users' click and filling conduct when login/ signing up. The controller calls the model to complete the read-write operation including viewing profile and make the appointment. The View will show the data in the web browser.

Architectural Decision 2: Architectural tactics

Software architectural tactics are a group of design decisions that influence quality attributes to transform a software design's behaviour. In spite of architectural patterns, which include trade-offs, tactic focus on specific quality attributes, and it does not consider trade-offs (should be considered by the designer).

1. **Decision:** Regarding using the MVC pattern, we choose the following:

(a) **Security tactics:**

- i. **Authentication users (Resisting Attacks) tactic:** It ensures a user or remote computer is actually who it purports to be. Passwords, one-time passwords, digital certificates, and bio-metric identifications provide authentication.
- ii. **Authorize Users (Resisting Attacks) tactic:** By allowing just the users who are allowed to modify or access specific data.

- iii. **Intrusion detection (Detecting Attacks) tactic:** Detecting an ongoing attack is vital to be ready to react and reduce its consequences.

(b) **Safety Tactics:**

- i. **Simplicity (Failure Avoidance) tactic:** By making the software simple we tried to reduce faults.
- ii. **Timeout (Failure Detection) tactic:** By assuming there would be failures in any software system, the timeout tactic would help to detect a possible failure.

(c) **Modifiability Tactics:**

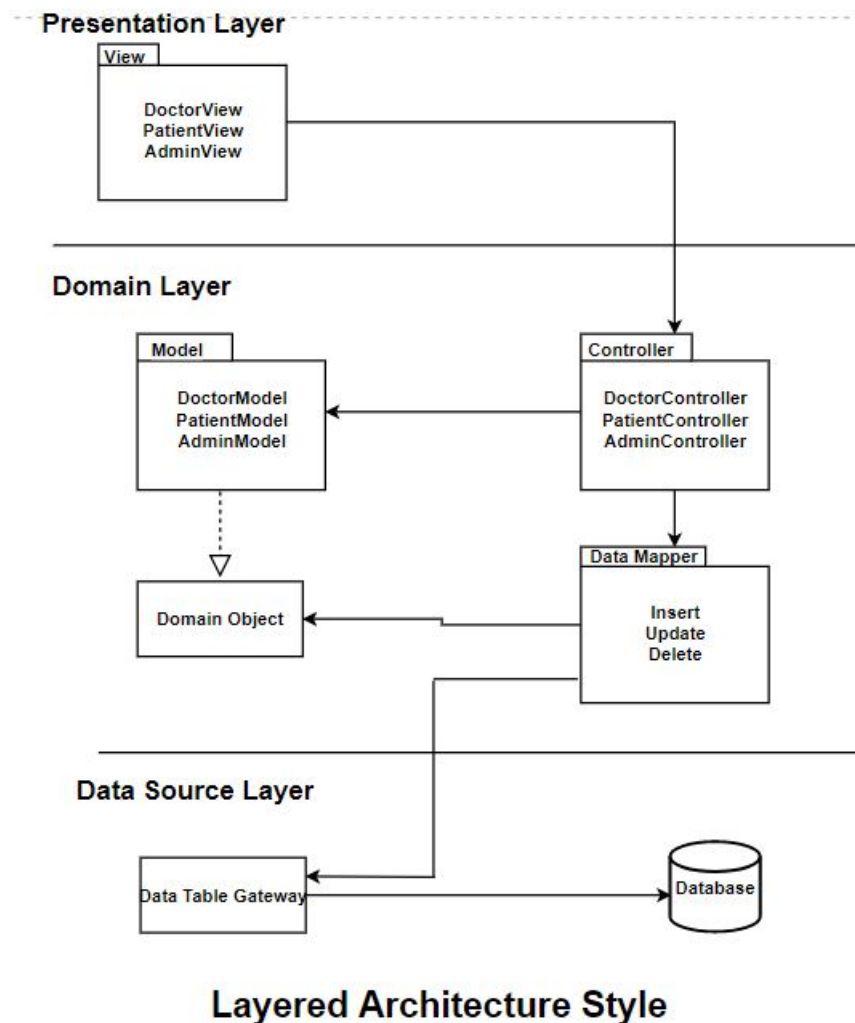
- i. **Increase semantic cohesion tactic:** Separates user interface responsibilities from the co-functionality of the system, so all the views can reside in common components.
- ii. **Using intermediary tactic:** Controllers are used as intermediaries between view and the data to break dependencies between view and the model.
- iii. **Run-time binding:** By deferring bindings time to run-time, they can be changed when the program is running.

Architectural Decision 3: Principles

MVC Architecture or Model View Controller Architecture basic principle is to divide the system into logic of the system and presentation of the system to make the system more maintainable. Hence the system is divided into 3 parts:

1. **View:** The presentation part of the system is in the View. The user will just interact with the view and will not be able to interact with any other part of the system. In iCare, the patient will be able to access this section when they use the login module, appointment booking, viewing doctor profile, viewing reports and such others. Similarly, the staff or doctor can view patient's profile, view booking schedule, reschedule appointments, and so on. Hence, basically, any interaction including the user input, viewing the system by the user is all processed through the view of the system and it also responsible for giving an appropriate response based on the input.
2. **Model:** It possess the logic of the system. The input obtained from view is controlled by the controller, processed, and passed to the model which then update the state of the system accordingly. The state of the system is managed by model and every time the state of the system is manipulated with the change in the input. For example, when the user clicks on the 'Sign-in' button to login, the data from the View are managed by the model and the state is updated by model.
3. **Controller:** It controls the above Model and View. The controller manages the data between the model and view. The data obtained from the user by the view is processed by the controller and passed on to the model to update the state of the system. Once, the model changes the state, the controller process that and an appropriate view is called by the controlled and view show that as a response to the user input.

Architectural Decision 4: Styles



Layered architectural style, otherwise known as n-level engineering style, it is one of the most widely recognized styles utilized in the software development life cycle. Icare application is following layered architecture. A presentation layer would be responsible for managing all user interface and browser communication logic, while a domain layer would be responsible for enforcing specific business rules pertaining to a request. The presentation layer does not need to know or worry about how to get doctors, patient data; it only needs to display that information on a screen format. Similarly, the domain layer does not need to be concerned about how to format doctors, patient data for display on a screen or even where the doctors, patient data is coming from, it only needs to get the data from the data source layer, perform business logic against the data and pass that information up to the presentation layer. In the event that a patient's needs to book an appointment, he will initially need to interact with the presentation layer(Book appointment form), which is the top-most layer, the main function of this layer is to translate tasks and results to something the patient can understand. And afterward the patient request is handled by middle layer which is called domain layer(bookingController). It is used to coordinate to application layer and makes logical decision by patient request. Domain layer

will validate the data received from booking form and then it goes to data source layer, which has a control to the actual database, to perform the database writing.

PART 6: IMPLEMENTATION

Tools and Technologies:

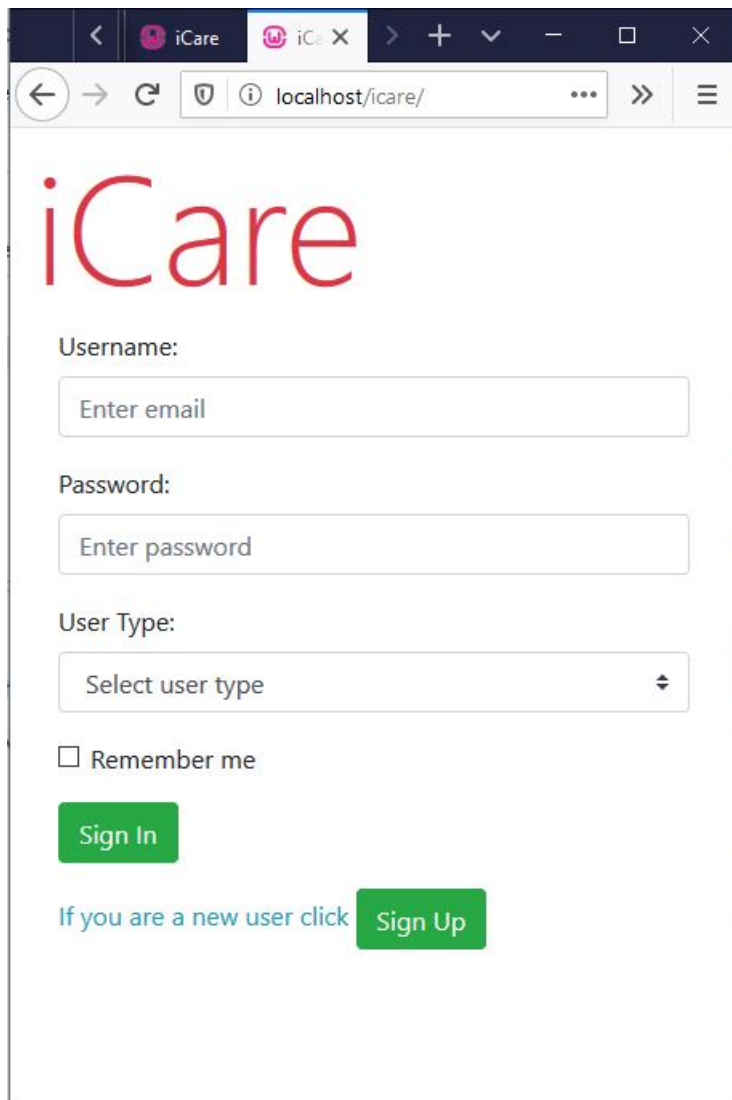
1. Tools:

- (a) Sublime Text Editor
- (b) Wamp Server(LocalHost)
- (c) Google Chrome
- (d) draw.io
- (e) creately.com
- (f) XMind

2. Technologies:

- (a) php
- (b) Bootstrap 4
- (c) MySQL
- (d) HTML
- (e) CSS
- (f) Javascript
- (g) Ajax

Homepage:



The screenshot shows a web browser window with two tabs, both labeled 'iCare'. The address bar displays 'localhost/icare/'. The page content includes the 'iCare' logo in red, followed by a login form. The form has three input fields: 'Username:' with a placeholder 'Enter email', 'Password:' with a placeholder 'Enter password', and 'User Type:' with a dropdown menu showing 'Select user type'. Below these fields is a checkbox labeled 'Remember me'. At the bottom of the form are two green buttons: 'Sign In' and 'Sign Up'. The 'Sign Up' button is preceded by the text 'If you are a new user click'.

Username:

Enter email

Password:

Enter password

User Type:

Select user type

☐ Remember me

Sign In

If you are a new user click Sign Up

Patient Sign Up:

iCare

Sign Up

Last Name

Other Name(s)

Email

Password

☒ Male ☐ Female

Date of Birth: Day Month Year

Height (cm)

Weight (lb)

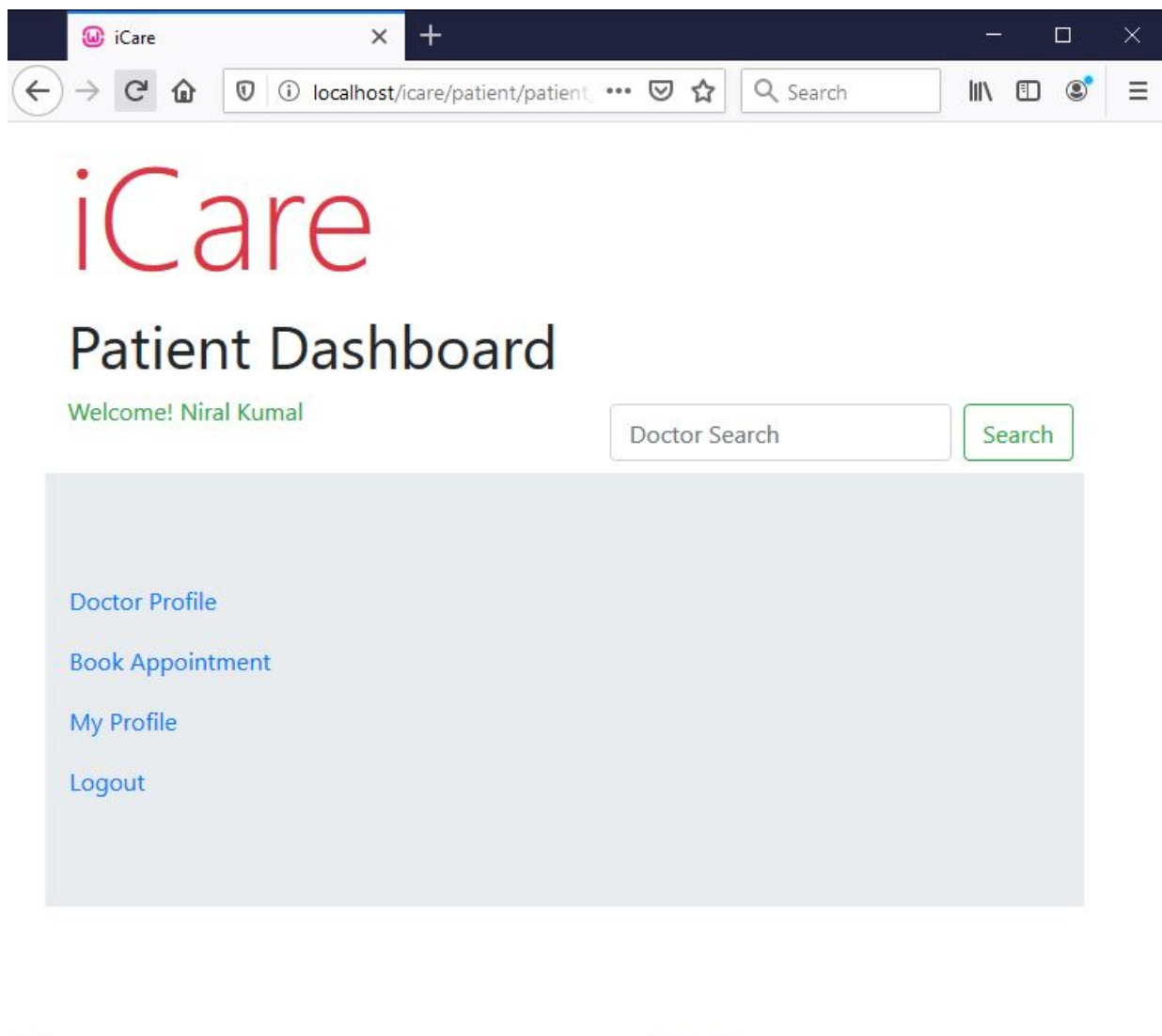
Address

City

State

Zip

Patient Dashboard:



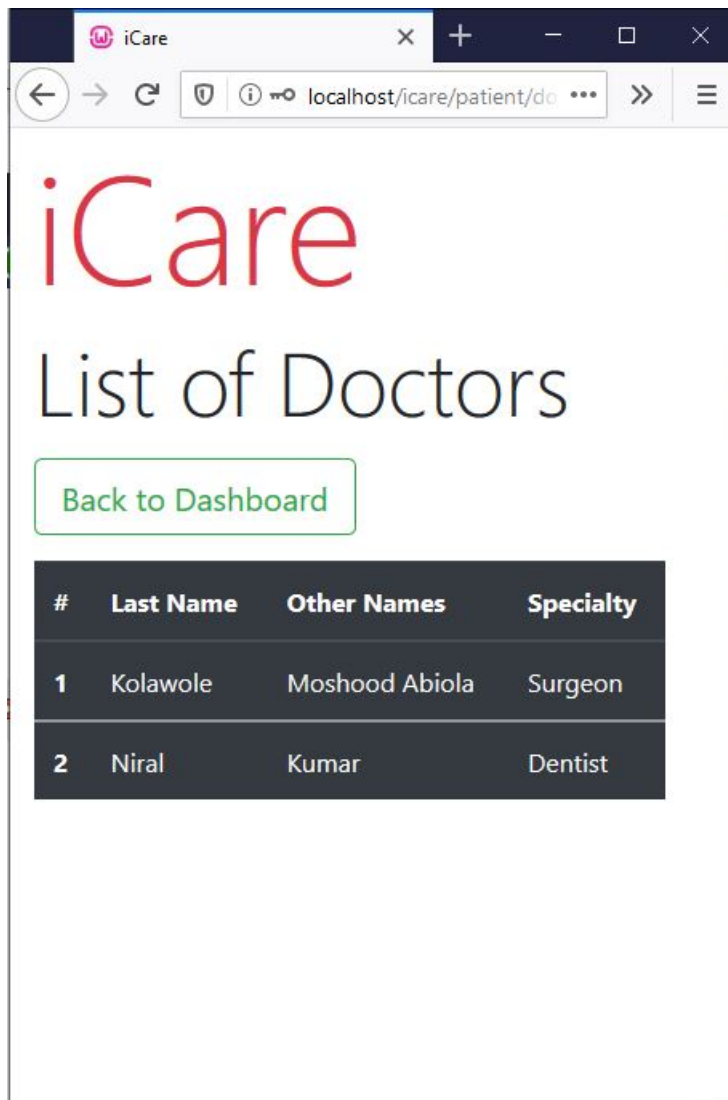
Book Appointment:

The screenshot shows a web browser window with the URL `localhost/icare/patient/bc`. The page features the **iCare** logo in red and the title **Appointment Booking** in black. A green button labeled **Back to Dashboard** is positioned below the title. A dark-themed table displays three appointment entries. The table has columns for **#**, **Book**, **Last Name**, **Other Names**, **Date**, and **Time**. The first two rows show appointments for 'Kolawole Moshood Abiola' on 2020-06-10 and 2020-06-08. The third row shows an appointment for 'Niral Kumar' on 2020-06-15. Below the table, there is a blue link that says **Confirm Appointment**.

#	Book	Last Name	Other Names	Date	Time
1		Kolawole	Moshood Abiola	2020-06-10	10:00
2		Kolawole	Moshood Abiola	2020-06-08	11:00
3	■	Niral	Kumar	2020-06-15	02:00

[Confirm Appointment](#)

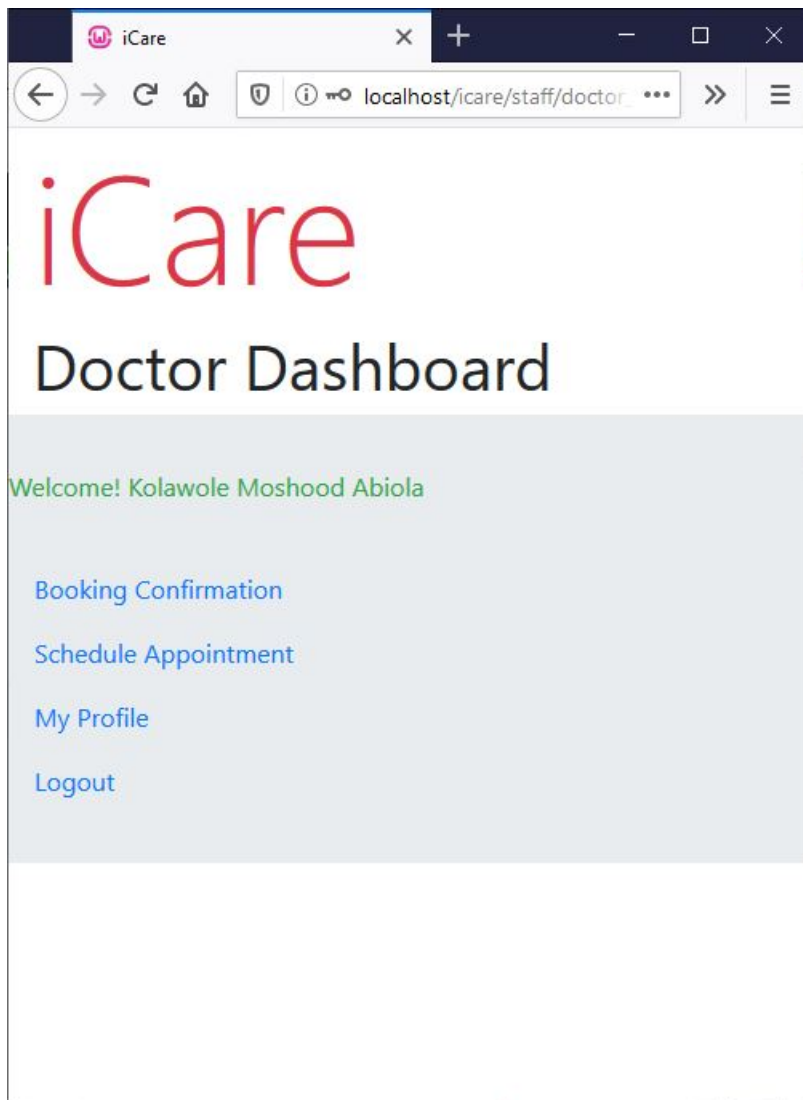
Doctor Profile:



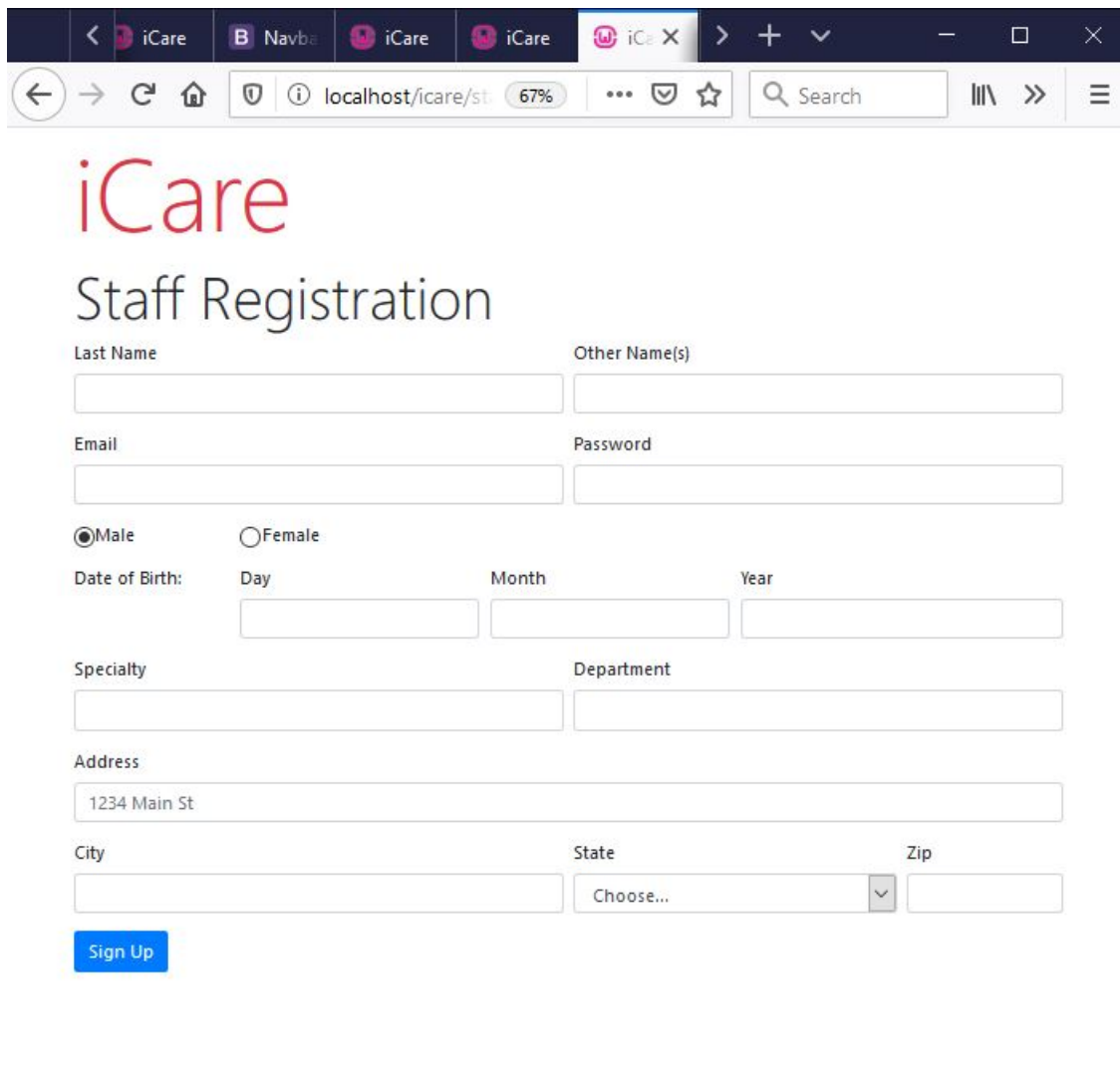
The screenshot shows a web browser window with the iCare logo and the title 'List of Doctors'. Below the title is a green button labeled 'Back to Dashboard'. A table with a dark background and white text lists two doctors. The table has four columns: '#', 'Last Name', 'Other Names', and 'Specialty'.

#	Last Name	Other Names	Specialty
1	Kolawole	Moshood Abiola	Surgeon
2	Niral	Kumar	Dentist

Doctor Dashboard:



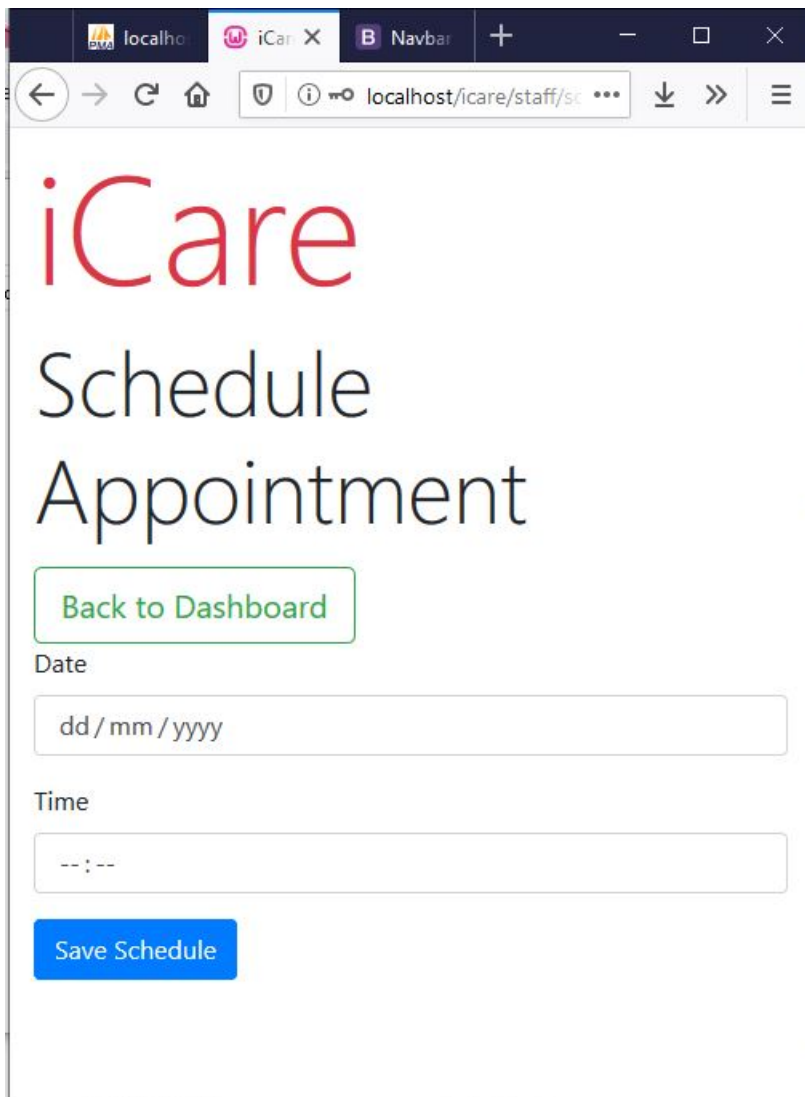
Staff Registration:



The screenshot shows a web browser window with the iCare logo and the title "Staff Registration". The browser's address bar shows "localhost/icare/st" and the page is loaded at 67%. The form contains the following fields and controls:

- Last Name**: Text input field.
- Other Name(s)**: Text input field.
- Email**: Text input field.
- Password**: Text input field.
- Gender**: Radio buttons for ☒ Male and ☐ Female.
- Date of Birth**: Three text input fields for **Day**, **Month**, and **Year**.
- Specialty**: Text input field.
- Department**: Text input field.
- Address**: Text input field with the placeholder "1234 Main St".
- City**: Text input field.
- State**: A dropdown menu with "Choose..." as the selected option.
- Zip**: Text input field.
- Sign Up**: A blue button at the bottom left of the form.

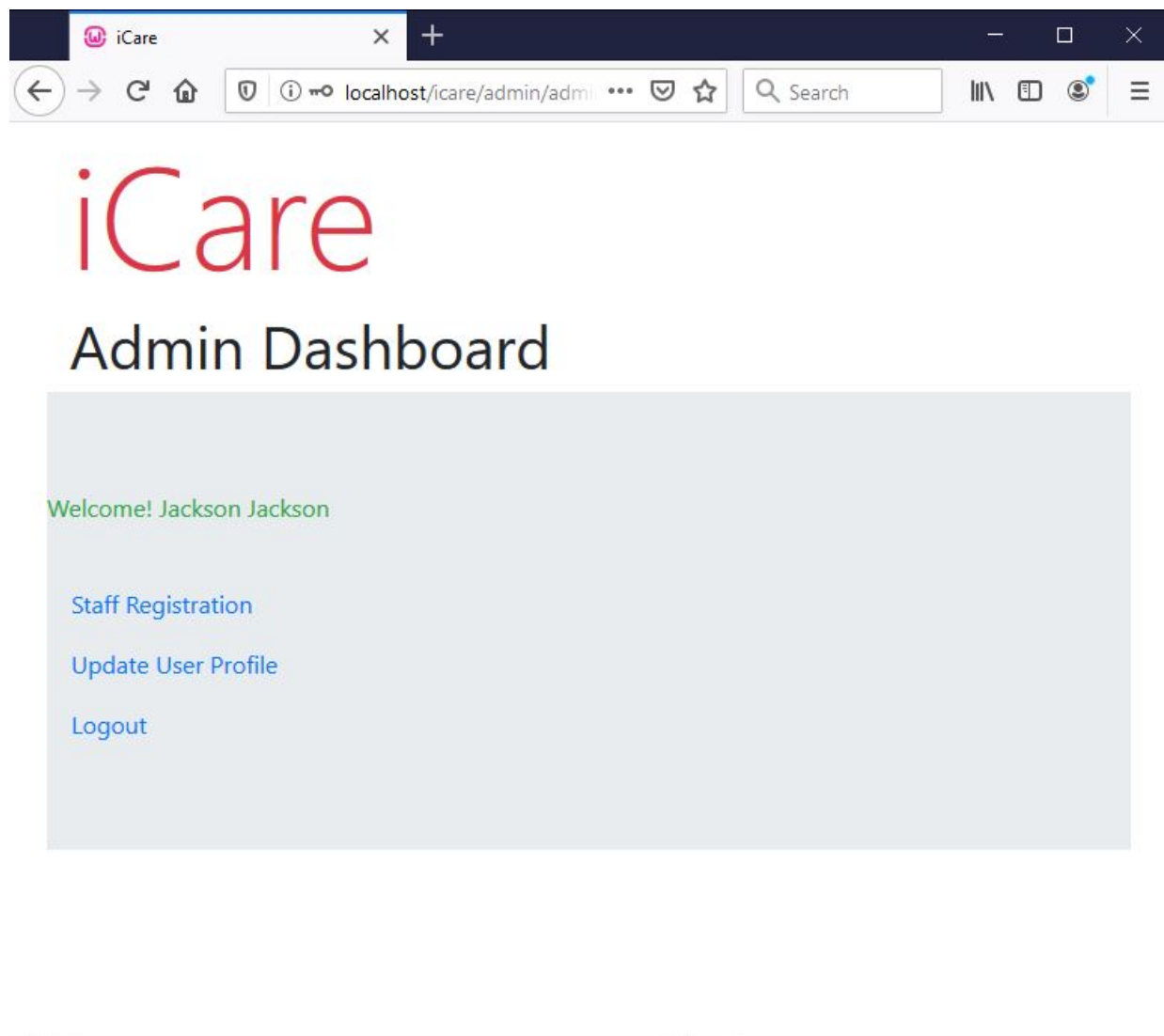
Schedule Appointment:



The screenshot shows a web browser window with the URL `localhost/icare/staff/sc`. The page has a dark blue header with the text "iCare X" and "Navbar". The main content area is white and contains the following elements:

- The "iCare" logo in red.
- The title "Schedule Appointment" in a large, dark grey font.
- A green button labeled "Back to Dashboard".
- A "Date" label above a text input field containing the placeholder "dd/mm/yyyy".
- A "Time" label above a text input field containing the placeholder "--:--".
- A blue button labeled "Save Schedule".

Admin Dashboard:



REFERENCES

1. Concordia University Logo: https://www.google.com/search?q=concordia+university+logo&sxsrf=ALeKk00nNpzxCfTcJxEwAQNk-ilil0DPkg:1591909168136&tbm=isch&source=iu&ictx=1&fir=Ccxmok0d1jX1RM%253A%252C01bcPIzeDtiMbM%252C_&vet=1&usg=AI4_-kS2jmerHVsa=X&ved=2ahUKEwiey-zj0_rpAhVNRjABHXHQDfAQ9QEwAHoECAsQIw&biw=1366&bih=625#imgsrc=fzpIxbXYP_q83M
2. M. Fowler, Patterns of Enterprise Application Architecture. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.
3. M. Kircher and J. Prashant, Pattern-Oriented Software Architecture. Patterns for Resource Management. West Sussex, England: Wiley series in software design patterns, 2004.
4. L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, Non-functional requirements in software engineering, vol. 5. Springer Science and Business Media., 2012.
5. R. E. Johnson, "Frameworks = (components + patterns)," Commun. ACM, vol. 40, no. 10, pp. 39–42, 1997.
6. Aditya Singh, Piyush Chawla, Karan Singh, Ashutosh Kumar Singh, Formulating an MVC Framework for Web Development in JAVA <https://ieeexplore-ieee-org.lib-ezproxy.concordia.ca/stamp/stamp.jsp?tp=&arnumber=8553746>
7. Stenly Ibrahim Adam, Stevani Andolo, A New PHP Web Application Development Framework Based on MVC Architectural Pattern and Ajax Technology <https://ieeexplore-ieee-org.lib-ezproxy.concordia.ca/stamp/stamp.jsp?tp=&arnumber=8874912>
8. https://www.student.cs.uwaterloo.ca/~cs446/1171/Arch_Design_Activity/Layered.pdf
9. <https://users.encs.concordia.ca/~cc/soen6461/>
10. <https://creately.com/diagram/example/indonknf1/New%20Patient%20Appointment%20System>
11. <https://www.htmlgoodies.com/beyond/php/article.php/3912211/Principles-Of-MVC-for-PHtm#:~:text=MVC%2C%20or%20Model-View-,be%20separated%20from%20its%20presentation.>
12. Bass, L., Clemens, P., Kazman, R.: Software Architecture in Practice. SEI Series in Software Engineering, Addison-Wesley, 2 edn. (2003)

13. Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., Sommerlad, P.: Security Patterns: Integrating Security and Systems Engineering. John Wiley and Sons (2006)
14. Wu, W., Kelly, T.: Safety Tactics for Software Architecture Design. In: Proc. 28th Annual International Computer Software and Applications Conference COMPSAC 2004. pp. 368–375 (2004)