

AN AUTOMATED STRABISMUS  
CLASSIFICATION USING CASE-BASED  
REASONING ALGORITHM FOR BINOCULAR  
VISION MANAGEMENT SYSTEM

MUHAMMAD AMIRUL ISYRAF BIN  
ROHISMADI

BACHELOR OF COMPUTER SCIENCE  
(SOFTWARE ENGINEERING)

UNIVERSITI MALAYSIA PAHANG

## UNIVERSITI MALAYSIA PAHANG

### DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : Muhammad Amirul Isyraf bin Rohismadi

Date of Birth

Title : An Automated Strabismus Classification using Case-Based Reasoning Algorithm for Binocular Vision Management System

Academic Session : Semester 1 2022/2023

I declare that this thesis is classified as:

- CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)\*
- RESTRICTED (Contains restricted information as specified by the organization where research was done)\*
- OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

\_\_\_\_\_  
(Student's Signature)

\_\_\_\_\_  
(Supervisor's Signature)

New IC/Passport Number  
Date: 29/01/2023

DR. ANIS FARIHAN BINTI MAT RAFFEI  
SENIOR LECTURER  
FACULTY OF COMPUTING  
COLLEGE OF COMPUTING & APPLIED SCIENCES  
UNIVERSITI MALAYSIA PAHANG  
26600 PEKAN, PAHANG DARUL MAKMUR  
TEL : 09-424 4634 FAX : 09-424 4666

NOTE : \* If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.





## SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Bachelor of Computer Science (Software Engineering)

A handwritten signature in blue ink, appearing to read 'Anis Farihan Binti Mat Raffei', is written above a horizontal line.

(Supervisor's Signature)

Full Name : DR. ANIS FARIHAN BINTI MAT RAFFEI

Position : PROJECT SUPERVISOR

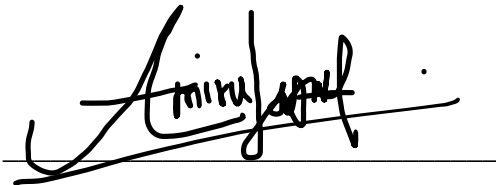
Date : 29 JANUARY 2023



## STUDENT'S DECLARATION



I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

A handwritten signature in black ink, appearing to read 'Amirul Isyraf', is written over a horizontal line.

(Student's Signature)

Full Name : MUHAMMAD AMIRUL ISYRAF BIN ROHISMADI

ID Number : CB20014

Date : 29 JANUARY 2023

AN AUTOMATED STRABISMUS CLASSIFICATION USING CASE-BASED  
REASONING ALGORITHM FOR BINOCULAR VISION MANAGEMENT  
SYSTEM

MUHAMMAD AMIRUL ISYRAF BIN ROHISMADI

Thesis submitted in fulfillment of the requirements  
for the award of the degree of  
Bachelor of Computer Science (Software Engineering)

Faculty of Computing  
UNIVERSITI MALAYSIA PAHANG

FEBRUARY 2023

## **ACKNOWLEDGEMENTS**

Alhamdulillah, praise be to Allah the Almighty of God the most Gracious and the most Merciful, first of all the writer would like to thank God for His Blessing in the completion of this thesis. The writer also deeply thankful to everyone who was involved in this project, as without their support and guidance, it would not have been possible to complete. Special thanks to the supervisor, Dr. Anis Farihan binti Mat Raffei, from the Faculty of Computing, University Malaysia Pahang, for providing valuable wisdom and guidance throughout the entire project. Not to forget, huge gratitude also to the lecturers, parents, family members, and friends for their unwavering support and encouragement from time to time in completing this project.

## **ABSTRACT**

Binocular vision is a type of vision that allows an individual to perceive depth and distance using both eyes to create a single image of their environment. However, there is an illness called strabismus, where it is difficult for some people to focus on seeing things clearly at a time. There are a lot of diagnosis need to be done for doctors to diagnose whether patients suffer from strabismus or not. One of them is to perform accommodate amplitude test, which is time-consuming. Thus, with the Agile methodology, the Binocular Vision Management system is proposed which comprised of two components, a web-based component for patient, treatment, and appointment management, and a machine learning component for automating the strabismus classification by using case-based reasoning algorithm. Therefore, this will significantly hasten the process of classifying strabismus and help keep all clinical records in one place.

## **ABSTRAK**

Penglihatan binokular ialah sejenis penglihatan yang membolehkan seseorang individu melihat kedalaman dan jarak menggunakan kedua-dua mata untuk mencipta satu imej persekitaran mereka. Walau bagaimanapun, terdapat penyakit yang dipanggil strabismus, di mana sukar bagi sesetengah orang untuk memberi tumpuan kepada melihat sesuatu dengan jelas pada satu masa. Terdapat banyak diagnosis yang perlu dilakukan untuk doktor mendiagnosis sama ada pesakit mengalami strabismus atau tidak. Salah satunya adalah untuk melakukan ujian “Accommodate Amplitude”, yang memakan masa. Oleh itu, dengan metodologi Agile, sistem “Binocular Vision Management” telah dicadangkan yang terdiri daripada dua komponen, iaitu komponen berasaskan web untuk pengurusan pesakit, rawatan dan temu janji, serta komponen pembelajaran mesin untuk mengautomasikan klasifikasi strabismus dengan menggunakan algoritma “Case-Based Reasoning”. Oleh itu, ini akan mempercepatkan proses pengelasan strabismus dengan ketara dan membantu menyimpan semua rekod klinikal di satu tempat.

## TABLE OF CONTENT

<b>DECLARATION</b>	
<b>TITLE PAGE</b>	
<b>ACKNOWLEDGEMENTS</b>	<b>ii</b>
<b>ABSTRACT</b>	<b>iii</b>
<b>ABSTRAK</b>	<b>iv</b>
<b>TABLE OF CONTENT</b>	<b>v</b>
<b>LIST OF TABLES</b>	<b>ix</b>
<b>LIST OF FIGURES</b>	<b>x</b>
<b>LIST OF SYMBOLS</b>	<b>xiii</b>
<b>LIST OF ABBREVIATIONS</b>	<b>xiv</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 INTRODUCTION	1
1.2 PROBLEM STATEMENTS	3
1.3 OBJECTIVES	4
1.4 SCOPE OF PROJECT	4
1.5 SIGNIFICANCE OF PROJECT	4
1.6 THESIS ORGANIZATION	5
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>6</b>
2.1 INTRODUCTION	6
2.2 REVIEW OF EXISTING SYSTEMS	6
2.2.1 Odoo Eye Clinic Management	6

2.2.2	Optic Clinic	8
2.2.3	Smart Eye Care	9
2.3	COMPARISON BETWEEN EXISTING SYSTEMS AND PROPOSED SYSTEM	12
2.4	CURRENT ALGORITHMS FOR CLASSIFICATION	15
2.4.1	Fuzzy Logic	15
2.4.2	Case-Based Reasoning	15
2.4.3	Artificial Neural Network	16
2.5	COMPARISON BETWEEN ALGORITHMS	17
2.6	CHAPTER SUMMARY	18
<b>CHAPTER 3 METHODOLOGY</b>		<b>19</b>
3.1	INTRODUCTION	19
3.2	METHODOLOGY	19
3.3	PROJECT REQUIREMENT	23
3.3.1	Functional Requirements	23
3.3.2	Non-Functional Requirement	26
3.3.3	Constraints	27
3.3.4	Limitations	27
3.4	STRABISMUS DIAGNOSIS	28
3.4.1	Accommodative Amplitude (AA)	28
3.4.2	Lag of Accommodation (LA)	29
3.5	CASE-BASED REASONING ALGORITHM JUSTIFICATION	30
3.6	PROPOSED DESIGN	31
3.6.1	Context Diagram	31
3.6.2	Use Case Diagram	32

3.6.3	Activity Diagram	46
3.7	DATA DESIGN	51
3.7.1	Entity Relationship Diagram	51
3.7.2	Data Dictionary	52
3.8	CASE-BASED REASONING PROCESSES	56
3.9	DATASETS	58
3.9.1	Dataset Collection	58
3.9.2	Data Pre-Processing	58
3.10	HARDWARE AND SOFTWARE SPECIFICATIONS	59
3.11	DESIGN PROTOTYPE	60
3.12	TESTING METHOD	66
3.12.1	Functional Testing	66
3.12.2	User Acceptance Test Plan	66
3.13	POTENTIAL USE OF PROPOSED SOLUTION	69
3.14	GANTT CHART	70
	<b>CHAPTER 4 RESULT AND DISCUSSION</b>	<b>71</b>
4.1	INTRODUCTION	71
4.2	DEVELOPMENT ENVIRONMENT	71
4.3	BINOCULAR VISION MANAGEMENT SYSTEM IMPLEMENTATION	72
4.4	CASE-BASED REASONING IMPLEMENTATION	82
4.4.1	Introduction	82
4.4.2	Algorithm Processes	83
4.5	SYSTEM OUTPUT	87
4.6	CASED-BASED REASONING PERFORMANCE EVALUATION	103
4.7	USER ACCEPTANCE TEST	107



<b>CHAPTER 5 CONCLUSION</b>	<b>110</b>
5.1 INTRODUCTION	110
5.2 OBJECTIVE REVISITED	110
5.2.1 Limitations	111
5.2.2 Future Works	112
<b>REFERENCES</b>	<b>113</b>
<b>APPENDIX A USER ACCEPTANCE TEST RESPONSES</b>	<b>114</b>
<b>APPENDIX B SOFTWARE REQUIREMENT SPECIFICATION FOR BINOCULAR VISION MANAGEMENT SYSTEM</b>	<b>125</b>
<b>APPENDIX C SOFTWARE DESIGN DESCRIPTION FOR BINOCULAR VISION MANAGEMENT SYSTEM</b>	<b>126</b>

## LIST OF TABLES

Table 2.1	Comparison between existing systems and proposed system	12
Table 2.2	Comparison between machine learning classification algorithms	17
Table 3.1	Non-functional requirements	26
Table 3.2	System constraints and its descriptions	27
Table 3.3	System limitations and its descriptions	27
Table 3.4	Data dictionary of table doctors	52
Table 3.5	Data dictionary of table users	52
Table 3.6	Data dictionary of table patients	52
Table 3.7	Data dictionary of table treatments	53
Table 3.8	Data dictionary of table treatment_doctors	53
Table 3.9	Data dictionary of table accommodative_amplitudes	54
Table 3.10	Data dictionary of table lag_accommodations	54
Table 3.11	Data dictionary of table appointment_events	55
Table 3.12	Data dictionary of table appointment_categories	55
Table 3.13	Hardware specifications	59
Table 3.14	Software specifications	59
Table 3.15	Test cases of the proposed system	66

## LIST OF FIGURES

Figure 1.1	Example of human eyes. (a) Normal eyes condition (b) Strabismus disease	2
Figure 2.1	Patient information interface of Odoo Eye Clinic Management	7
Figure 2.2	Appointment record interface of Odoo Eye Clinic Management	7
Figure 2.3	Manage patient interface of Optic Clinic	8
Figure 2.4	Booking interface of Optic Clinic	9
Figure 2.5	Patient details interface of Smart Eye Care	9
Figure 2.6	Add new patient interface of Smart Eye Care	10
Figure 2.7	Appointment module of Smart Eye Care	11
Figure 2.8	Basic representation of a neural network application	16
Figure 3.1	Agile methodology and its phases	20
Figure 3.2	Accommodative Amplitude diagnosis	28
Figure 3.3	Lag of Accommodation diagnosis	29
Figure 3.4	Context Diagram	31
Figure 3.5	Use Case Diagram	32
Figure 3.6	Use Case Description Diagram for Manage Login	33
Figure 3.7	Use Case Description Diagram for Manage Doctor	35
Figure 3.8	Use Case Description Diagram for Manage Patient	38
Figure 3.9	Use Case Description Diagram for Manage Treatment	41
Figure 3.10	Use Case Description Diagram for Manage Treatment	44
Figure 3.11	Activity Diagram for Manage Login	46
Figure 3.12	Activity Diagram for Manage Doctor	47
Figure 3.13	Activity Diagram for Manage Patient	48
Figure 3.14	Activity Diagram for Manage Treatment	49
Figure 3.15	Activity Diagram for Manage Appointment	50
Figure 3.16	Entity Relationship Diagram of the proposed system	51
Figure 3.17	Case-Based Reasoning processes	56
Figure 3.18	Dataset collection	58
Figure 3.19	Accommodate Amplitude dataset.	58
Figure 3.18	Gantt chart for system development	70
Figure 4.1	Development Environments	71
Figure 4.2	Project's structure in Visual Studio code	72
Figure 4.3	Doctor's table resource snippet code	74

Figure 4.4	Doctor's form resource snippet code	76
Figure 4.5	Patient's table resource snippet code	77
Figure 4.6	Patient's form resource snippet code	80
Figure 4.7	AA Classification function snippet code.	81
Figure 4.8	Jupyter Notebook IDE	82
Figure 4.9	Dataset reading code snippet.	83
Figure 4.10	Finding range value of features code snippet.	83
Figure 4.11	Dataset features sample.	84
Figure 4.12	AA measurement arguments code snippet.	84
Figure 4.13	Local similarity, global similarity calculations code snippet.	85
Figure 4.14	Print highest similarity result code snippet.	85
Figure 4.15	CBR retain process code snippet.	86
Figure 4.16	Login screen	87
Figure 4.17	Wrong username or password error message	87
Figure 4.18	Dashboard screen	88
Figure 4.19	Dashboard in dark mode theme	88
Figure 4.20	Doctor's list screen	89
Figure 4.21	Create doctor profile screen	89
Figure 4.22	View doctor screen	90
Figure 4.23	Edit doctor screen	91
Figure 4.24	Delete doctor screen	91
Figure 4.25	Search doctor screen	92
Figure 4.26	Export doctor record screen	92
Figure 4.27	Patient list screen	93
Figure 4.28	Create patient profile (patient detail) screen	93
Figure 4.29	Create patient profile (symptom) screen	94
Figure 4.30	View patient screen	94
Figure 4.31	Edit patient screen	95
Figure 4.32	Delete patient screen	96
Figure 4.33	Treatments list screen	96
Figure 4.34	Add treatment (patient) screen	97
Figure 4.35	Add treatment (accommodate amplitude treatment) screen	97
Figure 4.36	Accommodate amplitude treatment result screen	98
Figure 4.37	Add treatment (lag of accommodation treatment) screen	98
Figure 4.38	Lag of accommodation treatment result screen	99

Figure 4.39	Add treatment (person in charge) screen	99
Figure 4.40	Add treatment (remark) screen	100
Figure 4.41	Appointment calendar screen	100
Figure 4.42	View event details screen	101
Figure 4.43	View event details screen	102
Figure 4.44	Event list screen	102
Figure 4.45	Edit event screen	103
Figure 4.46	Model performance data statistics	103
Figure 4.47	Model performance test statistics	104
Figure 4.48	Model accuray overtime	105
Figure 4.49	Model performance measures	105
Figure 4.50	Model confusion matrix	106
Figure 4.51	Live meeting of user acceptance test	107
Figure 4.52	User acceptance test form introduction	107
Figure 4.53	Respondent personal details	108
Figure 4.54	Declaration agreement response	108
Figure 4.55	Respondent signatures and stamps	109

## LIST OF SYMBOLS

## LIST OF ABBREVIATIONS

AA	Accommodative Amplitude
AI	Artificial Intelligence
BAF	Binocular Accommodative Facility
CPU	Central Processing Unit
CRUD	Create, Read, Update, Delete
IIUM	International Islamic University Malaysia
LA	Lag of Accommodation
MAF	Monocular Accommodative Facility
NRA	Negative Relative Accommodation
PRA	Positive Relative Accommodation
RAF	Royal Air Force
SDD	Software Design Document
SRS	Software Requirement Specification
UAT	User Acceptance Test

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 INTRODUCTION**

Every human being is normally born with a pair of eyes that enable them to see the world. Binocular vision is a type of vision in which a human use both of their eyes to perceive a single three-dimensional view of their environment. Hence, they can keep both eyes focused on an object and create a single visual image concurrently (Stidwell & Fletcher, 2017). There are many advantages of binocular vision. One of them is human will have better depth perception, making it is easier to judge speed and velocity accurately. For example, human can estimate how quicky a car is approaching while crossing roads. Moreover, it also enables people to walk faster and more confidently over and around obstacles.

However, despite its advantages, some people suffer from eye illness that makes it difficult for them to focus on seeing things clearly at a time. The condition is called a squint or strabismus, where the eyes do not align properly and point in different direction (Repka, Lum, & Burugapalli, 2018). Figure 1.1 shows the comparison of a normal human eye condition and a strabismus disease. Also, strabismus is the most common cause of amblyopia, where both eyes does not develop properly during childhood. If the patient's is untreated, he or she will experience vision loss that is permanent and irreversible (Chen, Fu, Lo, & Chi, 2018). Thus, an automated strabismus recognition is essential, and early screening is necessary to prevent strabismus from worsening.



(a)

(b)

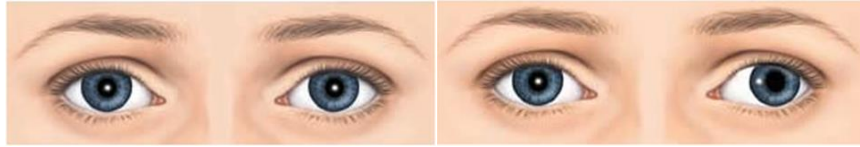


Figure 1.1 Example of human eyes. (a) Normal eyes condition (b) Strabismus disease

Strabismus can be diagnosed through several methods. One of these methods is evaluating the **accommodative amplitude (AA)**, which is the maximum amount that the eyes can focus on an object at different distances. The doctor may also assess the **lag of accommodation (LA)**, which is the delay in the eyes' ability to focus on an object as it moves closer or farther away. In addition, the doctor may also measure the **monocular accommodative facility (MAF)**, which is the speed at which the eyes can focus on an object when only one eye is used. On the other hand, the **binocular accommodative facility (BAF)** measures the speed at which the eyes can focus on an object when both eyes are used together. Next, the doctor may also evaluate the negative and positive relative accommodation of the eyes. **Negative relative accommodation (NRA)** occurs when the eyes are unable to focus on an object that is too close, while **positive relative accommodation (PRA)** occurs when the eyes are unable to focus on an object that is too far away. Overall, these tests help the doctor understand the extent of the misalignment and determine the appropriate treatment for strabismus.

However, these diagnoses are very time-consuming as doctor may need to perform these tests multiple times to get an accurate measurement. Some of the tests, such as the monocular and binocular accommodative facility, may also require the use of specialized equipment, which can further prolong the diagnostic process. Finally, the doctor may need to consider other factors when making a diagnosis, such as the patient's age, overall health, and any other underlying conditions that may be contributing to the misalignment of the eyes.

Therefore, binocular vision management system with the integration of machine learning is proposed to help doctors in diagnosing their patients whether they are suffering from strabismus or not. It would definitely ease the flow, as well as improve

the efficiency of the current strabismus test that requires a significant amount of skilled labour and time to complete.

## **1.2 PROBLEM STATEMENTS**

The traditional way of inspecting patient strabismus is time-consuming as it involves various tests that may need to be performed multiple times in order to get a reliable measurement. Besides, although some types of strabismus are visibly noticeable, there is a risk of human error in diagnosing it. It is also a challenging process since performing the diagnostic requires a high level of skills, and if mistakes are made, the strabismus condition may be misinterpreted.

Additionally, the traditional diagnostic process may not be as accurate as some newer methods that are available, such as computerized systems with artificial intelligence (AI) capabilities. Artificial intelligences are programmes that allow a machine to do a task without the need for human intervention (Shakya, 2020). Nowadays, AI algorithms, especially in the medical profession, has been widely used in computer vision and pattern recognition for many applications including psychological analysis, facial expression recognition, and medical diagnosis (Zhang, Cao, Yang, & Zhao, 2017). Therefore, these systems also may apply to optometry field where they can be used to quickly analyse large amounts of data and identify patterns so that the accuracy of the strabismus classification can be improved.

Other than that, doctors usually record their patient and treatment information in a piece of paper, which are prone to typos and other mistakes. This can lead to incorrect or incomplete information being recorded, which can have serious consequences for the patient's data. Furthermore, paper-based method is often difficult to access and update in a timely manner. For example, if a doctor needs to review a patient's medical history or treatment information, they may need to physically locate and review the relevant paper records. To make matters worse, if the patient file together with the result are missing, the doctors will have to repeat the strabismus test once again. Because of that, it is essential to keep all the patient and appointment details in a computerised system.

### **1.3 OBJECTIVES**

The objectives of this project are:

- To study current limitation of existing eye clinic systems.
- To develop a binocular vision management system that utilizes case-based reasoning algorithms for automating strabismus classification and treatment management.
- To test the functionality of the proposed system.

### **1.4 SCOPE OF PROJECT**

The scope of this project is to develop a web-based system for doctors who specialize in eye treatment. The system will be written in PHP and Python programming language and will use a machine learning algorithm to classify strabismus. The system will be accessible via the internet and will require an internet connection to be used.

Furthermore, it is also designed specifically for IIUM doctors and will be tailored to their needs and requirements. The system will be able to classify strabismus based on data input by the doctor, such as the results of various diagnostic tests, and will be able to make decision based on the inputs. Finally, the system will also have the ability to store and manage doctor, patient, and treatment records, as well as scheduling appointments and events, making it easier for doctors to access and update this information as needed.

### **1.5 SIGNIFICANCE OF PROJECT**

The significance of this project lies in its ability to assist doctors at the International Islamic University Malaysia (IIUM) in managing patient information and treatment records in a more efficient and organized manner. It can help the doctors to quickly access and update patient information, which can lead to better patient management. Additionally, the system's ability to classify strabismus using a machine learning algorithm can provide accurate and precise diagnosis, which can improve treatment outcomes. For example, by using the system, the doctors at IIUM can easily

access patient's medical history, current treatment records, and predict Accommodate Amplitude value accurately based on patient's age, which can help in making a more informed decision about the patient's treatment. Therefore, by using the proposed system, it definitely can improve the efficiency and effectiveness of the clinic system by reducing the amount of time spent on administrative tasks and allowing doctors to focus more on other important tasks.

## **1.6 THESIS ORGANIZATION**

In Chapter 1, the definition of strabismus and the methods used to diagnose it are discussed, as well as the objectives and criteria for evaluating the success of the proposed system. Chapter 2 reviews the existing literature on strabismus diagnosis and treatment, focusing on three relevant systems and their strengths and weaknesses. Chapter 3 outlines the methods used in the project, including the flow chart, hardware, and software. The results of the project are presented in Chapter 4, which includes a detailed discussion of the project's activities and outcomes, as well as the User Acceptance Testing (UAT) report. Finally, Chapter 5 concludes the project, presents future plans and recommendations, and summarizes the overall findings.

## **CHAPTER 2**

### **LITERATURE REVIEW**

#### **2.1 INTRODUCTION**

Chapter 2 reviews related studies regarding existing eye clinic systems and current machine learning algorithms used for classification and prediction. The comparison's findings will be used to influence the development of this project in order to improve user experience and system performance.

#### **2.2 REVIEW OF EXISTING SYSTEMS**

There are three existing systems that are somewhat similar to the proposed system. This allows a comparison of these systems to be made in order to discover their features, strengths, and weaknesses, as well as what aspects of the proposed system can be improved. The chosen eye clinic systems are Odoo Eye Clinic Management, Optic Clinic and Smart Eye Care.

##### **2.2.1 Odoo Eye Clinic Management**

Odoo Eye Clinic Management is a specific tool designed for ophthalmology professionals. It can be used to keep track of all of the patients record who come to the eye clinic for making appointment and treatment. The features also include scheduling, billing, treatment plan and so on. The system is a web-based system that runs on a web browser, and it costs \$500 to register.

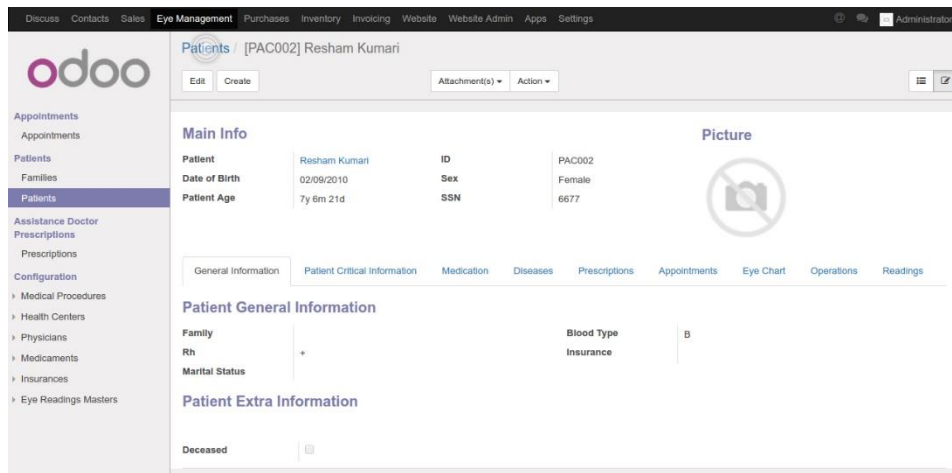


Figure 2.1 Patient information interface of Odoo Eye Clinic Management

Based on the Figure 2.1, doctors can view the patient information by their full name, date of birth, age, and sex. It can also keep records of the patient's family information, making it easy to schedule future appointments. Other treatment information including medication, operation and disease are recorded to provide doctors with further information about patient's eye.

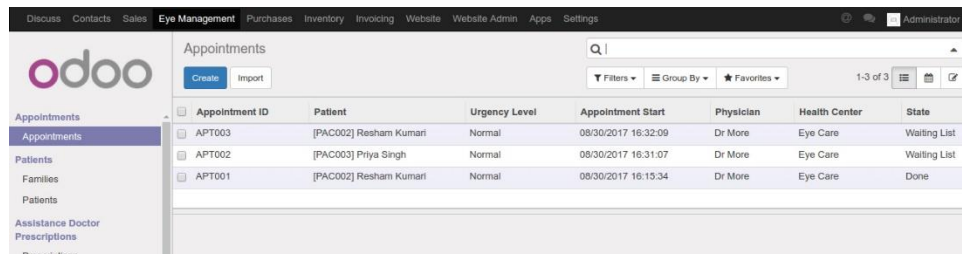


Figure 2.2 Appoinment record interface of Odoo Eye Clinic Management

In addition, doctors are able to record appointment record of their patients like shown in the Figure 2.2. Doctors can save appointment records by entering patient information, appointment date, and urgency level into the system. Doctors also can decide where to do the treatment as the health centre field is provided. The system also has a notification feature, in which appointments that must be completed on a specific day are notified automatically to the physician as a reminder.

## 2.2.2 Optic Clinic

Optic clinic is an open source web-based eye clinic management system that is available to download for free on Github.com. It is developed by a single developer with the intention of optimising the current healthcare practice. It has several capabilities including manages patient records, record prescription for patient, check patients' history, manage staff records and handle bookings. There are two users for the system which are doctor and patient.

Doctors can add new patient by click the <<add patient>> button in the manage patient interface like shown in the Figure 2.3. Patient's first name, last name and phone number are recorded for recognising each patient so that it is easier to trace them in the future. Doctors also able to edit or delete the patient in the system.

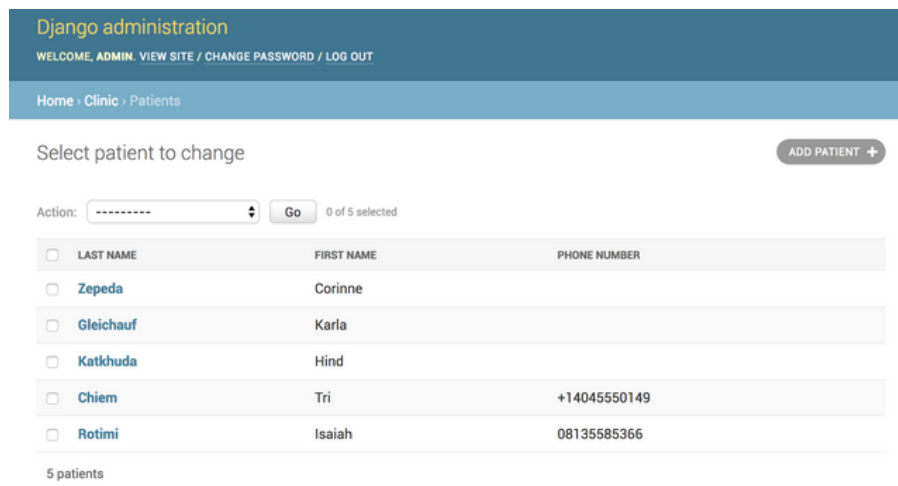


Figure 2.3 Manage patient interface of Optic Clinic

Patient can book their eye treatment appointment in the system. Meanwhile, doctors able to view all the booking that their patient made like shown in the Figure 2.4 below. They also can add booking by clicking the <<add book>> button which requires them to input the patient's name, phone number, and the prescribed date and time so that the patient can see them at that particular time. On the other hand, doctors also can view the patient's name that will be consulted on certain day by applying search filter that is provided. If the patient forgets his or her appointment, the doctor can remind them by calling the phone number that is given and rescheduling the appointment for the following day if it is too late.

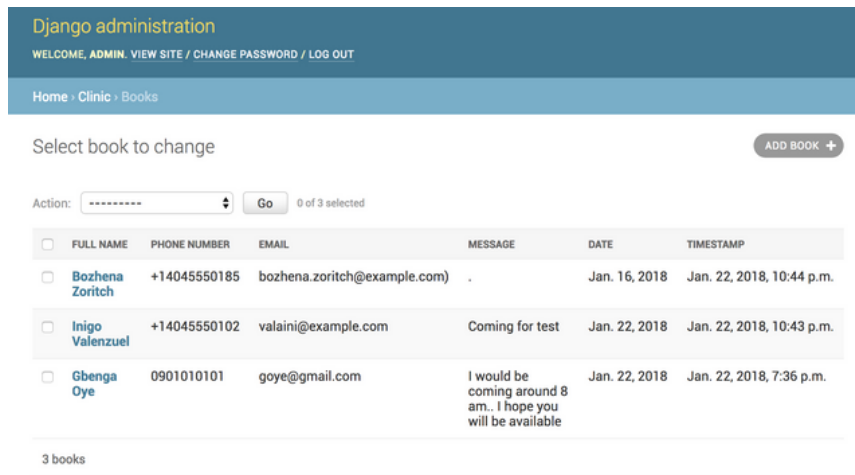


Figure 2.4 Booking interface of Optic Clinic

### 2.2.3 Smart Eye Care

Smart Eye Care is a famous eye hospital management software that is utilised by a lot of eye hospitals in India and around the world. It is developed by Spark Systech, a software development company based in India. Users need to contact their help support to purchase and download the system. The system is only available for Windows operating system and can be run offline without an active internet connection.

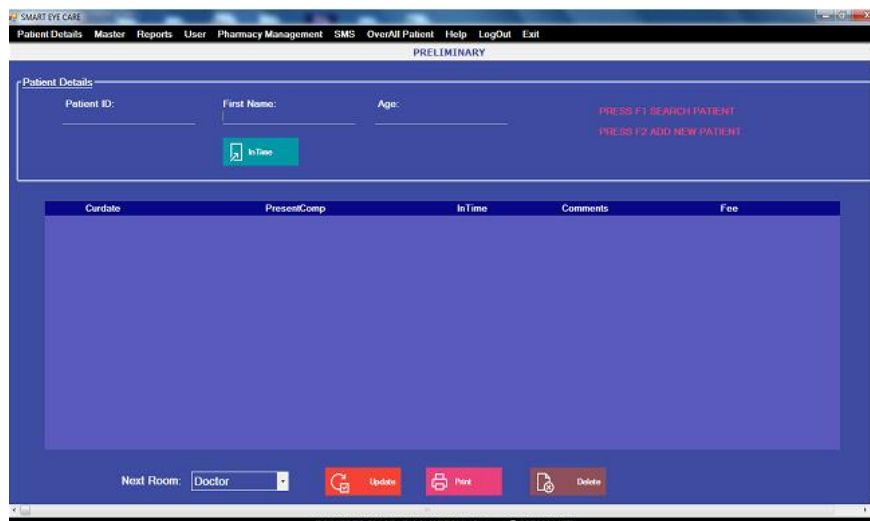


Figure 2.5 Patient details interface of Smart Eye Care



Figure 2.5 shows the patient details interface where doctors can manage patient information by add, update, delete and search for patients. Doctors can use the filters provided to search patients by entering the patient ID, first name, or age. The system also allows doctor to use keyboard shortcuts to add new patient or search patient. The print module is also available for doctors to print the patient information to view in physically on paper.

The screenshot displays a patient registration form with the following fields and controls:

- PatientID:** 11009
- Gender:** Radio buttons for Male (selected) and Female.
- Address 1:** Text input field.
- State:** Text input field.
- Mobile Number:** Text input field.
- Marital Status:** Dropdown menu with 'Married' selected.
- Photo:** Image upload area with a camera icon.
- First Name:** Text input field.
- Age:** Text input field.
- Address 2:** Text input field.
- Pin Code:** Text input field.
- Email ID:** Text input field.
- Assigned Doctor:** Dropdown menu with 'HarwoodS' selected.
- Add Finger Print:** Image upload area with a fingerprint icon.
- Sur Name:** Text input field.
- Date Of Birth:** Date picker showing '27/10/2020'.
- City:** Text input field.
- Phone Number:** Text input field.
- Patient Occupation:** Text input field.
- Patient Remarks:** Dropdown menu with 'REGULAR' selected.
- Referred By:** Text input field.
- Dr. Mobile No.:** Text input field.
- ID PROOF:** Text input field.

At the bottom of the form, there are four buttons: 'Save' (green), 'Print ID Card' (red), 'Print Case Sheet' (teal), and 'Exit' (orange).

Figure 2.6 Add new patient interface of Smart Eye Care

By referring to the Figure 2.6, doctors can add new patient by entering patient ID, full name, gender, age and so on. The system also offers add fingerprint module to verify patient identity. Doctors also can add patient photo so that it will be easier to recognise the patient when they visit the clinic. After finished filling in all required fields, doctors can click <<Save>> button to store the patient information in the database.

The screenshot shows the 'APPOINTMENTS' module interface. It features a form with the following fields and values:

- Patient ID:** 1104
- Patient Name:** A
- Mobile No:** 99432578
- Age:** 23 Years 1 Month
- Schedule Date:** 27-Oct-20
- Time:** 10:27:48 AM
- Address:** link
- Appointment Type:** (dropdown menu)
- Doctor Name:** Harshad

Below the form is a search section with the following fields:

- SEARCH:** (dropdown menu)
- Patient Name:** (text input)
- Mobile No:** (text input)
- From Date:** 27-Oct-20
- To Date:** 28-Nov-20

At the bottom of the interface, there is a table header with columns: PatientID, PatientName, Mobile, Age, Address, ScheduledDate, Time, Doctor\_Name, Appointment\_Type, and Modified\_Name. Below the header is a 'COUNT:' label and three action buttons: Update, Print, and SMS.

Figure 2.7 Appointment module of Smart Eye Care

Figure 2.7 illustrates the appointment module for the system to allow doctors record appointment information of patient's eye treatment. Doctors may enter the patient ID that is already added and the patient information like their name, mobile number and age field will automatically be shown in each text field. The system can also set the appointment date and time, as well as assign which doctors will assess the patient.

### 2.3 COMPARISON BETWEEN EXISTING SYSTEMS AND PROPOSED SYSTEM

The section compares existing systems and proposed system based on various aspects and parameters. Table 2.1 below summarises each specification.

Table 2.1 Comparison between existing systems and proposed system

Specification	Odoo Eye Clinic Management	Optic Clinic	Smart Eye Care	Proposed System
Platform	Web-Based	Web-Based	Windows	Web-Based
Connection Type	Online	Online	Offline	Online
Pricing	\$500 for lifetime	-	Not mentioned.	-
Features	<p><b>1) Book Appointment</b> Doctor can book appointment for specific patient by assigning time and date.</p> <p><b>2) Eye Reading</b> Through appointment, patient's eye reading, and chart can be viewed by doctors on particular eyes that are being diagnose.</p>	<p><b>1) Manage Patient Record</b> Doctors and staffs can manage patient record by add, edit, and delete patient.</p> <p><b>2) Check Patient History</b> Patient history can be traced through search patient function.</p>	<p><b>1) Patient Management with Fingerprint</b> Doctors can manage patient information along with their fingerprint for better security and identification.</p> <p><b>2) Consultation Module</b> Doctors can consult patient and give comments on the consultation.</p>	<p><b>1) Doctor Management</b> Doctor information can be managed, and their performed treatments can be keep tracked easily.</p> <p><b>2) Patient Management</b> Patient information can be managed, and doctors can easily follow up the treatment on specific patient.</p>

	<p><b>3) Billing</b> Billing for appointment can be made upon completing treatment and invoice will automatically be generated.</p>	<p><b>3) Handle Booking</b> Doctors can set booking date for patient treatment.</p>	<p><b>3) Treatment Schedule</b> Treatment schedule is automatically generated by the system based on eye disease.</p>	<p><b>3) Treatment Management</b> Keep track of all performed treatment in one place.</p> <p><b>4) Appointment Management</b> It can manage appointments and events, allowing for better scheduling and organization of the clinic.</p> <p><b>5) Automated Strabismus Classification</b> Able to classify accommodate amplitude and lag of accommodation diagnosis by predicting its result for both eyes accurately.</p>
--	---	---	---	---

Advantages	<ul style="list-style-type: none"> <li>• Ease of use where the user interface is user friendly and neat.</li> <li>• Cashless since the payment of treatment can be made online.</li> </ul>	<ul style="list-style-type: none"> <li>• Open source so anyone can modify the code.</li> </ul>	<ul style="list-style-type: none"> <li>• Faster accessibility where patient just need to scan their fingerprint to access their profile.</li> <li>• Treatment information are provided with infographics.</li> </ul>	<ul style="list-style-type: none"> <li>• Saves doctor time which machine learning will predict the binocular vision automatically.</li> </ul>
Disadvantages	<ul style="list-style-type: none"> <li>• No free or trial versions are provided.</li> </ul>	<ul style="list-style-type: none"> <li>• Bad user interface since all modules look the same.</li> <li>• Hard to navigate because there is no navigation menu provided.</li> </ul>	<ul style="list-style-type: none"> <li>• Bad user interface and poor choice of colours.</li> <li>• Only available on Windows operating system.</li> </ul>	<ul style="list-style-type: none"> <li>• May requires high-end machine to train the algorithm model.</li> </ul>

## **2.4 CURRENT ALGORITHMS FOR CLASSIFICATION**

The section reviews current machine learning algorithms used to classify or make prediction based on existing datasets. There are three classification algorithms that are being reviewed including Fuzzy Logic, Case-Based Reasoning, and Artificial Neural Network.

### **2.4.1 Fuzzy Logic**

Fuzzy logic is one of the classification algorithms that uses “degrees” of truth rather than the traditional “true or false” or Boolean value (0 or 1) that the modern computer is built on. Unlike two-valued Boolean logic, fuzzy logic is multi-valued that involves a range of logical values from 0 (totally false) to 1 (totally true). A numerical value that are being applied to the eye treatment such as age and date of birth, are required to model fuzzy logic. Then, fuzzy rules can be determined by combining the values that doctors are free to choose according to their preferences. In addition, the output value based on the fuzzy rules will be converted into crisp value through defuzzification. For example, rather than predicting whether a patient has strabismus or not, the fuzzy logic result will be how much (in percentage) of strabismus that the patient has.

### **2.4.2 Case-Based Reasoning**

Case-Based Reasoning is a type of analogical reasoning algorithm in which the solution to a new query case is derived from a database of previously solved cases (Lamy, Sekar, Guezennec, Bouaud, & Séroussi, 2019). The output of the new case in CBR will follow the one that has the highest similarity in existing datasets. There are two components that the CBR algorithm needs which are specification and solution. The specification is represented by a set of attributes and values, whereas the solution represents the problem solving for such case. Besides, CBR also contains four cycles including retrieve, reuse, revise and retain in order to make a decision.

### 2.4.3 Artificial Neural Network

The neural network architecture used in this system, as depicted in Figure 2.8, is composed of multiple layers of interconnected neurons. The input layer receives data from the user, such as accommodative amplitude and lag of accommodation measurements, and passes it through the network to the hidden layer. The hidden layer applies non-linear functions, such as the sigmoid or rectified linear unit (ReLU) activation function, to the input data to extract meaningful features. Finally, the output layer produces the final classification result, such as whether the patient has normal, accommodative excess or insufficient, or failed condition. Each neuron in the network is connected to the next layer through links, called synapses, which have numerical weights attached to them that indicate their importance in the classification process. These weights are adjusted during the training phase of the network to optimize its performance.

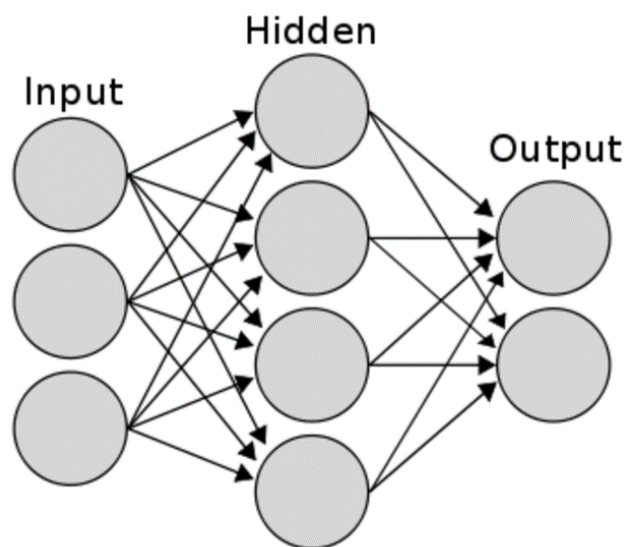


Figure 2.8 Basic representation of a neural network application

## 2.5 COMPARISON BETWEEN ALGORITHMS

Table 2.2 shows the comparison of machine learning classification algorithms including its methodology, advantages, and disadvantages.

Table 2.2 Comparison between machine learning classification algorithms

Criteria	Fuzzy Logic	Case-Based Reasoning	Artificial Neural Network
Methodology	<ul style="list-style-type: none"> <li>Deals with degrees of truth to predict the occurrence of an event.</li> <li>Output is based on membership function and fuzzy rules.</li> </ul>	<ul style="list-style-type: none"> <li>Solve new case based on past experience that has the highest similarity.</li> <li>Involves four cycles to predict output.</li> </ul>	<ul style="list-style-type: none"> <li>Structured with layers of neurons that are linked to each other.</li> <li>Weightage decides the importance of each neuron.</li> </ul>
Advantages	<ul style="list-style-type: none"> <li>Flexible since it allows modification on fuzzy rules any time.</li> <li>Easy to implement as the process is not as complicated as other two algorithms.</li> <li>Able to handle multiple sorts of inputs at the same time and make precise decision.</li> </ul>	<ul style="list-style-type: none"> <li>No knowledge elicitation required to create rules or methods.</li> <li>The larger the model, the more accurate the prediction.</li> </ul>	<ul style="list-style-type: none"> <li>Fault tolerance where if some cells are corrupted, it does not prevent the algorithm from predicting output.</li> <li>Information is stored in the network rather than database which provide better accessibility.</li> </ul>
Disadvantages	<ul style="list-style-type: none"> <li>Human intelligence and expertise are fully reliant on them.</li> <li>Not extensively used due to inaccuracy of dataset</li> </ul>	<ul style="list-style-type: none"> <li>Requires high-end computer to train and test large datasets.</li> <li>Old cases may be poor or inaccurate which will influence the output.</li> </ul>	<ul style="list-style-type: none"> <li>Requires high-end computer to train and test large datasets.</li> <li>Difficult to implement as there are many processes involved to model the algorithm.</li> </ul>



## **2.6 CHAPTER SUMMARY**

Based on the comparison on three existing eye clinic systems, the proposed system will be more functional, and features will be designed specifically for the Binocular Vision IIUM Clinic. The proposed classification algorithm that has been discussed will be determined on Chapter three based on parameters that the doctors used to classify binocular vision type.

## **CHAPTER 3**

### **METHODOLOGY**

#### **3.1 INTRODUCTION**

In this chapter, the methodology of the project will be discussed. The Software Development Life Cycle model also will be chosen as guidance on work phases for the development of Binocular Vision Management system. After enough consideration, Agile SDLC methodology is selected as it is the most suitable model for this project.

#### **3.2 METHODOLOGY**

This project's software development life cycle will be based on the Agile model. It focuses on incremental delivery methodologies, which execute minor increments while including the client in the development process for quick feedback and modifications. Also, it is a conceptual framework for software engineering that begins with a planning phase and progresses through iterative and incremental interactions throughout the project's life cycle (Alsaqqa, Sawalha, & Abdel-Nabi, 2020). Thus, overhead can be reduced during the development process since this model can adapt to changes without affecting the process or requiring extensive rework.

There are multiple reasons to choose the Agile model for this project. Among them is because client satisfaction is the highest priority, and it must be achieved early and continuously until all requirements are met and satisfied. In addition, even late in development, it welcomes changing requirements. This will allow the client to provide feedback on what needs to be altered or improved during the development process. Furthermore, since the project needs to be completed in a short time frame which is less than a year, Agile model is suitable, because it recommends the early and quick delivery

to satisfy client needs. Also, throughout the development process, the software developer and the client must collaborate often so that the client can provide feedback and answer the software developer's questions.



Figure 3.1 Agile methodology and its phases

By referring to Figure 3.1 Agile methodology and its phases, there are six phases involved in the software development of Binocular Vision Management System. The description of each phase is as follow:

**i) Phase 1: Requirements**

During the requirements phase, by referring to Appendix B, the Software Requirement Specification (SRS) is produced to outline the functionality of the proposed system. This SRS also can be used as an input for the following phase. Requirement analysis is a term that refers to all of the actions that are carried out in order to discover the needs of the stakeholders (Babu, Jalaiah, & Bhushanam, 2019). In addition, meetings with supervisor tasks to discuss the project will be held on a regular basis to ask for advice during the system's development. Interview sessions with IIUM eye doctor will also be conducted periodically to gather requirements of the proposed system and to study the basics of strabismus and its parameters to classify the disease. The client also needs to be close with

the developer during development until they are satisfied with the requirements. Furthermore, three existing eye clinic systems were analysed to identify their strengths and weaknesses, as well as which aspects of the weaknesses the proposed system could improve. The process is followed by the revision of the current machine learning classification algorithms for predicting the disease automatically. Thoughtfully, Case-Based Reasoning will be used for the algorithm in this project.

## **ii) Phase 2: Design**

Throughout the design phase, the Software Design Document (SDD) is created to define the architecture of the proposed system based on Laravel framework. Basically, this document specifies the design aspects that are required by the SRS that contains both the software's architectural description and its detailed design (Montalvo, Parra, & R. Polo, 2019). Besides, it also covers data design of the proposed system including the entity relationship diagram and the data dictionary. The prototype of the proposed system will be created in this phase to provide early representations of how the system would look in the future. As a result, before the development phase begins, the IIUM doctor can assess the prototype and request for modifications. In addition, the implementation of chosen machine learning algorithm is also discussed in this phase.

## **iii) Phase 3: Development**

During this phase, the developer will use programming languages and frameworks to build the various components of the system. The development phase can be broken down into several sub-phases, such as:

- i) Database design: Involves creating the database schema for storing patient and doctor information, treatment records, and other data related to the system.

- ii) Front-end development: Focuses on the creating the user interface of the system based on the designed prototype. The overall architecture and framework of the proposed system will be based on the Laravel and Livewire.
- iii) Back-end development: Handles the logic and data processing of the proposed system. In addition, server-side scripts using Node JS and Python library are used to handle the system's functionalities such as patient registration, treatment classification, and appointment management.

#### **iv) Phase 4: Testing**

This phase involves the testing of the proposed system to discover defects or bugs. Thus, the system will be free of bugs and remain functional during operation. In addition, a User Acceptance Test (UAT) will also be conducted to check that the system is complete and meets all the criteria specified during the requirement collecting stage. For example, the doctors will test the system's registration and login process to ensure that it is secure and easy to use. They will also test the patient management module to ensure that it is able to list, search, add, edit, and delete patient records as expected. Additionally, they will test the treatment management module to ensure that it is able to predict the expected AA value accurately, classify strabismus using machine learning algorithm, and display treatment information and its result correctly.

Once UAT is completed and any issues identified have been resolved, the system will be considered ready for deployment. Any flaws discovered will be fixed immediately so that doctors may utilise the system without difficulty.

#### **v) Phase 5: Deployment**

After confirming that the system is ready for release, the proposed system is installed on the servers and made available to the IIUM doctor. To check that the

proposed system is running properly, it will be tested and run on the client's web browser. This phase will also include user training, which will require more documentation. Hence, the doctor can learn how to use the system effectively even they are new users. When all of this is finished, the final iteration of the product may be released into production.

#### **vi) Phase 6: Review**

In the last phase of Agile methodology, the proposed system is examined to ensure that documentation such as requirements, system designs, code, test plans, and test cases are complete and adhere to the plan. The proposed application's developer will provide solutions for resolving problems that occurred throughout the previous phases. Following that, the steps of the software development lifecycle are restarted with a new iteration.

### **3.3 PROJECT REQUIREMENT**

#### **3.3.1 Functional Requirements**

The following are the functional requirements for the binocular vision management system, categorised by module:

##### **Manage Login**

- The system shall allow the doctor to login by entering username and password to access the system.
- The system shall display error message when wrong login credentials are entered.

##### **Dashboard**

- The system shall allow the doctor to view the number of the treatment made by today, this week, this month and overall.

- The system shall allow the doctor to view the statistics of treatment report.

### **Manage Doctor**

- The system shall be able to list all doctors in a table.
- The system shall be able to search the doctors whenever the doctor enters a search keyword in the search doctor field.
- The system shall allow the doctor to add new doctors by filling doctor's information.
- The system shall be able to display the doctor information and their performed treatments.
- The system shall allow the doctor to edit doctor according to the doctor's input.
- The system shall be able to delete the doctors after the doctor confirm their action on the deletion confirmation dialog.

### **Manage Patient**

- The system shall be able to list all registered patients in a table.
- The system shall be able to search the patients whenever the doctor enters a search keyword in the search patient field.
- The system shall allow the doctor to add new patients by filling in patient name, phone number, age, gender, date of birth, occupation, home address, office address, and parent's name.
- The system shall be able to display the patient information and their treatment history.
- The system shall allow the doctor to edit patients according to the doctor's input.
- The system shall be able to delete the patients after the doctor confirm their action on the deletion confirmation dialog.

## **Manage Treatment**

- The system shall allow the doctor to select specific patient for the treatment.
- The system shall classify the accommodate amplitude diagnosis for each eye (Normal or Failed) through Case-Based Reasoning algorithm.
- The system shall classify the lag of accommodation diagnosis for both eyes (Normal or Accommodate Insufficient or Accommodate Excess) through Case-Based Reasoning algorithm.
- The system shall be able to assign person in charge according to their role in the treatment.
- The system shall allow the doctor to add new treatments by filling treatment information.
- The system shall be able to list all treatments made in a table.
- The system shall be able to search the treatments whenever the doctor enters a search keyword in the search treatment field.
- The system shall be able to display the treatment information and its result.
- The system shall allow the doctor to edit treatments according to the doctor's input.
- The system shall be able to delete the treatments after the doctor confirm their action on the deletion confirmation dialog.

## **Manage Appointment**

- The system shall allow doctors to make appointment by creating new event.
- The system shall be able to display the created events in the calendar.
- The system shall allow doctors to drag and drop events to different date.
- The system shall be able to display event details.
- The system shall be able to edit event as required.
- The system shall be able to delete event as required.



### 3.3.2 Non-Functional Requirement

Table 3.1 below shows the non-functional requirements for the proposed system categorised by quality criteria.

Table 3.1 Non-functional requirements

Criteria	Descriptions
Performance	<ul style="list-style-type: none"> <li>• The system shall be able to classify Accommodate Amplitude and Lag of Accommodation diagnosis in not more than 30 seconds.</li> <li>• The system shall be able to classify Accommodate Amplitude and Lag of Accommodation diagnosis of a patient with less than 50% of CPU usage.</li> <li>• The system shall be able to query patient and treatment data without any latency.</li> </ul>
Security	<ul style="list-style-type: none"> <li>• The system shall identify doctor's credential before allowing them to use the system.</li> <li>• The system shall assure that the patient sensitive information is kept secured.</li> <li>• The system shall destroy doctor's session in one hour' time when idling.</li> </ul>
Usability	<ul style="list-style-type: none"> <li>• Novice doctors shall be able to perform login and registration functionality in less than five minutes.</li> <li>• Most doctors shall be able to manage patient and manage treatment within fifteen minutes after a 3-hours introduction to the system.</li> <li>• The system shall be able to adapt its interfaces in different screen sizes.</li> </ul>
Reusability	<ul style="list-style-type: none"> <li>• The system code shall be reusable to minimize the doctor's machine resource.</li> <li>• The system shall minimize coupling between modules by using Laravel design pattern.</li> </ul>

### 3.3.3 Constraints

Table 3.2 depicts the system constraints based on certain criteria.

Table 3.2 System constraints and its descriptions

Criteria	Descriptions
Privacy policy	<ul style="list-style-type: none"><li>• Patient's sensitive data and their treatment records shall not be shared or sold to public or third party.</li></ul>
Software developer	<ul style="list-style-type: none"><li>• The project is developed by only a single developer under the supervision of a supervisor.</li></ul>
Culture	<ul style="list-style-type: none"><li>• Any symbols or graphics that could be regarded offensive to any culture must be eliminated from the system.</li></ul>
Eye measuring tool	<ul style="list-style-type: none"><li>• The system requires the doctor to use RAF ruler to measure AA distance value (in Dioptre) of a patient's eyes.</li></ul>

### 3.3.4 Limitations

Table 3.3 depicts the system constraints based on certain criteria.

Table 3.3 System limitations and its descriptions

Criteria	Descriptions
Web browser version	<ul style="list-style-type: none"><li>• The system may not be supported with old version of web browsers.</li></ul>
Internet connectivity	<ul style="list-style-type: none"><li>• The system cannot be used without an active internet connectivity since it requires connection with database to access the patient and treatment data.</li></ul>
Server downtime	<ul style="list-style-type: none"><li>• Doctors may not be able to access the system during system maintenance since the system needs to be fixed and diagnosed periodically.</li></ul>
Lack of professional	<ul style="list-style-type: none"><li>• Because the system is new, and there may be no existing system that uses a case-based reasoning algorithm to classify strabismus, more skilled personnel with expertise in web application and machine learning will be necessary in the future to improve the system.</li></ul>

### 3.4 STRABISMUS DIAGNOSIS

#### 3.4.1 Accommodative Amplitude (AA)



Figure 3.2 Accommodative Amplitude diagnosis

Accommodative amplitude (AA) is a measure of the maximum amount that the eyes can focus on an object at different distances. It is one of the diagnostic tests used to evaluate a patient for strabismus. By referring to the figure 3.2, the test is typically performed using a device called a RAF ruler (Ruler for Accommodative Function). RAF ruler is a hand-held device that consist of a series of lines with different thickness and font size. The device is moved toward or away from the eyes while the patient is fixating a target, until the patient can no longer clearly read the lines. The distance at which this occurs is the patient's accommodative amplitude.

For example, if a patient has a 20 dioptre (a unit used for accommodative amplitude), this means that the patient can read the lines on the RAF ruler 20 dioptres away with clear vision. And if the patient has 10 dioptre accommodative amplitude, this means that the patient can only read the lines on the RAF ruler 10 dioptres away. The test is typically done for three repetitions to get accurate measurement for each eye. In short, accommodative amplitude is affected by age and it decreases as we get older. Patient that is older than 40 years old is no longer valid for this type of test.

### 3.4.2 Lag of Accommodation (LA)



Figure 3.3 Lag of Accommodation diagnosis

The lag of accommodation (LA) is the delay in the eyes' ability to focus on an object as it moves closer or farther away. It is another diagnostic test used to evaluate a patient for strabismus. The test is typically performed using a device called a plus lens flipper test. During this test (refer figure 3.3), the doctor will place a plus lens (convex lens) in front of the patient's eye, and then have the patient focus on a target, such as a small letter or symbol, at a fixed distance. Then, the doctor will gradually move the lens closer to the eye while the patient continues to fixate on the target. The point at which the patient can no longer maintain clear focus on the target is the patient's lag of accommodation.

For example, if a patient has a 1 dioptre lag of accommodation, this means that the patient's eyes take 1 dioptre closer for them to be able to focus on the target. This test is usually done for both eyes separately and the results are compared to each other and to the normal range for the patient's age.

The lag of accommodation test can be classified into three categories:

- i) **Accommodative Excess (AE):** This occurs when the patient's eyes take more dioptres of plus lens before they can maintain clear focus on the target. This means the eyes have difficulty focusing on nearby objects.
- ii) **Accommodative Insufficient (AI):** This occurs when the patient's eyes take fewer dioptres of plus lens before they can maintain clear focus on the target. This means that the eyes have difficulty focusing on distant objects.

- iii) **Normal (N):** When the patient's eyes take the appropriate dioptres of plus lens for their age, it's considered normal range for their lag of accommodation.

### **3.5 CASE-BASED REASONING ALGORITHM JUSTIFICATION**

Case-Based Reasoning (CBR) is a suitable algorithm for this project because it is well-suited for making decisions based on past cases. CBR is a form of Artificial Intelligence that learns by analysing and comparing previous cases. The system stores information about previous cases and their outcomes, and when faced with a new case, it retrieves the most similar cases from its memory and uses the information from those cases to make prediction.

For this project, the CBR algorithm will be able to learn from previous cases of strabismus and their corresponding diagnostic test results, such as accommodative amplitude, lag of accommodation, monocular and binocular accommodative facility, and relative accommodation. The system will then be able to make a decision on whether or not a new patient has strabismus based on the similarity of their case to previous cases. For example, assume that the system has previously encountered a case of a patient with a normal accommodative amplitude, and accommodative excess on monocular and binocular accommodative facility. The CBR algorithm will be able to retrieve this case from its memory and use the information from this case to decide for a new patient with similar test results.

Furthermore, CBR algorithm can be improved over time by feeding it more data and updating its knowledge base with new cases and their outcomes. This allows the system to become more accurate and efficient over time. In addition, CBR can be useful in this project as it can help to classify strabismus while considering the patient's age and other medical history and other underlying conditions, which may have a significant impact on the diagnosis. By including these additional factors in the case-based reasoning, the algorithm will be able to improve the accuracy of its classification.

Overall, CBR is a suitable algorithm for this project because it is well-suited for making decisions based on past cases, it can handle incomplete or ambiguous

information, it can learn from previous cases and it can also consider the patient's age or other medical history, which may be essential for the diagnosis.

### 3.6 PROPOSED DESIGN

#### 3.6.1 Context Diagram

In Figure 3.4, the system's context diagram represents the connection between external entities and internal system. There are two entities which are doctor and case-based reasoning algorithm. The arrows illustrate the system's process, whether it's coming in or going out.

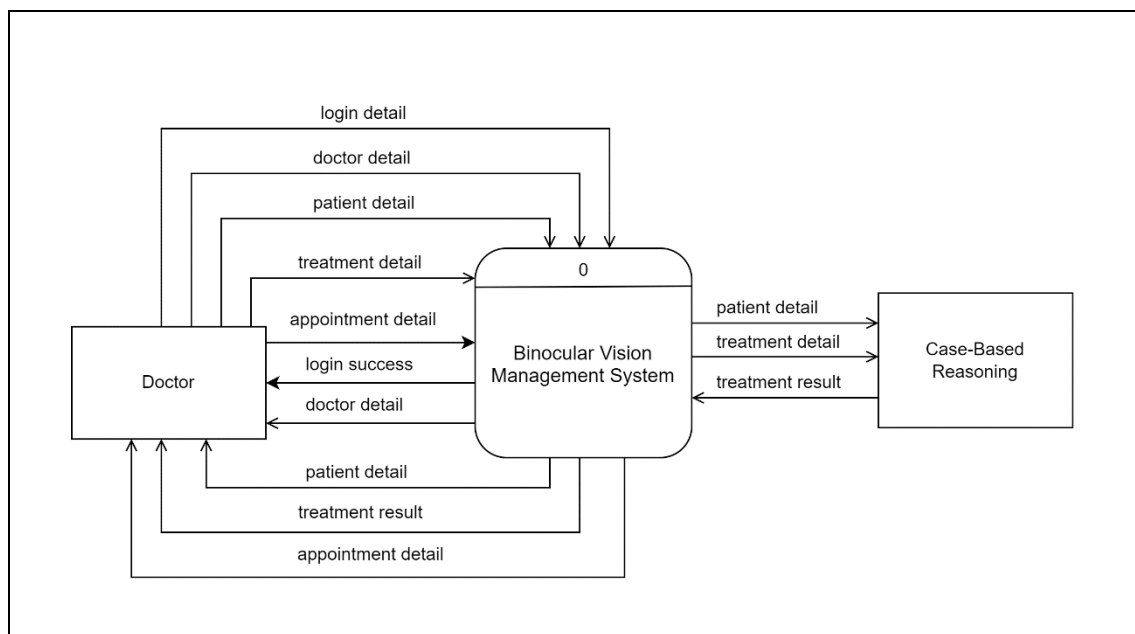


Figure 3.4 Context Diagram

### 3.6.2 Use Case Diagram

The Figure 3.5 shows the use case diagram of the proposed system. There are five modules in the system which are, manage login, manage doctor, manage patient, manage treatment and manage appointment.

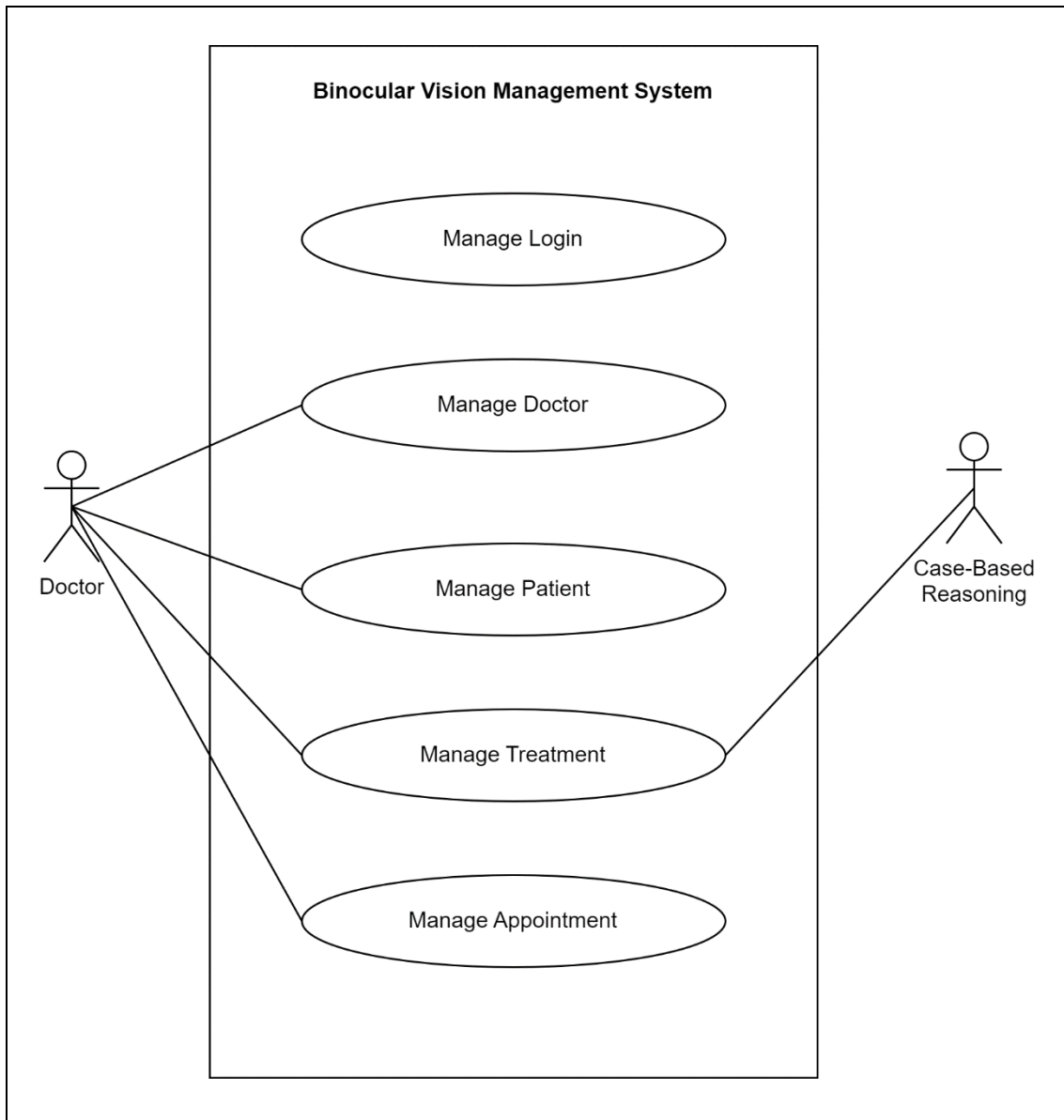


Figure 3.5 Use Case Diagram

### 3.6.2.1 Use Case Description of Manage Login Module

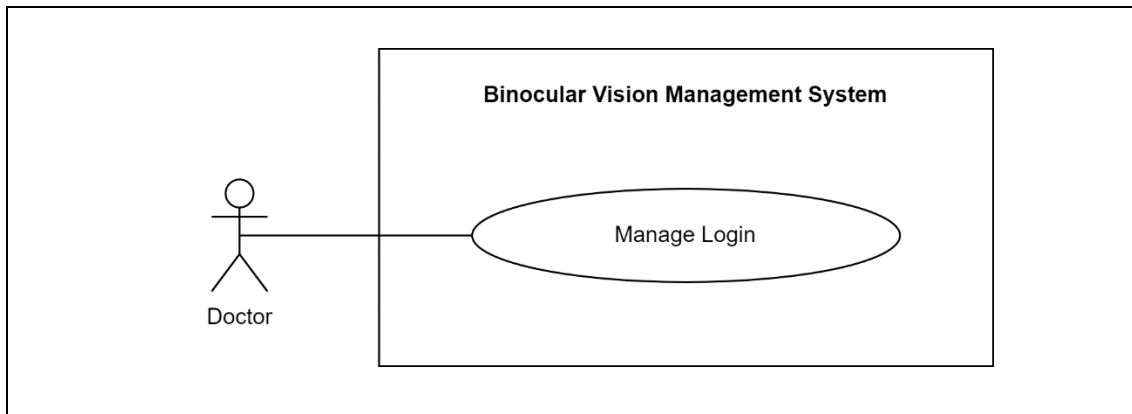


Figure 3.6 Use Case Description Diagram for Manage Login

<b>Use Case</b>	Manage Login
<b>Brief Description</b>	This use case is used to login using doctor's username and password in order to access the system.
<b>Actor</b>	Doctor
<b>Pre-Conditions</b>	The doctor account should be registered in the system for the login function to work.
<b>Basic Flow</b>	<ol style="list-style-type: none"> <li>1. The doctor may click on &lt;&lt;Sign Up&gt;&gt; if he or she does not have any account. [A1: Sign Up]</li> <li>2. The doctor enters username.</li> <li>3. The doctor enters password. [A2: Show password]</li> <li>4. The doctor clicks on &lt;&lt;Login&gt;&gt; button.</li> <li>5. The system validates and verifies the login credentials in the database. [E1: Invalid username or password]</li> <li>6. The system redirects the doctor to the dashboard page.</li> <li>7. The use case ends.</li> </ol>
<b>Alternative Flow</b>	<p><b>A1: Sign Up</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the doctor account registration page.</li> <li>2. The doctor creates a new username.</li> <li>3. The system checks whether the entered username is already existed in the database. [E2: Username already taken]</li> <li>4. The doctor enters email address.</li> </ol>



	<p>5. The system validates email address input. [E3: Invalid email address format]</p> <p>6. The doctor creates a new password.</p> <p>7. The doctor enters full name.</p> <p>8. The doctor enters phone number.</p> <p>9. The doctor clicks on &lt;&lt;Sign Up&gt;&gt; button to register new account.</p> <p>10. The system saves the doctor account information in the database.</p> <p>11. The system redirects the doctor to login page.</p> <p>12. The use case continues to step 2 in basic flow.</p> <p><b>A2: Show password</b></p> <p>1. The system converts the password format into plain text in the password input field.</p> <p>2. The use case continues to step 4 in basic flow.</p>
<p><b>Exception Flow</b></p>	<p><b>E1: Invalid username or password</b></p> <p>1. The system displays error message that the username or password is invalid.</p> <p>2. The doctor re-enter the valid username and password.</p> <p>3. The use case continues to step 4 in basic flow.</p> <p><b>E2: Username already taken</b></p> <p>1. The system displays error message that the username has already been taken.</p> <p>2. The doctor re-enter new username.</p> <p>3. The use case continues to step 4 in Alternative Flow A1.</p> <p><b>E3: Invalid email address format</b></p> <p>1. The system displays error message that the email address format is invalid.</p> <p>2. The doctor re-enter the valid email address.</p> <p>3. The use case continues to step 6 in Alternative Flow A1.</p>

<b>Post-Conditions</b>	The user can create a new account and able to access the system by login with their username and password.
<b>Rules</b>	-

### 3.6.2.2 Use Case Description of Manage Doctor Module

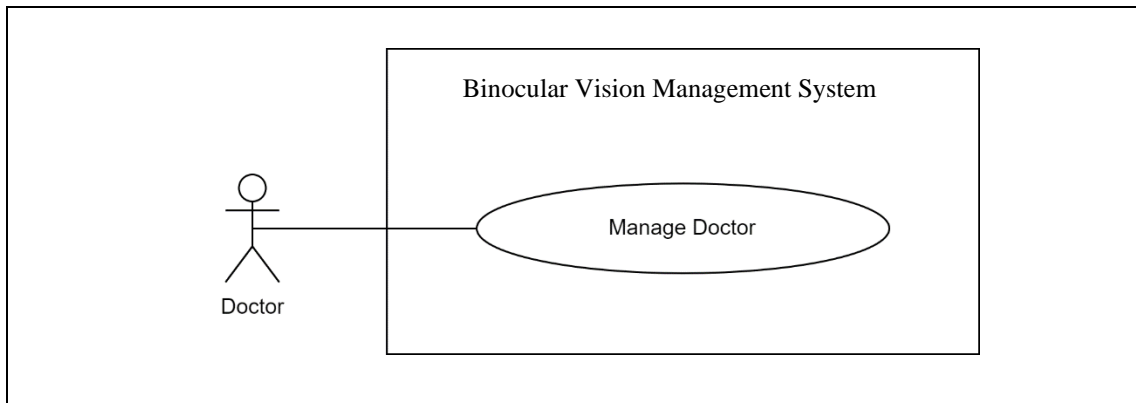


Figure 3.7 Use Case Description Diagram for Manage Doctor

<b>Use Case</b>	Manage Doctor
<b>Brief Description</b>	This use case indicates the manage doctor function where doctors able to add, edit, delete, and search for doctors. The doctors are also able to view the treatment history of a doctors.
<b>Actor</b>	Doctor
<b>Pre-Conditions</b>	<ol style="list-style-type: none"> <li>1. The doctor has logged in with their username and password.</li> <li>2. The doctor has an active internet connection.</li> </ol>
<b>Basic Flow</b>	<ol style="list-style-type: none"> <li>1. The doctor clicks on the manage doctor menu.</li> <li>2. The system redirects the doctor to the manage doctor page.</li> <li>3. The system displays registered doctor in a table.</li> <li>4. The doctor able to:             <ol style="list-style-type: none"> <li>a) Add new doctor [A1: Add doctor]</li> <li>b) Select one patient doctor and clicks on view icon [A2: View doctor]</li> <li>c) Select one doctor record and clicks on edit icon [A3: Edit doctor]</li> </ol> </li> </ol>

	<p>d) Select one doctor record and clicks on delete icon [A4: Delete doctor]</p> <p>e) Enter search keyword [A5: Search doctor]</p> <p>5. The use case ends.</p>
<p><b>Alternative Flow</b></p>	<p><b>A1: Add doctor</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the add new doctor page.</li> <li>2. The doctor enters doctor information.</li> <li>3. The doctor clicks on the &lt;&lt;Submit&gt;&gt; button.</li> <li>4. The system saves the doctor record in the database. [E1: Required field]</li> <li>5. The system displays success message.</li> <li>6. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A2: View doctor</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the view doctor information page.</li> <li>2. The system retrieves the doctor information from the database.</li> <li>3. The system retrieves the treatment history of the doctor if available.</li> <li>4. The doctor views the doctor information.</li> <li>5. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A3: Edit doctor</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the edit doctor page.</li> <li>2. The system retrieves the doctor information from the database.</li> <li>3. The system populates the doctor information to each input field.</li> <li>4. The doctor edits the doctor information.</li> <li>5. The doctor clicks on &lt;&lt;Update&gt;&gt; button.</li> <li>6. The system updates the doctor record in the database.</li> <li>7. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A4: Delete doctor</b></p> <ol style="list-style-type: none"> <li>1. The system displays doctor deletion confirmation dialog.</li> </ol>

	<ol style="list-style-type: none"> <li>2. If the doctor clicks yes, the system deletes the doctor record in the database. Else, the systems abort the operation.</li> <li>3. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A5: Search doctor</b></p> <ol style="list-style-type: none"> <li>1. The doctor enters search keyword in the search doctor field.</li> <li>2. The system updates the doctor table according to search input. [E2: Doctor record unavailable]</li> <li>3. The doctor empties the search doctor field.</li> <li>4. The use case continues to step 3 in basic flow.</li> </ol>
<b>Exception Flow</b>	<p><b>E1: Required field</b></p> <ol style="list-style-type: none"> <li>1. The system displays error message that the input field cannot be empty.</li> <li>2. The doctor input data to empty field.</li> <li>3. The use case continues to step 3 in Alternative Flow A1.</li> </ol> <p><b>E2: Doctor record unavailable</b></p> <ol style="list-style-type: none"> <li>1. The system displays alert message that the doctor record not found.</li> <li>2. The doctor re-enter valid search keyword.</li> <li>3. The use case continues to step 2 in Alternative Flow A5.</li> </ol>
<b>Post-Conditions</b>	The doctor record can be managed, and treatment of the doctor can be performed.
<b>Rules</b>	-

### 3.6.2.3 Use Case Description of Manage Patient Module

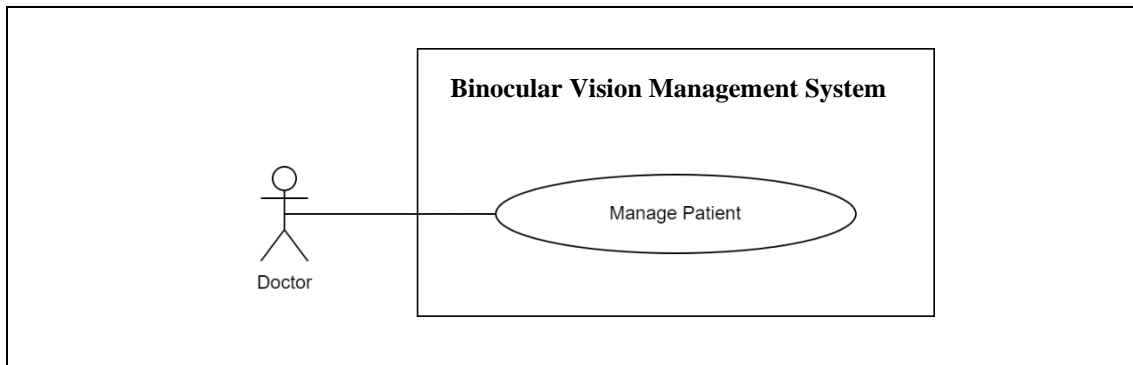


Figure 3.8 Use Case Description Diagram for Manage Patient

<b>Use Case</b>	Manage Patient
<b>Brief Description</b>	This use case indicates the manage patient function where doctors able to add, edit, delete, and search for patients. The doctors are also able to view the treatment history of a patient.
<b>Actor</b>	Doctor
<b>Pre-Conditions</b>	<ol style="list-style-type: none"> <li>1. The doctor has logged in with their username and password.</li> <li>2. The doctor has an active internet connection.</li> </ol>
<b>Basic Flow</b>	<ol style="list-style-type: none"> <li>1. The doctor clicks on the manage patient menu.</li> <li>2. The system redirects the doctor to the manage patient page.</li> <li>3. The system displays registered patients in a table.</li> <li>4. The doctor able to:             <ol style="list-style-type: none"> <li>a) Add new patient [A1: Add patient]</li> <li>b) Select one patient record and clicks on view icon [A2: View patient]</li> <li>c) Select one patient record and clicks on edit icon [A3: Edit patient]</li> <li>d) Select one patient record and clicks on delete icon [A4: Delete patient]</li> <li>e) Enter search keyword [A5: Search patient]</li> </ol> </li> <li>5. The use case ends.</li> </ol>
<b>Alternative Flow</b>	<b>A1: Add patient</b> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the add new patient page.</li> </ol>

2. The doctor enters patient name, phone number, age, gender, date of birth, occupation, home address, office address, and parent's name.
3. The doctor clicks on the <<Submit>> button.
4. The system saves the patient record in the database. [E1: Required field]
5. The system displays success message.
6. The use case continues to step 2 in basic flow.

#### **A2: View patient**

1. The system redirects the doctor to the view patient information page.
2. The system retrieves the patient information from the database.
3. The system retrieves the treatment history of the patient if available.
4. The doctor views the patient information.
5. The use case continues to step 2 in basic flow.

#### **A3: Edit patient**

1. The system redirects the doctor to the edit patient page.
2. The system retrieves the patient information from the database.
3. The system populates the patient information to each input field.
4. The doctor edits the patient information.
5. The doctor clicks on <<Update>> button.
6. The system updates the patient record in the database.
7. The use case continues to step 2 in basic flow.

#### **A4: Delete patient**

1. The system displays patient deletion confirmation dialog.
2. If the doctor clicks yes, the system deletes the patient record in the database. Else, the systems abort the operation.
3. The use case continues to step 2 in basic flow.

	<p><b>A5: Search patient</b></p> <ol style="list-style-type: none"> <li>1. The doctor enters search keyword in the search patient field.</li> <li>2. The system updates the patient table according to search input. [E2: Patient record unavailable]</li> <li>3. The doctor empties the search patient field.</li> <li>4. The use case continues to step 3 in basic flow.</li> </ol>
<p><b>Exception Flow</b></p>	<p><b>E1: Required field</b></p> <ol style="list-style-type: none"> <li>1. The system displays error message that the input field cannot be empty.</li> <li>2. The doctor input data to empty field.</li> <li>3. The use case continues to step 3 in Alternative Flow A1.</li> </ol> <p><b>E2: Patient record unavailable</b></p> <ol style="list-style-type: none"> <li>1. The system displays alert message that the patient record not found.</li> <li>2. The doctor re-enter valid search keyword.</li> <li>3. The use case continues to step 2 in Alternative Flow A5.</li> </ol>
<p><b>Post-Conditions</b></p>	<p>The patient record can be managed, and treatment of the patient can be performed.</p>
<p><b>Rules</b></p>	<p>-</p>

### 3.6.2.4 Use Case Description of Manage Treatment Module

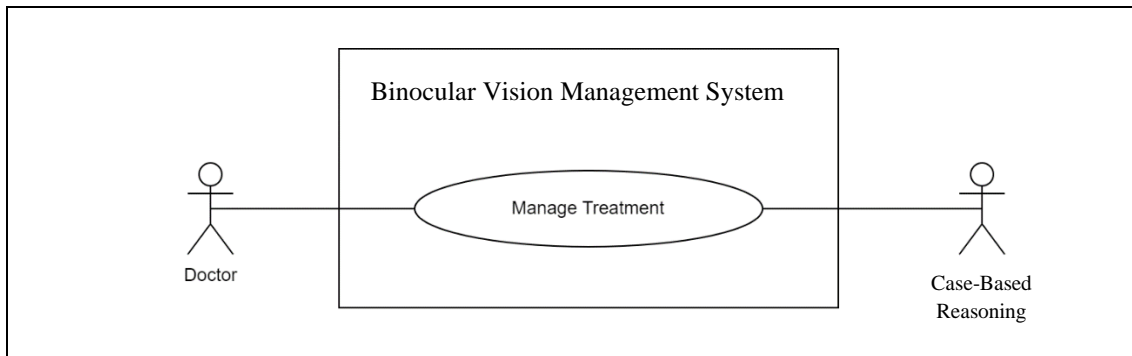


Figure 3.9 Use Case Description Diagram for Manage Treatment

<b>Use Case</b>	Manage Treatment
<b>Brief Description</b>	This use case indicates the manage treatment function where doctors able to add, edit, delete, and search for treatments.
<b>Actor</b>	Doctor, Case-Based Reasoning
<b>Pre-Conditions</b>	<ol style="list-style-type: none"> <li>1. The doctor has logged in with their username and password.</li> <li>2. The doctor has an active internet connection.</li> <li>3. The doctor has registered at least one patient.</li> </ol>
<b>Basic Flow</b>	<ol style="list-style-type: none"> <li>1. The doctor clicks on the manage treatment menu.</li> <li>2. The system redirects the doctor to the manage treatment page.</li> <li>3. The system displays treatment records in a table.</li> <li>4. The doctor able to: <ol style="list-style-type: none"> <li>a) Add new treatment [A1: Add treatment]</li> <li>b) Select one treatment record and clicks on view icon [A2: View treatment]</li> <li>c) Select one treatment record and clicks on edit icon [A3: Edit treatment]</li> <li>d) Select one treatment record and clicks on delete icon [A4: Delete treatment]</li> <li>e) Enter search keyword [A5: Search treatment]</li> </ol> </li> <li>5. The use case ends.</li> </ol>
<b>Alternative Flow</b>	<p><b>A1: Add treatment</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the add new treatment page.</li> <li>2. The doctor enters patient NRIC number.</li> </ol>



	<ol style="list-style-type: none"> <li>3. The system populates the patient information to each field. [E1: Patient not found]</li> <li>4. The system calculates the expected AA value according to patient's age.</li> <li>5. The doctor enters AA values for three repetitions to each patient eyes.</li> <li>6. The doctor enters treatment remark.</li> <li>7. The doctor clicks on the &lt;&lt;Calculate&gt;&gt; icon button.</li> <li>8. The case-based reasoning predicts the squint type based on the entered AA values.</li> <li>9. The doctor clicks on the &lt;&lt;Submit&gt;&gt; button.</li> <li>10. The system saves the treatment record in the database. [E2: Required field]</li> <li>11. The system displays success message.</li> <li>12. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A2: View treatment</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the view treatment information page.</li> <li>2. The system retrieves the treatment information from the database.</li> <li>3. The system retrieves the patient information from the database.</li> <li>4. The doctor views the treatment information.</li> <li>5. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A3: Edit treatment</b></p> <ol style="list-style-type: none"> <li>1. The system redirects the doctor to the edit treatment page.</li> <li>2. The system retrieves the patient information from the database.</li> <li>3. The system populates the patient information to each input field.</li> <li>4. The system retrieves the treatment information from the database.</li> <li>5. The system populates the treatment information to each input field.</li> </ol>
--	--

	<ol style="list-style-type: none"> <li>6. The doctor edits the treatment information.</li> <li>7. The doctor clicks on &lt;&lt;Update&gt;&gt; button.</li> <li>8. The system updates the treatment record in the database.</li> <li>9. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A4: Delete treatment</b></p> <ol style="list-style-type: none"> <li>1. The system displays treatment deletion confirmation dialog.</li> <li>2. If the doctor clicks yes, the system deletes the treatment record in the database. Else, the systems abort the operation.</li> <li>3. The use case continues to step 2 in basic flow.</li> </ol> <p><b>A5: Search treatment</b></p> <ol style="list-style-type: none"> <li>1. The doctor enters search keyword in the search treatment field.</li> <li>2. The system updates the treatment table according to search input. [E3: Treatment record unavailable]</li> <li>3. The doctor empties the search treatment field.</li> <li>4. The use case continues to step 3 in basic flow.</li> </ol>
<p><b>Exception Flow</b></p>	<p><b>E2: Patient not found</b></p> <ol style="list-style-type: none"> <li>1. The system displays error message that the patient NRIC number not found.</li> <li>2. The doctor re-enter valid patient NRIC number.</li> <li>3. The use case continues to step 4 in Alternative Flow A1.</li> </ol> <p><b>E2: Required field</b></p> <ol style="list-style-type: none"> <li>1. The system displays error message that the input field cannot be empty.</li> <li>2. The doctor input data to empty field.</li> <li>3. The use case continues to step 9 in Alternative Flow A1.</li> </ol> <p><b>E3: Treatment record unavailable</b></p> <ol style="list-style-type: none"> <li>1. The system displays alert message that the treatment record not found.</li> <li>2. The doctor re-enter valid search keyword.</li> </ol>

	3. The use case continues to step 2 in Alternative Flow A5.
<b>Post-Conditions</b>	The treatment record can be managed and the squint type can be classified using Case-Based Reasoning.
<b>Rules</b>	-

### 3.6.2.5 Use Case Description of Manage Treatment Module

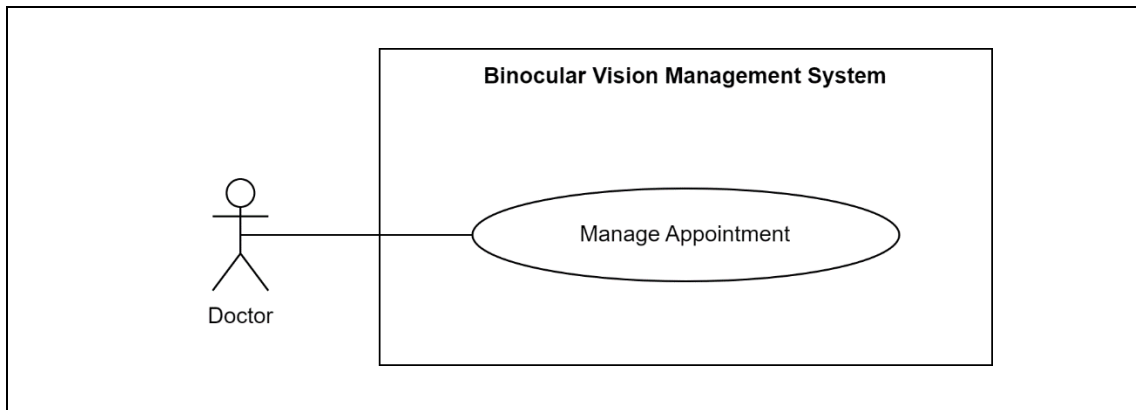


Figure 3.10 Use Case Description Diagram for Manage Treatment

<b>Use Case</b>	Manage Appointment
<b>Brief Description</b>	The Manage Appointment module allows doctors to create, view, edit, and delete appointments for their patients.
<b>Actor</b>	Doctor
<b>Pre-Conditions</b>	<ul style="list-style-type: none"> <li>• The doctor must be logged in to the system.</li> <li>• The doctor must have the necessary permissions to create and manage appointments.</li> </ul>
<b>Basic Flow</b>	<ol style="list-style-type: none"> <li>1. The use case begins when the doctor clicks on the "New Appointment" button on the Appointment page.</li> <li>2. The system prompts the doctor to enter the events subject, patient name, date, and time of the appointment.</li> <li>3. The doctor enters the required information and clicks on "Submit" button.</li> <li>4. The system creates the appointment and displays it on the calendar.</li> <li>5. The doctor can view the details of the appointment by clicking on it.</li> </ol>

	<p>6. The doctor can edit the appointment by clicking on the "Edit" button and making the necessary changes.</p> <p>7. The doctor can delete the appointment by clicking on the "Delete" button.</p>
<b>Alternative Flow</b>	If the doctor wants to change the date or time of an appointment, they can drag and drop the event to the desired date on the calendar.
<b>Exception Flow</b>	<ul style="list-style-type: none"> <li>• If the doctor tries to create an appointment for a date and time that has already passed, the system will prompt an error message.</li> </ul>
<b>Post-Conditions</b>	<ul style="list-style-type: none"> <li>• The appointment is successfully created, viewed, edited, or deleted as per the doctor's action.</li> <li>• The system records all the changes made to the appointment.</li> <li>• The appointment will be visible to the patient and the doctor in the calendar.</li> </ul>
<b>Rules</b>	<ul style="list-style-type: none"> <li>• The doctor can only create, view, edit and delete appointments that they have created.</li> <li>• The doctor can only view and edit the details of the appointment they have created.</li> <li>• The doctor can only delete the appointment they have created.</li> </ul>

### 3.6.3 Activity Diagram

The Activity Diagram of the Binocular Vision Management System depicts the behaviour of the system during different situations. The Figure 3.11 shows the activity diagram for manage login, the Figure 3.12 demonstrates the activity diagram for manage doctor module, Figure 3.13 is about the manage patient module, manage treatment module of activity diagram is shown in the Figure 3.14, and manage appointment is shown in the Figure 3.15.

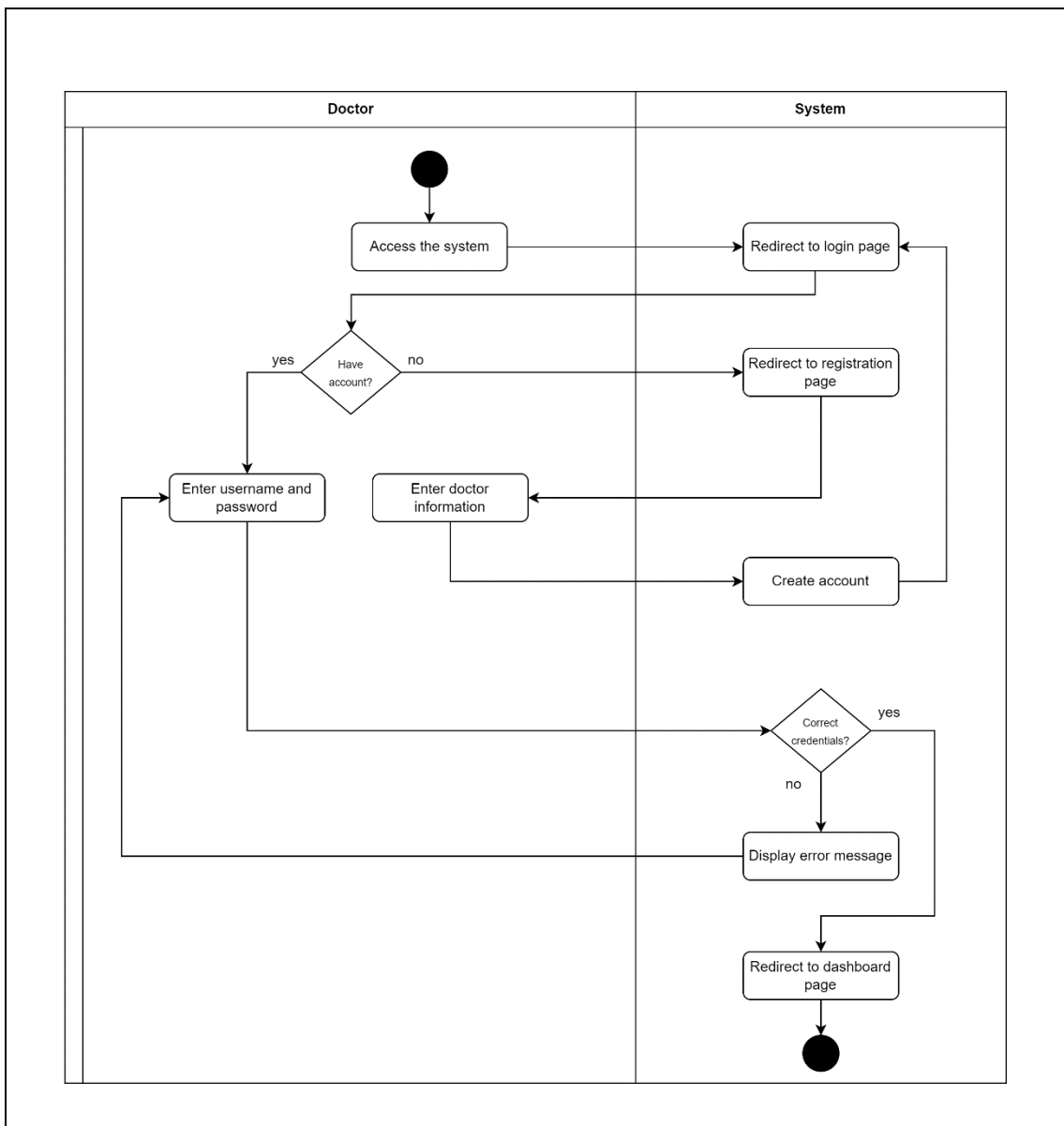


Figure 3.11 Activity Diagram for Manage Login

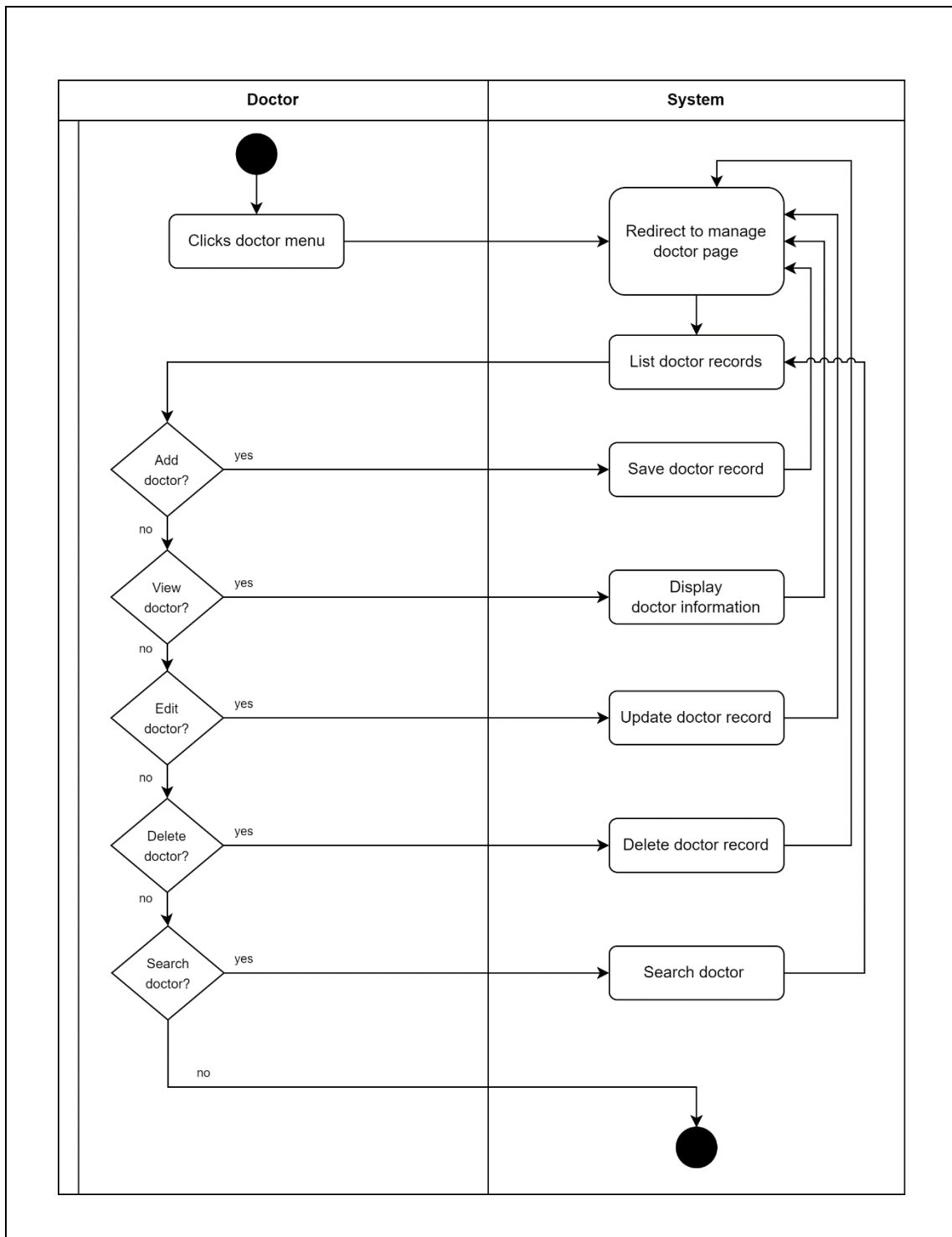


Figure 3.12 Activity Diagram for Manage Doctor

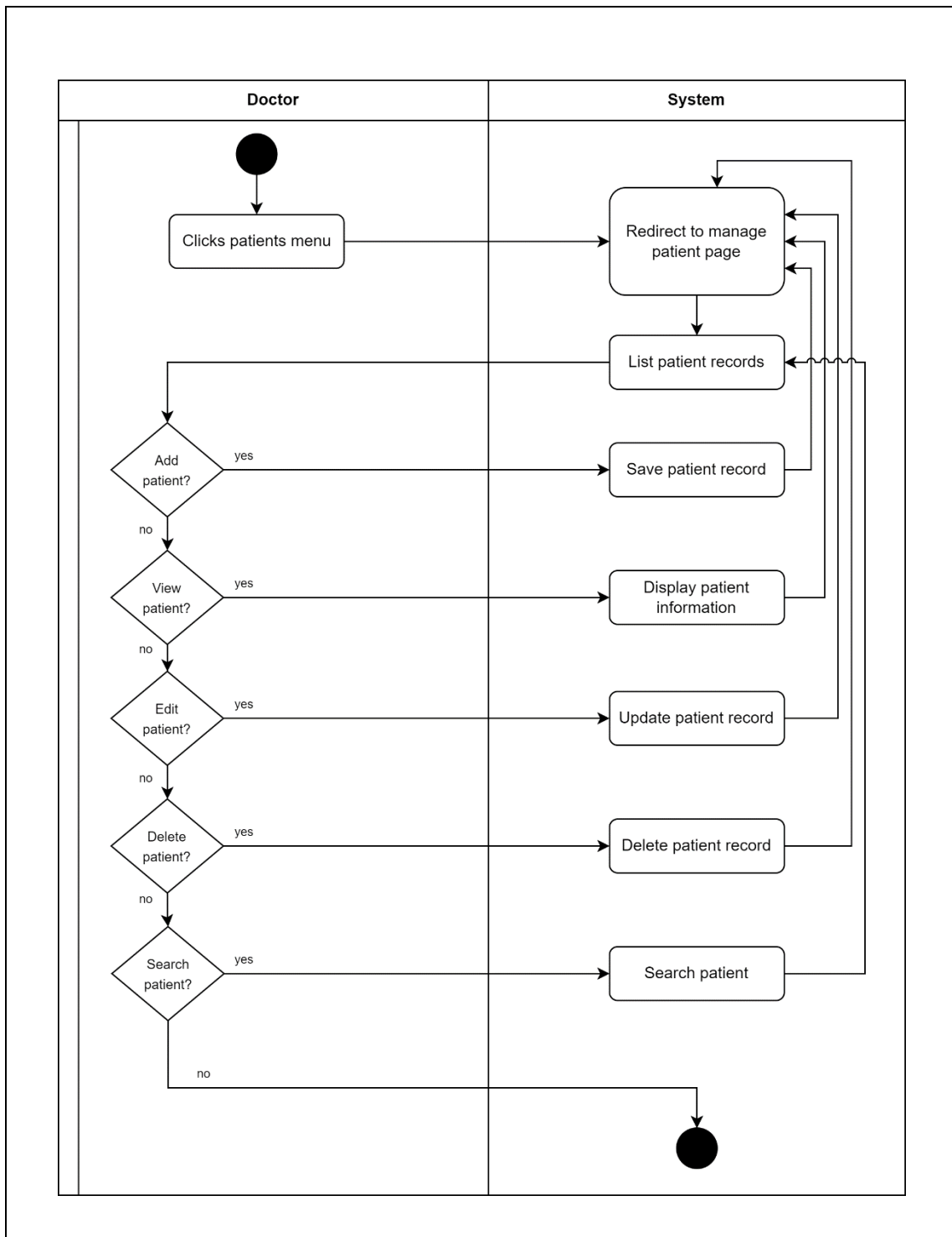


Figure 3.13 Activity Diagram for Manage Patient

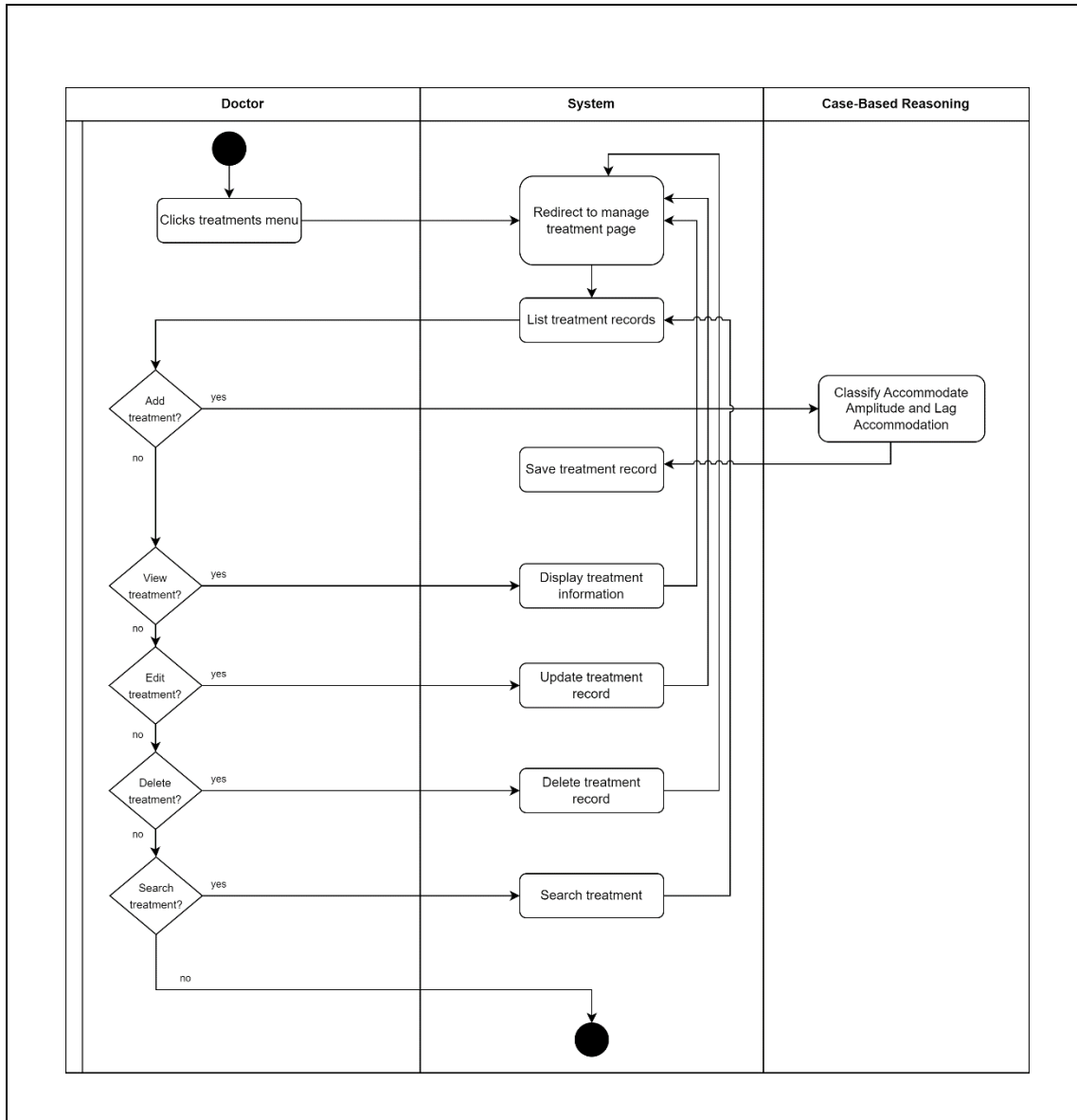


Figure 3.14 Activity Diagram for Manage Treatment



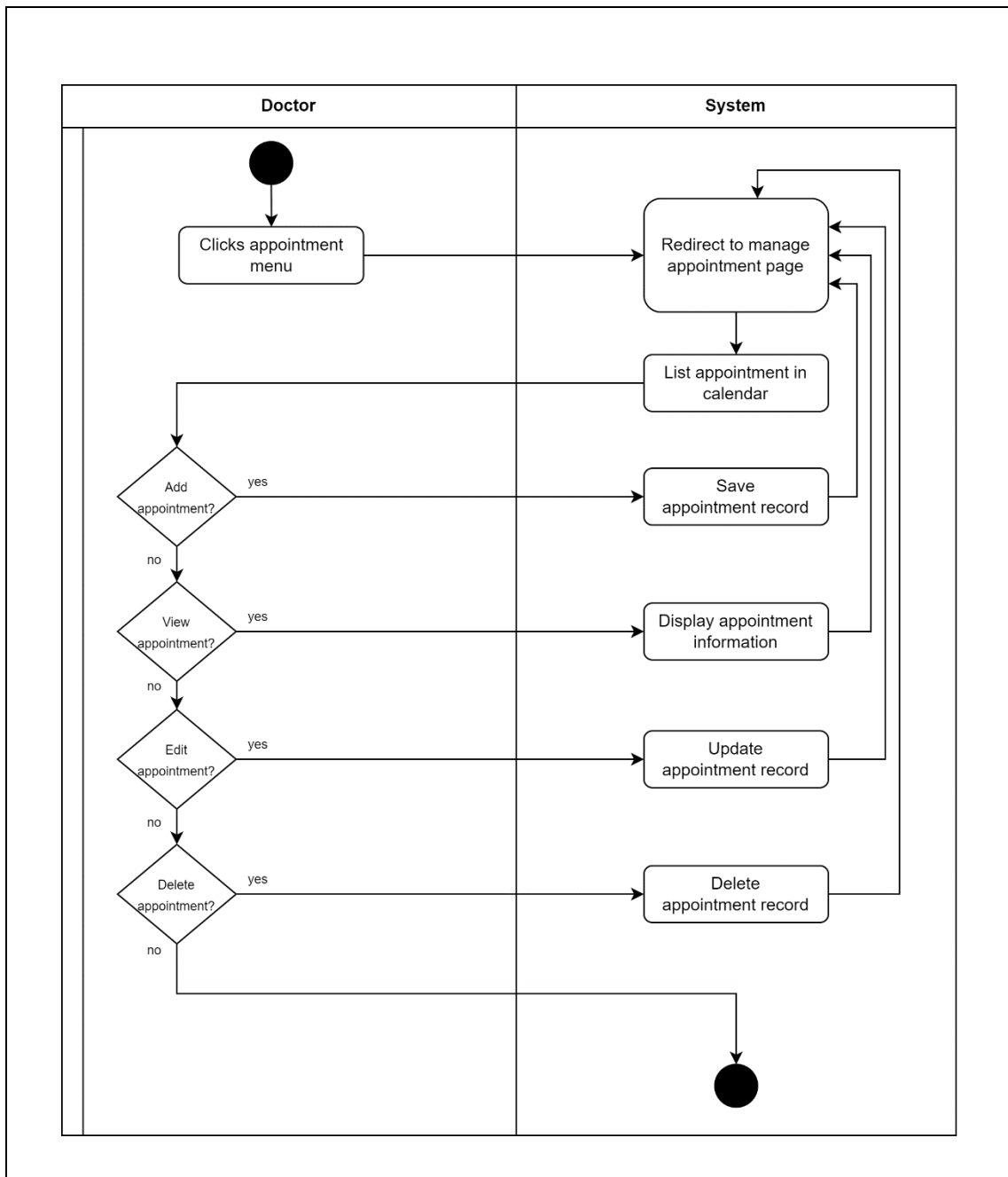


Figure 3.15 Activity Diagram for Manage Appointment

### 3.7 DATA DESIGN

#### 3.7.1 Entity Relationship Diagram

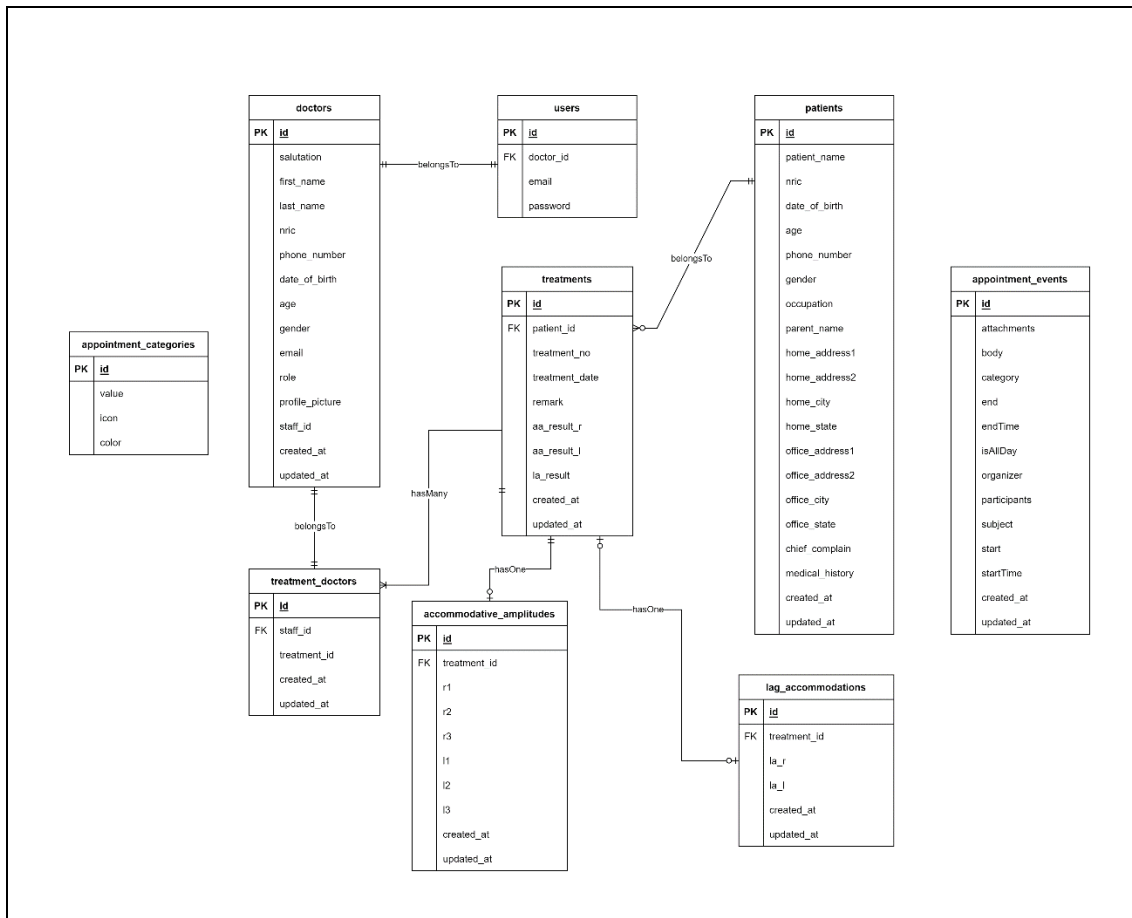


Figure 3.16 Entity Relationship Diagram of the proposed system

### 3.7.2 Data Dictionary

Table 3.4 Data dictionary of table doctors

Field Name	Description	Data Type	Constraint
id	Doctor auto-generated id	CHAR	PK
salutation	Doctor salutation	VARCHAR	NULL
first_name	Doctor first name	VARCHAR	
last_name	Doctor last name	VARCHAR	
nric	Doctor NRIC number	BIGINT	
phone_number	Doctor phone number	BIGINT	
date_of_birth	Doctor date of birth	DATE	
age	Doctor age	INT	
gender	Doctor gender	VARCHAR	
email	Doctor email	VARCHAR	NULL
role	Doctor role (Doctor, student, trainee)	VARCHAR	
profile_picture	Doctor profile picture path	VARCHAR	NULL
staff_id	Doctor staff ID	VARCHAR	UNIQUE
created_at	Doctor record time and date created	TIMESTAMP	NULL
updated_at	Doctor record time and date updated	TIMESTAMP	NULL

Table 3.5 Data dictionary of table users

Field Name	Description	Data Type	Constraint
id	User auto-generated id	CHAR	PK
doctor_id	Doctor_id who owns this account	VARCHAR	
email	Doctor email	VARCHAR	
password	Doctor password	VARCHAR	

Table 3.6 Data dictionary of table patients

Field Name	Description	Data Type	Constraint
id	Patient auto-generated id	CHAR	PK
patient_name	Patient name	VARCHAR	
nric	Patient NRIC number	VARCHAR	
date_of_birth	Patient date of birth	DATE	
age	Patient age	INT	
phone_number	Patient phone number	BIGINT	
gender	Patient gender	VARCHAR	

occupation	Patient occupation (if any)	VARCHAR	NULL
parent_name	Patient parent name or guardian (if patient is minor)	VARCHAR	NULL
home_address1	Patient home address line 1	VARCHAR	NULL
home_address2	Patient home address line 2	VARCHAR	NULL
home_city	Patient home city	VARCHAR	NULL
home_statre	Patient home state	VARCHAR	NULL
office_address1	Patient office address line 1	VARCHAR	NULL
office_address2	Patient office address line 2	VARCHAR	NULL
office_city	Patient office city	VARCHAR	NULL
office_state	Patient office state	VARCHAR	NULL
chief_complain	Patient chiefs complain	VARCHAR	NULL
medical_history	Patient medical history	VARCHAR	NULL
created_at	Patient record time and date created	TIMESTAMP	NULL
updated_at	Patient record time and date updated	TIMESTAMP	NULL

Table 3.7 Data dictionary of table treatments

Field Name	Description	Data Type	Constraint
id	Treatment auto-generated id	CHAR	PK
treatment_no	Treatment auto-generated no	VARCHAR	UNIQUE
patient_id	Patient who receives the treatment	CHAR	
treatment date	Treatment date	DATE	
remark	Treatment remark	VARCHAR	NULL
aa_result_r	Accommodate Amplitude result for right eye	VARCHAR	NULL
aa_result_l	Accommodate Amplitude result for left eye	VARCHAR	NULL
la_result	Lag accommodation result	VARCHAR	NULL
created_at	Treatment record time and date updated	TIMESTAMP	NULL
updated_at	Treatment record time and date updated	TIMESTAMP	NULL

Table 3.8 Data dictionary of table treatment\_doctors

Field Name	Description	Data Type	Constraint
id	Treatment doctors auto-generated id	CHAR	PK

staff_id	Doctor that involves in the treatment	VARCHAR	FK
treatment_id	Treatment id	CHAR	FK
created_at	Treatment doctors record time and date created	TIMESTAMP	NULL
updated_at	Treatment doctors record time and date updated	TIMESTAMP	NULL

Table 3.9 Data dictionary of table accommodative\_amplitudes

Field Name	Description	Data Type	Constraint
id	Accommodation amplitude auto-generated id	CHAR	PK
treatment_id	Treatment id	CHAR	FK
r1	Right eye of AA value (repetition 1)	DOUBLE	NULL
r2	Right eye of AA value (repetition 2)	DOUBLE	NULL
r3	Right eye of AA value (repetition 3)	DOUBLE	NULL
l1	Left eye of AA value (repetition 1)	DOUBLE	NULL
l2	Left eye of AA value (repetition 2)	DOUBLE	NULL
l3	Left eye of AA value (repetition 3)	DOUBLE	NULL
created_at	Accommodate amplitude record time and date created	TIMESTAMP	NULL
updated_at	Accommodate amplitude record time and date updated	TIMESTAMP	NULL

Table 3.10 Data dictionary of table lag\_accommodations

Field Name	Description	Data Type	Constraint
id	Lag accommodation auto-generated id	CHAR	PK
treatment_id	Treatment id	CHAR	FK
la_r	LA right eye	DOUBLE	NULL
la-l	LA left eye	DOUBLE	NULL
created_at	Lag accommodation record time and date created	TIMESTAMP	NULL

updated_at	Lag accommodation record time and date updated	TIMESTAMP	NULL
------------	--	-----------	------

Table 3.11 Data dictionary of table appointment\_events

Field Name	Description	Data Type	Constraint
id	Appointment event auto-generated id	CHAR	PK
attachment	Event attachment	JSON	NULL
body	Event body	LONGTEXT	NULL
category	Event category	VARCHAR	NULL
end	Event end date	DATE	
endTime	Event end time	TIME	NULL
isAllDay	Event is all day	TINYINT	NULL
organizer	Event organizer (doctor)	CHAR	NULL
participants	Event participants (patient)	JSON	NULL
subject	Event subject	LONGTEXT	
start	Event start date	DATE	
startTime	Event start time	TIME	NULL
created_at	Event created time and date	TIMESTAMP	NULL
updated_at	Event updated time and date	TIMESTAMP	NULL

Table 3.12 Data dictionary of table appointment\_categories

Field Name	Description	Data Type	Constraint
id	Appointment category auto-generated id	CHAR	PK
value	Appointment category value	VARCHAR	
icon	Appointment category icon	VARCHAR	NULL
color	Appointment category color	VARCHAR	NULL

### 3.8 CASE-BASED REASONING PROCESSES

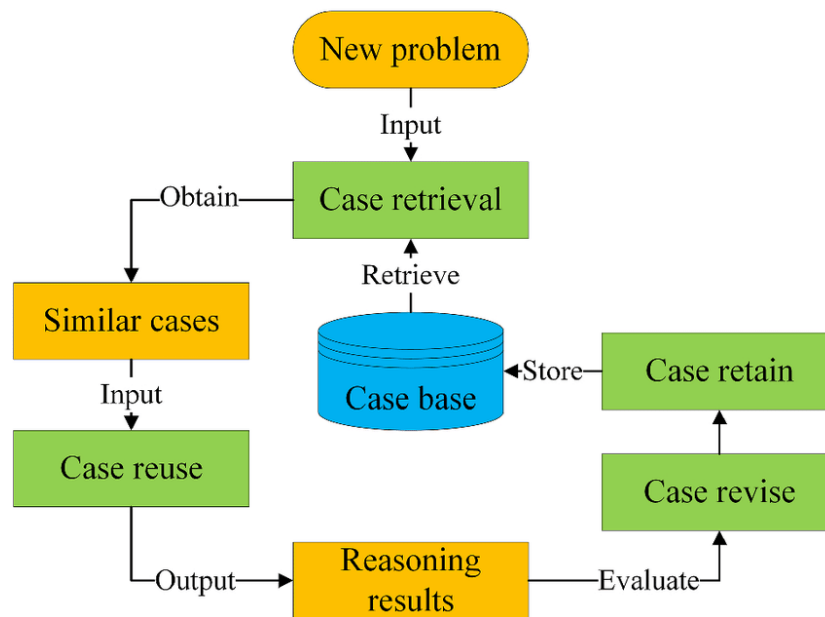


Figure 3.17 Case-Based Reasoning processes

The process of case-based reasoning (CBR) involves several processes, as illustrated in the figure 3.17. The explanation of each process are as follows:

- i) **New Problem:** This is the first step in the CBR process, where the doctor inputs the patient's age, Accommodate Amplitude and Lag of Accommodation test results of a new patient.
- ii) **Case Retrieval:** This step involves retrieving similar cases from the case base, which is a database of previous treatment cases. The system uses the information from the new problem to search for similar cases in the case base. For example, if a new patient has symptoms of strabismus, the system will search for cases in the case base that have similar symptoms.
- iii) **Case Base:** This is the database of previous treatment cases that the system uses for retrieval and storage. The case base is updated with new cases after each treatment. For example, after a patient has been diagnosed and treated

for strabismus, the doctor will store the patient's treatment information in the case base.

- iv) **Similar Cases:** After the system retrieves similar cases from the case base, it will present them to the doctor for evaluation. The CBR model will automatically choose the similar case and make a decision.
- v) **Reasoning Results:** The CBR model uses the information from the selected similar case to produce results for the new patient. The reasoning results are then displayed to the doctor in the web system.
- vi) **Case revise:** After evaluating the reasoning results, the doctor may revise them if necessary. This step allows the doctor to tailor the treatment plan to the new patient. For example, if the diagnosis and treatment plan from the similar case is not appropriate for the new patient, the doctor may revise it to better suit the new patient's needs.
- vii) **Case Retain:** The final step in the CBR process is to store the new case in the case base. This step ensures that the new patient's treatment information is available for future reference and retrieval. For example, after the new patient's treatment is completed, the doctor will store their treatment information in the case base so that it can be used for future reference.



### 3.9 DATASETS

#### 3.9.1 Dataset Collection

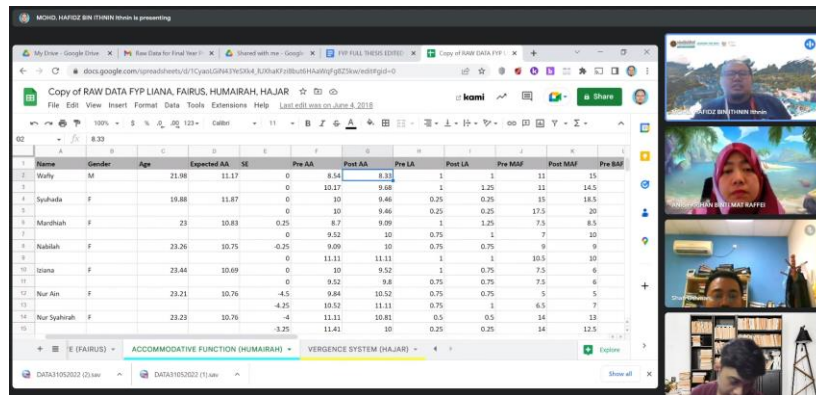


Figure 3.18 Dataset collection

Based on the figure 3,18, the data collection process for the case-based reasoning algorithm involved the gathering of real patient data from the IIUM Binocular Vision clinic, which was conducted through a live meeting with two IIUM doctors. The collected data includes patient information and various diagnostic results such as Accommodate Amplitude, Lag of Accommodation, and others.

#### 3.9.2 Data Pre-Processing

Age	R1	R2	R3	Result	Classification
19	8.25	6.25	5.25	F	Failed
19	9	9	7	F	Failed
19	8.75	7.25	5.75	F	Failed
19	11	10.25	8.25	F	Failed
19	13.25	12.25	11	T	Normal
19	16	15	14.75	T	Normal
19	8.25	7	6	F	Failed
19	13.75	13.5	11.75	T	Normal
19	13.5	12.5	12.25	T	Normal
19	8.75	8.75	6.75	F	Failed
19	10.25	10.25	9	F	Failed
19	10.5	10	9.5	F	Failed
19	14.5	12.5	11.5	T	Normal
19	17.25	15.25	14	T	Normal
19	9.5	7.5	7	F	Failed
19	8	6.75	5.75	F	Failed
19	16	15.25	14	T	Normal
19	8.5	8	6.25	F	Failed
19	17	15.5	15.25	T	Normal

Figure 3.19 Accommodate Amplitude dataset.

In figure 3.19, the pre-processing step involved removing unnecessary features, such as the patient’s name, and other diagnosis features that were not relevant to the

Accommodate Amplitude diagnosis. This was done to reduce the complexity of the data and make it easier to work with. The data are categorized into two classes which are “Normal” and “Failed”. The dataset includes a total of 151 records, with 70 records belonging to the “Normal” class and the remaining records belonging to the “Failed” class. The dataset is comprised of four features, which include patient age and three measurements of Accommodate Amplitude in dioptre format (R1, R2, and R3) for each eye. These four features are used as input for the model, and the algorithm outputs a prediction for the “Result” class. Currently, the dataset is limited to patients with an age range of 19 to 23 years old.

### 3.10 HARDWARE AND SOFTWARE SPECIFICATIONS

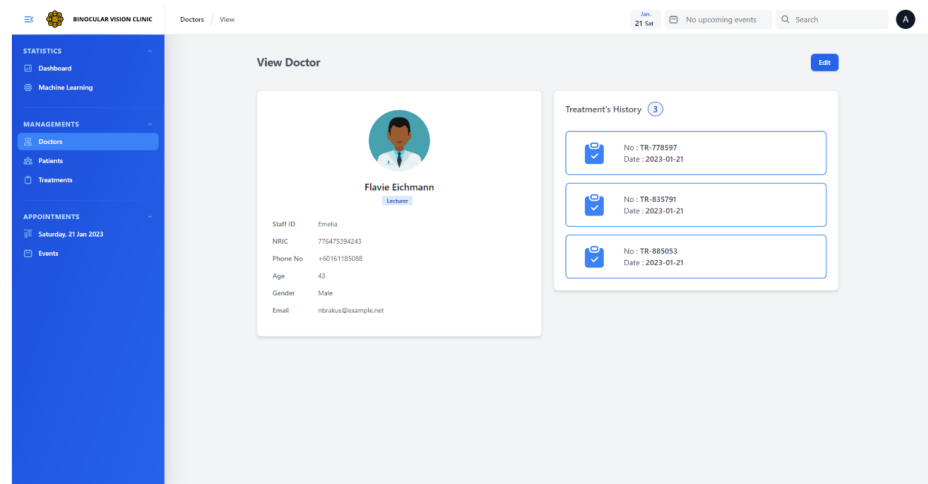
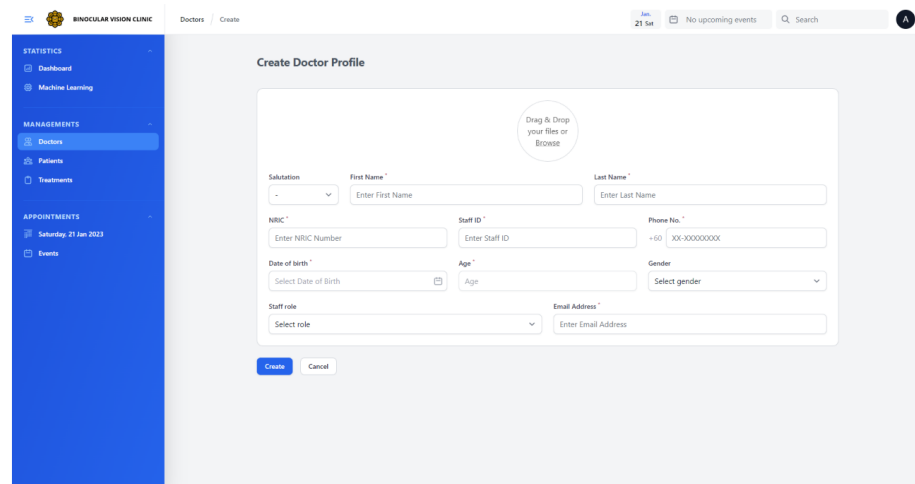
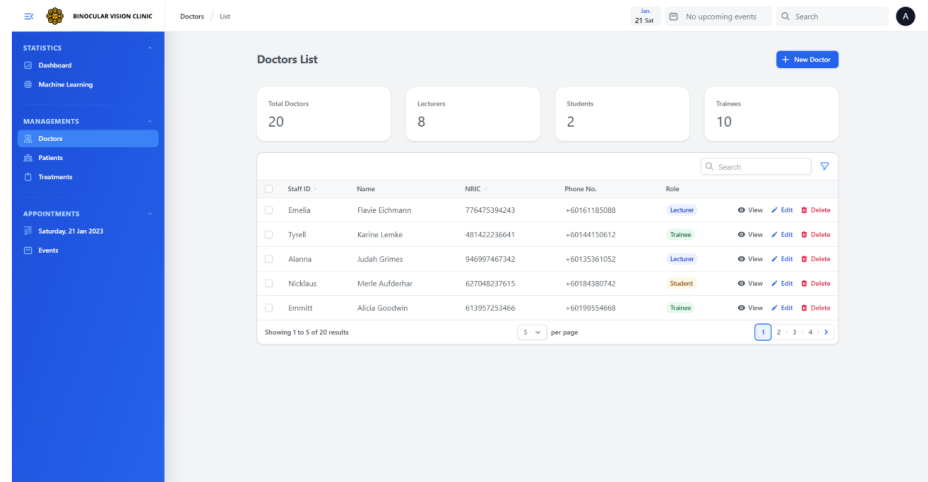
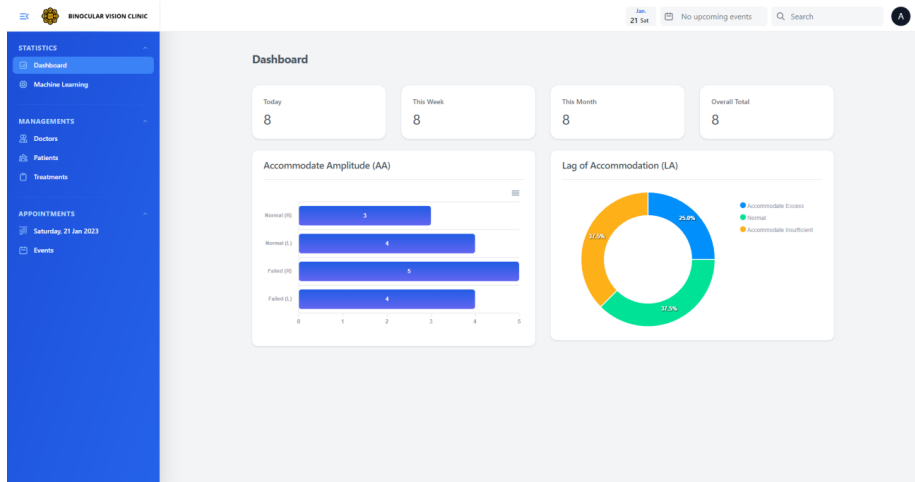
Table 3.13 Hardware specifications

Hardware	Specification	Function
Laptop	AMD Ryzen™ 5 5600H 16GB RAM RTX3060 6GB	To run and develop the Binocular Vision Management System.

Table 3.14 Software specifications

Software	Function
Microsoft Office 365	To write the thesis report.
Draw.IO	To draw UML diagrams.
Microsoft Visual Studio Code	To write code for the web-based component.
Jupyter Notebook	To write code for the machine learning component.
Laragon	To set up local host server.

### 3.11 DESIGN PROTOTYPE



**EDIT DOCTOR PROFILE**

**DOCTORS LIST**

Total Doctors: 20 | Lecturers: 8 | Students: 2 | Trainees: 10

Staff ID	Name	NBIC	Phone No.	Role	View	Edit	Delete
<input type="checkbox"/>	Emelia	Flavie Eichmann	+60161185088	Lecturer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Tyrell	Karime	+60144150612	Trainee	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Alanna	Judith	+60135361052	Lecturer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Nicklaus	Maria	+60184380742	Student	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Ernitt	Alicia Goodwin	61395723466	+6019954668	Trainee	<input type="checkbox"/>	<input type="checkbox"/>

Showing 1 to 5 of 20 results | 5 per page

**EXPORT**

File Name: 21 Jan 23 - 06:20:36pm

Format: XLSX

Staff ID |
  Name |
  NBIC |
  Phone No. |
  Role

**DOCTORS LIST**

Total Doctors: 20 | Lecturers: 8 | Students: 2 | Trainees: 10

Staff ID	Name	NBIC	Phone No.	Role	View	Edit	Delete
<input type="checkbox"/>	Emelia	Flavie Eichmann	776475394243	+60161185088	Lecturer	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Immanuel	Edward Herzog	834778532837	+60128006148	Trainee	<input type="checkbox"/>	<input type="checkbox"/>

Showing 1 to 2 of 2 results | 5 per page

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Patients** Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Patients / List

Jan 21 Sat No upcoming events Search

### Patients List

+ New Patient

Name	NRIC	Age	Gender	Phone No.	Treatments
<input type="checkbox"/> Jordy Mohr I	415396159509	18	Female	+60124527442	1 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Erich Goldner	253301089490	22	Female	+60181573508	3 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Zita Ziemann	980448194237	20	Female	+60113451479	2 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Joelle Kuvalls	25292259359	22	Male	+60173823022	2 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Lonnie Waelchi	869879905449	23	Female	+60167975380	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Annabell Schuster	176763924148	18	Male	+60113228871	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Dr. Jalden Flatley V	717026997898	23	Male	+60145196272	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Luisa Stedermann	62649825632	18	Female	+60147825587	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Reba Reichel	065838133038	20	Female	+60155157579	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Susan Heathcote	343547737973	23	Male	+60142249036	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

Showing 1 to 10 of 50 results 10 per page 1 2 3 4 5

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Patients** Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Patients / Create

Jan 21 Sat No upcoming events Search

### Create Patient Profile

PATIENT Enter patient information SYMPTOM Record medical symptom or complaint

Name  NRIC

Date of birth  Age  Phone No.  Gender

Occupation  Parent Name

Home Address  
Address Line 1  Address Line 2  City  State

Office Address  
Address Line 1  Address Line 2  City  State

[Create](#) [Cancel](#) [Next](#)

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Patients** Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Patients / Create

Jan 21 Sat No upcoming events Search

### Create Patient Profile

PATIENT Enter patient information SYMPTOM Record medical symptom or complaint

Chief complain

Medical history

[Back](#) [Create](#) [Cancel](#)

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Patients** Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Patients / List

Jan 21 Sat No upcoming events Search

### Patients List

+ New Patient

Name	NRIC	Age	Gender	Phone No.	Treatments
<input type="checkbox"/> Amirul Isyraf	990526116573	23	Male	+60145146752	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Jordy Mohr I	415396159509	18	Female	+60124527442	1 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Erich Goldner	253301089490	22	Female	+60181573508	3 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Zita Ziemann	980448194237	20	Female	+60113451479	2 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Joelle Kuvalls	25292259359	22	Male	+60173823022	2 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Lonnie Waelchi	869879905449	23	Female	+60167975380	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Annabell Schuster	176763924148	18	Male	+60113228871	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Dr. Jalden Flatley V	717026997898	23	Male	+60145196272	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Luisa Stedermann	62649825632	18	Female	+60147825587	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
<input type="checkbox"/> Reba Reichel	065838133038	20	Female	+60155157579	0 <a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

Showing 1 to 10 of 51 results 10 per page 1 2 3 4 5 6

**Delete Amirul Isyraf**

Are you sure you would like to do this?

[Cancel](#) [Confirm](#)

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Treatments**

APPOINTMENTS Saturday, 21 Jan 2023 Events

Treatments

+ New Treatment

No.	Patient	Date	Left AA	Right AA	LA	
TR-705929	Joelle Kuvallis	21/01/2023	Failed	Failed	Insufficient	View Delete
TR-885053	Zita Ziemann	21/01/2023	Normal	Failed	Excess	View Delete
TR-666744	Erich Goldner	21/01/2023	Normal	Normal	Normal	View Delete
TR-608301	Zita Ziemann	21/01/2023	Normal	Normal	Insufficient	View Delete
TR-589226	Erich Goldner	21/01/2023	Failed	Failed	Normal	View Delete
TR-835791	Joelle Kuvallis	21/01/2023	Normal	Failed	Excess	View Delete
TR-140091	Erich Goldner	21/01/2023	Failed	Failed	Normal	View Delete
TR-775997	Jordy Mohr 1	21/01/2023	Failed	Normal	Insufficient	View Delete

Showing 1 to 8 of 8 results 10 per page

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Treatments**

APPOINTMENTS Saturday, 21 Jan 2023 Events

Create Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

Treatment No. TR-637995 Patient NRIC Select Patient

Create Cancel Next

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Treatments**

APPOINTMENTS Saturday, 21 Jan 2023 Events

Create Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

Accommodate Amplitude Lag of Accommodation

Left Eye L1 L2 L3  
13 D 12.25 D 12.5 D

Right Eye R1 R2 R3  
8 D 7.5 D 6.75 D

Result Run diagnosis  
Left Eye Normal Right Eye Failed

Back Create Cancel Next

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients **Treatments**

APPOINTMENTS Saturday, 21 Jan 2023 Events

Create Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

Accommodate Amplitude Lag of Accommodation

Input Right eye Left eye  
-0.25 D -0.5 D

Result Run diagnosis  
Accommodative Excess

Back Create Cancel Next

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Treatments Create

21 Sat No upcoming events Search

### Create Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

Persons In Charge

Select Doctors

Role: Lecturer Staff ID: Emilia

Role: Student Staff ID: Nicklaus

Add doctor

Back Next

Create Cancel

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Treatments Create

21 Sat No upcoming events Search

### Create Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

B I U S Heading Subheading

Back

Create Cancel

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Treatments View

21 Sat No upcoming events Search

### View Treatment

PATIENT Choose patient TREATMENT Record medical parameters DOCTOR Select person in charge REMARK Fill in treatment remark

Treatment No.: TR-637995 Patient: NIBC

Next

STATISTICS Dashboard Machine Learning

MANAGEMENTS Doctors Patients Treatments

APPOINTMENTS Saturday, 21 Jan 2023 Events

Treatments List

21 Sat No upcoming events Search

### Treatments

+ New Treatment

No.	Patient	Date	Left AA	Right AA	LA	
TR-437995	Jordy Mohr I	21/01/2023	Normal	Failed	Excess	View Delete
TR-705929	Joelle Kuvallis	21/01/2023	Failed	Failed	Insufficient	View Delete
TR-485053	Zita Ziemann	21/01/2023	Normal	Failed	Excess	View Delete
TR-666744	ERIK			Normal	Normal	View Delete
TR-608301	ZB			Normal	Insufficient	View Delete
TR-589226	ERIK			Failed	Normal	View Delete
TR-815791	Joelle Kuvallis	21/01/2023	Normal	Failed	Excess	View Delete
TR-148091	Erich Goldner	21/01/2023	Failed	Failed	Normal	View Delete
TR-776597	Jordy Mohr I	21/01/2023	Failed	Normal	Insufficient	View Delete

Showing 1 to 9 of 9 results

10 per page

**Delete treatment**

Are you sure you would like to do this?

Cancel Confirm

**BINOULAR VISION CLINIC** Saturday, 21 Jan 2023 21:54 No upcoming events Search

21 Jan << >> + Create event

January 2023

Mo	Tu	We	Th	Fr	Sa	Su
26	27	28	29	30	31	1 Jan
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1 Feb	2	3	4	5

STATISTICS  
Dashboard  
Machine Learning

MANAGEMENTS  
Doctors  
Patients  
Treatments

APPOINTMENTS  
Saturday, 21 Jan 2023  
Events

**BINOULAR VISION CLINIC** Saturday, 21 Jan 2023 21:54

January 2023

Mo	Tu	We	Th	Fr	Sa	Su
26	27	28	29	30	31	
2	3	4	5	6	7	
9	10	11	12	13	14	
16	17	18	19	20	21	
23	24	25	26	27	28	
30	31	1 Feb	2	3	4	

STATISTICS  
Dashboard  
Machine Learning

MANAGEMENTS  
Doctors  
Patients  
Treatments

APPOINTMENTS  
Saturday, 21 Jan 2023  
Events

**Event**

Subject\*

Body  
B I U **Heading** Subheading

Participants  
Select an option

Category  
Select an option

All day

Start\* Jan 21, 2023 19:00

End Jan 21, 2023 19:30

Attachments

Submit Delete Cancel

**BINOULAR VISION CLINIC** Saturday, 21 Jan 2023 21:54

January 2023

Mo	Tu	We	Th	Fr	Sa	Su
26	27	28	29	30	31	
2	3	4	5	6	7	
9	10	11	12	13	14	
16	17	18	19	20	21	
23	24	25	26	27	28	
30	31	1 Feb	2	3	4	

STATISTICS  
Dashboard  
Machine Learning

MANAGEMENTS  
Doctors  
Patients  
Treatments

APPOINTMENTS  
Saturday, 21 Jan 2023  
Events

**Follow up patients**

Wednesday 23 Jan 2023 09:00 - 09:00 (Jan time)  
Review

- Zita Ziemann
- Annabell Schuster
- Lionie Waidich

**BINOULAR VISION CLINIC** Saturday, 21 Jan 2023 21:54 No upcoming events Search

Events List

STATISTICS  
Dashboard  
Machine Learning

MANAGEMENTS  
Doctors  
Patients  
Treatments

APPOINTMENTS  
Saturday, 21 Jan 2023  
Events

**Events** New Event

<input type="checkbox"/>	Subject	Body	Start	End	Category	
<input type="checkbox"/>	Check up patients	<p>Assess strabismus</p>	Jan 23, 2023 00:00:00	Jan 23, 2023 23:59:00	General check up	Edit
<input type="checkbox"/>	Test ML		Jan 26, 2023 00:00:00	Jan 26, 2023 23:59:00	Others	Edit
<input type="checkbox"/>	Follow up patients		Jan 25, 2023 00:00:00	Jan 25, 2023 23:59:00	Follow up	Edit

Showing 1 to 3 of 3 results 10 per page



### 3.12 TESTING METHOD

#### 3.12.1 Functional Testing

Functional testing is a type of testing that verifies that each function of the software application operates in conformance with the requirement specifications. In the context of this project, functional testing would involve testing the various functions of the binocular vision management system to ensure that they work as intended. For example, one functional test for the “Manage Appointment” module could include testing the ability to create a new event, ensuring that the event is correctly displayed in the calendar, and that it can be correctly edited or deleted as needed. Overall, functional testing would involve testing all of the various functions of the system to ensure that they work correctly, and that the system is able to correctly classify the strabismus condition.

#### 3.12.2 User Acceptance Test Plan

To analyse the functionality of the Binocular Vision Management System, a user acceptance test is created as part of the testing plan. This is to ensure that the system’s operation delivers the expected result. The test case will also uncover system flaws so that the developer can resolve them (Andre Scherr, Elberzhager, & Holl, 2018). Table 3.15 below shows the test cases to be conducted for the proposed system. During the UAT phase, the system would be tested by the actual users (IIUM clinicians) to ensure that it meets their needs and expectations, and any issues or bugs would be identified and reported back to the development team for resolution.

Table 3.15 Test cases of the proposed system

No.	Module	Process	Result	Remark
1	Login	Doctor able to login with correct login credentials	Pass / Fail	
2		Doctor is shown error message if incorrect username or password has been entered.	Pass / Fail	
3		Doctor able to logout from the system	Pass / Fail	

4	Dashboard	Doctor able to see the treatments made by today, this week, this month, and overall	Pass / Fail	
5		Doctor able to view the treatment statistics	Pass / Fail	
6	Manage Doctor	Doctor able to view the registered doctors in the table	Pass / Fail	
7		Doctor able to search doctors by entering search keyword	Pass / Fail	
8		Doctor able to add new doctors by filling in doctor personal information.	Pass / Fail	
9		Doctor able to view doctor's information and their treatment history.	Pass / Fail	
10		Doctor able to edit the doctor's record.	Pass / Fail	
11		Doctor able to delete the doctor record after clicking "yes" in the confirmation dialog.	Pass / Fail	
12		Doctor able to export doctors' records to excel, csv and pdf format.	Pass / Fail	
13	Manage Patient	Doctor able to view the registered patients in the table	Pass / Fail	
14		Doctor able to search patients by entering search keyword	Pass / Fail	
15		Doctor able to add new patients by filling in patient information.	Pass / Fail	
16		Doctor able to view patient information and their treatment history.	Pass / Fail	
17		Doctor able to edit the patient record.	Pass / Fail	
18		Doctor able to delete the patient after clicking "yes" in the confirmation dialog.	Pass / Fail	
19		Doctor able to export patient records to excel, csv and pdf format.	Pass / Fail	
20	Manage Treatment	Doctor able to choose patient for the treatment.	Pass / Fail	

21		Doctor able to classify the accommodate amplitude diagnosis for each eye automatically through Case-Based Reasoning algorithm.	Pass / Fail	
22		Doctor able to classify the lag of accommodation diagnosis for both eyes automatically through Case-Based Reasoning algorithm.	Pass / Fail	
23		Doctor able to assign person in charge according to their role in the treatment.	Pass / Fail	
24		Doctor able to add new treatments.	Pass / Fail	
25		Doctor able to view all registered treatments and its result in table.		
26		Doctor able to view treatment information.	Pass / Fail	
27		Doctor able to edit the treatment	Pass / Fail	
28		Doctor able to delete treatment's record after clicking "yes" in the confirmation dialog.	Pass / Fail	
29		Doctor able to export treatment records to excel, csv and pdf format.	Pass / Fail	
30	Manage Appointment	Doctor able to make appointment by creating new event.	Pass / Fail	
31		Doctor able to drag and drop event to different date.	Pass / Fail	
32		Doctor able to view event details.	Pass / Fail	
33		Doctor able to edit event as needed.	Pass / Fail	
34		Doctor able to delete event after clicking the “yes” button in the confirmation dialog.	Pass / Fail	

### **3.13 POTENTIAL USE OF PROPOSED SOLUTION**

Despite technological advancements, machine learning is still not widely used in medical applications, particularly in the field of ophthalmology in detecting type of eye diseases. Therefore, with a computerised system, the proposed solution will be able to assist eye doctors in easing and simplifying their strabismus disease diagnosis process, as well as improving patient and treatment management tasks.

Furthermore, it also provides a platform for the doctors to digitalize their patient and treatment records. By using the proposed system, doctors will be able to reduce their use of paper for patient file management while also saving time by not having to review each patient file individually every time they visit the clinic. Moreover, because the classification process is automated, the doctor's workload can be lessened, allowing him or her to focus on more important tasks.

### 3.14 GANTT CHART

The Gantt Chart that is based on Agile methodology for the Binocular Vision Management System is shown in the Figure 3.20 below.

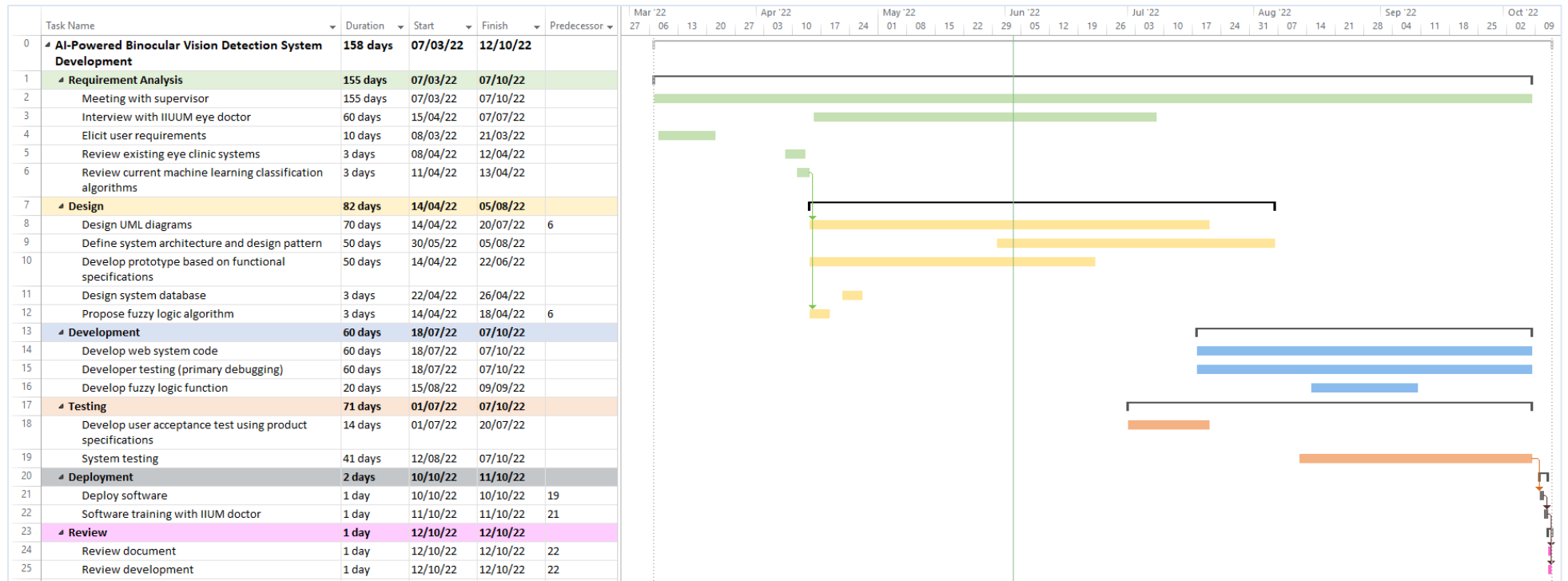


Figure 3.20 Gantt chart for system development

## CHAPTER 4

### RESULT AND DISCUSSION

#### 4.1 INTRODUCTION

This chapter explains deeper into the implementation and development of the strabismus classification system described in the previous chapter. It will provide a detailed description of the system's interfaces and their functions, including figures and explanations based on the functionalities outlined. The goal is to give a clear understanding of both Binocular Vision Management system and Case-Based Reasoning are designed and operated.

#### 4.2 DEVELOPMENT ENVIRONMENT



Figure 4.1 Development Environments

This system is composed of two main components which are the machine learning component and the web-based component. The machine learning component is responsible for classifying strabismus based on diagnostic test results such as accommodative amplitude and lag of accommodation. Based on figure 4.1, the web component, built using the Laravel framework, provides functionalities such as doctor

registration and login, patient and treatment management, and interfaces for training and testing the machine learning model.

In the machine learning component, Python and the Pandas library are used to train the model. Specifically, Panda's library is used to manipulate and analyse large sets of accommodate amplitude and lag of accommodation data where it allows to clean and pre-process the data and make the data ready for training which is important for the system to make accurate predictions.

Finally, Python is the programming language used for implementing the machine learning functions in the system. It is a powerful and versatile language that is well-suited for implementing machine learning algorithms. Overall, the combination of Laravel for the web component and Python for the machine learning component enables the system to effectively classify strabismus and improve patient and treatment management.

### 4.3 BINOCULAR VISION MANAGEMENT SYSTEM IMPLEMENTATION

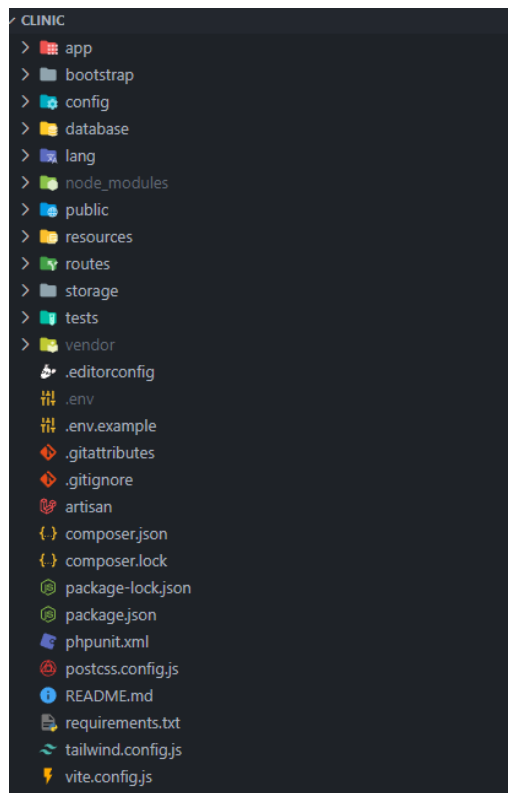


Figure 4.2 Project's structure in Visual Studio code

The structure of this project in Visual Studio Code is shown in the figure 4.2. As can be seen, the project is separated into several packages, each serving a particular function. The system's fundamental functionality, including the controllers, models, and views needed to manage user input and provide the necessary data, may be found under the "app" folder. Static assets, like photos and CSS stylesheets, are kept in the "public" folder and are utilised to improve the user interface. Files that are used to train and test the machine learning algorithm also can be found in the "public" folder. Unit tests are also included in the "tests" folder to make sure the system is operating properly.

```
public static function table(Table $table): Table
{
    return $table
        ->columns([
            Tables\Columns\TextColumn::make('staff_id')->wrap()-
            >sortable()->searchable()->label('Staff ID'),

            Tables\Columns\TextColumn::make('name')->wrap()-
            >sortable()->searchable()->label('Name')
            ->getStateUsing(function(Doctor $record) {
                $name = $record->salutation . " " . $record-
            >first_name . " " . $record->last_name;
                return $name;
            })),

            Tables\Columns\TextColumn::make('nric')->sortable()-
            >searchable()->label('NRIC'),
            Tables\Columns\TextColumn::make('phone_number')-
            >label('Phone No.')->prefix('+60'),
            Tables\Columns\TextColumn::make('role')->sortable()-
            >label("Role"),

            Tables\Columns\BadgeColumn::make('role')
                ->colors([
                    'primary' => 'Lecturer',
                    'warning' => 'Student',
                    'success' => 'Trainee',
                ]),

        ])->defaultSort('updated_at', 'desc')
        ->actions([
            Tables\Actions\ViewAction::make()->label('View'),
            Tables\Actions>EditAction::make()->label('Edit'),
```



```

        Tables\Actions\DeleteAction::make()->label('Delete'),
    ])
    ->bulkActions([
        FilamentExportBulkAction::make('export')-
>disablePreview()->timeFormat('d M y - h:i:sa'),
        Tables\Actions\DeleteBulkAction::make(),
    ]);
}

```

Figure 4.3 Doctor's table resource snippet code

In figure 4.3, his code snippet is defining a doctor table structure in the project. It specifies the columns that the table should have, such as “staff\_id”, “name”, “nric”, “phone\_number”, and “role”. The column “name” is also customised to be a combination of the “salutation”, “first\_name” and “last\_name” fields from the Doctor model. It also specifies sorting, searching, and labelling for each column. Additionally, the table has actions such as “View”, “Edit” and “Delete” for each record, as well as bulk actions for the whole table like “export” and “Delete All” which can be done using FilamentExportBulkAction and DeleteBulkAction classes respectively.

```

public static function form(Form $form): Form
{
    return $form
    ->schema([

        Card::make()
        ->schema([
            Forms\Components\FileUpload::make('profile_picture')
                ->label('Profile Picture')
                ->image()
                ->avatar()->columnSpanFull(),

            Forms\Components\Grid::make()
            ->schema([
                Forms\Components\Select::make('salutation')
                ->options([
                    "Dr." => "Dr.",
                    "Mr." => "Mr.",
                    "Ms." => "Ms.",
                    "Prof." => "Prof.",
                ])
                ->default('')->placeholder('-'),
            ])
        ])
    ])
}

```

```

Forms\Components\TextInput::make('first_name')-
>required()->maxLength(50)->label('First Name')->placeholder('Enter First
Name')->columnSpan(3)->disableAutocomplete(),

Forms\Components\TextInput::make('last_name')-
>required()->maxLength(50)->label('Last Name')->placeholder('Enter Last
Name')->columnSpan(3)->disableAutocomplete(),
])->columns(7),

Forms\Components\Grid::make()
->schema([
Forms\Components\TextInput::make('nric')
->label(__('NRIC'))->numeric()-
>unique(ignoreRecord: true)->required()->placeholder('Enter NRIC Number')
->mask(fn
(Forms\Components\TextInput\Mask $mask) => $mask->pattern('000000-00-
0000'))->disableAutocomplete(),

Forms\Components\TextInput::make('staff_id')-
>label(__('Staff ID'))->placeholder('Enter Staff ID')->required()-
>unique(ignorable: fn ($record) => $record)->disableAutocomplete(),

Forms\Components\TextInput::make('phone_number')-
>placeholder('XX-XXXXXXXX')->label('Phone No.')->numeric()-
>maxLength(11)->prefix('+60')->required()->disableAutocomplete()
->mask(fn (Forms\Components\TextInput\Mask $mask)
=> $mask->pattern('00-00000000')),
])->columns(3),

Forms\Components\Grid::make()
->schema([

Forms\Components\DatePicker::make('date_of_birth')-
>placeholder('Select Date of Birth')->displayFormat('d/m/Y')->required()-
>reactive()

->afterStateUpdated(function
(callable $set, callable $get){
$age =
Carbon::parse($get('date_of_birth'))->age;
$set('age', $age);
}),

Forms\Components\TextInput::make('age')-
>placeholder('Age')->disabled()->required(),

```

```

        Forms\Components\Select::make('gender')-
>placeholder('Select gender')
    ->options([
        'Male' => 'Male',
        'Female' => 'Female',
    ]),

    ])->columns(3),

        Forms\Components\Select::make('role')->label('Staff
role')->placeholder('Select role')
    ->options([
        'Lecturer' => 'Lecturer',
        'Student' => 'Student',
        'Trainee' => 'Trainee',
    ]),

        Forms\Components\TextInput::make('email')->required()-
>email()->label('Email Address')->placeholder('Enter Email Address')-
>disableAutocomplete(),
    ])
    ->columns(2)
]);
}

```

Figure 4.4 Doctor's form resource snippet code

Figure 4.4, this code defines a form for creating and editing doctor profiles in the system. It includes fields for the staff's name, NRIC, staff ID, phone number, date of birth, age, gender, role, email and profile picture. The form uses a number of components such as text input, select, date picker, file upload, grid and card to build the layout of the form. The form also includes validation and constraints such as unique, required, email format and max length for each field. Additionally, there are also some features added such as masking for phone number, prefix for phone number, avatar for profile picture and disable autocomplete for some fields.

```

public static function table(Table $table): Table
{
    return $table
        ->columns([
            Tables\Columns\TextColumn::make('patient_name')-
>sortable()->searchable()->label('Name'),
            Tables\Columns\TextColumn::make('nric')->sortable()-
>searchable()->label('NRIC'),

            Tables\Columns\TextColumn::make('age')
            ->getStateUsing(function(Patient $record) {
                return Carbon::parse($record->date_of_birth)->age;;
            }),
            Tables\Columns\TextColumn::make('gender')->sortable(),
            Tables\Columns\TextColumn::make('phone_number')-
>label('Phone No.')->prefix('+60'),
            Tables\Columns\TextColumn::make('treatment')-
>label('Treatments')
            ->getStateUsing(function (Patient $record){
                $patient_id = $record->id;
                $treatment_count = DB::table('treatments')
                    ->where('patient_id', $patient_id)
                    ->count();
                return $treatment_count;
            })
        ])->defaultSort('updated_at', 'desc')
        ->filters([
            //
        ])
        ->actions([

            Tables\Actions\ViewAction::make()->label('View'),
            Tables\Actions\EditAction::make()->label('Edit'),
            Tables\Actions\DeleteAction::make()->label('Delete'),

        ])
        ->bulkActions([
            FilamentExportBulkAction::make('export')-
>disablePreview()->timeFormat('d M y - h:i:sa'),
            Tables\Actions\DeleteBulkAction::make(),
        ]);
}

```

Figure 4.5 Patient's table resource snippet code

In figure 4.5, this code creates a table in a web application, displaying patient information such as name, NRIC, age, gender, phone number, and treatment count. The

table columns can be sorted and searched by name, NRIC, and updated at time. The table also includes action buttons for viewing, editing, and deleting a patient's information, and bulk actions for exporting and deleting multiple patients at once. Additionally, the age column is calculated by getting the age of the patient based on their date of birth. The treatment column is calculated by counting the number of treatments for each patient based on the patient\_id.

```
public static function form(Form $form): Form
{
    return $form
        ->schema([

            Forms\Components\Wizard::make([
                Forms\Components\Wizard\Step::make('Patient')->
                >icon('heroicon-o-user')
                ->description("Enter patient information")
                ->schema([
                    Forms\Components\Card::make()
                    ->schema([

                        Forms\Components\Grid::make()
                        ->schema([
                            Forms\Components\TextInput::make('patient_name')->required()->label('Name')->placeholder('Enter patient name'),

                            Forms\Components\TextInput::make('nric')
                                ->label(__('NRIC'))->numeric()->unique(ignoreRecord: true)->required()
                                ->mask(fn (Forms\Components\TextInput\Mask $mask) => $mask->pattern('000000-00-0000'))
                                ->placeholder('Enter patient NRIC number')

                        ],->columns(2),

                        Forms\Components\Grid::make()
                        ->schema([
                            Forms\Components\DatePicker::make('date_of_birth')->placeholder('Select date of birth')->displayFormat('d/m/Y')->required()->reactive()
```

```

        ->afterStateUpdated(function
(callable $set, callable $get){
            $age =
Carbon::parse($get('date_of_birth'))->age;
            $set('age', $age);
        }
    ),
    Forms\Components\TextInput::make('age')->placeholder('Patient age')->disabled()->required(),
    Forms\Components\TextInput::make('phone_number')->label('Phone No.')->placeholder('XX-XXXXXXX')->numeric()->maxLength(11)->prefix('+60')->required()
        ->mask(fn
(Forms\Components\TextInput\Mask $mask) => $mask->pattern('00-00000000')),
    Forms\Components\Select::make('gender')->placeholder('Select gender')->required()
        ->options([
            'Male' => 'Male',
            'Female' => 'Female',
        ]),
    ]->columns(4),

    Forms\Components\Grid::make()
        ->schema([
            Forms\Components\TextInput::make('occupation')->placeholder('Enter patient occupation'),
            Forms\Components\TextInput::make('parent_name')->hint('Required only if patient is minor')->label('Parent Name')->placeholder('Enter patient parent name'),
        ]->columns(2),

    Forms\Components\Section::make("Home Address")
        ->compact()
        ->schema([
            Forms\Components\TextInput::make('home_address1')->label("Address Line 1")->placeholder("Address Line 1"),
            Forms\Components\TextInput::make('home_address2')->label("Address Line 2")->placeholder("Address Line 2"),
            Forms\Components\TextInput::make('home_city')->label('City')->placeholder("City"),
            Forms\Components\Select::make('home_state')->placeholder('Select state')->label('State')
        ]->columns(4),

```

```

        Forms\Components\Section::make("Office
Address")
        ->compact()
        ->schema([
            Forms\Components\TextInput::make('office_address1')->label("Address Line 1")->placeholder("Address Line 1"),
            Forms\Components\TextInput::make('office_address2')->label("Address Line 2")->placeholder("Address Line 2"),
            Forms\Components\TextInput::make('office_city')->label('City')->placeholder("City"),
            Forms\Components\Select::make('office_state')->placeholder('Select state')->label('State')
        ]->columns(4),
    ])
]),
    Forms\Components\Wizard\Step::make('Symptom')->
    >icon('heroicon-o-clipboard-list')
    ->description("Record medical symptom or
complaint")
    ->schema([
        Forms\Components\Card::make()
        ->schema([
            Forms\Components\RichEditor::make('chief_complain'),
            Forms\Components\RichEditor::make('medical_history')
        ])
    ]),
]),
    ]->columns(1);
}

```

Figure 4.6 Patient's form resource snippet code

Based on figure 4.6, this code defines a form for creating a new patient in the system, using the Laravel framework. It contains fields for the patient's name, NRIC number, date of birth, age, phone number, gender, occupation, and parent's name. It also includes sections for the patient's home and office addresses, including fields for address lines, city, and state. It also includes a wizard for guiding the user through the form with different steps, as well as some validation for required fields and unique NRIC number

for the patient. The form also contains some additional functionality such as masking for phone numbers and NRIC number and age calculation based on date of birth.

```
public function aa_classify(){

    if(!isset($_COOKIE['age']) || !isset($_COOKIE['aa_l1']) ||
!isset($_COOKIE['aa_l2']) || !isset($_COOKIE['aa_l3']) ||
!isset($_COOKIE['aa_r1']) || !isset($_COOKIE['aa_r2']) ||
!isset($_COOKIE['aa_r3'])){
        return "0";
    }else{

        $age = $_COOKIE['age'];
        $l1 = $_COOKIE['aa_l1'];
        $l2 = $_COOKIE['aa_l2'];
        $l3 = $_COOKIE['aa_l3'];
        $r1 = $_COOKIE['aa_r1'];
        $r2 = $_COOKIE['aa_r2'];
        $r3 = $_COOKIE['aa_r3'];

        if($this->validateInput($l1) && $this->validateInput($l2) &&
$this->validateInput($l3) && $this->validateInput($r1) && $this-
>validateInput($r2) && $this->validateInput($r3)){
            $left_cmd =
escapeshellcmd("cbr\aa\classification\aa_classify.py $age $l1 $l2 $l3");
            $left = shell_exec($left_cmd);

            $right_cmd =
escapeshellcmd("cbr\aa\classification\aa_classify.py $age $r1 $r2 $r3");
            $right = shell_exec($right_cmd);

            $decodeRight = urldecode($right);
            $rightResult = substr($decodeRight, 0, 1);

            $decodeLeft = urldecode($left);
            $leftResult = substr($decodeLeft, 0, 1);

            setcookie("aa_result_r", $rightResult, time() + 3600);
            setcookie("aa_result_l", $leftResult, time() + 3600);

            return $left . $right;

        }else{
            return "1";
        }
    }
}
```

Figure 4.7 AA Classification function snippet code.



In figure 4.7, this function classifies a patient's Accommodate Amplitude (AA) diagnosis based on their age, left and right AA measurement values. The function first checks whether the necessary input values are present in cookies, if not it returns “0”. Next, it validates the input values using the validateInput function, if the inputs are invalid it returns “1”. If input values are valid, it calls the python script ‘aa\_classify.py’ and passing it the age, left and right AA measurement values as arguments. The script then returns a string output of the classification. This output is then stored as a cookie and returned to the treatment page.

## **4.4 CASE-BASED REASONING IMPLEMENTATION**

### **4.4.1 Introduction**



Figure 4.8 Jupyter Notebook IDE

In figure 4.8, Jupyter Notebook is used in this project as the machine learning component's development environment. It offers visual aids, such as graphs, to help users better understand and interact with the data utilised in the Case-Based Reasoning (CBR) process. It also enables organised and structured Python code writing.

#### 4.4.2 Algorithm Processes

In a case-based reasoning (CBR) system, the “Retrieve” process is responsible for finding the most similar case to the current problem at hand, in order to inform the decision-making process. Based on the figure 4,9, this code is reading in a dataset file named “aa\_dataset.csv” and storing it in a variable called “aa\_data” using the pandas library’s “read\_csv” function. This dataset will be used in the Retrieve step of the CBR process, where relevant past cases are retrieved from the dataset.

```
#STEP 1: RETRIEVE
#read original dataset file
aa_data = pd.read_csv('../aa_dataset.csv')
```

Figure 4.9 Dataset reading code snippet.

Then, in figure 4.10, it is finding the minimum and maximum values for each feature in the dataset, which is stored in the “min\_value” and “max\_value” variables, respectively. The following features are the patient age, R1 (AA reading 1), R2 (AA reading 2), and R3 (AA reading 3) like shown in the figure 4.11. Lastly, the function will calculate the range of values for each feature by subtracting the minimum value from the maximum value and storing the result in the “range\_value” variable. These calculations are done to normalize the data for the similarity calculations in next step.

```
#find min values and max values for each feature
min_value = case_base.min()
max_value = case_base.max()

#calculate range value
range_value = max_value-min_value
```

Figure 4.10 Finding range value of features code snippet.

	Age	R1	R2	R3
0	19	8.25	6.25	5.25
1	19	9.00	9.00	7.00
2	19	8.75	7.25	5.75
3	19	11.00	10.25	8.25
4	19	13.25	12.25	11.00

Figure 4.11 Dataset features sample.

```
#get arguments from laravel
age=sys.argv[1]
r1=sys.argv[2]
r2=sys.argv[3]
r3=sys.argv[4]

train = np.array([age, r1, r2, r3])
```

Figure 4.12 AA measurement arguments code snippet.

By referring to figure 4.12, this code block is retrieving data passed from the web system as command line arguments in the form of age, r1, r2, and r3. It then assigns these values to a variable train which is an array of these values. This data is used to train a model and predict results in the later steps of the CBR process.

After that, based on the figure 4.13, the CBR process continues with the calculation of the local similarity value between new case and case base. The local similarity is calculated as 1 minus the absolute difference between the input case and the case in the case base, divided by the range of that feature. Then it calculates the global similarity which is the overall similarity score for the new case and the case vase in the dataset by taking the sum of all local similarities multiplied by the weightage assigned to each feature. Then, it finds the index of the highest similarity case and return the highest similarity max value.

Finally, after all calculations done, the decision is made by the CBR where it prints the result of the highest similarity case and send the result to the web treatment interface so that the doctor can view it like shown in the figure 4.14.

```

for i in range(case_base.shape[0]):

    total_similarity = 0

    for j in range(case_base.shape[1]):
        local_similarity[i, j] = 1-(abs(train[j] - case_base.iloc[i,j])/range_value.iloc[j])

    total_similarity+=((local_similarity[i,j])*weightage[j])

    global_similarity[i] = round((1/total_weightage) * (total_similarity),4)
    highest_similarity[i] = global_similarity[i]

highest_index=np.argmax(highest_similarity)+1
highest_similarity_max = highest_similarity.max();

```

Figure 4.13 Local similarity, global similarity calculations code snippet.

```

print(aa_data.loc[highest_index-1, "Result"])

```

Figure 4.14 Print highest similarity result code snippet.

Based on the figure 4.15, this code is performing the “retain” process in the Case-Based Reasoning (CBR) system. The retain process is used to update the case base with new information by adding new cases to the dataset. In this specific code, it is reading in values passed in from a controller, creating a dictionary with these values and the additional fields of “result” and “classification”, and then appending this dictionary as a new row in the 'aa\_dataset.csv' file. This allows for the case base to be updated with new information, which can be used to improve the performance of the CBR model.

```

def append_dict_as_row(file_name, dict_of_elem, field_names):
    # Open file in append mode
    with open(file_name, 'a+', newline='') as write_obj:
        # Create a writer object from csv module
        dict_writer = DictWriter(write_obj, fieldnames=field_names)
        # Add dictionary as wor in the csv
        dict_writer.writerow(dict_of_elem)

# get values from controller
age=int(sys.argv[1])
r1=float(sys.argv[2])
r2=float(sys.argv[3])
r3=float(sys.argv[4])
result=sys.argv[5]
if(result=='F'):
    classification="Failed"
else:
    classification="Normal"

field_names = ['Age','R1','R2','R3','Result', 'Classification']
row_dict = {'Age': age, 'R1': r1, 'R2':r2, 'R3':r3, 'Result':result, 'Classification':classification}
# Append a dict as a row in csv file
append_dict_as_row('../aa_dataset.csv', row_dict, field_names)

```

Figure 4.15 CBR retain process code snippet.

## 4.5 SYSTEM OUTPUT

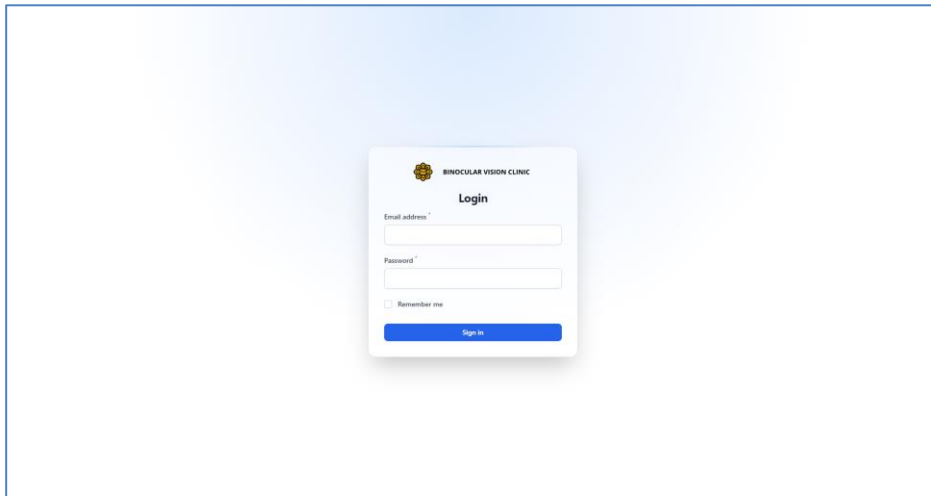


Figure 4.16 Login screen

As seen in figure 4.16, the doctor is presented with a login page where they can enter their username and password to access the system. To make their experience more convenient, the system also includes a “Remember Me” function that allows the doctor to save their credentials for future use, eliminating the need for them to enter their login information every time they access the system. In the event that the provided credentials are incorrect, the system will display an error message, as depicted in figure 4.17, alerting the doctor that the entered username or password does not match any records in the system’s database.

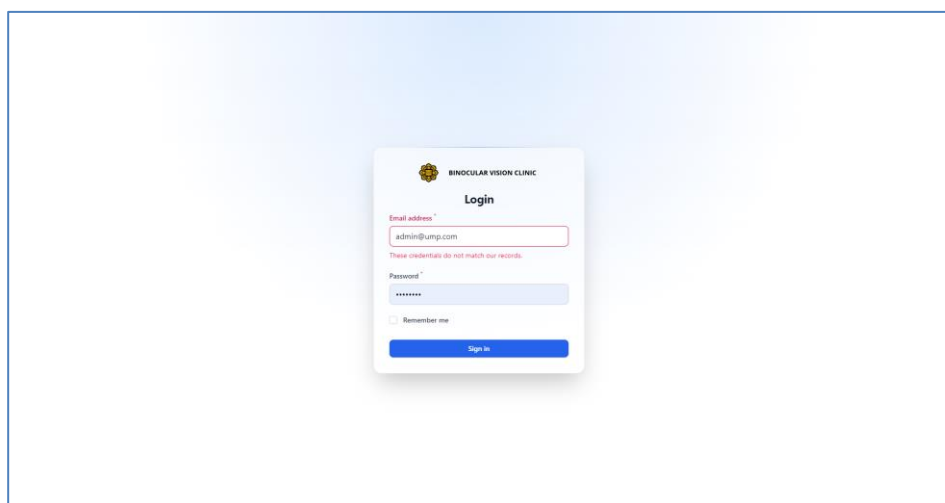


Figure 4.17 Wrong username or password error message

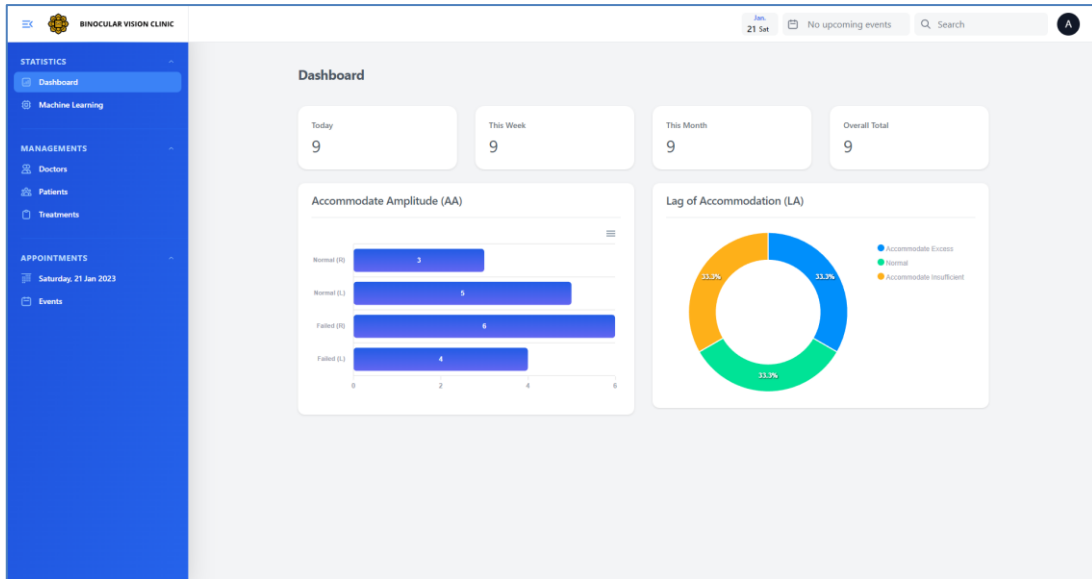


Figure 4.18 Dashboard screen

Upon successful login, the doctor will be directed to the system's dashboard screen, as depicted in figure 4.18. This screen provides a quick overview of the treatments made by the doctor on a daily, weekly, monthly, and overall basis. Additionally, the dashboard includes visual representations of previous treatment results for Accommodate Amplitude and Lag of Accommodation, allowing the doctor to easily track their progress. The doctor also has the option to switch to a dark theme by toggling the option in the user menu, as seen in figure 4.19, for a more comfortable viewing experience.

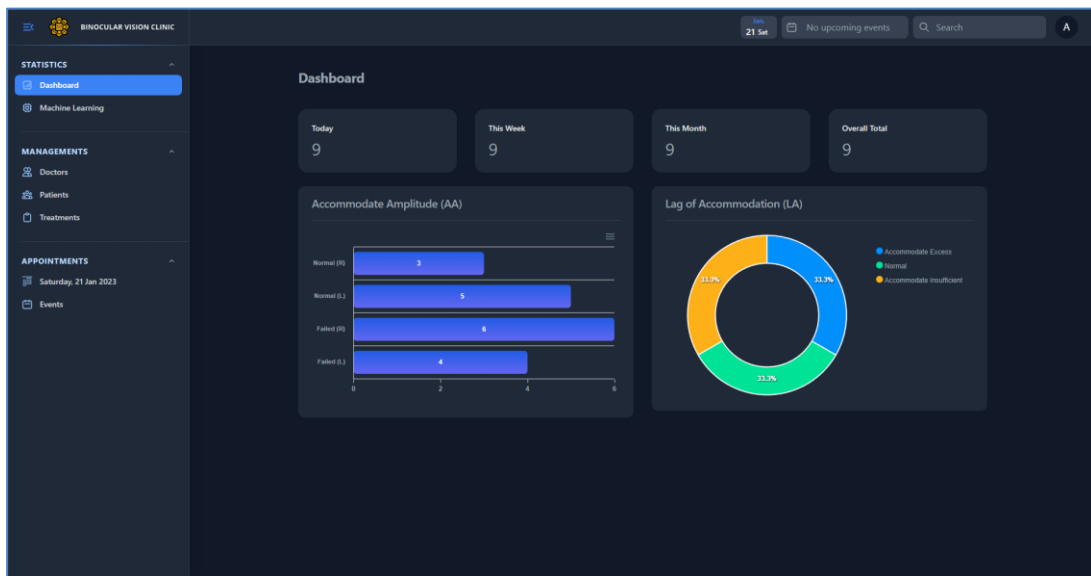


Figure 4.19 Dashboard in dark mode theme

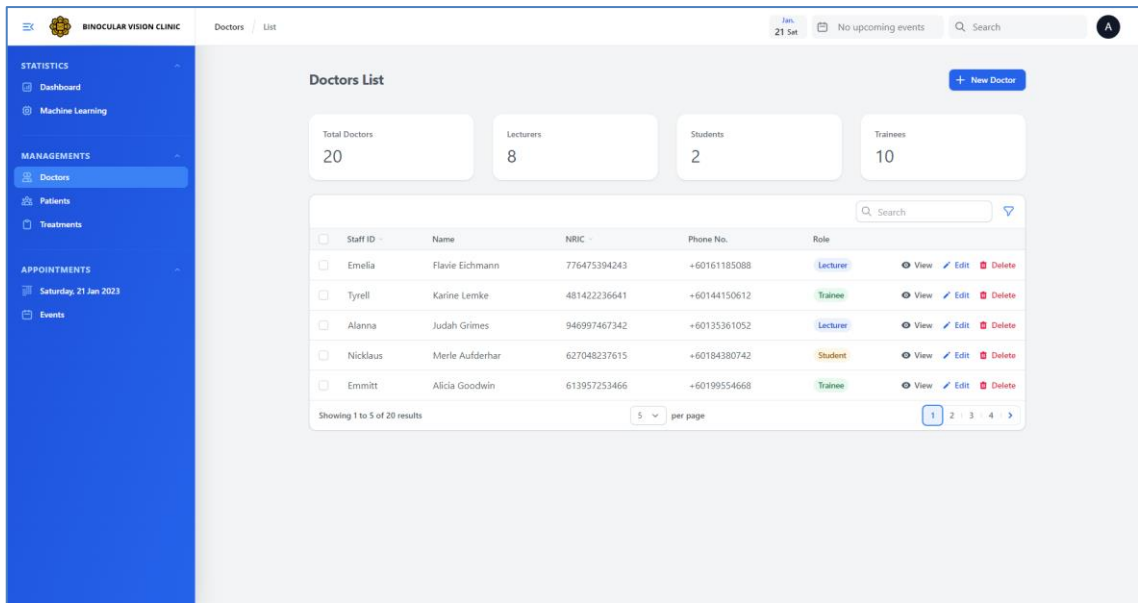


Figure 4.20 Doctor’s list screen

The doctor can access a comprehensive list of all the registered doctors in the system by navigating to the “Doctors” menu in the sidebar, as depicted in figure 4.20. The total number of doctors by role is displayed on the top widgets, providing a quick overview of the available staff. Additionally, from this page, the doctor has the ability to add new doctors, edit existing ones, delete them, or simply view their records. To add a new doctor, the doctor can click on the “New Doctor” button, which will redirect them to the page for creating a new doctor profile, as illustrated in figure 4.21.

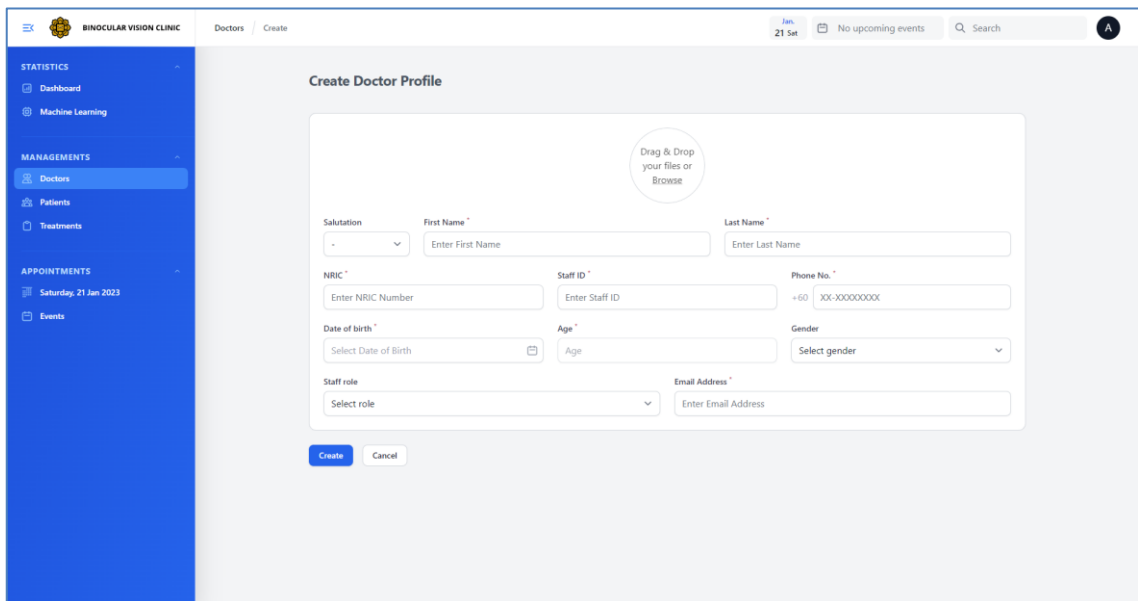


Figure 4.21 Create doctor profile screen



From this, the doctor can create a new doctor record by uploading a profile picture and filling in the salutation, doctor name, nric, staff ID, phone number and other personal details. After completing the form, they can submit it by clicking the “Create” to save the record in the database.

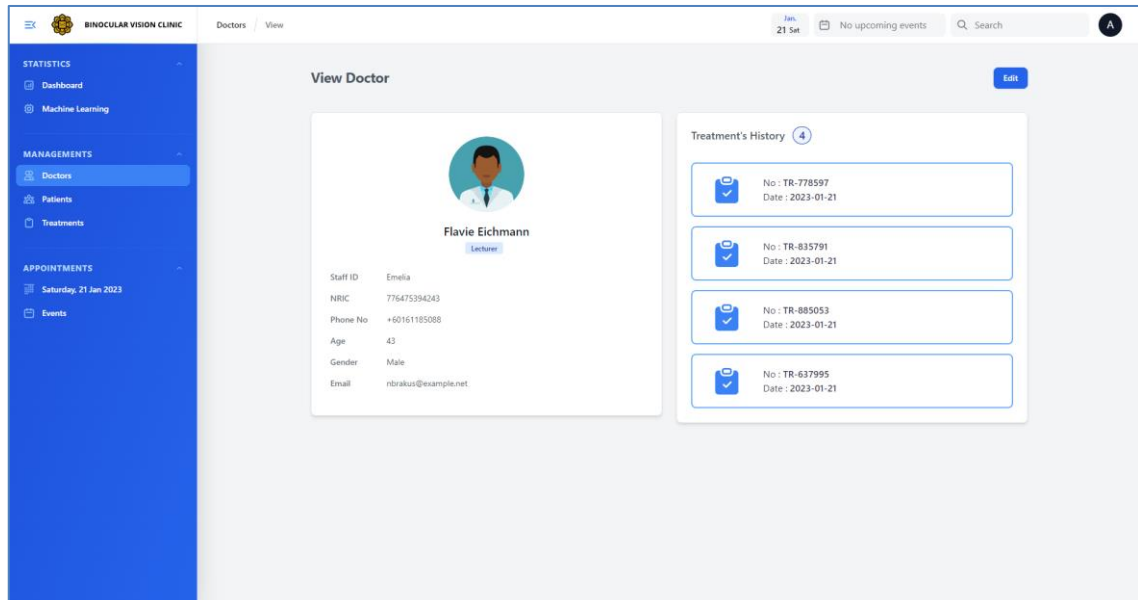


Figure 4.22 View doctor screen

In figure 4.22, the doctor can view the detailed profile of a specific doctor, including their personal information and a list of treatments they have performed previously. By clicking on any of the treatment records listed, the doctor can view further details about the specific treatment. Additionally, the doctor has the ability to edit the doctor’s record by clicking the “Edit” button, which redirects them to the edit doctor page as seen in figure 4.23. This allows the doctor to easily update or make changes to the doctor’s information as needed.

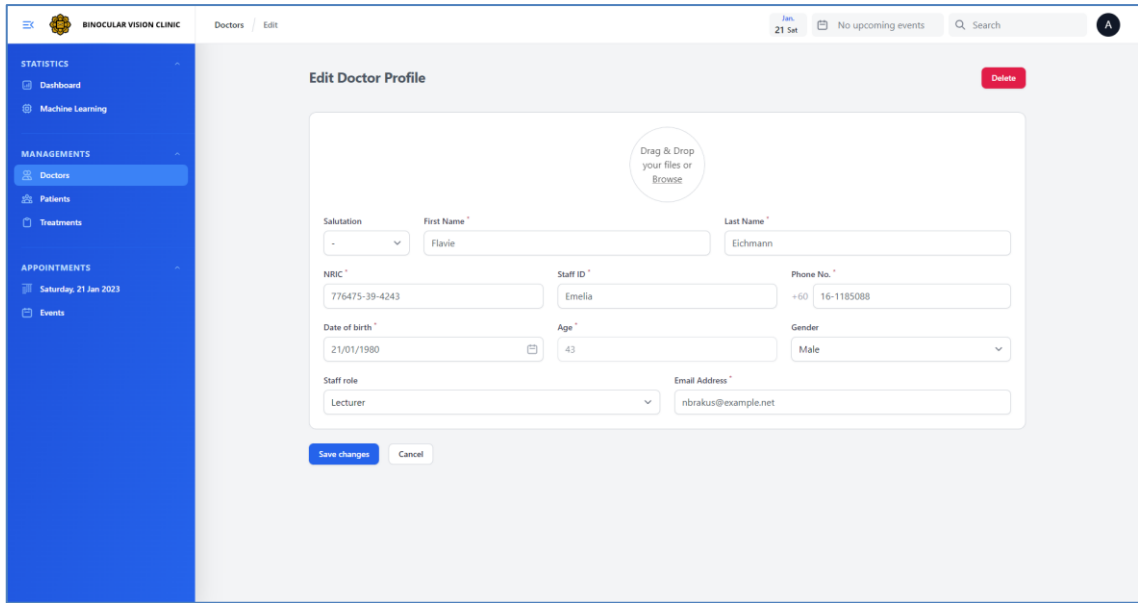


Figure 4.23 Edit doctor screen

In this edit page, the doctor can update the personal details of the doctor by making any necessary changes. To delete the record, the doctor can simply click the delete icon and confirm the deletion by clicking “Yes” on the confirmation dialog, as shown in figure 4.24.

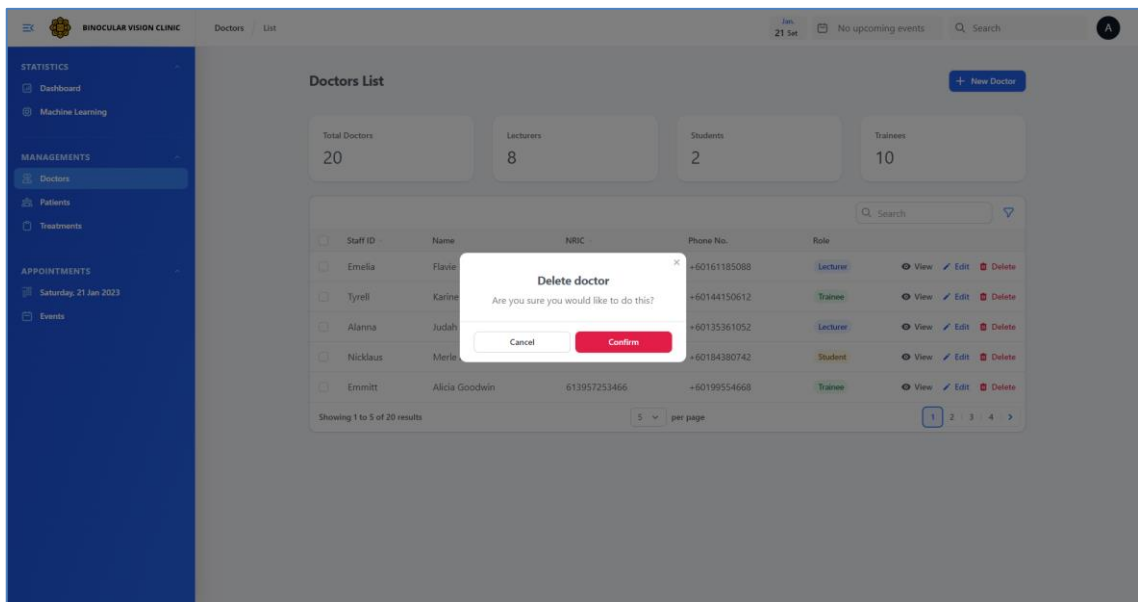


Figure 4.24 Delete doctor screen

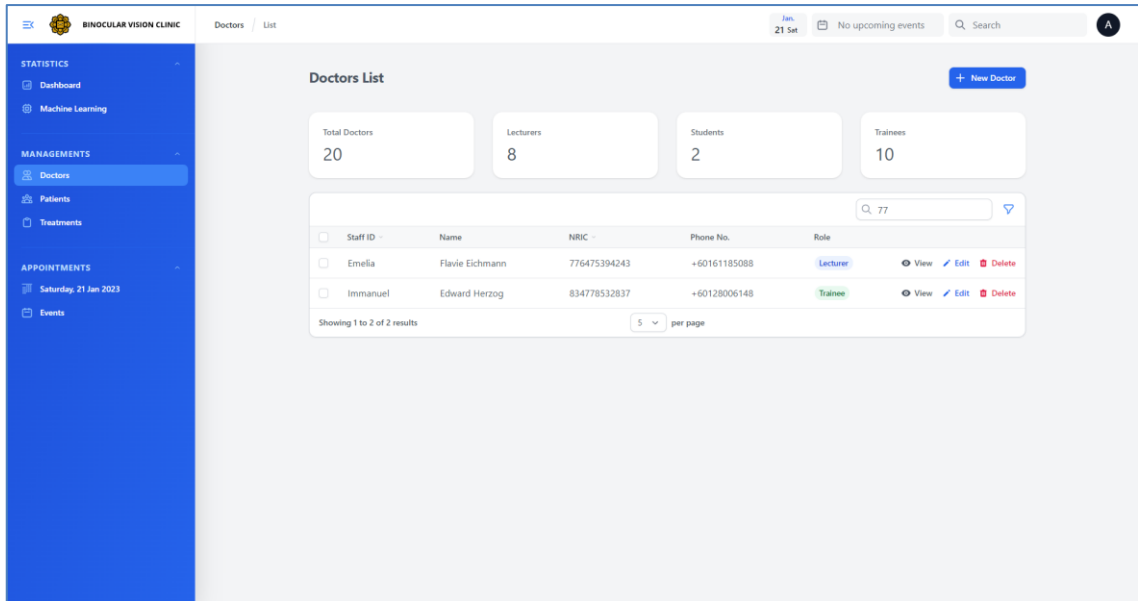


Figure 4.25 Search doctor screen

Moreover, the doctor can search through the doctor record by entering the search keyword like shown in the figure 4.25. The table then will update the table doctor view according to the entered keyword. Based on the figure 4.26, the doctor can export the doctor record list to PDF, Excel or CSV format by clicking the “Export” button. They also able to print the record if they wish to do so by clicking the “Print” button. The additional columns functionality is also available for column addition but it must comes with default value to make it available.

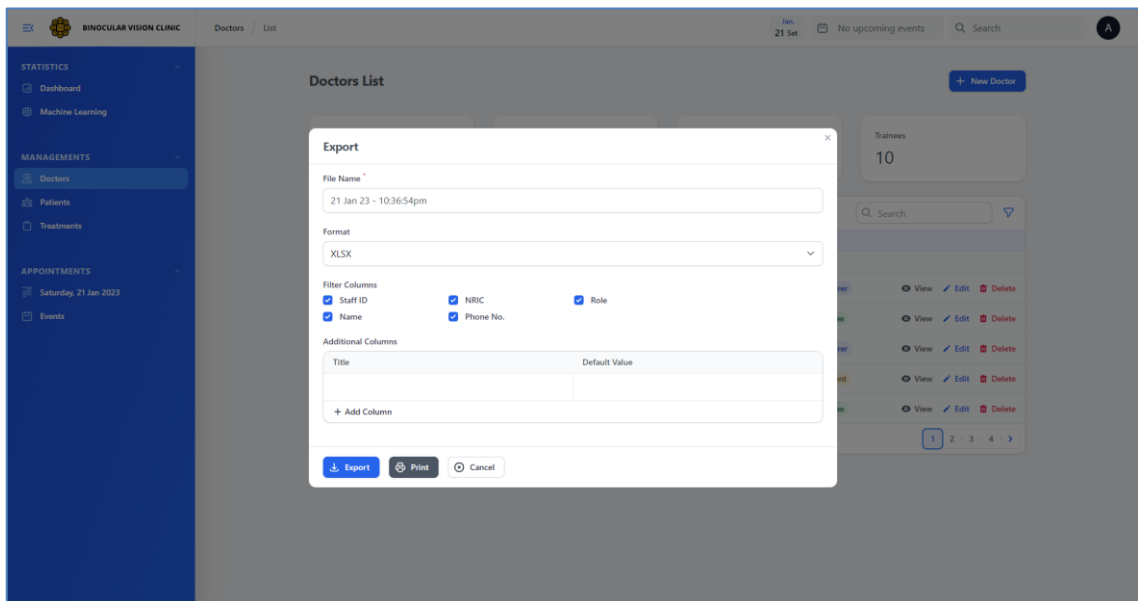


Figure 4.26 Export doctor record screen

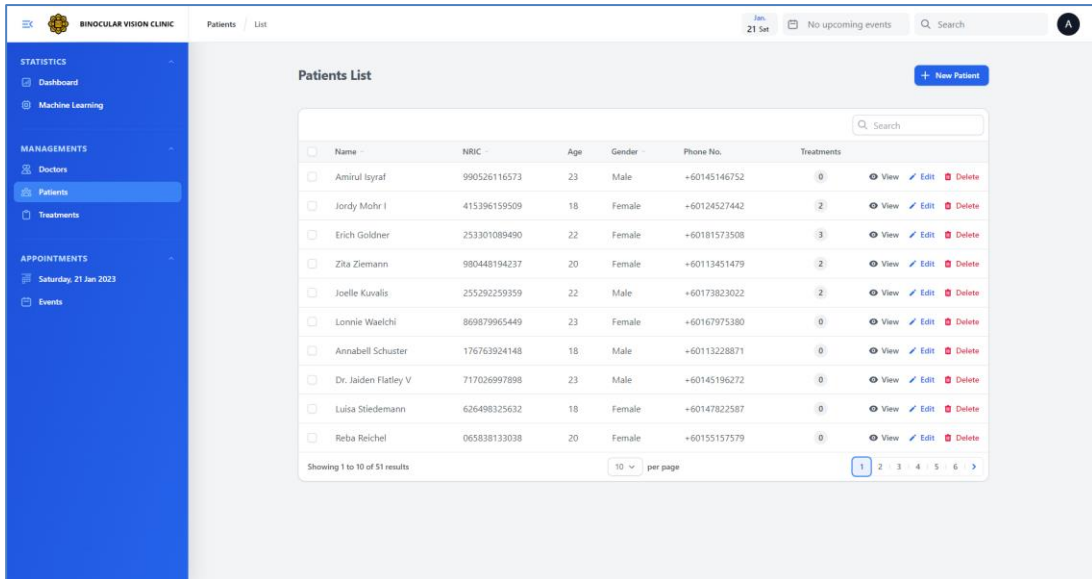


Figure 4.27 Patient list screen

The figure 4.27 shows the manage patient interface for the doctor to view the list of patients that have been added in the system. From the record, the doctor is provided with actions that they can interact. The “eye” icon indicates that the functionality is for viewing patient information, the “pen” icon shows that the functionality is for editing the patient, and the “trash bin” icon indicates that the function is for deleting the patient. Furthermore, the doctor also can search the patient by entering search keyword on the provided search box.

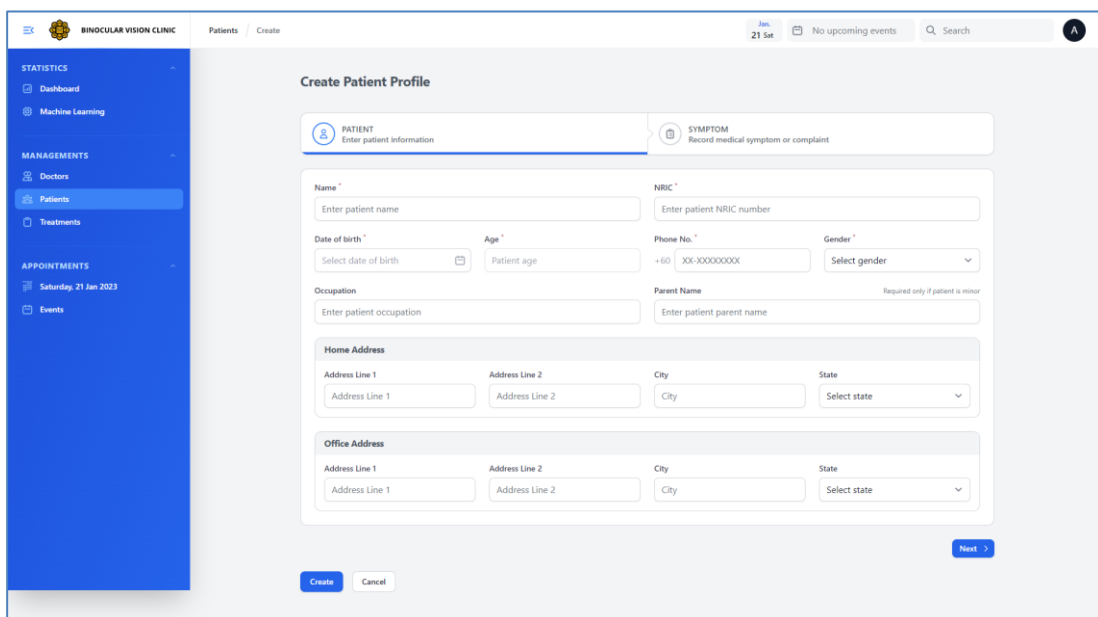


Figure 4.28 Create patient profile (patient detail) screen

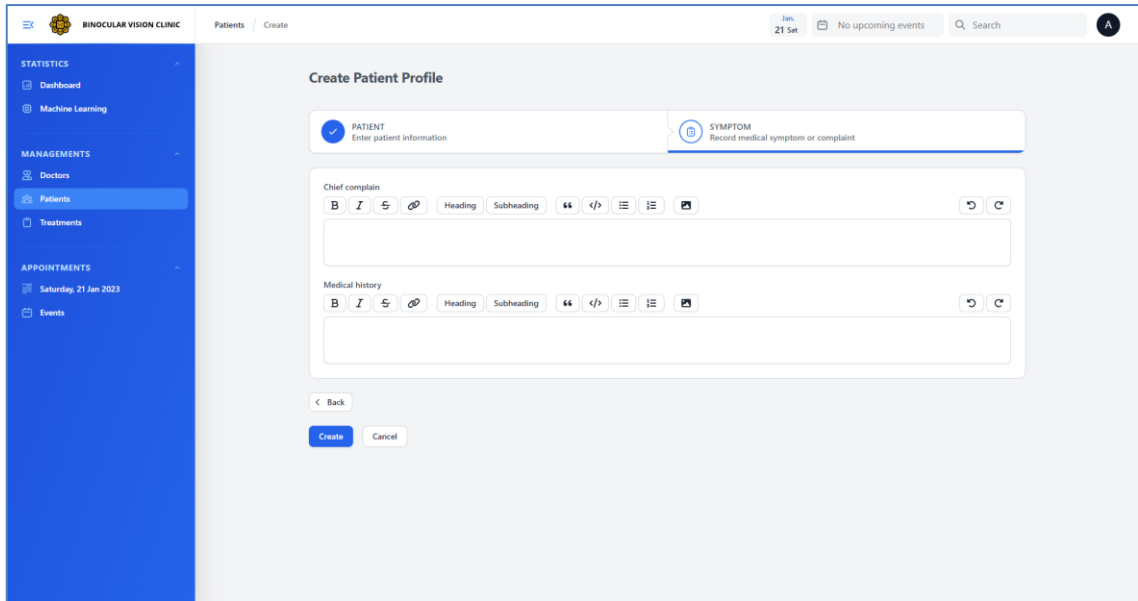


Figure 4.29 Create patient profile (symptom) screen

The add new patient interface in figure 4.28 allows the doctor to enter patient information such as patient name, phone number, age, gender, date of birth, occupation, home address, office address and parent’s name. The “red asterisk” symbols indicate that the fields are required while the rest are optional. The parent’s name should be entered only if the patient is minor. The process is followed by entering the medical symptom if there is any in figure 4.29. After finished entering the patient information, the doctor can click the “Create” button to save the record in the database.

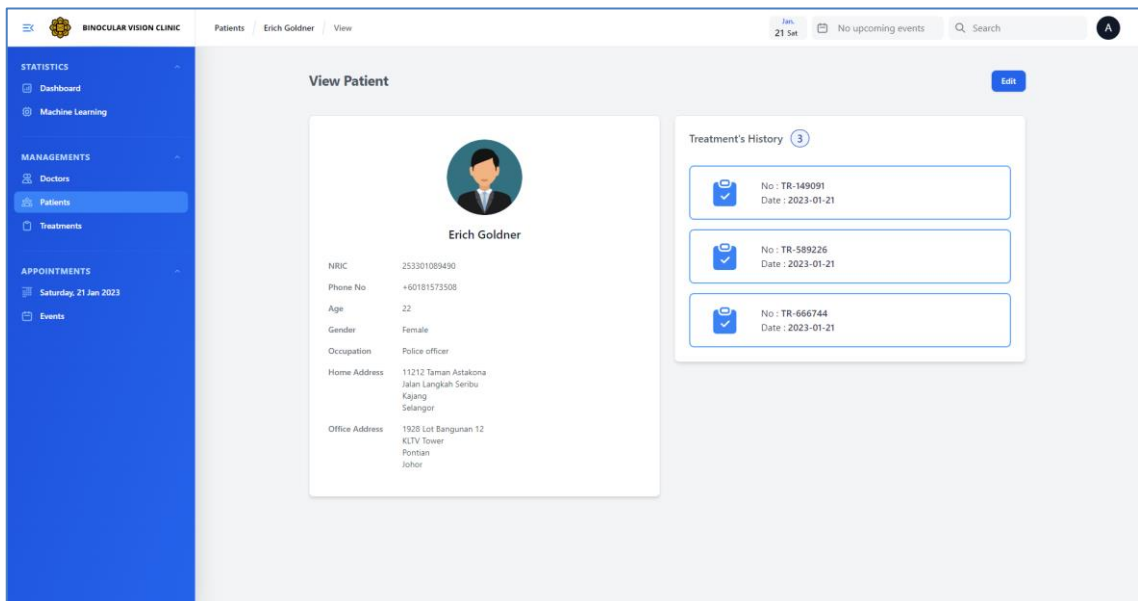


Figure 4.30 View patient screen

If the doctor chooses to click the eye icon in figure 4.27, the doctor will be redirected to the view patient information interface like shown in the figure 4.30. From this page, doctor can view the detailed information of the patient and their treatment history. They also can keep track of the treatment date that has been conducted previously.

The screenshot displays the 'Edit Patient Profile' interface. The left sidebar contains navigation menus for 'STATISTICS', 'MANAGEMENTS', and 'APPOINTMENTS'. The main area is titled 'Edit Patient Profile' and features two tabs: 'PATIENT' (active) and 'SYMPTOM'. The 'PATIENT' tab includes the following fields: Name (Erich Goldner), NRIC (253301-08-9490), Date of birth (21/01/2001), Age (22), Phone No. (+60 18-1573508), Gender (Female), Occupation (Police officer), and Parent Name (Chasity Wiza). Below these are sections for Home Address and Office Address, each with Address Line 1, Address Line 2, City, and State fields. At the bottom, there are 'Save changes', 'Cancel', and 'Next >' buttons.

Figure 4.31 Edit patient screen

If the edit icon in the figure 4.27 is clicked, the system will redirect the doctor to the edit patient page like shown in the figure 4.31. The doctor can edit the patient information to any fields that they want to modify. The “Save Changes” button is provided to allow the doctor to update the record in the database after finished modifying the fields. Moreover, if the doctor clicks the delete icon in figure 4.27, the system will display the doctor deletion confirmation box (figure 4.32), allowing them to double-check their action in the event of an accident. There are two buttons provided, where “Yes” button is used to confirm the deletion, and “Cancel” button will cancel the operation and exit the dialog.

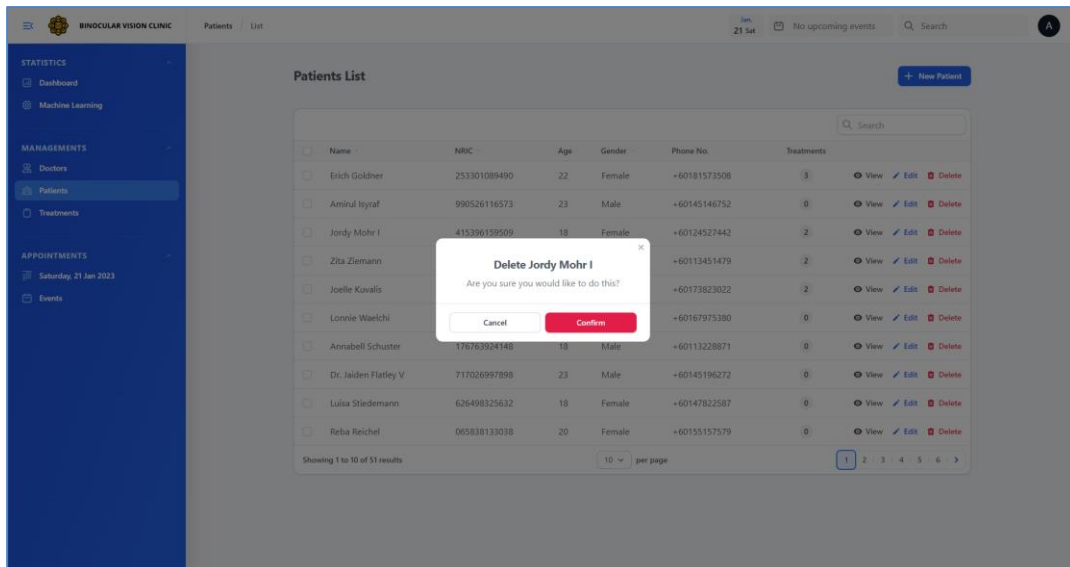


Figure 4.32 Delete patient screen

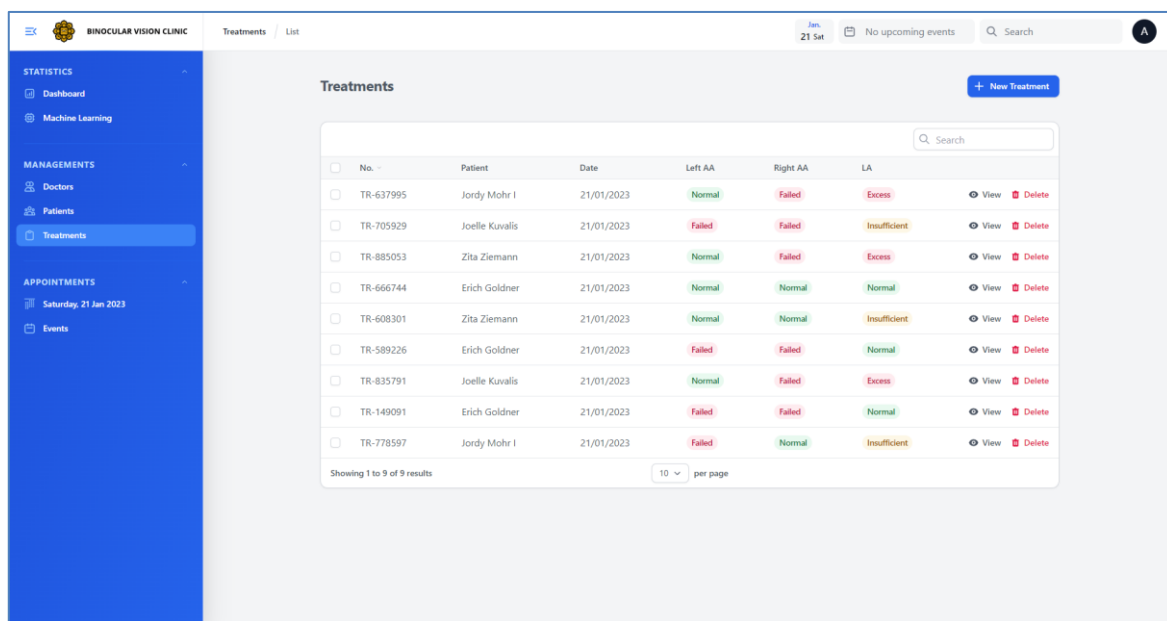


Figure 4.33 Treatments list screen

In figure 4.33, the doctor can easily access and review the treatment records of patients stored in the system's database. The records presented in the table display the results of AA and LA treatments. The color-coded badges displayed in the table provide an indication of the patient's treatment outcomes, with green indicating a normal result, and orange and red indicating that the patient has been diagnosed with strabismus condition.

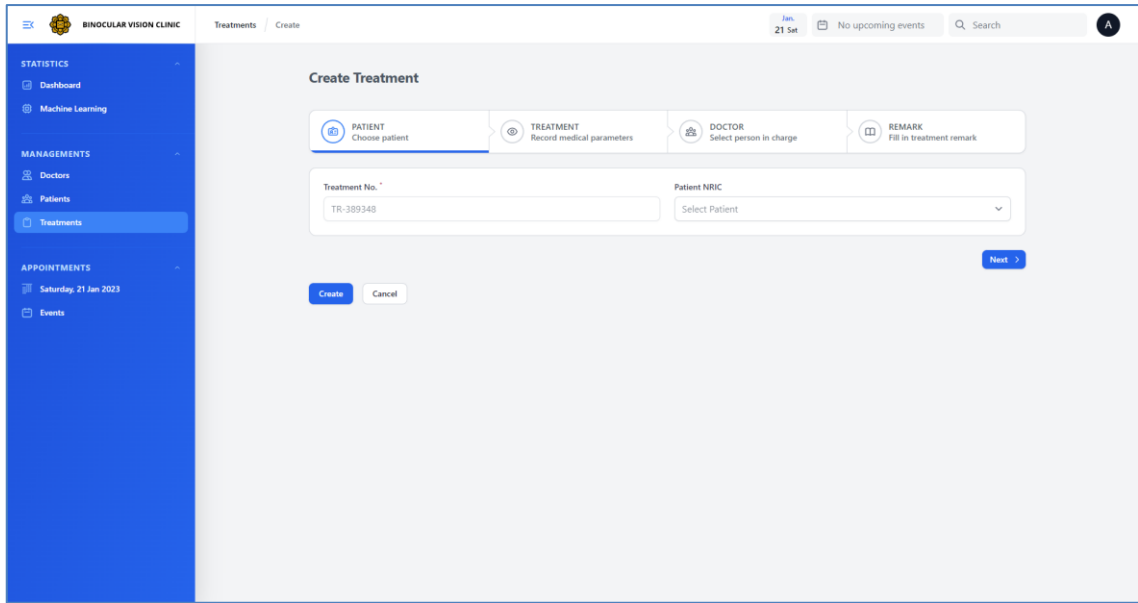


Figure 4.34 Add treatment (patient) screen

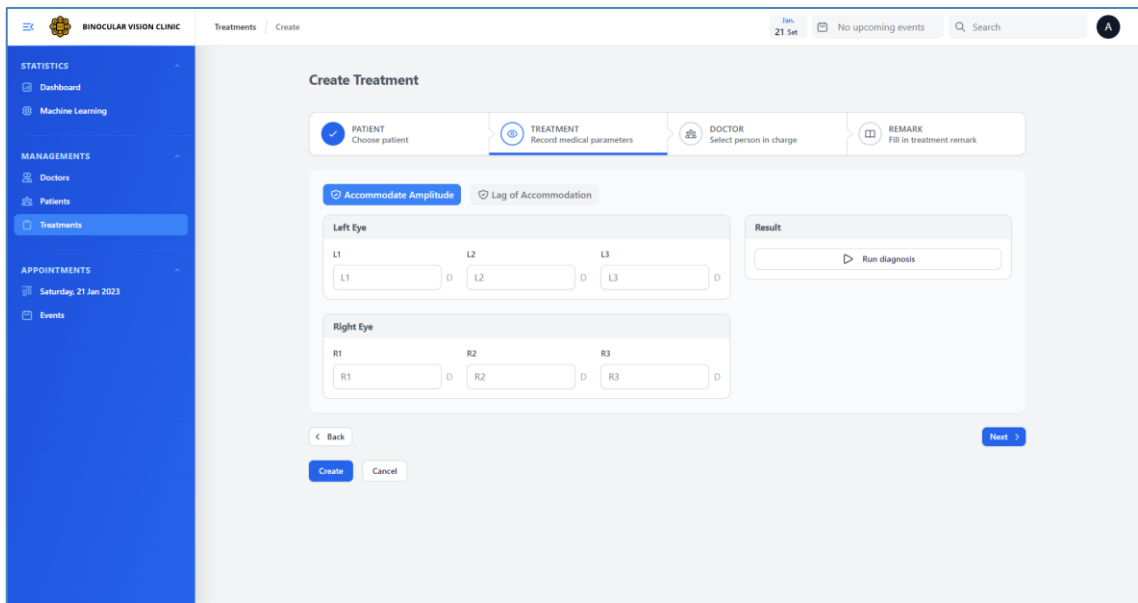


Figure 4.35 Add treatment (accommodate amplitude treatment) screen

In figure 4.34, the system allows doctors to select a patient for treatment by searching for the patient's NRIC in the database. By clicking the “Next” button, the doctor is then able to proceed to the treatment section. In figure 4.35, the doctor is able to input the accommodate amplitude values for each eye for three repetitions and run the diagnosis using the case-based reasoning (CBR) algorithm. The results of the AA treatment are displayed in figure 4.36.



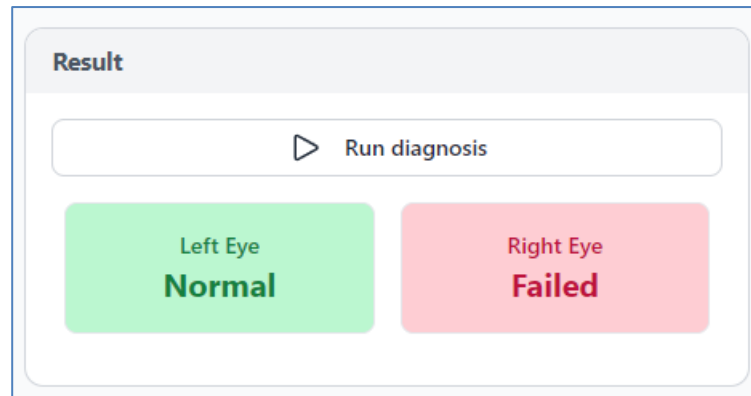


Figure 4.36 Accommodate amplitude treatment result screen

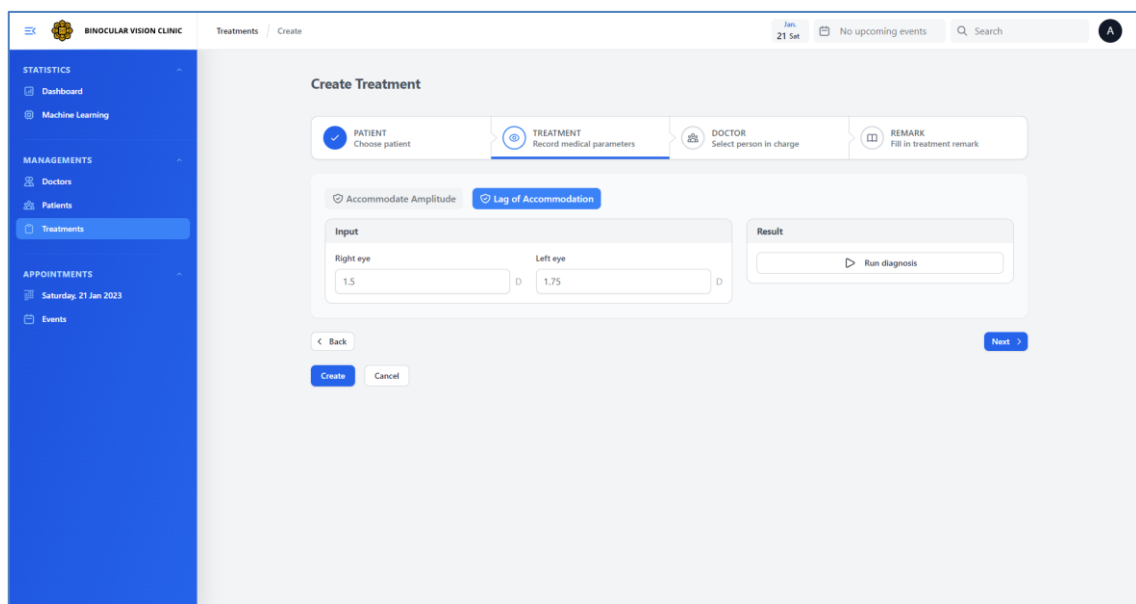


Figure 4.37 Add treatment (lag of accommodation treatment) screen

By referring to figure 4.37, the doctor can proceed with the diagnosis of lag of accommodation by inputting the LA values for each eye. The system will then use the CBR algorithm to classify the condition based on the entered parameters. The result of the diagnosis, whether it is classified as Normal (N), Accommodate Excess (AE) or Accommodate Insufficient (AI), will be displayed in the figure 4.38 for the doctor to review.

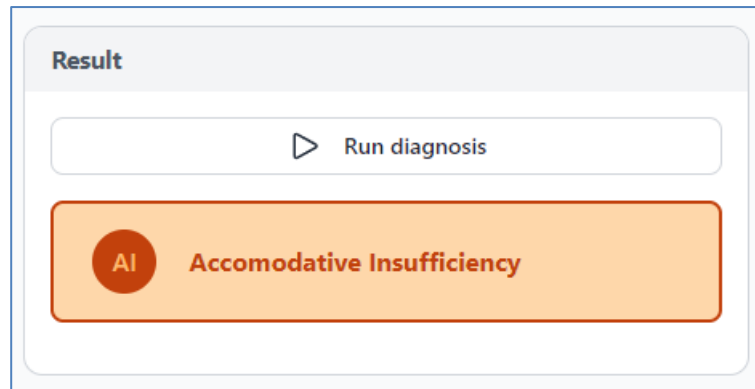


Figure 4.38 Lag of accommodation treatment result screen

The process of selecting the person in charge of the treatment is an important aspect of the system’s functionality, as it ensures that the patients receive the appropriate personnel for their treatment. The figure 4.39 illustrates the user interface for this feature, where the doctor is able to select individuals for the treatment based on their role. Using the repeater field, doctors can add multiple individuals to the treatment team, providing them with the flexibility to assign a team of clinicians that can best cater to the patient’s needs. Additionally, the doctor can easily remove any team members they may have mistakenly added by clicking on the “delete” icon.

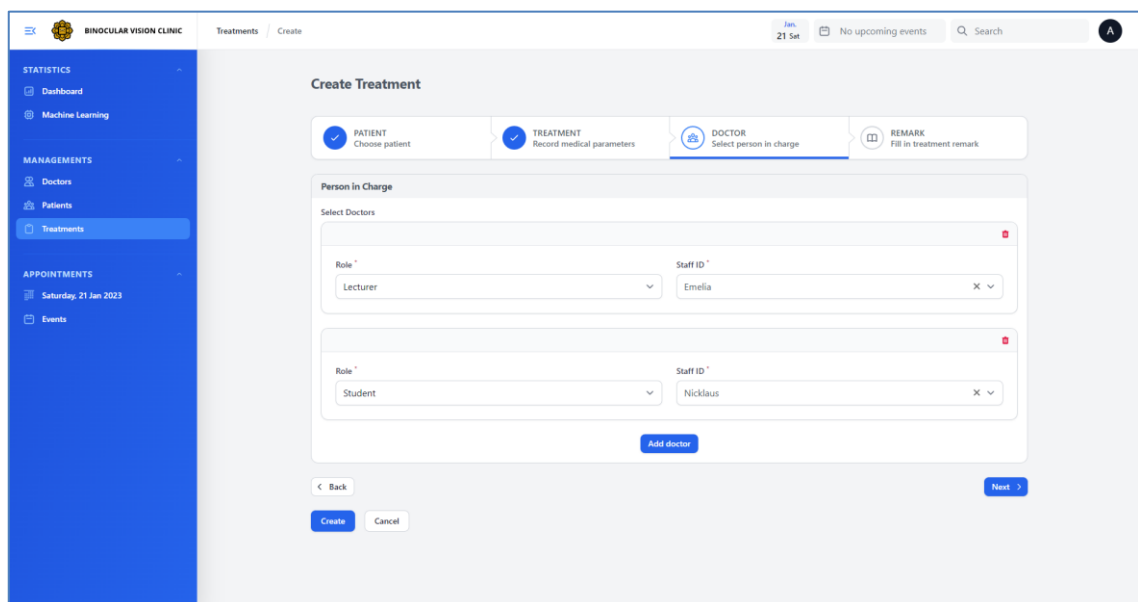


Figure 4.39 Add treatment (person in charge) screen

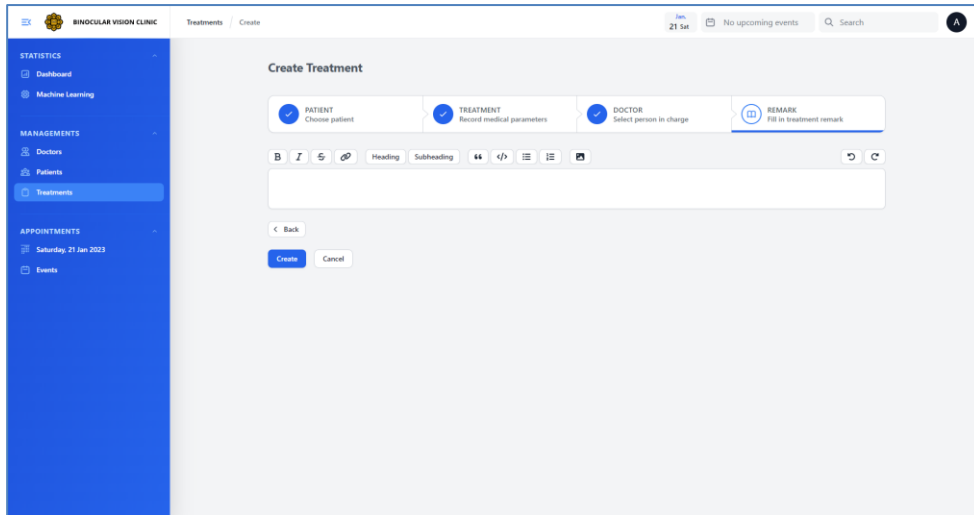


Figure 4.40 Add treatment (remark) screen

In figure 4.40, the doctor has the ability to add a detailed remark for the treatment that has been conducted, including any observations or insights that may be relevant to the patient’s diagnosis. For example, the doctor may include information about the patient’s response to treatment, any challenges that were encountered during the diagnosis process, or any other relevant information that could aid in the patient’s ongoing care. Additionally, the doctor can also note any recommendations for follow-up treatment or further diagnostic testing that may be necessary. By clicking on the “create” button, the remark will be saved and stored in the treatment’s record for future reference in the database.

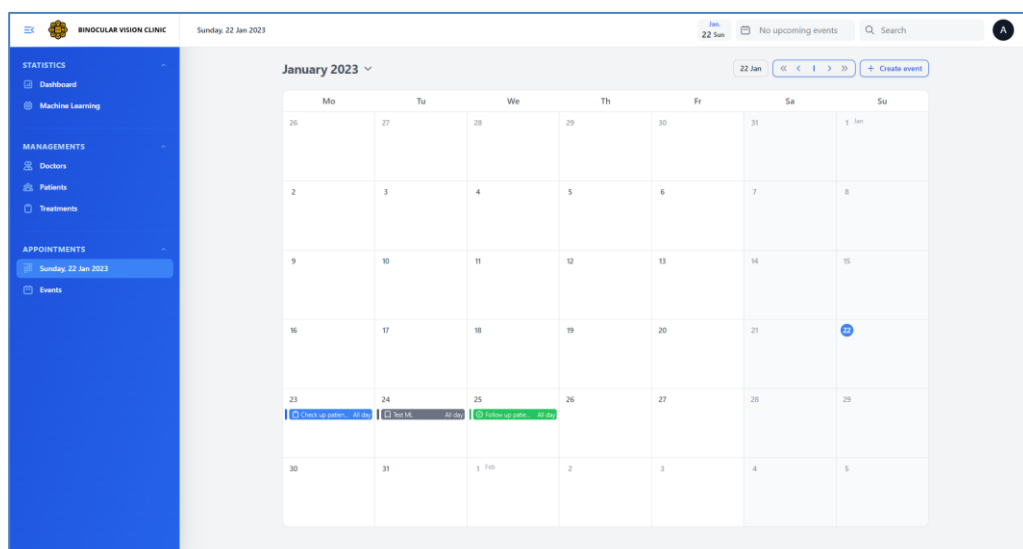


Figure 4.41 Appointment calendar screen

In figure 4.41, the appointment calendar is depicted, displaying the records of all previously scheduled appointments. The doctor can navigate through different months by clicking the “January 2023” drop-down menu, which allows them to select the desired month. To schedule a new appointment, the doctor can simply click on the “Create Event” button. To view the details of an existing appointment, the doctor can simply click on the event in the calendar, and a modal window will appear, displaying all relevant information, as shown in figure 4.42. The modal displays the date of the appointment, its category and patients that will be involved for the appointment.

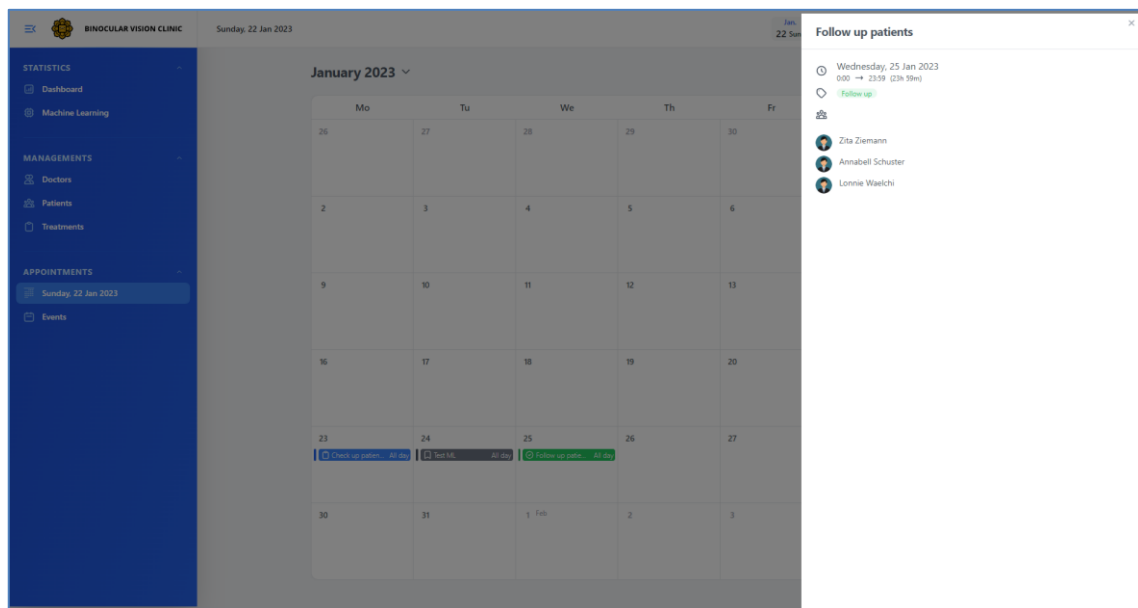


Figure 4.42 View event details screen

By clicking on the “Create Event” button in the figure 4.41, a side modal appears, allowing the doctor to schedule a new appointment which can be seen in the figure 4.43. The doctor can fill in the subject of the appointment, such as “Follow-up Consultation” or “Strabismus Surgery Planning” in the “Subject” field. They can also provide additional details about the appointment in the “Body” field. The “Participants” field allows the doctor to select the patient or patients who will be involved in the appointment. Additionally, the “Category” field allows the doctor to categorize the appointment as a general check-up, follow-up, or other type of appointment. The “All day” switch allows the doctor to schedule the appointment for the entire day. Lastly, the doctor can also attach relevant documents or images to the appointment by using the “Attachment” field.

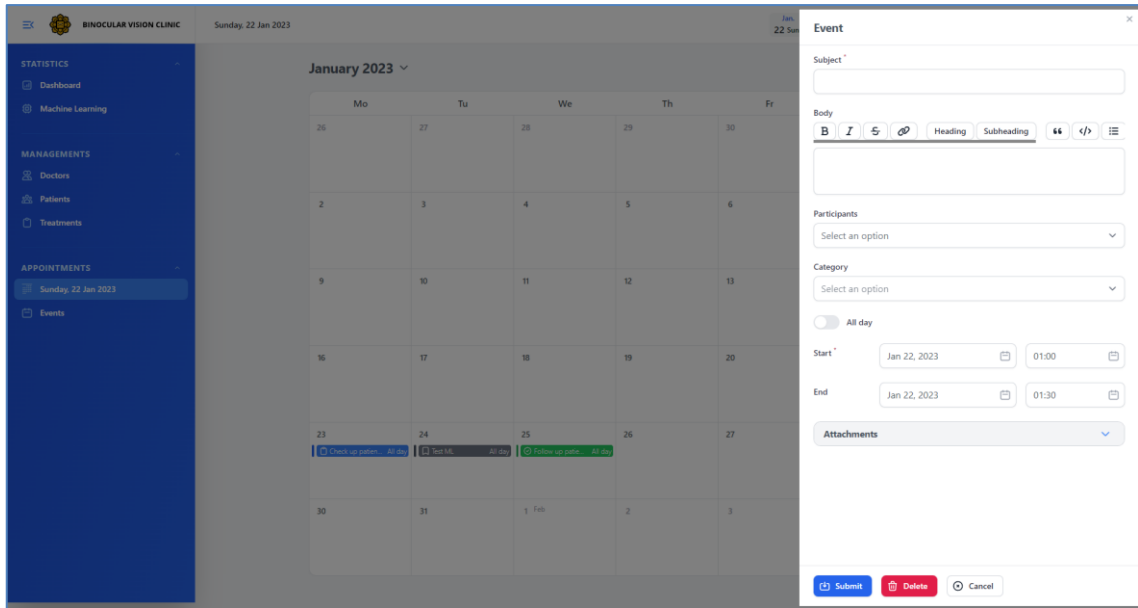


Figure 4.43 View event details screen

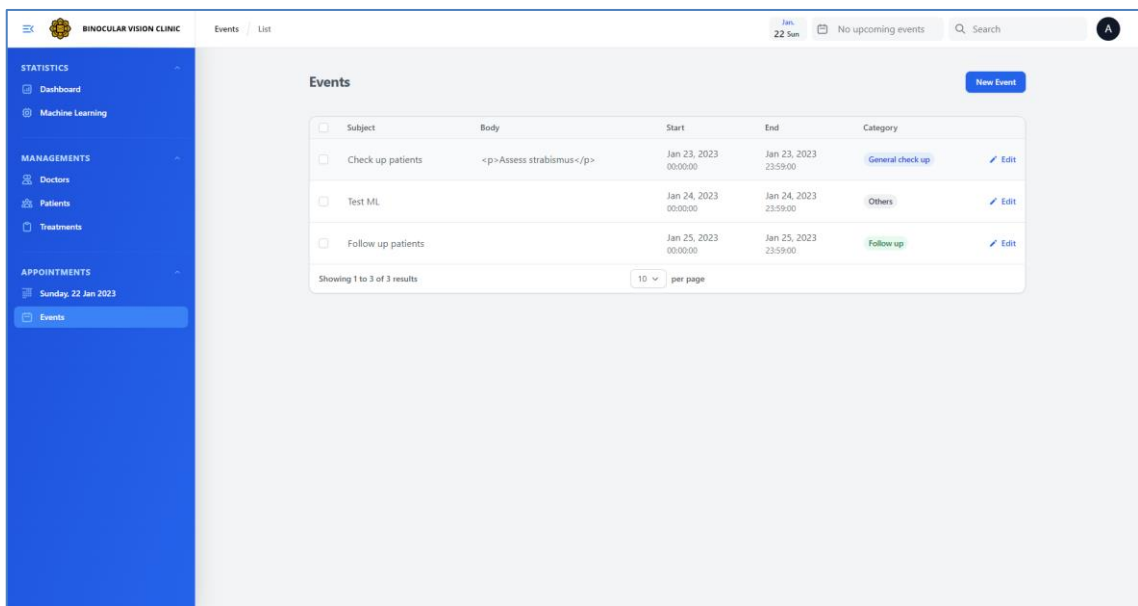


Figure 4.44 Event list screen

In figure 4.44, the event list is displayed in a table for easy viewing of scheduled appointments. From this view, doctors are able to quickly view event details and categorization. Based on figure 4.45, the edit functionality allows doctors to make any necessary changes to the appointment, such as rescheduling or updating participant information. This way, it provides them the ability to make updates and adjustments as needed, ensuring the efficiency of the appointment’s feature. They also able to delele the event by clicking on the “Delete” button.

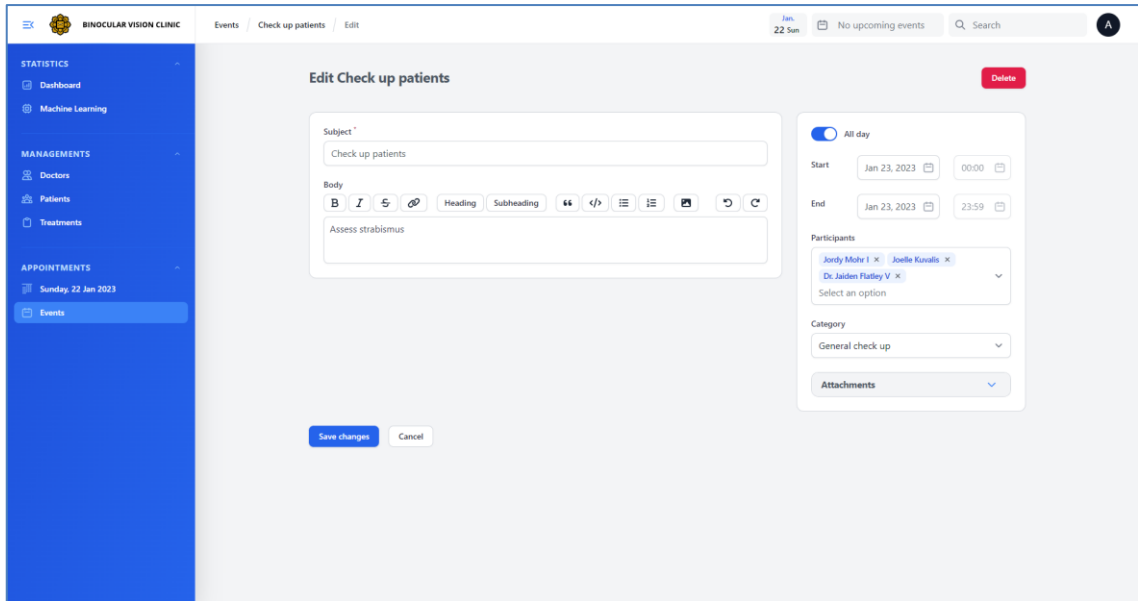


Figure 4.45 Edit event screen

#### 4.6 CASED-BASED REASONING PERFORMANCE EVALUATION

The performance of the Case-Based Reasoning (CBR) model for Accommodate Amplitude (AA) diagnosis was evaluated using various metrics, including accuracy, precision, recall, and F1-score. The figure 4.46 presents the statistics of the model performance for the system. The last run of testing was conducted today, and the total number of tests performed so far is 3. The dataset used for these tests consists of 151 cases. The data was split into training and testing datasets, with a 60:40 ratio, respectively. This split allowed the system to train on a significant amount of data and then test its performance on the remaining data to evaluate its accuracy.

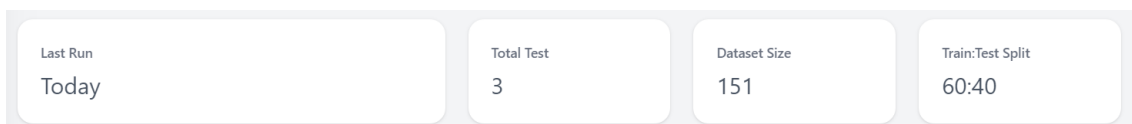


Figure 4.46 Model performance data statistics

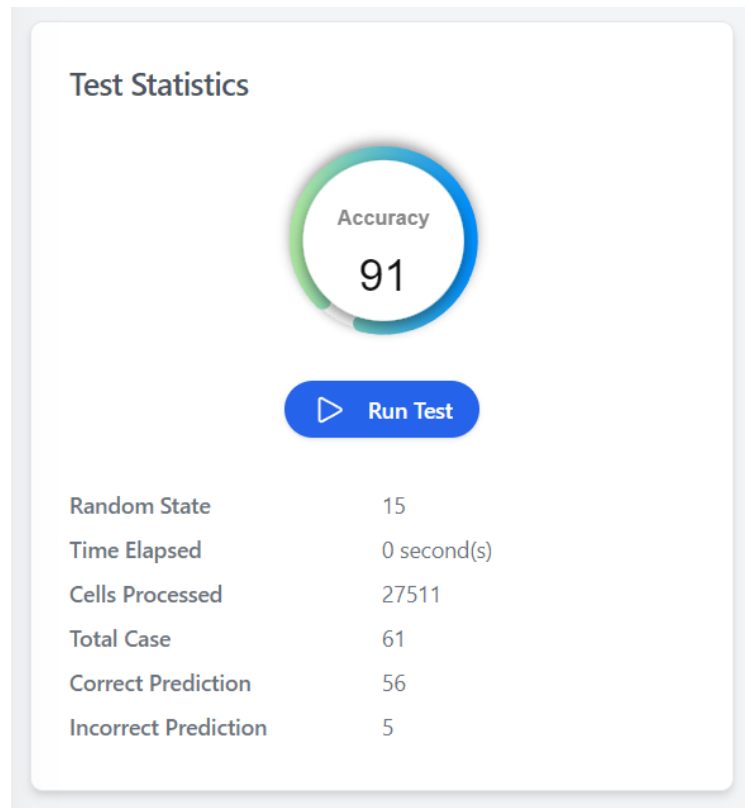


Figure 4.47 Model performance test statistics

The results shown in the figure 4.47 indicate that the model achieved an overall accuracy of 91 percent with a train and test split configuration of 60:40. In order to measure the performance of the model, a random sample of 26 was used. The figure also provides information about the time elapsed during the test, the number of processed cells, the total number of cases, the number of correct and incorrect predictions. These statistics provide valuable insights into the performance of the model.

In addition, the figure 4.48 provides a visual representation of the accuracy of the Case-Based Reasoning model over time. The graph displays the results of multiple test runs, allowing doctors to track the progress and improvement of the model. It is also based on a specific random sample, and the accuracy may be different if a different sample set is used. This highlights the importance of thoroughly testing the model using various sample sets to ensure robust and consistent performance.

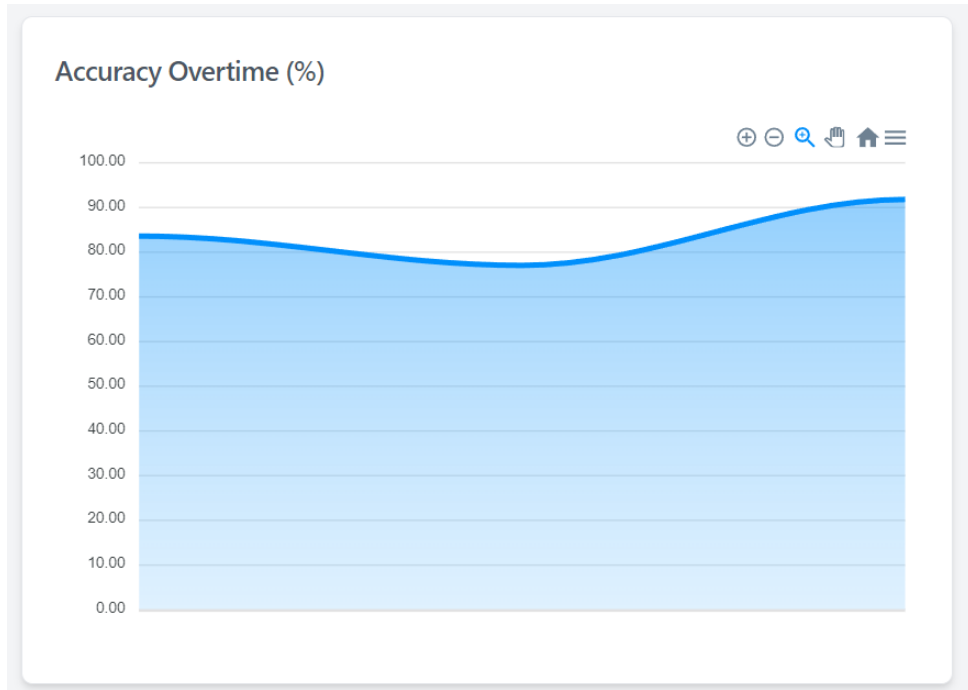


Figure 4.48 Model accuracy overtime

The figure 4.49 provides a comprehensive evaluation of the model's performance by plotting four crucial metrics, which are the accuracy, precision, recall, and F1 score. The high results of all the metrics indicate that the model has a high degree of accuracy and is able to make correct predictions. This suggests that the model has been well trained and is capable of providing accurate results for Accommodate Amplitude diagnosis.



Figure 4.49 Model performance measures



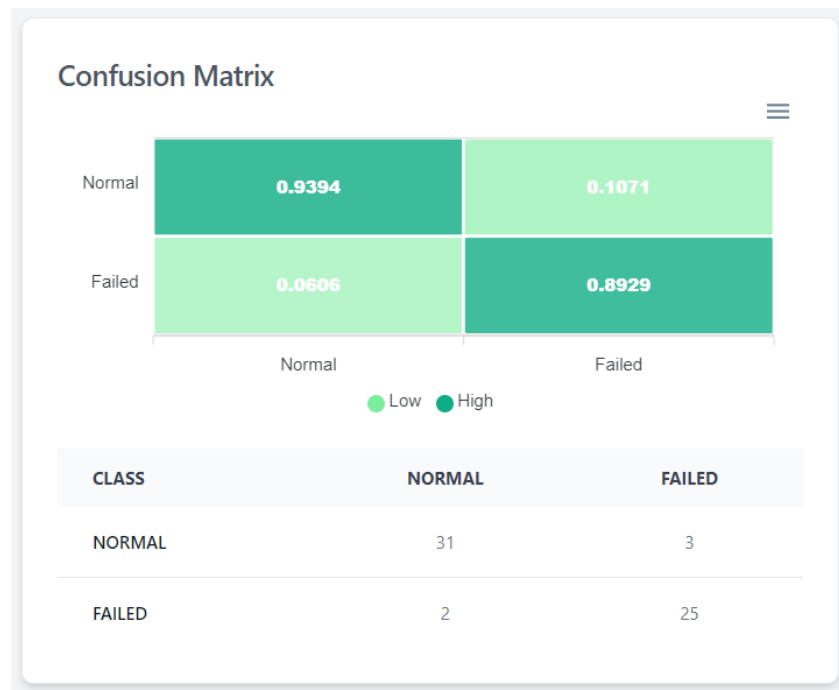


Figure 4.50 Model confusion matrix

The confusion matrix in the figure 4.50 helps to evaluate the accuracy of the model by illustrating the number of true positive, false positive, true negative, and false negative predictions made by the model for the "normal" and "failed" classes.

## 4.7 USER ACCEPTANCE TEST

User acceptance testing (UAT) is a critical phase in the software development process, as it allows end-users to test the system and provide feedback on its functionality and usability. For this project, UAT would involve a series of tests that are designed to ensure the system meets the needs and expectations of the IIUM doctors who will be using it. Based on the figure 4.51, The UAT would be conducted by having the IIUM doctors test the system in a live meeting, and they are required to provide feedback by filling out a Google Form like shown on the figure 4.52.

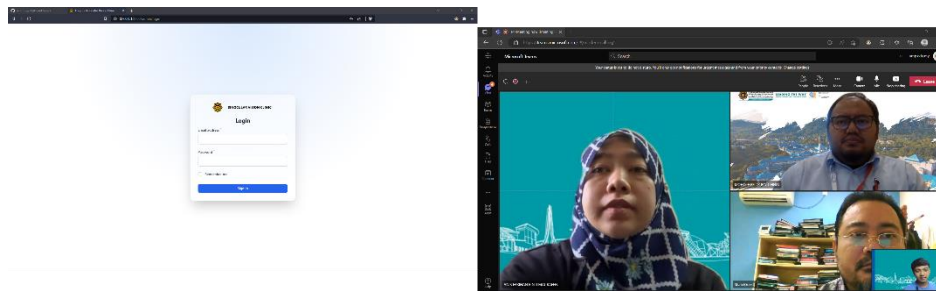


Figure 4.51 Live meeting of user acceptance test

Figure 4.52 User acceptance test form introduction

In figure 4.53, the system displays the details of the respondents who are participating in the user acceptance testing (UAT) process. This includes their name and contact information, allowing the developer to easily reach out to them in case of any questions or concerns. Additionally, for a more detailed overview of the UAT responses, refer to appendix A for the complete responses of the user acceptance test.

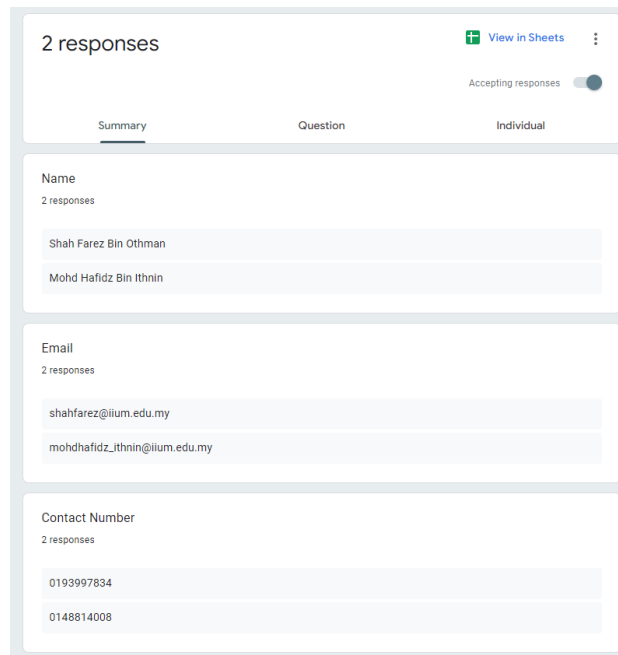


Figure 4.53 Respondent personal details

The figure 4.54 includes the tester declaration to confirm that they have completed and submitted the UAT form. This serves as a way to formally document and track the completion of the UAT process. It confirms that the testers have fully evaluated the system and provided their feedback, allowing the developer to move forward with confidence in the system’s readiness for deployment. Finally, as shown in figure 4.50, the signature and stamp of the testers serve as a proof and verification of the feedback provided. This adds a level of authenticity and accountability to the UAT process, ensuring that the feedback is coming from a verifiable source and can be trusted.

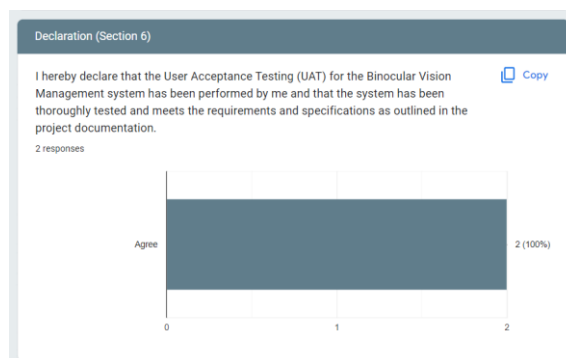


Figure 4.54 Declaration agreement response

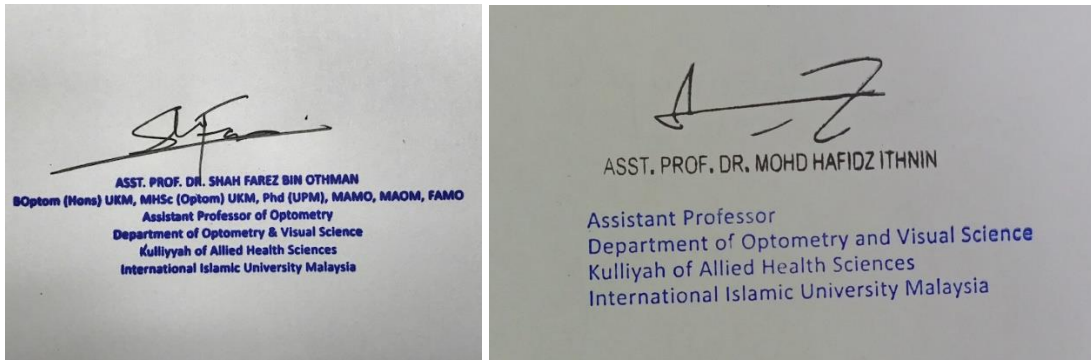


Figure 4.55 Respondent signatures and stamps

## **CHAPTER 5**

### **CONCLUSION**

#### **5.1 INTRODUCTION**

The thesis is divided into five chapters. The first chapter provides an overview of the proposed system, including its objectives, scope, and the problems that led to its development. The second chapter compares the proposed system with three existing clinic systems, highlighting their advantages and disadvantages. Based on the comparison, some parts of the system such as the machine learning classification function were inspired by existing clinic systems as they do not have the ability to classify strabismus using machine learning algorithms.

Furthermore, the third chapter dives into the methodology and algorithms used in the system's development. For example, it explains how Agile methodology was used throughout the development process and the inner workings of the case-based reasoning algorithm used in this proposed system. It also covers the system flowcharts and the machine learning framework. In chapter four, the output of the developed system is presented, including its interfaces and some sample codes. The testing and results of the proposed system are discussed in this chapter to ensure that it meets the requirements outlined in the SRS. Overall, the proposed system does have its constraints and limitations, therefore there is potential for further improvement in the future.

#### **5.2 OBJECTIVE REVISITED**

The objectives of this project have been fully achieved. A comprehensive study of existing clinic systems, such as Odoe Eye Clinic, Optic Clinic, and Smart Eye Care, was conducted to understand the limitations of current eye clinic systems. This study revealed that all systems lack the use of machine learning for automating strabismus classification. Thus, it served as a foundation for the development of a binocular vision

management system with Case-Based Reasoning as the machine learning algorithm for classifying strabismus automatically.

Furthermore, the proposed system was developed, and its functionality was thoroughly tested. The results of the testing have shown that the system is able to accurately classify strabismus using the diagnostic test results generated by the case-based reasoning algorithm. The system also includes functionalities such as doctor registration, patient, treatment, and appointment management, and interfaces for training and testing the machine learning model, making it a complete and useful tool for eye doctors in the diagnosis and treatment of strabismus. In summary, the objectives of this project were met, and the proposed binocular vision management system with the integration of Case-Based Reasoning algorithm will be a valuable addition to IIUM doctors in the field of ophthalmology.

### **5.2.1 Limitations**

The proposed system has a few limitations that need to be considered. Firstly, at the moment, it only supports two strabismus diagnosis which are accommodative amplitude and lag of accommodation. As a result, it cannot classify strabismus with other types of diagnosis like negative or positive relative accommodation, monocular or binocular accommodative facility. Another limitation is that the system's classification accuracy may be affected by the small dataset size. The system was trained and tested on a dataset of patients aged between 19 and 23, which means it may not function as effectively for patients outside of this age range.

Moreover, the system only uses numerical parameters and does not support the use of eye images or videos for strabismus classification, which means it misses out on the additional information that can be obtained from these types of diagnostic data. For example, if an image or video is taken of a patient's eyes, the system would not be able to use this information to classify the patient's strabismus. Finally, because the system is web-based, an internet connection is necessary to use it. This means that the system cannot be used without an online connection and might not be usable in places with no internet connectivity.

### **5.2.2 Future Works**

The future plan of this system includes expanding the number of strabismus diagnoses that it can perform. Currently, it only performs diagnosis of accommodative amplitude and lag of accommodation, but in the future, it could be developed to also perform other diagnoses such as negative and positive relative accommodation, monocular and binocular accommodative facility. This would allow for a more comprehensive diagnosis and ultimately lead to a more accurate strabismus prediction for patients.

In addition, for future development, the plan also is to expand the dataset. By having a larger and more diverse dataset, the system would be able to better generalize to a wider range of patients and increase its accuracy for patients outside of the initial dataset's age range. This could be achieved by collecting more data from patients with different characteristics or symptoms and incorporating that data into the training and testing process.

## REFERENCES

- Alsaqqa, S., Sawalha, S., & Abdel-Nabi, H. (2020). Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies (iJIM)*, 14(11):246.
- Andre Scherr, S., Elberzhager, F., & Holl, K. (2018). Acceptance Testing of Mobile Applications - Automated Emotion Tracking for Large User Groups. *2018 IEEE/ACM 5th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, 247-251.
- Babu, R., Jalaiah, S., & Bhushanam, P. (2019). Quality improvement measures for Software. *The International journal of analytical and experimental modal analysis*, 336.
- Chen, Z., Fu, H., Lo, W.-L., & Chi, Z. (2018). Strabismus Recognition Using Eye-Tracking Data and Convolutional Neural Networks.
- Lamy, J.-B., Sekar, B., Guezennec, G., Bouaud, J., & Séroussi, B. (2019). Explainable artificial intelligence for breast cancer: A visual case-based reasoning approach. *Artificial Intelligence in Medicine*, 42-53.
- Montalvo, A., Parra, P., & R. Polo, O. (2019). Towards the Use of Model-Driven Technologies in an Integral Software Development Process. *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE.
- Repka, Lum, & Burugapalli. (2018). Strabismus, strabismus surgery, and reoperation rate in the united. *Ophthalmology*, 1646–1653.
- Shakya, S. (2020). Analysis of Artificial Intelligence based Image Classification Techniques. *Journal of Innovative Image Processing*, 44-54.
- Stidwell, D., & Fletcher, R. (2017). *Normal Binocular Vision: Theory, Investigation and Practical Aspects*. John Wiley & Sons.
- Zhang, L., Cao, Y., Yang, F., & Zhao, Q. (2017). Machine Learning and Visual Computing. *Applied Computational Intelligence and Soft Computing*.

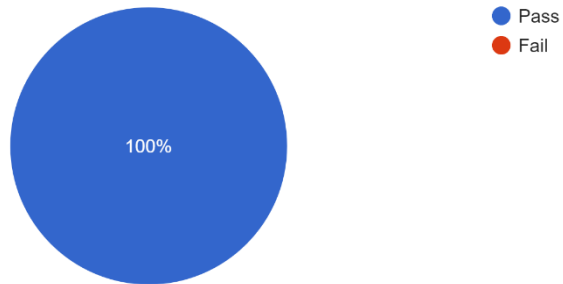


## APPENDIX A USER ACCEPTANCE TEST RESPONSES

### a) Manage Login Module

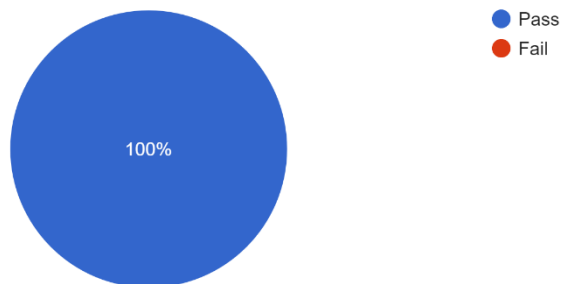
1. The user is able to login by entering the correct username and password.

2 responses



2. The user is shown an error message if incorrect login credentials are entered.

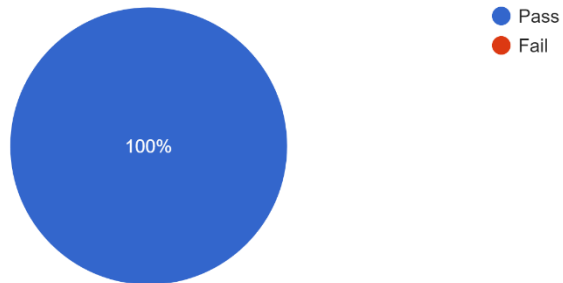
2 responses



## b) Manage Doctor Module

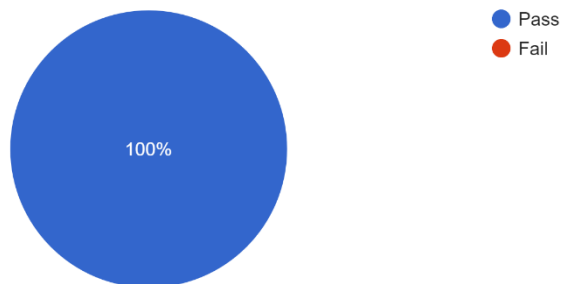
1. The user is able to add new doctor by filling in the doctor's name, NRIC and other personal details.

2 responses



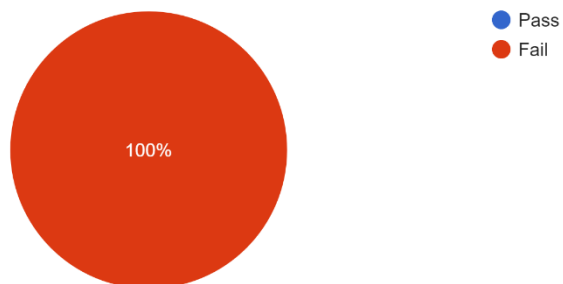
2. The user is able to view all registered doctors in a table.

2 responses

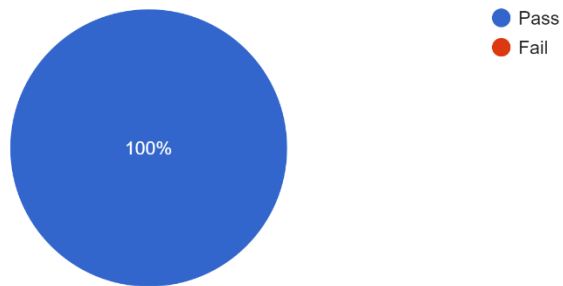


3. The user is able to view doctor's details and their performed treatments.

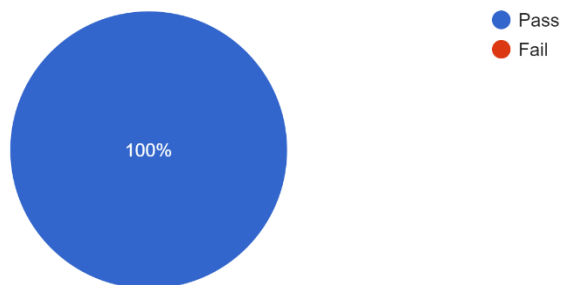
2 responses



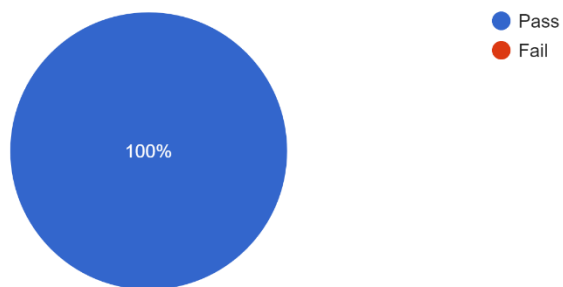
4. The user is able to search for doctors by entering search keyword in the search doctor field.  
2 responses



5. The user is able to edit doctor information when needed.  
2 responses

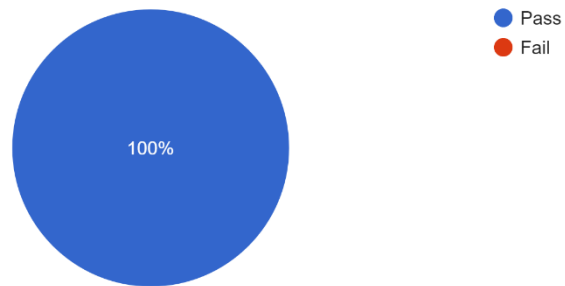


6. The user is able to delete doctor record after clicking "yes" in the confirmation dialog.  
2 responses



7. The user is able to export doctor's record to PDF, CSV and Excel Format.

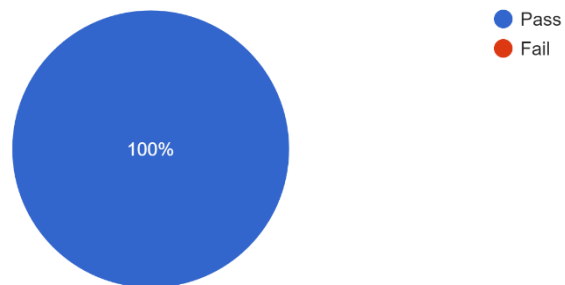
2 responses



### c) Manage Patient Module

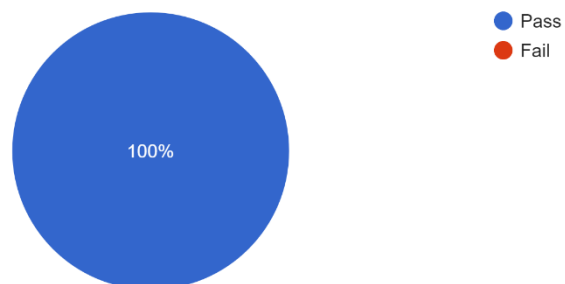
1. The user is able to add new patient by filling in the patient's name, NRIC and other personal details.

2 responses

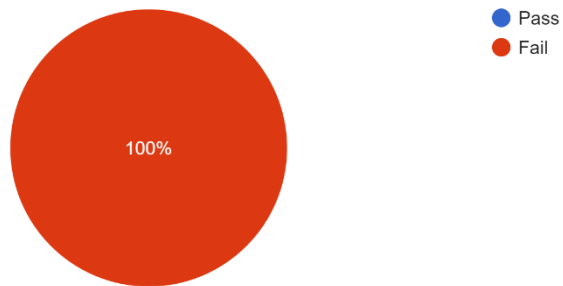


2. The user is able to view all registered patients in a table.

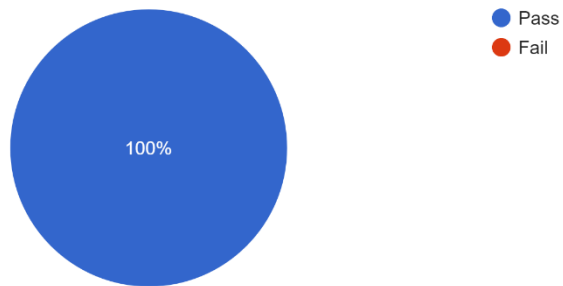
2 responses



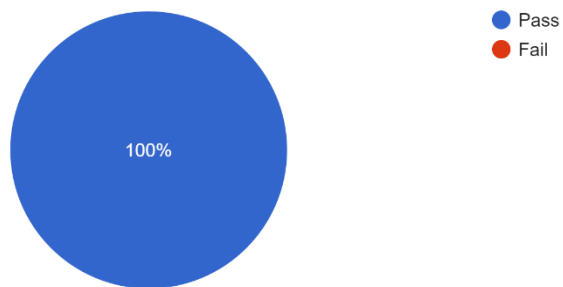
3. The user is able to view patient's details and their treatment history.  
2 responses



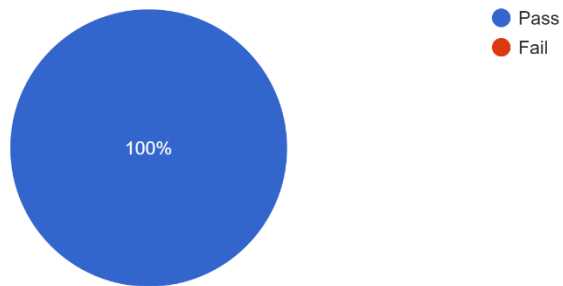
4. The user is able to search for patients by entering search keyword in the search patient field.  
2 responses



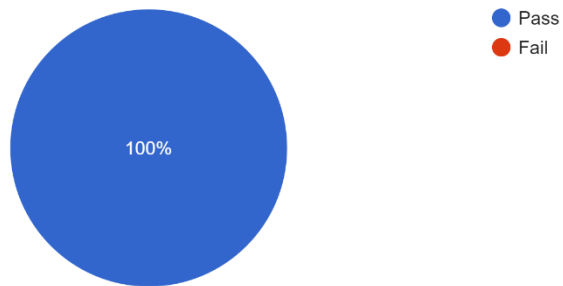
5. The user is able to edit patient information when needed.  
2 responses



6. The user is able to delete patient's record after clicking "yes" in the confirmation dialog.  
2 responses

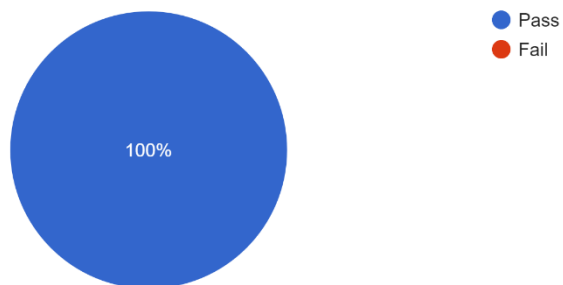


6. The user is able to delete patient's record after clicking "yes" in the confirmation dialog.  
2 responses



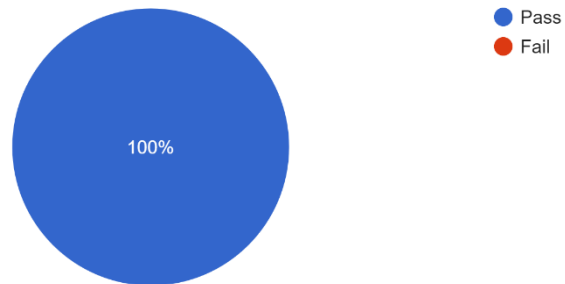
#### d) Manage Treatment Module

1. The user is able to choose patient for the treatment.  
2 responses



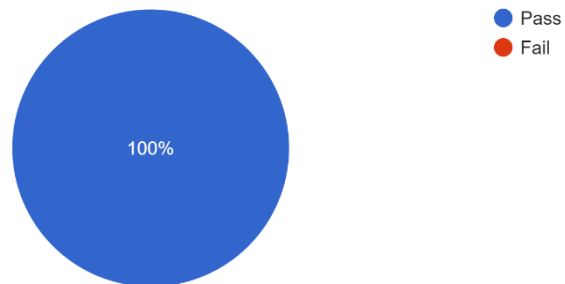
2. The system is able to accurately classify the accommodate amplitude diagnosis for each eye (Normal/Failed) through Case-Based Reasoning algorithm.

2 responses



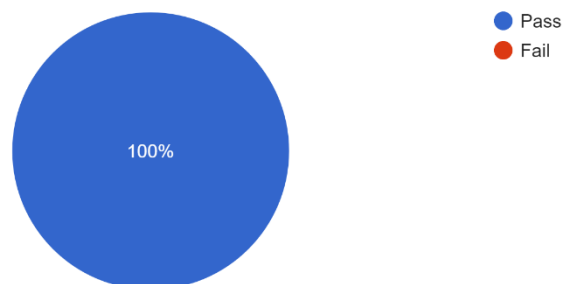
3. The system is able to accurately classify the lag of accommodation diagnosis for both eyes (Normal/Accommodate Insufficient/Accommodate ...ess) through Case-Based Reasoning algorithm.

2 responses



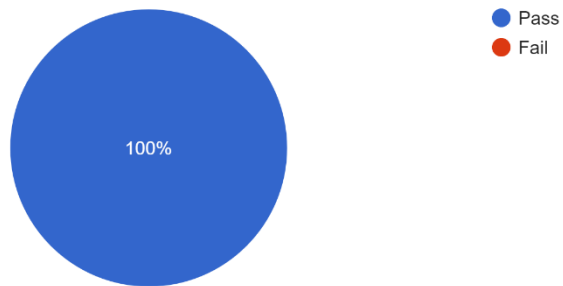
4. The user is able to assign person in charge (doctors) according to their role in the treatment.

2 responses



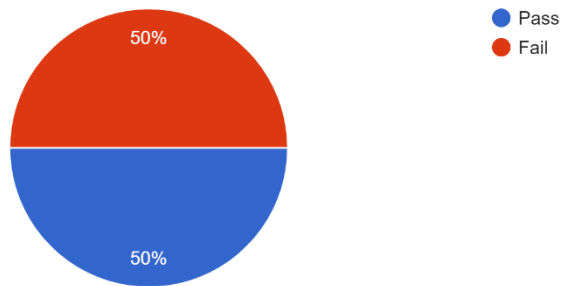
5. The user is able to add the treatment.

2 responses



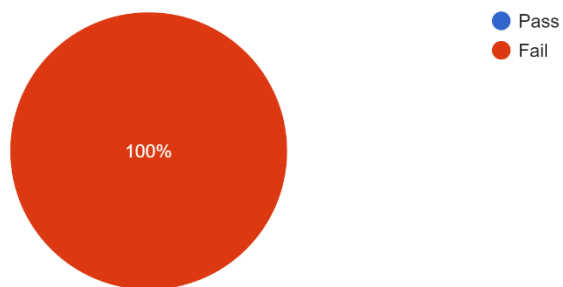
6. The user is able to view all registered treatments and its results in a table.

2 responses



7. The user is able to view treatment details along with the results.

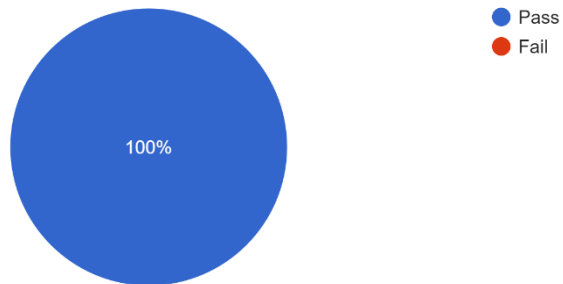
2 responses





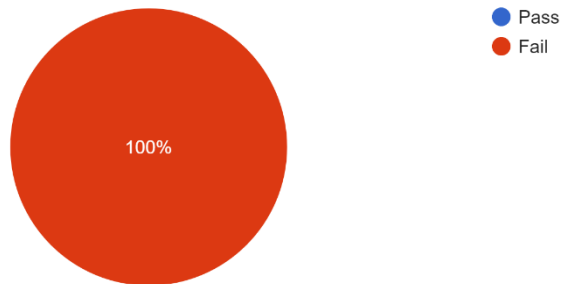
8. The user is able to search for treatments by entering search keyword in the search treatment field.

2 responses



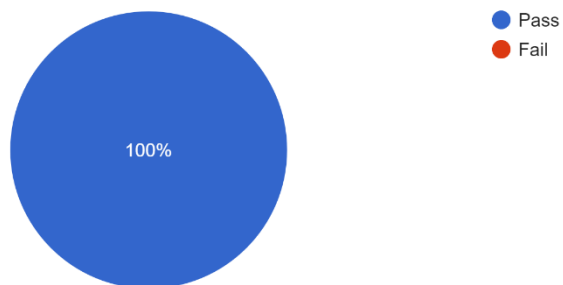
9. The user is able to edit treatment information when needed.

2 responses

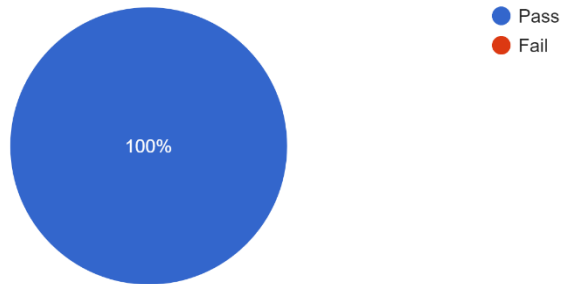


10. The user is able to delete treatment's record after clicking "yes" in the confirmation dialog.

2 responses

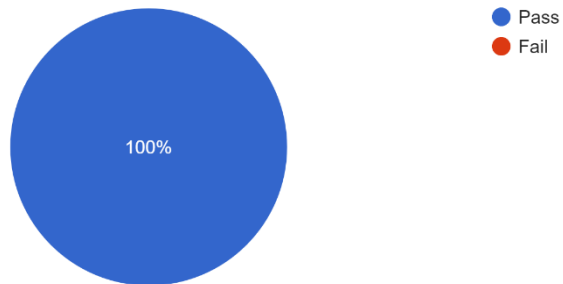


11. The user is able to export treatment's record to PDF, CSV and Excel Format.  
2 responses

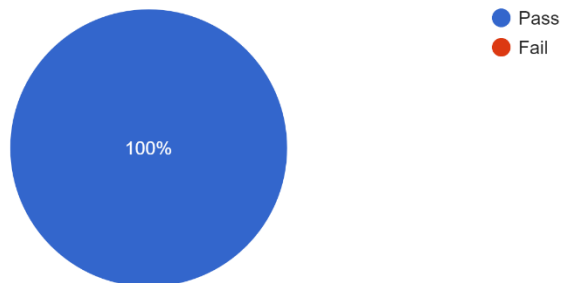


### e) Manage Appointment Module

1. The user is able to make appointment by creating new event.  
2 responses

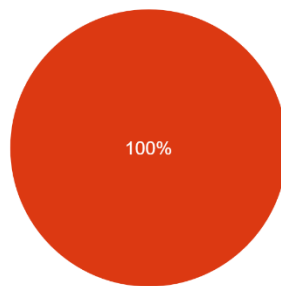


2. The user is able to drag and drop event to different date.  
2 responses



3. The user is able to view event details.

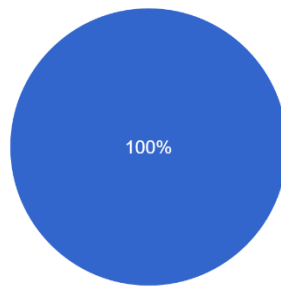
2 responses



● Pass  
● Fail

4. The user is able to edit the event as needed.

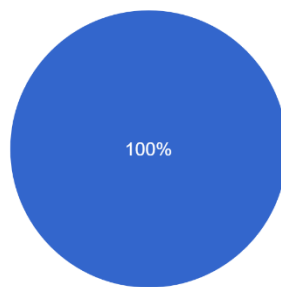
2 responses



● Pass  
● Fail

5. The user is able to delete the event after clicking the "yes" button in the confirmation dialog.

2 responses



● Pass  
● Fail

**APPENDIX B**  
**SOFTWARE REQUIREMENT SPECIFICATION FOR BINOCULAR VISION**  
**MANAGEMENT SYSTEM**


2023

# SOFTWARE REQUIREMENT SPECIFICATION (SRS)

AN AUTOMATED STRABISMUS  
CLASSIFICATION USING CASE-BASED  
REASONING ALGORITHM FOR BINOCULAR  
VISION MANAGEMENT SYSTEM



## DOCUMENT APPROVAL

	Name	Date
<b>Authenticated by:</b>  _____	MUHAMMAD AMIRUL ISYRAF BIN ROHISMADI	20/01/2023
<b>Approved by:</b>  _____  Client		

Software : Microsoft Office 365

Archiving Place : OneDrive

**TABLE OF CONTENT**

CONTENT	PAGE
DOCUMENT APPROVAL.....	ii
TABLE OF CONTENT.....	iii
LIST OF FIGURES.....	iv
1.1 PROJECT DESCRIPTION.....	1
1.2 SYSTEM IDENTIFICATION.....	2
1.3 CONTEXT DIAGRAM.....	2
1.4 DATA FLOW DIAGRAM.....	3
2.1 USE CASE DIAGRAM AND DESCRIPTION.....	4
2.1.1 Use Case Description.....	5
2.2 SEQUENCE DIAGRAM.....	5
3.1 INTERFACE DESIGN.....	10
3.2 HARDWARE AND SOFTWARE SPECIFICATION.....	10

**LIST OF FIGURES**

Figure 1.1	Context Diagram.....	2
Figure 1.1	Data Flow Diagram.....	3



# CHAPTER 1

## 1.1 PROJECT DESCRIPTION

The Binocular Vision Management System is a comprehensive tool for doctors to manage their patients and treatments for strabismus or squints disease. The system is divided into two main components: a web-based application and a machine learning component. The web-based application is designed to allow doctors to interact with the various system modules, including the ability to manage login credentials, doctor profiles, patient records, treatment plans, and appointments. The “Manage Login” module allows doctors to securely log in to the system and access the other modules. The “Manage Doctor” module allows doctors to view and manage the profiles of other doctors in the system, including the ability to add, edit, and delete profiles.

In addition, the “Manage Patient” module enables doctors to view, add, edit, and delete patient records, as well as view and update patient demographics, medical history, and treatment history. The “Manage Treatment” module provides doctors with the ability to view, add, edit, and delete treatment records, along with the ability to perform diagnosis and classification of strabismus conditions by integrating the machine learning component that uses case-based reasoning algorithm. Lastly, the “Manage Appointment” module enables doctors to schedule and manage appointments for patients, including the ability to create new events, view and edit existing events, and view event details.

## 1.2 SYSTEM IDENTIFICATION

The Software Requirement Specification (SRS) belongs to the AI-Powered Binocular Vision Detection System.

<b>System Title</b>	: Binocular Vision Management System.
<b>System Abbreviation</b>	: BVMS
<b>Establish</b>	: 2023
<b>System Version</b>	: Version 1.0
<b>Document Type</b>	: SRS
<b>Developer</b>	: Muhammad Amirul Isyraf bin Rohismadi

## 1.3 CONTEXT DIAGRAM

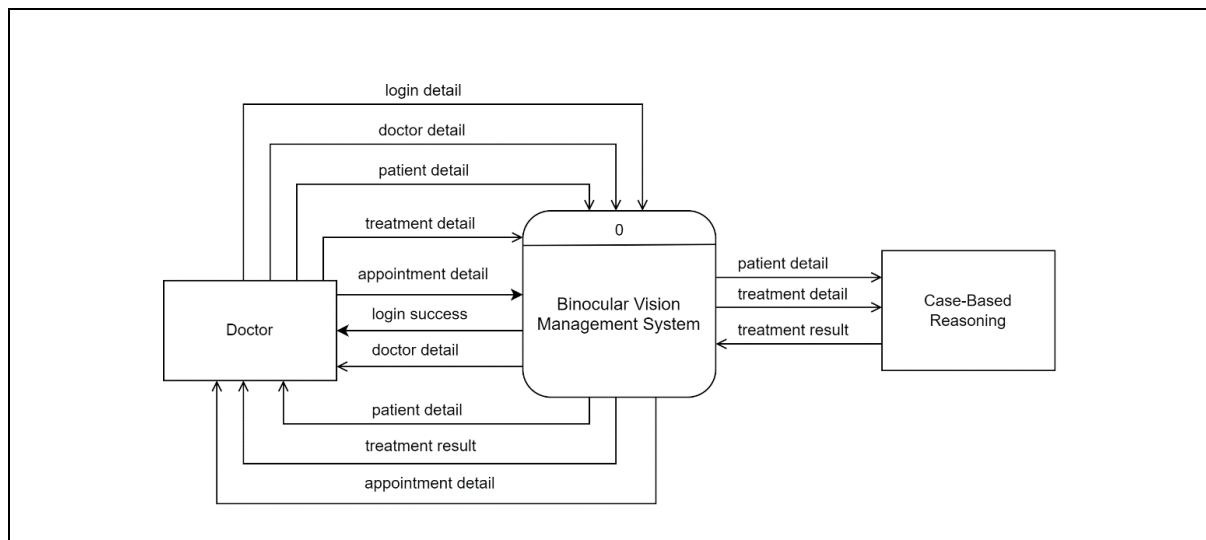


Figure 1.1 Context Diagram

Figure 1.1 shows the context diagram of Binocular Vision Management System that describes the relationship between external modules and internal system. The process is started from the doctor in which the doctor sends the **login detail** the system and the system will respond the process into **login success** indicating that the doctor has successfully login into the system. In addition, the doctor can manage doctor by sending the **doctor detail** into the system and system will output the doctor detail. Then, the doctor can manage the patient by inserting **patient detail** into the system and the system will reply with **patient detail** indicating that the doctor can view the registered patient. Lastly, the doctor can manage the treatment by inputting **treatment detail** into the system and the system will transfer it to the case-based reasoning for

classification process and the case-based reasoning will output the **treatment result** to the system and the system will display the **treatment result** to the doctor.

## 1.4 DATA FLOW DIAGRAM

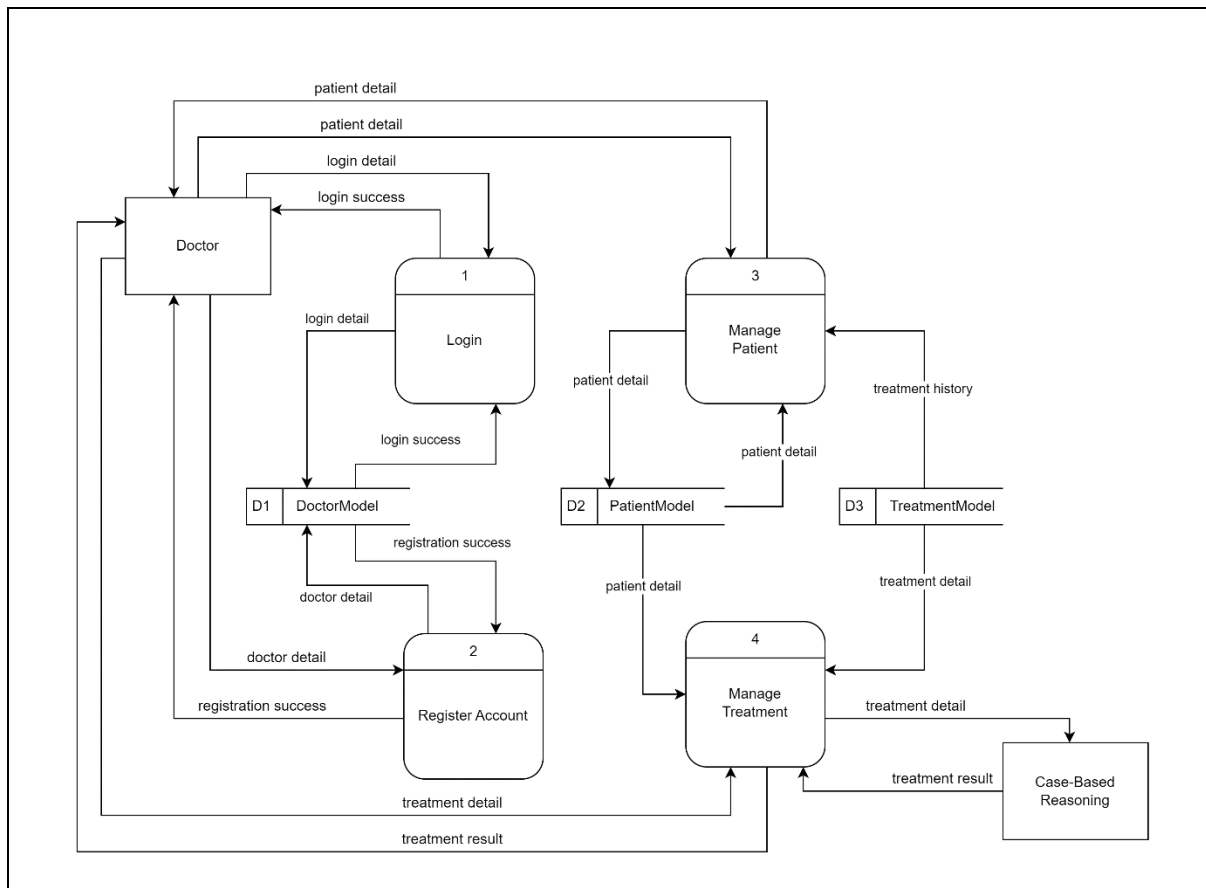
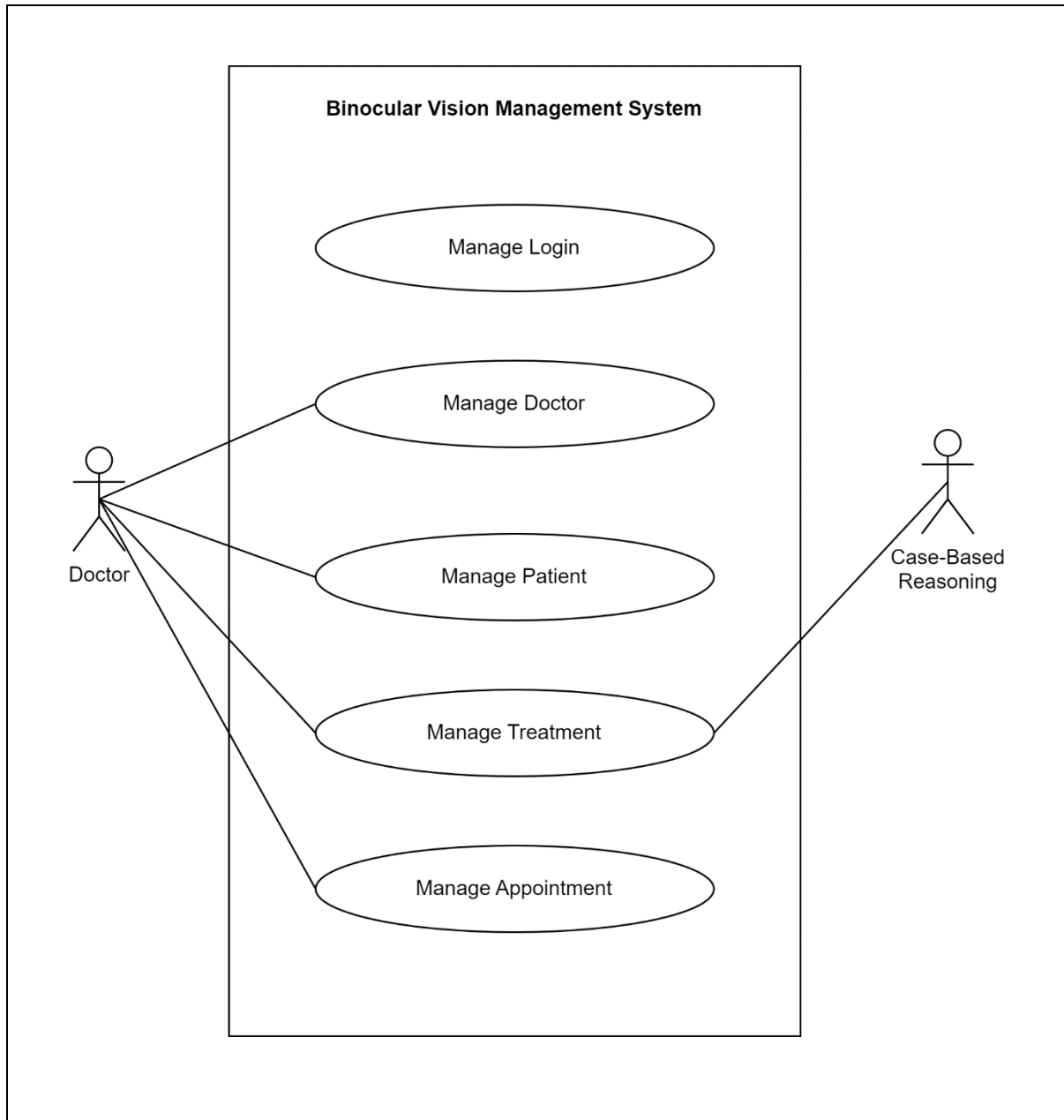


Figure 1.1 Data Flow Diagram

Figure 1.1 illustrates the data flow diagram of the proposed system that is an extended version of the context diagram in the Figure 1.1. From here, the client can see more clearly about the processes involved in the internal system. Login, register an account, manage patients, and manage treatments are the four processes involved in this system. The data stores include DoctorModel, PatientModel, and TreatmentModel which are the database of this proposed system.

## CHAPTER 2

## 2.1 USE CASE DIAGRAM AND DESCRIPTION

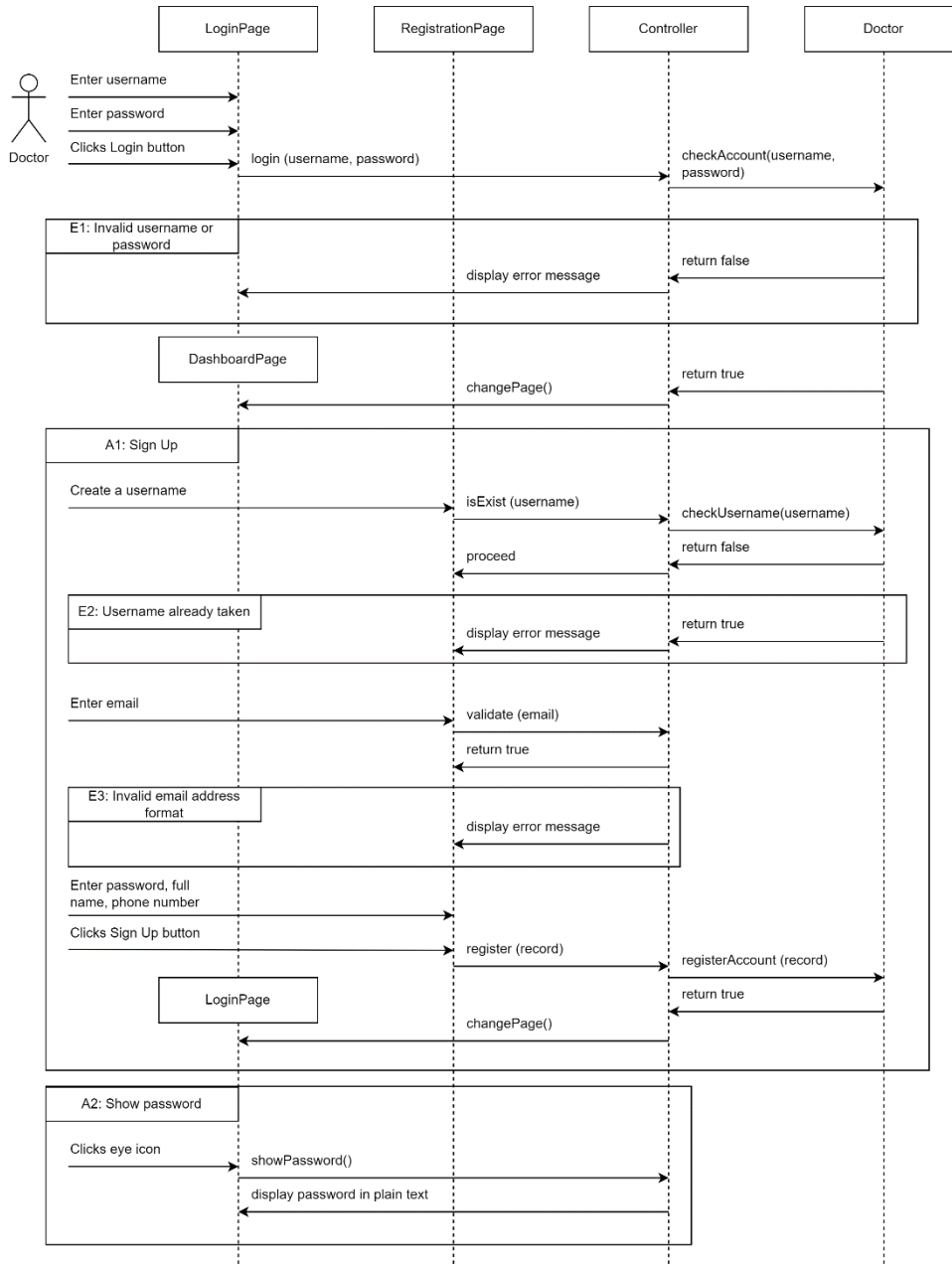


### 2.1.1 Use Case Description

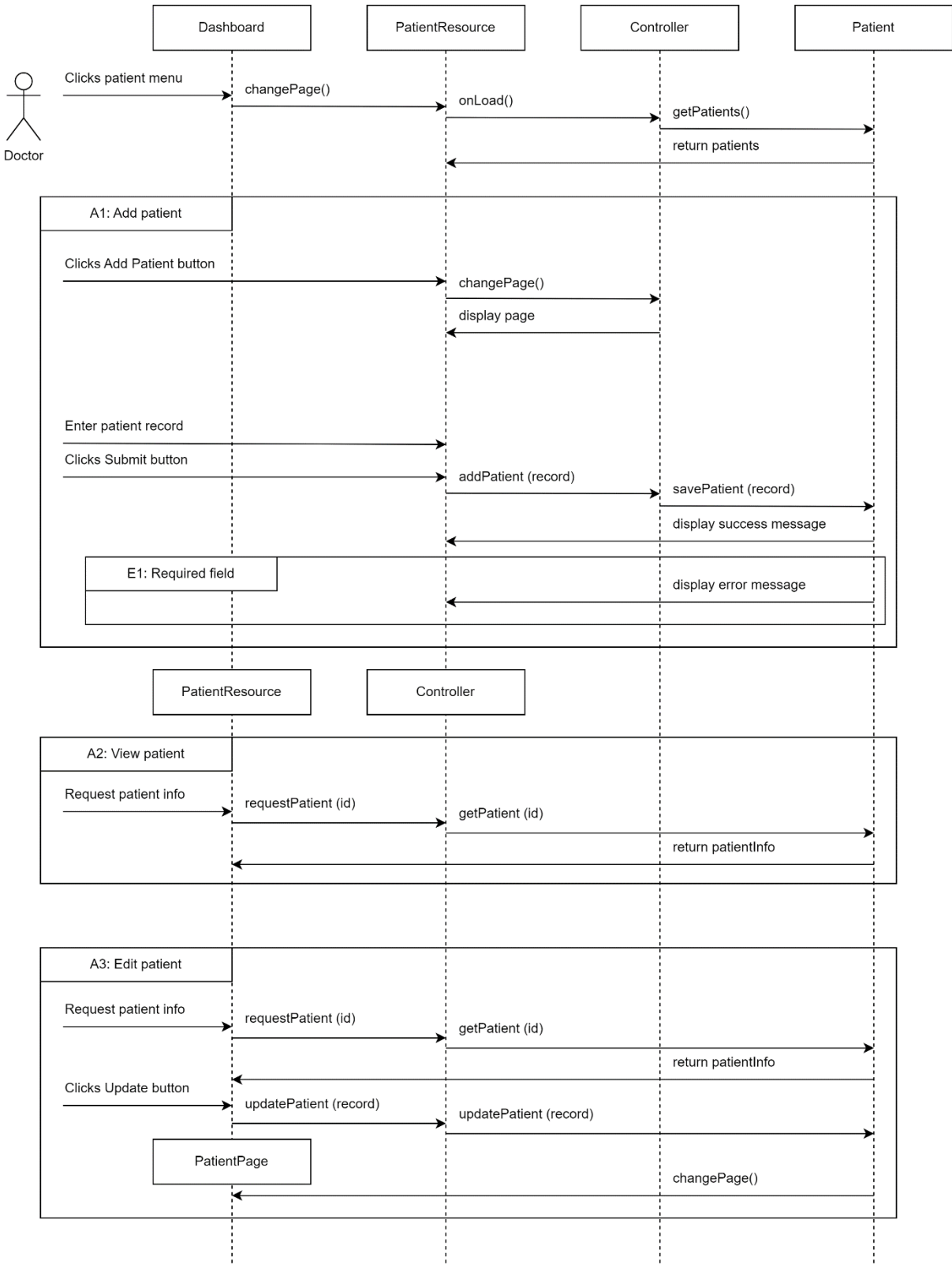
For the use case description, refer to report 3.6.2.1.

## 2.2 SEQUENCE DIAGRAM

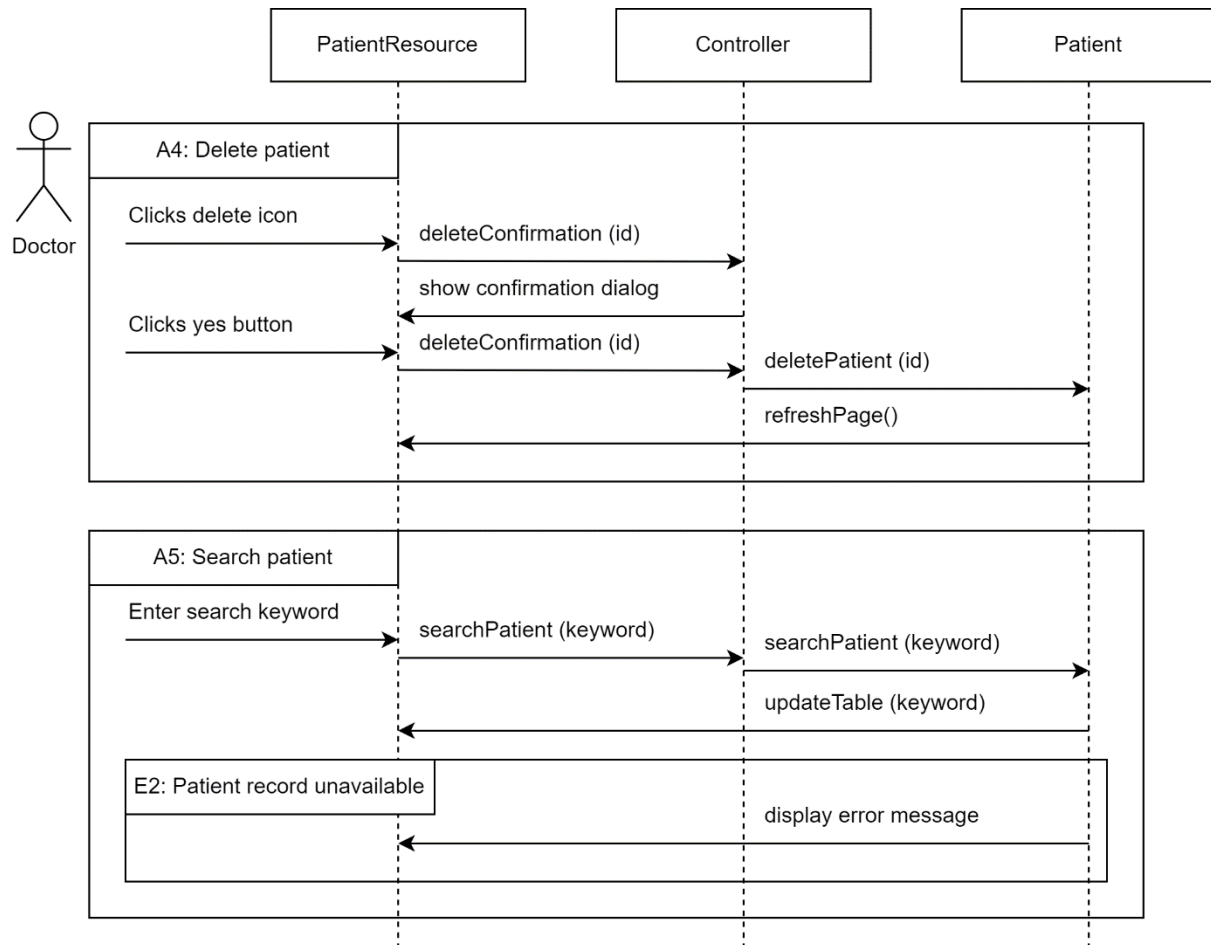
### Manage Login Module



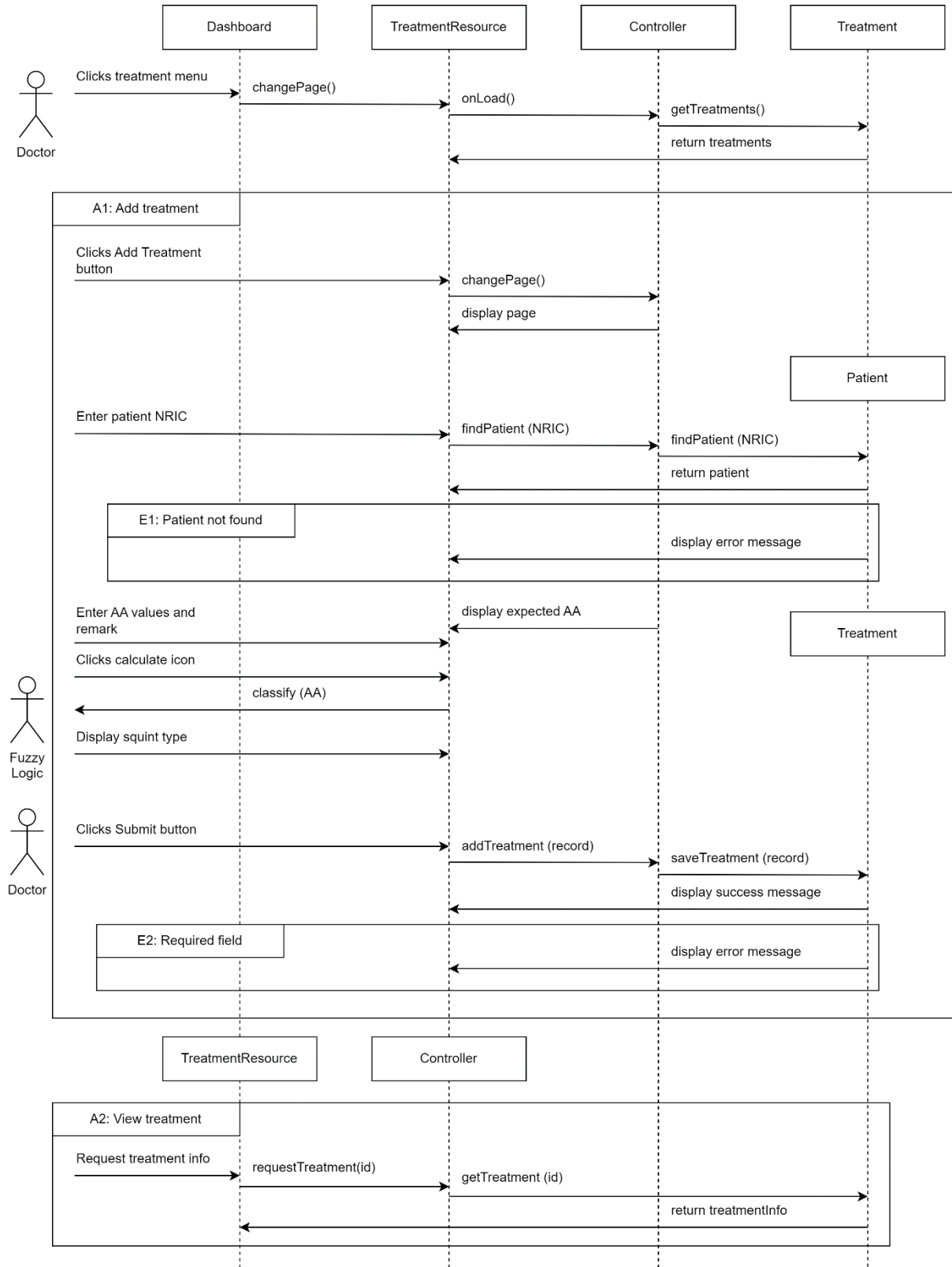
### Manage Patient Module



## Manage Patient Module (cont..)

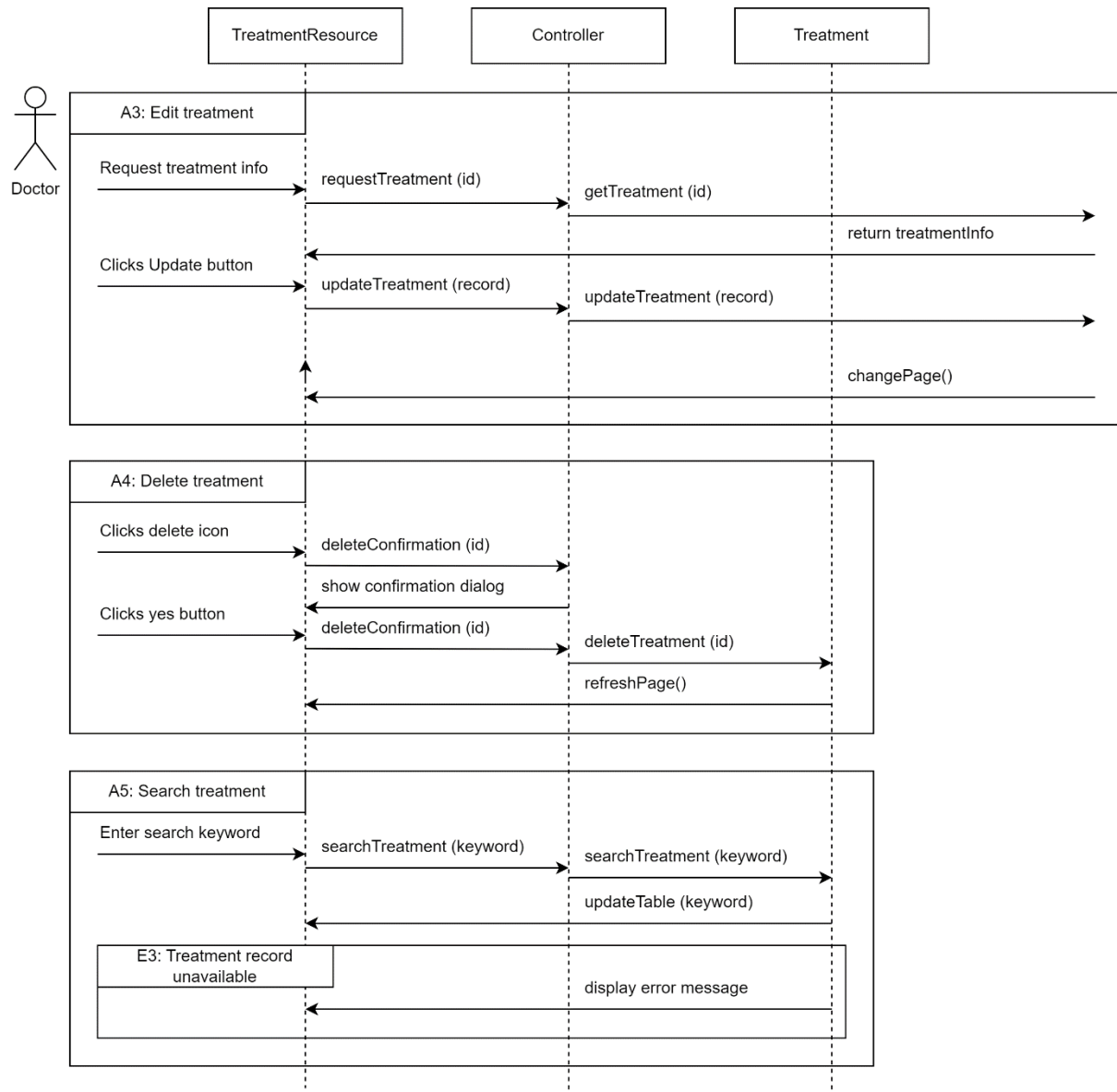


Manage Treatment Module





Manage Treatment Module (cont..)



## CHAPTER 3

### 3.1 INTERFACE DESIGN

*For the interface design, refer to report 3.8.*

### 3.2 HARDWARE AND SOFTWARE SPECIFICATION

<b>Hardware Specification</b>	
<b>CPU</b>	AMD Ryzen 5600H 4,2 GHz, 45W TDP
<b>GPU</b>	Nvidia RTX3060 6GB GDDR6
<b>Memory</b>	16GB
<b>Measuring Tool</b>	RAF Ruler
<b>Software Specification</b>	
<b>IDE</b>	Microsoft Visual Studio Code and Jupyter Notebook
<b>Environment</b>	PHP and Python programming language
<b>Architecture Framework</b>	Laravel Web Artisan, Livewire, NodeJS
<b>Design Framework</b>	AlpineJS
<b>Database</b>	MySQL
<b>Localhost Server</b>	Laragon
<b>Operating System</b>	Microsoft Windows
<b>Web Browser</b>	Brave Browser and Microsoft Edge

**APPENDIX C**  
**SOFTWARE DESIGN DESCRIPTION FOR BINOCULAR VISION**  
**MANAGEMENT SYSTEM**


2023

# SOFTWARE DESIGN DESCRIPTION (SDD)

AN AUTOMATED STRABISMUS  
CLASSIFICATION USING CASE-BASED  
REASONING ALGORITHM FOR BINOCULAR  
VISION MANAGEMENT SYSTEM



## DOCUMENT APPROVAL

	Name	Date
<p><b>Authenticated by:</b></p>  <hr/> <p>Name</p>	MUHAMMAD AMIRUL ISYRAF BIN ROHISMADI	20/01/2023
<p><b>Approved by:</b></p> <hr/> <p>Client</p>		

Software : Microsoft Office 365

Archiving Place : OneDrive

**TABLE OF CONTENT**

CONTENT	PAGE
DOCUMENT APPROVAL.....	ii
TABLE OF CONTENT .....	iii
1.1 PROJECT DESCRIPTION .....	1
1.2 SYSTEM IDENTIFICATION .....	1
1.3 PACKAGE ARCHITECTURE.....	2
1.4 ARCHITECTURE DESCRIPTION.....	4
1.4.1 Manage Doctor.....	4
1.4.2 Manage Patient.....	5
1.4.3 Manage Treatment .....	6
2.1 RESOURCE DETAILED DESCRIPTION .....	7
2.1.1 DoctorResource.....	7
2.1.2 PatientResource.....	10
2.1.3 TreatmentResource .....	12
2.2 ENTITY RELATIONSHIP DIAGRAM.....	14
2.3 DATA DICTIONARY .....	14

# CHAPTER 1

## 1.1 PROJECT DESCRIPTION

The Binocular Vision Management System is a comprehensive tool for doctors to manage their patients and treatments for strabismus or squints disease. The system is divided into two main components: a web-based application and a machine learning component. The web-based application is designed to allow doctors to interact with the various system modules, including the ability to manage login credentials, doctor profiles, patient records, treatment plans, and appointments. The “Manage Login” module allows doctors to securely log in to the system and access the other modules. The “Manage Doctor” module allows doctors to view and manage the profiles of other doctors in the system, including the ability to add, edit, and delete profiles.

In addition, the “Manage Patient” module enables doctors to view, add, edit, and delete patient records, as well as view and update patient demographics, medical history, and treatment history. The “Manage Treatment” module provides doctors with the ability to view, add, edit, and delete treatment records, along with the ability to perform diagnosis and classification of strabismus conditions by integrating the machine learning component that uses case-based reasoning algorithm. Lastly, the “Manage Appointment” module enables doctors to schedule and manage appointments for patients, including the ability to create new events, view and edit existing events, and view event details.

## 1.2 SYSTEM IDENTIFICATION

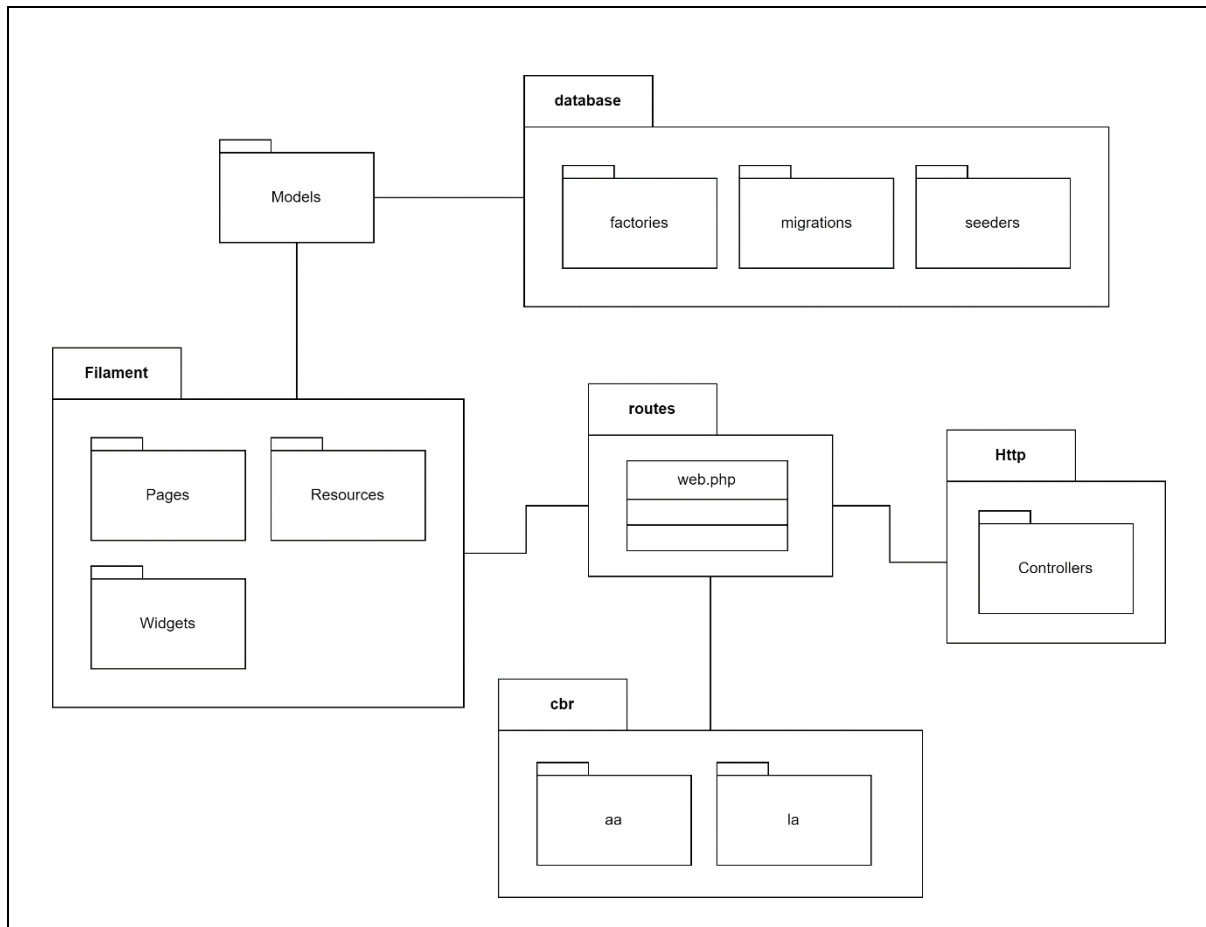
**System Title:** Binocular Vision Management System

**System Abbreviation:** BVMS

**System Identification Number:** SDD\_BVMS\_V1.0 2023

**Version number:** Version 1.0.0

### 1.3 PACKAGE ARCHITECTURE



#### i) **Routes**

Facade of the application. Handle URL requests of the application.

#### ii) **Filament (view package)**

- a) **Pages**: Filament custom pages for viewing dashboard and viewing records.
- b) **Resources**: Filament resource modules for doctor, patient, and treatment.
- c) **Widgets**: Filament widgets for statistics and charts.

#### iii) **cbr**

Case-Based Reasoning algorithm API integrations.



**iv) Http.Controllers**

Contains controllers of the application that act as the middleware between boundary and entity classes.

**v) Http.Model**

Contains entity classes of the application for CRUD operations.

**vi) Config**

Contains various files required for the Laravel application including database, session, auth, and file system configuration.

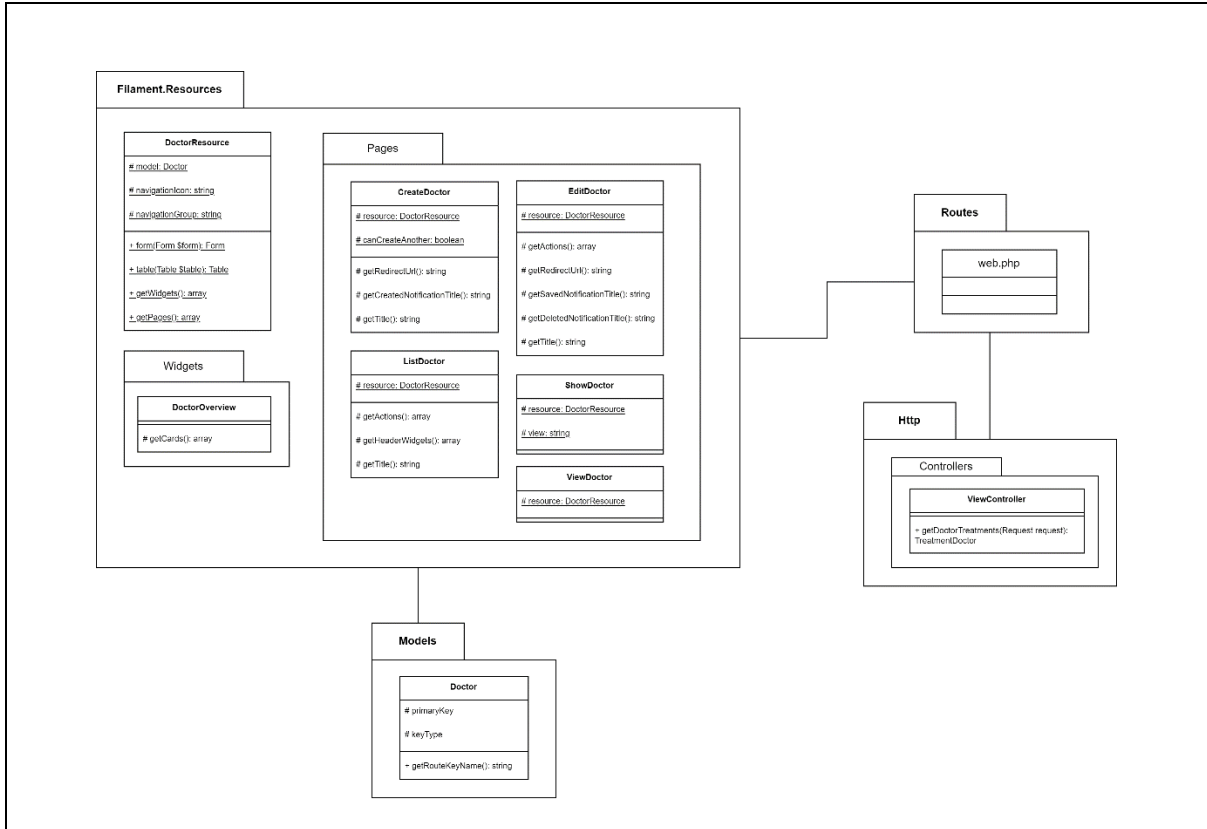
**vii) Database**

Contains database files including factories, migration, and seeds.

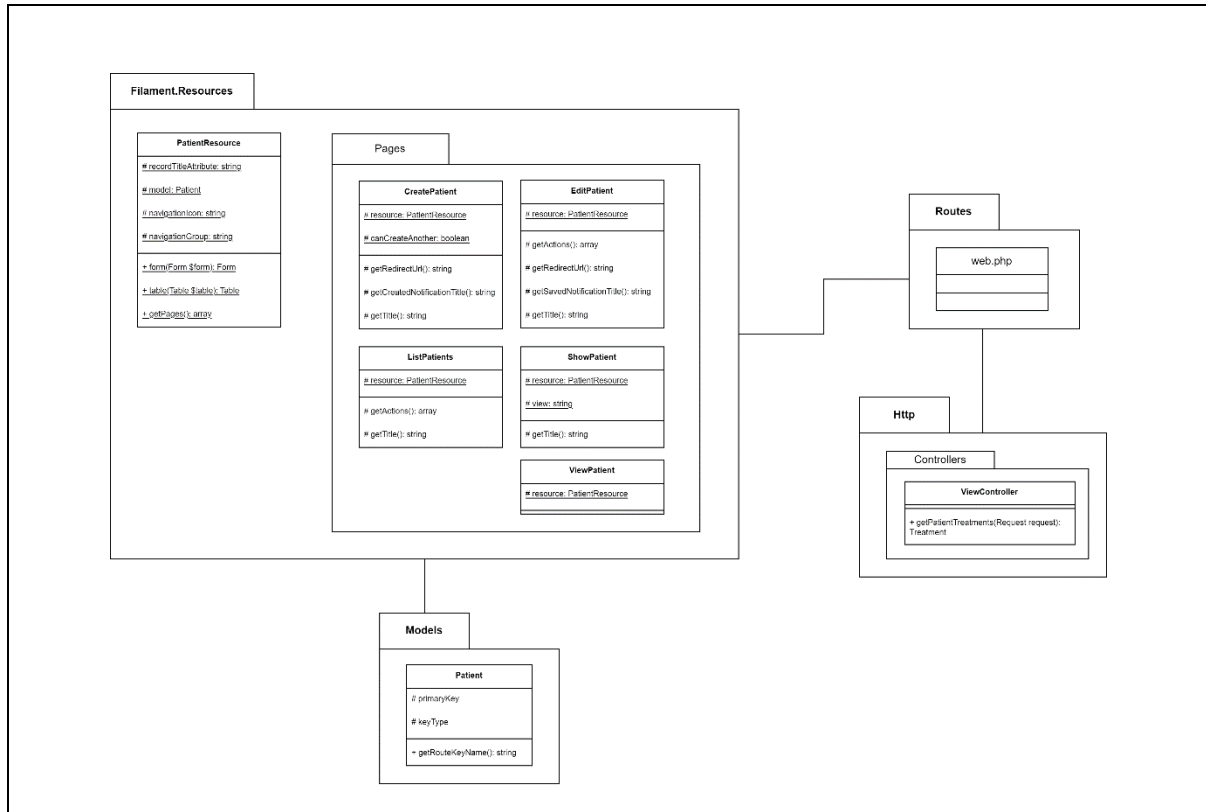
- a) **Factories:** The factories folder is used to generate a huge number of data records.
- b) **Migrations:** The migrations folder is used to migrate the database in web application.
- c) **Seeds:** The seeds folder contains the classes used to perform unit testing database.

## 1.4 ARCHITECTURE DESCRIPTION

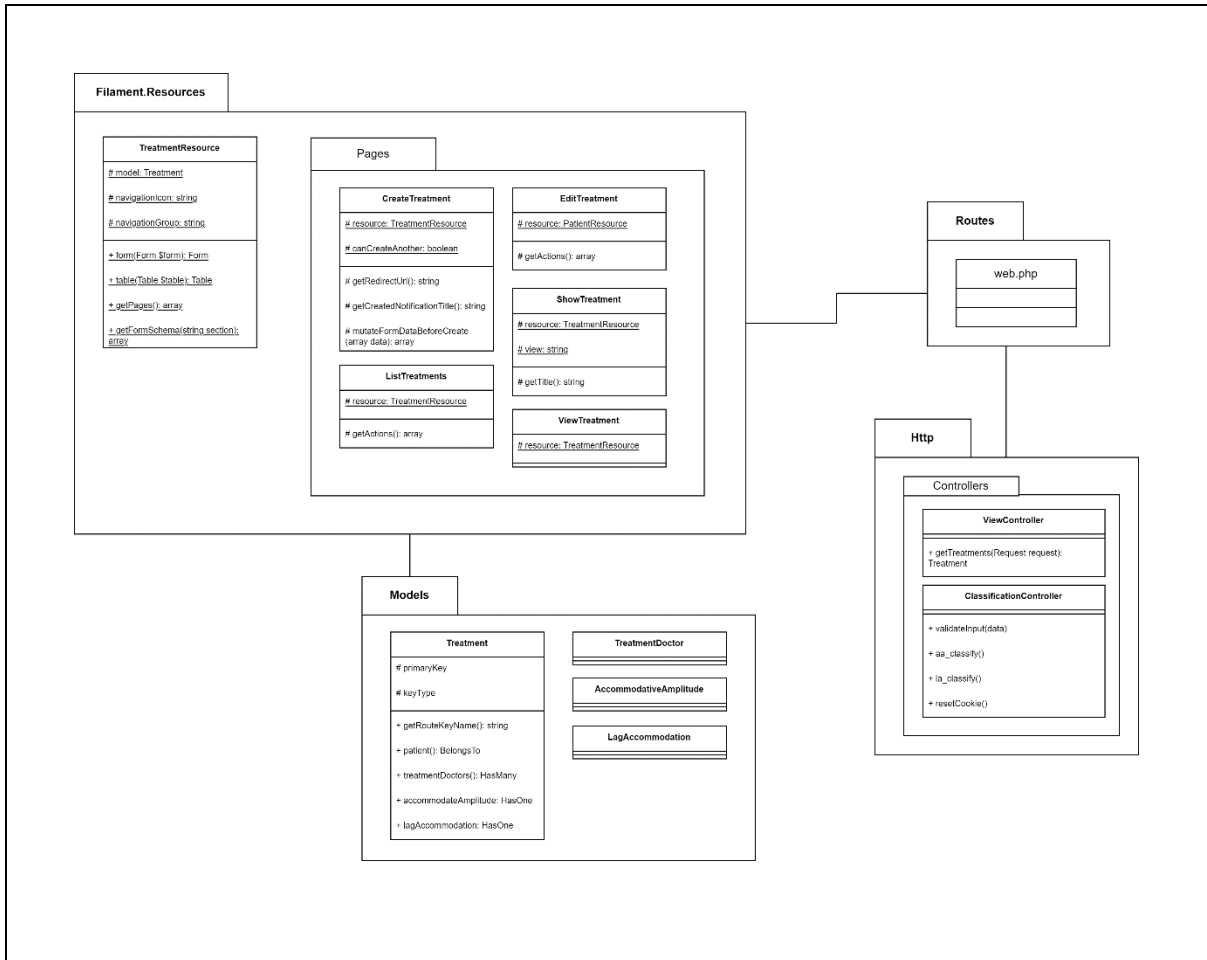
### 1.4.1 Manage Doctor



## 1.4.2 Manage Patient



### 1.4.3 Manage Treatment



## CHAPTER 2

## 2.1 RESOURCE DETAILED DESCRIPTION

## 2.1.1 DoctorResource

Class Type	Filament Resource	
Responsibility	A Filament resource for managing doctor module	
Attributes	Attributes Name	Attributes Type
	model	Doctor
	navigationIcon	string
	navigationGroup	string
Methods	Method Name	Description
	form(Form \$form)	To create form schema for doctor
	table(Table \$table)	To create table for doctor
	getWidgets()	To get doctor widgets
	getPages()	To get all doctor pages
Algorithm	<p><b>FUNCTION</b> form(Form \$form)</p> <p>MAKE FileUpload "profile_picture" with label "Profile Picture", image type and avatar</p> <p>MAKE Grid</p> <p>MAKE Select "Salutation" with options "Dr.", "Mr.", "Ms.", "Prof." and default value and placeholder "-"</p> <p>MAKE TextInput "first_name" with required, max length 50, label "First Name", placeholder "Enter First Name", column span 3 and disable autocomplete</p> <p>MAKE TextInput "last_name" with required, max length 50, label "Last Name", placeholder "Enter Last Name", column span 3 and disable autocomplete</p> <p>MAKE Grid</p> <p>MAKE TextInput "nric" with label "NRIC", numeric, unique (ignoring record), required, placeholder "Enter NRIC Number", mask pattern "000000-00-0000" and disable autocomplete</p> <p>MAKE TextInput "staff_id" with label "Staff ID", placeholder "Enter Staff ID", required, unique (ignoring current record) and disable autocomplete</p> <p>MAKE TextInput "phone_number" with placeholder "XX-XXXXXXX", label "Phone No.", numeric, max length 11,</p>	

prefix "+60", required and disable autocomplete, mask pattern "00-00000000"

MAKE Grid

MAKE DatePicker "date\_of\_birth" with placeholder "Select Date of Birth", display format "d/m/Y", required and reactive, after state updated set age based on date of birth

MAKE TextInput "age" with placeholder "Age", disabled, required

MAKE Select "gender" with placeholder "Select gender" and options "Male" and "Female"

MAKE Select "role" with label "Staff role", placeholder "Select role" and options "Lecturer", "Student" and "Trainee"

MAKE TextInput "email" with required, email validation, label "Email Address", placeholder "Enter Email Address" and disable autocomplete

RETURN \$form with schema containing above components and 2 columns

**END FUNCTION****FUNCTION** table(Table \$table): Table

SET \$table columns as

TextColumn 'staff\_id' with wrap, sortable and label "Staff ID"

TextColumn 'name' with wrap, label "Name", state using function that gets the name using salutation, first\_name and last\_name, searchable using a query that searches for first\_name

TextColumn 'nric' with sortable, searchable and label "NRIC"

TextColumn 'phone\_number' with label "Phone No." and prefix "+60"

TextColumn 'role' with sortable, label "Role" and align center

BadgeColumn 'role' with colors primary for 'Lecturer', warning for 'Student' and success for 'Trainee'

SET \$table default sorting as updated\_at in descending order

SET \$table filters as

Filter 'created\_at' with form containing

Select 'role' with placeholder "Select role"

DatePicker 'created\_from' with placeholder as current date

	<pre>DatePicker 'created_until' with placeholder as current date query that filters based on created_at date range indicateUsing function that returns indicators for created_from and created_until SET \$table actions as View, Edit and Delete actions with corresponding labels SET \$table bulkActions as FilamentExportBulkAction 'export' with disablePreview andtimeFormat "d M y - h:i:sa" DeleteBulkAction RETURN \$table <b>END FUNCTION</b>  <b>FUNCTION</b> getPages(): array RETURN array with routes for 'index' as Pages\ListDoctors with route '/' 'create' as Pages\CreateDoctor with route '/create' 'edit' as Pages\EditDoctor with route '/edit/{record}' 'view' as Pages\ShowDoctor with route '/view/{record}' <b>END FUNCTION</b></pre>
--	---

## 2.1.2 PatientResource

Class Type	Filament Resource	
Responsibility	A Filament resource for managing patient module	
Attributes	Attributes Name	Attributes Type
	recordTitleAttribute	string
	model	Patient
	navigationIcon	string
	navigationGroup	string
Methods	Method Name	Description
	form(Form \$form)	To create form schema for patient
	table(Table \$table)	To create table for patient
	getPages()	To get all patient pages
Algorithm	<pre> <b>FUNCTION form(Form \$form): Form</b>   MAKE Wizard   MAKE Step "Patient"   MAKE Grid   MAKE TextInput "patient_name"   MAKE TextInput "nric"   MAKE Grid   MAKE DatePicker "date_of_birth"   MAKE TextInput "age"   MAKE TextInput "phone_number"   MAKE Select "gender"   MAKE Grid   MAKE TextInput "occupation"   MAKE TextInput "parent_name"   MAKE Section "Home Address"   MAKE TextInput "home_address1"   MAKE TextInput "home_address2"   MAKE TextInput "home_city"   MAKE Select "home_state"   MAKE TextInput "home_postcode"   MAKE TextInput "home_country" <b>END FUNCTION</b>  <b>FUNCTION table(Table \$table): Form</b>   Set the columns of the table to include:   'patient_name' as a sortable and searchable text   column with label 'Name' </pre>	



```
'nric' as a sortable and searchable text column with
label 'NRIC'
'age' as a text column with the value obtained using
a callback function that calculates the age of the
patient using the date of birth
'gender' as a sortable text column
'phone_number' as a text column with label 'Phone
No.' and a prefix of '+60'
'treatment' as a badge column with label 'Treatments'
and aligned center, with the value obtained using a
callback function that counts the number of
treatments for the patient
Set the default sort order of the table to be by
'updated_at' in descending order
Add no filters to the table
Add actions to the table:
'View' with label 'View'
'Edit' with label 'Edit'
'Delete' with label 'Delete'
Add bulk actions to the table:
'export' with FilamentExportBulkAction, with preview
disabled and time format 'd M y - h:i:sa'
'Delete' with Tables\Actions\DeleteBulkAction
Return the table
```

**END FUNCTION**

**FUNCTION** `getPages(): array`

```
RETURN array with routes for
'index' as Pages\ListPatients with route '/'
'create' as Pages\CreatePatient with route '/create'
'edit' as Pages\EditPatient with
route '/edit/{record}'
'view' as Pages\ShowPatient with route
'/view/{record}'
```

**END FUNCTION**

### 2.1.3 TreatmentResource

Class Type	Filament Resource	
Responsibility	A Filament resource for managing treatment module	
Attributes	Attributes Name	Attributes Type
	model	Treatment
	navigationIcon	string
	navigationGroup	string
Methods	Method Name	Description
	form(Form \$form)	To create form schema for treatment
	table(Table \$table)	To create table for treatment
	getPages()	To get all doctor treatment
Algorithm	<pre> <b>FUNCTION form(Form \$form): Form</b>   Step 1: "Patient"   Description: "Choose patient"   Icon: "heroicon-o-identification"   Schema: getFormSchema('section_patient')    Step 2: "Treatment"   Description: "Record medical parameters"   Icon: "heroicon-o-eye"   Schema: getFormSchema('section_treatment')    Step 3: "Doctor"   Description: "Select person in charge"   Icon: "heroicon-o-user-group"   Schema: getFormSchema('section_doctor')    Step 4: "Remark"   Description: "Fill in treatment remark"   Icon: "heroicon-o-book-open"   Schema: getFormSchema('section_remark')    RETURN \$form with columns 1  <b>END FUNCTION</b>  <b>FUNCTION table(Table \$table)</b>   RETURN \$table   SET columns </pre>	

```
TextColumn: treatment_no, sortable, searchable, label 'No.'
TextColumn: patient_name, label 'Patient', using patient_name of related Patient
TextColumn: treatment_date, label 'Date', using formatted treatment_date
BadgeColumn: aa_result_l, label 'Left AA', using switch case to determine 'Normal' or 'Failed' color
BadgeColumn: aa_result_r, label 'Right AA', using switch case to determine 'Normal' or 'Failed' color
BadgeColumn: la_result, label 'LA', using switch case to determine 'Normal', 'Excess', or 'Insufficient' color
SET defaultSort 'updated_at', 'desc'
SET actions
ViewAction
DeleteAction
SET bulkActions
FilamentExportBulkAction, disablePreview, timeFormat 'd M y - h:i:sa'
DeleteBulkAction
```

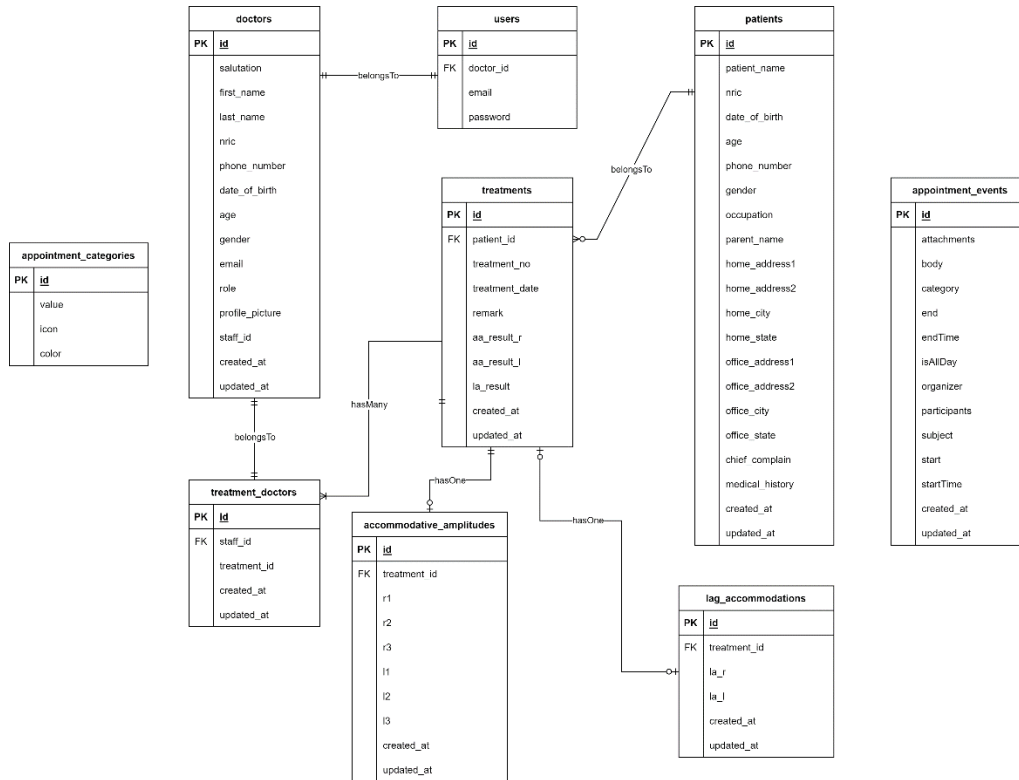
**END FUNCTION**

**FUNCTION** `getPages(): array`

```
RETURN array with routes for
'index' as Pages\ListTreatments with route '/'
'create' as Pages\CreateTreatment with route '/create'
'edit' as Pages\EditTreatment with route '/edit/{record}'
'view' as Pages\ShowTreatment with route '/view/{record}'
```

**END FUNCTION**

## 2.2 ENTITY RELATIONSHIP DIAGRAM



## 2.3 DATA DICTIONARY

For data dictionary, refer to 3.7.2 in report.