Cardiff Metropolitan University

Smart Clinic Management System using RFID

Submitted in May 2019

By

Sujeban Elankeswaran

(Student ID - ST20147493 – JF/BSCSD/03/16)

This dissertation is submitted in partial fulfillment

Of the requirements for the degree of

Bachelor of Science (Honours)

BSc (Hons) Software Engineering

DECLARATION

This work is being submitted in partial fulfillment of the requirements for the degree of **BSc (Hons) Software Engineering** and has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed - ..... (Candidate)

Date - .....

SUPERVISOR'S DECLARATION STATEMENT

Student Name – Sujeban

Supervisor's Name - Mr. Sriharan

I acknowledge that the above named student has regularly attended the meeting, and actively engaged in the dissertation supervision process.

Signed - ……. (Supervisor)

Date - …….....

Coursework Cover Sheet

Student Details (Student should fill the content)				
Name	Sujeban			
Student ID	ST20147493			
Scheduled unit details				
Unit code	CIS6002			
Unit title	Software Engineering Dissertation Project			
Unit enrolment details	Year	3		
	Study period	2018-2019		
Lecturer	Mr. Harshadewa Ariyasinghe			
Mode of delivery	Full Time			
Assignment Details				
Nature of the Assessment				
Topic of the Case Study				
Learning Outcomes covered				
Word count				
Due date / Time	3th May 2019			
Extension granted?	Yes	No	Extension Date	
Is this a resubmission?	Yes	No	Resubmission Date	

Declaration			
I certify that the attached material is my original work. No other person's work or ideas have been used without acknowledgement. Except where I have clearly stated that I have used some of this material elsewhere, I have not presented it for examination / assessment in any other course or unit at this or any other institution			
Name/Signature		Date	

Submission	
Return to:	

Result				
Marks by 1st Assessor		Signature of the 1st Assessor		Agreed Mark
Marks by2nd Assessor		Signature of the 2nd Assessor		

Comments on the Agreed Mark.			
For Office use only (hard copy assignments)			
Receipt date		Received by	

CMU B.Sc. (HONS) SE- ASSIGNMENT FEEDBACK SHEET –ICBT CAMPUS

STUDENT NAME: Sujeban Elankeswaran	STUDENT NUMBER: 20147493
Module Number & Title: 6002	Semester:
Assignment Type & Title:	
For student use: Critical feedback on the individual progression towards achieving the assignment outcomes	

For the Assessors' feedback

Indicate the Task number strength and Weaknesses and the marks for each task

Task No/Question No	Strengths (1st Assessor)	Weaknesses (1st Assessor)

Task No / Question No	Strengths (2nd Assessor)	Weaknesses (2nd Assessor)

Areas for future improvement	
Comments by 1st Assessor	Comments by 2nd Assessor

Marks				
Task /Question No	Marks by 1st Assessor	Marks by 2nd Assessor	Marks by IV (if any)	IV comments (If Any)
Total Marks				

Name and the Signature of the 1st Assessor		Date:	
Name & Signature of the 2ndAssessor :		Date :	
Name & Signature of the IV: (If any)		Date :	

Acknowledgment

I would like to thank Mr. Sriharan my lecturer and supervisor for this project he has supported me throughout the project giving me advice, tips and tricks. He also helped me to finish my project in time. Apart from this I would also like to thank Cardiff Metropolitan University UK and ICBT Campus.

Last but not least I would also kindly thank my friends and family for their kind support throughout completing my project

Table of Contents

List of Acronyms

SCMS – Smart Clinic Management System

JS – JAVA Script

AWS – Amazon Web Server

API – Application Programming Interface

JSON - JavaScript Object Notation

RFID - Radio-Frequency Identification

FIFO – First in First Out

WBS – Work Breakdown Structure

DFD – Data Flow Diagram

ERD - Entity Relationship Diagram

UI -User Interface

DB – Database

MySQL – My Structured Query Language

List of figures

Abstract

This document is all about the Smart Clinic Management System. The system uses the help of RFID readers to digitalize Jaffna Teaching Hospital the (main hospital for the Northern Province of Sri Lanka). The current process of clinic management is done manually where clinic patient brings an 80 pgs. copy with the fear of losing the copy. If the patient is already register for a daily checkup or also called clinic need to give the clinic copy to the attender in order to register for the queue. The given copy would be stacked in FIFO order and separated according to the doctor's specialization. At the doctor's consultation also the treatment and details of the patient and prescription are handwritten on the clinic copy only. There many issues involved in the manual system which is discussed deeply below. To solve this problem I have come to a solution to implement a Smart Clinic Management System where all the patient details are store in cloud storage and the only thing the patient need to bring is the RFID card and in case if it lost there is no problem with small fee the patient can get a new one without worrying about anything. One more solution is reduced waiting time and give the patient more possibilities to do within the waiting time. Since the patient would be notified via SMS before two patients to be ready in the lobby. This help the patient to wait in the outside park go to the nearby shop and etc.

1 Introduction and Background

1.1 Introduction

The Smart Hospital Management System using RFID is a digital process of the current clinic management in Government Hospital of Jaffna.

The current manual process in the Government Hospital Jaffna is that the patient must file their clinic book in a stack method (First in First Out). In a day there would be two times where the patients can go for a check-up one in the morning and one in the afternoon. Approximately more than 400 people will gather for a check-up for morning and afternoon check-up. And the clinic book is not computerized they have it all in a paper. This is not secure from natural disaster and not secure from the theft. This can also cause misuse of clinic books because any person can write on the clinic book without any evidence. In some cases, we also identified that there is an issue in identifying the prescription from the doctors by the pharmacists, this could be dangerous for the patient.

To solve the above problem, I have come up with an idea of a smart clinic management system. This system enables the patient to scan on the RFID reader which is also a stack method (First in First Out). After scanning the card, the patient receives an SMS which will show the token number and approximate waiting time. If the patient has enough time he can go out to a nearby shop or sit outside the hospital park than waiting inside the hospital. A second message would be sent to the patient that he would be the next to go to the doctor after three patients. And a third message which shows the room number that the appropriate patient must visit. Soon after the patient is called the appropriate doctor would see all the patient information on the computer and the doctor can even greet the patient with the name to make the patient more comfortable and to gain trust. Inside the system, the doctor can see all the past information of the patient including all the X-ray, Blood report, Cholesterol test, Urine test and etc. This helps the doctor to find the right treatment and cause much easier. Once the doctor has found the cause he can prescribe the medicals on the system itself. After that, the patient would go to the pharmacy and before the patient arrives at the pharmacy counter the pharmacist would have prepared all the medicals for the patient and even here, we can save the patient waiting time and avoid the issue of not understanding the doctor’s prescription. The doctor prescription would be sent

as an API request from the Smart Clinic Management System and integrated with their pharmacy application system.

1.2 Background

Jaffna Teaching Hospital is a full government hospital in Jaffna, Sri Lanka. This teaching hospital is a leading hospital in the northern province of Sri Lanka which is controlled by the central government in Colombo. It also acts as the main clinic teaching faculty for the University of Jaffna Faculty of Medicine. Apart from this hospital has 1236 beds, 21 ICU beds, 61 special units, 13 operating rooms. Jaffna Teaching Hospital consists of 1500 staffs which are working 247/7 in three shifts. Out of 1500 staffs whom 76 are a consultant in their special fields. In a daily basis the hospital treats around 4500 patient a day. This hospital has many health service such as Anaesthesiology, Cardiology, Chemical Pathology, Dental, Dermatology, Diabetes & Endocrinology Unit, ENT, Gastroenterology, Haematology, Histopathology, Judicial Medical Unit, Medical Unit, Microbiology, Nephrology, Neurology Unit B, Neurosurgery, Obstetrics and Gynaecology, OPD, Ophthalmology, Oral & Maxillofacial Unit, Orthopaedic, Paediatric, Plastic Surgery, Psychiatric, Radiology, Respiratory Unit, Rheumatology, Surgical Unit and Urology. Since it is the leading hospital in northern peninsula it is 24/7 open. (Jaffna, 2019)

In Jaffna Teaching Hospital we can go for a clinic consultation in morning and evening. As already stated, this is the leading hospital in the northern peninsula with a population of 1,058,762 (Anon., 2012) peoples. This amount of population may come for the Jaffna Teaching Hospital for special treatments. It has also the facilities to go for a monthly or weekly check-up called clinics. The daily number of patients attending for the clinic would be around 100 or more people they all go for different doctors who are specialized for the type of health issues. Currently, the patient must file the clinic card where all the patient past records are recorded. The clinic files taken to the queue of FIFO (Frist In First Out) approach. After a certain time, the attended would call the patient according to FIFO to manage all the patient to write doctors more the five attendees or nurses needed to manage the crowd efficiently. Since the patient doesn’t know how long and after how many patients, they would be called they have to sit near the clinic's room which usually uncomfortable because it

could take more than 2 hours in a busy day. And the keeping patient data in a piece of a small copy if the not correct way in the digital age. To overcome these two issues, I have come up with a web application that would solve the problem.

1.3 Literature review (Areas Depth)

In research paper called "An assessment of patient waiting and consultation time in a primary healthcare clinic" states that patient waiting for the doctor is common is issue around the world but however the waiting time differs with various factors such as whether the particular country is developed or not, government hospital or private hospital and etc. This particular research was conducted for 4 weeks of audit in a primary health care clinic. They state that according to their analysis a patient must wait 41 minutes on average until he gets consulted by the doctor. And the average consulting time would take 18.21 minutes. If break down the 41 minutes process there are a number of processes such as time for registration, pre-consultation, appointment and etc. The primary clinic has an average of 60 patient attendances per day. They have their own Queue Management System which basically records the time in and out of every station. A diagram of this research paper states 91.93% of the patient wait less than 90 minutes. And the consultation time is for almost all the patient is less than 30 minutes. (Ahmad BA, 2017)

Overall according to the researchers, they say that clinic waiting time must be improved to give the best service to the patients. According to the above problem, I can reduce the number of waiting time with Smart Clinic Management System this system eliminates the time for an appointment, the time for registering, and even the time at the pharmacy because right after doctor consultation prescriptions are passed to the pharmacist to make ready all the medicals for the patients.

Another research paper called **"Outpatient Waiting Time in Health Services and Teaching Hospitals: A Case Study in Iran"** states that Iran's educational hospital takes an average time of 161 minutes for the patients which is more than the above research paper. According to their research paper, it states that between the age of 26-40 has the highest distribution of age patients. In this researched hospital they use the FIFO approach for a queueing system which usually done in every hospital. To

provide the best service to the patients is very important that researchers states. According to the research they have found that there are three main factors that affect the waiting time of patients such as registration process, shortage of physician and shortage of professional staff. (Rafat Mohebbifar, 2014)

In summary, the above research paper shows there is a huge waiting time for the patient which is 161 minutes which is too long for the patient and this not called quality service. These times are affected by the registration process, shortage of physician and shortage of professional staff. In the Smart Clinic Management System has a great advantage because the patient registers only once and after that, they just need to scan the RFID at the RFID reader to register therefor, we reduce almost 15 minutes already and staff is also reduced. Since the system has been implemented with SMS gateway the patient actually can go nearby shop or outside the hospital garden to relax. Once the patient is the next after two patients, he gets an SMS saying that he needs to wait at the lobby. Shortage of physician is a problem that has to be solved by hospital management itself and the same as professional staffs.

Another research paper titled **"Booked inpatient admissions and hospital capacity: a mathematical modeling study".** This paper tells that unlike other operation, hospital operation is most complex because it always has to do with high capacity and overload. Three factors make very difficult to manage the operation such as variability, blocking, reserve capacity, unpredictable variability. (Steve Gallivan, 2002)

The above paper highlights that hospital operation is very complex which is true and since almost all hospital have a great capacity of patients coming to the hospital, they need more staff to handle the process. However, since the above paper tells about booking system in a hospital the factors that affect the operation applies to the Clinic Management too.

1.4 Objective

The objective of this project is to **reduce clinic patient time by 15 min** and to **digitalize the manual process** of clinic management in Jaffna Teaching Hospital.

1.5 Scope

The scope of the smart clinic management system is to enable the clinic patient to attend the doctor consultation much **faster** than before and to **digitalize the current process** to make **patient information more reliable and secure** for future requirement.

1.6 Limitation

1.6.1 Sample size

Gathering samples from the government hospital is a limitation since the sample data are from the patient which is unacceptable to share the private information. Therefore, gathering sample size to test the application is limitation instead of this information we need to test with fake information's.

1.6.2 Equipment

Since the cost of high-end RFID Reader/Writer is expensive I am going with a cheaper device that has the same functionality only the lifetime and reading distances might differ.

1.7 WBS

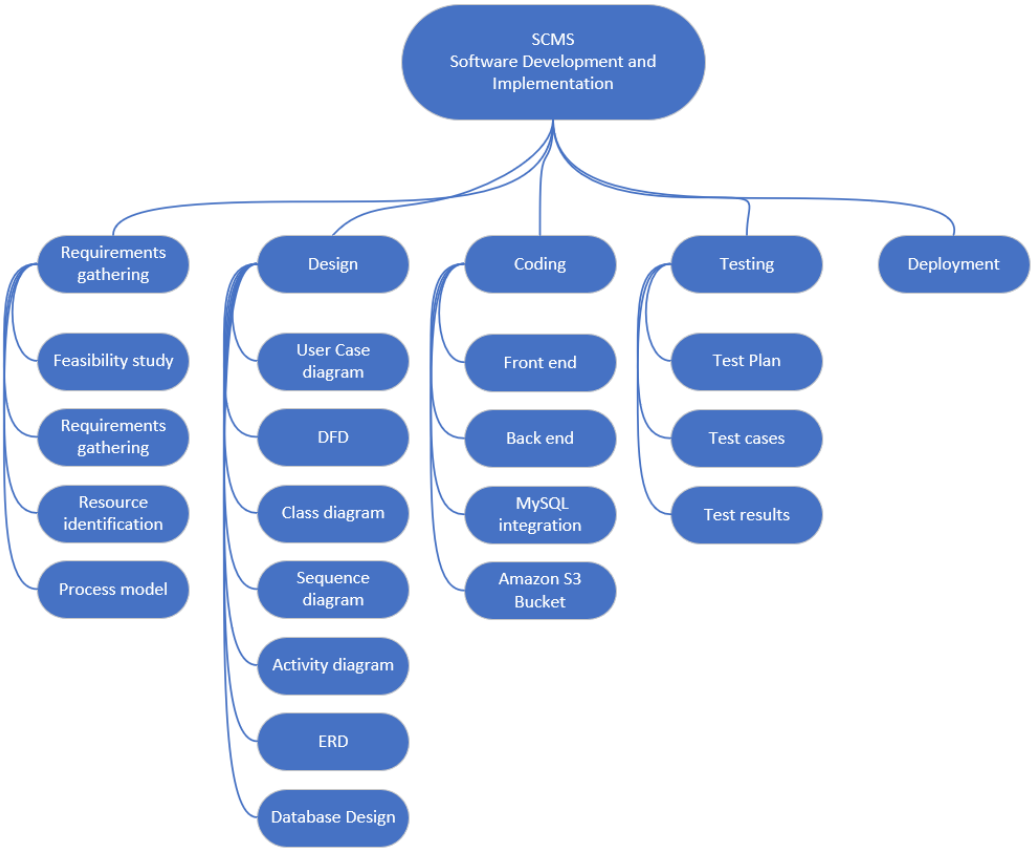


Figure 1 Work Break Down Diagram for SCMS

The above WBS Work Breakdown Structure is to show the main deliverables that I need to finish in order to build this system successfully. Basically, it is breaking a huge work into a smaller segment for completing in a smaller task as the name itself says.

Above we can see 4 main segments which are requirements gathering, design, coding the actual phase to build the application, testing the application and deployment instruction. All the main segments have smaller subsegments. For the Requirement gather task, we broke down into smaller task such as feasibility study, requirement gathering, Resource identification, and process model. The next phase is designed task which is broken down into eight tasks such as use case diagram, DFD, Class diagram, sequence diagram, Activity diagram, ERD and database design. The third phase is coding. In this phase, the actual development of the application starts to make it simple we have broken down in smaller task such as front end, backend, MySQL

integration, and Amazon S3 integration. After development, we need to test the application to validate that all the function and feature are working properly as they should do.

The last phase is the deployment task since the task is already small we cannot break it down further, therefore, we leave it as it is.

1.8 Resource allocation

#	Task	Optimistic (Hours)	Most likely (Hours	essimistic (Hours	Effort (hours)		Start date	End date	Status	Actual End Date	Key Dependency
1	**Initializing**				**15.33**		**24 - Feb**	**28 - Feb**			
	Prepare all the needed diagrams	7	10	10					Open		
1.1	Database Mapping (ERD)	5	8	10	7.83		3 - Mar	4 - Mar	Open		
1.2	Database Implementation	3	8	10	7.50		5 - Mar	5 - Mar	Open		1.1
2	**Architecture**				**15.00**		**1 - Mar**	**2 - Mar**			
2.1	System Architecture (Diagram)	8	16	18	15.00		6 - Mar	7 - Mar	Open		
3	**Development**				**#REF!**		**2 - Mar**	**20 - Jun**			
3.1	**Front-end development**				**88.00**		**2 - Mar**	**13 - Mar**	Open		
3.1.1	AngularJS environement setup								Open		1,2
3.1.2	Login UI								Open		1,2
3.1.3	Dashboard UI								Open		1,2
3.1.4	CRUD Doctor								Open		1,2
3.1.5	CRUD Patient								Open		1,2
3.1.6	Adding all kinds of document of patient								Open		1,2
3.1.7	RFID Setup and Integration								Open		1,2
3.1.8	Patient Real-time Queue Algorithm								Open		1,2
3.1.9	Allocating Patient to Doctor (Algorithm)								Open		1,2
3.1.10	Real-time waiting list display for outdoor monitor								Open		1,2
3.2	**Back-end development**				**200.00**		**18 - Mar**	**11 - Apr**	Open		
3.2.1	Spring boot environment setup								Open		1,2
3.2.2	Cloud integration								Open		1,2
3.2.3	User Login and JWT Token Integration								Open		1,2
3.2.4	Bodylocation API								Open		1,2
3.2.5	Issue API								Open		1,2
3.2.6	Symptom API								Open		1,2
3.2.7	Patient Test Result API								Open		1,2
3.2.8	Prescription API								Open		1,2
3.2.9	Clinic API								Open		1,2
3.2.10	Patient API								Open		1,2
3.2.11	Doctor API								Open		1,2
3.2.12	User Role								Open		1,2
3.2.13	Validation								Open		1,2
4	**Testing**				**32.00**		**14 - Apr**	**18 - Apr**	Open		
4.1	Error handling	10	16	22	16.00		3 - Apr	4 - Apr	Open		1,2,3
4.2	Performance testing	10	16	22	16.00		5 - Apr	6 - Apr	Open		1,2,3
5	**Implementation**				**8.00**		**18 - Apr**	**22 - Apr**			
5.1	Software and Hardware Requirement	1	2	3	2.00		7 - Apr	7 - Apr	Open		1,2,3
5.2	Deployment Instruction	1	2	3	2.00		7 - Apr	7 - Apr	Open		1,2,3
5.3	System Installation Guide	2	4	6	4.00		7 - Apr	7 - Apr	Open		1,2,3

1.9 Milestone Plan

#	Milestone(s)	Planned(End) Date	Status	Actual Date	Comments
1	Initializing	24 - Feb	Open		
2	Architecture	1 - Mar	Open		
3	Front-end development	2 - Mar	Open		
4	Back-end development	18 - Mar	Open		
5	Testing	14 - Apr	Open		
6	Implementation	18 - Apr	Open		

1.10 Cost Plan

No.	Item name	Price (£)
1	RFID Reader/ Write (1x RFID card)	8.54
2	Laptop	300
3	Hosting (one month)	2.14
4	Cloud storage (Amazon Web Service S3 Bucket)	Approx. 5
5	Domain (one month)	7
6	For 126 SMS (SMS Gateway)	8.95
	Total	331.63

2 The System

2.1 What kind of a system is it?

The system is called a Smart Clinic Management System it is basically a web application. This System will contain all the information of the patient who is coming for a regular check-up. Each patient would have an RFID card for identification purpose. The system stores all the information about patient treatment, prescription, reports, scans and etc. This will enable the doctor to have all the information in one view to give better treatment and this information would also help to identify the disease that is increasing the country in order to take appropriate action as soon as possible. The patient receives an SMS containing the token number and the name of the room that he needs to visit. When entering the correct room, the doctor would automatically see the patient information on the desktop and once he leaves the room the doctor would prescribe the medical on the system. After that, the patient would go to pharmacist and scan once again to get the prescription list from the system. The patient's information is stored entirely on the cloud which is more secure and less expensive. Moreover, the important part of this system is the queue system. The queue system enables the patient to scan his RFID card on the reader to register in the queue and he could wait in the lobby or outside the park after specific time like before three patient a message would be sent to the patient to wait in the lobby for doctor consultation. This enables to get rid of long queues in the bench and it also gives equality among public usually if the staff is known for a patient, he would give priority and allow him first, therefore, this system could solve the problem while it is fully digitalized.

2.2 Why is this system important?

This Smart Clinic Management System is important to **solve long-standing queue** in Jaffna Teaching Hospital because all the patient who comes to the clinic are aged patient and according to their disease, they need to get food and drink on time, therefore, **reducing the waiting time** is a big movement for the sake of the patient. Moreover the patient carries a small 80-page copy to record their clinic details and patient test result are also held by patient this **causes loss of copy and test result** when the patients are aged and might **easily forget the clinic copy or lose**

eventually, therefore, I also implemented add clinic and patient test results into Smart Clinic Management System to solve this issue. Another important thing is that the patient can get reports of an emerging disease in a particular hospital which helps full to take appropriate actions for the government. Implementing this system can also eliminate other issues such **as duplicate data, lack of security, unproductive by implementing a large number of staffs, risk of data and the possibility of giving wring medicals to the patient because of unclear prescription of a doctor**. Considering all this factor it could be imagined why the SCMS so important.

2.3 User Requirements

The user requirements of the Smart Clinic Management System are that all the eight users need to have specific permission to change only the appropriate information's. In this system, we have many user requirements which are listed below according to its user type.

Admin – The admin has the privilege to create, edit, update and delete users in the system.

Director – The director has permission to view the hospital reports and staff details.

Doctor – The doctor has permission to view the patient's full information and view patient test result. More importantly, he is the only one user type that has the privilege to create a clinic session for a patient.

Laborist - The laborist has the permission to view patient test result needed list and to upload patient test result to the system

Attender – The attendee has the feature to create a new patient for the clinic by validating all the real evidence.

Nurse – The nurse has the privilege to view the patient clinic information only.

RFID – The RFID is not a real person who is going to take control of the system it just used for the RFID reader and to show the Real-Time Queueing system.

Pharmacist – The pharmacist has only the privilege to get prescription by id only which will be used to integrate with the existing Pharmacy Management System.

2.4 Current manual system explanation

In Jaffna Teaching Hospital the outside patient comes for a daily clinic check up to the respective doctor. The patient carries a small 80-page copy which is provided by the Jaffna Teaching Hospital. When the patient wants to register to consult to a doctor he has to approach to an attendee and say to him that he needs to visit the doctor. The attendee then takes the patient's clinic copy and files in a FIFO (First in First Out) method and according to the doctor's specialization. Once all the doctors have arrived the attendee will start to call out the patient's names in a FIFO manner. Until this process, the patient cannot go anywhere other than sitting on the bench for hours. Finally, when the time has come to visit the doctor we have to go to the particular doctor and the attendee then gives the patient clinic copy to read previous clinic session and to write today's clinic session. Once the doctor has gone through consultation the patient might get prescribed some medicine on the clinic copy and some other information's. The medicine can be taken by the pharmacist which is inside the hospital by providing the clinic copy there are issues with the doctors handwriting I have heard from some pharmacist that they are unable to read the doctor handwriting which is really not a good sign for the patient because the pharmacist might give wrong medicine accidentally due to the worse handwriting.

2.5 Drawback of the current manual system

To go for a clinic in a Government Hospital of Jaffna we need to file all the patient's clinic book one by one in stack method and the patient are called in the same order. This, however, could take more than two hour waiting time.

Many people who are working in the weekday usually find very difficult to go for a clinic because of the long waiting time. And this generation of people usually doesn't waste their time on waiting. And overall the current process of clinic management is manually done. Each and every patient has its own clinic book where all the details are entered by the doctors.

In case of lost clinic book, it very difficult and time-consuming to get all the records back from the manual process. And the patient previous reports and scan are giving to

a patient only and if lost it is difficult and costly to get all the new test done. Due to the manual process, it could lead to duplicate data, space consuming, lack of security, unproductive, misplacement and risk of data.

The duplicate data might occur when creating a new patient file because the patient has lost his clinic book therefore, they would have two files of the same patient. All the patient information is recorded in a book and for doing this it will take many books while going through all these years. Therefore, having the recorded data in a book is very space consuming. Lack of security manual process is mainly considered as a lack of security because anyone can secretly access the patient record and change the way they want. Managing this clinic manually is considered as unproductive because more labors are needed for manual management. Documents can be easily misplaced in a hurry or tension. Risk of data is that paper document could be easily accessed therefor it is the risk of personal data.

Another issue is that I have encountered at the government hospital is that the doctor's prescription letter is sometimes difficult to read by the pharmacist this leads to a dangerous problem which is giving the wrong medicals to the patient that can cause serious side effect.

2.6 Explanation of the proposed system to solve the problem

This problem can be solved with the help of available technology such as with the help of web application we can create a management system to manage all the patient who comes for the clinic. Waiting time can be made much comfortable by sending the approximate waiting time and send SMS to the patient before that they are the next to see a doctor after three patients.

The manual process can be digitalized by storing all the information in the cloud. This will enable us to keep the information safe forever. It is kept safe from any natural disaster and theft. The system cannot be misused because for each and every time someone accesses the patient personal data, it would record and store a log file with information who, when what and from whom he accessed the information. And even all the changes are recorded to detect misuse and investigation easier.

With this system we can avoid duplicate data because searching for a patient is much easier because of we able to search through a query. Having all the information stored in an AWS cloud storage it would take less space than before. While we have all the information on the cloud, we can assure that the data is in a secure place and the system would encrypt the information and then store to the cloud and database. This system reduces the labor force while it automatically shows the order on the main monitor in which room where a patient must go. The digitalization helps to prevent misplacement of patient's data. Inbuild log record helps us to see the users who access the patient personal data with all the information.

The pharmacist can also view the patient prescription when the patient RFID card is scanned. It can also allow API request from the Smart Clinic Management System so that can be integrated with the available Pharmacy Management System or Hospital Management System.

3 Requirement analysis

In the first step of the requirement analysis phase, we have to clearly analyze the problem and define the process to solve the problem. Moreover, this document is also used to define the expectation of the systems end user's requirement. A feasibility study is also done in order to make sure that everything is feasible for this system. The feasibility study includes whether the project is commercial, technologically, economic, legal, operational and scheduling feasible. Moreover, the resources required for the system are all identified as hardware, software, and other resources. The exact process of the system is also discussed.

3.1 Feasibility

3.1.1 Commercial feasibility

This Smart Clinic Management System can be a very profitable project for all the hospital sectors. Mostly in all hospital sectors in Sri Lanka waiting time is a major drawback. Therefore, both government and private hospital sectors could use this system in order to solve this issue.

3.1.2 Technical feasibility

The SCMS is a web application made using ReactJS as front-end and Spring Boot as back-end and cloud as storage. This gives the capability of creating a centralized database for the whole patient in the country. And the project is technically feasible because most of the tool is free.

3.1.3 Economic feasibility

For this project, we need to have a computer or laptop to start developing the web application in ReactJS and Spring Boot. Then we also need an RFID reader and RFID PVC card for the development process. Apart from this expense if we want to deploy this system to a real-world situation for example to a private hospital, we have an estimation of 0.94 pounds to register a patient to their hospital. And from the hospital side, we need to allocate for each room a sperate computer and an RFID reader. A sperate monitor is also needed to display the order of the patient in the waiting room.

3.1.3.1 Cost of developing the project

No.	Item name	Price (£)
1	RFID Reader/ Write (1x RFID card)	8.54
2	Laptop	300
3	Hosting (one month)	2.14
4	Cloud storage (Amazon Web Service)	Approx. 5
5	Domain (one month)	7
6	For 126 SMS (SMS Gateway)	8.95
	Total	331.63

3.1.4 Legal feasibility

The project is legally feasible because I can get permission from the Government Hospital of Jaffna to get an insight and the exact process of the patient going to the clinic. And for the development, I am going to use fake data because of the **Data Protection Act.**

3.1.5 Operational feasibility

The project would be operational also feasible because the end user would find it very **easy to manage** all the hundreds of patients. In the manual system, they need to call out each and every patient. This system has certain user groups such as the **doctor, pharmacist, nurse, director, laborist, doctor and admin.**

3.1.6 Scheduling feasibility

This project is all feasible in scheduling according to the System Development Life Cycle. It takes the below shown time:

1. Planning → 5 days
2. Analysis → 7 days
3. Design → 5 days
4. Development → 30 days
5. Testing → 10 days
6. Implementation → 3 day

Totally it would take 60 days to fully complete the system.

3.2 System Development Life Cycle (SDLC)

SDLC has mainly used in many IT industries it is basically a life cycle of steps to develop software. This process also helps to improve the quality of the software since we are going through all the important steps. The steps are planning, analyzing, designing, implementing, testing and integrating and maintenance. The above steps are the basic things to be done for any applications. But we have certain SDLC model for each situation most popular SDLC models are the:

1. Waterfall model
2. Iterative model
3. Spiral model
4. V-shaped model
5. Agile model

The **Waterfall SDLC** model is a process going from the phases of, requirement analysis, system design, implementation, testing, deployment, and maintenance. Waterfall model is mostly known to be used for the small project because they are easy to use and understand. For complex and object-oriented projects, it not recommended. Another drawback is that software is only ready after the last stage is over.

The **Iterative SDLC** model we get the chance to develop some function for system quickly at the beginning of the development lifecycle. The phases for this model are analysis, design, coding, testing and repeat and finally the implementation. The main advantage of this model is that we can parallelly develop the system and the development can be measured easily, easy to manage the risk since we can start to develop the high-risk task first. This type of model is not recommended for small projects. The disadvantages of the iterative model are that is difficult to manage the overall process.

The **Spiral SDLC** model is a combination of waterfall and iterative model. The main problem in this model is to find the right moment to make a step into the next stage. It

has greater scalability because if we want to add new functionality, we can implement it even at a very late stage.

Another model is the **V-shaped SDLC** model it is actually an expanded version of the classic waterfall model. It is so called “Validation and verification” model it makes everything sure before moving to the next stage. This model is well fitted for a project that has clear and fixed requirements and it also a great advantage that testing takes place in the early stage. However, it has drawbacks too it has a lack of flexibility. And is not recommended for small projects.

The **Agile SDLC model** has become popular very recently. The main feature in an agile model is that the customer can inspect the result after every development iteration whether he is satisfied or not. From the customer perspective, it is a very good model to make sure that his product will be developed as he had planned. Since the change could vary in the future development it very difficult to estimate the resources and development cost in advance. The short weekly meeting is also compulsory to show the result to the customer.

Since my project is a mid-size project and I have a clear requirement I will go with the waterfall model. It is also simple and easy to get used to the model. The other models are more likely suited for development with a team or more complex and large size projects. For reference, a waterfall model diagram is shown below.

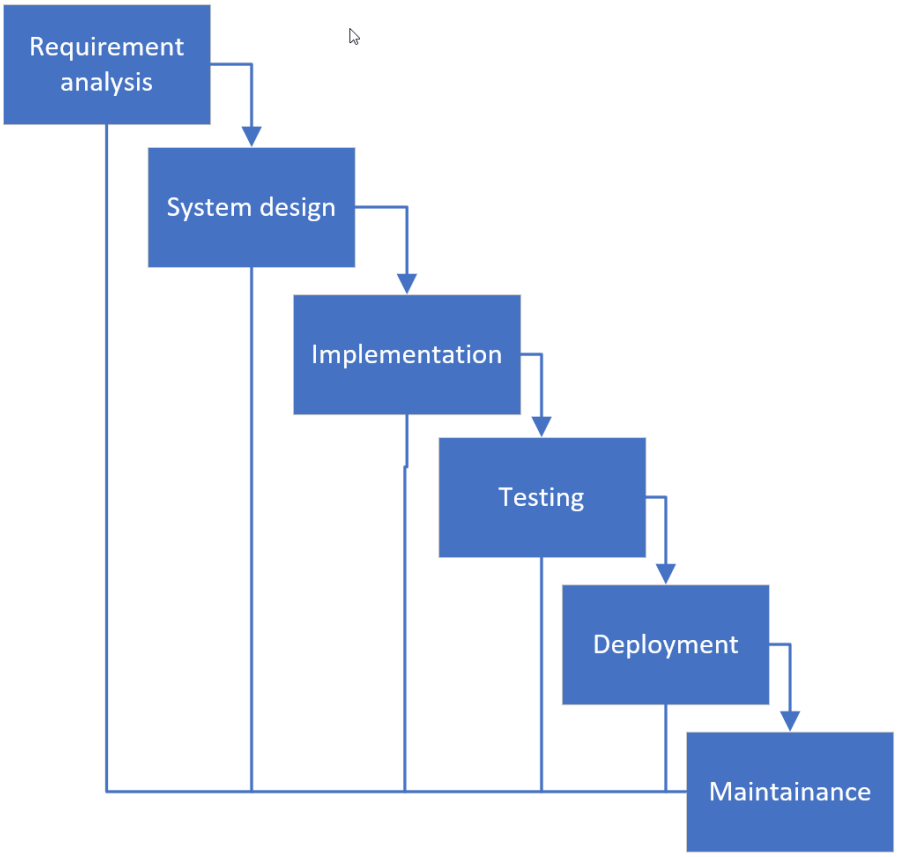


Figure 2 SDLC Waterfall diagram for SCMS

3.3 Requirement of Smart Clinic Management System (SCMS)

3.3.1 Functional requirements

In the SCMS we have

- Pharmacist
- RFID
- Nurse
- Attender
- Laborist
- Doctor
- Director
- Admin

The **Pharmacist** is allowed as a user only to check what medicals are available but we can also connect existing Pharmacy Management System with our API to GET the patients prescription list.

E.g. APIBaseURL +/api/clinic/prescription/{patientid}

The **RFID** is not a real user but work must make the system login in the early morning before the patient arrives for the clinic. The RFID is connected with large TV so that the patient can view their real-time progress of queue while sitting in the lobby. This also allows making the first scan of the patient to register to the queue system in order to display it on the monitor.

The **Nurse** can see only the information of the patient. The Nurse would need the system very rarely but I have added the nurse user type for the future purpose.

The **Attender** is the one who validates patient information manually and then it enters the system to create a new patient for a clinic and he can also edit patient information and delete patient's information.

The **Laborist** is the one who can see the patient clinic information and upload the patient test result.

The **Doctor** has the permission to view all patient and clinic information such as body location. Issues, symptoms, patient test result, prescription list and etc. And the doctor is one who closes the queue with a click of a button which is named as "closeSession".

The **Director** can only view the overall information such as the increasing disease, number of clinic patient, most used medicine, most popular test and etc.

The Admin has access to all the user permissions and has almost the full control over the SCMS but however, for a security purpose, admins are not allowed to view patient's clinic information. The admin can create new user, update a user and delete a user.

3.3.2 Non – functional requirement

Features and characteristic of Smart Clinic Management System.

3.3.2.1 Operability

The Smart Clinic Management System must have good UI and UX which makes any kind of users to understand the system since the workers in Sri Lanka has less computer literacy we need to make sure that the system is easy to understand and to operate in my side I have to make sure any inappropriate action is handled with exception to avoid unrelated data feeding.

3.3.2.2 Reliability

The reliability is another important factor since the system can also be implemented into other government hospitals upon how successful the system is. And handling multiple patient and doctor at the same time is also important for the system. We have to make sure that backend can handle multiple data at once.

3.3.2.3 Portability

Since we use web application for this system, we don’t have portability issues for this system.

3.3.2.4 Response & processing time

The response and processing time must be much faster than the manual system. Since we use the queue system, we need to make sure that the response time and processing time are quick enough to handle a large number of users and patient at the same time.

3.3.2.5 Security

The security is a very important factor for the SCMS since we handle a large amount of patient data which complies with the **Data Protection Act**. Since we store the data in the AWS cloud, we can make sure that the data is kept safe. Apart from that, we use API Gateway which can only be retrieved or stored with a valid JWT token.

3.3.2.6 Simplicity

Keeping the system UI and UX simple as possible helps the user to understand without any tutor at all. Therefore, keeping the system simple helps to reduce the training time for the system.

3.3.2.7 Accuracy

The accuracy of information is also important because we handle sensitive information therefore, we must make sure to return success or error message after an insertion.

3.3.2.8 Efficiency

Since the system is going to run from morning to evening, we have to make sure that performance does not drop after greater run time.

3.4 Gathering requirements

Gathering requirement is to know what kind of features the particular system needs and functions, usability and etc. There are many techniques to gather information from system users. Every project has its own suitable technique.

Some of the examples are given below:

1. Brainstorming
2. Document Analysis
3. Interview
4. Observation
5. Prototyping
6. Survey
7. Questionnaire

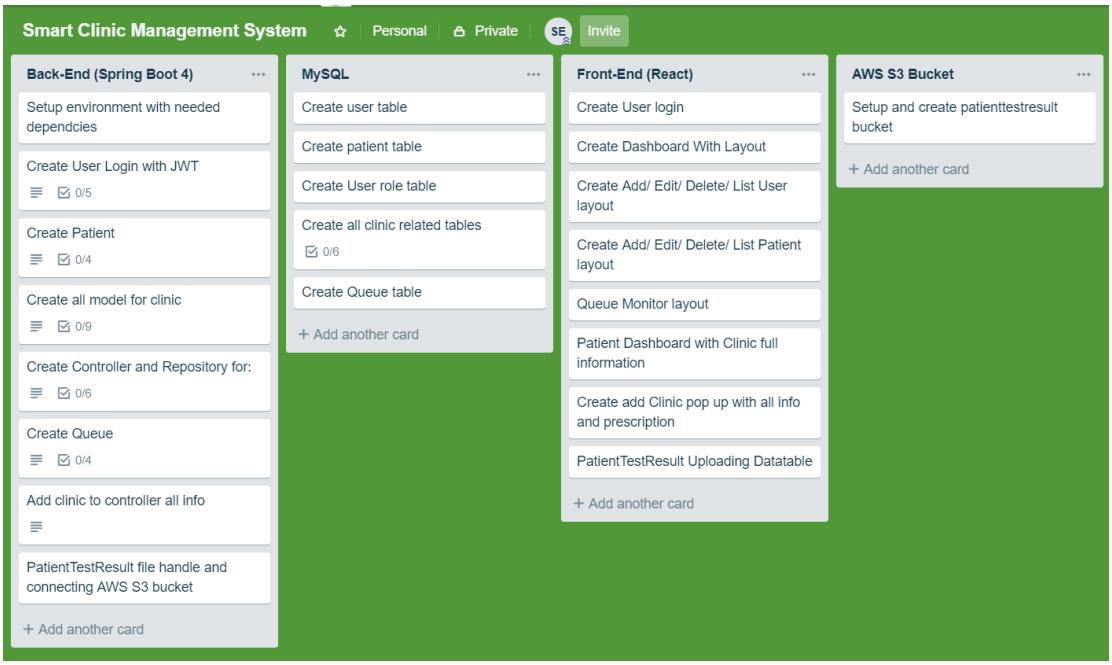
From above the example, I have used **brainstorming**, **questionnaire**, and **observation**.

Since I had already gone a couple of time to the clinic with my father, I know the process of going to the clinic which comes under **observation**. And an attendee working in Jaffna Teaching Hospital who is a relation of mine has supported me to answer my question all the **questions** are below.

1. How many patients would visit the clinic in a day?
2. How many divisions of doctors are available in the hospital?
3. How many staffs are needed for the clinic process?
4. Where is the patient test result taken?
5. Does pharmacist use management tools?
6. Is there any specific time to visit the clinic?
7. How is the current manual process?
8. What issue you have faced in a manual process?
9. What is the most popular disease?
10. Was there a situation where the patient says he has lost his test results?

At last, **brainstorming** is included in brainstorming special feature and how the current problem could be solved using the latest technology.

3.5 Project Management



The above is a screenshot of a web application named “Trello” it is actually projected management tool that can be accessed on any device and anywhere. The above project management show for the main list named Back-end (Spring boot 4), MySQL, Front-end (React) and Amazon S3 bucket. Inside every list, there are a number of cards that help to break down the task and make it even better to monitor the current status and progress.

3.6 Resource identification

3.6.1 Hardware

- RFID PVC card
- RFID Reader/Writer
- Desktop /Laptop
-

3.6.2 Software

- Windows7/8/10 Operating System
- Visual Studio Code
- JAVA and JavaScript
- MySQL Workbench
- Photoshop
- Brackets
- Apache Tomcat server
- Spring Boot
- RFID software

3.6.3 Other resources

- U.P.S
- Stationary
- Miscellaneous assets
- Hospital organizer
- Doctor
- Patient

3.7 Process model

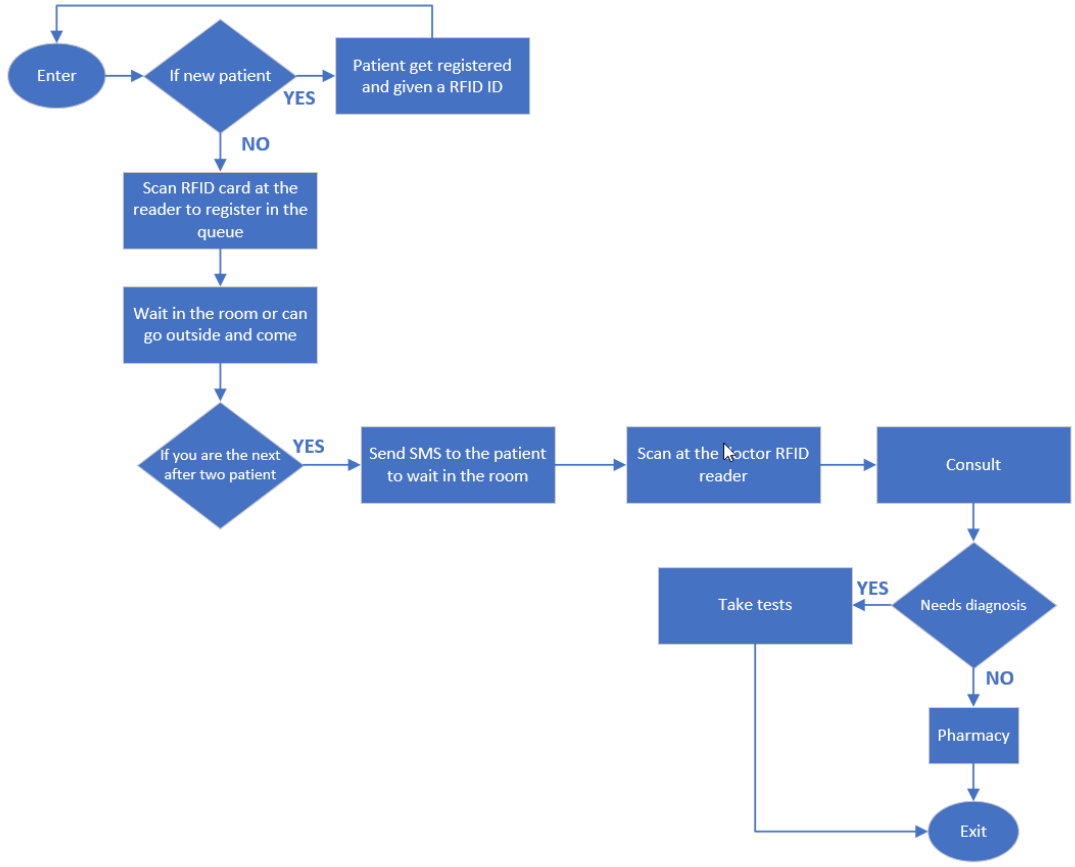


Figure 3 Process model for SCMS

The above diagram represents the main process of the system. At first, a patient enters to the Jaffna Teaching Hospital if the patient is new to the clinic and wants to register then he would be registered in SCMS and given an RFID ID card. Once the patient has registered into the system and got an RFID ID card, he is able to scan the RFID ID card on the reader in order to register in the queue. After that, the patient would need to wait for some minutes the waiting time or position can be watched in real time in the lobby TV monitor. The patient would also get an SMS in order to inform when the patient goes out or nearby. Once the patient is next after two patient an SMS would be sent to the patient in order inform that he would be the next to visit the doctor after two patients. After scanning the RFID ID on the doctor's system, the doctor can view all the information about the patient such as clinic history and patient test result and etc. Once the clinic session is over the patient would be send to take some test and come next time or the doctor would prescribe medicine to the patient

and inform to visit next time. After the patient leaves the room the doctor would click on close session button in order to allow the next patient to come.

4 Project Infrastructure

The project infrastructure of the system is shown in two diagrams below.

4.1 System Architecture diagram

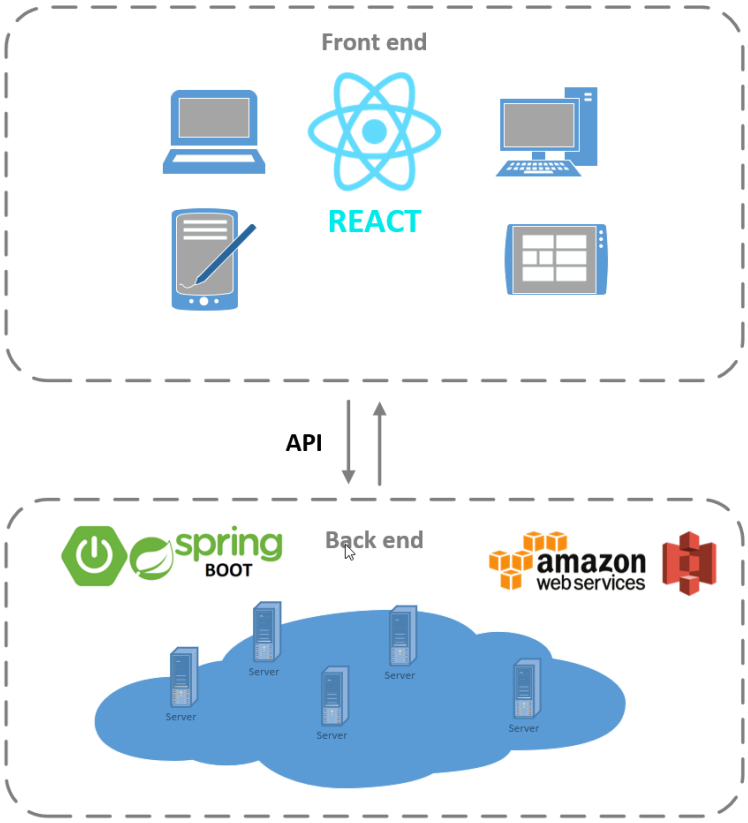


Figure 4 System Architecture diagram for SCMS

The above system architecture diagram shows two component separately as front-end and back-end. The front-end is programmed using React JavaScript library and this front end can be seen on almost all device which has web browsing facility however to make use of RFID Reader we need a laptop or computer. The second component is the back end. The back end is fully programmed using Java with Spring Boot 4 framework this framework makes any backend work fast and easy it used by many enterprise industries. We also connect the spring with AWS S3 bucket to upload patient test result. In between these two components is the API which makes all the POST, GET, PUT and DELETE request and this ties up the two components together as one application to use.

4.2 Tree tier architecture diagram

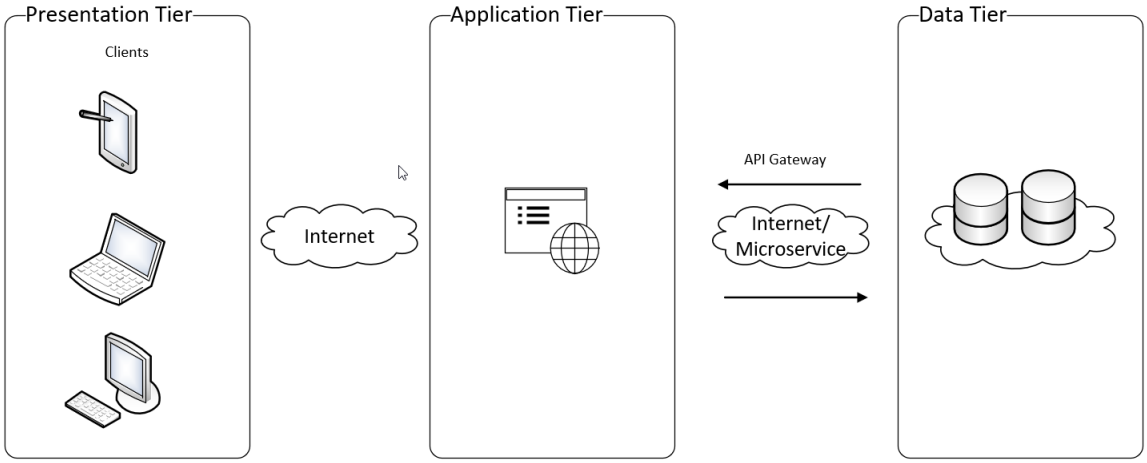


Figure 5 Three-Tier-Architecture for SCMS

The Smart Clinic Management System's tree tier architecture diagram is shown above. There we can see it is separated into three parts such as presentation tier, Application Tier and Data Tier. In the presentation tier we can see some devices such as phone, laptop and computer this is because the web application can be accessed on any devices that support browsing capability. The presentation layer is mainly involved in providing the user interface from the server. The second tier is the application tier that holds the actual React application. The application tier provides the UI to the clients. After that for every react by the client it communicates with the data tier through API request. Since the Data-tier has the main Spring boo application hosted in a cloud server, we can make data request through API calls. Each and every API called are authorized with a token to prevent hacking and misusing of API's.

5 Design

The design phase includes diagrams that support the development process of the next phase. There are many diagrams designed in order to make the system clear for development such diagrams are use case diagram, DFD diagram, Class diagram, Sequence diagram, Activity diagram, ERD and database design.

5.1 Use case diagram

The use case diagram is used represent the behavior of each actor and user cases. Service, function and actions are represented by use cases. It also comes under the UML.

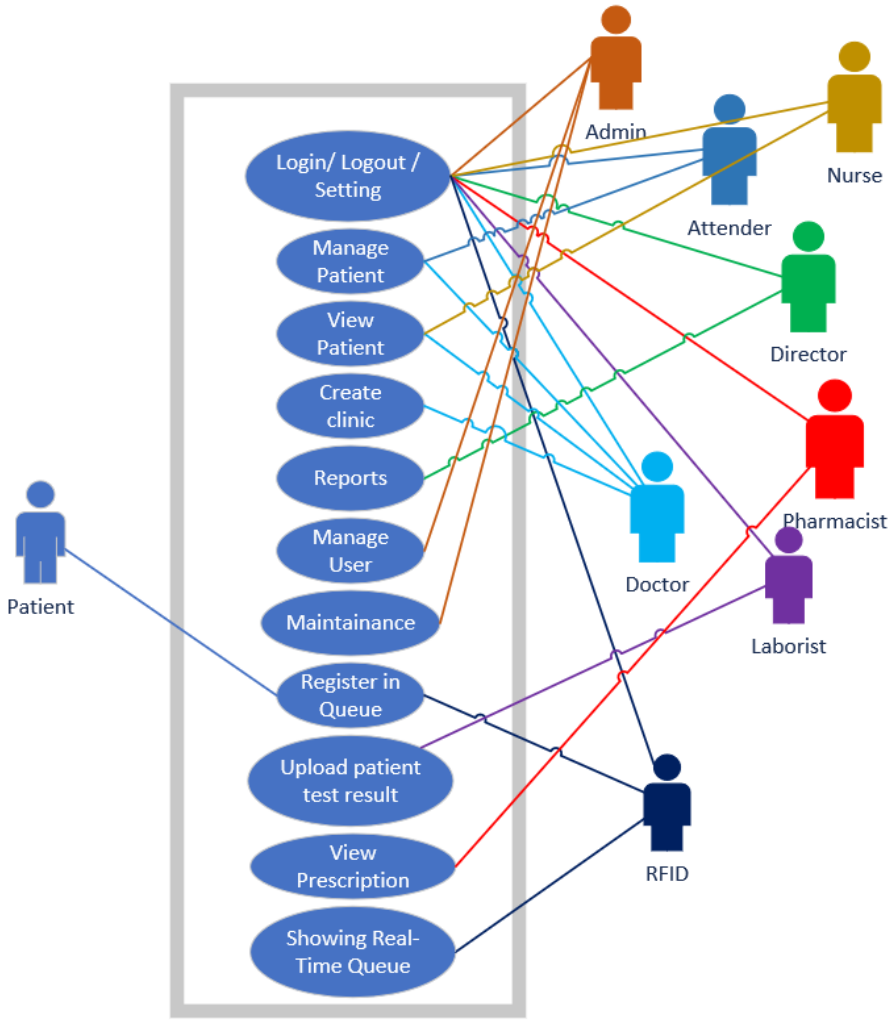


Figure 6 Use Case Diagram for SCMS

The above Use Case diagram shows the different use cases for each user type. On the left side, we have the external stakeholder which is the patient and on the right side, we have internal stakeholder which are admin, nurse, attender, director, pharmacist, doctor, laborist, and RFID. All the internal stakeholder has one thing common which

is the login, , and view setting use case. Apart from that, they have some specific use cases for each type of user types. In the diagram, we can see that the **admin** has the allowance to manage user and maintenance. The **Nurse** has the privilege to view the patient. The **attendee** has a use case to patient. The **director** has the privilege to view reports. The **pharmacist** has the allowance to view the prescription list. And the **doctor** has the use case to manage patient, view patient and create a new clinic. At last, the **RFID** has the use case to register for a queue. The external stakeholder **patient** has the use case to register the queue too because he scans his RFID ID card on the reader in order to perform the progress.

5.2 Data Flow Diagram

The data flow diagram also known as DFD is a graphical representation of how the data flows through the system. To represent this, we have certain level DFD such as a content diagram, DFD level 2 and DFD level 3. For this system, I have given the context level diagram and DFD level 2 diagram which is shown below.

5.2.1 Context Diagram

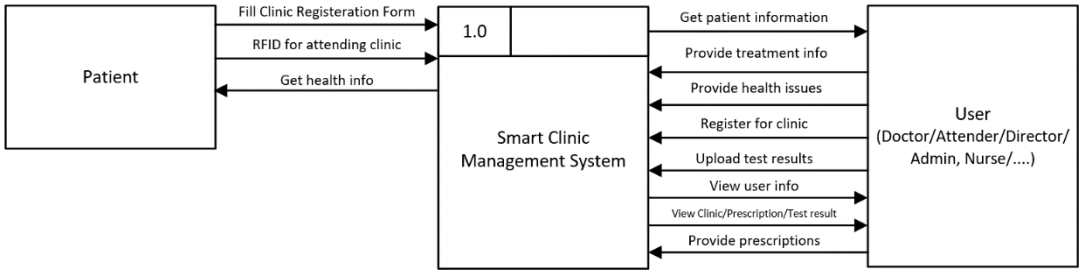


Figure 7 Context Diagram

The above context diagram represents how the data from the patient flows the system and from how to flows to the particular users. As users can see there 11 processes happening each of the describes the flow of data.

5.2.2 DFD Level 2

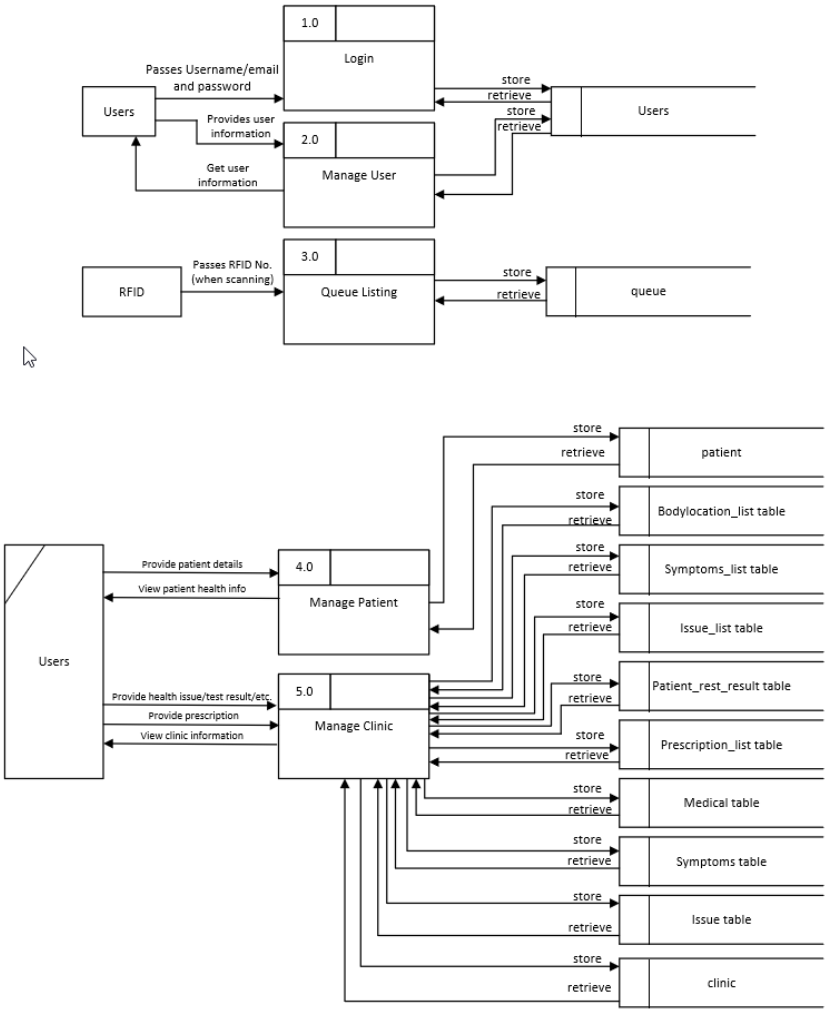


Figure 8 DFD Level 2 Diagram for SCMS

In the DFD level 2 diagram, we have even more detail version of data flowing through the system. The process number 1.0 is the login process it clearly shows how the user passes the username or email and password data to the system make the login process and get appropriate data from the database user. The Users entity also provides useful information to the mange user process 2.0 there also makes the connection between the user's table. The user also retrieves user information from the process 2.0.

The manage patient process 4.0 provides the patient details to the user and views patient health info this process handles with the patient database.

The manage clinic process 5.0 is the process that is most involved with database tables. The three data process are providing health issue details such as body location, issue, symptom, test. Another data flow process is providing a prescription. The last data flow is viewing clinic information. This particular process 5.0 is involved with multiple databases such as bodylocation_list, symptom_list, issue_list, patient_test_result. Prescription_list, medical, symptoms, issue and clinic tables.

5.3 Class diagram

The class diagram is a diagram that shows all the classes of a the SCMS system and all the functions also.

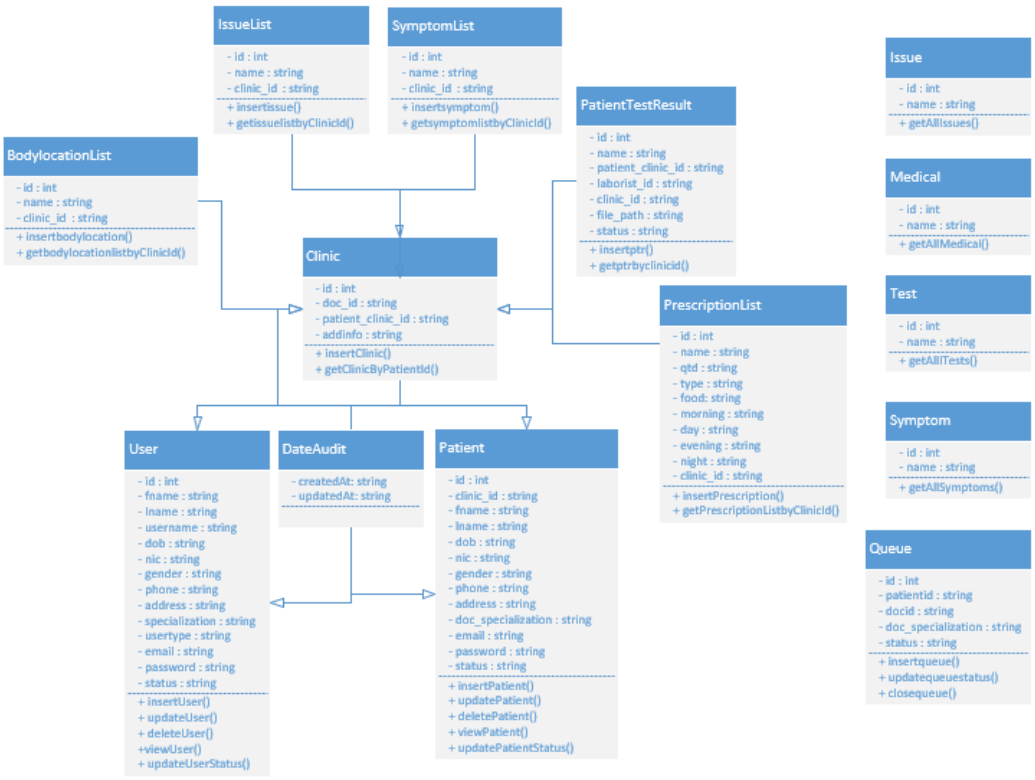


Figure 9 Class diagram for SCMS

5.4 Sequence diagram

The sequence diagram comes under UML diagram type as behavioural diagram. The sequence diagram basically shows the interaction of an operation.

Login Sequence diagram

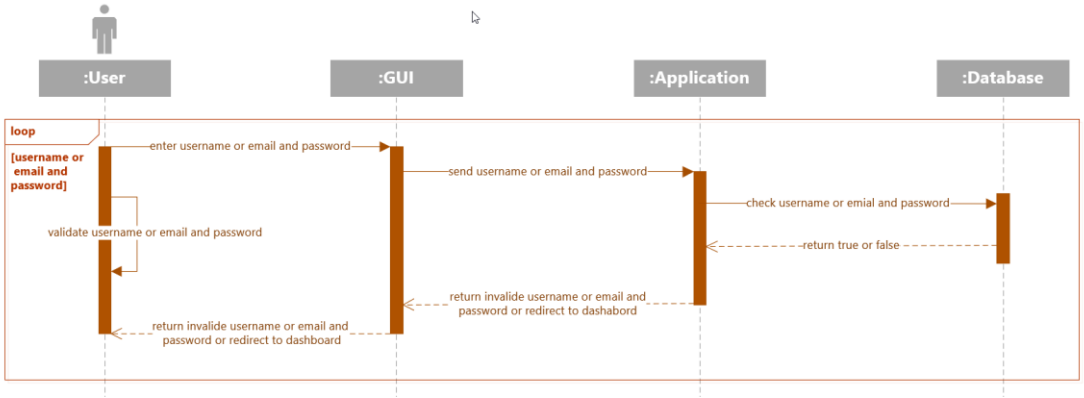


Figure 10 Sequence diagram for SCMS Login operation

The above sequence diagram is made for the login operation. The user entity enters the username or email and password. And then it will be sent to the application to check with the database whether the credentials are correct or not. The message then returns one by one from the database to the GUI.

5.5 Activity diagram

The activity diagram is another UML diagram which comes under behavior diagram. It is basically used to describe the dynamic process of an application like the main activities that happened in the Smart Clinic Management System. The below activity diagram for SCMS show the dynamic process of the patient going through a clinic using the application. At the beginning the patient scans the RFID card in case if the patient is new, he needs to register a new patient and exit. If the patient is already a registered patient he will be automatically registered for the queue. After that, he will go to consult the doctor and once the consultation is finished the patient would three activities to do one is that he needs the get prescription and the second is the get test and the third is to get both things done. After this, the queue will be closed and then the patient can exit.

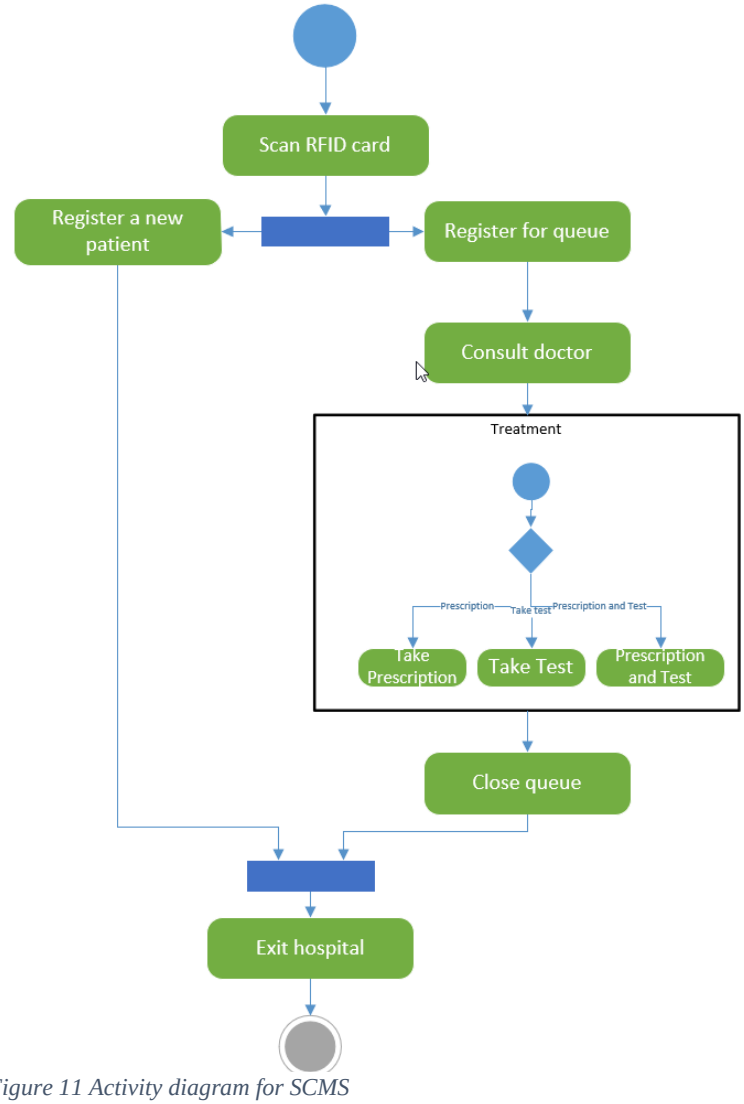


Figure 11 Activity diagram for SCMS

5.6 Entity Relationship Diagram

An entity relationship diagram shows the relationship between each entity of the system. Beside relationship we all so define the attributes for each of the entities. We also indicate what type of relationship it is between entities. There are three different types of entities such one to one, one to many and many to many. Entity relationship diagram is mainly used for database mapping we have to convert from a diagram to an actual database table scheme.

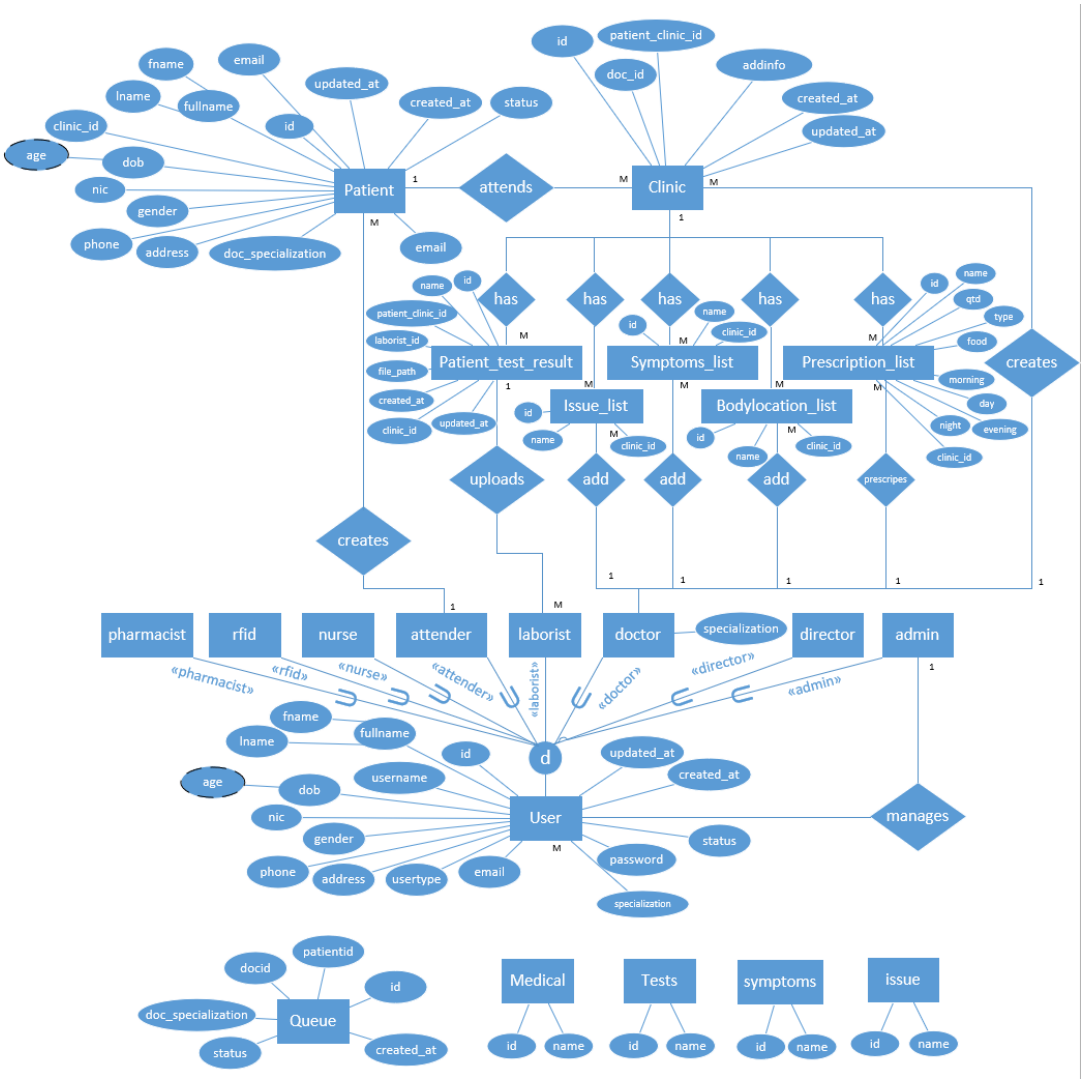


Figure 12 Entity Relationship Diagram for SCMS

In the above ERD, we can see 13 entities and many attributes. The entities are patient, clinic, patient_test_result, issue_list, symptoms_list, bodylocation_list, prescription_list, user, queue, medical, tests, symptoms, and issue. There are 5 entities

without any relationship this is because the medical, tests, symptoms, and issue is used to feed a large amount of data option for the select tag. And the queue entity is just used for the allocation the patient and doctor correctly this table will be truncated every day. Apart from the user entity has many user types which are shown as sub-entities.

5.7 Database design

The below database design is show to represent the mapped Entity relationship diagram with relationship.

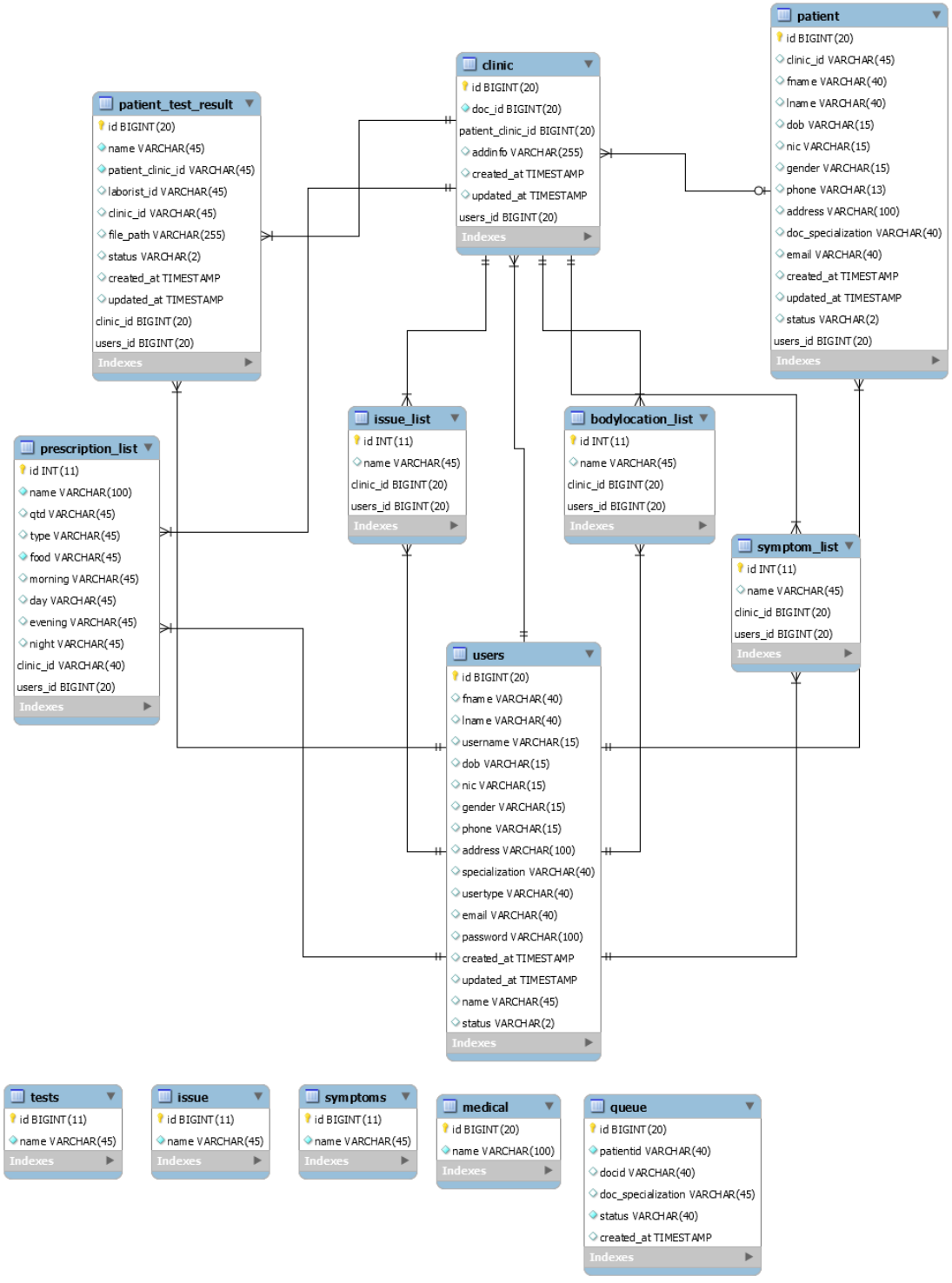


Figure 13 Database design for SCMS

6 User interface

6.1 Login UI

Figure 14 Login UI

The above UI represents a simple login with two fields for the username or email and password. There is also a login button with its label to send the information to the server to validate the credentials. If the credentials are not correct with a stay on the page showing an error message. If the username or email and password is correct it will be redirected to the user welcome page shown below.

6.2 Welcome UI

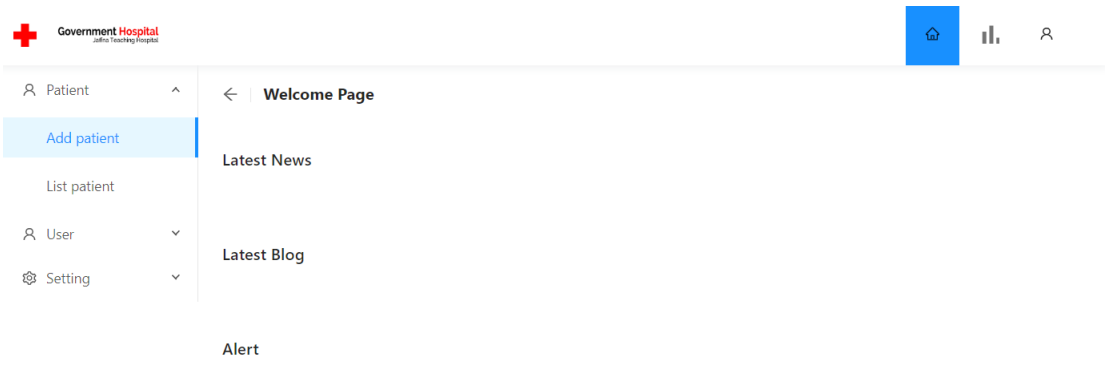


Figure 15 Welcome UI

The above UI shows the Welcome UI with a heading of Latest News which will show up all the latest news in the hospital. Then we have the latest blog for the future purpose where doctors can write blogs for the hospital to represent the foreign countries about its development and new treatments. There is also an alert section to pass information to all the user for a specific problem such as machine is out of service or doctors are needed and etc.

6.3 Add new Patient UI

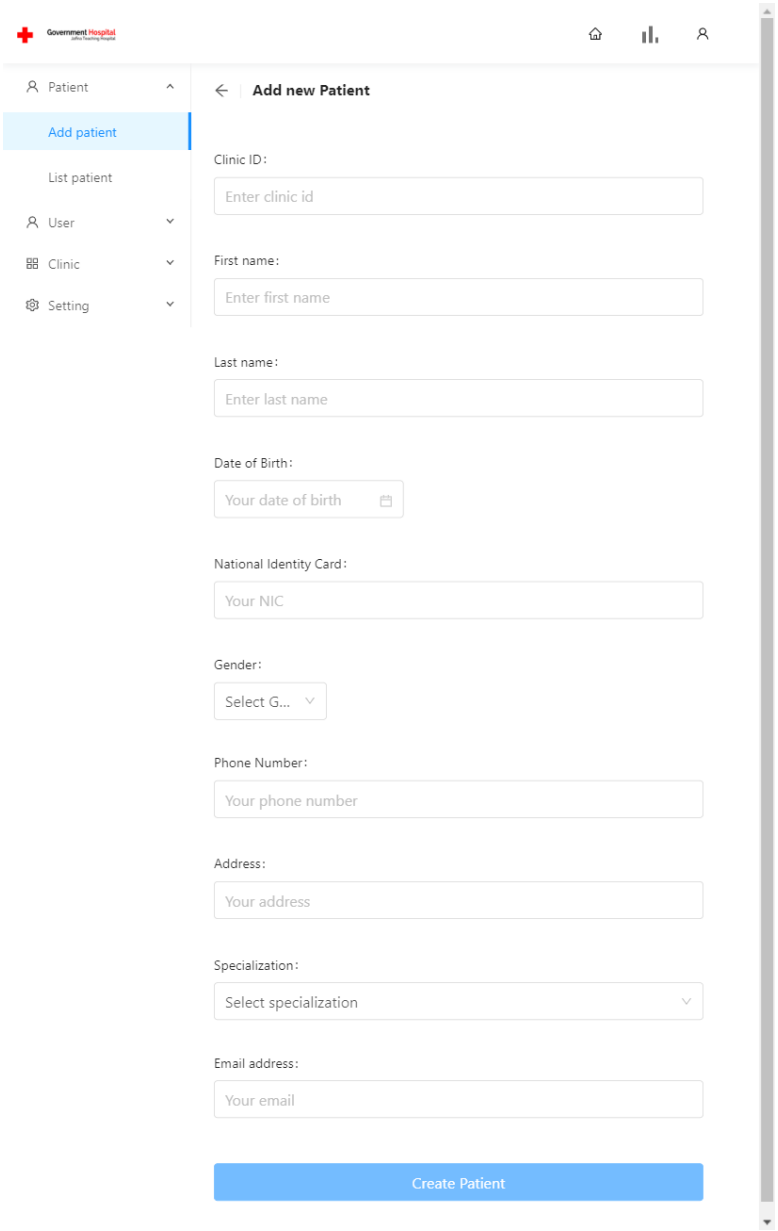


Figure 16 Add patient UI

The above user interface shows the add user UI is the layout used to add a new user for the Jaffna Teaching Hospital. As you can see in the above screenshot, we can see 10 inputs. The first input is to enter the RFID ID which is a unique ID throughout the system. The next two inputs are to enter the first name and last name of the patient. The fourth input is to enter the patient date of birth by clicking the calendar icon a pop window will pop up as a calendar for the user easily to select the date of birth. The next input is to enter the National Identification Card which is also unique throughout the system. The sixth input is the select the gender of the user whether the particular patient is male or female. The eight input is to enter the phone number of the patient this data is important for the hospital in order to make urgent information to the patient. The next input is for entering the patient address for security purposes. The next input is selection box that includes all the doctor's specialization here we have to choose for what kind of specific problem the patient would like to have clinic such as diabetes, skin care, pressure, surgery and etc. At last email address in order to inform the patient about new treatments available in the future.

6.4 List patient UI

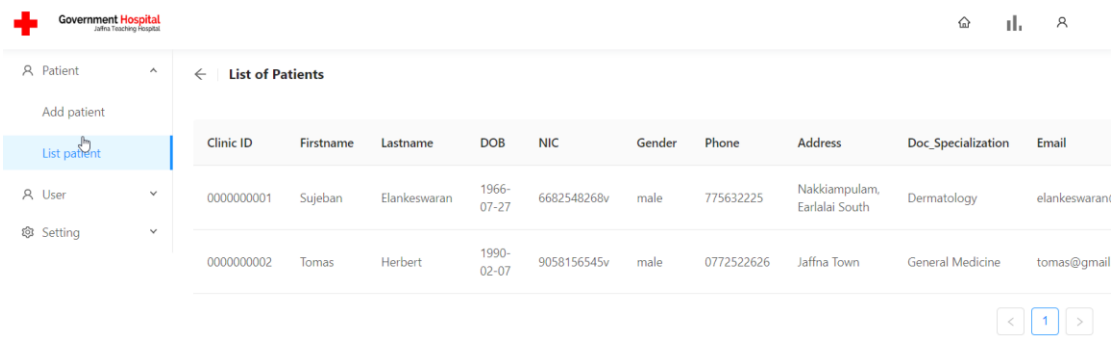


Figure 17 List patient UI

The above user interface is made for listing all the patient in the hospital. As you can see there are 10 columns named as Clinic ID, Firstname, Lastname, DOB, NIC, Gender, Phone, Address, Doc_Specialization, and Email. The fetched data has also been added with pagination.

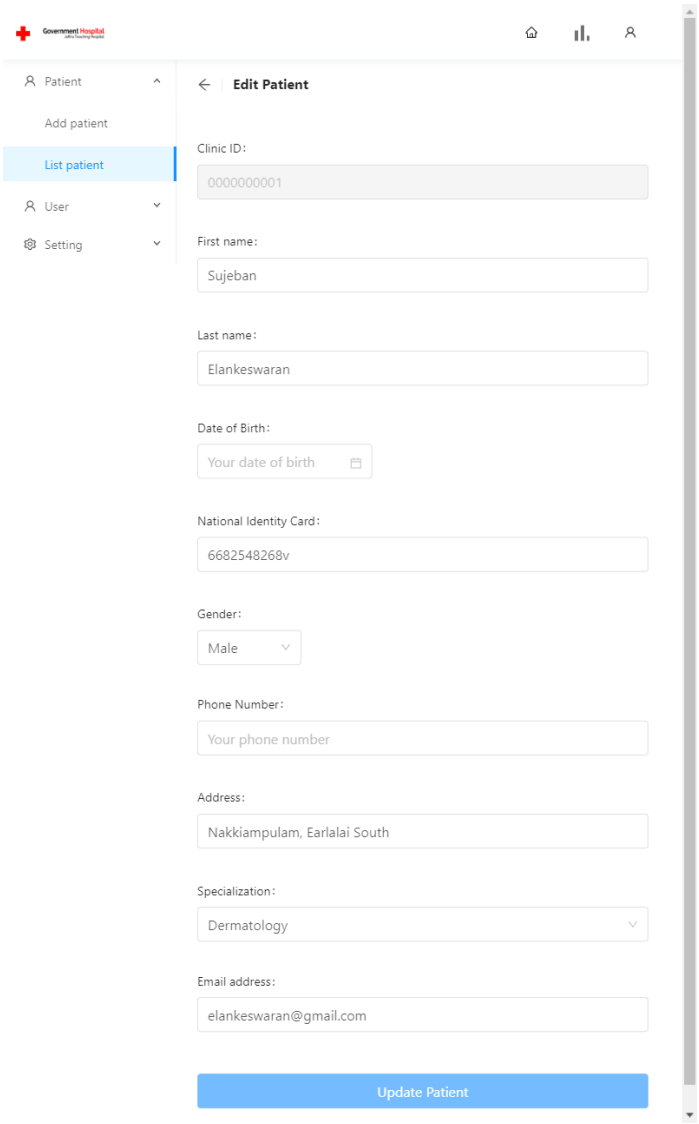


Figure 18 Edit patient UI

6.5 Edit patient

The edit patient UI has 7 input elements, 1 date element, and two select input. The first input is disabled input which displays the patient clinic ID and the next two input is for first name and last name. The date of birth input is made to select the date of birth of the patient. The fifth input is to insert the patient National Identification Card. Then we have the gender input to enter whether the patient is male or female. Next input is input the phone number of the patient. Then wean input to enter the location of the patient. The ninth element is to select input to select the need doctor specialization. At last, we have the email address of the patient to send new treatment

details in the future. It is exactly the same layout as the add patient UI but here the existing data are already filled for the user to know what he is changing.

6.6 Delete patient

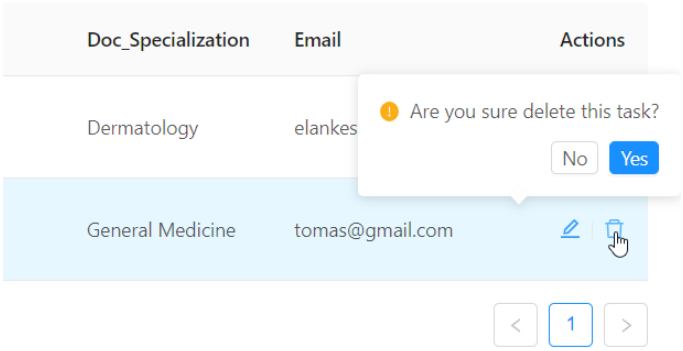


Figure 19 Delete patient UI

The web application can also delete a patient as shown in the above screenshot. When the user clicks on the trash icon a pop up will appear to make sure that you didn't accidentally click the button. After clicking the "YES" button the patient would successfully delete with popup response.

6.7 Add user UI

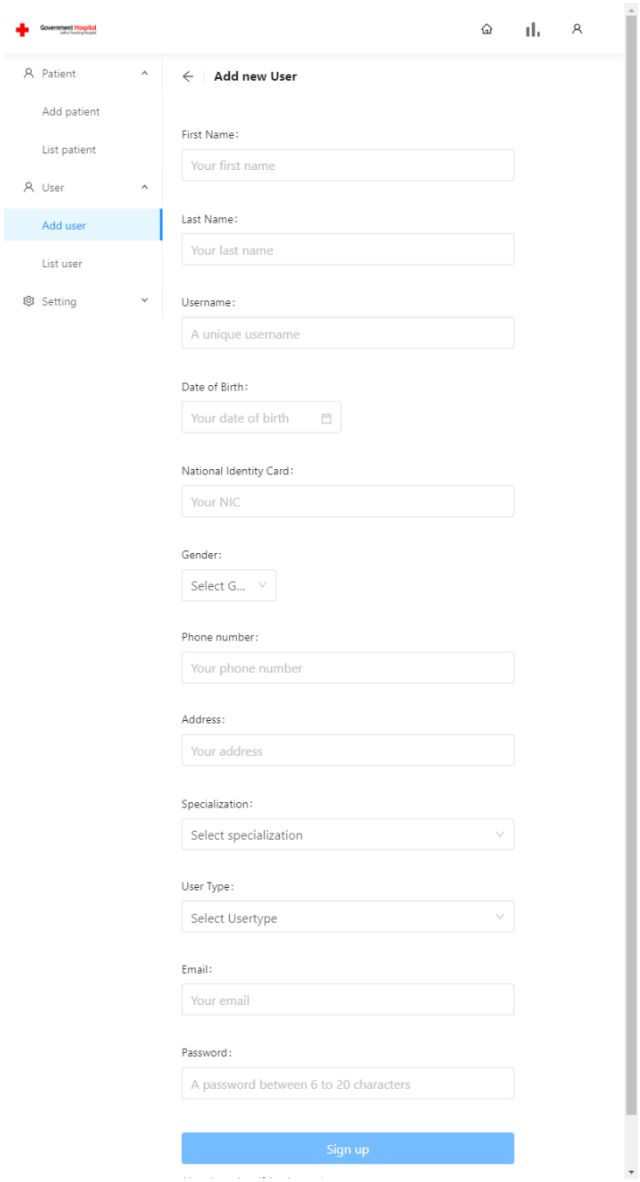


Figure 20 Add user UI

The add user UI 12 input component. The first and second component is for entering the first name and last name. Next, the username which ensures that unique username is entered throughout the SCMS. Next, the date of birth input is made to select the date of birth of the user. Next input is National Identification Card input to enter the users NIC which is also ensured to be unique. Sixth input is the gender select input which has two option, one male and female. Further, there is a phone number input its type is set to number to make sure that the user enters numbers only. The eight input

is the address of the user to security purpose. Now again we have the select input which has a lot option for doctors' specification. The next user types select input with multiple options whether the user is a doctor, nurse, director, RFID, laborist, pharmacist and etc. Next, we have email and password to enable the user to login with its credentials.

6.8 Edit user UI

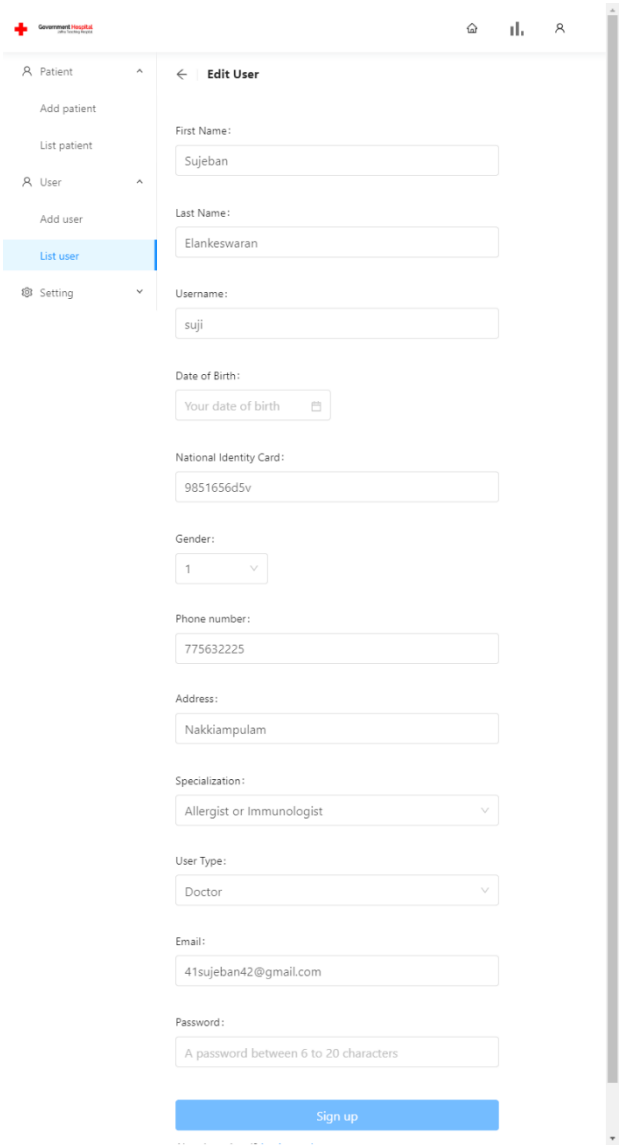


Figure 21 Edit user UI

The above layout represents the edit form for the user. This form automatically feed information based on the user id in the URL. Here it is the same component as explained above the only difference is, we can leave the password blank if the user doesn't want to change its password.

6.9 List user UI

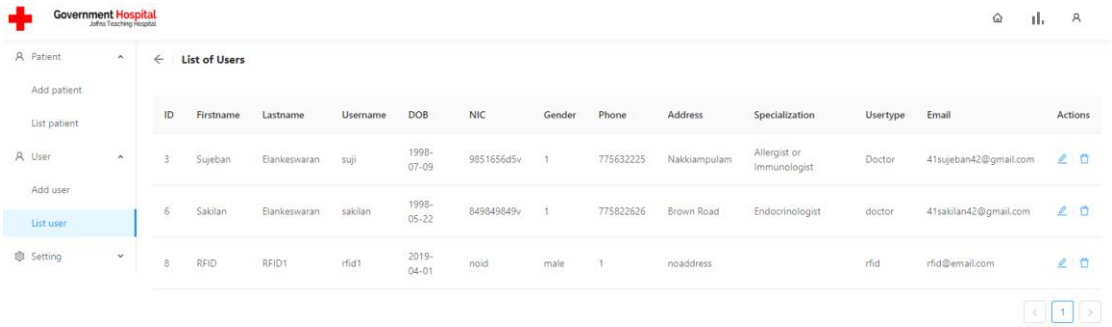


Figure 22 List user UI

The List user UI is layout which displays all the information of all users in the application. There are 13 columns in totals such as id, Firstname, Lastname, Username, DOB, NIC, Gender, Phone, Address, Specification, User type, Email, and Actions.

The last column actions have two buttons with an icon. The first button is to edit the user's information which redirects us to another page to edit the user. And the second button is to delete the user.

6.10 Patient Main Dashboard

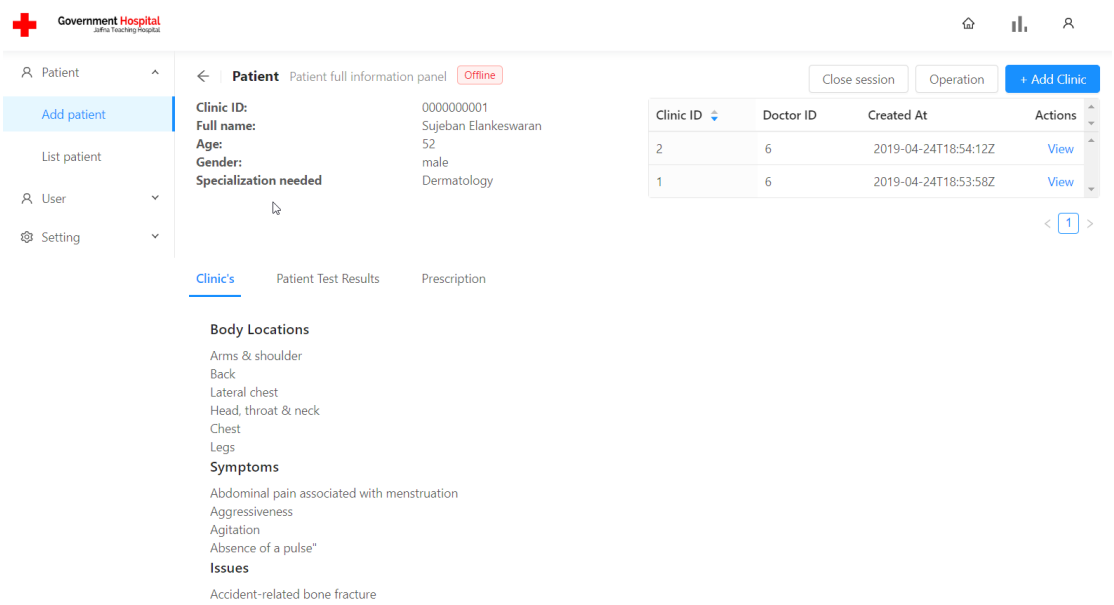


Figure 23 Patient Main Dashboard UI

The screenshot represented above shows the core UI of the system this is a page where doctors are automatically redirected when a patient scans with RFID card. Above we can the page header named as patient and besides there is red Offline box which will change after closing the session. Then we can see five essential information about the patient such as clinic id, full name, age, gender and specialization needed. Besides this information we can see the small table with all the clinic records of the patient. In the action column, we can see a view anchor tag. Once we click the view link then it will call all the information of the particular clinic such as body location, symptoms, issues, patient test result, and prescription.

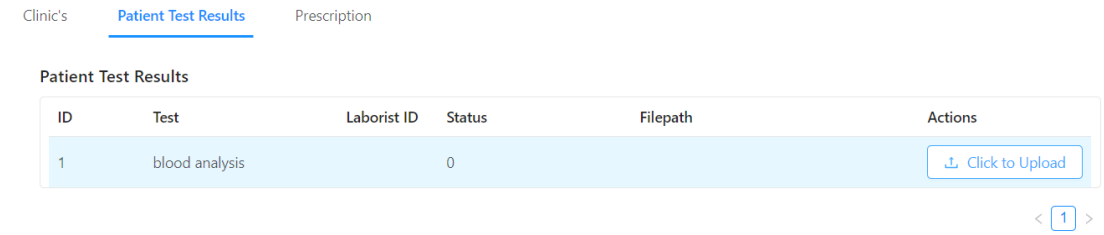


Figure 24 Patient Test Result upload UI

Once we click the patient test result tab, we can see another table containing the test and files like shown above. From the patient dashboard, we can see three buttons such as the “Close session” button, “Operation” button and “Add clinic” button. The “Close session” button is it to remove the patient from the queue and updating the patient status to normal. The “operation” button is a dummy button for future purpose. And “Add Clinic” button is to add a new clinic for the patient.

6.11 Add Clinic Step 1 UI

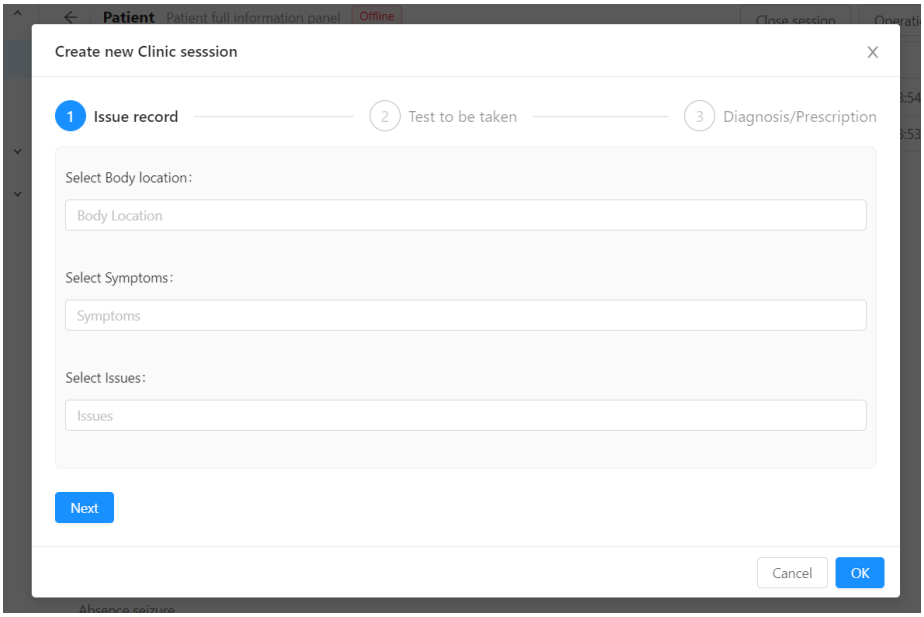


Figure 25 Add clinic Step 1 UI

The add clinic UI has in total three steps and the first step UI is shown above which has three input where the doctor can select multiple options for body location all the option are fetched from the database. Body location input is to mention in which body location the problem occurs. And the second input is the symptom input where it tells the multiple symptoms the patient has now. The last input is to select the issue of whether the patient has specific issues like how it came to a problem such as accident, abortion, ankle injury, back pain and etc. After clicking the next button, we can to the second step which is “Test to be taken” step which will be explained below.

6.12 Add Clinic Step 2

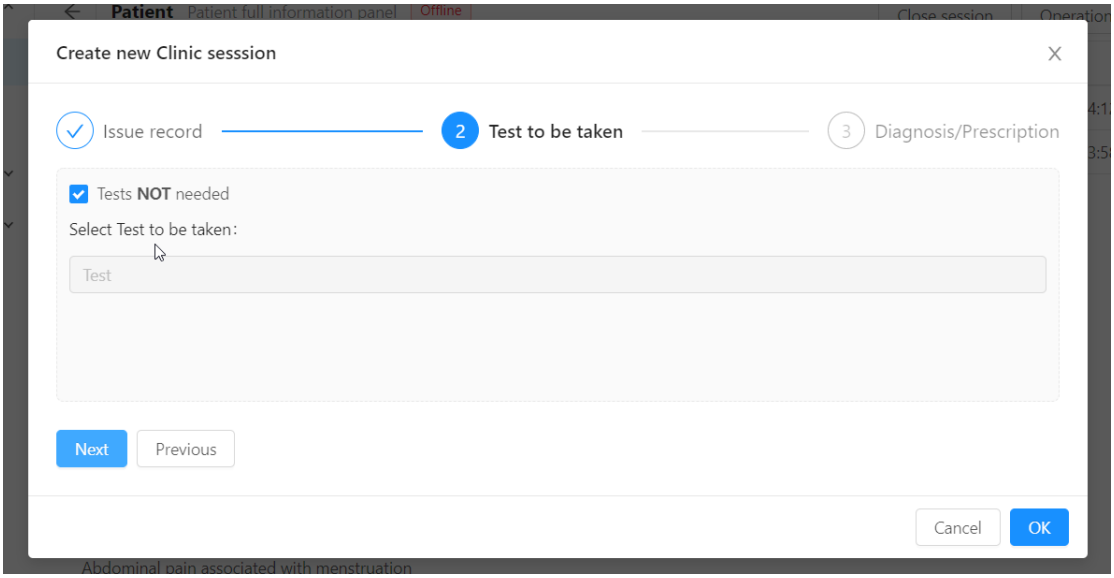


Figure 26 Add clinic Step 2 UI

The above UI represents the add Clinic modal step 2. In this, we can see a checkbox which is prechecked by the system. In case if the doctor thinks that the patient needs a certain test to be taken for further treatment then the doctor would uncheck the box which will automatically enable the below input and select the appropriate tests from the option that needs to be taken.

6.13 Add Clinic Step 3

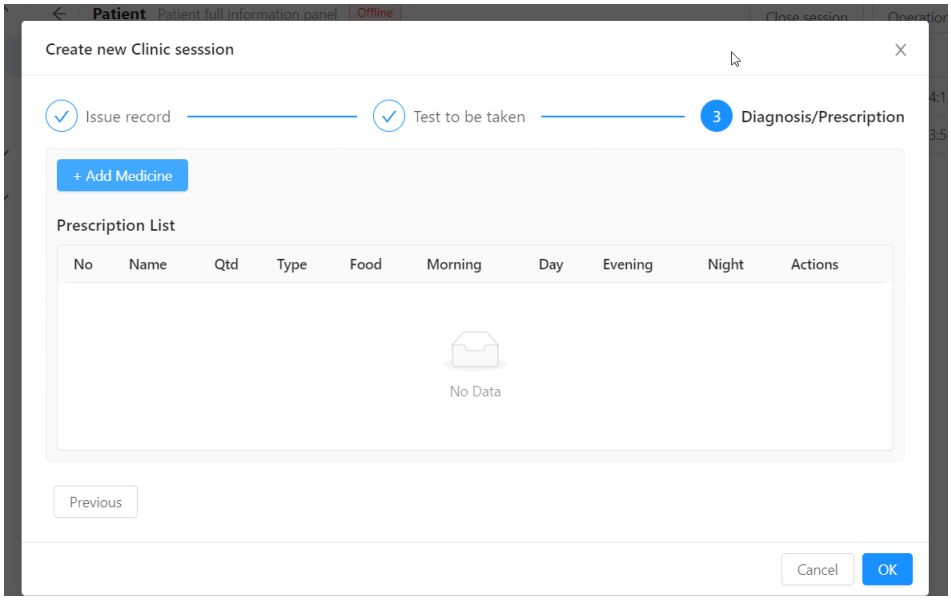


Figure 27 Add clinic Step 3 UI

Step 3 of add clinic UI is the step to prescribe medicine to the patient to do so we need to click the "+ Add Medicine" button to open another window to select a medicine to the patient which will be shown below.

6.14 Add Clinic Step 3.1 UI

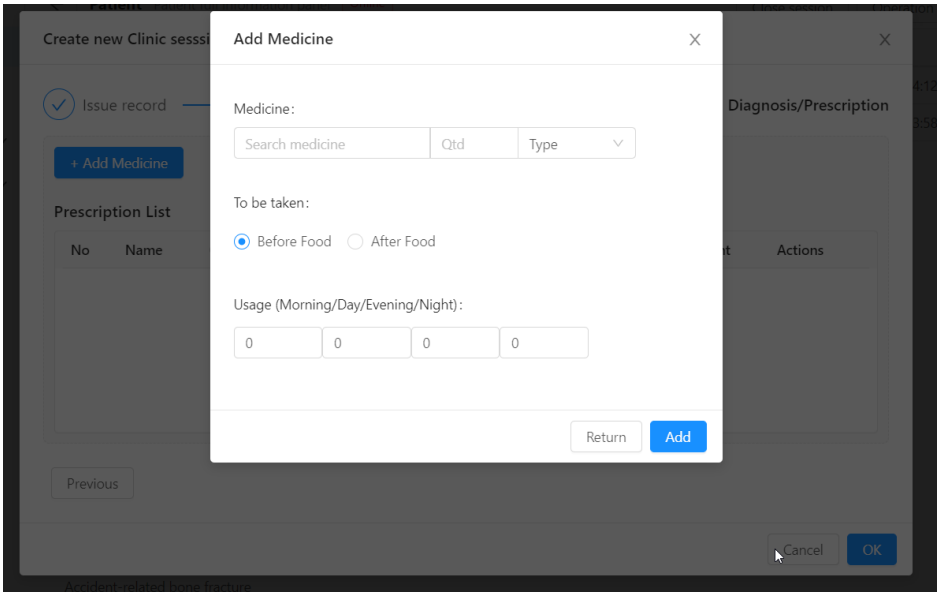


Figure 28 Add clinic Step 3.1 UI

In this modal, we can see 8 inputs. The first row is about medicine once we click the input, we can select one medicine which fetches from the database. Another input is to enter how many tablets to give. The Third input is to specify what type of quantity it is. The next row is the specify whether the medicine needs to be taken before or after food. At the last row, we input the usage of the medicals such as how many tablets to take in the morning, day, evening and night. After clicking the add button the prescription will be listed in the prescription list table.

After that, we can click the "OK" button to create the clinic with all the information given.

6.15 Queue Monitoring UI

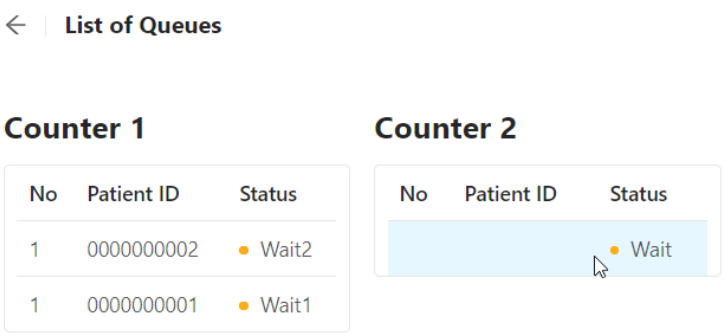


Figure 29 Real time Queue monitoring UI

The above UI represents the Queue Monitoring UI where the main TV is placed in the lobby to show this layout and to inform the patient about the clinic process and queue process.

7 Testing

In this step, we are going to test Smart Clinic Management System with different test cases. Moreover, we have planned to do overall testing, performance testing, auditing, and speed test. Test cases are used to manually test whether all the validation from the databases returned are working. And the overall testing is manual testing where I check everything manually enter the records of the testing. The performance testing used to determine how fast the web application loads. And auditing helps to know whether it loads fast and whether is SEO friendly and whether it has used best practice and etc. Speed test used to compare the website loading speed between the low 3g network and fast 3g network.

7.1 Test Plan and Test cases

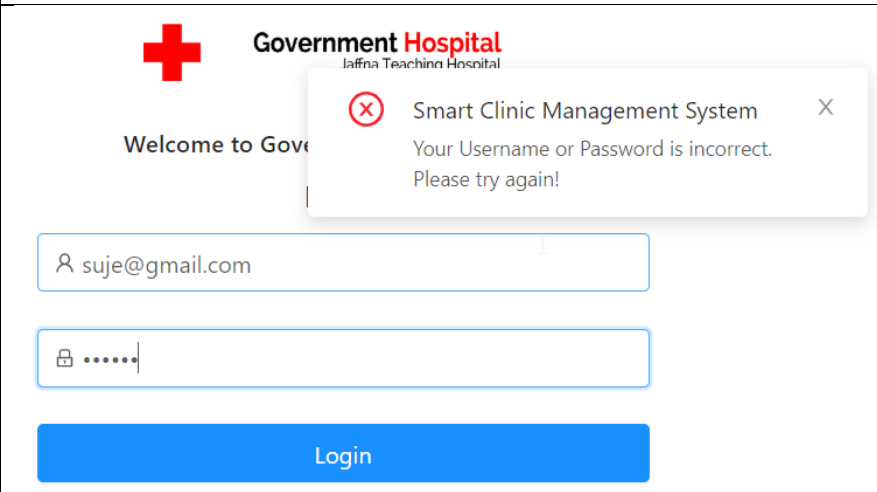
7.1.1 Test Plan

For this Smart Clinic Management System, I have performed:

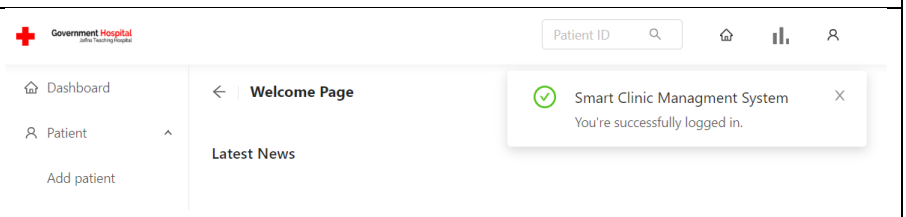
- Test cases
- Overall testing
- Performance testing
- Auditing
- Speed test

7.1.2 Test cases

Test Case 1 – Testing the login system

Test case no	TC01
Test case summary	Checking the username or email and password is working or not
Authority to user	Admin, Director, Doctor, Nurse, Pharmacist, RFID, Laborist
Input	Wrong username or email and password
Test procedure	Enter wrong username or email and password and then click login button
Expected result	Entering the welcome page
Testing data	Username/Email: suje@gmail.com Password: 123456
Status	Failed
Reason	Since the email and password does not exist in the database it return an error message.
Actual result	Your Username or Password is incorrect. Please try again!
Screenshot	
Test Environment	Smart Clinic Management System – Client Side (React)

Test Case 2 – Testing the login system 2

Test case no	TC02
Test case summary	Checking the username or email and password is working or not
Authority to user	Admin, Director, Doctor, Nurse, Pharmacist, RFID, Laborist
Input	Correct username or email and password
Test procedure	Enter correct username or email and password and then click login button
Expected result	Entering the welcome page
Testing data	Username/Email: 41sakilan42@gmail.com Password: 41sakilan42
Status	Passed
Reason	Since the username or email and password exist it lets us to enter the welcome page.
Actual result	You're successfully logged in.
Screenshot	
Test Environment	Smart Clinic Management System – Client Side (React)

7.2 Overall testing

Test No.	Test description	Step to test	Expected results	Status
1	Login testing	Enter a valid username or email and password	Redirects to welcome page	**PASS**
2	Logout	Click logout button	Redirects to login page	**PASS**
3	Add a user	Insert data to add a new user	New User successfully registered!	**PASS**
4	Edit user	Insert data to edit an existing user	User Updated Successfully user updated!	**PASS**
5	View user	Click list user menu	List of users fetched	**PASS**
6	Delete user	Click delete user button	User Deleted Successfully user deleted!	**PASS**
7	Add patient	Insert data to add a new patient	New patient successfully registered!	**PASS**
8	Edit patient	Insert data to edit an existing patient	Patient successfully updated!	**PASS**
9	View patient	Click on List Patient menu	List of Patient is shown	**PASS**
10	Delete patient	Click delete icon to delete the patient	Patient successfully deleted	**PASS**

11	Add clinic session	Insert empty data to create new clinic record	Please check your inputs!	**FAIL**
12	Register for queue	Scan RFID to register queue	Successfully register in queue	**PASS**
13	Search patient	Insert wrong RFID ID to search patient	Could not find such patient. Please try again	**FAIL**
14	RFID scan in doctor console	Scan RFID in doctor console to change patient status in queue	Status changed	**PASS**
15	Upload patient test result	Upload particular patient test result	Successfully file inserted	**FAIL**
16	View clinic sessions of a patient	Scan RFID of a patient	All clinic session of a patient shown	**PASS**

7.3 Performance testing

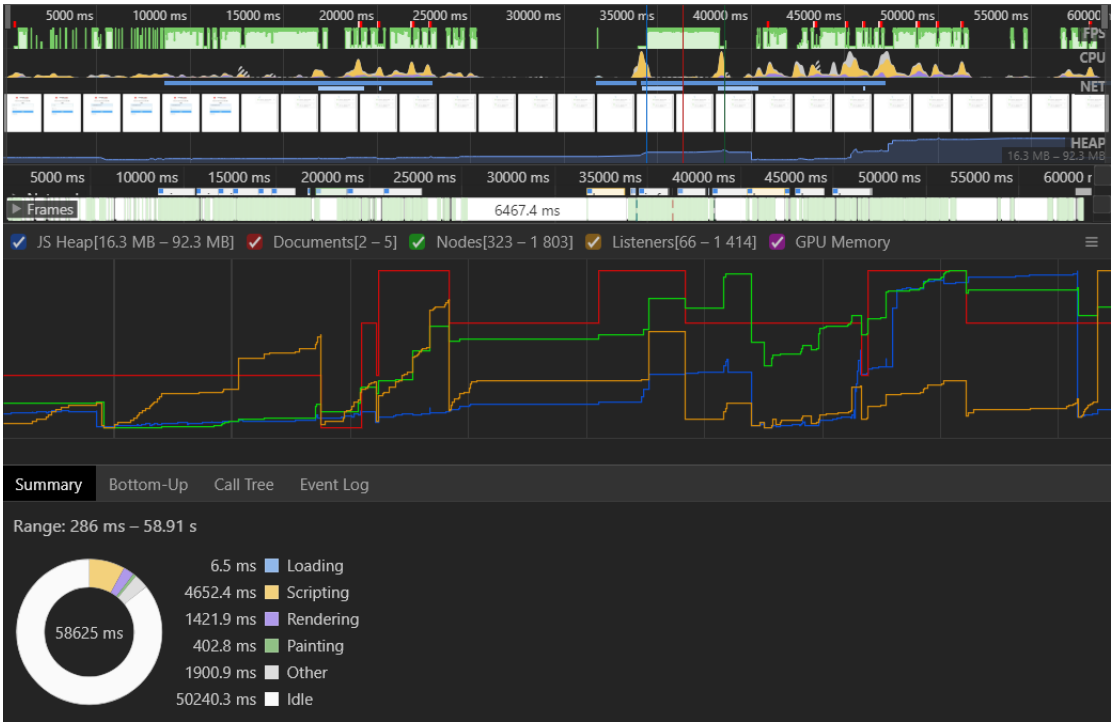


Figure 30 Performance Testing for SCMS

The above performance testing shows that totally it took 6.5ms to load the patient dashboard layout. From the 6.5 ms, the scripting takes 4652.4 ms, rendering 1421.9 ms, painting 402.8 ms and other 1900.9 ms.

7.4 Auditing – Patient Dashboard

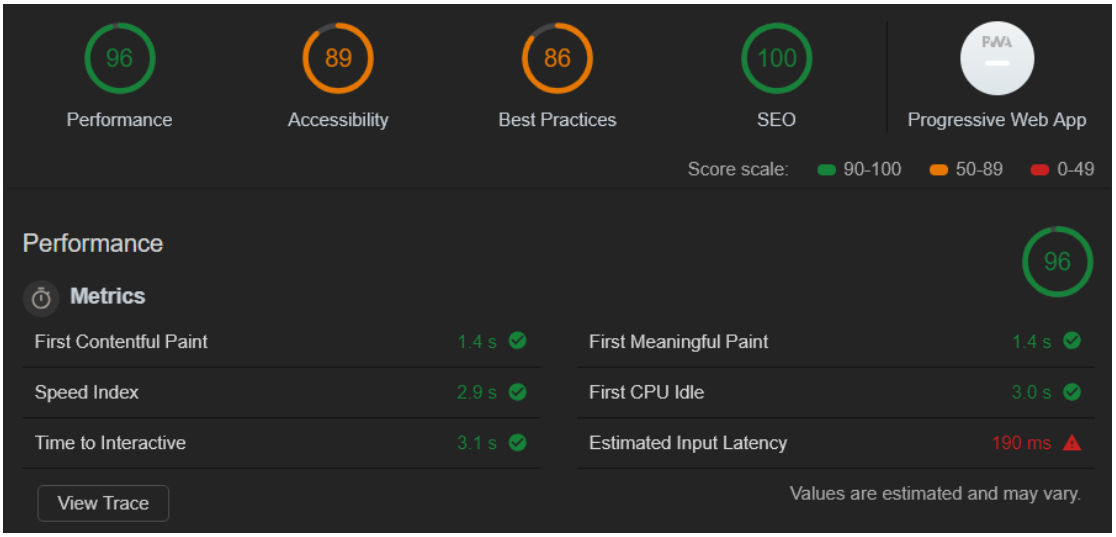


Figure 31 Auditing - Patient Dashboard

While taking an auditing test for the most loaded layout which is the patient dashboard it tells that the performance has 96 marks, accessibility 89 marks, best practice 86 marks and SEO 100 marks. Therefore, we can estimate that the Smart Clinic Management System has good performance overall.

7.5 Performance in Fast 3G VS Slow 3G

7.5.1 Fast 3G

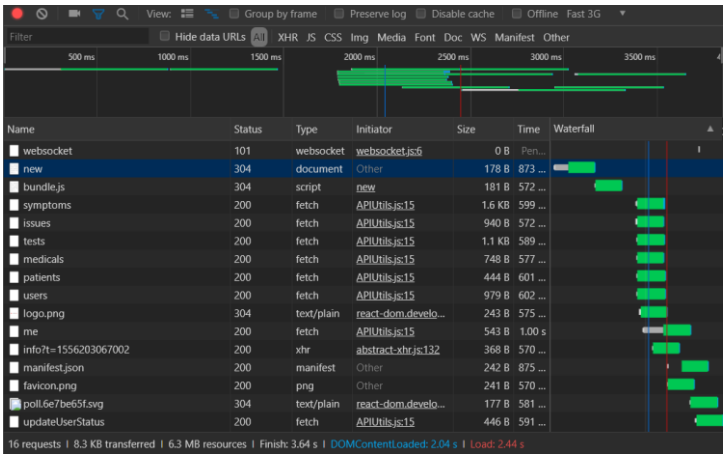


Figure 32 Fast 3G Speed Test

The above testing shows that with fast 3g the website loading speed is **2.44 s**.

7.5.2 Slow 3G

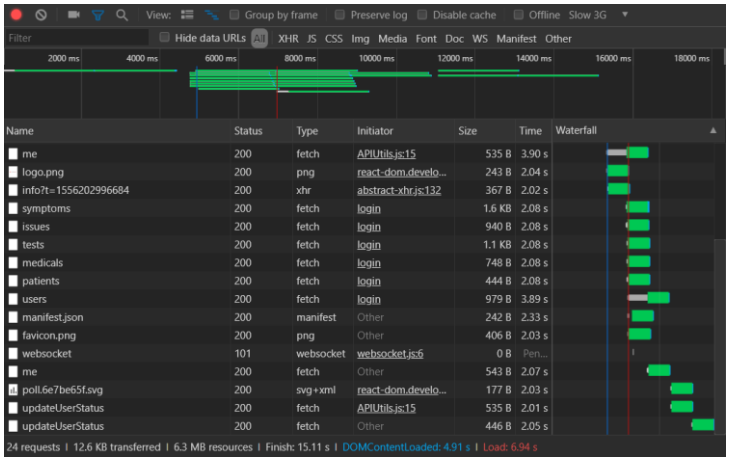


Figure 33 Slow 3G Speed Test

The above slow 3g testing shows that the application took **6.94 s** to load.

In conclusion the between the fast 3g and slow 3g network the difference was 4.5 s which is not bad. But however, for better performance of the web application we need to have at least fast 3g.

8 Implementation

In this Smart Clinic Management System, we have separated into 3 parts Front End, Back End, and Storage. The front end is made using REACT library, it is a JavaScript library specially made for building user interfaces. React makes us easy to developed complex UI in a simple manner. The main concept of React is to break apart all the UI into small components and if possible, we can break down to even more component. By breaking the component, we are able to use the component every time we need just by importing the particular component.

The back end is the second part which is made with Spring Boot 4 JavaScript framework. The JavaScript framework helps us to connect the database or cloud storage with spring application in order to create API. We create API in order to fetch data into the react application and post data into spring as well. It is basically HTTP communication between the React application and Spring application.

For third part storage, we use the traditional MySQL for text data and AWS S3 cloud storage to store files from the patient test results.

8.1 Software and Hardware Requirement for Implementation

The Software requirement for the implementation is Visual Studio, MySQL workbench, Apache Tomcat Server, Spring Boot 4, Postman and Google Chrome.

The Visual Studio Code is an IDE that is used to program the front – end SCMS using react library. In this IDE we have built in terminal to execute react app and see the progress there itself it also supports many extensions such as Beautify, ES Lint and etc. The MySQL Workbench is an application to create a database scheme and tables to integrate with Spring Boot Application. Next, the apache tomcat server needs to run the Spring Boot Application on the localhost. The Spring Boot 4 is a customized eclipse IDE just to create Spring application with ease. The Postman is used to test the create API request with server and MySQL. The Google Chrome is used to see the user interface created using react and google chrome is also the best browser to inspect the design issue in inspecting mode.

There are is also a hardware required to implement the system for the development purpose we would need a laptop or a computer and an RFID reader only. A picture of the RFID Reader and Writer used for the development of this project is shown below.

Figure 34 RFID Reader/Write and RFID Card

8.2 MySQL Storage or AWS S3

MySQL is a relational database management system which is now maintained by Oracle. The MySQL run on all platform virtually. This component is widely used by many developers to create a database for an application. Most of the top website in the world is the MySQL RDBMS. The developers mostly use web development platforms like LAMP, XAMP, and WAMP. LAMP used by the Linux operating system and XAMP can be used by any operating system and WAMP is used for the Windows operating system. These web development platforms have Apache which is the web server and MySQL as a database management system and PHP for as a scripting language. In our case, we don't need a web development platform to use MySQL we can install the MySQL Workbench 8.0 CE. Along with this, we connect with Spring to create API.

MySQL integration with Spring

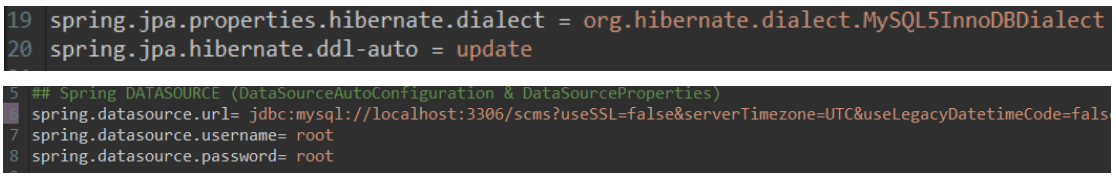


Figure 35 MySQL integration

In the above code snippet, we all the MySQL library to make a connection with JPA. And the next line is updating any changes in dB migration. In the second code snippet, we mention the URL where we can access our database. And the next two lines are there to mention the username and password to make the connection available with the MySQL.

AWS S3 Bucket integration with Spring

In AWS S3 Bucket I have created a folder called "patienttestresult" where we can upload all the uploaded patient test result file in the cloud safely.

Figure 36 AWS Integration coding snippet

In the above piece of code, we declare the AWS access key, secret code, the region of the AWS and the bucket name. All this information are write-in <application.properties> file.

8.3 Back end (Spring Boot 4)

The back end of this system was developed using Spring Boot 4. It is framework writing in Java and it is used to create a microservice. This framework has developed the team called Pivotal team. Developers use spring boot to create an Application Program Interface (API) gateway. An API allows as to make all HTTP request such as POST, GET, DELETE, PUT and etc. using custom URL. For example, if we would type this URL in our browser "http:localhost:5000/api/users" we could GET all the users from the database if we have written the appropriate coding in Spring Boot. Spring Boot application can also be deployed in Amazon Web Service to make it work on cloud as well.

Below we could see a sample code of mapping an API. As you can see in the image there is an annotation "@RestController" this is used to tell the Spring that we are going to create an API. The meaning for REST is Representational State Transfer which is basically a method to allow client and server to communicate. Then we see another annotation "@RequestMapping("/api/queues")" this annotation tells to the spring the if we type URL ending with "/api/queues", it would automatically know that we call the QueueController. Inside the QueueController now if by seeing the below screenshot we can see another annotation called "@GetMapping" it tells that if we use GET method with the above API link it would execute the below function which is a function to findAll queues and return it into List data type. Another important thing in spring is that we use **JPARepository** and **CRUDRepository** this makes us as a developer to skip all the basic queries to write again and again. It mainly covers all the basic queries such as save(), saveAll(), getOne(), findOne(), findById(), findAllById(), findAll(), existById(), deleteAll(), delete(), deleteById() and etc.

```
28 @RestController
29 @RequestMapping("/api/queues")
30 public class QueueController {
```

Figure 38 @RestController snippet

```
42     @GetMapping
43     public List<Queue> getQueues() {
44         return queueRepository.findAll();
45     }
```

Figure 37 @GetMapping snippet

8.4 JWT Token

JWT stands for JSON Web Token is basically a method that helps represent claims securely. JWT has industry standard RFC 7519 method. JWT main feature is to decode, verify and generate tokens. I used JWT to make every API request securely to allow only our application users if we don't protect with such as authorization then anyone can access our data and manipulate and that I don't want to happen.

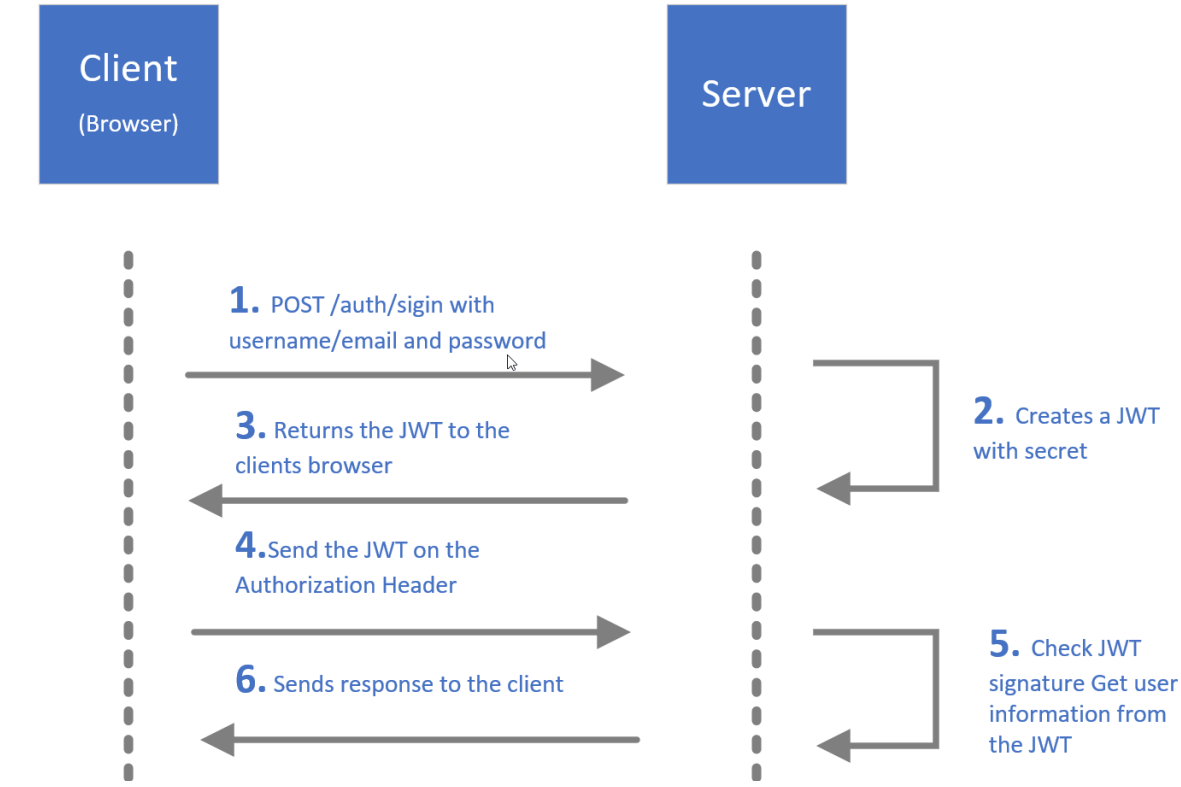


Figure 39 JWT concept

The above diagram shows the sequence of how the JWT is actually used and works. In the first step, we post the username /email and password of the user from the browser. The second step continues in the server where it creates a JWT with a secret when a username or email and password of the user are correct. In the JWT we can also the payloads such as user roles and etc. and this payload can be decoded in the server again set role permissions. The third step we return the JWT to the client browser. In the fourth step, we send the JWT on the Authorization Header and request for user information. In the fifth step, the server validates the JWT signature to all the

user information to return to the client browser. And in the sixth step, the response is sent to the client browser.

8.4.1 Importing Json Web token

```
47          <!-- For Working with Json Web Tokens (JWT) -->
48          <dependency>
49              <groupId>io.jsonwebtoken</groupId>
50              <artifactId>jjwt</artifactId>
51              <version>${jjwt.version}</version>
52          </dependency>
```

Figure 40 Import JWT code

8.4.2 Generating Json Web token

The above coding shows how we import json web token into the pom.xml.

```
23      public String generateToken(Authentication authentication) {
24
25          UserPrincipal userPrincipal = (UserPrincipal) authentication.getPrincipal();
26
27          Date now = new Date();
28          Date expiryDate = new Date(now.getTime() + jwtExpirationInMs);
29
30          return Jwts.builder()
31                  .setSubject(Long.toString(userPrincipal.getId()))
32                  .setIssuedAt(new Date())
33                  .setExpiration(expiryDate)
34                  .signWith(SignatureAlgorithm.HS512, jwtSecret)
35                  .compact();
36      }
37
```

Figure 41 Generating JWT code

In our Spring application, we have to create a function to generateToken() after a successful login. As shown in the coding we can see that I have set users id as subject and in the issueAt, we declare the current date of token generation. In the setExpiration() we declare the how long the token is valid. Finally, we sign with SignatureAlgorithm.HS512 and jwtSecret.

8.5 Login function

The coding shown above is POST API for sign in this where the username or email and password are authenticated. We authenticate the username and password using the authentication manager since we use an add-on in spring which is the Spring Security, I am able to use that feature. It is basically a container that helps to authenticate username and password.

```
51    @Autowired
52    JwtTokenProvider tokenProvider;
53
54    @PostMapping("/signin")
55    public ResponseEntity<?> authenticateUser(@Valid @RequestBody LoginRequest loginRequest) {
56
57        Authentication authentication = authenticationManager.authenticate(
58                new UsernamePasswordAuthenticationToken(
59                        loginRequest.getUsernameOrEmail(),
60                        loginRequest.getPassword()
61                )
62        );
63
64        SecurityContextHolder.getContext().setAuthentication(authentication);
65
66        String jwt = tokenProvider.generateToken(authentication);
67        return ResponseEntity.ok(new JwtAuthenticationResponse(jwt));
68    }
```

Figure 42 Login function code

To make the users password unreadable to anyone we used the passwordEncoder.encode() function to specifically encrypt the user's password. The coding for this function is shown below.

```
93
94          user.setPassword(passwordEncoder.encode(user.getPassword()));
95
```

Figure 43 Password encryption code snippet

```
@Entity
@Table(name = "users", uniqueConstraints = {
        @UniqueConstraint(columnNames = {
            "username"
        }),
        @UniqueConstraint(columnNames = {
            "email"
        })
})
public class User extends DateAudit {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotBlank
    @Size(max = 40)
    private String fname;
```

Figure 44 Code Snippet from Model User

The above coding is from model "User" in this coding we can see many annotations. The first annotation is the "@Entity" which is used to declare the User as an entity throughout the application. And the next annotation is the "@Table" which is used to declare that this entity database table name is "users". Furthermore, the "@UniqueConstraint" annotation we use to make sure by the server that the users don't insert duplicate username or email. For the user class, I extended DateAudi class that add two more columns to add the end of user table which is the create_at and updated_at. The "@Id" annotation is used to declare that the Long id variable is the primary key of the User table. The "@GeneratedValue(strategy = GenerationType.IDENTITY)", helps to generate auto-increment the id. Another one is the "@NotBlank" annotations which are used to state that the fname column should not be empty. And the "@Size(max = 40)" annotation tells the Spring application not to exceeds first name beyond 40 characters.

8.6 Front end (React)

For the front – end I have used the React java-script library it one of the famous libraries used by many developers. This framework has a great life span since it is developed by Facebook Developer. There is also a great community in all platform to get help in this framework. Comparing with Angular the major difference is that Angular is a framework and React is just a library. Similar to Angular in React also we break down everything into smaller components. The biggest advantage of separating into components is that we can always import the component without rewriting the code. In React we can also use Node Package Manager (NPM) which it holds thousands of javascript libraries.

8.6.1 Libraries used in React

8.6.1.1 react-router-dom

This library is a DOM binding for React Router it consists of BrowserRouter, Route, Link, NavLink, Switch and etc. These are component used to do routing for every web page. For example, to show user dashboard when they type a URL like this www.scms.com/dashboard/user/0001.

8.6.1.2 react-barcode-reader

It is a component that helps to read barcode from the barcode devices that are working as a keyboard. This component is actually made for barcode readers but they work for the RFID reader too because they also act as a keyboard, therefore, this library works without any issues.

8.6.1.3 antd

Ant Design is a library with an enterprise level UI design language and React implementation. This library is very useful to create complex UI in a short time it really saved my time I really recommend it to use others and I would definitely use in the future.

8.6.1.4 axios

Axios is a simple library that makes API calls save and easy it is a promise-based HTTP client for the node.js and browser. It gives full support on Promise API. It automatically converts for JSON data. And it even protects against XSRF (Cross-site-request forgery).

APIUtils.js

The below screenshot shows coding part of the file APIUtil.js. In this file have all the API Request set to called throughout the application. The request function sends an API request with a parameter or URL, method, and data. Before the request is sent, I need to set the “Content-Type” to “application/json”. Thereafter we get access JWT token from the local storage which is already set when we sign in. Then we set a second header called “Authorization” and pass a value as “Bearer <Access Token>”. After this we can send with a request with the appropriate URL now the Spring Application would verify if the token is valid and allow us to make all request.

```
import { API_BASE_URL, ACCESS_TOKEN } from '../constants';

const request = (options) => {
    const headers = new Headers({
        'Content-Type': 'application/json',
    })
    console.log(localStorage.getItem(ACCESS_TOKEN));
    if(localStorage.getItem(ACCESS_TOKEN)) {
        headers.append('Authorization', 'Bearer ' + localStorage.getItem(ACCESS_TOKEN))
    }

    const defaults = {headers: headers};
    options = Object.assign({}, defaults, options);

    return fetch(options.url, options)
    .then(response =>
        response.json().then(json => {
            if(!response.ok) {
                return Promise.reject(json);
            }
            return json;
        })
    );
};

export function getAllUsers() {
    return request({
        url: API_BASE_URL + "/users",
        method: 'GET'
    });
}
```

Figure 45 Code snippet from APIUtils.js

pom.xml

The pom.xml is the file where we mention all the libraries, we need for this application to run. For the first time of application installation we need to call the "npm install" command after the npm would search for the pom.xml file and install all dependencies mention in the pom file.

Figure 46 pom.xml (library dependency)

PrivateRoute Component

```
import React from 'react';
import {
    Route,
    Redirect
  } from "react-router-dom";


const PrivateRoute = ({ component: Component, authenticated, ...rest }) => (
    <Route
      {...rest}
      render={props =>
        authenticated ? (
          <Component {...rest} {...props} />
        ) : (
          <Redirect
            to={{
              pathname: '/login',
              state: { from: props.location }
            }}
          />
        )
      }
    />
);

export default PrivateRoute
```

Figure 47 PrivateRoute React Component

The above coding snippet is one of the components in react. The component name is PrivateRoute apart from the normal route from the react-router-dom this route allows only authorized user to enter the specific route if the user is not authorized and could not be validated by the PrivateRoute component it would not allow to enter and redirect to the login path. This helps to prevent unauthorized access to the system layout. And protect also hackers from viewing the coding.

Queue algorithm

```
console.log("1" + this.state.queuelistgeneralmedicine);

    getAllActiveQueues().then((response) => {
       for (let i = 0; i < response.length; i++) {
            if (response[i].doc_specialization === 'General Medicine') {
                    this.state.queuelistgeneralmedicine.splice(0, this.state.queuelistgeneralmedicine.length);
              getPatientsbyPatientsId(response[i].patientid).then((response1) => {
                let count = 1;
             if(i===0 && response1.status==='2'){
              console.log("inprogress");

               this.state.queuelistgeneralmedicine.push({
                 key: i,
                 no:count,
                 id: response[i].id,
                 patientid: response[i].patientid,
                 status: response1.status,
                });
              count++;
               this.forceUpdate();
            }else if(i===0 && response1.status==='1'){
              console.log("REady");

               this.state.queuelistgeneralmedicine.push({
                 key: i,
                 no: count,
                 id: response[i].id,
                 patientid: response[i].patientid,
                 status: 3,
                });
               count++;
               this.forceUpdate();
            }else{
              console.log("wait");
```

Figure 48 Queue algorithm code snippet

The above queue coding calls API function getAllActiveQueues and then loops all the response to filter the active queues by doctor's specialization called "General Medicine". After that, we empty the this.state.queuelistgeneralmedicine array. Then we call another API function called getPatientbyPatientsId() this function is called in order to get the current patient status. Thereafter we see a count variable that counts to rank the queue list. Then we see the if statement the condition tells if the patient is in the first row and the status is 2 then the patient is "InProgress" and in the second condition if the patient is in the first row and status is 1 then he is "Ready" to visit the doctor and in else statement it would say the patient to "wait".

RFID Handler

```
// RFID scanner handler
handleScan(data) {
  this.setState({
    result: data,
  })
  if(this.state.currentUser.usertype === 'rfid'){
      console.log(this.state.patientinfo);
        getPatientsbyPatientsId(data).then((response) => {
          if(response.status===0){
            const queue = {
              patientid: data,
              doc_specialization: response.doc_specialization,
            };
          //Create new Queue
          createQueue(queue);
          //Update patient status
          updatePatientStatus(1, response.clinic_id);
            message.success('Patient Registered ' + data);
          }else{
              message.error('Patient already scanned');
          }
          this.state.patientinfo = ({
```

Figure 49 RFID handle code snippet

The above code snippet shows the handleScan function which is executed right after a successful RFID scan. Once the function is executed it checks if the current user is user-type “RFID” if yes then it would get the particular patients information and compare if the patient status is equal to 0 if it is equal to 0 it would get the RFID id and create new queue in the database and update the patient status to “1”. If the patient status is not 0 it would return a message that the “patient is already scanned”.

Likewise if the user type is equal to “doctor” it would redirect to

“APIBaseURL/patient/dashboard/<RFID ID>”

ServerError Component

```
import React, { Component } from 'react';
import './ServerError.css';
import { Link } from 'react-router-dom';
import { Button } from 'antd';

class ServerError extends Component {
    render() {
        return (
            <div className="server-error-page">
                <h1 className="server-error-title">
                    500
                </h1>
                <div className="server-error-desc">
                    Oops! Something went wrong at our Server. Why don't you go back?
                </div>
                <Link to="/"><Button className="server-error-go-back-btn" type="primary" size="large">Go Back</Button></Link>
            </div>
        );
    }
}
                class ServerError
export default ServerError;
```

Figure 50 ServerError Component in React

The ServerError component is simple component that will be executed in case if there are server related issues. It displays the error with “Go back button”.

8.7 API Design and Explanation

No.	Resource	GET (Read)	POST (Create)	PUT (Update)	DELETE (delete)
1.	/api/auth/signin		Login		
2.	/api/auth/updateUserstatus		Update user status to “1” or “2” or “0”.		
3.	/api/auth/signup		Creating a new user		
4.	/api/bodylocationlists		Create new bodylocation item		

5.	/api/bodylocationlists/ bodylocationlistsbyclinicid	Get list of bodylocation based on clinic id			
6.	/api/clinics/{clinicid}	Get All clinic information and bodylocationlist, symptomlist, issuelist, patienttestresult list and prescriptionlist			
7.	/api/clinics		Create new clinics which would add new bodylocationlist, symptomlist, issuelist, patienttestresultlist and prescriptionlist		
8.	/api/clinics/clinicsbypatid/ {patient_clinic_id}	Get only clinic lists of a particular patient ID			
9.	/api/files		Uploading patient test result file to AWS S3 Bucket		
10.	/api/delete				Removing uploaded

					patient test results
11.	/api/issues/	Get all issues list to feed select option list			
12.	/api/medicals	Get all medical list to feed select option list			
13.	/api/symptoms	Get all symptom list to feed the select option			
14.	/api/tests	Get all test list to feed select option list			
15.	/api/patients		Create new patient		
16.	/api/patients/{patientId}	Get patient information based on patient id	Update patient using patient id		Delete a patient with patient id
17.	/api/patients/clinic/{clinicId}	Get patient information based on RFID number			
18.	/api/patients/updatePatientStatus		Update patient status from “1”, “2” or “0”.		
19.	/api/queues	Get list of queues	Create new queue		

20.	/api/queues/close		Close queue by updating status to “1”		
21.	/api/queues/activequeues	Get all active queues			
22.	/api/queues/getlatestactivequeue /{special}	Get the latest active queue based on doctor specialization			
23.	/api/users	Get list of users			
24.	/api/users/{userId}		Update user based on ID		Delete user based on user ID
25.	/api/users/one/{id}	Get single user information based on user ID			
26.	/api/user/me	Get summary of a the logged in user			
27.	/api/user/checkUsernameAvailability	Get true or false. It checks whether the username is available.			
28.	/api/user/checkEmailAvailability	Get true or false. It checks whether the email is available.			
29.	/api/users/{username}	Get some information of the particular			

		user for the Profile layout.			

8.8 Deployment instruction

Deployment is uploading hosting the actual web application online in the World Wide Web.

8.8.1 Back-End deployment instruction

Step 1

Download and Install Heroku CLI

Step 2

Login using the email and password

Step 2.1

Enter “heroku login” command in CLI.

```
heroku login
```

Step 2.2

It will ask for credentials. Once logged in it will say that you are logged in displaying the email.

```
Enter your Heroku credentials:
Email: email@example.com
Password: **************
Logged in as email@example.com
```

Step 3

Upload the spring application to the local git repository.

Step 4

In Heroku cli type “heroku create”. Heroku chooses a unique name itself for the app which can altered later.

```
$ heroku create
Creating app... done, ⬢ apple-custard-79879897
https://apple-custard-79879897.herokuapp.com/ |
https://git.heroku.com/apple-custard-79879897.git
```

Step 5

Now we have to rename the project to our project

heroku apps:rename --app <OLD_NAME> <scms>

Step 6

Deploy SCMS into Heroku by simply typing the below command. It will push the project will to the Heroku master created above.

```
git push heroku master
```

Step 7

Once the long deployment is finished you can type the below command to check if the spring application works

```
heroku open
```

8.8.2 Front-End deployment instruction

Step 1

We need to create git hub account and push the react application.

(Note. gitignore - node)

Step 2

Once the application is upload, we can buy any web hosting and push the application to their server.

Step 3

Once the application is pushed to the server, we need to run the command npm install it will install all the package dependencies into the server itself.

Step 4

Change the API_BASE_URL from the local host to the URL where we have uploaded the Spring Application in Heruko cloud platform. (inside application folder src -> constant -> index.js)

As shown below in the screen shot

```
export const API_BASE_URL = 'http://localhost:5000/api';
//export const API_BASE_URL = '/api';
```

Figure 51 API_BASE_URL for spring integration

8.9 System Installation Guide

Since this a web application there no need of an installation just with a correct username or email and password it can be access.

9 Risk Management

In the field of software development, we can have many risks involved those risks we need to identify, analyze, plan, track, and control. Considering risk management in software development helps to control the risk and ensure a successful project. By planning risk management, we can inform our entire team and make sure that they are ready to act when the risks occur. Usually, the project manager is the person who monitors risk during the software development.

No	Risk Description	Probability of Occurrence	Loss Size (Days)	Risk Exposure (Days)
1.	Lack of sample data to test and validate.	45%	6	2.7
2.	Since the computer literacy rate is low, we have to face a risk of providing more effort on the staff/user guiding.	80%	4	1.6
3.	Not able to test in the real-world scenario	10%	5	0.5
4.	Not enough Testing time to validate on all browser and OS types.	45%	6	2.7

In the above Risk Management Plan, we have five columns name no, risk description, probability of occurrence, loss size and risk exposure. The no columns are used to identify the risk by there numbers. And the risk description gives a small explanation of the risk. The probability of occurrence guesses the probability of this risk could actually occur. Measuring and stating the negative impact to a project is called Loss Size which usually mentioned in hours or days, The Risk exposure is a kind of a measure that would consider the potential future loss that could result from the specific risk.

10 Project Summary

10.1 Solution Evaluation

The solution to creating a Smart Clinic Management System using RFID was a success. The web application works as it should be and all the planned solution are done even better than initially planned. Therefore, I can confidently say that the solution has been given for the mentioned problem.

10.2 System Limitation

The Smart Clinic Management System has a certain limitation. The **first limitation** is that is very unpractical to use the web application on mobile devices because of large tables and information. The **second limitation** is that the application will not work if there is no internet or electricity. Because of this reason only I insert every queue request to the database rather than in spring memory. While inserting a queue request in the database I have to set a Cron job to automatically truncate the queue table every midnight.

10.3 Future Enhancements

In the future, I would like to enhance the Smart Clinic Management System to make patient waiting for 5 minutes to meet the doctor. For this, I have to develop and mobile application to make the patient book their clinic appointment from their home. And I would develop a website to allow patient to view their clinic information from anywhere using their login credentials. Using advance RFID reader, the system would detect the patient RFID card within 20 meters and checking in the patient system whether he has booked for the clinic we can identify that the patient came for a clinic session, therefore, the system would automatically add the patient to the queue right after he enters the lobby.

Allowing prescription list private API to authorized pharmacies allows us to buy medicals from other private pharmacies when it is not available in the government hospital pharmacy. This helps to solve the problem of the badly written prescription list by a doctor.

We can also inbuilt this system to a robot to make the patient interact with the robot for asking question and information which makes to reduce more staffs. In the future

Another future work is to identify the problem with data of existing body location, issue, symptoms and test result using Artificial Intelligence. This could help to identify problems that even experienced doctor cannot identify. But this, however, must be tested for years before allowing it into the real-world application.

10.4 Lessons learned

In this final project, I have applied almost all the knowledge that was appropriate for this project to use. Rather than this I know I have learned a ton of information. In developing this project learned about a new library called React which is a really great library I definitely look forward to using it in my future project. Since the React library is basically breaking down everything in component I have also learned about the component, props, state, es6 and etc. Since I have mastered react it would be easy also to learn Angular framework too. I also learned to create enterprise-level API gateway using spring boot 4 which would definitely help in my future work. The usage of JWT token between client and server also I have learned which would be very useful in future software development. Moreover, the usage of JSON in both front end and back end thought me lot things like listing and merge multiple listing and etc. I also learned to use POSTMAN to test the API with populating header, authorization and etc. Another thing is AWS S3 Bucket for me cloud storage is very new I have already heard about in many blogs but I thought I would be very difficult and expensive to set up but however I definitely wanted to learn about AWS Cloud storage and I am taking the decision to integrate my patient test result into AWS S3 Bucket. Actually, I got it successfully working in a day which is very easy. And here after I look forward to using cloud storage without any problem. Apart from this my skill of fixing bug has increased a lot I have come across many problems in both front – end and back end too but however, I have fixed and got the application run smoothly.

10.5 Conclusion

At last the Smart Clinic Management using RFID works as planned and has gone through all the relevant testing. All the API work perfectly fine considering the

security of the system with JWT token is also made sure. At the start of this project, I had great confidence that I will be able to finish the system on time perfectly as that it happened. As already said, I have learned a lot in developing this system and gave me great knowledge for my future work and strongly consider to make more project likes for my own use. I hope this system would be very useful for all the hospital on the island. And I also like to enhance the system further with face recognition and using artificial intelligence and machine learning to push the technology further.

10.6 References

Ahmad BA, K. K. F. A., 2017. An assessment of patient waiting and consultation time in a primary healthcare clinic. *Malays Fam Physician,* 12(12), pp. 14-21.

Anon., 2012. *Sri Lanka Census of Population and Housing, 2011.* [Online]
Available at:
http://www.statistics.gov.lk/PopHouSat/CPH2011/index.php?fileName=pop42&gp=Activities&tpl=3
[Accessed 8 May 2019].

Anon., 2019. *Amazon Web Service.* [Online]
Available at:
https://aws.amazon.com/free/?sc_channel=PS&sc_campaign=acquisition_IN&sc_publisher=google&sc_medium=ACQ-P%7CPS-GO%7CBrand%7CDesktop%7CSU%7CCore%7CCore%7CIN%7CEN%7CText&sc_content=Brand_Core_HV_e&sc_detail=aws&sc_category=Core&sc_segment=293607067653&sc_ma
[Accessed 24 February 2019].

Anon., 2019. *JWT.* [Online]
Available at: https://jwt.io/
[Accessed 5 April 2019].

Anon., 2019. *Spring.* [Online]
Available at: https://spring.io/
[Accessed 24 February 2019].

Anon., 2019. *Spring Boot Tutorial.* [Online]
Available at: https://www.tutorialspoint.com/spring_boot/index.htm
[Accessed 5 February 2019].

Banks, A. & Porcello, E., 2017. Learning React: Functional Web Development with React and Redux. 1st ed. s.l.:O'Reilly Media.

Crookshanks, M. E., 2017. Just Enough Requirements and SDLC: Requirements Documentation, Waterfall, and Agile. 1st ed. s.l.:CreateSpace Independent Publishing Platform.

Existek, 2019. *SDLC Models.* [Online]
Available at: https://existek.com/blog/sdlc-models/
[Accessed 5 May 2019].

Gulabani, S., 2015. *Amazon S3 Essentials.* 1st ed. s.l.:Packt Publishing.

Hinkula, J., 2018. Hands-On Full Stack Development with Spring Boot 2.0 and React: Build modern and scalable full stack applications using the Java-based Spring Framework 5.0 and React. 1st ed. s.l.:Packt Publishing - ebooks Account.

Jaffna, T. H., 2019. *Teaching Hospital Jaffna.* [Online]
Available at: thjaffna.lk
[Accessed 9 May 2019].

Jin, B., Sahni, S. & Shevat, A., 2018. *Designing Web APIs: Building APIs That Developers Love.* 1st ed. s.l.:O'Reilly Media.

Making, S., 2018. *Abstact Factory Design Pattern.* [Online]
Available at: https://sourcemaking.com/design_patterns/abstract_factory
[Accessed 4 August 2018].

Murray, A., 2016. *Information Technology Law: The Law and Society.* 3rd ed. s.l.:Oxford University Press.

Patton, R., 2005. *Software Testing (2nd Edition).* 1st ed. s.l.:Sams Publishing.

Rafat Mohebbifar, E. H. M. M. M. S. O. K. H. M. I., 2014. Outpatient Waiting Time in Health Services and Teaching Hospitals: A. *Global Journal of Health Science,* Volume 6, pp. 172 - 180.

Steve Gallivan, M. U. T. T. O. V., 2002. Booked inpatient admissions and hospital capacity:. *Information in practice,* Volume 324, pp. 280-282.

Annexure

Turnitin report

20147493 _Elankeswaran_CIS6002.docx

by Elankeswran Sujeban

Submission date: 01-May-2019 06:06PM (UTC+0100)

Submission ID: 106004715

File name: 58858_Elankeswran_Sujeban_20147493__Elankeswaran_CIS6002_530653_1957858906.docx (3.49M)

Word count: 15552

Character count: 76516

1 Acknowledgment

I would like to thank Mr. Sriharan my lecturer and supervisor for this project he has supported me throughout the project giving me advice, tips and tricks. He also helped [redacted] finish [redacted] time. Apart from [redacted] Cardiff Metropolitan University UK and ICBT Campus.

Last but not least I would also kindly thank my friends and family for their kind support throughout completing my project

Table of Contents

List of Acronyms

SCMS – Smart Clinic Management System

JS – JAVA Script

AWS – Amazon Web Server

API – Application Programming Interface

JSON - JavaScript Object Notation

RFID - Radio-Frequency Identification

FIFO – First in First Out

WBS – Work Breakdown Structure

DFD – Data Flow Diagram

ERD - Entity Relationship Diagram

UI -User Interface

DB – Database

MySQL – My Structured Query Language

2 List of figures

3 Abstract

This document is all about the Smart Clinic Management System. The system uses the help of RFID readers to digitalize [redacted] for the Northern Province of Sri Lanka). The current process of clinic management is done manually where clinic patient brings an 80 pgs. copy with the fear of losing the copy. If the patient is already register for a daily checkup or also called clinic need to give the clinic copy to the attender in order to register for the queue. The given copy would be stacked in FIFO order and separated according to the doctor's specialization. At the doctor's consultation also the treatment and details of the patient and prescription are handwritten on the clinic copy only. There many issues involved in the manual system which is discussed deeply below. To solve this problem I have come to a solution to implement a Smart Clinic Management System where all the patient details are store in cloud storage and the only thing the patient need to bring is the RFID card and in case if it lost there is no problem with small fee the patient can get a new one without worrying about anything. One more solution is reduced waiting time and give the patient more possibilities to do within the waiting time. Since the patient would be notified via SMS before two patients to be ready in the lobby. This help the patient to wait in the outside park go to the nearby shop and etc.

4 Introduction and Background

4.1 Introduction

The Smart Hospital Management System using RFID is a digital process of the current clinic management in Government Hospital of Jaffna.

The current manual process in the Government Hospital Jaffna is that the patient must file their clinic book in a stack method (First in First Out). In a day there would be two times where the patients can go for a check-up one in the morning and one in the afternoon. Approximately more than 400 people will gather for a check-up for morning and afternoon check-up. And the clinic book is not computerized they have it all in a paper. This is not secure from natural disaster and not secure from the theft. This can also cause misuse of clinic books because any person can write on the clinic book without any evidence. In some cases, we also identified that there is an issue in identifying the prescription from the doctors by the pharmacists, this could be dangerous for the patient.

To solve the above problem, I have come up with an idea of a smart clinic management system. This system enables the patient to scan on the RFID reader which is also a stack method (First in First Out). After scanning the card, the patient receives an SMS which will show the token number and approximate waiting time. If the patient has enough time he can go out to a nearby shop or sit outside the hospital park than waiting inside the hospital. A second message would be sent to the patient that he would be the next to go to the doctor after three patients. And a third message which shows the room number that the appropriate patient must visit. Soon after the patient is called the appropriate doctor would see all the patient information on the computer and the doctor can even greet the patient with the name to make the patient more comfortable and to gain trust. Inside the system, the doctor can see all the past information of the patient including all the X-ray, Blood report, Cholesterol test, Urine test and etc. This helps the doctor to find the right treatment and cause much easier. Once the doctor has found the cause he can prescribe the medicals on the system itself. After that, the patient would go to the pharmacy and before the patient arrives at the pharmacy counter the pharmacist would have prepared all the medicals for the patient and even here, we can save the patient waiting time and avoid the issue

of not understanding the doctor's prescription. The doctor prescription would be sent as an API request from the Smart Clinic Management System and integrated with their pharmacy application system.

4.2 Background

Jaffna Teaching Hospital is a full government hospital in Jaffna, Sri Lanka. This teaching hospital is a leading hospital in the northern province of Sri Lanka which is controlled by the central government in Colombo. It also acts as the main clinic teaching faculty for the University of Jaffna Faculty of Medicine. Apart from this hospital has 1236 beds, 21 ICU beds, 61 special units, 13 operating rooms. Jaffna Teaching Hospital consists of 1500 staffs which are working 247/7 in three shifts. Out of 1500 staffs whom 76 are a consultant in their special fields. In a daily basis the hospital treats around 4500 patient a day. This hospital has many health service such as Anaesthesiology, Cardiology, Chemical Pathology, Dental, Dermatology, Diabetes & Endocrinology Unit, ENT, Gastroenterology, Haematology, Histopathology, Judicial Medical Unit, Medical Unit, Microbiology, Nephrology, Neurology Unit B, Neurosurgery, Obstetrics and Gynaecology, OPD, Ophthalmology, Oral & Maxillofacial Unit, Orthopaedic, Paediatric, Plastic Surgery, Psychiatric, Radiology, Respiratory Unit, Rheumatology, Surgical Unit and Urology. Since it is the leading hospital in northern peninsula it is 24/7 open. (Jaffna, 2019)

In Jaffna Teaching Hospital we can go for a clinic consultation in morning and evening. As already stated, this is the leading hospital in the northern peninsula with a population of 1,058,762 (Anon., 2012) peoples. This amount of population may come for the Jaffna Teaching Hospital for special treatments. It has also the facilities to go for a monthly or weekly check-up called clinics. The daily number of patients attending for the clinic would be around 100 or more people they all go for different doctors who are specialized for the type of health issues. Currently, the patient must file the clinic card where all the patient past records are recorded. The clinic files taken to the queue of FIFO (Frist In First Out) approach. After a certain time, the attended would call the patient according to FIFO to manage all the patient to write doctors more the five attendees or nurses needed to manage the crowd efficiently.

Since the patient doesn't know how long and after how many patients, they would be called they have to sit near the clinic's room which usually uncomfortable because it could take more than 2 hours in a busy day. And the keeping patient data in a piece of a small copy if the not correct way in the digital age. To overcome these two issues, I have come up with a web application that would solve the problem.

4.3 Literature review (Areas Depth)

In research paper called "An assessment of patient waiting and consultation time in a primary healthcare clinic" states that patient waiting for the doctor is common is issue around the world but however the waiting time differs with various factors such as whether the particular country is developed or not, government hospital or private hospital and etc. This particular research was conducted for 4 weeks of audit in a primary health care clinic. They state that according to their analysis a patient must wait 41 minutes on average until he gets consulted by the doctor. And the average consulting time would take 18.21 minutes. If break down the 41 minutes process there are a number of processes such as time for registration, pre-consultation, appointment and etc. The primary clinic has an average of 60 patient attendances per day. They have their own Queue Management System which basically records the time in and out of every station. A diagram of this research paper states 91.93% of the patient wait less than 90 minutes. And the consultation time is for almost all the patient is less than 30 minutes. (Ahmad BA, 2017)

Overall according to the researchers, they say that clinic waiting time must be improved to give the best service to the patients. According to the above problem, I can reduce the number of waiting time with Smart Clinic Management System this system eliminates the time for an appointment, the time for registering, and even the time at the pharmacy because right after doctor consultation prescriptions are passed to the pharmacist to make ready all the medicals for the patients.

Another research paper called **"Outpatient Waiting Time in Health Services and Teaching Hospitals: A Case Study in Iran"** states that Iran's educational hospital takes an average time of 161 minutes for the patients which is more than the above

research paper. According to their research paper, it states that between the age of 26-40 has the highest distribution of age patients. In this researched hospital they use the FIFO approach for a queueing system which usually done in every hospital. To provide the best service to the patients is very important that researchers states. According to the research they have found that there are three main factors that affect the waiting time of patients such as [redacted] and shortage of professional staff. (Rafat Mohebbifar, 2014)

In summary, the above research paper shows there is a huge waiting time for the patient which is 161 minutes which is too long for the patient and this not called quality service. These times are affected by the registration process, shortage of physician and shortage of professional staff. In the Smart Clinic Management System has a great advantage because the patient registers only once and after that, they just need to scan the RFID at the RFID reader to register therefor, we reduce almost 15 minutes already and staff is also reduced. Since the system has been implemented with SMS gateway the patient actually can go nearby shop or outside the hospital garden to relax. Once the patient is the next after two patients, he gets an SMS saying that he needs to wait at the lobby. Shortage of physician is a problem that has to be solved by hospital management itself and the same as professional staffs.

Another research paper titled **"Booked inpatient admissions and hospital capacity: a mathematical modeling study".** This paper tells that unlike other operation, hospital operation is most complex because it always has to do with high capacity and overload. Three factors make very difficult to manage the operation such as variability, blocking, reserve capacity, unpredictable variability. (Steve Gallivan, 2002)

The above paper highlights that hospital operation is very complex which is true and since almost all hospital have a great capacity of patients coming to the hospital, they need more staff to handle the process. However, since the above paper tells about booking system in a hospital the factors that affect the operation applies to the Clinic Management too.

4.4 Objective

The objective of this project **clinic patient by 15 min** and to **digitalize the manual process** of clinic management in Jaffna Teaching Hospital.

4.5 Scope

The scope of the smart clinic management system is to enable the clinic patient to attend the doctor consultation much **faster** than before and to **digitalize the current process** to make **patient information more reliable and secure** for future requirement.

4.6 Limitation

4.6.1 Sample size

Gathering samples from the government hospital is a limitation since the sample data are from the patient which is unacceptable to share the private information. Therefore, gathering sample size to test the application is limitation instead of this information we need to test with fake information's.

4.6.2 Equipment

Since the cost of high-end RFID Reader/Writer is expensive I am going with a cheaper device that has the same functionality only the lifetime and reading distances might differ.

4.7 WBS

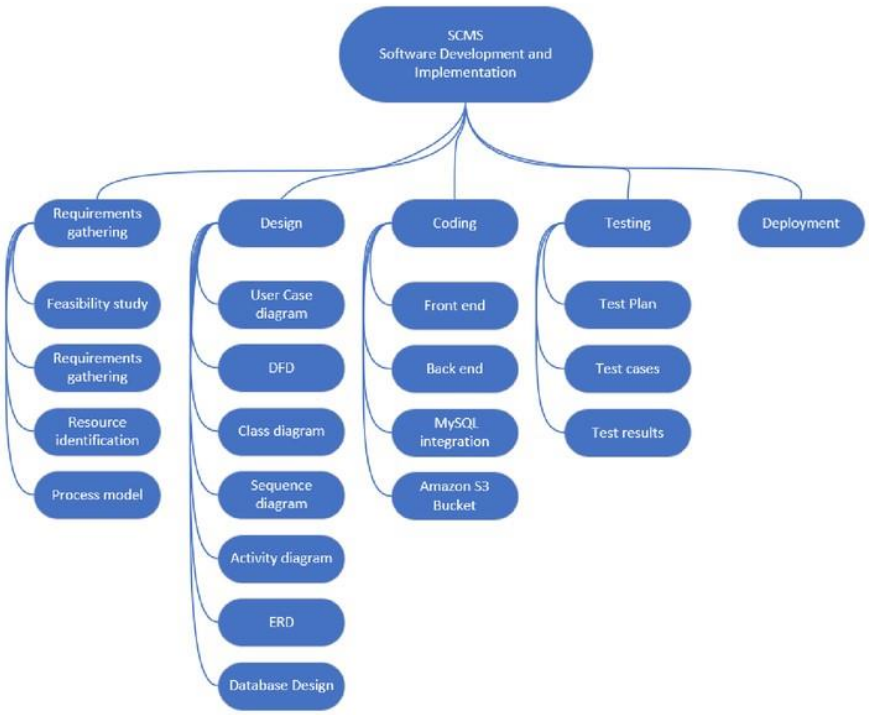


Figure 1 Work Break Down Diagram for SCMS

The above WBS Work Breakdown Structure is to show the main deliverables that I need to finish in order to build this system successfully. Basically, it is breaking a huge work into a smaller segment for completing in a smaller task as the name itself says.

Above we can see 4 main segments which are requirements gathering, design, coding the actual phase to build the application, testing the application and deployment instruction. All the main segments have smaller subsegments. For the Requirement gather task, we broke down into smaller task such as feasibility study, requirement gathering, Resource identification, and process model. The next phase is designed task which is broken down into eight tasks such as use case diagram, DFD, Class diagram, sequence diagram, Activity diagram, ERD and database design. The third phase is coding. In this phase, the actual development of the application starts to make

it simple we have broken down in smaller task such as front end, backend, MySQL integration, and Amazon S3 integration. After development, we need to test the application to validate that all the function and feature are working properly as they should do.

The last phase is the deployment task since the task is already small we cannot break it down further, therefore, we leave it as it is.

4.8 Resource allocation

#	Task	Optimistic (Hours)	Most likely (Hours	essimistic (Hours	Effort (hours)	Start date	End date	Status	Actual End Date	Key Dependency
1	**Initializing**				15.33	24 - Feb	28 - Feb			
	Prepare all the needed diagrams	7	10	10				Open		
1.1	Database Mapping (ERD)	5	8	10	7.83	3 - Mar	4 - Mar	Open		
1.2	Database Implementation	3	8	10	7.50	5 - Mar	5 - Mar	Open		1.1
2	**Architecture**				15.00	1 - Mar	2 - Mar			
2.1	System Architecture (Diagram)	8	16	18	15.00	6 - Mar	7 - Mar	Open		
3	**Development**				#REF!	2 - Mar	20 - Jun			
3.1	**Front-end development**				88.00	2 - Mar	13 - Mar	Open		
3.1.1	AngularJS environement setup							Open		1.2
3.1.2	Login UI							Open		1.2
3.1.3	Dashboard UI							Open		1.2
3.1.4	CRUD Doctor							Open		1.2
3.1.5	CRUD Patient							Open		1.2
3.1.6	Adding all kinds of document of patient							Open		1.2
3.1.7	RFID Setup and integration							Open		1.2
3.1.8	Patient Real-time Queue Algorithm							Open		1.2
3.1.9	Allocating Patient to Doctor (Algorithm)							Open		1.2
3.1.10	Real-time waiting list display for outdoor monitor							Open		1.2
3.2	**Back-end development**				200.00	18 - Mar	11 - Apr	Open		
3.2.1	Spring boot environment setup							Open		1.2
3.2.2	Cloud integration							Open		1.2
3.2.3	User Login and JWT Token Integration							Open		1.2
3.2.4	Bodylocation API							Open		1.2
3.2.5	Issue API							Open		1.2
3.2.6	Symptom API							Open		1.2
3.2.7	Patient Test Result API							Open		1.2
3.2.8	Prescription API							Open		1.2
3.2.9	Clinic API							Open		1.2
3.2.10	Patient API							Open		1.2
3.2.11	Doctor API							Open		1.2
3.2.12	User Role							Open		1.2
3.2.13	Validation							Open		1.2
4	**Testing**				32.00	14 - Apr	18 - Apr	Open		
4.1	Error handling	10	16	22	16.00	3 - Apr	4 - Apr	Open		1,2,3
4.2	Performance testing	10	16	22	16.00	5 - Apr	6 - Apr	Open		1,2,3
5	**Implementation**				8.00	18 - Apr	22 - Apr			
5.1	Software and Hardware Requirement	1	2	3	2.00	7 - Apr	7 - Apr	Open		1,2,3
5.2	Deployment Instruction	1	2	3	2.00	7 - Apr	7 - Apr	Open		1,2,3
5.3	System Installation Guide	2	4	6	4.00	7 - Apr	7 - Apr	Open		1,2,3

4.9 Milestone Plan

#	Milestone(s)	Planned(End) Date	Status	Actual Date	Comments
1	Initializing	24 - Feb	Open		
2	Architecture	1 - Mar	Open		
3	Front-end development	2 - Mar	Open		
4	Back-end development	18 - Mar	Open		
5	Testing	14 - Apr	Open		
6	Implementation	18 - Apr	Open		

4.10 Cost Plan

No.	Item name	Price (£)
1	RFID Reader/ Write (1x RFID card)	8.54
2	Laptop	300
3	Hosting (one month)	2.14
4	Cloud storage (Amazon Web Service S3 Bucket)	Approx. 5
5	Domain (one month)	7
6	For 126 SMS (SMS Gateway)	8.95
	Total	331.63

5 The System

5.1 What kind of a system is it?

The system is called a Smart Clinic Management System it is basically a web application. This System will contain all the information of the patient who is coming for a regular check-up. Each patient would have an RFID card for identification purpose. The system stores all the information about patient treatment, prescription, reports, scans and etc. This will enable the doctor to have all the information in one view to give better treatment and this information would also help to identify the disease that is increasing the country in order to take appropriate action as soon as possible. The patient receives an SMS containing the token number and the name of the room that he needs to visit. When entering the correct room, the doctor would automatically see the patient information on the desktop and once he leaves the room the doctor would prescribe the medical on the system. After that, the patient would go to pharmacist and scan once again to get the prescription list from the system. The patient's information is stored entirely on the cloud which is more secure and less expensive. Moreover, the important part of this system is the queue system. The queue system enables the patient to scan his RFID card on the reader to register in the queue and he could wait in the lobby or outside the park after specific time like before three patient a message would be sent to the patient to wait in the lobby for doctor consultation. This enables to get rid of long queues in the bench and it also gives equality among public usually if the staff is known for a patient, he would give priority and allow him first, therefore, this system could solve the problem while it is fully digitalized.

5.2 Why is this system important?

This Smart Clinic Management System is important to **solve long-standing queue** in Jaffna Teaching Hospital because all the patient who comes to the clinic are aged patient and according to their disease, they need to get food and drink on time, therefore, **reducing the waiting time** is a big movement for the sake of the patient. Moreover the patient carries a small 80-page copy to record their clinic details and patient test result are also held by patient this **causes loss of copy and test result**

when the patients are aged and might **easily forget the clinic copy or lose eventually**, therefore, I also implemented add clinic and patient test results into Smart Clinic Management System to solve this issue. Another important thing is that the patient can get reports of an emerging disease in a particular hospital which helps full to take appropriate actions for the government. Implementing this system can also eliminate other issues such **as duplicate data, lack of security, unproductive by implementing a large number of staffs, risk of data and the possibility of giving wring medicals to the patient because of unclear prescription of a doctor**. Considering all this factor it could be imagined why the SCMS so important.

5.3 User Requirements

55

The user requirements of the Smart Clinic Management System are that all the eight users need to have specific permission to change only the appropriate information's. In this system, we have many user requirements which are listed below according to its user type.

Admin – The admin has the privilege to create, edit, update and delete users in the system.

Director – The director has permission to view the hospital reports and staff details.

Doctor – The doctor has permission to view the patient's full information and view patient test result. More importantly, he is the only one user type that has the privilege to create a clinic session for a patient.

Laborist - The laborist has the permission to view patient test result needed list and to upload patient test result to the system

Attender – The attendee has the feature to create a new patient for the clinic by validating all the real evidence.

Nurse – The nurse has the privilege to view the patient clinic information only.

RFID – The RFID is not a real person who is going to take control of the system it just used for the RFID reader and to show the Real-Time Queueing system.

Pharmacist – The pharmacist has only the privilege to get prescription by id only which will be used to integrate with the existing Pharmacy Management System.

5.4 Current manual system explanation

In Jaffna Teaching Hospital the outside patient comes for a daily clinic check up to the respective doctor. The patient carries a small 80-page copy which is provided by the Jaffna Teaching Hospital. When the patient wants to register to consult to a doctor he has to approach to an attendee and say to him that he needs to visit the doctor. The attendee then takes the patient's clinic copy and files in a FIFO (First in First Out) method and according to the doctor's specialization. Once all the doctors have arrived the attendee will start to call out the patient's names in a FIFO manner. Until this process, the patient cannot go anywhere other than sitting on the bench for hours. Finally, when the time has come to visit the doctor we have to go to the particular doctor and the attendee then gives the patient clinic copy to read previous clinic session and to write today's clinic session. Once the doctor has gone through consultation the patient might get prescribed some medicine on the clinic copy and some other information's. The medicine can be taken by the pharmacist which is inside the hospital by providing the clinic copy there are issues with the doctors handwriting I have heard from some pharmacist that they are unable to read the doctor handwriting which is really not a good sign for the patient because the pharmacist might give wrong medicine accidentally due to the worse handwriting.

5.5 Drawback of the current manual system

To go for a clinic in a Government Hospital of Jaffna we need to file all the patient's clinic book one by one in stack method and the patient are called in the same order. This, however, could take more than two hour waiting time.

Many people who are working in the weekday usually find very difficult to go for a clinic because of the long waiting time. And this generation of people usually doesn't waste their time on waiting. And overall the current process of clinic management is

manually done. Each and every patient has its own clinic book where all the details are entered by the doctors.

In case of lost clinic book, it very difficult and time-consuming to get all the records back from the manual process. And the patient previous reports and scan are giving to a patient only and if lost it is difficult and costly to get all the new test done. Due to the manual process, it could lead to duplicate data, space consuming, lack of security, unproductive, misplacement and risk of data.

The duplicate data might occur when creating a new patient file because the patient has lost his clinic book therefore, they would have two files of the same patient. All the patient information is recorded in a book and for doing this it will take many books while going through all these years. Therefore, having the recorded data in a book is very space consuming. Lack of security manual process is mainly considered as a lack of security because anyone can secretly access the patient record and change the way they want. Managing this clinic manually is considered as unproductive because more labors are needed for manual management. Documents can be easily misplaced in a hurry or tension. Risk of data is that paper document could be easily accessed therefor it is the risk of personal data.

Another issue is that I have encountered at the government hospital is that the doctor's prescription letter is sometimes difficult to read by the pharmacist this leads to a dangerous problem which is giving the wrong medicals to the patient that can cause serious side effect.

5.6 Explanation of the proposed system to solve the problem

This problem can be solved with the help of available technology such as with the help of web application we can create a management system to manage all the patient who comes for the clinic. Waiting time can be made much comfortable by sending the approximate waiting time and send SMS to the patient before that they are the next to see a doctor after three patients.

The manual process can be digitalized by storing all the information in the cloud. This will enable us to keep the information safe forever. It is kept safe from any natural

disaster and theft. The system cannot be misused because for each and every time someone accesses the patient personal data, it would record and store a log file with information who, when what and from whom he accessed the information. And even all the changes are recorded to detect misuse and investigation easier.

With this system we can avoid duplicate data because searching for a patient is much easier because of we able to search through a query. Having all the information stored in an AWS cloud storage it would take less space than before. While we have all the information on the cloud, we can assure that the data is in a secure place and the system would encrypt the information and then store to the cloud and database. This system reduces the labor force while it automatically shows the order on the main monitor in which room where a patient must go. The digitalization helps to prevent misplacement of patient's data. Inbuild log record helps us to see the users who access the patient personal data with all the information.

The pharmacist can also view the patient prescription when the patient RFID card is scanned. It can also allow API request from the Smart Clinic Management System so that can be integrated with the available Pharmacy Management System or Hospital Management System.

6 Requirement analysis

In the first step of the requirement analysis phase, we have to clearly analyze the problem and define the process to solve the problem. Moreover, this document is also used to define the expectation of the systems end user's requirement. A feasibility study is also done in order to make sure that everything is feasible for this system. The feasibility study includes whether the project is commercial, technologically, economic, legal, operational and scheduling feasible. Moreover, the resources required for the system are all identified as hardware, software, and other resources. The exact process of the system is also discussed.

6.1 Feasibility

6.1.1 Commercial feasibility

This Smart Clinic Management System can be a very profitable project for all the hospital sectors. Mostly in all hospital sectors in Sri Lanka waiting time is a major drawback. Therefore, both government and private hospital sectors could use this system in order to solve this issue.

6.1.2 Technical feasibility

The SCMS is a web application made using ReactJS as front-end and Spring Boot as back-end and cloud as storage. This gives the capability of creating a centralized database for the whole patient in the country. And the project is technically feasible because most of the tool is free.

6.1.3 Economic feasibility

For this project, we need to have a computer or laptop to start developing the web application in ReactJS and Spring Boot. Then we also need an RFID reader and RFID PVC card for the development process. Apart from this expense if we want to deploy this system to a real-world situation for example to a private hospital, we have an estimation of 0.94 pounds to register a patient to their hospital. And from the hospital side, we need to allocate for each room a sperate computer and an RFID reader. A sperate monitor is also needed to display the order of the patient in the waiting room.

6.1.3.1 Cost of developing the project

No.	Item name	Price (£)
1	RFID Reader/ Write (1x RFID card)	8.54
2	Laptop	300
3	Hosting (one month)	2.14
4	Cloud storage (Amazon Web Service)	Approx. 5
5	Domain (one month)	7
6	For 126 SMS (SMS Gateway)	8.95
	Total	331.63

6.1.4 Legal feasibility

The project is legally feasible because I can get permission from the Government Hospital of Jaffna to get an insight and the exact process of the patient going to the clinic. And for the development, I am going to use fake data because of the **Data Protection Act.**

6.1.5 Operational feasibility

The project would be operational also feasible because the end user would find it very **easy to manage** all the hundreds of patients. In the manual system, they need to call out each and every patient. This system has certain user groups such as the **doctor, pharmacist, nurse, director, laborist, doctor and admin.**

6.1.6 Scheduling feasibility

This project is all feasible in scheduling according to the System Development Life Cycle. It takes the below shown time:

1. Planning → 5 days
2. Analysis → 7 days
3. Design → 5 days
4. Development → 30 days
5. Testing → 10 days
6. Implementation → 3 day

Totally it would take 60 days to fully complete the system.

6.2 System Development Life Cycle (SDLC)

SDLC has mainly used in many IT industries it is basically a life cycle of steps to develop software. This process also helps to improve the quality of the software since we are going through all the important steps. The steps are planning, analyzing, designing, implementing, testing and integrating and maintenance. The above steps are the basic things to be done for any applications. But we have certain SDLC model for each situation most popular SDLC models are the:

1. Waterfall model
2. Iterative model
3. Spiral model
4. V-shaped model
5. Agile model

The **Waterfall SDLC** model is a process going from the phases of, requirement analysis, system design, implementation, testing, deployment, and maintenance. Waterfall model is mostly known to be used for the small project because they are easy to use and understand. For complex and object-oriented projects, it not recommended. Another drawback is that software is only ready after the last stage is over.

The **Iterative SDLC** model we get the chance to develop some function for system quickly at the beginning of the development lifecycle. The phases for this model are analysis, design, coding, testing and repeat and finally the implementation. The main advantage of this model is that we can parallelly develop the system and the development can be measured easily, easy to manage the risk since we can start to develop the high-risk task first. This type of model is not recommended for small projects. The disadvantages of the iterative model are that is difficult to manage the overall process.

The **Spiral SDLC** model is a combination of waterfall and iterative model. The main problem in this model is to find the right moment to make a step into the next stage. It has greater scalability because if we want to add new functionality, we can implement it even at a very late stage.

Another model is the **V-shaped SDLC** model it is actually an expanded version of the classic waterfall model. It is so called "Validation and verification" model it makes everything sure before moving to the next stage. This model is well fitted for a project that has clear and fixed requirements and it also a great advantage that testing takes place in the early stage. However, it has drawbacks too it has a lack of flexibility. And is not recommended for small projects.

The **Agile SDLC model** has become popular very recently. The main feature in an agile model is that the customer can inspect the result after every development iteration whether he is satisfied or not. From the customer perspective, it is a very good model to make sure that his product will be developed as he had planned. Since the change could vary in the future development it very difficult to estimate the resources and development cost in advance. The short weekly meeting is also compulsory to show the result to the customer.

Since my project is a mid-size project and I have a clear requirement I will go with the waterfall model. It is also simple and easy to get used to the model. The other models are more likely suited for development with a team or more complex and large size projects. For reference, a waterfall model diagram is shown below.

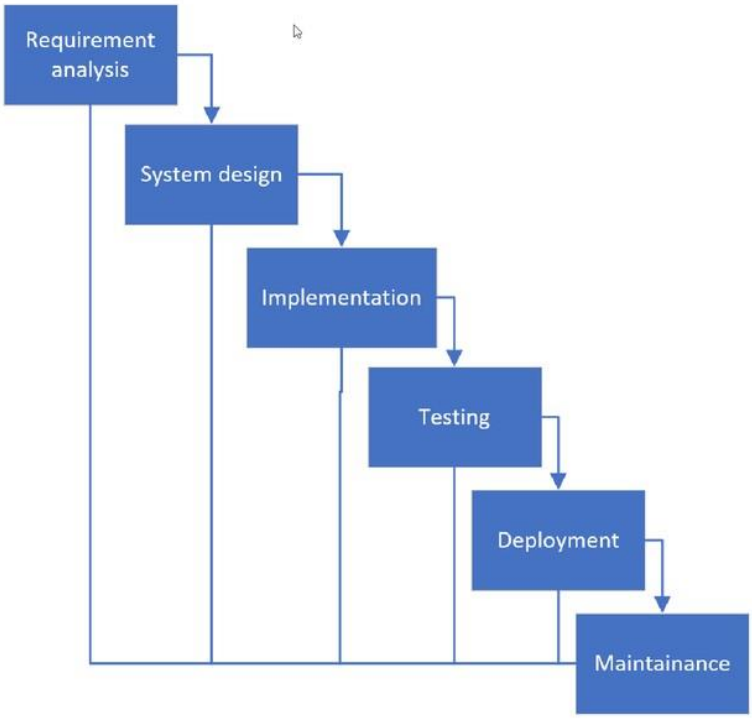


Figure 2 SDLC Waterfall diagram for SCMS

6.3 Requirement of Smart Clinic Management System (SCMS)

6.3.1 Functional requirements

In the SCMS we have

- Pharmacist
- RFID
- Nurse
- Attender
- Laborist
- Doctor
- Director
- Admin

The **Pharmacist** is allowed as a user only to check what medicals are available but we can also connect existing Pharmacy Management System with our API to GET the patients prescription list.

E.g. APIBaseURL +/api/clinic/prescription/{patientid}

The **RFID** is not a real user but work must make the system login in the early morning before the patient arrives for the clinic. The RFID is connected with large TV so that the patient can view their real-time progress of queue while sitting in the lobby. This also allows making the first scan of the patient to register to the queue system in order to display it on the monitor.

The **Nurse** can see only the information of the patient. The Nurse would need the system very rarely but I have added the nurse user type for the future purpose.

The **Attender** is the one who validates patient information manually and then it enters the system to create a new patient for a clinic and he can also edit patient information and delete patient's information.

The **Laborist** is the one who can see the patient clinic information and upload the patient test result.

The **Doctor** has the permission to view all patient and clinic information such as body location. Issues, symptoms, patient test result, prescription list and etc. And the doctor is one who closes the queue with a click of a button which is named as "closeSession".

The **Director** can only view the overall information such as the increasing disease, number of clinic patient, most used medicine, most popular test and etc.

The Admin has access to all the user permissions and has almost the full control over the SCMS but however, for a security purpose, admins are not allowed to view patient's clinic information. The admin can create new user, update a user and delete a user.

6.3.2 Non – functional requirement

Features and characteristic of Smart Clinic Management System.

6.3.2.1 Operability

The Smart Clinic Management System must have good UI and UX which makes any kind of users to understand the system since the workers in Sri Lanka has less computer literacy we need to make sure that the system is easy to understand and to operate in my side I have to make sure any inappropriate action is handled with exception to avoid unrelated data feeding.

6.3.2.2 Reliability

The reliability is another important factor since the system can also be implemented into other government hospitals upon how successful the system is. And handling multiple patient and doctor at the same time is also important for the system. We have to make sure that backend can handle multiple data at once.

6.3.2.3 Portability

Since we use web application for this system, we don't have portability issues for this system.

6.3.2.4 Response & processing time

The response and processing time must be much faster than the manual system. Since we use the queue system, we need to make sure that the response time and processing time are quick enough to handle a large number of users and patient at the same time.

6.3.2.5 Security

The [redacted] SCMS since we handle a large amount of patient data which complies with the **Data Protection Act**. Since we store the data in the AWS cloud, we can make sure that the data is kept safe. Apart from that, we use API Gateway which can only be retrieved or stored with a valid JWT token.

6.3.2.6 Simplicity

Keeping the system UI and UX simple as possible helps the user to understand without any tutor at all. Therefore, keeping the system simple helps to reduce the training time for the system.

6.3.2.7 Accuracy

The accuracy of information is also important because we handle sensitive information therefore, we must make sure to return success or error message after an insertion.

6.3.2.8 Efficiency

Since the system is going to run from morning to evening, we have to make sure that performance does not drop after greater run time.

6.4 Gathering requirements

Gathering requirement is to know what kind of features the particular system needs and functions, usability and etc. There are many techniques to gather information from system users. Every project has its own suitable technique.

Some of the examples are given below:

24

1. Brainstorming
2. Document Analysis
3. [redacted]
4. [redacted]
5. Prototyping
6. Survey
7. Questionnaire

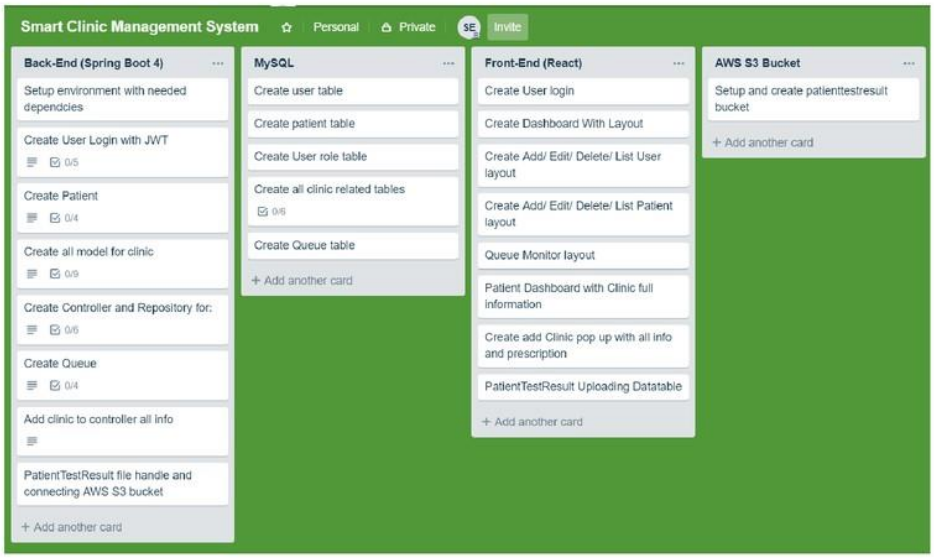
From above the example, I have used **brainstorming**, **questionnaire**, and **observation**.

Since I had already gone a couple of time to the clinic with my father, I know the process of going to the clinic which comes under **observation**. And an attendee working in Jaffna Teaching Hospital who is a relation of mine has supported me to answer my question all the **questions** are below.

1. How many patients would visit the clinic in a day?
2. How many divisions of doctors are available in the hospital?
3. How many staffs are needed for the clinic process?
4. Where is the patient test result taken?
5. Does pharmacist use management tools?
6. Is there any specific time to visit the clinic?
7. How is the current manual process?
8. What issue you have faced in a manual process?
9. What is the most popular disease?
10. Was there a situation where the patient says he has lost his test results?

At last, **brainstorming** is included in brainstorming special feature and how the current problem could be solved using the latest technology.

6.5 Project Management



The above is a screenshot of a web application named "Trello" it is actually projected management tool that can be accessed on any device and anywhere. The above project management show for the main list named Back-end (Spring boot 4), MySQL, Front-end (React) and Amazon S3 bucket. Inside every list, there are a number of cards that help to break down the task and make it even better to monitor the current status and progress.

6.6 Resource identification

6.6.1 Hardware

- RFID PVC card
- RFID Reader/Writer
- Desktop /Laptop
-

6.6.2 Software

- Windows7/8/10 Operating System
- Visual Studio Code
- JAVA and JavaScript
- MySQL Workbench
- Photoshop
- Brackets
- Apache Tomcat server
- Spring Boot
- RFID software

6.6.3 Other resources

- U.P.S
- Stationary
- Miscellaneous assets
- Hospital organizer
- Doctor
- Patient

6.7 Process model

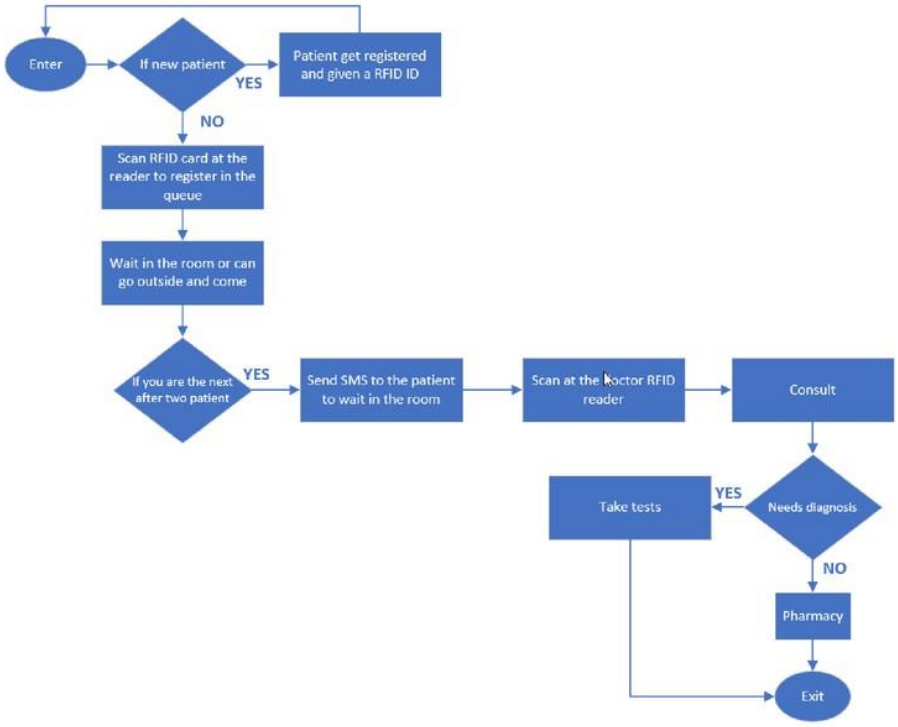


Figure 3 Process model for SCMS

The above diagram represents the main process of the system. At first, a patient enters to the Jaffna Teaching Hospital if the patient is new to the clinic and wants to register then he would be registered in SCMS and given an RFID ID card. Once the patient has registered into the system and got an RFID ID card, he is able to scan the RFID ID card on the reader in order to register in the queue. After that, the patient would need to wait for some minutes the waiting time or position can be watched in real time in the lobby TV monitor. The patient would also get an SMS in order to inform when the patient goes out or nearby. Once the patient is next after two patient an SMS would be sent to the patient in order inform that he would be the next to visit the doctor after two patients. After scanning the RFID ID on the doctor's system, the doctor can view all the information about the patient such as clinic history and patient test result and etc. Once the clinic session is over the patient would be send to take

some test and come next time or the doctor would prescribe medicine to the patient and inform to visit next time. After the patient leaves the room the doctor would click on close session button in order to allow the next patient to come.

7 Project Infrastructure

The project infrastructure of the system is shown in two diagrams below.

7.1 System Architecture diagram

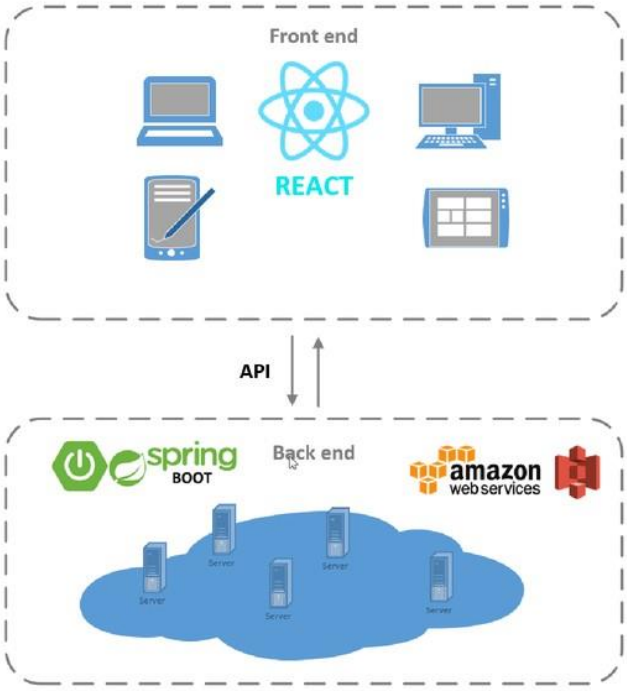


Figure 4 System Architecture diagram for SCMS

The above system architecture diagram shows two component separately as front-end and back-end. The front-end is programmed using React JavaScript library and this front end can be seen on almost all device which has web browsing facility however to make use of RFID Reader we need a laptop or computer. The second component is the back end. The back end is fully programmed using Java with Spring Boot 4 framework this framework makes any backend work fast and easy it used by many enterprise industries. We also connect the spring with AWS S3 bucket to upload patient test result. In between these two components is the API which makes all the POST, GET, PUT and DELETE request and this ties up the two components together as one application to use.

7.2 Tree tier architecture diagram

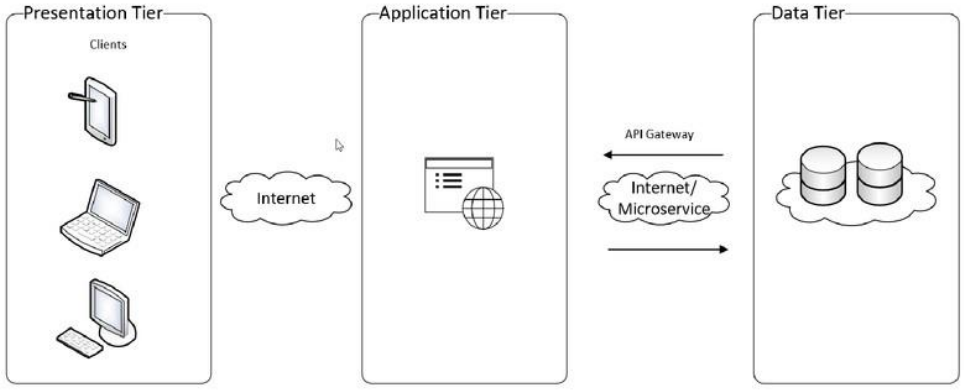


Figure 5 Three-Tier-Architecture for SCMS

Smart Clinic Management System's tree tier architecture diagram is shown above. There we can see it is separated [redacted] Application Tier and Data Tier. In the presentation tier we can see some devices such as phone, laptop and computer this is because the web application can be accessed on any devices that support browsing capability. The presentation layer is mainly involved in providing the user interface from the server. The second tier is the application tier that holds the actual React application. The application tier provides the UI to the clients. After that for every react by the client it communicates with the data tier through API request. Since the Data-tier has the main Spring boo application hosted in a cloud server, we can make data request through API calls. Each and every API called are authorized with a token to prevent hacking and misusing of API's.

8 Design

The design phase includes diagrams that support the development process of the next phase. There are many diagrams designed in order to make the system clear for development such diagrams are use case diagram, DFD diagram, Class diagram, Sequence diagram, Activity diagram, ERD and database design.

8.1 Use case diagram

The use case diagram is used represent the behavior of each actor and user cases. Service, function and actions are represented by use cases. It also comes under the UML.

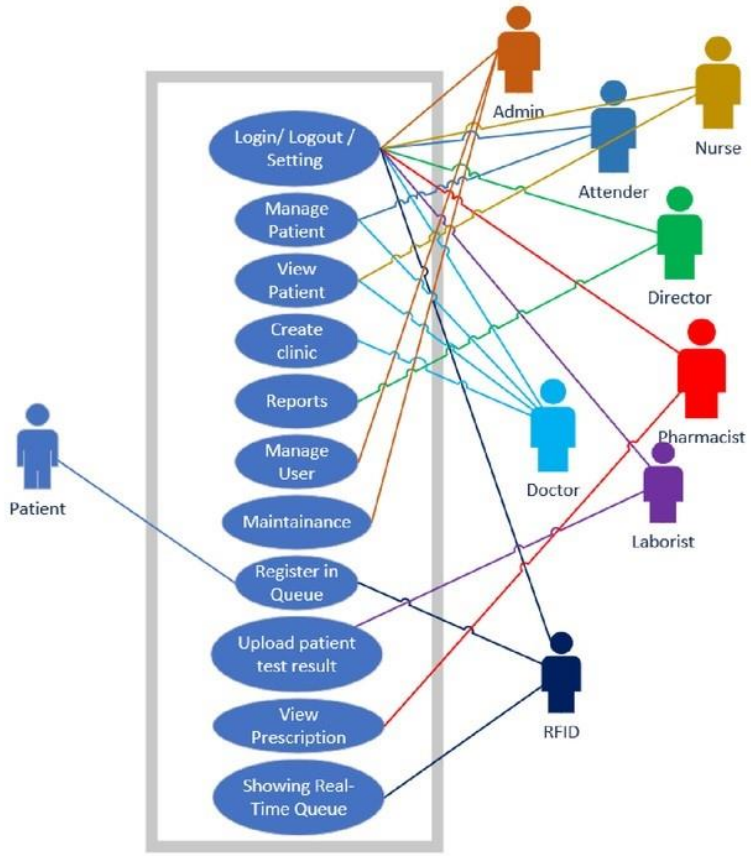


Figure 6 Use Case Diagram for SCMS

The above Use Case diagram shows the different use cases for each user type. On the left side, we have the external stakeholder which is the patient and on the right side, we have internal stakeholder which are admin, nurse, attender, director, pharmacist,

doctor, laborist, and RFID. All the internal stakeholder has one thing common which is the login, , and view setting use case. Apart from that, they have some specific use cases for each type of user types. In the diagram, we can see that the **admin** has the allowance to manage user and maintenance. The **Nurse** has the privilege to view the patient. The **attendee** has a use case to patient. The **director** has the privilege to view reports. The **pharmacist** has the allowance to view the prescription list. And the **doctor** has the use case to manage patient, view patient and create a new clinic. At last, the **RFID** has the use case to register for a queue. The external stakeholder **patient** has the use case to register the queue too because he scans his RFID ID card on the reader in order to perform the progress.

8.2

The data flow diagram also known as DFD is a graphical representation of how the through the system. To represent this, we have certain level DFD such as a content diagram, and 3. this system, I have given the context level diagram and DFD level 2 diagram which is shown below.

8.2.1 Context Diagram

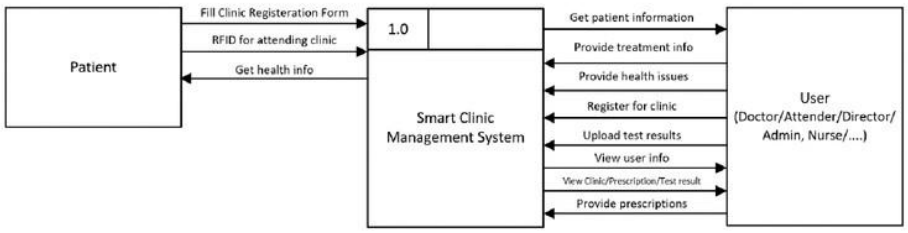


Figure 7 Context Diagram

represents how the data from the patient flows the system and from how to flows to the particular users. As users can see there 11 processes happening each of the describes the flow of data.

8.2.2 DFD Level 2

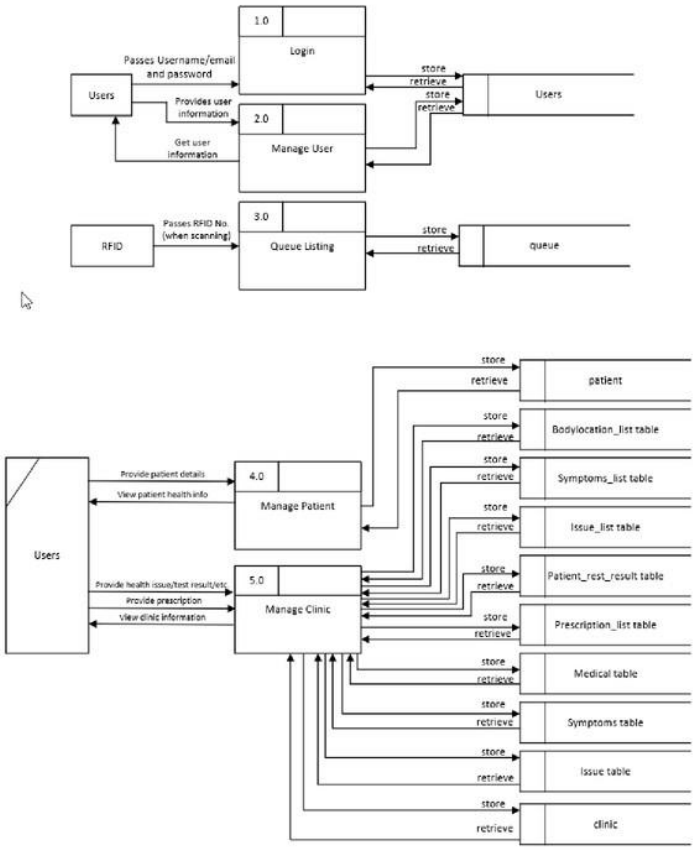


Figure 8 DFD Level 2 Diagram for SCMS

In the DFD level 2 diagram, we have even more detail version of data flowing through the system. The process number 1.0 is the login process it clearly shows how the user passes the [redacted] data to [redacted] make the login process and get appropriate data from the database user. The Users entity also provides useful information to the mange user process 2.0 there also makes the connection between the user's table. The user also retrieves user information from the process 2.0.

The manage patient process 4.0 provides the patient details to the user and views patient health info this process handles with the patient database.

The manage clinic process 5.0 is the process that is most involved with database tables. The three data process are providing health issue details such as body location, issue, symptom, test. Another data flow process is providing a prescription. The last data flow is viewing clinic information. This particular process 5.0 is involved with multiple databases such as bodylocation_list, symptom_list, issue_list, patient_test_result. Prescription_list, medical, symptoms, issue and clinic tables.

8.3 Class diagram

[redacted] that shows all [redacted] classes of a the SCMS system and all the functions also.

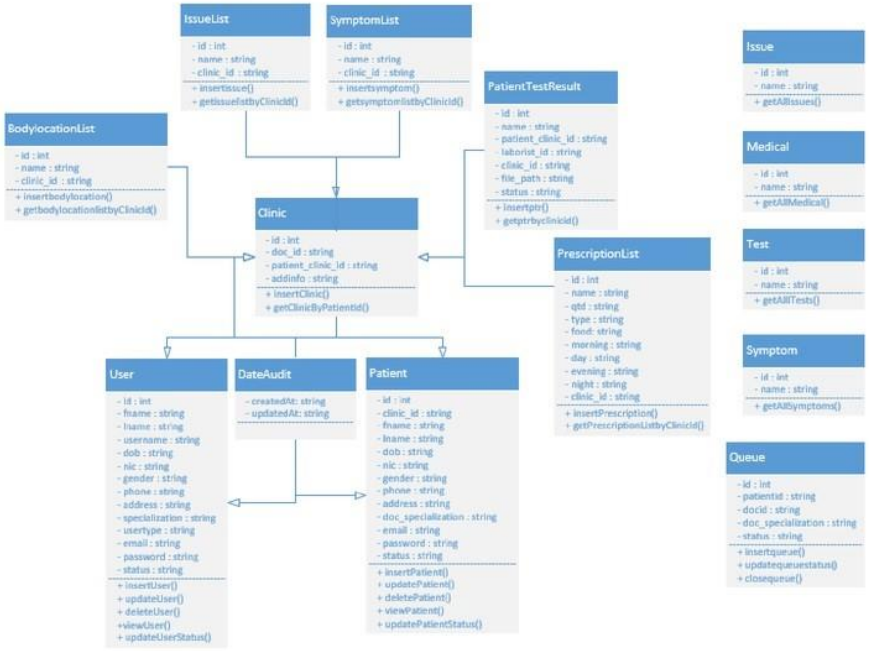


Figure 9 Class diagram for SCMS

8.4 Sequence diagram

The sequence diagram comes under UML diagram type as behavioural diagram. The sequence diagram basically shows the interaction of an operation.

Login Sequence diagram

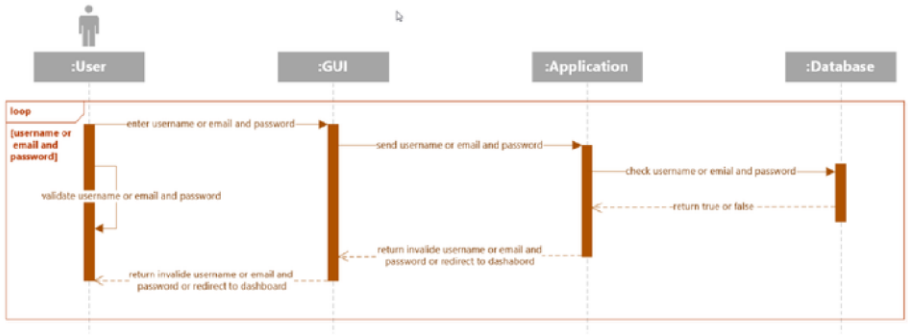


Figure 10 Sequence diagram for SCMS Login operation

The above sequence diagram is made for the login operation. The user entity enters the username or email and password. And then it will be sent to the application to check with the database whether the credentials are correct or not. The message then returns one by one from the database to the GUI.

8.5 Activity diagram

The activity diagram is another UML diagram which comes under behavior diagram. It is basically used to describe the dynamic process of an application like the main activities that happened in the Smart Clinic Management System. The below activity diagram for SCMS show the dynamic process of the patient going through a clinic using the application. At the beginning the patient scans the RFID card in case if the patient is new, he needs to register a new patient and exit. If the patient is already a registered patient he will be automatically registered for the queue. After that, he will go to consult the doctor and once the consultation is finished the patient would three activities to do one is that he needs the get prescription and the second is the get test and the third is to get both things done. After this, the queue will be closed and then the patient can exit.

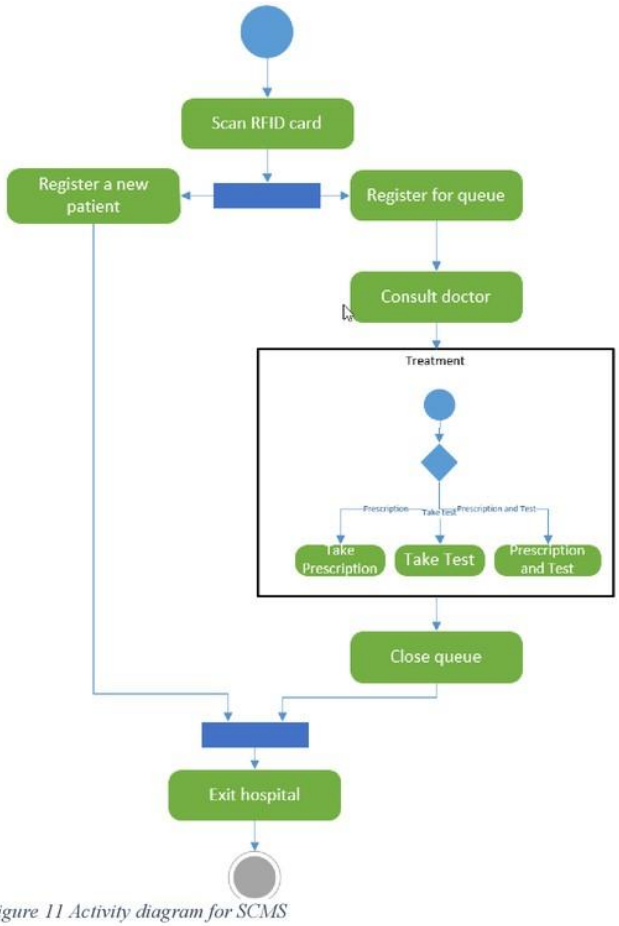


Figure 11 Activity diagram for SCMS

8.6

An shows the relationship between each entity of system. Beside relationship we all so define the attributes for each of the entities. We also indicate what type of relationship it is between entities. There are three different types of entities such one to one, one to many and many to many. Entity relationship diagram is mainly used for database mapping we have to convert from a diagram to an actual database table scheme.

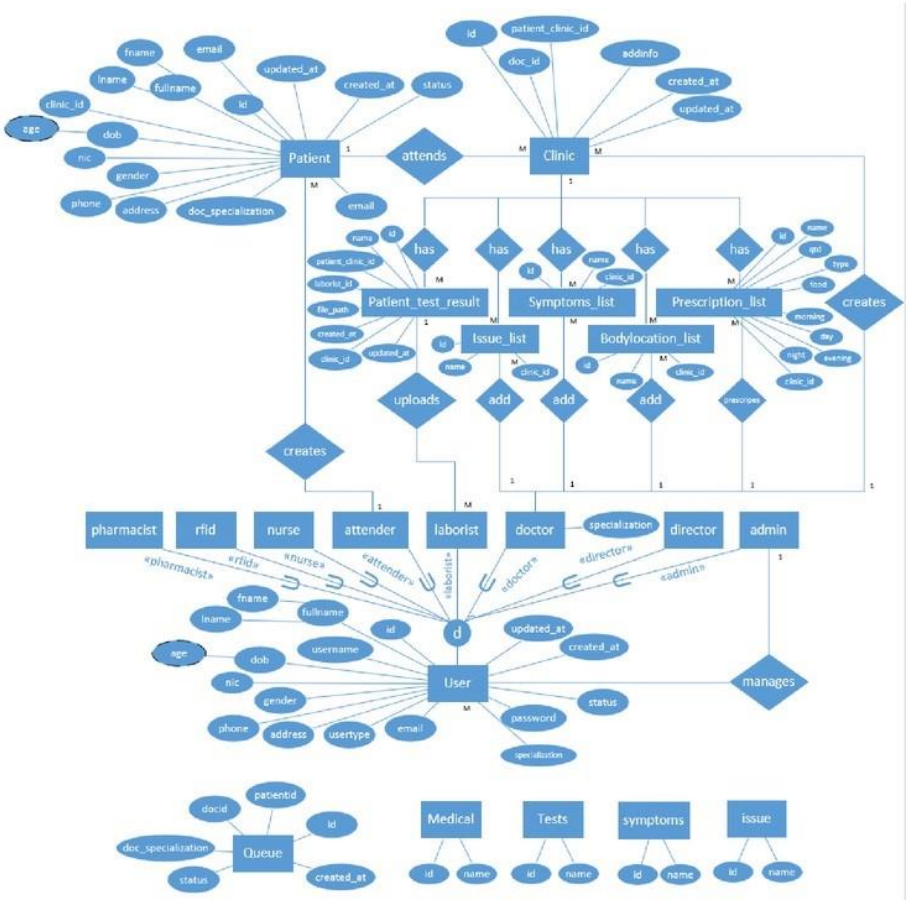


Figure 12 Entity Relationship Diagram for SCMS

In the above ERD, we can see 13 entities and many attributes. The entities are patient, clinic, patient_test_result, issue_list, symptoms_list, bodylocation_list,

prescription_list, user, queue, medical, tests, symptoms, and issue. There are 5 entities without any relationship this is because the medical, tests, symptoms, and issue is used to feed a large amount of data option for the select tag. And the queue entity is just used for the allocation the patient and doctor correctly this table will be truncated every day. Apart from the user entity has many user types which are shown as sub-entities.

8.7 Database design

The below database design is show to represent the mapped Entity relationship diagram with relationship.

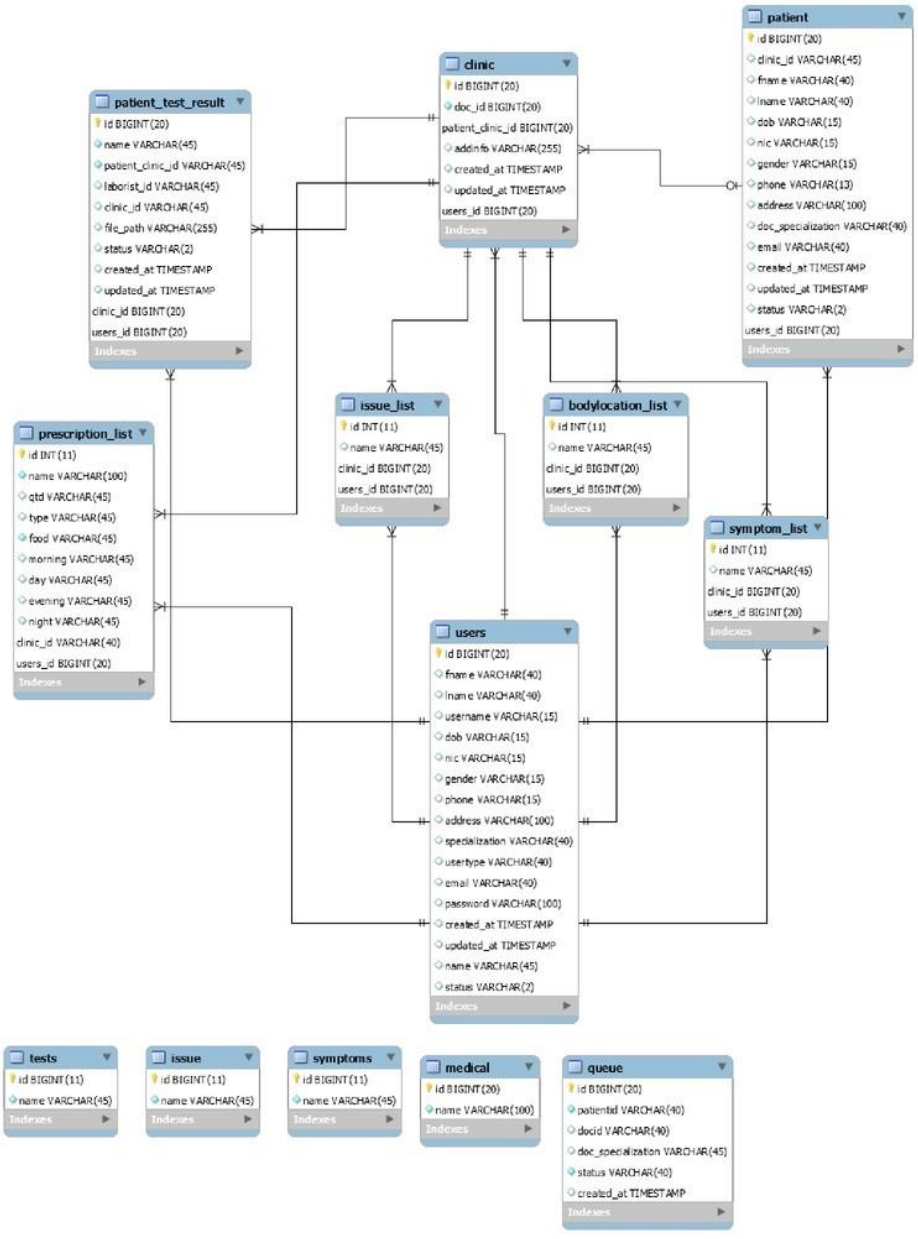


Figure 13 Database design for SCMS

9 User interface

9.1 Login UI

Figure 14 Login UI

The above UI represents a simple login with two fields for the username or email and password. There is also a login button with its label to send the information to the server to validate the credentials. If the credentials are not correct with a stay on the page showing an [redacted] is correct it will be redirected to the user welcome page shown below.

9.2 Welcome UI

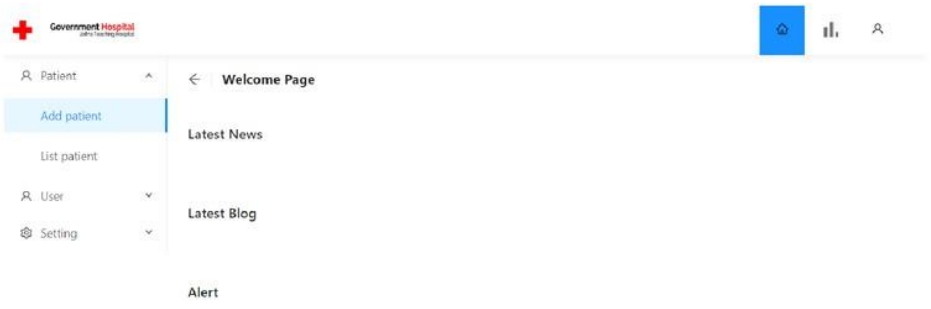


Figure 15 Welcome UI

The above UI shows the Welcome UI with a heading of Latest News which will show up all the latest news in the hospital. Then we have the latest blog for the future purpose where doctors can write blogs for the hospital to represent the foreign countries about its development and new treatments. There is also an alert section to pass information to all the user for a specific problem such as machine is out of service or doctors are needed and etc.

9.3 Add new Patient UI

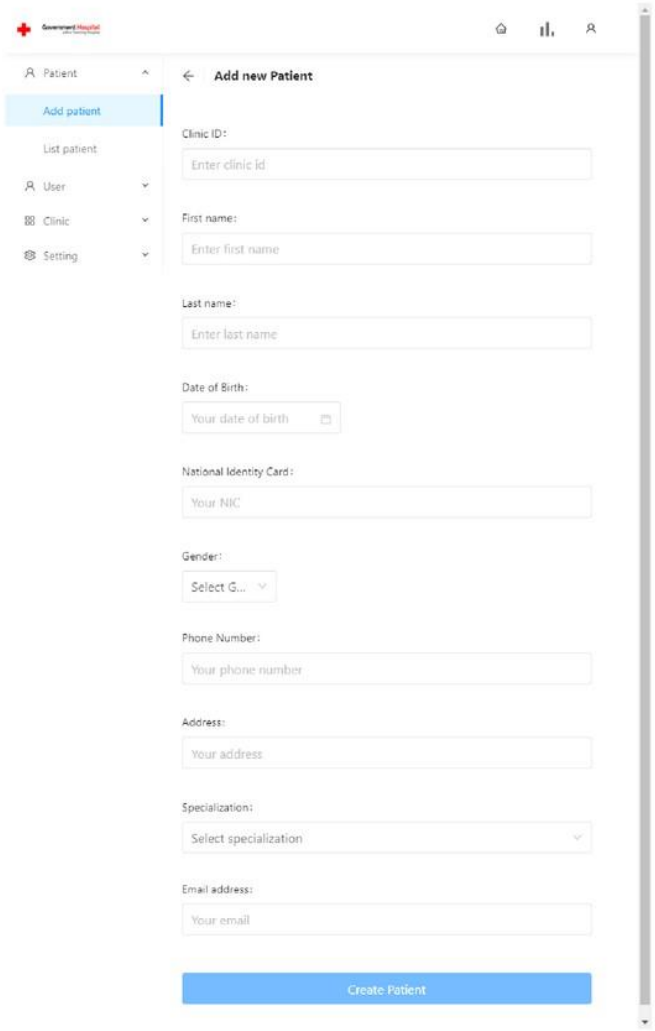


Figure 16 Add patient UI

The above user interface shows the add user UI is the layout used to add a new user for the Jaffna Teaching Hospital. As you can see in the above screenshot, we can see 10 inputs. The first input is to enter the RFID ID which is a unique ID throughout the system. The next two inputs are to enter the first name and last name of the patient. The fourth input is to enter the patient date of birth by clicking the calendar icon a pop window will pop up as a calendar for the user easily to select the date of birth. The next input is to enter the National Identification Card which is also unique throughout the system. The sixth input is the select the gender of the user whether the particular patient is male or female. The eight input is to enter the phone number of the patient this data is important for the hospital in order to make urgent information to the patient. The next input is for entering the patient address for security purposes. The next input is selection box that includes all the doctor's specialization here we have to choose for what kind of specific problem the patient would like to have clinic such as diabetes, skin care, pressure, surgery and etc. At last email address in order to inform the patient about new treatments available in the future.

9.4 List patient UI

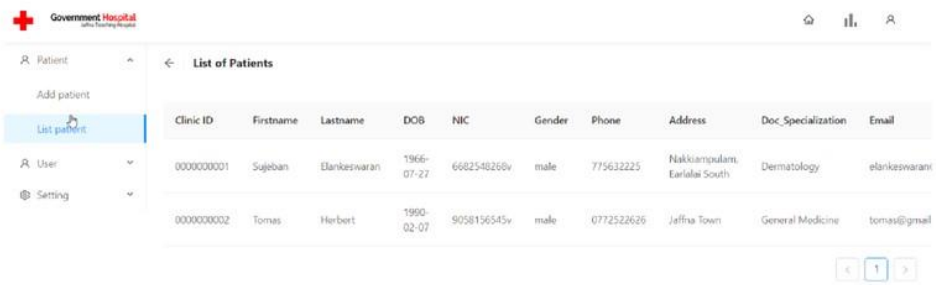


Figure 17 List patient UI

The above user interface is made for listing all the patient in the hospital. As you can see there are 10 columns named as Clinic ID, Firstname, Lastname, DOB, NIC, Gender, Phone, Address, Doc_Specialization, and Email. The fetched data has also been added with pagination.

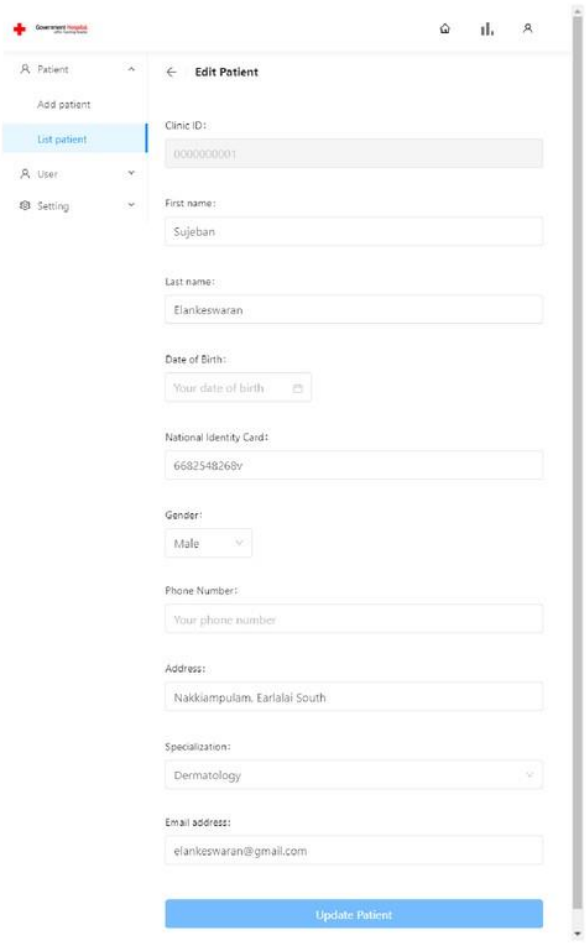


Figure 18 Edit patient UI

9.5 Edit patient

The edit patient UI has 7 input elements, 1 date element, and two select input. The first input is disabled input which displays the patient clinic ID and the next two input is for first name and last name. The date of birth input is made to select the date of birth of the patient. The fifth input is to insert the patient National Identification Card. Then we have the gender input to enter whether the patient is male or female. Next input is input the phone number of the patient. Then wean input to enter the location of the patient. The ninth element is to select input to select the need doctor

specialization. At last, we have the email address of the patient to send new treatment details in the future. It is exactly the same layout as the add patient UI but here the existing data are already filled for the user to know what he is changing.

9.6 Delete patient

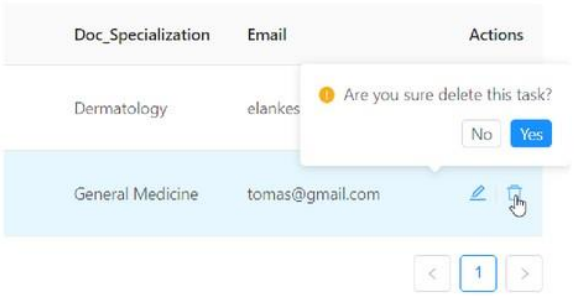


Figure 19 Delete patient UI

The web application can also delete a patient as shown in the above screenshot. When the user clicks on the trash icon a pop up will appear to make sure that you didn't accidentally click the button. After clicking the "YES" button the patient would successfully delete with popup response.

9.7 Add user UI

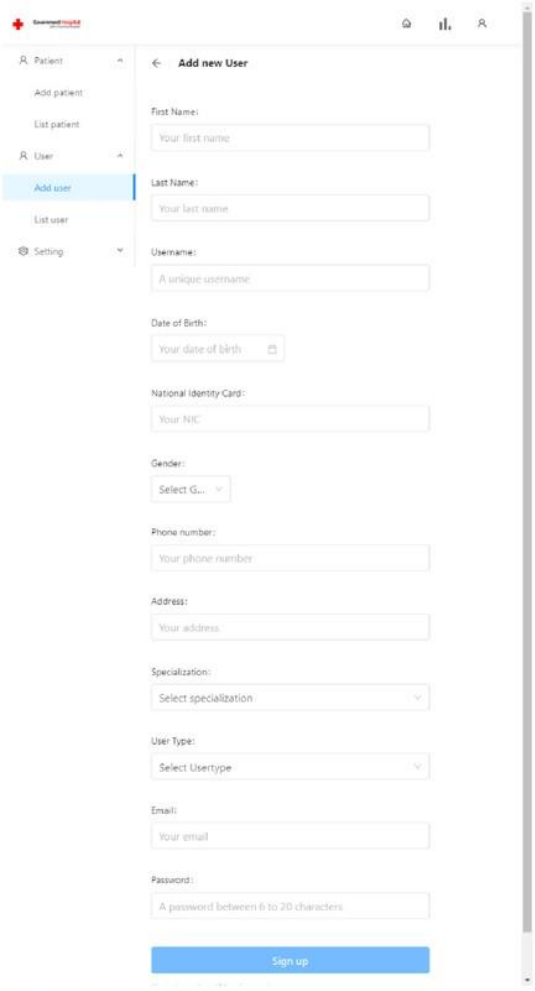


Figure 20 Add user UI

The add user UI 12 input component. The first and second component is for entering the first name and last name. Next, the username which ensures that unique username is entered throughout the SCMS. Next, the date of birth input is made to select the date of birth of the user. Next input is National Identification Card input to enter the users NIC which is also ensured to be unique. Sixth input is the gender select input which has two option, one male and female. Further, there is a phone number input its

type is set to number to make sure that the user enters numbers only. The eight input is the address of the user to security purpose. Now again we have the select input which has a lot option for doctors' specification. The next user types select input with multiple options whether the user is a doctor, nurse, director, RFID, laborist, pharmacist and etc. Next, we have email and password to enable the user to login with its credentials.

9.8 Edit user UI

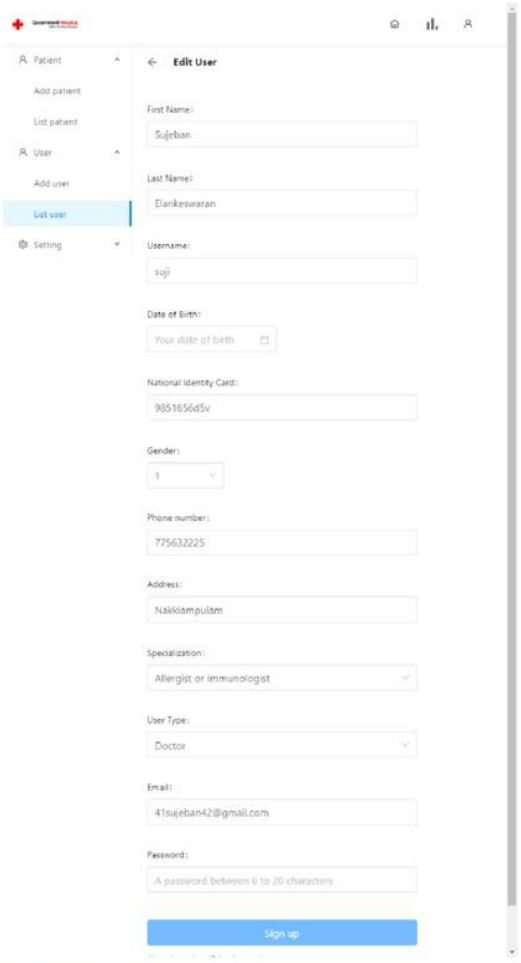


Figure 21 Edit user UI

[redacted] above layout represents the edit form for the user. This form automatically feed information based on the user id in the URL. Here it is the same component as

explained above the only difference is, we can leave the password blank if the user doesn't want to change its password.

9.9 List user UI

Figure 22 List user UI

The List user UI is layout which displays all the information of all users in the application. There are 13 columns in totals such as id, Firstname, Lastname, Username, DOB, NIC, Gender, Phone, Address, Specification, User type, Email, and Actions.

The last column actions have two buttons with an icon. The first button is to edit the user's information which redirects us to another page to edit the user. And the second button is to delete the user.

9.10 Patient Main Dashboard

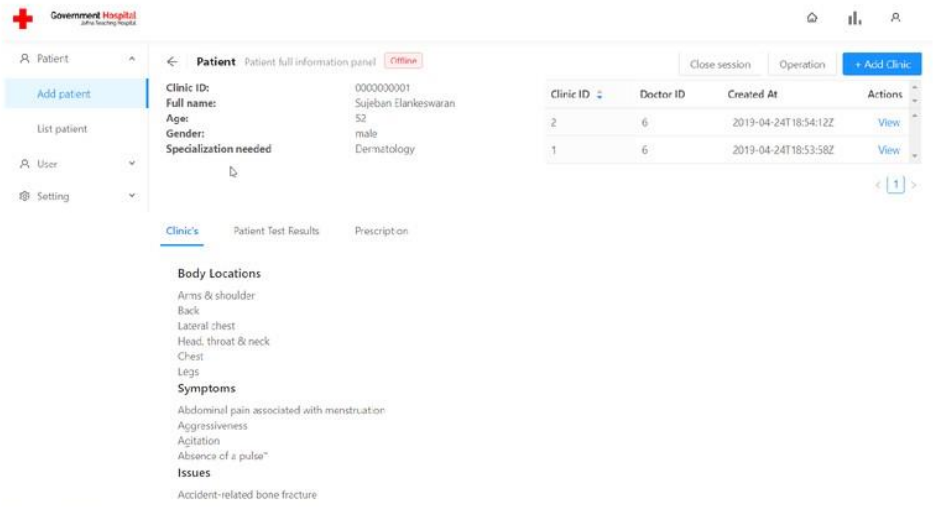


Figure 23 Patient Main Dashboard UI

The screenshot represented above shows the core UI of the system this is a page where doctors are automatically redirected when a patient scans with RFID card. Above we can the page header named as patient and besides there is red Offline box which will change after closing the session. Then we can see five essential information about the patient such as clinic id, full name, age, gender and specialization needed. Besides this information we can see the small table with all the clinic records of the patient. In the action column, we can see a view anchor tag. Once we click the view link then it will call all the information of the particular clinic such as body location, symptoms, issues, patient test result, and prescription.

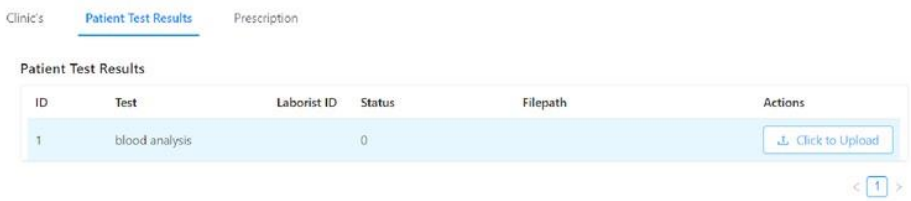


Figure 24 Patient Test Result upload UI

Once we click the patient test result tab, we can see another table containing the test and files like shown above. From the patient dashboard, we can see three buttons such as the "Close session" button, "Operation" button and "Add clinic" button. The "Close session" button is it to remove the patient from the queue and updating the patient status to normal. The "operation" button is a dummy button for future purpose. And "Add Clinic" button is to add a new clinic for the patient.

9.11 Add Clinic Step 1 UI

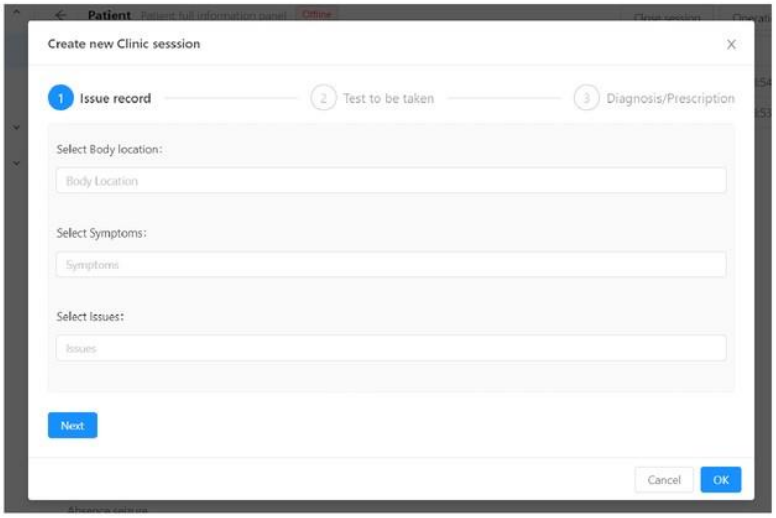


Figure 25 Add clinic Step 1 UI

The add clinic UI has in total three steps and the first step UI is shown above which has three input where the doctor can select multiple options for body location all the option are fetched from the database. Body location input is to mention in which body location the problem occurs. And the second input is the symptom input where it tells the multiple symptoms the patient has now. The last input is to select the issue of whether the patient has specific issues like how it came to a problem such as accident, abortion, ankle injury, back pain and etc. After clicking the next button, we can to the second step which is "Test to be taken" step which will be explained below.

9.12 Add Clinic Step 2

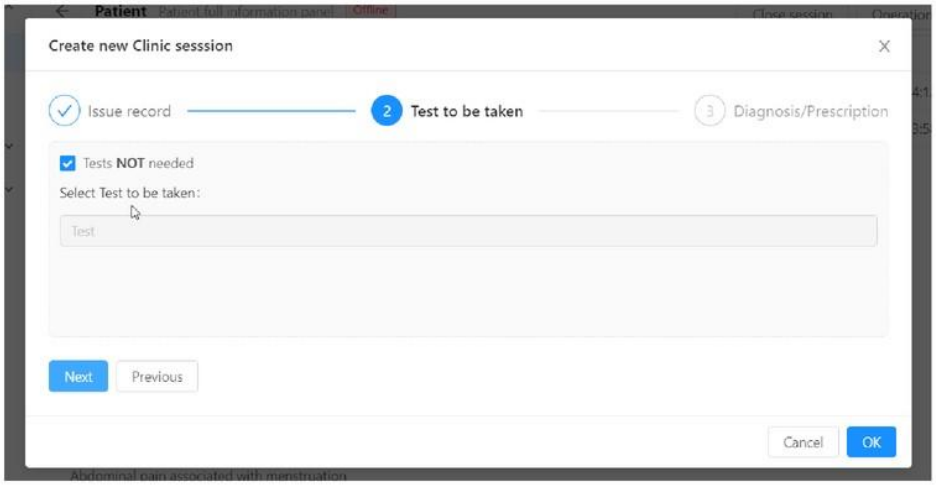


Figure 26 Add clinic Step 2 UI

The above UI represents the add Clinic modal step 2. In this, we can see a checkbox which is prechecked by the system. In case if the doctor thinks that the patient needs a certain test to be taken for further treatment then the doctor would uncheck the box which will automatically enable the below input and select the appropriate tests from the option that needs to be taken.

9.13 Add Clinic Step 3

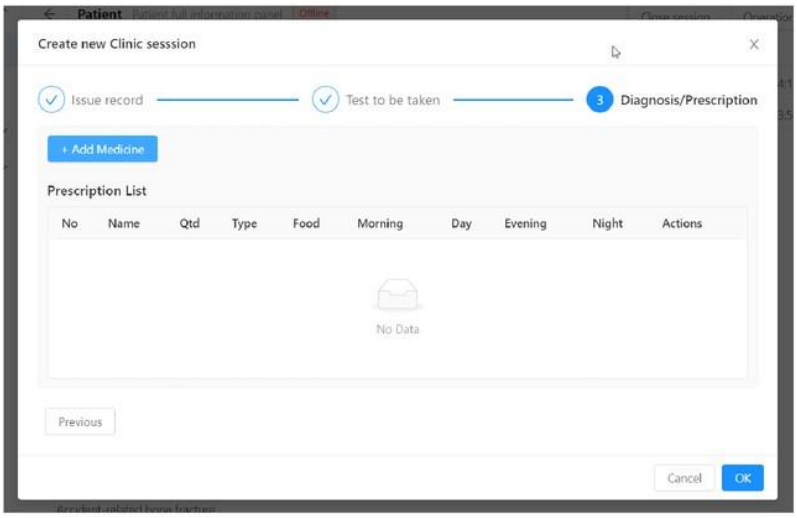


Figure 27 Add clinic Step 3 UI

Step 3 of add clinic UI is the step to prescribe medicine to the patient to do so we need to click the "+ Add Medicine" button to open another window to select a medicine to the patient which will be shown below.

9.14 Add Clinic Step 3.1 UI

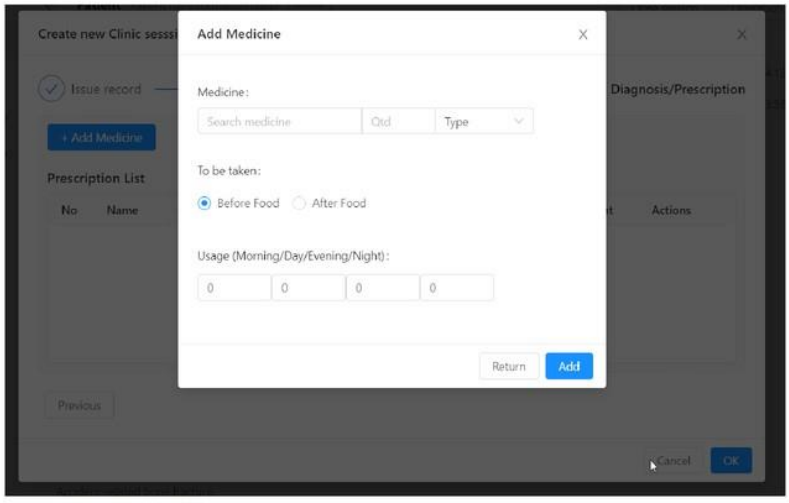


Figure 28 Add clinic Step 3.1 UI

In this modal, we can see 8 inputs. The first row is about medicine once we click the input, we can select one medicine which fetches from the database. Another input is to enter how many tablets to give. The Third input is to specify what type of quantity it is. The next row is the specify whether the medicine needs to be taken before or after food. At the last row, we input the usage of the medicals such as how many tablets to take in the morning, day, evening and night. After clicking the add button the prescription will be listed in the prescription list table.

After that, we can click the "OK" button to create the clinic with all the information given.

9.15 Queue Monitoring UI

Figure 29 Real time Queue monitoring UI

The above UI represents the Queue Monitoring UI where the main TV is placed in the lobby to show this layout and to inform the patient about the clinic process and queue process.

10 Testing

In this step, we are going to test Smart Clinic Management System with different test cases. Moreover, we have planned to do overall testing, performance testing, auditing, and speed test. Test cases are used to manually test whether all the validation from the databases returned are working. And the overall testing is manual testing where I check everything manually enter the records of the testing. The performance testing used to determine how fast the web application loads. And auditing helps to know whether it loads fast and whether is SEO friendly and whether it has used best practice and etc. Speed test used to compare the website loading speed between the low 3g network and fast 3g network.

10.1 Test Plan and Test cases

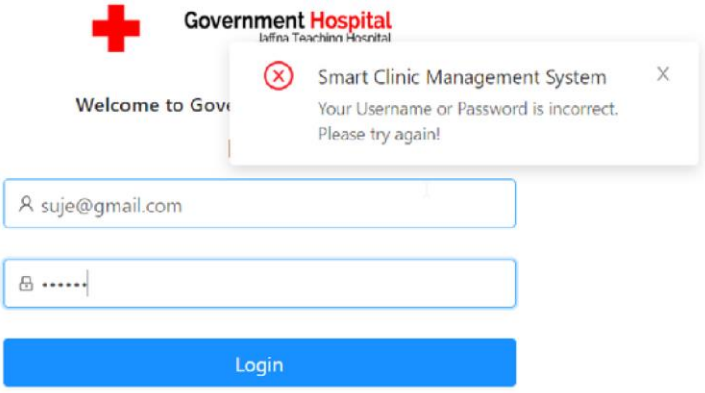
10.1.1 Test Plan

For this Smart Clinic Management System, I have performed:

- Test cases
- Overall testing
- Performance testing
- Auditing
- Speed test

10.1.2 Test cases

Test Case 1 – Testing the login system

Test case no	TC01
Test case summary	Checking the username or email and password is working or not
Authority to user	Admin, Director, Doctor, Nurse, Pharmacist, RFID, Laborist
Input	Wrong username or email and password
Test procedure	Enter wrong username or email and password and then click login button
Expected result	Entering the welcome page
Testing data	Username/Email: suje@gmail.com Password: 123456
Status	Failed
Reason	Since the email and password does not exist in the database it return an error message.
Actual result	Your Username or Password is incorrect. Please try again!
Screenshot	
Test Environment	Smart Clinic Management System – Client Side (React)

Test Case 2 – Testing the login system 2

Test case no	TC02
Test case summary	Checking the username or email and password is working or not
Authority to user	Admin, Director, Doctor, Nurse, Pharmacist, RFID, Laborist
Input	Correct username or email and password
Test procedure	Enter correct username or email and password and then click login button
Expected result	Entering the welcome page
Testing data	Username/Email: 41sakilan42@gmail.com Password: 41sakilan42
Status	Passed
Reason	Since the username or email and password exist it lets us to enter the welcome page.
Actual result	You're successfully logged in.
Screenshot	
Test Environment	Smart Clinic Management System – Client Side (React)

10.2 Overall testing

Test No.	Test description	Step to test	Expected results	Status
1	Login testing	Enter a valid username or email and password	Redirects to welcome page	PASS
2	Logout	Click logout button	Redirects to login page	PASS
3	Add a user	Insert data to add a new user	New User successfully registered!	PASS
4	Edit user	Insert data to edit an existing user	User Updated Successfully user updated!	PASS
5	View user	Click list user menu	List of users fetched	PASS
6	Delete user	Click delete user button	User Deleted Successfully user deleted!	PASS
7	Add patient	Insert data to add a new patient	New patient successfully registered!	PASS
8	Edit patient	Insert data to edit an existing patient	Patient successfully updated!	PASS
9	View patient	Click on List Patient menu	List of Patient is shown	PASS

10	Delete patient	Click delete icon to delete the patient	Patient successfully deleted	**PASS**
11	Add clinic session	Insert empty data to create new clinic record	Please check your inputs!	**FAIL**
12	Register for queue	Scan RFID to register queue	Successfully register in queue	**PASS**
13	Search patient	Insert wrong RFID ID to search patient	Could not find such patient. Please try again	**FAIL**
14	RFID scan in doctor console	Scan RFID in doctor console to change patient status in queue	Status changed	**PASS**
15	Upload patient test result	Upload particular patient test result	Successfully file inserted	**FAIL**
16	View clinic sessions of a patient	Scan RFID of a patient	All clinic session of a patient shown	**PASS**

10.3 Performance testing

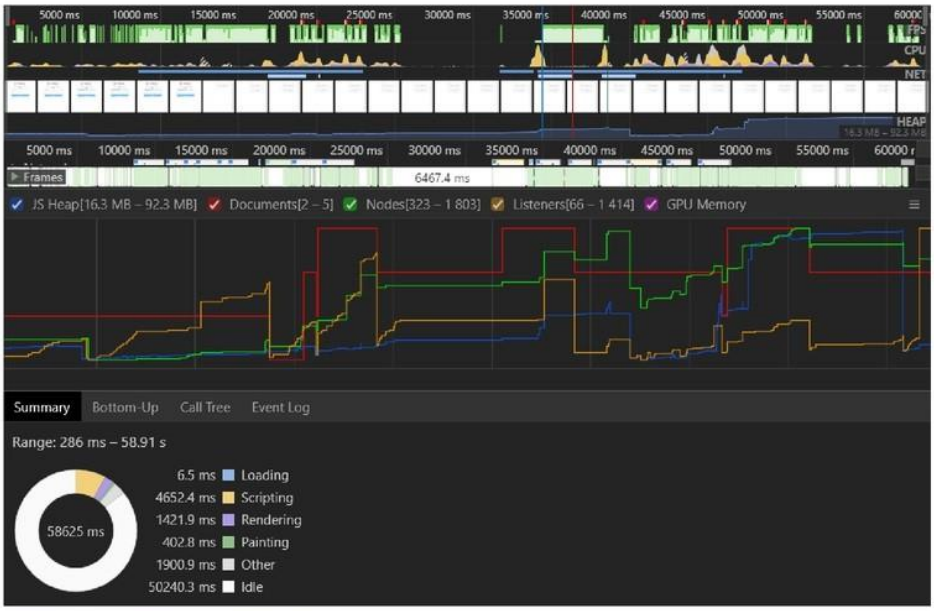


Figure 30 Performance Testing for SCMS

The above performance testing shows that totally it took 6.5ms to load the patient dashboard layout. From the 6.5 ms, the scripting takes 4652.4 ms, rendering 1421.9 ms, painting 402.8 ms and other 1900.9 ms.

10.4 Auditing – Patient Dashboard

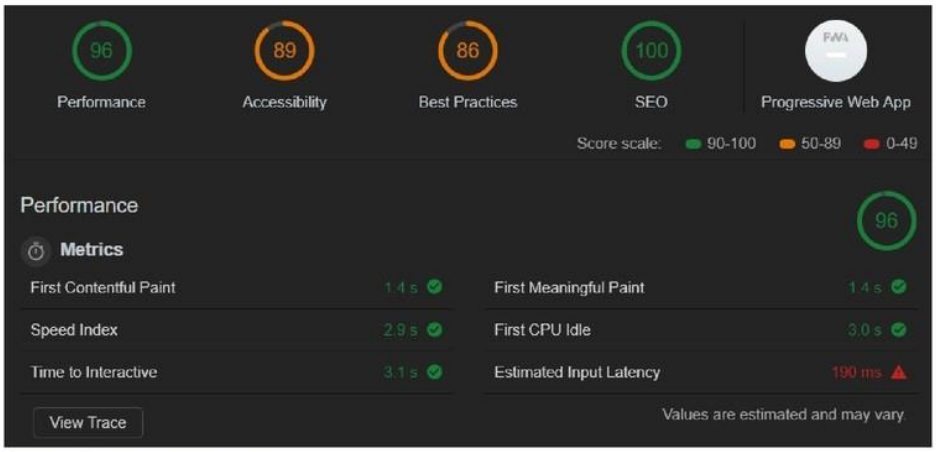


Figure 31 Auditing - Patient Dashboard

While taking an auditing test for the most loaded layout which is the patient dashboard it tells that the performance has 96 marks, accessibility 89 marks, best practice 86 marks and SEO 100 marks. Therefore, we can estimate that the Smart Clinic Management System has good performance overall.

10.5 Performance in Fast 3G VS Slow 3G

10.5.1 Fast 3G

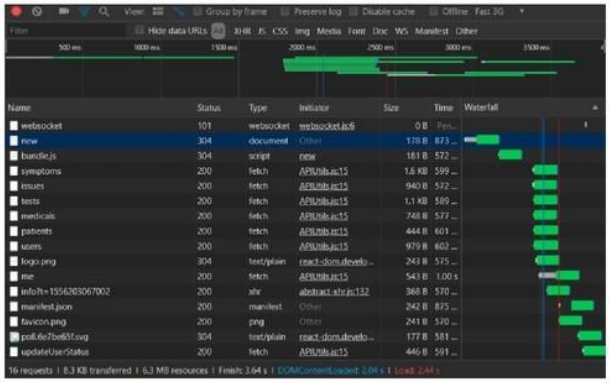


Figure 32 Fast 3G Speed Test

The above testing shows that with fast 3g the website loading speed is **2.44 s**.

10.5.2 Slow 3G

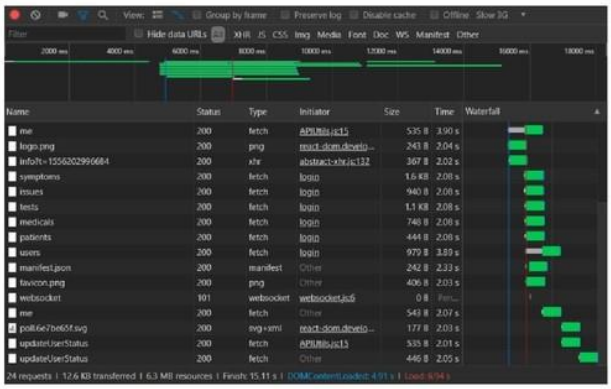


Figure 33 Slow 3G Speed Test

The above slow 3g testing shows that the application took **6.94 s** to load.

In conclusion the between the fast 3g and slow 3g network the difference was 4.5 s which is not bad. But however, for better performance of the web application we need to have at least fast 3g.

11 Implementation

In this Smart Clinic Management System, we have separated into 3 parts Front End, Back End, and Storage. The front end is made using REACT library, it is a JavaScript library specially made for building user interfaces. React makes us easy to developed complex UI in a simple manner. The main concept of React is to break apart all the UI into small components and if possible, we can break down to even more component. By breaking the component, we are able to use the component every time we need just by importing the particular component.

The back end is the second part which is made with Spring Boot 4 JavaScript framework. The JavaScript framework helps us to connect the database or cloud storage with spring application in order to create API. We create API in order to fetch data into the react application and post data into spring as well. It is basically HTTP communication between the React application and Spring application.

For third part storage, we use the traditional MySQL for text data and AWS S3 cloud storage to store files from the patient test results.

11.1 Software and Hardware Requirement for Implementation

The Software requirement for the implementation is Visual Studio, MySQL workbench, Apache Tomcat Server, Spring Boot 4, Postman and Google Chrome.

The Visual Studio Code is an IDE that is used to program the front – end SCMS using react library. In this IDE we have built in terminal to execute react app and see the progress there itself it also supports many extensions such as Beautify, ES Lint and etc. The MySQL Workbench is an application to create a database scheme and tables to integrate with Spring Boot Application. Next, the apache tomcat server needs to run the Spring Boot Application on the localhost. The Spring Boot 4 is a customized eclipse IDE just to create Spring application with ease. The Postman is used to test the create API request with server and MySQL. The Google Chrome is used to see the

user interface created using react and google chrome is also the best browser to inspect the design issue in inspecting mode.

There are is also a hardware required to implement the system for the development purpose we would need a laptop or a computer and an RFID reader only. A picture of the RFID Reader and Writer used for the development of this project is shown

Figure 34 RFID Reader/Write and RFID Card

11.2 MySQL Storage or AWS S3

MySQL is a relational database management system which is now maintained by Oracle. The MySQL run on all platform virtually. This component is widely used by many developers to create a database for an application. Most of the top website in the world is the MySQL RDBMS. The developers mostly use web development platforms like LAMP, XAMP, and WAMP. LAMP used by the Linux operating system and XAMP can be used by any operating system and WAMP is used for the Windows operating system. These web development platforms have Apache which is the web server and MySQL as a database management system and PHP for as a scripting language. In our case, we don't need a web development platform to use MySQL we can install the MySQL Workbench 8.0 CE. Along with this, we connect with Spring to create API.

MySQL integration with Spring

```
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect
spring.jpa.hibernate.ddl-auto = update
```

```
## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url= jdbc:mysql://localhost:3306/scms?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=fals
spring.datasource.username= root
spring.datasource.password= root
```

Figure 35 MySQL integration

In the above code snippet, we all the MySQL library to make a connection with JPA. And the next line is updating any changes in dB migration. In the second code snippet, we mention the URL where we can access our database. And the next two lines are there to mention the username and password to make the connection available with the MySQL.

AWS S3 Bucket integration with Spring

In AWS S3 Bucket I have created a folder called "patienttestresult" where we can upload all the uploaded patient test result file in the cloud safely.

```
aws.access.key.id =
aws.access.key.secret =
aws.region = ap-south-1
aws.s3.audio.bucket = patienttestresult
```

Figure 36 AWS Integration coding snippet

In the above piece of code, we declare the AWS access key, secret code, the region of the AWS and the bucket name. All this information are write-in <application.properties> file.

11.3 Back end (Spring Boot 4)

The back end of this system was developed using Spring Boot 4. It is framework writing in Java and it is used to create a microservice. This framework has developed the team called Pivotal team. Developers use spring boot to create an Application Program Interface (API) gateway. An API allows as to make all HTTP request such as POST, GET, DELETE, PUT and etc. using custom URL. For example, if we would type this URL in our browser "http:localhost:5000/api/users" we could GET all the users from the database if we have written the appropriate coding in Spring Boot. Spring Boot application can also be deployed in Amazon Web Service to make it work on cloud as well.

Below we could see a sample code of mapping an API. As you can see in the image there is an annotation "@RestController" this is used to tell the Spring that we are going to create an API. The meaning for REST is Representational State Transfer which is basically a method to allow client and server to communicate. Then we see another annotation "@RequestMapping("/api/queues")" this annotation tells to the spring the if we type URL ending with "/api/queues", it would automatically know that we call the QueueController. Inside the QueueController now if by seeing the below screenshot we can see another annotation called "@GetMapping" it tells that if we use GET method with the above API link it would execute the below function which is a function to findAll queues and return it into List data type. Another important thing in spring is that we use **JPARepository** and **CRUDRepository** this makes us as a developer to skip all the basic queries to write again and again. It mainly covers all the basic queries such as save(), saveAll(), getOne(), findOne(), findById(), findAllById(), findAll(), existById(), deleteAll(), delete(), deleteById() and etc.

```
28 @RestController
29 @RequestMapping("/api/queues")
30 public class QueueController {
```

Figure 38 @RestController snippet

```
42     @GetMapping
43     public List<Queue> getQueues() {
44         return queueRepository.findAll();
45     }
```

Figure 37 @GetMapping snippet

11.4 JWT Token

JWT stands for JSON Web Token is basically a method that helps represent claims securely. JWT has industry standard RFC 7519 method. JWT main feature is to decode, verify and generate tokens. I used JWT to make every API request securely to allow only our application users if we don't protect with such as authorization then anyone can access our data and manipulate and that I don't want to happen.

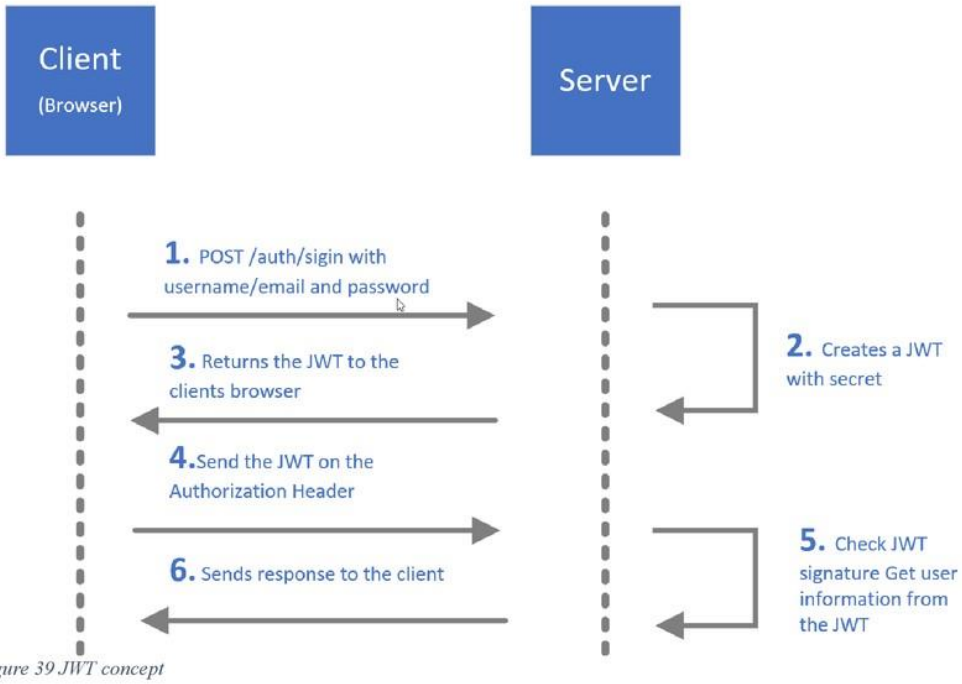


Figure 39 JWT concept

The above diagram shows the sequence of how the JWT is actually used and works. In the first step, we post the [redacted] from [redacted] browser. The second step continues in the server where it creates a JWT with a secret when a username or email and password of the user are correct. In the JWT we can also the payloads such as user roles and etc. and this payload can be decoded in the server again set role permissions. The third step we return the JWT to the client browser. In the fourth step, we send the JWT on the Authorization Header and request for user information. In the fifth step, the server validates the JWT signature to all the

user information to return to the client browser. And in the sixth step, the response is sent to the client browser.

11.4.1 Importing Json Web token

```
47          <!-- For Working with Json Web Tokens (JWT) -->
48          <dependency>
49              <groupId>io.jsonwebtoken</groupId>
50              <artifactId>jjwt</artifactId>
51              <version>${jjwt.version}</version>
52          </dependency>
```

Figure 40 Import JWT code

11.4.2 Generating Json Web token

The above coding shows how we import json web token into the pom.xml.

```
23      public String generateToken(Authentication authentication) {
24
25          UserPrincipal userPrincipal = (UserPrincipal) authentication.getPrincipal();
26
27          Date now = new Date();
28          Date expiryDate = new Date(now.getTime() + jwtExpirationInMs);
29
30          return Jwts.builder()
31                  .setSubject(Long.toString(userPrincipal.getId()))
32                  .setIssuedAt(new Date())
33                  .setExpiration(expiryDate)
34                  .signWith(SignatureAlgorithm.HS512, jwtSecret)
35                  .compact();
36      }
37
```

Figure 41 Generating JWT code

In our Spring application, we have to create a function to generateToken() after a successful login. As shown in the coding we can see that I have set users id as subject and in the issueAt, we declare the current date of token generation. In the setExpiration() we declare the how long the token is valid. Finally, we sign with SignatureAlgorithm.HS512 and jwtSecret.

11.5 Login function

The coding shown above is POST API for sign in this where the username or email and password are authenticated. We authenticate the username and password using the authentication manager since we use an add-on in spring which is the Spring Security, I am able to use that feature. It is basically a container that helps to authenticate username and password.

```
51    @Autowired
52    JwtTokenProvider tokenProvider;
53
54    @PostMapping("/signin")
55    public ResponseEntity<?> authenticateUser(@Valid @RequestBody LoginRequest loginRequest) {
56
57        Authentication authentication = authenticationManager.authenticate(
58                new UsernamePasswordAuthenticationToken(
59                        loginRequest.getUsernameOrEmail(),
60                        loginRequest.getPassword()
61                )
62        );
63
64        SecurityContextHolder.getContext().setAuthentication(authentication);
65
66        String jwt = tokenProvider.generateToken(authentication);
67        return ResponseEntity.ok(new JwtAuthenticationResponse(jwt));
68    }
```

Figure 42 Login function code

To make the users password unreadable to anyone we used the passwordEncoder.encode() function to specifically encrypt the user's password. The coding for this function is shown below.

```
93
94          user.setPassword(passwordEncoder.encode(user.getPassword()));
95
```

Figure 43 Password encryption code snippet

```
@Entity
@Table(name = "users", uniqueConstraints = {
        @UniqueConstraint(columnNames = {
            "username"
        }),
        @UniqueConstraint(columnNames = {
            "email"
        })
})
public class User extends DateAudit {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotBlank
    @Size(max = 40)
    private String fname;
```

Figure 44 Code Snippet from Model User

The above coding is from model "User" in this coding we can see many annotations. The first annotation is the "@Entity" which is used to declare the User as an entity throughout the application. And the next annotation is the "@Table" which is used to declare that this entity database table name is "users". Furthermore, the "@UniqueConstraint" annotation we use to make sure by the server that the users don't insert duplicate username or email. For the user class, I extended DateAudi class that add two more columns to add the end of user table which is the create_at and updated_at. The "@Id" annotation is used to declare that the Long id variable is [redacted] User table. The "@GeneratedValue(strategy = GenerationType.IDENTITY)", helps to generate auto-increment the id. Another one is the "@NotBlank" annotations which are used to state that the fname column should not be empty. And the "@Size(max = 40)" annotation tells the Spring application not to exceeds first name beyond 40 characters.

11.6 Front end (React)

For the front – end I have used the React java-script library it one of the famous libraries used by many developers. This framework has a great life span since it is developed by Facebook Developer. There is also a great community in all platform to get help in this framework. Comparing with Angular the major difference is that Angular is a framework and React is just a library. Similar to Angular in React also we break down everything into smaller components. The biggest advantage of separating into components is that we can always import the component without rewriting the code. In React we can also use Node Package Manager (NPM) which it holds thousands of javascript libraries.

11.6.1 Libraries used in React

11.6.1.1 react-router-dom

This library is a DOM binding for React Router it consists of BrowserRouter, Route, Link, NavLink, Switch and etc. These are component used to do routing for every web page. For example, to show user dashboard when they type a URL like this www.scms.com/dashboard/user/0001.

11.6.1.2 react-barcode-reader

It is a component that helps to read barcode from the barcode devices that are working as a keyboard. This component is actually made for barcode readers but they work for the RFID reader too because they also act as a keyboard, therefore, this library works without any issues.

11.6.1.3 antd

Ant Design is a library with an enterprise level UI design language and React implementation. This library is very useful to create complex UI in a short time it really saved my time I really recommend it to use others and I would definitely use in the future.

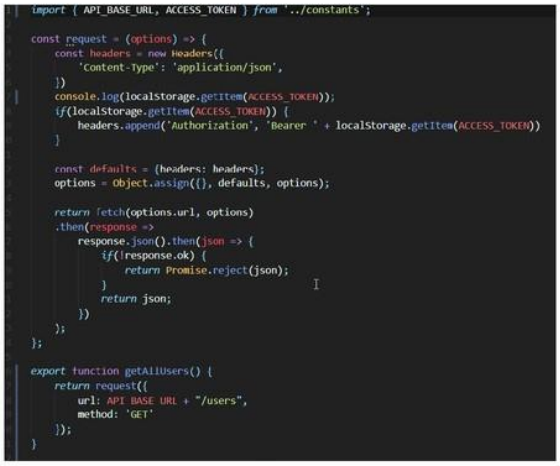
11.6.1.4 axios

Axios is a simple library that makes API calls save and easy it is a promise-based HTTP client for the node.js and browser. It gives full support on Promise API. It

automatically converts for JSON data. And it even protects against XSRF (Cross-site-request forgery).

APIUtils.js

The below screenshot shows coding part of the file APIUtil.js. In this file have all the API Request set to called throughout the application. The request function sends an API request with a parameter or URL, method, and data. Before the request is sent, I need to set the "Content-Type" to "application/json". Thereafter we get access JWT token from the local storage which is already set when we sign in. Then we set a second header called "Authorization" and pass a value as "Bearer <Access Token>". After this we can send with a request with the appropriate URL now the Spring Application would verify if the token is valid and allow us to make all request.



```
import { API_BASE_URL, ACCESS_TOKEN } from '../constants';

const request = (options) => {
    const headers = new Headers({
        'Content-Type': 'application/json',
    })
    console.log(localStorage.getItem(ACCESS_TOKEN));
    if(localStorage.getItem(ACCESS_TOKEN)) {
        headers.append('Authorization', 'Bearer ' + localStorage.getItem(ACCESS_TOKEN))
    }

    const defaults = {headers: headers};
    options = Object.assign({}, defaults, options);

    return fetch(options.url, options)
    .then(response =>
        response.json().then(json => {
            if(!response.ok) {
                return Promise.reject(json);
            }
            return json;
        })
    );
};

export function getAllUsers() {
    return request({
        url: API_BASE_URL + "/users",
        method: 'GET'
    });
}
```

Figure 45 Code snippet from APIUtils.js

pom.xml

The pom.xml is the file where we mention all the libraries, we need for this application to run. For the first time of application installation we need to call the "npm install" command after the npm would search for the pom.xml file and install all dependencies mention in the pom file.

```
{
  "name": "smart-clinic-management-system",
  "version": "0.1.0",
  "private": true,
  "proxy": "http://localhost:5000",
  "dependencies": {
    "antd": "^3.16.1",
    "aws-sdk": "^2.441.0",
    "axios": "^0.18.0",
    "bluebird": "^3.5.4",
    "express": "^4.16.4",
    "file-type": "^10.11.0",
    "fs": "0.0.1-security",
    "multiparty": "^4.2.1",
    "react": "^16.5.2",
    "react-barcode-reader": "0.0.1",
    "react-dom": "^16.5.2",
    "react-router-dom": "^4.3.1",
    "react-scripts": "1.1.5",
    "react-table": "^6.9.2",
    "sweetalert2": "^8.8.1",
    "sweetalert2-react-content": "^1.1.0"
  },
  "scripts": {
    "start": "react-app-rewired start",
    "build": "react-app-rewired build",
    "test": "react-app-rewired test --env=jsdom",
    "eject": "react-scripts eject"
  },
  "devDependencies": {
    "babel-plugin-import": "^1.6.5",
    "react-app-rewire-less": "^2.1.0",
    "react-app-rewired": "^1.4.1"
  }
}
```

Figure 46 pom.xml (library dependency)

PrivateRoute Component

```
import React from 'react';
import {
    Route,
    Redirect
  } from "react-router-dom";


const PrivateRoute = ({ component: Component, authenticated, ...rest }) => (
    <Route
      {...rest}
      render={props =>
        authenticated ? (
          <Component {...rest} {...props} />
        ) : (
          <Redirect
            to={{
              pathname: '/login',
              state: { from: props.location }
            }}
          />
        )
      }
    />
);

export default PrivateRoute
```

Figure 47 PrivateRoute React Component

The above coding snippet is one of the components in react. The component name is PrivateRoute apart from the normal route from the react-router-dom this route allows only authorized user to enter the specific route if the user is not authorized and could not be validated by the PrivateRoute component it would not allow to enter and redirect to the login path. This helps to prevent unauthorized access to the system layout. And protect also hackers from viewing the coding.

Queue algorithm

```
console.log('1' + this.state.queuelistgeneralmedicine);

    getAllActiveQueues().then((response) => {
      for (let i = 0; i < response.length; i++) {
        if (response[i].doc_specialization === 'General Medicine') {
              this.state.queuelistgeneralmedicine.splice(0, this.state.queuelistgeneralmedicine.length);
          getPatientsbyPatientsId(response[i].patientid).then((response1) => {
            let count = 1;
         if(i===0 && response1.status==='2'){
          console.log("inprogress");

          this.state.queuelistgeneralmedicine.push({
            key: i,
            no:count,
            id: response[i].id,
            patientid: response[i].patientid,
            status: response1.status,
          });
          count++;
           this.forceUpdate();
        }else if(i===0 && response1.status==='1'){
          console.log("REady");

          this.state.queuelistgeneralmedicine.push({
            key: i,
            no: count,
            id: response[i].id,
            patientid: response[i].patientid,
            status: 3,
          });
          count++;
           this.forceUpdate();
        }else{
          console.log("wait");
```

Figure 48 Queue algorithm code snippet

The above queue coding calls API function getAllActiveQueues and then loops all the response to filter the active queues by doctor's specialization called "General Medicine". After that, we empty the this.state.queuelistgeneralmedicine array. Then we call another API function called getPatientbyPatientsId() this function is called in order to get the current patient status. Thereafter we see a count variable that counts to rank the queue list. Then we see the if statement the condition tells if the patient is in the first row and the status is 2 then the patient is "InProgress" and in the second condition if the patient is in the first row and status is 1 then he is "Ready" to visit the doctor and in else statement it would say the patient to "wait".

RFID Handler

```
// RFID scanner handler
handleScan(data) {
  this.setState({
    result: data,
  })
  if(this.state.currentUser.usertype === 'rfid'){
      console.log(this.state.patientinfo);
        getPatientsbyPatientsId(data).then((response) => {
          if(response.status===0){
            const queue = {
              patientid: data,
              doc_specialization: response.doc_specialization,
            };
          //Create new Queue
          createQueue(queue);
          //Update patient status
          updatePatientStatus(1, response.clinic_id);
            message.success('Patient Registered ' + data);
          }else{
              message.error('Patient already scanned');
          }
          this.state.patientinfo = ({
```

Figure 49 RFID handle code snippet

The above code snippet shows the handleScan function which is executed right after a successful RFID scan. Once the function is executed it checks if the current user is user-type "RFID" if yes then it would get the particular patients information and compare if the patient status is equal to 0 if it is equal to 0 it would get the RFID id and create new queue in the database and update the patient status to "1". If the patient status is not 0 it would return a message that the "patient is already scanned".

Likewise if the user type is equal to "doctor" it would redirect to

"APIBaseURL/patient/dashboard/<RFID ID>"

ServerError Component

```
import React, { Component } from 'react';
import './ServerError.css';
import { Link } from 'react-router-dom';
import { Button } from 'antd';

class ServerError extends Component {
    render() {
        return (
            <div className="server-error-page">
                <h1 className="server-error-title">
                    500
                </h1>
                <div className="server-error-desc">
                    Oops! Something went wrong at our Server. Why don't you go back?
                </div>
                <Link to="/"><Button className="server-error-go-back-btn" type="primary" size="large">Go Back</Button></Link>
            </div>
        );
    }
}
                class ServerError
export default ServerError;
```

Figure 50 ServerError Component in React

The ServerError component is simple component that will be executed in case if there are server related issues. It displays the error with "Go back button".

11.7 API Design and Explanation

No.	Resource	GET (Read)	POST (Create)	PUT (Update)	DELETE (delete)
1.	/api/auth/signin		Login		
2.	/api/auth/updateUserstatus		Update user status to "1" or "2" or "0".		
3.	/api/auth/signup		Creating a new user		

4.	/api/bodylocationlists		Create new bodylocation item		
5.	/api/bodylocationlists/ bodylocationlistsbyclinicid	Get list of bodylocation based on clinic id			
6.	/api/clinics/{clinicid}	Get All clinic information and bodylocationlist, symptomlist, issuelist, patienttestresult list and prescriptionlist			
7.	/api/clinics		Create new clinics which would add new bodylocationlist, symptomlist, issuelist, patienttestresultlist and prescriptionlist		
8.	/api/clinics/clinicsbypatid/ {patient_clinic_id}	Get only clinic lists of a particular patient ID			
9.	/api/files		Uploading patient test result		

			file to AWS S3 Bucket		
10.	/api/delete				Removing uploaded patient test results
11.	/api/issues/	Get all issues list to feed select option list			
12.	/api/medicals	Get all medical list to feed select option list			
13.	/api/symptoms	Get all symptom list to feed the select option			
14.	/api/tests	Get all test list to feed select option list			
15.	/api/patients		Create new		
16.	/api/patients/{patientId}	Get patient information based on patient id	Update patient using patient id		Delete a patient with patient id
17.	/api/patients/clinic/{clinicId}	Get patient information based on RFID number			

18.	/api/patients/updatePatientStatus		Update patient status from "1", "2" or "0".		
19.	/api/queues	Get list of queues	Create new queue		
20.	/api/queues/close		Close queue by updating status to "1"		
21.	/api/queues/activequeues	Get all active queues			
22.	/api/queues/getlatestactivequeue /{special}	Get the latest active queue based on doctor specialization			
23.	/api/users	Get list of users			
24.	/api/users/{userId}		Update user based on ID		Delete user based on user ID
25.	/api/users/one/{id}	Get single user information based on user ID			
26.	/api/user/me	Get summary of a the logged in user			
27.	/api/user/checkUsernameAvailability	Get true or false. It checks whether the username is available.			
28.	/api/user/checkEmailAvailability	Get true or false. It checks			

		whether the email is available.			
29.	/api/users/{username}	Get some information of the particular user for the Profile layout.			

11.8 Deployment instruction

Deployment is uploading hosting the actual web application online in the World Wide Web.

11.8.1 Back-End deployment instruction

Step 1

Download and Install Heroku CLI

Step 2

Login using the email and password

Step 2.1

Enter "heroku login" command in CLI.

```
heroku login
```

Step 2.2

It will ask for credentials. Once logged in it will say that you are logged in displaying the email.

```
16
Email: email@example.com
Password: **************
Logged in as email@example.com
```

Step 3

Upload the spring application to the local git repository.

Step 4

In Heroku cli type "heroku create". Heroku chooses a unique name itself for the app which can altered later.

```
$ heroku create
Creating app... done, ⬢ apple-custard-79879897
https://apple-custard-79879897.herokuapp.com/ |
https://git.heroku.com/apple-custard-79879897.git
```

Step 5

Now we have to rename the project to our project

heroku apps:rename --app <OLD_NAME> <scms>

Step 6

Deploy SCMS into Heroku by simply typing the below command. It will push the project will to the Heroku master created above.

```
git push heroku master
```

Step 7

Once the long deployment is finished you can type the below command to check if the spring application works

```
heroku open
```

11.8.2 Front-End deployment instruction

Step 1

We need to create git hub account and push the react application.

(Note. gitignore - node)

Step 2

Once the application is upload, we can buy any web hosting and push the application to their server.

Step 3

Once the application is pushed to the server, we need to run the command npm install it will install all the package dependencies into the server itself.

Step 4

Change the API_BASE_URL from the local host to the URL where we have uploaded the Spring Application in Heruko cloud platform. (inside application folder src -> constant -> index.js)

As shown below in the screen shot

```
export const API_BASE_URL = 'http://localhost:5000/api';
//export const API_BASE_URL = '/api';
```

Figure 51 API_BASE_URL for spring integration

11.9 System Installation Guide

Since this a web application there no need of an installation just with a correct [redacted] it can be access.

12 Risk Management

In the field of software development, we can have many risks involved those risks we need to identify, analyze, plan, track, and control. Considering risk management in software development helps to control the risk and ensure a successful project. By planning risk management, we can inform our entire team and make sure that they are ready to act when the risks occur. Usually, the project manager is the person who monitors risk during the software development.

No	Risk Description	Probability of Occurrence	Loss Size (Days)	Risk Exposure (Days)
1.	Lack of sample data to test and validate.	45%	6	2.7
2.	Since the computer literacy rate is low, we have to face a risk of providing more effort on the staff/user guiding.	80%	4	1.6
3.	Not able to test in the real-world scenario	10%	5	0.5
4.	Not enough Testing time to validate on all browser and OS types.	45%	6	2.7

In the above Risk Management Plan, we have five columns name no, risk description, probability of occurrence, loss size and risk exposure. The no columns are used to identify the risk by there numbers. And the risk description gives a small explanation of the risk. The probability of occurrence guesses the probability of this risk could actually occur. Measuring and stating the negative impact to a project is called Loss Size which usually mentioned in hours or days. The Risk exposure is a kind of a measure that would consider the potential future loss that could result from the specific risk.

13 Project Summary

13.1 Solution Evaluation

The solution to creating a Smart Clinic Management System using RFID was a success. The web application works as it should be and all the planned solution are done even better than initially planned. Therefore, I can confidently say that the solution has been given for the mentioned problem.

13.2 System Limitation

The Smart Clinic Management System has a certain limitation. The **first limitation** is that is very unpractical to use the web application on mobile devices because of large tables and information. The **second limitation** is that the application will not work if there is no internet or electricity. Because of this reason only I insert every queue request to the database rather than in spring memory. While inserting a queue request in the database I have to set a Cron job to automatically truncate the queue table every midnight.

13.3 Future Enhancements

In the future, I would like to enhance the Smart Clinic Management System to make patient waiting for 5 minutes to meet the doctor. For this, I have to develop and mobile application to make the patient book their clinic appointment from their home. And I would develop a website to allow patient to view their clinic information from anywhere using their login credentials. Using advance RFID reader, the system would detect the patient RFID card within 20 meters and checking in the patient system whether he has booked for the clinic we can identify that the patient came for a clinic session, therefore, the system would automatically add the patient to the queue right after he enters the lobby.

Allowing prescription list private API to authorized pharmacies allows us to buy medicals from other private pharmacies when it is not available in the government hospital pharmacy. This helps to solve the problem of the badly written prescription list by a doctor.

We can also inbuilt this system to a robot to make the patient interact with the robot for asking question and information which makes to reduce more staffs. In the future

Another future work is to identify the problem with data of existing body location, issue, symptoms and test result using Artificial Intelligence. This could help to identify problems that even experienced doctor cannot identify. But this, however, must be tested for years before allowing it into the real-world application.

13.4 Lessons learned

In this final project, I have applied almost all the knowledge that was appropriate for this project to use. Rather than this I know I have learned a ton of information. In developing this project learned about a new library called React which is a really great library I definitely look forward to using it in my future project. Since the React library is basically breaking down everything in component I have also learned about the component, props, state, es6 and etc. Since I have mastered react it would be easy also to learn Angular framework too. I also learned to create enterprise-level API gateway using spring boot 4 which would definitely help in my future work. The usage of JWT token between client and server also I have learned which would be very useful in future software development. Moreover, the usage of JSON in both front end and back end thought me lot things like listing and merge multiple listing and etc. I also learned to use POSTMAN to test the API with populating header, authorization and etc. Another thing is AWS S3 Bucket for me cloud storage is very new I have already heard about in many blogs but I thought I would be very difficult and expensive to set up but however I definitely wanted to learn about AWS Cloud storage and I am taking the decision to integrate my patient test result into AWS S3 Bucket. Actually, I got it successfully working in a day which is very easy. And here after I look forward to using cloud storage without any problem. Apart from this my skill of fixing bug has increased a lot I have come across many problems in both front – end and back end too but however, I have fixed and got the application run smoothly.

13.5 Conclusion

At last the Smart Clinic Management using RFID works as planned and has gone through all the relevant testing. All the API work perfectly fine considering the security of the system with JWT token is also made sure. At the start of this project, I had great confidence that I will be able to finish the system on time perfectly as that it happened. As already said, I have learned a lot in developing this system and gave me great knowledge for my future work and strongly consider to make more project likes for my own use. I hope this system would be very useful for all the hospital on the island. And I also like to enhance the system further with face recognition and using artificial intelligence and machine learning to push the technology further.

13.6 References

Ahmad BA, K. K. F. A., 2017. An assessment of patient waiting and consultation time in a primary healthcare clinic. *Malays Fam Physician,* 12(12), pp. 14-21.

Anon., 2012. *Sri Lanka Census of Population and Housing, 2011.* [Online]
Available at:
http://www.statistics.gov.lk/PopHouSat/CPH2011/index.php?fileName=pop42&gp=Activities&tpl=3
[Accessed 8 May 2019].

Anon., 2019. *Amazon Web Service.* [Online]
Available at:
https://aws.amazon.com/free/?sc_channel=PS&sc_campaign=acquisition_IN&sc_publisher=google&sc_medium=ACQ-P%7CPS-GO%7CBrand%7CDesktop%7CSU%7CCore%7CCore%7CIN%7CEN%7CText&sc_content=Brand_Core_HV_e&sc_detail=aws&sc_category=Core&sc_segment=293607067653&sc_ma
[Accessed 24 February 2019].

Anon., 2019. *JWT.* [Online]
Available at: https://jwt.io/
[Accessed 5 April 2019].

Anon., 2019. *Spring.* [Online]
Available at: https://spring.io/
[Accessed 24 February 2019].

Anon., 2019. *Spring Boot Tutorial.* [Online]
Available at: https://www.tutorialspoint.com/spring_boot/index.htm
[Accessed 5 February 2019].

Banks, A. & Porcello, E., 2017. Learning React: Functional Web Development with React and Redux. 1st ed. s.l.:O'Reilly Media.

Crookshanks, M. E., 2017. Just Enough Requirements and SDLC: Requirements Documentation, Waterfall, and Agile. 1st ed. s.l.:CreateSpace Independent Publishing Platform.

Existek, 2019. *SDLC Models.* [Online]
Available at: https://existek.com/blog/sdlc-models/
[Accessed 5 May 2019].

Gulabani, S., 2015. *Amazon S3 Essentials.* 1st ed. s.l.:Packt Publishing.

Hinkula, J., 2018. Hands-On Full Stack Development with Spring Boot 2.0 and React: Build modern and scalable full stack applications using the Java-based Spring Framework 5.0 and React. 1st ed. s.l.:Packt Publishing - ebooks Account.

Jaffna, T. H., 2019. *Teaching Hospital Jaffna.* [Online]
Available at: thjaffna.lk
[Accessed 9 May 2019].

Jin, B., Sahni, S. & Shevat, A., 2018. *Designing Web APIs: Building APIs That Developers Love.* 1st ed. s.l.:O'Reilly Media.

Making, S., 2018. *Abstact Factory Design Pattern.* [Online]
Available at: https://sourcemaking.com/design_patterns/abstract_factory
[Accessed 4 August 2018].

Murray, A., 2016. *Information Technology Law: The Law and Society.* 3rd ed. s.l.:Oxford University Press.

Patton, R., 2005. *Software Testing (2nd Edition).* 1st ed. s.l.:Sams Publishing.

Rafat Mohebbifar, E. H. M. M. M. S. O. K. H. M. I., 2014. Outpatient Waiting Time in Health Services and Teaching Hospitals: A. *Global Journal of Health Science,* Volume 6, pp. 172 - 180.

Steve Gallivan, M. U. T. T. O. V., 2002. Booked inpatient admissions and hospital capacity:. *Information in practice,* Volume 324, pp. 280-282.

14

20147493 _Elankeswaran_CIS6002.docx

ORIGINALITY REPORT

7%	2%	1%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Wales Institute, Cardiff Student Paper	2%
2	Submitted to University of Greenwich Student Paper	<1%
3	thjaffna.lk Internet Source	<1%
4	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1%
5	Submitted to RDI Distance Learning Student Paper	<1%

6	www.castsoftware.com Internet Source	<1%
7	Submitted to South Birmingham College Student Paper	<1%
8	Olga Filipova, Rui Vilão. "Software Development From A to Z", Springer Nature America, Inc, 2018 Publication	<1%
9	Submitted to MCC Training Institute Student Paper	<1%
10	Submitted to Universiti Teknikal Malaysia Melaka Student Paper	<1%
11	Submitted to University of Newcastle Student Paper	<1%

12	Submitted to Buckinghamshire Chilterns University College Student Paper	<1%
13	Submitted to De Montfort University Student Paper	<1%
14	Submitted to National University of Ireland, Galway Student Paper	<1%
15	Submitted to iGroup Student Paper	<1%
16	Submitted to Higher Education Commission Pakistan Student Paper	<1%
17	Mohebbifar, Rafat, Edris Hasanpoor, Mohammad Mohseni, Mobin Sokhanvar, Omid Khosravizadeh, and Haleh Mousavi Isfahani. "Outpatient Waiting Time in Health Services	<1%

	and Teaching Hospitals: A Case Study in Iran", Global Journal of Health Science, 2013. Publication	
	Submitted to Swansea Metropolitan University 18 Student Paper	<1%
	Submitted to CSU, San Jose State University 19 Student Paper	<1%
20	Submitted to DeVry University Student Paper	<1%
21	Submitted to City University of Hong Kong Student Paper	<1%
22	Submitted to King's College Student Paper	<1%
23	Submitted to University of Moratuwa Student Paper	<1%
24	Submitted to Middlesex University Student Paper	<1%
25	ijirt.org Internet Source	<1%

26	www.primeradio.lk Internet Source	<1%
27	Submitted to IPMC Labone Student Paper	<1%
28	cdn.outsource2india.com Internet Source	<1%
29	www.ijser.org Internet Source	<1%
30	Submitted to Staffordshire University Student Paper	<1%
31	"Advanced Design and Manufacturing Based on STEP", Springer Nature, 2009 Publication	<1%
32	Submitted to University of Sydney Student Paper	<1%

33	Submitted to Imperial College of Science, Technology and Medicine Student Paper	<1%
	repository.maranatha.edu 34 Internet Source	<1%
35	Submitted to HELP UNIVERSITY Student Paper	<1%
36	Submitted to Gulf College Oman Student Paper	<1%
37	Submitted to Regis University Student Paper	<1%
38	e-mfp.org Internet Source	<1%
39	Submitted to University College London Student Paper	<1%

40	Submitted to College of North West London, London Student Paper	<1%
41	Submitted to Kingston University Student Paper	<1%
42	Submitted to University of Hertfordshire Student Paper	<1%
43	Submitted to Prague College Student Paper	<1%
44	Submitted to University of Keele Student Paper	<1%
45	Submitted to University of Wolverhampton Student Paper	<1%
46	d1ldz4te4covpm.cloudfront.net Internet Source	<1%

47	Submitted to Alhosn University Student Paper	<1%
48	Submitted to CSU, San Jose State University Student Paper	<1%
49	Submitted to Northcentral Student Paper	<1%
50	Submitted to Manchester Metropolitan University Student Paper	<1%
51	www.theseus.fi Internet Source	<1%
52	Submitted to University of Kurdistan Hawler Student Paper	<1%
53	Submitted to University of South Australia Student Paper	<1%

54	Submitted to University of Westminster Student Paper	<1%
55	Submitted to Sunway Education Group Student Paper	<1%
56	Submitted to Clayton College & State University Student Paper	<1%
57	Submitted to University of Central England in Birmingham Student Paper	<1%
58	Submitted to University of Mauritius Student Paper	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		