



# **Web Based Medical Center Management System For Karunaratna Clinic**

K.W.M.R. Fonseka

BIT registration number: R100878

Index number: 1008781

Name of the supervisor: Miss. T. W. Thilini Wimalasiri

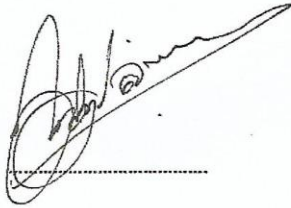
**2017/2018**



**This dissertation is submitted in partial fulfilment of the requirement of the  
Degree of Bachelor of Information Technology (external) of the  
University of Colombo School of Computing**

## Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.



K. W. M. R. FONSEKA

(Name of Candidate)

Date: 2017-09-19

Countersigned by:



Miss. T. W Thilini Tharanga

(Name of Supervisor)

Date: 2017-09-19

## **Abstract**

Clinic of Dr. Karunarathna is a leading medical center where its system runs in a half automated system with open source software for Electronic Medical Recording (EMR).

In many clinics, and medical organizations, registration and payments are very important criteria which are used for various purposes. These purposes include record keeping, medical information of patients, and security aspects of confidential medical information. Managing patient's medical recording during diagnosing diseases and symptoms of diseases has become a difficult challenge.

The expectation is to develop this kind of a system was to cater the functionalities which were not catered by the existed automated system .A web based system with the development of a Healthcare Management module and Electronic Medical Recording module using TAB is proposed to overcome the challenge of hard difficulties

An automatic Healthcare Management System and Electronic Medical Recording system using TAB would provide the needed solution. A Healthcare Management system is software developed for daily patient entries and doctor profiles in clinics and hospitals. It facilitates access to the information of a particular patient or doctor in a particular clinic. This system will also help in generating reports and evaluating the performance of a doctor and medical status of a patient. An Electronic Medical Recording system is software develop for frequent medical recordings in clinic. It facilitates to maintain the medical recording accurately and to generate reports and evaluating due disease and suitable medications.

The core system is developed based on open source technologies. PHP is the main programming language used to develop the system. MySQL is used as the database management system. Apache is used as the web server. The system has been used Hierarchical Model View Controller (HMVC) architecture and Object Oriented techniques. The system was thoroughly tested in a multi-stage testing process which include unit testing, integrated testing, system testing and acceptance testing.

# **Acknowledgment**

I take this opportunity to express my gratitude to everyone who helped me and guided me to complete the project successfully. In first place I would like to thank the staff of UCSC including the Director, BIT Coordinator and the academic staff for creating such a great programme.

Secondly, I would like to express the deepest appreciation to my Supervisor, Miss Thilini Wimalasiri, who has the attitude and the substance of a genius: she continually and convincingly conveyed a spirit of adventure in regard to develop, and an excitement in regard to teaching. Without her guidance and persistent help this dissertation would not have been possible.

I would like to thank my client, Dr Karunarathne for giving me such an opportunity by not considering I am new to the industry. I would also like to convey thanks to Dr. Karunarathne and staff members for providing the financial support and user support.

Special thanks also to my graduate friends Anuradha Ahangama and Anoj Saranga for sharing the knowledge and invaluable assistance, not forgetting their best friends who always there for my inconvenience time.

My parents and my brother and sister being very much supportive to finish this project successfully so, I would like to thank them also.

Special thanks for my other friends also help in many ways to overcome my hardships.

# Table of Contents

<b>Declaration .....</b>	<b>i</b>
<b>Abstract.....</b>	<b>ii</b>
<b>Acknowledgment.....</b>	<b>iii</b>
<b>Table of Contents .....</b>	<b>iv</b>
<b>List of Figures.....</b>	<b>vi</b>
<b>List of Table.....</b>	<b>vii</b>
<b>List of Acronyms .....</b>	<b>viii</b>
<b>Chapter 1 – Introduction .....</b>	<b>1</b>
1.1 Motivation for Project .....	1
1.2. Objectives of the Project.....	2
1.3. Scope.....	2
1.4. Structure of the Dissertation .....	3
1.4.1. Chapter 2 –Analysis .....	3
1.4.2. Chapter 3 – Design .....	4
1.4.3. Chapter 4 – Implementation .....	4
1.4.4. Chapter 5 – Evaluation .....	4
1.4.5. Chapter 6 – Conclusion.....	4
<b>Chapter 2 – Analysis.....</b>	<b>5</b>
2.1. Requirement Gathering .....	5
2.2. Analysing the current manual system .....	5
2.3. Existing Similar Solutions .....	6
2.3.1. Open EMR [WWW2] .....	6
2.4. Functional Areas of the System .....	7
2.4.1. View clinical schedules .....	7
2.4.2. Online view/Manage user profiles .....	8
2.4.3. User Registration .....	8
2.4.4. Send Emails .....	8
2.4.5. Record Medical data .....	9
2.4.6. Record Patient Medical data .....	9
2.4.7. View Medical Records.....	9
2.4.8. View clinic history .....	9
2.4.9. Analyze/Produce/Retrieve patient’s medical data .....	9
2.4.10. Manage Faculty.....	9
2.4.11.Check Doctor online. ....	10

2.4.12. Patient checkout .....	10
2.4.13. Take patients feedback.....	10
2.5. Non-Functional Requirements .....	10
<b>Chapter 3 - Design .....</b>	<b>11</b>
3.1 Alternative Solutions .....	11
3.2. Justification of the Selected System .....	11
3.3. Identifying classes.....	12
3.4. Who and How? .....	13
3.5. Relevant diagrams.....	14
3.5.1 Recording Patient Data System .....	16
3.6 Activity Diagram .....	17
3.7. Class Diagram of the System.....	18
3.8. Database Design .....	18
3.9. User interface .....	19
<b>Chapter 4 – Implementation .....</b>	<b>22</b>
4.1. Hardware and Software Requirements .....	22
4.2. Module Structure of the System .....	24
4.3 Major Code Segments.....	25
4.4 Reused Module structures.....	29
<b>Chapter 5 - Evaluation .....</b>	<b>30</b>
5.1 Introduction.....	30
5.2 The Testing Spectrum .....	30
5.2.1 Unit Testing .....	30
5.2.2 Integration Testing .....	30
5.2.3 Black Box Testing .....	31
5.2.4 White Box Testing .....	31
5.2.5 System Testing.....	31
5.2.6 Acceptance Testing.....	31
5.3 Main Test Cases .....	32
5.4. User evaluation .....	33
<b>Chapter 6 – Conclusion .....</b>	<b>35</b>
6.1. Introduction.....	35
6.2. Problems Encountered .....	35
6.3. Lessons Learnt .....	35
6.4. Critical assessment of the project .....	36
6.5. Future Work.....	37
<b>References .....</b>	<b>38</b>

<b>Appendix A – System documentation .....</b>	<b>39</b>
<b>Appendix B – Design Documentation .....</b>	<b>41</b>
<b>Appendix C – User Documentation.....</b>	<b>44</b>
<b>Appendix D – Management Reports.....</b>	<b>47</b>
<b>Appendix E - Test Results .....</b>	<b>49</b>
<b>Appendix F –Code Listing .....</b>	<b>51</b>
<b>AppendixG – Client Certification .....</b>	<b>64</b>
<b>Glossary .....</b>	<b>66</b>
<b>Index.....</b>	<b>67</b>

## List of Figures

Figure 2.1 –OpenEMR Screen .....	6
Figure 3.1 –Class Diagram .....	13
Figure 3.2 – Use case Diagram .....	15
Figure 3.3 – Record Patient Medical data System.....	16
Figure 3.4 – Process registration of a patient and assigning to a clinic .....	17
Figure 3.5 – Entity Relationship Diagram .....	18
Figure 3.6 – Home Page .....	19
Figure 3.7 – doctor Dash Board.....	20
Figure 3.8 – System Administrator Dashboard .....	21
Figure 4.1 – Code Structure of the System .....	24
Figure A.1 WAMP installation .....	39
Figure A.2 Directory.....	39
Figure A.3 Folder in Directory .....	40
Figure B.1 Patient Registration Activity Diagram.....	41
Figure B.2 Patient Renewal Activity Diagram .....	42
Figure B.3 Patient Medical Recorder Activity Diagram .....	43
Figure C.1 Main Interface.....	44
Figure C.2 Login Interface.....	45
Figure C.3 – doctor Dash Board .....	45
Figure C.4 – System Administrator Dashboard.....	46
Figure D.1 Progress Report .....	47

Figure D.2 Budget Report..... 48

## **List of Table**

Table 3.1 – Record Patient Medical data System ..... 16  
Table 5.1 – Test Cases for User Management ..... 33  
Table 5.2 – Summery of Questionnaire Result ..... 34  
Table E.1 – Test Cases for Feedback Model ..... 50  
Table E.2 – Test Cases for Inquires Model ..... 50



## List of Acronyms

<b>ADSL</b>	–	Asynchronous Digital Subscriber Line
<b>DB</b>	–	Data Base
<b>DBMS</b>	–	Data Base Management System
<b>GB</b>	–	Giga Byte
<b>GHz</b>	–	Giga Hertz
<b>GNU</b>	–	GNU's Not Unix
<b>GPL</b>	–	General Public License
<b>GUI</b>	–	Graphical User Interface
<b>HTML</b>	–	Hyper Text Markup Language
<b>HTTP</b>	–	Hyper Text Transfer Protocol
<b>IDE</b>	–	Integrated Development Environment
<b>IT</b>	–	Information Technology
<b>LAMP</b>	–	Linux, Apache, MySQL, PHP
<b>MAMP</b>	–	Macintosh, Apache, MySQL, PHP
<b>MB</b>	–	Mega Byte
<b>MAC</b>	–	Macintosh
<b>MS</b>	–	Microsoft
<b>My SQL</b>	–	My Structured Query Language
<b>OO</b>	–	Object Oriented
<b>OOP</b>	–	Object Oriented Programming
<b>OS</b>	–	Operating System
<b>PHP</b>	–	PHP Hypertext Preprocessor
<b>UI</b>	–	User Interface
<b>URL</b>	–	Uniform Resource Locator
<b>WAMP</b>	–	Windows Apache MySQL PHP
<b>WWW</b>	–	World Wide Web
<b>XML</b>	–	Extensible Markup Language
<b>XAMPP</b>	–	Cross platform, Apache, MySQL, PHP, Perl

# Chapter 1 – Introduction

With the rapid development of the health sector, managing processes between patients and doctors has become a major challenge. To meet satisfaction in between system users in health management, web based architecture is a most convenient methodology in present technology. With the support of open source technology, web based health management system lead for cost-effective, efficient and reliable functionality for users in health sector.

## 1.1 Motivation for Project

Dr Karunarithne Medical Centre is a well-established medical centre in Battaramulla area which conducts several consulting clinics per a day from 30 years of history. Large no. of patients attends to these clinics. Currently Dr. B. Karunarithne handles half of these tasks manually, so he faces some problems. He uses open source software to record patient's data. Other data are handled in manual way.

Problems are as follows:

- There is no way to view the status of medical reports for both administrative staff and the patients.
- Front desk tasks (such as registering a patient and booking a doctor, medical report maintenance etc.) are done by manually.
- Doctors do not have a way to track their patient's clinic history (other than checking manually).
- There isn't a method to notify clinic dates and other important details to both doctors and patients.

- Because the Management has to deal with large number of books and files, It is very time consuming and very tiring job.
- Hard to generate accurate reports using the current system.
- Doctors do not have a way to analyze details regarding types of diseases they had to medicate end of the clinic.
- Doctors do not have an easy way to enter patient medical notes.
- Doctors do not have a way to look for details regarding pharmaceutical products and their effects.
- Data redundancy.
- Data inconsistency

It was decided by the management to implement a proper information system to overcome the issues listed and enhance the productivity.

## 1.2. Objectives of the Project

- To provide accessible, platform-independent, user friendly system
- Increase the productivity of all the activities within the scope
- To provide a reliable way to get information about patient's history.
- Support the decisions taken inside the clinic by using the system.
- Apply the Knowledge that I gathered throughout the entire course
- Provide accurate medical management throughout the automation system

## 1.3. Scope

The future system will perform the tasks which make the ease of activities of main actors in the system. They are,

- Doctors
- Patients
- Front desk
- Management

- Administrator

The scope for the project will be as followed within the client requirements.

- Manage Patients Enrolments.
- Create or modify user details according to reference number when user is attending to the clinic.
- Provide facilities to update status of medical reports.
- Provide facilities to retrieve patient's historical medical reports.
- Provide facilities to analyse and make reports of patient's historical medical reports.
- Provide an interface to administrator/front desk user for update reference data and clinic schedules
- Provide an interface to doctor to enter, update data of a patient.
- Provide an interface to doctor to enter, update medical treatments of a patient.
- Manage Faculty.

## 1.4. Structure of the Dissertation

The following stages will provide the details about the future system and what efforts should be carry out during the process of developing the system.

### 1.4.1. Chapter 2 –Analysis

In here the process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning will be happened. This involves studying the business processes, gathering operational data, understand the information flow, finding out bottlenecks and evolving solutions for overcoming the weaknesses of the system so as to achieve the organizational goals. System Analysis also includes subdividing of complex process involving the entire system, identification of data store and manual processes. Appropriate UML documents will be drawn in here to precede the success.

### 1.4.2. Chapter 3 – Design

Based on the user requirements and the detailed analysis of a new system, the new system will be designed. The logical system design arrived at as a result of system analysis and will be converted into physical system design. The logical design produced during the analysis will turn into a physical design. Input, output, databases, forms, codification schemes and processing specifications will be drawn up in detail. Data structure, control process, equipment source, workload and limitation of the system, Interface, documentation, training, procedures of using the system, taking backups and staffing requirement will be decided at this stage.

Following tools and techniques used for describing the system design of the system. Flowchart, Data flow diagram (DFD), Structured English, Decision table etc.

### 1.4.3. Chapter 4 – Implementation

The system design will be implemented to make it a workable system. Which means the interfaces, data bases and engines will be developed by coding. And further implementation environment will be discussed.

### 1.4.4. Chapter 5 – Evaluation

This will include the testing process. Unit testing and system testing will lead to discuss about what errors has found and what modifications should be done to accomplish the wish system.

### 1.4.5. Chapter 6 – Conclusion

As the last chapter, in here the path will be opened to discuss the lessons got through the entire project and will be provided the closing details of the project.

## **Chapter 2 – Analysis**

“Analysis is the process of breaking a complex topic or substance into smaller parts to gain a better understanding of it” [WWW1]. The main steps are getting the domain knowledge about the system, collect functional and nonfunctional requirements.

### **2.1. Requirement Gathering**

Problem solving techniques used to decomposes the system into components pieces for the purpose of studying how well those components parts works and interact to accomplish their purpose.

Interviews, Observation, Gathering sample materials,analyse current similar projects, do a survey to take people ideas mainly were used as fact gathering techniques.

Following primary users were interviewed to gather facts.

- Doctors
- Patients
- Front desk
- Management
- Administrator

### **2.2. Analysing the current manual system**

Dr.Karunarathne Medical center use Open source software to record medical data only. They are doing all other processes manually so lot of time and other resources are wasted. Apart of it doctors are not satisfied with open source software with their own functional and non- functional requirements.

Medical center faces with large number of patients per day. Clinic starts in the morning to night. More than 50 patients are participating to this clinic daily. Keeping records of

every patient and enrolling new patients are done manually by front desk. Updating clinical schedules and communication with doctors are done in manual way. When they need to generate reports it's really difficult and take long time to generate them.

Patients attend the clinic and channeling and payments are done at the same time. Also patients do not have a way to know clinic schedules or their previous clinical records and doctor schedule other than visit to the clinic.

## 2.3. Existing Similar Solutions

Research has been taken out in similar systems to gain experience and knowledge about the system to be developed. There are several similar systems available and bellow mentioned one of them.

### 2.3.1. Open EMR [WWW2]

Open EMR is a web based medical recording system. Bellow Figure 2.1 shows a main page of the system



Figure 2.1 –OpenEMRScreen

This system provides facility to record medical data of a patient and to retrieve medical information about diseases. This system is difficult to manage by doctors as they are encountering navigation for their real needs. System not provides any information

about medical details. System also provides facility to put patient information and allow registration of a patient.

## 2.4. Functional Areas of the System

- Manage/ View clinics schedules
- View/Manage user profiles
- User Registration
- Send Email
- Record medical data
- Record patient's medical data
- View medical records
- View clinic history
- Analyze/Produce/Retrieve patient's medical data
- Manage faculty
- Check doctor online
- Patient Checkout
- Take patient feedbacks

### 2.4.1. View clinical schedules

Patient can view clinic details when they visit to the site. Time of the clinic, which doctor conduct the clinic date of the clinic, maximum number of patients allowed for the clinic can be viewed on site when patient visits to the site.

Front desk uploads the products details to the site and he may update the clinic details time to time. Details of the clinics can be changed time to time as doctor's availability.



### 2.4.2. Online view/Manage user profiles

User of the systems as follows.

- Doctors
- Patients
- Super Administrator
- Front desk
- Management

All users can visit the site online and log in to the relevant profile (Patients don't have facility to have their own profile according to security managements. Hence no log in for patients.). Super administrator handles the user profiles and monitor user profiles to find active members and inactive members.

### 2.4.3. User Registration

Users can registered to the site by providing their basic information and when the registration successful, users can be log-in to the system by providing their username and password. Registered users can view their past activities. Doctors and front desk employees are registered by administrator in the system. To channel the doctor or to participate for the clinic, Patient must be a registered patient. When a patient tries to channel a doctor or booking a place in a clinic, front desk will ask patient's name or ID. If the patient is not registered front desk will register him/her to the system.

### 2.4.4. Send Emails

When front desk assign a clinic schedule or patient check up for a doctor, automatically it will send an email to the doctor by informing new schedules. When front desk cancel the clinic or delete or update the schedule according to the circumstances, an email will be sent automatically to the doctor again to inform updating of schedules.

#### 2.4.5. Record Medical data

Doctors can record medical data for further use in future to take decisions and to share knowledge with other doctors.

#### 2.4.6. Record Patient Medical data

When doctor check the patient, doctor can enter medical data suggested medicine regarding patient's illness. Doctor can enter data to the patient's profile. This data can be used for further information in upcoming clinics and to analyze the disease.

#### 2.4.7. View Medical Records

Doctors can view medical records to support their decision. When an active patient visit the clinic, doctor can check or view relevant patient medical history to have knowledge about the condition of patient.

#### 2.4.8. View clinic history

Management can view clinic history to support decision taking. Front desk can view previous clinic schedules regarding patient inquiries.

#### 2.4.9. Analyze/Produce/Retrieve patient's medical data

Doctors can view medical history to support decision taking. After analyzing system will automatically produce reports of patients. Doctors can send clinical history of the patient to another hospital only. When doctor request a report of the patient, system will generate a report and allow doctor to send the report for a hospital. Doctors are not allowed to insert emails due to security reasons. And they can only generate their clinic history but other doctors. Doctors can generate prescription after they enter details about medicine what give for the patient. Then this prescription can be printed. It will reduce the errors occur due to hand writings.

#### 2.4.10. Manage Faculty

Management can visit the site and can search for the clinic schedules; doctors' availability and number of patients attend for the clinics per day basic or per month basic, year basic.

Reports will generate by the system to the requirements of management. To analyze the profit end of the month, increment of patient base, satisfaction of patient's, views will be linked with management profiles.

#### 2.4.11. Check Doctor online.

Patient can visit to the site and can check the availability of the doctor, specialty of the doctor, time schedules relevant to the doctor.

#### 2.4.12. Patient checkout.

Front desk can check out to find next patient is available for the session or can assist next patient immediately.

#### 2.4.13. Take patients feedback.

Patient can visit the site and give feedback with feedback option on the site. This will increase the productivity of the service in medical center and satisfaction of patient. Management can analyze the feedback and take decision regarding doctors and procedure of the medical center.

## 2.5. Non-Functional Requirements

The Following Non Functional Requirements were also discovered.

- Time consuming is one of the major drawbacks in the current manual system. Then the automated system should respond users as quickly as possible in data retrieval & calculations.
- The user interfaces should be simple to operate.
- In case of hardware or software failure, the system should function without a loss to the data.
- There should be frequent back up mechanism to preserve data in order to achieve the reliability of the system
- The formulas used by the system should not be able to change by the primary users
- The management information such as profit for a particular period, private details of a particular user, analysis reports referred by the management of the business should not be able to access by the ordinary users. [Access control]

## **Chapter 3 - Design**

### **3.1 Alternative Solutions**

- Alternative 1 – Customized a CMS and builds a system to suite the requirements.

This is good when all the requirements are supported by the CMS. But time consuming.

- Alternative 2 – Customize an open source electronic medical recording system to suite the requirements.

Some of the client's requirements cannot be satisfied by these systems. Therefore, some modules have to be developed from the beginning. And already they use an open source Electronic Medical Recording system with having unsatisfied functionality.

- Alternative 3- Develop and implement a LAMP/WAMP based solution

Since most of the development tools are free and relatively less expensive hosting charges, this is considered as one of the best development solution. Furthermore, lots of reusable communities' written plugins are available to use with the system.

### **3.2. Justification of the Selected System**

Since this program will be running on multiple computers and will be used by lot of users, a web based solution would be appropriate for this project. And the system will be used in lot of places where geographically different. Hence the suitable solution is a web based system.

Then discussing with the owner & users & observing their book keeping methods, kind of a proposed web based system was found matching with them.

Since that familiar with developing applications with PHP and MySQL, alternative 3 seems to be suitable for fulfill client's requirements and it's also a cost effective solution. Therefore considering those facts, developing a LAMP/WAMP based system is chosen.

### 3.3. Identifying classes

When designing the system using object oriented design techniques, the class diagram is used to identify the classes – which represent the parts of the system with its attributes(properties) & operations or behaviors (which the class can do) with relationship among classes.

The identified classes & some of their Attributes & Behaviors can be stated as follows.

- Login – Attributes can be stated as Login time, User name & Password. User validation, Giving System Access & Log out can be taken as behaviors.
- Front desk – Front desk will do activities such as Input Data, Delete Data
- Administrator –Behaviors are like validate data, Generate Reports & Monitor operator activities.
- Management- Behaviors are like validate data, Generate Reports & Monitor operator activities.
- Patient- Attributes can be stated as Patient’s name, Contact details & Clinic enrolled. Behaviors are Register, Check clinical schedules, log in to profile.
- Doctor- Attributes can be stated as Doctor name, Contact details. Behaviors are log in profile, view and change their profile, Generate Reports, Record medical data, Search medical data.

Figure 3.1 represents the Class Diagram of the system.

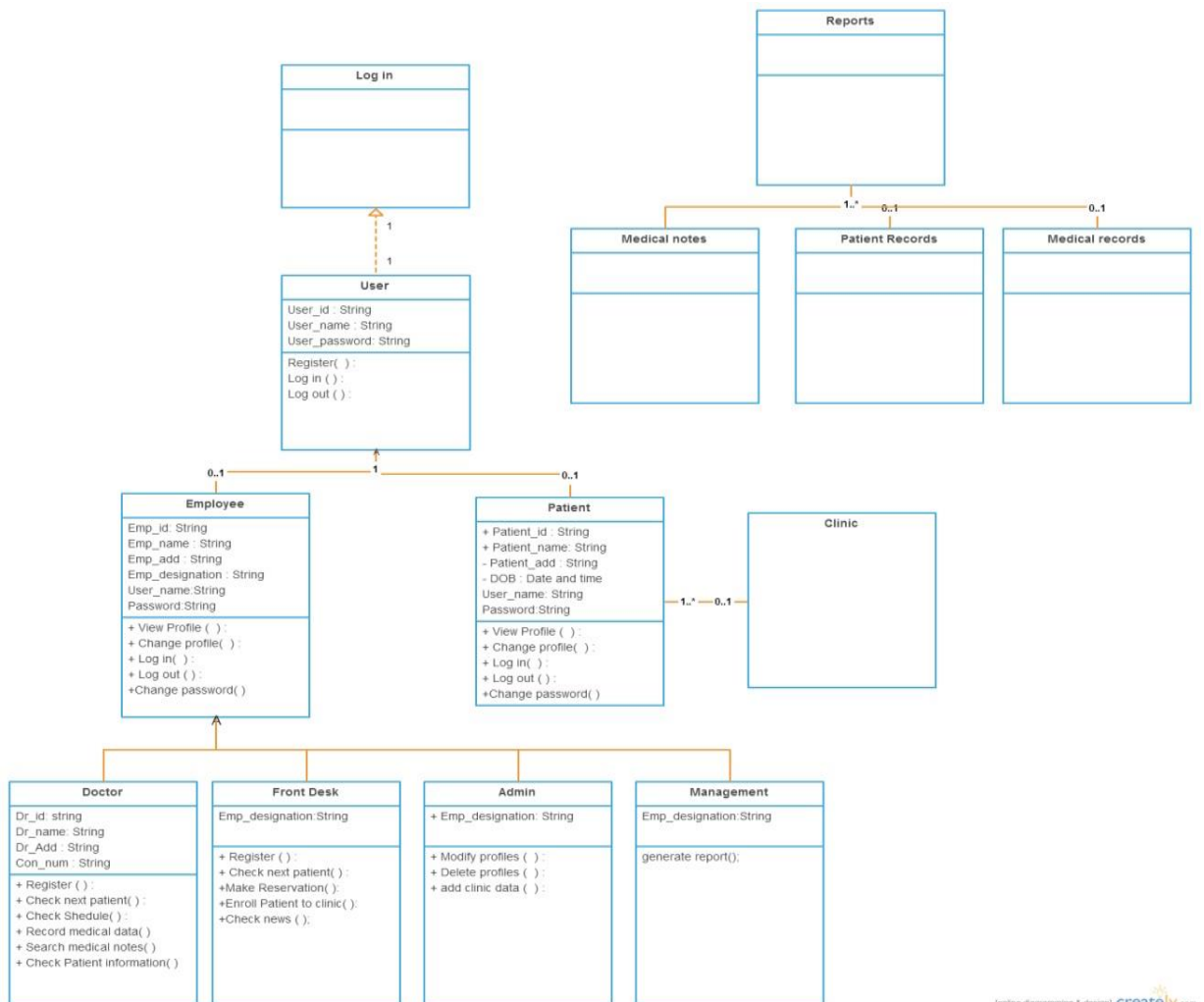


Figure 3.1 –Class Diagram

### 3.4. Who and How?

Almost all computerized systems, it is vital to understand clearly when analyzing the current process & the designing of proposed system, what kind of users interact with the system & the ways & means they interact with it. Through UML, this can be clearly understood by using a Use Case diagram which depicts the Actors as anyone or anything that needs to interact with the system to exchange information & with the Use Cases which describe system functions from the perspective of external users in a manner they understand.

### 3.5. Relevant diagrams

Figure 3.2 shows the use case diagram of the system. There are four user types in the system. They are

- Administrator
- Doctor
- Front desk.
- Patient

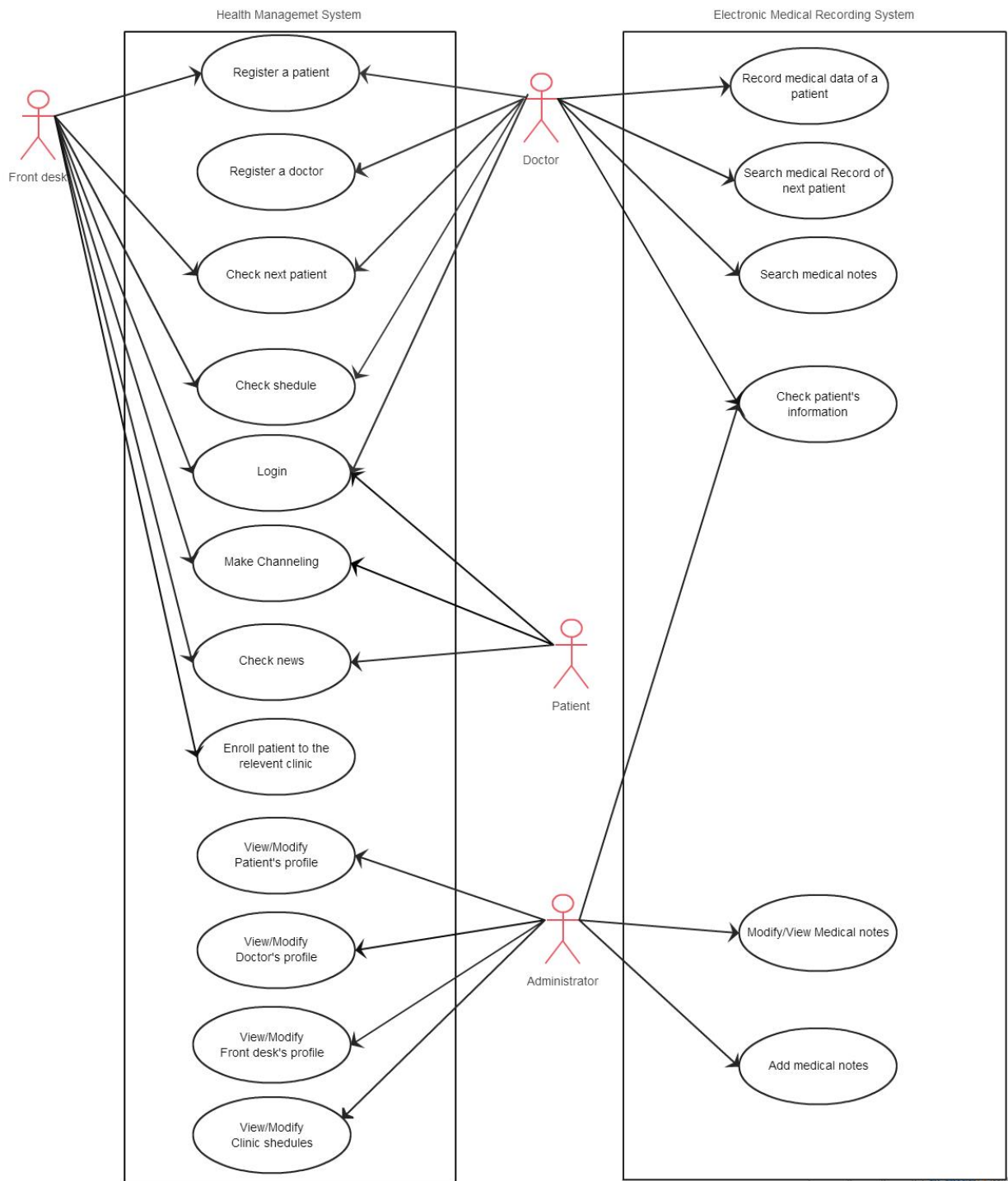


Figure 3.2 – Use case Diagram



### 3.5.1 Recording Patient Data System

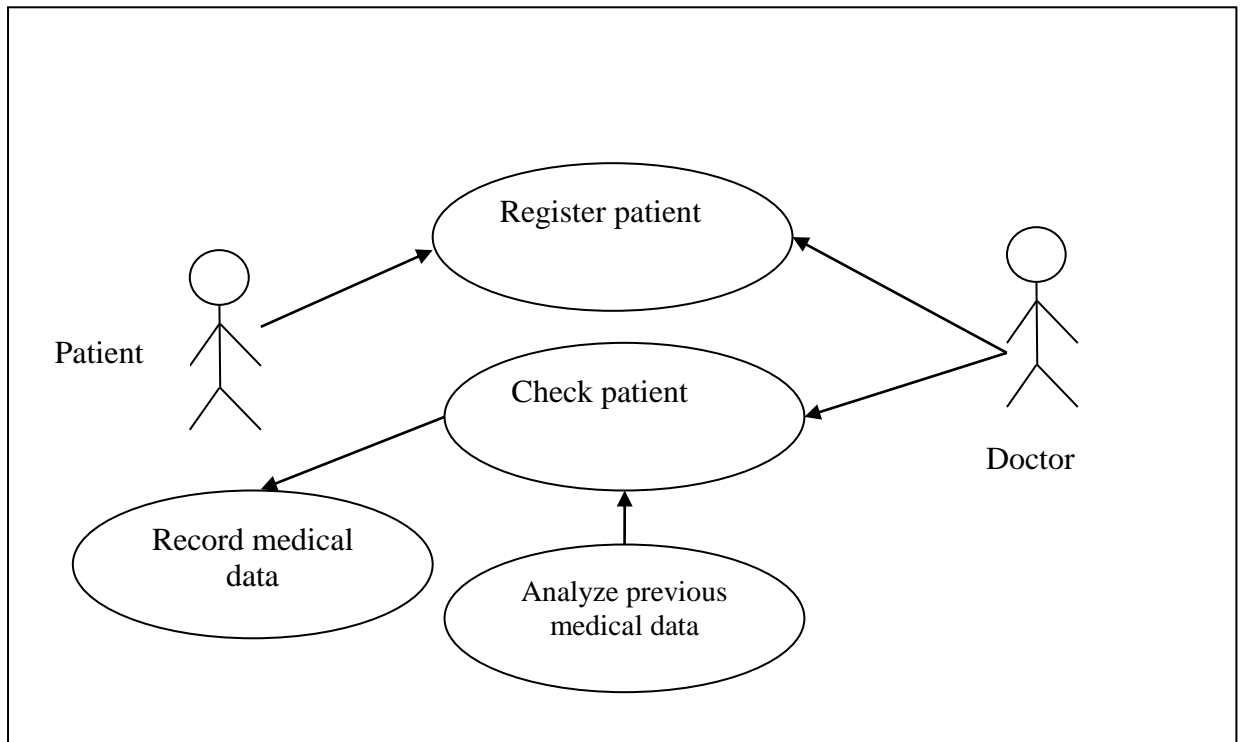


Figure 3.3 – Record Patient Medical data System

#### Use-Case Description for Record Patient Medical Data system

<b>Use case</b>	Record Medical Data
<b>Actor</b>	Doctor, Patient
<b>Overview</b>	
Record patient medical data	
<b>Preconditions</b>	
Registration of patient	
<b>Flow of event</b>	
<ol style="list-style-type: none"> <li>1. Patient can register himself or doctor can register the patient</li> <li>2. When a registered patient examine in a clinic retrieve previous data to analyse</li> <li>3. Decide the disease</li> <li>4. Enter new medical data in to patient profile</li> </ol>	

Table 3.1 – Record Patient Medical data System

### 3.6 Activity Diagram

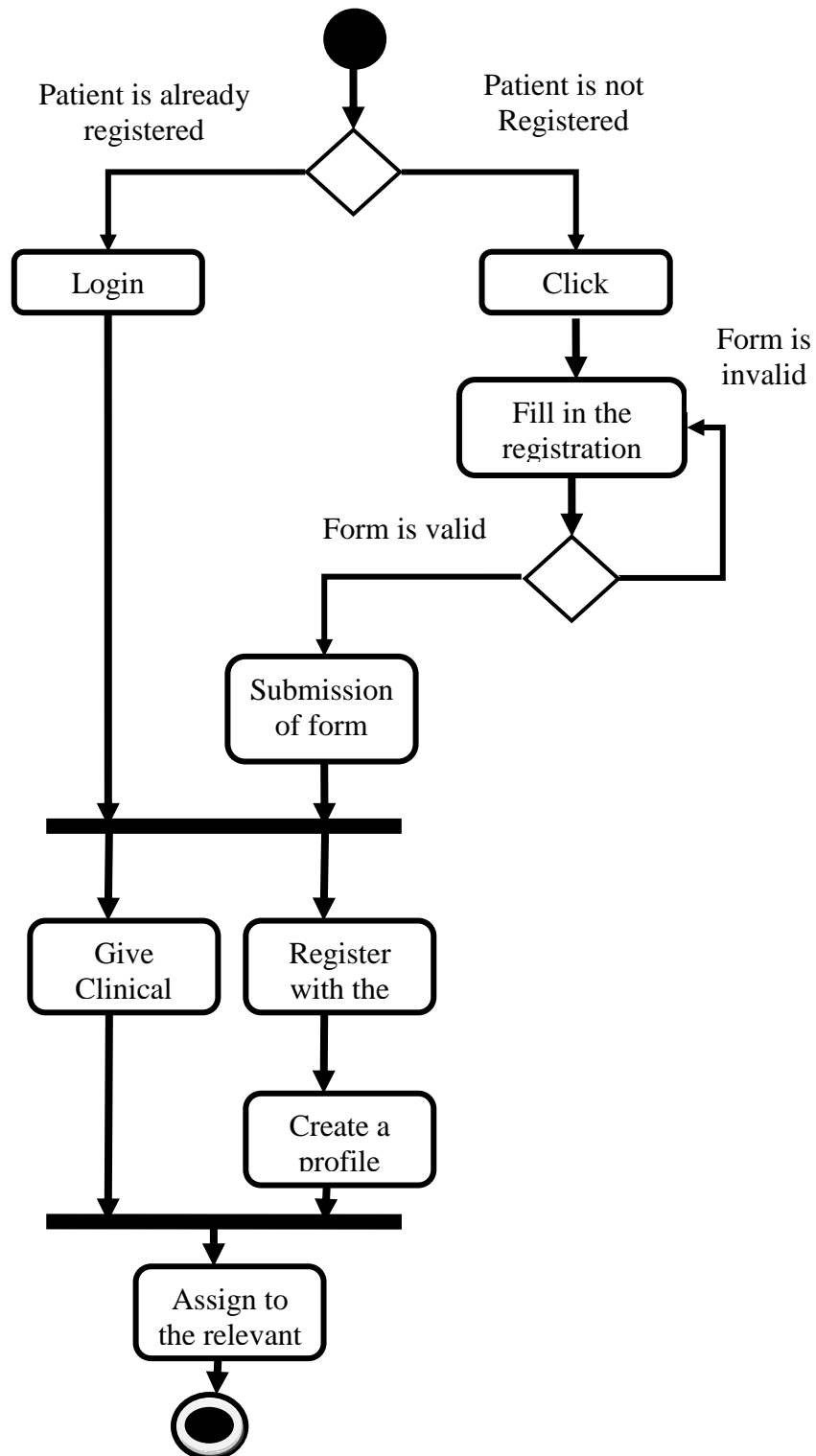


Figure 3.4 – Process registration of a patient and assigning to a clinic

### 3.7. Class Diagram of the System

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations and the relationships among the classes.

### 3.8. Database Design

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. [WWW5] Entity relationship diagram shows the graphical representation of entities and their relationships to each other. In this system there are twenty entities identified and shows the relationships between those entities. Figure 3.5 shows the entity relationship diagram of this system.

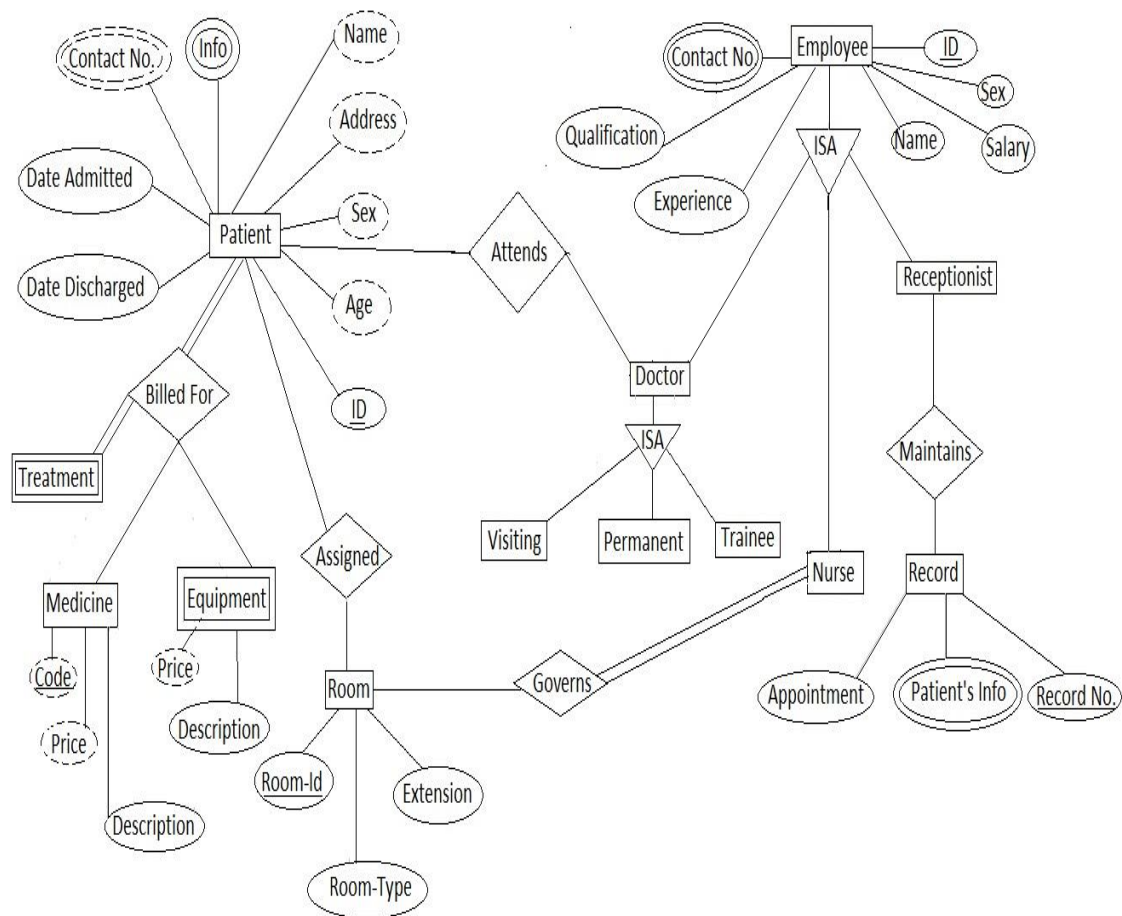
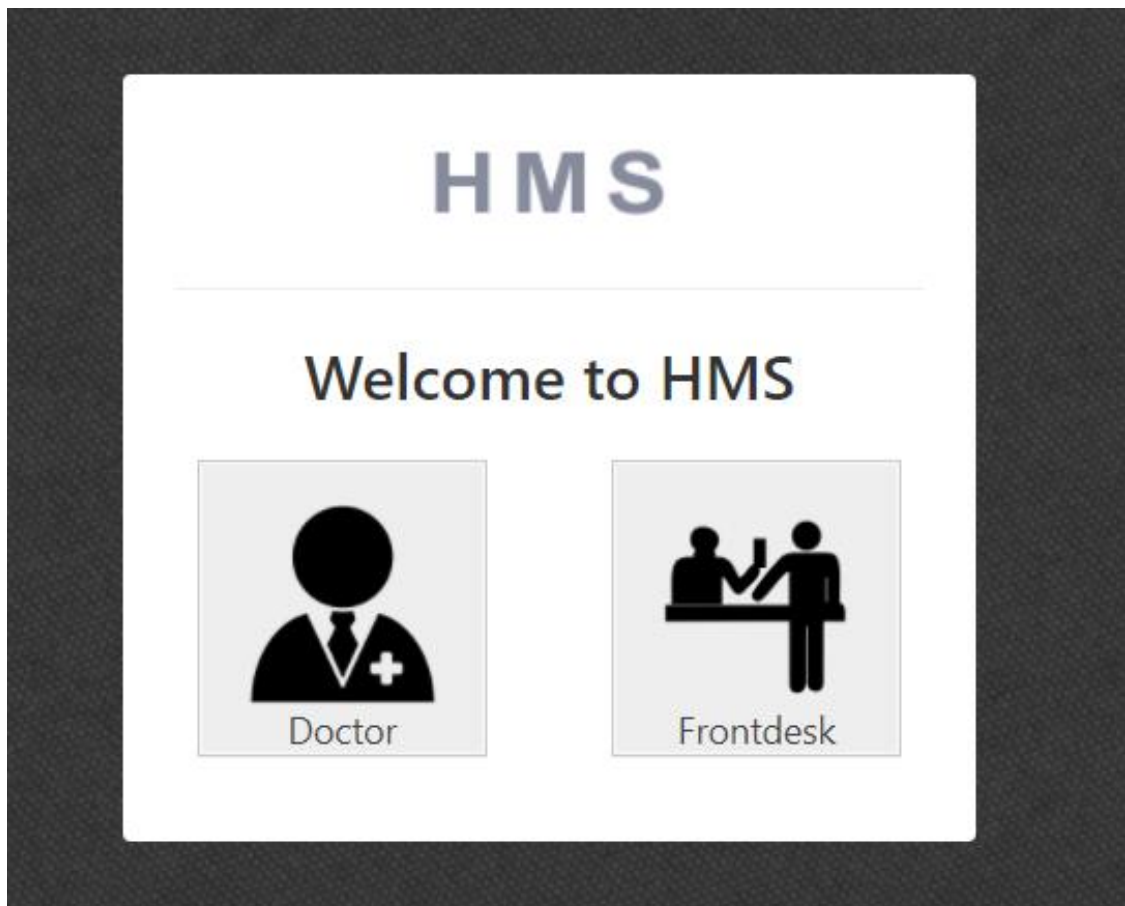


Figure 3.5 – Entity Relationship Diagram

### 3.9. User interface

The following section provides only a few user interfaces in order to show the interface structure of the system. Here shows only home page, doctor patient, system administrator patient. Please Refer Appendix C - User Documentation for the rest of the interface designs

#### **Home page**



*Figure 3.6 – Home Page*

Figure 3.6 shows the home page of the site. Create home page light colourful and attractively to take Patient attraction. Home page contain products images in slider, image gallery and categories section. These images use to display to encourage increasing health as a wealth.

## Doctor Dash board

The screenshot shows the HMS Doctor Dashboard. At the top, there are navigation tabs for Dashboard, Calendar, Patients, and My Notes. The main header reads 'All Patients; Doctor' and identifies the user as Alexander Tsiakopoulos, Doctor. A search bar is present for 'Patient ID, Name or Mobile'. Below this is a table listing patient records.

Patient Id	Sex	First Name	Last Name	Dob	Mobile	Address	City	State Province	Postal Code	Actions
155	Female	Jayani	Kottigoda	2017-10-02	0777100506	540, Aggona	Angoda	Western Province	2222	View History
154	Male	Madhara	Fonseka	1988-05-23	0723673556	444/AThalangama	colombo			View History
153	Male	Sarath	Kodagoda	1956-02-06	7709799408	444/AThalangama	Nawala	Western Province	17456	View History
152	Male	Tanjana	Fonseka	1983-11-28	0777100307	420/a, Nawala	Koswatta	Western Province	17456	View History
146	Female	Gouri	Silva	1969-04-24	0718665458	408 Blick Ford Suite 832	West Evertbury	Hawaii	79173	View History
145	Male	Gunapala	Robert	1970-07-15	0726753556	14/1, Kotahena	Kadawatha	Western Province	85804-7050	View History
144	Male	Siripala	Malhothra	1993-12-27	0765623547	769 Gibson Mawatha	Katukurunda	Western Province	64980-8806	View History
142	Female	Nadini	Kosgama	1983-02-10	0783659236	16397 sigiri Lakes	Kurunagala	Sabaragamuwa Province	76988	View History

Figure 3.7 –doctor Dash Board

## System Administrator Dashboard

Figure 3.8 shows the system administrator patient. By using this patient system admin can easily add news, change clinic schedules, view clinic details, view feedbacks, view inquiries, and add users to site. Apart from those things system admin can view reports about patient’s login and doctor’s logins; also he can send emails to patients to inform new events.

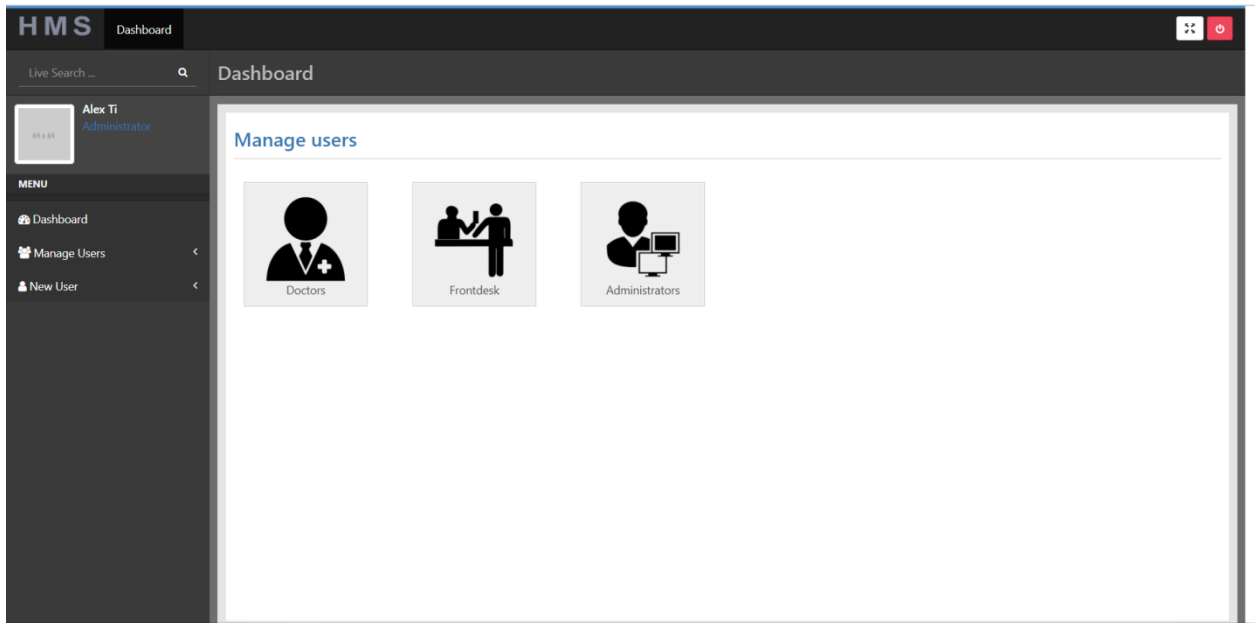


Figure 3.8 –System Administrator Dashboard

## **Chapter 4 – Implementation**

This chapter will describe about the implementation process, information related to the project. This will include hardware and software requirements of the system, tools and techniques used while developing the system and some of the important code segments of the project.

### **4.1. Hardware and Software Requirements**

Web base system has two ends. Back end and front end. Back end use to store the whole system and client can access it from their PC (which is the client side). Hardware and Software requirements of these two sides list down in below.

#### **Hardware**

- Intel Pentium 4 - 2.8Ghz or higher processor
- 512MB or Higher RAM
- 40GB or more Hard Disk
- SVGA monitor with minimum 1024 x 768 resolution
- Internet Connection
- Bar chord scanner

#### **Software**

- Windows XP or Higher Windows version
- Microsoft Internet Explorer 8.0 or Higher(for Windows),
- Firefox 3.5 or Higher(for both Windows and Linux)
- Apache 2.2.11 Server
- PHP 5.3.0
- MySQL Server 5.1.36

- Enterprise Architect
- Notepad++
- Adobe Dreamweaver CS3
- Adobe Photoshop CS3
- Adobe Flash CS3

### **Technology**

- PHP (Hypertext Pre Processor) -Main development language
- MySQL–Database development
- CSS - Develop plain interfaces
- JavaScript -Client-side validation.
- JQuery

### **Client side**

### **Hardware**

- Intel Pentium 4 – 2.0Ghz or higher processor
- 256MB or Higher RAM
- 20GB or more Hard Disk
- Internet Connection
- Bar chord scanner

### **Software**

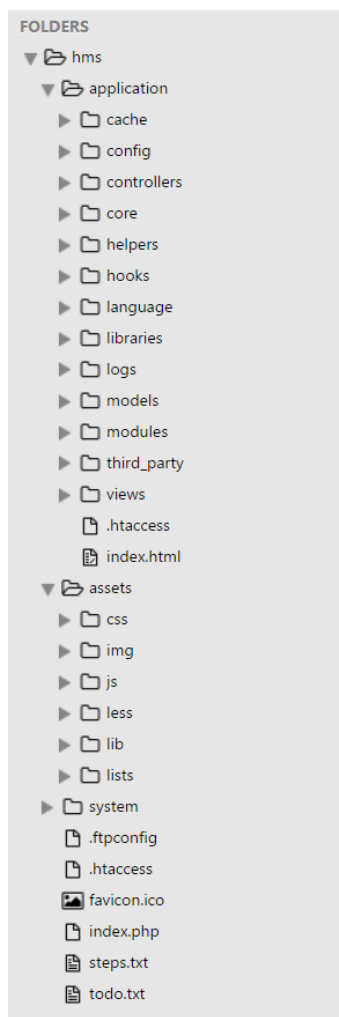
- Windows XP or Higher Windows version or Ubuntu 9.04 or Higher
- Microsoft Internet Explorer version 8.0 or Higher(for Windows),
- Firefox 3.5 or Higher (for both Windows and Linux)
- Adobe Flash Player version 10.0 or higher
- For the Server Only: - Apache 2.2.x Server
- PHP 5.2.3 or Higher
- MySQL Server 5 or Higher



## 4.2. Module Structure of the System

As mentioned early, this system is build using HMVC architecture. The Model is used for database interaction. Model contains functions that help to retrieve, insert, and update information in the database. Controller serves as an intermediary between the models, the view. It is send requests and responses back and forth between models, views and helpers. Views represent the GUI interfaces. It is the being presented the information to the users. Figure 4.1 shows how system code structure is organized.

The Controllers folder contents all the controller files in the system and models folder content model files. All the pages in the site includes in the views folder. In the views folder there is header and footer pages.



*Figure 4.1 – Code Structure of the System*

### 4.3 Major Code Segments

This application contains a login process. It gets the username and password from the user and validates them.

Login form has been included in the home page. After the username and password are submitted, login.php file is executed.

login.php

In the submission, the password is validated. This code checks whether the submitted password is empty or not. If not empty, it executes the database queries. If empty, it gives an error message mentioning that the password cannot be empty.

```
public function index($type=null)
    {
        if(!$type)
            redirect('home');
        if($this->input->post())
            $this->validateLogin($type);
        $data['type'] = ucfirst($type);
        $this->load->view('login',$data);
    }
```

Values submitted from the login form are sent by the POST method. Here those values have been stored in two separated variables `$username` and `$password`. After that it connects with the database.

This code selects the user details from the login table in the database where the user id equals to the submitted username.

`//$this->input->post()` is a built in CI function that holds all the values

```
private function validateLogin($type){
    $email = $this->input->post('email');
    $password = $this->input->post('password');
    $this->form_validation->set_rules('email', 'Email',
    'required|valid_email');
    $this->form_validation->set_rules('password', 'Password',
    'required',
    array('required' => 'You must provide a %s.')
    );
}
```

If one row is returned by the above query, it fetches the relevant password and stores it in a session.

```
If( mysqli_num_rows($result)==1){
    $row = mysqli_fetch_array($result);
    $_SESSION['password']=$row['password'];
}
```

Then it compares the user entered password with the fetched password from the database. If those two are equal then the login succeeds.

Some other validations also have been done.

Both username and password should be provided.

Submission with a blank username is not allowed.

Submission with a blank password is not allowed.

## REGISTRATION

Database connection is set first. All the data submitted from the form are fetched by the POST method. Those values are inserted in to the patient table in the database. If insertion is successful, then it gives a message mentioning about the successful registration and the username of the user.

```
class Patients extends MX_Controller {
```

All the functions that must be executed after user's login should be on this controller. If user is NOT logged in and then tries to navigate to this controller "front desk" Then he/she will be automatically redirected to "login" controller. This is why we use construct function that is being executed before any other function on this controller.

```
class Patient_registration extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        $this->load->helper("url");
        $this->load->model("Patient_model");
    }
    public function index()
    {
        $last_id      =$this->Patient_model->get_last_id();
        $user['page_id']  = 'Patient_registration_view';
        $user['get_last_id'] =$last_id;
        $this->load->view('frontdeskTemplate',$user);
    }
    redirect('frontdesk/patients/all');
}
/*
```

```
* Listing of patients
*/
function all($page=1, $search=null)
{

    $params['limit'] = 20;
    $params['offset'] = 20 * ($page - 1);
    $this->load->library('pagination');

    $config = $this->config->item('pagination');
    $config['base_url'] = base_url('frontdesk/patients/all');
    $config['total_rows'] = $this->Patients_model->get_all_patients_count();
    $config['use_page_numbers'] = TRUE;
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_tag_open'] = '<li>';
    $config['first_tag_close'] = '</li>';
    $config['last_tag_open'] = '<li>';
    $config['last_tag_close'] = '</li>';
    $config['next_tag_open'] = '<li>';
    $config['next_tag_close'] = '</li>';
    $config['prev_tag_open'] = '<li>';
    $config['prev_tag_close'] = '</li>';
    $config['cur_tag_open'] = '<li class="disabled"><span>';
    $config['cur_tag_close'] = '</span></li>';
    $config['num_tag_open'] = '<li>';
    $config['num_tag_close'] = '</li>';

    $this->pagination->initialize($config);
    $data['patients'] = $this->Patients_model->get_all_patients($params);
    $data['title'] = "All Patients | Frontdesk";
    $data['custom_class'] = "patients";

    $this->load->view('header',$data);
    $this->load->view('patients/all');
    $this->load->view('footer');
}

/*
* Adding a new patient
*/
function add()
{

    $this->load->library('form_validation');

    $this->form_validation->set_rules('mobile','Mobile','required|numeric');
    $this->form_validation->set_rules('dob','Dob','required');
    $this->form_validation->set_rules('sex','Sex','required|alpha');
```

```
        $this->form_validation->set_rules('last_name','Last
Name','required|alpha');
        $this->form_validation->set_rules('first_name','First
Name','required|alpha');
        $this->form_validation->set_rules('home_phone','Home
Phone','numeric');
        $this->form_validation->set_rules('emergency_mobile','Emergency
Mobile','numeric');
        $this->form_validation->set_rules('family_doctor_first_name','Family
Doctor First Name','alpha');
        $this->form_validation->set_rules('family_doctor_last_name','Family
Doctor Last Name','alpha');
        $this->form_validation-
>set_rules('family_doctor_phone_number','Family Doctor Phone Number','numeric');
        $this->form_validation->set_rules('emergency_first_name','Emergency
First Name','alpha');
        $this->form_validation->set_rules('middle_name','Middle
Name','alpha');
        $this->form_validation->set_rules('marital_status','Marital
Status','alpha');

        if($this->form_validation->run())
        {
            $params = array(
                'sex' => $this->input->post('sex'),
                'marital_status' => $this->input->post('marital_status'),
                'first_name' => $this->input->post('first_name'),
                'middle_name' => $this->input->post('middle_name'),
                'last_name' => $this->input->post('last_name'),
                'dob' => $this->input->post('dob'),
                'home_phone' => $this->input->post('home_phone'),
                'mobile' => $this->input->post('mobile'),
                'street_number' => $this->input->post('street_number'),
                'city' => $this->input->post('city'),
                'state_province' => $this->input->post('state_province'),
                'postal_code' => $this->input->post('postal_code'),
                'emergency_first_name' => $this->input-
>post('emergency_first_name'),
                'emergency_last_name' => $this->input-
>post('emergency_last_name'),
                'emergency_mobile' => $this->input-
>post('emergency_mobile'),
                'emergency_relationship' => $this->input-
>post('emergency_relationship'),
                'family_doctor_first_name' => $this->input-
>post('family_doctor_first_name'),
                'family_doctor_last_name' => $this->input-
>post('family_doctor_last_name'),
                'family_doctor_phone_number' => $this->input-
>post('family_doctor_phone_number'),
```

```
);

$patient_id = $this->Patients_model->add_patient($params);
Modules::run('generic/flash', 'success', 'New Patient Was Added!');
redirect('frontdesk/patients/edit/'.$patient_id);
}
else
{

    $data['title'] = 'Add New Patient';
    $this->load->view('header',$data);

    $this->load->view('patients/add');
    $this->load->view('footer');

}
}
```

#### 4.4 Reused Module structures

1. PHP image slider code was extract from Sourcecodester website [WWW1]. It was modified as necessary to the system.
2. Login, registration codes, validation codes and most usable codes were extracted from reference book [Effortless E-commerce 2012]. And it was modified as necessary to the system.

# **Chapter 5 - Evaluation**

## **5.1 Introduction**

Software testing is performed to verify that the completed software package functions according to the expectations defined by the requirements. The overall objective is to not to find every software bug that exists, but to uncover situations that could negatively impact the customer, usability and/or maintainability.

From the module level to the application level, different types of testing have been chosen to succeed the system development.

Testing allows developers to deliver software that meets expectations, prevents unexpected results, and improves the long term maintenance of the application.

## **5.2 The Testing Spectrum**

### **5.2.1 Unit Testing**

Unit Testing is done at the lowest level. It tests the basic unit of software, which is the smallest testable piece of software.

### **5.2.2 Integration Testing**

Integration Testing is performed when two or more tested units are combined into a larger structure. The test is often done on both the interfaces between the components and the larger structure being constructed, if its quality property cannot be assessed from its components.

### 5.2.3 Black Box Testing

The technique of testing without having any knowledge of the interior workings of the application is Black Box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

### 5.2.4 White Box Testing

White box testing is the detailed investigation of internal logic and structure of the code. White box testing is also called glass testing or open box testing. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code.

### 5.2.5 System Testing

System Testing tends to affirm the end-to-end quality of the entire system. System test is often based on the functional/requirement specification of the system. Non-functional quality attributes, such as reliability, security, and maintainability, are also checked.

### 5.2.6 Acceptance Testing

Acceptance Testing is done when the completed system is handed over from the developers to the customers or users. The purpose of acceptance testing is rather to give confidence that the system is working than to find errors.



### 5.3 Main Test Cases

#### User Management Model

	Test case	Expected Output	Actual Output	Status
<b>New user registration</b>				
1	Enter correct data with email address which is not in the table	Insert record to the table and give success message	Insert record to the table and give success message	Pass
2	Enter invalid email address	Display an error message	Give error message	Pass
3	Enter email address which is exist in the table	Display an error message	Display an error message	Pass
4	Enter invalid phone number	Display an error message	Display an error message	Pass
5	First name field empty	Display an error message	Display an error message	Pass
6	Last name field empty	Display an error message	Display an error message	Pass
7	Address field empty	Display an error message	Display an error message	Pass
8	City field empty	Display an error message	Display an error message	Pass
9	Phone Number field empty	Display an error message	Display an error message	Pass
10	Enter invalid mobile number	Display an error message	Display an error message	Pass
11	Email field empty	Display an error message	Display an error message	Pass
12	Password field empty	Display an error message	Display an error message	Pass
13	Confirm Password field empty	Display an error message	Display an error message	Pass
14	Submit form with no mobile number	Insert record to the table and give success message	Insert record to the table and give success message	Pass
15	Submit form with incorrect data	Display an error messages in relevant fields	Display an error messages in relevant fields	Pass
16	Submit form with mismatching password	Display an error messages in confirm password fields	Display an error messages in confirm password fields	Pass
17	Submit form with not enter the captcha value	Display an error messages in captcha image fields	Display an error messages in captcha image field	Pass
18	Submit form with mismatching captcha	Display an error messages in	Display an error messages in	Pass

	value	captcha image fields	captcha image fields	
19	Enter valid email and keep other fields null	Display an error messages in relevant fields	Display an error messages in relevant fields	Pass
20	Submit form with no field filled	Display an error messages in relevant fields	Display an error messages in relevant fields	Pass
<b>System login</b>				
21	Click login button without filled the username and password	Display an error message	Display an error message	Pass
22	Enter invalid email address as username	Display an error message	Display an error message	Pass
23	Click login with valid email and No password	Display an error message	Display an error message	Pass
24	Enter a password with valid email which is not in login table and click login button	Display an error message	Display an error message	Pass
25	Enter a password with valid email which is not in login table	Display an error message	Display an error message	Pass
26	Enter correct email which is exist the login table with wrong password	Display an error message	Display an error message	Pass
27	Enter email which is not exist the login table with correct password	Display an error message	Display an error message	Pass
28	Enter correct email and password which are exist in the login table and log as customer	Load the home page and display username in the header bar and with logout button	Load the home page and display username in the header bar and with logout button	Pass
29	Enter correct email address and password and log as system administrator	Load the system bashbord with relevant pages	Load the system bashbord with relevant pages	Pass
30	Enter correct email address and password and log as system operator	Load the system bashbord with relevant pages	Load the system bashbord with relevant pages	Pass
31	Enter correct email address and password and log as management	Load the management patient	Load the management patient	Pass

*Table 5.1 – Test Cases for User Management*

#### 5.4. User evaluation

Unit testing and system testing were done by developer using his own data set. When the developer finished testing the system, few random users were selected to test the

system. Then this system gave to the client for acceptance testing. There for a questionnaire was given to the 10 client side users. Table 5.2 shows the summery result of that survey.

<b>Question</b>	<b>Very Good</b>	<b>Good</b>	<b>Fair</b>	<b>Bad</b>
Easy of understanding the system	7	2	1	
Coverage of required functionalities	6	4		
Interactivity of the system	8	2		
Degree of Information provided in Reports	9	1		
Ability to understand error messages	6	3	1	
Efficiency of the system	2	6	2	
Easiness of report generation	6	3	1	
Site content	4	6		
Site Images	3	5	2	
Potential benefits gain through the system	7	3		

*Table 5.2 – Summery of Questionnaire Result*

## **Chapter 6 – Conclusion**

### **6.1. Introduction**

In the duration of the project development quiet priority has been given to fulfil the client requirements. And it was succeed for the dedication of effort. Functional and non-functional requirements were accomplish in the development of the system.

After completing the analysis phase, functional requirement and non-functional requirements were parallel compared in the designing phase and implementation phase to accomplish the goal of client satisfaction.

User interfaces were built simple as client requirements. It was developed with the ease of use as most users are non-technical users.

### **6.2. Problems Encountered**

Lack of knowledge was the major problem was encountered in the development of the system. It was gathered by references books, internet resources and from discussion with technical persons.

Adding the bar chord was bit confused since bar chord scanners technology was new for the vocabulary.

### **6.3. Lessons Learnt**

Proper time management and planning that are the most important aspects of thisproject are the areas that were improved during and as a result of the project. Time scheduling is a major aspect of the project. It worth as it was effect the cost of the project.

Knowledge gain through this project is really valuable. By doing this project I learn how to build up a project from the initial stage. How to gather information about

project and analyzed the gathered requirement. Then I learn about how to follow software lifecycle.

Gain knowledge about HMVC architecture and how to develop a project using HMVC architecture. Gain knowledge about OOP and codeigniter framework. Learn how develop a system using framework.

When create the UI of the system learn a lot about CSS and what are the ways to

- Knowledge of developing applications using PHP and MySQL.
- Knowledge of developing a system using CodeIgniter framework
- Knowledge of jQuery ajax, json
- Develop web applications to comply with web standards.
- Work with limited resources and time to achieve project goals.
- Working and interacting with real world clients and people

#### 6.4. Critical assessment of the project

However while achieving the requirements analysed in the early stages of the project, some requirements were hard to handle and achieve due to lack of experience in this health sector area. But with the help of the supervisor and gaining knowledge by researching about the health sector and suitable technologies, these requirements were achieved successfully.

Following are the critical modules achieved during this project with a great effort.

- Managing patient's appointments by using a calendar.

It was hard to select a calendar which is easy and simple as doctor's requirement. Although it should be simple from front end of the system for users, back end was critical to handle.

This simple calendar was design to give functionality for email handling from backend. When a patient appointment is enrolled by the front desk or an appointment was cancelled or changes the date of the appointment an email was sent for the doctor to notify about the new event.

This simple calendar was achieved by using of full calendar java script handler.

- Making a data base for all types and all brands for medicines with necessary dosages.

It was analysed that more than twenty thousands of medicines are in used at this moment with thousands of brands. Every medicine has different dosages.

It was hard to create a data base for this requirement with lack of time and human forces. It takes time and as well as it is costly to pay for data entry workers. Then have to spent more cost in storage media.

Hence with the discussion with client, it was sorted out to make a simple data base with frequently used medicine only. So it was reduced the work load and complexity of the data base.

Although the system was implemented with all requirements expected by the client some requirement were postponed to the future work due to inconvenience of the client's budget.

Following is the module was postponed due to inconvenience.

- Connecting an SMS gateway.

It was decide to implement this module in the future to send SMS for patients, as it takes more cost to implement an SMS gateway and for choosing an SMS service provider. Client has agreed to provide necessary cost in the future with the development of his clinic.

## 6.5. Future Work

Connecting an SMS gateway will give the facility to communicate the users and it will increase the communication interaction between doctors and the patients.

Managing reference data budgeting will lead to success of the medical center in future.

## References

- [WWW1]<http://www.sourcecodester.com/tutorials/php/4917/image-slideshow-using-php-and-simple->
- [WWW2] Database eLearning. <http://db.grussell.org> [21.07.2013]
- [WWW3] tutorialspoint.<http://www.tutorialspoint.com/index.htm>
- Murach's PHP and MySQL- JOEL MURACH, RAY HARRIS-Second Indian Reprint October 2012

# Appendix A – System documentation

## System Manual

Step 01

Install WAMP

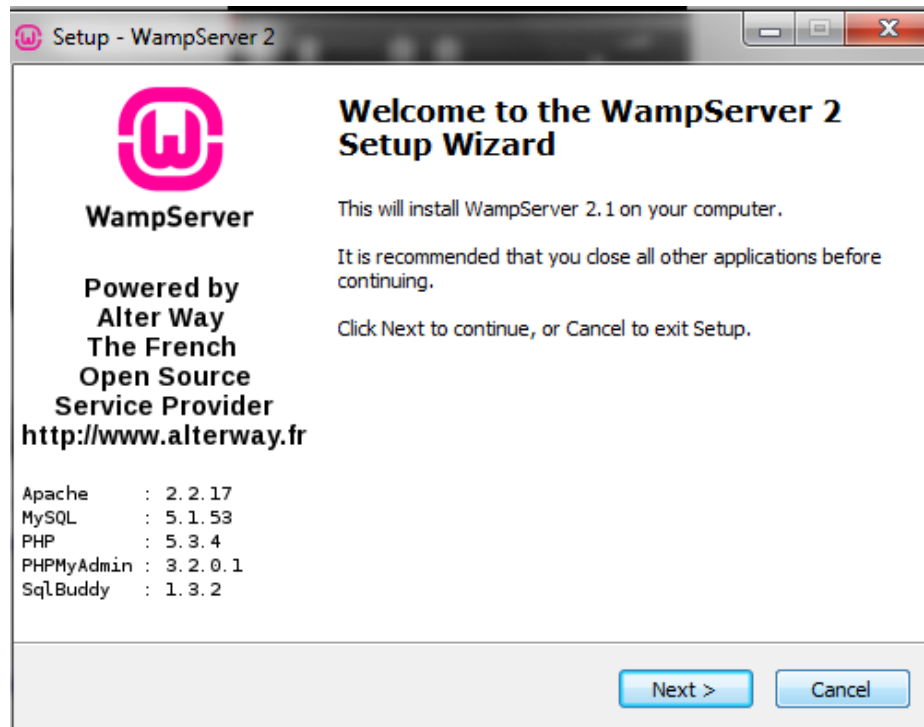


Figure A.1 WAMP installation

Step 02

Go to www Directory

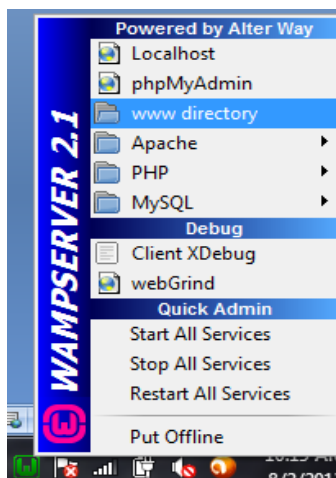


Figure A.2 Directory

Step03

Copy pastes the open arc folder in to it.



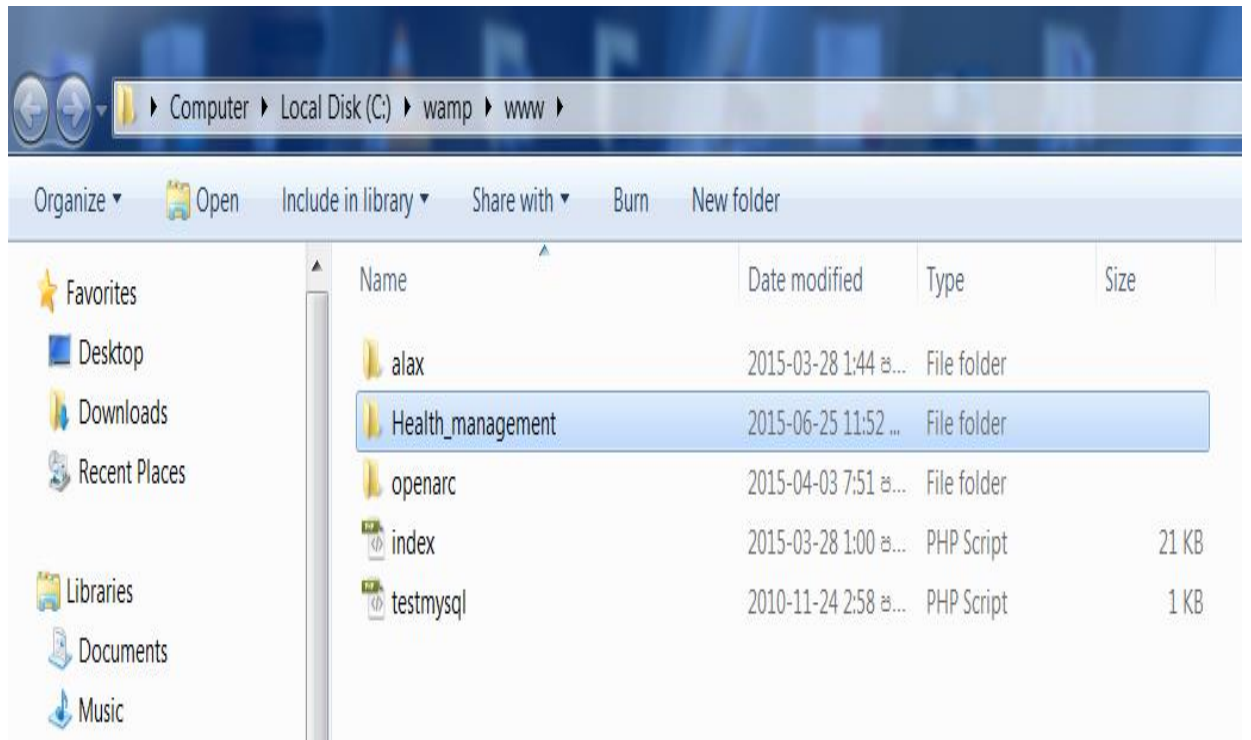


Figure A.3 Folder in Directory

## Appendix B – Design Documentation

Activity Diagram for Patient Registration at front desk

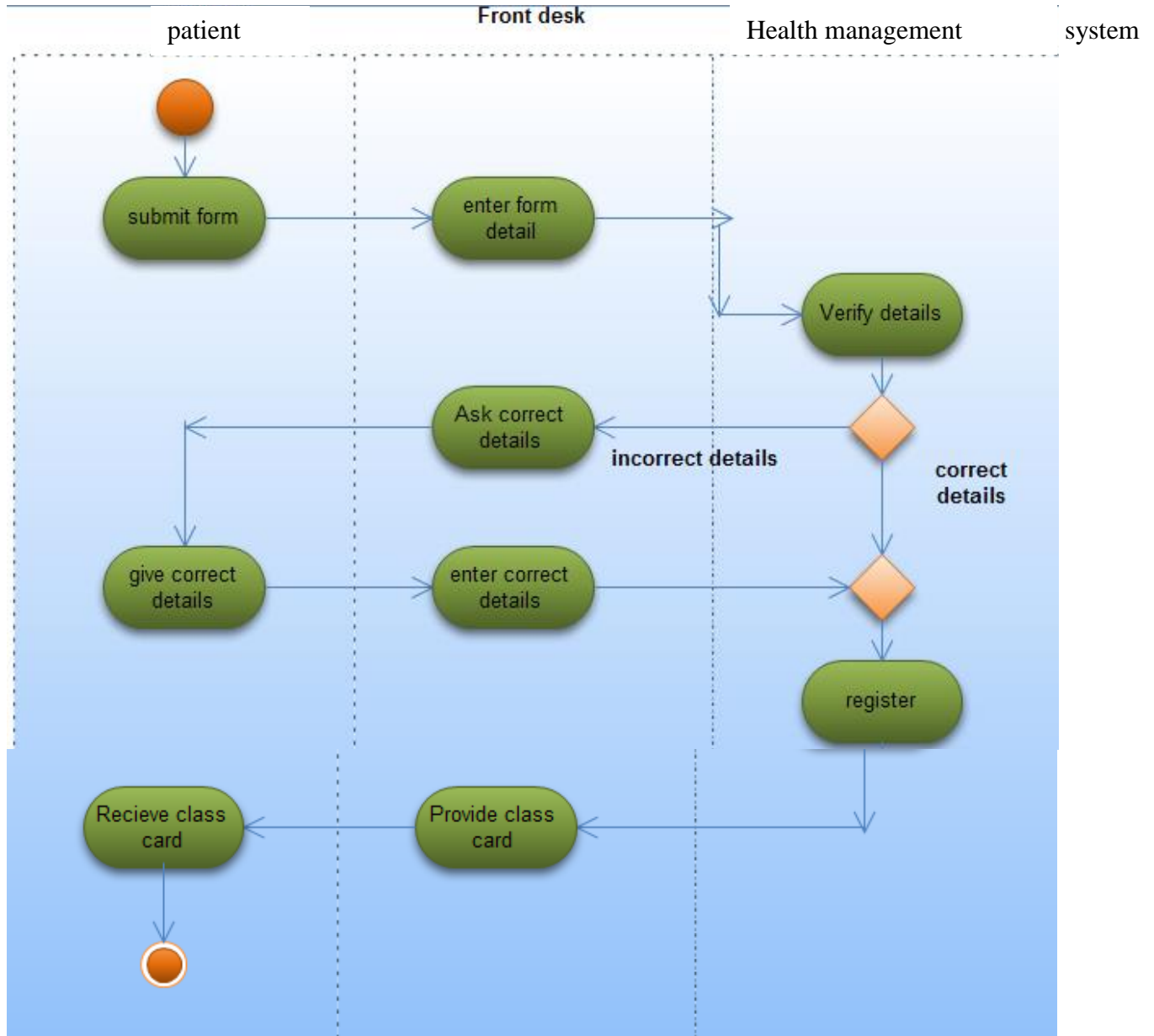


Figure B.1 Patient Registration Activity Diagram

Activity Diagram for Patient Registration online

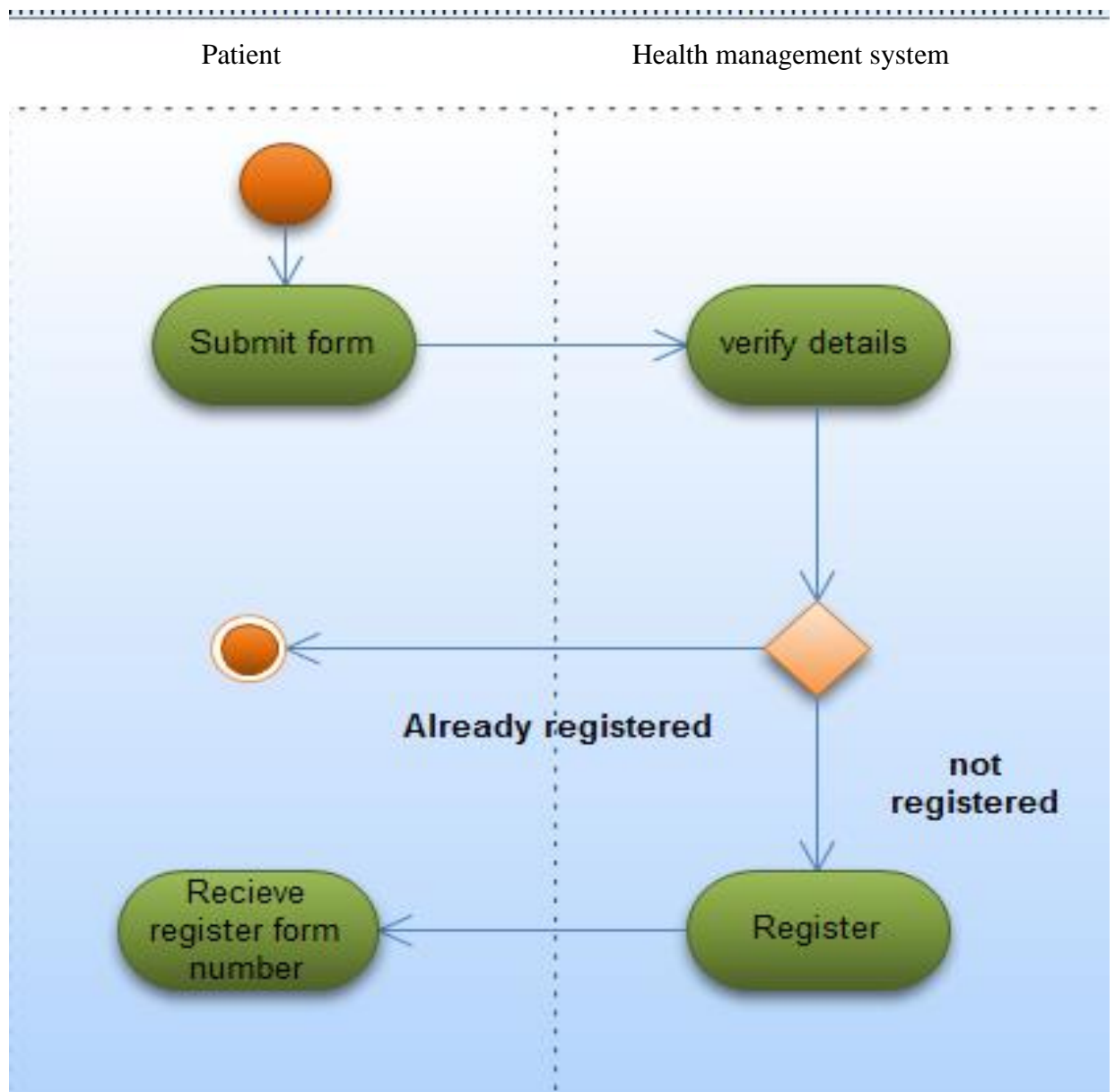


Figure B.2 Patient Renewal Activity Diagram

Activity Diagram for Patient Medical recording

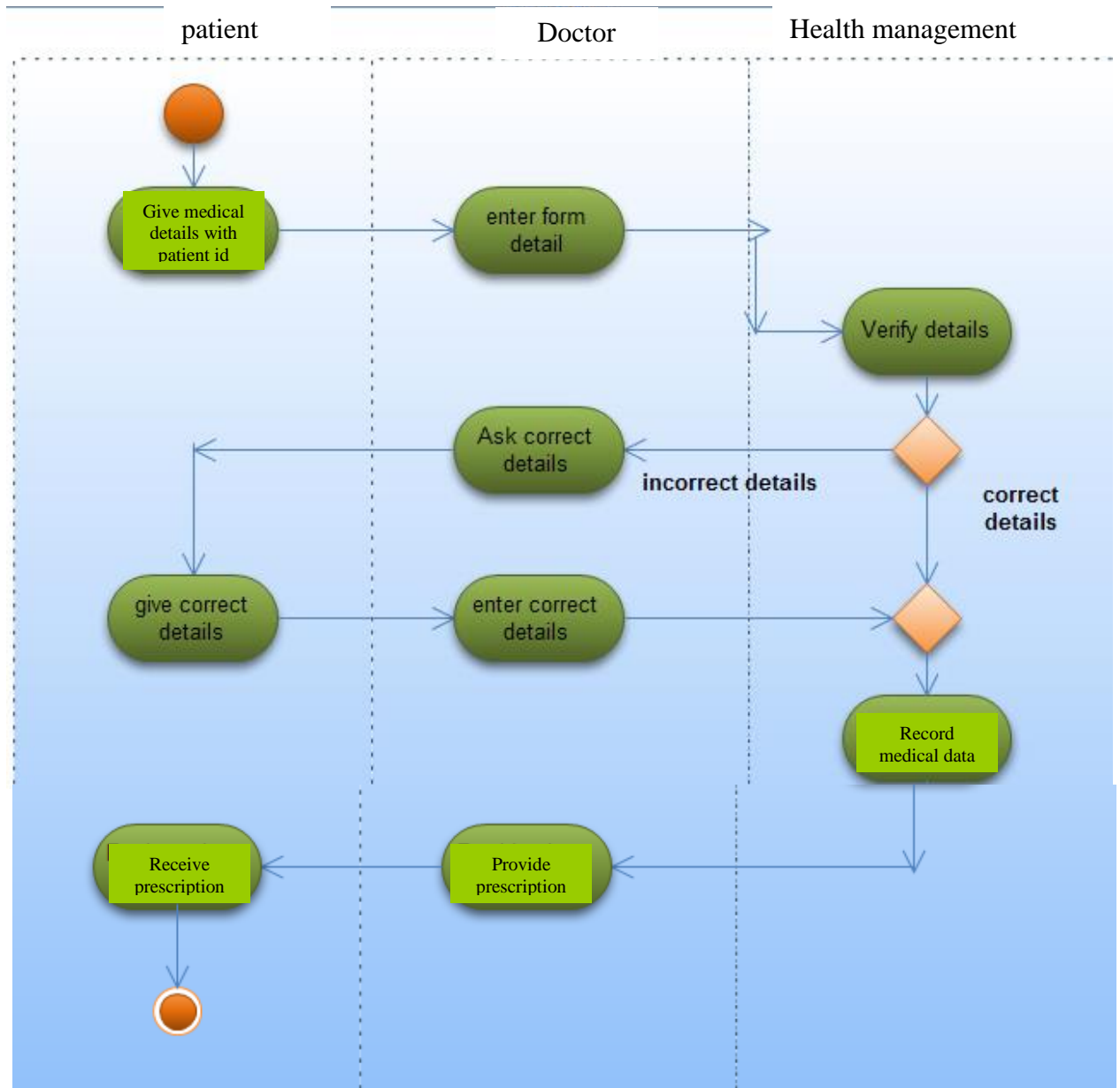
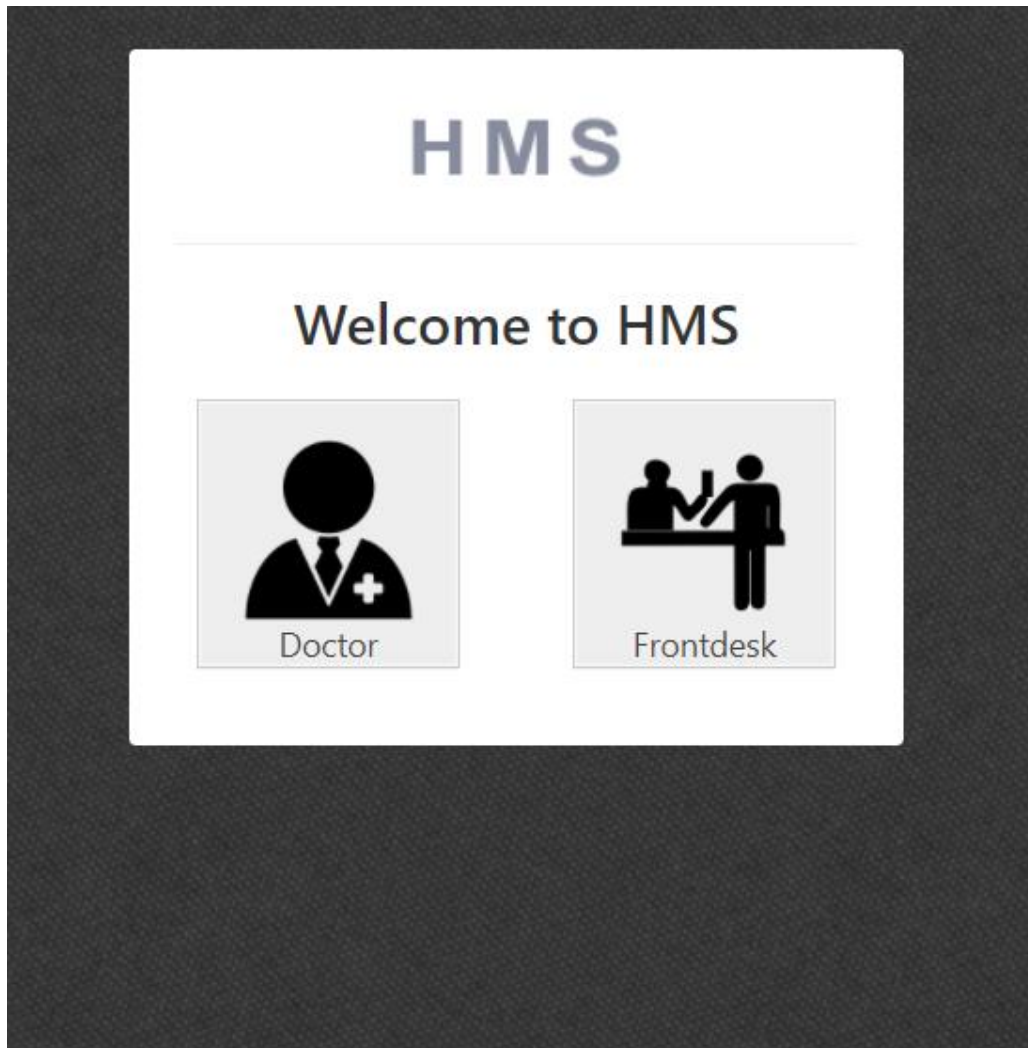


Figure B.3 Patient Medical Recorder Activity Diagram

## **Appendix C – User Documentation**

### **Main Interface**



*Figure C.1 Main Interface*

In main interface log in facility and registration facility has been provided. A video is displaying to guide to use the system.

## Login Interface

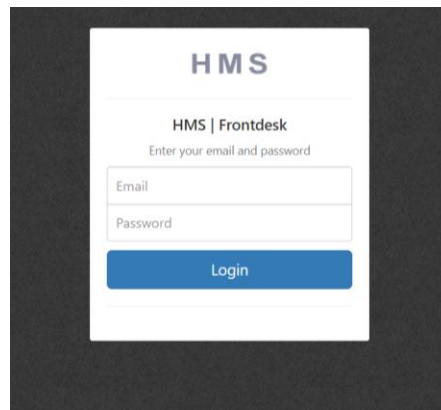


Figure C.2 Login Interface

User name is the unique bar chord. When the user inserts the correct User name and the Password System will direct the user to the User profile interface.

## Doctor Dash board

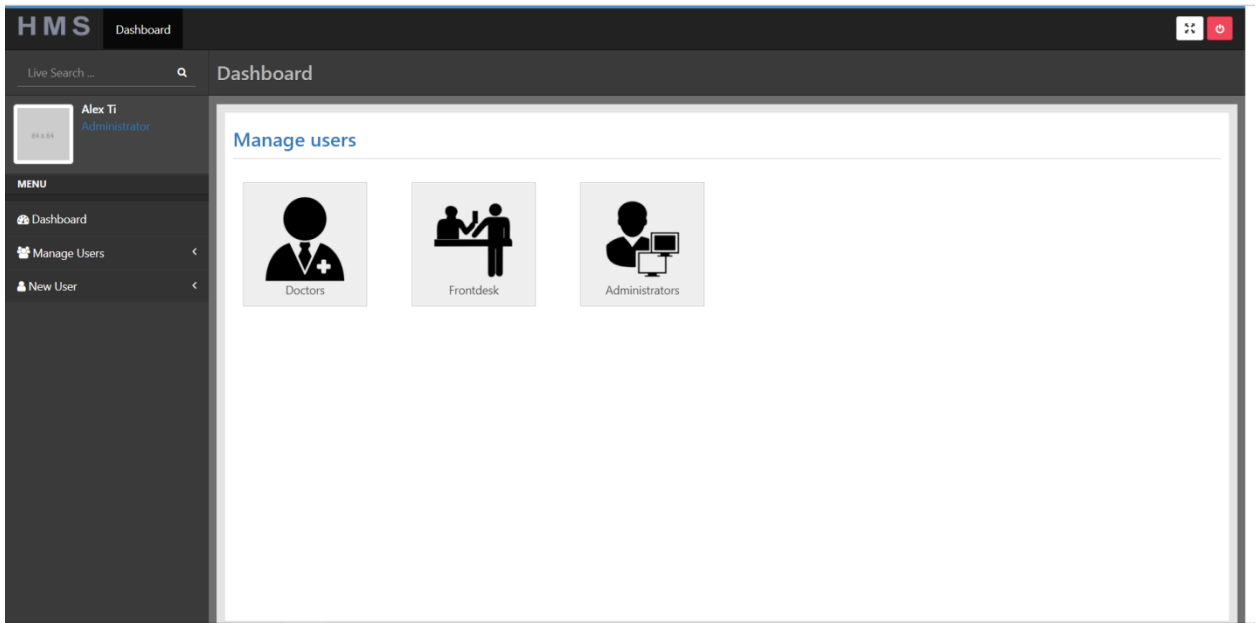
Patient Id	Sex	First Name	Last Name	Dob	Mobile	Address	City	State Province	Postal Code	Actions
155	Female	Jayani	Kottigoda	2017-10-02	0777100506	540, Aggona	Angoda	Western Province	2222	<a href="#">View</a> <a href="#">History</a>
154	Male	Madhara	Fonseka	1988-05-23	0723673556	444/AThalangama	colombo			<a href="#">View</a> <a href="#">History</a>
153	Male	Sarath	Kodagoda	1956-02-06	7709799408	444/AThalangama	Nawala	Western Province	17456	<a href="#">View</a> <a href="#">History</a>
152	Male	Tanjana	Fonseka	1983-11-28	0777100307	420/a, Nawala	Koswatta	Western Province	17456	<a href="#">View</a> <a href="#">History</a>
146	Female	Gouri	Silva	1969-04-24	0718665458	408 Blick Ford Suite 832	West Evertbury	Hawaii	79173	<a href="#">View</a> <a href="#">History</a>
145	Male	Gunapala	Robert	1970-07-15	0726753556	14/1, Kotahena	Kadawatha	Western Province	85804-7050	<a href="#">View</a> <a href="#">History</a>
144	Male	Siripala	Malhothra	1993-12-27	0765623547	769 Gibson Mawatha	Katukurunda	Western Province	64980-8806	<a href="#">View</a> <a href="#">History</a>
142	Female	Nadini	Kosgama	1983-02-10	0783659236	16397 sigiri Lakes	Kurunagala	Sabaragamuwa Province	76988	<a href="#">View</a> <a href="#">History</a>

Figure C.3 –doctor Dash Board

## System Administrator Dashboard

Figure 3.10 shows the system administrator patient. By using this patient system admin can easily add news, change clinic schedules, view clinic details, view feedbacks, view inquiries, and add users to site. Apart from those things system admin can view reports

about patient's login and doctor's logins; also he can send emails to patients to inform new events.



*Figure C.4 –System Administrator Dashboard*

## Appendix D – Management Reports

Since Dr Krunarathne is a small scale clinic they decided to generate their reports and update to the system.

### Patient Report

Generate by Doctor.

A	B	C	D	E
<b>Patient Name</b>	Tanjana Fonseka			
<b>Age</b>	33 years old   1983-11-28			
<b>Sex</b>	Male			
<b>Address</b>	420/a, Nawala, Koswatta, Western Province,17456			
<b>Date</b>	<b>Doctor Name</b>	<b>Doctor Notes</b>	<b>Symptoms</b>	<b>Diseases</b>
2017-10-14 02:07:27	George Maratof		Mouth sores; Abdominal pain and cramping; Fatigue; Diarrhea	Zika Virus; Water-related Diseases; Salmonella; Heart Disease; Hepatitis; Meningitis; Methicillin-resistant Staphylococcus aureus; Epilepsy; Flu; Chronic Fatigue Syndrome; Fetal Alcohol Spectrum Disorders
2017-10-16 14:11:27	Alexander Tisakopoulos		Blood in your stool; Delayed growth or sexual development, in children; Inflammation of the liver or bile ducts; Inflammation of skin, eyes and joints	Heart Disease; Hepatitis; Genital Herpes; Flu; Fetal Alcohol Spectrum Disorders

Figure D.1 Progress Report



## Prescription

Budget report is generated by the accountant and then update to the system is done y administrator.

<b>Prescription</b>		Issued: Nov` 06, 2017   15:23	No: 18
<b>Patient Name</b>	Alex Tisakov		
<b>Address</b>	Navarinou 6, P. Faliro, Athens, Greece		
<b>Age</b>	45 years old   1985-12-02		
<b>Sex</b>	Male		
<b>Medicine Title:</b>	<b>Doctor notes</b>		
Alimta			
Gardasil			

*Figure D.2 Prescription*

## Appendix E - Test Results

This section includes the other test cases that use to test the system. Table E.1 to E.4 shows the test cases for feedback model, inquiry model, and user profile management model.

### Feedback Model

	Test case	Expected Output	Actual Output	Status
1	Click send feedback button without fill the required fields	Display an error messages in relevant fields	Display an error messages in relevant fields	Pass
2	Message field is empty	Display an error messages	Display an error messages	Pass
If logged customer				
3	Message field empty	Display an error messages	Display an error messages	Pass
4	Email address field	Display logged user email address in the email field	Display logged user email address in the email field	Pass
5	Name field	Display logged user name in the name field	Display logged user name in the name field	Fail
6	Email field empty and phone number field empty	Display an error messages	Display an error messages	Pass
7	Click send feedback with invalid email	Display an error messages	Display an error messages	pass
8	Click send feedback with invalid phone number	Display an error messages	Display an error messages	pass
9	Click send feedback with correct data	Insert recode to the table and give success message	Insert recode to the table and give success message	Pass
Anonymous customer				
10	Empty name field	Display an error messages	Display an error messages	Pass
11	Empty message field	Display an error messages	Display an error messages	Pass
12	Empty both email address and phone number	Display an error messages	Display an error messages	Pass
13	Captcha code	Display captcha image	Display captcha image	Pass
14	Empty captcha code	Display an error messages	Display an error messages	Pass
15	Mismatching captcha code	Display an error messages	Display an error messages	Pass
16	Invalid email address	Display an error messages	Display an error messages	Pass
17	Invalid phone number	Display an error messages	Display an error messages	Pass

18	Click send feedback with correct data	Insert recode to the table and give success message		
----	---------------------------------------	---	--	--

*Table E.1 –Test Cases for Feedback Model*

**Inquiries**

	<b>Test case</b>	<b>Expected Output</b>	<b>Actual Output</b>	<b>Status</b>
1	Click send inquiry button without fill the required fields	Display an error messages in relevant fields	Display an error messages in relevant fields	Pass
2	Message field is empty	Display an error messages	Display an error messages	Pass
3	Inquiry type field empty	Display an error messages	Display an error messages	Pass
<b>If logged Patient</b>				
4	Message field empty	Display an error messages	Display an error messages	Pass
5	Email address field	Display logged user email address in the email field	Display logged user email address in the email field	Pass
6	Name field	Display logged user name in the name field		Fail
7	Email field empty and phone number field empty	Display an error messages	Display an error messages	Pass
8	Click send inquiry with invalid email	Display an error messages	Display an error messages	pass
9	Click send inquiry with invalid phone number	Display an error messages	Display an error messages	Pass
10	Click send inquiry with correct data	Insert recode to the table and give success message		
<b>Anonymous customer</b>				
11	Empty name field	Display an error messages	Display an error messages	Pass
12	Empty message field	Display an error messages	Display an error messages	Pass
13	Empty both email address and phone number	Display an error messages	Display an error messages	Pass
14	Captcha code	Display captcha image	Display captcha image	Pass
15	Empty captcha code	Display an error messages	Display an error messages	Pass
16	Mismatching captcha code	Display an error messages	Display an error messages	Pass
17	Invalid email address	Display an error messages	Display an error messages	Pass

*Table E.2 –Test Cases for Inquires Model*

## Appendix F –Code Listing

Some of the Important code parts are included below.

### Patient Registration (View)

```
function __construct(){

    parent::__construct();

    if(!$this->session->logged_in || $this->session->user_type != 'frontdesk')
    {
        redirect('login/frontdesk');
    }
    //$this->form_validation->set_error_delimiters('<div class="text-center form_errors
">', '</div>');
    $this->load->model('Patients_model');
}
```

```
function index(){
    redirect('frontdesk/patients/all');
}
/*
 * Listing of patients
 */
function all($page=1, $search=null)
{
```

```
    $params['limit'] = 20;
    $params['offset'] = 20 * ($page - 1);
    $this->load->library('pagination');

    $config = $this->config->item('pagination');
    $config['base_url'] = base_url('frontdesk/patients/all');
    $config['total_rows'] = $this->Patients_model->get_all_patients_count();
    $config['use_page_numbers'] = TRUE;
    $config['full_tag_open'] = '<ul class="pagination">';
    $config['full_tag_close'] = '</ul>';
    $config['first_tag_open'] = '<li>';
    $config['first_tag_close'] = '</li>';
    $config['last_tag_open'] = '<li>';
    $config['last_tag_close'] = '</li>';
    $config['next_tag_open'] = '<li>';
    $config['next_tag_close'] = '</li>';
    $config['prev_tag_open'] = '<li>';
    $config['prev_tag_close'] = '</li>';
    $config['cur_tag_open'] = '<li class="disabled"><span>';
    $config['cur_tag_close'] = '</span></li>';
    $config['num_tag_open'] = '<li>';
    $config['num_tag_close'] = '</li>';
```

```
$this->pagination->initialize($config);
$data['patients'] = $this->Patients_model->get_all_patients($params);
$data['title'] = "All Patients | Frontdesk";
$data['custom_class'] = "patients";

$this->load->view('header',$data);
$this->load->view('patients/all');
$this->load->view('footer');
}

/*
 * Adding a new patient
 */
function add()
{

$this->load->library('form_validation');

$this->form_validation->set_rules('mobile','Mobile','required|numeric');
    $this->form_validation->set_rules('dob','Dob','required');
    $this->form_validation->set_rules('sex','Sex','required|alpha');
    $this->form_validation->set_rules('last_name','Last Name','required|alpha');
    $this->form_validation->set_rules('first_name','First Name','required|alpha');
    $this->form_validation->set_rules('home_phone','Home Phone','numeric');
    $this->form_validation->set_rules('emergency_mobile','Emergency
Mobile','numeric');
    $this->form_validation->set_rules('family_doctor_first_name','Family Doctor
First Name','alpha');
    $this->form_validation->set_rules('family_doctor_last_name','Family Doctor
Last Name','alpha');
    $this->form_validation->set_rules('family_doctor_phone_number','Family
Doctor Phone Number','numeric');
    $this->form_validation->set_rules('emergency_first_name','Emergency First
Name','alpha');
    $this->form_validation->set_rules('middle_name','Middle Name','alpha');
    $this->form_validation->set_rules('marital_status','Marital Status','alpha');

    if($this->form_validation->run())
    {
        $params = array(
            'sex' => $this->input->post('sex'),
            'marital_status' => $this->input->post('marital_status'),
            'first_name' => $this->input->post('first_name'),
            'middle_name' => $this->input->post('middle_name'),
            'last_name' => $this->input->post('last_name'),
            'dob' => $this->input->post('dob'),
            'home_phone' => $this->input->post('home_phone'),
            'mobile' => $this->input->post('mobile'),
            'street_number' => $this->input->post('street_number'),
            'city' => $this->input->post('city'),
            'state_province' => $this->input->post('state_province'),
            'postal_code' => $this->input->post('postal_code'),
            'emergency_first_name' => $this->input-
>post('emergency_first_name'),
```

```
        'emergency_last_name' => $this->input-
>post('emergency_last_name'),
        'emergency_mobile' => $this->input-
>post('emergency_mobile'),
        'emergency_relationship' => $this->input-
>post('emergency_relationship'),
        'family_doctor_first_name' => $this->input-
>post('family_doctor_first_name'),
        'family_doctor_last_name' => $this->input-
>post('family_doctor_last_name'),
        'family_doctor_phone_number' => $this->input-
>post('family_doctor_phone_number'),
    );

    $patient_id = $this->Patients_model->add_patient($params);
    Modules::run('generic/flash', 'success', 'New Patient Was Added!');
    redirect('frontdesk/patients/edit/'.$patient_id);
}
else
{

    $data['title'] = 'Add New Patient';
    $this->load->view('header',$data);

    $this->load->view('patients/add');
    $this->load->view('footer');

}
}

/*
 * Editing a patient
 */
function edit($patient_id)
{
    // check if the patient exists before trying to edit it
    $data['patient'] = $this->Patients_model->get_patient($patient_id);

    if(isset($data['patient']['patient_id']))
    {
        $this->load->library('form_validation');

        $this->form_validation->set_rules('mobile','Mobile','required|numeric');
        $this->form_validation->set_rules('dob','Dob','required');
        $this->form_validation->set_rules('sex','Sex','required|alpha');
        $this->form_validation->set_rules('last_name','Last Name','required|alpha');
        $this->form_validation->set_rules('first_name','First Name','required|alpha');
        $this->form_validation->set_rules('home_phone','Home Phone','numeric');
        $this->form_validation->set_rules('emergency_mobile','Emergency
Mobile','numeric');
        $this->form_validation->set_rules('family_doctor_first_name','Family Doctor
First Name','alpha');
        $this->form_validation->set_rules('family_doctor_last_name','Family Doctor
Last Name','alpha');
```

```

        $this->form_validation->set_rules('family_doctor_phone_number','Family
Doctor Phone Number','numeric');
        $this->form_validation->set_rules('emergency_first_name','Emergency First
Name','alpha');
        $this->form_validation->set_rules('middle_name','Middle Name','alpha');
        $this->form_validation->set_rules('marital_status','Marital Status','alpha');

        if($this->form_validation->run())
        {
            $params = array(
                'sex' => $this->input->post('sex'),
                'marital_status' => $this->input-
>post('marital_status'),
                'first_name' => $this->input->post('first_name'),
                'middle_name' => $this->input->post('middle_name'),
                'last_name' => $this->input->post('last_name'),
                'dob' => $this->input->post('dob'),
                'home_phone' => $this->input->post('home_phone'),
                'mobile' => $this->input->post('mobile'),
                'street_number' => $this->input-
>post('street_number'),
                'city' => $this->input->post('city'),
                'state_province' => $this->input-
>post('state_province'),
                'postal_code' => $this->input->post('postal_code'),
                'emergency_first_name' => $this->input-
>post('emergency_first_name'),
                'emergency_last_name' => $this->input-
>post('emergency_last_name'),
                'emergency_mobile' => $this->input-
>post('emergency_mobile'),
                'emergency_relationship' => $this->input-
>post('emergency_relationship'),
                'family_doctor_first_name' => $this->input-
>post('family_doctor_first_name'),
                'family_doctor_last_name' => $this->input-
>post('family_doctor_last_name'),
                'family_doctor_phone_number' => $this->input-
>post('family_doctor_phone_number'),
            );

            $this->Patients_model->update_patient($patient_id,$params);
            redirect('frontdesk/patients/edit/'.$patient_id);
        }
        else
        {
            $data['title'] = 'Edit Patient | '.$data['patient']['first_name'].'
'.$data['patient']['last_name'];
            $this->load->view('header',$data);

            $this->load->view('patients/edit');
            $this->load->view('footer');
        }
    }
    else

```

```
        show_error('The patient you are trying to edit does not exist.');
```

```
    }
```

```
/*
```

```
 * Deleting patient
```

```
*/
```

```
function remove($patient_id)
```

```
{
```

```
    $patient = $this->Patients_model->get_patient($patient_id);
```

```
    // check if the patient exists before trying to delete it
```

```
    if(isset($patient['patient_id']))
```

```
    {
```

```
        $this->Patients_model->delete_patient($patient_id);
```

```
        redirect('frontdesk/patients/all');
```

```
    }
```

```
    else
```

```
        show_error('The patient you are trying to delete does not exist.');
```

```
}
```

```
/*
```

```
 * Return results for patient ajax search
```

```
*/
```

```
function searchPatient(){
```

```
    if(!$this->input->post('search'))
```

```
        return;
```

```
    $like = $this->input->post('search');
```

```
    $result = $this->Patients_model->searchPatient($like);
```

```
    if(!$result)
```

```
        echo "0";
```

```
    else
```

```
        echo json_encode($result);
```

```
}
```

```
/*
```

```
 * Patient search NON-ajax
```

```
 * Same as all() method but limited to search criteria
```

```
*/
```

```
function search($search)
```

```
{
```

```
    $like = $this->input->post('search');
```

```
    $data['patients'] = $this->Patients_model->searchPatient($search, 20);
```

```
    $data['title'] = "Patient Search : ".$search;
```

```
    $data['custom_class'] = "patients";
```

```
    $this->load->view('header',$data);
```

```
    $this->load->view('patients/all');
```

```
    $this->load->view('footer');
```

```
}
```



```
function view($patient_id)
{
    $data['patient'] = $this->Patients_model->get_patient($patient_id);
    $data['title'] = 'View Patient | '.$data['patient']['first_name'].' '.$data['patient']['last_name']
;
    $data['doctors'] = Modules::run('frontdesk/getDoctors');

    $this->load->view('header',$data);
    $this->load->view('patients/view');
    $this->load->view('footer');
}

//While in Patient View page when frontdesk user clicks on the "Schedule a Meeting with
Doctor"
//we run this function to set the session with Patient details and then redirect to Calendar
View
//By this way when in Calendar view we can help Frontdesk user to avoid
//searching through hundreds of Patients, because we store the information we
//need in a session variable and pass it to Calendar
function setSchedule($patient_id, $doctor_id){
    $patient = $this->Patients_model->get_patient($patient_id);

    //Get a doctor details. because the get doctor function is located
    //in another controller, we use Modules::run to get the result
    $doctor = Modules::run('frontdesk/getDoctor',$doctor_id);

    //If patient or doctor does not exist -> redirect to All Patients table
    if(!$patient || !$doctor)
        redirect(base_url('frontdesk/patients/all'));

    //Set the session variables with patient & doctor details
    $this->session->set_userdata('patient_id', $patient_id);
    $this->session->set_userdata('patient_name', $patient['first_name'].'
.$patient['last_name']);
    $this->session->set_userdata('doctor_id', $doctor->doctor_id);
    $this->session->set_userdata('doctor_name', $doctor->first_name.' '.$doctor->last_name);
    $this->session->set_userdata('doctor_specification', $doctor->specification);
    $this->session->set_userdata('doctor_email', $doctor->email);

    redirect(base_url('frontdesk/calendar'));
}

function unsetSchedule(){
    $this->session->unset_userdata('patient_id');
    $this->session->unset_userdata('patient_name');
    $this->session->unset_userdata('doctor_id');
    $this->session->unset_userdata('doctor_name');
    $this->session->unset_userdata('speciality');
    $this->session->unset_userdata('doctor_email');
```

```
    }  
}  
  
Front desk  
  
function __construct(){  
    parent::__construct();  
  
    if(!$this->session->logged_in || $this->session->user_type != 'frontdesk')  
    {  
        redirect('login/frontdesk');  
    }  
    $this->form_validation->set_error_delimiters('<div class="text-center form_errors ">',  
'</div>');  
    $this->load->model('frontdesk_model');  
  
    }  
  
public function index()  
    {  
        $data['title'] = 'Dashboard | Frontdesk';  
        $data['script'] = ' <script src="'. base_url(). 'assets/js/frontdesk.js"></script>';  
        $this->load->view('header', $data);  
        $this->load->view('frontdesk');  
        $this->load->view('footer');  
    }  
  
public function settings()  
    {  
  
        if($this->input->post())  
            Modules::run('user/changepassword');  
  
        $data['title'] = $this->session->first_name.' '.$this->session->last_name.' | Settings';  
        $data['script'] = ' <script src="'. base_url(). 'assets/js/frontdesk.js"></script>';  
  
        $this->load->view('header', $data);  
        $this->load->view('settings');  
        $this->load->view('footer');  
    }  
  
//Get Doctor by ID  
function getDoctor($doctor_id){  
    return $this->frontdesk_model->getDoctor($doctor_id);  
}  
  
//Get All Doctors  
function getDoctors(){  
    return $this->frontdesk_model->getDoctors();  
}
```

Calendar in front desk

```
div class="row">
  <div class="col-md-6">
    <h3 class="text-primary">Calendar
    <?php if($this->session->doctor_id): ?>
      of Dr. <?= $this->session->doctor_name; ?>
    <?php endif; ?>
  </h3>
</div>
</div>
<hr/>
<?php if($this->session->patient_id): ?>
  <div class="alert alert-info text-center">
    Select an Appointment Time:<br>
    Patient: <strong><?= $this->session->patient_name; ?></strong><br>
    Doctor: <strong><?= $this->session->doctor_name; ?></strong>
    <a href="#" class="close" id="unsetPatient" >&times;</a>
  </div>
<?php endif; ?>
<?php
  echo $this->session->flashdata('alert');
?>
<div class="row">
  <div class="col-lg-12">
```

```
<div class="box">

    <div id="calendar_content" class="body">

        <div id='calendar'></div>

    </div>

</div>

</div>

<div class="modal fade">

    <div class="modal-dialog">

        <div class="modal-content">

            <div class="modal-header">

                <button type="button" class="close" data-dismiss="modal"><span aria-
hidden="true">&times;</span><span class="sr-only">Close</span></button>

                <h4 class="modal-titl">

                    <?php if($this->session->patient_id): ?>Add an Appointment for Dr. <?= $this-
>session->doctor_name; ?> and patient <strong><?= $this->session->patient_name; ?><?php
else: ?>Add an Event <?php endif; ?></strong></h4>

                </div>

            <div class="modal-body">

                <div class="error"></div>

                <form class="form-horizontal" id="crud-form">

                    <input type="hidden" id="start">

                    <input type="hidden" id="end">

                    <div class="form-group">

                        <label class="col-md-3 control-label" for="title">Title</label>

                        <div class=" input-group col-md-8">
```

```
<input id="title" name="title" type="text" value="" class="form-control
input-md" />

<?php if($this->session->patient_id): ?>

    <small id="autoTitle"> Click to select title: "<span><?php if($this-
>session->patient_id): ?>Appointment for Dr. <?= $this->session->doctor_name; ?> and
patient <?= $this->session->patient_name; ?><?php endif; ?></span>"

    </small>

    <?php endif; ?>

</div>

</div>

<div class="form-group">

    <label class=" col-md-3 control-label" for="description">Description</label>

    <div class="input-group col-md-8">

        <textarea class="form-control" id="description"
name="description"></textarea>

    </div>

</div>

<div class="form-group">

    <label class="col-md-3 control-label" for="color">Color</label>

    <div class="input-group col-md-8">

        <input id="color" name="color" type="text" class="form-control input-md"
readonly="readonly" />

        <span class="help-block">Click to pick a color</span>

    </div>

</div>

<div class="form-group">

    <label class="col-md-3 control-label checkbox"> All Day Event</label>

    <div class="col-md-8" style="padding-top: 6px;">

        <input type="checkbox" id="allDay" value="allDay">
```

```
        </div>

        </div>

        <input id="csrfhash" type="hidden" name="csrfHash" value="<?= $this-
>security->get_csrf_hash(); ?>">

        </form>

    </div>

    <div class="modal-footer">

        <button type="button" class="btn btn-default" data-
dismiss="modal">Cancel</button>

    </div>

</div>

</div>

</div>

</div>

<style media="screen">

#autoTitle{

    cursor: pointer;

}

.alert-al {

    padding: 15px;

    margin-bottom: 20px;

    border: 1px solid transparent;

    border-radius: 4px;

}

.fc-time-grid-container {

    display: table;

}
```

```
</style>
```

### Doctor controller

```
function __construct(){

    parent::__construct();

    if(!$this->session->logged_in || $this->session->user_type != 'doctor')
    {
        redirect('login/doctor');
    }
    $this->form_validation->set_error_delimiters('<div class="text-center form_errors ">',
'</div>');
    $this->load->model('doctor_model');

}

public function index()
{
    $data['title'] = 'Dashboard | Doctor';
    $data['script'] = ' <script src="'. base_url(). 'assets/js/doctor.js"></script>';
    $this->load->view('header', $data);
    $this->load->view('doctor');
    $this->load->view('footer');
}

public function settings()
{

    if($this->input->post())
    Modules::run('user/changepassword');

    $data['title'] = $this->session->first_name.' '.$this->session->last_name.' | Settings';
    $data['script'] = ' <script src="'. base_url(). 'assets/js/doctor.js"></script>';

    $this->load->view('header', $data);
    $this->load->view('settings');
    $this->load->view('footer');
}

//Get Doctor by ID
function getDoctor($doctor_id){
    return $this->doctor_model->getDoctor($doctor_id);
}

//Get All Doctors
function getDoctors(){
    return $this->doctor_model->getDoctors();
}

function get_symptoms($sids){
    $sids = json_decode($sids);
    return $this->doctor_model->get_symptoms($sids);
}
```

```
function get_diseases($ids){  
    $ids = json_decode($ids);  
    return $this->doctor_model->get_diseases($ids);  
}  
  
function get_medicines($ids){  
    $ids = json_decode($ids);  
    return $this->doctor_model->get_medicines($ids);  
}
```



## AppendixG – Client Certification

BIT Degree,  
University of Colombo,  
School of Computing.

Dear Sir,

**Acknowledgment of Handling over the "Automata" Health Care Management System.**

This is to inform that I acknowledge the aforesaid system is as per my requirements defined. Also I hereby certify the quality is as per my required standards.

Yours Faithfully,

\_\_\_\_\_  
Dr Karunaratne,

Dr Karunaratne Medical Center,

**DR. B. KARUNARATNE**  
M.B.B.S. D.M. (Col) D.T.C.D. (Col) M.C.G.P. (SL)  
CMC. Reg. No. 7788  
240A, Kaduwela Road,  
Koswatta, Talangama.

# USER EVALUATION FORM FOR “AUTOMATA” HEALTH CARE MANAGEMENT SYSTEM

**Tester Name** : Dr Karunaratne

**Tester Role** : Managing Director

Please mark whether you are satisfied with the following modules in the system implementation.

No	Module	Satisfied
01	Patient Registration Module	✓
02	Past Patient Renewal Registration	✓
03	Medical data Recorder	✓
04	Patient details analyzer	✓
05	Communication module	✓

Signature-.....

Date... 4/9/15.....

**Dr. B. KARUNARATNE**  
M.B.B.S. DFM (Col) DTCD (Col) M.C.G.P. (SL)  
CMC. Reg. No. 7788  
240A, Kaduwela Road,  
Koswatta, Talangama.

# Glossary

**Activity diagrams** - graphical representations of workflows of stepwise activities and actions with support

**Administration**- the performance or management of business operations

**Agile** - software development methods based on iterative and incremental development,

**Ajax (Asynchronous JavaScript and XML)** - is a group of interrelated web development methods used on the client-side to create interactive web applications.

**Architecture** - the structure or design of anything

**Apache**- is an open source web server. Mostly for Unix, Linux and Solaris platforms.

**Black box testing** - is a method of software testing that examines the functionality of an application

**Class Diagram** - is a type of static structure diagram that describes the structure of a system

**Database**- is an organized collection of data for one or more purposes, usually in digital form.

**Framework** - is an abstraction in which software providing generic functionality can be selectively changed.

**Functional requirements** - defines what the system must do.

**Integration** - is the movement of minority groups such as ethnic minorities

**JQuery** - is a cross-browser JavaScript library designed to simplify the client-side scripting of HTML.

**Module**– a separable component

**Security** - is the degree of resistance to, or protection from, harm.

**Server** - is a system (software and suitable computer hardware) that responds to requests across a computer network

**Testing** - is an investigation conducted to provide stakeholders with information about the quality of the product or service under test

**Test Case** - is a set of conditions or variables under which a tester will determine whether an application.

**Unit testing** - is a method by which individual units of source code, sets of one or more computer program modules together.

# Index

<b>A</b>		<b>M</b>	
Activity Diagram .....	18	<b>Module</b> .....	25, 29, 53
<b>Administration</b> .....	53	HMVC.....	ii, 25, 36
Apache .....	ii, 23, 24, 53	MySQL .....	ii, 12, 23, 24, 36, 37
<b>C</b>		<b>P</b>	
Class Diagram.....	19	PHP .....	ii, 12, 23, 24, 29, 36, 37
<b>D</b>		<b>S</b>	
<b>Database</b> .....	9, 19, 24, 27, 37, 53	<b>Security</b> .....	53
<b>F</b>		<b>T</b>	
<b>Framework</b> .....	53	<b>Test Case</b> .....	53
<b>Functional requirements</b> .....	53	<b>U</b>	
<b>I</b>		unit testing.....	ii
integrated testing.....	ii	use case diagram .....	15
<b>J</b>			
<b>JQuery</b> .....	24, 53		