Clinic Management System

By

Cheah Ming Lay

SCHOOL OF ARTS AND SCIENCE
TUNKU ABDUL RAHMAN COLLEGE
KUALA LUMPUR

ACADEMIC YEAR
2010/2011
Clinic Management System

By

Cheah Ming Lay

Supervisor: Mr. Andrew Tan Khin Huat

A project report submitted to the School of Arts and Science
in partial fulfillment of the requirement for the
Bachelor of Science, Campbell University, U.S.A,
And
Advanced Diploma in Science.

Division of Internet Technology
School of Arts and Science
Tunku Abdul Rahman College
Kuala Lumpur

2010/2011

Copyright 2010 by Tunku Abdul Rahman College.
All rights served. No part of this project documentation may be reproduced,
stored in retrieval system, or transmitted in any form or by any means without
prior permission of Tunku Abdul Rahman College.
Declaration

The project submitted herewith is a result of my own efforts in totality and in every aspects of the project works. All information that has been obtained from other sources had been fully acknowledged. I understand that any plagiarism, cheating or collusion or any sorts constitutes a breach of College rules and regulations and would be subjected to disciplinary actions.

__________________________
Student’s Name : Cheah Ming Lay

Programme : Advanced Diploma in Internet Technology
Abstract

Purpose:
Clinic Management System which calls CMS in short is a management system which specially designed for most of the general clinic for keeps track their daily clinic operation information. The main purpose of doing this project is in partial fulfilment of the requirement for Bachelor of Science. Besides that, we will get to know in detail regarding how to complete a project in the real working environment by following the standard methodology during development of a system such as prepare documentation, system development and system testing.

Scope:
This Clinic Management System is an offline system that use by the clinic staff and doctor. The system cover all the basic modules include staff maintenance module, clinic information maintenance module, patient profile maintenance module, patient appointment module, patient queue module, patient visit module, inventory manager module, and report module.

Methodology:
This project is implementing by using three-tier architecture which easy in future maintenance and to protect the clinic data in a secure way. The tool that uses to develop this system is Microsoft Visual Studio 2010 and for the database storing is Microsoft Office Access 2007. The language used is VB.NET.

Assessment criteria used:
The assessment areas that this system has undergone are this system is user friendliness and it’s brings really convenience to the end user by eliminate all the paper work that suppose to be in a clinic. In fact this system can absolutely doing clinic’s activities in an easy way by reducing the paper work and faster the processing time of each activity because it is designed base on the user requirement.
Abstract

**Development phases:**

This project is following the SDLC phase which include the planning, analyse, design, testing and implementation. During this development phase, the documents that required by each phase are Proposal of the system, System Requirement Specification, System Development Diagram (such as Use Case Diagram, Sequence Diagram and etc), test plan and etc.

**Results:**

This Clinic Management System helps the clinic to store their confidential data especially patient data in a very secure way and easy to keep track in future when they want to use it. Furthermore, only certain authority user could access to the confidential data in order to protect the patient privacy. As a result, it helps the clinic daily operation goes on effectiveness and efficiency with the aid of the great features.

**Conclusions and Recommendations:**

During the development of this project, I had gained extra knowledge out of classroom and experienced on how to develop a complete system as I am doing this project individually. To develop a very useful system is not as easy as we think as there is no perfect system in this world. All we need to do is go through many times of the phase to figure out which part is not logical and needed to modify in order to complete the system. After complete this project, my skill in planning for a project and skill for programming as well as skill in debugging for error had improved. Besides that, I had learned additional knowledge about the programming language VB.NET from online resources.
Acknowledgement

I wish to express my sincere gratitude to my project supervisor, Mr Andrew Tan Khin Huat, for his guidance, constant support and encouragement throughout the completion of this final year project. I would like to convey my appreciation to Mr Andrew for every precious advice that he gave during the system preview of the final year project. Furthermore, I would like to thank to Mr Wong Yoke Seng for contributing his ideas and in-depth knowledge in the field.

A warm thank is extended to my classmate for kindly sharing out their resources, opinions, knowledge, experience and skills in programming and development methodology with me, so generously. Besides that, I would also want to personally express my thankfulness to my fellow friends from other course in TAR College who always there to provide me with the best solution to struggle against the problem before I can complete this final year project.

Last, but not least, I wishes to acknowledge the unwavering support shown by my family members who always there for me to give me endless support when I needed them.
# Table of Contents

Declaration ........................................................................................................................................ iii

Abstract .......................................................................................................................................... iv

Acknowledgement ........................................................................................................................... vi

1. **System Planning** .................................................................................................................. 2
   1.1 Project Aims and Objectives ................................................................................................. 2
   1.2 Company Background ........................................................................................................... 4
   1.3 Project Scope ....................................................................................................................... 5
   1.4 Project Schedule .................................................................................................................. 8
   1.5 Outline of approach/Methodology Used ............................................................................... 9
   1.6 Linkage to Seminar .............................................................................................................. 10

2. **Requirements Analysis** ................................................................................................... 11
   2.1 Fact Gathering .................................................................................................................... 12
      2.1.1 Observation ................................................................................................................ 12
      2.1.2 Interview ..................................................................................................................... 12
   2.2 Fact Recording .................................................................................................................. 13
      2.2.1 Functional Requirement .............................................................................................. 13
      2.2.2 Non-Functional Requirement ..................................................................................... 15
   2.3 Hardware and Software Requirement for Development ................................................. 16
      2.3.1 Software Requirement ............................................................................................... 16
      2.3.2 Hardware Requirement ............................................................................................... 16
   2.4 Hardware and Software Requirement for Operational ..................................................... 17
      2.4.1 Software Requirement ............................................................................................... 17
      2.4.2 Hardware Requirement ............................................................................................... 17
   2.5 System Architecture Diagram ............................................................................................ 18
   2.6 Use Case Diagram and Use Case Description ..................................................................... 19
   2.7 Activity Diagram ............................................................................................................... 35
   2.8 Class Diagram ................................................................................................................... 44

3. **System Design** ................................................................................................................ 45
   3.1 Database Design ................................................................................................................. 46
      3.1.1 Data Dictionary .......................................................................................................... 46
      3.1.2 Normalization ............................................................................................................. 50
   3.2 Sequence Diagram ............................................................................................................ 53
   3.3 Collaboration Diagram ....................................................................................................... 60
   3.4 User Interface Design ......................................................................................................... 67

4. **Programming** ................................................................................................................ 92
   4.1 Coding .................................................................................................................................. 93

5. **Software Testing** .......................................................................................................... 107
   5.1 Testing Technique .............................................................................................................. 108
      5.1.1 Unit Testing ................................................................................................................. 108
      5.1.2 Validation Testing ....................................................................................................... 108
      5.1.3 User Acceptance Testing ........................................................................................... 109
   5.2 Test Plan ............................................................................................................................ 110
5.3 Validation and Verification ........................................................................................................ 121

6. Conclusion ..................................................................................................................................... 149
   6.1 Evaluation against Project ........................................................................................................ 147
      6.1.1 Project Strength ............................................................................................................... 147
      6.1.2 Project Weaknesses ........................................................................................................ 148
   6.2 Personal Reflection .................................................................................................................. 148

7. References ..................................................................................................................................... 149

8. Appendices .................................................................................................................................... 15250

9. User Guide .................................................................................................................................... 151
Chapter 1

System Planning
1. **System Planning**

1.1 **Project Aims and Objectives**

The aim and objective of this project is to fully automate the clinic daily operation by leading the clinic to operate in effective and efficient way throughout the help of the system so that they could eliminate the paper work that originally happened in a clinic. The following is the benefits that come with the clinic management system:

- **Shrink the clinic needed space on keeping paper**
  
  Ordinary clinic will probably record their patient information in a paper and as the patient visit often the clinic nurse will add in paper to that patient record in order to let the doctor have enough space to write down the diagnosis to that patient. The system provides the ability to store the patient information into database so that the data is stored in a more organized manner and do not wasted any single space in a clinic. Furthermore, the database supports the clinic to store up to millions of patient record in an easy managerial way.

- **Immediate response on searching data**
  
  The end user can search for the patient records easily by only input some criteria such as the patient ID or the patient name instead of search the patient record from a hundred or thousand of patient record from the cabinet. The system will immediately respond to the user with the patient information base on the criteria and it definitely can save up a lot of time and work on searching.

- **Save the communication time**
  
  As all the staff will sharing the same database among the clinic, so the nurse do not need to run over here and there to pass the patient record to the doctor or get back the patient record from doctor. All the staff inside the clinic could get up to date information from the sharing database.
• **Increase the privacy level**

Patient privacy is important to a clinic as patient will only trusted your clinic and come often if you only can keep their previous visited history well and confidentially. Clinic management system provides a secure way towards the patient information by restrict the access right to authority user only. Only the authorized user such as doctor could access to the patient medical history. Even the nurse at the counter can only view the basic information of the patient.

• **Standardize documentation**

The documentation such as Medical Certificate (MC) and payment receipt is well prepared in a standard format. The nurse can use it at anytime by just print it out instead of writing their own by hand.
1.2 Company Background

Lim Huat Bee Clinic was established in 1990 and now offers nearly 11 years experience in the field of treating patient. This clinic is in charge by the creator of the clinic Dr. Lim Huat Bee and he is the only doctor in the clinic with the aid of help by three clinic staff.

The clinic operates 6 days per week from 9am to 8pm. No matter if you are a resident of the area, a vacationer or an employer, the clinic do provide you with the best medical care possible. The clinic offer qualified, well trained staff or physician assistant to serve you. The clinic allow patient to walk-in anytime or call to schedule an appointment. For schedule the appointment, the patient must be the regular patient from the clinic.
1.3 Project Scope

This system will provide clinic a high efficiency management tools, computerize and systematic patients record, inventory record and appointment detail. It basically will do all the daily activity as a clinic will do. The following is the sub module and its description:

- **Staff Maintenance Module**

  This module basically will record all the clinic staff basic information. Once the staff is registered, he/she is allowed to login to the system. The authority for access right of the user is separate into three types which are admin, doctor and staff. All level user can perform add and edit staff record and change their login password. Only admin level user can delete the staff record and reset the staff login password if the staff had forgotten the login password.

- **Inventory Maintenance Module**

  This module contains all the medicine record in a clinic and each of the medicine inventory level so that the user could order the medicine stock once the medicine quantity level is found low to avoid lack of stock problem occur. Furthermore, it also store the medicine function, price and supplier contact details.
• **Clinic Information Maintenance Module**

This module allow the user to manage the clinic information such as the clinic name that wanted to print on all the report, the clinic contact number, address and the operation hours. The operation hours that set by the user will decide the appointment hours at the patient appointment module.

• **Report Module**

This module prepared several type of report that useful to a clinic which includes patient card, patient Medical Certificate (MC), patient dispensary report, and patient payment receipt. Staff will print out a patient card for the new registered patient so that the patient could use the card for the following visit. The patient dispensary report will list out all the medicine and treatment that prescript by the doctor. After the patient consult the doctor, the staff will collect the medicine for the patient base on the patient dispensary report. The patient payment receipt is a report that list out all the price of the medicine and treatment that the patient had took. For the patient MC, the staff will generate it when only its request by the patient.

• **Patient Profile Maintenance Module**

This module allowed staff and doctor to add and edit the patient profile which consist only very basic information about the patient without any of the patient symptom information. For the admin level user, they can perform delete patient record action.

• **Patient Appointment Scheduling Module**

This module allowed the staff to add, edit and delete the appointment that make by the patient. The time slot between each appointment is half an hour. The staff is only allowed to add in new appointment on blank slot. The doctor can check back the appointment that made by the patient by using date to filter the patient appointment.
- **Patient Queue Module**

  This module was act like an intermediary between doctor and staff. When there is a patient come for consulting doctor, the staff will add the patient into the queue list so that the doctor can select which patient to be consulted from the queue list. There is a screen for calling patient purpose that will showing out the patient name when there is their turn to consult doctor or taking medicine.

- **Patient Visit History Module**

  This module is only restricted to doctor use for record down the patient symptom, test conducted to the patient, diagnosis and dispensary to the patient. The doctor will create a new visit record every time the patient coming for consulting. After done the diagnosis, the doctor will add the medicine and note down the instruction for how to take the medicine. The system will then calculate the total price of all the medicine and treatment that took by the patient.
1.4 Project Schedule

<table>
<thead>
<tr>
<th>Task Name</th>
<th>2010</th>
<th>2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. System Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 Research</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2 Project Proposal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3 Project Scope</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4 Project Specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5 Project Schedule</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. System Analysis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1 Fact Finding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2 Feasibility Study</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3 Draft User Requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4 Hardware and Software Requirement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5 Create UML Diagram</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. System Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1 Database Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2 Class Diagram</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3 Develop Functional Specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4 Prototype Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. System Implementation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1 Create Database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2 Coding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3 System Install</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4 System Testing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Documentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1 System Documentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.2 User Guide</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1.5 Outline of approach/Methodology Used

The methodology that uses to develop this clinic management system is:

- **VB.NET with three-tier architecture**

  The programming language that I used to develop this system is VB.NET with the three-tier architecture. 3-tier architecture is the new growth area for client/server environment because it meets the requirement of large-scale Internet and intranet client/server application. In theory, 3-tier systems are more scalable, robust, and flexible. In addition, they can integrate data from multiple sources. 3-tier applications are easier to manage and deploy on the network --- most of the code runs on the servers. Also, 3-tier applications minimize network interchanges by creating abstract levels of service. Instead of interacting with the database directly, the client calls application logic on the server. The application logic then accesses the database on the client’s behalf. 3-tier substitutes a few server calls for many database queries and updates, so it performs much better than 2-tier. [Robert Orfali, Dan Harkey, and Jeri Edwards, 1999]

- **Microsoft Office Access 2007**

  I am doing an offline system and do not contributed in any sharing data throughout online, so I decided to use Microsoft Office Access as the database to store data for the system. Microsoft Access is a relational database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. I’m currently developing the system for a small clinic only, so it is not necessary to use very powerful database management system like mySql or SQL server. [Wikipedia, 2011]
1.6 Linkage to Seminar

A bar code is a graphic representation of data that is machine-readable which is conducted by using alpha, numeric or both of it. They may be thought of as another way of writing, because they replace key data entry as a method of gathering data. In business, use of bar codes can reduce inefficiencies and improve a company’s productivity. Therefore, bar codes are an easy, fast and accurate way of data entry. A bar code basically does not contain descriptive data yet it is a reference number that is readable by a device which is called bar code scanner to use to search for an associated record that contains descriptive data and other private information. [Russell F. Lewis and Ned Simpson, 2003]

My seminar title is “Barcode Technology” which do research on barcode technology include research on several type of barcodes. Base on the research on all type of barcodes, Code 128 would be the most suitable barcode symbology to apply in my Clinic Management System because Code 128 is recommended to be used in medical industry because Code 128 is having the low error rate and it is critical in care giving environment.
Chapter 2

Requirements Analysis
2. Requirements Analysis

2.1 Fact Gathering

2.1.1 Observation

Before I can design a system for the clinic, I do visit to the clinic to do an observation on how is their daily business operation. It is an ordinary clinic which done everything manually without any technology help. I do observed and record down in detail how is the progress flow from patient register on the counter until the patient take medicine on the dispensary counter. Through the progress, I can see that there’s a lot of paper needed to use in order to complete this progress. I did question on them when I found out some problem that I do not truly understand on it and the staff was kindly answer every question and explain to me in detail. Furthermore, the staff had shown me several type of document that they usually do for reference purpose. Most of the document is done manually in paper by the counter staff includes calculation on stock, generate receipt, generate MC and etc.

2.1.2 Interview

End user is the one who are going to have the interaction with the system, thus interview the end user is one of the most effective ways on gathering facts. Throughout the interview session I can get the exact information and definite requirement that the end user expected on the system. Doctor and staff requirement are different as the job that they carry out is different from each other. However, the current business model of the clinic is having weak privacy on patient medical history. So when I come to the design stage I must consider the privacy of the patient medical history issue.
2.2 Fact Recording

2.2.1 Functional Requirement

Login Module

The system only can access by the authority user which is an official clinic staff who is registered. The authority user must use username and password to login to the system. Validation on username and password that input is required to deny the invalid user login to the system.

Staff Maintenance Module

Basically this module is required to store the clinic staff and doctor information includes the username and password for login purpose. Staff is allowed to do all the basic maintenance sort like add new staff, update their profile, search their record through staff ID and change their login password as well.

Inventory Module

System should store the medicine information include the function of the medicine, price, quantity on hand and the supplier contact information. All the basic function such as add new medicine, update medicine information and remove the medicine from database is required to carried out by the staff.

Patient Record Module

Patient medical history is a confidential data which suppose to view by doctor only. So only doctor can add and update the patient medical history. Staff level user could only add new patient record and update patient record with patient basic information such as contact detail.
Patient Appointment Module

Staff user is the one who dealing with the patient appointment module. When the patient is calling for booking appointment with the doctor, the staff will provide patient with the doctor empty slot to let the patient choose. Or if the patient offer’s time is already booked by others the staff will suggest the patient to select other time slot. The appointment module is allowed the staff to add new appointment, edit appointment if the patient wishes to change the time and delete the appointment if the patient cancels the appointment.

Report Module

The basic report that usually needed to be carried out during business operation is required to prepare for the staff. The detail that going to display on the report must decided by the user.
2.2.2 Non-Functional Requirement

Usability Requirement
- The system will decrease the amount of text box that input by the user and increase the selection input method such as combo box and radio button. Within this will eliminate the possibility of entry error that made by the user when they enter record into the system.
- Besides that, the user is definitely not an IT field people. Thus, the system should very straight forward to the user and easy to operate by the user. The error message that shown when error detected must in the understandable language to the user.

Efficiency Requirement
- The system must allow the staff to search the patient record in an easy and efficient way from a large amount of data.
- The system response time must be fast and the system should allow the user to open several task at one time when they using it.

Reliability Requirement
- Make sure there is an additional server to backup the clinic data in case when the server is down, there is still a backup server to support the system to continue running the daily business.
- The error rate of this system must be shrinking. Furthermore, adequacy error message should prompt to the user when there is any validation error occurred.

Security Requirement
- The system must provide a highly security on protecting the patient privacy.
- Some confidential data should restrict to only authorize user to access it.
2.3 Hardware and Software Requirement for Development

2.3.1 Software Requirement

- Microsoft Visual Studio 2010 for developing the system with programming VB.NET
- Microsoft Office Access 2007 for developing the database for the system
- Microsoft Office Visio 2007 for drawing the diagram such as ERD diagram, Activity Diagram and etc.
- Adobe Photoshop CS4 for designing the graphic that applies in the system interface.

2.3.2 Hardware Requirement

- Notebook for developing the system and viewing the output of the system. The following is my notebook specification:
  - Processor – Intel® Core™2 Duo CPU T6400@2.00GHz
  - Memory(RAM) – 4.00GB
  - System type – 32-bit Operating System
- Basic input devices for writing code, input documentation, and testing the system include:
  - Optical mouse, and
  - Keyboard
  - Barcode Scanner
- Output devices for testing system purpose are:
  - Printer for printing the documentation and system report purpose.
  - 22” LCD Display for testing some module purpose.
- Wi-Fi Modem for connecting to the internet for looking some resources.
- External hard disk and thumb drive for backup the system and all the documentation.
2.4 Hardware and Software Requirement for Operational

2.4.1 Software Requirement

- Window XP Professional or Vista to support a better performance for the system.
- Microsoft Office Access 2007 to support to view the database.

2.4.2 Hardware Requirement

- Several basic personal computers which at least run Window XP.
- A server which store the clinic database and link to all the personal computer within the clinic.
- All the basic input devices include mouse, and keyboard.
- Barcode scanner for reading barcode purpose.
- Output devices such as printer and LCD screen for displaying the system output and printing the report.
- Modem
2.5 System Architecture Diagram

System architecture of the clinic
2.6 Use Case Diagram and Use Case Description

Overview Use Case Diagram for Clinic Management System
Use Case Description for Clinic Information Maintenance

Use Case Name: Clinic Information Maintenance

Brief Description: This use case allows user to view and edit the clinic information.

Actor: Admin, Doctor and Staff

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login to the system.</td>
<td>2. Verify User ID and Login Password.</td>
</tr>
<tr>
<td>3. Click Clinic Information Button.</td>
<td>4. Display Clinic Information Page.</td>
</tr>
<tr>
<td>5. Edit Clinic Information</td>
<td>7. Verify all data and update to database.</td>
</tr>
<tr>
<td>6. Click on Save button to update the information.</td>
<td></td>
</tr>
</tbody>
</table>

Alternative Flows:

A-1 Step 2: Invalid User ID or Password will prompt error message.

A-2 Step 7: Invalid data format will prompt error message.
**Detail Use Case Diagram for Staff Profile Maintenance Module**

![Staff Profile Maintenance Use Case Diagram]

**Use Case Description for Staff Profile Maintenance**

**Use Case Name:** Staff Profile Maintenance

**Brief Description:** This use case allows the user to add, view, edit, delete, change login password and reset login password.

**Actor:** Admin, Doctor and Staff

**Main Flow:**

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>3. Click Staff Maintenance Button.</td>
<td>4. Display Staff Maintenance Page.</td>
</tr>
<tr>
<td>5. Input particular Staff ID.</td>
<td>6. Display Staff Record base on the criteria.</td>
</tr>
<tr>
<td>7. Double click on particular record to edit.</td>
<td>8. Display selected staff detail in particular field.</td>
</tr>
<tr>
<td>10. Click Save Button.</td>
<td></td>
</tr>
</tbody>
</table>
12. Click Add New button.
14. Key in new staff info.
15. Click Add Button.
17. Click Delete Button.
19. Click Change Password Button.
21. Input the old password and new password.
22. Click Save Button.
24. Click Reset Password Button.

<table>
<thead>
<tr>
<th>13. Enable all blank textbox.</th>
<th>16. Verify all data and add to database.</th>
</tr>
</thead>
</table>

Alternative Flows:

A-1 Step 2: Invalid User ID or Password will prompt error message.
A-2 Step 11: Invalid data format will prompt error message.
A-3 Step 16: Invalid data key in will prompt error message.
A-4 Step 23: Invalid old password entered will prompt error message.
Use Case Description for Inventory Maintenance

Use Case Name: Inventory Maintenance

Brief Description: This use case allows the user to add, view, edit, and delete inventory items.

Actor: Admin, Doctor and Staff

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>3. Click Inventory Maintenance Button.</td>
<td>4. Display Inventory Maintenance Page.</td>
</tr>
<tr>
<td>5. Input particular Item ID.</td>
<td>6. Display Item Record based on the criteria.</td>
</tr>
<tr>
<td>7. Double click on particular record to edit.</td>
<td>8. Display selected item detail in particular field.</td>
</tr>
<tr>
<td>10. Click Save Button.</td>
<td>13. Enable all blank textbox.</td>
</tr>
<tr>
<td>12. Click Add New button.</td>
<td></td>
</tr>
<tr>
<td>14. Key in new item info.</td>
<td></td>
</tr>
<tr>
<td>15. Click Add Button.</td>
<td>16. Verify all data and add to database.</td>
</tr>
<tr>
<td>17. Click Delete Button.</td>
<td>18. Remove particular item record from database.</td>
</tr>
</tbody>
</table>

Alternative Flows:

A-1 Step 2: Invalid User ID or Password will prompt error message.

A-2 Step 11: Invalid data format will prompt error message.

A-3 Step 16: Invalid data key in will prompt error message.
**Use Case Description for Patient Profile Maintenance**

Use Case Name: Patient Profile Maintenance

Brief Description: This use case allows the user to add, view, edit, and delete the patient record. Besides that, it also can generate patient member card, print dispensary report and payment receipt.

Actor: Admin, Doctor and Staff

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>7. Double click on particular record to edit.</td>
<td>8. Display selected patient detail in particular field.</td>
</tr>
<tr>
<td>10. Click Save Button.</td>
<td></td>
</tr>
</tbody>
</table>
12. Click Add New button.
15. Click Add Button.
17. Click Delete Button.
19. Click Print Patient Member Card Button.
21. Click Print Dispensary Report Button.
23. Click Print Payment Receipt Button.

<table>
<thead>
<tr>
<th>13. Enable all blank textbox.</th>
</tr>
</thead>
<tbody>
<tr>
<td>16. Verify all data and add to database.</td>
</tr>
<tr>
<td>18. Remove particular patient record from database.</td>
</tr>
<tr>
<td>20. Display Patient Member Card with particular patient data.</td>
</tr>
<tr>
<td>22. Display particular patient’s dispensary report.</td>
</tr>
<tr>
<td>24. Display particular record on the payment receipt.</td>
</tr>
</tbody>
</table>

**Alternative Flows:**

A-1 Step 2: Invalid User ID or Password will prompt error message.

A-2 Step 11: Invalid data format will prompt error message.

A-3 Step 16: Invalid data key in will prompt error message.
**Detail Use Case Diagram for Patient Appointment Module**

![Diagram showing use case diagram for patient appointment module]

**Use Case Description for Patient Appointment Maintenance**

Use Case Name: Patient Appointment Maintenance

Brief Description: This use case allows the user to add, view, edit, and delete the patient appointment with the doctor.

Actor: Admin, Doctor and Staff

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>5. Select a date.</td>
<td>6. Display the appointment base on the date selected.</td>
</tr>
<tr>
<td>7. Double click on particular appointment record to edit.</td>
<td>8. Display selected appointment in particular field.</td>
</tr>
<tr>
<td>10. Click Save Button.</td>
<td>13. Enable all blank textbox.</td>
</tr>
<tr>
<td>12. Double click on blank column to add</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>15.</td>
<td>Click Add Button.</td>
</tr>
<tr>
<td>17.</td>
<td>Click Delete Button.</td>
</tr>
<tr>
<td>16.</td>
<td>Verify all data and add to database.</td>
</tr>
<tr>
<td>18.</td>
<td>Remove particular appointment record from database.</td>
</tr>
</tbody>
</table>

Alternative Flows:

- **A-1 Step 2**: Invalid User ID or Password will prompt error message.
- **A-2 Step 11**: Invalid data format will prompt error message.
- **A-3 Step 16**: Invalid data key in will prompt error message.
**Detail Use Case Diagram for Patient Queue List Maintenance Module**

![Use Case Diagram for Patient Queue List Maintenance Module](image)

**Use Case Description for Patient Queue List Maintenance**

Use Case Name: Patient Queue List Maintenance

Brief Description: This use case allows the user to add, view, edit, delete and clear all the queue list record. Besides that, it also allowed to print dispensary report, payment receipt and medical certificate.

Actor: Admin, Doctor and Staff

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>5. Select particular record from queue list to edit.</td>
<td>6. Display selected patient queue detail in particular field.</td>
</tr>
<tr>
<td>7. Edit info.</td>
<td></td>
</tr>
<tr>
<td>8. Click Save Button.</td>
<td>9. Verify all data and update database.</td>
</tr>
</tbody>
</table>
10. Click Add New button.
12. Key in new patient queue information.
13. Click Add Button.
15. Click Delete Button.
17. Click Clear All Button.
19. Click Print Dispensary Report Button.
21. Click Print Payment Receipt Button.
23. Click Print Patient MC Button.
25. Insert the particular field that required.

11. Enable all blank textbox.
14. Verify all data and add to database.
16. Remove particular patient queue record from database.
18. Remove all the queue list record from database.
20. Display particular patient’s dispensary report.
22. Display particular record on the payment receipt.
24. Display print patient MC page.
26. Verify all data and print the MC.

Alternative Flows:

A-1 Step 2: Invalid User ID or Password will prompt error message.
A-2 Step 9: Invalid data format will prompt error message.
A-3 Step 14: Invalid data key in will prompt error message.
A-4 Step 26: Invalid data key in will prompt error message.
Use Case Description for Patient Visit History Maintenance

Use Case Name: Patient Visit History Maintenance

Brief Description: This use case allows the user to add, view, edit, and delete the patient visit record. Besides that, it also allows the user to add and edit the prescription detail.

Actor: Admin and Doctor

Main Flow:

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>3. Click Visit History Tab in Patient Profile.</td>
<td>4. Display All Patient Visit History Record.</td>
</tr>
<tr>
<td>5. Select particular visit record.</td>
<td>6. Display selected visit record detail in particular filed.</td>
</tr>
<tr>
<td>8. Click Save Visit Record Button.</td>
<td>11. Display selected prescription record detail in particular filed.</td>
</tr>
<tr>
<td>10. Select a prescription record from the</td>
<td></td>
</tr>
</tbody>
</table>

Detail Use Case Diagram for Patient Visit History Module
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.</td>
<td>Edit record information.</td>
</tr>
<tr>
<td>13.</td>
<td>Click Save Prescription Button.</td>
</tr>
<tr>
<td>15.</td>
<td>Click Delete Prescription Button.</td>
</tr>
<tr>
<td>17.</td>
<td>Click Add New Visit Button.</td>
</tr>
<tr>
<td>19.</td>
<td>Key in new visit record information.</td>
</tr>
<tr>
<td>20.</td>
<td>Click Add Prescription Button.</td>
</tr>
<tr>
<td>22.</td>
<td>Key in new prescription detail.</td>
</tr>
<tr>
<td>23.</td>
<td>Click Add Prescription Button.</td>
</tr>
<tr>
<td>25.</td>
<td>Click Add Visit Button.</td>
</tr>
<tr>
<td>27.</td>
<td>Click Delete Visit Record Button.</td>
</tr>
</tbody>
</table>

Alternative Flows:

A-1 Step 2: Invalid User ID or Password will prompt error message.
A-2 Step 9: Invalid data format will prompt error message.
A-3 Step 14: Invalid data format will prompt error message.
A-4 Step 24: Invalid data key in will prompt error message.
A-5 Step 26: Invalid data key in will prompt error message.
**Detail Use Case Diagram for Report Generation Module**

![Diagram](image)

**Use Case Description for Report Generation**

**Use Case Name:** Report Generation

**Brief Description:** This use case allows the user to print patient card, patient dispensary report, patient payment receipt and patient medical certificate.

**Actor:** Admin, Doctor and Staff

**Main Flow:**

<table>
<thead>
<tr>
<th>Actor Action</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User login into system.</td>
<td>2. Verify User ID and Password.</td>
</tr>
<tr>
<td>3. Click Print Patient Member Card.</td>
<td>4. Display all the patient member cards on the report.</td>
</tr>
<tr>
<td>7. Select particular record to print on the report.</td>
<td>8. Display selected record on patient dispensary report.</td>
</tr>
<tr>
<td>9. Click Print Payment Receipt Button.</td>
<td>10. Display print payment receipt report page.</td>
</tr>
<tr>
<td>11. Select particular record to print on the report.</td>
<td>12. Display particular payment receipt</td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>13.</td>
<td>Click Print Patient MC Button.</td>
</tr>
<tr>
<td>15.</td>
<td>Select a patient and insert the particular field that required.</td>
</tr>
<tr>
<td>16.</td>
<td>Verify all data and print the MC on report.</td>
</tr>
</tbody>
</table>
2.7 Activity Diagram

Activity Diagram for Clinic Information Maintenance

Start

Admin/Doctor/Staff

Login

Display login form

Enter user ID and password

Verify user ID and password

Invalid

Valid

Prompt error message

Display main menu

Select Clinic Info Maintenance

Display Clinic Info Page

Edit Clinic Information

Verify clinic detail

Invalid

Valid

Prompt error message

Prompt successful message

Press Save button

End
Activity Diagram for Staff Profile Maintenance

Start

Admin/Doctor/Staff

- Login
- Enter user ID and password
- Select staff profile
- Search staff record by staff ID

System

- Display login form
- Verify user ID and password
- Display main menu
- Display registration form with generated staff ID
- Verify all data format that input
- Verify whether the staff ID exist or not

Flow:
- Valid ➔ Prompt error message ➔ Display main menu
- Invalid ➔ Prompt error message ➔ Display login form
- Press Add button ➔ Fill in registration form
- Yes ➔ Prompt successful message ➔ End
- No ➔ Prompt error message ➔ Display particular staff record
Activity Diagram for Inventory Maintenance

Start:

- Login
- Enter user ID and password

Admin/Doctor/Staff:

- Select inventory management
- Search item ID
- Delete item
- Press Yes button
- Edit item detail
- Modify item detail
- Press Save button

System:

- Display login form
- Verify user ID and password
- Display main menu
- Display inventory management Page
- Display add new item's form with generated Item ID
- Verify whether the item ID exist or not
- Press Add button
- Verify all input data format
- Prompt error message
- Prompt successful message
- Prompt confirmation message
- Prompt successful message
- Display edit item form
- Verify item detail
- Prompt error message
- Prompt successful message

End
Activity Diagram for Patient Profile Maintenance
Activity Diagram for Patient Appointment Scheduling Maintenance
Activity Diagram for Patient Queue List Maintenance

Start

Admin/Doctor/Staff

Login

Enter user ID and password

Verify user ID and password

System

Display login form

Valid

Prompt error message

Display main menu

Select Queue List Management

Display Queue List Page

Add patient to queue list

Fill in the detail

Press Add Button

Verify all input data format

Invalid

Valid

Prompt error message

Prompt Successful Message

Select particular record from queue list

Display particular queue record

Remove record from queue list

Prompt confirmation message

Press Yes button

Prompt successful message

Modify record

Press Save button

Verify queue detail

Invalid

Valid

Prompt error message

Prompt Successful Message

End
Activity Diagram for Patient Visit History Maintenance

1. **Start**
   - Admin/Doctor: Login
   - System: Display login form

2. **Enter user ID and password**
   - Admin/Doctor: Verify user ID and password
   - System: Display main menu

3. **Select patient visit history tab from patient profile**
   - Admin/Doctor: Display patient visit history page
   - System: Display new visit record page with generated visit ID

4. **Add new visit record**
   - Admin/Doctor: Add new prescription to visit record
   - System: Display new prescription page

5. **Add new prescription to visit record**
   - Admin/Doctor: Verify the new prescription record detail
   - System: Prompt error message
   - Valid: Prompt successful message

6. **Select a medicine and fill in the form**
   - Admin/Doctor: Press Add Prescription button
   - System: Prompt error message
   - Valid: Prompt successful message

7. **Press Add Visit button**
   - Admin/Doctor: Verify the new visit record detail
   - System: Prompt error message
   - Valid: Prompt successful message

8. **Select patient visit record to view**
   - Admin/Doctor: Display particular visit record
   - System: Prompt confirmation message
   - Prompt successful message

9. **Edit Visit record**
   - Admin/Doctor: Delete patient visit record
   - System: Display edit visit record form

10. **Modify visit record detail**
    - Admin/Doctor: Press Yes button
    - System: Verify visit record detail
    - Invalid: Prompt error message
    - Valid: Prompt successful message

11. **Press Save button**
    - Admin/Doctor: Prompt error message
    - System: Prompt successful message

End
Activity Diagram for Report Generation

Start

Admin/Doctor/Staff

Login

Enter user ID and password

Verify user ID and password

Display main menu

Prompt error message

System

Display login form

Invalid

Valid

Display all patient’s member card in report

Display print dispensary report page

Display particular record on patient dispensary report

Display print payment receipt page

Display particular patient payment receipt on report

Display print MC report page

Press Print Button

Select Print Patient Card

Select particular record

Select Print Patient Dispensary Report

Select particular record

Select Print Patient Payment Receipt

Select particular record

Select Print Patient MC report

Select particular record and fill in require info

Press Print Button

Verify info that input

Invalid

Valid

Print patient MC on the report
Chapter 2: Requirements Analysis

2.8 Class Diagram

[Diagram of class relationships and attributes for appointment, patient, personal, staff, login, queue, visit_history, prescription, inventory, supplier_detail]
Chapter 3

System Design
3. **System Design**

3.1 **Database Design**

3.1.1 Data Dictionary

**Table Name: Staff**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>staff_id</td>
<td>VARCHAR</td>
<td>20</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>staff_name</td>
<td>VARCHAR</td>
<td>50</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_ic_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_birth_date</td>
<td>DATETIME</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_address</td>
<td>VARCHAR</td>
<td>100</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_contact_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_gender</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_position</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_status</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>login_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>password</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
</tbody>
</table>

**Table Name: Patient**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient_id</td>
<td>VARCHAR</td>
<td>20</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>patient_name</td>
<td>VARCHAR</td>
<td>50</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_ic_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_gender</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_birth_date</td>
<td>DATETIME</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_marital_status</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_register_date</td>
<td>DATETIME</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_status</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_address</td>
<td>VARCHAR</td>
<td>100</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>patient_contact_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
</tbody>
</table>
### Table Name: Inventory

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>item_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>item_name</td>
<td>VARCHAR</td>
<td>100</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>item_unit</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>item_qty</td>
<td>INTEGER</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>item_price</td>
<td>DOUBLE</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>item_function</td>
<td>VARCHAR</td>
<td>200</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>supplier_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
</tbody>
</table>

### Table Name: Supplier detail

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>supplier_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>supplier_name</td>
<td>VARCHAR</td>
<td>50</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>supplier_contact_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>supplier_address</td>
<td>VARCHAR</td>
<td>100</td>
<td>Null</td>
<td></td>
</tr>
</tbody>
</table>

### Table Name: Appointment

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>appointment_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>patient_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>contact_no</td>
<td>VARCHAR</td>
<td>20</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>staff_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>appointment_date</td>
<td>DATETIME</td>
<td>-</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>appointment_time</td>
<td>VARCHAR</td>
<td>10</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>bookmark</td>
<td>VARCHAR</td>
<td>50</td>
<td>Null</td>
<td></td>
</tr>
</tbody>
</table>
### Table Name: Queue List

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>queue_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>queue_date</td>
<td>DATETIME</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>patient_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>patient_contact</td>
<td>VARCHAR</td>
<td>20</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>staff_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>queue_status</td>
<td>VARCHAR</td>
<td>10</td>
<td></td>
<td>Null</td>
</tr>
</tbody>
</table>

### Table Name: Visit History

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>visit_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>visit_date</td>
<td>DATETIME</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>patient_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>staff_id</td>
<td>VARCHAR</td>
<td>10</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>symptom</td>
<td>VARCHAR</td>
<td>200</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>test_conducted</td>
<td>VARCHAR</td>
<td>200</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>diagnosis</td>
<td>VARCHAR</td>
<td>200</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>prescription_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>total_amount</td>
<td>DOUBLE</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
</tbody>
</table>
### Table Name: Prescription

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Length</th>
<th>Key</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>prescription_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Primary Key</td>
<td>Not Null</td>
</tr>
<tr>
<td>item_id</td>
<td>VARCHAR</td>
<td>10</td>
<td>Foreign Key</td>
<td>Null</td>
</tr>
<tr>
<td>quantity</td>
<td>INTEGER</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>discount_price</td>
<td>DOUBLE</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>grand_total</td>
<td>DOUBLE</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_method</td>
<td>VARCHAR</td>
<td>10</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_qty</td>
<td>INTEGER</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_unit</td>
<td>VARCHAR</td>
<td>10</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_time_daily</td>
<td>INTEGER</td>
<td>-</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_instruction</td>
<td>VARCHAR</td>
<td>10</td>
<td></td>
<td>Null</td>
</tr>
<tr>
<td>use_explanation</td>
<td>VARCHAR</td>
<td>100</td>
<td></td>
<td>Null</td>
</tr>
</tbody>
</table>
3.1.2 Normalization

First Normalize

System Details (STAFF_ID, STAFF_NAME, STAFF_IC_NO, STAFF_BIRTH_DATE, STAFF_ADDRESS, STAFF_CONTACT_NO, STAFF_GENDER, STAFF_POSITION, STAFF_STATUS, LOGIN_ID, PASSWORD, PATIENT_ID, PATIENT_NAME, PATIENT_IC_NO, PATIENT_GENDER, PATIENT_BIRTH_DATE, PATIENT_MARITAL_STATUS, PATIENT_REGISTER_DATE, PATIENT_STATUS, PATIENT_ADDRESS, PATIENT_CONTACT_NO, APPOINTMENT_ID, APPOINTMENT_DATE, APPOINTMENT_TIME, BOOKMARK, QUEUE_ID, QUEUE_DATE, QUEUE_STATUS, VISIT_ID, VISIT_DATE, SYMPTOM, TEST_CONDUCTED, DIAGNOSIS, PRESCRIPTION_ID, ITEM_ID, ITEM_NAME, ITEM_UNIT, ITEM_QTY, ITEM_PRICE, ITEM_FUNCTION, SUPPLIER_ID, SUPPLIER_NAME, SUPPLIER_CONTACT_NO, SUPPLIER_ADDRESS, TOTAL_AMOUNT, QUANTITY, DISCOUNT_PRICE, GRAND_TOTAL, USE_METHOD, USE_QTY, USE_UNIT, USE_TIME_DAILY, USE_INSTRUCTION, USE_EXPLANATION)
**Second Normalize**

**Staff** (STAFF_ID, STAFF_NAME, STAFF_IC_NO, STAFF_BIRTH_DATE, STAFF_ADDRESS, STAFF_CONTACT_NO, STAFF_GENDER, STAFF_POSITION, STAFF_STATUS)

**Login** (LOGIN_ID, PASSWORD, STAFF_ID, STAFF_POSITION, STAFF_NAME)

**Patient** (PATIENT_ID, PATIENT_NAME, PATIENT_IC_NO, PATIENT_GENDER, PATIENT_BIRTH_DATE, PATIENT_MARITAL_STATUS, PATIENT_REGISTER_DATE, PATIENT_STATUS, PATIENT_ADDRESS, PATIENT_CONTACT_NO)

**Appointment** (APPOINTMENT_ID, PATIENT_ID, PATIENT_NAME, CONTACT_NO, STAFF_ID, STAFF_NAME, APPOINTMENT_DATE, APPOINTMENT_TIME, BOOKMARK)

**Queue** (QUEUE_ID, QUEUE_DATE, PATIENT_ID, PATIENT_NAME, PATIENT_CONTACT, STAFF_ID, STAFF_NAME, QUEUE_STATUS)

**Visit History** (VISIT_ID, VISIT_DATE, PATIENT_ID, PATIENT_NAME, STAFF_ID, STAFF_NAME, SYMPTOM, TEST_CONDUCTED, DIAGNOSIS, PRESCRIPTION_ID, TOTAL_AMOUNT)

**Prescription** (PRESCRIPTION_ID, ITEM_ID, ITEM_NAME, ITEM_FUNCTION, ITEM_PRICE, QUANTITY, DISCOUNT_PRICE, GRAND_TOTAL, USE_METHOD, USE_QTY, USE_UNIT, USE_TIME_DAILY, USE_INSTRUCTION, USE_EXPLANATION)

**Inventory** (ITEM_ID, ITEM_NAME, ITEM_UNIT, ITEM_QTY, ITEM_PRICE, ITEM_FUNCTION, SUPPLIER_ID, SUPPLIER_NAME, SUPPLIER_CONTACT_NO, SUPPLIER_ADDRESS)

**Supplier** (SUPPLIER_ID, SUPPLIER_NAME, SUPPLIER_CONTACT_NO, SUPPLIER_ADDRESS)
Third_Normalize

Staff (STAFF_ID, STAFF_NAME, STAFF_IC_NO, STAFF_BIRTH_DATE, STAFF_ADDRESS, STAFF_CONTACT_NO, STAFF_GENDER, STAFF_POSITION, STAFF_STATUS)

Login (LOGIN_ID, PASSWORD, STAFF_ID)

Patient (PATIENT_ID, PATIENT_NAME, PATIENT_IC_NO, PATIENT_GENDER, PATIENT_BIRTH_DATE, PATIENT_MARITAL_STATUS, PATIENT_REGISTER_DATE, PATIENT_STATUS, PATIENT_ADDRESS, PATIENT_CONTACT_NO)

Appointment (APPOINTMENT_ID, PATIENT_ID, STAFF_ID, APPOINTMENT_DATE, APPOINTMENT_TIME, BOOKMARK)

Queue (QUEUE_ID, QUEUE_DATE, PATIENT_ID, STAFF_ID, QUEUE_STATUS)

Visit History (VISIT_ID, VISIT_DATE, PATIENT_ID, SYMPTOM, TEST_CONDUCTED, DIAGNOSIS, PRESCRIPTION_ID, TOTAL_AMOUNT)

Prescription (PRESCRIPTION_ID, ITEM_ID, QUANTITY, DISCOUNT_PRICE, GRAND_TOTAL, USE_METHOD, USE_QTY, USE_UNIT, USE_TIME_DAILY, USE_INSTRUCTION, USE_EXPLANATION)

Inventory (ITEM_ID, ITEM_NAME, ITEM_UNIT, ITEM_QTY, ITEM_PRICE, ITEM_FUNCTION, SUPPLIER_ID)

Supplier (SUPPLIER_ID, SUPPLIER_NAME, SUPPLIER_CONTACT_NO, SUPPLIER_ADDRESS)
3.2 Sequence Diagram

Sequence Diagram for Clinic Information Maintenance
Sequence Diagram for Staff Profile Maintenance

- Actor
- :Staff UI
  - Start Interface()
  - Validate Staff ID()
  - Display Staff Detail()
- :Staff Control
  - Get Staff Detail()
  - Return Staff Detail()
- :Staff
  - Update Staff()
  - Delete Staff()
- :Login
  - Update Password()
**Sequence Diagram for Inventory Maintenance**

- **Input Item ID()**
  - Validate Item ID()
  - Start Interface()

- **Click Edit Button()**
  - Display Item Detail()
  - Get Item Detail()
  - Return Item Detail()

- **Update Item Detail()**
  - Validate Data()

- **Click Delete Button()**
  - Delete Record()
  - Delete Item()

- **Click AddNew Button()**
  - Enter New Item Detail()
  - Validate Data()
  - Add Item()
Sequence Diagram for Patient Profile Maintenance

Actor

Patient UI

Start Interface()

Validation Patient ID()

Display Patient Detail()

Get Patient Detail()

Patient Control

Return Patient Detail()

Update Patient()

Delete Patient()

Patient

Update Data()

Delete Record()

Add Patient()
Sequence Diagram for Patient Appointment Scheduling

Actor

:Appointment UI
- Start Interface()
- Validate Appointment ID()
- Display Appointment Detail()
- Validate Data()
- Update Appointment()
- Delete Record()
- Delete Appointment()
- Add Appointment()

:Appointment Control
- Get Appointment Detail()
- Return Appointment Detail()

:Appointment
- Enter New Appointment Detail()
- Validate Data()
Sequence Diagram for Patient Queue List Maintenance

Actor

:Queue UI

- Start Interface()
- Validate Queue ID()
- Display Queue Detail()

:Queue Control

- Get Queue Detail()
- Return Queue Detail()

:Queue

- Update Queue()
- Delete Record()
- Delete Queue()

:Patient

- Get Patient Detail()
- Return Patient Detail()

- Add Queue()

- Remove All Queue()

- Validate Data()

- Enter New Queue Detail()

- Click Clear All Button()

- Enter PatientID()

- Click AddNew Button()

- Update Queue Detail()

- Click Delete Button()

- Click Edit Button()

- Input Queue ID()
3.3 Collaboration Diagram

Collaboration Diagram for Clinic Information Maintenance

- Admin/Doctor/Staff
  - 2: Update Clinic Detail()
  - 3: Click Update Button()

  **:Clinic Information Maintenance UI**
  - 1: Start Interface()
  - 3.1: Verify Clinic Detail()

  **:Clinic Information Maintenance Control**
  - 3.1.1: Update Clinic Detail()

  **:Clinic Information**
Collaboration Diagram for Staff Profile Maintenance

1: Start Interface()

2: Validate Staff ID()
3: Update Staff Details()
4: Click Update Button()
5: Click Delete Button()
6: Click Add New Button()
7: Enter New Staff Details()
8: Click Save Button()
9: Click Change Password Button()
10: Enter New Password()
11: Click Save Button()
12: Click Reset Password()

Admin/Doctor/Staff

:Staff Maintenance UI

2.1: Verify Staff ID()
4.1: Verify Staff Details()
5.1: Determine Option()
6.1: Determine Option()
8.1: Verify Staff Details()
9.1: Determine Option()
11.1: Verify Data()
12.1: Generate New Password()

:Staff Maintenance Control

11.1.1: Update Password()
12.1.1: Update Password()

:Login

:Staff Detail
**Collaboration Diagram for Inventory Maintenance**

1: Start Interface()  
2: Input Item ID()  
3: Update Item Detail()  
4: Click Update Button()  
5: Click Delete Button()  
6: Click Add New Button()  
7: Enter New Item Details()  
8: Click Save Button()

Admin/Doctor/Staff

:Inventory Maintenance UI

2.1: Verify Item ID()  
4.1: Verify Item Detail()  
5.1: Determine Option()  
6.1: Determine Option()  
8.1: Verify Item Detail()

:Inventory Maintenance Control

2.1.1: Get Item Detail()  
4.1.1: Update Item Detail()  
5.1.1: Delete Item()  
8.1.1: Add Item()

:Inventory
Collaboration Diagram for Patient Profile Maintenance

1: Start Interface()
   2.1: Verify Patient Staff ID()
   4.1: Verify Patient Detail()
   5.1: Determine Option()
   6.1: Determine Option()
   8.1: Verify Patient Detail()

2: Input Patient ID()
3: Update Patient Detail()
4: Click Update Button()
5: Click Delete Button()
6: Click Add New Button()
7: Enter New Patient Details()
8: Click Save Button()

Admin/Doctor/Staff

:Patient Maintenance UI

:Patient Maintenance Control

:Patient
Collaboration Diagram for Patient Appointment Scheduling

2: Input Appointment ID()
3: Update Appointment Detail
4: Click Update Button()
5: Click Delete Button()
6: Click Add New Button()
7: Enter New Appointment Details()
8: Click Save Button()

Admin/Doctor/Staff

Appointment Maintenance UI

1: Start Interface()

2.1: Verify Appointment ID()
4.1: Verify Appointment Detail()
5.1: Determine Option()
6.1: Determine Option()
8.1: Verify Appointment Detail()

Appointment Maintenance Control

2.1.1: Get Appointment Detail()
4.1.1: Update Appointment Detail()
5.1.1: Delete Appointment()
8.1.1: Add Appointment()
Collaboration Diagram for Patient Queue List Maintenance

2: Input Queue ID()
3: Update Queue Detail()
4: Click Update Button()
5: Click Delete Button()
6: Click Add New Button()
7: Enter Patient ID()
8: Enter New Queue Details()
9: Click Save Button()
10: Click Clear All Button()

1: Start Interface

:Queue Maintenance UI

2.1: Verify Queue ID()
4.1: Verify Queue Detail()
5.1: Determine Option()
6.1: Determine Option()
7.1: Verify Patient ID()
9.1: Verify Queue Detail()
10.1: Determine Option()

:Queue Maintenance Control

2.1.1: Get Queue Detail()
4.1.1: Update Queue Detail()
5.1.1: Delete Queue()
9.1.1: Add Queue()
10.1.1: Remove All Queue()

:Patient

:Queue

Admin/Doctor/Staff
**Collaboration Diagram for Patient Visit History Maintenance**

2: Input Visit ID()
3: Update Visit Detail()
4: Click Update Visit Button()
5: Click Delete Visit Button()
6: Input Prescription ID()
7: Update Prescription Detail()
8: Click Update Prescription Button()
9: Click Delete Prescription Button()
10: Click Add New Visit Button()
11: Enter New Visit Details()
12: Click Add New Prescription Button()
13: Enter Medicine ID()
14: Enter New Prescription Detail()
15: Click Add Prescription Button()
16: Click Save Visit Button()

1: Start Interface

Admin/Doctor/Staff

:Visit History UI

2.1: Verify Visit ID()
4.1: Verify Visit Detail()
5.1: Determine Option()
6.1: Determine Prescription ID()
8.1: Verify Prescription Detail()
9.1: Determine Option()
10.1: Determine Option()
12.1: Determine Option()
13.1: Verify Medicine ID()
15.1: Verify Prescription Detail()
16.1: Verify Visit Detail()

:Visit History Control

2.1.1: Get Visit Detail()
4.1.1: Update Visit Detail()
5.1.1: Delete Visit()
16.1.1: Add New Visit()

:Visit

12.1.1: Get Medicine Detail()

6.1.1: Get Prescription Detail()
8.1.1: Update Prescription Detail()
9.1.1: Delete Prescription()
15.1.1: Add New Prescription()

:Inventory

:Prescription
3.4 User Interface Design

Login Page

This is the login page of the system. It will appear once the user runs the system. The user must enter the valid username and password to login before they can start using system.

Main Page

This is the main menu after the user successful login to the system. The top of the page is a menu strip that contains the way to go to all the function of the system. The bar at the left hand side is the short cut button to some page that the user often used.
Staff Maintenance

This is the staff list which listed all the staff on the data grid view. The user can press the add button to register new staff. If the user want to modified or delete the staff record, double click the selected staff or select a staff record and press the edit button. The top right hand side is search function which allowed the user to search staff by staff ID or staff name.
Add New Staff

This is register new staff record page. The system will auto generate a unique staff ID for the new register staff. Press Add button to register the new staff after the user fill in all the required information.
Edit/ Delete Staff Record

This is edit or delete staff record page. The system will display the staff details in particular field. Press Save button after user finish modified the staff details. The Change Password button will lead the user to change password page to change their login password. The Reset Password button only allowed admin level user to press for reset the staff login password in case the staff had forgotten his/her password. The Delete button is allowed the user to delete the staff record.
Change Password

This is the change password page. The user must enter the correct old password before they can successfully change their login password.

Reset Password

This is reset password page. The system will generate a random 6 numbers as the temporary new password for the staff.
Inventory Maintenance

This is the inventory management page. The top right hand side corner is the search function to allow the user to search the item by item ID or item name. When the user press the Add button, all the input textbox will enable to let the user enter the new item details. When the user double click on the item record from the data grid view or select a record and press the Edit button, the item record detail will display on the particular field. User can modify the item details and press the Save button to save it or delete the item record from the database by press on the Delete button.
Clinic Information Maintenance

This is the clinic information management page. The user can change the clinic details such as contact number and address of the clinic. To update the clinic information, press the save button after finish modifying.
Calling Form Setting

This is the setting page for the patient calling form. The user can set the start position for the patient calling form when running the system.

Patient Calling Page

This is the patient calling page that will display on a LCD TV or screen to let the clinic patient see. The patient name will show on the screen when there is their turn to go in the consultation room or take medicine on the dispensary counter.
Patient Maintenance

This is the patient list which listed all the patient record on the data grid view. The top right hand side corner is search function which allowed the user to search the patient by patient ID or patient name. User can scan the barcode on the patient card to search for the patient record too. Press the Add New button to register new patient to the database. Double click the record on the data grid view or select a record and press Edit button to go to the Edit patient profile page. To print the patient card, select any record from the data grid view and press the Print Patient Card button.
Add New Patient

This is register new patient record page. The system will auto generate a unique patient ID for the new register patient. Press Add button to register the new patient after the user fill in all the required information.
Edit/Delete Patient Record

This is edit or delete patient record page. The system will display the patient details in particular field. Press the Save button to update the patient record after done the modification on the patient details. The Delete button is allowed the user to remove the patient record from database. The Print Patient Card button is to let the user to print the patient card for such patient.

Patient Card

This is how the patient card looks like. The bottom of the card is a bar of barcode which represent the patient ID. The user will scan the barcode by scanner to search for the patient record every time the patient visit to the clinic.
Patient Appointment Scheduling

This is the patient appointment scheduling. The patient can book for their next appointment with the doctor as long as the doctor is free on that slot. The user can click on the empty cell to add new appointment and click on the cell that showing appointment ID to edit or delete the appointment. The date time picker allowed the user to view or add the appointment on the date that the patient request.
Add New Appointment

This page is add new patient appointment page. The system will auto generate a unique appointment ID for the new patient appointment. Press Add button to add new appointment after the user fill in all the required information.
Edit/Delete Appointment Record

This is edit or delete patient appointment record page. The system will display the patient appointment details in particular field. Press the Save button to update the appointment details after done the modification on the details. The Delete button is allowed the user to remove the particular appointment record from database. The Follow Up button purpose is when the patient coming for the appointment, the user can press this button and add the patient to the queue list.
Patient Queue Maintenance

This is the patient queue page which allowed the user to register the patient into the queue list by clicking the Add button when the patient is come to the clinic for consultation. The Edit button is allowed the user to edit or delete the queue record details. The queue list will auto refresh every 5 seconds as the user can see the most latest queue status when other user change it. The Clear All Queue Record button is let the user to remove all the queue record from the database so that they can restart the queue number every day.
Add New Queue Record

This is add new queue record page. The system will auto generate a unique queue ID for the new queue record. Press Add button to add the new queue record after the user fill in all the required information.
Edit/Delete Queue Record

This is edit or delete queue record page. The system will display the queue record details in particular field. Press the Save button to update the queue details after done the modification on the details. The Delete button is allowed the user to remove the particular queue record from database. The Go button purpose is let the user direct go to the patient profile when there is the patient turn to consult the doctor. The View Dispensary Report button will bring the user to the patient dispensary report for collect medicine purpose. The Print Patient MC button allowed the user to print MC for the patient when the patient is request for it. The Print Payment Receipt button will generate the payment receipt for the user.
Patient Visit History

This is the patient visit history list which under the visit history’s tab at patient profile. The user is allowed to add new visit record by press the New Visit button. To edit the visit record, select either a record from the list and press the View Visit button. To view the dispensary report, select on the record from the list and press the View Dispensary Report button.
Add New Patient Visit Record

This is add new patient visit record page. The system will auto generate a unique visit ID for the new visit record. Press the Add Medicine button to add new medicine as prescription. Press Add Treatment button to add new treatment price. Press Remove button to remove any miss added medicine or treatment. Press Add button to add the new queue record after the user fill in all the required information and done the prescription.
Edit/Delete Patient Visit Record

This is edit or delete visit record page. The system will display the visit record details in particular field. Press the Save button to update the visit details after done the modification on the details. The Delete button is allowed the user to remove the particular visit record from database. The Add Medicine button purpose is let the doctor to add the medicine as prescription to the patient. The total amount textbox will auto calculate the total for all the medicine.
Add Medicine to prescription

This is add new medicine page. The top right hand side corner is search function to let the user search for the medicine in a faster way. The user needs to select a medicine from the data grid view at first. After that the medicine function and the price will auto display on the particular textbox. Press Add button to add the new prescription after the user fill in all the other information.
Edit/Delete Prescription Record

This is edit or delete prescription record page. The system will display the prescription record details in particular field. Press the Save button to update the details after done the modification. The Remove button is allowed the user to remove the particular prescription record from database.
Print Patient Dispensary Report

This is the print patient dispensary report. The user is required to select which patient and which visit record’s prescription to print.

Patient Dispensary Report

This is the outcome of the dispensary report. It will list out all the prescription that the doctor given to the patient with all the instruction and price.
**Print Payment Receipt**

This is the print payment receipt page. The user is required to select which patient and which visit record’s receipt to print.

**Payment Receipt**

This is how the payment receipt looks like. It will simply put which item did the patient purchase, price, quantity, discount price and the total amount.
Print Patient Medical Certificate (MC)

The user is required to select how many days and the reasons for the MC before they can successfully generate MC for the patient.

Patient Medical Certificate (MC)

This is the outcome of the patient MC. The doctor needs to sign on the MC personally to prove that the MC is accepted.
Chapter 4

Programming
4. Programming

4.1 Coding

Register New Staff Coding

Data Access Layer

```vbnet
Public Shared Sub AddNew(ByVal aStaff As StaffPD)
    sID = aStaff.staffID
    sName = aStaff.staffName
    sIC = aStaff.staffIC
    sBirthDate = aStaff.staffBirthDate
    sAddress = aStaff.staffAddress
    sContactNo = aStaff.staffContact
    sGender = aStaff.staffGender
    sEmerName = aStaff.staffEmerName
    sEmerNo = aStaff.staffEmerNo
    sEmerRelationship = aStaff.staffEmerRelationship
    sPosition = aStaff.staffPosition
    sStatus = aStaff.staffStatus
    sJoinDate = aStaff.staffJoinDate
    sResignDate = aStaff.staffResignDate
    sLoginID = aStaff.staffLoginID
    sLoginPassword = aStaff.staffLoginPassword

    Dim value As String = "INSERT INTO tblStaffDetail VALUES ('" & sID & ", '" & sName & ", '" & sIC & ", '" & sBirthDate & ", '" & sAddress & ", '" & sContactNo & ", '" & sGender & ", '" & sEmerName & ", '" & sEmerNo & ", '" & sEmerRelationship & ", '" & sPosition & ", '" & sStatus & ", '" & sJoinDate & ", '" & sResignDate & ", '" & sLoginID & ", '" & sLoginPassword & ")"
    Try
        Initialize()
        Dim adpAddStaff As New OleDbDataAdapter()
        adpAddStaff.InsertCommand = New OleDbCommand(value)
        adpAddStaff.InsertCommand.Connection = conSearch
        adpAddStaff.InsertCommand.ExecuteNonQuery()
    Catch sqle As OleDb.OleDbException
        MessageBox.Show(sqle.Message)
    End Try
    Terminate()
End Sub
```

Business Logic Layer

```vbnet
Public Sub AddNew()
    StaffDA.AddNew(Me)
End Sub
```
Presentation Layer

'==Add Staff=='

Private Sub btnAdd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAdd.Click
Try
    theStaff = New StaffPD(txtStaffID.Text, txtStaffName.Text.ToUpper(),
    txtIC1.Text, dtpBirthDate.Text, txtAddress.Text.ToUpper(), txtContactNo.Text,
    gender.ToString(),
    txtEmerName.Text.ToUpper(), txtEmerNo.Text, txtEmerRelationship.Text,
    cmbPosition.SelectedItem.ToString(), status.ToString(),
    dtpJoinDate.Text, "0",
    txtLoginID.Text.ToUpper, txtPassword.Text)
    theStaff.AddNew()
    MessageBox.Show("New Staff has been Added!")
clearInputField()
    Me.Close()
Catch ex As Exception
    MessageBox.Show(ex.Message)
End Try
End Sub
Display Staff Detail Coding

Data Access Layer

```vbnet
'==Display staff detail=='
Public Shared Function DisplayDetail(ByVal anID As String) As StaffPD
displayStaff = Nothing

Try
    Initialize()
    Dim sqlQuery As String = "SELECT * FROM tblStaffDetail WHERE staff_id = ''' & anID & '''"
    Dim adpDisplayDetail As New OleDbDataAdapter(sqlQuery, conSearch)
adpDisplayDetail.Fill(dsDisplayDetail, "DisplayRecord")
    If dsDisplayDetail.Tables("DisplayRecord").Rows.Count > 0 Then
        Dim SearchRow As DataRow
        SearchRow = dsDisplayDetail.Tables("DisplayRecord").Rows(0)
        sID = SearchRow.Item("staff_id").ToString()
        sName = SearchRow.Item("staff_name").ToString()
        sIC = SearchRow.Item("ic_no").ToString()
        sBirthDate = SearchRow.Item("birth_date").ToString()
        sAddress = SearchRow.Item("address").ToString()
        sContactNo = SearchRow.Item("contact_no").ToString()
        sGender = SearchRow.Item("gender").ToString()
        sEmerName = SearchRow.Item("emer_contact_name").ToString()
        sEmerNo = SearchRow.Item("emer_contact_no").ToString()
        sEmerRelationship = SearchRow.Item("emer_contact_relationship").ToString()
        sPosition = SearchRow.Item("position").ToString()
        sStatus = SearchRow.Item("status").ToString()
        sJoinDate = SearchRow.Item("join_date").ToString()
        sResignDate = SearchRow.Item("resign_date").ToString()
        sLoginID = SearchRow.Item("login_id").ToString()
        sLoginPassword = SearchRow.Item("password").ToString()
        displayStaff = New StaffPD(sID, sName, sIC, sBirthDate, sAddress, sContactNo, sGender, sEmerName, sEmerNo, sEmerRelationship, sPosition, sStatus, sJoinDate, sResignDate, sLoginID, sLoginPassword)
    Else
        Throw New NotFoundException("No Info in the database!")
    End If
End Try
```
End If
    dsDisplayDetail = Nothing
Finally
    Terminate()
End Try
Return displayStaff
End Function

Business Logic Layer

'==Display staff detail=='
Public Shared Function DisplayDetail(ByVal anID As String) As StaffPD
    Return StaffDA.DisplayDetail(anID)
End Function

Presentation Layer

'==Display staff to edit=='
Public Sub editStaff()
    Try
        frmStaffProfile.Show()
        Dim i As Integer
        i = DataGridView1.CurrentRow.Index
        frmStaffProfile.txtStaffID.Text = DataGridView1.Item(0, i).Value
        theStaff = StaffDA.DisplayDetail(frmStaffProfile.txtStaffID.Text)
        frmStaffProfile.txtStaffName.Text = theStaff.staffName.ToString()
        frmStaffProfile.txtIC1.Text = theStaff.staffIC.ToString()
        If (theStaff.staffGender.ToString() = "Male") Then
            frmStaffProfile.radMale.Checked = True
        Else
            frmStaffProfile.radFemale.Checked = True
        End If
        If (theStaff.staffStatus.ToString() = "Active") Then
            frmStaffProfile.radActive.Checked = True
        Else
            frmStaffProfile.radInactive.Checked = True
        End If
        frmStaffProfile.dtpBirthDate.Text = theStaff.staffBirthDate.ToString()
        frmStaffProfile.txtAddress.Text = theStaff.staffAddress.ToString()
        frmStaffProfile.txtContactNo.Text = theStaff.staffContact.ToString()
        frmStaffProfile.txtEmerName.Text = theStaff.staffEmerName.ToString()
        frmStaffProfile.txtEmerNo.Text = theStaff.staffEmerNo.ToString()
        frmStaffProfile.txtEmerRelationship.Text = theStaff.staffEmerRelationship.ToString()
        frmStaffProfile.dtpJoinDate.Text = theStaff.staffJoinDate.ToString()
        If (theStaff.staffResignDate = "0") Then
            frmStaffProfile.dtpResignDate.Checked = False
        Else
            frmStaffProfile.dtpResignDate.Checked = True
        End If
    End Try
End Sub
frmStaffProfile.dtpResignDate.Text =
theStaff.staffResignDate.ToString()

End If

frmStaffProfile.cmbPosition.Text = theStaff.staffPosition.ToString()
frmStaffProfile.txtLoginID.Text = theStaff.staffLoginID.ToString()
frmStaffProfile.txtPassword.Text = theStaff.staffLoginPassword.ToString()

Catch ex As Exception
    MessageBox.Show(ex.Message)
End Try

End Sub

---

**Edit Staff Detail Coding**

**Data Access Layer**

```vbnet
'==Edit staff=='
Public Shared Sub Update(ByVal aStaff As StaffPD)
    Dim sqlUpdate As String
    sID = aStaff.staffID
    sName = aStaff.staffName
    sIC = aStaff.staffIC
    sBirthDate = aStaff.staffBirthDate
    sAddress = aStaff.staffAddress
    sContactNo = aStaff.staffContact
    sGender = aStaff.staffGender
    sEmerName = aStaff.staffEmerName
    sEmerNo = aStaff.staffEmerNo
    sEmerRelationship = aStaff.staffEmerRelationship
    sPosition = aStaff.staffPosition
    sStatus = aStaff.staffStatus
    sResignDate = aStaff.staffResignDate

    sqlUpdate = "UPDATE tblStaffDetail SET [staff_name] = "+ sName +",[ic_no] = "+ sIC +",[birth_date] = "+ sBirthDate +",[address] = "+ sAddress +",[contact_no] = "+ sContactNo +"
    sqlUpdate = sqlUpdate +",([staff_position] = " + sPosition + ",[staff_status] = " + sStatus + ",[staff_resign_date] = " + sResignDate + ")"
    sqlUpdate = sqlUpdate + ",[staff_id] = " + sID + "},

    Dim rs As New SqlConnection(ConnectionString)
    rs.Open()
    cmd = rs.CreateCommand()
    cmd.CommandText = sqlUpdate
    cmd.CommandType = CommandType.Text
    cmd.ExecuteNonQuery()
    rs.Close()
End Sub
```
Business Logic Layer

`==Edit Staff==`
Public Sub Update()
    StaffDA.Update(Me)
End Sub

Presentation Layer

Private Sub btnAdd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnAdd.Click
    Try
        theStaff = New StaffPD(txtStaffID.Text, txtStaffName.Text.ToUpper(),
            txtIC1.Text, dtpBirthDate.Text, txtAddress.Text.ToUpper(),
            txtContactNo.Text, gender.ToString(),
            txtEmerName.Text.ToUpper(), txtEmerNo.Text, txtEmerRelationship.Text,
            cmbPosition.SelectedItem.ToString(), status.ToString(),
            dtpJoinDate.Text, outDate.ToString(),
            txtLoginID.Text.ToUpper, txtPassword.Text)
Delete Staff Record Coding

Data Access Layer

'==Delete staff =='
Public Shared Sub Delete(ByVal aStaff As StaffPD)
    Dim sqldelete As String
    theStaff.Update()
    MessageBox.Show("Staff record has been updated!")
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub
sID = aStaff.staffID

sqldelete = "DELETE FROM tblStaffDetail WHERE staff_id = '' & sID & '''"

Try
    Initialize()
    Dim adpDeleteStaff As New OleDbDataAdapter()
    adpDeleteStaff.DeleteCommand = New OleDbCommand(sqldelete)
    adpDeleteStaff.DeleteCommand.Connection = conSearch
    adpDeleteStaff.DeleteCommand.ExecuteNonQuery()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    Finally
        Terminate()
    End Try
End Sub

Business Logic Layer

'==Delete staff =='
Public Sub Delete()
    StaffDA.Delete(Me)
End Sub

Presentation Layer

'==Delete staff=='
Private Sub btnDelete_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDelete.Click
    If MessageBox.Show("Confirm to Delete?", "Caption", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) = Windows.Forms.DialogResult.OK Then
        Try
            theStaff = New StaffPD(txtStaffID.Text, txtStaffName.Text.ToUpper(), txtIC1.Text, dtpBirthDate.Text, txtAddress.Text.ToUpper(), txtContactNo.Text, gender.ToString(), txtEmerName.Text.ToUpper(), txtEmerNo.Text, txtEmerRelationship.Text, cmbPosition.SelectedItem.ToString(), status.ToString(), dtpJoinDate.Text, dtpResignDate.Text.ToString(), txtLoginID.Text.ToUpper, txtPassword.Text)
            theStaff.Delete()
            MessageBox.Show("Staff Record has been Deleted!")
            clearInputField()
            Me.Close()
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    Else
        Exit Sub
    End If
End Sub
Search Staff Record Coding

Data Access Layer

'==Search Staff by key(ID)=='

Public Shared Function FindByKey(ByVal keys As String) As DataTable
    Dim ds As New DataSet
    Try
        Initialize()
        Dim sqlQuery As String = "SELECT [staff_id],[staff_name],[contact_no],[position],[status] FROM tblStaffDetail WHERE staff_id LIKE '" & keys & "%'"
        Dim adpSearch As New OleDbDataAdapter(sqlQuery, conSearch)
        adpSearch.Fill(ds, "tblStaffDetail")
        Catch e As OleDb.OleDbException
            MessageBox.Show(e.Message)
        End Try
        Terminate()
        Return ds.Tables(0)
    End Function

Business Logic Layer

'==Search by staff ID=='

Public Function FindByKey(ByVal aKey As String) As DataTable
    Return StaffDA.FindByKey(aKey)
End Function

Presentation Layer

'==Search staff by staff ID key=='

Private Sub txtsearch_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtsearch.TextChanged
    Try
        Dim dt As New DataTable
        dt = theStaff.FindByKey(txtsearch.Text)
        If Not dt Is Nothing Then
            If dt.Rows.Count > 0 Then
                DataGridView1.DataSource = dt
            Else
                DataGridView1.DataSource = Nothing
            End If
        Else
           DataGridView1.DataSource = Nothing
        End If
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub
List All Patient at Data Grid View Coding

Data Access Layer

```
'==Data grid view @ patient list=='
Public Shared Function GetPatient() As DataTable
    Dim ds As New DataSet
    Dim value As String = "SELECT [patient_id],[patient_name],[ic_no],[gender],[birth_date],[contact_no],[status] FROM tblPatient"
    Try
        Initialize()
        Dim adpAddPatient As New OleDbDataAdapter()
        adpAddPatient.SelectCommand = New OleDbCommand(value)
        adpAddPatient.SelectCommand.Connection = conSearch
        adpAddPatient.Fill(ds, "tblPatient")
        Catch sql As OleDb.OleDbException
            MessageBox.Show(sql.Message)
        End Try
    Terminate()
    Return ds.Tables(0)
End Function
```

Business Logic Layer

```
'==datagrid view patient=='
Public Function getallPatient() As DataTable
    Return PatientDA.GetPatient()
End Function
```

Presentation Layer

```
Private Sub frmPatientSearch_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Me.DataGridView1.DataSource = thePatient.getallPatient()
    cmbSearch.SelectedIndex = 0
End Sub
```
**Validate Patient ID Coding**

**Data Access Layer**

```vbnet
'==validate patient id=='
Public Shared Function validatePatientID(ByVal strname As String) As Boolean
    Dim sqlSelect As String
    Dim ds As New DataSet
    sqlSelect = "SELECT [patient_id] FROM tblPatient WHERE [patient_id] = " & strname & ""
    Try
        Initialize()
        Dim adpSelectPatient As OleDbDataReader
        Dim cmdSelectPatient As New OleDbCommand(sqlSelect, conSearch)
        adpSelectPatient = cmdSelectPatient.ExecuteReader()
        If (adpSelectPatient.HasRows) Then
            Terminate()
            Return True
        Else
            Terminate()
            Return False
        End If
    Catch e As Exception
        MessageBox.Show(e.Message)
    Finally
        End Try
        Return False
    End Function
```

**Business Logic Layer**

```vbnet
'==validate patient id=='
Public Function validatePatientID(ByVal strname As String) As Boolean
    Return PatientDA.validatePatientID(strname)
End Function
```

**Presentation Layer**
'==validation patient ID=='
If Not thePatient.validdatePatientID(txtPatientID.Text) Then
    MessageBox.Show("Error! Please enter a valid patient ID.")
    txtPatientID.Clear()
    txtPatientID.Focus()
    Exit Sub
End If

Auto Generated ID Coding

Data Access Layer

'==auto no @ Queue=='
Public Shared Function GetMaxID() As String
    Dim ds As New DataSet
    Dim strid As String = "0"
    Dim value As String = "SELECT MAX(queue_id) FROM tblQueue"
    Try
        Initialize()
        Dim cmditem As New OleDbCommand(value, conSearch)
        Dim ardAddQueue As OleDbDataReader
        ardAddQueue = cmditem.ExecuteReader()
        If ardAddQueue.HasRows() Then
            While ardAddQueue.Read()
                strid = ardAddQueue.Item(0).ToString()
            End While
        End If
        If (strid = "") Then
            strid = "Q001"
        Else
            strid = "Q" + (Integer.Parse(strid.Substring(1, 3)) + 1).ToString("D3")
        End If
        'strid = (Integer.Parse(strid) + 1).ToString()
        Catch sql As OleDb.OleDbException
            MessageBox.Show(sql.Message)
        End Try
        Terminate()
    Return strid
Business Logic Layer

```vbnet
'==auto ID @ Queue=='
Public Function getMaxID() As String
    Return queueDA.GetMaxID()
End Function
```

Presentation Layer

```vbnet
Private Sub frmQueueDetail_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    txtQueueID.Text = theQueue.getMaxID().ToString()
End Sub
```

Clear All Queue Record in Database Coding

Data Access Layer

```vbnet
'==clear queue record=='
Public Shared Sub clearQueue()
    Dim dropTable As String = "DROP TABLE tblQueue "
    Dim addTable As String = "CREATE TABLE tblQueue ([queue_id] TEXT(20) NOT NULL PRIMARY KEY, [date] DATE DEFAULT NULL, [patient_id] TEXT(20) DEFAULT NULL, [patient_contact] TEXT(20) DEFAULT NULL, [doctor_name] TEXT(20) DEFAULT NULL, [queue_status] TEXT(20) DEFAULT NULL, [queue_bookmark] TEXT(20) DEFAULT NULL)"
    Try
        Initialize()
        '==Drop table=='
        Dim cmdDrop As New OleDbCommand(dropTable, conSearch)
        cmdDrop.ExecuteNonQuery()
        '==Create table=='
        Dim cmdCreate As New OleDbCommand(addTable, conSearch)
        cmdCreate.ExecuteNonQuery()
    Catch sql As OleDb.OleDbException
        MessageBox.Show(sql.Message)
    End Try
    Terminate()
End Sub
```
Business Logic Layer

```vbnet
'==clear queue record=='
Public Sub clearQueue()
    queueDA.clearQueue()
End Sub
```

Presentation Layer

```vbnet
'==clear queue record=='
Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnClear.Click

    If MessageBox.Show("Confirm to clear all the queue record?", "Caption", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) = Windows.Forms.DialogResult.OK Then
        Try
            theQueue.clearQueue()
            MessageBox.Show("Queue Record has been cleared!")
            Me.DataGridView1.DataSource = theQueue.getAllQueue()
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    Else
        Exit Sub
    End If

End Sub
```

Load Appointment Schedule Table Coding

Data Access Layer

```vbnet
' === Get All Doctor Name === '
Public Shared Function GetAllDoctorName() As DataTable
    Dim dsStaff As New DataSet()
    Try
        Initialize()
        Dim sqlQuery As String = "SELECT * FROM tblStaffDetail WHERE [position] = 'Doctor'"
        adpstaff As New OleDbDataAdapter(sqlQuery, conSearch)
        adpstaff.Fill(dsStaff, "tblStaffDetail")
    Catch e As OleDb.OleDbException
        MessageBox.Show(e.Message)
    Finally
        Terminate()
    End Try
    Return dsStaff.Tables(0)
End Function
```
Business Logic Layer

```vbnet
'==Get doctor name=='
Public Shared Function GetAllStaffName() As DataTable
    Return StaffDA.GetAllDoctorName()
End Function
```

Presentation Layer

```vbnet
Public Sub LoadAppointmentTable()
    dgvdoctorslot.DataSource = Nothing

    Dim dtapp As New DataTable
    dtapp.Columns.Add("Time")

    ' Get All Doctor Name From Database '
    Dim dtdoctor As New DataTable
    dtdoctor = StaffPD.GetAllStaffName()
    For i As Integer = 0 To dtdoctor.Rows.Count - 1
        ' Add Column to Datatable '
        dtapp.Columns.Add(dtdoctor.Rows(i).Item("staff_name").ToString())
    Next

    Dim ds As New DataSet()
    Dim operationHours = "SELECT * FROM tblOperationHours "
    Dim adpGet As New OleDbDataAdapter(operationHours, conSearch)
    adpGet.Fill(ds)

    If (Not ds Is Nothing) Then
        If (ds.Tables(0).Rows.Count > 0) Then
            For j As Integer = 0 To ds.Tables(0).Rows.Count - 1
                Dim dr As DataRow
                dr = dtapp.NewRow
                For k As Integer = 0 To dtapp.Columns.Count - 1
                    If (k = 0) Then
                        dr(k) = ds.Tables(0).Rows(j).Item(0).ToString()
                    Else
                        Dim dtdocapp As New DataTable
                        If Not (AppointmentPD.SearchAppByDocName(dtapp.Columns(k).ColumnName, dtpAppDate.Value.ToShortDateString()) Is Nothing) Then
                            dtdocapp = AppointmentPD.SearchAppByDocName(dtapp.Columns(k).ColumnName, dtpAppDate.Value.ToShortDateString()).Tables(0)
                        Else
                            dtdocapp = Nothing
                        End If
                        If Not (dtdocapp Is Nothing) Then
                            If dttdocapp.Rows.Count > 0 Then
                                For a As Integer = 0 To dttdocapp.Rows.Count - 1
                                    appstarttime = dttdocapp.Rows(a).Item("app_time_start")
                                    Dim appstarttime As String =
                                Next
                            End If
                        End If
                    End If
                Next
            Next
        End If
    End If
End Sub
```
Dim appendtime As String =
    dtocapp.Rows(a).Item("app_time_end")
Dim todydate As String =
    ds.Tables(0).Rows(j).Item(0).ToString()
If ds.Tables(0).Rows(j).Item(0).ToString() =
    dtocapp.Rows(a).Item("app_time_start").ToString Or_
        (DateTime.Parse(appstarttime) <=
        (DateTime.Parse(appstarttime) Or_
            (DateTime.Parse(appendtime)) >
        DateTime.Parse(ds.Tables(0).Rows(j).Item(0).ToString())) Then
    dr(k) =
    dtocapp.Rows(a).Item("appointment_id").ToString
        Exit For
    Else
    dr(k) = ""
    End If
Next
Else
    dr(k) = ""
End If
Else
    dr(k) = ""
End If
Next
dtapp.Rows.Add(dr)
Next
End If
End If
dgvdoctorslot.DataSource = dtapp
End Sub
Chapter 5

Software Testing
5. **Software Testing**

5.1 **Testing Technique**

5.1.1 **Unit Testing**

Unit Testing is to make sure the module that been develop can run smoothly and executable without any bugs or system error. Unit testing basically will test on the particular part or unit such as function, procedures, or modules. It is to ensure that the particular function that has been developed can fulfil the requirement of the user and without any error. Unit testing is normally test by the programmer who writes the code. Every time after they had finish writing the code, they will run the debugger and test whether the function of source code can run smoothly or not. If there are bugs or system errors occur, the developer will try to solve it and then run the debugger again. If the function can run smoothly without error and can meet the required function, the developer will test the whole module all over again.

5.1.2 **Validation Testing**

Validation testing is one kind of testing which will test on whether the system is able to handle the wrong data that entered by the user or not. It also use to ensure that the data entered by the user is relevant and in the correct format or correct data types. For this testing, we will test whether the system can handle the data that entered by the user. For example, if the user is required to enter the numeric data but they are enter the string data, will the system able to track the data entered by user and prompt error message to inform the user to enter the correct one. This testing is very important because the wrong data entered may cause the system error or accidentally insert the wrong data into the database and misleading the system.
5.1.3 User Acceptance Testing

User Acceptance Testing is a testing to determine whether the system that had been developed is satisfied and accepted by the user or not. It will carry out after all testing are done and will test by the user who will use the system. In this testing, the user also can decide whether the acceptance level to the system or perhaps the user can provide some feedback to the programmer so that the programmer can make enhancement on the system.
## 5.2 Test Plan

<table>
<thead>
<tr>
<th>Module Name: Staff Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Task Name</strong></td>
</tr>
<tr>
<td>Validation</td>
</tr>
<tr>
<td>Add Staff</td>
</tr>
<tr>
<td>Edit Staff</td>
</tr>
<tr>
<td>Search Staff</td>
</tr>
<tr>
<td>List All Staff</td>
</tr>
<tr>
<td>Delete Staff</td>
</tr>
<tr>
<td>Change Login Password</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Reset Login Password</td>
</tr>
<tr>
<td>Module Name: Clinic Information Maintenance</td>
</tr>
<tr>
<td>--------------------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Task Name</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>Validation</td>
</tr>
<tr>
<td>Add Item</td>
</tr>
<tr>
<td>Edit Item</td>
</tr>
<tr>
<td>Search Item</td>
</tr>
<tr>
<td>List All Item</td>
</tr>
<tr>
<td>Delete Item</td>
</tr>
</tbody>
</table>
## Module Name: Patient Maintenance

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Test Objective</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>To check whether the validation will be executed if the required fields are blank or wrong data type.</td>
<td>Prompt out error message to inform the user that the blank fields or incorrect data types are entered.</td>
<td>Prompt out the error message.</td>
<td>Pass</td>
</tr>
<tr>
<td>Add Patient</td>
<td>Check whether the new patient can be added into the database.</td>
<td>Successful added the patient into the database.</td>
<td>Successful added into the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Edit Patient</td>
<td>Check whether the existing patient can be successful edit.</td>
<td>Successful update the patient details into database.</td>
<td>Successful update the particular patient details in the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Search Patient</td>
<td>Check whether the system can search the correct information that request by user.</td>
<td>Display the correct information based on the search criteria.</td>
<td>Display the correct information.</td>
<td>Pass</td>
</tr>
<tr>
<td>List All Patient</td>
<td>Check whether the system can list out all the patient record on data grid view.</td>
<td>Display all patient record that retrieved from the database.</td>
<td>Display all patient record.</td>
<td>Pass</td>
</tr>
<tr>
<td>Delete Patient</td>
<td>Check whether the patient record can be deleting.</td>
<td>Successful delete the patient record.</td>
<td>Successful removed patient record from database.</td>
<td>Pass</td>
</tr>
</tbody>
</table>
## Module Name: Patient Appointment Maintenance

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Test Objective</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>To check whether the validation will be executed if the required fields are blank or wrong data type.</td>
<td>Prompt out error message to inform the user that the blank fields or incorrect data types are entered.</td>
<td>Prompt out the error message.</td>
<td>Pass</td>
</tr>
<tr>
<td>Add Appointment</td>
<td>Check whether the new appointment can be added into the database.</td>
<td>Successful added the appointment into the database.</td>
<td>Successful added into the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Edit Appointment</td>
<td>Check whether the existing appointment can be successful edit.</td>
<td>Successful update the appointment details into database.</td>
<td>Successful update the particular appointment details in the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>List All Appointment</td>
<td>Check whether the system can display out all the appointment record on the appointment scheduling table.</td>
<td>Display all appointment record that retrieved from the database.</td>
<td>Display all appointment record.</td>
<td>Pass</td>
</tr>
<tr>
<td>Delete Appointment</td>
<td>Check whether the appointment record can be deleting.</td>
<td>Successful delete the appointment record.</td>
<td>Successful removed particular appointment record from database.</td>
<td>Pass</td>
</tr>
</tbody>
</table>
### Module Name: Patient Queue List Maintenance

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Test Objective</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>To check whether the validation will be executed if the required fields are blank or wrong data type.</td>
<td>Prompt out error message to inform the user that the blank fields or incorrect data types are entered.</td>
<td>Prompt out the error message.</td>
<td>Pass</td>
</tr>
<tr>
<td>Add Queue</td>
<td>Check whether the new queue record can be added into the database.</td>
<td>Successful added the queue record into the database.</td>
<td>Successful added into the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Edit Queue</td>
<td>Check whether the existing queue record can be successful edit.</td>
<td>Successful update the queue details into database.</td>
<td>Successful update the particular queue details in the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>List All Queue</td>
<td>Check whether the system can display out all the queue record on the data grid view.</td>
<td>Display all queue record that retrieved from the database.</td>
<td>Display all queue record.</td>
<td>Pass</td>
</tr>
<tr>
<td>Delete Queue</td>
<td>Check whether the queue record can be deleting.</td>
<td>Successful delete the queue record.</td>
<td>Successful removed particular queue record from database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Clear All Queue</td>
<td>Check whether the queue record has been clear.</td>
<td>Successful clear all the queue record.</td>
<td>Successful removed all queue record from database.</td>
<td>Pass</td>
</tr>
</tbody>
</table>
## Module Name: Patient Visit History Maintenance

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Test Objective</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>To check whether the validation will be executed if the required fields are blank or wrong data type.</td>
<td>Prompt out error message to inform the user that the blank fields or incorrect data types are entered.</td>
<td>Prompt out the error message.</td>
<td>Pass</td>
</tr>
<tr>
<td>Add Visit</td>
<td>Check whether the new visit record can be added into the database.</td>
<td>Successful added the visit record into the database.</td>
<td>Successful added into the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Edit Visit</td>
<td>Check whether the existing visit record can be successful edit.</td>
<td>Successful update the visit details into database.</td>
<td>Successful update the particular visit details in the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>List All Visit</td>
<td>Check whether the system can display out all the visit record from particular patient on the data grid view.</td>
<td>Display all visit record that retrieved from the database.</td>
<td>Display all visit record.</td>
<td>Pass</td>
</tr>
<tr>
<td>Delete Visit</td>
<td>Check whether the visit record can be deleting.</td>
<td>Successful delete the visit record.</td>
<td>Successful removed particular visit record from database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Add Prescription</td>
<td>Check whether the new prescription can be added into the database.</td>
<td>Successful added the prescription into the database.</td>
<td>Successful added into the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>Edit Prescription</td>
<td>Check whether the existing prescription record can be successful edit.</td>
<td>Successful update the prescription details into database.</td>
<td>Successful update the particular prescription details in the database.</td>
<td>Pass</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------</td>
<td>----------------------------------------------------------</td>
<td>---------------------------------------------------------------</td>
<td>-------</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Test Objective</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>To check whether the validation will be executed if the required fields are blank or wrong data type.</td>
<td>Prompt out error message to inform the user that the blank fields or incorrect data types are entered.</td>
<td>Prompt out the error message.</td>
<td>Pass</td>
</tr>
<tr>
<td>Print Report</td>
<td>Check whether the system can display the correct data on the report base on the user criteria.</td>
<td>Display the correct data on the particular field in report.</td>
<td>Correct data display on the report.</td>
<td>Pass</td>
</tr>
</tbody>
</table>
5.3 Validation and Verification

Login Module

If the user had entered the invalid login ID and password, the system will prompt out the error message to inform the user.
Clinic Information Maintenance

Edit Clinic Information

If the user missed out some input data, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when clinic information is being updated into database.
Staff Maintenance

Add New Staff

If the user missed out some input data when add staff, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new registered staff is being added into database.
Edit Staff Details

The system will prompt successful message to inform the user when details of current staff record is being updated into database.
Delete Staff Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete staff record action when user clicks on the Delete Button.

The system will prompt successful message to inform the user when the current staff record is being deleted from the database.
Change Login Password

The system will check on whether the old password that entered by the staff is correct or not before the staff could successful change their login password. Error message will prompt out to inform the user if wrong password had entered.

The system will prompt successful message to inform the user when the new login password is being updated into database.
Reset Password

The system will prompt a confirmation message to ask the user whether want to continue to perform reset password action when user clicks on the Reset Password Button.

The system will prompt successful message to inform the user and the message will display the new login password.
Inventory Maintenance

Add New Item

If the user missed out some input data when add new item, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new item is being added into database.
Edit Item Details

The system will prompt successful message to inform the user when current item details is being updated into database.
Delete Item Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete item when user click on the Delete Button.

The system will prompt successful message to inform the user when current item is successful deleted from the database.
Patient Maintenance

Add New Patient

If the user missed out some input data when add new patient, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new patient is being added into database.
Edit Patient Details

The system will prompt successful message to inform the user when current patient details is being updated into database.
Delete Patient Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete patient when user click on the Delete Button.

The system will prompt successful message to inform the user when current patient is successful deleted from the database.
Patient Appointment Scheduling

Add New Appointment

If the user missed out some input data when add new appointment record, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new appointment record is being added into database.
Edit Appointment Details

![Appointment Details GUI](image)

The system will prompt successful message to inform the user when current appointment details is being updated into database.
Delete Appointment Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete appointment record when user clicks on the Delete Button.

The system will prompt successful message to inform the user when current appointment record is successful deleted from the database.
Patient Queue Maintenance

Add New Queue Record

If the user missed out some input data when add new queue record, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new queue record is being added into database.
Edit Queue Details

The system will prompt successful message to inform the user when current queue record details is being updated into database.
Delete Queue Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete queue record when user clicks on the Delete Button.

The system will prompt successful message to inform the user when current queue record is successful deleted from the database.
Clear All Queue Record

The system will prompt a confirmation message to ask the user whether want to continue to perform clear all queue record when user clicks on the Clear All Queue Button.

The system will prompt successful message to inform the user when all queue record is successful deleted from the database.
Patient Visit History Maintenance

Add New Visit Record

If the user missed out some input data when add new visit record, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new visit record is being added into database.
Delete Visit Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete visit record when user clicks on the Delete Button.

The system will prompt successful message to inform the user when current visit record is successful deleted from the database.
Add New Prescription

If the user missed out some input data when add new prescription record, the system will prompt error message to the user to inform them to enter the data. The system will also prompt the error message when wrong data format enter by the user.

The system will prompt successful message to inform the user when new prescription record is being added into database.
Edit Prescription Detail

The system will prompt successful message to inform the user when current prescription details is being updated into database.
Remove Prescription Record

The system will prompt a confirmation message to ask the user whether want to continue to perform delete prescription record when user clicks on the Remove Button.

The system will prompt successful message to inform the user when current prescription record is successful deleted from the database.
Chapter 6

Conclusion
6. Conclusion

6.1 Evaluation against Project

6.1.1 Project Strength

- Faster response time toward patient

  Due to all information are already store in the database, so the staff can retrieve everything for example patient details through the system in a fastest way instead of searching the patient record from a bunch of messy document on the cabinet. This Clinic Management System is to achieve the patient satisfaction on shorter waiting time and better service provided to the patient. With this system, the clinic can save up a lot of response time to handle a patient. So, the clinic can provide better service to more patients within an operation day.

- Easy to maintain clinic data

  Everything are digitalize including all the data that belongs to the clinic will store in a well organized database. Staff is easy to maintain such well organized data as it is easy when they want to modified some detail from particular records, searching for data, add in some new data or delete some data with the help of the system. They can perform all the activity that I mentioned in an efficient way.

- Secure database

  The user of the system is allowed to access to those data that authority to his/her level. For example, the patient private information is restricted to access only by doctor level user at the clinic.

- Reduce number of worker needed

  With the help of the system, the clinic does not need to hire for so many workers to cope with bunch of patient because the operation of the clinic is digitalized. Most of the things could do by the system effectively and efficiency, thus the number of worker can be reduced.
6.1.2 Project Weaknesses

- **Do not support online use**

  It is an offline system which only used by the clinic staff. So it is impossible to let the patient to check back their medical history through the system because it do not support online.

6.2 Personal Reflection

Through the developing of this project, I gain a lot of extra knowledge. I had learned additional knowledge for VB.NET which is the programming language that I used to develop my project. Besides that, I got to know how to debug the system to find the error in an efficient way instead of go through the code one by one.

It is not easy to successful create a project because there is no perfect system as we only can try our best to make the system fulfil the user needs. User will have endlessly expectation towards the system since the day they start interact with the system. As a programmer, I must try to come out with solution to fulfil their requirement as long as the requirement is logical.

I had faced kind of coding problems, logical problem as well as system flow problem when I developing this project. Fortunately, I can able to solve all these problems with the aid of help from my supervisor, my friends and of course the solutions from Google search. Besides that, my supervisor does provide me with precious suggestion and advice on enhancing my system to a better one. With the advices and suggestion that he gave, it was bring a lot of benefits to us.

Although this project is a toughest task ever to me but I believe that it could really help me a lot when I come out to working. Last but not least, I know the effort that I’d pay and the time that contributed to this project is worth the knowledge and experience that I gain. I would like to use this opportunity to show my gratefulness to everyone that kindly given out their helping hand to me when I needed them. Thanks again.
7. Reference

Viewed on 29 October 2010.
Available from:
<http://www.pdfstop.com/view/aHR0cDovL3d3dy5qb3VybmsLmF1LmVkdS9pamNpbS8yMDAxL21heTAxL2FyY2hpGVjdHVyZS5kb2M=>

Viewed on 28 February 2010.
Available from:
<http://www.pdfstop.com/view/aHR0cDovL3d3dy5oaW1zcy5vcmcvY29udGVudC9mawaXlcy9JbXBsZWI1bnRhdGlvbI9ndWlkZS5wZGY=>.

Viewed on 28 February 2010.
Available from:
<http://en.wikipedia.org/wiki/Microsoft_Access>
8. **Appendices**

**Sample of Medical Certificate (MC) from clinic**

![Sample of Medical Certificate](image1)

**Sample of label on medicine from clinic**

![Sample of Medicine Label](image2)
9. **User Guide**

**Login**

1. Enter the login user name into the textbox.
2. Enter the login password into the textbox.
3. Click Login button to login to the system and go to main page.
4. Click Cancel button to close the system.
Main Page

1. System Task Menu - Logout and Exit

2. Master Data Menu - Treatment Maintenance, Staff Maintenance, Inventory Maintenance, Clinic Data Maintenance, and Calling Form Setting


5. Report Menu - Patient Card, Patient Dispensary Report, and Payment Receipt

6. Go to Patient List page short cut button.

7. Go to Patient Appointment page short cut button.

8. Go to Patient Queue page short cut button.

9. Go to Patient Dispensary Report short cut button.

10. Go to Print Payment Receipt short cut button.
Treatment Maintenance

1. Enter the treatment detail.
2. Click Add New button to add new treatment record to database.
3. List of the treatment. Double click the record to edit the treatment.
4. Click Save button to update the treatment detail to database.
5. Click Delete button to delete selected treatment record from database.
6. Click Cancel button to exit treatment manager page.
Staff Maintenance

1. Search Selection Menu - search by Staff ID or Staff Name.

2. Enter search criteria to filter the staff list.

3. List of the staff. Double click the record to edit the staff profile.

4. Click Add New button to go to Add New Staff page.

5. Click Edit button to go to Edit Staff Profile page for particular staff.

6. Click Cancel button to exit Staff List page.
Add New Staff

1. Enter new staff details.
2. Click Add button to add the new staff into database.
3. Click Cancel button to cancel add new staff.
Edit/ Delete Staff Record

1. Enter the staff details.

2. Click Save button to update the staff details into database.

3. Click Delete button to delete the staff record from database.

4. Click Change Password button to change new login password.

5. Click Reset Password button to reset the login password.

6. Click Cancel button to cancel edit staff.
Change Password

1. Enter change password details.
2. Click Save button to update new password into database.
3. Click Cancel button to cancel change password.
Inventory Maintenance

1. Enter the item details.
2. Click Add New button to add new item record to database.
3. List of the item. Double click the record to edit the item detail.
4. Click Save button to update the item details into database.
5. Click Delete button to delete selected item record from database.
6. Click Cancel button to exit Inventory Manager page.
7. Search Selection Menu - search by Item ID or Item Name.
8. Enter search criteria to filter the item list.
Clinic Information Management

1. Enter Clinic Information.
2. Click Save button to update clinic information into database.
3. Click Cancel button to exit Clinic Information page.
Calling Form Setting

1. Enter form setting.

2. Click Save button to update setting into database.

3. Click Cancel button to exit setting page.
Patient Maintenance

1. Search Selection Menu - search by Patient ID or Patient Name.
2. Enter search criteria to filter the patient list.
3. List of the patient. Double click the record to edit the patient profile.
4. Click Add New button to go to Add New Patient page.
5. Click Edit button to go to Edit Patient Profile page for particular patient.
6. Click Cancel button to exit Patient List page.
7. Click Print Patient Card button to print particular patient card.
Add New Patient

1. Enter new patient details.

2. Click Add button to add the new patient into database.

3. Click Cancel button to cancel add new patient.
Edit/Delete Patient Record

1. Enter the patient details.

2. Click Save button to update the patient details into database.

3. Click Print Patient Card button to print current patient’s patient card.

4. Click Delete button to delete the patient record from database.

5. Click Cancel button to cancel edit patient.
Patient Appointment Scheduling

1. Select which day appointment to display from date time picker.

2. Click Today button to go back to today date.

3. Double click on column that has record to edit the record.

4. Double click on blank column to add new appointment record.

5. Click Cancel button to exit the page.
Add New Appointment

1. Enter new appointment details.

2. Click Add button to add the new appointment into database.

3. Click Cancel button to cancel add new appointment.
Edit/Delete Appointment Record

1. Enter the appointment details.
2. Click Save button to update the appointment details into database.
3. Click Delete button to delete the appointment record from database.
4. Click Cancel button to cancel edit appointment.
5. Click Follow Up button to go to Add New Queue page.
Patient Queue Management

1. List of Queue. Double click on record to edit the queue record.

2. Click Add button to add new record into queue list.

3. Click Edit button to edit selected queue record.

4. Click Cancel button to exit Patient Queue page.

5. Click Clear All Queue Record button to remove all the queue record from database.
Add New Queue Record

1. Enter new queue details.

2. Click Add button to add the new queue into database.

3. Click Cancel button to cancel add new queue.
Edit/Delete Queue Record

1. Enter the queue details.

2. Click Save button to update the queue details into database.

3. Click Delete button to delete the queue record from database.

4. Click Cancel button to cancel edit queue.

5. Click Go button to go current patient profile.

6. Click View Dispensary Report button to view current visit’s dispensary report.

7. Click Print Patient MC Button to go to print patient MC page.

8. Click Print Payment Receipt button to print current visit’s payment receipt.
Print Patient Medical Certificate (MC)

1. Enter MC detail.
2. Click Print button to print the MC.
3. Click Cancel button to cancel print MC.
Patient Visit History

1. List of patient visit history.
2. Click New Visit button to add new visit record.
3. Click View Visit button to view and edit the selected visit record.
4. Click View Dispensary Report button to view selected visit record’s dispensary report.
Add New Patient Visit Record

1. Enter new visit details.

2. List of prescription. Double click on record from list to edit the prescription record.

3. Click Add Medicine button to add medicine into prescription.

4. Click Add Treatment button to add treatment into prescription.

5. Click Remove button to remove prescription item from prescription.

6. Click Add Visit button to add the new visit into database.

7. Click Cancel button to cancel add new visit.
Edit/Delete Patient Visit Record

1. Enter visit detail.
2. List of prescription. Double click on record from list to edit the prescription record.
3. Click Add Medicine button to add medicine into prescription.
4. Click Add Treatment button to add treatment into prescription.
5. Click Remove button to remove prescription item from prescription.
6. Click Save button to update the visit into database.
7. Click Delete Visit button to delete visit record from database.
8. Click Cancel button to cancel edit visit.
Add Medicine to prescription

1. Enter search criteria to filter the medicine list.
2. Select a medicine as prescription by double click a medicine from list.
3. Enter new prescription details.
4. Click Add button to add new prescription into database.
5. Click Cancel button to cancel add new prescription.
Edit/Delete Prescription Record

1. Enter search criteria to filter the medicine list.
2. Change medicine by double click a medicine from list.
3. Enter prescription details.
4. Click Save button to update prescription details into database.
5. Click Cancel button to cancel edit prescription.