UNIVERSITI TEKNIKAL MALAYSIA MELAKA

FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

W O R K S H O P 1

R E P O R T

Name : Muhammad Iqbal bin Mohamad Yunus

Matric Number : B031910263

Course : BITI

Project Title : Music Store Sales Management System

Supervisor Name : Dr. Noor Fazilla binti Abd Yusof

Supervisor Signature : _____

Evaluator Name : Ts. Dr. Norzihani binti Yusof

TABLE OF CONTENTS

CHAPTER 1

BACKGROUND OF THE PROJECT

1.1 INTRODUCTION

Information Technology has revolutionized the life of human beings and has made lives easier by the various kinds of applications. The project is concerned with developing a Music Store Management System for monitoring and controlling the transactions in the store which make it more efficient and easier to handle. Music Store Management System project is developed using C++ and MySQL.

1.2 PROBLEM STATEMENT

Most of the music stores are operated manually by a group of people. These people keep records regarding the books and user, check the books manually and keep records on issued books. All these things have to be carried out manually and if the store is very large, proper record keeping will become major problem as manual record keeping has never been a reliable method because people tend to forget things.

1.3 BACKGROUND OF PROJECT

The project is based on the concept of managing music store. Talking about the project, it contains lots of features. The store manager can manage all the records, such as adding new items in database, edit the items description, and delete any item. The store workers will be able to calculate bills and show the total items in stock.

1.4 OBJECTIVES

This project embarks on the following objectives:

1. To build the music store sales management system and make it easier for the users in managing the music store.
2. To provide a solution to keep track of music albums stock

1.5 SCOPE

i. Target Users

The target users of this system consist of admin, manager and staff of the music store.

ii. Module

The module of the target user is divided into three categories which are admin, manager and staff modules based on Table 1.1 below:

Table 1.1 : Module of Target Users

CATEGORY	MODULE
Admin	• Login • Add new registration
Manager	• Login • Add item in stock • Delete item in stock • Edit item description • Display item in stock
Staff	• Login • Calculate customer bills • Display item in stock

1.6 PROJECT SIGNIFICANT

This project is a solution to the music store by improving the sales management system by replacing the currently used manual system in store. It also can provide a more systematic transaction which can save a lot of time in making transaction.

1.7 HARDWARE AND SOFTWARE APPLICATION

1) Software Application

- XAMPP
- Microsoft Visual Studio 2019
- Microsoft Office Word 365
- Microsoft Visio 2016

2) Hardware Application

- Laptop Asus TUF FX505DU

1.8 CHAPTER SUMMARY

This chapter explain the project background, problem statement, objectives, scope and project significant. The Music Store Sales Management System is aimed to provide a more systematic way in managing music album sales and handling transaction.

CHAPTER 2
ANALYSIS OF PROBLEM

2.1 INTRODUCTION

This chapter outlines the problem description, problem decomposition and structure chart of the Music Store Sales Management System. Problem analysis is important to understand more about the problem, so it can be addressed accordingly.

2.2 DESCRIPTION OF THE PROBLEM

The current system serves as a major problem in managing music store's sales. The staff need to record all transaction manually into the record book. The sales record is not well recorded because the staff sometimes forget to record transaction into the record book. This happened many times because staff need to deal with many customers in the store and that made them forget to record the transaction into the record book.

2.3 PROBLEM DECOMPOSITION

Based on the observation made, there are few problems that need to overcome as shown in Table 2.1 below:

Table 2.1 : Problems Faced by the Manager and Staff

Problems	Solutions
When the store is crowded with customers, the staff sometimes forget to record the sales in the specific record book.	The system is created to overcome this problem as this system automatically record the store's sales.
As the store is quite big, sometimes customer ask the staff whether the album he/she wanted to buy is still available or out of stock.	The system can be used to identify how many albums left in stock in the store.
Some of the customers want to buy an album but they do not know where to find them as the music store is big.	Staff can use the system to search the album based on its artist name or the album name to identify the genre of the album as all albums in the music store is arranged based on genre.

2.4 STRUCTURED CHART

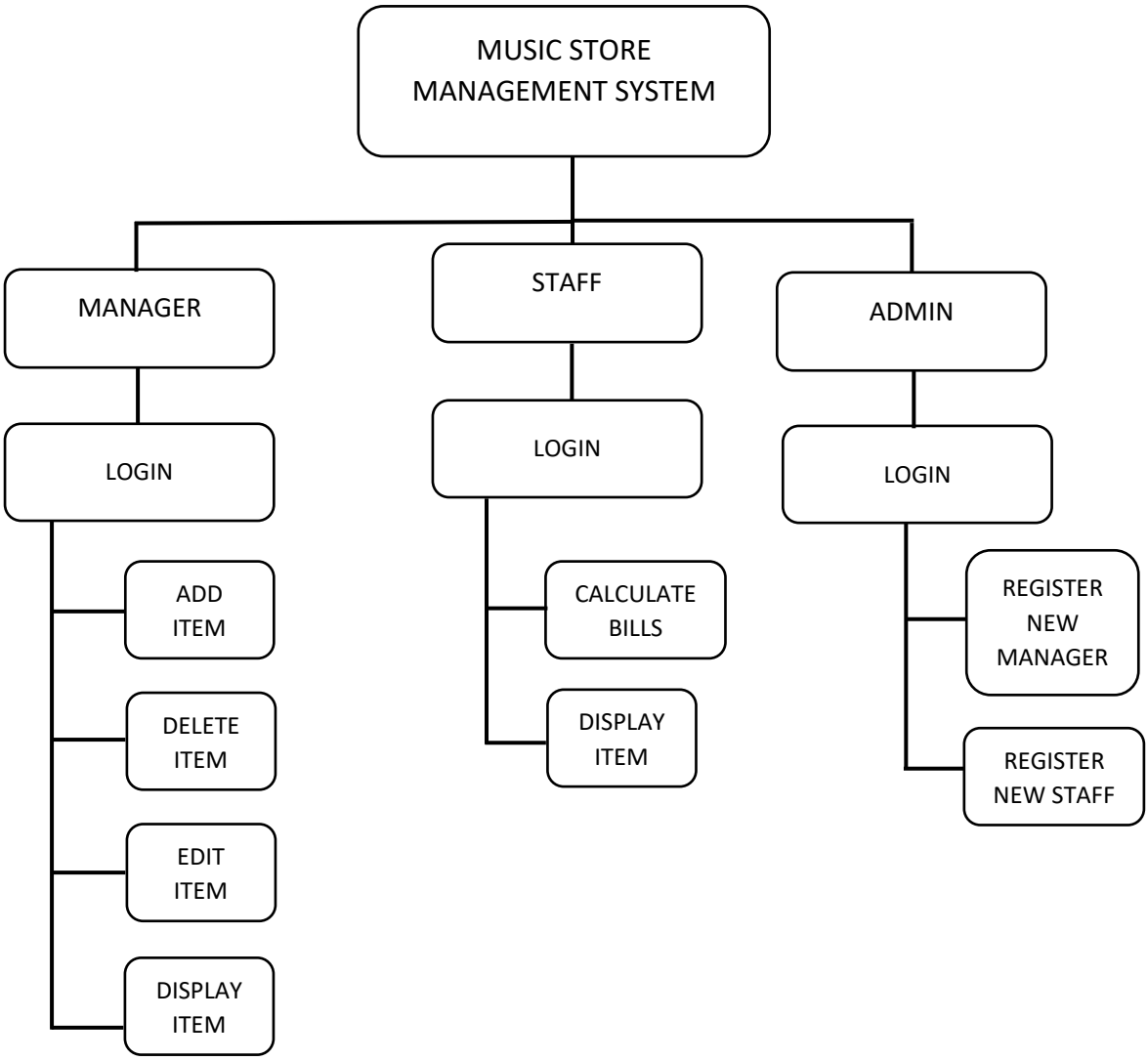

Figure 2.1 : Structured Chart of Music Store Management System

Figure 2.1 shows the structured chart of the Music Store Management System. Music Store Management System users are divided into three users i.e, manager, staff and admin. All the users need to login to be able to use the system. The manager is able to add item, delete item, edit item description and display item. The staff is able to make transactions and display item. The admin is needed to register new staff and new manager.

2.5 BUSINESS RULES

Business rules can apply to many aspects of an organization and can be expressed in a variety of ways. In general, business rules define specific instructions or constraints on how certain day-to-day actions should be performed.

- There must be only 1 manager in the music store.
- An album is made by 1 artist.
- An artist can made 1 or more albums.
- Each customer can buy 1 or more album at each purchase.

2.6 CHAPTER SUMMARY

In this chapter explain the problem and flaws of the previous system flow of process. This allow us to understand the old system that could be improved for the proposed system. I came out with the proposed system to ensure that the old system problem could be solved with the proposed system.

CHAPTER 3

DESIGN

3.1 INTRODUCTION

This chapter presents the design phase of this system. Design phase play an important role in developing a project. In this phase developer need to determine what needs to do for creating a new system interface. The design that has been proposed need to be studied to accomplish the project objective.

3.2 FLOWCHART

3.2.1 Manager

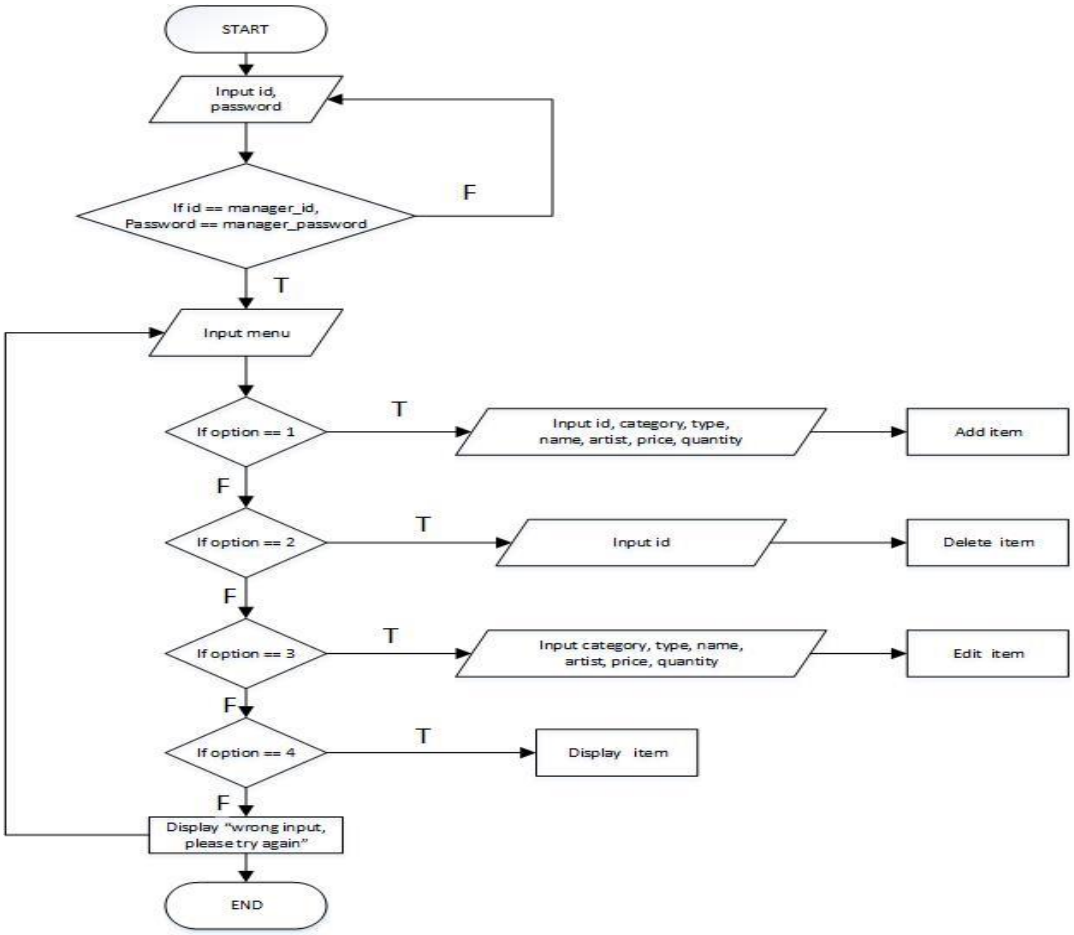

Figure 3.1 : Manager Tasks Flowchart

Figure 3.1 shows the flowchart for the manager task. In order to access the system, the manager need to login as manager with the manager ID and password. A manager will have access to add items, delete items, edit items description and display items.

3.2.2 Staff

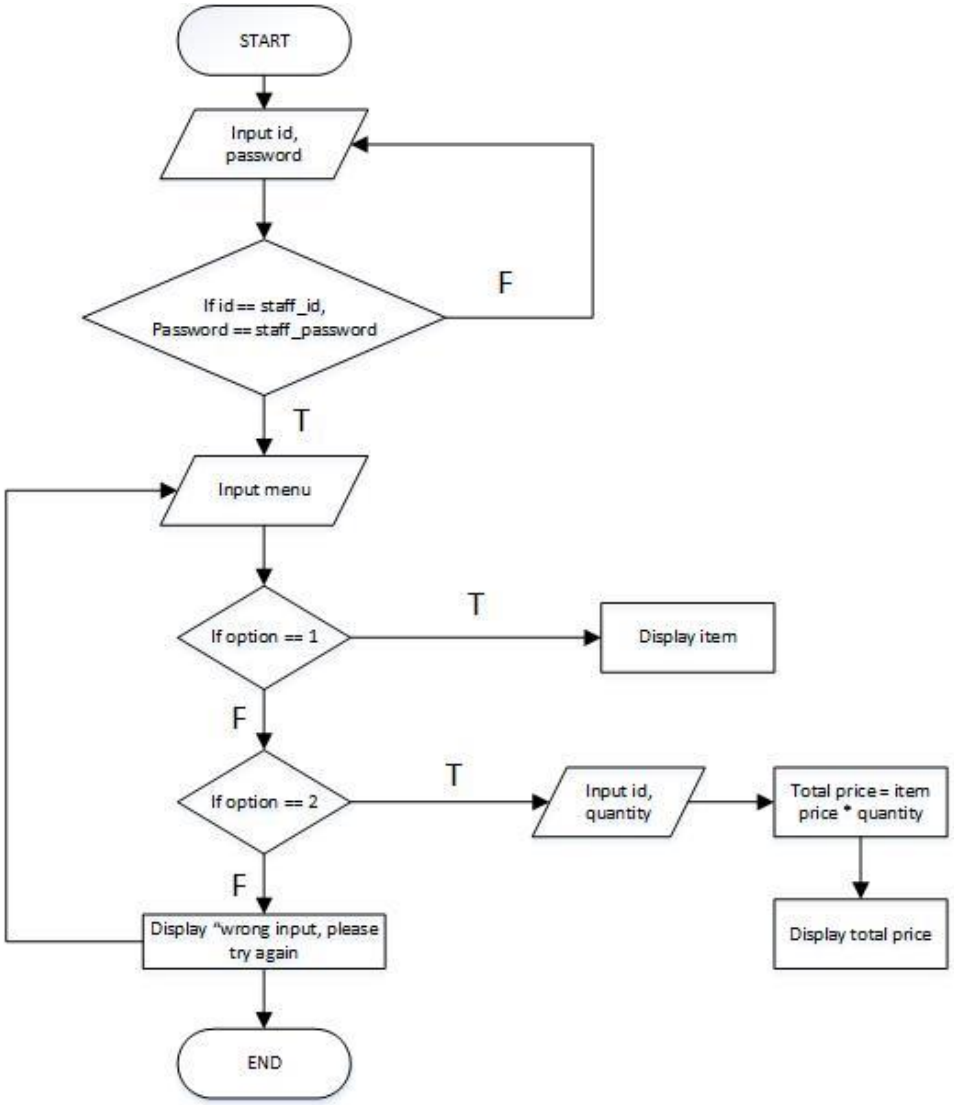

Figure 3.2 : Staff Task Flowchart

Figure 3.2 shows the flowchart for the staff tasks. The staff has access to search item and make transactions. In order to do that, staff need to login using staff ID and password.

3.2.3 Admin

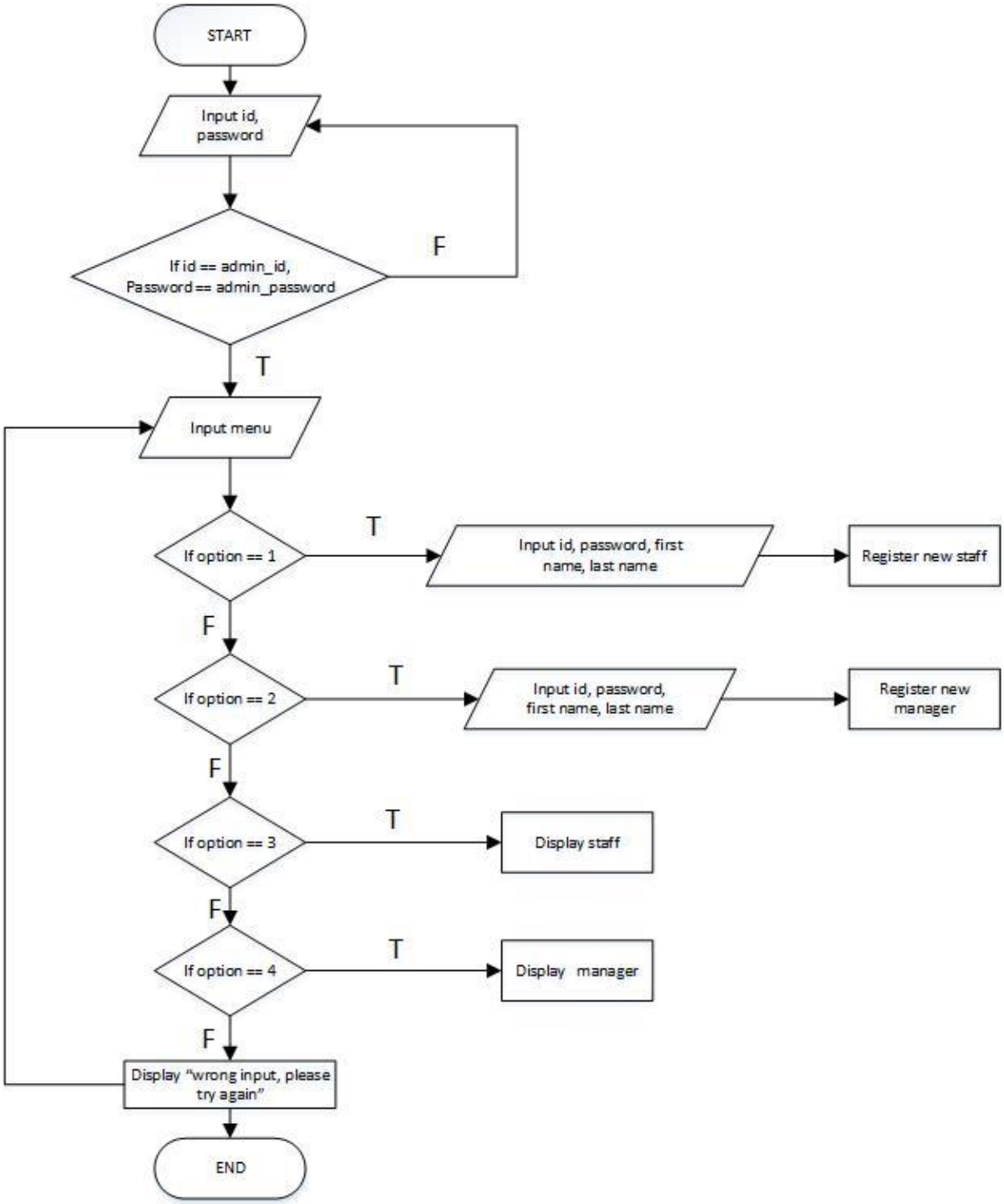

Figure 3.3 : Admin Tasks Flowchart

Figure 3.3 shows the flowchart for admin tasks. The admin is responsible for registering new staff or new manager. In order to do that, admin will need to login with admin ID and password. Admin will have access on registering new staff, registering new manager, display staff information and display manager information.

3.3 ENTITY RELATIONSHIP DIAGRAM (ERD)

This section shows the entities and their relationship to each other in the database for this system. It is very important to ensure that each tables in the database system were correctly build for avoiding redundancy.

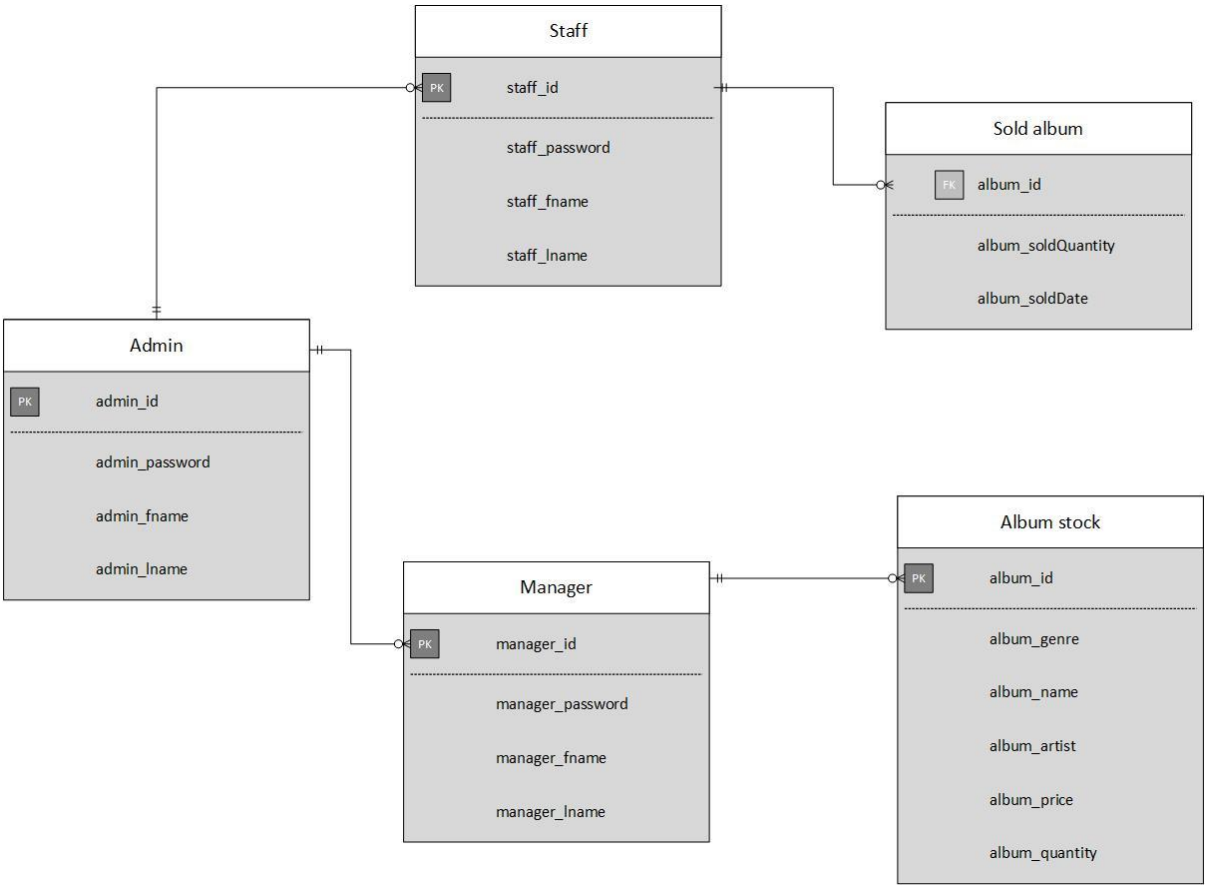

Figure 3.4 : ER Diagram for Staff, Manager, Admin, Sold Album and Album Stock

3.4 Data Dictionary

Data dictionary is an important component in any relational database. System admin will only interest with the data dictionary. Data dictionary defines a file or a set of files that contains in the database. In this section shows, data dictionary records in the database for this system, such as ownership, data relationship to other object, and other data.

Table 3.1 : Structure Table for Staff

Name	Type	Null	Default
staff_id	Varchar (50)	No	
staff_password	Varchar (50)	No	
staff_fname	Varchar (50)	No	
staff_lname	Varchar (50)	No	

Table 3.2 : Structure Table for Manager

Name	Type	Null	Default
manager_id	Varchar (50)	No	
manager_password	Varchar (50)	No	
manager_fname	Varchar (50)	No	
manager_lname	Varchar (50)	No	

Table 3.3 : Structure Table for Admin

Name	Type	Null	Default
admin_id	Varchar (50)	No	
admin_password	Varchar (50)	No	
admin_fname	Varchar (50)	No	
admin_lname	Varchar (50)	No	

Table 3.4 : Structure Table for Album Stock

Name	Type	Null	Default
album_id	Varchar (50)	No	
album_genre	Varchar (50)	No	
album_name	Varchar (50)	No	
album_artist	Varchar (50)	No	
album_price	Varchar (50)	No	
album_quantity	Int (50)	No	

3.5 Interface Design

3.5.1 Home Menu

Figure 3.5 shows the home menu interface where the must need to choose their account type i.e, Staff Account, Manager Account or Admin Account.

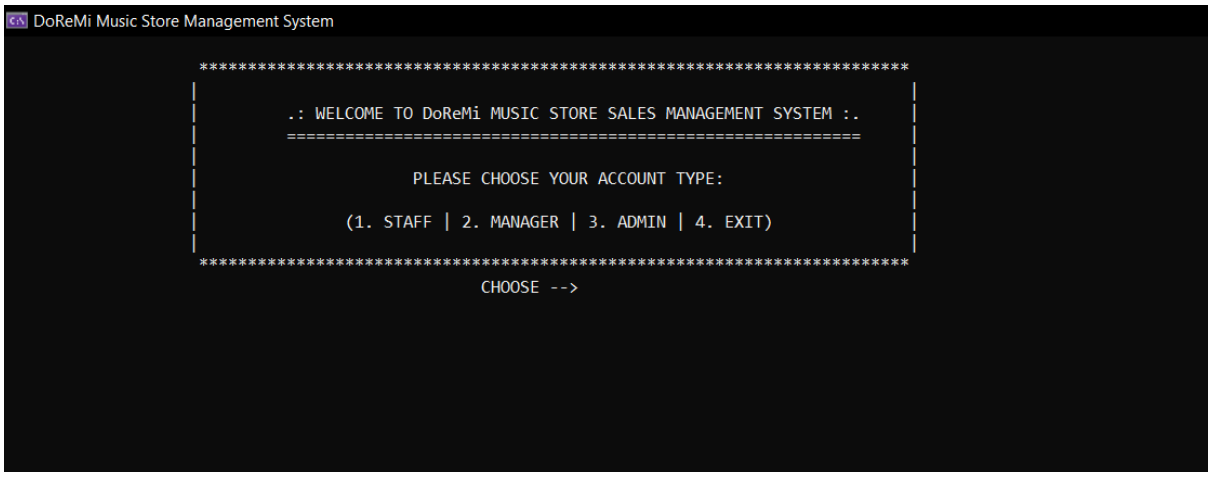

Figure 3.5 : Home Menu Interface

3.5.2 Staff Login

Figure 3.6 shows the login menu interface where the staff need to login using their Staff ID and password before using the system.

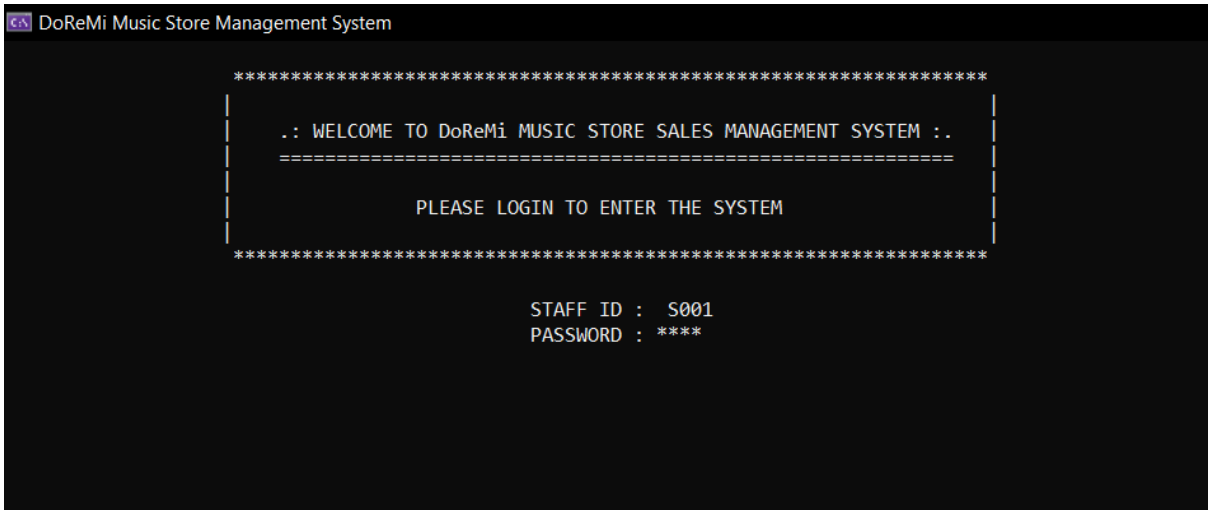

Figure 3.6 : Staff Login Interface

3.5.3 Login Successful

Figure 3.7 shows an interface to inform user that login has successful.

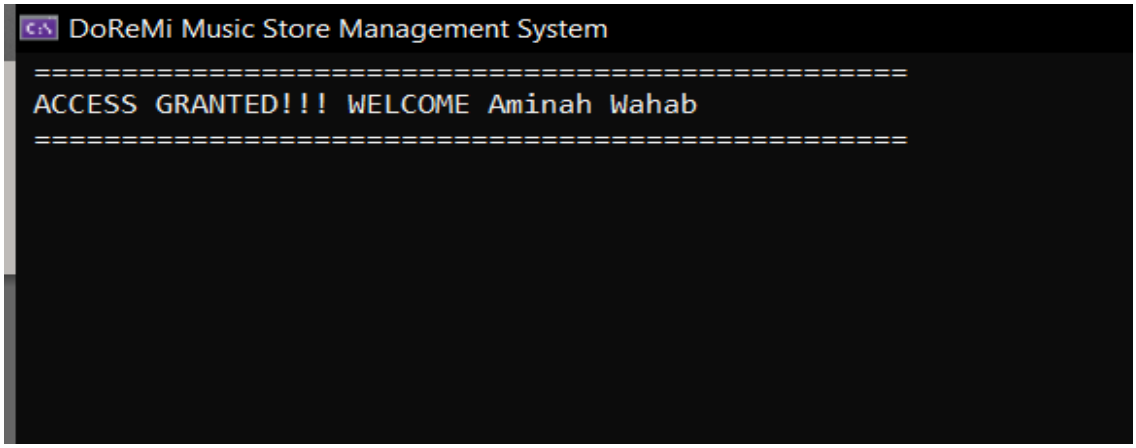

Figure 3.7 : Login Successful Interface

3.5.4 Staff Menu

Figure 3.8 shows the Staff Menu interface. There are three functions that the staff can access in this system which is Check Stock, Create Orders and Check Album Sales.

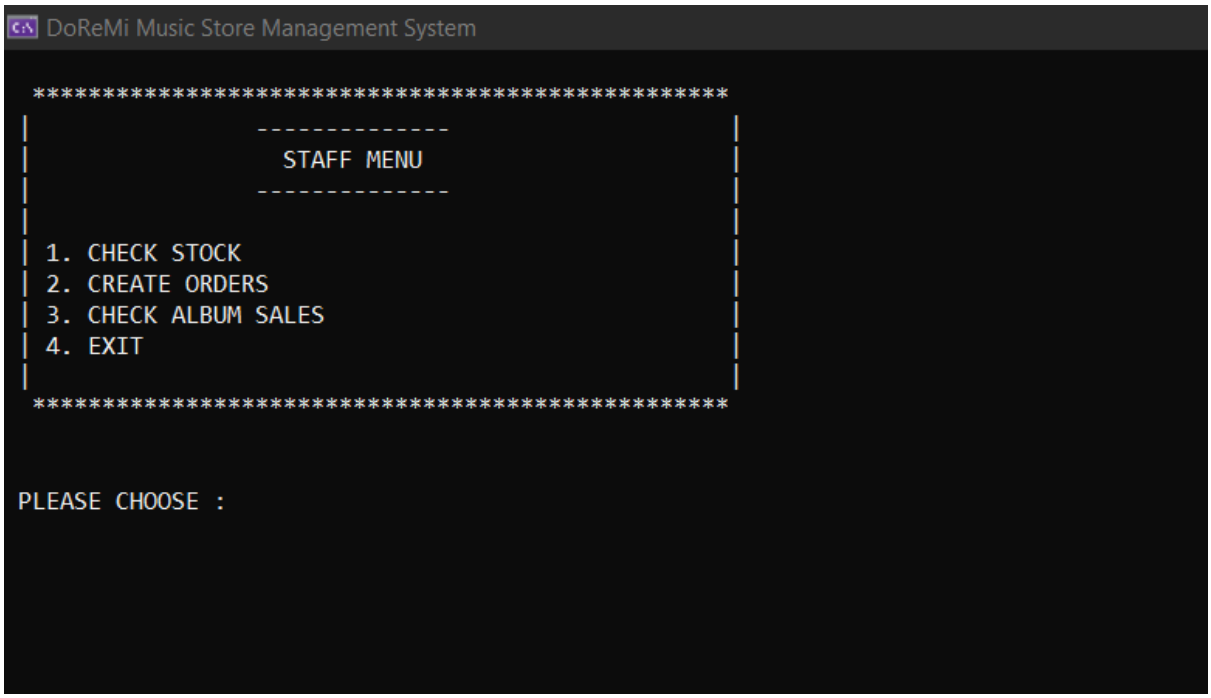

Figure 3.8 : Staff Menu Interface

3.5.5 Check Album Stock

Figure 3.9 shows the system display all of the album stock that is still available in the music store. In this interface, the staff can easily figure out whether the album is out of stock or not.

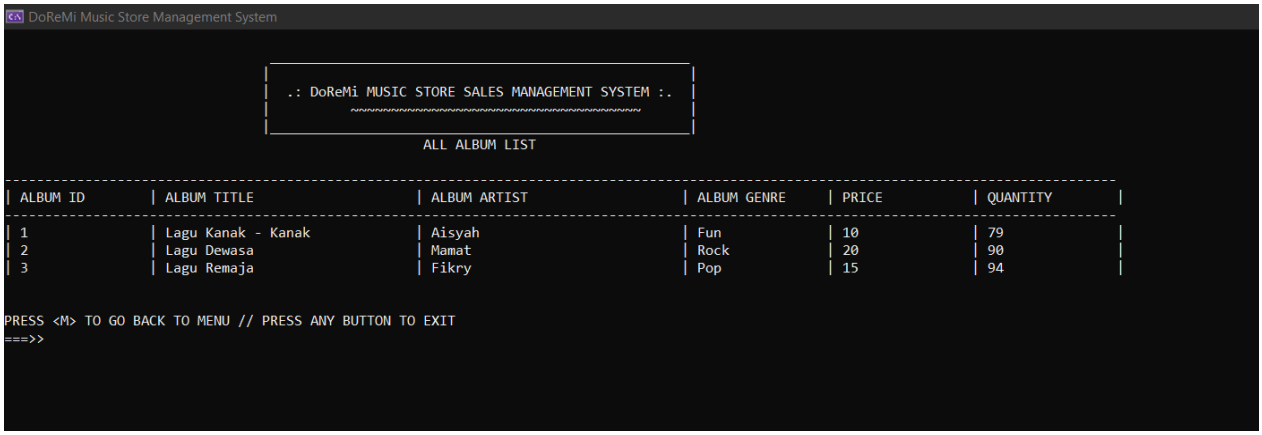

Figure 3.9 : Check Album Stock Interface

3.5.6 Create Order

Figure 3.10 shows a create order interface. The album stock is diplayed on top of the interface to alert staff whether the album is available or not. Staff need to insert the album ID to add order. To proceed to transaction, user need to press ‘Q’ to exit order and then press ‘B’ to buy album. To remove order, staff need to press ‘E’.

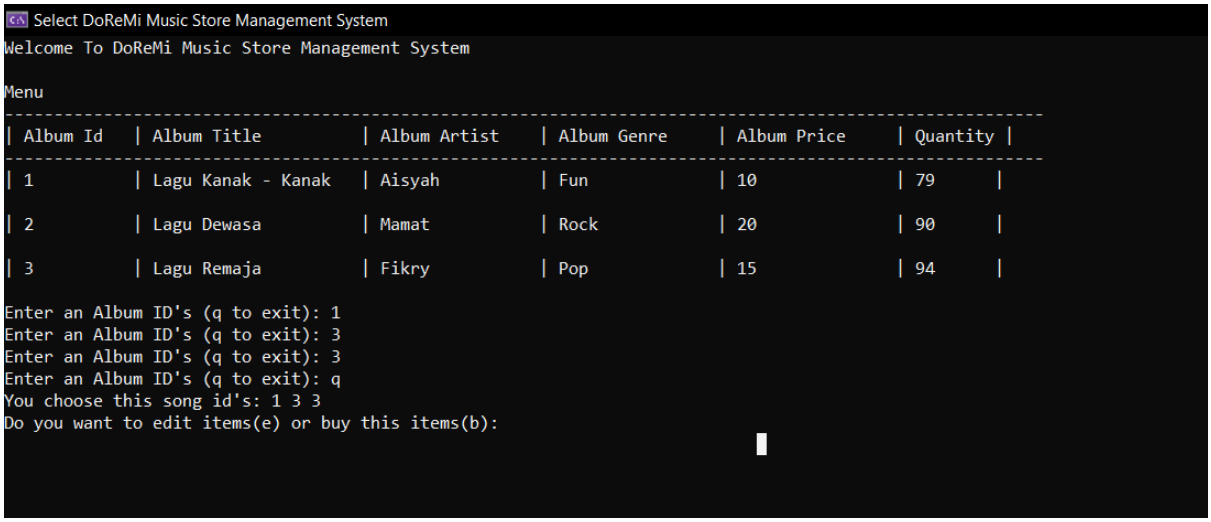

Figure 3.10 : Create Order Interface

3.5.7 Sold Album Record

Figure 3.11 shows the sold album record for the music store. In this interface, staff will be able to see the record of every transaction made.

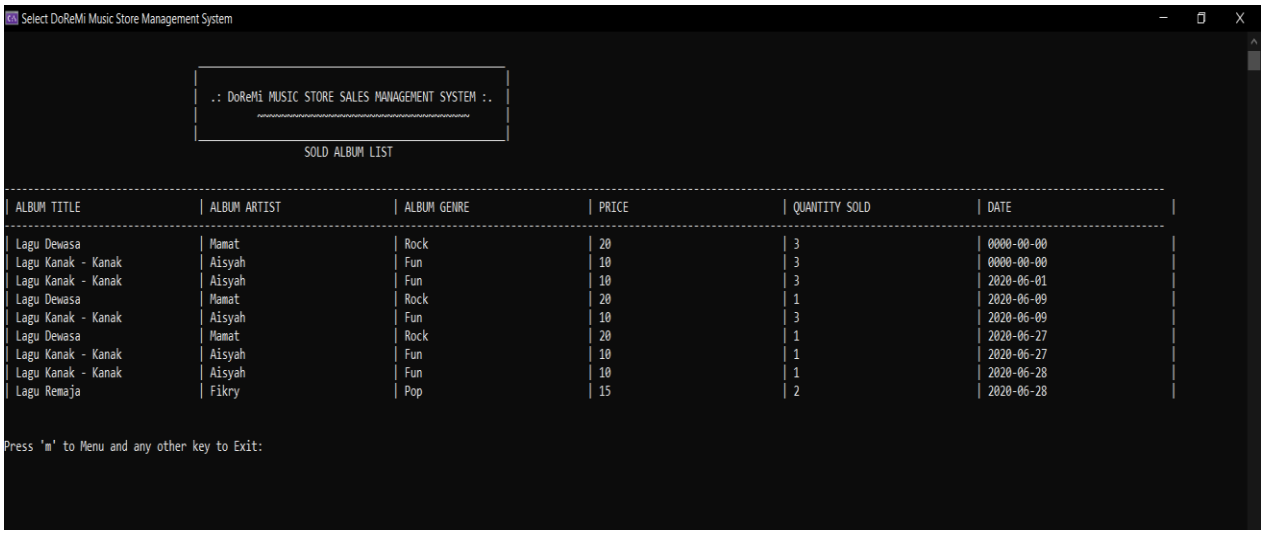

Figure 3.11 : Sold Album Record Interface

3.5.8 Manager Login

Figure 3.12 shows the login menu interface where the manager needs to login using their Manager ID and password before using the system.

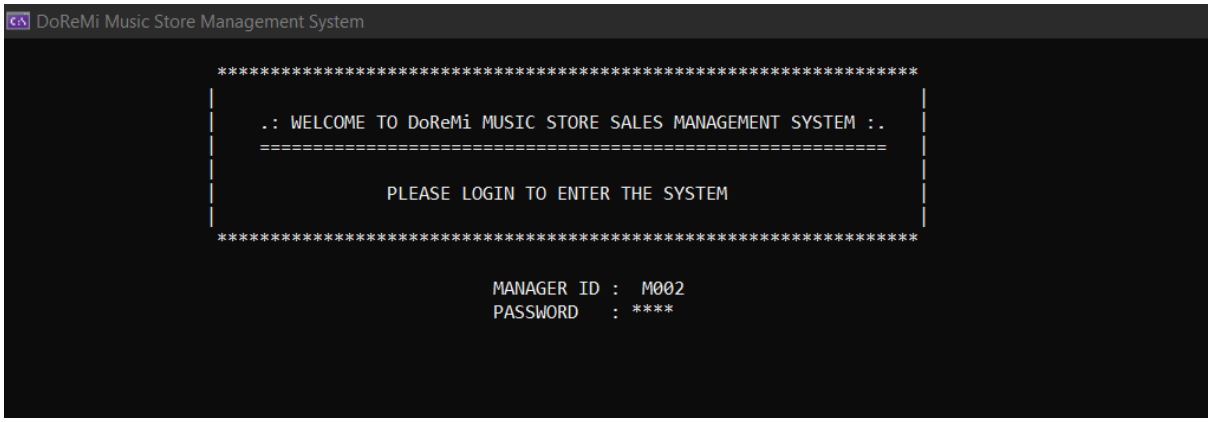

Figure 3.12 : Manager Login Interface

3.5.9 Manager Menu

Figure 3.13 shows the Manager Menu interface. There are four functions that the manager can access in this system which is Add Stock, Delete Stock, Edit Stock Information and Display Item.

Figure 3.13 : Manager Menu Interface

3.5.10 Add Stock

Figure 3.14 shows the interface where the manager can add stock into the system. The manager needs to insert new album title, artist, genre, price and quantity.

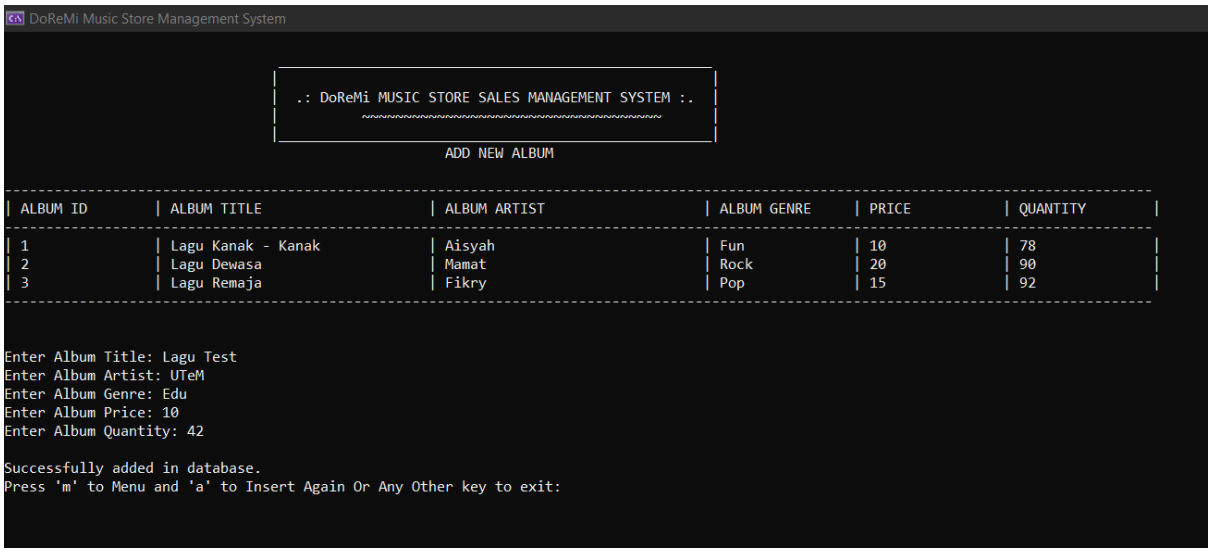

Figure 3.14 : Add Stock Interface

3.5.11 Delete Stock

Figure 3.15 shows the delete stock interface. All of the stock that is available in the system will be displayed on top of the interface. In order to delete stock, the manager need to insert an album ID manager wants to delete.

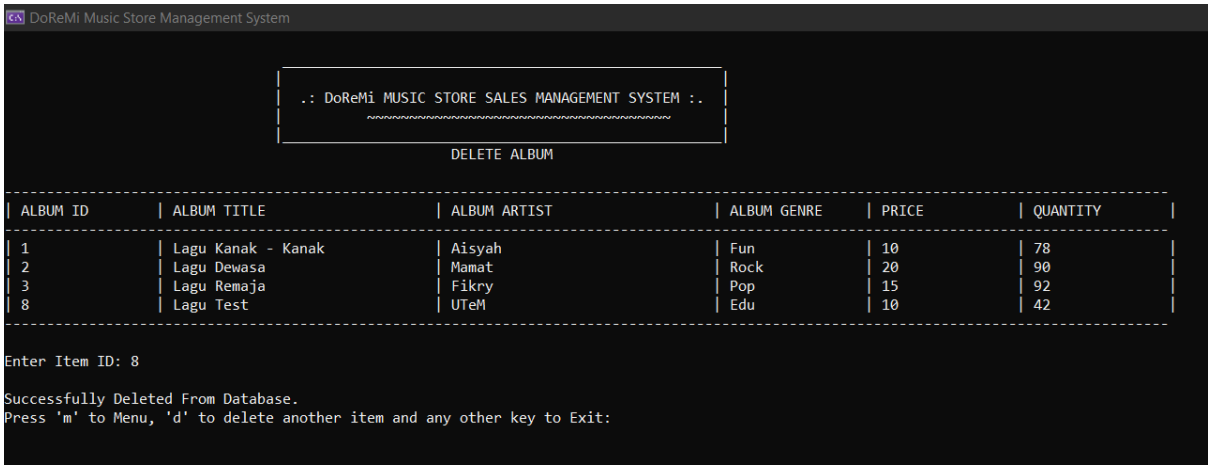

Figure 3.15 : Delete Stock Interface

3.5.12 Edit Album Information

Figure 3.16 shows an interface where the manager can edit album information. The manager needs to choose album to change its information by choosing the album ID. Then, the selected album will be displayed. After that, the manager can insert new title, artist, genre, price and quantity. If the manager does not want to change the information, manager needs to insert '-'.

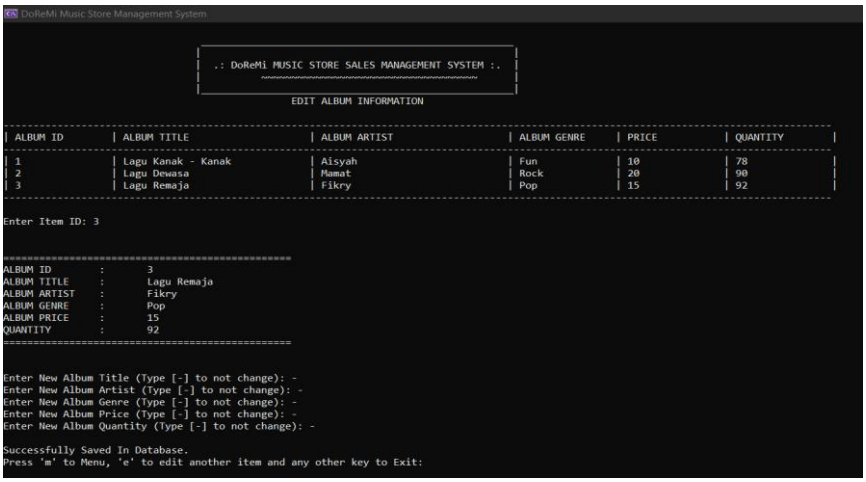

Figure 3.16 : Edit Album Information Interface

3.5.13 Display Items

Figure 3.17 shows the system display all of the album stock that is still available in the music store. In this interface, the manager can easily figure out whether the album is out of stock or not.

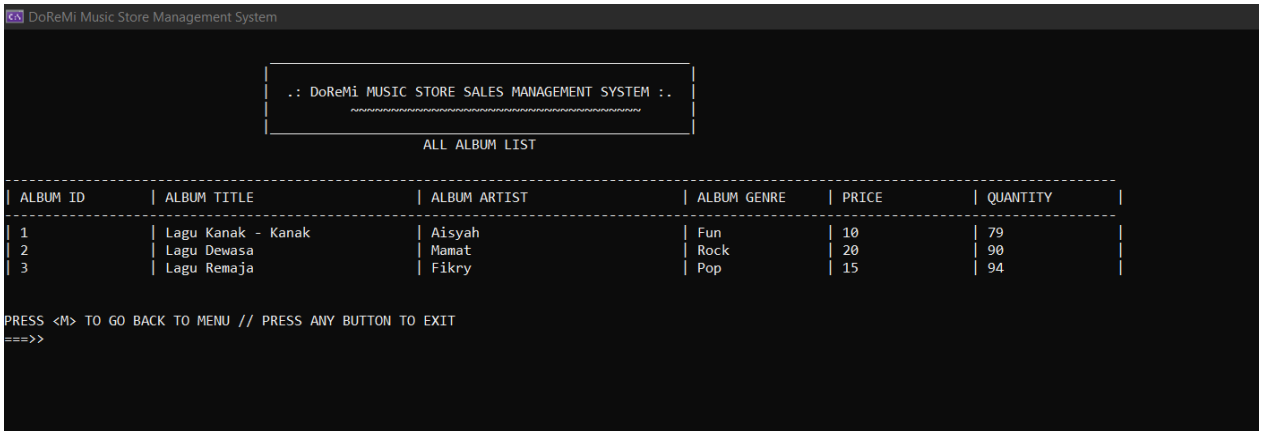

Figure 3.17 : Display Item Interface

3.5.14 Admin Login

Figure 3.18 shows the login menu interface where the admin needs to login using their Admin ID and password before using the system.

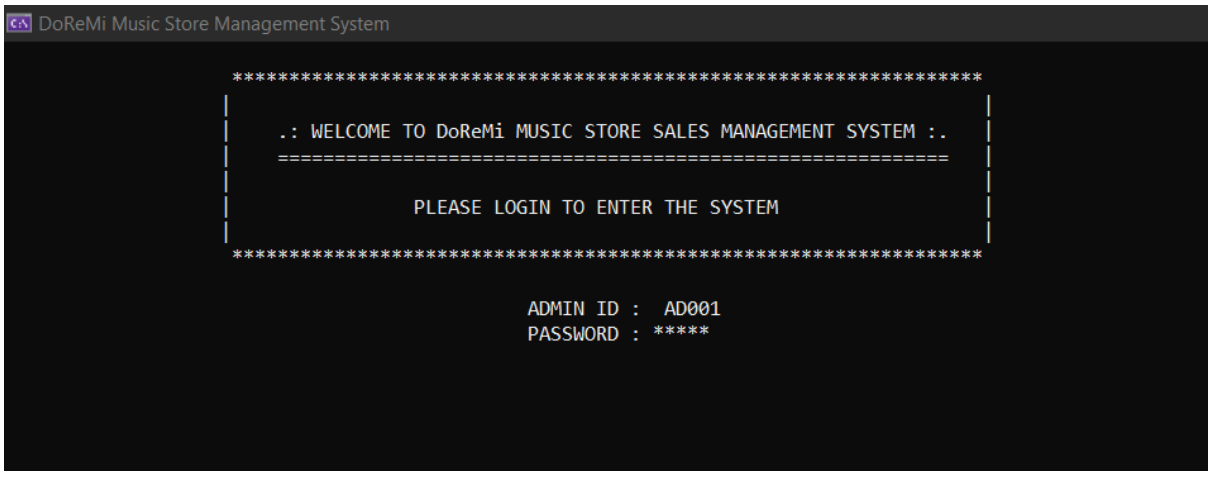

Figure 3.18 : Admin Login Interface

3.5.15 Admin Menu

Figure 3.19 shows the Admin Menu interface. There are three functions that admin can access in this system which is New Staff Registration, Change New Manager and Delete Staff Information.

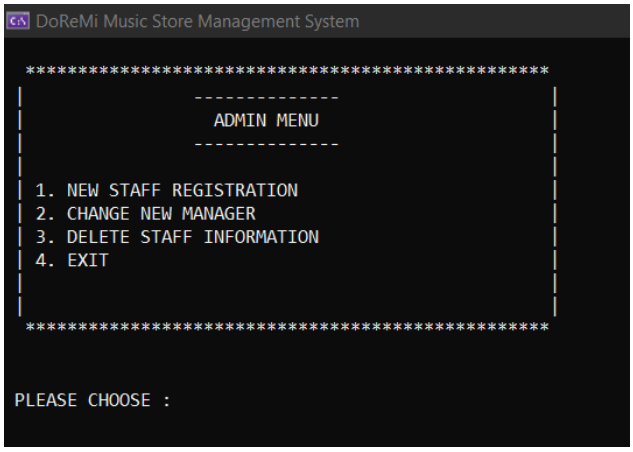

Figure 3.19 : Admin Menu

3.5.16 New Staff Registration

Figure 3.20 shows an interface where admin can register new staff. Admin needs to insert new Staff ID, password, first name and last name in order to register new staff.

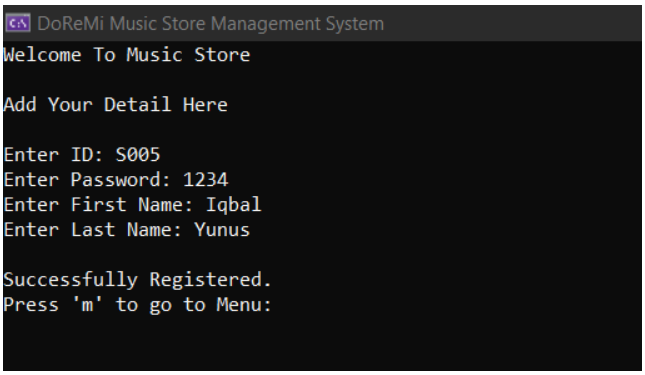

Figure 3.20 : New Staff Registration Interface

3.5.17 Change New Manager

Figure 3.21 shows an interface where admin can change manager details when new manager is hired. Admin can change the details by inserting new manager ID, password, first name and last name of the new manager.

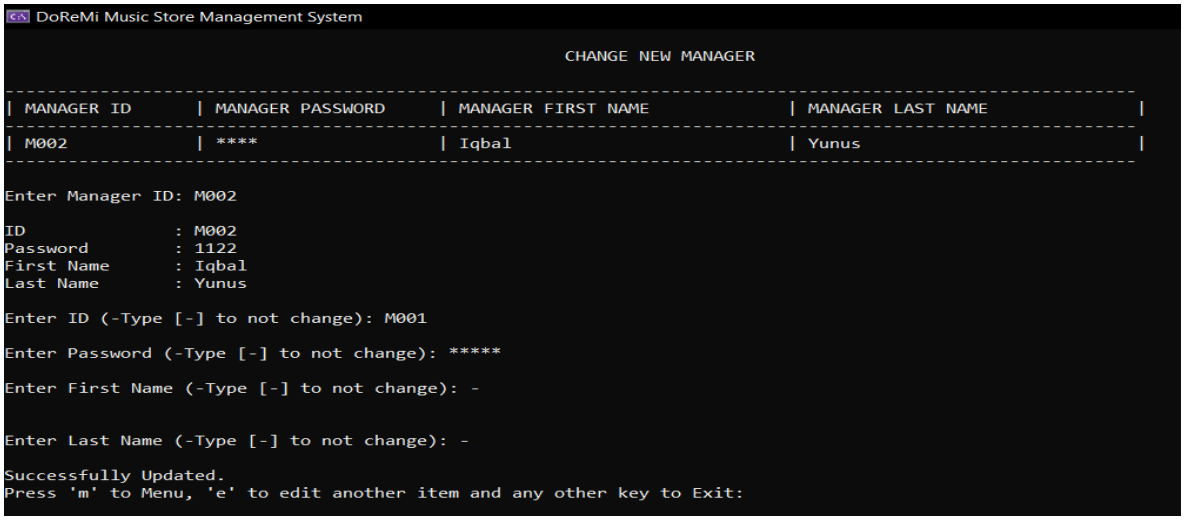

Figure 3.21 : Change New Manager Interface

3.5.18 Delete Staff Information

Figure 3.22 shows the delete staff information interface. All of the staff information that is working in the music store will be displayed on top of the interface. In order to delete staff information, admin need to insert the staff ID that admin wants to delete.

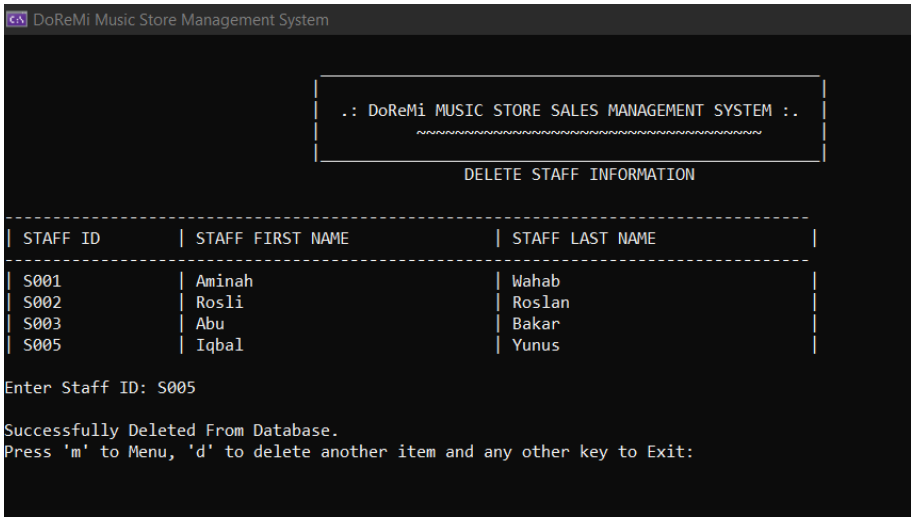

Figure 3.22 : Delete Staff Information Interface

CHAPTER 4

IMPLEMENTATION

4.1 INTRODUCTION

Implementation is the phase to test the system if there is any faulty and focusing on the programming and also testing the system. There are variety of programming codes that being used in this system to perform variety of duties as directed. In this system, there are four main code implementation which insert, delete, update and select. This code is important to ensure that the information system is operate and to be used.

4.2 CODING IMPLEMENTATION

For this system, there are several codes need to compile to make the system functioning well. These codes are required to solve any faulty occur on the system.

INSERT

The ‘insert’ statement is used to register new staff. This code also is being used in this system to insert new item into the database. In figure 4.1 shows the ‘insert’ code for adding new item into the database.

```
cout << endl << endl;
cin.ignore(1, '\n');
cout << "Enter Album Title: ";
getline(cin, title);
cout << "Enter Album Artist: ";
getline(cin, artist);
cout << "Enter Album Genre: ";
getline(cin, genre);
cout << "Enter Album Price: ";
cin >> price;
cout << "Enter Album Quantity: ";
cin >> quantity;

stringstream streamPrice, streamQuan;
string sprice, squan;
streamPrice << price;
streamPrice >> sprice;
streamQuan << quantity;
streamQuan >> squan;


string insert_query = "insert into album_table (album_title, album_artist, album_genre, album_price, album_quantity) values ('" +
    title + "','" + artist + "','" + genre + "','" + sprice + "','" + squan + "')";

const char* q = insert_query.c_str();

qstate = mysql_query(conn, q);

if (!qstate)
{
    cout << endl << "Successfully added in database." << endl;
}
else
{
    cout << "Query Execution Problem!" << mysql_errno(conn) << endl;
}
```

Figure 4.1 : Code for Adding New Item into Database

DELETE

The delete statement is used to delete item from the database. This system also can delete staff information. From figure 4.2 below shows code to delete staff information.

```
    cout << endl;
    cout << "Enter Staff ID: ";
    cin >> staffId;
    cout << endl;
}
catch (exception e)
{
    cout << "Please Enter a valid ID." << endl;
    HaveException = true;
    goto ExitMenu;
}

if (HaveException == false)
{
    stringstream streamid;
    string strid;
    streamid << staffId;
    streamid >> strid;

    for (int i = 0; i < indexForId; i++)
    {
        if (strid != staff[i])
        {
            NotInDatabase = true;
        }
        else
        {
            NotInDatabase = false;
            break;
        }
    }

    if (NotInDatabase == false)
    {
        string delete_query = "delete from staff_table where staff_id = '" + strid + "'";
        const char* qd = delete_query.c_str();
        qstate = mysql_query(conn, qd);

        if (!qstate)
        {
            cout << "Successfully Deleted From Database." << endl;
        }
        else
        {
            cout << "Failed To Delete!" << mysql_errno(conn) << endl;
        }
```

Figure 4.2 : Code to Delete Staff Information

UPDATE

The update statement is used to update existing item in the database. In this system, user can update item information in the database. In figure 4.3 shows the code for update item information.

```
cin.ignore(1, '\n');
cout << "Enter New Album Title (Type [-] to not change): ";
getline(cin, title);
if (title == "-")
{
    title = storetitle;
}
cout << "Enter New Album Artist (Type [-] to not change): ";
getline(cin, artist);
if (artist == "-")
{
    artist = storeartist;
}
cout << "Enter New Album Genre (Type [-] to not change): ";
getline(cin, genre);
if (genre == "-")
{
    genre = storegenre;
}
cout << "Enter New Album Price (Type [-] to not change): ";
cin >> price;
if (price == "-")
{
    price = storeprice;
}
cout << "Enter New Album Quantity (Type [-] to not change): ";
cin >> quantity;
if (quantity == "-")
{
    quantity = storequantity;
}

string update_query = "update album_table set album_title = '" + title + "', album_artist = '" + artist + "', album_genre = '" +
    genre + "', album_price = '" + price + "', album_quantity = '" + quantity + "' where album_id = '" + strid + "'";
const char* qu = update_query.c_str();
qstate = mysql_query(conn, qu);

if (!qstate)
{
    cout << endl << "Successfully Saved In Database." << endl;
}
else
{
    cout << "Failed To Update!" << mysql_errno(conn) << endl;
}
```

Figure 4.3 : Code for Update Item Information

SELECT

The select statement is used to select data from the table in the database. For example, the select statement is used for the user to login into the system as shown in figure 4.4.

```
    cout << "\n                    |    ============================================================   |";
    cout << "\n                    |                                                                 |";
    cout << "\n                    |               PLEASE LOGIN TO ENTER THE SYSTEM                  |";
    cout << "\n                    |                                                                 |";
    cout << "\n                     ***************************************************************** ";
    cout << "\n                                                                    ";
    cout << "\n                                          ADMIN ID :  ";
    cin >> adminid;
    cout << "                                        PASSWORD : ";


p:
    c = _getch();
    cout << "*";
    if (c != 13)
    {
        adminpass = adminpass + c;
        goto p;
    }

    qstate = mysql_query(conn, "select admin_id, admin_password, admin_fname, admin_lname from admin_table");
    if (!qstate)
    {
        res = mysql_store_result(conn);
        while ((row = mysql_fetch_row(res)))
        {

            if (adminid == row[0] && adminpass == row[1])
            {
                system("cls");
                cout << " ================================================ ";
                cout << " \n ACCESS GRANTED!!! WELCOME " << row[2] << " " << row[3];
                cout << "\n ================================================ ";
                std::this_thread::sleep_for(3s);
                MenuAdmin();

            }
        }
```

Figure 4.4 : Code for Login the System

ERROR HANDLING

For this part, DoReMi Music Store Sales Management System use the error handling to prevent users from enter wrong credential to login into the system as admin or staff. The code shows in figure 4.5 below.

```
        char ka;
        system("cls");
        cout << "\n ================================================================================== ";
        cout << "\n ACCESS DENIED!!! USERNAME AND PASSWORD NOT FOUND IN DATABASE!!!";
        cout << "\n ================================================================================== ";
        cout << "\n PRESS m TO GO BACK OR r TO LOGIN AGAIN. \n";
        cin >> ka;
        switch (ka)
        {
        case 'm':
        case 'M':
            system("cls");
            Prompt();
            break;
        case 'r':
        case 'R':
            system("cls");
            AdminLogin();
            break;
        default:
            system("cls");
            cout << " ENTER AGAIN !!!!\n";
            MenuAdmin();
        }
    }
    else
    {
        cout << "\n\n Query Execution Problem!" << mysql_errno(conn) << endl;
    }
}
```

Figure 4.5 : Error Handling in Admin Login Page

4.3 CHAPTER SUMMARY

Overall, implementation applied in this system mostly to display the data from database. All the code use on this system is simple and easy to notify if any problem occurs. The four-code statement use in this system also enable to call or get data easily from the database to make this system work.

CHAPTER 5

CONCLUSION

5.1 INTRODUCTION

During the previous phases, DoReMi Music Store Sales Management System has shown the strengths and limitation observed from the first until the last phase. These strengths and limitation have been analysed to let the developer to improve the system in order to fulfil the user's satisfaction. In the process of generating this system various types of resistance have been passed. Various errors have been found and many new things have been learned to make sure the system can be set up and functioned properly.

5.2 SYSTEM LIMITATION

The limitations of this system are if the customer wants to buy 50 albums at a time, the user which is the staff need to key in the album ID 50 times in order to make 50 orders. Secondly, the system shows all of the album sales from the beginning. So let say that the system existed for 3 years from 2017 until 2020 and the user wants to search sales from July 2018, the user need to search the record one by one.

5.3 SYSTEM STRENGTH

The strengths of this system are all of the transactions made are automatically recorded and saved into database, so the staff do not need to bother record the transaction manually. Secondly, the interface of the system is user – friendly and it is easy to use.

5.4 PROPORTION FOR IMPROVEMENT

Although the system has satisfied the user requirement, it is still possible to improve the system as the user requirement can be changed from time to time. In future, the system should allow user to search sales record by month to simplify their search. Secondly, the system also should improve the transaction system by letting the user insert the quantity of item customer wants to buy instead of insert the item ID one by one. If the system could do that, it would save a lot of time to make a transaction.

5.5 CONCLUSION

The main purpose of DoReMi Music Store Sales Management System is to help staff to record their daily album sales. The development of this system also is to make work easier for the users in managing the music store and to provide a solution to keep track of music albums stock.

REFERENCES

1) W3school.com (1999). [Online] Retrieved on http://www.w3schools.com/cpp

2) Stackoverflow.com (2008) [Online] Retrieved on http://stackoverflow.com/

3) Cplusplus.com (2000) [Online] Retrieved on http://www.cplusplus.com

4) Tutorialspoint [Online] Retrieved on http://turorialspoint.com/cplusplus

5) SoloLearn [Online] Retrieved on https://www.sololearn.com/Course/CPlusPlus/

6) Youtube(2011)[Online]"thenewboston" https://www.youtube.com/watch?v=tvC1WCdV1XU&list=PLAE85DE8440AA6B83