



اونيورسيتي تيكنيكل مليسيا ملاك

**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

PREPARED BY : JEYA JASSVINE A/P JEYABALA SUNDRAM

MATRICS NUMBER : B031910136

SUPERVISOR NAME : TS. DR. SEK YONG WEE

EVALUATOR NAME : TS. DR. NORZIHANI BINTI YUSOF

TITLE : RESTAURANT MANAGEMENT SYSTEM

-----LIST OF CONTENT-----

*Chapter 1*

- 1.1 Introduction
- 1.2 Problem Statement
- 1.3 Background of Project
- 1.4 Objectives
- 1.5 Scope

Chapter 2

- 2.1 Problem Statement
- 2.2 Problems & Solutions
- 2.3 Structured Chart

*Chapter 3*

- 3.1 Flowchart
- 3.2 Pseudocode
- 3.3 ERD
- 3.4 Business Rules
- 3.5 Data Dictionary
- 3.6 I/O
- 3.7 Interface

*Chapter 4*

- 4.1 Programming Techniques
- 4.2 Code Explanation
- 4.3 Calculation
- 4.4 Error Handling

*Chapter 5*

- 5.1 Conclusion
- 5.2 Error Handling

*Appendix*

- Appendix 1 Database

## **Chapter 1**

### **1.1 INTROUDCTION**

Malaysians love for food has made the Food and Beverage sector a fast growing and broad sector here in our country. Our country consists of many different cultures which has resulted in many different cuisines that adapt to different cultural adaptations and even mixed cultures. The Malaysian chain of food outlets is even home to traditional food and beverages even outside of our country.

These days there are many restaurants around to cater for people's different taste buds. This promotes heavy competition between these restaurant owners and it is crucial for them to manage the restaurant systematically.

And as for the customers itself, going through the process of queuing up for a table, navigating through a menu as well as waiting for the food to be ready can be pretty time consuming and frustrating.

However thankfully in our modern world we are able to overcome this problem by means of a simple restaurant management system. The main objective is to cater for the convenience of the customers as well as organized management for the restaurant itself such as improvement of receiving and taking orders and so on.

This management system will consist of a log in module and a registration module, a module to list down menus as well as store orders as well as billing modules. This is a cost efficient way of managing a restaurant system digitally whilst also providing quality services to customers.

## **1.2 PROBLEM STATEMENT**

- People are looking for easier and more convenient methods to order food.
- People are looking for less time consuming methods to order food.
- Companies are looking for organized structures to manage their restaurants.

## **1.3 BACKGROUND OF PROJECT**

The motivation and inspiration that made me choose this project is that these recommendation system has become widely used during the midst of the covid-19 pandemic. Going to a restaurant and navigating through the long menu, waiting for the waiter to place the order and then holding-up till the food gets ready is really hard and time consuming. It can make the customers frustrated and irritated for which a good restaurant should not compromise on the comfort of the customer. So, the goal of this management system is to deal with the customers efficiently. Most of the restaurants possess a single place for all their customers which is surely easier to manage but is not suitable for all kinds of gatherings. Keeping a record of the orders and bills of all the customers along with the priority ones and going through it manually is a tedious task plus it does not ensure correctness. With the help of software based mechanism, these problems can be minimized without compromising accuracy and precision.

## **1.4 OBJECTIVES**

- (i) To build a system that allows customers to choose from a few range of recommendations to suit their convenience.
- (ii) To manage the restaurant better and avoid hassle.
- (iii) For the customers to be able to access and edit menu, order and billing details.

## **1.5 SCOPE**

Modules

Admin :

- LOG IN
- INSERT NEW MENU
- VIEW LIST OF MENU
- SEARCH MENU
- UPDATE MENU INFORMATIONS
- DELETE MENU

Customers :

- REGISTER
- LOG IN
- INSERT NEW ORDER
- VIEW LIST OF ORDERS
- SEARCH ORDER
- UPDATE ORDER
- DELETE ORDER

Order :

- SHOW THE SUB TOTAL BY EACH MENU ID
- SHOW THE TOTAL THAT CUSTOMER MUST PAY

Ratings :

- CUSTOMER CAN INSERT RATING
- ADMIN CAN VIEW RATINGS
- ADMIN CAN SEARCH RATINGS

Report :

- ADMIN CAN VIEW MIN & MAX RATES GIVEN BY THE CUSTOMERS

Target audience :

All types of customers and admin is in charge of the whole restaurant. There is only one admin for this system.

## Chapter 2

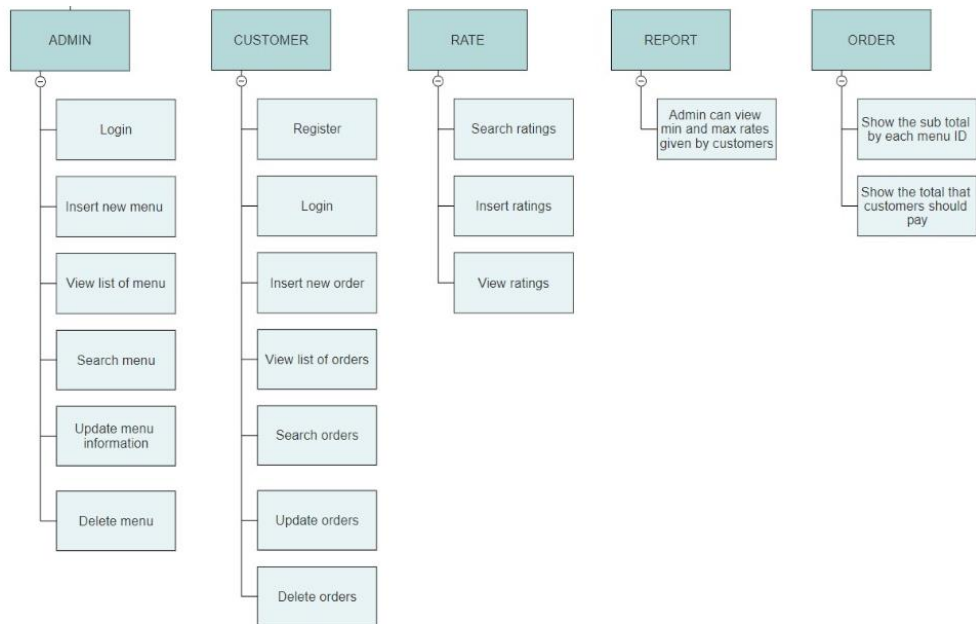
### 2.1 DETAILS OF PROBLEM

Going to a restaurant and navigating through the long menu, waiting for the waiter to place the order and then holding-up till the food gets ready is really hard and time consuming. It can make the customers frustrated and irritated for which a good restaurant should not compromise on the comfort of the customer. So, the goal of this management system is to deal with the customers efficiently. Most of the restaurants possess a single place for all their customers which is surely easier to manage but is not suitable for all kinds of gatherings. Keeping a record of the orders and bills of all the customers along with the priority ones and going through it manually is a tedious task plus it does not ensure correctness. With the help of software based mechanism, these problems can be minimized without compromising accuracy and precision.

### 2.2 PROBLEMS AND SOLUTIONS

PROBLEM	SOLUTION
Too many data to handle on pen and paper	System database created
Too many orders to view	Can filter just by inserting Order ID and Cust ID
Calculation manually is time consuming	Automated calculations

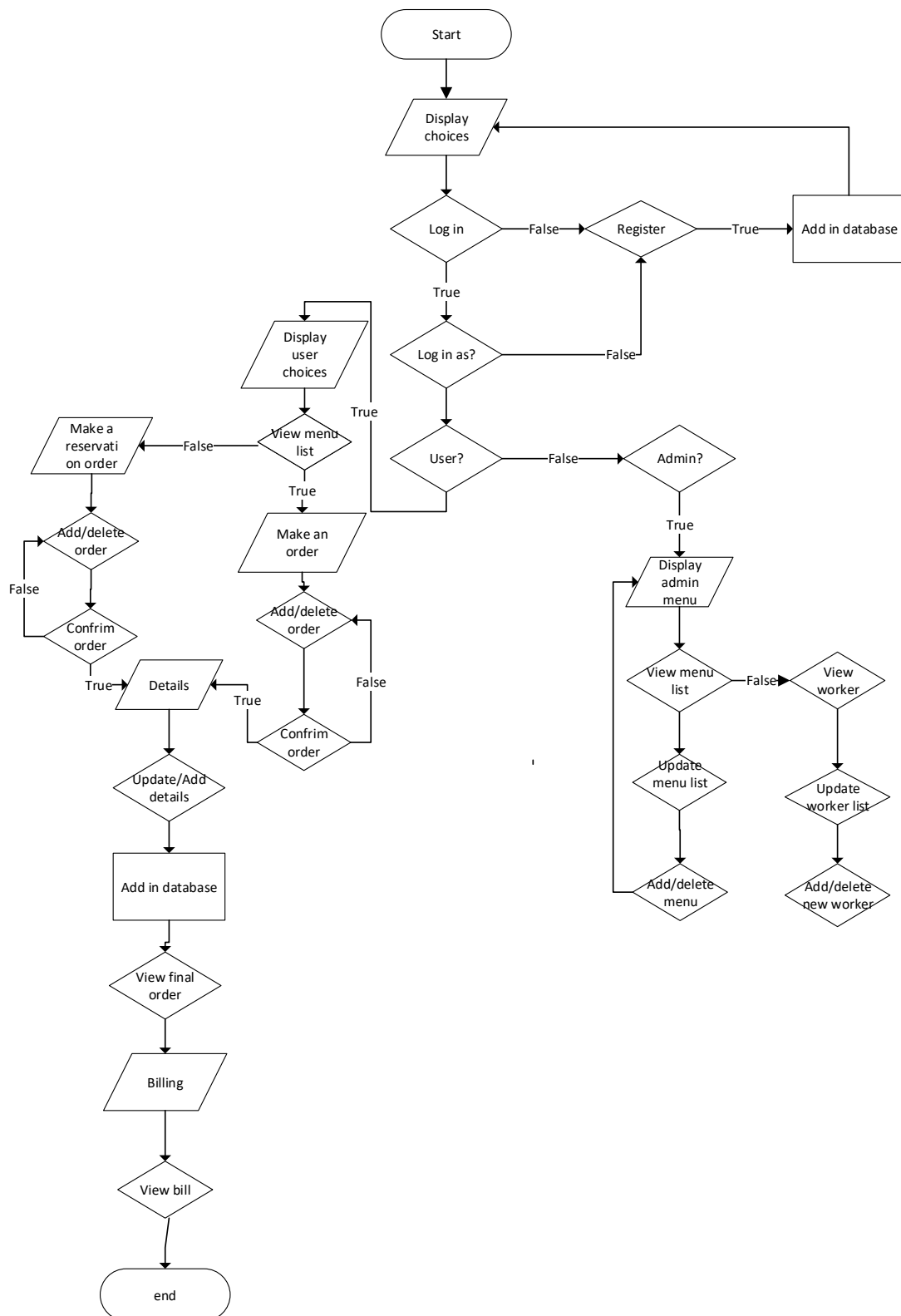
## 2.3 STRUCTURED CHART





## CHAPTER 3

### 3.1 DESIGN OF FLOWCHART



## 3.2 PSEUDO CODE

### ADMIN MODULE

1. Start
2. Login
3. InsertMenu()
  - 3.1 Start
  - 3.2 Add Menu
  - 3.3 AdminMenu()
  - 3.4 End
4. ListMenu()
  - 4.1 Start
  - 4.2 View Menu
  - 4.3 AdminMenu()
  - 4.4 End
5. UpdateDelMenu()
  - 5.1 Start
  - 5.2 If Update
    - 5.2.1 Choose MenuID
    - 5.2.2 Else Delete
    - 5.2.3 ChooseMenuID
  - 5.3 AdminMenu()
  - 5.4 End
6. SearchMenu()
  - 6.1 Search by MenuName
  - 6.2 AdminMenu()
  - 6.3 End

### CUSTOMER MODULE

1. Start
2. Login
3. InsertOrder()
  - 3.1 Start

- 3.2 Add Order
  - 3.3 ustomerMenu()
  - 3.4 End
- 4. ListOrder()
  - 4.1 Start
  - 4.2 View Order
  - 4.3 CustomerMenu()
  - 4.4 End
- 5. UpdateDelOrder()
  - 5.1 Start
  - 5.2 If Update
    - 5.2.1 Choose OrderID
    - 5.2.2 Else Delete
    - 5.2.3 Choose OrderID
  - 5.3 CustomerOrder()
  - 5.4 End
- 6. SearchMenu()
  - 6.1 Search by MenuName
  - 6.2 AdminMenu()
  - 6.3 End

## ORDERING MODULE

- 1. Start
- 2. InsertOrder()
- 3. ViewOrder()
  - 3.1 Display totalprice
- 4. SearchOrder()
  - 4.1 Display totalprice

## RATING MODULE

1. Start
2. Initialize variables to accept customers data
3. Accept user input into variables
4. Allow admin to view Ratings
5. Allow Admin to search Ratings
6. End

## REPORT MODULE

1. Start
2. Allow Admin to view MinRate()
  - 2.1 else MaxRate()
3. End



### 3.5 DATA DICTIONARY

Table	Attribute Name	Content	Type	Form	Required	PK/ FK	FK table reference
Admin	AdminID	Admin ID	Varchar	XXX	Yes	PK	
	AdminUsername	Admin Login Username	Varchar	XXX	Yes		
	AdminPw	Admin Login Password	Varchar	XXX	Yes		
Customer	CustID	Customer ID	Varchar	XXX	Yes	PK	
	CustPhone	Customer Contact	Int	012	Yes		
	CustAddress	Customer Address	Varchar	XXX	Yes		
	CustUsername	Customer Login Username	Varchar	XXX	Yes		
	CustPw	Customer Login Password	Varchar	XXX	Yes		
	CustName	Customer Name	Varchar	XXX	Yes		

Menu	MenuID	Menu's ID	Varchar	XXX	Yes	PK	
	MenuName	Menu's Name	Varchar	XXX	Yes		
	MenuPrice	Menu's Price	Int	012	Yes		
	MenuDescription	Menu's Description	Varchar	XXX	Yes		
Ordering	OrderID	Orders' ID	Int	012	Yes	PK	
	OrderDate	Order's date	Date	Y/m/d	Yes		
	CustID	Customer's ID	Int	012	Yes	FK	Customer
	Quantity	Order's Quantity	Int	012	Yes		
	MenuID	Menu's ID	Int	012	Yes	FK	Menu

Rating	ID	Rating ID	Int	012	Yes	PK	
	Rating	Rating given by Customer	Int	012	Yes		
	Comments	Comments given by customer	Varchar	XXX	Yes		
	CustID	Customer's ID	Int	012	Yes	FK	Customer
	AdminID	Admin's ID	Int	012	Yes	FK	Menu

### 3.6 I/O

#### Input

- Login details & Registration Details

#### Process

- Adding menu, Update/Deleting Menu, View Menu, Search Menu, Input Order, Update/delete Order, View Order, Search Order, save in database

#### Output

- Display all the details fetched from database



### 3.7 INTERFACE

```
WELCOME TO M&M COLLAB RESTAURANT

WE HOPE YOU HAVE A GOOD EXPERIENCE DINING IN OUR RESTAURANT

PLEASE FOLLOW SOP MEASURES AND STAY SAFE!

1. LOG INTO YOUR ACCOUNT
2. REGISTER A NEW ACCOUNT
3. EXIT .

*****

Please enter your choice. What would you like to do? :
```

*Main interface*

```
RESTAURANT MANAGEMENT SYSTEM
*****

HI ADMIN

THIS IS THE MENU SECTION OF M&M COLLAB RESTAURANT T

YOU MAY CHOOSE YOUR TASK
1. INSERT A NEW M&M MENU
2. LIST OF THE AVAILABLE M&M MENU I
3. VIEW RATINGS FROM CUSTOMER
4. REPORTING
5. Exit to MAIN MENU
*****

Please enter your choice :
```

*Admin Main Interface*

```
WELCOME TO M&M COLLAB RESTAURANT

WE HOPE YOU HAVE A GOOD EXPERIENCE DINING IN OUR RESTAURANT

PLEASE FOLLOW SOP MEASURES AND STAY SAFE!

.. YOU CAN MAKE YOUR ORDERS HERE ..

*****

1. TO INSERT A NEW ORDER

2. TO VIEW ORDERS

3. RATE & SUGGEST

*****

Please input your choice : _
```

*Customer Main Interface*

```
RESTAURANT MANAGEMENT SYSTEM

MenuID      Name      Description      Price
2           Nachos Chili Con      Starters      RM 19.00
3           Calamari Rings      Starters      RM 19.00
4           Spicy Chicken Wings      Starters      RM 15.00
5           DEVIL'S CURRY      Local Fav      RM 13.00
6           SEA BASS CURRY      Local Fav      RM 13.00
7           CHILLI PADI CHICKEN      Local Fav      RM 15.00
8           PASTA Oglio Olio      Pasta      RM 17.00
9           PASTA Carbonara      Pasta      RM 13.00
10          Spicy Bacon Pizza      Pizza      RM 17.00
11          Molten Lava      Dessert      RM 10.00
12          Ice cream      DESSERT      RM 10.00

===== Choose (e) (E) to edit Menu <=====
===== Choose (s) (S) to search Menu <=====
===== Choose (m) (M) to head back to Main Menu <=====
```

*Menu listing interface*

```
RESTAURANT MANAGEMENT SYSTEM
You Have Ordered :

=====
Customer ID          Your total for all orders
=====
ID - 10              RM      153
ID - 15              RM      28
ID - 20              RM      51

=====

***** Choose (e) (E) to edit Order <*****

***** Choose (s) (S) to search Order <*****

***** Choose (m) (M) to head back to Customer Menu <*****

PLEASE INPUT YOUR CHOICE :
```

*Total cost to be paid by Customer interface*

```
RESTAURANT MANAGEMENT SYSTEM

MenuID      Date      (Sub) Total Price      Cust ID      Order ID
8           1999-08-03      RM      153           10           8
12          2021-01-18      RM      28           15          21
8           2021-01-09      RM      51           20          23

***** Choose (u) (U) to edit Order *****

***** Choose (d) (D) to delete Order *****
```

*Sub Total to be paid by Customer (By MenuID) interface*

## CHAPTER 4

Herewith attached in this segment are example segments of the techniques implemented

### 4.1 PROGRAMMING TECHNIQUES

- multi way if-else statements

```
res = mysql_store_result(conn);
if (res == NULL) {
    cout << "Username is already exist. Press Enter to Try Again...";
    _getch();
    Register();
}
else
{
    string InsertCustomer_query = "Insert into Customer (CustName, CustPhone, CustAddress, CustUsername, CustPa, CustID) values ('" + name + "', '" + phone + "', '" + address + "', '" + username + "', '" + password + "', '" + CustID + "')";
    cout << res << InsertCustomer_query << endl;
    update = mysql_query(conn, &);
}

if (!update)
{
    cout << endl << "You have been registered. Press Enter to Continue...";
    _getch();
    menu();
}
else
{
    cout << "Query Execution Problem!" << mysql_error(conn) << endl;
}
```

- two way if-else statements

```
cout << "Hi welcome! Choose (1) (1) to add New Customer ";
cout << "Hi welcome! Choose (2) (2) to search New Customer ";
cout << "Hi welcome! Choose (3) (3) to go back to Main Menu screen ";
cin >> choice;
if (choice == '1' || choice == '2')
    InsertNewMenu();
else if (choice == '3' || choice == '0')
    SearchMenu();
else if (choice == '4' || choice == '5')
    ShowMenu();
```

- nested if

```
if (!update)
{
    res = mysql_store_result(conn);
    if (res == NULL) {
        cout << "Username is already exist. Press Enter to Try Again...";
        _getch();
        Register();
    }
    else
    {
        string InsertCustomer_query = "Insert into Customer (CustName, CustPhone, CustAddress, CustUsername, CustPa, CustID) values ('" + name + "', '" + phone + "', '" + address + "', '" + username + "', '" + password + "', '" + CustID + "')";
        cout << res << InsertCustomer_query << endl;
        update = mysql_query(conn, &);
    }

    if (!update)
    {
        cout << endl << "You have been registered. Press Enter to Continue...";
        _getch();
        menu();
    }
    else
    {
        cout << "Query Execution Problem!" << mysql_error(conn) << endl;
    }
}
else
{
    cout << "Query Execution Problem!" << mysql_error(conn) << endl;
}
```

- switch

```
void CustomerMenu()
{
    system("cls");
    cout << endl;
    cout << setw(30) << "***** Welcome to our online restaurant *****" << endl;
    cout << setw(30) << "***** WE HOPE YOU HAVE A GOOD EXPERIENCE DINING IN OUR RESTAURANT*****" << endl;
    cout << endl;
    cout << setw(30) << "***** PLEASE FOLLOW OUR PROGRESS AND ORDER YOURS *****" << endl;
    cout << setw(30) << "***** YOU CAN MAKE YOUR ORDER HERE *****" << endl;
    cout << endl;
    cout << setw(30) << "***** 1. TO INSERT A NEW ORDER *****" << endl;
    cout << setw(30) << "***** 2. TO VIEW ORDER*****" << endl;
    cout << setw(30) << "***** 3. MAKE A SUBMIT*****" << endl;
    cout << endl;
    cout << "Please Input your choice : ";
    cin >> choice;
    switch (choice)
    {
        case 1:
            InsertNewOrder(); //function to insert new data
            break;
        case 2:
            ListOrders();
            break;
        case 3:
            InsertCustomerData();
            break;
        default:
            cout << "Please choose between 1 - 3. Press Enter to Continue...";
            _getch(); // get user // pause console //
            system("cls"); // clear console //
            CustomerMenu(); //
    }
}
```

### 4.1.2 CONTROL TECHNIQUE

- While loop

```

        cout << "Is ***** Choice (a) (i) to add Order *****?";
        cout << "Is ***** Choice (d) (i) to delete Order *****?";
        cin >> choose;

    } while (choose != "a" && choose != "d" && choose != "e" && choose != "0");

    cout << "Enter Order ID: ";
    cin >> OrderID;

```

- Do while loop

```

    char chance;
    do
    {
        cout << "Do you want add another word? (y/n) ";
        cin >> chance;
        if (chance == "y" || chance == "Y")
        {
            InsertWord(word);
        }
        else if (chance == "n" || chance == "N")
        {
            ExitWordPhone();
        }
    } while (chance != "y" || chance == "Y" || chance != "n" || chance == "N");
}

```

### 4.1.3 FUNCTIONS

- Function declaration

```

42
43
44 void Register();
45 void Login();
46 void AdminMenu();
47 void CustomerMenu();
48 void InsertMenu();
49 void ListMenu();
50 void UpdateMenu();
51 void SearchMenu();
52 void ListMenu();
53 void InsertMenu();
54 void ListMenu();
55 void UpdateMenu();
56 void SearchMenu();
57 void InsertCustomerRate();
58 void ViewCustomerRate();
59 void report();
60 void Menu();
61 void SearchRate();
62 void MenuRate();
63 void MenuRate();

```

- Function definition

```
int main()

{
    system("cls");

    system("title Welcome to Account Management System");
    system("color 0a");

    do_request(connectionFunction());

    int chooseDefFuncMenu = 0;

    cout << setw(80) << "===== Welcome to emdi =====\n";

    cout << setw(80) << "1. LOGIN TO VIEW CLIENT RECORDS\n";
    cout << setw(80) << "2. WE HOPE YOU HAVE A GOOD EXPERIENCE SERVING OUR CUSTOMERS\n";
    cout << "PLEASE FOLLOW OUR REQUESTS AND STAY SAFE!\n";
    cout << setw(80) << "3. LOG INTO YOUR ACCOUNT Now";
    cout << setw(80) << "4. REGISTER A NEW ACCOUNT Now";
    cout << setw(80) << "5. EXIT ->end\n";

    cout << setw(80) << "===== Welcome to emdi =====\n";

    cout << "Please enter your choice. What would you like to do : ";

    cin >> chooseDefFuncMenu;

    switch (chooseDefFuncMenu)
    {
        case 1:
        {
            login();
            break;
        }
        case 2:
        {
            Register();
            break;
        }
        case 3:
        {
            cout << "Program will exit now...bye!" << endl;
            exit(0);
        }
        default:
        {
            cout << "Please choose between 1 - 5. Press Enter to Continue...";
            _getch(); // get char // space console //
            system("cls"); // clear console
            main(); //
            break;
        }
    }
}
```

- Function calling

```

90 switch (chooseDefFromMenu)
91 {
92     case 1:
93         login();
94     break;
95     case 2:
96         register();
97     break;
98     case 3:
99         cout << "Program will exit now...bye!" << endl;
100         exit(0);
101     default:
102         cout << "Please choose between 1 - 3. Press Enter to Continue..."
103             << endl; // get char // guess console //
104             system("cls"); // clear console
105             main(); //
106             break;
107 }

```

## 4.2 CODE EXPLANATION

### - Main

#### a. Database connection

```
//main.cpp
#include <iostream>
using namespace std;

//function to connect to database
void connectDB() {
    //connect to database
    cout << "Please enter your username : ";
    string username;
    while (true) {
        if (username.empty()) {
            cout << "Please enter your username : ";
            continue;
        }
        //check if user is admin or customer
        if (username == "admin") {
            //admin login
            cout << "Please enter your password : ";
            string password;
            while (true) {
                if (password.empty()) {
                    cout << "Please enter your password : ";
                    continue;
                }
                //check if password is correct
                if (password == "12345") {
                    //admin login successful
                    cout << "Welcome to the Restaurant Management System\n";
                    return;
                }
                //password incorrect
                cout << "Invalid password. Please try again.\n";
                password = "";
            }
        } else {
            //customer login
            cout << "Please enter your email : ";
            string email;
            while (true) {
                if (email.empty()) {
                    cout << "Please enter your email : ";
                    continue;
                }
                //check if email is correct
                if (email == "customer@gmail.com") {
                    //customer login successful
                    cout << "Welcome to the Restaurant Management System\n";
                    return;
                }
                //email incorrect
                cout << "Invalid email. Please try again.\n";
                email = "";
            }
        }
        //username incorrect
        cout << "Invalid username. Please try again.\n";
        username = "";
    }
}

//main function
int main() {
    connectDB();
    return 0;
}
```

Diagram above shows the connection of c++ Visual Studio with the database Restaurant

#### b. Login

```
//login.cpp
#include <iostream>
using namespace std;

//function to login
void login() {
    //connect to database
    cout << "Please enter your username : ";
    string username;
    while (true) {
        if (username.empty()) {
            cout << "Please enter your username : ";
            continue;
        }
        //check if user is admin or customer
        if (username == "admin") {
            //admin login
            cout << "Please enter your password : ";
            string password;
            while (true) {
                if (password.empty()) {
                    cout << "Please enter your password : ";
                    continue;
                }
                //check if password is correct
                if (password == "12345") {
                    //admin login successful
                    cout << "Welcome to the Restaurant Management System\n";
                    return;
                }
                //password incorrect
                cout << "Invalid password. Please try again.\n";
                password = "";
            }
        } else {
            //customer login
            cout << "Please enter your email : ";
            string email;
            while (true) {
                if (email.empty()) {
                    cout << "Please enter your email : ";
                    continue;
                }
                //check if email is correct
                if (email == "customer@gmail.com") {
                    //customer login successful
                    cout << "Welcome to the Restaurant Management System\n";
                    return;
                }
                //email incorrect
                cout << "Invalid email. Please try again.\n";
                email = "";
            }
        }
        //username incorrect
        cout << "Invalid username. Please try again.\n";
        username = "";
    }
}

//main function
int main() {
    login();
    return 0;
}
```

Diagram above shows the Login of both customers and admin

### c. Register

```
#! Register()

system("cls");
string name, phone, address, username, password; CoutID;

cout << info[0] << "***** Welcome To Our Online Registration ***** \n\n"; // endl;
cout << info[1] << "      WE HOPE YOU HAVE A GOOD EXPERIENCE DURING IN OUR APPLICATION*****\n\n";
cout << info[2] << "          PLEASE FOLLOW THE MOMENTS AND START LAUNCH 'Welcome'\n\n";
cout << info[3] << "      THIS IS YOUR REGISTRATION FORM, PLEASE FILL IT IN WITH YOUR DETAILS 'Name',\n\n";
cout << info[4] << "          Thank you!! 'Welcome'*****\n\n"; // endl;
cout << "NAME : ";
cin.ignore(); // '\n';
getline(cin, name);
cout << "CONTACT NUMBER : ";
getline(cin, phone);
cout << "ADDRESS : ";
getline(cin, address);
cout << "CUSTOMER ID : ";
getline(cin, CustomerID);
cout << "Create your unique CUSTOMER ID : ";
getline(cin, CoutID);
cout << "ENTER YOUR PASSWORD : ";
char ch;
while ((ch = getch()) != '\n')
{
    password += ch;
    cout << "\a";
}

string checker_query = "select * from Customer where CustomIDname = '" + username + "'";
cout << "Enter q -> checker_query_s_upd();
qstate = mysql_query(con, sql);

if (!qstate)
{
    res = mysql_store_result(con);
    if (res->num_rows == 1)
        {
            cout << "Username is already exist. Press Enter to Try Again...";
            getch();
            Register();
        }
    else
    {
        string InsertCustomer_query = "insert into Customer (CustomID, ContactNo, CustAddress, CustomIDname, CoutPhy, CoutID) values ('" + name + "','" + phone + "','" + address + "','" + username + "','" + password + "','" + CoutID + "')";
        cout << "Enter q -> InsertCustomer_query_s_upd();
        qstate = mysql_query(con, sql);

        if (!qstate)
        {
            cout << endl << "You have been registered. Press Enter to Continue...";
            getch();
            main();
        }
        else
        {
            cout << "Query Execution Problem!" << mysql_error(con) << endl;
        }
    }
}
else
```

Diagram above shows the registration of customers

- Admin

a. Admin Menu

[illegible]

Diagram above shows the code of Admin Main Menu

### b. Admin Insert Menu

```

insertMenu(menu)
{
    update("%[a?]",
        string name, description, price);

    cout << "Insert Insert New Menu" << endl;
    cin >> name, "%[a?]",
    cout << "Name of menu : ";
    getline(cin, name);
    cout << "Price of menu : ";
    getline(cin, price);
    cout << "Description of menu : ";
    getline(cin, description);

    //insert menu in default data format - menu

    string insert_query = "Insert into Menu (MenuName,MenuDescription,MenuPrice) values ('" + name + "','" + description + "','" + price + "','" + price + "');";
    cout << "Insert query : " << insert_query << endl;
    update = exec(insert_query, db);

    if (update) //query return false
    {
        cout << endl << "Menu is successful added to database." << endl;
    }
    else
    {
        cout << "Query execution problem" << endl;
    }
}

//for delete
del
{
    cout << "Do you want add another menu? (y/n) : ";
    cin >> choice;
    if (choice == "y") // choice == "y"
    {
        insertMenu(menu);
    }
    else if (choice == "n") // choice == "n"
    {
        addMenu();
    }
} while (choice != "y" && choice != "n" && choice != "Y" && choice != "N");

}

//for update
updateMenu("%[a?]",

```

Diagram above shows the code of Admin inserting new menu

### c. Admin Update/Delete Menu

[illegible]

Diagram above shows the code of admin to insert and delete menu

#### d. Admin View

```

void ListMenu()
{
    system("cls");
    setlocale(LC_ALL, "Russian");
    select MenuID, MenuName, MenuDescription, MenuPrice from Menu;

    cout << endl << "Header";
    cout << setw(31) << "MenuID" << setw(30) << "Menu" << setw(31) << " " << setw(31) << "Description" << setw(30) << " " << setw(31) << "Price" << endl;
    for (int i = 0; i < MenuID; i++)
    {
        // body table
        int a = MenuID - MenuID[i];
        while (true = MenuID[i], MenuPrice[i]) // check every row until the end of data
        {
            cout << setw(31) << MenuID[i] << setw(37) << Menu[i] << setw(37) << Menu[i] << Menu[i] << Menu[i] << Menu[i] << Menu[i] << Menu[i] << endl;
        }
    }

    cout << endl;
    cout << "1" << "===== Choose (0) to add Menu =====>";
    cout << "2" << "===== Choose (1) to delete Menu =====>";
    cout << "3" &;< "===== Choose (2) to update Menu =====>";
    cout << "4" << "===== Choose (3) to find back to main Menu =====>";

    int n = choose;
    if (choose == "0" || choose == "1")
    {
        UpdateMenu();
    }
    else if (choose == "2" || choose == "3")
    {
        SearchMenu();
    }
    else if (choose == "4")
    {
        choose == "0";
    }
}

```

Diagram above shows the code for admin to view the list of menu

### e. Admin Search

[illegible]

Diagram above shows the code to search a specific menu by MenuName

- Customer

### a. Customer Menu

[illegible]

*Diagram above shows the Main Customer Menu*

### b. Customer Insert Order

[illegible]



[illegible]

Diagram above shows the code for Customer to insert a new order

### c. Customer Update/Delete Order

[illegible]

```

end if (Order Not Received ?)
    generateNulls, nulls();
    using delete_query = "delete ordering set NumID = ?" where OrderID = ?" + OrderID + " ?"; (changing order id
    const auto & delete_query = del();
    update & delete_query(con, 0);
}
else if (chanceRate == 2)
{
    end if (chanceRate == 2)
    generateNulls, nulls();
    using delete_query = "delete ordering set Quantity = ?" + quantity + " ?" where OrderID = ?" + OrderID + " ?"; (changing order qty
    const auto & delete_query = del();
    update & delete_query(con, 0);
}
else if (chanceRate == 4)
{
    end if (chanceRate == 4)
    generateNulls, nulls();
    using delete_query = "delete ordering set OrderDate = ?" + date + " ?" where OrderID = ?" + OrderID + " ?"; (changing date
    const auto & delete_query = del();
    update & delete_query(con, 0);
}
else if (chanceRate == 8)
{
    generateNulls();
    deleteOrder();
}
else (true)
{
}
}

else if (chance == "0" || chance == "W")
{
    clear(chance);
}
else
{
    end if (Take you want to remove your Order (quit) ?);
    quit = chance();
    if (chance == "Y" || chance == "W")
    {
        using delete_query = "delete from ordering where OrderID = ?" + OrderID + " ?";
        const auto & delete_query = del();
        update & delete_query(con, 0);
        if (update)
        {
            end if ("The Order has been removed. Press Enter to Continue...");
            getch();
            clearOrderall();
        }
        else
        {
            deleteOrderall();
        }
        end if ("Query Execution Problem" or append_error_message == end);
    }
}
} while (chance != "Y" || chance != "W" || chance != "W" || chance != "W");
}

```

Diagram above shows the code for customer to update or delete their orders

#### d. Customer View Order

[illegible]

### e. Customer Search Order

```

1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
182
```

## -Rating

a. Customer Insert Rate

[illegible]

## b. Admin View Rate

[illegible]

Diagram above shows the code for admin to view ratings

### c. Admin Search Rate

[illegible]

- Report

a. Admin view Minimum Rate

[illegible]

Diagram above shows the code for Admin to view Minimum Rate (report)

b. Admin view Maximum Rating

[illegible]

Diagram above shows the code for Admin to view Maximum Rate (report)

### 4.3 CALCULATION

```

//first set "Your unique CUSTOMER ID is : "
cout << "You Have Ordered :VA/VA";
cout << "-----";
cout << left << setw(18) << "Customer ID" << right << setw(80) << right << setw(11) << endl;
cout << "-----";

//first set setw(11) << "OrderID" << setw(28) << "OrderDate/yyyymmdd" << setw(11) << "MonthID" << setw(15) << "LastID" << setw(28) << "OrderQuantity" << endl;
if (update)
{
    //body table
    row = mysql_store_result(conn);
    while (row = mysql_fetch_row(row)) //Fetch every row until the end of data
    {
        cout << setw(8) << "ID" << " " << setw(1) << row[0] << setw(8) << "Date" << setw(8) << row[1] << endl; // setw(28) << row[2] << setw(28) << row[3] << row[4] << setw(8) << row[5] << endl; //still gotta fix spacing
    }
}
//okay by

```

Diagram above shows the Calculation of overall total cost to be paid by Customers by their CustID for their overall Orders.

```
qstate = mysql_query(conn, "SELECT menuid, OrderDate, (quantity * menuprice) as totalprice, (Select custid from customer where custid = Ordering.custid) as custid, ordering.orderid FROM Menu Right JOIN Ordering USING (menuid)");

//set width //header
cout << setw(11) << "MenuID" << setw(28) << "Date " << setw(44) << " (Sub) Total Price " << setw(12) << "Cust ID" << setw(34) << " Order ID " << setw(8) << endl;
if (!qstate)
{

```

Diagram above shows the subcost to be paid by the customers by their OrderID.

## 4.4 ERROR HANDLING

```

1 # Print results for the first 1000 iterations
2
3 if (i % 1000 == 0) {
4   # Print out the current log likelihood
5   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
6   # Print out the current mean and standard deviation of the parameters
7   mean = mean(param[,1:10])
8   sd = sd(param[,1:10])
9   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
10 }
11
12 # Print results for the last 1000 iterations
13 if (i % 1000 == 0) {
14   # Print out the current log likelihood
15   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
16   # Print out the current mean and standard deviation of the parameters
17   mean = mean(param[,1:10])
18   sd = sd(param[,1:10])
19   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
20 }
21
22 # Print results for the last 1000 iterations
23 if (i % 1000 == 0) {
24   # Print out the current log likelihood
25   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
26   # Print out the current mean and standard deviation of the parameters
27   mean = mean(param[,1:10])
28   sd = sd(param[,1:10])
29   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
30 }
31
32 # Print results for the last 1000 iterations
33 if (i % 1000 == 0) {
34   # Print out the current log likelihood
35   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
36   # Print out the current mean and standard deviation of the parameters
37   mean = mean(param[,1:10])
38   sd = sd(param[,1:10])
39   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
40 }
41
42 # Print results for the last 1000 iterations
43 if (i % 1000 == 0) {
44   # Print out the current log likelihood
45   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
46   # Print out the current mean and standard deviation of the parameters
47   mean = mean(param[,1:10])
48   sd = sd(param[,1:10])
49   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
50 }
51
52 # Print results for the last 1000 iterations
53 if (i % 1000 == 0) {
54   # Print out the current log likelihood
55   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
56   # Print out the current mean and standard deviation of the parameters
57   mean = mean(param[,1:10])
58   sd = sd(param[,1:10])
59   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
60 }
61
62 # Print results for the last 1000 iterations
63 if (i % 1000 == 0) {
64   # Print out the current log likelihood
65   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
66   # Print out the current mean and standard deviation of the parameters
67   mean = mean(param[,1:10])
68   sd = sd(param[,1:10])
69   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
70 }
71
72 # Print results for the last 1000 iterations
73 if (i % 1000 == 0) {
74   # Print out the current log likelihood
75   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
76   # Print out the current mean and standard deviation of the parameters
77   mean = mean(param[,1:10])
78   sd = sd(param[,1:10])
79   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
80 }
81
82 # Print results for the last 1000 iterations
83 if (i % 1000 == 0) {
84   # Print out the current log likelihood
85   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
86   # Print out the current mean and standard deviation of the parameters
87   mean = mean(param[,1:10])
88   sd = sd(param[,1:10])
89   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
90 }
91
92 # Print results for the last 1000 iterations
93 if (i % 1000 == 0) {
94   # Print out the current log likelihood
95   cat(paste0("Iteration: ", i, " Log Likelihood: ", logLik, "\n"), file = "output.txt", append = TRUE)
96   # Print out the current mean and standard deviation of the parameters
97   mean = mean(param[,1:10])
98   sd = sd(param[,1:10])
99   cat(paste0("Mean: ", mean, " SD: ", sd, "\n"), file = "output.txt", append = TRUE)
100 }

```

Diagram above shows that system will print out error message “Invalid username or password. Would you want to try again?” when user inputs wrongly as a method of error handling.

```

end do "Please enter your choice : ",
cin <> choice
until (choice) = 0
    case 1
        ShowMainMenu() //Function to show main menu
        break;
    case 2
        ListMenu();
        break;
    case 3
        ShowCustomerRate();
        break;
    case 4
        Close();
        break;
    case 5
        menu();
        break;
end if
endif
end do "Please choose between 1-5. Press Enter to Continue...",
getch() // get char of user's keyboard so
system("cls") // clear screen
ShowMainMenu() //
break;

```

Diagram above shows that system will print out error message “Please choose between 1-5” when user inputs wrongly as a method of error handling,

## **Chapter 5**

### **5.1 CONCLUSION**

As a conclusion, I have created this restaurant management system for restaurants so that it would help make our lifestyle easier. Managing a restaurant with an organized database is much simpler compared to managing with a pen and paper. It offers a systematic lifestyle and assists both employees and users. This is also a more modern and convenient way to organize a system. Having such a system can help you boost sales as well as make better marketing strategies. Not only that, you will also be easily managing the numerous orders coming in on a daily scale. The implementation of programming technique and database system knowledge would improvise the management of restaurants by ensuring less mistakes and errors to occur as well as reduce the usage of man power in the management side. Besides a well secured system, it also helps in reducing the cost of labor and other operational costs of restaurants. In a nutshell, it is a very productive system to be used.

### **5.2 Improvements**

- Use a better language such as Python to create more better and challenging experiences
- Use a GUI to make it more convenient for the users.

## APPENDIX 1

### Database

Filters									
Containing the word: <input type="text"/>									
Table	Action	Rows	Type	Collation	Size	Overhead			
<input type="checkbox"/> admin	★ Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-			
<input type="checkbox"/> customer	★ Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	32.0 KiB	-			
<input type="checkbox"/> menu	★ Browse Structure Search Insert Empty Drop	11	InnoDB	utf8mb4_general_ci	32.0 KiB	-			
<input type="checkbox"/> ordering	★ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	64.0 KiB	-			
<input type="checkbox"/> rate	★ Browse Structure Search Insert Empty Drop	13	InnoDB	utf8mb4_general_ci	48.0 KiB	-			
5 tables	Sum	36	InnoDB	utf8mb4_general_ci	192.0 KiB	0 B			

*Diagram above shows the Restaurant database*

Show all

|

Number of rows:

25

Filter rows:

Search this table

+ Options

AdminID

AdminUsername

AdminPw

1

Admin

admin123

Check all

With selected:

Show all

|

Number of rows:

25

Filter rows:

Search this table

*Admin Table*

+ Options						
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	CustID	CustPhone	CustAddress	CustUsername	CustPw	CustName
	9	165217006	75, Kg Sg Ramal Luar, 43000, Kajang, Selangor	Ally776	ally776	Alicia
	10	196787991	No 17, 449-B, Jalan 5, 17499, KI	Moo2	moomoo222	Moosy
	11	165204006	36, Persiaran Bekor 17, Taman Pertama, 30100,	J123	jass1234	Jassvine
	12	135634612	9, Jalan USJ, Taman Seafeld Jaya, 46700	LukeR87	luke909	Luke Ryan
	13	12576110	3A, Plaza Mont Kiara, Block C-2, Jalan 1/70C, 5048	Alexaaa99	ally0101	Alexa
	14	176534781	Jalan SS 21/39, Damansara Utama, 47400	tk98	w1898	Taasha K
	15	175207009	new yorks	mira	mira15	Amirah
	20	175217006	Deltroid, Conneticut	maya	maya20	Maya

*Customer Table*





+ Options

CustID	totalPrice
10	153
15	20
20	51

*Calculation - totalPrice by CUSTID*