

ONLINE SHOPPING MANAGEMENT SYSTEM

BY

LOK WERN KAI

A RIGOROUS ASSESSMENT

SUBMITTED TO

Asia Pacific University

JULY 2022

TABLE OF CONTENTS

| No. | Content | Page |
|-----|-------------------------|---------|
| 1 | Solution Design | 3 |
| 2 | C++ Programming Concept | 4 – 8 |
| 3 | Object-Oriented Concept | 9 - 13 |
| 4 | Program Screenshot | 14 - 37 |
| 5 | Limitation | 38 |
| 6 | Conclusion | 38 |

SOLUTION DESIGN

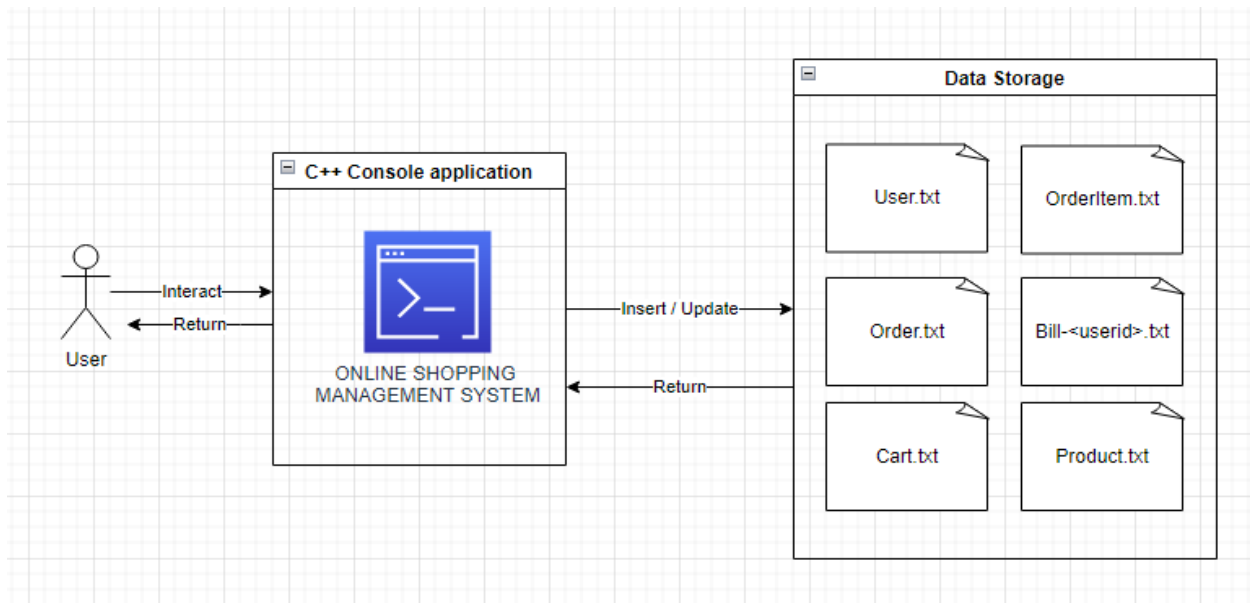


Figure 1 System Architecture Flow

The objective of this project is to develop an Online Shopping Management System for both business and customer user using C++ Object-Oriented Programming (OOP) concepts. The system should allow business user to manage user, inventory, and order. On the other side, customer user should be able to browse product, place order and check bill via the system.

Since the main objective is to focus on application layer with OOP concepts, a simple console application with text file serve as data storage can meet the requirement and ease the development process. Console application can reduce the time on User Interface (UI) design and text file as data storage can avoid the time on setting up a Relational Database Management System (RDBMS). Hence, developer will have more time focus on application layer.

The console application will retrieve instruction from the user then save the input into the relative text file as data storage for future purpose. The user will be able to view the data stored in the text file via the application as well.

C++ PROGRAMMING CONCEPT

1. Variable

Variable is one of the basic concepts in C++ programming. It is used to store and hold data values. In the application, variable such as username, password, file was declared to store the data for later use.

```
class User {
private:
    string username, password;
public:
    User(){}

    User(const string& username, const string& password) {
        this->username = username;
        this->password = password;
    }

    int Auth() {
        ifstream file("User.txt");
        string
            fid, fname, faddr, fhp
            , fusername, fpassword, permission, role;
    }
}
```

2. Function

A function is a block of code that are ready to be re-use. In C++, function can be used to perform certain action. Function such as toLower(), removeNewLine() and getNewOrderId were defined to fulfill the needs to convert string value to lower case, remove new line in the data and get new order id for new order.

```
1  #pragma once
2  #include <cctype>
3  #include <string>
4  #include <regex>
5
6  using namespace std;
7
8  string toLower(string data) {
9
10     string result = "";
11
12     for (int i = 0; i < data.length(); i++) { ... }
13
14     return result;
15 }
16
17
18
19 string removeNewLine(string data) {
20
21     string result = "";
22     regex newlines_re("\n+");
23
24     result = regex_replace(data, newlines_re, "");
25
26     return result;
27 }
28
29 int getNewOrderId() {
30     ifstream file("Order.txt");
31     string
32         oid, cid, noofitem;
33     int result = 0;
34
35     while (file && !file.eof()) { ... }
36
37     file.close();
38
39     return result + 1;
40 }
41
42
43
44
45
46
47
48
49
50
51
52 }
```

3. Control Statement

Control statement in C++ is common use to redirect the flow of the program. It is heavily used in the application to deal with different condition and scenario.

a. Conditional Statement

i. If ... else / else if ... Statement

Below screenshot shows an example of a nested If Else statement which return different string value with different condition such as is the product exists in database and is the class method successfully delete the product.



```
432  
433 string DeleteProduct(int productid) {  
434     Product product(productid);  
435     string signal = "";  
436  
437     if (product.isProductExists()) {  
438         if (product.deleteProduct()) {  
439             signal = "success";  
440         }  
441         else {  
442             signal = "fail";  
443         };  
444     }  
445     else {  
446         signal = "notfound";  
447     }  
448     return signal;  
449 }  
450
```

ii. Switch Statement

Switch statement is another common conditional statement to deal with multiple condition. In the application, it is used to route user to the correct menu based on the user permission.

```
24  switch (screen.getPermission()) {
25      case 1: {
26          AdminMenu adminScreen(screen.getUsername());
27
28          do { ... } while (!adminScreen.getIsExit());
31
32          screen.setPermission(9);
33          screen.setIsExit(false);
34
35          break;
36      }
37      case 2: { //manager
38          ManagerMenu managerScreen(screen.getUsername());
39
40          do { ... } while (!managerScreen.getIsExit());
43
44          screen.setPermission(9);
45          screen.setIsExit(false);
46
47          break;
48      }
49      case 3: { //customer
50          CustomerMenu customerScreen(screen.getUsername());
51
52          do { ... } while (!customerScreen.getIsExit());
55
56          screen.setPermission(9);
57          screen.setIsExit(false);
58
59          break;
60      }
61      case 0: //invalid
62          cout << screen.getUsername() + " is invalid";
63          break;
64      }
65  } while (!screen.getIsExit());
```

b. Loops

In C++, looping is one of the control statements that will execute the code until it meets the defined condition.

i. While Loop

A while loop in C++ will check the condition before it execute the code.

Below screenshot is an example of while loop, it will retrieve the data in the text file line by line until it reaches the end of the file.

```
string viewProduct() {
    ifstream file("Product.txt");
    string
        pid, pname, pprice, pstock, pisfragile, result;
    result += "-----\n";
    result += "                        Product Details                        =\n";
    result += "-----\n";
    result += "Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)\n";
    result += "-----";

    while (file && !file.eof()) {
        getline(file, pid, ',');
        getline(file, pname, ',');
        getline(file, pprice, ',');
        getline(file, pstock, ',');
        getline(file, pisfragile);

        result += "\n" + pid + " | " + pname + " | " + pprice + " | " + pstock + " | " + pisfragile;
    }
    result += "\n-----\n";
    file.close();

    return result;
}
```

ii. For Loop

In C++, for loop can take up to 3 statement to execute the code. Below screenshot shows a for loop statement that will loop every single character in a string data, convert it to lower case and append the converted character to a string data.

```
string toLower(string data) {
    string result = "";

    for (int i = 0; i < data.length(); i++) {
        result += tolower(data[i]);
    };

    return result;
};
```

iii. Do...While Loop

Do While loop is a reverse version of while loop. Do while loop will execute the block code for the first time then loop in if the condition is met. In the application, it was used to repeat menu screen to user until user opt to exit the application.

```
int main()
{
    MainMenu screen;

    do {
        screen.returnMenu();

        switch (screen.getPermission()) {
            case 1: { ... }
            case 2: { ... }
            case 3: { ... }
            case 0: //invalid
                cout << screen.getUsername() + " is invalid";
                break;
        }
    } while (!screen.getIsExit());
}
```


OBJECT ORIENTED CONCEPT

1. Class

A Class in C++ programming is the main aspect of OOP. It allows the developer to follow the “Don’t Repeat Yourself” principle during development. In below screenshot, multiple class such as User, Admin, Manager and Customer were defined to create different object in the application.

```
1  #pragma once
2  #include ...
6
7  using namespace std;
8
9  class User { ... };
230
231 class Admin { ... };
404
405 class Manager { ... };
527
528 class Customer { ... };
```

2. Constructor

A Constructor is a special class method that will be execute when a new object is created. In the application, the Order class have different constructor to handle different scenario when the application creates an Order object.

```
303
304 class Order {
305     private:
306         int orderid, numberofitem;
307         string customerid;
308     public:
309
310     Order(){
311         this->orderid = 0;
312         this->customerid = "";
313         this->numberofitem = 0;
314     }
315
316     Order(int orderid) {
317         this->orderid = orderid;
318         this->customerid = "";
319         this->numberofitem = 0;
320     }
321
322     Order(int orderid, string customerid, int numberofitem) {
323         this->orderid = orderid;
324         this->customerid = customerid;
325         this->numberofitem = numberofitem;
326     }
327 }
```

3. Encapsulation

Encapsulation in OOP is to ensure that data are hidden from the users and can only access by the class itself. From below screenshot, variable such as password, isExit, permission and username fall under private access specifier. User only allow to access the data via in-class get and set method.

```
311 class MainMenu {
312
313     private:
314         string password;
315         bool isExit = false;
316         int permission = 9;
317         string username;
318
319     public:
320         bool getIsExit() {
321             return this->isExit;
322         }
323
324         void setIsExit(bool isexit) {
325             this->isExit = isexit;
326         }
327
328         int getPermission() { ... }
331
332         void setPermission(int permission) { ... }
335
336         string getUsername() { ... }
339
340         void setUsername(string username) { ... }
343
```

4. Inheritance

Inheritance in OOP is to allow developer defined sub-classes from a base class, so that the method and variable in the based class can be re-use by the sub-classes. From below screenshot, Admin class is a sub-class of User class. When an Admin object require to list out customer, it can call the viewCustomer() method without create another User object.

```
8
9  class User {
10     private:
11         string username, password;
12     public:
13         User(){}
14
15         User(const string& username, const string& password) { ... }
16
17         /*int Auth(const string& username, const string& password) {*/
18         int Auth() { ... }
19
20         string getUserId(string username) { ... }
21
22         void SearchUser(const string& username) { ... }
23
24         bool isUserExists(const string& username) { ... }
25
26         bool isUserIdExists(const string& userid) { ... }
27
28         bool isCustomer(const string userid) { ... }
29
30         void viewCustomer() { ... }
31
32         string SearchProduct(string productname) { ... }
33
34         string ViewProduct() { ... }
35
36     };
37
38  class Admin: public User{
39     private:
40         string username;
41     public:
```

5. Polymorphism

Polymorphism allow OOP in C++ to have method with same name but work in different way. From below screenshot, there are 3 different Order method which behave differently and overload each other. It was designed to fit different use case in the application.

```
303
304 class Order {
305     private:
306         int orderid, numberofitem;
307         string customerid;
308     public:
309
310     Order(){
311         this->orderid = 0;
312         this->customerid = "";
313         this->numberofitem = 0;
314     }
315
316     Order(int orderid) {
317         this->orderid = orderid;
318         this->customerid = "";
319         this->numberofitem = 0;
320     }
321
322     Order(int orderid, string customerid, int numberofitem) {
323         this->orderid = orderid;
324         this->customerid = customerid;
325         this->numberofitem = numberofitem;
326     }
327
```

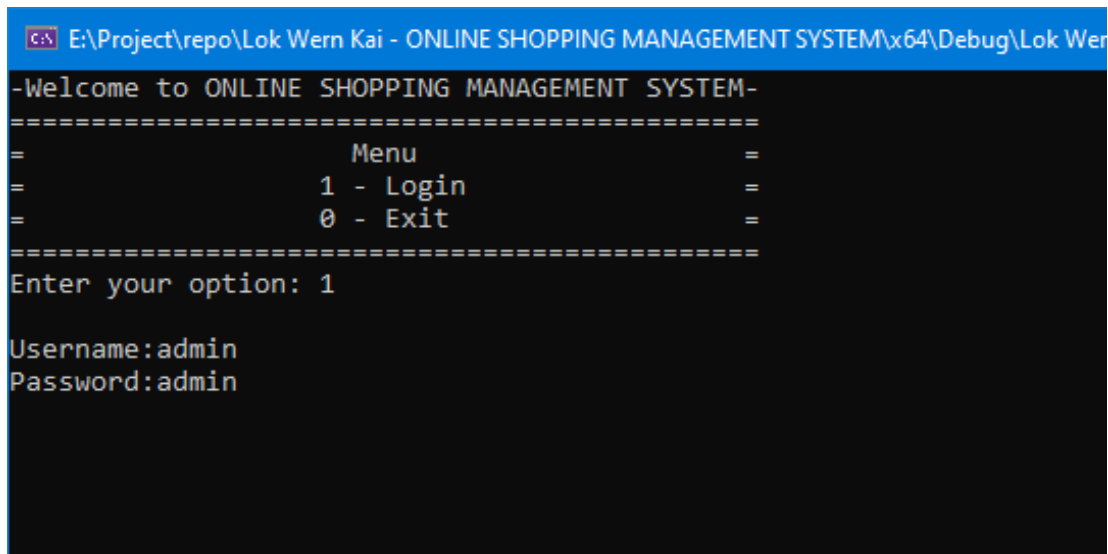
6. Abstraction

Encapsulation in OOP is to ensure that data are hidden from the users and not accessible outside from the class. From below screenshot, variable such as password, isExit, permission, username and delCart() method falls under private access specifier which user can't access the data from the outside. However, public method such as Cart are callable when the user creates a Cart object.

```
866 class Cart {
867     private:
868         string username, userid, productname;
869         int productid, quantity, amount, packagefee, orderid;
870
871     bool delCart() {
872         ifstream file("Cart.txt");
873         ofstream fileNewItem("carttemp.txt");
874
875         string
876             oid, cid, username, pid, pname, quantity, amount, packagefee
877             , toRemain, line;
878
879         while (file && !file.eof()) { ... }
880
881         file.close();
882
883         //cout << toRemain;
884         fileNewItem << toRemain;
885
886         fileNewItem.close();
887
888         remove("Cart.txt");
889
890         if (rename("carttemp.txt", "Cart.txt") == 0) { ... }
891         else { ... }
892     }
893     public:
894     Cart(string username, string userid, int orderid) { ... }
895     Cart(
```

PROGRAM SCREENSHOT

Main Menu



```
C:\> E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM.exe

-Welcome to ONLINE SHOPPING MANAGEMENT SYSTEM-
=====
=                      Menu                      =
=                      1 - Login                  =
=                      0 - Exit                  =
=====
Enter your option: 1

Username:admin
Password:admin
```

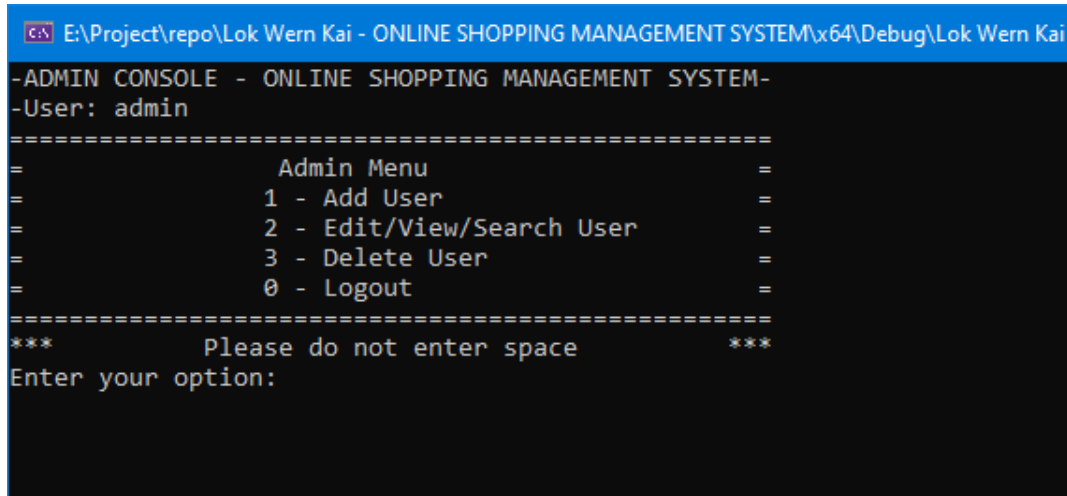
Above screen shot is the main menu of the application.

To login, user will need to insert “1” and press enter. The username and password will prompt for user input. If the user does exist in the system, it will route the user to correct menu, else the user will not be able to login.

The main menu screen will remain until user insert “0” then the application will shut down.

Admin

1. Admin Menu

A screenshot of a Windows command prompt window. The title bar shows the file path: E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai. The console output shows the program has reached the admin console, the user is 'admin', and it displays an 'Admin Menu' with four options: 1 - Add User, 2 - Edit/View/Search User, 3 - Delete User, and 0 - Logout. The prompt asks the user to enter an option.

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
=                      Admin Menu                      =
=                      1 - Add User                      =
=                      2 - Edit/View/Search User          =
=                      3 - Delete User                    =
=                      0 - Logout                        =
=====
***          Please do not enter space          ***
Enter your option:
```

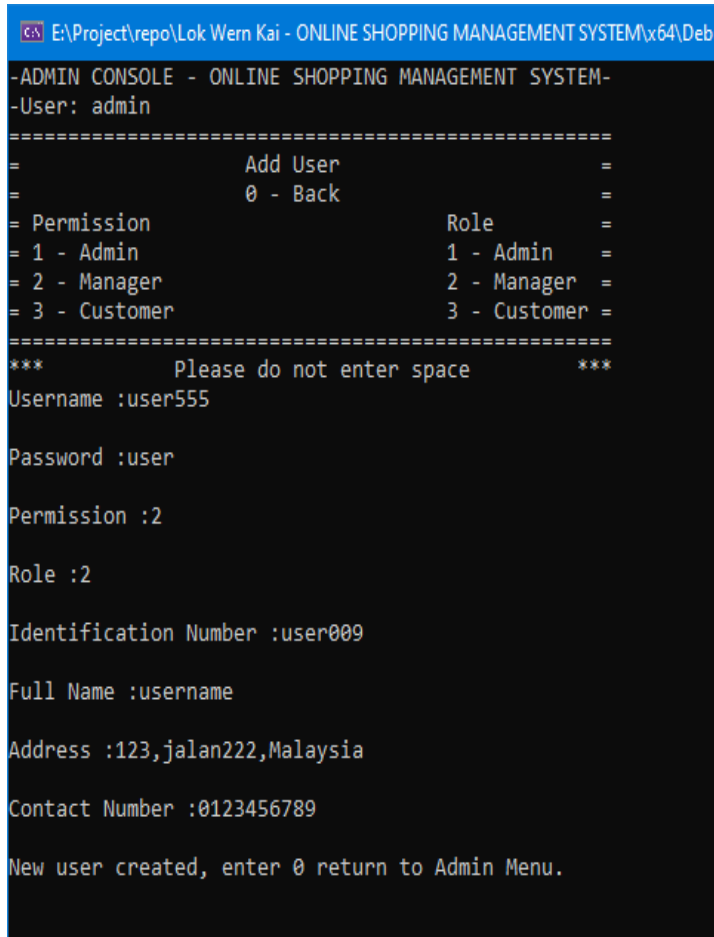
Above screenshot is the admin menu.

Only user with admin permission will be able to see the screen.

It provides option for admin to manage the user of the system by using the functionality such as Add User, Edit / View / Search User and Delete User.

If the user opts to logout from the admin account, insert “0” and the application will bring the user back to the main menu.

2. Create User



```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\64\Deb
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
=                Add User                =
=                0 - Back                 =
= Permission                                Role =
= 1 - Admin                                1 - Admin =
= 2 - Manager                              2 - Manager =
= 3 - Customer                              3 - Customer =
=====
***          Please do not enter space          ***
Username :user555
Password :user
Permission :2
Role :2
Identification Number :user009
Full Name :username
Address :123,jalan222,Malaysia
Contact Number :0123456789
New user created, enter 0 return to Admin Menu.
```

Above screenshot is the screen for admin to create new user.

Admin user will require to insert new user details such as username, password, user permission, user role, user identification number, etc.

These details will then be stored in the User.txt.

So that, the newly created user will be able to login the application with the username and password.

3. Delete User

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\De
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
=                      Delete User                      =
=                      0 - Back                          =
=====
***      Please do not enter space      ***
Username :user555

User deleted, enter 0 return to Admin Menu.
```

Above screenshot is for the delete user function.

It is a simple screen; admin user is required to insert the username of the user in order to remove the user access.

Once the operation complete, the user details will be removed from the User.txt.

Hence, the user will no longer have the access to the application.

4. View User

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai - ONLINE SHOPPING MANA
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
=                      User Details                      =
=                      1 - Edit                          =
=                      2 - View                          =
=                      3 - Search                        =
=                      0 - Back                          =
=====
***      Please do not enter space      ***
Enter your option: 2

-----
=      Permission      Role      =
=      1 - Admin       1 - Admin  =
=      2 - Manager     2 - Manager =
=      3 - Customer    3 - Customer =
-----

Id | Name | Address | Contact | Username | Password | Permission | Role
-----
A1006 | WernKai | N0.1,Jalan 999 | 0123456789 | admin | admin | 1 | 1
999avs | kaikai | 123,,asdasd | 1234 | po | 123 | 2 | 2
manager123 | manager | 123,ppp | 165223636 | manager | manager | 2 | 2
C001 | John | 123, | 123456789 | customer | customer | 3 | 3
1 | customer | 123 | 123 | test | test | 3 | 3
CA999 | kai | 123,jalansasa | 1230 | customer3 | customer3 | 3 | 3
admin001 | adminintest | 123 | 123 | user123 | user123 | 1 | 1
user009 | username | 555,jalanmerdeka,malaysia | 123456789 | user555 | user | 2 | 2
-----

Please enter 0 to return to Admin Menu.
```

Above screenshot is the view user function.

Once the admin user enters the view option, all user details stored in User.txt will be retrieve and display in the application.

5. Edit User

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\Debug\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM.exe
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
User Details
=====
1 - Edit
2 - View
3 - Search
0 - Back
=====
*** Please do not enter space ***
Enter your option: 1

=====
Permission                                     Role
1 - Admin                                     1 - Admin
2 - Manager                                  2 - Manager
3 - Customer                                3 - Customer
=====
Id | Name | Address | Contact | Username | Password | Permission | Role
-----
A1006 | WernKai | N0.1,Jalan 999 | 0123456789 | admin | admin | 1 | 1
999avs | kaikai | 123,,asdasd | 1234 | po | 123 | 2 | 2
manager123 | manager | 123,ppp | 165223636 | manager | manager | 2 | 2
C001 | John | 123, | 123456789 | customer | customer | 3 | 3
1 | customer | 123 | 123 | test | test | 3 | 3
CA999 | kai | 123,jalansasa | 1230 | customer3 | customer3 | 3 | 3
admin001 | admin | 123 | 123 | user123 | user123 | 1 | 1
user009 | username | 123,jalan222,Malaysia | 123456789 | user555 | user | 2 | 2
=====

Please enter a user name to edit.
user555

Current record:
user009 | username | 123,jalan222,Malaysia | 123456789 | user555 | user | 2 | 2

Please enter details (enter 0 if unchange):

Address: 555,jalanmerdeka,malaysia

Password: 0

Contact: 0123456789

Permission: 0

Role: 0

Successfully update user details.

Edited record:
user009 | username | 555,jalanmerdeka,malaysia | 123456789 | user555 | user | 2 | 2

Please enter 0 to return to Admin Menu.
```

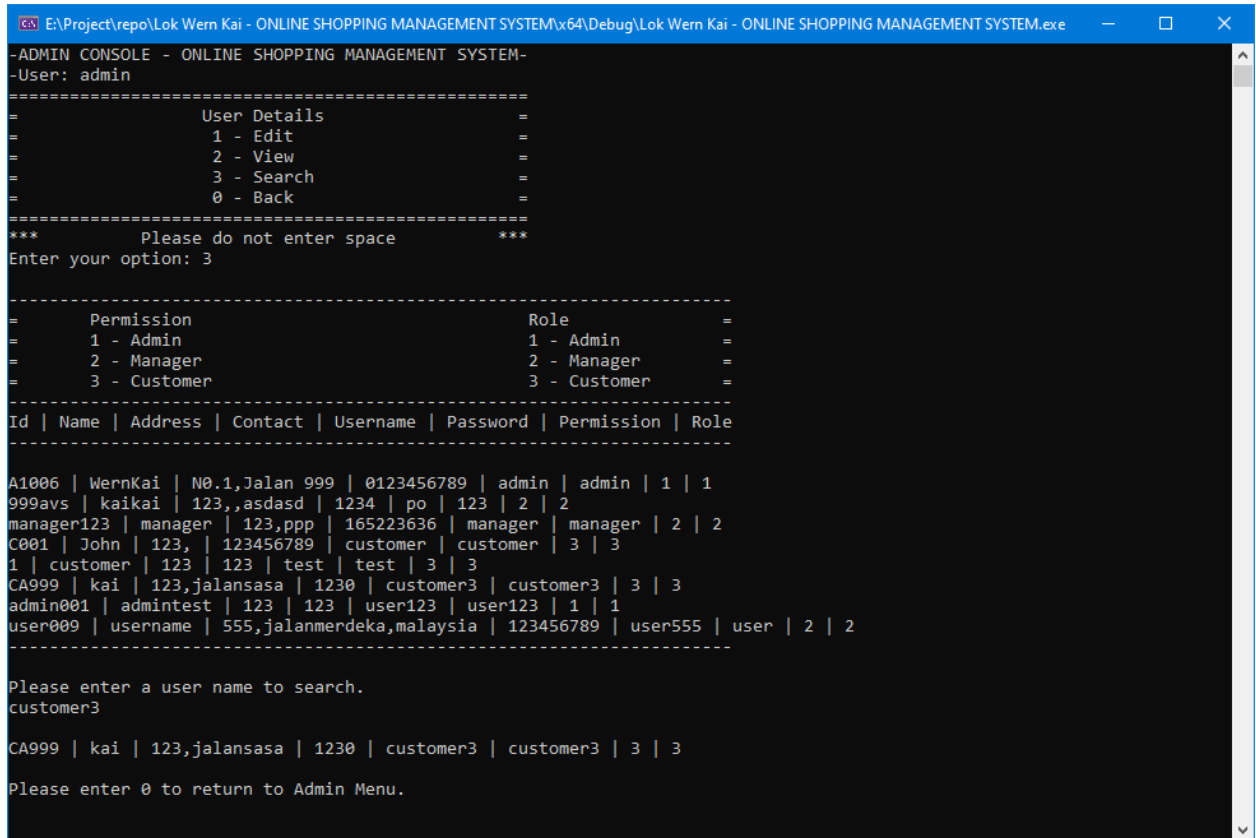
In this screen, admin user is required to enter the usernames that are going to be edit.

The application will show the user's current details from User.txt.

Then multiple fields will prompt for admin user to update the user details.

Once the admin user completes the operation, the updated details of the user will prompt from application as a verification.

6. Search User



```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: admin
=====
=                               User Details                               =
=                               1 - Edit                               =
=                               2 - View                               =
=                               3 - Search                             =
=                               0 - Back                               =
=====
***      Please do not enter space      ***
Enter your option: 3

=====
=                               Permission                               =
=                               1 - Admin                               =
=                               2 - Manager                             =
=                               3 - Customer                             =
=====
Id | Name | Address | Contact | Username | Password | Permission | Role
-----
A1006 | WernKai | N0.1,Jalan 999 | 0123456789 | admin | admin | 1 | 1
999avs | kaikai | 123,,asdasd | 1234 | po | 123 | 2 | 2
manager123 | manager | 123,ppp | 165223636 | manager | manager | 2 | 2
C001 | John | 123, | 123456789 | customer | customer | 3 | 3
1 | customer | 123 | 123 | test | test | 3 | 3
CA999 | kai | 123,jalansasa | 1230 | customer3 | customer3 | 3 | 3
admin001 | admintest | 123 | 123 | user123 | user123 | 1 | 1
user009 | username | 555,jalanmerdeka,malaysia | 123456789 | user555 | user | 2 | 2
=====

Please enter a user name to search.
customer3

CA999 | kai | 123,jalansasa | 1230 | customer3 | customer3 | 3 | 3

Please enter 0 to return to Admin Menu.
```

Search user screen which allows admin user to search for user.

Admin user will have to insert the username to search.

If the user does exist in the application, the details will display on the application.

Manager

Manager Menu

```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                Manager Menu                =
=          1 - Add Product                    =
=          2 - Edit/View/Search Product       =
=          3 - Delete Product                 =
=          4 - Add Order                      =
=          5 - View/Search Order               =
=          6 - Delete Order                   =
=          7 - Add Order Item                 =
=          8 - Edit/View/Search Order Item    =
=          9 - Delete Order Item              =
=          0 - Logout                         =
=====
***                Please do not enter space                ***
Enter your option:
```

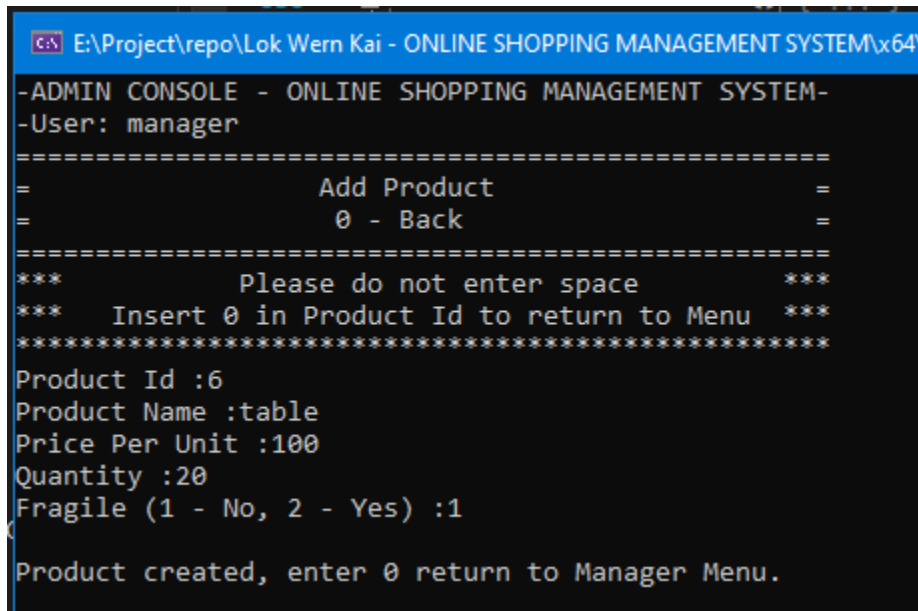
Above screenshot is the manager menu.

Only user with manager permission will be able to see the screen.

It provides option for manager to manage the product, order or order item of the system by using the functionality such as Add, Edit / View / Search and Delete.

If the manager opts to logout from the manager account, insert “0” and the application will bring the user back to the main menu.

1. Create Product



```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                      Add Product                      =
=                      0 - Back                          =
=====
***      Please do not enter space      ***
***      Insert 0 in Product Id to return to Menu      ***
*****
Product Id :6
Product Name :table
Price Per Unit :100
Quantity :20
Fragile (1 - No, 2 - Yes) :1
Product created, enter 0 return to Manager Menu.
```

Above screenshot is the screen for manager to create new product.

Manager user will require to insert new product details such as product id, product name, price, quantity, fragile, etc.

These details will then be stored in the Product.txt.

So that, the newly created product will be available in the shopping system.

2. Delete Product

```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                      Delete Product                      =
=                      0 - Back                             =
=====
***      Please do not enter space      ***
***  Insert 0 in Product Id to return to Menu  ***
*****

-----
=                      Product Details                      =
-----
Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)
-----
1 | bread | 2.500000 | 99 | 1
2 | handphone | 100.000000 | 99 | 2
3 | glass | 5.000000 | 99 | 2
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
6 | table | 200.000000 | 19 | 1
-----

Product Id :6

Product deleted, enter 0 return to Manager Menu.
```

Above screenshot is for the delete product function.

It is a simple screen; manager user is required to insert the product id of the product in order to remove the product.

Once the operation complete, the product details will be removed from the Product.txt.

Hence, the product will no longer available in the system.

3. View Product

```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                      Product Details                      =
=                      1 - Edit                              =
=                      2 - View                              =
=                      3 - Search                             =
=                      0 - Back                               =
=====
***          Please do not enter space          ***
*****
Enter your option: 2

-----
=                      Product Details                      =
-----
Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)
-----
1 | bread | 2.500000 | 99 | 1
2 | handphone | 100.000000 | 99 | 2
3 | glass | 5.000000 | 99 | 2
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
6 | table | 100.000000 | 20 | 1
-----

Please enter 0 to return to Manager Menu.
```

Above screenshot is the view product function.

Once the manager user enters the view option, all user details stored in Product.txt will be retrieve and display in the application.

4. Edit Product

```
=====
                        Product Details
=====
Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)
=====
1 | bread | 2.500000 | 99 | 1
2 | handphone | 100.000000 | 99 | 2
3 | glass | 5.000000 | 99 | 2
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
6 | table | 50.000000 | 19 | 1
=====

Please enter a product name to edit.
table

Current record:
6 | table | 50.000000 | 19 | 1

Please enter details:

Price Per Unit (enter -1 if unchange): 200

Stock Quantity: 0

Fragile (1 - No, 2 - Yes, 0 - Unchange): 0

Edited record:
6 | table | 200.000000 | 19 | 1

Product updated, enter 0 return to Manager Menu.
```

In this screen, manager user is required to enter the product name that are going to be edit.

The application will show the product's current details from Product.txt.

Then multiple fields will prompt for manager user to update the product details.

Once the manager user completes the operation, the updated details of the product will prompt from application as a verification.

5. Search Product

```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                      Product Details                      =
=                      1 - Edit                             =
=                      2 - View                             =
=                      3 - Search                           =
=                      0 - Back                             =
=====
***          Please do not enter space          ***
*****
Enter your option: 3

-----
=                      Product Details                      =
-----
Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)
-----
1 | bread | 2.500000 | 99 | 1
2 | handphone | 100.000000 | 99 | 2
3 | glass | 5.000000 | 99 | 2
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
6 | table | 200.000000 | 19 | 1
-----

Please enter a product name to search.
shirt

5 | shirt | 5.000000 | 100 | 1

Please enter 0 to return to Manager Menu.
```

Search product screen which allows manager user to search for product.

Manager user will have to insert the product name to search.

If the product does exist in the application, the details will display on the application.

6. Create Order

```
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                Add Order                =
=                0 - Back                  =
=====
***          Please do not enter space      ***
***  Insert 0 in Order Id to return to Menu  ***
*****
Order Id :2
Customer Id :C001

Order created, enter 0 return to Manager Menu.
```

Above screenshot is the screen for manager to create new order.

Manager user will require to insert new order details such as order id and customer id.

These details will then be stored in the Order.txt.

So that, the newly created order will be ready to add order item.

7. Delete Order

```
-----
=      Order Item Details      =
-----
Order Id | Product Id | Quantity
-----
1 | 1 | 2
1 | 2 | 2
1 | 3 | 2
2 | 1 | 20
-----

Order Id :2

Order deleted, enter 0 return to Manager Menu.
```

Above screenshot is for the delete order function.

It is a simple screen; manager user is required to insert the order id of the order to remove the order.

Once the operation complete, the order details will be removed from the Order.txt.

Hence, the order will no longer available in the system and the associated order item will be remove as well.

8. View Order

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai - ONLINE SHC

=              Order Details              =
=              1 - View                    =
=              2 - Search                  =
=              0 - Back                    =
=====
***          Please do not enter space      ***
*****
Enter your option: 1

-----
=              Customer Details            =
-----
Id | Name | Address | Contact | Username | Password | Permission | Role
-----
C001 | John | 123, | 123456789 | customer | customer | 3 | 3
U001 | UserName | 123,JalanTupai,Selangor | 123456789 | user01 | user01 | 3 | 3
-----

-----
=              Order Details              =
-----
Order Id | Customer Id | Number of Item
-----
1 | C001 | 3
2 | C001 | 0
-----

Please enter 0 to return to Manager Menu.
```

Above screenshot is the view order function.

Once the manager user enters the view option, all order details stored in Order.txt will be retrieve and display in the application.

9. Search Order

```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\x64\Debug\Lok Wern Kai - ONLINE SH
Enter your option: 2

=====
                        Customer Details
=====
Id | Name | Address | Contact | Username | Password | Permission | Role
-----
C001 | John | 123, | 123456789 | customer | customer | 3 | 3
U001 | UserName | 123,JalanTupai,Selangor | 123456789 | user01 | user01 | 3 | 3
-----

=====
                        Order Details
=====
Order Id | Customer Id | Number of Item
-----
1 | C001 | 3
2 | C001 | 2
-----

Please enter an order id to search.
2

2 | C001 | 2

Total amount: 123

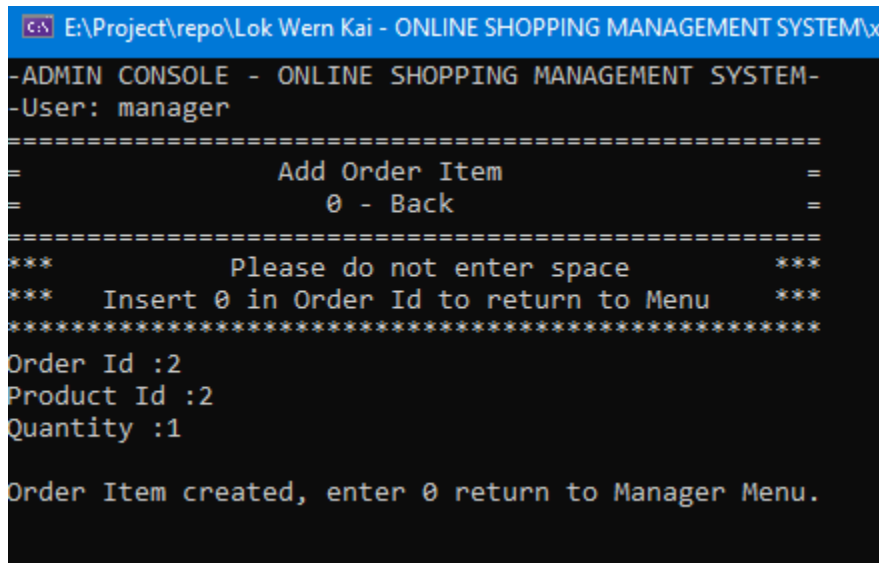
Please enter 0 to return to Manager Menu.
```

Search order screen which allows manager user to search for order.

Manager user will have to insert the order id to search.

If the order does exist in the application, the details will display on the application.

10. Create Order Item



```
E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYSTEM\src>
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                Add Order Item                =
=                0 - Back                        =
=====
***                Please do not enter space        ***
***  Insert 0 in Order Id to return to Menu  ***
*****
Order Id :2
Product Id :2
Quantity :1

Order Item created, enter 0 return to Manager Menu.
```

Above screenshot is the screen for manager to create new order item.

Manager user will require to insert new order item details such as order id, product id, price, and quantity.

These details will then be stored in the OrderItem.txt.

11. Delete Order Item

```
-----  
=      Order Item Details      =  
-----  
Order Id | Product Id | Quantity  
-----  
1 | 1 | 2  
1 | 2 | 2  
1 | 3 | 2  
2 | 1 | 10  
2 | 2 | 1  
-----  
  
Order Id :2  
  
Product Id :2  
  
Order Item deleted, enter 0 return to Manager Menu.
```

Above screenshot is for the delete order item function.

It is a simple screen; manager user is required to insert the order id and product id to remove the order item.

Once the operation complete, the order item details will be removed from the OrderItem.txt.

Hence, the order item will no longer available in the system.

12. View Order Item

```
Select E:\Project\repo\Lok Wern Kai - ONLINE SHOPPING MANAGEMENT SYST
-ADMIN CONSOLE - ONLINE SHOPPING MANAGEMENT SYSTEM-
-User: manager
=====
=                Order Item Details                =
=                1 - Edit                            =
=                2 - View                            =
=                3 - Search                          =
=                0 - Back                            =
=====
***                Please do not enter space                ***
*****
Enter your option: 2

-----
=      Order Item Details      =
-----
Order Id | Product Id | Quantity
-----
1 | 1 | 2
1 | 2 | 2
1 | 3 | 2
2 | 1 | 10
2 | 2 | 1
-----

Please enter 0 to return to Manager Menu.
```

Above screenshot is the view order item function.

Once the manager user enters the view option, all order item details stored in OrderItem.txt will be retrieve and display in the application.

13. Edit Order Item

```
=      Order Item Details      =  
-----  
Order Id | Product Id | Quantity  
-----  
  
1 | 1 | 2  
1 | 2 | 2  
1 | 3 | 2  
2 | 1 | 10  
-----  
  
Please enter a order id to edit.  
2  
  
Please enter a product id to edit.  
1  
  
Current record:  
2 | 1 | 10  
  
Please enter details (enter -1 if unchange):  
  
Stock Quantity: 20  
  
Edited record:  
2 | 1 | 20  
  
Order Item updated, enter 0 return to Manager Menu.
```

In this screen, manager user is required to enter the order id and product id that are going to be edit.

The application will show the order item's current details from OrderItem.txt.

Then multiple fields will prompt for manager user to update the order item details.

Once the manager user completes the operation, the updated details of the order item will prompt from application as a verification.

14. Search Order Item

```
=          2 - View          =
=          3 - Search        =
=          0 - Back          =
=====
***          Please do not enter space          ***
*****
Enter your option: 3

-----
=      Order Item Details      =
-----
Order Id | Product Id | Quantity
-----
1 | 1 | 2
1 | 2 | 2
1 | 3 | 2
2 | 1 | 10
2 | 2 | 1
-----

Please enter a order id to search.
2

2 | 1 | 10

2 | 2 | 1

Please enter 0 to return to Manager Menu.
```

Search order item screen which allows manager user to search for order item.

Manager user will have to insert the order id to search.

If the order does exist in the application, the details will display on the application.

Customer

```
-          ONLINE SHOPPING MANAGEMENT SYSTEM          -
-User: customer
=====
=                  Customer Menu                        =
=                  1 - Search Product                  =
=                  2 - Cart                             =
=                  3 - Bill                             =
=                  0 - Logout                           =
=====
***          Please do not enter space          ***
Enter your option:
```

Above screenshot is the customer menu.

Only user with customer permission will be able to see the screen.

It provides option for customer to search for available product, add product to cart and view the bill.

If the customer opts to logout from the customer account, insert “0” and the application will bring the user back to the main menu.

1. Search Product

```
-          ONLINE SHOPPING MANAGEMENT SYSTEM          -
-User: customer
=====
=                  Search Product                      =
=                  0 - Back                            =
=====
***          Please do not enter space                ***
*****

-----
=                  Product Details                      =
-----
Product Id | Name | Price Per Unit | In Stock | Fragile (1 - No / 2 - Yes)
-----
1 | bread | 2.500000 | 99 | 1
2 | handphone | 100.000000 | 99 | 2
3 | glass | 5.000000 | 99 | 2
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
-----

Please enter a product name to search.
mug

4 | mug | 10.000000 | 100 | 2

Please enter 0 to return to Customer Menu.
```

Search product screen which allows customer user to search for product.

Customer user will have to insert the product name to search.

If the product does exist in the application, the details will display on the application.

This function share same Search Product functionality from Manager user.

2. Add Product to Cart

```
4 | mug | 10.000000 | 100 | 2
5 | shirt | 5.000000 | 100 | 1
-----

Adding to cart...
Product Id :4
Quantity :2

Item added to cart.

Continue add into cart? (1 - Yes, 0 - No)
1

Adding to cart...
Product Id :3
Quantity :2

Item added to cart.

Continue add into cart? (1 - Yes, 0 - No)
0

Check Out done...

Order created.....

You may check you bill now.....

Enter 0 return to Customer Menu.
```

Above screenshot is the screen that allow customer user to add product into cart.

Customer are allowed to add multiple but not duplicated product, if customer opt to stop adding product, insert "0" and the application will bring the user back to customer menu.

These details will be stored in the Cart.txt

3. View Bill

```
=====
=                               =
=           Customer Menu       =
=           1 - Search Product  =
=           2 - Cart            =
=           3 - Bill            =
=           0 - Logout          =
=====
***           Please do not enter space       ***
Enter your option: 3

-----
=                               =
=                               =
=           Bill                =
=                               =
= Product Name | Quantity | Amount | Packaging =
=-----
***Order Id: 1
bread | 2 | 4 | 1
handphone | 2 | 200 | 2
glass | 2 | 10 | 2
-----
Total Amount:219

***Order Id: 2
mug | 2 | 20 | 2
glass | 2 | 10 | 2
-----
Total Amount:34

Enter 0 return to Customer Menu.
```

View bill screen is the screen that display the customer bills.

After customer finish adding product into cart, once the user exits the Cart function, a bill will be created and store the details in a file with “Bill-<username>.txt” naming format. The bill will show the total amount of each order including the packing fees.

LIMITATION

Beside of the functionality of the application, limitation comes together.

Below are some limitations with the possible solution to make the application more advance:

1. Implement more data verification such as address validation, phone number format checker to avoid potential data format issue.
2. In the current system, there's no encryption feature for user password. Encryption such as AES should be implemented to protect user data.
3. Currently, each user bind to a specific permission to access the application. Once the application getting more usage and function, granting user with permission is not sufficient to manager user access. A role for each user should be introduce, so that we can group user in a specific role that share similar permission.
4. With the current design, customer is the only user that allow to generate bill. Even though manager is allowed to add order and order item, however, bill will not be generated. This feature should be added in order to make the application more complete.
5. Console application can only serve application with simple usage. Once more and more function come in, a better Graphic User Interface (GUI) should be introduced to ease the way user interact with the application. GUI library for C++ such Gtkmm or Qt will be a potential solution.
6. Limitation of data storage using text file is also one of the limitations, a RDBMS should be introduced and integrate with the application to provide better data storage and data security.

CONCLUSION

As a conclusion, a simple C++ console application was developed as an online shopping system with C++ programming and object-oriented concept. The application can fulfill the basic function as requested in the requirement. However, in application development, there's no ending in development. More function and features should be considered and implement into the application to make it more complete.