Topic : Online Laundry Service

Group no : MLB_08.02_3

Campus : Malabe

Submission Date : 20th May 2022

We declare that this is our own work and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	Name	Contact Number
IT21167300	Sumanasekara P.D.M	0701224964
IT21166556	I.U.N.Nadeeshan	0742605606
IT21167546	M.G.T Rashmika	0716367106
IT21173868	Gamlathge G.G.A.U.	0758846449
IT21170270	Maduranga D.B.W.N	0776944353

Requirements as a Description

Mylaundry.lk is a sophisticated laundry service that is built to provide a seamless experience. Once a customer is registered and logged in, they can view our services, add services to the cart then finally if they want, place an order. But not to worry if they have a change of mind, they can edit the cart before placing an order and they can cancel an order after they place a one. When placing an order customers can pay in a wide variety of payment methods for added ease of access. Customers will receive notification about their order in every step. They can also visit the order page to see order status. If it is not enough, they can inquire us via a contact us form or by mobile or email. Customers will be provided an invoice for every order; these invoices are system generated but our accountants will manually check them before issuing. Regular customers can save money by buying a package. It is much appreciated if customers can provide us feedback so that we can improve our service. However, our staff members will moderate the publicly available feedbacks to avoid spam and feedback misuse.

Staff members, delivery person and accountants are given admin accounts to fulfil their responsibilities. All of them can view and update order status in real time with the admin accounts. However, order cancellation is only done by staff members as they are trained to handle order cancellations. Accountants will use the system data to generate all kind of reports so that we can evaluate system performance and other metrics.

Requirements as User Stories

1. Customers must be able to receive notifications so that they get informed.
2. Customers must be able to inquire about the laundry service so that they get informed.
3. Customers must be able to register and login to a customer account for the service so that it increases the efficiency of the service.
4. Customers must be able to view laundry items.
5. Customers must be able to add laundry items to the cart.
6. Customers must be able to edit the cart.
7. Customers must be able to place orders as it is the main laundry service provided by the system.
8. Customers must be able to pay for the orders online so that they are at ease.
9. Customers must be able to view order status so that they can stay informed about its situation.
10. Customers must be able to cancel orders so that they can avoid needless orders.
11. Customers must be provided with invoices for the orders so that they can use them for tax purposes.
12. Customers must be able to submit feedback so that the service can improve based on it.
13. Customer must be able to buy a package to save money.

14. Staff members must be able to log in to a staff account so that they can provide the service.
15. Staff members must be able to view order details so that they can provide the service.
16. Staff members must be able to update order status so that customers can know order status.
17. Staff members must be able to approve order cancellations so that customers are at ease.
18. Staff members must be able to view customer feedback & inquiries so that they can improve the business.

19. Delivery persons must be able to log in to a staff account so that they can provide the service.

20. Delivery persons must be able to view order details so that they can deliver or pick up the laundry.
21. Delivery persons must be able to update order status so that customers and staff members know order status.

22. Accountants must be able to log in to a staff account so that they can provide the service.
23. Accountants must be able to view payment information so that they can create invoices.
24. Accountants must be able to generate reports so that the business can speculate finance.

Identified Classes

Laundry Item
Customer
Notification
Cart
Order
Invoice
Feedback
Staff Member
Inquiry
Delivery Person
Accountant
Payment
Report

CRC Cards

Laundry Item	
Responsibilities:	**Collaboration:**
➢ Add new laundry items ➢ Display laundry item details ➢ Edit laundry item details	

Customer	
Responsibilities:	**Collaboration**:
➢ Register user details ➢ Log in the user ➢ Store user details ➢ Edit user details	

Notification	
Responsibilities:	**Collaboration:**
➢ Send new notification ➢ Show description	➢ Customer

Cart	
Responsibilities:	**Collaboration:**
➢ Store customer order details ➢ Calculate the customer order price ➢ Add customer order item ➢ Remove customer order item	➢ Order ➢ Order ➢ Order ➢ Order

Order	
Responsibilities:	**Collaboration:**
➢ Check if laundry item services are offered ➢ Place order ➢ Order validation ➢ Store order details ➢ Cancel order	➢ Laundry item ➢ Customer, Delivery Person ➢ Payment ➢ Payment, Staff Member

Invoice	
Responsibilities:	**Collaboration:**
➢ Create a new invoice ➢ Store invoice details ➢ Get order details ➢ Get payment details ➢ Get customer details	➢ Accountant ➢ Order ➢ Payment ➢ Customer

Feedback	
Responsibilities:	**Collaboration:**
➢ Add new feedback ➢ Store feedback details ➢ Get customer details ➢ Add feedback reply	➢ Customer ➢ Staff Member

Staff Member	
Responsibilities:	**Collaboration:**
➢ Log in the staff member	

Inquiry	
Responsibilities:	**Collaboration:**
➢ Add new inquiry ➢ Store inquiry details ➢ Get customer details ➢ Add inquiry reply	➢ Customer ➢ Staff Member

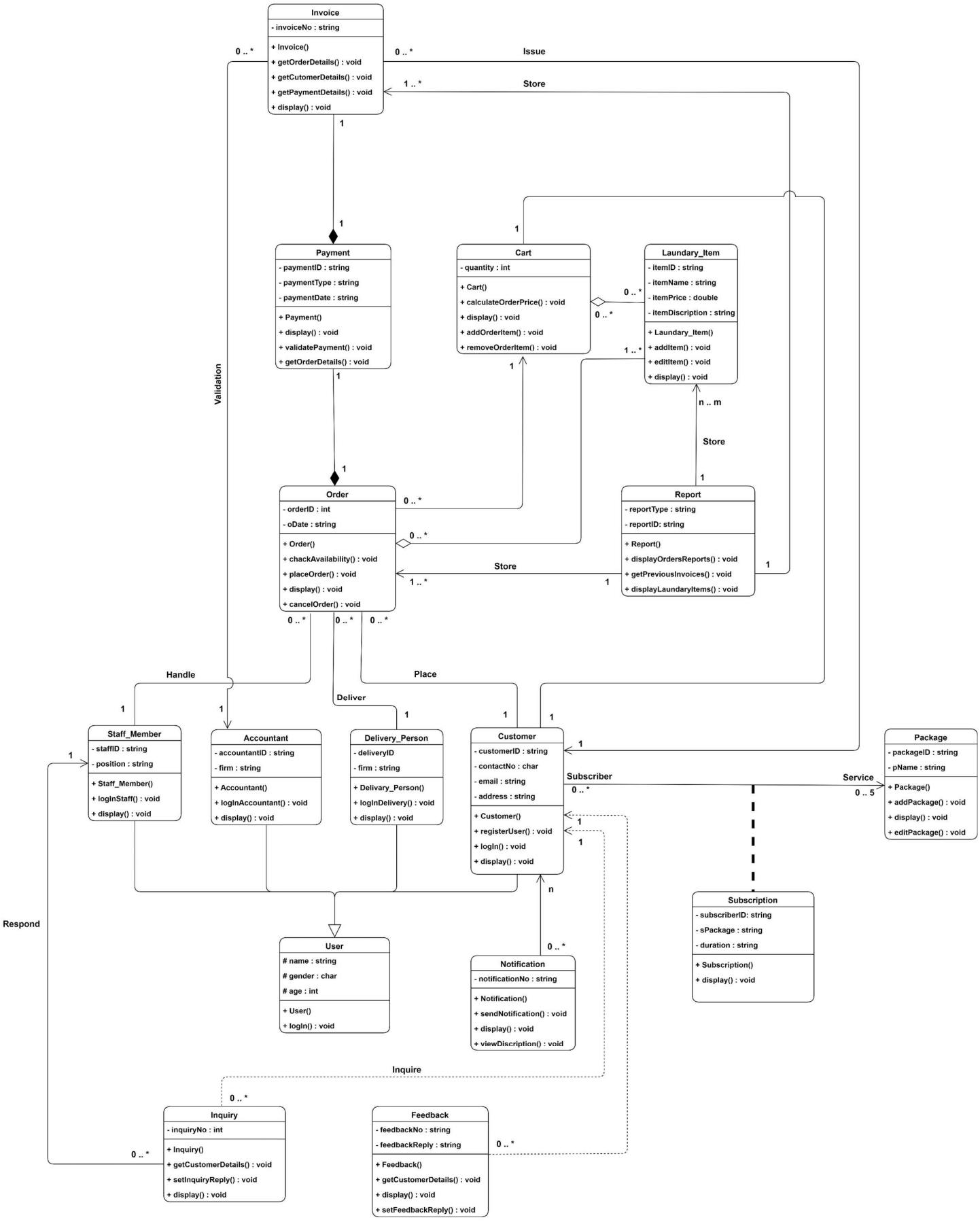
Delivery Person	
Responsibilities:	**Collaboration:**
➢ Log in the delivery person	

Accountant	
Responsibilities:	**Collaboration:**
➢ Log in the accountant	

Payment	
Responsibilities:	**Collaboration:**
➢ Add new payment ➢ Store payment details ➢ Validate payment	➢ Order

Report	
Responsibilities:	**Collaboration:**
➢ List of previous orders ➢ List of invoices ➢ List of laundry items	➢ Order ➢ Invoice ➢ Laundry Item

Exercise 2


Invoice
- invoiceNo : string
+ Invoice()
+ getOrderDetails() : void
+ getCutomerDetails() : void
+ getPaymentDetails() : void
+ display() : void
Issue
Store
Payment
- paymentID : string
- paymentType : string
- paymentDate : string
+ Payment()
+ display() : void
+ validatePayment() : void
+ getOrderDetails() : void
Cart
- quantity : int
+ Cart()
+ calculateOrderPrice() : void
+ display() : void
+ addOrderItem() : void
+ removeOrderItem() : void
Laundary_Item
- itemID : string
- itemName : string
- itemPrice : double
- itemDiscription : string
+ Laundary_Item()
+ addItem() : void
+ editItem() : void
+ display() : void
Validation
Store
Order
- orderID : int
- oDate : string
+ Order()
+ chackAvailability() : void
+ placeOrder() : void
+ display() : void
+ cancelOrder() : void
Report
- reportType : string
- reportID: string
+ Report()
+ displayOrdersReports() : void
+ getPreviousInvoices() : void
+ displayLaundaryItems() : void
Handle
Deliver
Place
Staff_Member
- staffID : string
- position : string
+ Staff_Member()
+ logInStaff() : void
+ display() : void
Accountant
- accountantID : string
- firm : string
+ Accountant()
+ logInAccountant() : void
+ display() : void
Delivery_Person
- deliveryID
- firm : string
+ Delivery_Person()
+ logInDelivery() : void
+ display() : void
Customer
- customerID : string
- contactNo : char
- email : string
- address : string
+ Customer()
+ registerUser() : void
+ logIn() : void
+ display() : void
Subscriber
Service
Package
- packageID : string
- pName : string
+ Package()
+ addPackage() : void
+ display() : void
+ editPackage() : void
Subscription
- subscriberID: string
- sPackage : string
- duration : string
+ Subscription()
+ display() : void
Respond
User
name : string
gender : char
age : int
+ User()
+ logIn() : void
Notification
- notificationNo : string
+ Notification()
+ sendNotification() : void
+ display() : void
+ viewDiscription() : void
Inquire
Inquiry
- inquiryNo : int
+ Inquiry()
+ getCustomerDetails() : void
+ setInquiryReply() : void
+ display() : void
Feedback
- feedbackNo : string
- feedbackReply : string
+ Feedback()
+ getCustomerDetails() : void
+ display() : void
+ setFeedbackReply() : void

Individual Contribution for Exercise 2

Maduranga D.B.W.N (IT21170270)-->

- Payment
- Invoice
- Report

Sumanasekara P.D.M (IT21167300) -->

- Order
- Cart
- Laundary_Item

I.U.N.Nadeeshan (IT21166556) -->

- Staff_Memeber
- Accountant
- User
- Delivery_Person

M.G.T Rashmika (IT21167546) -->

- Feedback
- Notification
- Inquiry

Gamlathge G.G.A.U (IT21173868) -->

- Package
- Subscription
- Customer

Exercise 3

```
#include <iostream>

#include <string>


#define MAX_REPORT_INVOICE_QUANTITY 10

#define MAX_REPORT_ORDER_QUANTITY 10

#define MAX_ORDER_ITEM_SIZE 10

#define MAX_CART_ITEM_SIZE 10


using namespace std;


/* ------------------------- Class Definitions --------------------*/


/*------------------------- User class Definition IT21166556 ----------*/


class User{

	protected:

		string name;

		string gender;

		int age;


	public:

		virtual void login() = 0;

		void display();

};


/*---------------------------- Staff_Member class Definition IT21166556 ----------------------*/
```

```
class Staff_Member : public User{
	private:
		string staffID;
		string position;

	public:
		Staff_Member();
		Staff_Member(string ID, string posi, string n, string g, int a);
		void login();
		void display();
};

/*----------------------------- Accountant class Definition IT21166556 ----------------------*/

class Accountant : public User{
	private:
		string accountantID;
		string firm;

	public:
		Accountant();
		Accountant(string Aid, string frm, string n, string g, int a);
		void login();
		void display();
};

/*----------------------------- Delivery_Person class Definition IT21166556 ----------------------*/
```

```
class Delivery_Person : public User{
        private:
                string deliveryID;
                string firm;

        public:
                Delivery_Person();
                Delivery_Person(string Did, string frm, string n, string g, int a);
                void login();
                void display();
};

/*------------------------------Package class Definition IT21173868----------------------*/

class Package {

private:
                string packageID;
                string pName;


public:
                Package();
                Package(string pID, string name);
                void addpackage();
                void editPackage();
                void display();
```

```
};

/*----------------------------Customer class Definition IT21173868---------------------*/

class Customer {

private:
                string customerID;
                int contactNo;
                string email;
                string address;


public:
                Customer();
                Customer(string CID,int CNO,string mail,string Caddress);
                void regiterUser();
                void login();
                void editUser();
                void display();

};
/*----------------------------Subscription class Definition IT21173868---------------------*/

class Subscription {

private:
```

```
                string subscriberID;
                string sPacakge;
                string duraiton;
                Customer *customer;
                Package *package;


public:
                Subscription();
                Subscription(string subID, string pkg,string dura);
                void displaySubscription();
                void display(Customer*customer,Package*package);
};

/*----------------------------Invoice Class Header IT21170270-------------------------------*/
class Invoice {
  private:
    string invoiceNo;          //aggrigation with Payment Class
    Customer *customer;       //association with Cusatomer Class
    Accountant *accountant;     //association with Accountant Class

  public:
    Invoice();
    Invoice(string no);
          void display();
    void getOrderDetails();
          void getCustomerDetails();
          void getPaymentDetails();
```

```
        ~Invoice();
};

/*------------------------------Payment Class Header IT21170270---------------------------------*/
class Payment
{
        private:
                Invoice *invoice;   //aggrigation with Invoice Class
                string paymentID;
                string paymentType;
                string PaymentDate;

        public:
    Payment();
                Payment(string payID , string payType , string patDate , string No );
                void display();
                void validatePayment();
                void getOrderDetails();
                ~Payment();
};

/*------------------------------Laundry_Item class Definition IT21167300 ----------------------*/

class Laundry_Item {
```

```
        private:

                string itemID;
                string itemName;
                float itemPrice;
                string itemDescription;

        public:

                Laundry_Item();
                Laundry_Item(string id, string name, float price, string desc);
                void display();
                void addItem();
                void editItem();
};

/*------------------------------ Cart class Definition IT21167300 ----------------------*/

class Cart {

        private:

                int itemQuantity;
                Laundry_Item * items[MAX_CART_ITEM_SIZE];          //association with
Laundry_Item class
                Customer * customer;
        //association with Customer class

        public:
```

```
        Cart();
        Cart(int quantity);
        void addLaundryItem( Laundry_Item *item, int index);
        void addCustomer(Customer * cus);
        void display();
        void calculatePrice();
        void removeItem();
};

/*---------------------------- Order class Definition IT21167300 ----------------------*/

class Order {

    private:

        string orderID;
        string oDate;
        int itemAmount;                                         // Laundry_Item amount in a single order
        Payment * payment;                                      // Composition with payment
        Laundry_Item * items[MAX_ORDER_ITEM_SIZE];          // Aggregation with Laundry_Items
        Cart *cartOrder;                                        // Association with cart
        Customer *customer;                                     // Association Customer
        Delivery_Person *delivery;                          // Association Delivery_Person
        Staff_Member *staff;                                    // Association Staff_Member
```

```
	public:

		Order();
		Order(string oID, string oD, int amount, string payID, string payType, string payDate,
string No);
		void addLaundryItem( Laundry_Item *item, int index); // Add Laundry_Items to the
order
		void addCustomer(Customer *cus);
		void addDelivery_Person(Delivery_Person *d);
		void addStaff_Member(Staff_Member *s);
		void display();
		void checkAvailability();
		void placeOrder();
		void orderValidation();
		void cancelOrder();
		~Order();
};
/*-----------------------------Report Class Header IT21170270---------------------------------*/

class Report
{
	private:
		string reportID;
		string reportType;
		Invoice *invoice[MAX_REPORT_INVOICE_QUANTITY];   //aggrigation with Payment
Class
		Order *order[MAX_REPORT_ORDER_QUANTITY];      //association with Payment
Class
```

```
        public:
           Report();
                Report(string ID , string );
                void display();
                void displayOrdersReports();
                void displayInvoiceReports();
                void displayLaundaryItemReports();
};
/*-----------------------------Inquiry Class Header IT21167546 ---------------------------------*/

class Inquiry
{
        private:
                int inquiryNo;

        public:
           Inquiry();
                Inquiry(int no);
                display();
           void getCustomerDetails(Customer *customer);
                void setInquiryReply();
};

/*-----------------------------Feedback Class Header IT21167546--------------------------------*/
class feedback
{
        private:
                string feedbackNo;
```

```
        string feedbackReply;


    public:
        feedback();
        feedback(string fNo , string fReply);
        void getCustomerDetails(Customer *customer);
        void display();
        void setfeedbackReply();

};

/*----------------------------Notification Class Header IT21167546--------------------------------*/


class Notification
{
    private:
        string notificationNo;
        Customer *customer;
    public:
        Notification();
        Notification(string nNo /*, Customer *nCus*/);
        void display();
        void viewDiscription();

};

/* ######################### Class Implementations ######################*/
```

```
/*----------------------------Staff_Member class Implementation-------------------------*/


Staff_Member::Staff_Member(){
	staffID = " ";
	position = " ";
	name = " ";
	gender = " ";
	age = 0;


}
Staff_Member::Staff_Member(string ID, string posi, string n, string g, int a)
{
	staffID = ID;
	position = posi;
	name = n;
	gender = g;
	age = a;
}
void Staff_Member::login(){


}
void Staff_Member::display(){
	cout<<"Staff Member details -> " << endl;
	cout << "Staff Member Name :" << name << endl;
	cout << "Staff Member Gender :" << gender << endl;
	cout << "Staff Member Age :" << age << endl;
	cout << "Staff Member ID :" << staffID << endl;
```

```
	cout << "Staff Member Position :" << position << endl << endl;
}

/*----------------------------Accountant class Implementation-------------------------*/

Accountant::Accountant(){
	accountantID = "";
	firm = "";
	name = "";
	gender = "";
	age = 0;
}
Accountant::Accountant(string Aid, string frm, string an, string ag, int aa){
	accountantID = Aid;
	firm = frm;
	name = an;
	gender = ag;
	age = aa;
}
void Accountant::login(){

}
void Accountant::display(){
	cout <<"Accountant details -> " << endl;
	cout << "Accountant Name :" << name << endl;
	cout << "Accountant Gender :" << gender << endl;
	cout << "Accountant Age :" << age << endl;
	cout << "Accountant ID:" << accountantID << endl;
```

```
        cout << "Accountant Firm :" << firm << endl << endl;
}

/*----------------------------Delivery_Person class Implementation-----------------------*/

Delivery_Person::Delivery_Person(){
        deliveryID = "";
        firm = "";
        name = "";
        gender = "";
        age = 0;
}
Delivery_Person::Delivery_Person(string Did, string frm, string n, string g, int a){
        deliveryID = Did;
        firm = frm;
        name = n;
        gender = g;
        age = a;
}
void Delivery_Person::login(){

}
void Delivery_Person::display(){
        cout <<"Delivery Person details -> " << endl;
        cout << "Delivery Person Name :" << name << endl;
        cout << "Delivery Person Gender :" << gender << endl;
        cout << "Delivery Person Age :" << age << endl;
        cout << "Delivery Person ID:" << deliveryID << endl;
```

```
        cout << "Delivery Person Firm :" << firm << endl << endl;
}

/*----------------------------Package class implementation---------------------*/

Package :: Package (){

        packageID = "";
        pName = "";
}

Package :: Package(string pID, string name){

                packageID = pID ;
                pName = name ;
}
void Package ::addpackage(){

}
void Package ::editPackage(){

}
void Package ::display(){

        cout << endl << "Package Info " << endl ;
        cout << "Package ID : " << packageID << endl ;
        cout << "Package Name : " << pName << endl << endl ;
}
```

```
/*----------------------------Customer class implementation--------------------*/

Customer :: Customer(){

        customerID = "";
   contactNo = 0;
        email = "";
        address = "";



}
Customer :: Customer(string CID,int CNO,string mail,string Caddress){

        customerID = CID;
        contactNo = CNO;
        email    = mail;
        address = Caddress;
}
void Customer ::regiterUser(){

}
void Customer ::login(){

}
void Customer ::editUser(){
```

```
}
void Customer ::display(){

        cout << endl << "customer Info" << endl ;
        cout << "customerID : " << customerID << endl ;
        cout << "Contact NO : " << contactNo << endl  ;
        cout << "email : " << email << endl ;
        cout << "Address : " << address << endl << endl  ;
}

/*-----------------------------Subscription class implementation----------------------*/
Subscription :: Subscription(){
        subscriberID = "";
        sPacakge = "";
        duraiton = "";

}
Subscription ::   Subscription(string subID, string pkg,string dura){

        subscriberID = subID;
        sPacakge = pkg;
        duraiton = dura;

}
void Subscription ::displaySubscription(){

}
void Subscription ::display(Customer*customer,Package*package){
```

```
	cout << endl << "Subscriber Info" << endl ;
	cout << "subscriberID : " << subscriberID << endl ;
	cout << "Subscriber Pacakge  : " << sPacakge << endl  ;
	cout << "Duration : " << duraiton << endl << endl;

	customer->display();
	package->display();

}

/*----------------------------Report class Implemnetation--------------------------------*/

Report::Report()
{
	 reportType = "";
	reportID = "";
}

Report::Report(string ID , string type)
{
	reportID = ID;
	reportType = type;
}

void Report:: display()
{
	cout << endl << "Report Infro ->" << endl ;
	cout << "Report ID   : " << reportID << endl ;
```

```
    cout <<"Report Type : " << reportType << endl << endl ;
}

void Report::displayOrdersReports()
{

}
void displayInvoiceReports()
{

}
void displayLaundaryItemReports()
{

}

/*-----------------------------Invoice class Implemnetation--------------------------------*/

Invoice::Invoice()

{
        invoiceNo = "";
}

Invoice::Invoice(string No)
{
        invoiceNo = No ;
}
```

```
void Invoice::display()
{
        cout << "Invoice Infor ->" << endl ;
        cout << "Invoice No   : " << invoiceNo << endl << endl ;
}



void Invoice::getOrderDetails()
{


}

void Invoice::getCustomerDetails()
{


}

void Invoice::getPaymentDetails()
{


}

Invoice::~Invoice()
{
                cout << "Deleted Invoice " << invoiceNo << endl ;
}
```

```
/*----------------------------Payment class implementation--------------------------------*/


Payment::Payment()

{

	invoice = new Invoice("");

	paymentID = "" ;

	paymentType = "";

	PaymentDate = "";

}


Payment::Payment(string payID , string payType , string payDate , string No)

{

   invoice = new Invoice(No);

   paymentID = payID ;

	paymentType = payType;

	PaymentDate = payDate;

}


void Payment::display()

{

	cout << "Payment Infro ->" << endl ;

	cout << "Patment ID   : " << paymentID << endl ;

   cout <<"Payment Type : " << paymentType << endl ;

	cout << "payment Date : " << PaymentDate << endl ;

	invoice->display();

}


void Payment::validatePayment()
```

```
{


}

void Payment::getOrderDetails()
{


}

Payment::~Payment()
{
	cout << "Deleted Payment " << paymentID << endl ;
	delete invoice;
}

/*-----------------------------Laundry_Item class Implementation-------------------------*/

Laundry_Item::Laundry_Item(){

	itemID = "";
	itemName = "";
	itemPrice = 0.0;
	itemDescription = "";
};

Laundry_Item::Laundry_Item(string id, string name, float price, string desc){
	itemID = id;
	itemName = name;
```

```
	itemPrice = price;
	itemDescription = desc;
};

void Laundry_Item::display(){
	cout<<"Item ID : "<<itemID << endl;
	cout<<"Item Name : "<<itemName << endl;
	cout<<"Item Price : "<<itemPrice << endl;
	cout<<"Item Description : "<<itemDescription <<endl<<endl;
};

/*----------------------------- Cart class Implementation-------------------------*/

Cart::Cart(){
	itemQuantity = 0;
};

Cart::Cart(int quantity){
	itemQuantity = quantity;
};

void Cart::addLaundryItem( Laundry_Item *item, int index){
	items[index] = item;
};

void Cart::addCustomer(Customer * cus){
	customer = cus;
};
```

```
void Cart::display(){
	cout<<"Cart Details >>" << endl;
	cout<< "Cart Item Quantity : "<<itemQuantity << endl;
	for ( int i = 0; i < itemQuantity; i++){
		items[i] -> display();
	};
	cout<<endl;
}

/*---------------------------- Order class Implementation-------------------------*/

Order::Order(){
	orderID = "";
	oDate = "";
	itemAmount = 0;
	payment = new Payment();
};

Order::Order(string oID, string oD, int amount, string payID, string payType, string payDate, string
No){
	orderID = oID;
	oDate = oD;
	itemAmount = amount;
	payment = new Payment(payID,payType,payDate,No);
};

void Order::addLaundryItem( Laundry_Item *item, int index){
```

```
        items[index] = item;
};

void Order::addCustomer(Customer *cus){
        customer = cus;
};

void Order::addDelivery_Person(Delivery_Person *d){
        delivery = d;
};

void Order::addStaff_Member(Staff_Member *s){
        staff = s;
};

void Order::display (){
        cout<<"Order Info ->"<< endl;
        cout<<"Order ID   :"<< orderID << endl;
        cout<<"Order Date :"<< oDate << endl;
        cout<<"Order Item Amount :"<< itemAmount << endl<<endl;
        cout<<"Order Item Details >>" << endl;
        for ( int i = 0; i < itemAmount ; i++){
                items[i] -> display();
        };
        cout<<endl;
        payment -> display();
};
```

```
Order::~Order(){
	cout << "Deleted Order "<< orderID <<endl;
	delete payment;
};

/*----------------------------Feedback class implementation-------------------------------*/



feedback::feedback(){
	feedbackNo= "";
	feedbackReply="";
}

feedback::feedback(string fNo , string fReply){
	feedbackNo=fNo;
	feedbackReply=fReply;

}



void feedback::getCustomerDetails(Customer *customer)
{
	customer->display();

}
```

```
void feedback::display(){

	cout<<"Customer Feedback "<< endl;
	cout<<"Feedback Number :" << feedbackNo << endl;
	cout<<"Feedback : " << feedbackReply << endl;



}

void feedback::setfeedbackReply(){



}




/*----------------------------Notification class implementation--------------------------------*/



Notification::Notification(){

	notificationNo = "";

}

Notification::Notification(string nNo /*, Customer *nCus*/){
```

```
	notificationNo = nNo;
	//customer = nCus;
}

void Notification::display(){

	cout << "\n" ;
	cout << "Notification Number : " << notificationNo << endl ;
}

void Notification::viewDiscription(){



}



/*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Main
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/

int main (){

/*---------------------- IT21166556 Nadeeshan I.U.N --------------------*/
	Staff_Member *s;
	s = new Staff_Member("S1011", "Washer", "Saman", "Male", 28);
	s -> display();

	Accountant *a;
	a = new Accountant("A1234", "Finance", "Sashi", "Female", 25);
```

```
        a -> display();


        Delivery_Person *d;
        d = new Delivery_Person("D1234", "Rider", "Kasun", "Male", 30);
        d -> display();

/*---------------------- IT21173868 Gamlathge G.G.A.U --------------------*/
        Package *pkg ;
        pkg = new Package("P1000", "Silver");
        pkg -> display();


        Customer *cus ;
        cus = new Customer("C1000", 76785643,"nima@gmail.com","NO 123 KAndy");
        cus -> display();


        Subscription *sub ;
        sub = new Subscription("S1000", "Silver","2 year");
        sub -> display(cus,pkg);


        delete pkg;
        delete cus;
        delete sub;

/*---------------------- IT21170270 Maduranga D.B.W.N --------------------*/

/*----------------------------Payment class--------------------------------*/

        Payment * payObj;
```

```
        payObj = new Payment("P1020" , "Paypal" , "2021/05/20" , "I1020");
        payObj -> display();  //Display sample details


/*-----------------------------Report class---------------------------------*/

  Report * repoObj1;
  repoObj1 = new Report("R567" , "InvoiceFile");
  repoObj1->display();   //Display sample details


/*-----------------------------Invoice class---------------------------------*/

  Invoice * invoObj;
  invoObj = new Invoice("I2033");
  invoObj->display();

/*-----------------------------Delete Objects---------------------------------*/

  delete payObj;
  delete repoObj1;
  delete invoObj;

/*----------------------  IT21167300 Sumanasekara PDM--------------------*/

        Laundry_Item *l1 = new Laundry_Item("L001","Shirt",200.00,"Shirt Wash");
        Laundry_Item *l2 = new Laundry_Item("L002","Short",300.00,"Short Wash");
```

```
        Cart * cart = new Cart(2);
        cart -> addLaundryItem(l1,0);
        cart -> addLaundryItem(l2,1);
        cart -> display();

        Order * order = new Order("O1001","2021/05/16", 2,
"P1001","Paypal","2021/05/16","l1001");
        order -> addLaundryItem(l1,0);
        order -> addLaundryItem(l2,1);
        order -> display();
        delete order;
        delete l1,l2;
        delete cart;

/*-------------------------------- M.G.T Rashmika IT21167546 --------------------*/

/*----------------------------Inquiry class-------------------------------*/

        Inquiry * inquObj;
        //inquObj = new Inquiry(78);
        //inquObj->display();

/*----------------------------Feedback class-------------------------------*/

        feedback *feedbackobj;
        feedbackobj = new feedback("678","Service was good as well");
        feedbackobj->display();
```

```
/*----------------------------Notification class--------------------------------*/


    Notification *notifiobj;

    notifiobj = new Notification("678");

            notifiobj->display();




   delete inquObj;

   delete feedbackobj;


   return 0;

}
```