Topic **:** Online Bookstore

Group no **:** MLB_03.02_11

Campus **:** Malabe

Submission Date: 5/19/2022

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**
IT21240942	Wijethunga R.D.K. G	0716846441
IT21239298	Bandara E.M.S. S	0712476454
IT21240706	Dhananjana B.K. T	0776927338
IT21238512	Soysa W.M. Y	0703968965
IT21239670	Bandara D.M.H.G.I. G	0716418132

User requirements

1. Users are of two types called publishers and customers and they can login to the website by entering the correct username and password.
2. Once the user gets registered, they can view, delete or edit their profiles.
3. Users can search books and filter results by category, price and author on the website.
4. Customers can view the descriptions of books, authors, categories and publishers before purchasing books.
5. Customers can add their preferred books to cart.
6. Customers can make orders for books and make payment with a credit card or a debit card.
7. Publishers can upload their books to the website after admin approve them.
8. Publishers can withdraw money at the end of the month.
9. Members can give feedbacks about the website and services provided.
10. Users can submit an inquiry to the admin if they have any issues.
11. Admin also needs to login to the website by entering the username or email and password.
12. Admin can approve books uploaded by the publisher.
13. Admin can view member statistics and income statistics.
14. Admin can delete member accounts and existing books.
15. Admin can generate reports.
16. Admin can view inquiries of customers and publishers and reply to those inquiries with relevant solutions.

Noun/ Verb Analysis

Nouns- 

Verbs - 

1. Users are of two types called publishers and customers and they can login to the website by entering the correct username and password.
2. Once the user gets registered, they can view, delete or edit their profiles.
3. Users can search books and filter results by category, price and author on the website.
4. Customers can view the descriptions of books, authors, categories and publishers before purchasing books.
5. Customers can add their preferred books to cart.
6. Customers can make orders for books and make payment with a credit card or a debit card.
7. Publishers can upload their books to the website after admin approve them.
8. Publishers can withdraw money at the end of the month.
9. Members can give feedbacks about the website and services provided.
10. Users can submit an inquiry to the admin if they have any issues.
11. Admin also needs to login to the website by entering the username or email and password.
12. Admin can approve books uploaded by the publisher.
13. Admin can view member statistics and income statistics.
14. Admin can delete member accounts and existing books.
15. Admin can generate reports.
16. Admin can view inquiries of customers and publishers and reply to those inquiries with relevant solutions.

Nouns	Verbs
Website Username First name Last name Email Password User Publisher Customer Profile Book Result Category Price Author Description Cart Order Payment Credit card Debit card Admin Money Month Member Feedback Service Inquiry Issues Member statistics Income statistics Member accounts Existing books Reports Solution	Sign up Login Entering Registered View Delete Edit Search Filter Purchasing Add Make Upload Approve Withdraw Give Submit Generate Reply

Identified classes by noun verb analysis

Reasons for rejecting nouns

- **Redundant –** In an Online Book Store both ‘member’ and ‘user’ refers to the same thing.
- **An event or an operation –** Viewing member statistics and income statistics, and deleting member accounts and existing books are the operations performed by the admin.
  Filtering search results is an operation performed by the users of the library system.
- **Outside scope of system –** website, profile, money, month, services, issues, solutions
- **Meta language –** existing books
- **Attributes –** Username, first name, last name, email and password are attributes of customer, publisher and admin
  Category, price, author, description are attributes of books
  Credit card and Debit card are attributes of payment

Classes

1. Customer
2. Publisher
3. Admin
4. Book
5. Order
6. Inquiries
7. Payment
8. Report
9. Cart
10. Feedback

CRC cards

Customer	
Responsibilities	**Collaboration**
Login providing details	
View profile	
Delete profile	
Edit profile	
Set password	
Purchase a book	Book

Feedback	
Responsibilities	**Collaboration**
Add feedback	Customer, Publisher
Display feedback	

Book	
Responsibilities	**Collaboration**
Add book details	Publisher
Update book details	Publisher
Delete book details	Admin
Check the validity of books	Admin

Cart	
Responsibilities	**Collaboration**
Add preferred book	Customer
Remove book	Customer
Add quantity	Customer

Admin	
Responsibilities	**Collaborations**
Approve books	
View inquiry	Inquiry
Remove existing books	Book
Manage user account	Customer, Publisher

Report	
Responsibilities	**Collaborations**
Store report details	Admin
Display report details	

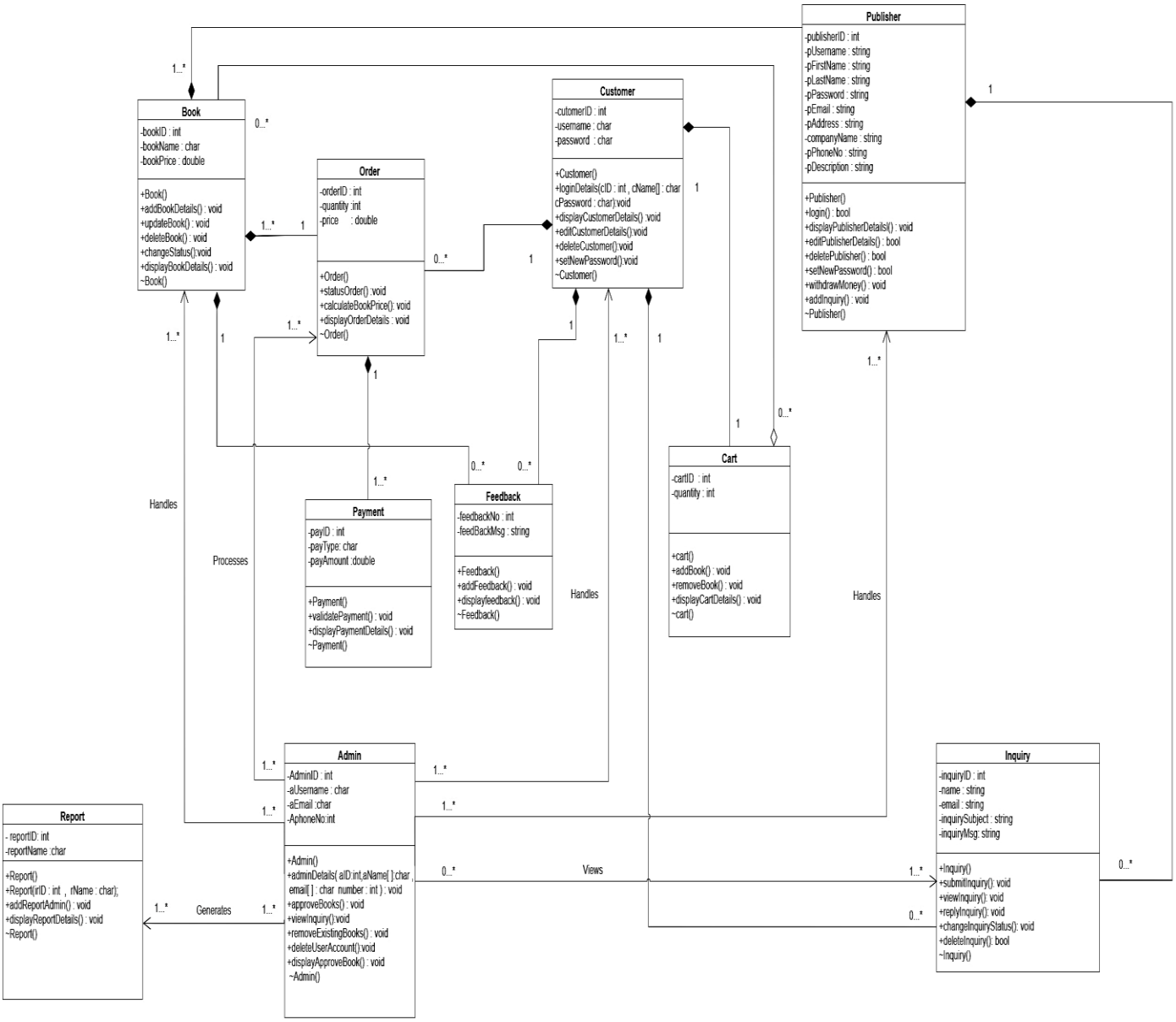
Inquiry	
Responsibility	**Collaborations**
Submit inquiry	Publisher, Customer
View inquiry	Admin
Reply to the inquiry	Admin
Update the status of the inquiry	Admin
Delete inquiry	Publisher, Customer

Publisher	
Responsibility	**Collaborations**
Login to the system	
View profile	
Edit profile	
Delete profile	
Set new password	
Withdraw money	
Add inquiry	Inquiry

Order	
Responsibilities	**Collaboration**
Place an order	
Status of order	
Confirm order	Payment

Payment	
Responsibilities	**Collaboration**
Validate payment	
Store payment details	

Class diagram



Code..

```
#include <iostream>
#include <cstring>
#define SIZE 10
using namespace std;
// inquiry.h
class Inquiry {

protected:
    int inquiryID;
    string name;
    string email;
    string inquirySubject;
    string inquiryMsg;

public:
    Inquiry();
    Inquiry(int id, string iName, string iEmail, string iSubject, string iMsg);
    void submitInquiry();
    void viewInquiry();
    void replyInquiry();
    void inquiryStatus();
    bool deleteInquiry();
    ~Inquiry();
```

```
};

//Payment.h
class Payment {
private:
    int payID;
    char payType[20];
    double payAmount;



public:
    Payment();
    Payment(int pID, const char ppayType[], double ppayAmount);
    void checkPayment();
    void confirmPayment();
    void displayPaymentDetails();
    ~Payment();
};
```

```
//Order.h
class Order
{
private:

    int orderID;
    int orderQty;
    double price;
    Payment* pymnt[2];
public:
    Order();
    Order(double num1, double num2);
    void statusOder();
    void calculateBookPrice();
    int getOrderID();
    void displayOderDetails();
    ~Order();
};
```

```
//Publisher.h
class Publisher {
private:
    int publisherID;
    string pUsername;
    string pFirstName;
    string pLastName;
    string pPassword;
    string pEmail;
    string pAddress;
    string companyName;
    string pPhoneNo;
    string pDescription;
    Inquiry* inquiries[SIZE];
    int noOfInquiries;
public:
    Publisher();
    Publisher(int pID, string uName, string fName, string lName, string psw,
      string email, string address, string compName, string phone, string
      description);
    bool login(string uName, string psw);
    void displayPublisherDetails();
```

```
    bool editPublisherDetails(string newUserName, string newFirstName, string
      newLastName, string newEmail, string newAddress, string newCompName,
      string newPhone, string newDescription);
    bool deletePublisher();
    bool setNewPassword(string oldPassword, string newPassword);
    void withdrawMoney(double amount);
    void addInquiry(Inquiry* newInquiry);
    void displayInquiries();
    ~Publisher();
};

//FeedBack.h
class FeedBack
{
private:
    int feedBackNo;
    string feedBackMsg;
public:

    FeedBack();
    void addFeedBack();
    void displayFeedBack();
    ~FeedBack();

};
```

```
//Report.h
class Report
{
private:
    int reportID;
    char reportName[50];
public:
    Report();
    Report(int rID, const char rName[]);
    void addReportAdmin();
    void displayReportDetails();
    ~Report();
};
//Book.h
class Book
{
private:
    int bookID;
    char bookName;
    double bookPrice;

    Report* bReport[2];
    FeedBack* bFeed[2];
    Order* bOrder[2];
    Publisher* bPublisher[2];
```

```
public:

    Book();
    Book(int bID, const char bName, double bPrice, int bRpt1, int bRpt2, int
      feed1, int feed2, int order1, int order2, int bPub1, int bPub2);
    void  displayBookDetails();
    void addBookDetaiks();
    void updateBook();
    void deleteBook();
    void changeStatus();
    ~Book();
};
//Cart.h
class Cart
{
private:
    int cartID;
    int quantity;
    Book* books[2];
public:
    Cart();
    void removeBook();
    void addBook(int cID, int qunty, Book* books1, Book* books2);
    void displayCartDetails();
    ~Cart();
};
```

```
// customer.h
class Customer
{
private :
    int customerId;
    char username[50];
    char password[10];

    Cart* crt[2];
    FeedBack* bFeed[2];
    Order* Ord[2];
    Report* Rpt[2];
    Inquiry* inquiry[2];
public:
    Customer();
    Customer(int cart1, int cart2, int feed1, int feed2, int ord1, int ord2, int rep1,
      int rep2 ,int inq1 , int inq2);
    void loginDetails(int cID , const char cName , char const cPassword[]);
    void displayCustomerDetails();
    void editCustomerDetails();
    void deleteCustomer();
    void setNewPassword();
    ~Customer();

};
```

```
//Admin.h
class Admin
{
private:
    int adminID;
    char aUsername[50];
    char aEmail[30];
    int aPhoneNo;
    Report* rep;
    Book* bk;
    Customer* cust;
    Publisher* pub;
    Inquiry* inq;
    Order* odr;

public:
    Admin();
    void adminDetails(int aID, const char aName[], const char email[], int
      number);
    void approveBooks();
    void viewInquiry();
    void removeExistingBooks();
    void displayAdminDetails();
    void deleteUserAccount();
    void displayApproveBook();
```

```
    ~Admin();
};
// implementation .....

//customer implementation
  Customer::Customer()
   {
      crt[0] = new Cart[001];
      crt[1] = new Cart[002];

      bFeed[0] = new FeedBack[111];
      bFeed[1] = new FeedBack[222];

      Ord[0] = new Order[011];
      Ord[1] = new Order[022];

      Rpt[0] = new Report[010];
      Rpt[1] = new Report[020];

      inquiry[0] = new Inquiry[100];
      inquiry[1] = new Inquiry[101];

   }
  Customer::Customer(int cart1, int cart2, int feed1, int feed2, int ord1, int ord2,
     int rep1, int rep2, int inq1, int inq2)
  {
```

```
    crt[0] = new Cart[cart1];
    crt[1] = new Cart[cart2];

    bFeed[0] = new FeedBack[feed1];
    bFeed[1] = new FeedBack[feed2];

    Ord[0] = new Order[ord1];
    Ord[1] = new Order[ord2];

    Rpt[0] = new Report[rep1];
    Rpt[1] = new Report[rep2];

    inquiry[0] = new Inquiry[inq1];
    inquiry[1] = new Inquiry[inq2];
}
  void  Customer :: loginDetails(int cID, const char cName, char const
    cPassword[])
 {

 }
  void Customer::displayCustomerDetails()
  {

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

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

}
void Customer::setNewPassword()
{

}
Customer ::  ~Customer()
{
    cout << "Deleting customer id" << endl;
    {
        delete crt[0];
        delete crt[1];

        delete bFeed[0];
        delete bFeed[1];

        delete Ord[0];
        delete Ord[1];

        delete Rpt[0];
        delete Rpt[1];
```

```
            cout << "the end " << endl;


        }
    }



//Report Implementation
Report::Report()
{


}
Report::Report(int rID, const char rName[])
{


}
void Report::addReportAdmin()
{


}
void Report::displayReportDetails()
{


}
Report :: ~Report()
{
```

```
}

//Admin Implementation
Admin::Admin()
{


}
void Admin::adminDetails(int aID, const char aName[], const char email[], int
      number)
{


}
void Admin::approveBooks()
{


}
void Admin::viewInquiry()
{


}
void Admin::removeExistingBooks()
{


}
void Admin::displayAdminDetails()
{
```

```
}
void Admin::deleteUserAccount()
{

}
void Admin::displayApproveBook()
{

}
Admin :: ~Admin()
{

}

//FeedBack implementation
   FeedBack :: FeedBack()
   {

   }
   void FeedBack::addFeedBack()
   {

   }
   void FeedBack::displayFeedBack()
   {
```

```
}
FeedBack ::  ~FeedBack()
{

}
//Book implementation
Book::Book()
{
    bReport[0] = new Report[123];
    bReport[1] = new Report[456];

    bFeed[0] = new FeedBack[001];
    bFeed[1] = new FeedBack[002];

    bOrder[0] = new Order[0101];
    bOrder[1] = new Order[0202];

    bPublisher[0] = new Publisher[111];
    bPublisher[1] = new Publisher[222];
}

Book:: Book(int bID, const char bName, double bPrice, int  bRpt1, int bRpt2,
  int feed1, int feed2, int order1, int order2, int bPub1, int bPub2)
{
    bReport[0] = new Report[bRpt1];
```

```
    bReport[1] = new Report[bRpt2];

    bFeed[0] = new FeedBack[feed1];
    bFeed[1] = new FeedBack[feed2];

    bOrder[0] = new Order[order1];
    bOrder[1] = new Order[order2];

    bPublisher[0] = new Publisher[bPub1];
    bPublisher[1] = new Publisher[bPub2];
}

void Book::addBookDetaiks()
{
}
void Book::updateBook()
{

}
void Book::deleteBook()
{

}
void Book::changeStatus()
{
```

```
}
void Book::displayBookDetails()
{

}



Book :: ~Book()
{
   cout << "Deleting Book ID " << endl;
   {
      delete bReport[0];
      delete bReport[1];

      delete bFeed[0];
      delete bFeed[1];

      delete bOrder[0];
      delete bOrder[1];

      delete bPublisher[0];
      delete bPublisher[1];

      cout << "The end" << endl;
   }
```

```
// cart implementation

Cart::Cart()
{

}
void Cart::addBook(int cID, int qunty, Book* books1, Book* books2)
{
    books[0] = books1;
    books[1] = books2;
}
void Cart::removeBook()
{

}
void Cart::displayCartDetails()
{
    for (int i = 0; i < 2; i++)
        books[i]->displayBookDetails();

}
Cart :: ~Cart()
{
    cout << "Deleting Cart details" << endl;

}
```

```
// payment implementation
Payment::Payment()
{
   payID = 0;
   strcpy_s(payType, "");
   payAmount = 0;
}
Payment::Payment(int pID, const char ppayType[], double ppayAmount)
{
   payID = pID;
   strcpy_s(payType, ppayType);
   payAmount = ppayAmount;
}

void Payment::checkPayment()
{
}
void Payment::confirmPayment()
{
}
void Payment::displayPaymentDetails()
{
}
Payment::~Payment()
```

```
{
}
// order implementation
 Order::Order()
{
}
 Order::Order(double num1, double num2)
 {
     pymnt[0] = new Payment[1500];
     pymnt[1] = new Payment[750];
 }
void Order::statusOder()
{
}
void Order::calculateBookPrice()
{
}
void Order::displayOderDetails()
{
}
int Order::getOrderID()
{
    return orderID;
}
Order::~Order()
{
```

```
    cout << "Delete order details " << endl;
    {
        delete pymnt[0];
        delete pymnt[1];
    }
}

// Publisher implementation

Publisher::Publisher() {

    publisherID = 0;
    pUsername = "";
    pFirstName = "";
    pLastName = "";
    pPassword = "";
    pEmail = "";
    pAddress = "";
    companyName = "";
    pPhoneNo = "";
    pDescription = "";
    noOfInquiries = 0;
}
```

```
Publisher::Publisher(int pID, string uName, string fName, string lName, string
      psw, string email, string address, string compName, string phone, string
      description) {
   publisherID = pID;
   pUsername = uName;
   pFirstName = fName;
   pLastName = lName;
   pPassword = psw;
   pEmail = email;
   pAddress = address;
   companyName = compName;
   pPhoneNo = phone;
   pDescription = description;
   noOfInquiries = 0;
}
bool Publisher::login(string uName, string psw)
{
   return true;
}


void Publisher::displayPublisherDetails() {}


bool Publisher::editPublisherDetails(string newUserName, string newFirstName,
      string newLastName, string newEmail, string newAddress, string
      newCompName, string newPhone, string newDescription)
{
   return true;
```

```
}

bool Publisher::deletePublisher()
{
    return true;
}

bool Publisher::setNewPassword(string oldPassword, string newPassword)
{
    return true;
}

void Publisher::withdrawMoney(double amount) {}

void Publisher::addInquiry(Inquiry* newInquiry) {
    inquiries[noOfInquiries++] = newInquiry;
}

void Publisher::displayInquiries() {}

Publisher::~Publisher() {
    cout << "Deleting Publisher " << publisherID << endl;
}
```

```
// Inquiry implementation

Inquiry::Inquiry() {
    inquiryID = 0;
    name = "";
    email = "";
    inquirySubject = "";
    inquiryMsg = "";
}

Inquiry::Inquiry(int id, string iName, string iEmail, string iSubject, string iMsg) {
    inquiryID = id;
    name = iName;
    email = iEmail;
    inquirySubject = iSubject;
    inquiryMsg = iMsg;
}

void Inquiry::submitInquiry() {}

void Inquiry::viewInquiry() {}

void Inquiry::replyInquiry() {}

void Inquiry::inquiryStatus() {}
```

```
bool Inquiry::deleteInquiry()
{
    return true;
}

Inquiry::~Inquiry() {
    cout << "Deleting Inquiry " << inquiryID << endl;
}



// main program.....................
int main()
{
    Inquiry* inquiry1 = new Inquiry(1, "Yashodini Zoyza",
      "yashodini@gmail.com", "Can't track my shipments", "I can't locate the
      tracking number given to me on the shipping website");


    Inquiry* inquiry2 = new Inquiry(2, "Ayesha Fonseka", "ayesha@gmail.com",
      "Book is taking too long to be approved", "I have uploaded a book one week
      ago and it is not approved yet");
    Order* ord = new Order();


    Publisher* publisher1 =
        new Publisher(1, "YashodiniZoyza", "Yashodini", "Zoyza", "#$$%$%",
            "yashodini@gmail.com", "Panadura", "Zoysa publications",
            "07612334567", "Publishing great authors since 1997");
```

```
Publisher* publisher2 =
    new Publisher(2, "Ayesha Fonseka", "Ayesha", "Fonseka", "#$$%$%",
        "ayesha@gmail.com", "Galle", "Fonseka publications",
        "0771465789", "Proud publishers of great writers and gifted
   storytellers");
Report* report = new Report();
Book* book = new Book();
Cart* crt = new Cart();
Customer * cust = new Customer();
Admin* adm = new Admin();




publisher1->addInquiry(inquiry1);
publisher2->addInquiry(inquiry2);
ord->displayOderDetails();
report->displayReportDetails();
book->displayBookDetails();
crt->displayCartDetails();
cust-> displayCustomerDetails();
adm->displayAdminDetails();
adm->displayApproveBook();
```

```
    delete inquiry1;
    delete inquiry2;
    delete publisher1;
    delete publisher2;
    delete ord;
    delete book;
    delete crt;
    delete cust;

    return 0;


}
```

INDIVIDUAL CONTRIBUTION

Student_ID number	Contribution
IT21240942	-Customer class -Feedback class
IT21239298	-Book class -Cart class
IT21240706	-Admin class -Report class
IT21238512	-Publisher class -Inquiry class
IT21239670	-Payment class -Order class