



**Course Number: 420-CT2-AS**  
**OBJECT ORIENTED PROGRAMMING CONCEPTS**  
**Teacher: Alex Vilvert**  
**Session: Summer 2020**

**FINAL PROJECT**

**On**

**BookBiz Management System**

**Submitted to**

**Alex Vilvert**

**By**

**Rahul Pipaliya(2012728)**

## Contents

Project Description.....	3
Interface Design .....	4
1)Login Page .....	4
2)Employee Management Form .....	5
3)Sales Manager Form .....	6
4)Inventory Management Form.....	7
5)Order Management Form.....	8
Code Design .....	9
1)Project Structure.....	9
2)Database Design .....	10
Implementation .....	12
1)Validations .....	12
2)Operations .....	14
Conclusion.....	19

## Project Description

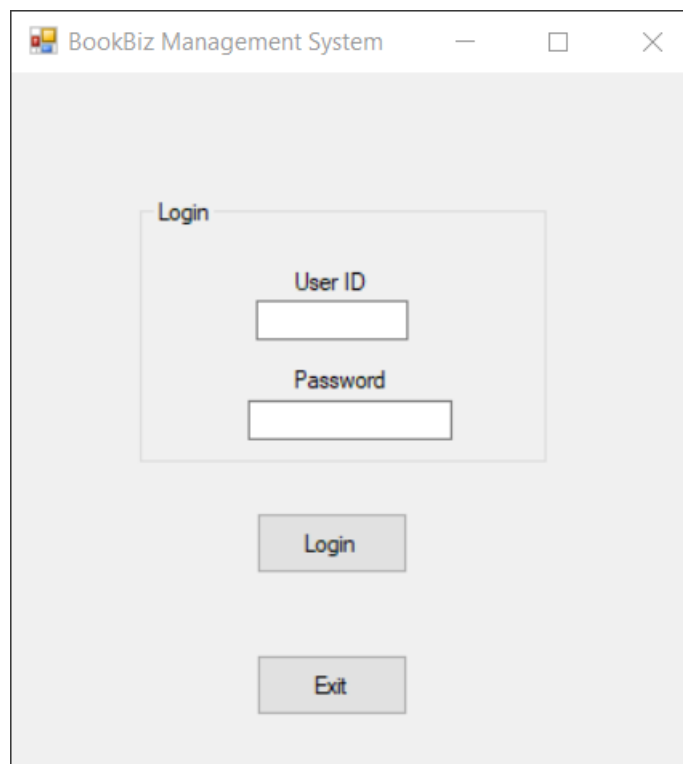
- This Project Aims to Create a Software in C# to help **BookBiz Distribution Inc.** to supply computer science books nearly all Quebec Universities and in colleges using a Windows based application **BookBiz Management System**.
- Currently in system we have 5 users and 4 user roles
  1. MIS Manager
  2. Sales Manager
  3. Inventory Controller
  4. Order Clerk

Users	Operations
MIS Manager (Henry Brown)	<ul style="list-style-type: none"><li>• Add user information</li><li>• Search/List user information</li><li>• Add employee information</li><li>• Search/list employee information</li></ul>
Sales Manager (Thomas Moore)	<ul style="list-style-type: none"><li>• Add client information</li><li>• Search/List client information</li></ul>
Inventory Controller (Peter Wang)	<ul style="list-style-type: none"><li>• Add/search/list book information and related information</li></ul>
Order Clerks <ul style="list-style-type: none"><li>- Mary Brown</li><li>- Jennifer Bouchard</li></ul>	<ul style="list-style-type: none"><li>• Add/Cancel clients' orders</li><li>• Search/List clients' orders</li></ul>

## Interface Design

### 1)Login Page

The first form user will be seeing is this login page. I have already set userid , password for above mentioned users. Each login will redirect to company's Employee/User to their Department Page.



The screenshot shows a window titled "BookBiz Management System". Inside the window, there is a login form. The form has a title "Login" and two input fields: "User ID" and "Password". Below the input fields are two buttons: "Login" and "Exit".

Eg. Login of henry brown will be redirected to Employee Management Form...etc.

Id/passwords:

Id	Password	Roles
henryb	henryb123	MIS Manager
thomasm1	thomas123	Sales Manager
peter123	peter123	Inventory Controller
maryb1	12345	Order Clerk
jennibo1	12345	Order Clerk

## 2)Employee Management Form

Using credentials of Henry Brown, you shall see this form.

The screenshot shows a web application window titled "Manager Portal". It contains several input fields and buttons for managing employees. The fields are organized into two rows: the first row has "First Name", "Last Name", and "Employee is :" (a dropdown menu); the second row has "User ID", "Password", and "Phone Number" (a field with a format mask "( ) - -"). Below these are a "Search by Name" field, a "Search" button, and a "Clear All Fields" button. To the right of the input fields are two large buttons: "Add Employee" and "List Employees". At the bottom, there is a table displaying a list of employees. An "Exit" button is located in the bottom right corner.

First Name	Last Name	Phone Number	Role
Henry	Brown	(514) 431-5658	mis_manager
thomas	moore	(111) 111-1111	sales_manager
peter	wang	(222) 222-2222	inventory_controller
mary	brown	(333) 333-3333	order_clerk
jennifer	bouchard	(444) 444-4444	order_clerk

Here, Managers Functionalities are.

1. Add new employees' information
2. assign userid and password to the employees
3. search employees using their name
4. view list of currently working Employees/users in the organization

### 3)Sales Manager Form

Using credentials of Thomas Moore, you shall see this form.

The screenshot shows a window titled "Sales Manager Form" with standard Windows window controls (minimize, maximize, close). The form contains several input fields and buttons. At the top, there are four input fields labeled "Name", "Street", "City", and "Phone Number". Below these are three more input fields labeled "Postal Code", "Fax number", and "Credit Limit". To the right of these fields is a large button labeled "Add Client". Below the input fields, there is a "Search" label next to an input field, followed by three buttons: "Search", "clear all fields", and "View List of clients". At the bottom of the form is a table with six columns: "Name", "Street", "City", "Postal Code", "Fax Number", and "Credit Limi". The table has eight rows, with the first row containing headers and the subsequent rows being empty. At the bottom right of the window is an "Exit" button.

Name	Street	City	Postal Code	Fax Number	Credit Limi

Here, Sales Manager can,

1. Add client's information
2. View organizations current clients
3. Search clients

#### 4)Inventory Management Form

credentials of peter wang shall redirect you to this form.

Manage Inventory of Bookbiz

**Book**

ISBN  Book Name  Unit Price  Quality On Hand

Category

**Authors**

First Name  Last Name  Publishers Name  Year Of Publication

Email Address

**Add To Inventory**

**List Inventory**

Search by Book Name  **Search** **Clear Fields** **Exit**

Book Title	Price	Quantity on hand	Author Name

Using this System Inventory Manager can..

1. Add Books Info
2. Add Author as well as Publishers info
3. View book information including current quantity on hand
4. Search book by name

## 5)Order Management Form

Using id/password of Mary Brown/ Jennifer Bouchard will lead you to this form

The screenshot shows a web application window titled "Order Management". It contains several input fields and buttons. At the top, there are three fields: "Book Name" (a dropdown menu), "Quantity" (a text input), and "Date" (a date picker showing "Wednesday, August 5"). Below these are "Clients Name" (a dropdown menu) and a "Taken by" section with three radio buttons: "Phone", "Email", and "Fax". To the right of the "Taken by" section is a "Reload List" button. Further right are two large buttons: "Place Order" and "Cancel Order". Below the "Clients Name" field, there is a text instruction: "to search/cancel an order please select book and client name from above dropdown". Below this instruction are two buttons: "Search" and "Clear". At the bottom of the form is a table with four columns: "Order Date", "Client Name", "Book Name", and an empty column. The table has five rows. At the bottom right of the window is an "Exit" button.

Order Date	Client Name	Book Name	

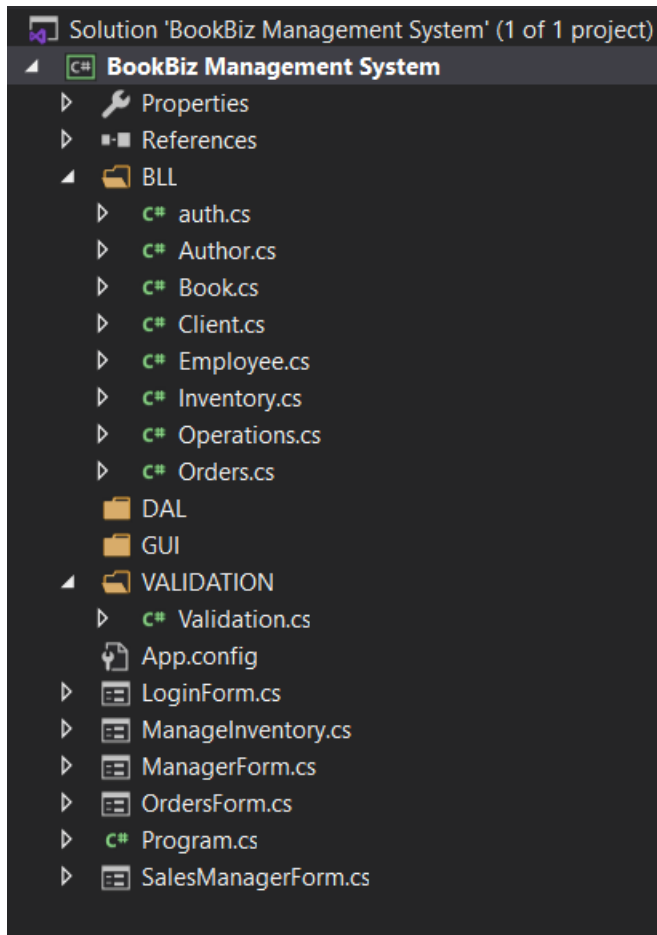
Using this form order clerk will be able to...

1. Add Clients Orders
2. View Clients Orders
3. Cancel Clients Orders
4. Search Clients Orders using Client Name and Book Name combined.












## Code Design

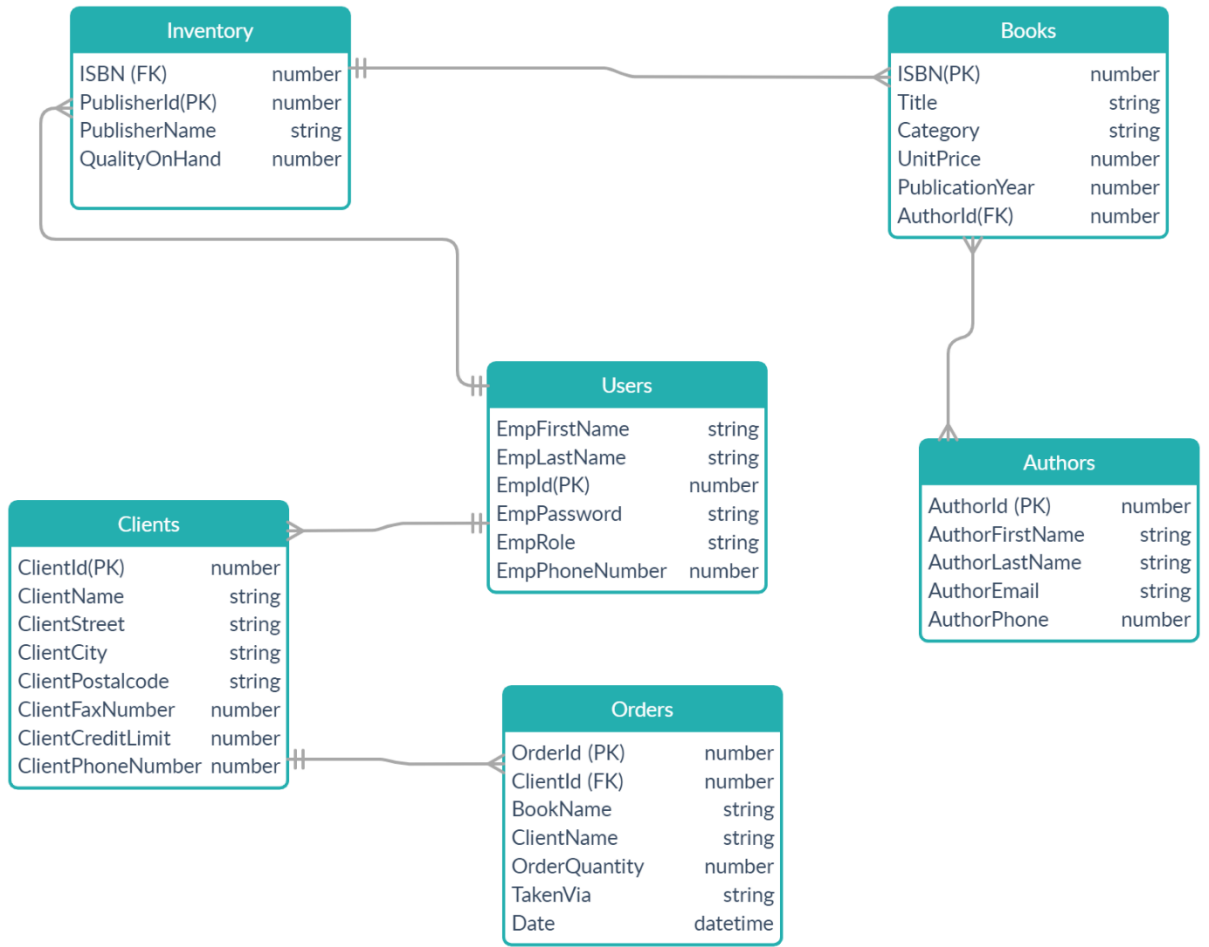
### 1)Project Structure



- \* All the classes, Database Operations are in the BLL.
- \* Validations are in VALIDATION
- \* Unfortunately, I could not place all the forms in GUI Folder it was not accessible for some reason. So, its outside but accessible.

## 2)Database Design

Name	Date modified	Type	Size
 BookBiz Management System.exe	8/5/2020 12:43 AM	Application	50 KB
 Authors.dat	8/4/2020 10:17 PM	DAT File	1 KB
 Books.dat	8/4/2020 10:17 PM	DAT File	1 KB
 Clients.dat	8/4/2020 10:10 PM	DAT File	1 KB
 Inventory.dat	8/4/2020 11:12 PM	DAT File	0 KB
 Orders.dat	8/4/2020 11:28 PM	DAT File	0 KB
 Users.dat	8/4/2020 4:37 PM	DAT File	1 KB
 BookBiz Management System.pdb	8/5/2020 12:43 AM	Program Debug D...	136 KB
 BookBiz Management System.exe.config	7/29/2020 10:36 PM	XML Configuration...	1 KB



## Implementation

Source Code : <https://github.com/HoruScodes/bookbiz-management-system.git>

Almost all the methods are the same.

I have implemented Validations and Operations.

### 1)Validations

#### *a) Email Validation*

I have used Regex to validate emails.

```
public static bool isValidEmail(TextBox text)
{
    //using regex for email
    Regex regex = new Regex(@"^([\w\.-]+)@([\w\.-]+)((\.(\w){2,3})+)$");
    Match match = regex.Match(text.Text);
    if (match.Success)
    {
        return true;
    }
    return false;
}
```

#### *b) Number Validation and Phone Number/Fax Number validations*

```
public static bool IsValidContactNumber(TextBox text)
{
    if (text.TextLength == 10 && isValidNumber(text)){
        return true;
    }
    return false;
}

public static bool isValidNumber(TextBox text)
{
    int id;
    if(Int32.TryParse(text.Text, out id)){
        return true;
    }
    return false;
}
```

Code reuse in function isValidContactNumber

### *c)Name Validations*

```
public static bool IsValidName(TextBox text)
{
    //if the text length is 0 then return false and show error message
    if (text.TextLength == 0)
    {
        MessageBox.Show("this field Can Not be empty.", "INVALID NAME");
        text.Clear();
        text.Focus();
        return false;
    }
    else
    {
        //run a loop through text to check if it has number then return false
        //With error message
        for (int i = 0; i < text.TextLength; i++)
        {
            if (char.IsDigit(text.Text, i) || char.IsWhiteSpace(text.Text, i))
            {
                MessageBox.Show("it seems you have number or space in your
                field please enter another name.", "INVALID NAME");

                text.Clear();
                text.Focus();
                return false;
            }
        }
    }
    return true;
}
```

## 2)Operations

### *a)List Contents*

```
public static void ListContents(ListView listViewItems)
{
    //read the file
    StreamReader sReader = new StreamReader(authdb);
    listViewItems.Items.Clear();
    //read lines
    string line = sReader.ReadLine();
    // read until the end of file(until line becomes null)
    while (line != null)
    {
        //split lines and set each field to appropriate item
        string[] fields = line.Split(',');
        ListViewItem item = new ListViewItem(fields[0]);
        item.SubItems.Add(fields[1]);
        item.SubItems.Add(fields[5]);
        item.SubItems.Add(fields[4]);
        listViewItems.Items.Add(item);
        // read new line
        line = sReader.ReadLine();
    }
    sReader.Close();
}
```

I have written almost similar functions to list contents so I won't be posting all the code Steps are.

=>Read the file

=>Read the lines and split lines by commas and store into strings[] array

=>Set the items using item.SubItems.Add()

=>Run the loop until End of file.

### *b)Add to Database*

by far this was little complex for me as I was adding to 3 databases (Author , Books , Inventory) at once in inventory form.

Also I had to generate authorId , publisherId , it was little complex for me but I think I Came with good solution..

=>read the database file

=>get the last line of file if its empty then set id to 0 else get the last lines id and return  
new id = id + 1;

```
private static string getLatestAuthorId()
{
    // read the file
    StreamReader f = new StreamReader(authordb);
    string lastline = "";
    string temp;
    //get last line of the file
    while ((temp = f.ReadLine()) != null)
    {
        lastline = temp;
    }
    // if last line is null/empty then set id = 0 else get lastlines field that
    // Includes id and return id = id + 1
    if (String.IsNullOrEmpty(lastline))
    {
        f.Close();
        return "1";
    }
    else
    {
        f.Close();
        string[] fields = lastline.Split(',');
        int id = Convert.ToInt32(fields[0]);
        string latestId = (id + 1).ToString();
        return latestId;
    }
}
```

And To add Data to Database..

```

public static void addItemToInventory(Inventory item ,Author auth , Book bk)
{

    //generate latest ids
    string generateAuthId = getLatestAuthorId();
    string generatePubId = getLatestPublisherId();

    //read all the files
    StreamWriter bkWriter = new StreamWriter(bookdb, true);
    StreamWriter itemWriter = new StreamWriter(inventorydb, true);
    StreamWriter authWriter = new StreamWriter(authordb, true);

    //write data using streamwriter object
    bkWriter.WriteLine(bk.ISBN + "," + bk.Title + "," + bk.Category + "," +
    bk.UnitPrice + "," + bk.PublicationYear + "," + generateAuthId);

    itemWriter.WriteLine(bk.ISBN + "," + generatePubId + "," + item.PublisherName
    + "," + item.QOH);

    authWriter.WriteLine(generateAuthId + "," + auth.AuthorFname + "," +
    auth.AuthorLname + "," + auth.AuthorEmail);

    //close writer objects
    bkWriter.Close();
    itemWriter.Close();
    authWriter.Close();
    MessageBox.Show("Book Data has been saved.");
}

```

*c)search in the database*

=>Read the file

=> Read the file

=>Read the lines and split lines by commas and store into strings [] array

=>Search if item == fields []

=>if yes then add that line to item.SubItems.Add() and then show it in the listviewbox

In form.



```

public static Boolean searchOrder(String book, String client , ListView
listViewItems)
{
    // create stream reader object
    StreamReader sReader = new StreamReader(orderdb);
    string line = sReader.ReadLine();
    // run through end of file
    while (line != null)
    {
        string[] fields = line.Split(',');
        // if match found then add it to items else check next line
        if (book == fields[1] && client == fields[2])
        {
            listViewItems.Items.Clear();
            ListViewItem item = new ListViewItem(fields[5]);
            item.SubItems.Add(fields[2]);
            item.SubItems.Add(fields[1]);
            item.SubItems.Add(fields[3]);
            listViewItems.Items.Add(item);
            sReader.Close();
            return true;
        }
        else
        {
            line = sReader.ReadLine();
        }
    }
    sReader.Close();
    return false;
}

```

Here I have searched for specific order using bookname and client name.

*d)removing a record.*

```
public static Boolean CancelOrder(String book, String client)
{
    // create a reader object to read from orderdb
    StreamReader sReader = new StreamReader(orderdb);
    string line = sReader.ReadLine();
    // create a writer object to write into new temp file
    StreamWriter sWriter = new StreamWriter(temp, true);
    // run loop until end of line
    while (line != null)
    {
        string[] fields = line.Split(',');
        // add all fields exluding book and fields
        if (book != fields[1] && client != fields[2])
        {
            sWriter.WriteLine(fields[0] + "," + fields[1] + "," + fields[2] + ","
                + fields[3] + "," + fields[4] + "," + fields[5]);
        }
        // read next line
        line = sReader.ReadLine();
    }
    sReader.Close();
    sWriter.Close();
    // delete orderdb
    File.Delete(orderdb);
    // set temp as new orderdb
    File.Move(temp, orderdb);
    return true;
}
```

Creating a new file and saving the all the data exluding specific line and replace/set

That file as new database will do the work. That is what I have done.

## Conclusion

the most important is that it is not about technology or creating a good project plan

Ability to see the big picture of the project and focus on particular details when needed deserves a second place in my list. For each project this "details" level would be different, but the key is the ability to shift your focus.

Proactivity would go to the third place. A good Developer should not wait for something to happen - he/she needs to anticipate things and constantly look for ways to make the project better. It may be a friendly reminder of something you agreed with your client verification of assumptions or risk mitigation steps, but it saves a lot of trouble and delays on most projects.

On a technical side , This project allowed me to use and test the many software development methods learned during the classes in ADOBE CONNECT. The Project had consisted of a set of tasks that requires validations and system requirements from users. using classes, we can see how could be easy to implement and also reuse. Last 3 lectures helped me so much.

Thank you alex for teaching me , it is my honor to be your student.