

# *Library Management System*



**Avigyan Dasgupta**

**Standard: X; Division: A**

**Roll No. 12**

**Vidya Valley School**

## Contents

|                                |     |
|--------------------------------|-----|
| Acknowledgement .....          | 4   |
| A Brief .....                  | 5   |
| Prototype – Basic Design ..... | 6   |
| Algorithm .....                | 11  |
| Flowchart .....                | 12  |
| Variable Descriptions .....    | 19  |
| CONNECTION WITH DATABASE ..... | 19  |
| LOGIN PAGE .....               | 19  |
| LIBRARIAN CHOICES .....        | 19  |
| LIBRARIAN MASTER UPDATE .....  | 19  |
| READER MASTER .....            | 19  |
| NEW READER .....               | 20  |
| MODIFY READER .....            | 20  |
| DELETE READER .....            | 21  |
| BOOK MASTER .....              | 22  |
| NEW BOOK .....                 | 22  |
| MODIFY BOOK .....              | 23  |
| DELETE BOOK .....              | 24  |
| LIBRARIAN MAINTAINANCE .....   | 24  |
| TRANSACTION MENU .....         | 25  |
| BOOK ISSUE .....               | 25  |
| BOOK RETURN .....              | 26  |
| REPORT MENU .....              | 27  |
| BOOK SEARCH .....              | 27  |
| READER SEARCH .....            | 28  |
| DUE REPORT .....               | 28  |
| LENDING REPORT .....           | 28  |
| ADMIN MENU .....               | 29  |
| NEW USER .....                 | 29  |
| MODIFY USER .....              | 29  |
| REMOVE USER .....              | 30  |
| CHANGE PASSWORD .....          | 31  |
| Code .....                     | 32  |
| Output .....                   | 141 |

|   |     |
|---|-----|
| Interfacing Screens (Sample) .....          | 141 |
| Reports .....                               | 147 |
| Installation Procedure .....                | 150 |
| Operation Manual / User Guide .....         | 153 |
| How to Start .....                          | 153 |
| How to Create User .....                    | 153 |
| How to Update Book Master .....             | 153 |
| How to Update Reader Master .....           | 153 |
| How to Capture Book Issue .....             | 153 |
| How to Capture Book Return .....            | 153 |
| How to Capture “Book Lost” .....            | 153 |
| How to Change Self Password .....           | 154 |
| How to Change The Password For Others ..... | 154 |
| Conclusion .....                            | 155 |

## Acknowledgement

I am using this opportunity to express my gratitude and deepest appreciation to everyone who supported me throughout the Project. I am thankful for their aspiring guidance, invaluable constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project and especially about JAVA.

I would like to express my special thanks of gratitude to our teacher Respected Nilima Madam who gave me the golden opportunity to do this wonderful project on the topic (Library Management System), which also helped me in doing a lot of research and I came to know about so many new things.

Secondly I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

## A Brief

The basic essence of this application is to automate the different processes in Book Library. This application has the feature to register more than one Librarian in the system. Two databases are being maintained to update the detail about Books and the readers. Books can be issues to Readers only. Before starting any transaction, User (of type Librarian) has to login, and for audit trail login id is being stored for each transaction. System has the facility to capture the necessary detail in case the book is damage.

Admin user has the privilege to create, modify users, readers and Book detail. System has the facility to have “General” users also, that type of user can generate different reports only.

## Prototype – Basic Design

### Library Management System

Login

Password

### Admin Menu

#### **Maintenance Menu**

New User  
Modify User  
Remove User  
Change Password

Sign Out

### Librarian Menu

#### **Master Update**

Reader Menu  
Book Menu

#### **Maintenance**

Change Password for Self

#### **Transaction**

Book Issue  
Book Return

#### **Reports**

Book Search  
Lending Report  
Due Report

Sign Out

### Report Menu

#### **Reports**

Book Search  
Lending Report  
Due Report

Change Password (self)

Exit

## Admin Menu

### Maintenance Menu

New User

Modify User

Remove User

Change Password

#### New User

Login

Password

Name

Category   
Librarian  
General User

Add

Back

Sign Out

#### Modify User

Login

Password

Name

Category   
Librarian  
General User

Add

Back

Sign Out

#### Remove User

Login

Name

Remark

Remove

Back

Sign Out

#### Change Password

Login

Name

Category

Old Password

New Password

Confirm Password

Change

Back

Sign Out

## Librarian Menu

### Master Update

Reader Master

Book Master

**New Reader**  
**Modify Reader**  
**Delete Reader**

**New Book**  
**Modify Book**  
**Delete Book**

#### New Reader

Name   
Standard  Division   
Roll No.   
Unique Id

Add

Back

Sign Out

#### New Book

Name   
Author   
Price  Currency   
Unique Id  Genre

Add

Back

Close

#### Modify Reader Master

Unique Id   
Name   
Standard  Division   
Roll No.

Modify

Back

Sign Out

#### Modify Book Information

Unique Id   
Name   
Author   
Price  Currency   
Genre

Modify

Back

Sign Out

#### Delete Reader

Unique Id  Remark   
Name   
Standard  Standard   
Roll No.

Delete

Back

Sign Out

#### Delete Book Information

Unique Id  Remark   
Name   
Author   
Price  Currency   
Genre

Delete

Back

Sign Out



### Change Password

|                  |                          |
|------------------|--------------------------|
| Login            | <input type="text"/>     |
| Name             | <input type="text"/>     |
| Category         | <input type="text"/>     |
| Old Password     | <input type="password"/> |
| New Password     | <input type="password"/> |
| Confirm Password | <input type="password"/> |

### Librarian Menu

## Maintenance

Change Password for Self

### Librarian Menu

## Transaction

Book Issue  
Book Return

### Book Issue

|            |   |   |
|------------|---|---|
| Reader Id  | <input type="text"/>                    | Avigyan Dasgupta                                    |
| Book Id    | <input type="text"/>                    | A Tale Of Two Cities                                |
| Issue Date | <input type="text" value="DD/MM/YYYY"/> | Return Date <input type="text" value="DD/MM/YYYY"/> |

### Book Return

|            |   |   |
|------------|---|---|
| Reader Id  | <input type="text"/>                    | Avigyan Dasgupta                                    |
| Book Id    | <input type="text"/>                    | A Tale Of Two Cities                                |
| Issue Date | <input type="text" value="DD/MM/YYYY"/> | Return Date <input type="text" value="DD/MM/YYYY"/> |

## Librarian Menu

### Reports

Book Search  
Lending Report  
Due Report

| Book No. | Genre     | Name       | Author | Price | Currency |
|----------|-----------|------------|--------|-------|----------|
| F001     | Fiction   | AAAAAA     | XXXX   | 200   | INR      |
| F002     | Fiction   | BBBBBBB    | YYYY   | 300   | INR      |
| F003     | Fiction   | CCCCCCC    | ZZZZ   | 50    | INR      |
| A001     | Adventure | DDDDDDDDDD | JJJJ   | 120   | INR      |
| A002     | Adventure | EEEEEE     | PPPP   | 200   | INR      |
| A003     | Adventure | FFFFFF     | LLL    | 230   | INR      |

| Book No. | Book Name  | Reader   | Lending Date | Return Date |
|----------|------------|----------|--------------|-------------|
| F001     | AAAAAA     | Rohit    | 1-Sep-14     | 15-Sep-14   |
| F002     | BBBBBBB    | Mohit    | 12-Sep-14    | 26-Sep-14   |
| F003     | CCCCCCC    | Raj      | 8-Sep-14     | 22-Sep-14   |
| A001     | DDDDDDDDDD | Rishi    | 3-Sep-14     | 17-Sep-14   |
| A002     | EEEEEE     | Jaya     | 14-Sep-14    | 28-Sep-14   |
| A003     | FFFFFF     | Narendra | 10-Sep-14    | 24-Sep-14   |

## Algorithm

**Step 1:** Start

**Step 2:** Validation of Login and Password

**Step 3:** If user type is “Librarian” then go to Step 4, if the type is “Admin” then step 6 and if the type is “General User” then go to Step 8

**Step 4:** “Librarian” type User can “Add – Modify – Delete” Reader and Book information. Here one can change the password for the “Self”. As “Librarian” type all the library transaction like “Book Issue”, “Book Return” can be captured. “Librarian” will have facility to generate different reports also.

**Step 5:** Go to Step 10

**Step 6:** As “Admin” type one can Add New User, Modify existing User, Remove User, Change the Password for “Self” and for any other user.

**Step 7:** Go to step 10

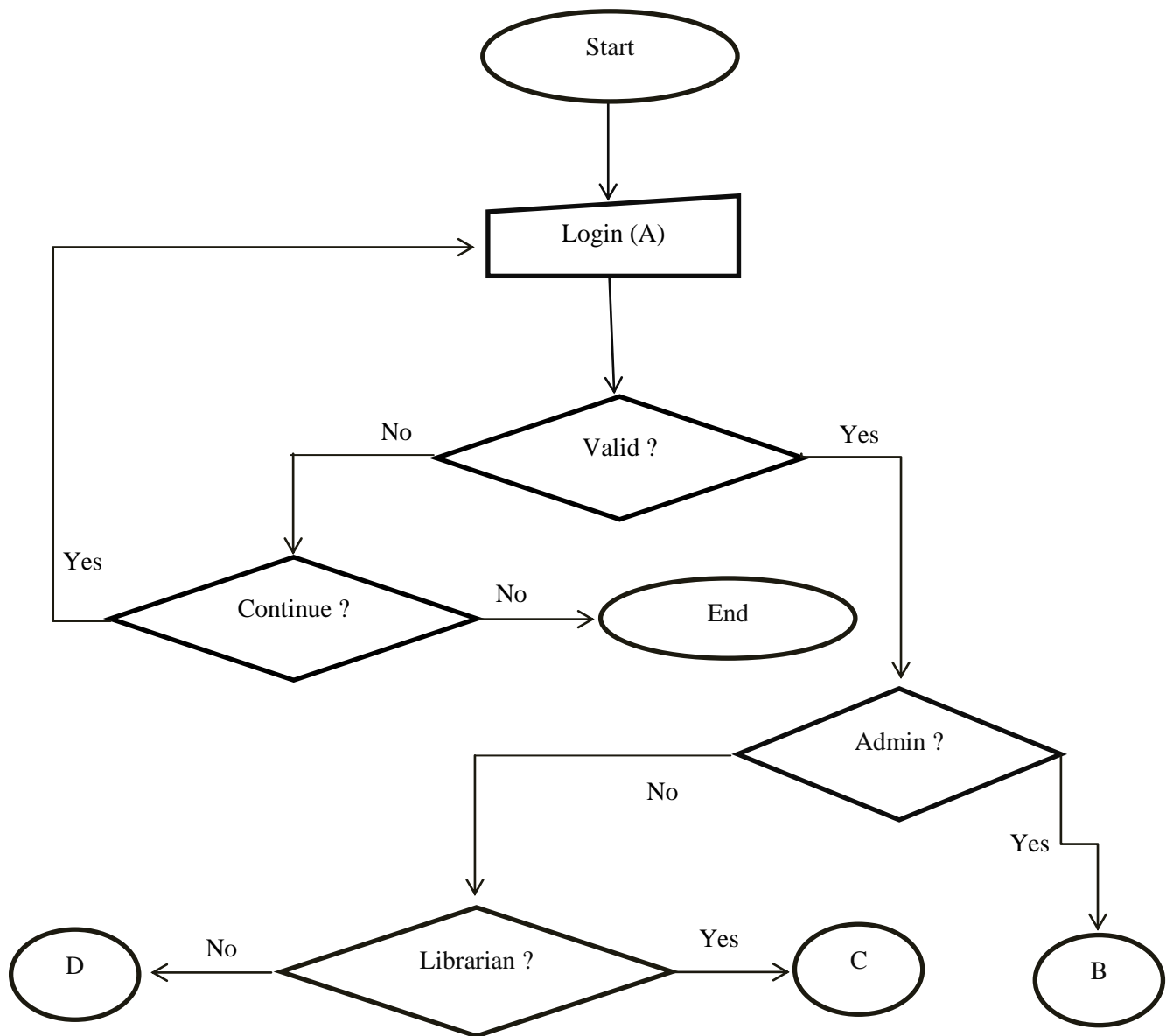
**Step 8:** As “General User” type different reports like Book Search, Lending Report, Due Report etc.

**Step 9:** Go to step 10

**Step 10:** Log off

**Step 11:** end

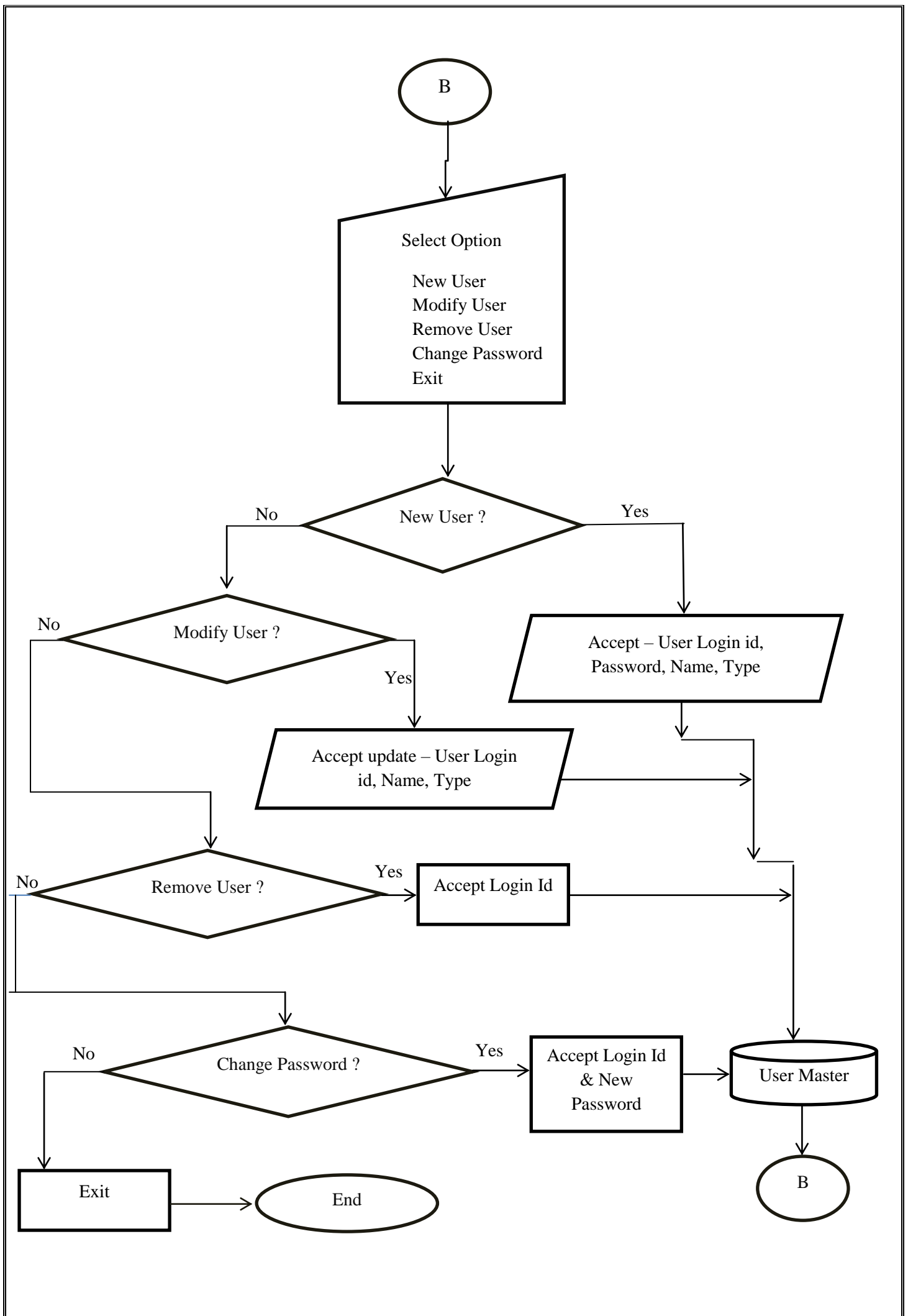
## Flowchart

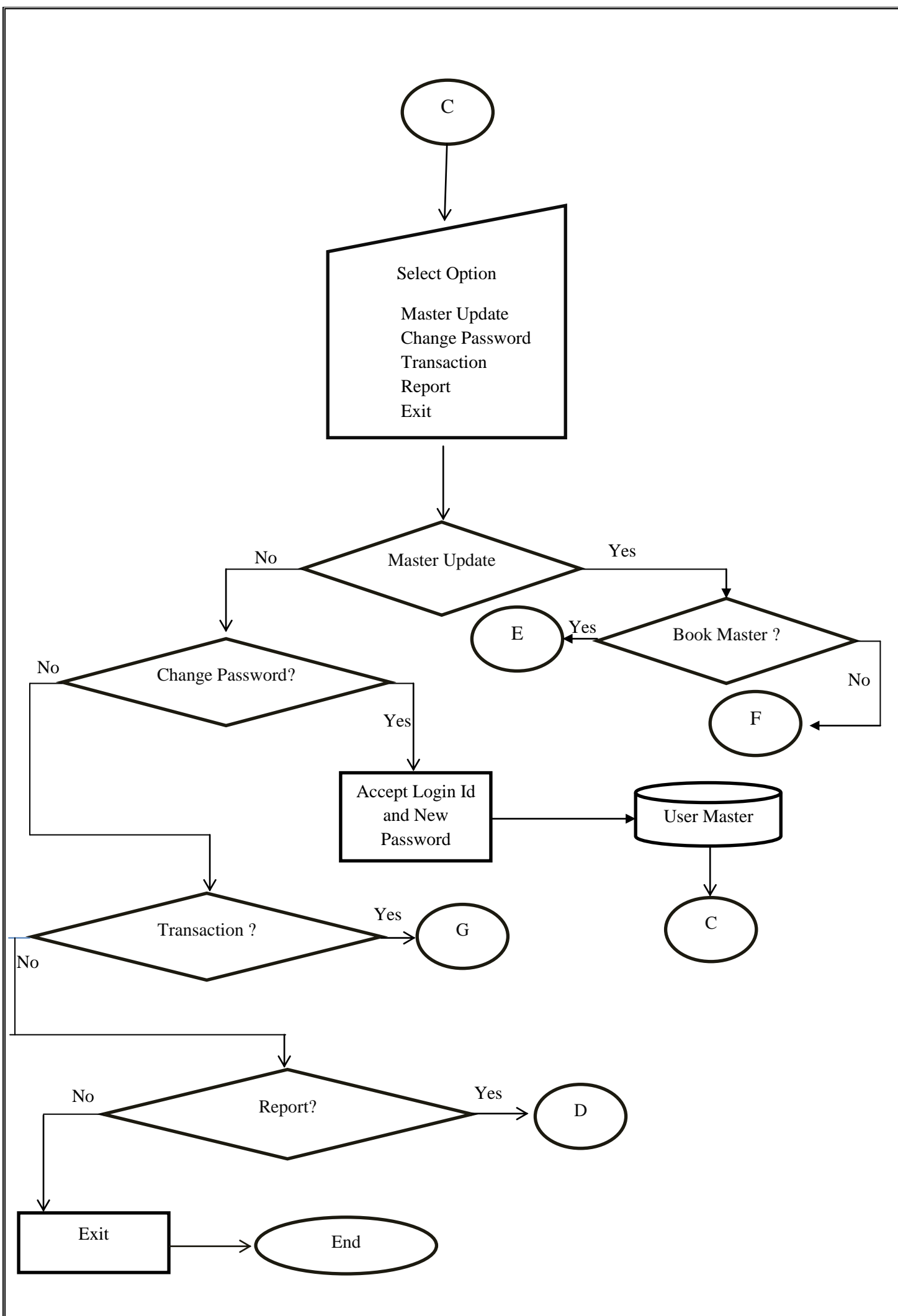


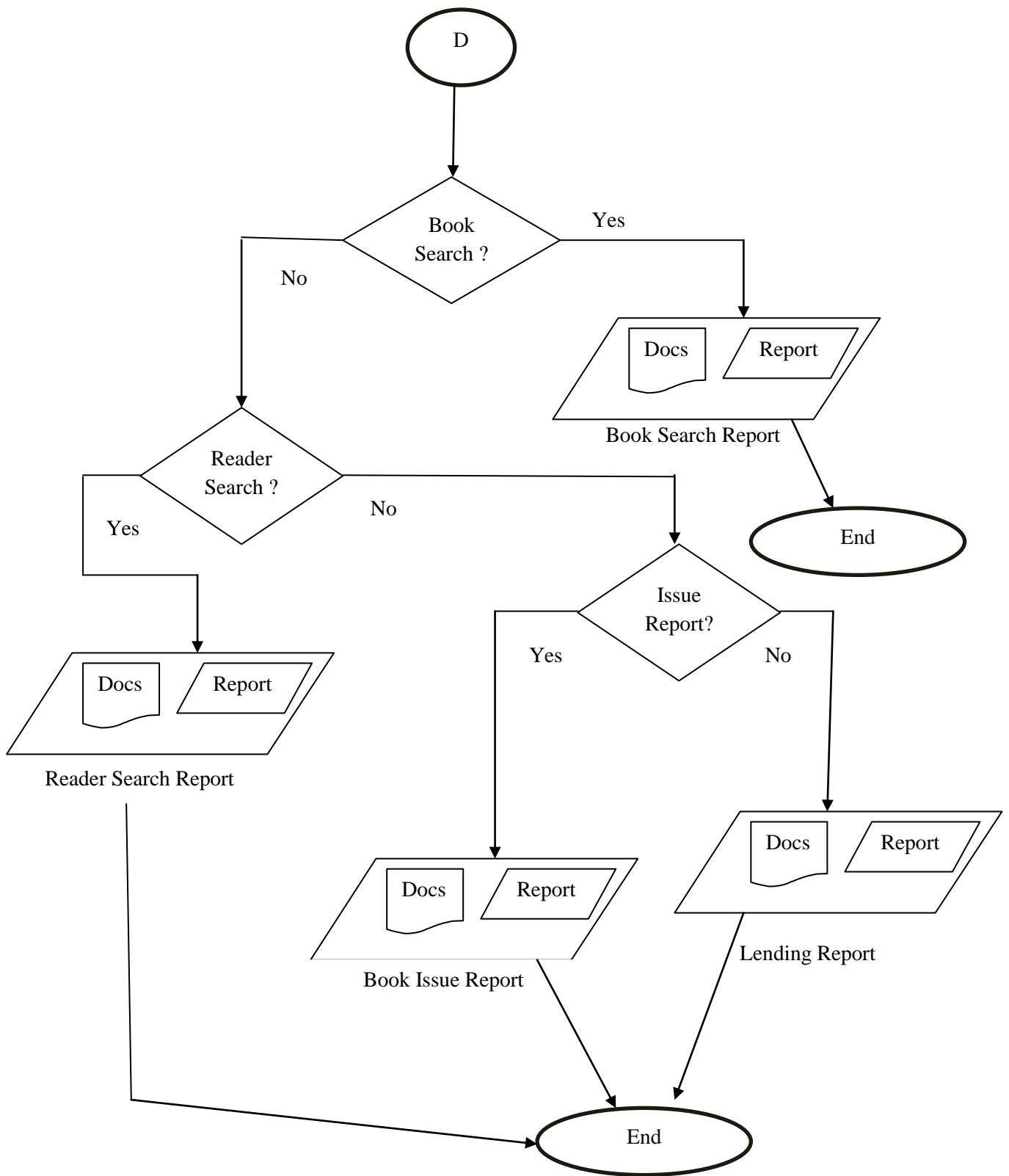
Admin Menu → B

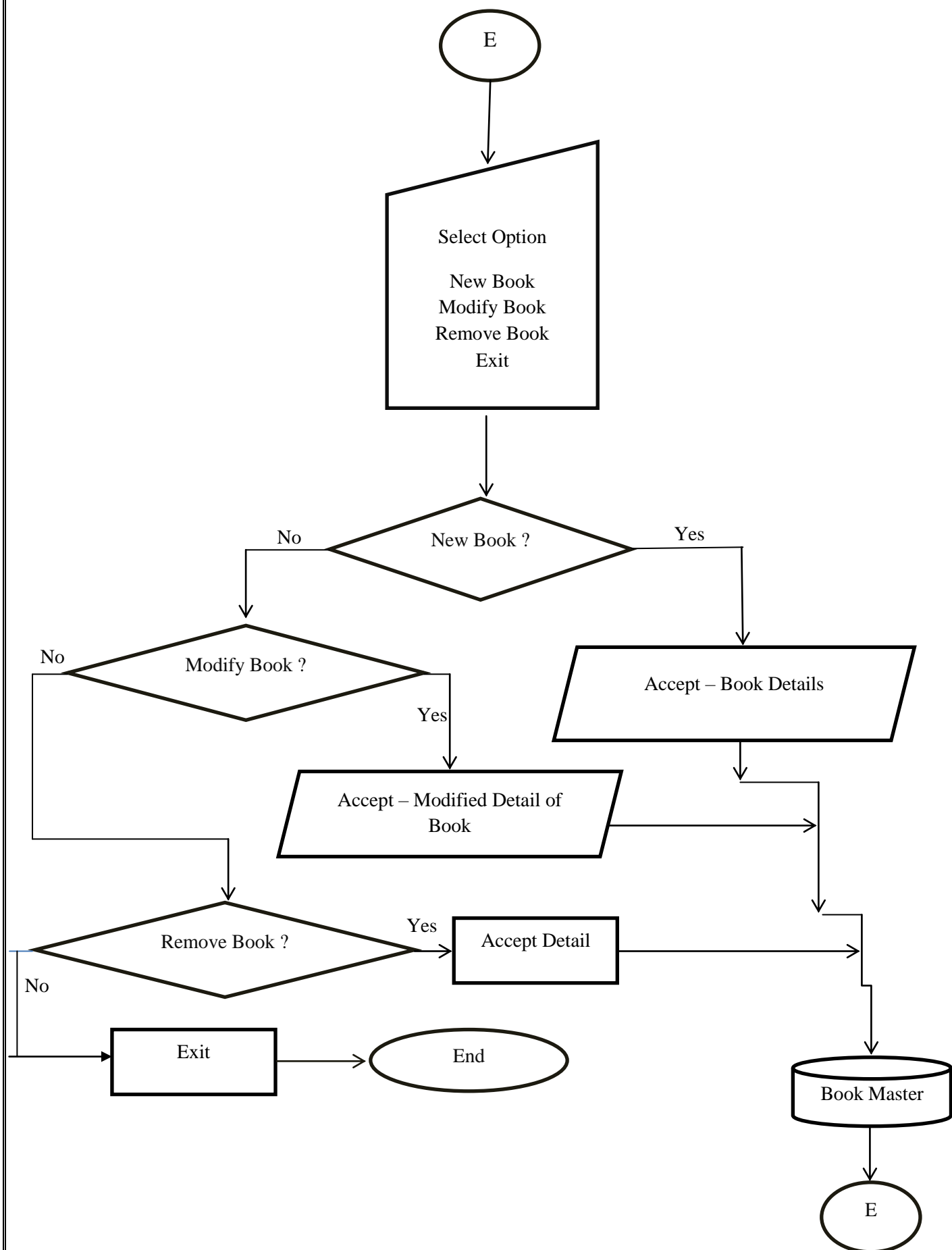
Librarian Menu → C

Report Menu (General User) → D

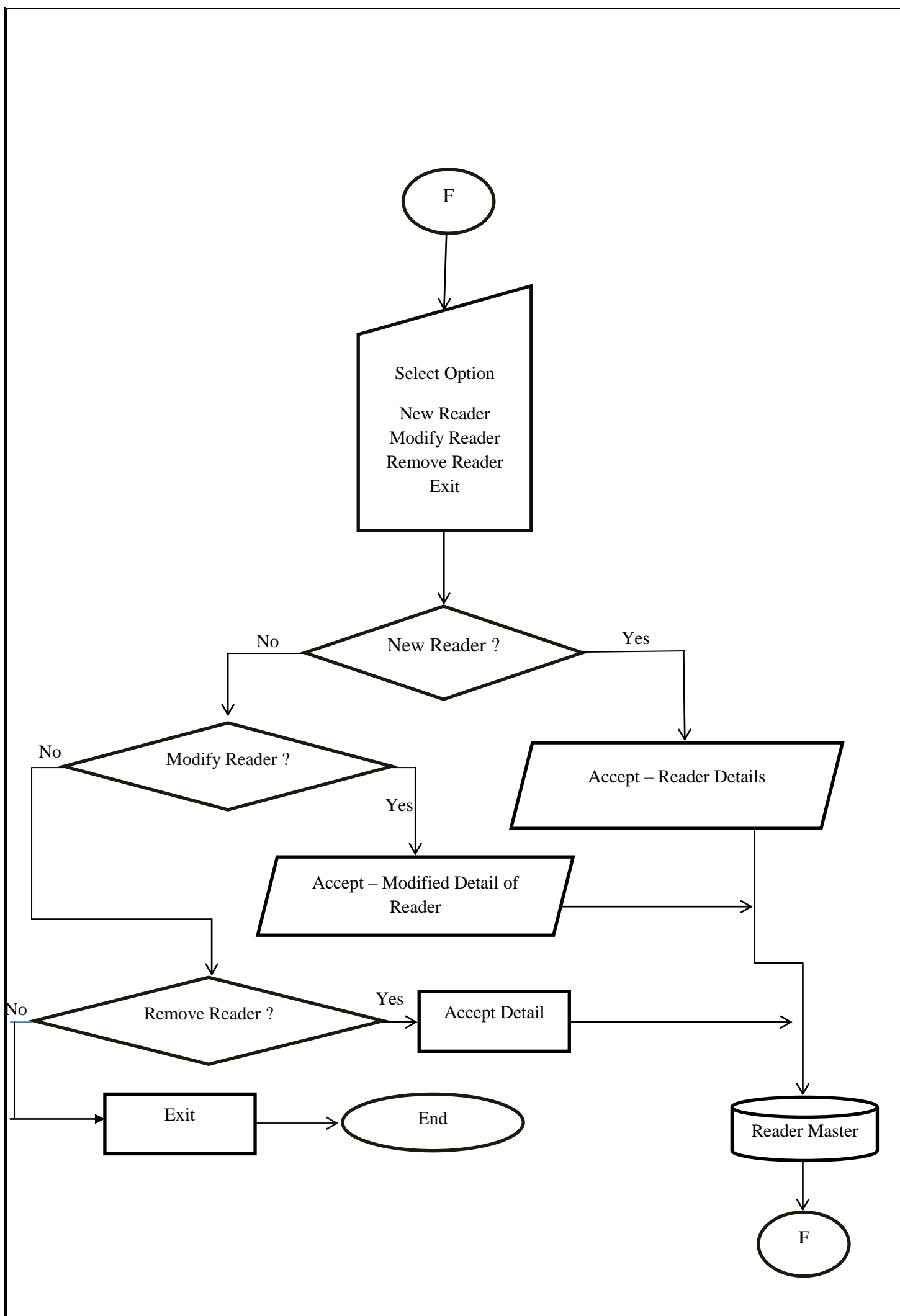


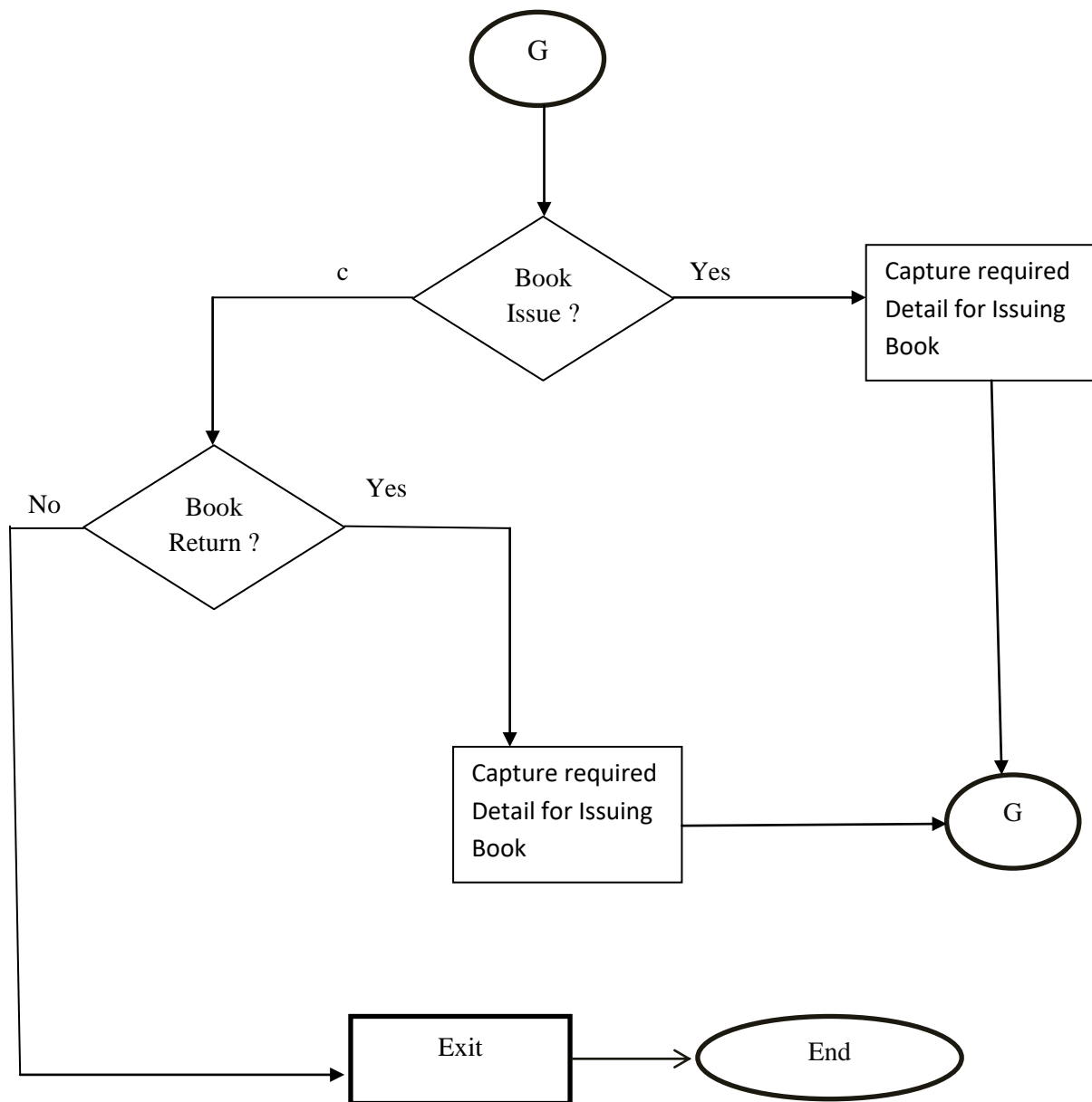












## Variable Descriptions

### CONNECTION WITH DATABASE

1. Connection con
2. Statement st
3. String driver and String db : used for the database connection

### LOGIN PAGE

1. JFrame loginframe : Frame for user login
2. JLabel l : Label saying "LOGIN ID"
3. JLabel l1 : Label saying "PASSWORD"
4. JTextField t : Text field for Login id
5. JPasswordField t1 : Password field for Password
6. JButton b : Button with the label "LOGIN"
7. JPanel p : Panel for user login in which the contents are added
8. GridBagConstraints gbcLoginPage : Used for the page layout
9. String sql : Stores the sql which is runned
10. ResultSet rs : It runs the sql
11. String user : Used to store the input of login id
12. String pass : Used to store the input of password
13. String USER\_TYPE[] : Used to store the three user types "Admin", "General" and "Librarian"
14. String type : It stores the user category against that user as it is from the database
15. int temp : It stores the position of the string in the USER\_TYPE which matches String type
16. int trial : It is used for running the loop till the String type matched a string in USER\_TYPE
17. int selectedOption : Stores the value of the entry in the ConfirmDialog(in case of invalid id)

### LIBRARIAN CHOICES

1. JFrame LibMenuCHOICES : Frame for choices for Librarian
2. JPanel panel1 : Panel for choices in which the contents are added
3. GridBagConstraints gbcLibMenuChoices : Used for the page layout
4. JButton b1 : Button with the label "MASTER UPDATE"
5. JButton b2 : Button with the label "MAINTAINANCE MENU"
6. JButton b3 : Button with the label "TRANSACTION MENU"
7. JButton b4 : Button with the label "REPORT MENU"
8. JButton b5 : Button with the label "SIGN OUT"

### LIBRARIAN MASTER UPDATE

1. JFrame LibMenuMASTERUPDATE : Frame for choices for Master Update
2. JPanel panelmu : Panel for choices in which the contents are added
3. JButton mu1 : Button with the label "READER MASTER"
4. JButton mu2 : Button with the label "BOOK MASTER"
5. JButton mu3 : Button with the label "BACK"
6. JButton mu4 : Button with the label "SIGN OUT"
7. GridBagConstraints gbcLibMenuMasterUpdate : Used for the page layout

### READER MASTER

1. JFrame LibMenuRM : Frame for choices for Reader Master
2. JPanel panelrm : Panel for choices in which the contents are added

3. JButton rm1 : Button with the label “NEW READER”
4. JButton rm2 : Button with the label “MODIFY READER”
5. JButton rm3 : Button with the label “DELETE READER”
6. JButton rm4 : Button with the label “BACK”
7. JButton rm5 : Button with the label “SIGN OUT”
8. GridBagConstraints gbcLibMenu : Used for the page layout

## NEW READER

1. JFrame murmnr : Frame for New Reader
2. String STANDARD[] : String to be included in the combo box
3. String ROLLNO[] : String to be included in the combo box
4. JLabel murmnrfn : Label for “First Name”
5. JLabel murmnrln : Label for “Last Name”
6. JLabel murmnrs : Label for “Standard”
7. JLabel murmnrd : Label for “Division”
8. JLabel murmnrrn : Label for “Roll no”
9. JLabel murmnrid : Label for “Id”
10. JTextField murmnrfn1 : Text field to take input “First Name”
11. JTextField murmnrln1 : Text field to take input “Last Name”
12. JTextField murmnrd1 : Text field to take input “Division”
13. JTextField murmnrid1 : Text field to take input “Id”
14. JComboBox murmnrs1 : Combo box for Standard
15. JComboBox murmnrrn1 : Combo box for roll no
16. JPanel murmnrp : Panel in which the contents are added
17. JButton murmnradd : Button with the label “Add”
18. JButton murmnrback : Button with the label “Back”
19. JButton murmnrso : Button with the label “Sign Out”
20. GridBagConstraints murmnrgbc : Used for the page layout
21. String sqlmurmnr : Stores the sql which is runned
22. ResultSet rsmurmnr : Used to run the sql
23. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
24. String fnamemurmnr : Stores the data entry in JTextField murmnrfn1 (ie; first name)
25. String lnamemurmnr : Stores the data entry in JTextField murmnrln1 (ie; last name)
26. String standardmurmnr : Stores the data selection in JComboBox murmnrs1 (ie; standard)
27. String divisionmurmnr : Stores the data entry in JTextField murmnrd1 (ie; division)
28. String strollnomurmnr : Stores the data selection in JComboBox murmnrrn1 (ie; roll no)
29. int rollnomurmnr : Stores String strollnomurmnr (ie; roll no) in the integer format
30. String IDmurmnr : Stores the data entry in JTextField murmnrid1 (ie; id)
31. String standardtempmurmnr : Stores the standard from each line in the database – one at a time
32. String divisiontempmurmnr : Stores the division for the corresponding standard
33. String strollnotempmurmnr : Stores the roll no for the corresponding standard and division
34. int rollnotempmurmnr : Stores the roll no in the integer format
35. String idtempmurmnr : Stores the id from each line in the database – one at a time

## MODIFY READER

1. JLabel murmmrentryidlabel : Label for “Enter the Reader id”
2. JTextField murmmrentryidtext : Text field to take input “Reader id”
3. JPanel murmmrentryidpanel : Panel in which the contents are added
4. int selectedoptionIDModifyReader : Stores the value of the entry in the ConfirmDialog

5. String ID : Stores the data entry in JTextField murmmrentryidtext (ie; Reader Id)
6. JFrame murmmr : Frame for Modify Reader
7. JPanel murmmrp : Panel in which the contents are added
8. String sqlLibMenumurmmr : Stores the sql which is to be runned
9. ResultSet rsLibMenumurmmr : ResultSet which runs the sql sqlLibMenumurmmr
10. JLabel murmmrfn : Label for "First Name"
11. JLabel murmmrln : Label for "Last Name"
12. JLabel murmmrs : Label for "Standard"
13. JLabel murmmrd : Label for "Division"
14. JLabel murmmrrn : Label for "Roll no"
15. JLabel murmmrid : Label for "Id"
16. JTextField LibMenuMURMMRfn1 : Text field to take input "First Name"
17. JTextField LibMenuMURMMRln1 : Text field to take input "Last Name"
18. JComboBox LibMenuMURMMRs1 : Combo box for Standard
19. JTextField LibMenuMURMMRd1 : Text field to take input "Division"
20. JComboBox LibMenuMURMMRrn1 : Combo box for Roll no
21. JTextField LibMenuMURMMRid1 : Text field to take input "Reader id"
22. int temporaryLibMenuMURMMR : Restricts entry of line of execution if some earlier conditions have not been satisfied
23. JButton murmmrmodify : Button with the label "Modify"
24. JButton murmmrback : Button with the label "Back"
25. JButton murmmrso : Button with the label "Sign Out"
26. GridBagConstraints gbcmurmmr : Used for the page layout
27. String murmmrFN : Stores the data entry in JTextField LibMenuMURMMRfn1(ie; first name)
28. String murmmrLN : Stores the data entry in JTextField LibMenuMURMMRln1(ie; last name)
29. String murmmrS : Stores the data selection in JTextField LibMenuMURMMRs1(ie; standard)
30. String murmmrD : Stores the data entry in JTextField LibMenuMURMMRd1(ie; division)
31. String strmurmmrRN : Stores the data selection in JTextField LibMenuMURMMRrn1(ie; roll no)
32. int murmmrRN Stores String strmurmmrRN (ie; roll no) in the integer format
33. String murmmrID : Stores the data entry in JTextField LibMenuMURMMRid1(ie; id)

## DELETE READER

1. JTextField murmdrentryidtext : Text field to take input "Reader id"
2. JLabel murmdrentryidlabel : Label for "Enter the Reader id"
3. JPanel murmdrentryidpanel : Panel in which the contents are added
4. selectedoptionIDDeleteReader : Stores the value of the entry in the ConfirmDialog
5. JFrame murmdr : Frame for Delete Reader
6. String ID : Stores the data entry in JTextField murmdrentryidtext (ie; Reader Id)
7. String sqlLibMenumurmdr : Stores the sql to be runned
8. ResultSet rsLibMenumurmdr : ResultSet that runs the sql
9. int temporaryLibMenuMURMDR : Restricts entry of line of execution if some earlier conditions have not been satisfied
10. String strLibMenuMURMDRidcount1 : Stores the value of the number of books with the reader
11. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
12. JPanel murmdrp : Panel in which the contents are added
13. GridBagConstraints gbcmurmdr : Used for the page layout
14. JLabel murmdrfn : Label saying "First Name"
15. JLabel murmdrln : Label saying "Last Name"
16. JLabel murmdrs : Label saying "Standard"

17. JLabel murmdrd : Label saying “Division”
18. JLabel murmdrrn : Label saying “Roll No”
19. JLabel murmdrid : Label saying “Id”
20. JLabel drremarklabel : Label saying “Remark”
21. JLabel LibMenuMURMDRfn1 : Label to display first name of the reader with id equal to ID
22. JLabel LibMenuMURMDRln1 : Label to display last name of the reader with id equal to ID
23. JLabel LibMenuMURMDRs1 : Label to display standard of the reader with id equal to ID
24. JLabel LibMenuMURMDRd1 : Label to display division of the reader with id equal to ID
25. JLabel LibMenuMURMDRrn1 : Label to display roll no of the reader with id equal to ID
26. JLabel LibMenuMURMDRid1 : Label to display id of the reader with the id equal to ID
27. JTextField drremark : TextField to obtain the remark as to why the reader is being deleted
28. JButton murmdrdelete : Button with the label “Delete”
29. JButton murmdrback : Button with the label “Back”
30. JButton murmdrso : Button with the label “Sign Out”
31. String stremarkdr : Stores the text entered in JTextField drremark (ie; Remark)
32. int temporarydr : Restricts entry of line of execution if some earlier conditions have not been satisfied

## BOOK MASTER

1. JFrame LibMenubm : Frame for choices for Book Master
2. JPanel panelbm : Panel for choices in which the contents are added
3. JButton bm1 : Button with the label “New Book”
4. JButton bm2 : Button with the label “Modify Book”
5. JButton bm3 : Button with the label “Delete Book”
6. JButton bm4 : Button with the label “Back”
7. JButton bm5 : Button with the label “Sign out”
8. GridBagConstraints gbcLibMenubm : Used for the page layout

## NEW BOOK

1. JFrame mubmnb : Frame for New Book
2. JPanel mubmnbp : Panel in which the contents are added
3. JLabel mubmnbbn : Label saying “Book Name”
4. JLabel mubmnbau : Label saying “Author”
5. JLabel mubmnbpr : Label saying “Price”
6. JLabel mubmnbcu : Label saying “Currency”
7. JLabel mubmnbg : Label saying “Genre”
8. JLabel mubmnbid : Label saying “ID”
9. String CURRENCY[] : String array storing the currencies
10. String GENRE[] : String array storing the genres
11. JTextField mubmnbbn1 : Text field to take input “Book Name”
12. JTextField mubmnbau1 : Text field to take input “Author”
13. JTextField mubmnbpr1 : Text field to take input “Price”
14. JComboBox mubmnbcu1 : Combo Box to take input “Currency”
15. JComboBox mubmnbg1 : Combo Box to take input “Genre”
16. JTextField mubmnbid1 : Text field to take input “ID”
17. JButton mubmnbad : Button with the label “Add”
18. JButton mubmnback : Button with the label “Back”
19. JButton mubmnbs : Button with the label “Sign out”
20. GridBagConstraints mubmnbgc : Used for the page layout

21. String idmubmb : String storing the input in mubmbid1(ie; Id)
22. String bnamemubmb : String storing the input in mubmbbn1(ie; Book Name)
23. String authormubmb : String storing the input in mubmbau1(ie; Author)
24. String strpricemubmb : String storing the input in mubmbpr1(ie; Price)
25. int pricemubmb : Stores the price in integer format
26. String currencymubmb : String storing the input in mubmbcu1(ie; Currency)
27. String genremubmb : String storing the input in mubmbgn1(ie; Genre)
28. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
29. String sqlmubmb : Stores the sql to be runned
30. ResultSet rsmubmb : ResultSet that runs the sql
31. JFrame genresmubmb : Frame for Combination of genres
32. JPanel genrespanel : Panel in which the contents are added
33. GridBagConstraints gbcgenres : Used for the page layout
34. JLabel genrestriallabel : Label saying "GENRES"
35. JTextField genrestrial : Text field to take input "Genre"
36. JButton ok : Button with the label "OK"
37. JButton back : Button with the label "BACK"
38. JButton so : Button with the label "SIGN OUT"
39. String idtempmubmb : Stores all the values of Book ids in the database (one at a time)

## MODIFY BOOK

1. JTextField mubmbentryidtext : Text Field for taking the id of the book to be modified
2. JLabel mubmbentryidlabel : Label saying "Please enter the Book id"
3. JPanel mubmbentryidpanel : Panel in which the contents are added
4. int selectedoptionIDModifyBook : Stores the value of the entry in the ConfirmDialog
5. JFrame mubmb : Frame for Modify Book
6. String ID : Stores the text entered in JTextField mubmbentryidtext
7. JPanel mubmbp : Panel in which the contents are added
8. JLabel mubmbbn : Label saying "Book Name"
9. JLabel mubmbau : Label saying "Author"
10. JLabel mubmbpr : Label saying "Price"
11. JLabel mubmbcu : Label saying "Currency"
12. JLabel mubmbgn : Label saying "Genre"
13. JLabel mubmbid : Label saying "ID"
14. JTextField mubmbbn1 : Text field to take input "Book Name"
15. JTextField mubmbau1 : Text field to take input "Author"
16. JTextField mubmbpr1 : Text field to take input "Price"
17. JComboBox mubmbcu1 : Combo Box to take input "Currency"
18. JComboBox mubmbgn1 : Combo Box to take input "Genre"
19. JTextField mubmbid1 : Text field to take input "ID"
20. String sqlLibMenuMubmb : Stores the sql to be runned
21. ResultSet rsmubmb : ResultSet that runs the sql
22. JButton mubmbmodify : Button with the label "Modify"
23. JButton mubmbback : Button with the label "Back"
24. JButton mubmbso : Button with the label "Sign out"
25. GridBagConstraints gbcMubmb : Used for the page layout
26. String mubmbID : String storing the input in mubmbid1(ie; Id)
27. String mubmbBN : String storing the input in mubmbbn1(ie; Book Name)
28. String mubmbAU : String storing the input in mubmbau1(ie; Author)
29. String strMubmbPR : String storing the input in mubmbpr1(ie; Price)
30. int mubmbPR : Stores the price in integer format
31. String mubmbCU : String storing the input in mubmbcu1(ie; Currency)
32. String mubmbGN : String storing the input in mubmbgn1(ie; Genre)

33. JFrame genresmubmmb : Frame for Combination of genres
34. JPanel genrespanel : Panel in which the contents are added
35. GridBagConstraints gbcgenres : Used for the page layout
36. JLabel genrestriallabel : Label saying “GENRES”
37. JTextField genrestrial : Text field to take input “Genre”
38. JButton ok : Button with the label “OK”
39. JButton back : Button with the label “BACK”
40. JButton so : Button with the label “SIGN OUT”

## DELETE BOOK

1. JTextField mubmdbentryidtext : Text Field for taking the id of the book to be deleted
2. JLabel mubmdbentryidlabel : Label saying “Please enter the Book id”
3. JPanel mubmdbentryidpanel : Panel in which the contents are added
4. int selectedoptionIDDeleteBook : Stores the value of the entry in the ConfirmDialog
5. String ID : Stores the text entered in JTextField mubmmmbentryidtext
6. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
7. String sqlmubmdb : Stores the sql to be runned
8. ResultSet rsmubmdb : ResultSet for running the sql
9. String strBook\_availabel : Stores the value of B\_AVAILABEL for the book whose id was entered
10. JFrame mubmdb : Frame for Book Delete
11. JPanel mubmdbp : Panel in which the contents are added
12. GridBagConstraints gbcmubmdb : Used for the page layout
13. JLabel mubmdbbn : Label saying “Book Name”
14. JLabel mubmdbau : Label saying “Author”
15. JLabel mubmdbpr : Label saying “Price”
16. JLabel mubmdbcu : Label saying “Currency”
17. JLabel mubmdbgn : Label saying “Genre”
18. JLabel mubmdbid : Label saying “Id”
19. JLabel mubmdbremarklabel : Label saying “Remark”
20. JLabel mubmdbbn1 : Label to display book name of the book with id equal to ID
21. JLabel mubmdbau1 : Label to display author of the book with id equal to ID
22. JLabel mubmdbpr1 : Label to display price of the book with id equal to ID
23. JLabel mubmdbcu1 : Label to display currency of the book with id equal to ID
24. JLabel mubmdbgn1 : Label to display genre of the book with id equal to ID
25. JLabel mubmdbid1 : Label to display id of the book with the id equal to ID
26. JTextField dbremarktext : TextField to obtain the remark as to why the book is being deleted
27. JButton mubmdbdelete : Button with the label “Delete”
28. JButton mubmdbback : Button with the label “Back”
29. JButton mubmdbso : Button with the label “Sign Out”
30. String remarkdeletebook : Stores the text entered in JTextField dbremarktext (ie; Remark)

## LIBRARIAN MAINTAINANCE

1. String sqllmmm : Stores the sql to be runned
2. ResultSet rslmmm : ResultSet which runs the sql
3. LibMenuMM : Frame for Change Password For Self
4. JPanel LibMenuMMp : Panel in which the contents are added
5. JLabel Llmmm : Label saying “Login ID”
6. JLabel UNlmmm : Label saying “User Name”
7. JLabel PWoldlmmm : Label saying “Old Password”
8. JLabel PWnewlmmm : Label saying “New Password”
9. JLabel PWconfirmlmmm : Label saying “Confirm Password”
10. JLabel Clmmm : Label saying “Category”



11. Pwoldlmmm : Password field for Old Password
12. pwnewlmmm : Password field for New Password
13. pwconfrmlmmm : Password field for Confirm Password
14. JButton lmmmchangepassword : Button with the label “Change”
15. JButton lmmmb : Button with the label “Back”
16. JButton lmmso : Button with the label “Sign out”
17. GridBagConstraints gbclmmm : Used for the page layout
18. String strlilmmm : Stores the User Login
19. String strunlmmm : Stores the User Name
20. String strcalmmm : Stores the User Category
21. String strpasslmmm : Stores the User Password
22. JLabel lilmmm : Label to display String strlilmmm
23. JLabel unlmmm : Label to display String strunlmmm
24. JLabel calmmm : Label to display String strcalmmm
25. String oldchangeasslibrarianmenu : Stores the entry in pwoldlmmm(ie; Old Password)
26. String newchangeasslibrarianmenu : Stores the entry in pwnewlmmm(ie; New Password)
27. String confirmchangeasslibrarianmenu : Stores the entry in pwconfrmlmmm(ie; Confirm Password)

## TRANSACTION MENU

1. SimpleDateFormat sdf : Object Reference to get the date in our desired format
2. Calendar dt1 : Stores today’s date
3. Calendar dt2 : Stores seven days later’s date
4. String today : Stores dt1 in String format
5. String duedate : Stores dt2 in String format
6. JFrame LibMenuTM : Frame for Transaction Menu Choices
7. JPanel paneltm : Panel in which the contents are added
8. JButton tm1 : Button with the label “Book Issue”
9. JButton tm2 : Button with the label “Book Return”
10. JButton tm3 : Button with the label “Back”
11. JButton tm4 : Button with the label “Sign out”
12. GridBagConstraints gbcLibMenutm : Used for the page layout

## BOOK ISSUE

1. JFrame tmbi : Frame for Book Issue
2. JPanel tmvip : Panel in which the contents are added
3. JLabel tmbirid : Label saying “Reader ID”
4. JLabel tmbibid : Label saying “Book ID”
5. JLabel tmbiid : Label saying “Issue Date”
6. JLabel tmbiprd : Label saying “Due Date”
7. JLabel tmbiuid : Label saying “ID”
8. JTextField tmbirid1 : Text Field for Reader id
9. JTextField tmbibid1 : Text Field for Book id
10. JTextField tmbiid1 : Text Field for Issue Date (showing today’s date by default)
11. JTextField tmbiprd1 : Text Field for Due Date (showing 7 days later’s date by default)
12. JLabel tmbiuid1 : Label to display id of the librarian whose account it is
13. JButton tmbiissue : Button with the label “Issue”
14. JButton tmbiback : Button with the label “Back”
15. JButton tmbiso : Button with the label “Sign out”
16. GridBagConstraints tmbigbc : Used for the page layout
17. int proceed : Restricts entry of line of execution if some earlier conditions have not been satisfied
18. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied

19. String readeridtmbl : Stores the entry in tmbirid1 (ie; Reader id)
20. String bookidtmbl : Stores the entry in tmbibid1 (ie; Book id)
21. String bookissuetmbl : Stores the entry in tmbiid1 (ie; Issue Date)
22. String bookplanneddatetmbl : Stores the entry in tmbiprd1 (ie; Due Date)
23. java.util.Date dttrial1 : Stores bookissuetmbl in the date format
24. java.util.Date dttrial2 : Stores bookplanneddatetmbl in the date format
25. String sqltmbitrnmenu : Stores sql for Transaction Menu
26. String sqltmbibkmenu : Stores sql for Book Master
27. String sqltmbirdmenu : Stores sql for Reader Master
28. rstmbirdmenu : Runs sqltmbirdmenu
29. rstmbibkmenu : Runs sqltmbibkmenu
30. rstmbitrnmenu : Runs sqltmbitrnmenu
31. String strReaderIdTrnMenu : Stores the number of books with the reader
32. ReaderIdTrnMenu : Stores String strReaderIdTrnMenu in the Integer format
33. String readeractive : Stores whether the reader is active or not
34. String bookactive : Stores whether the book is active or not
35. String bookavailable : Stores whether the book is available or not
36. int selected\_option\_trn\_menu : Stores the value of the entry in the ConfirmDialog

## BOOK RETURN

1. JTextField bi\_input : Text Field for taking the book id which is to be returned
2. JPanel biinputpanel : Panel in which the contents are added
3. int selected\_option\_book\_return : Stores the value of the entry in the ConfirmDialog
4. int temp\_b\_return : Restricts entry of line of execution if some earlier conditions have not been satisfied
5. String B\_ID\_INPUT : Stores the entry in JTextField bi\_input
6. sql\_book\_return\_trnmenu : Stores the sql for Transaction Menu
7. rs\_book\_return\_trnmenu : Runs the sql for Transaction Menu
8. String r\_id : Stores the corresponding reader id
9. String b\_id : Stores the corresponding book id
10. String b\_issue\_date : Stores the corresponding issue date
11. String b\_planned\_return\_date : Stores the corresponding due date
12. String u\_login : Stores the corresponding user login who had issued the book
13. Calendar dt3 : Stores today's date
14. String today : Stores dt3 in the String format
15. JFrame b\_return\_frame : Frame for Book Return
16. JLabel r\_id\_label : Label saying "Reader Id"
17. JLabel r\_id\_data : Label to display Reader id
18. JLabel b\_id\_label : Label saying "Book Id"
19. JLabel b\_id\_data : Label to display Book id
20. JLabel b\_issue\_date\_label : Label saying "Book Issue Date"
21. JLabel b\_issue\_date\_data : Label to display Book Issue Date
22. JLabel b\_planned\_return\_date\_label : Label saying "Book Due Date"
23. JLabel b\_planned\_return\_date\_data : Label to display Due date
24. JLabel u\_login\_label : Label saying "Issue User"
25. JLabel u\_login\_data : Label to display Issuer User id
26. JLabel b\_actual\_return\_date\_label : Label saying "Book Return Date"
27. JTextField b\_actual\_return\_date\_data : Text Field to take the actual return
28. JButton return\_b\_rtn : Button with the label "Return"
29. JButton back\_b\_rtn : Button with the label "Back"
30. JButton so\_b\_rtn : Button with the label "Sign Out"
31. GridBagConstraints gbc\_book\_return : Used for the page layout
32. JPanel Book\_return\_panel : Panel in which the contents are added
33. String actualreturndate : Stores the entry in JTextField b\_actual\_return\_date\_data
34. int proceed : Restricts entry of line of execution if some earlier conditions have not been satisfied
35. java.util.Date dttrial3 : Stores String actualreturndate in the date format

36. String sql\_book\_return\_rtnmenu : sql for return menu
37. String sql\_book\_return\_trnmenu : sql for transaction menu
38. ResultSet rs\_Book\_return\_rtnmenu : ResultSet for sql\_book\_return\_rtnmenu
39. ResultSet rs\_book\_return\_trnmenu : ResultSet for sql\_book\_return\_trnmenu
40. String sql1 : sql for Book Master
41. String sql2 : sql for Reader Master
42. ResultSet rs1 : ResultSet for sql1
43. ResultSet rs2 : ResultSet for sql2
44. int num : Stores the number of books with the reader
45. String str\_num : Stores int num in the string format

## REPORT MENU

1. JButton back : Button with the label “Back” (present only when opened through a librarian’s account)

### *COMMON FOR LIBRARIAN’S REPORTS AS WELL AS REPORT USER’S REPORTS*

1. JFrame report\_menu\_choices : Frame for report menu choices
2. JPanel report\_menu\_choices\_panel : Panel in which the contents are added
3. GridBagConstraints gbc\_report\_menu\_choices : Used for the page layout
4. JButton book\_search : Button with the label “Book Search”
5. JButton reader\_search : Button with the label “Reader Search”
6. JButton due\_report : Button with the label “Due Report”
7. JButton lending\_report : Button with the label “Lending Report”
8. JButton sign\_out : Button with the label “Sign Out”

## BOOK SEARCH

1. JFrame report\_menu\_book\_search : Frame for Book Search
2. Panel report\_menu\_book\_search\_panel : Panel in which the contents are added
3. GridBagConstraints gbc\_report\_book\_search : Used for the page layout
4. JButton search : Button with the label “Search”
5. JButton back : Button with the label “Back”
6. JButton sign\_out : Button with the label “Sign Out”
7. JLabel book\_name : Label saying “Book Name”
8. JLabel author : Label saying “Author”
9. JLabel genre : Label saying “Genre”
10. String GENRE\_RM[] : String array storing the genres
11. JTextField book\_name\_tf : Text Field for Book Name
12. JTextField author\_tf : Text Field for Author
13. JComboBox genre\_tf : Combo Box for Genre
14. String bn : Stores the text entered in JTextField book\_name\_tf (ie; Book Name)
15. String au : Stores the text entered in JTextField author\_tf (ie; Author)
16. String gn : Stores the selected data in JComboBox genre\_tf
17. int temp : Restricts entry of line of execution if some earlier conditions have not been satisfied
18. String sql : Stores the sql to be runned
19. JTextField genre\_actual : Text Field to take the genres in case of Combination of Genres
20. JPanel genre\_actual\_panel : Panel in which the contents are added
21. int so\_genre\_actual : Stores the value of the entry in the ConfirmDialog
22. ResultSet rs\_bs : ResultSet for running String sql
23. String rs\_id : Stores the Book id from the table one at a time
24. String rs\_bn : Stores the Book Name from the table one at a time

25. String rs\_au : Stores the Book Author from the table one at a time
26. String rs\_pr : Stores the Book Price from the table one at a time
27. String rs\_cu : Stores the Book Currency from the table one at a time
28. String rs\_gn : Stores the Book Genre from the table one at a time
29. String rs\_av : Stores the Book Availabel from the table one at a time
30. String rs\_ac : Stores the Book Active from the table one at a time
31. int len\_rs\_bn : Stores the length of String rs\_bn
32. int len\_rs\_au : Stores the length of String rs\_au
33. int len\_rs\_pr : Stores the length of String rs\_pr
34. int len\_rs\_gn : Stores the length of String rs\_gn
35. int a : is used for running for loops

## READER SEARCH

1. JFrame report\_menu\_reader\_search : Frame for Reader Search
2. JPanel report\_menu\_reader\_search\_panel : Panel in which the contents are added
3. GridBagConstraints gbc\_report\_reader\_search : Used for the page layout
4. JButton Search : Button with the label "Search"
5. JButton back : Button with the label "Back"
6. JButton sign\_out : Button with the label "Sign out"
7. JLabel first\_name : Label saying "First Name"
8. JLabel last\_name : Label saying "Last Name"
9. JLabel standard : Label saying "Standard"
10. JTextField first\_name\_tf : Text Field for First Name
11. JTextField last\_name\_tf : Text Field for Last Name
12. JTextField standard\_tf : Text Field for Standard
13. String fn : Stores the text entry in JTextField first\_name\_tf
14. String ln : Stores the text entry in JTextField last\_name\_tf
15. String sta : Stores the text entry in JTextField standard\_tf
16. String sql : Stores the sql to be runned
17. ResultSet rs\_rs : ResultSet for running String sql
18. String rs\_id : Stores the Reader id from the table one at a time
19. String rs\_fn : Stores the Reader First Name from the table one at a time
20. String rs\_ln : Stores the Reader Last Name from the table one at a time
21. String rs\_sta : Stores the Reader Standard from the table one at a time
22. String rs\_di : Stores the Reader Division from the table one at a time
23. String rs\_rn : Stores the Reader Roll No from the table one at a time
24. String rs\_ridc : Stores the Reader No of books with the reader from the table one at a time
25. String rs\_ac : Stores the Reader active from the table one at a time
26. int len\_rs\_fn : Stores the length of String rs\_fn
27. int len\_rs\_ln : Stores the length of String rs\_ln
28. int a : is used for running the for loops

## DUE REPORT

1. String sql : Stores the sql to be runned
2. ResultSet rs\_dr : ResultSet for running String sql
3. String rs\_rid : Stores the Reader id from the table one at a time
4. String rs\_bid : Stores the Book id from the table one at a time
5. String rs\_id : Stores the issue date from the table one at a time
6. String rs\_dd : Stores the due date from the table one at a time
7. String rs\_ul : Stores the user login issuer from the table one at a time

## LENDING REPORT

1. String sql : Stores the sql to be runned
2. ResultSet rs\_lr : ResultSet for running the String sql
3. String rs\_rid : Stores the Reader id from the table one at a time

4. String rs\_bid : Stores the Book id from the table one at a time
5. String rs\_id : Stores the issue date from the table one at a time
6. String rs\_dd : Stores the due date from the table one at a time
7. String rs\_rd : Stores the return date from the table one at a time
8. String rs\_ul : Stores the user login issuer from the table one at a time
9. String rs\_ulr : Stores the user login receiver from the table one at a time

## ADMIN MENU

1. JFrame AC : Frame for Admin Choices
2. JPanel ACP : Panel in which the contents are added
3. GridBagConstraints gbc : Used for the page layout
4. JButton NU : Button with the label “New User”
5. JButton MU : Button with the label “Modify User”
6. JButton RU : Button with the label “Remove User”
7. JButton CP : Button with the label “Change Password”
8. JButton SO : Button with the label “Sign Out”

## NEW USER

1. JFrame AdminMenuNU : Frame for New User
2. JPanel AdminMenuNU : Panel in which the contents are added
3. JLabel LInu : Label saying “Login id”
4. JLabel UNnu : Label saying “User Name”
5. JLabel PWnu : Label saying “Password”
6. JLabel Cnu : Label saying “Category”
7. String categorynu[] : String array storing the user categories
8. JComboBox cnu : Combo Box for category
9. JTextField linu : Text Field for login id
10. JTextField unnu : Text Field for user name
11. JPasswordField pwnu : Password Field for Password
12. JButton amnuadd : Button with the label “Add”
13. JButton amnub : Button with the label “Back”
14. JButton amnuso : Button with the label “Sign out”
15. GridBagConstraints gbc1 : Used for the page layout
16. String sqlnu : Stores the sql to be runned
17. ResultSet rsnu : ResultSet for running String sqlnu
18. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
19. String LoginIdnu : Stores the text entry in JTextField linu
20. String UserNamenu : Stores the text entry in JTextField unnu
21. String PassWordnu : Stores the password entry in JPasswordField pwnu
22. String Categorynu : Stores the data selection in JComboBox cnu
23. String usertempnu : Stores all the login ids from the database on ata time

## MODIFY USER

1. JLabel amlabeluserentrymodify : Label saying “Login id”
2. JTextField amuserentrymodify : Text Field for login id
3. JPanel amentrymodify : Panel in which the contents are added
4. int muSelectedOption : Stores the value of the entry in the ConfirmDialog
5. int tempmu : Restricts entry of line of execution if some earlier conditions have not been satisfied
6. String sqlrsmu : Stores the sql to be runned
7. ResultSet rsmu : ResultSet for running String sqlrsmu
8. JTextField limu : Text Field for login id
9. JTextField unmu : Text Field for User Name

10. String PWmu1 : Stores the password
11. JFrame AdminMenuMU : Frame for Modify User
12. JPanel AdminMenuMUp : Panel in which the contents are added
13. JLabel LImu : Label saying "Login Id"
14. JLabel UNmu : Label saying "User Name"
15. JLabel PWmu : Label saying "Password"
16. JLabel Cmu=new JLabel("Category");
17. String categorymu[] : String array storing the categories
18. JComboBox cmu : Combo Box for categories
19. JButton ammumodify : Button saying "Modify"
20. JButton ammub : Button saying "Back"
21. JButton ammuso : Button saying "Sign Out"
22. GridBagConstraints gbc1 : Used for the page layout
23. int PWmu1len : Stores the length of String PWmu1
24. char chtempammu : Stores '\*' to replace the characters of the password
25. String PWmu2 : Stores the string with each character of the password replaced with '\*'
26. int temp\_in\_ammu : Runs the for loop
27. JLabel pwmu : Label saying String PWmu2
28. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied
29. String LoginIdmu : Stores the entry in JTextField limu
30. String UserNamemu : Stores the entry in JTextField unmu
31. String Categorymu : Stores the selection in JComboBox cmu

## REMOVE USER

1. int tempru : Restricts entry of line of execution if some earlier conditions have not been satisfied
2. JLabel amlabeluserentryremove : Label saying "Login Id"
3. JTextField amuserentryremove : Text Field for login id
4. JPanel amentryremove : Panel in which the contents are added
5. int ruSelectedOption : Stores the value of the entry in the ConfirmDialog
6. String aruuserentry : Stores the entry in JTextField amuserentryremove
7. String sqlru : Stores the sql to be runned
8. ResultSet rsru : ResultSet running String sqlru
9. JFrame AdminMenuRU : Frame for Remove User
10. JPanel AdminMenuRUp
11. JLabel LIru : Label saying "Login Id"
12. JLabel UNru : Label saying "User Name"
13. JLabel PWru : Label saying "Pasword"
14. JLabel Cru : Label saying "Category"
15. JLabel RMru : Label saying "Remark"
16. JTextField RMru1 : Text Field for Remark
17. String strliru : Stores the login id
18. String strunru : Stores the user name
19. String strcaru : Stores the Category
20. String PWru1 : Stores the password
21. JButton amrurremove : Button with the label "Remove"
22. JButton amrub : Button with the label "Back"
23. JButton amruso : Button with the label "Sign Out"
24. GridBagConstraints gbc1 : Used for the page layout
25. int PWru1len : Stores the length of String PWru1
26. char chtempamru : Stores '\*' to replace the characters of the password
27. String PWru2 : Stores the string with each character of the password replaced with '\*'
28. int temp\_in\_amru : Runs the loop
29. JLabel pwru Label saying String PWru2
30. JLabel liru : Label saying String strliru
31. JLabel unru : Label saying String strunru



32. JLabel caru : Label saying String strcaru
33. String remark : Stores the entry in Text Field RMrul
34. int temporary : Restricts entry of line of execution if some earlier conditions have not been satisfied

## CHANGE PASSWORD

1. int tempcp : Restricts entry of line of execution if some earlier conditions have not been satisfied
2. JLabel amlabeluserentrychangepassword : Label saying "Login id"
3. JTextField amuserentrychangepassword : Text Field for Change Password
4. JPanel amentrychangepassword : Panel in which the contents are added
5. int cpSelectedOption : Stores the value of the entry in the ConfirmDialog
6. String acpuserentry : Stores the entry in JTextField amuserentrychangepassword
7. String sqlcp : Stores the sql to be runned
8. ResultSet rscp : ResultSet for running String sqlcp
9. JFrame AdminMenuCP : Frame for Change Password
10. JPanel AdminMenuCPp : Panel in which the contents are added
11. JLabel LIcp : Label saying "Login id"
12. JLabel UNcp : Label saying "User Name"
13. JLabel PWoldcp : Label saying "Old Password"
14. JLabel PWnewcp : Label saying "New Password"
15. JLabel PWconfirmcp : Label saying "Confirm Password"
16. JLabel Ccp : Label saying "Category"
17. JPasswordField pwoldcp : Password Field for Old Password
18. JPasswordField pwnewcp : Password Field for New Password
19. JPasswordField pwconfirmcp : Password Field for Confirm Password
20. JButton amcpchangepassword : Button with the label "Change"
21. JButton amcpb : Button with the label "Back"
22. JButton amcpso : Button with the label "Sign Out"
23. GridBagConstraints gbcamcp : Used for the page layout
24. String strlicp : Stores the login id
25. String struncp : Stores the user name
26. String strcpcp : Stores the category
27. JLabel licp : Label saying String strlicp
28. JLabel uncp : Label saying String struncp
29. JLabel cacp : Label saying String strcpcp
30. String oldchangeassadminmenu : Stores the entry in Text Field pwoldcp
31. String newchangeassadminmenu : Stores the entry in Text Field pwnewcp
32. String confirmchangeassadminmenu : Stores the entry in Text Field pwconfirmcp

## Code

```
import java.io.*;
import java.lang.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import java.util.*;
import java.text.*;
public class Trial
{
    //CONNECT
    Connection con;
    Statement st;
    ResultSet rs;

    //LOGIN PAGE
    JFrame loginframe=new JFrame("USER LOGIN");
    JLabel l=new JLabel("LOGIN ID:");
    JLabel l1=new JLabel("PASSWORD:");
    JTextField t=new JTextField(20);
    JPasswordField t1=new JPasswordField(20);
    JButton b=new JButton("LOGIN");
    String user,pass;
    String USER_TYPE[]{"ADMIN","GENERAL USER","LIBRARIAN"};

    //LIBRARIAN CHOICES
    JFrame LibMenuCHOICES;

    //LIBRARIAN MASTER UPDATE
    JFrame LibMenuMASTERUPDATE;

    //READER MASTER
    JFrame LibMenuRM;

    //NEW READER
    JFrame murmnr;
    JTextField murmnrfn1,murmnrln1,murmnrd1,murmnrid1;
    String
    STANDARD[]{"Nursery","Jr.Kg","Sr.Kg","1","2","3","4","5","6","7","8","9","10","11","12"};
    String ROLLNO[]=new String[100];
    JComboBox murmnrs1,murmnrrn1;
    ResultSet rsmurmnr;

    //MODIFY READER
    JTextField murmmrentryidtext;
```



```

    JTextField
    LibMenuMURMMRfn1,LibMenuMURMMRln1,LibMenuMURMMRd1,LibMenuMURMMRid1
    ;

    JComboBox LibMenuMURMMRs1,LibMenuMURMMRrn1;
    ResultSet rsLibMenumurmmr;
    JFrame murmmr;

    //DELETE READER
    JTextField murmdrentryidtext;
    JLabel
    LibMenuMURMDRfn1,LibMenuMURMDRln1,LibMenuMURMDRs1,LibMenuMURMDRd1,LibMenuMURMDRrn1,LibMenuMURMDRid1;
    ResultSet rsLibMenumurmdr;
    JFrame murmdr;
    JTextField drremark;

    //BOOK MASTER
    JFrame LibMenubm;

    //NEW BOOK
    JFrame mubmnb;
    JComboBox mubmnbcu1,mubmnbgn1;
    String
    CURRENCY[]={ "INR","EUR","USD"},GENRE[]={ "Autobiography","Fiction","Fantasy","Suspense","Romance","Non-fiction","Mythology","Philosophy","Psychology","Reference Books","Dictionaries and Thesaurus","Others","Combination of genres"};
    String idmubmnb,bnamemubmnb,authormubmnb,currencymubmnb;
    int pricemubmnb;
    JTextField mubmnbbn1,mubmnbau1,mubmnbpr1,mubmnbid1,genretrial;
    ResultSet rsmubmnb;
    String genremubmnb;
    int temporary;

    //MODIFY BOOK
    JFrame mubmmb;
    JComboBox mubmmbcu1,mubmmbgn1;
    JTextField mubmmbentryidtext;
    String mubmmbID,mubmmbBN,mubmmbAU,mubmmbCU,mubmmbGN;
    int mubmmbPR;
    JTextField mubmmbbn1,mubmmbau1,mubmmbpr1,mubmmbid1;
    ResultSet rsmubmmb;

    //DELETE BOOK
    JTextField mubmdbentryidtext;
    JLabel mubmdbbn1,mubmdbau1,mubmdbpr1,mubmdbcu1,mubmdbgn1,mubmdbid1;
    ResultSet rsmubmdb;
    JFrame mubmdb;
    JTextField dbremarktext;

```

```

//TRANSACTION MENU
JFrame LibMenuTM;
String today,duedate;
SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy");

//BOOK ISSUE
JFrame tmbi;
JTextField tmbirid1,tmbibid1,tmbiid1,tmbiprd1;
ResultSet rstmbitrnmenu,rstmbibkmenu,rstmbirdmenu;
String readeridtmbi,bookidtmbi;
int ReaderIdTrnMenu;

//BOOK RETURN
ResultSet rs_book_return_trnmenu;
ResultSet rs_Book_return_rtnmenu;
JFrame b_return_frame;
JTextField b_actual_return_date_data;
String r_id="",b_id="",b_issue_date="",b_planned_return_date="",u_login="";
String sql_book_return_trnmenu;

//LIBRARIAN MAINTAINANCE
JFrame LibMenuMM;
ResultSet rslmmm;
JPasswordField pwoldlmmm,pwnewlmmm,pwconfrmlmmm;
String strpasslmmm;

//REPORT MENU
JFrame report_menu_choices,report_menu_book_search,report_menu_reader_search;
JTextField book_name_tf,author_tf;
JComboBox genre_tf;
JTextField first_name_tf,last_name_tf,standard_tf;
ResultSet rs_bs,rs_rs,rs_dr,rs_lr;

//ADMIN MENU
ResultSet rsnu;
ResultSet rsmu;
ResultSet rsru;
ResultSet rscp;
String amuserentry="";
JTextField linu,unnu;
JTextField limu,unmu;
JPasswordField pwnu;
JPasswordField pwoldcp,pwnewcp,pwconfirmcp;
JComboBox cnu;
JComboBox cmu;
JFrame AC,AdminMenuNU,AdminMenuMU,AdminMenuRU,AdminMenuCP;
String strpwcp;

```

```

JTextField RMru1;

public Trial()
{
    initialize();
    connect();
    frame();
}

public void initialize()
{
    for(int i=1;i<=100;i++)
    {
        ROLLNO[(i-1)]=String.valueOf(i);
    }
}

public void connect()//CONNECTION WITH THE DATABASE
{
    try
    {
        String driver="sun.jdbc.odbc.JdbcOdbcDriver";
        Class.forName(driver);
        String db="jdbc:odbc:Final";
        con=DriverManager.getConnection(db);

        st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
        ABLE);
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}

public void frame()//LOGIN PAGE DESIGNING
{
    loginframe.setSize(370,300);
    loginframe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    loginframe.setVisible(true);
    JPanel p=new JPanel(new GridBagLayout());
    t.setText("");
    t1.setText("");
    user="";
    pass="";

    GridBagConstraints gbcLoginPage=new GridBagConstraints();

```

```

gbcLoginPage.insets=new Insets(10,10,10,10);
gbcLoginPage.gridx=0;
gbcLoginPage.gridy=0;
p.add(l,gbcLoginPage);

gbcLoginPage.insets=new Insets(10,10,10,10);
gbcLoginPage.gridx=1;
gbcLoginPage.gridy=0;
p.add(t,gbcLoginPage);

gbcLoginPage.insets=new Insets(10,10,10,10);
gbcLoginPage.gridx=0;
gbcLoginPage.gridy=1;
p.add(l1,gbcLoginPage);

gbcLoginPage.insets=new Insets(10,10,10,10);
gbcLoginPage.gridx=1;
gbcLoginPage.gridy=1;
p.add(t1,gbcLoginPage);

gbcLoginPage.insets=new Insets(10,10,10,10);
gbcLoginPage.gridx=1;
gbcLoginPage.gridy=2;
p.add(b,gbcLoginPage);

loginframe.add(p);
b.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e)
    {
        btnAction();
    }
});
}

public void btnAction()//BUTTON ACTION FOR "LOGIN"
{
    user=t.getText().trim();
    pass=t1.getText().trim();
    try
    {
        String sql="select * from USER_MST where U_LOGIN='"+user+"'and
U_PASSWORD='"+pass+"'";
        rs=st.executeQuery(sql);
        String type="";
        if(rs.next())
        {
            int temp=0;
            type=rs.getString("U_CATEGORY");

```

```

        for(int trial=0;trial<=2;trial++)
        {
            temp=trial;
            if(type.equals(USER_TYPE[trial]))
            {
                trial=10;
            }
        }
        if(temp==0)
        {
            rs.close();
            loginframe.setVisible(false);
            JOptionPane.showMessageDialog(null,"You have opened through Admin Login");
            AdminChoice();
        }
        else if(temp==1)
        {
            rs.close();
            loginframe.setVisible(false);
            JOptionPane.showMessageDialog(null,"You have opened through Report User");
            ReportMenu();
        }
        else
        {
            rs.close();
            loginframe.setVisible(false);
            JOptionPane.showMessageDialog(null,"You have opened through Librarian Login");
            LibMenu();
        }
    }
    else
    {
        int selectedOption=JOptionPane.showConfirmDialog(null,"It is a wrong LOGIN or
PASSWORD. Do you want to re enter?","INVALID ENTRY",JOptionPane.YES_NO_OPTION);
        if(selectedOption==JOptionPane.NO_OPTION)
            System.exit(0);
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception");
}
}

public void LibMenu()//DESIGNING OF PAGE FOR LIBRARIAN'S CHOICES
{
    LibMenuCHOICES=new JFrame("CHOICES");
    LibMenuCHOICES.setVisible(true);
}

```

```

LibMenuCHOICES.setSize(700,500);
LibMenuCHOICES.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JPanel panel1=new JPanel(new GridBagLayout());
GridBagConstraints gbcLibMenuChoices=new GridBagConstraints();
JButton b1,b2,b3,b4,b5;
b1=new JButton("  Master Update  ");
b2=new JButton("Maintanance Menu");
b3=new JButton(" Transaction Menu ");
b4=new JButton("   Report Menu   ");
b5=new JButton("      Sign out      ");
b1.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuCHOICES.setVisible(false);
        LibMenuumu();
    }
});
b2.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuCHOICES.setVisible(false);
        JOptionPane.showMessageDialog(null,"YOU HAVE THE OPTION TO CHANGE
ONLY YOUR OWN PASSWORD");
        LibMenuumm();
    }
});
b3.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuCHOICES.setVisible(false);
        date();
    }
});
b4.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuCHOICES.setVisible(false);
        LibMenuReportMenu();
    }
});
b5.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuCHOICES.setVisible(false);
        main();
    }
});
gbcLibMenuChoices.insets=new Insets(10,10,10,10);

```

```

gbcLibMenuChoices.gridx=0;
gbcLibMenuChoices.gridy=1;
panel1.add(b1,gbcLibMenuChoices);
gbcLibMenuChoices.insets=new Insets(10,10,10,10);
gbcLibMenuChoices.gridx=0;
gbcLibMenuChoices.gridy=2;
panel1.add(b2,gbcLibMenuChoices);
gbcLibMenuChoices.insets=new Insets(10,10,10,10);
gbcLibMenuChoices.gridx=0;
gbcLibMenuChoices.gridy=3;
panel1.add(b3,gbcLibMenuChoices);
gbcLibMenuChoices.insets=new Insets(10,10,10,10);
gbcLibMenuChoices.gridx=0;
gbcLibMenuChoices.gridy=4;
panel1.add(b4,gbcLibMenuChoices);
gbcLibMenuChoices.insets=new Insets(10,10,10,10);
gbcLibMenuChoices.gridx=0;
gbcLibMenuChoices.gridy=5;
panel1.add(b5,gbcLibMenuChoices);
LibMenuCHOICES.add(panel1);
}

public void LibMenuMu()//DESIGNING OF PAGE FOR MASTER UPDATE OPTIONS
{
    LibMenuMASTERUPDATE=new JFrame("Master Update");
    LibMenuMASTERUPDATE.setVisible(true);
    LibMenuMASTERUPDATE.setSize(700,500);
    LibMenuMASTERUPDATE.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel panelmu=new JPanel(new GridBagLayout());
    JButton mu1=new JButton("Reader Master");
    JButton mu2=new JButton(" Book Master ");
    JButton mu3=new JButton(" Back ");
    JButton mu4=new JButton(" Sign out ");
    GridBagConstraints gbcLibMenuMasterUpdate=new GridBagConstraints();
    mu1.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent aemu)
        {
            LibMenuMASTERUPDATE.setVisible(false);
            LibMenuReaderMaster();
        }
    });
    mu2.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent aemu)
        {
            LibMenuMASTERUPDATE.setVisible(false);
            LibMenuBookMaster();
        }
    });
}

```

```

    });
    mu3.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuMASTERUPDATE.setVisible(false);
            LibMenu();
        }
    });

    mu4.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuMASTERUPDATE.setVisible(false);
            main();
        }
    });

    gbcLibMenuMasterUpdate.insets=new Insets(10,10,10,10);
    gbcLibMenuMasterUpdate.gridx=0;
    gbcLibMenuMasterUpdate.gridy=0;
    panelmu.add(mu1,gbcLibMenuMasterUpdate);
    gbcLibMenuMasterUpdate.insets=new Insets(10,10,10,10);
    gbcLibMenuMasterUpdate.gridx=0;
    gbcLibMenuMasterUpdate.gridy=1;
    panelmu.add(mu2,gbcLibMenuMasterUpdate);
    gbcLibMenuMasterUpdate.insets=new Insets(10,10,10,10);
    gbcLibMenuMasterUpdate.gridx=0;
    gbcLibMenuMasterUpdate.gridy=2;
    panelmu.add(mu3,gbcLibMenuMasterUpdate);
    gbcLibMenuMasterUpdate.insets=new Insets(10,10,10,10);
    gbcLibMenuMasterUpdate.gridx=0;
    gbcLibMenuMasterUpdate.gridy=3;
    panelmu.add(mu4,gbcLibMenuMasterUpdate);
    LibMenuMASTERUPDATE.add(panelmu);
}

public void LibMenuReaderMaster()//DESIGNING OF PAGE FOR READER MASTER
OPTIONS
{
    LibMenuRM=new JFrame("Reader Master");
    LibMenuRM.setVisible(true);
    LibMenuRM.setSize(700,500);
    LibMenuRM.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panelrm=new JPanel(new GridBagLayout());

    JButton rm1=new JButton(" New  Reader ");
    JButton rm2=new JButton("Modify Reader");

```



```

JButton rm3=new JButton("Delete Reader");
JButton rm4=new JButton("    Back    ");
JButton rm5=new JButton("  Sign out  ");

GridBagConstraints gbcLibMenumurm=new GridBagConstraints();

rm1.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent aemu)
    {
        LibMenuRM.setVisible(false);
        LibMenumurmnr();
    }
});
gbcLibMenumurm.insets=new Insets(10,10,10,10);
gbcLibMenumurm.gridx=0;
gbcLibMenumurm.gridy=0;
panelrm.add(rm1,gbcLibMenumurm);

rm2.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuRM.setVisible(false);
        LibMenumurmmr();
    }
});
gbcLibMenumurm.insets=new Insets(10,10,10,10);
gbcLibMenumurm.gridx=0;
gbcLibMenumurm.gridy=1;
panelrm.add(rm2,gbcLibMenumurm);

rm3.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuRM.setVisible(false);
        LibMenumurmdr();
    }
});
gbcLibMenumurm.insets=new Insets(10,10,10,10);
gbcLibMenumurm.gridx=0;
gbcLibMenumurm.gridy=2;
panelrm.add(rm3,gbcLibMenumurm);

rm4.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuRM.setVisible(false);
        LibMenumu();
    }
});

```

```

        }
    });
    gbcLibMenuurm.insets=new Insets(10,10,10,10);
    gbcLibMenuurm.gridx=0;
    gbcLibMenuurm.gridy=3;
    panelrm.add(rm4,gbcLibMenuurm);

    rm5.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuRM.setVisible(false);
            main();
        }
    });
    gbcLibMenuurm.insets=new Insets(10,10,10,10);
    gbcLibMenuurm.gridx=0;
    gbcLibMenuurm.gridy=4;
    panelrm.add(rm5,gbcLibMenuurm);

    LibMenuRM.add(panelrm);
}

public void LibMenuurmnr()//DESIGNING OF PAGE FOR ADDING A READER (TO
TAKE THE INPUTS)
{
    murmnr=new JFrame("New Reader");
    murmnr.setVisible(true);
    murmnr.setSize(700,500);
    murmnr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel murmnrp=new JPanel(new GridBagLayout());
    JLabel murmnrfn=new JLabel("First Name");
    JLabel murmnrln=new JLabel("Last Name");
    JLabel murmnrs=new JLabel("Standard");
    JLabel murmnrd=new JLabel("Division");
    JLabel murmnrrn=new JLabel("Roll No");
    JLabel murmnrid=new JLabel("ID");
    murmnrfn1=new JTextField(20);
    murmnrln1=new JTextField(20);
    murmnrs1=new JComboBox(STANDARD);
    murmnrd1=new JTextField(10);
    murmnrrn1=new JComboBox(ROLLNO);
    murmnrid1=new JTextField(10);
    JButton murmnradd=new JButton("Add");
    murmnradd.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e)
        {
            btnActionmurmnr();
        }
    })
}

```

```

});
JButton murmnrback=new JButton("Back");
murmnrback.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        murmnr.setVisible(false);
        LibMenuReaderMaster();
    }
});
JButton murmnrso=new JButton("Sign out");
murmnrso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        murmnr.setVisible(false);
        System.exit(0);
    }
});
GridBagConstraints murmnr gbc=new GridBagConstraints();
murmnrfn1.setText("");
murmnrln1.setText("");

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=0;
murmnrp.add(murmnrfn,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=0;
murmnrp.add(murmnrfn1,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=1;
murmnrp.add(murmnrln,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=1;
murmnrp.add(murmnrln1,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=2;
murmnrp.add(murmnrns,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);

```

```

murmnr gbc.gridx=1;
murmnr gbc.gridy=2;
murmnr p.add(murmnrsl,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=3;
murmnr p.add(murmnr d,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=3;
murmnr p.add(murmnr d1,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=4;
murmnr p.add(murmnr rn,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=4;
murmnr p.add(murmnr rn1,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=5;
murmnr p.add(murmnr id,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=5;
murmnr p.add(murmnr id1,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=0;
murmnr gbc.gridy=6;
murmnr p.add(murmnr add,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=1;
murmnr gbc.gridy=6;
murmnr p.add(murmnr back,murmnr gbc);

murmnr gbc.insets=new Insets(10,10,10,10);
murmnr gbc.gridx=2;
murmnr gbc.gridy=6;

```

```

        murmnrp.add(murmnrso,murmnrabc);

        murmnr.add(murmnrp);
        JOptionPane.showMessageDialog(null,"PLEASE ENTER THE GR NO OF THE READER
        FOR THE ID");
    }

    public void btnActionmurmnr()//BUTTON ACTION FOR "ADD" (ADDING IN THE
    DATABASE)
    {
        try
        {
            String sqlmurmnr="select * from READER_MST";
            rsmurmnr=st.executeQuery(sqlmurmnr);
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while connecting");
        }
        int temporary=1;
        String IDmurmnr=murmnrId1.getText().trim();
        String fnamemurmnr=murmnrFn1.getText().trim();
        String lnamemurmnr=murmnrLn1.getText().trim();
        String standardmurmnr=murmnrS1.getSelectedItem().toString();
        String strrollnomurmnr=murmnrRn1.getSelectedItem().toString();
        int rollnomurmnr=Integer.parseInt(strrollnomurmnr);
        String divisionmurmnr=murmnrD1.getText().trim();
        try
        {
            if(IDmurmnr.equals("") || fnamemurmnr.equals("") || lnamemurmnr.equals("") ||
            divisionmurmnr.equals(""))
            {
                JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been left
                without data");
                temporary++;
            }
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while blank textfield check");
        }
        try
        {
            while(temporary==1 && rsmurmnr.next())
            {
                String
                standardtempmurmnr=rsmurmnr.getString("R_STANDARD"),divisiontempmurmnr=rsmurmnr.ge
                tString("R_DIVISION"),strrollnotempmurmnr=rsmurmnr.getString("R_ROLLNO"),idtempmurm
                nr=rsmurmnr.getString("R_ID");
            }
        }
    }

```

```

        int rollnotempmurmnr=Integer.parseInt(strollnotempmurmnr);
        if(standardmurmnr.equals(standardtempmurmnr) &&
rollnomurmnr==rollnotempmurmnr && divisionmurmnr.equals(divisiontempmurmnr))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another reader in the same
standard and division with the same roll no");
            temporary++;
        }
        if(idtempmurmnr.equals(IDmurmnr))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another reader with the
same id");
            temporary++;
        }
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while checking 1");
}
try
{
    while(temporary==1 && rsmurmnr.previous())
    {
        String
standardtempmurmnr=rsmurmnr.getString("R_STANDARD"),divisiontempmurmnr=rsmurmnr.ge
tString("R_DIVISION"),strollnotempmurmnr=rsmurmnr.getString("R_ROLLNO"),idtempmurm
nr=rsmurmnr.getString("R_ID");
        int rollnotempmurmnr=Integer.parseInt(strollnotempmurmnr);
        if(standardmurmnr.equals(standardtempmurmnr) &&
rollnomurmnr==rollnotempmurmnr && divisionmurmnr.equals(divisiontempmurmnr))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another reader in the same
standard and division with the same roll no");
            temporary++;
        }
        if(idtempmurmnr.equals(IDmurmnr))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another reader with the
same id");
            temporary++;
        }
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while checking 2");
}
try

```

```

{
    if(temporary==1)
    {
        rsmurmnr.moveToInsertRow();
        rsmurmnr.updateString("R_FNAME",fnamemurmnr);
        rsmurmnr.updateString("R_LNAME",lnamemurmnr);
        rsmurmnr.updateString("R_STANDARD",standardmurmnr);
        rsmurmnr.updateString("R_DIVISION",divisionmurmnr);
        rsmurmnr.updateInt("R_ROLLNO",rollnomurmnr);
        rsmurmnr.updateString("R_ID",IDmurmnr);
        rsmurmnr.updateString("R_ACTIVE","Yes");
        rsmurmnr.updateString("RID_COUNT","0");
        rsmurmnr.insertRow();
        rsmurmnr.close();
        st.close();

st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
ABLE);
        JOptionPane.showMessageDialog(null,"Reader Added");
        murmnr.setVisible(false);
        LibMenuReaderMaster();
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while adding");
}
}

public void LibMenumurmnr()//TAKING THE RAEDER ID WHICH IS TO BE MODIFIED
{
    murmmrentryidtext=new JTextField(10);
    JLabel murmmrentryidlabel=new JLabel("Please enter the reader id");
    JPanel murmmrentryidpanel=new JPanel();
    murmmrentryidpanel.add(murmmrentryidlabel);
    murmmrentryidpanel.add(murmmrentryidtext);
    int
selectedoptionIDModifyReader=JOptionPane.showConfirmDialog(null,murmmrentryidpanel,"ID"
,JOptionPane.OK_CANCEL_OPTION);
    if(selectedoptionIDModifyReader==JOptionPane.OK_OPTION)
    {
        btnActionIDmurmnr();
    }
    else
    {
        LibMenuReaderMaster();
    }
}
}

```

```
public void btnActionIDmurmrmr()//DESIGNING OF PAGE FOR MODIFYING A READER  
(TO MODIFY THE DATA)
```

```
{  
    String ID=murmmrentryidtext.getText().trim();  
    int temporaryLibMenuMURMMR=0;  
    LibMenuMURMMRfn1=new JTextField(20);  
    LibMenuMURMMRln1=new JTextField(20);  
    LibMenuMURMMRs1=new JComboBox(STANDARD);  
    LibMenuMURMMRd1=new JTextField(10);  
    LibMenuMURMMRrn1=new JComboBox(ROLLNO);  
    LibMenuMURMMRid1=new JTextField(10);  
    try  
    {  
        String sqlLibMenumurmmr="select * from READER_MST where R_ID='"+ID+"' and  
R_ACTIVE='Yes'";  
        rsLibMenumurmmr=st.executeQuery(sqlLibMenumurmmr);  
    }  
    catch(Exception ex)  
    {  
        JOptionPane.showMessageDialog(null,"Exception Before Button");  
    }  
    try  
    {  
        if(rsLibMenumurmmr.next())  
        {  
            LibMenuMURMMRfn1.setText(rsLibMenumurmmr.getString("R_FNAME"));  
            LibMenuMURMMRln1.setText(rsLibMenumurmmr.getString("R_LNAME"));  
            LibMenuMURMMRd1.setText(rsLibMenumurmmr.getString("R_DIVISION"));  
            LibMenuMURMMRid1.setText(rsLibMenumurmmr.getString("R_ID"));  
        }  
        else  
        {  
            JOptionPane.showMessageDialog(null,"ID DOES NOT EXIST");  
            temporaryLibMenuMURMMR++;  
            LibMenuReaderMaster();  
        }  
    }  
    catch(Exception ex)  
    {  
        JOptionPane.showMessageDialog(null,"Exception Again");  
    }  
    if(temporaryLibMenuMURMMR==0)  
    {  
        murmmr=new JFrame("Modify Reader");  
        murmmr.setVisible(true);  
        murmmr.setSize(700,500);  
        murmmr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```



```

JPanel murmmrp=new JPanel(new GridBagLayout());
JLabel murmmrfn=new JLabel("First Name");
JLabel murmmrln=new JLabel("Last Name");
JLabel murmmrs=new JLabel("Standard");
JLabel murmmrd=new JLabel("Division");
JLabel murmmrrn=new JLabel("Roll No");
JLabel murmmrid=new JLabel("ID");
JButton murmmrmodify=new JButton("Modify");
murmmrmodify.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e)
    {
        btnActionReaderModify();
    }
});
JButton murmmrback=new JButton("Back");
murmmrback.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        murmmr.setVisible(false);
        LibMenuReaderMaster();
    }
});
JButton murmmrso=new JButton("Sign out");
murmmrso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        murmmr.setVisible(false);
        main();
    }
});
GridBagConstraints gbcmurmmr=new GridBagConstraints();
gbcmurmmr.insets=new Insets(10,10,10,10);
gbcmurmmr.gridx=0;
gbcmurmmr.gridy=0;
murmmrp.add(murmmrfn,gbcmurmmr);

gbcmurmmr.insets=new Insets(10,10,10,10);
gbcmurmmr.gridx=1;
gbcmurmmr.gridy=0;
murmmrp.add(LibMenuMURMMRfn1,gbcmurmmr);

gbcmurmmr.insets=new Insets(10,10,10,10);
gbcmurmmr.gridx=0;
gbcmurmmr.gridy=1;
murmmrp.add(murmmrln,gbcmurmmr);

gbcmurmmr.insets=new Insets(10,10,10,10);

```

```
gbcmurmmr.gridx=1;  
gbcmurmmr.gridy=1;  
murmmp.add(LibMenuMURMMRln1,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=0;  
gbcmurmmr.gridy=2;  
murmmp.add(murmmrs,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=1;  
gbcmurmmr.gridy=2;  
murmmp.add(LibMenuMURMMRs1,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=0;  
gbcmurmmr.gridy=3;  
murmmp.add(murmmrd,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=1;  
gbcmurmmr.gridy=3;  
murmmp.add(LibMenuMURMMRd1,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=0;  
gbcmurmmr.gridy=4;  
murmmp.add(murmmrn,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=1;  
gbcmurmmr.gridy=4;  
murmmp.add(LibMenuMURMMRrn1,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=0;  
gbcmurmmr.gridy=5;  
murmmp.add(murmmrid,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=1;  
gbcmurmmr.gridy=5;  
murmmp.add(LibMenuMURMMRid1,gbcmurmmr);
```

```
gbcmurmmr.insets=new Insets(10,10,10,10);  
gbcmurmmr.gridx=0;  
gbcmurmmr.gridy=6;
```

```

        murmmrp.add(murmmrmodify,gbcmurmmr);

        gbcmurmmr.insets=new Insets(10,10,10,10);
        gbcmurmmr.gridx=1;
        gbcmurmmr.gridy=6;
        murmmrp.add(murmmrback,gbcmurmmr);

        gbcmurmmr.insets=new Insets(10,10,10,10);
        gbcmurmmr.gridx=2;
        gbcmurmmr.gridy=6;
        murmmrp.add(murmmrso,gbcmurmmr);

        murmmr.add(murmmrp);
        JOptionPane.showMessageDialog(null,"PLEASE NOTE THAT THE STANDARD OF
        THE READER IS BEING SHOWN NURSERY AND ROLL NO 1. CHANGE IT IF
        REQUIRED");
    }
}

    public void btnActionReaderModify()//BUTTON ACTION FOR "MODIFY" (MODIFYING
    IN THE DATABASE)
    {
        String murmmrFN=LibMenuMURMMRfn1.getText().trim();
        String murmmrLN=LibMenuMURMMRln1.getText().trim();
        String murmmrS=LibMenuMURMMRs1.getSelectedItem().toString();
        String murmmrD=LibMenuMURMMRd1.getText().trim();
        String strmurmmrRN=LibMenuMURMMRrn1.getSelectedItem().toString();
        int murmmrRN=Integer.parseInt(strmurmmrRN);
        String murmmrID=LibMenuMURMMRid1.getText().trim();
        if(murmmrFN.equals("") || murmmrLN.equals("") || murmmrD.equals("") ||
        murmmrID.equals(""))
        {
            JOptionPane.showMessageDialog(null,"SORRY BUT ONE OR MORE TEXTFIELDS
            HAVE BEEN LEFT WITHOUT DATA");
        }
        else
        {
            try
            {
                rsLibMenumurmmr.updateString("R_FNAME",murmmrFN);
                rsLibMenumurmmr.updateString("R_LNAME",murmmrLN);
                rsLibMenumurmmr.updateString("R_STANDARD",murmmrS);
                rsLibMenumurmmr.updateString("R_DIVISION",murmmrD);
                rsLibMenumurmmr.updateInt("R_ROLLNO",murmmrRN);
                rsLibMenumurmmr.updateString("R_ID",murmmrID);
                rsLibMenumurmmr.updateRow();
                JOptionPane.showMessageDialog(null,"Record Updated");
                murmmr.setVisible(false);
            }
            catch (Exception e)
            {
                JOptionPane.showMessageDialog(null,"Error: "+e.getMessage());
            }
        }
    }
}

```

```

        LibMenuReaderMaster();
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT THE READER ID MATCHES
        THAT OF AN EXISTING READER");
    }
}

}

public void LibMenumurmdr();//TAKING THE RAEDER ID WHICH IS TO BE DELETED
{
    murmdrentryidtext=new JTextField(10);
    JLabel murmdrentryidlabel=new JLabel("Please enter the reader id");
    JPanel murmdrentryidpanel=new JPanel();
    murmdrentryidpanel.add(murmdrentryidlabel);
    murmdrentryidpanel.add(murmdrentryidtext);
    int
    selectedoptionIDDeleteReader=JOptionPane.showConfirmDialog(null,murmdrentryidpanel,"ID",J
   OptionPane.OK_CANCEL_OPTION);
    if(selectedoptionIDDeleteReader==JOptionPane.OK_OPTION)
    {
        btnActionIDmurmdr();
    }
    else
    {
        LibMenuReaderMaster();
    }
}

public void btnActionIDmurmdr();//DESIGNING OF PAGE FOR MODIFYING A READER
(TO DELETE THE DATA)
{
    String ID=murmdrentryidtext.getText().trim();
    int temporaryLibMenuMURMDR=0;
    LibMenuMURMDRfn1=new JLabel();
    LibMenuMURMDRln1=new JLabel();
    LibMenuMURMDRs1=new JLabel();
    LibMenuMURMDRd1=new JLabel();
    LibMenuMURMDRrn1=new JLabel();
    LibMenuMURMDRid1=new JLabel();
    String strLibMenuMURMDRidcount1;
    int temporary=0;
    try
    {
        String sqlLibMenumurmdr="select * from READER_MST where R_ID='"+ID+"' and
        R_ACTIVE='Yes'";
        rsLibMenumurmdr=st.executeQuery(sqlLibMenumurmdr);
    }
}

```

```

    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception Before Button");
    }
    try
    {
        if(rsLibMenumurmdr.next())
        {
            strLibMenuMURMDRidcount1=rsLibMenumurmdr.getString("RID_COUNT");
            if(strLibMenuMURMDRidcount1.equals("0"))
            {

            }
            else
            {
                temporary++;
                JOptionPane.showMessageDialog(null,"SORRY BUT THE USER HAS A BOOK/S
ISSUED. SO, HE/SHE CANNOT BE DELETED");
                LibMenuReaderMaster();
            }
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    if(temporary==0)
    {
        try
        {
            if(rsLibMenumurmdr.first())
            {
                LibMenuMURMDRfn1.setText(rsLibMenumurmdr.getString("R_FNAME"));
                LibMenuMURMDRln1.setText(rsLibMenumurmdr.getString("R_LNAME"));
                LibMenuMURMDRs1.setText(rsLibMenumurmdr.getString("R_STANDARD"));
                LibMenuMURMDRd1.setText(rsLibMenumurmdr.getString("R_DIVISION"));
                LibMenuMURMDRrn1.setText(rsLibMenumurmdr.getString("R_ROLLNO"));
                LibMenuMURMDRid1.setText(rsLibMenumurmdr.getString("R_ID"));
            }
            else
            {
                JOptionPane.showMessageDialog(null,"ID DOES NOT EXIST");
                temporaryLibMenuMURMDR++;
                LibMenuReaderMaster();
            }
        }
    }
    catch(Exception ex)

```

```

{
    JOptionPane.showMessageDialog(null,"Exception Again");
}
if(temporaryLibMenuMURMDR==0)
{
    murmdr=new JFrame("Delete Reader");
    murmdr.setVisible(true);
    murmdr.setSize(700,500);
    murmdr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel murmdrp=new JPanel(new GridBagLayout());
    JLabel murmdrfn=new JLabel("First Name");
    JLabel murmdrln=new JLabel("Last Name");
    JLabel murmdrs=new JLabel("Standard");
    JLabel murmdrd=new JLabel("Division");
    JLabel murmdrrn=new JLabel("Roll No");
    JLabel murmdrid=new JLabel("ID");
    JLabel drremarklabel=new JLabel("Remark");
    drremark=new JTextField(20);
    JButton murmdrdelete=new JButton("Delete");
    murmdrdelete.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e)
        {
            btnActionReaderDelete();
        }
    });
    JButton murmdrback=new JButton("Back");
    murmdrback.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            murmdr.setVisible(false);
            LibMenuReaderMaster();
        }
    });
    JButton murmdrso=new JButton("Sign out");
    murmdrso.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            murmdr.setVisible(false);
            main();
        }
    });
    GridBagConstraints gbcmurmdr=new GridBagConstraints();
    gbcmurmdr.insets=new Insets(10,10,10,10);
    gbcmurmdr.gridx=0;
    gbcmurmdr.gridy=0;
    murmdrp.add(murmdrfn,gbcmurmdr);

```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=1;
gbcmurmdr.gridy=0;
murmdrp.add(LibMenuMURMDRfn1,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=0;
gbcmurmdr.gridy=1;
murmdrp.add(murmdrln,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=1;
gbcmurmdr.gridy=1;
murmdrp.add(LibMenuMURMDRln1,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=0;
gbcmurmdr.gridy=2;
murmdrp.add(murmdrs,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=1;
gbcmurmdr.gridy=2;
murmdrp.add(LibMenuMURMDRs1,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=0;
gbcmurmdr.gridy=3;
murmdrp.add(murmdrd,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=1;
gbcmurmdr.gridy=3;
murmdrp.add(LibMenuMURMDRd1,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=0;
gbcmurmdr.gridy=4;
murmdrp.add(murmdrrn,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=1;
gbcmurmdr.gridy=4;
murmdrp.add(LibMenuMURMDRrn1,gbcmurmdr);
```

```
gbcmurmdr.insets=new Insets(10,10,10,10);
gbcmurmdr.gridx=0;
```

```

        gbcmurmdr.gridy=5;
        murmdrp.add(murmdrid,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=1;
        gbcmurmdr.gridy=5;
        murmdrp.add(LibMenuMURMDRid1,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=0;
        gbcmurmdr.gridy=6;
        murmdrp.add(drremarklabel,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=1;
        gbcmurmdr.gridy=6;
        murmdrp.add(drremark,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=0;
        gbcmurmdr.gridy=7;
        murmdrp.add(murmdrdelete,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=1;
        gbcmurmdr.gridy=7;
        murmdrp.add(murmdrback,gbcmurmdr);

        gbcmurmdr.insets=new Insets(10,10,10,10);
        gbcmurmdr.gridx=2;
        gbcmurmdr.gridy=7;
        murmdrp.add(murmdrso,gbcmurmdr);

        murmdr.add(murmdrp);
    }
}
}

```

```

public void btnActionReaderDelete()//BUTTON ACTION FOR "DELETE" (DELETING IN
THE DATABASE)
{
    String strremarkdr=drremark.getText().trim();
    int temporarydr=0;
    if(strremarkdr.equals(""))
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE ENTER THE REASON
WHY READER IS BEING DELETED");
        temporarydr++;
    }
}

```



```

    }
    else
    {
        if(temporarydr==0)
        {
            try
            {
                rsLibMenumurmdr.updateString("R_ACTIVE","No");
                rsLibMenumurmdr.updateString("R_REMARK",streremarkdr);
                rsLibMenumurmdr.updateRow();
                JOptionPane.showMessageDialog(null,"Record Deleted");
                murmdr.setVisible(false);
                LibMenuReaderMaster();
            }
            catch(Exception ex)
            {
                JOptionPane.showMessageDialog(null,"Exception while deleting");
            }
        }
    }
}

public void LibMenuBookMaster()//DESIGNING OF PAGE FOR BOOK MASTER OPTIONS
{
    LibMenubm=new JFrame("Book Master");
    LibMenubm.setVisible(true);
    LibMenubm.setSize(700,500);
    LibMenubm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel panelbm=new JPanel(new GridBagLayout());

    JButton bm1=new JButton(" New Book ");
    JButton bm2=new JButton("Modify Book");
    JButton bm3=new JButton("Delete Book");
    JButton bm4=new JButton(" Back ");
    JButton bm5=new JButton(" Sign out ");

    GridBagConstraints gbcLibMenumubm=new GridBagConstraints();

    bm1.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenubm.setVisible(false);
            LibMenumubmn();
        }
    });
    gbcLibMenumubm.insets=new Insets(10,10,10,10);

```

```

gbcLibMenuubm.gridx=0;
gbcLibMenuubm.gridy=0;
panelbm.add(bm1,gbcLibMenuubm);

bm2.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenubm.setVisible(false);
        LibMenuubmmmb();
    }
});
gbcLibMenuubm.insets=new Insets(10,10,10,10);
gbcLibMenuubm.gridx=0;
gbcLibMenuubm.gridy=1;
panelbm.add(bm2,gbcLibMenuubm);

bm3.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenubm.setVisible(false);
        LibMenuubmdb();
    }
});
gbcLibMenuubm.insets=new Insets(10,10,10,10);
gbcLibMenuubm.gridx=0;
gbcLibMenuubm.gridy=2;
panelbm.add(bm3,gbcLibMenuubm);

bm4.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenubm.setVisible(false);
        LibMenuubm();
    }
});
gbcLibMenuubm.insets=new Insets(10,10,10,10);
gbcLibMenuubm.gridx=0;
gbcLibMenuubm.gridy=3;
panelbm.add(bm4,gbcLibMenuubm);

bm5.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenubm.setVisible(false);
        main();
    }
});

```

```

gbcLibMenuubm.insets=new Insets(10,10,10,10);
gbcLibMenuubm.gridx=0;
gbcLibMenuubm.gridy=4;
panelbm.add(bm5,gbcLibMenuubm);

LibMenuubm.add(panelbm);
}

public void LibMenuubmnb()//DESIGNING OF PAGE FOR ADDING A BOOK (TO TAKE
THE INPUTS)
{
    mubmnbn=new JFrame("New Book");
    mubmnbn.setVisible(true);
    mubmnbn.setSize(700,500);
    mubmnbn.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel mubmnbnp=new JPanel(new GridBagLayout());
    JLabel mubmnbnbn=new JLabel("Book Name");
    JLabel mubmnbnau=new JLabel("Author");
    JLabel mubmnbnpr=new JLabel("Price");
    JLabel mubmnbcu=new JLabel("Currency");
    JLabel mubmnbgcn=new JLabel("Genre");
    JLabel mubmnbnid=new JLabel("ID");
    mubmnbnbn1=new JTextField(20);
    mubmnbnau1=new JTextField(20);
    mubmnbnpr1=new JTextField(10);
    mubmnbnbcu1=new JComboBox(CURRENCY);
    mubmnbnbgcn1=new JComboBox(GENRE);
    mubmnbnid1=new JTextField(10);
    JButton mubmnbnadd=new JButton("Add");
    mubmnbnadd.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e)
        {
            btnActionmubmnbn();
        }
    });
    JButton mubmnbnback=new JButton("Back");
    mubmnbnback.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            mubmnbn.setVisible(false);
            LibMenuBookMaster();
        }
    });
    JButton mubmnbnso=new JButton("Sign out");
    mubmnbnso.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            mubmnbn.setVisible(false);

```

```

        main();
    }
});
GridBagConstraints mubmnbgbc=new GridBagConstraints();
mubmnbbn1.setText("");
mubmnbau1.setText("");
mubmnbpr1.setText("");
mubmnbid1.setText("");

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=0;
mubmnbp.add(mubmnbbn,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=0;
mubmnbp.add(mubmnbbn1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=1;
mubmnbp.add(mubmnbau,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=1;
mubmnbp.add(mubmnbau1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=2;
mubmnbp.add(mubmnbpr,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=2;
mubmnbp.add(mubmnbpr1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=3;
mubmnbp.add(mubmnbcu,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=3;

```

```

mubmnbp.add(mubmnbcu1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=4;
mubmnbp.add(mubmnbgcn,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=4;
mubmnbp.add(mubmnbgcn1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=5;
mubmnbp.add(mubmnbid,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=5;
mubmnbp.add(mubmnbid1,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=0;
mubmnbgbc.gridy=6;
mubmnbp.add(mubmnbadd,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=1;
mubmnbgbc.gridy=6;
mubmnbp.add(mubmnback,mubmnbgbc);

mubmnbgbc.insets=new Insets(10,10,10,10);
mubmnbgbc.gridx=2;
mubmnbgbc.gridy=6;
mubmnbp.add(mubmnbs0,mubmnbgbc);

mubmnbp.add(mubmnbp);
}

public void btnActionmubmnbp()//BUTTON ACTION FOR "ADD"
{
    try
    {
        String sqlmubmnbp="select * from BOOK_MST";
        rsmubmnbp=st.executeQuery(sqlmubmnbp);
    }
}

```

```

catch(Exception ex)
{
    ex.printStackTrace();
}
temporary=1;
idmubmbnb=mubmbnbid1.getText().trim();
bnamemubmbnb=mubmbnbbn1.getText().trim();
authormubmbnb=mubmbnbau1.getText().trim();
String strpricemubmbnb=mubmbnbpr1.getText().trim();
try
{
    pricemubmbnb=Integer.parseInt(strpricemubmbnb);
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE ENTER A NUMERIC
VALUE FOR PRICE");
    temporary++;
}
currencymubmbnb=mubmbnbcu1.getSelectedItem().toString();
genremubmbnb=mubmbnbg1.getSelectedItem().toString();
if(idmubmbnb.equals("") || bnamemubmbnb.equals("") || authormubmbnb.equals("") ||
strpricemubmbnb=="")
{
    JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been left
without data");
    temporary++;
}
if(temporary==1 && genremubmbnb.equals("Combination of genres"))
{
    temporary++;
    final JFrame genresmubmbnb;
    genresmubmbnb=new JFrame("PLEASE ENTER THE GENRES");
    mubmbnb.setVisible(false);
    genresmubmbnb.setVisible(true);
    genresmubmbnb.setSize(500,300);
    genresmubmbnb.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JLabel genrestriallabel=new JLabel("GENRES");
    genrestrial=new JTextField(20);
    JButton ok=new JButton("OK");

    ok.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            genremubmbnb=genrestrial.getText().trim();
            genresmubmbnb.setVisible(false);
            temporary=1;
            AddTheBook();
        }
    });
}

```

```

    }
});
JButton back=new JButton ("BACK");
back.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        temporary++;
        genresmubmbnb.setVisible(false);
        mubmbnb.setVisible(true);
    }
});
JButton so=new JButton ("SIGN OUT");
so.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        temporary++;
        genresmubmbnb.setVisible(false);
        main();
    }
});

JPanel genrespanel=new JPanel(new GridBagLayout());
GridBagConstraints gbcgenres=new GridBagConstraints();

gbcgenres.insets=new Insets(10,10,10,10);
gbcgenres.gridx=0;
gbcgenres.gridy=0;
genrespanel.add(genrestriallabel,gbcgenres);

gbcgenres.insets=new Insets(10,10,10,10);
gbcgenres.gridx=1;
gbcgenres.gridy=0;
genrespanel.add(genrestrial,gbcgenres);

gbcgenres.insets=new Insets(10,10,10,10);
gbcgenres.gridx=0;
gbcgenres.gridy=1;
genrespanel.add(ok,gbcgenres);

gbcgenres.insets=new Insets(10,10,10,10);
gbcgenres.gridx=1;
gbcgenres.gridy=1;
genrespanel.add(back,gbcgenres);

gbcgenres.insets=new Insets(10,10,10,10);
gbcgenres.gridx=2;
gbcgenres.gridy=1;

```

```

        genrespanel.add(so, gbcgenres);

        genresmubmnb.add(genrespanel);
    }
    try
    {
        while(temporary==1 && rsmubmnb.next())
        {
            String idtempmubmnb=rsmubmnb.getString("B_ID");
            if(idtempmubmnb.equals(idmubmnb))
            {
                JOptionPane.showMessageDialog(null,"Sorry but there is another Book with the
same id");
                temporary++;
            }
        }
        while(temporary==1 && rsmubmnb.previous())
        {
            String idtempmubmnb=rsmubmnb.getString("B_ID");
            if(idtempmubmnb.equals(idmubmnb))
            {
                JOptionPane.showMessageDialog(null,"Sorry but there is another Book with the
same id");
                temporary++;
            }
        }
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while checking");
    }
    if(temporary==1)
    {
        AddTheBook();
    }
}

public void AddTheBook();//ADDING IN THE DATABASE
{
    try
    {
        rsmubmnb.moveToInsertRow();
        rsmubmnb.updateString("B_NAME",bnamemubmnb);
        rsmubmnb.updateString("B_AUTHOR",authormubmnb);
        rsmubmnb.updateInt("B_PRICE",pricemubmnb);
        rsmubmnb.updateString("B_CURRENCY",currencymubmnb);
        rsmubmnb.updateString("B_GENRE",genremubmnb);
        rsmubmnb.updateString("B_ID",idmubmnb);
        rsmubmnb.updateString("B_ACTIVE","Yes");
    }
}

```



```

        rsmubmbnb.updateString("B_AVAILABLE","Yes");
        rsmubmbnb.insertRow();
        rsmubmbnb.close();
        st.close();

st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
ABLE);

        JOptionPane.showMessageDialog(null,"Book Added");
        mubmbnb.setVisible(false);
        LibMenuBookMaster();
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while Added");
    }
}

public void LibMenuMubmbnb()//TAKING THE RAEDER ID WHICH IS TO BE MODIFIED
{
    mubmbmbentryidtext=new JTextField(10);
    JLabel mubmbmbentryidlabel=new JLabel("Please enter the Book id");
    JPanel mubmbmbentryidpanel=new JPanel();
    mubmbmbentryidpanel.add(mubmbmbentryidlabel);
    mubmbmbentryidpanel.add(mubmbmbentryidtext);
    int
selectedoptionIDModifyBook=JOptionPane.showConfirmDialog(null,mubmbmbentryidpanel,"ID",
JOptionPane.OK_CANCEL_OPTION);
    if(selectedoptionIDModifyBook==JOptionPane.OK_OPTION)
    {
        btnActionIDmubmbmb();
    }
    else
    {
        LibMenuBookMaster();
    }
}

public void btnActionIDmubmbmb()//DESIGNING OF PAGE FOR MODIFYING A BOOK
(TO MODIFY THE DATA)
{
    String ID=mubmbmbentryidtext.getText().trim();
    temporary=0;
    mubmbmbbn1=new JTextField(20);
    mubmbmbau1=new JTextField(20);
    mubmbmbpr1=new JTextField(10);
    mubmbmbcu1=new JComboBox(CURRENCY);
    mubmbmbgn1=new JComboBox(GENRE);
    mubmbmbid1=new JTextField(10);

```

```

try
{
    String sqlLibMenuubmmb="select * from BOOK_MST where B_ID='"+ID+"' and
B_ACTIVE='Yes'";
    rsmubmmb=st.executeQuery(sqlLibMenuubmmb);
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception Before Button");
}
try
{
    if(rsmubmmb.next())
    {
        mubmmbbn1.setText(rsmubmmb.getString("B_NAME"));
        mubmmbau1.setText(rsmubmmb.getString("B_AUTHOR"));
        mubmmbpr1.setText(rsmubmmb.getString("B_PRICE"));
        mubmmbid1.setText(rsmubmmb.getString("B_ID"));
    }
    else
    {
        JOptionPane.showMessageDialog(null,"ID DOES NOT EXIST");
        temporary++;
        LibMenuBookMaster();
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception Again");
}
if(temporary==0)
{
    mubmmb=new JFrame("Modify Book");
    mubmmb.setVisible(true);
    mubmmb.setSize(700,500);
    mubmmb.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel mubmmbp=new JPanel(new GridBagLayout());
    JLabel mubmmbbn=new JLabel("Book Name");
    JLabel mubmmbau=new JLabel("Author");
    JLabel mubmmbpr=new JLabel("Price");
    JLabel mubmmbcu=new JLabel("Currency");
    JLabel mubmmbgn=new JLabel("Genre");
    JLabel mubmmbid=new JLabel("ID");
    JButton mubmmbmodify=new JButton("Modify");
    mubmmbmodify.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e)
        {
            btnActionBookModify();
        }
    });
}

```

```

    }
});
JButton mubmmbback=new JButton("Back");
mubmmbback.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        mubmmb.setVisible(false);
        LibMenuBookMaster();
    }
});
JButton mubmmbso=new JButton("Sign out");
mubmmbso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        mubmmb.setVisible(false);
        main();
    }
});
GridBagConstraints gbcubmmb=new GridBagConstraints();
gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=0;
gbcubmmb.gridy=0;
mubmmbp.add(mubmmbbn,gbcubmmb);

gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=1;
gbcubmmb.gridy=0;
mubmmbp.add(mubmmbbn1,gbcubmmb);

gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=0;
gbcubmmb.gridy=1;
mubmmbp.add(mubmmbau,gbcubmmb);

gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=1;
gbcubmmb.gridy=1;
mubmmbp.add(mubmmbau1,gbcubmmb);

gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=0;
gbcubmmb.gridy=2;
mubmmbp.add(mubmmbpr,gbcubmmb);

gbcubmmb.insets=new Insets(10,10,10,10);
gbcubmmb.gridx=1;
gbcubmmb.gridy=2;

```

```
mubmmbp.add(mubmmbpr1,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=0;  
gbcmubmmb.gridy=3;  
mubmmbp.add(mubmmbcu,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=1;  
gbcmubmmb.gridy=3;  
mubmmbp.add(mubmmbcu1,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=0;  
gbcmubmmb.gridy=4;  
mubmmbp.add(mubmmbgn,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=1;  
gbcmubmmb.gridy=4;  
mubmmbp.add(mubmmbgn1,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=0;  
gbcmubmmb.gridy=5;  
mubmmbp.add(mubmmbid,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=1;  
gbcmubmmb.gridy=5;  
mubmmbp.add(mubmmbid1,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=0;  
gbcmubmmb.gridy=6;  
mubmmbp.add(mubmmbmodify,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=1;  
gbcmubmmb.gridy=6;  
mubmmbp.add(mubmmbback,gbcmubmmb);
```

```
gbcmubmmb.insets=new Insets(10,10,10,10);  
gbcmubmmb.gridx=2;  
gbcmubmmb.gridy=6;  
mubmmbp.add(mubmmbso,gbcmubmmb);
```

```

        mubmmb.add(mubmmbp);
        JOptionPane.showMessageDialog(null,"PLEASE NOTE THAT THE CURRENCY
        BEING SHOWN IS INR AND GENRE AUTOBIOGRAPHY. PLEASE CHANGE IT AS PER
        REQUIREMENT");
    }
}

public void btnActionBookModify()//BUTTON ACTION FOR "MODIFY"
{
    temporary=1;
    mubmmbBN=mubmmbbn1.getText().trim();
    mubmmbAU=mubmmbau1.getText().trim();
    String strmubmmbPR=mubmmbpr1.getText().trim();
    mubmmbCU=mubmmbcu1.getSelectedItem().toString();
    mubmmbGN=mubmmbgn1.getSelectedItem().toString();
    mubmmbID=mubmmbid1.getText().trim();
    if(mubmmbBN.equals("") || mubmmbAU.equals("") || strmubmmbPR.equals("") ||
    mubmmbID.equals(""))
    {
        JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been left
        without data");
        temporary++;
    }
    try
    {
        mubmmbPR=Integer.parseInt(strmubmmbPR);
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"PLEASE ENTER A NUMERIC VALUE FOR
        PRICE");
        temporary++;
    }
    if(temporary==1 && mubmmbGN.equals("Combination of genres"))
    {
        temporary++;
        final JFrame genresmubmmb;
        genresmubmmb=new JFrame("PLEASE ENTER THE GENRES");
        mubmmb.setVisible(false);
        genresmubmmb.setVisible(true);
        genresmubmmb.setSize(500,300);
        genresmubmmb.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel genrestriallabel=new JLabel("GENRES");
        genrestrial=new JTextField(20);
        JButton ok=new JButton("OK");

        ok.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent ae)

```

```

        {
            mubmmbGN=genretrial.getText().trim();
            genresmubmmb.setVisible(false);
            temporary=1;
            ModifyTheBook();
        }
    });
    JButton back=new JButton ("BACK");
    back.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            temporary++;
            genresmubmmb.setVisible(false);
            mubmmb.setVisible(true);
        }
    });
    JButton so=new JButton ("SIGN OUT");
    so.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            temporary++;
            genresmubmmb.setVisible(false);
            main();
        }
    });

    JPanel genrespanel=new JPanel(new GridBagLayout());
    GridBagConstraints gbcgenres=new GridBagConstraints();

    gbcgenres.insets=new Insets(10,10,10,10);
    gbcgenres.gridx=0;
    gbcgenres.gridy=0;
    genrespanel.add(genretriallabel,gbcgenres);

    gbcgenres.insets=new Insets(10,10,10,10);
    gbcgenres.gridx=1;
    gbcgenres.gridy=0;
    genrespanel.add(genretrial,gbcgenres);

    gbcgenres.insets=new Insets(10,10,10,10);
    gbcgenres.gridx=0;
    gbcgenres.gridy=1;
    genrespanel.add(ok,gbcgenres);

    gbcgenres.insets=new Insets(10,10,10,10);
    gbcgenres.gridx=1;
    gbcgenres.gridy=1;

```

```

        genrespanel.add(back,gbcgenres);

        gbcgenres.insets=new Insets(10,10,10,10);
        gbcgenres.gridx=2;
        gbcgenres.gridy=1;
        genrespanel.add(so,gbcgenres);

        genresmubmmb.add(genrespanel);
    }

    if(temporary==1)
    {
        ModifyTheBook();
    }
}

public void ModifyTheBook()//MODIFYING IN THE DATABASE
{
    try
    {
        rsmubmmb.updateString("B_NAME",mubmmbBN);
        rsmubmmb.updateString("B_AUTHOR",mubmmbAU);
        rsmubmmb.updateInt("B_PRICE",mubmmbPR);
        rsmubmmb.updateString("B_CURRENCY",mubmmbCU);
        rsmubmmb.updateString("B_GENRE",mubmmbGN);
        rsmubmmb.updateString("B_ID",mubmmbID);
        rsmubmmb.updateRow();
        JOptionPane.showMessageDialog(null,"Record Updated");
        mubmmb.setVisible(false);
        LibMenuBookMaster();
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT THE ID REFERS TO THAT OF
AN EXISTING BOOK");
    }
}

public void LibMenumubmdb()//TAKING THE RAEDER ID WHICH IS TO BE DELETED
{
    mubmdbentryidtext=new JTextField(10);
    JLabel mubmdbentryidlabel=new JLabel("Please enter the Book id");
    JPanel mubmdbentryidpanel=new JPanel();
    mubmdbentryidpanel.add(mubmdbentryidlabel);
    mubmdbentryidpanel.add(mubmdbentryidtext);
    int
selectedoptionIDDeleteBook=JOptionPane.showConfirmDialog(null,mubmdbentryidpanel,"ID",J
OptionPane.OK_CANCEL_OPTION);

```

```

        if(selectedoptionIDDeleteBook==JOptionPane.OK_OPTION)
        {
            btnActionIDmubmdb();
        }
        else
        {
            LibMenuBookMaster();
        }
    }

    public void btnActionIDmubmdb()//DESIGNING OF PAGE FOR MODIFYING A BOOK
    {
        String ID=mubmdbentryidtext.getText().trim();
        temporary=0;
        mubmdbbn1=new JLabel();
        mubmdbau1=new JLabel();
        mubmdbpr1=new JLabel();
        mubmdbcu1=new JLabel();
        mubmdbgn1=new JLabel();
        mubmdbid1=new JLabel();
        try
        {
            String sqlmubmdb="select * from BOOK_MST where B_ID='"+ID+"' and
B_ACTIVE='Yes'";
            rsmubmdb=st.executeQuery(sqlmubmdb);
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception Before Button");
        }
        try
        {
            if(rsmubmdb.next())
            {
                String strBook_availabel=rsmubmdb.getString("B_AVAILABLE");
                if(strBook_availabel.equals("No"))
                {
                    temporary++;
                    JOptionPane.showMessageDialog(null,"SORRY BUT THE BOOK HAD BEEN
ISSUED AND YET NOT RETURNED. SO, IT CANNOT BE DELETED");
                    LibMenuBookMaster();
                }
            }
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
    }

```



```

if(temporary==0)
{
    try
    {
        if(rsmubmdb.first())
        {
            mubmdbbn1.setText(rsmubmdb.getString("B_NAME"));
            mubmdbau1.setText(rsmubmdb.getString("B_AUTHOR"));
            mubmdbpr1.setText(rsmubmdb.getString("B_PRICE"));
            mubmdbcu1.setText(rsmubmdb.getString("B_CURRENCY"));
            mubmdbgn1.setText(rsmubmdb.getString("B_GENRE"));
            mubmdbid1.setText(rsmubmdb.getString("B_ID"));
        }
        else
        {
            JOptionPane.showMessageDialog(null,"ID DOES NOT EXIST");
            temporary++;
            LibMenuBookMaster();
        }
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception Again");
    }
}
if(temporary==0)
{
    mubmdb=new JFrame("Delete Book");
    mubmdb.setVisible(true);
    mubmdb.setSize(700,500);
    mubmdb.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel mubmdbp=new JPanel(new GridBagLayout());
    JLabel mubmdbbn=new JLabel("B_NAME");
    JLabel mubmdbau=new JLabel("B_AUTHOR");
    JLabel mubmdbpr=new JLabel("B_PRICE");
    JLabel mubmdbcu=new JLabel("B_CURRENCY");
    JLabel mubmdbgn=new JLabel("B_GENRE");
    JLabel mubmdbid=new JLabel("B_ID");
    JLabel mubmdbremarklabel=new JLabel("Remark");
    dbremarktext=new JTextField(20);
    JButton mubmdbdelete=new JButton("Delete");
    mubmdbdelete.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e)
        {
            btnActionBookDelete();
        }
    });
}

```

```

        JButton mubmdbback=new JButton("Back");
mubmdbback.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        mubmdb.setVisible(false);
        LibMenuBookMaster();
    }
});
JButton mubmdbso=new JButton("Sign out");
mubmdbso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        mubmdb.setVisible(false);
        main();
    }
});
GridBagConstraints gbcmubmdb=new GridBagConstraints();
gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=0;
gbcmubmdb.gridy=0;
mubmdbp.add(mubmdbbn,gbcmubmdb);

gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=1;
gbcmubmdb.gridy=0;
mubmdbp.add(mubmdbbn1,gbcmubmdb);

gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=0;
gbcmubmdb.gridy=1;
mubmdbp.add(mubmdbau,gbcmubmdb);

gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=1;
gbcmubmdb.gridy=1;
mubmdbp.add(mubmdbau1,gbcmubmdb);

gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=0;
gbcmubmdb.gridy=2;
mubmdbp.add(mubmdbpr,gbcmubmdb);

gbcmubmdb.insets=new Insets(10,10,10,10);
gbcmubmdb.gridx=1;
gbcmubmdb.gridy=2;
mubmdbp.add(mubmdbpr1,gbcmubmdb);

```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=0;
gbcsubmdb.gridy=3;
submdbp.add(submdbcu,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=1;
gbcsubmdb.gridy=3;
submdbp.add(submdbcu1,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=0;
gbcsubmdb.gridy=4;
submdbp.add(submdbgn,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=1;
gbcsubmdb.gridy=4;
submdbp.add(submdbgn1,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=0;
gbcsubmdb.gridy=5;
submdbp.add(submdbid,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=1;
gbcsubmdb.gridy=5;
submdbp.add(submdbid1,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=0;
gbcsubmdb.gridy=6;
submdbp.add(submdbremarklabel,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=1;
gbcsubmdb.gridy=6;
submdbp.add(dbremarktext,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=0;
gbcsubmdb.gridy=7;
submdbp.add(submdbdelete,gbcsubmdb);
```

```
gbcsubmdb.insets=new Insets(10,10,10,10);
gbcsubmdb.gridx=1;
```

```

        gbcsubmdb.gridy=7;
        mubmdbp.add(mubmdbback,gbcsubmdb);

        gbcsubmdb.insets=new Insets(10,10,10,10);
        gbcsubmdb.gridx=2;
        gbcsubmdb.gridy=7;
        mubmdbp.add(mubmdbso,gbcsubmdb);

        mubmdb.add(mubmdbp);
    }
}
public void btnActionBookDelete()//TO DELETE THE DATA
{
    int temporary=0;
    String remarkdeletebook=dbremarktext.getText().trim();
    if(remarkdeletebook.equals(""))
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT A REMARK IS REQUIRED AS
TO WHY THE BOOK IS BEING DELETED");
        temporary++;
    }
    else
    {
        if(temporary==0)
        {
            try
            {
                rsmubmdb.updateString("B_ACTIVE","No");
                rsmubmdb.updateString("B_AVAILABLE","No");
                rsmubmdb.updateString("B_REMARK",remarkdeletebook);
                rsmubmdb.updateRow();
                JOptionPane.showMessageDialog(null,"Record Deleted");
                mubmdb.setVisible(false);
                LibMenuBookMaster();
            }
            catch(Exception ex)
            {
                JOptionPane.showMessageDialog(null,"Exception while deleting");
            }
        }
    }
}

public void date()//TO OBTAIN TODAY'S AND & DAYS LATER'S DATE
{
    Calendar dt1=Calendar.getInstance();
    Calendar dt2=Calendar.getInstance();
    dt1.setTime(new java.util.Date());

```

```

dt1.add(Calendar.DATE,7);
dt2.setTime(new java.util.Date());
today=(sdf.format(dt2.getTime())).toString();
duedate=(sdf.format(dt1.getTime())).toString();
LibMenuTransactions();
}

public void LibMenuTransactions()//DESIGNING THE PAGE FOR TRANSACTION MENU
CHOICES
{
    LibMenuTM=new JFrame("Transactions");
    LibMenuTM.setVisible(true);
    LibMenuTM.setSize(700,500);
    LibMenuTM.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JPanel paneltm=new JPanel(new GridBagLayout());

    JButton tm1=new JButton (" Book Issue ");
    JButton tm2=new JButton (" Book Return ");
    JButton tm3=new JButton (" Back ");
    JButton tm4=new JButton (" Sign out ");

    GridBagConstraints gbcLibMenutm=new GridBagConstraints();

    tm1.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent aemu)
        {
            LibMenuTM.setVisible(false);
            LibMenutmbr();
        }
    });
    gbcLibMenutm.insets=new Insets(10,10,10,10);
    gbcLibMenutm.gridx=0;
    gbcLibMenutm.gridy=0;
    paneltm.add(tm1,gbcLibMenutm);

    tm2.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuTM.setVisible(false);
            LibMenutmbr();
        }
    });
    gbcLibMenutm.insets=new Insets(10,10,10,10);
    gbcLibMenutm.gridx=0;
    gbcLibMenutm.gridy=1;
    paneltm.add(tm2,gbcLibMenutm);

```

```

tm3.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuTM.setVisible(false);
        LibMenu();
    }
});
gbcLibMenutm.insets=new Insets(10,10,10,10);
gbcLibMenutm.gridx=0;
gbcLibMenutm.gridy=3;
paneltm.add(tm3,gbcLibMenutm);

tm4.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuTM.setVisible(false);
        main();
    }
});
gbcLibMenutm.insets=new Insets(10,10,10,10);
gbcLibMenutm.gridx=0;
gbcLibMenutm.gridy=4;
paneltm.add(tm4,gbcLibMenutm);

LibMenuTM.add(paneltm);
}

public void LibMenutmbi()//DESIGNING THE PAGE FOR BOOK ISSUE
{
    JOptionPane.showMessageDialog(null,"Please enter the date in the DD-MM-YYYY
format");
    tmbi=new JFrame("Book Issue");
    tmbi.setVisible(true);
    tmbi.setSize(700,500);
    tmbi.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel tmbip=new JPanel(new GridBagLayout());
    JLabel tmbirid=new JLabel("Reader ID");
    JLabel tmbibid=new JLabel("Book ID");
    JLabel tmbiid=new JLabel("Issue Date");
    JLabel tmbiprd=new JLabel("Due Date");
    JLabel tmbiuid=new JLabel("ID");
    tmbirid1=new JTextField(20);
    tmbibid1=new JTextField(20);
    tmbiid1=new JTextField(20);
    tmbiprd1=new JTextField(20);
    JLabel tmbiuid1=new JLabel(user);
    JButton tmbiissue=new JButton("Issue");
    tmbiissue.addActionListener(new ActionListener(){

```

```

        public void actionPerformed(ActionEvent e)
        {
            btnActiontmbi();
        }
    });
    JButton tmbiback=new JButton("Back");
    tmbiback.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            tmbi.setVisible(false);
            LibMenuTransactions();
        }
    });
    JButton tmbiso=new JButton("Sign out");
    tmbiso.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            tmbi.setVisible(false);
            main();
        }
    });
    GridBagConstraints tmbigbc=new GridBagConstraints();
    tmbirid1.setText("");
    tmbibid1.setText("");
    tmbiid1.setText(today);
    tmbiprd1.setText(duedate);

    tmbigbc.insets=new Insets(10,10,10,10);
    tmbigbc.gridx=0;
    tmbigbc.gridy=0;
    tmbip.add(tmbirid,tmbigbc);

    tmbigbc.insets=new Insets(10,10,10,10);
    tmbigbc.gridx=1;
    tmbigbc.gridy=0;
    tmbip.add(tmbirid1,tmbigbc);

    tmbigbc.insets=new Insets(10,10,10,10);
    tmbigbc.gridx=0;
    tmbigbc.gridy=1;
    tmbip.add(tmbibid,tmbigbc);

    tmbigbc.insets=new Insets(10,10,10,10);
    tmbigbc.gridx=1;
    tmbigbc.gridy=1;
    tmbip.add(tmbibid1,tmbigbc);

```

```

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=0;
tmbigbc.gridy=2;
tmbip.add(tmbiid,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=1;
tmbigbc.gridy=2;
tmbip.add(tmbiid1,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=0;
tmbigbc.gridy=3;
tmbip.add(tmbiprd,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=1;
tmbigbc.gridy=3;
tmbip.add(tmbiprd1,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=0;
tmbigbc.gridy=4;
tmbip.add(tmbiuid,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=1;
tmbigbc.gridy=4;
tmbip.add(tmbiuid1,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=0;
tmbigbc.gridy=5;
tmbip.add(tmbiissue,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=1;
tmbigbc.gridy=5;
tmbip.add(tmbiback,tmbigbc);

tmbigbc.insets=new Insets(10,10,10,10);
tmbigbc.gridx=2;
tmbigbc.gridy=5;
tmbip.add(tmbiso,tmbigbc);

tmbi.add(tmbip);
}

```



```

public void btnActiontmbi()//BUTTON ACTION FOR "ISSUE"
{
    int proceed=0;
    int temporary=1;
    readeridtmbi=tmbirid1.getText().trim();
    bookidtmbi=tmbibid1.getText().trim();
    String bookissuetmbi=tmbiid1.getText().trim();
    String bookplanneddatetmbi=tmbiprd1.getText().trim();
    try
    {
        java.util.Date dttrial1=sdf.parse(bookissuetmbi);
        java.util.Date dttrial2=sdf.parse(bookplanneddatetmbi);
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Sorry,but please enter the date in the DD-MM-
        YYYY format");
        proceed++;
    }
    if(proceed==0)
    {
        String sqltmbitrnmenu="select * from BOOK_TRN";
        String sqltmbibkmenu="select * from BOOK_MST where B_ID='"+bookidtmbi+"'";
        String sqltmbirdmenu="select * from READER_MST where R_ID='"+readeridtmbi+"'";
        try
        {
            if(readeridtmbi.equals("") || bookidtmbi.equals("") || bookissuetmbi.equals("") ||
            bookplanneddatetmbi.equals(""))
            {
                JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been
                left without data");
                temporary++;
            }
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while blank textfield check");
        }
        try
        {
            if(temporary==1)
            {
                rstmbirdmenu=st.executeQuery(sqltmbirdmenu);
                if(rstmbirdmenu.next())
                {
                    String strReaderIdTrnMenu = rstmbirdmenu.getString("RID_COUNT");
                    ReaderIdTrnMenu=Integer.parseInt(strReaderIdTrnMenu);
                }
            }
        }
    }
}

```

```

        String readeractive=rstmbirdmenu.getString("R_ACTIVE");
        if(readeractive=="No")
        {
            JOptionPane.showMessageDialog(null,"Sorry but the Reader has been deleted");
            temporary++;
        }
    }
    else
    {
        JOptionPane.showMessageDialog(null,"Sorry but the Reader does not exist");
        temporary++;
    }
    rstmbirdmenu.close();
}
}
catch(Exception ex)
{
    ex.printStackTrace();
    JOptionPane.showMessageDialog(null,"Exception while user availability check");
}
try
{
    if(temporary==1)
    {
        rstmbibkmenu=st.executeQuery(sqltmbibkmenu);
        if(rstmbibkmenu.next())
        {
            String bookactive=rstmbibkmenu.getString("B_ACTIVE");
            if(bookactive=="No")
            {
                JOptionPane.showMessageDialog(null,"Sorry but the Book has been deleted");
                temporary++;
            }
            String bookavailable=rstmbibkmenu.getString("B_AVAILABLE");
            if(bookavailable.equals("No"))
            {
                JOptionPane.showMessageDialog(null,"Sorry but the Book has been issued");
                temporary++;
            }
            rstmbibkmenu.close();
        }
    }
    else
    {
        JOptionPane.showMessageDialog(null,"Sorry but the Book does not exist");
        temporary++;
    }
}
}

```

```

    }
    catch(Exception ex)
    {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null,"Exception while book active and available
check");
    }
    try
    {
        if(temporary==1)
        {
            int selected_option_trn_menu=0;
            if(ReaderIdTrnMenu!=0)
            {
                selected_option_trn_menu=JOptionPane.showConfirmDialog(null,"THIS USER
ALREADY HAS "+ReaderIdTrnMenu+" BOOKS ISSUED EXCLUDING THE ONE TAKEN
NOW. DO YOU STILL WANTTO PROCEED?","PLEASE
CONFIRM",JOptionPane.YES_NO_OPTION);
            }
            if(selected_option_trn_menu==JOptionPane.YES_OPTION)
            {
                rstmbitrnmenu=st.executeQuery(sqltmbitrnmenu);
                rstmbitrnmenu.moveToInsertRow();
                rstmbitrnmenu.updateString("R_ID",readeridtmmbi);
                rstmbitrnmenu.updateString("B_ID",bookidtmmbi);
                rstmbitrnmenu.updateString("B_ISSUE_DATE",bookissuetmmbi);

rstmbitrnmenu.updateString("B_PLANNED_RETURN_DATE",bookplanneddatetmmbi);
                rstmbitrnmenu.updateString("U_LOGIN",user);
                rstmbitrnmenu.insertRow();
                rstmbitrnmenu.close();
                st.close();

st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
ABLE);

                JOptionPane.showMessageDialog(null,"Book Issued");
            }
            else
            {
                temporary++;
            }
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(null,"Exception while issuing");
    }
    if(temporary==1)

```

```

        {
            tmbi.setVisible(false);
            RequiredChanges();
        }
    }
}

public void RequiredChanges()//CHANGES TO BE MADE IN THE READER AND BOOK
MASTER TABLE
{
    try
    {
        String sqltmbirdmenuchanges="select * from READER_MST where
R_ID='"+readeridtmbr+"";
        ResultSet rstmbirdmenuchanges=st.executeQuery(sqltmbirdmenuchanges);
        ReaderIdTrnMenu++;
        String strIncReaderIdTrnMenu=String.valueOf(ReaderIdTrnMenu);
        if(rstmbirdmenuchanges.next())
        {
            rstmbirdmenuchanges.updateString("RID_COUNT",strIncReaderIdTrnMenu);
            rstmbirdmenuchanges.updateRow();
            rstmbirdmenuchanges.close();
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    try
    {
        String sqltmbibkmenuchanges="select * from BOOK_MST where
B_ID='"+bookidtmbr+"";
        ResultSet rstmbibkmenuchanges=st.executeQuery(sqltmbibkmenuchanges);
        if(rstmbibkmenuchanges.next())
        {
            rstmbibkmenuchanges.updateString("B_AVAILABLE","No");
            rstmbibkmenuchanges.updateRow();
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    LibMenuTransactions();
}

public void LibMenuTmbr()//DESIGNING THE PAGE FOR BOOK RETURN
{

```

```

        JTextField bi_input=new JTextField(10);
        JPanel biinputpanel=new JPanel();
        biinputpanel.add(bi_input);
        int
        selected_option_book_return=JOptionPane.showConfirmDialog(null,biinputpanel,"PLEASE
        ENTER THE BOOK ID",JOptionPane.OK_CANCEL_OPTION);
        int temp_b_return;
        if(selected_option_book_return==JOptionPane.OK_OPTION)
        {
            temp_b_return=0;
        }
        else
        {
            LibMenuTransactions();
            temp_b_return=1;
        }
        if(temp_b_return==0)
        {
            String B_ID_INPUT=bi_input.getText().trim();
            if(B_ID_INPUT.equals(""))
            {
                JOptionPane.showMessageDialog(null,"Please enter something");
                LibMenutmbr();
            }
            try
            {
                sql_book_return_trnmenu="select * from BOOK_TRN where
                B_ID='"+B_ID_INPUT+"'";
                rs_book_return_trnmenu=st.executeQuery(sql_book_return_trnmenu);
            }
            catch(Exception ex)
            {
                JOptionPane.showMessageDialog(null,"Exception while connection");
                ex.printStackTrace();
            }
            try
            {
                if(rs_book_return_trnmenu.next())
                {
                    r_id=rs_book_return_trnmenu.getString("R_ID");
                    b_id=rs_book_return_trnmenu.getString("B_ID");
                    b_issue_date=rs_book_return_trnmenu.getString("B_ISSUE_DATE");

                    b_planned_return_date=rs_book_return_trnmenu.getString("B_PLANNED_RETURN_DATE");
                    u_login=rs_book_return_trnmenu.getString("U_LOGIN");
                }
                else
                {

```

```

        JOptionPane.showMessageDialog(null,"Sorry but the book was not with any reader");
        temp_b_return++;
        LibMenutmbr();
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while extracting");
    ex.printStackTrace();
}
if(temp_b_return==0)
{
    Calendar dt3=Calendar.getInstance();
    today=(sdf.format(dt3.getTime())).toString();
    b_return_frame=new JFrame("Book Return");
    b_return_frame.setSize(700,500);
    b_return_frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    b_return_frame.setVisible(true);
    JLabel r_id_label=new JLabel("Reader Id");
    JLabel r_id_data=new JLabel(r_id);
    JLabel b_id_label=new JLabel("Book Id");
    JLabel b_id_data=new JLabel(b_id);
    JLabel b_issue_date_label=new JLabel("Book Issue Date");
    JLabel b_issue_date_data=new JLabel(b_issue_date);
    JLabel b_planned_return_date_label=new JLabel("Book Due Date");
    JLabel b_planned_return_date_data=new JLabel(b_planned_return_date);
    JLabel u_login_label=new JLabel("Issue User");
    JLabel u_login_data=new JLabel(u_login);
    JLabel b_actual_return_date_label=new JLabel("Book Return Date");
    b_actual_return_date_data=new JTextField(10);
    b_actual_return_date_data.setText(today);
    JButton return_b_rtn=new JButton("Return");
    return_b_rtn.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            btnAction_Book_Return();
        }
    });
    JButton back_b_rtn=new JButton("Back");
    back_b_rtn.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            b_return_frame.setVisible(false);
            LibMenuTransactions();
        }
    });
    JButton so_b_rtn=new JButton("Sign Out");

```

```

so_b_rtn.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        b_return_frame.setVisible(false);
        main();
    }
});
GridBagConstraints gbc_book_return=new GridBagConstraints();
JPanel Book_return_panel=new JPanel(new GridBagLayout());

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=0;
gbc_book_return.gridy=0;
Book_return_panel.add(r_id_label,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=1;
gbc_book_return.gridy=0;
Book_return_panel.add(r_id_data,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=0;
gbc_book_return.gridy=1;
Book_return_panel.add(b_id_label,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=1;
gbc_book_return.gridy=1;
Book_return_panel.add(b_id_data,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=0;
gbc_book_return.gridy=2;
Book_return_panel.add(b_issue_date_label,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=1;
gbc_book_return.gridy=2;
Book_return_panel.add(b_issue_date_data,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=0;
gbc_book_return.gridy=3;
Book_return_panel.add(b_planned_return_date_label,gbc_book_return);

gbc_book_return.insets=new Insets(10,10,10,10);
gbc_book_return.gridx=1;

```

```

        gbc_book_return.gridy=3;
        Book_return_panel.add(b_planned_return_date_data,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=0;
        gbc_book_return.gridy=4;
        Book_return_panel.add(u_login_label,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=1;
        gbc_book_return.gridy=4;
        Book_return_panel.add(u_login_data,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=0;
        gbc_book_return.gridy=5;
        Book_return_panel.add(b_actual_return_date_label,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=1;
        gbc_book_return.gridy=5;
        Book_return_panel.add(b_actual_return_date_data,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=0;
        gbc_book_return.gridy=6;
        Book_return_panel.add(return_b_rtn,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=1;
        gbc_book_return.gridy=6;
        Book_return_panel.add(back_b_rtn,gbc_book_return);

        gbc_book_return.insets=new Insets(10,10,10,10);
        gbc_book_return.gridx=2;
        gbc_book_return.gridy=6;
        Book_return_panel.add(so_b_rtn,gbc_book_return);

        b_return_frame.add(Book_return_panel);
    }
}

public void btnAction_Book_Return();//BUTTON ACTION FOR "RETURN"
{
    String actualreturndate=b_actual_return_date_data.getText().trim();
    int proceed=0;

```



```

        try
        {
            java.util.Date dttrial3=sdf.parse(actualreturndate);
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Sorry,but please enter the date in the DD-MM-
            YYYY format");
            b_return_frame.setVisible(true);
            proceed++;
        }
        if(proceed==0)
        {
            try
            {
                String sql_book_return_rtnmenu="select * from B_RTN";
                rs_Book_return_rtnmenu=st.executeQuery(sql_book_return_rtnmenu);
                rs_Book_return_rtnmenu.moveToInsertRow();
                rs_Book_return_rtnmenu.updateString("R_ID",r_id);
                rs_Book_return_rtnmenu.updateString("B_ID",b_id);
                rs_Book_return_rtnmenu.updateString("B_ISSUE_DATE",b_issue_date);

rs_Book_return_rtnmenu.updateString("B_PLANNED_RETURN_DATE",b_planned_return_date
);

                rs_Book_return_rtnmenu.updateString("U_LOGIN",u_login);

rs_Book_return_rtnmenu.updateString("B_ACTUAL_RETURN_DATE",actualreturndate);
                rs_Book_return_rtnmenu.updateString("U_LOGIN_RECEIVER",user);
                rs_Book_return_rtnmenu.insertRow();
                rs_Book_return_rtnmenu.close();
                st.close();

st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
ABLE);
                JOptionPane.showMessageDialog(null,"Book Returned");
            }
            catch(Exception ex)
            {
                JOptionPane.showMessageDialog(null,"Exception while adding");
                ex.printStackTrace();
            }

            try
            {
                rs_book_return_trnmenu=st.executeQuery(sql_book_return_trnmenu);
                if(rs_book_return_trnmenu.next())
                {
                    rs_book_return_trnmenu.deleteRow();
                }
            }

```

```

        rs_book_return_trnmenu.close();
        st.close();

    st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
    ABLE);
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while deleting");
        ex.printStackTrace();
    }
    ResultSet rs1,rs2;
    String sql1="select * from BOOK_MST where B_ID='"+b_id+"'";
    String sql2="select * from READER_MST where R_ID='"+r_id+"'";
    try
    {
        rs2=st.executeQuery(sql2);
        if(rs2.next())
        {
            int num=Integer.parseInt(rs2.getString("RID_COUNT"));
            num--;
            String str_num=String.valueOf(num);
            rs2.updateString("RID_COUNT",str_num);
            rs2.updateRow();
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    try
    {
        rs1=st.executeQuery(sql1);
        if(rs1.next())
        {
            rs1.updateString("B_AVAILABLE","Yes");
            rs1.updateRow();
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}

public void LibMenu()//DESIGNING THE PAGE FOR MAINTAINANCE MENU
{

```

```

try
{
    String sqlImmm="select * from USER_MST where U_LOGIN='"+user+"'";
    rslmmm=st.executeQuery(sqlImmm);
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while connecting");
    LibMenuMM();
}
LibMenuMM=new JFrame("Change Password For Self");
LibMenuMM.setVisible(true);
LibMenuMM.setSize(700,500);
LibMenuMM.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JPanel LibMenuMMp=new JPanel(new GridBagLayout());
JLabel LIImmm=new JLabel("Login ID");
JLabel UNImmm=new JLabel("User Name");
JLabel PWoldImmm=new JLabel(" Old Password");
JLabel PWnewImmm=new JLabel(" New Password");
JLabel PWconfirmImmm=new JLabel(" Confirm Password");
JLabel Clmmmm=new JLabel("Category");

pwoldImmm=new JPasswordField(20);
pwnewImmm=new JPasswordField(20);
pwconfirmImmm=new JPasswordField(20);
JButton Immmchangepassword=new JButton("Change");
Immmchangepassword.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e)
    {
        btnActionLMMMCHANGEPASSWORD();
    }
});
JButton Immmmb=new JButton("Back");
Immmmb.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuMM.setVisible(false);
        LibMenu();
    }
});
JButton Immmso=new JButton("Sign out");
Immmso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        LibMenuMM.setVisible(false);
        main();
    }
}

```

```

});
GridBagConstraints gbcImmm=new GridBagConstraints();
pwoldImmm.setText("");
pwnewImmm.setText("");
pwconfirmImmm.setText("");
String strlImmm="",strunImmm="",strcalImmm="";
try
{
    if(rslmmmm.next())
    {
        strlImmm=rslmmmm.getString("U_LOGIN");
        strunImmm=rslmmmm.getString("U_NAME");
        strcalImmm=rslmmmm.getString("U_CATEGORY");
        strpassImmm=rslmmmm.getString("U_PASSWORD");
    }
    else
    {
        LibMenuMM.setVisible(false);
        JOptionPane.showMessageDialog(null,"SORRY BUT THE USER ID DOES NOT
EXIST");
        LibMenu();
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while allotting the text");
}
JLabel lilmmmm=new JLabel(strlImmm);
JLabel unlmmmm=new JLabel(strunImmm);
JLabel calmmmm=new JLabel(strcalImmm);

gbcImmm.insets=new Insets(10,10,10,10);
gbcImmm.gridx=0;
gbcImmm.gridy=0;
LibMenuMMp.add(LlImmm,gbcImmm);

gbcImmm.insets=new Insets(10,10,10,10);
gbcImmm.gridx=1;
gbcImmm.gridy=0;
LibMenuMMp.add(lilmmmm,gbcImmm);

gbcImmm.insets=new Insets(10,10,10,10);
gbcImmm.gridx=0;
gbcImmm.gridy=1;
LibMenuMMp.add(Unlmmmm,gbcImmm);

gbcImmm.insets=new Insets(10,10,10,10);
gbcImmm.gridx=1;

```

```

gbclmmm.gridy=1;
LibMenuMMp.add(unlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=0;
gbclmmm.gridy=2;
LibMenuMMp.add(PWoldlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=1;
gbclmmm.gridy=2;
LibMenuMMp.add(pwoldlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=0;
gbclmmm.gridy=3;
LibMenuMMp.add(PWnewlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=1;
gbclmmm.gridy=3;
LibMenuMMp.add(pwnewlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=0;
gbclmmm.gridy=4;
LibMenuMMp.add(PWconfrmlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=1;
gbclmmm.gridy=4;
LibMenuMMp.add(pwconfrmlmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=0;
gbclmmm.gridy=5;
LibMenuMMp.add(Clmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=1;
gbclmmm.gridy=5;
LibMenuMMp.add(calmmm,gbclmmm);

gbclmmm.insets=new Insets(10,10,10,10);
gbclmmm.gridx=0;
gbclmmm.gridy=6;
LibMenuMMp.add(lmmmchangepassword,gbclmmm);

```

```

        gbcLmmm.insets=new Insets(10,10,10,10);
        gbcLmmm.gridx=1;
        gbcLmmm.gridy=6;
        LibMenuMMp.add(lmmmb,gbcLmmm);

        gbcLmmm.insets=new Insets(10,10,10,10);
        gbcLmmm.gridx=2;
        gbcLmmm.gridy=6;
        LibMenuMMp.add(lmmms0,gbcLmmm);

        LibMenuMM.add(LibMenuMMp);
    }

    public void btnActionLMMMCHANGEPASSWORD()//BUTTON ACTION FOR "CHANGE"
    {
        String newchangeasslibrarianmenu=pwnewlmmm.getText().trim();
        String confirmchangeasslibrarianmenu=pwconfrmlmmm.getText().trim();
        String oldchangeasslibrarianmenu=pwoldlmmm.getText().trim();
        if(oldchangeasslibrarianmenu.equals(strpasslmmm))
        {
            if(newchangeasslibrarianmenu.equals(confirmchangeasslibrarianmenu))
            {
                try
                {
                    rslmmm.updateString("U_PASSWORD",newchangeasslibrarianmenu);
                    rslmmm.updateRow();
                    JOptionPane.showMessageDialog(null,"Password Changed");
                    LibMenuMM.setVisible(false);
                    LibMenu();
                }
                catch(Exception ex)
                {
                    JOptionPane.showMessageDialog(null,"Exception while changing");
                    ex.printStackTrace();
                }
            }
            else
            {
                JOptionPane.showMessageDialog(null,"SORRY BUT YOUR NEW PASSWORD AND
CONFIRMED PASSWORD ARE NOT THE SAME");
            }
        }
        else
        {
            JOptionPane.showMessageDialog(null,"SORRY BUT OLD NEW PASSWORD AND
YOUR ACTUAL PASSWORD ARE NOT THE SAME");
        }
    }

```

```

    }
}

public void LibMenuReportMenu()//DESIGNING THE PAGE FOR CHOICES FOR REPORT
MENU FOR LIBRARIAN
{
    report_menu_choices=new JFrame("Report Menu");
    report_menu_choices.setVisible(true);
    report_menu_choices.setSize(700,500);
    report_menu_choices.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel report_menu_choices_panel=new JPanel(new GridBagLayout());
    GridBagConstraints gbc_report_menu_choices=new GridBagConstraints();
    JButton book_search,reader_search,due_report,lending_report,back,sign_out;
    book_search=new JButton    (" Book Search ");
    reader_search=new JButton   ("Reader Search ");
    due_report=new JButton      (" Due Report  ");
    lending_report=new JButton   ("Lending Report");
    back=new JButton            ("    Back    ");
    sign_out=new JButton        (" Sign out  ");
    book_search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_choices.setVisible(false);
            LibMenuReportMenuBookSearch();
        }
    });
    reader_search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_choices.setVisible(false);
            LibMenuReportMenuReaderSearch();
        }
    });
    due_report.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuReportMenuDueReport();
        }
    });
    lending_report.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            LibMenuReportMenuLendingReport();
        }
    });
    back.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {

```

```

        report_menu_choices.setVisible(false);
        LibMenu();
    }
});
sign_out.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        report_menu_choices.setVisible(false);
        main();
    }
});
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=0;
report_menu_choices_panel.add(book_search,gbc_report_menu_choices);
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=1;
report_menu_choices_panel.add(reader_search,gbc_report_menu_choices);
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=2;
report_menu_choices_panel.add(due_report,gbc_report_menu_choices);
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=3;
report_menu_choices_panel.add(lending_report,gbc_report_menu_choices);
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=4;
report_menu_choices_panel.add(back,gbc_report_menu_choices);
gbc_report_menu_choices.insets=new Insets(10,10,10,10);
gbc_report_menu_choices.gridx=0;
gbc_report_menu_choices.gridy=5;
report_menu_choices_panel.add(sign_out,gbc_report_menu_choices);
report_menu_choices.add(report_menu_choices_panel);
}

public void LibMenuReportMenuBookSearch()//DESIGNING THE PAGE FOR BOOK
SEARCH
{
    report_menu_book_search=new JFrame("Book Search");
    report_menu_book_search.setVisible(true);
    report_menu_book_search.setSize(700,500);
    report_menu_book_search.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel report_menu_book_search_panel=new JPanel(new GridBagLayout());
    GridBagConstraints gbc_report_book_search=new GridBagConstraints();
    JButton search,back,sign_out;

```



```

search=new JButton("Search");
back=new JButton("Back");
sign_out=new JButton("Sign out");

JLabel book_name=new JLabel("Book Name");
JLabel author=new JLabel("Author");
JLabel genre=new JLabel("Genre");
String GENRE_RM[]={ "
","Autobiography","Fiction","Fantasy","Suspense","Romance","Non-
fiction","Mythology","Philosophy","Pshycology","Reference Books","Dictionaries and
Thesaurus","Others","Combination of genres"};

book_name_tf=new JTextField(20);
author_tf=new JTextField(10);
genre_tf=new JComboBox(GENRE_RM);

search.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        ReportMenuBookSearchOutput();
    }
});
back.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        report_menu_book_search.setVisible(false);
        LibMenuReportMenu();
    }
});
sign_out.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        report_menu_book_search.setVisible(false);
        main();
    }
});

gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(book_name,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(author,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;

```

```

gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(genre,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(book_name_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(author_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(genre_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(search,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(back,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(sign_out,gbc_report_book_search);
report_menu_book_search.add(report_menu_book_search_panel);
}

```

```

public void LibMenuReportMenuBookSearchOutput()//PRINTING THE REPORT FOR BOOK SEARCH
{

```

```

    String bn=book_name_tf.getText().trim();
    String au=author_tf.getText().trim();
    String gn=genre_tf.getSelectedItem().toString().trim();
    int temp=0;
    String sql="";
    if(gn.equals("Combination of genres"))
    {
        JTextField genre_actual=new JTextField(20);
        JPanel genre_actual_panel=new JPanel();
        genre_actual_panel.add(genre_actual);
        int so_genre_actual=JOptionPane.showConfirmDialog(null,genre_actual_panel,"Enter the genres",JOptionPane.OK_CANCEL_OPTION);
        if(so_genre_actual==JOptionPane.OK_OPTION)
        {
            gn=genre_actual.getText().trim();
        }
    }
}

```

```

else
{
    temp++;
}
}
if(temp==0)
{
    if(bn.equals("") && au.equals("") && gn.equals(""))
    {
        sql="select * from BOOK_MST where B_ACTIVE='Yes';"
    }
    else if(gn.equals(""))
    {
        if(au.equals(""))
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_ACTIVE='Yes';"
        }
        else if(bn.equals(""))
        {
            sql="select * from BOOK_MST where B_AUTHOR='"+au+"' and
B_ACTIVE='Yes';"
        }
        else
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and
B_AUTHOR='"+au+"' and B_ACTIVE='Yes';"
        }
    }
    else if(au.equals(""))
    {
        if(gn.equals(""))
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_ACTIVE='Yes';"
        }
        else if(bn.equals(""))
        {
            sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_ACTIVE='Yes';"
        }
        else
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_GENRE='"+gn+"'
and B_ACTIVE='Yes';"
        }
    }
    else if(bn.equals(""))
    {
        if(au.equals(""))
        {

```

```

        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_ACTIVE='Yes';
    }
    else if(gn.equals(""))
    {
        sql="select * from BOOK_MST where B_AUTHOR='"+au+"' and
B_ACTIVE='Yes';
    }
    else
    {
        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and
B_AUTHOR='"+au+"' and B_ACTIVE='Yes';
    }
    }
    else
    {
        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_AUTHOR='"+au+"'
and B_NAME='"+bn+"' and B_ACTIVE='Yes';
    }
    }
    if(temp==0)
    {
        try
        {
            rs_bs=st.executeQuery(sql);
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
    }
    if(temp==0)
    {
        System.out.println("ID \t NAME                                AUTHOR
PRICE          CURRENCY    GENRE                                AVAILABLE
ACTIVE");
    }
    if(temp==0)
    {
        try
        {
            if(rs_bs.next())
            {
                rs_bs.previous();
                while(rs_bs.next())
                {
                    String rs_id=rs_bs.getString("B_ID");
                    String rs_bn=rs_bs.getString("B_NAME");
                    String rs_au=rs_bs.getString("B_AUTHOR");

```

```

String rs_pr=rs_bs.getString("B_PRICE");
String rs_cu=rs_bs.getString("B_CURRENCY");
String rs_gn=rs_bs.getString("B_GENRE");
String rs_av=rs_bs.getString("B_AVAILABLE");
String rs_ac=rs_bs.getString("B_ACTIVE");
int len_rs_bn=rs_bn.length();
int len_rs_au=rs_au.length();
int len_rs_pr=rs_pr.length();
int len_rs_gn=rs_gn.length();
System.out.print(rs_id+"\t"+rs_bn);
for(int a=1;a<=(53-len_rs_bn);a++)
{
    System.out.print(" ");
}
System.out.print(rs_au);
for(int a=1;a<=(56-len_rs_au);a++)
{
    System.out.print(" ");
}
System.out.print(rs_pr);
for(int a=1;a<=(25-len_rs_pr);a++)
{
    System.out.print(" ");
}
System.out.print(rs_cu+"      "+rs_gn);
for(int a=1;a<=(51-len_rs_gn);a++)
{
    System.out.print(" ");
}
System.out.print(rs_av+"      "+rs_ac);
System.out.println();
}
rs_bs.close();
}
else
{
    JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE CHECK THE
ENTRIES AS THE DO NOT MATCH TO ANY RECORD IN THE DATABASE");
    temp++;
}
}
catch(Exception ex)
{
    ex.printStackTrace();
}
System.out.println();
System.out.println();
System.out.println();

```

```

        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
    }
}

public void LibMenuReportMenuReaderSearch()//DESIGNING THE PAGE FOR READER
SEARCH
{
    report_menu_reader_search=new JFrame("Reader Search");
    report_menu_reader_search.setVisible(true);
    report_menu_reader_search.setSize(700,500);
    report_menu_reader_search.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel report_menu_reader_search_panel=new JPanel(new GridBagLayout());
    GridBagConstraints gbc_report_reader_search=new GridBagConstraints();
    JButton search,back,sign_out;

    search=new JButton("Search");
    back=new JButton("Back");
    sign_out=new JButton("Sign out");

    JLabel first_name=new JLabel("First Name");
    JLabel last_name=new JLabel("Last Name");
    JLabel standard=new JLabel("Standard");

    first_name_tf=new JTextField(20);
    last_name_tf=new JTextField(20);
    standard_tf=new JTextField(10);

    search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            ReportMenuReaderSearchOutput();
        }
    });
    back.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_reader_search.setVisible(false);
            LibMenuReportMenu();
        }
    });
    sign_out.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)

```

```

        {
            report_menu_reader_search.setVisible(false);
            main();
        }
    });

    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=0;
    gbc_report_reader_search.gridy=0;
    report_menu_reader_search_panel.add(first_name,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=1;
    gbc_report_reader_search.gridy=0;
    report_menu_reader_search_panel.add(last_name,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=2;
    gbc_report_reader_search.gridy=0;
    report_menu_reader_search_panel.add(standard,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=0;
    gbc_report_reader_search.gridy=1;
    report_menu_reader_search_panel.add(first_name_tf,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=1;
    gbc_report_reader_search.gridy=1;
    report_menu_reader_search_panel.add(last_name_tf,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=2;
    gbc_report_reader_search.gridy=1;
    report_menu_reader_search_panel.add(standard_tf,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=0;
    gbc_report_reader_search.gridy=2;
    report_menu_reader_search_panel.add(search,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=1;
    gbc_report_reader_search.gridy=2;
    report_menu_reader_search_panel.add(back,gbc_report_reader_search);
    gbc_report_reader_search.insets=new Insets(10,10,10,10);
    gbc_report_reader_search.gridx=2;
    gbc_report_reader_search.gridy=2;
    report_menu_reader_search_panel.add(sign_out,gbc_report_reader_search);
    report_menu_reader_search.add(report_menu_reader_search_panel);
}

public void LibMenuReportMenuReaderSearchOutput()//PRINTING THE READER SEARCH
REPORT
{

```

```

String fn=first_name_tf.getText().trim();
String ln=last_name_tf.getText().trim();
String sta=standard_tf.getText().trim();
String sql;
if(fn.equals("") && ln.equals("") && sta.equals(""))
{
    sql="select * from READER_MST where R_ACTIVE='Yes'";
}
else if(fn.equals(""))
{
    if(ln.equals(""))
    {
        sql="select * from READER_MST where R_STANDARD='"+sta+"' and
R_ACTIVE='Yes'";
    }
    else if(sta.equals(""))
    {
        sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_STANDARD='"+sta+"' and R_ACTIVE='Yes'";
    }
}
else if(ln.equals(""))
{
    if(fn.equals(""))
    {
        sql="select * from READER_MST where R_STANDARD='"+sta+"' and
R_ACTIVE='Yes'";
    }
    else if(sta.equals(""))
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_STANDARD='"+sta+"' and R_ACTIVE='Yes'";
    }
}
else if(sta.equals(""))
{
    if(fn.equals(""))
    {

```



```

        sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_ACTIVE='Yes'";
    }
    else if(ln.equals(""))
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_LNAME='"+ln+"' and R_ACTIVE='Yes'";
    }
}
else
{
    sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_STANDARD='"+sta+"' and R_FNAME='"+fn+"' and R_ACTIVE='Yes'";
}
int temp=0;
try
{
    rs_rs=st.executeQuery(sql);
}
catch(Exception ex)
{
    ex.printStackTrace();
}
System.out.println("ID \t FIRST NAME                LAST NAME
STANDARD          DIVISION    ROLL NO          BOOKS WITH HIM/HER
ACTIVE");
try
{
    if(rs_rs.next())
    {
        rs_rs.previous();
        while(rs_rs.next())
        {
            String rs_id=rs_rs.getString("R_ID");
            String rs_fn=rs_rs.getString("R_FNAME");
            String rs_ln=rs_rs.getString("R_LNAME");
            String rs_sta=rs_rs.getString("R_STANDARD");
            String rs_di=rs_rs.getString("R_DIVISION");
            String rs_rn=rs_rs.getString("R_ROLLNO");
            String rs_ridc=rs_rs.getString("RID_COUNT");
            String rs_ac=rs_rs.getString("R_ACTIVE");
            int len_rs_fn=rs_fn.length();
            int len_rs_ln=rs_ln.length();
            System.out.print(rs_id+"\t"+rs_fn);

```

```

        for(int a=1;a<=(60-len_rs_fn);a++)
        {
            System.out.print(" ");
        }
        System.out.print(rs_ln);
        for(int a=1;a<=(59-len_rs_ln);a++)
        {
            System.out.print(" ");
        }
        System.out.print(rs_sta+"                "+rs_di+"                "+rs_rn+"
"+rs_ridc+"                "+rs_ac);
        System.out.println();
    }
    rs_rs.close();
}
else
{
    JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE CHECK THE
ENTRIES AS THE DO NOT MATCH TO ANY RECORD IN THE DATABASE");
    temp++;
}
}
catch(Exception ex)
{
    ex.printStackTrace();
}
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
}

public void LibMenuReportMenuDueReport();//PRINTING THE DUE REPORT
{
    try
    {
        String sql="select * from BOOK_TRN";
        rs_dr=st.executeQuery(sql);
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}

```

```

        System.out.println("READER ID   BOOK ID   ISSUE DATE       DUE DATE       USER
LOGIN(ISSUER)");
        try
        {
            while(rs_dr.next())
            {
                String rs_rid=rs_dr.getString("R_ID");
                String rs_bid=rs_dr.getString("B_ID");
                String rs_id=rs_dr.getString("B_ISSUE_DATE");
                String rs_dd=rs_dr.getString("B_PLANNED_RETURN_DATE");
                String rs_ul=rs_dr.getString("U_LOGIN");
                System.out.print(rs_rid+"          "+rs_bid+"          "+rs_id+"          "+rs_dd+"
"+rs_ul);
                System.out.println();
            }
            rs_dr.close();
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
    }

    public void LibMenuReportMenuLendingReport()//PRINTING THE LENDING REPORT
    {
        try
        {
            String sql="select * from B_RTN";
            rs_lr=st.executeQuery(sql);
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println("READER ID   BOOK ID   ISSUE DATE       DUE DATE
RETURN DATE       USER LOGIN(ISSUER)       USER LOGIN(RECEIVER)");
        try
        {
            while(rs_lr.next())

```

```

        {
            String rs_rid=rs_lr.getString("R_ID");
            String rs_bid=rs_lr.getString("B_ID");
            String rs_id=rs_lr.getString("B_ISSUE_DATE");
            String rs_dd=rs_lr.getString("B_PLANNED_RETURN_DATE");
            String rs_rd=rs_lr.getString("B_ACTUAL_RETURN_DATE");
            String rs_ul=rs_lr.getString("U_LOGIN");
            String rs_ulr=rs_lr.getString("U_LOGIN_RECEIVER");
            System.out.print(rs_rid+"      "+rs_bid+"      "+rs_id+"      "+rs_dd+"
"+rs_rd+"      "+rs_ul+"      "+rs_ulr);
            System.out.println();
        }
        rs_lr.close();
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
}

//SAME AS THAT IN CASE OF LIBRARIAN'S REPORTS
public void ReportMenu()
{
    report_menu_choices=new JFrame("CHOICES");
    report_menu_choices.setVisible(true);
    report_menu_choices.setSize(700,500);
    report_menu_choices.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel report_menu_choices_panel=new JPanel(new GridBagLayout());
    GridBagConstraints gbc_report_menu_choices=new GridBagConstraints();
    JButton book_search,reader_search,due_report,lending_report,sign_out;
    book_search=new JButton    (" Book Search  ");
    reader_search=new JButton   ("Reader Search ");
    due_report=new JButton      (" Due Report  ");
    lending_report=new JButton   ("Lending Report");
    sign_out=new JButton        (" Sign out  ");
    book_search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_choices.setVisible(false);
            ReportMenuBookSearch();
        }
    });
}

```

```

        }
    });
    reader_search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_choices.setVisible(false);
            ReportMenuReaderSearch();
        }
    });
    due_report.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            ReportMenuDueReport();
        }
    });
    lending_report.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            ReportMenuLendingReport();
        }
    });
    sign_out.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            report_menu_choices.setVisible(false);
            main();
        }
    });
    gbc_report_menu_choices.insets=new Insets(10,10,10,10);
    gbc_report_menu_choices.gridx=0;
    gbc_report_menu_choices.gridy=0;
    report_menu_choices_panel.add(book_search,gbc_report_menu_choices);
    gbc_report_menu_choices.insets=new Insets(10,10,10,10);
    gbc_report_menu_choices.gridx=0;
    gbc_report_menu_choices.gridy=1;
    report_menu_choices_panel.add(reader_search,gbc_report_menu_choices);
    gbc_report_menu_choices.insets=new Insets(10,10,10,10);
    gbc_report_menu_choices.gridx=0;
    gbc_report_menu_choices.gridy=2;
    report_menu_choices_panel.add(due_report,gbc_report_menu_choices);
    gbc_report_menu_choices.insets=new Insets(10,10,10,10);
    gbc_report_menu_choices.gridx=0;
    gbc_report_menu_choices.gridy=3;
    report_menu_choices_panel.add(lending_report,gbc_report_menu_choices);
    gbc_report_menu_choices.insets=new Insets(10,10,10,10);
    gbc_report_menu_choices.gridx=0;
    gbc_report_menu_choices.gridy=4;

```

```

        report_menu_choices_panel.add(sign_out,gbc_report_menu_choices);
        report_menu_choices.add(report_menu_choices_panel);
    }

    public void ReportMenuBookSearch()
    {
        report_menu_book_search=new JFrame("Book Search");
        report_menu_book_search.setVisible(true);
        report_menu_book_search.setSize(700,500);
        report_menu_book_search.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel report_menu_book_search_panel=new JPanel(new GridBagLayout());
        GridBagConstraints gbc_report_book_search=new GridBagConstraints();
        JButton search,back,sign_out;

        search=new JButton("Search");
        back=new JButton("Back");
        sign_out=new JButton("Sign out");

        JLabel book_name=new JLabel("Book Name");
        JLabel author=new JLabel("Author");
        JLabel genre=new JLabel("Genre");
        String GENRE_RM[]={ "
        ", "Autobiography", "Fiction", "Fantasy", "Suspense", "Romance", "Non-
        fiction", "Mythology", "Philosophy", "Pshycology", "Reference Books", "Dictionaries and
        Thesaurus", "Others", "Combination of genres" };

        book_name_tf=new JTextField(20);
        author_tf=new JTextField(10);
        genre_tf=new JComboBox(GENRE_RM);

        search.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent ae)
            {
                ReportMenuBookSearchOutput();
            }
        });
        back.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent ae)
            {
                report_menu_book_search.setVisible(false);
                ReportMenu();
            }
        });
        sign_out.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent ae)
            {
                report_menu_book_search.setVisible(false);
                main();
            }
        });
    }

```

```

    }
});

gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(book_name,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(author,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;
gbc_report_book_search.gridy=0;
report_menu_book_search_panel.add(genre,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(book_name_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(author_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;
gbc_report_book_search.gridy=1;
report_menu_book_search_panel.add(genre_tf,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=0;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(search,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=1;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(back,gbc_report_book_search);
gbc_report_book_search.insets=new Insets(10,10,10,10);
gbc_report_book_search.gridx=2;
gbc_report_book_search.gridy=2;
report_menu_book_search_panel.add(sign_out,gbc_report_book_search);
report_menu_book_search.add(report_menu_book_search_panel);
}

public void ReportMenuBookSearchOutput()
{
String bn=book_name_tf.getText().trim();
String au=author_tf.getText().trim();
String gn=genre_tf.getSelectedItem().toString().trim();

```

```

int temp=0;
String sql="";
if(gn.equals("Combination of genres"))
{
    JTextField genre_actual=new JTextField(20);
    JPanel genre_actual_panel=new JPanel();
    genre_actual_panel.add(genre_actual);
    int so_genre_actual=JOptionPane.showConfirmDialog(null,genre_actual_panel,"Enter the
genres",JOptionPane.OK_CANCEL_OPTION);
    if(so_genre_actual==JOptionPane.OK_OPTION)
    {
        gn=genre_actual.getText().trim();
    }
    else
    {
        temp++;
    }
}
if(temp==0)
{
    if(bn.equals("") && au.equals("") && gn.equals(""))
    {
        sql="select * from BOOK_MST where B_ACTIVE='Yes'";
    }
    else if(gn.equals(""))
    {
        if(au.equals(""))
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_ACTIVE='Yes'";
        }
        else if(bn.equals(""))
        {
            sql="select * from BOOK_MST where B_AUTHOR='"+au+"' and
B_ACTIVE='Yes'";
        }
        else
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and
B_AUTHOR='"+au+"' and B_ACTIVE='Yes'";
        }
    }
    else if(au.equals(""))
    {
        if(gn.equals(""))
        {
            sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_ACTIVE='Yes'";
        }
        else if(bn.equals(""))

```



```

    {
        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from BOOK_MST where B_NAME='"+bn+"' and B_GENRE='"+gn+"' and B_ACTIVE='Yes'";
    }
}
else if(bn.equals(""))
{
    if(au.equals(""))
    {
        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_ACTIVE='Yes'";
    }
    else if(gn.equals(""))
    {
        sql="select * from BOOK_MST where B_AUTHOR='"+au+"' and B_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_AUTHOR='"+au+"' and B_ACTIVE='Yes'";
    }
}
else
{
    sql="select * from BOOK_MST where B_GENRE='"+gn+"' and B_AUTHOR='"+au+"' and B_NAME='"+bn+"' and B_ACTIVE='Yes'";
}
}
if(temp==0)
{
    try
    {
        rs_bs=st.executeQuery(sql);
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
}
if(temp==0)
{
    System.out.println("ID \t NAME \t PRICE \t CURRENCY \t GENRE \t AUTHOR \t AVAILABLE");
}

```

```

if(temp==0)
{
    try
    {
        if(rs_bs.next())
        {
            rs_bs.previous();
            while(rs_bs.next())
            {
                String rs_id=rs_bs.getString("B_ID");
                String rs_bn=rs_bs.getString("B_NAME");
                String rs_au=rs_bs.getString("B_AUTHOR");
                String rs_pr=rs_bs.getString("B_PRICE");
                String rs_cu=rs_bs.getString("B_CURRENCY");
                String rs_gn=rs_bs.getString("B_GENRE");
                String rs_av=rs_bs.getString("B_AVAILABLE");
                String rs_ac=rs_bs.getString("B_ACTIVE");
                int len_rs_bn=rs_bn.length();
                int len_rs_au=rs_au.length();
                int len_rs_pr=rs_pr.length();
                int len_rs_gn=rs_gn.length();
                System.out.print(rs_id+"\t"+rs_bn);
                for(int a=1;a<=(53-len_rs_bn);a++)
                {
                    System.out.print(" ");
                }
                System.out.print(rs_au);
                for(int a=1;a<=(56-len_rs_au);a++)
                {
                    System.out.print(" ");
                }
                System.out.print(rs_pr);
                for(int a=1;a<=(25-len_rs_pr);a++)
                {
                    System.out.print(" ");
                }
                System.out.print(rs_cu+"          "+rs_gn);
                for(int a=1;a<=(51-len_rs_gn);a++)
                {
                    System.out.print(" ");
                }
                System.out.print(rs_av+"          "+rs_ac);
                System.out.println();
            }
            rs_bs.close();
        }
    }
    else

```

```

        {
            JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE CHECK THE
ENTRIES AS THE DO NOT MATCH TO ANY RECORD IN THE DATABASE");
            temp++;
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    System.out.println();
    }
}

```

```

public void ReportMenuReaderSearch()
{
    report_menu_reader_search=new JFrame("Reader Search");
    report_menu_reader_search.setVisible(true);
    report_menu_reader_search.setSize(700,500);
    report_menu_reader_search.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JPanel report_menu_reader_search_panel=new JPanel(new GridBagLayout());
    GridBagConstraints gbc_report_reader_search=new GridBagConstraints();
    JButton search,back,sign_out;

    search=new JButton("Search");
    back=new JButton("Back");
    sign_out=new JButton("Sign out");

    JLabel first_name=new JLabel("First Name");
    JLabel last_name=new JLabel("Last Name");
    JLabel standard=new JLabel("Standard");

    first_name_tf=new JTextField(20);
    last_name_tf=new JTextField(20);
    standard_tf=new JTextField(10);

    search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {

```

```

        ReportMenuReaderSearchOutput();
    }
});
back.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        report_menu_reader_search.setVisible(false);
        ReportMenu();
    }
});
sign_out.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        report_menu_reader_search.setVisible(false);
        main();
    }
});

gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=0;
gbc_report_reader_search.gridy=0;
report_menu_reader_search_panel.add(first_name,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=1;
gbc_report_reader_search.gridy=0;
report_menu_reader_search_panel.add(last_name,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=2;
gbc_report_reader_search.gridy=0;
report_menu_reader_search_panel.add(standard,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=0;
gbc_report_reader_search.gridy=1;
report_menu_reader_search_panel.add(first_name_tf,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=1;
gbc_report_reader_search.gridy=1;
report_menu_reader_search_panel.add(last_name_tf,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=2;
gbc_report_reader_search.gridy=1;
report_menu_reader_search_panel.add(standard_tf,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=0;
gbc_report_reader_search.gridy=2;
report_menu_reader_search_panel.add(search,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);

```

```

gbc_report_reader_search.gridx=1;
gbc_report_reader_search.gridy=2;
report_menu_reader_search_panel.add(back,gbc_report_reader_search);
gbc_report_reader_search.insets=new Insets(10,10,10,10);
gbc_report_reader_search.gridx=2;
gbc_report_reader_search.gridy=2;
report_menu_reader_search_panel.add(sign_out,gbc_report_reader_search);
report_menu_reader_search.add(report_menu_reader_search_panel);
}

public void ReportMenuReaderSearchOutput()
{
    String fn=first_name_tf.getText().trim();
    String ln=last_name_tf.getText().trim();
    String sta=standard_tf.getText().trim();
    String sql;
    if(fn.equals("") && ln.equals("") && sta.equals(""))
    {
        sql="select * from READER_MST where R_ACTIVE='Yes'";
    }
    else if(fn.equals(""))
    {
        if(ln.equals(""))
        {
            sql="select * from READER_MST where R_STANDARD='"+sta+"' and
R_ACTIVE='Yes'";
        }
        else if(sta.equals(""))
        {
            sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_ACTIVE='Yes'";
        }
        else
        {
            sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_STANDARD='"+sta+"' and R_ACTIVE='Yes'";
        }
    }
    else if(ln.equals(""))
    {
        if(fn.equals(""))
        {
            sql="select * from READER_MST where R_STANDARD='"+sta+"' and
R_ACTIVE='Yes'";
        }
        else if(sta.equals(""))
        {

```

```

        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_STANDARD='"+sta+"' and R_ACTIVE='Yes'";
    }
}
else if(sta.equals(""))
{
    if(fn.equals(""))
    {
        sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_ACTIVE='Yes'";
    }
    else if(ln.equals(""))
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_ACTIVE='Yes'";
    }
    else
    {
        sql="select * from READER_MST where R_FNAME='"+fn+"' and
R_LNAME='"+ln+"' and R_ACTIVE='Yes'";
    }
}
else
{
    sql="select * from READER_MST where R_LNAME='"+ln+"' and
R_STANDARD='"+sta+"' and R_FNAME='"+fn+"' and R_ACTIVE='Yes'";
}
int temp=0;
try
{
    rs_rs=st.executeQuery(sql);
}
catch(Exception ex)
{
    ex.printStackTrace();
}
System.out.println("ID \t FIRST NAME \t\t\t\t LAST NAME
STANDARD \t\t\t DIVISION \t\t ROLL NO \t\t\t\t BOOKS WITH HIM/HER
ACTIVE");
try
{
    if(rs_rs.next())
    {
        rs_rs.previous();

```

```

while(rs_rs.next())
{
    String rs_id=rs_rs.getString("R_ID");
    String rs_fn=rs_rs.getString("R_FNAME");
    String rs_ln=rs_rs.getString("R_LNAME");
    String rs_sta=rs_rs.getString("R_STANDARD");
    String rs_di=rs_rs.getString("R_DIVISION");
    String rs_rn=rs_rs.getString("R_ROLLNO");
    String rs_ridc=rs_rs.getString("RID_COUNT");
    String rs_ac=rs_rs.getString("R_ACTIVE");
    int len_rs_fn=rs_fn.length();
    int len_rs_ln=rs_ln.length();
    System.out.print(rs_id+"\t"+rs_fn);
    for(int a=1;a<=(60-len_rs_fn);a++)
    {
        System.out.print(" ");
    }
    System.out.print(rs_ln);
    for(int a=1;a<=(59-len_rs_ln);a++)
    {
        System.out.print(" ");
    }
    System.out.print(rs_sta+"          "+rs_di+"          "+rs_rn+"
"+rs_ridc+"          "+rs_ac);
    System.out.println();
}
rs_rs.close();
}
else
{
    JOptionPane.showMessageDialog(null,"SORRY BUT PLEASE CHECK THE
ENTRIES AS THE DO NOT MATCH TO ANY RECORD IN THE DATABASE");
    temp++;
}
}
catch(Exception ex)
{
    ex.printStackTrace();
}
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();
System.out.println();

```

```

    }

    public void ReportMenuDueReport()
    {
        try
        {
            String sql="select * from BOOK_TRN";
            rs_dr=st.executeQuery(sql);
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println("READER ID   BOOK ID   ISSUE DATE       DUE DATE       USER
LOGIN(ISSUER)");
        try
        {
            while(rs_dr.next())
            {
                String rs_rid=rs_dr.getString("R_ID");
                String rs_bid=rs_dr.getString("B_ID");
                String rs_id=rs_dr.getString("B_ISSUE_DATE");
                String rs_dd=rs_dr.getString("B_PLANNED_RETURN_DATE");
                String rs_ul=rs_dr.getString("U_LOGIN");
                System.out.print(rs_rid+"      "+rs_bid+"      "+rs_id+"      "+rs_dd+"
"+rs_ul);
                System.out.println();
            }
            rs_dr.close();
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
    }

    public void ReportMenuLendingReport()
    {
        try

```



```

        {
            String sql="select * from B_RTN";
            rs_lr=st.executeQuery(sql);
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println("READER ID   BOOK ID   ISSUE DATE   DUE DATE
RETURN DATE   USER LOGIN(ISSUER)   USER LOGIN(RECEIVER)");
        try
        {
            while(rs_lr.next())
            {
                String rs_rid=rs_lr.getString("R_ID");
                String rs_bid=rs_lr.getString("B_ID");
                String rs_id=rs_lr.getString("B_ISSUE_DATE");
                String rs_dd=rs_lr.getString("B_PLANNED_RETURN_DATE");
                String rs_rd=rs_lr.getString("B_ACTUAL_RETURN_DATE");
                String rs_ul=rs_lr.getString("U_LOGIN");
                String rs_ulr=rs_lr.getString("U_LOGIN_RECEIVER");
                System.out.print(rs_rid+"      "+rs_bid+"      "+rs_id+"      "+rs_dd+"
"+rs_rd+"      "+rs_ul+"      "+rs_ulr);
                System.out.println();
            }
            rs_lr.close();
        }
        catch(Exception ex)
        {
            ex.printStackTrace();
        }
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
        System.out.println();
    }

    public void AdminChoice()//DESIGNING THE PAGE FOR ADMIN MENU CHOICES
    {
        final JFrame AC=new JFrame("CHOICES");
        AC.setSize(500,400);
        AC.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        AC.setVisible(true);
    }

```

```

JPanel ACP=new JPanel(new GridBagLayout());
GridBagConstraints gbc=new GridBagConstraints();

JButton NU=new JButton("    New User    ");
JButton MU=new JButton("    Modify User   ");
JButton RU=new JButton("    Remove User  ");
JButton CP=new JButton("Change Password");
JButton SO=new JButton("    Sign Out    ");

NU.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AC.setVisible(false);
        AdminMenuNewUser();
    }
});
MU.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AC.setVisible(false);
        AdminMenuModifyUser();
    }
});
RU.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AC.setVisible(false);
        AdminMenuRemoveUser();
    }
});
CP.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AC.setVisible(false);
        AdminMenuChangePassword();
    }
});
SO.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AC.setVisible(false);
        main();
    }
});
gbc.insets=new Insets(10,10,10,10);
gbc.gridx=0;

```

```

        gbc.gridy=0;
        ACP.add(NU,gbc);
        gbc.insets=new Insets(10,10,10,10);
        gbc.gridx=0;
        gbc.gridy=1;
        ACP.add(MU,gbc);
        gbc.insets=new Insets(10,10,10,10);
        gbc.gridx=0;
        gbc.gridy=2;
        ACP.add(RU,gbc);
        gbc.insets=new Insets(10,10,10,10);
        gbc.gridx=0;
        gbc.gridy=3;
        ACP.add(CP,gbc);

        gbc.insets=new Insets(10,10,10,10);
        gbc.gridx=0;
        gbc.gridy=4;
        ACP.add(SO,gbc);

        AC.add(ACP);
    }

    public void AdminMenuNewUser()//DESIGNING THE PAGE FOR NEW USER
    {
        AdminMenuNU=new JFrame("New User");
        AdminMenuNU.setVisible(true);
        AdminMenuNU.setSize(700,500);
        AdminMenuNU.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel AdminMenuNUup=new JPanel(new GridBagLayout());
        JLabel LInu=new JLabel("Login ID");
        JLabel UNnu=new JLabel("User Name");
        JLabel PWnu=new JLabel("Password");
        JLabel Cnu=new JLabel("Category");
        String categorynu[]={"ADMIN","GENERAL USER","LIBRARIAN"};
        cnu=new JComboBox(categorynu);
        linu=new JTextField(20);
        unnu=new JTextField(20);
        pwnu=new JPasswordField(20);
        JButton amnuadd=new JButton("Add");
        amnuadd.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e)
            {
                btnActionAMNUADD();
            }
        });
        JButton amnub=new JButton("Back");
    }

```

```

amnub.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AdminMenuNU.setVisible(false);
        AdminChoice();
    }
});
JButton amnuso=new JButton("Sign out");
amnuso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AdminMenuNU.setVisible(false);
        main();
    }
});
GridBagConstraints gbc1=new GridBagConstraints();
linu.setText("");
unnu.setText("");
pwnu.setText("");

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=0;
AdminMenuNUp.add(LInu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=0;
AdminMenuNUp.add(linu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=1;
AdminMenuNUp.add(UNnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=1;
AdminMenuNUp.add(unnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=2;
AdminMenuNUp.add(PWnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;

```

```

gbc1.gridy=2;
AdminMenuNUp.add(pwnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=3;
AdminMenuNUp.add(Cnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=3;
AdminMenuNUp.add(cnu,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=4;
AdminMenuNUp.add(amnuadd,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=4;
AdminMenuNUp.add(amnub,gbc1);

gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=2;
gbc1.gridy=4;
AdminMenuNUp.add(amnuso,gbc1);

AdminMenuNU.add(AdminMenuNUp);
}

```

```

public void btnActionAMNUADD()//BUTTON ACTION FOR "ADD" (ADDING IN THE
DATABASE)

```

```

{
    try
    {
        String sqlnu="select * from USER_MST";
        rsnu=st.executeQuery(sqlnu);
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while connecting");
    }
    int temporary=1;
    String LoginIdnu=linu.getText().trim();
    String UserNamenu=unnu.getText().trim();
    String PassWordnu=pwnu.getText().trim();
    String Categorynu=cnu.getSelectedItem().toString();
}

```

```

try
{
    if(LoginIdnu.equals("") || UserNamenu.equals("") || PassWordnu.equals(""))
    {
        JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been left
without data");
        temporary++;
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while blank textfield check");
}
try
{
    while(temporary==1 && rsnu.next())
    {
        String usertempnu=rsnu.getString("U_LOGIN");
        if(usertempnu.equals(LoginIdnu))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another user with the same
Login Id");
            temporary++;
        }
    }
    while(temporary==1 && rsnu.previous())
    {
        String usertempnu=rsnu.getString("U_LOGIN");
        if(usertempnu.equals(LoginIdnu))
        {
            JOptionPane.showMessageDialog(null,"Sorry but there is another user with the same
Login Id");
            temporary++;
        }
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while user id");
}
try
{
    if(temporary==1)
    {
        rsnu.moveToInsertRow();
        rsnu.updateString("U_NAME",UserNamenu);
        rsnu.updateString("U_LOGIN",LoginIdnu);
        rsnu.updateString("U_PASSWORD",PassWordnu);
    }
}

```

```

        rsnu.updateString("U_CATEGORY",Categorynu);
        JOptionPane.showMessageDialog(null,"User Added");
        AdminMenuNU.setVisible(false);
        AdminChoice();
        rsnu.updateString("U_ACTIVE","Yes");
        rsnu.insertRow();
        rsnu.close();
        st.close();

st=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDAT
ABLE);
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while adding");
}
}

public void AdminMenuModifyUser()//TAKING THE LOGIN ID OF THE USER TO BE
MODIFIED
{
    JLabel amlabeluserentrymodify=new JLabel("LOGIN ID:");
    JTextField amuserentrymodify=new JTextField(20);
    JPanel amentrymodify=new JPanel();
    amentrymodify.add(amlabeluserentrymodify);
    amentrymodify.add(amuserentrymodify);
    int
muSelectedOption=JOptionPane.showConfirmDialog(null,amentrymodify,"LOGIN",JOptionPane.
OK_CANCEL_OPTION);
    if(muSelectedOption==JOptionPane.OK_OPTION)
    {
        amuserentry=amuserentrymodify.getText().trim();
        btnActionAdminMenuModifyUser();
    }
    else
    {
        AdminChoice();
    }
}

public void btnActionAdminMenuModifyUser()//DESIGNING THE PAGE FOR MODIFY
USER
{
    int tempmu=0;
    try
    {
        String sqlmu="select * from USER_MST where U_LOGIN='"+amuserentry+"' and
U_ACTIVE='Yes'";

```

```

        rsmu=st.executeQuery(sqlmu);
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while connecting");
        AdminMenuModifyUser();
    }
    limu=new JTextField(20);
    unmu=new JTextField(20);
    String PWmu1="";
    try
    {
        if(rsmu.next())
        {
            limu.setText(rsmu.getString("U_LOGIN"));
            unmu.setText(rsmu.getString("U_NAME"));
            PWmu1=rsmu.getString("U_PASSWORD");
            JOptionPane.showMessageDialog(null,"PLEASE NOTE THAT THE CATEGORY
OF THE THE USER BEING SHOWN IS ADMIN, CHANGE IT AS PER REQUIREMENT");
        }
        else
        {
            JOptionPane.showMessageDialog(null,"SORRY BUT THE USER ID DOES NOT
EXIST");
            tempmu++;
            AdminChoice();
        }
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while allotting the text");
    }
    if(tempmu==0)
    {
        AdminMenuMU=new JFrame("Modify User");
        AdminMenuMU.setVisible(true);
        AdminMenuMU.setSize(700,500);
        AdminMenuMU.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel AdminMenuMUp=new JPanel(new GridBagLayout());
        JLabel LImu=new JLabel("Login ID");
        JLabel UNmu=new JLabel("User Name");
        JLabel PWmu=new JLabel("Password");
        JLabel Cmu=new JLabel("Category");
        String categorymu[]={"ADMIN","GENERAL USER","LIBRARIAN"};
        cmu=new JComboBox(categorymu);
        JButton ammumodify=new JButton("Modify");
        ammumodify.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e)

```



```

        {
            btnActionAMMUMODIFY();
        }
    });
    JButton ammub=new JButton("Back");
    ammub.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            AdminMenuMU.setVisible(false);
            AdminChoice();
        }
    });
    JButton ammuso=new JButton("Sign out");
    ammuso.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            AdminMenuMU.setVisible(false);
            main();
        }
    });
    GridBagConstraints gbc1=new GridBagConstraints();

    int PWmu1len=PWmu1.length();
    char chtempammu='*';
    String PWmu2="";
    for(int temp_in_ammu=0;temp_in_ammu<PWmu1len;temp_in_ammu++)
    {
        PWmu2=PWmu2+chtempammu;
    }
    JLabel pwmu=new JLabel(PWmu2);

    gbc1.insets=new Insets(10,10,10,10);
    gbc1.gridx=0;
    gbc1.gridy=0;
    AdminMenuMUp.add(LImu,gbc1);

    gbc1.insets=new Insets(10,10,10,10);
    gbc1.gridx=1;
    gbc1.gridy=0;
    AdminMenuMUp.add(limu,gbc1);

    gbc1.insets=new Insets(10,10,10,10);
    gbc1.gridx=0;
    gbc1.gridy=1;
    AdminMenuMUp.add(UNmu,gbc1);

    gbc1.insets=new Insets(10,10,10,10);

```

```

        gbc1.gridx=1;
        gbc1.gridy=1;
        AdminMenuMUp.add(unmu,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=0;
        gbc1.gridy=2;
        AdminMenuMUp.add(PWmu,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=1;
        gbc1.gridy=2;
        AdminMenuMUp.add(pwmu,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=0;
        gbc1.gridy=3;
        AdminMenuMUp.add(Cmu,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=1;
        gbc1.gridy=3;
        AdminMenuMUp.add(cmu,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=0;
        gbc1.gridy=4;
        AdminMenuMUp.add(ammumodify,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=1;
        gbc1.gridy=4;
        AdminMenuMUp.add(ammub,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=2;
        gbc1.gridy=4;
        AdminMenuMUp.add(ammuso,gbc1);

        AdminMenuMU.add(AdminMenuMUp);
    }
}

```

```

    public void btnActionAMMUMODIFY()//BUTTON ACTION FOR "MODIFY"
(MODIFYING IN THE DATABASE)

```

```

    {
        int temporary=1;
        String LoginIdmu=limu.getText().trim();

```

```

String UserNamemu=unmu.getText().trim();
String Categorymu=cmu.getSelectedItem().toString();
try
{
    if(LoginIdmu.equals("") || UserNamemu.equals(""))
    {
        JOptionPane.showMessageDialog(null,"Sorry but one or more Textfields have been left
without data");
        temporary++;
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while blank textfield check");
}
if(temporary==1)
{
    try
    {
        rsmu.updateString("U_NAME",UserNamemu);
        rsmu.updateString("U_LOGIN",LoginIdmu);
        rsmu.updateString("U_CATEGORY",Categorymu);
        rsmu.updateRow();
        JOptionPane.showMessageDialog(null,"User Modified");
        AdminMenuMU.setVisible(false);
        AdminChoice();
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT THE LOGIN IS THAT OF AN
EXISTING USER");
    }
}

public void AdminMenuRemoveUser()//DESIGNING THE PAGE FOR REMOVE USER
{
    int tempru=0;
    String aruserentry="";
    JLabel amlabeluserentryremove=new JLabel("LOGIN ID:");
    JTextField amuserentryremove=new JTextField(20);
    JPanel amentryremove=new JPanel();
    amentryremove.add(amlabeluserentryremove);
    amentryremove.add(amuserentryremove);
    int
ruSelectedOption=JOptionPane.showConfirmDialog(null,amentryremove,"LOGIN",JOptionPane.
OK_CANCEL_OPTION);
    if(ruSelectedOption==JOptionPane.CANCEL_OPTION)

```

```

        {
            AdminChoice();
            tempru++;
        }
    else if(ruSelectedOption==JOptionPane.OK_OPTION)
    {
        aruuserentry=amuserentryremove.getText().trim();
    }
    else
    {
        AdminChoice();
        tempru++;
    }
    if(tempru==0)
    {
        try
        {
            String sqlru="select * from USER_MST where U_LOGIN='"+aruuserentry+"' and
U_ACTIVE='Yes'";
            rsru=st.executeQuery(sqlru);
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while connecting");
            AdminMenuRemoveUser();
        }
        AdminMenuRU=new JFrame("Remove User");
        AdminMenuRU.setVisible(true);
        AdminMenuRU.setSize(700,500);
        AdminMenuRU.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel AdminMenuRUp=new JPanel(new GridBagLayout());
        JLabel LIru=new JLabel("Login ID");
        JLabel UNru=new JLabel("User Name");
        JLabel PWru=new JLabel("Password");
        JLabel Cru=new JLabel("Category");
        JLabel RMru=new JLabel("Remark");
        RMru1=new JTextField(20);
        String strliru="",strunru="",strcaru="";
        String PWru1="";
        JButton amruremove=new JButton("Remove");
        amruremove.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e)
            {
                btnActionAMRUREMOVE();
            }
        });
        JButton amrub=new JButton("Back");
        amrub.addActionListener(new ActionListener(){

```

```

        public void actionPerformed(ActionEvent ae)
        {
            AdminMenuRU.setVisible(false);
            AdminChoice();
        }
    });
    JButton amruso=new JButton("Sign out");
    amruso.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae)
        {
            AdminMenuRU.setVisible(false);
            main();
        }
    });
    GridBagConstraints gbc1=new GridBagConstraints();
    try
    {
        if(rsru.next())
        {
            strliru=rsru.getString("U_LOGIN");
            strunru=rsru.getString("U_NAME");
            PWru1=rsru.getString("U_PASSWORD");
            strcaru=rsru.getString("U_CATEGORY");
        }
        else
        {
            AdminMenuRU.setVisible(false);
            JOptionPane.showMessageDialog(null,"SORRY BUT THE USER ID DOES NOT
EXIST");
            AdminChoice();
        }
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(null,"Exception while allotting the text");
    }
    int PWru1len=PWru1.length();
    char chtempamru='*';
    String PWru2="";
    for(int temp_in_amru=0;temp_in_amru<PWru1len;temp_in_amru++)
    {
        PWru2=PWru2+chtempamru;
    }
    JLabel pwru=new JLabel(PWru2);
    JLabel liru=new JLabel(strliru);
    JLabel unru=new JLabel(strunru);
    JLabel caru=new JLabel(strcaru);

```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=0;
AdminMenuRUp.add(LIru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=0;
AdminMenuRUp.add(liru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=1;
AdminMenuRUp.add(UNru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=1;
AdminMenuRUp.add(unru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=2;
AdminMenuRUp.add(PWru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=2;
AdminMenuRUp.add(pwru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=3;
AdminMenuRUp.add(Cru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
gbc1.gridy=3;
AdminMenuRUp.add(caru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=0;
gbc1.gridy=4;
AdminMenuRUp.add(RMru,gbc1);
```

```
gbc1.insets=new Insets(10,10,10,10);
gbc1.gridx=1;
```

```

        gbc1.gridy=4;
        AdminMenuRUp.add(RMru1,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=0;
        gbc1.gridy=5;
        AdminMenuRUp.add(amrremove,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=1;
        gbc1.gridy=5;
        AdminMenuRUp.add(amrub,gbc1);

        gbc1.insets=new Insets(10,10,10,10);
        gbc1.gridx=2;
        gbc1.gridy=5;
        AdminMenuRUp.add(amruso,gbc1);

        AdminMenuRU.add(AdminMenuRUp);

    }
}

public void btnActionAMRUREMOVE()//BUTTON ACTION FOR "REMOVE"
(REMOVING IN THE DATABASE)
{
    String remark=RMru1.getText().trim();
    int temporary=0;
    if(remark.equals(""))
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT A REMARK IS REQUIRED AS
        TO WHY THE USER IS BEING DELETED");
        temporary++;
    }
    if(temporary==0)
    {
        try
        {
            rsru.updateString("U_ACTIVE","No");
            rsru.updateString("U_REMARK",remark);
            rsru.updateRow();
            JOptionPane.showMessageDialog(null,"User Removed");
            AdminMenuRU.setVisible(false);
            AdminChoice();
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while removing");

```

```

    }
    }
}

public void AdminMenuChangePassword()//DESIGNING THE PAGE FOR CHANGE
PASSWORD
{
    int tempcp=0;
    String acpuserentry="";
    JLabel amlabeluserentrychangepassword=new JLabel("LOGIN ID:");
    JTextField amuserentrychangepassword=new JTextField(20);
    JPanel amentrychangepassword=new JPanel();
    amentrychangepassword.add(amlabeluserentrychangepassword);
    amentrychangepassword.add(amuserentrychangepassword);
    int
    cpSelectedOption=JOptionPane.showConfirmDialog(null,amentrychangepassword,"LOGIN",JOpt
    ionPane.OK_CANCEL_OPTION);
    if(cpSelectedOption==JOptionPane.CANCEL_OPTION)
    {
        AdminChoice();
        tempcp++;
    }
    else if(cpSelectedOption==JOptionPane.OK_OPTION)
    {
        acpuserentry=amuserentrychangepassword.getText().trim();
    }
    else
    {
        AdminChoice();
        tempcp++;
    }
    if(tempcp==0)
    {
        try
        {
            String sqlcp="select * from USER_MST where U_LOGIN='"+acpuserentry+"' and
            U_ACTIVE='Yes'";
            rscp=st.executeQuery(sqlcp);
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while connecting");
            AdminMenuChangePassword();
        }
        AdminMenuCP=new JFrame("Change Password");
        AdminMenuCP.setVisible(true);
        AdminMenuCP.setSize(700,500);
        AdminMenuCP.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

```



```

JPanel AdminMenuCPp=new JPanel(new GridBagLayout());
JLabel LIcp=new JLabel("Login ID");
JLabel UNcp=new JLabel("User Name");
JLabel PWoldcp=new JLabel("Old Password");
JLabel PWnewcp=new JLabel("New Password");
JLabel PWconfirmcp=new JLabel("Confirm Password");
JLabel Ccp=new JLabel("Category");
pwoldcp=new JPasswordField(20);
pwnewcp=new JPasswordField(20);
pwconfirmcp=new JPasswordField(20);
JButton amcpchangepassword=new JButton("Change");
amcpchangepassword.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e)
    {
        btnActionAMCPCHANGEPASSWORD();
    }
});
JButton amcpb=new JButton("Back");
amcpb.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AdminMenuCP.setVisible(false);
        AdminChoice();
    }
});
JButton amcpso=new JButton("Sign out");
amcpso.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae)
    {
        AdminMenuCP.setVisible(false);
        main();
    }
});
GridBagConstraints gbcamcp=new GridBagConstraints();
String strlicp="",struncp="",strcacp="";
try
{
    if(rscp.next())
    {
        strlicp=rscp.getString("U_LOGIN");
        struncp=rscp.getString("U_NAME");
        strpwcp=rscp.getString("U_PASSWORD");
        strcacp=rscp.getString("U_CATEGORY");
    }
    else
    {
        AdminMenuCP.setVisible(false);

```

```

EXIST");
        JOptionPane.showMessageDialog(null,"SORRY BUT THE USER ID DOES NOT
EXIST");
        AdminChoice();
    }
}
catch(Exception ex)
{
    JOptionPane.showMessageDialog(null,"Exception while allotting the text");
}
JLabel licp=new JLabel(strlicp);
JLabel uncp=new JLabel(struncp);
JLabel cacp=new JLabel(strcacp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=0;
gbcamcp.gridy=0;
AdminMenuCPp.add(LIcp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=1;
gbcamcp.gridy=0;
AdminMenuCPp.add(licp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=0;
gbcamcp.gridy=1;
AdminMenuCPp.add(UNcp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=1;
gbcamcp.gridy=1;
AdminMenuCPp.add(uncp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=0;
gbcamcp.gridy=2;
AdminMenuCPp.add(PWoldcp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=1;
gbcamcp.gridy=2;
AdminMenuCPp.add(pwoldcp,gbcamcp);

gbcamcp.insets=new Insets(10,10,10,10);
gbcamcp.gridx=0;
gbcamcp.gridy=3;
AdminMenuCPp.add(PWnewcp,gbcamcp);

```

```

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=1;
        gbcamcp.gridy=3;
        AdminMenuCPp.add(pwnewwcp,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=0;
        gbcamcp.gridy=4;
        AdminMenuCPp.add(PWconfirmwcp,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=1;
        gbcamcp.gridy=4;
        AdminMenuCPp.add(pwconfirmwcp,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=0;
        gbcamcp.gridy=5;
        AdminMenuCPp.add(Ccp,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=1;
        gbcamcp.gridy=5;
        AdminMenuCPp.add(cacp,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=0;
        gbcamcp.gridy=6;
        AdminMenuCPp.add(amcpchangeppassword,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=1;
        gbcamcp.gridy=6;
        AdminMenuCPp.add(amcpb,gbcamcp);

        gbcamcp.insets=new Insets(10,10,10,10);
        gbcamcp.gridx=2;
        gbcamcp.gridy=6;
        AdminMenuCPp.add(amcpso,gbcamcp);

        AdminMenuCP.add(AdminMenuCPp);

    }
}

    public void btnActionAMCPCHANGEPPASSWORD()//CHANGING THE PASSWORD IN
    THE DATABASE
    {

```

```

String oldchangeadminmenu=pwoldcp.getText().trim();
String newchangeadminmenu=pwnewcp.getText().trim();
String confirmchangeadminmenu=pwconfirmcp.getText().trim();
if(oldchangeadminmenu.equals(strpwcp))
{
    if(newchangeadminmenu.equals(confirmchangeadminmenu))
    {
        try
        {
            rscp.updateString("U_PASSWORD",newchangeadminmenu);
            rscp.updateRow();
            JOptionPane.showMessageDialog(null,"Password Changed");
            AdminMenuCP.setVisible(false);
            AdminChoice();
        }
        catch(Exception ex)
        {
            JOptionPane.showMessageDialog(null,"Exception while changing");
        }
    }
    else
    {
        JOptionPane.showMessageDialog(null,"SORRY BUT YOUR NEW PASSWORD AND
CONFIRM PASSWORD ARE NOT THE SAME");
    }
}
else
{
    JOptionPane.showMessageDialog(null,"SORRY BUT YOUR OLD PASSWORD AND
ACTUAL PASSWORD ARE NOT THE SAME");
}
}

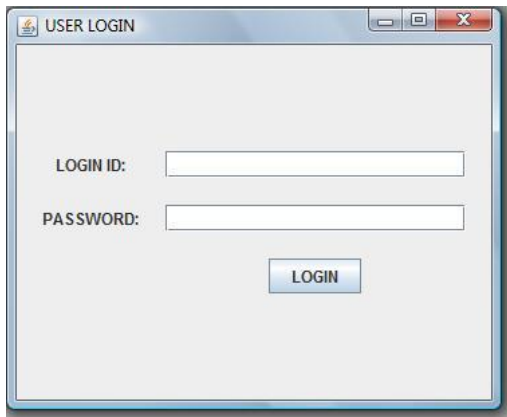
public void main()
{
    new Trial();
}
}

```

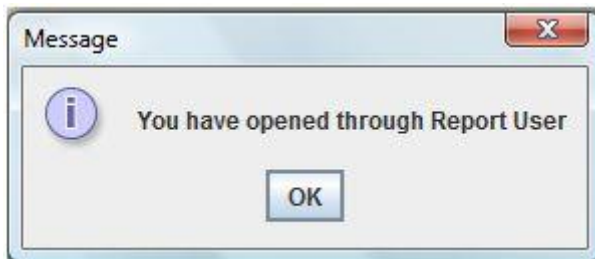
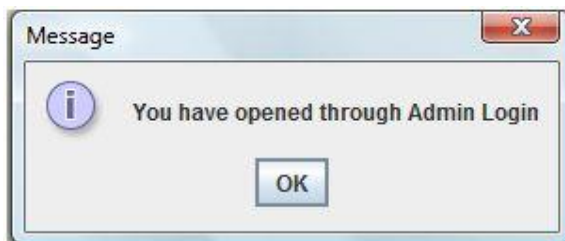
## Output

### Interfacing Screens (Sample)

Login Screen along with Confirmation screen for different Category of Users after Login



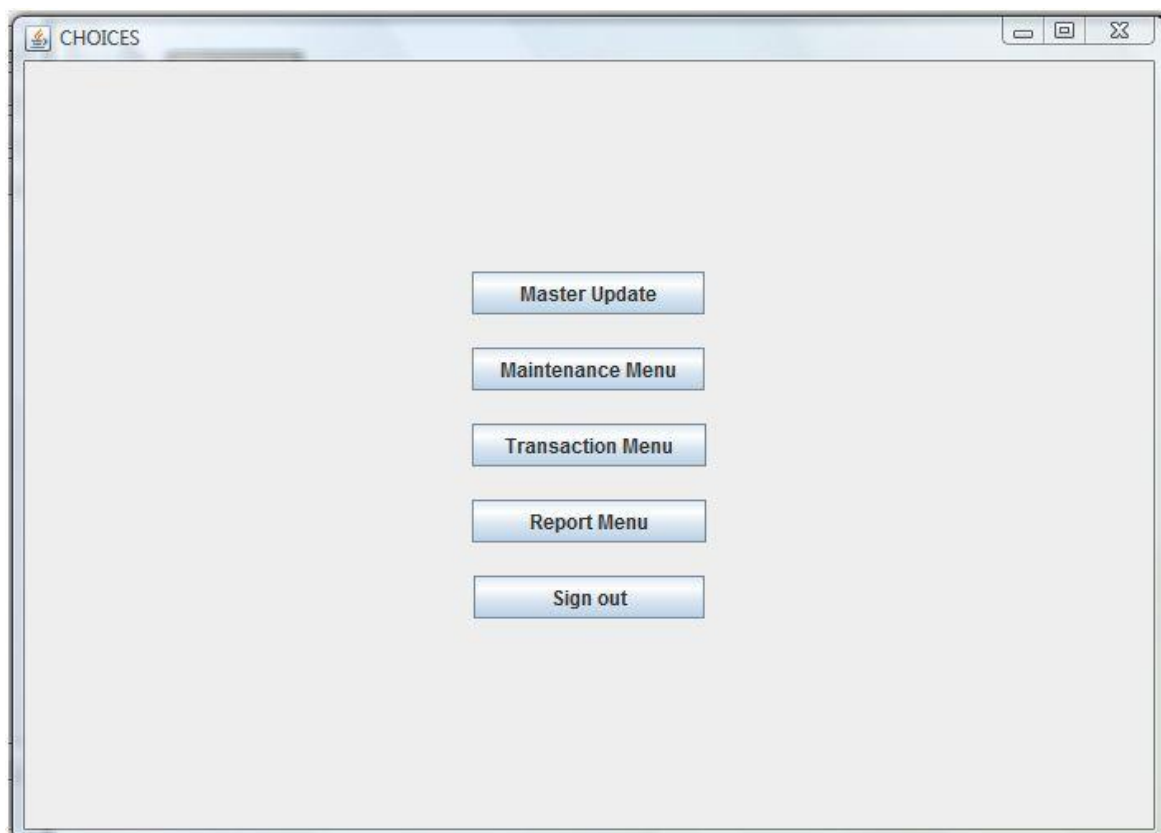
A screenshot of a 'USER LOGIN' window. It features a title bar with a minimize, maximize, and close button. The main area contains two text input fields: 'LOGIN ID:' and 'PASSWORD:'. Below these fields is a 'LOGIN' button.



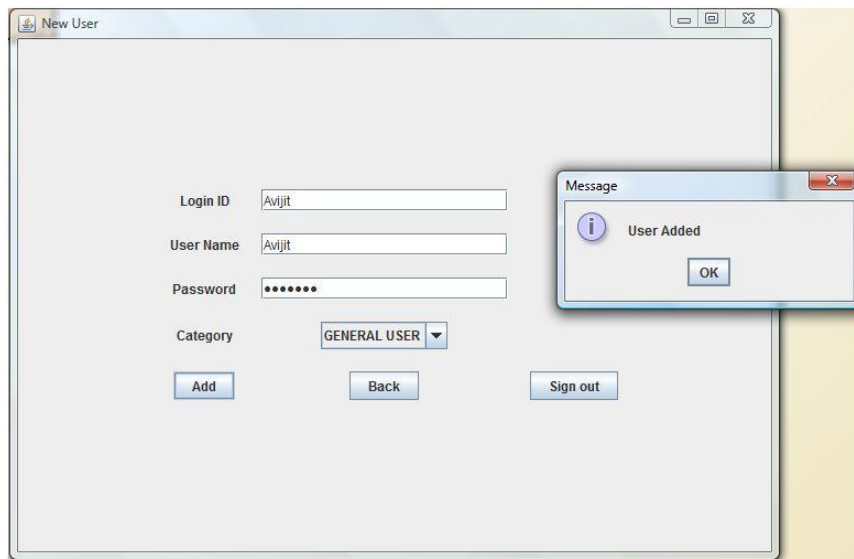
### Menu Option for Admin Type user



### Menu for Librarian type Users



## New User Creation

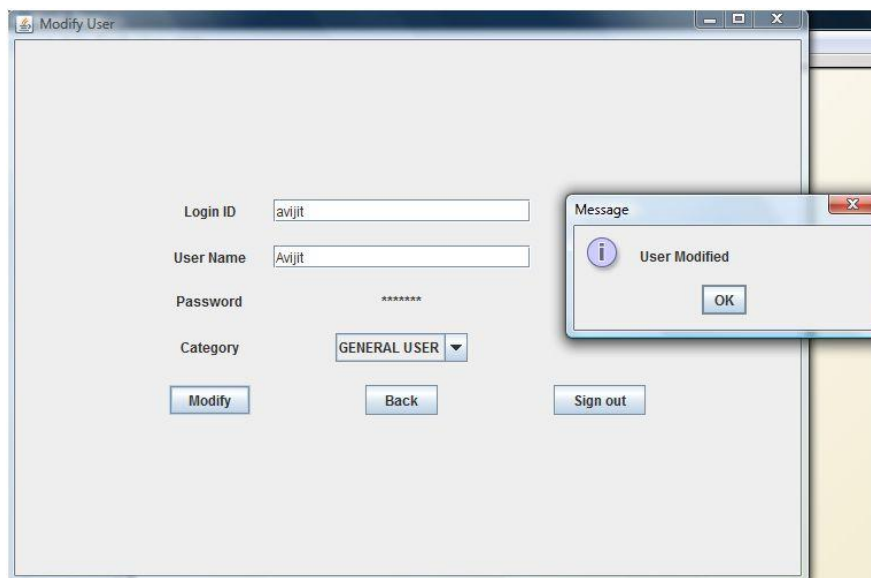


The 'New User' form contains the following fields and controls:

- Login ID:
- User Name:
- Password:
- Category:
- Buttons: Add, Back, Sign out

A 'Message' dialog box is displayed with the text 'User Added' and an 'OK' button.

## Modify User

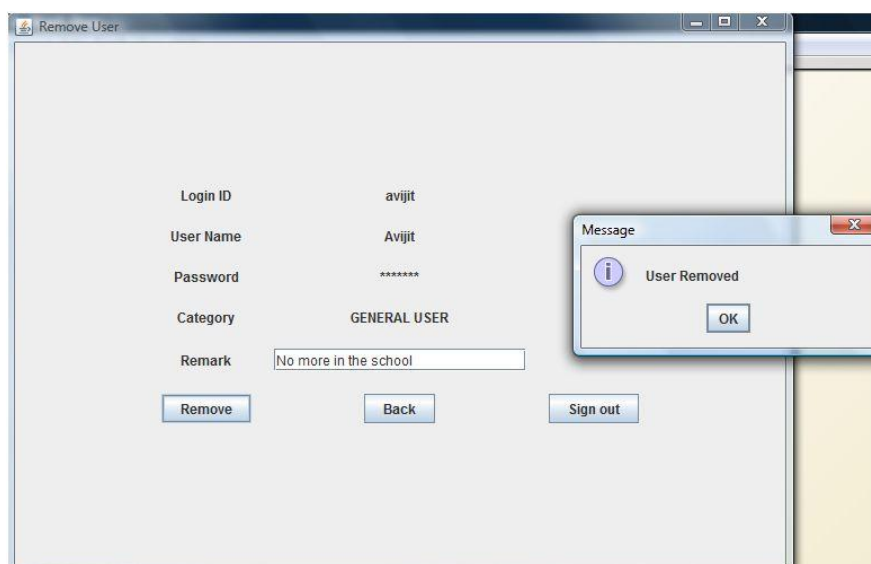


The 'Modify User' form contains the following fields and controls:

- Login ID:
- User Name:
- Password:
- Category:
- Buttons: Modify, Back, Sign out

A 'Message' dialog box is displayed with the text 'User Modified' and an 'OK' button.

## User Removal

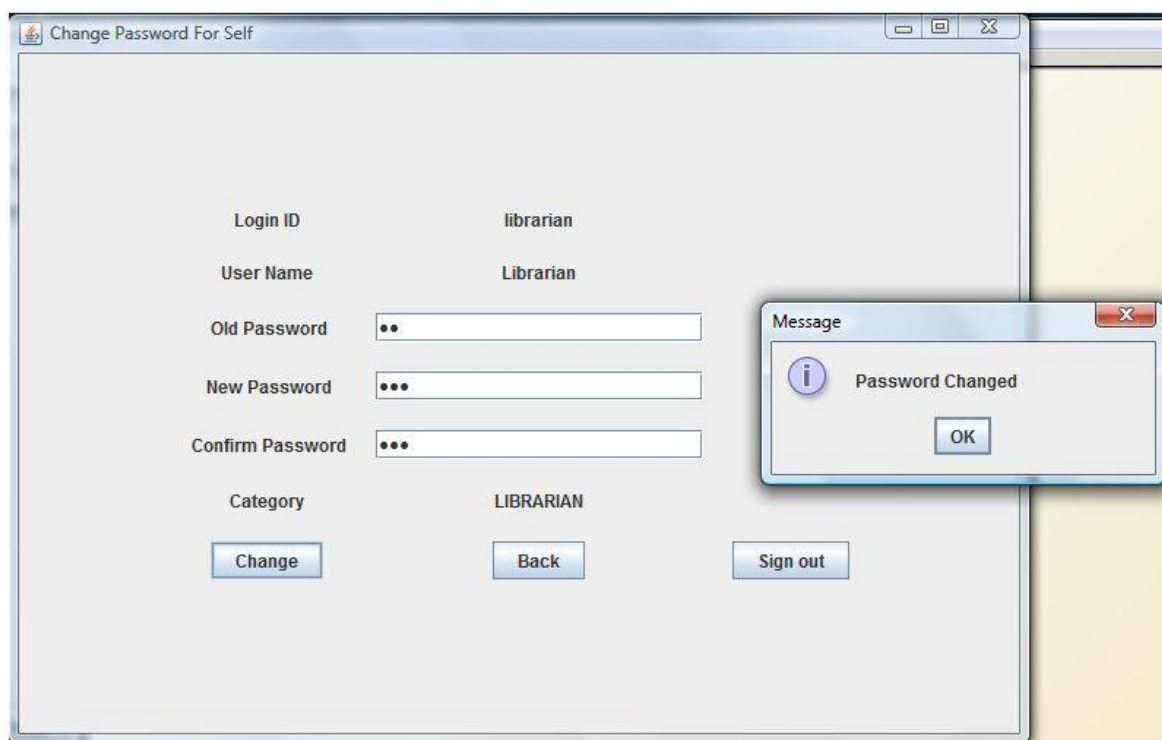


The 'Remove User' form contains the following fields and controls:

- Login ID:
- User Name:
- Password:
- Category:
- Remark:
- Buttons: Remove, Back, Sign out

A 'Message' dialog box is displayed with the text 'User Removed' and an 'OK' button.

## Password Change

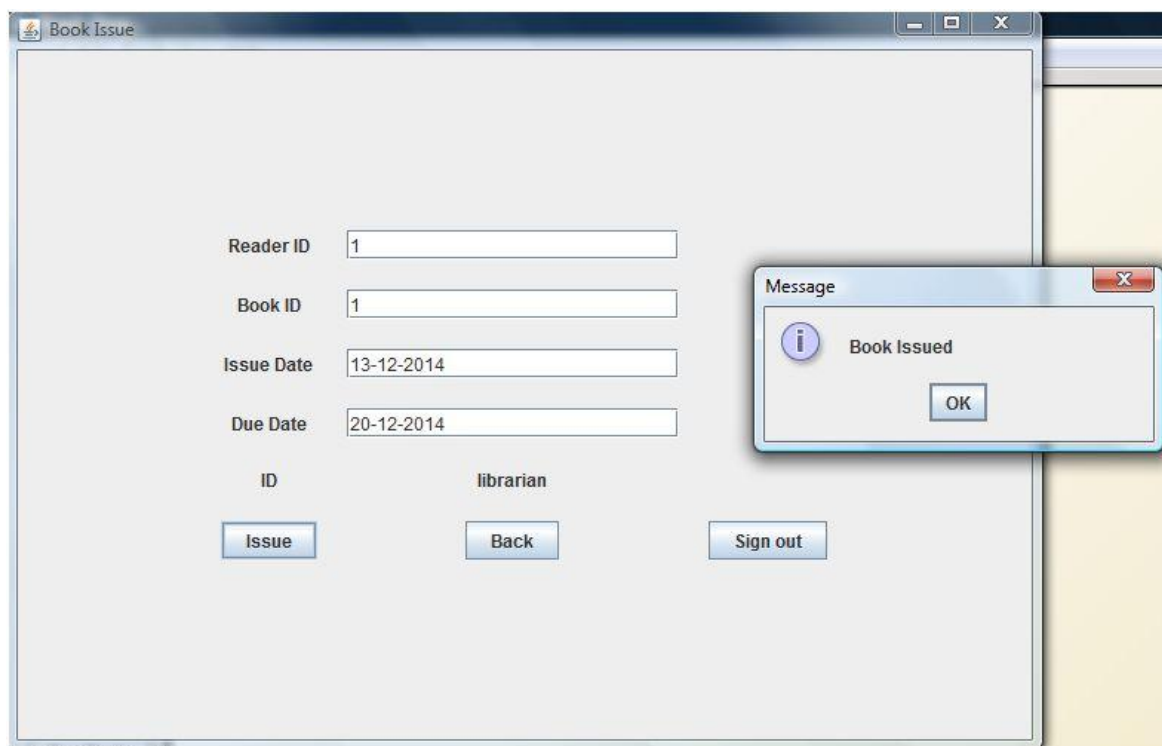


The screenshot shows a web application window titled "Change Password For Self". The form contains the following fields and values:

| Field            | Value     |
|------------------|-----------|
| Login ID         | librarian |
| User Name        | Librarian |
| Old Password     | ••        |
| New Password     | •••       |
| Confirm Password | •••       |
| Category         | LIBRARIAN |

At the bottom of the form are three buttons: "Change", "Back", and "Sign out". A "Message" dialog box is overlaid on the right side of the window, displaying an information icon, the text "Password Changed", and an "OK" button.

## Book Issue



The screenshot shows a web application window titled "Book Issue". The form contains the following fields and values:

| Field      | Value      |
|------------|------------|
| Reader ID  | 1          |
| Book ID    | 1          |
| Issue Date | 13-12-2014 |
| Due Date   | 20-12-2014 |
| ID         | librarian  |

At the bottom of the form are three buttons: "Issue", "Back", and "Sign out". A "Message" dialog box is overlaid on the right side of the window, displaying an information icon, the text "Book Issued", and an "OK" button.



## Book Return

The screenshot shows a web application window titled "Book Return". The main content area contains a form with the following fields and values:

| Field            | Value      |
|------------------|------------|
| Reader Id        | 1          |
| Book Id          | 1          |
| Book Issue Date  | 13-12-2014 |
| Book Due Date    | 20-12-2014 |
| Issue User       | librarian  |
| Book Return Date | 13-12-2014 |

At the bottom of the form, there are three buttons: "Return", "Back", and "Sign Out".

Overlaid on the right side of the window is a "Message" dialog box. It contains an information icon (i) and the text "Book Returned". Below the text is an "OK" button.

## Few Validation - Window Pop ups

The 'Book Issue' window contains the following fields and controls:

- Reader ID:
- Book ID:
- Issue Date:
- Due Date:
- ID:
- Buttons: Issue, Back, Sign out

A 'Message' dialog box is displayed with the text: "Sorry but one or more Textfields have been left without data". The dialog has an 'OK' button.

The 'Change Password For Self' window contains the following fields and controls:

- Login ID:
- User Name:
- Old Password:
- New Password:
- Confirm Password:
- Category:
- Buttons: Change, Back, Sign out

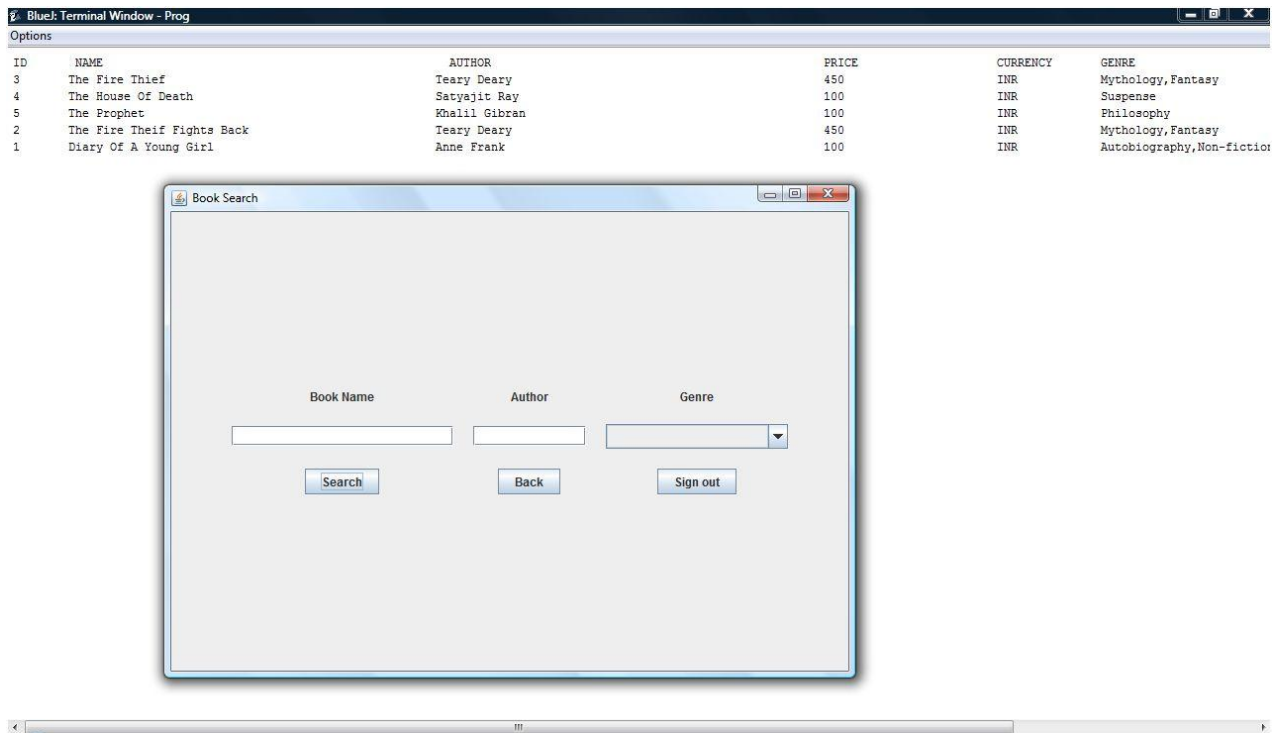
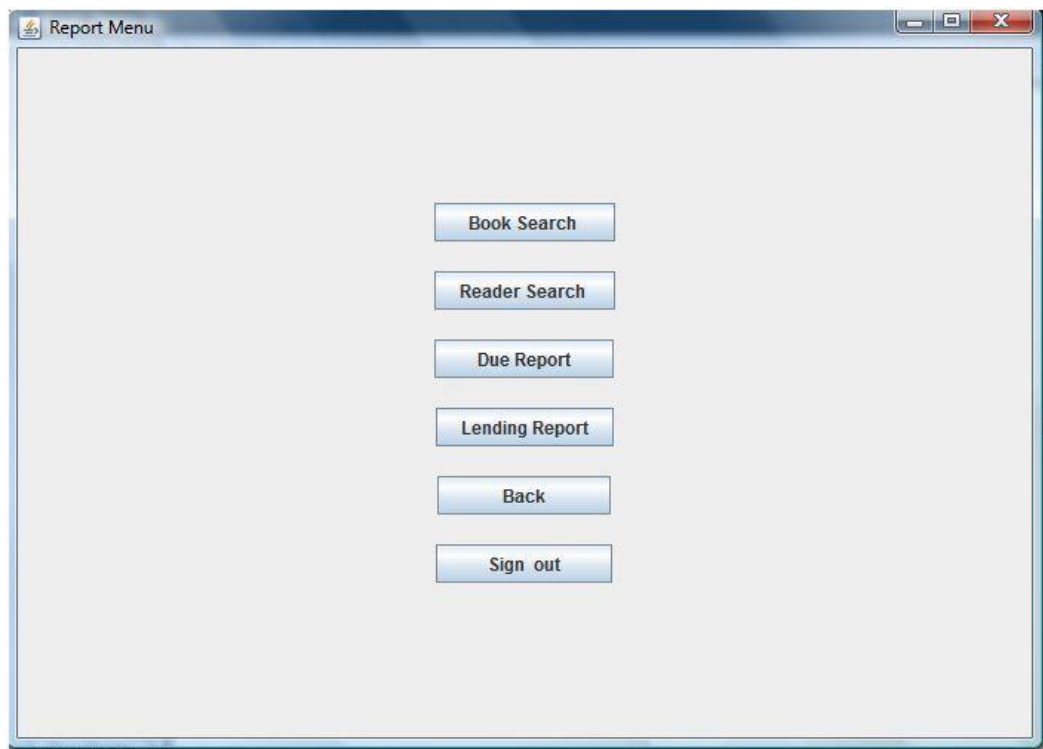
A 'Message' dialog box is displayed with the text: "SORRY BUT OLD NEW PASSWORD AND YOUR ACTUAL PASSWORD ARE NOT THE SAME". The dialog has an 'OK' button.

The 'Remove User' window contains the following fields and controls:

- Login ID:
- User Name:
- Password:
- Category:
- Remark:
- Buttons: Remove, Back, Sign out

A 'Message' dialog box is displayed with the text: "SORRY BUT A REMARK IS REQUIRED AS TO WHY THE USER IS BEING DELETED". The dialog has an 'OK' button.

Reports



Blue: Terminal Window - Prog

Options

| ID | FIRST NAME | LAST NAME | STANDARD | DIVISION | ROLL NO |
|----|------------|-----------|----------|----------|---------|
| 2  | Hrutvik    | Joshi     | 10       | A        | 14      |
| 3  | Shaunak    | Padhye    | 10       | A        | 32      |
| 4  | Dheer      | Furia     | 10       | B        | 11      |
| 1  | Avigyan    | Dasgupta  | 10       | A        | 12      |
| 5  | Abhiroop   | Roy       | 10       | A        | 4       |
| 7  | Hareet     | Singh     | 10       | A        | 13      |
| 6  | Abhimanyu  | Thakur    | 10       | A        | 2       |

Reader Search

First Name      Last Name      Standard

Search      Back      Sign out

Blue: Terminal Window - Prog

Options

| ID | FIRST NAME | LAST NAME | STANDARD | DIVISION | ROLL NO |
|----|------------|-----------|----------|----------|---------|
| 2  | Hrutvik    | Joshi     | 10       | A        | 14      |
| 3  | Shaunak    | Padhye    | 10       | A        | 32      |
| 4  | Dheer      | Furia     | 10       | B        | 11      |
| 1  | Avigyan    | Dasgupta  | 10       | A        | 12      |
| 5  | Abhiroop   | Roy       | 10       | A        | 4       |
| 7  | Hareet     | Singh     | 10       | A        | 13      |
| 6  | Abhimanyu  | Thakur    | 10       | A        | 2       |

| ID | FIRST NAME | LAST NAME | STANDARD | DIVISION | ROLL NO |
|----|------------|-----------|----------|----------|---------|
| 2  | Hrutvik    | Joshi     | 10       | A        | 14      |

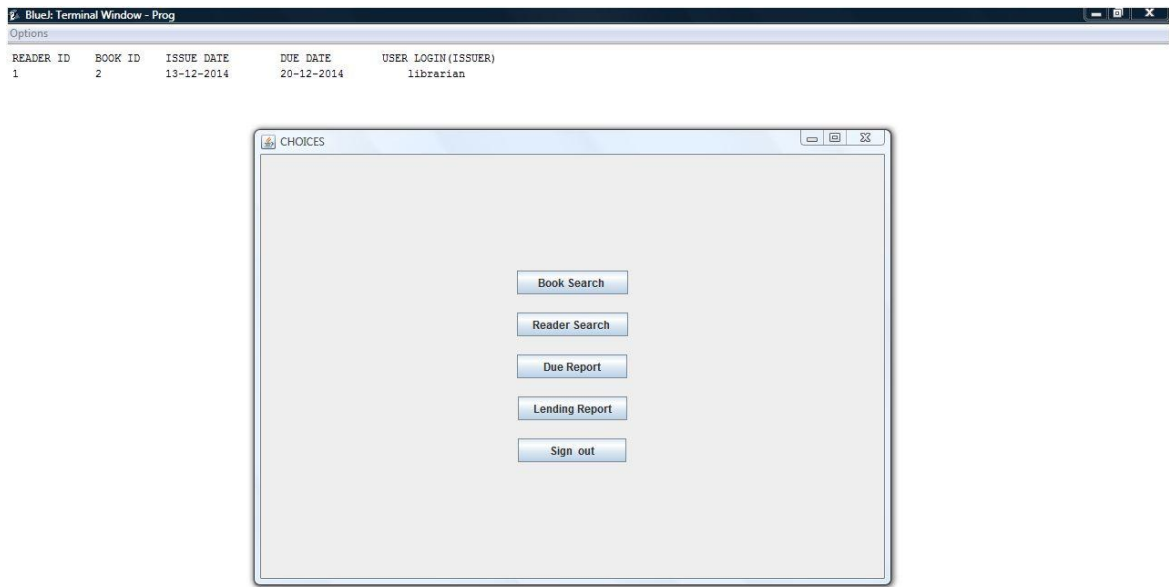
Reader Search

First Name      Last Name      Standard

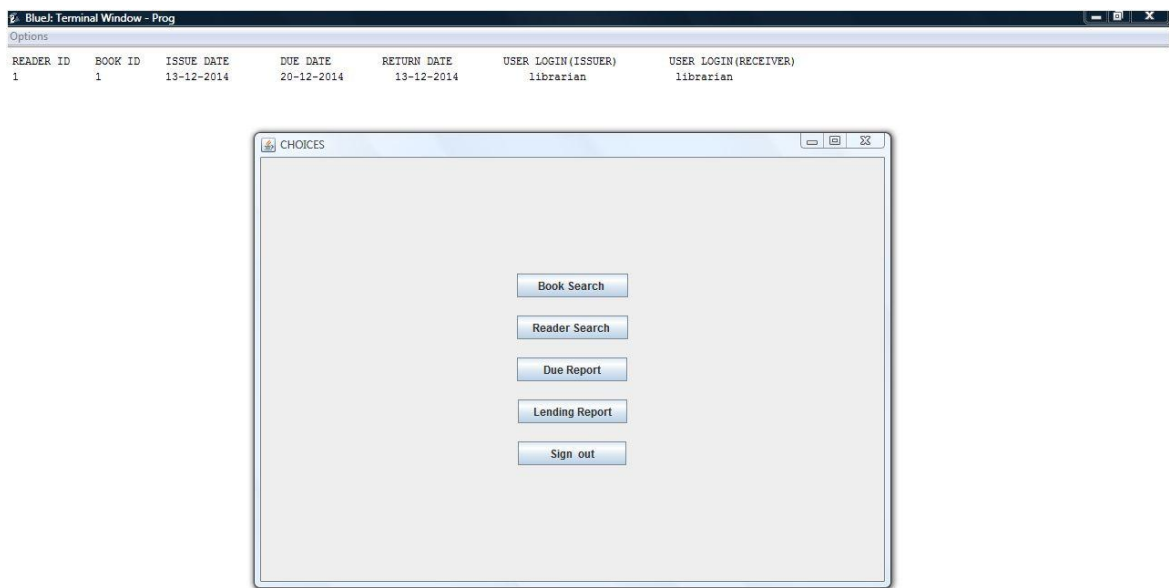
     Joshi     

Search      Back      Sign out

# Due Report



# Lending Report

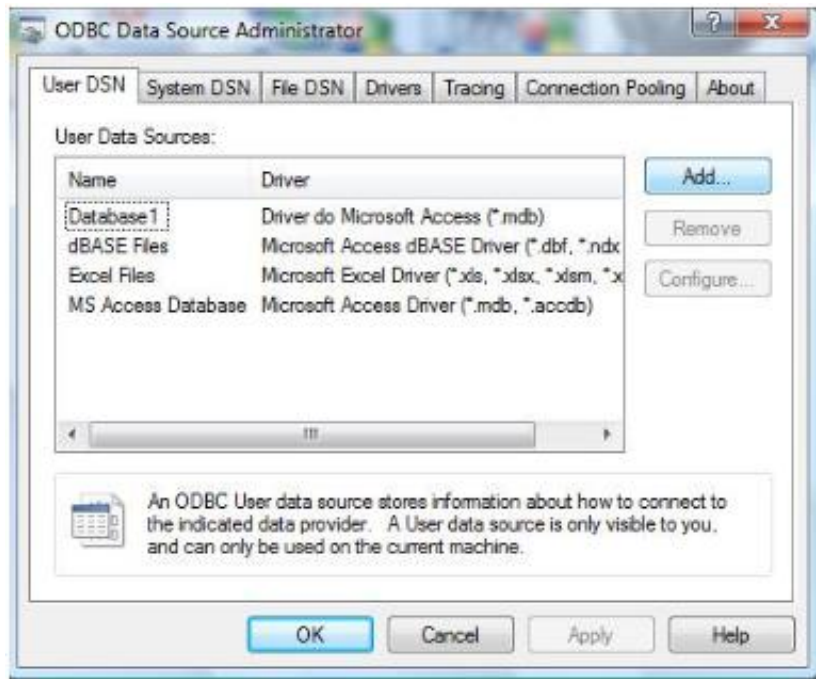


## Installation Procedure

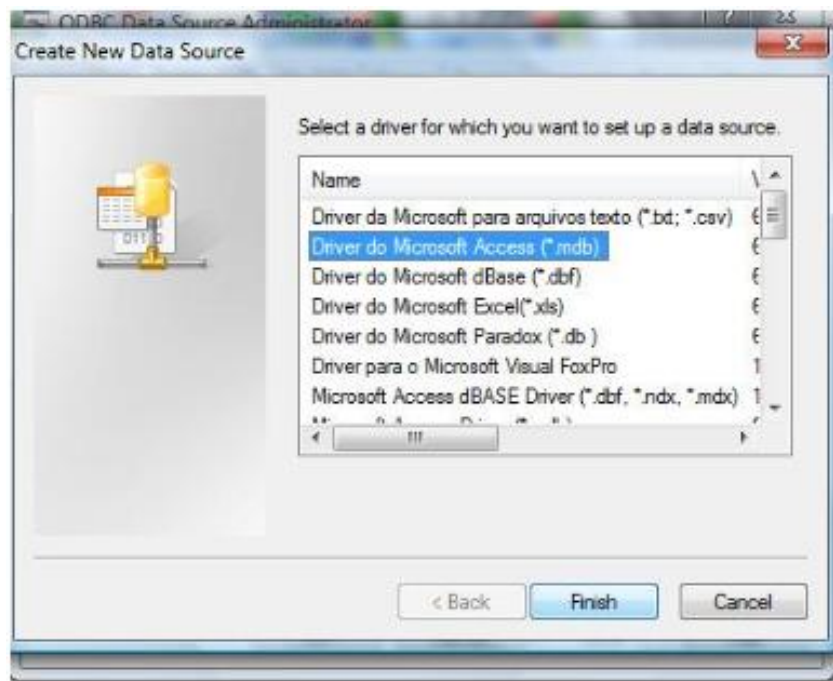
Step 1: Need to Copy the Database (VVSLibDB.MDB) from CD (Folder DB) to C:\VVSLib\DB

Step 2: Need to setup ODBC

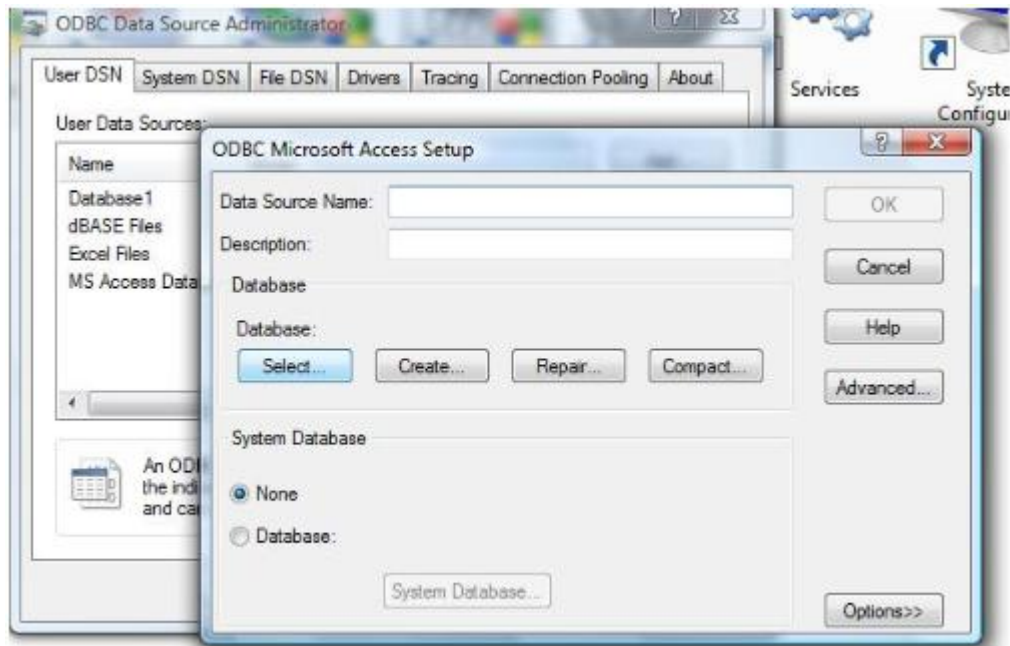
Control Panel – Admin Tool --- Data Source (ODBC)



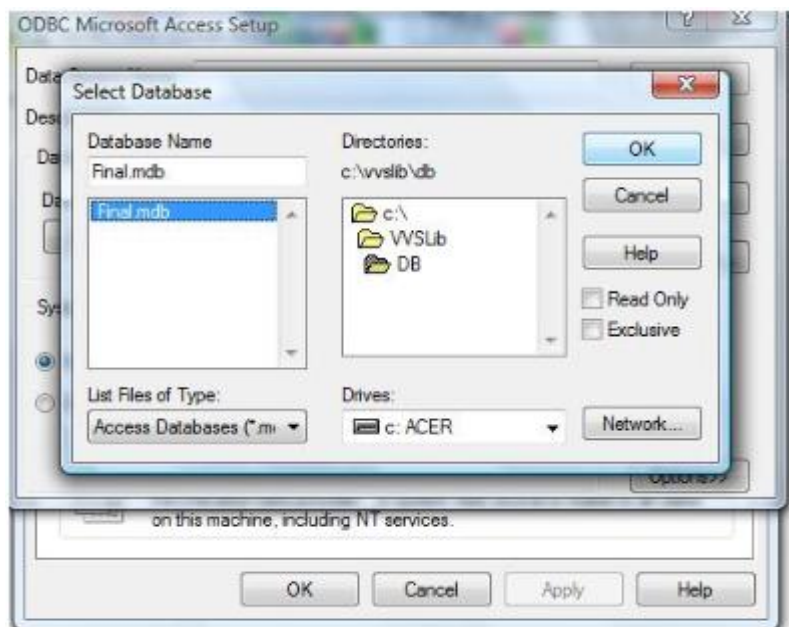
Click on “Add”



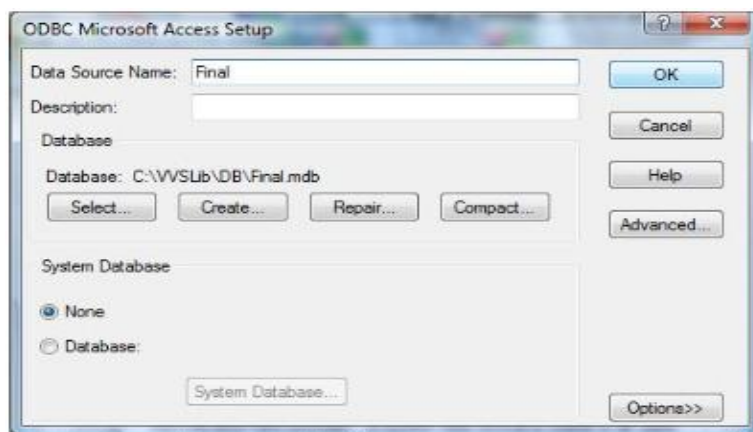
Driver do Microsoft Access (\*.mdb) --- > Finish



Select Database



Select "Final.mdb" from C:\VVSlib\DB and give the Data Source Name "Final", exactly this word.



Complete the activity by Clicking “OK”

Step 3: Need to Copy the Program from CD (Folder Prog) to C:\VVSLib\Prog

Step 4: Check the permission (attribute) of the Database file, It should not be in Read Only Mode.

Step 5: Start BlueJ

Step 6: Open Project “Prog”

Step 7: Only “admin” login will work with password “ADM”

Step 8: Change the Password of “Admin” account, immediately after login

Step 9: Create users with type Librarian to start transaction



## Operation Manual / User Guide

### How to Start

- a. Start the application with “admin” login [Specified in Installation Procedure Section]
- b. Create login for Librarian (User Account with Type “Librarian”)
- c. Logout from the system
- d. Re-login as Librarian User
- e. Create / update Books detail (Add Books Option)
- f. Create / update Reader Detail (Add Reader Option)
- g. Capture transaction of book using “Book Issue” / “Book Return” options
- h. If case any Reader is no more active, modify the “Reader Master”
- i. In case any book is not in readable condition, modify the “Book Delete”
- j. In case book is lost, capture the transaction in “Book Delete”
- k. In case of any Delete Option use “Remark” field to capture detail

### How to Create User

- a. Start the application with “Admin” login [Specified in Installation Procedure Section]
- b. Go to Maintenance Menu then “New User”

### How to Update Book Master

- a. Start the application, and login as Librarian (user with Librarian Type)
- b. Go to Maintenance Menu then “NewBook”

### How to Update Reader Master

- a. Start the application, and login as Librarian (user with Librarian Type)
- b. Go to Maintenance Menu then “New Reader”

### How to Capture Book Issue

- a. Start the application, and login as Librarian (user with Librarian Type)
- b. Go to Transaction Menu then “Book Issue”

### How to Capture Book Return

- a. Start the application, and login as Librarian (user with Librarian Type)
- b. Go to Transaction Menu then “Book Return”

### How to Capture “Book Lost”

- a. Start the application, and login as Librarian (user with Librarian Type)
- b. Go to Master Menu then “Delete Book”

### **How to Change Self Password**

- a. Start the application
- b. Go to Maintenance Menu then “Change Password”

### **How to Change The Password For Others**

- a. Start the application, and login as Adm
- b. Go to Maintenance Menu then “Change Password ”

## Conclusion

Although efforts were made to incorporate all the possible and required features still there are lot of scope of improvements remain. Here, basic requirements were mostly common and regular transaction along with necessary inputs our Librarian (and obviously with necessary guidance from respected Nilima Madam).

In conclusion section I'm specifying the untouched featured so that, all these points can be taken up in future to make this application much more robust.

1. Basic security of database (Anyone can open the database and can modify the data)
2. Back up & Restore options of data
3. Option to Capture "Fine" (Financial Transaction)
4. Configuration facility (Default Return Date, Number of books one can take at a time etc.)
5. Roll number can go up to 100 (hard coded)
6. Roll No. Of Readers handled through Drop Down to avoid the number validity Check
7. User Login uniqueness handled in coding, database constraints were not used.
8. Copy Book Option (for Multiple Copies of the same book with different Book Id No.) is not available
9. Purging of Old data
10. Proper installation program
11. Inbuilt validation to refrain Admin to delete "Admin" account
12. Count of number of variable can be reduced by code optimization
13. Releasing the space for memory variable also can be taken up
14. Online Help