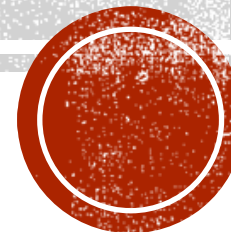


LIBRARY SYSTEM

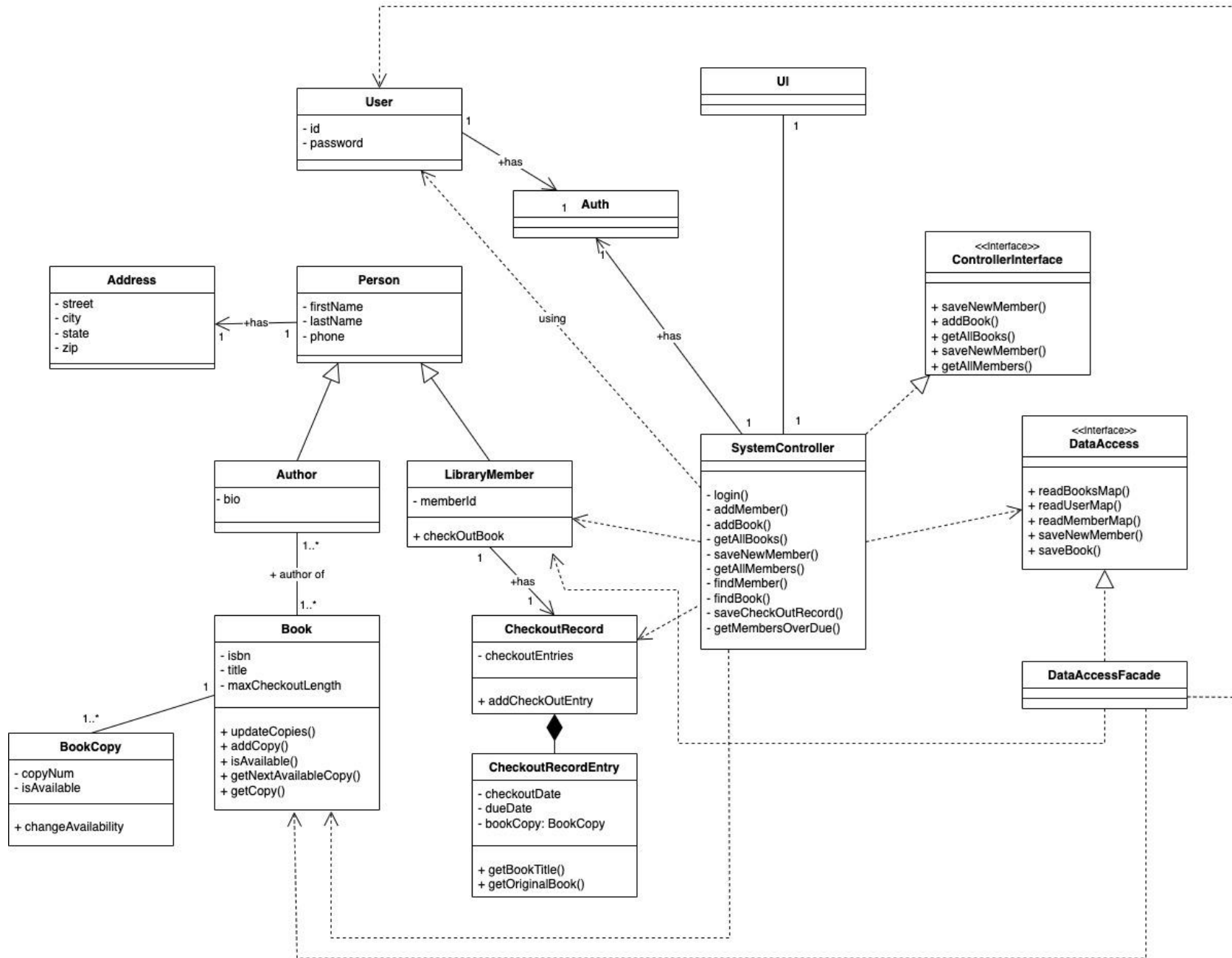


MEMBER

KAMAL GURAGAYIN
SADHANA KHADKA
ARUN MANDAL
JINXU ZHENG
PRATIMA SHAH

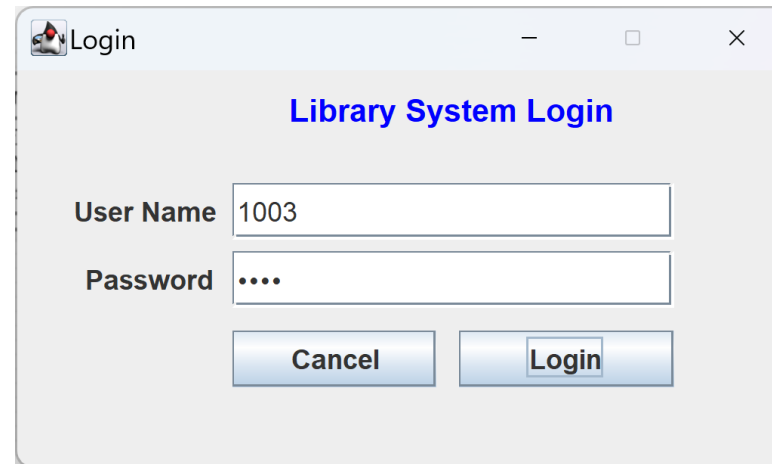


FINAL CLASS DIAGRAM



1. Login

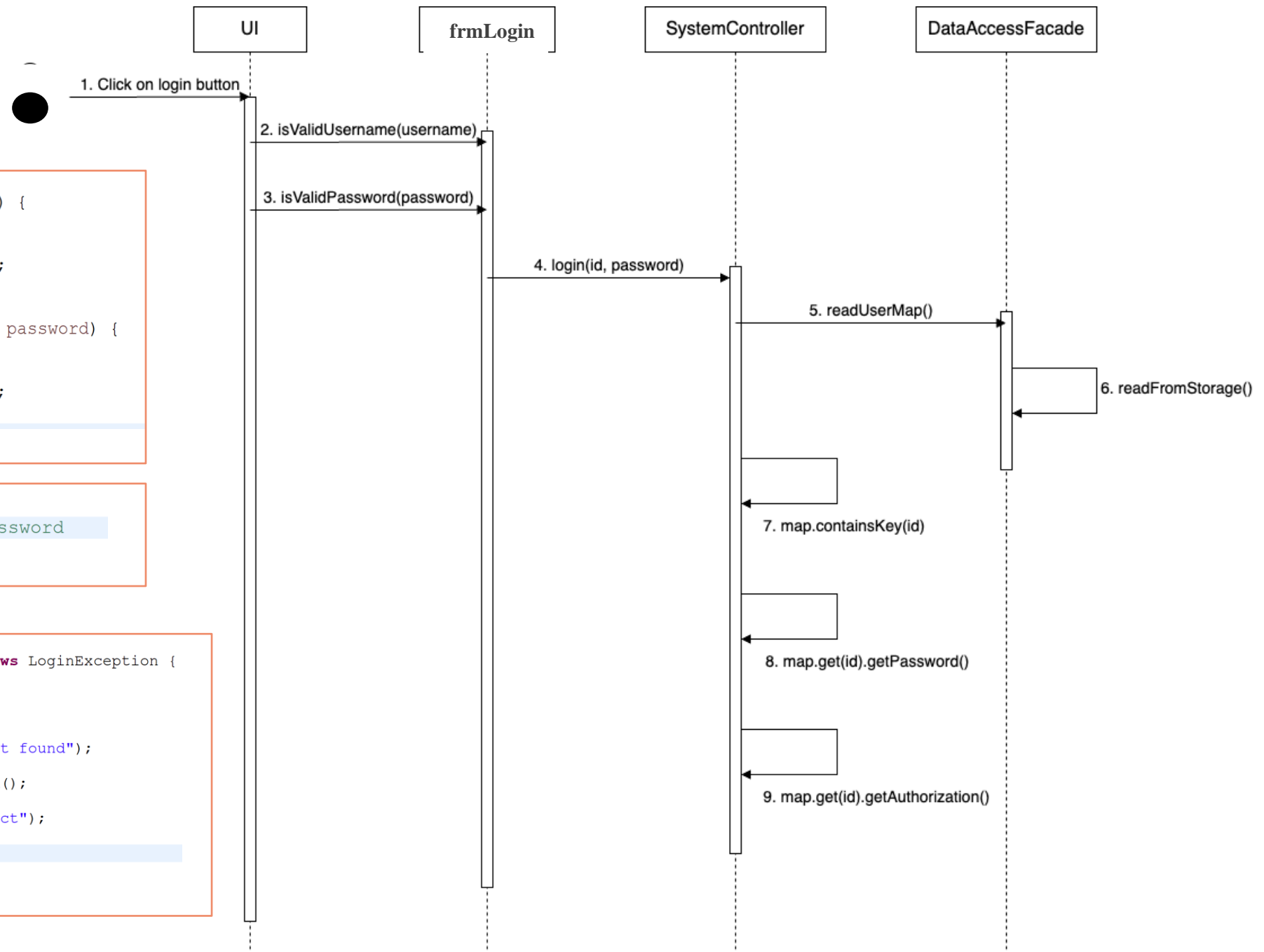
- The first screen a user can see while using this application is login, which requests ID and password. When the login button is clicked, the ID is looked up in the data store. If this ID can be found, and if the password for this ID matches the password submitted, the authorization level is returned. Authorization levels are LIBRARIAN, ADMIN, and BOTH. If login is successful, UI features are made available according to the authorization level of the user.



The screenshot shows a standard Windows-style dialog box titled "Login". Inside the dialog, the text "Library System Login" is displayed in blue. Below this, there are two input fields: "User Name" with the value "1003" and "Password" with masked characters "....". At the bottom of the dialog, there are two buttons: "Cancel" and "Login".



1. Login Sequence Diagram

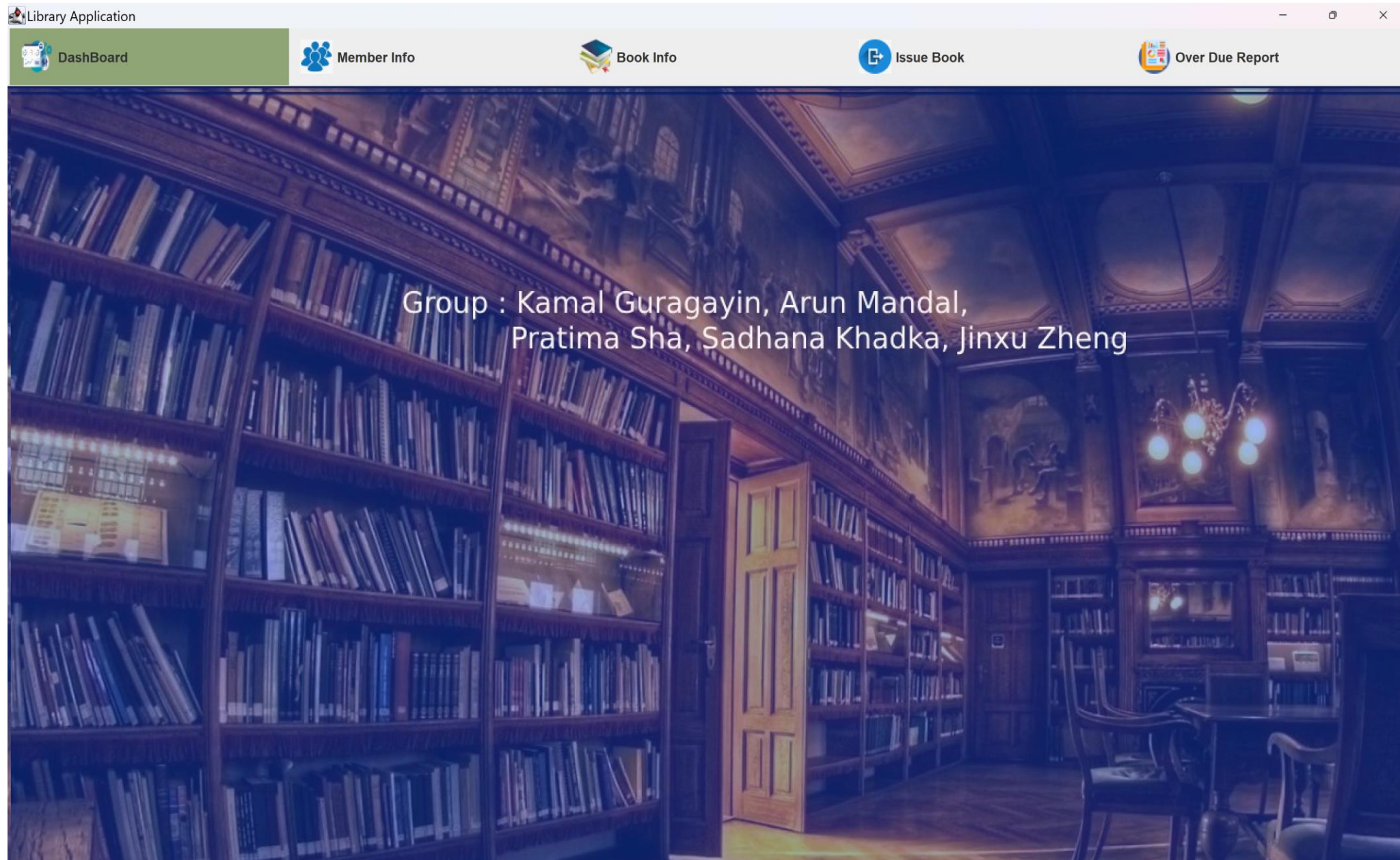


```
static boolean isValidUsername(String username) {  
    String regex = "[a-z0-9_]{3,16}$";  
    Pattern pattern = Pattern.compile(regex);  
    return pattern.matcher(username).matches();  
}  
  
public static boolean isValidPassword(String password) {  
    String regex = "[0-9a-zA-Z]+$";  
    Pattern pattern = Pattern.compile(regex);  
    return pattern.matcher(password).matches();  
}
```

```
//For the validation of the ID and Password  
systemController.login(id, password);
```

```
public void login(String id, String password) throws LoginException {  
    DataAccess da = new DataAccessFacade();  
    HashMap<String, User> map = da.readUserMap();  
    if(!map.containsKey(id)) {  
        throw new LoginException("ID " + id + " not found");  
    }  
    String passwordFound = map.get(id).getPassword();  
    if(!passwordFound.equals(password)) {  
        throw new LoginException("Password incorrect");  
    }  
    currentAuth = map.get(id).getAuthorization();  
}
```


USER INTERFACE OF PROJECT



```
public void changeSelectMenu(String menuName) {

    MenuType mt = MenuType.getMenuType(menuName);

    if (SystemController.currentAuth.equals(Auth.ADMIN)) {
        if (mt == MenuType.CHECKOUT) {
            JOptionPane.showMessageDialog(this, "You Role is " + SystemController.currentAuth +
                ". Sorry you are not allow to access!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
    } else if (SystemController.currentAuth.equals(Auth.LIBRARIAN)) {
        if (mt == MenuType.BOOK) {
            JOptionPane.showMessageDialog(this, "Your Role is " + SystemController.currentAuth +
                " Sorry you are not allow to access!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
    }

    for (JLabel menu : menus) {

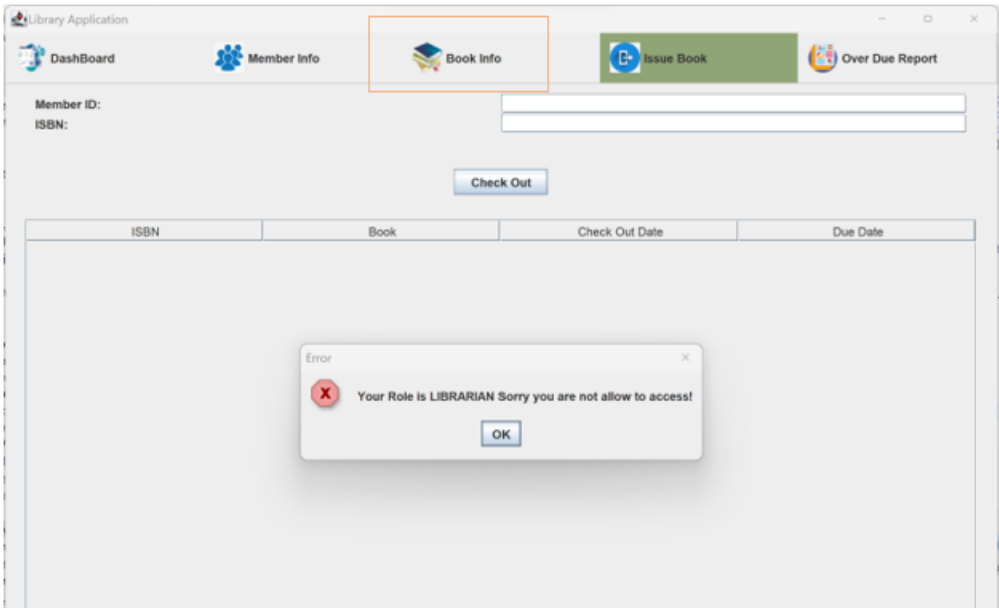
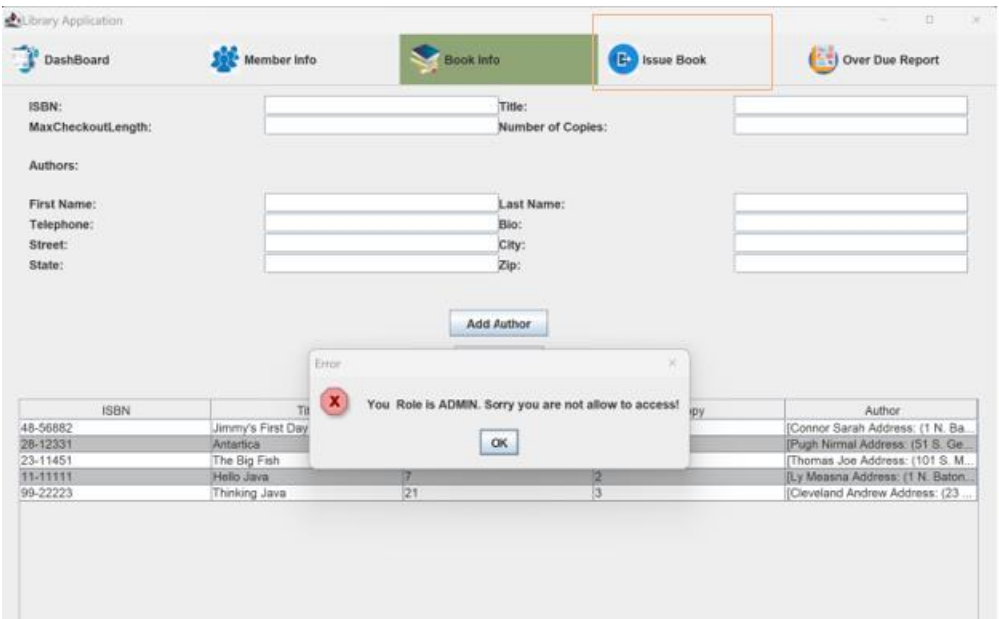
        if (menu.getText() != menuName) {
            menu.setBackground(mainPanel.getBackground());
        } else {
            menu.setBackground(Colors.primary);
        }

        CardLayout cl = (CardLayout) (cardPanel.getLayout());
        cl.show(cardPanel, menuName);

    }

    CardLayout cl = (CardLayout) (cardPanel.getLayout());
    cl.show(cardPanel, menuName);
}
```

- 1. If a user has logged in through Librarian credentials, then he is not allowed to operate **BOOK info** related operation
- 2. If a user has logged in through Admin credentials, then he is not allowed for **Issue BOOK** info related operation



2. Member Info

Member info is the module where user can view members registered with the library management system. Here user has the facility to add and edit user details. Users can also search members id by their member id.

When an Administrator selects the option to add a new member, then he is presented with a form with fields: member id, first name, last name, street, city, state, zip, telephone number. After the data is entered and submitted, it is persisted using the persistence mechanism for this project.




```

private void addMember() {

    Address address = new Address(
        this.streetField.getText(),
        this.cityField.getText(),
        this.stateField.getText(),
        this.zipField.getText()
    );

    LibraryMember newMember = new LibraryMember(
        this.memberIdField.getText(),
        this.firstNameField.getText(),
        this.lastNameField.getText(), this.telephoneField.getText(), address, null
    );

    this.libraryMembers.add(newMember);
    this.addMemberToTable(newMember);

    this.ci.saveNewMember(newMember);
    this.clearFormFields();
    JOptionPane.showMessageDialog(this, "Member added", "Successful", JOptionPane.INFORMATION_MESSAGE);
    this.memberIdField.requestFocusInWindow();

}

```

DashBoard
 Member Info
 Book Info
 Issue Book
 Over Due Report

Search by ID:

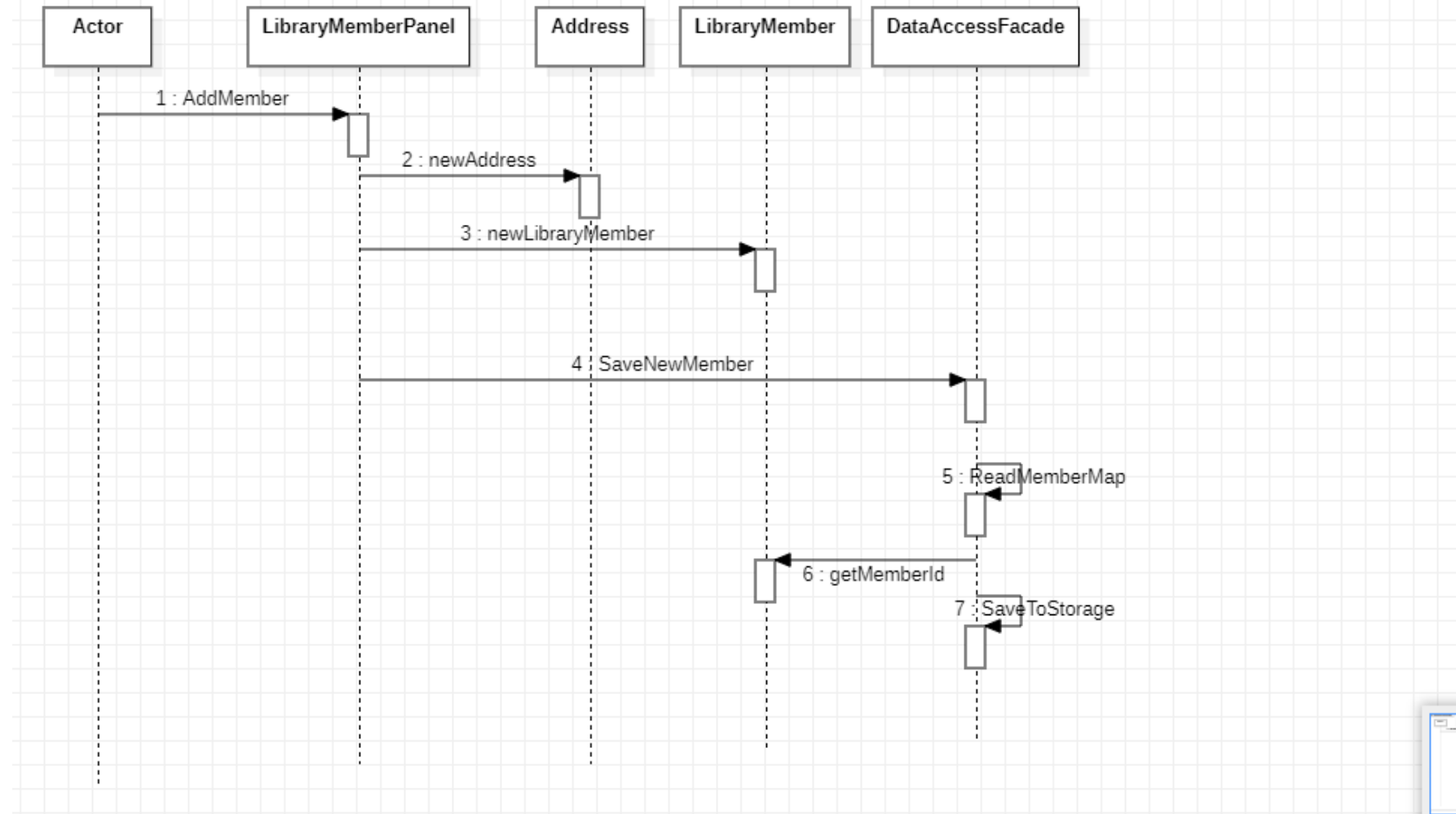
Member ID:
 First Name:
 Last Name:
 Street:
 City:
 State:
 ZIP:
 Telephone:

Member ID	First Name	Last Name	Telephone
1004	Ricardo	Montalbahn	641-472-2871
1003	Sarah	Eagleton	451-234-8811
1002	Drews	Stevens	702
1001	Andy	Rogers	641-223-2211
2000	La	Lazi	641-223-2211

Checked Books

ISBN	Title	Checkout Date	Due Date
------	-------	---------------	----------

SequenceDiagram1



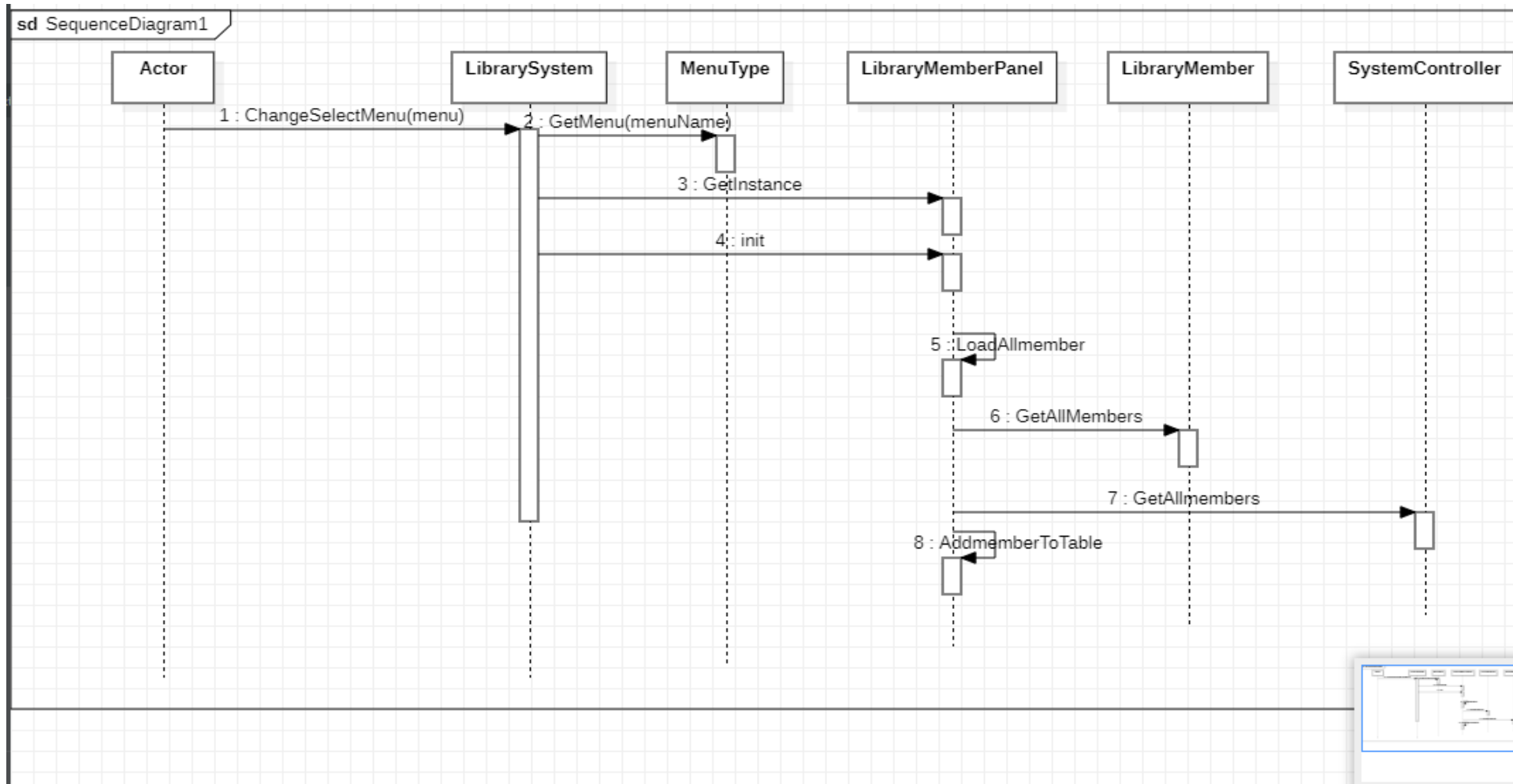
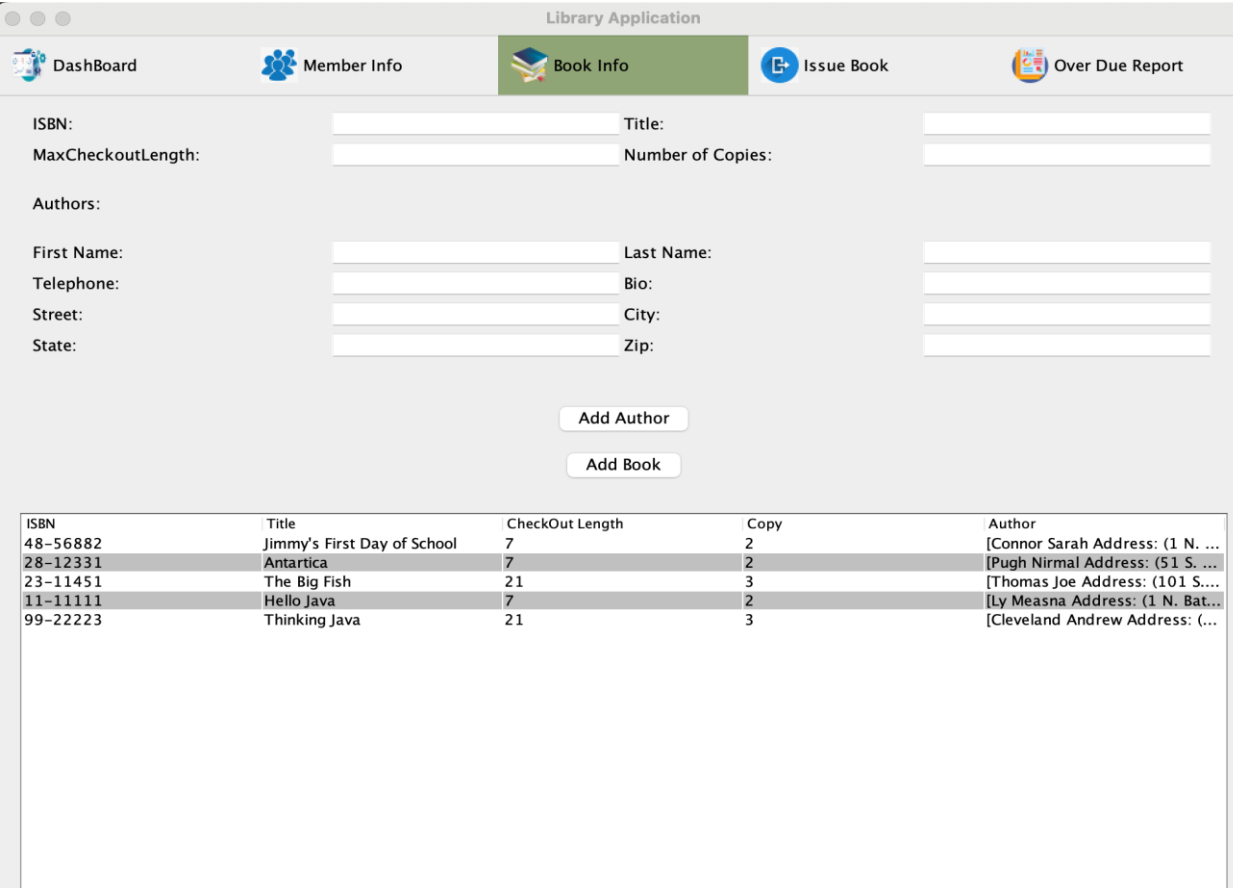


Fig- View Member Sequence diagram



BOOK INFO

The book info window by default shows the list of all books that is in the system.



ISBN	Title	CheckOut Length	Copy	Author
48-56882	Jimmy's First Day of School	7	2	[Connor Sarah Address: (1 N. ...
28-12331	Antartica	7	2	[Pugh Nirmal Address: (51 S. ...
23-11451	The Big Fish	21	3	[Thomas Joe Address: (101 S....
11-11111	Hello Java	7	2	[Ly Measna Address: (1 N. Bat...
99-22223	Thinking Java	21	3	[Cleveland Andrew Address: (...



🔗 If User tries to add book without adding a single author ,it displays error.

Library Application

Dashboard Member Info **Book Info** Issue Book Over Due Report

ISBN: 12-3456 Title: other side of the midnight
MaxCheckoutLength: 12 Number of Copies: 2

Authors:

First Name: Sydney Last Name: Sheldon
Telephone: 12345 Bio: new work times best selling author
Street: 4th City: fairfield
State: ia Zip: 52557

Add Author

Error

Please input author information!

OK

ISBN	Title	Author
48-56882	Jimmy's First Day of	[Connor Sarah Address: (1 N. ...
28-12331	Antartica	[Pugh Nirmal Address: (51 S. ...
23-11451	The Big Fish	[Thomas Joe Address: (101 S. ...
11-11111	Hello Java	[Ly Measna Address: (1 N. Bat...
99-22223	Thinking Java	[Cleveland Andrew Address: (...]



❌ If any of the input fields for adding book are empty, error prompts on screen and book cannot be added

Library Application

Dashboard Member Info **Book Info** Issue Book Over Due Report

ISBN: Title:

MaxCheckoutLength: Number of Copies:

Authors:

First Name: Last Name:

Telephone: Bio:

Street: City:

State: Zip:

[Sheldon Sydney Address: (4th, fairfield, 52557), aaa isabel Address: (3rd, www, 32456)]

Error

Please input isbn!

OK

ISBN	Title			Author
48-56882	Jimmy's First Day of School			[Connor Sarah Address: (1 N. ...
28-12331	Antartica			[Pugh Nirmal Address: (51 S. ...
23-11451	The Big Fish			[Thomas Joe Address: (101 S....
11-11111	Hello Java			[Ly Measna Address: (1 N. Bat...
12-3456	other side of the midnight	12	1	[Sheldon Sydney Address: (4th...
99-22223	Thinking Java	21	3	[Cleveland Andrew Address: (...



❌ If any of the input fields for adding author are empty, error prompts on screen and author cannot be added



Library Application

Dashboard Member Info **Book Info** Issue Book Over Due Report

ISBN: 11-222 Title: hello world
MaxCheckoutLength: 12 Number of Copies: 12

Authors:

First Name: Last Name:
Telephone: Bio:
Street: City:
State: Zip:

Error
Please input first name!
OK

ISBN	Title	Author
48-56882	Jimmy's First Day of School	[Connor Sarah Address: (1 N. ...
28-12331	Antartica	[Pugh Nirmal Address: (51 S. ...
22-3333	beginning with java	[Sheldon Sydney Address: (4th...
23-11451	The Big Fish	[Thomas Joe Address: (101 S....
11-11111	Hello Java	[Ly Measna Address: (1 N. Bat...
12-3456	other side of the midnight	[Sheldon Sydney Address: (4th...
99-22223	Thinking Java	[Cleveland Andrew Address: (...



 The user can add multiple authors for a book and add a book

Library Application

DashBoard

Member Info

Book Info

Issue Book

Over Due Report

ISBN:

Title:

MaxCheckoutLength:

Number of Copies:

Authors:

First Name:

Last Name:

Telephone:

Bio:

Street:

City:

State:

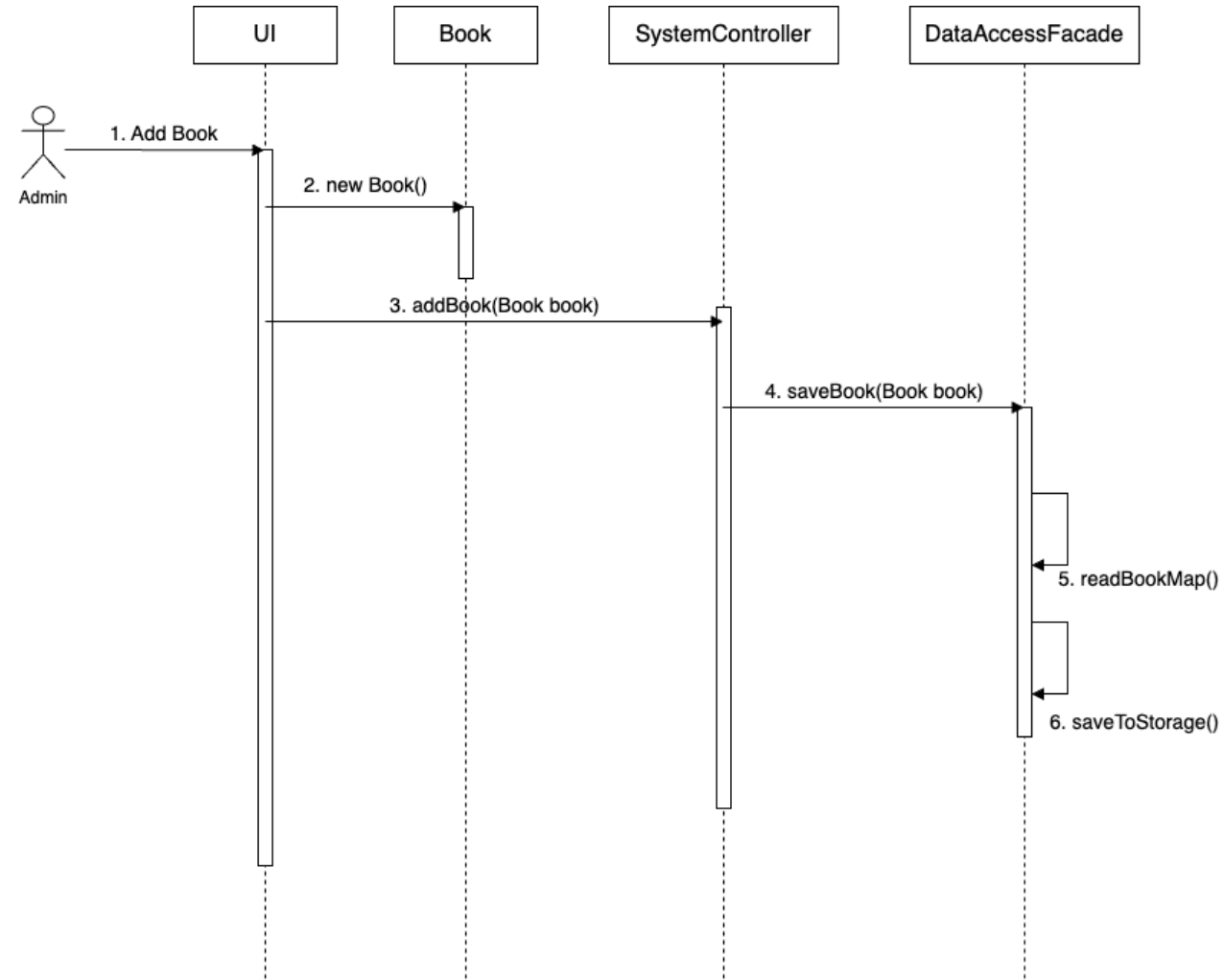
Zip:

[Sheldon Sydney Address: (4th, fairfield, 52557), aaa isabel Address: (3rd, www, 32456)]

Add Author

Add Book

ISBN	Title	CheckOut Length	Copy	Author
48-56882	Jimmy's First Day of School	7	2	[Connor Sarah Address: (1 N. ...
28-12331	Antartica	7	2	[Pugh Nirmal Address: (51 S. ...
23-11451	The Big Fish	21	3	[Thomas Joe Address: (101 S....
11-11111	Hello Java	7	2	[Ly Measna Address: (1 N. Bat...
12-3456	other side of the midnight	12	1	[Sheldon Sydney Address: (4th...
99-22223	Thinking Java	21	3	[Cleveland Andrew Address: (...



4. checkout

- A librarian can enter in a form a member ID and an ISBN number for a book and ask the system whether the requested item is available for checkout. If ID is not found, the system will display a message indicating this, or if the requested book is not found or if none of the copies of the book are available, the system will return a message indicating that the item is not available. If both member ID and book ID are found and a copy is available, a new checkout record entry is created, containing the copy of the requested book and the checkout date and due date. This checkout entry is then added to the member's checkout record. The copy that is checked out is marked as unavailable. The updated checkout record is displayed on the UI and is also persisted. The display of the checkout record uses a JavaFX TableView, with all cells of the table read-only.




```

private void initEvent() {
    checkOutButton.addActionListener(e -> {

        String memberId = memberIdField.getText().trim();
        String isbn = isbnField.getText().trim();

        if (memberId.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please input member id!", "Error", JOptionPane.ERROR_MESSAGE);
            memberIdField.requestFocus();
            return;
        }

        if (isbn.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please input isbn!", "Error", JOptionPane.ERROR_MESSAGE);
            isbnField.requestFocus();
            return;
        }

        LibraryMember member = ci.findMember(memberId);
        Book book = ci.findBook(isbn);

        if (member == null) {
            JOptionPane.showMessageDialog(this, "Member not found!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        if (book == null) {
            JOptionPane.showMessageDialog(this, "Book not found!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        if (!book.isAvailable()) {
            JOptionPane.showMessageDialog(this, "Book is not available!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        CheckoutRecord record = member.checkOutBook(book);

        if (record == null) {

```

DashBoard
 Member Info
 Book Info
 Issue Book
 Over Due Report

Member ID:
ISBN:

ISBN	Book	Check Out Date	Due Date
99-22223	Thinking Java	07/07/2023	07/28/2023

Error

This member hasn't returned this book from last time!

