

# APPLICATION PROGRAMMING IN JAVA PROJECT REPORT

Library management system

“COMPILEMILE” TEAM

Github URL:

<https://github.com/iuthub/group-project-compilemile>

Team:

Yarkinov Ulugbek U1910140 sec 003

Kattakhodjaev Boiskhon U1910171 sec 004

Keldiyarov Bekzod U1910142 sec 004

Musaev Azamat U1910177 sec 004

Zaynutdinov Zufarbek U1910139 sec 003

INHA UNIVERSITY IN TASHKENT



Submission date:2021-01-10

# Table of contents

<b>Contents .....</b>	<b>2</b>
<b>USER ACCOUNTS .....</b>	<b>3</b>
<b>DESCRIPTIION OF IMPLEMENTED FUNCTIONALITY .....</b>	<b>3</b>
<b>ENTITY-RELATIONSHIP DIAGRAM.....</b>	<b>13</b>
<b>UML DIAGRAMS.....</b>	<b>16</b>

# User Accounts

Account	Username	Password
Admin	Ulugbek	123
Librarian	Zufar	456
Student	Bois	789

## Description of Implemented Functionality

For our project we implemented JavaFX and Derby

We have three active roles that could use our product: admin, librarian and student

### **Login package – Boiskhon, Ulugbek contributed**

1. In class LogInRepository we connected to our database using method connectDB()
2. In class LoginController we implemented the method logIn() which checks correctness and existing of typed data.
3. Then, depending on the state of method logIn() we introduce function handleLogIn(), which will navigate us through the windows of admin, librarian and student (getting the role string).

Log In

— □ ×

# Log In

Enter your Username and Password

Ulugbek

...

Log In

Admin Window

— □ ×

Librarians Students Books

First Name Last Name Username

Zufarbek Zufarbekov Zufar

ID:

First Name:

Last Name:

Username:

Password:

first name

last name

username

password

Add

Edit

Delete

Log out

## Admin package – Bekzod and Zufarbek collaborated

Objections of administrators

1. In class Controller, method `handleLogOut()` is responsible for sending you on login page

Log out

2. Methods `addlib()`, `addStud()`, are created to generate new librarians/students and also there included a verification for distinct data that is controlled and prevented

Admin Window

Librarians Students Books

First Name	Last Name	Username
Zufarbek	Zufarbekov	Zufar
Ben	Mac	BM

ID:

First Name:

Last Name:

Username:

Password:

Ben Mac BM ...

Add Edit Delete

Log out

3. Methods `deleteLibrarian()` and `deleteStudent()` remove users.

Admin Window

Librarians Students Books

First Name	Last Name	Username
Zufarbek	Zufarbekov	Zufar

ID: 4

First Name: Benny

Last Name: Mac

Username: BMac

Password: 123

Benny Mac BMac ...

Add Edit Delete

Log out

4. Methods `updateLibrarian()` and `updateStudent()` alter primarily given data to the modified ones and make a verification for distinct elements.

Admin Window

Librarians Students Books

First Name	Last Name	Username
Zufarbek	Zufarbekov	Zufar
Benny	Mac	BMac

ID: 4

First Name: Ben

Last Name: Mac

Username: BM

Password: 123

Benny Mac BMac

Add Edit Delete

Log out

5. Methods `addBook()`, `updateBook()` and `deleteBook()` are responsible for editing and deleting data related to books.

Admin Window

Librarians Students Books

Find book by: Author Name search...

Title	Author	ISBN	Publish Date
OOP	12321312	Deitel	21.12.20
OOP	12321312	Deitel	21.12.20
Calculus	Markov	100123	03.12.2015
Death to Sm...	Prentice Gris...	767639980-7	13.06.2020
Gunfighters	Beltran Brett...	718976933-1	27.07.2020
Meatballs III	Cecile Kamin...	284457806-3	03.03.2019
Babbitt	Bryn Canny	769084988-1	14.04.2019
xcs	xcacac	acas	21.12.20

ID: 8

Title: xcs

Author: xcacac

ISBN: acas

Publish Date: 21.12.20

Taken By: 1

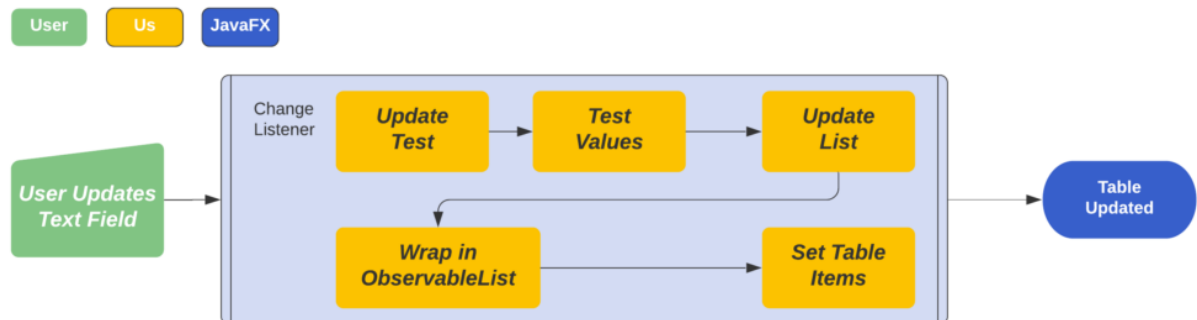
Java S.Abdulvaliev 13333333 21.12.20

Add Edit Delete

Log out

6. Method `initialize` fills the tables of librarians, students, and books by data taken from data base and adding them to `ObservableLists` of librarians, students, and books

7. Method filter; dynamically filtering content can be achieved by setting a Predicate on a FilteredList. This predicate should be updated as user input changes the search criteria. Placing a listener on the search box TextField is a common way to achieve this.



8. Methods displayLibrarian/Student/BookDetails() made for interactive purposes to display all the data as a column on our application

The screenshot shows the 'Admin Window' application. It has three tabs: 'Librarians', 'Students', and 'Books'. The 'Books' tab is selected. Below the tabs, there is a search bar with a dropdown menu set to 'Author Name' and a text input field containing 'search...'. Below the search bar is a table with the following data:

Title	Author	ISBN	Publish Date
OOP	12321312	Deitel	21.12.20
OOP	12321312	Deitel	21.12.20
Calculus	Markov	100123	03.12.2015
Death to Sm...	Prentice Gris...	767639980-7	13.06.2020
Gunfighters	Beltran Brett...	718976933-1	27.07.2020
Meatballs III	Cecile Kamin...	284457806-3	03.03.2019
Babbitt	Bryn Canny	769084988-1	14.04.2019
Java	13333333	S.Abdualiev	21.12.20

To the right of the table, the details for the selected book 'Gunfighters' are displayed:

ID: 5  
 Title: Gunfighters  
 Author: Beltran Bretton  
 ISBN: 718976933-1  
 Publish Date: 27.07.2020  
 Taken By: 1

At the bottom of the window, there are four input fields for adding a new book: 'Java', 'S.Abdualiev', '13333333', and '21.12.20'. Below these fields are three buttons: 'Add', 'Edit', and 'Delete'. In the bottom right corner, there is a 'Log out' button.

## Librarian package – Ulugbek, Azamat collaborated

1. In class Controller of Librarian package, method handleLogOut() is responsible for sending you on login page
2. Method addStud(), is created to generate new students and also there included a verification for distinct data that is controlled and prevented. Screenshot is the same as in admin because they share the same functionality.
3. Method deleteStudent() removes students from local database. Screenshot is the same as in admin because they share the same functionality
4. Methods updateStudent() alters primarily given data to the modified one and makes a verification for distinct elements. Screenshot is the same as in admin because they share the same functionality.
5. Methods addBook(), updateBook() and deleteBook() are responsible for editing and deleting data related to books. Screenshot is the same as in admin because they share the same functionality.
6. Method initialize fills the tables of students and books by data taken from data base and adding them to ObservableLists of students and books
7. Method filter; dynamically filtering content can be achieved by setting a Predicate on a FilteredList. This predicate should be updated as user input changes the search criteria. Placing a listener on the search box TextField is a common way to achieve this.

It allows the user to filter books by its author and title

Librarian Window

Students Books

Find book by: ✓ Title

Title	Author Name	ISBN	Publish Date	
Academic E...	Williams	500520	12.01.2009	

ID: 6

Title: *Meatballs III*

Author: *Cecile Kaminski*

ISBN: 284457806-3

Publish Date: 03.03.2019

Taken By: 1

Add Edit Delete  ...



8. Method `displayBookDetails()` made for interactive purposes to display all the data as a column on our JavaFX application. Screenshot is the same as in admin because they share the same functionality.
9. Method `displayStudentDetails()` allows the librarian to see all the details of a particular student. Screenshot is the same as in admin because they share the same functionality.
10. Method `handleGiveBook()` allows the librarian issue a particular book to a Student by inserting his username and password that could be checked in `displayStudentDetails()`

The screenshot shows a JavaFX window titled "Librarian Window". It has two tabs: "Students" and "Books". The "Books" tab is active. Below the tabs, there is a search bar labeled "Find book by:" with a dropdown menu set to "Title" and a text input field containing "search...".

Title	Author	ISBN	Publish Date
Academic E...	Williams	500520	12.01.2009
OOP2	Deitel	100123	03.12.2015
Calculus	Markov	100123	03.12.2015
Death to S...	Prentice Gri...	767639980-7	13.06.2020
Gunfighters	Beltran Brett...	718976933-1	27.07.2020
Meatballs III	Cecile Kami...	284457806-3	03.03.2019
Babbitt	Bryn Canny	769084988-1	14.04.2019
This Christm...	Corliss Whitl...	499415320-5	25.12.2020
Machine Gu...	Evangelina ...	759082658-8	26.02.2020
Hellbenders	Ignace Tribb...	169754199-2	23.04.2020

To the right of the table, the details for the selected book "Meatballs III" are displayed:

- ID: 6
- Title: Meatballs III
- Author: Cecile Kaminski
- ISBN: 284457806-3
- Publish Date: 03.03.2019
- Taken By: 1

Below the table, there are four input fields for "title", "author", "isbn", and "publish date". Below these fields are four buttons: "Add", "Edit", "Delete", and "Bois". To the right of the "Bois" button is a button labeled "Give chosen book". In the bottom right corner, there is a "Log out" button.

## Student package – Bekzod and Zufarbek contributed

1. In class Controller of Student package, method `handleLogOut()` is responsible for sending you on login page
2. Method `initialize()` fills the tables of books and `myBooks` (represents the books taken by a particular student) and adding them to `ObservableLists` of books and `myBooks`

3. Method `displayBookDetails()` made for interactive purposes to display all the data as a column on our JavaFX application

The screenshot shows a JavaFX application window titled "Student Window". It has two tabs: "Books" and "My books". Under the "Books" tab, there is a search bar with a dropdown menu set to "Title" and a text input field labeled "search...". Below the search bar is a table with four columns: "Title", "Author", "ISBN", and "Publish". The table contains the following data:

Title	Author	ISBN	Publish
Academic English 3	Williams	500520	12.01.20
OOP2	Deitel	100123	03.12.20
Calculus	Markov	100123	03.12.20
Death to Smoochy	Prentice Grishenkov	767639980-7	13.06.20
Gunfighters	Beltran Bretton	718976933-1	27.07.20
Meatballs III	Cecile Kaminski	284457806-3	03.03.20
Babbitt	Bryn Canny	769084988-1	14.04.20
This Christmas	Corliss Whitloe	499415320-5	25.12.20
Machine Gun Preacher	Evangeline Millier	759082658-8	26.02.20
Hellbenders	Ignace Tribbeck	169754199-2	23.04.20

Below the table, there is a section for book details for the selected book, "Death to Smoochy". The details are displayed as follows:

ID: 4

Title: Death to Smoochy

Author: Prentice Grishenkov

ISBN: 767639980-7

Publish Date: 13.06.2020

Taken By: 1

At the bottom right of the window, there is a "Log out" button.

4. Method `DisplayMyBookDetails()` is designed to display particular Student's borrowed books.

The screenshot shows the same JavaFX application window titled "Student Window". The "My books" tab is selected. Below the tab, there is a table with three columns: "Title", "Author", and "ISBN". The table contains the following data:

Title	Author	ISBN
Meatballs III	Cecile Kami...	284457806-3

Below the table, there is a section for book details for the selected book, "Meatballs III". The details are displayed as follows:

ID: 6

Title: Meatballs III

Author: Cecile Kaminski

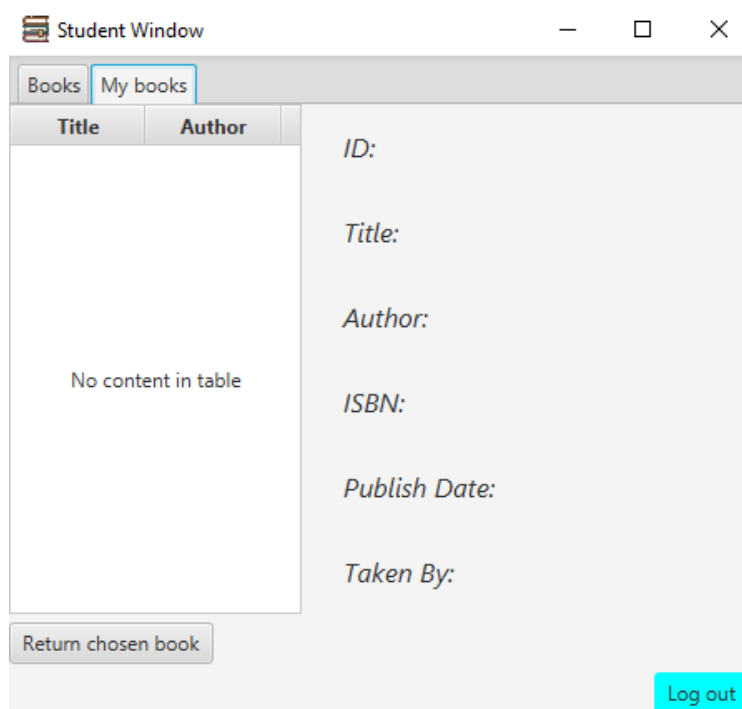
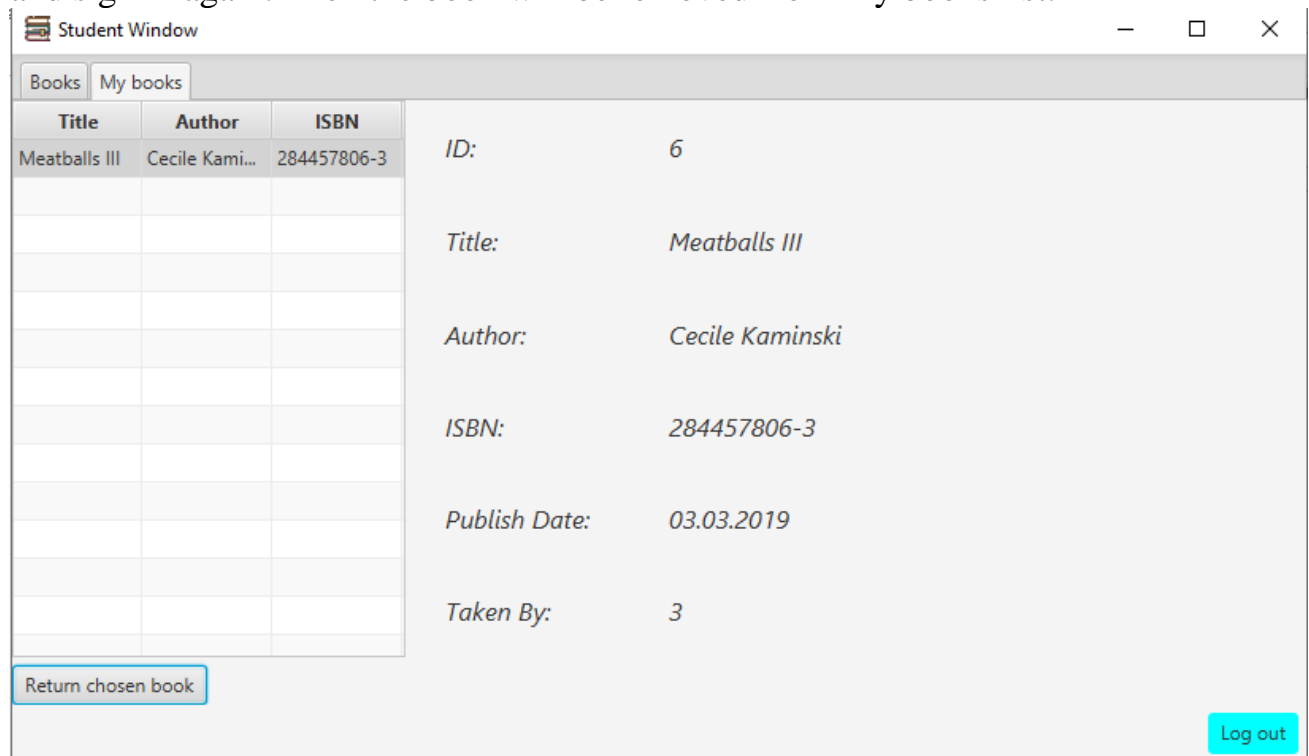
ISBN: 284457806-3

Publish Date: 03.03.2019

Taken By: 3

At the bottom left of the window, there is a "Return chosen book" button. At the bottom right, there is a "Log out" button.

5. Method filter; dynamically filtering content can be achieved by setting a Predicate on a FilteredList. This predicate should be updated as user input changes the search criteria. Placing a listener on the search box TextField is a common way to achieve this. It allows the user to filter books by its author and title. Screenshot is the same as in admin because they share the same functionality.
6. Method `handleReturnBook()` allows the Student to return issued books. However, books are not removed from the table instantly. You should log out and sign in again. Then the book will be removed from My books list.



## **ExtraClasses package - Ulugbek, Azamat, Boiskhon, Bekzod and Zufarbek collaborated**

### **Books class**

1. Setters and getters are created to initialize the data

### **User class**

1. Setters and getters are created to initialize the data

### **BookRepository class**

This class is used to connect with Data Base and set all necessary Queries such as retrieving the data by getBook() and getAllBooks methods, adding, and deleting books and also updating. In this class we used SIGNAL TONE design pattern to not allow to create the object of this class two times, because the data base reference must be created only one time

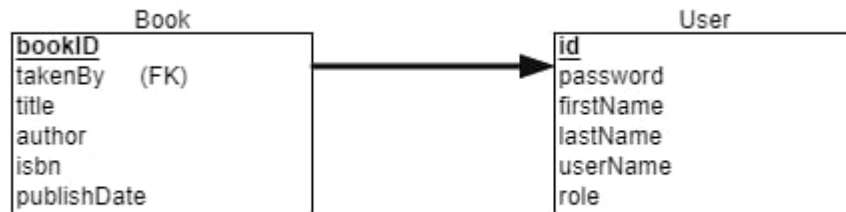
### **UserRepository class**

This class is used to connect with Data Base and set all necessary Queries such as retrieving the data by getAllLivrarians() and getAllLibrarians methods, adding, and deleting users and also updating. In this class we used SIGNAL TONE design pattern to not allow to create the object of this class two times, because the data base reference must be created only one time

## **Overall Experience**

Developing such program was very challenging experience for us because of number of reasons, starting from inability to collaborate together face-to-face and finishing by the lack of experience and knowledge in this sphere. Also it was difficult to organize all colleagues as there were winter holidays and nobody wanted to work these times. Also, very poor knowledge of the sphere of databases also impacted on us, as we were struggling to manage our own methods due to lack of necessary details.

# Entity Relationship diagram



In our database we have used only two tables, “User” and “Book” for storing data about users and books respectively. The table “User” is responsible for all the registered users in our application (Administrators, Librarians, Students). One of the benefits of designing a single table for Users is that it allows us to add as many “roles” as we want. For example, we can add a new role “System Administrator” in the row “role” and sign in to the application via `login()` method. We do not need to change our database by creating a new table for the new “roles”.

There are also some drawbacks. For example, if we want to add new fields in “User” table we should make changes for all users in our database.

In the Book table, the `takenBy` has the value of student’s id as a foreign key to scan the book that are assigned (borrowed) by a particular student. However, the `takenBy` column has a value ‘1’ by default and shows that the book is in a library, not taken by a student.

# UML diagram

