

**School of Computers and Information Engineering**

**STUDENT LEARNING ACTIVITY**

Fall Semester 2019

**Application Programming in JAVA**

***Library Management System***

Student Learning Assignment

Submitted by

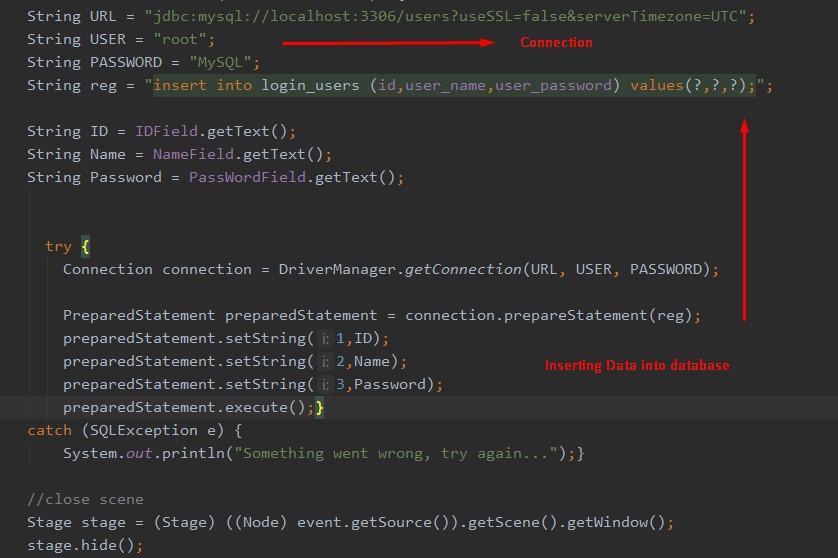
U1810017, Arifdjanov Sarvar (Group 1)

U1810014, Smirnova Ekaterina (Group 1)

U1810050, Gapurova Zarnigor (Group 1)

GitHub URL: <https://github.com/kattrine99/Library-project>

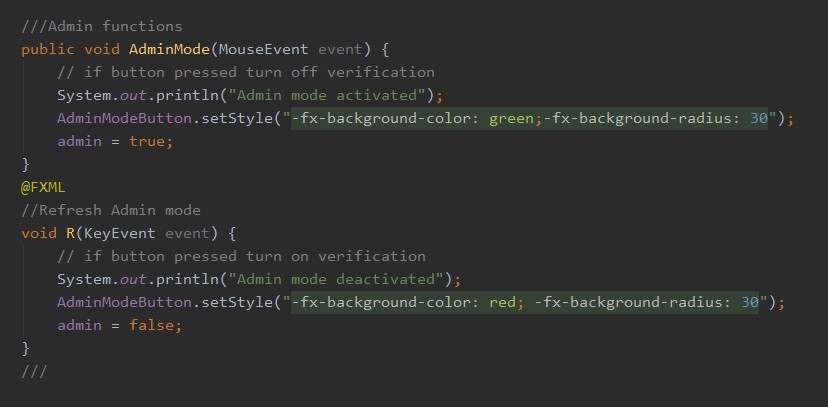
Description of Implemented Functionality



Inserting Data into Database

In order to insert data, there is used the “INSERT INTO”. This statement is used to insert new record in a table.

Within application our data was inserted into “ login\_users”.

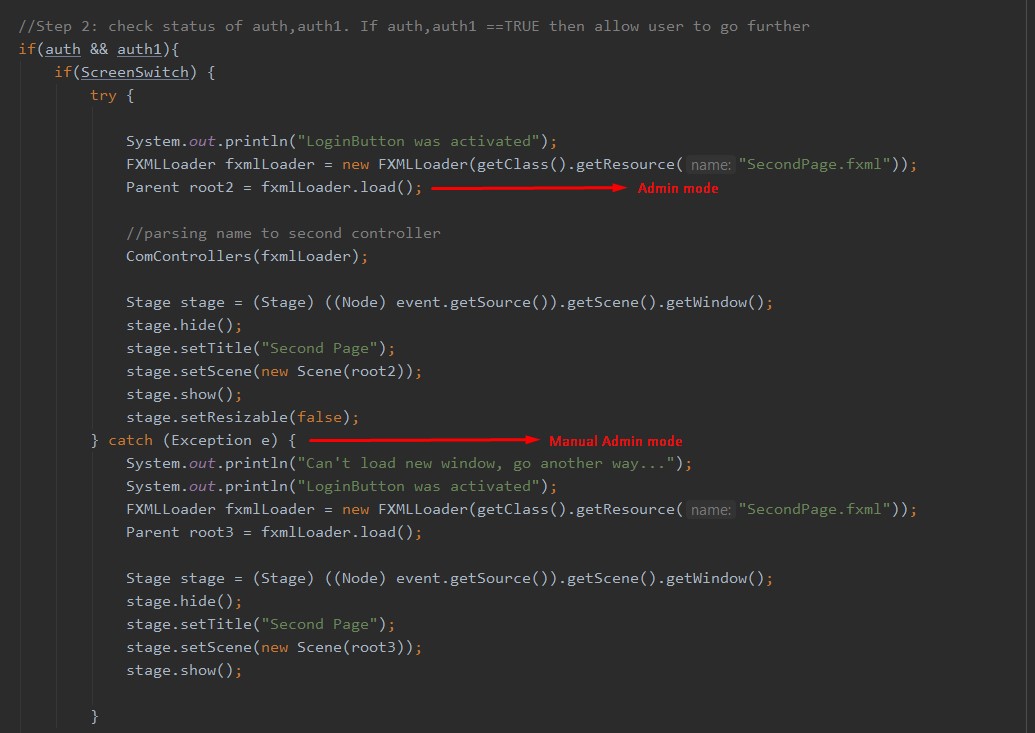


Admin Functions

This function controls:

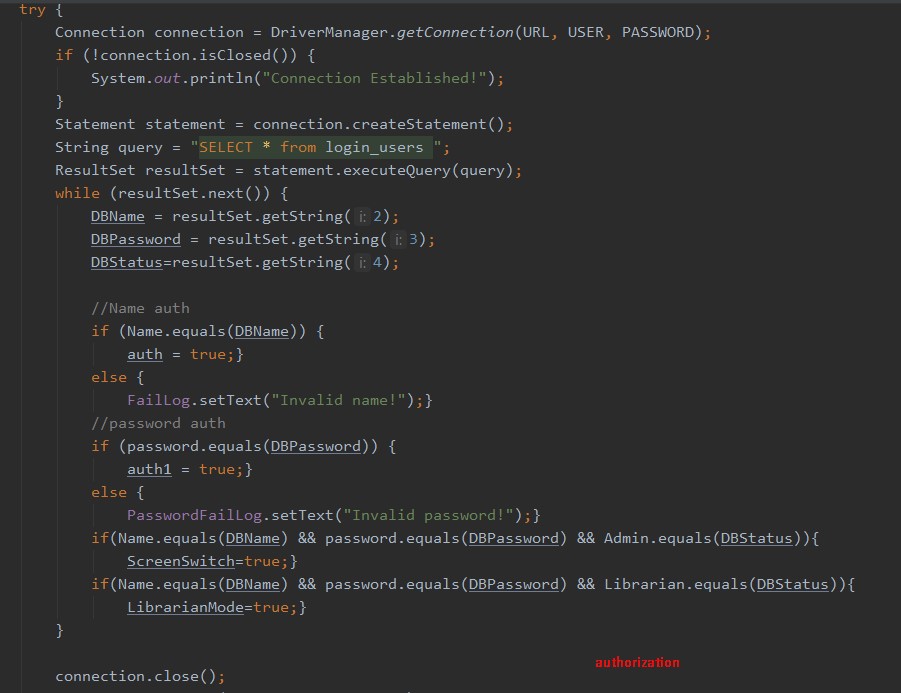
IF BUTTON pressed “Turn ON” – admin is activated.

IF BUTTON pressed “Turn OFF” – admin is deactivated.

****

Admin Mode

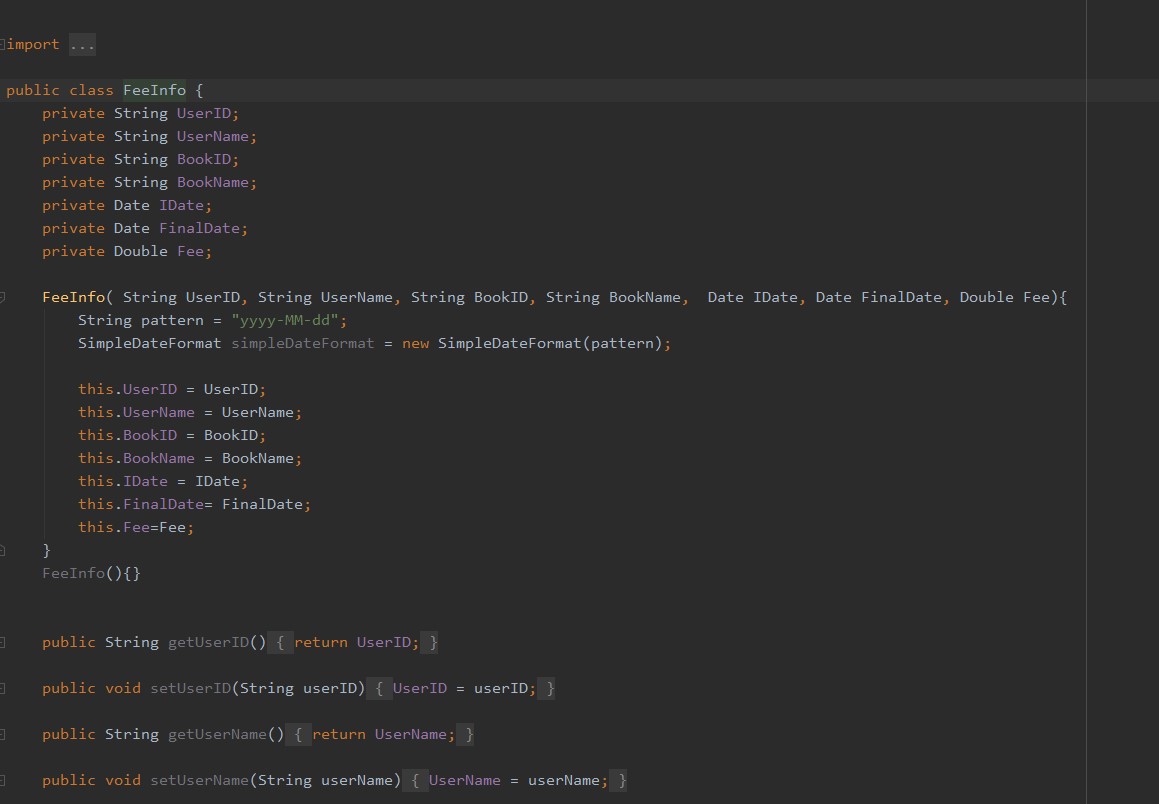
In this part of the program, the administrator status is checked . If everything fits, he can use his permissions.

****

Authorization

In this part we establish the connection with URL, USER and PASSWORD.

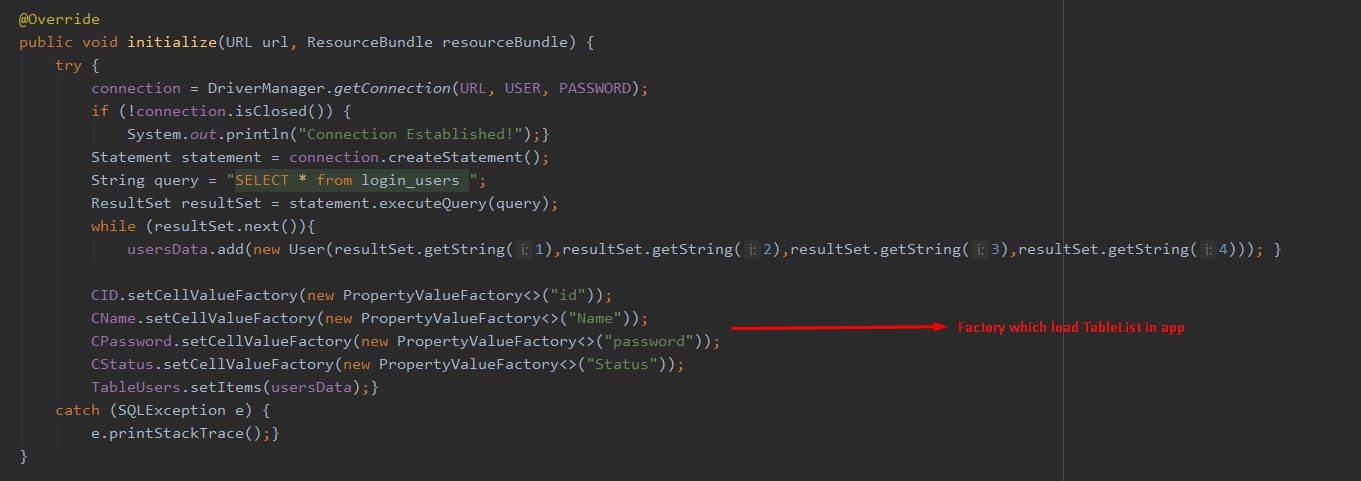
After that, there is a check and access to user data; name and password verification.

****

Fee Function

In this part of application, all DATA within “Class FeeInfo” is private because it is unique information of user.

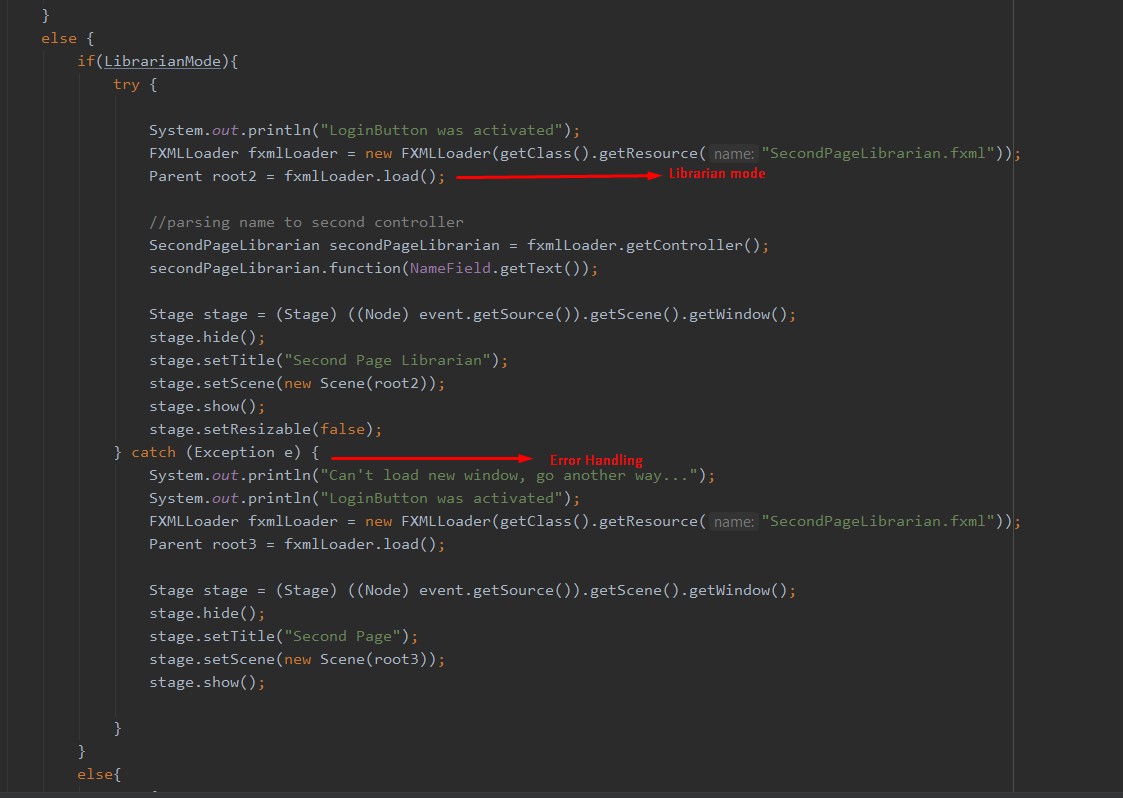
In this part students/professors book some books and write their name, ID and all information about book.

****

Initialization

The initialize method is called after all @FXML annotated members have been injected.

Then all information is loaded in application .

****

Librarian Mode

FXMLLoader automatically calls any suitably annotated no-argument initialize() method defined by the controller.

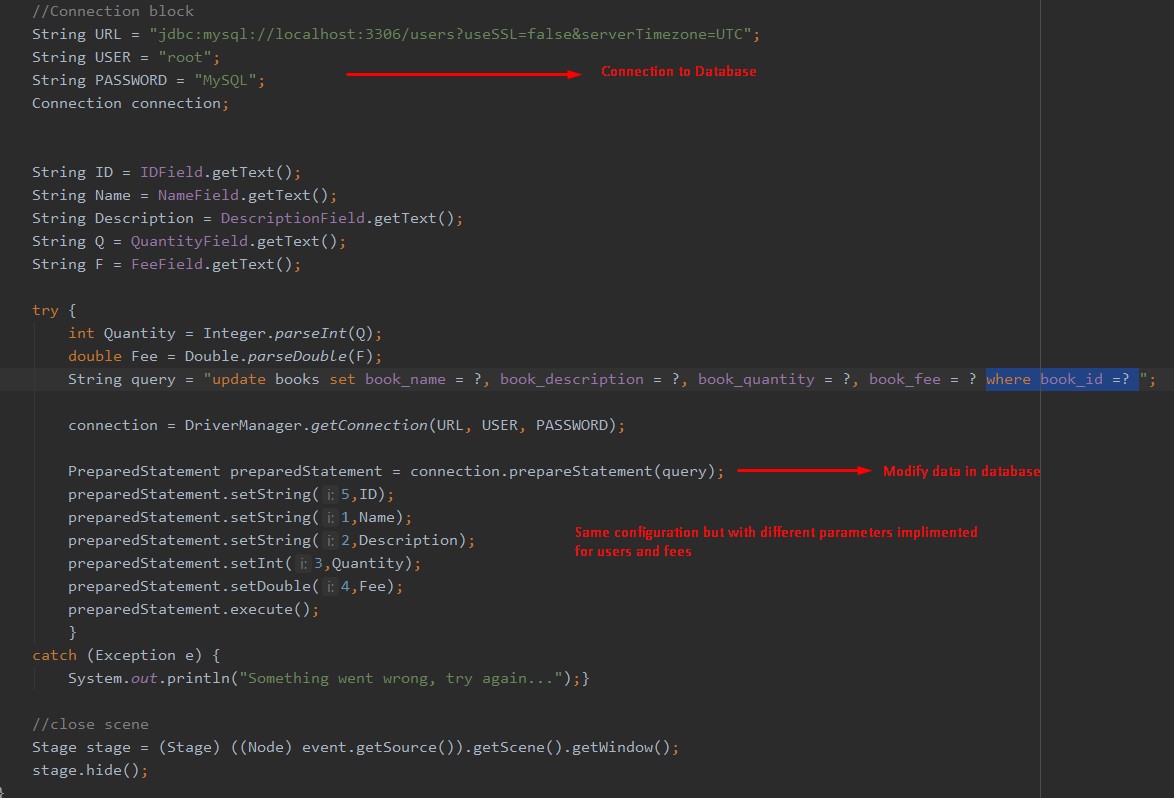
Furteremore, there is “Error Handling” which is included for another chance to access

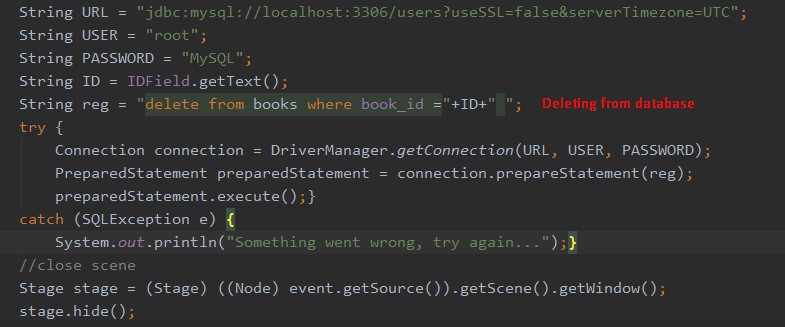
Connection to database

Un this part we are connecting block and also modify data in database.

A prepared statement is used to execute the same statement repeatedly with high efficiency. The prepared statement execution consists of two stages: prepare and execute.

In our case it is implemented for users and fees.

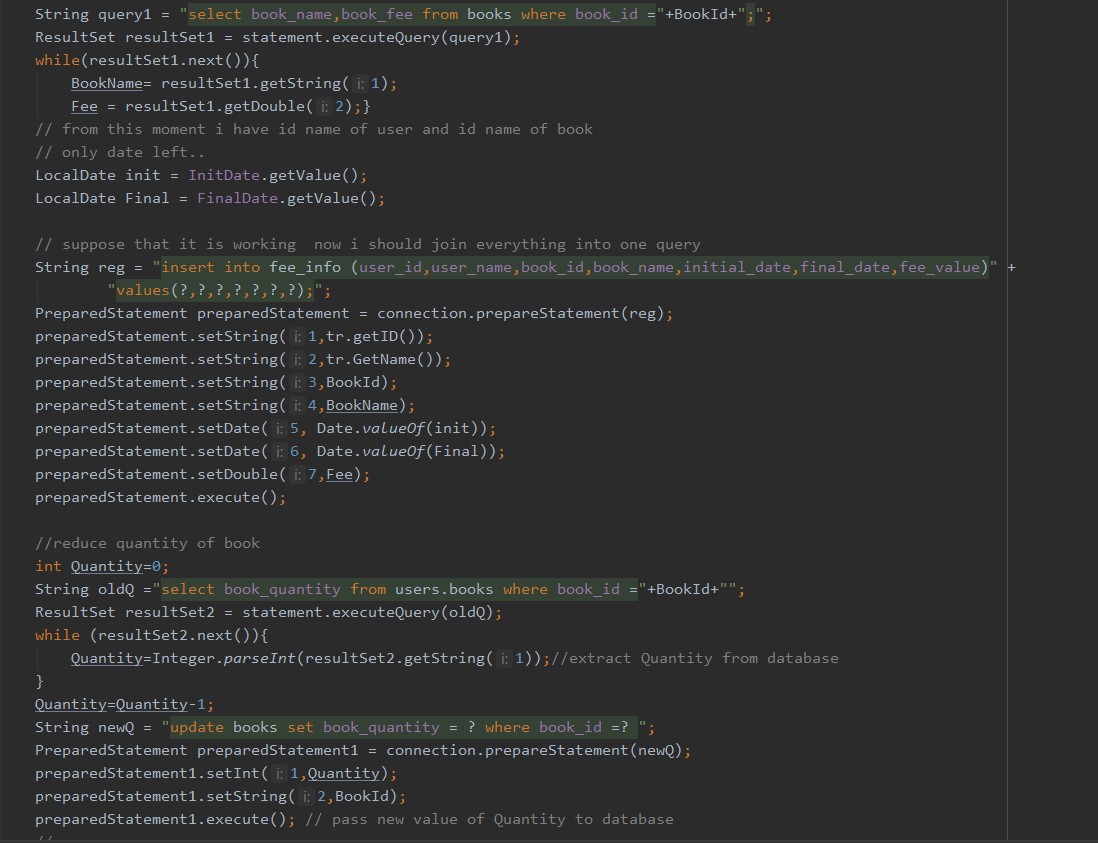
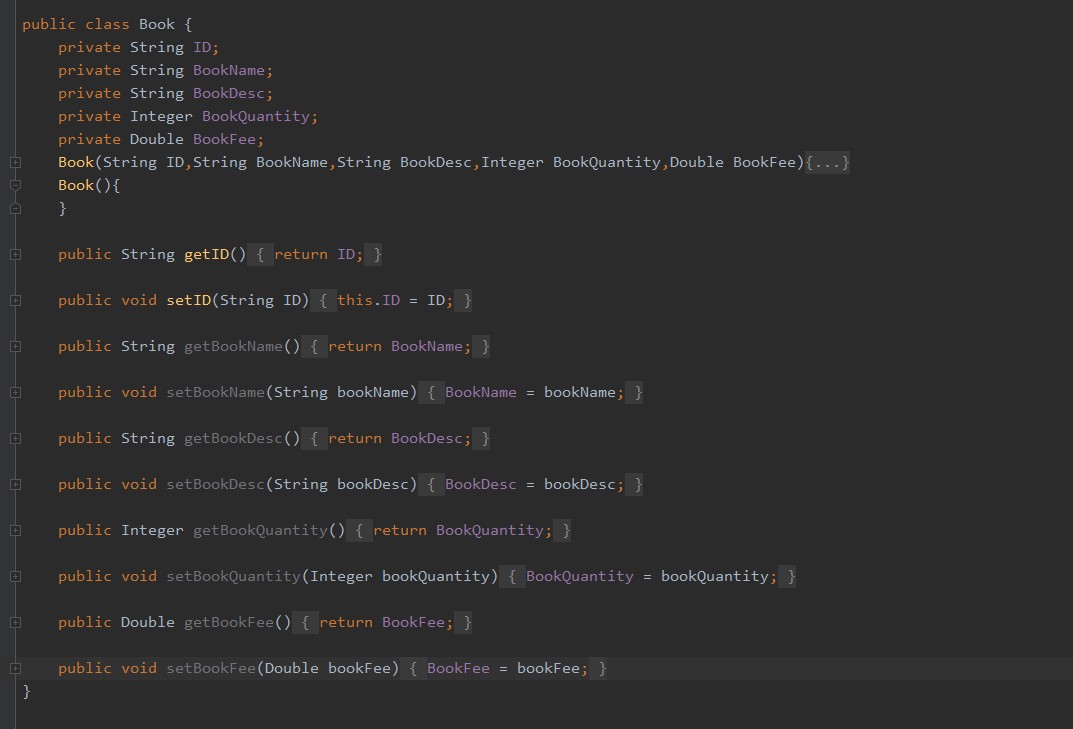
****

****

Deleting from Database

In this part of program there is deleting only one particular book which is booked on some student.

If the program does not find such a book or IDI, it asks to enter the data again

****

Class Book

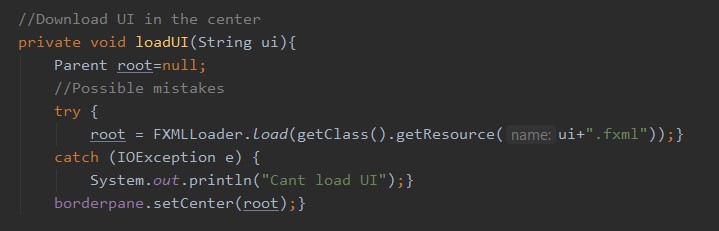
Within “Class Book” we have:

* ID
* Book’s Name
* BookDesc
* Quantity of book
* Fee

Reservation

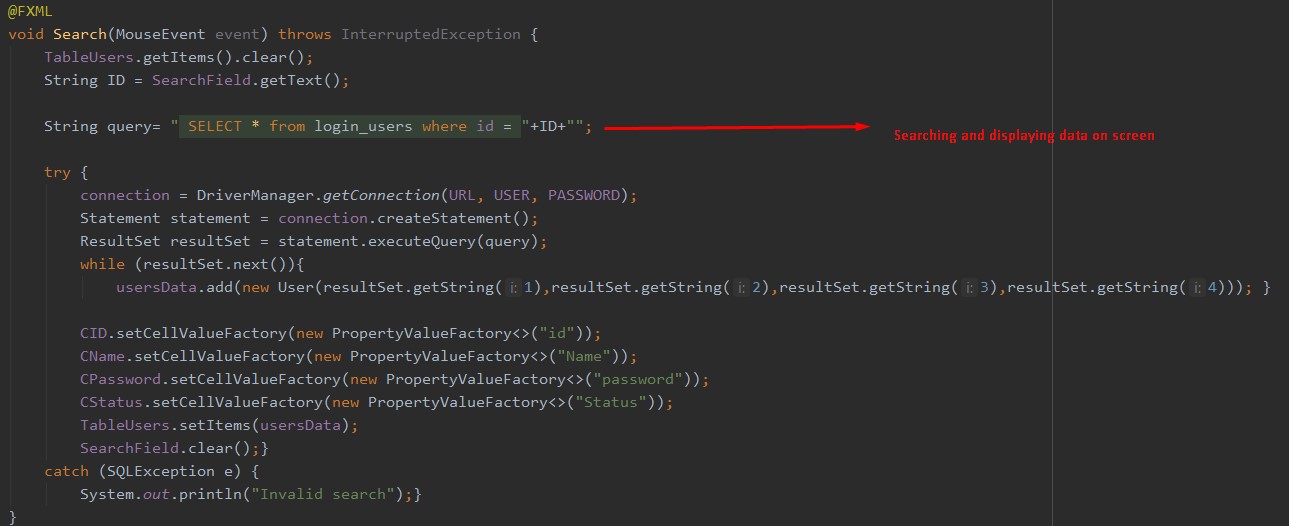
In this part of the code, a student’s debt record is created if he forgot to bring any book back to the library.

It also takes the number of books out of the public

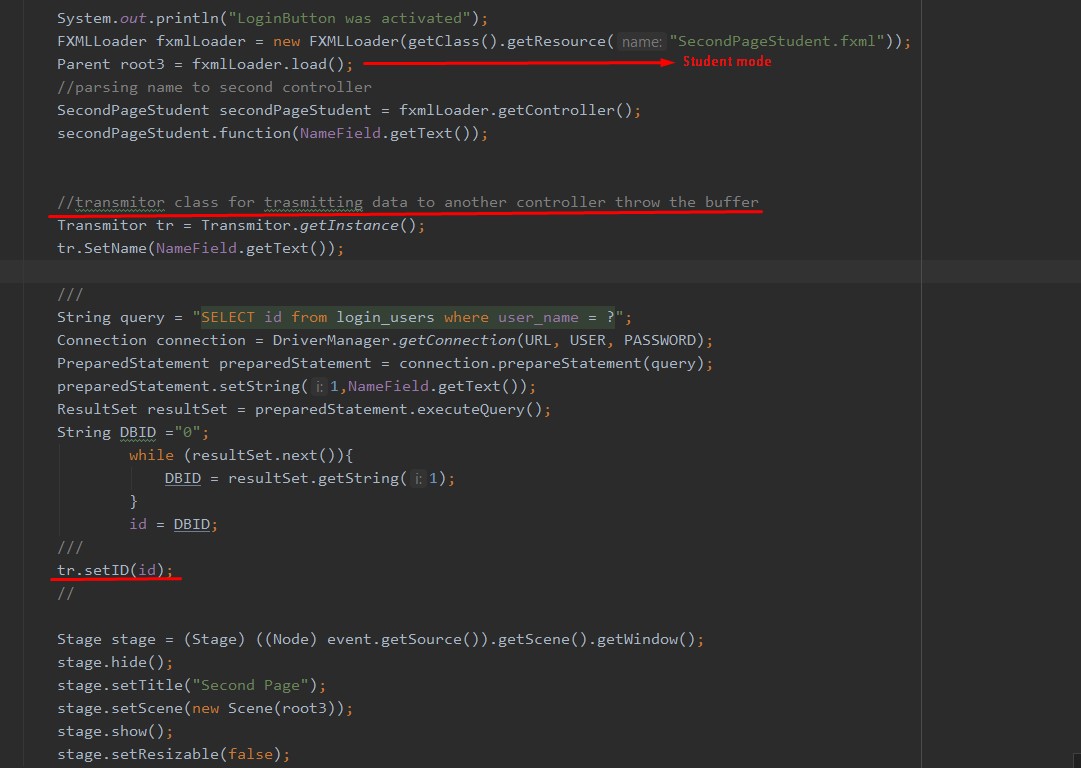
****

UI

In this part of our program we are downloading UI in the center.

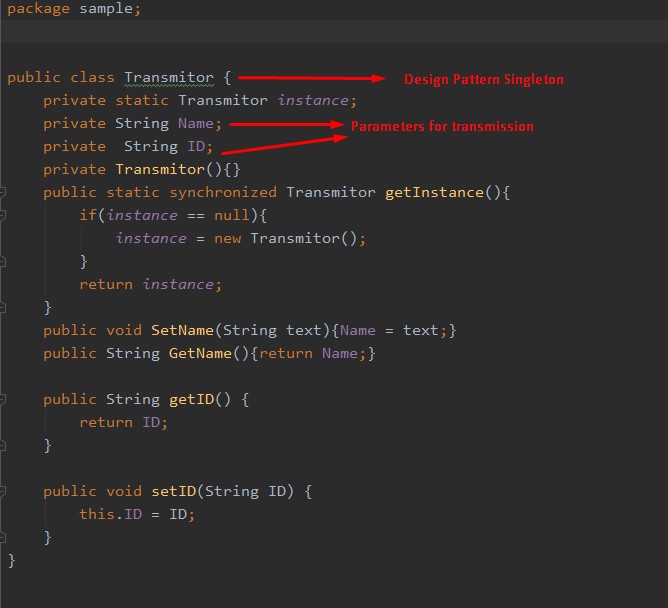
****

There is searching and displaying data. We can easily find any particular information and after that operation our application will show it to us.

****

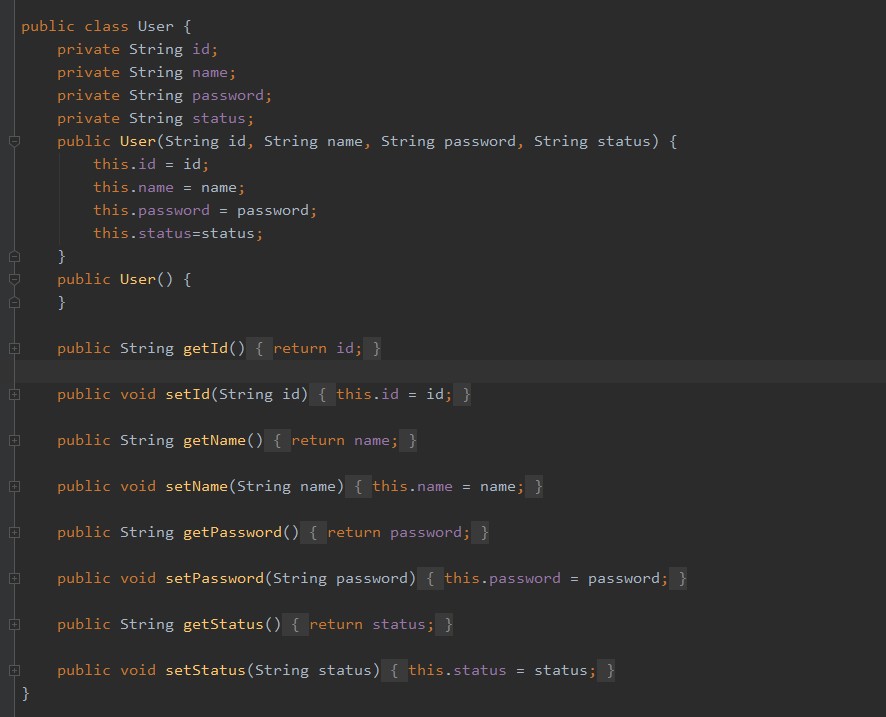
To get the interface definition from a file, we are using the getClass () method.getResource ("Main.fxml").

The Method FXMLLoader.load() returns an object of type Parent, which we can pass to the constructor of the Scene object, and thus our application will get the interface from fxml.

****

Transmission

In this piece of code occurs transmission parameters and more precisely “name” and “ID”

****

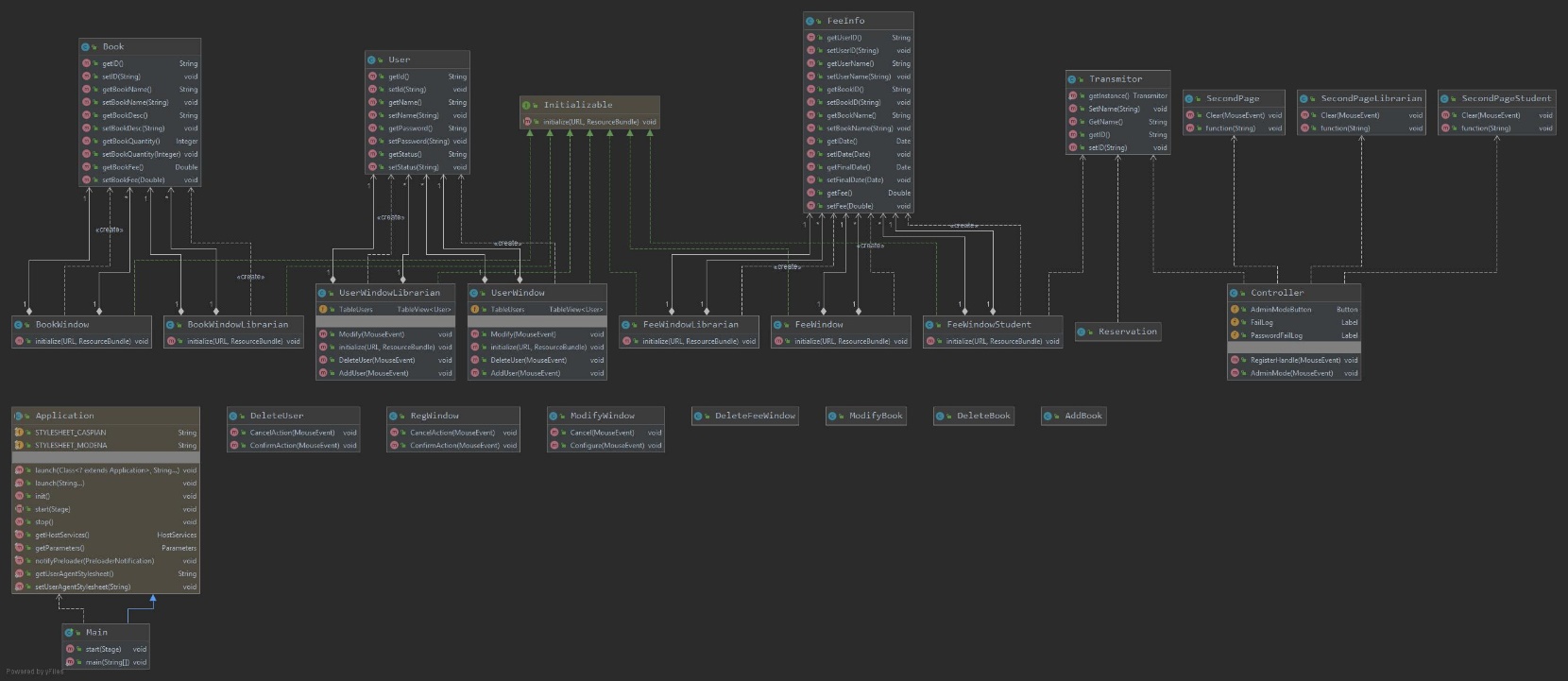
This part of program describes class User. There is ID, name, password and current status.

|  |  |  |
| --- | --- | --- |
| ID | Full name | Responsible for particular feature |
| U1810017 | Arifdjanov Sarvar (Group 1) | Inserting and connected code in Database |
| U1810014 | Smirnova Ekaterina (Group1) | Visualization and working with actions in the application |
| U1810050 | Gapurova Zarnigor (Group 1) | Creating and choosing design for application |

**3. User Accounts in LMS:**

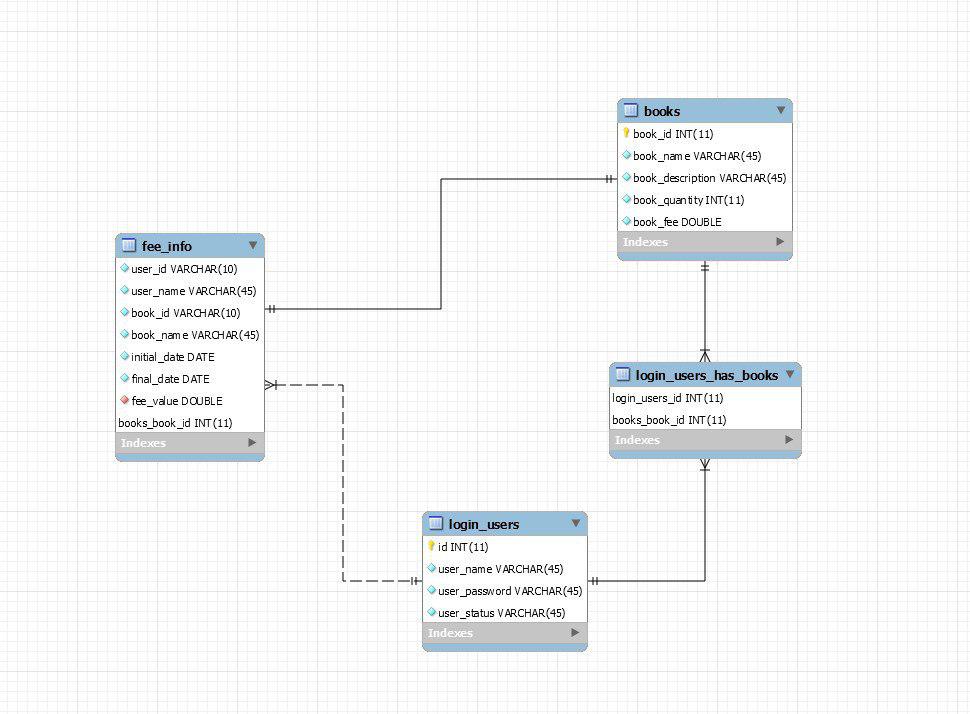
|  |  |  |
| --- | --- | --- |
| *Accounts* | *Login* | *Password* |
| Administrator | OverLord | 123456789 |
| Librarian | Elizabeth II | Eli1997 |
| Student | Spy12 | 12345 |

**4. UML Diagrams**



* Design patterns have two major benefits. First, they provide with a way to solve issues related to software development using a proven solution. The solution facilitates the development of highly cohesive modules with minimal coupling. They isolate the variability that may exist in the system requirements, making the overall system easier to understand and maintain. Second, design patterns make communication between designers more efficient.

**5. Entity-Relationship Diagram**

****

**Benefits and Limitations of Database**

* *Benefits*

1. ID and name can have any format of spelling (any numbers or letters).
2. Databases are mutually independent. Therefore we can make a lot of requests and changes in one database will not affect the other database

* *Limitations*

1. The database is local and maintain only one user.
2. Since Databases are mutually independent synchronization is more difficult.
3. ID can not be more than 10 characters and the name is more than 45