Name-Surname: Nouredeen Ahmed Mahmoud Ali Hammad
Student No: 2121221362

BLM19104E Computer Programming II Project

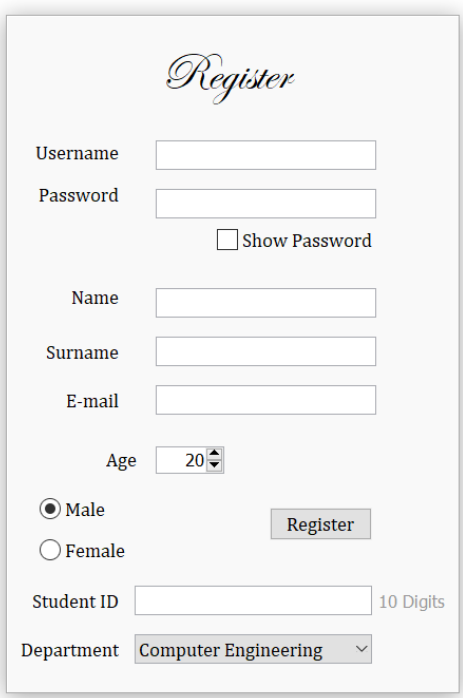
Project Concept:

This is a library management application meant to be used by a university library.
There are two roles in the application: Student and Admin.
Students can rent books from the library for a chosen amount of time and return them later.
Admins have full control of the books in the library, they can add, edit and remove books. They can send warning messages to renting students and they can also ban them from the system.
Admins can also manage students' profile edit requests. Because this is a university library, students cannot change their information without approval from an administrator.

Project Section Breakdown:

Login Section:

I) Register Tab:



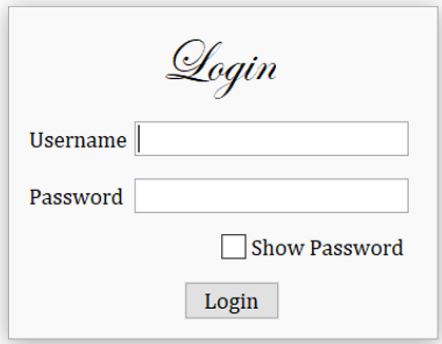
* Four *Regular expressions* were used to check the validity of all inputs:

1) The username and password can be any character except # $ ‘ “ because they can cause SQL errors.
2) The name and surname can only be consist of letters.
3) The e-mail has to follow the following format "_____@____.edu.tr", where each underscore can be any letter, number, underscore, period or hyphen.
4) The Student ID must only consist of 10 digits.

* When the register button is pressed, the following processes run.

1) All input fields are checked if they are empty, a JOptionPane warning appears if they are.
2) The Regular expressions are run, a JOption pane warning appears depending on which regular expression fails.
3) The program accesses the database and checks whether or not the entered username is unique. It checks in both the Students and Admins table. If the username is already registered, a JOptionPane warning message is shown.
4) The entered Student ID is also checked if it is unique. You cannot create more than 1 account with the same Student ID. If the entered Student ID has been banned before, a JOptionPane message tells the student to contact the university technical department.
5) The Email is also checked if it is unique.
6) The password is hashed with SHA-256 algorithm.
7) A new student row is added to the database, and “Student registered successfully” message is shown.

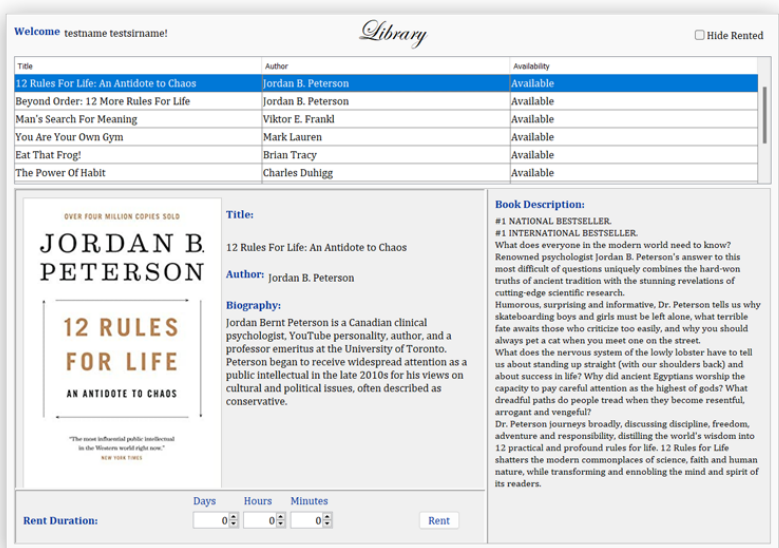
II) Login Tab:



When the login button is pressed:
1- All input fields are checked if they are empty or if they have illegal characters.
2- The username is taken and searched in both the students and admins table, when it is found, the table name is returned.
3- All that user's info is retrieved from their table,
4- If the user is banned, they can't log in and a JOptionpane message is shown that they are banned.
5- The entered and the stored passwords are compared, if they match then login is successful.
6- the *StudentHomeFrame* or the *AdminHomeFrame* is launched depending on what type of user logged in.

Student Section:

I) Library Tab:



In the library tab, all the books in the database are shown on the jtable, each book's title, author and availability is shown.

Unavailable books can be hidden with the "Hide Rented" checkbox.

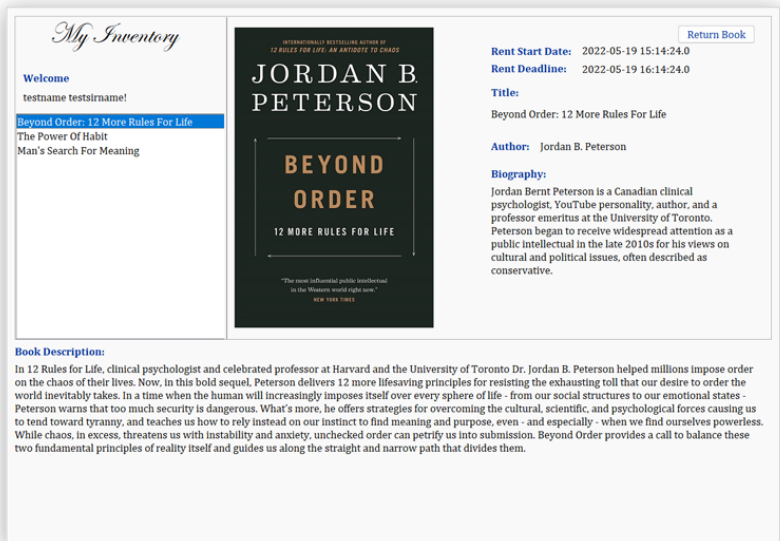
When a book is clicked on, all the book's details are shown on the frame.

The rent button is disabled if the selected book is unavailable (rented to someone else)

The student can choose a duration for their book rental (cannot be 0,0,0) and rent the selected book, when the rent button is clicked, the following occurs:

1- The selected book's ID is taken from libBookArr or hideModelIDs (depending on if the "hide rented" button is pressed or not)
2- The deadline timestamp is calculated depending on the entered duration and the current system time.
3- A new row is added to the RentInfo table in the database, including the BookID, the StudentID, RentTimestamp and DeadlineTimestamp.
4- The Books table is updated, and the selected book's RentedTo column is updated with the current student's ID.

II) My Inventory Tab:



This tab shows all of the current student's rented books.

When an item is selected on the Jlist, the selected book's details are shown, and the rent and deadline dates are shown.

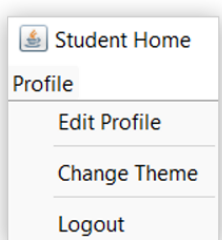
The deadline date will be colored red if the deadline date has passed.

The student can return their book by clicking return.

When return is clicked the following occurs:

1- Gets the selected RentInfo ID from the rentInfoIDs with the selected jlist index.
2- Deletes the RentInfo row from the database table with the selected RentInfoID.
3- Updates the Book table, changes the RentedTo column for the selected book back to 0.

III) Menus:



** Edit Profile*
This menu item launches StudentProfileFrame, where the student can change any of their information. The same regular expression checks used in the registration screen are used here.
When "Send Edit Request" is clicked, a new edit request is created in the EditRequests database table, admins can then review the request and approve or reject it.
If the student already has an active edit request, they are asked if they want to overwrite it, they can choose to cancel or overwrite it.

> **Reset Password**: Launches the ResetPassDia dialog, this dialog is a generic class, it is used to reset password for both students and admins.

** Change Theme*
This menu item launches a JColorChooser dialogue, the student can choose a color theme fo their frame.
The student's choice is saved in the database, and the previously chosen color theme will appear when that student logs in again.

** Logout*
Shows an "Are you sure you want to logout?" JOptionPane, if the student presses yes, the current frame is disposed of, and the login frame is launched.

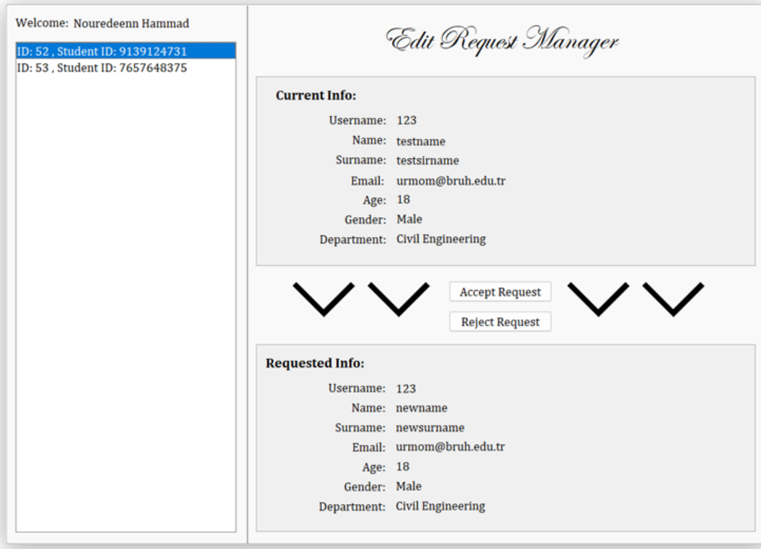
IV) On Launch:

On launch the following occurs:

1- The student's id is searched in the Messages database table, and any messages matching the search are returned and showed to the student in JOptionPanes as notifications.
2- The student's rent info is searched and if any of their book return deadlines have passed compared to the current system time, warning messages will show which books have passed their deadlines.

Admin Section:

I) Edit Request Manager Tab:



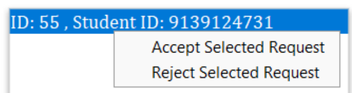
All active edit requests are shown on the left JList.

When one is selected, all the details of the edit request are displayed on the right section of the screen.

The top part displays the student's current info, the bottom part displays the requested info.

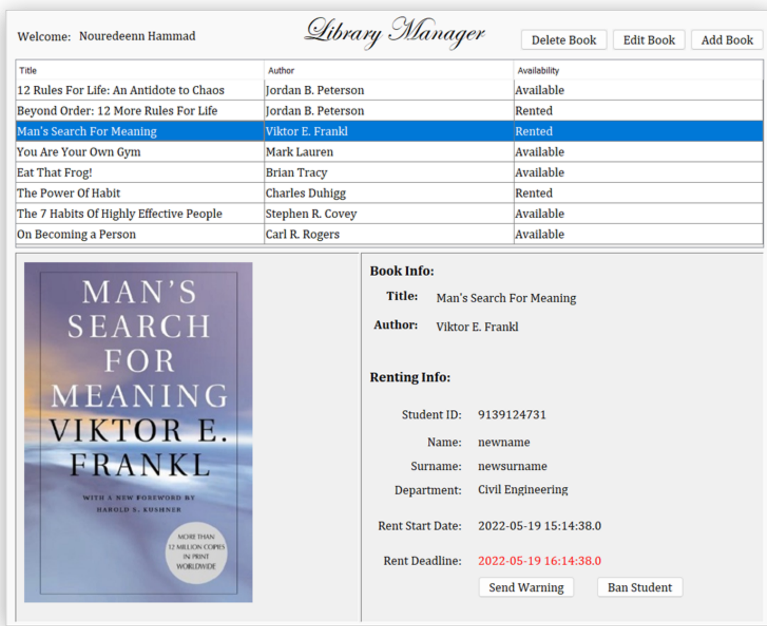
When accept request is pressed:
1-The selected student's info is updated into the requested info in the Students database table.
2-The edit request is removed from the database.
3-A message notification is sent to that student.

When reject request is pressed:
1-The admin must enter a reason for rejection, and the reason is sent in a message notification to that student.
2-The requested info is discarded.
3-The edit request is removed from the database.



**Popup Menu:*
You can accept or reject the selected edit request from the popup menu as well.

II) Library Manager Tab:



All books in the system are shown and can be managed from here.

When a book is selected from the table, its book cover, basic info and renting info is displayed.

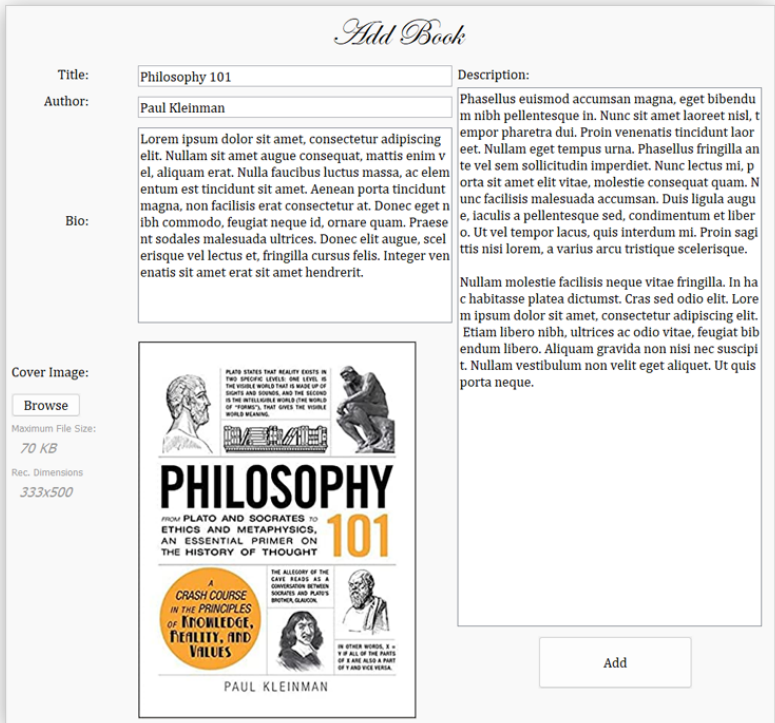
The admin can send a warning notification message to the student renting the selected book (in case of a missed deadline for example)

The admin can ban the student renting the selected book. When a student is banned, all the books they were renting are returned to the library.

** Popup Menu:*
By right clicking on the JTable a popup menu appears, the admin can add, edit or delete selected book. It performs the same operations as explained below.

*Add Book

The add book button opens the NewBookFrame and the admin can enter all the new book's info, and browse their pc for a book cover.



The book cover can be a maximum of 70KB

When the add button is pressed:

1- All fields are checked whether they're empty.
2- The strings inside the Bio and Description JTextAreas are processed, and all new-line characters (/n) are replaced with "
" in order to correctly preserve new lines so that they are displayed properly in the interface.
3- The image file is converted into a binary input stream, then it is uploaded to the Books database table as a BLOB (Binary Large Object)

*Edit Book

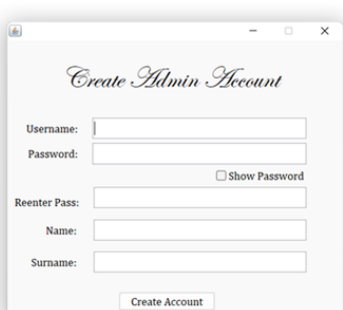
The edit book button opens the EditBookFrame, it has very similar function to the AddBookFrame except that the book's current info is filled in the fields.

*Delete Book

After giving the admin a confirmation JOptionPane, the delete book button deletes the selected book from the Books database table.

If the selected book is rented to someone, an alert is shown to the admin, if the admin then proceeds with the deletion then that student get a notification alert that a book they've rented has been deleted.

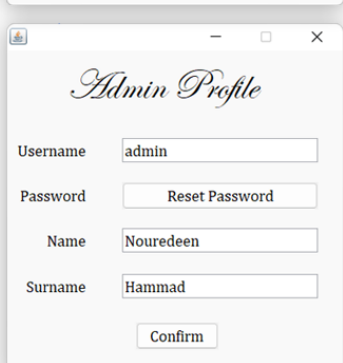
III) Menus:



* *Create Admin Account:*

Launches a small form to create a new admin account since admins cannot register normally, another admin has to create a new admin account.

In this form, when the create account button is pressed, the same regex and database checks that are in the registration frame run here.



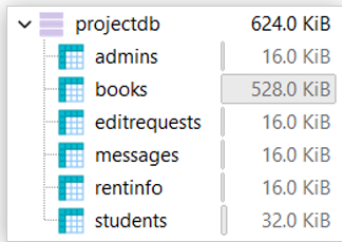
* *Edit Profile:*
Launches the AdminProfileFrame and fills fields with current information, the admin can change any of their profile information directly from here.
Unlike the student profile frame, no approval is required and changes are applied instantly.

> **Reset Password**: Launches the ResetPassDia dialog, this dialog is a generic class, it is used to reset password for both students and admins.

* *Logout:*
Shows an "Are you sure you want to logout?" JOptionPane, if the student presses yes, the current frame is disposed of, and the login frame is launched.

* *Popup Menu:*
Shows an "Are you sure you want to logout?" JOptionPane, if the student presses yes, the current frame is disposed of, and the login frame is launched.

Project Database Architecture:



The database contains 6 tables:

1- admins
2- books
3- editrequests
4- messages
5- rentinfo
6- students

1) Admins:

-AdminID is the primary key, and it auto-increments.
-Username is set to unique.

#	Name	Datatype
1	**AdminID**	**INT**
2	Username	TINYTEXT
3	Password	TEXT
4	Name	TINYTEXT
5	Surname	TINYTEXT

2) Books:

-BookID is the primary key, and it auto-increments.
-RentedInfo's value is 0 by default, when the book is not rented it stays 0, and when it's rented the renting student's StudentID is placed in this column.
-The ImageBlob column has the BLOB (Binary Large Object) datatype, and it saves the book's cover image as a binary stream.

#	Name	Datatype
1	**BookID**	**INT**
2	Title	TINYTEXT
3	Author	TINYTEXT
4	Bio	TEXT
5	Description	TEXT
6	RentedTo	TINYTEXT
7	ImageBlob	BLOB

3) EditRequests:

-RequestID is the primary key, and it auto-increments.
-NewIsMale is 0 if it's Female, and 1 if it's Male.
-All the columns starting with "New" are the student's requested information.
-All the student's current information can be retrieved using the StudentID in a SELECT statement.

#	Name	Datatype
1	**RequestID**	**INT**
2	StudentID	TINYTEXT
3	NewUsername	TINYTEXT
4	NewName	TINYTEXT
5	NewSurname	TINYTEXT
6	NewEmail	TINYTEXT
7	NewAge	TINYINT
8	NewIsMale	TINYINT
9	NewDepartme...	TINYTEXT

4) Messages:

-MessageID is the primary key, and it auto-increments.
-StudentID holds the ID of the student that this message will display to.
-Event holds the title or the purpose of the message.
-Content holds the actual contents of the message.

#	Name	Datatype
1	**MessageID**	**INT**
2	StudentID	TINYTEXT
3	Event	TINYTEXT
4	Content	TINYTEXT

5) RentInfo:

-RentInfoID is the primary key, and it auto-increments.
-RentTimestamp and DeadlineTimestamp use the TIMESTAMP datatype.
-RentTimestamp stores the timestamp at the time of renting on the student's system clock.
-DeadlineTimestamp stores the timestamp of the deadline, which is calculated depending on the student's system time at the time of renting and the chosen rent duration.

#	Name	Datatype
1	**RentInfoID**	**INT**
2	BookID	INT
3	StudentID	TINYTEXT
4	RentTimestamp	TIMESTAMP
5	DeadlineTime...	TIMESTAMP

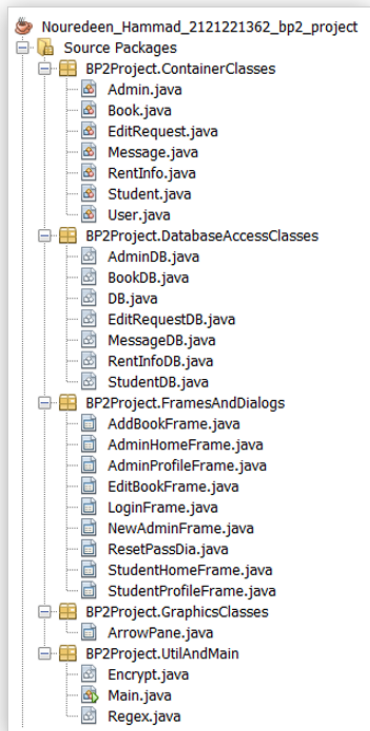
6) Students:

-This Table has no Primary auto-incrementing column.
-StudentID is a 10 digit unique non-incremental number, so it is used as if it was the primary key.
-Username and Email are also set to be Unique.
-IsMale is 0 if the student is Female and it's 1 if the student is Male.
-EditRequestID column's default value is 0, when the student doesn't have an active edit request it stays 0, when the student creates an edit request, the EditRequestID is placed in this column.
-IsBanned column's default value is 0 which means "not banned", and it becomes 1 when the student is banned. (Banned students are prevented from login)
-ThemeColorR, ThemeColorG, ThemeColorB are used to store the RGB values of that student's chosen theme color. They are 0,0,0 (Black) by default.

#	Name	Datatype
1	StudentID	TINYTEXT
2	Username	TINYTEXT
3	Password	TINYTEXT
4	Name	TINYTEXT
5	Surname	TINYTEXT
6	Email	TINYTEXT
7	Age	TINYINT
8	IsMale	TINYINT
9	Department	TINYTEXT
10	EditRequestID	INT
11	IsBanned	TINYINT
12	ThemeColorR	INT
13	ThemeColorG	INT
14	ThemeColorB	INT

NOTE: *StudentID is stored as a string, because it is a 10 digit long number, and this passes the boundaries of INT*

Project Class Architecture:



The Project contains 27 classes divided into 5 categories:

1- Container Classes
2- Database Access Classes
3- Frames and Dialogs
4- Graphics Classes
5- Utility and Main

1) Container Classes

These classes are used as data containers throughout the code to maintain code cleanliness and readability. They are also used as return types or parameteres for some functions and constructors.
The variables in these classes correlate directly (with some exceptions) with their database table columns.
The Admin and Student classes both inherit from the User class.

2) Database Access Classes

These are all abstract classes, and they manage all communications to and from the MySQL database.
DB is the main database access class, it contains the Connection object and a number of abstract functions for general use.
All other database access classes inherit from DB. They are divided into separate classes in order to maintain class sizes and general order.
All database processes that relate to Students are put in StudentDB, any that relate to Books are put in BookDB etc.

3) Frames and Dialogs

All the JFrames and JDialogs that are shown in the User Interface are in this category.
The UI was explained in detail in the *Project Section Breakdown* part of the report.

4) Graphics Classes

This category only contains one class ArrowPane, it extends JPanel.
The ArrowPanel's paintComponent function was overwritten to draw two arrows pointing downwards, this is used for decoration in the AdminHomeFrame's EditRequestManger tab.

5) Utility and Main Classes

This category has the project's one and only main function.
Regex and Encrypt are used as utility classes.
Regex includes multiple regular expressions that are used throughout the project's frames.
Encrypt contains one function named hash, this function takes a string as a parameter, hashes it with SHA-256 algorithm and returns the hashed string.
The hash function is used whenever a password is entered in any of the login or register form.

Tools used in development:

This project was developed on Netbeans 8.2 without using Maven.
XAMPP was used to host MySQL to my localhost.
HeidiSQL was used as a my MySQL administration tool.