**Digital Library management**

**Criteria C – Development**

**List of techniques**

1. GUI Programming
   1. Navigation
   2. Jtables
   3. Forms
2. Object-Oriented Programming
   1. Polymorphism
   2. Encapsulation
   3. Inheritance
3. Class and Objects
4. Sorting and Searching
5. Exception handling with validation and prompts
6. External Libraries
   1. Jasper report
   2. Jcalender
   3. JfreeChart
   4. PostgresSQL
7. SQL Database
   1. Database Connectivity
   2. Database Structure
   3. SQL Commands and Queries
8. Data Structure
9. Nested Loops and Complex Conditional Statements
10. Dynamic Chart Generation
11. Report Generation

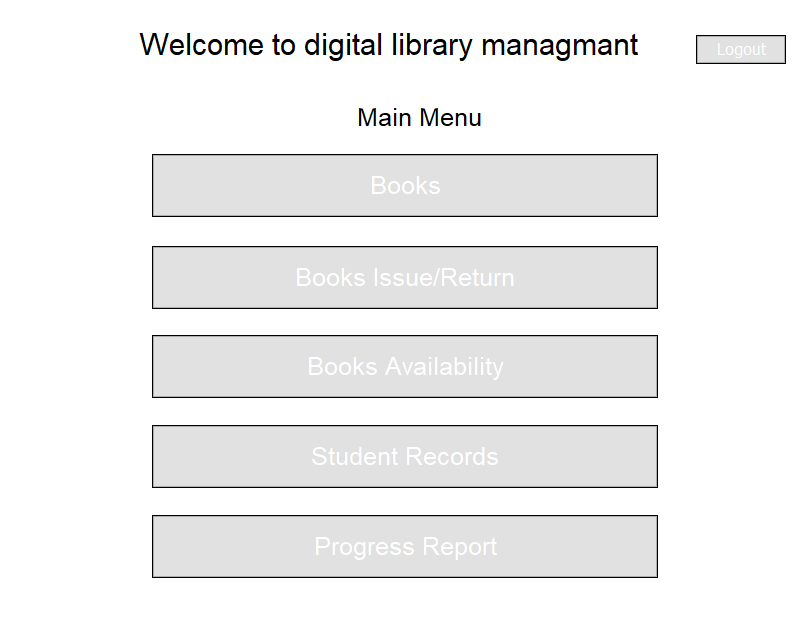
**1)** **GUI Programming**

**1.1) Navigation**

To help client navigate between different screens, JButton is used in the dashboard form to connect the client to different panels. Also, a BACK button is provided to help the client navigate back to the main screen.



Figure 1 – The code used for using JButtons to navigate between the screen and example here is button to go to the book availability form



JButton are used allowing the client to navigate between the screens.

Figure 2 – The dashboard form with Jbutton to navigate to other windows

**1.2)** **JTable**[[1]](#footnote-2)

JTable is used to store and display information to the client about all the information from the database after certain events are executed. The JTables in the code is used at various places but here the example taken is of student form where the JTable is being used.

Graphical user interface, text, application

Description automatically generated

All fields are input from the student form and stored as string values into the table.

Figure 3 – Code for loading the values into the JTable from the database

Graphical user interface, text, application, email

Description automatically generated

Code for the propertities of the table

Code for the title is on top of each column in JTable.

Figure 4 – Code for the properties, title and the setting of the JTable

Table

Description automatically generated

We can see the title and the properties set in figure 4 and the output is like this.

Figure 5 – Output of JTable in the student form for depicting the students who are part of the library program

JTable is used to auto-populate book details on the screen on click of the particular book entry as stored in the database

Figure 6 – OnClick Mouse event used when the user clicks on a row of the JTable then the values will automatically be loaded into the text fields in the student form section for updating/deleting the value

Text

Description automatically generated

Now the values which are received before is now added into the specific textbox

These statements are used to retrieve values when a specific row is clicked.

Graphical user interface

Description automatically generated

The blue color depicts that this row of the JTable has been selected.

Figure 7 – Output when a book is chosen then its value is automatically loaded in the textbox.

**1.3)** **JForms**

JForms are used to make the GUI of this desktop app. The JForms and their functionalities are used everywhere. To depict the functionality of the JForm, I will take login form as an example.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Figure 8 – Code of designing the JForm

Graphical user interface, application

Description automatically generated

The output of the title, textbox’s, label and button

Figure 9 – The code for designing the Jforms output looks like this

**2) Object-oriented programming**

**2.1) Abstraction[[2]](#footnote-3)**

Abstraction refers to minimum information being visible to the end-user and the complexity is hidden from the end-user. In this program, abstraction is used when a method is called to get the values/information for generating a graph but the code for this is not directly visible to the end-user. Therefore, it allows the repetitive use of code and information hiding.

Graphical user interface, application

Description automatically generated

Figure 10 – The method is defined to get all books issued records for a particular student from the database

A picture containing logo

Description automatically generated

Figure 11 – Calling the method to get the information and hidden complexity

**2.2) Encapsulation[[3]](#footnote-4)**

Encapsulation is used to bind all data together to be used as a single unit. As used at various instances throughout the program. One such example of Encapsulation is below.

Text

Description automatically generated

In this example, encapsulation is used as the use of members and try-catch statements and many more are all kept together in a class which used as a container

Figure 12 – Code is an example of encapsulation

**2.3)** **Inheritance[[4]](#footnote-5)**

The classes used in this program are inherited from pre-defined Java methods such as Javax.swing.JFrame.

The word extend depicts that it a pre-defined method is inherited into a Java class



Figure 13 – Code is an example of inheritance

**3)** **Class and objects**

A Java program is made using a Java Class and all the information related to the class is accessed by the objects created and implemented. In this software, classes and objects are used in various places throughout the program.

For instance, we will be taking ConnectDB as a class. The purpose of this class is to build the connection between the database and the frontend program. The object dbconnect is used to access the ConnectDB class.

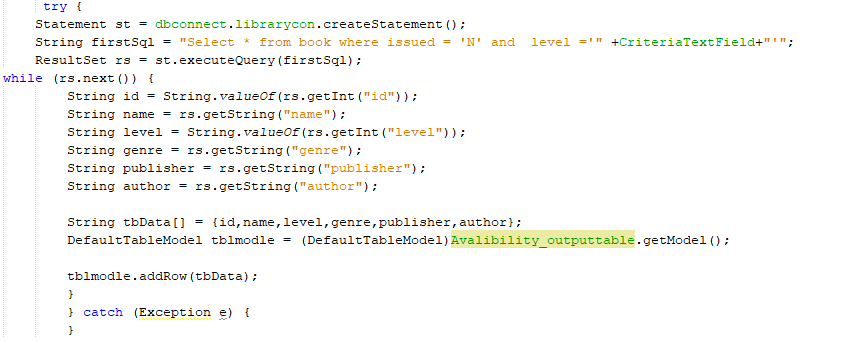


The connectDB method is used to build connection between the Java forms and data base to allow retrieval and storing of data

Figure 14 – Depicts the ConnectDB Class being implemented



Figure 15 – Depicts the ConnectDB class being converted into dbConnect object



After converting the connectDB method to dbconnect. Then we can see it is used here to retrieve that method and used in this piece of code to retrieve data from the database.

Figure 16 – Depicts the implication of the dbConnect method

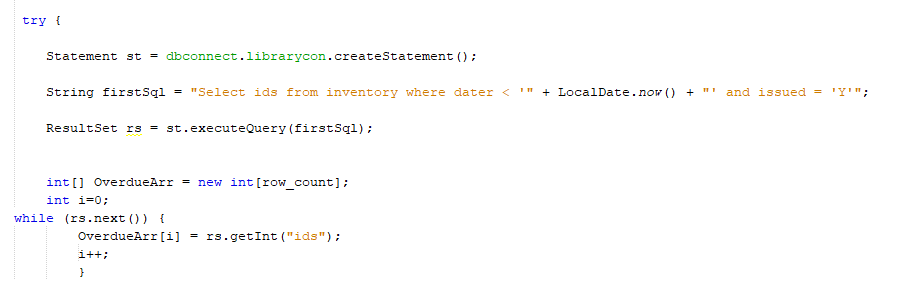
**4)** **Sorting and searching**

In the program, sorting and searching algorithms are used together to provide warnings and resistance to the client from issuing books to students who have overdue books already. Bubble sort[[5]](#footnote-6) is used to arrange the list of student IDs of students with overdue books in ascending order. Then making use of binary search[[6]](#footnote-7) to cross-check if a particular Student ID is there in the array.



The values from the database are being retrieved to see how many overdue books there are, and each value present means the total values count will be stored in row count variable

Figure 17 – Code for setting the size of the array using the row count.



Here all the student ID is being stored in an Array “OverdueArr”. Also, the size of the array is defined by the row count set before.

Using this the books which are overdue are being selected from the inventory database

Figure 18 – Code for storing the student ID who have overdue books in the array

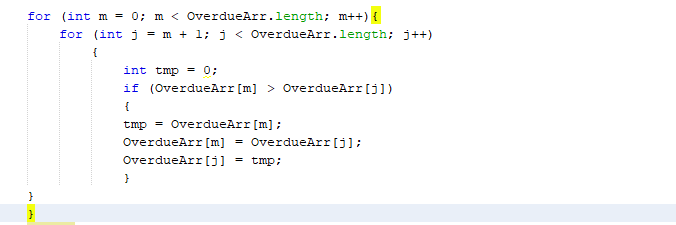
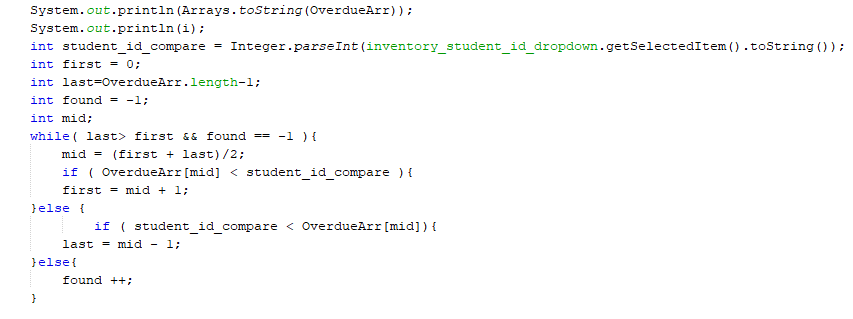
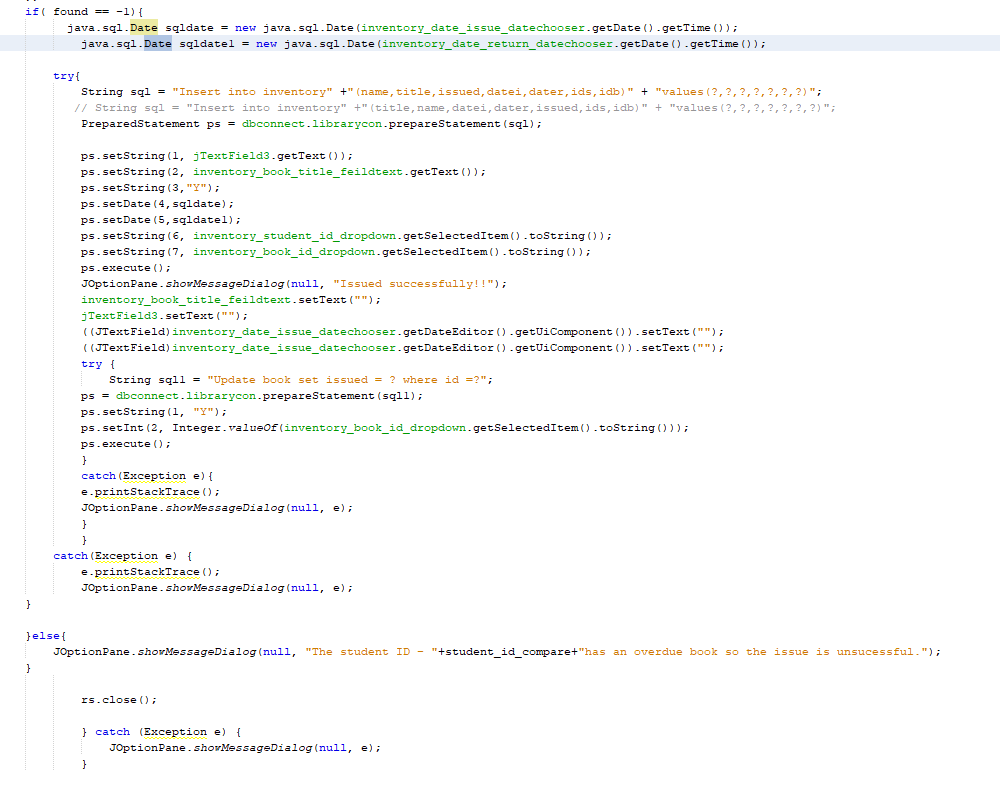


Figure 19 – Code for Bubble sort to arrange the student ID in the array in an ascending order. This is done since in binary search requires a sorted array.



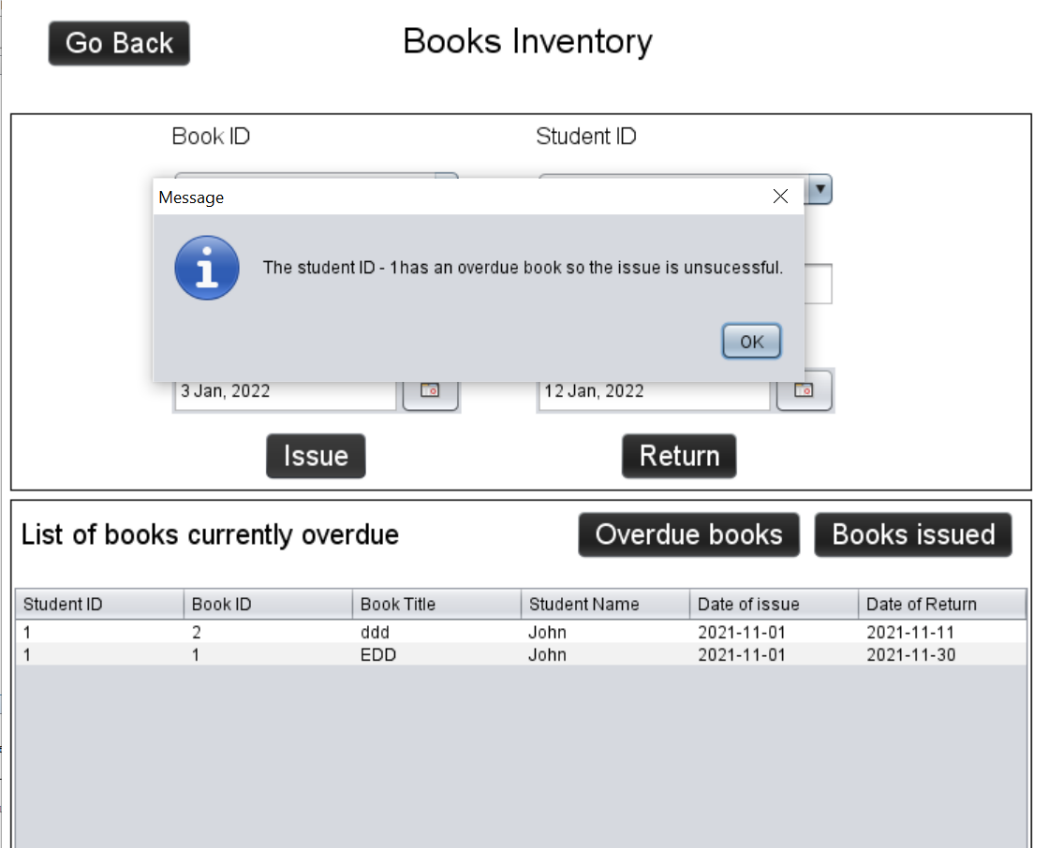
In the binary search, variable “found” is used to see if the student ID is present in the overdueArr or not

Figure 20 – Code for Binary search to check if the Student ID to whom the book is being issued has any overdue book or not.



When the found variable == -1 it means that the student has no overdue book. Thereafter the book issue process continues. On the other hand, If found has any other value then a warning is given.

Figure 21 – Based on the student ID found or not in the overdue array, action will be taken using the above code.

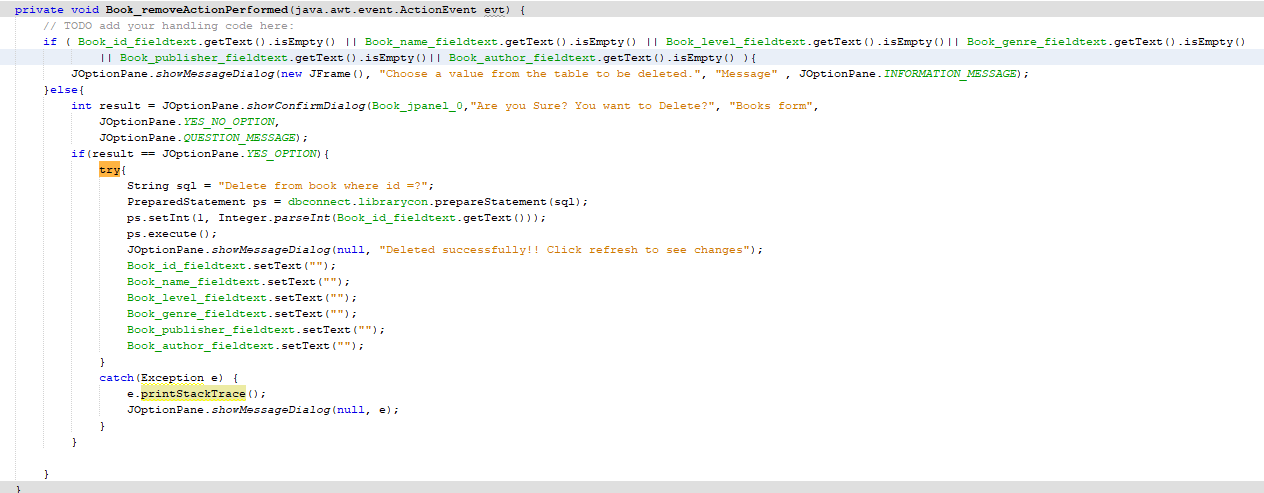


This warning message will be displayed and the book will not be issued until the student return back the overdue book.

Figure 22 – Output when a student has an overdue book then this kind of a warning comes.

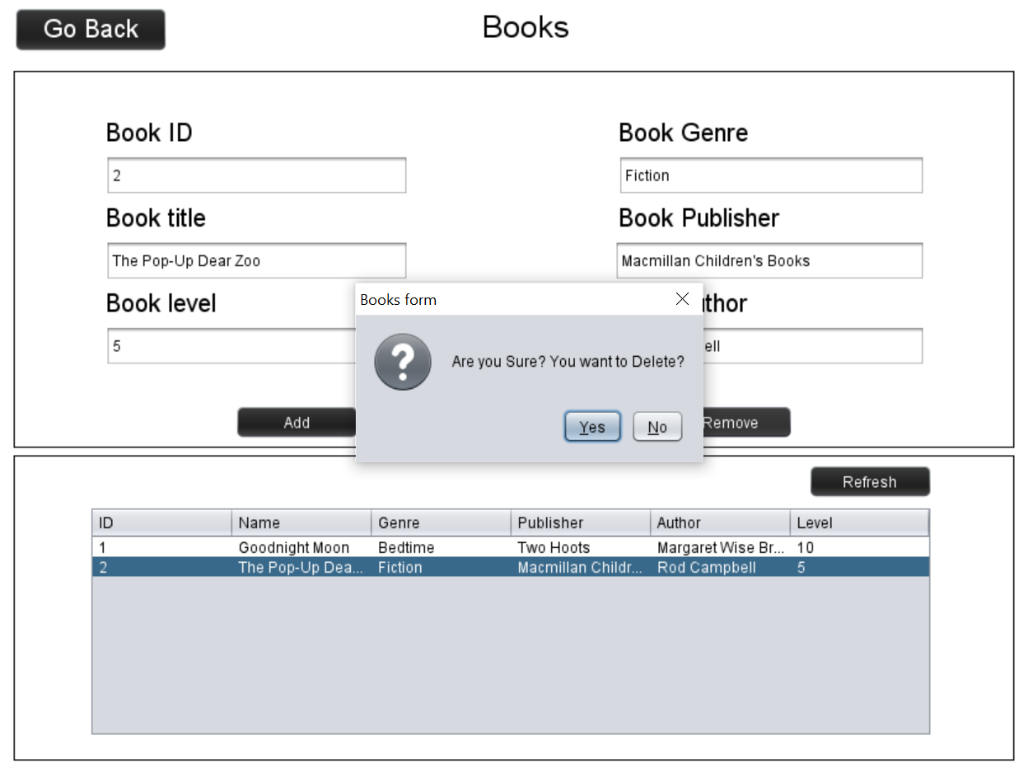
**5) Exception handling[[7]](#footnote-8)** **with validation and prompts**

This system makes use of structured programming hence the client needs to receive error messages or information when invalid input is entered. This can act as a validation to inform the user about what mistakes are they making. Also, to avoid any runtime error or database error exception handling is used such as Try-Catch statement.



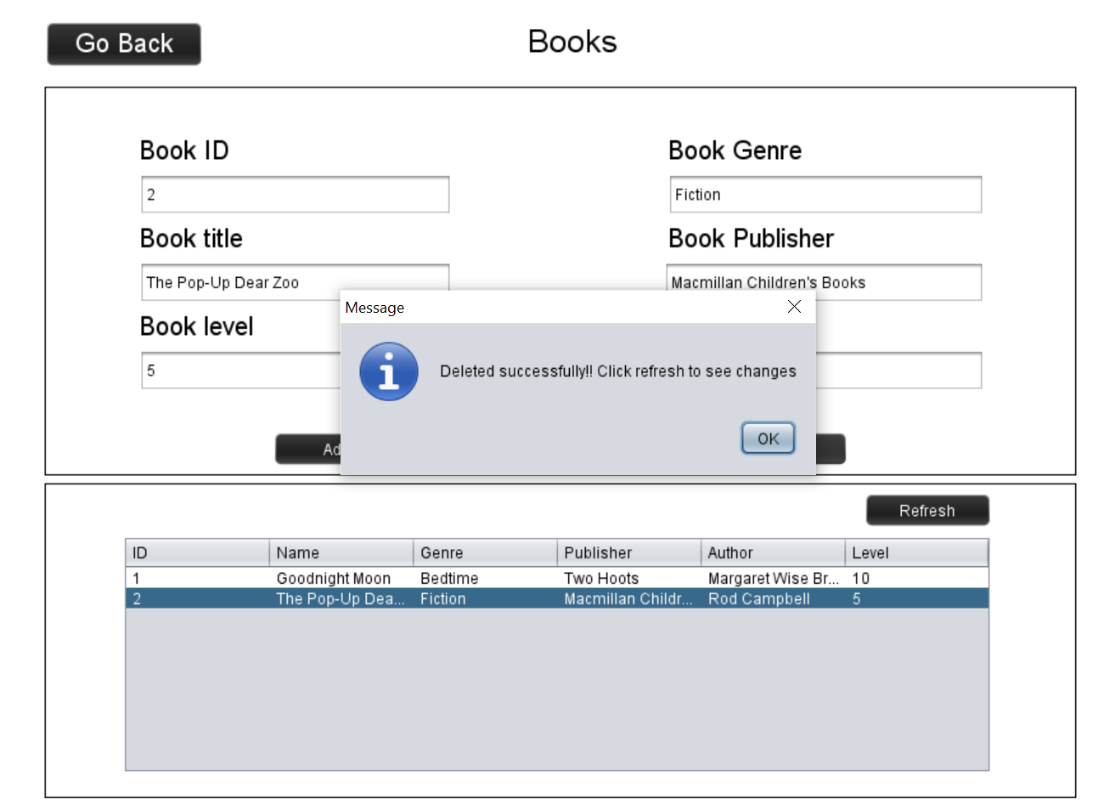
We can see the code of removing a book and this is covered with Try- Catch statement which helps avoid any run-time error.

Figure 23 – Code for removing a book from the database which has been removed from the library



This is a confirmation note. This is validation prompt re-validating if the client wants to delete the book or was its a by mistake click.

Figure 24 – Output when a book is chosen to be delete.



Deleted error box depicting the book has been deleted successfully.

Figure 25 – Output when a book is successfully deleted.

Use of If statement to check if nothing is entered then appropriate dialogue box is visible and similarly is wrong username or password is entered then there is warning message as well.

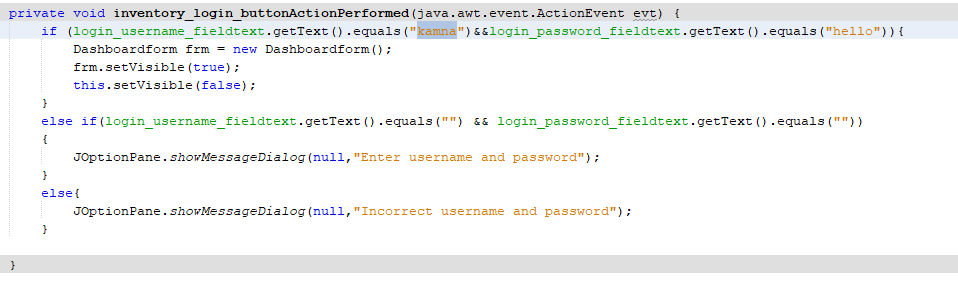
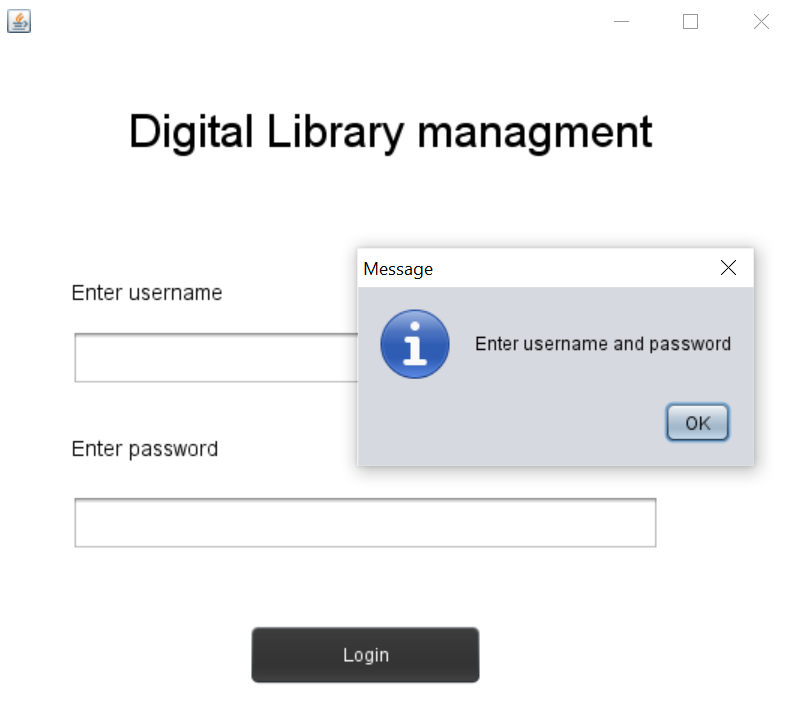


Figure 26 – Code for validating if the password and the username entered matches the set password and username



This error message appears when no information is entered and the login button is clicked.

Figure 27 – Output of a case where there is no login details are entered but the login button is pressed.

**6)** **External libraries**

These are the external libraries used.



Figure 28 – List of all external libraries used in the solution

**6.1)** **Jasper Report[[8]](#footnote-9)**

Jasper Report is used to generate a dynamic report to depict the progress of the student by listing all the books the student has read in a specific period.

Text, letter

Description automatically generated

Figure 29 – Importing Jasper report in the progress report form

Graphical user interface, text

Description automatically generated

Storing the report on a local file space for now

Figure 30 – Implementing the Jasper Report library to create an automatic report

**6.2)** **Jcalender[[9]](#footnote-10)**

Jcalender library is used to provide a calendar like a framework where the user can easily choose the date from instead of manually typing it.

A picture containing text, calculator, screenshot, electronics

Description automatically generated

On click of this button the calendar view will appear.

Figure 31 – JCalendar is used in Book Inventory form to let the user choose the date

**6.3) JFreeChart[[10]](#footnote-11)**

JFreeChart is used to generate a dynamic chart for the progress of the student on the monthly basis from the chosen period by the client.

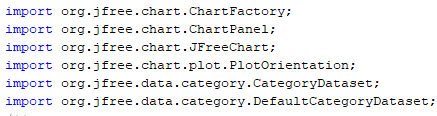


Figure 32 – Importing JFreeChart in the progress report form

Text

Description automatically generated with medium confidence

Code where the bar chart dataset is being created and the data values are stored.

The labels of the bar chart produced

Figure 33 – Implementing the JFreeChart library with the code to generate a chart with hashmap collection where the data is stored from the database

**7)** **SQL Database**

In this product, PostgreSQL is used as a database to store all the information related to the students, books, track of books issued and returned and progress reports.

**7.1)** **Database connectivity[[11]](#footnote-12)**

The database is connected with the GUI program using the connection method. JDBC driver is used to making the connection.

This is where the database is being connected to Java GUI

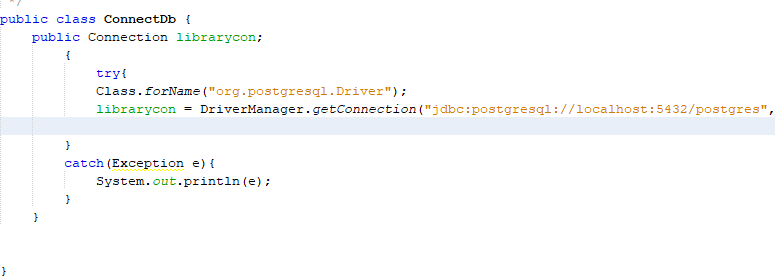
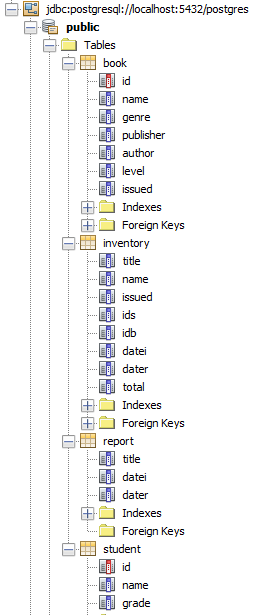


Figure 34 – Code for connecting the database with the GUI application

**7.2)** **Database structure**

My database connection is made and tested with a local server for proper functioning.



The red colour depicts that it is a primary key

These are the list of fields inside the book table

This is the book table

Figure 35 – List of the tables and fields used in the application

**7.3)** **SQL Commands and Queries**

The SQL queries commands techniques are used to insert, update, filter, order by, select and search information with specific criteria.

Graphical user interface, text, application, email

Description automatically generated

Insert query statement used here

Values entered in field

Figure 36 – Insert query to add new books in the database

Graphical user interface, text, application, email

Description automatically generated

Update query statement to update the book details

Figure 37 – Update query to update an existing book detail in the database

Graphical user interface, text, application

Description automatically generated

Delete query statement to delete a book

Figure 38 – Delete query to delete an existing book detail in the database

Between is used to set a range

Graphical user interface, text, application, email

Description automatically generated

Extract is used to extract especially month from Datei. So this means extracting a specific detail.

Count is used to count how many values does the column datei has.

“Select \* From … where …..”Query used to retrieve data from the database

Figure 39 – Various SQL commands used to create bar Chart

**8)** **Data Structure**

One dimensional array has been used to store the student ID of students who have an overdue book. This is for warning and restricting the clients from issuing new books in the book to students who have overdue books. This array can store integers and this array size is set based on the row count of the students who have overdue books. Also, sorting and searching algorithms are used on this array later on.Graphical user interface, text, application

Description automatically generated

We can see the student Id (“ids”) being stored in an Array. The row count is the size of available student ID which is calculated before.

OverdueArr is used to store the list of student ID which have overdue books

Figure 40 – Code for storing the student Id of students who have overdue books

**9)** **Nested Loops and Complex Conditional Statements**

In the program, there is the use of complex and nested loops with conditional statements. It is used to make the program more user-friendly such as creating automatic graphs, reports and many more.

Graphical user interface, application

Description automatically generated

In this program, there is combination of loops and conditional statements combined together. We can see while loop is used with switch and Case- break conditional statement

Figure 41 – Code for getting all the data records about the books issued to a particular student. Using these records to create a graph in the progress report form.

Text

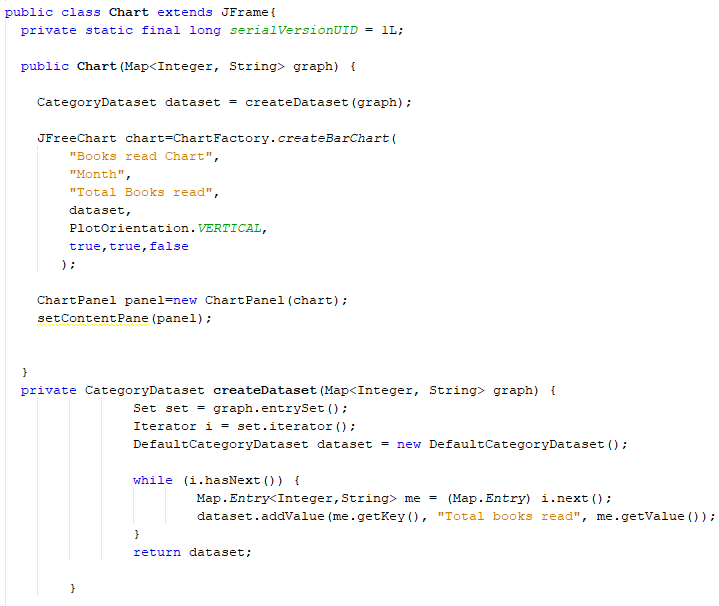
Description automatically generated with medium confidence

A combination of If else statement along with while loop to make use of complex loops and conditional statement nested together.

Figure 42 – Part of the code used to check if a student has an overdue book or not before issuing a book to the student

**10)** **Dynamic chart generation**

To make it convent for the client to analyze the regular progress of the student over some time, a monthly Bar chart is generated using JFreeChart external library.



CreateDataset Method is used to create a dataset of all the values need to create the barchart.

This is the method chart where the code of creating the bar chart is placed

Figure 43 – Code for creating the Bar chart

Graphical user interface, text, application

Description automatically generated

Figure 44 – Code for setting the charts window

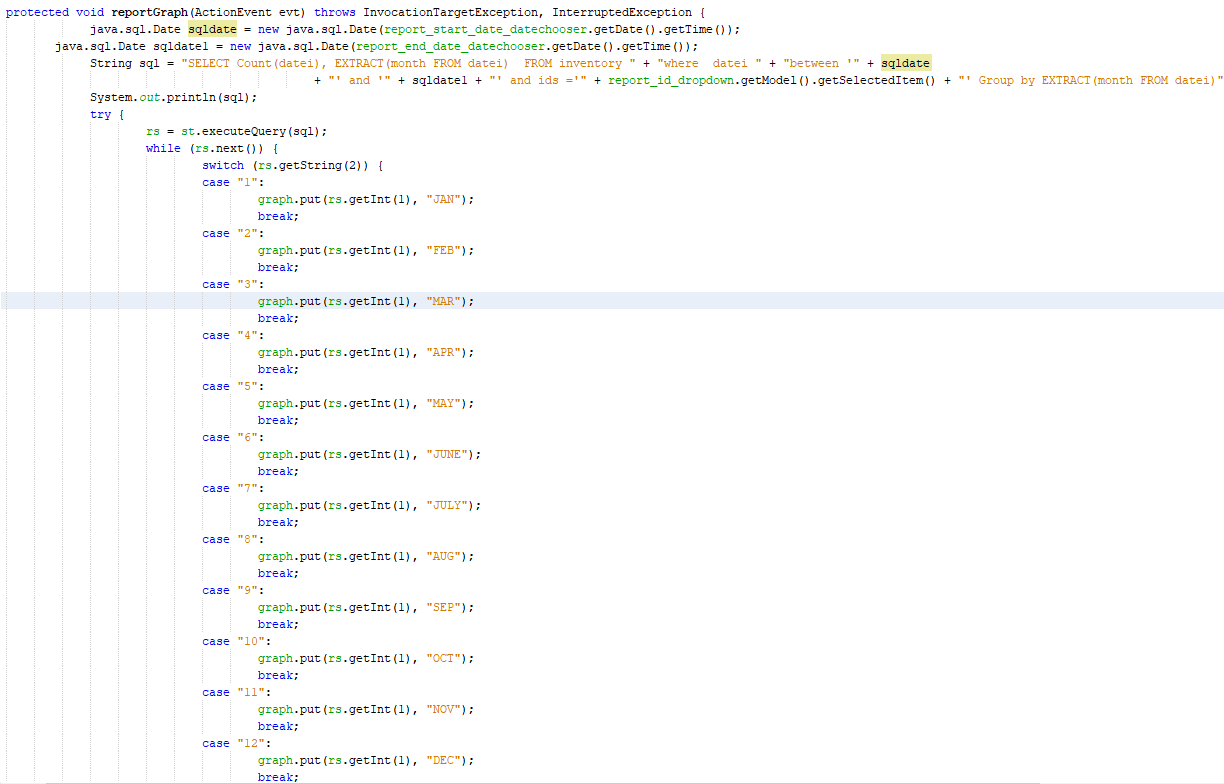


Figure 45 – Code for retrieving data from the database for creating the dataset to create the barchart

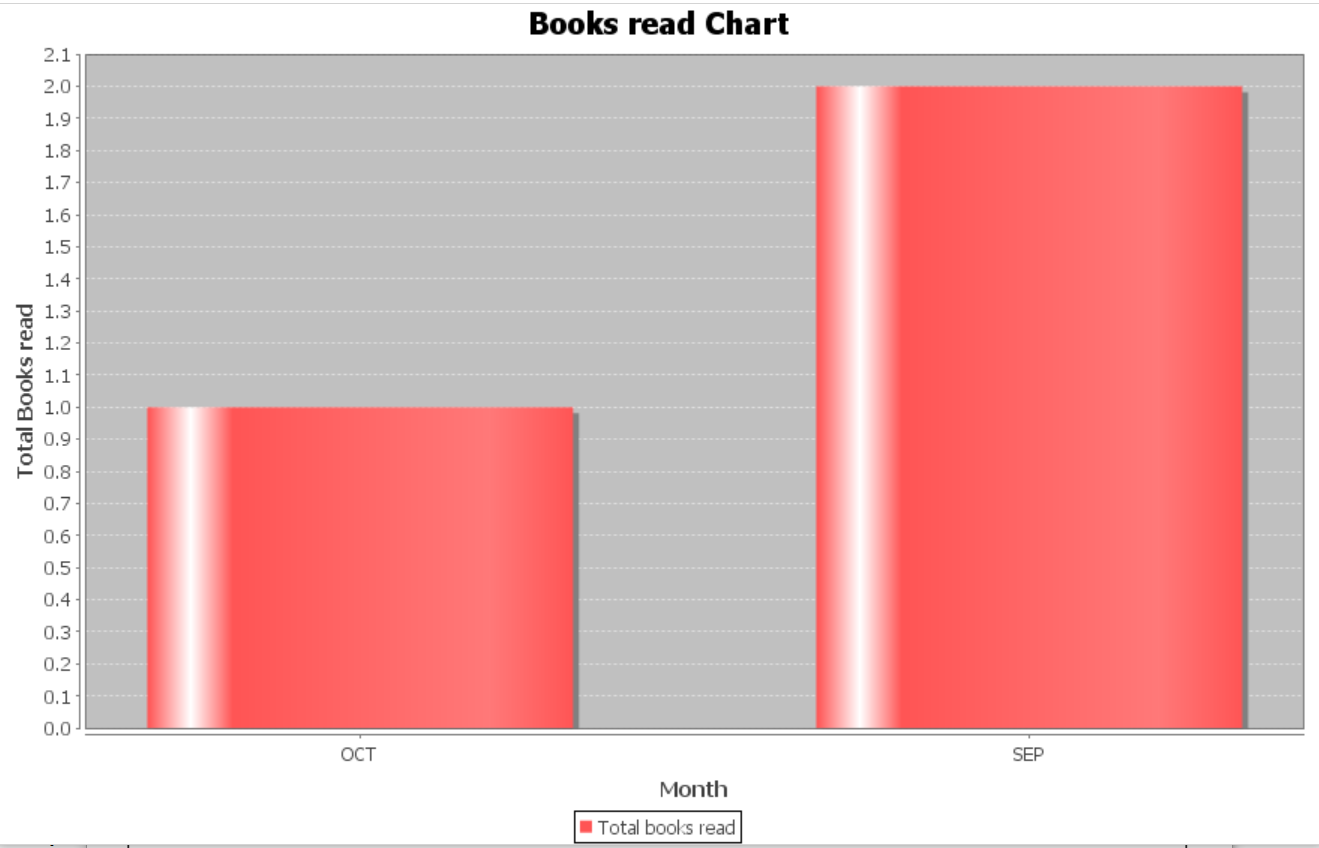


Figure 46 – This is the output chart which is generated

**11)** **Report Generation**

A downloadable and printable report is automatically generated for the client to share the list of the books the student has read over a period of time with the parents of the student. The report is coded using XML with the Jasper Report.

This is the address to the local file

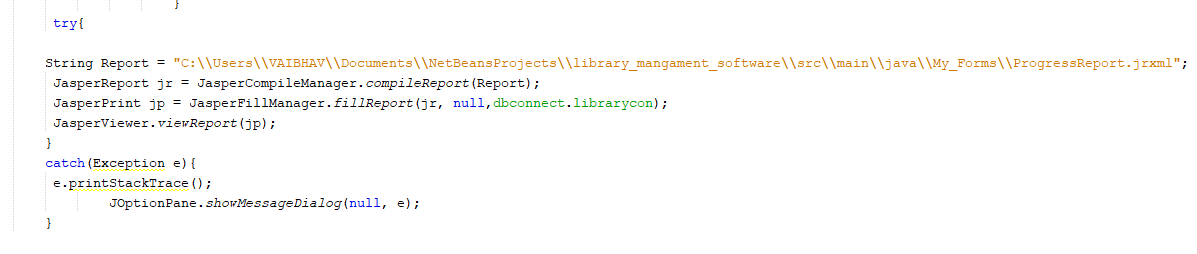


Figure 47 – This is the code for initiating the progress report generation to a local file

Text

Description automatically generated

Figure 48 – This is the code for setting up the layout of the report

Text

Description automatically generated

Figure 49 – This is the code for retrieving the information from the report field to the Progress Report

This statement is used to select data from the database

These statements are the fields of the report table in database

Text

Description automatically generated with medium confidence

Figure 50 – This is the code for the text field for retrieving data from the database

Text

Description automatically generated with medium confidence

Figure 51 – This is the code for the text labels under which the database values are printed with the heading and sub heading of the report.

**Word count - 870**

1. https://www.youtube.com/watch?v=CQMpXGwHeYQ [↑](#footnote-ref-2)
2. https://www.tutorialspoint.com/java/java\_abstraction.htm [↑](#footnote-ref-3)
3. https://www.geeksforgeeks.org/encapsulation-in-java/ [↑](#footnote-ref-4)
4. https://www.simplilearn.com/tutorials/java-tutorial/inheritence-in-java [↑](#footnote-ref-5)
5. https://www.geeksforgeeks.org/bubble-sort/ [↑](#footnote-ref-6)
6. https://www.javatpoint.com/binary-search-in-java [↑](#footnote-ref-7)
7. https://www.programiz.com/java-programming/exception-handling [↑](#footnote-ref-8)
8. https://community.jaspersoft.com/wiki/build-jasper-report-using-jasper-library-using-net-beans-ide [↑](#footnote-ref-9)
9. https://toedter.com/jcalendar/ [↑](#footnote-ref-10)
10. https://www.jfree.org/jfreechart/ [↑](#footnote-ref-11)
11. https://www.youtube.com/watch?v=Nzxqg8I0tcQ [↑](#footnote-ref-12)