Project #2 (Part 3)

Library DB

CSE 3330

By:

Muhammad Muawiz Farooqi

Tahera Fatima

Faith Gutierrez

HONOR CODE

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor Code.

Muhammad Muawiz Farooqi Tahera Fatima Faith Gutierrez

Task 1

-- Query 1:

Add an extra column 'Late' to the Book_Loan table. Values will be 0-for non-late returns, and 1-for late returns. Then update the 'Late' column with '1' for all records that have a return date later than the due date and with '0' for those were returned on time.

```
ALTER TABLE Book_Loans

ADD Late INTEGER;

UPDATE Book_Loans

SET Late = IIF(JULIANDAY(Returned_Date) > JULIANDAY(Due_Date), 1, 0);
```

```
sqlite> SELECT * FROM Book_Loans;
Book_Id  Branch_Id  Card_No  Date_Out    Due_Date    Returned_Date  Late
-------  ---------  -------  ----------  ----------  -------------  ----
1        1          123456   2022-01-01  2022-02-01  2022-02-01     0
2        1          789012   2022-01-02  2022-02-02  2022-02-02     0
3        2          345678   2022-01-03  2022-02-03  2022-02-03     0
4        3          901234   2022-01-04  2022-02-04  2022-02-04     0
5        1          567890   2022-01-05  2022-02-05  2022-02-09     1
6        2          234567   2022-01-06  2022-02-06  2022-02-10     1
7        2          890123   2022-01-07  2022-02-07  2022-03-08     1
8        3          456789   2022-01-08  2022-02-08  2022-03-10     1
9        1          111111   2022-01-09  2022-02-09  2022-02-06     0
10       2          222222   2022-01-10  2022-02-10  2022-02-07     0
11       1          333333   2022-03-01  2022-03-08  2022-02-08     0
12       3          444444   2022-03-03  2022-03-10  2022-03-10     0
13       3          555555   2022-02-03  2022-03-03  2022-02-18     0
14       1          565656   2022-01-14  2022-02-14  2022-03-31     1
15       3          676767   2022-01-15  2022-02-15  2022-02-21     1
16       2          787878   2022-03-05  2022-03-12  2022-02-24     0
17       3          989898   2022-03-23  2022-03-30  2022-03-30     0
18       3          121212   2022-01-18  2022-02-18  2022-02-18     0
19       1          232323   2022-03-24  2022-03-31  2022-03-31     0
20       3          343434   2022-01-21  2022-02-21  2022-02-21     0
21       3          454545   2022-01-24  2022-02-24  2022-02-24     0
sqlite> SELECT changes();
changes()
---------
21
```

-- Query 2:

Add an extra column 'LateFee' to the Library_Branch table, decide late fee per day for each branch and update that column.

-- explanation: the late fee for each branch is set to 3 times the branch's ID

```
ALTER TABLE Library_Branch

ADD LateFee INTEGER;

UPDATE Library_Branch

SET LateFee = rowid * 3;
```

```
sqlite> SELECT * FROM LIBRARY_BRANCH;
Branch_Id  Branch_Name  Branch_Address                  LateFee
---------  -----------  ------------------------------  -------
1          Main Branch  123 Main St, New York, NY 10003  3
2          West Branch  456 West St, Arizona, AR 70622   6
3          East Branch  789 East St, New Jersy, NY 32032  9
sqlite> SELECT changes();
changes()
---------
3
```

-- Query 3:

Create a view vBookLoanInfo that retrieves all information per book loan. The view should have the following attributes:

```
CREATE VIEW vBookLoanInfo AS SELECT Card_No, Name Borrower_Name, Date_Out, Due_Date,
Returned_Date, (JULIANDAY(Returned_Date) - JULIANDAY(Date_Out)) AS TotalDays, Title
Book_Title, (IIF(Late=0, 0, (JULIANDAY(Returned_Date) - JULIANDAY(Due_Date)))) AS
Days_Late, Branch_Id, (IIF(Late=0, 0, (JULIANDAY(Returned_Date) -
JULIANDAY(Due_Date))))*LateFee AS LateFeeBalance

FROM Borrower NATURAL JOIN Book_Loans NATURAL JOIN Book NATURAL JOIN Library_Branch;
```

```
sqlite> SELECT * FROM vBookLoanInfo;
Card_No  Borrower_Name    Date_Out    Due_Date    Returned_Date  TotalDays  Book_Title                          Days_Late  Branch_Id  LateFeeBalance
-------  ---------------  ----------  ----------  -------------  ---------  ---------------------------------  ---------  ---------  --------------
123456   John Smith       2022-01-01  2022-02-01  2022-02-01     31.0       To Kill a Mockingbird              0          1          0
789012   Jane Doe         2022-01-02  2022-02-02  2022-02-02     31.0       1984                               0          1          0
345678   Bob Johnson      2022-01-03  2022-02-03  2022-02-03     31.0       Pride and Prejudice                0          2          0
901234   Sarah Kim        2022-01-04  2022-02-04  2022-02-04     31.0       The Great Gatsby                   0          3          0
567890   Tom Lee          2022-01-05  2022-02-05  2022-02-09     35.0       One Hundred Years of Solitude      4.0        1          12.0
234567   Emily Lee        2022-01-06  2022-02-06  2022-02-10     35.0       Animal Farm                        4.0        2          24.0
890123   Michael Park     2022-01-07  2022-02-07  2022-03-08     60.0       The Catcher in the Rye             29.0       2          174.0
456789   Laura Chen       2022-01-08  2022-02-08  2022-03-10     61.0       Lord of the Flies                  30.0       3          270.0
111111   Alex Kim         2022-01-09  2022-02-09  2022-02-06     28.0       Brave New World                    0          1          0
222222   Rachel Lee       2022-01-10  2022-02-10  2022-02-07     28.0       The Picture of Dorian Gray         0          2          0
333333   William Johnson  2022-03-01  2022-03-08  2022-02-08     -21.0      The Alchemist                      0          1          0
444444   Ethan Martinez   2022-03-03  2022-03-10  2022-03-10     7.0        The God of Small Things            0          3          0
555555   Grace Hernandez  2022-02-03  2022-03-03  2022-02-18     15.0       Wuthering Heights                  0          3          0
565656   Sophia Park      2022-01-14  2022-02-14  2022-03-31     76.0       The Hobbit                         45.0       1          135.0
676767   Olivia Lee       2022-01-15  2022-02-15  2022-02-21     37.0       The Lord of the Rings              6.0        3          54.0
787878   Noah Thompson    2022-03-05  2022-03-12  2022-02-24     -9.0       The Hitchhiker's Guide to the Galaxy  0       2          0
989898   Olivia Smith     2022-03-23  2022-03-30  2022-03-30     7.0        The Diary of a Young Girl          0          3          0
121212   Chloe Park       2022-01-18  2022-02-18  2022-02-18     31.0       The Da Vinci Code                  0          3          0
232323   William Chen     2022-03-24  2022-03-31  2022-03-31     7.0        The Adventures of Huckleberry Finn 0          1          0
343434   Olivia Johnson   2022-01-21  2022-02-21  2022-02-21     31.0       The Adventures of Tom Sawyer       0          3          0
454545   Dylan Kim        2022-01-24  2022-02-24  2022-02-24     31.0       A Tale of Two Cities               0          3          0
sqlite> SELECT count(*) FROM vBookLoanInfo;
count(*)
--------
21
```

Task 2

Requirement 1:

SQL Queries:

Check if enough copies of the book are available:

```
SELECT IIF(No_of_Copies>0, 1, 0) FROM Book_Copies WHERE Book_Id=? and
Branch_Id=?;
```

Check if unique information entered by the user:

```
SELECT EXISTS (SELECT 1 FROM Book_Loans WHERE Book_Id=? and
Branch_Id=? and Card_No=?);
```

If unique information entered, insert the book into the database:

```
INSERT INTO Book_Loans(Book_Id, Branch_Id, Card_No, Date_Out) VALUES
(?, ?, ?, DATE('now'))
```

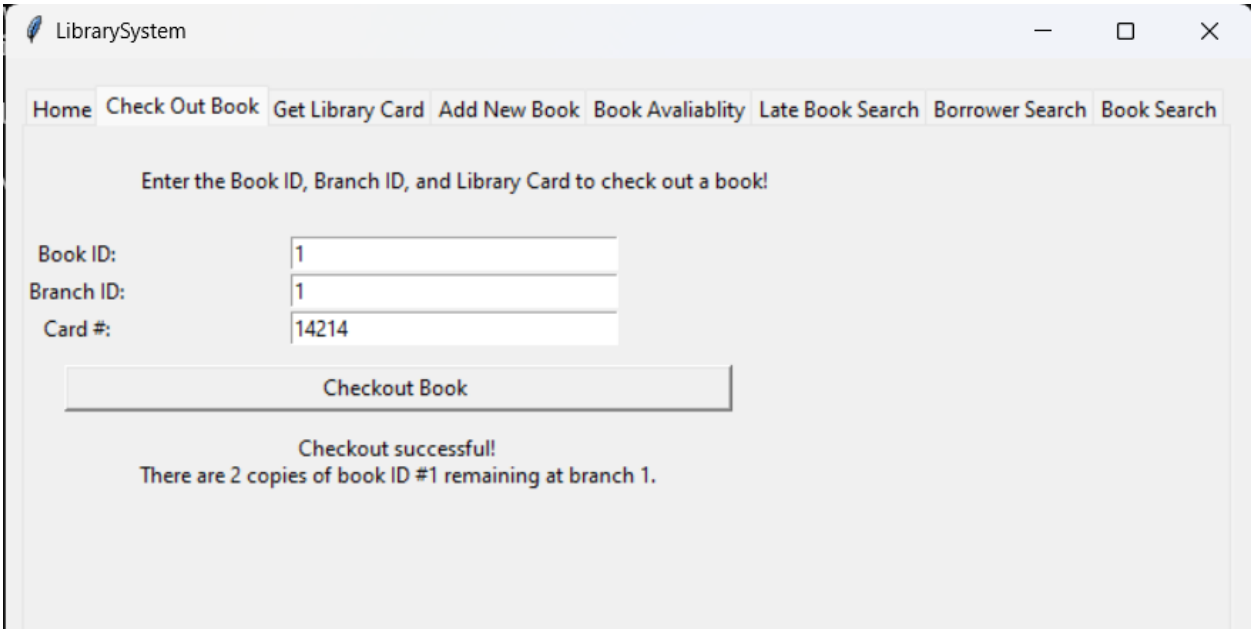
Update the database by decreasing the number of copies at the branch:

```
UPDATE Book_Copies SET No_Of_Copies = No_Of_Copies - 1 WHERE Book_Id=?
AND Branch_Id=?
```

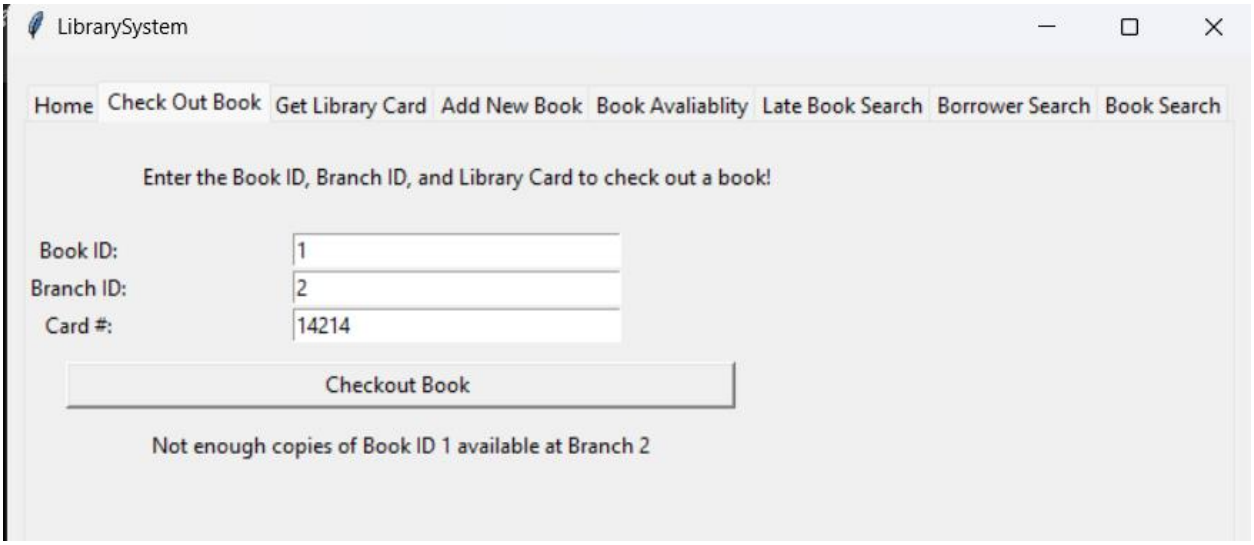
Get the new number of copies left at the branch:

```
SELECT * FROM Book_Copies WHERE Book_Id=? AND Branch_Id=?
```

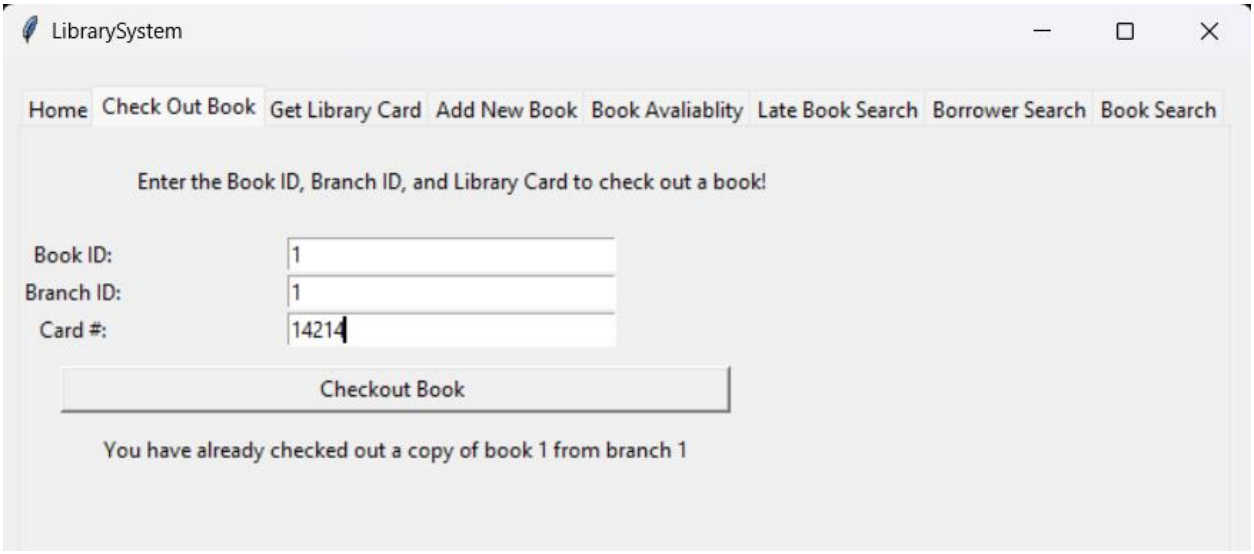
GUI screenshots:



Successful book checkout



Not enough book copies for checkout



Attempted duplicate checkout

Requirement 2:

SQL Queries:

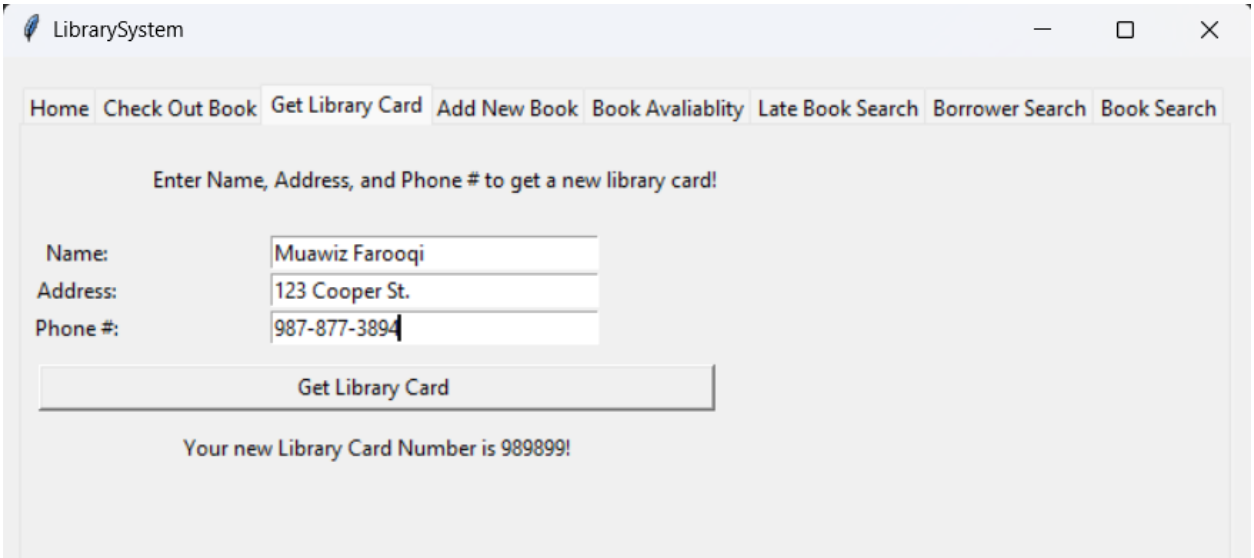
Add a new borrower into the database:

```
INSERT INTO Borrower(Name, Address, Phone) VALUES (?, ?, ?)
```

Get the library card number for the new borrower

```
SELECT Card_No FROM Borrower WHERE Name=? AND Address=? AND Phone=?
```

GUI screenshots:



New library card successfully granted

Requirement 3:

SQL Queries:

Check if user entered a new publisher:

```
SELECT EXISTS (SELECT 1 FROM PUBLISHER WHERE Publisher_Name=? COLLATE
NOCASE);
```

If user entered new publisher, get publisher with correct case:

```
SELECT Publisher_Name FROM PUBLISHER WHERE Publisher_Name=? COLLATE
NOCASE;
```

Add book and publisher into the database:

```
INSERT INTO BOOK(Title, Publisher_Name) VALUES (?, ?);
```

Get the book ID of the newly entered book:

```
SELECT Book_Id FROM BOOK WHERE Title=? and Publisher_Name=?;
```

Check if book already exists in the database:

```
SELECT EXISTS(SELECT 1 FROM Book_Authors WHERE Book_Id=? AND
Author_Name=?);
```

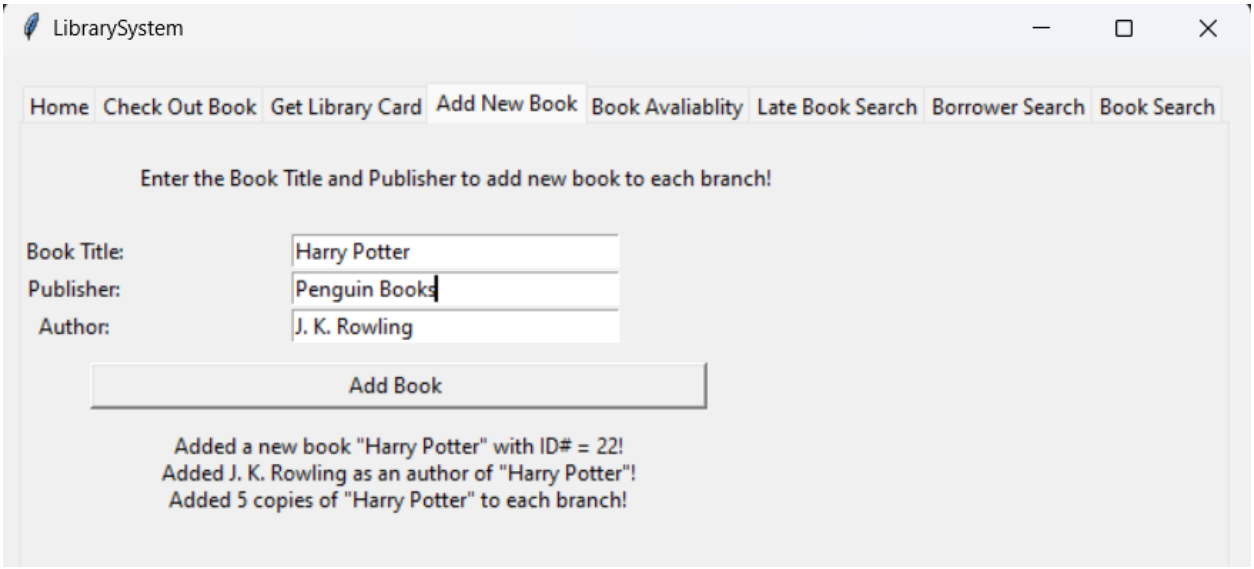
Add author information into the database:

```
INSERT INTO Book_Authors VALUES (?, ?);
```

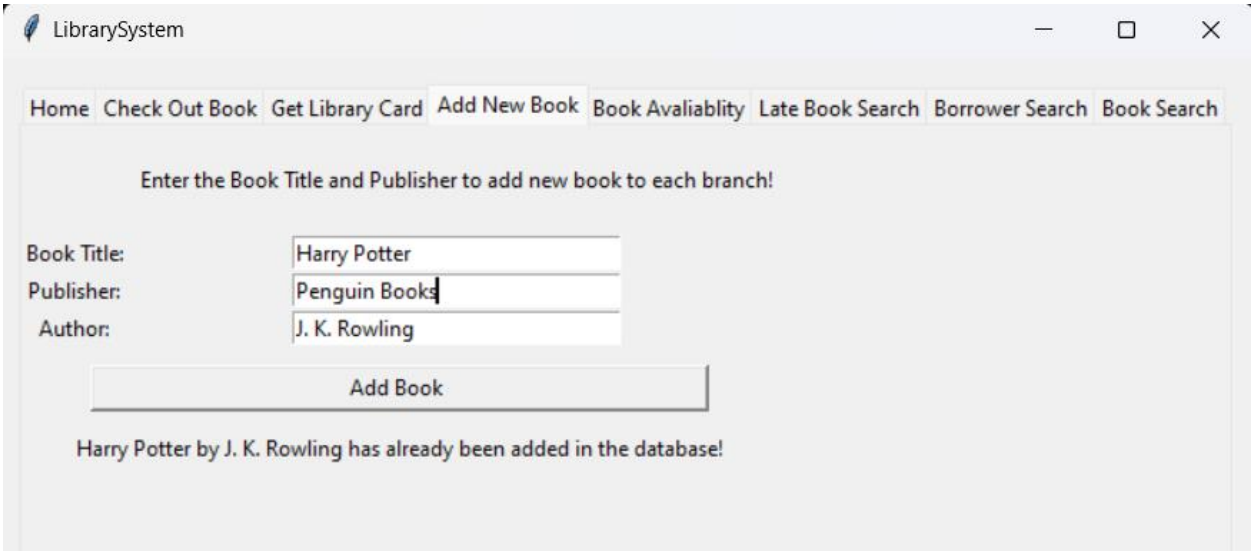
Add 5 copies of the book into each branch in the database:

```
INSERT INTO Book_Copies VALUES (?, 1, 5), (?, 2, 5), (?, 3, 5);
```

GUI screenshots:



Book successfully added



Attempted duplicate book addition

Requirement 4:

SQL Queries:

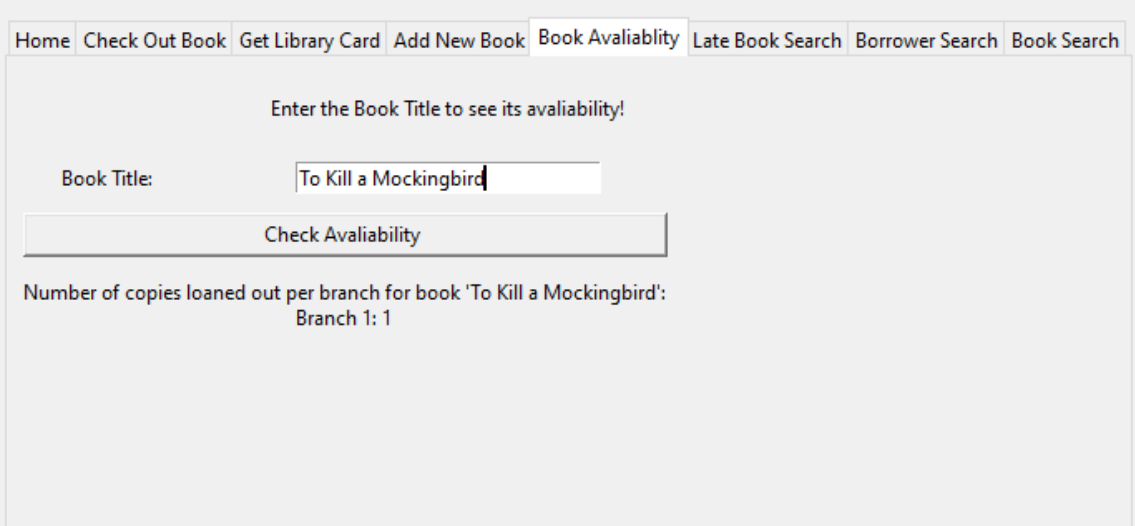
Book id for the given book title (case insensitive):

```
SELECT Book_Id FROM BOOK WHERE Title = ? COLLATE NOCASE
```

Getting the number of copies loaned out per branch:

```
SELECT Branch_Id, COUNT(*) AS Copies_Loaned_Out FROM BOOK_LOANS
WHERE Book_Id = ?;
```

GUI screenshots:



Requirement 5:

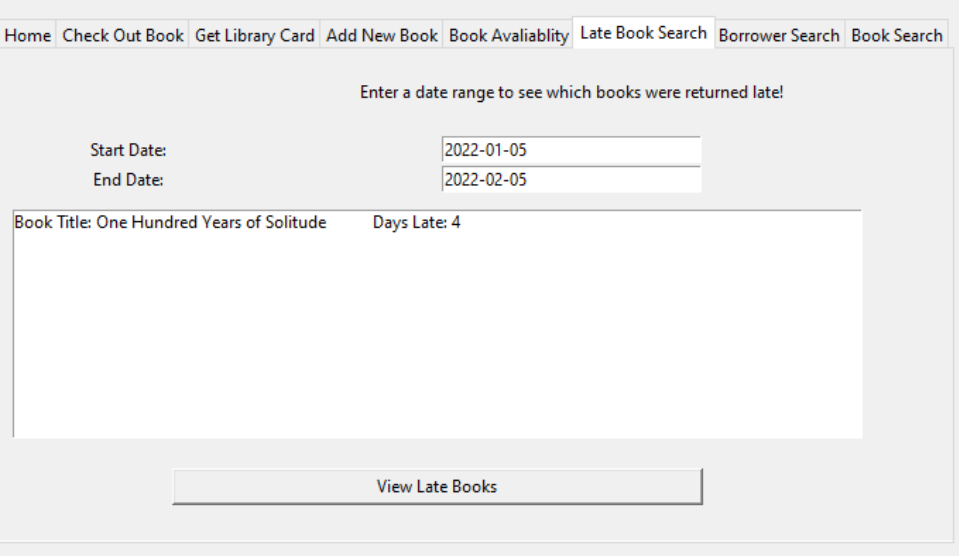
SQL Queries:

```
SELECT bl.book_id,  bl.card_no, bl.Date_Out, bl.due_date,
bl.Returned_Date,

julianday(bl.Returned_Date) - julianday(bl.due_date) AS
days_late

FROM

BOOK_LOANS AS bl

WHERE

bl.Returned_Date > bl.due_date
```

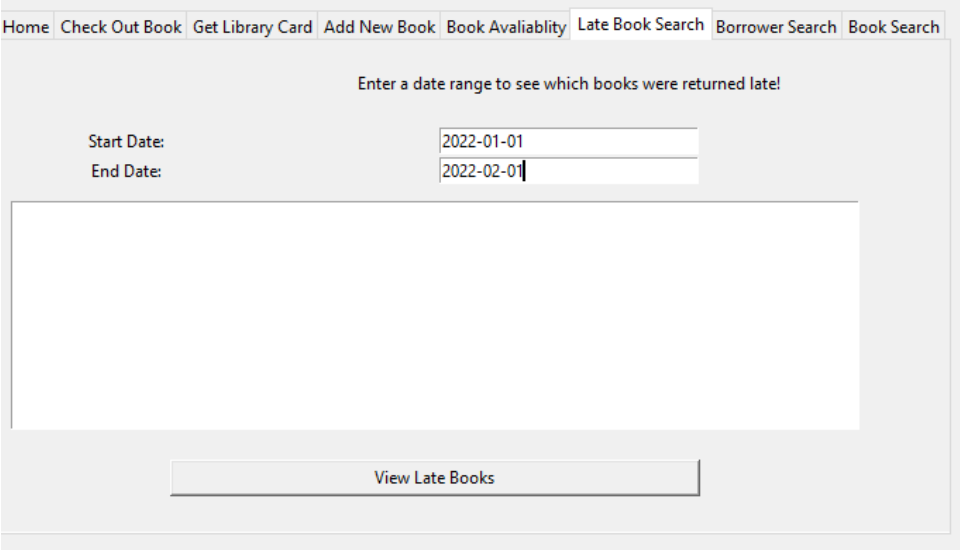
```
AND bl.due_date >= ?
AND bl.due_date <= ?;
```

GUI screenshots:

Date range for a book that was overdue:



Date range for a book that was returned on time:



Since it was on time nothing is returned

Requirement 6a:

SQL Queries:

If user enters id number

```
SELECT Card_No, Borrower_Name, LateFeeBalance FROM vBookLoanInfo
WHERE Card_No = ?;
```

If user enters name and not id number

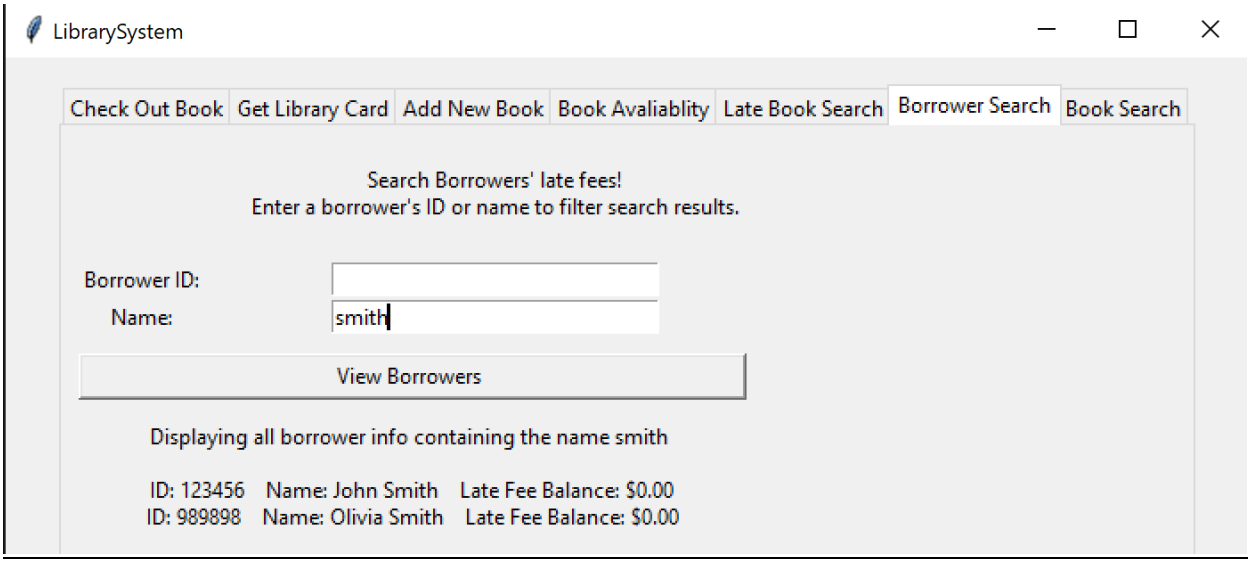
```
SELECT Card_No, Borrower_Name, LateFeeBalance FROM vBookLoanInfo
WHERE Borrower_Name LIKE ? COLLATE NOCASE ORDER BY
LateFeeBalance DESC;
```

If user does not enter name or id number

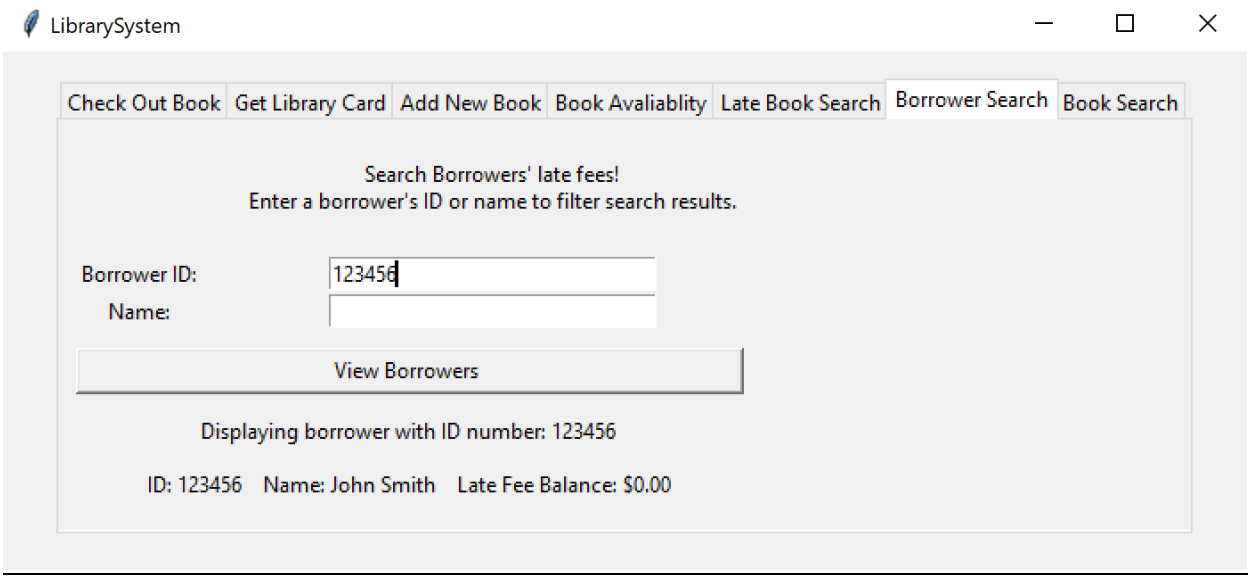
```
SELECT Card_No, Borrower_Name, LateFeeBalance FROM vBookLoanInfo
ORDER BY LateFeeBalance DESC;
```

GUI screenshots:

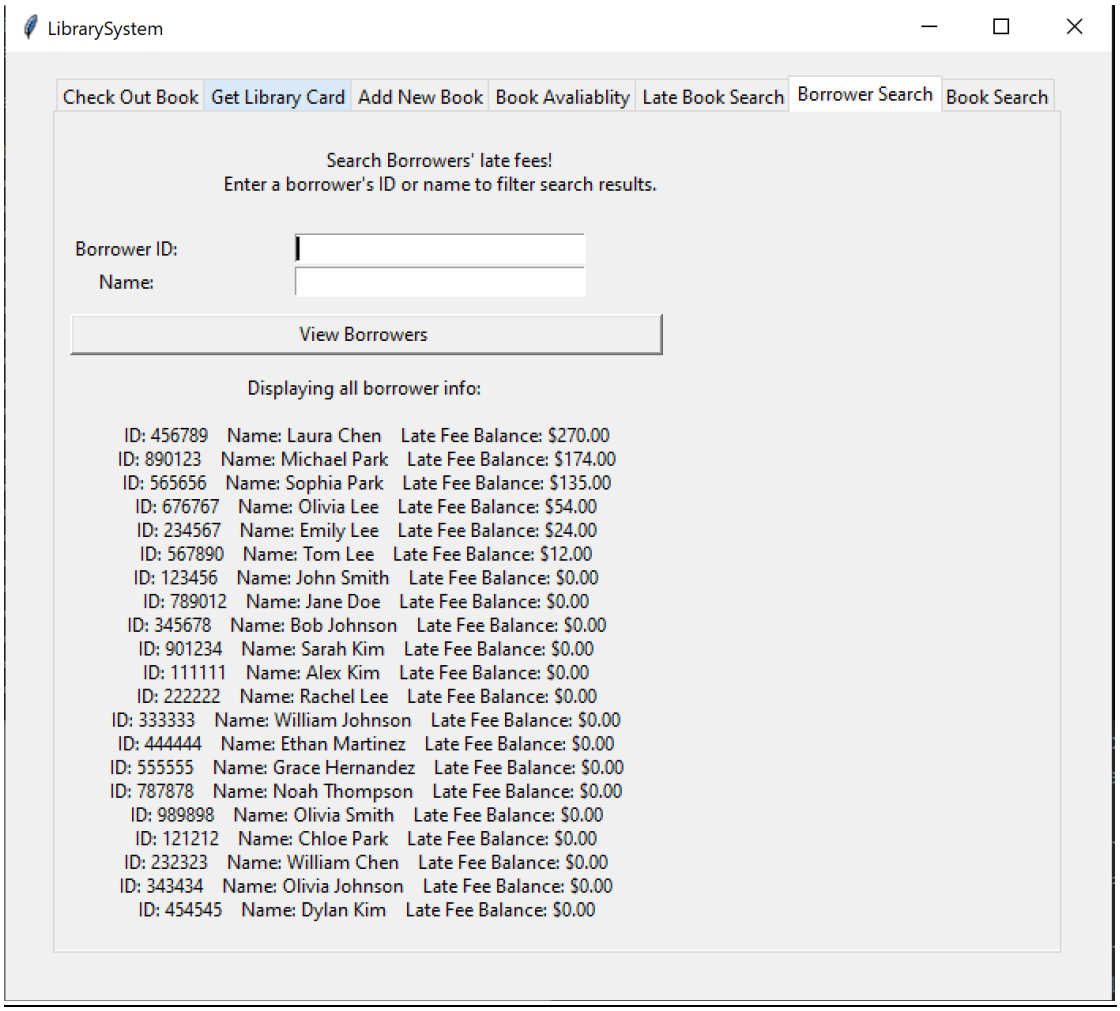
Here, the user can find the library card holders name, card number, and late fee balance. They can…



Search by Name



Search by ID Number



Display all Card Holders

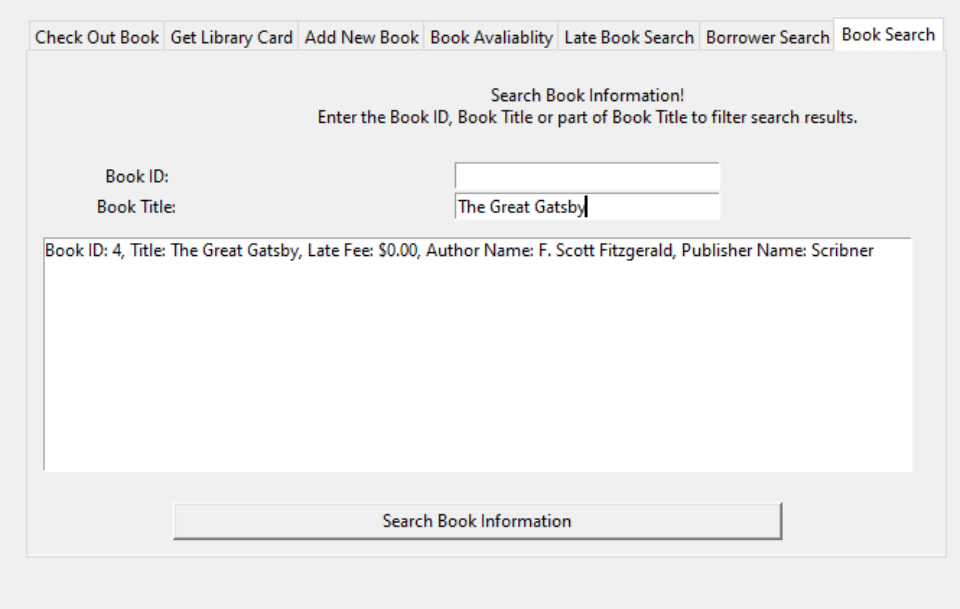
Requirement 6b:

SQL Queries:

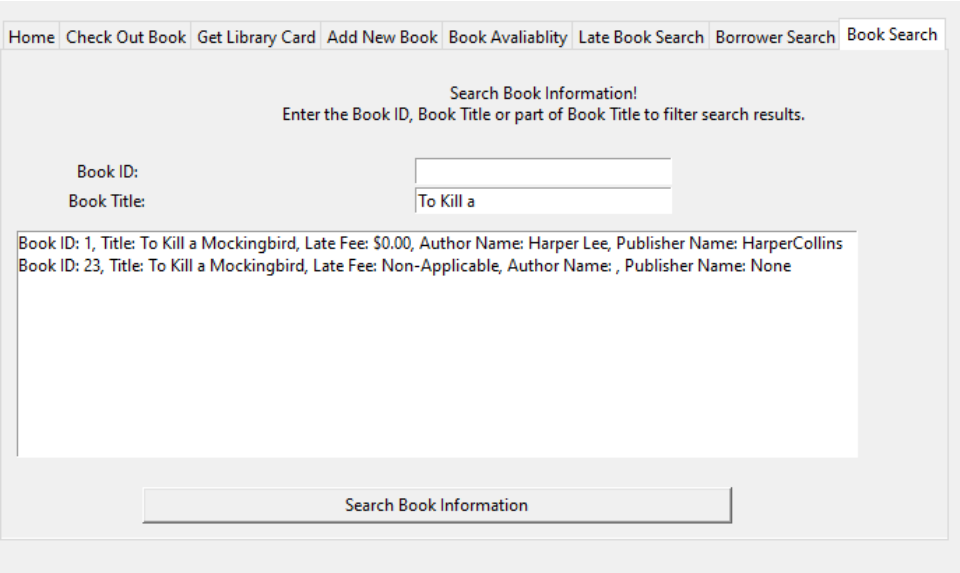
```
SELECT
    book.book_id AS "Book ID", book.title AS "Title",
    CASE
        WHEN book_loans.Returned_Date IS NULL THEN
(julianday('now') - julianday(book_loans.due_date)) * 0.25
        ELSE 0
    END AS "Late Fee",
    book_authors.author_name AS "Author Name",
    publisher.publisher_name AS "Publisher Name"
FROM
    book
    LEFT JOIN BOOK_AUTHORS ON book.book_id =
BOOK_AUTHORS.book_id
    LEFT JOIN book_loans ON book.book_id = book_loans.book_id
    LEFT JOIN publisher ON book.publisher_name =
publisher.publisher_name
WHERE
  book.book_id LIKE ? OR book.title LIKE ? OR book.title LIKE ?
ORDER BY
    "Late Fee" DESC NULLS LAST;
```

GUI screenshots:

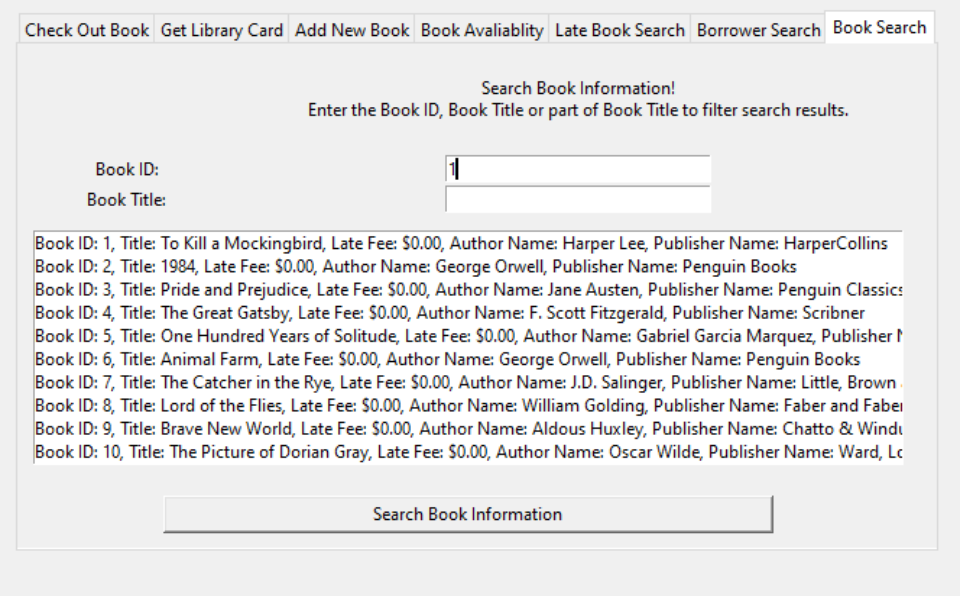
Search by Book Title:



Search by Part of Book Title:



Search by Book ID:



Contributions

Muhammad Muawiz Farooqi

Task 1:

- SQL Queries 1-3

Task 2:

- Implemented interface for requirements 6a and 6b
- Added Button functionality for requirements 1-3
- Created database backup LMS.sql
- SQL queries and GUI screenshots of requirements 1-3.

Tahera Fatima

Task 2:

- Added button functionality for requirements 4, 5, and 6b
- SQL queries for requirements 4,5, and 6b
- GUI screenshots of requirements 4, 5, and 6b

Faith Gutierrez

Task 2:

- Created GitHub repository.
- Implemented tab functionality.
- Implemented interface for requirements 1-5.
- Added button functionality for requirement 6a.