CS353 Database Systems - Spring 2031

Group 24

School Library System

Project URL: https://ozgur-abi.github.io/CS353-SchoolLibraryDatabase/

Ali Doğaç Urkaya - 21903213 - Section 3

Özgür Abi - 21902358 - Section 3

Jankat Berslan Dinçer - 21902035 - Section 3

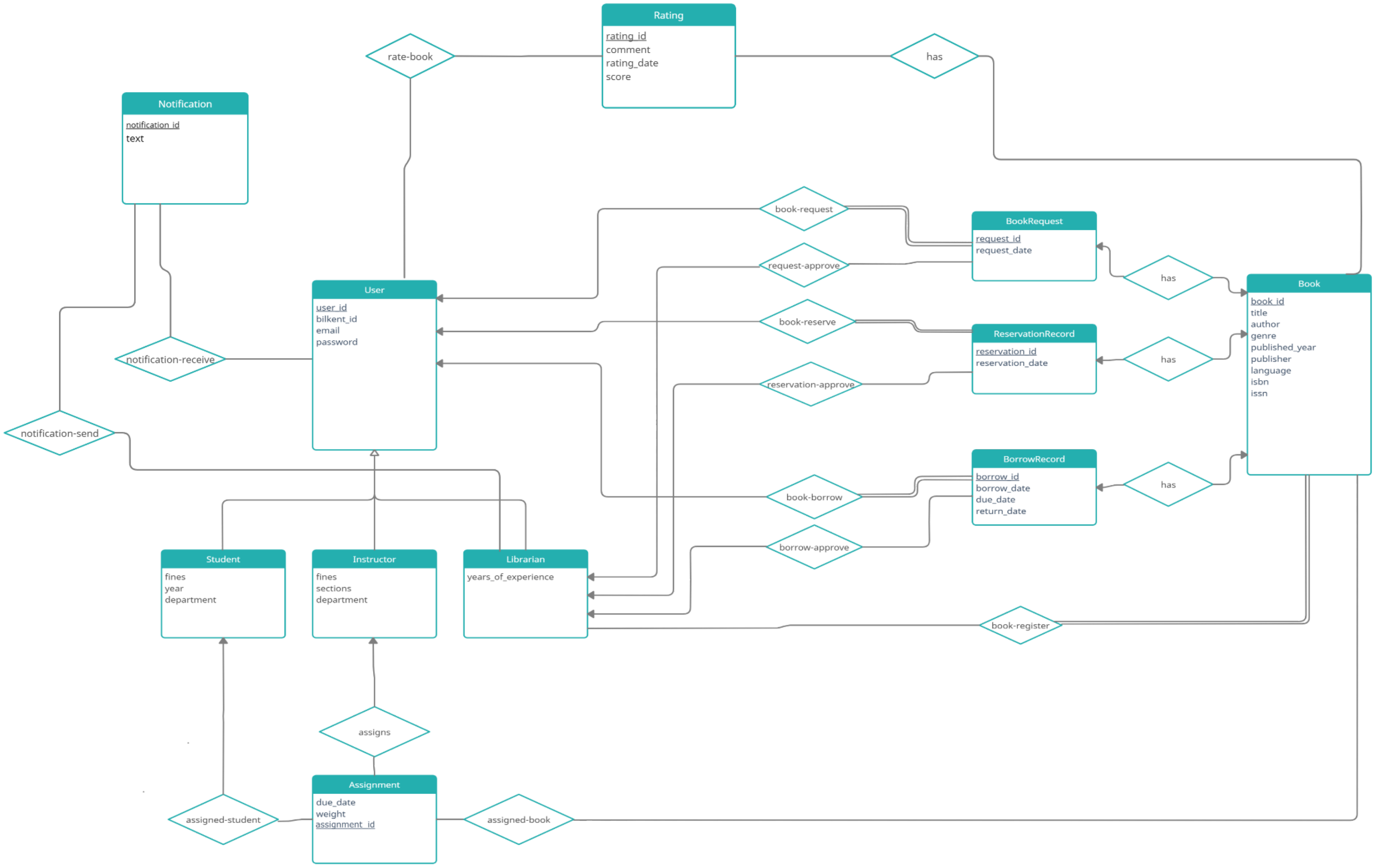
Serhat Gürgenyatağı - 21903288 - Section 3

Table of Contents

1. Revised E/R Model

We have made the following changes in our E/R diagram after getting feedback from the teaching assistant:

- Redesigned many of the relationships between entities to reduce unnecessary complexity.
- Reduced the number of ternary relationships and instead converted them into binary relationships with additional entities.
- Added primary keys for several entities that were lacking them such as Notifications.
- Redesigned the assignment system, added the Assignment entity.
- Redesigned the rating system.
- Removed foreign keys from the diagram.
- Converted some relationships into total participation relationships, fixed some other errors with relationships.
- Added missing attributes to some entities such as Students, Instructors, Librarians etc.
- Made the user types disjoint instead of overlapping.
- Removed the view relationship between notifications and users.


Rating
rating_id
comment
rating_date
score
rate-book
has
Notification
notification_id
text
book-request
BookRequest
request_id
request_date
request-approve
has
User
user_id
bilkent_id
email
password
book-reserve
ReservationRecord
reservation_id
reservation_date
has
Book
book_id
title
author
genre
published_year
publisher
language
isbn
issn
notification-receive
reservation-approve
notification-send
BorrowRecord
borrow_id
borrow_date
due_date
return_date
book-borrow
has
borrow-approve
Student
fines
year
department
Instructor
fines
sections
department
Librarian
years_of_experience
book-register
assigns
Assignment
due_date
weight
assignment_id
assigned-student
assigned-book

2. Table/Relation Schemas

2.1 User

Relational model:
user(<u>user_id</u>, first_name, last_name, bilkent_id, email, password)

Candidate keys:
{ (user_id), (bilkent_id) }

Primary key:
(user_id)

Table Definition:
```
CREATE TABLE user(
     user_id    char(11) PRIMARY KEY,
     first_name varchar(20),
     last_name  varchar(20),
     bilkent_id char(8),
     email      varchar(100),
     password   varchar(100));
```

3NF:
First name, last name, bilkent id, email and password are unique to the user and can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.2 Student

Relational model:
student(user_id, fines, year, department)

Candidate keys:
{ (user_id) }

Primary key:
(user_id)

Foreign keys:
{ (user_id) }

Table Definition:
```
CREATE TABLE student(
     user_id    char(11) PRIMARY KEY references user,
     fines      double(5, 2),
     year       int(3),
     department varchar(50));
```

3NF:
Fines year and department can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.3 Instructor

Relational model:
instructor(user_id, fines, sections, department)

Candidate keys:
{ (user_id) }

Primary key:
(user_id)

Foreign keys:
{ (user_id) }

Table Definition:
```
CREATE TABLE instructor(
     user_id    char(11) PRIMARY KEY references user,
     fines      double(5, 2),
     sections   varchar(500),
     department varchar(50));
```

3NF:
Fines and sections can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.4 Librarian

Relational model:
instructor(<u>user_id</u>, years_of_experience)

Candidate keys:
{ (user_id) }

Primary key:
(user_id)

Foreign keys:
{ (user_id) }

Table Definition:
```
CREATE TABLE instructor(
      user_id           char(11) PRIMARY KEY references user,
      years_of_experience   int(32));
```
3NF:
There are only two attributes.This relation is in 3NF. All attributes are dependent on the primary key.

2.5 Book

Relational model:
book(<u>book_id</u>, title, author, genre, published_year, publisher, language, isbn, issn)

Candidate keys:
{ (book_id), (isbn), (issn) }

Primary key:
(book_id)

Table Definition:
```
CREATE TABLE book(
     book_id          char(11) PRIMARY KEY references user,
     title            varchar(500),
     author           varchar(500),
     genre            varchar(500),
     published_year   int(32),
     publisher        varchar(500),
     language         varchar(500),
     isbn             varchar(500),
     issn             varchar(500));
```
3NF:
Title, author, genre, published year, publisher, language, isbn and issn can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.6 Book Request

Relational model:
book_request(<u>request_id</u>, request_date, book_id, requester_id, approver_id)

Candidate keys:
{ (request_id) }

Primary key:
(request_id)

Table Definition:
```
CREATE TABLE book_request(
     request_id      char(11) PRIMARY KEY references user,
     request_date         varchar(10),
     book_id         varchar(11) references book,
     requester_id         varchar(11) references user,
     approver_id          varchar(11) references librarian);
```

3NF:
request date, book id, requester id and approver id can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.7 Reservation Record

Relational model:
book_reservation(<u>reservation_id</u>, reservation_date, book_id, reserver_id, approver_id)

Candidate keys:
{ (reservation_id) }

Primary key:
(reservation_id)

Table Definition:
```
CREATE TABLE book_reservation(
     reservation_id       char(11) PRIMARY KEY references user,
     reservation_date     varchar(10),
     book_id        varchar(11) references book,
     reserver_id          varchar(11) references user,
     approver_id          varchar(11) references librarian);
```

3NF:
reservation date, book id, reserver id and approver id can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.8 Borrow Record

Relational model:
book_borrow(borrow_id, borrow_date, due_date, return_date, book_id, borrower_id, approver_id)

Candidate keys:
{ (borrow_id) }

Primary key:
(borrow_id)

Table Definition:
```
CREATE TABLE book_reservation(
     borrow_id       char(11) PRIMARY KEY references user,
     borrow_date          varchar(10),
     due_date        varchar(10),
     return_date          varchar(10),
     book_id         varchar(11) references book,
     borrower_id          varchar(11) references user,
     approverr_id         varchar(11) references librarian);
```
3NF:
Borrow date, due date, return date, book id, borrower id and approver id can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.9 Book Rating

Relational model:
book_rating(<u>rating_id</u>, rating_date, book_id, rater_id, score, comment)

Candidate keys:
{ (rating_id) }

Primary key:
(rating_id)

Table Definition:
```
CREATE TABLE book_rating(
     rating_id          char(11) PRIMARY KEY references user,
     rating_date             varchar(10),
     book_id            varchar(11) references book,
     rater_id           varchar(11) references user,
     score              int(32),
     comment            varchar(5000));
```

3NF:
Rating date, book id, rater id, score and comment can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.10 Notification

Relational model:
notification(notification_id, notification_date, sender_id, receiver_id, text)

Candidate keys:
{ (notification_id) }

Primary key:
(notification_id)

Table Definition:
```
CREATE TABLE book_rating(
     notification_id      char(11) PRIMARY KEY references user,
     notification_date    varchar(10),
     sender_id       varchar(11) references librarian,
     receiver_id          varchar(11) references user,
     text            varchar(5000));
```

3NF:
notification date, sender id, receiver id and text can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

2.11 Assignment

Relational model:
assignment(assignment_id, due_date, instructor_id, student_id, assigned_book_id, weight)

Candidate keys:
{ (assignment_id) }

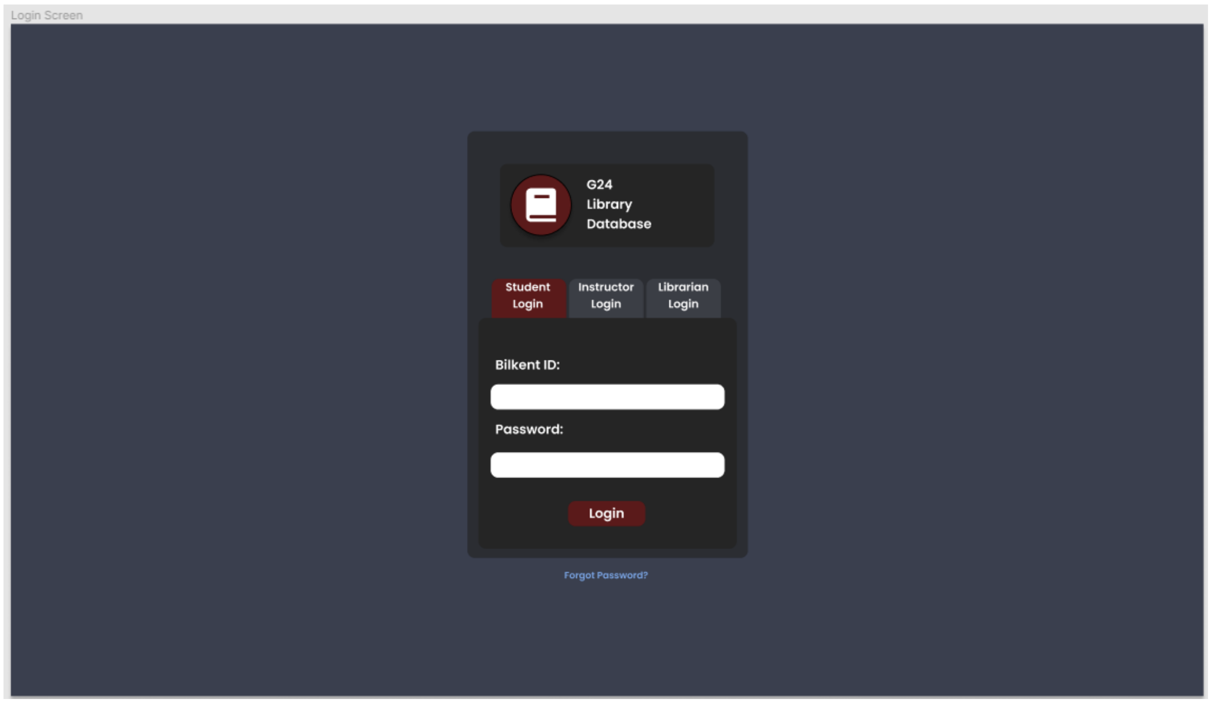
Primary key:
(assignment_id)

Table Definition:
```
CREATE TABLE assignment(
	assignment_id		char(11) PRIMARY KEY references user,
	due_date		varchar(10),
	instructor_id		varchar(11) references instructor,
	student_id		varchar(11) references student,
	assigned_book_id	varchar(11) references book,
	weight			varchar(500));
```

3NF:
Due date, instructor id, student id, assigned book id and weight can not be derived from one another.This relation is in 3NF. All attributes are dependent on the primary key.

3. UI Design and SQL Statements

3.1 Login



SQL Statements
Inputs: @bilkentid, @password

```
SELECT *
from user
where bilkent_id= @bilkentid and password = @password;
```

3.2 Register

SQL Statements

Common Inputs: @bilkentid, @password, @email, @generateduserid

First check if a user with the same Bilkent ID or email already exists

```
SELECT *
from users
where bilkent_id = @bilkentid or email = @email;
```

If result is empty, continue the registration

```
INSERT into user
values (@user_id, @bilkent_id, @email, @password);
```

For Student Registration
Specific Inputs: @year, @department

```
INSERT into student
values (@user_id, 0, @year, @department);
```

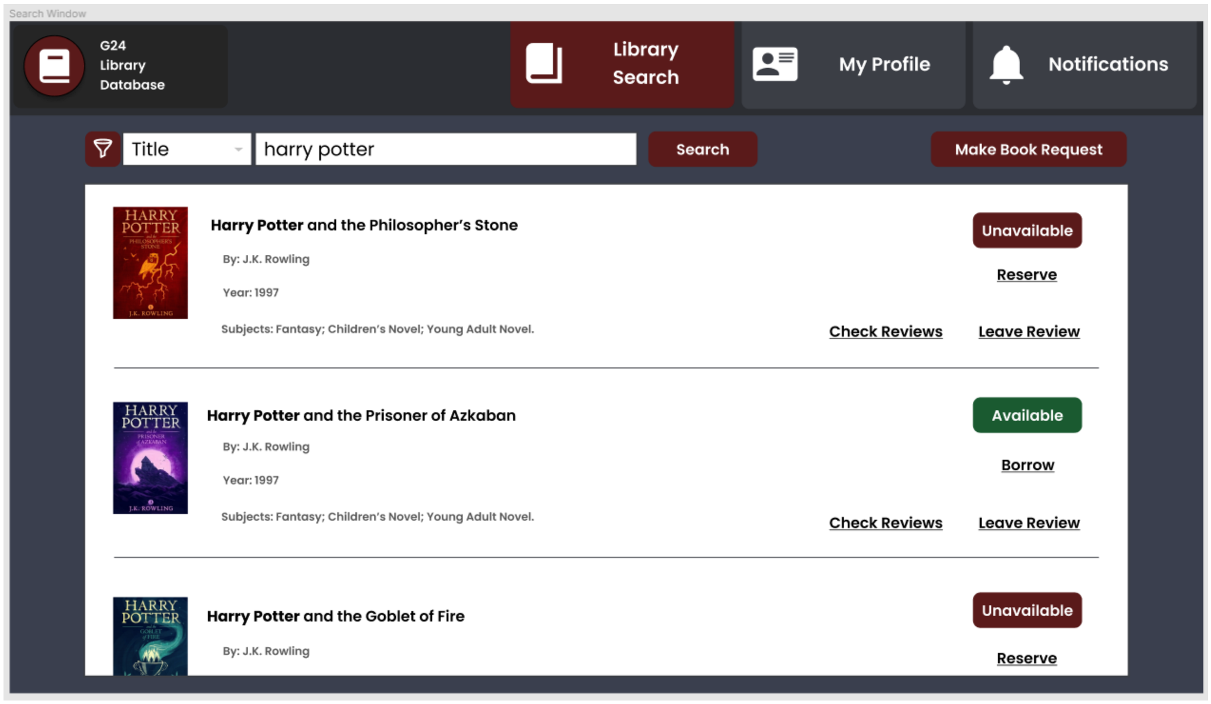
For Instructor Registration
Specific Inputs: @sections, @department

```
INSERT into instructor
values (@user_id, 0, @sections, @department);
```

For Librarian Registration

```
INSERT into librarian
values (@user_id, 0);
```

3.3 Search the book by criteria and apply filters



Harry Potter and the Philosopher's Stone
By: J.K. Rowling
Year: 1997
Subjects: Fantasy; Children's Novel; Young Adult Novel.
Unavailable
Reserve
Check Reviews
Leave Review

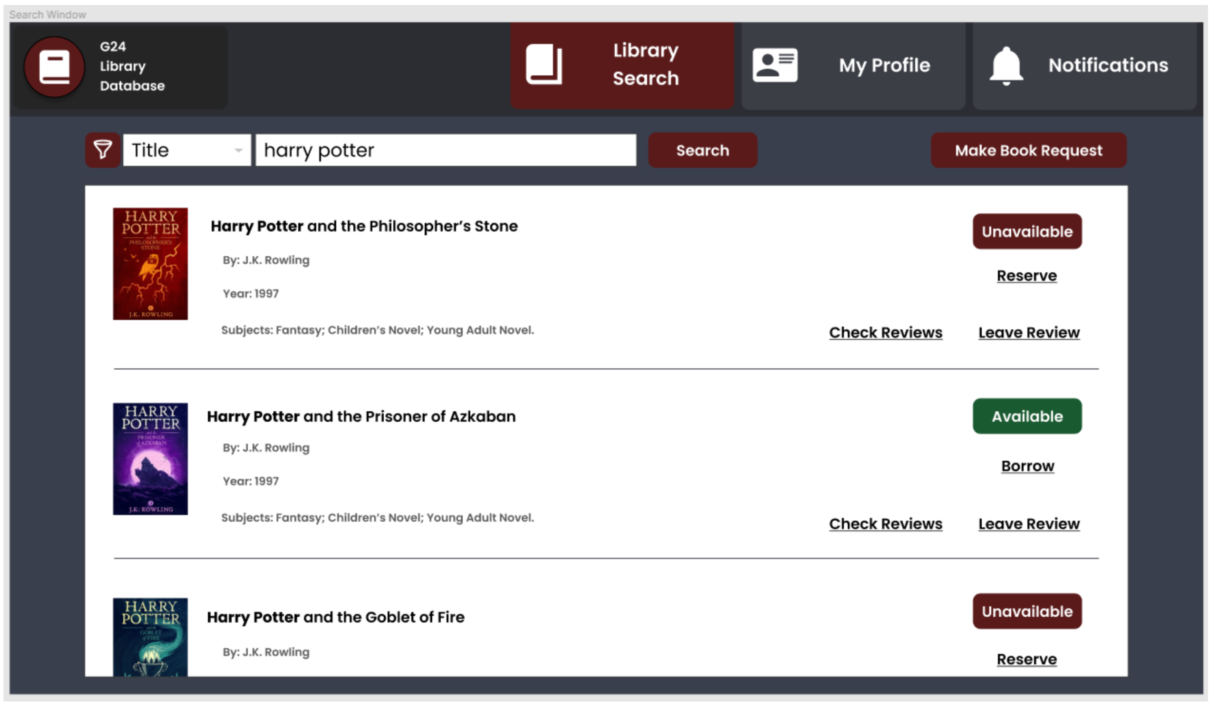
Harry Potter and the Prisoner of Azkaban
By: J.K. Rowling
Year: 1997
Subjects: Fantasy; Children's Novel; Young Adult Novel.
Available
Borrow
Check Reviews
Leave Review

Harry Potter and the Goblet of Fire
By: J.K. Rowling
Unavailable
Reserve

SQL Statements
Inputs (Example): @title, @betweenyear1, @betweenyear2, @author

```
SELECT *
from book
where title = @title and year >= @betweenyear1 and year <=
@betweenyear2 and author = @author;
```

3.4 Select the book and put a hold (send borrow request) (by the user)



Harry Potter and the Philosopher's Stone
By: J.K. Rowling
Year: 1997
Subjects: Fantasy; Children's Novel; Young Adult Novel.

Harry Potter and the Prisoner of Azkaban
By: J.K. Rowling
Year: 1997
Subjects: Fantasy; Children's Novel; Young Adult Novel.

Harry Potter and the Goblet of Fire
By: J.K. Rowling

The available / unavailable information will be retrieved using the following SQL query:
Inputs: @bookid

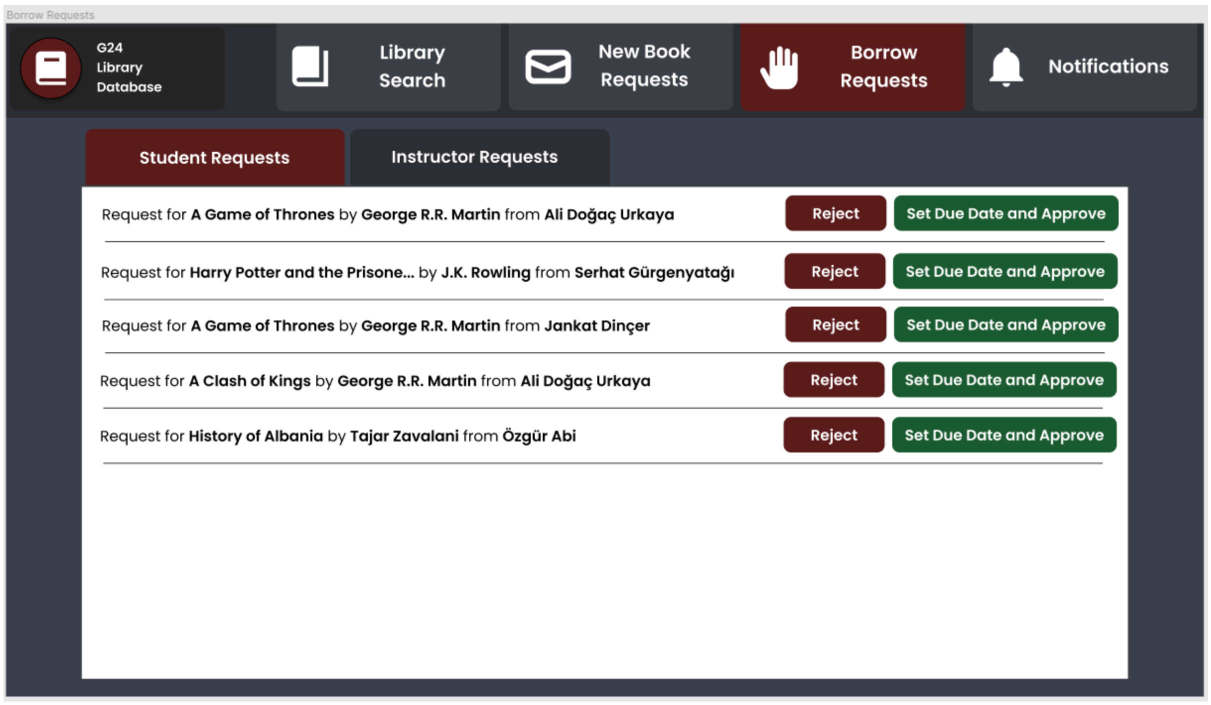
```
SELECT *
from book_borrow
where book_id = @bookid and return_date IS NULL and approver_id IS
NOT NULL;
```

If the result is empty, that means book is available. Otherwise return_date would not be null. When return_date is null and approver_id is not null, it means that a borrow request has been approved and book has not been returned yet.

Then, borrow request will be sent using the following SQL query (Borrow id will be generated):
Inputs: @generatedborrowid, @bookid, @borrowdate, @borrowerid)

```
INSERT into book_borrow
values (@generatedborrowid, @borrowdate, NULL, NULL, @bookid,
@borrowerid, NULL);
```

3.5 Borrow operation (by the librarian)



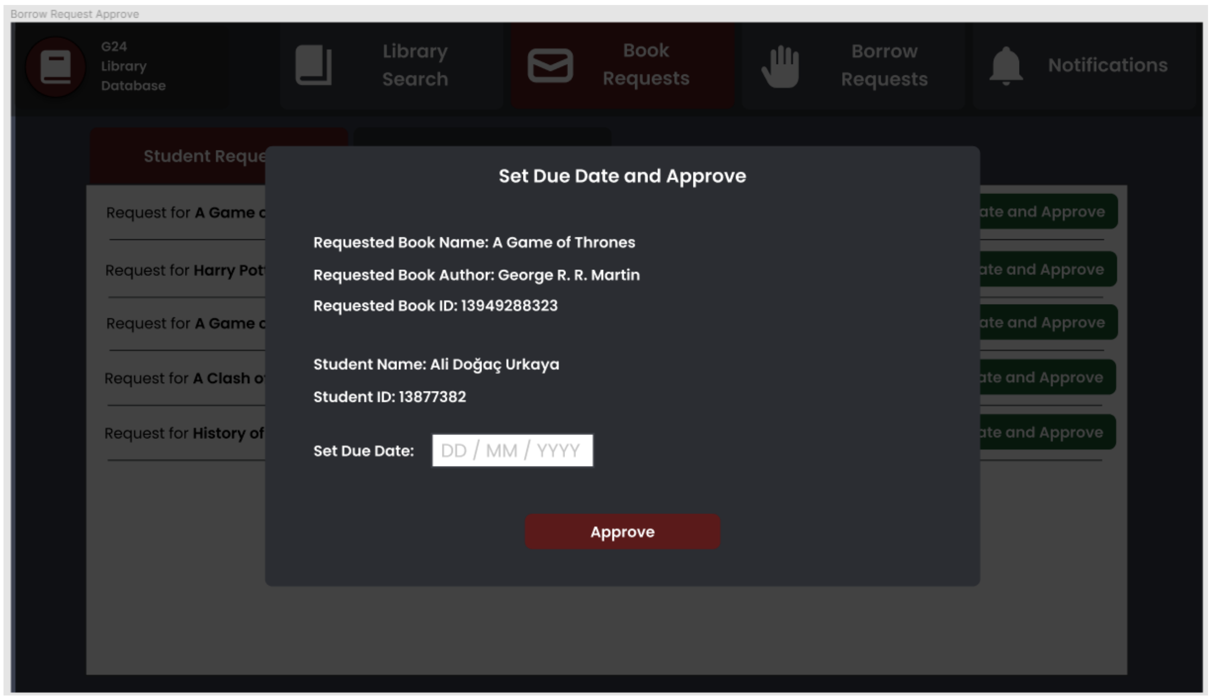
Request for **A Game of Thrones** by **George R.R. Martin** from **Ali Doğaç Urkaya** Reject Set Due Date and Approve

Request for **Harry Potter and the Prisone...** by **J.K. Rowling** from **Serhat Gürgenyatağı** Reject Set Due Date and Approve

Request for **A Game of Thrones** by **George R.R. Martin** from **Jankat Dinçer** Reject Set Due Date and Approve

Request for **A Clash of Kings** by **George R.R. Martin** from **Ali Doğaç Urkaya** Reject Set Due Date and Approve

Request for **History of Albania** by **Tajar Zavalani** from **Özgür Abi** Reject Set Due Date and Approve



Set Due Date and Approve

Requested Book Name: A Game of Thrones
Requested Book Author: George R. R. Martin
Requested Book ID: 13949288323

Student Name: Ali Doğaç Urkaya
Student ID: 13877382

Set Due Date: DD / MM / YYYY

Approve

If librarian approves the borrow request, following SQL query is executed:
Inputs: @bookid, @userid, @duedate, @approverid

```
UPDATE book_borrow
set due_date = @duedate, approver_id = @approverid
where book_id = @bookid and borrower_id = @userid;
```

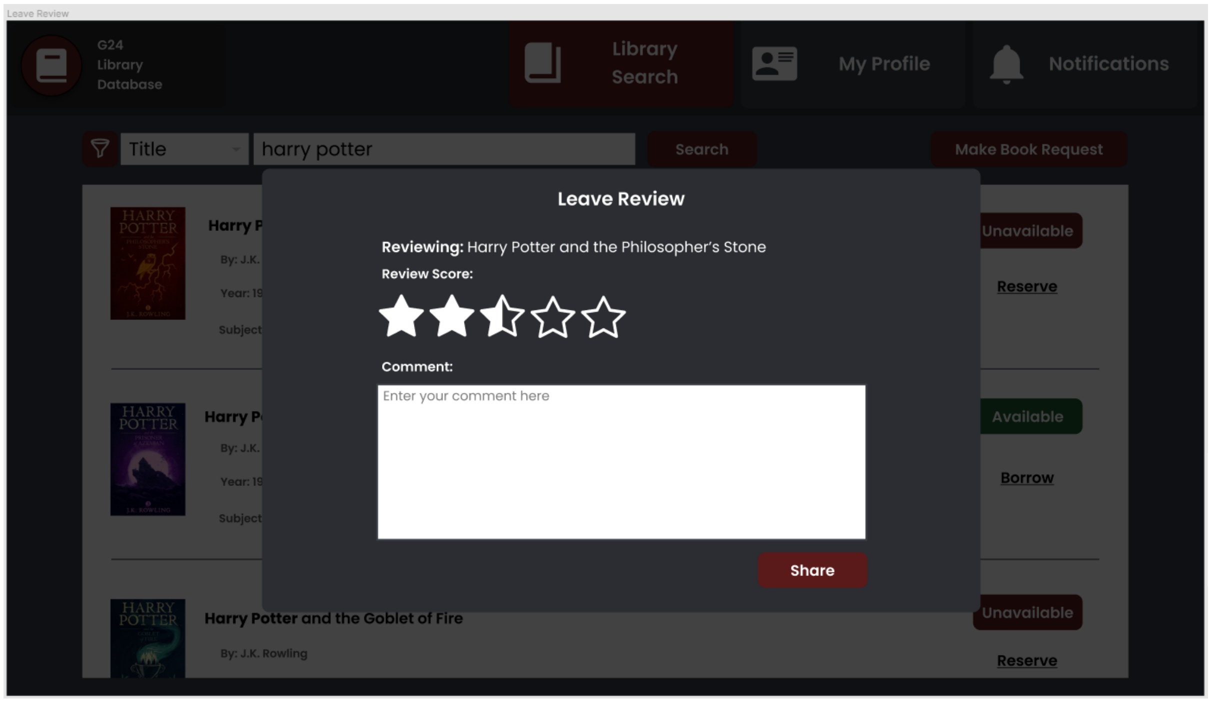
If the librarian denies the borrow request, following SQL query is executed:

```
DELETE
from book_borrow
where book_id = @bookid and borrower_id = @userid;
```

User’s borrowed book list can be retrieved using the following SQL query:
Inputs: @userid

```
SELECT *
from book_borrow
where borrowerid = @userid;
```

3.6 Rate a book (Additional Functional Requirement)


Leave Review

G24 Library Database | Library Search | My Profile | Notifications

Title | harry potter | Search | Make Book Request

Leave Review

Reviewing: Harry Potter and the Philosopher's Stone

Review Score:

Comment:

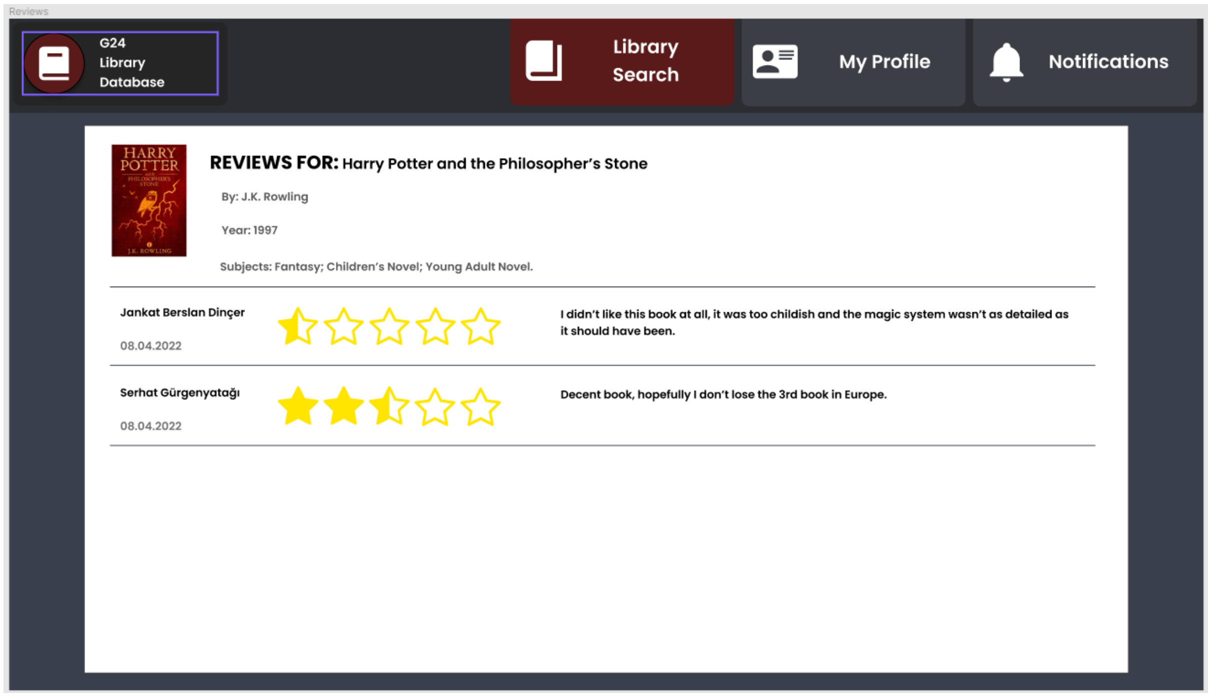
Enter your comment here

Share

Harry P… By: J.K. … Year: 19… Subject… Unavailable Reserve

Harry P… By: J.K. … Year: 19… Subject… Available Borrow

Harry Potter and the Goblet of Fire By: J.K. Rowling Unavailable Reserve


Reviews

G24 Library Database | Library Search | My Profile | Notifications

REVIEWS FOR: Harry Potter and the Philosopher's Stone

By: J.K. Rowling

Year: 1997

Subjects: Fantasy; Children's Novel; Young Adult Novel.

Jankat Berslan Dinçer
08.04.2022
I didn't like this book at all, it was too childish and the magic system wasn't as detailed as it should have been.

Serhat Gürgenyatağı
08.04.2022
Decent book, hopefully I don't lose the 3rd book in Europe.

SQL Statements
Inputs: @userid, @bookid, @generatedratingid, @score, @comment, @rating_date

```
INSERT into book_rating
values (@generatedratingid, @rating_date, @bookid, @userid,
@score, @comment)
```