Bilkent University
CS 353
Design Report

Group 12
Social Networking for Readers

Ahmet Tuna Baykal - 21803279
Cemil Mert Özdemir - 21803303
Efe Karaköylü - 21901510
Efe Kerem Kesgin - 21902857

1.Revised ER Diagram

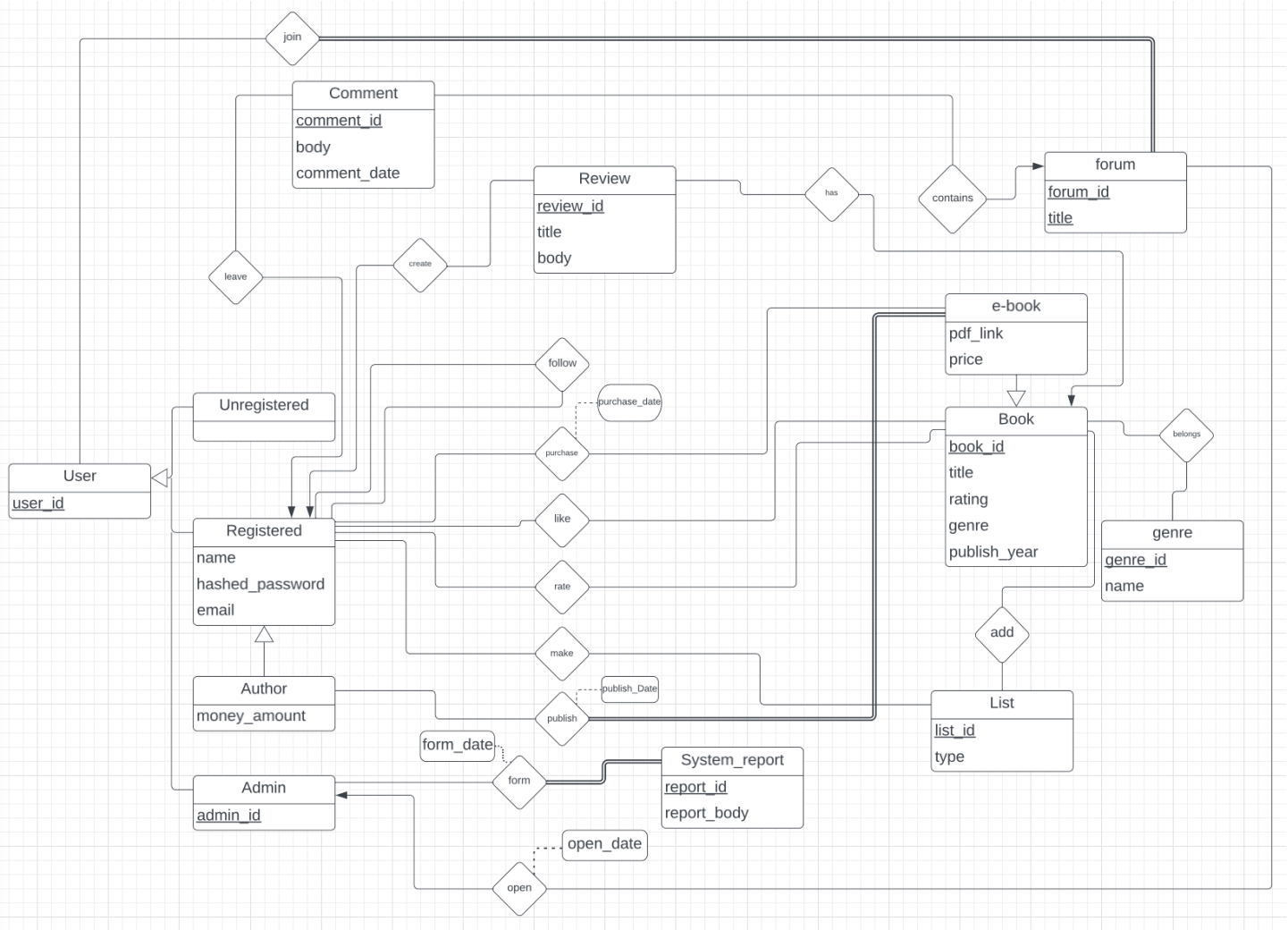


Figure 1: Revised ER Diagram

2.Database Schema

2.1 user

Creation

```
"CREATE TABLE user( " +
```

```
        " user_id VARCHAR(5) primary key" +
        ") engine=innodb;";
```

Relational Schema
user(user_id: int)

Candidate Keys
None

2.2 registered

Creation

```
 "CREATE TABLE registered( " +
        "user_id VARCHAR(5) primary key," +
         "name VARCHAR(20) ," +
        "hashed_password VARCHAR(15)," +
        "email  VARCHAR(30)) engine=innodb;";
```

Relational Schema
registered(user_id: int, name: varchar(20),
hashed_password: varchar(15),
email: varchar(30))
user_id: Foreign key to user
Candidate Keys
email

2.3 author

Creation

```
 "CREATE TABLE author(" +
        "user_id VARCHAR(5) primary key," +
         "name VARCHAR(20) ," +
        "money_amount FLOAT) engine=innodb; ";
```

Relational Schema

author(user_id: int, money_amount: float)

user_id: Foreign key to user

Candidate Keys

None

2.4 book

Creation

```
"CREATE TABLE book(" +
      "book_id CHAR(8) primary key," +
       "title VARCHAR(20)," +
       "rating FLOAT," +
      "publish_date DATE) engine=innodb;";
```

Relational Schema

book(book_id: int, title: varchar(20), rating: float, publish_date: date)

Candidate Keys

None

2.5 genre

Creation

```
"CREATE TABLE genre(" +
      "genre_id CHAR(8) primary key," +
      "genre_name VARCHAR(10) ) engine=innodb";
```

Relational Schema

genre(genre_id: int, genre_name: varchar(10))

Candidate Keys

None

2.6 belongs

Creation

```
"CREATE TABLE belongs(" +
        "book_id CHAR(8) references book(book_id)," +
       "genre_name VARCHAR(10) ) engine=innodb;";
```

Relational Schema

belongs(book_id: int, genre_id: int)
book_id: Foreign key to book
genre_id: Foreign key to genre

Candidate Keys

None

2.7 list

Creation

```
"CREATE TABLE list("  +
    "list_id CHAR(8) primary key," +
     "type VARCHAR(10) ) engine=innodb; ";
```

Relational Schema

list(list_id: int, type: varchar(10))

Candidate Keys

None

2.8 makes

Creation

```
"CREATE TABLE makes("  +
    "user_id  VARCHAR(5) REFERENCES registered(user_id)," +
     "list_id CHAR(8) REFERENCES list(list_id)) engine=innodb; ";
```

Relational Schema

makes(user_id: int, list_id: int)
user_id: Foreign key to registered
list_id: Foreign key to list

Candidate Keys

None

2.9 admin

Creation

```
"CREATE TABLE admin("  +
      "user_id VARCHAR(5) primary key," +
       "admin_id CHAR(8)  ) engine=innodb; ";
```

Relational Schema

admin(user_id: int, admin_id: int)
user_id: Foreign key to user

Candidate Keys

admin_id

2.10 forum

Creation

```
"CREATE TABLE forum(" +
     "forum_id CHAR(8) primary key," +
     "title VARCHAR(20)" +
    "creation_date DATE ) engine=innodb;" ;
```

Relational Schema

forum(forum_id: int, title varchar(20), creation_date: date)

Candidate Keys

title

2.11 system_report

Creation

```
"CREATE TABLE system_report(" +
    "report_id CHAR(8) primary key," +
     "body VARCHAR(30) ) engine=innodb;" ;
```

Relational Schema

report(report_id: int, body: varchar(30))

Candidate Keys

None

2.12 open

Creation

```
"CREATE TABLE open("  +
    "user_id VARCHAR(5) references admin(user_id)," +
    "forum_id CHAR(8) references forum(forum_id) ) engine=innodb;" ;
```

Relational Schema

open(user_id: int, forum_id: int)
        user_id: Foreign key to admin
        forum_id: Foreign key to forum

Candidate Keys

None

2.13 form

Creation

```
"CREATE TABLE form(" +
```

```
  "user_id VARCHAR(5) references admin(user_id)," +
  "report_id CHAR(8) references  system_report(report_id) ) engine=innodb; ";
```

Relational Schema

form(<u>user_id: int, report_id: int</u>)
user_id: Foreign key to admin
report_id: Foreign key to system_report

Candidate Keys

None

2.14 ebook

Creation

```
"CREATE TABLE ebook("  +
    "book_id CHAR(8) primary key, " +
    "pdf_link VARCHAR(100)," +
    "price FLOAT ) engine=innodb;";
```

Relational Schema

ebook(<u>book_id: int</u>, pdf_link: varchar(100), price: float)
book_id: Foreign key to book

Candidate Keys

pdf_link

2.15 follow

Creation

```
"CREATE TABLE follow(" +
    "user_id1 VARCHAR(5) references user(user_id)," +
```

```
    "user_id2 VARCHAR(5) references user(user_id)) engine=innodb;";
```

Relational Schema

follow(user_id1: int, user_id2: int)
    user_id1: Foreign key to user
    user_id2: Foreign key to user

Candidate Keys

None

2.16 review

Creation

```
"CREATE TABLE review("+
    "review_id CHAR(8) primary key," +
    "title VARCHAR(50)," +
    "body VARCHAR(250) ) engine=innodb" ;
```

Relational Schema

review(review_id: int, title: varchar(50), body: varchar(250))

Candidate Keys

None

2.17 likee

Creation

```
"CREATE TABLE likee(" +
     "user_id VARCHAR(5) references user(user_id)," +
    "book_id CHAR(8) references book(book_id) ) engine=innodb; ";
```

Relational Schema

likee(user_id: int, book_id:int)
    user_id: Foreign key to user
    book_id: Foreign key to book

Candidate Keys

None

2.18 rate

Creation

```
"CREATE TABLE rate("+
      "user_id VARCHAR(5) references user(user_id)," +
      "book_id CHAR(8) references book(book_id) ) engine=innodb; ";
```

Relational Schema

rate(user_id: int, book_id: int)
        user_id: Foreign key to user
        book_id: Foreign key to book

Candidate Keys

None

2.19 comment

Creation

```
"CREATE TABLE comment(" +
      "comment_id CHAR(8) primary key," +
      "body VARCHAR(250),"   +
      "date DATE ) engine=innodb;";
```

Relational Schema

comment(comment_id: int, body: varchar(250), date: date)

Candidate Keys

None

2.20 leavee

Creation

```
"CREATE TABLE leavee(" +
        "comment_id CHAR(8) references comment(comment_id)," +
        "user_id VARCHAR(5) references registered(user_id) ) engine=innodb;";
```

Relational Schema

leavee(comment_id: int, user_id: int)
comment_id: Foreign key to comment
user_id: Foreign key to registered

Candidate Keys

None

2.21 contains

Creation

```
"CREATE TABLE containss(" +
        "comment_id CHAR(8) references comment(comment_id)," +
        "forum_id CHAR(8) references forum(forum_id)) engine=innodb;";
```

Relational Schema

containss(comment_id: int, forum_id: int)
comment_id: Foreign key to comment
forum_id: Foreign key to forum

Candidate Keys

None

2.22 has

Creation

```
"CREATE TABLE has(" +
        "review_id CHAR(8) references review(review_id)," +
       "book_id CHAR(8) references book(book_id) ) engine=innodb;";
```

Relational Schema

has(review_id: int, book_id: int)
review_id: Foreign key to review
book_id: Foreign key to book

Candidate Keys

None

2.23 includee

Creation

```
"CREATE table includee(" +
       "user_id VARCHAR(5) references author(user_id)," +
       "book_id CHAR(8) references book(book_id) ) engine=innodb;";
```

Relational Schema

includee(user_id: int, book_id: int)
user_id: Foreign key to author
book_id: Foreign key to book

Candidate Keys

None

2.24 purchase

Creation

```
"CREATE table purchase(" +
        "user_id VARCHAR(5) references registered(user_id)" +
        "book_id CHAR(8) references ebook(book_id)" +
```

```
        "purchase_date DATE ) engine=innodb;";
```

Relational Schema

purchase(<u>user_id: int, book_id: int</u>, purchase_date: date)
        user_id: Foreign key to registered
        book_id: Foreign key to book

Candidate Keys

None

2.25 add

Creation

```
CREATE TABLE add("+
        "list_id CHAR(8) references list(list_id)," +
        "book_id CHAR(8) references book(book_id) ) engine=innodb";
```

Relational Schema

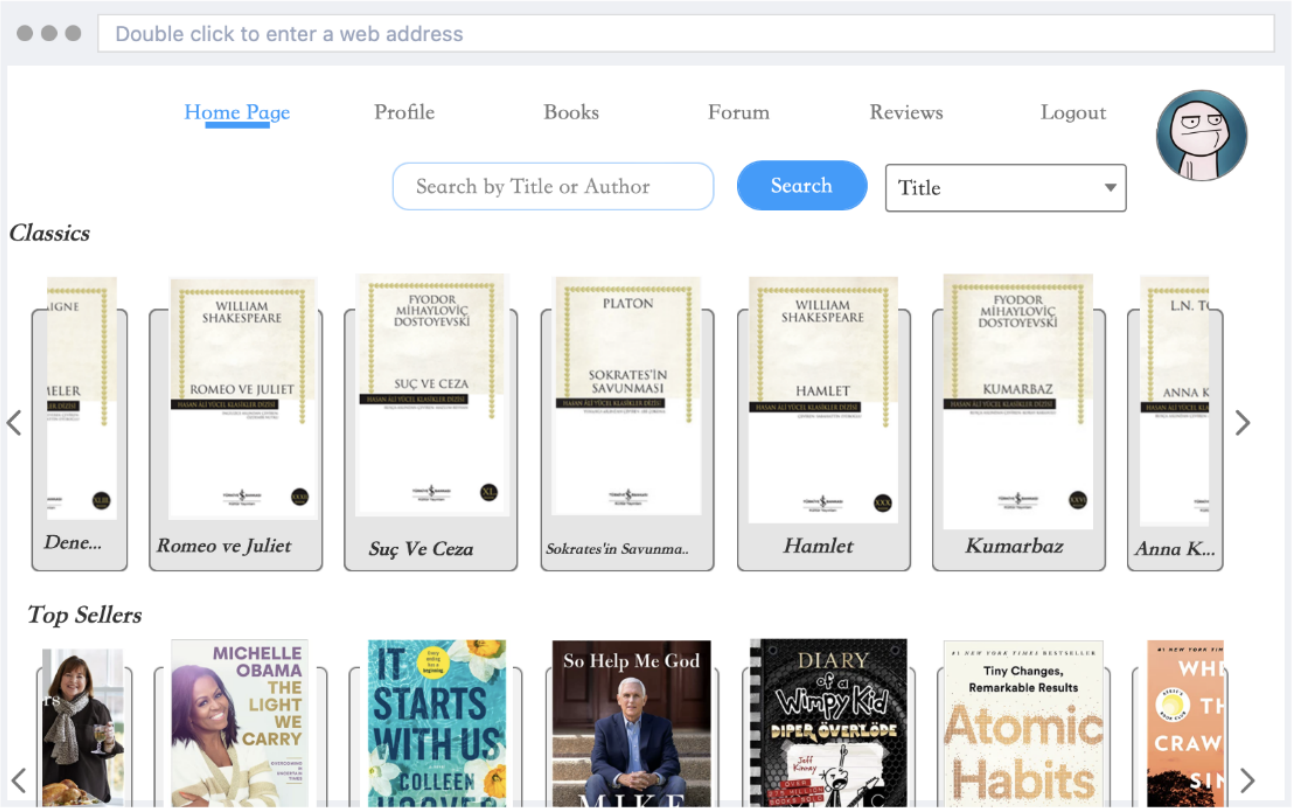
add(<u>list_id: int, book_id: int</u>)
        list_id: Foreign key to list
        book_id: Foreign key to book

Candidate Keys

None

3.User Interface Design & SQL Statements

3.1 Home Page



This is the home page that every type of user encounters with this page. From the top panel they can be navigated to the Books page, Forum page or Login page and Reviews page. Otherwise they can click highlighted books or search for a desired book by title, author or genre.

Search queries:

- Search by genre query

```
"SELECT B.title, G.genre_name " +
      "FROM book B, genre G, belongs BG " +
      "WHERE BG.book_id = B.book_id AND BG.genre_id = G.genre_id AND G.genre_name
LIKE '%" + input + "%';";
```

This returns the books that only have the genre type that the user inputs.

- Search by rating query

```
"SELECT B.title, B.rating " +
      "FROM book B " +
      "WHERE " + input + " < B.rating";
```

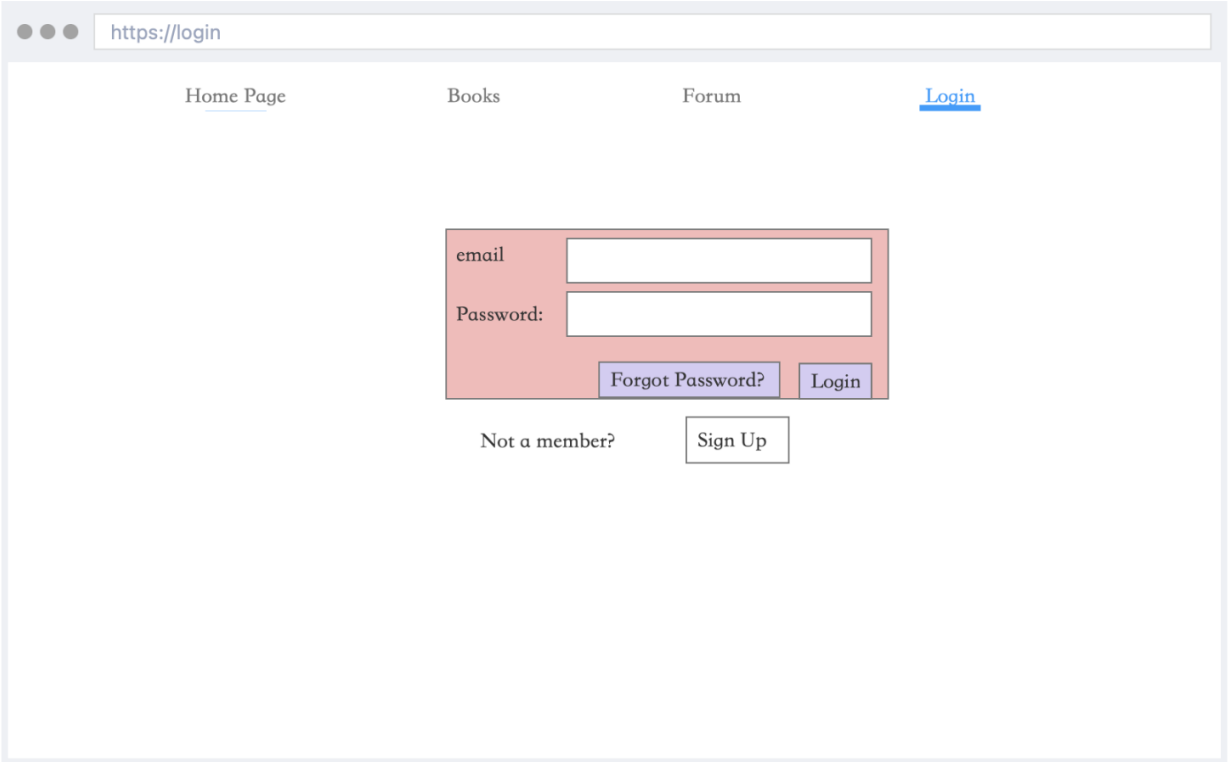
This returns the books that are rated higher than the input of the user.

- Search by author query

```
"SELECT B.title, R.name " +
    "FROM author A, book B, registered R, includee I " +
    "WHERE R.user_id = A.user_id AND I.user_id = A.user_id AND I.book_id = B.book_id
AND R.name LIKE '%" + input + "%'";
```

This returns the books from the author that user inputs.

3.2 Login Page



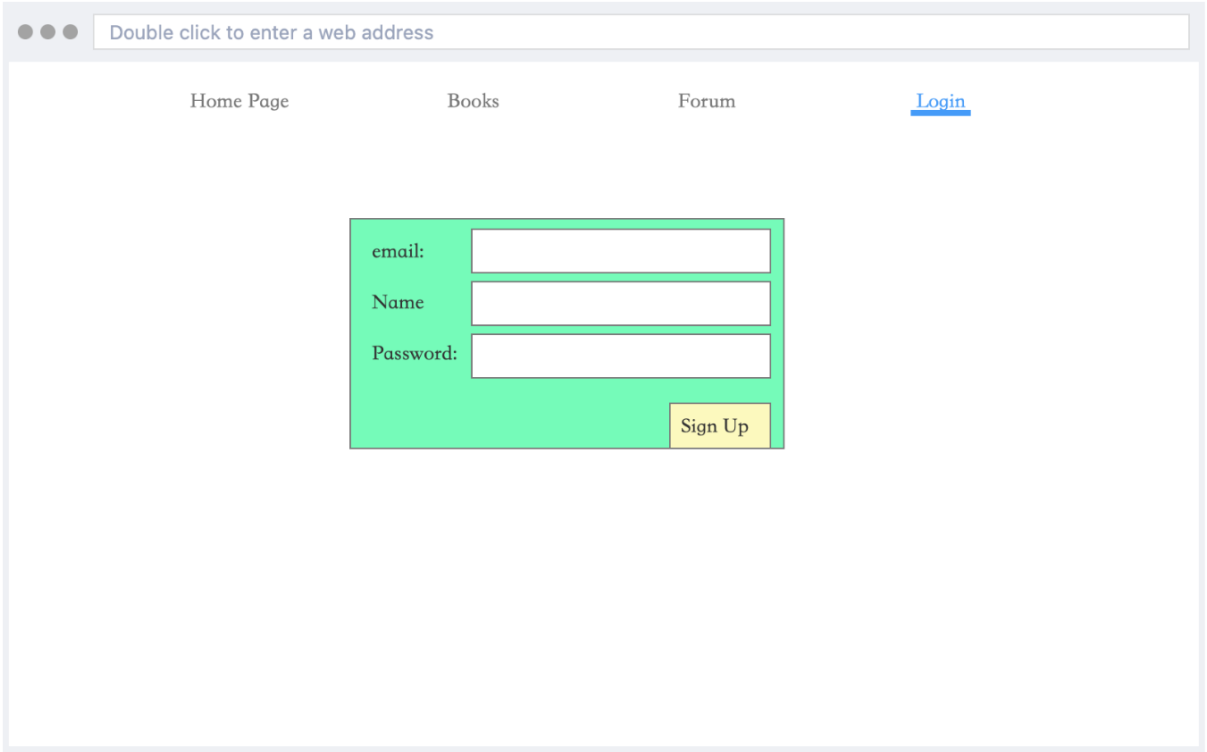
In the login page the user can login his/her account, after entering his credentials and click the login button. If the password of the account is forgotten, they can click Forgot Password? button and refresh the password. If the user is not a member of the site, he can be navigated to the Sign Up page by clicking to the Sign Up button.

- User validation query

```
"SELECT * " +
    "FROM  registered " +
    "WHERE email = @input_email AND hashed_password = @input_hashed_password ;"
```

This checks if the information that is entered matches the database values.

3.3 Sign Up Page



If the user is not a member of the site. He can be signed up to the site by giving an email, name and password.

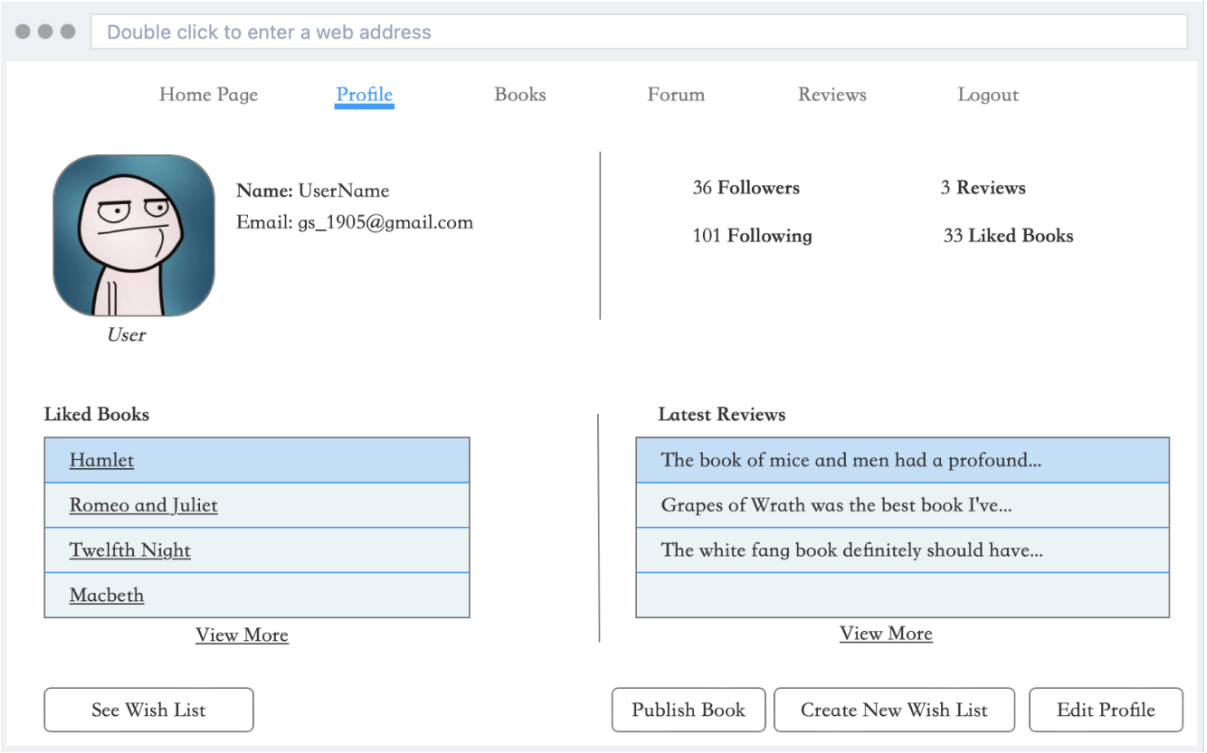
- Insertion into database query

```
"INSERT INTO registered values('@user_id', '@hashed_password', '@email')";
"INSERT INTO user values('@user_id')";
```

This queries both insert into user and registered tables.

3.4 Profile Page

3.4.1 Profile Page for Regular User



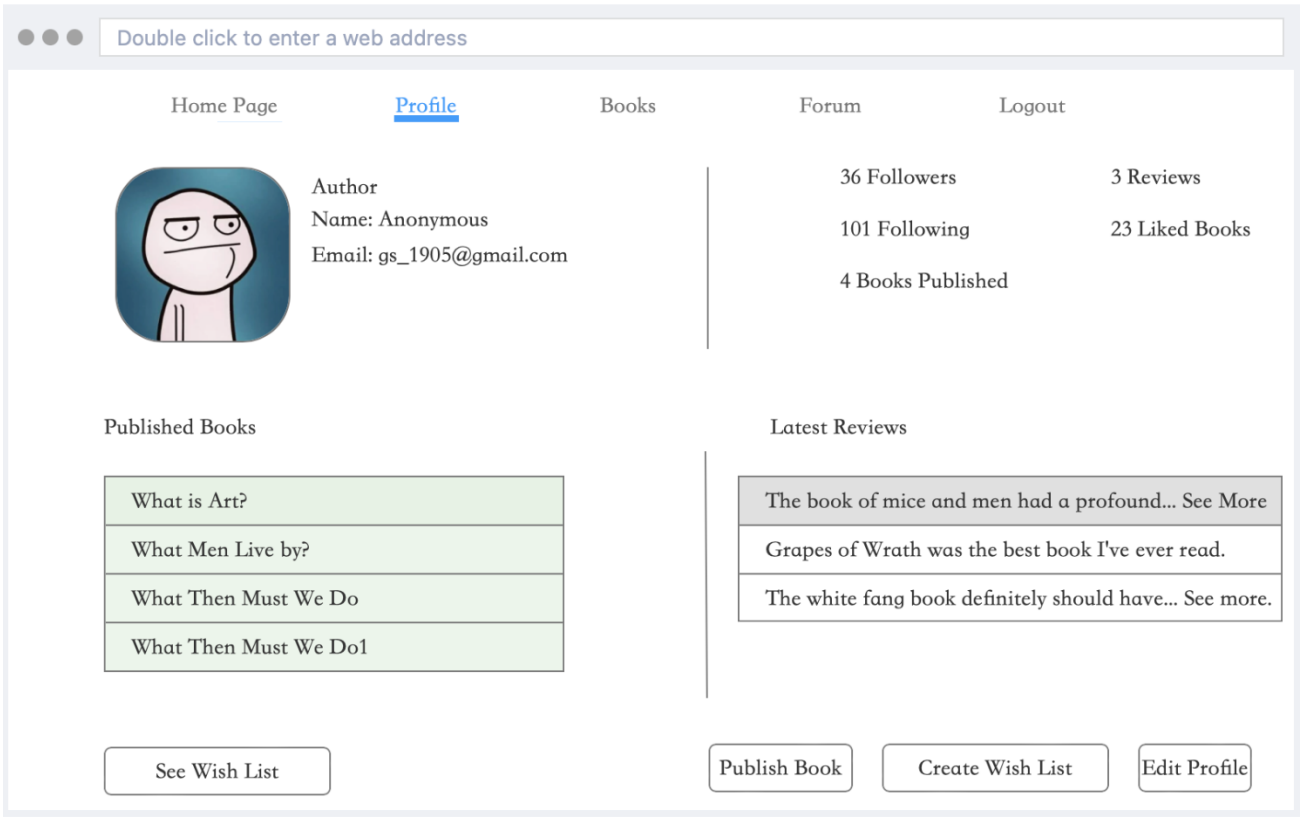
In the profile page the user can see how many people are following him or how many people he is following, how many reviews he has and how many likes he has given to the books. In addition to this, on the left-bottom there are names of the liked books and on the right-bottom there are reviews that he has given to the books. At the bottom the user can click his wish list and see it or edit his profile or create a new wish list or publish a book.

- Display user information query

```
"SELECT R.name, R.email " +
    "FROM registered R " +
    "WHERE R.user_id = @user_id ";
```

This query returns user information.

3.4.2 Profile Page for Author Type Users



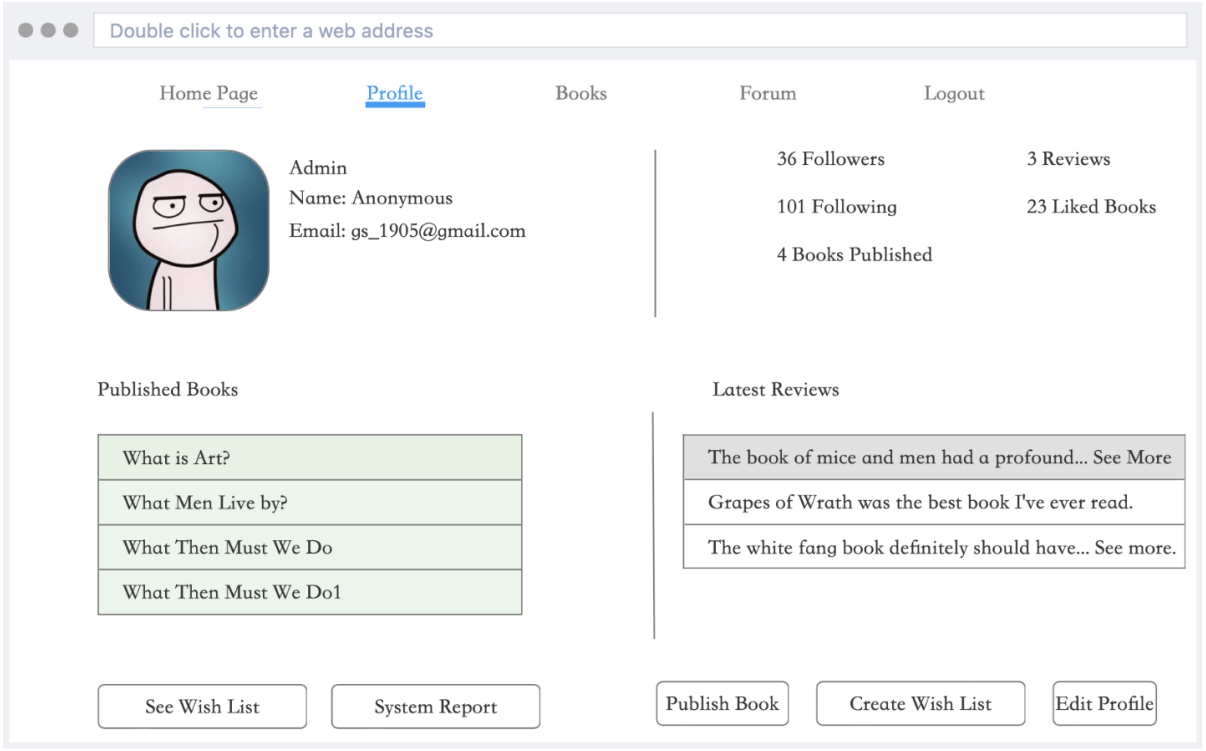
If the user publishes a book the user type changes and at the bottom-left of the page the published books appear.

- Insert book query

```
"INSERT INTO book values('@book_id', '@title', '@rating', '@publish_date')";
```

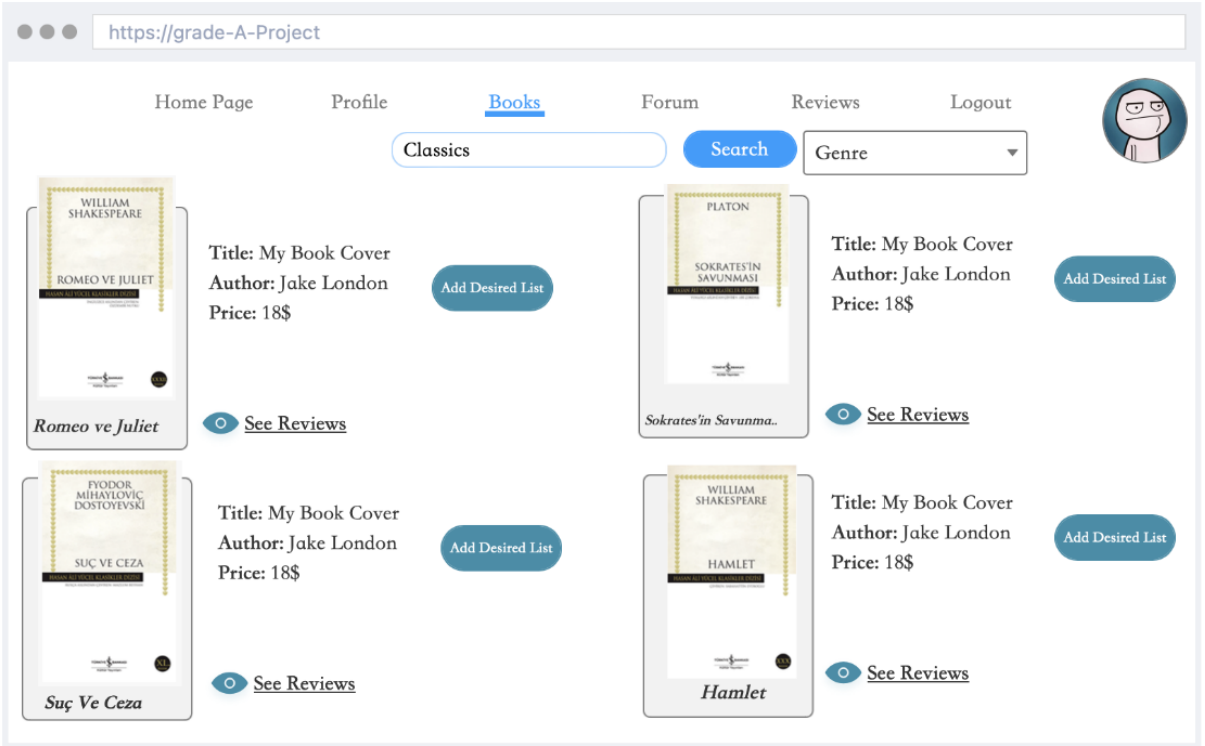
3.4.3 Profile Page for

Admin Type Users



Admin type users also create system report from this page.

3.5 Books Search Page

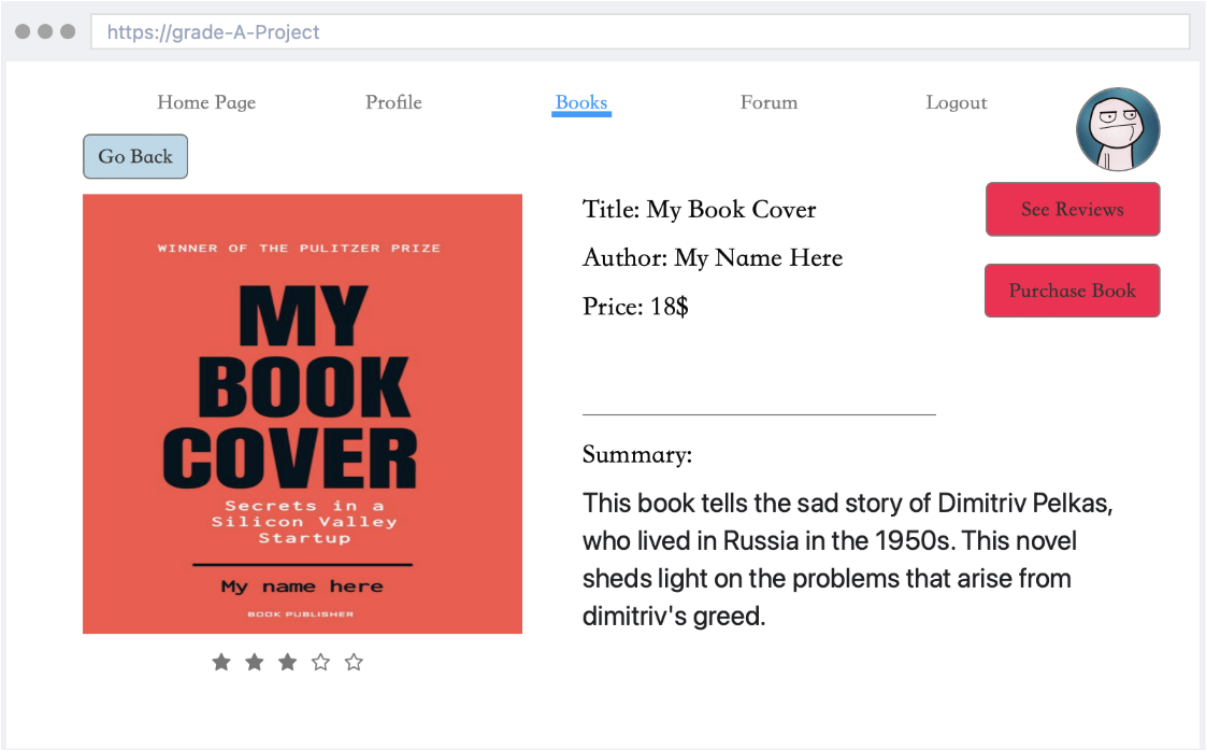


In the Books Page users can search for a desired book by its title, author and genre. In addition to this, near any occurred book there is an Add Desired List and See Reviews buttons that allow users to add the book to the desired list or see its reviews published by other users. Also users can see book's title, author and price.

- Insert into list query

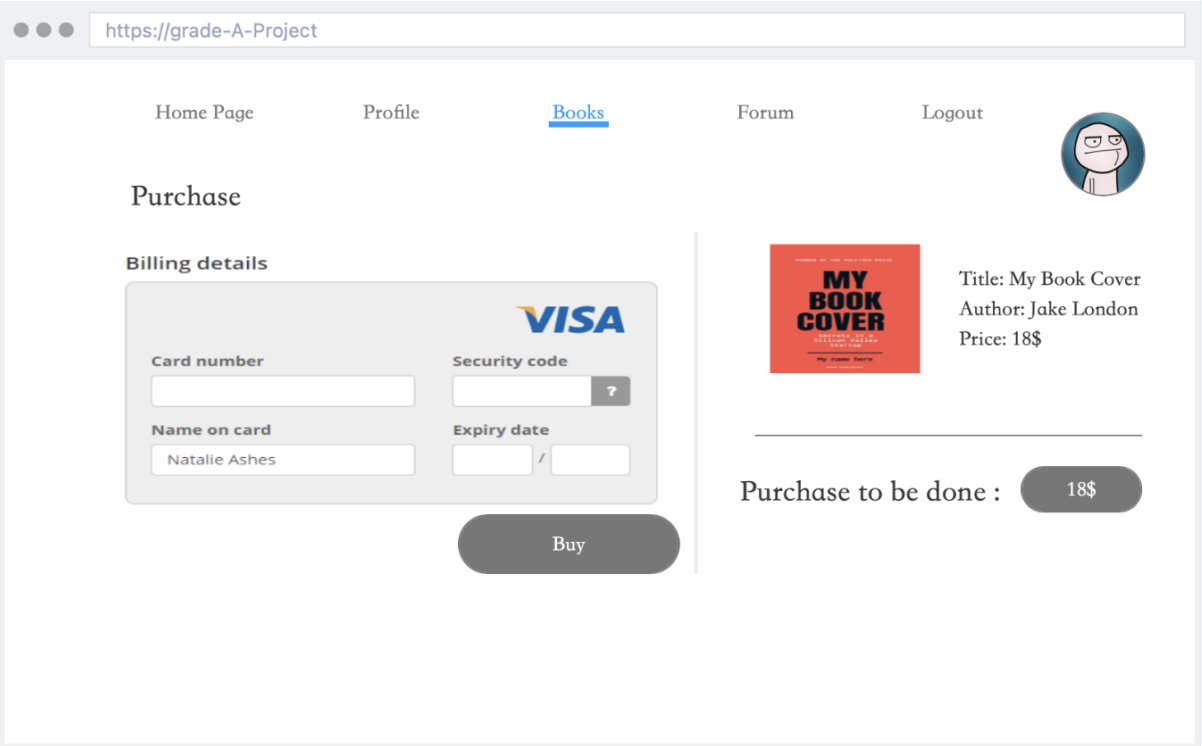
```
"INSERT INTO list values('@list_id', '@type')"
```

3.6 Book Page



If the user clicks any of the books from the Book Search Page or from the Home page, this book page opens, which is specific for every book. In this page users can see a summary of the book and the rating. If they desire they can turn back to the previous page or See Reviews, or Purchase the book or give a rating.

3.7 Purchase Page

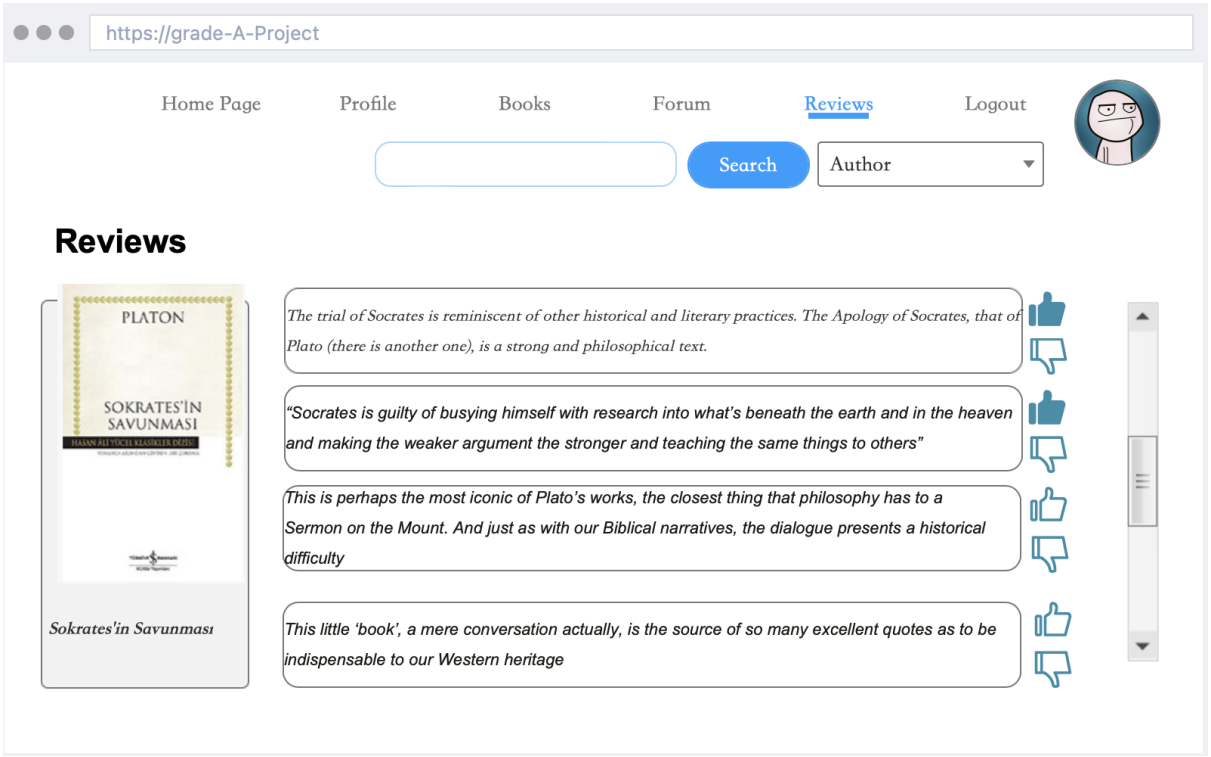


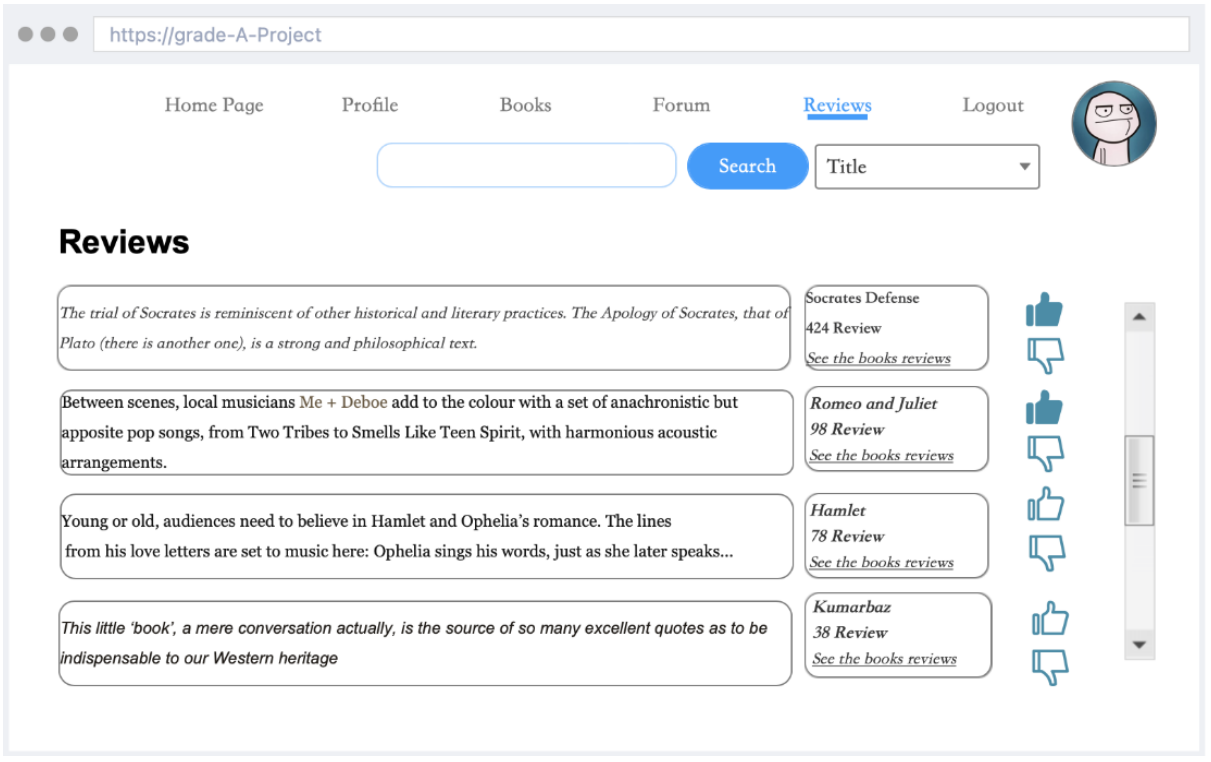
By clicking the purchase book from the Book Page, users are navigated to this page. If they give credit card information, they can buy the book. On the right hand side of the page there is information about a book, which is wanted to be purchased. During the purchase process after the card information is entered, users will be directed to the bank's internet page to confirm further details.

- Insert into purchase query

```
"INSERT INTO purchase values('@user_id', '@book_id', '@purchase_date')"
```

3.9 Reviews Page





In these pages users can search book review's by title, author or genre. And users can give approval or disapproval to the reviews if they want.

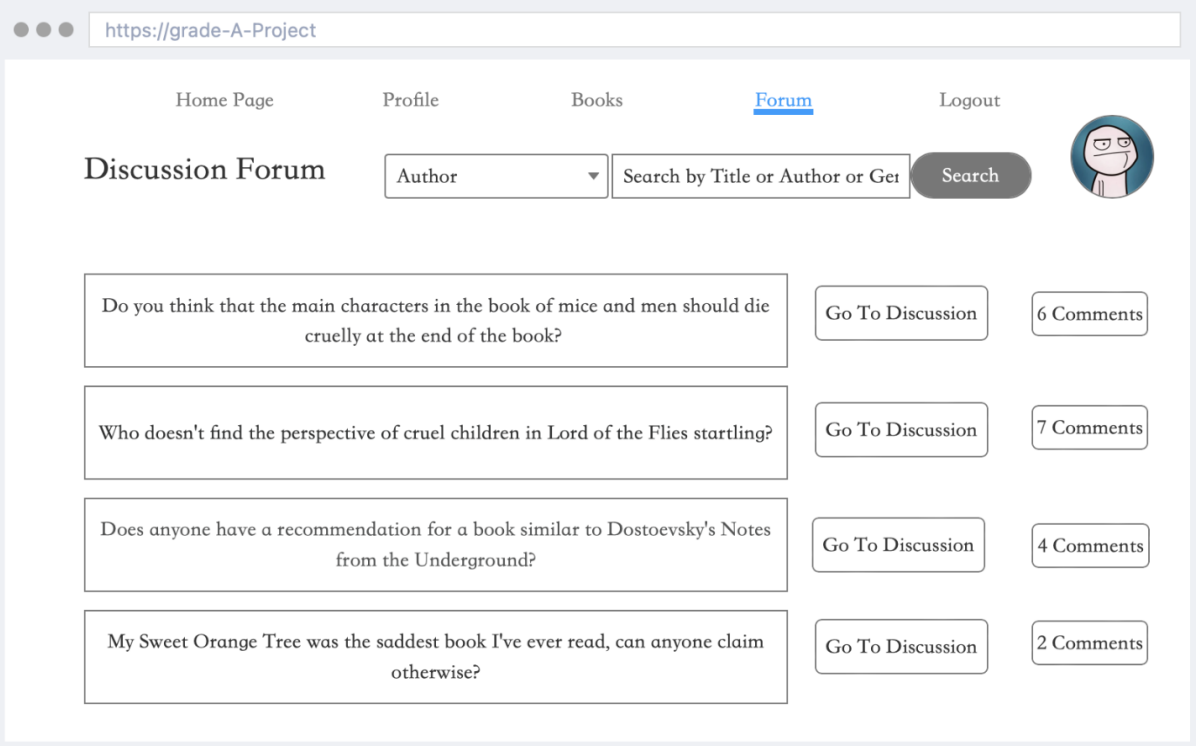
- See book reviews query

```
"SELECT R.title, R.body " +
    "FROM review R, has H, book B " +
```

```
    "WHERE R.review_id = H.review_id AND H.book_id = B.book_id AND B.title = @title";
```

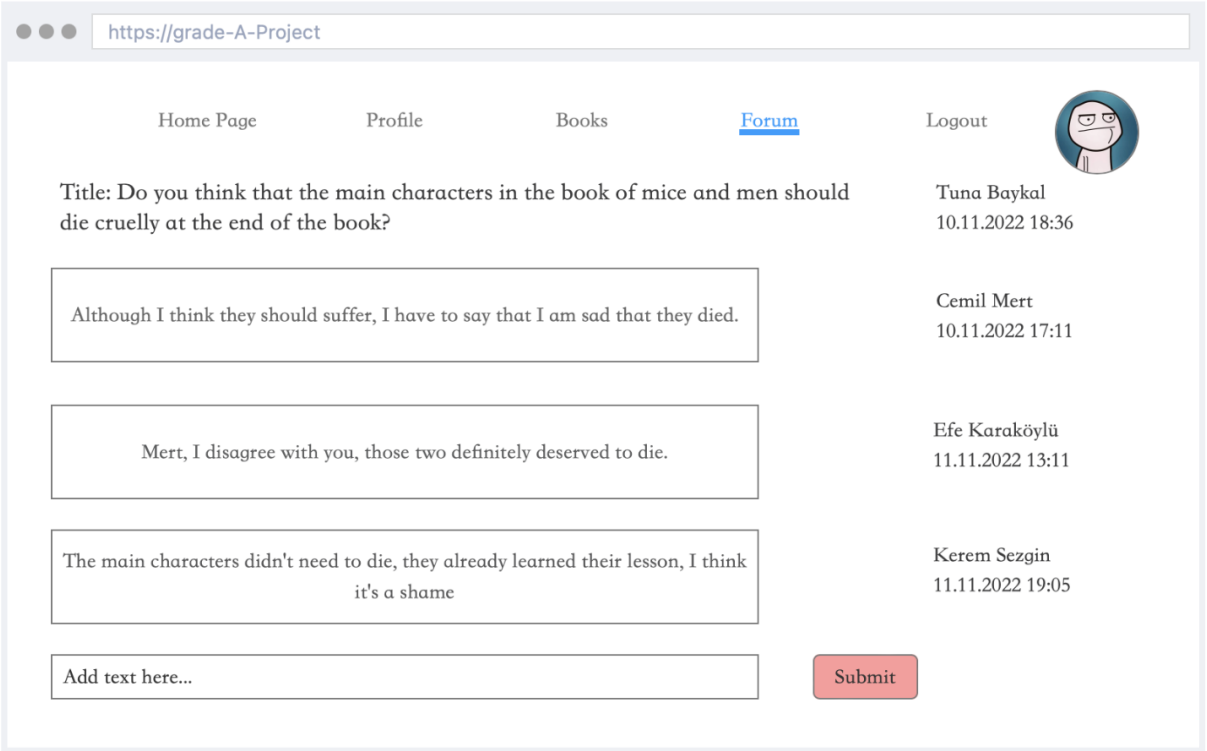
This returns the reviews from the book that is selected.

3.10 Forum Page



In the Forum Page users can see the Title of the discussions and they can go to the inner page of the discussions. Also users can see how many comments have written to the specific discussion.

3.11 Forum Inner Page



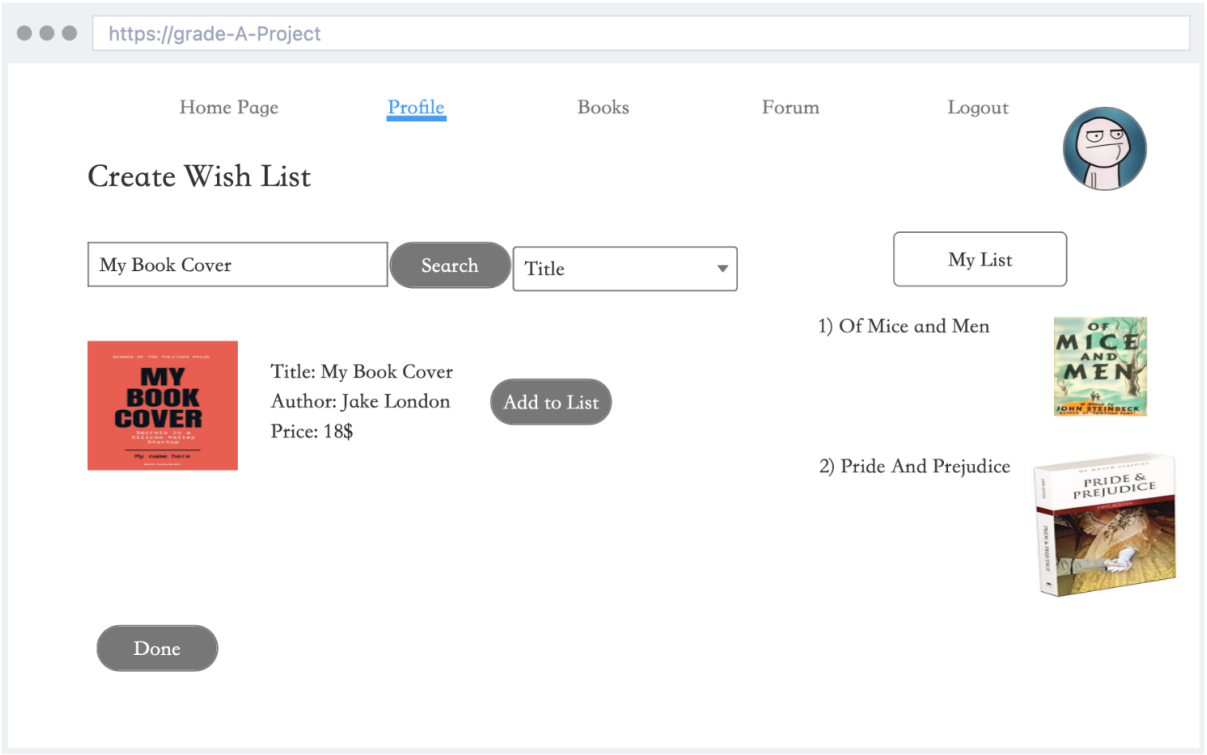
Inner forum page contains the comments that are written into the discussion titles. Every author of the comments can be seen. And in this page users can write and post their ideas about that discussion.

- Display comments in forum query

```
"SELECT CO.body
FROM forum F, containss C, comment CO
WHERE C. = G.forum_id AND C.comment_id = CO.comment_id AND G.forum_title =
@forum_title ORDER BY CO.date DESC ";
```

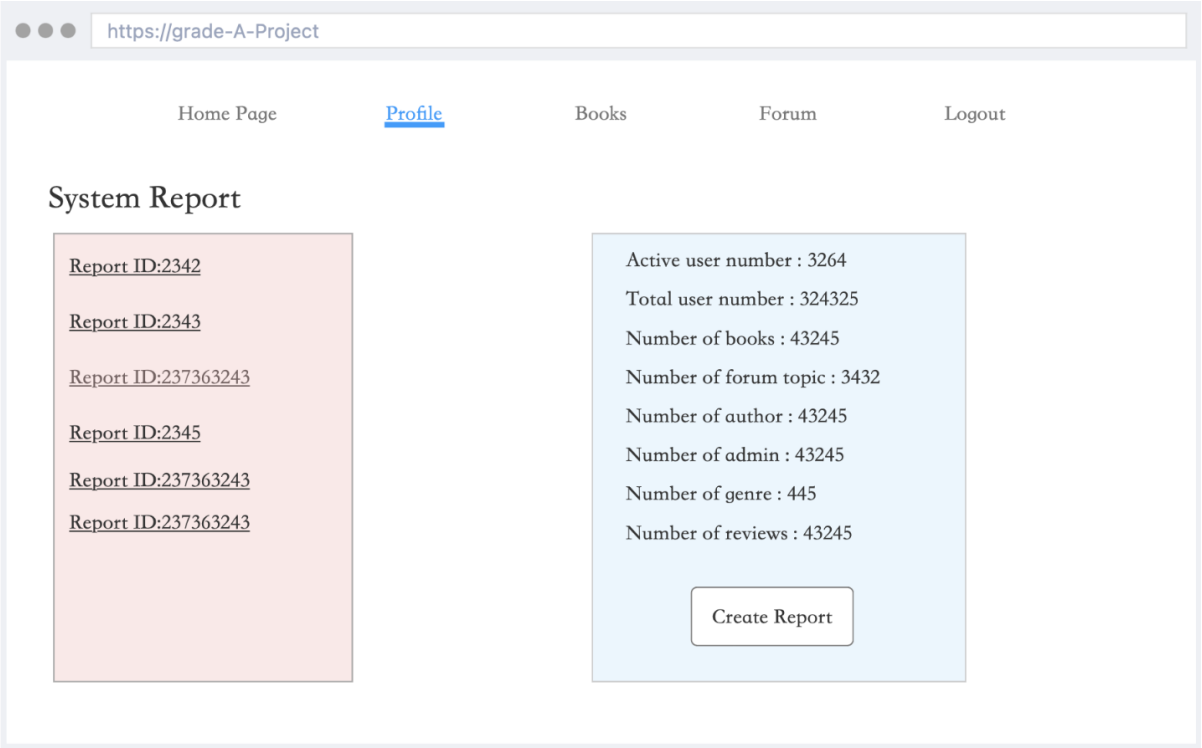
Returns the associated forums comments.

3.12 Wish List Page



In the profile page there is a create wish list button, which directs users to this page. In this page users can search books and add them to the wish list if they want. On the right-hand side there are books that have been added to the wish list.

3.13 System Report Page



Admin's can be directed to this page from their profile page. Also they can create report from this page according to current statistics.

- Select reports query

```
"SELECT * " +
      "FROM system_report"
```

3.14 All of the insert queries

```
"INSERT INTO user values('@user_id')";
```

```
"INSERT INTO admin values('@admin_id', '@user_id')";
```

```
"INSERT INTO book values('@book_id', '@title', '@rating', '@publish_date')";
```

```
"INSERT INTO genre values('@genre_id', '@genre_name')";
```

```
"INSERT INTO belongs values('@book_id', '@genre_id')";
```

```
"INSERT INTO comment values('@comment_id', '@body')";
```

```
"INSERT INTO containss values('@comment_id', '@group_id')";
```

```
"INSERT INTO ebook values('@book_id', '@pdf_link', '@price')";

"INSERT INTO follow values('@user_id', '@user_id')";

“INSERT INTO form values('@user_id', '@report_id')";

"INSERT INTO forum values('@forum_id', '@title')";

"INSERT INTO groupe values('@group_id, @group_title')";

"INSERT INTO purchase values('@purchase_date')";
```

4. Advanced Database Components

4.1. Reports

The following reports will show information about the system.

Total Number of Forums

```
"SELECT count(*) AS forum_cnt FROM FORUM;"
```

Total Number of Books per genre

```
"SELECT G.name, count(*) as book_cnt" +
"FROM Genre G, Book B" +
"GROUP BY G.name";
```

Total number of registered users

```
"SELECT count(*) AS registered_users_count_for_admin"+
"FROM Regitered;"
```

4.2 Views

Authors view for Admins

Admins are going to use this view to see all authors.

```
"CREATE VIEW authors_for_admin AS" +
"SELECT A.user_id, A.money_amount, R.name FROM Author A, Registered R"  +
```

```
"WHERE R.user_id = A.user_id;"
```

Registered users view for Admins

Admins are going to use this view to see all registered users.

```
"CREATE VIEW registered_users_for_admin AS"+
"SELECT R.user_id, R.name, R.email FROM Registered R"
```

Book view for Admins

Admins are going to use this view to see all books and their information.

```
"CREATE VIEW book_for_admin AS" +
"SELECT title, genre FROM BOOK;"
```

4.3 Triggers

These triggers automatically do some operations to enhance the consistency of the system.

Set Authors' Money Amount When a Book is Sold.

```
"CREATE TRIGGER set_author AFTER INSERT ON purchase" +
    "SET money_amount = money_amount + (" +
    "SELECT price FROM BOOK b WHERE NEW.book_id = purchase.book_id );"
```

Set Books' price when the book is rated.

```
"CREATE TRIGGER book AFTER INSERT ON rate" +

  "UPDATE book"
  "SET rating = rating + ( SELECT rating FROM book b WHERE NEW.book_id =
book.book_id);"
```

4.4 Constraints

1. Users must register to the system in order to see or publish books.
2. Passwords are at most 10 and at least 6 characters long.
3. A user can review at most 1 time for each book.
4. An author can publish 1 book in a month.
5. Users can create multiple wish lists without deleting existing ones.
6. In order to purchase books or review/discuss a book either from a forum or review page users must register to the system.
7. Authors cannot review or rate their own books.

Other constraints are specified in the table creation statements, such as primary keys, foreign keys and conditions on what to do when a foreign key is updated or deleted, along with not null values.

4.5 Stored Procedures

The following stored procedures will be used due to the frequent need for them.

Login Check

Returns a table with 1 row if login successful, no rows if else.

```
DELIMITER //
CREATE PROCEDURE login(IN uid int, IN pass varchar(20)) BEGIN
    SELECT *
    FROM user
    WHERE user_id = uid AND hashed_password = pass;
END //
DELIMITER ;
```

Browse Books by Title

Most common search method, with the only filter being the title of the item.

```
DELIMITER //
CREATE PROCEDURE search_by_title(IN search_title varchar(20)) BEGIN
    SELECT *
    FROM book_item
    WHERE title = search_title;
END //
DELIMITER ;
```

5. Implementation Details

As our database management system (DBMS), we will use MariaDB. For the backend implementation of our application, we will use the Java library, along with the PHP, CSS, and HTML for the frontend. The testing of the application will be done locally, and later will be deployed through GitHub facilities.HTML and CSS will be used in the design and styling of the user interface.

6. Website

The project can be accessed from the following link:
https://github.com/E-Kerem/CS353-Database-Project

7. References

[1] A. Silberschatz, H. F. Korth, and S. Sudarshan, Database system concepts. New York, NY: McGraw-Hill, 2020.
[2] Nishadha, “Entity relationship diagram (ERD): Er diagram tutorial”,19-Oct-2022. Available: https://creately.com/blog/diagrams/er-diagrams-tutorial/.