

CS4610 Project Report

Recommended Reads

Website for Readers and Book Recommendations

1 December 2021

Justin Nichols

Table of Contents

Introduction	3
Project Overview	3
Related Works	3
System Architecture Design	4
ER Diagram	5
Relational Tables	6
Relational Table: user_login.....	6
Relational Table: user_info	6
Relational Table: book_info	7
Relational Table: ratings_reviews	7
SQL Statements	8
Detailed Component Descriptions.....	8
1. Sign Up	10
2. Login.....	11
3. Logout.....	11
4. Profile	11
4.1 Edit Profile Information	12
4.2 Edit Ratings and Reviews.....	13
5. Browse	13
5.1 Rate and Review Book	15
5.2 Write Review	16
5.3 Add Book.....	16
Conclusion	17

Introduction

Project Overview

Recommended Reads is a user-driven website that allows people to recommend books to other readers. A user can browse through books by genre and see the rating as well as other users' reviews of the book. They will also be able to rate and leave a review of a book they have already read to let other people know their opinion. Subsequently, a user can add a book that currently does not exist and allow others to rate and review it. This website aims to create an easy-to-use space for book lovers to exchange their appraisals or criticisms of books they read.

The implementation of the project was designed as a three-tier web application completely separating the client-side presentation, server-side logic, and database system. The presentation layer is composed of static content delivered to the user as a front-end interface interpreted by a web browser to facilitate a dynamic user experience. As the user interacts with different components of the website, the browser sends requests and receives responses from the server. The server, which is indirectly connected to the database, manipulates the database system and obtains the needed information to send back to the browser for the user to see. The database system provides persistent, structured storage as well as fast retrieval for all the data associated with the website and its users.

Related Works

The core functionality of Recommended Reads draws inspiration from Goodreads, a social website where users can become a part of a community of readers and browse for new books to read. However, Recommended Reads strips this idea down and focuses mainly on providing user-built ratings and reviews of books. Recommended Reads provides a platform to get honest ratings and reviews for a book to make it easier to find the next best book or author.

System Architecture Design

The system architecture follows the general three-tier web application design. The web application is deployed across two Amazon EC2 instances, with one containing the presentation layer and the other containing the logic and data layers. The first instance controls the static content using Apache Webserver and the other manages the Java EE servlets using Apache Tomcat and the database using MySQL. The direct link to the website can be found using <http://18.216.124.81/RecommendedReads/>.

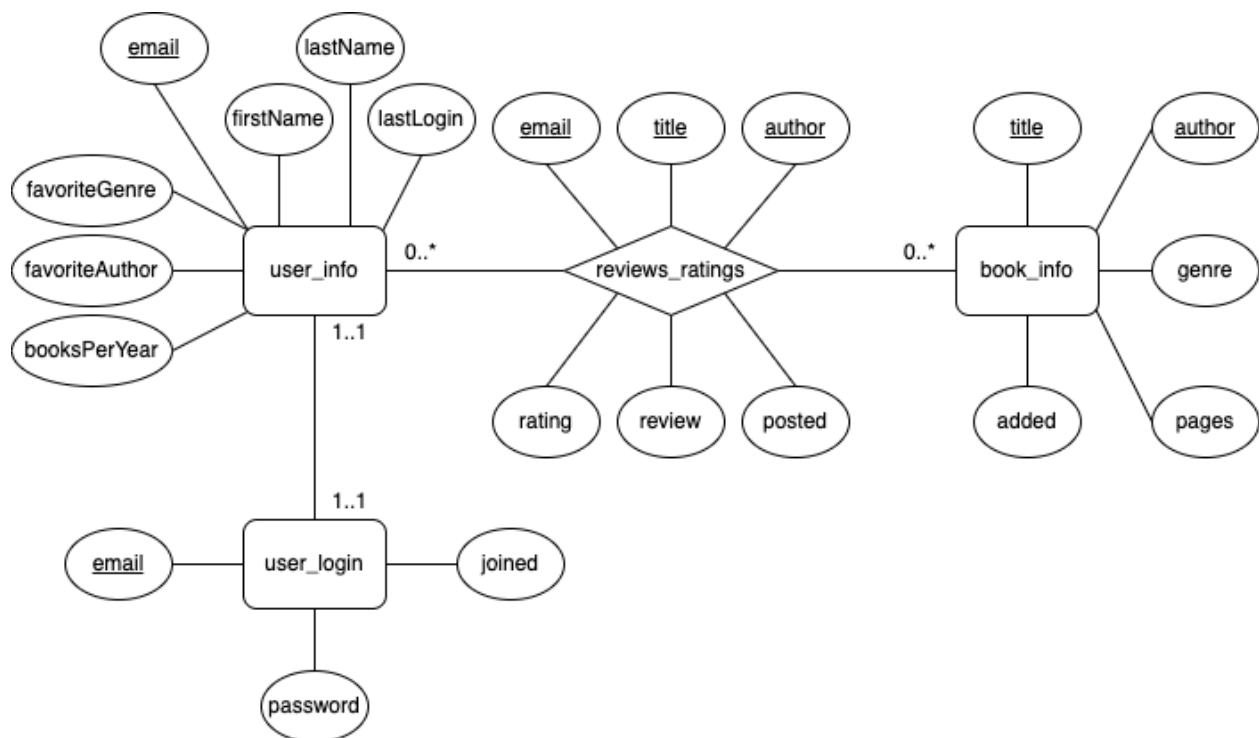
Regarding the static content, HTML, CSS, and JavaScript were used as the foundation of the webpage. HTML was used to create the actual tangible content on the webpage. CSS was used to design and present the HTML elements in a user-focused fashion. JavaScript was used to add dynamic functionality to changing HTML elements. To create a modular, consistent, and responsive webpage, the Bootstrap framework was utilized through the website. For example, Bootstrap validation was useful in verifying form input. Likewise, for ease of use, jQuery was implemented when necessary to reduce the amount of JavaScript code. In order to handle errors and to freely access server attributes on the front-end, JavaServer Pages (JSP) was used to provide a link between the server and the web content.

Regarding the dynamic content, Java EE Servlets was created to handle form inputs and user requests made on the website. The servlets act as a middleman between the static content and the database. They input and retrieve data to and from the database and display it to the user. The servlets interact with the database, send and receive data, redirect to other webpages, handle validation and errors, and implement backend logic. Subsequently, the servlets used HttpSession in order to manage each user session and uphold the security of the website. When a user logged in, a unique session was created for that user. Contrarily, when a user logged out, the unique

session was destroyed. The Apache Tomcat server was connected with the Apache Webserver with the mod_jk configuration.

Regarding the database, there are four tables: user_login, user_info, book_info, and reviews_ratings. These tables were implemented using MySQL. Below is the ER Diagram for the database structure. Overall, an email address uniquely identifies a user since a user can only make one account with the website. Likewise, the title and author of a book uniquely identify a book since a particular book can exist only once on the website. Preventing multiple instances of the same book discourages fragmented ratings and reviews. Overall, MySQL provided a relational database with persistent storage and ease of integration with the Java servlets. Below is the entity-relationship model for the database.

ER Diagram



Relational Tables

The ER diagram was then transformed into relational tables for use in the MySQL database. This allowed for the servers to insert new data, query and update current data, and delete no longer needed data. Below brief descriptions of each relational table and its use within the developed application are provided along with a table displaying the contents of the relational table.

Relational Table: user_login

The user_login table stores the login credentials needed to associate a user to an account. It is also used to validate a user. The table holds the email address and the associated password for each user as well as the date and time the user last logged into their account. Since there can only be one account per user, the email address is the primary key because it can uniquely identify a user.

<u>email</u>	password	lastLogin
varchar (32)	varchar (32)	datetime

Relational Table: user_info

The user_info table stores the personal identifying information about each user. This information is displayed on the profile page of the user and gives them the option to update the necessary fields of their choice. The table holds the user's first name, last name, email, favorite genre, favorite author, number of books read per year, and the date and time the user joined the website. Since there can only be one account per user, the email address is the primary key because it can uniquely identify a user.

firstName	lastName	<u>email</u>	favoriteGenre	favoriteAuthor	booksPerYear	joined
-----------	----------	--------------	---------------	----------------	--------------	--------

varchar(32)	varchar (32)	varchar (32)	varchar (32)	varchar (32)	int	datetime
-------------	--------------	--------------	--------------	--------------	-----	----------

Relational Table: book_info

The book_info table stores the identifying information for each book listed on the website. This book information, grouped by genre, will be displayed to the user to browse and find new books they want to read based on the ratings and reviews. The table holds the book's title, author, genre, page count, and year of publication. Since a book can only be listed once to prevent fragmented ratings and reviews, the book title and author is the primary key. This is because multiple books can have the same title or the same author but not both the same title and the same author.

<u>title</u>	<u>author</u>	genre	pages	year
varchar (64)	varchar (32)	varchar (32)	int	int

Relational Table: ratings_reviews

The ratings_reviews table stores the ratings and reviews generated by users for each book. These entries will be associated with a user account and a specific book. They will primarily be shown for each book listing, but users will have the ability to see books they have rated and reviewed on their profile. Not only will users have the ability to make ratings and reviews, but they will be able to delete them if they wish. The table holds the email address of the user who wrote the review, the title and author of the book, the rating and review given by the user to the book, and the date and time it was posted. Since a user can rate and review any number of books and a book can be reviewed and rated by any number of users, the primary key is the email address of the user and the title and author of the book. This is because a user can only rate and review a book once.

email	title	author	rating	review	posted
varchar (32)	varchar (32)	datetime	float	varchar (256)	datetime

SQL Statements

```
CREATE DATABASE recommended_reads;
```

```
CREATE TABLE user_login (email VARCHAR(32), password
VARCHAR(32), lastLogin DATETIME, PRIMARY KEY (email));
```

```
CREATE TABLE user_info (firstName VARCHAR(32), lastName
VARCHAR(32), email VARCHAR(32), favoriteGenre VARCHAR(32),
favoriteAuthor VARCHAR(32), booksPerYear INT, joined DATETIME,
PRIMARY KEY (email));
```

```
CREATE TABLE book_info (title VARCHAR(64), author VARCHAR(32),
genre VARCHAR(32), pages INT, year INT, PRIMARY KEY (title,
author));
```

```
CREATE TABLE ratings_reviews (email VARCHAR(32), title
VARCHAR(64), author VARCHAR(32), rating FLOAT, review
VARCHAR(256), posted DATETIME, PRIMARY KEY (email, title,
author));
```

Detailed Component Descriptions

Below is a detailed description of each functional component of the Recommended Reads web application. Each component is a Java Servlet within the web application, and components such as Profile and Browse contain subcomponents that are also implemented as servlets but are mainly tied to their respective parent components. All servlets forward POST requests to the

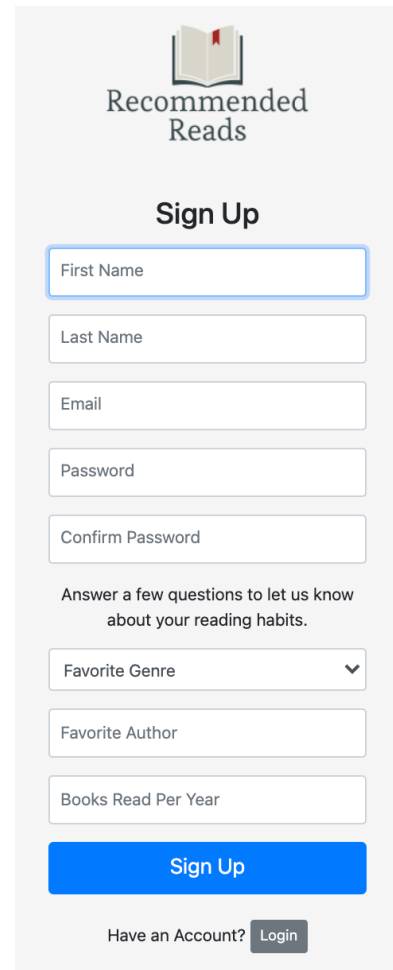
doGet method. All servlets also use the method sendRedirect to allow for easy, safe, and continuous movement around the website. Some servlets utilize RequestDispatcher to forward a request to the same webpage instead of redirecting as usual for minimal updates to that specific webpage. On top of these developmental choices, HttpSession was used to manage unique user sessions and update the website using the personal information of the user. Not only does it allow to control user sessions, but it provides security and prevents access to private pages without a generated session. Other than the homepage and the user log in and signup pages of the website, a user is not able to use a URL to obtain access into private pages unless they login and a session is created for their account. Additionally, when a user logs out, their current session is terminated. Throughout the entire website, the HttpSession keeps track of the session state of the user and output any error messages the user may involuntarily generate. Pertaining to the data layer, a JDBC connection is used as a link for all communications between the servlets and the database. For each servlet component listed, an image is provided for visual purposes.



1. Sign Up

From the homepage of the website, a user can click the “Sign Up” button to create an account.

Upon this request, the website will redirect the user to the sign-up page. The SignUp servlet handles all actions within this component. The user is asked to provide their first name, last name, email address, a valid password. The valid password must be 8 characters in length and should only contain alphanumeric characters. The user must also confirm the chosen password for security reasons. On top of this, the user is asked to provide some personal information regarding their reading habits. This includes a dropdown menu to select their favorite genre, a field to type in their favorite author, and a number wheel to select the number of books they read annually. All fields are required in order for the user to successfully create an account and errors will be generated if one or more fields are incorrect. The server also checks the email provided is associated with an account already. If so, the webpage will ask them to provide a different email address. After all fields are validated, both the user_login and user_info databases are updated with the new user account information. Finally, a login session is created, and the user is then redirected inside the website to the book browse page.



The image shows a 'Sign Up' form for a website called 'Recommended Reads'. At the top is the logo, which consists of an open book icon and the text 'Recommended Reads'. Below the logo is the title 'Sign Up'. The form contains several input fields: 'First Name', 'Last Name', 'Email', 'Password', and 'Confirm Password'. Below these is a section titled 'Answer a few questions to let us know about your reading habits.' which includes a dropdown menu for 'Favorite Genre', a text field for 'Favorite Author', and a text field for 'Books Read Per Year'. At the bottom of the form is a blue 'Sign Up' button. Below the button is a link 'Have an Account?' followed by a 'Login' button.

Recommended Reads

Sign Up

First Name

Last Name

Email

Password

Confirm Password

Answer a few questions to let us know about your reading habits.

Favorite Genre

Favorite Author

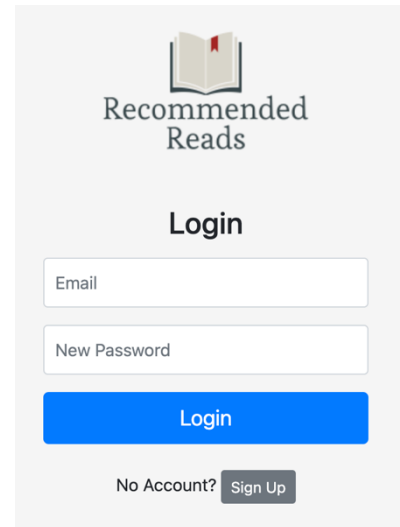
Books Read Per Year

Sign Up

Have an Account? Login

2. Login

From the homepage of the website, a user can click the “Login” button to log in to their account. Upon this request, the website will redirect the user to the login page. The Login servlet handles all actions within this component. The user is asked to provide their email address and password. If either the email address provided does not exist or the password does not match the email, an error will be shown to the user. From this, the user can click the “Sign Up” button, if they do not have an account, which will redirect them to the sign-up page. If the user provides valid login credentials, then a login session will be created, and they will be redirected to the inside the website to the browse page. Finally, the user_login database is updated with a last login time value.

A login form for a website called "Recommended Reads". At the top is a logo of an open book with the text "Recommended Reads" below it. The title "Login" is centered above the input fields. There are two text input fields: "Email" and "New Password". Below these is a blue "Login" button. At the bottom, there is a link "No Account?" followed by a "Sign Up" button.

3. Logout


If a user is logged in, they can log out of their account using the navigation bar. Their current login session will be terminated, and they will be redirected back to the homepage. The Logout servlet handles this implementation.



4. Profile

If the user is logged in, they can use the navigation bar to view the profile page. Once redirected to the profile page, the user can view their personal profile information as well as past ratings and reviews they have made on books. These functions are handled by the Profile servlet. The server connects to the user_info, user_login, and ratings_reviews databases and retrieves the

user-specific information to be displayed on the profile page. The user is given the ability to edit any personal information they choose to update as well as delete any unwanted ratings and reviews they have made. These two separate, but similar actions, are divided into two additional subcomponent servlets.



Justin Nichols

Account **Ratings & Reviews**

Email
nicholsrjustin@gmail.com

Password
Basketball23

Favorite Genre
Memoir

Favorite Author
Tim O'Brien


Books Read Per Year
7

Joined
2021-11-14 19:40:42.0

[Edit Profile](#)

4.1 Edit Profile Information

Under the Account tab, by clicking the “Edit Profile” button, a user can edit their personal profile information. They can update fields such as their name, email, password, and reading preference answers from the questions they were asked during the sign-up process. Similar to the sign-up component, the server validates all changed fields, and updates



Account

First Name
Justin

Last Name
Nichols

Email
nicholsrjustin@gmail.com

Password
.....

Confirm Password
.....

Favorite Genre
Memoir

Favorite Author
Tim O'Brien

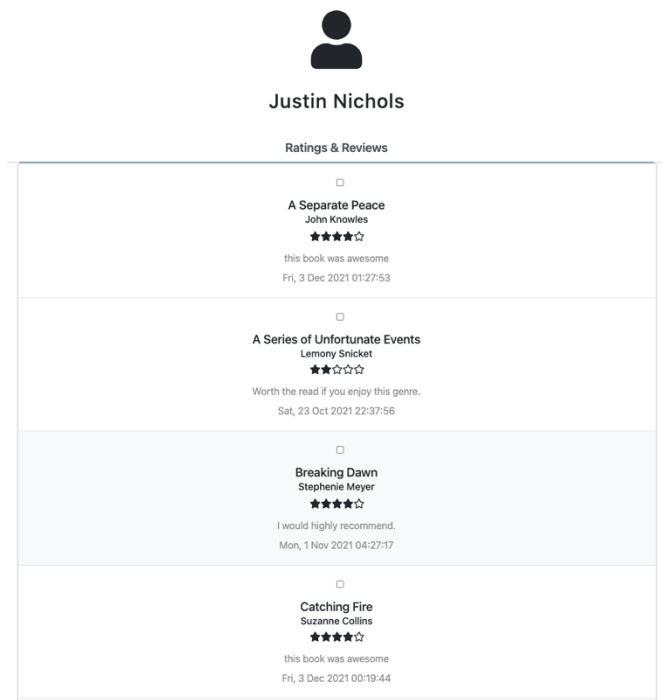
Books Read Per Year
7

[Save](#) [Cancel](#)

user_info and user_login databases accordingly if the user clicks the “Save” button. The only special case is if the user alters their email address. This will result in the server checking if the new email address already exists or not. If it does not exist, then no errors will occur, and the new information will be updated successfully. The user is then redirected back to the base profile page either after saving their edits or if they click the “Cancel” button. These functions are handled by the ProfileEdit1 servlet.

4.2 Edit Ratings and Reviews

Under the Ratings & Reviews tab, by clicking the “Edit” button, a user can delete previously made ratings and reviews they made on books. The user can select as many ratings and reviews to delete. Upon deletion, the ratings_reviews database is updated with any changes. The user is then redirected back to the base profile page either after verifying their deletions or if they click the “Cancel” button. These functions are handled by the ProfileEdit2 servlet.




5. Browse

The user is automatically redirected to the browse page if they sign-up or log in to the website. The user can also view the browse page using the navigation bar. The browse page is the most content-dense page on the entire website. It performs three main tasks, generates the book

listings by genre, shows the ratings and reviews for each book, and gives the user the ability to add a book to the listings. The showing of books listed by genre is handled by the Browse servlet. The individual ratings and reviews made by each user for the book are handled by the RatingsReviews servlet and the ability to add a book to the listings is handled by the AddBook servlet. For showing each book, the servlet connects to the book_info database and collects all the books stored. If the user wants all genres shown, then the servlet will output all books present in the database. If the user wants a specific genre shown, then the servlet will output only the books that are in that genre. Each book listing generated contains a picture of the book cover, a picture of the author, the title, the name of the author, the average rating among all users, the number of total ratings given to the book, the genre, the page count, and the year of publication. To compute the average rating among all users, the servlet connects to the ratings_reviews database, sums all ratings found for that specific book, and divides it by the total ratings found for the book. Additionally, a “See Reviews” and “Write Review” button is generated for each listing to allow the user to see all reviews for the book or write their own review. Subsequently, if the user clicks on the “Write Reviews” button and they have already rated and reviewed that book, an error message is displayed back to them. Ultimately, all other pages that request the Browse Servlet, go indirectly through the RatingsReviews servlet to make certain that all updates

to the books and their ratings and reviews are shown accordingly.

All Genres



A Separate Peace


John Knowles

★★★★☆

3.42 12 ratings

See Reviews


Write Review



Genre: Coming-of-age

Pages: 236

Year of Publication: 1959



A Series of Unfortunate Events


Lemony Snicket

★★★★☆

3.0 12 ratings

See Reviews

Write Review




Genre: Children's

Pages: 849

Year of Publication: 2011

5.1 Rate and Review Book

If the user clicks the “See Reviews” button on the browse page, the RatingsReviews servlet is provoked. The servlet generates each individual user rating and review for the book specified. The servlet connects to the ratings_review database and accumulates all the reviews each book listed. The servlet also connects to the user_info database to obtain the name of the user based on their email. For each rating and review, the name of the user, the star rating, the review, and the date it was posted is shown.



Justin Nichols

★★★★☆

'Harry Potter and the Sorcerer's Stone' is a red-blooded adventure movie, dripping with atmosphere, filled with the gruesome and the sublime, and surprisingly faithful to the novel.

Mon, 8 Nov 2021 18:53:26

5.2 Write Review

If the user clicks the “Write Review” button on the browse page, the WriteReview servlet is provoked. The servlet generates a form input for the user to specify their rating and review. The rating is calculated by allowing the user to dynamically select the number of stars they think the book earned. The review is generated by the user typing in a max of 256 characters description. After posting their rating and review, the servlet connects to the ratings_reviews database and inserts it.

Write Review



5.3 Add Book

If the user clicks the “Add Book” button on the browse page, the AddBook servlet is provoked. The servlet generates a form input for the user to specify the information of the new book, such as the title, author, genre, number of pages, and year of publication. Upon the user clicking the “Add” button, the servlet connects to the book_info database and checks it already exists. If it exists, the book will not be added, and an error message will be displayed to the user. On the contrary, if the book does not exist, the book will be added to the book_info database and a success message will be displayed.

Add Book

Add Book

Title

Author

Favorite Genre

Number of Pages

Year of Publication

Add

Conclusion

The final implementation of the Recommended Reads website delivers a fully functional three-tier web application hosted at <http://18.216.124.81/RecommendedReads/>. The website achieved the goal of providing an easy-to-use place for reading enthusiasts to rate and review books as well as see other book recommendations and opinions from other users. The development process took place over the course of two months and all goals outlined in the project proposal were met. The application has been extensively tested and has no observable errors or bugs under normal usage and runs responsively and efficiently with a reliable Internet connection. The website is able to translate relatively well to mobile devices since the Bootstrap framework was used. The main area for future improvement to better the user experience is an improvement in security. All lines of communication between components within the application should be encrypted and sensitive user information should be securely stored. Ultimately, the application is free to use by any general member of the public that makes an account and shows an interest in involving themselves with a community of book enthusiasts.