Library Management System
CS 157A - Team #22

Michael Leonard
Nicholas Papano
Sarah Saber

Project Overview

Description

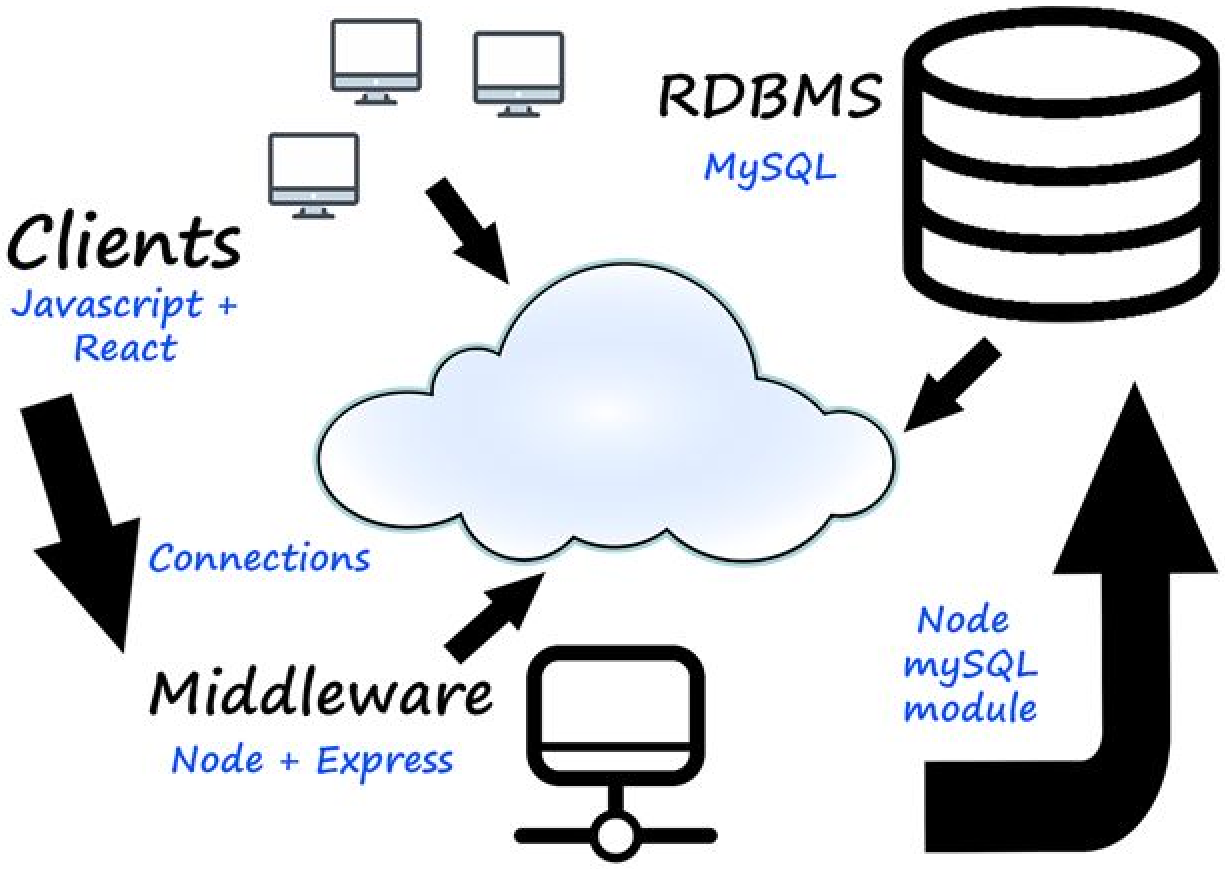
The three-tier database application we will be building is a library management system. The system will perform all fundamental features that a library will need in daily operation. These actions include operations such as issuing library cards, searching for books, checking books in and out, and placing holds. It will also include extra privileges for library staff, such as managing accounts, adding and removing books, and paying fines. The organization that oversees the library will be able to request general analytics and telemetry from the librarians. We as the developers will be designing and implementing the project, making sure all development goals are met. The system will be a web-accessible login-based application and maintain a user interface that is clean, clear, and informative so that all of the various users can navigate the application with ease.

Stakeholders

- Librarians
- Library Staff
- Library Users
- City Government
- Developers

System Environment

Structure of the system



Software used

● Node.js (and Node mySQL module), Express.js, React.js

RDBMS

● MySQL version 8.0.17

Application Languages

●HTML, CSS, JavaScript, Node MySQL module

Hardware used:

2018 15” Macbook Pro

OS: MacOS Catalina version 10.15.1

Processor: 6-Core Intel Core i7 @ 2.2 GHz

Memory: 16 GB of RAM

Graphics: Radeon Pro 555X 4 GB graphics card with Intel UHD integrated Graphics 630 1536 MB

Functional Requirements

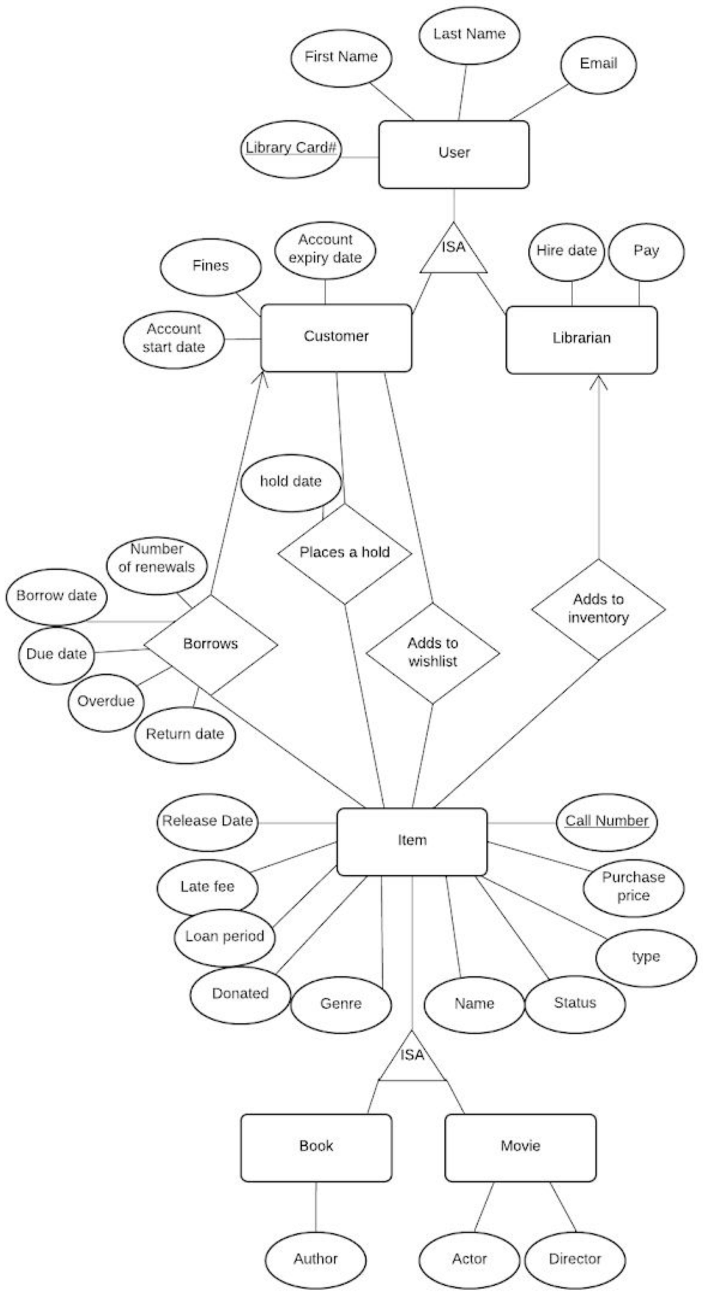
There are two groups of users for this system, the customers of the library and the staff at the library. Library customers will use the system to see specific information about the books they have borrowed, or to find new books to borrow. Library staff will use the system to manage library inventory and the accounts of the library customers. Both groups of users will access the system by logging in through a web page using their email address or library card number and a password.

Features of the system will be developed to support expected use cases for different user groups. Users will visit the home page and either log in using email address or library card number and a password. Based on the type of user logged in, different options will be made available on the following page.

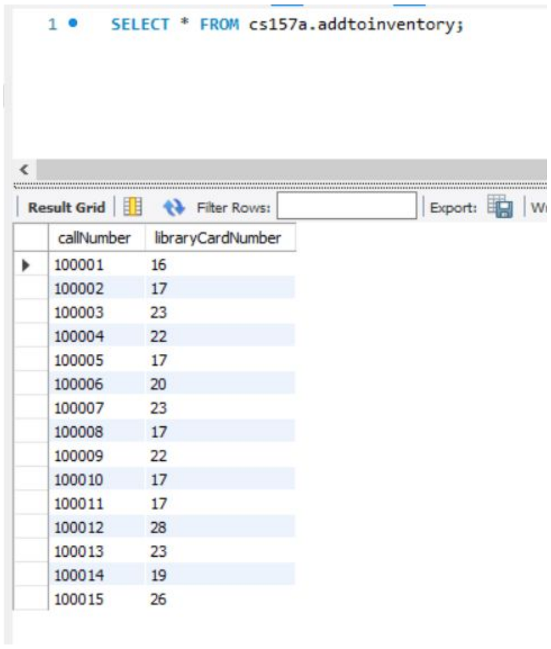
Library customers will be presented with features allowing them to perform tasks such as searching for books/items in the library inventory, viewing their history of borrowed items, place a hold on a book, renew a borrowed item, check due dates for borrowed items, check late fees associated with their account and create a list of books they would like to borrow. Most of these features involve simple queries from the database. To place a hold on a book or add a book to a list to borrow in the future, information in the database will be modified.

Library staff will have additional features that are not accessible to customers. These features will support use cases such as the ability to create a new account for a customer, checkout or return books, add new books to the inventory, change the check-out duration of an item if it is in demand, remove holds and late fees from an account upon receipt of payment, and generate analytic reports of items that are high in demand or of how many books are late. Most of these additional features allow library staff to modify or add to information in the database.

Updated Entity-Relationship Diagram



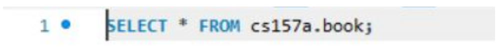
Sample Tables:



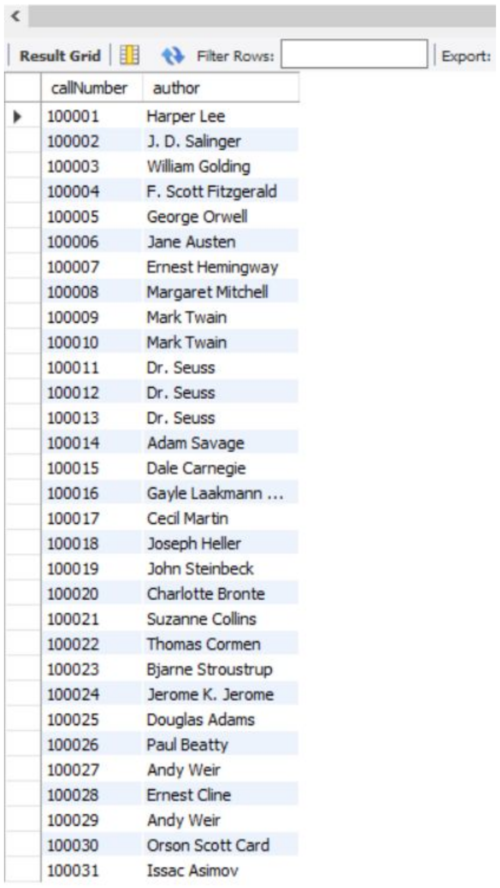
```
SELECT * FROM cs157a.addtoinventory;
```

Result Grid | Filter Rows: | Export:

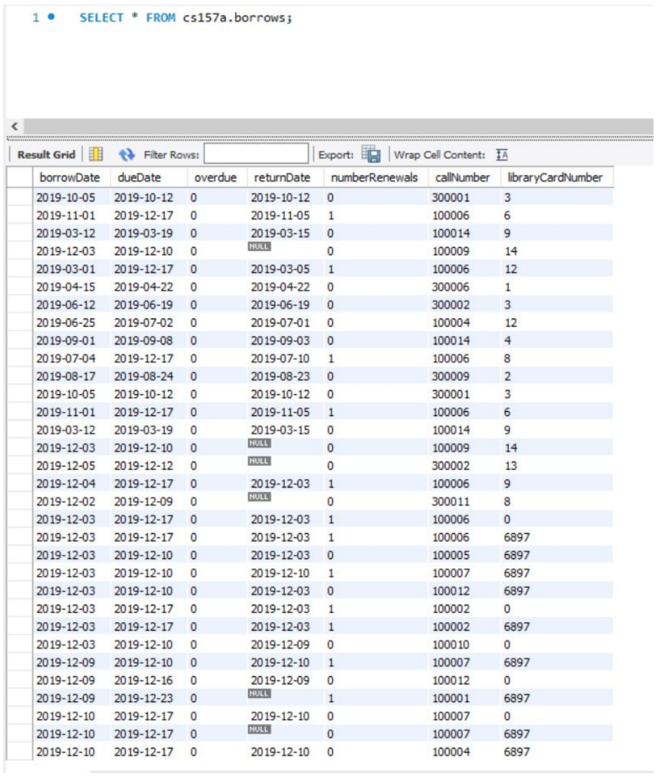
callNumber	libraryCardNumber
100001	16
100002	17
100003	23
100004	22
100005	17
100006	20
100007	23
100008	17
100009	22
100010	17
100011	17
100012	28
100013	23
100014	19
100015	26



```sql
SELECT * FROM cs157a.book;
```

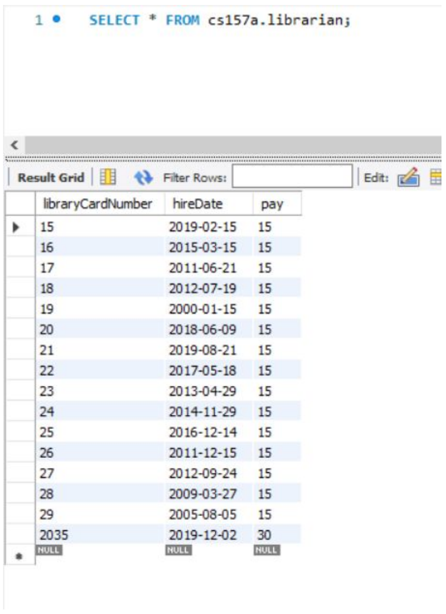


callNumber	author
100001	Harper Lee
100002	J. D. Salinger
100003	William Golding
100004	F. Scott Fitzgerald
100005	George Orwell
100006	Jane Austen
100007	Ernest Hemingway
100008	Margaret Mitchell
100009	Mark Twain
100010	Mark Twain
100011	Dr. Seuss
100012	Dr. Seuss
100013	Dr. Seuss
100014	Adam Savage
100015	Dale Carnegie
100016	Gayle Laakmann ...
100017	Cecil Martin
100018	Joseph Heller
100019	John Steinbeck
100020	Charlotte Bronte
100021	Suzanne Collins
100022	Thomas Cormen
100023	Bjarne Stroustrup
100024	Jerome K. Jerome
100025	Douglas Adams
100026	Paul Beatty
100027	Andy Weir
100028	Ernest Cline
100029	Andy Weir
100030	Orson Scott Card
100031	Issac Asimov



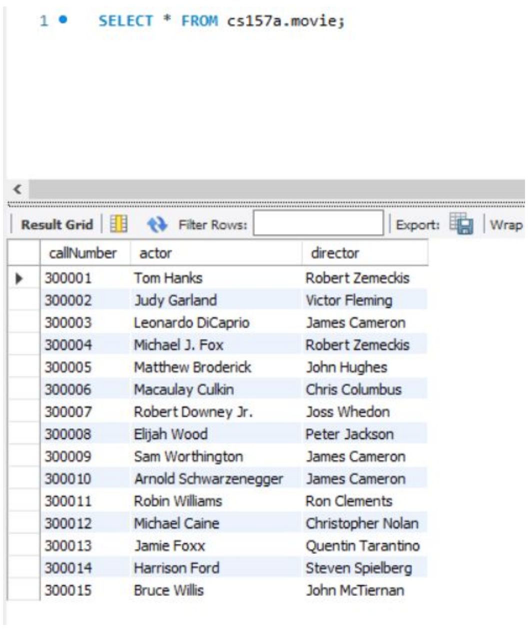
```sql
SELECT * FROM cs157a.borrows;
```

borrowDate	dueDate	overdue	returnDate	numberRenewals	callNumber	libraryCardNumber
2019-10-05	2019-10-12	0	2019-10-12	0	300001	3
2019-11-01	2019-12-17	0	2019-11-05	1	100006	6
2019-03-12	2019-03-19	0	2019-03-15	0	100014	9
2019-12-03	2019-12-10	0	NULL	0	100009	14
2019-03-01	2019-12-17	0	2019-03-05	1	100006	12
2019-04-15	2019-04-22	0	2019-04-22	0	300006	1
2019-06-12	2019-06-19	0	2019-06-19	0	300002	3
2019-06-25	2019-07-02	0	2019-07-01	0	100004	12
2019-09-01	2019-09-08	0	2019-09-03	0	100014	4
2019-07-04	2019-12-17	0	2019-07-10	1	100006	8
2019-08-17	2019-08-24	0	2019-08-23	0	300009	2
2019-10-05	2019-10-12	0	2019-10-12	0	300001	3
2019-11-01	2019-12-17	0	2019-11-05	1	100006	6
2019-03-12	2019-03-19	0	2019-03-15	0	100014	9
2019-12-03	2019-12-10	0	NULL	0	100009	14
2019-12-05	2019-12-12	0	NULL	0	300002	13
2019-12-04	2019-12-17	0	2019-12-03	1	100006	9
2019-12-02	2019-12-09	0	NULL	0	300011	8
2019-12-03	2019-12-17	0	2019-12-03	1	100006	0
2019-12-03	2019-12-17	0	2019-12-03	1	100006	6897
2019-12-03	2019-12-10	0	2019-12-03	0	100005	6897
2019-12-03	2019-12-10	0	2019-12-10	1	100007	6897
2019-12-03	2019-12-10	0	2019-12-03	0	100012	6897
2019-12-03	2019-12-17	0	2019-12-03	1	100002	0
2019-12-03	2019-12-17	0	2019-12-03	1	100002	6897
2019-12-03	2019-12-10	0	2019-12-09	0	100010	0
2019-12-09	2019-12-10	0	2019-12-10	1	100007	6897
2019-12-09	2019-12-16	0	2019-12-09	0	100012	0
2019-12-09	2019-12-23	0	NULL	1	100001	6897
2019-12-10	2019-12-17	0	2019-12-10	0	100007	0
2019-12-10	2019-12-17	0	NULL	0	100007	6897
2019-12-10	2019-12-17	0	2019-12-10	0	100004	6897



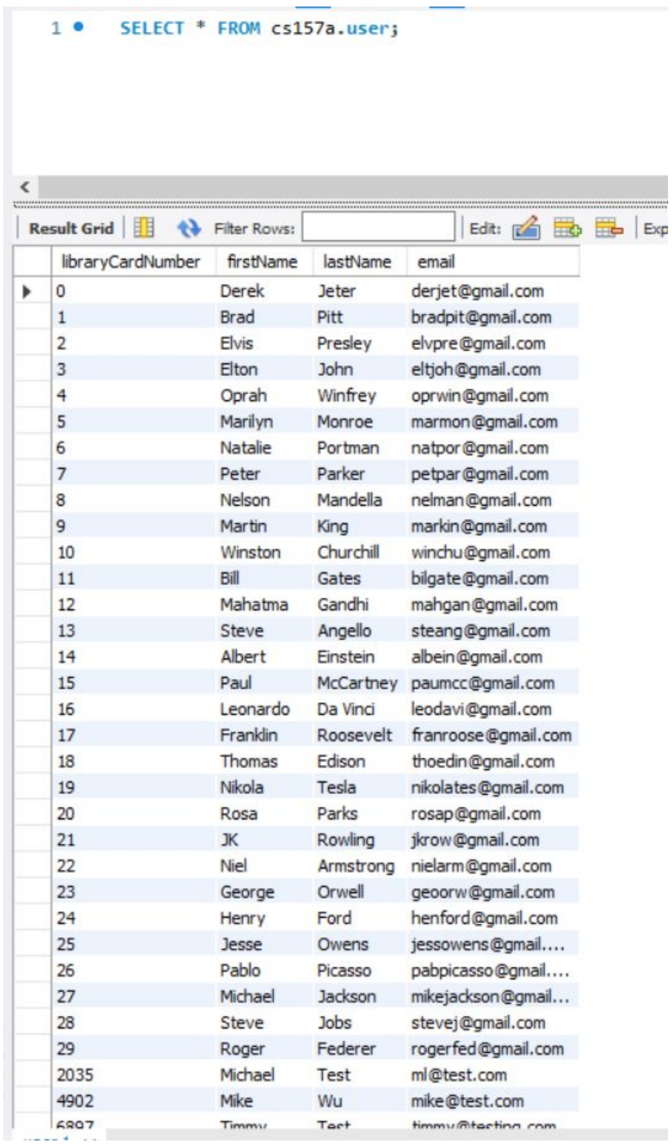
```sql
SELECT * FROM cs157a.librarian;
```

libraryCardNumber	hireDate	pay
15	2019-02-15	15
16	2015-03-15	15
17	2011-06-21	15
18	2012-07-19	15
19	2000-01-15	15
20	2018-06-09	15
21	2019-08-21	15
22	2017-05-18	15
23	2013-04-29	15
24	2014-11-29	15
25	2016-12-14	15
26	2011-12-15	15
27	2012-09-24	15
28	2009-03-27	15
29	2005-08-05	15
2035	2019-12-02	30
NULL	NULL	NULL



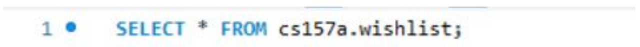
```
SELECT * FROM cs157a.movie;
```

callNumber	actor	director
300001	Tom Hanks	Robert Zemeckis
300002	Judy Garland	Victor Fleming
300003	Leonardo DiCaprio	James Cameron
300004	Michael J. Fox	Robert Zemeckis
300005	Matthew Broderick	John Hughes
300006	Macaulay Culkin	Chris Columbus
300007	Robert Downey Jr.	Joss Whedon
300008	Elijah Wood	Peter Jackson
300009	Sam Worthington	James Cameron
300010	Arnold Schwarzenegger	James Cameron
300011	Robin Williams	Ron Clements
300012	Michael Caine	Christopher Nolan
300013	Jamie Foxx	Quentin Tarantino
300014	Harrison Ford	Steven Spielberg
300015	Bruce Willis	John McTiernan

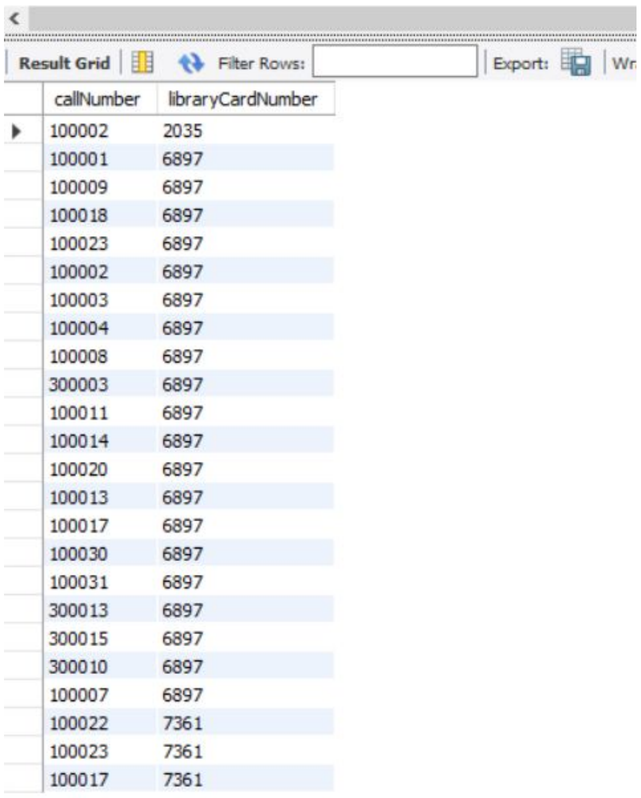


```sql
SELECT * FROM cs157a.user;
```

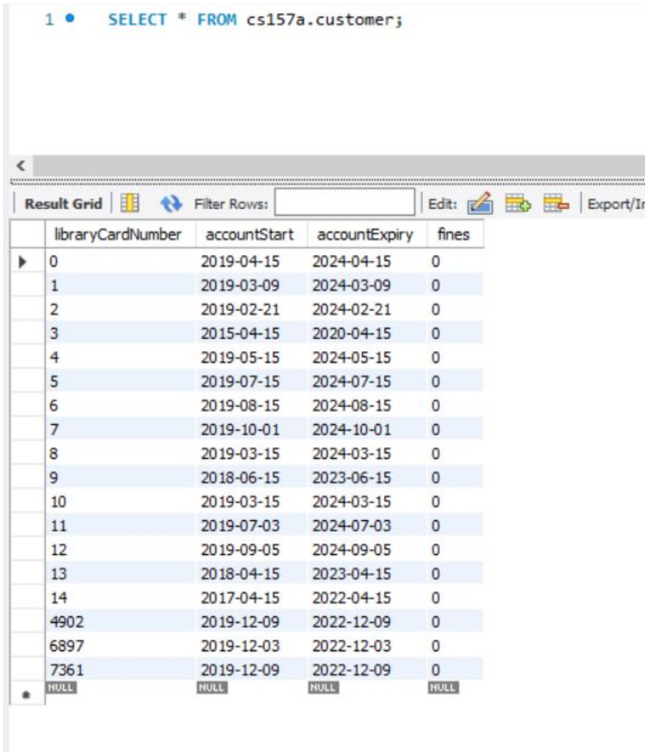
libraryCardNumber	firstName	lastName	email
0	Derek	Jeter	derjet@gmail.com
1	Brad	Pitt	bradpit@gmail.com
2	Elvis	Presley	elvpre@gmail.com
3	Elton	John	eltjoh@gmail.com
4	Oprah	Winfrey	oprwin@gmail.com
5	Marilyn	Monroe	marmon@gmail.com
6	Natalie	Portman	natpor@gmail.com
7	Peter	Parker	petpar@gmail.com
8	Nelson	Mandella	nelman@gmail.com
9	Martin	King	markin@gmail.com
10	Winston	Churchill	winchu@gmail.com
11	Bill	Gates	bilgate@gmail.com
12	Mahatma	Gandhi	mahgan@gmail.com
13	Steve	Angello	steang@gmail.com
14	Albert	Einstein	albein@gmail.com
15	Paul	McCartney	paumcc@gmail.com
16	Leonardo	Da Vinci	leodavi@gmail.com
17	Franklin	Roosevelt	franroose@gmail.com
18	Thomas	Edison	thoedin@gmail.com
19	Nikola	Tesla	nikolates@gmail.com
20	Rosa	Parks	rosap@gmail.com
21	JK	Rowling	jkrow@gmail.com
22	Niel	Armstrong	nielarm@gmail.com
23	George	Orwell	geoorw@gmail.com
24	Henry	Ford	henford@gmail.com
25	Jesse	Owens	jessowens@gmail....
26	Pablo	Picasso	pabpicasso@gmail....
27	Michael	Jackson	mikejackson@gmail...
28	Steve	Jobs	stevej@gmail.com
29	Roger	Federer	rogerfed@gmail.com
2035	Michael	Test	ml@test.com
4902	Mike	Wu	mike@test.com
6897	Timmy	Test	timmy@testing.com



```sql
SELECT * FROM cs157a.wishlist;
```

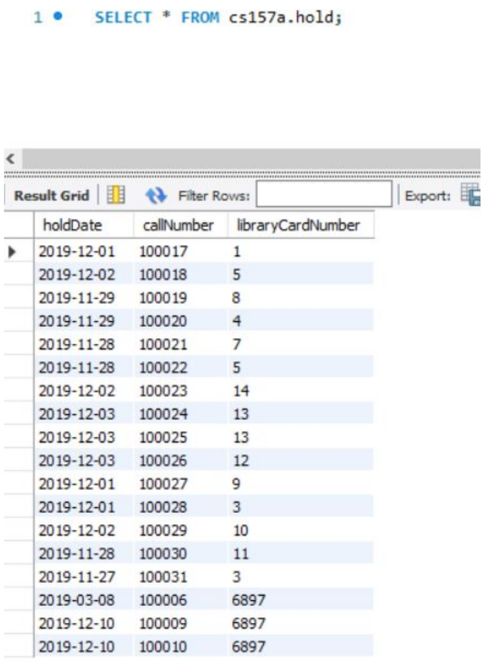


callNumber	libraryCardNumber
100002	2035
100001	6897
100009	6897
100018	6897
100023	6897
100002	6897
100003	6897
100004	6897
100008	6897
300003	6897
100011	6897
100014	6897
100020	6897
100013	6897
100017	6897
100030	6897
100031	6897
300013	6897
300015	6897
300010	6897
100007	6897
100022	7361
100023	7361
100017	7361



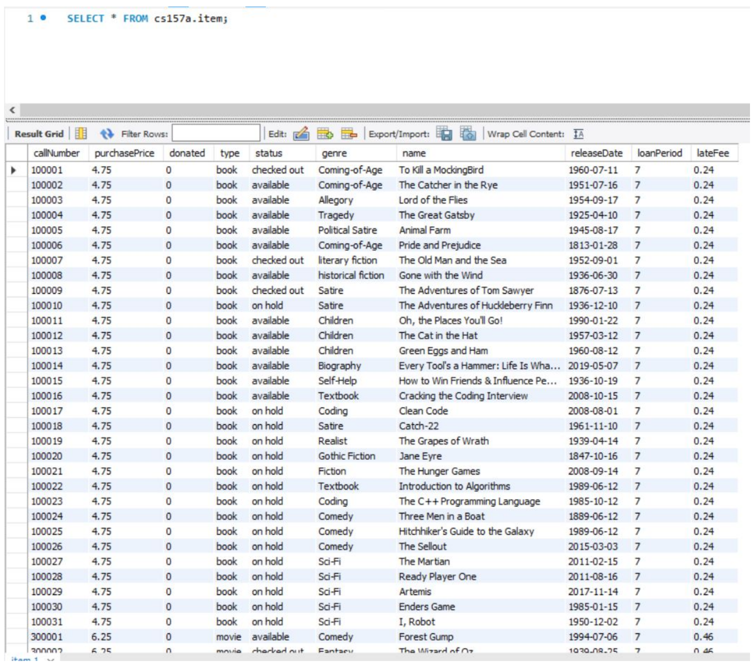
```
SELECT * FROM cs157a.customer;
```

libraryCardNumber	accountStart	accountExpiry	fines
0	2019-04-15	2024-04-15	0
1	2019-03-09	2024-03-09	0
2	2019-02-21	2024-02-21	0
3	2015-04-15	2020-04-15	0
4	2019-05-15	2024-05-15	0
5	2019-07-15	2024-07-15	0
6	2019-08-15	2024-08-15	0
7	2019-10-01	2024-10-01	0
8	2019-03-15	2024-03-15	0
9	2018-06-15	2023-06-15	0
10	2019-03-15	2024-03-15	0
11	2019-07-03	2024-07-03	0
12	2019-09-05	2024-09-05	0
13	2018-04-15	2023-04-15	0
14	2017-04-15	2022-04-15	0
4902	2019-12-09	2022-12-09	0
6897	2019-12-03	2022-12-03	0
7361	2019-12-09	2022-12-09	0
NULL	NULL	NULL	NULL



```
SELECT * FROM cs157a.hold;
```

holdDate	callNumber	libraryCardNumber
2019-12-01	100017	1
2019-12-02	100018	5
2019-11-29	100019	8
2019-11-29	100020	4
2019-11-28	100021	7
2019-11-28	100022	5
2019-12-02	100023	14
2019-12-03	100024	13
2019-12-03	100025	13
2019-12-03	100026	12
2019-12-01	100027	9
2019-12-01	100028	3
2019-12-02	100029	10
2019-11-28	100030	11
2019-11-27	100031	3
2019-03-08	100006	6897
2019-12-10	100009	6897
2019-12-10	100010	6897



```
SELECT * FROM cs157a.item;
```

callNumber	purchasePrice	donated	type	status	genre	name	releaseDate	loanPeriod	lateFee
100001	4.75	0	book	checked out	Coming-of-Age	To Kill a MockingBird	1960-07-11	7	0.24
100002	4.75	0	book	available	Coming-of-Age	The Catcher in the Rye	1951-07-16	7	0.24
100003	4.75	0	book	available	Allegory	Lord of the Flies	1954-09-17	7	0.24
100004	4.75	0	book	available	Tragedy	The Great Gatsby	1925-04-10	7	0.24
100005	4.75	0	book	available	Political Satire	Animal Farm	1945-08-17	7	0.24
100006	4.75	0	book	available	Coming-of-Age	Pride and Prejudice	1813-01-28	7	0.24
100007	4.75	0	book	checked out	literary fiction	The Old Man and the Sea	1952-09-01	7	0.24
100008	4.75	0	book	available	historical fiction	Gone with the Wind	1936-06-30	7	0.24
100009	4.75	0	book	checked out	Satire	The Adventures of Tom Sawyer	1876-07-13	7	0.24
100010	4.75	0	book	on hold	Satire	The Adventures of Huckleberry Finn	1936-12-10	7	0.24
100011	4.75	0	book	available	Children	Oh, the Places You'll Go!	1990-01-22	7	0.24
100012	4.75	0	book	available	Children	The Cat in the Hat	1957-03-12	7	0.24
100013	4.75	0	book	available	Children	Green Eggs and Ham	1960-08-12	7	0.24
100014	4.75	0	book	available	Biography	Every Tool's a Hammer: Life Is Wha...	2019-05-07	7	0.24
100015	4.75	0	book	available	Self-Help	How to Win Friends & Influence Pe...	1936-10-19	7	0.24
100016	4.75	0	book	available	Textbook	Cracking the Coding Interview	2008-10-15	7	0.24
100017	4.75	0	book	on hold	Coding	Clean Code	2008-08-01	7	0.24
100018	4.75	0	book	on hold	Satire	Catch-22	1961-11-10	7	0.24
100019	4.75	0	book	on hold	Realist	The Grapes of Wrath	1939-04-14	7	0.24
100020	4.75	0	book	on hold	Gothic Fiction	Jane Eyre	1847-10-16	7	0.24
100021	4.75	0	book	on hold	Fiction	The Hunger Games	2008-09-14	7	0.24
100022	4.75	0	book	on hold	Textbook	Introduction to Algorithms	1989-06-12	7	0.24
100023	4.75	0	book	on hold	Coding	The C++ Programming Language	1985-10-12	7	0.24
100024	4.75	0	book	on hold	Comedy	Three Men in a Boat	1889-06-12	7	0.24
100025	4.75	0	book	on hold	Comedy	Hitchhiker's Guide to the Galaxy	1989-06-12	7	0.24
100026	4.75	0	book	on hold	Comedy	The Sellout	2015-03-03	7	0.24
100027	4.75	0	book	on hold	Sci-Fi	The Martian	2011-02-15	7	0.24
100028	4.75	0	book	on hold	Sci-Fi	Ready Player One	2011-08-16	7	0.24
100029	4.75	0	book	on hold	Sci-Fi	Artemis	2017-11-14	7	0.24
100030	4.75	0	book	on hold	Sci-Fi	Enders Game	1985-01-15	7	0.24
100031	4.75	0	book	on hold	Sci-Fi	I, Robot	1950-12-02	7	0.24
300001	6.25	0	movie	available	Comedy	Forest Gump	1994-07-06	7	0.46
300002	6.25	0	movie	checked out	Fantasy	The Wizard of Oz	1939-08-25	7	0.46

Functional Dependencies:

For item: An item is any item in the library inventory that can be loaned to customers. It is the superclass of book and movie. It is made to support other types of items such if they were to be created

CallNumber -> Name Price Status Genre Donated LoanPeriod LateFee PublishDate

For movie: A movie is a type of item. Movies have the additional fields of the single lead actor and a director. It represents the movie subsection in the digital media section of a real library

CallNumber -> Actor Director

For book: A book is a type of item. Books have the additional attribute author. This is to represent the most important item in the library.

CallNumber -> Author

For user: A user is any person that will use the system. They are the superclass used by customer and librarian.

LibCardNumber -> Name Email Password

For customer: A customer is a type of user. Customers are the user accounts that are not administrators. They also are the standard “user” outside of the business.

LibCardNumber -> Fines StartDate ExpireDate

For librarian: A librarian is a type of user. Librarians have additional administrator privileges like checking in and out books, pay fines, and adding and removing books to the library.

LibCardNumber -> hireDate Pay

For hold: Holds represent holds on items in the library. A hold is when an item that is checked out is requested by another user. Therefore when it is checked back in, it’s status is on hold to the most recent person who put a hold on the item.

CallNumber LibCardNumber -> dueDate

For borrows: Borrows are the record for a general transaction at the library. Checking an item out adds it to the borrows table. It represents the history of items checked out as well.

CallNumber LibCardNumber -> BorrowDate DueDate Overdue RenewNumber

For addToInventory: addToInventory is another table storing the records for items so a history can be recorded. When an item is added to the library's database, it records who added it and what the call number is. This history allows administrators to see who added what books.

No FDs

For wishlist: wishlist is a table that allows users to have books they see and want to read all in one list.

No FDs

Implementation

The backend server has different routes which handle HTTP POST and GET requests coming from the front-end React component. All the database insertion, deletion, and update processes are handled by the backend component (Node.js and Express.js) using the MySQL npm package. Google's firebase is used to authenticate existing and new users by sending the password the user enters to be authenticated. Firebase also handles the login and log out functionalities as well as the user sessions. All other user-entered information in the sign up process is handled by our backend server and is stored in the MySQL database locally.

The frontend is implemented as a single page application using React.js as well as the Material UI library for styling different components. React router handles different routes and navigating to different pages on the frontend while the npm axios package makes HTTP requests to the backend server which handles routes through Express.js

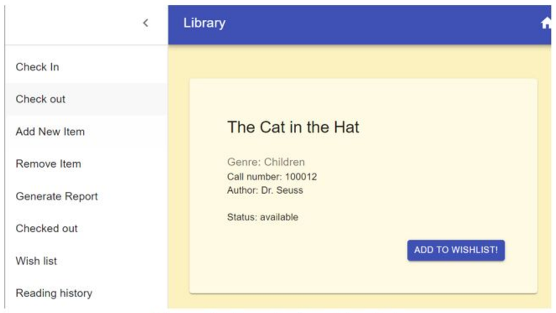
The database we used is MySQL database. We updated the data to our local database using MySQL workbench as the server for the database.

Functions

Check in item

- Library staff shall be able to change the status of an item in the database from “checked-out” to “available” by navigating to the “check in” page and entering the call number of the item to be checked in.

- The system shall reflect those changes for all users to see.

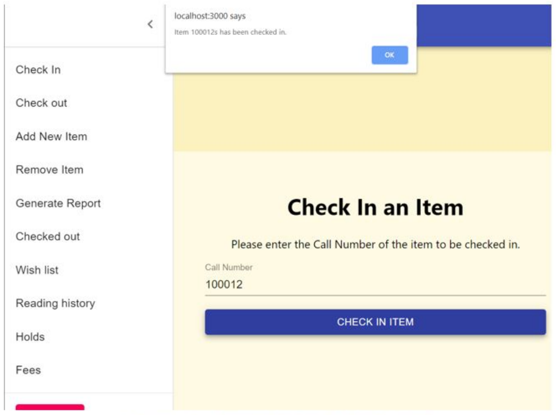


The Cat in the Hat

Genre: Children
Call number: 100012
Author: Dr. Seuss

Status: available

Figure 1: Item before Check-In



localhost:3000 says

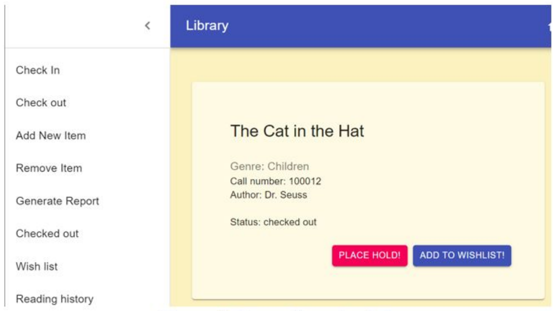
Item 100012s has been checked in.

Check In an Item

Please enter the Call Number of the item to be checked in.

Call Number
100012

Figure 2: Librarian checks in the item.



The Cat in the Hat

Genre: Children
Call number: 100012
Author: Dr. Seuss

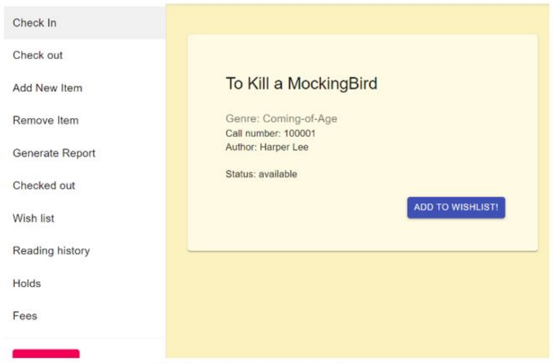
Status: checked out

Figure 3: Item after check-in

Check out item

- Library staff shall be able to check out an item to a user by navigating to the “check out” page and entering the library card of the user and the call number of the item to be checked out

- The system shall verify that the item is not already checked out or placed on hold. It shall also update the status of the item in the database and add the item to the checked-out list of the user with the date and time it was checked out.

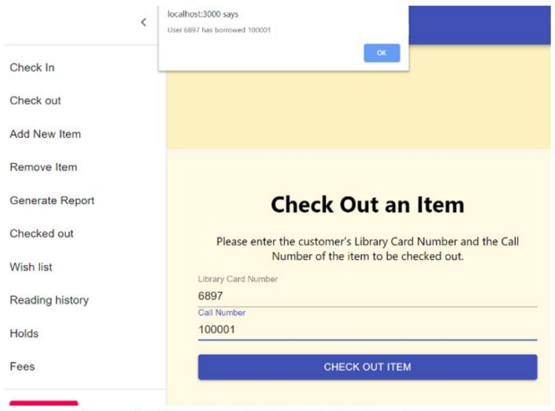


To Kill a MockingBird

Genre: Coming-of-Age
Call number: 100001
Author: Harper Lee

Status: available

Figure 4: Item before check out.



localhost:3000 says

User 6897 has borrowed 100001

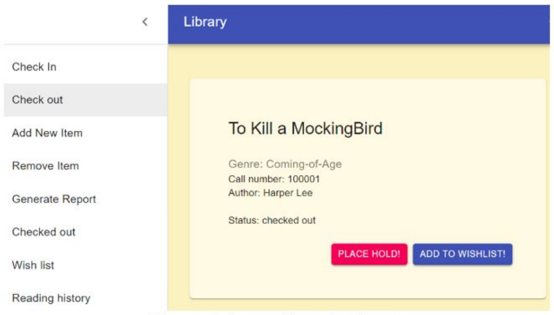
Check Out an Item

Please enter the customer's Library Card Number and the Call Number of the item to be checked out.

Library Card Number: 6897

Call Number: 100001

Figure 5: Librarian checks out the item to a user.



To Kill a MockingBird

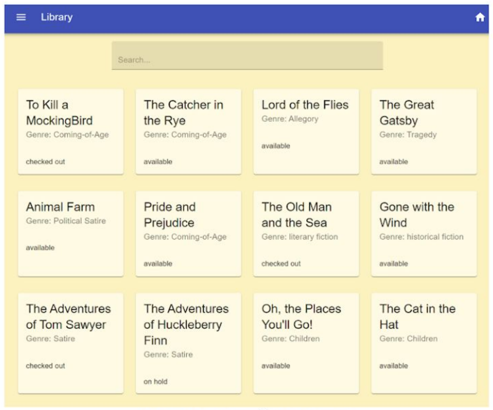
Genre: Coming-of-Age
Call number: 100001
Author: Harper Lee

Status: checked out

Figure 6: Item after check out.

Browse inventory and Search for item

- Library staff/users shall be able to browse all items in inventory whether on hold, checked in, or checked out. They will be able to click on any item to view further details.
- Library staff/users shall be able to search the inventory for items by providing information about the item such as name, genre, or author in a search bar.



To Kill a MockingBird
Genre: Coming-of-Age
checked out

The Catcher in the Rye
Genre: Coming-of-Age
available

Lord of the Flies
Genre: Allegory
available

The Great Gatsby
Genre: Tragedy
available

Animal Farm
Genre: Political Satire
available

Pride and Prejudice
Genre: Coming-of-Age
available

The Old Man and the Sea
Genre: literary fiction
checked out

Gone with the Wind
Genre: historical fiction
available

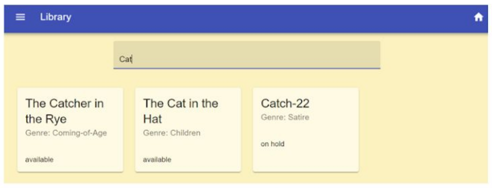
The Adventures of Tom Sawyer
Genre: Satire
checked out

The Adventures of Huckleberry Finn
Genre: Satire
on hold

Oh, the Places You'll Go!
Genre: Children
available

The Cat in the Hat
Genre: Children
available

Figure 7: View library items



Cat

The Catcher in the Rye
Genre: Coming-of-Age
available

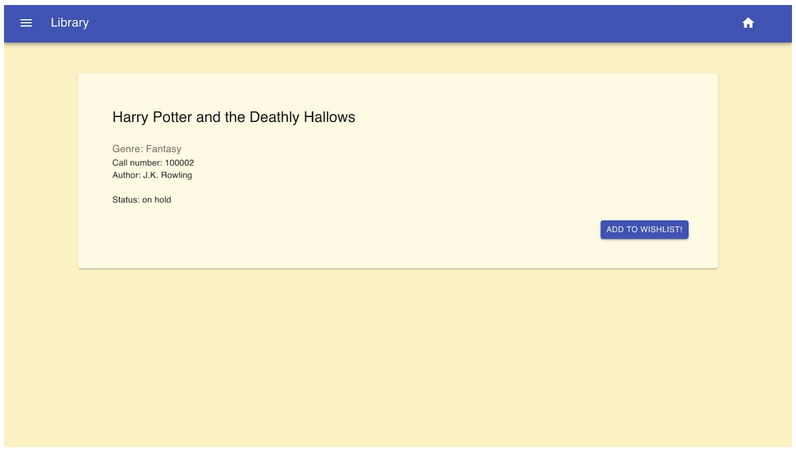
The Cat in the Hat
Genre: Children
available

Catch-22
Genre: Satire
on hold

Figure 8: Search for items.

View item information

- Library staff/users shall be able to view item information by clicking the item from the main items page.
- The system shall display additional information about the item. The system will also allow users to add the item to their wish list or place the item on hold if it is checked out.



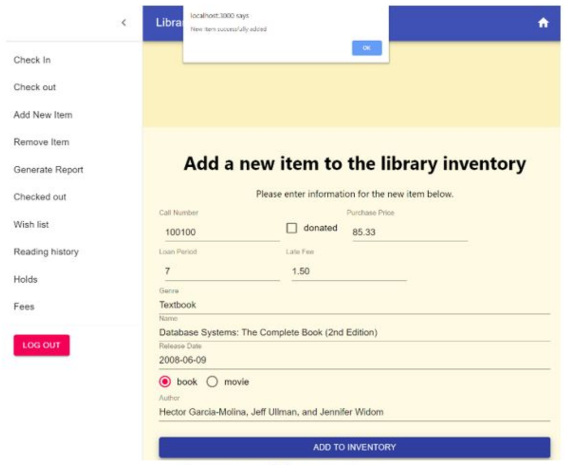
Harry Potter and the Deathly Hallows

Genre: Fantasy
Call number: 100002
Author: J.K. Rowling

Status: on hold

Add/Remove item

- Library staff shall be able to add an item to the library inventory by adding its information (such as name, barcode, call number, checkout duration, …) to the database. Items can come from purchases or donations.
- Library staff shall be able to remove an item from the library inventory by navigating to the “Remove Item” page and entering the call number.
- The system shall allow staff members to add items by filling out a form. The system shall also update the database with information entered and reflect it on the web application.



localhost:3000 says

New item successfully added

Add a new item to the library inventory

Please enter information for the new item below.

Call Number: 100100

☐ donated

Purchase Price: 85.33

Loan Period: 7

Late Fee: 1.50

Genre: Textbook

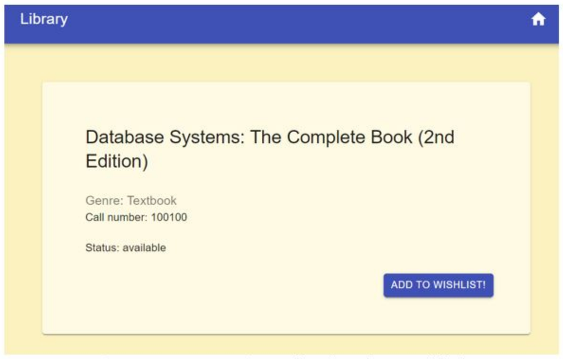
Name: Database Systems: The Complete Book (2nd Edition)

Release Date: 2008-06-09

[✓ book] movie

Author: Hector Garcia-Molina, Jeff Ullman, and Jennifer Widom

Figure 9: Add a new item.



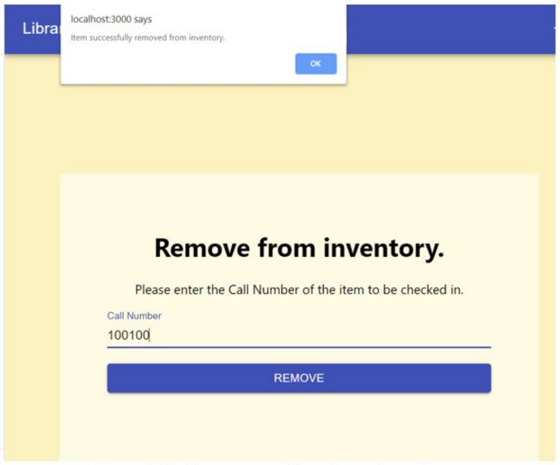
Database Systems: The Complete Book (2nd Edition)

Genre: Textbook

Call number: 100100

Status: available

Figure 10: A new item that has been added.



localhost:3000 says

Item successfully removed from inventory.

Remove from inventory.

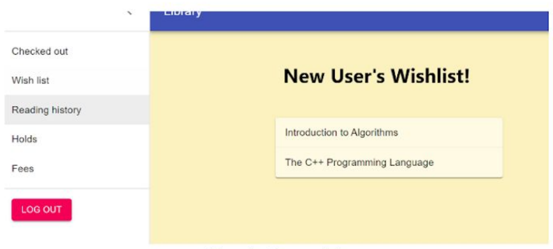
Please enter the Call Number of the item to be checked in.

Call Number: 100100

Figure 11: Remove an item from inventory.

Add item to wishlist

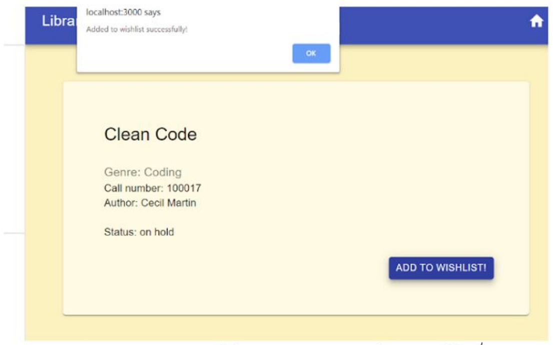
- Library users shall be to click an “Add to wish list” button next to an item being viewed.
- The system shall add that item to an ordered list of items the user would like to check out in the future.



New User's Wishlist!

Introduction to Algorithms
The C++ Programming Language

Figure 12: Wishlist before adding a new item.



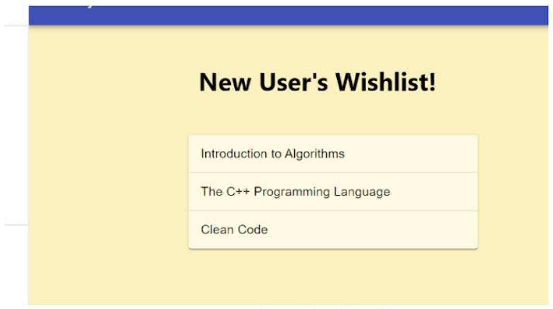
localhost:3000 says
Added to wishlist successfully!

Clean Code

Genre: Coding
Call number: 100017
Author: Cecil Martin

Status: on hold

Figure 13: User adds a new item to their wishlist.



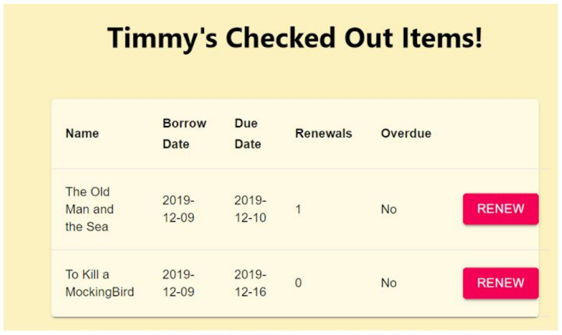
New User's Wishlist!

Introduction to Algorithms
The C++ Programming Language
Clean Code

Figure 14: User's updated wishlist.

Renew an item

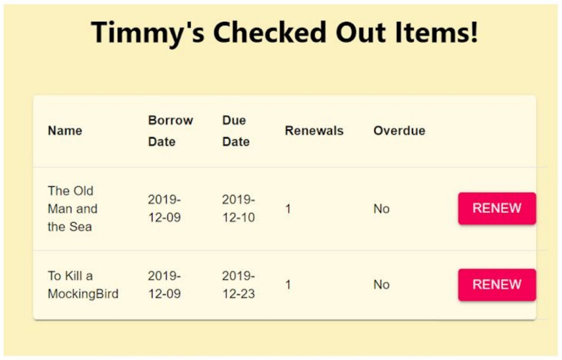
- Users of the system shall be able to click a "Renew" button when viewing item(s) that they currently have checked out.
- The system will update the due date for the item.



Timmy's Checked Out Items!

Name	Borrow Date	Due Date	Renewals	Overdue	
The Old Man and the Sea	2019-12-09	2019-12-10	1	No	RENEW
To Kill a MockingBird	2019-12-09	2019-12-16	0	No	RENEW

Figure 15: User's currently checked out items.



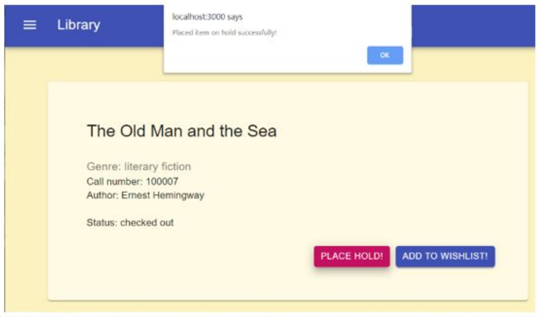
Timmy's Checked Out Items!

Name	Borrow Date	Due Date	Renewals	Overdue	
The Old Man and the Sea	2019-12-09	2019-12-10	1	No	RENEW
To Kill a MockingBird	2019-12-09	2019-12-23	1	No	RENEW

Figure 16: After the user renews an item.

Place hold on item

- When viewing an item, if the item status is not “available” users shall be able to click a button to place a hold on the item.
- The system will add the user to an ordered list of users waiting to check out the item.
- The system will prevent users from checking out the item unless they are the next user on the hold list, or the list is empty.

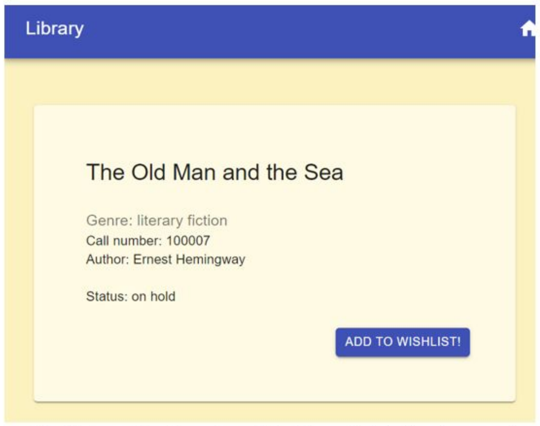

localhost:3000 says
Placed item on hold successfully!

The Old Man and the Sea

Genre: literary fiction
Call number: 100007
Author: Ernest Hemingway

Status: checked out

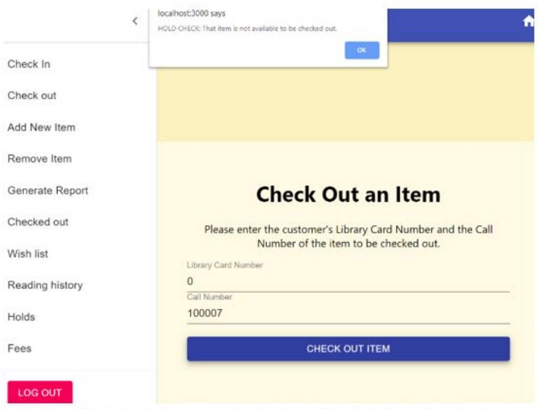
Figure 17: User places a hold on an item that is currently checked out.


The Old Man and the Sea

Genre: literary fiction
Call number: 100007
Author: Ernest Hemingway

Status: on hold

Figure 18: Status of the item is updated to on hold after it was returned.

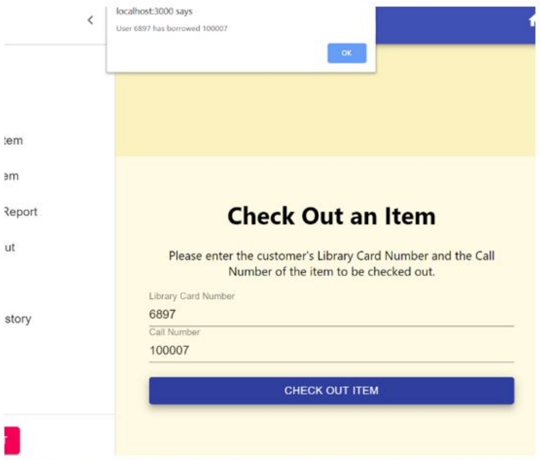

localhost:3000 says
HOLD CHECK: That item is not available to be checked out.

Check Out an Item

Please enter the customer's Library Card Number and the Call Number of the item to be checked out.

Library Card Number: 0
Call Number: 100007

Figure 19: The system prevents the item from being checked out to a user other than the one who placed the hold.


localhost:3000 says
User 6897 has borrowed 100007

Check Out an Item

Please enter the customer's Library Card Number and the Call Number of the item to be checked out.

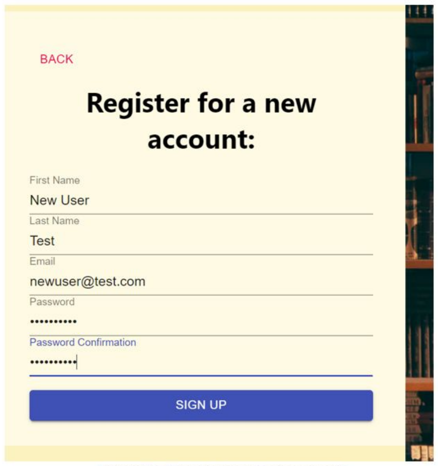
Library Card Number: 6897
Call Number: 100007

Figure 20: The user that placed the hold is allowed to check out the item.

Create account

- Library users shall be able to create a new account by choosing “Sign up Now!” from the main login page and entering the user information such as name, email address, and password.

- The system shall update the database with the information for the new user.



Register for a new account:

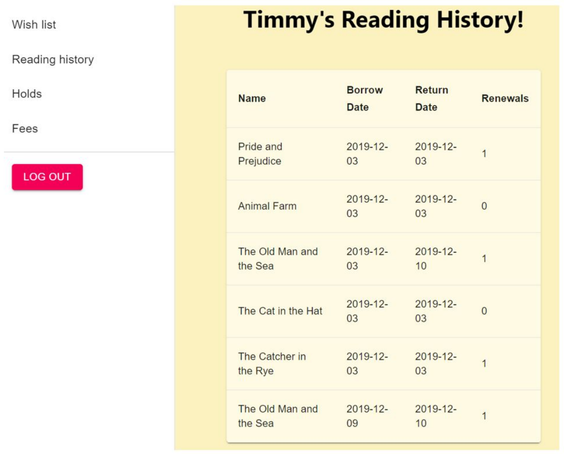
First Name: New User
Last Name: Test
Email: newuser@test.com
Password: ••••••••••
Password Confirmation: ••••••••••

Figure 21: Create a new user account

View reading history

- When viewing their account, users shall be able to click a button to view reading history.

- The system shall display a list of all items the user has previously checked out.



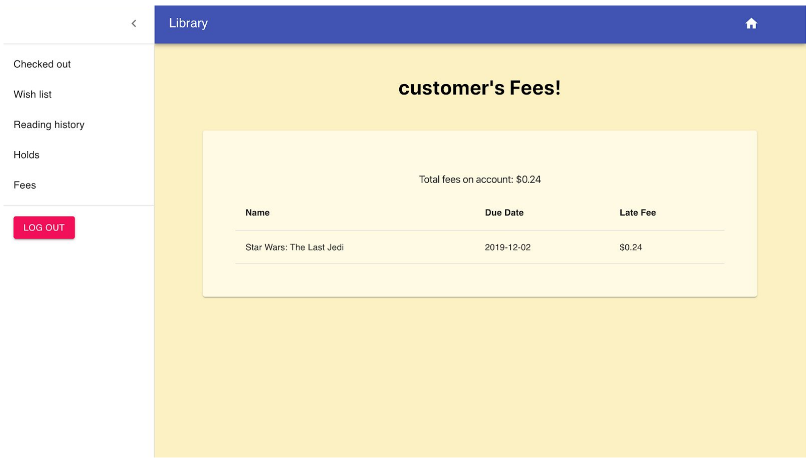
Timmy's Reading History!

Name	Borrow Date	Return Date	Renewals
Pride and Prejudice	2019-12-03	2019-12-03	1
Animal Farm	2019-12-03	2019-12-03	0
The Old Man and the Sea	2019-12-03	2019-12-10	1
The Cat in the Hat	2019-12-03	2019-12-03	0
The Catcher in the Rye	2019-12-03	2019-12-03	1
The Old Man and the Sea	2019-12-09	2019-12-10	1

Figure 22: View reading history.

View late fees

- While looking at account details, library users can clearly see any outstanding fines they need to pay
- System will display current fines and when clicked, will provide additional information regarding what incurred the fine



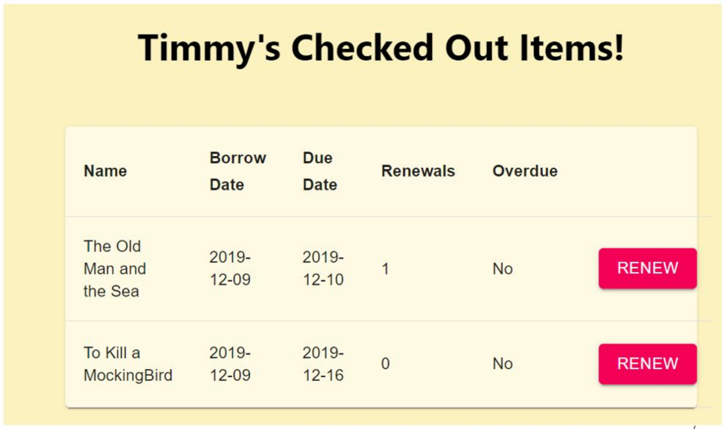
customer's Fees!

Total fees on account: $0.24

Name	Due Date	Late Fee
Star Wars: The Last Jedi	2019-12-02	$0.24

View due date of items

- While viewing their account, users can click to see their checkout history displayed as a list
- Currently checked out books will show their check-out date



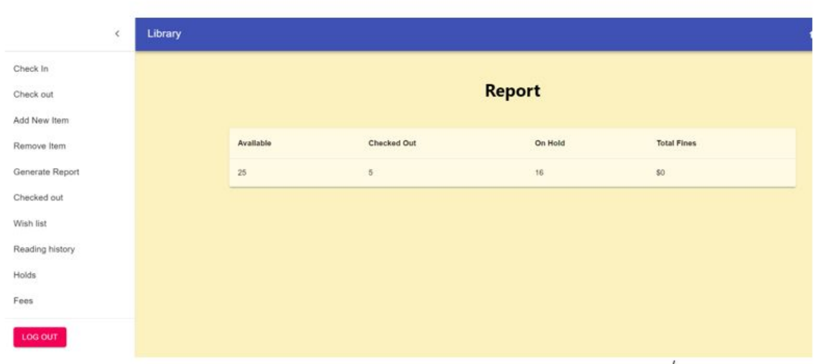
Timmy's Checked Out Items!

Name	Borrow Date	Due Date	Renewals	Overdue	
The Old Man and the Sea	2019-12-09	2019-12-10	1	No	RENEW
To Kill a MockingBird	2019-12-09	2019-12-16	0	No	RENEW

Figure 23: User can see the due date of checked out items.

Generate analytic report

- Library staff can generate a detailed report to see analytic data the system collects about the library
- The report will show:
  - The current amount of books in stock, checked out, and on hold.
  - Total fines owed to the library



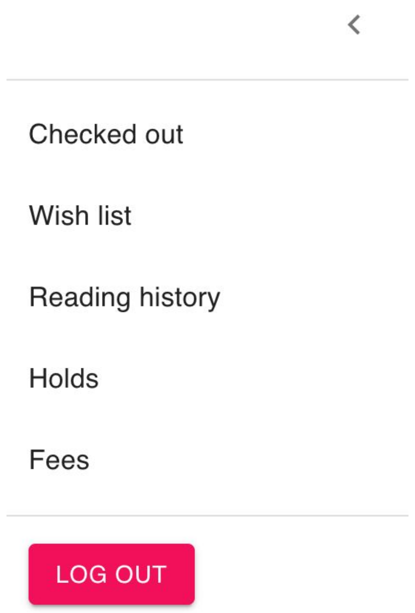
Report

Available	Checked Out	On Hold	Total Fines
25	5	16	$0

Figure 24: Report generated by librarian.

Login/Logout

- The user will be able to log in and out from any computer
- Users can then view any account information from there



Non-Functional Requirements

Performance

- Application response time for each use case should be efficient.
    - No transactions or queries with the exception of generating the analytic report should take no longer than 0.5 seconds. The analytic report feature is expected to be used infrequently and it is acceptable for the report generation to take a few seconds to complete.

GUI

- React.js will be used along with other UI frameworks to develop the GUI for the application.
    - After the initial login page, the user interface will include a sidebar with buttons to access different features depending on the type of user account logged in. The main portion of the user interface will be for displaying information or text entry fields. The information displayed could be the result of a search for a book, a particular user's reading history or wish list, a list display of all books due on a particular day, or the results of one of the analytic reports. Text entry fields would be displayed to support features for library staff such as adding new items to inventory or adding new user accounts.

Security

- Handle login/logout for different types of users.
    - Users will login to the system using a combination of their password and either their email address or ID number. Users that are classified as library staff

will have access to the full set of features in the application while users that are classified as library customers will have access only to a limited feature set.

- Securely store passwords in database.
    - Passwords will not be stored in plain text but will first be hashed and salted.
- Allow users to reset passwords securely if they forget their credentials
    - Users can request a password reset using their email address. A temporary password will be emailed to the user and they will be prompted to change their password once they have logged in to the system.

Setup Guide

1) Make sure you have the following installed: Node.js, MySQL, MySQL Workbench
2) Clone GitHub repository from https://github.com/CS157A-Team-22/CS157A-Team-22
3) Make the cs157a database your default schema in MySQL workbench
4) Open the "DEMO_DB.sql" file in Workbench and run the script to upload the data in your database
5) cd to your cloned repository folder on the terminal
6) Download back-end dependencies by using "cd backend" then "npm install"
7) Start back-end server by using "node index.js"
8) Open a new terminal window, cd to the root of your cloned repository folder
9) Download front-end dependencies by using "cd frontend" then "npm install"
10) Create a "development.env" file in your frontend folder and paste in it the following Firebase API key information:
```
API_KEY=AIzaSyDJ6B_VTPYq084_nGdcglJPTkPJbGtpHoo
AUTH_DOMAIN=cs157a-f19-team22.firebaseapp.com
DATABASE_URL=https://cs157a-f19-team22.firebaseio.com
PROJECT_ID=cs157a-f19-team22
STORAGE_BUCKET=cs157a-f19-team22.appspot.com
MESSAGING_SENDER_ID=971699168955
```
11) Save the file then start the front-end application using "npm start"
12) Go to "http://localhost:3000/" to use the application

Project Conclusion

Lessons learned

Sarah: I had some experience working with React applications before starting this project. However, while working on our Library Management System application, I was able to gain significant exposure to different React features that I had not used before such as the React Router and how to handle route navigation on both the frontend and backend. I also gained more robust knowledge of how to make API calls from the frontend to the backend by using the React lifecycle methods. Similarly, my knowledge of MySQL was very basic before working on this project. I was able to write many MySQL queries, which increased my understanding of the material taught in class. I also had a chance to gain more experience working with Node.js and Express.js by writing the logic for some of the routes.

Michael: This project gave me the opportunity to learn more about front end web development using react. I got to learn a little bit about how node/express works and got experience passing data between the frontend and backend using post and get requests. I did not work on many of the SQL queries, but as I was testing the application I got to examine many of the SQL queries to learn how they work. I also had no experience using MySQL Workbench and got a chance to learn how to manage tables using workbench.

Nicholas: I came into this class with no knowledge of database systems or web applications. Since my group was confident and knowledgeable in Javascript and the various libraries required to build a website, I opted to work on mostly the non-Javascript stuff and SQL. While working on this part, I got to spend a lot of time listening to my partners talk about Javascript and learn enough Javascript to implement some of the backend Firebase account integration so passwords are hashed and a session key is stored during the experience. I also learned that React can store environment variables and the file can be git-ignored so you don't upload the API key to a public github. Since I also got to do some SQL, I learned a bunch of stuff about that. I'm still a bit hesitant on some of the more complex commands, but overall I still got a good taste of SQL.

Future improvements

Since the database has a good hierarchy for items already in place, adding additional types of items would be the first improvement. For example, adding CDs, audiobooks, and journals. After that, adjusting the way holds are done so that there is a more formal queue for holds per book and not just a large table. Another two modifications would be modifying the addToInventory table to store the date it was added, and decoupling the actor column from the movie so that multiple actors could be kept track of per movie. Finally, modifying the overall UI slightly. It currently conforms to the basics of the Google Material UI standard, but we could make better use of the more advanced UI tricks such as global color schemes.