CS 157A Final Project Report

Three-Tier Web Application

SJSU Bookie

Cole McKinnon, Jonathan Van, Yu Xiu

Team 4

Advisor: Dr. Mike Wu

Dec. 10, 2019

Content Table

Project Requirement

Project Description

SJSU Bookie will be a web application that is similar to how craiglist operates. The application will be for SJSU students such that they can post used textbooks, and students who are looking for those books can interact and purchase the book off of them. The goal of SJSU Bookie is to give used books a second usage after their owners are done using them for a semester. Users will be able to create an account, search for or create posts for textbooks, view posts, delete posts, filter and/or sort through their posts, and more. SJSU Bookie's functionalities are simple and clear: connect students with other students who require books to help them save money. This application was inspired by the idea of craigslist, and the need to have a more secure version in order to benefit SJSU students. Currently, there exists SJSU Sammy's Buy and Sell feed, however, the sorting algorithm seems clunky, the UI is not friendly for buying, and it leaves much to be desired. This web application will allow students to find, sell, and buy textbooks more efficiently and safely.

System Environment

Bookie Three-Tier Architecture Diagram Explanation

In the diagram below, our client will be run on any browser that has an up to date javascript version. Bookie will be displayed using javascript running with Reactjs. The GUI aspect of Bookie will be displayed through usage of various HTML and TailwindCSS as our main CSS source. The actual application will be hosted on the website hosting program, Netlify. When making a request, a command is sent to our web server using NodeJS, hosted on Heroku. That way we do not depend on using local host when running this application. NodeJS will then send an SQL query to retrieve data from the database. Our Webserver has credentials to access the database that is on Google Cloud Platform. The MySQL database will then send a response, and based on the response that the NodeJS receives, it will send readable information/a JSON form of data back to the front end on ReactJS.

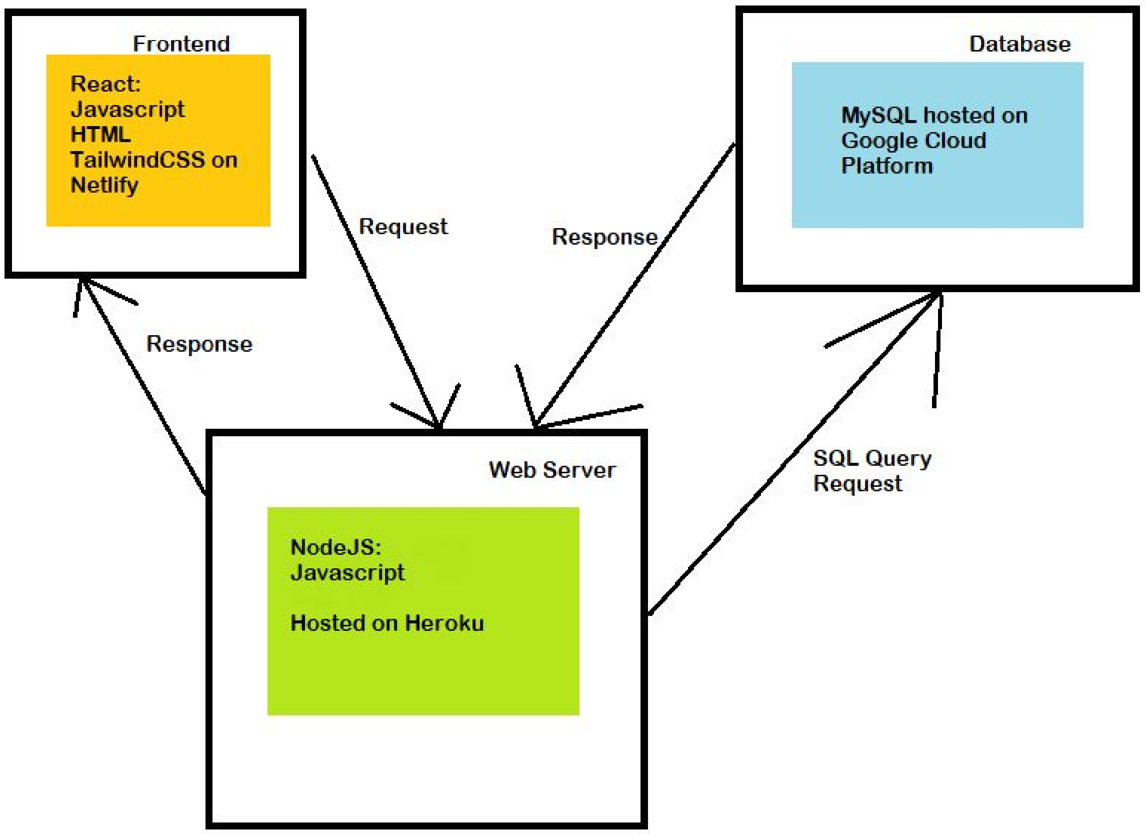


Figure 1: 3 Tier Architecture Design for SJSU Bookie

Hardware & Software

- Netlify to host SJSUBookie
    - We hosted SJSU Bookie on Netlify so that it is not dependent on localhost.
- Heroku to host SJSUBookie's Webserver
    - We hosted SJSU Bookie's NodeJS Webserver on Heroku so that the dynamic server can be accessed by SJSU Bookie's frontend without requiring the local host at all times.
- Google Cloud Platform to host SJSUBookie's MySQL Database
    - By hosting the database online, it can be accessed by SJSU Bookie's backend without having to be dependent on a local machine to host the MySQL Database at all times.
- ReactJS

    - ReactJS is a library that allows us to create a GUI for the user to use the features of SJSU Bookie.
- NodeJS
    - Express, or NodeJS is used to communicate to the MySQL database when the front requires data or wants to make modifications to SJSU Bookie.
- TailwindCSS
    - TailwindCSS simplifies and standardizes CSS such that we do not have to worry about learning much CSS, it is user friendly.
- Github
    - Github will be used to host our code so that Netlify and Heroku can have access and host it. It will also be used as our version control for SJSU Bookie so that we can revert if there are any issues, and collaborate with one another.
- Github Desktop
    - A GUI version of Github in order to make github usage more friendly for those who are not skilled at github commands.
- iMessage
    - iMessage will be used as our means of communication with one another in the case we cannot see each other in person.

Functional Requirements

Bookie is meant to be used by students with other students and/or people affiliated with their same university. All users will sign up to create profiles before buying and selling books. Buying or selling books will be simple, as these are the two most important functions of the system. When signed in, users can easily search through available textbooks by name or course ID via the bar that will be visible from most interfaces.

1. **Login**

    1.1. SJSU Bookie has a login feature in order to grant User controls/commands to the user.

1.2. Login system should grant access when the user enters correct credentials

1.3. Login system will prevent access to incorrect credentials.

2. Sign Up

2.1. SJSU Bookie will have a registration system to allow new users to create accounts.

3. Forget Password

3.1. SJSU Bookie will offer the option to reset a password if a user wants to change or has forgotten their password.

4. Reset Password

4.1. SJSU Bookie will offer the user an option to reset the password if they fulfill the forget password requirements.

5. Searching

5.1. SJSU Bookie will have a search function that is available for users that are not logged in and users that are logged in.

5.2. Search can be done by inputting the name of the book or the course id.

5.3. If there is time SJSU Bookie will allow filtering by price and dates.

6. Posting

6.1. SJSU bookie will allow users to post on its page that the user is selling a textbook. The user will be able to view their own post, edit it, and delete their posting. Any other user can view that page, but they cannot delete or edit it because it is not theirs. They can comment on the post if they want however. A post has the name of the book, the class it is for, and the price the seller is trying to sell it for.

7. Messaging & Friending

7.1. SJSU Bookie allows users to send friend requests to users that they need to communicate with privately in order to make sure the public does not know about private information between the two of them.

8. Profile View

8.1. A user can have access to their profile to modify their password

8.2. A user can have access to their profile to view posts that they have posted, favorited, or posts that have either been closed or deleted.

9. **Friend List**

9.1. SJSU bookie will allow the users to send connection requests to one another in order to message one another.

9.2. Friends cannot be deleted once added.

9.3. Users can send and accept friend requests on SJSU Bookie

Non-functional Issues

1. Graphical User Interface

Our SJSU Bookie used-book sell and buy will be designed mainly as a web application, but we are also working on the phone application. We will ReactJS, which is a JavaScript library, to create our Graphical User Interface. We will have several web pages to serve the need of the application. We have 13 pages, and they are home page, login page, register page, profile editing page, search result page, post and comments page, post creation page, reset page, forget password page, history page, saved post page, friends page, and friend requests page between two users. In general, in each of the page, we will have a toolbar on the top.

1.1 Home page

In home page, we would have a full background picture, and on the top bar, we display our Bookie logo on the left corner, our web page's title, and in the right corner of the top bar, there would be either login, register, or "Hi, user's name". In the middle of the home page, there would be the main section to search for a book's name, course name, and maybe a filter of prices. At the bottom, there is a bar with "Profile", "Comments", "Posts", and "Logout" buttons.

1.2 Login page

From the home page, if the user clicked "login" on the right top corner, the user would be led to the login page, which contains boxes for users to type in their user's name and password. There would be a "CANCEL" and a "CONTINUE" button under the user's name and password boxes. We may have a section of "forget user's name?" or "forget password?". Since our Bookie targeted SJSU students, so if we can use "Connecting to SJSU" or "SJSU Single Sign-in" which we use to access our Canvas, home page would forward the user to the Sign-in page, and our login page would be replaced as the SJSU Sign in page.

1.3 Register page

This page is for new customers who either come to buy or sell textbooks. This page might contain the user's name box, password box, and "CANCEL" and "CONTINUE" buttons. Similar to the login page, if we can use "SJSU Sign-in" system, this page would be the student school sign-in page.

1.4 Profile editing page

In this page, customers can edit or change their profile. We will have change user's name, change passwords, cancel, and continue sections. We may have a section for users to edit their book posts, such as descriptions and pictures.

1.5 Search result page

After users type in the book's title and course name, and maybe select the price filter, the user would be led to a search result page, which lists all the related books to the customer. This page may contain the main column in the center of the page and a background picture. In the main column, there are several sections, which contains different seller's selling information. For example, a user is looking for *Database Systems The Complete Book second edition*, CS 157A, the result page would display the seller's registered name, book title, course, author, price, condition description, and the book's picture.

1.6 Post Creation page

Customers post their used books with the required information on the post page. In this page, we will have the main section for users to type in the book's title, course, author, price, condition description, and upload a picture of the book. We also have a submit button at the end of the page.

1.7 Post and comments page

After a user selects one post in the result page, and he or she goes into the post and comments page. In this page, the selected post was displayed with the book's picture and information. In the top of this page, there would be a toolbar, and under the toolbar on the left hand, there is a picture of the book. Right next to the picture is the information about the details of the book. In the bottom of this post page, there is a collapsed comment section. When a user clicks on the "show comments" button, the comments list would be expanded, and the user would be able to see all the comments list under the posting page.

1.8 Reset page

Users can reset their passwords in this reset page. There would be a box to ask user to type the new passwords and a confirm box to confirm the change.

1.9 Forget password page
If a user forgets his or her password, he or she will be led to the forget password page either to find the old password back by using register email or reset the password.

1.10 History page
In the history page, user can see his or her posts and purchases history in the history page. This page contains a list of post.

1.11 Saved post page
A user can save his or her favorite post page in the post page. It contains the picture and information of the book.

1.12 Friends page
In the friends page, users can find their friends list here. Also, users can check messages sent between them and their friends by clicking the friend request button, which would lead the user to the friends request page. There are two buttons which are “Friends” and “Friend Request”. By clicking “Friends” button, the user can see his or her friends list. By clicking “Friend Request” button, the user will be sent to the friends request page and see the messages between friends.

1.13 Friends request page
In the friends request page, the user can see the friend’s name and the message between friends. The left column of the page will show the list of friends, and the right column of the page will show the list of the messages of friends.

2. Security

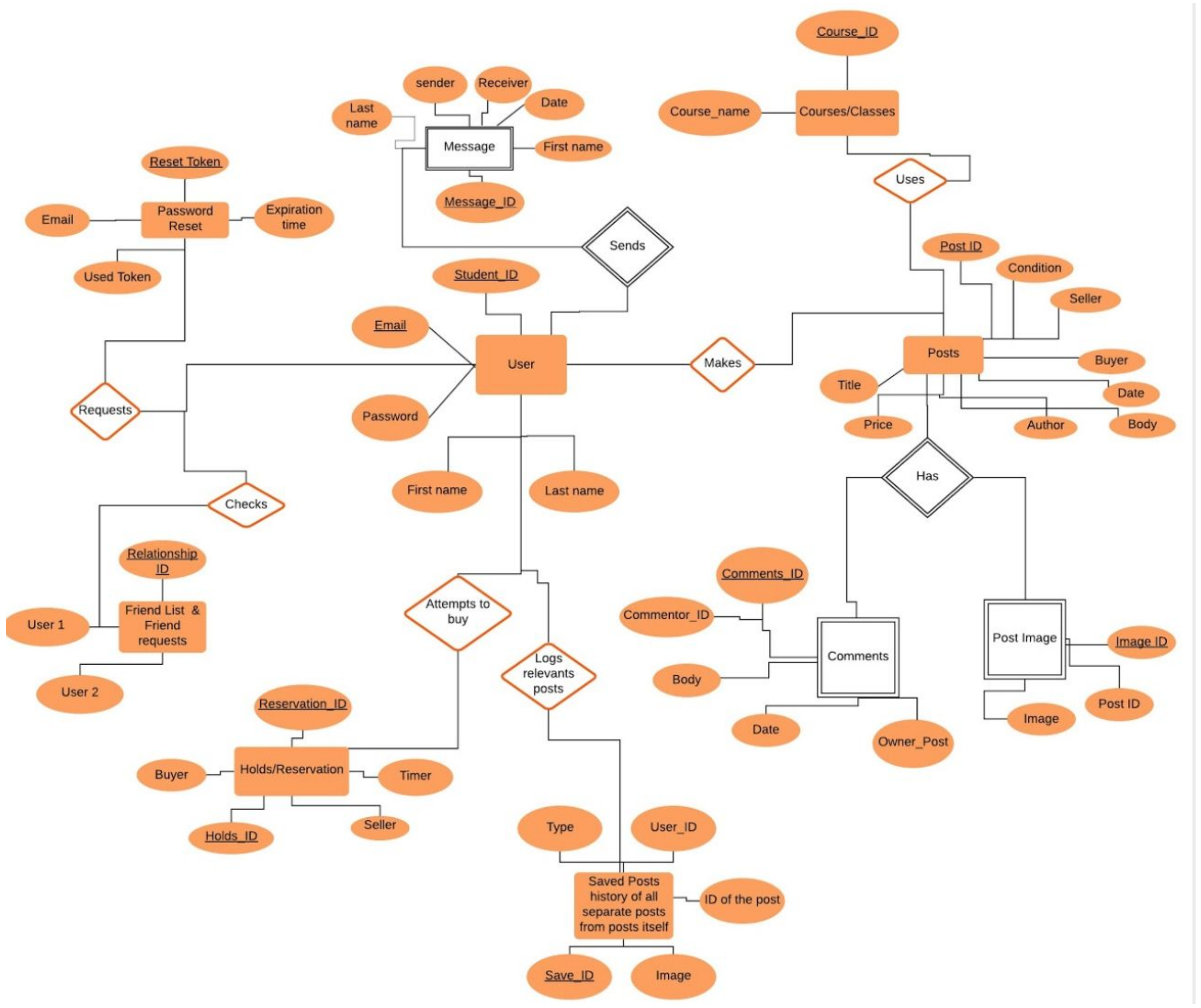
We hosted SJSU Bookie’s NodeJS Web Server on Heroku. Each customer has an isolated user account, which can not be edited by other users. Users information, especially for the transactions, would be securely stored in our server. Bookie web application makes it safe to make transactions online when they sell or buy books. Also, one customer cannot access to other customers’ posts. To protect the security and privacy of all the customers, each customer can only manage his or her own posts. Our data has durability in our relational database management system.

3. Access Control

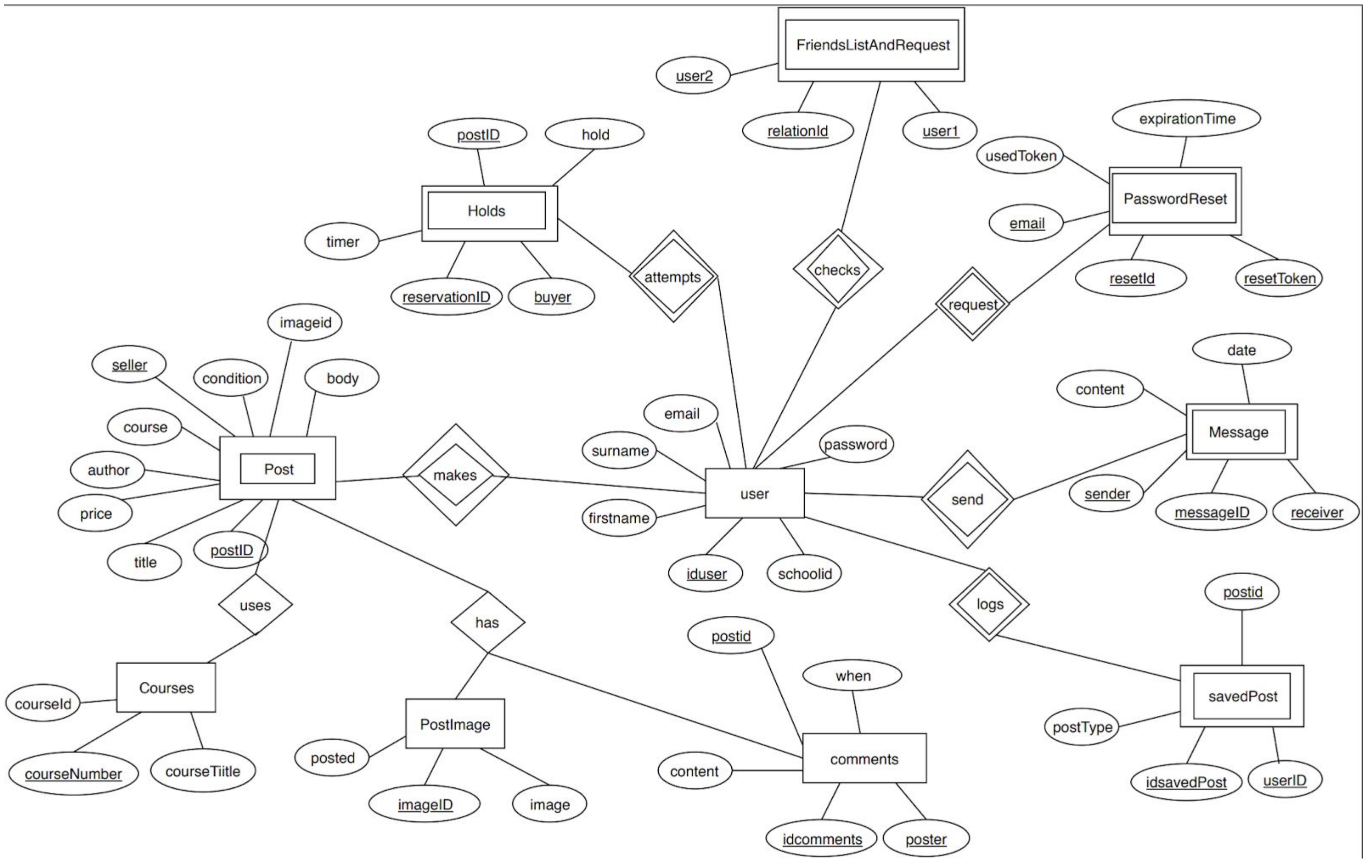
We maintain isolation between users. In other words, each user can only edit his or her own account which separates from other users’. Without a registered account, a customer can not purchase a book.

Project Design

Previous ERD:



Updated ERD Model



DB Design Schema, non-trivial FDs and Table Screenshots

Schema

- Users(<u>iduser</u>, schoolid, firstname, surname, email, password)
- Message(<u>iduser</u>, <u>MessageID</u>, <u>receiver</u>, <u>sender</u>, content, date)
- Holds(<u>iduser</u>, <u>reservationID</u>, <u>buyer</u>, <u>postID,</u> hold, timer)
- SavedPost(<u>iduser</u>, i<u>dSavePost</u>, <u>UserID</u>, postType, <u>postid</u>)
- FriendListAndRequest(<u>iduser</u>, <u>relationshipId</u>, <u>user1</u>, <u>user2</u>)
- PasswordReset(<u>iduser</u>, <u>resetId</u>, resetToken, <u>email</u>, expirationTime, usedToken)
- Post(<u>PostID</u>, title, <u>author</u>, seller, course, condition, body, date, imageid, price)
- Comments(id<u>comments</u>, poster, content, <u>postid</u>, when)
- Courses(<u>courseNumber</u>, courseId, courseTitle)
- PostImage(<u>ImageID</u>, posted, image)
- Relations:
- has(<u>PostID</u>, title, <u>author</u>, seller, course, condition, body, date, imageid, price, <u>ImageID</u>, posted, image, id<u>comments</u>, poster, content, <u>postid</u>, when)
- uses(<u>PostID</u>, title, <u>author</u>, seller, course, condition, body, date, imageid, price, <u>courseNumber</u>, courseId, courseTitle)

Non-trivial FDs

- iduser -> schoolid, firstname, surname, content, date
- MessageID -> receiver, sender, content, date
- reservationID -> hold, timer, buyer -> postID
- idSavePost, UserID, postid -> postType
- Relationship -> user1, user2
- resetId, email -> resetToken, expirationTime, usedToken
- PostID, author -> title, seller, course, condition, body, date, imageid, price
- Idcomments, postid -> poster, content, when
- imageID -> posted, image

Explanations for E/R Diagram

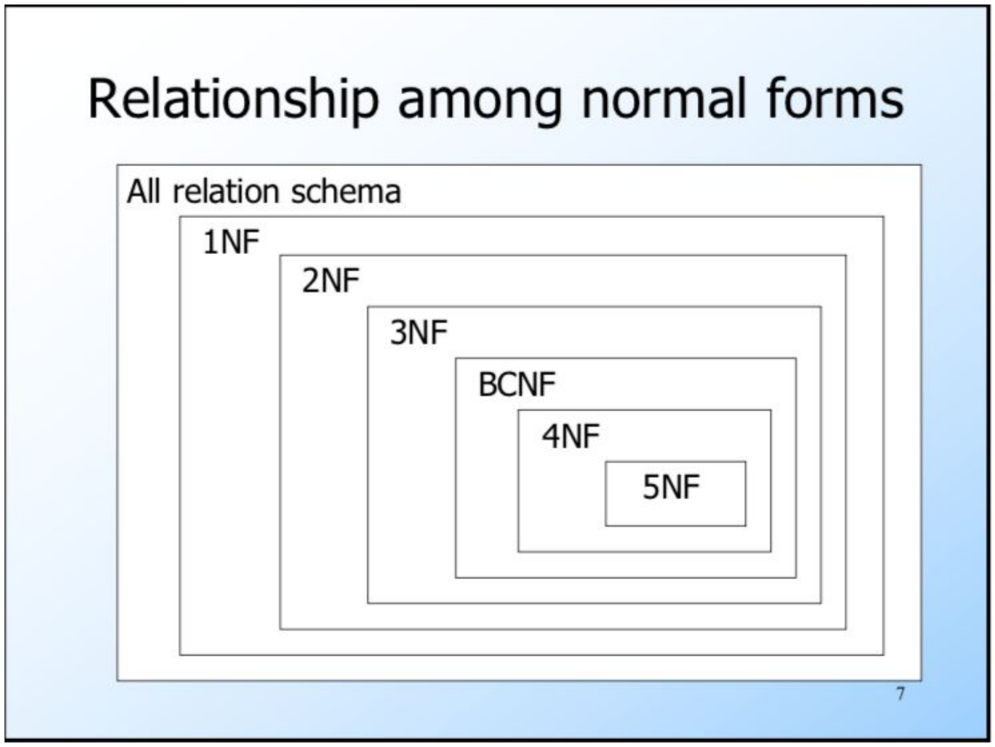
- **Tables**
  - User
    - The users who are using our SJSU Bookie. The Unique keys are the user's email and student ID.
  - Password Reset
    - Unique key is the reset token where this allows the user to be able to reset their password in case they forgot it. Token is used as validation and identification.
  - Post
    - Posted by a user, this entity has details of the book they are trying to sell. Another user can put a hold, view or comment on it, but only the original poster can edit/delete the posting. These are recognized by the post id.
  - Saved Posts
    - This is a log/list of posts that is relevant to a user. These posts can include posts that you've made, posts that you've commented on, or posts that you favorited. These are identified by the post_id and saved_id
  - Messages
    - Users can send messages to other users, but the constraint is the person must be a user, then he or she can send messages to other users. The message ID number is the unique key.
  - FriendList/Friend Friend Requests
    - A user is able to add another user as their friend on SJSUBookie in order to maintain communication or be able to communicate with one another. A friend request is considered accepted when both users recognize that the other is the friend on the database. These are uniquely identified by the relationship ID.

- Holds/Reservations
  - A hold is basically a contract stating that one user is interested in another user's post. This is temporary let other users know that the book might be bought. Identified by Hold/Reservation_ID.
- Comments
  - Users can leave comments under a post. The unique key of comments is comment ID, which we use to identify the comments.
- Post Image
  - It will be a table of images the user uploaded. The users would upload images when they are creating a selling post. The unique key is the image ID.
- Course/Classes
  - Course is to help posts identify which class/course the book belongs to. Identified by the course_ID like (CS157A, CMPE165… etc)

- ENTITY RELATIONSHIP EXPLANATION
  - **FORMAT: ENTITY_1 RELATIONSHIP ENTITY_2**
  - User requests Password Reset
    - A user will have the option to reset their password if they've forgotten it.
  - User Checks User Friend request
    - A user has the option to see all pending friend requests and accept friend requests from other users.
  - User Attempts to buy holds/reservation
    - A user must be able to put a reservation on hold if they are interested in the listing but unable to buy immediately. This will prevent another user from buying the listed book.
  - User Logs Saved Posts
    - Any post the user has saved, posted, or removed will be logged, providing the option to view at a later time.
  - User Makes Posts
    - A user needs to be able to make a post to sell a textbook.
  - Posts have comments
    - Every available post can have comments regarding the book or the seller.
  - Posts have post image
    - A post should have a corresponding image matching the textbook cover.
  - Posts use course/classes
    - Each book posting can have relevant classes associated with to improve search results.
  - User sends messagesso
    - Buyers and sellers can communicate by sending messages to increase transaction security.

Normalization process - BCNF

Normalization overview:



Source: lecture 15 slides

Boyce-Codd Normal Form holds for :
A relation R is in BCNF if and only if for every non-trivial FD A1, A2,...An -> B for R, {A1, A2,...An} is a superkey for R.

Bookie non-trivial FDs:

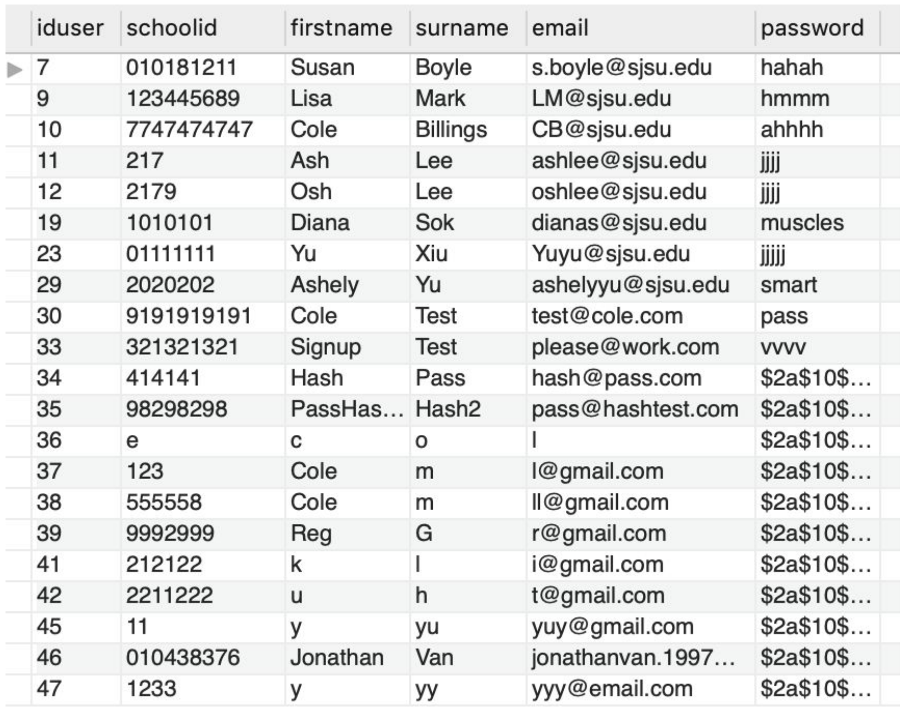
- iduser -> schoolid, firstname, surname, content, date
  - For the user table, iduser is the only key, so this holds for BCNF
- MessageID -> receiver, sender, content, date
  - For message table, messageID can determines all the other attributes, so this holds for BCNF
- reservationID, buyer -> postID, hold, timer
  - reservationID -> hold, timer violates BCNF
  - buyer -> postID
  - Decomposition:
  - R1 = reservationID+ = {reservationID, hold, timer} valid in BCNF
  - R2 = R - R1 + {reservationID} = {reservationID, buyer, postID}
  - buyer -> postID

    - R21 = buyer+ = {buyer, postID} valid in BCNF
    - R22 = {reservationID, buyer} valid in BCNF
- idSavePost, UserID, postid -> postType
    - idSavePost, UserID, postid is the super key, so it holds BCNF
- Relationship -> user1, user2
    - Relationship is the superkey, holds for BCNF
- resetId, email -> resetToken, expirationTime, usedToken
    - restetId, email is the superkey, so it hold BCNF
- PostID, author -> title, seller, course, condition, body, date, imageid, price
    - Left hand is the super key, holds for BCNF
- Idcomments, postid -> poster, content, when
    - Left hand is the super key, holds for BCNF
- imageID -> posted, image
    - imageID is the only key, so it holds BCNF

Tables

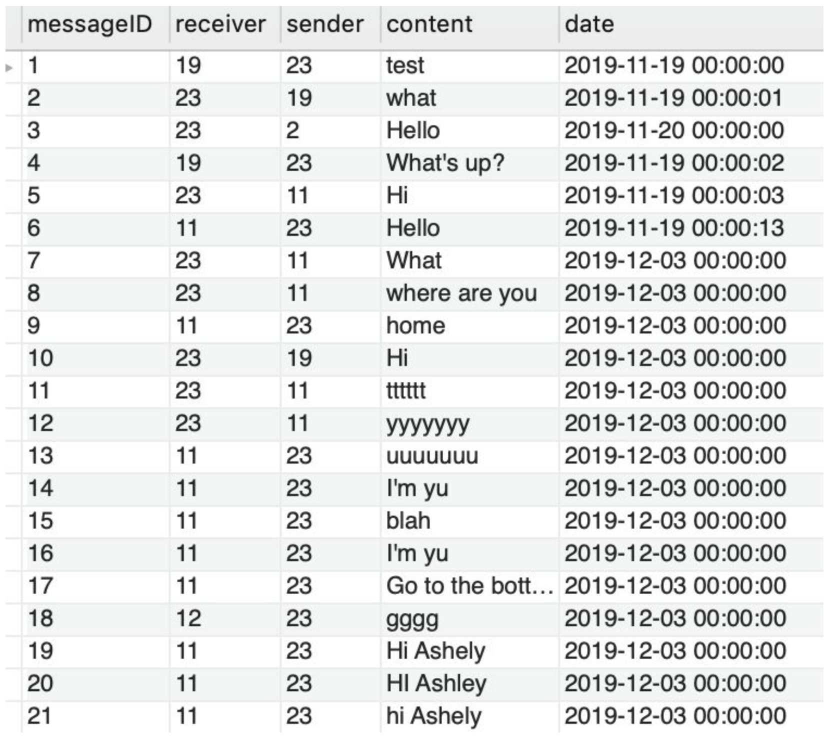
With at least 15 tuples

- user:



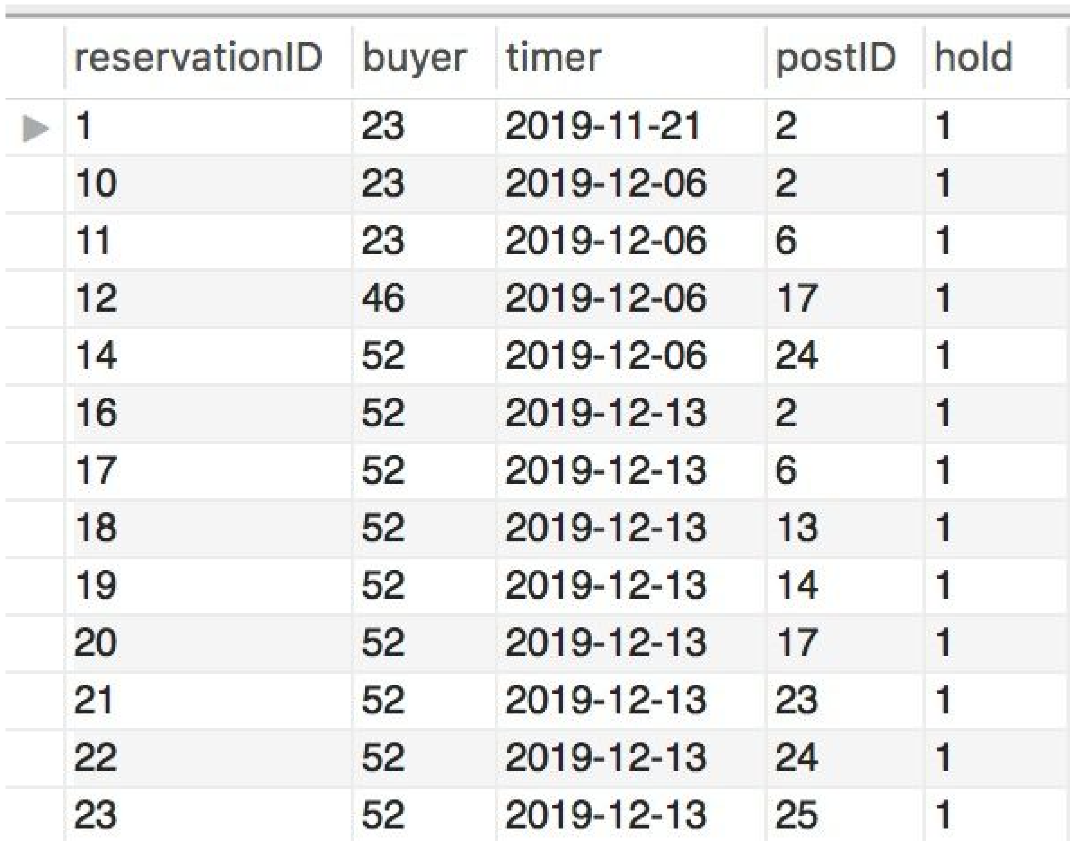
iduser	schoolid	firstname	surname	email	password
7	010181211	Susan	Boyle	s.boyle@sjsu.edu	hahah
9	123445689	Lisa	Mark	LM@sjsu.edu	hmmm
10	7747474747	Cole	Billings	CB@sjsu.edu	ahhhh
11	217	Ash	Lee	ashlee@sjsu.edu	jjjj
12	2179	Osh	Lee	oshlee@sjsu.edu	jjjj
19	1010101	Diana	Sok	dianas@sjsu.edu	muscles
23	01111111	Yu	Xiu	Yuyu@sjsu.edu	jjjjj
29	2020202	Ashely	Yu	ashelyyu@sjsu.edu	smart
30	9191919191	Cole	Test	test@cole.com	pass
33	321321321	Signup	Test	please@work.com	vvvv
34	414141	Hash	Pass	hash@pass.com	$2a$10$…
35	98298298	PassHas…	Hash2	pass@hashtest.com	$2a$10$…
36	e	c	o	l	$2a$10$…
37	123	Cole	m	l@gmail.com	$2a$10$…
38	555558	Cole	m	ll@gmail.com	$2a$10$…
39	9992999	Reg	G	r@gmail.com	$2a$10$…
41	212122	k	l	i@gmail.com	$2a$10$…
42	2211222	u	h	t@gmail.com	$2a$10$…
45	11	y	yu	yuy@gmail.com	$2a$10$…
46	010438376	Jonathan	Van	jonathanvan.1997…	$2a$10$…
47	1233	y	yy	yyy@email.com	$2a$10$…

- Message:



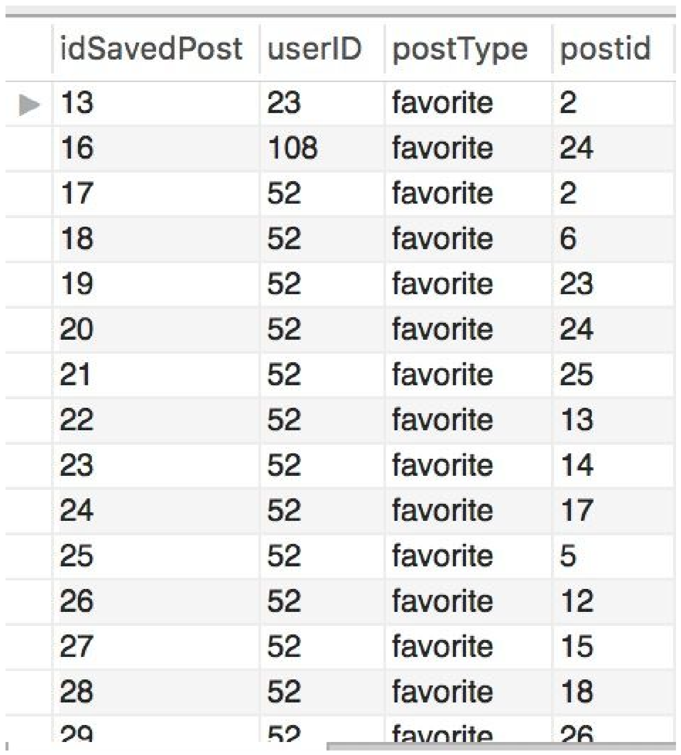
messageID	receiver	sender	content	date
1	19	23	test	2019-11-19 00:00:00
2	23	19	what	2019-11-19 00:00:01
3	23	2	Hello	2019-11-20 00:00:00
4	19	23	What's up?	2019-11-19 00:00:02
5	23	11	Hi	2019-11-19 00:00:03
6	11	23	Hello	2019-11-19 00:00:13
7	23	11	What	2019-12-03 00:00:00
8	23	11	where are you	2019-12-03 00:00:00
9	11	23	home	2019-12-03 00:00:00
10	23	19	Hi	2019-12-03 00:00:00
11	23	11	tttttt	2019-12-03 00:00:00
12	23	11	yyyyyyy	2019-12-03 00:00:00
13	11	23	uuuuuuu	2019-12-03 00:00:00
14	11	23	I'm yu	2019-12-03 00:00:00
15	11	23	blah	2019-12-03 00:00:00
16	11	23	I'm yu	2019-12-03 00:00:00
17	11	23	Go to the bott…	2019-12-03 00:00:00
18	12	23	gggg	2019-12-03 00:00:00
19	11	23	Hi Ashely	2019-12-03 00:00:00
20	11	23	HI Ashley	2019-12-03 00:00:00
21	11	23	hi Ashely	2019-12-03 00:00:00

- Holds



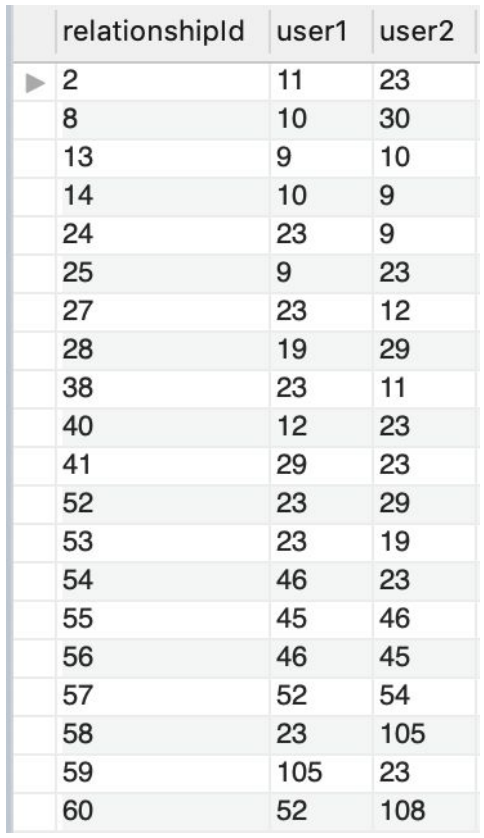
reservationID	buyer	timer	postID	hold
1	23	2019-11-21	2	1
10	23	2019-12-06	2	1
11	23	2019-12-06	6	1
12	46	2019-12-06	17	1
14	52	2019-12-06	24	1
16	52	2019-12-13	2	1
17	52	2019-12-13	6	1
18	52	2019-12-13	13	1
19	52	2019-12-13	14	1
20	52	2019-12-13	17	1
21	52	2019-12-13	23	1
22	52	2019-12-13	24	1
23	52	2019-12-13	25	1

- SavedPost



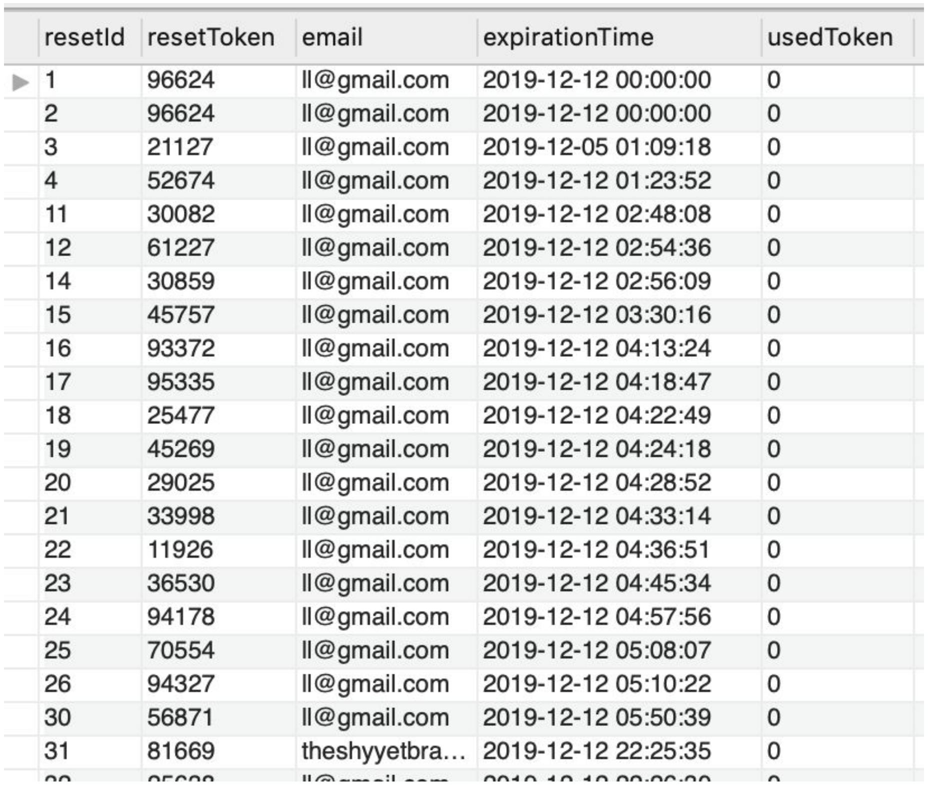
idSavedPost	userID	postType	postid
13	23	favorite	2
16	108	favorite	24
17	52	favorite	2
18	52	favorite	6
19	52	favorite	23
20	52	favorite	24
21	52	favorite	25
22	52	favorite	13
23	52	favorite	14
24	52	favorite	17
25	52	favorite	5
26	52	favorite	12
27	52	favorite	15
28	52	favorite	18
29	52	favorite	26

- FriendListAndRequest



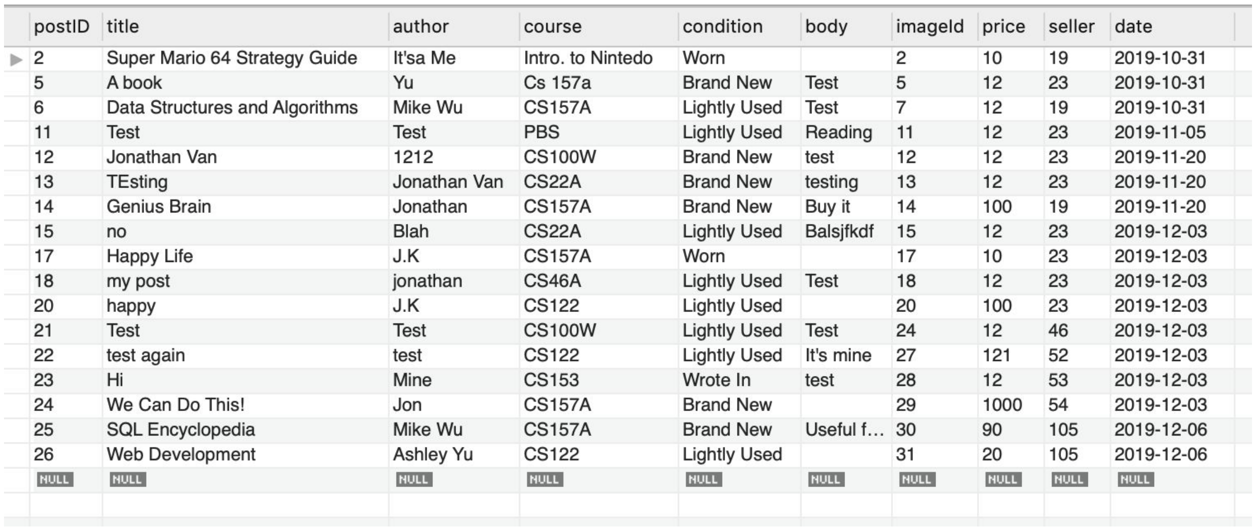
relationshipId	user1	user2
2	11	23
8	10	30
13	9	10
14	10	9
24	23	9
25	9	23
27	23	12
28	19	29
38	23	11
40	12	23
41	29	23
52	23	29
53	23	19
54	46	23
55	45	46
56	46	45
57	52	54
58	23	105
59	105	23
60	52	108

- PasswordReset



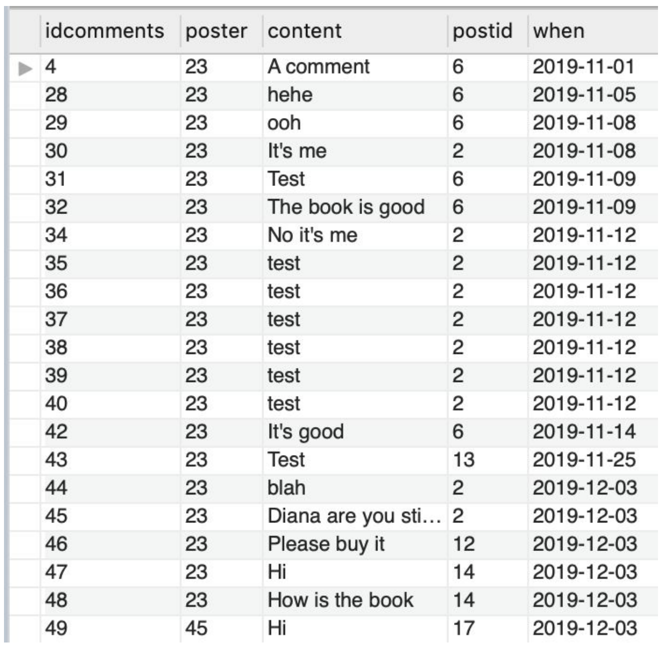
resetId	resetToken	email	expirationTime	usedToken
1	96624	ll@gmail.com	2019-12-12 00:00:00	0
2	96624	ll@gmail.com	2019-12-12 00:00:00	0
3	21127	ll@gmail.com	2019-12-05 01:09:18	0
4	52674	ll@gmail.com	2019-12-12 01:23:52	0
11	30082	ll@gmail.com	2019-12-12 02:48:08	0
12	61227	ll@gmail.com	2019-12-12 02:54:36	0
14	30859	ll@gmail.com	2019-12-12 02:56:09	0
15	45757	ll@gmail.com	2019-12-12 03:30:16	0
16	93372	ll@gmail.com	2019-12-12 04:13:24	0
17	95335	ll@gmail.com	2019-12-12 04:18:47	0
18	25477	ll@gmail.com	2019-12-12 04:22:49	0
19	45269	ll@gmail.com	2019-12-12 04:24:18	0
20	29025	ll@gmail.com	2019-12-12 04:28:52	0
21	33998	ll@gmail.com	2019-12-12 04:33:14	0
22	11926	ll@gmail.com	2019-12-12 04:36:51	0
23	36530	ll@gmail.com	2019-12-12 04:45:34	0
24	94178	ll@gmail.com	2019-12-12 04:57:56	0
25	70554	ll@gmail.com	2019-12-12 05:08:07	0
26	94327	ll@gmail.com	2019-12-12 05:10:22	0
30	56871	ll@gmail.com	2019-12-12 05:50:39	0
31	81669	theshyyetbra…	2019-12-12 22:25:35	0

- Posts



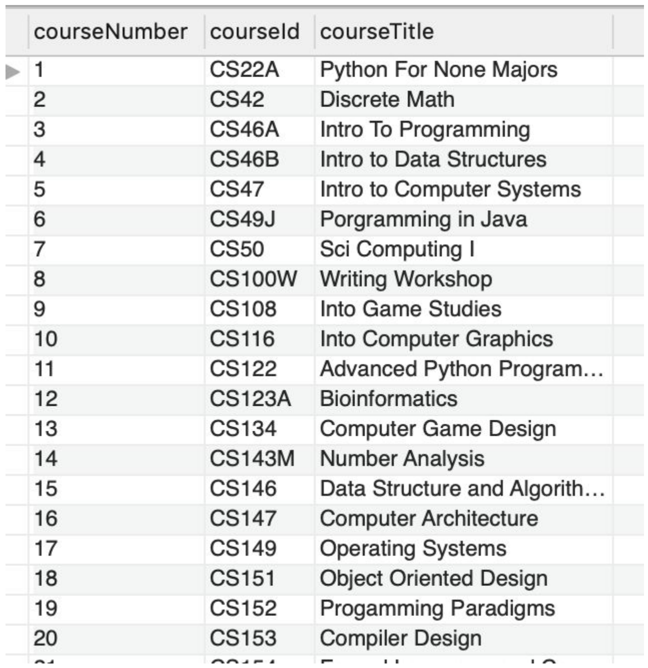
postID	title	author	course	condition	body	imageId	price	seller	date
2	Super Mario 64 Strategy Guide	It'sa Me	Intro. to Nintedo	Worn		2	10	19	2019-10-31
5	A book	Yu	Cs 157a	Brand New	Test	5	12	23	2019-10-31
6	Data Structures and Algorithms	Mike Wu	CS157A	Lightly Used	Test	7	12	19	2019-10-31
11	Test	Test	PBS	Lightly Used	Reading	11	12	23	2019-11-05
12	Jonathan Van	1212	CS100W	Brand New	test	12	12	23	2019-11-20
13	TEsting	Jonathan Van	CS22A	Brand New	testing	13	12	23	2019-11-20
14	Genius Brain	Jonathan	CS157A	Brand New	Buy it	14	100	19	2019-11-20
15	no	Blah	CS22A	Lightly Used	Balsjfkdf	15	12	23	2019-12-03
17	Happy Life	J.K	CS157A	Worn		17	10	23	2019-12-03
18	my post	jonathan	CS46A	Lightly Used	Test	18	12	23	2019-12-03
20	happy	J.K	CS122	Lightly Used		20	100	23	2019-12-03
21	Test	Test	CS100W	Lightly Used	Test	24	12	46	2019-12-03
22	test again	test	CS122	Lightly Used	It's mine	27	121	52	2019-12-03
23	Hi	Mine	CS153	Wrote In	test	28	12	53	2019-12-03
24	We Can Do This!	Jon	CS157A	Brand New		29	1000	54	2019-12-03
25	SQL Encyclopedia	Mike Wu	CS157A	Brand New	Useful f…	30	90	105	2019-12-06
26	Web Development	Ashley Yu	CS122	Lightly Used		31	20	105	2019-12-06
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- Comments



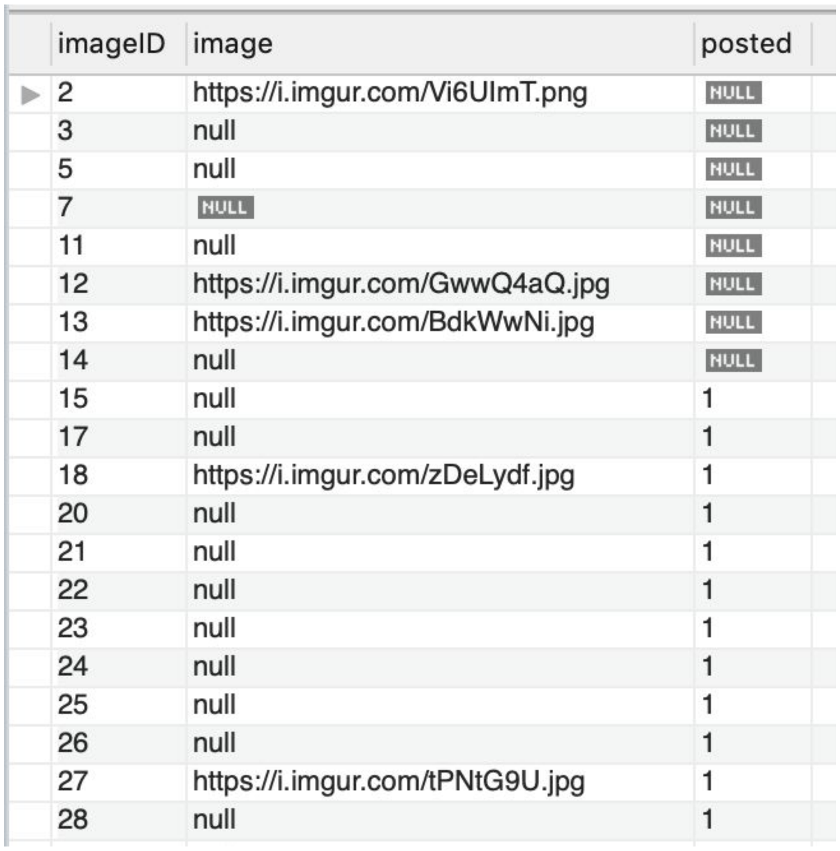
idcomments	poster	content	postid	when
4	23	A comment	6	2019-11-01
28	23	hehe	6	2019-11-05
29	23	ooh	6	2019-11-08
30	23	It's me	2	2019-11-08
31	23	Test	6	2019-11-09
32	23	The book is good	6	2019-11-09
34	23	No it's me	2	2019-11-12
35	23	test	2	2019-11-12
36	23	test	2	2019-11-12
37	23	test	2	2019-11-12
38	23	test	2	2019-11-12
39	23	test	2	2019-11-12
40	23	test	2	2019-11-12
42	23	It's good	6	2019-11-14
43	23	Test	13	2019-11-25
44	23	blah	2	2019-12-03
45	23	Diana are you sti…	2	2019-12-03
46	23	Please buy it	12	2019-12-03
47	23	Hi	14	2019-12-03
48	23	How is the book	14	2019-12-03
49	45	Hi	17	2019-12-03

- Courses



courseNumber	courseId	courseTitle
1	CS22A	Python For None Majors
2	CS42	Discrete Math
3	CS46A	Intro To Programming
4	CS46B	Intro to Data Structures
5	CS47	Intro to Computer Systems
6	CS49J	Porgramming in Java
7	CS50	Sci Computing I
8	CS100W	Writing Workshop
9	CS108	Into Game Studies
10	CS116	Into Computer Graphics
11	CS122	Advanced Python Program…
12	CS123A	Bioinformatics
13	CS134	Computer Game Design
14	CS143M	Number Analysis
15	CS146	Data Structure and Algorith…
16	CS147	Computer Architecture
17	CS149	Operating Systems
18	CS151	Object Oriented Design
19	CS152	Progamming Paradigms
20	CS153	Compiler Design

- PostImage



imageID	image	posted
2	https://i.imgur.com/Vi6UImT.png	NULL
3	null	NULL
5	null	NULL
7	NULL	NULL
11	null	NULL
12	https://i.imgur.com/GwwQ4aQ.jpg	NULL
13	https://i.imgur.com/BdkWwNi.jpg	NULL
14	null	NULL
15	null	1
17	null	1
18	https://i.imgur.com/zDeLydf.jpg	1
20	null	1
21	null	1
22	null	1
23	null	1
24	null	1
25	null	1
26	null	1
27	https://i.imgur.com/tPNtG9U.jpg	1
28	null	1

Implementation

Functional Requirements Review

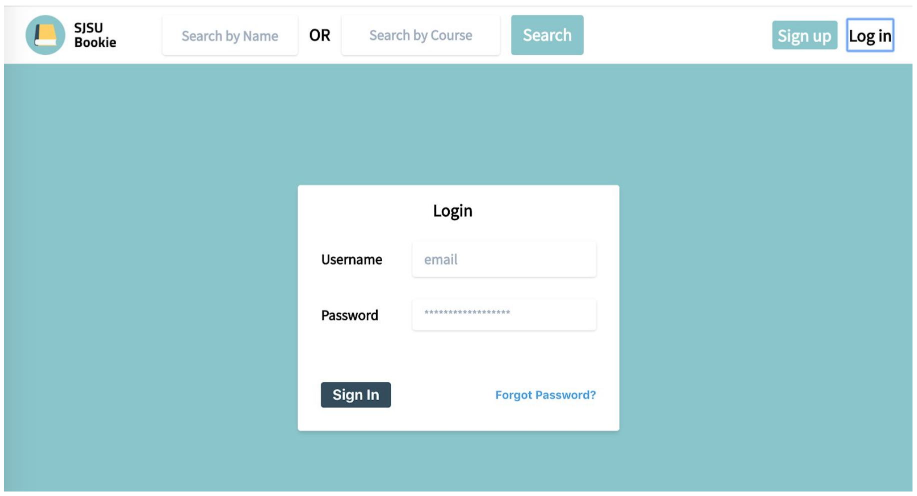
1. **Login**
2. **Sign up**
3. **Forget passwords**
4. **Reset passwords**
5. **Searching for a book**
6. **Create a post**
7. **Message and make friends**
8. **Profile page**
9. **Friends list**

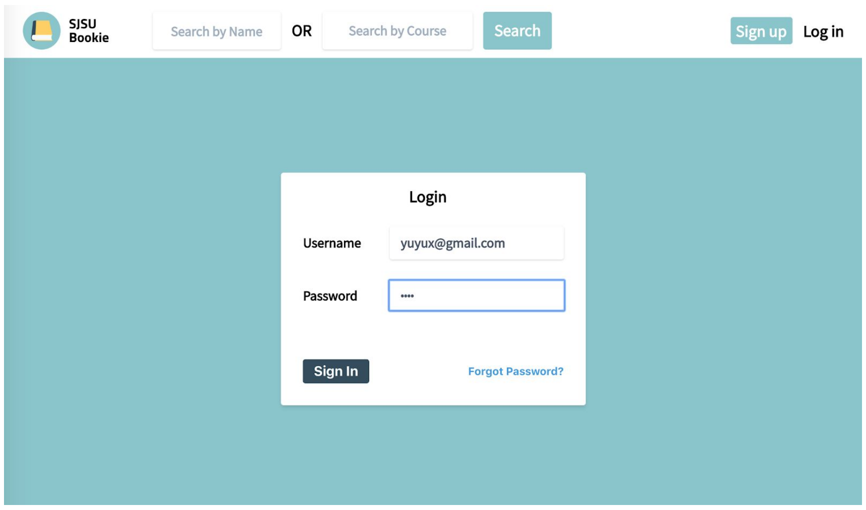
Implementation explanation with screenshots

We implemented our DB design based on our functional requirements. The following are detailed explanation of our DB application system design with screenshots:

1. **Login**
   1.1. SJSU Bookie has a login feature in order to grant User controls/commands to the user.
   1.2. Login system should grant access when the user enters correct credentials
   1.3. Login system will prevent access to incorrect credentials.

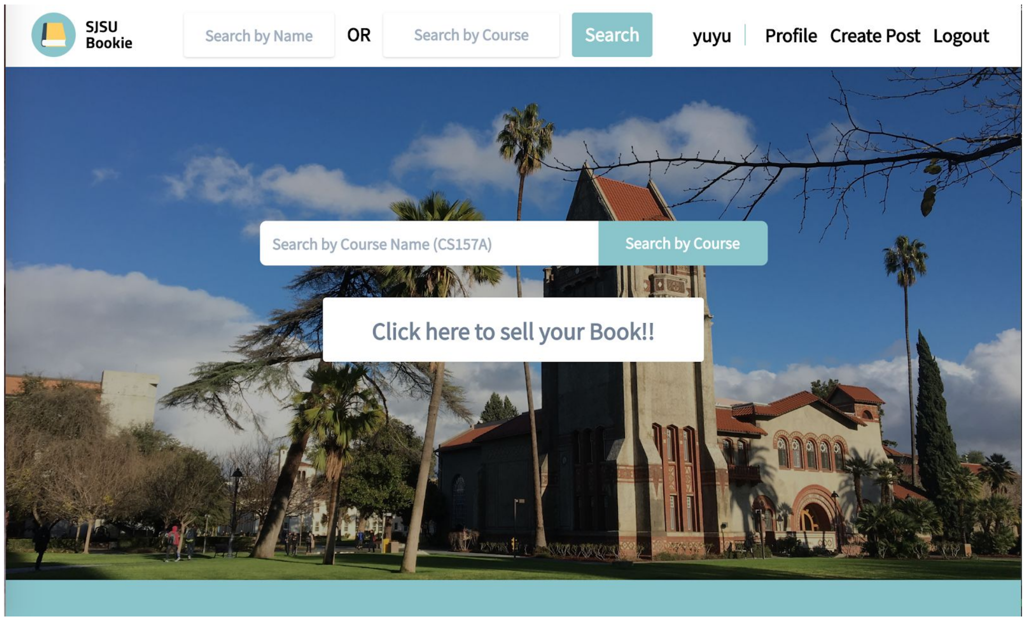
The login page:



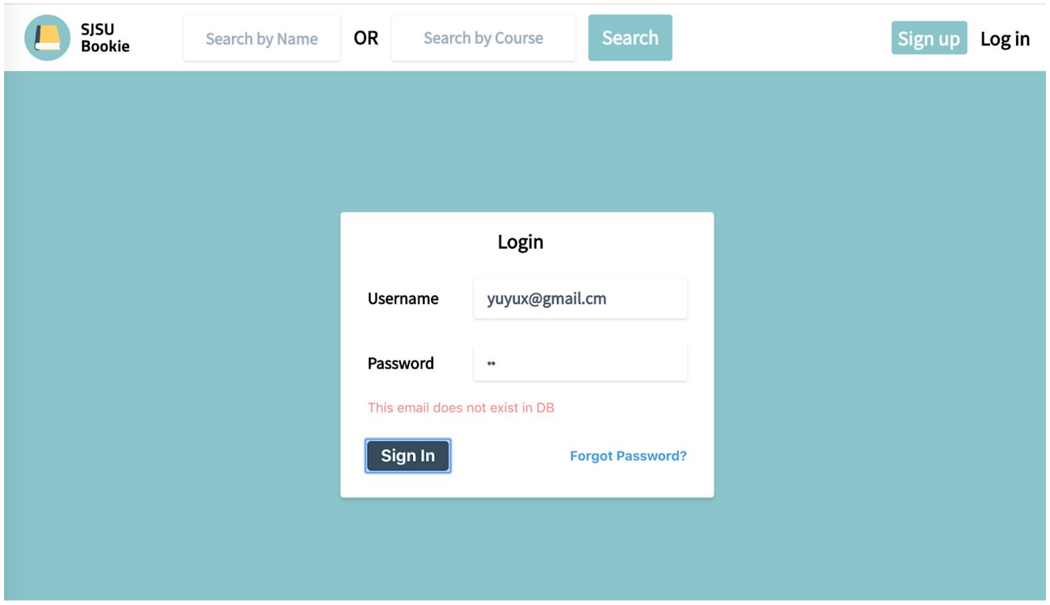


Username: yuyux@gmail.com

When clicking "Sign in", the user's name is on the top bar:



If the user typed a wrong username, Bookie reminds the user "This email does not exist in DB ":

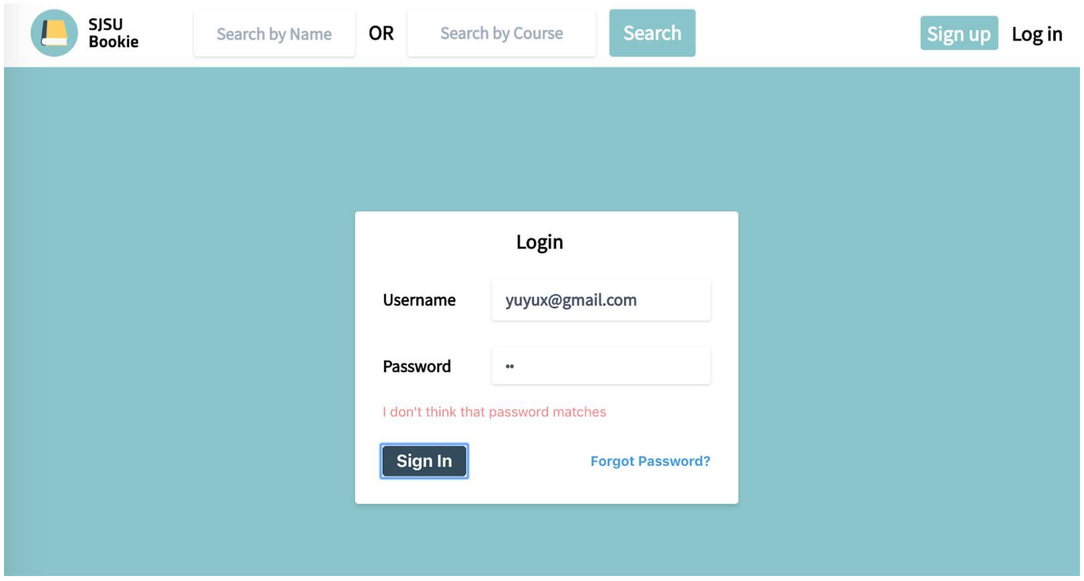


Username: yuyux@gmail.cm

Password: ••

This email does not exist in DB

If the user typed a wrong username, Bookie reminds the user "I don't think the password matches ":


SJSU Bookie

Search by Name OR Search by Course Search

Sign up Log in

Login

Username: yuyux@gmail.com

Password: ••

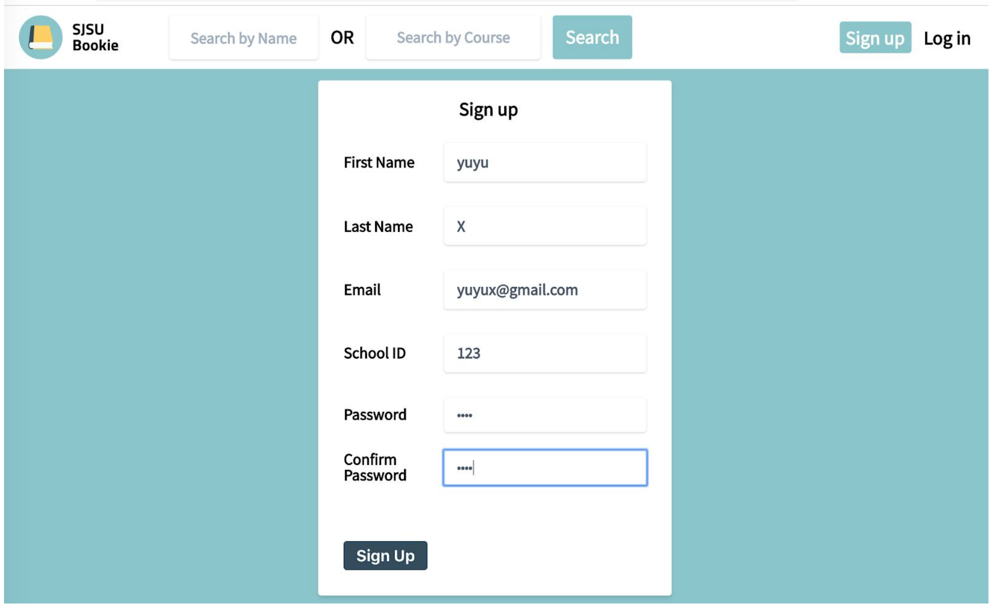
I don't think that password matches

Sign In

Forgot Password?

2. Sign Up

2.1. SJSU Bookie will have a registration system to allow new users to create accounts.


SJSU Bookie

Search by Name OR Search by Course Search

Sign up Log in

Sign up

First Name: yuyu

Last Name: X

Email: yuyux@gmail.com

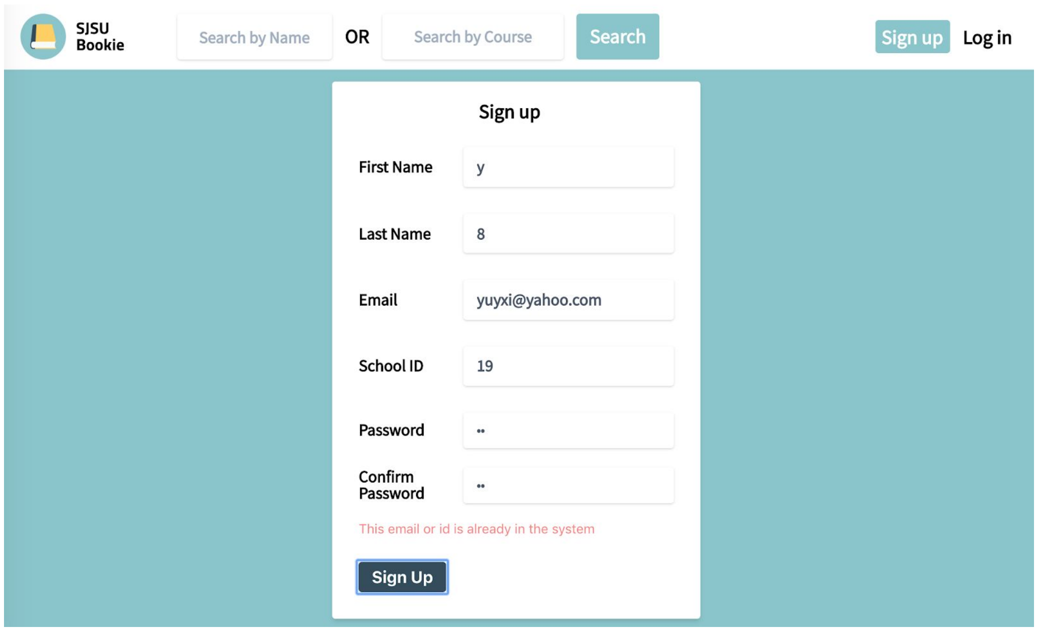
School ID: 123

Password: ••••

Confirm Password: ••••

Sign Up

If the email or id already exist, and system would reminds the user:



Sign up

First Name: y

Last Name: 8

Email: yuyxi@yahoo.com

School ID: 19

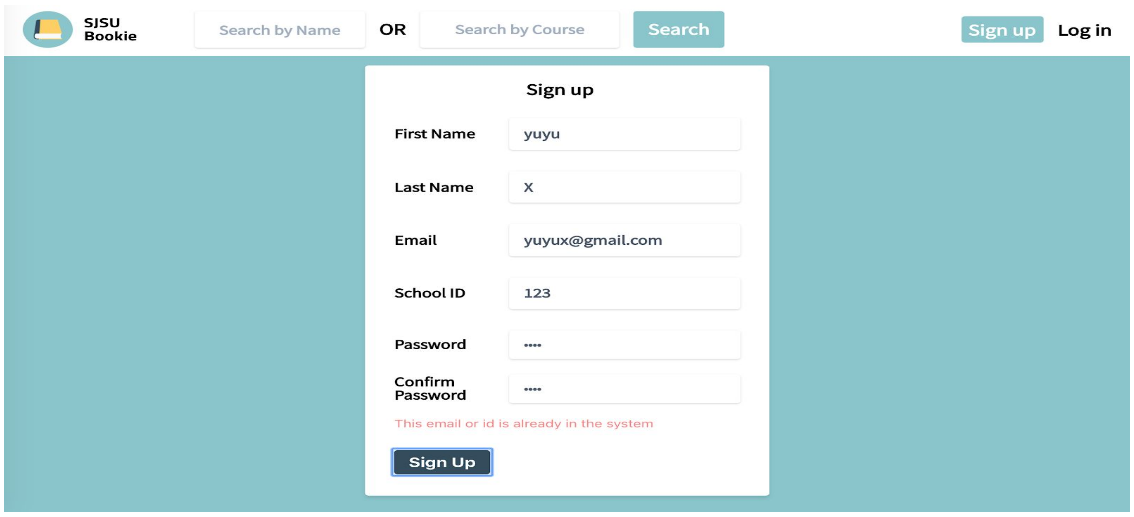
Password: ••

Confirm Password: ••

This email or id is already in the system

Sign Up

It checks the duplicates and send a warning message:



Sign up

First Name: yuyu

Last Name: X

Email: yuyux@gmail.com

School ID: 123

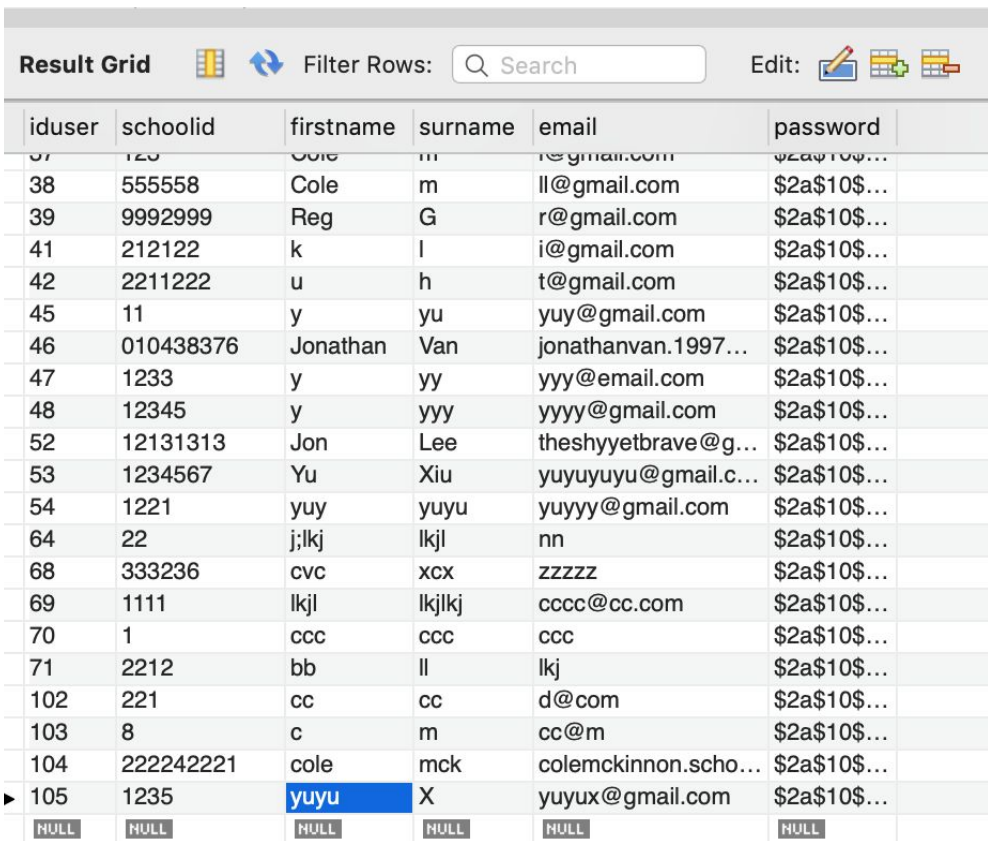
Password: ••••

Confirm Password: ••••

This email or id is already in the system

Sign Up

After changing the ID to 1235, the account created successfully and saved into workbench:

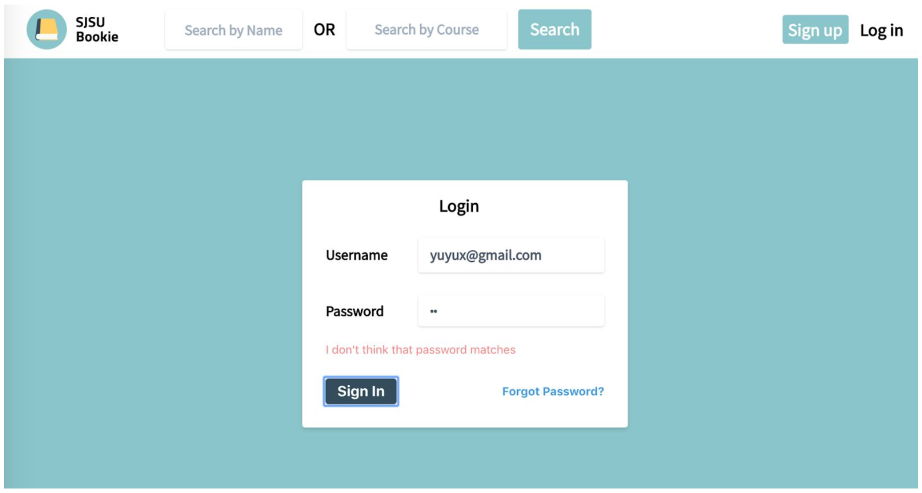

Result Grid Filter Rows: Search Edit:

iduser	schoolid	firstname	surname	email	password
37	123	Cole	m	r@gmail.com	$2a$10$...
38	555558	Cole	m	ll@gmail.com	$2a$10$...
39	9992999	Reg	G	r@gmail.com	$2a$10$...
41	212122	k	l	i@gmail.com	$2a$10$...
42	2211222	u	h	t@gmail.com	$2a$10$...
45	11	y	yu	yuy@gmail.com	$2a$10$...
46	010438376	Jonathan	Van	jonathanvan.1997...	$2a$10$...
47	1233	y	yy	yyy@email.com	$2a$10$...
48	12345	y	yyy	yyyy@gmail.com	$2a$10$...
52	12131313	Jon	Lee	theshyyetbrave@g...	$2a$10$...
53	1234567	Yu	Xiu	yuyuyuyu@gmail.c...	$2a$10$...
54	1221	yuy	yuyu	yuyyy@gmail.com	$2a$10$...
64	22	j;lkj	lkjl	nn	$2a$10$...
68	333236	cvc	xcx	zzzzz	$2a$10$...
69	1111	lkjl	lkjlkj	cccc@cc.com	$2a$10$...
70	1	ccc	ccc	ccc	$2a$10$...
71	2212	bb	ll	lkj	$2a$10$...
102	221	cc	cc	d@com	$2a$10$...
103	8	c	m	cc@m	$2a$10$...
104	222242221	cole	mck	colemckinnon.scho...	$2a$10$...
105	1235	yuyu	X	yuyux@gmail.com	$2a$10$...
NULL	NULL	NULL	NULL	NULL	NULL

3. Forget Password

3.1. SJSU Bookie will offer the option to reset a password if a user wants to change or has forgotten their password.

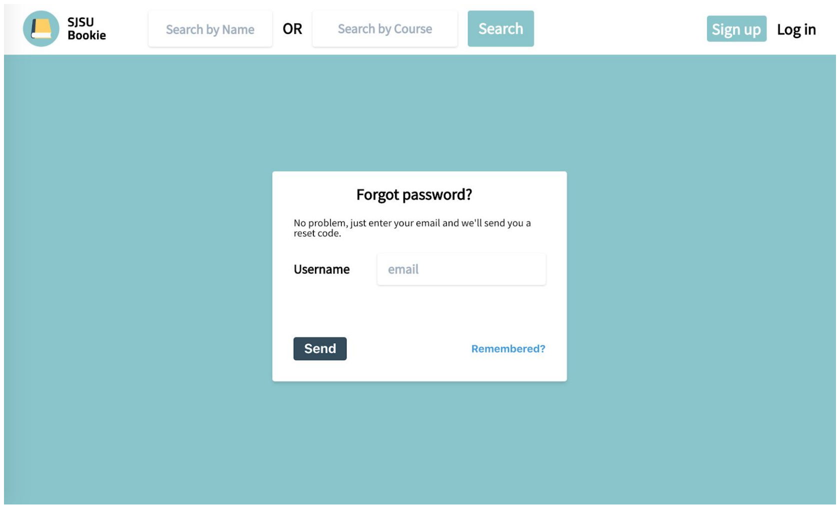
If a user forgets password, it can be found back:



yuyux@gmail.com

I don't think that password matches

Click on “Forget Password”:

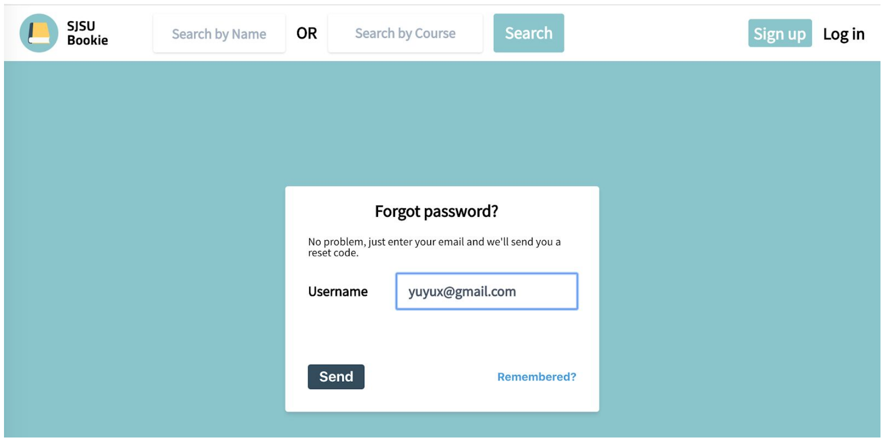


No problem, just enter your email and we'll send you a reset code.

4. Reset Password

4.1. SJSU Bookie will offer the user an option to reset the password if they fulfill the forget password requirements.

Type in username:

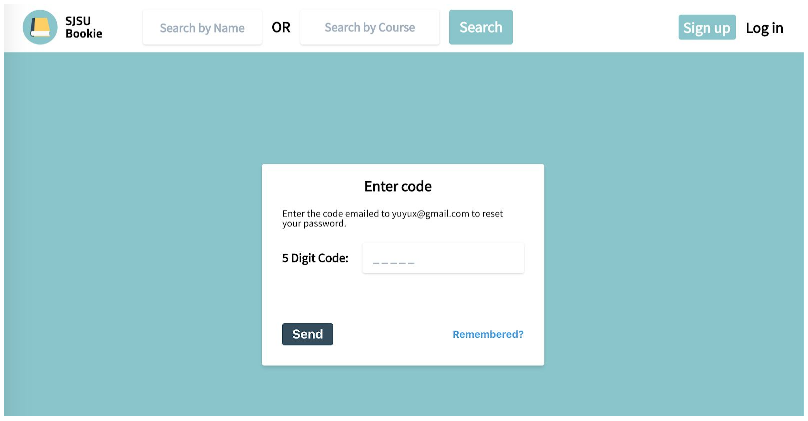


Forgot password?

No problem, just enter your email and we'll send you a reset code.

Username yuyux@gmail.com

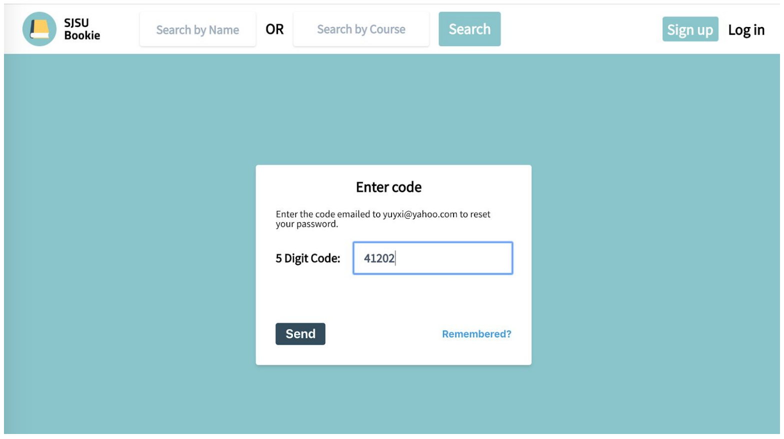
Click “Send”: A user must check their email for the 5 digit code



Enter code

Enter the code emailed to yuyux@gmail.com to reset your password.

5 Digit Code: _____



Enter code

Enter the code emailed to yuyxi@yahoo.com to reset your password.

5 Digit Code: 41202

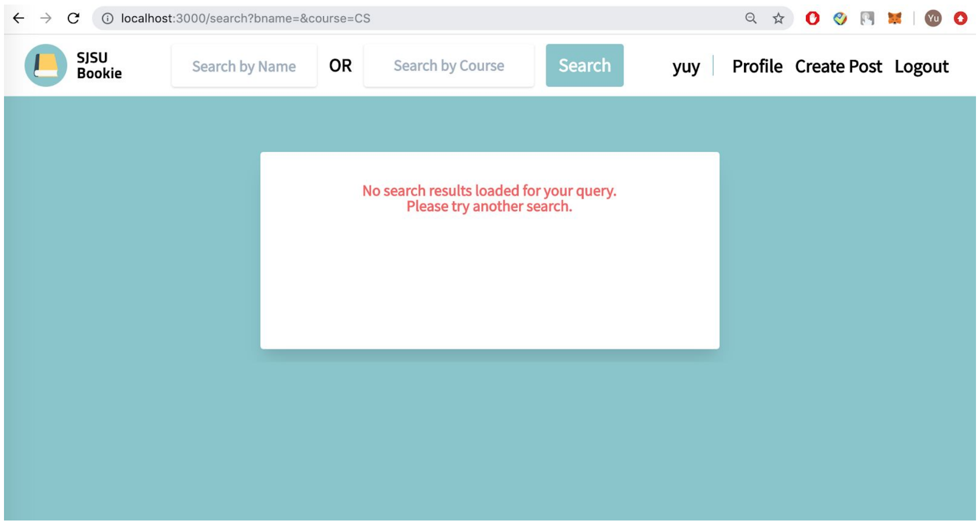
5. Searching

5.1. SJSU Bookie will have a search function that is available for users that are not logged in and users that are logged in.

5.2. Search can be done by inputting the name of the book or the course id.

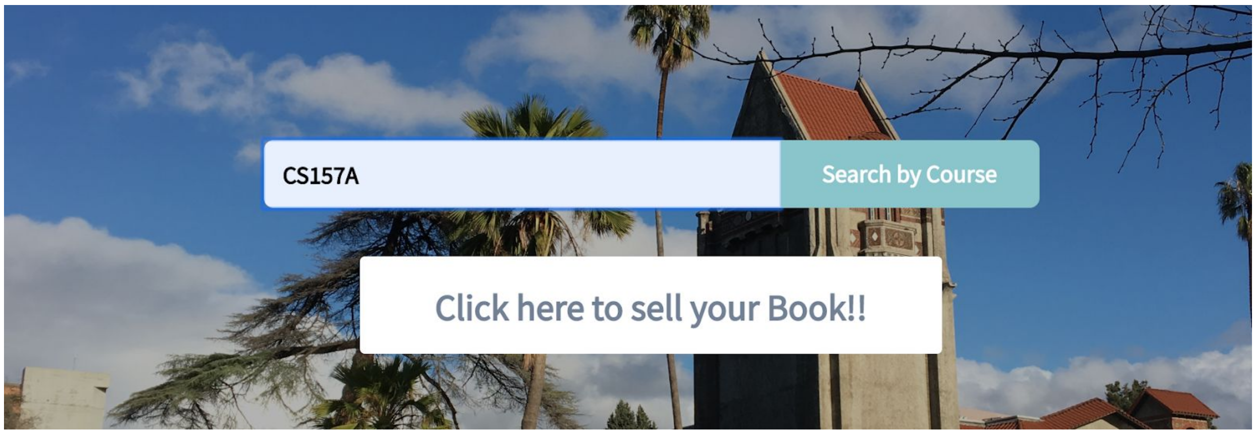
5.3. If there is time SJSU Bookie will allow filtering by price and dates.

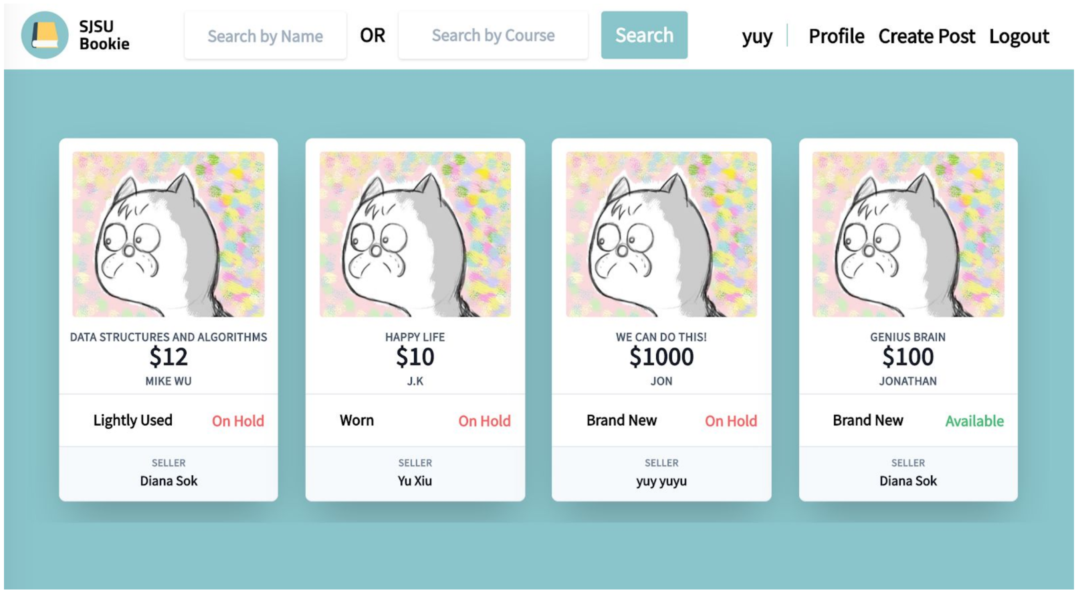
When searching non exist book:



No search results loaded for your query.
Please try another search.

When search for CS157A:



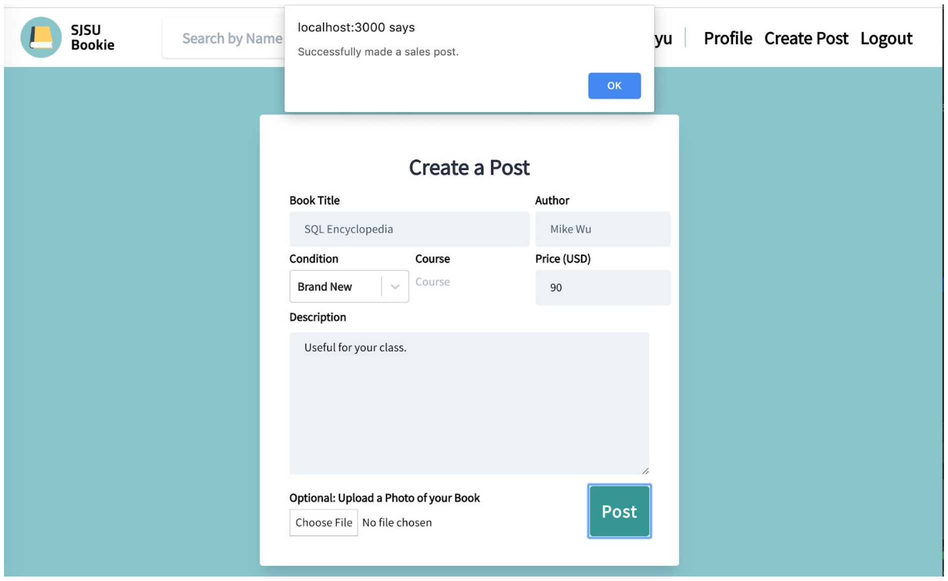


DATA STRUCTURES AND ALGORITHMS	HAPPY LIFE	WE CAN DO THIS!	GENIUS BRAIN
$12	$10	$1000	$100
MIKE WU	J.K	JON	JONATHAN
Lightly Used / On Hold	Worn / On Hold	Brand New / On Hold	Brand New / Available
SELLER Diana Sok	SELLER Yu Xiu	SELLER yuy yuyu	SELLER Diana Sok

6. **Posting**

6.1. SJSU bookie will allow users to post on its page that the user is selling a textbook. The user will be able to view their own post, edit it, and delete their posting. Any other user can view that page, but they cannot delete or edit it because it is not theirs. They can comment on the post if they want however. A post has the name of the book, the class it is for, and the price the seller is trying to sell it for.

Under yuyu user's account, create new post:



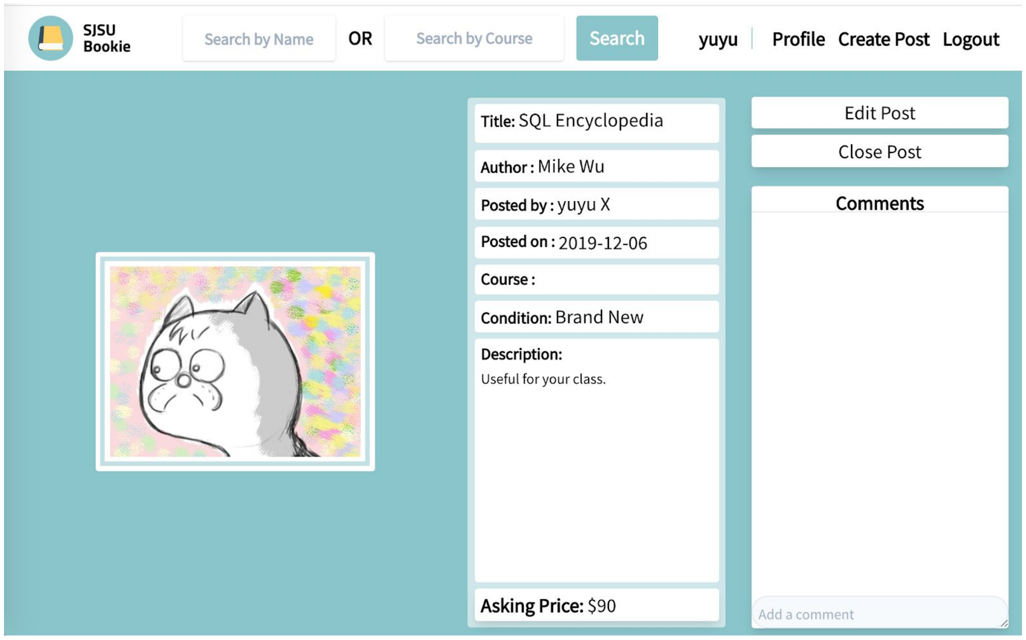
localhost:3000 says

Successfully made a sales post.

Create a Post

Book Title	SQL Encyclopedia
Author	Mike Wu
Condition	Brand New
Course	Course
Price (USD)	90
Description	Useful for your class.
Optional: Upload a Photo of your Book	No file chosen

The created post:



Title: SQL Encyclopedia

Author : Mike Wu

Posted by : yuyu X

Posted on : 2019-12-06

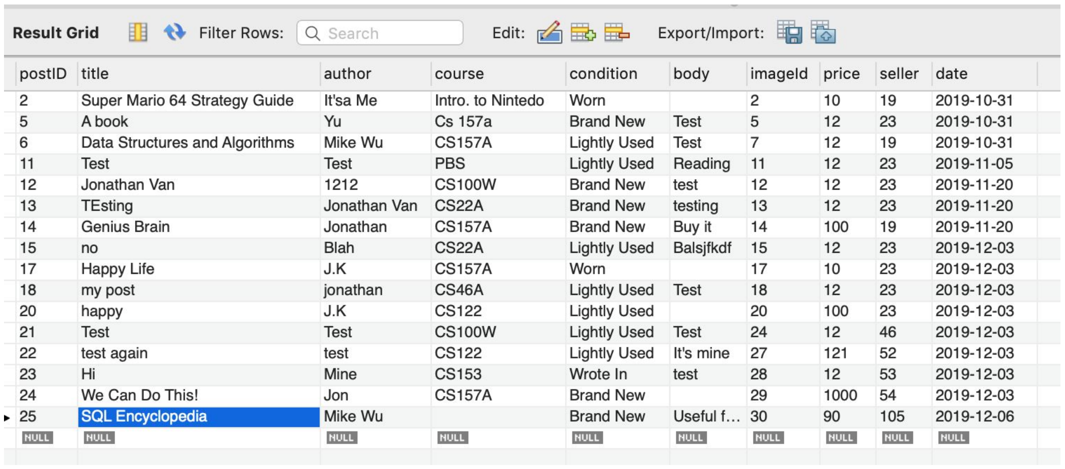
Course :

Condition: Brand New

Description:
Useful for your class.

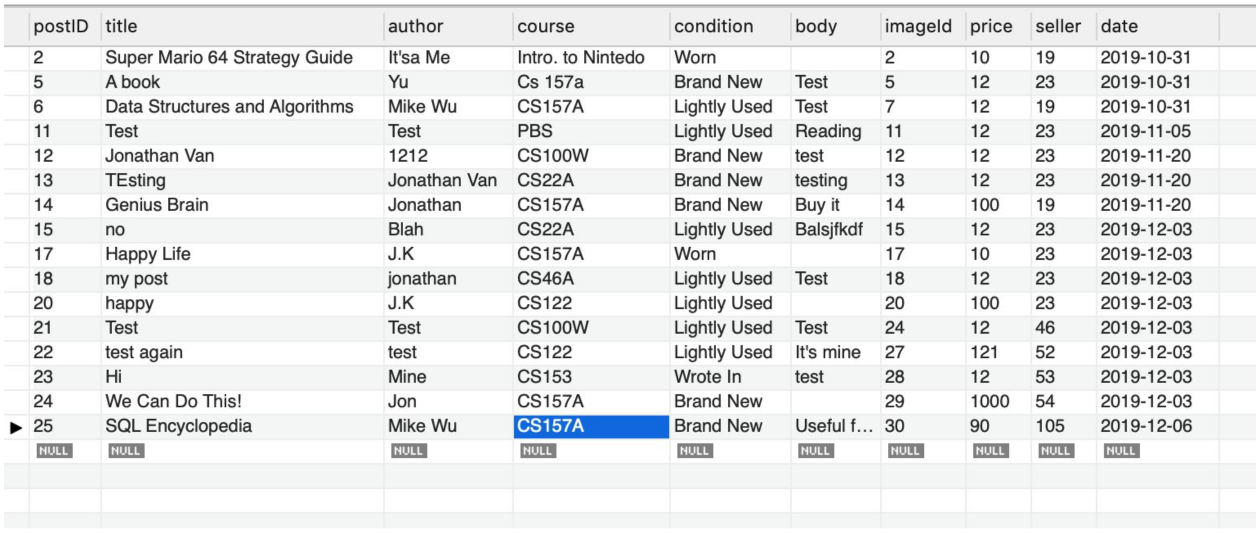
Asking Price: $90

Going to workbench, the newest created book is saved:

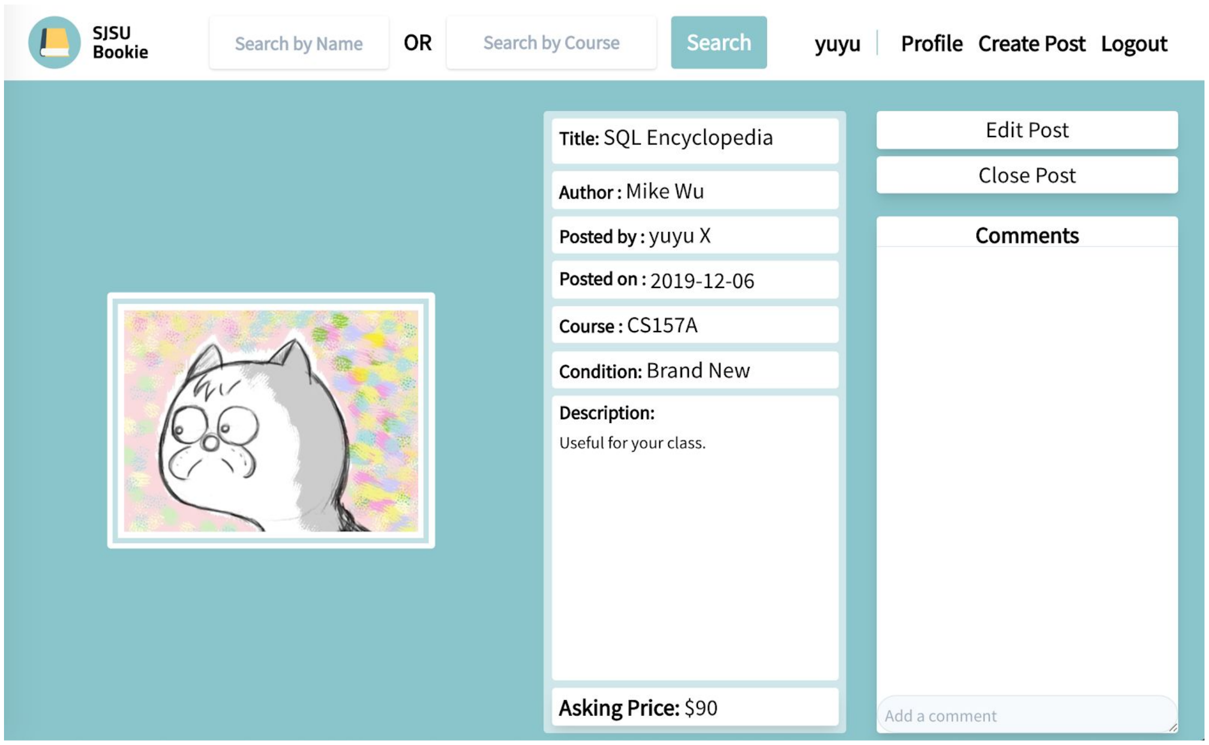


postID	title	author	course	condition	body	imageId	price	seller	date
2	Super Mario 64 Strategy Guide	It'sa Me	Intro. to Nintedo	Worn		2	10	19	2019-10-31
5	A book	Yu	Cs 157a	Brand New	Test	5	12	23	2019-10-31
6	Data Structures and Algorithms	Mike Wu	CS157A	Lightly Used	Test	7	12	19	2019-10-31
11	Test	Test	PBS	Lightly Used	Reading	11	12	23	2019-11-05
12	Jonathan Van	1212	CS100W	Brand New	test	12	12	23	2019-11-20
13	TEsting	Jonathan Van	CS22A	Brand New	testing	13	12	23	2019-11-20
14	Genius Brain	Jonathan	CS157A	Brand New	Buy it	14	100	19	2019-11-20
15	no	Blah	CS22A	Lightly Used	Balsjfkdf	15	12	23	2019-12-03
17	Happy Life	J.K	CS157A	Worn		17	10	23	2019-12-03
18	my post	jonathan	CS46A	Lightly Used	Test	18	12	23	2019-12-03
20	happy	J.K	CS122	Lightly Used		20	100	23	2019-12-03
21	Test	Test	CS100W	Lightly Used	Test	24	12	46	2019-12-03
22	test again	test	CS122	Lightly Used	It's mine	27	121	52	2019-12-03
23	Hi	Mine	CS153	Wrote In	test	28	12	53	2019-12-03
24	We Can Do This!	Jon	CS157A	Brand New		29	1000	54	2019-12-03
25	SQL Encyclopedia	Mike Wu		Brand New	Useful f...	30	90	105	2019-12-06
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

When we modified the database to add a course, it appears to front end:



	postID	title	author	course	condition	body	imageId	price	seller	date
	2	Super Mario 64 Strategy Guide	It'sa Me	Intro. to Nintedo	Worn		2	10	19	2019-10-31
	5	A book	Yu	Cs 157a	Brand New	Test	5	12	23	2019-10-31
	6	Data Structures and Algorithms	Mike Wu	CS157A	Lightly Used	Test	7	12	19	2019-10-31
	11	Test	Test	PBS	Lightly Used	Reading	11	12	23	2019-11-05
	12	Jonathan Van	1212	CS100W	Brand New	test	12	12	23	2019-11-20
	13	TEsting	Jonathan Van	CS22A	Brand New	testing	13	12	23	2019-11-20
	14	Genius Brain	Jonathan	CS157A	Brand New	Buy it	14	100	19	2019-11-20
	15	no	Blah	CS22A	Lightly Used	Balsjfkdf	15	12	23	2019-12-03
	17	Happy Life	J.K	CS157A	Worn		17	10	23	2019-12-03
	18	my post	jonathan	CS46A	Lightly Used	Test	18	12	23	2019-12-03
	20	happy	J.K	CS122	Lightly Used		20	100	23	2019-12-03
	21	Test	Test	CS100W	Lightly Used	Test	24	12	46	2019-12-03
	22	test again	test	CS122	Lightly Used	It's mine	27	121	52	2019-12-03
	23	Hi	Mine	CS153	Wrote In	test	28	12	53	2019-12-03
	24	We Can Do This!	Jon	CS157A	Brand New		29	1000	54	2019-12-03
▶	25	SQL Encyclopedia	Mike Wu	CS157A	Brand New	Useful f…	30	90	105	2019-12-06
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



Title: SQL Encyclopedia

Author : Mike Wu

Posted by : yuyu X

Posted on : 2019-12-06

Course : CS157A

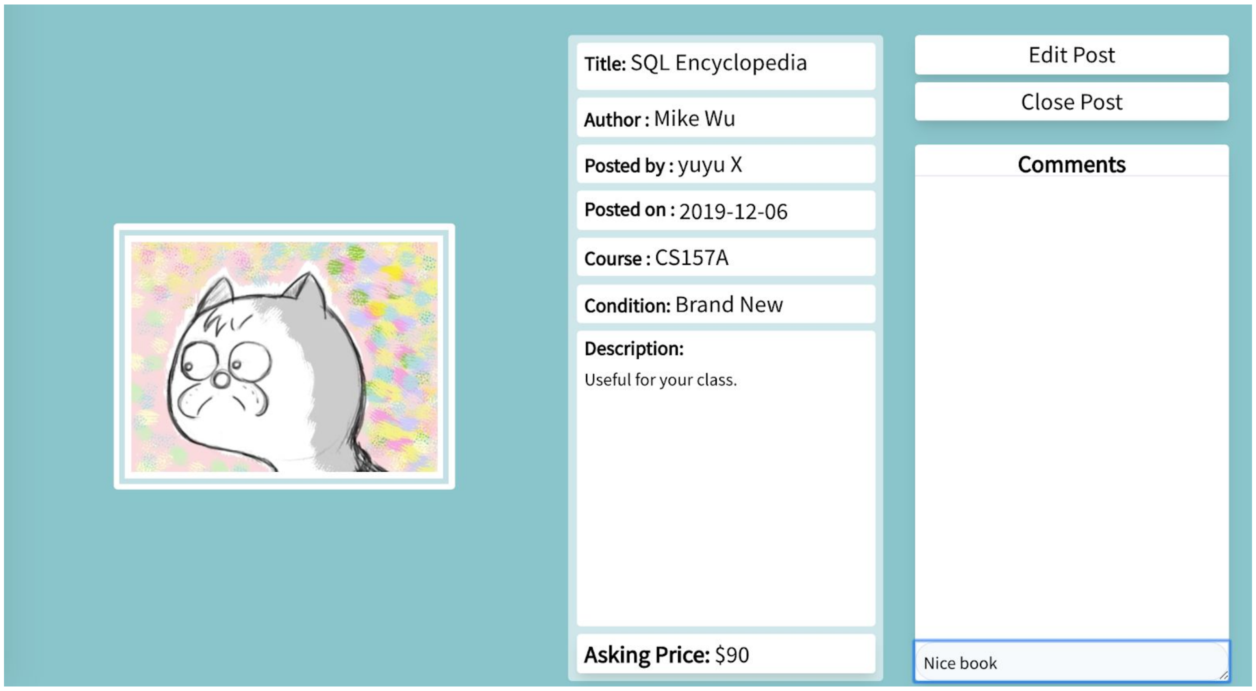
Condition: Brand New

Description:

Useful for your class.

Asking Price: $90

Leave a comment:



Title: SQL Encyclopedia

Author : Mike Wu

Posted by : yuyu X

Posted on : 2019-12-06

Course : CS157A

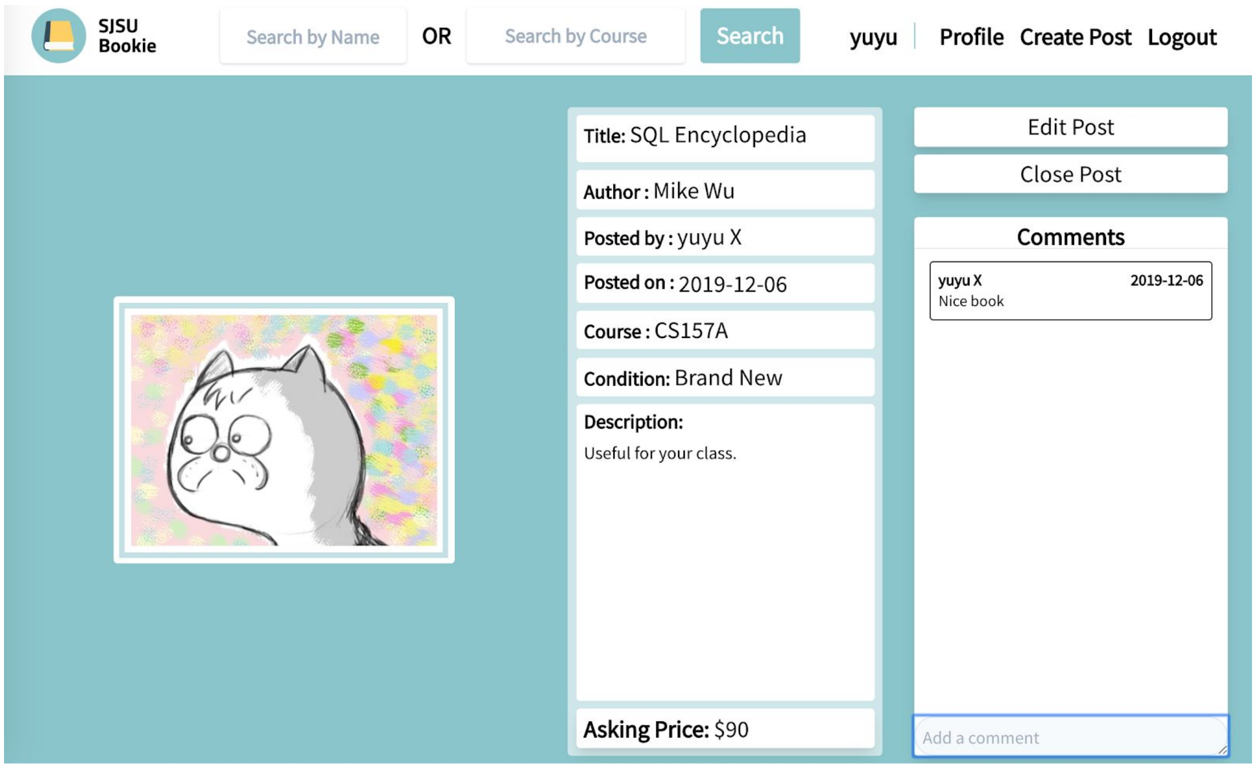
Condition: Brand New

Description:

Useful for your class.

Asking Price: $90

Click enter, it shows in the comments box:



Title: SQL Encyclopedia

Author : Mike Wu

Posted by : yuyu X

Posted on : 2019-12-06

Course : CS157A

Condition: Brand New

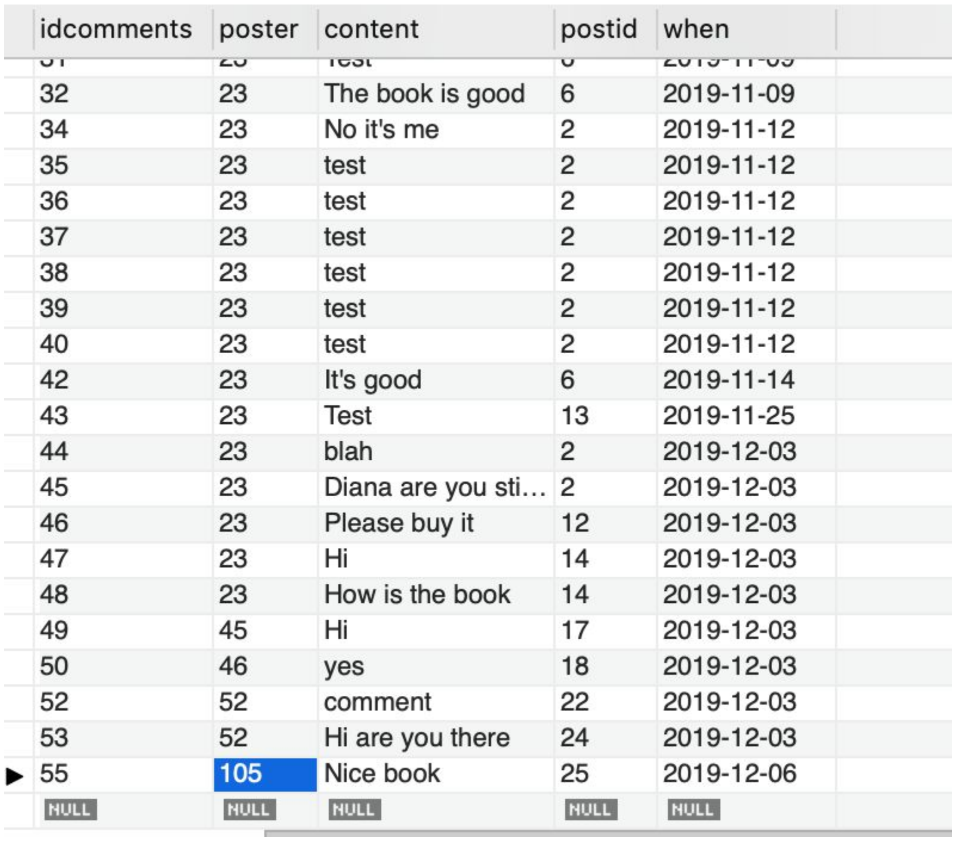
Description:

Useful for your class.

Asking Price: $90

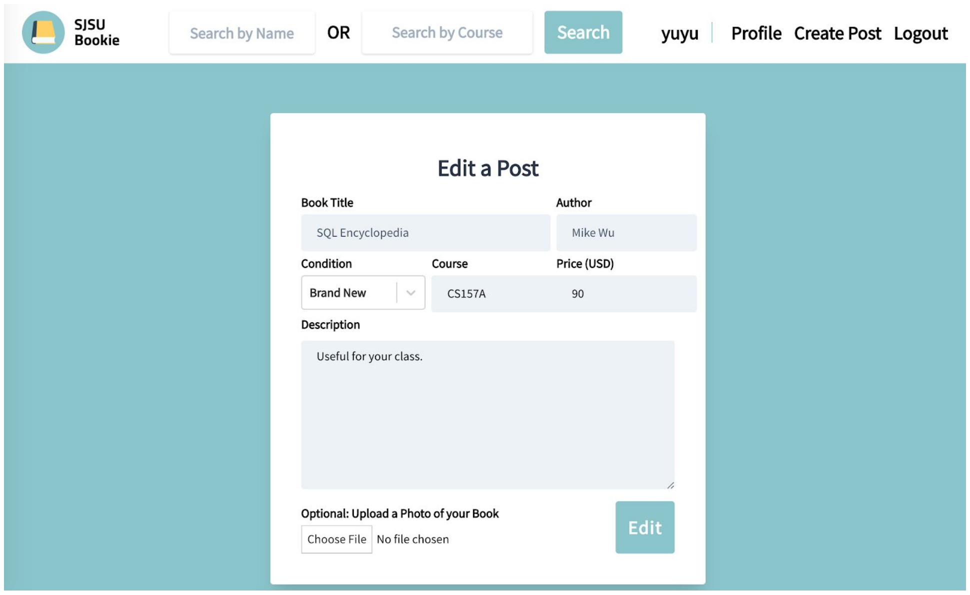
yuyu X 2019-12-06
Nice book

The comment saved into workbench:



idcomments	poster	content	postid	when
32	23	The book is good	6	2019-11-09
34	23	No it's me	2	2019-11-12
35	23	test	2	2019-11-12
36	23	test	2	2019-11-12
37	23	test	2	2019-11-12
38	23	test	2	2019-11-12
39	23	test	2	2019-11-12
40	23	test	2	2019-11-12
42	23	It's good	6	2019-11-14
43	23	Test	13	2019-11-25
44	23	blah	2	2019-12-03
45	23	Diana are you sti…	2	2019-12-03
46	23	Please buy it	12	2019-12-03
47	23	Hi	14	2019-12-03
48	23	How is the book	14	2019-12-03
49	45	Hi	17	2019-12-03
50	46	yes	18	2019-12-03
52	52	comment	22	2019-12-03
53	52	Hi are you there	24	2019-12-03
55	105	Nice book	25	2019-12-06
NULL	NULL	NULL	NULL	NULL

User can edit post:



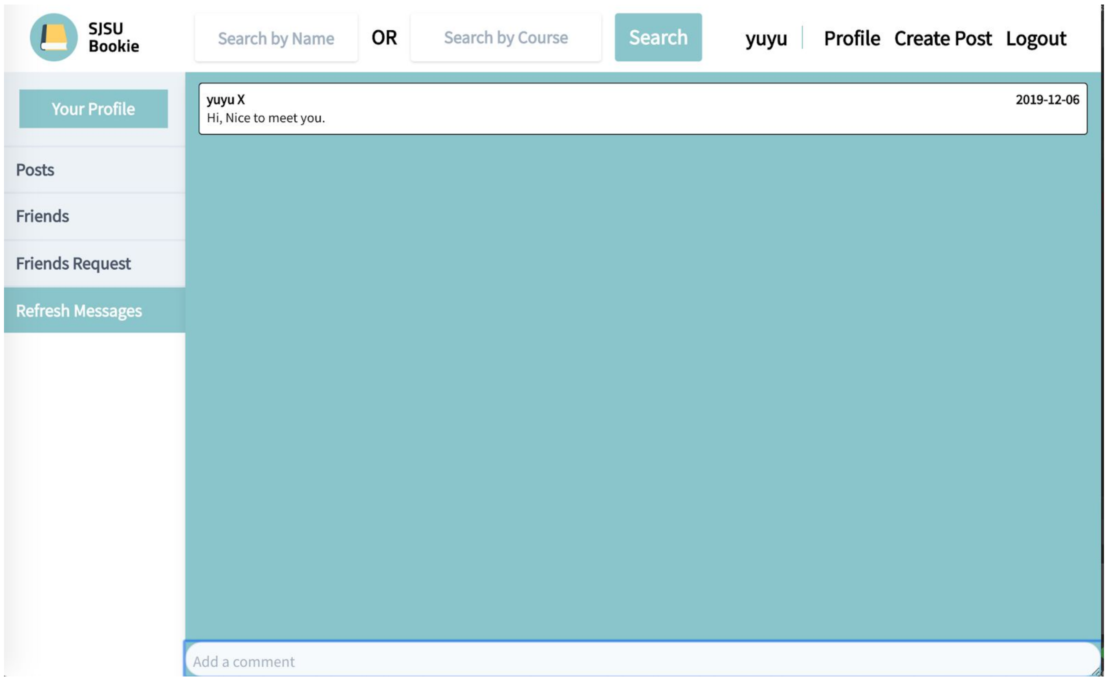
Edit a Post

Book Title: SQL Encyclopedia
Author: Mike Wu
Condition: Brand New
Course: CS157A
Price (USD): 90
Description: Useful for your class.
Optional: Upload a Photo of your Book
Choose File No file chosen
Edit

User can also delete table by click close table.

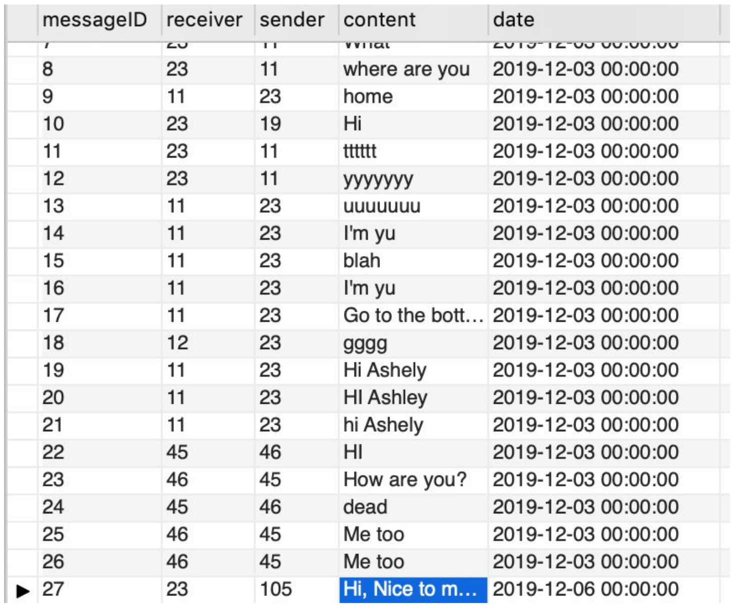
7. Messaging & Friending

7.1. SJSU Bookie allows users to send friend requests to users that they need to communicate with privately in order to make sure the public does not know about private information between the two of them.



yuyu X 2019-12-06
Hi, Nice to meet you.

It saved in our workbench message table:

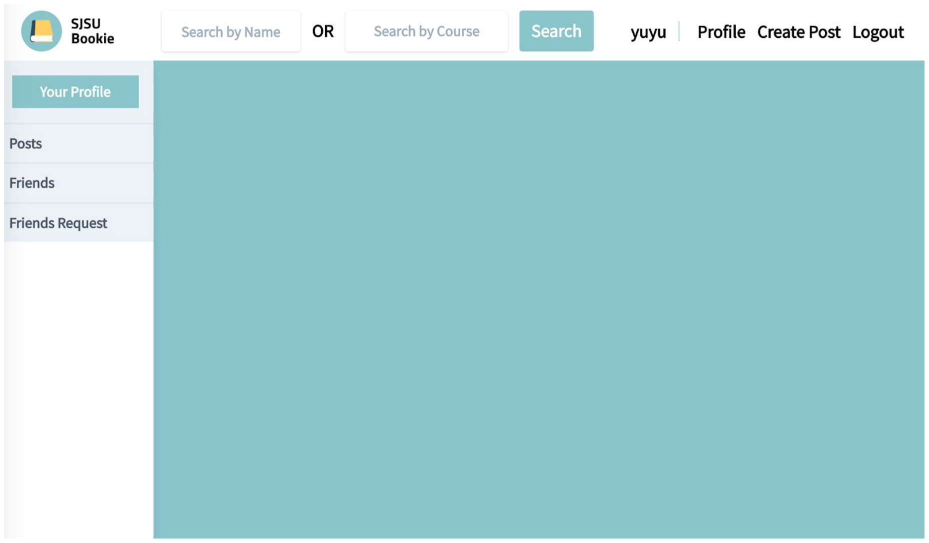


	messageID	receiver	sender	content	date
	8	23	11	where are you	2019-12-03 00:00:00
	9	11	23	home	2019-12-03 00:00:00
	10	23	19	Hi	2019-12-03 00:00:00
	11	23	11	tttttt	2019-12-03 00:00:00
	12	23	11	yyyyyyy	2019-12-03 00:00:00
	13	11	23	uuuuuuu	2019-12-03 00:00:00
	14	11	23	I'm yu	2019-12-03 00:00:00
	15	11	23	blah	2019-12-03 00:00:00
	16	11	23	I'm yu	2019-12-03 00:00:00
	17	11	23	Go to the bott…	2019-12-03 00:00:00
	18	12	23	gggg	2019-12-03 00:00:00
	19	11	23	Hi Ashely	2019-12-03 00:00:00
	20	11	23	HI Ashley	2019-12-03 00:00:00
	21	11	23	hi Ashely	2019-12-03 00:00:00
	22	45	46	HI	2019-12-03 00:00:00
	23	46	45	How are you?	2019-12-03 00:00:00
	24	45	46	dead	2019-12-03 00:00:00
	25	46	45	Me too	2019-12-03 00:00:00
	26	46	45	Me too	2019-12-03 00:00:00
▶	27	23	105	Hi, Nice to m…	2019-12-06 00:00:00

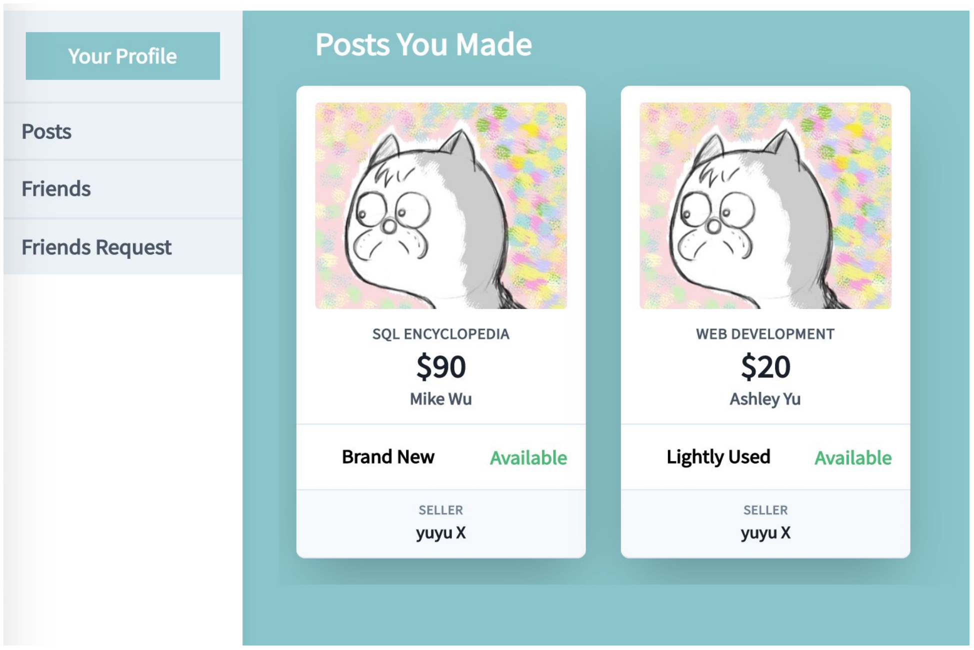
8. Profile View

8.1. A user can have access to their profile to view posts that they have posted, favorited, or posts that have either been closed or deleted.

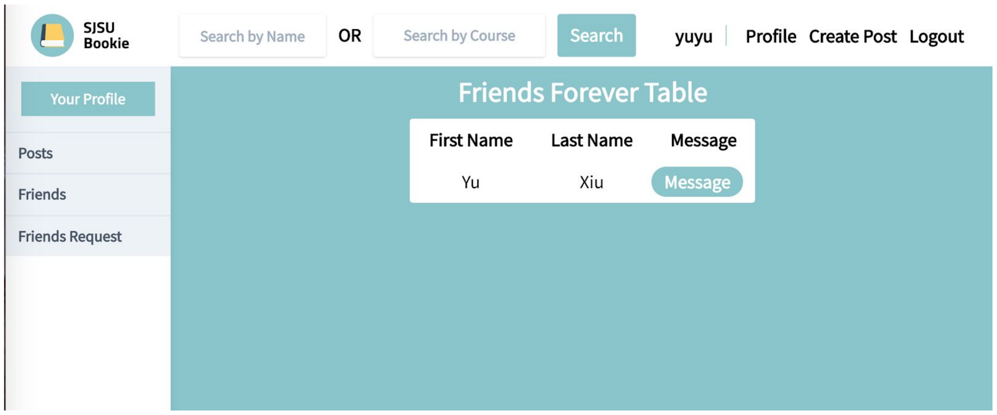
When pressing "Profile" tab, user can see this page:



When clicking on "Posts":



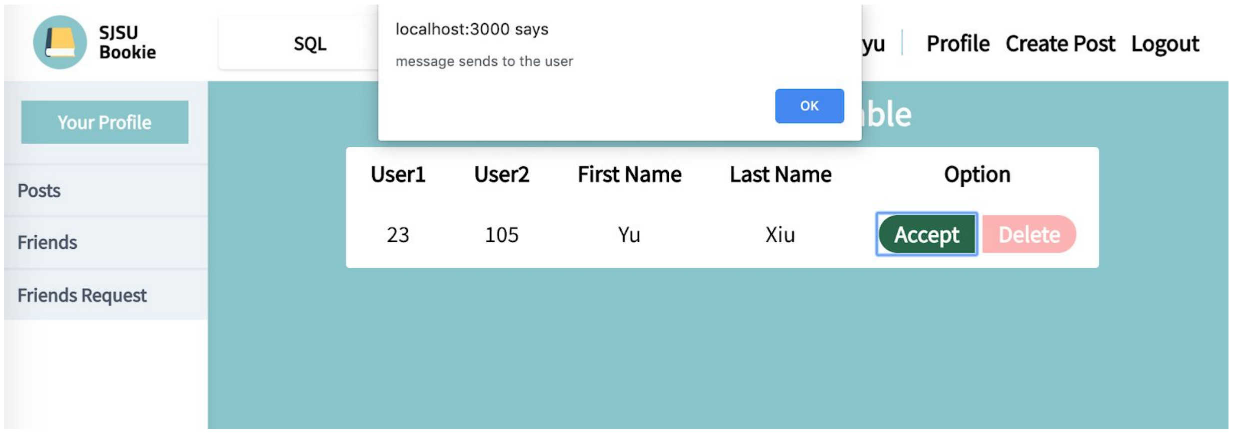
When clicking on “Friends”:



Friends Forever Table

First Name	Last Name	Message
Yu	Xiu	Message

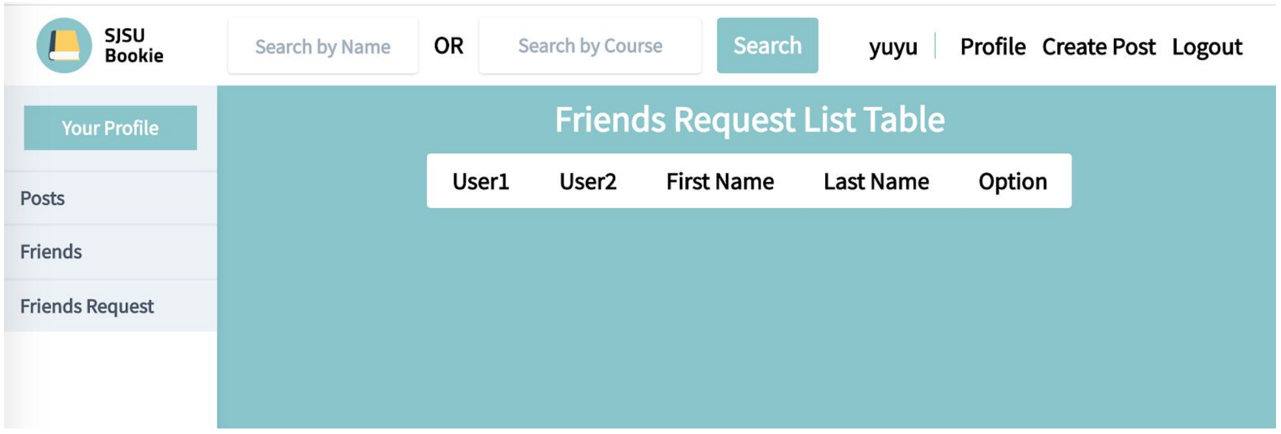
When clicking on “Friends Request” if there is a request list it, and if not, it is empty:



localhost:3000 says

message sends to the user

User1	User2	First Name	Last Name	Option
23	105	Yu	Xiu	Accept Delete

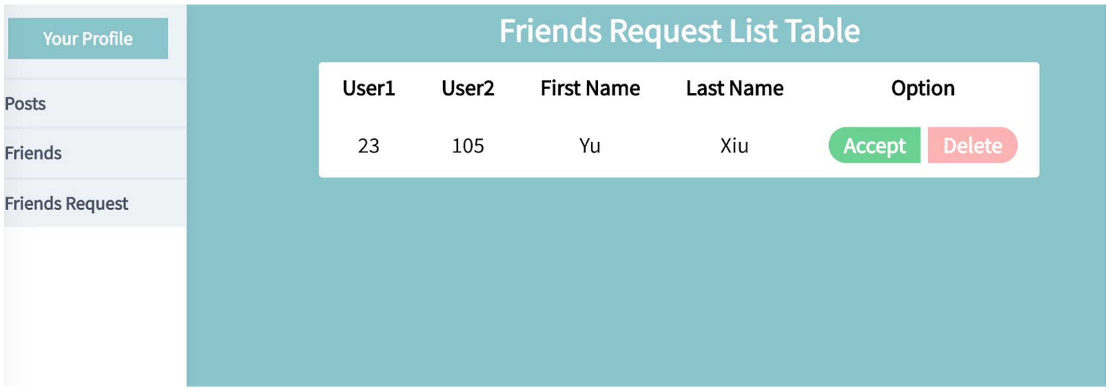


Friends Request List Table

User1	User2	First Name	Last Name	Option

9. Friend List

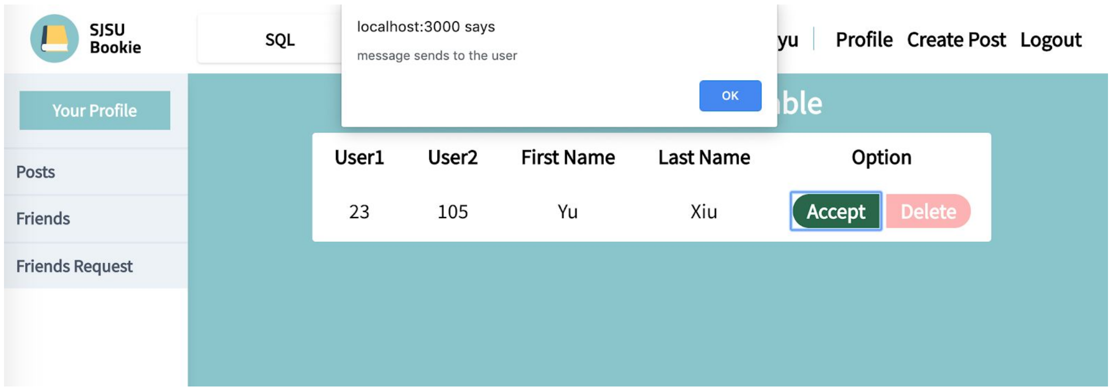
When a user sends a friend request, it shows in the Friends request bar:



Friends Request List Table

User1	User2	First Name	Last Name	Option
23	105	Yu	Xiu	Accept Delete

When the user accept the request:

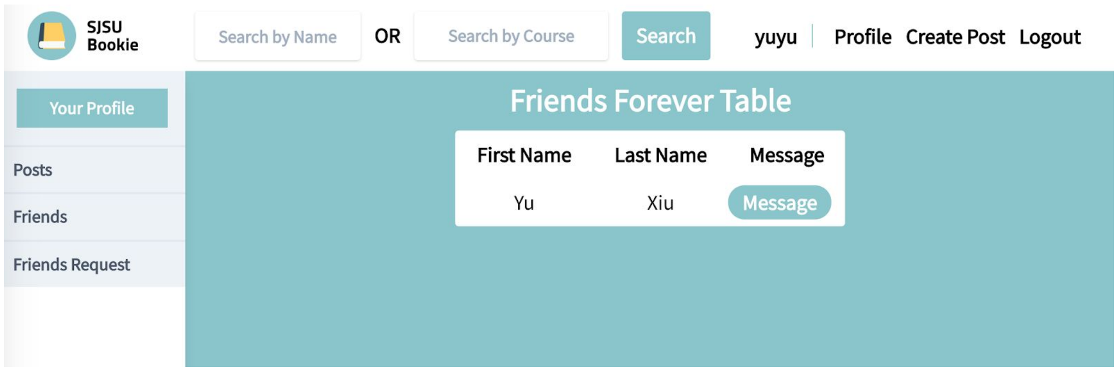


localhost:3000 says

message sends to the user

User1	User2	First Name	Last Name	Option
23	105	Yu	Xiu	Accept Delete

If we click on Friends bar, we see the friend had been accepted:



Friends Forever Table

First Name	Last Name	Message
Yu	Xiu	Message

Procedure of how to set up and run Bookie

Assess to our website: https://sjsubookie.netlify.com

To install everything yourself:

Critical tools:

1) **Node.js**
2) **Javascript runnable IDE or environment (IntelliJ or Visual Studio Code)**
3) **MySQL**

Steps:

Open the terminal, or terminal on either IntelliJ or Visual Studio Code

1. **Make a directory to hold the files and make sure you are within that folder directory (cd into it)**
2. **Run:**

   git clone --recurse-submodules

   https://github.com/CS157A-Team4/CS157A-Team4.git
3. **Type cd CS157A-Team4/**
4. **Type npm install --save**
5. **If there are any issues, type npm audit fix**
6. **Type cd SJSUBookieBackEnd/**
7. **Type npm install --save again**
8. **Again, if there are any issues, type npm audit fix**
9. **In the SJSUBookieBackEnd directory, please type "vim routes/database.js"**
10. **On line 20, please replace the connection with these credentials connection:**

```
let connection = mysql.createConnection({
  host: "34.94.36.13",
  user: "Team4Boss",
  database:"cs157a",
  password: "Googler",
  multipleStatements: true
});
```

11. **Afterwards, do an "npm start" while in the SJSUBookieBackEnd directory.**
12. **Then if successful, then open a new terminal and then do an "npm start" after navigating into the CS157A-Team4 Directory we cloned.**
13. **Afterwards, the application should be running successfully.**

Project Conclusion

Statements from each team member about Lesson Learned from this DB project

Yu Xiu: I learned how to make a three tier architecture web application with tailwindCSS, JavaScript, Nodejs and MySQL. It is good to know how to query data from database and how to connect front-end to the back-end out of this project. From this project, I know what is front-end and back-end. This is the first time I have such a complicated group project, so I also learned how to work with a group, and my partners helped me a lot.

Jonathan Van: Before, I only knew how to do frontend work using ReactJS and TailwindCSS, and SQL so it was new to me to be doing backend on top of that. I am thankful that our professor provided us with SQL lessons to improve our projects and I am glad he did because it helped my understanding of the code. (I wish he taught it sooner, like when we were starting the projects). It was really fun learning how to deploy the servers and the front end so that anyone can access it. Ultimately, this project improved my SQL skills, and taught me how to do front and backend, so I am thankful for that. Furthermore, setting up and learning the advantages of Google Cloud Platform's 5.7 SQL Server was really beneficial because it is something we can use in the future for industry with its robustness, size and disaster recovery ability. (Let's be real our project was probably one of the best looking professor, and we should get Google Jobs right?)

Cole McKinnon: I used to be frightened by the idea of implementing a database in a project! I had briefly used a NoSQL DB and to me the whole thing seemed like magic. Now I can confidently say that I have a thorough understanding of SQL databases, and I'm looking forward to using SQL in my future projects. Because each of us was responsible for the full stack of each of our features, I learned not only how to get user input, but also how to package that input, send it, and store it in the database. My React and Javascript skills also improved dramatically while working on this project, and I'm so proud of not only my own work, but also the work of my whole team. Databases are such a crucial part of nearly all modern applications, and I'm so thankful that I had the opportunity to learn about and implement a SQL database.

Future improvement of the DB application

In the future, we will try to reduce database redundancy and make a better DB design. We will also try to design our front-end or web pages look nicer and mobile friendly (we were attempting it, but realized it was too time consuming). Also, the back-end can be more neat and customer friendly. Furthermore, we would try to make our messaging system use sockets rather than having to refresh it every time. I think in the future we would try to make it applicable to all items, and not just books. I think we can better optimize it for some query speeds, but other than that, we are really proud of our project.

The Real Conclusion

It's still vivid the first day we sit in the library to make the blueprint of our Bookie web application, and now, it's near the time to say goodbye. Most projects required a large team to be able to implement a smooth looking product like this, and we managed with 3. We suffered when our database was hacked, we struggled when our web crashed, and we laughed when we saw our beautiful and functional web application. All the things that we learned and suffered from this project make us stronger. We've improved in that we know how to build and deploy MySQL Web Applications, we learned to work in a group, we learned how to debug when things go wrong, and most importantly, we were able to deliver a finished product. This is a very good project, and we were so glad to work with each other. We walked from the dark to lights and experienced from vague to clear, so we built trust with each other. We all had our own strengths and managed to cover for and teach one another when it came to this project. The most treasurable things that we learned from this project are never give up and trust your partners. In the future, we will tell ourselves, "We can do it!"

Thank you, Professor Wu.