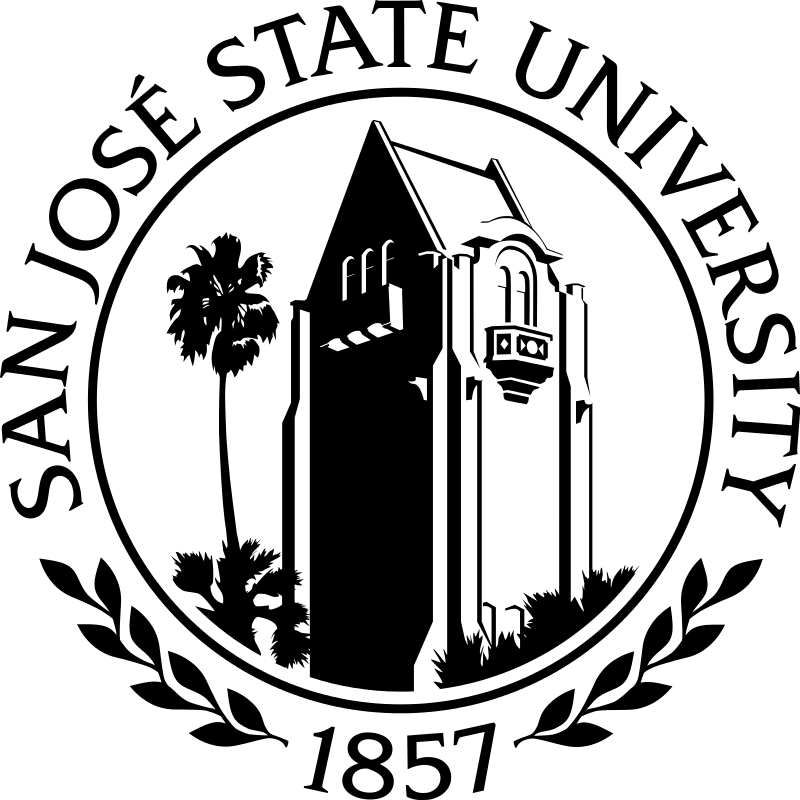
**CS 157A Project Requirement**

**Three-Tier Web Application**

**SJSU Bookie**

****

Cole McKinnon, Jonathan Van, Yu Xiu

Team 4­­­­­­

Advisor: Dr. Mike Wu

Sep. 22, 2019

**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.

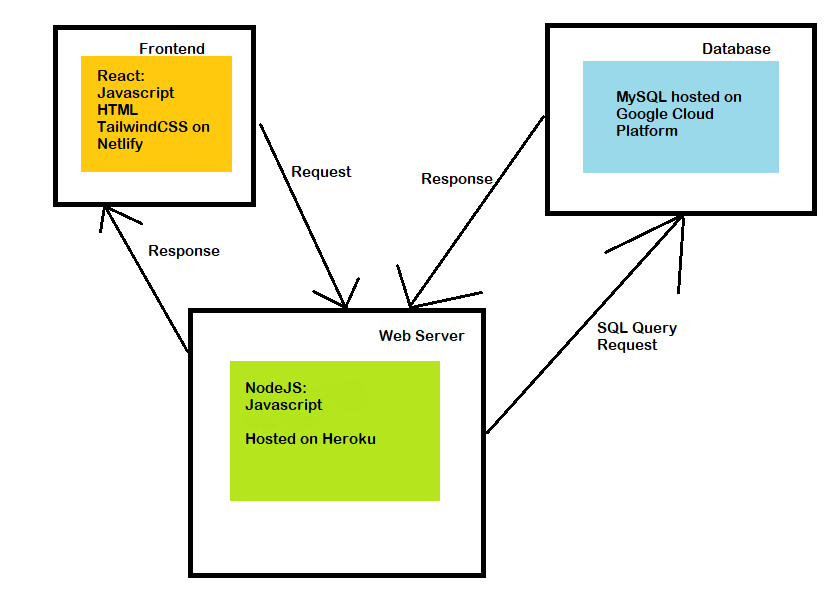


Figure 1: 3 Tier Architecture Design for SJSU Bookie

**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**

**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.

1. 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.

1. 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.