PROJECT DOCUMENTATION: BOOK EXCHANGE PLATFORM

By Daniel Elendu

At the moment, book exchange (and sale) on the campus is done on a Facebook group named 'Free and For Sale'. People advertise their books as posts and then interested students can comment and commence negotiations on the platform. This is fine but then one begins to get a lot of unwanted notifications about these posts. There is the option to mute notifications for a group but this can cause a student to miss an important post. The Book Exchange platform is an application that would provide a similar service as the Free and For Sale group but would approach notifying users/students in a different manner. When a book is posted, the poster indicated the course and section that this book is for. Every user of the app will indicate the set of courses that they want to follow and this way, they only see posts and get notified about books for the courses they indicated interest in.

SYSTEM ARCHITECTURE

The software stack I intend to use for this app are:

i.) Node.js

ii.) SQLite3

iii.) HTML

iv.) CSS

v.) JavaScript

vi.) Mocha/Chai (*for Unit Tests)*

vii.) Bootstrap

viii.) Passport (*for authentication*)

My choices might not be limited to these pieces of software since I had planned to add to my design as necessary once the actual process of application development starts. I believe there will be various node modules that will be added to this stack as project dependencies.

APPLICATION DEVELOPMENT

A. SYSTEM DESIGN AND MOCKUPS

a. Login and Authentication

At the moment, I plan to use password-local for authentication of the user. Once this has been established, I would test out adding passport-google to allow the user to be authenticated using their Vassar email account. I thought this would be most ideal since the app would ideally be for users within the Vassar ecosystem.

b. Posts

These features below will be available to the user on the main screen of the application as a floating action button or accessible under a tab. The plan is to have them as options under a floating action button or have them as selectable pills under a tab.

i. Textbooks

An authenticated user would be able to post textbooks to the app. This would be done using a form. The user will indicate the course and section this book is for and then the title of the textbook as well as the cost. If the book is for free, the user can indicate this in the cost field as 0.0. Any other description about the book (extra details the user will like to share), can also be included.

ii. Requests

An authenticated user would also be able to post a request. A request is like a regular post, but the fields available here are just for a description of the request for a textbook and then the user can select the course and section this textbook is required for so that users that follow that course can view the request.

iii. **Comments**

Comments happen on posts and requests and users can make remarks about a post. These will show up in descending order using the date and time of the post to sort (i.e. newer posts show up at the top)

c. **New Textbooks Notifications**

Users will be welcomed to the app with a view of a list of posts about textbooks. It will be under a tabbed view but will be the default tab. The user can scroll and see a list of posts in chronological order with newer textbook posts showing up first.

d. **New Requests Notifications**

Users can view notifications within their circle (courses they follow) and just like the newly posted textbooks, they can scroll and see requests in chronological order with the newer requests showing up first.

e. **View and Edit Profile**

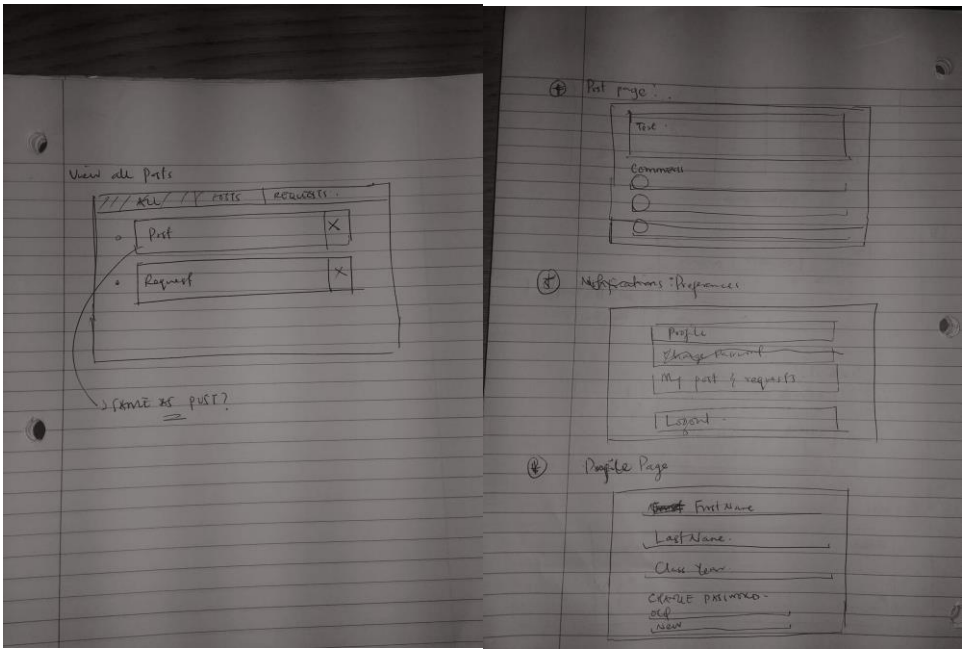
The user will also be able to edit things on their public profile such as their name and class year, change their password or delete their account

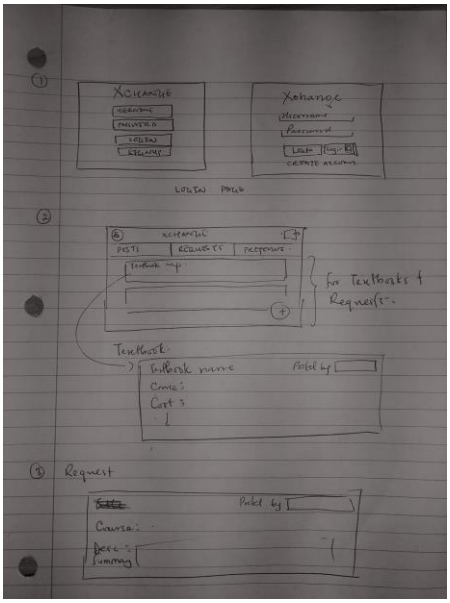
f. **View and Delete All Posts (Book Posts, Requests and Comments)**

In this portion of the app the user will be able to view all their posts and requests and delete them individually.

g. **Follow Courses**

Users will be able to search courses which will appear on a list. Users can then select the ones they want to follow (get notifications for).





Mockup Diagrams

B. DATABASE DESIGN

The preliminary Database Design is highlighted below. The database tables are:

i.) Account

The fields here are id (primary key), email and password. These are necessary identification parameters of the user.

a. User

The user inherits from the Account and add details that the other relations in the database will use. The fields in this table are first_name, last_name, middle_name , class_year and posts (a set of posts the user has made).

Relationship(s):

- User has a one-to-many relationship with the Post relation.

ii.) Post

An instance of the post relation has the fields id, post_time, user, views and comments which are common to all forms of posts that will be highlighted below.

Relationship(s):

- Post has a many-to-one relationship with User.
- Post has a one-to-many relationship with Comments.

a. Comment

The Comment relation inherits from the Post relation. An instance of it has the fields comment_text for the actual comment and Post (the post it is related to).

Relationship(s):

- Comments have a many-to-one relationship with Post.

b. Book_Post

Book inherits from the Post relation. An instance of it has the fields description_text for descriptions about the book in the post and textbooks (the set of textbooks it is related to).

Relationship(s):

- Book_Post has a one-to-many relationship with Textbook.

c. Request

Request inherits from the Post relation. An instance of it has the fields request_text which contains information about the request and section (the course section the request post is related to).

Relationship(s):

- It has a many-to-one relationship with Section.

iii.) Textbook

An instance of the textbook relation has the fields id, book_title which is the name of the textbook, author, course which is the course the textbook is for, cost which is the amount required to exchange/purchase the textbook and the book_post this textbook is related to.

Relationship(s):

- It has a many-to-one relationship with Book_Post.
- It has a many-to-one relationship with Course.

iv.) Course

An instance of the course relation has the fields id, course_code, course_title and textbooks which is the set of textbooks related to an instance of the course

Relationship(s):

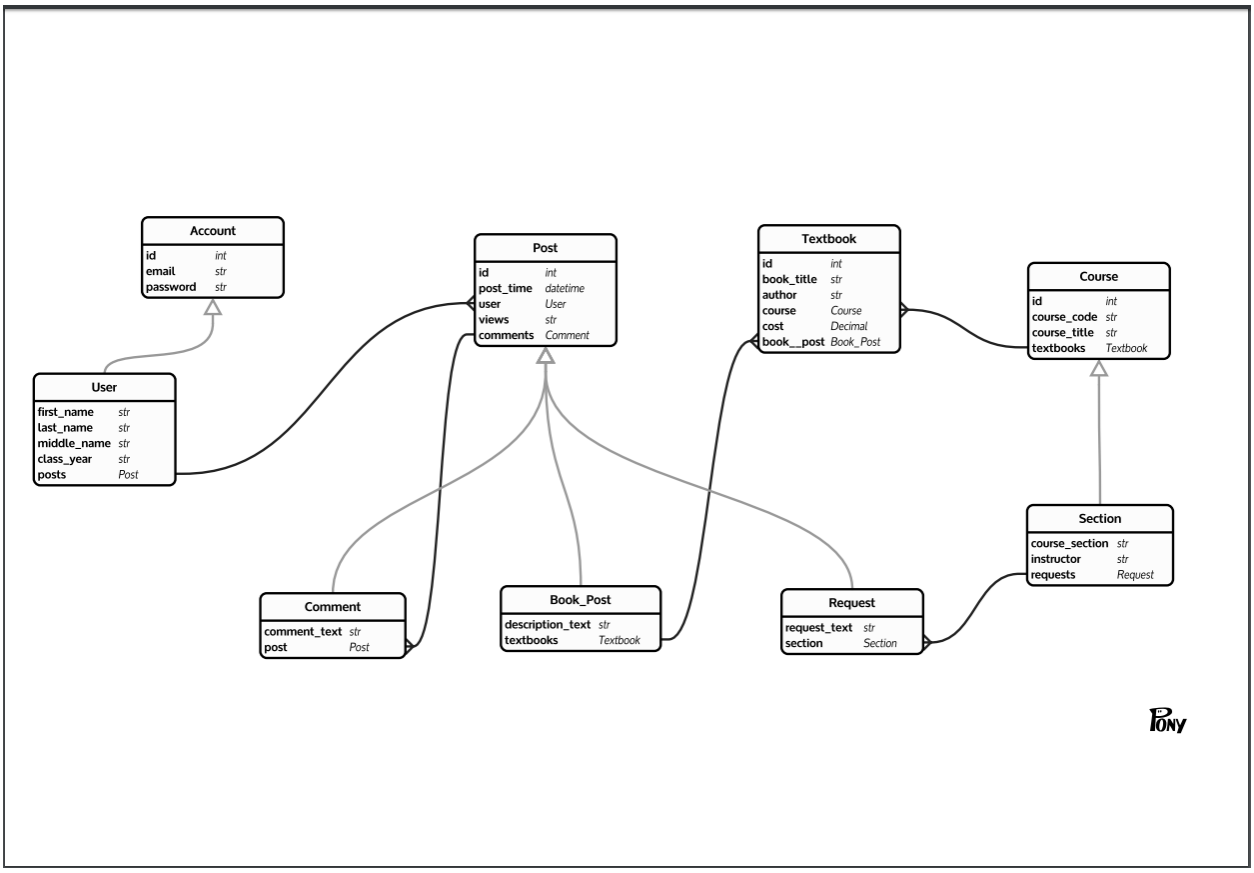
- It has a one-to-many relationship with Textbook.

a. Section

Section inherits from course and has the fields course_section which is the course section id, instructor and requests which is which is the set of all requests related to an instance of the section of a course.

Relationship(s):

- It has a one-to-many relationship with Request.



ER Diagram for App Database.

PROJECT STATUS

At the moment, I only have my ER Diagram and Mockups of what my project and have not started working on the actual code for my project. This was a result of early deadlines for intensive projects and moving forward things I would have loved to implement are things like having a UML Diagram to describe by models for the JS code on the front-end and back-end of my app. I would have also liked to have a light demo for authentication on using passport local so I could start working and just add on other passport strategies. All the things I listed above except DATABASE DESIGN are things that I would have loved to document in my final report. I would have also loved to have a markdown documentation using GitHub that could have proper images of a working model but at the moment, this is not possible due to the time constraint.