Code review
Renewal system

Overview

The code you see is the entire implementation of our project. The project implemented with Meteor.js. We also used MongoDB for storing the data. In meteor we used react js for displaying data and easily manipulate react component via standard routing.

Goals

Building a stable system which provides privileges for users according to their respective type in the Library system.
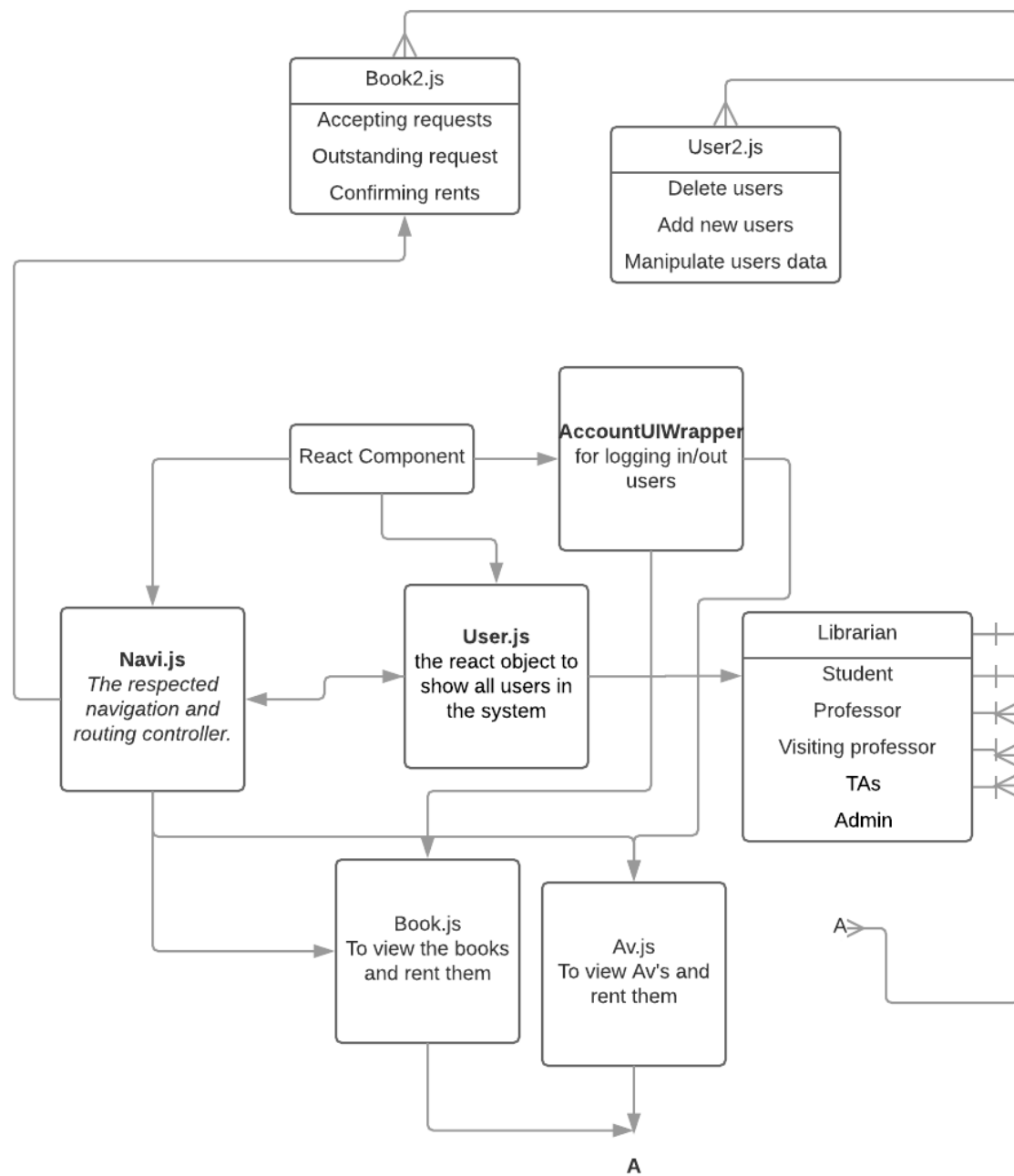
Usage Scenarios

According to user's type it is possible for him to add documents add patron cards and accept renting request if the user has a privilege to do so and to rent documents if he was a patron, we can conclude in:

A. Creating users.
B. Manipulating users Data.
C. Creating documents.
D. Editing Documents.
E. Ability to perform outstanding requests and manipulate the order of the queue.

Class Diagrams

Class diagram of most part of our system. Admin panel classes extends it (do not directly inherit)

Book2.js
- Accepting requests
- Outstanding request
- Confirming rents

User2.js
- Delete users
- Add new users
- Manipulate users data

React Component

AccountUIWrapper
for logging in/out users

Navi.js
The respected navigation and routing controller.

User.js
the react object to show all users in the system

Librarian
- Student
- Professor
- Visiting professor
- TAs
- Admin

Book.js
To view the books and rent them

Av.js
To view Av's and rent them

A

A

Code
To be reviewed

Full project:

Categories for review
1. Design decisions
2. API design
[2.1][Reviewer1]. Not every document has property 'reference' (REMOVED, lines: 42 => 10-12).
[2.1] Owner: The are either a reference or not, that why it is not optional. **Closed.**
[2.2][Reviewer1]. Document of type 'Journal Article' should also have a property 'Issue Publication Date' (REMOVED)
[2.2] Owner: I agree, **to_be_implemented**.
3. Architecture (including inheritance hierarchy, if any)
[3.1] Reviewer2. In class Student: fields degree and dateOfStudyBeginning shouldn't be optional
[3.1] owner: **TO_BE_FIXED**

[3.2] Reviewer3. In class Faculty. There is no VP (visiting professor), should be added
[3.2] owner: there is a visiting professor class please check:[REMOVED]
[3.2] Reviewer3. I mean identifiers in WorkPosition in faculty.js
[3.2] owner: I understand, **to_be_fixed**.

[3.3] Reviewer3. In class Faculty. fields.position should not be optional
[3.3] owner: why the fields should not be optional?
[3.3] Reviewer3. Every faculty member should have exact type of position (either 'Professor', 'Instructor', 'TA' or 'VP').
[3.3] Owner: I agree, **to_be_fixed**.

[3.4] Reviewer3. In class Student. fields.name should not be optional
[3.4] owner: I agree, **To_be_implemented**.

[3.5] Reviewer3. Overall. Would be better to store constants (like "TA", "Student" strings) in specific file: will helps to translate app to other languages and to not make mistakes
[3.5] owner: Could you please be more specific, I could not follow your Idea.

[3.5] Reviewer3. It would be better to have separate file to store constants which you use in different classes. That will help you in future to do localization much faster

[3.5] Owner: the strings are small in size, and writing them as strings is easier for a coder rather than writing a name of a constant variable, so my question do you suggest to name for example "visiting professor" as visitingProfessor rather than typing the string itself?

[3.5] Ok, I agree with you, that is your choice. Yes, my suggestion was about that. I meant something like that stackoverflow.com/questions/5191666/

[3.5] Owner: interesting point, I will keep it in mind. **Closed**.

4. Implementation techniques, in particular choice of data structures and algorithms

[4.1] Reviewer2. In halper changeGroupe of class User: switch case would be better.

[4.1] Owner: Less LOC, great advice, **to_be_fixed**.

[4.2] Reviewer2. In halper delNotification of class User: I think you should remove notifications not by title but by special unique ID.

[4.2] Owner: why it is better? Is it more efficient in terms of time complexity or it would delete multiple notifications if they are the same?

[4.2] Reviewer2, If there are similar notifications the deletion will happen to all of them.

[4.2] Owner: I agree, **to_be_fixed**.

5. Exceptions handling - Contracts

6. Programming style, names

[6.1] Reviewer4. Almost in every class: In final version there should not be "In progress" things. Everything should be completed for the deadline.

[6.1] Owner: it is for the next deadline. **Closed**.

[6.2] Reviewer4. What will be instead of "In Progress", in almost every class?

[6.2] Owner: there should be further implementation for upcoming deadlines and new requirements. **Closed.**

[6.3] Reviewer3. In librarian.js. Grammar error: should be 'privilege' not 'privilegy'

[6.3] I agree, **to_be_fixed**.

[6.4] Reviewer5. Almost in every class code style errors: in import sometimes you have spaces between curly brackets, sometimes have not. Determine your code style.

[6.4]Owner: Our coding style is camel style specially for JS. However, I agree, **to_be_fixed**

[6.5] Reviewer5. In admin.js and special.js: you have not comments "//In progress..." in properties of "fields", while you have these comments in another classes.

[6.5] Owner: Redundant comment for Mr Ali comment before. **Closed.**

[6.6] Reviewer5. In faculty.js, librarian.js, special.js, student.js and user.js code style errors: you importing two classes from single path in different lines, when you need to do it in one line by separating them with commas.

[6.6] Owner: Redundant imports, **to_be_fixed**.

[6.7][Reviewer1]. Up to you, but I think names of columns in MongoDB should be in camelCase for the reason that MongoDB speaks to javascript (source: https://stackoverflow.com/questions/5916080/what-are-naming-conventions-for-mongodb).

[6.7] Owner: I agree, to_be_fixed.

[6.8] Reviewer3. In librarian.js. Why decided to use list of strings ['priv1', 'priv2', 'priv3'], not list of integers?

[6.8] list of integers would be more efficient in terms of time complexity, I give you that but a list of strings would be easier to manipulate data, which one is better -in your opinion- easier to code or more time-complexity efficient?

[6.8] Agree, in this case would be better to leave as strings.
 Closed.

7. Comments and documentation

[7.1] Reviewer4. Overall: Write comments only when you implement something difficult. You should not comment simple things in every line.

[7.1] Thanks for the observation, you are correct, **Closed**.

[7.2] Reviewer6. Please, add descriptions for classes to documentation.

[7.2] Owner: Do you mean the documentation in github, or the UML diagram above here in this doc?

[7.2] I think in the UML diagram, your UML diagram lacks the description of the classes.

[7.2] Owner: I agree, **to_be_fixed**.