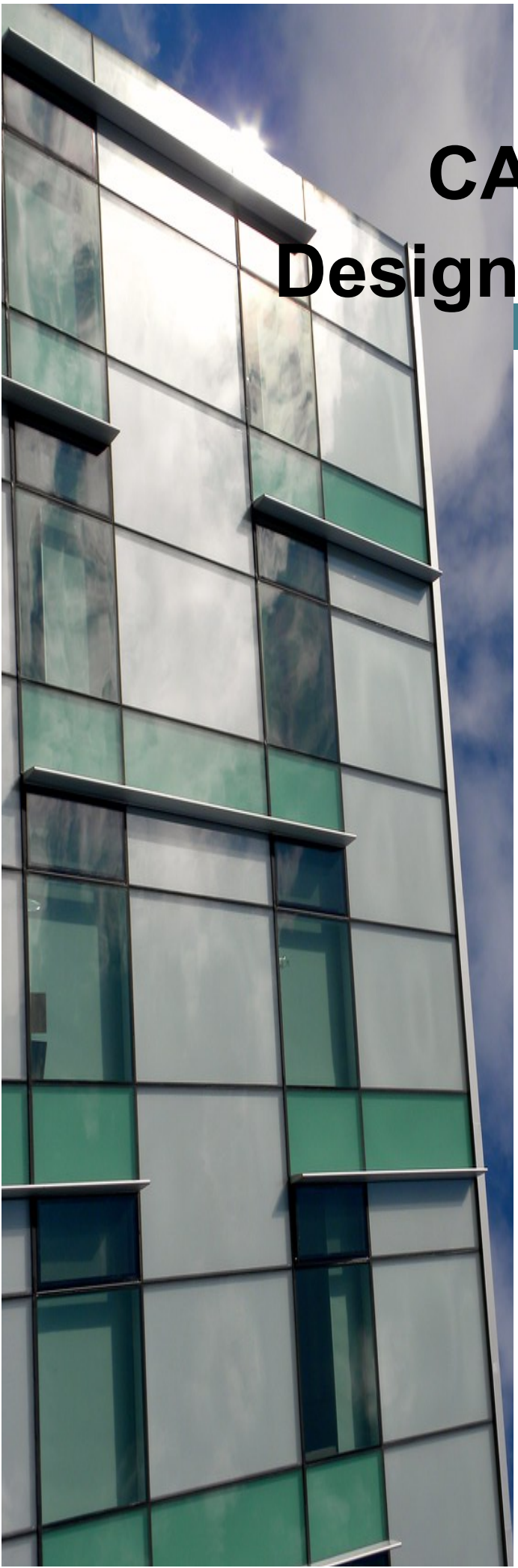


CATKINS - System Design Documentation

By: Mario Liao
Patricia Nagatani
Ilya Kostin
Yangkun Li
Howie Chen
Ran Zi

Table of Contents

Page 1 - Title Page
Page 2 - Table of Contents
Page 3 - 4 - CRC Cards
Page 5 - **6** - Architectural Diagram

CRC Cards

Class Name: Interface - DAO	
Subclasses (if any): List all the subclasses separated by a comma	
Responsibilities ● Create a post ● Create a group	**Collaborators** ● Post ● User ● Group

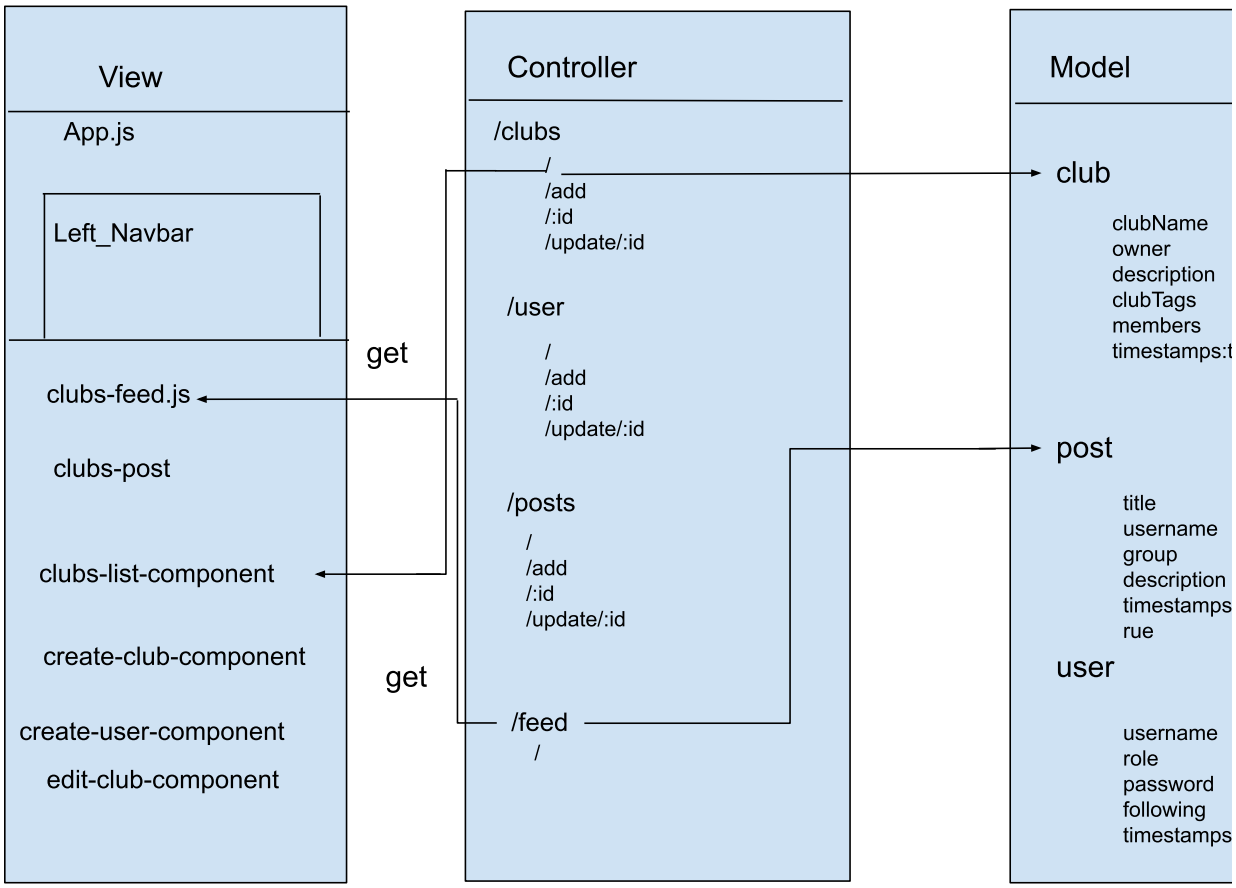
Class Name: User	
Subclasses (if any): Student, Professor, Department	
Responsibilities ● Has an ID ● Knows username/password ● Knows groups that user is following	**Collaborators**

Class Name: Group	
Subclasses (if any): Courses, Clubs	
Responsibilities ● Stores users as members ● Stores its own information ● Will store posts related to that group	**Collaborators** ● User

Class Name: Post	
Parent Class (if any): List the parent class if applicable **Subclasses (if any):** List all the subclasses separated by a comma	
Responsibilities ● Has ID ● Knows posting user ● Knows posting time	**Collaborators** ● User ● Group

Class Name: Comment	
Parent Class (if any): List the parent class if applicable **Subclasses (if any):** List all the subclasses separated by a comma	
Responsibilities ● Has ID ● If it's root ● Replies ● Knows commenting user ● Knows commenting time	**Collaborators** ● User ● Post

Architectural Diagram



App.js - the page with properties applied to all other pages (e.g. navbars)

Left_Navbar - part of the Navbars (
Comments:
will break later for more of them
)

Clubs-feed - component, which renders all posts and does the filtering for them (
Comment: later will be done using another endpoint
)

Clubs-post - component, which takes props of data from the Model and applies styling to it.
Used in Clubs-feed component

Clubs-list-component - gets and renders all clubs in the Model

Create-club-component - creates club by values passed from the user (
Comment: not yet implemented

)

Edit-club-component - allows user to edit chosen club if the user is club owner (
Comment: not yet implemented
)

The backend endpoints do what their name says (except for /:id's ones, they can get and delete data, depending on the request type)

*View interactions with Controller made by components (they are not exactly same as plain functions), because this is how ReactJS is structured: all code is broken by some components.
Controller is made using Node and ExpressJS. It is done according to REST API principles with some ExpressJS specifics (organizing some common routes in separate files so that the code is organized)
Model is implemented using Mongoose Schemas. Controllers can import them and access data, when the View sent the request
Another assumption made about our document is that it is assuming you are using our default database which is:
ATLAS_URI=mongodb+srv://Manager:ManagerPassword@cluster0.mc9b6f1.mongodb.net/?retryWrites=true&w=majority
And it should be located in .env which is inside the setup/backend folder

The summary is: View just sends requests to the Controller, which has access to the Model