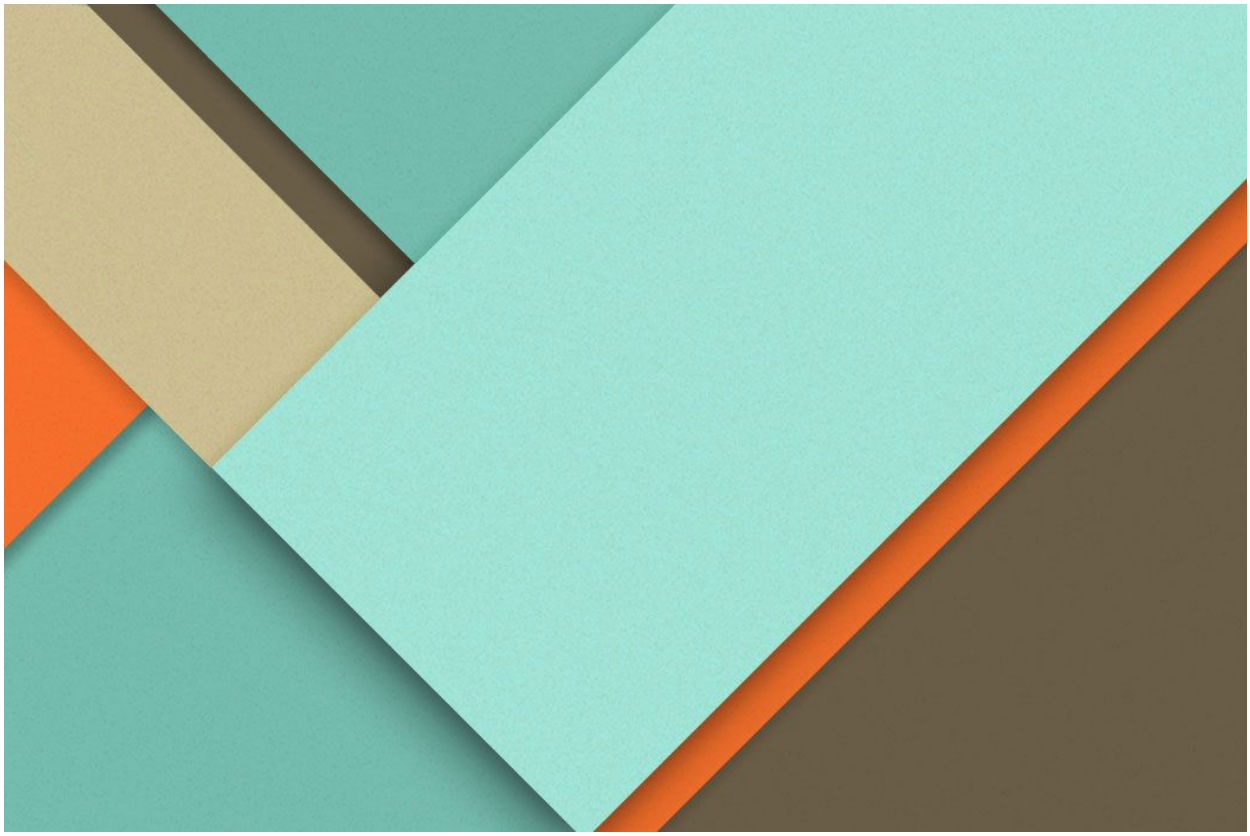


Club Talk Product Guide

COS333 Spring 2018

David Major (Project Leader)

Uri Schwartz

Michael Stambler

Club Talk User Guide

Intro

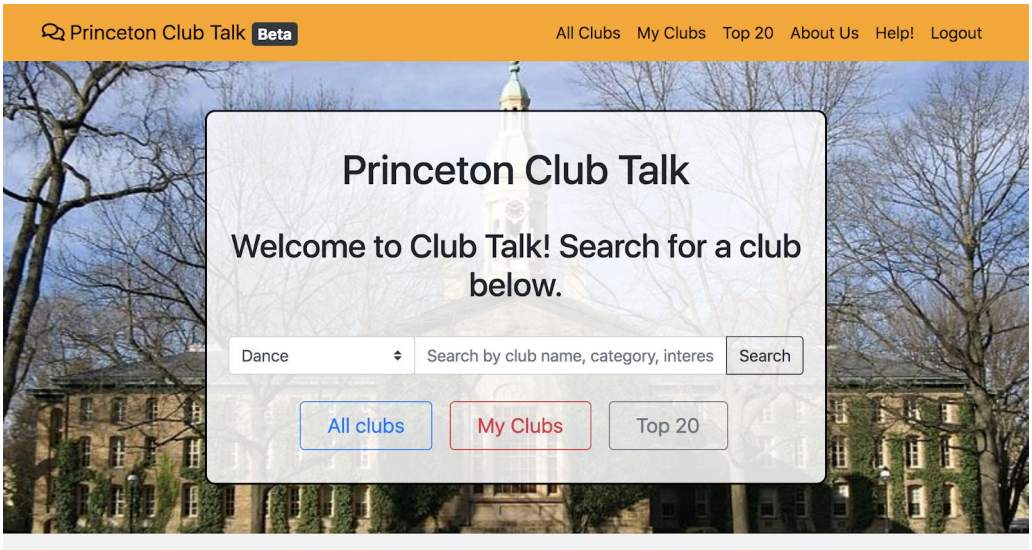
Princeton Club Talk is web application that can be reached at https://clubtalk.herokuapp.com/. Club Talk is a way to explore all of Princeton's clubs in a clear and easy-to-use manner and to read honest, anonymous feedback on clubs. The website is CAS protected and so needs a Princeton netID to access anything past the homepage.

Student Side: Exploring for a Club

Let's imagine you're a first-year at Princeton looking for an extracurricular club to get involved with. You have a lot of different interests (dance for example) and aren't sure what to pursue. You want a club that will be a good place to meet friends, but also that takes its work seriously.

Searching

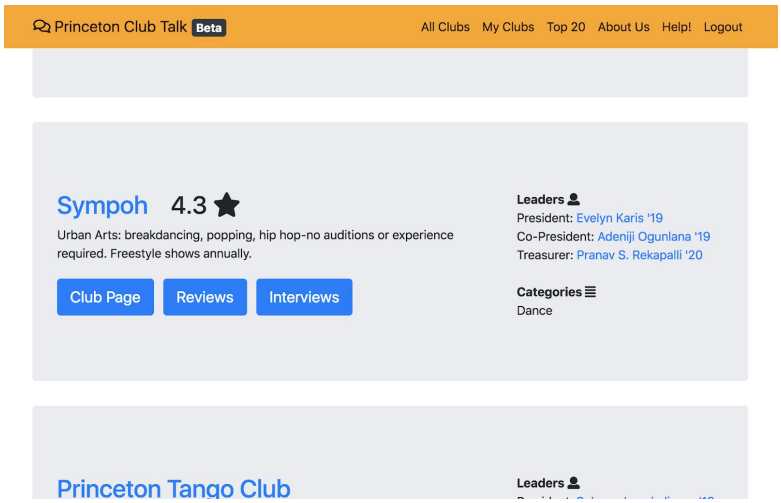
Suppose you enter the website and come up to the homepage as below. You can search for a specific club, certain club features (i.e. "piano" or "writing"), limit by category, and any combination thereof. You can also look at a list of all clubs or the top 20 ranked clubs. Since you are interested in dance, you decide to search by dance.

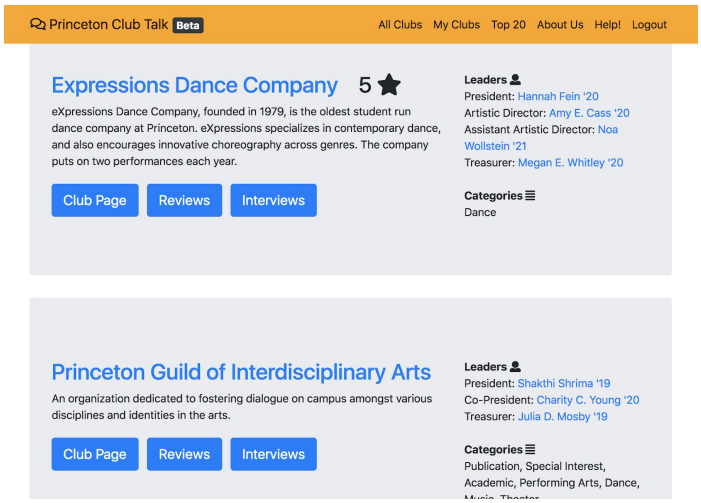


Scrolling through the website

On the search results page and others, clubs are organized into boxes that feature some (but not all) information. In particular, they feature the club name, description, leadership info, and star ranking, which all can be perused quickly while scrolling.

You scroll through clubs and see a group called Sympoh has only 4.3 stars, plus it's breakdancing which you hate. Pass. You also see a club called Expressions with 5 stars and decide to check it out.

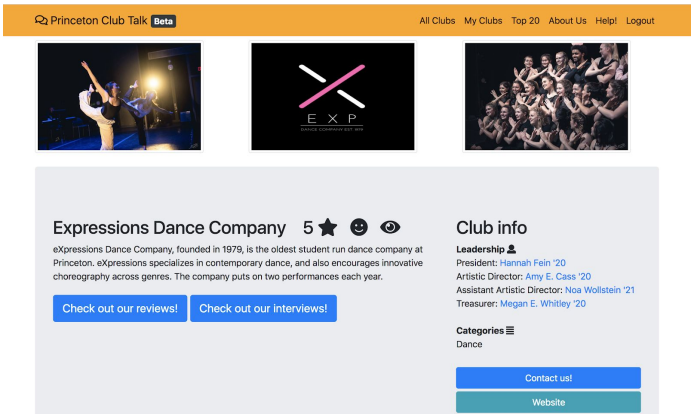


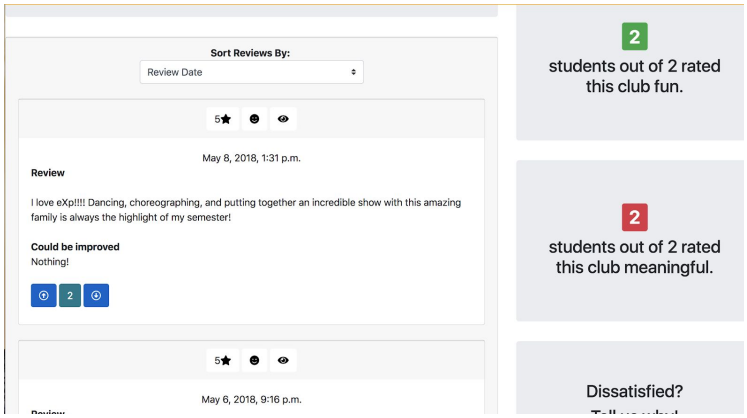


Club Page

Now you're on the Expressions page and get a lot of information. The club has 5 stars and after hovering over the smiley face and eye icons, you see a majority of reviewers found it fun and meaningful. You click on "Check out our reviews" and read more specific feedback on the club. The reviews are initially sorted by date (newest to oldest), but you want the best feedback possible so you decide to sort by review ranking (based on upvotes/downvotes given to each review by other users). A couple of reviews are particularly helpful so you give them an upvote. Some, however, were a bit vague and unclear so you downvote them.

You decide you want to be part of Expressions so you click "Express Interest" on their club page so they'll get your name and NetID.



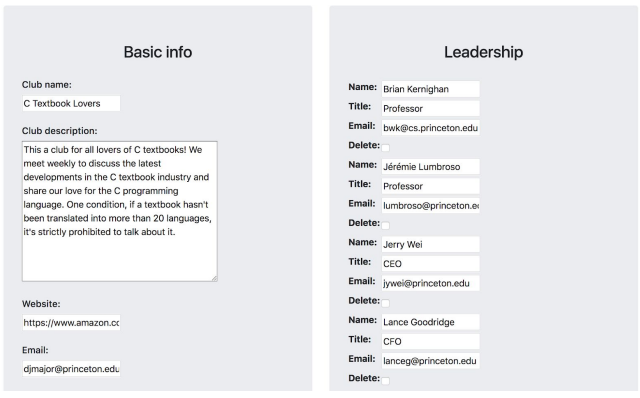


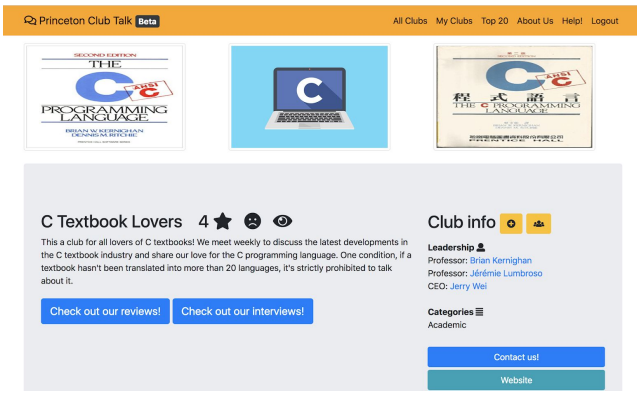
A week later you have an audition with Expressions and come back to Club Talk to see what their auditions are like. You don't need to search for Expressions again; you go straight to "My Clubs" where you can see all the clubs you've expressed interest in, including Expressions. You next check out their interviews to see how you can prepare for their audition, using the "good experience" and "hard" hover-over icons to quickly get a sense of what the audition is like.

Club Side: Editing Your Club Page

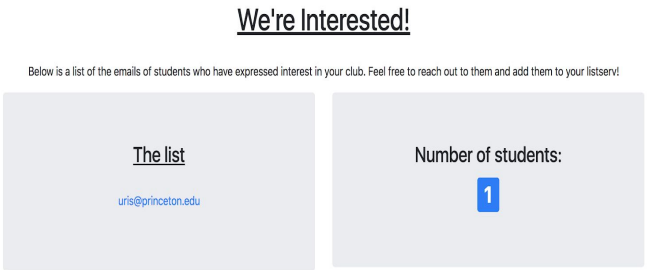
Now imagine you're the leader of a club, say about C textbooks. You go to the "My Clubs" page and see your club there. You can see your club's rating, plus any other clubs you have expressed interest in (try this out! All the professors and TAs have access to the club "C Textbook Lovers").

You click on the page and see it's kind of bland. So you click the "+" button and begin editing your page. You add a longer description, your fellow club leaders (who now can add edits themselves), and a website. The pictures feature gives you three options: three pictures are displayed as boxes, two as mini-banners, and one as a big banner (the exact specs of each are specified on the website). You decide to add three. Voila! Your club page looks much better now.





Next, you want to see who has expressed interest in your club. You click on the people button (that has a clear hover-over to explain its function) and see a list of emails of people you can reach out to. The emails are easy to copy and paste into any email.



Club Talk Developer Guide

Back-end Systems

Princeton Club Talk holds data on over 300 extracurricular clubs. To hold this data, we used a MySQL database, whose tables are found in page/models.py. The tables are described below:

- Club: This table holds basic information of the club with fields such as name, descr (the club's description), email, and website. We also include a many-to-many field with Category tables (described below). Furthermore, this table has fields which keep track of characteristic aggregates that allow us to give more detailed information about the club to our users. These fields include fun_count, meaning_count, positive_count, hard_count, and total_stars. All of this information is acquired from the Reviews and Interviews themselves. Lastly, this table contains three fields for optional photographs (photo1, photo2, and photo3), which allow club leaders to customize their club page by adding pictures.
- Category: This table simply holds the name of a club category. These are tied to club through a many-to-many, and allow searching clubs by filtering by category.
- Leader: This table holds basic information about a club's leader. This information includes the leader's name, title, and email. The email is particularly important as it is used to verify a user's access to edit a club. Furthermore, each leader contains a club field, which ties it to a Club model as a Foreign Key.
- Review: This table holds data of a reviews text, fun rating (as an integer field of 0 or 1), meaningful rating (as an integer field of 0 or 1), stars (as an integer field from 1

to 5), and CBI (the "could be improved" portion of the review). Furthermore, this table holds the rating of the review (as calculated through upvotes and downvotes), the date the review was created, the review's creator (as a Student Foreign Key), and the review's club (as a Club Foreign Key).

- Interview: This table holds data of an interview's text, positive rating (as an integer field of 0 or 1), and hard rating (as an integer field of 0 or 1). Furthermore, this table holds the date the interview was created, the interview's creator (as a Student Foreign Key), and the interview's club (as a Club Foreign Key).
- Student: Lastly we have our Student table, which is essentially how we are implementing our users. Each Student is tied to a user in Django, and is automatically created following a login though CAS. This table also contains the student's netid (which is used for verification of a club leader). Furthermore, the clubs_reviewed, club_interviews_reviewed, and clubs_interested fields are many-to-many fields tied to Clubs, which keep track of clubs that pertain to a student in different ways. Furthermore, the review_upvotes and review_downvotes fields keep track of what Reviews a student has already voted on.

Scraping and Data Collection

In order to acquire all of the data from the ODUS student groups site, we used the BeautifulSoup library to write a scraper (scraper.py) that took all of the data from the ODUS site, and put it in a JSON fixture that was tailored to fit our models. The fixture itself is found the file called clubs.json.

Front-end Systems

Relevant folders: (page/static/* and page/templates/page/*)

Bootstrap

At first, we built our website from scratch and then configured it to work on skeleton, then reconfigured to be based on Bootstrap. Bootstrap was convenient for giving us a lot of design features automatically, such as buttons, page structure, navbar, and much more, and giving the website a clean, professional look that was easily scalable.

We customized the Bootstrap CSS by writing CSS files that overwrite certain aspects: these can be found in business-frontpage.css (for all templates except for the homepage) and homepage.css (CSS specifically for the homepage). For the overall website design, we wrote a base html file (baseB.html) that extends to all pages except for the homepage and the "about us" page.

Communication between the front and back ends

Django views and templates

Most of our business and backend logic was done through Django views and templates. For some of the views, the logic was quite simple, and only involved loading the corresponding template. However, for some of the other ones, the logic was much more complicated. For example, loading a club's page involved loading the club's pictures, figuring out if a majority found the club fun, etc., and other various calculations, before passing that data to the associated templates.

Ajax

Our webpage does not really involve tons of responsive content that needs to be updated asynchronously, so Ajax plays a fairly minor role in our site. Our two main uses of Ajax were for updating upvotes and downvotes and expressing interest in a club, both of which look nicer and work more smoothly as asynchronous requests rather than having to refresh the page.

Hosting services

Heroku

Our website was hosted on Heroku. The database we provisioned on Heroku was a ClearDB MySQL database.

AWS S3

Our static and media files are served through an AWS S3 server, rather than Django's WhiteNoise library.