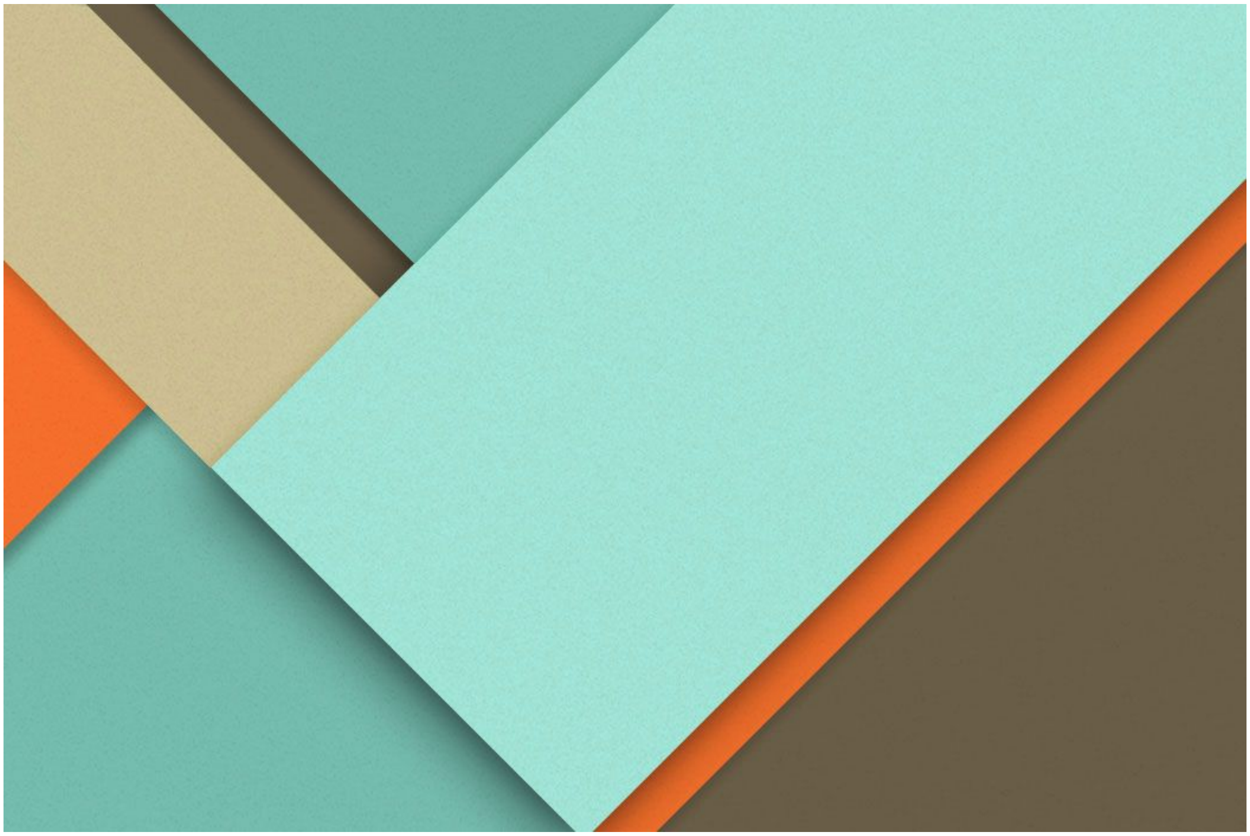

Club Talk Final Report

COS333 Spring 2018

David Major (Project Leader)

Uri Schwartz

Michael Stambler

Planning

Timeline

On our design document, we included a fairly detailed timeline describing our initial goals for how to approach the project. It was slightly difficult to make our timeline, however, since we did not really know how long each step of the development process would take. None of us had ever used any of the technologies and tools that we were planning on using, and therefore had to basically guess what a reasonable timetable would look like.

It is hard to say whether or not we completely stuck to our timeline. For example, scraping data from the ODUS site and putting it in a fixture that was ready to populate our database turned out to be more time consuming than we had initially thought. On the other hand, setting up our github and understanding the intricacies of the Django framework went pretty smoothly and we were able to get going on that fairly quickly.

Overall, we were able to generally stick to our timeline, and continuously found ourselves ahead of schedule. Once we had all of our data, and a basic agreement of how we wanted our site to actually look, we were able to split up responsibilities in a way that allowed us to work quickly and efficiently. Because of the fact that we were decently ahead of our initial goals, we were able to think about stretch goals that we did not even put in our initial design document, and implement them fully. These included giving club leaders the ability to edit their club's page, and add pictures and leaders. We were also able to implement the "Express Interest" feature of our site, which allows students to express interest in a club, and gives that club's leaders the ability to receive that student's netid and add them to the club's listserv.

We were able to deploy a fairly complete version of our app on to Heroku about a week before our demo, which gave us ample time for user testing, feedback, and appropriate tweaking.

Overall Goals

Our goals and even overall vision for our website certainly changed throughout the design process. This was partly a result of new ideas we thought of, but was largely the result of feedback from early users and friends who we discussed our website with from the beginning. We are really indebted to those friends who not only were creative, but helped us understand the user perspective that was harder and harder to grasp alone the more time we spent working on the website.

At first, our website primarily revolved around the anonymous review and interview features of our website. We thought of our website as essentially a version Glassdoor for Princeton extracurricular clubs. We sought to really build the website primarily around these two pillars. Club pages were meant to have some additional information (name, a short description, leader names) to put the club in context, but we actually were quite hesitant to consider other features that we felt would pollute the page and distract from reviews and interviews. We strongly felt (and still do feel) that anonymous information about clubs was both available (i.e. supply) and wanted (demand), particularly when we looked at past and current options such as YikYak and RealTalk Princeton where students have both submitted and sought after this information in droves.

As we thought about our website further, we realized there was so much more potential in what we were doing around Princeton clubs. As we really delved into the needs of the campus community and explored the only available option online to explore clubs, we saw how hard it is for clubs to reach out to students at Princeton. While there have been many websites made to help Princeton students explore courses (all made in COS333), nothing like this has been done for Princeton clubs. The available options for clubs - the once-a-year club fair, emails to public listservs, and posters - are antiquated! We realized there was a true need and desire for a centralized platform where students can learn about clubs and where clubs can also customize their page to suit their particular needs, build their club's reputation and brand, and attract students. We also knew we needed a way for students to express interest in a club through our website, to have the interaction come full circle. In this sense, our ideal website became essentially a virtual club fair.

Adapting Plans

Our plan to build the website matched these changing goals well. Initially, we had planned to focus on implementing reviews and interviews, and we considered adding other features as stretch goals that could be implemented given time. With a lot of work, we finished our initial goals fairly quickly, and so were able to expand into our several reach goals (including comprehensive customization features, "My Clubs" page, and ability to express interest in clubs) that were just then taking concrete shape in our minds.

The way we planned our initial website made it fairly simple to later implement our reach goals. We purposely planned our original goal as revolving around web pages that had the additional features of reviews and interviews. Since we had the essentially blank canvas of club pages, this allowed us to expand on them and add the other features that meshed nicely with the page as the main way of navigating the website.

Design

UI

UI was something we spent a lot of time thinking about. In particular we wanted our website to really revolve around the experience of exploring clubs (the same way you would classes) for information, anonymous feedback, or anything else you were interested in. Much like a real-life club fair, we wanted it to be easy to "move around" and see what may be of interest, focus in on one club to learn more and maybe interact with the leaders, and then keep exploring. In particular, we needed to make sure that all our extra features (reviews and interviews, customizability, rankings, Top 20) would blend into this experience and not hamper the exploration process.

Our means of achieving this was to have the club pages that would be a hub for everything related to a club: information, reviews/interviews, rankings, and expressing interest. The organization of the website really revolved around these club pages. Searching and scrolling through lists of clubs was based on making miniature preview boxes of the club pages that had enough information to potentially entice a student, but not nearly as much as the full club page (where an interested student would then have to go). In addition, the club pages offered previews of reviews and interviews, but to see all of them you'd have to go to their respective sub-pages. We tried very hard to not clutter the club pages and make them easy to use. We did this by putting all the information and means of expressing interest clearly on the top, and then those who really want details on the club can scroll down and go deeper to see reviews and interviews.

In the reviews and interviews section of the website, we tried to strike a balance between making the reviews quick to read (for those who just want to scroll through) but also able to contain detailed information for those interested. For each review, we had users rank the club from 1-5 stars, say whether it is fun, and whether it is meaningful. We thought these three simple questions, displayed as pictures with scroll-overs to make them quickly understandable, could between them contain most of the information a user may want to know about a club. We wanted keep the discourse of the website constructive, so added a "what could be improved" question that is required.

Database

When we started the project, we had initially decided on using a MongoDB database for our app. We were soon to realize that MongoDB is not fully compatible with Django, and that making it work with Django would take a great deal of effort. We decided that this was not worth it, and chose to use a MySQL database. Django initially defaults to using an

SQLite database, so we had to change that, but after the change was made, our database worked very well.

Once our database was set up, we were able to create our models. We first made simple models for Clubs, Categories, and Leaders, that were able to store the data we had scraped from the ODUS site. After creating our JSON fixture, and making sure that our database was populated properly, we added models for Reviews, Interviews, and Students (our users).

As we continued developing our app, and started coming across features we wished to implement, we had to add fields to our models. For example, we added a rating field and a created_at field for each Review, which allowed us to implement a functionality where users can sort reviews by their rating (how many upvotes they have), or simply by the date they were added (created_at).

We had a few debates as to what fields of the Club and Review models should be allowed to be optional and which should not, and so this led to constant modification of these fields. All of these were easy changes, and we believe have led to the best and most logical implementation of our models.

Languages and Tools

Beautiful Soup and Django

We used Python and the Beautiful Soup package to scrape the data we needed to get started. As mentioned above, the only available option to look at clubs was on the ODUS page, which actually made our job much easier, since all the basic data we needed was already online. The ODUS page includes a list of all the clubs, their descriptions, leaders, and links to their websites. It was thus fairly easy to use the Beautiful Soup library to scrape the data. Originally, we simply output the data line by line, because we weren't what we were going to do with it once we read it in. We quickly realized that we needed to put it in a fixture for Django, in order to pre-populate our database. We thus edited the script to output the data into a JSON file that followed the specifications of Django fixtures. It was then easy to get all our data into the database.

We used the Django web framework to do the bulk of our project. There were several reasons we chose to use Django. To some extent, since all of us had no experience with web development, our choice was somewhat arbitrary. However, after getting used to Python through several of the earlier assignments, Django seemed like a good choice since we wouldn't have to learn an entirely new language and syntax, thus making it possible to hit the ground running. Furthermore, some basic research showed that Django's QuerySet abstraction made it easy to make complicated database queries and manipulate large sets of data, which is exactly what we intended to do. Lastly, Django provides some built-in protection, such as protecting from SQL injection attacks, making it easier on our part to not have to worry about issues like that. While we don't know how it would've been with other frameworks, we found that Django's extensive documentation and straight-forward API made it easy to work with!

Front-end

Our front-end design evolved quite a bit throughout the development process. Originally, we made the front-end essentially from scratch, using pieces of random templates from online and testing out different designs we liked. The problem was ultimately that the website didn't have a clean look, and it was difficult to expand the design to new pages. At first, we rebuilt the design on skeleton to support different screen sizes, and then we redesigned again to implement the website using Bootstrap, JQuery, and Ajax. We found Bootstrap extremely helpful as it allowed us to take the core features of our old design and incorporate them into a much cleaner design that was very stable, easy to use, and easy to expand.

We feel we definitely benefited from starting off from scratch. It forced us to understand HTML and CSS on a deeper level that enabled us to customize and adapt Bootstrap CSS later on and avoid the problem of generic design. It also made us think about our website design separate from any templates that may have limited our creativity. The result is the amalgamation of a lot of different ideas at various time - we hope you like the result!

Hosting

We used Heroku to host our website online. Once again, the choice was slightly arbitrary, but slightly informed by talking to past groups and other people. We saw that Heroku was a popular choice and thus chose to use it. It was very simple to understand what the requirements were to host a Django app on Heroku, since their documentation is also very thorough and straightforward. Once we switched to a MySQL database, we used the ClearDB MySQL Heroku add-on, since we were no longer using Heroku's default database. Setting that up through Heroku was quite easy, since all that was needed was to add the database info to our settings.py.

Since Django doesn't have support for serving static files, we originally chose to use WhiteNoise in order to serve our static assets. For a while, this worked perfectly fine and ran smoothly. However, once we started allowing clubs to upload their own images, even after configuring our media settings, we discovered that the images were not showing for some reason. A quick search revealed that when using Heroku's free version, which only allows one dyno that sleeps after 30 minutes, you couldn't really allow users to upload their own images because they weren't persistent. We thus decided to host our static files and media uploads on an AWS S3 server. We first tried to only host the media on the S3 server, but discovered that that too was not possible, and we had to move all our files to the S3 server. However, after that, hosting our static and media files on AWS went relatively well and fixed our problems.

Testing

Self-Testing:

We did a great deal self-testing before sending out our link to real users. We tested our app by searching and filtering different things. We added reviews and interview comments. We upvoted and downvoted reviews. We sorted reviews, used our Top 20 page, and used clubs that we were leaders of in order to test our ability to edit their pages and add photos. We were confident that the overall functionality of our code was in order. Still, we wanted to get fresh eyes onto our page to tell us what they thought of the app.

User Testing:

Before we were even close to having a beta version of our app, we talked to a lot of students about what they thought our app should feature. This is how we were able to decide on including a “Could be Improved” section and to limit users to one review per club.

Once we had an actual product to share, we sent it out to many people and requested their feedback. We were able to get 420 students to access our page. From these students we were able to get 72 reviews, and 10 interview comments posted on our site. We received a lot of feedback on the functionality of our app. A main change that we made was removing the feature that restricted users’ access to the All Reviews and All Interviews pages for clubs until that user had submitted one review himself/herself. We got a lot of feedback telling us that this feature was confusing, and that people felt immediately restricted when getting on the page. Once we removed this restriction, our users seemed to be much more satisfied, and more comfortable browsing the site.

We were also able to get club leaders to edit their club’s pages. There are now a few pages on our site that have been customized by their leaders, which is great to see.

Some groups were not on our website because they are not under ODUS. Once this was brought to our attention, we tested our ability to add a new club to our site through our admin page, and ensure that it works properly.

Onward

Plans for future

We definitely plan to keep our website going after the semester ends. In particular, we are very excited to advertise at the beginning of the semester when it is likely going to get the most use. We are in talks with ODUS to have them advertise the website and pay for server costs. We hope to especially market to incoming first-years who can benefit the most from the website.

In terms of future features, we are happy with the features we have now. We thought a lot about some easy to implement features such as club Q&A and the like, but were concerned these would distract from core features. In the future, we may want to add more.

We plan to add Office of Religious Life groups, Pace Center groups, and club sports in the future. These are not funded by ODUS and so were not on the website we scraped for information, and similar information about them (like we currently have for clubs) would need to be acquired on a group-by-group basis.

Tips for future COS 333 students

Overall, this project was an incredibly useful and educational experience. Some things we learned along the way that would be useful to future groups:

- While the deadline may initially seem far away, it creeps up quite quickly. Work just as hard as you did for the assignments every week, even though there's no Friday deadline to meet.
- No matter what you think are the best UI decisions, your users are actually the ones who determine what you keep or change. Many of our big implementation decisions that we were sure we wanted quickly changed once we got people to actual start using our site.
- It's definitely useful and worthwhile to split front-end vs back-end, etc. However, at least for the first week or so, in order for everyone to be on the same page about broader concepts, it's worthwhile to work on things together so that there's no confusion later about huge components of the app.
- People work at different paces, but progress needs to be made -- distribute the work evenly and find ways to hold your partners accountable, since this is not a project that can be put off.

- Have fun! The project is definitely stressful and lots of work, but it's incredible to finish and have an entire functional app that you can say you made. Plus, you learn an incredible amount while working on it. So enjoy!