Princeton Courses Final Report

Group Reflection

We feel three things were essential to the success of our project: starting early, meeting frequently, and having a shared vision of our product that was responsive to user input. We met several times a week at the beginning of the semester, and the purpose of the meetings evolved as our project did. We first met to discuss potential project ideas and select one, then to debate potential features for our site and how to implement them in an organized and timely way, and later to provide updates to one another about production and continue to discuss further features to implement. As a result, we had a working prototype before our first TA meeting, and this initial momentum carried us through the rest of the semester as we continued to add new features to our site and adapt the design based on feedback. In the beginning, these frequent meetings were crucial in maintaining our quick pace and sticking to our proposed timeline, but as we realized we would meet our key goals and our project was mostly completed, they began to taper off, and the weekly post-TA meetings almost entirely replaced them. It would have been helpful to continue meeting in the same structured way at the end of the semester, as it would have kept us all accountable for our final tasks and ensured that everything was wrapped up before the demo took place.

Within the team, everyone generally worked independently for most tasks. Because the project was divided up between front-end and backend, and from there into sub-tasks and responsibilities, having everyone focus on a different aspect of the project led to faster results and more pieces simultaneously coming together. Each week, a team member would focus on a task and generally be the only person working on it. If anyone needed help or an issue needed to be discussed in between meetings, communication would take place in group chats via Facebook or a GitHub issue was created. While this was generally sufficient communication to get things accomplished, there were several instances when being so independent of one another led to multiple people individually developing code for the same feature and much of it having to be scrapped. These redundancies led to time being wasted and general frustration for all those involved. Not collaborating more closely with one another also led to a modularization of knowledge almost, as the front-end team is not comfortable with the back-end and vice versa. Likely, had there been more communication across sub-groups and individual members, everyone would have been learned more from the project, not to mention there would have been fewer bugs to catch.

As a final note, we are extremely glad that we took the time to learn what real users thought about the site and how they approached using it. While we spent many hours debating on design decisions, features, and functionality, our thoughts on what the site should be weren't always in line with those of other Princeton students. Our personal investment in the site and desire to prioritize what we considered to be most important was kept in check by feedback from users who were seeing it with fresh eyes. Alpha testing with our friends gave us an idea of what new users find intuitive and

confusing, while the open “Feedback” button in beta-testing allowed us to receive requests for features we had overlooked simply because we hadn’t prioritized them in our own experiences.

Usage Statistics

When it comes to assessing how useful students find Princeton Courses, we believe the numbers speak for themselves. Through Google Analytics we can directly measure the impact Princeton Courses has had on the student body. As of May 8th, 2017, Princeton Courses has 1455 users who have spent a cumulative 950 hours browsing the website. This group represents approximately 37% of the students participating in Fall 2017 course selection. Of these users, 60% are return users to Princeton Courses, and 30% visited the site at least five times. 24% of users spent more than one hour on Princeton Courses and 35% spent more than 30 minutes on the website.

The usage statistics also reveal where there remains room for improvement. Only 21% of users have marked any courses as “favorites”. We believe that favoriting courses is one of Princeton Courses’ most powerful features, so it is somewhat disappointing that we weren’t able to educate more users about this tool. Doing so has become a goal for the future.

By these metrics, Princeton Courses has proven to serve a much-needed role for Princeton students when it comes to investigating courses.[1] We are extremely happy to see how much traction the site has gained around campus, and it provides us with an immense sense of satisfaction to hear from other students how much they enjoy using the site. That said, we are committed to spreading Princeton Courses to the rest of the student population. We aim to keep improving by continuing to add the features that students would most like and have often requested, such as Recal integration, so that this site becomes an indispensable tool for all Princeton students.

Backend Reflections

Caterina

This project was my first experience working with team where everyone was so invested in the end product and not just the final grade. As a result, our group meetings where we discussed potential features and did cost-benefit analyses were really constructive, as everyone made sure each idea they pitched was practical, implementable, and worth doing since the rest of the team had to be convinced in order for the idea to get the go-ahead. It was eye-opening to see what features other people value and to face that not all of my ideas are necessarily good ones, that they may be impractical, not very useful to our end-purpose, or both.

Before this project, I thought I understood the value of testing, but I didn’t really. It wasn’t until the stakes were higher than just an assignment grade that I really understood how vital it is that an algorithm give back correct results 100% of the time. For our site, if a function gives back the wrong result, not only is the product not up to standards, but it may result in users not returning since they feel it is unreliable. Testing did indeed take just as long, if not way longer, than writing the actual code,

[1] Jessica Zheng ‘19, of the competing ReCourse team, is the user who has logged the most sessions: 55. Jessica has spent 2h48m on our website. We hope that Jessica has found our website useful in her course selection decisions.

but running through the cases line by line gives a peace of mind that makes the screen-induced headaches worth the while. (Although I also learned that it really is best to leave the testing to the "user". They will always find something you didn't catch.) Considering what the user might do, or the quality and completeness of data we would be working with was also a great learning experience. How much can we assume that the variables we'll be accessing from the database will be in the format we want? Or even exist? What if our user wants to do X but doesn't have Y, what do we return?

In the end, I wish we could have done more with the actual written comments for each course, and if we'd had more time (and much more data) I would have liked to really dive into this. I felt at the start of our project that there was a lot of untapped potential there, but ultimately trying to figure out a way to standardize responses and analyze them (without hard-coding anything or going through the responses one by one ourselves) in order to glean a small shred of information wasn't worth it this time around. At first, I didn't want to admit this and spent weeks trying out NLP packages and testing different strategies, and only begrudgingly came to the conclusion myself after my teammates delicately implied in one of our fantastic team meetings that this would be a lot of work for not particularly useful or exciting results.

Lastly, I came into this project having never done any CS-related work outside of Princeton classes, and so the prospect of dealing with so many new aspects of computer science (databases, web design, scraping, etc.) was really daunting. While I do believe I am more innately interested in algorithms and machine learning than other areas of computer science, and feel more comfortable with tasks involving those things, I also recognize I let myself be intimidated by just how much I didn't know (and how much others seemed to know) and stuck to working within my comfort zone. As a result, I don't think I learned as much as I could have, nor do I think I helped my team as much as I could have. Honestly, whenever I wasn't petrified by the magnitude of all I didn't know and actually considered trying something new, I often worried I would just be getting in the way of everyone else who did know what they were doing. In the end, I limited my own growth because I know all my team members would have gladly helped me if I had asked for help. I really love our project and how it turned out, and I'm very proud of the end result and all the hours that went into it, I just wish I myself could have made a larger impact.

Sebastian

I am thrilled with how this project has unfolded, both in terms of our impact on fellow Princeton students and what I have learnt.

Impact on Princeton Students

As our usage statistics show, our work benefitted hundreds of Princeton students. I derive great satisfaction from knowing that my work has helped my peers. Walking through campus and overhearing people recommending Princeton Courses to their friends is delightful. This experience has taught me how much I value having a tangible impact on people. I hope that this will guide my in my career decisions to work on projects where I can have a large positive and quantifiable impact on real users.

Technical Learnings

I entered this project with some fairly significant front-end development experience. I had dabbled in the back-end, and I saw this as my chance to learn more. I now feel very competent as a back-end developer for Node.js-based apps. My JavaScript skills have improved: my code now feels much more elegant, whereas before it was functional but sloppy. I've learnt how to build a full web app: serving page requests, a REST-style web API, integrating a database, and more. These skills position me well for pursuing future web projects. This summer, off the back of this project, I will be working on the team at MongoDB that builds the "driver" that allows Node.js apps to interact with MongoDB.

Collaboration

When the Princeton Courses group first came together, it fairly quickly became apparent that I had the most experience building web apps. This meant that, at the beginning of the project, I was often helping my fellow group members learn how to interact with and extend the web app. I really enjoyed these beginning weeks because it was a chance for me to practice clearly explaining how a complicated system works. Before long, the whole group was running quickly and each person was exhibiting mastery of their specific area of responsibility.

I would have liked to work more closely with Caterina. Normally I was working on the core of the back-end while Cat was working on self-contained projects (like the course clash detection algorithm). If we had collaborated more closely our code quality might have been better and we might have had fewer bugs (this is the philosophy behind pair-programming).

I found challenging the way that our flat organisational structure and loose specification led people to independently make design decisions that required other people to put in considerable effort to implement. For example, sometimes, with little consultation the front-end team would move forwards on implementing a particular search feature that was challenging to implement on the backend. Also, because GitHub's "Issues" feature functioned both as our "ideas board" and "to-do list", sometimes it was unclear whether a posted issue was a task that someone felt strongly needed to be implemented or whether an issue was simply an idea someone's friend had suggested for further consultation. I think that more communication in this area would have been helpful.

Front-End Reflections

As a whole, we gained a lot of experience in front-end implementation. One of largest issues we faced was checking in with one another and talking through design decisions before we individually tested them. Although this allowed for the project to develop in interesting ways, multiple people were working on the same piece from time to time without knowing that another person was also working on it. Because of this, some major pieces of code we individually wrote ended up being repetitive and were not used. If we were to work on this project again, we would put more emphasis on communication within ourselves in order to save both time and effort.

Bensu

I started working on the project thinking I had enough front end experience to be comfortable, but I was really wrong. I learned so much over the course of the semester in terms of HTML/CSS/JS, and about how to work on a single big project in a group of so many talented coders. Initially, my code was really sloppy, but as our project got longer and more complex in time, I understood the value of modularization. I was always scared of using GitHub for collaboration in the past, but this project made me feel so much more comfortable with source control.

Looking into the future, I would like to learn more about back-end development, since I was often clueless about how the back-end was put together. Furthermore, this project made me realize I would really enjoy being a Software Developer, more so than following a theory track in Computer Science. Often times I found myself procrastinating my other school work, courses, and obligations by working on our project and building new features.

Kara

As group leader, I feel that I could have done a better job checking in with everyone individually. This was a very interesting experience to be working with a group of peers who were all entirely competent and extremely driven, and this led to me taking a more laissez-faire leadership approach within the structure of our group. However, I've learned that a group can always use more overarching guidance while still allowing individual freedom to explore new options for the project.

With regard to my working style, I've learned that I personally need space and time to develop my thoughts. Throughout this semester, I've found that I need to break up my time into small chunks of coding and designing in order to think through the problem at hand without being physically in front of my computer. Working in a group, I've found that I work best if I vocalize a weekly realistic goal for myself and working independently to develop this goal, later coming back to the group to develop a plan for moving forward.

Mel

I came into this with pretty much zilch in terms of developing experience and I feel very fortunate to have been able to work alongside a group of motivated and capable teammates. Thanks to our relatively vanilla framework on the frontend (jQuery and Bootstrap are intuitive and have great documentation) it was easy to get started, although it becomes harder to maintain structure as more code is written. (This is in contrast to React, which I also learnt a bit about in another project this semester, but oh my god the learning curve is ridiculous for someone just starting.) Another thing that proved immensely helpful for front-end design was the flexbox - something capable of solving a vast array of vertical and horizontal positioning problems. Funnily enough, I was reluctant to learn it at first (HTML/CSS/JS was enough of a mouthful already) but after seeing many off-hand references to it here and there as I scrolled through Stack Overflow, I decided that it was time to launch a little investigation. Lo and behold, I became enlightened by the elegance of flexbox and have never turned back since!

One of the things I found most useful while working on the front end was to work on the visual appearance of the code in isolation. Using the online development environment CodePen, I could try out any new ideas quickly (and discreetly) and without fear that I would break the functionality of the actual site. Even though I don't think my teammates ever really succumbed to my frequent (but unpaid) advertising of CodePen, I found it extremely convenient to use because of the way the code (all 3 of HTML/CSS/JS) is displayed right next to the rendered page. In this way, it was possible to do a complete makeover of the front-end (which was necessary a few times) with relatively few design bugs, as these could be ironed out before being implemented.