

Final Report

ProgressReport

By: Marisa Chow, Isabelle Ingato, Allison Chang, Andrew Kim

10th May, 2016



Introduction


Our initial goal coming into this project was to create a platform to better serve students in understanding their academic progress, as well as a tool to allow students to accomplish their academic goals, while simultaneously exploring areas that they may not have been aware of initially. Essentially, we wanted our platform to be both a way for students to see how they are doing in areas that they already know they are interested in, as well as expose them to new classes, certificates and majors to promote exploration of different areas. Now, a whole semester later, we have accomplished our goal with ProgressReport and are very satisfied with the work that we have all done to make our project a success.

Although our project seemed challenging initially, it wasn't until later on (despite Professor Moretti's well intentioned warning), that we realized exactly how big of a challenge this project would be. However, despite the bumps that we've faced in the development process, we are happy to present to you our final product and we hope that you enjoy it!

Reflections

I. Planning

- A. We started out with a fairly detailed timeline where each individual's responsibilities were planned out, along with where we wanted the project to be during each stage of the prototype, alpha and beta stages. While there were some features that we were not able to implement due to time constraints, we managed to implement all of the core features that we wanted the final product to have by the deadline. While we were ambitious in our initial goal, we were aware from the beginning of the risks and planned accordingly to prioritize certain features over others to ensure successful completion our product.
- B. One of the milestones that we missed was the completion of collection of data for the database for all majors and certificates because we had to manually gather the data into text files, rather than simply being able to scrape the information off of departmental web-pages like we initially hoped to be able to do so. Furthermore, for some of the departments, their




major requirements were quite vague and difficult to code in a hard format, leading to difficulties in representing them in the database. Despite the setback with the data gathering and database population, we were able to continue working on the rest of our project by making use of the data that we did have in the databases to start building the rest of the features, ensuring that we did not fall behind schedule.

II. Design Decisions

A. User Interface Design

1. We purposefully tried to keep the user interface as clean and relatively simple as possible, making this essentially a one-page app. (This is more challenging than one might think, by the way.) Our reasoning was that understanding academic progress is difficult enough and part of the actual goal of our project is to make everything simpler. We tried to strike a balance between giving the user explicit instructions and letting the user figure out interesting features (like the feature which allows you to see all the requirements a course fulfills when you click on its name) of our project on their own out of their own curiosity and intuition (which is something we as users of other products had found we tended to enjoy). In the future, however, we might consider adding more tooltips so that users can become aware of these features quicker. A sometimes frustrating but interesting surprise was that almost every time we added a new feature or component to the interface, we suddenly realized another feature that would have to be put in place as well to allow for all types of users (not just the average user). For example, we only allowed the user to upload transcripts for course input for the first five weeks, but then remembered that not everyone would want to give us that information, so we had to provide another option to that kind of user (manual input). The central theme or goal going into user interface design was to highlight the comparisons between majors (based on GPA, number of courses completed, etc.) and we really kept this goal in mind throughout (by sorting by these features, etc.) which was great and speaks to our team's shared mindset.

B. Database Design


- 
1. One of the initial design decisions that we had to make was what type of database we wanted to use, MongoDB or PostgreSQL. With PostgreSQL, each table is required to have a set schema, which limited how we could represent each major and certificate in the database, whereas in MongoDB, there are no schemas. We were initially concerned that a rigid schema would prevent us from being able to do the necessary operations and comparisons that we wanted to do between courses and their requirements.
 2. Ultimately, we decided to use PostgreSQL, because it was easier to learn and to integrate with our mid-tier and front-end systems, and because we found a way to do the necessary operations for our project to work by adding an extra table in the database that mapped courses to the majors and certificates they fulfilled. This table, along with a table that listed all the courses that were either prerequisites or electives for a major in different tracks allowed us to determine which courses fulfilled which majors, as well as what track it fulfilled within that specific major.

C. Course Suggestion Algorithm

1. When we first came up with the idea of building a course suggestion algorithm, we had a million ideas flying around - let's use their grades in courses to suggest similar courses! Let's use easypce reviews to suggest courses! While these were all potentially good ideas, they missed the point of what we were trying to accomplish *uniquely* with our product. When we got to actually developing the algorithm, we realized that we didn't want to try to tell people they should only take courses, for example, that they had done well in (as Professor Moretti got us thinking: Would it be ethical or right for us to do so?). Instead, we focused on an algorithm that basically informed students of the courses that fulfilled (with greatest overlap) the most tracks in the most majors they were interested in, and this fused well with the rest of our product. In the future, we could ask the user to weight or rank their interest in majors on their board to improve this feature even more.

III. Languages/Systems

A. Heroku/Flask - Front-End

- 
1. We decided to work with Flask because it was a lightweight frontend Python framework that is easy to start working in. Flask contained all the functionality that we needed for our app, but it was also simple enough for our frontend developers to learn and start working in after a few hours. We could also test easily on localhost with Flask's built in debugger. Additionally, one of our members had previous experience working in Flask.

B. Python - Midtier


1. Having worked in Python for most of the assignments throughout the semester (and also since Python is compatible with Flask and Heroku), we decided to use it as our core language. In retrospect, this was a great choice not only because we didn't have to deal with a big learning curve but because it also gave us access to packages like pdfquery and allowed us to harness what we learned about working with regular expressions in Python from class for parts of this project.

C. PostgreSQL - Back-End

1. Most of us had little to no experience with PostgreSQL, but after finding some tutorials online, we were able to mostly figure out what needed to be done in setting up the database itself, as well as inputting data into the table. We tried to come up with a database schema that would allow for fast and simple queries. We considered a many-to-many scheme before settling on our current version. We believe we mostly achieved what we set out to do with our database. The main flaw in our current schema is that it does not take advantage of more advanced concepts than we were not familiar with or uncomfortable using (such as foreign keys and joins), so we end up using more space than necessary.
2. Given more time, we would like to be able to explore these more advanced concepts in database structure and design to make our database tables and schemas more space efficient.

D. GitHub - Version Control

1. We made use of the wonderful tool that GitHub is for version control in our project. This made it much easier for us to pool our code and data together with the least amount of confusion possible. We highly



recommend any future COS 333 students to use GitHub for version control, as it makes the collaboration process much simpler. It's a great tool to use!


IV. Testing

A. Self-Testing

1. Between the four of us, we only had access to about six transcripts (all with similar courses from similar areas since we're all CS or ELE majors!). Therefore it was important to move into beta testing quickly so that we could evaluate potential bugs related to all different kinds of inputs.
2. However, we were able to test most of our core functionalities, such as GPA calculation, transcript parsing, accuracy of track information, and the course suggestion algorithm before releasing our product to our beta testers. Throughout the creation of our product, we kept careful tabs on whether or not the addition of a new functionality might affect the rest of our product, allowing us to ensure that we wouldn't break anything when adding new features.

B. Student Beta Testers

1. We found beta testers among our classmates, and it appeared that interest in our project was quite high. Many of our classmates expressed that they would like to use our project to help them figure out certificates that they could complete, or unexpected majors that they could concentrate in. The testing process was relatively simple, we simply allowed them to open the website, upload their transcript and play around with the different features. We were able to catch several bugs in our product that we didn't notice before because of this, such as: taking into account A+'s in an individual's transcript, manual uploading of courses, and some inaccurate data that we were not aware of
2. Overall, the beta testing experience was really interesting, as it allowed us not only to debug our code, but also to see how students would react to the functionality of the site, hear any possible suggestions they would have for improvement, as well as determine if the website was intuitive enough for first-time users to be able to figure out most of the functionality on their own. We were able to



see what parts of the website was confusing to students, and which parts seemed to come to them naturally. For example, CAS login is almost second nature to Princeton students, so they had no problem with the logging in process, but the decryption of the transcript could be confusing at times, which is why we added instructions on how to decrypt in our actual site.

3. It was unexpected how interested our peers were in our project, which was very encouraging to know that we created something that our peers would want to use to improve their Princeton academic experience.

Final Thoughts


This project idea was at times both great and frustrating. While it was original and, if executed correctly, would be able to really serve an actual need that we as students could relate to, it also required an enormous amount of manual labor for data collection on top of the software development. We are really proud of what we were able to accomplish but do wonder if we could have accomplished even more on the software side if we had gone for a less manual-labor-intensive project idea.

We think the milestones were achieved almost always on time, and we think a big part of this was getting the transcript parsing (a potential bottleneck) completed over spring break. We learned so much throughout this project with most of us having never worked with databases or Flask essentially at all before. One surprise for us was how easy it can be to go a bit crazy over trying to strike the perfect balance between providing flexibility in the product for the user and not overloading the user with choice and information; we honestly did not realize how incredibly hard this is to do until working on this project and, for example, considering the many different ways users might want to input or remove elements of their course history and progress.

We would love to have more time on the user interface and middle tier. By the end of the project, we had done a lot of work building up the database and the backend that we actually could have harnessed these even further to do things like show whether you are improving or declining in GPA in a major every time you re-upload a new transcript. It would be nice to add d3 visualizations and make the page more responsive as well. We would also maybe like to reconfigure the database slightly to take advantage of things we were not aware of at the start of this project, such as foreign keys and joins. We're happy to have gotten to work on this project with this team.

As for future COS 333 students, the best advice that we feel that we can give you is that you should work ahead of time on all aspects of the project. Coming into the first few weeks of the year with an idea of what our project was going to be meant that we were able to get our project approved much earlier on with Professor Moretti, allowing us to avoid the rush later on in the semester to meet with him. Furthermore, it meant that we had a better idea of the roadblocks that might face us earlier in time, from the advice that he gave to us when we first gave our project proposal.

Also, don't forget to try new things! While it may seem daunting to have to create a whole product from (mostly) scratch, it is also incredibly rewarding. There are many



resources out there to help you with your project, whether it is the course TA's or just guides on the internet that can jumpstart you into a new language or platform, so don't be afraid to ask questions and to try new things!

