Progress Report

March 15th, 2016

Team Leader: Marisa Chow

Team Members:

Allison Chang, archang@princeton.edu

Marisa Chow, mlchow@princeton.edu

Isabelle Ingato, iingato@princeton.edu

Andrew Kim, ak11@princeton.edu

OVERVIEW

Our project is a centralized, personalized portal for students to see their major, certificate, and degree progress. We hope to be able to provide students with an easy, intuitive way to see a "big picture" view of their degree. In order to implement this, we will show students how far through their degree they are as well as inform them of the requirements that they still need to finish and what requirements they have already fulfilled, thus giving them a clearer idea of which courses they need to graduate. We will not only provide this progress for majors, but also for certificate progress and possibly distribution requirements. Furthermore, for students who have not yet chosen their majors or certificates, we will provide an easy way to compare and contrast the course requirements for different majors and certificates. Additionally, we will recommend different majors and certificates to students given the courses that they have already completed.

This is meant to be a tool for students to see and understand their academic progress in an intuitive and user-friendly way and at *any* time during their tenure at Princeton, which we find is not accomplished as well through the existing TigerHub site. We hope that this tool will be able to allow students to better plan out their future semesters at Princeton by providing students with an easy way to find and organize degree requirements, which can be difficult given that degree requirements on the course

registrar page are often not updated, and students currently have to hunt through various departmental websites to find the necessary degree requirements.

REQUIREMENTS AND TARGET AUDIENCES

As aforementioned, our target audience is Princeton students and our product is meant to be an alternative to the degree progress system in TigerHub. Currently, the TigerHub degree progress tool makes it difficult for students to see overall how much of their degree they have completed and does not offer clear guidelines regarding what requirements they still have yet to complete (and how they can complete them). We hope to address this by creating a better visualization of a student's degree progress as well as by providing clear guidelines for the different requirements for different majors. Furthermore, there is currently no easy way for students to compare the different requirements for different majors. Thus, for students still deciding what majors or certificates that they want to pursue, we will be providing an easy way to access and compare all these requirements in one location, rather than having to search through various departmental pages by themselves.

Additionally, there are many certificates or majors of which students may not even be aware. These include certificates which students might be able to receive with only a bit more work outside of what they have already completed. We hope to be able to suggest these certificates for which they might have 'inadvertently' completed a large part of already. We will be able to suggest certain courses that students may use to complete their requirements, reducing the amount of time that students have to spend on finding classes that will fulfill the requirements they are still missing. All in all, we hope to make student's degree/certificate progress more streamlined by reducing the amount of time they need to spend figuring out what requirements they need to complete and what classes can fulfill their missing requirements.

FUNCTIONALITY

After the user has signed up and uploaded at least some of their course history, we take them to their main page where they can see their progress in different programs side by side. By entering a search term or choosing a previously "saved" major or certificate from the sidebar, they can decide what programs to view their progress in at a given time. They can also get a closer view of what a single course is fulfilling or ask for a suggested set of courses to be able to fulfill a major and set of certificates by

graduation.

Use Case #1: So… what are my options here?

Jim the Ambitious Freshman has interests in Astrophysics, Classics, EEB, and Computer Science too (after all, all the cool kids on campus are majoring in it). Near the end of the year, Jim hears about our app, signs up immediately, and then manually inputs into our system the nine courses he has taken thus far. We show Jim his progress in the majors where he has taken at least a prerequisite or departmental as well as his progress towards distribution requirements. Jim types in a search term and looks for a few other majors in our system, considering their requirements.

Use Case #2: Now let's get certified!

Jim the Weary Sophomore has scaled back his interests a bit and is pretty decided on Computer Science as a major. Still, he wants to pursue his many other interests via elective courses and still have something to show for it - and that means certificates! Jim uploads his transcript to update his course history in our system. He also manually inputs his AP Credit which he forgot to include last time. We show his progress in the certificates in which he has completed at least a course. We also suggest additional certificates which have the most overlap with his major requirements.

Use Case #3: But what is the worth of this course anyway?

Jim the Exhausted Junior is in the middle of an existential crisis! What *is* this course in [esoteric topic] really doing for him anyway? Jim signs into our app and enters the title of the course as a search term or clicks on it from its listing under the progress section for one of his certificates. We show Jim the major, certificates, and/or distribution requirements which the course is helping to earn him. Jim is relieved to find that the course is an SA *and* can be double counted for *two* of his certificates! His day a little brightened, he continues work on his IW.

Use Case #4: How do I get all of this done by graduation?!?!

At the beginning of the year, Jim the Psyched Senior wants only to finish up his CS degree and two certificates in the minimal number of courses possible. Jim signs into our app, confirms his choice of major and certificates, and lets us suggest a set of courses which represent the minimum number needed to complete all his remaining requirements.

DESIGN

1. User Interface

   We will use standard HTML, CSS, and JavaScript on the front-end and probably (Twitter) Bootstrap to improve the look of our webpage. We will also be using d3 to enhance our visualizations of a student's progress, offering different graphics to represent how courses add up to a degree. We have considered pie graphs, bar charts, etc. as possible options and intend to look into this more. To allow a user to separate what progress areas they view at a time, we may offer four main tabs (an "All" view, a "Major" progress view, a "Certificates" progress view, and a "Distribution Requirements" view) along with a sidebar with a search bar and "favorites" section. We hope for our application to be as interactive as possible and will likely use jQuery plugins to accomplish this by, for example, making page elements (such as courses) draggable and droppable so the user can even add or remove potential courses to and from their progress tabs.

2. Process

   We will be using Python in the middle-tier of our application. Not only is it compatible with Heroku and will interact well with our front-end (through frameworks like Flask), but it also has the types of libraries we need for this project, such as pdfquery and BeautifulSoup, which we use in extraction and parsing of information from uploaded transcripts. We also have access to the Python scripts for implementing CAS login, which is helpful.

   Besides generally assisting in transfer of information between the front-end and the user data table as well as triggering queries of the major/certificate table in the database on searches for requirements by the user, one of the most important aspects of the middle-tier of our application will be the major/certificate/course suggestion engine. In the first iteration of this suggestion engine, we will be simply calculating and rating how much of each major - of which there are a limited, i.e., computationally feasible, number - a student has fulfilled and then suggesting the highest *n* rated majors. We intend to hopefully take further steps with more complex algorithms once we are able to accept more user information such as a user's ratings of the courses they took. The middle-tier will also be responsible for suggesting (the minimal

number of) courses to complete a chosen degree by graduation. This is a (fairly simple) optimization problem especially as most undergraduates only choose one major and one certificate.

3. Backend/Data Management

   For the backend, we plan to use a postgreSQL database. We will have two main tables, one for user data and one for major/certificate information. In the user data database, we will include the user's name, netid, school year, their major, their certificates, the courses they have taken thus far, as well as their grades in those courses (should they choose to share them), interested concentrations, interested certificates, as well as the number of pdfs that they have completed.

   For the major/certificate database, we will maintain for each major and certificate, the number of courses required for the certificate, which classes are prerequisites, which classes are in which sub-requirement group (e.x. in the computer science department, you need to have two systems, two theory and two applications classes, so we will store the classes that fall into these different categories into three groups). For easy querying and search, it may be helpful to store a mapping from every course to *all* requirements in *all* areas it fulfills. We also may potentially create a platform for individuals to give feedback or tips on different majors, which will require the storage of comments. We also might store a ratings field for different course ratings.

TIMELINE

	Marisa	Allison	Isabelle	Andrew	ALL
March 7th - 13th	Design Doc: Section 6 Section 4	Design Doc: Section 1 Section 2 Section 4 Section 5	Design Doc: Section 3 Section 4	Design Doc: Section 4	Design Doc
March 14th - 20th Design Doc Due (March 14th)	Set up basic website with Flask/Heroku/Bootstrap Basic UI display	Getting Data for database	Learn about scraping PDF for transcripts Project Status website CAS Login set up	Getting Data for database	
March 21st - 27th PROJECT STATUS	Working site with db input; database editing; dynamic UI display	Have a minimal database set up for some majors Determine how	Grab data from user's transcript and update in database; send course /progress information to front-end (pipeline)	Have a minimal database set up for some majors Determine how to	Prototype Requirements: Basic Login flow - User account input and data creation

WEBSITE DUE		to get data to midtier		get data to midtier	Display progress for some majors (engineering TA Meetings
March 28th - April 3rd PROTOTYPE DUE	CAS login; format progress from db; database post	Work on database integration Side by side panel of different majors	Efficiently grab data from major/certificate table in database for easy search	Work on database integration	Progress to alpha Major progress Certificate progress work on elevator speech
April 4th - April 10th	Suggestion UI, continue working on frontend; add page for comments?	At the very very very latest, complete database Commenting system for database	Suggestion algorithm for majors - ideas: simple majority rule-based, clustering, collaborative filtering if want to base off other users' data	At the very latest, complete database	Progress to alpha courses to major courses to certificate work on elevator speech
April 11th - 17th	Integrate suggestion algorithm UI	Help with user interface and algorithm, testing database	Suggestion algorithm for certificates in conjunction with majors; bolster suggestion algorithm to be robust to missing data, etc.	Help with user interface and algorithm, testing database	Progress to alpha Alpha testing work on elevator speech
April 18th - April 24th ALPHA DUE	Add course review links (popups ideally), testing	Help other team members Highest rated courses for your major from EASYPCE	Incorporate additional features such as user's course star ratings, grades in a department, easypce ratings, time limitations (i.e. are they a freshman or senior?) etc. into major/certificate suggestion engine; course suggestion algorithm = optimization problem	Help other team members	Be done with implementation
April 25th - May 1st BETA DUE	Test whole product	Beta Testing	help with UI Beta Testing	Beta Testing	Practice DEMO Beta Testing Major/Minor comparison? EasyPCE integration - highest rated courses for your major? How many pdfs?
May 2nd - 8th	DEMO WEEK Complete Final Progress Report				
May 10th	DEAN'S DATE Submit Final Documentation				

RISKS AND OUTCOMES

As all of us have limited experience with web application programming, we anticipate some difficulty learning Heroku, database management, and linking between tiers. We have all programmed using Python, but programming standalone assignments in Python is different from programming Python for web, so we also anticipate spending time to learn web Python with Flask.

Because a concentration or certificate requirement may be unusual and non-modularizable with other requirements, we may need to hard-code portions of our project to deal with special edge cases. For example, the certificate for the Program in Dance requires 20 hours of technical work in assisting the dance program's productions. Hard-coding will take time to implement. Writing code to deal with each concentration's and certificate's particular requirements may be the most complex part of the project.

This project can be completed in increments, so if we run into unexpected difficulties, we may simply not complete as many features as we originally intended to. The core project comprises CAS authentication, parsing input of classes taken, and displaying concentration/certificate progress (which requires databases of department requirements and user data). Once we complete those, we can add features such as easyPCE integration, a course suggestion algorithm, concentration/certificate reviews, time limitations, etc., depending on our progress and what we think will be most feasible in the time we have left.

PROJECT STATUS WEBSITE

http://mlchow.github.io/cos333-project/