ISS 495S: Research Capstone
Spring 2019

A Better What-If

Professor Szabo

Amani Ahmed, Ariel Burde, Peter Ciporin,
Sierra Lorenzini, Stone Matthers, Gaby Salvatore, Rachel Settle

Table of Contents

I: Planning and Initial Design

Why This Project?

This project came to be because we all collectively realized we, and many of our friends, had encountered frustrations while planning courses to fulfill graduation requirements. The process as it currently works can be confusing and there are many steps at which things can go wrong or be left out so that a student may not realize until it is too late that one of their requirements hasn't been fulfilled. Most students find that they have to plan their courses both within DukeHub and somewhere outside of DukeHub, whether that be in a spreadsheet, iPhone note, or on paper. We wanted to re-work this process so hopefully it can be re-designed to make it more efficient and clear.

Personas

To begin, personas were created so as to be able to see how certain students approach searching for and planning classes. A persona is a document that describes the ways in which certain types of people will use the product and show the goals users will be trying to achieve. In this case, personas were created for students in each year as the ways in which people plan and schedule classes changes based on where they are in their academic career.

Key Goals

- Explore classes without making any definite commitments
- Find cool classes
- Need to know that planning classes exists, but might not use this tool right now

Behaviors

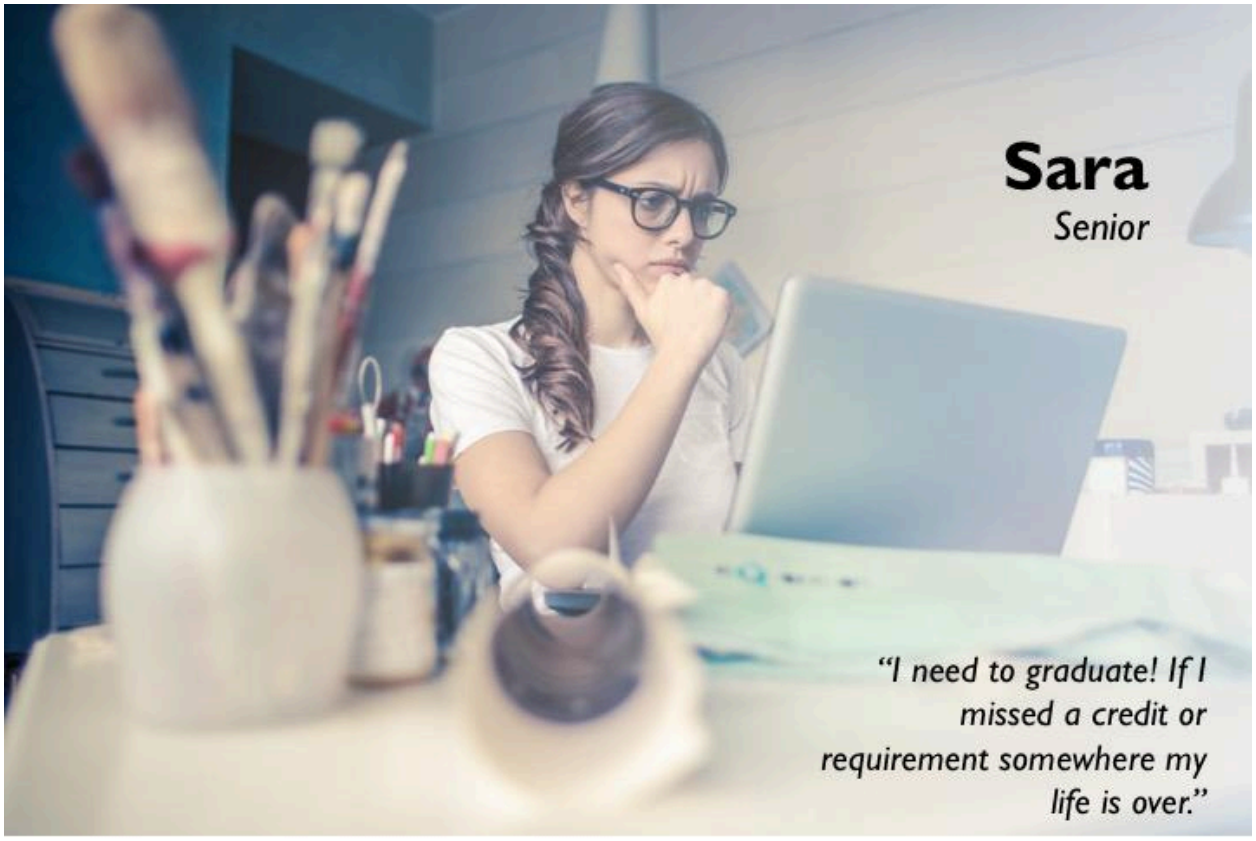
- Reliance on freshman year advisor
- Taking intro classes
- Taking classes because they look interesting and not because they fulfill a requirement

We must

- Allow him to explore classes
- Allow him to include AP/IB credits

We must not

- Overwhelm him with options
- Allow him to plan classes that have prerequisites he hasn't planned for/for semesters they aren't offered
- Lock him in to a potential major



Key Goals

- Graduate with enough credits
- Fulfill school graduation requirements (Trinity/Pratt)
- Fulfill major/minor/certificate graduation requirements

Behaviors

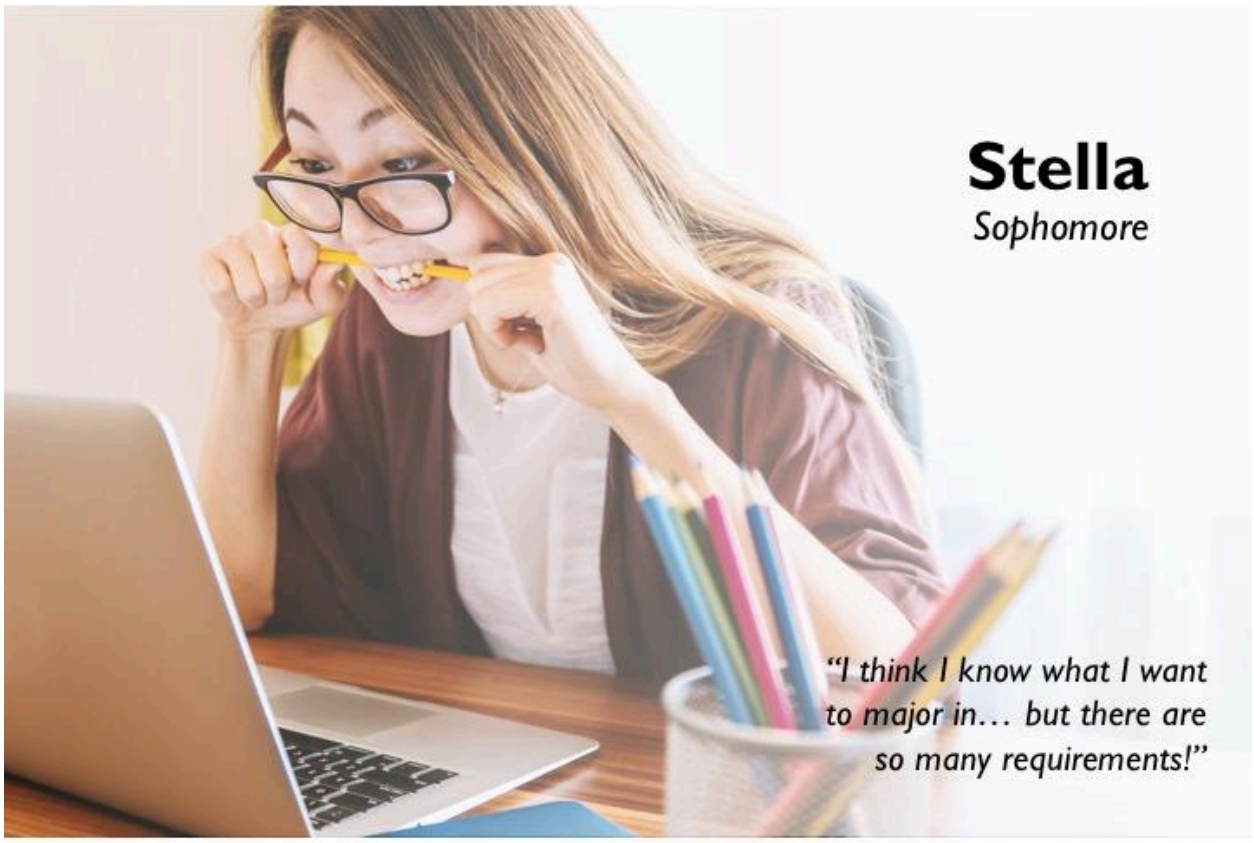
- Frantically searching for classes to fulfill that one random requirement she never got done.
- Potential to part-time

We must

- Help her make sure she's hit all her requirements in her plan
- Make sure she has enough credits

We must not

- Make her feel hopeless if something was missed



Key Goals

- Plan out potential majors
- See if her planned major/minor/certificate programs can all be completed by graduation

Behaviors

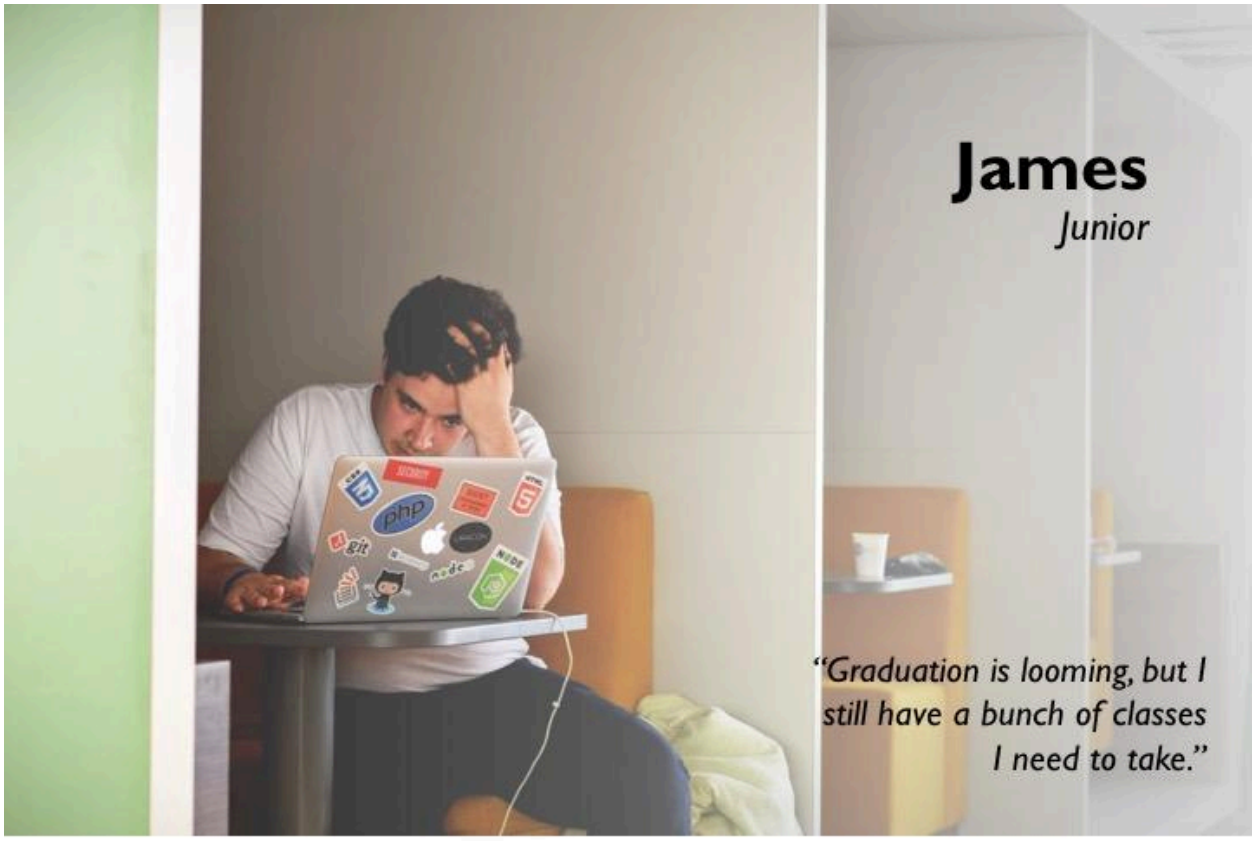
- Picking classes to fit with chosen/planned major declaration
- Declaring major
- Meeting with major advisor(s)
- Frustration is likely
- Likely to feel overwhelmed/confused

We must

- Allow her to plan for going abroad
- Remind her that things like T-Reqs exist
- Allow her to plan different courses of study

We must not

- Overwhelm her with all the requirements
- Make her feel locked in to one major



Key Goals

- Plan semi-definitive classes
- Feel secure in his course choices for his second half of college

Behaviors

- Going abroad
- Seriously thinking about what classes he wants to take before he graduates
- Likely frustration

We must

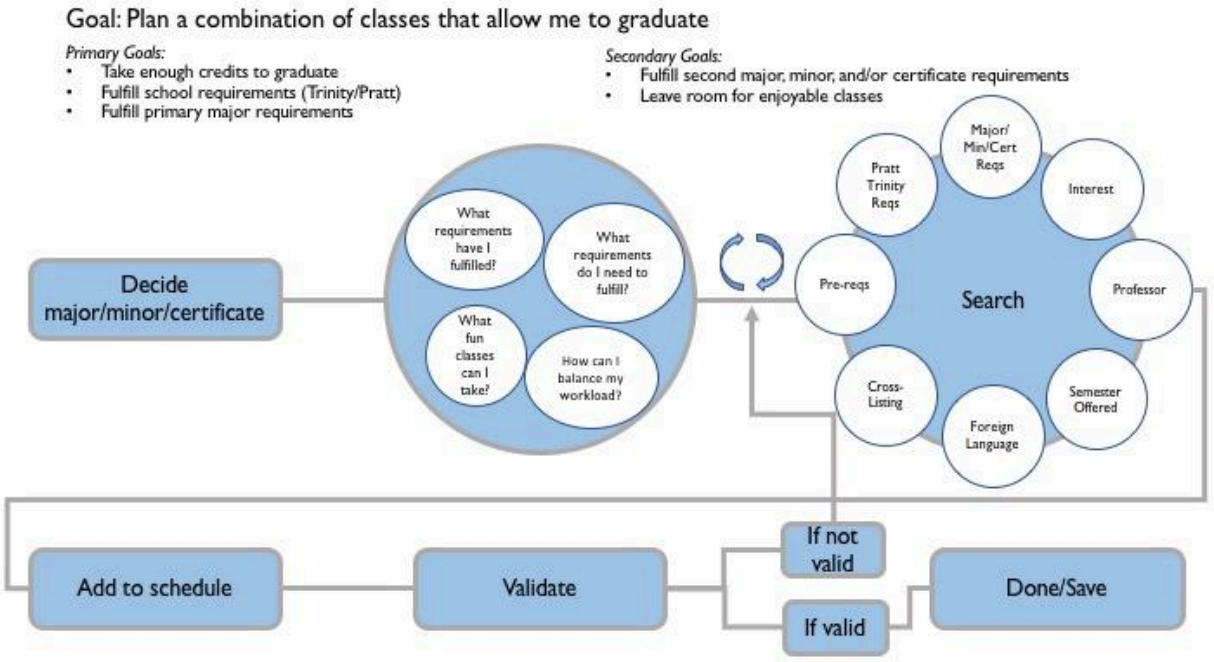
- Help him make sure he's hit all his requirements in his plan
- Allow for changes in major/minor/certificate
- Allow him to search for classes with his specific requirements

We must not

- Make him feel trapped within his major choice

Task Model

A task model was created to visually map how a user approaches the task of planning courses. Task models show what users do, the behavior they adopt, and specific requirements at each stage. A task model also acts as a type of vision document that identifies and communicates how a system needs to behave to match user expectations. It gives focus and direction to the experience that needs to be created for a successful product.

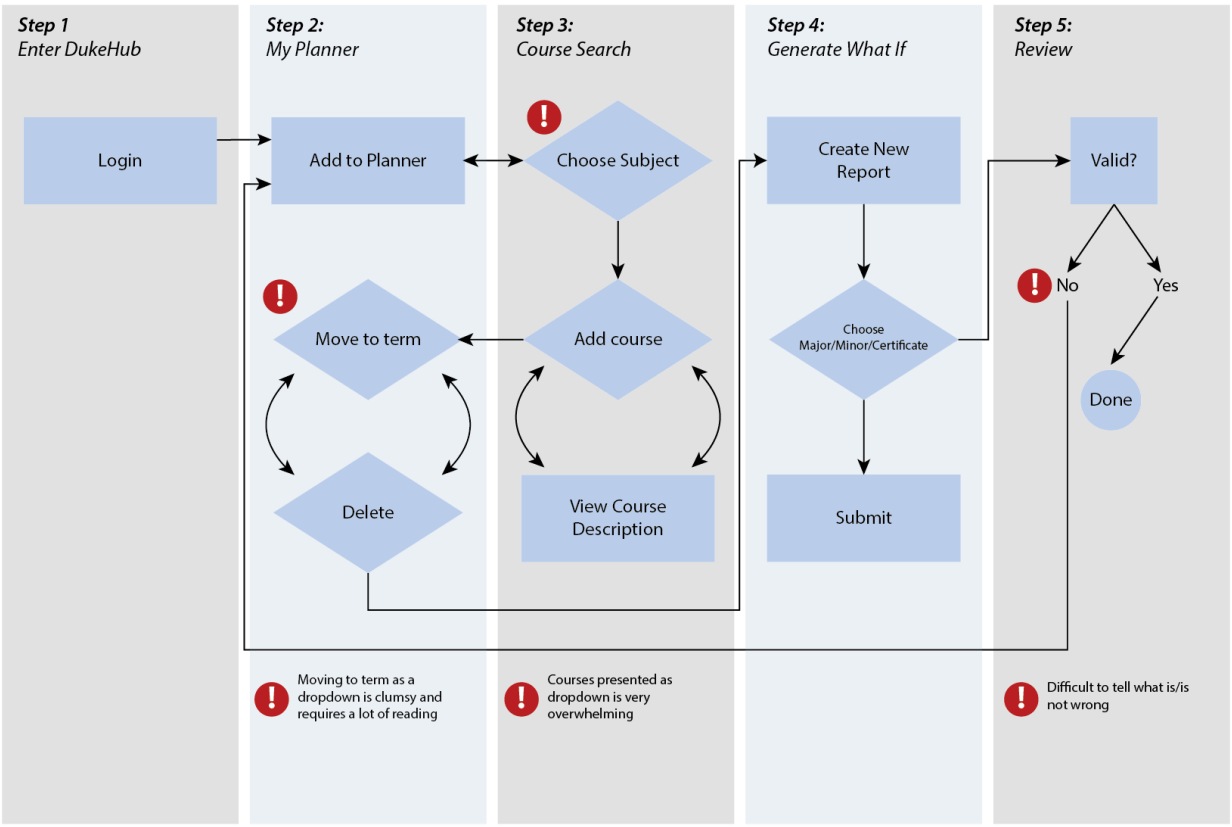


User Journey

A user journey details the exact steps a user goes through to complete a task or goal. It differs from a task model in that it shows the required interactions and paths through a system rather than being a representation of desired user behavior. In this case, a user journey was created of the current process students go through using DukeHub. Highlighted in red are the “pain points,” or areas where students tend to get frustrated. This allows the new design to avoid these issues and improve upon them.

User Journeys (Current)

Goal: Plan and Validate Classes

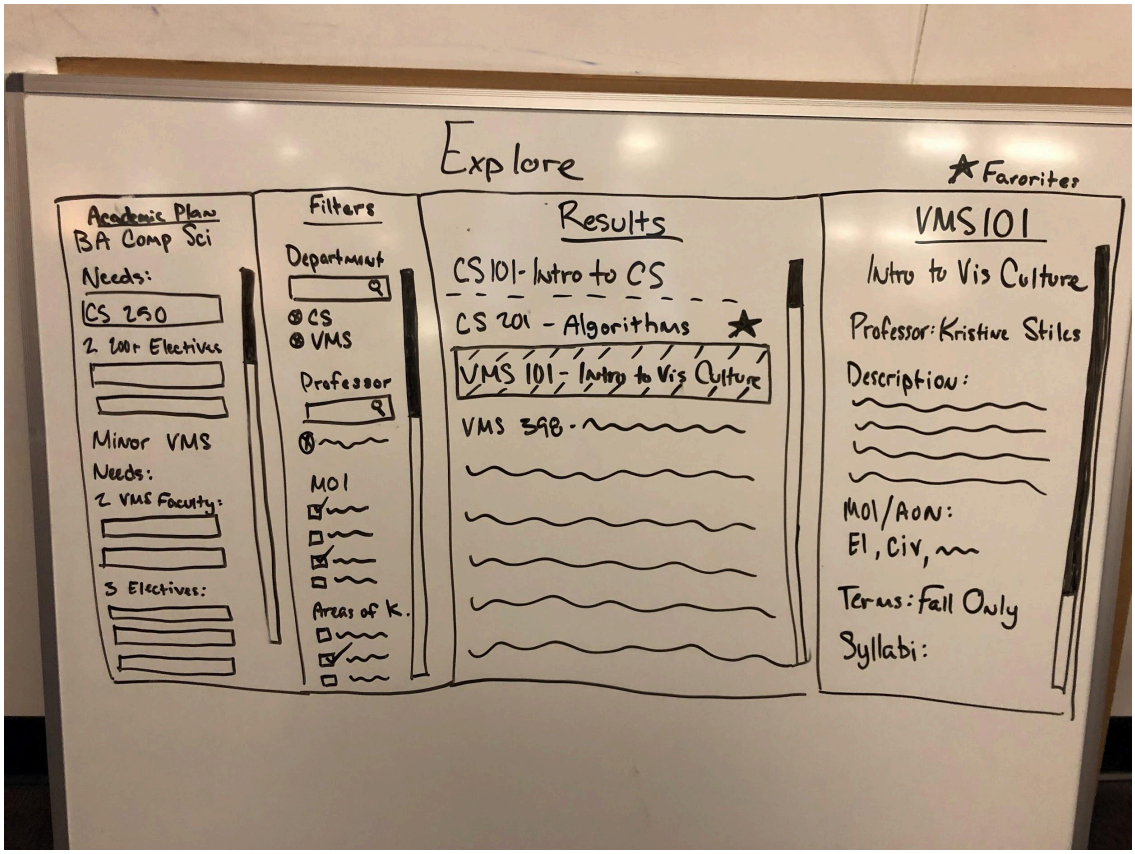


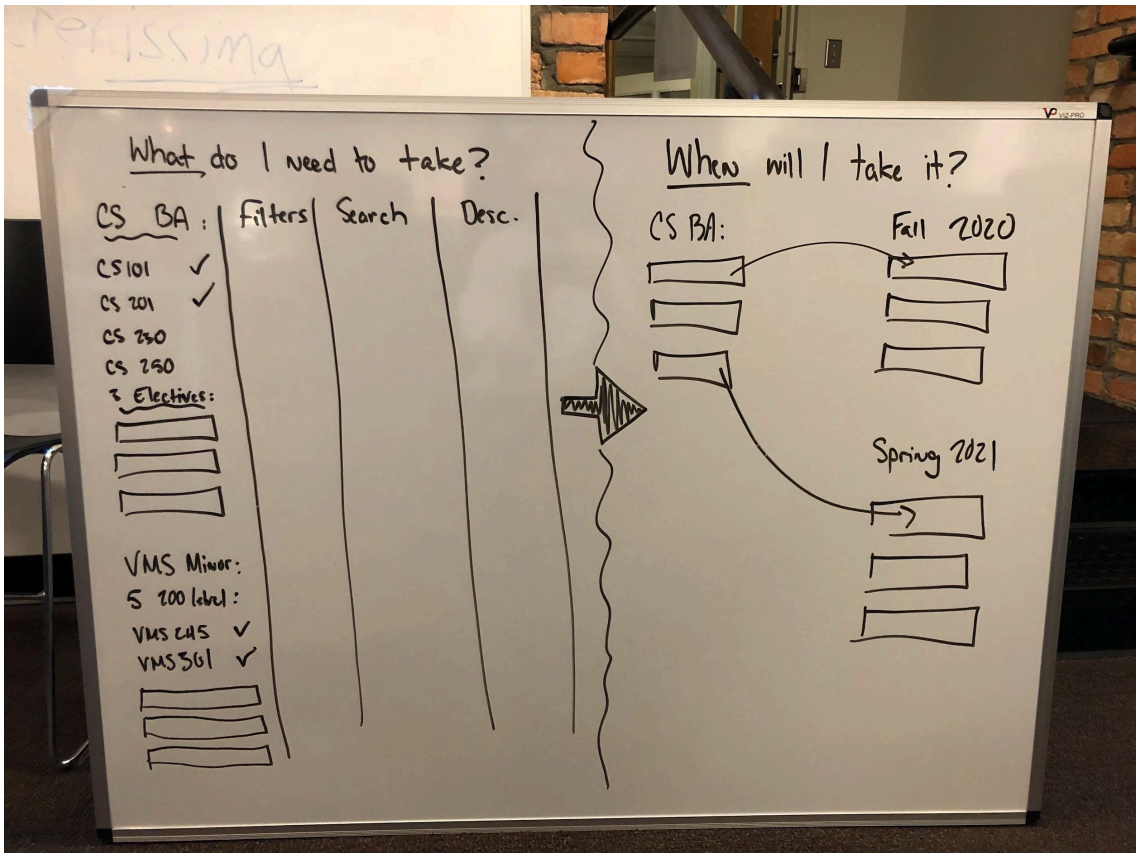
Wireframing

The next step in the process was to consult with the team and create some wireframes to be used for the interactive prototype. These were made with the input of the whole class. The general idea was to have the least number of different pages while also not overwhelming the user with information. After discussing, we realized there were two main stages to planning courses:

1. Choosing courses: What do I need/want to take?
2. Scheduling courses: When do I need/want to take it?

Because of this, it made the most sense to separate the process into two pages as demonstrated by the second picture.


Explore
Favorites
Academic Plan
BA Comp Sci
Needs:
CS 250
2 200+ Electives
Minor VMS
Needs:
2 VMS Faculty:
3 Electives:
Filters
Department
CS
VMS
Professor
MOI
Areas of K.
Results
CS 101 - Intro to CS
CS 201 - Algorithms
VMS 101 - Intro to Vis Culture
VMS 398 -
VMS 101
Intro to Vis Culture
Professor: Kristine Stiles
Description:
MOI/AON:
EI, Civ,
Terms: Fall Only
Syllabi:


What do I need to take?
When will I take it?
CS BA:
Filters
Search
Desc.
CS 101
CS 201
CS 230
CS 250
3 Electives:
VMS Minor:
5 200 level:
VMS 245
VMS 361
CS BA:
Fall 2020
Spring 2021

II: Acquiring Data

Meeting with experts

In order to improve the course planning systems available to students at Duke, we first had to gain a better understanding of those systems. To gain an administrative perspective, we met with the Duke University Registrar. During our meeting, the Registrar showed us how Duke's class and course data was organized in Tableau. We realized that acquiring the data we were interested in analyzing would not be as easy as we initially thought due to the structure of the database. Additionally, the Registrar was unable to share certain pieces of data with us for one of two reasons. First, some of the data was considered curricular and academic advisement data, which the registrar was unable to share with us. Secondly, we could not gain access to any data that could be traced back to a particular student. Distributing identifiable information is against FERPA, so we were unable to collect any information about the students enrolled in particular classes.

After a discussion with the Registrar, we decided that as a bare minimum we would need detailed information on the general course level and the course offering level. The general course level details include information such as the subject area, catalog number, enrollment restrictions, a description of the class, curricular codes, and cross listed courses. When analyzing specific offerings of the courses, we were interested in seeing the meetings patterns (days of the week and times), professors and class enrollment broken down by cross listing. Unfortunately, the Registrar could not provide us with all of the data we were interested in due to restrictions on sharing information related the Duke's Curricular Codes. When we discussed the best way for our team to access the data, we decided that for the amount of data we require and the goals of the project, CSV downloads would be more time efficient and secure than setting up an API. He did however send us some of the data that we requested in a CSV file while we researched other ways of accessing the data we needed. Additionally, he connected us with Richard Outten. Mr. Outten is the Senior Manager of the IT department at Duke, and he was able to direct us to a class data API. Through this API we were able to access all of the data we were interested in for our class.

To gain a broader student perspective, we met with the Duke Student Government VP of Academic Affairs, Saheel Chodavadia. The Academic Affairs committee strives to improve the academic environment on campus by looking at ways to improve resources, creating new and innovative programs, and receiving feedback from students. Their list of ongoing projects actually includes "developing a tool to help navigate courses in majors". We discussed the potential of using the Academic Affairs Syllabus archive to get a more detailed idea of what a class will be like, but unfortunately the syllabi are watermarked PDFs, so we would have been unable to run text analysis on them. Overall, this meetings confirmed the idea that students are unsatisfied with the what if report and course planning.

Pulling data from the API

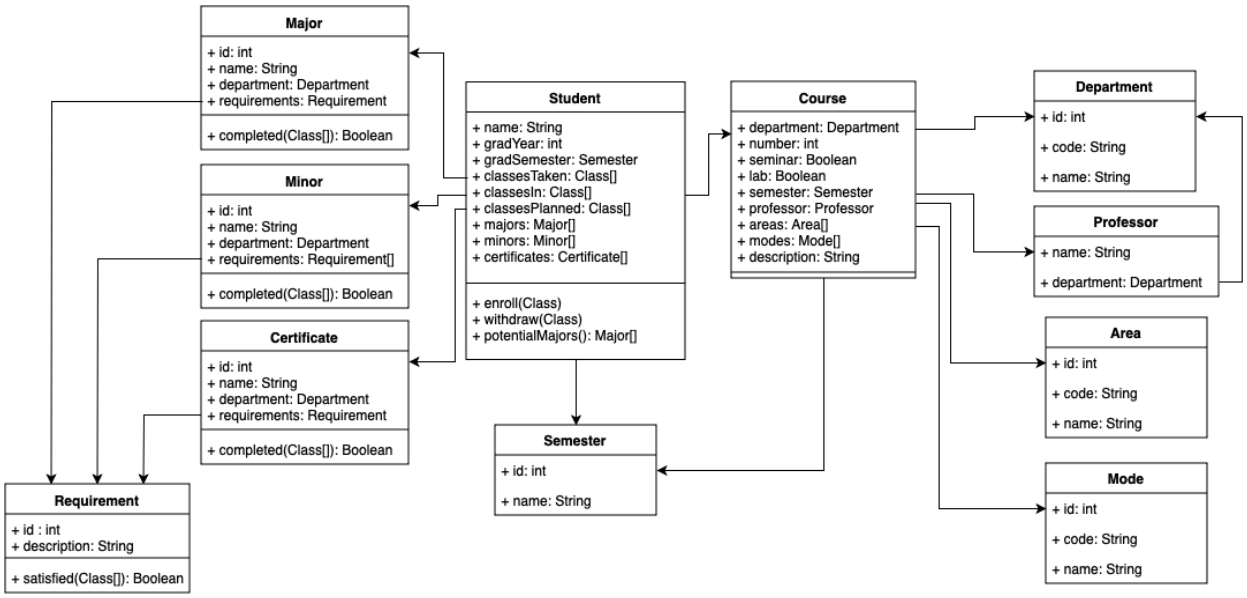
Once Mr. Outten directed us the the API, we had to decide whether we wanted to pull the data from the API into CSVs or if we wanted to poll directly. In order to allow half of the team to work on data visualizations and have a broader picture of the data, we decided to pull it into CSVs. The API has six queries related to course and class data. In order to pull the data we were interested in, we used four of these queries.

Our first query was used to obtain a list of every course ID and course offer number listed under a number of pre-selected subjects. We included all subjects that offer classes to undergraduate students. Once we had a list of all of the courses in these subjects, we used a second query to obtain details about those courses including the subject area, catalog number, enrollment restrictions, a description of the class, curricular codes, and cross listed courses. Next, we used the a list of all of the courses in these subjects and a list of term IDs to search for class offering information for each course for every term at Duke since 2012. The output of this query included the class meeting patterns and instructor, and other information used to make the fourth query which gave us the enrollment for each class section. The scripts used to call the API and record the data can be found here.

III: Creating the Database

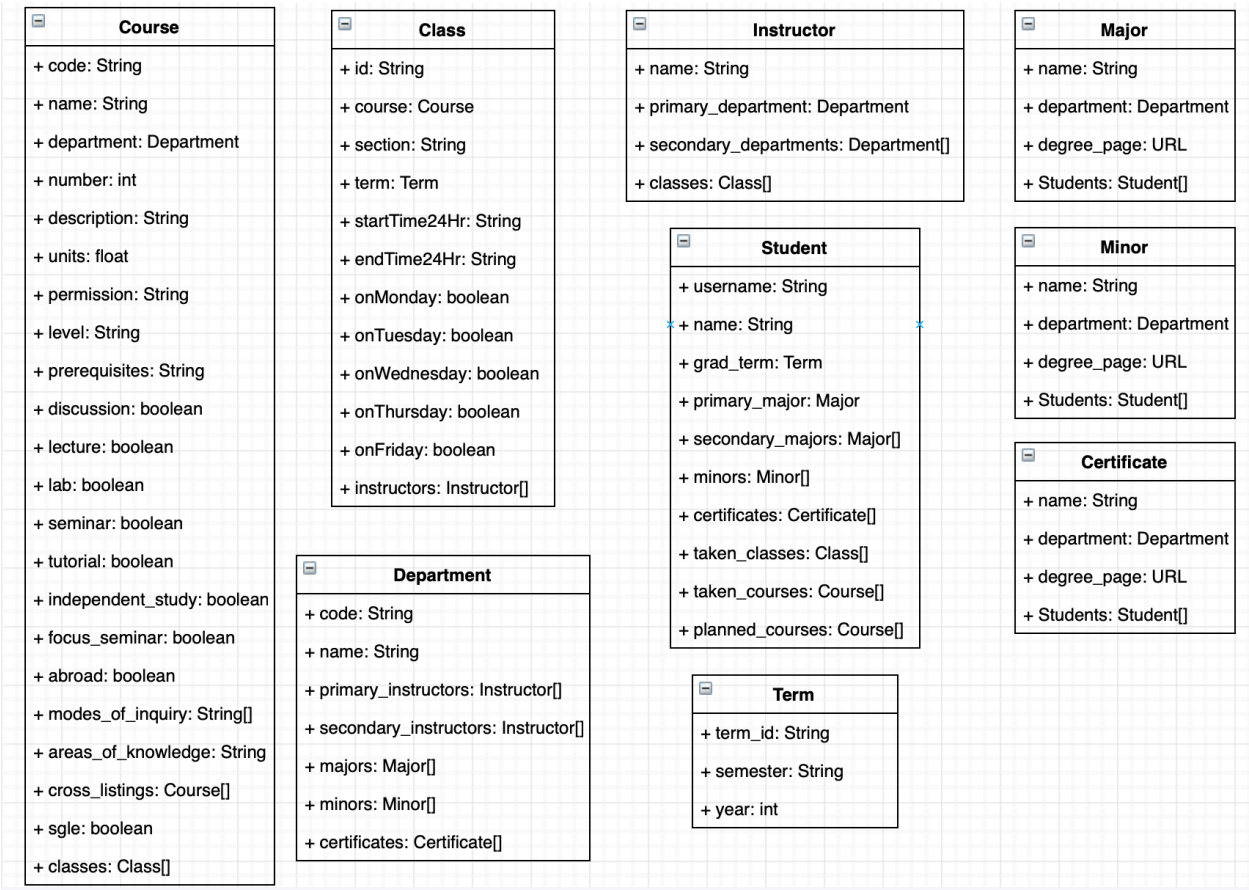
Organizing the Data

Before creating our database, we had to decide on the data points that we would need to maintain in order to provide the desired functionality. From the beginning it was clear that we would need Duke course data for the primary functionalities including course search, academic planning, and the “What If?” report. In order to implement the desired filters, this would have to include details such as course attributes, areas of knowledge, and cross-listings. For registration scheduling and class analytics, we would need to collect class data such as start times, end times, offering days, and instructors. After it was determined that we could not pull a student’s course history into our website through the registrar, we had to generate a table of users that would include planned degrees, courses taken, classes taken, and planned courses.
Once the data points were solidified, we had to design a database schema. This lays out the structure of the database. What data structures are needed? Which properties should each structure have? What interaction will it provide to users and to other structures? With these questions answered, we were able to visualize our database with a UML diagram.



Initial UML Diagram

The initial schema, however, has changed over time. Courses have been split into Classes, which are instances of Courses and contain offering-specific information. Requirements were removed entirely from the database after deciding that they were too difficult to generalize and too messy to generate. Thus, their logic would be handled in the codebase. At this point in time, our database adheres to the following UML.

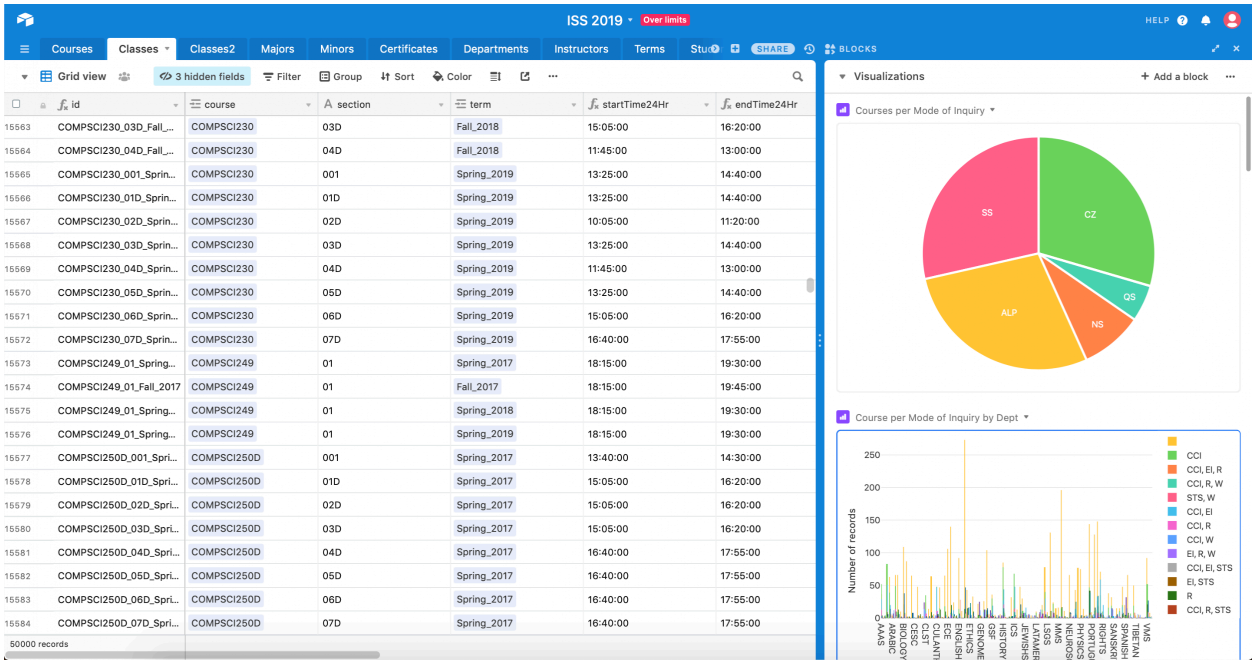


Updated UML Diagram

Choosing a Database

When exploring platforms with which to create our database, a few considerations guided our decision. First, there were clear relationships between the structures containing our data. Thus, a relational database was the most intuitive option. Second, we knew that the exact schema would continue to fluctuate throughout the prototyping phase. Therefore, we needed the capability to quickly and simply restructure the database. Lastly, we needed the ability to feed a dynamically updating website.

After considering traditional choices such as MongoDB and MySQL, we eventually settled on Airtable. Airtable has a simple and intuitive UI, making it easy to quickly create complicated yet clean relational databases. It has a variety of complex field types, such as functions, dropdowns, and links, which prevent inconsistencies and simplify complexities. Intelligent copy-and-paste functionality makes it easy to map from primitive CSV files to Airtable's field types. Looking ahead, Airtable automatically generates an API to read and update your database and includes a suite of basic data visualization tools with the Chart block. The current database adheres to the most recent schema.



Airtable Data Visualizations

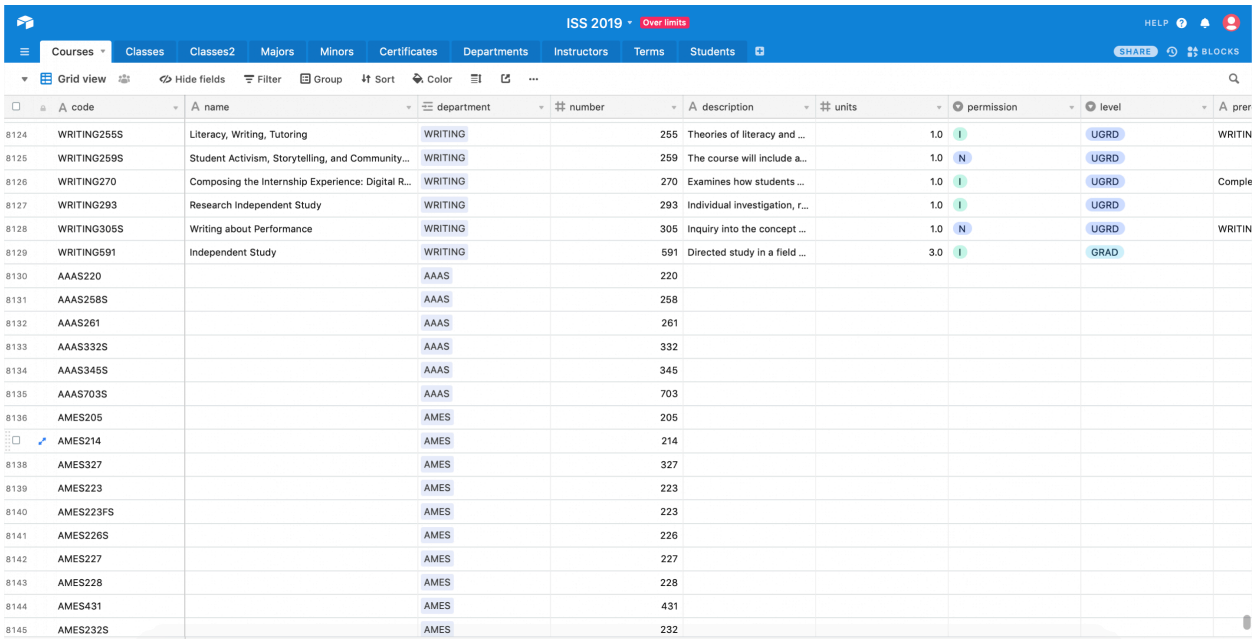
Despite its benefits, Airtable still has issues handling the large number of entries that our database required. The bulk addition of entries proved difficult, though not impossible. Airtable does provide tools to import CSV files, either to create a new table or to add to an existing one. However, these tools are capped at 2MB. This became problematic when attempting to add our 4.4MB Course data file, and even more so when adding nearly 30MB of Class data. Manual copy-and-paste operations were limites as well, only allowing around 5,000 entries at a time. Thus, this process became relatively slow and tedious.

The larger problem, however, is Airtable's limitation on database entries. Even with the most expensive publicly available plan, our database is only allowed 50,000 total entries. Considering that our database contains over 65,000 entries before user data, this limit was quickly exceeded. In addition, there is a limit of 50,000 entries per table within the database. Because of this, we had to split our 51,224 Class entries into two separate tables, Classes and Classes2. If continuing with Airtable in the future, it would be necessary to contact them about an Enterprise-level account that includes an increased limit on entries.

Cleaning the Data

Beyond the above issues, entering the data into Airtable was relatively painless. Additionally, because we had the Airtable structure and copy-and-paste mechanics in mind when storing the data pulled from the registrar, adjustments within Airtable were minimal. Clearly defined field types forces values to adhere to the schema. Even the more complex dropdown and link fields could be translated from CSV.

There were some inconsistencies within the registrar's database that could not be caught in the data acquisition phase. These fell primarily in the course codes. When creating our database, we added the registrar's Course data first. Then, upon adding the Class data, we found about 400 Courses that had not been listed before. Some of these, such as AAAS 332S, had attributes, such as a seminar or lab, added or taken away over time. While this information was updated in the registrar course dataset, the old course code remained in the class dataset. Thus, the Courses table did contain the updated version. However, some classes, such as VMS 460S, were absent from the Course database entirely. These inconsistent courses occupy the bottom of the Courses table in the Airtable base, beginning with AAAS 220 at row 8130.

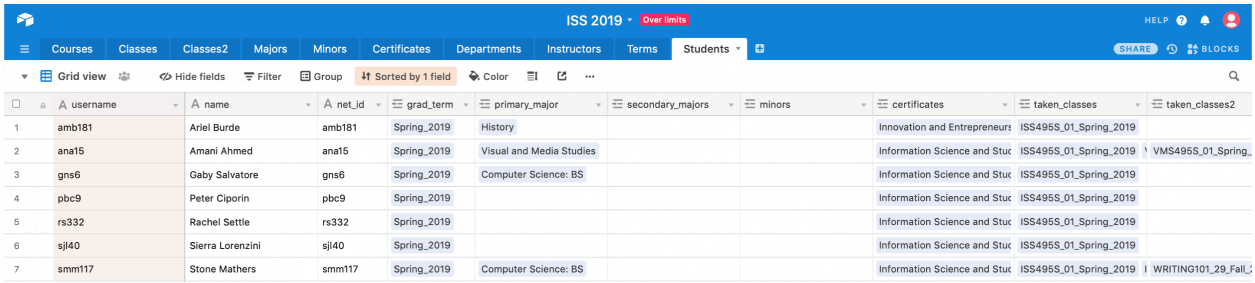


	code	name	department	number	description	units	permission	level	prer
8124	WRITING255S	Literacy, Writing, Tutoring	WRITING	255	Theories of literacy and ...	1.0	I	UGRD	WRITIN
8125	WRITING259S	Student Activism, Storytelling, and Community...	WRITING	259	The course will include a...	1.0	N	UGRD	
8126	WRITING270	Composing the Internship Experience: Digital R...	WRITING	270	Examines how students ...	1.0	I	UGRD	Comple
8127	WRITING293	Research Independent Study	WRITING	293	Individual investigation, r...	1.0	I	UGRD	
8128	WRITING305S	Writing about Performance	WRITING	305	Inquiry into the concept ...	1.0	N	UGRD	WRITIN
8129	WRITING591	Independent Study	WRITING	591	Directed study in a field ...	3.0	I	GRAD	
8130	AAAS220		AAAS	220					
8131	AAAS258S		AAAS	258					
8132	AAAS261		AAAS	261					
8133	AAAS332S		AAAS	332					
8134	AAAS345S		AAAS	345					
8135	AAAS703S		AAAS	703					
8136	AMES205		AMES	205					
	AMES214		AMES	214					
8138	AMES327		AMES	327					
8139	AMES223		AMES	223					
8140	AMES223FS		AMES	223					
8141	AMES226S		AMES	226					
8142	AMES227		AMES	227					
8143	AMES228		AMES	228					
8144	AMES431		AMES	431					
8145	AMES232S		AMES	232					

Inconsistent Course Entries

User Data

Whenever a new user logs in to our website, they will be added to the Students table of the database. When the user logs in through Auth0, we receive a data structure that contains the user's "nickname," which is the local-part of username of the email used to sign in. This nickname is stored as that user's unique identifier. To prevent overlap, we will only accept duke email addresses. From that point on, a returning user's data can be easily retrieved without our database ever storing a password or other private identifier. After their initial log in, the user can begin to select their majors, minors, certificates, previously take classes, and planned courses. All of these values will be stored in various fields the user's entry within the Students table.

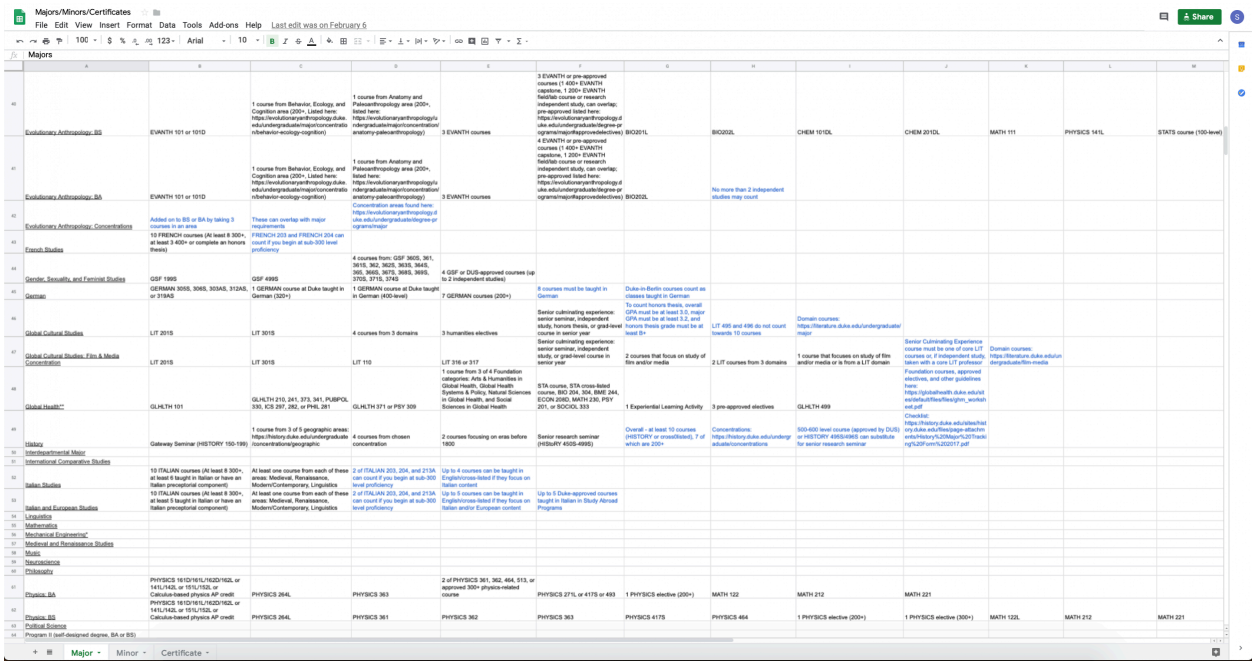


	username	name	net_id	grad_term	primary_major	secondary_majors	minors	certificates	taken_classes	taken_classes2
1	amb181	Ariel Burde	amb181	Spring_2019	History			Innovation and Entrepreneurs	ISS495S_01_Spring_2019	
2	ana15	Amani Ahmed	ana15	Spring_2019	Visual and Media Studies			Information Science and Stuc	ISS495S_01_Spring_2019	VMS495S_01_Spring_
3	gns6	Gaby Salvatore	gns6	Spring_2019	Computer Science: BS			Information Science and Stuc	ISS495S_01_Spring_2019	
4	pbc9	Peter Ciporin	pbc9	Spring_2019				Information Science and Stuc	ISS495S_01_Spring_2019	
5	rs332	Rachel Settle	rs332	Spring_2019				Information Science and Stuc	ISS495S_01_Spring_2019	
6	sjl40	Sierra Lorenzini	sjl40	Spring_2019				Information Science and Stuc	ISS495S_01_Spring_2019	
7	smm117	Stone Mathers	smm117	Spring_2019	Computer Science: BS			Information Science and Stuc	ISS495S_01_Spring_2019	WRITING101_29_Fall_

Student Data

“What If?” Logic

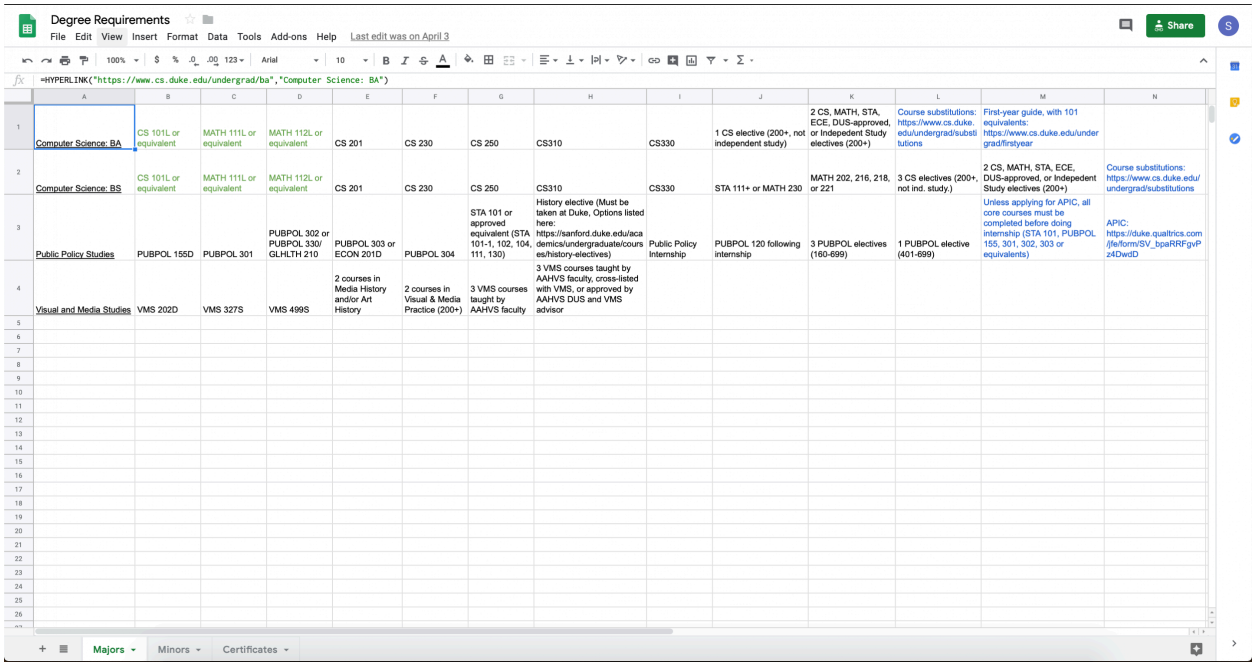
Although it would still no longer be present in the database, it was still necessary to compile the details behind completing each major, minor, or certificate degree for the new “What If?” report. There is currently no single database for degree requirements, so this required manually combing through each department website and translating their requirements into a common Boolean language. Every possible path for every degree had to be organized into its most succinct logical form. In the spreadsheet containing this information, each row represents a degree and each column contains a requirement.



Initial Degree Requirement Spreadsheet

It is quickly evident that this spreadsheet was never completed. Upon deciphering each department’s version of a required curriculum, it soon became clear that they would be far harder to generalize than anticipated. Each department has its own structure, discrepancies, and exceptions to their own rules. Some degrees had hundreds of permutations which, while great for academic flexibility, becomes a tedious and time-consuming problem over dozens of

departments. Some required data the does not currently exist in a digital form, such as an instructor's primary departmental affiliation. Others would require manual data entry by the student that our system would have no means of verifying, such as the Public Policy major's internship requirement. As a result, the logic behind each degree's completion would need to be individually programmed, with very little overlapping code. Due to these unexpected complications, we decided to reduce our scope to four departments: Computer Science, Public Policy, Visual Media Studies, and Information Science & Studies. With this change, we could now focus more on accounting for the nuances of each department, which have been compiled into pseudocode.

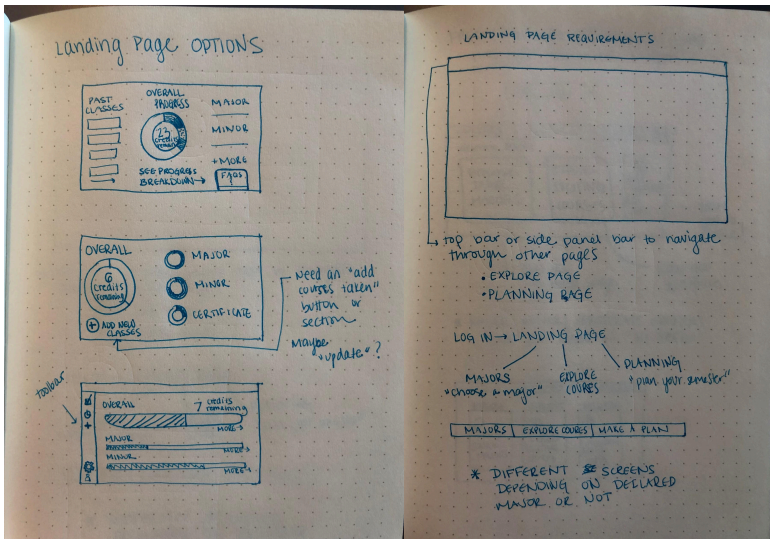


Reduced Scope Degree Requirement Spreadsheet

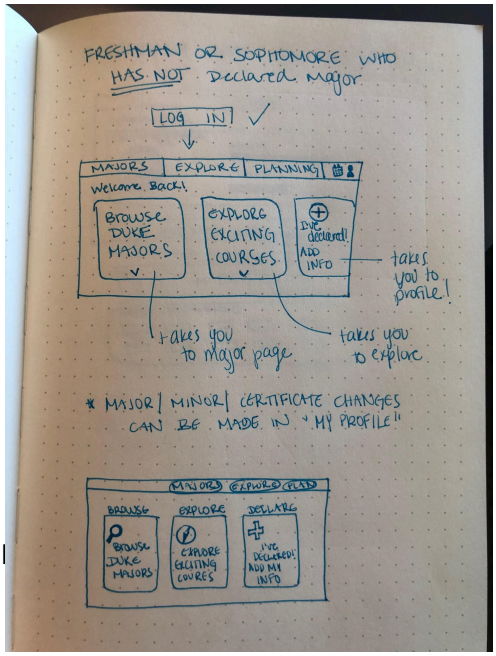
IV: Landing Page and Major Pathways

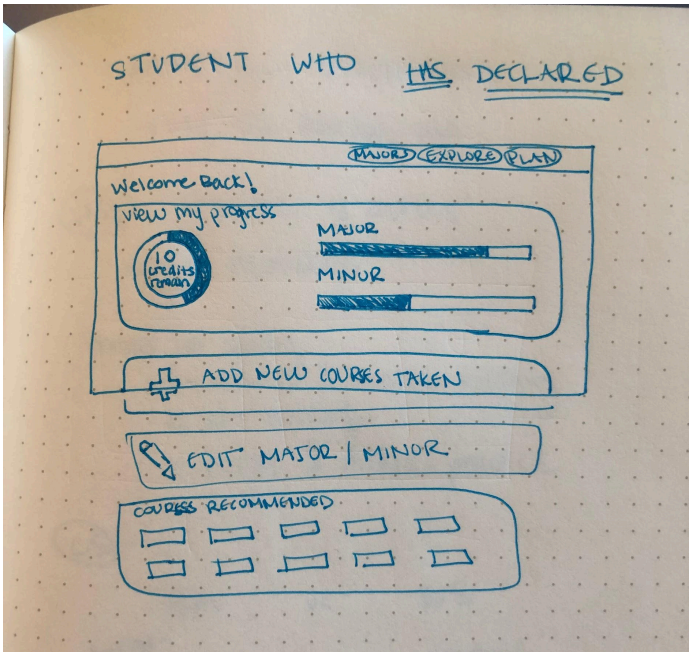
Landing Page (Dashboard)

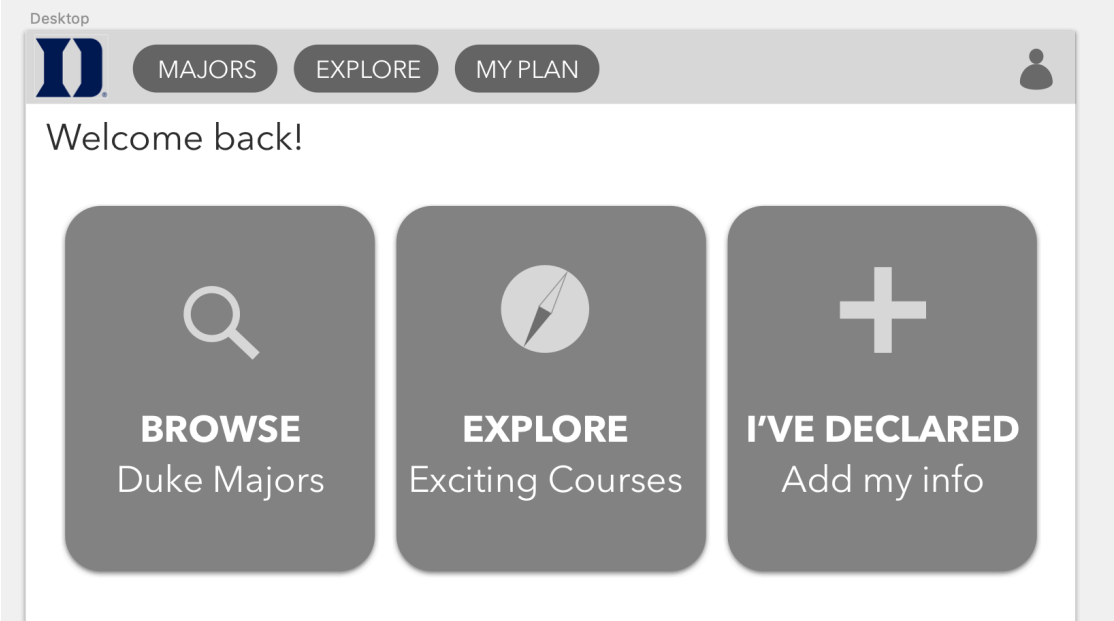
The Landing Page and Major Pathways page was designed in order to providing a start space for the user. The Landing Page, otherwise referred to as the Dashboard, is the first screen shown after the student logs in. It is important to note that the Landing Page front-end differs based on a “declared/not declared” major status. For our personas freshman Frank and sophomore Stella, they fall into the “not declared” major status. They are logging into explore interesting classes, take introductory courses, or potentially fulfill a Trinity requirement.



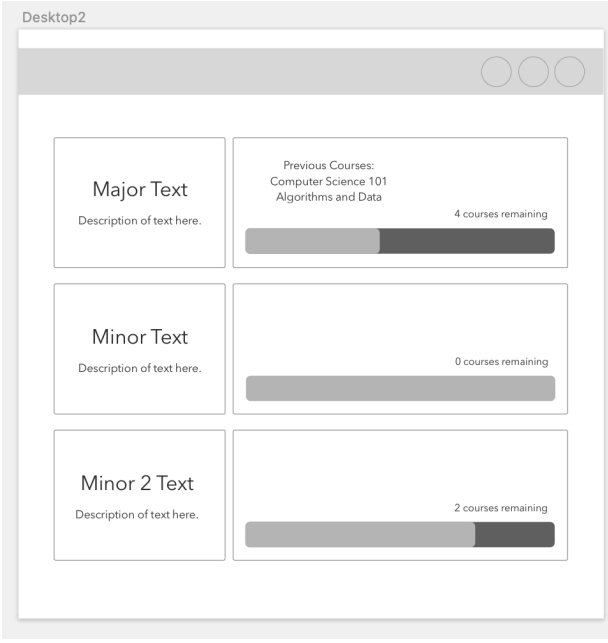
Once logged in, “not declared” students will see a unique page. It contains three main sections. “Browse Duke Majors”, “Explore Exciting Courses”, and “I’ve Declared! Add Info.” Each of these sections take the user to a different page. The “Browse Duke Majors” links the user to the Major Landing Page (shown later). The “Explore Exciting Courses” section brings you to the “explore” page designed by the entire class and prototyped by Sierra. Reference this in Section V: Class Search. Major, minor, and certificate changes can be made in the “My Profile” section. Sketches of “not declared”/ “declared” pages can be seen below.



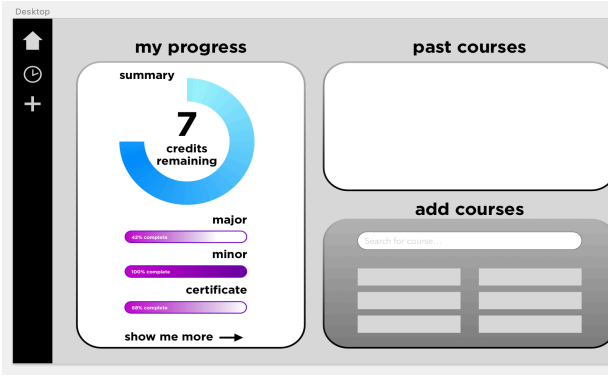




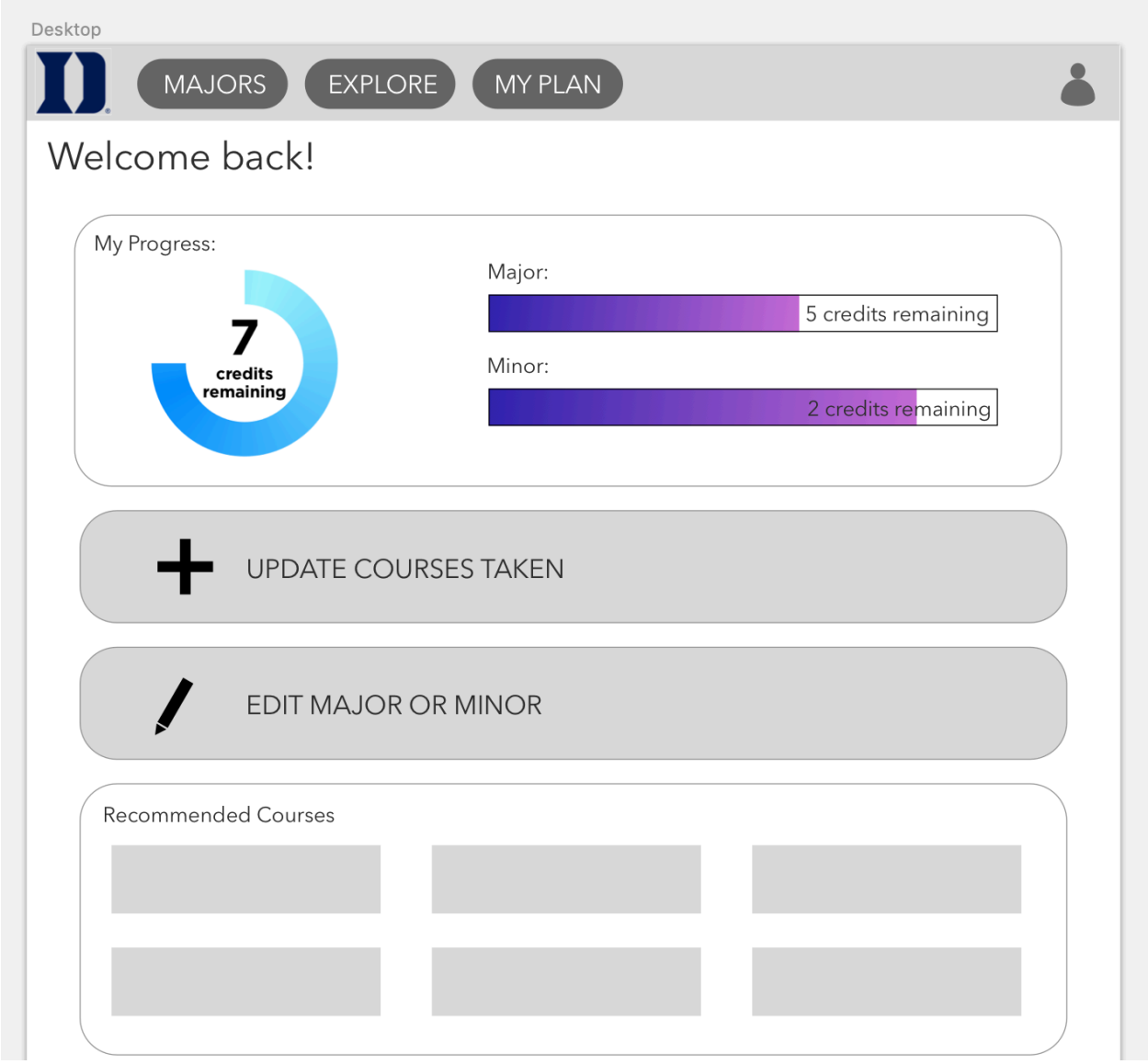
"Non declared" Landing Page Sketch



"Declared" Landing Page Sketch First Draft



"Declared" Landing Page Sketch Second Draft



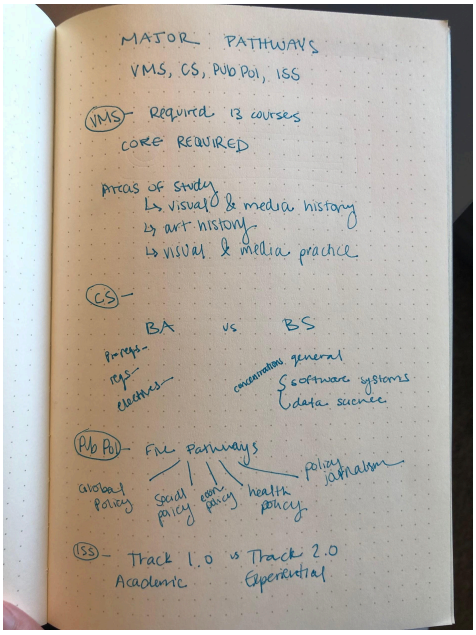
Sketch Final Draft

The final version of the “declared” landing page combines a few different aspects of the previous versions. The progress towards the user’s degree remains the most important aspect of the page. If a user is to click on the My Progress section, they will be taken to a breakdown of requirements completed versus requirements remaining. The “Update Courses Taken” is an important feature because without the user manually updated their list of courses, it is impossible for our prototype to know information such as the credits remaining. Using our personas as a guide, we do not want them to feel like they are ever locked into a major or minor. By providing an “Edit Major or Minor” button on the Landing Page, it ensures the user is aware they are not stuck in this system once they begin to input class data. Finally, a list of recommended courses will pop up based on the users previous courses and popular Duke courses in general.

Recommendations for moving forward include curating recommended courses using an algorithm that finds the highest rated course based on evaluation, previous courses taken, and most well-liked professors at Duke. Recommended courses would give new and intriguing courses a chance to shine and make sure that students do not skim over them as could happen in the current system.

Another recommendation would be to include some mouse hover over features for the “My Progress” section. This would allow the user to gain quick insights. For example, when hovering over “Minor: 2 credits remaining” it could show the two courses the user needs to take to complete their minor.

Major Pathways Page



When looking at the Major Pathways Page, it is important to notice the ability add majors in the future. For this page, the focus was on the four example majors: Visual & Media Studies, Computer Science, Public Policy, and the Information Science and Studies certificate. When breaking down each major, it is clear that some majors have true “paths” while others are divided by degree type. To look at our four example majors:

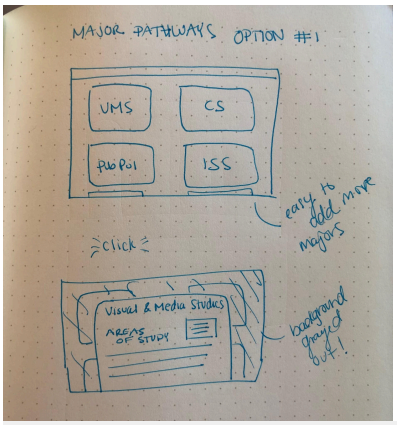
Visual & Media Studies has 13 required courses but can be divided into Areas of Study such as Visual & Media History, Art History, and Visual & Media practice. Knowing these three options can enable a user to make an educated choice about their major.

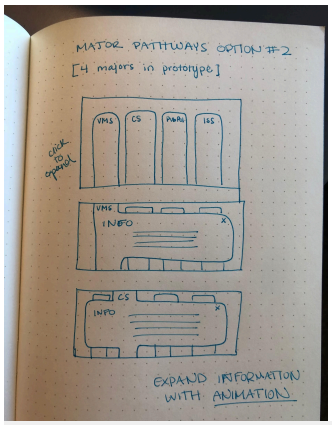
Computer Science is divided by the Bachelor of Arts and Bachelor of Science degrees. The courses required depend on the degree a student is pursuing. Helpful tips can be found within the Major Pathways Page such as, if a student is planning on double majoring, the BA is recommended, while if this is the student’s only major, the BS is recommended.

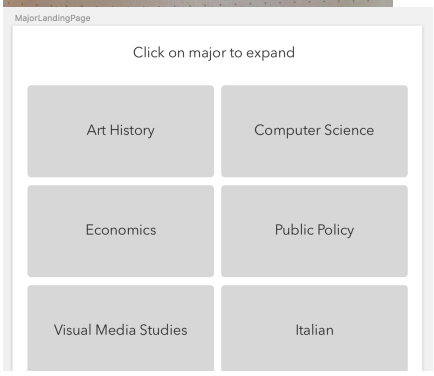
Public Policy major is broken into five distinct pathways, of which the user must choose. These pathways include: Global Policy, Social Policy, Economic Policy, Health Policy, and Policy Journalism. When declaring the Public Policy major, one must select a path as well. This is important for a prospective student to be away of how the pathways differ. A future recommendation would include an animation showing each of the courses filing into their respective pathways.

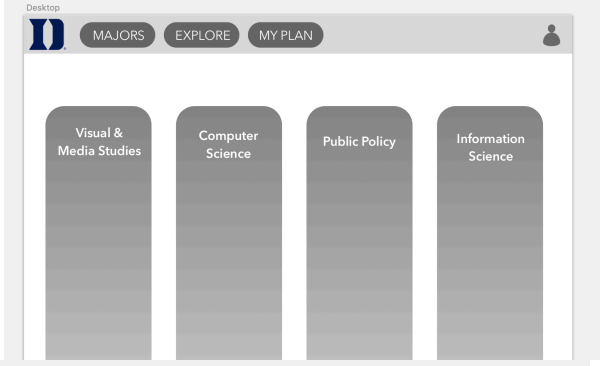
The Information Science and Studies certificate is broken into Track 1.0 and Track 2.0. The two differ in that one is academic and the other is experiential. For this prototype, only Track 1.0 Academic Option would be supported.

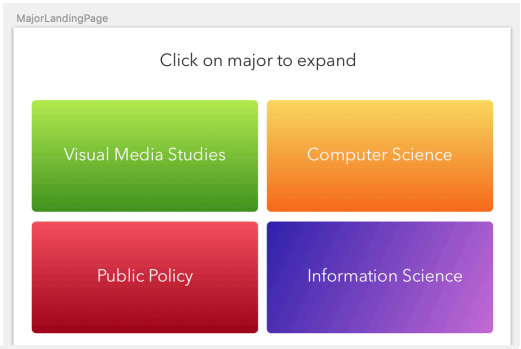
The front-end design for this page would be supported by JavaScript animations.

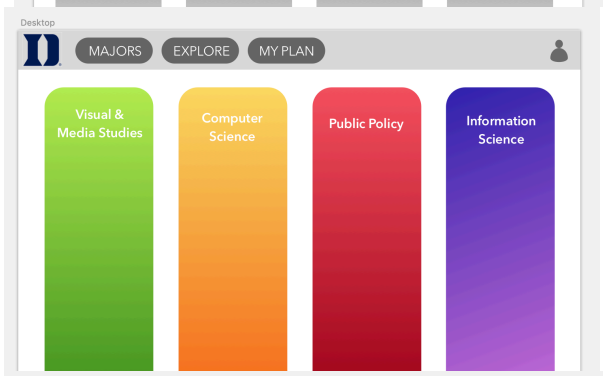

MAJOR PATHWAYS OPTION #1
VMS
CS
PubPol
ISS
easy to add more majors
click
Visual & Media Studies
AREAS OF STUDY
background grayed out!

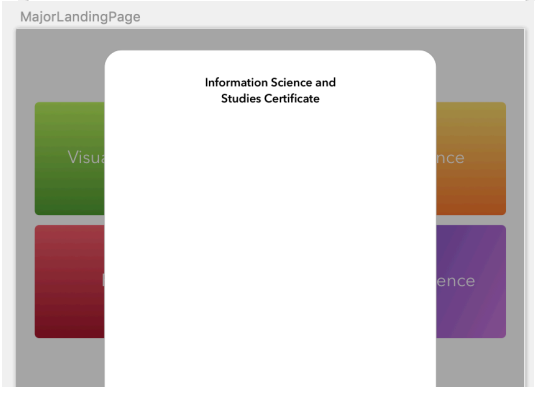

MAJOR PATHWAYS OPTION #2
[4 majors in prototype]
click to expand
VMS
CS
PubPol
ISS
INFO
EXPAND INFORMATION WITH ANIMATION

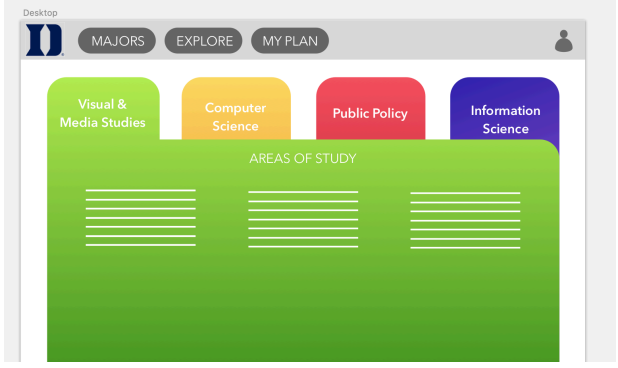

MajorLandingPage
Click on major to expand
Art History
Computer Science
Economics
Public Policy
Visual Media Studies
Italian


Desktop
MAJORS
EXPLORE
MY PLAN
Visual & Media Studies
Computer Science
Public Policy
Information Science


MajorLandingPage
Click on major to expand
Visual Media Studies
Computer Science
Public Policy
Information Science


Desktop
MAJORS
EXPLORE
MY PLAN
Visual & Media Studies
Computer Science
Public Policy
Information Science


MajorLandingPage
Information Science and Studies Certificate


Desktop
MAJORS
EXPLORE
MY PLAN
Visual & Media Studies
Computer Science
Public Policy
Information Science
AREAS OF STUDY

V: Class Search Design

Software Research

Once I had completed the steps found in Section I, the next step was to make higher fidelity mockups in a digital format. The first question I had to answer was which application to use to make them. This ended up being a bit of trial and error. I started in Adobe XD and was able to link it to AirTable. However, the original plan was to make an interactive prototype and XD didn't support the types of interactions needed (i.e drag and drop, multiple scrollable areas). I did a lot of research on various software for interactive prototyping. Eventually, I landed on using a combination of Sketch and Flinto (an application used on top of Sketch for more advanced animations and interactions). By using a Sketch plugin, I was also able to import data from Airtable. This took a bit of effort as at the beginning the plugin was broken so I had to contact the creator multiple times before it was fixed. Eventually, however, I was able to import the data my classmates worked so hard to acquire and organize in Airtable.

After conferring with my classmates, we realized that prototyping entirely in Sketch/Flinto wouldn't make sense since those applications are not built to work with large data sets. As a result, we decided to use a Sketch plugin called Anima to export my Sketch files to HTML/CSS so that Peter and Sloan could work with the code and incorporate more of the data.

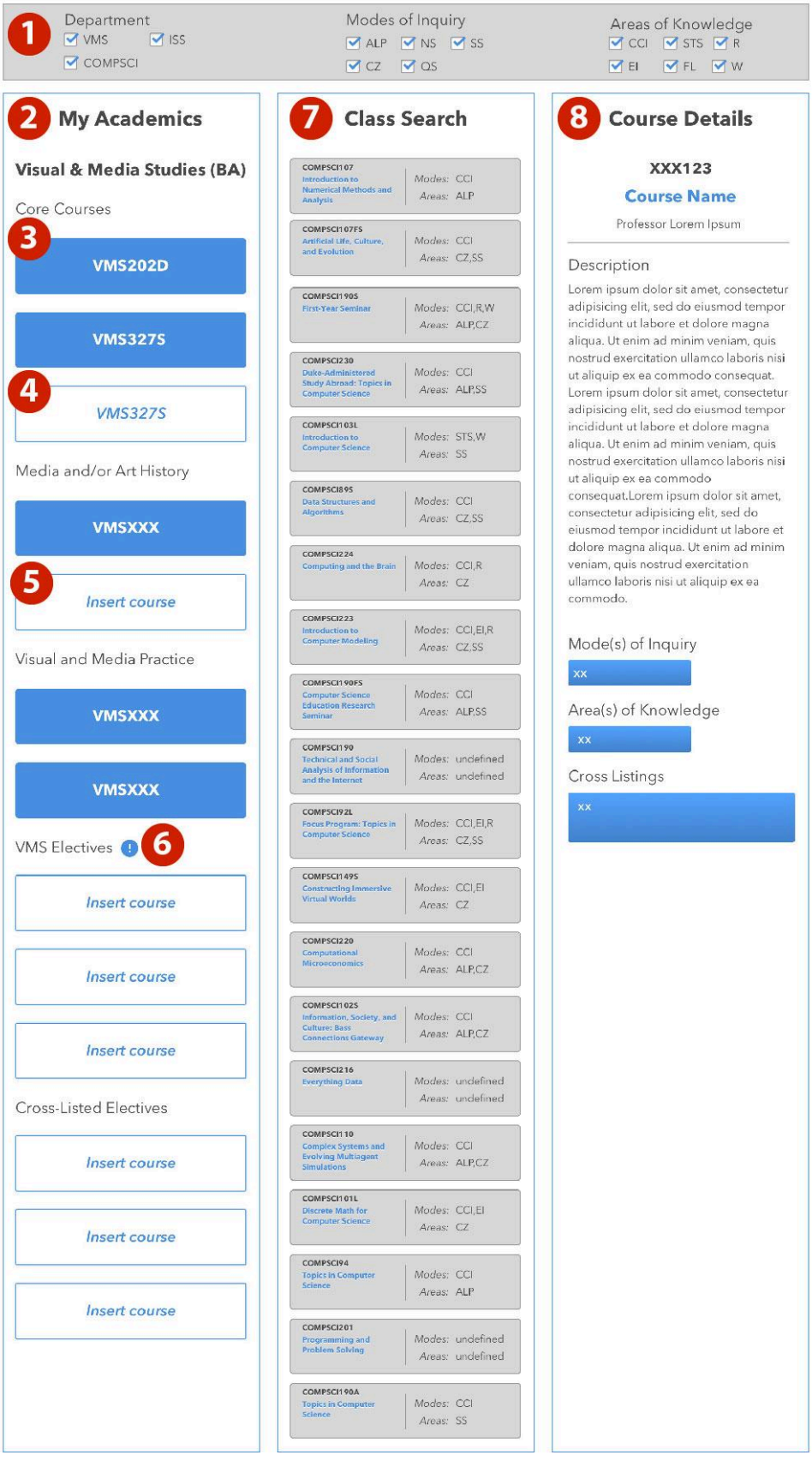
Digital Wireframes

Most of my work was done in Sketch to make digital wireframes. Although time didn't allow me to create interactive prototypes, I have used images here to reference the ideal interactivity for the future.

Class Search/ "What" Page

As mentioned in Section I, the process of planning courses is split into two parts, the first of which asks the question: What do I need to take? The wireframe here is of that step.

What do I need to take



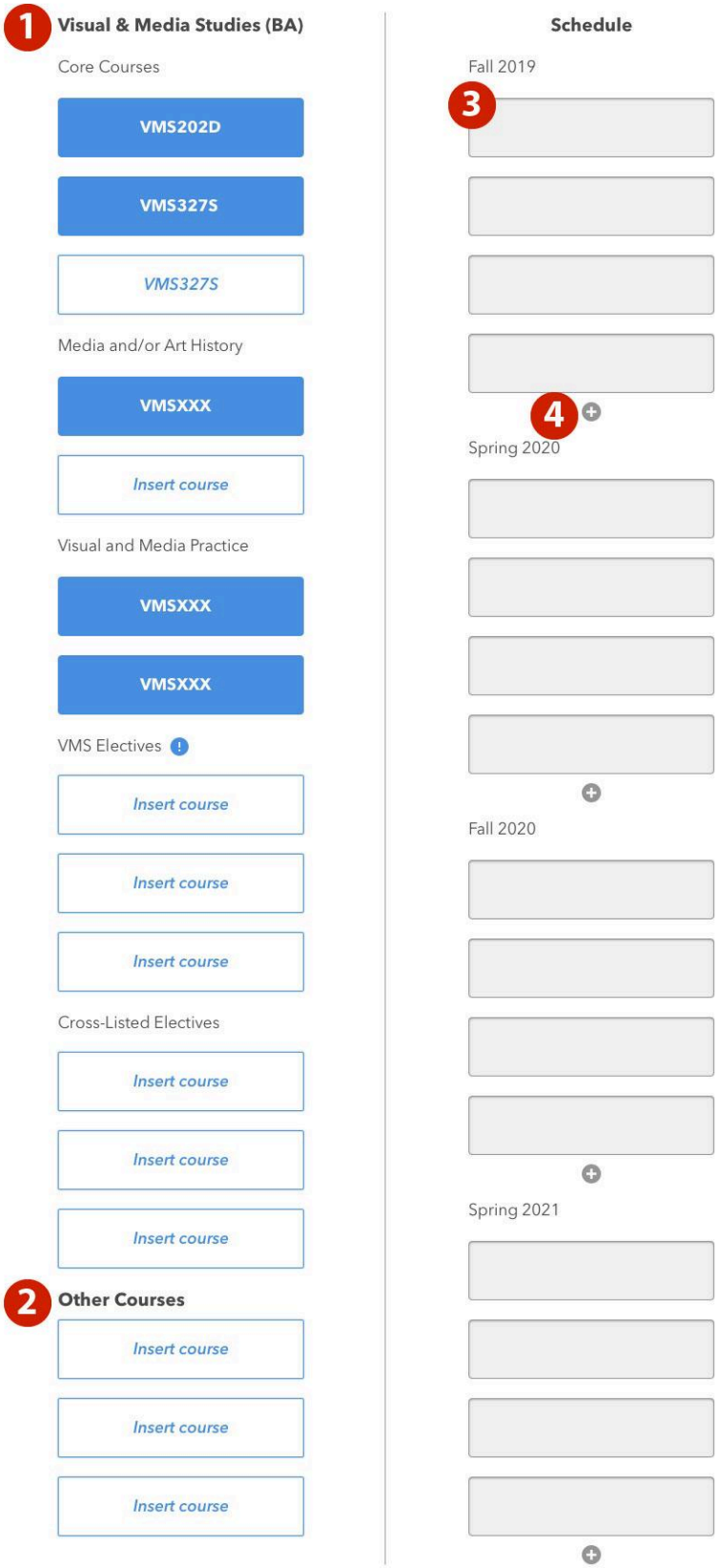
Key:

1. The top part of the page displays filters. Other filters could be added such as Semester Offered, Class Number, etc...
2. The “My Academics” tab shows the student’s current academic plan, i.e their current major/minor/certificate combination. This tab also shows what courses the student has taken to date, as well as what slots need to be filled
3. Example of a core required course that the student has already taken
4. Example of a core required course that the student has yet to take
5. Example of an empty course slot. An idea I was was that when one of these slots is clicked, the search automatically filters to the courses that would fit in this slot. These are drag-and-droppable into the slots in the “My Academics Tab”
6. The warning symbol here refers to certain stipulations surrounding a course in the category. In this example, it refers to the VMS requirement that students must take 3 VMS Electives specifically taught by VMS faculty. This warning will appear on hover.
7. The “Class Search” tab shows the courses as currently filtered. These courses were imported directly from AirTable
8. The “Course Details” tab shows up when a course is selected, and shows more detailed information about the course. Additional information could be added such as semesters offered, example syllabi, etc...

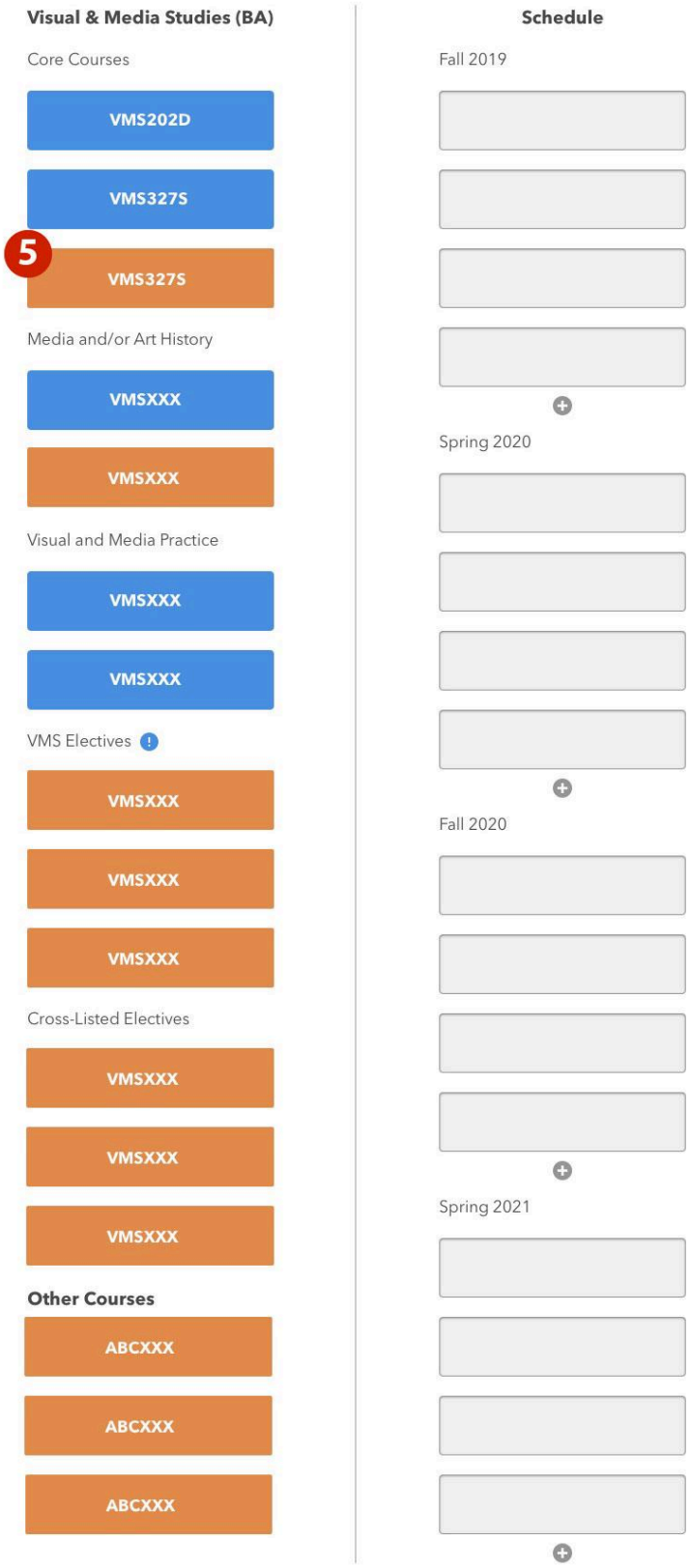
Class Planning/ “When” Page

These wireframes refer to the second step of the process, after students have selected the courses they want to take and need to decide when they are going to take them.

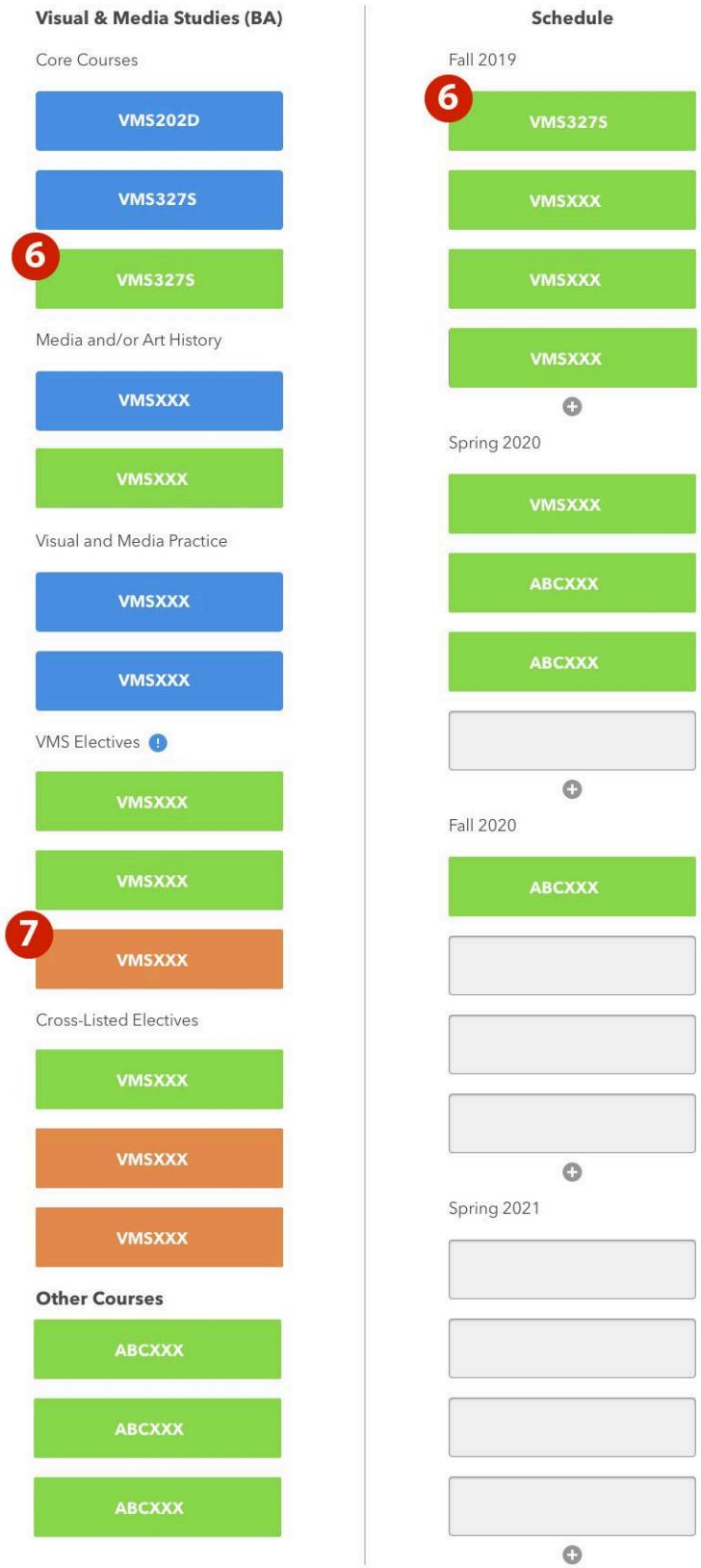
When do I need to take it



When do I need to take it



When do I need to take it



Key:

1. This tab mimics the “My Academics” tab in the “What” page
2. “Other Courses” refers to courses a student wants to take that don’t specifically fit within a major/minor/certificate requirement. For example, if they need a QS course but their major doesn’t require it.
3. Empty schedule slot.
4. The “+” could be to add an extra course slot if a student is planning on overloading OR to add a Summer Session section if the student is planning ons taking courses over the summer
5. The courses in orange refer to courses that the student has selected in the previous step, but has not yet scheduled in their planner
6. The courses in green refer to courses that the student has both selected in the previous step and placed in their schedule. Note that the course is present on both the left and right
7. Orange again denotes courses that have not yet been placed in the schedule planner

VI: Data Visualization and AirTable Blocks

Why Data Visualization

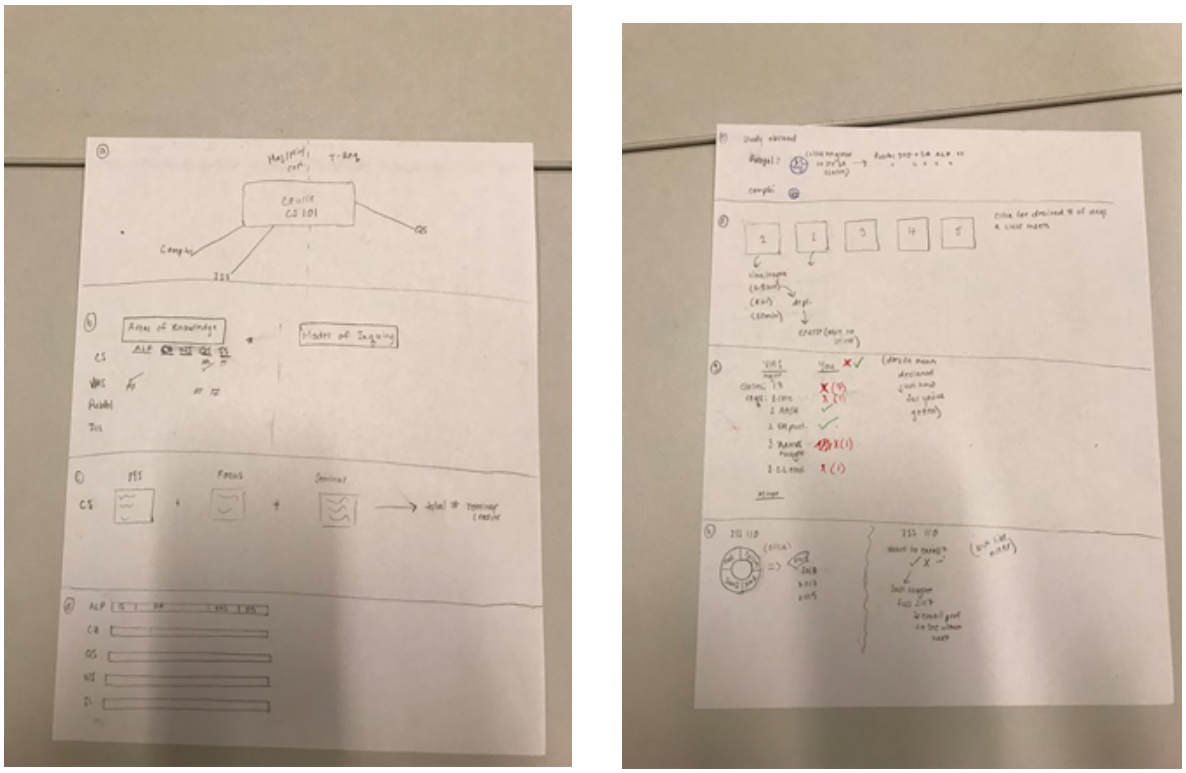
When we began this project, we knew we wanted there to be a data visualization component. Data is only as powerful as its legibility, and visualizations are a huge part of making it easier to read. We felt it was important that in reimagining the academic planning process, data visualizations could make the experience more engaging and interactive. Everyone has heard the saying “a picture is worth a thousand words,” and visualizations are a way of conveying a large amount of data in an efficient way. In addition, visualizations are aesthetically appealing which further makes the data more accessible for viewers to analyze the information and make conclusions/decisions.

As there is so much information and so many things to consider when thinking about majors and various academic programs, some visuals can help put into perspective or offer a new way of looking at the information. Originally, one of our inspirations for data visualizations and visual storytelling was pudding.cool. The site also has a “How-to” section which offers guidance and instruction for how to code and design different visualizations. This gave us a place to start when thinking about what kinds of methods and options there are when we began researching the best platforms to create visualizations.

Initial Brainstorms

Our first task was to brainstorm different opportunities for visualizations with the information we might have. Some of these initial thoughts were graphs showing the distribution of areas of knowledge or modes of inquiries across majors/departments, graphs that show which majors

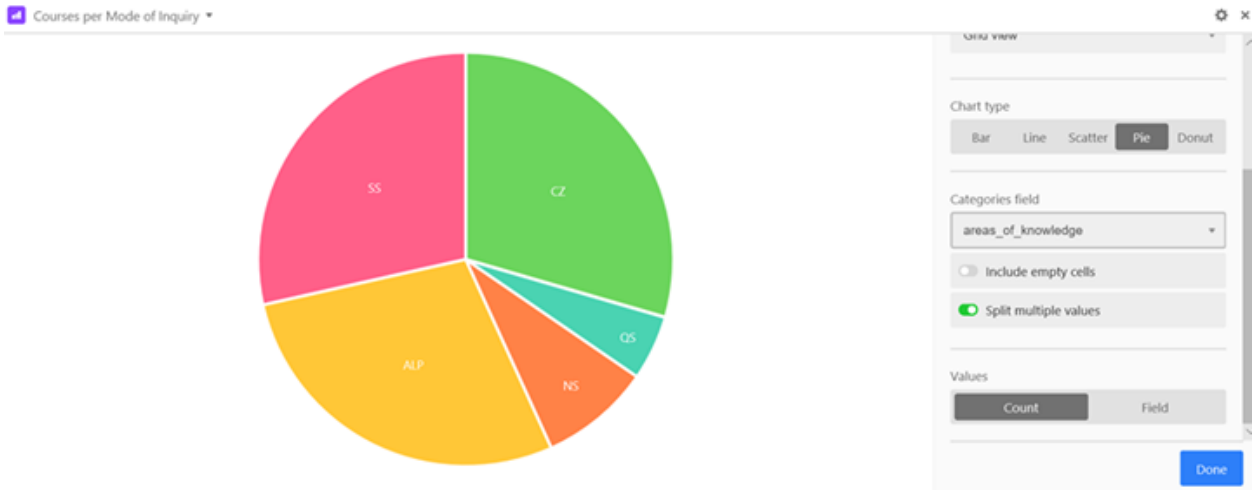
have bigger or smaller classes (or which departments have the most seminar-size classes), or visualizations displaying frequency classes meet in a week. We thought there might also be a neat way to visualize study abroad opportunities in a major. This was a more complex idea, as it involved a visualization where each major would be represented by a globe; the globe would be bigger and smaller depending on how many study abroad classes or programs are there for that major. Clicking on a globe would show where you could take classes and would offer more details. We also thought there might be an interesting way to present how many times/how often a class has been taught in past years. Ultimately, what kind of visualizations we would be able to make was dependent upon how much time we had and what kind of data we would get. Below are some images of sketches of diagrams for graphics we initially planned on pursuing.



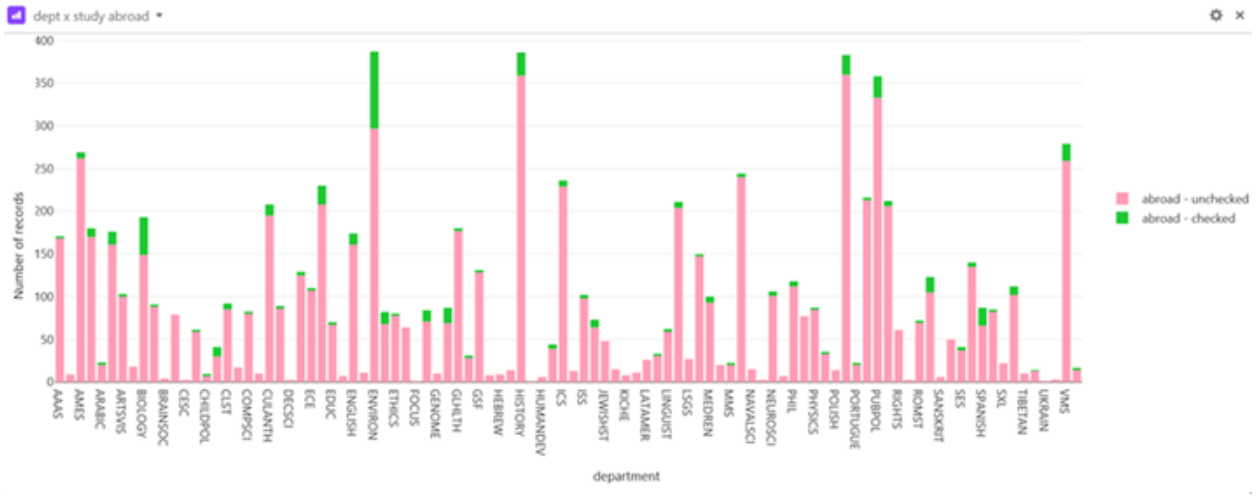
Original sketches for planned graphics.

Airtable Blocks

Once we started getting data, one of the advantageous aspects of using Airtable to collect the information was the ability to use Blocks. Blocks are a feature within Airtable that allows the user to make graphs, charts, maps, timelines, or other kinds of designs using the data in the table. Working with Blocks gave us the opportunity to identify patterns and trends internally, as these graphics were not easily exportable to other platforms. Some of these graphics are included below.

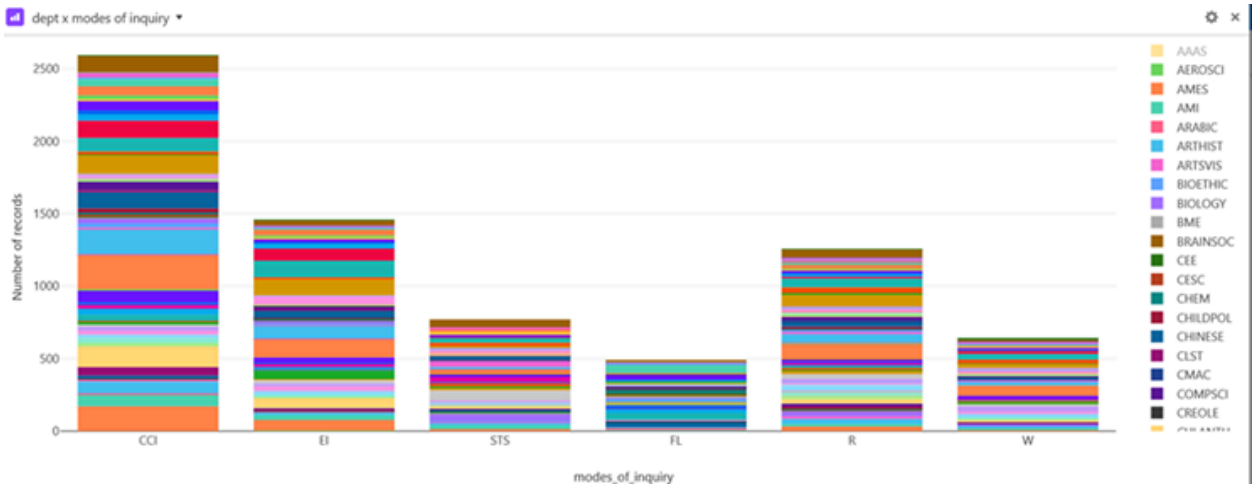


Pie chart showing distribution of modes of inquiry across departments.

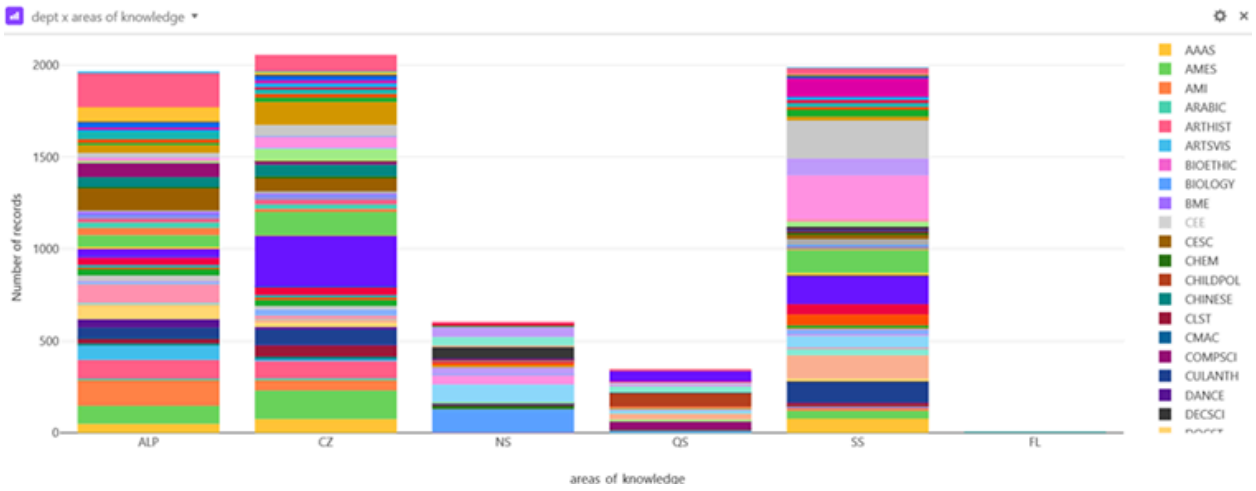


Bar graph of number of study abroad classes per department.

I also created a bar graph that charted number of classes in each department that meet each Mode of Inquiry and a graph that did the same thing for Areas of Knowledge. These graphs are inserted below.

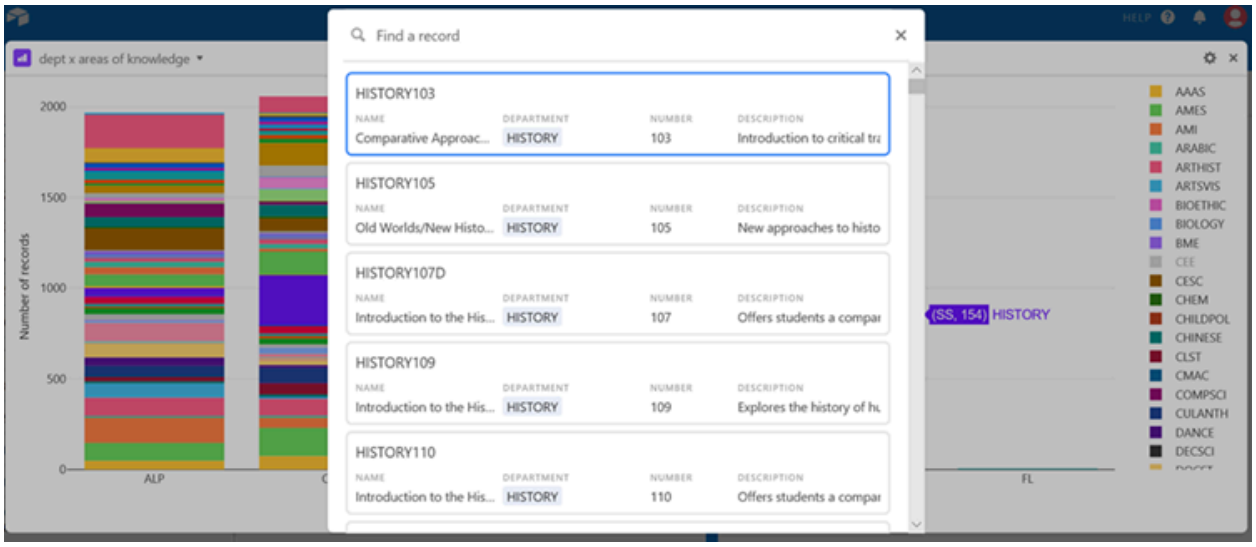


Bar graph of number of classes fulfilling modes of inquiry in each department.



Bar graph of number of classes fulfilling areas of knowledge in each department.

What was particularly useful in Blocks was how interactive these graphics were, as they updated as the data in the table changed. Moreover, hovering over the graph would allow you to click on a specific part and show you which specific records in the table is represented in that section. For example, inserted below is what my screen looks like once I've clicked on the History department color in the Social Sciences (SS) bar in the Areas of Knowledge bar graph.



Screenshot of clicking through to data recorded in the graph.

This image shows a list of all the history classes which are designated as a social science in the areas of knowledge. In a final, more-advanced version of our intended project, this would be an important aspect to try to include if possible because it not only offers a nice way to visualize the data, but it also allows the user to actually interact with the data and get some more detailed information based on which parts of the visualization they are interested in. While this kind of functionality is unavailable in some of the other visualization programs we used, it was nevertheless a useful feature that we were able to use within the system to help inform what kinds of graphics we might want to make and expand on with other visualization-creation programs. It also helped us know how to organize our data to make the most compelling visualizations as we began working with those other programs.

VII: Data Visualizations, Tableau, and RAWGraphs

Cleaning Data In Excel

Cleaning the pulled data from the registrar was an important task in order to use Tableau and Raw. The main focus going through the data included grouping the T-Requirements by major. I formatted an excel spreadsheet to with the Areas of Knowledge and Modes of Inquiry laid out for each course offered in every major. Because we decided to only focus on the departments of Computer Science, Information Science and Studies, Public Policy, and Visual Media Studies, I was able to prioritize cleaning and analyzing the Modes of Inquiry and Areas of Knowledge for those departments. In Excel, I used different equations like SUMIF, AND, OR, REPLACEIF, IF, and more to clean and organize the data. One of the most important cleaning steps was organizing and calculating the total number of Areas of Knowledge and Modes of Inquiry for each department. Below are examples of the tables I created with the data:

Department	EI AVG	CCI AVG	STS AVG	R AVG	W AVG	QS AVG	NS AVG	SS AVG	CZ AVG	FL AVG
COMPSCI	16%	0%	28%	48%	8%	76%	8%	7%	2%	7%
ISS	11%	12%	49%	23%	4%	28%	3%	28%	43%	0%
PUBPOL	31%	24%	11%	22%	12%	4%	1%	78%	16%	2%
VMS	14%	41%	20%	20%	6%	7%	3%	24%	59%	7%

Percentage of Modes of Inquiry, Areas of Knowledge, and Foreign Language by department

code	SS	ALP	NS	QS	CCI	R	W	EI	STS	FL	CZ
COMPSCI89S	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI92L	n/a	n/a	n/a	QS	n/a	n/a	n/a	EI	STS	n/a	n/a
COMPSCI94	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI101L	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI102S	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI103L	n/a	n/a	NS	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI107	SS	n/a	n/a	QS	n/a	n/a	n/a	n/a	STS	n/a	n/a
COMPSCI107FS	SS	n/a	n/a	QS	n/a	n/a	n/a	n/a	STS	n/a	n/a
COMPSCI110	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	STS	n/a	CZ
COMPSCI116	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	STS	n/a	n/a
COMPSCI149S	n/a	n/a	n/a	QS	n/a	n/a	n/a	EI	STS	n/a	n/a
COMPSCI190	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI190FS	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI190S	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI201	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI216	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI220	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI223	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI224	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI230	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI249	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI250D	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI260	n/a	n/a	NS	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI270	n/a	n/a	n/a	QS	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI288	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI290	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI290A	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
COMPSCI290S	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

Example of Modes of Inquiry and Areas of Knowledge by course for Computer Science

Department	CZ	SS	ALP	NS	QS	Total Areas of Knowledge	CCI	R	W	EI	STS	Total Modes of Inquiry	FL
Compsci	1	4	0	5	45	55	0	12	2	4	7	25	0
ISS	17	11	40	1	11	80	10	19	3	9	40	81	0
Pubpol	60	206	25	3	12	306	93	61	40	115	37	346	3
VMS	89	36	186	4	10	325	112	55	16	38	54	275	11

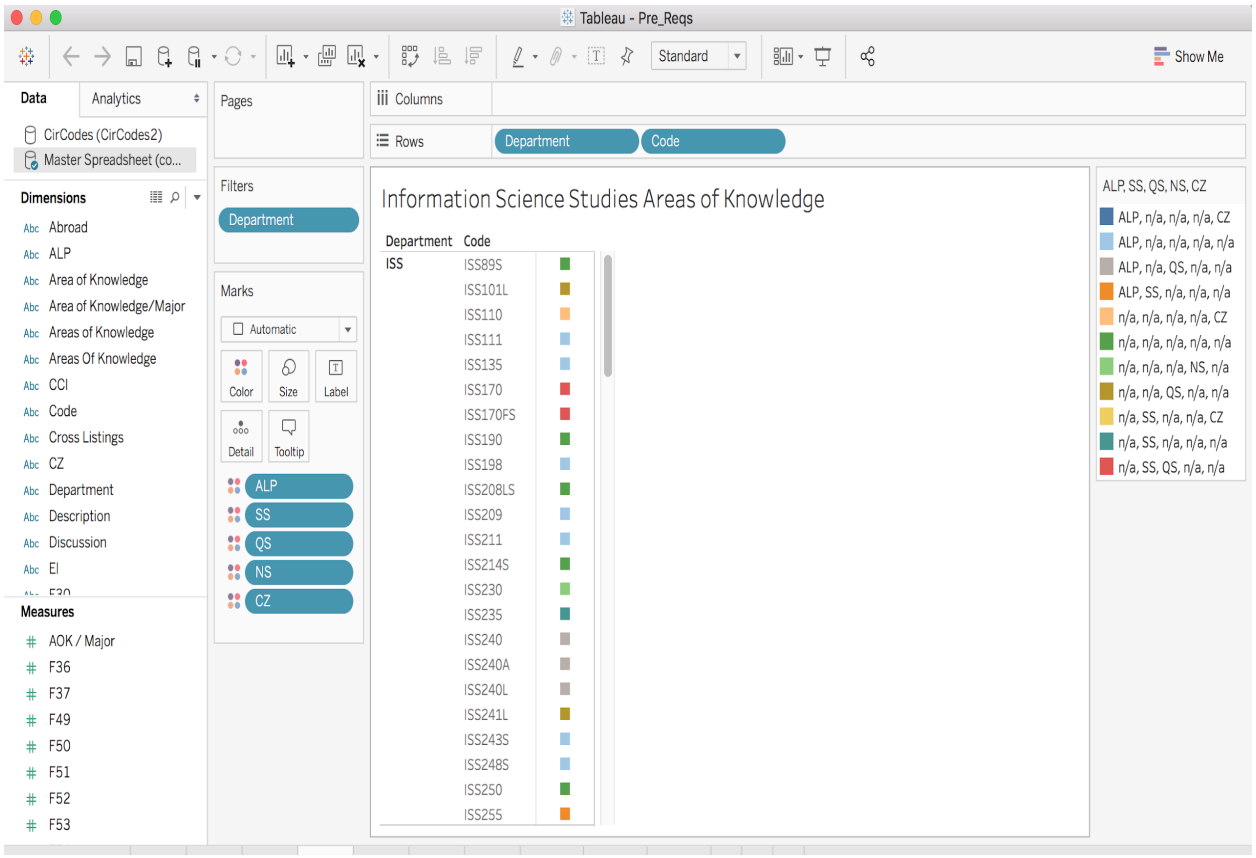
Number of individual Areas of Knowledge and Modes of Inquiry per department

Work in Tableau and RAW

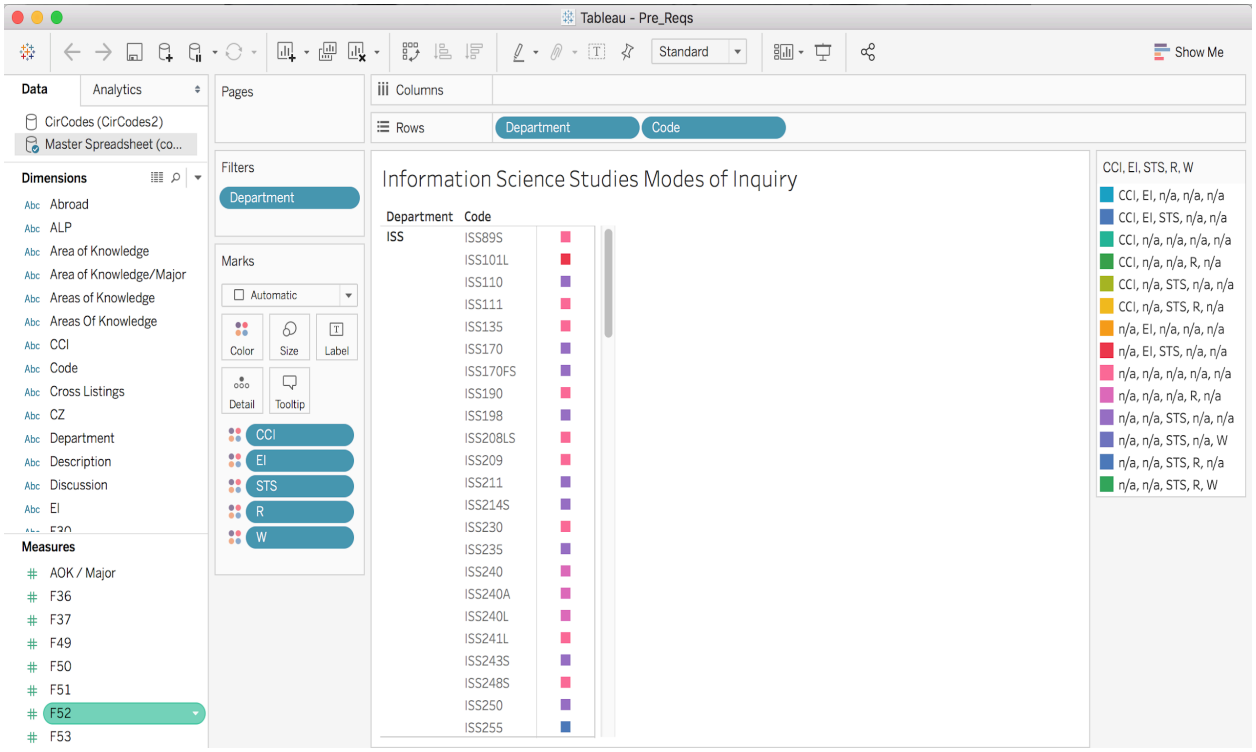
The overarching idea of using Tableau was to visual the data to create a dynamic visual with which users could interact. Tableau is a data visualization application. As the work progressed in Tableau, we that our time was better suited to creating static visuals that could be used as a summary about Modes of Inquiry and Areas of Knowledge for each department.

In Tableau, I was able to visualize the data in different types of graphs like bar graphs, pie charts, and list formats to name a few. Tableau was a useful platform to use to visuals and filter the data beyond a spreadsheet. I was able to create interactive visualizations that showed each of the Areas of Knowledge and each of the Modes of Inquiry for Computer Science, Information Science

Studies, Public Policy, and Visual Media Studies. For example, I was able to filter and color-code courses by which Areas of Knowledge and Modes of Inquiry they included. Below are the Information Science and Studies examples of the color-coded visuals.

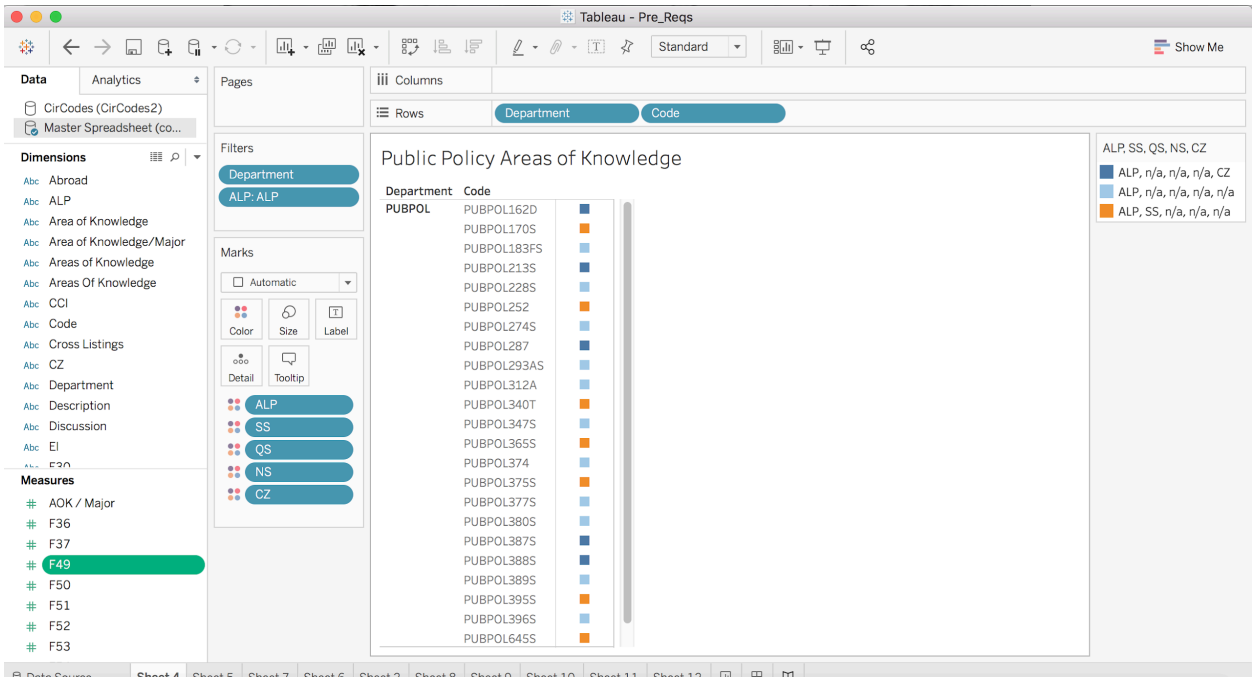


Information Science and Studies – Areas of Knowledge by Course



Information Science and Studies – Modes of Inquiry by Course

The color-coded charts were useful to some extent. We could manually filter to view a specific Area of Knowledge, like ALP which is displayed below.



Public Policy courses filtered to those that included an ALP credit

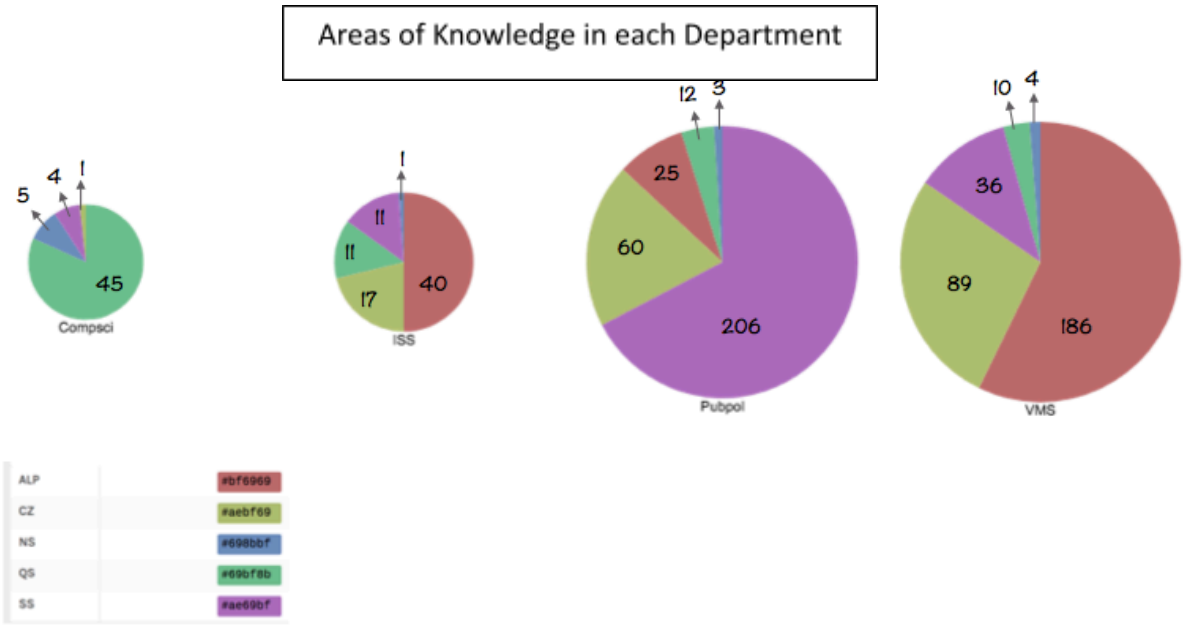
However, this type of filtering is tedious and not a realistic way for students or faculty to browse through courses. Because of this, I decided to pivot my approach and look to summarize the data in more of a static visualization for the departments as a whole. To do this, I began to work on RAWGraphics.IO (RAW).

RAW is a website that pulls in specific data and allows users to interact and build different graphs and visuals. After creating graphs and visuals in RAW, there to download as an image or embed HTML code into a website for graphics. The ability to embed a graphic will be useful as we develop the website component of the project. In RAW, I uploaded the data below in order to create a static, summary-type visual:

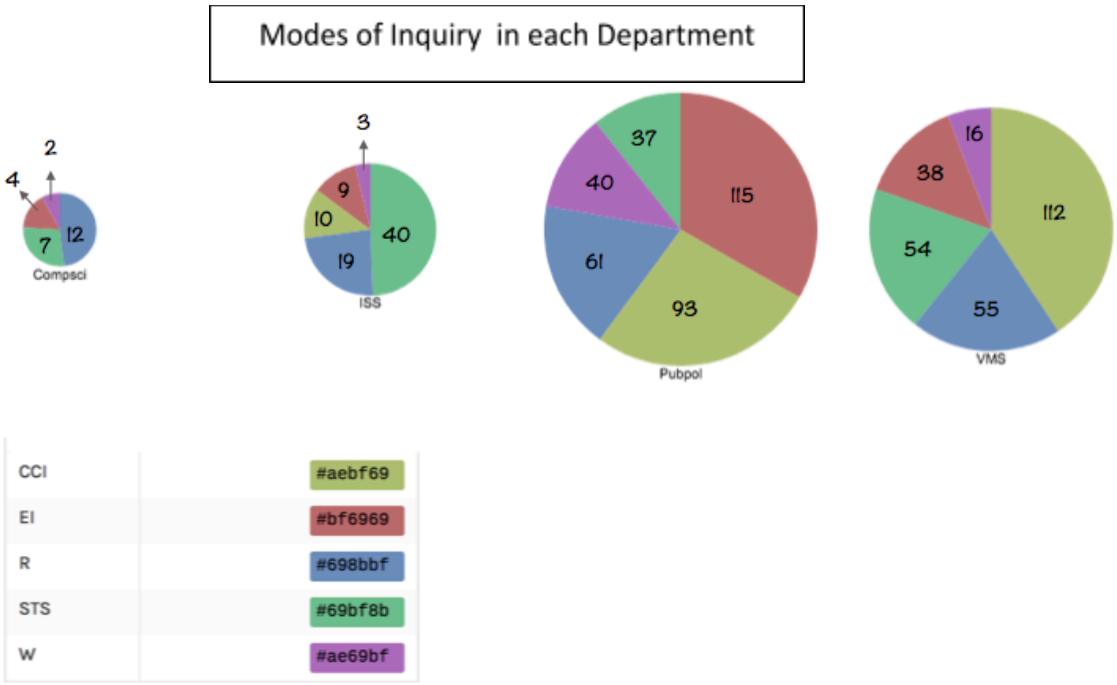
Department	CZ	SS	ALP	NS	QS	Total Areas of Knowledge	CCI	R	W	EI	STS	Total Modes of Inquiry	FL
Compsci	1	4	0	5	45	55	0	12	2	4	7	25	0
ISS	17	11	40	1	11	80	10	19	3	9	40	81	0
Pubpol	60	206	25	3	12	306	93	61	40	115	37	346	3
VMS	89	36	186	4	10	325	112	55	16	38	54	275	11

Data Used in RAW

With this data, I was able to choose different graphs to illustrate the Modes of Inquiry and Areas of Knowledge. I decided a pie chart would clearly indicate how many Modes of Inquiry and Areas of Knowledge in each department. Below are two visuals that I created.

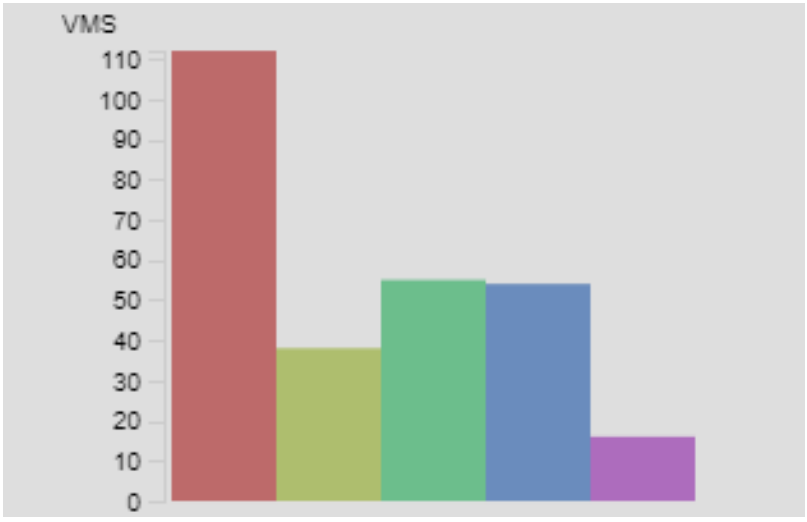


Areas of Knowledge in each department



Modes of Inquiry in each department

In RAW, we were also able to create bar graphs demonstrating the total Areas of Knowledge and Modes of Inquiry. This offered a general way to visualize the data in a summarized format. The graph for Visual Media Studies is below:



Visual Media Studies Modes of Inquiry

Dynamic Visuals

While Tableau and RAW allowed us to explore data visualizations for the course data, the hope is that these visuals can be made dynamic in the future. The static graphs and pictures are great to learn something in a glance but for students to interact with the courses and their attached credits and requirements more interactive visualizations and diagrams will be useful. For example, it would be helpful to click on a Major, select a Mode of Inquiry, and see all the courses that are coded for that Mode of Inquiry. The tableau visualizations got into this a little bit with filtering, but the hope is to take this to the next level.

VIII: Shibboleth and Web hosting

User Management

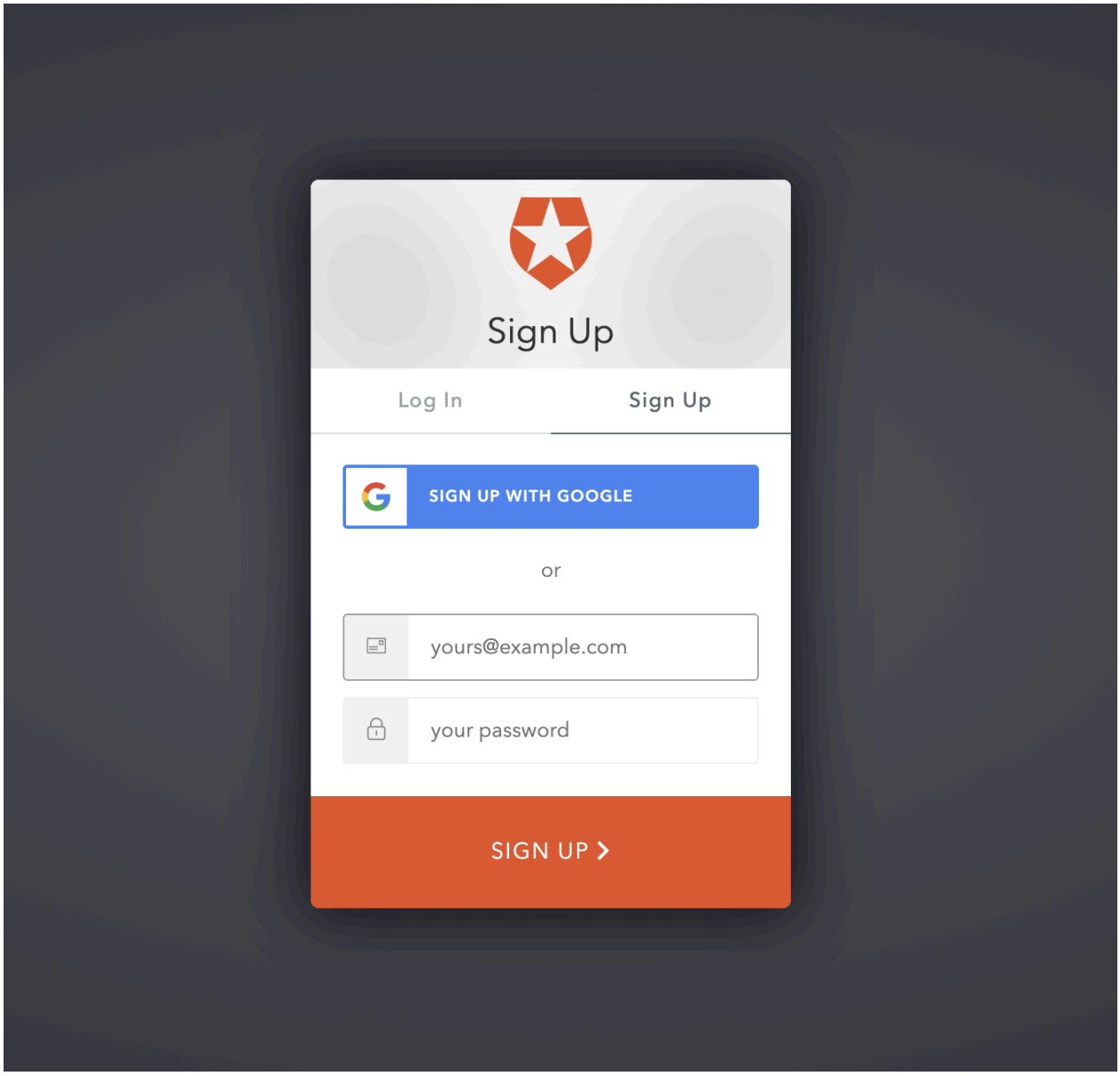
At the beginning of the semester, we decided that Duke's Shibboleth user/identity management system made the most sense to use for our project. Each student already has an account associated with his or her netID, and these accounts contain all the class data from the registrar that we would need for our project. Unfortunately, integrating Shibboleth into our application proved to be much more difficult than we initially anticipated.

In order to begin using Shibboleth (according to this link), we needed to install the Shibboleth Service Provider onto our application server. At this point, although our project data was hosted in Airtable, we had not decided where we would host our actual application. Shibboleth is most compatible with servers running Linux, but only one of a select few distributions (Red Hat Enterprise, CentOS 6, 7 or SUSE Linux Enterprise Server 11-SP3, 11SP4, 12SP3, 12SP4). We decided to opt for the RHEL distribution, reserving a virtual machine through Duke's Virtual Computing Manager (VCM) to function as our server.

Proceeding in installing Shibboleth proved exceedingly difficult without a GUI, and so we began trying to install the GUI on the virtual machine. Duke's documentation explained how to do this for Debian/Ubuntu distributions but not RHEL. After lots of online research and back-and-forth emails with people in the OIT department, we realized that much of what we needed could not be accomplished without access to Red Hat's proprietary services- something which we did not have.

Rather than continue down the Shibboleth rabbit-hole, we concluded that our time would be better spent finding a simpler authentication service for the purpose of our demonstration (although if this project were to be continued, we would urge the developers to further pursue Shibboleth as an identity management system). We settled on a free platform called Auth0 that had exactly the functionality we needed. Furthermore, it could be run on any server using

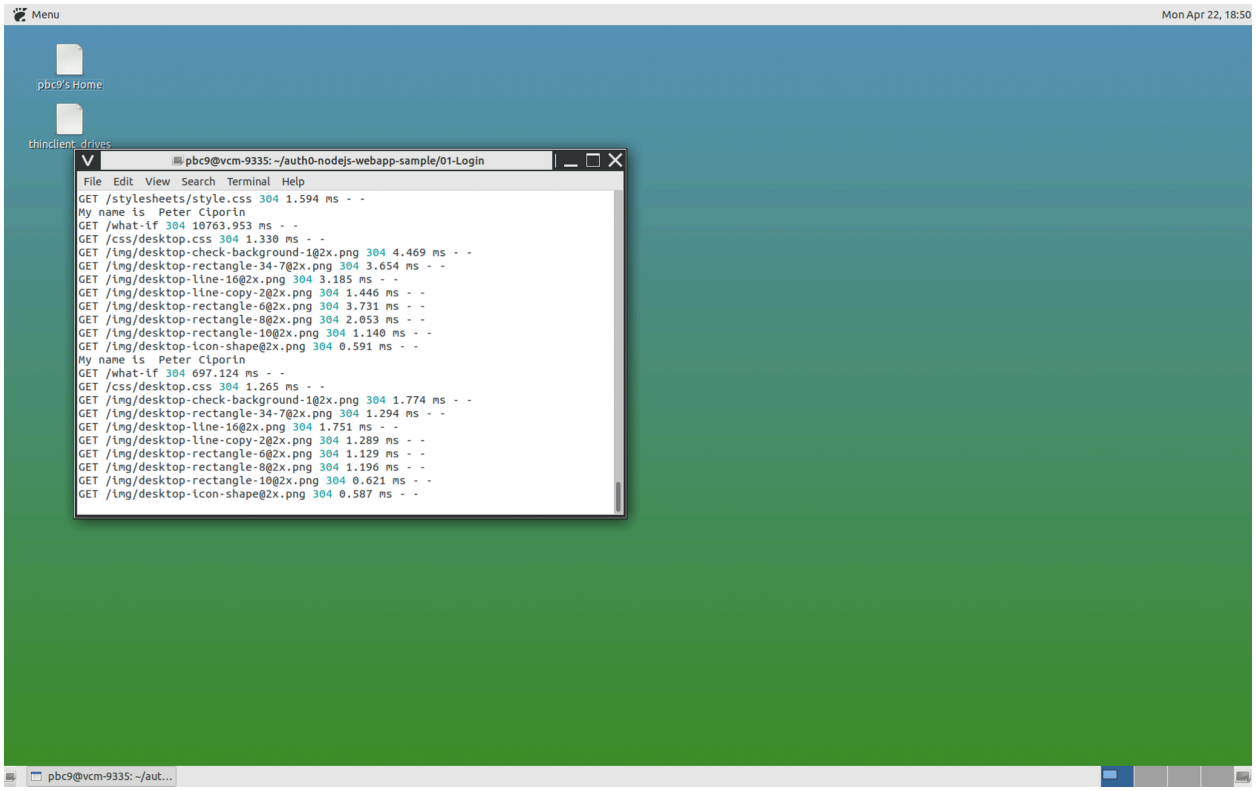
Node.js- a JavaScript runtime environment with which our group was already familiar. An image of the Auth0 login/sign-up process in action has been included below.



We initially intended to host this Node.js application on Reclaim Hosting (where we reserved an instance under the domain betterwhatif.org). Reclaim's support for Node.js proved limited, however. On two separate occasions we needed to contact Reclaim's (thankfully very responsive) support team to perform a hard "reset" on our account (by terminating and recreating it) in order to stop the running app. Rather than continuing to do this whenever our group modified the code in any way, we decided to go a different route.

We soon migrated our code base to a newly reserved virtual machine through Duke- this time running Ubuntu. We modified our Reclaim Hosting codebase so that the domain now redirects clients to the Ubuntu server running our code on Duke's network. We will need to contact OIT to make sure that this server can continue to run beyond the conclusion of this Spring 2019 semester.

The server is set up so that it never automatically powers off, ensuring that the Node.js app is running and listening to client requests on port 80 (the default port for most web clients). Thus, you can reach our hosted app by navigating directly to http://vcm-9335.vm.duke.edu/, however we also set up our betterwhatif.org domain to redirect users to our server as soon as the page loads. Below is a screenshot of the console running on our server (you can see the GET requests that are received whenever a client accesses the web page).



Coding our Tool

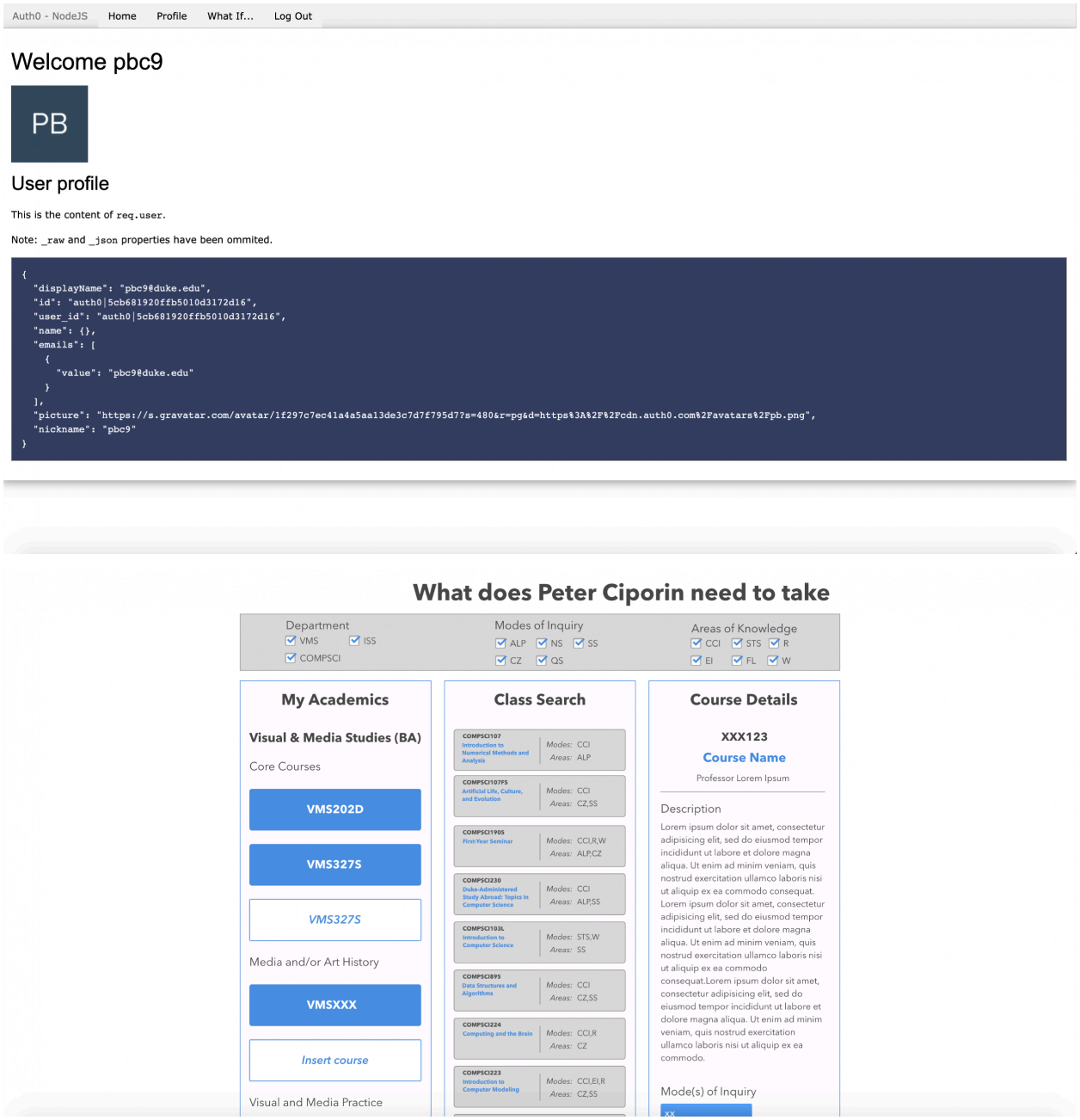
Sierra's design work was done mostly in Sketch, and since she had put such a visually appealing mock-up together already, we programmers were concerned we would have to start from scratch and mimic her work to the best of our ability. It would have been very difficult to put together a page like the one she designed by coding purely in HTML and CSS, so we began by investigating ways to build off of what Sierra had already done.

Some basic research yielded a very helpful tool called Anima that allowed us to export Sierra's work in Sketch directly to HTML/CSS files. These files looked great and were compatible with

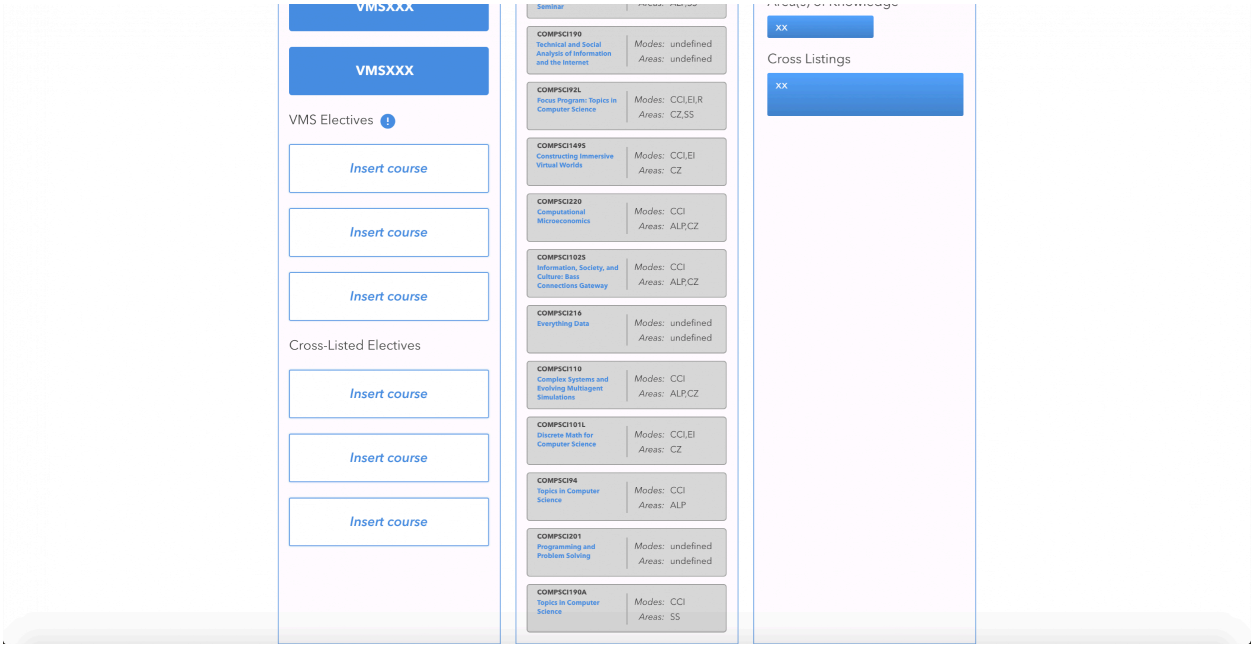
most browsers, but unfortunately, their content was static. To build the What-If tool that we had sought out to, we would need not only to integrate dynamically queried Airtable data, but we would also need to make sure the websites were compatible with the Auth0 app that we had already began building.

The Node.js app that authenticates users through Auth0 uses the Jade view engine, meaning that webpages are rendered dynamically from a .pug file rather than directly from HTML. By using a free online HTML-to-Jade converter, we converted the exported HTML files from Sierra into the .pug file we needed to integrate with the existing Auth0 application. From there, we created the relevant middleware files to route users to the appropriate landing page when they choose the "What-If" tab visible from the landing page (after users have logged in).

When first visiting the site, users will be prompted to register an account through Auth0. Ultimately, it would be best to use Zapier (or a similar service) to load this account data into Airtable, however we did not have the time to implement this feature. For now, any member of the class may create an account using their "netID@duke.edu" email address, and since their netID is stored within Airtable, they should see their full name dynamically rendered when they open the "What If..." tab on the page. The first of the two images included below shows what the Profile tab on Auth0 looks like. The second two images show what happens when a user clicks the "What If..." tab. If we were to have more time, we could have remedied the design inconsistencies between the Auth0 sample app and our What-If tool.

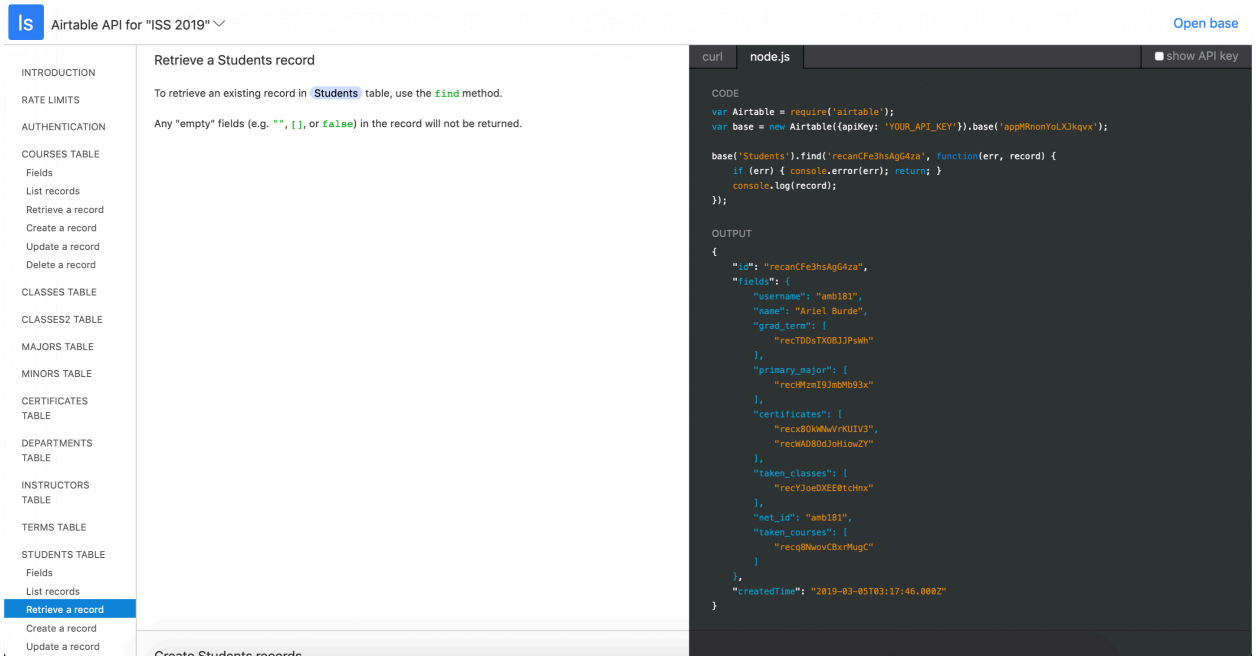

Auth0 - NodeJS
Home
Profile
What If...
Log Out
Welcome pbc9
PB
User profile
This is the content of req.user.
Note: _raw and _json properties have been ommited.
{
"displayName": "pbc9@duke.edu",
"id": "auth0|5cb681920ffb5010d3172d16",
"user_id": "auth0|5cb681920ffb5010d3172d16",
"name": {},
"emails": [
{
"value": "pbc9@duke.edu"
}
],
"picture": "https://s.gravatar.com/avatar/1f297c7ec41a4a5aa13de3c7d7f795d7?s=480&r=pg&d=https%3A%2F%2Fcdn.auth0.com%2Favatars%2Fpb.png",
"nickname": "pbc9"
}

What does Peter Ciporin need to take
Department
VMS
ISS
COMPSCI
Modes of Inquiry
ALP
NS
SS
CZ
QS
Areas of Knowledge
CCI
STS
R
EI
FL
W
My Academics
Visual & Media Studies (BA)
Core Courses
VMS202D
VMS327S
VMS327S
Media and/or Art History
VMSXXX
Insert course
Visual and Media Practice
Class Search
COMPSCI107
Introduction to Numerical Methods and Analysis
Modes: CCI
Areas: ALP
COMPSCI107FS
Artificial Life, Culture, and Evolution
Modes: CCI
Areas: CZ,SS
COMPSCI190S
First-Year Seminar
Modes: CCI,R,W
Areas: ALP,CZ
COMPSCI230
Duke-Administered Study Abroad: Topics in Computer Science
Modes: CCI
Areas: ALP,SS
COMPSCI103L
Introduction to Computer Science
Modes: STS,W
Areas: SS
COMPSCI89S
Data Structures and Algorithms
Modes: CCI
Areas: CZ,SS
COMPSCI224
Computing and the Brain
Modes: CCI,R
Areas: CZ
COMPSCI223
Introduction to Computer Modeling
Modes: CCI,EI,R
Areas: CZ,SS
Course Details
XXX123
Course Name
Professor Lorem Ipsum
Description
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo.
Mode(s) of Inquiry
xx



IX: Connecting the Website to Airtable

While Airtable does not claim to act as the backend for fully-fledged dynamic websites, it certainly provides plenty of tools to make this possible. As soon as you create your database, Airtable begins automatically generating a RESTful API, coupled with an API key, that allows you to read, create, update, and delete records. The API encodes objects using JSON, relies on standard HTTP codes, and allows 5 requests per second before returning a 429 status code and blocking request for 30 seconds. The official API client is airtable.js, which can easily be imported to an external project using Node.js. There are also three community-built clients – airtable, airrecord, and airtable.net – which run on Ruby, Ruby, and .NET respectively.



Our Airtable API

Our API allowed us to quickly integrate Airtable into a fully dynamic website. While we have not yet implemented our desired features, we have tested and laid the foundation for interactions between the user, frontend, and backend. Before rendering a given page, we collect all the needed data in its route file. Once the data is done being fetched, we assign it to variables that are accessible by the accompanying pug file and render the page. An example of this can be found in the website's repository at 01-Login/routes/what-if.js, which retrieves the user's name in Airtable to the be displayed on the "What If?" page.

X: Conclusion

Ultimately, the academic planning process through DukeHub is a disconnected and confusing process. Duke students often need to use other resources to plan their classes and majors. Students need a system where they do not need to leave the DukeHub application because everything should work within the one service. Moreover navigating DukeHub requires the user to click through too many windows and tabs to complete any tasks. There are too many steps that could truly be distilled to just a few. The user journeys generated for this project should serve future developers well in outlining the various places that DukeHub is lacking in the academic planning process. The new design created for this project was built to have the fewest number of pages; in addition, no page is overwhelming with information. Such a succinct version of what DukeHub could look like will hopefully be a productive starting point for those continuing this project.

Working on this project made it abundantly clear that every department has their own take for how fulfilling a degree works. There is so much nuance and so many exceptions in every

department and it becomes really confusing for students. Even the language used to describe majors and programs often overlap in logic and could be explained more simply. The fact that there are hundreds of permutations of majors/minors/certificates and other programs is great for flexibility and having a customized experience for a student; on the other hand, the lack of consistency, and clarity across departments can be overwhelming. Thus from a student perspective, a more structured process for departments to describe their programs would be beneficial. Even academic advisors can get confused because there is so much to know and no efficient way to collect the information. Working on a way to make this information more consistent and transparent would be a great way to move forward so students can make informed decisions.

The data received from the registrar of course offerings and class information was also sometimes frustrating. Some of the data was repetitive or required intense cleaning or understanding to make sense of it. There is certainly room for improvement to make the data more accessible.

One of the other frustrating parts of this projects was trying to work with OIT, as their department is quite disconnected. There were constant challenges in communicating with them and it is not very clear how and when to use their services. We ultimately decided to use a free online alternative to Shibboleth because attempts to implement Shibb never yielded any results because of the difficulty of working with OIT. Future developers should keep this in consideration and perhaps try to partner with OIT early on in their project.

Going forward, visualizations can really transform the academic planning experience. They can show students information about departments and which courses can fulfill their requirements in a visually compelling way. Over time, these visualizations can be more comprehensive and represent more important information in engaging ways. Visualizations illuminate patterns and allow viewers to quickly understand complicated information (as we've learned, academic planning information is remarkably complex).

One of the most important takeaways from our project is that students need the ability to explore academic options in new ways. If future iterations take nothing else from our project, they should at least think about producing a way to see all majors in one place. Right now, every department has their own page/site, and it would be so helpful for students to see all these majors in one place and compare. A lot of students would also benefit from a recommendation system, whether it's recommended based on previous courses or highest rated evaluation at the end of the semester. It would be a great way to highlight new courses (undiscovered gems!) that students might be interested in or might not have knowledge about.

A small way to start to improve DukeHub would be to update course descriptions and information because many are not accurate, which is a disservice to both students and the course instructors and departments. Future iterations of this project might consider reevaluating the process for getting a course taught off of Duke's campus approved for one's curriculum. Right now, this procedure is convoluted, so a streamlined, more informative method for doing so would be advantageous. Similarly, creating a better way for students to explore study abroad programs and which departments certain programs are affiliated with and method for getting non-Duke study abroad programs approved would be greatly beneficial.

After completing a new student interface, if developers have time and energy, they should revisit the staff system. It seems that as it is now, the overhead for submitting syllabi or course information is fairly complicated.

Lastly, what is central to reconfiguring the academic planning process is getting student insight. This student project originated out of personal concerns with DukeHub from each member of the class. After a survey was conducted among over twenty of our classmates, we narrowed in on several pain points of academic planning through DukeHub, particularly the What-If procedure. If this project were to be undertaken by the registrar, working with students should be crucial to the entire operation. Duke Student Government and the Academic Affairs committee could be a great resource, as they are focused on getting student perspectives on all aspects of the academic experience.

Given the concerns about DukeHub laid out through this project and the work we've demonstrated can be done to improve, the team hopes that this project serves as a productive and comprehensive starting point for future developers to upgrade a service that greatly requires reconstruction.