# ClassMaps Design Document

**Team:**
(PM) Adrian Tong: atong@princeton.edu
David Todd: dtodd@princeton.edu
Yang Song: yangsong@princeton.edu
Andy Lin: andyl@princeton.edu

## Section 1 ("Overview")

While many previous projects have worked with the Course Offerings page pertaining to class selection, few projects have offered a way for students to find classes' time and location metadata in an effective manner for general use throughout the year.

ClassMaps will provide an easy way for students to find their classes, as well as interesting lectures to sit in on, using an intuitive and interactive map interface. All of this information is publicly available, but is difficult to comprehend and find without going through multiple steps. By combining data from Course Offerings and OIT's "Places" in a visually appealing format, ClassMaps will provide a platform for users to easily see where(literally where on a map) and when classes of interest meet. Given the potential for students to use ClassMaps as a navigational tool on the go, we also plan to make our interface compatible with mobile phones as a potential second phase for the project.

## Section 2 ("Requirements and Target Audiences")

**Problem**

Currently, the only way to get a course's location is to find the buildings of relevant precepts or lectures and then separately search for the locations of these buildings individually on a campus map. This is both unnecessarily tedious and time consuming. The existing course offerings website and course selection projects do not adequately address the need of seeing the spatial and temporal locations of classes simultaneously.

**Users**

We envision Princeton students using ClassMaps periodically throughout the year, as a tool to find interesting classes to drop in on during free time or to switch precepts due to time conflicts, among other possibilities.

Specifically, ClassMaps will enable users to easily find:

- What classes are available at a given time in a building
- Upcoming and finished classes in a building

- The location of a class and other relevant meta-data
- Available classes and precepts to switch into
- Most searched classes
- “Favorited” classes that the user has marked

With this functionality, ClassMaps will make accessing information and locations quicker and easier thereby filling the gap left by current platforms.

## Section 3 ("Functionality")

**Minimal Features**

ClassMaps will contain a large interactive map of Princeton’s campus with a large search bar and slidable timeline at the top of the page. Users can search for courses in the search bar and a popup menu will appear based on the results filtered by time/location, from which users can select a section. Buildings can also be selected to display the current classes meeting. Finally, a time slider will allow a user to search for classes at different times.

**Extra Features**

ClassMaps will allow users to authenticate through CAS, which will enable them to add courses to their “favorites” list and save classes across multiple sessions, making it more customizable and user friendly. There will also be a section for the “hottest classes” as defined by largest volume of user searches, which will give users an additional means to find interesting courses. Since most people will be on phones throughout the day, we will also try to make ClassMaps compatible with tablets and phone.

**User Cases**

**Scenario 1:**
Bob hears about a cool/fun course that his friend is in and decides to sit in one of the lectures. He opens up our app, searches the class, and quickly finds out the time and location of the next upcoming section.

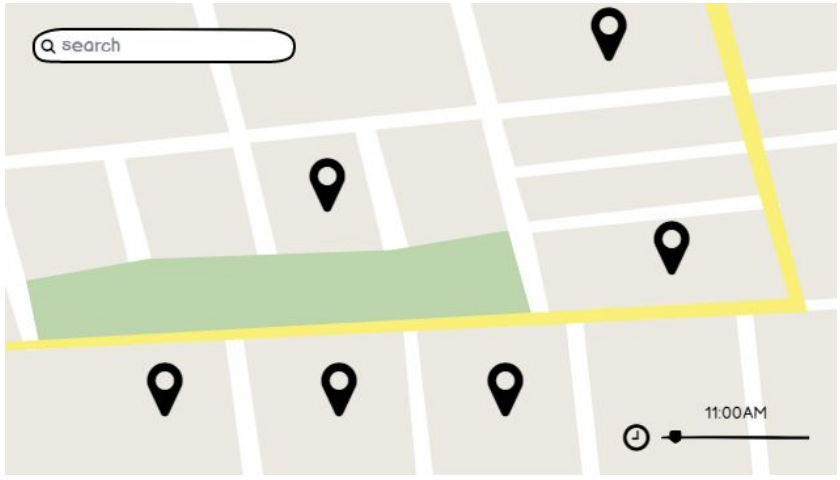
**Scenario 2:**
It is the first week of classes so Sarah does not remember the location of her next class. She opens up our application 10 minutes before class and simply searches for her next class “cos 126.” Without needing to specify precept or lecture, ClassMaps displays the location of her next COS 126 section and relevant data on the map, which enables Sarah to make it to lecture on time.

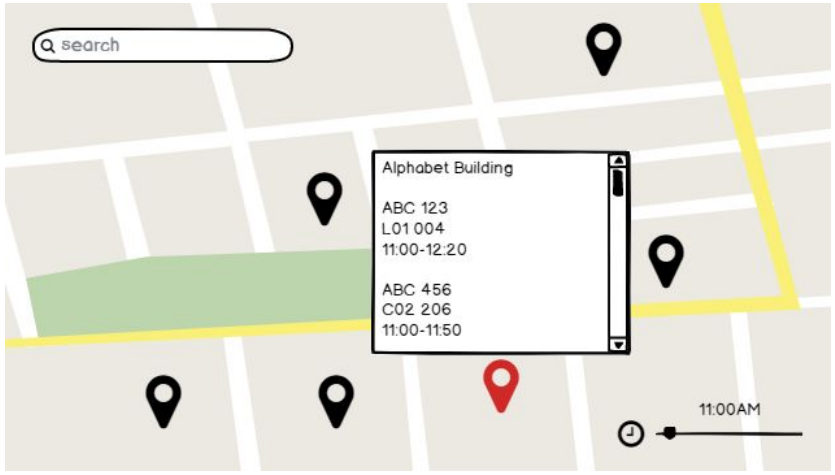
**Scenario 3:**
Joe has an upcoming interview this week which conflicts with one of his precepts and needs to attend another one to make up for missing it. He opens ClassMaps and searches the conflicting class. He sets a time that works for him using the time slider, and quickly finds another section that works with his schedule quickly.

**Scenario 4:**
Hannah arrives to her scheduled class a bit early and sees some interesting notes on the board. She opens up ClassMaps to figure out which the class just got out in that classroom, and obtains all pertinent information about the course.

## Section 4 ("Design")

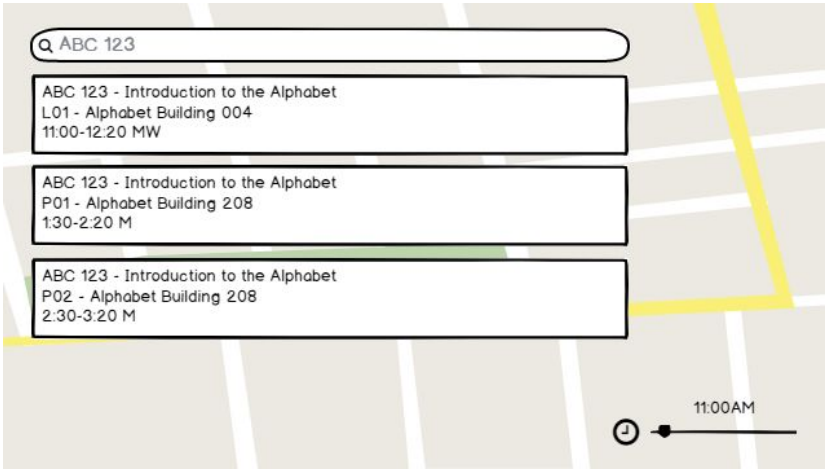
**Client - (HTML/CSS Bootstrap/JavaScript jQuery)**

- Basic UI Mockup
    - Home Page - consists of the map, a search bar, a time slider (default current time), and markers if there is a class at that building at the time given by the time slider:
        - 

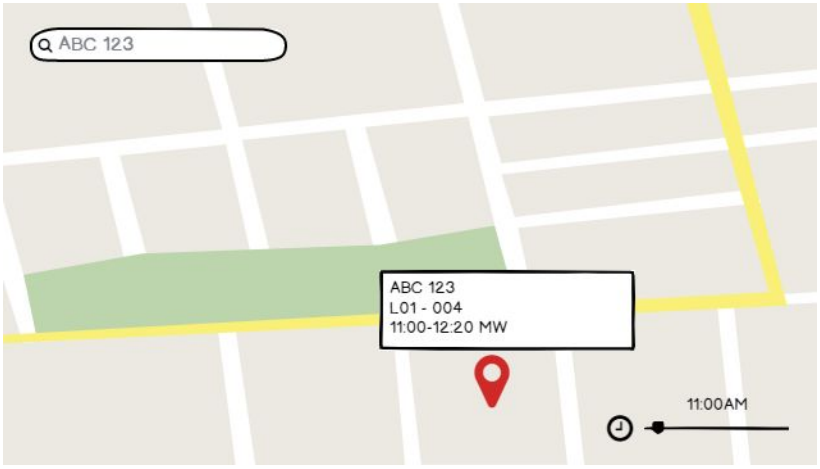
    - Click on a location - gives you the list of all the classes going on in that building right now:
        - 

    - Using time slider - Classes change based on the time on the time slider, although it is also possible to not specify a time:


- Using search bar - menu pops up to select class with options to select the lecture, precepts, etc if the query is legit:




- Clicking on a search - pinpoints the building of the class you just clicked on




The majority of the front end will be Javascript, which would be the natural choice to deal with the visual logic and interactive features of our site, which may get fairly complex as we add features. We

will also need AJAX to display a list of search results as the user types. To create the map, we will use the Google Maps JavaScript API, which allows for scrolling, zooming, click events, creating markers, and many other functionalities.

**Business/Server and Database - (Python/Django and MongoDB)**

**Scraping**

To obtain the data for our app, we will scrape the course offerings provided by the registrar and the location information provided by OIT. The first can be accomplished by modifying the Python script written by Alex Ogier '13 and maintained by Christopher Moretti and Brian Kernighan. Because of changing course information (P99 precepts will not initially have locations assigned, for instance), we will probably have to re-scrape and update our information about once per week. We will then use this script as a model to write our own tool to scrape the OIT feed to get location data. Since buildings are only rarely added or renamed, it should be sufficient to scrape once per semester.

**Maintaining the Database**

Using the Django framework, we will store the scraped information, which includes class department, number, name, time, building, and location, in a MongoDB database. Because MongoDB is a NoSQL database, we will have more flexibility to query our data in many different formats. This database will have a collection of courses, which will store the scraped information and number of queries to gauge popularity, and users (if we implement our extra features). If we choose to have user accounts (instead of giving each student the same interface), we will store recent searches and favorites for specific users. After passing CAS Authentication, the users would have access to edit a database table in which they can store which classes they have "favorited." We can use python to handle the meshing between scraping the two data sources and the final database. This will be somewhat similar to the setup we had in Assignment 4.

**Handling User Queries**

A user query will consist of two possible forms: a string (entered through the search bar) or a building selected by a click. Both will also include the time (or lack of a time) on the slider. These will trigger separate API calls within the Django framework. In the first case, we will split the string on whitespace and treat the tokens as separate terms that could match the class department, number, or distribution and search for courses that match the intersection. In the second case, we search for courses that match the building and time. In both cases, we then search our database for matching results and return the results to the client in a standard form (likely a json).

**Hosting**

For now, we are planning to go with Heroku as our hosting site as it has been proven to work in the past projects, and is a free service.

## Section 5 ("Timeline")

**March 5th:**

- Project meeting with Professor Kernighan

**March 17th:**

- Submitted Design Document

**Week of March 19th (Spring Break):**

- Get familiarized with JavaScript/jQuery, CSS/Bootstrap, Django, MongoDB, and Git
- Set up project repository on Github

**Week of Match 26th:**

- Adapt the script in Assignment 4 as necessary to scrape class information
- Adapt Assignment 4 searching to filter class data
- Create the website and add a minimal UI that features a search bar

**Week of April 2nd:**

- Write database code to store/retrieve scraped data
- Scrape location information from OIT feeds
- Add a map to the minimal UI with dummy class information for each building

**Week of April 9th:**

- Continue to improve the look/design of the front end
- Replace the dummy course/map data in the front end with the real data
- Integrate the search bar and map by marking search results on map and filters search results
- Goal: Prototype with a:
    - Working interactive map
    - Working search bar

**Week of April 16th:**

- Recruit users to get/incorporate early feedback
- Integrate the scraping scripts with the database
- Implement slider for time that changes class information for each building

**Week of April 23th:**

- Begin features beyond core/basic functionality:
    - Add CAS authentication and the option to “favorite classes” so that they stay marked on the map and other customizable features for uses
    - Display which classes are most often searched
    - Make front end smartphone friendly for users on the go
- Goal: Alpha test with:
    - Cleanly working search bar and map
    - Time slider

**Week of April 30th:**

- Incorporate user feedback
- Bug fixing/final touches
- Goal: Beta test on a fully functional version

**Week of May 7th:**

- Prepare a presentation to demo our awesome completed project

## Section 6 ("Risks and Outcomes")

**Project Feasibility**

By having simple core functionality with many possibilities for extra features, we avoid a project that comes together (or more likely than not fails to come together) in one moment. Many previous teams have separately worked with course offerings data and map/locational data, so we feel confident we will at worst complete the basic components of our project while having time to add our other ideas and user suggested features.

**Data**

All the course and location data needed is available through the OIT feeds mentioned above. Because course information may be constantly changing, though, we will likely need to scrape these sources more than once per semester to avoid inconsistencies. Specifically, in the first two weeks, many precepts are labeled P99 and not assigned rooms while people are adding/dropping courses frequently.

**Safety Concerns**

With the prevalence of school shootings in recent years, there is concern about providing too much information about the location of students at a given time. With that in mind, we will be careful to simply extend the information from the registrar to a more visually appealing map format and avoid our project devolving into a “heatmap” of students’ current locations.