Sprint 1 Plan

Team Roles

- Scrum Master: Stephanie Lam
- Product Owner: Hannah Ifekoya
- Developer 1: Aaron Xu
- Developer 2: Neel Singh
- Developer 3: Yuvanesh Rajamani
- Developer 4: Hao Zhu

Customer Meeting Date/Time/Place

- Meeting Date: 1/30/2025 -> Weekly meeting every Thursday
- Meeting Time: 2PM-3PM
- Meeting Place: Online through Google Meets

Summary

Our client needs a more efficient way to assign TAs and graders for classes every semester. The current implementation uses a python script that will take all the applicants and assign the most qualified candidates to classes. However this script needs to be rerun in its entirety whenever applicants drop out, teachers change their preferences or new classes are added in. Our client wants a streamlined approach to this problem. The stakeholders aim to solve this problem by developing a solution that will make it simpler for the client to automatically make assignments and adjustments without the need to rerun the entire algorithm.

Our client also requires that the product makes it easier to apply for positions as a TA/Grader and for faculty members to be able to view applicants to make recommendations. The application should allow applicants to either accept or deny their position if awarded one. The client should then be able to export all the applicant's info into a CSV file for HR purposes. The application will also allow the ability to add classes for new semesters in a simpler way via CSV import.

User Stories

Feature: Prioritize PhD student
As an organizer
So that I can assign all vacant position to PhD student first
I want to make sure PhD students are felt prioritized

Feature: Blacklist student
As an organizer
So that I can secretly penalize student application
I want to punish behaviour of student promise to TA for a position but change in the last minute

Feature: Login via NetID
As an user
So that I can login to save and to check my progress
I want to track and use the functionality securely

Feature: Assign TA to classes automatically
As an organizer
So that I do not have to manually do it
I want to assign graduate students to classes they would be most helpful in automatically

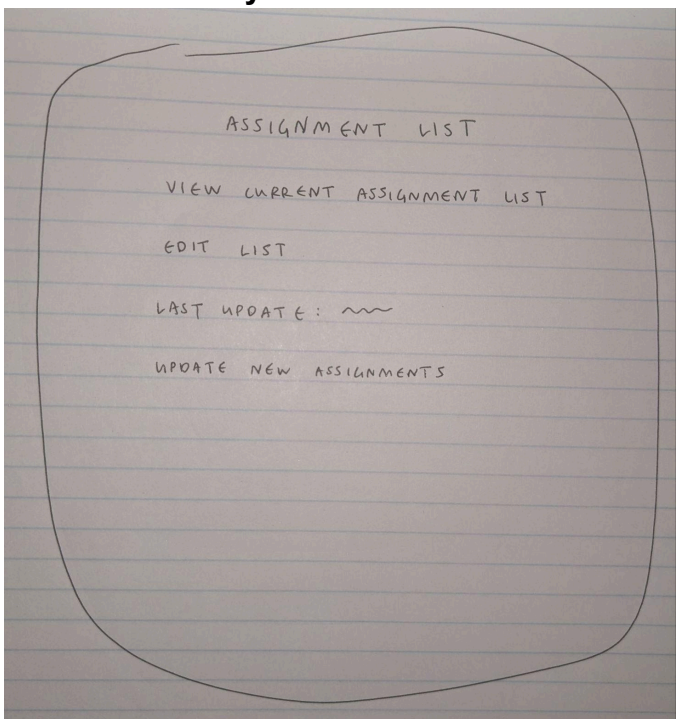
Feature: Recommend student
As a faculty
So that I get the student I recommended as the TA or grader
I want to be able to recommend specific students I want to be a TA or grader in my class

Feature: Applying to be TA or grader
As a graduate student
So that I can TA or grade for a class that I had chosen
I want to be able to TA or grade for a class that I know the materials being taught

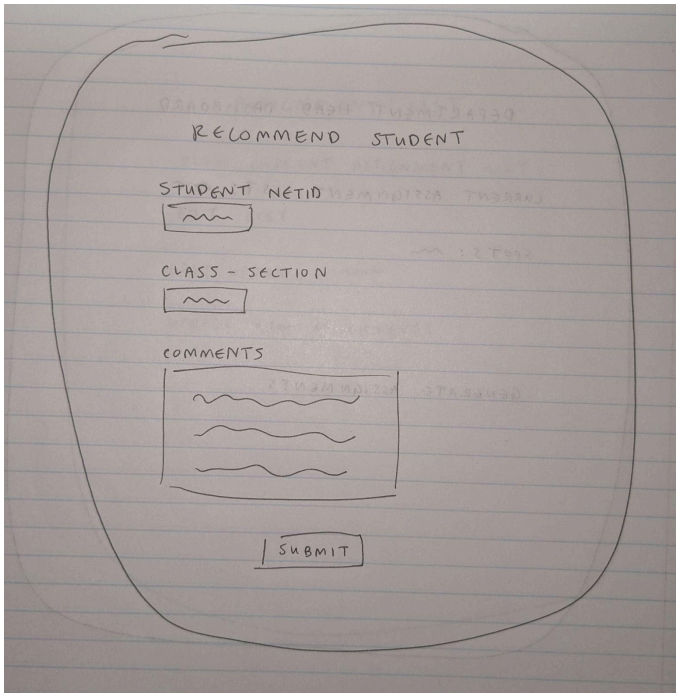
Feature: Exporting hirees to HR
As a department head
So that I do not have to manually send each potential hiree to HR
I want to be able to send it all at once to HR

User Interface

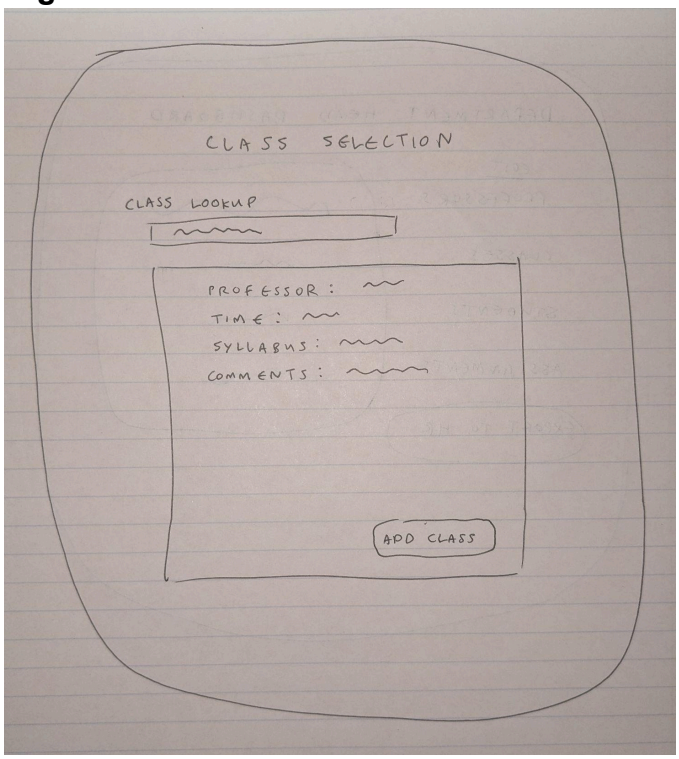
Assign TA to classes automatically



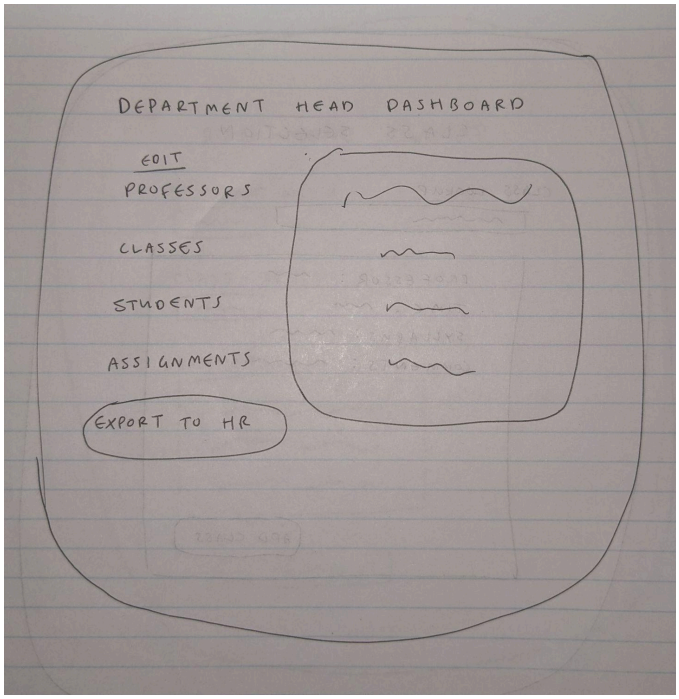
Recommend Student



Applying to be a TA grader



Exporting hirees to HR



Sprint backlog

Goal: To implement the core hiring and TA/grader assignment features along with the login feature in a deployable website/portal.

Sprint Stories:

Feature	Story	Story Points	Assigned To
Creation of website/portal	Set up website/portal framework for faculty and students	4	Hao Stephanie
Login via NetID	Implement the TAMU NetID SSO authentication for users to login to the website/portal	5	Aaron Hannah
Application for students to be TAs or graders	Develop functionality for students to apply to available classes as TAs or graders	3	Yuvanesh
Faculty recommendation system	Allow faculty to recommend students for TA/grader roles	3	Neel

Sprint Time and Task Estimates:

Feature	Task	Estimated Time(Hours)	Assigned To
Creation of website/portal	Set up project framework	3	Hao Stephanie
	Implement basic UI and CSS	5	Hao Stephanie
Login via NetID	Integrate TAMU API authentication	5	Aaron Hannah
	Develop login page for users	3	Aaron Hannah
Application for students to be TAs or graders	Create the application portal	3	Yuvanesh
Faculty recommendation system	Develop interface for faculty to recommend students	3	Neel

	Allow faculty to lookup students and display their info	6	Neel

Links

Github:

- https://github.com/stephanielam3211/CSCE606_Project

Github Project board:

- https://github.com/users/stephanielam3211/projects/2/views/1

Documentation:

- https://docs.security.tamu.edu/docs/identity-security/attribute-services/api/
- http://tagd.cse.tamu.edu/help:css-themes
- http://tagd.cse.tamu.edu/main:layout

Legacy Code Improvement

Introduction and Context:

The client needs a more efficient way to assign TAs and graders each semester. They need an easier application process for TA and grader positions, plus a way for faculty to view applicants and make recommendations. Applicants should be able to accept or decline offers, and the client should be able to export data from our application to a CSV or similar file for HR. These are the core functionalities listed, our main objectives are to improve and implement these to detail and implement a user friendly UI.

Existing Code Base:

This project does not have a pre-developed APP, but rather some existing python code. These codes implement simple logic using only Numpy, Scipy and Pandas libraries from Python,

The code will input and parse several CSV files that contain name, ID, and their application to their desired positions. These files can contain student, professor, and course information. The algorithm provided to us will parse these parameters and return a file with student name, UIN, their assigned section and a calculated score to measure each student's compatibility.

Strategy for Learning:

Addition to learning Ruby, Rails and other tools that help us develop and deploy our APP, we will also need to learn the logic implemented by the currently provided code.

Our sponsor requires us to add additional functionality to existing code. In Addition to being able to assign TAs from a CSV sheet, TA positions need to be adaptive to changes such as people leaving due to sickness or job opportunities. This requires us to understand the current logic of the algorithm, and add function to add, remove, and change TA position on top of the assignment function. Another functionality we have to add is that all PhD students have to be hired. Since the current system assigns a score to every student, some master students can get assigned a higher score than a PhD student which is against the sponsor;'s desired functionality.

We plan to improve upon the previous legacy implementation by first solving the issues our client had with the legacy project. In order to do that we first need to fully understand the python script given to us.

Strategy for Improvement:

The main issue with the previous implementation was that the algorithm would need to be rerun with every change. For example if there are 20 vacancies after the first round of offers went out then those vacancies will need to be filled. However the algorithm will simply run and reassign Graders and TAs to the classes again without taking into account already filled positions. We will improve upon this function through python first since we can invoke the python files through a python API. Then we will try to migrate these to Ruby if conditions allow. The second user story mentioned regarding prioritizing PhD students can be purely done in ruby as we will parse the file again to select the PhD student with the highest score. The seventh

user story mentioned wanting to export all the information to HR with one click rather than manually needing to send each individual's information to HR which we can do by adding a functionality to the code.