Purple Professor Student Tracker Final Report

Project Summary

The Purple Professor Student Tracker is a web-based application designed to allow university professors to manage and track students and teaching assistants enrolled in their courses. The main customer need was a simple, clean platform where professors could add, edit, and manage student enrollment data without manual spreadsheets or ad-hoc systems. The application also allows bulk CSV import of students, tracking enrollment statuses, whether students had a previous class with the professor, and searching/filtering by different attributes.

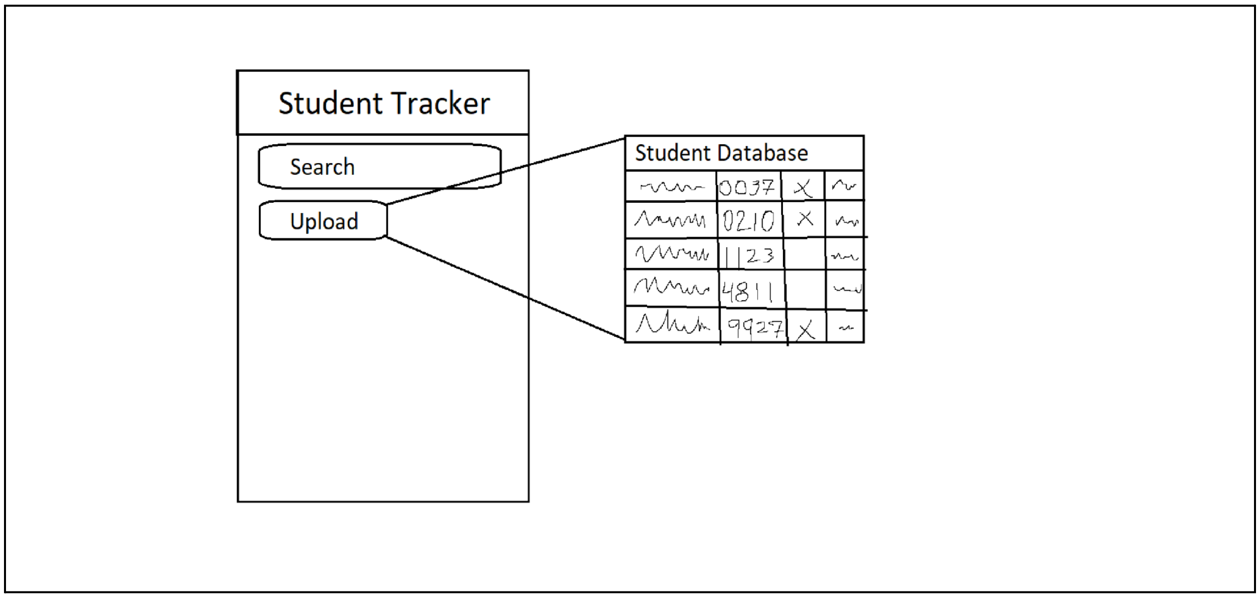
Our stakeholders included the professors who would use the system, administrators who might review enrollments across departments, and future student workers who might help maintain or extend the system. Compared to the Iteration 0 plan, the project evolved to prioritize a more reliable CSV import flow, user role management (Professor vs Admin), and better search/filter functions. In addition to this, we implemented Google Oauth to prevent sensitive data from being accessed by unauthorized individuals. Overall, the current project meets all requirements set by our stakeholders creating a functional student management app.

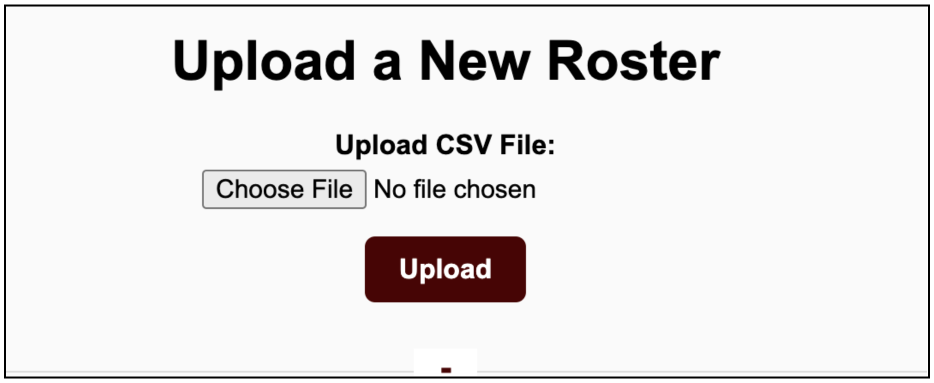
User Stories

Summary	(Story point estimate)	Implementation Status	Changed
Create student records table in database	3.0	Changed	Records table became obsolete due to database overhaul
Upload student records button	3.0	Complete	
Online Deployment of the App	5.0	Complete	
Add a CSV file upload button have the CSV stored locally	1.0	Complete	
Create a parser for the CSV and upload parsed data to database	2.0	Complete	CSV Parser updated for database overhaul
Add testing for the CSV parser	3.0	Complete	
Create a view for the student records table in the database	1.0	Removed	Records table became obsolete due to database overhaul
Create a CSS file to style the student records table	1.0	Removed	Records table became obsolete due to database overhaul
Add testing for the student records table view	2.0	Removed	Records table became obsolete due to database overhaul
Add a sort by button and CSS for styling	3.0	Changed	CSS updated continuously throughout the project
Add functionality to the sort by button with name, ID, and email sorting	2.0	Complete	
Add testing for the sort by functionality	2.0	Complete	
Add test for teachers controller	1.0	Complete	

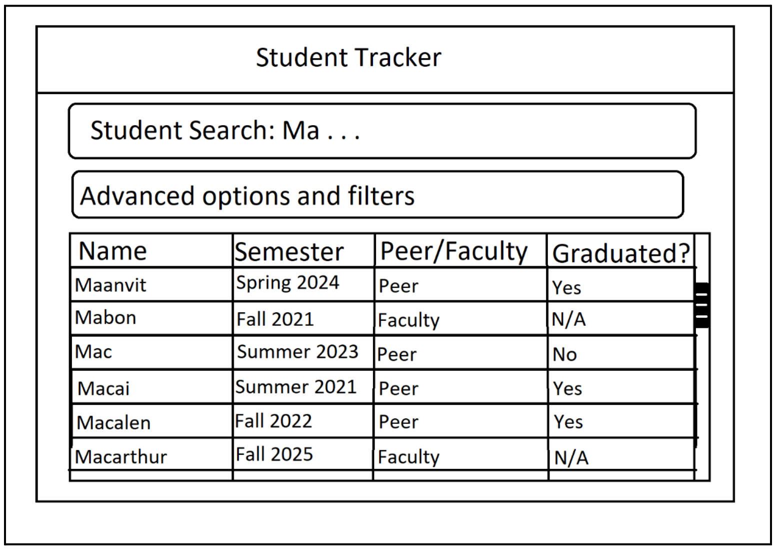
Refactor database to update design of the database.	5.0	Complete	
Integrate Google OAuth with the application	4.0	Complete	
Search function	3.0	Complete	
Update views for peer teachers	2.0	Complete	
Add CRUD functionality for peer teachers	3.0	Complete	
Help with Model update and testing	2.0	Complete	
Create Cucumber and Rspec tests for OAuth system	2.0	Complete	
Testing new views and buttons	2.0	Complete	
Test PT enrollment	1.0	Complete	
Test search function	1.0	Complete	
drop down for enrollment	1.0	Complete	
drop down for pt_enrollment	1.0	Complete	
display enrollment in student view page	1.0		
statistics	4.0	Complete	Updated to include new statistics
improve css over all pages	4.0	Complete	
graphs	5.0	Incomplete	Did not implement graphs due to time constraint
peer teacher cvs parser and test	1.0	Complete	
graphs need to be done for all letter grades	2.0	Incomplete	Did not implement graphs due to time constraint
update csv parser to read filename and generate random letters grades	2.0	Changed	Replaced randomly generated grades with N/A placeholder
test updated roster parser	1.0	Complete	

Lo-fi UI mockups and corresponding screenshots:

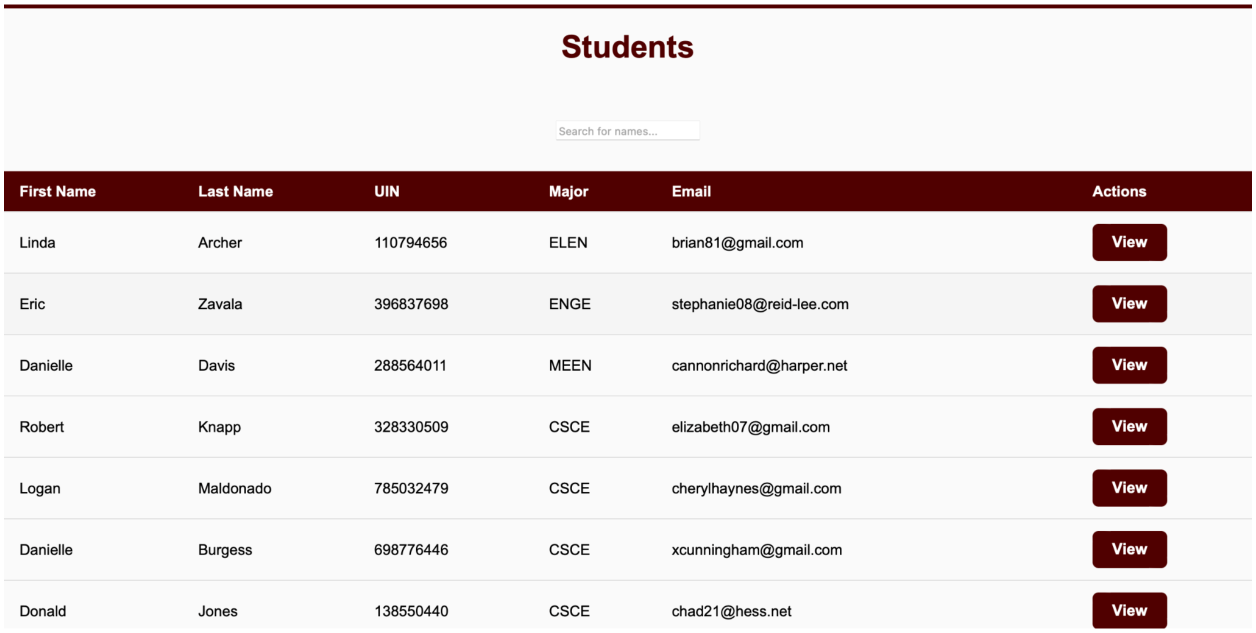



Upload a New Roster

Upload CSV File:



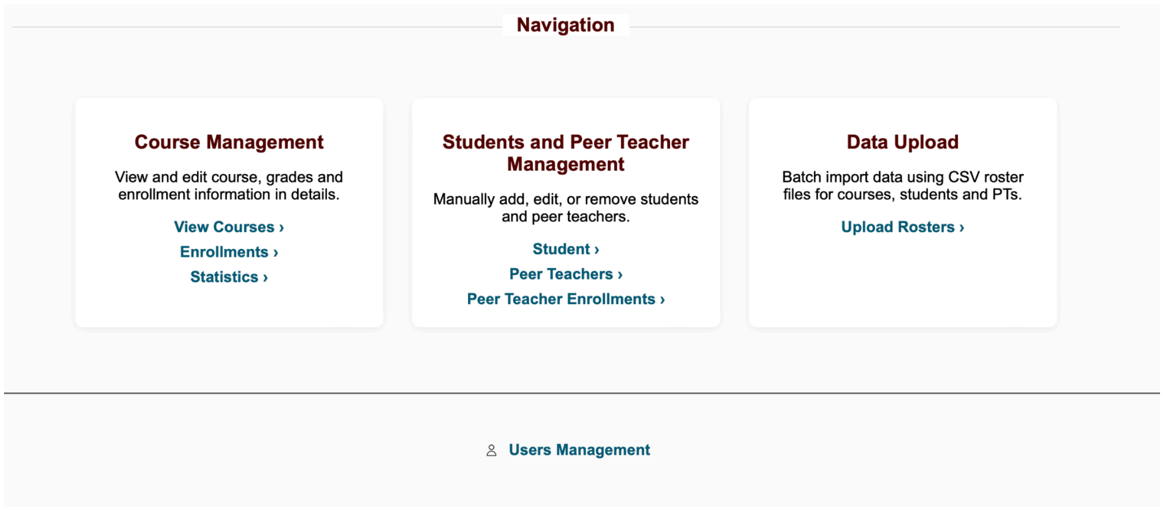
Name	Semester	Peer/Faculty	Graduated?
Maanvit	Spring 2024	Peer	Yes
Mabon	Fall 2021	Faculty	N/A
Mac	Summer 2023	Peer	No
Macai	Summer 2021	Peer	Yes
Macalen	Fall 2022	Peer	Yes
Macarthur	Fall 2025	Faculty	N/A



Students

Search for names...

First Name	Last Name	UIN	Major	Email	Actions
Linda	Archer	110794656	ELEN	brian81@gmail.com	View
Eric	Zavala	396837698	ENGE	stephanie08@reid-lee.com	View
Danielle	Davis	288564011	MEEN	cannonrichard@harper.net	View
Robert	Knapp	328330509	CSCE	elizabeth07@gmail.com	View
Logan	Maldonado	785032479	CSCE	cherylhaynes@gmail.com	View
Danielle	Burgess	698776446	CSCE	xcunningham@gmail.com	View
Donald	Jones	138550440	CSCE	chad21@hess.net	View



Navigation

Course Management

View and edit course, grades and enrollment information in details.

View Courses ›
Enrollments ›
Statistics ›

Students and Peer Teacher Management

Manually add, edit, or remove students and peer teachers.

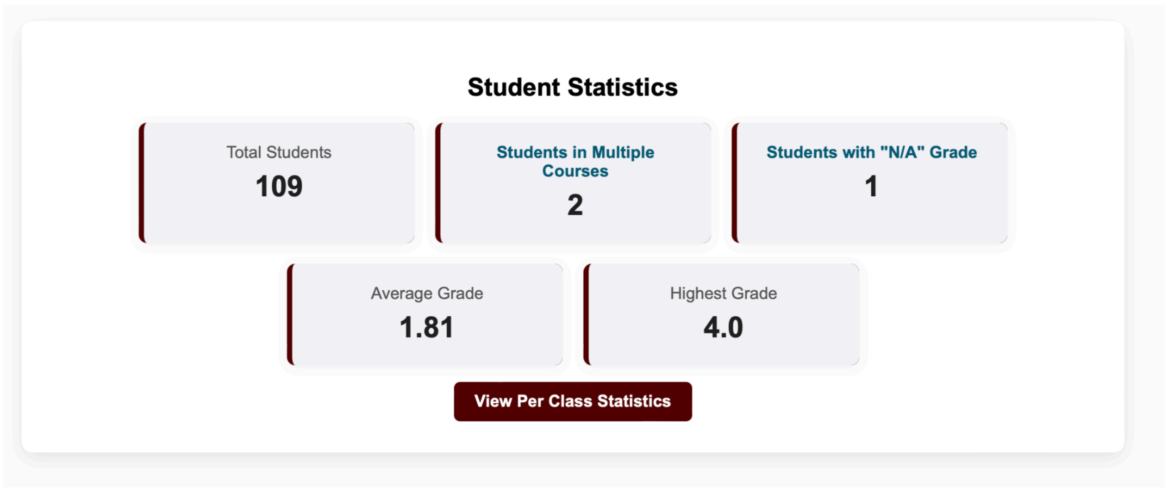
Student ›
Peer Teachers ›
Peer Teacher Enrollments ›

Data Upload

Batch import data using CSV roster files for courses, students and PTs.

Upload Rosters ›

Users Management



Student Statistics

Total Students	Students in Multiple Courses	Students with "N/A" Grade
109	2	1

Average Grade	Highest Grade
1.81	4.0

View Per Class Statistics

Team Roles

Sprint1:

- Adnan Moheddin (Scrum Master)
- Xiaoyun Chu (Frontend developer)
- Steven Luo (Frontend developer)
- Jasmine Pena (Backend developer)
- Evan Wu (Backend developer)
- Susan Ritchey (Product Owner)

Sprint2:

- Adnan Moheddin (Frontend developer)
- Xiaoyun Chu (Backend developer)
- Steven Luo (Backend developer)
- Jasmine Pena (Frontend developer)
- Evan Wu (Scrum Master)
- Susan Ritchey (Product Owner)

Sprint3:

- Adnan Moheddin (Frontend developer)
- Xiaoyun Chu (Backend developer)
- Steven Luo (Backend developer)
- Jasmine Pena (Frontend developer)
- Evan Wu (Scrum Master)
- Susan Ritchey (Product Owner)

Sprint4:

- Adnan Moheddin (Frontend developer)
- Xiaoyun Chu (Backend developer)
- Steven Luo (Backend developer)
- Jasmine Pena (Frontend developer)
- Evan Wu (Scrum Master)
- Susan Ritchey (Product Owner)

Role adjustments:

- After Iteration 1, Scrum Master rotated once due to availability.

Scrum Iterations Summary

Iteration	Accomplishments	Points Completed
1	Basic CRUD (Create, Read, Update, Delete) for students	20
2	CSV Import, Search/Filter functions, data displayed in a table	19
3	Admin View, UI polish, Final testing & bug fixes, add Google OAuth	25
4	Add graphs, add statistics	19

Story Points Per Member

Member	Total Stories	Total points
Haodong Luo	6	18
Adnan Moheddin	4	16
Jasmin Pena	5	15
Xiaoyun Chu	3	16
Evan Wu	2	18

Customer Meeting Dates

Date	Activity
Iteration 1 Feb. 5th 4:00PM	Discussed initial stories, showed Add Student mockup
Iteration 2 Feb. 26th 4:00PM	Showed the Demo,got PT uploaded files
Iteration 3 Mar. 26th. 2:45PM	Demoed working CSV upload and student list filtering
Iteration 4 Apr. 6th. 4:00pm	Full system walkthrough and feedback gathering

BDD/TDD Process

We practiced Test-Driven Development (TDD) for new models, services, (especially the CSV importer) and every added feature.

We also followed Behavior-Driven Development (BDD) using Cucumber for major user flows. For behavior-driven development, we used lo-fi UI mockups as they can be utilized to easily communicate application functionality with our client.

Benefits:
- Helped catch bugs early, especially CSV parsing issues.

Problems:
- Writing cucumber feature files took more time than expected. Sometimes UI changes made existing feature steps break.

Configuration Management

- GitHub for version control
- Heroku for deployment
- CodeClimate for code quality
- SimpleCov for test coverage (goal >90%)

Spikes:
- We had no spike work for this project

Branches:
- We had 31 branches total merged into our ‘main’ branch for the entire project after peer reviews

Releases:
- v1.0 (First CRUD complete)
- v2.0 (CSV import and search/filter)
- v3.0 (Admin dashboard created, CSS added)
- v4.0 (Modified Homepage and Statistics added)

Production Release Issues

We had some small issues setting up Heroku the first time. Missing an add-on, forget to set up an environment variable, didn't add an admin member for login use Google Oauth. But all issues were resolved in one day.

Tools / Gems Used

Tool/Gem	Benefit / Comment
Rails 8.0.2	Latest features, secure and modern
Puma	Default performant web server for Rails
PostgreSQL	Required for Heroku deployment
Propshaft	New lightweight replacement for Sprockets
Devise	Manages user sessions (Professors/Admins)
Omniauth-Google-OAuth2	Enables Google sign-in
dotenv-rails	Manage sensitive credentials (OAuth keys)
Cucumber-Rails	High-level behavior-driven tests
RSpec-Rails	Unit, model, and controller tests
Capybara	Simulates real user interactions
Selenium-WebDriver	Used for feature testing
SimpleCov	Measures % of test coverage
Shoulda-Matchers	Simplifies common Rails tests
Rails-Controller-Testing	Supports controller tests after Rails 5 deprecation
DatabaseCleaner-ActiveRecord	Ensures clean state between tests
Brakeman	Catches potential security vulnerabilities
Rubocop-Rails-Omakase	Rails official Ruby style guide
Web Console	Live Rails console in the browser for error pages
Bootsnap	Speeds up app startup

Other Tools :
GitHub: Version control, Collaboration via pull requests
Heroku: Production hosting, Deployed live site
CodeClimate: Code quality metrics, Monitor maintainability
Jira: Project management board, Track user stories and tasks

Repository Contents & Deployment Process

- app/: MVC structure (models, controllers, views)
- features/: Cucumber BDD tests
- spec/: RSpec unit and controller tests
- .env: Google Oauth client variables

Local Setup

Clone the Repo

```
git clone
https://github.com/Steven7zzz/CSCE-606-SP2025-Purple-Professor-Student
-Tracker.git
cd CSCE-606-SP2025-Purple-Professor-Student-Tracker
```

Install Gems

```
bundle install
```

Setup Database

```
rails db:create
rails db:migrate
```

Set up a .env file for Google OAuth

Create a .env file in the root of the project with the following content:

```
GOOGLE_CLIENT_ID=your_google_client_id
GOOGLE_CLIENT_SECRET=your_google_client_secret
```

Run Locally

```
rails server
```

Access at http://localhost:3000.

Deploy to Heroku

Login to Heroku

```
heroku login
```

Create a New App

```
heroku create purple-prof-tracker-2025
```

Add PostgreSQL
```
heroku addons:create heroku-postgresql:hobby-dev
```

Push Code to Heroku
```
git push heroku main
```

Migrate Database
```
heroku run rails db:migrate
```

Setup Google OAuth (Required!)

This app **uses Google OAuth** for login.

You **must set** the following environment variables on Heroku:

```
heroku config:set GOOGLE_CLIENT_ID=your_google_client_id
heroku config:set GOOGLE_CLIENT_SECRET=your_google_client_secret
```

If you don't want to use Google OAuth, you must replace the authentication system.

Otherwise, users will **not be able to log in**!

(*Tip:* you can generate these credentials via the Google Cloud Console.)

Important Links:

Github Repository
Deployed App
Project Management Page

Presentation Video Link
Demo Link