
SE 492 Bi-Weekly Report 05

Start Date - End Date: Oct 26th - Nov 8th 2021

Group Number: 19

Project Title: Canvas LTI Student Climate Dashboard

Client: Henry Duwe

Advisor: Nick Fila

Bi-Weekly Summary:
The overall goal for this increment was to 1) Integrate the Frontend, Data Analysis Pipeline, Canvas API Wrapper, and Backend Databases all together 2) Finish up our first iteration of the MVP for our project. We were very successful in both and are extremely proud of the work the entire team has done to make this happen. We have the baseline MVP of our product done and just need to make a few more improvements to produce our MVP for the semester and hand off the product to our client.

Past Week Accomplishments:

1. ***UI Team (Kira, Emma, Josh)*** -

- Created Client-Side Javascript package for calling and storing graph data from the Data Analysis Pipeline (DAP)
- Integrated Graph to read in stored data and display students
- Implemented the toggling of displaying student data depending on whether the student is checked in a list or not
- Extended the graphing functionality to graph groups of students as well as groups
- Implemented the toggling of displaying group data depending on whether the group is checked in a list or not
- Added proper routing to application infrastructure

2. ***Canvas API Team (Andrew, Emma, Josh)*** -

- Containerized the Canvas API and deployed using K8s objects to our private cloud
- Identified the necessary Canvas Endpoints we need to hit and data needed from each
- Added filterable attributes to some endpoints
- Retrieved aggregate data from multiple endpoints from the canvas api
- Created custom methods for paging with the Canvas API
- Created posts in the Canvas community forum for help on some aspects
- Lots of endpoint research for

- Refactored the entire codebase to cut request and deserialization time in half.

3. ***Data Analysis Team (Zach, Josh)*** -
- Finished the MVP for the DAP and all of its interconnected microservices
- Implemented error/malformed data handling in all scorer services (resonance, achievement, sentiment, engagement)
- Created Python Stub for our Canvas API Service for modularity
- Parallelized blocking GRPC and API Requests to reduce load time to ⅓ of its previous value.
- Updated the stubbed versions of DAP calls with the proper backend/Canvas API Functions
- Set up relational database schemas for all of the data collection and storage we will need
- Created endpoints for storing/getting data out of the databases in python in order to be read by the data analysis modules

Pending Issues:

- Need to find a way to decrease the time it takes to load the application. Although it works fine for our tests of 5 students, it is currently in an unscalable state.

Individual Contribution:

NAME	Individual Contributions	Hours this Increment	HOURS Cumulative
Andrew Dort	- Filterable attributes to url - complete refactor of backend for optimization - custom methods for grabbing page from response header - endpoint research - discussion post on canvas forum - worked with Josh on formatting and grabbing the data he needed for his python program for data analysis.	20	85

Kira (Ashley) Pierce	- Built foundation of the graph's data population to be passed onto Josh for bug fixes and finalization	20	77
Emma Paskey	- Graphing groups of students - Toggling the students to be displayed on the graph (pair programming with Joshua) - Adding routing and updated file structure to application	20	109
Zachary Borchard	- Set up relational database schemas for all of the data collection and storage we will need - Created endpoints for storing/getting data out of the databases in python in order to be read by the data analysis modules	20	95
Joshua Slagle	UI Work: - Created Client-Side Javascript package for calling and storing graph data from the Data Analysis Pipeline (DAP) - Integrated Graph to read in stored data and display students - Implemented the toggling of displaying student data depending on whether the student is checked in a list or not - Extended the graphing functionality to graph groups of students as well as groups - Implemented the toggling of displaying group data depending on whether the group is checked in a list or not Canvas API Work - Containerized the Canvas API and deployed using K8s objects to our private cloud - Helped explore and identify the necessary Canvas Endpoints we need to hit and	55	154

	data needed from each Data Analysis Pipeline Work: - Finished the MVP for the DAP and all of its interconnected microservices - Implemented error/malformed data handling in all scorer services (resonance, achievement, sentiment, engagement) - Created Python Stub for our Canvas API Service for modularity - Parallelized blocking GRPC and API Requests to reduce load time to ⅓ of its previous value. - Updated the stubbed versions of DAP calls with the proper backend/Canvas API Functions		

Comments and Extended Discussion:

Plans for the Upcoming Weeks:

Wrapper Tasks:

- Decrease load time
- Implement Textual Response Getter
- Implement Likert Scale Getter
- Make less endpoint calls

Frontend tasks:

- Finalize Implementation of Filter Save/Load
- Finalize Implementation of Login/Register Page
- Introduce Accessibility Settings:
    - Color Blindness Mode
    - Screen Reader
- Finalize Implementation of hovering over data points to see more information

Data Analysis Tasks:

- Identify commonalities between significant changes in students' resonance. Ideas:

    - Similar words
    - Similar scores on assignments
    - Similar group feedback scores
- Implement Database Wrapper Functions
- Make the data put into canvas tell a story for when we present to people