University of Ontario Institute of Technology (UOIT)
Faculty of Engineering and Applied Science (FEAS)
Department Of Electrical, Computer, And Software Engineering (ECSE)

Software Project Management (SOFE 3490)

Lab #2 Submission | Project Selection

Okwudili Ezeme

Winter 2020

Group Members
Aryan Kukreja (100651838)
Emil Ilnicki (100659072)
Jamieson Leibovitch (100612845)

Project: Class Application

Introduction: Reason for Selecting Project

We have decided to select the Class Application. An important reason for selecting this application as our project is because it provides a significant advantage over the traditional de-centralized student and alumni system that the university currently uses; with this new system, the alumni, student, and management services will be centralized into a single application. In addition, since the web and mobile version of the application will be developed synchronously, it will be easy to coordinate their development to ensure that they are built on the same tech stack (making them easy to maintain and upgrade as required).

We also hope to follow the Agile method of development. This will allow for frequent delivery of increments that the client (the university) can test for errors or bugs, and report back. Each increment will go through testing before it is deployed on the client's side.

Project Objectives

The following objectives need to be met during the development phase of the project. These objectives consist of both **tangible** and **intangible** objectives:

1. The tech stack used to develop the web and mobile application should be similar; this will simplify maintaining and upgrading the server in the future.
2. The system should use the school's *Banner* login feature as authentication for all types of users (management, current student, or alumni), so that the new application is able to communicate with the existing systems of the university.
3. The application should have 3 different modules to differentiate between the 3 different types of users that are expected to use the system. Students and alumni **should not** be able to see the *management* module upon logging in.
4. The system should have a messaging/notification system that allows professors and current students to interact with each other easily.
5. Through the alumni profiles, management should be able to gather statistics that tell them how their alumni are performing in the real world.
6. The application should also allow management **only** to add classes, and to track discussions and messaging amongst students and alumni to see if any abuse/harassment.

Measure of Success

The following list will be used to measure the success of the class application product.

1. The application supports modules for students, management, and alumni.
2. Only the management console allows for the creation of different classes for students.
3. Management is able to track current students and alumni.
4. Management is able to send messages to individuals or classes within the application.
5. Students can create chat rooms that are either one-on-one or in a group, and they are able to contact alumni.
6. Students and alumni are able to use a filtering function that can search for anyone within the application.
7. The student platform is moved to the alumni platform upon graduation.
8. Application development does not exceed the contract budget.
9. Students will be given updates on class cancellations.
10. Students will be able to login with their student or banner ID and password.
11. Positive feedback from using the application

Hardware and Software Infrastructure

The requirements state that it needs to support both web and mobile, which will require us to choose the technologies that support both. The software architecture that we choose to use will require both a frontend/GUI, a server/API and a database to store the students. Although this project will be software-based, it will require hardware localization and cloud tools for deployment. The following sections will outline the technologies that we will use for this project.

Frontend

We have decided to use React Native as our frontend so that we can develop for mobile and web concurrently.

Server

We will be using Go as our HTTP server language, as it has a built-in concurrency model via go-routines. For this application, it will mostly act as an API either by RESTful API endpoints or GraphQL. NGINX will be the reverse proxy for our server.

Database

The database will be relational and will contain user permissions (i.e. management, student or alumni), and data about each user. We will need to handle updates/news that management creates, and allow users to subscribe to them. The database will handle any events for the application such as abuse. We will also need to include a message system in the database so that people can contact each other.

Deployment and Production

We will use AWS to manage our hardware resources such as the servers for the API and database. AWS will assist in the access of scaling during high traffic, as well as offer simple setup for deployment. Combined with Docker, deployment of AWS's VMs can be done through a Continuous Deployment pipeline.

Security

Authentication will be handled by the University, as they contain data in their system about all of the students and staff. We will use SSL encryption to ensure that we aren't getting man-in-the-middle attacks or data sniffing. In order to access the server, we will require a bearer token so that authorization is required to access the server.

Testing

We will implement a Continuous Integration Pipeline using online technologies such as TravisCI. We need to ensure that our application is working as intended and that anytime something changes, the code still passes our tests. We will also require UAT testing using each platform (mobile and web).