SOFTWARE ENGINEERING OF TEACHING TOOLS – A CANVAS APPLICATION

ABSTRACT

Canvas is a website used by instructors and students at UWB to access and manage their courses. Teaching Tools is an application in the early stages of development that integrates with Canvas to provide more functionality for students and faculty. It can be used to simplify connections with external tools such as Google Drive and Microsoft Office, and automate repetitive tasks to increase work efficiency. Current functionality includes features such as setting up course groups with shared Google Drive folders, adding a comment to a group assignment submission that identifies the submitter, and importing course settings from an old section to a new one.

The majority of the work I did on Teaching Tools was refactoring the back-end Python code. I redesigned the architecture to improve the logic and flow of the program and remove redundancies. The original code did not have well-defined layers for accessing the Canvas and Google APIs or for processing the data. I began by reorganizing the structure so that Canvas related functions were separated from Google related functions. The implementations for Canvas API calls were specific to the current features in Teaching Tools, so if a new feature was to be added, the required API calls would need to be researched and implemented, possibly duplicating work. I wanted to provide a simple way to access the Canvas API, so I added an API wrapper which encapsulates a substantial number of Canvas API calls into a single service. This will help streamline the process of interacting with Canvas and allow for new features to be implemented more quickly. In addition, I created a pure data model layer that stores the data retrieved from Canvas so the data can be more easily processed.

The work I did created a solid base for future students and developers to build off. The improved architecture is simpler to navigate and more modular, which will make the software easier to develop and maintain. Furthermore, the work I did greatly increased my knowledge and skill in the Python language, in software patterns and architecture, and many other areas of software engineering.