CS 410 Project Proposal

Project Name: Enhanced LiveDataLab
Project Topic: System Extension

Team Member # 1: Jiayuan Li (jiayuan8@illinois.edu) [Captain]
Team Member # 2: Zehang Qiu (zehangq2@illinois.edu)

1. System Chosen & Main Goal

We will choose the LiveDataLab as the system we will be working on. Our main goal is to incorporate LiveDataLab with a text editor so that students can implement their MPs by using the text editor within the LiveDataLab or submit a file directly to the LiveDataLab just like PrairieLearn.

2. Tentative Functionalities and Expected Output

The functionalities we want to implement include: 1. an online text editor on LiveDataLab so that students can edit code / submit code online; 2. A cache functionality that students can restore from the code where they left last time; 3. A history functionality where students can restore code from their previous submission and view their previous submissions' grade.

The expected output of our implementation is that

(1) Students can edit and submit code via the online text editor and the grader can return the submission's result.
(2) If the student leave the webpage and re-enter, the editor should be able to restore the code (either from local storage or cloud storage, still deciding)
(3) After the grader runs, the student should be able to view his/her previous submissions' grade in an area of the editor's webpage and be able to restore to any previous submission.

3. Communication with Existing System

Following the best practice of the software design pattern, we want to keep the functionalities of our implementation as independent as we can with the existing code of the LiveDataLab. We plan to design several RPC for the text editor to communicate with the existing grading system. We will also design a database schema and implement a data access layer for the communication of our project and Azure database.

4. Programming Language and Frameworks

Since the project we will be working on is a full-stack project including both frontend and backend, we will be using React for frontend implementation and Python for backend implementation. We also aim to make our output compatible with the Microsoft Azure API and database. (Or whatever language or framework that the current LiveDataLab codespace using)

5. Expected Workload (Total workload: 20 * 2=40hr)

	Expected workload
Understanding the Existing Code of LiveDataLab	3hr * 2 = 6h
Integrate LiveDataLab with an online text editor (frontend side)	10hr
Design database schema and data access layer of the functionalities	2hr
Implement data access layer and API (backend side)	10hr
Cache Functionality: Make sure students can restore from the code where they left last time.	4hr
History Functionality: Users are able to restore from each previous submission.	4hr
Test code with different test accounts, test files on both local environment and cloud environment.	4hr
Total	40hr