CS 410 Project Proposal

Team Name: Money

- Evan Eckels (Captain) - eeckels2@illinois.edu
- Bhargav Yadavalli - bhargavy@illinois.edu
- Preston Chao - preston7@illinois.edu

Chosen System: System-Extension LiveDataLab

LiveDataLab Subtopic: More friendly UI (web-interface)

Project Summary:

For our project we are going to be improving the UI of LiveDataLab so that students are better able to find, create, and complete their MPs within the class of CS410. Right now, the UI gets the job done. But we could definitely make it more user friendly for students to complete their tasks faster and with less confusion. By making the UI more clear and user friendly, we anticipate that there will be a higher completion rate of MPs and also less questions to course staff when it comes to completing MPs or finding information within LiveDataLab. On top of that, it will be more delightful for students to look at when they work on MPs. An optimized UI would definitely improve LiveDataLabs effectiveness.

Datasets, algorithms, and techniques:

We do not plan to use any algorithms or datasets for this project since it focuses on UI/UX design. We will be using some design techniques to help us build the best UI for the project. These include: wireframes, high-fidelity mockups, A/B testing, user testing, user interviews, audience analysis, user flow diagrams, user research, information architecture analysis, interaction design, graphic design, typography design, mobile design, and color theory and design. We will also be using different coding techniques to code and implement our resulting designs in React on LiveDataLab. Most will revolve around front-end techniques such as CSS layout structuring, React component design, and other techniques that we hope to discover while we are completing our high-fidelity mockups that result from our user research and wireframes.

Defining Success (improving a function):

To prove that our work is actually improving LiveDataLab, we are going to be doing a lot of initial interviews with existing and new users (students) to identify pain points and any areas of confusion they may have when using LiveDataLab. We ultimately want to make it easier to complete tasks related to MPs for students, so we will be measuring base metrics for us to improve with our project. These metrics include success rate of completing a task like cloning an MP, the average time it takes to complete that task, overall happiness with the experience (survey-based). These metrics relate to if students can successfully clone MPs and how quickly. We will measure similar things for tasks like linking a GitHub, completion of non-coding MP's (validating datasets), and checking submission details and leaderboards. Once we have all these metrics, we can compare them to the results of the same experiments we run with our improved designs. If we find they improve the experience, we will implement the designs in React.

Using LiveDataLab (utilizing the system):

LiveDataLab's front end is made in React, so we will be contributing to the GitHub repo so that we can add our changes if necessary.

Programming Languages:

HTML, React, JavaScript, SCSS/SASS/CSS

Anticipated Work (3 students):

- Initial Data Collection - 5 hours total
  - Define success metrics - 1 hour
  - Collecting initial metrics of current LiveDataLab UI - 4 hours total

        - Test on existing users - 2 hours
        - Test on new users - 2 hours
- User Research - 15 hours total
    - Conduct user interviews - 4 hours total
        - New users - 2 hours
        - Existing users - 2 hours
    - Audience analysis - 2 hours
    - Identify scenarios & use cases - 0.5 hours
    - Create user flow diagrams - 0.5 hours
    - Create wireframes - 8 hours total
        - Desktop - 4 hours
        - Mobile - 4 hours
- High-fidelity mockups - 18 hours total
    - Create high-fidelity Desktop mockups - 12 hours
    - Create high-fidelity Mobile mockups - 6 hours
- Test high-fidelity mockups - 8 hours total
    - Collecting new success metrics of proposed LiveDataLab UI - 4 hours
        - Test on new users - 2 hours
        - Test on existing students - 2 hours
    - Repeat for A/B testing to try different options - 4 hours for each different option (we will assume 2 options)
- Implement designs - 40 hours
    - Implement designs in React - 30 hours
    - Add CSS - 8 hours
    - Revise/create graphics - 2 hours

Total: 86 hours of work