Project Documentation

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 improved 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 is more delightful for students to look at when they work on MPs. An optimized UI would definitely improve LiveDataLabs effectiveness.

1. **Briefly describe any datasets, algorithms or techniques you plan to use**

We did not use any algorithms or datasets for this project since it focuses on UI/UX design. We used 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 also used different coding techniques to code and implement our resulting designs in React on LiveDataLab. Most revolved around front-end techniques such as CSS layout structuring, React component design, and other techniques that we discovered while we completed our high-fidelity mockups that resulted from our user research and wireframes.

1. **If you are adding a function, how will you demonstrate that it works as expected? If you are improving a function, how will you show your implementation actually works better?**

To prove that our work is actually improving LiveDataLab, we did 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 measured similar things for tasks like linking a GitHub, completion of non-coding MP's (validating datasets), and checking submission details and leaderboards. After getting all these metrics, we compared them to the results of the same experiments we ran with our improved designs. We found that users had better success with our UI based on our collected data.

We found that for finding a certain MP, our UI enabled students to find it about 30% faster while also increasing success rate to 100% from our measured 95% (source: user interviews with ~20 students). We also found that for managing linked accounts, our UI reduced errors of deleting linked accounts almost completely because we moved the delete button to somewhere far away from logging out (source: user interviews with ~20 students).

1. **How will your code communicate with or utilize the system? It is also fine to build your own systems, just please state your plan clearly**

LiveDataLab's front end is made in React, so the UI can easily be added to the existing LiveDataLab repository.

1. **Which programming language do you plan to use?**

HTML, React, JavaScript, SCSS/SASS/CSS

1. **Please justify that the workload of your topic is at least 20*N hours, N being the total number of students in your team. You may list the main tasks to be completed, and the estimated time cost for each task.**

Resulting 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

Evan: worked on the implementation of the UI and most of the CSS

Bhargav: worked on the react implementation and functionality

Preston: worked on the user research, interviews, and high-fidelity mockups

How to run the software

To run the software, all you need to do is pull the repository and download the code. Then in terminal, navigate to the folder "LiveDataLabUI" and then run the following commands:

1. yarn
2. yarn start

Then the website will run on localhost and you can navigate around and use the website.

How to implement the software

To implement this software with the existing LiveDataLab all that needs to be done is implement the components that are in the code we wrote. The data.js in the folder called data (LiveDataLabUI/src/**data**/) will show what data needs to be passed to the different components. The components are all modular so the UI can be easily applied or changed if necessary for LiveDataLab. All the react components are in the components folder, and pages can be added easily with the react router. The UI is also mobile responsive and the CSS is reusable so additions can be added with the existing CSS (all in global.scss).