Description of Tech Stack

Our final product will be in the form of a npm library. We will implement this with Javascript with the help of some external libraries such as terminal-kit. We will use nodejs as our run-time environment and npm as our dependency management tool. We will use jira as our project management tool.

We have wide-range of clients, anyone who wants to use a terminal text editor in their JS project will be able to use our library. The reason we choose to make our final product a npm library but not something else is to make it user-friendly. With a npm library, we can simply add our product to the npm library manager, and other people can install it with one command.

All of our team members completed their PEY, which is why we chose the javascript project. Our group consists of people with different skill sets since we all work on very different tasks during our PEY . Lantao worked with terminals most of the time so he is very familiar with text editors in terminals (vim). Wenshuo worked as a java developer so he has a lot of experience with OOP. Yujie worked as a support developer so he is very familiar with software testing. All of us have taken CSC309 and CSC301, which taught us some javascript basics and used javascript to build two applications. We know each other pretty well and we all have similar workloads/schedules, therefore, we have decided to split the work evenly among us, so that each of us will be in charge of an area that he is best with.

We have three different operating systems. Yujie is using Linux, Wenshuo is using Windows, and Lantao is using Mac. We would create a library that would function on all operating systems. Yujie has an old laptop so our code wouldn't be computational heavy. Since javascript and nodejs is supported on all three platforms and doesn't require much hardware power, all three of us will have no hardware issue using this tech stack.

For some of the design elements we gave up during our discussion, at first glance, we were planning to create our own terminal library. However, we quickly gave up this idea because of two reasons. 1. The workload to create a new terminal-related library is too heavy, and we don't have enough time to do that. 2. The emphasis on this project is on creating a terminal text editor rather than building a terminal library, and there already exists a well-built library.

We were also planning to submit javascript source code and make the user embed it in their own code. But we quickly gave up this idea because it is very inconvenient for the user and no one will be using our library if we design it like this.

Preliminary Project Development Plan

Milestone 1:

Understand the source code of termit and identify the dependencies required for this project (i.e. terminal-kit)

Build the basic structure of the application, that includes:

- Design the class structure with tools like CRC cards
- Design/Create function headers for some of the essential features like render text, read/write

team member's responsibilities:

- Understand the source code individually
- Discuss the structure as a team

Milestone 2:

Adding basic text editor features such as:

- Rendering Text
- Copy and paste
- Read/Write file

team member's responsibilities:

- Yujie
  - Testing each features, Read/Write file
- Wenshuo
  - Rendering Text
- Lantao
  - Copy and paste

Milestone 3:

Adding undo and redo features. If we have time, we will add other features such as:

- Find and replace
- Syntax highlighting
- Customized macros
- Cache (restore unsaved file for user after close)

team member's responsibilities:

- Research and implement algorithms related to redo/undo
- Identify and test edge cases

Milestone 4:

Continue working on redo, undo and additional features from milestone 3. Packaging the project into a npm library.

team member's responsibilities:

- Research on npm library packaging and publishing
- Final testing on all the implemented features to make sure bug free.