Team: Type.io

Members: Rose, Samuel, Evan

Pair Programming Report:
https://docs.google.com/document/d/14SlvkswMWqPAyCj2IuLnucqqu16R2Wtd-rIlxuhKtEY/

Editor.vue:
https://github.com/samuelnunoo/SWE-Team-Project/blob/accef6b660cde3e62ec0e5d83a04f0fd6815851b/nuxt/pages/Editor.vue

primary author: Rose

P4B.1 Pair Programming Exercise

In order to make the most of this pair programming exercise, we decided to use pair programming in working on my (Rose's) component, and paired Samuel and I together, as he has had previous experience that I haven't with the Tiptap library and the front end programming, as well as ways of testing and verifying the work that is being done. This pairing was very beneficial, as it allowed me to orient myself in the first session. We had to set up and make sure we were on the same page on what technology we were using to run and test the code, which proved to be a large task. It also allowed for me to use Samuel as a quick resource, and his previous experience saved considerable time looking up syntax that did not come naturally from previous experience with other programming languages. For this initial meeting, we mostly were thinking and planning, and getting prepared for future sessions where we worked on the code, and went through the experience together of how to extend the default editor provided by TipTap to give it the required functionality for this project. In total we spent around 3-4 hours pair programming in 4 separate meetings, yet due to the complexity of the component we were tackling, we had an experience that was slightly outside of the guidelines for this exercise and what pair programming usually entails. We ran into many problems with set up, and getting all required dependencies set up in the repository. We spent the majority of our pair programming time thinking and planning, but we think as it resulted in proper set up and planning as well as completing all necessary functionality for the Editor Component, it was a worthwhile exercise. A large portion of the time we spent programming together involved going through library documentation together and going through online resources in order to better understand how to manipulate the editor.

For this exercise, we adopted a common style of pair programming where one team member was coding and the other was actively watching and making comments and suggestions. Since a large portion of the work we did was setting up the dependencies, Samuel mostly led during these portions. Since a lot of our pair programming devolved into brainstorming and planning, we were unable to stick to this method as closely as we would have hoped, and instead of having someone coding at all times, we were both frequently attempting to find references and sharing what we had found. All of our sessions were organized by think-code sessions, where we would pick a small problem to tackle, discuss it and perform necessary research where we needed to further understand the task, then implement the lines of code that we settled on. The section we completed through the pair review was not code heavy, and the complexity came from attempting to piece together the Editor using various documentation and brainstorming, as well as attempting to set steps and narrow down what is reasonable for this component moving forward. Our final pair programming session was mostly organized into code-review, where I mostly took the reins in completing the methods for the Editor and Samuel watched, made suggestions, and reviewed them as I finished. Since my methods used requests to his Client API component, it was especially helpful to have him present while I was coding with his knowledge of the parameters and outputs of his ClientAPI methods.

As discussed in the readings, our pair programming partnering was similar to a novice-expert pairing, where I lacked prior experience with front-end programming and JavaScript, and Samuel's previous work on similar projects gave him the valuable background and familiarity with TipTap and it's documentation. We decided to choose this match up for pair programming because we wanted to be as efficient as possible considering the project time constraints, and we acknowledged ahead of time the possible risk of my lack of experience with these technologies. While both Samuel and I agreed that Samuel took a leadership role in the experience, and I got to benefit from his advice and experience (as well as quick recollection of syntax), Samuel benefitted from the refresher on how the Editor is integrated into the project, and getting a more in depth view into Rose's component and her plan is useful when planning for future integration. Brainstorming was a more even distribution of our time, where working together made searching through the documentation for what we needed much quicker, and allowed for conversations about the feasibility of different strategies.

When working alone for this project, I found myself getting stuck, running into issues that were often as simple as a missing dependency that I was not aware of due to my lack of experience with Javascript. Pair programming accelerated the process of setting up my files significantly, and I was able to pick up on and remember syntax and useful commands that sped up the process when I was working alone. Working together we also discovered that there was more to think about than we had anticipated, which accelerated the process but the extra brainstorming was necessary. We originally did not plan to take as many pair programming sessions, but we struggled to get much code down in the earlier sessions as there was so much to set up and consider. Working with a partner forced us both to question the decisions we were making, which slowed the process but ended up being valuable as it exposed where there was some previous lack of understanding or inadequate planning. I also believe that without these early pair programming sessions, my progress would have been slow, and being able to get a tutorial in a way through the novice-expert pairing ultimately accelerated the process.

As a result of pair programming, Samuel and I slowed down the process and dedicated much time to thinking and planning, producing much higher quality code. While the code we produced was limited, we did not produce any code that was not considered, and we were thorough with our approach, working on very small sections of code at a time. We were also able to standardize the technology and methods that we were using as a team to a certain extent, contributing to an overall more cohesive and higher quality repository. Working alone I ran into many issues when attempting to work with Tiptap, and many of them had easy fixes such as needing to install a package that Samuel could quickly point out. The main issues I was running into stemmed from the variety of documentation out there on the libraries we are using, and I was struggling to narrow down what did and did not apply to my component, and which ones I could make use of. A lot of the things I initially tried were slightly off due to slightly misunderstanding the documentation of a method, and pair programming with Samuel almost eliminated mistakes such as these as we discussed our ideas while we worked and reached an understanding before we proceeded. We also discovered that some specifications from earlier in the process were not as complete as we thought, and together we added functionality for deleting documents, something we initially did not decide would be part of the Editor page.

Especially considering how lost I felt initially approaching the implementation of my component, this experience was invaluable. The Editor component has proven more complicated to implement than it initially seemed, and struggling with less than straightforward documentation was a process that was made much more efficient with not only two people, but with the addition of Samuel, who had previously gone through and attempted to understand the documentation on a previous personal project. Being able to talk through ideas and what I was confused about made the experience much more pleasant, and situations where I would have been stuck had I been working alone we were able to reason through the issue together. We did experience some technological problems attempting to pair program over Zoom, and regret that we did not have the opportunity to take this class on campus where we could have worked on it in person. The most unpleasant but unavoidable part of the experience was having to question and rethink some of our initial plans and decisions, and the time constraint placed extra stress on our sessions. We initially planned to stick to the ~100 lines of code requirement, but after our first 3 meetings, we decided that despite not producing as much code, it was more important to use our time to make sure we were making good decisions and not rushing into code with inadequate plans, and focused on the set up and brainstorming aspects instead of forcing ahead into coding. Our last meeting was when we completed the majority of the code in the module, and as it was the last component left to complete in the project, it was easy to design and most of our implementation decisions were guided by what would be most compatible with the other completed components for integration.

This pair programming experience was unique for me, as I have not previously worked on a project where I have had no prior experience in the language or with the general set up and organization of a larger project with front and back end components. The process looked very different than it usually does due to time constraints and lack of foresight regarding how complicated the component would be to implement. As a team, we agree that we should have met more, and done the repository set up, gone over testing examples, and other activities that we left until later earlier in the process before we all began our individual components. If I had been more familiar with the technology, the documentation, and what it took to implement the editor, our pair programming experience could have been more focused on coding and testing. Considering the circumstances, there is not much that we could have done differently with the pair programming process, but the overall process could have been drastically different if our group foresight was better, and if I had been able to start gaining experience with the technologies earlier in the semester, instead of underestimating the risk there.

We were unfortunately unable to fully follow this assignment format, as prior to realizing we needed to document the experience with commits we began pair programming using VS Code LiveShare, a tool that allowed us to quickly and easily switch between who is editing, and eliminated the need to follow the Github commit guidelines that were outlined for this section of the project. We also traveled between branches throughout the process, resulting in an incomplete portrait of how we pair programmed through Github. I am going to provide a summary of the work we completed together in our 4 meetings, focusing on the code we completed in order to implement the editor component.

The file that holds the contents of the editor component and its implementation is Editor.vue, located in our github at the path SWE-Team-Projet/nuxt/pages/. After Samuel and I thought through using Vue to implement the Editor, provided all necessary dependencies and updated the package.JSON, and re-thought through the necessary functionality and methods in the context of the larger project, some changes were made in terms of my component and its role in the larger system, especially as we began to work on integration as a group. The DataConverter became obsolete, handled in the API, and to the Editor we added a remove method, and a save method in addition to a load document method that loaded previously created documents into the editor, but with document creation functionality moved to the ClientAPI. This also involved setting up the Editor page for vue with the HTML at the top of the code, where we added buttons for the browser page that trigger different Editor methods (save button, delete button).