## Communication

The main way we followed the XP principle of Communication is to have more communication than out scrums. Beyond our scrums on designated day to check in, we used our discord channel constantly to keep each other up to date on how changes were going or if there was anything blocking us. This was additionally important for two reasons. First, with a pretty heavy split between people working on the front and back end, it was important to communicate what backend work was blocking the frontend and either get it implemented or fixed. This allowed us to bring up blocking issues and get them fixed without having to wait for scrums. The other way it helped was because we had people learning new technologies from each other. Having a way to ask questions of somebody with more experience, and even work directly with another person are valuable ways to keep information flowing smoothly and get people productive faster.

## Simplicity

We followed the value of respect by using frameworks to simplify complex processes and by programming to the requirements and not adding feature bloat. For the frameworks, we picked frameworks that allowed us to focus on the functionality relating to user stories. Using Hibernate for example allows us to abstract away a lot of the fiddling with SQL syntax and easily turn database rows into code objects and deal with foreign keys easily. Similarly, features like Springboot annotations allow for the simplification of code at the endpoint level. Additionally, by keeping good separation of responsibilities across our project, we kept each class from becoming bloated. For programming to requirements, with some early exceptions of implementing CRUD operations that may not be needed, we kept the features added to a minimum to support the requirements. This means that some features that might be needed as the services scales were left out intentionally, theoretically to be added later at need.