Refactoring

Our team had to refactor portions of our codebase to utilise design patterns such as Singleton, Builder, and to improve code reuse by building reusable Java Swing components. This was made in order to improve the overall quality, performance, and maintainability of the codebase.

One of the first design patterns we implemented was the Singleton pattern. We used this pattern to make sure that classes like our database connection and user session class only have one instance throughout the entire application. This improved memory usage and avoided potential bugs that could arise from multiple instances of the same object.

The Builder pattern was also implemented to abstract away the creation of complex objects. Specifically, we used this pattern to create user and course objects. This allowed us to reduce the complexity of building objects and simplify the codebase overall.

Finally, we made sure to create reusable UI components in order to reduce the amount of code duplication and increase code reuse throughout our application. We broke down larger components into smaller, reusable components that could be used in other parts of the codebase. This lead to an increase in maintainability and reduction in errors due to code changes.