Documentation

The Process

We worked on this exam in a team of two. We divided tasks between us and mostly managed to not work on the same stuff at the same time. We tried to divide our task based on our individual strengths and collaborated on some of the features we found to be more complex.

We used Git for version Control all along, allowing us to work on different features which allowed us to work on different features at the same time without much conflict, although we did run into a few problems regarding the Git at some points of our task.

For C# .NETs Modern Features, we used nullable types, properties and method overloading. These features can be seen throughtout the code, for instance in the Question class.

We adhered to SOLID principles. In the Quiz class for example, it follows the Single Responsibility Principle, as it only handles quiz-related logic.

We used a layered architecture for the application, and separated into data access, business logic and presentation layers. We stored our data in a local SQL database.

We used the Singleton pattern for our database connection, and also the Repository pattern for the data access.

We used async/aait for non-blocking I/O operations, like in our quizController class, to ensure the UI remained responsive. The code is thread-safe.

We struggled to get the testing working from the get-go, so we did not implement formal testing frameworks, but conducted manual testing throughout the development process to ensure functionality.

Github: https://github.com/theGitCaptain/scaling-bassoon

Sources:

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

https://docs.microsoft.com/en-us/dotnet/csharp/

https://www.dofactory.com/net/design-patterns

https://docs.microsoft.com/en-us/dotnet/core/tutorials/using-with-xplat-cli