Assignment 6

The new CEO has decided that this needs to be the everything app. The company swiftly recruited a bunch of other developers for the new functionalities. Some have been assigned to your team, better start onboarding them.

- Create a new GitHub repository with the name: UBB-SE-2024-[your subgroup] (e.g. UBB-SE-2024-921-1). Invite the client (imre.zsigmond@ubbcluj.ro) as a collaborator so they can take a look. When adding to this project make sure to only add source files and not files generated by your ide.
- You are now one team with two team leads. Grades will be awarded for the new team. The two team leads will break down the grade separately between their old teams. If there is some fraction left that would be lost for one team (let's say a 0.125), then they may donate it to the other team. Technically one team lead is enough to present the project, but they must be able to respond to any question and fix any issue.
- Merge your solution with the other team in the room (Erasmus teams merge into one). Copy your code to the new GIT repo and merge the two structures into one. Make sure that the GUI can access all functionalities. If your team's code used mock functions to get data from the other team's application, then use the real functions now.
- I strongly suggest that you use separate branches in git when working in the bigger team, that you pay attention to task allocation, and that you start in time.
- If there are unfinished functionalities that prevent or make the merge harder, you may finish them. But do it withing reason, do not burn yourself or your team out over this. You may do this for any of the original projects, so if you decide that one of the original projects needs more attention you should allocate your efforts accordingly.
- Switch your data repository to use entity framework core. Use "code first" and migrations to create your database. The entire project will use the same database and not two different ones. You may still save/load files for anything that is not domain related (e.g. sound/video recordings) but xml store and ado.net are gone.
- Put your new repository on a server, and by server, I mean an ASP.NET core web API project that runs separately from the main application. Each team member may run the server locally, or the team may deploy it on an external machine. This does mean that application will not connect to the database directly, but though this web API.
- Use a dependency injection framework for your project, and have it do all the instantiations.
- Create sequence and/or flow diagrams for distinct, non-CRUD, non-trivial functionalities (1/student).
- For the implemented modules, if there are two versions, make sure both can be used, interchangeably through a setting.
- Code readability, testability and coupling rules apply from the previous assignment, as well as the use of StyleCop with the same ruleset. Do not make the application static. Do not have any business logic in the UI.
- You may use memory caches, just make sure they are cleared when appropriate.

Deadline: lab 6