Server

The server was implemented in .Net 6 and Entity Framework, using SQL Server to store data.

The server was implemented using good practices of programming and SOLID principles. N Layer Architecture was used on the server, specifically clean architecture. So, the server has four layers:

- MuseumApp.Core: It is the core layer of the application. Contains the entities, models and repository layer interfaces. This layer does not depend on the others.
- MuseumApp.Infrastructure: Contains the implementation of Repository Pattern, which separate logic business of data in domain layer.
- MuseumApp.Services: This layer contains business logic of the app. It has interfaces and abstract classes with the objective of make the solution as generic and extensible as possible.
- MuseumApp.API: Contains controllers, and the necessary configuration to receive the request from the client.

User Interface

The UI has implemented using Angular v14.

It is separated in Models, Services and components. Each component has independent functionality; the models are typescript classes to bind data from the view to the components. Finally, the services contain all function common to the components and the implementation of the functions to communicate with the server.

Execution

To execute the server, .net 6 and SQL Server must be installed. Then follow the next steps using the dotnet CLI:

1. In MuseumApp.API/appsettings.json change the connection string and use your own.
2. Run the command **dotnet build** in the directory of the solution .sln.
3. Open a console in MuseumApp.Infrastructure and run the command
   dotnet ef –startup-project ../MuseumApp.API/MuseumApp.API.csproj migrations add myMigration
4. Then run the command **dotnet ef database update**
5. Finally in the directory of MuseumApp.API run the command **dotnet run.**

To execute the UI (must have Angular installed):

1. Run the command **npm install**
2. Run the command **ng serve --o**