**MVC PROJECT TASK DESCRIPTION**

The Actual project **architecture** consists of a 3-Tier Application

1 – Core Tier: Contains parts of the project like Entities, and Interfaces(Interface for repositor pattern).

2- Data Tier: Contains the Data Context for the project and repository implementations.

3- Web Api Tier: Contains the ASP.NET Core Web Api for the web project.

**Principles** contained in this project:

Encapsulation is a principle used in the project to help application components and layers adjust their internal implementation without breaking their collaborators as long as external contracts are not violated

Dependency inversion implementation details can be written to depend on and implement higher-level abstractions, rather than the other way around. Which can be seen in the implementation of Repository Pattern.

**Considerations** about the project:

In the Web Tier the repository pattern was applied using the actual DataContext to deal with context. An example of the usage of the repository pattern is shown inside CoursesController.

Validations were executed in the controller but could have been executed applying a different pattern using FluentValidation which is usually my choice in larger projects and even using JavaScript, helping in the separation of concerns and applying more complex validations.

AutoMapper could have been used and is generally used for commercial applications but there were no need for this project.

All the patterns and principles were applied thinking in maintainability and components to be tested.

Database used was SqlServer created using Entity Framework Code First.

This command should be enough to create the database

Update-database

Database sample data

[

{

"name":"Sauce",

"description":"Marteli"

},

{"name":"Spaghetti","description":"Baresa"},

{

"name":"Popcorn",

"description":"microwave"

},

{"name":"Bread","description":"White bread"}]

]