Development for Chama Process

Geciane Débora Junges

Architecture and technology decision

DDD

RESTful Api

ORM

Azure

CI

Tools and technologies used

- Asp.Net Core and .Net Core
- Entity Framework Core
- Azure Service Bus
- Azure SQL Database
- CircleCi for Continuous Integration
- Built-in IoC Container
- NUnit 3
- Moq
- Automapper
- Log4net

Step by step

API

I decided to use RESTful API with .Net Core and EF Core and create the database in the Azure. I created the synchronous endpoint to sign up the student and update the course information.

Scaling

To scaling the application I chose Azure Service Bus message queue. I created the asynchronous endpoint to put the request from the user in a queue and once the request is processed the student sign up and course update will be done. The user will receive a notification by email, in this case, I'm just logging the information if success or failure.

Step by step

Querying

In the beginning, I created a SQL view to calculate in the database, but I had issues with EF Core, I'll talk a bit more about it in the problem session, and I think it wasn't the correct understanding of the requirement, so I decided after sign up the student calculate and update course information. This way, everything is saved in the database and available for consultation whenever we need it.

Problems and Challenges

Azure Service Bus

Technical debit, as I have never worked with Azure Service Bus before, I had to study and learn before implementing, so I spent some time in this activity, but in the end I was very happy to finally understand and be able to use !!!

EF Core

As I commented earlier, my first idea to solve the querying problem was to create a SQL view, but I figure out that in EF Core we can no longer run free raw SQL, we need to set a DbSet for the class or FromSql will not return anything.

Problems and Challenges

DI (Singleton)

When my message was returning from the Azure Service Bus, I was not able to persist the data, there was a "Cannot access a disposed object..." error that I figure out was the DBContext that was no longer alive. To solve this problem I had to change the ServiceLifetime for the WriteRepository to the Singleton and change the configuration of the DBContext also to the ServiceLifetime.Singleton as well.

What can be improved and how

- Integration tests
- Continuous Deployment
- New queue to calculate course information such MinimumAge, MaximumAge, AverageAge and so one.
- Create validations if there is a failure in the save sign up step or update the course. In case of failure, don't complete the message so that it can be processed again in the future.

Thank you for the opportunity to participate in this process !!!