**Relatório**

Separei o projeto em duas partes, uma Web Api em .Net Core 2.0 e uma outra aplicação em NodeJs + EJS como Engine para as Views, fazendo com que a Api rode em um container separado do Front.

Separando em duas aplicações, entramos no conceito de micro serviços as vantagens são: Facilidade em manutenção e escalabilidade, ambém utilizei o Docker, tornando possível executar a Api em containers.

**Web Api .Net Core:** O projeto foi feito com a arquitetura DDD (Domain-Driven Design) que é atualmente muito utilizada e nos últimos meses comecei a usa-la em aplicações do meu atual emprego, escolhi ela porque essa arquitetura é voltada para o domínio de negócios, ou seja o design é focado nas regras de negócio do software (Domain).

O DDD deixa seu projeto organizado, facilitando com que caso acontece de o negócio se expandir, seja fácil realizar uma manutenção ou acréscimos de regras no domínio e também facilitando o desenvolvimento delas.

Em minha opinião o DDD é uma arquitetura robusta que permite facilmente uma manutenção no futuro, onde no Domain temos nossas regras e interfaces e depois criamos pequenos (micros) projetos (Services, Infrastructures) que implementam o que está no Domain.

**App em NodeJs:** Criei uma aplicação em NodeJs por dois fatores, na data 25/04/2018 a Bruna Quinteros perguntou se eu tinha interessa em uma vaga de Nodejs e também conforme mencionado na entrevista pessoal, não tenho muita experiência e conhecimento no desenvolvimento Front-End, então optei pelo Nodejs + EJS (pois já fiz alguns sistemas com essas tecnologias ) e para facilitar no desenvolvimento das Views com HTML, CSS e JQuery por causa do EJS e o Node consumindo e enviando dados para a Api.

O App em NodeJs foi criado seguindo a arquitetura MVC (Model-View-Controller), onde utilizei da Engine EJS(Effective JavaScript templating) que facilita o transporte de dados do Back-End (Node) para o Front (Views), basicamente conseguimos usar Javascript no HTML.