

Padrão CQRS

Padrão CQRS

- Sigla de Command and Query Responsibility Segregation
- Padrão arquitetural que tem como proposta separar as responsabilidades de consulta (Query) e comandos (Command)
 - Consultas visam buscar dados, não atualizando o estado do sistema
 - Por outro lado, comandos atualizam o estado do sistema
- É possível utilizar tanto um banco de dados para ambas as operações trabalharem, ou um para cada, resultando em dois modelos de dados separados

Padrão CQRS

- Por exemplo, podemos utilizar o SQL Server como banco de leitura, e o MongoDB como banco de escrita
- A atualização do banco de dados de leitura pode ser feita via eventos capturados tanto em memória quanto através de mensageria/event streaming com ferramentas como RabbitMQ ou Apache Kafka, por exemplo
- Mesmo com apenas um banco de dados, este padrão arquitetural contribui para separação de responsabilidades e um código mais limpo



Até a próxima aula!

Sobre o MediatR

Sobre o MediatR

- **MediatR** é uma biblioteca popular em .NET que implementa o padrão de design Mediator, facilitando o desacoplamento na comunicação entre objetos.
- Ela permite que objetos se comuniquem indiretamente por meio de mediadores para lidar com solicitações, comandos, mensagens, ou eventos
- Ou seja, ao invés do Controller conhecer para qual Handler ele vai encaminhar os dados, o objeto mediador é quem vai decidir para qual será encaminhado
- Biblioteca unificada **MediatR**

Sobre o MediatR

- De forma bem resumida, as principais interfaces a se conhecer são:
 - **IRequest:** a classe que representa seus modelos de entrada ou consulta deve implementar essa interface (não tem métodos definidos)
 - Nos Commands, geralmente será o modelo de entrada da aplicação
 - Nas consultas, geralmente será construída com base nos parâmetros de URL e Query
 - **IRequestHandler:** esta classe será a que vai processar a classe que implementa o IRequest
 - É uma boa prática implementar um por classe, para evitar classes muito longas e com múltiplas dependências
 - **IMediator:** utilizada para publicar um comando ou consulta para seu correspondente Handler
- **Como configurar o MediatR**
 - `services.AddMediatR(config => config.RegisterServicesFromAssemblyContaining<T>());`



Até a próxima aula!