CQRS

Command query responsibility segregation

Command

به دستوراتی در سیستم می گویند که قرار است یک state یا وضعیتی را تغییر دهد. به طور مثال entity را add یا update یا delete می کند یعنی یک وضعیتی را تغییر می دهد

مثال : ویرایش کاربر یا اضافه کردن یا ثبت نام یا حذف کاربر

Query

درخواست های که فقط یک Data را واکشی می کند می گویند و هیچ state را تغییر نمی دهد و فقط واکشی داده دارد.

مثال : دریافت اطلاعات یک کاربر یا یک محصول

CQRS

باید command ها و query ها از یک دیگر جدا باشد و هر کدام باید مدل مختص به خود را داشته باشند هر command یا query باید model خود را داشته باشد مثل creteUserCommand و GetUserByIdQuery و برای کاهش پیچیدگی کار از الگوی mediator استفاده می کنیم که کتابخانه مختص به خود را دارد به نام MediatR در سی شارپ و پیاده سازی cqrs یا command query segregation را برای ما راحت تر می کند و از پیچیدگی cqrs را برای ما کم می کند و می تواند در سطح های مختلف این الگو استفاده شود

Code یا معماری برای جدا کردن command , query ها که سبب رعایت اصل single responsibility نیز می شود که سبب این می شود که هر فایل یک دلیل برای تغییر داشته باشد

سطح دیگر برای پیاده سازی DataBase است.

یک database برای command ها و یکی برای Query ها وقتی این کار انجام می شود سبب افزایش عملکرد db شده و درخواست ها بین این 2 share می شود و واکشی ها در یک db و c-w-d نیز در یک db جداگانه اتفاق می افتد و می توان از یک no-sql برای query استفاده کرد برای افزایش سرعت و مدل های ذخیره سازی query و command را هم جدا کرده

اعمال در سطح db می تواند در برخی سیستم ها مشکل باشد و در برخی نه

به طور مثال زمانی که یک command ارسال می شود یک Data تغییر می کند زمانی را میبرد که db مربوط به query ما update شود بین میلی ثانیه تا ساعت بسته به پیاده سازی ما و پروژه در حالتی هست که می خواهیم در لحظه تغییرات را ببینیم و دیتا ها با یک دیگر sync نیست و سبب eventual consistency می شود و سیستم ما حتی در حالت چند ثانیه نیز مشکل دارد اما این مشکل نباشد می توان پیاده سازی کرد این روش را

**روش دیگر**

پیاده سازی در هر دو سطح code معماری و data base که سبب سرعت بهتر و ساختار معماری بهتر می شود.

**مزایا:**

ورودی و خروجی سیستم ما در یک نقطه متمرکز است و می تواند سبب کش کردن راحت تر است.

اگر تسلط روی الگو یا سیستم خیلی تسلط وجود نداشته باشد ممکن است سبب سخت کردن کار شود برای پروژه های ساده نیاز است اما در پروژه های پیچیده و توسعه پذیر بررسی کنید که مناسب است یا نه و با توجه به بودجه زمان و توانایی ها اقدام کنید.

