Grzegorz Wasylow

Projektowanie nowoczesnych aplikacji internetowych z wykorzystaniem wzorca Mediator oraz CQRS (Command Query Responsibility Segregation) w środowisku .NET i Azure

Expert Summit 2022

All about Microsoft® technologies ONLINE

Sala B

Godz. 13:00

lingaro

Future Processing

pwc

C.H. ROBINSON

Chmurowisko

ANEGIS

summ-it

integrity partners

Demant

emagine

REPLY CLUSTER

spark-IT

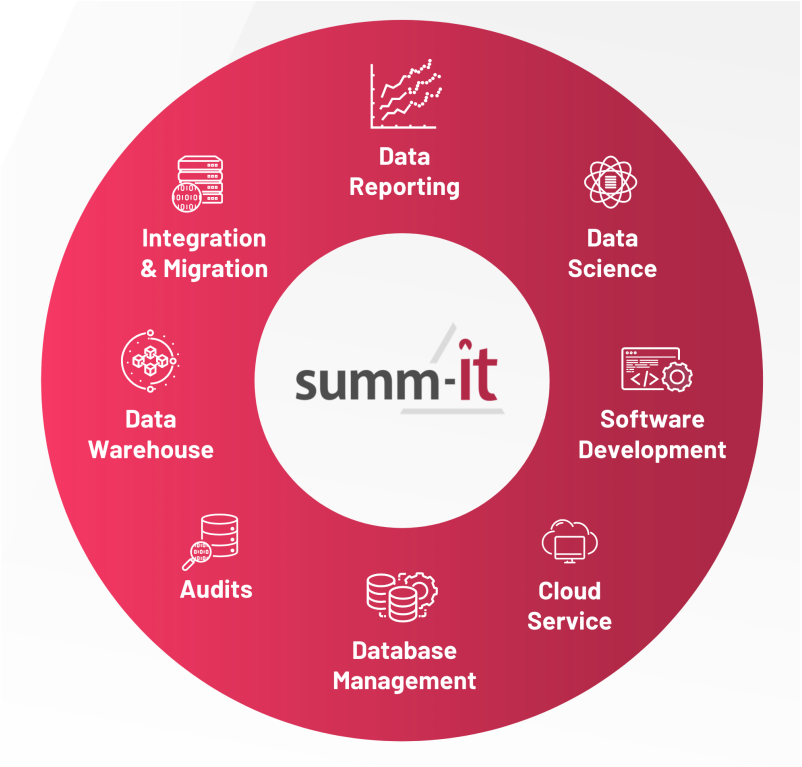
Praktyczne projektowanie aplikacji internetowych z wykorzystaniem CQRS oraz wzorca mediator (MediatR) w Azure.

summ-it

- **Bio** & **summ-it**,
- **CQRS** historia - spojrzenie praktyczne,
- Wzorzec **Mediator**,
- **MediatR** - spojrzenie praktyczne,
- API w REST czy Monolicie? Plusy dodatnie i ujemne,
- **Event Sourcing**? Spojrzenie holistyczne na architekturę,
- GitHub .NET Core & **Azure sample** (CI/CD GitHub Actions).

https://github.com/gwasylow/expert-summit-cqrs-mediatr

Bio Grzegorz Wasylów



Head of Software Development & Architecture Lead
w summ-it s. a.

- Microsoft Certified Solutions Developer - .Net Web Applications

- W środowisku .NET Frameworka od wersji 1.1
- 14 lat komercyjnego doświadczenia w IT
- Wykładowca – Wyższa Szkoła Bankowa

- Microsoft Advanced Specialization in Web Apps Modernization

https://summ-it.pl/

https://www.linkedin.com/in/grzegorzwasylow/

Twoje dane w dobrych rękach

summ-it

Analityka biznesowa

Audyty baz danych

Bazy danych w chmurze

Wsparcie systemów baz danych

Superb DBA

Software Development

RekrutujeMY

Dołącz do summ-it team

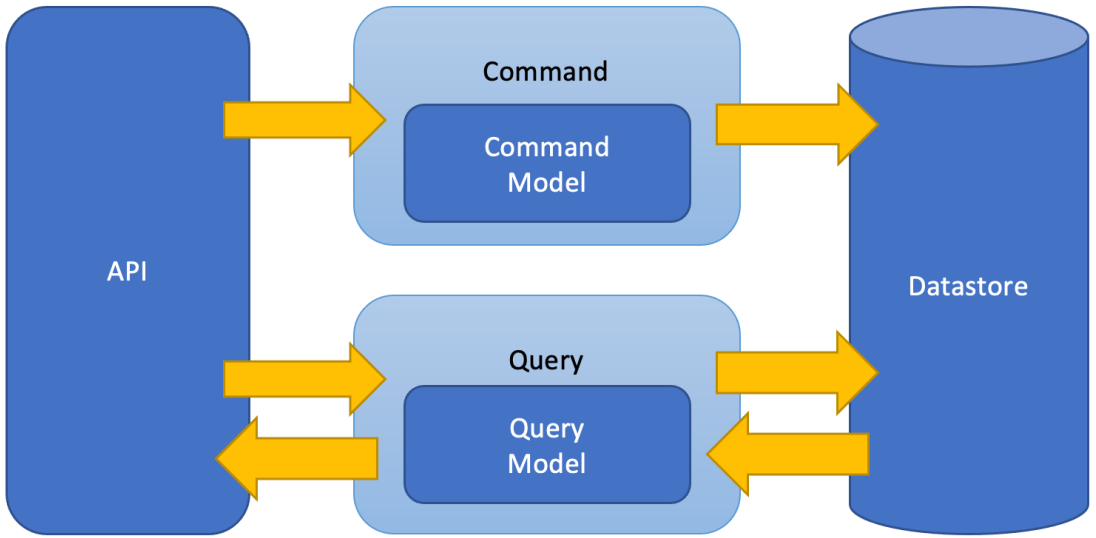
summ-it.pl

summ-it.pl/kariera/

CQRS Command and Query Responsibility Segregation

summ-it

CQRS to wzorzec, który oddziela operacje odczytu i aktualizacji dla danych poprzez projektowanie **Komend** oraz **Zapytań**. Wdrożenie CQRS w Twojej aplikacji może zmaksymalizować jej wydajność, skalowalność i bezpieczeństwo – przede wszystkim **ujednolici** sposób w jaki zespół projektuje i wdraża oprogramowanie, ponieważ wymusimy **konwencję pisania kodu,** łatwiejszy **rozwój i praca zespołowa.**



Elastyczność uzyskana dzięki migracji do CQRS pozwala systemowi na lepszą ewolucję w czasie i zapobiega powodowaniu konfliktów wynikających z cyklu życia oprogramowania.

Wada: zwiększamy poziom skomplikowania kodu poprzez dodatkowe warstwy.

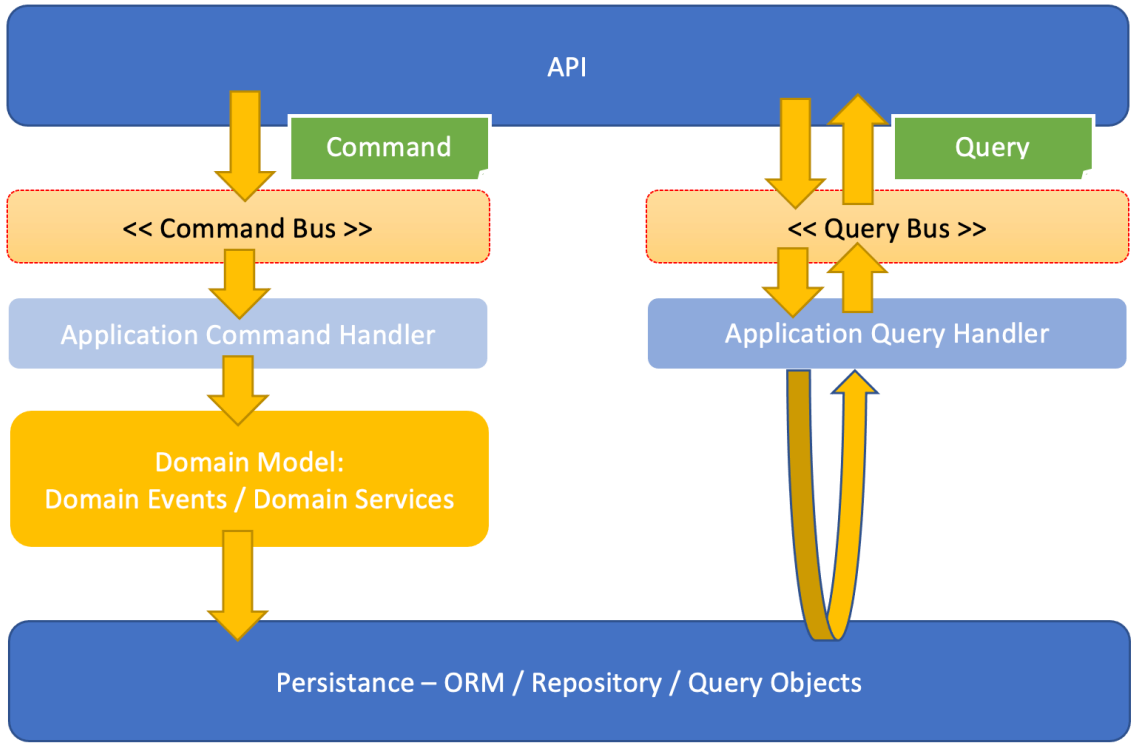
CQRS Command and Query Responsibility Segregation

summ-it

Zakładamy fakt tego, że konstrukcja naszego systemu budowana jest jednolicie:

Komenda:
- Może pobierać dane wejściowe,
- Modyfikuje dane w aplikacji,
- Nie zwraca danych*.

Zapytanie:
- Może pobierać dane wejściowe,
- Nie modyfikuje danych w aplikacji,
- Zwraca zestaw danych.



* https://stackoverflow.com/questions/43433318/cqrs-command-return-values

Mediator jako wzorzec projektowy

summ-it

Behawioralny (dotyczy zależności pomiędzy obiektami) wzorzec projektowy, który pozwala zredukować niejednoznaczne zależności między obiektami.

Wzorzec ogranicza bezpośrednią komunikację między obiektami i **zmusza je do współpracy** tylko za pośrednictwem obiektu **mediatora**.

- Definiuje uproszczoną komunikację między klasami.
- Konkretyzuje obiekt, który zawiera sposób interakcji zestawu obiektów.
- Mediator promuje luźne sprzężenie, uniemożliwiając wyraźne odwoływanie się obiektów do siebie i umożliwia niezależne różnicowanie ich interakcji.

- Przykładowe zastosowanie: uniwersalna magistrala komunikacyjna, Air Traffic Control.

MediatR – spojrzenie praktyczne

summ-it

Relatywnie prosta w nauce i obsłudze biblioteka środowiska Microsoft .NET, która pre-implementuje wzorzec na potrzeby mediacji obiektów.

Najważniejsze zalety:

- **Decoupling**: oddzielenie kodu aplikacji od kodu najwyższego poziomu Frameworka, sprowadzamy aplikację do **części wspólnych**,
  - W ASP.NET Core ostatecznie konwertujemy żądanie HTTP na **żądanie aplikacji**. To żądanie jest zaś całkowicie oddzielone od konkretnej struktury najwyższego poziomu i może być wywoływane z dowolnego miejsca.
  - Wspomniane żądania aplikacji obsługiwane są: **Handlerami,** za pomocą **Pipelines, Notyfikacjami i Walidatorami** – charakterystyczna metoda biblioteki służąca do mediacji pomiędzy warstwami to **Send();**

- **Szybka implementacja CQRS** dzięki wykorzystaniu reużywalnych części kodu, niezależnie od przeznaczenia aplikacji (np. Webowa czy WebAPI itd.)

MediatR

https://github.com/jbogard/MediatR

Najważniejsza wada:

- MediatR wykonuje swoje akcje w modelu **in-proc**. Oznacza to, że gdziekolwiek proces wywołuje metodę **Send();** jest również tym samym procesem, który wykonuje odpowiednią procedurę obsługi dla tego żądania, czyli jego handler.

MediatR - spojrzenie praktyczne

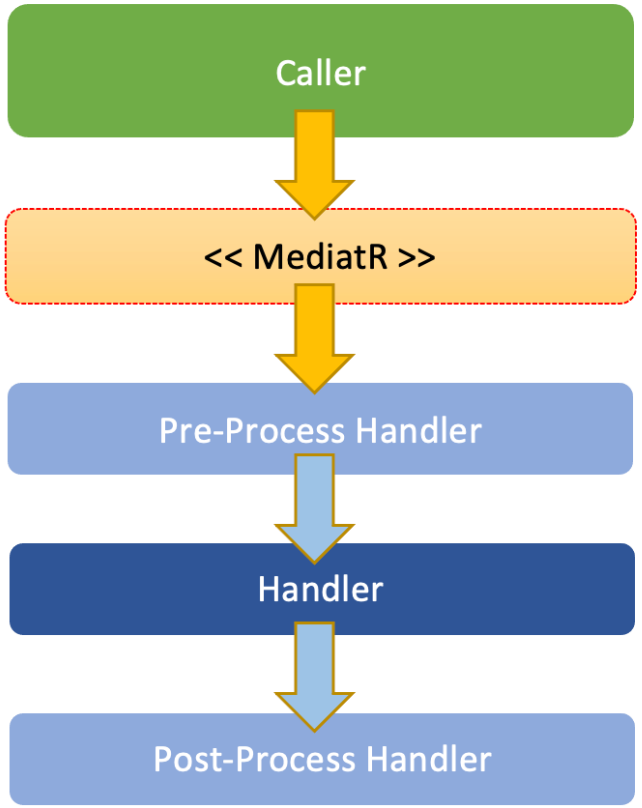
summ-it

MediatR **ułatwia implementację** CQRS poprzez wbudowane mechanizmy:

Mechanizmy **wspólne**:
- Mediujemy w warstwach niższych za pomocą metody **Send();**
- **Behaviours** (odpowiednik .NET Middleware) dziedziczący po **IPipeline<**TRequest, TResponse**>**
- **Notyfikacje** dziedziczące po **INotification**<OurNotification>

Komendy:
- Jednolita budowa
- Dziedziczą po **IRequest<**Command**>**
- Może posiadać **Walidator** dziedziczący z **IValidation<**Command**>**
- **Handler** dziedziczy po **IRequest**<Command, ReturnedType**>**
  - Handler posiada asynchroniczną metodę **Handle();**
  - Co do zasady Komenda nic nie zwraca*

Zapytania:
- Jednolita budowa
- Dziedziczą po **IRequest<**Query**>**
- **Handler** dziedziczy po **IRequest**<Query, ReturnedType**>**
  - Handler posiada asynchroniczną metodę **Handle();**
  - Zapytanie zwraca wynik z naszej aplikacji



API w REST oraz Monolit, co zrobić?

summ-it

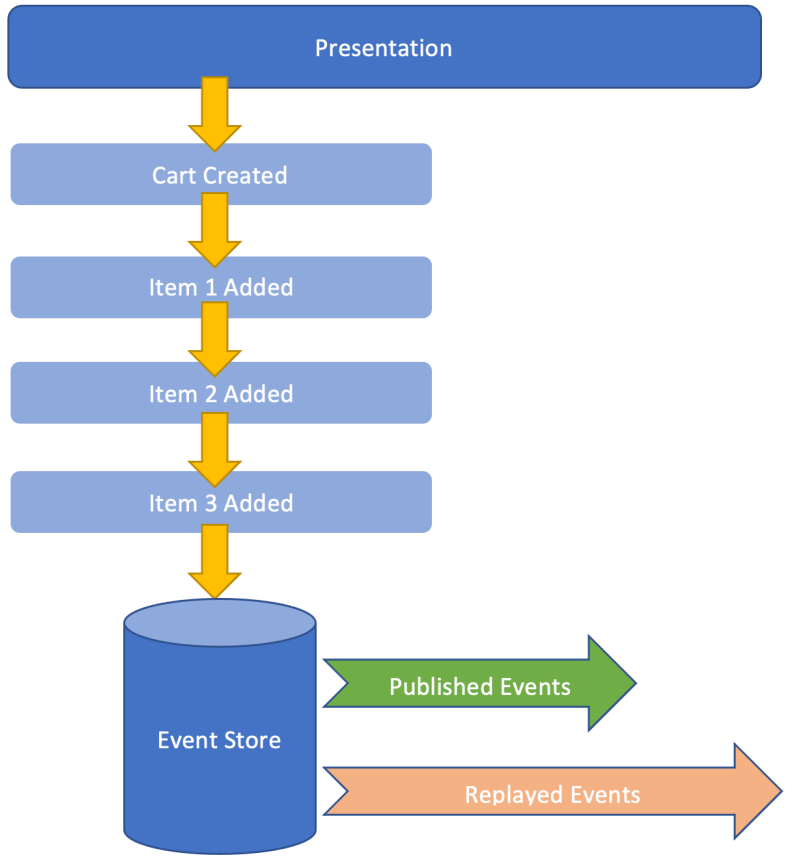
Architektura - API Mikroserwisów	Architektura - Monolit
• Przemyśl **API Gateway**, aby oddzielić usługi (wersję mobilną, stronę Klienta oraz inne usługi), • Podejmując wyzwanie przenoszenia funkcjonalności z Monolitu można pozostawić istniejące rozwiązanie vs nową implementację, następnie przekierować ruch w Gateway stopniowo, kiedy osiągniemy w pełni przetestowane rozwiązanie, • Ułatwione **testowanie** (wydzielone funkcjonalności vs całe moduły), • Łatwiejszy **load balancing** usług, • Naczelna zasada: **Dziel i rządź.**	• Rozwijany od dawna – najczęściej wiele lat, • Wraz z rozwojem i cyklem życia programu, staje się **mniej kontrolowany** oraz **trudniejszy w utrzymaniu,** • Na przestrzeni czasu zmieniają się technologie – oprogramowania zazwyczaj się nie przepisuje, dodanie nowej funkcjonalności wymaga dopasowania się do istniejącej konwencji i bywa mocno utrudnione – narasta **tzw. „dług technologiczny".** **Co zrobić?** • Nie podejmujemy drastycznych kroków, przenosimy lub dobudowujemy nowe rozwiązania jako Mikroserwisy, • Przenosimy funkcjonalności krok po kroku.

CQRS – Event Sourcing

summ-it

Jest to mechanizm wspierający CQRS. Zamiast przechowywać aktualny stan aplikacji (danych w domenie) tworzymy **magazyn zdarzeń.** Rejestrujemy w nim jawnie całą serię akcji wykonanych w ramach domeny.

- Może uprościć zadania w złożonych modelach domenach, **unikając konieczności synchronizowania** modelu danych i domeny biznesowej, jednocześnie poprawiając wydajność, skalowalność i responsywność.
- Może również zapewnić spójność danych transakcyjnych oraz zachować **pełne ślady audytu i historię**, które mogą umożliwić działania kompensujące w przypadku ich wykorzystania.
- Przy rozdzieleniu Komend i Zapytań implementacja ES jest **relatywnie prosta** i z góry podzielona na odpowiednie zadania.

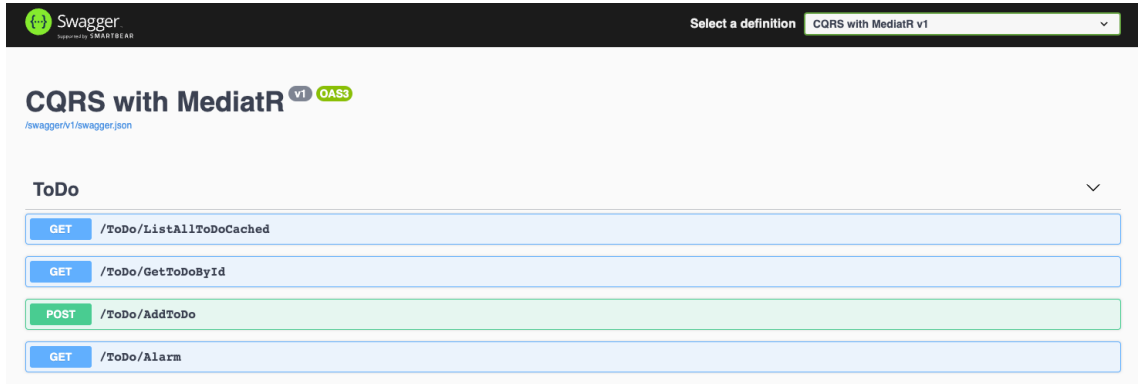


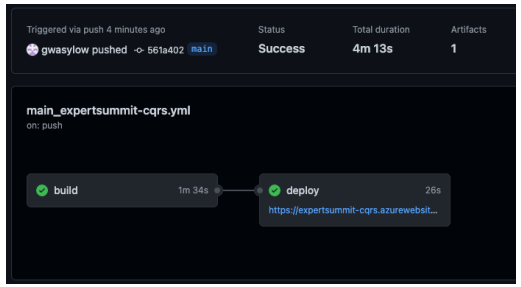
CQRS i MediatR przykładowa Implementacja w Azure

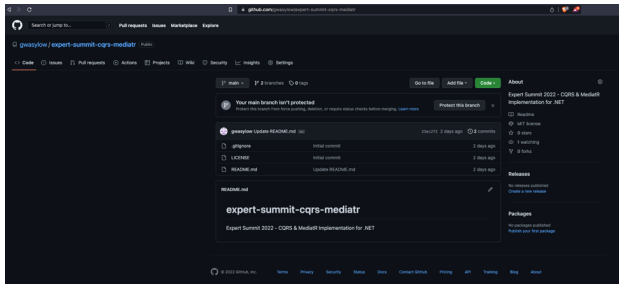
summ-it

Repozytorium kodu GitHub:

- Opis aplikacji:
  - **WebApi** serwujące dane dostępne w **Azure** (ToDoList)
- Konfiguracja:
  - **Swagger** UI
  - MediatR config
  - Mechanizm CI/CD poprzez **GitHub Actions**
- **MediatR**:
  - Wykorzystanie w kontrolerze
  - Behaviour (np. mechanizm Cache / Logowanie zdarzeń)
  - Notyfikacja (np. mechanizm alarmowania)
  - Walidacja
  - Wspólna baza generyczna pod odpowiedź Http
- **CQRS** na podbudowie MediatR:
  - Implementacja Komend oraz walidacji
  - Implementacja Zapytań





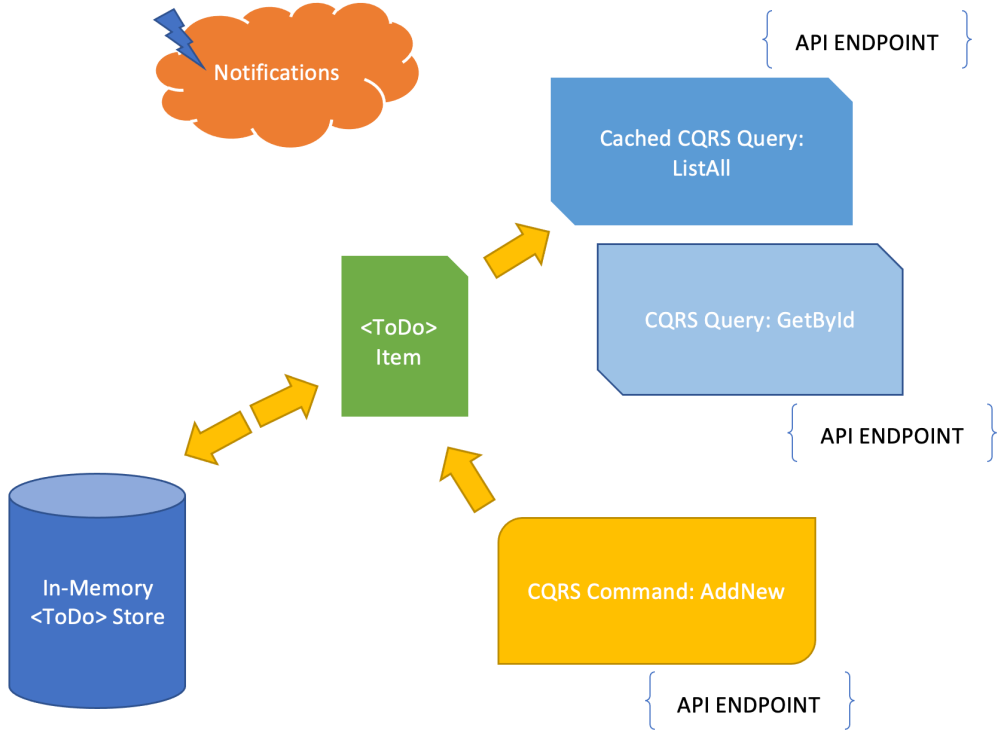


https://github.com/gwasylow/expert-summit-cqrs-mediatr

https://expertsummit-cqrs.azurewebsites.net/swagger/index.html

CQRS i MediatR przykładowa Implementacja w Azure

summ-it



https://github.com/gwasylow/expert-summit-cqrs-mediatr

https://expertsummit-cqrs.azurewebsites.net/swagger/index.html

Q&A porozmawiajmy

summ-it

Zapraszam do zadawania pytań.

RekrutujeMY