ONTWIKKELTEST – SCOPR

Analyse Documentatie

Contents

Inleiding 2

Toelichting 3

- Projectstructuur 3
- Data Flow 4
- MongoDB Schema 4
- Implementatie van CQRS 5

Inleiding

Dit project richt zich op het creëren van een robuuste applicatie die gegevens van verschillende externe webservices verzamelt, opslaat en presenteert aan de eindgebruiker. Het doel is om een gebruiksvriendelijke interface te bieden die niet alleen de benodigde informatie over landen en hun wisselkoersen toegankelijk maakt, maar ook de mogelijkheid biedt om deze gegevens te rapporteren in een overzichtelijk PDF-formaat.

De applicatie maakt gebruik van een moderne architectuur, gebaseerd op de principes van Onion Layer Architecture en CQRS (Command Query Responsibility Segregation). Deze aanpak zorgt voor een duidelijke scheiding van verantwoordelijkheden, waardoor de applicatie schaalbaar en onderhoudbaar is. Door gebruik te maken van een MongoDB-database kunnen historische gegevens efficiënt worden opgeslagen en opgehaald, wat essentieel is voor het genereren van rapportages over de gemiddelde eigenschappen van geselecteerde landen.

In dit document wordt een analyse gepresenteerd van de architectuur, de gebruikte technologieën, de datastromen en de interacties tussen de verschillende componenten van het systeem.

Toelichting

Projectstructuur

De oplossing is opgedeeld in verschillende projecten volgens de "Clean Architecture" principes:

1. **SCOPR.API**: ASP.NET Core Web API project
    - Controllers die de API-endpoints bevatten
    - Swagger integratie voor API documentatie
    - Dependency Injection configuratie
2. **SCOPR.Application**: De applicatielaag
    - Commands, Queries en bijbehorende Handlers
    - Service interfaces
    - Data Transfer Objects (DTOs)
3. **SCOPR.Domain**: De domeinlaag
    - Domain entities
    - Value objects
    - Enums en constanten
4. **SCOPR.Infrastructure**: De infrastructuurlaag
    - Repositories voor MongoDB
    - API clients voor externe services (RestCountries en Fixer)
    - PDF generator met QuestPDF
5. **SCOPR.UI** (Optioneel): Een aparte frontend applicatie
    - ASP.NET Core MVC of Blazor WebAssembly

Data Flow

1. **Ophalen van landgegevens**:
    - Een scheduled job of API endpoint triggert een `FetchCountriesCommand`
    - De command handler gebruikt `ICountryApiClient` om data op te halen van RestCountries
    - De opgehaalde data wordt opgeslagen via `ICountryRepository` in MongoDB
2. **Ophalen van wisselkoersen**:
    - Een scheduled job of API endpoint triggert een `FetchExchangeRatesCommand`
    - De command handler gebruikt `IExchangeRateApiClient` om data op te halen van Fixer.io
    - De opgehaalde data wordt opgeslagen via `IExchangeRateRepository` in MongoDB
3. **Genereren van rapporten**:
    - Een API endpoint triggert een `GenerateReportCommand`
    - De command handler haalt gegevens op via repositories
    - `IReportGenerator` genereert een PDF-rapport met QuestPDF
    - De controller stuurt het PDF-bestand terug naar de client

MongoDB Schema

De MongoDB database bevat twee collecties:

1. **Countries**: Bevat landgegevens met geneste currency informatie
2. **ExchangeRates**: Bevat historische wisselkoersen

Implementatie van CQRS

De CQRS implementatie maakt gebruik van MediatR:

- **Commands**: Wijzigen de staat van het systeem (bijv. ophalen en opslaan van data)
- **Queries**: Lezen data zonder de staat te wijzigen (bijv. ophalen van samenvatting)