NestJs Rest API & Graphql uitwerking

Naam	Wessel Vrolijks
Datum	25 aug 2021
Opleiding	Informatica
Hogeschool	Avans
Locatie	Breda

Doel
Het doel van deze course was om een backend te bouwen met de technieken in de onderstaande tabel. Daarnaast laat het project ook het verschil zien tussen een REST en een GraphQL API. Een evaluatie tussen GraphQL, REST is hier terug te vinden.

NestJS
NestJS is een JavaScript framework dat opgebouwd is met behulp TypeScript. Omdat het native TypeScript ondersteund, biedt het framework ondersteuning voor static typing. Hierdoor is NestJS geschikt om te gebruiken als backend voor grootschalige applicaties.

Dependency management
NestJS is opgebouwd middels een aantal basisprincipes die vergelijkbaar zijn met Angular. Zo worden dependencies opgesteld middels modules. Binnen modules zijn vervolgens classes beschikbaar die publiek gesteld moeten worden met behulp van providers. Door deze opzet wordt je als developer geforceerd om je code modulair op te bouwen. Hierdoor is NestJS een framework waarmee een microservice architectuur makkelijker gerealiseerd kan worden dan bijvoorbeeld een framework als Symfony.

Het nadeel van dit systeem is alleen dat wordt geadviseerd om businesslogica middels modules van elkaar te scheiden. Dit kan op lange termijn problemen opleveren op het gebied van dependency management omdat elk stukje businesslogica in een aparte module is gezet. Symfony had in de eerdere jaren een soortgelijk systeem genaamd Symfony Bundles. Hoewel het principe hierachter is om code modulair op te bouwen, blijkt het meestal lastig te implementeren in de praktijk omdat het resulteert in een fragmentatie van businesslogica. Dit heeft weer op lange termijn een negatieve impact op de onderhoudbaarheid van een systeem indien dit op een onjuiste wijze wordt geïmplementeerd. Om die reden is Symfony vanaf versie 4.0 afgestapt van deze implementatievorm.

GraphQL vs REST
GraphQL is een taal waarmee de datastructuur aangegeven kan worden door de client. Hierdoor is GraphQL een stuk flexibeler in vergelijking tot een REST api. Omdat de structuur van tevoren aangegeven kan worden, is er ook een reductie in de hoeveelheid requests die nodig is om queries de benodigde data op te halen. Zo kan alle data die een client nodig heeft in één request uitgevoerd worden. Een voorbeeld is terug te zien in de afbeelding hiernaast. Hier zijn de REST requests terug te zien die nodig zijn om gebruikers, posts en volgers op te halen. Binnen een REST api is het gebruikelijk om per datatype een aparte endpoint aan te maken. Hierdoor zijn er 3 requests nodig om de benodigde data op te kunnen halen. De server bepaald dus hoe data in dit geval opgehaald moet worden.

In het geval van een GraphQL api bepaald de cliënt het datamodel. De server bepaald dan middels een resolver hoe data opgehaald moet worden in de database. Hierdoor hoeft een client maar een request te doen om alle benodigde data op te kunnen halen.