Component/Module

May 16, 2016

Onze applicatie heeft een gelaagde architectuur. Deze lagen hangen niet van elkaar af en kunnen onafhankelijk aangepast worden. Hieronder vindt u een schema waarin u kunt aflezen hoe deze modules met elkaar communiceren.

Elke module voldoet aan een Interface. Deze interface zorgt ervoor dat de eigenlijke implementatie van de module afgeschermd blijft voor de andere modules.

Hieronder worden de verschillende modules en de modellen afzonderlijk besproken.

1 Modellen

1.1 Database Modellen - Postgres

Deze modellen zijn een exacte weerspiegeling van de tabellen in de Database. Dit laat toe makkelijk de data in de database toe te voegen, te manipuleren en verwijderen. De modellen voldoen allemaal aan eenzelfde interface. Deze interface laat het toe makkelijk queries op te stellen voor de klasse.

1.2 Database Modellen – Mongo

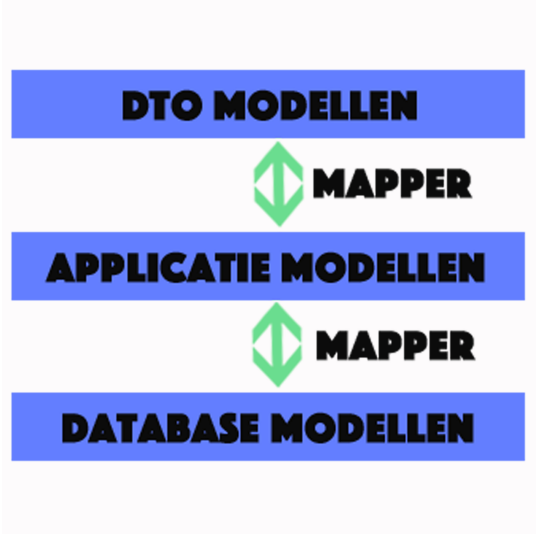
Er wordt voor het opslaan van de evenementen gebruik gemaakt van MongoDB. Met oog op uitbereidingen is er een interface gemaakt voor klassen die worden opgeslagen in de MongoDB. Alles klassen moeten deze implementeren om een makkelijke manipulatie van de gegevens in de database toe te laten.

1.3 Applicatie Modellen

De applicatie modellen bevatten de logische representatie van de klassen. Bij het aanmaken en manipuleren van deze modellen worden er checks uitgevoerd op de geldigheid van de verschillende in te vullen velden. Indien er een ongeldigheid optreed zullen deze modellen excepties opgooien.

1.4 DTO Modellen

De DTO modellen representeren de gegevens die doorgegeven worden aan de API. Deze modellen bevatten de nodige attributen en de nodige getters en setters. Elk attribuut van de klasse bevat een Jsonpath annotatie om een goede omzet te garanderen naar het json formaat. Er worden geen controles uitgevoerd op de geldigheid van de velden in de klassen.



2 Mappers

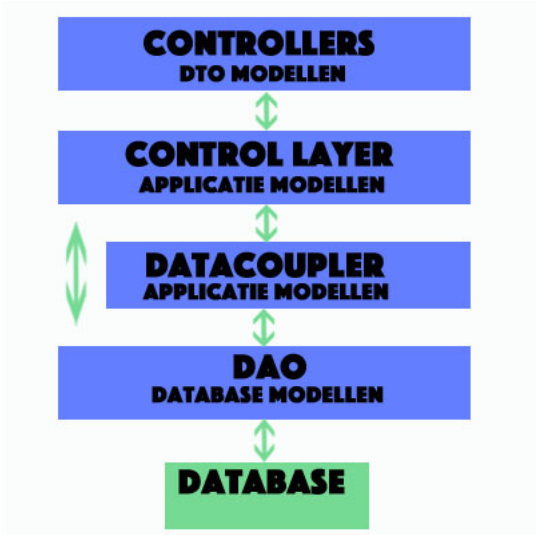
2.1 DTO Mappers

Deze mappers laten toe de Applicatie modellen om te zetten in een DTO model en omgekeerd.

2.2 Database Mappers

De database mappers laten toe de applicatie modellen om te zetten in database modellen en omgekeerd. Daarboven op zorgen sommige mappers ervoor dat alle gegevens correct ingevuld worden vanuit de database of de database cache. Vb: Indien met een Database User wil omzetten in een applicatie User zullen alle velden van de User ingevuld worden. Ook alle locaties en routes zullen hier volledig worden aangevuld.

Layers



3 Control Layer

3.1 DAC – Data Access Context

De DAC is een abstractie van een open connectie naar een databank. Alle communicatie moet via deze klasse verlopen. Het biedt de Data Access Objects aan die nodig zijn voor de communicatie met de databanken. De DAC laat ook toe de connectie met de database af te sluiten als deze ten einde komt.

3.2 DAP – Data Access Provider

Deze klasse is een abstractie van de database component in de applicatie. Deze klasse bied de DAC aan die verdere connectie toelaat met de database. Ook biedt deze klasse de datasource aan die gegevens ophaalt uit externe bronnen.

3.3 Database

Deze component in de module omvat alle manipulaties mogelijk met de databanken en data in de applicatie. Deze klasse gebruikt de onderliggende lagen voor het correct uitvoeren van de aangeboden methodes. Op deze manier beschermd deze klasse de onderliggende lagen op foutieve operaties.

3.4 Database Cache

Deze component voorziet een cache module voor de klasse Database. Recent gebruikte objecten worden hierin bijgehouden voor een snellere datatoegang toe

te laten.

4 DAO Layer

4.1 DAO - Data Access Objects

De data access objecten laten toe de data in databank te manipuleren.

4.2 CRUDDAO

Er is één gemeenschappelijke klasse voor de toegang naar de Postgres Database. Doordat alle Database Modellen dezelfde interface implementeren kunnen er, via dezelfde methodes, voor verschillende klassen gegevens gemanipuleerd worden. Dit zorgt ervoor dat we niet zo zeer een DAO nodig hebben voor elke klasse, maar wel voor elke database die we gebruiken.

4.3 EventDAO

Deze klasse zorgt voor de communicatie met de MongoDB. Alle events worden in de database opgeslagen, gemanipuleerd en verwijderd via deze klasse.

5 Data Source

Deze module staat in voor het ophalen van gegevens uit een externe source. Het zorgt voor uniformiteit onder de externe bronnen in het ophalen van gegevens.

5.1 Waze Data Source

De enige externe bron van informatie die momenteel gebruikt wordt binnen de applicatie is Waze. Deze klasse implementeert uiteraard de interface Data Source.

5.2 Data Fetcher

De data fetcher zorgt ervoor dat alle data uit externe bronnen correct wordt ingeladen in de applicatie. Deze klasse bereidt de klasse thread uit en gaat om bepaald aantal minuten nieuwe data ophalen van Waze. Aan de hand van enkele hulpklassen wordt de data van Waze omgezet in evenementen en vervolgens toegevoegd aan de databank.

6 Data Coupler

De Data Coupler koppelt de evenementen aan een Locatie of een Route.