Imre Boersma

is.boersma@student.han.nl

Opleverdocument

Spotitube

Inhoud

[1. Inleiding 2](#_Toc85203666)

[2. Package diagram 3](#_Toc85203667)

[1.1. Alternatieven 3](#_Toc85203668)

[3. Deployment diagram 4](#_Toc85203669)

[3.1. Alternatieven 4](#_Toc85203670)

[4. Ontwerpkeuzes 5](#_Toc85203671)

[5. Conclusie 6](#_Toc85203672)

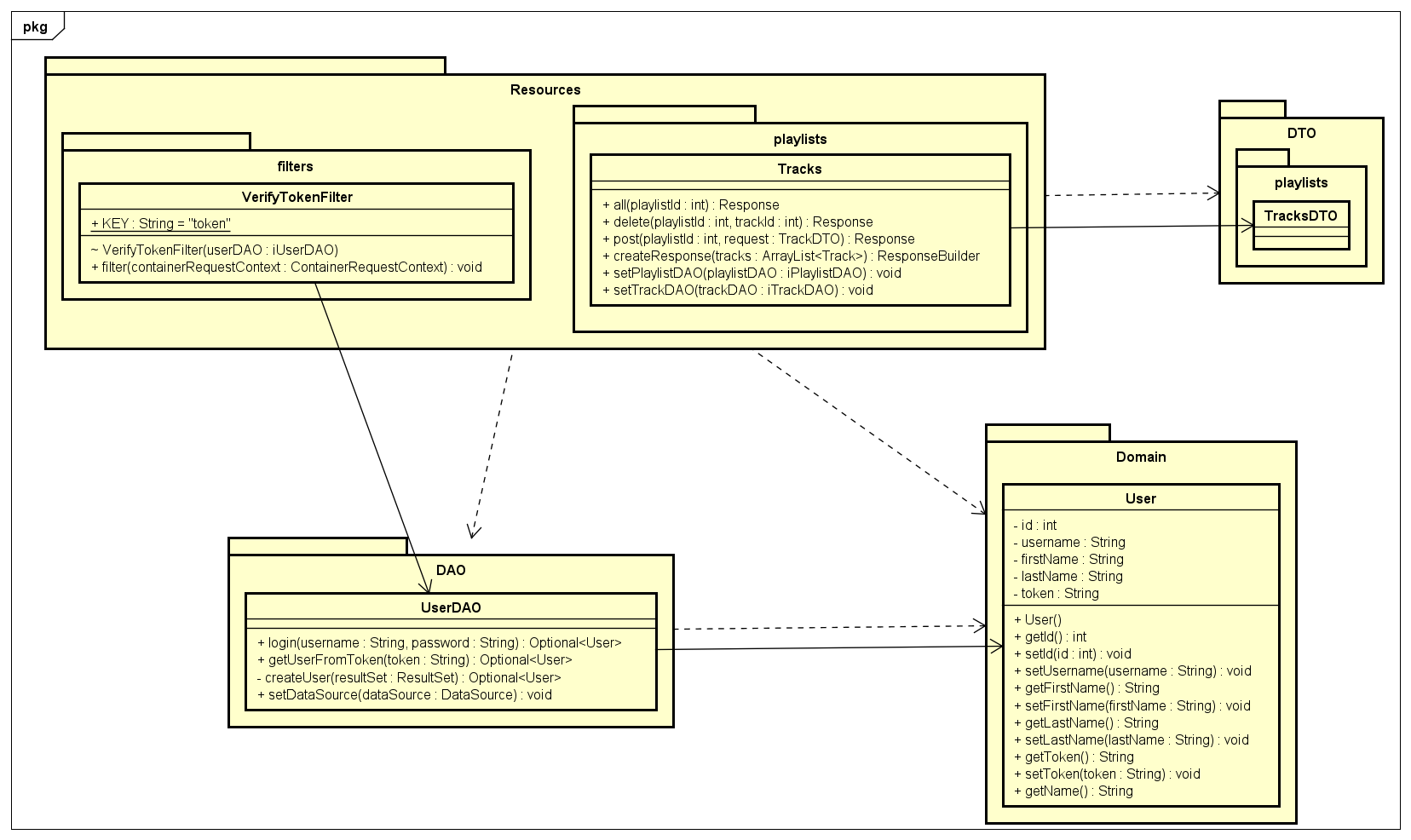
# Inleiding

Dit opleverdocument is voor de casus Spotitube voor de HAN. Er wordt in het package diagram beschreven hoe de structuur van de broncode eruitziet, inclusief uitleg hierover. Verder wordt er ingegaan op de runtime omgeving van de applicatie, welke protocollen er gebruikt worden en overige structuren in het systeem. Ook wordt er ingegaan op overige ontwerpkeuzen en beslissingen.

# Package diagram

Hieronder is het package diagram weergegeven. Er wordt gebruik gemaakt van het Layered Application Design pattern. De presentatie laag bestaat uit REST resources waar de client requests naar toe kan sturen via het HTTP-protocol. De requests gaan mogelijk via een filter om requirements aan resources te controleren. Deze requirements bestaan meestal uit een user token controleren.

De Data Access Object (DAO) laag zorgt ervoor dat data uit de database de applicatie in komt. De gegevens uit de database worden gemapped naar domain object welke vervolgens naar de resources gaan.

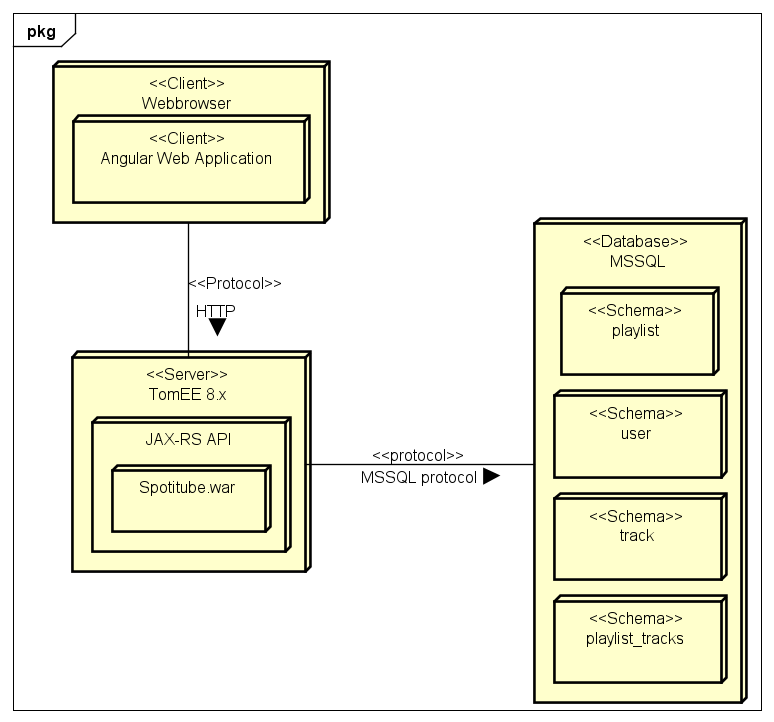


## Alternatieven

Een alternatief voor de structuur van de applicatie zou een Model-View-Controller (MVC)-structuur zijn. Hier zorgt de controller dat de data uit de database komt en mapped dit dat naar models. De view kan deze models op zijn beurt naar de client sturen. Er is niet gekozen voor deze aanpak omdat er een sterke verbinding zit tussen de view en de models. Hierdoor is het moeilijker om classes te uit te wisselen omdat er rekening moet worden gehouden met veel contracten tussen functies.

# Deployment diagram

In het diagram hieronder is te zien dat het systeem bestaat uit 3 verschillende nodes. Op de client draait een webbrowser waar een angular webapplicatie draait. Deze communiceert met de API-server via HTTP. Deze handelt requests van de webapplicatie af doormiddel van JAX-RS op een TomEE server. Deze server haalt data uit een SQL Server database.



## Alternatieven

Gezien de structuur van de data is het mogelijk om gebruik te maken van een NoSQL database. Het voordeel hiervan zou zijn dat het makkelijker zou zijn om data uit de database te krijgen. Ook is er dan minder kans dat queries zich misdragen omdat ze verkeerd zijn geschreven.

In plaats van JAX-RS kon ook Spring worden gebruikt. Dit is een alternatief waar nog geen verder onderzoek naar gedaan is.

# Ontwerpkeuzes

Om de DAO’s te testen is er gebruik gemaakt van integration testing. Ook is er gebruik gemaakt van een test database. Deze is te vinden in het bestand script.sql in de test package. Er is gebruikt gemaakt van integration testing zodat queries en de database ook getest worden. Ook is het erg onpraktisch om DAO’s te unit testen. Dit zorgt er namelijk voor dat er veel klassen moeten gemockt worden wat veel tijd en opzet kost.

In de presentatie laag is er gebruik gemaakt van filters. Dit zijn annotaties in JAX-RS die voor en na een request of response code uitvoeren. Dit filter wordt hergebruikt door annotaties bij de REST resources zodat de code uitgevoerd wordt. Er is in de applicatie momenteel één filter. Deze controleert of het token in de database staat en het token bij de user past.

De DAO’s erven over van een type-specifieke interface en deze interface erft over van een DAO-interface. Dit zorgt ervoor dat er gemakkelijk meerdere concrete implementaties toegevoegd kunnen worden.

# Conclusie

De Spotitube API is succesvol gerealiseerd en geïmplementeerd in het huidige systeem. Er is gebruik gemaakt van het layer pattern door REST resources, DAO’s en domein objecten gescheiden te houden en te gebruiken. Ook is er gebruik gemaakt van JAX-RS om de REST resources open te stellen naar de client.

Ook zijn meer dan 80% van de klassen getest door middel van unit en/of integratie tests.