# Koppeling front-end met MS SQL Server

Wij hebben gekozen voor het gebruik van Angular als front-end met het Spring Boot framework als basis voor de back-end. Spring Boot verzorgd in de applicatie de feitelijke database koppeling door onder andere gebruik te maken van de Java Persistence API (JPA). In de configuratie van de Spring Boot applicatie is het volgende te zien:

*#Databaseconnection***spring.datasource.url**=**jdbc:sqlserver://localhost;databaseName=gameparadisespring  
spring.datasource.username**=**sa  
spring.datasource.password**=**yourStrong(!)Password  
spring.datasource.driverClassName**=**com.microsoft.sqlserver.jdbc.SQLServerDriver  
spring.jpa.show-sql**=**true  
spring.jpa.hibernate.dialect**=**org.hibernate.dialect.SQLServer2012Dialect  
spring.jpa.hibernate.ddl-auto**=**validate***#Flyway***flyway.enabled**=**false***#SQL Debug***logging.level.org.hibernate.SQL**=**DEBUG  
logging.level.org.hibernate.type.descriptor.sql.BasicBinder**=**TRACE***# Naming strategy***spring.jpa.hibernate.naming-strategy** = **org.hibernate.cfg.ImprovedNamingStrategy  
  
domainsList**=**127.0.0.1,nickhartjes.nl,http://localhost:4200**

Onder het kopje “#Databaseconnection” staat de verbinding met de lokale MS SQL Database geconfigureerd.

Een andere SQL tool voor Spring Boot is in hetzelfde bestand ook te bedienen, Flyway. Flyway zorgt ervoor dat database migratie makkelijk uitgevoerd kan worden. Omdat wij binnen de casus vijf SQL scripts gebruiken om de database te updaten gebruiken we Flyway zodat de SQL scripts automatisch in de juiste volgorde uitgevoerd worden. Hiervoor moeten de betreffende SQL script bestanden wel in de juiste volgorde hernoemd worden met een prefix (V1, V2, V3 ... etc.).

Zodra de lokale MS SQL Database draait kan de Spring Boot applicatie gestart worden. Deze zal dan automatisch verbinding maken met de SQL Server en tenslotte de SQL scripts hierop uitvoeren. Daarna is de laatste versie van de database klaar voor gebruik.

Door gebruik te maken van CRUD Repositories en Database Transfer Objects (DTO) binnen Spring Boot programmeren we binnen Spring Boot, middels Java, de diverse database acties die we willen uitvoeren voor het ophalen en wegschrijven van data. Deze data wordt gemapped op de DTO objecten binnen Spring Boot. Deze objecten stellen de entiteiten in de database voor. Op deze manier hebben we de entiteiten in Java code beschikbaar in onze back-end en kunnen we de data vanuit de database hiernaartoe schrijven.

De Angular applicatie kan middels een API call op een van de REST controllers van de Spring Boot applicatie tenslotte data ontvangen die via Spring Boot en met behulp van JPA wordt opgehaald uit de MS SQL Database. Deze data wordt verpakt in het formaat van de DTO objecten zoals deze in Spring Boot zijn geprogrammeerd.