Verantwoordingsdocument

Backend systeem voor een dierenarts

Auteur: **Tanja van Hunen**

Datum: **Februari 2022**

Inleiding

Dit document behoort samen met het technisch ontwerp, installatiehandleiding en source code tot een geheel voor de afronding van het vak ‘backend’ aan Hogeschool Novi te Utrecht. In dit document vind je uitleg over beslissingen en keuzes die gemaakt zijn tijdens het ontwikkelproces.

Inhoudsopgave

Verantwoordingen

1. Java

In eerste instantie heb ik gekozen voor java LTS versie 11. Dit omdat ik gewend ben om te werken in java 8 en hierbij de minste verschillen zaten. Totdat ik de excepties gingen maken en ik toch wel netjes een @Serial wilde gebruiken. Daardoor ben ik toch overgegaan op versie 14 LTS.

2. Repository

Voor de repository heb ik gekozen voor een JPA repository ipv een CRUD repository. Dit omdat JPA repository een extensie is van CRUD repository en ik daarom geen sql queries hoef te maken waardoor er ook minder kans is op fouten.

3. Security

Wat betreft security heb ik gekozen voor basic authentication. Graag had ik een beter beveiligde security gehad maar wegens tijdgebrek ben ik er niet aan toegekomen om mij te verdiepen in de verschillende security mogelijkheden. De basic authentication hebben we tijdens de lessen behandeld en was daarbij een prima keuze om te implementeren.

4. Wachtwoord encryptie

Voor basic authentication is het niet verplicht een wachtwoord te encrypten. Echter omdat het mij niet lekker zat dat ik geen tijd had voor een betere beveiliging heb ik gekozen om de wachtwoorden encrypted op te slaan in de database. Hierbij is gebruik gemaakt van Bcrypt.

5. Database

Voor de database heb ik gekozen om gebruik te maken van Docker. Docker is erg makkelijk in gebruik en platform onafhankelijk. Dit betekend dat iemand met 1 commando een database kan opzetten wat een hoop tijd scheelt.

6. File uploader

Ook dit punt heeft te maken met tijdgebrek. Een file uploader is een eis voor de eindopdracht. De applicatie bevat een file uploader maar deze bestanden komen op een grote hoop wat zeker niet gewenst is. Als ik meer tijd had gehad zou ik dit verder uitwerken tot bijvoorbeeld het uploaden van facturen onder een huisdier of eigenaar. En een losstaand endpoint voor het uploaden van mri/echo/röntgen bestanden wat weer te koppelen is aan een huisdier.

7. Afsprakensysteem

Graag had ik een afsprakensysteem ingebouwd echter heb ik hier niet goed over nagedacht en was dit iets te optimistisch. Een afsprakensysteem vraagt veel kennis en tijd en beveiliging. Ik vind niet dat de applicatie op dit moment goed genoeg beveiligd is voor een afsprakensysteem. Daarnaast is er ook niet genoeg tijd om dit in te bouwen. Met een afsprakensysteem zou er een gebruikersrol bij komen voor een eigenaar om zelf afspraken in te plannen bij een dierenarts. Of voor een receptioniste mogelijkheid geven afspraken in te plannen voor een eigenaar waarbij een afspraakbevestiging wordt gemaild naar de eigenaar.

8. DTO’s

In de applicatie is gebruik gemaakt van Dto’s voor het opvragen van gegevens bij requests. Dit om te voorkomen dat sql injecties plaatsvinden of dat mensen de verkeerde gegevens doorgeven die niet goed verwerkt kunnen worden.

9. Validaties

Door het gebruik van dto’s kon ik hier validaties op plaatsen zoals bijvoorbeeld hoe de geboortedatum ingevuld moet worden bij een huisdier. Dit om te voorkomen dat er maanden en dagen gewisseld worden en de geboortedatum daardoor niet goed staat. Ook zijn er in de servicelaag nog validaties. Dit is een extra validatie bovenop de basic authentication. Dit omdat een gebruiker zijn of haar eigen wachtwoord mag aanpassen. Daarbij wil je dus controleren of de huidige gebruiker wel de gebruiker is waar het wachtwoord aangepast wordt. Daarnaast mag een administrator ook het wachtwoord van andere mensen aanpassen, denk bijvoorbeeld aan een medewerker die op vakantie gaat en zijn of haar wachtwoord niet meer weet bij terugkomst. Daarom wordt er ook gecontroleerd of een huidige gebruiker een administrator rol heeft.

10. Generated ID’s

Voor het gebruik van eigenaren en huisdieren is gebruik gemaakt van id’s. Bij de gebruikers is de gebruikersnaam uniek en kan daarom als id gebruikt worden. Voor de eigenaren had ik zelf een id generator kunnen maken of een request kunnen doen naar de database om het laatst toegevoegde id op te vragen en daar een nummer bij op te tellen, die vervolgens te controleren of het niet voorkomt en dan een id mee geven aan de database voor die gebruiker. Echter heeft Javax persistence een mooie functie genaamd GeneratedValue hiermee kan automatisch een id worden gegenereerd van het meegegeven type (in dit systeem is dat een Long). Ik had hier ook een Integer van kunnen maken maar een Long is meer toekomstproof mochten er heel veel huisdieren worden toegevoegd aan het systeem.

11. Unit tests

Tijdens het maken van de unit testen liep ik tegen veel problemen aan. Voornamelijk bij de controller testen. Deze bleef een 403 code teruggeven terwijl er een mock user gebruikt moest worden wat nu niet lijkt te gebeuren. Helaas door de tijdnood en gebrek aan kennis heb ik dit niet af gekregen.