Deelopdracht 2 Verantwoordingsdocument GDPR-tool
Novi Hogeschool
Student: Floris Henkelman
Opleiding: Full Stack Developer Voltijd
Duur: november 2023 – maart 2025
Datum: 13 november 2025

Inleiding
Dit is het verantwoordingsdocument voor mijn GDPR- tool. In dit document reflecteer ik op mijn leerproces en keuzes, en probeer ik zoveel mogelijk uit te leggen over mijn keuzes. Ik ben een aantal ontwikkelingen vergeten, en ik heb heel veel ideeën om de applicatie verder te ontwikkelen.

Inleiding
Ik heb de applicatie ontwikkeld met een klein beetje maar heel beperkte voorkennis van de basics van Object Oriented Programming, door kleine cursussen gedaan te hebben zoals in Codecademy en een begin gemaakt te hebben aan de gratis Harvard CS50 introductiecursus. Ik wist meteen wat ik wilde bouwen: een tool om de AVG toe te passen. Ik had het idee al jaren. Ik had alleen geen idee waar ik zou moeten beginnen. Het technisch ontwerp heeft mij geholpen om het beter in kaart te brengen. Ik heb mij niet altijd even goed gehouden aan het technisch ontwerp. Dat is omdat ik heel onderwerpen via internet en voornamelijk door YouTube-videos heb gemaakt. Mijn focus is voornamelijk geweest op de functionaliteiten die ik wil laten zien, en hoe makkelijk ik hier informatie over heb kunnen vinden.

De link naar mijn github-project: https://github.com/florishenkelman/gdprtoolfinal

Keuzes

Al mijn keuzes zijn gebaseerd op het beperken van de ontwikkelingstijd. Vanaf het begin van deze studie heb ik eerst lesstof van EdHub gebruikt. Meestal kwam ik hier niet heel ver mee, omdat de onderwerpen nog nieuw waren en ik vaak niet goed kon begrijpen waar iets voor gebruikt wordt. We hebben bijvoorbeeld in lessen geoefend met kleinere opdrachten zoals het maken van classes in een kleine applicatie met auto's en kenmerken, maar hoe zo'n functionaliteit nou bijvoorbeeld in een echte applicatie zou werken kon ik mij niet inbeelden. Dat ligt niet per se aan de leerstof, maar ik heb ADD en ik heb de neiging om veel te moeilijk na te denken over onderwerpen.

Voor het bouwen van applicatie en voor alle programmeertalen die gebruikt zijn heb ik aan de hand van de lesstof YouTube als voornaamste bron gebruikt. Voorbeelden zijn de profielen:
- Programming with Mosh
- Bro Code
- freeCodecamp.org

Waarom ben je van conventies of architecturale richtlijnen afgeweken?

De leerstof van de cursus was een goede start, maar op veel vlakken niet genoeg voor wat ik wilde ontwikkelen. Ik heb heel keuzes gebaseerd op wat ik online heb kunnen vinden over een onderwerp of functionaliteit die relevant was voor wat ik wilde ontwikkelen. Met name YouTube is een van de belangrijkste tools geweest, ook om een userinterface te ontwikkelen. De kernfunctionaliteiten van mijn applicatie, zoals een zoekfunctie of een task-manager zijn al heel vaak ontwikkeld. Het heeft het ontwerpen en ontwikkelen een stuk makkelijker gemaakt.

Welke doorontwikkelingen zijn er mogelijk of misschien zelfs wenselijk, en waarom heb je deze zelf niet door kunnen voeren?

Er zijn heel veel functionaliteiten mogelijk en wenselijk. Ik heb niet alle functionaliteiten die ik wil implementeren toegevoegd door een gebrek aan tijd. Ik wil alle verplichtingen uit de AVG die vereisen dat je iets bijhoudt in de applicatie verwerken. Ik wil bijvoorbeeld registers in de applicatie

ontwikkelen, zoals het register voor verwerkingsactiviteiten, register voor datalekken, register voor rechten van betrokkenen die zijn behandeld. Dit zijn functies die ik in de toekomst wil toevoegen, maar die voor deze eerste versie niet noodzakelijk zijn.

Ik wil jurisprudentie toevoegen, waarbij de gebruiker op de hoogte gehouden wordt van ontwikkelingen. Ik weet nog niet hoe ik dat precies wil doen, maar ik denk bijvoorbeeld aan een soort nieuwsfeed die verwijst naar bestaande nieuwswebsites of zelfgeschreven artikelen toevoegen in de applicatie.

Op dit moment kan iedere gebruiker alle gebruikers zien, en de gebruikersrol. Ook kun je klikken op profiel wijzigen van een andere gebruiker, alleen afhankelijk van je rol werkt het niet. Dit is iets wat ik in de toekomst wil verbeteren zodat de functionaliteiten alleen zichtbaar zijn op basis van de rol.

Mijn leerproces en reflectie
Ik heb het lang geworsteld met de bootcamp, en heb de start als heel moeizaam ervaren. Het heeft heel lang geduurd voordat ik de basisprincipes van bijvoorbeeld Java door had. Mijn grootste struikelblok is geweest dat ik mij niet kon inbeelden waar een bepaalde functie voor was. Bijvoorbeeld classes of entities. Ik kon het uiteindelijk wel begrijpen in een huiswerkopdracht, maar nog steeds had ik er geen beeld bij hoe het in de praktijk gebruikt zou worden. Ik heb hier heel veel tijd mee verloren. Daardoor begon ik achter te lopen met de frontend. De frontend vond ik echter makkelijk te volgen. Een volgende keer zou ik eerder om hulp vragen. Ik heb heel erg het gevoel gehad dat ik achterliep, en daardoor wilde ik het alleen inhalen. Ik heb het als heel moeilijk ervaren.

Ik ben heel blij dat ik het na een jaar een werkende applicatie heb. Het is een hele grote stap in een nieuw vakgebied, en na heel veel frustratie is het toch gelukt. Toch ben ik niet tevreden over mijn leerproces en ontwikkeling. Ik heb heel veel tijd besteed aan het oplossen van problemen met de database. PgAdmin4 heb ik continu gebruikt, maar zo liep ik een aantal weken geleden tegen het probleem aan dat mijn applicatie vanuit Java niet startte. Ik heb de storingen in de IDE uitgelezen, forums gecheckt, en alles wees naar een probleem met de localhost 5173 of port 8080 op mijn MacBook. Ik heb eindeloos gezocht naar de problemen die hier mee te maken kunnen hebben. Dit was niet het probleem. Het probleem bleek te zijn dat de applicatie bleef zoeken naar een database, terwijl PgAdmin niet goed liep. Dat probleem had ik voorheen niet, en ik kon het programma recent wel zonder problemen gebruiken. Na PgAdmin opnieuw te installeren kreeg ik de applicatie weer aan de praat.

Daarnaast ben ik heel veel vergeten, of kwam ik niet verder. Op dit moment zul je in de applicatie zien dat er bij het inlogscherm geen ‘wachtwoord vergeten’ is. Ik had het wel genoteerd in mijn functional requirements document voordat ik begon met ontwikkelen, maar ik ben het door verschillende problemen vergeten.

In de GDPR-articles kun je artikelen opslaan. Dit is nog niet interactief. Je kunt een artikel wel opslaan, maar je kunt er verder niks mee. In de toekomst wil ik dat je op het artikel kunt klikken., zodat je doorverwezen wordt naar het hele artikel.

De applicatie heeft een zoekfunctie, en gelukkig werkt deze. Ik wil in de toekomst dat de woorden die je in een artikel als resultaat hebt gemarkeerd worden, zodat het meer overzichtelijk is.

Een document kan geüpload worden, maar het downloaden of de zichtbaarheid van een geüpload document werkt nog niet helemaal.

Ik denk dat het belangrijkste voor een volgend project of doorontwikkeling is om een beter overzicht te houden. Dat heb ik niet altijd even goed gedaan omdat ik te gefocust was op het oplossen van backend problemen.

Technische keuzes

1. Dark theme

In eerder stadium heb ik een dark-theme toggle gemaakt met Tailwind. Met een darkMode class kun je namelijk eenvoudig een schakelaar toevoegen. Het voordeel hiervan is dat je minder code hoeft te schrijven en het daarom makkelijker kunt beheren. Omdat Tailwind niet gebruikt mocht worden heb ik uiteindelijk per pagina code voor de frontend toegevoegd. Het voordeel is dat ik heb geleerd om het 'handmatig' te doen. Ik geef over het algemeen de voorkeur aan bepaalde de opmaak die ik met Tailwind heb kunnen maken. Dit heeft voor mij veel meer mogelijkheden om de opmaak te krijgen zoals ik in gedachte had. Ook geloof ik dat het een plus om het zowel handmatig als met Tailwind te kunnen doen. Werkgevers zoeken vaak iemand die iets makkelijk en snel kan oplossen.

2. Foutafhandeling met toastmeldingen

Ik gebruik react-hot-toast voor het weergeven van fout- en succesberichten. Toast-meldingen bieden niet-opdringerige gebruikersfeedback, wat handig is om gebruikers te informeren over bijvoorbeeld een succesvolle of mislukte handeling. Het gebruik van react-hot-toast vereenvoudigt het proces en zorgt voor een schone, consistente weergave van meldingen.

3. Cascading delete regels

Ik gebruik cascading delete (ON DELETE CASCADE en ON DELETE SET NULL) om gerelateerde data automatisch te beheren bij verwijderingen. Dit voorkomt handmatige database-updates en zorgt voor dataconsistentie. Wanneer een gebruiker wordt verwijderd blijven taken behouden maar de creator/assignee wordt op NULL gezet. Bij het verwijderen van een taak worden comments en attachments automatisch verwijderd, en bij het verwijderen van een gebruiker worden ook zijn opgeslagen artikelen verwijderd. Het gebruik van cascading delete vereenvoudigt het databasebeheer en zorgt voor een schone, efficiënte manier om verwijderingen af te handelen.

4. Gebruik van enumeraties

Eumeraties zorgen voor dataconsistentie en voorkomen ongeldige invoer doordat alleen vooraf gedefinieerde waarden mogelijk zijn. Dit maakt validatie sneller en veiliger dan bij een VARCHAR-kolom in een database. Bovendien verbeteren enumeraties de leesbaarheid en onderhoudbaarheid van code en kunnen ze efficiënt worden opgeslagen als numerieke waarden in databases. Dankzij hun flexibiliteit kunnen extra functies zoals prioriteitsniveaus worden toegevoegd. Het maakt je code makkelijker te begrijpen en te onderhouden.

5. Maven excetuable

Ik gebruik de Maven Wrapper (mvnw) om consistentie, draagbaarheid en eenvoud te garanderen in Maven-projecten. Developers hoeven Maven niet handmatig te installeren of bij te werken, omdat iedereen automatisch met dezelfde versie werkt. Dit voorkomt versieconflicten wanneer aan verschillende projecten met verschillende Maven-versies wordt gewerkt. Bovendien maakt de wrapper Maven-upgrades eenvoudiger, zonder dat er veel aangepast hoeft te worden. Hierdoor hoeft er minder geconfigureerd te worden of geïnstalleerd, meer stabiliteit en een soepelere samenwerking. Ik heb bij het proberen te installeren van Maven op mijn MacBook namelijk heel veel problemen gehad. Via een forum heeft iemand aangeraden om een wrapper te gebruiken.

6. Groepering fronted gebaseerd op functionaliteit in plaats van type bestand (domain driven)

Ik heb de frontend ingedeeld op functionaliteit in plaats van bestandstype. Hierdoor blijven gerelateerde componenten, zoals UserProfile en Dashboard, bij elkaar, wat de code duidelijker, beter schaalbaar en makkelijker te onderhouden maakt. Dit past bij Domain-Driven Design (DDD) en maakt het ontwikkelen, debuggen en uitbreiden eenvoudiger, zonder door losse CSS-, JS- en HTML-mappen te zoeken.

7. Focus op het scheiden van controllers, services en repositories

Ik heb me goed verdiept in het maken van controller-, service- en repository-mappen. Ik vond het ingewikkeld en ik heb hier veel YouTube-video's voor gebruikt. De code is per onderwerp gescheiden, waardoor het overzichtelijk is en de logica voor een bepaalde functionaliteit gescheiden is. Dit maakt de doorontwikkeling van de applicatie in de toekomst eenvoudiger, omdat ik de applicatie daadwerkelijk wil uitbrengen en hier mogelijke professionele ontwikkelaars aan zullen gaan werken.

8. Tokengebaseerde authenticatie (JSON Web Tokens)

Ik heb ook gekeken naar Spring Security (zonder JWT), omdat dit gemakkelijker in gebruik schijnt te zijn. Ik heb echter gekozen voor JWT omdat de bevestiging sneller is, het beter schaalbaar is en minder afhankelijk is van externe factoren zoals servers. Spring Security is minder belastend voor de applicatie.