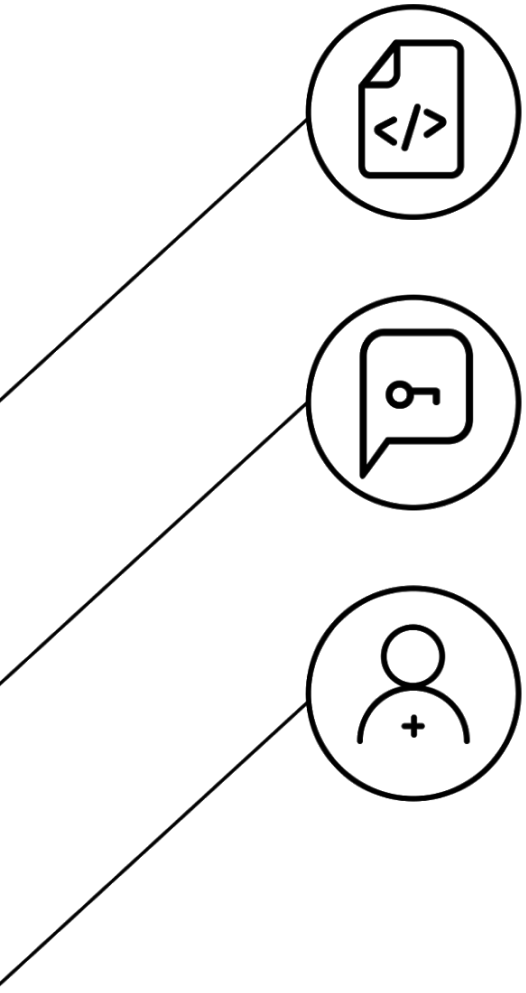

Eindopdracht

Bootcamp Full Stack Developer

Backend & Frontend

Opdrachtbeschrijving en deelopdrachten

Randvoorwaarden, structuur en beoordeling

NOVI HOGESCHOOL

Inhoud

Integrale eindopdracht Bootcamp FSD (60 EC)

De bootcamp Full Stack Developer behandelt zowel de leerlijn Backend als de leerlijn Frontend. Om de bootcamp af te ronden dien je de volgende leeruitkomsten aan te tonen:

Frontend

1. **HTML & CSS**
   De student bouwt statische webpagina's met HTML, voorziet deze van styling en layout door CSS en ontwerpt een prototype in Figma (of Adobe XD, deze software is echter niet langer gratis) waarmee hij een visueel design omzet naar een webpagina.
2. **Javascript**
   De student schrijft en test schone, gestructureerde JavaScript code waarmee hij webpagina's voorziet van interactie en gebruikt hierbij een REST API.
3. **React**
   De student bouwt een interactieve en modulaire webapplicatie in React en maakt hierbij gebruik van herbruikbare interface elementen, state management en de life cycles van React.

Backend

4. **Java Programming**
   De student programmeert in Java, waarbij hij OOP-structuren toepast. Hierbij past de student automatisch testen toe en beheert hij externe code met behulp van Maven, waardoor men in een team aan Java-projecten kan werken.
5. **Backend Documentatie**
   De student stelt, op basis van de Software Development Life Cycle, technische documentatie op voor de backend van een applicatie, gebruik makend van UML-diagrammen.
6. **Database Development**
   De student ontwerpt een relationele database, waarin data met onderlinge relaties veilig opgeslagen en uitgelezen kan worden, aan de hand van een technisch ontwerp document. Tevens beheert de student de data en rechten van databasegebruikers en voert hij CRUD-opdrachten uit op de database.
7. **Spring Boot**
   De student zet een backend applicatie op met behulp van het Spring-boot framework en maakt gebruik van verschillende architecturale lagen binnen Spring. De student test zijn applicatie met unit-testen en het *mocken* van klassen en tevens communiceert de applicatie met een database.
8. **Design Patterns & Clean Code**
   De student schrijft zijn code volgens de afgesproken conventies van Clean Code en ontwikkelt highly cohesive en loose coupled code, door de toepassing van Design Patterns en SOLID.

Om de bootcamp Full Stack Developer met succes af te ronden is een voldoende nodig voor de integrale eindopdracht; met de deelopdrachten kunnen geen losse cursussen worden afgerond.

Algemene opdrachtbeschrijving

Dagelijks gebruiken we online applicaties zoals Microsoft Teams, Google Drive en Youtube. Al deze applicaties zijn voorzien van zowel een frontend als een backend. De backend is verantwoordelijk voor de opslag van data, de frontend zorgt ervoor dat deze data op een gebruiksvriendelijke manier aan de gebruiker wordt getoond.

Tijdens deze bootcamp zul je vakinhoudelijke kennis gaan opdoen in beide gebieden. Bovendien ga je jouw vaardigheden op het gebied van software development toepassen, door zelf een applicatie te bouwen. Door al jouw opgedane kennis toe te passen in deze eindopdracht, laat je zien dat je je hebt ontwikkelt tot Junior Full Stack Developer.

Je gaat een applicatie bedenken en bouwen waarvan je zowel de backend als de front end gaat programmeren. Je bedenkt welk probleem je wilt oplossen met jouw product en aan welke eisen de applicatie moet voldoen. Voordat je begint met programmeren, maak je een gestructureerd plan van aanpak en tijdens het ontwikkelen documenteer je jouw keuzes. Al deze stappen zijn opgedeeld in deelopdrachten.

Mocht je zelf geen idee hebben wat je zou kunnen programmeren, dan kun je de voorbeeldcasus gebruiken voor jouw eindopdracht. Uiteraard ben je vrij om zelf iets bedenken.

Op te leveren producten:

- Functioneel ontwerp
- Technisch ontwerp
- Broncode frontend
- Broncode backend
- Installatiehandleiding
- Verantwoordingsdocument

Deze documenten worden ingeleverd als één ZIP-bestand.

Voorbeeldcasus

DJ Don Diablo - Promo Delivery system

De Nederlandse DJ Don Diablo is op zoek naar een systeem dat producers in staat stelt om makkelijk een demo in te zenden via een browser portal. Binnen dit portaal kan de producer zijn mp3 uploaden en gegevens invullen. Als de producer een ander bestandstype uploadt dan moet het automatisch worden omgezet naar 320 kbps MP3. De producer dient automatisch een bevestigingsmelding te ontvangen na het inzenden van zijn demo.

Demo's worden aan de achterkant beluisterd door het team van Don. Hier moet het ook mogelijk zijn om de demo te downloaden. Ook moet er een functie zijn om een reactie op de demo te geven met meerdere vooraf ingestelde teksten die aanpasbaar zijn. De voornaam van de geadresseerde moet al automatisch worden ingevuld.

Voorbeeld van een demo drop formulier: https://www.hexagonhq.com/demo

Deelopdrachten

De eindopdracht wordt verdeeld in vier fases. Je begin met de ideefase: hierin bedenk je wat voor applicatie je gaat ontwikkelen en stel je kaders. Vervolgens ga je vaststellen aan welke eisen jouw applicatie moet voldoen en vertaal je dit naar een functioneel- en technisch ontwerp. Dit noem je de ontwerpfase. Wanneer je hiermee klaar bent zul je aan de programmeerfase beginnen en ten slotte maak je jouw project klaar voor oplevering in de opleveringsfase.

Je begint met een probleem, dat jij wilt gaan oplossen met de applicatie die je gaat bouwen. Je kunt kiezen om een eigen casus te bedenken, of de wensen van de voorbeeldcasus vertalen naar een idee voor jouw webapplicatie. Beschrijf welk probleem je met jouw eindopdracht wilt oplossen voor jouw gebruikers en hoe je dit zou willen doen. Let op: Dit is geen bedrijfsplan. Je mag gerust een idee gebruiken dat al bestaat. Je beschrijft je idee in **maximaal 250 woorden** en dit lever je **ter goedkeuring** aan bij jouw docenten.

De docenten beoordelen jouw idee op de volgende punten:

- Is de hoeveelheid werk te weinig, voldoende of te veel;
- Worden alle eisen die gesteld zijn behandeld?

Na goedkeuring van de docent over jouw probleemstelling en de oplossing die jij daarvoor in gedachte hebt, kun je aan de slag met de eerste deelopdracht.

Mocht je zelf geen idee hebben wat je zou kunnen programmeren, dan kun je de voorbeeldcasus gebruiken voor jouw eindopdracht.

Deelopdracht 1. Ontwerpfase

Wanneer jouw idee is goedgekeurd ga je beginnen. Voordat je aan de slag kunt met het ontwikkelen van de applicatie moet eerst nog het één en ander worden uitgezocht. In deze deelopdracht ga je informatie verzamelen en achterhalen aan welke eisen jouw applicatie moet voldoen om het probleem van de gebruiker op te lossen. Om dit in kaart te brengen schrijf je een uitgebreide lijst van functionele- en niet functionele eisen. Vervolgens kun je jouw use-cases maken, waarin je zowel nadenkt over het positieve als het alternatieve scenario. Vervolgens ga je nadenken over de indeling en het uiterlijk van de pagina's door middel van inspiratiebronnen, wireframes en tenslotte schermontwerpen. Een functioneel ontwerp is bedoeld als basis voor het doorontwikkelen van de applicatie en moet ook door iemand zonder IT-achtergrond te begrijpen zijn.

Op basis van het functioneel ontwerp maak je een technisch ontwerp. Uit het technisch ontwerp is eenvoudig te lezen wat ontwikkeld en opgeleverd gaat worden in de backend. Je maakt een klassendiagram en minimaal twee relevante sequentiediagrammen. Indien wenselijk stel je ook andere diagrammen op. Stel onderstaande diagrammen volgens geldende richtlijnen op en maak correct gebruik van relevante technieken.

De sequentiediagrammen moeten van twee relevante use-cases zijn. Daar bedoelen we mee dat als jij een reserverings-applicatie maakt voor een kapper, dat je bijvoorbeeld een sequentiediagram maakt voor het plaatsen van een reservering en niet voor iets wat elke applicatie heeft, zoals *registreren* of *inloggen*.

Je gaat een functioneel en technisch ontwerp opleveren dat voldoet aan de volgende eisen:

- Bevat een titelblad, inleiding en inhoudsopgave. In het documenten zitten geen verwijzingen naar afbeeldingen en informatie buiten het document zelf.
- Beschrijft het probleem en de manier waarop deze applicatie dat probleem oplost.
- Beschrijft wat de applicatie moet kunnen middels een sommering van tenminste 50 functionele- en niet-functionele eisen. Dit hoeft niet evenredig verdeeld te zijn.
- Bevat een verzameling van minimaal 3 verschillende inspiratiebronnen (screenshots of foto's van andere applicaties) die gebruikt zijn om de visuele stijl van de applicatie te bepalen.
- Beschrijf de belangrijkste gebruikershandelingen in minimaal vier use case tabellen. Deze use case tabellen ga je uiteindelijk uitwerken tot vier belangrijke kern-functionaliteiten van jouw project.
- Bevat de gescande (of gefotografeerde) wireframes, geschetst op papier, van tenminste vijf pagina's. Deze dienen goed leesbaar te zijn, inclusief begeleidende tekst waarin duidelijk wordt over welke pagina het gaat.
- Bevat screenshots van de schermontwerpen met titels en beschrijvingen.
- Bevat een klassendiagram van alle entiteiten.
- Bevat minimaal twee uitgewerkte sequentiediagrammen.
- Bevat een componentendiagram (optioneel).

Op basis van deze informatie ga je schermontwerpen maken in Figma/Adobe XD. Deze bevatten dusdanig veel uitgewerkte details zodat je tijdens het programmeren geen ontwerpbeslissingen meer hoeft te maken.

Op te leveren:

- Het functioneel en technisch ontwerp van de webapplicatie in PDF (.pdf) of markdown (.md).
- Link naar het (openbare) Figma project óf bijgevoegde Adobe XD project in .xd formaat.

Deelopdrachten 2 & 3: Programmeerfase

Je hebt zojuist de ontwerpfase afgerond, het is nu tijd om de applicatie te bouwen! In de programmeerfase ga je zowel de backend als de frontend uitwerken. Hieronder vind je de bijbehorende deelopdrachten. De wijze waarop je de applicatie bouwt, sluit aan bij de eisen die je hebt opgesteld in de ontwerpfase. Je applicatie dient een multi tier applicatie te zijn.

In de programmeerfase houd je rekening met de volgende algemene punten:

- Multi-tier application: frontend, backend en database;
- Je beheert je code met gebruik van Git om met versiebeheer de voortgang van het project vast te leggen.
- Je applicatie bevat een vorm van media-upload. Een eindgebruiker moet bijvoorbeeld een afbeelding, een mp3-bestand of een pdf kunnen uploaden;
- Je applicatie bevat unit-testen.

Deelopdracht 2. Programmeerfase - backend

De backend van de applicatie ontwikkel je in Java, met behulp van het Spring Boot framework. Afhankelijk van de complexiteit van het op te lossen probleem, gebruik je OOP-structuren zoals overerving, interfaces en abstracte klassen. Met behulp van Spring Security pas je autorisatie en authenticatie toe.

Je gebruikt HTTP-methods om de vertaalslag te maken naar acties met de data. Je gebruikt verschillende componenten van Spring-boot, zoals (maar niet gelimiteerd tot) de annotaties en configuratie van de applicatie. Je let hierbij op de principes van Clean Code, Design Patterns en SOLID.

Je test geschreven klassen individueel met unittests die gebruik maken van de drie A's, waarbij de test coverage minimaal 50% is exclusief getters en setters. Ook voer je integratie-testen uit door de application context van de applicatie te testen gebruikmakend van Spring Boot test en WebMvc.

Je past de principes Maven build lifecycle toe bij het beheren van externe code en libraries. Alle data sla je op in een SQL database waarbij je zorg draagt voor de autorisatie en authenticatie in de database.

Houd verder bij het programmeren rekening met onderstaande punten:

- Je ontwikkelt volgens OOP-principes en met behulp van Java & Spring-boot;
- Je maakt gebruik van Maven;
- Je applicatie is beveiligd en bevat meerdere user rollen;
- Je maakt gebruik van een relationele database.

Op te leveren:

- Broncode backend

Deelopdracht 3. Programmeerfase - frontend

Naast een backend heeft jouw applicatie natuurlijk ook een frontend gedeelte. Voor de frontend maak je gebruik van de React library. Je bouwt de applicatie met semantische HTML-elementen en maakt gebruik van schone, gestructureerde CSS door modulaire styling aan specifieke componenten te koppelen. Je maakt gebruik van functies, arrays en objecten en past clean code principes toe. Verder deel je de applicatie op logische wijze op in herbruikbare componenten waarbij data wordt doorgegeven middels properties. Je voorziet die componenten van passende interactie en externe data door correct gebruik te maken van de React Life Cycle Hooks en AJAX.

Daarnaast houd je rekening met onderstaande punten:

- Gebruikers kunnen een account aanmaken en daarmee inloggen. Een gedeelte van de content is alleen te bezoeken voor geregistreerde gebruikers (zoals bijvoorbeeld een profiel-pagina);
- De applicatie kan bestanden zoals afbeeldingen, pdf-bestanden of muziek-bestanden verwerken en opslaan;
- Er wordt gebruik gemaakt van React Context;
- Je schrijft jouw styling zelf en maakt géén gebruik van out-of-the-box styling systemen zoals Bootstrap, Material-UI of Tailwind;
- Je maakt gebruik van NPM.

Op te leveren:

- Broncode frontend

Deelopdracht 4. Opleveringsfase

Zodra je begint met programmeren, ga je ook beginnen aan het verantwoordingsdocument. Hierin leg je vast welke technische ontwerpbeslissingen je maakt en *waarom* je deze keuzes gemaakt hebt. Dit werkt het beste als je dit al tijdens het ontwikkelen van jouw product begint te maken. Waarom heb je deze specifieke npm package of Java-library gebruikt en niet een andere? Waarom ben je van conventies of architecturale richtlijnen afgeweken? Welke doorontwikkelingen zijn er mogelijk of misschien zelfs wenselijk, en waarom heb je deze zelf niet door kunnen voeren? Heb je bijvoorbeeld iets achterwege gelaten in verband met een tekort aan tijd? Leg dan uit wat je liever had willen doen als je meer tijd had gehad. Reflecteer hierbij ook op je eigen leerproces: wat ging goed en wat kan de volgende keer beter? Let op: technieken die als randvoorwaarden gesteld zijn in de eindopdracht tellen niet mee.

Daarnaast schrijf je een installatiehandleiding die uitlegt hoe de applicatie geïnstalleerd kan worden en hoe deze gebruikt kan worden. Je neemt hierin een lijst op van benodigdheden om de applicatie te kunnen runnen, een lijst met (test)gebruikers en user-rollen, een lijst van rest endpoints (inclusief JSON-voorbeelden) en op welke manier deze beveiligd zijn. Ook voeg je een stappenplan met installatie instructies toe.

Op te leveren:

- Verantwoordingsdocument in PDF (.pdf) of markdown (.md) met daarin:
    - Minimaal 10 beargumenteerde technische keuzes.
    - Een beschrijving van limitaties van de applicatie en beargumentatie van mogelijke doorontwikkelingen.
- Installatiehandleiding in PDF (.pdf) of markdown (.md).

Randvoorwaarden

Hieronder vind je een aantal randvoorwaarden waaraan je eindopdracht moet voldoen. Deze randvoorwaarden hebben een verplicht karakter.

- Alleen React met JavaScript wordt geaccepteerd als programmeertaal in de frontend (geen TypeScript).
- Alleen Java wordt geaccepteerd als programmeertaal in de backend. Je maakt gebruik van het Spring-boot framework.
- Je hebt Maven en NPM gebruikt als dependency manager.
- De applicatie start op zonder te crashen.
- De projecten zijn geüpload naar een GitHub repository: deze repository staat op public.
- Het ingeleverde bestand bevat geen 'node_modules'- of '.idea'-map, 'target'-map of 'out'-map.
- Het frontend project bevat de JavaScript linter ESLint. Bij gebruik van create-react-app is dit standaard meegeleverd.
- De wireframes zijn getekend op papier.
- Het prototype is ontworpen met Figma. Indien je liever Adobe XD wil gebruiken, mag dit ook. Deze software is helaas niet langer gratis.
- Er is géén gebruik gemaakt van out-of-the-box styling systemen zoals Bootstrap, Material-UI of Tailwind.
- Het volledige project en bijbehorende documenten wordt aangeleverd d.m.v. een **ZIP-bestand van maximaal 50 MB**. (Met de extensie .zip. Projecten die als .rar worden aangeleverd, worden niet geaccepteerd.)

Het ZIP-bestand bevat de volgende elementen:

- **Readme** (*in markdown*)
    - Beschrijft waar alle documenten en applicaties te vinden zijn
    - Bevat een link naar de GIT repository waarnaar het project (of projecten) geüpload is.
- **Installatiehandleiding** *(in .pdf of markdown)*
- **Functioneel ontwerp** *(in .pdf of markdown)*
- **Technisch ontwerp** *(in .pdf of markdown)*
- **Broncode van beide projecten** (Let op: dus niet alleen de link naar het Github-project)
- **Verantwoordingsdocument** *(in .pdf of markdown)*

Structuur

De installatiehandleiding en het technisch- en functioneel ontwerp zijn beide voorzien van een inhoudsopgave en inleiding. Ze zijn verzorgd en hebben een titelblad met datum en naam van de auteur. De documenten bevinden zich ook op een logische plaats in het inleverbestand.

Naast de bijgevoegde losse schermontwerpen introduceer je deze schermontwerpen ook al te bij het functioneel ontwerp. Ook wireframes behoren voorzien te worden van paginatitels en beschrijving zodat het duidelijk is over welke pagina het gaat. Er is geen harde richtlijn met betrekking tot het aantal woorden, maar zorg ervoor dat je beknopt en bondig schrijft.

In de documenten (installatiehandleiding en het technisch- en functioneel ontwerp) zitten geen verwijzingen naar afbeeldingen, diagrammen of modellen buiten het document zelf. Ook zijn de diagrammen en afbeeldingen goed leesbaar en tekstueel uitgelegd of beschreven. Een goed uitgangspunt hier is wanneer het document geprint wordt, deze nog steeds volledig beoordeeld kan worden.

Beoordelingscriteria

De eindopdracht wordt beoordeeld op basis van de volgende beoordelingscriteria. Per criterium kent de beoordelaar een aantal punten toe. Hierbij wordt zowel gekeken naar de feitelijke realisatie van de leeruitkomst als naar het door de student getoonde inzicht in de bijbehorende theorie.

# Deelopdracht	**Leeruitkomsten en beoordelingscriteria**	**Weging**	**Score in cijfers van 1 t/m 10**	**Weging maal score**
1. Ontwerpfase	**In deze deelopdracht worden aspecten van de cursussen Backend Documentatie en HTML & CSS getoetst.**	**20%**		
Criterium 1.1	De student schrijft een functioneel ontwerp, voorzien van een helder omschreven probleem en de manier waarop de applicatie dit probleem oplost, bij de beoogde oplossing passende functionele en niet-functionele eisen (minimaal 50), een verzameling van visuele inspiratiebronnen (minimaal 3 verschillende), use case tabellen (minimaal 4) en wireframes (minimaal 5 pagina's).	5%		
Criterium 1.2	De student ontwerpt een kwalitatief prototype met Figma/Adobe XD door middel van minimaal 5 schermontwerpen.	5%		
Criterium 1.3	De student structureert de ontworpen functionaliteiten in een klassendiagram, waarbij hij rekening houdt met het vertalen van het klassendiagram naar een databasemodel (RRM) door Spring. Het klassendiagram is taal- en platformonafhankelijk en hoeft geen methodes te bevatten.	5%		
Criterium 1.4	De student legt in het technisch ontwerp op logische en correcte wijze alle verschillende architecturale lagen op juiste wijze vast in de twee sequentiediagrammen en levert deze aan volgens de principes van de Software Development Life Cycle. Deze sequentiediagrammen bevatten de juiste klasse- en methode namen.	5%		
Feedback door de docent				

2. Backend	**In deze deelopdracht worden aspecten van de gehele leerlijn Backend getoetst.**	**35%**		
Criterium 2.1	De student maakt Java applicaties waarbij hij op de juiste momenten, afhankelijk van de complexiteit van het op te lossen probleem, OOP-structuren gebruikt zoals overerving, interfaces en abstracte klassen en de student schrijft Java-code op basis van Clean Code, Design Patterns en SOLID.	10%		
Criterium 2.2	De student gebruikt HTTP-methods om de vertaalslag te maken naar acties met de data.	5%		
Criterium 2.3	De student past autorisatie en authenticatie toe met behulp van Spring Security en de student gebruikt verschillende componenten van Spring-boot zoals, maar niet gelimiteerd tot, de annotaties en configuratie van de applicatie.	5%		
Criterium 2.4	De student voert integratie-tests uit door de application context van zijn applicatie te testen. Hiervoor gebruikt hij Spring-Boot test en WebMvc. Tevens voert de student unittests uit die gebruikmaken van de drie A's, waarbij de test coverage minimaal 50% is exclusief getters en setters).	5%		
Criterium 2.5	De student past de principes Maven build lifecycle toe bij het beheren van externe code en zijn libraries en de student beheert zijn code met gebruik van Git om met versiebeheer de voortgang van het project vast te leggen. De student heeft kleine commits, maakt pull requests en mergt regelmatig.	5%		
Criterium 2.6	De student leest en bewerkt data met behulp van SQL, JPA en Hibernate en draagt zorg voor de autorisatie en authenticatie in de database.	5%		
Feedback door de docent				
3. Frontend	**In deze deelopdracht worden aspecten van de gehele leerlijn Frontend getoetst.**	**35%**		
Criterium 3.1	De student bouwt de applicatie met semantische HTML-elementen en maakt gebruik van zelfgeschreven, schone, gestructureerde CSS door modulaire styling aan specifieke componenten te koppelen.	5%		

Criterium 3.2	De student schrijft schone, gestructureerde JavaScript code waarin op correcte wijze functies, arrays en objecten worden gebruikt en clean code principes worden toegepast.	5%		
Criterium 3.3	De student deelt zijn applicatie op logische wijze op in kleine herbruikbare componenten waarbij data op de juiste manier wordt doorgegeven middels properties en callback properties.	10%		
Criterium 3.4	De student voorziet componenten van passende interactie en externe data door correct gebruik te maken van alle React Life Cycle Hooks en AJAX.	10%		
Criterium 3.5	De student implementeert vier belangrijke kern-functionaliteiten op correcte wijze, waaronder het authenticatie-proces van registratie/inloggen, bestands-up- en download en twee overige zelfbedachte, complete use cases.	5%		
Feedback door de docent				
4. Opleveringsfase	**In deze deelopdracht worden aspecten van de cursus Backend Documentatie getoetst.**	**10%**		
Criterium 4.1	De student schrijft een duidelijk geschreven installatiehandleiding waarmee de applicatie door derden in een andere omgeving kan worden geïnstalleerd, voorzien van stappenplan, lijst van benodigdheden, testgebruikers, userrollen, rest-endpoints en voorbeeld JSON.	5%		
Criterium 4.2	De student levert een verantwoordingsdocument op waarbij hij reflecteert op zijn code en beargumenteert minimaal 10 gemaakte technische keuzes. De student beschrijft realistische limitaties van zijn applicatie en welke doorontwikkelingen mogelijk en/of wenselijk zijn.	5%		
Feedback door de docent				
	Totaal	**100%**		