# Template herkansingsfeedback

|  |  |
| --- | --- |
| Leerlijn: Fullstack Developer | Versie eindopdracht: |
| Naam student: Mustafa Dogan | Startmoment: |
| Studentnummer: | Poging 1 |

\* Bij meerdere pogingen geef je in de tabellen aan welke feedback bij welke poging hoort.

Heb je jouw eindopdracht ingeleverd en heb je een onvoldoende gehaald? Dan verwerk je een extra hoofdstuk in jouw eindopdracht waarin je aantoont wat je precies hebt verbeterd naar aanleiding van de feedback van de docent.

Dit doe je per beoordelingscriterium, waarin je ook de feedback en originele cijfer benoemt. Gebruik hiervoor onderstaand template. In het geval dat je feedback *niet* verbeterd hebt, geef je dit ook aan.

|  |  |
| --- | --- |
| Beoordelingscriteria 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). | |
| *Feedback docent poging 1:* Je hebt een document ingeleverd met de naam functioneel ontwerp, maar het document voldoet niet aan de randvoorwaarden. Jouw functioneel ontwerp mist een heldere probleem omschrijving, 50 functionele en niet functionele eisen, 4 kwalitatieve use case's, 3 inspiratiebronnen en 5 duidelijke vormgegeven wireframes. Er kunnen geen punten worden gegeven aan dit onderdeel. | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1:  Verbetering: Gedaan het bijgewerkte functionele ontwerp en technische ontwerpdocument, samen met de bijgewerkte sequentiediagrammen en klassediagram. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| Beoordelingscriteria 2: De student ontwerpt een kwalitatief prototype met Figma/Adobe XD door middel van minimaal 5 schermontwerpen. | |
| *Feedback docent poging 1:* Je hebt screenshots ingeleverd van jouw gemaakte applicatie. Dit is geen schermontwerp. Er is geen link naar het figma project in de documenten te vinden. Daarnaast zijn er geen afbeeldingen van de gemaakte schermontwerpen in het functioneel ontwerp (dus geen screenshots van de browser hier plaatse, maar echt schermontwerpen!). Er kunnen geen punten worden gegeven aan dit onderdeel. | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| Beoordelingscriteria 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. | |
| *Feedback docent poging 1:* Je klassendiagram is automatisch gegenereerd. Het is geeft geen duidelijke weergave van de applicatie structuur. Het is niet leesbaar. | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| Criterium 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. | |
| *Feedback docent poging 1:* Je diagrammer geven niet de architecturale structuur van je backend weer.. Je hoort te beginnen vanuit je gebruiker, niet vanuit je frontend home pagina | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* Je applicatie heet in je POM "les16jwt" en in je Main "les17Testen". Je structuur is slordig, je frontend broncode en documenten staat IN je map met backend broncode. Je gebruikt soms onzinnge benamingen zoals "uuid", terwijl het gewoon een String is. Je vacancy krijgt een status "open" of " close". Zet dit er niet als comment bij, maar forceer het, door bijvoorbeeld een enum te gebruiken. Je architecturale lagen zijn niet compleet. Waar is bijvoorbeeld je RoleService? Je spreekt nu direct de RoleRepository aan in de RoleController. | *3* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd, POM en Main, De code is opgeschoond, ongebruikte referenties zijn verwijderd, De UUID is verwijderd, de status is veranderd om ENUM te gebruiken. Er is nu geen toegang meer tot RoleRepository vanuit RoleController. De functionaliteit van Role zit nu in UserController. en de code volgt het MVC-ontwerppatroon met behulp van Java en Springboot-framework. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| De student gebruikt HTTP-methods om de vertaalslag te maken naar acties met de data. | |
| *Feedback docent poging 1:* Je upload werkt, maar je download niet. Het is heel leuk dat je frontend via je backend beschikbaar is, maar het is niet nodig. Je hebt te weinig functionaliteit. Ik mis met name nog veel CRUD functies. Je maakt netjes gebruik van DTO's en ook verschillende DTO's voor verschillende doelen. Je hebt alleen geen validatie op je DTO's | *4* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd en ook De download-CV-functie werkt nu. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* Je hebt loze endpoints in je security staan ("/static" en "/index.html"). Je gebruikt rollen, maar maakt uiteindelijk geen onderscheid. Je "preAuthorize" werkt niet, ik kan bijvoorbeeld gewoon een vacancy posten als karel. Je kunt het makkelijkst verschillende antmatchers in je securityfilterchain zetten. Je wachtwoord word netjes encrypted opgeslagen. | *4* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd ook Het autorisatieprobleem is opgelost met PreAuthorize-annotatie in Spingboot. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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). | |
| *Feedback docent poging 1:* Je hebt geen tests | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd Testcases zijn toegevoegd om de backend-API's en logica te dekken. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* Je project bevat de maven structuur. Je hebt 1 commit met een weinig betekenisvolle message en geen branches. | *4* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd, Meer commits and branches, Ik wil ook mededelen, Date er problemen met mijn computer is, Waardoor alles weg is, En dat dit 3x is gebeurd, Waardoor ik telkens steeds alles moet installleren en instellen. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| De student leest en bewerkt data met behulp van SQL, JPA en Hibernate en draagt zorg voor de autorisatie en authenticatie in de database. | |
| *Feedback docent poging 1: Je hebt niet alle entiteiten gevuld in je data.sql. Dat is jammer, want je hebt wel functionaliteit om 10, 20, of 30 vacatures weer te geven. Het is wel zo handig als je dan ook zoveel vacatures hebt, zodat je die functie in werking kan zien. Je hebt een aantal custom queries gemaakt.* | *4* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd meer entiteiten zijn toegevoegd. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* Binnen je project is het onduidelijk wat pagina's zijn en wat componenten zijn. Dit had je beter kunnen onderverdelen is respectievelijke folders binnen je project. Momenteel ontvangen weinig van jouw gemaakte JS bestanden callback properties of properties, hierdoor zijn de componenten niet herbruikbaar. Daarnaast zijn sommige componenten een beetje apart, bijvoorbeeld Contact.js heeft enkel een container erin zitten. Deze componenten hebben dus weinig nut. Daarnaast zijn er geen bouwstenen van een applicatie gemaakt, denk hierbij bijvoorbeeld aan herbruikbare componenten zoals buttons, input elementen of zelfs afbeeldingen en kaartjes! Hier ligt nog veel potentie om hier een verbeterslag in te slaan. | *2* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| De student voorziet componenten van passende interactie en externe data door correct gebruik te maken van alle React Life Cycle Hooks en AJAX. | |
| *Feedback docent poging 1:* Er zijn zes asynchrone functies gedeclareerd met ieder een callback property van params, echter zit er niks binnen deze asynchrone functies. Je manier van asynchrone functies schrijven doe je met .then maar je implementeerd geen .catch, het lijkt er op dat je oude en nieuwe schrijfwijze door elkaar heen gebruikt. Dit is niet de bedoeling. Tevens is er geen gebruik gemaakt van de life cycle methods, hierdoor kunnen er geen punten worden gegeven aan dit onderdeel. | *2* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* In het project wordt er geen gebruik gemakt van semantische elementen zoals article, sectipn, main, nav etc. Dit is een gemiste kans om juist semantische elementen in te zetten, met deze elementen bouw je een bepaalde structuur op van je applicatie of website, zodat deze ook toegankelijk is voor iederen. Het gebruik van semantische elementen is aan te raden, probeer niet enkel div-jes te gebruiken. Van alle aanwezig button elementen is er maar 1 voorzien van een type attribuut. Je hebt geen gebruik gemaakt van de aangeleerde mappenstructuur, en hebt alles enkel in de src folder geplaatst. Het is daardoor onduidelijk wat een component is, wat een pagina is, waar jouw context staat en waar alles onder elkaar valt. Kijk hier nog even goed naar en maak dit overzichtelijk voor zowel jezelf als voor medecollega's! | *4* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd en de frontend-code is gescheiden in modules om de code meer gestructureerd te maken, en er zijn enkele semantische syntaxis-problemen en try-catch-problemen in ajax-oproepen opgelost. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| De student schrijft schone, gestructureerde JavaScript code waarin op correcte wijze functies, arrays en objecten worden gebruikt en clean code principes worden toegepast. | |
| *Feedback docent poging 1:* De algemene kennis van Javascript en React is beide onvoldoende. In het project maak je gebruik van variabelen zoals var, gebruik je nog de class components opzet van React, terwijl de aangeleerde methode vanuit NOVI functionele componenten met hooks zijn. In moderne React projecten wordt er weinig gebruik gemaakt van class components, sinds de overstap naar functional. Daarnaast heb je geen gebruik gemaakt van Life Cycle Methods, waardoor jouw applicatie bij iedere toets aanslag gaat renderen, dit vergt enorm veel geheugen voor applicaties en is bad practice. Er ligt voor zowel Javascript als React veel kans om een verbeterslag te treffen in de code. Daarnaast staat er in het document nog vrij veel console.log's die bij developen handig zijn, maar uiteindelijk mogen worden verwijderd, en zit er hier en daar nog wat stukjes uitgecommente code tussen of code die niet gebruikt wordt. | *3* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd en Alle console.log zijn verwijderd, evenals alle gecommentarieerde code. De code is opgeschoond, ongebruikte referenties zijn verwijderd, en de code volgt het MVC-ontwerppatroon met behulp van Java en Springboot-framework. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1: Binnen je project is het onduidelijk wat pagina's zijn en wat componenten zijn. Dit had je beter kunnen onderverdelen is respectievelijke folders binnen je project. Momenteel ontvangen weinig van jouw gemaakte JS bestanden callback properties of properties, hierdoor zijn de componenten niet herbruikbaar. Daarnaast zijn sommige componenten een beetje apart, bijvoorbeeld Contact.js heeft enkel een container erin zitten. Deze componenten hebben dus weinig nut. Daarnaast zijn er geen bouwstenen van een applicatie gemaakt, denk hierbij bijvoorbeeld aan herbruikbare componenten zoals buttons, input elementen of zelfs afbeeldingen en kaartjes! Hier ligt nog veel potentie om hier een verbeterslag in te slaan.* | *2* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd en het project is opgedeeld per scherm. | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1:* *Het registreren van een gebruiker werkt naar behoren. Wanneer de gebruiker ingelogd is komt deze op een lege pagina terecht en is het vrij onduidelijk wat de appllicatie doet en wat een gebruiker kan doen. Daarnaast kan je als gebruiken je CV uploaden, dit werkt prima en komt ook onder diens profiel te staan. Als admin kan je vacatures plaatsen, en deze kan je als gebruiker inzien. Dit werkt verder allemaal prima, tevens het verwijderen van een applicatie. Echter voelt de applicatie in diens functionaliteiten wel wat summier aan, het zijn kleine functionaliteiten. Bijvoorbeeld een gebruikersprofiel aanpassen, en een foto te kunnen uploaden had al een grotere en interessantere functionaliteit kunnen zijn om te plaatsen. Daarnaast kon de student ook ervoor zorgen dat de gebruiker niet naar een kale pagina zit te kijken, door middel van styling.* | *6* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: | |
| Toelichting verbeteringen student poging 2: | |

|  |  |
| --- | --- |
| 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. | |
| *Feedback docent poging 1: Je hebt een document ingeleverd met de naam installatiehandleiding, maar de onderdelen die erin horen te zitten, zitten er niet in. Met de huidige installatiehandleiding kan een first-time-gebruiker de applicatie niet opstarten. Binnen de installatiehandleiding is er geen sprake van een duidelijke stappenplan, lijst met benodigheden. De huidige installatiehandleiding is te kort, niet duidelijk en voldoet niet aan de kwaliteitseisen van de opdracht.* | *2* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: Verbeterd Uitgebereid installatiehandleiding. | |
| Toelichting verbeteringen student poging 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. | |
| *Feedback docent poging 1: Je hebt een document ingeleverd met de naam verantwoordingsdocument, maar de onderdelen die erin horen te zitten, zitten er niet in. Je hebt geen 10 technische keuze's beargumenteerd en eventuele doorontwikkeligen beargumenteerd. Er kunnen geen punten worden gegeven aan dit onderdeel.* | *1* |
| *Feedback docent poging 2:* | *[cijfer]* |
| Toelichting verbeteringen student poging 1: 10 technische keuzes zijn toegevoegd en beargumenteerd. | |
| Toelichting verbeteringen student poging 2: | |