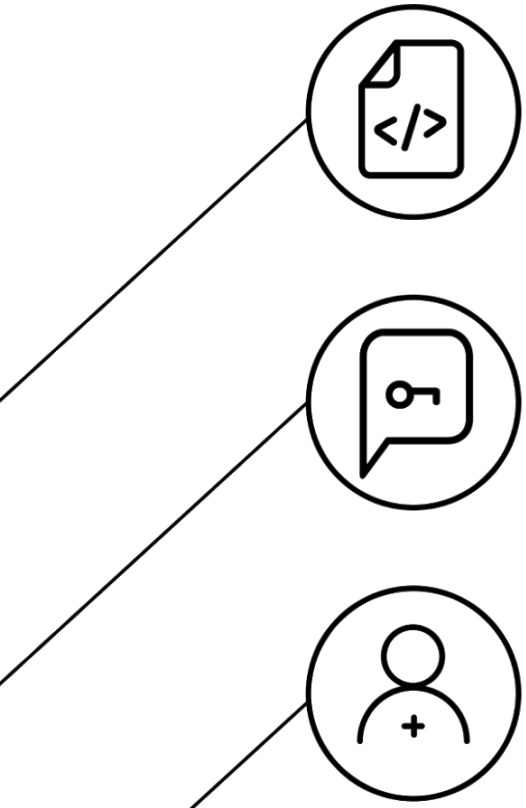

Eindopdracht

Hbo-bachelor Software Development

Leerlijn Frontend Programming

Opdrachtbeschrijving en deelopdrachten

Randvoorwaarden, structuur en beoordeling

NOVI HOGESCHOOL

Inhoud

Integrale eindopdracht Front End Programming (30 EC)

De leerlijn Front End Programming bevat de cursussen HTML & CSS, JavaScript, React. Om deze leerlijn af te ronden dien je de volgende leeruitkomsten aan te tonen:

1. **HTML & CSS**
   De student bouwt statische webpagina's met HTML, voorziet deze van styling en layout door CSS en ontwerpt een prototype in Adobe XD waarmee hij een visueel design omzet naar een webpagina. (LU1)
2. **JavaScript**
   De student schrijft en test schone, gestructureerde JavaScript code waarmee hij webpagina's voorziet van interactie en gebruikt hierbij een API. (LU2)
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. (LU3)

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. Bij frontend development ligt de focus op wat gebruikers visueel te zien krijgen in hun browser. Je ontwikkelt het uiterlijk van een webpagina en bent daarmee verantwoordelijk voor het creëren van de gebruikerservaring van het product.

Opdracht: je gaat een applicatie bedenken en bouwen waarvan je alleen de frontend gaat programmeren. Je bedenkt welk probleem je wil oplossen met jouw product en aan welke eisen de applicatie moet voldoen. Hiervoor ga je de applicatie eerst uittekenen door schermontwerpen te maken. Daarna ga je de frontend code schrijven.

Door deze opdracht uit te voeren toon je aan een volwaardige webapplicatie te kunnen bouwen in de frontend. In het volgende hoofdstuk vind je meer informatie over de deelopdrachten.

Om de leerlijn Front End Programming met succes af te ronden is een voldoende nodig voor de integrale toets; met de deelopdrachten kunnen geen losse cursussen worden afgerond.

Op te leveren producten:

- Functioneel ontwerp met daarin o.a. de probleembeschrijving, functionele- en niet-functionele-eisen, use cases, geschetste wireframes en prototype;
- De broncode van een React webapplicatie;
- Zelfgeschreven installatiehandleiding voor beide applicaties als README.md;
- Verantwoordingsdocument.

Deelopdrachten

We gaan jouw vaardigheden als frontend ontwikkelaar toetsen door een applicatie te bouwen. In een frontend applicatie kun je geen data opslaan, maar omdat we je toch willen laten oefenen met het bouwen van een applicatie met inlog-functionaliteit, stellen we daarvoor een NOVI-backend beschikbaar waarin je nieuwe gebruikers kunt toevoegen en bestaande gebruikers kunt valideren. Indien de functionaliteit van de NOVI-backend niet toereikend is voor jouw idee zul je zelf een backend moeten maken. Hier zijn wel voorwaarden aan verbonden (zie 'Randvoorwaarden').

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.

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. Verwerk in deze beschrijving ook welke API je gaat gebruiken en of je de NOVI-backend gaat gebruiken voor het inloggen en registreren of dat je hier zelf iets voor bedenkt. 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 onderstaande casus gebruiken voor jouw eindopdracht.

Voorbeeldcasus

Het onderstaande voorbeeld beschrijft het probleem wat de fictieve student zou willen oplossen en een globale beschrijving van hoe hij dat zou willen doen.

Mijn grootste hobby is Netflixen. Het aanbod van Netflix is locatiespecifiek, de ene film kan in Brazilië wel beschikbaar zijn en in Nederland niet. Als ik op vakantie ga (en dat ga ik nogal vaak) weet ik nooit welke films en series Netflix beschikbaar heeft op mijn vakantiebestemming. Daarom ga ik een applicatie programmeren waarin ik naar films en series kan zoeken. De applicatie geeft dan weer of de content überhaupt beschikbaar is op Netflix en zo ja, in welke landen. Ook wil ik zien welke Nederlandse content de komende maand van Netflix afgehaald zal worden en welke nieuwe content er verwacht wordt. Zo hoef ik nooit meer iets te missen. Ik zorg ervoor dat gebruikers kunnen inloggen en zo kunnen opslaan in welk land ze wonen, zodat de nieuwe- en verwijderde content op deze locatie is toegespitst. Om dit te doen maak ik HTTP requests naar de *unogsNG API* (https://english.api.rakuten.net/unogs/api/unogsng/endpoints) om zo de juiste data op te vragen en te verwerken in mijn applicatie.

Kijk ook eens op https://apilist.fun/ om je te oriënteren op andere API's die je kunt gebruiken voor jouw project.

Deelopdracht 1. Functioneel ontwerp

Wanneer je een webapplicatie gaat bouwen kun je niet zomaar beginnen met programmeren. Het is belangrijk eerst te inventariseren welk probleem het product oplost en welke functionele en niet-functionele eisen hieraan verbonden zijn. Vervolgens kun je visuele inspiratie op gaan doen voor het uiterlijk van het eindproduct en ruwe schetsen maken de indeling van de belangrijkste pagina's (*wireframes*).

Je gaat een functioneel ontwerp maken 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 cases.
- Bevat de gescande (of gefotografeerde) wireframes, geschetst op papier, van tenminste vijf pagina's.

Op basis van deze informatie ga je schermontwerpen maken in Adobe XD. Deze voeg je samen tot een functioneel prototype.

Op te leveren:

- Het functioneel ontwerp van de webapplicatie in PDF (.pdf) of mark down (.md).
- Het Adobe XD prototype in .xd formaat

Deelopdracht 2. Verantwoordingsdocument

Zodra je begint met programmeren, ga je ook beginnen aan het verantwoordingsdocument. Hierin leg je vast welke 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 ervoor gekozen om CSS Grid te gebruiken voor je pagina layout? Waarom heb je deze specifieke npm package gebruikt en niet een andere? 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 ook op je eigen leerproces: wat ging goed en wat kan de volgende keer beter?

Op te leveren:

- Verantwoordingsdocument in PDF (.pdf) of mark down (.md) met daarin:
    - Minimaal 10 beargumenteerde keuzes
    - Een beschrijving van limitaties aan de applicatie en mogelijke doorontwikkelingen

Deelopdracht 3. Broncode React

In de eerste opdracht heb je uitgewerkt wat de applicatie moet kunnen en welk probleem het gaat oplossen. In deze opdracht ga je jouw prototype als basis gebruiken om de webapplicatie te implementeren in React. Jouw webapplicatie heeft minimaal de volgende eigenschappen:

- Een gedeelte van de content is alleen beschikbaar voor geregistreerde gebruikers (zoals bijvoorbeeld een profiel-pagina). Gebruikers kunnen zich aanmelden en vervolgens inloggen met hun gebruikersaccount.
- Er wordt gebruik gemaakt van externe data om zo, naast eigen content, ook andere informatie te kunnen gebruiken. Deze data moet worden opgehaald middels een API. Voorbeelden hiervan zijn weersvoorspelling data, de IMDB-database, Google maps, een receptendatabase, de data van Sky Scanner, etc.
- Er wordt gebruik gemaakt van routing waardoor gebruikers naar verschillende webpagina's in de applicatie kunnen navigeren.
- 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

Op te leveren:

- Projectmap met daarin de broncode, inclusief de link naar de Github repository.

Deelopdracht 4. Installatiehandleiding

In de voorgaande opdrachten heb je jouw ontwikkelwerk afgerond. Om ervoor te zorgen dat ook andere ontwikkelaars jouw project kunnen gebruiken, is het belangrijk een installatiehandleiding te schrijven waarin beschreven wordt wat zij hiervoor nodig hebben.

Het bevat:

- Een inleiding met korte beschrijving van de functionaliteit van de applicatie en screenshot van de belangrijkste pagina van de applicatie.
- Lijst van benodigdheden om de applicatie te kunnen runnen (zoals bijvoorbeeld een API key of gegevens van een externe backend).
- Een stappenplan met daarin installatie instructies.
- Met welke gegevens er ingelogd kan worden wanneer er al accounts beschikbaar zijn.
- Welke andere npm commando's er nog beschikbaar zijn in deze applicatie.

Op te leveren:

- Zelfgeschreven README.md in de root van de React projectmap

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 (geen TypeScript).
- Er mag geen Redux gebruikt worden.
- Het project is geüpload naar een GitHub repository: deze repository staat op *public*.
- Het project wordt ingeleverd *zonder* node_modules-map/.idea-map.
- Het project bevat de JavaScript linter ESLint.
- De wireframes zijn getekend op papier.
- Het prototype is ontworpen met Adobe XD. (Let op: Adobe XD werkt niet op Linux-systemen. Indien je op een Linux-systeem werkt, mag je zelf een **gratis** prototyping tool uitkiezen die wel geschikt is voor Linux.)
- De applicatie start op zonder crashes.
- Er is géén gebruik gemaakt van out-of-the-box styling systemen zoals Bootstrap, Material-UI of Tailwind.
- Wanneer er gebruik gemaakt wordt van een eigen backend (dit is optioneel), mag dit alléén:
  - een Firebase backend zijn en dienen de inloggegevens ook te worden verstrekt aan de docent
  - óf de backend zijn die eerder is gemaakt in de Backend-leerlijn. Deze moet dan opnieuw worden ingeleverd samen met deze opdracht.
- Het wordt aangeleverd d.m.v. een ZIP-bestand van maximaal 50 MB.

Structuur

Functioneel ontwerp
Het functioneel ontwerp is een verzorgd document met een titelblad, inhoudsopgave en inleiding. Het verantwoordingsdocument mag toegevoegd worden als laatste hoofdstuk in het functioneel ontwerp.
Het bevat geen verwijzingen naar afbeeldingen of documenten buiten het document zelf. Er is geen harde richtlijn met betrekking tot het aantal woorden, maar zorg ervoor dat je beknopt en bondig schrijft.

Installatiehandleidingen
De installatiehandleiding is een verzorgd document met een inhoudsopgave en inleiding.

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.* Functioneel ontwerp**	***Met deze opdracht worden aspecten van de leeruitkomst van de cursus HTML & CSS (LU1) getoetst.***	**30%**		
Criterium 1.1	De student schrijft een compleet functioneel ontwerp, inclusief: heldere omschrijving probleem én oplossing, minimaal 50 passende functionele en niet-functionele eisen, minimaal 4 kwalitatieve use-cases, verzameling van minimaal 3 verschillende inspiratiebronnen en ten minste 5 duidelijk vormgegeven wireframes. (LU1)	20%		
Criterium 1.2	De student ontwerpt een prototype met Adobe XD door middel van schermontwerpen. (LU1)	10%		
Feedback door de docent				
2. Verantwoordings document	***Met deze opdracht worden aspecten van de leeruitkomsten van de cursussen JavaScript (LU2) en React (LU3) getoetst.***	**10%**		
	De student reflecteert op zijn (schone en gestructureerde) JavaScript/React code en beargumenteert minimaal 10 gemaakte keuzes. (LU2, LU3)	5%		
	De student beschrijft realistische limitaties van zijn applicatie en beschrijft tevens beargumenteerd welke doorontwikkelingen mogelijk en/of wenselijk zijn. (LU2 en LU3)	5%		
Feedback door de docent				
3. Broncode React	***Met deze opdracht worden aspecten van de leeruitkomsten van de cursussen HTML & CSS (LU1), JavaScript (LU2) en React (LU3) getoetst.***	**50%**		
	De student bouwt zijn applicatie, op basis van het functioneel ontwerp en het prototype, met semantische HTML-elementen en maakt gebruik van zelfgeschreven, schone, gestructureerde CSS door	5%		

	modulaire styling aan specifieke componenten te koppelen. (LU1, LU2 en LU3)			
	De student schrijft schone, gestructureerde JavaScript en React code waarin hij op correcte wijze functies, variabelen, arrays en objecten (OOP) gebruikt en clean code principes toepast. (LU2 en LU3)	10%		
	De student werkt op een juiste manier met asynchrone functies om externe data op te halen via een API en handelt Promises af. (LU2)	5%		
	De student installeert relevante NPM packages als dependency, om zijn code uit te breiden. (LU2)	5%		
	De student test zijn code met zelfgeschreven unit tests. (LU2 en LU3)	5%		
	De student deelt zijn applicatie op logische wijze op in herbruikbare componenten waarbij data op juiste wijze wordt doorgegeven via properties. (LU3)	10%		
	De student voorziet componenten van interactie en externe data door correct gebruik te maken van de React Life Cycle hooks. (LU3)	10%		
Feedback door de docent				
Feedback door de docent				
4. Installatiehandleiding	***Met deze opdracht worden aspecten van de leeruitkomst van de cursus React (LU3) getoetst.***	**10%**		
	De student schrijft met gebruik van de clean code principes een duidelijke installatiehandleiding waarmee iemand zonder kennis van het project de gebouwde applicatie zelfstandig kan draaien (LU3)	10%		
Feedback door de docent				
	Totaal	**100%**		