Eindopdracht

Leerlijn Frontend (30 EC)

Inhoudsopgave

Eindopdracht Frontend

Integrale eindopdracht

De leerlijn Frontend bevat de cursussen HTML & CSS, JavaScript en 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 schermontwerpen in een design tool (zoals Figma of Adobe XD) waarmee een visueel design wordt omgezet naar een webpagina. (LU1)
2. **JavaScript**
   De student schrijft schone, gestructureerde JavaScript code waarmee webpagina's worden voorzien van interactie en data wordt opgehaald via 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 eerst een plan schrijven en schermontwerpen 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 Frontend met succes af te ronden is een voldoende nodig voor de integrale eindopdracht; 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 case tabellen, geschetste wireframes en schermontwerpen.
- De broncode van een React webapplicatie.
- Zelfgeschreven installatiehandleiding als README.md.
- Verantwoordingsdocument.

Deze producten worden ingeleverd als één ZIP-bestand.

Deelopdrachten

We gaan jouw vaardigheden als frontend ontwikkelaar toetsen door een applicatie te bouwen die gebruik maakt van externe data. Daarnaast zul je authenticatie implementeren door gebruik te maken van een NOVI-backend waarin je nieuwe gebruikers kunt toevoegen en bestaande gebruikers kunt valideren. Indien de functionaliteit van de NOVI-backend niet toereikend is voor jouw idee mag je in sommige gevallen zelf een backend 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.

Voordat je kunt starten met de eerste deelopdracht lever je eerst jouw casus in ter goedkeuring bij de docent. Je mag ervoor kiezen om de wensen van de voorbeeldcasus te vertalen naar een idee voor jouw webapplicatie, of om een eigen casus te bedenken. In dat geval ga je eerst op zoek naar een passende API en bedenk je hoe jij deze data kunt gebruiken binnen jouw casus. We raden het af om gebruik te maken van kaarten of GPS-gerelateerde onderwerpen, gezien dit veel complexiteit met zich meebrengt.

Je beschrijft in maximaal 250 woorden:

- Welk probleem je wil oplossen met deze applicatie;
- Welke API je hiervoor gaat gebruiken (inclusief link naar de documentatie);
- Wat de 4 belangrijkste functionaliteiten van jouw applicatie zullen zijn. Hierin is één functionaliteit al vergeven aan registreren en inloggen;
- Of je de NOVI-backend zult gebruiken voor registratie en inlog of, indien je een andere backend wil gebruiken, welke dit zal zijn.

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.

Voorbeeldcasus

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

Weet jij ook nooit wat je vanavond moet koken? Lijkt het alsof jouw creativiteit en besluitvaardigheid rond etenstijd ook ineens verdwenen is? Persoonlijk ervaar ik dit ongeveer iedere dag. Daarom ga ik een applicatie programmeren waar ik een aantal grappige vragen kan beantwoorden over o.a. mijn stemming, eetgezelschap en motivatie-om-te-koken-niveau, waarna deze applicatie een aantal passende recepten voorstelt om uit te kiezen. Wanneer iemand een rotdag heeft gehad, zal de applicatie bijvoorbeeld meer comfort-food recepten voorstellen. Wanneer iemand geen zin heeft om te koken, worden er snelle en makkelijke recepten getoond. Op de dagen dat de gebruiker zich wel besluitvaardig voelt, is er ook een mogelijkheid om door alle beschikbare recepten te browsen en op zoek te gaan naar specifieke recepten met behulp van een zoekfunctie. Ten slotte heeft de gebruiker ook de mogelijkheid om recepten te bekijken op basis van de ingrediënten die nog in de koelkast liggen. Om dit te doen maak gebruik ik de *Edamam API* (https://www.edamam.com/) om zo recepten op te vragen en te verwerken in mijn applicatie.

Wil je liever een eigen casus bedenken? Kijk eens op https://github.com/hogeschoolnovi/frontend-api-list om je te oriënteren op mogelijke API's die je kunt gebruiken voor een eigen casus. Deze lijst is door NOVI samengesteld en gecontroleerd. Wanneer je een andere API kiest, is dit op eigen risico.

Deelopdracht 1. Functioneel ontwerp

Wanneer je een webapplicatie gaat bouwen kun je niet zomaar beginnen met programmeren. Het is belangrijk eerst te inventariseren hoe het product moet werken (door use case tabellen op te stellen van de belangrijkste kernfunctionaliteiten van de applicatie) en te bepalen 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*). Ten slotte vertaal je deze naar uitgewerkte digitale schermontwerpen. Wanneer je jouw uitgewerkte functioneel ontwerp aan een ontwikkelaar zou geven, moet deze direct begrijpen wat er precies ontwikkeld moet worden.

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.
- Beschrijf het probleem en de manier waarop deze applicatie dat probleem oplost.
- Beschrijf de belangrijkste gebruikershandelingen in minimaal vier use case tabellen: één voor authenticatie (registreren *of* inloggen) en drie voor zelfbedachte functionaliteit. Dit dienen de kernfunctionaliteiten van de applicatie te zijn, in het format zoals je hebt geleerd tijdens de leerlijn. Ieder main success scenario wordt gecomplementeerd met 1 tot 3 alternatieve scenario's.
- 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 ter inspiratie voor de visuele stijl van de applicatie, inclusief toelichting waarin je uitlegt waarom dit voor jou een inspiratiebron is en wat je ervan gaat gebruiken in je eigen ontwerp.
- 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 ten minste 5 schermontwerpen met titels en beschrijvingen. Deze bevatten dusdanig veel uitgewerkte details zodat je tijdens het programmeren geen ontwerpbeslissingen meer hoeft te maken. Aanvullend lever je ook de link naar (of bestanden van) het originele project in Figma/Adobe XD aan.

Let op! Voldoe je niet aan de minimale eisen per onderdeel? Dan ontvang je 0 punten.
Lees zorgvuldig nog de beoordelingscriteria door op pagina 13.

Op te leveren:

- Het functioneel ontwerp van de webapplicatie in PDF (.pdf).
- Link naar het openbare project (in Figma of Adobe XD) met jouw 5 schermontwerpen.

Deelopdracht 2. Verantwoordingsdocument

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. Het gaat hierbij alleen om de technische programmeerkeuzes betreffende JavaScript en React en dus niet om esthetische keuzes. Je begint al tijdens het ontwikkelen van jouw product met het schrijven van het verantwoordingsdocument. Hierin geef je antwoord op vragen als: Waarom heb je ervoor gekozen om een component te bouwen voor deze HTML-elementen? Was deze implementatie de enige optie, of heb je ook andere alternatieven overwogen? Waarom heb je ervoor gekozen om sommige acties te implementeren als helperfuncties? 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 omdat dit toch ingewikkelder bleek dan je had verwacht? Leg dan uit wat je zeker nog zou implementeren 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.

Op te leveren:

- Verantwoordingsdocument in pdf (.pdf) met daarin:
  - Minimaal 5 beargumenteerde technische programmeerkeuzes betreffende JavaScript en React, inclusief reflectie.
  - Een beschrijving van minimaal 5 limitaties van de applicatie en argumentatie van mogelijke doorontwikkelingen. Het gaat hierbij om limitaties betreffende de functionaliteit van de applicatie, niet om de styling.
- Indien je een herkansing inlevert na het krijgen van feedback, lever je ook het ingevulde 'Template herkansingsfeedback' in. Dit Worddocument kun je vinden in Teams.

Let op! Voldoe je niet aan de minimale eisen per onderdeel? Dan ontvang je 0 punten.
Lees zorgvuldig nog de beoordelingscriteria door op pagina 13.

Deelopdracht 3. Broncode React

In de eerste opdracht heb je uitgewerkt wat de applicatie moet kunnen en hoe deze eruit moet zien. In deze opdracht ga je jouw schermontwerpen 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 ingelogde gebruikers (bijvoorbeeld een profielpagina). Gebruikers kunnen zich zowel registreren als inloggen. Hiervoor wordt gebruikgemaakt van React Context en de NOVI-backend*. De documentatie voor het gebruik van deze backend vind je hier.
- Er wordt gebruikgemaakt van externe data om zo, naast eigen content, ook andere informatie te kunnen gebruiken. Deze data moet worden opgehaald door middel van **netwerk requests naar een API****. Voorbeelden hiervan zijn weersvoorspelling data, de IMDB-database, een receptendatabase, de Fake Store API, etc.
- De applicatie ondersteunt, naast inloggen/registreren, nog 3 belangrijke kernfunctionaliteiten die je hebt bedacht rondom het gebruik van de API. Bijvoorbeeld:
    - Kernfunctionaliteit 1: Als gebruiker kun je je registreren en inloggen.
    - Kernfunctionaliteit 2: Als ingelogde gebruiker kun je aangeven naar wat voor weer je opzoekt bent, uitgedrukt in bijvoorbeeld: temperatuur, zon (in UV), windrichting, windkracht, regen.
    - Kernfunctionaliteit 3: Als ingelogde gebruiker …
    - Kernfunctionaliteit 4: Als ingelogde gebruiker …
- Er wordt gebruikgemaakt van routing waardoor gebruikers naar verschillende webpagina's in de applicatie kunnen navigeren.
- Je voorziet de applicatie van globale en lokale opmaak en maakt layouts met behulp van CSS Flexbox. Je schrijft je styling zelf en maakt géén gebruik van out of the box styling systemen zoals Bootstrap, Material-UI of Tailwind. Het gebruik van iconpacks zoals Fontawesome is wel toegestaan.
- Je maakt gebruik van Git om jouw project te beheren en zet daarom jouw project zo snel mogelijk op GitHub. Je zorgt voor kleine beschrijvende commits, maakt pull requests voor iedere nieuwe feature en mergt regelmatig naar de main branch.

Let op! Voldoe je niet aan de minimale eisen per onderdeel? Dan ontvang je 0 punten.
Lees zorgvuldig nog de beoordelingscriteria door op pagina 13.

* Tenzij je een andere backend gebruikt. Indien hiervoor gekozen wordt, mag dit alleen: een Firebase backend zijn óf de backend zijn die eerder is gemaakt in de Backend leerlijn. Deze moet dan opnieuw worden ingeleverd samen met deze opdracht. Hiervoor zullen geen extra punten worden toegekend.
** Gebruik je een eigen backend? Ook dan maak je daarnaast nog gebruik van een externe API.

Op te leveren:

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

Deelopdracht 4. Installatiehandleiding

In de voorgaande opdrachten heb je je 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 ze hiervoor nodig hebben. Je schrijft je installatiehandleiding voor een mede-developer, maar zorgt ervoor dat dit ook te volgen is wanneer deze persoon geen enkele ervaring heeft binnen het frontend-landschap.

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 runtime environments, een API key of gegevens van een externe backend). Let op: je vraagt de nakijkende docent nooit zelf een API key aan te maken. Jij levert zelf je API key aan in de handleiding;
- Een stappenplan met daarin installatie-instructies.
- Met welke gegevens er ingelogd kan worden indien er al accounts beschikbaar zijn.
- Welke andere npm commando's er nog beschikbaar zijn in deze applicatie en waar deze voor dienen.

Let op! Voldoe je niet aan de minimale eisen per onderdeel? Dan ontvang je 0 punten.
Lees zorgvuldig nog de beoordelingscriteria door op pagina 13.

Op te leveren:

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

Quickscan

Hieronder vind je een aantal randvoorwaarden waaraan de eindopdracht moet voldoen. De beoordelaar kan op basis van deze eisen de eindopdracht teruggeven en zal deze niet verder nakijken tot aan de eisen zijn voldaan.

Gebruik de quickscan om te zien of je voldoet aan de inlevereisen.

Algemene eisen:

- ☐ Documentatie ingeleverd als .pdf.
- ☐ Documentatie bevat geen bronnen of verwijzingen buiten het document (linkjes etc. behalve die naar GitHub en eventueel Figma)
- ☐ De eindopdracht is goed leesbaar zonder storende aanwezigheid van grammatica- en spellingsfouten.
- ☐ Het volledige project en bijbehorende documenten wordt aangeleverd d.m.v. één ZIP-bestand van maximaal 50 MB (geen .rar!).
- ☐ Het ZIP-bestand bevat de volgende elementen:
    - Functioneel ontwerp (in .pdf)
    - Broncode van jouw project met daarin de installatiehandleiding als README.md (Let op! Dus niet alleen de links naar het GitHub-project)
    - Bronbestanden van de schermontwerpen
    - Verantwoordingsdocument (in .pdf)
- ☐ Indien je een herkansing inlevert na het krijgen van feedback, is het ingevulde 'template herkansingsfeedback' bijgevoegd.

Inhoudelijke eisen:

- ☐ Alle deelopdrachten zijn uitgewerkt en de gevraagde deelproducten zijn aanwezig.
- ☐ Het project is geüpload naar een GitHub repository: deze repository staat op public.
- ☐ De applicatie is geprogrammeerd met JavaScript en React (geen TypeScript) en er is gebruik gemaakt van React Context (geen Redux).
- ☐ Er is GEEN gebruik gemaakt van out-of-the box styling systemen zoals Bootstrap, Material-UI of Tailwind.
- ☐ De broncode wordt ingeleverd zonder 'node_modules'-map en '.idea'-map
- ☐ De wireframes zijn getekend op papier.
- ☐ De schermontwerpen zijn ontworpen met een design tool zoals Figma of Adobe XD.
- ☐ Er wordt gebruikgemaakt van een externe API en een backend voor het inloggen en registreren van gebruikers (NOVI backend of backend uit eerdere leerlijn of Firebase).
- ☐ De applicatie start op zonder te crashen.

Structuur

De integrale eindopdracht bestaat uit verschillende documenten. Houd hierin de volgende structuur aan:

Functioneel ontwerp

- **Inleiding**
  De inleiding fungeert net zoals de inhoudsopgave als een routekaart voor de lezer: hier leg je uit wat de lezer kan verwachten van jouw verslag.
- **Algemene omschrijving applicatie**
  Voordat je direct de diepte in duikt, leg je globaal uit welk probleem je hebt willen oplossen met jouw applicatie en wat de belangrijkste kernfunctionaliteiten zijn.
- **Use case tabellen**
  Iedere use case tabel bestaat uit een main success scenario en 1 tot 3 alternatieve scenario's.
- **Functionele en niet-functionele eisen**
  Eisen zijn beschreven in het format "Als [rol] kan/wil/doe/gebruik ik…" en voorzien van nummering en duidelijke categorisering.
- **Wireframes**
  Ingevoegde schetsen zijn voorzien van paginatitels en beschrijving.
- **Inspiratiebronnen**
  Ingevoegde screenshots zijn voorzien van toelichting.
- **Schermontwerpen**
  Ingevoegde screenshots zijn voorzien van paginatitels en beschrijving.

Verantwoordingsdocument

- Inleiding
- Verantwoording keuzes
- Mogelijke doorontwikkelingen
- Template herkansingsfeedback*

Beide documenten zijn netjes, verzorgd en voorzien van titelblad en inhoudsopgave.

Installatiehandleiding

1. Inleiding
2. Screenshot
3. Benodigdheden
4. De applicatie draaien
5. Overige commando's
6. Testgebruikers**

De installatiehandleiding is een nette, verzorgde README.md die voorzien is van inhoudsopgave en formatting in MarkDown.

* Alleen bij herkansing na feedback.
** Indien relevant.

Beoordelingscriteria

De eindopdracht wordt beoordeeld op basis van de volgende beoordelingscriteria. Per criterium kent de beoordelaar een aantal punten toe.

Let op! Indien je niet voldoet aan de minimale eis van een beoordelingscriterium, ontvang je 0 punten voor dat onderdeel.

Deelopdrachten/ producten		***Weging***
1. Functioneel ontwerp	**Met deze opdracht worden aspecten van de leeruitkomst van de cursus HTML & CSS (LU1) getoetst.**	**20 %**
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 case tabellen en een verzameling van minimaal 3 verschillende inspiratiebronnen. (LU1)	15%
Criterium 1.2	De student tekent minimaal 5 unieke en duidelijke weergegeven wireframes, die de basis vormen voor ten minste 5 kwalitatieve schermontwerpen, geïnspireerd op de inspiratiebronnen, die voldoende details bevatten om tijdens het programmeren geen designkeuzes te hoeven maken. (LU1)	5 %
2. Verantwoordings-document	**Met deze opdracht worden aspecten van de leeruitkomsten van de cursussen JavaScript (LU2) en React (LU3) getoetst.**	**10%**
Criterium 2.1	De student beargumenteert minimaal 5 gemaakte technische keuzes (JavaScript/React) en reflecteert op ieder punt. (LU2, LU3)	5%
Criterium 2.2	De student beschrijft minimaal 5 realistische limitaties van de functionaliteit van de applicatie en beargumenteert welke doorontwikkelingen mogelijk en/of wenselijk zijn. (LU2 en LU3)	5%
3. Broncode React	**Met deze opdracht worden aspecten van de leeruitkomsten van de cursussen JavaScript (LU2) en React (LU3) getoetst.**	**65%**

Criterium 3.1	De student bouwt de applicatie, op basis van het functioneel ontwerp en de schermontwerpen op, met semantische HTML-elementen met de juiste attributen en logische structuur. (LU1, LU2, en LU3)	5%
Criterium 3.2	De student bouwt een responsieve applicatie met gestructureerde CSS door modulaire styling aan specifieke componenten te koppelen en Flexbox op de juiste manier toe te passen. (LU1, LU2 en LU3)	5%
Criterium 3.3	De student schrijft schone, gestructureerde JavaScript en React code waarin op correcte wijze functies, variabelen, arrays en objecten worden gebruikt en clean code principes worden toegepast. (LU2 en LU3)	10%
Criterium 3.4	De student implementeert op een juiste manier 6 asynchrone functies om externe data op te halen via API, en handelt errors en laadtijd af door deze te communiceren in de UI. (LU2)	5%
Criterium 3.5	De student implementeert vier belangrijke kern-functionaliteiten op correcte wijze, waaronder het authenticatie-proces van registratie/inloggen en drie zelfbedachte, complete use cases.	10%
Criterium 3.6	De student deelt de applicatie in logische wijze op, in minimaal 6 herbruikbare componenten waarbij data op juiste wijze wordt doorgegeven via properties en callback properties (LU3)	10%
Criterium 3.7	De student voorziet de applicatie van interactie en externe data door correct gebruik te maken van statemanagement en de verschillende React Life Cycle Hooks. (LU3)	10%
Criterium 3.8	De student past routing op de juiste manier toe en maakt gebruik van dynamic routing en private routes. (LU3)	5%
Criterium 3.9	De student beheert de code met correct van GIT, maakt hierbij gebruik van minimaal 20 kleine commits, met compacte zinvolle commits messages en maakt minimaal 5 pull requests die naar de main branch gemerged zijn.	5%
4. Installatiehandleiding	**Met deze opdracht worden aspecten van de leeruitkomst van de cursus React (LU3) getoetst.**	**5%**
Criterium 4.1	De student schrijft op professionele wijze een installatiehandleiding waarmee iemand zonder kennis	5%

	van het project de gemaakte applicatie zelfstandig kan draaien. (LU3)	
Totaal		100%