Reflectie verslag

Project	:	The KEMBIT Times, nieuws
Opdrachtgever	:	Mike Wijnen
Auteur	:	Sandrine Prumpeler
Datum	:	15/05/’25
Versie	:	1.0

Inhoudsopgave

Inleiding

Kembit

KEMBIT is een full-range IT -bedrijf dat zich specialiseert in oplossingen voor Healthcare, Life Sciences en Corporate. KEMBIT is gevestigd op meerdere locaties: de hoofdlocatie te Kasteel Wijnandsrade, de High Tech Campus te Eindhoven en de Brightlands Chemelot Campus te Geleen. KEMBIT telt op dit moment ongeveer 140 medewerkers, die verspreid over de eigen locaties alsmede de klantlocaties werkzaam zijn.

Gestart als een detacheringsbedrijf pur sang, is KEMBIT zich in de loop der jaren gaan toeleggen op een steeds breder palet aan IT- en businessoplossingen voor een steeds specifiekere doelgroep. Met een sterke focus op kennis, kwaliteit en klanttevredenheid levert KEMBIT applicaties-, consultancy- en services diensten op het gebied van Werkplek, Applicaties, Security en Data. Van A tot Z bedenkt, bouwt en beheert KEMBIT duurzame IT-oplossingen die passen binnen de huidige omgeving van haar klanten en daarnaast bijdragen aan een toekomstbestendige infrastructuur.
KEMBIT kijkt niet alleen vanuit technisch perspectief naar organisaties, maar ook vanuit organisatorisch, beleidsmatig en menselijk perspectief. Daarmee positioneert KEMBIT zich niet alleen als een IT-leverancier, maar meer als een IT-partner voor haar klanten.

Het project

Onze opdracht was het ontwikkelen van een nieuws-site. Deze site heeft de naam KEMBIT Times gekregen. Het doel van de site is om gebruikers continue te voorzien van nieuwsberichten. Deze nieuwsberichten worden verzameld uit diverse bronnen, en in diverse talen. Deze worden continue opgehaald en in een database geplaatst worden. De gebruikers kunnen een abonnement afsluiten en kiezen welke nieuwsberichten ze geïnteresseerd in zijn. Dit kan bijvoorbeeld op taal of genre.

Het project is opgesplits in 3 modules, een front-end, een back-end en een feeder.

de front-end

De front-end is een user-interface, ontwikkeld in React, die beschikbaar gesteld wordt voor gebruikers om de nieuwsberichten te lezen, te filteren. De front-end maakt het mogelijk om te navigeren tussen diverse pagina's. De data die de front-end nodig heeft, wordt beschikbaar gesteld door de back-end.

De back-end

De back-end is een API. Deze heeft als doel om nieuwsberichten op een veilige manier beschikbaar te stellen aan de front-end. De back-end levert diverse data zoals nieuwsberichten, beschikbare talen en genres.
Daarnaast levert het back-end ook functionaliteit voor de feeder. De feeder kan middels de back-end nieuwe nieuwsberichten opslaan in de database.

De feeder

De feeder is een microservice. Het doel van deze service is het periodiek scannen naar nieuwe nieuwsberichten. Hiervoor gebruikt het een lijst van bronnen. De feeder zorgt ervoor dat berichten, indien beschikbaar, in meerdere talen wordt opgeslagen in de database.

Mijn rol binnen het project

Mijn hoofdtaak was het ontwikkelen van de gebruikersinterface (UI) met React en TypeScript. Ik was verantwoordelijk voor het vormgeven en bouwen van een overzichtelijke en gebruiksvriendelijke applicatie, het implementeren van de zoekfunctionaliteit, en het integreren van API-koppelingen voor real-time dataweergave.

Project aanpak

We hebben het project opgesplitst in drie duidelijke onderdelen: UI, Feeder en API. Door deze taakverdeling kon ieder aan zijn eigen stukje van het project werken. Via regelmatige stand-ups, feedbackmomenten en een gezamenlijke Trello met een duidelijke planning zorgden we ervoor dat we afgestemd bleven.

Welke technologieën zijn gebruikt

- **Front-end:**
  - React: voor het bouwen van de gebruikersinterface
  - TypeScript: voor striktere typecontrole en betrouwbaardere code
  - ESLint: voor het bewaken van codekwaliteit en consistentie

- **Back-end:**
  - ASP.NET Core: voor het ontwikkelen van de API en de feeder
  - Business Library Class: voor het structureren van de logica binnen de feeder

- **Database:**
  - Microsoft SQL Server: als hoofd-databasesysteem
  - SQLite: gebruikt in ontwikkel- en testomgevingen
  - Entity Framework Core: voor database-interactie
  - Database Migrations: om wijzigingen in het datamodel gecontroleerd door te voeren
  - SQL Server Management Studio: voor beheer en query's op de database

- **Ontwikkelomgeving:**
  - Visual Studio: als belangrijkste IDE voor back-end en databaseontwikkeling

- **Samenwerking & communicatie:**
  - Azure DevOps: Versiebeheer via Gitkrakken
  - Scrum: Wekelijkse stand-ups en Planning bijhouden in de gezamenlijke trello
  - Teams: voor dagelijkse communicatie en teamoverleg
  - Outlook: het plannen van afspraken/meeting, waarbij er van je verwacht wordt dat je actief RSVP gebruikt
  - AFAS: voor inzicht in personeels- en administratieve processen

Planning

Voor de planning gebruikten we een gezamenlijke Trello waarin we onze voortgang bij hielden. We werkten volgens een de planning die ieder voor zich had gemaakt, en bespraken dagelijks of alles nog volgens schema verliep. Taken werden aangepast waar nodig was.

Wat hebben we opgeleverd

- **Front-end - UI:** Gebouwd in React met typescript, ook hebben we linting gebruikt voor het corrigeren van code. Want als iedereen (in een bedrijf) met linting werkt is het overzichtelijk als je kijkt naar de code van een ander wat er staat, het corrigeert namelijk je manier van coderen, zodat iedereen ongeveer hetzelfde typt.
- **Backend - API:** REST API voor gebruikersbeheer, voorkeuren en nieuwsberichten.
- **Feeder:** Een automatisch script dat nieuwsberichten ophaalt, opslaat en op de website toont.
- **Notificatiesysteem:** Gebruikers ontvangen wekelijks hun favoriete nieuws via e-mail, SMS of Teams.

Persoonlijke reflectie op het ontwikkelproces

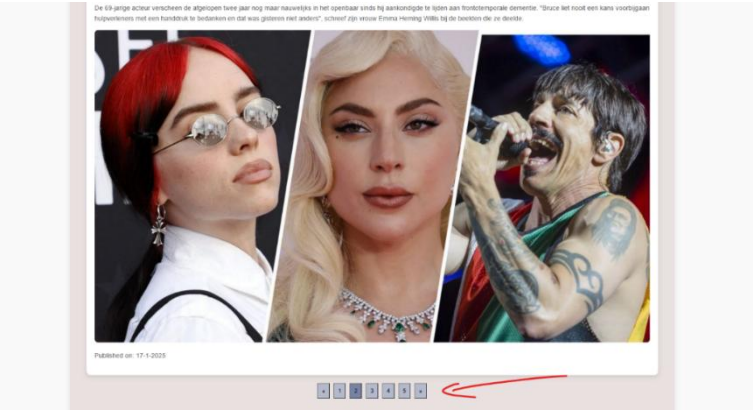
Samenwerking en planning

Het verdelen van de taken in UI, API en Feeder werkte goed. Ieder teamlid kon zich focussen op onze onderdelen, terwijl we gezamenlijk regelmatig overlegden om integratieproblemen te voorkomen. Vooral de afstemming tussen front-end en API was noodzakelijk.

Uitdagingen en oplossingen | UI

Wat heb ik geleerd:
Ik heb geleerd hoe je een front-end in React structureert met componenten en hoe je met TypeScript typeveiligheid en structuur aanbrengt. Ook heb ik de werking van paginering en data-binding beter leren begrijpen.

Uitdagingen:
Een lastig punt was het correct tonen van nieuwsberichten op de pagina, vooral bij de zoekfunctie. Daarnaast werden datums in eerste instantie niet weergegeven. Met hulp van Christian heb ik dit opgelost.
Ook kreeg ik feedback van de klant dat hij liever een andere paginering wilde. Het duurde even voordat ik de nieuwe oplossing goed werkend kreeg, maar het is uiteindelijk wel gelukt.
Wat ging goed, wat kon beter:
Het structureren van de interface en het gebruik van componenten ging goed.
Beter had gekund:
Eerder feedback vragen aan Mike (of mijn groepje) om sneller te verbeteren. Ook had ik eerder vragen moeten stellen als ik iets niet volledige snapte of ergens vasthing en er niet uit kwam.



Wat heb ik geleerd

Technische groei:
Ik heb veel geleerd over het bouwen van een complete full-stack app. Vooral de samenwerking tussen front-end en back-end, en het correct opzetten van notificatiesystemen, waren nieuw voor mij. Ook was het nog vrij nieuw voor mij om te werken met backend, hier heb ik ook veel van kunnen leren.

Probleemoplossend denken:
Vooral in react heb ik mezelf goed kunnen uitdagen met het koppelen van de backend. Maar toch was de grootste uitdaging de backend (API).

Security & Privacy:
Door het toepassen van hashing, veilige API-routes en AVG-regels heb ik meer geleerd over veilige softwareontwikkeling.

Gebruik van design patterns:
Het splitsen van logica en presentatie in React maakt de app goed onderhoudbaar en overzichtelijk. Componenten zoals knoppen en lijsten kunnen hergebruikt worden. Het gebruik van service componenten levert een gestructureerde manier om de communicatie met API's te isoleren.

Wat zou ik volgende keer anders doen

Vroeger testen:
Door eerder feedback te vragen aan de klant/begeleider hadden we sneller UI-problemen kunnen opsporen.
Ook door eerder dingen te bespreken met de groep/begeleider zouden we niet te lang hadden hoeven vasthangen bij bepaalde stukken.

Conclusie

Dit project heeft me niet alleen geholpen om mijn technische vaardigheden te verbeteren, vooral op het gebied van front-end en samenwerking met backend. Maar ook om beter te plannen en samen te werken in een team, de samenwerking binnen het team verliep goed en we hebben waardevolle feedback aan elkaar gegeven.
Door de combinatie van API, Feeder en UI hebben we een complete nieuws-website geleverd voor The KEMBIT Times. Ik ben trots op het resultaat en neem deze kennis mee voor toekomstige projecten.
Daarnaast ben ik er ook achter gekomen dat het niet alleen handiger, maar ook leerzaam is om bij vastlopen niet te blijven hangen in het zelf proberen uit te zoeken, maar juist op tijd vragen te stellen. Hierdoor kun je problemen sneller oplossen en leer je efficiënter werken.

Begrippenlijst

Front-end
Dit is het deel van een website dat je kunt zien en gebruiken. Het gaat om de knoppen, teksten en alles wat op het scherm staat. Je maakt het bijvoorbeeld met HTML, CSS en React.

Back-end
Dit is de 'achterkant' van een website. Hier gebeurt alles wat je niet ziet, zoals het ophalen van gegevens of het opslaan van iets in een database. Meestal wordt dit gebouwd met programmeertalen zoals C# of Java.

Database
Een plek waar gegevens worden opgeslagen. Denk aan een grote digitale kast met lades vol informatie, zoals nieuwsberichten of gebruikers.

Feeder
Een klein programma dat automatisch nieuwe nieuwsberichten ophaalt van andere websites en deze in de database zet.

Linting
Een hulpmiddel dat controleert of je code netjes en op dezelfde manier is geschreven. Zo blijft de code overzichtelijk voor jou en je teamgenoten.

Component (in React)
Een blokje code dat een deel van de website maakt, bijvoorbeeld een knop of een nieuwskaart. Je kunt zo'n blokje vaker gebruiken in je website.

API
Een soort brug tussen de front-end en back-end. De front-end stelt een vraag, en de back-end geeft antwoord. Bijvoorbeeld: "Geef mij het laatste nieuwsbericht."

TypeScript
Een programmeertaal die lijkt op JavaScript, maar waarbij je extra duidelijk moet zijn over wat voor soort gegevens je gebruikt. Hierdoor maak je minder fouten in je code.

Entity Framework
Een hulpmiddel om makkelijker met de database te werken in je code. Je hoeft dan minder zelf SQL-commando's te schrijven.

Paginering
Als er veel nieuwsberichten zijn, worden ze verdeeld over meerdere pagina's. Dit heet paginering. Je kunt dan bijvoorbeeld op "Volgende pagina" klikken.