Technisch ontwerp

Project	:	NewsSite
Opdrachtgever:		Mike Wijnen
Auteur	:	Christian Schroth
Datum	:	10-11-2025
Versie	:	1.0

1. Inleiding

Onze NewsSite is ontworpen om gebruikers snel en betrouwbaar toegang te geven tot het laatste nieuws. Met een intuïtieve interface kunnen bezoekers eenvoudig artikelen vinden die relevant zijn voor hun interesses. De site ondersteunt zowel nationale als internationale nieuwsbronnen en biedt gepersonaliseerde aanbevelingen op basis van gebruikersvoorkeuren. Dit document beschrijft de technische aspecten van de ontwikkeling en implementatie van de NewsSite.

2. Oplossing in kaart

We gaan een nieuwswebsite maken die voor iedereen beschikbaar is. De website zal een breed scala aan onderwerpen bestrijken, variërend van politiek en economie tot sport en entertainment. In plaats van een zoekfunctie, categoriseren we nieuws in verschillende soorten, zodat gebruikers eenvoudig toegang hebben tot de onderwerpen die hen interesseren. Gebruikers kunnen zich registreren en inloggen om zich te abonneren op specifieke nieuwscategorieën. Ze ontvangen dan automatisch e-mail meldingen wanneer er nieuwe artikelen worden gepubliceerd binnen hun geselecteerde interesses. De site wordt geoptimaliseerd voor zowel desktop als mobiele apparaten om een soepele gebruikerservaring te garanderen. Daarnaast maken we gebruik van een robuust backend-systeem om een snelle laadtijd en hoge beschikbaarheid te waarborgen.

3. Architectuurontwerp

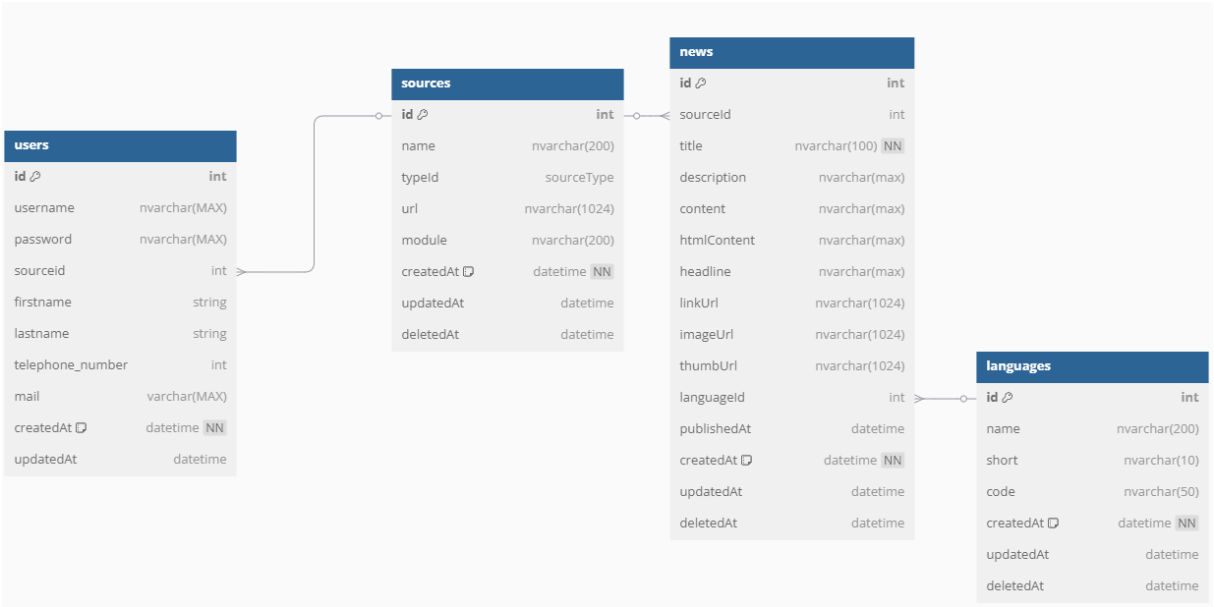
- **Overzicht van de architectuur**: Er wordt een website gemaakt met React. Deze website wordt gekoppeld aan een ASP.NET Core API, die alle nodige data uit een SQL Server database ophaalt. Het nieuws wordt via een Feeder opgehaald van internet en verwerkt in de database, zodat gebruikers altijd up-to-date artikelen kunnen bekijken.
- **Technologieën en frameworks:** De frontend wordt ontwikkeld met React en TypeScript, terwijl de backend is gebaseerd op ASP.NET Core en Dapper voor database-interacties. De SQL Server database slaat nieuwsartikelen, gebruikersgegevens en abonnementen op. Voor e-mailnotificaties maken we gebruik van een externe SMTP-service.

- **Component Diagram:** Hier wordt een visueel diagram toegevoegd dat de interactie tussen frontend, backend, database en externe diensten (zoals de Feeder en e-mailservice) laat zien.

- **Communicatie tussen componenten:**
  De frontend communiceert met de backend via RESTful API's. De API is opgebouwd volgens een gelaagde architectuur met een Controller Layer, een Business Layer en een Repository Layer**:**
    - **Controller Layer:** Ontvangt API-verzoeken van de frontend en valideert deze.
    - **Business Layer:** Bevat de logica voor het verwerken van nieuwsfeeds en gebruikersacties. Bij belangrijke updates stuurt de backend automatisch e-mailnotificaties naar abonnees.
    - **Repository Layer:** Beheert de database-interacties en haalt of slaat gegevens op.De API haalt gegevens uit de database en verwerkt nieuwsfeeds.

4. Databaseontwerp

- **ERD (Entity-Relationship Diagram):**
  De database is visueel gemodelleerd met behulp van dbdiagram en bevat de belangrijkste entiteiten zoals gebruikers, nieuwsartikelen, categorieën en abonnementen. Dit diagram toont de relaties tussen tabellen en hoe gegevens worden beheerd.



https://dbdiagram.io/d/NewsApi-672df869e9daa85acac808ce

- **Tabellen en relaties:**
  De database bevat de volgende tabellen:
  - **Sources:** Bevat informatie over nieuwsbronnen, inclusief naam, type, URL en module.
  - **Languages:** Opslag van talen waarin nieuwsartikelen beschikbaar zijn, inclusief naam, code en afkorting.
  - **News:** Bevat nieuwsartikelen met een titel, beschrijving, inhoud, HTML-versie, afbeeldingen en een koppeling naar de bron. Elk artikel is gekoppeld aan een bron en een taal.

- **Belangrijke constraints en indexes:**
  - Primaire sleutels op ID-kolommen.
  - Foreign key-relaties:
    - news.sourceId verwijst naar sources.id.
    - news.languageId verwijst naar languages.id.

5. API Ontwerp

De API vormt de kern van de communicatie tussen de frontend (React) en de backend (SQL Server-database). De API is ontwikkeld met ASP.NET Core en is opgezet volgens een gelaagde architectuur om een duidelijke scheiding van verantwoordelijkheden, onderhoudbaarheid en uitbreidbaarheid te garanderen.

De API bestaat uit drie hoofdcomponenten:

- **Controller Layer:**
  - Ontvangt en verwerkt inkomende HTTP-verzoeken van de frontend.
  - Valideert gebruikersinput om verkeerde of schadelijke data te voorkomen.
  - Stuurt de geldige verzoeken door naar de Business Layer.
  - Retourneert het resultaat als een gestandaardiseerde HTTP-response (meestal in JSON-formaat).
- **Business Layer:**
  - Bevat de toepassingslogica van het systeem.
  - Verwerkt nieuwsfeeds, gebruikersacties en abonnementen.
  - Roept de Repository Layer aan voor data-opvraging of -opslag.

  Verantwoordelijk voor:

  - Het filteren van nieuws op categorie of taal;
  - Het verwerken van binnenkomende nieuwsfeeds (via de Feeder);
    Het genereren van meldingen of notificaties;

- Het uitvoeren van validaties op bedrijfsniveau (bijv. dubbele artikelen voorkomen).

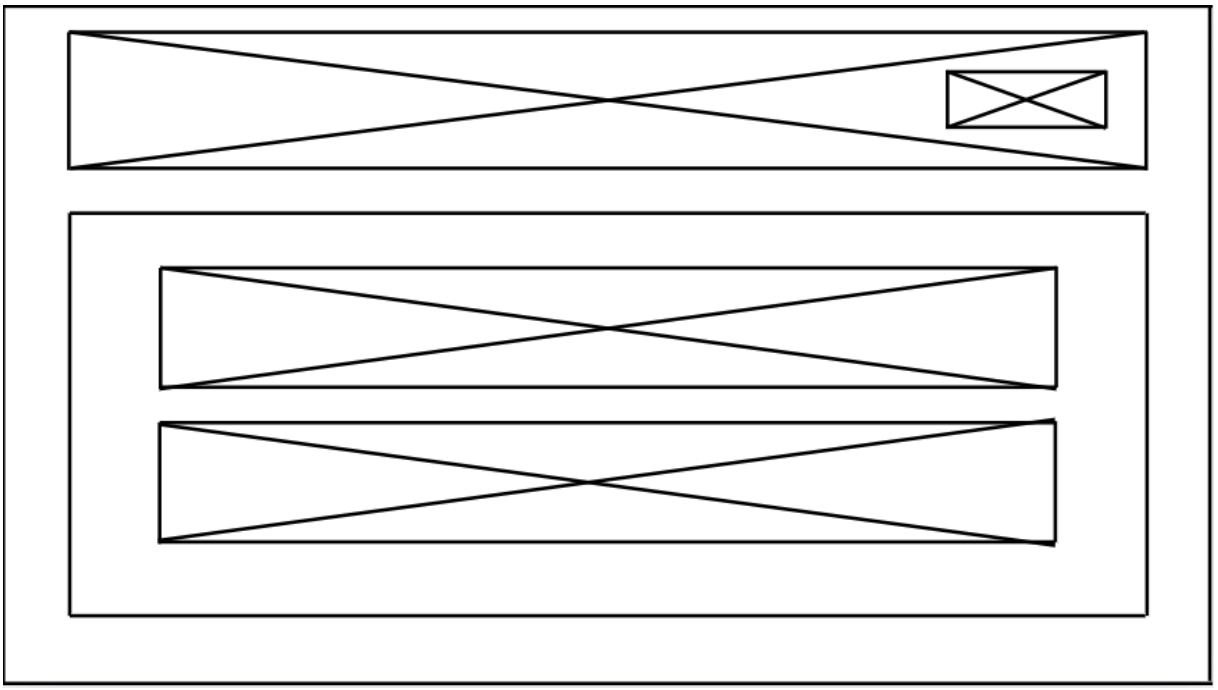
- **Repository Layer:**
  - Beheert de database-interacties via Dapper.
  - Voert query's uit op de SQL Server-database en retourneert de resultaten aan de Business Layer.
- **API-endpoints:**
  - GET /api/news – Haalt de laatste nieuwsartikelen op.
  - GET /api/news/{id} – Haalt een specifiek nieuwsartikel op.

- **Foutafhandeling:**
  - Gebruikt middleware voor het afvangen en loggen van fouten.

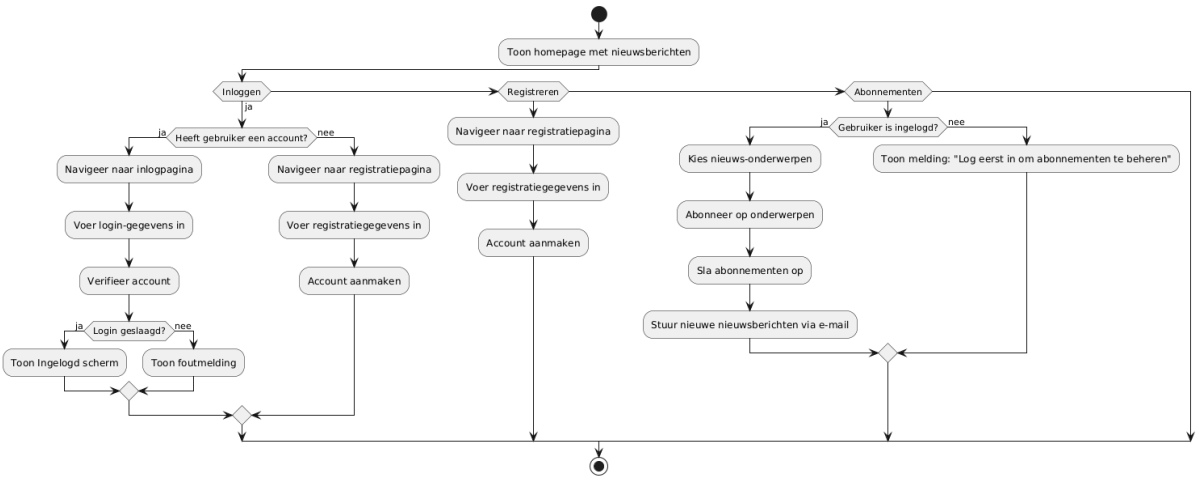
6. Frontend Ontwerp

- **UI/UX:**
  (https://www.canva.com/design/DAGf8CetjVs/0Dl0I7-Y1MsxMSCTSTvkoQ/edit**)**

- **Wireframe:**



- **Activiteit Diagram**



---

7. Onderhoud en Schaalbaarheid

Een goed onderhoudsplan en schaalbaarheid strategie zijn essentieel om de betrouwbaarheid en prestaties van de newssite te garanderen, vooral bij toenemende gebruikersaantallen en nieuwsbronnen.

Onderhoudsstrategie:

- Regelmatige updates van de API en frontend om beveiligingslekken en bugs te verhelpen.
- Gebruik van logging en monitoring om fouten en prestatieproblemen vroegtijdig te detecteren.
- Versiebeheer via Git, met een gestructureerde workflow zoals feature branches en pull requests.
- Automatische tests via unit tests en handmatig.

Schaalbaarheid Opties:

- **Database-optimalisatie**: Indexen en caching strategieën toepassen om database query's te versnellen.
- **Load balancing**: Verkeer verdelen over meerdere servers om piekbelasting op te vangen.

- **Cloud-gebaseerde hosting**: Gebruik van **Azure App Services**

Performanceoptimalisaties:

- **Lazy loading** implementeren in de frontend om alleen de noodzakelijke content te laden.

8. Security / Beveiliging

Om de veiligheid en betrouwbaarheid van de NewsSite te waarborgen, worden verschillende technische en organisatorische maatregelen geïmplementeerd. Deze maatregelen zijn gericht op het beschermen van gebruikersdata, het voorkomen van misbruik en het garanderen van continuïteit van de dienstverlening.

Communicatie Beveiliging

- **Veilige verbindingen (HTTPS/TLS):** Alle communicatie tussen de frontend, backend en externe diensten verloopt via HTTPS om te voorkomen dat data tijdens transport onderschept of gewijzigd kan worden.
- **CORS-beperkingen:** Cross-Origin Resource Sharing (CORS) wordt strikt ingesteld zodat alleen vertrouwde domeinen toegang hebben tot de API.

Authenticatie en Autorisatie

- **JWT (JSON Web Tokens):** Wordt gebruikt voor gebruikersauthenticatie en sessiebeheer. Elk verzoek van een ingelogde gebruiker bevat een token dat op de server wordt gevalideerd.

Gegevensopslag en versleuteling

- **Beveiligde opslag van gevoelige gegevens:** Connection strings, API-sleutels en andere gevoelige informatie worden opgeslagen in Azure Key Vault of User Secrets , zodat deze niet in de broncode of configuratiebestanden staan.

Input Validatie en Aanval Preventie

- **Input Validatie:** Alle input van gebruikers wordt gecontroleerd op geldigheid, lengte en type. Dit voorkomt SQL-injecties, XSS-aanvallen en andere vormen van misbruik.
- **Parameterized queries:** Alle database query's in Dapper maken gebruik van parameterized queries, waardoor directe SQL-injecties worden voorkomen.
- **Rate limiting:** Het aantal verzoeken per gebruiker of IP-adres wordt beperkt om brute-force-aanvallen en overbelasting van de API te voorkomen.

Logging en Monitoring

- **Logging van kritieke gebeurtenissen:** Inloggen, mislukte logins, accountwijzigingen en beheeracties worden gelogd voor auditdoeleinden.
- **Monitoring en alerts:** Integratie met Azure Monitor en Application Insights stelt beheerders in staat om performance problemen, fouten en verdachte activiteiten real-time te volgen en hierop te reageren.

Privacy

- **Minimale dataverzameling:** Alleen gegevens die noodzakelijk zijn voor functionaliteit en abonnementen worden opgeslagen.

Aanvullende maatregelen

- **Beveiliging van third-party services:** Externe diensten, zoals e-mail via SMTP, zijn beveiligd met een versleutelde verbinding en een unieke API-sleutel voor toegang.
- **Regelmatige updates en patches:** De backend, frontend en database worden regelmatig geüpdatet om beveiligingslekken te dichten.
- **Back-ups:** Er worden periodieke back-ups gemaakt van de database om gegevensverlies te voorkomen bij incidenten.