Rapport

TEAM
Achraf Yandouzi,
Yonah Thienpont,
Yousra Smits,
Victor Peña Alvarado
Youssef El Aseri,
Judith Hershko

Inhoud

Voorwoord

Met trots stellen we onze applicatie voor. Voor deze opdracht hebben we een webapplicatie gebouwd die nieuws van verschillende nieuwsbronnen verzamelt en toont. We hebben enorm veel geleerd bij het maken van dit project. In dit rapport willen we graag een paar belangrijke aspecten van de webapplicatie uitgebreid toelichten.

Bij het maken van dit project hebben we de kans gekregen om onze technische en persoonlijke vaardigheden verder te ontwikkelen.

We hebben geleerd om te werken met verschillende technologieën en hoe ze samen te integreren om een gebruiksvriendelijke applicatie te maken. Een van onze belangrijkste aandachtspunten was gebruiksvriendelijkheid, schaalbaarheid en betrouwbaarheid. Daarnaast hebben we ook aandacht besteed aan het personaliseren van feeds. Op die manier kan een gebruiker nieuwsberichten ontvangen die aansluiten bij hun interesse.

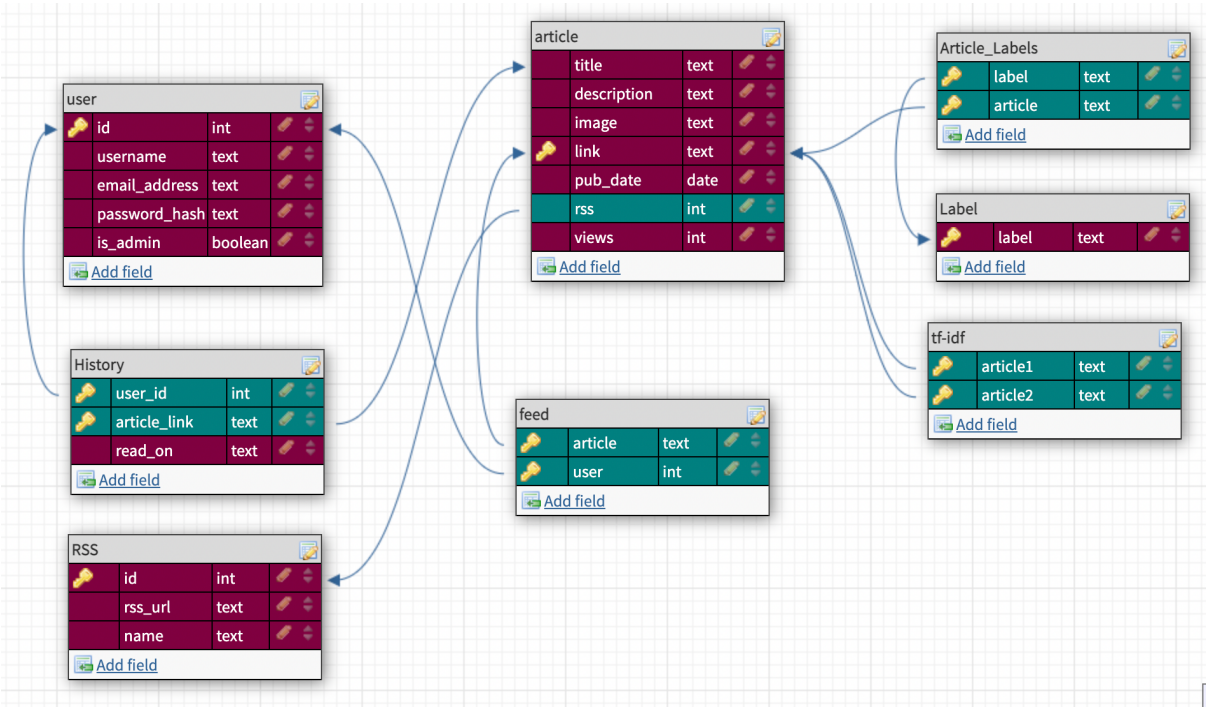
Naast technische vaardigheden hebben we belangrijke lessen geleerd over communicatie,samenwerking en projectbeheer. Gedurende de loop van dit project hebben we effectief kunnen samenwerken, ideeën uitgewisseld en elkaar ondersteund om onze doelen te bereiken. We gaven elkaar regelmatig feedback. Dit hielp ons om persoonlijk te verbeteren en als team te groeien.

Als laatste willen we prof. B. Goethals, Mr. J. De Pauw en Mevr. A. De Bruyn bedanken voor hun waardevolle ondersteuning en begeleiding gedurende dit project.

Database

Voor de nieuwsapp hebben we een database gemaakt die de relevante informatie van de applicatie bijhoudt. Op die manier kunnen we alle (nodige) informatie efficiënter ophalen. Er werd ervoor gekozen om de database met sqlAlchemy te maken. De reden hiervoor is dat het op deze manier gemakkelijk was om de database te configureren met de rest van de website. Daarboven was er voor sqlAlchemy veel documentatie beschikbaar.

De onderstaande afbeelding toont een diagram van de database die we hebben geïmplementeerd.



In deze database is het mogelijk om alle kolommen in meer detail te onderzoeken. De groene kolommen stellen de foreign-keys voor. Met de pijltjes weten we hoe de gegevens in de verschillende tabellen met elkaar verbonden zijn.

User

Dit zijn de gegevens die we van een gebruiker bijhouden. Bij het aanmaken van een account, worden de gebruikersnaam, e-mailadres en password hash bijgehouden. Elke gebruiker krijgt ook een unieke id. Als de gebruiker een admin is , komt het veld ‘is_admin’ op true te staan. Dit zal de gebruiker toegang geven tot extra functionaliteiten in de applicatie.

History

In deze tabel wordt de gebruiker geschiedenis bijgehouden. Wanneer een gebruiker op een artikel klikt, komt de gebruiker id, link van het artikel en de datum en tijd dat het artikel gelezen werd in de tabel te staan.

RSS

In deze tabel worden de RSS-feeds van de artikelen opgeslagen. Een admin kan de RSS-link en de naam van de nieuwsbron meegeven. Elke RSS-link krijgt dan automatisch een unieke id.

Article

Via de scraper worden de artikelen in deze tabel toegevoegd . Indien een artikel een beschrijving heeft, kan deze eventueel gebruikt worden om de tf-idf te optimaliseren. Indien een gebruiker op een artikel klikt, wordt de ‘views’-kolom met 1 verhoogd. Deze informatie kan gebruikt worden om artikelen te sorteren op populariteit.

Feed:

De feed tabel wordt gebruikt om bij te houden welke artikelen een gebruiker te zien krijgt. De link van het artikel en de gebruikers-id worden in deze tabel bijgehouden.

Article_Labels

Elke artikel kan meerdere labels hebben. Er is dus een veel-op-veel relatie tussen de artikelen en de labels. Deze tabel houdt bij welke labels aan welk artikel werd toegekend.

Label

Dit tabel houdt alle mogelijke labels bij die een artikel kan hebben.

tf-idf

Om te voorkomen dat twee gelijkaardige artikelen op de feed van dezelfde gebruiker komen, wordt de tf-idf berekend tussen alle artikelen. Deze tabel gaat twee verschillende artikelen met elkaar koppelen. De links worden gesorteerd vooraleer ze in de tabel komen om te voorkomen dat artikelen twee keer (in twee verschillende kolommen) met elkaar worden vergeleken.

Wanneer kolommen van verschillende tabellen afhankelijk zijn van elkaar, hebben we de nodige ‘cascading policies’ in plaats gezet om te voorkomen dat een kolom naar iets refereert dat niet bestaat.

In ConnectDB.py werd een wrapper geschreven om operaties op de database, tabellen en/of kolommen uit te voeren. Op die manier is er geen voorkennis nodig over hoe de database in elkaar zit om ze te gebruiken.

Testomgeving

Om functies op de database te testen, werd een testomgeving opgesteld. Deze is te vinden in db_tests.py. De testomgeving voorkomt dat de inhoud van de database van de app zelf wordt beïnvloed bij het uitvoeren van testen. Zelfs al was er geen nood om ingewikkelde queries op te stellen, staat de testomgeving klaar voor gebruik.
Om te voorkomen dat de directories steeds moeten worden aangepast bij het uitvoeren van de tests, wordt de src-map behandeld als een library via setup.py.

Sessie

Anonieme gebruiker

Indien een gebruiker niet is ingelogd, wordt hem een (unieke) cookie toegekend. Dit wordt gedaan in de homepage .Zolang gebruikers vanuit dezelfde computer en browser verbinden, kan er dus een state bijgehouden worden.

Registreren

Een gebruiker kan zich registreren op de site. In routes.py wordt de gebruiker via register_user() in de database toegevoegd.
Het wachtwoord dat wordt opgeslagen in de database heeft een hash + salt met behulp van bcrypt. Een hashed versie van het wachtwoord wordt bijgehouden. Zo kan de gebruiker zich inloggen met de username en het wachtwoord dat hij heeft gekozen bij het registreren. Zolang de gebruiker ingelogd blijft, wordt alle informatie van die gebruiker bijgehouden en opgeslagen in de database. We doen dit aan de hand van de klik geschiedenis (dit is de functie click() in routes.py). Bij het klikken op een artikel worden de datum van de klik, het artikel waarop de gebruiker heeft geklikt en de ID van de gebruiker bijgehouden in de History tabel. Het registeren gaat enkel als het paswoord tenminste 6 karakters heeft. Dit werd gedaan om veiligheidsredenen.

Login

Bij het inloggen worden de opgegeven gebruikersnaam en (hashed) paswoord vergeleken met wat er in de database staat.

Admin

Een nieuwe admin wordt aangemaakt in register_admin(). In de database hebben we een veld is_admin toegevoegd om na te kijken of een gebruiker al dan niet een admin is. Op die manier kunnen we toegang tot de admin page weigeren voor gebruikers die geen admin zijn.

Als een admin ingelogd is, komt er boven op de website een "dashboard"-veld te staan. De beheerder krijgt hier toegang tot de RSS feeds en andere beheerders van de website.
Om deze informatie te tonen, worden de functies get_admins() en get_feeds() aangeroepen.

Bij het verwijderen van een admin en/of feed worden delete_admin() en delete_feed() respectievelijk aangeroepen.
Als een admin een geldig feed of admin toevoegt, worden die automatisch in de database bijgehouden. Er wordt echter eerst nagekeken of de admin geldige gegevens heeft ingevuld bij het toevoegen van een admin of feed.

TF-IDF

Het TF-IDF algoritme werd geschreven met behulp van NLTK en gensim. De geïmplementeerde functie kan twee .txt-files met titels en beschrijvingen vergelijken en een dictionary teruggeven met waarden die duiden op hoe gelijkaardig de inhoud is. De code is gebaseerd op deze post.

Artikelen Linken

Wanneer de artikelen in de database zijn ingeladen, kan de functie link_articles() in link_articles.py worden gerund. Deze functie gebruikt het net benoemde TF-IDF algoritme. Wanneer het algoritme een score hoger dan 0.8 teruggeeft, zullen de titels als gelijk worden beschouwd.
Merk op: een artikel wordt ook vergeleken met zichzelf en zal een score van 1.0 of hoger teruggeven. Daarom wordt ook nagekeken dat de inhoud van twee artikelen niet identiek is. Sommige nieuws-outlets zullen heel gelijkaardige titels hebben, beschouw het volgende voorbeeld:

- ‘CLUBNIEUWS APRIL. De belangrijkste nieuwtjes over de Limburgse basketbalclubs.’
- ‘CLUBNIEUWS APRIL. De belangrijkste nieuwtjes over de Limburgse voetbalclubs.’.

Dit zijn uiteraard twee verschillende artikelen die wel gematcht worden. Het is niet verkeerd om er van uit te gaan dat dezelfde site nooit twee keer hetzelfde artikel zal plaatsen (als ze dat toch doen is het waarschijnlijk de bedoeling dat je beide artikels ook ziet), dus wordt er gekeken of de url hetzelfde is met behulp van een reguliere expressie: ‘https:\/\/([0-z]+\.)+[0-z]*\/’. Hierdoor zullen artikelen ook niet als duplicaat van zichzelf in de database worden toegevoegd.

Op de website worden de gelijkaardige artikelen aan elkaar gekoppeld. De gebruiker krijgt dan favicons te zien van de andere mogelijke bronnen en kan kiezen welke hij wil bekijken.

RSS

Scraper

In RSSScraper.py worden de RSS feeds gescrapt.
De RSS scraper wordt periodisch aangeroepen, zodat onze website steeds up-to-date is.

We hebben ervoor gekozen om voor het scrapen alle oude artikelen (met een publicatiedatum > 30 dagen) te verwijderen. Daarna kan de parser worden aangeroepen om alle artikelen uit de RSS te halen.

Parser

Voor het parsen van RSS feeds, maken we gebruik van de library feedparser. Op die manier hebben we support voor al de volgende standaarden:

- RSS 0.90
- Netscape RSS 0.91
- Userland RSS 0.91
- RSS 0.92
- RSS 0.93
- RSS 0.94
- RSS 1.0
- RSS 2.0
- Atom 0.3
- Atom 1.0
- CDF
- JSON feeds

De parser loopt over alle artikelen in de feed om de volgende informatie uit een artikel kan halen:

- titel
- link
- afbeelding
- beschrijving
- label(s)
- publicatiedatum

Sommige artikelen hebben simpelweg geen afbeelding, in dat geval tonen we een “No Image” afbeelding op de website. Dit is om uniformiteit op de feed van de gebruiker te behouden.

Nadat de nieuwe artikelen werden toegevoegd wordt de voorheen besproken link_articles() aangeroepen om alle gelijkaardige artikelen te koppelen met elkaar.

Feed

Om de artikelen op de feed van een gebruiker te weergeven hebben we een aparte klasse geschreven, namelijk ArticlesFetcher.py Deze gaan de artikelen (met hun gekoppelde gelijkaardige artikelen) uit de database halen. Op die manier kunnen ze op de feed van een gebruiker worden weergegeven.

Sorting options

We hebben ervoor gekozen om de gebruiker 3 opties te geven om de artikelen op de feed te sorteren, namelijk:

- recentheid
- trending
- aanbevolen

Elke sorteeroptie heeft zijn eigen fetch()-functie die de juiste artikelen uit de database haalt.

recentheid

Om de artikelen te sorteren op recentheid wordt er gekeken naar de publicatiedatum.

trending

Om te voorkomen dat de oudste artikelen ook worden beschouwd als “meest bezochte” artikelen, worden de views van artikelen die gepubliceerd werden binnen een tijdspanne van 7 dagen vergeleken met elkaar.

aanbevolen

Hiervoor wordt er gebruikgemaakt van de klik-geschiedenis van de gebruiker. De artikelen die een gebruiker bezocht heeft, worden vergeleken met alle andere artikelen in de database via get_resemblance(). *(Deze functie werd oorspronkelijk geschreven voor het TF-IDF algoritme en kan nu hergebruikt worden)*

Op die manier krijgt elk artikel een gelijkenis factor. Indien die factor hoger is dan een welbepaalde drempel, raden we dat artikel aan aan de gebruiker.

Infinite scrolling

Om het ophalen van artikelen sneller te maken, werd infinite scrolling geïmplementeerd.
Er werd gebruikgemaakt van de Infinite Scroller component uit de React-dom bibliotheek.
In ArticlesFetcher.py, kunnen de fetch() functies een pagina-argument meekrijgen.
Dit argument bepaalt welke pagina, of groep van een bepaald aantal artikelen (in dit geval 10) wordt gehaald uit de resultaten van de uitgevoerde query.
Wanneer de gebruiker voorbij een bepaald punt scrollt, neemt dit argument toe met 10 en wordt de fetch() functie opnieuw aangeroepen.

Sharing

Er werd ervoor gezorgd dat een gebruiker artikelen kan delen. Bij elke artikel wordt een share-knop voorzien. Iedere knop heeft een intuïtief icoon en het delen gebruikt de query's van het sociaal medium. Ook is er een knop om eenvoudig naar het clipboard te kopiëren.

Labels

De gebruiker kan artikelen filteren op labels. Boven op de website zijn de labels zichtbaar die uit de artikelen werden gescraped. De gebruiker kan op deze labels klikken om de artikelen te bekijken die die labels hebben. Het is onder andere ook mogelijk om meerdere labels tegelijkertijd te selecteren.

Excluding Lables

Naast het selecteren van labels, kan de gebruiker ervoor kiezen om artikelen in zijn feed te krijgen waar bepaalde labels niet voorkomen. De excluded labels worden meegegeven helemaal vanonder in de homepage. In routes.py worden in get_articels() de excluded labels meegegeven. Op die manier kunnen voor elke fetch-functie (voor recentheid, populariteit en trending) de artikelen worden weggefilterd.

Search

Naast labels, is het ook mogelijk om zoektermen in te geven in de zoekbalk, dan worden de artikelen hierop gefilterd.
Om het doorzoeken van artikelen zo efficiënt mogelijk te maken, wordt elasticsearch gebruikt. De code is te zien in search.py.
De zoektermen worden vergeleken met de titel en beschrijving van het artikel. Er wordt echter prioriteit gegeven aan de titel.
Ook wordt er rekening gehouden met typfouten. In de query wordt dit behandeld met de "fuzziness"-factor. Alweer wordt er prioriteit gegeven aan exacte matches.

De volledige query ziet er als volgt uit:

```
# Define the query
    query = {
        "size": 1000,
        "query": {
            "bool": {
                "should": [
                    {"term": {"title": {"value": input_string, "boost": 3}}},
                    {"term": {"description": {"value": input_string, "boost": 2}}}
                ],
                "must": {
```

```
                "multi_match": {
                    "query": input_string,
                    "fields": ["title^2", "description"],
                    "type": "best_fields",
                    "fuzziness": "AUTO"
                }
            }
        }
    }
}
```

Veiligheid

CSRF Protection

Onze website is bestendig tegen CSRF aanvallen. Bij het indienen van formulieren wordt een CSRF token meegegeven en gecheckt. Dit is te vinden in get_csrf_token(). Dit zal CSRF-aanvallen voorkomen en de site veiliger maken. Voor meer informatie op CSRF verwijzen we naar de wikipedia pagina.

Tech Stack

ReactTS

Er werd veel nagedacht over welke bibliotheken er gebruikt zullen worden om het design van de website te ontwikkelen. Uiteindelijk werd er besloten om React te gebruiken.
Deze is een van de populairste Javascript-framework in de industrie. Het is dus mogelijk om veel documentatie te vinden en uitgebreide features te gebruiken.
Daarnaast gebruikt React component based architectuur. Dit zorgt ervoor dat de UI in kleinere herbruikbare componenten kan worden opgesplitst.

React is ook een goede optie bij de mobile first development. Met React Native is het mogelijk om dezelfde omgeving te gebruiken voor mobiel- en webapplicatie.

Bij het ontwikkelen van de front-end wordt er gebruikgemaakt van React typescript. Deze zal ervoor zorgen dat de structuur en vorm van de code verbeterd wordt.

Vite

Daarnaast wordt er voor de front-end gebruik gemaakt van Vite. Deze is optimaal om voor een snelle en responsieve ontwikkeling te zorgen. Indien een wijziging wordt gemaakt in de front-end, worden ze onmiddellijk toegepast. De modules worden direct bijgewerkt zonder de hele pagina opnieuw te moeten laden.

Sass

Om de CSS code te optimaliseren wordt er ook gebruikgemaakt van Sass. Deze maakt het onderhouden en schrijven van CSS-stijlbladen eenvoudiger en leesbaar.

Front-End Design

In deze link wordt het design getoond van de website. Er werd ervoor gekozen om een lichtblauwe kleurpatroon te kiezen voor de website. Dit gaat vooral om de leesbaarheid en betrouwbaarheid van de nieuws website te verbeteren.
Lichtblauw biedt een hoog contrast met de inhoud van de artikelen. Het is ook een neutrale achtergrondkleur die de nadruk legt op de inhoud van de website. De rustgevende kleur biedt de mogelijkheid aan de gebruiker om de artikelen met gemak te bekijken.