University of Antwerp
Faculty of Science

Programming Project Databases

Opgave 2022–2023

Universiteit Antwerpen

News Aggregator

Docent:
Prof. Bart Goethals

Begeleiders:
Joey De Pauw
Annick De Bruyn

15 februari 2023

1 Inleiding

In het vak *programming project databases* leer je een uitgebreid software project ontwikkelen in teamverband. Deze opgave werd opgesteld met de volgende leerdoelstellingen in gedachte:

- De theorie van het vak "Introduction to Databases" in de praktijk omzetten.
- Onafhankelijk in team kunnen werken.
- Werk plannen en taken verdelen.
- Creatief, probleemoplossend denken.
- Duidelijk rapporteren over vooruitgang en de gemaakte keuzes.
- Kwaliteitsvolle software schrijven, met oog voor bruikbaarheid, efficiëntie en uitbreidbaarheid.

Dit is een erg omvangrijk projectvak. Er vindt **geen** schriftelijk examen plaats in juni, en er is **geen** tweede zittijd mogelijk. Jullie worden in groep beoordeeld op basis van het opgeleverde software product, rekening houdend met bovenstaande doelstellingen.

In dit document vind je naast deze inleiding ook nog sectie 2.1 over de klant, sectie 2 waarin de basisvereisten van de opdracht van dit jaar worden uitgelegd en sectie 3 waar alle praktische aangelegenheden en het verloop van evaluaties uitgelegd worden.

2 Opdracht

De opdracht bestaat erin een webapplicatie te bouwen die nieuws van verschillende nieuwsbronnen verzamelt en toont. Er bestaan reeds enkele voorbeelden hiervan op de markt zoals Google News, Digg en The Factual. Er worden enkel titels, beschrijvingen en foto's getoond, waarna men het artikel via een link op de desbetreffende nieuwsbron zelf kan lezen (in een ander tab-blad).

Artikels worden in een eenvoudige layout aan de gebruiker getoond (bvb. een eenvoudige lijst, waarin de artikels onder elkaar staan). Een recommender systeem bepaalt via een API de volgorde waarin artikels aan een gebruiker getoond worden. Een recommender systeem bestaat naast de API uit verschillende technieken om de relevantie van artikels voor een gebruiker te bepalen. Een zeer eenvoudig recommender systeem rangschikt artikels bijvoorbeeld op publicatie timestamp. Dit noemen we een recency-based recommender. Een meer geavanceerde recommender techniek, zoals *collaborative filtering*, zal de volgorde voor een gebruiker bepalen al naargelang de artikels die reeds gelezen werden door *gelijkaardige* gebruikers. Die gelijkaardigheid van gebruikers wordt dan weer berekend aan de hand van het aantal dezelfde artikels die ze lezen.

Een belangrijke moeilijkheid bestaat eruit dat meerdere nieuwsbronnen zeer gelijkaardige (of soms identieke) artikels publiceren. Slechts 1 versie van zulke gelijkaardige artikels mag maar getoond worden, met links naar de verschillende nieuwsbronnen. Een eenvoudig classificatie algoritme kan worden gebruikt om te bepalen of twee artikels gelijkaardig genoeg zijn om samen genomen te worden door bijvoorbeeld te tellen of het aantal gelijkaardige woorden in de titel groot genoeg is. Extra: Denk na over welke versie best getoond wordt.

In sectie 2.1 staat meer achtergrond over de klant van de opdracht. Secties 2.2 en 2.3 leggen de basisvereisten van de applicatie uit in termen van entiteiten en functionaliteit. Sectie 2.4 geeft een lijst van technologieën die gebruikt kunnen worden.

2.1 De Klant

De klant waarvoor we deze opdracht maken is Froomle, een spin-off van de Universiteit Antwerpen die aan de hand van Artificiële Intelligentie technieken voor de personalisatie zorgt van online kranten. Meer bepaald wordt er voor elke lezer voorspeld welke artikels voor die lezer het meest interessant zijn op basis van diens leesgedrag. Froomle's SAAS produkt wordt wereldwijd gebruikt door onder andere Het Nieuwsblad (BE), De Standaard (BE), Gazet Van Antwerpen (BE), De Tijd/L'Echo (BE), The Boston Globe (VS), De Telegraaf (NL), La Reppublica (IT), Independent Online (ZA), en meer. De Head of Product en Head of Engineering van Froomle zullen betrokken worden bij de evaluatie van jullie opgeleverde product.

2.2 Entiteiten

De volgende entiteiten moeten minimaal herkenbaar zijn in jullie applicatie.

Gebruiker
: De gebruiker is een bezoeker van de website en wordt geïdentificeerd a.d.h.v. diens cookie. Een gebruiker krijgt een lijst van nieuws artikels (foto, titel, bron) te zien. De klik-geschiedenis van elke gebruiker wordt bijgehouden. Een gebruiker kan artikels filteren op basis van labels (bvb. categorieën zoals *Binnenland*, *Internationaal*, *Sport*, *Politiek*, *Economie*, ...).

  Via een admin pagina kan een administrator inloggen en nieuwsbronnen beheren.

Nieuwsbron
: Een nieuwsbron publiceert nieuws artikels online en maakt deze beschikbaar via één of meerdere RSS feeds. Een nieuwsbron heeft een naam, een url van de bijhorende krant, en één of meerder RSS feeds.

Artikel
: Een nieuws artikel bestaat uit een titel, een beschrijving, een foto (url), een link (url), een nieuwsbron, een publicatie datum en tijd en nul of meerdere labels (e.g. categorieën, onderwerpen).

  Gelijkaardige artikels moeten ook gegroepeerd worden op basis van een similarity maat, bijvoorbeeld tf-idf.

Merk op dat dit "logische" entiteiten zijn, maar daarom niet 1 op 1 moeten mappen op database entiteiten. Het is bijvoorbeeld aangeraden om de authenticatie gegevens van een gebruiker te scheiden van zijn/haar persoonlijke informatie (voor performantie en security). Denk goed na over jullie database design en beargumenteer alle keuzes grondig in het rapport.

2.3 Basisfunctionaliteit

Deze sectie beschrijft alle basisvereisten in termen van functionaliteit. Het is verplicht voor elk team om minimaal deze functionaliteit te implementeren en te demonstreren tijdens de eindevaluatie om te slagen voor het vak (zie Sectie 3.7: Planning).

Hoe deze functionaliteit ondersteund wordt, is aan jullie. Wij willen vooral zien dat er als team nagedacht is over de features die geïmplementeerd worden en dat jullie alle gemaakte keuzes kunnen onderbouwen. Jullie dienen ons te zien als "klant" en "product owner" waarvoor jullie een oplossing moeten bouwen. Het is dus noodzakelijk om ons vragen te stellen tijdens de contactmomenten over hoe we sommige features zouden gebruiken en wat we belangrijk vinden.

Sessies
: Gewone gebruikers worden geïdentificeerd met een cookie. Zolang gebruikers vanuit dezelfde computer en browser verbinden, kan er dus een state bijgehouden worden. Dit wordt ook een sessie genoemd (die gerust over meerdere dagen kan lopen).

Login
: Admins kunnen inloggen met een gebruikersnaam en wachtwoord. Deze gebruikers hebben meer machtigingen op de website, waaronder bijvoorbeeld RSS feeds toevoegen. Admins kunnen ook nieuwe admin accounts aanmaken en beheren.

  Voor de basisvereisten wordt *geen* rekening gehouden met security. Het is dus niet erg als de webapplicatie niet volledig veilig is tegen aanvallen, maar alle measures die genomen worden om de website te beveiligen kunnen wel extra punten opleveren, beschrijf ze dus zeker in jullie rapport (bvb: password hashing with salt).

RSS Feed Toevoegen
: Een RSS feed heeft altijd een vaste structuur in XML formaat waar een lijst van items uit gehaald kan worden. Admins hebben toegang tot een pagina waar ze RSS feeds kunnen beheren. Dit houdt in dat de feed toegevoegd, aangepast en verwijderd kan worden. Om een nieuwe feed toe te voegen moet de url opgegeven worden en moet een mapping gemaakt worden tussen de velden van de nieuwsbron en die binnen jullie eigen applicatie. Sommige bronnen zullen bijvoorbeeld *"title"* gebruiken voor de titel en andere *"titel"*. Deze moeten ook aangepast kunnen worden omdat nieuwsbronnen de velden kunnen wijzigen.

Scraping
: Alle toegevoegde RSS feeds moeten periodisch gequeried worden voor updates. Er moet dus een apart proces voorzien worden los van de web applicatie dat de database om de zoveel tijd aanvult met de nieuwste artikels.

Cleanup & Consistency
: Wanneer een entity verwijderd wordt, moeten ook alle relaties verwijderd worden (cascade policy). Zo voorkom je dat het geheugen te vol komt te staan met data die niet meer bruikbaar is. Bijvoorbeeld wanneer een nieuwsbron verwijderd wordt, dan moeten de geassocieerde feeds ook volgen.

Zorg voor een gebruiksvriendelijke web interface die er professioneel uitziet, zowel mobiel (op gsm/tablet) als op grotere schermen. Hiervoor gebruik je best het "mobile first design"

principe, waar je er van uit gaat dat de gebruiker slechts een beperkte hoeveelheid informatie tegelijk op zijn/haar scherm kan bekijken. Meer concreet kan je hiervoor gebruik maken van een CSS library, zoals uitgelegd in Sectie 2.4: Technologie.

Daarnaast is het nodig om gebruik te maken van een **A**pplication **P**rogramming **I**nterface (API) om sommige van de features te implementeren. De API moet voldoen aan het REST design principe, wat o.a. inhoudt dat de API geen state bijhoudt en dat de entiteiten centraal staan (niet de operaties).

De API voorziet een manier om rechtstreeks vanuit de front end met de server te communiceren, zonder de pagina te moeten verversen of herladen, wat de gebruiksvriendelijkheid enorm kan bevorderen. Om samen te vatten: denk goed na over hoe jullie de features implementeren. Een goed doordachte keuze en design kunnen veel onnodig werk voorkomen. Verantwoord deze keuzes ook altijd grondig in jullie rapportering.

2.4 Technologie

Inherent aan dit project is dat jullie met verscheidene nieuwe technologieën zullen moeten leren werken. Er is een groot aanbod aan tools, frameworks, libraries en services die jullie moeten combineren om de opgave te implementeren. We geven een suggestie van deze "technologieën" per categorie.

Laat de keuze van met welke technologieën jullie verder gaan niet te lang aanslepen. Wij raden aan om zo snel mogelijk enkele opties met elkaar te vergelijken en binnen jullie team te bespreken met welke jullie verder gaan. Nadat de belangrijkste keuzes gemaakt zijn, kunnen jullie beginnen met het lezen van documentatie en tutorials om jullie te verdiepen in de gekozen technologieën.

Webserver

- Flask https://flask.palletsprojects.com/
  Het Flask framework in Python is aangeraden, maar een andere programmeertaal en/of webserver is ook toegestaan.

Web Design

- HTML + CSS + Js
  Dit zijn de basiselementen van elke webpagina.
- Front end framework
  Moderne webpagina's worden vaak geschreven met veel logica in de client en slim gebruik van asynchrone calls om zogenaamde "single page applications" te maken. Deze frameworks ondersteunen dit design:
  - Vue.js https://vuejs.org/
  - React https://reactjs.org/
  - Angular https://angular.io/
- CSS Framework
  Het gebruik van bestaande code voor CSS kan nuttig zijn, vooral voor mobile first

development. Dit zijn enkele voorbeelden van populaire CSS frameworks:

  - Bulma https://bulma.io/
  - Bootstrap https://getbootstrap.com/
  - Material https://material.io/

Databank

- PostgreSQL
  Het gebruik van PostgreSQL is verplicht.

Database Design

- DBdiagram https://dbdiagram.io/
  Online tool voor het tekenen van ER-diagrammen.
- DBdesigner https://www.dbdesigner.net/
  Idem.

API

- JSON Web Tokens https://jwt.io/
  Standaard voor token based claims. Kan gebruikt worden voor stateless authenticatie.
- Apiary https://apiary.io/
  Online tool voor het documenteren (en testen) van APIs.

Data processing

- pandas https://pandas.pydata.org/
  Python library voor het behandelen van tabulaire data.
- numpy https://numpy.org/
  Python library voor grootschalige numerieke operaties.
- scikit-learn https://scikit-learn.org/
  Python library met data science en machine learning functionaliteit.

Teamwork & Planning

- Git
  Het gebruik van een versiecontrolesysteem is verplicht. Vaak bieden deze services ook o.a. *projects* en *issues* aan voor planning en organisatie. Zodra een repository is aangemaakt, geef je `@JoeyDP` ook read access. Kies uit volgende drie services:

  - GitHub https://github.com/
  - Bitbucket https://bitbucket.org/
  - Gitlab https://gitlab.com/
- Miro https://miro.com/
  Miro is een "all-in-one workspace" die gebruikt kan worden voor plannen, organiseren en structureren. Een handige tool voor het bespreken en uitwerken van ideeën.

- Jira https://www.atlassian.com/software/jira
  Agile development tool met ondersteuning voor backlog, sprints, roadmaps en kanban boards. Het gebruik van Jira is verplicht (met gratis licentie).

3 Praktisch

3.1 Teams

Dit project dient uitgevoerd te worden in teams van 5 of 6 studenten. De teams werden op voorhand door ons willekeurig samengesteld. Deze kunnen teruggevonden worden op Blackboard.

3.2 Sprints

Om het project te managen, houden we ons aan de principes van Agile software development. Tijdens de eerste les wordt hier een uitgebreide introductie over gegeven. Vervolgens start jullie eerste *sprint*. Een sprint is een periode van 2–3 weken waarin je aan een specifiek doel werkt. Annick De Bruyn begeleidt jullie hierbij en zal telkens aan het einde/begin van een sprint een uur met jullie de zogenaamde *scrum meeting* houden. Deze gaan door in het vergaderlokaal M.G.023 naast het leslokaal (M.G.026).

Aangezien er maximum 4 meetings gehouden kunnen worden per week, hebben we de teams opgedeeld in twee groepen: A en B en verspreid over de weken. Teams met een oneven nummer zitten in groep A en teams met een even nummer in groep B:

Groep A: Teams 1, 3, 5 en 7
Groep B: Teams 2, 4, 6 en 8

Zie sectie 3.7: Planning voor de planning over wanneer jullie verwacht worden. Deelname in persoon aan deze meetings is verplicht voor alle teamleden en dit wordt in rekening gebracht voor de eindscore.

Naast deze uitgebreide meeting aan het einde van een sprint, is het natuurlijk ook aangeraden om tussendoor samen te komen om dingen te bespreken en samen te werken. We raden jullie aan om onderling af te spreken wanneer jullie aan het project willen werken en om op die momenten te beginnen met een korte meeting genaamd: *daily* of *stand up*.

Een handige tool om het overzicht te bewaren en de agile principes toe te passen is Jira. Per team moeten jullie hier een project aanmaken en ons uitnodigen zodat wij dit ook van nabij kunnen opvolgen. Voeg de volgende gebruikers toe:

- annick.debruyn@hotmail.com
- joey.depauw@uantwerpen.be
- bart.goethals@uantwerpen.be

Bij het uitlijnen van de doelen van de sprint, moet ook rekening gehouden worden met de *Definition of Done (DOD)*. Dit is een checklist van voorwaarden waaraan alle features moeten voldoen voor deze volledig afgewerkt beschouwd kan worden. Jullie spreken samen af wat de DOD exact inhoudt, en deze moet minstens het volgende bevatten:

- Code werkt (geen bugs).
- Code is getest (automatische testen of manueel met gedocumenteerd scenario van stappen).
- Documentatie is aangevuld (niet enkel comments in code, ook rapport).
- Feature werd gereviewd door een teamlid.

3.3 Discord

Als aanvulling op persoonlijk samenkomen in een klaslokaal, kunnen afspraken doorgaan op Discord. Speciaal voor dit vak, werd een virtueel klaslokaal ingericht als alternatief. Om toegang te krijgen, gebruik je de volgende invite link: `https://discord.gg/vmQwYfNj4Y`. Deze link brengt je naar de welkom pagina van de server waar je meer praktische informatie vindt over hoe je gebruik kan maken van dit virtueel klaslokaal.

Het is aangeraden om je te registreren voor een permanent account op Discord en om de desktopapplicatie te installeren. Zo hoef je niet telkens opnieuw gebruik te maken van de invite link en kan je vlotter inloggen.

3.4 Template

Om jullie alvast op weg te helpen, is er een template web applicatie te vinden op Blackboard. Begin met deze code lokaal werkende te krijgen voor elk teamlid apart (a.d.h.v. de tutorial in de README). Daarna kunnen jullie samen een versie op de productie server zetten (zie Sectie 3.5: Hosting).

3.5 Hosting

Exclusief voor dit vak wordt een budget op het Google Cloud Platform voorzien via het educational program van Google. Jullie kunnen dit bedrag gebruiken om per team een server te huren gedurende de periode van het vak. Het is uiteraard niet toegestaan om dit krediet voor persoonlijke doeleinden te gebruiken!

Hoe je precies aan de slag gaat met het Google Cloud Platform om een webapplicatie te hosten, zal worden uitgelegd tijdens de tweede praktijkles (22/02/2023). Om dit voor te bereiden moet één iemand van het team zijn of haar gmail email adres hebben doorgegeven via Blackboard. Dit is nodig om jullie te koppelen aan het educational credit.

Tegen het einde van de eerste sprint moet (minstens) de gegeven webapplicatie template werken op de server. Deze template kan incrementeel uitgebreid worden om de opgave uit te voeren. Maak gebruik van een git repository (Github/Gitlab/Bitbucket) waarvan de *production* of *master* branch op de server staat. Er wordt bovendien ook een DNS naam gekoppeld aan jullie server in de vorm van: `team[x].ua-ppdb.me`.

Er moet ten alle tijden een werkende versie van jullie systeem draaien op de server. Daarom is het belangrijk dat elk teamlid lokaal een development environment voorziet met een lokale test databank. Zo kan je vlot en onafhankelijk nieuwe features implementeren zonder de productie versie plat te leggen.

De geïnteresseerden kunnen gebruik maken van Continuous Integration and Continuous Delivery tools zoals Jenkins om dit proces te automatiseren.

3.6 Rapporteren & Presenteren

De evaluatie gebeurt aan de hand van drie (tussentijdse) *rapporten* en *demo's*. Onderschat het belang van rapporteren, documenteren en presenteren niet! Functionaliteit waarvan we niet weten dat ze bestaat, kan niet beoordeeld worden. Daarnaast is een cool idee niets waard als het niet grondig uitgelegd en beargumenteerd is.

Rapport

Technische documentatie is een deel van het eindproduct. Daarin staat het design van het programma als geheel, een database diagramma met uitleg en een beschrijving van de functionaliteit. Samengevat dus alle technische informatie die nodig is bij het opgeleverde systeem en alle relevante informatie die nodig is voor de klant om het product te integreren in het eigen platform.

Opgelet: Zorg dat elk teamlid bij de implementatie betrokken wordt, en niet enkel bij het projectbeheer of het maken van documentatie.

Presentatie

Voor de presentaties verwachten we een werkende demonstratie, waarbij jullie feedback krijgen van de jury. Een groot deel van de punten zal gebaseerd zijn op het al dan niet werken van de vereiste functionaliteit. Tijdens het semester worden twee tussentijdse presentaties georganiseerd, gevolgd door een eindpresentatie tijdens de examenperiode van een online beschikbare webapplicatie.

Voor een demo gebruiken jullie je eigen laptop(s), dus we raden sterk aan dat jullie op voorhand alles grondig controleren (internet verbinding, connectie met projector, etc.) opdat de demo vlekkeloos verloopt. Voorzie voldoende data en bereid op voorhand een use case voor. Waar het rapport dient voor technische informatie, willen we bij de demo vooral functionaliteit zien. Gebruik de presentatie vooral om de features te demonstreren waar jullie trots op zijn. Slides zijn in principe niet nodig, zolang de demonstratie duidelijk is.

Opgelet: Zorg er voor dat iedereen aan bod komt bij de presentaties.

3.7 Planning

Een overzicht van alle belangrijke datums is gegeven in Tabel 1 (weken lopen van woensdag tot woensdag). Tenzij anders vermeld, is het uur voor deadlines 23u59.

Tabel 1: Overzicht van planning en deadlines (rood†).

Week	Datum	Deadline/Planning
1	15/02/2023	Opdracht, scrum theorie en planning Sprint 1 – teams A+B
	19/02/2023†	Gmail adres Blackboard
2	22/02/2023	Introductie Google Cloud Platform
3	01/03/2023	Sprint 2 – teams A
4	08/03/2023	Sprint 2 – teams B
5	15/03/2023	Sprint 3 – teams A + milestone 1
6	22/03/2023	Sprint 3 – teams B + milestone 1
7	29/03/2023	Sprint 4 – teams A
8 9		Paasvakantie
10	19/04/2023	Sprint 4 – teams B
11	26/04/2023	Sprint 5 – teams A + milestone 2
12	03/05/2023	Sprint 5 – teams B + milestone 2
13	10/05/2023	Sprint 6 – teams A
14	17/05/2023	Sprint 6 – teams B
16-	Examen	Finaal rapport (Blackboard, datum afhankelijk van examen) Finale presentatie (Zie examenplanning voor datum)

Voor de *eerste milestone* verwachten we dat alle keuzes gemaakt zijn rond het ontwerp, de database en de taakverdeling. Voorzie mockups (pagina's zonder achterliggende koppeling met de databank of tekeningen) voor pagina's die nog niet geïmplementeerd zijn.

Voor de *tweede milestone* worden jullie geëvalueerd op basis van een online beschikbare demo van de applicatie, die voldoet aan al de basisvereisten.

Bij de *finale presentatie* verwachten we een live versie van het volledige afgewerkte geheel, met eventueel eigen uitbreidingen.

3.8 Evaluatie

Naast de functionaliteit worden jullie ook beoordeeld op de volgende criteria:

- Teamwork en planning. (4)
- Kwaliteit van de rapporten en presentaties. (2)
- Kwaliteit van de programma-code. (4)
- Kwaliteit van het databank ontwerp en de SQL queries. (3)
- Kwaliteit, originaliteit en nuttigheid van de opgeleverde features. (3)
- Kwaliteit van de demonstraties. (2)
- Gebruiksvriendelijkheid van de applicatie. (2)

Hoewel het vak "Programming Project Databases" heet, ligt de focus zeker niet enkel op het database aspect, maar ook op het programmeren van een groot project en het onafhankelijk kunnen samenwerken in team.

Bij een eerlijke taakverdeling, en als iedereen in staat is zijn deel te presenteren en vragen te beantwoorden, krijgen alle leden van de groep dezelfde punten.

3.9 Contact

Elke woensdag van **9u tot 13u** is lokaal **M.G.026** gereserveerd om samen te komen voor vergaderingen of om te werken aan het project. Daarnaast is ook altijd mogelijk om af te spreken in het virtuele klaslokaal op Discord.

De praktijkassistent is aanwezig (op Discord en in persoon) om specifieke vragen en problemen bij de uitvoering van het project te bespreken. Bij dringende vragen over het project, persoonlijke problemen tussen teamleden of bij technische problemen, kan je contact opnemen via email: joey.depauw@uantwerpen.be.

Veel succes!