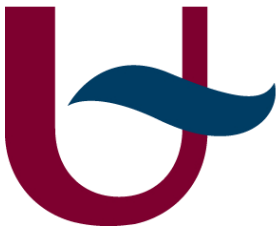

Programming Project Databases

Opgave 2020–2021

Deel 1

Recommendation Explorer

Docent:
Bart Goethals

Begeleider:
Joey De Pauw

10 februari 2021

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.
- Een software systeem in productie zetten en draaiende houden.

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 een rapport en presentatie, rekening houdend met bovenstaande doelstellingen.

De opgave bestaat uit twee delen: dit deel bevat sectie 2 die de basisvereisten van de opdracht van dit jaar uitlegt en sectie 3 waar alle praktische aangelegenheden zoals het vormen van teams en het verloop van evaluaties uitgelegd worden. Daarnaast wordt in de loop van het semester een uitbreiding op de opdracht geven in een ander document genaamd "Opgave deel 2".

2 Opdracht

De opdracht bestaat erin een webapplicatie te bouwen die het onderzoeken, evalueren en vergelijken van recommendation algoritmes faciliteert. Meer specifiek, focussen we ons op de subklasse van implicit feedback collaborative filtering recommendation algoritmes.

Het idee van deze algoritmes is simpel: we willen een gebruiker aanbevelingen (recommendations) geven op basis van zijn geschiedenis van bekeken items (implicit feedback) gebruikmakende van de geschiedenis van andere gelijkaardige gebruikers (collaborative filtering). Je vertrekt dus van een vector per gebruiker die nullen en enen bevat, naargelang zijn/haar interacties met bepaalde items. Op basis van deze vectoren moet een geordende lijst van items berekend worden die de gebruiker mogelijks ook interesseren. Wanneer we deze user-vectoren samenvoegen als de rijen in een matrix, krijgen we X, zoals hier afgebeeld:

$$X = \begin{array}{c} \begin{matrix} A & B & C & D \end{matrix} \\ \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{matrix}$$

In dit voorbeeld zien we de interacties van zes users ($U_{1..6}$) met vier items (A, B, C en D). Hoewel er voor gebruiker U_1 een nul staat voor item B, wilt dit eigenlijk zeggen dat niet geweten is of dit item relevant is voor de gebruiker. Het doel van de recommendation algoritmes die we willen bestuderen is dus om voor elke nul in de matrix te berekenen of dit effectief een nul moet zijn of mogelijks toch relevant is voor de gebruiker.

Een reeks verschillende recommendation algoritmes wordt reeds voorzien. Het is dus niet nodig om je hierin te verdiepen vooraleer je kan beginnen aan het project. Het doel van dit project is om een "recommendation explorer" te maken. Dit houdt in dat, gegeven een dataset, een scenario en een algoritme, jullie interessante en interactieve visualisaties moeten maken om de berekende recommendations te verkennen.

Secties 2.1 en 2.2 leggen de basisvereisten van de applicatie uit in termen van entiteiten en functionaliteit. Sectie 2.3 geeft een lijst van technologieën die gebruikt kunnen worden en sectie 2.4 beschrijft een dataset die als voorbeeld gegeven wordt. Tenslotte geeft sectie 2.5 een idee van wat het tweede deel van de opgave kan inhouden.

2.1 Entiteiten

De volgende entiteiten moeten minimaal herkenbaar zijn in jullie applicatie.

Gebruiker
: Niet te verwarren met een gebruiker van het recommendation algoritme: deze gebruiker is een data scientist die gebruik maakt van jullie applicatie. Een gebruiker kan inloggen in het systeem en gebruik maken van de services zoals datasets toevoegen, scenario's aanpassen en experimenten delen. Daarnaast wordt voor elke gebruiker een minimum aan persoonlijke informatie opgeslagen zoals: naam, gebruikersnaam en email adres.

Dataset
: Een dataset voor deze applicatie bestaat uit twee aspecten. Enerzijds is er de binaire interactie matrix X zoals eerder uitgelegd. Deze moet kunnen geüpload worden als CSV bestand met maximaal drie kolommen: *user_id*, *item_id* en optioneel *timestamp*. De eerste twee kolommen duiden interacties aan tussen users en items en de laatste optionele kolom geeft het moment van de interactie aan.

  Anderzijds is er item metadata die gekoppeld moet kunnen worden aan de dataset. Deze moet ook als een (of meerdere) CSV file(s) geüpload kunnen worden die minstens de kolom *item_id* bevat(ten). Verschillende soorten datatypes moeten ondersteund worden zoals numeriek, string en image (als url).

  Datasets zijn persoonlijk per gebruiker, maar kunnen wel gedeeld worden. Daarnaast moet de data van een dataset ook in de database opgeslagen worden voor efficiënte toegang en indexatie.

Scenario
: Scenario's kunnen gezien worden als een collectie van preprocessing en postprocessing stappen die respectievelijk toegepast worden op de dataset en de recommendations. Zo wil je bijvoorbeeld vergelijken wat het effect is als je enkel de meest recente interacties gebruikt voor het model of welke recommendations er gegeven worden wanneer item retargeting al dan niet wordt toegestaan (items met waarde 1 opnieuw aanbevelen).

Gebruikers kunnen scenario's definiëren op een dataset door verschillende simpele transformaties toe te voegen. Deze transformaties kunnen parameters hebben, zoals bijvoorbeeld: *filter interacties* $> timestamp$.

Model

Een model in de context van machine learning beschrijft de gewichten van een functie die inputs op outputs mapt. Het kan gaan van een lineair model $y = ax + b$ dat elke input x op y mapt met de gewichten a en b of een complexer model dat een user vector kan mappen op recommendations. De gewichten van dergelijke modellen worden geleerd door een algoritme op basis van de data. Een model hangt dus af van het scenario, het algoritme en de parameters van het algoritme.

Experiment

Wanneer we een reeks modellen samen nemen en bestuderen, dan spreken we van een experiment. Bijvoorbeeld wanneer je twee algoritmes wilt vergelijken op een bepaald scenario of wanneer je de invloed van retargeting wilt inspecteren voor hetzelfde algoritme.

Naast modellen, kan een experiment ook views, plots of andere visualisaties bevatten die opgeslagen moeten worden. Je kan experimenten dus ook zien als een soort van "workspace". Ten slotte moeten experimenten gedeeld kunnen worden met andere gebruikers, die vervolgens dingen kunnen toevoegen en veranderen aan het experiment.

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.

2.2 Basisfunctionaliteit

Deze sectie beschrijft alle basisvereisten in termen van functionaliteit. Het is verplicht voor elk team om minimaal deze functionaliteit te implementeren en demonstreren tijdens de evaluatie om te slagen voor het vak. 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. De basisvereisten moeten afgewerkt zijn tegen de tweede tussentijdse evaluatie (zie Sectie 3.6: Planning).

Registratie & Login

Gebruikers kunnen minstens inloggen met wachtwoord. Andere meer uitgebreide mechanismes (email verificatie, login met Facebook/Google/...) zijn toegestaan, maar niet verplicht. Denk goed na over wie toegang moet hebben tot welke informatie. Datasets kunnen gevoelige informatie bevatten en mogen enkel toegankelijk zijn voor de eigenaar en gebruikers met wie deze gedeeld zijn.

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).

Datasets

Om custom datasets te ondersteunen in de applicatie, moeten volgende stappen mogelijk zijn:

- Een naam geven aan de dataset.
- Uploaden van een CSV file voor de interacties.
- Item metadata inlezen van mogelijks meerdere CSV files.
- Datatypes van item metadata kunnen instellen (minstens: numerical, string, url as image).
- Functionele informatie aan kolommen toekennen (*user_id*, *item_id*, *timestamp*, *primary_identifier*, ...). Deze informatie is nodig om de data correct te interpreteren en weer te geven.
- Sample van metadata tonen.
- Sample van interacties tonen.
- Samenvattende statistieken van dataset tonen (aantal items, users en interacties).
- Datasets delen met andere gebruikers. Het delen moet ook beëindigd kunnen worden.

Datasets kunnen niet aangepast worden eens aangemaakt aangezien dit alle scenario's en modellen ongeldig zou maken.

Scenarios

Scenario's beschrijven de pre- en postprocesing stappen op de data. Voorzie een intuïtieve interface om het volgende te doen:

- Scenario's aanmaken en een naam geven.
- Stappen toevoegen aan een scenario.
- Scenario's "kopiëren" en aanpassen (om vlot variaties te kunnen aanmaken).
- Gelijkaardige weergaves tonen als voor dataset (een scenario definieert een subset van de dataset).

Eens een scenario is aangemaakt, kunnen dingen die de data zouden veranderen niet meer aangepast worden. Het moet nog wel mogelijk zijn om de naam aan te passen bijvoorbeeld.

Preprocessing Stappen

Als onderdeel van een scenario, moeten de volgende preprocessing stappen ondersteund worden:

- Interacties filteren op timestamp (parameter t)
- Users filteren op minimale of maximale activiteit (parameter $interaction$). Bijvoorbeeld alle users verwijderen met minder dan 5 items in hun history.
- Items filteren op minimale of maximale activiteit (parameter $interaction$).

Postprocessing Stappen

Implementeer als postprocessing stap een filter voor retargetting. Deze moet er voor zorgen dat items uit de user history niet opnieuw aanbevolen kunnen worden.

Model aanmaken

Een model wordt aangemaakt op basis van een scenario, een algoritme en de parameters van het algoritme. Eens de selectie is aangemaakt, kan het model getraind worden door het algoritme. Voor deze opdracht worden de datasets klein gehouden en de algoritmes simpel waardoor dit niet enorm lang zou mogen duren, maar wel te lang om op de main thread te berekenen. Gebruik dus beter een proces in de achtergrond om te voorkomen dat jullie webapplicatie bevriest. Houd voor elk model ook interessante informatie bij zoals hoelang het trainen duurde en hoeveel geheugen het model inneemt (in mega/gigabytes on disk).

Voorbeeld implementaties van algoritmes en hun parameters worden voorzien in de loop van het vak.

Experiment

Een experiment dient om modellen te onderzoeken, evalueren en vergelijken. Voor de basisvereisten mag dit echter nog beperkt blijven tot het onderzoeken van één enkel model. Implementeer een visualisatie die voor arbitraire users zowel hun geschiedenis kan tonen als hun top-k recommendations. Het idee is om inzicht te krijgen in het model door te bekijken welke items aanbevolen worden aan welk "soort" users op basis van hun geschiedenis. Daarnaast kunnen verschillende users vergeleken worden door deze toe te voegen aan de visualisatie.

Users moeten op de volgende manieren toegevoegd kunnen worden:

- Lege user.
- Random user uit dataset.
- User met random items (parameter $amount$).
- Alle users met item x in hun geschiedenis.
- Als kopie van een user in de lijst.

Daarnaast moeten items ook verwijderd kunnen worden uit de geschiedenis van een gebruiker (enkel voor de visualisatie, niet effectief uit de dataset) of toegevoegd kunnen worden (met een zoekfunctie of vanuit de recommendations).

Het visualiseren van items zal ook afhangen van de dataset. De functionele informatie kan gebruikt worden om bijvoorbeeld de *primary_identifier* meer prominent te tonen of een property met tag *image* kan aangeven welke image getoond moet worden voor een item etc. Daarnaast kan ook berekende informatie getoond worden voor elk item zoals de eerste interactie of de populariteit (percentage users). Zorg er voor dat niet alle item metadata standaard getoond wordt, maar dat deze aan/uit gezet kan worden vanuit de webapplicatie.

Tenslotte moet een experiment gedeeld kunnen worden en gebruikers met wie een experiment gedeeld is, moeten dit kunnen aanpassen. Wanneer je een experiment deelt, deel je ook de onderliggende dataset automatisch.

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 dataset verwijderd wordt, dan moeten de geassocieerde scenario's en modellen 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 2.3: Technologie.

Merk op dat gezien de aard van deze applicatie, het toegestaan is om voor sommige pagina's een uitzondering te maken op het mobile first design principe. Bijvoorbeeld voor visualisaties van experimenten die niet altijd triviaal op een klein scherm passen.

Daarnaast is het aangeraden 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. Dit kan bijvoorbeeld nuttig zijn voor het aanmaken van scenario's en experimenten, waar je als gebruiker niet voor elke klik op een knop de hele pagina opnieuw wilt laden.

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.

2.3 Technologie

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

Tegen **17/02/2021** moet er binnen jullie team beslist zijn over de belangrijkste opties. Gebruik dus de 1ste lesweek om enkele opties met elkaar te vergelijken en binnen jullie team te bespreken hoe jullie het project willen aanpakken (zie Sectie 3.1: Teams). 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.
- jQuery https://jquery.com/
  Deze Javascript library wordt vaak gebruikt om simpelere code te schrijven en voor asynchrone calls (ajax).
- CSS Framework
  Het gebruik van bestaande code voor CSS kan nuttig zijn, vooral voor mobile first development. Wij raden de volgende frameworks aan:
  - 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.

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/
- Notion https://www.notion.so/
  Notion is een "all-in-one workspace" die gebruikt kan worden voor plannen, organiseren, structureren en zelfs documenteren. Een handige tool voor het coördineren en opvolgen van jullie werk.
- Trello https://trello.com/
  Online tool voor Kanban boards.

2.4 Data

Een subset van de Movielens20M dataset werd gecreëerd voor dit project. Deze dataset bevat rating data van 138.000 gebruikers over 27.000 films en series. Hier werd een subset van genomen door enkel ratings van 4 of hoger te zien als een positieve interactie en enkel items bij te houden met 20 of meer interacties en users met 5 of meer interacties. Daarnaast werd metadata toegevoegd op basis van The Movie Database API zodat er ook posters gekoppeld zijn aan de films en series.

Het resultaat is een interactie dataset met 1.058.027 interacties tussen 18.391 users en 4.284 items en twee item metadata files die respectievelijk titels en genres bevatten en urls, images en omschrijvingen van de items.

Hoewel het de bedoeling is dat jullie applicatie werkt voor eender welk soort data, wordt enkel deze dataset gegeven. Het is altijd mogelijk om andere datasets te downloaden of zelf een mini dataset aan te maken met een klein aantal manueel ingevulde users bijvoorbeeld.

2.5 Deel 2

Mogelijke uitbreidingen worden gegeven in deel twee van de opgave. Het doel van uitbreidingen is om jullie applicatie met standaard functionaliteit te onderscheiden van de andere groepen. Wij geven één verplichte uitbreiding en enkele suggesties die voldoende uitdagend zijn op technisch vlak, maar het liefst van al zien we nieuwe ideeën.

Deel twee van de opgave wordt online gezet na de eerste tussentijdse evaluatie (zie Sectie 3.6: Planning).

3 Praktisch

3.1 Teams

Dit project dient uitgevoerd te worden in teams van 4 tot 5 studenten. Jullie stellen zelf de teams samen via Blackboard. Onder de sectie "teams" schrijf je je in bij één van de acht teams. Let op het aantal toegestane studenten per team:

- Teams 1 - 6: plaats voor 5 studenten.
- Teams 7 - 8: plaats voor 4 studenten.

De deadline om je in te schrijven voor een groep is **12/02/2021** om 14u00. Studenten die na deze deadline nog geen team hebben, zullen door ons ingedeeld worden. Het is dus mogelijk dat de bestaande teams nog een extra lid krijgen.

We verwachten per team per week een coördinator die iets meer verantwoordelijkheid draagt. De eerste 5 (of 4) weken roteert deze rol zodat ieder teamlid exact 1 keer coördinator is geweest. Daarna kan binnen het team zelf beslist worden wie deze rol vervult. De taken van de coördinator zijn als volgt:

- Het overzicht bewaren van het project, weten waar alle teamleden mee bezig zijn en zorgen dat iedereen op z'n minst weet hoe hij/zij zich nuttig kan inzetten. Communicatie en coördinatie zijn niet te onderschatten bij grote groepswerken.
- Opvolgen van deadlines (van de opgave zowel als jullie eigen afgesproken deadlines).
- Het faciliteren van de vergadering(en).
- Doorsturen van het wekelijks verslag (zie sectie 3.5: Rapporteren & Presenteren).
- De na-wekelijkse opvolgingsmeeting bijwonen met coördinatoren van alle teams. Elke **woensdag om 10:30u** komen de coördinatoren van de **voorbije** week samen met de assistent om de vooruitgang binnen hun team te bespreken. Het is aangeraden dat je als coördinator op voorhand kort overlegt met je team om voldoende op de hoogte te zijn van waar jullie staan. De eerste na-wekelijkse opvolgingsmeeting vindt plaats op 24/02/2021. De meeting kan zowel op de campus (M.G.023) als online (Discord) gevolgd worden.

Verder zijn jullie vrij om andere rollen en verantwoordelijkheden te bepalen binnen het team. Let er wel op dat de taakverdeling gebalanceerd blijft. Het is bijvoorbeeld niet de bedoeling dat iemand alle HTML pagina's maakt, iemand anders alle verslagen schrijft en nog iemand anders enkel SQL queries schrijft bijvoorbeeld. Het is zinvoller om goed afgelijnde componenten te verdelen over het team.

Opgelet: Coördinator zijn levert niet, op zichzelf, meer punten op.

3.2 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.com/invite/BBeNeYb6h4`. 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.

3.3 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 3.4: Hosting).

3.4 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 (17/02/2021). Tegen dan zijn de teams gevormd en 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 **23/02/2021** moeten jullie het gegeven webapplicatie template laten werken op de server. Deze template kan incrementeel uitgebreid worden om de opgave uit te voeren. Gebruik 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 (op een andere branch) en 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.

3.5 Rapporteren & Presenteren

De evaluatie gebeurt aan de hand van *wekelijkse verslagen* en drie (tussentijdse) *rapporten* en *presentaties*. 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.

Wekelijks Verslag

We verwachten dat jullie elke lesweek een vergadering houden waarbij de status en de planning van jullie project wordt besproken. Het resultaat hiervan is een wekelijks verslag met aanwezigheden, opvolging van geplande taken, etc. Baseer jullie verslag op het gegeven template op Blackboard (de elementen in deze template moeten minimaal aanwezig zijn). Stuur dit verslag **wekelijks op woensdag voor 23u59** door als PDF via email naar joey.depauw@uantwerpen.be met onderwerp: *"PPDB 2020-2021: Wekelijks Verslag Team [X]"*. Zorg dat het nummer van jullie team zeker correct in het onderwerp vermeld staat (de hyperlink van het email adres vult al automatisch de meeste informatie in).

Het is perfect mogelijk dat er een week minder of zelfs niet aan het project gewerkt is. Jullie zijn uiteindelijk zelf verantwoordelijk om het project tot een goed einde te brengen. Wanneer

dit het geval is, vermeld het dan kort in de template en dien het verslag zo in.

Rapport

Naast de wekelijkste verslagen moet drie keer een rapport worden ingediend via Blackboard. Zie Sectie 3.6: Planning voor de exacte deadlines. We verwachten onder andere het design van het programma als geheel, een ER-diagram van de databank en een beschrijving van de functionaliteit, alsook een overzicht van de afgewerkte taken van elk teamlid. Samengevat dus alle technische informatie die nodig is bij het opgeleverde systeem en alle relevante informatie over hoe deze tot stand is gekomen.

Opgelet: Zorg dat elk teamlid bij de implementatie betrokken moet zijn, 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.

3.6 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. Naast deze data, moet ook wekelijks een verslag ingediend worden op woensdag.

Voor de eerste tussentijdse presentatie 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. Zorg dat jullie beslissingen, ideeën en de planning duidelijk aanwezig zijn in het rapport en de presentatie.

Voor de tweede tussentijdse presentatie 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 eigen uitbreidingen (zie deel twee van de opgave).

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

Week	Datum	Deadline/Planning
1	12/02/2021†	Teams vormen (voor 14u00)
	16/02/2021†	gmail adres invullen op Blackboard
2	17/02/2021	Introductie Google Cloud Platform
	17/02/2021†	Selectie Technologiën
	23/02/2021†	Hello World op Google Cloud Platform
3	24/02/2021	Eerste na-wekelijkse opvolgingsmeeting
5	07/03/2021†	Eerste tussentijds rapport (Blackboard)
	10/03/2021	Eerste tussentijdse presentatie
11	18/04/2021†	Tweede tussentijds rapport (Blackboard)
	21/04/2021	Tweede tussentijdse presentatie
16-	Examen -	Finaal rapport (Blackboard, datum afhankelijk van examen)
	Periode	Finale presentatie (Zie examenplanning voor datum)

3.7 Evaluatie

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

- Teamwork en planning. (2)
- Kwaliteit van de wekelijkse verslaggeving, rapporten en presentaties. (2)
- Kwaliteit van de programma-code. (4)
- Kwaliteit van het databank ontwerp en de SQL queries. (4)
- Kwaliteit, originaliteit en nuttigheid van de opgeleverde features. (4)
- Kwaliteit van de online 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.

3.8 Contact

Elke woensdag van 9u tot 13u is lokaal M.G.025 gereserveerd om samen te komen voor vergaderingen of om te werken aan het project. Om grote drukte te vermijden, wordt deze tijdsspanne in twee blokken opgedeeld: de eerste helft van **9u tot 11u** en de tweede helft van **11u tot 13u**, met tussendoor om 10:30u de voortgangsmeeting (dewelke ook online te volgen is). In beide helften is plaats voor 4 teams, zodat iedereen de kans krijgt om naar de campus te komen. Reserveren voor één van de twee helften is verplicht als je wilt komen!

Dit gebeurt op Discord via het `#reservations` kanaal. Daarnaast is het natuurlijk 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!