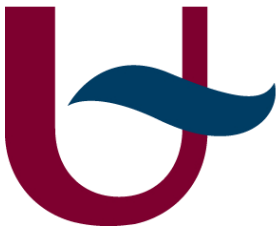


Programming Project Databases

Opgave 2019-2020

Deel 1

Campus Carpool

Docent:
Bart Goethals

Begeleiders:
Joey De Pauw
Len Feremans

12 februari 2020

1 Inleiding

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

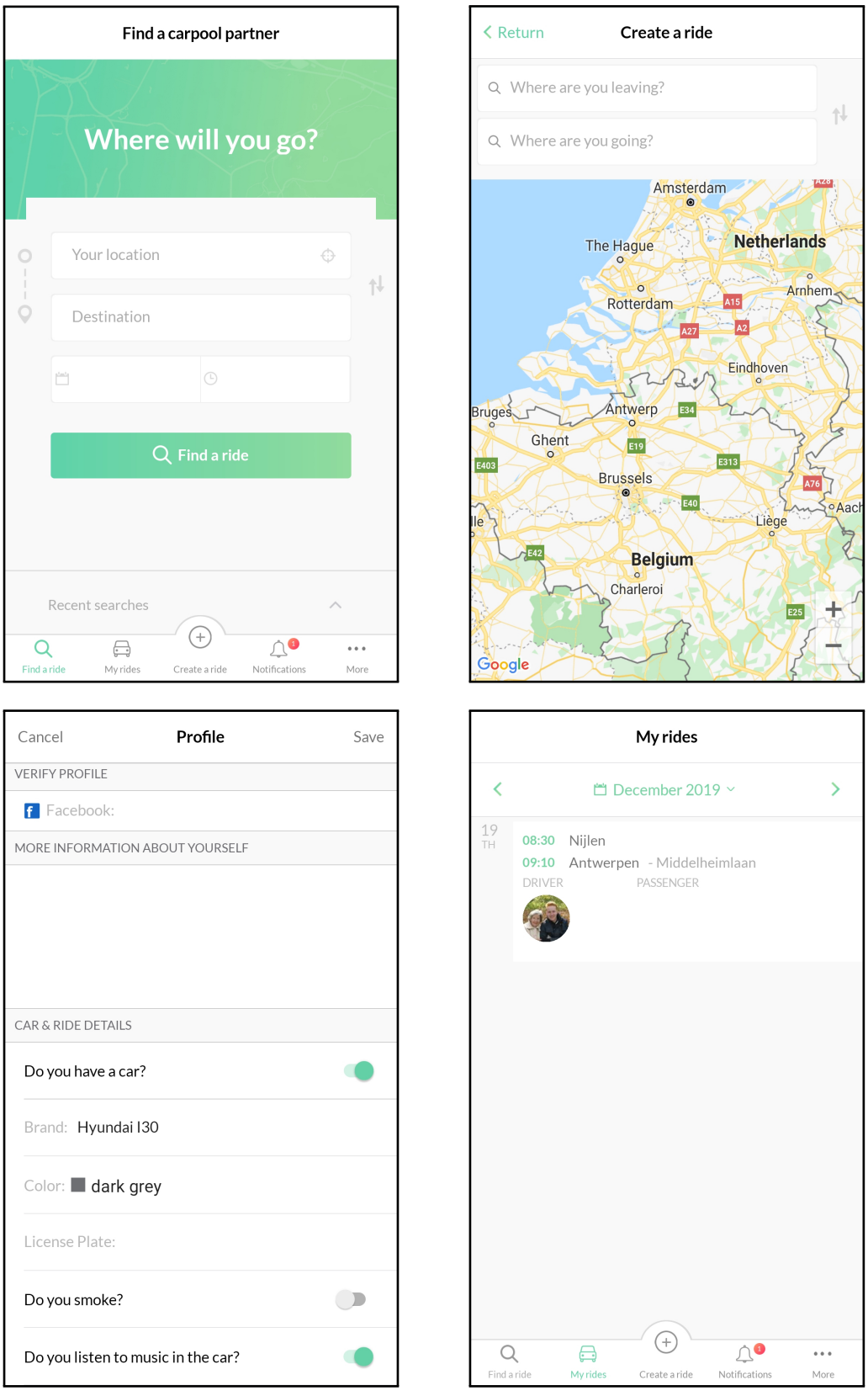
2 Opgave

De opgave bestaat erin een webapplicatie te bouwen die carpoolen faciliteert. Er bestaan reeds enkele concurrenten op de markt zoals:

- `https://campus.carpool.be` (Zie voorbeeld Figuur 1)
- `https://www.nl.blablacar.be/`
- `https://www.waze.com/nl/carpool` (Niet beschikbaar in België)
- `https://www.uber.com/be/nl/` (Niet beschikbaar in België)

Een deel van de opgave is om jullie eigen applicatie te onderscheiden van de andere spelers op de markt. Of jullie dit doen door een heel specifiek publiek te targetten, een hoge graad van automatisatie en flexibiliteit te voorzien of door een creatieve draai te geven aan het klassieke carpool verhaal, staat jullie vrij. Wij willen vooral zien dat er als team nagedacht is over de features die geïmplementeerd worden en dat jullie alle gemaakte keuzes kunnen onderbouwen.

Secties 2.1 en 2.2 leggen de basisvereisten van de applicatie uit in termen van entiteiten en functionaliteit. Secties 2.3 en 2.4 geven een beginpunt in de vorm van een minimale template om van te vertrekken en een lijst van technologieën die gebruikt kunnen worden.



Figuur 1: Voorbeeld Carpool app (https://campus.carpool.be).

2.1 Entiteiten

De volgende entiteiten moeten minimaal herkenbaar zijn in jullie applicatie.

Gebruiker
: Een gebruiker kan inloggen in het systeem en gebruik maken van de services zoals ritten aanmaken of verzoeken om deel te nemen aan een rit. Daarnaast wordt voor elke gebruiker een minimum aan persoonlijke informatie opgeslagen: naam, email adres, leeftijd, geslacht, profielfoto, etc.

Auto
: Elke gebruiker kan optioneel een auto hebben. Zorg dat gebruikers de belangrijkste informatie van hun auto kunnen opslaan en aanpassen. Denk bijvoorbeeld aan kleur, merk, nummerplaat, zitplaatsen, verbruik, bouwjaar en brandstof. Als uitbreiding kan je meerdere auto's per gebruiker ondersteunen, maar dit is niet nodig voor de basisvereisten.

Rit
: Ritten kunnen worden aangemaakt door gebruikers (met of zonder auto). Een rit bevat de locatie en tijd van vertrek en aankomst. Het is de hoofdfunctie van de carpool applicatie om ritten met elkaar te linken en gebruikers tussen ritten te laten zoeken. Je kan het onderscheid maken tussen ritten die geadverteerd worden en ritten die hebben plaatsgevonden. Voeg eventueel ook de notie van marge/flexibiliteit op aankomsttijd en locatie toe.

Voorkeuren
: Gebruikers kunnen een voorkeur aangeven op vlak van auto en medereizigers. Schat zelf in welke filters zinvol zijn. Enkele goede voorbeelden zijn leeftijd en geslacht van de chauffeur en verbruik van de wagen.

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.

Ten eerste bouwen jullie een *website* die voldoet aan bepaalde minimale vereisten. Daarnaast moet een *REST API* geïmplementeerd worden voor programmatische toegang tot jullie applicatie. We verwachten dat deze basisvereisten afgewerkt zijn tegen de tweede tussentijdse evaluatie (zie Sectie 3.4: Planning).

Website

De hoofdinterface van jullie applicatie is de website. Zorg voor een gebruiksvriendelijke interface die er professioneel uitziet, zowel mobiel (op gsm/tablet) als op grotere schermen. Hiervoor gebruik je best het "mobile first design" idioom, 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.

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 welke informatie mag zien, welke informatie enkel zichtbaar mag zijn na inloggen en welke data publiek mag 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).

CRUD Operaties voor Entiteiten
: **C**reate, **R**ead, **U**pdate and **D**elete operaties moeten ondersteund zijn voor alle logische entiteiten. Bijvoorbeeld: gebruikers moeten een rit kunnen aanmaken, bekijken, aanpassen of verwijderen.

Rit Zoeken
: In het simpelste geval moeten gebruikers aan ritten kunnen deelnemen door deze zelf op te zoeken. Logische criteria zijn de aankomsttijd en vertrek en aankomst locatie. Houd rekening met een bepaalde marge (zowel op tijd als locatie) zodat niet enkel een exacte match teruggevonden wordt.

Geocoding
: Het vertalen van adressen naar coördinaten moet ondersteund zijn. Dit is nodig voor het berekenen van afstanden en/of routes tussen twee punten. Routes moeten niet in rekening gebracht worden voor de basisvereisten.

Visualisatie
: Bij het tonen van ritten, is een visualisatie op kaart essentieel. Zorg voor intuitieve en overzichtelijke visualisaties waar nuttig. Simpelweg coördinaten tonen in een tabel is niet nuttig voor de gebruiker en moet vermeden worden.

API

Naast de website, implementeren jullie ook een **A**pplication **P**rogramming **I**nterface (API) die voldoet aan het REST design principe. Dit houdt o.a. in dat de API geen state bijhoudt en dat de entiteiten centraal staan (niet de operaties).

De publieke API heeft minstens de volgende functionaliteit:

Registratie & Login
: Implementeer een token-based login systeem voor de REST API.

CRUD Operaties voor Ritten
: **C**reate, **R**ead, **U**pdate and **D**elete operaties moeten ondersteund zijn voor ritten in de API.

Rit Zoeken
: Ondersteun ook een zoekfunctie voor ritten in de API. Zorg er voor dat deze functie beschikbaar is zonder eerst te moeten inloggen.

Een minimale specificatie van de API is vastgezet en hier beschreven: https://ppdb.docs.apiary.io/. Wanneer minimaal *register*, *auth* en *create drive* geïmplementeerd zijn, kan ja live data beginnen ontvangen op de server (zie Sectie 3.2: Hosting).

2.3 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 3.2: Hosting).

2.4 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 **19/02/2020** 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 sterk 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.
- *PostGIS* https://postgis.net/
  Gebruik de PostGIS extensie voor spatial queries en optimization.

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.

Map Service

- *Geopy* https://pypi.org/project/geopy/
  Geopy is een Python module die gebruikt kan worden in combinatie met een geocoding service voor het uitvoeren van geografische berekeningen.
- *Open Street Map (OSM)* https://www.openstreetmap.org/
  Service voor kaarten.
- *Open Source Routing Machine (OSRM)* http://project-osrm.org/
  Service voor routes.
- *GraphHopper* https://www.graphhopper.com/
  Service voor routes.
- *Openrouteservice* https://openrouteservice.org/
  Service voor routes.
- *Google Maps-platform* https://developers.google.com/maps/
  Service voor routes en kaarten.

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. Enkele voorbeelden zijn:
  - GitHub https://github.com/
  - Bitbucket https://bitbucket.org/
  - Gitlab https://gitlab.com/

- *Toggl Plan/Teamweek* https://toggl.com/plan/
  Een handige webapplicatie voor het plannen van taken. Toggl Plan (ook bekend als Teamweek) ondersteunt zowel een planning over tijd als een planning van taken in de vorm van een Kanban board. Samenwerking en taakverdeling zijn zeer intuïtief.
- *Trello* https://trello.com/
  Online tool voor Kanban boards.

2.5 Deel 2

Mogelijke uitbreidingen worden gegeven in deel twee van de opgave. Het doel van uitbreidingen is om jullie carpool applicatie met standaard functionaliteit te onderscheiden van de andere groepen en van de concurrenten op de markt. 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 3.4: 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 "groepen" schrijf je je in bij 1 van de 6 teams. Let op het aantal toegestane studenten per team:

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

De deadline om je in te schrijven voor een groep is **14/02/2020** 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 zelf afgesproken deadlines).
- Het faciliteren van de vergadering(en).
- Doorsturen van het wekelijks verslag (zie sectie 3.3: Rapporteren & Presenteren).
- De na-wekelijkse opvolgingsmeeting bijwonen met coördinatoren van alle teams. Elke **woensdag om 10u** komen de coördinatoren van de **voorbije** week samen met de

assistent om de vooruitgang binnen hun team te bespreken. De eerste na-wekelijkse opvolgingsmeeting vindt plaats op 19/02/2020.

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 Hosting

Exclusief voor dit vak wordt 50 USD credit op het Google Cloud Platform voorzien per student via het educational program van Google. Jullie kunnen dit bedrag gebruiken om een server te huren gedurende de periode van het vak. Zorg er zelf voor dat je toekomt met dit krediet. Het is bijvoorbeeld niet de bedoeling dat elk teamlid apart een server instance en databank aanmaakt.

Hoe je precies aan de slag gaat met het Google Cloud Platform om een webapplicatie te hosten, zal worden uitgelegd tijdens de 2de praktijkles (19/02/2020). Tegen dan wordt een link voorzien waarmee je het krediet kan claimen (1 registratie per student). Het is uiteraard niet toegestaan om dit krediet voor persoonlijke doeleinden te gebruiken!

Tegen **25/02/2020** 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. In jullie wekelijks verslag vermeld je het publieke IP adres van de server. Er wordt dan een DNS naam aan gekoppeld in de vorm van: `team[x].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.

3.3 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 2019-2020: 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 3.4: 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.

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 zullen er twee tussentijdse presentaties georganiseerd worden, 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 scenario 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.

3.4 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.

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

Week	Datum	Deadline/Planning
1	14/02/2020	Teams vormen (voor 14u00)
	19/02/2020	Selectie Technologiën
2	19/02/2020	Introductie Google Cloud Platform
	25/02/2020	Hello World op Google Cloud Platform
		Public IP van server in verslag
4	08/03/2020	Eerste tussentijds rapport (Blackboard)
	11/03/2020	Eerste tussentijdse presentatie
8	01/04/2020	Uitzonderlijk geen practicum, deadline naar vrijdag
	03/04/2020	Deadline wekelijks verslag (i.p.v. woensdag)
10	19/04/2020	Tweede tussentijds rapport (Blackboard)
	22/04/2020	Tweede tussentijdse presentatie
15-	Examen -	Finaal rapport (Blackboard, datum afhankelijk van examen)
	Periode	Finale presentatie (Zie examenplanning voor datum)

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).

3.5 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.

3.6 Contact

Lokaal M.G.026 is gereserveerd elke woensdag van 9u tot 15u. In deze periode mag het lokaal gebruikt worden om samen te komen, te vergaderen en te werken. Van 10u tot (vermoedelijk) 10u30 à 11u vindt de na-wekelijkste voortgangsmeeting plaats, daarna tot minstens 12u is een assistent aanwezig en kunnen specifieke vragen en problemen bij de uitvoering van het project besproken worden.

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!