



Snooty Snakes

06-01-2021

[Analyse project veeteelt management](#)

Dave Saenen, Wouter Pardon, Niels De Cat, Fabio Puissant

U Hasselt Martelarenlaan 42, 3500 Hasselt

Probleemstelling

Productieproces

Voor veeteelt houders is het moeilijk om data over hun bedrijf bij te houden op een georganiseerde manier. Zo zijn er bijvoorbeeld 5 cruciale parameters om de evolutie van een varkensbedrijf op te volgen. (Voedingspatroon, Groei (kg/dag), Kosten van het voer, Opname voedsel/dag, Kwaliteit vlees) Deze spelen een belangrijke rol voor het opvolgen van het productieproces.

Deze parameters worden per groep dieren opgevolgd, zodat veranderingen in groepen zichtbaar worden.

Om dit te verduidelijken gaan we uit van het volgend voorbeeld: een varkenshouder wil een nieuw voedsel type uittesten, dit type kan gevolgen hebben voor de groei van de varkens maar ook voor de kosten van het voer en de opname van voedsel per varken of per dag.

Aan de hand van dit voorbeeld wordt er duidelijk gemaakt dat er nood is aan een systeem om structuur in het productieproces te hebben, aangezien op het moment alles via spreadsheets bijgehouden wordt door de klant.


Doordat dit systeem deze data op een gestructureerde manier bijhoudt kan dit gevisualiseerd worden, om zo de varkenshouder een inzicht te geven in het productieproces van zijn bedrijf.

Rollen van het systeem

Het systeem moet zo eenvoudig te gebruiken dat de gebruiker een minimum aan tijd nodig heeft om acties te voltooien. Het systeem is immers een oplossing om het veehouder proces te vergemakkelijken. Een boer die werknemers in dienst heeft, moet in staat zijn deze werknemers gespecificeerde taken te kunnen laten volbrengen en dit te kunnen opvolgen in het systeem. Er zijn mogelijk managers, administratief bedienden en andere type werknemers. Een boerderij is immers ook een bedrijf!

Niet elke werknemer van de boerderij heeft dezelfde verantwoordelijkheden en moet dus begrensd zijn in de acties die hij/zij kan nemen binnen het systeem. Een werknemer die de wegingen doet mag bijvoorbeeld niet in staat zijn voeder aankopen in te geven.

Om de eenvoud en flexibiliteit te bewaren bij het toevoegen van werknemers van bepaalde afdelingen, moet de klant in staat zijn elke werknemer rechten te kunnen toekennen, aanpassen en verwijderen. Omwille van de vele mogelijkheden zal het systeem moeten



opgebouwd worden uit claims die rechten geven aan werknemers tot het systeemgebruik in plaats van statisch gedefinieerde type gebruikers (zoals administrator, gebruiker, manager, ...).

Agenda

Veeteelt houders kunnen soms ook een slecht overzicht hebben van hun werknemers als ze een groot bedrijf hebben. Ze steken te veel tijd in het zoeken van hun personeel en het uitdelen van taken wat elke dag opnieuw gebeurt. Ook als er wordt gewerkt met dagelijkse of wekelijkse werkroosters neemt dit ook tijd in beslag voor een boer (of administratieve medewerker) om zo'n rooster te maken en uit te delen evenals de tijd die het personeel gebruikt om al zijn/haar werkuren in dit rooster in te vullen en af te geven. Deze tijd kan beter worden benut! Ons systeem maakt gebruik van een agenda waar een gebruiker met de rechten "agendabeheerder" taken kan aanmaken en verwijderen/bevestigen - in ons systeem is een bevestiging geassocieerd met het verwijderen van een taak - in functie van groepen (dieren). Elke andere gebruiker met de rechten "werknemer" kan taken voltooien. Een taak kan aangemaakt worden op twee manieren:

- door een agendabeheerder die een gewone taak aanmaakt. Dit betekent dat de taak direct actief is (zichtbaar is in de kalender).
- door het systeem dat automatisch taken aanmaakt op basis van acties van gebruikers met een beheerdersrol (vb: als een manager een nieuwe groep dieren aanmaakt dan maakt het systeem een taak "nieuwe weging voor groep X").

Use cases

Hieronder vindt u een kleine legende van alle rollen in ons systeem.

Gebruiker
Werknemer
Administratieve medewerker
Manager
Agendabeheerder
Voedings-beheerder
Weger
Baas
Tijd

Inloggen

Brief use case

Actoren: Gebruiker

Beschrijving:

een gebruiker geeft zijn logingegevens aan het systeem. Het systeem controleert of deze gegevens correct zijn. De gebruiker wordt doorverwezen.

Casual use Case

Actoren: Gebruiker

Hoofdscenario:

een gebruiker geeft zijn logingegevens aan het systeem.[A] Het systeem controleert of deze gegevens correct zijn.[B] De gebruiker wordt doorverwezen.

Alternatief scenario A:

Een gebruiker geeft een foutief geformatteerd e-mailadres in. Het systeem zal dan met een melding duidelijk maken dat dit geen geldig e-mailadres is.

Alternatief scenario B:

Een gebruiker geeft een verkeerde combinatie in van e-mailadres en wachtwoord. Het systeem controleert deze gegevens en toont een melding op het scherm.

Fully dressed use case

Use Case : Inloggen

Actoren : Gebruiker, wilt inloggen en wilt op een duidelijke manier weten of zijn combinatie gebruikersnaam en wachtwoord geldig is.

Precondities en Postcondities:

precondities: /, postcondities: gebruiker wordt doorverwezen naar het systeem of krijgt een melding dat zijn e-mailadres en wachtwoord niet gevonden wordt in de database.

Beschrijving :**Main success flow**

Navigatie naar login page	Het systeem toont een formulier waarin de gebruiker zijn inloggegevens kan ingeven
Gebruiker geeft zijn E-mailadres en zijn wachtwoord op	Het systeem controleert of het adres aan de voorwaarde van een correct E-mailadres voldoet
De gebruiker drukt op enter of op de login button om het inloggen te voltooien	Het systeem controleert de combinatie van gebruikersnaam en wachtwoord. Nadien wordt de gebruiker doorverwezen naar het systeem.

Alternative flow

Gebruiker geeft een fout formaat van e-mailadres op	Het systeem toont een melding dat dit geen geldig e-mailadres is
---	--

Alternative flow

Gebruiker geeft een foute combinatie van	Het systeem toont een melding dat deze
--	--

e-mail en wachtwoord op

combinatie van e-mailadres en wachtwoord
niet bestaat in de databank

Rechten beheren

Brief use case

Actoren: Baas

Beschrijving:

De rechten van de gebruiker zijn beperkt tot de rechten die aan hem zijn toegekend. Alleen de baas kan rechten aanpassen van een andere gebruiker.

Casual use Case

Actoren: Gebruiker

Hoofdscenario:

Een gebruiker gaat een bewerking doen waarvoor hij gerechtigd is. De bewerking zal dan correct uitgevoerd kunnen worden.

Gebruikers beheren

Brief use case

Actoren: Baas

Beschrijving:

De baas navigeert naar de pagina waar hij gebruikers kan beheren. Op deze pagina kan hij de volgende operaties uitvoeren:

1. gebruikers bekijken
2. gebruikers toevoegen
3. gebruikers wijzigen
4. gebruikers verwijderen

let op: als een baas operaties 1 tot 3 uitvoert, worden ook rechten toegekend aan een gebruiker. Meerdere rechten kunnen worden toegekend aan een gebruiker dus een gebruiker kan zo bijvoorbeeld administratieve medewerker en manager zijn.

Casual use Case

Actoren: Baas

Hoofdscenario:

De baas navigeert naar de instellingen van een gebruiker 'of maakt een nieuwe gebruiker aan) en gaat deze gebruiker in het systeem aanpassen of verwijderen. Als de gerechtigde gebruiker zijn bewerkingen doorvoert zal dit zichtbaar worden door een melding die op het scherm verschijnt.

Alternatief scenario:

De baas voert een verkeerde input in. Het systeem zal dan duidelijk maken welke input er verwacht wordt door een foutmelding op het scherm te tonen

Alternatief scenario:

De baas gaat een persoon aanmaken die al bestaat. Dan zal het systeem een melding geven dat deze gebruiker al bestaat.

Alternatief scenario:

De baas gaat een persoon aanpassen of verwijderen die niet (meer) bestaat. Dan zal het systeem een melding geven dat deze gebruiker niet gevonden kan worden.

Visualisaties bekijken

Brief use case

Actoren: Baas

Beschrijving:

De baas navigeert naar het dashboard om visualisaties te krijgen over de gegevens. Het systeem gaat aan de hand van allerlei berekeningen verschillende grafieken tonen. De baas kan ook filteren welke grafieken voor hem het belangrijkste zijn. Het systeem zal dan alleen de gefilterde grafieken tonen. De belangrijkste parameters die gevisualiseerd worden zijn degenen die in de probleemstelling aangehaald worden.

Casual use Case

Actoren: Baas

Hoofdscenario:

De baas navigeert naar het dashboard en krijgt een overzicht van visualisaties van de reeds ingevoerde gegevens.

Alternatief scenario:

De baas navigeert naar het dashboard en filtert welke grafieken hij wilt bekijken. Het systeem zal dan alleen deze grafieken tonen die aan de filter(s) voldoen

Alternatief scenario:

De baas wilt een grafiek bekijken waarvoor geen of onvoldoende data ter beschikking is. Dan zal het systeem een melding geven dat er geen gegevens zijn voor deze soort grafiek.

Voedingspatroon beheren

Brief use case

Actoren: Voedings-beheerder

Beschrijving:

De voedings-beheerder navigeert naar voer management. Hij kan de volgende operaties uitvoeren:

1. voeding bestellingen bekijken
2. voeding bestellingen toevoegen
3. voeding bestellingen wijzigen
4. voeding bestellingen verwijderen

Casual use Case

Actoren: Voedings-beheerder

Hoofdscenario:

De voedings-beheerder navigeert naar voer management. Hij gaat daar dan voeding bestellingen bekijken, toevoegen, wijzigen of verwijderen [A][B]. Als een van deze acties voltooid zijn toont het systeem een melding op het scherm.

Alternatief scenario A:

De voedings-beheerder probeert een ongeldige hoeveelheid voeding te bestellen. Het systeem zal dan een melding geven dat deze bewerking niet kan worden voltooid

Alternatief scenario B:

De voedings-beheerder probeert een bestelling van voer aan te passen die niet (meer) bestaat. Het systeem zal dan een melding tonen op het scherm dat deze niet meer bestaat.

Taken beheren

Brief use case

Actoren: Agendabeheerder, werknemer

Beschrijving:

De agendabeheerder navigeert naar taakmanagement. Hij kan de volgende operaties uitvoeren:

1. een taak lezen./,
2. een taak aanmaken
3. een taak wijzigen
4. een taak verwijderen

Casual use Case

Actoren: Agendabeheerder, werknemer

Hoofdscenario:

De agenda beheerder navigeert naar taakmanagement en gaat een taak lezen, aanmaken, wijzigen of verwijderen. [A] Het systeem toont een melding op het scherm dat de bewerking is voltooid. Het systeem stuurt ook een melding naar de werknemer die aan de taak gelinkt is dat er een taak is aangemaakt, verwijderd of aangepast [B]. Het systeem zet deze taak dan ook in een agenda.

Alternatief scenario A:

De agenda beheerder probeert een bewerking op een taak uit te voeren die niet (meer) bestaat. Dan zal het systeem een melding tonen dat deze niet meer bestaat.

Alternatief scenario B:

Er is geen werknemer gelinkt aan een taak. Het systeem stuurt geen melding.

Repeterende taken beheren

Brief use case

Actoren: Agendabeheerder

Beschrijving:

Wanneer een nieuwe groep wordt aangemaakt, worden automatisch taken aangemaakt door het systeem die bij iedere groep van toepassing zijn.

Casual use Case

Actoren: Agendabeheerder

Hoofdscenario:

De agenda beheerder navigeert naar taakmanagement en kan daar een nieuwe taak aanmaken, aanpassen of verwijderen die hergebruikt kunnen worden. Zo kunnen taken automatisch gelinkt worden wanneer er een actie gebeurt zoals aanmaken van een groep.

Taken toekennen

Brief use case

Actoren: Agendabeheerder, werknemer

Beschrijving:

De agendabeheerder navigeert naar taakmanagement en kan bij het aanmaken of aanpassen van een taak een werknemer linken aan de taak.

Casual use Case

Actoren: Agendabeheerder, werknemer

Hoofdscenario:

De agenda beheerder navigeert naar taakmanagement en gaat bij het aanmaken of aanpassen werknemers linken aan een taak. Als deze opdracht gelukt is krijgt de agenda beheerder een melding dat deze bewerking voltooid is. De werknemer krijgt ook een melding dat hij wordt toegewezen aan de taak.

Alternatief scenario:

De agenda beheerder probeert een onbestaande werknemer te koppelen aan een taak. Het systeem zal dan op zijn beurt een foutmelding tonen.

Taken opnemen

Brief use case

Actoren: werknemer, agendabeheerder

Beschrijving:

De werknemer kan door te navigeren naar taakmanagement een taak toewijzen aan zichzelf. Het systeem linkt de taak aan de werknemer.

Casual use Case

Actoren: werknemer, agendabeheerder

Hoofdscenario:

De werknemer navigeert naar taakmanagement en krijgt een overzicht van alle taken. Hij kan dan een taak selecteren en aan hem toewijzen [A].

Alternatief scenario A:

De werknemer probeert een taak aan zichzelf toe te wijzen die niet meer bestaat. Het systeem zal dan een melding tonen op het scherm.

Taken voltooien

Brief use case

Actoren: Werknemer, Baas

Beschrijving:

De werknemer gaat naar de agenda. Het systeem toont een agenda. De werknemer selecteert een bestaande taak. Het systeem toont een overzicht van de taak in detail. De werknemer voltooit deze taak. Het systeem stuurt een notificatie naar de baas.

Casual use Case

Actoren: werknemer, Baas

Hoofdscenario:

De werknemer navigeert naar zijn agenda. Het systeem toont een agenda met de taak die de werknemer wilt voltooien. De werknemer kan dan deze taak selecteren waarop het programma een gedetailleerde beschrijving geeft van de taak. Hierin kan de werknemer een taak op voltooid zetten. Deze actie gaat de taak niet verwijderen maar gaat de taak op voltooid zetten en stuurt een notificatie naar de baas. Deze actie zal ook gelogd worden in het systeem.

Alternatief scenario:

De agenda beheerder probeert een taak te voltooien die niet (meer) bestaat. Dan zal het systeem een melding tonen dat deze niet meer bestaat.

Gegevens van groepen importeren

Brief use case

Actoren: Administratieve medewerker

Beschrijving:

De administratieve medewerker voorziet een .csv bestand dat het systeem kan importeren in een juiste formaat en importeert deze. Het systeem gaat alle data uit het bestand halen en slaat dit op in een databank.

Casual use Case

Actoren: Administratieve medewerker

Hoofdscenario:

De administratieve medewerker navigeert naar het import paneel en kan via een file explorer het bestand selecteren dat op zijn computer staat. Eenmaal dit is ingelezen worden deze gegevens opgeslagen in de databank. De administratieve medewerker krijgt dan een melding op het scherm dat het importeren is gelukt

Alternatief scenario:

De administratieve medewerker upload een ongeldig bestand. Het systeem zal dan een foutmelding geven op het scherm.

Groepen bekijken

Brief use case

Actoren: Werknemer

Beschrijving:

De werknemer gaat naar het overzicht van groepen en kan filteren welke groepen hij wilt bekijken. Het systeem toont een overzicht van alle groepen in het systeem. De werknemer kan deze bekijken, maar geen operaties op uitvoeren.

Casual use Case

Actoren: Werknemer

Hoofdscenario:

De werknemer navigeert naar het overzicht van groepen [B]. De gebruiker kan dan een groep kiezen om een meer gedetailleerde beschrijving van een groep te bekijken [A].

Alternatief scenario A:

De werknemer klikt een groep aan die niet meer bestaat. Het systeem zal dan een foutmeldingen tonen.

Alternatief scenario B:

De werknemer filtert de groepen.

Groepen beheren

Brief use case

Actoren: Manager

Beschrijving:

De manager navigeert naar groep management en kan zijn groepen beheren en bekijken. De manager heeft dan de mogelijkheid om volgende operaties uit te voeren:

1. groepen lezen
2. groepen toevoegen
3. groepen wijzigen
4. groepen verwijderen

Casual use Case

Actoren: Manager

Hoofdscenario:

De manager navigeert naar groep management en gaat een groep bewerken, toevoegen, aanpassen of verwijderen. Als deze actie voltooid is zal het systeem een melding geven als deze gelukt is.[A]

Alternatief scenario A:

De manager voert een actie uit op een groep die niet meer bestaat. Het systeem zal dan een foutmeldingen tonen.

Weging uitvoeren

Brief use case

Actoren: Weger

Beschrijving:

De weger navigeert naar groep management. Het systeem toont een lijst van groepen. De weger kan op elke groep een weging updaten.

Casual use Case

Actoren: Weger

Hoofdscenario:

De weger navigeert naar groep management en duidt een groep aan waarvoor hij een weging wil invoegen. Na het invoegen van een weging zal het systeem een melding op het scherm tonen dat deze actie is voltooid. [A] [B]

Alternatief scenario A:



De weger wilt een weging doen voor een groep die niet meer bestaat. Het systeem zal dan een melding geven dat deze niet meer bestaat.

Alternatief scenario B:

De weger geeft een ongeldige waarde in voor het toevoegen van een weging. Het systeem zal dan een melding geven.

URPS

Usability:

Het systeem moet een visuele indicatie geven wanneer er een fout optreedt.

De gebruiker moet 90% van de functionaliteit kunnen uitvoeren zonder documentatie te moeten lezen.

Reliability:

Het systeem moet foute invoer van data kunnen verwerken, zodat het systeem niet crashed.

Het systeem gebruikt enkel niet afgeronde waardes om berekeningen te doen. Enkel om waarden weer te geven mogen deze afgerond worden tot 3 decimalen

Performance:

Bij ieder interactie met het systeem moet er een reactie zijn binnen een seconde.


Supportability:


Het systeem moet zonder bestaande features aan te passen toch uitbreidbaar zijn, zodat dit eventueel kan evolueren naar een volledig administratief veeteelt platform.

Wanneer het systeem wordt aangepast moeten alle bestaande testen steeds blijven slagen eer het aangepaste systeem kan gereleased worden.

Functionele requirements

- 1) Het systeem moet toelaten een gebruiker te kunnen inloggen aan de hand van een e mail paswoord combinatie.
- 2) Het systeem moet in staat zijn om een foutmelding te weergeven wanneer dat een gebruiker niet kan inloggen.
- 3) Het systeem gebruikers onderscheiden op basis van claims (rechten).
- 4) Het systeem moet toelating tot functionaliteiten geven aan gebruikers op basis van de gebruikers claims.
- 5) Het systeem moet toegang verbieden aan gebruikers tot functionaliteiten op basis van de claims van een gebruiker.
- 6) Het systeem moet nieuwe gebruikers kunnen toevoegen door een gebruikersinteractie van een gerechtigde gebruiker.
- 7) Het systeem moet een gerechtigde gebruiker in staat stellen claims van gebruikers te kunnen toevoegen, verwijderen.
- 8) Het systeem moet gerechtigde gebruikers andere gebruikers kunnen laten verwijderen.
- 9) Het systeem moet ingevoerde gegevens over groepen dieren kunnen visualiseren.
- 10) Het systeem moet gegevens over de voedingspatronen opslaan in een database.
- 11) Het systeem moet gegevens over werknemers opslaan in een database.
- 12) Het systeem moet opgeslagen gegevens met betrekking op groepen dieren visueel kunnen vergelijken.
- 13) Het systeem moet gegevens kunnen ophalen uit de database op basis van, door de gebruiker, meegegeven filters.
- 14) Het systeem moet een afgeschermd omgeving opzetten voor elk nieuw bedrijf.
- 15) Het systeem moet omgevingen afschermen van gebruikers die niet tot dat bedrijf toebehoren.

- 
- 16) Het systeem moet een takenlijst per bedrijf bijhouden.
 - 17) Het systeem moet een gerechtigde gebruiker toelaten om taken aan te maken in de agenda die behoort tot de eige environment.
 - 18) Het systeem moet een gerechtigde gebruiker toelaten om taken toe te kennen aan gebruikers.
 - 19) Het systeem moet taken bijhouden die door een gebruiker zijn opgesteld.
 - 20) Het systeem moet taken kunnen updaten.
 - 21) Het systeem moet taken kunnen verwijderen aan de hand van een status van een taak.
 - 22) Het systeem moet taken kunnen genereren die een gerechtigde gebruiker als 'repetitief' beschouwd.
 - 23) Het systeem moet een melding geven wanneer een taak door een gebruiker uitgevoerd is.
 - 24) Het systeem moet gebruikers toelaten om taken te kunnen afronden.
 - 25) Het systeem moet de uitgevoerde taken met gebruiker kunnen loggen.
 - 26) Het systeem moet de gebruiker de functionaliteit bieden om de gegevens van ≥ 1 groepen dieren te beheren.
 - 27) Het systeem moet standaard eerst de gegevens van de huidige groep(en) dieren weergeven.
 - 28) Het systeem moet gegevens van oudere groepen dieren kunnen ophalen op aanvraag van de gerechtigde gebruiker.
 - 29) Het systeem moet gegevens van groepen dieren kunnen importeren aan de hand van csv en/of xlsx bestanden.
 - 30) Het systeem moet database records met informatie over de dieren groepen kunnen exporteren naar csv en/of xlsx bestanden.
 - 31) Het systeem moet database records met informatie over inkoop van voeding kunnen exporteren naar csv en/of xlsx bestanden.

- 
- 32) Het systeem moet gerechtigde gebruikers in staat stellen om gegevens van groep(en) dieren toe te voegen.
 - 33) Het systeem moet gegevens van voer/producten, die het bedrijf inkoopt, kunnen beheren(toevoegen, updaten, verwijderen).
 - 34) Het systeem moet gegevens van voer/producten, die het bedrijf inkoopt, kunnen opslaan in een database.
 - 35) Het systeem moet gerechtigde gebruikers in staat stellen weging gegevens van groepen dieren te kunnen beheren.
 - 36) Het systeem moet gerechtigde gebruikers in staat stellen weging gegevens van groepen dieren te kunnen opvragen op basis van datum en groep-identificer van de dieren.
 - 37) Het systeem moet de gegevens van de wegingen opslaan in een database.
 - 38) Het systeem moet in staat zijn weging gegevens van verschillende groepen dieren te kunnen visualiseren om zo een vergelijking te kunnen geven aan de gebruiker.

Technologie & effort

Technologie:

Het systeem is een REST-webapplicatie die gehost wordt in azure waarbij we makkelijk CI/CD pipelines kunnen integreren samen met de testen van de code. De database taal zal bestaan uit PostgreSQL. De backend zal geschreven worden in '.net core' waar we een ORM gebruiken om de database aan te spreken. De front-end wordt geschreven in React/typescript.

We kiezen voor een webapplicatie omdat veel van de team members al wat ervaring hebben met deze technologie en het is ook bruikbaar in iedere browser dus ook op mobiele toestellen.

We gebruiken SQL omdat de data perfect in een relationele manier kan gestructureerd worden.

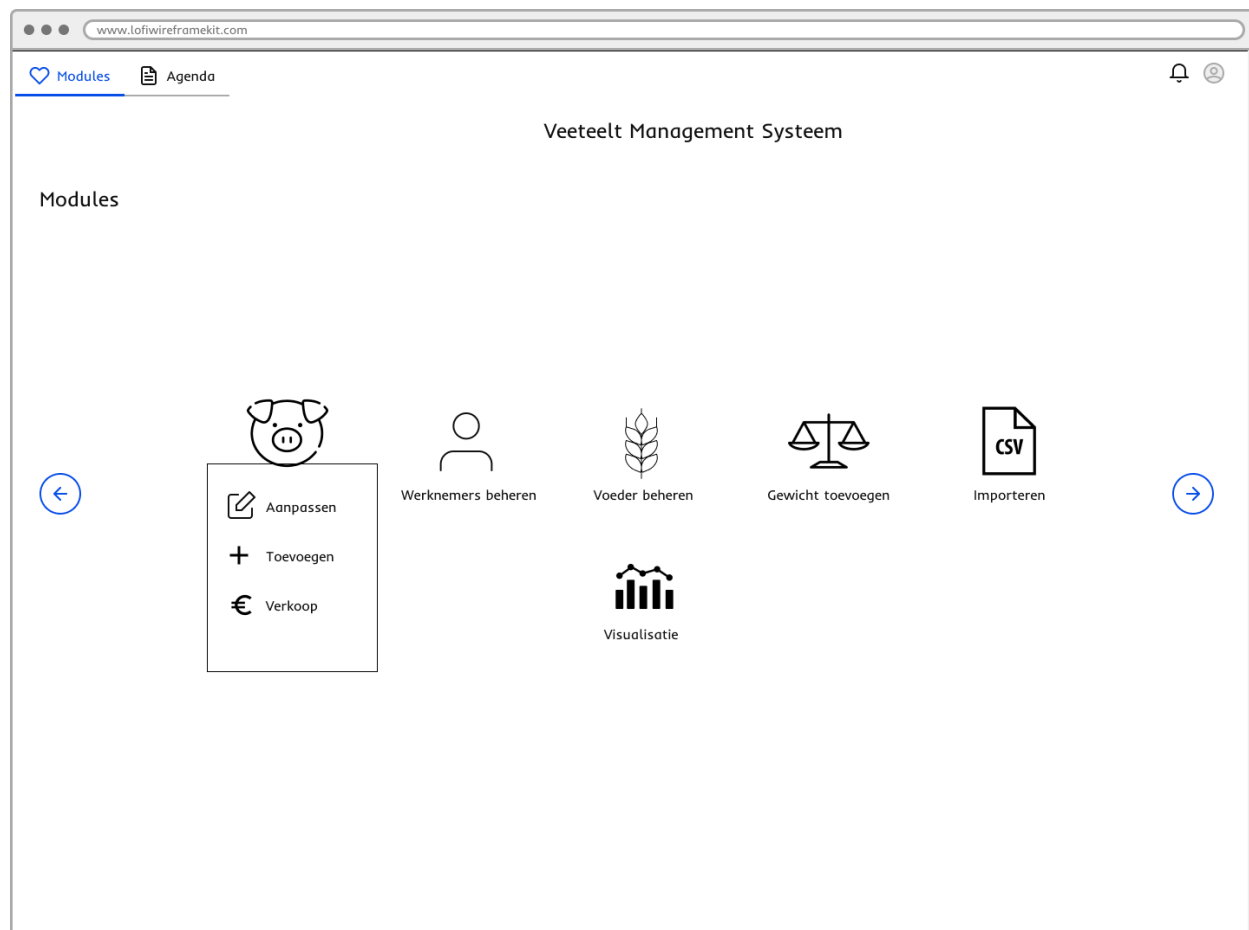
We gebruiken '.net core' omdat ook hier wat team members al ervaring mee hebben en het heeft een goede integratie met ORM's.

Effort

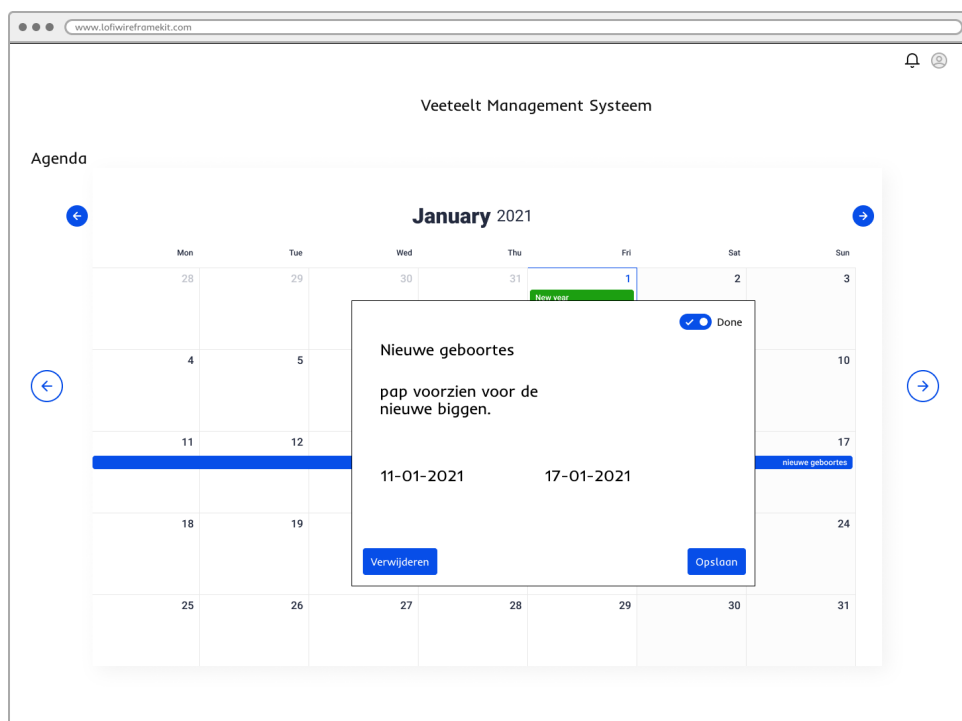
Omschrijving	Inschatting in dagen
Analyse	3
inloggen/uitloggen	1
Grafieken bekijken	10
Gebruikers beheren (+rechten)	6
Beheren van een groep	4
Een groep importeren	5
Taken beheren	4
Taken aanmaken	3
Taken voltooien	2
Een weging doen	2
Groepen bekijken	1
Voer beheren	6
Totaal:	47

Mockups

Startscherm met alle functies waartoe de gebruiker rechten heeft.



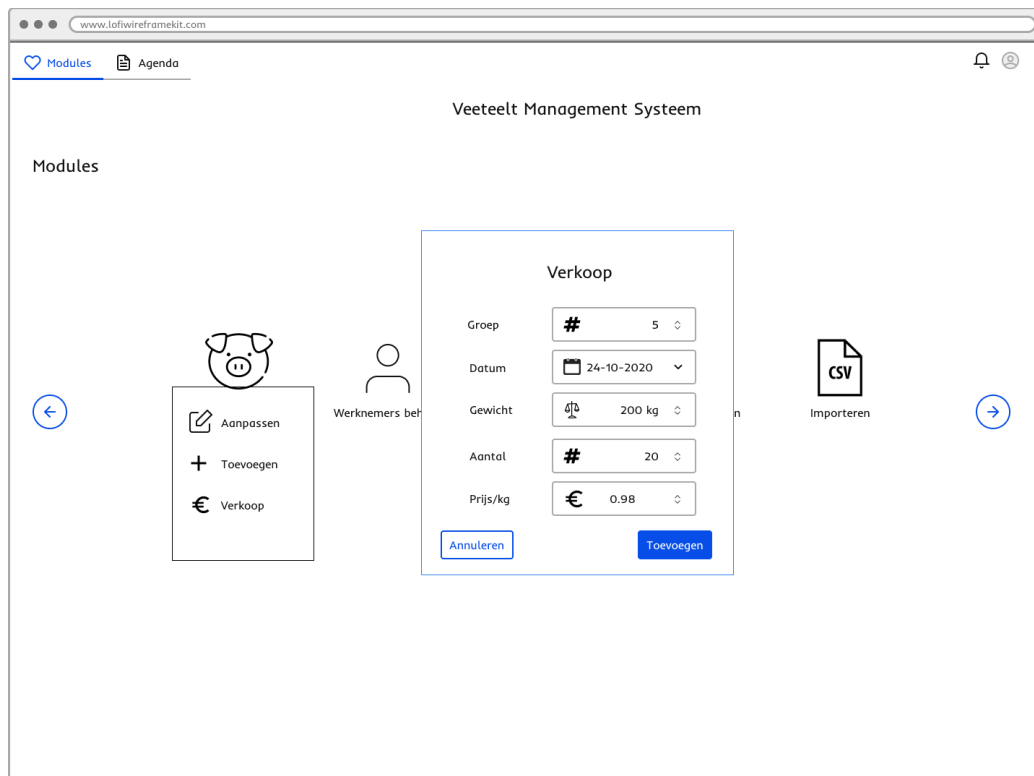
Kalender



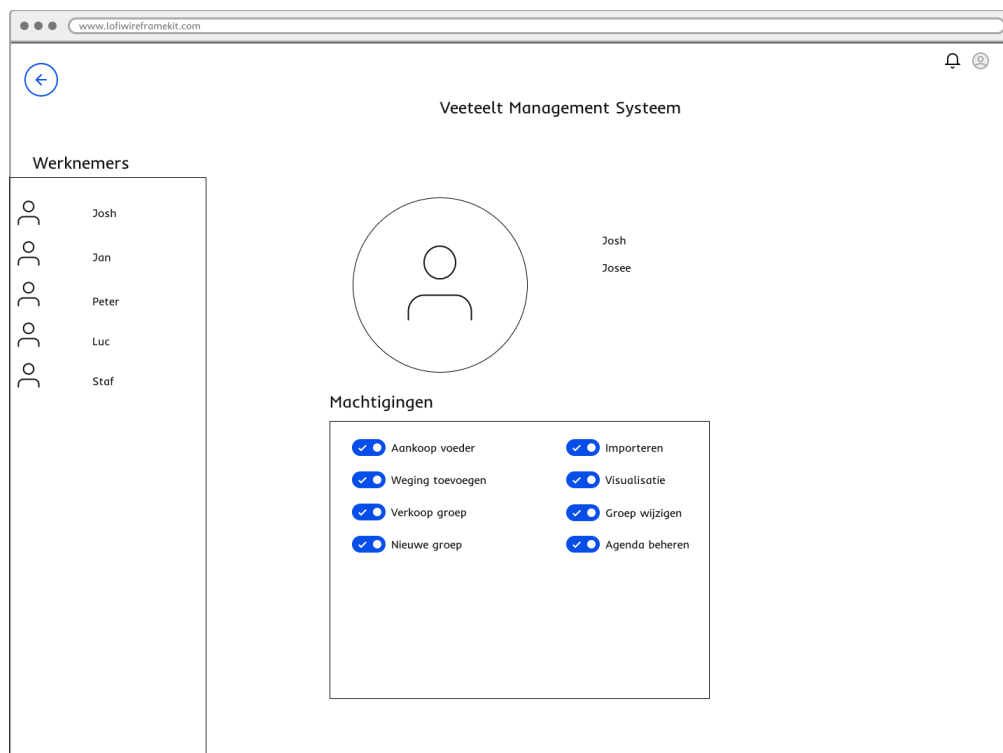
Grafieken



Verkoop



Gebruikers beheren






Voeder aankoop




www.lofiwireframekit.com

Modules Agenda

Veeteelt Management Systeem

Modules

←  Aanpassen + Toevoegen € Verkoop  Werknemers be...  Importeren →

Voeder Aankoop

Groep # 5; 7; 8

Datum 24-10-2020

Gewicht 200 kg

Prijs/kg € 20

Annuleren Toevoegen




Nieuwe groep




www.lofiwireframekit.com

Modules Agenda

Veeteelt Management Systeem

Modules

←  Aanpassen + Toevoegen € Verkoop  Werknemers be...  Importeren →

Nieuwe Groep

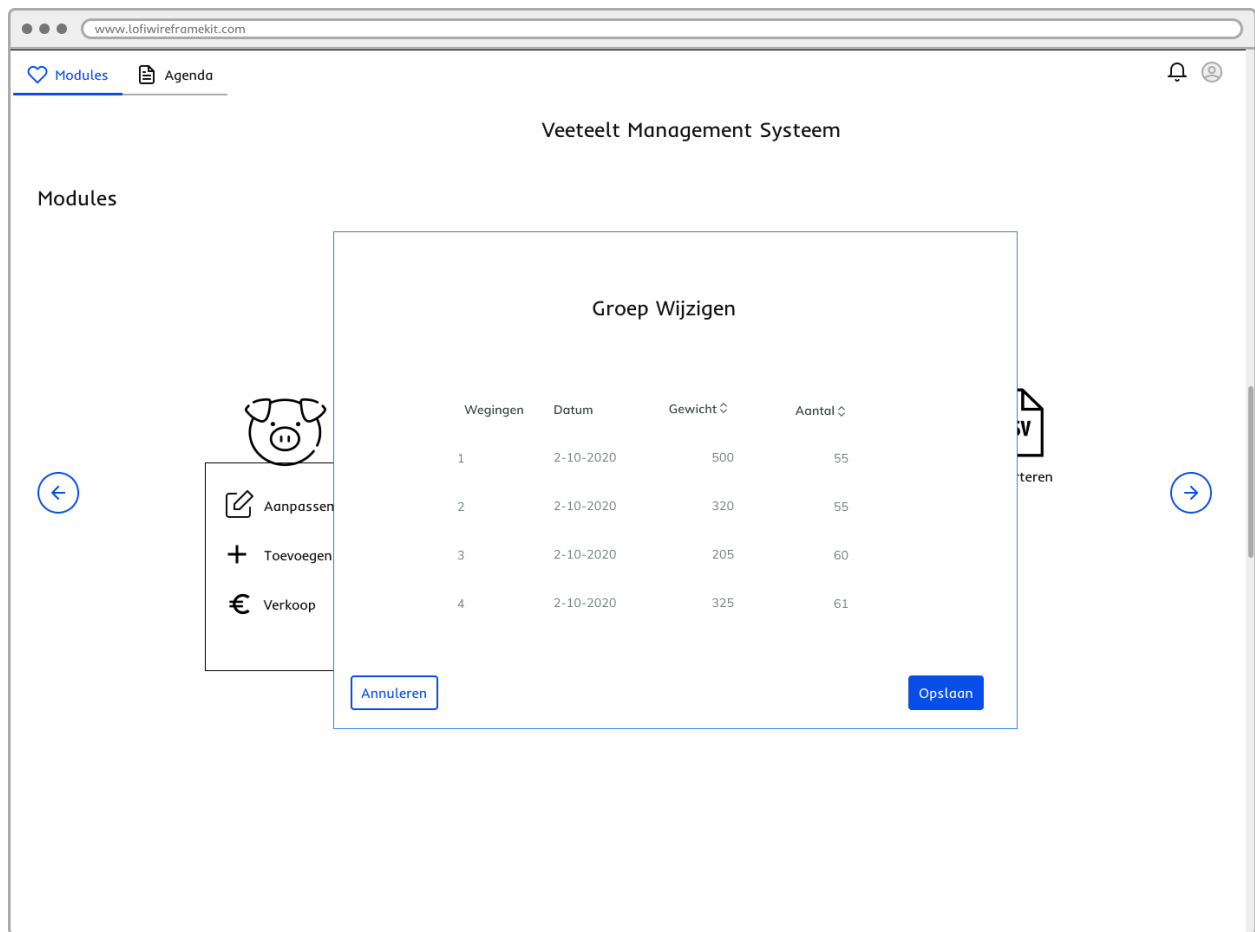
Datum 24-10-2020

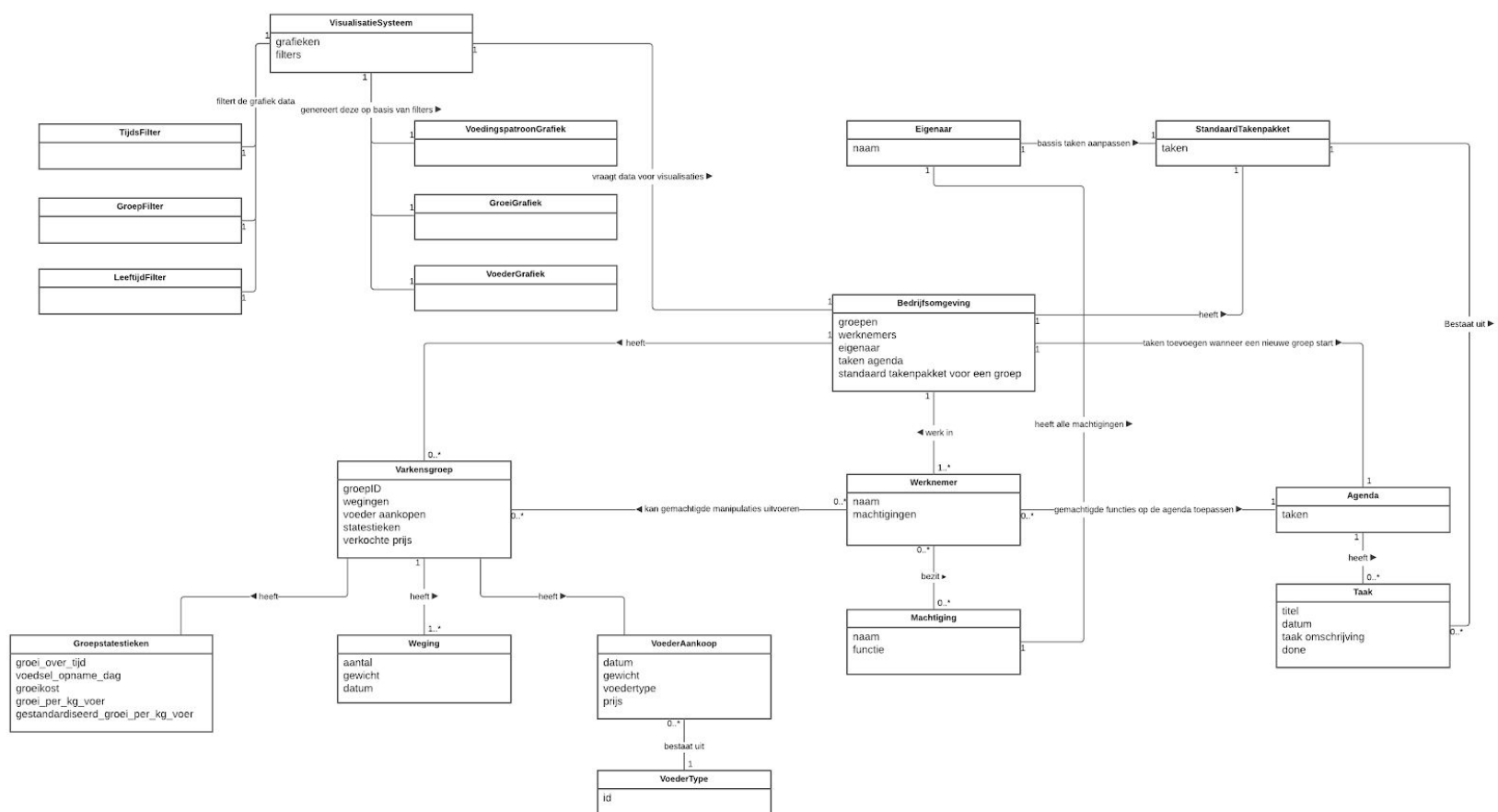
Gewicht 200 kg

Aantal # 20

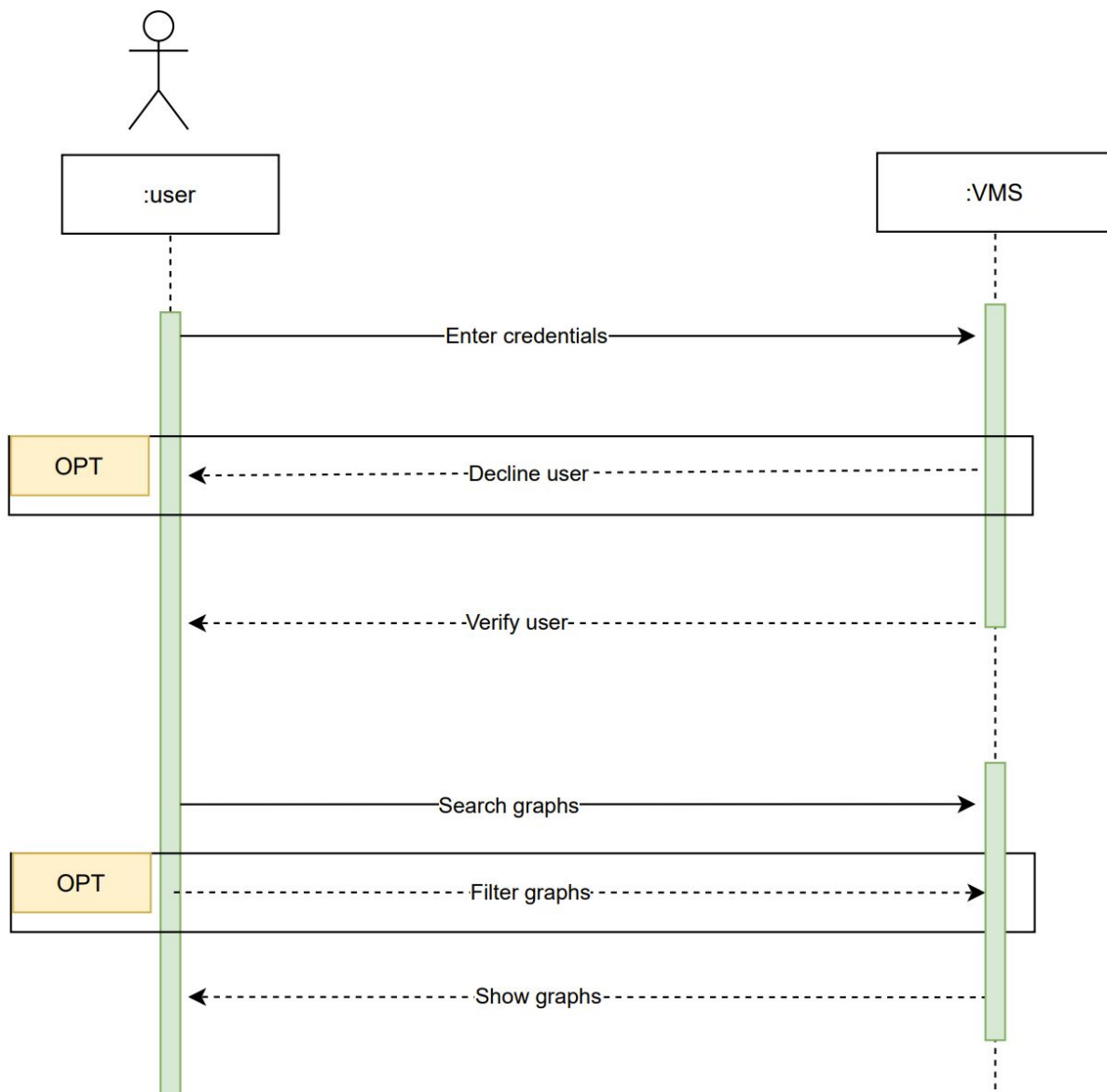
Annuleren Toevoegen

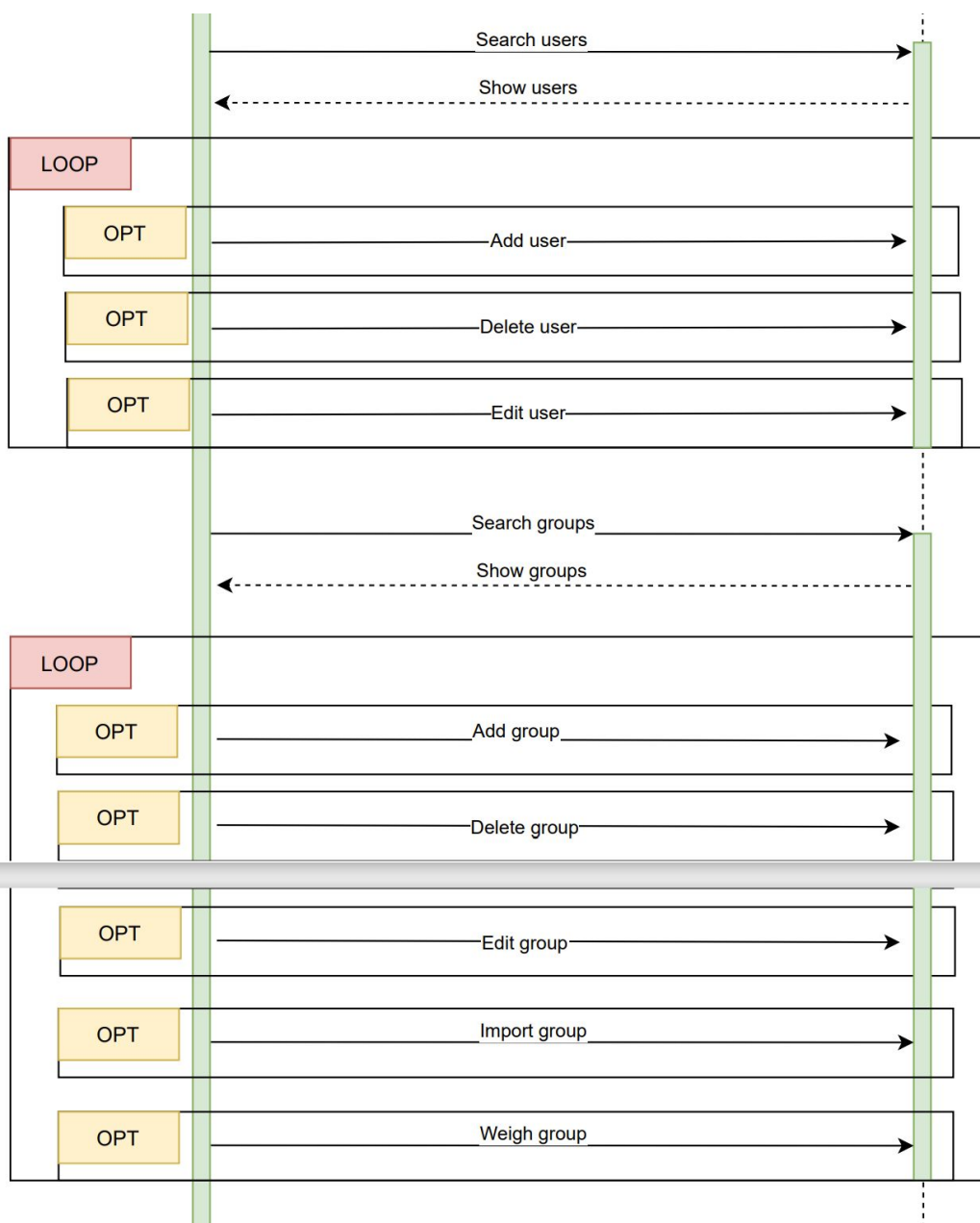
Groep wijzigen

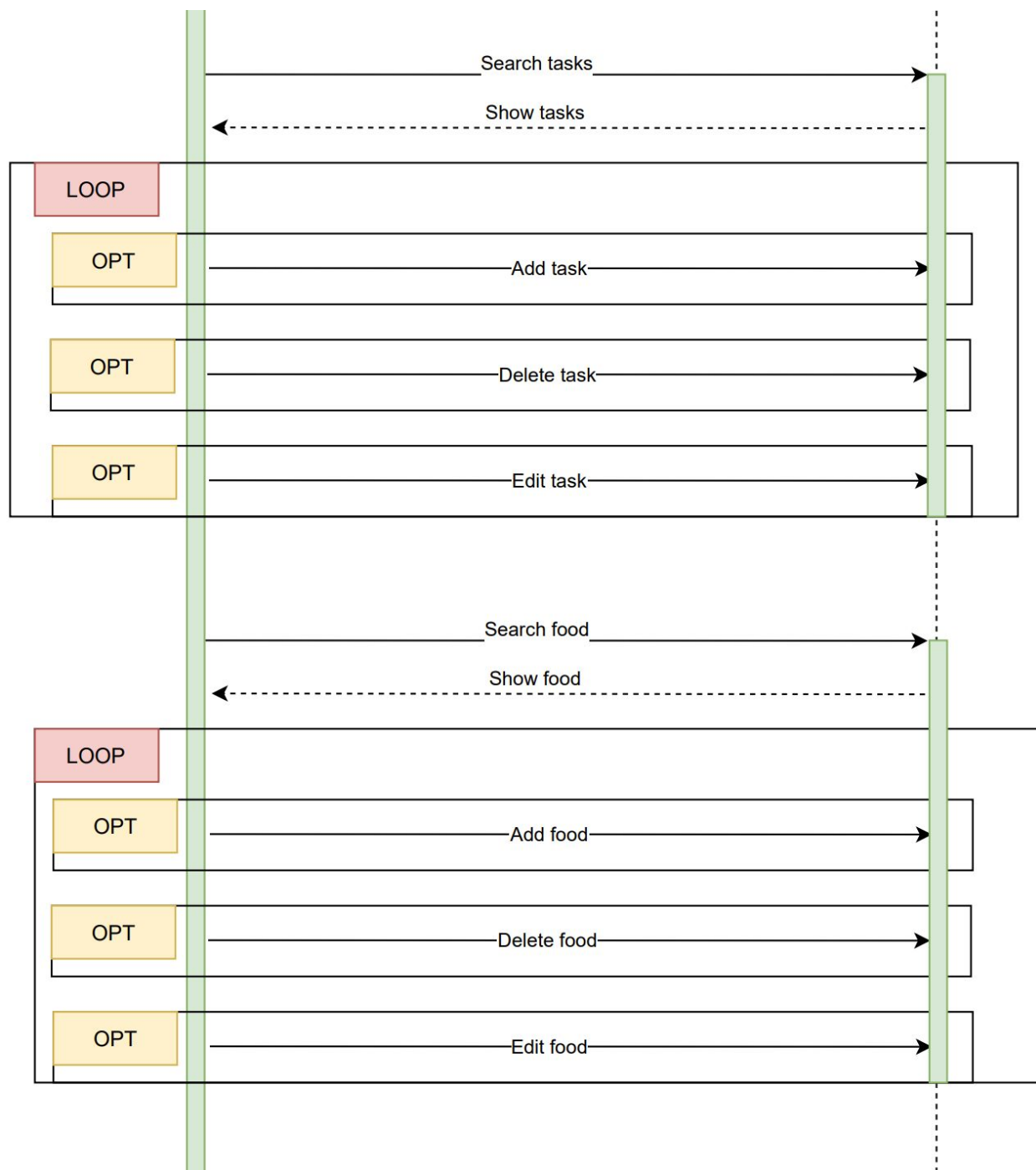


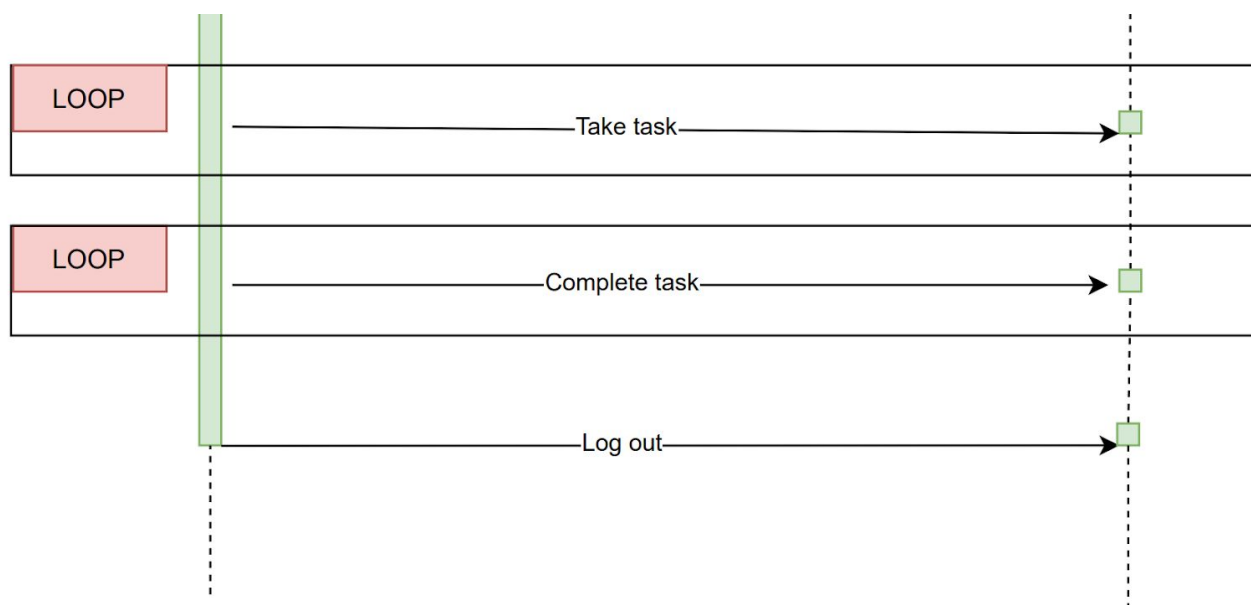


System sequentie diagram (SSD)









Contracten

Enter credentials

login(username: string, password: string)

Cross-reference: Use cases: inloggen

Preconditions: username en password is niet "null".
username lengte is korter dan 30 karakters.
username heeft enkel de volgende speciale karakters:

- liggend streepje [_]
- streepje [-]
- punt [.]

username - en password lengte is langer dan 0 letters.

password lengte is korter dan 128 karakters.

password bevat minimum 1 kleine letter.

password bevat minimum 1 grote letter.

password bevat minimum 1 getal.

password mag alle speciale karakters bevatten (niet verplicht).

username en password moeten samen een bestaande user definiëren.

Postconditions: het systeem voert de methode "declineUser()" of "verifyUser()" uit afhankelijk van de invoergegevens.

Decline User

declineUser(message: string)

Cross-reference: Use cases: inloggen [alternatief pad B].

Preconditions: methode "login(username:string, password:string)" is uitgevoerd.
er is geen user gevonden voor de combinatie van username en password.

Postconditions: het login scherm toont een foutboodschap.

Verify User

authenticate(token: string)

Cross-reference: Use cases: inloggen

Preconditions: methode "login(username:string, password:string)" is uitgevoerd.
de combinatie van username en password heeft een gebruiker gevonden.

Postconditions: het systeem slaagt een token op voor de huidige gebruiker.
het systeem doet een automatische navigatie naar de volgende pagina.
het systeem verbergt/toont items en pagina's afhankelijk van de huidige gebruiker zijn rechten.

Search graphs

getAllGraphs()

Cross-reference: Use cases: visualisaties bekijken

Preconditions: de gebruiker die deze actie uitvoert moet de juiste rechten hebben.
de gebruiker die deze actie uitvoert moet ingelogd zijn.
de gebruiker moet zich op de juiste pagina (grafiek pagina) bevinden.

Postconditions: het systeem maakt grafieken en toont deze op de juiste pagina.

Filter graphs

filterGraph(graphType: integer)

Cross-reference: Use cases: visualisaties bekijken

Preconditions: de gebruiker die deze actie uitvoert moet de juiste rechten hebben.
de gebruiker die deze actie uitvoert moet ingelogd zijn.
graphType is een natuurlijk getal en ligt tussen 1 en 3 (dus: 1, 2 of 3).

Postconditions: een visualisatie verandert van type als graphType != huidigeGraphType.

Note: graphType == 1 → lijndiagram
graphType == 2 → staafdiagram
graphType == 3 → cirkeldiagram

Search users

searchUsers()

Cross-reference: Use cases: Gebruikers beheren

Preconditions: De gebruiker moet ingelogd zijn in het systeem.

De gebruiker die deze actie uitvoert moet de juiste rechten hebben.

Postconditions: De return waarde 'users' moet een array van Users waarin alle users van het bedrijf in zitten.

Add user

addUser(newUser: User)

Cross-reference: Use cases: gebruikers beheren, taken toekennen, rechten beheren

Preconditions: newUser mag niet null zijn

newUser moet uniek zijn

De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

Postconditions: Het object 'newUser' wordt toegevoegd aan de lijst van users voor dit bedrijf.

Edit user

editUser(id: UUID, updatedUser: User)

Cross-reference: Use cases: Gebruikers beheren, taken toekennen, rechten beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

De 'id' moet een bestaande id zijn en kan niet null zijn

'updatedUser' mag niet null zijn.

Postconditions: De bestaande gebruiker met id: 'id' wordt aangepast naar de waarde 'updatedUser'.

Delete user

deleteUser(id: UUID)

Cross-reference: Use cases: Gebruikers beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

De 'id' mag niet null zijn

De 'id' moet een id zijn van een bestaande User

Postconditions: De gebruiker met id : 'id' wordt verwijderd uit de lijst met gebruikers voor dit bedrijf.

Search groups

searchGroups()

Cross-reference: Use cases: Groepen bekijken, Groepen beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben.

De gebruiker die deze actie uitvoert moet ingelogd zijn.

Postconditions: De return waarde 'groups' moet een array van groepen waarin alle groepen van het bedrijf in zitten.

Add group

addGoup(newGroup: Group)

Cross-reference: Use cases: Groepen beheren

Preconditions: 'newGroup' mag niet null zijn

'newGroup' moet uniek zijn

De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

Postconditions: Het object 'newGroup' wordt toegevoegd aan de lijst van groepen voor dit bedrijf.

Edit group

editGroup(id: UUID, updatedGroup: Group)

Cross-reference: Use cases: Groepen beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

De 'id' moet een bestaande id zijn en kan niet null zijn
'updatedGroup' mag niet null zijn.

Postcondities: De bestaande groep met id: 'id' wordt aangepast naar de waarde 'updatedGroup'.

Delete group

deleteGroup(id: UUID)

Cross-reference: Use cases: Groepen beheren

Precondities: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

De 'id' mag niet null zijn

De 'id' moet een id zijn van een bestaande group

Postcondities: De groep met id : 'id' wordt verwijderd uit de lijst van groepen voor dit bedrijf.

Import group

importGroup(data: DataFrame)

Cross-reference: Use cases: groepen importeren

Precondities: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

'data' mag niet null zijn

'data' moet voldoen aan de juiste formaat voorwaarden

Postcondities: Alle groepen die zich bevinden in het 'data' DataFrame worden toegevoegd aan de lijst van groepen voor dit bedrijf samen met alle gegevens die gekoppelt zijn aan deze groepen.

Weigh group

weighGroup(groupId: UUID, weighing: Weighing)

Cross-reference: Use cases: Weging uitvoeren

Precondities: De gebruiker die deze actie uitvoert moet de juiste rechten hebben

De gebruiker die deze actie uitvoert moet ingelogd zijn

De 'groupId' mag niet null zijn

De 'groupId' moet een id zijn van een bestaande groep

weighing mag niet null zijn

weighing moet een nieuwe unieke instantie van de Weighingklasse zijn

Postconditions: Voor de groep met id: 'groupId' wordt de weging: 'weighing' toegevoegd aan de lijst van wegingen voor deze groep.

Search task

searchTask(id: UUID)

Cross-reference: Use cases: Taken Beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben;

De gebruiker die deze actie uitvoert moet ingelogd zijn;

De id moet van een geldige universally unique identifier zijn;

Postconditions: De gevraagde task met hetzelfde id als de parameter wordt teruggegeven. Wanneer deze niet gevonden is, wordt een foutboodschap getoond.

Find all tasks

searchTasks()

Cross-reference: Use cases: Taken Beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben;

De gebruiker die deze actie uitvoert moet ingelogd zijn;

Er moeten al reeds taken bestaan;

Postconditions: een collectie van alle taken uit de gekoppelde database wordt teruggegeven.

Add Task

createTask(task: TaskObject)

Cross-reference: Use cases: Taken Beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een nieuwe Task aan te maken;

Het meegegeven Task object heeft voor alle verplichte velden een waarde;

Het meegegeven Task object bezit voor het id veld GEEN waarde;

Postconditions: Het meegegeven Task object wordt opgeslagen in de database die gekoppeld is aan het systeem;

Het meegegeven Task object heeft een id toegekend gekregen door de, aan het systeem gekoppelde, database;

Delete Task

deleteTask(id: UUID)

Cross-reference: Use cases: Taken Beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een Tasks aan te verwijderen;

De meegegeven id is een geldig universally unique identifier;

Postconditions: De taak met het meegegeven id wordt uit het systeem verwijderd;

Indien er geen taak gevonden is met het meegegeven id, wordt er een foutboodschap getoond.

Edit Task

updateTask(id: UUID, updatedTask: TaskObject)

Cross-reference: Use cases: Taken Beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een Tasks aan te passen;

De meegegeven id is een geldig universally unique identifier;

Het meegegeven Task object heeft voor alle verplichte velden een waarde;

Het meegegeven Task object bezit geen waarde voor het id veld;

Postconditions: De velden van het task object dat gevonden werd op basis van de meegegeven id parameter zijn hetzelfde als de waardes van de velden van het meegegeven Task object;

Indien er geen taak gevonden is met de meegegeven id, wordt een foutboodschap getoond.

Search Food

findFood(id: UUID)

Cross-reference: Use cases: Voedingspatroon beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben;

De gebruiker die deze actie uitvoert moet ingelogd zijn;

De id moet van een geldige universally unique identifier zijn;

Postconditions: Het gevraagde Food object, met hetzelfde id als de parameter, wordt teruggegeven. Wanneer deze niet gevonden is, wordt een exceptie gegooid die de aanroeper moet kunnen afhandelen.

Find all Food records

findFoods()

Cross-reference: Use cases: Food Beheren

Preconditions: De gebruiker die deze actie uitvoert moet de juiste rechten hebben;

De gebruiker die deze actie uitvoert moet ingelogd zijn;

Er moeten al reeds Food objecten bestaan;

Postconditions: een collectie van alle Food objecten uit de gekoppelde database wordt teruggegeven.

Add Food

createFood(newFood: FoodObject)

Cross-reference: Use cases: Voedingspatroon beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een nieuwe Food object aan te maken;

Het meegegeven Food object heeft voor alle verplichte velden een waarde;

Het meegegeven Food object bezit voor het id veld GEEN waarde;

Postconditions: Het meegegeven Food object wordt opgeslagen in de database die gekoppeld is aan het systeem.

Delete Food

deleteFood(id: UUID)

Cross-reference: Use cases: Voedingspatroon beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een Food aan te verwijderen;

De meegeven id is een geldig universally unique identifier;

Postconditions: Het Food object met het meegegeven id wordt uit het systeem verwijderd;

Indien er geen Food object gevonden is met het meegeven id, wordt er een exceptie gegooit die door de aanroeper van de functie moet afgehandeld worden;

Edit Food

updateFood(id: UUID, updatedFood: FoodObject)

Cross-reference: Use cases: Voedingspatroon beheren

Preconditions: De aanroeper is een ingelogde gebruiker;

De aanroeper heeft de juiste rechten om een Food objecten aan te passen;

De meegeven id is een geldig universally unique identifier;

Het meegegeven Food object heeft voor alle verplichte velden een waarde;

Het meegegeven Food object bezit geen waarde voor het id veld;

Postconditions: De velden van het Food object dat gevonden werd op basis van de meegegeven id parameter zijn hetzelfde als de waardes van de velden van het meegeven Food object;

Indien er geen taak gevonden is met het meegeven id, wordt er een exceptie gegooit die door de aanroeper van de functie moet afgehandeld worden;

Take Task

assignTask(taskId: UUID, employeeld: UUID)

Cross-reference: Use cases: Taken opnemen

Preconditions: De taskId en de employeeld mogen niet "null" zijn en moeten een geldig formaat hebben

taskId bestaat in de database als een identifier voor een taak en employeeld analoog voor werknemer.

done (attribuut van de taak instantie) moet op false staan.

Postconditions: een instantie van een taak wordt opgeroepen. de id van deze taak is gelijk aan taskId.

assignedEmployeeld (attribuut van de taak instantie) is gelijk aan employeeld.

Complete Task

completeTask(taskId: UUID)

Cross-reference: Use cases: taken voltooien

Preconditions: taskId mag niet "null" zijn en moet groter zijn dan 0.

taskId bestaat in de database als een identifier voor een taak en employeeld analoog voor werknemer.

done (attribuut van de taak instantie) == false.

Postconditions: een instantie van een taak wordt opgeroepen. de id van deze taak == taskId.

done (attribuut van de taak instantie) == true.

Log out

logout()

Cross-reference: Use cases: inloggen

Preconditions: er is een token bewaard voor een sessie (m.a.w. er is een gebruiker ingelogd).

Postconditions: het bewaarde token is verwijderd.

het systeem navigeert naar de login pagina.

Use case diagram

