

Analysedocument

ACI Rental system

Fontys Hogeschool | Tilburg

Auteur: Job Verwiël, Siebren Kraak, Marco Ketelaars, Myron Antonissen, Angelo Bruggemans
Locatie: Tilburg
Versie: 1.0
Datum: 23 feb 2021

1 Versiebeheer

Versie	Wijziging	Datum
0.1	Opzet document	23-02-2021
0.2	Toevoegen requirements, toevoegen use case diagram	01-03-2021
1.0	Toevoegen domeinmodel, wijzigen requirements	02-03-2021

Inhoud

1	Versiebeheer	2
2	Inleiding	4
3	Context	4
4	Requirements	5
4.1	Functionele requirements	5
4.2	Non functional requirements	6
4.3	Definition of Done	7
4.4	Kosten.....	7
4.5	Documentatie kwaliteit.....	7
4.6	Ease of use.....	7
4.7	Schaalbaarheid	8
5	Use case diagram.....	9
6	Domeinmodel	10

2 Inleiding

In dit document wordt al beschreven wat voor functionaliteiten in de applicatie komen, krijgen deze functionaliteiten een prioriteit en worden er ook beperkingen gegeven aan deze prioriteiten zodat ze binnen de scope blijven en het een duidelijke requirement blijft. Met behulp van Use Cases krijgen deze requirements meer vorm door ze in een context te plaatsen. Ook wordt er met een Use Case diagram getoond wat mogelijk is voor welke actoren binnen de applicatie. Hiernaast worden er ook Non Functionals opgesteld om de kwaliteiten van de applicatie te beschrijven en wordt er al een kleine schets gemaakt doormiddel van een domeinmodel.

3 Context

Fontys Academy for Creative Industries (Fontys ACI) is onderdeel van Fontys Hogescholen, dat met ruim 40.000 studenten en ongeveer 4.000 medewerkers de grootste hbo-instelling is in het zuiden van Nederland.

Bij deze opleiding wordt er regelmatig gewerkt met apparatuur die voor de meeste studenten niet te betalen is. Denk hierbij aan professionele camera's, filmapparatuur en andere apparatuur. Daarom heeft ACI een balie waar deze apparatuur geleend kan worden zodat hier gebruik gemaakt van kan worden tijdens de opleiding.

De opdrachtgever van dit project meneer Harmen Kuppens. Binnen ACI is Harmen Kuppens onderwijsassistent en medewerker bij de uitleenbalie. Tijdens dit project zal hij de rol van 'product owner' op zich nemen. Ook zal Jacques de Roij regelmatig bij gesprekken aanwezig zijn en zich bezighouden met dit project.

Echter voldoet de huidige oplossing niet aan alle eisen die door ACI worden gesteld. Omdat er meer dan 40.000 studenten en 4.000 medewerkers zijn die potentieel gebruik maken van het nieuwe systeem is het van belang dat de software goed schaalbaar en uitbreidbaar is. Ook is het gewenst dat de nieuwe software niet in één keer de oude software vervangt, maar dat elke sprint een nieuw stuk wordt opgeleverd en geïmplementeerd wordt. Het maken van microservices is met deze requirements onmisbaar.

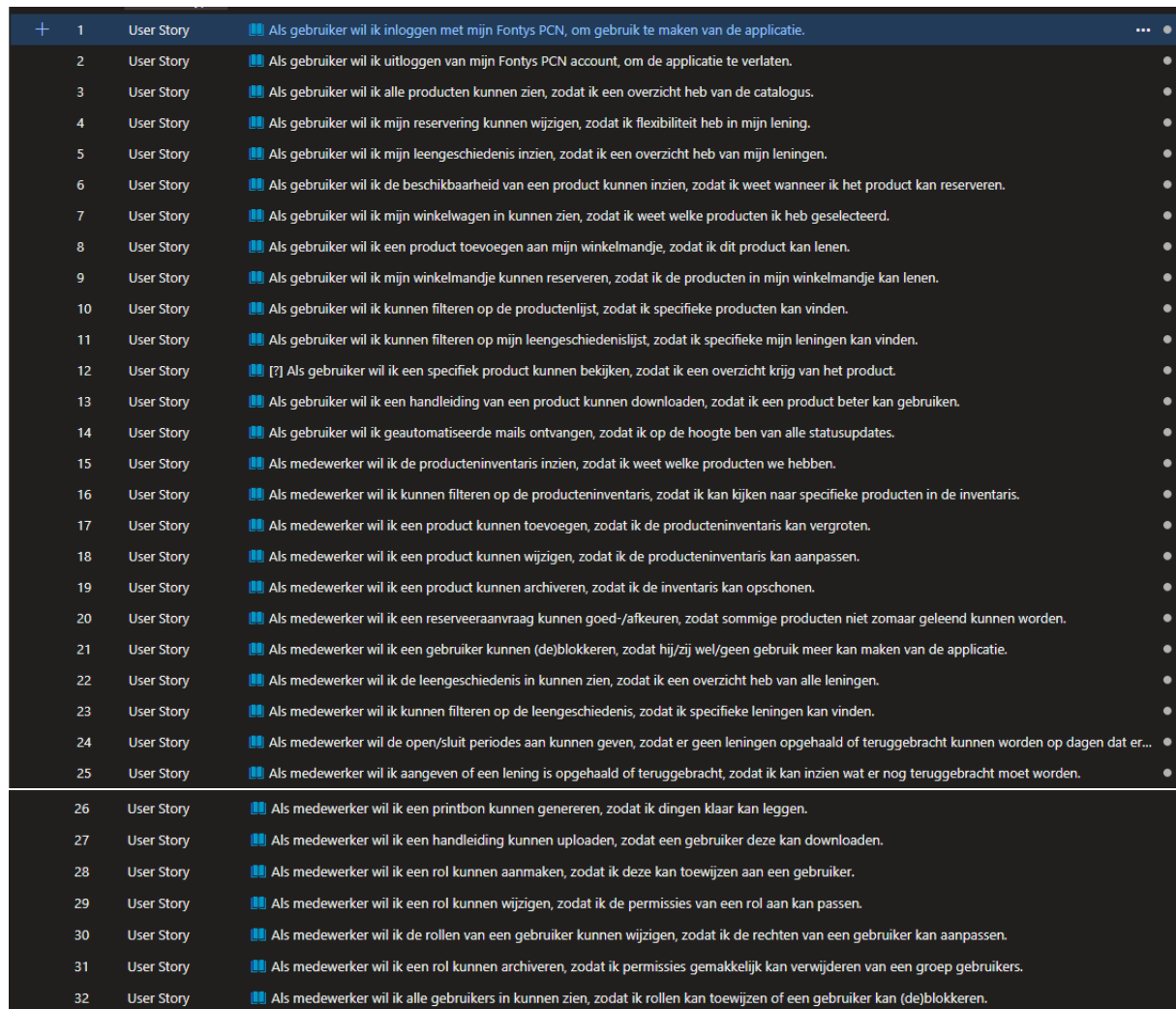
Het systeem zal features moeten bevatten voor medewerkers en voor studenten. Studenten moeten apparatuur kunnen bekijken en reserveren, oude reserveringen in kunnen zien en boetes kunnen bekijken en betalen. Medewerkers moeten ook apparatuur in kunnen zien, de inventaris wijzigen, reserveringen eerder beëindigen (eerder teruggebracht), notities plaatsen en reserveerhistorie van studenten kunnen bekijken.

Wanneer het over duurdere apparatuur gaat, kan er mogelijk borg of een additionele check op een apparaat zitten. Deze borg moet dan ook betaald kunnen worden via de applicatie en automatisch teruggestort worden. Wanneer er een additionele check nodig is, moet een medewerker goedkeuring geven wanneer een reservering aangevraagd wordt. Dit kan gedaan worden op basis van de reserveerhistorie van de student die de reservering heeft aangevraagd.

4 Requirements

4.1 Functionele requirements

Voor dit project zijn functionele requirements opgesteld na gesprekken met de opdrachtgever. Deze requirements zijn in de vorm van User Stories in de backlog gezet in de DevOps omgeving.



+	1	User Story	Als gebruiker wil ik inloggen met mijn Fontys PCN, om gebruik te maken van de applicatie.	...
	2	User Story	Als gebruiker wil ik uitloggen van mijn Fontys PCN account, om de applicatie te verlaten.	
	3	User Story	Als gebruiker wil ik alle producten kunnen zien, zodat ik een overzicht heb van de catalogus.	
	4	User Story	Als gebruiker wil ik mijn reservering kunnen wijzigen, zodat ik flexibiliteit heb in mijn lening.	
	5	User Story	Als gebruiker wil ik mijn leengeschiedenis inzien, zodat ik een overzicht heb van mijn leningen.	
	6	User Story	Als gebruiker wil ik de beschikbaarheid van een product kunnen inzien, zodat ik weet wanneer ik het product kan reserveren.	
	7	User Story	Als gebruiker wil ik mijn winkelwagen in kunnen zien, zodat ik weet welke producten ik heb geselecteerd.	
	8	User Story	Als gebruiker wil ik een product toevoegen aan mijn winkelmandje, zodat ik dit product kan lenen.	
	9	User Story	Als gebruiker wil ik mijn winkelmandje kunnen reserveren, zodat ik de producten in mijn winkelmandje kan lenen.	
	10	User Story	Als gebruiker wil ik kunnen filteren op de productenlijst, zodat ik specifieke producten kan vinden.	
	11	User Story	Als gebruiker wil ik kunnen filteren op mijn leengeschiedenislijst, zodat ik specifieke mijn leningen kan vinden.	
	12	User Story	[?] Als gebruiker wil ik een specifiek product kunnen bekijken, zodat ik een overzicht krijg van het product.	
	13	User Story	Als gebruiker wil ik een handleiding van een product kunnen downloaden, zodat ik een product beter kan gebruiken.	
	14	User Story	Als gebruiker wil ik geautomatiseerde mails ontvangen, zodat ik op de hoogte ben van alle statusupdates.	
	15	User Story	Als medewerker wil ik de producteninventaris inzien, zodat ik weet welke producten we hebben.	
	16	User Story	Als medewerker wil ik kunnen filteren op de producteninventaris, zodat ik kan kijken naar specifieke producten in de inventaris.	
	17	User Story	Als medewerker wil ik een product kunnen toevoegen, zodat ik de producteninventaris kan vergroten.	
	18	User Story	Als medewerker wil ik een product kunnen wijzigen, zodat ik de producteninventaris kan aanpassen.	
	19	User Story	Als medewerker wil ik een product kunnen archiveren, zodat ik de inventaris kan opschonen.	
	20	User Story	Als medewerker wil ik een reserveeraanvraag kunnen goed-/afkeuren, zodat sommige producten niet zomaar geleend kunnen worden.	
	21	User Story	Als medewerker wil ik een gebruiker kunnen (de)blokkeren, zodat hij/zij wel/geen gebruik meer kan maken van de applicatie.	
	22	User Story	Als medewerker wil ik de leengeschiedenis in kunnen zien, zodat ik een overzicht heb van alle leningen.	
	23	User Story	Als medewerker wil ik kunnen filteren op de leengeschiedenis, zodat ik specifieke leningen kan vinden.	
	24	User Story	Als medewerker wil de open/sluit periodes aan kunnen geven, zodat er geen leningen opgehaald of teruggebracht kunnen worden op dagen dat er...	
	25	User Story	Als medewerker wil ik aangeven of een lening is opgehaald of teruggebracht, zodat ik kan inzien wat er nog teruggebracht moet worden.	
	26	User Story	Als medewerker wil ik een printbon kunnen genereren, zodat ik dingen klaar kan leggen.	
	27	User Story	Als medewerker wil ik een handleiding kunnen uploaden, zodat een gebruiker deze kan downloaden.	
	28	User Story	Als medewerker wil ik een rol kunnen aanmaken, zodat ik deze kan toewijzen aan een gebruiker.	
	29	User Story	Als medewerker wil ik een rol kunnen wijzigen, zodat ik de permissies van een rol aan kan passen.	
	30	User Story	Als medewerker wil ik de rollen van een gebruiker kunnen wijzigen, zodat ik de rechten van een gebruiker kan aanpassen.	
	31	User Story	Als medewerker wil ik een rol kunnen archiveren, zodat ik permissies gemakkelijk kan verwijderen van een groep gebruikers.	
	32	User Story	Als medewerker wil ik alle gebruikers in kunnen zien, zodat ik rollen kan toewijzen of een gebruiker kan (de)blokkeren.	

Figuur 1 De user stories op de Azure DevOps omgeving

4.2 Non functional requirements

Niet functionele requirements zijn eisen aan het systeem. Die vooral de kwaliteit van het systeem moeten aangeven

Id	Categorie ISO 25010	Omschrijving
Q1	Prestatie-efficiëntie	Het systeem is opgebouwd in losse componenten zodat het systeem schaalbaar is.
Q2	Bruikbaarheid	Het systeem kan door 95% van de doelgroep binnen 10 minuten worden gebruikt.
Q3	Bruikbaarheid	Het systeem is bruikbaar voor mensen die niet Nederlands praten.
Q4	Beveiligbaarheid	Wachtwoorden worden versleuteld opgeslagen.
Q5	Beveiligbaarheid	Gegevens moeten beschikbaar zijn voor de correct geautoriseerde gebruiker.
Q6	Beveiligbaarheid	Het systeem controleert de identiteit van de gebruiker.
Q7	Onderhoudbaarheid	Het systeem is opgebouwd in losse componenten zodat het gemakkelijk te wijzigen of toe te voegen is.
Q8	Onderhoudbaarheid	Het systeem is modulair opgebouwd.
Q9	Onderhoudbaarheid	Source code is gedocumenteerd.
Q10	Onderhoudbaarheid	Het systeem heeft 80% unit test code coverage.
Q11	Overdraagbaarheid	Het systeem is zo opgebouwd dat onderdelen makkelijk vervangbaar zijn.

4.3 Definition of Done

Gedefinieerd in projectplan.

4.4 Kosten

Dit project is in ontwikkeling genomen door studenten die momenteel in semester 6 zitten. Dit betekent dat uurloon of andere compensatie voor het maken van dit project niet van toepassing is. Echter zijn er wel andere kosten verbonden aan het ontwikkelen van dit project:

- Azure DevOps omgeving
- Azure DevOps job agents
- Visual Studio 2019
- Office-producten (Office 365)
- Tijd van docenten en opdrachtgever

Wanneer het huidige semester eindigt zal de software in productie genomen worden. Dit betekent dat er dan ook kosten komen voor het hosten van het front-end, backends en database. Omdat er gebruik gemaakt gaat worden van microservices i.c.m. Docker, is de software goed schaalbaar en zijn de kosten lager dan zonder het gebruik van containers.

4.5 Documentatie kwaliteit

Omdat dit project voor een langere tijd in ontwikkeling gaat zijn (en door verschillende proftaak teams) is het van belang dat de overdraagbaarheid goed is. Eén van de opties die helpen om de overdraagbaarheid te verbeteren is om documentatie te schrijven die duidelijk, straight-to-the-point en up-to-date is. Ook is het belangrijk dat keuzes en sourcecode gedocumenteerd is.

Alle documentatie van dit project wordt opgeslagen in de publieke OneDrive. Alle documentatie heeft dezelfde huisstijl en bevat altijd de context van de opdracht.

De sourcecode is gedocumenteerd door overal dezelfde code-stijl te gebruiken, goede functienamen en lastige stukken code met commentaar te beschrijven. Om deze code-stijl recht te houden wordt er gebruik gemaakt van SonarQube en de bijbehorende plug-ins in de IDE. Ook kan code alleen gemerged worden wanneer er minimaal één persoon de pull-request beoordeeld heeft.

4.6 Ease of use

Bij het uitwerken van het project wordt rekening gehouden met de doelgroep. Omdat deze doelgroep niet altijd veel technische kennis heeft, moet hier rekening mee gehouden worden in het design van de applicatie en zijn functionaliteiten. Zo wordt er op gelet dat er niet al te veel kliks nodig zijn om een bepaalde functionaliteit aan te roepen. Daarnaast wordt erop gelet dat pagina's niet te onoverzichtelijk worden voor gebruikers. In een later stadium zal dit ook getest worden door mensen met minder technische kennis.

4.7 Schaalbaarheid

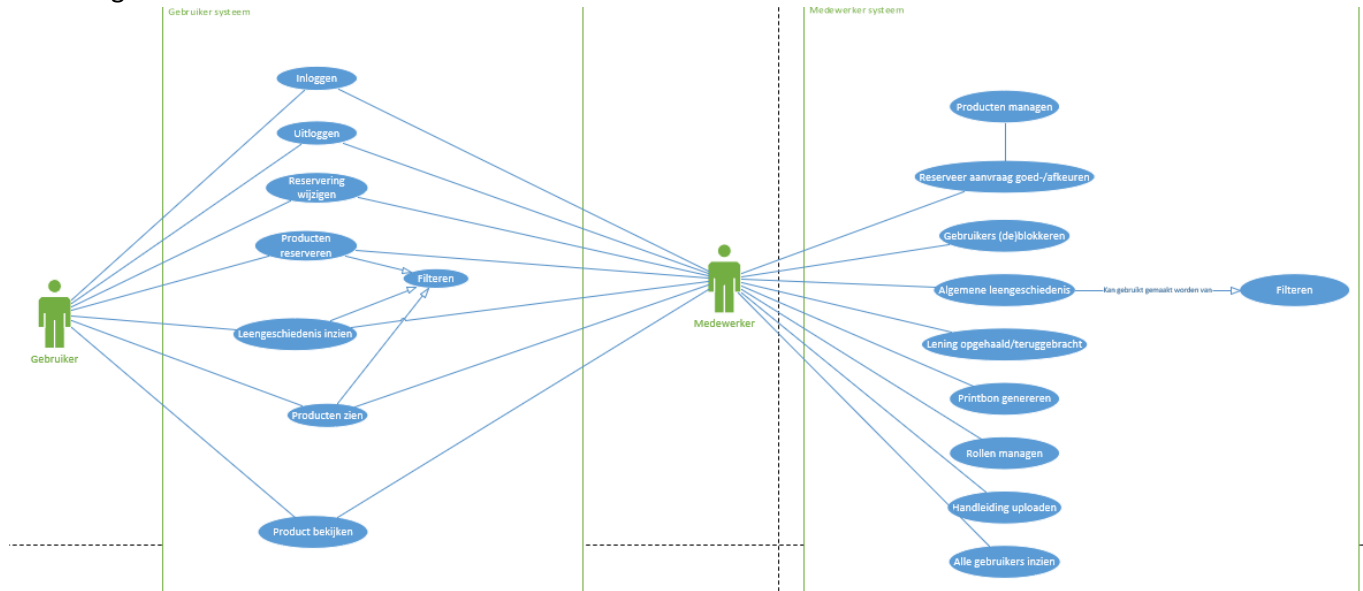
Een leerdoel in dit project is om software te bouwen die goed schaalbaar en uitbreidbaar is. Om dit mogelijk te maken wordt dit project opgezet op basis van microservices. Dit betekent dat alles wat een los component kan zijn ook als een los component gebouwd wordt.

Wanneer de backend is opgedeeld in losse componenten, moeten deze componenten met elkaar kunnen blijven communiceren. Hiervoor wordt een 'Message Broker' gebruikt.

Alle losse componenten worden gedeployed in een Docker container/image. Wanneer er dan meer capaciteit nodig is, kan er een extra Docker container gestart worden en deze zich aanmelden bij de load balancer. De load balancer verdeelt dan de requests tussen de losse componenten om zo het systeem vloeiend te laten werken.

5 Use case diagram

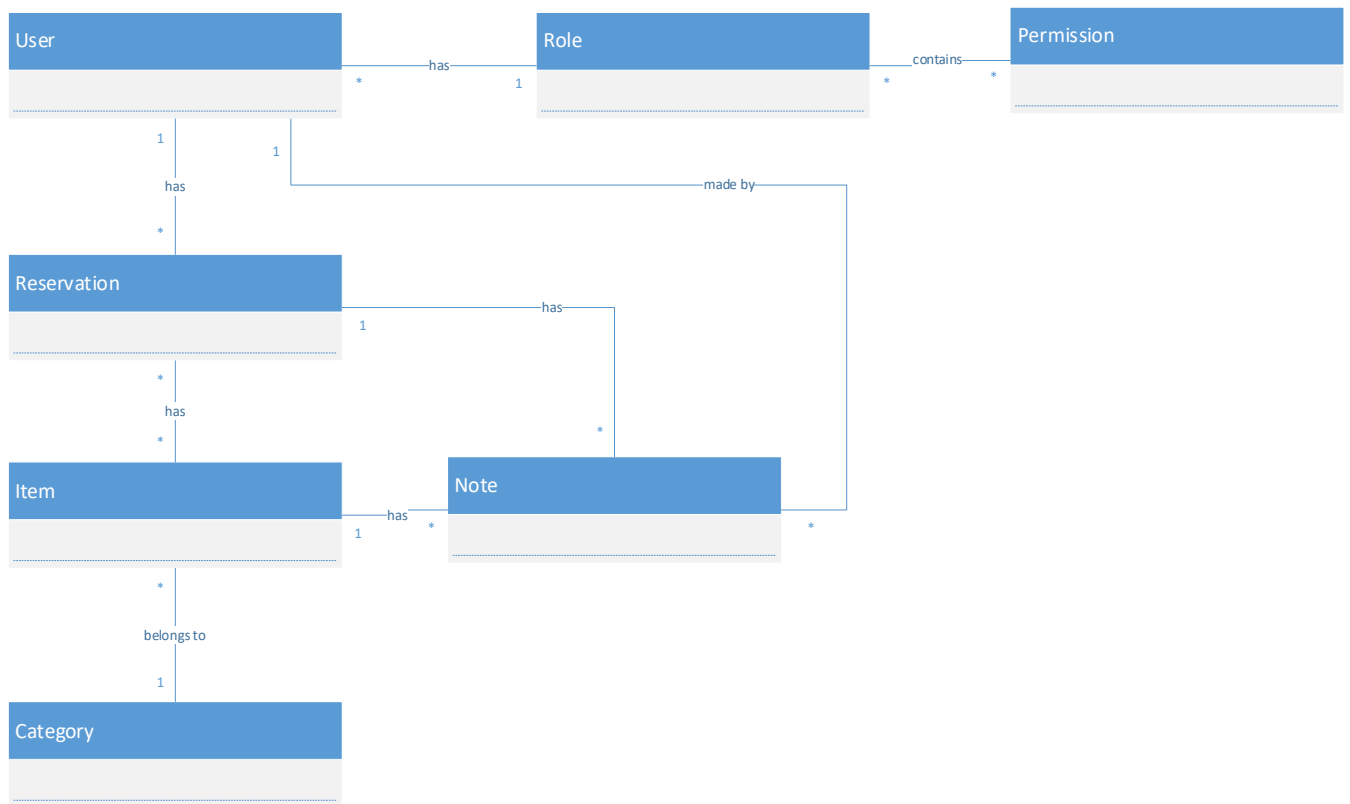
Om een duidelijk overzicht te geven van welke use cases er bij welke actor horen, is er een use case diagram gemaakt. Hierin staat met lijnen aangegeven welke use case er bij welke actor hoort. Dit is ook terug te vinden in de use cases zelf.



Figuur 2 Use case diagram voor het Fontys ACI rental systeem

6 Domeinmodel

In onderstaande model staan op een hoog niveau alle mogelijke actoren en classes genoteerd, met de daarbij horende relaties.



Figuur 3 Het domeinmodel van het Fontys ACI rental systeem