Reno Muijsenberg

r.muijsenberg@student.fontys.nl

S3 software portfolio

Festival planner



Contents

[Leeruitkomsten uitgelegd 4](#_Toc121829783)

[Aantonen leerdoelen 5](#_Toc121829784)

[1. Web application 5](#_Toc121829785)

[User story 1 (Quick overview of upcoming festivals) 5](#_Toc121829786)

[User story 2 (Detail page of upcoming festival) 6](#_Toc121829787)

[User story 3 (Able to add festival to Google calendar) 7](#_Toc121829788)

[Geleerd van Web Application 8](#_Toc121829789)

[Bijlage 8](#_Toc121829790)

[2. Software quality 9](#_Toc121829791)

[Tests 9](#_Toc121829792)

[Sonar Cloud 10](#_Toc121829793)

[Geleerd van Code quality 10](#_Toc121829794)

[Bijlage 10](#_Toc121829795)

[3. Agile method 11](#_Toc121829796)

[Sprint planning 11](#_Toc121829797)

[Daily stand-up/ stand-down 11](#_Toc121829798)

[Sprint review 11](#_Toc121829799)

[Sprint retrospectieve 11](#_Toc121829800)

[Geleerd van Agile method 12](#_Toc121829801)

[Bijlage 12](#_Toc121829802)

[4. CI/CD 13](#_Toc121829803)

[Builden en testen 13](#_Toc121829804)

[Quality check 13](#_Toc121829805)

[Docker hub 13](#_Toc121829806)

[Leesbaarheid 13](#_Toc121829807)

[Azure 13](#_Toc121829808)

[Geleerd van CI/CD 14](#_Toc121829809)

[Bijlage 14](#_Toc121829810)

[5. Cultuur 15](#_Toc121829811)

[Wat is cultuur? 15](#_Toc121829812)

[Wat zijn wel bekende dimensies van culturelere verschillen? 15](#_Toc121829813)

[Voorbeelden over culturele verschillen? 15](#_Toc121829814)

[Wat is mijn cultuur? 15](#_Toc121829815)

[Cultuur binnen GP-groep: 16](#_Toc121829816)

[Geleerd van cultuur 16](#_Toc121829817)

[Bijlage 16](#_Toc121829818)

[6. Requirements and design 17](#_Toc121829819)

[Stakeholder analysis 17](#_Toc121829820)

[Requirements analysis 17](#_Toc121829821)

[Softwaredesign and architecture 17](#_Toc121829822)

[UI design 17](#_Toc121829823)

[C4 model GP: 18](#_Toc121829824)

[C4 model IP 19](#_Toc121829825)

[Geleerd van Requirements and design 20](#_Toc121829826)

[Bijlage 20](#_Toc121829827)

[7. Business processes 21](#_Toc121829828)

[BPMN-model GP 21](#_Toc121829829)

[Geleerd van business processes 21](#_Toc121829830)

[Bijlage 21](#_Toc121829831)

[8. Professional 22](#_Toc121829832)

[Professioneel binnen groep 22](#_Toc121829833)

[Professioneel met stakeholder(s) 22](#_Toc121829834)

[Professioneel van mezelf 22](#_Toc121829835)

[Geleerd van professional 22](#_Toc121829836)

[Bijlage 22](#_Toc121829837)

[Links tussen leerdoelen 23](#_Toc121829838)

[Web Application & Software quality 23](#_Toc121829839)

[Web Application & Agile 23](#_Toc121829840)

[Web Application & CI/CD 23](#_Toc121829841)

[Web Application & Cultural differences and ethics 23](#_Toc121829842)

[Web Application & Requirements and design 23](#_Toc121829843)

[Web Application & Business process 23](#_Toc121829844)

[Web Application & Professional 23](#_Toc121829845)

[Software quality & Agile 24](#_Toc121829846)

[Software quality & CI/CD 24](#_Toc121829847)

[Software quality & Cultural differences and ethics 24](#_Toc121829848)

[Software quality & Requirements and design 24](#_Toc121829849)

[Software quality & Business process 24](#_Toc121829850)

[Software quality & Professional 24](#_Toc121829851)

[Agile & CI/CD 24](#_Toc121829852)

[Agile & Cultural differences and ethics 25](#_Toc121829853)

[Agile & Requirements and design 25](#_Toc121829854)

[Agile & Professional 25](#_Toc121829855)

[CI/CD & Cultural differences and ethics 25](#_Toc121829856)

[CI/CD & Requirements and design 25](#_Toc121829857)

[CI/CD & Professional 25](#_Toc121829858)

[Cultural differences and ethics & Requirements and design 25](#_Toc121829859)

[Cultural differences and ethics & Professional 25](#_Toc121829860)

[Requirements and design & Professional 26](#_Toc121829861)

# Leeruitkomsten uitgelegd

#### 1. Web application

**User friendly:** You apply best practices when creating user interfaces and basic user experience testing and development techniques.

**Full stack:** You design and build a full stack application using a commonly accepted front end JavaScript framework and back-end application implementing relevant communication protocols, persistence of data by usage of ORM and addressing asynchronous communication issues.

#### 2. Software quality

**Tooling and method:** Conduct, monitor, and report on unit integration, regression, and system tests, with attention for security and performance aspects, as well as applying static code analysis and code reviews.

#### 3. Agile method

**Choose:** You are aware of the most popular agile methods and their underlying agile principles. Your choice of a method is motivated and based on well-defined selection criteria and context analyses.

#### 4. CI/CD

**Design and implement:** You design a release process and implement a continuous integration and deployment solution (using e.g., Gitlab CI and Docker).

#### 5. Cultural differences and ethics

**Recognize:** Recognition is based on theoretically substantiated awareness of cultural differences and ethical aspects in software engineering.

**Consider:** Adapt your communication, working, and behavior styles to reflect project stakeholders from different cultures.

#### 6. Requirements and design

**Multiple types of test techniques:** You apply user acceptance testing and stakeholder feedback to confirm the quality of the requirements. You evaluate the quality of the design (e.g., by testing or prototyping) considering the formulated quality properties like security and performance.

#### 7. Business processes

**Simple:** Involving stakeholders, predominantly sequential processes with one or two alternative paths.

**Related:** Business processes during which the software that you are developing will be used (business processes that the software must support by fully or partially automating them).

#### 8. Professional

**Professional manner:**You develop software as a team effort according to a prescribed software method and following team agreements. You can track your work progress and communicate your progress with the team.

You actively ask and apply feedback from stakeholders and recommend them on the best technical and design (architectural) solutions.  
You choose and substantiate solutions for a given problem.

# Aantonen leerdoelen

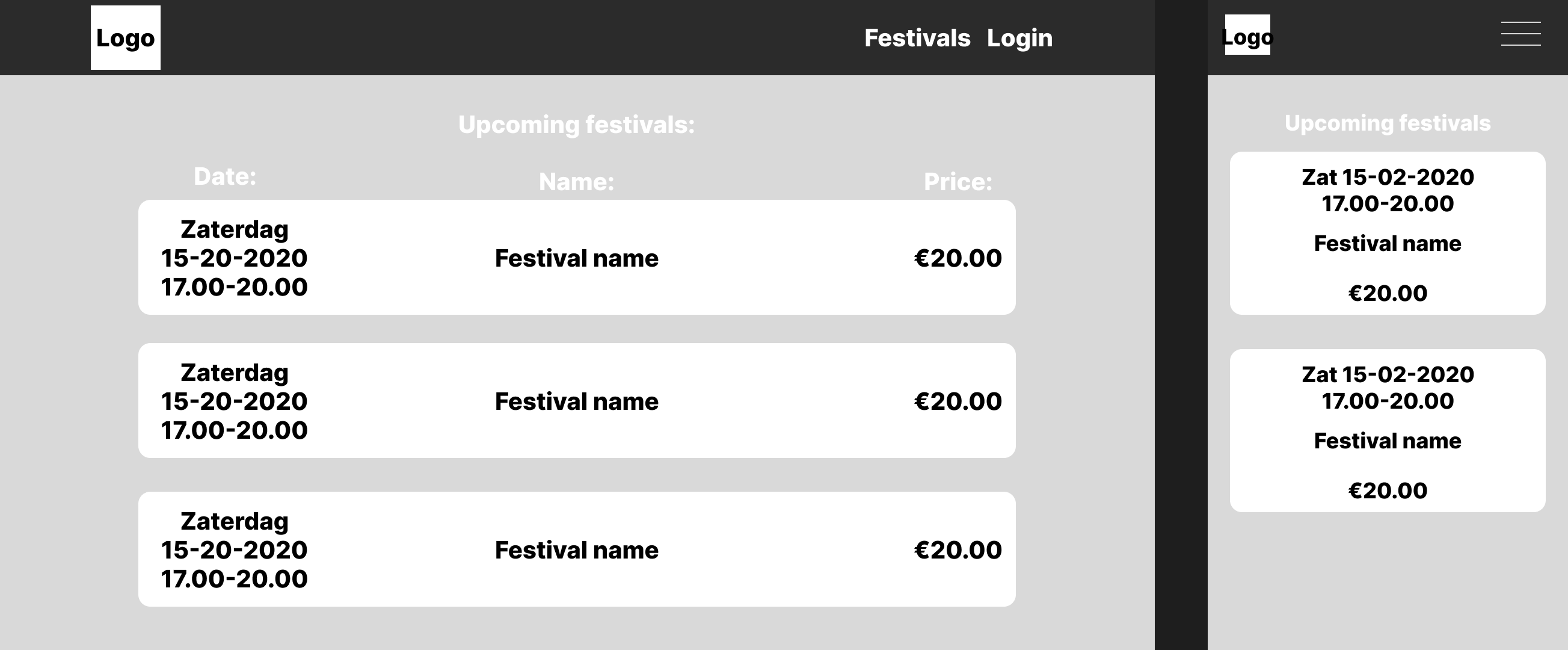
## [1.](#_1._Web_application) Web application

Ik heb gekozen om het front-end framework Angular te gebruiken ([uitgewerkte front-end keuze](https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular/wiki/Front-end-choice#front-end-language-choice)) en als backend heb ik ervoor gekozen om ASP.net te gaan gebruiken.

### User story 1 (Quick overview of upcoming festivals)

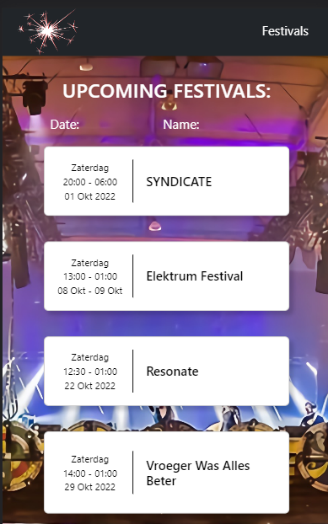
#### UI design

Na het kiezen van mijn programeer talen ben ik begonnen met het designen van de pagina die ik voor de eerste userstory nodig heb. Ik heb een snel efficiënt overzicht gemaakt van de festivals waardoor de gebruiker makkelijk kan zien wat de opkomende festivals zijn.



Wireframe quick-overview front page

#### UI website

Vervolgens ben ik begonnen met het opzetten van het front-end project en ben ik een [video tutorial](https://www.youtube.com/watch?v=k5E2AVpwsko&t=3408s) gaan kijken over hoe Angular werkt. En heb met de kennis van deze tutorial de pagina van de wireframe nagemaakt in HTML, CSS (Screenshot bevat hardcoded data).

Graphical user interface, website

Description automatically generated

Front page quick-overview in HTML, CSS

#### API dataBackend API-data

Nadat ik de hoofdpagina had gemaakt ben ik begonnen met het opzetten van de festival service. Ik ben hier mee begonnen zodat ik dynamische content op de hoofdpagina kan tonen. Ik he been project gestart in ASP.net en de endpoint gemaakt voor het ophalen van alle festivals door middel van ORM, ik heb dit voor nu met hardcoded data in de backend gedaan zodat ik kon testen of ik het kon ophalen via een HTTP request. Toen ik zag dat de endpoint werkte heb ik een database opgezet en deze geconnect met de app zodat de data ook echt uit de database komt.

API-data user story 1

### User story 2 (Detail page of upcoming festival)

Voor user story 2 is het de bedoeling dat wanneer een gebruiker op een festival klikt hij/zij wordt gebracht naar een pagina met meer details over dit bepaald festival. Hiervoor moest ik dus een nieuwe pagina designen.

#### UI design

Aangezien deze pagina over maar een bepaald festival gaat heb ik meer ruimte om te besteden. Ik heb een snel design gemaakt wat niet alle data bevat wat uiteindelijk op de website staat aangezien ik nog niet zeker wist hoe ik dit wou gaan neerzetten. Ik ben op het volgende design uitgekomen waar de witte plek nog verder benut wordt.

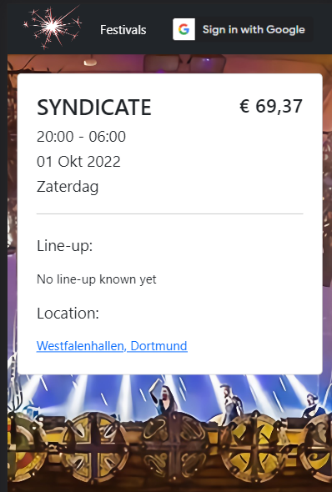
Afbeelding met tekst

Automatisch gegenereerde beschrijving

UI-design festival detail page

#### UI website

Vervolgens heb ik dit gemaakt in de Angular, al ziet dit er nog een beetje leeg uit zou erna maten dat ik meer data over een festival toevoeg dit ook minder leeg worden. Vooral wanneer de line-up aanwezig is zou deze plek beter benut worden.

Afbeelding met tekst, taart, chocolade, schermafbeelding

Automatisch gegenereerde beschrijving

Front page detail in HTML, CSS

#### Backend API-data

Vervolgens was het enige wat er nog moest gebeuren dat de data dynamisch vanuit de API wordt opgehaald. Ik heb in mijn ASP.net project een endpoint aangemaakt om een specifiek festival op te halen aangezien ik al de endpoint voor meerdere festival op te halen had was dit zo gedaan. Het backend project heb ik omgezet in drie lagen de controller laag, de service laag en de data laag zodat die goed te testen en onderhouden is. Dit alles resulteert in de data van een specifiek festival.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

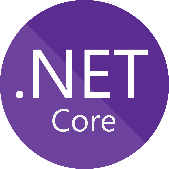
API-data user story 2

### User story 3 (Able to add festival to Google calendar)

Voor user story 3 is het de bedoeling dat een gebruiker met Google kan inloggen en vervolgens een festival in zijn/haar kalender kan zetten met een klik op een knop. Voor deze user story heb ik eerst een POC gemaakt voor zowel het inloggen als het toevoegen aan de kalender. Ik heb besloten om de login knop te verwerken in de navigatie bar van de applicatie aangezien dit een common practices is. Wanneer er op de knop geklikt wordt zou het standaard Google login schermpje verschijnen. Dit schermpje zou vervolgens vragen of de applicatie rechten mag hebben van de gebruikers zijn/haar kalender. Wanneer iemand ingelogd is en op een knop klikt voor het toe te voegen aan de kalender wordt er een call naar google gestuurd met de datum en word deze datum in de kalender gezet.

Navbar met google login

### Geleerd van Web Application

Dit semester heb ik ervoor gekozen om een nog onbekend voor mij JavaScript framework te kiezen als front-end van mijn applicatie. Ik had hiervoor gekozen omdat ik van alle framework een feel wou krijgen zodat ik in de toekomst makkelijk kan kiezen welke het beste/makkelijkste werkt voor mij.

Verder heb ik geleerd om te werken met een aantal Google-services aangezien mijn applicatie te maken had met het toevoegen van data in een kalender door middel van een knop op mijn website. Ik heb geleerd en Google login toe te voegen en te configureren in mijn applicatie en vervolgens het vragen aan een gebruiker voor toegang voor dingen in zijn/haar kalender toe te voegen en ook daadwerkelijk deze dingen toevoegen. Ik heb van deze Google dingen dan ook POC’s gemaakt zodat ik de volgende keer kan kijken hoe ik had gedaan en dat weer te gebruiken.

Verder heb ik uiteraard ook nog mijn kennis van .NET core web API’s uitgebreid met dit project maar hier was ik al wat bekender mee.

### Bijlage

* GitHub repository front-end (<https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular>)
* GitHub repository back-end (<https://github.com/RenoMuijsenberg/S3-Festival-Service>)
* UI wireframe: (<https://www.figma.com/file/gGmgoPGxSWRXgjk7fLqMJq/Wireframe-festival-planner>)

## [2. Software quality](#_2._Software_quality)

### Tests

Afbeelding met tekst

Automatisch gegenereerde beschrijvingVoor de kwaliteit van de applicatie zo goed mogelijk te maken heb ik ervoor gezorgd dat de services in de API getest zijn aangezien hier de logica gebeurt. Om ervoor te zorgen dat het front-end ook echt data binnenkrijgt heb ik voor het front-end ook unit test geschreven. Bij het testen wordt mock data gebruikt om te testen of deze ook werken met andere backend systemen. Ik heb getest op basis van de userstory’s en acceptance criteria zo heb ik bijvoorbeeld deze userstory met acceptance criteria.

Acceptance criteria user story 1

Afbeelding met tekst

Automatisch gegenereerde beschrijvingZo heb ik bijvoorbeeld bij user story 1 dat de back-end data van de database moet kunnen ophalen. Dit heb ik getest met een mock van de repository. Door de mock gebruikt ik niet de repository zelf maar simuleer ik deze. In het arrange stukje van de test zorg ik dat de repository wordt gesimuleerd en ik zorg ervoor dat die repository de correcte data teruggeeft. Daarna roep ik die functie van de repository aan en dan kijk ik of de data die ik heb aangegeven overeenkomt met de data die ik terugkrijg.

Acceptance criteria 1 van user story 1 test

Afbeelding met tekst

Automatisch gegenereerde beschrijvingVervolgens heb ik ook nog getest in de front-end of de data kan worden opgehaald vanuit de back-end. Dit heb ik weer met een mock gedaan. Bij deze heb ik de functie met de http-call gemockt zodat er geen echt call wordt gemaakt. Vervolgens heb ik ook nog gecheckt of deze call maar een keer wordt uitgevoerd om ongewenst dubbele data te ontvangen.

Acceptance criteria 2 van user story 1 test

### Sonar Cloud

Ik heb in de CI/CD voor mijn beide projecten Sonar Cloud toegevoegd. Ik heb dit gedaan zodat wanneer ik mijn code push naar GitHub er een analyse van de code wordt gemaakt. Dit scant dan of er bijvoorbeeld bugs/vulnerabilities zijn. Ik check dan regelmatig na een push wat de score is en of ik iets moet aanpassen/oplossen in mijn code zodat ik de best mogelijke score krijg. Wanneer er een onveiligheid in de code is krijg ontvang ik hier ook een email van zodat ik dit direct kan oplossen. Verder heb ik de instellingen aangepast zodat de test coverage ook echt alleen de bestande checkte die ik ook echt daadwerkelijk getest wil hebben.



Sonar Cloud scanner front-end project

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Sonar Cloud scanner backend project

### Karma · GitHubGeleerd van Code quality

Ik heb dit semester een aantal dingen geleerd als het om code quality gaat. Zo heb ik mijn tests geschreven met behulp van Karma en Jasmine omdat dit inbegrepen zat bij Angular.

Karma is een tool die webserver aanmaakt die de code van een project tests en de geschreven tests in deze browser runt. Jasmine is een open-source testing framework dat vaak wordt gebruikt om JavaScript/TypeScript applicaties te testen. Met behulp van Jasmine is het ook mogelijk om mocks te maken van bepaalde functies in code.

Verder heb ik uiteraard geleerd hoe ik Sonar Cloud toe kan passen in een JavaScript en .NET core project kan toevoegen. Ik heb geleerd hoe ik kan instellen wat de criteria zijn voor de code coverage en ook welke bestanden excluded worden in de coverage. Ik heb regelmatig gekeken naar deze scanner zodat mijn code zo optimaal is gemaakt.

### Bijlage

* Sonar Cloud front-end (<https://sonarcloud.io/project/overview?id=RenoMuijsenberg_S3-Festival-Planner-Angular>)
* Sonar Cloud back-end (<https://sonarcloud.io/project/overview?id=RenoMuijsenberg_S3-Festival-Service>)

## [3. Agile method](#_3._Agile_method)

Afbeelding met tekst, buiten, schermafbeelding

Automatisch gegenereerde beschrijvingVoor de proftaak werken wij de scrum methode. We werken aan ons project in sprints van 3 weken. Voor de user story’s gebruiken wij een Github bord dat we hebben ingedeeld met de categorieën: “User Story’s”, “Backlog”, “To Do This Sprint”, “In Progress,” “In Review” en “Done”

User story's met story points

### Sprint planning

Elke sprint beginnen we met een meeting met de product owner en gaan we samen kijken welke user story’s komende sprint gaan worden opgepakt en welke de hoogste prioriteit hebben. Dit wordt in overleg gedaan zodat we kunnen afstemmen met de product owner wat haalbaar is voor komende sprint. Dit besluit wordt genomen gebaseerd op het aantal story points wat wij sprints eerder behaald hebben. Deze story points zijn relatieve inschattingen van hoe lastig dat wij denken dat een user story gaat worden. Deze punten geven wij aan de user story’s voordat we in gesprek gaan met de product owner.

### Daily stand-up/ stand-down

Elke dag dat we gaan werken aan dit project starten we met een stand-up. In deze stand-up gaan we samen kijken wat iedereen deze dag gaat doen. We pakken ons bord erbij en verdelen de issues over de mensen die aanwezig zijn. Dit gesprek wordt geleid door de scrum master. De scrum master in onze groep verwisselt elke sprint zodat iedereen een keer aan de beurt is geweest. De scrum master let dan ook op of het doel wat we hebben vastgesteld te halen is en of er nog extra werk moet gedaan worden. Op het einde van de dag wordt er rond gevraagd en op het github bord gekeken wat iedereen af heeft en waarom iemand iets niet af heeft terwijl hij/zij dit wel als taak had voor deze dag. Als blijkt dat de taak te moeilijk was voor de persoon dan spreken we af om de volgende dag erna te kijken met iemand die hier iets meer verstand van heeft.

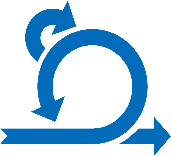
### Sprint review

In de sprint review gaan we opleveren en presenteren wat we afgelopen sprint gehaald hebben. We doen dit aan de hand van een presentatie en een kleine demo van onze app. In de presentatie leggen we uit wat onze doelen waren deze sprint, welke van deze doelen we wel of niet gehaald hebben en dan ook waarom we deze niet gehaald hebben. Ook leggen we bepaalde keuzes uit die we gemaakt hebben en onderbouwen we waarom we voor dit hebben gekozen. De product owner geeft ons dan feedback/ complimenten over wat we afgelopen sprint gedaan hebben en ze stellen ons nog eventuele vragen als iets niet duidelijk was.

### Sprint retrospectieve

Aan het einden van onze sprint review dan houden we met ons groepje een retrospectieve, in deze retrospectieve bedenken we eerst allemaal individueel minimaal drie dingen die goed gingen deze sprint en drie dingen die minder goed gingen deze sprint. Vervolgens vullen we deze punten in op Github bord genaamd “Retrospectieve sprint (1, 2 etc.)”. De scrum master leest deze punten op en vraagt aan de persoon of hij/zij hier toelichting bij kan geven. De persoon legt dit dan uit en dan bedenken we met z’n allen wat we hier komende sprint aan kunnen veranderen of het juist zo laten. Dit wordt dan ook op het Github bord ingevuld onder het kopje “Action points”.

### Geleerd van Agile method

Uiteraard heb ik geleerd om op een professionele manier in een team te werken door middel van de scrum methode. Hier heb ik van geleerd om goed te plannen voordat je begint met het programmeren. Wij hebben gekozen om elke sprint een nieuwe scrum master te hebben zodat we hier allenmaal een beetje van kunnen proeven. Toen ik scrum master was heb ik natuurlijk een stukje communicatie geleerd omdat jij als scrum master het project naar de eind weg wilt brengen, je moet dus goed kunnen kijken of mensen die misschien iets minder kennis hebben of daar alles goed mee gaat en dit anders ook kunnen zeggen tegen hen. Verder heb ik geleerd om met scrum poker te bepalen hoe groot een user story nu is zodat we goed onze sprints kunnen afstellen aan wat we aankunnen per sprint.

### Bijlage

* Scrum bord IP (<https://github.com/users/RenoMuijsenberg/projects/1>)
* Scrum bord GP (<https://github.com/orgs/S3-Proftaakgroep-3/projects/1>)
* Retrospectieve
  + <https://github.com/orgs/S3-Proftaakgroep-3/projects/2>
  + <https://miro.com/app/board/uXjVPFunwoo=/>
  + <https://github.com/orgs/S3-Proftaakgroep-3/projects/4>

## [4. CI/CD](#_4._CI/CD)

### Builden en testen

Voor de CI/CD ben ik begonnen met het kijken van een [tutorial](https://www.youtube.com/watch?v=R5ppadIsGbA). Ik heb hier met mee gevolgd en had uiteindelijk een CI die het project deed builden en testen. Aangezien deze tests nu nog leeg zijn liet hij dit slagen. Hierna heb ik een aantal tests geschreven zodat ik kon valideren of het ook echt werkte dit gebeurt op elke branch aangezien je het overal nodig hebt.

### Quality check

Graphical user interface

Description automatically generatedNadat ik dit werkend heb ben ik begonnen met het toevoegen van Sonar Cloud (een staties code analyse tool). Ik heb deze tool toegevoegd in de CI zodat elke keer wanneer ik een push maak naar mijn code dit gescand wordt op bijvoorbeeld sensitieve data zoals een connection string dit gebeurt op elke branch aangezien je het overal nodig hebt.

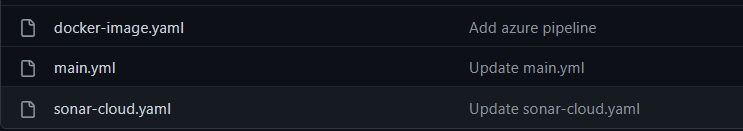
CI/CD

### Docker hub

Hierna ben ik gaan kijken hoe ik mijn container image op een push naar Docker hub kon pushen. Dit heb ik gedaan aan de hand van de [documentatie van Docker](https://docs.docker.com/language/java/configure-ci-cd/). Wanneer er naar de “Main” of “Dev” branch wordt gepusht wordt de Docker image naar Docker hub gepusht.

### Leesbaarheid

Na het maken van dit stuk CI ben ik gaan kijken hoe ik dit overzichtelijk kon maken dat bijvoorbeeld een taak niet afhankelijk was van een andere taak en dat het leesbaarder werd in plaats van alles in een bestand. Ik heb dit gedaan door de workflow bestanden op te delen in eigen bestanden. Zo heb ik een workflow gemaakt voor het builden en testen, de Sonar Cloud scan en voor de image te pushen naar github.

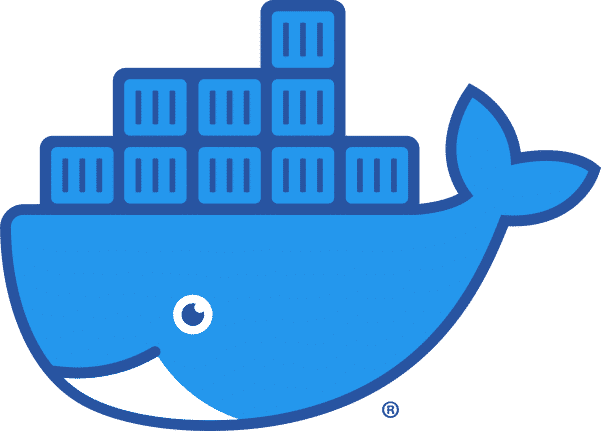


### Azure

Nadat ik de CI had staan wou ik ervoor zorgen dat mijn website naar een live server gepusht wordt elke keer dat er maar de “Main” branch wordt gepusht. Ik ben eerst gaan kijken naar een aantal opties waar ik de server op kon hosten. Ik heb gekeken naar Heroku, Digital Ocean en Azure. De eerste twee opties vielen al af aangezien ik hier geld voor uit moest geven en/of mijn creditcard informatie moest invullen dus heb ik gekozen voor het gebruik van Microsoft Azure aangezien we hier $100 per jaar ontvangen. Ik ben naar een aantal opties gaan kijken hoe we onze code via de Docker image op Azure konden zetten en ben uiteindelijk uitgekomen op [deze tutorial](https://circleci.com/blog/deploy-dockerized-dotnet-core-to-azure/). Ik heb deze tutorial gevolg voor het opzetten van de server en de container registry allen deze tutorial had niet het github actions stuk wat ik nodig had om het te pushen op push naar de “main” branch dus ben ik verder gaan zoeken en heb ik [deze tutorial](https://brendanthompson.com/posts/2022/02/using-github-actions-to-publish-container-images-to-azure-container-registry) gevolgd met aanpassing naar mijn eigen wensen. Dit zorgde er voor dat elke keer data er naar de “Main” branch wordt gepusht wordt de Docker image naar de Azure registry gepusht en vervolgens ziet de server dit en wordt de image gebuild op de server waardoor de server live staat en ik CD heb toegepast.

### GitHub Actions · GitHubGeleerd van CI/CD

Van CI/CD had ik voor dit semester nog geen enkele kennis op dat het bestond na, ik heb dus alles wat ik dit semester in de CI/CD heb zitten geleerd. Ik heb geleerd om een Angular en een .NET core applicatie te builden en te testen door middel van de CI/CD.

 Verder heb ik geleerd om de Sonar Cloud quality check toe te voegen aan de CI/CD. Deze quality scan was in het voorbeeld in een Windows omgeving gemaakt maar ik heb die omgezet naar een Ubuntu omgeving. Ook heb ik nog toegevoegd dat de test coverage wordt gemeten in de CI/CD zodat deze ook op Sonar Cloud wordt weergegeven.

Verder heb ik geleerd hoe ik een Docker image door middel van de CI/CD naar Docker hub kan pushen en ik heb nog geleerd hoe ik mijn Docker image naar een Azure container registry kan pushen die vervolgens wordt gedeployed op een live server van Azure.

### Bijlage

* Workflow bestanden front-end (<https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular/tree/main/.github/workflows>)
* Workflow bestanden back-end (<https://github.com/RenoMuijsenberg/S3-Festival-Service/tree/main/.github/workflows>)
* Azure website back-end (<https://festival-service-api.azurewebsites.net/api/festival/festivals>)
* DockerHub image front-end (<https://hub.docker.com/repository/docker/renomuijsenberg/festival-angular>)
* DockerHub image back-end (<https://hub.docker.com/repository/docker/renomuijsenberg/festival-service>)

## [5. Cultuur](#_5._Cultural_differences)

### Wat is cultuur?

Cultuur zijn de gedragsregels en normen die vallen binnen een bepaalde bevolking of stam. Zo heb je bijvoorbeeld in Frankrijk dat afhankelijk van de plek waar je bent je bij een begroeting elkaar een aantal keer op de wang zoent. De hoeveelheid van deze zoenen hangt af van waar je bent in Frankrijk, in een ander cultuur kan dit juist heel vreemd zijn.

### Wat zijn wel bekende dimensies van culturelere verschillen?

#### Machtsafstand

Machtsafstand wordt gezien als de mate waarin de minder machtige leden/werknemers accepteren dat de macht minder gelijkt verdeeld is in een land of organisatie. Als deze verwachting laag is dan wordt dit gezien als een kleine machtsafstand. Is dit hoog dan wordt dit gezien als een grote machtsafstand.

#### Individualisme versus collectivisme

Een samenleving is individualistisch wanneer banden tussen individuen ongedwongen zijn. In deze samenleving zorgen mensen voor zichzelf en voor zijn/haar familie. Een samenleving is collectivistisch wanneer mensen vanaf hun geboorte opgenomen worden in groepen waar mensen levenslang voor elkaar zorgen in de ruil voor loyaliteit.

#### Femininiteit versus masculiniteit

Masculiniteit is wanneer de emotionele seks rollen gescheiden zijn. Bij mannen ligt het accent op prestatie en succes en bij vrouwen licht de nadruk op bescheidenheid, tederheid en de kwaliteit van het bestaan. Een samenleving is feminien wanneer de seks rollen elkaar overlappen. Zowel mannen als vrouwen dienen bescheiden en teder te zijn.

#### Onzekerheidsvermijding

Onzekerheidsvermijding is de mate waarin mensen in een samenleving zich bedreigd voelen door onzekere of onbekende situaties en deze situaties proberen te vermijden. Landen met een hoog onzekerheidsvermijding hebben behoefte aan regels en formaliteit.

#### Termijngerichtheid

Kortetermijngerichtheid is het nastreven van positieve eigenschappen gericht op het verleden en het heden, respect voor tradities en het voldoen aan sociale verplichtingen. Langetermijngerichtheid staat voor het streven naar beloningen in de toekomst, vooral via vasthoudendheid en spaarzaamheid.

### Voorbeelden over culturele verschillen?

Als voorbeeld heb je bij een groet vaak culturele verschillen. Zo heb je in Japan dat het normaal is om te buigen bij een groet, bij afscheid, als je iemand bedankt en als je je wilt verontschuldigen.  In Thailand buig je je hoofd: houd je handen voor je borst en leg je handpalmen tegen elkaar aan. Buig nu je hoofd totdat je je vingertoppen raakt. In Italië is het dan weer normaal om geknuffeld of gezoend te worden. In andere landen is de hoeveelheid van kussen weer anders dan in andere landen.

### Wat is mijn cultuur?

Zelf kom ik uit het Nederlandse Limburg. Hier zijn veel cultuur verschillen te zien in vergelijking met Nederland. Het bekendste verschil is natuurlijk wel dat een verjaardag gehouden wordt met een stukje vlaai. Ook zijn er veel verschillende dialecten te vinden, per dorp zijn er wel woorden die anders worden uitgesproken zoals het woord koud kan op de volgende manieren uitgesproken worden: koad, kalt, kaat etc. Ook wordt in Limburg carnaval optimaal gevierd, iedereen verkleedt zich en er is een bepaalde muziekstijl die je nergens anders hoort met carnaval wordt hier dan ook het LVK (Limburgs Vastelaovesleedjes Konkoer) voor gehouden.

### Cultuur binnen GP-groep:

Binnen de proftaak groep hebben we in het begin van het semester een aantal regels met elkaar afgesproken bijvoorbeeld over het te laat komen. Deze afgesproken regels vormen ook een soort cultuur. Zo hebben we de volgende afspraken:

* Wanneer iemand 20 minuten te laat is en het komt niet door het openbaar vervoer wordt dit bijgehouden aan het eind van de proftaak legt deze persoon dan 6-pack bier in voor het aantal keer dat hij/zij te laat is gekomen. En wordt dit samen opgedronken.
* Wanneer er een onenigheid is wordt er door iedereen gestemd en de meeste stemmen gelden.

### Geleerd van cultuur

Ik heb geleerd dat er heel veel vormen van cultuur bestaan. Een cultuur kan een hele grote of juist een klein groepje van mensen bestaan. Zo kun je een cultuur bijvoorbeeld Nederlanders pakken, dit zijn vaak mensen die wat gierig zijn met hun geld en bekend staan door het eten van kaas. Maar je kunt ook een klein groepje cultuur hebben zoals bijvoorbeeld bij de proftaak. Regels en normen die we hebben afgesproken maken een soort cultuur zo heb je bijvoorbeeld dat elke dag voordat we aan de proftaak beginnen iedereen te laat in verband met openbaar vervoer. Dit zijn allemaal dingen waar rekening mee gehouden moet worden in je project. Zorg dus dat je project globaal accessibel is zodat ook mensen van een ander cultuur dit zouden kunnen overnemen.

### Bijlage

* Cultuur afspraken binnen GP (<https://github.com/S3-Proftaakgroep-3/Mediaan-DMA/wiki/Samenwerkingsovereenkomst-Mediaan-DMA>)

## [6. Requirements and design](#_6._Requirements_and)

### Stakeholder analysis

Voor de proftaak hebben we een stakeholder analysis gemaakt, we hebben gekeken naar wie er allemaal een rol speelde in het maken, gebruiken en beheren van het project tijden het maken en tijdens de release en wat de ze mensen wilde bereiken met het project.

* Customers: Restaurants
  + Wants an application so that their customers can order food.
  + Want an application that their employees can use for managing orders.
* Product-owner: Mediaan
  + Wants an application that can be delivered to their customers.
  + Communicates needs from customer to project team members.
* Project team members: MDMA (me)
  + Wants to develop an application to keep the product owner happy and to keep the promises of the product owner to the customer.
* Users: People eating at the restaurant and people working at a restaurant
  + Wants to order food in the restaurant.
* Users: Employees of the restaurant
  + Want an application that is for managing customer orders keeping track of the status.

### Requirements analysis

We zijn in ons groepsproject begonnen met het bedenken van de requirements vanuit de opdracht. We hebben met z’n alle de opdracht beschrijving doorgelezen en hier requirements uitgehaald en omgezet in issues op ons github bord.

In de proftaak zijn de requirements voor ons klaargezet in een vorm van een verhaal die de klant vertelde. Hieruit hebben wij user story’s gemaakt en hier acceptance criteria aan toegevoegd. Vervolgens hebben wij deze besproken met de product owner en op basis van hun feedback hebben we deze een klein beetje aangepast. Vervolgens hebben we de user story’s geprioriteerd. Daarna zijn we samen gaan zitten met onze groep en hebben we van de user story’s tickets gemaakt van specifieke taken. Dit is hoe wij de requirements deze proftaak hebben gemaakt.

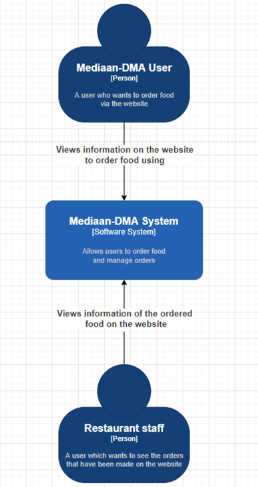
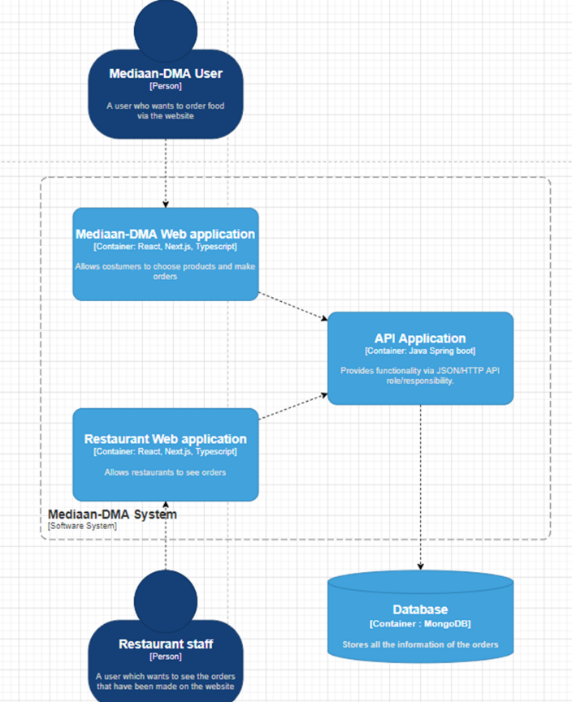
### Softwaredesign and architecture

Wanneer we een keuze gaan maken over de architectuur zoals bijvoorbeeld een taal die we gebruiken of een database die we gebruiken hebben we met elkaar besproken, wanneer iedereen het eens is met deze keuze gebruiken we hem pas. Als iemand het niet eens is met de keuze die we gemaakt hebben dan wordt er overlegd waarom hij/zij deze keuze wel/niet wil en wat daar de argumenten voor zijn. Keuzes die we dan gemaakt hebben binnen ons project laten we in de sprint oplevering zien aan de product owner en leggen we uit waarom we deze keuze gemaakt hebben. Verder hebben we de architectuur nog vastgelegd door middel van een C4 model.

### UI design

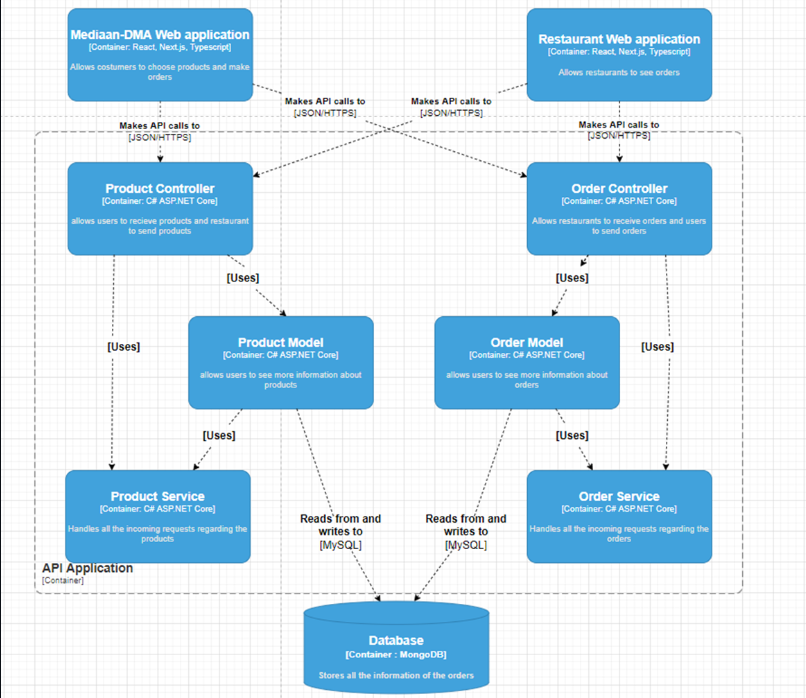
Voor de UI design van onze applicatie zijn we gaan kijken naar applicaties die al bestaan. Aangezien onze proftaak applicatie een app is waar je eten kunt bestellen hebben we gekeken naar soort gelijke applicaties waar je eten kunt bestellen. Zo hebben we voor het design gekeken naar Thuisbezorgd, Uber eats etc. Hieruit is een design gekomen wat niet te moeilijk is voor mensen om te gebruiken maar waar je wel nog genoeg informatie hebt over wat het product dat verkocht wordt is.

### C4 model GP:



C2

C1



C3

### C4 model IP

Ik heb een start gemaakt aan het maken van het C4 model. Ik heb als voorbereiding een [video](https://www.youtube.com/watch?v=x2-rSnhpw0g) gekeken over een conferentie waar dit uitgelegd werd. En vervolgens heb ik C1, C2 en C3 gemaakt.

Diagram

Description automatically generated

C4 model V1

In de eerste sprint had ik feedback gevraagd op mijn C4 model. Tijdens de feedback kwam ik erachter dat ik C2 en C3 door elkaar had gehaald dus heb ik van deze twee modellen er een gemaakt. Verder heb ik wat kleurtjes gegeven aan de componenten zodat dit nog duidelijker is.

Diagram

Description automatically generated

C4 model V2

Nadat ik ben begonnen met het maken van een Python web scrapper was er ook een van mijn service niet meer nodig. Eerst wou ik het zo maken dat de data door een admin gebruiker moest worden ingevuld waardoor ik dus een accountservice nodig had omdat de admin moest kunnen inloggen. Nu ik de scrapper gebruik is deze service dus niet nodig en heb ik hierop het C4 model aangepast. Verder heb ik nog in plaats van eigen gemaakt vormen de standaard gebruikte C4 model vormen gebruikt.

Diagram

Description automatically generated

C4 model V3

### Geleerd van Requirements and design

Van requirements and design heb ik een aantal dingen geleerd. Ik ben dit semester begonnen met het uitzoeken van hoe een C4 model in elkaar zit. Deze heb ik vervolgens voor mij IP en GP gemaakt tot C3 aangezien C4 te veel in detail gaat. Verder heb ik voor mezelf over het groeps-project een stakeholder analysis gemaakt. Ik heb gekeken wie de betrokken partijen waren en wat deze partijen wilde bereiken met het maken van deze applicatie.

De requirements analyse hebben we gedaan samen met de product owner en ons groepje. Uiteraard heb ik hieraan meegeholpen en hebben we ook input gegeven aan ons product owners als we bijvoorbeeld dachten meer/minder aan te kunnen.

De keuze die we hebben gemaakt hebben we samen over gespard met ons team. We hebben gekeken welke talen waarom wel of juist niet wouden gebruiken. En hebben toen gekozen voor de sterkste argumenten. Dit hebben wij vervolgens omgezet en vastgelegd in het C4 model. Ik heb hiervan geleerd dat het niet altijd kan zoals je wilt als anders wat anders willen en dit moet dan ook geaccepteerd worden.

Als laatste voor het UI design heb ik geleerd om te developen op basis van een design. Mike heeft in ons groepje het UI design gemaakt op basis van bestaande applicaties. Hij heeft onze feedback gevraagd wanner hij iets had gemaakt of wok het hier allemaal mee eens waren. En wij hebben dit design mooi gevolgd.

### Bijlage

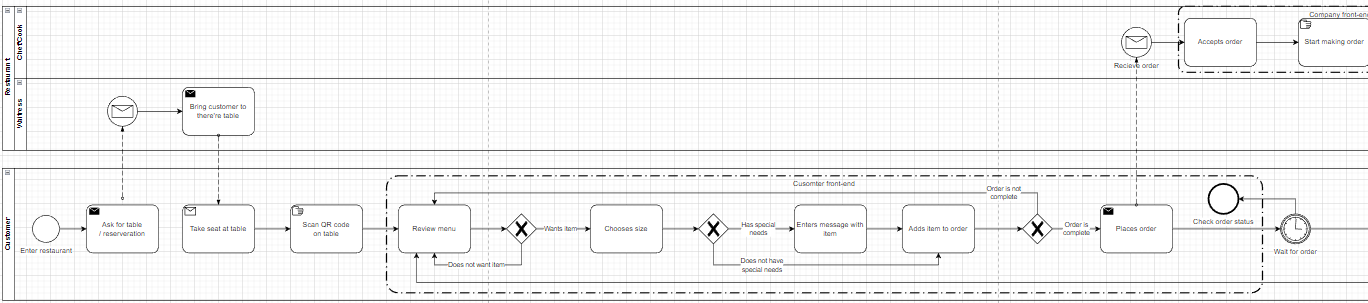
* C4-model GP (<https://github.com/S3-Proftaakgroep-3/Mediaan-DMA/wiki/C4-model>)
* C4-model IP (<https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular/wiki/C4-model>)
* UI design GP (<https://github.com/S3-Proftaakgroep-3/Mediaan-DMA/blob/main/MDMA.xd>)

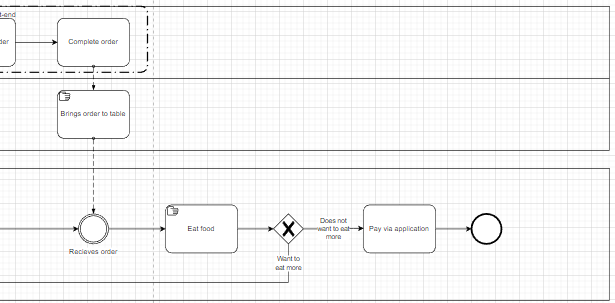
## [7. Business processes](#_7._Business_processes)

Ik ben begonnen met het uitzoeken wat de BMPN-notatie is. Ik kwam een video tegen van de makers van Lucidchart. Deze video ging over het bestellen en krijgen van ijs. Uit deze video heb ik de kennis gehaald van wat de icoontjes/taken allemaal betekenen vervolgen heb ik dit omgezet in een BPMN-model.

### BPMN-model GP

Voor de proftaak heb ik voor mezelf een BPMN-model gemaakt. In dit model heb ik de milestones verwerkt die wij ook echt daadwerkelijk moeten maken. Ik heb de volgende lanes gemaakt: “Customer”, “Chef/Cook” en “Waitress”. Ik heb het proces omschreven in dit model hoe het zou gaan als je een restaurant zou gaan eten waar onze applicatie gebruikt wordt. ([Link naar draw.io bestand](https://github.com/RenoMuijsenberg/S3-Software/blob/main/BPMN-proftaak.drawio) (Deze kan gedownload worden en in draw.io worden ingeladen))





### Geleerd van business processes

Van dit leerdoel heb ik ten eerste geleerd om na te denken over de flow van een applicatie en deze uit te werken met behulp van de business procesmodel notatie. Ik heb geleerd wat bepaalde icoontjes van deze notatie inhouden en hoe ik dit kan omzetten in een correct model. Aangezien je hier schematisch het proces van de applicatie omschrijft is het ook makkelijk om te kijken of er misschien stukjes zijn die beter/sneller kunnen.

### Bijlage

* BPMN-model (<https://github.com/RenoMuijsenberg/S3-Software/blob/main/BPMN-proftaak.drawio>)

## [8. Professional](#_8._Professional)

### Professioneel binnen groep

Met de proftaak werken we in een groep van zes personen. Binnen deze groep werken we op een zo professionele manier mogelijk met elkaar. We zorgen dat we iedereen in zijn/haar waarde laten en we zo goed mogelijk met elkaar omgaan. Wanneer iemand zijn/haar mening over iets geeft of een discussie hier over wil hebben waarom hij/zij iets wel of juist niet vindt dat we bijvoorbeeld moeten gebruiken dan luisteren we naar de argumenten van dit persoon en besluiten we met z’n alle of dit een goed idee is of juist niet. In het begin van de proftaak hebben we met z’n alle een aantal regels bedacht die we hanteren zoals: “Wanneer er een onenigheid bevindt maken we met z’n alle een besluit hierover en de meeste stemmen gelden”.

### Professioneel met stakeholder(s)

Met de stakeholder proberen wij op een zo professioneel mogelijke manier te communiceren. We zorgen dat we geen grof taal gebruiken wanneer we contact hebben met de stakeholder en proberen zo goed mogelijk aan hun eisen en wensen te voldoen. Verder zorgen we ervoor dat we geen rare kleding aan hebben (al is dit bijna nooit het geval).

### Professioneel van mezelf

Aangezien ik al redelijk veel kennis heb is het natuurlijk ook fijn voor andere als ik zoveel mogelijk van deze kennis probeer te delen waardoor ons project beter/ professioneler kan worden gebruikt. Zo heb ik met nog iemand het besluit gemaakt om bijvoorbeeld onze git branches te beschermen door middel van ‘required review’. Zo voorkomen we schoonheidsfoutjes in ons code door bijvoorbeeld code die niet goed is nagekeken of getest. Verder hebben we naar aanleiding van mij en nog een ander het besluit gemaakt om git flow te gebruiken binnen ons project. Ik heb geholpen om bij iedereen dit te initialiseren zodat iedereen dit makkelijk in gebruikt kon nemen. Wanneer iemand hulp nodig heeft probeer ik zo goed mogelijk te helpen en dan ook uit te leggen wat ik doe zodat de ander dit ook leert. Zo zorg je dat andere uit je groep ook professionele worden of zelfs beter dan dat ze nu al zijn.

### Business man, businessman, executive, man, suit icon - Download on IconfinderGeleerd van professional

Deze proftaak heb ik geleerd om me op een zo professioneel mogelijke manier om te gaan met iedereen betrokken bij het project. Uiteraard moet je professioneel omgaan met je stakeholder/product owners. Maar je moet ook op een professioneel en serieuze manier met je team genoten om kunnen gaan aangezien iedereen op school zit om te leren. Als de mensen met wat meer kennis binnen een groep de mensen met wat minder kennis dingen leren dan zorg je er samen voor dat bepaalde technieken en de tijd die een persoon hieraan besteed minder wordt waardoor je ook een beter product kan neer zetten. Dit laat ook leiderschap zijn als je vrijwillig iemand vraagt of die hulp nodig heeft wanneer je dit ziet.

### Bijlage

* GitFlow-research (<https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular/wiki/Git-Flow-research>)
* CORS-research (<https://github.com/RenoMuijsenberg/S3-Festival-Planner-Angular/wiki/CORS-research>)

## Links tussen leerdoelen

### Web Application & Software quality

Voor deze twee leerdoelen is het vrij vanzelfsprekend. Je schrijft code, je wilt doe code zo goed mogelijk maken/valideren. Dit doe je uiteraard door tests voor te schrijven. Verder kun je ook nog een code analysis tool toevoegen die scant op bijvoorbeeld duplicated code fragments of code smells zoals unused import statements. Verder kun je de code ook nog scannen door het gebruikt van bijvoorbeeld ESlint.

### Web Application & Agile

Agile is uiteraard een van de meest gebruikte methodes voor het process van een applicatie te ontwikkelen. De Agile method zorgt ervoor dat code in kleine beheersbare stukje wordt opgeleverd in plaats van dat alle code in een keer opgeleverd wordt. De methode die ik gebruik voor agile is scrum. De link tussen deze leerdoelen is uiteraard het beheren van het development process zodat dit op een soepele manier verloopt en zodat je oplevering met zo weinig mogelijk hoofdpijn verlopen.

### Web Application & CI/CD

Web application en CI/CD zijn twee leerdoelen die samen in hand lopen. Aangezien de code van de Web application in een versie beheer zoals GitHub staan. Kun je hier workflows van maken wanneer je naar een bepaalde branch een verandering pusht. Dit zorgt ervoor dat de Web Application op een zo efficiënt mogelijke manier wordt getest/gedeployed.

### Web Application & Cultural differences and ethics

“Cultural differences and ethics” zijn ook weer iets minder aan je IP gelinkt dan aan je GP maar toch komt het hier wel in terug. Zo heb ik ervoor gezorgd dat mijn GitHub in het Engels staat zodat het toegankelijk is voor zoveel mogelijk mensen. Aangezien mijn GitHub open public staat is dit uiteraard belangrijk stel er zou iemand aan willen werken.

### Web Application & Requirements and design

De link tussen deze twee leerdoelen in vrijwel logisch, de requirements zijn een indicatie van wat de applicatie moet doen en het design beschrijft hoe dit geïmplementeerd moet gaan worden. Zo heb je de voor de requirements user story’s met acceptatiecriteria en hier hangen weer tickets aan van wat er specifiek in kleine taakjes moet gebeuren. Dan heb je nog de dingen zoals het C4 model en het UI design, dit is dan weer het design aangezien dit specificeert hoe het eruit moet gaan zien/ wat er moet gaan gemaakt worden met welke technieken.

### Web Application & Business process

Deze twee leerdoelen gaan uiteraard hand in hand, een business proces gaat vaak over een proces dat vaak gedaan wordt in de normale wereld zoals het bijhouden van bestellingen van klanten. Door het te maken van een webapplicatie kan dit worden opgelost op een simpele manier die vaak tijd bespaard voor bijvoorbeeld een bedrijf. Hier worden dan modellen van gemaakt zoals bijvoorbeeld een BPMN-model.

### Web Application & Professional

De leerdoelen “Web application” en “professional” heb uiteraard met elkaar te maken. Je zet jezelf natuurlijk zo professioneel mogelijk op wanneer je een applicatie gaat ontwikkelen. Je zorgt dat alles op een zo professioneel mogelijk manier gebeurt zodat wanneer iemand anders bijvoorbeeld wil gaan programmeren aan je applicatie dit ook zonder te veel raar gedoe kan gebeuren. Verder is het stukje professioneel bij de proftaak uiteraard dat je ook op een zo fatsoenlijk mogelijke manier aan de applicatie werkt samen met je team en de stakeholders. Ook is de research die geschreven wordt over een onderwerp wat aanbod kwam bij het maken van de web application een zo professioneel mogelijke manier gebeurt namelijk op de werkwijze van het dot framework.

### Software quality & Agile

Agile scrum is een framework voor het beheren van project. Dit legt de nadruk op samenwerking, flexibiliteit en herhaaldelijk verbeteren. Het doel van scrum is een hoge kwaliteit software te kunnen opleveren in kleine stukjes genaamd sprints. Aangezien je elke sprint een oplevering hebt zo kijk je samen met de product owner wat de kwaliteit van de applicatie is en of er wat aangepast moet worden volgende sprint zodat het een nog beter project wordt.

### Software quality & CI/CD

De twee leerdoelen lopen hand in hand aangezien de softwarekwaliteit en stuk geautomatiseerd kan worden door de CI/CD. Wanneer je de CI/CD het project laat testen op elke push kun je de kwaliteit dus ook goed bewaren. Verder kan je ook nog code analysis tools toevoegen om de code nog beter te maken.

### Software quality & Cultural differences and ethics

Verschillende culturen kunnen verschillende aanpakken hebben met het maken van een project. Dit kan leiden tot conflicten tussen team genoten omdat er een miscommunicatie is tussen deze twee partijen. Dit kan leiden tot een slechter product wat je niet wilt hebben daarom is het belangrijk wanneer je richtlijnen hebt als je met andere culturen werkt om de kwaliteit te bewaren.

### Software quality & Requirements and design

De requirements en design zijn de uitlijning van hoe je applicatie moet werken/ wat er allenmaal in moet zitten. Wanneer er dingen ontbreken uit bijvoorbeeld het design dan is de softwarekwaliteit uiteraard ook iets minder. Verder kun je deze requirements nog bewaken door het schrijven van tests. Hoe beter er is nagedacht over de requirements hoe makkelijker is het om kwalitatieve code hiervoor te schrijven.

### Software quality & Business process

De kwaliteit van de applicatie kan veel impact hebben op de efficiency waardoor dus bijvoorbeeld een applicatie niet goed genoeg kan zijn voor het op te lossen van het probleem. Wanneer je ervoor zorgt dat de kwaliteit van de software goed is dan kan dit ook goed worden gebruikt in bijvoorbeeld een bedrijf of om een probleem op te lossen.

### Software quality & Professional

Professionals die werken binnen de softwarewereld zijn verantwoordelijk voor het opleveren van hoge kwaliteit software. Wanneer iemand zich opstelt en gedraagt als een professional zou de rest van het team dit overnemen, de product owner merkt dan ook aan je ook echt een goed product wil opleveren wanneer je een professionele uitstraling hebt.

### Agile & CI/CD

CI/CD wordt gebruikt om het agile proces te ondersteunen. Agile focust op het herhaaldelijk opleveren van een project in sprints. De CI/CD tool kan dit ondersteunen door bijvoorbeeld een CD op te zetten die het project naar een live server pusht. Zo kan je makkelijk en snel de herhaaldelijke verbeteringen doorvoeren.

### Agile & Cultural differences and ethics

Agile en culturele verschillen zijn twee leerdoelen die samengaan aangezien agile gefocust is het samenwerken en communicatie. Agile werkt op de manier dat er steeds communicatie en feedback is. Dat kan oplossen dat er verschillen komen in de culturen van groepen. Wanneer dit nog steeds het geval is kan dit altijd opgelost worden door de product owner.

### Agile & Requirements and design

Met agile werk je in sprints waar aan het einde van elke sprint een oplevering is aan de product owner. Om de requirements en het design zo goed mogelijk op te volgen kan dit bij elke oplevering besproken worden met de product owner. Het design van de applicatie kan dan ook worden gecontroleerd. Wanneer er een fout is bij de oplevering kan de product owner hier een opmerking over geven wanneer je toch afwijkt van het design kan hier ook altijd over gesproken worden zolang je maar een goed argument hebt waarom je bent afgeweken van het design.

### Agile & Professional

Agile en professional zijn dicht aan elkaar gelinkt aangezien de agile methode vaak in een team wordt uitgevoerd. Wanneer je professioneel met je team omgaat kun je er ook voor zorgen dat het Agile process zo professioneel mogelijk gebeurt. Door agile wordt het project in kleinere beheersbare stukje opgedeeld die op een effectieve en professionele manier kunnen worden opgepakt.

### CI/CD & Cultural differences and ethics

Deze leerdoelen zijn niet direct met elkaar verbonden op het feit na dat ze beide behoren bij het maken van een project. Ook al hebben ze samen geen direct connectie kunnen ze wel een rol spelen bij het maken van een succesvol project.

### CI/CD & Requirements and design

De CI/CD tool zorgt ervoor dat de code herhaaldelijk getest wordt. Wanneer je tests hebt om te valideren dat de requirements in een project zitten dan worden elke keer wanneer de CI runt de requirements gevalideerd. Wanneer een test faalt kun je snel en makkelijk zien waar dit probleem zich bevindt zodat dit zo spoedig mogelijk kan worden opgelost zonder te veel zoek werk.

### CI/CD & Professional

De verbinding tussen CI/CD en professional is dat de CI/CD gebruikt kan worden om het project te testen/builden en deployen dit zorgt ervoor dat het project op een professionele /effectieve en efficiënte manier kan worden beheerd. Wanneer je ziet dat er bijvoorbeeld iets niet goed ging in de CI/CD kun je hier een issue van maken op het project bord zodat dit kan worden opgepakt.

### Cultural differences and ethics & Requirements and design

Deze leerdoelen zijn in de hand lopend aangezien culturele verschillen invloed kunnen hebben op de requirements en design. De mensen betrokken bij het project kunnen andere culture hebben dan jouwzelf wanneer dit het geval is kan het zijn dat hun standaards afwijken met die van jou. Dit zou in gedachte moeten genomen worden wanneer je met andere mensen gaat werken.

### Cultural differences and ethics & Professional

Deze twee leerdoelen gaan samen wanneer je in een team werkt met verschillende culturen. Je gaat dan op een professionele manier om met de andere personen van andere culturen. Je zorgt dat je ze in hun waarden laat. Wanneer er een overeenkomst is door de twee culturen los je dit op een professionele manier op.

### Requirements and design & Professional

Een professional zorgt dat het product wat wordt opgeleverd zo goed mogelijk aan de eisen en wensen voldoet. Om te zorgen dat je op een professionele manier dit behandeld zorg je dat je jezelf houdt aan het design van de applicatie en zorgt dat alle requirements zo goed mogelijk afgehandeld zijn.