Reno Muijsenberg

r.muijsenberg@student.fontys.nl

Abstract

S3 software portfolio

Festival planner



Contents

[Leeruitkomsten uitgelegd 2](#_Toc119924109)

[Aantonen leerdoelen 3](#_Toc119924110)

[1. Web application 3](#_Toc119924111)

[2. Software quality 4](#_Toc119924112)

[3. Agile method 5](#_Toc119924113)

[Sprint planning 5](#_Toc119924114)

[Daily stand-up/ stand-down 5](#_Toc119924115)

[Sprint review 5](#_Toc119924116)

[Sprint retrospective 5](#_Toc119924117)

[4. CI/CD 6](#_Toc119924118)

[Builden en testen 6](#_Toc119924119)

[Quality check 6](#_Toc119924120)

[Docker hub 6](#_Toc119924121)

[Leesbaarheid 6](#_Toc119924122)

[Azure 6](#_Toc119924123)

[5. Cultuur 8](#_Toc119924124)

[Wat is cultuur? 8](#_Toc119924125)

[Wat zijn wel bekende dimensies van culturelere verschillen? 8](#_Toc119924126)

[Voorbeelden over culturele verschillen? 8](#_Toc119924127)

[Wat is mijn cultuur? 8](#_Toc119924128)

[6. Requirements and design 10](#_Toc119924129)

[Stakeholder analysis 10](#_Toc119924130)

[Requirements analysis 10](#_Toc119924131)

[Softwaredesign and architecture 10](#_Toc119924132)

[C4 model proftaak: 11](#_Toc119924133)

[UI design 12](#_Toc119924134)

[7. Business processes 13](#_Toc119924135)

[8. Professional 15](#_Toc119924136)

[Professioneel binnen groep 15](#_Toc119924137)

[Professioneel met stakeholder(s) 15](#_Toc119924138)

[Professioneel van mezelf 15](#_Toc119924139)

# 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 methodology:** Carry out, 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 behaviour 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 validate 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 methodology and following team agreements. You can track your work progress and communicate your progress with the team.

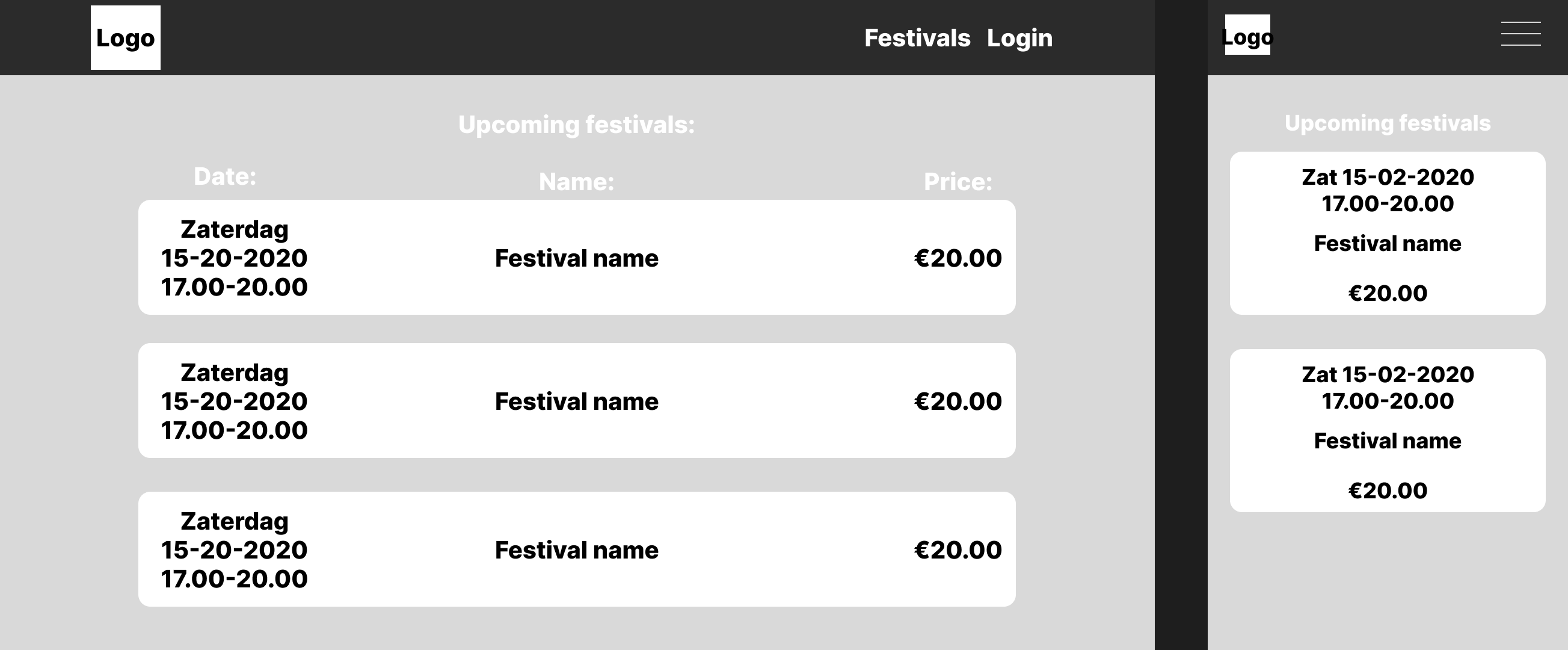
You actively ask and apply feedback from stakeholders and advise them on the most optimal technical and design (architectural) solutions.  
You choose and substantiate solutions for a given problem.

# 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.

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 front page

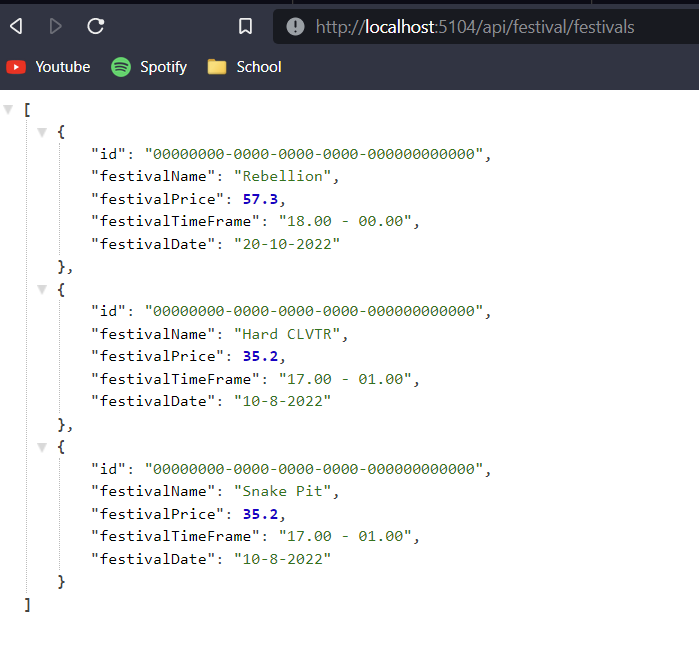
Vervolgens ben ik begonnen met het opzetten van het front-end project en ben ik een [video tutorial](https://www.youtube.com/watch?v=k5E2AVpwsko&t=3408s) gaan kijken over hoe Angular werkt. En heb met de kennis van deze tutorial de pagina van de wireframe nagemaakt in HTML, CSS (Screenshot bevat hardcoded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

Graphical user interface, website

Description automatically generated

Front page in HTML, CSS

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 middle 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

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

Ik heb om ervoor te zorgen dat er zo weinig mogelijk bugs in zitten een static code analysis toegevoegd in mijn CI/CD hiervoor heb ik SonarCloud gebruikt.

A picture containing graphical user interface

Description automatically generated

SonarCloud scanner

## [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 retrospective

Aan het einden van onze sprint review dan houden we met ons groepje een retrospective, in deze retrospective 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 “Retrospective 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”.

## [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 SonarCloud (een static 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.

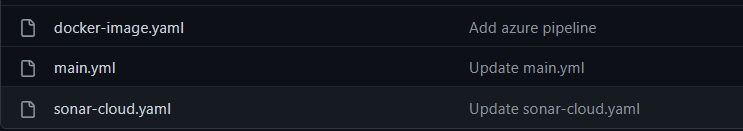
CI/CD

### 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.

## [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 sociaale 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.

## [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 that is nice in use for their customers for ordering food.
  + Wants an application that is nice in use for their employees for managing food.
* Product-owner: Mediaan
  + Wants to deliver a good working and good-looking application to the customer.
  + Communicates needs from customer to project team members.
* Project team members: MDMA (me)
  + Wants to develop a good working and good-looking 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
  + Want an application that is nice in use and looking nice for ordering food.
* Users: Employees of the restaurant
  + Want an application that is easy to use for managing customer orders keeping track of the status.

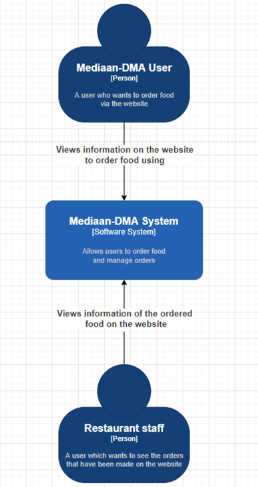
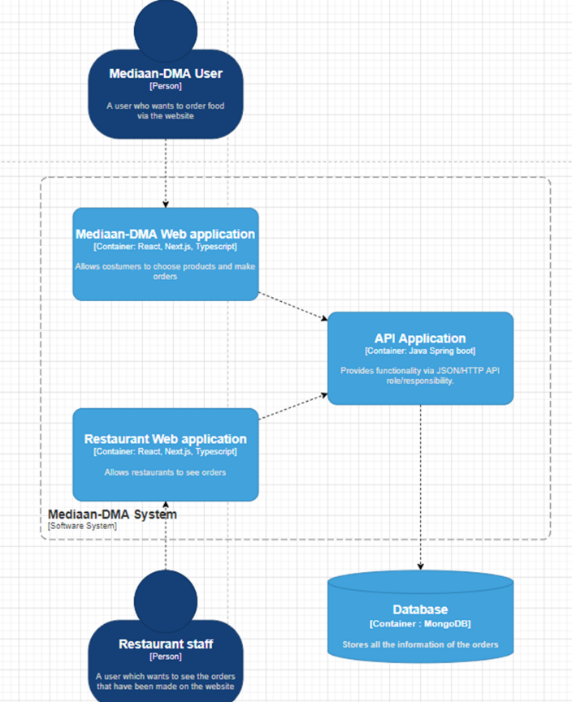
### Requirements analysis

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.

### Softwaredesign and architecture

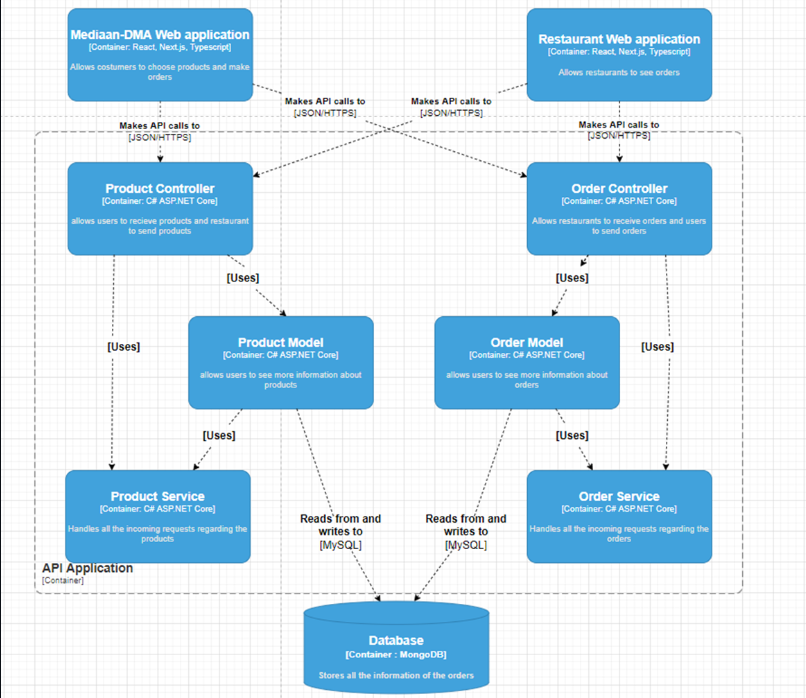
Wanneer we een keuze gaan maken over de architectuur zoals bijvoorbeeld een taal die we gebruiken of een database die we gebruiken hebben we met elkaar besproken, wanneer iedereen het eens is met deze keuze gebruiken we hem pas. Als iemand het niet eens is met de keuze die we gemaakt hebben dan wordt 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.

### C4 model proftaak:



C2

C1



C3

### 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.

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

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

## [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 protecten 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.