HANZEHOGESCHOOL

Informatica

Afstudeer scriptie

25 augustus 2014, Hoogeveen

Auteur:
Marcel Horlings
351254
Opleiding: Informatica

Stagedocent:
Jacob Mulder
Hanzehogeschool Groningen

Stagebedrijf:
Nidaros
Pascal Hakkers
Stoekeplein 1a
7902 HM, Hoogeveen

Nidaros

Hanzehogeschool Groningen
University of Applied Sciences

Woord vooraf

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nidaros

Hanzehogeschool Groningen
University of Applied Sciences

Samenvatting

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nidaros

Hanzehogeschool Groningen
University of Applied Sciences

Inhoud

1 Inleiding

Inleiding

1.1 Nidaros

Nidaros is een bedrijf dat adviseert en ondersteuning biedt bij IT-gerelateerde bedrijfsprocessen. Het doel van Nidaros is het stimuleren van bedrijfsprocessen, het informeren van procesverantwoordelijkheden en het bewust worden van wat IT kan toevoegen aan bedrijven en organisaties. Nidaros is een klein bedrijf met ongeveer twaalf werknemers, die op veel markten in de IT bezig is. Het biedt advies op het gebied van softwareontwikkeling en geven advies op basis van testmanagement en op basis van een informatieanalyse. Nidaros maakt zelf websites en software voor klanten en voeren optimalisaties door in bestaande websites. Daarnaast doet ze ook een stuk ICT-beheer binnen bedrijven, en voeren ze reparaties van computers en iPhones uit.

Consultancy

In de Consultancy tak van Nidaros worden werknemers gedetacheerd naar andere bedrijven om daar te helpen met het inbrengen van externe systemen, en voor het testen van systemen.

Solutions

Bij Solutions worden producten verkocht aan gebruikers. Deze producten kunnen ingekocht zijn, zelf gemaakt zijn of een combinatie hiervan. Daarnaast levert Solutions ook diensten op het gebied van ICT.

Organisatiestructuur

Nidaros is een klein bedrijf met 12 werknemers in dienst.

De primaire processen

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a,

molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

1.2 Probleemstelling

Bij elke aankoop die gedaan wordt in een winkel wordt er een bonnetje meegegeven. Met dit bonnetje kan de garantie van een product verhaald worden wanneer het product het begeeft. Voor veel mensen is het bijhouden van alle bonnetjes die bij elk apparaat verkregen wordt en lastige taak. Ze vergeten wanneer hoe lang het bonnetje nog geldig is of wanneer de garantie afloopt en veel bonnetjes raken ook kwijt. Voor de mensen die hier last van hebben wordt de ScanjeBon app ontwikkeld. Met deze app kunnen de bonnetjes gemakkelijk via de smartphone opgeslagen worden. Waardoor gebruikers de bonnetjes altijd op één centrale plek hebben. Gebruikers van de app zullen ook genotificeerd worden wanneer bonnen aflopen en ze kunnen de bonnetjes overzichtelijk uit elkaar houden.

1.3 Inhoud van dit rapport

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2 Plan van aanpak

De looptijd van het project bedraagt twintig weken. Deze twintig weken zal verdeeld worden over vier fases zoals beschreven in RUP. De eerste fase is Inception, in deze fase wordt er voor gezorgd dat iedereen betreffende het project hetzelfde beeld krijgt over het project. De tweede fase, Elaboration genaamd, wordt gebruikt als de eerste iteratie.

Zo wordt er een werkend product opgeleverd, maar worden ook dingen gedaan als het opzetten van de ontwikkelomgeving en het versiebeheersysteem. De derde fase, Construction, is waar het ontwikkelen gebeurd. Hier zal het overgrote deel van de Scanjebon app gemaakt worden. Dit wordt gedaan in iteraties en na elke iteratie wordt de app getoond aan de deelnemende stakeholders.

De laatste fase is de Transition, de Scanjebon app is hier klaar voor gebruik en kan in de markt gezet worden. Daarnaast zal hier de overdracht plaats vinden van de broncode, de documentatie en de ontwikkelomgeving. De fases zullen er samen voor zorgen dat er een Scanjebon app voor minimaal één platform gemaakt zal worden met daarnaast een ontwikkelomgeving waar een volgende ontwikkelaar mee verder kan. De planning staat als gantt chart in B.

3 Versiebeheersysteem

Tijdens het bouwen van de applicatie is er gebruikt van het versiebheersysteem Git. Voor de stageperiode begon werd er binnen Nidaros geen gebruik gemaakt van een dergelijk systeem, daarom moest tijdens de stage uitgezocht worden wat het beste zou werken voor de stage en voor de rest van het bedrijf.

3.1 CVCS of DVCS?

Voor de keuze van van versiebeheersystemen zijn er twee opties een Centralized Version Control System (CVCS) of een Distributed Version Control System (DVCS). Voor CVCS is de bekenste optie Subversion (SVN) en voor de DVCS zijn Git en Mercurial de grootste opties. Aan beide kanten worden er versies bijgehouden van de code die gescrheven wordt en het is bij allebei mogelijk om de code te delen tussen verschillende manieren, ... Maar er is wel een verschil hoe dat gebeurd en dat verschil is ook merkbaar in het gebruik en heeft zijn voor en nadelen.

Servers

Om de code te kunnen delen bij een CVCS is er een server nodig. Op deze server worden alle versies bijgehouden, wat erg handig is want zo kan iedereen bijhouden wat er met de code gebeurd en weet iedereen waar een ander mee bezig is. Het grote probleem met een CVCS server is dat als de server stuk gaat en de data op de server verloren gaat is alle code en vooral de geschiedenis van de code ook weg. Wat dan overblijft is alle code die de mensen op dat moment lokaal op de werkplek hebben staan en geen hele geschiedenis van alle code meer. Bij het werken met CVCS-en wordt er altijd maar een deel van de code naar de locale machine gekopieerd. Wanneer de server voor een tijd offline gaat kan niemand hun code meer delen, opslaan of verder gaan met een ander stuk. Als er een aftakking van de code gemaakt wordt zodat daar nieuwe functies aan toegevoegd kunnen worden heeft de gebruiker alleen toegang tot die code. De code van de nieuwe functies die een collega op dat moment maakt blijven alleen op de server staan.

Bij DVCS is een zelfde opzet mogelijk, er kan een server neergezet worden waar alle code beschikbaar op is en waar wijzegingen bekend blijven. Net als bij de CVCS kan hier de code ingezien worden door iedereen en kan iedereen zien waar een ander mee bezig is. Mocht er bij een DVCS de server stuk gaan of offline, dan onstaat hier wel een verschil met de CVCS. Bij een DVCS staat staat op elke werkplek alle code en wijzegingen. Wijzegingen kunnen opgeslagen worden zonder verbinding met het internet, dit betekent niet alleen dat er door gewerkt kan worden tijdens het offline gaan van de server maar ook wanneer er gewerkt wordt vanuit een trein zonder internet verbinding. Het stukgaan van de server betekent bij DVCS niet dat er geen code meer gedeeld kan worden onderling, want iedereen kan als server fungeren en van iedereen kan de code opgehaald worden. Elke werkplek is hierdoor een backup van de server en van elkaar. Dit kan ook gedaan worden omdat bij een DVCS alle code opgehaald wordt van de server en niet alleen het stukje waar de ontwikkelaar op dat moment mee bezig is. Dit betekent dat als de ontwikkelaar een stuk bij wil dragen aan een stuk code waar hij niet bij betrokken was hij dit in een handomdraai kan doen

VCS -> Git

Een groot verschil tussen de eerder versiebeheersystemen en Git is hoe ze code en vooral veranderingen daarin opslaan. Bij de eerder VCS-en wordt elke keer dat er een stuk code toegevoegd wordt wordt er een nieuwe versie van dat bestand opgeslagen en de wijzeging er naast. Bij het delen van de code worden alleen wijzegingen en bestanden opgeslagen van de bestanden die dan aangepast zijn. Bij Git gaat het opslaan anders, Git kijkt bij elke toevoeging van code naar alle bestanden en onthoud van alle bestanden wat de status op dat moment was. Dit doet Git niet door de bestanden opnieuw op te slaan maar door te verwijzen naar de laatste aanpassing van dat bestand.

Branches

Elke wijzeging die toegevoegd wordt bij een VCS is een commit. Alle commits zijn stukjes code die op elkaar volgen, bij meerdere commits komt er als het ware een lijn van commits. Branches zijn aftakkingen van de reguliere lijn (master branch) aan commits. Een branch wordt gemaakt

met het geven van een naam, deze naam is een beschrijving van wat de wijzegingen gaan doen, een bug-fix of de naam van de functie die in deze branch wordt toegevoegd. De commits in deze branch zijn ook alleen hier zichtbaar en te gebruiken totdat deze branch samengevoegd wordt met de reguliere lijn aan commits, mergen genoemd.

Het maken van branches is een goede manier om mee te werken, omdat de master branch blijft werken en pas wanneer een branch voltooid is die wijzigingen erbij komen. Zo raakt de master branch niet vervuild met versies die maar half werken of nog getest moeten worden maar gebeurd dat al in de branch. Met branches kan er ook tegelijk aan een nieuwe functie van de applicatie gewerkt worden en aan een bug in de code. Wanneer er een bug gemeld wordt in de applicatie kan de ontwikkelaar naar de master branch gaan en vanaf daar een nieuwe branch aanmaken om de bug te maken. Wanneer deze bug gemaakt is kan het getest worden en daarna op de master branch gezet zodat het direct opgeleverd kan worden.

Tussen Git en oudere versiebeheersystemen is er een groot verschil in het maken en gebruiken van branches. In oudere versiebeheersystemen kost het veel tijd en moeite om een branch aan te maken, aangezien het hier vaak betekend dat alle code in de repsoitry verplaatst moet worden naar een nieuwe map. Dit kost tijd wat afhankelijk is van de grote van een repository.

In git is het maken van een branch niet meer dan het wegschrijven van 41 karakters. Doordat het maken van een branch in Git niet meer is dan een verwijzing naar een vorige commit. Dit wordt gedaan in een SHA-1 checksum van 40 karakters die bij de laatste commit hoorden en een enter. Hier hoeven dus verder geen bestanden voor gekopieerd of verplaatst te worden en hoeft er bij Git niet gewacht te worden om een branch aan te maken.

Mergen

3.2 Tools

Stash - github - bitbucket - gitlab - github enteprise

4 Ontwerp keuzes

4.1 Besturingsysteem

Voordat er begonnen kon worden aan het project moest eerst duidelijk zijn op welk platform de applicatie zou draaien. Door de grote overmacht van Android en iOS op het gebied van besturingssystemen voor de smartphone is er in het onderzoek voornamelijk hierop de nadruk gelegd. Hieronder zal uitgelegd worden waarom er voor een iOS applicatie is gekozen en wat de voor en nadelen van beide besturingssystemen zijn

Keuze

4.2 Communicatie

4.3 Frameworks

5 Architectuur

5.1 Architecturele eisen

Dit onderdeel beschrijft de eisen die aan de software wordt gesteld bij het maken van belang zijn binnen de softwarearchitectuur.

Niet-functionele eisen

Deze niet-functionele requirements zijn van belang voor de architectuur van het te bouwen software.

Naam	Architecturele relevatie
Doorlooptijd	De applicatie wordt ontwikkeld tijdens de stage periode van de stagair, wat 20 weken bedraagt.
Beveiliging van verbinding	Voor het versturen van gegevens en het registreren en inloggen, is het van belang dat de verbinding waarover dit verstuurd wordt beveiligd is.

Use Case View (functionele requirements)

Voor de architectuur zijn de volgende functionele requirements van belang.

Naam	Architecturele relevatie	Geaddres
Foto maken of laden	De applicatie moet een ruimte hebben waar het foto's kan opslaan. tevens moet het de mogelijkheid hebben om de camera en de eerder gemaakte foto's te benaderen	UC1: Foto
Gegevens opslaan op de server	Er moet een database draaien op de server, waar de API toegang toe heeft.	UC2: Geg
Gegevens versturen naar de server	Er moet een beveiligde verbinding zijn van de applicatie naar de server.	UC5: Vers
Opslaan van foto's op de server	Er moet ruimte zijn voor foto's van alle gebruikers en alle bonnen die zij aanmaken.	UC5: Vers

UC1: Foto maken of inladen

Naam	UC1: Foto maken of inladen
Samenvatting	De gebruiker is in het bezit van een nog niet opgeslagen bon en maakt hier een foto van of heeft deze foto al op zijn telefoon staan en haalt hem daar op.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft een nieuwe bon die hij in wil voeren.
Resultaat	De gebruiker ziet het gemaakte of gekozen bonnetje staan onder de velden van de gegevens die hij kan invoeren of ingevoerd heeft.
Beschrijving	De gebruiker navigeert naar het scherm om een nieuwe bon te maken, heeft hij twee opties: Maak foto en Kies bestaande foto. Bij het klikken op de eerste knop krijg de gebruiker de standaard iOS foto scherm voor zich en kan hij een foto maken. Door te klikken op de tweede optie krijg de gebruiker de mappen voorzich die op zijn telefoon staan en kan hier een foto uit kiezen. Voor beide opties geld dat de gebruiker daarna terugkomt de gebruiker terug in het scherm om een nieuwe bon te maken en ziet daar zijn foto staan.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt.
MoSCoW	M

UC2: Gegevens invoeren en opslaan

Naam	UC2: Gegevens invoeren en opslaan
Samenvatting	De gebruiker kan de gegevens van de bon als datum, garantitijd, naam en gekocht bij invoeren en vervolgens opslaan zodat het bewaard blijft.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft een nieuwe bon die hij in wil voeren.
Resultaat	De gebruiker ziet het gemaakte of gekozen bonnetje staan onder de velden van de gegevens die hij kan invoeren of ingevoerd heeft.
Beschrijving	De gebruiker navigeert naar het scherm om een nieuwe bon aan te maken en kan hier de gegevens datum, garantietijd, naam en gekocht bij invoeren. Op deze zelfde pagina kan de use case UC1: Foto maken of inladen ingevoerd worden. Wanneer alles is ingevoerd kan de gebruiker op de knop opslaan klikken en worden alle gegevens opgeslagen in de database van de applicatie en de foto bewaard in een map die gespecificeerd is voor de applicatie.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt.
MoSCoW	M

UC3: Overzicht van alle bonnen weergeven

Naam	UC3: Overzicht van alle bonnen weergeven
Samenvatting	Alle bonnen worden in een lijst weergegeven aan de hand van een categorie, zodat snel te zien is welke bonnen er in die categorie al zijn.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft meerder bonnen opgeslagen.
Resultaat	Alle bonnen worden in een lijst getoond.
Beschrijving	De gebruiker is ingelogd en selecteert een categorie in die categorie ziet de gebruiker vervolgens een lijst met alle bonnetjes die bij die categorie horen.
Alternatief	Als er geen bonnetjes zijn ziet hij een lege lijst.
MoSCoW	M

UC4: Invoeren van nieuwe categorieën

Naam	UC4: Invoeren van nieuwe categorieën
Samenvatting	De gebruiker kan een nieuwe categorie aanmaken waarmee hij de bonnen overzichtelijk van elkaar kan scheiden.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd op de applicatie.
Resultaat	Er komt een nieuwe categorie bij in de lijst met categorieën en er is de mogelijkheid bijgekomen om daar bonnen aan toe te voegen.
Beschrijving	De gebruiker is op het scherm met alle categorieën en klikt op de knop om een nieuwe categorie aan te maken. Vervolgens komt er een invoerscherm waar de gebruiker de naam van de categorie in kan zetten. Zodra de gebruiker op opslaan klikt wordt de categorie opgeslagen en ziet de gebruiker deze categorie in de lijst met categorieën.
Alternatief	Als er iets bij het aanmaken verkeerd gaat wordt daar een melding over gegeven.
MoSCoW	M

UC4: Overzicht van alle categorieën weergeven

Naam	UC4: Overzicht van alle categorieën weergeven
Samenvatting	De gebruiker ziet een alle categorieën in een lijst
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd en heeft één of meer categorieën ingevoerd.
Resultaat	De gebruiker ziet alle categorieën die hij eens heeft aangemaakt.
Beschrijving	Na het inloggen is dit het eerste scherm wat de gebruiker ziet met alle categorieën.
Alternatief	Als er geen categorieën zijn wordt er een lege lijst getoond.
MoSCoW	M

UC5: Versturen van bonnen naar de server

Naam	UC5: Versturen van bonnen naar de server
Samenvatting	Na het opslaan van een bon op de mobiel wordt de bon automatisch verstuurd met een POST naar de server.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft een nieuwe bon opgeslagen.
Resultaat	De bon is zowel op de telefoon als op de server opgeslagen worden en kan benaderd worden via de applicatie en website.
Beschrijving	Wanneer de gebruiker de bon opslaat in UC2: Gegevens invoeren en opslaan worden de gegevens van de bon automatisch opgeslagen met een POST. Wanneer dat gelukt is wordt de foto van UC1: Foto maken of inladen verstuurd naar de server.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt.
MoSCoW	M

UC6: Bonnen verwijderen

Naam	UC6: Bonnen verwijderen
Samenvatting	De bon wordt verwijderd van de applicatie en server en is hierna niet weer terug te zien een lijst met bonnen.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft eerder al een bon ingevoerd.
Resultaat	Na het verwijderen is de bon zowel van de applicatie als van de server verdwenen.
Beschrijving	De gebruiker gaat naar een categorie en ziet hier een lijst met bonnen. Hij kiest een bon uit die hij wil verwijderen en houd deze bon een seconde ingedrukt. Hierna verschijnt een popup die vraagt of de gebruiker het echt wil verwijderen. Zodra de gebruiker op ja drukt is de bon verwijderd.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt. Drukt de gebruiker op nee gebeurd er verder niks en blijft de gebruiker alle bonnen in de categorie zien.
MoSCoW	S

UC7: Bonnen aanpassen

Naam	UC7: Bonnen aanpassen
Samenvatting	De gebruiker kan de gegevens van de bon aanpassen en een nieuwe foto selecteren die vervolgens worden op geslagen bij de bon.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft eerder al een bon aangemaakt.
Resultaat	De gegevens van de bon zijn aangepast, zowel op in de applicatie als op de server.
Beschrijving	De gebruiker gaat naar de bon toe die hij wil aanpassen. Daar klikt de gebruiker op de + in het rechterbovenhoek. Hier heeft de gebruiker een dezelfde mogelijkheden als in UC2: Gegevens invoeren en opslaan en UC1: Foto maken of inladen. Hierna kan op opslaan geklikt worden en zijn de gegevens van dezelfde bon zowel op de applicatie als op de server gewijzigd.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt.
MoSCoW	S

UC8: OCR op bonnen uitvoeren

Naam	UC8: OCR op bonnen uitvoeren
Samenvatting	Na het invoeren van een bon wordt door middel van Optical character recognition (OCR) de gegevens van de bon uitgelezen en voorgedefinieerd.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft een nieuwe foto ingevoerd.
Resultaat	Er wordt automatisch een aantal velden ingevoerd op basis van de ingescande bon.
Beschrijving	De gebruiker navigeert naar het scherm om een nieuwe bon aan te maken of naar de pagina om een aan te passen. Vervolgens laad de gebruiker een nieuwe foto in door middel van UC1: Foto maken of inladen. Aan de hand van deze foto worden de gegevens ingevuld of aangevuld. Als de gebruiker het hier mee eens is klikt hij op opslaan en is de bon opgeslagen.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt.
MoSCoW	W

UC9: Account aanmaken

Naam	UC9: Account aanmaken
Samenvatting	De gebruiker kan zich registeren in de app, waardoor hij een account heeft op zowel de applicatie als de server. Hierdoor weet de applicatie welke bonnen bij de gebruiker horen.
Actoren	Gebruiker
Aannames	De gebruiker heeft nog geen account en het e-mail adres is nog niet gebruikt
Resultaat	Er is een nieuw account aangemaakt waarmee ingelogd kan worden.
Beschrijving	De gebruiker is nog niet ingelogd en navigeert naar registreren. Hier vult de gebruiker zijn email adres en wachtwoord in, waarna hij geregistreerd is en direct ingelogd is.
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt. De gebruiker is dan niet ingelogd
MoSCoW	S

UC10: Inloggen

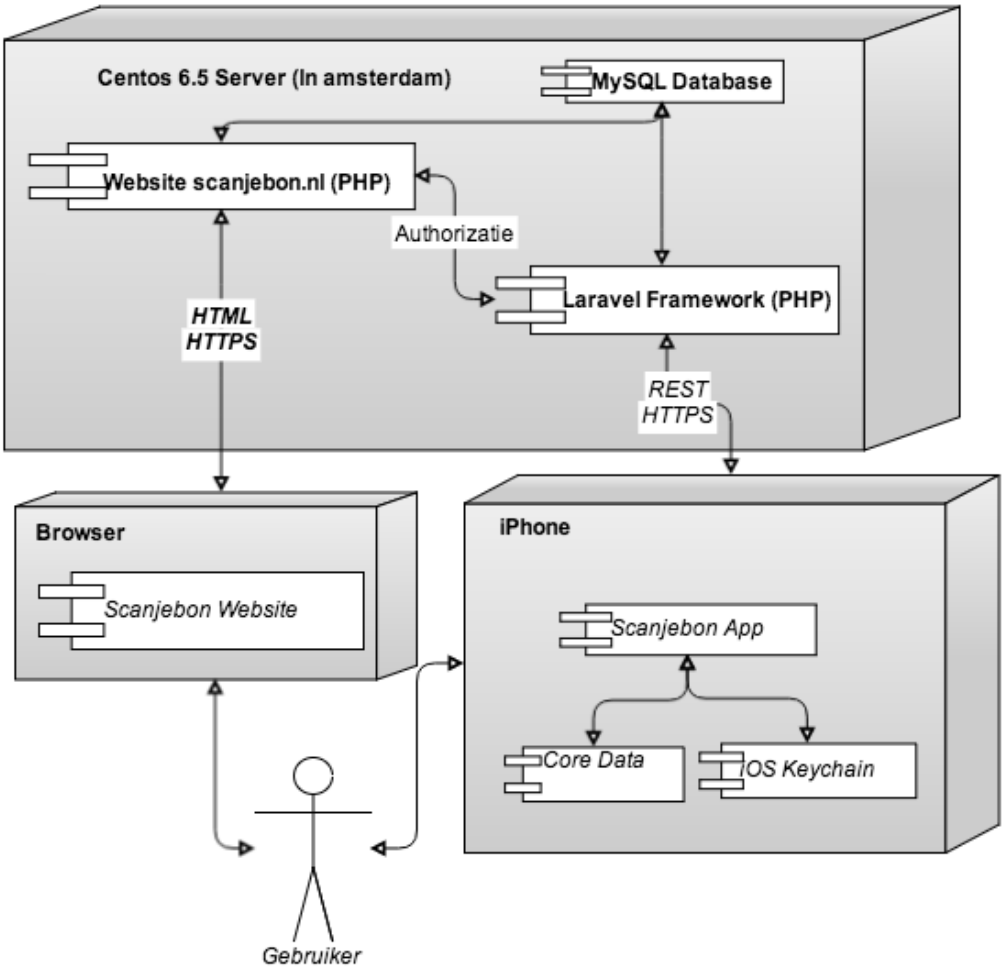
Naam	UC10: Inloggen
Samenvatting	De gebruiker kan inloggen met zijn account waardoor de applicatie weet welke gebruiker het is en welke gegevens de gebruiker mag zien
Actoren	Gebruiker
Aannames	De gebruiker heeft een account aangemaakt en is niet ingelogged
Resultaat	De gebruiker is ingelogd en ziet alleen zijn eigen gegevens
Beschrijving	De gebruiker start de app op en ziet het scherm inloggen. Hier vult de gebruiker zijn eerder gekozen emailadres en wachtwoord. Daarna klikt de gebruiker op de knop inloggen en ziet zijn categorieën
Alternatief	Als er bij het maken iets verkeerd gaat, wordt hier een melding over gemaakt en wordt de gebruiker niet ingelogged.
MoSCoW	S

UC11: Pushnotificaties weergeven

Naam	UC11: Pushnotificaties weergeven
Samenvatting	Bij het bereiken van een einttijd van een bon krijgt de gebruiker hier een pusnotificatie voor.
Actoren	Gebruiker
Aannames	De gebruiker heeft een account aangemaakt, is een keer ingelogd op de applicatie en heeft een bon aangemaakt met een garantie tot tijd.
Resultaat	Er wordt een push notificatie getoond aan de gebruiker die weergeeft welke bon afloopt.
Beschrijving	Aan de server kant wordt er gekeken naar alle bonnen met een garantie tot tijd. Wanneer een van die bonnen binnen de gekozen tijd afloopt worden de gegevens van die bon vertstuurd naar de mobiel van de gebruiker en krijgt de gebruiker een pushnotificatie waar de gebruiker op kan klikken om de bon en alle gegevens daar van kan zien.
MoSCoW	C

UC12: Syncrhonisatie

Naam	UC12: Synchronisatie
Samenvatting	Bonnen kunnen geschynchroniseerd worden met een druk op de knop. Er staan dan dezelfde bonnen op de server (website) als op de app.
Actoren	Gebruiker
Aannames	De gebruiker is ingelogd in de applicatie en heeft een bon aangemaakt via de website.
Resultaat	Na het synchroniseren staan dezelfde bonnen op de server als op de applicatie.
Beschrijving	Bij het klikken op het synchroniseren knop worden alle bonnen die op de applicatie staan .
MoSCoW	S

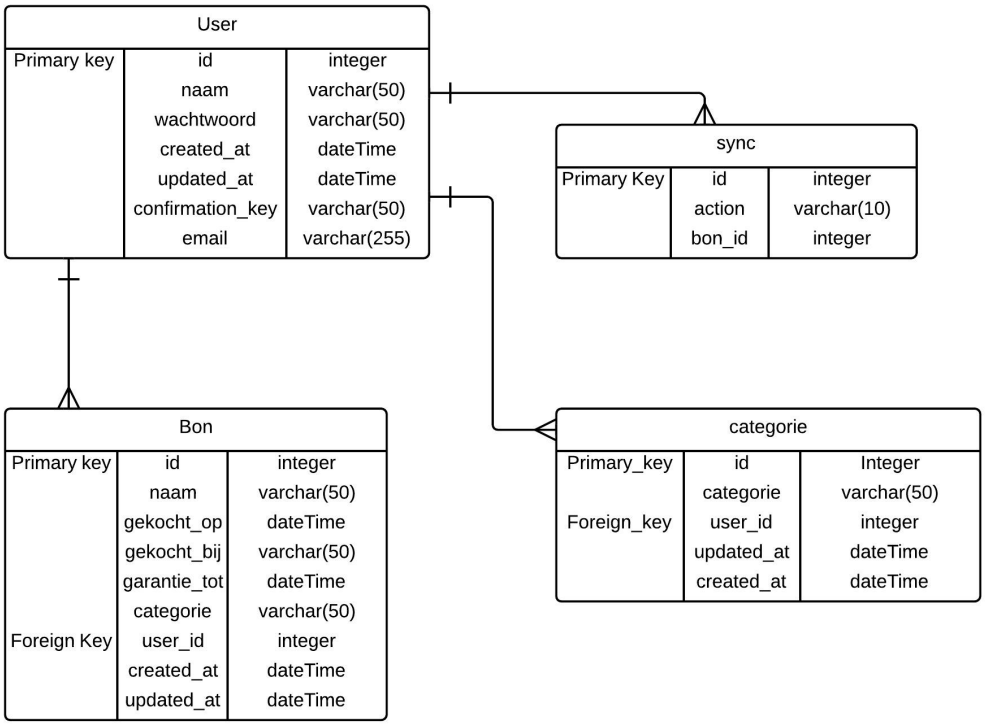


Figuur 5.1: Deployment view van ScanjeBon

5.2 Deployment View

5.3 Infrastructuur

Database



Figuur 5.2: De entity-relationship diagram voor ScanjeBon

5.4 Logical View

In dit onderdeel worden de verschillende lagen van het systeem beschreven en daarmee de logische opbouw van het systeem. Het beschrijven van deze lagen gaat volgens het 4-lagen systeem. Daarnaast zullen een aantal Use Cases uit 5.1 worden beschreven in de vorm van Sequence Diagrams.

Lagen

De ScanjeBon applicatie heeft de volgende 4-lagen:

- Presentatie
- Service
- Domein
- Data

Presentatie

De applicatie heeft twee onderdelen in de presentatie laag. De website en de iPhone applicatie gebruiken dezelfde database en presenteren dezelfde data.

Service

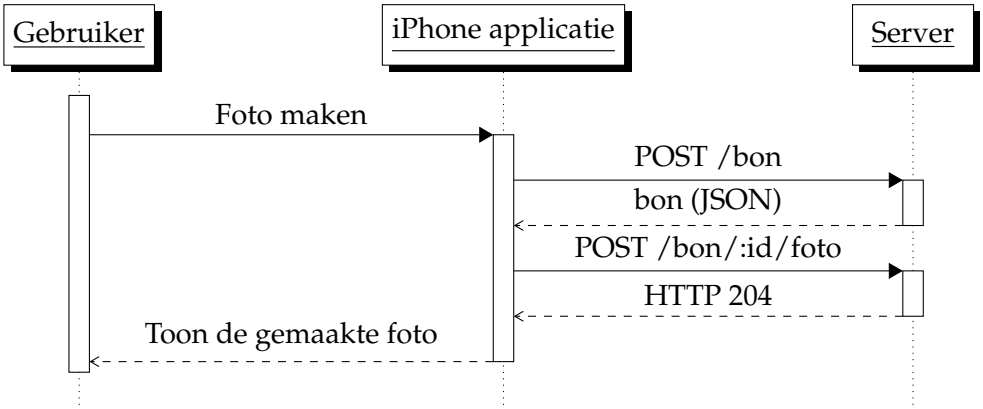
De Service laag zorgt ervoor dat de prestatielagen de gegevens krijgen. Deze laag maakt gebruik van een JSON-API voor de iPhone applicatie. Met de JSON-API wordt gecommuniceerd door middel van HTTP request met de server.

Domein

De Business Logic is de verantwoordelijkheid van de de domeinlaag. In deze laag zit de represenatie van de gegevens, het uitvoeren van veranderingen op de data, het synchroniseren tussen de server en de iPhone applicatie.

Data

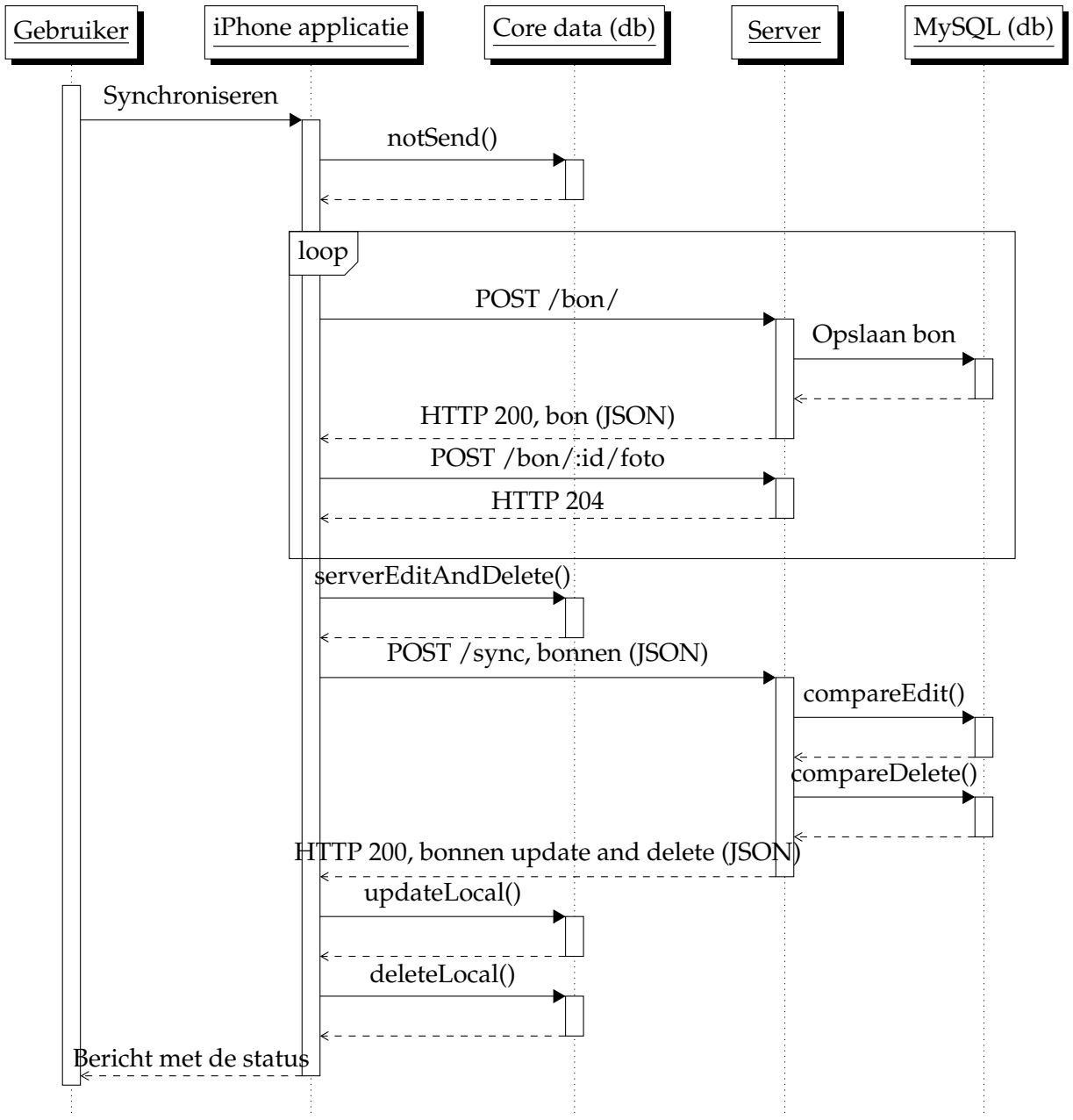
In de Data laag wordt de het opslaan en die integriteit van de gegevens gewaarborgd. De gegevens worden opgevraagd en gebruikt door het Domein. Voor de opslag van de gegevens wordt een database gebruikt en de foto's worden als bestand opgeslagen in een mappenstructuur.



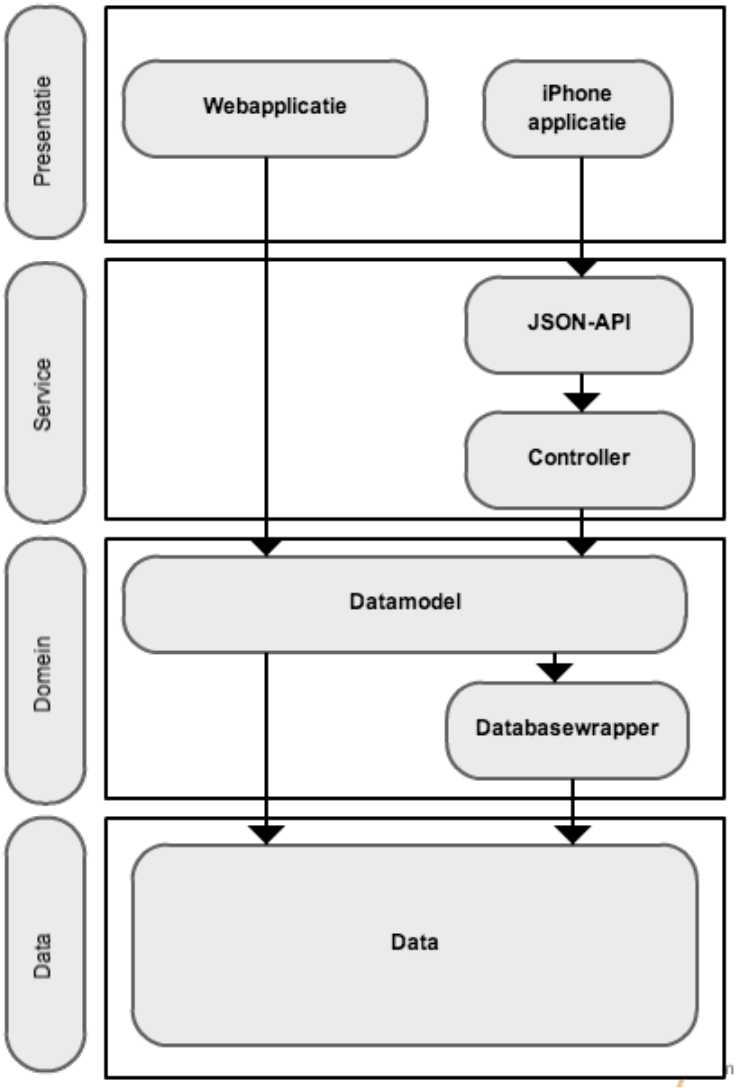
Figuur 5.3: De gebuiker maakt een nieuwe bon aan

5.5 Deelsystemen

De applicatie kent de webapplicatie en de iPhone applicatie. In Figuur 5.5 staat de samenhang tussen de deelsystemen en het 4-lagenmodel.



Figuur 5.4: De gebruiker synchroniseert de bonnen



Figuur 5.5: Deelsystemen in het 4-lagenmodel

6 RESTFul

7 Implementatie

8 Conclusie

A Persoonlijke ontwikkeling

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

B Planning