Afstudeerverslag

Damian Jansen

Versie	3.0
Plaats	Nijverdal
Gepubliceerd	26-10-2020
Onderwijsinstelling	Saxion Enschede
Stagebegeleider	Dhr. R. Hommels
Bedrijfsbegeleider	Dhr. D. Gortemaker
Studentnummer	433928

Informatiepagina

De afstudeeropdracht is gemaakt bij Dynfos Business Solutions B.V. Tijdens het project was Dennis Gortemaker mijn bedrijfsbegeleider. De begeleider vanuit het Saxion was Rogier Hommels.

Tabel 1, Informatie Dynfos

Dynfos Business Solutions B.V.	
Adres	James Wattstraat 1
Telefoon nummer	0548-622247
Website	www.dynfos.nl
Postcode	7442 DC Nijverdal
Aantal personen	12

Tabel 2, Informatie Dennis Gortemaker

Bedrijfsbegeleider	
Roepnaam	Dennis
Achternaam	Gortemaker
Email	d.gortemaker@dynfos.nl
Telefoon nummer	0548-622247
Functie	General Manager

Tabel 3, Informatie Rogier Hommels

Afstudeerbegeleider	
Roepnaam	Rogier
Achternaam	Hommels
Email	r.m.hommels@saxion.nl

Tabel 4, Informatie Damian Jansen

Afstudeerder	
Roepnaam	Damian
Achternaam	Jansen
Email	433928@student.saxion.nl
Telefoon nummer	06-53902936

Versiebeheer

Tabel 5, Versiebeheer

Versie nr.	Inhoud van de versie	Datum
0.1	Minimale opzet van koppen, inhoudsopgave en footer/header	17-04-2020
0.2	Titelpagina + Contactgegevens + Versiebeheer	24-04-2020
0.3	Inleiding	21-5-2020
0.4	Literatuuronderzoek	25-5-2020
0.5	Methodologie + implementatie	26-5-2020
0.6	Aanpassingen onderzoek + conclusie + discussie	27-5-2020
0.7	Aanbevelingen + reflectie	28-5-2020
0.8	Samenvatting + voorwoord	29-5-2020
0.9	Voorwoord + spellingcheck	2-5-2020
1.1	Implementatie feedback dhr. Gortemaker	8-6-2020
2.0	Requirements	11-9-2020
2.1	Ontwerp	18-9-2020
2.2	Implementatie + conclusie + aanbevelingen	25-9-2020
2.3	Afronden implementatie + conclusie + aanbevelingen + onderzoek	29-9-2020
2.4	Implementatie feedback dhr. Gortemaker	30-9-2020
2.5	Implementatie feedback dhr. Hommels	2-10-2020
2.6	Implementatie feedback 2e lezer	9-10-2020
3.0	Verbeteringen verslag	26-10-2020

Voorwoord

Als laatstejaars student van de opleiding HBO-ICT Software Engineering op het Saxion in Enschede. Het doel was een bedrijf te vinden dat geschikt was voor het afstuderen. Bij een bedrijvenmarkt, die het Saxion geregeld had, heb ik de kans gehad om met Dynfos te spreken. Na goed geïnformeerd te zijn over de bezigheden van Dynfos, besloot ik om de site te bekijken voor de opdrachten die ze open hadden staan. Op de site waren heel wat leuke opdrachten te vinden, dus besloot ik om een gesprek aan te vragen. Tijdens dat gesprek werd mij goed uitgelegd wat er gedaan moest worden en werd mij de software die Dynfos maakt laten zien. Het gesprek verliep soepel en positief. Ook de opdracht leek uitdagend. Dit zijn de redenen voor het kiezen voor Dynfos als afstudeerbedrijf.

Ik wil Dynfos graag bedanken voor het voorrecht om bij hun de afstudeerperiode te mogen doen. Gedurende het project heb ik veel geleerd. Ik heb prettig kunnen werken op mijn mooie werkplek die mij door Dynfos is aangewezen. Ook wil ik graag alle werknemers van Dynfos bedanken voor de hulp die ik van hun gekregen heb. Ik wil ook dhr. Gortemaker hartelijk danken. De samenwerking met dhr. Gortemaker verliep erg fijn. Tijdens de afstudeerperiode heb ik gebruik kunnen maken van de expertise die dhr. Gortemaker heeft. Dhr. Gortmaker heeft details kunnen aanwijzen die ik anders niet zo snel had opgemerkt, daarnaast was het erg handig om een andere zienswijze op het project te hebben. De feedback die dhr. Gortemaker leverde was erg welkom.

Ik wens u veel leesplezier.

Damian Jansen

Nijverdal, 2 november 2020

Samenvatting

Het afstudeerproject mocht bij Dynfos gemaakt worden. Dynfos is een distributeur van 3D Catsoftware genaamd IronCAD en maakt zelf een ERP-systeem gemaakt Dynfos. Dynfos heeft een mobiele applicatie die voor verouderde scanners is gemaakt die op Windows CE draaien. De scanners zullen binnenkort niet meer leverbaar zijn, dus moet er een alternatief komen. Om te beginnen is er een onderzoek gestart met als hoofdvraag: Hoe kan er een nieuwe applicatie ontwikkeld worden, die de klant informatie kan ontsluiten, rekening houdend met veiligheid en distributie?

Een groot deel van het onderzoek is met deskresearch gedaan. Om gedeeltes van de vragen te controleren is er doormiddel van fieldresearch getest, om de statements die gedaan zijn, te onderbouwen.

Uit het onderzoek zijn de belangrijkste features en technologieën naar voren gekomen die bijdragen tot de veiligheid van de applicatie. Tijdens het project zal er gebruik gemaakt worden van DevExpress, een framework voor het maken van business applicaties. Er zijn technieken gevonden die gebruikt kunnen worden om de applicatie veiliger te maken. Om een veilige verbinding met de database te verkrijgen, kan er een API gebruikt worden die door middel van REST, de data kan manipuleren. Het gebruik van een speciale database gebruiker en views, kan ertoe bijdragen dat alleen de benodigde data uit de database kan worden gemanipuleerd. Om de gebruiker van de applicatie te authentiseren, wordt er gebruikt gemaakt van een set met inloggegevens. Het inloggen van de gebruiker zal naast een normaal gebruikersnaam/wachtwoord combinatie, ook een extra parameter bevatten die het mogelijk maakt de gebruiker naar de juiste administratie te sturen. De applicatie zal uiteindelijk gepubliceerd kunnen worden op de Play Store en de App Store.

Tijdens het onderzoek werd er bekend gemaakt dat de methode voor het maken van een mobiele applicatie: XAF mobile, weggehaald werd en door XAF Blazor werd vervangen. Tijdens de implementatie fase van het project is er geprobeerd met Blazor de applicatie te maken, omdat dit voor Dynfos de toekomst is voor een webapplicatie. Door het complexe Dynfos project bleek de Blazor bèta niet over genoeg functionaliteiten te bezitten om de opdracht te voltooien. De keuze is uiteindelijk gemaakt om toch de oude methode voor het maken van mobiele applicaties te gebruiken, ondanks dat deze niet meer ondersteunt wordt. Om dit te kunnen doen moest er een oude versie van DevExpress gebruikt worden. Tijdens de implementatie van de applicatie zijn er veel problemen getackeld om te voldoen aan de requirements van het project. De applicatie is als proof of concept goed te gebruiken voor de toekomst. Hierbij is niet alleen rekening gehouden om met de oude methode een nieuwe mobiele applicatie te maken, maar ook om een applicatie te maken met Blazor.

Er zijn aanbevelingen voor de toekomst van dit project. Er zal na dit project meer onderzoek naar Blazor gedaan kunnen worden. Ook is het belangrijk dat er goede regels worden afgesproken met het netwerkbeheer voor mobiele apparaten, die door de firewall de applicatie mogen benaderen. Dit is voor zowel Dynfos en de klanten met een firewall van belang.

Inhoudsopgave

Figuren- en tabellenlijst

Hieronder zullen alle referenties staan die verwijzen naar figuren of tabellen.

Tabellenlijst

Figurenlijst

Leeswijzer

Inleiding
Dit hoofdstuk zal de aanleiding van de opdracht beschrijven. De vragen voor het onderzoek hier getoond worden.

DevExpress
Wat DevExpress is en waarom er voor DevExpress is gekozen, wordt hier beschreven. Verder wordt er beschreven hoe de huidige structuur van het Dynfos project er uit ziet. Als laatste wordt er besproken hoe het Dynfos project gebruik maakt van DevExpress.

Onderzoek
Dit hoofdstuk zal ingaan op vragen die gesteld worden aan de hand van de opdracht. Het oplossen van deze vraagstukken zal de implementatie van de applicatie bevorderen.

Methodologie
In dit hoofdstuk zal er worden beschreven hoe het onderzoek is uitgevoerd. Het hoofdstuk beantwoordt de vraag: welke data-analysemethode(n) zijn aangehouden tijdens het onderzoek?

Requirements
Het hoofdstuk requirements gaat over de wensen die Dynfos heeft voor het project. Deze wensen helpen de koers van het product te wijzen.

Ontwerp
In dit hoofdstuk wordt besproken hoe het nieuwe systeem is ontwerpen. Ook zal er worden uitgelegd wat de reden is voor de ontwerpkeuzes die gemaakt zijn.

Implementatie
In dit hoofdstuk staat een beschrijving van de implementatie van de applicatie. Ook wordt er besproken tegen welke problemen zijn aangelopen en hoe deze zijn opgelost.

Conclusie
Dit is een conclusie van het gehele project. Er zal worden ingegaan op welke requirements voltooid zijn. Als laatste zal het hoofdstuk laten zien hoe de kwaliteit van de code is gewaarborgd.

Proces
Het hoofdstuk zal het proces dat aangehouden is bespreken.

Aanbevelingen
Er worden aanbevelingen gedaan over de implementatie van de applicatie voor de toekomst. Dit kunnen verbeter punten zijn, maar bijvoorbeeld ook hele toevoegingen aan de applicatie.

Reflectie
In dit hoofdstuk zal er een reflectie plaatsvinden over het gehele project.

1. Inleiding

Als laatstejaars student van het Saxion in Enschede, heb ik de opdracht gekregen om een afstudeerproject te voltooien bij een bedrijf waar ik als afstudeerder zal dienen. Het bedrijf waar ik de afstudeeropdracht zal afronden, is bij Dynfos Business Solutions B.V.

Dynfos is een bedrijf met twee producten. Eén van de producten is een ERP-oplossing. Deze ERP-oplossing wordt verkocht aan andere bedrijven. Het tweede product is het verkopen van IronCAD, dit is een 3D CAD-software. Het resultaat van deze twee producten is een volledig ERP-systeem dat goed integreert met IronCAD. Ook heeft Dynfos een app die functies beschikt van het ERP-systeem, zoals het registreren van tijd en het bekijken van urenverantwoordingen.

De opdracht die uiteindelijk gekozen is, staat in verband met scanners die Dynfos en hun klanten gebruiken voor het inscannen van producten en het uitvoeren van bepaalde acties. Deze scanners zullen binnenkort niet meer beschikbaar zijn.

1.1 Probleemstelling

Voor het scannen van barcodes op producten en locaties, is er een Windows CE applicatie gemaakt. Het scannen van de barcodes wordt gebruikt voor het verplaatsen, in- en uitboeken en het opvragen van informatie. Er zijn een aantal redenen waarom Dynfos de Windows CE applicatie wil vervangen. De eerste reden is dat de applicatie is verouderd. Een tweede reden is dat er geen ondersteuning meer is voor de scanners. De laatste reden is het installatie proces. Het installatie proces moet handmatig gedaan worden. Bovendien zijn de scanners die Dynfos gebruikt voor de applicatie binnenkort niet meer beschikbaar. De applicatie draait nu in een andere omgeving als het Dynfos ERP-systeem, maar gebruikt wel enkele resources die de Dynfos App ook gebruikt. Dit is erg lastig, omdat alle resources hierdoor dubbel te vinden zijn. Mocht het zo zijn dat er een nieuwe aanpassing of feature bij het ERP-systeem wordt toegevoegd die direct gevolgen heeft voor de scan applicatie, dan moet dit later ook weer aangepast worden in de scan applicatie.

De toekomstvisie van Dynfos, is één nieuwe mobiele applicatie die data in real-time uit de database kan halen. Ook zal er gekeken worden hoe de applicatie gedistribueerd moet worden, om deze makkelijk te kunnen installeren. Deze nieuwe applicatie zal twee systemen vervangen. De functies van de Windows CE applicatie en de Dynfos App, zullen te vinden zijn in deze nieuwe applicatie. Om dit uiteindelijk te realiseren zijn er wel grote veranderingen die aangebracht moeten worden in de applicatie. Zo moet er worden gezorgd dat de klanten kunnen inloggen en bij hun eigen administratie moeten kunnen komen, maar het moet niet mogelijk zijn dat er in de andere administraties ingekeken kan worden.

Tijdens dit project wordt er gefocust op de authenticatie en hoe de gebruikers in het systeem komen en hoe de applicatie de databases benadert. Voor deze features is een proof of concept gemaakt die laat zien hoe deze features zouden kunnen werken.

Aan de hand van het probleem is er een hoofdvraag opgesteld:

Hoe kan er een nieuwe applicatie ontwikkeld worden, die de klant informatie kan ontsluiten, rekening houdend met veiligheid en distributie?

Deze hoofdvraag kan makkelijker beantwoord worden met behulp van een aantal deelvragen. Dit zijn de deelvragen:

- Hoe kan er een verbinding vanuit de mobiele applicatie naar de productiedatabase worden gemaakt, die veilig is?
- Hoe kan de toegang tot de database gelimiteerd worden tot de data die belangrijk is voor de mobiele gebruikers?
- Wat is een methode van authenticatie die herbruikbaar is voor verschillende apparaten van een klant?
- Hoe kan de applicatie toegang verlenen tot een systeem aan de klant zonder toegang te geven tot systemen van andere klanten?
- Wat is een manier waarop de applicatie kan worden gedistribueerd, waarbij regelmatige updates een belangrijk aspect zijn?

2. DevExpress

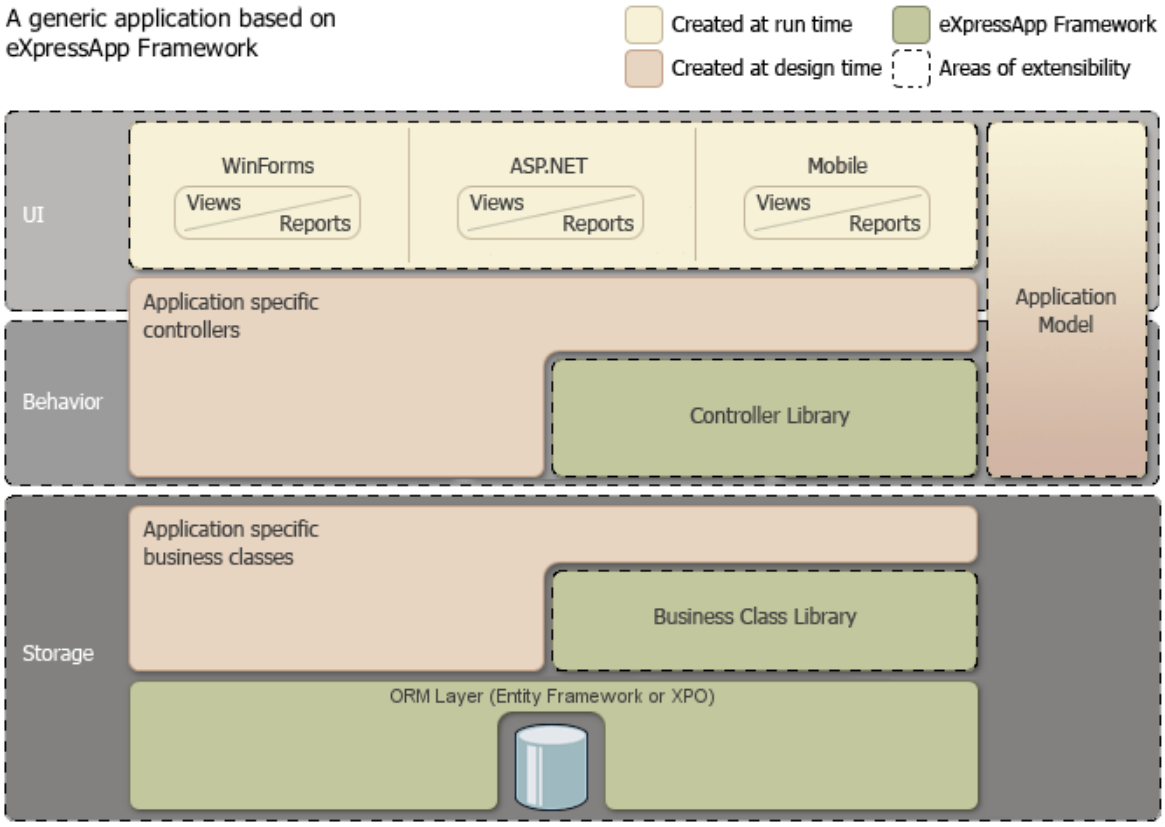
De keuze is gemaakt om een framework van DevExpress te gebruiken, dit is een randvoorwaarde die Dynfos gesteld heeft aan het project. Deze keuze is gemaakt omdat de grote hoeveelheid ervaring die Dynfos met DevExpress heeft, erg handig is voor dit project. Dynfos werkt al met DevExpress sinds 2008.

Het ERP-systeem dat Dynfos als een product aanbiedt is gemaakt met behulp van DevExpress. Als het nieuwe project met DevExpress werkt, zorgt dat ervoor dat één van de redenen dat de oude oplossing aan vervanging toe is, al opgelost is. Als er met DevExpress geïmplementeerd wordt, zal de code die het ERP-systeem en de te vervangen mobiele applicatie nodig hebben, op één plek kunnen staan en is het niet meer nodig om de ontwikkeling van beide applicaties dubbel te onderhouden.

De grote hoeveelheid ervaring van DevExpress die aanwezig is bij Dynfos en het feit dat het ERP-systeem ook met DevExpress werkt, is de reden dat DevExpress voor dit project is gekozen.

2.1 Introductie

DevExpress is een leverancier van een verscheidenheid van tools die helpen bij het maken van applicaties. Het Dynfos ERP-systeem is ook gemaakt met behulp van een aantal tools van DevExpress. Er is gekozen om de mobiele applicatie ook te maken in DevExpress. Dynfos maakt gebruik van het XAF (eXpressApp Framework), dit is een framework die het makkelijk maakt om een business applicatie te maken. Dit framework kan ook worden gebruikt voor het maken van een mobiele applicatie. Voor de mobiele applicatie zal het XAF framework ook gebruikt worden, zodat het mogelijk is om de mobiele applicatie samen met het al bestaande Dynfos ERP-systeem te onderhouden. Daarmee zal er ook geen probleem meer zijn met dubbele code. Het XAF framework werkt volgens de architectuur in het ondergaande figuur[6].



Figuur 1, XAF architectuur

De architectuur van XAF begint met een ORM laag. Deze ORM tools zorgen ervoor dat de database waar de klanten hun data in hebben staan, bewerkt kan worden door middel van code structuren, classes, eigenschappen en attributen. Dit maakt het aanpassen van data gemakkelijker dan de traditionele manier van het aanroepen van een database en daarna pas mutaties uit te voeren.

XAF heeft een ingebouwde set met business classes die je kunt gebruiken. Deze classes representeren tabellen in de database. De classes kunnen op zich zelf gebruikt worden of kunnen worden uitgebreid door er eigen classes voor te maken.

Controllers worden gebruikt om user interactie af te handelen. XAF heeft ook een aantal ingebouwde controllers die gebruikt kunnen worden. De ingebouwde controllers zijn voor het managen van data, voorbeelden zijn: het verwijderen van records, een tekst search en het maken van nieuwe records. De controllers kunnen ook zelf gemaakt of uitgebreid worden.

De UI wordt gescheiden van de business data, zodat het mogelijk is om verschillende UI's te maken als het om dezelfde data gaat. Dit is bijvoorbeeld handig als je een Windows applicatie en een mobiele applicatie maakt die dezelfde data moeten kunnen beheren, wat in deze situatie het geval is.

Het applicatie model zorgt ervoor dat de UI automatisch gegenereerd kan worden aan de hand van informatie uit classes, labels en data types. Het model slaat alles op in een XML bestand en kan ook worden aangepast.

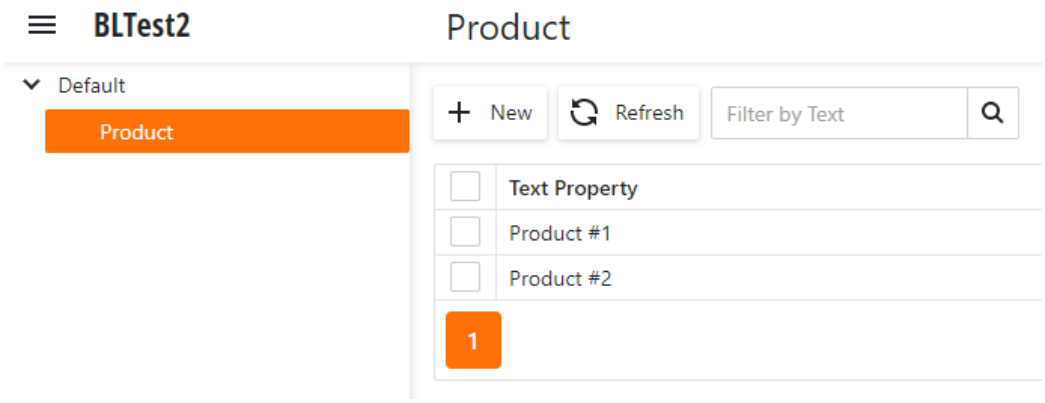
2.2 Blazor & Mobile

Het XAF framework stelt de gebruiker ertoe instaat om code te schrijven die gebruikt kan worden voor verschillende platformen. Deze platform onafhankelijke code kan gedeeld worden door Windows, web en mobiele applicaties.

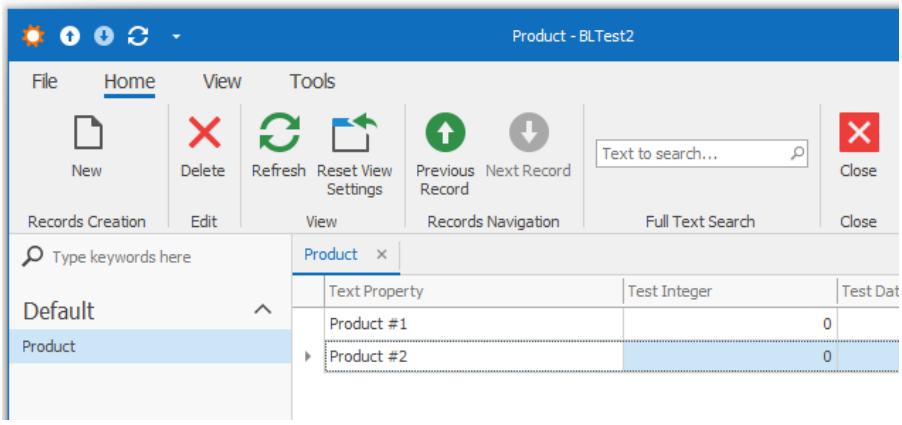
DevExpress is actief bezig om de mobiele ondersteuning te verbeteren. Tijdens het onderzoek heeft DevExpress er voor gekozen om de oude methode om mobiele applicaties te maken via XAF te gaan veranderen. DevExpress is Blazor voor XAF aan het ontwikkelen als vervanging van de oude mobiele methode die ze gebruiken[7] . DevExpress heeft de oude mobiele methode al stopgezet voor de versie van XAF die gebruikt wordt bij Dynfos, dit heeft als uitwerking dat het niet meer ondersteund wordt door DevExpress. Dit is geen optimale situatie aangezien XAF Blazor nog niet volledig volgroeid is en de oude mobiele methode niet meer goed zal zijn voor de toekomst.

Na overleg met de opdrachtgever is de keuze gemaakt om de applicatie zo ver mogelijk te implementeren met Blazor, het alternatief dat DevExpress aan het maken was. Er is gekozen om dit te doen, omdat Blazor wel support krijgt van DevExpress. Dit is voor de toekomst beter, aangezien de applicatie dan niet meteen verouderd is.

De keuze om Blazor te gaan proberen bracht een probleem met zich mee. Het onderzoek dat gedaan is, ging grotendeels over de oude mobiele implementatie van de applicatie. Het was moeilijk om Informatie over XAF Blazor te vinden omdat er weinig documentatie over de implementatie van Blazor door DevExpress was vrijgegeven. Om erachter te komen of het mogelijk was om Blazor te gebruiken is er een test applicatie gemaakt. Deze test of Blazor toegevoegd zou kunnen worden aan een Windows applicatie, net zoals de nieuwe applicatie dat ook zou doen.



Figuur 2, Snippet Blazor test



Figuur 3, Snippet Win test

De configuratie van Blazor werkt iets anders dan de configuratie van XAF mobile. Het heeft wat tijd gekost om de configuratie van Blazor op te stellen. De configuratie is aangepast voor het Blazor project. Blazor maakt geen gebruik van een configuratie bestand, waar alle opties voor het project te configureren zijn. Om de configuratie te beheren voor Blazor, moet er in de code, rekening gehouden worden met instelbare configuraties, zoals de connectie string.

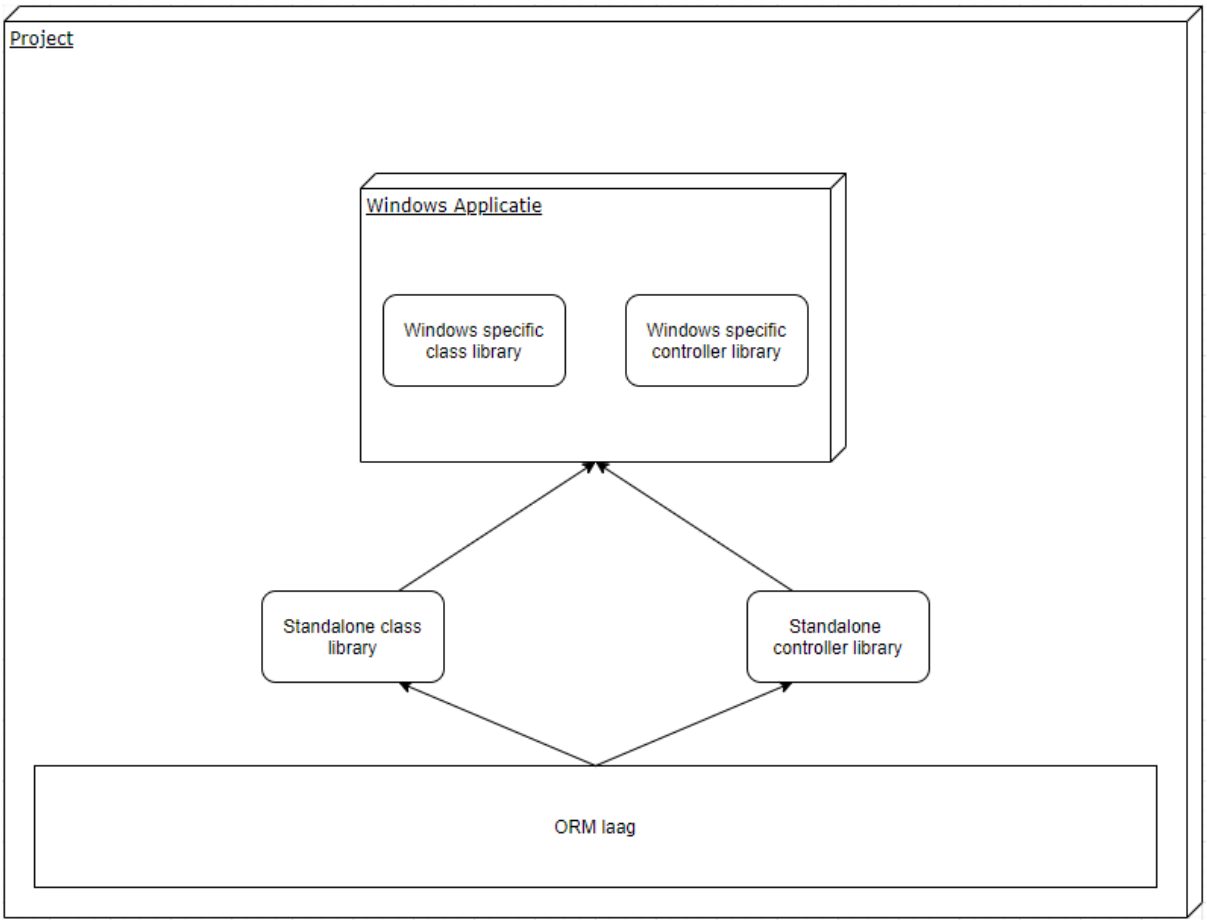
Het samenvoegen van de test met het Dynfos project ging echter minder goed. Met hulp van de opdrachtgever en het DevExpress team zijn er een heel aantal foutmeldingen verholpen. Er was echter één probleem dat niet op te lossen was. In overleg met het DevExpress, bleek dat er wat extra modules zijn die het Dynfos project heeft, die nog niet in de bèta van Blazor zijn geïmplementeerd. De module voor het maken van rapporten, die door DevExpress ter beschikking is gesteld, bleek niet nog niet beschikbaar te zijn in Blazor. Na contact te hebben gehad liet het DevExpress team weten, dat de module in kwestie ook niet binnenkort beschikbaar gemaakt zullen worden voor Blazor. Het bleek dus niet mogelijk te zijn om het Blazor project in de huidige staat van Blazor samen te voegen met het Dynfos project voor Windows.

Aangezien Blazor niet gebruikt kon worden om het project in dit traject tot een succesvol einde te brengen is er besloten de nieuwe applicatie te maken met een oudere versie van DevExpress. Deze versie heeft nog toegang tot de oude methode voor het maken van mobiele applicaties. Deze methode wordt niet meer ondersteund, maar beschikt wel over alle benodigde functionaliteiten, zoals het gebruik van een ORM.

Deze tests met Blazor hebben veel tijd gekost, maar ze zijn uiteindelijk handig gebleken voor de toekomst. Mocht Dynfos later willen overgaan op Blazor, dan zijn de mogelijkheden met Blazor bekend. De implementatie fase van Blazor heeft veel inzichten geboden. Zo zullen alle functionaliteiten die nodig zijn voor de applicatie beschikbaar zijn in Blazor als deze wordt vrijgegeven. Zodra Blazor vrijgegeven wordt, zal de applicatie omgezet kunnen worden naar Blazor. De verwachting is dat XAF Blazor uitgegeven zal worden in het begin van het jaar 2021.

2.3 Dynfos

Dynfos maakt gebruik van de XAF-architectuur. De architectuur van het ERP-systeem ziet er als volgt uit.



Figuur 4, Dynfos structuur

Eén manier om een projectstructuur via XAF op te zetten, is met het gebruik van een modulair systeem. Dit systeem gebruikt het huidige Dynfos project. Dit systeem maakt gebruik van een standalone module die wordt gebruikt voor de drie verschillende applicatie platformen. In deze module staan de basis objecten die gebruikt kunnen worden voor alle applicaties. Dit kunnen ook business objecten zijn die een item in een database moeten voorstellen, zoals een persoon. De applicatie kan er zelf voor zorgen dat hij voor de database items aanmaakt aan de hand van classes.

Voor het al bestaande Dynfos project is er al een uitgebreide standalone module waar alle Business objecten in staan. Daarnaast zijn er ook heel wat andere classes die niet in de standalone module staan, dit is noodzakelijk voor de functies die alleen voor Windows applicaties bedoeld zijn.

De stand-alone module bezit class- en controller libraries die in XAF zitten. Deze module is platform onafhankelijk en zal te vinden zijn in alle XAF-applicaties die gemaakt worden. Alle ingebouwde classes en controllers zijn bij default uitgeschakeld om een kale applicatie te geven. Deze functie zorgt ervoor dat er geen onnodige classes en controllers zijn. Als je echt een voorgebouwde controller of class nodig hebt kan je die inschakelen.

Naast de stand-alone module heeft een XAF-applicatie altijd een platform afhankelijke module. Deze module zal verschillen per platform. Ook is het mogelijk dat een XAF-applicatie meerdere platform afhankelijke modules heeft, als de applicatie voor verschillende platformen wordt gemaakt. In deze module staan alle classes en controllers in die specifiek voor dit platform zijn. Deze module erft van de standalone module, zodat er gebruik gemaakt kan worden van alles wat er in de standalone module is gemaakt.

De module die als laatst wordt gebruikt is de platform afhankelijke module die verantwoordelijk is voor het uitvoeren van de applicatie. In deze module staat alle configuratie die nodig is om de applicatie in te stellen. Verder kunnen er hier extra lettertypes en resources toegevoegd worden.

3. Onderzoek

Het onderzoek is cruciaal voor de implementatie van de proof of concept applicatie. Het onderzoek zal ingaan op ontwerp keuzes en beste werkwijzen die gevonden zijn. Deze keuzes en beste werkwijzen zullen ook verwerkt worden in het proof of concept.

3.1 Onderzoeksvragen

Om een onderzoek goed uit te voeren is er voor gekozen om stappen te plannen voor het onderzoek. Deze plannen worden gemaakt aan de hand van deelvragen die samen de hoofdvraag moet beantwoorden.

De deelvragen zullen dus ook altijd bijdragen aan het onderzoek, sommige deelvragen zullen met elkaar overlappen.

De hoofd- en deelvragen zijn benoemd in hoofdstuk **1.1.**

3.2 Methodologie

Het grootste gedeelte van het onderzoek is kwalitatief, maar er zijn ook gedeeltes die kwantitatief zijn, zoals de informatie over QR-codes en de informatie over app downloads.

De data die gebruikt is voor de kwantitatieve stukken komen van het internet en zijn dus niet zelf verkregen. De informatie die is gebruikt voor de kwalitatieve stukken komt niet altijd van het internet. Delen van de informatie zijn verkregen door het testen van applicaties, zoals het publiceren van een applicatie en het gebruik van de oude applicatie.

De literatuur die gebruikt is komt van het internet. Het grootste gedeelte van de informatie komt van de DevExpress site, maar er zijn ook bronnen die verwijzen naar andere sites.

Het onderzoek is volgens deskresearch vergaard. Alle informatie die is verkregen, was al bekend. Er zijn geen onderzoeken, interviews of enquêtes gehouden, wel zijn er tijdens het onderzoek kleine applicaties gemaakt om te testen of gedeeltes van het onderzoek ook werken als het gaat om implementatie.

3.2.1 Onderzoek verloop

Het onderzoek verliep in een redelijk tempo. Voor sommige onderwerpen heeft het echter wat langer geduurd dan gedacht. De reden hiervoor was dat de documentatie die DevExpress had voor minder recente versie van DevExpress was, dan dat er gebruikt werd voor de implementatie.

Het uiteindelijke onderzoek is verlopen zoals in het onderzoeksopzet van het plan van aanpak besproken is.

3.3 Deelvraag 1: Een veilige database verbinding

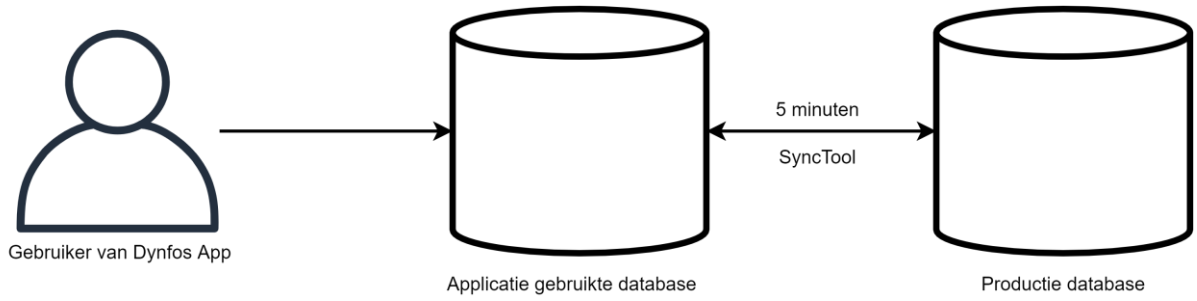
Deelvraag 1 luidt: Hoe kan er een verbinding worden gemaakt met de productiedatabase, die veilig is?

De applicatie die gemaakt moet worden, moet een verbinding kunnen maken met de database. Dit moet kunnen zodat gebruikers de data kunnen inzien die zij beheren. Deze verbinding moet veilig zijn, door de toegang tot de database te limiteren.

De veilige verbinding met de database kan niet alle veiligheidsproblemen voorkomen. Zo is het mogelijk dat een hacker de gegevens van één van de gebruikers kan achterhalen, doordat er niet goed op deze gegevens gelet is door de gebruiker zelf. De standaard van veiligheid die door Dynfos graag aangehouden wordt, is dat het een hacker langer kost om het systeem in te breken dan het de softwareontwikkelaars kost om te achterhalen dat er een hacker in de database probeert te komen.

3.3.1 Huidige situatie van de Dynfos App

Het Dynfos project voor Windows maakt ook verbinding met een database waar alle data voor de gebruikers in staat. Deze database staat bij de klant, maar wordt door de gebruikers benaderd door een externe database. Hieronder zal te zien zijn hoe er verbinding gemaakt wordt met de database.



Figuur 5, Oude database architectuur

De verschillende gebruikers van de Dynfos App kunnen alleen inloggen op een externe database om hun data op te halen. Deze externe database wordt gehost op een server van SmarterASP. De productiedatabase kan alleen benaderd worden door middel van de externe database die de data kopieert van de productiedatabase. Dit resulteert in een interne database die niet kan worden bereikt door de buitenwereld. Deze situatie is daarom ook veilig, aangezien er niet direct toegang kan worden gekregen tot de interne productiedatabase. Er is wel een nadeel aan deze huidige situatie, de synchronisatie van data is niet real-time. Doordat dit niet real-time verloopt worden de twee databases elke vijf minuten gesynchroniseerd. Doordat de Dynfos App gebruikt wordt tijdens het synchroniseren van data, kunnen er synchronisatie fouten ontstaan. Deze synchronisatie problemen gebeuren elke dag en moeten dan ter plekke ook opgelost worden.

3.3.2 Opties

De mobiele applicatie heeft ook een manier nodig om te communiceren met de database. Om een beslissing te maken over welke methode het meest geschikt zou zijn voor dit project zullen alle verschillende opties voor het verbinden met een database hier besproken worden.

- Externe database
- REST API service
- Directe verbinding

Om een oordeel te maken over de beste optie voor dit project zijn er een aantal eigenschappen waar ze op beoordeeld worden. De eigenschappen in kwestie zijn:

- Veiligheid
- Onderhoudbaarheid
- Integratie met DevExpress

Van deze eigenschappen weegt veiligheid het zwaarst mee, omdat de data van de klanten privé moet worden gehouden. Het is echter wel vereist dat de oplossing geïntegreerd kan worden met DevExpress.

3.3.3 Externe database

De externe database waar het hier over gaat is een database die de inhoud van de productiedatabase kopieert om hier vervolgens mutaties op uit te voeren. Dit gebruik van een externe database wordt ook gebruikt door het Dynfos project voor Windows. Na dit in de praktijk te hebben gebruikt, is het de wens om deze methode niet te gebruiken als er een ander alternatief is. Figuur 5, Oude database architectuur laat zien hoe dit eruit ziet.

3.3.3.1 Veiligheid

De mate van veiligheid die het gebruik van een externe database biedt, is redelijk goed. Als een gebruiker zich aanmeldt bij de applicatie zal de aanvraag verstuurd worden naar de externe database en niet de interne productiedatabase, hierdoor kan de gebruiker ook niet achterhalen hoe de applicatie verbinding maakt met de productiedatabase.

Het gebruik van een externe database is ook veiliger in verband met de firewall. De firewall die zorgt ervoor dat alleen de verbinding van de externe database is toegestaan tot de interne database, dit is een extra veiligheid die deze manier van verbinden met zich meebrengt.

3.3.3.2 Onderhoudbaarheid

Er is zeker onderhoud aanwezig met deze methode van verbinden met de database. Net zoals met het Dynfos project voor Windows krijg je te maken met synchronisaties die regelmatig gedaan moeten worden. Als er een synchronisatie plaatsvindt, kunnen er foutmeldingen plaatsvinden als er data is veranderd die nog gesynchroniseerd moet worden, dit veroorzaakt een conflict in resultaten. Dit conflict moet handmatig opgelost worden. Dit resulteert in veel onderhoud, vooral omdat deze synchronisatie problemen meer dan eens op een dag kunnen plaatsvinden.

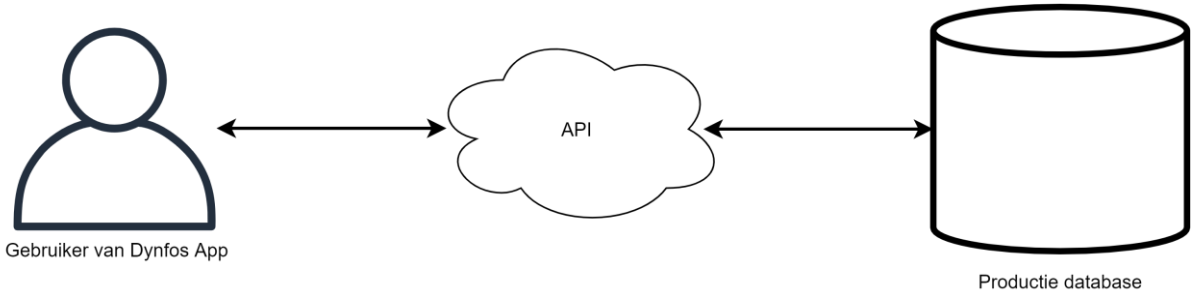
3.3.3.3 *Integratie met DevExpress*

De integratie met DevExpress verloopt soepel. De aanpassingen die moeten gedaan worden om dit te integreren met DevExpress zijn miniem. De connectie met de database wordt gemaakt aan de hand van een voor opgestelde connectie string. Deze string zal moeten aangepast worden om verbinding te maken met de externe database in plaats van de interne productiedatabase.

3.3.4 REST API service

Een REST API is code die twee softwareprogramma's ertoe in staat stelt om met elkaar te communiceren door middel van HTTP. REST is een technologie die weinig bandbreedte gebruikt, waardoor het goed is voor mobiele apparaten omdat deze niet altijd een stabiele verbinding hebben[8] .

Een REST API kan gebruikt worden om de database van data te voorzien en data op te halen zonder direct met een database te communiceren. Dit resulteert vaak in een API die erg simpel is om aan te roepen, zodat er geen uitwerking van database instructies in de applicatie staat, maar op de server waar de REST API draait. Het scenario zal er zo uitzien:



Figuur 6, API

3.3.4.1 *Veiligheid*

Het gebruik van een REST API is redelijk veilig. Zoals bij het gebruik van een externe database kan er niet een directe verbinding opgezet worden tussen de applicatie en de productiedatabase. Alleen de API kan data aanvragen en manipuleren van de database. Dit heeft als voordeel dat de API het enige systeem is dat door de firewall kan.

3.3.4.2 *Onderhoudbaarheid*

De REST API heeft ook onderhoud nodig, maar geen onderhoud dat dagelijks moet worden uitgevoerd. De REST API draait op een server, dit betekent dat er moet worden gezorgd dat de server online blijft, als deze wel offline is dan kan de applicatie ook niet worden gebruikt op een functioneel gebied.

Als de logica voor het ophalen en manipuleren van data aangepast moet worden kan dit niet in de applicatie zelf, maar moet dit gebeuren in de API.

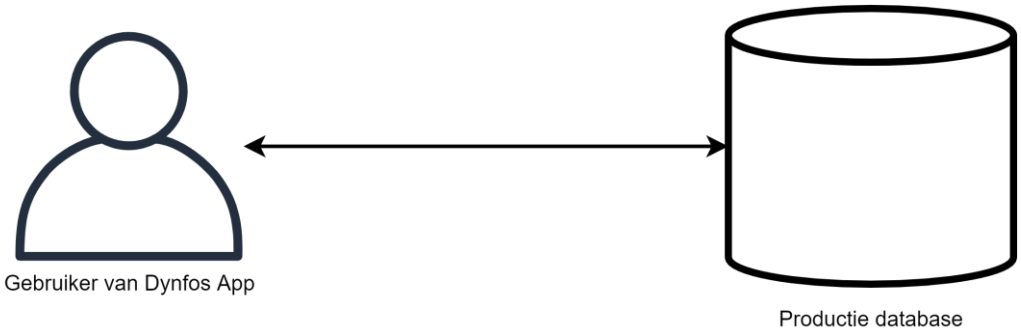
3.3.4.3 *Integratie met DevExpress*

De integratie met DevExpress is aanwezig, maar wel in een gelimiteerde wijze. Het is niet mogelijk om een eigen REST API te maken. Dit komt doordat de ORM laag van DevExpress

specifieke calls stuurt naar de API om de data goed te bewerken. Dit kan vervelend zijn, want er is niet te achterhalen wat de API precies uitvoert. Het is echter wel mogelijk om gebruik te maken van de REST API die DevExpress zelf gebruikt. Dit werkt zowel voor de Blazor als de oude mobiele implementatie van DevExpress. Het gebruik van een REST API kan alleen als er XPO als ORM gekozen wordt. Deze API is gemaakt om een goede verbinding met de database naar keuze te maken, zodat de voordelen van een REST API ook te gebruiken zijn in XAF.

3.3.5 Directe verbinding

Een directe verbinding naar de database is eenvoudig en snel op te zetten. Hierdoor zal de applicatie direct query-en naar de database van Dynfos. Het volgende scenario is geschetst:



Figuur 7, Directe verbinding

3.3.5.1 Veiligheid

Hoewel deze methode van verbinden nog wel gebruik maakt van een firewall, is het niet te voorkomen dat er verschillende verbindingen toegestaan zijn. Ieder mobiel apparaat wat verbinding probeert te maken met de database, moet een afzonderlijke connectie naar de database hebben. Alle gebruikers moeten dus worden toegestaan om de verbinding met de database aan te leggen.

3.3.5.2 Onderhoudbaarheid

In tegenstelling tot de andere oplossingen vereist deze methode van verbinden geen extra onderhoud naast het online houden van de productiedatabase.

3.3.5.3 Integratie met DevExpress

De integratie met DevExpress is erg simpel, om een directe verbinding met de database te krijgen hoeft er niets veranderd te worden. XAF is automatisch gemaakt om direct met de database te verbinden, bij de distributie van de applicatie zal het verbinding proberen maken met de ingestelde database.

3.3.6 Conclusie

Om te kiezen welke methode van verbinden het best bij het project past worden de voor en nadelen van de verschillende methoden tegen elkaar opgewogen. Hieronder zullen de plus en minpunten te zien zijn van de verschillende methoden, deze zijn gebaseerd op de uitleg hierboven.

Tabel 6, Database verbinding keuzetabel

	Veiligheid	Onderhoudbaarheid	Integratie met DevExpress
Externe database	+	-	+
REST API	+	+	+
Directe verbinding	+/-	+	+

De integratie met DevExpress is bij alle drie de methoden te realiseren. Er zijn geen extra stappen vereist om de REST API te integreren met DevExpress. De resultaten van de integratie zijn allemaal positief, dit betekent dat er bij het kiezen van de verbinding methode deze eigenschap niet wordt meegenomen.

Van de twee criteria die overblijven is veiligheid het belangrijkste onderdeel. De directe verbinding is makkelijk qua onderhoud. De veiligheid speelt echter een grote rol in de database verbinding, daarom zal deze methode voor het verbinden met de database afvallen.

De twee overgebleven verbindingsmethoden zijn de REST API en de externe database. Er is al ervaring met de verbindingsmethode die gebruik maakt van een externe database, aangezien deze methode al gebruikt wordt voor het Dynfos project op Windows. Uit ervaring kan er gezegd worden dat deze methode niet fijn is in het gebruik door het vele onderhoud dat het nodig heeft. Dit is de reden waarom de REST API het handigst lijkt voor dit project. Er moet echter wel rekening gehouden worden met de flexibiliteit van de API, aangezien de API niet zelf gemaakt kan worden. De gegenereerde API van XAF is gemaakt om alle data objecten die in de code staan te kunnen verwerken, dus zou dit geen groot probleem moeten zijn voor het project.

3.4 Deelvraag 2: Database toegang

Deelvraag 2 luidt: Hoe kan de toegang tot de database gelimiteerd worden?

De database die wordt onderhouden door Dynfos is erg groot. De database bevat enorm veel informatie voor verschillende doeleinden. Alle informatie die in de database staat wordt gebruikt door de Dynfos applicatie voor Windows. Al deze informatie is echter niet nodig voor de mobiele applicatie. De mobiele applicatie kan gebruikt worden door monteurs voor het opvragen van producten, het verplaatsen daarvan en het registreren van tijd.

Er zijn een aantal mogelijkheden om de toegang van de mobiele applicatie te verminderen, zodat de applicatie niet bij alle andere belangrijke data kan. Hieronder worden opties besproken die gebruikt zouden kunnen worden. Al deze opties kunnen samen gebruikt worden, er hoeft dus geen keuze gedaan te worden tussen de verschillende opties. Wel moet de keuze gemaakt worden of de optie gebruikt wordt of niet.

3.4.1 Database User

Als de applicatie een verbinding met de database wil leggen, moet er worden ingelogd op de database. Het inloggen op de database verloopt hetzelfde als het inloggen van een gebruiker,

er wordt een gebruikersnaam en wachtwoord verwacht om toegang te verkrijgen tot de database. Met welke gegevens er ingelogd wordt op de database, kan in de applicatie veranderd worden. Voor de Dynfos applicatie op Windows wordt er ingelogd met een standaard database gebruiker. Deze standaard gebruiker heeft standaard alle rechten op alle data die er zich bevindt in de database.

Om data af te schermen voor mobiele apparaten, kan er een user aangemaakt worden die specifiek gebruikt kan worden voor de mobiele applicatie. Deze user kan dan aparte rechten gegeven worden, zodat deze niet bij bepaalde data kan.

3.4.1.1 Schema's

Een schema is een collectie van database objecten. De collectie kan voorzien worden van rechten. Als deze collectie eenmaal een keer is aangemaakt kan dit schema toegepast worden op meerdere users[15] . Dit voorkomt het herhaaldelijke proces van rechten toekennen. Bij het gebruik van een andere user is het mogelijk om rechten af te wijzen en toe te zeggen. Aangezien Dynfos erg veel data heeft zijn dat veel regels om op te stellen. Dit is veel werk om één keer te doen, maar als dit vaker moet gebeuren is dit erg onhandig. In het geval dat het vaker moet gebeuren kan het handig zijn om een schema op te stellen.

3.4.2 Views

Naast het gebruik van andere users om data af te schermen kan er ook gebruik worden gemaakt van views.

Een view is een set van resultaten van een opgeslagen query. Deze query is van tevoren opgesteld op de data die gekozen is. Net zoals een tabel kan een view gewoon aangeroepen worden. Deze view kan bijvoorbeeld resultaten van verschillende tabellen bevatten die zijn samengevoegd. Dit is de reden waarom views ook wel eens virtuele tabellen worden genoemd[14] . Een view is erg handig, maar een complexe view kan alleen gebruikt worden voor het ophalen van data.

Views kunnen data opvragen van de database die belangrijk is voor de mobiele applicatie zonder de rest van de data te laten zien of open te stellen. Dit zorgt ervoor dat de database veiliger is dan zonder het gebruik van views.

Naast dat een view voordeel behaald op de veiligheid zijn er nog andere voordelen die erg handig zijn voor dit project, zoals:

- Simpel om data op te halen van meerdere sources
- Gebruikt geen extra ruimte
- Vermindering van onnodige data

Data die normaal uit meerdere sources gehaald moet worden, kan nu opgehaald worden door één view. De view gebruikt geen extra ruimte, omdat het alleen een referentie naar een query is. In de database van Dynfos staat veel data. Het kan zo zijn dat de Dynfos Windows applicatie een productenlijst heeft, waarvan elk product circa 60 velden met informatie over dat product heeft. De mobiele applicatie zal niet al deze informatie nodig hebben. Met het gebruik van views is het mogelijk om een paar velden van elk product op te vragen.

In andere situaties zou een database view, performance heavy kunnen zijn, aangezien de query van de view elke keer opnieuw uitgevoerd zal moeten worden. In het geval van dit project zal het juist beter voor de performance zijn, aangezien er veel minder dataverkeer is wat verwerkt moet worden. Eén nadeel van het gebruik van views is het schrijven naar de database. Het schrijven naar de database via een view is niet mogelijk. Om toch te kunnen schrijven naar de database, zullen er custom SQL-statements gestuurd worden naar de database.

3.4.3 Procedures

[16] [17] Een procedure is een set van SQL-statements. De procedure is opgeslagen in de database. De procedures kunnen de data aanpassen, maar de procedures zijn niet specifiek aan één database gebonden. Dit heeft een aantal voordelen:

- Veiligheid & Data integriteit
- Productiviteit
- Netwerkactiviteit

Het gebruik van procedures is veiliger, omdat de gebruiker zelf geen veranderingen kan maken aan procedures. De gebruikers kunnen wel variabelen instellen, maar wat er met die variabelen gebeurd is van tevoren al vast gesteld. De productiviteit is verbeterd door het hergebruik van procedures in meerdere databases. De SQL-statements hoeven maar 1 keer gemaakt te worden in procedures. Doordat een procedure meerdere SQL-statements kan bevatten hoeft de procedure maar één keer aangeroepen te worden door de code om meerdere statements uit te voeren, dit scheelt een aantal calls naar de database.

3.4.4 Conclusie

De methoden en technieken die in dit hoofdstuk beschreven zijn kunnen erg goed gebruikt worden voor verschillende doeleinden. Het gebruik van users komt nagenoeg zonder nadelen. Het enige nadeel is dat het complexiteit toevoegt aan het beheer van de database. Dit weegt echter niet op tegen de voordelen die het biedt, het kan dan ook erg goed benut worden om de veiligheid van de database te verbeteren. Als optionele toevoeging kan er ook gebruik gemaakt worden van schema's. Dit zal voor dit project geen voordeel behalen, maar als er in de toekomst met andere Users toegevoegd wordt dan kan dit erg handig zijn.

Het gebruik van Views brengt veelal voordelen met zich mee. Views kunnen dus erg goed gebruikt worden voor dit project.

De procedures hebben geen nadelen dus kunnen zo ook gebruikt worden voor het project. De voordelen van het gebruik van procedures zullen ook effectief zijn voor het project aangezien ze voor de veiligheid of performance van de applicatie goed zijn.

Het gebruik van users en views zullen erg veel bijdragen aan het project, deze zullen dan ook gebruikt worden in de implementatie. De procedures kunnen ook gebruikt worden, dit zal handig zijn omdat het niet mogelijk is om te schrijven naar een database via een view.

Deze verschillende maatregelen werken goed samen. De views en procedures zullen worden gebruikt voor het manipuleren en ophalen van data. De toegang tot de tabellen kan ontzegd worden door de database user. Dit is een goede manier om het gebruik van views en procedures te forceren. Het resultaat hiervan is gelimiteerde toegang tot de data.

3.5 Deelvraag 3: Authenticatie methode

Deelvraag 3 luidt: Wat is een methode van authenticatie die herbruikbaar is voor verschillende apparaten van een klant?

[1] Om de gebruikers toegang te geven tot de database moet de gebruiker zich authentiseren. Er zijn veel verschillende methoden voor authentiseren. In dit hoofdstuk wordt bekeken wat de beste manier van authentiseren is voor de gebruikers van Dynfos.

Natuurlijk staat de effectiviteit van de beveiliging vooraan, maar gedurende het proces kan het de gebruikers wel zo gemakkelijk mogelijk gemaakt worden voor om in te loggen. Deze applicatie zal op verschillende scanners geïnstalleerd worden, daarom zal er rekening moeten worden gehouden met de gegevens die gebruikt gaan worden gedurende de authenticatie fase.

3.5.1 Authenticatie factoren

Er zijn een aantal factoren die gebruikt kunnen worden voor het authentiseren van een gebruiker:

1. Knowledge factor
2. Possession factor
3. Inherence factor

3.5.1.1 Knowledge factor

Deze factor houdt in dat een gebruiker iets weet, zoals zijn gebruikersnaam en wachtwoord. Dit kan onhandig zijn, omdat de applicatie gebruikt moet kunnen worden door iedereen die een scanapparaat bediend en er veel scanapparaten zullen zijn. Wel zou het een optie kunnen zijn om iedereen het gebruikersnaam en wachtwoord te geven.

3.5.1.2 Possession factor

Deze factor houdt in dat de gebruiker alleen een fysiek object heeft om zich te authentiseren, zoiets als een smart card, security token of een fysieke sleutel.

Deze factor zou wel geschikt zijn voor een klant aangezien er bijvoorbeeld een fysieke sleutel kan worden meegegeven/opgehangen ergens rond het bedrijf.

3.5.1.3 Inherence factor

Deze factor is alleen voor persoonlijk gebruik van een service, dit komt doordat deze factor gebruik maakt van biometrie, zoals vingerafdrukken, iris scanners of voice activation.

Dit zou eventueel gebruikt kunnen worden, echter is het dan wat moeilijker om scanners te vinden die één van deze functies ook daadwerkelijk kan uitvoeren.

3.5.2 Voor- en nadelen

Elk van deze factoren heeft zijn voor- en nadelen. Om een keuze te maken tussen de verschillende factoren worden die voor en nadelen tegen elkaar afgewogen. Van de verschillende factoren weegt de veiligheid het zwaarst mee, aangezien alleen de gebruikers van de app zich moeten kunnen authentiseren.

3.5.2.1 Veiligheid

Het gebruik van een gebruikersnaam en wachtwoord kan veilig zijn, maar de veiligheid ligt aan eisen die eraan worden gesteld. Zo kan er worden gekozen om speciale tekens te gebruiken in wachtwoorden of/en een minimale lengte voor het wachtwoord te eisen. Hoe veilig deze factor is ligt voor het grootste gedeelte bij de gebruiker, aangezien deze de inloggegevens aanmaakt en ook beheert.

De possession factor is erg veilig in gebruik. Een fysiek voorwerp kan verloren of gestolen worden, maar het is wel duidelijk als deze niet meer in het bezit is van de gebruiker, in tegenstelling tot alleen de inloggegevens.

Het gebruik van de inherence factor is ook erg veilig. Het gebruik van biometrie is persoonsgebonden en kan dus niet door andere personen geactiveerd worden. Dit zorgt ervoor dat het niet mogelijk is de veiligheid bij een gebruiker zelf neer te leggen.

3.5.2.2 Onderhoudbaarheid

Het onderhouden van een set inloggegevens vereist geen nieuwe methoden voor Dynfos. De inloggegevens zouden in de database opgeslagen kunnen worden. De verschillende inloggegevens kunnen verwijderd en aangepast worden als een gebruiker zijn wachtwoord bijvoorbeeld vergeet.

Het onderhoud van een fysiek voorwerp is veel lastiger. Er moet worden bijgehouden welke voorwerpen er in de omloop zijn en het distribueren is lastiger dan een set inloggegevens. Ook zal er bij het kwijtraken van een voorwerp een nieuw voorwerp gegeven moeten worden. Hier zijn natuurlijk ook kosten verbonden. Ook zal de sleutel moeten kunnen werken op de mobiele apparaten die worden gebruikt. Het fysieke voorwerp kan ook verloren worden of kan breken.

Het onderhoud van biometrie is lastiger dan bij de voorgenoemde methoden. Ten eerste moeten er faciliteiten zijn om deze manier van authentiseren mogelijk te maken, zoals een camera of finger-print scanner. Deze faciliteiten kunnen kapot gaan, waardoor de applicatie niet meer gebruikt kan worden. Deze faciliteiten kunnen duur zijn om te vervangen. Als tweede punt is het andere data die opgeslagen moet worden in de database. Dit zal niet zo gemakkelijk zijn als het gebruik van een fysiek voorwerp of inloggegevens.

3.5.2.3 Gebruiksgemak

De gebruiker hoeft alleen wat te typen om zich in te loggen, het aantal handelingen om dit te doen, is beperkt. Een nadeel is echter wel dat de gebruiker zijn inloggegevens kan vergeten.

Het gebruik van een fysiek voorwerp is in gebruik simpel. De gebruiker moet echter wel onthouden het voorwerp mee te nemen als dit gebruikt moet worden.

Het gebruik van biometrie is voor de users niet veel werk. Hoe makkelijk het precies is, ligt aan de faciliteiten waarmee de authenticatie verloopt. Het kan namelijk zijn dat er verscheidene keren gescand moet worden om de gebruiker te herkennen, dit kan het gebruiksgemak beïnvloeden op een negatieve manier.

3.5.3 Conclusie

De applicatie kan gebruikmaken van verschillende methoden voor authenticatie. Er is gekeken wat de voor-en nadelen van de verschillende methoden van authenticatie zijn. Hieronder zullen de plus en minpunten te zien zijn van de verschillende methoden, deze zijn gebaseerd op de uitleg hierboven.

Tabel 7, Authenticatiefactor keuzetabel

	Veiligheid	***Onderhoudbaarheid***	***Gebruiksgemak***
Knowledge factor	+/-	+	+/-
Possession factor	+	-	+
Inherence factor	+	-	+

Het gebruiksgemak van alle criteria is goed. Het is ook mogelijk om gebruiksgemak en onderhoud op te geven voor het gebruik van Multi-factor authenticatie. In dit geval kunnen er twee van deze factoren gebruikt worden in plaats van één. Dit kan tot gevolg hebben dat de veiligheid beter wordt. Voor dit project is het onderhoud en gebruiksgemak ook belangrijk, dus kan het een voor of nadeel zijn om een tweede authenticatie factor te gebruiken.

De andere twee criteria verschillen wel per authenticatie factor. Van de twee verschillende criteria is veiligheid wel belangrijker dan onderhoud, maar het onderhoud brengt ook kosten met zich mee. De veiligheid van de Knowledge factor ligt aan de eisen die aan de inloggegevens gesteld worden, dit kan dus veiliger gemaakt worden ten koste van gebruiksgemak.

Als het gaat om onderhoud is er een erg groot verschil tussen de verschillende factoren. De Knowledge factor is minder goed in het gebruikersgemak, omdat de gebruikers hun gegevens moeten onthouden. Dit is niet zo met de andere factoren. De knowledge factor is goed te onderhouden omdat inloggegevens makkelijk zijn op te slaan in de database. Er is geen onderhoud nodig voor fysieke objecten. Dit geldt ook voor faciliteiten om aan de inherence factor te voldoen. Het gebruik van een gebruikersnaam en wachtwoord lijkt dan ook de beste optie te zijn voor dit project.

3.6 Deelvraag 4: Systeem toegang

Deelvraag 4: Hoe kan de applicatie toegang verlenen tot een systeem aan de klant zonder toegang te geven tot systemen van andere klanten?

De gebruikers van de applicatie willen alle informatie waar de applicatie over beschikt kunnen zien. Alleen de data die voor de gebruiker bestemd is mag te zien zijn.

Om in te loggen moet er gecheckt worden of de inloggegevens in de database staan. Als er ingelogd wordt is het niet de bedoeling dat alle gebruikers van verschillende administraties in één database staan. Het gevolg hiervan is dat het mogelijk is om bij een andere administraties in te kunnen loggen als je de juiste inloggegevens van iemand hebt. Daarnaast is het niet mogelijk om dubbele namen in de database te beheren. Dubbele namen kunnen wel voorkomen als alle administraties de gebruikers in één tabel hebben staan. Het zal in deze situatie goed te zien zijn welke andere administraties in de database staan. Dynfos wil dit graag privé en gescheiden houden.

Eén oplossing voor dit probleem is het opslaan van inloggegevens bij de administratie zelf, en niet centraal op één plaats. Deze oplossing is veiliger dan alle informatie bij elkaar op slaan, omdat de data zo gescheiden is. Dit brengt echter wel een probleem met zich mee. Als een gebruiker nu inlogt, weet de applicatie niet waar de inloggegevens gecheckt moeten worden. Voor dit probleem moet een oplossing gezocht worden. De oplossing zal op twee criteria beoordeeld worden, Veiligheid en gebruiksgemak. Van deze twee criteria weegt veiligheid zwaarder mee, omdat alleen de juiste gebruiker zich mag authentiseren.

3.6.1 Oplossing

3.6.1.1 Extra inloggegeven

Een mogelijke oplossing voor dit probleem is het gebruik van een extra inlogveld. In dit veld kan een code worden gezet die overeenkomt met de administratie van de gebruiker. Dit zorgt ervoor dat de applicatie precies weet welke administratie er gebruikt gaat worden en waar de gegevens van de gebruiker gecheckt moeten worden.

Het gebruik van een code is niet positief voor het gebruiksgemak, het is namelijk weer een extra gegeven die onthouden moet worden. Voor de veiligheid van de gegevens is het wel beter als ze bij hun eigen administratie bijgehouden worden. Zo zal het niet mogelijk zijn dat een administratie gegevens van andere administraties kunnen zien.

3.6.1.2 QR-codes

Een andere oplossing is het gebruik van QR-codes. De QR-codes kunnen gebruikt worden gebruikt om een code te scannen. De mobiele apparaten die gebruikt worden, moeten al kunnen scannen om de voorraad te manipuleren, dus zal het scannen van een QR-code geen probleem zijn. Het is makkelijk om QR-codes uit te delen, dit lijkt een goede manier om een bedrijfscode te integreren.

De QR-codes kunnen worden uitgedeeld aan alle medewerkers die de scanners gaan gebruiken, maar ze kunnen ook opgehangen worden op de plekken waar de scanners liggen om het inloggen gemakkelijk te maken.

[2] Een QR-code heeft wel een limiet aan data die erop gezet kan worden. Er bestaan verschillende QR-codes, er zijn 40 verschillende versies van de QR-code. Elke volgende versie van een QR-code beginnend van versie 1 kan meer data vasthouden. De grootste versie van de

QR-code heeft een grootte van 177 x 177 vierkanten, dit resulteert in 31.329 plekken om data op te slaan. Dat aantal is genoeg voor 3KB aan data. Dit is genoeg voor een bedrijfscode, ook als deze versleuteld is. De bedrijfscode zou versleuteld kunnen worden om de QR-code alleen leesbaar te maken voor de applicatie.

QR-codes zouden onveilig kunnen zijn als het gaat om linkjes die naar sites wijzen. Die zouden vervalst kunnen worden door QR-codes te vervangen met andere QR-codes die naar andere sites werken, dit kan natuurlijk niet gezien worden door mensen. Dit probleem zal voor dit project geen bedreiging vormen aangezien de data die in de QR-code staat in de applicatie afgehandeld zal worden. Als de inhoud van de QR-code niet overeenkomt met de data die nodig is, dan kan dit afgehandeld worden.

Een QR-code zorgt ervoor dat er een fysiek/digitaal voorwerp moet worden gescand. Dit zorgt voor een extra laag beveiliging. Dit is voor het gebruiksgemak niet echt bevorderlijk. Een QR-code kan makkelijk verspreid worden, dit is nadelig als het gedeeld wordt met anderen die niet bij de administratie horen waar de QR-code voor is gemaakt.

3.6.2 Conclusie

De voor-en nadelen van beide oplossingen zullen hier in een tabel te vinden zijn.

Tabel 8, Authenticatie parameter keuzetabel

	Veiligheid	***Gebruiksgemak***
Bedrijfscode	**+/-**	**+/-**
QR-code	**+**	**-**

De veiligheid voor beide oplossingen is beter dan alle gebruikers in een samengevoegde lijst. Het is dus zeker dat één van de methoden gebruikt gaat worden. De beide oplossingen zijn gebaseerd op hetzelfde idee, namelijk een bedrijfscode. Dan is de vraag of het gebruik van een QR-code genoeg toevoegt om daar voor te kiezen.

De veiligheid van de applicatie is erg belangrijk dus wordt daar meer waarde aan gehecht. Het gebruiksgemak is natuurlijk ook belangrijk, maar in dit geval wordt de veiligheid belangrijker gevonden dan gebruiksgemak. Het gebruik van een QR-code lijkt dan ook het beste te zijn voor dit project.

3.7 Deelvraag 5: Applicatie distributie

Deelvraag 5: Wat is een manier waarop de applicatie kan worden gedistribueerd, waarbij regelmatige updates een belangrijk aspect is?

De applicatie die gemaakt zal worden zal voor klanten goed en makkelijk beschikbaar moeten zijn voor het gemak van de klant. De applicatie moet naast het goed beschikbaar zijn, ook

makkelijk te zijn in het publiceren, er moeten bijvoorbeeld niet te veel randvoorwaarden zitten aan het uploaden van een applicatie om te voorkomen dat het publiceren van een nieuwe versie veel tijd gaat kosten.

De applicatie die in Blazor wordt gemaakt hoeft niet te worden gedistribueerd, omdat dit gehost wordt op een website die bezocht kan worden. De oude methode voor het maken van een mobiele applicatie wordt ook gehost op een website om het gebruik van een API mogelijk te maken, maar wordt door middel van PhoneGap omgezet naar een applicatie. Dit kan voor zowel Android als iOS.

3.7.1 Distributie platform

[4] [5] In 2019 zijn er 194 miljard mobiele apps gedownload over het hele jaar, 105 miljard van die apps zijn gedownload via de iOS App Store en de Google Play Store, dit is 54% van de totale hoeveelheid applicaties die gedownload zijn. Het is dus logisch minstens één van deze twee app stores te kiezen om de applicatie op te publiceren.

Om te weten wat de randvoorwaarden zijn voor het uploaden van een applicatie op de twee verschillende app stores, wordt er gekeken naar het proces van uploaden. Het proces van het uploaden naar de Play Store is zeker makkelijker dan het uploaden van de applicatie naar de App Store. Voor het uploaden naar de Play Store is een account nodig. Dynfos heeft al een account voor de Play Store, dus die randvoorwaarden zijn al behaald.

De randvoorwaarden die de App Store heeft zijn uitgebreider dan die van de Play Store. Zo moet er onder andere een account gegeven worden, zodat ze de applicatie kunnen testen op de voorwaarden. Dit wil Dynfos natuurlijk niet, omdat dit toegang verleent tot de data die Dynfos bewaard.

3.7.2 Conclusie

Voor de applicatie die gemaakt wordt met behulp van DevExpress Blazor, is er geen distributie platform voor mobiele apparaten die gebruikt kan worden. Dit wordt op het web gehost.

Voor de mobiele applicatie kan er wel een applicatie gedistribueerd worden op verscheidene platformen. PhoneGap stelt ertoe in staat om in samenwerking met DevExpress een applicatie te genereren die op de Apple App Store en de Google Play Store geplaatst kan worden. Het is makkelijker om de applicatie te publiceren op de Play Store, maar het is wel mogelijk om de applicatie op de App Store te publiceren.

Er wordt aangeraden om de applicatie op de Play store te publiceren, om de data van de applicatie zo veilig mogelijk te houden. Het kan echter wel zo zijn dat de klanten geen Android apparaten hebben. In dit geval kan Dynfos er voor kiezen om een speciale set met inloggegevens te maken voor Apple. Deze kan verwezen worden naar test data, zodat Apple geen toegang heeft tot de echte data. Dynfos moet uiteindelijk beslissen of de vraag voor de iOS applicatie het werk rechtvaardigt om dit te publiceren op de Apple Store.

4. Requirements

Het probleem dat beschreven is moet worden opgelost. Om tot een goede oplossing te komen waar Dynfos tevreden mee kan zijn, is er een lijstje van requirements op te stellen die de wensen van Dynfos goed uit laten komen. Deze lijst met requirements is tevens een leidraad tijdens de implementatie van het product.

De requirements die hier beschreven worden zijn 'SMART' opgesteld zijn. Dit houdt in dat de requirements Specifiek, Meetbaar, Acceptabel, Realiseerbaar en Tijdgebonden zijn. Daarnaast zijn de requirements geprioriteerd doormiddel van de MoSCoW methode. De MoSCoW methode heeft vier verschillende prioriteitseenheden.

M – must haves: Deze requirements moeten in het product komen.

S – should haves: Dit zijn requirements die wel gewild zijn, maar ze zijn niet nodig voor het eindproduct.

C – could haves: Dit zijn requirements die behandeld kunnen worden als er tijd over is.

W – won't haves: Dit zijn requirements voor de toekomst die niet worden behandeld tijdens dit project.

4.1 Functionele requirements

Functionele requirements zijn requirements die een functie van de applicatie uitleggen.

4.1.1 Must

1. De applicatie geeft de gebruikers toegang op basis van hun gebruikersnaam, wachtwoord en bedrijfscode.
2. De applicatie verbindt met de database van de gebruiker, aan de hand van de meegegeven inloggegevens.
3. De applicatie stelt de gebruiker ertoe in staat om een lijst van producten te bekijken.

4.1.2 Should

4. De applicatie stelt de gebruiker ertoe in staat om details van één specifiek product te bekijken.
5. De applicatie stelt de gebruiker ertoe in staat om de kwantiteit van een product aan te passen.
6. De applicatie stelt de gebruiker ertoe in staat om een mailverzoek te sturen over de urenverantwoording van de gebruiker.

4.1.3 Could

7. De applicatie stelt de gebruiker ertoe in staat om werktijd te registreren.
8. De applicatie ondersteunt de gebruikers met het invullen van de tijdregistratie.
9. De applicatie stelt de gebruiker ertoe in staat om een bestaande tijdregistratie aan te passen.
10. De applicatie kan QR-codes scannen om acties uit te voeren voor het toevoegen en aanpassen van producten en tijdregistratie.
11. De applicatie onthoudt de gegevens van de gebruiker als deze eenmaal heeft ingelogd.
12. De applicatie stelt de gebruiker ertoe in staat om producten op te zoeken met gebruik van een productnaam.

13. De applicatie stelt de gebruiker ertoe in staat om de locatie van een product aan te passen.

4.1.4 Won't

14. De applicatie stelt de gebruiker ertoe in staat om een product aan te maken door een naam en aantal in te vullen.
15. De applicatie kan de gebruiker uitloggen, zodat de gegevens van de gebruiker opnieuw ingevuld moeten worden om toegang te krijgen tot de applicatie.
16. De applicatie stelt de gebruiker ertoe in staat om zijn wachtwoord te veranderen.

4.2 Niet functionele requirements

Niet functionele requirements zijn requirements die iets zeggen over de kwaliteit van de applicatie.

4.2.1 Must

1. De applicatie heeft een verbinding met de database die minder onderhoud nodig heeft dan de huidige situatie.
2. De verbinding met de database zal gelimiteerd worden tot gebruikers van de applicatie.
3. De applicatie heeft alleen toegang tot tabellen en views voor de tijdregistraties en producten.

4.2.2 Should

4. De applicatie zal de producten ophalen van de database en laten zien in de applicatie binnen 3 seconden.

4.2.3 Could

5. De gebruiker moet in één klik de applicatie kunnen downloaden en installeren.
6. De gebruiker moet in twee klikken kunnen uitloggen.

4.2.4 Won't

7. De applicatie start op binnen 3 seconden.

5. Functioneel ontwerp

Dit hoofdstuk zal laten zien wat het ontwerp is van de applicatie en wat de redenen zijn voor het ontwerp. Naast het ontwerp van de applicatie zal ook het ontwerp van de database verbinding besproken worden.

5.1 Context diagram

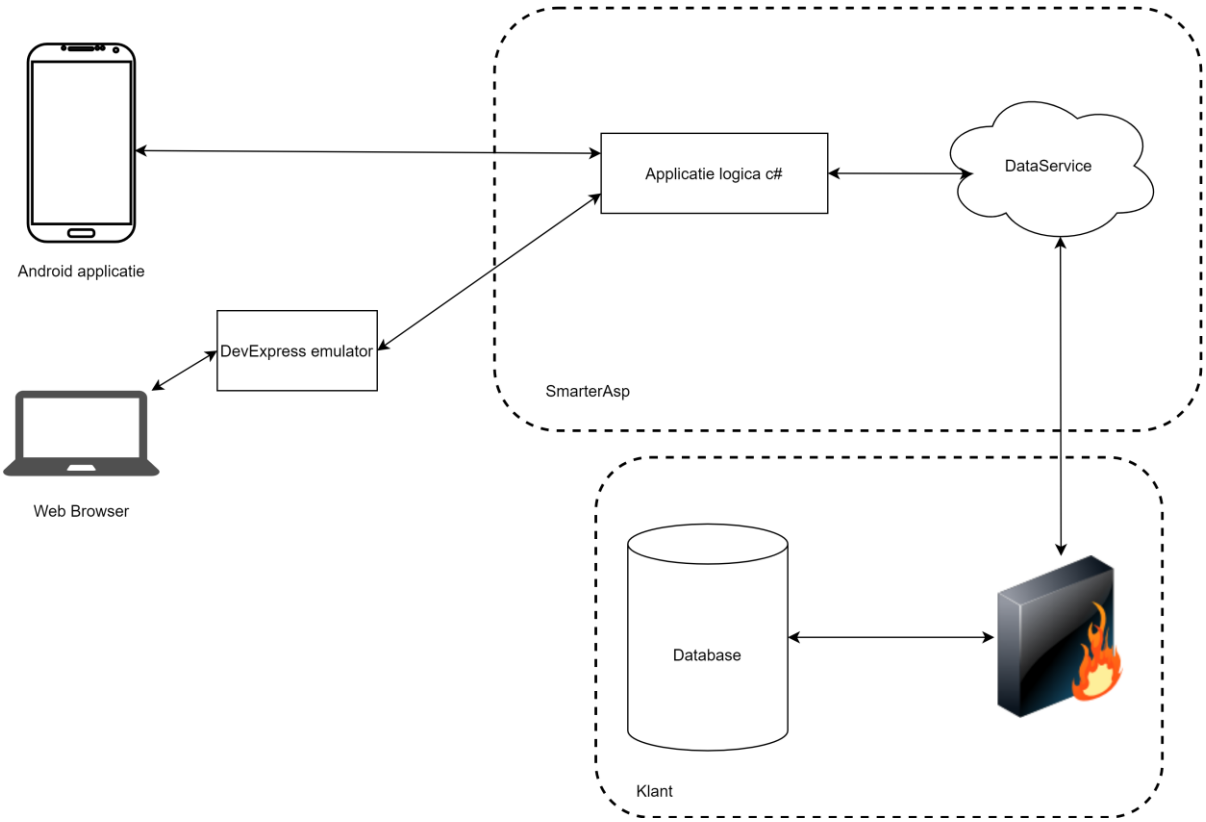
De applicatie wordt gemaakt voor gebruikers van het Dynfos systeem. Niet alle gebruikers van het Dynfos systeem zullen ook de Dynfos App gebruiken. De applicatie is een ondersteuning voor de gebruikers die geen toegang hebben tot de Windows versie van het Dynfos systeem, maar toch Dynfos nodig hebben tijdens het werk. Door het gebruik van een context diagram zal het duidelijk worden wat de gebruiker kan doen met de applicatie en wat voor informatie de applicatie de gebruiker geeft.

In het **Context diagram** is te zien dat de applicatie is gemaakt voor twee soorten gebruikers. De normale gebruikers zijn veel belangrijker dan de administrator, omdat het niet nodig is om de rol van de administrator in de applicatie te gebruiken. De administrator is als ondersteuning gemaakt voor het toevoegen en toekennen van rollen, dit kan ook rechtstreeks vanuit de database gedaan worden, maar is niet gewenst.

De applicatie geeft de gebruikers een lijst met producten en tijdregistraties. Deze lijsten zullen de data laten zien die voor de gebruikers belangrijk is, zoals een naam, ID of hoeveelheid. In deze lijsten kunnen afzonderlijke items bekeken en aangepast kunnen worden door de gebruikers.

5.2 Architectuur diagram

Om een beter beeld te krijgen van de werking van de applicatie kan een architectuur diagram handig zijn. Een architectuur diagram helpt met het visualiseren van de benodigdheden om de applicatie te gebruiken. Het diagram zal een high level structuur van de applicatie en het systeem eromheen beschrijven.



Figuur 6, Architectuur diagram

In het diagram is er te zien dat er twee manieren zijn om de applicatie te benaderen, de gewenste methode is een verbinding met de mobiele applicatie. Het is ook mogelijk om de applicatie via een online emulator te benaderen al is dit niet het uiteindelijke gebruiksdoel.

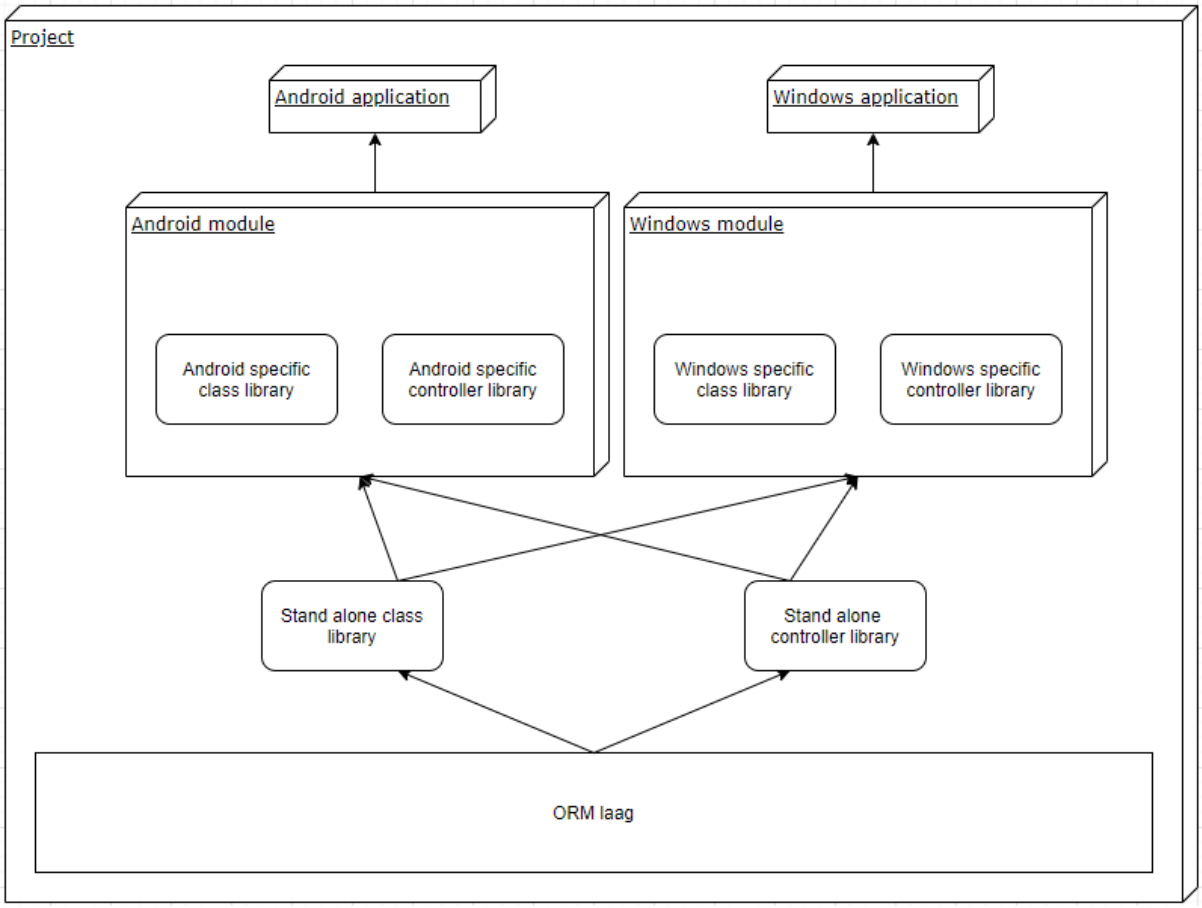
De applicatie zal verbinden met een dataservice die samen met de applicatie zelf gehost wordt op SmarterASP. SmarterASP is een hosting site voor ASP.NET applicaties. ASP.NET is een server-side web applicatie framework om dynamische web pagina's te gebruiken. Het Microsoft .NET framework kan gebruik maken van ASP.NET.

[12] Deze dataservice zal een open API(in dit geval OData) bevatten die het mogelijk maakt om de applicatie logica te verbinden met de database. De dataservice kan niet aangepast worden, omdat DevExpress de verbinding met de dataservice opzet. Daarnaast is DevExpress verantwoordelijk voor het aanroepen van de API aan de hand van de code die in de applicatie logica staat. Als er iets veranderd moet worden aan de connectie of de dataservice zal dit gedaan moeten worden door de code van het project.

De API in de vorm van de dataservice zal verbinding maken met de database. In het diagram is er een gedeelte te zien dat bij de klant hoort. Hoe dit ingedeeld wordt kan door iedere klant verschillend zijn ingedeeld, zo kan een klant ervoor kiezen om geen firewall te gebruiken. Voor de invulling van het gedeelte 'klant', is de indeling van Dynfos gebruikt. Zoals te zien is, zal de verbinding met de database eerst door een firewall gaan die Dynfos beheerd.

5.3 Projectstructuur

In het onderzoek is er onder andere gekeken naar de structuur van het Dynfos project. Nu is het de bedoeling om deze mobiele applicatie toe te voegen aan het Dynfos project. De architectuur van het project zal dus ook veranderen. Dit hoofdstuk zal die nieuwe architectuur bespreken. Hieronder is de nieuwe architectuur te zien:



Figuur 8, Nieuwe structuur voor het project

De structuur van de nieuwe situatie heeft inspiratie opgedaan van de oude structuur. De structuur begint bij de ORM laag die alle code zal gaan omzetten naar acties voor de database. De stand-alone modules zijn nog aanwezig, dit is gedaan om het project één geheel te maken. Deze modules zorgen ervoor dat de classes en data die ze moeten kunnen beheren maar één keer voorkomen. De platform specifieke modules hebben classes en controllers die niet in de stand-alone modules kunnen, omdat ze niet compatibel zijn met beide applicatie types.

5.4 Database architectuur

De database van Dynfos zal worden gebruikt voor de mobiele applicatie. Deze database staat intern bij Dynfos en stelt de gebruikers van Dynfos ertoe in staat om via de Dynfos App, die zij gebruiken, een verbinding te maken met de Dynfos database.

Er zijn echter wat redenen waarom de database structuur die Dynfos op dit moment gebruikt nog wat aangepast kan worden. In het onderzoek is besproken hoe de database verbinding verbeterd kan worden op het gebied van veiligheid en onderhoudbaarheid.

5.4.1 Oude database architectuur Dynfos App

De oude database flow van de Dynfos App staat beschreven in hoofdstuk 3.3.1. Deze architectuur is niet geschikt voor de nieuwe applicatie, dit is de reden dat er een nieuwe database architectuur gemaakt is.

5.4.2 Nieuwe database architectuur

Het is belangrijk dat de nieuwe architectuur niet dezelfde problemen heeft als de vorige architectuur, maar de nieuwe database moet ook veilig zijn. Snelheid, onderhoudbaarheid, veiligheid en integratie met DevExpress worden allemaal meegenomen in de nieuwe architectuur. De Nieuwe database architectuur is te vinden in de bijlagen.

Zodra de applicatie opgestart is kan de gebruiker zijn bedrijfscode, gebruikersnaam en wachtwoord invoeren. De gebruiker kan dan ingelogd worden op de voor de gebruiker bestemde database.

De applicatie zal als eerst kijken of de bedrijfscode die de gebruiker gegeven heeft geldig is. Dit wordt gedaan door te kijken of er een record van de bedrijfscode in de database staat die bedoeld is voor het omleiden naar de andere databases.

Niet alle data die in de database staat mag verstrekt worden aan de gebruiker. De database zal worden gelimiteerd om alleen tabellen en views te laten zien die vereist zijn voor de gebruiker, zoals producten en tijdregistraties. De verbinding met de database zal in het geval van Dynfos, door een firewall gaan om ervoor te zorgen dat de productiedatabase veilig is. Deze verbinding moet worden toegelaten door de firewall. Hoe de configuratie er bij de klant uitziet is niet bekend, maar een firewall wordt aangeraden. De applicatie zal op de database inloggen onder een speciale user die helpt met het limiteren van de verschillende soorten data.

Dit systeem zorgt ervoor dat de gebruiker van de mobiele applicatie niet de toegang heeft tot tabellen en data die niet nodig zijn voor het uitvoeren van de taken die de gebruiker moet doen. Er zal ook een extra laag beveiliging zijn door het de database user die gebruikt wordt door de productiedatabase.

5.4.3 Ontwerp van de code

Om dit ontwerp te realiseren moet er tijdens het uitvoeren van de applicatie van database veranderd moeten worden. Het is belangrijk om dit te doen voordat de gebruiker geauthentiseerd wordt. De authenticatie gebeurt namelijk bij de database waar de gebruiker toe behoort, dit is gedaan zodat er geen dubbele namen van verschillende administraties kunnen voorkomen. Daarnaast zal het niet mogelijk zijn om het account van een andere gebruiker te gebruiken aangezien er nu nog een extra bedrijfscode is, waar rekening mee moet worden gehouden. Het veranderen van database zal wel moeten gebeuren na het bekijken van de bedrijfscode. Met deze twee eisen in gedachte, kan er dus worden

geconcludeerd dat het veranderen van database zal moeten gebeuren tijdens het authentiseren.

Het veranderen van database zal gedaan worden door de connectie string die gebruikt wordt om verbinding te maken met een database, dynamisch aan te passen. De applicatie zal dan de authenticatie afmaken met een verbinding die is aangepast.

6. Implementatie

Dit hoofdstuk zal beschrijven wat er allemaal gedaan is voor de implementatie van de applicatie. Het hoofdstuk zal beschrijven tegen welke problemen er aangelopen is, hoe deze zijn opgelost en het uiteindelijke product.

6.1 XAF Mobile

Nadat er geprobeerd was de applicatie te implementeren in Blazor kon er geconcludeerd worden dat het gebruik van Blazor, nog niet haalbaar is met de huidige status van Blazor. Als Blazor officieel wordt uitgegeven met alle functionaliteiten die DevExpress gepland heeft, zal Blazor gebruikt kunnen worden om de applicatie te maken.

De keuze is dus gemaakt om de applicatie te gaan implementeren met de oude mobiele methode die DevExpress aanbiedt.

6.1.1 Custom authenticatie

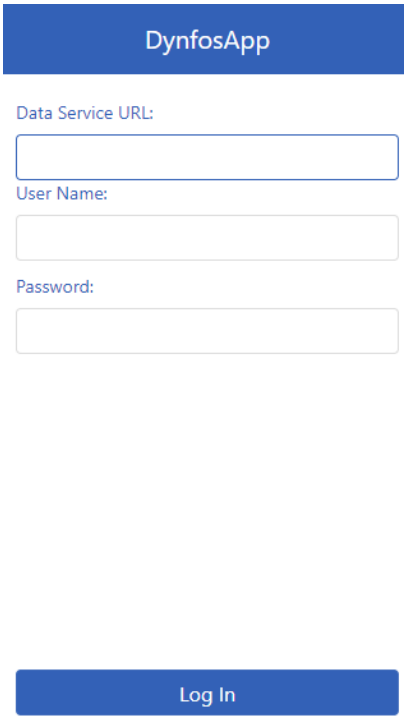
De gebruiker moet aan de hand van zijn/haar inloggegevens kunnen inloggen bij de administratie van het bedrijf waar zij toe behoren.

DevExpress heeft een standaard authenticatie module die een standaard authenticatie kan uitvoeren. Dit is een authenticatie methode die een gebruikersnaam en wachtwoord checkt met gegevens uit een database.

Het is niet mogelijk om een extra parameter zoals een bedrijfscode toe te voegen aan de standaard authenticatie parameters. Om dit toch te realiseren, is er een nieuwe class gemaakt die de standaard parameters class van DevExpress gebruikt en daarop uitbreidt met de extra bedrijfscode parameter.

Naast het gebruik van een custom authenticatie parameter class, is er ook een class gemaakt voor het authentiseren van de gebruiker. Deze class is een uitbreiding van de standaard authenticatie class. Deze custom authenticatie class maakt gebruik van de custom authenticatie parameter class om de gebruiker te authentiseren.

Om in te loggen zullen de gebruikers het volgende scherm te zien krijgen:



Figuur 9, Inlogscherm app

Het authenticatie proces heeft echter wel zijn problemen opgeleverd. Het authentiseren was erg lastig aangezien er een bug in de XAF mobile zit als er custom authenticatie gebruikt wordt, waardoor de applicatie meerdere keren geïnitieerd wordt. Als de applicatie probeert in te loggen, zal dit meerdere keren geprobeerd worden. Het resultaat is dat er meerdere authenticatie pogingen gedaan worden met lege inloggegevens. Dit probleem was uiteindelijk niet te verhelpen, maar er is geprobeerd dit zo goed mogelijk te verminderen voor de gebruiker. Tijdens de authenticatie wordt er gekeken of de applicatie lege inloggegevens gestuurd krijgt. De applicatie zal niet laten weten dat er een poging tot inloggen is gedaan als de gegevens leeg zijn. Dit heeft als gevolg dat de gebruiker geen bericht krijgt als de inloggegevens niet zijn ingevuld. Dit kan nadelig zijn als de gebruiker probeert in te loggen zonder gegevens in te vullen.

6.1.2 Database verbinding

Tijdens het maken van het ontwerp van de database architectuur, is er een kleine test applicatie gemaakt die de functionaliteit van het ontwerp heeft gevalideerd. De functie die getest werd, is de functie die de gebruiker naar de goede administratie verwijst aan de hand van de inloggegevens van de gebruiker. Deze test is gemaakt met behulp van XAF, maar er is ervoor gekozen om dit met de standaard Windows module te doen. Op dat moment leek het een simpelere methode om de functionaliteit te testen. De applicatie werkte en de functionaliteit van het ontwerp leek voldoende.

Tijdens de implementatie van de opdracht werd hetzelfde concept dat de test applicatie gebruikt ook geïmplementeerd in de officiële mobiele applicatie. Toen de implementatie hiervan gecheckt werd, bleek de applicatie niet te werken.

Het proces om uit te vinden wat er precies verkeerd was gegaan met de implementatie was lang en lastig. Na veel communicatie met het DevExpress team is er gebleken dat er niet getest was of de implementatie van de functionaliteit ook werkt op de mobiele applicatie. DevExpress heeft uiteindelijk geconcludeerd dat de implementatie van het ontwerp voor de database verbinding niet mogelijk is, dus moet er een nieuw ontwerp gemaakt worden, die wel te implementeren is.

6.1.2.1 Nieuw ontwerp database verbinding

DevExpress heeft bevestigd dat het oude ontwerp van de data verbinding niet te realiseren is. Dit is de reden dat er besloten om een andere oplossing te ontwerpen voor het verbinden met verschillende databases gedurende het uitvoeren van de applicatie.

De **Nieuwe database architectuur V2** zal in de bijlagen te vinden zijn.

Het nieuwe ontwerp zal erg anders werken dan het vorige ontwerp. Dit ontwerp maakt gebruik van meerdere applicaties en dataservices om de connectie naar de verschillende databases waar te maken.

Er zal één applicatie zijn die de gebruiker krijgt. Deze applicatie werkt precies hetzelfde voor de gebruiker als bij het vorige ontwerp, alleen zal de applicatie nu anders omgaan met de bedrijfscode die de gebruiker opgeeft.

De applicatie zal aan de hand van de bedrijfscode een URL uitkiezen. Deze URL verwijst naar een kopie applicatie, die behalve de connectie string niet veranderd is. Deze applicatie heeft een dataservice, deze dataservice heeft een verbinding met de database die al van te voren is ingesteld.

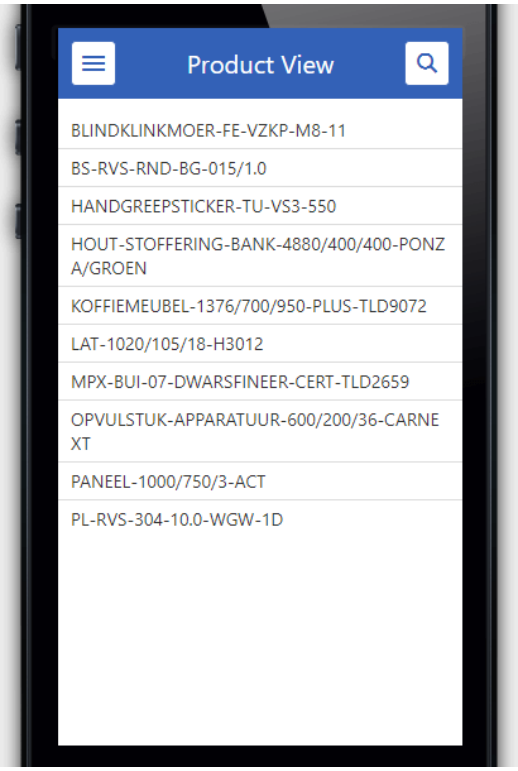
Het resultaat is een applicatie, die terwijl het uitgevoerd wordt, kan wisselen van dataservice. De applicatie kan zo verbinden met de database die de gebruiker nodig heeft.

Het nadeel van dit ontwerp is dat er voor elke administratie er een applicatie moet worden gepubliceerd. Dit heeft als gevolg dat de configuratie van de klant van tevoren moet worden geïmplementeerd in de applicatie. Na dit te hebben gedaan zal er een nieuwe applicatie worden gepubliceerd. Pas dan kan er vanaf de applicatie verwezen worden naar de DataService voor de nieuwe klant.

6.1.3 Producten

De applicatie laat producten zien die in de database staan. Het tabel producten bestaat uit erg veel velden met informatie. Niet alle informatie die er bekend is over een product, is belangrijk voor de gebruiker. Wat wel belangrijk is voor de gebruiker is een naam, ID, eigenaar en een hoeveelheid. Om alleen deze informatie uit de tabellen te krijgen is er een database view gemaakt. Deze view in combinatie met het gebruik van een database user die geen rechten heeft op tabellen, zorgt ervoor dat alleen de benodigde informatie te krijgen is. Ook is er in de code uitgeschakeld dat er nieuwe producten aangemaakt kunnen worden. Er is een

optie om te zoeken naar producten. Deze zoekfunctie zal handig zijn als de applicatie vol staat met producten. De productenlijst ziet er als volgt uit:



Figuur 10, Productenlijst

De details van een product kunnen bekeken worden. Als de details van het product opgevraagd worden, zullen de overige velden van de view worden laten zien. Dit is een voorbeeld van een product:

Figuur 11, Product details

De producten die zijn opgenomen in de applicatie komen uit de productiedatabase. Om de informatie over de producten te limiteren is er gebruik gemaakt van views. De view voor de producten ziet er als volgt uit:

SELECT Id, CmpIdOwner, NameShort, TechnicalStock FROM dbo.Products ORDER BY Id DESC

Het producten tabel heeft in totaal 84 kolommen. De kolom 'Id' wordt gebruikt, omdat dit de unieke sleutel is voor een product. De 'NameShort' en 'TechnicalStock' kolommen worden gebruikt om te zien hoeveel er van een specifiek product aanwezig is in de administratie. Dit is ook belangrijk voor het veranderen van dat aantal. De kolom CmpIdOwner is toegevoegd, zodat de klant kan zien bij welke administratie het product in kwestie hoort. Dit is cruciaal voor klanten met meerdere administraties.

6.1.4 Mailverzoek

Bij de urenverantwoording van de gebruiker zal er per week een overzicht te zien zijn van de uren die zijn geklokt. De gebruiker kan het overzicht in de app bekijken, maar kan er ook voor kiezen deze te mailen naar zijn/haar mailadres. De app zal een regel toevoegen aan een tabel voor het verzoeken van mailtjes. Dynfos gebruikt software om regels uit deze tabel uit te lezen en uit te voeren.

6.1.5 Publicatie

Het publiceren van de applicatie is anders dan bij een native mobiele applicatie. De mobiele applicatie is gebaseerd op ASP.NET. Om ASP.NET uit te kunnen voeren hebben we een webbrowser nodig.

Om de applicatie via de telefoon te kunnen openen moet de applicatie gedraaid worden op een server, zodat de applicatie op de telefoon deze altijd kan bereiken.

De server die Dynfos gebruikt is een server van SmarterASP. Op deze site kan je applicaties publiceren. Om dit te doen moet de applicatie met de publiceer optie in Visual Studio gepubliceerd worden naar een folder. De inhoud van deze folder kan met behulp van FTP overgezet worden naar de SmarterASP server.

Toen de bestanden eenmaal op de server stonden, kon de URL gebruikt worden die SmarterASP geleverd heeft, om de applicatie te starten in een emulator.

6.1.5.1 Problemen

Tijdens het publiceren van de applicatie zijn er veel problemen geweest, die opgelost moesten worden. De oorzaken van de problemen waren niet makkelijk te achterhalen.

Het eerste probleem waartegen aan werd gelopen was dat de applicatie aangaf dat er een bestand niet geladen kon worden. Na gekeken te hebben of de gepubliceerde goed zijn overgekomen naar de server en dat de benodigde rechten aan de bestanden zijn gegeven, leek het dat het probleem door iets anders zou zijn veroorzaakt.

De applicatie werkte wel als deze lokaal werd gestart in de ontwikkelomgeving. Doordat de applicatie lokaal werkte moest het liggen aan de publicatie van de applicatie. De eerste gedachte was dat DevExpress misschien wel wist wat er aan de hand was, aangezien de foutmelding vanuit het DevExpress framework werd gestuurd. Na in contact te zijn geweest met DevExpress, waren er wat opties die bekeken zouden kunnen worden. Alle suggesties die

DevExpress deed, konden alleen toegepast worden op de server waar de applicatie draait. Na contact te hebben gehad met DevExpress is er contact opgenomen met SmarterASP om de suggesties van DevExpress voor te stellen. Het bleek dat alle suggesties die DevExpress gedaan hadden, al waren ingesteld op de SmarterASP server, dus moest er verder worden gekeken naar het probleem. Na alle stappen voor het publiceren van de applicatie een paar keer te hebben doorlopen was het probleem gevonden. Niet alle benodigde DevExpress bestanden werden gepubliceerd met de applicatie. Dit resulteerde in missende files.

Het tweede probleem dat tegen gekomen is, had te maken met het openen van de database. De eerste fase van het probleem was dat de database niet gevonden kon worden. Na wat overleg over de oorzaak van deze foutmelding, leek het dat de firewall van Dynfos connecties van buiten stopt. Om dit op te lossen zijn er gesprekken geweest met het netwerkbeheer van Dynfos. Tijdens de gesprekken zijn er wat methoden besproken om de verbinding veiliger te maken. Het doel was vooral om het IP waarop de database staat niet rechtstreeks open te zetten. De keuze is gemaakt om een NAT te gebruiken. Een Network Address Translation kan gebruikt worden om het echte IP-adres van Dynfos onbekend te houden. De connectie naar de database kan nu via één IP en poort combinatie te verkrijgen zijn. Ook zal het niet mogelijk zijn om direct te verbinden met het IP van de database server.

De applicatie had hierna nog een foutmelding, maar deze foutmelding was veranderd nadat de firewall opengezet was. Om uit te vinden of de foutmelding niet door de firewall veroorzaakt werd, is er getest of de applicatie wel zou werken op een externe computer. De applicatie werkte naar behoren, dus is er contact opgenomen met SmarterASP om uit te zoeken waarom er geen verbinding met de database gemaakt kan maken vanuit SmarterASP. Uit dit gesprek bleek dat het IP en poort dat gebruikt wordt om met de database te verbinden, ge-whitelist moest worden op SmarterASP.

De databases van de klanten zullen op eigen locatie plaatsvinden. Hierdoor zal het niet altijd mogelijk zijn om de oplossing die bij Dynfos gekozen is, ook bij de klant toe te passen. De klant zal een eigen verantwoording hebben om de connectie naar de database mogelijk te maken, maar er zullen wel wat adviezen gegeven worden om deze verbinding veilig te maken.

Er zal nog een vergadering plaatsvinden met het netwerkbeheer om te bespreken wat de mogelijkheden zijn om de verbinding met de database, device gebonden te maken. Dit zal niet geïmplementeerd kunnen worden in het huidige tijdsbestek, maar deze informatie kan goed van pas komen voor de toekomst van deze applicatie.

Het laatste probleem gaat over het veranderen van dataservice. Dit werkt wel als de applicatie lokaal is gepubliceerd, maar dit werkt nog niet als de applicatie gepubliceerd is op SmarterASP. Om dit probleem te verhelpen is er contact met DevExpress en SmarterASP gezocht. Om te kunnen switchen van dataservice moet de applicatie meerdere keren gepubliceerd worden. Door dit op twee URL's te doen zal dit probleem zich voordoen. Eén van de mogelijke oplossing hiervoor zou zijn, om de verschillende applicaties op één URL te publiceren. Na het gesprek met SmarterASP was het duidelijk dat dit ingesteld kan worden met een remote IIS(Internet Information Services) manager. De IIS manager wordt al gebruikt om lokaal de applicatie te publiceren, maar om dit remote te doen is er een extensie nodig. Deze extensie zou door Microsoft geleverd worden, maar is sinds september 2020 niet

beschikbaar. De reden dat deze extensie niet beschikbaar is, is niet bekend. Dit probleem is nog niet opgelost.

De problemen die opgelost zijn tijdens het publiceren van de applicatie waren niet makkelijk op te lossen. Deze concept applicatie laat zien waartegen aan gelopen zal worden als de echte applicatie gepubliceerd moet worden. Dit geldt niet alleen voor de huidige mobiele methode die gebruikt wordt, maar ook voor het publiceren van een Blazor applicatie. Aangezien de Blazor applicatie ook een web applicatie is, zal dit ook voor de toekomst bekend zijn.

7. Conclusie

De implementatie van de applicatie was niet zonder problemen. De meeste problemen die zijn tegengekomen duurden lang om op te lossen. Om de problemen op te lossen is er gebruik gemaakt van expertise van Dynfos, DevExpress en SmarterASP.

Om een conclusie te trekken van dit project, kan de implementatie van de applicatie vergeleken worden met de requirements die gesteld zijn door Dynfos.

7.1 Requirements gehaald

De requirements die gesteld zijn aan het project geven aan wat de verwachte voortgang is voor de proof of concept applicatie. Om een goed beeld te krijgen over de status van de applicatie kan er gekeken worden naar de requirements. Welke requirements gehaald zijn, staat in het bijgeleverde bestande genaamd “Conclusie requirements”. De requirements die gehaald zijn, zijn van hoge prioriteit. Er zijn ook wat requirements die niet gehaald zijn, maar deze zijn van lagere prioriteit. Veel van deze requirements betreffen het verzenden van data naar de database, omdat deze handelingen minder belangrijk zijn voor Dynfos. Er kan geconcludeerd worden dat de implementatie van de applicatie is geslaagd. De gemaakte applicatie zou in theorie in productie kunnen gaan. In praktijk zijn er wat dingen die aangepast zouden kunnen worden. Een voorbeeld hiervan is het gebruik van een bedrijfscode. Deze bedrijfscode wordt nu nog met een URL aangeduid. Hoewel de functionaliteit er wel is, is het niet bevorderlijk voor het gebruiksgemak. Daarnaast zal elke klant die de applicatie wil gebruiken, een eigen configuratie moeten aangeven. Dit geldt bijvoorbeeld voor de URL die naar de productiedatabase wijst.

7.2 Kwaliteit

De kwaliteit van het product is gewaarborgd door de lokale tests die zijn gedaan. Alle taken zijn getest op een lokale database die over een set van oude data beschikt. Deze oude data komt vanuit een oudere versie van de productiedatabase.

Tijdens het testen wordt er gekeken of het gedrag van de data uit de database doet wat ervan verwacht wordt. Voorbeelden hiervan zijn het inloggen en het ophalen van producten. Als de data in de database veranderd worden, zal dit dan ook direct te zien zijn als er een bepaalde actie wordt uitgevoerd op de applicatie?

Als er nieuwe features aan de applicatie toegevoegd worden, moet het mogelijk zijn om de functies die al getest zijn, op dezelfde manier te testen als dat het de eerste keer gedaan is. Dit zorgt ervoor dat het makkelijk te zien is, als er functionaliteiten anders gaan reageren als dat ze bij de initiële test gedaan hebben.

De tests die uitgevoerd zijn, moeten dus herhaalbaar zijn. Om de tests herhaalbaar op te stellen, is er gebruik gemaakt van test cases. Deze test cases zullen voor elke functionaliteit beschrijven hoe er getest is en met welke data dat gedaan is. Hieronder volgt een voorbeeld waar de test cases worden laten zien. Deze test cases beschrijven hoe de gebruiker kan inloggen.

Tabel 9, Voorbeeld test cases

Test Case	TC-001	TC-002
Test Titel	Valideer het inloggen van de applicatie	Valideer het inloggen van de applicatie
Test Stappen	1. Open de applicatie 2. Voer de inloggegevens in 3. Klik op inloggen 4. Valideer of de inhoud van de app te zien is	1. Open de applicatie 2. Voer de inloggegevens in 3. Klik op inloggen 4. Valideer of de inhoud van de app te zien is
Test Data	DataService URL: http://localhost:2065/DataService.svc/ User name: djansen Password: djansen	DataService URL: http://localhost:2065/DataService.svc/ User name: djansen Password: *******
Verwachte Resultaten	De gebruiker zal niet geauthentiseerd worden, omdat het wachtwoord verkeerd is	De gebruiker zal geauthentiseerd worden
Resultaat	De gebruiker is niet geauthentiseerd	De gebruiker is geauthentiseerd
Status	Pass	Pass

Het volledige bestand met test cases is toegevoegd.

De kwaliteit van de code kan bekeken worden aan de hand van de code standaarden die Dynfos heeft. Dit geldt niet alleen voor de code maar ook de database. Alle tabellen in de database zullen meervoud zijn. De business objecten die in de code staan die gelinkt zijn aan een tabel zullen dezelfde naam hebben, maar dan enkelvoud. De namen van variabelen en hun properties zijn ook gelinkt aan de kolomnamen van een tabel.

Om deze code standaarden beter te handhaven, wordt er binnen Dynfos gebruik gemaakt van een software genaamd QAP. Dynfos heeft deze software zelf ontwikkeld. QAP staat voor Quality Assurance Program. De software wordt gebruikt om de code die gemaakt is, om te zetten naar de standaard die Dynfos aanhoudt. Alles is instelbaar, waardoor het een handige tool is voor alle classes, maar het is vooral gemaakt voor business objecten, zoals producten en tijdregistraties. Deze software verzekert code die voldoet aan de eisen die Dynfos stelt. Deze eisen zijn configureerbaar in de QAP software. De QAP software is ook gebruikt voor dit project. De code van het project zal dus ook voldoen aan QAP eisen.

8. Proces

Gedurende het project is er gewerkt volgens de Agile methode Kanban. Er is gekozen om met Kanban te werken in plaats van andere iteratieve methoden, zoals Scrum. Bij Scrum zijn er vaste bijeenkomsten die moeten worden gehouden. Aangezien dit project wordt uitgevoerd door één persoon is dit niet handig.

Dat er gewerkt is met Kanban houdt in dat er sprints zijn gehouden met hun bijbehorende taken. De sprints die gebruikt zijn voor dit project zijn ieder een week lang. Er zijn in totaal 8 sprints gepland voor de implementatie van de applicatie. Bij elke sprint worden de taken bijgehouden op een Trello bord. Dit bord heeft meerdere categorieën waar een taak zich onder kan bevinden. De verschillende categorieën die gebruikt worden zijn:

- Wensen
- Bugs
- Ontwikkeling
- Testen
- Productie/goedgekeurd
- Afgekeurd
- Vragen

In de categorie Wensen staan de taken die op de lijst staan om uitgevoerd te worden. De categorie Bugs verzameld alle bugs die zijn gevonden tijdens het implementeren. De derde categorie Ontwikkeling, wordt gebruikt voor taken die op dat moment worden geïmplementeerd. De categorie Testen wordt gebruikt als de implementatie van een taak af is, want dan wordt er gekeken of de implementatie werkt in de applicatie. De categorie voor Productie/goedgekeurd geeft aan wanneer taken succesvol getest zijn. Als taken niet succesvol getest zijn zullen de taken in de categorie Afgekeurd gezet worden. De laatste categorie die gebruikt wordt is de categorie voor vragen. Deze categorie laat taken zien waar nog vragen over zijn of vragen die niet met specifieke taken te maken hebben.

Door de voortgang in sprints bij te houden was het makkelijker te zien of de implementatie op schema lag of niet.

8.1 Afspraken

Tijdens het project is er gecommuniceerd met de opdrachtgever en de afstudeerbegeleider. Elke week is er een afspraak gepland met de opdrachtgever. Tijdens deze afspraak werd besproken wat er de afgelopen week is gedaan en hoe de volgende week ingepland staat. De werkzaamheden die besproken werden in deze afspraken gingen voornamelijk over de inhoud van de applicatie.

Naast dat er elke week een afspraak met de opdrachtgever was gepland, was er ook een wekelijkse afspraak met de afstudeerbegeleider. Tijdens deze afspraken zijn de werkzaamheden van de afgelopen week besproken. De plannen van de volgende week werden ook benoemd. Naast dat te hebben besproken, werd er ook feedback over bestanden zoals het verslag besproken.

9. Aanbevelingen

Het doel van dit hoofdstuk is om het werk dat gedaan is aan het onderzoek en de implementatie van de applicatie, niet achter te laten zonder houvast voor de volgende medewerker die op het project moet voortborduren.

Op basis van de resultaten van zowel het onderzoek, de implementatie zullen er aanbevelingen worden gedaan. Als eerst zullen er aanbevelingen worden gedaan op het onderzoek de aanbevelingen op de implementatie volgen daarop.

9.1 Blazor onderzoek

De eerste aanbeveling die ik heb is het onderzoek naar Blazor. De introductie van Blazor kwam op een laat tijdstip tijdens het project, waardoor er niet echt grondig onderzoek heeft kunnen plaatsvinden naar Blazor. Als er een onderzoek naar Blazor wordt gehouden, kan er een betere beslissing genomen worden over de toekomst van Blazor in samenhang met Dynfos.

9.2 Blazor releases

De Blazor release die nu getest is, is een bèta versie die DevExpress uitgebracht heeft als test. Als Blazor de vervanger wordt van de mobiele oplossing die DevExpress nu aanbiedt, wordt de oude methode niet meer ondersteund. De bèta versie is nu nog niet compatibel met het Dynfos project, omdat het Dynfos project te complex is voor de minimalistische Blazor support die ze nu hebben. Zou Blazor langzamerhand meer functies krijgt van het DevExpress team, kan het oude systeem vervangen worden met Blazor, hopelijk zonder aanpassingen.

9.3 Implementatie applicatie

De applicatie die is gemaakt, is een proof of concept. De applicatie zal niet over genoeg functies bezitten om de oude applicatie volledig te vervangen. Een aanbeveling is om deze applicatie af te maken. Het is echter wel de vraag of het afmaken van de applicatie efficiënt is. Met de opkomst van DevExpress Blazor integratie, kan het mogelijk zijn dat het meer werk kost om het project van de oude mobiele methode te migreren naar XAF Blazor.

9.4 Verbinding met de database

De verbinding met de database moet veilig zijn. De gesprekken die hebben plaatsgevonden met het netwerkbeheer van Dynfos zijn van goede hulp geweest om een veilige verbinding te maken voor een connectie met de Dynfos database. Het kan echter zo zijn dat klanten geen firewall of andere maatregelen hebben om deze verbinding veilig te maken. Een aanbeveling hiervoor is om adviezen te maken voor de connectie met de database. Zo is de klant geïnformeerd dat de verbinding veiliger kan zijn als de klant de benodigde stappen zet.

Van de aanbevelingen die ik gedaan heb, zijn de aanbevelingen die met het onderzoek te maken hebben het belangrijkst op de korte termijn. Het bijhouden van de Blazor releases is een proces dat niet zal stoppen. De applicatie veranderen van een proof of concept naar een echte volledige applicatie is veel werk. De urgentie van de implementatie van de nieuwe applicatie zal beslissen of het handig is om eerst te wachten op de volledige Blazor support.

10. Reflectie

De reflectie is gemaakt via de STARR methode.

10.1 Situatie

De afstudeerperiode is voltooid bij Dynfos. Er is onderzoek gedaan naar methoden die informatie aan de klant kan laten zien op een veilige wijze. Dit onderzoek is ook in de praktijk gebracht door de implementatie van een applicatie. De tijd die ik bij Dynfos heb mogen besteden tijdens mijn afstudeerproject was erg fijn. Door de openheid en de goede sfeer die bij Dynfos te vinden is, was het communiceren erg makkelijk. Dit was makkelijk ondanks dat communiceren niet mijn sterkste punt is.

Iedereen bij Dynfos had interesse voor het afstudeerproject, dit was te merken tijdens de werkoverleggen die er waren. Bij elk werk overleg werd ik meegenomen in het proces, net zoals alle andere werknemers.

Als ik problemen had kon ik bij mijn medewerkers terecht. Als het iets was met de code dan kon dhr. Gortemaker mij vaak helpen. Als ik administrator rechten nodig had voor het installeren van een programma of iets anders op de computer, dan kon ik dat gewoon even vragen en werd het meteen gedaan.

Toen COVID-19 ervoor zorgde dat er vanuit huis gewerkt moest worden heb ik regelmatig met dhr. Gortemaker aan de telefoon gezeten om voortgang te bespreken, hulp te krijgen enzovoorts. Dit heeft zeker geholpen in de lastige omstandigheden waarin wij zaten.

10.2 Taak

De taak was het onderzoek en de implementatie van de applicatie. Tijdens het uitvoeren van deze taak is er meermaals een afspraak ingepland met de opdrachtgever en afstudeerbegeleider. De voortgang van deze taak werd dan besproken. Het verslag is ook besproken tijdens deze afspraken. Aan het einde van het project werd dit regelmatig gedaan. Het was beter geweest om eerder inlever momenten aan te vragen, zodat ik eerder in het traject veranderingen aan mijn verslag had kunnen maken. Dit had mij waarschijnlijk veel tijd gescheeld, aangezien ik dan eerder had opgemerkt dat ik een verkeerde richting opging met het verslag.

10.3 Actie

De uitvoering van het onderzoek was niet goed. De eerste versie van het onderzoek was niet echt een onderzoek, maar meer een handleiding. Dit heeft ervoor gezorgd dat er veel uitloop was van de opdracht. De tweede versie van het onderzoek is beter gelukt. Achteraf was het beter geweest als het onderzoek was uitgevoerd met criteria die objectiever zijn dan de huidige criteria.

Na de opdracht gekregen te hebben, wist ik eigenlijk niet goed wat DevExpress inhield. Vooral het maken van de UI van een mobiele applicatie vind ik altijd lastig. Het was bijzonder te horen dat er niet heel veel UI werk gedaan moest worden om een werkende mobiele applicatie te krijgen, aangezien ik nog nooit een mobiele applicatie heb gemaakt zonder de volledige UI te ontwerpen. Na wat onderzoek naar DevExpress had ik een hele andere kijk op

de opdracht. Het werken met een framework die zo drastisch het maken van een applicatie verandert, heb ik nog nooit gedaan. Het was dus echt een opdracht waar ik veel van heb geleerd.

Het implementeren van de mobiele applicatie was toch moeilijker dan verwacht. Er waren ongelofelijk veel foutmeldingen, waarvan een paar zelfs niet lieten zien waar ze vandaan kwamen. Dit was toch wel een groot nadeel aan het XAF framework. Bij het plannen van het project had ik hier wel wat meer rekening mee mogen houden.

Het publiceren van de mobiele applicatie heeft veel moeite gekost. Er komt toch meer kijken bij het publiceren van een .NET web applicatie die ook op de mobiele telefoon geïnstalleerd kan worden, dan een native applicatie te publiceren.

10.4 Resultaat

De resultaten van het project kunnen goed van pas komen voor de toekomst. De kennis van het onderzoek kan goed gebruikt worden voor het afmaken van de mobiele applicatie, maar ook voor de implementatie van XAF Blazor.

De opdracht is uiteindelijk redelijk ver gevorderd. De meeste belangrijke requirements zijn behaald. Er was echter wel één should requirement die niet is gehaald: ‘De applicatie stelt de gebruiker ertoe in staat om de kwantiteit van een product aan te passen.’. Dit is uiteindelijk niet afgemaakt. De theorie van deze requirement is wel terug te vinden in het mailverzoek die een gebruiker kan doen. Deze zal ook data naar de database sturen. Ik ga er dan ook vanuit dat deze requirement geen nieuwe technieken vereist.

10.5 Reflectie

Ik ben niet geheel tevreden met het uiteindelijke onderzoek, omdat deze toch minder objectief was dan eerst gedacht. Wel ben ik erg tevreden over de verbetering van dit onderzoek ten opzichte van de eerste versie van het onderzoek. Ik heb van de onderzoeksperiode veel geleerd, zowel voor het maken van een onderzoek als de onderzochte vragen.

Met de applicatie ben ik tevreden, omdat alle technieken die in de applicatie moeten zitten aanwezig zijn. Er zijn wel wat dingen die aangepast moeten worden, zoals het gebruik van een echte bedrijfscode in plaats van een URL. Het is fijn om te zien dat de applicatie werkt ondanks alle problemen die zijn tegengekomen tijdens de implementatie fase.

11. Literatuurlijst

[1] Rouse, M. (2014, 1 december). *Authentication factor*. Geraadpleegd op 19 maart 2020, van https://searchsecurity.techtarget.com/definition/authentication-factor

[2] *QR Codes - How Much Data Can A QR Code Store?* (z.d.). Geraadpleegd op 20 maart 2020, van http://qrcode.meetheed.com/question7.php?s=s

[3] *Add a Mobile Application Project | eXpressApp Framework | DevExpress Documentation*. (z.d.). Geraadpleegd op 24 maart 2020, van https://docs.devexpress.com/eXpressAppFramework/116362/getting-started/xaf-mobile-maintenance-mode-tutorial/add-a-mobile-application-project?v=18.2

[4] Iqbal, M. (2019, 19 november). *App Download and Usage Statistics (2019)*. Geraadpleegd op 25 maart 2020, van https://www.businessofapps.com/data/app-statistics/

[5] *App Store Review Guidelines - Apple Developer*. (z.d.-b). Geraadpleegd op 27 maart 2020, van https://developer.apple.com/app-store/review/guidelines/

[6] *XAF Architecture | eXpressApp Framework | DevExpress Documentation*. (z.d.). Geraadpleegd op 21 april 2020, van https://docs.devexpress.com/eXpressAppFramework/112559/fundamentals/xaf-architecture

[7] OBSOLETE - XAF Mobile UI (Maintenance Mode) - XAF ASP.NET Core Blazor UI Is Coming. (2020, 21 juli). DevExpress. https://supportcenter.devexpress.com/ticket/details/t356939/obsolete-xaf-mobile-ui-maintenance-mode-xaf-asp-net-core-blazor-ui-is-coming

[8] Rouse, M. (2020, 7 april). RESTful API (REST API). SearchAppArchitecture. https://searchapparchitecture.techtarget.com/definition/RESTful-API

[9] Higginbotham, J. (2017, 7 maart). Five Tips for Managing Database Servers in Production. dzone.com. https://dzone.com/articles/5-tips-for-managing-database-servers-in-production

[10] Chapman, C. (2019, 2 augustus). Why you shouldn't connect your mobile application to a database. Chapman World. https://chapmanworld.com/2015/07/02/why-you-shouldnt-connect-your-mobile-application-to-a-database/

[11] F, B. (2020, 4 maart). How to Connect an Android App to a MySQL Database. SkySilk Cloud Blog. https://www.skysilk.com/blog/2018/how-to-connect-an-android-app-to-a-mysql-database/

[12] WCF Data Services. (2013, 5 maart). Microsoft Docs. https://docs.microsoft.com/en-us/previous-versions/cc668792(v=vs.103)?redirectedfrom=MSDN

[13] Ribunal, M. (2018, 15 januari). How To Limit User Access To Your Database. SQL, Code, Coffee, Etc. https://marlonribunal.com/limiting-user-access-to-your-database/

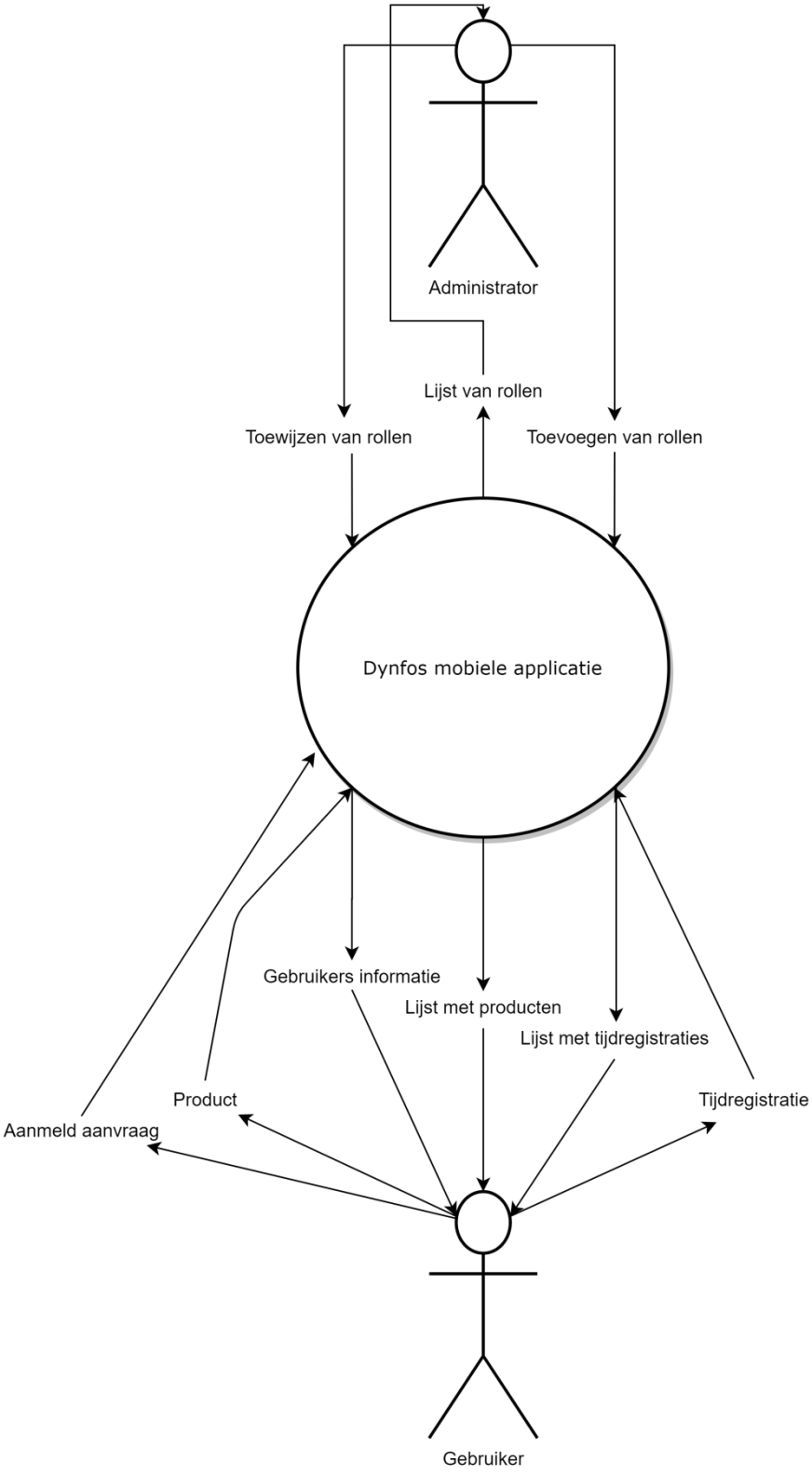
[14] What is a relational database view. (z.d.). essentialsql. https://www.essentialsql.com/what-is-a-relational-database-view/

[15] Jones, A. D. (z.d.). What Is a Schema? | Managing Database Objects in SQL | InformIT. https://www.informit.com/articles/article.aspx?p=1216889&seqNum=2

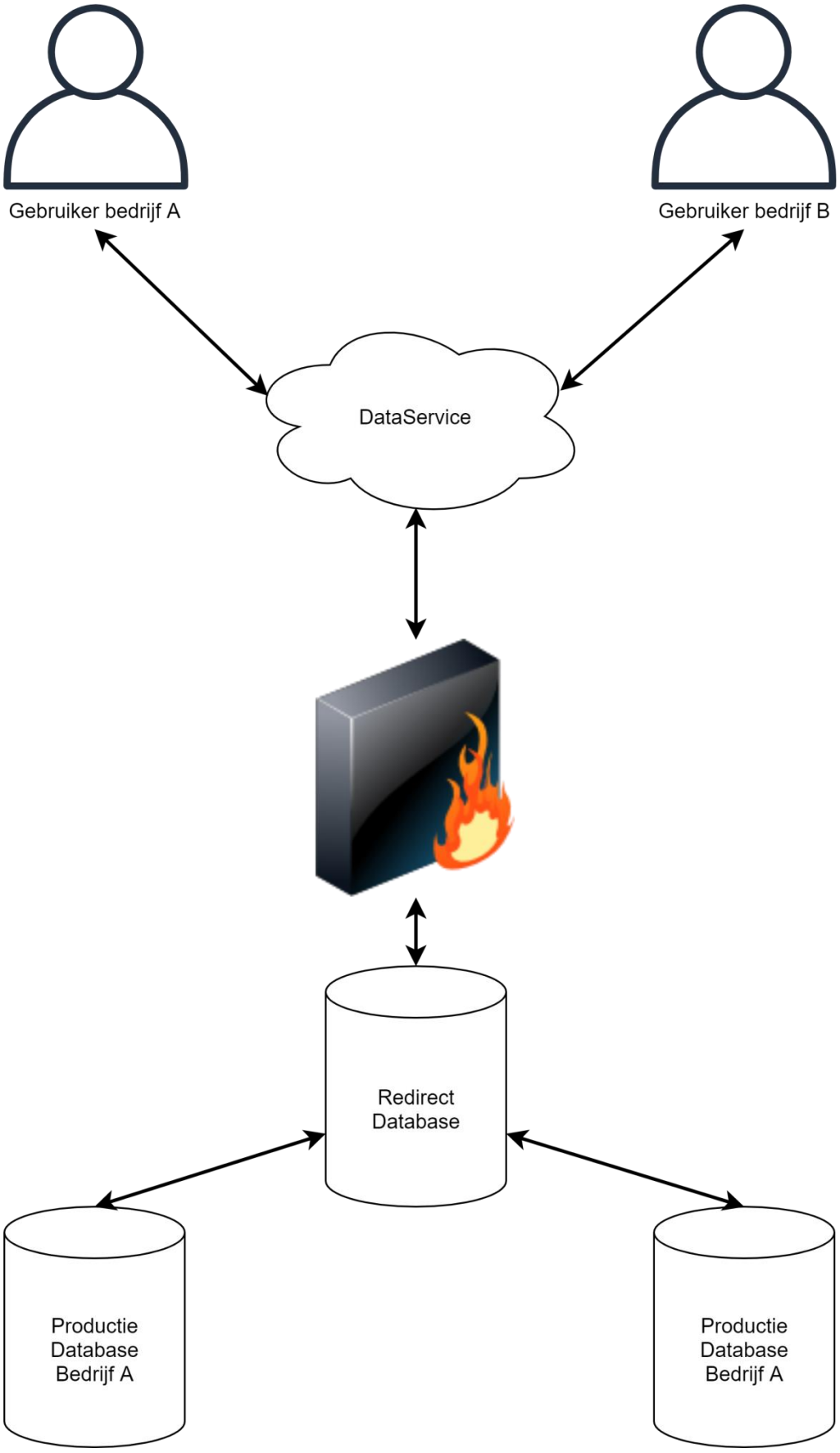
[16] SQL Stored Procedures. (z.d.). W3Schools. https://www.w3schools.com/sql/sql_stored_procedures.asp

[17] Rouse, M. (2019, 8 april). stored procedure. SearchOracle. https://searchoracle.techtarget.com/definition/stored-procedure

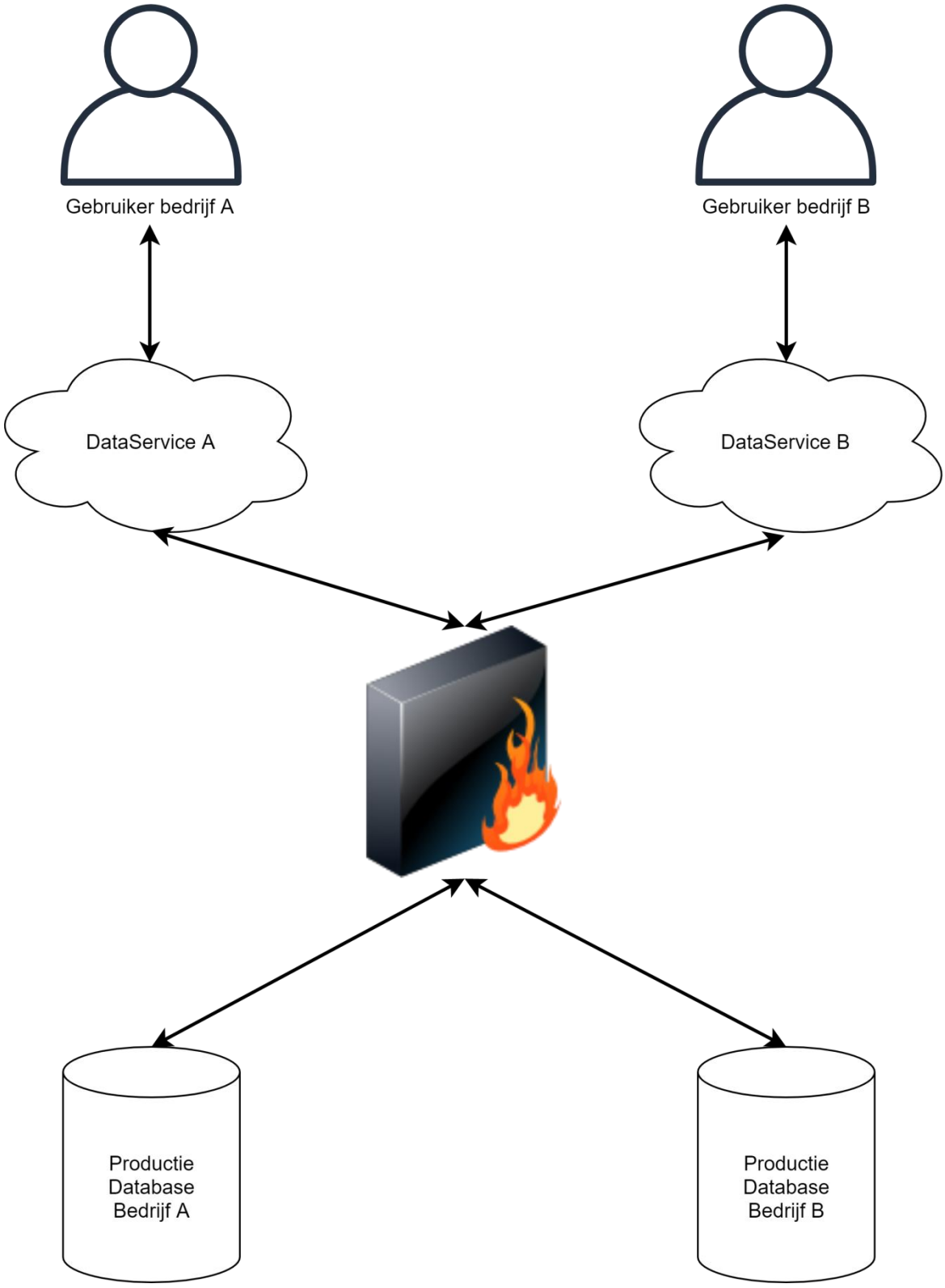
12. Bijlagen



Figuur 12, Context diagram



Figuur 13, Nieuwe database architectuur



Figuur 14, Nieuwe database architectuur V2