HOGESCHOOL PXL

Document 3: Sjabloon rapport

(OPM: haal deze titel weg bij het definitieve rapport)

Herschrijven glasbewegingen scanners

(je noteert hier het onderwerp van je Graduaatsproef. Zorg voor een beknopte, veelzeggende omschrijving van je onderwerp. Het is dus niet je onderzoeksvraag die hier genoteerd wordt!)

Dit project heeft als doel de huidige glasbeweging scanners en de achterliggende services te herschrijven via voornamelijk 'green field development'. Dit betekent dat de grootste delen van dit proces vanaf nul worden herschreven. Dit houdt de scanner in en de achterliggende service waarmee de scanner communiceert. Enkel de database blijft behouden.

De functie van deze scanner applicatie is de locatie van glazen bij te houden en hun bewegingen te registreren. Dit gebeurt door de drager, in de meeste gevallen een raambok of een kar, te scannen en het glas dat op deze drager geplaats wordt te scannen. In de database wordt deze beweging geregistreerd. Daarbovenop heeft het nog enkele functionaliteiten om dit werk te versimpelen voor de 'operator'.

Deze software draait op verouderde software en hardware en is toe aan vernieuwing om toekomstige problemen te vermijden. Het onderzoek richt zich op welke technologieën gepast zijn om te gebruiken, en welke functionaliteiten moeten worden behouden of toegevoegd.

Sleurs Ben
Meerten Wouter – PXL-coach
Proost Peter – Werkplekcoach
Profel
Europalaan 17, 3900 Pelt
België

Woord vooraf

Dit project is uitgevoerd als onderdeel van mijn opleiding graduaat programmeren aan PXL Hasselt. Het idee voor dit project kwam van mijn stagebegeleider Peter Proost. De niet vervangbare scanners waren al een tijd lang een doorn in het oog maar er was nooit tijd om deze te vervangen. Het was een geschikt project voor mij om zelf de analyse te maken en te communiceren met de gebruikers.

Ik wil dan ook in eerste instantie Peter bedanken om mij bij Profel te begeleiden en mij dit project aan te bieden. Daarnaast wil ik ook Koen, Glenn en Werner bedanken. Koen heeft mij geholpen met het begrijpen van de oude code en de processen die van toepassingen waren voor deze applicatie. Glenn heeft mij geholpen met het opzetten en gebruiken van een web API. En Werner (key-user) heeft mij geholpen door steeds goede feedback te geven en duidelijk te zijn met wat hij wil en niet wil voor deze applicatie.

Ten slotte wil ik alle IT'ers van Profel bedanken om steeds voor een fijne werksfeer te zorgen en mij te helpen met alle Profel gerelateerde problemen.

Ben Sleurs

Inhoudsopgave

De inhoudsopgave kan automatisch gegenereerd worden in MS Word. Een extra nazicht of de titels en bladzijden kloppen kan echter nooit kwaad. De structuur van de inhoudsopgave heeft een logische opbouw en gaat bij voorkeur niet verder dan drie niveaus (bv. 1.1.1)

Inhoud

Woord vooraf 2

Inhoudsopgave 3

1 Bedrijfsvoorstelling 4

2 Projectvraag, onderzoeksacties en resultaten 4

2.1 Situering probleemstelling 4

2.1.1 Algemene scanner problemen 4

2.1.2 Scanners glasbewegingen 5

2.1.3 Server glasbewegingen 5

2.2 Projectvraag en deelvragen 5

2.2.1 Projectvraag 5

2.2.2 deelvragen 5

2.3 Onderzoeksdoel 6

2.4 As-Is analyse 7

2.5 Keuze nieuwe infastructuur 11

2.5.1 Web API vs. Lokaal 11

2.5.2 Xamarin vs. MAUI 11

2.5.3 Nieuw project of samen voegen 13

2.6 To Be analyse 13

2.7 Verzamelde resultaten ..... **Error! Bookmark not defined.**

3 Conclusies en aanbevelingen 16

3.1 Conclusies 16

3.2 Aanbevelingen 17

4 Persoonlijke reflecties en kritische kanttekeningen 17

5 Referentielijst 18

1 Bedrijfsvoorstelling

Profel is een Belgisch bedrijf dat gespecialiseerd is in de productie van onder andere ramen en deuren. Het bedrijf is in 1948 opgericht door de broers Frans en Albert Lauwers onder de naam FAL (Frans Albert Lauwers). In 2014 heeft de familie Lauwers de dagelijkse werking uit handen gegeven. Profel heeft zijn hoofdafdeling in Profel en 2 andere vestigingen in Achel, België en Eersel, Nederland.

Als marktleider plaatst Profel jaarlijks 190.000 ramen bij 30.000 gezinnen in België en Nederland via een groot netwerk aan verdelers [1]. Met een tewerkstelling van meer dan 1200 werknemers en een omzet van 217 miljoen euro is Profel een van de belangrijkere bedrijven in Noord-Limburg.

De IT-afdeling bevindt zich op de hoofdafdeling in Overpelt op de Europalaan 17. Of de software afdelingen werken een tiental interne- en externe programmeurs en functionele analisten. Naast de software afdeling is er ook een hardware afdeling en een afdeling WinPro.

De functie van de IT-afdeling bij Profel is puur ondersteunend. Over de jaren is er een heel pakket aan software geschreven om het werk van andere afdelingen makkelijker te maken. Profel schrijft het grootste deel van hun software zelf waardoor deze software perfect op maat is gemaakt op processen van Profel.

Binnen Profel wordt voornamelijk gewerkt in een Microsoft omgeving. De meest voorkomende programmeertalen zijn C# en VB binnen .NET. Andere producten uit de Microsoft omgeving die gebruikt worden zijn bijvoorbeeld Azure, Microsoft SQL, SharePoint

2 Projectvraag, onderzoeksacties en resultaten

In dit gedeelte kan je achtereenvolgens de probleemstelling situeren (zie hiervoor o.a. aanleiding), je onderzoeks-/projectvraag en bijbehorende deelvragen toelichten. Daarnaast kan je ook beknopt beschrijven welke acties (zie hiervoor o.a. logboek) je ondernomen hebt om je gegevens te verzamelen, en geef je vervolgens weer wat de resultaten van je uitgevoerde (onderzoeks-)acties zijn.

Je kan dit deel dus opdelen in meerdere onderdelen. Nummer deze en gebruik voor de titels ervan Stijlkop 2. Bij voorkeur worden niet meer dan drie niveaus (bv. 1.1.1 = Stijlkop 3) gebruikt.

2.1 Situering probleemstelling

2.1.1 Algemene scanner problemen

De vorige generatie scanners binnen Profel lopen op Windows CE. Windows CE is een windows gebaseerd ‘operating system (OS)’ voor kleinere machines. De laatste update van deze OS heeft het laatst een update gekregen op 13 Juni 2013. Doordat deze OS al 10 jaar geen updates meer krijgt is het moeilijk om software hiervoor aan te passen. Daarbovenop kan het ook veiligheidsrisico’s met zich meebrengen. Scanners met een moderne ‘Operating System’ zijn dus nodig.

Nieuwe software betekent in dit geval ook nieuwe hardware. Het meest voorkomende ‘operating system’ voor mobiele machines is Android [2] met 43 procent van de ‘market share’. Profel heeft dan ook gekozen om de nieuwe generatie op Android te laten lopen. De oude generatie scanners kunnen Android niet installeren.

De nieuwe scanners kunnen de oude software niet draaien omdat deze niet met Windows CE werken. Ook worden er geen nieuwe machines van de oude generaties meer gekocht. Het gevolg hiervan is dat defecte scanners vaak niet vervangen kunnen worden. Alle processen moeten dus binnenkort moderne software lopen.

Nieuwe hardware en software zal er niet enkel voor zorgen dat het proces kan blijven lopen, maar zal ook voor veel verbeteringen kunnen zorgen. Veel voorkomende klachten van de 'operators' zijn korte batterij duur, omslachtige 'user interfaces' en trage werking. Via een 'green field development' van de software en de juiste keuze van nieuwe hardware kunnen deze problemen opgelost worden en kunnen de 'operators' beter werken.

2.1.2 Scanners glasbewegingen

De scanners voor het glasbewegingen proces hebben veel last van de problemen die beschreven worden in 2.1.1 . Zo heeft de huidige software veel overbodige functionaliteiten die voor veel ophoping zorgen in de 'user interface' (UI) en kunnen nieuwe gevraagde functionaliteiten niet meer toegevoegd worden.

De hardware waar de glasbeweging scanners op lopen zijn niet sterk genoeg om zelf alle berekeningen en query's uit te voeren. Hierdoor moet een deel van deze berekeningen afgestaan worden aan 'job'. Een 'job' is software die periodiek draait op een machine om bepaalde acties uit te voeren. In dit geval zal het de commando's die de scanner heeft doorgestuurd uitvoeren en de aanpassingen uitvoeren op de database. Aangezien deze software om de paar minuten loopt duurt het soms even lang voor aanpassingen gemaakt zijn.

2.1.3 Server glasbewegingen

Direct aansluitend op het vorige punt over 'jobs' sluiten de problemen met de server side aan. Veel database acties zoals 'inserts', 'updates' en 'deletes' worden gedaan via deze 'job' waardoor de werkelijke staat van de glaslocaties en die op de database niet altijd overeenkomen. Dit kan opgelost worden door de nieuwe krachtigere scanners zelf alle operaties te laten doen maar dit zorgt voor andere problemen zoals validatie.

Een ander probleem momenteel is dat de meeste validatie lokaal op de scanners loopt terwijl een 'best practice' is dat dit 'server side' gebeurd. We willen dat de scanner zelf zo weinig mogelijk berekeningen en validaties gaat doen. Een nieuw achterliggend 'framework' is nodig.

2.2 Projectvraag en deelvragen

2.2.1 Projectvraag

Kan de huidige glasbewegingen applicatie die gebruik maakt van Windows CE 'geport' worden naar een moderne infrastructuur die ook op Android scanners werkt tegen het einde van mijn stage bij Profel (2 Juni)?

2.2.2 deelvragen

- Hoe werkt de 'client side' applicatie momenteel.
- Hoe werkt de API momenteel.
- Is er een modern 'framework' beschikbaar dat deze applicatie kan vervangen en op moderne scanners(en mogelijks smartphones) werkt.
- Gaat de data achterliggen op dezelfde manier verwerken.
- In welke mate gaat de applicatie UI veranderen.

2.3 Onderzoeksdoel

In de eerste plaats is het doel van dit onderzoek het kiezen van een goede, langdurige infrastructuur waarop de nieuwe software kan geschreven worden. De nieuwe infrastructuur moet zorgen dat alle huidige problemen opgelost kunnen worden en dat de IT-afdeling hierop verder kan bouwen bij gevraagde aanpassingen of opkomende problemen.

Het nieuwe 'framework' moet ervoor zorgen dat alle huidige functionaliteiten die behouden gaan worden en gevraagde functionaliteiten geïmplementeerd kunnen worden op de nieuwe Android scanners, De Microsoft SQL database en de IT-infrastructuur van Profel. Daarbovenop moet het ook langdurig kunnen werken en onderhouden worden. Het idee is dat er niet nog een 'green field development' moet gebeuren in de nabije toekomst maat dat deze software mee kan geüpdatet worden met nieuwe veranderingen in de IT-wereld.

Ook niet onbelangrijk is dat de infrastructuur makkelijk onderhouden kan worden door de IT-afdeling van Profel zelf. Het heeft weinig nut een onderhoudbare infrastructuur te kiezen als niemand er goed mee kan werken. Dit limiteert de keuzen maar het .Net 'framework', waar de meeste programmeurs bij Profel mee bekend zijn, heeft een groot aanbod.

Het kiezen van de juiste infrastructuur is heel belangrijk. Dit zal grotendeels de onderhoudbaarheid, functionaliteit en levensduur bepalen. Dit deel zal goed overlegd worden met de verantwoordelijke programmeurs voor de glasbewegingen.

In de tweede plaats wordt de huidige software onderzocht. Er zal een As-Is analyse gemaakt worden om vast te stellen hoe de huidige applicatie en server werkt. In samenspraak met Werner(key-user) en Koen(verantwoordelijke programmeur) zal beslist worden welke functionaliteiten behouden kunnen worden en welke functionaliteiten moeten worden toegevoegd. Ook zal er direct feedback over de huidige applicatie opgenomen worden om mogelijke verbeteringen meteen te kunnen aanbrengen.

Bijkomend wordt er onderzocht of er betere 'performance' kan worden behaald. Dit zal vooral gebeuren bij het implementeren van de SQL query's. Zo zal er bijvoorbeeld bekeken worden of er niet te veel data opgehaald wordt, meerdere query's samen gezet kunnen worden, etc… Data in de database ophalen en updaten zal een groot deel van de berekening kracht en tijd vragen en hier zal mogelijks veel 'performance' kunnen gewonnen worden.

Ten slotte wordt bekeken in welke mate dit geïmplementeerd kan worden tegen 2 juni 2023. Welke delen van de applicatie kunnen al geschreven worden, in welke mate kunnen deze getest worden etc... Dit zal van interne factoren afhangen zoals de kennis van het gekozen netwerk maar ook van externe factoren zoals de beschikbaarheid van Android scanners om gebruikers mee te laten testen.

2.4 As-Is analyse

Het eerste deel van het onderzoek richt zich op de huidige applicatie en software. Een eerste stap hierin is op de werkvloer kijken hoe de huidige applicatie werkt. Er wordt gevraagd aan ‘operators’ om te tonen hoe ze de applicatie gebruiken, wat er nog gebruikt wordt en wat niet. In dit deel wordt ook meteen de feedback gevraagd over problemen en mogelijk nieuwe functionaliteiten.

Uit dit deel van de analyse zien we op welke manier een raambok of kar gescand wordt en hoe de glazen gescand worden. Op figuur 1 is het hoofdscherm van de glasbewegingen scanner te zien. Hieruit valt af te leiden dat de raambok/kar en de glazen op hetzelfde scherm gescand worden. Er is ook een scherm met alle gescande glazen voor deze raambok plus extra informatie over de bestemming. Om een raambok te scannen moet er eerst op het raambok veldje geduwd worden en om een glas te scannen moet er eerst op het glasnummer veldje geduwd worden. Bij het scannen van een glas in het glasnummer veld zal deze verwerkt worden zonder een confirmatie dialoog.

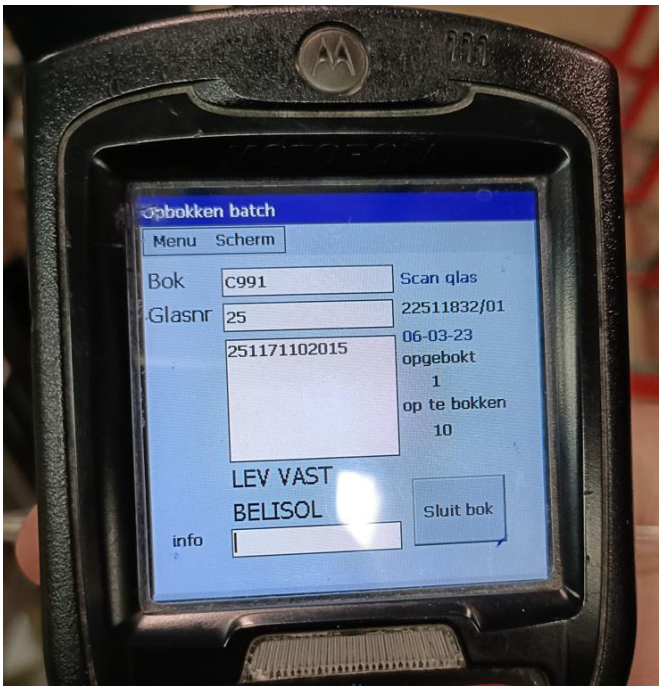
Door op het info veldje te duwen een glas te scannen openen we het info scherm over alle glazen met dezelfde bestemming als het gescande glas.

Het infoscherm bestaat uit 3 tabbladen. Op figuur 2 zien we het ‘Aantal’ tabblad. Hierop wordt elke raambok met het aantal glazen dat dezelfde bestemming heeft als het gescande glas getoond. Deze informatie is nodig om andere glazen die nodig zijn te vinden.

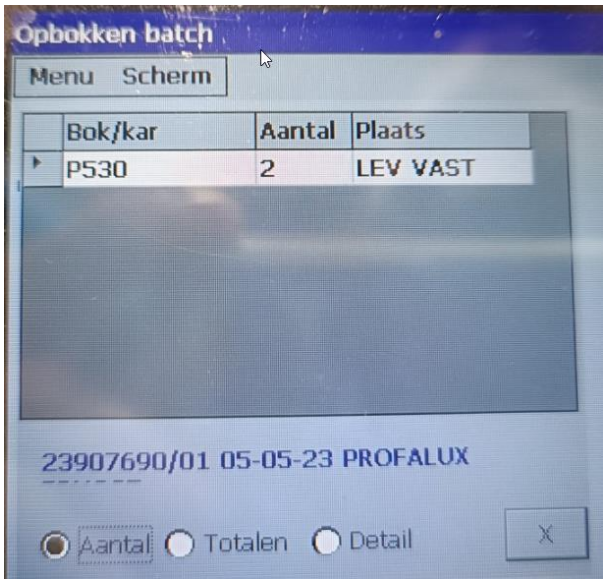
Op figuur 3 wordt het ‘Totalen’ tabblad getoond. Hierop zijn het aantal glazen te zien dat dezelfde bestemming heeft als het gescande glas. Zouden er bijvoorbeeld 2 rijen zijn in het ‘Aantal’ tabblad met dezelfde ‘Plaats’ en elk een aantal van 2, dan zou er op het ‘Totalen’ tabblad een rij te zien zijn met die ‘Plaats’ als leverplaats en een ‘totaal’ van 4.

Daarnaast zien we op figuur 4 het ‘Detail’ tabblad. Op dit tabblad zijn de details van alle glazen te zien met dezelfde bestemming als het gescande glas.

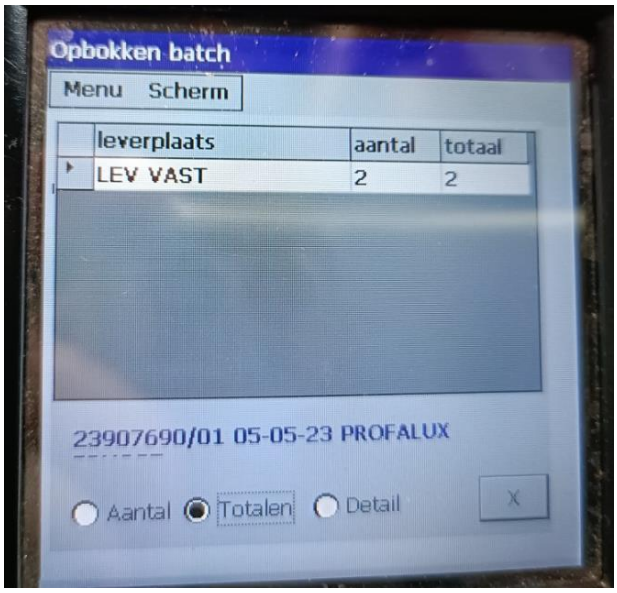
Andere afdelingen gebruiken een gelijkaardige applicatie met een andere UI en een licht afwijkende validatie. Indien mogelijk mogen deze allemaal in dezelfde applicatie met validatie gebaseerd op inlog gegevens.



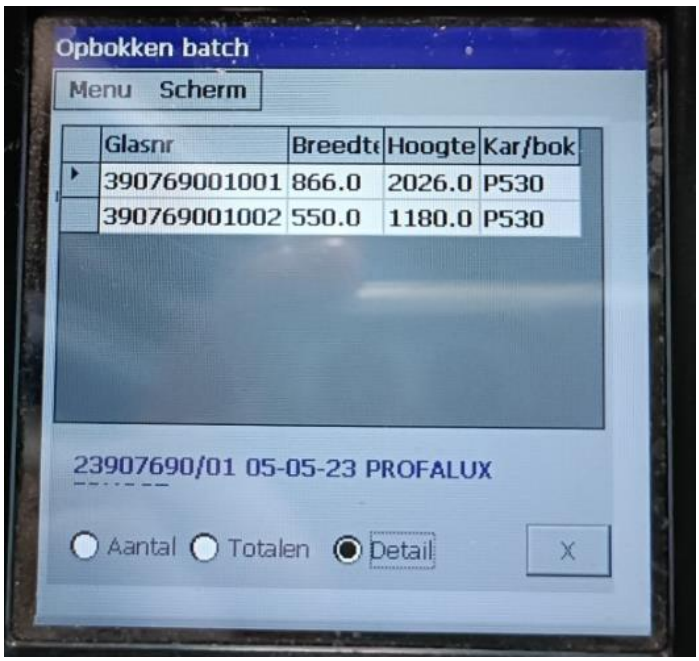
Figuur 1: Hoofdscherm glasbewegingen scanner



Figuur 2: Infoscherm op 'Aantal' tabblad



Figuur 3: Infoscherm op 'Totalen' tabblad



Figuur 4: Infoscherm op 'Detail' tabblad

Ten tweede wordt de code zelf bekeken. Belangrijke delen die we kunnen linken met de analyse op de werkvloer worden gezocht zoals de navigatie tussen schermen, gebruikte query's, validaties etc… Het is belangrijk dat de uiteindelijke bewerkingen in de database identiek blijven zoals ze op de oude software gebeuren.

In de originele Visual Basic (VB) code staan veel stukken die gebruikt kunnen worden in de nieuwe applicatie. Zo zijn er de SQL query's te vinden die overeenkomen met de info schermen. Op figuur 5 is de query te zien die overeenkomt met figuur 2 maar ok voor figuur 3 en 4 zijn de query's te vinden. Hoewel deze query's er relatief goed uitzien is er mogelijks optimalisatie mogelijk maar dit word pas bekeken bij het implementeren van de code. Zo zouden mogelijks de 3 query's herwerkt kunnen worden naar 1 query.

Daarnaast is er ook validatie code te vinden zoals in figuur 6. De voorbeeld code controleert of de leverplaatsen van het laatst gescande glas compatibel zijn met die van de glazen op de raambok. Hier is duidelijk te zien dat er verbetering mogelijk is, beide in performantie en leesbaarheid. In de originele code is de validatie vaak een reeks 'if-else statements' achter elkaar. Door deze algemener en korter te herschrijven zal de validatie sneller gebeuren en ook leesbaarder zijn.

```
apter("SELECT ed012.omsch30_1 as bokkar, sbstl as plaats, COUNT(*) AS aantal FROM ec021 " & _
  " left outer join ed012 ON ec021.GRBCN = ed012.GRBCN AND ec021.GRBCT = ed012.GRBCT " & _
  " inner join ec020 ON ec021.glasn = ec020.glasn " & _
  " WHERE ec021.GLASN IN (SELECT GLASN FROM EC020 a WHERE (REFER = (SELECT REFER FROM EC020 b WHERE (b.GLASN = '" & glasnr & "'))))" & _
  " AND (ec021.VOLGN = (SELECT MAX(a.volgn) FROM ec021 a WHERE a.glasn = ec021.glasn and stats='05')) " & _
  " and not exists (select * from ec024 where ec024.oglas = ec021.glasn) " & _
  " GROUP BY ed012.omsch30_1, sbstl " & _
  " ORDER BY  sbstl,ed012.omsch30_1 ", New SqlClient.SqlConnection(connectionString))
```

Figuur 5: Originele query voor figuur 2 scherm

```
If glasinfo.Leverplaats <> "HOUT" And opbokleverplaats = "HOUT" Then
    MsgBox("Bok heeft leverplaats hout en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If glasinfo.Leverplaats <> "FRANCE" And opbokleverplaats = "FRANCE" Then
    MsgBox("Bok heeft leverplaats france en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If glasinfo.Leverplaats <> "GOES" And opbokleverplaats = "GOES" Then
    MsgBox("Bok heeft leverplaats goes en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If


If (glasinfo.Leverplaats <> "LEV VAST" And glasinfo.Leverplaats <> "LEV LOS") And opbokleverplaats = "LEV VAST" Then
    MsgBox("Bok heeft leverplaats LEV VAST en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If (glasinfo.Leverplaats <> "LEV LOS" And glasinfo.Leverplaats <> "LEV VAST") And opbokleverplaats = "LEV LOS" Then
    MsgBox("Bok heeft leverplaats LEV LOS en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If (glasinfo.Leverplaats <> "ALU BOUW" And glasinfo.Leverplaats <> "PVC BOUW") And opbokleverplaats = "ALU BOUW" Then
    MsgBox("Bok heeft leverplaats ALU BOUW en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If (glasinfo.Leverplaats <> "PVC BOUW" And glasinfo.Leverplaats <> "ALU BOUW") And opbokleverplaats = "PVC BOUW" Then
    MsgBox("Bok heeft leverplaats PVC BOUW en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If

If (glasinfo.Leverplaats <> "LEV LOS" And glasinfo.Leverplaats <> "LEV VAST") And opbokleverplaats = "LEV LOS" Then
    MsgBox("Bok heeft leverplaats LEV LOS en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If


If glasinfo.Leverplaats <> "AFH VAST" And opbokleverplaats = "AFH VAST" Then
    MsgBox("Bok heeft leverplaats AFH VAST en glas " & glasinfo.Leverplaats, MsgBoxStyle.Exclamation, "Fout")
    OpbokkenGoedkeuring = False
End If
```

Figuur 6: Validatie code uit originele VB code

De UI is gemaakt met Windows Forms zonder een 'Model-View-ViewModel' (MVVM) implementatie. Hoewel dit technisch en functioneel geen probleem is maakt het ook hier sommige code moeilijk leesbaar. Op Figuur 7 is een voorbeeld te zien van hoe de UI veranderd moet worden bij een bepaalde actie. Veranderingen aanbrengen in dit soort code is omslachtig en moeilijk leesbaar. Ook hier is verbetering mogelijk.

```
lblContainerTotaal.Visible = True
lstContainer.Visible = True
lblContainerRefer.Visible = True
lblContainerGlasnLabel.Visible = False
lblContainerGlasn.Visible = False
lstContainer.Visible = True
'cmdContainerDetail.Visible = True
btnContainerOk.Visible = True
lblOpmerking.Text = ""
strScanVeld = "Container"
HideWaitCursor()
```

Figuur 7: UI code

2.5 Keuze nieuwe infastructuur

2.5.1 Web API vs. Lokaal

Een eerste keuze die gemaakt moet worden is of de database operaties lokaal op de scanner zelf gaan gebeuren of er 'requests' naar een 'web API' (web Application Programming Interface) die de operaties gaat uitvoeren. Zoals te zien was in de As-Is analyse worden momenteel de meeste acties lokaal uitgevoerd en sommigen naar een server (geen 'web-API') gestuurd.

De voordelen van alles lokaal te doen is, is dat er geen tijd verloren gaat met het uitsturen van een 'request' naar een server en de scanner direct toegang heeft tot de database. Naast dit voordeel heeft ook nadelen. Zo zal alles lokaal meer computerkracht vragen van de scanner wat tot een kortere batterijduur of een tragere applicatie kan zorgen. Daarnaast brengt dit ook veiligheidsrisico's met zich mee. Ergens op de scanner moet er connectie gemaakt worden met de database, met deze connectie zijn veel meer database operaties mogelijk dan nodig is. Hierbij moet wel gezegd worden dat deze connectie misbruiken moeilijk is en onwaarschijnlijk zal voorkomen.

Een laatste nadeel is dat het gebruik van een 'web API' de front-end en backend code beter opsplitst. Zouden en nieuwe scanners komen die nieuwe code nodig hebben is het niet nodig om de code van de 'web API' te vervangen, de nieuwe applicatie moet enkel de juiste 'requests' sturen. Als we de voor- en nadelen afwegen is het duidelijk dat het gebruik van een 'web API' de juiste keuze is.

2.5.2 Xamarin vs. MAUI

Voor het intern ontwikkelen van een mobiele applicatie bij Profel die ook intern onderhouden kan worden zijn er maar 2 opties. Beide Xamarin en MAUI zijn cross-platform frameworks voor mobiele applicaties die gebruiken van .Net. Alle programmeurs binnen Profel kunnen werken met .Net in C# of VB. Hoewel het idee van deze frameworks is er ontwikkeld kan worden voor meerdere platformen en de scanners altijd op Android zullen lopen is het het niet waard om de applicatie in een ander 'framework' en andere taal te schrijven die veel moeilijker onderhoudbaar zal zijn binnen Profel.

De keuze ligt dus tussen MAUI en Xamarin. Xamarin is het iets oudere 'framework' dat zich ondertussen al heeft kunnen bewijzen als functioneel platform. Dit is ook het 'framework' waar andere nieuwe mobiele applicaties bij Profel op draaien. Het grootste nadeel aan Xamarin is dat het maximaal ondersteund wordt op .NET Standerd 2.1 en er ook geen plannen zijn om Xamarin te laten draaien op de nieuwe .NET Core versies.

Dit komt omdat MAUI, de opvolger van Xamarin, in leven werd gebracht met op .NET Core kunnen draaien als één van haar doelen. MAUI is uitgegeven op 23 Mei 2022 en op tijdstip van schrijven nog maar net een jaar oud. MAUI wordt wel ondersteund op de nieuwste versies van .NET Core en zou dus in theorie langer onderhoudbaar zijn dan Xamarin. Het grootste nadeel aan MAUI is dat het zich nog niet heeft kunnen bewijzen en nog altijd meer onverwachte uitkomsten heeft dan Xamarin.

Hoewel beide 'framework' een goede mogelijkheid waren is er toch gekozen voor Xamarin. .NET Standard heeft alle functionaliteiten die nodig zijn en Xamarin werkt. MAUI blijft een beetje een gok en .NET Core is voorlopig nog niet nodig. Ten slotte zijn er mogelijkheden om Xamarin naar MAUI te porten zou het nodig zijn.

2.5.3 Nieuw project of samen voegen

Sinds er besloten is om Xamarin te gebruiken is er de mogelijkheid om deze applicatie aan een andere scanner applicatie te hangen en functionaliteiten toe te kennen op basis van de inlog gegevens. Het voordeel hieraan is dat er veel 'overhead' gedaan is zoals het opstarten van een project, Profel unieke 'controls' en 'services' waardoor de programmatie tijd beduidend korter wordt. Een nadeel is dat de code van de programma's onafhankelijk zijn en het dus niet nodig is om deze samen te zetten maar dat er toch door fouten bij het programmeren van één applicatie de andere stuk kan gaan.

Een nieuw project heeft verschillende voordelen. Zo is het makkelijk om meteen van .NET Standard 2.0, waar de magazijnbewegingen applicatie op draait, naar .NET Standard 2.1 te gaan. Ook blijven de 2 applicaties volledig onafhankelijk en ten laatste is er de keuze om voor andere, soms betere, 'frameworks' te gaan binnen Xamarin. Zo wordt er momenteel MVVM Cross gebruikt voor de navigatie en het MVVM gedeelte terwijl de 'Xamarin community toolkit' deze mogelijkheden ook aanbied.

Na overleg is er besloten om de applicatie vast te hangen aan de magazijnbewegingen applicatie. De totale applicatie is 'light weight' genoeg om beide tegelijk te draaien en de rechtentoekenning is simpeler te houden. Daarbovenop zou het nieuw project ook gebruik moeten maken van MVVM Cross in plaats van de 'Xamarin community toolkit' om de applicatie onderhoudbaar te houden door andere programmeurs.

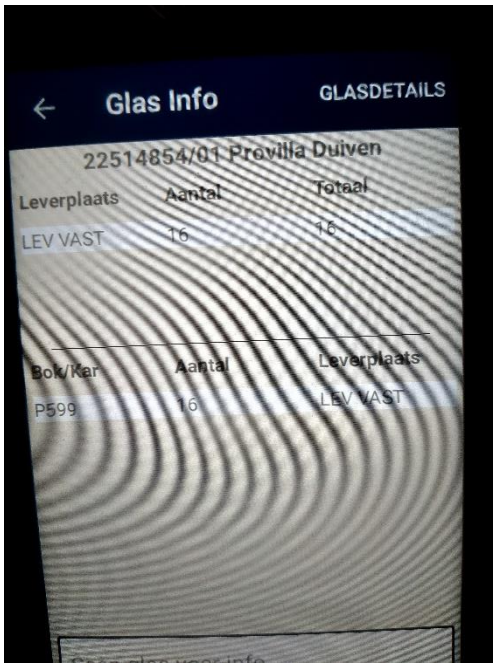
2.6 To Be analyse

De nieuwe applicatie zal over het algemeen dezelfde functionaliteiten behouden. De grootste verandering is dat voor elke 'request' de applicatie via een 'web API' zal gaan waardoor dubbele validatie een mogelijkheid wordt. Een andere grote verandering is dat de applicatie op een Android platform zal draaien in plaats van een Windows platform. De UI zal dus meer gefocust moeten worden op 'touchscreen' vriendelijke functionaliteiten.

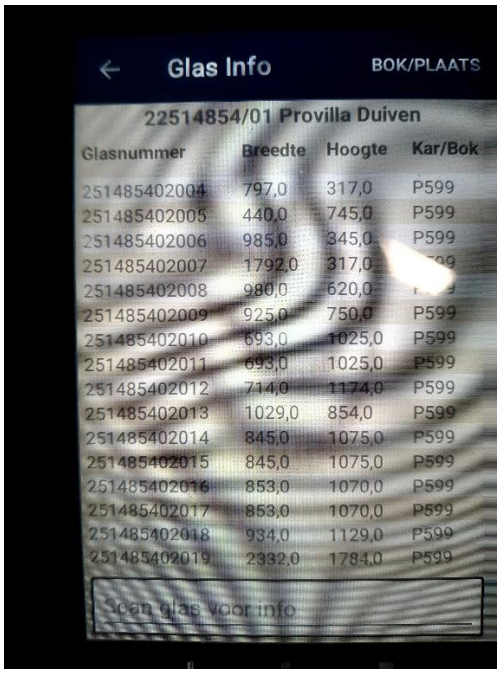
Een extra voordeel is dat de nieuwe scanners een groter scherm hebben. Hierdoor kan de informatie in een groter 'fontsize' getoond worden en worden de knoppen makkelijker te gebruiken. Ook kunnen enkele overbodige velden verplaatst of verwijderd worden. Zo kan het info 'Entry' veld verplaatst worden naar een 'toolbaritem' bijvoorbeeld. Een eerste versie leverde de UI uit figuur 8 en 9 op. (TODO: FIGUUR 8 en 9 TOEVOEGEN).

Het info scherm heeft ook ruimte voor verbetering. De 3 tabbladen moeten naar 2 of zelfs liefst 1 gaan indien mogelijk. Indien er naar 1 scherm gegaan wordt moet er info getoond worden afhankelijk van het geselecteerde item in het totaal en aantal tabblad. Hierdoor worden niet alle glasnummers tegelijk getoond maar enkel de glasnummers die op de geselecteerde raambok staan.

Een andere optie is om te gaan naar 2 schermen. In dit geval is er wel ruimte om alle info te tonen zonder raambokken te moeten selecteren. In dit geval zullen het 'Aantal' en 'Totaal' tabblad in samengevoegd worden in 1 tabblad. Op het tweede tabblad blijven de glazen. Een voorbeeld van deze versie is te zien op figuur 10 en 11. Via de knop rechtsboven kan gewisseld worden tussen de info tabbladen.



Figuur 10: Aantal en Totaal info scherm



Figuur 11: Info scherm met glas nummers

Hoewel er geen functionele veranderingen komen, gaat de werking van de nieuwe applicatie wel veranderen. De belangrijkste verandering is dat de verwerking van bewegingen achter liggend niet meteen bij het scannen van een glasnummer zal gebeuren. De ‘operator’ zal eerst elk glas dat op de bok verplaatst is scannen. Deze glazen komen terecht in een lijst op het scherm waar ze makkelijk terug verwijderd kunnen worden. Pas bij het drukken op de verwerk knop zullen de bewegingen in de database geregistreerd worden.

De applicatie zal 4 ‘requests’ kunnen sturen naar de web-API:

- Bok controleren
- Glas scannen en controleren
- Bewegingen verwerken
- Details over glas opvragen

De eerste request die de applicatie kan sturen is om een bok te controleren. Deze request wordt gestuurd als de ‘operator’ een raambok scant. De web API controleert of dit een geldige raambok en stuurt de bijhorende informatie over de raambok terug naar de applicatie indien deze geldig is. De info komt terecht op het scherm en de ‘operator’ kan beginnen met scannen van de glazen.

Een tweede request gebeurt als de ‘operator’ een glas scant die op het voordien gescande bok verplaatst is. De applicatie stuurt het glasnummer en het raamboknummer naar de web API. De API voert de nodige validaties uit. Als de validaties geslaagd zijn stuurt de API de nodige informatie over het glas terug naar de applicatie.

Als de operator of de verwerk knop duwt, stuurt de applicatie een derde request naar de web-API. De applicatie stuurt de lijst met in gescande glazen en het boknummer door naar de API. Alweer voert de API eerst de nodige validaties uit. Nadien zal ze proberen de bewegingen te verwerken en op te slaan in de database. De applicatie krijgt een geslaagd bericht of een foutbericht terug van de API.

De vierde request gebeurt vanuit het info scherm. Als de ‘operator’ hier een glas scant gaat de API alle nodige informatie terugsturen over alle glazen met hetzelfde ordernummer als het gescande glas. Hierdoor kan de ‘operator’ makkelijk info krijgen over waar de rest van de opdracht staat. Dit kan bijvoorbeeld aan de hand van het boknummer en het aantal glazen dat hier op staat zoals te zien op figuur 10. De ‘operator’ kan via dit boknummer makkelijk de rest van de glazen vinden zoals te zien op figuur 12. De raambok identificatie is te zien op de gele plakkaten boven de raambokken.

Figuur 12: Raambokken in de hal van Profel

Ook zal veel code herschreven worden door deze te ‘refactoren’. Dit betekent dat de code in een mooier jasje gestoken wordt. Puur functioneel verandert er niets maar de code wordt beter onderhoudbaar in de toekomst. Zoals te zien op figuur 6 voldoet niet alle code van de oude applicatie aan moderne normen. Ten slotte worden de query’s verbeterd door gebruik te maken van ‘profiling’. De bestaande query’s worden geanalyseerd er indien mogelijk verbeterd op vlak van snelheid.

3 Conclusies en aanbevelingen

In dit gedeelte geef je je conclusies die je vanuit je uitgevoerde onderzoek kan afleiden, weer. Je beantwoordt hier je onderzoeksvraag: je geeft op een synthetische en goed beargumenteerde manier een antwoord op je onderzoeksvraag en staat stil bij de belangrijkste vaststellingen van je onderzoek. Je geeft tips en aanbevelingen voor je werkplek naar voor schuiven die je vanuit je onderzoek aangereikt krijgt/gekregen hebt.

3.1 Conclusies

Het herschrijven van de applicatie in een modern framework is gelukt en heeft voor verschillende verbeteringen gezorgd. Naast directe verbeteringen zoals moderne scanners en een moderne ‘up to date’ OS zijn er ook indirecte verbeteringen. De code is van de grond af opnieuw geschreven met de huidige kennis van de vorige ‘codebase’. Hierdoor is de code moderner, efficiënter en beter onderhoudbaar.

Sinds de software voor Android is geschreven zal deze ook draaien op Android smartphones. Sinds de meeste smartphones geen barcode scanner heeft zal een manier toegevoegd moeten worden op barcodes te scannen via de camera.

Ten slotte valt de concluderen dat de functionaliteit van de oude applicatie grotendeels is behouden. Hoewel de achterliggende systemen achter de applicatie ouderwets waren, waren de 'flows' van de applicatie goed en wouden de gebruikers deze overwegend behouden.

3.2 Aanbevelingen

Er kan goed gebruik gemaakt worden van dit soort projecten om processen te verbeteren. 'Green field development' is een gouden kans om verbeteringen toe te passen en opnieuw te kijken naar implementatie van geschreven software. Het biedt ook een kans aan om de code te 'refactoren'.

4 Persoonlijke reflecties en kritische kanttekeningen

Je noteert hier hoe jij het werken aan je Graduaatsproef ervaren hebt. Belangrijke persoonlijke leerinzichten en kritische kanttekeningen die je meeneemt vanuit het doorgemaakte proces kunnen hier hun plaats krijgen.
Je beschrijft wat voor jou persoonlijk de meerwaarde van de uitwerking van je Graduaatsproef is; wat kan/wil je ***voor jezelf*** *vanuit het doorlopen van je onderzoek als* ***leerinzichten*** *naar voor schuiven, koppel deze leerinzichten naar de OLR. Daarnaast zijn ook jouw persoonlijke kritische kanttekeningen belangrijk om te vermelden: waar zit voor jou de* ***meerwaarde en eventuele beperkingen van je onderzoek zoals jij dit doorlopen hebt****? Belangrijk hierbij is dat je in deze reflecties de link legt met hetgeen bovenstaand werd neergeschreven en dat je in je reflecties dus verwijst naar concrete informatie/acties… vanuit je Graduaatsproef.*

Op puur technisch vlak was deze graduaatsproef goed doenbaar en was ik zeker voorbereid om deze applicatie te herschrijven. Ik had al ervaring met beide '.NET mobile development' en het met het gebruik van een web API. Ook met het gebruik van scanners had ik ondertussen ervaring door andere projecten bij Profel.

Op andere vlakken heb ik wel meer bijgeleerd. Zo heb ik zelf moeten onderzoeken wat de gepaste infrastructuur was voor dit project, iets wat mij bij andere projecten meestal voorgeschoteld werd. Het was hier belangrijk dat ik rekening hield met een toekomst zonder mij voor de applicatie, andere mensen moesten hieraan verder kunnen werken zonder mijn inbreng.

Ten tweede was het weer een nieuw bedrijfsproces dat ik goed moest begrijpen voor ik hieraan kon werken. Gelukkig had ik voor dit project wel toegang tot alle code. Ook kon ik makkelijk op de vloer gaan kijken hoe de applicatie gebruikt werd sinds de gebruikers in hetzelfde gebouw als de IT-afdelingen zaten.

Ten laatste was dit mijn eerste project waar ik in direct contact stond met eindgebruikers. Ik heb hier veel geleerd over wat gebruikers willen tegenover wat ontwikkelaars en analisten willen. Mijn interpretatie van hoe de applicatie het beste zou gemaakt worden kwam regelmatig niet overeen met hoe de gebruikers dit in gedachten hadden. Ik had niet meer totale vrijheid zoals ik in veel persoonlijke of schoolprojecten had.

5 Referentielijst

De referenties die je raadpleegde en die belangrijk zijn in de uitwerking van je Graduaatsproef dienen hier vermeld te worden. Je noteert dit op een consequente en correcte manier.

[1] https://www.profel-group.com/nl-be/over-de-profel-group/kerncijfers-geschiedenis/

[2] https://www.scaler.com/topics/operating-system-market-share/