Professionele Bachelor Toegepaste Informatica

Automatisatie van betalingen in restaurants

Jana Giebens

Promotoren:

Steffen Brans — Appwise
Jan Willekens — Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2018-2019

Professionele Bachelor Toegepaste Informatica

Automatisatie van betalingen in restaurants

Jana Giebens

Promotoren:

Steffen Brans — Appwise
Jan Willekens — Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2018-2019

Dankwoord

Ik wil graag verschillende mensen bedanken om dit eindwerk mogelijk te maken.
Ten eerste wil ik dhr. Steffen Brans bedanken om mij de kans te geven om mijn stage bij Appwise te doen. Alsook voor de goede begeleiding tijdens mijn stage. Ik wil ook iedereen binnen Appwise bedanken om altijd klaar te staan om mij te helpen en om mijn stageperiode ongelofelijk aangenaam te maken.
Ik wil ook mijn hogeschoolpromotor, dhr. Jan Willekens, bedanken voor zijn goede begeleiding bij mijn stage en onderzoek.
Ten laatste wil ik ook mijn mama bedanken om mij altijd bij te staan tijdens mijn studies.

Abstract

Het doel van deze opdracht is de automatisatie van betalingen in een restaurant. Een klant moet in staat zijn om, zonder tussenkomst van een ober, de rekening te ontvangen en te betalen. Het proces verloopt als volgt: de klant identificeert zijn tafel, de klant ontvangt de rekening die bij zijn tafel hoort op zijn smartphone en de klant betaalt deze rekening.

In dit project werd voor de volgende implementatie gekozen. De klant identificeert zijn tafel door een QR-code te scannen. Door deze QR-code te scannen, wordt er een verzoek gestuurd naar de *api* van het kassasysteem. De klant krijgt zijn kassaticket te zien en indien deze correct is, kan de klant betalen. Er werd gekozen voor Stripe Connect om de betaling af te handelen. Door gebruik te maken van een provider wordt er geen gevoelige data door de server van de applicatie gestuurd. Een correcte implementatie van de Stripe-*api* garandeert dat de betaling verloopt overeenkomstig de Europese wetgeving.

In de tekst wordt onderzocht of de automatisatie van het proces mogelijk is en wat de efficiëntste manier is voor iedere stap. Het onderzoek bestaat uit drie delen. Ten eerste wordt bekeken welke de meest gebruiksvriendelijkste manier is om een tafel te identificeren. Hier wordt gekeken naar QR-codes en OCR. De gebruiksvriendelijkheid wordt onderzocht aan de hand van een *hallway test*. Hierbij hebben verschillende, willekeurige, mensen de twee mogelijkheden uitgeprobeerd en enkele vragen beantwoord. Ten tweede is onderzocht hoe de applicatie geïntegreerd kan worden met een kassasysteem. Ook wordt er een vergelijking gemaakt tussen twee *payment service providers*, namelijk Stripe Connect en Adyen MarketPay. Beide providers bieden de mogelijkheid om betalingen te ontvangen en door te sturen naar derden, in dit geval het restaurant. In de vergelijking wordt onder andere gekeken naar de kostprijs, snelheid en verschillende mogelijkheden van de oplossingen.

Aangezien de applicatie bedoeld is voor iOS wordt er ontwikkeld in Swift. XCode is de gebruikte programmeeromgeving. De backend is geschreven in PHP en er is gebruikgemaakt van het framework Laravel.

Inhoudsopgave

Lijst van gebruikte figuren

Lijst van gebruikte tabellen

Lijst van gebruikte afkortingen

API	Application Programming Interface
ISA	Internal Security Assessor
KYC	Know Your Customer
NDA	Non-disclosure Agreement
OCR	Optical Character Recognition
PCI	Payment Card Industry
PCI DSS	Payment Card Industry Data Security Standard
PSD	Payment Service Directive
PSD2	Second Payment Service Directive
QR	Quick Response
QSA	Qualified Security Assessor
ROI	Return On Investment
SAQ	Self-Assessment Questionnaire
SDK	Software Development Kit
SQRC	Secret-function-equipped Quick Response Code
URL	Uniform Resource Locator

Inleiding

Het doel van dit eindwerk is om het betalingsproces in restaurants te automatiseren. Wanneer klanten naar een restaurant gaan, omvat dit proces de volgende stappen: de klant roept de ober, de klant ontvangt het kassaticket en de klant betaalt de rekening met de bankkaart of met cash geld. Het einddoel van deze opdracht is om dit proces te laten verlopen zonder de tussenkomst van een ober. Het proces verloopt dan als volgt: de klant identificeert zijn tafel, de klant ontvangt de rekening die bij zijn tafel hoort op zijn smartphone en de klant betaalt deze rekening. In dit proces zijn er enkele obstakels die dit proces bemoeilijken. De klant moet de tafel kunnen identificeren. De rekening moet op de smartphone verkregen worden. De rekening moet betaald kunnen worden via de app.

Op de vraag hoe dit proces moet worden aangepakt, wordt in dit eindwerk een antwoord geformuleerd. Er wordt een vergelijking gemaakt tussen QR-codes en OCR om de optimale manier te bepalen om een tafel te identificeren. Deze oplossingen worden vergeleken op basis van snelheid en gebruiksvriendelijkheid. De gebruiksvriendelijkheid wordt onderzocht aan de hand van een *hallway test*. De snelheid wordt bepaald door het interval tussen het begin en het einde van de operatie te meten. De rekening kan op de smartphone verkregen worden door integratie met een kassasysteem. Er wordt geïntegreerd met het kassasysteem van Untill. De applicatie kan een betaling uitvoeren door gebruik te maken van een *payment service provider*. In dit onderzoek wordt een vergelijking gemaakt tussen Stripe Connect en Adyen MarketPay. In de vergelijking wordt onder andere gekeken naar de kostprijs, snelheid en verschillende mogelijkheden van de oplossingen.

In de literatuurstudie wordt eerst dieper ingegaan op zowel QR-codes als OCR. Er wordt uitgelegd wat het is en hoe het werkt. Vervolgens wordt het online betalingsproces besproken. Wat het proces is en hoe *payment service providers* hierbij kunnen helpen. Vervolgens worden de twee *payment service providers* Stripe Connect en Adyen MarketPay verder toegelicht.

I. Stageverslag

1 Bedrijfsvoorstelling

Appwise werd opgericht in 2012 en is een bedrijf dat gespecialiseerd is in het maken van apps. De hoofdfocus ligt op het maken van iOS en Android apps. Appwise ontwikkelt hoofdzakelijk smartphone en tablet oplossingen voor hun klanten wereldwijd. Het bedrijf bouwt applicaties volgens hun eigen Shape-Build-Activate proces ondersteunt door een multidisciplinair team van 28 enthousiaste medewerkers. Het proces begint met het vertalen van de wensen van de klant. Er worden analyses gemaakt en de verschillende features van de app bepaald. Vervolgens worden er aan de hand van deze analyse designs en prototypes gemaakt, waarbij er sterk gefocust wordt op gebruikerservaring, bruikbaarheid en aantrekkelijkheid. Er wordt ook altijd meteen terug met de klant overlegd. De gemaakte apps moeten de FSE-test doorstaan. Dit wilt zeggen dat de Fast, Sexy en Easy moeten zijn. Vervolgens worden de apps native gemaakt voor Android en/of iOS. Deze apps worden gemaakt in een multidisciplinair team op een *agile* manier. Het bouwen van een app verloopt in verschillende sprints en kan door verschillende iteraties gaan. Bij Appwise houden de teams ook *stand-up* en *scrum meetings* en maken ze gebruik van een *tool* voor het *sprint management*, Kanban-bord,…. De apps worden ook steeds uitvoerig getest door meerdere personen. Een goede app moet ook gebruikers hebben daarom helpt Appwise ook bij het activeren van apps in de markt of organisatie. Een belangrijk onderdeel is de co-creatie met de klanten.

2 Voorstelling stageopdracht

2.1 Automatiseren van betalen in restaurants

Het doel van deze stageopdracht is om te kijken of het mogelijk is om het proces van betalen in restaurants te automatiseren. Het huidige proces is als volgt. Een klant roept de ober en vraagt om de rekening. De ober gaat de rekening halen en brengt deze naar de klant. De klant betaalt daarna de rekening met een bankkaart of met cash geld. Indien de ober het druk heeft, kan het even duren voordat de klant zijn rekening ontvangt en kan betalen.

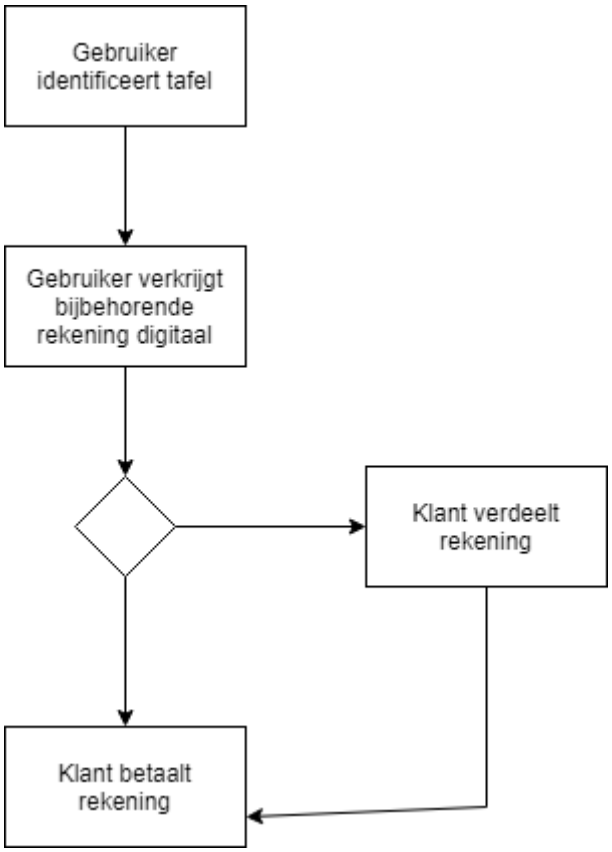
In deze stageopdracht wordt geprobeerd om een applicatie te maken waarbij een klant zijn rekening kan krijgen en betalen zonder tussenkomst van een ober. De klant moet via de applicatie in staat zijn om de rekening te ontvangen, de rekeningen te splitsen met anderen en de rekening te betalen. De app is gemaakt voor iOS en is ontwikkeld in Swift 4. De programmeeromgeving is XCode. De backend is geschreven in PHP en er is gebruikgemaakt van het framework Laravel. Verder is er voor de integratie met een kassasysteem gebruikgemaakt van de *api* en testomgeving van Untill.

3 Uitwerking stageopdracht

3.1 Analyse van de opdracht

Bij aanvang van de stage werd er een analyse van de opdracht uitgevoerd. Tijdens deze analyse werd het huidige proces in kaart gebracht. Deze figuur is te zien in de bijlagen. Omwille van de grootte van de figuur is deze gesplitst in bijlagen A en B. In bijlage A is deel 1 te zien en in bijlage B het vervolg.

De scope van deze stageopdracht beperkt zich tot de laatste vijf stappen. Vervolgens zijn deze laatste vijf stappen vertaald naar een mogelijke flow voor de app. In figuur 1 wordt de flow van de app schematisch weergegeven. De gebruiker identificeert zijn tafel en ontvangt zijn rekening. Hierna heeft de gebruiker de mogelijkheid om de rekening te splitsen met zijn tafelgenoten en daarna de rekening te betalen of meteen na ontvangst de rekening te betalen.



Figuur 1: Schematische voorstelling van de flow van de app

Uit deze voorstelling worden de drie onderdelen van de opdracht duidelijk. De gebruiker moet de tafel kunnen identificeren. Nadat de gebruiker de tafel geïdentificeerd heeft, moet deze de rekening kunnen ontvangen op de smartphone. Vervolgens moet de gebruiker de rekening kunnen betalen zonder de ober of een betaalterminal.

Om de tafel te identificeren is gekeken naar de volgende technieken: QR-codes en OCR. Op de tafel wordt een QR-code aangebracht die de gebruiker met de app kan scannen of de gebruiker kan een foto maken van het tafelnummer en via OCR kan de info uit de foto gehaald worden.

De rekening ontvangen op de smartphone kan op twee manieren: met of zonder gebruik van het kassaticket. De eerste manier is de beste voor de opdracht, omdat de ober hiervoor niet nodig is. Als voor deze optie gekozen wordt, is het noodzakelijk om de toepassing te verbinden met het kassasysteem. De tweede manier is minder ideaal, omdat de klant nog steeds de rekening moet vragen aan de ober. De klant kan dan een foto nemen van het kassaticket en de app maakt het mogelijk via OCR de nodige gegevens van het kassaticket te lezen. Het betalen van de rekening verloopt wel nog zonder tussenkomst van de ober. Aangezien de eerste optie de beste is, wordt er verder met een kassasysteem gewerkt om de rekening te ontvangen.

Om de rekening te betalen moet de app in staat zijn om geld van de gebruiker te ontvangen en door te sturen naar de rekening van het restaurant. In de tekst wordt eerst beschreven hoe dit verwezenlijkt kan worden. De gevonden oplossing maakt gebruik van een *payment service provider* die ondersteuning biedt voor marktplaatsen. *PSP*'s zorgen onder andere voor de uitvoering van banktransacties en dat er geen kaartgegevens via de bedrijfsserver doorgegeven moeten worden. Op deze manier is het betaalproces van de applicatie beveiligd. De ondersteuning voor marktplaatsen betekent dat het bedrijf geld kan ontvangen en doorsturen naar geconnecteerde accounts.

3.2 Aanpak

Voor deze opdracht moest er een *proof-of-concept* gemaakt worden. De ideale aanpak van de opdracht is om eerst een onderdeel te onderzoeken, bijvoorbeeld wat de beste manier is om de tafel te identificeren, en vervolgens die oplossing te implementeren in de app. Het eerste onderdeel was het identificeren van de tafel. Er is een prototype uitgewerkt met een QR-codescanner en een prototype waarbij de tafelnummer gelezen werd aan de hand van OCR. Hierna werden er verschillende testen uitgevoerd en is er gekozen om de QR-codescanner te gebruiken. Deze testen worden in het onderzoek verder besproken. Vervolgens werd er gekeken naar het betalingsproces. Uit de analyse is gebleken dat de app geld moet kunnen ontvangen van gebruikers en dat een restaurant dit geld moet kunnen ontvangen met de applicatie als tussenpersoon. Stripe Connect en Mollie Connect kunnen dit probleem oplossen. Aangezien Mollie Connect nog in de ontwikkelingsfase zit wat betreft uitbetalingen aan derden en er nog niet voldoende informatie voorhanden is, is er gekozen om meteen Stripe Connect te implementeren om zo meer te leren over *payment service providers*. Tijdens het implementeren van Stripe Connect heb ik een andere PSP gevonden die wel voldoende documentatie heeft, namelijk Adyen MarketPay. Voor het maken van de vergelijking is met deze PSP verdergegaan. Voor het ontvangen van de rekening op de smartphone is de integratie met een kassasysteem noodzakelijk. Er werden aanvragen gestuurd naar verschillende makers, zoals Lightspeed, Untill en Gastrofix. Uiteindelijk heeft Untill toegang gegeven tot de *api* en met dit kassasysteem is verder gewerkt. Wegens de lange wachttijd tussen het aanvragen van toegang tot de *api* en het krijgen van de toegang is er ook gekeken naar andere manieren om het kassaticket te ontvangen. Specifiek is er gekeken om met OCR alle gegevens van een rekening te halen. Dit wordt verder toegelicht in punt 3.3.

3.3 Implementatie QR-codescanner en integratie met Untill

Voor de implementatie van de QR-scanner wordt gebruikgemaakt van het framework AVFoundation. Dit framework heeft de nodige klassen om een beeldopname te beginnen en een actie uit te voeren zodra er een QR-code gedetecteerd is. Indien de opgeslagen tekst in de QR-code het correcte formaat heeft, kan het tafelnummer uit de code gehaald worden. Vervolgens wordt er een *request* gedaan naar het kassasysteem om de rekening op te vragen van de gescande tafel. De volgende informatie is nodig om de *request* kunnen uit te voeren: de tafelnummer en het tafeldeel. Deze gegevens zijn ook nodig om de rekening af te kunnen sluiten in het kassasysteem.

Om de gegevens van een rekening te ontvangen wordt geïntegreerd met een kassasysteem. De toegang verkrijgen tot de *api* van een kassasysteem verliep echter niet gemakkelijk. Daarom is er tijdens de stage ook geëxperimenteerd met OCR om de gegevens van een rekening op de smartphone te verkrijgen. Er wordt een foto gemaakt van de rekening en de OCR zet dit om naar tekst. In deze tekst staat onder andere de naam en het adres van het restaurant, het tafelnummer, de producten, de controlegegevens, enz. Voor deze implementatie moeten de producten, per product het aantal en prijs per stuk eruit gehaald worden. De moeilijkheid bestaat er in een algoritme schrijven dat deze gegevens correct uit de data haalt. Verschillende kassasystemen printen hun kassatickets anders af. Ze hebben allemaal een andere lay-out. Figuren 2 en 3 zijn voorbeelden van verschillende lay-outs.

Schaliehuys
Dorpsplein 11 bus 1
3830 Wellen

REK#9236 13/04/2019 22:20
000000#5096 Patron 1 [0014]
Tafel9

BTW KASTICKET

Vorig saldo		0.00
4x Glas wit huis	4.00	16.00 A
2x KL macaroni	12.00	24.00 B
2x Koffie	2.30	4.60 A
Subtotaal		44.60
(A)TI.21% 20.60	BTW 21%	3.58
NET 21% ST 17.02		
(B)TI.12% 24.00	BTW 12%	2.57
NET 12% ST 21.43		

Cash 44.60

ASHA0021000236/ Ver1.66FB

CONTROLE GEGEVENS:
2019/04/13;21:43:07;94ADDCF0
000035306/000119323; NS
59de5babc1e7d196ba8b;b5c97529f57326625f1d
RET05032669;0465490627-001

Bedankt en tot ziens!
Tel.: 012/74.14.47
BE 0465.490.627

Figuur 2: Kassaticket Schaliehuys

Via Media 1
3500 Hasselt
BE0456479723
Tel:011/75.34.35

Rekening # 341498 | Tafel 17
Davina | 06/04/2019 00:12:03

2 WW Glas [4.00]	8,00 A
1 TAP Grieks Schoteltje	19,00 B

Totaal EUR 27,00

Cash 27,00

	incl.	excl.	BTW
A 21 %	8,00	6,61	1,39
B 12 %	19,00	16,96	2,04
	27,00	23,58	3,42

Controlegegevens :
05/04/2019 23:57:37
Ticketteller : 338943/1168314 NS
Ticket handtekening :
2e642b1d4b793b25c6d6b15932927a588c1790a3
FDM-id : RET05040930
VSC-id : 0456479723-001

PLU-Hash : 3e5b8e0f
Software : R-POS versie 6.0.247
Prod.nr. : BKRS001010511A
Terminal : kco1 (00-E0-0D-90-21-7B)

Bedankt en tot ziens!

Figuur 3: Kassaticket 5th Avenue

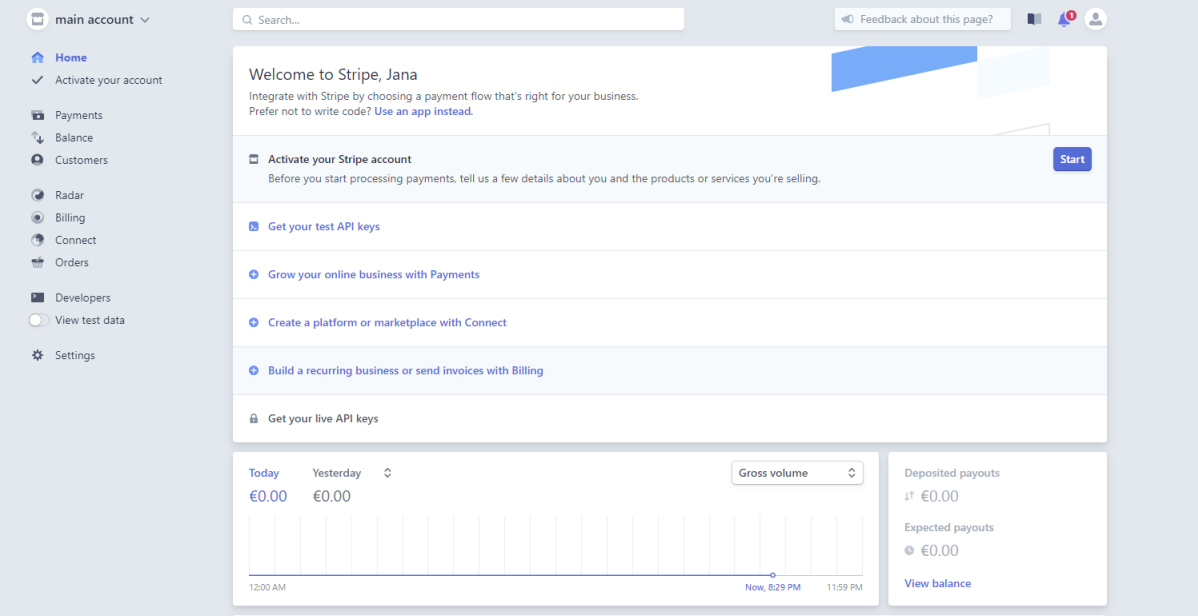
Op het eerste kassaticket wordt de hoeveelheid met een getal gevolgd door een ‘x’ voorgesteld, maar op het tweede ticket wordt er enkel een getal weergegeven. De prijs per stuk wordt in beide tickets ook anders aangeduid. Op het eerste ticket staat de prijs per stuk zonder haken en op het tweede ticket tussen vierkante haken. Het algoritme moet met zoveel mogelijk verschillende kassatickets rekening houden. Het resultaat was een app waarmee een gebruiker een foto kan nemen van zijn kassaticket. Uit deze foto werden de verschillende bestelde items gehaald en een gebruiker kon deze items splitsen met zijn tafelgenoten en de rekening betalen. Deze oplossing is niet optimaal, omdat er nog steeds een ober nodig is en het omzetten van foto naar tekst verliep zelden zonder fouten. Als hetzelfde kassaticket omgezet wordt naar tekst dan is het 21 keer van de 100 correct omgezet.

Untill is een softwarebedrijf dat gespecialiseerd is in horecabedrijven, zoals restaurants, hotels en bars. Voor ontwikkelaars is het mogelijk om integraties te maken voor deze systemen door middel van hun *api.* Om toegang te krijgen tot de *api* en bijbehorende testomgeving moet er een *Non-disclosure Agreement* (NDA) ondertekend worden. Omwille van deze NDA wordt er in dit eindwerk niet dieper ingegaan op de concrete implementatie. De *api* voldoet aan de volgende vereisten: het is mogelijk om de bestelgegevens van een tafel op te vragen en te laten weten dat de rekening van een tafel betaald is. Deze opties waren nodig voor de implementatie. In de app wordt het splitsen van de rekening als volgt opgelost: een gebruiker kan tussen zijn contacten kiezen met wie hij eet en vervolgens de items tussen deze mensen verdelen. Vervolgens kan er een bericht naar de contacten gestuurd worden met het door hun te betalen bedrag. De Untill-*api* heeft de mogelijkheid om een rekening in delen te betalen. Iedereen zou zijn eigen deel kunnen betalen. Hier is tijdens deze stage niet verder op ingegaan, maar zou een goede uitbreiding zijn voor de app.

3.4 Implementatie Stripe Connect

De implementatie van Stripe Connect bestaat uit twee delen. De applicatie moet geld ontvangen van de gebruiker en vervolgens dit geld doorsturen naar het restaurant. Om dit te verwezenlijken wordt gebruikgemaakt van de Stripe-*api*. De backend van de applicatie communiceert met de Stripe-*api*. De backend is geschreven in het PHP-framework Laravel.

Om te beginnen moet de backend verschillende publieke en private sleutels hebben om gebruik te kunnen maken van de Stripe-*api*. Deze sleutels worden bekomen door een Stripe-account aan te maken. Figuur 4 toont het beginscherm van het Stripe dashboard.



Figuur 4: Stripe dashboard

De gebruiker moet eerst aan de Stripe-*api* een persoonlijke sleutel vragen. Figuur 5 toont de backend code om een *ephemeral key* te genereren. De Stripe-versie is de enige verplichte parameter. Het *id* van de gebruiker wordt meegegeven, zodat Stripe voor alle verdere betalingen aan de hand van de *ephemeral key* weet over welke gebruiker het gaat.

```
$key = \Stripe\EphemeralKey::create(
    ["customer" => $customerStripeId],
    ["stripe_version" => $apiVersion]
);
```

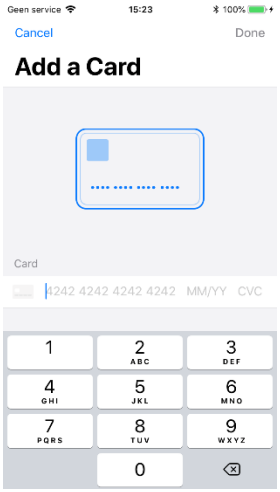
Figuur 5: code om ephemeral key te genereren

Dit is te zien in figuur 6. Aan de methode wordt de *source*-parameter meegegeven, deze wordt gebruikt om de correcte gebruiker te identificeren. De parameter *stripe_account* is het Stripe-account van het restaurant waarnaar het geld doorgestuurd moet worden en de parameter *application_fee_amount* is het bedrag dat de applicatie vraagt om de transactie uit te voeren.

```php
public function pay($source, $amount, $restaurantName)
{
    Stripe::setApiKey(env('STRIPE_SECRET'));
    $charge = \Stripe\Charge::create([
        "amount" => $amount,
        "currency" => "eur",
        "source" => $source,
        "application_fee_amount" => 123,
    ], ["stripe_account" => $this->restaurants[$restaurantName]]);
}
```

Figuur 6: Code om een betaling uit te voeren

De Stripe iOS SDK heeft zowel de nodige componenten, als een aangename gebruikersinterface, om de kaartgegevens van een gebruiker te vragen. In figuur 7 wordt een voorbeeld gegeven van de gebruikersinterface van Stripe. In dit voorbeeld kan de gebruiker zijn kaartgegevens ingeven.

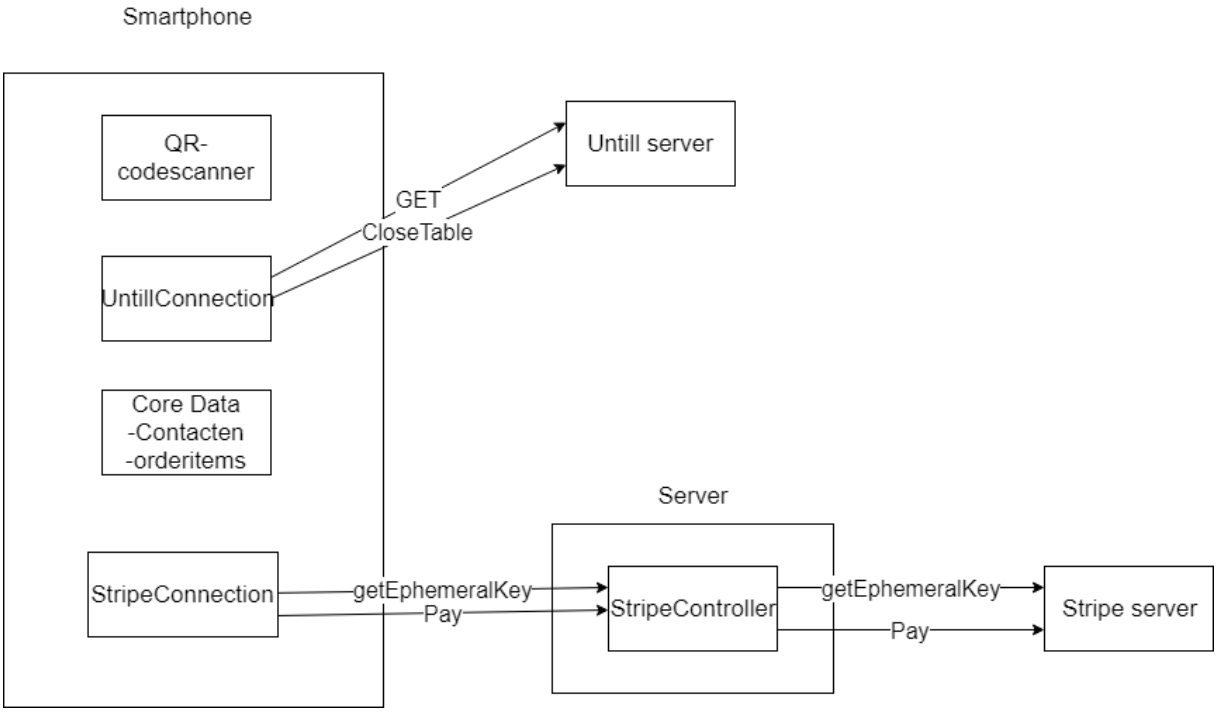


Figuur 7: Screenshot van de Stripe gebruikersinterface

3.5 Resultaten

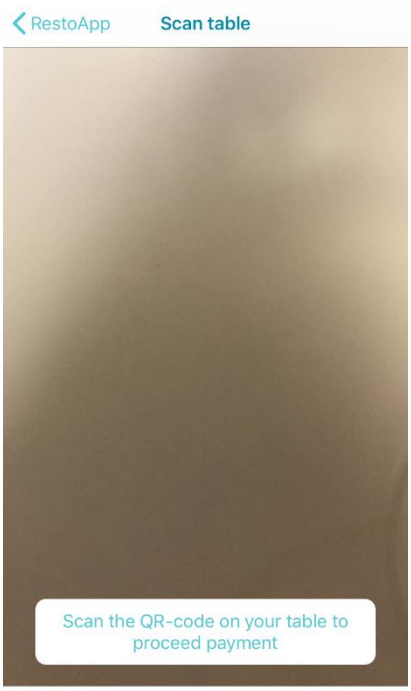
Het opgeleverde product is een *proof-of-concept* waarbij een gebruiker door middel van een QR-code te scannen zijn tafel kan identificeren. Vervolgens wordt na deze identificatie de rekening van de tafel via een *api request* opgehaald en getoond op de smartphone. Hierna kan een gebruiker kiezen om de rekening te splitsen tussen zijn tafelgenoten of meteen de rekening te betalen. Als de rekening gesplitst moet worden, kan de gebruiker nu uit zijn contacten selecteren met wie hij aan tafel zit. Vervolgens kan de gebruiker de producten naar de correcte persoon slepen. Hierna opent een scherm waar voor iedere persoon te zien is hoeveel hij moet betalen. De gebruiker kan nu naar iedereen een bericht sturen met het bedrag. Vooraleer de gebruiker betaalt, kan hij een kaart kiezen om mee te betalen of eerst een kaart toevoegen. Vervolgens kan de gebruiker betalen. In deze app moet de gebruiker van de app de volledige rekening betalen. Een goede uitbreiding zou zijn om in het bericht naar de contacten een QR-code te steken waarmee de anderen meteen kunnen terugbetalen.

In figuur 8 wordt een overzicht getoond van de applicatie. De iOS-app bevat een klasse om een QR-code te scannen. Uit de QR-code wordt de tafelnummer gehaald en deze wordt doorgegeven aan de klasse die de connectie met de server van Untill afhandelt. Deze klasse kan een aanvraag uitvoeren om de rekening te ontvangen en kan de openstaande rekening van een tafel afsluiten. Er wordt in de app gewerkt met Core Data om de gekozen contacten en bestelde items bij te houden en aan elkaar te linken. De twee entiteiten zijn 'Contacten' en 'OrderItems'. De laatste klasse zorgt voor de connectie met de backend.

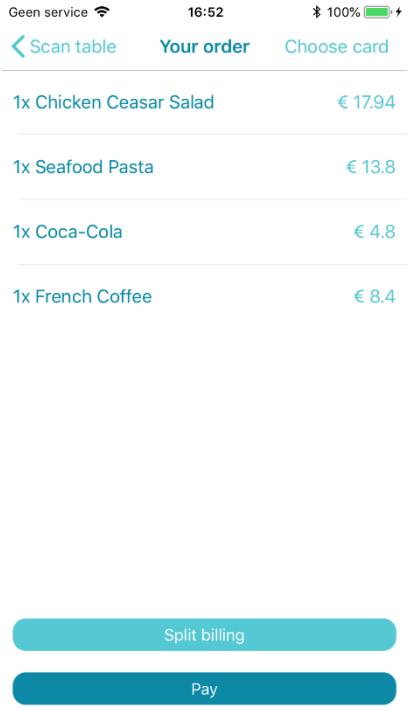


Figuur 8: Overzicht architectuur van de applicatie

De backend handelt de connectie met Stripe af. Voor iedere nieuwe gebruiker wordt een Stripe *customer*-account aangemaakt. De gegenereerde id wordt opgeslagen in een database zodat wanneer de gebruiker de app opnieuw gebruikt zijn kaartgegevens opgehaald kunnen worden bij Stripe. De backend vraagt voor gebruikers ook de *ephemeral key* aan en handelt de betaling af. Deze *key* geeft de SDK de toestemming om gedurende een sessie een specifiek *customer*-object op te halen en aan te passen. In de volgende figuren wordt de flow van de app getoond. Figuur 9 toont de QR-codescanner met een beetje informatie voor de gebruiker. In figuur 10 worden de bestelde items van een tafel getoond.

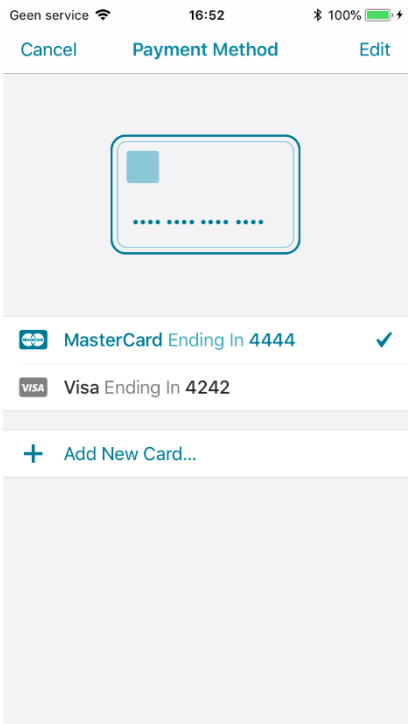


Figuur 9: QR-codescanner

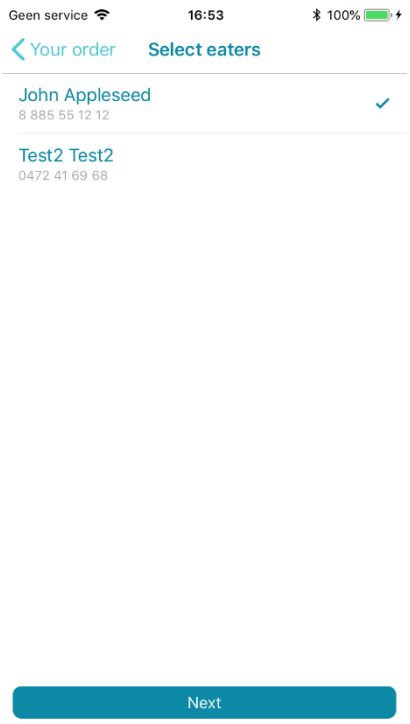


Figuur 10: Visuele voorstelling bestelde items

Wanneer gebruikers op *choose card* klikken, wordt een *view* getoond om een kaart te kiezen zoals getoond in figuur 11. Dit scherm is een standaardscherm van Stripe voor iOS. Een gebruiker kan ook de rekening splitsen. In figuur 12 wordt getoond hoe dit scherm eruit ziet. Een gebruiker krijgt een lijst met al zijn contacten te zien en kan hieruit selecteren met wie hij aan tafel zit.

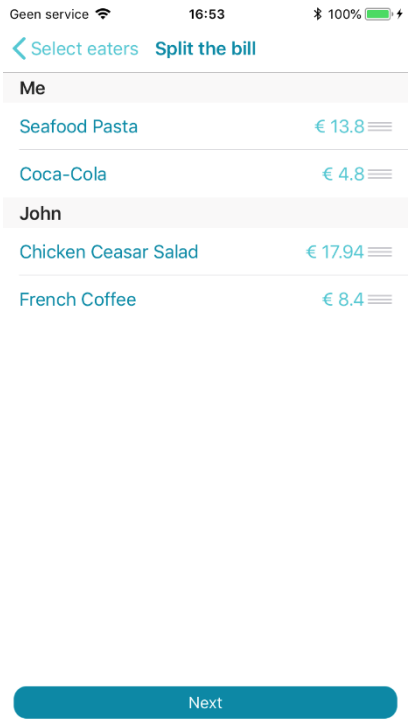


Figuur 11: Stripe UI om een kaart te kiezen

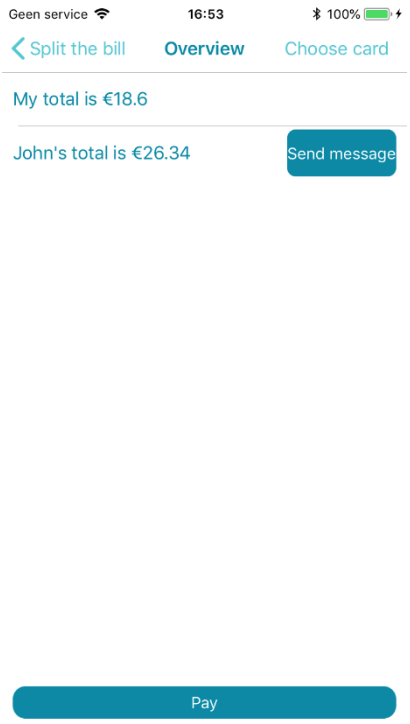


Figuur 12: Keuze tafelgenoten

Vervolgens kunnen de verschillende items tussen de verschillende contacten verdeeld worden, zoals te zien in figuur 13. Voordat de klant de rekening kan betalen wordt eerst nog een overzicht getoond waarbij de gebruiker ervoor kan kiezen om een bericht te sturen naar zijn tafelgenoten met daarin het bedrag dat zij moeten betalen. Dit wordt getoond in figuur 14.

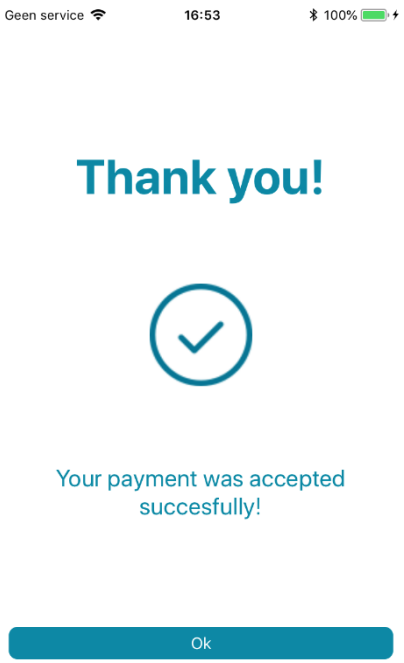


Figuur 13: Splitbilling



Figuur 14: Overzicht-scherm

In figuur 15 wordt het scherm getoond dat gebruikers zien na het succesvol betalen van de rekening.



Figuur 15: Betaling goedgekeurd

4 Besluit

Tijdens deze stage ben ik in contact gekomen met een nieuwe programmeertaal en verschillende nieuwe technologieën. Ik heb leren programmeren in Swift en leren werken met het PHP-framework Laravel. Ik ben in het begin van mijn stage veel bezig geweest met het leren van Swift. Hierbij heb ik naar enkele cursussen gekeken en veel kleine projecten gemaakt om verschillende technieken aan te leren. Ik kon ook altijd hulp vragen aan verschillende ervaren programmeurs in het bedrijf. De opleiding heeft mij hierbij geholpen, omdat tijdens de lessen mij de basisprincipes van het OO-programmeren aangeleerd zijn. Er lag ook een nadruk op zelfstudie. In de app moesten ook betalingen afgehandeld worden en zo ben ik in aanraking gekomen met *payment service providers*. Dit was helemaal nieuw voor mij. Er moest ook geïntegreerd worden met een kassasysteem. Hiervoor heb ik contact moeten opnemen met verschillende makers van kassasystemen. Dit was nieuw voor mij en ik moest hiervoor ook uit mijn comfortzone komen. De ondervonden problemen hadden meestal te maken met het programmeren. Door de nieuwe taal en technologieën gebeurde het regelmatig dat ik vast zat. Door veel zoekwerk op het internet en *trial-and-error* kwam ik altijd tot een oplossing. Zoals eerder vermeld heb ik deze stage met een nieuwe programmeertaal en nieuwe technologieën leren werken. Doordat ik Swift niet kende, voelde ik mij in het begin een beetje overweldigd. Ik heb vervolgens Swift geleerd in functie van de app die ik moest maken. Eerst heb ik mij gefocust op de basisprincipes van Swift en vervolgens leerde ik de onderdelen die nodig waren voor de app, zoals AVFoundation. Waar ik persoonlijk het meeste moeite mee had, was om hulp te vragen. Ik vind dit heel moeilijk, omdat ik bang ben om domme dingen te vragen. Als ik eerder iets durf te vragen, zouden sommige problemen sneller opgelost kunnen worden. Aan de andere kant zet ik door. Ik blijf volhouden tot iets werkt. Wanneer ik iets oplos geeft dit ook veel voldoening. Het eindresultaat is een iOS-app waarmee gebruikers een QR-code scannen om hun rekening te ontvangen. Vervolgens kunnen de gebruikers de rekening betalen en eventueel splitsen met tafelgenoten.

II. Onderzoekstopic

1 Probleemstelling

In dit project wordt een applicatie gemaakt, waarbij klanten van een restaurant automatisch hun rekening kunnen vragen, splitsen en betalen zonder tussenkomst van een ober. Er zijn enkele obstakels die de automatisatie van dit proces bemoeilijken. Klanten moeten in staat zijn om hun tafel te identificeren. Vervolgens moeten klanten hun rekening op hun smartphone kunnen ontvangen. Ook moeten klanten in staat zijn om hun rekening te betalen zonder tussenkomst van een ober.

Voor de identificatie van tafels worden twee technologieën onderzocht, namelijk *quick response*-codes (QR) en *Optical Character Recognition* (OCR). Hierbij wordt onderzocht welke oplossing het snelst, het meest efficiënt en het meest gebruiksvriendelijk is.

Om de rekening op de smartphone te kunnen ontvangen, is het nodig om de applicatie te integreren met het kassasysteem. Er wordt gebruikgemaakt van de *api* van Untill Kassasystemen. Hierbij worden de volgende punten onderzocht: de snelheid en robuustheid van de *api*. Er wordt ook nagegaan of er mogelijke problemen kunnen optreden wanneer klanten hun rekening via een app kunnen opvragen.

Voor de betalingen wordt een vergelijking gemaakt tussen twee *payment service providers*, namelijk Stripe Connect en Adyen MarketPay. De twee providers worden vergeleken op de volgende criteria: prijs, snelheid van uitbetalen, aantal mogelijkheden en veiligheid.

2 Onderzoeksmethode

Om de tafel te kunnen identificeren wordt er gekeken naar twee technologieën, namelijk QR-codes en OCR. Er wordt bepaald welke de meest optimale oplossing is aan de hand van de volgende testen. De snelheid van de oplossing wordt bepaald door de tijdsduur tussen het begin van de scan en het einde van de scan. De gebruiksvriendelijkheid van de oplossing wordt gecontroleerd door middel van *hallway testing*. Een *Hallway test* wordt opgezet op een drukke plaats en er wordt aan voorbijgangers gevraagd om het product te testen. De testen vinden steeds plaats in een restaurant, zodat de testers in de correcte omgeving zitten. De testers proberen de oplossing met de QR-code en met OCR en krijgen vervolgens na iedere technologie een kleine vragenlijst. Daarna wordt aan iedere tester de geprefereerde manier gevraagd. Om de efficiëntie van de oplossingen te testen, wordt er gekeken naar het energieverbruik. Hiervoor wordt de XCode Debugger Tool gebruikt.

Om de rekening op de smartphone te ontvangen wordt er geïntegreerd met de *api* van Untill. Er wordt bepaald hoe snel de integratie is door de tijdsduur te meten tussen het verzenden van de *request* en het ontvangen van het antwoord. Vervolgens wordt bepaald hoe robuust de integratie is door te kijken wat het effect is van verschillende *requests* tegelijk te versturen, gedurende één minuut iedere seconde een *request* te versturen en een rekening op te vragen met veel producten op. Deze testen worden uitgevoerd door gebruik te maken van Apache JMeter. Ten laatste wordt ook bekeken wat mogelijke bemerkingen zouden kunnen zijn op de integratie.

Om de rekening via de app te kunnen betalen wordt er gebruikgemaakt van een *payment service provider*. In dit onderzoek wordt er gekeken naar Stripe Connect en Adyen MarketPay. De vergelijking tussen Stripe Connect en Adyen MarketPay wordt gedaan aan de hand van een vergelijkingsmatrix. In de matrix worden de volgende parameters opgenomen: kost, snelheid van betalen, verschillende mogelijkheden en veiligheid.

3 Uitwerking onderzoek

3.1 QR-code

Een QR-code is een tweedimensionale, vierkante barcode waarin gecodeerde data opgeslagen wordt [1]. Deze data kan verschillende types hebben, zoals numerieke en alfabetische karakters, binair, Kanji of Kana [2]. Deze data wordt uit de QR-code gehaald met behulp van een QR-code scanner. Een eventueel bijbehorende actie wordt uitgevoerd, zoals het openen van een website [3].
QR-codes kunnen uit iedere hoek gescand worden in tegenstelling tot een barcode die horizontaal gescand moet worden [4].

In figuur 16 worden de verschillende vaste onderdelen van een QR-code aangeduid.

Figuur 16: Verschillende onderdelen van een QR-code [5]

De drie paarse zones duiden de rand aan van de QR-code aan. In de twee oranje strepen zijn afwisselende witte en zwarte blokjes te zien. Deze geven aan waar de kolommen en rijen van witte en zwarte blokjes zijn. De informatie over het type data dat zich in de QR-code bevindt, wordt opgeslagen in de vier rode strepen. Enkele mogelijke data types zijn een URL of numerieke informatie. De twee groene zones geven de versie van de QR-code aan [5]. Iedere versie heeft een ander aantal modules. Een module is een wit of zwart blokje waaruit de QR-code bestaat [6]. Deze zones tref je aan in QR-code versies 6 tot en met 40. In oudere versies hebben scanners deze informatie niet nodig, omdat ze die kunnen afleiden van andere zones. De gele zone dient als de marker voor de uitlijning. Grotere QR-codes kunnen meerdere van deze markers hebben om de nauwkeurigheid te verbeteren [5].

In tabel 1 worden de verschillende types van QR-codes opgelijst.

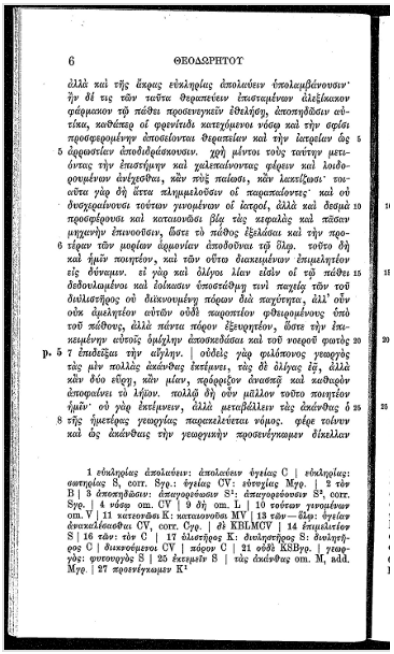
Naam	Uitleg	# cijfers dat het kan coderen	Maximum versie
QR-code Model 1	De originele code	1167	14 (73x73 modules)
QR-code Model 2	Verbeterde versie. De QR-code kan ook gelezen worden wanneer de code vervormd is.	7089	40 (177x177 modules)
Micro QR-code	Deze code heeft slechts één marker om de positie te detecteren. De marge rond de code moet een breedte hebben van twee modules, in tegenstelling tot een gewone QR-code met een marge van vier modules.	35	M4 (17x17 modules)
iQR-code	Deze code kan meer data bevatten dan een gewone QR-code. Er kunnen kleine codes gegenereerd worden. Rechthoekige vormen zijn ook mogelijk. Ze hebben een hogere herstelcapaciteit.	± 40.000	61 (422x422 modules)
SQRC	*Secret-function-equipped QR code*. Deze codes kunnen publieke en private data bevatten. De private data is veilig, omdat deze enkel gelezen kunnen worden aan de hand van een cryptografische sleutel.	Idem QR-code Model 2	Idem QR-code Model 2
Frame QR	Deze QR-code heeft een frame waarin een afbeelding toegevoegd kan worden.	Idem QR-code Model 2	Idem QR-code Model 2

Tabel 1: Verschillende types QR-code [7]

3.2 OCR

Optical Character Recognition is het proces waarbij een afbeelding wordt omgezet naar tekst. Enkele voorbeelden van het gebruik van OCR zijn het omzetten van handgeschreven tekst naar digitale tekst en het automatisch herkennen van nummerplaten [8].

In dit project is gebruikgemaakt van de Tesseract OCR van Google. Deze OCR ondersteunt standaard 100 talen en kan voor andere talen getraind worden. Voordat een tekst gebruikt kan worden voor OCR moet deze geoptimaliseerd worden [9]. Tesseract maakt gebruik van de *library* Leptonica om de afbeelding te verwerken voordat deze aan de OCR gegeven wordt. Er zijn nog enkele manieren waarop een gebruiker zijn afbeeldingen kan optimaliseren. Wanneer er te veel verschillende kleuren en lichtschakeringen in de afbeelding zitten, moeten deze weggewerkt worden. De tekst op de afbeelding moet zo recht mogelijk liggen. Verder mag er geen zwarte rand rond de tekst zijn, omdat de OCR dit als een letter gaat proberen te interpreteren. In figuur 17 wordt een voorbeeld getoond van een tekst met een te grote zwarte rand [10].



Figuur 17: Voorbeeld van een tekst met een te grote zwarte rand [10]

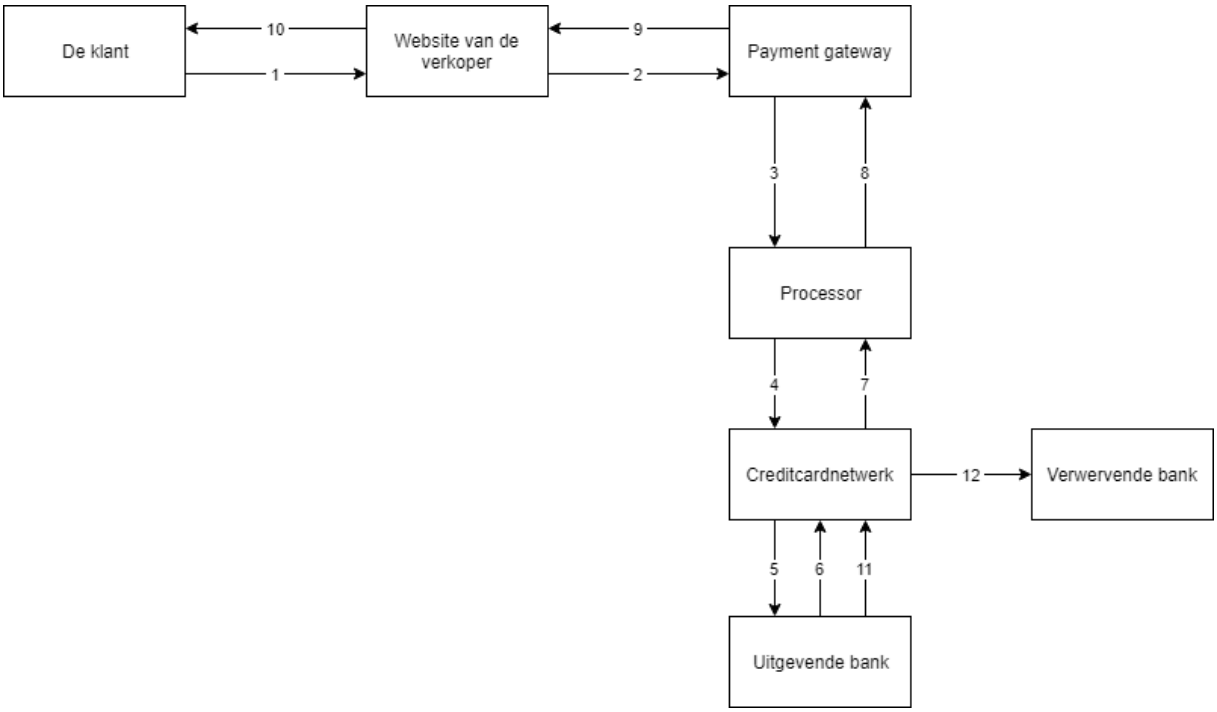
Tijdens het OCR-proces moeten de letters individueel uit de tekst gehaald worden. Dit gebeurt door middel van een algoritme dat alle zwarte objecten, die omringd zijn door wit, behandelt als een letter. Vervolgens moet zo'n patroon vertaald worden naar een letter [8].

3.3 Het betalen zonder tussenkomst van een ober

Om de rekening te betalen wordt gebruikgemaakt van een *payment service provider*. Eerst wordt het algemene betalingsproces uitgelegd. Vervolgens worden de twee providers, Stripe Connect en Adyen MarketPay, toegelicht net als de manier waarop de integratie met deze systemen het betalingsproces voor de gebruiker vergemakkelijkt.

3.3.1 Het online betalingsproces

In figuur 18 wordt een algemene voorstelling getoond van het online betalingsproces. In deze voorstelling wordt gebruikgemaakt van een creditcard.



Figuur 18: Schematische voorstelling online betalingsproces [11]

1. De klant voert zijn kaartgegevens in en betaalt zijn aankopen via de website van de verkoper.
2. De verkoper verzendt de transactie naar een *payment gateway* .
3. De *payment gateway* encrypteert de gegevens en verzendt deze naar de processor.
4. De processor verzendt de transactie naar het creditcardnetwerk.
5. Het creditcardnetwerk verzendt de gegevens naar de uitgevende bank. Dit is de bank van de klant.
6. De uitgevende bank controleert de gegevens en bepaalt of de klant genoeg geld heeft. Indien de transactie goedgekeurd wordt, verzendt de bank een bericht dat de transactie geautoriseerd is, anders dat het geweigerd is [12].
7. Het creditcardnetwerk verzendt de goedkeuring of weigering naar de processor.
8. De processor verzendt het antwoord naar de *payment gateway*.
9. Het antwoord wordt naar de website van de verkoper verstuurd.
10. De klant krijgt een melding of de betaling geaccepteerd of geweigerd is.
11. De uitgevende bank verzendt het geld naar het creditcardnetwerk.
12. Het creditcardnetwerk verzendt het geld naar de verwervende bank.

3.3.2 Payment service providers

Payment service providers zorgen ervoor dat handelaren betalingen met creditcard of debitcard kunnen ontvangen. De providers zijn de link tussen de handelaren en het elektronische, financiële systeem [13]. In dit onderzoek wordt gekeken naar twee *payment service providers*, namelijk Stripe en Adyen.

Payments service providers moeten voldoen aan verschillende vereisten. Enkele belangrijke vereisten worden hier kort beschreven. *Know your customer (KYC)* is het proces waarbij bedrijven de identiteit van hun klanten controleren. De bedoeling van KYC is om illegale activiteiten, zoals witwaspraktijken en corruptie, te voorkomen. Door klanten te controleren, weten bedrijven dat hun zaken legaal verlopen [14]. *Second Payment Service Directive (PSD2)* is de herziening van de *Payment Service*

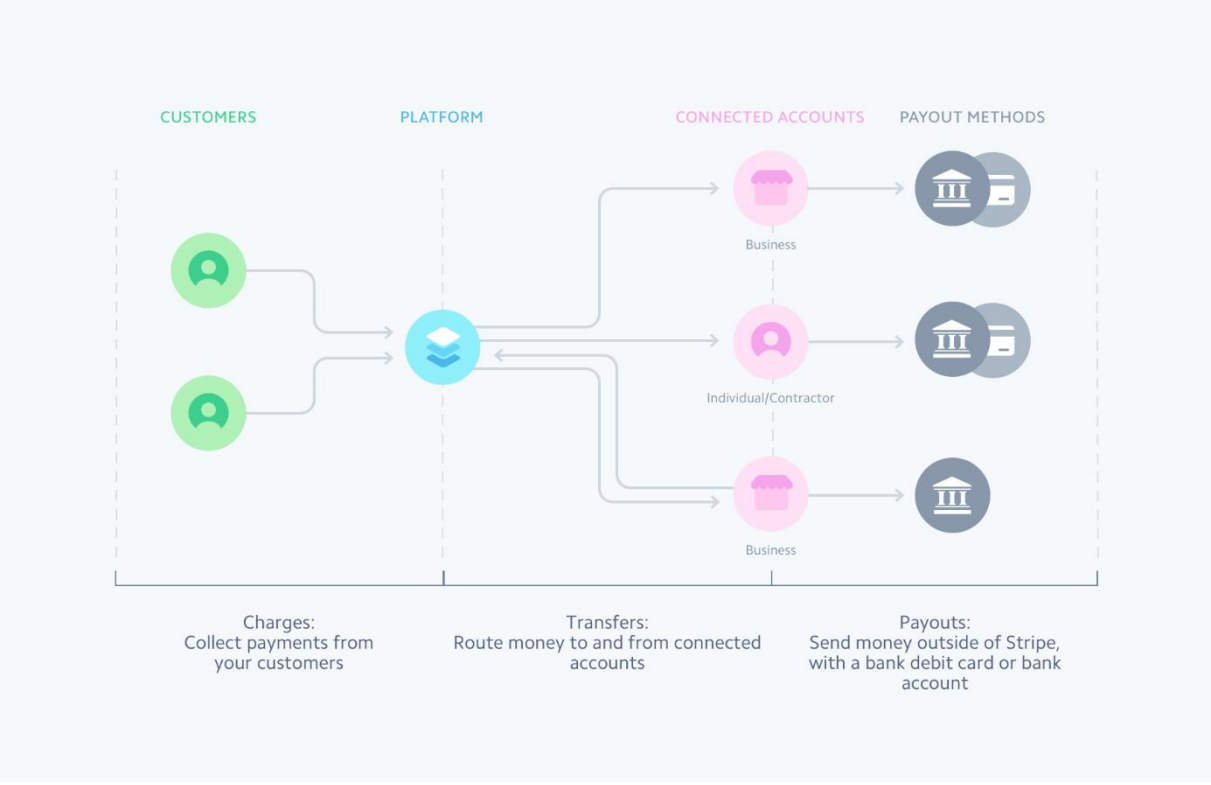
Directive (PSD). Het doel van de PSD is om grensoverschrijdend betalingsverkeer even gemakkelijk, veilig en efficiënt te maken als binnenlands verkeer. De PSD2 bevat verschillende wijzigingen en uitbreidingen [15].

Payment Card Industry Data Security Standard (PCI DSS) werd ontworpen door de *PCI Security Standards Council* om creditcardfraude tegen te gaan. Een bedrijf moet de nodige stappen ondernemen om de gegevens van de kaarthouders te beschermen, zodat deze niet gebruikt kunnen worden bij een cyberaanval. De controle of een bedrijf voldoet aan de eisen wordt gedaan door een *Qualified Security Assessor* (QSA), een *Internal Security Assessor* (ISA) of door middel van een *Self-assessment Questionnaire (*SAQ). De questionnaire is enkel mogelijk voor bedrijven met weinig kaarthoudergegevens [16].

3.3.3 Stripe Connect

Stripe Connect wordt gebruikt voor het ontvangen en het doorsturen van geld naar derde partijen. Via de *api* en bijgeleverde tools kan een ontwikkelaar de integratie personaliseren. Een gebruiker kan bijvoorbeeld betaalmethoden toekennen, kiezen hoe geld van de klant naar de gebruiker en van de gebruiker naar een derde partij gaat [17].

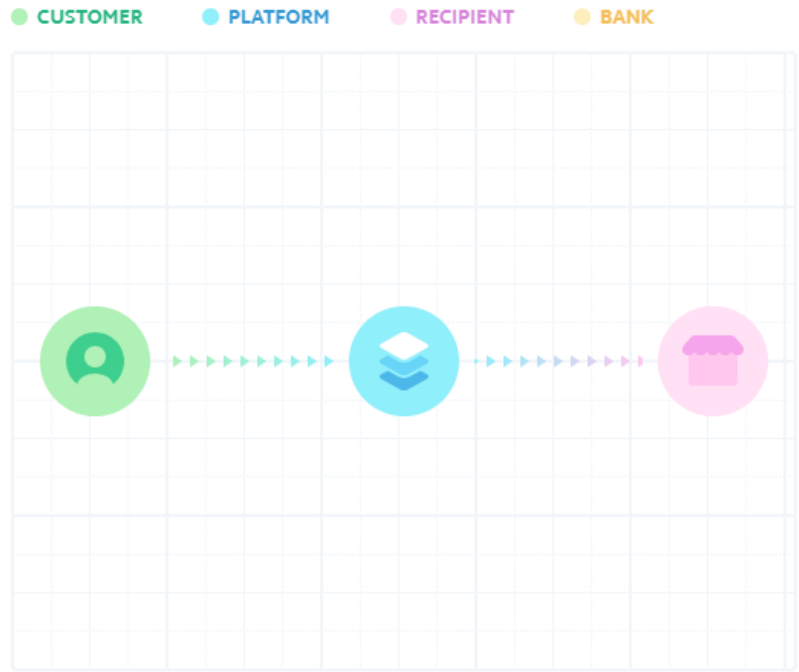
Figuur 19 toont de mogelijkheden van Stripe Connect. Met Stripe Connect is het mogelijk om betalingen te ontvangen van klanten, geld te ontvangen van geconnecteerde accounts en geld te sturen naar geconnecteerde accounts, fondsen uit te betalen aan bankaccounts en debitcards [18].



Figuur 19: Mogelijkheden Stripe Connect [18]

Om geld van klanten naar derde partijen te sturen heeft Stripe verschillende routing mogelijkheden. Enkele voorbeelden zijn *one-to-one, one-to-many, many-to-many* en *Account debits*.

In figuur 20 is een schematische voorstelling weergegeven van de route *one-to-one*, omdat deze de interessantste optie is voor dit onderzoek. Deze optie is de interessantste omdat de app geld moet ontvangen van een gebruiker en dit geld moet doorsturen naar de ontvanger, in dit geval het correcte restaurant.



Figuur 20: Schematische voorstelling one-to-one route [17]

Door Stripe te gebruiken voldoet de gebruiker meteen aan de verschillende wetgevingen. Stripe voldoet aan de vereisten van de *Payment Card Industry* (PCI) door gebruik te maken van tokens. Het gebruik van tokens zorgt ervoor dat er geen gevoelige data op de server van de gebruiker komt. De nodige betalingsgegevens worden doorgegeven aan de servers van Stripe zonder tussenkomst van de server van de gebruiker. De gegevens worden nooit volledig getoond, enkel in het Stripe dashboard wordt een beetje informatie getoond. Het gaat hier over de laatste vier getallen van de kaart, het merk van de kaart en de vervaldatum [19]. De systemen van Stripe om gegevens van gebruikers te monitoren en verifiëren zorgen voor de vereisten van KYC [17].

3.3.3.1 Het Stripe Connect proces

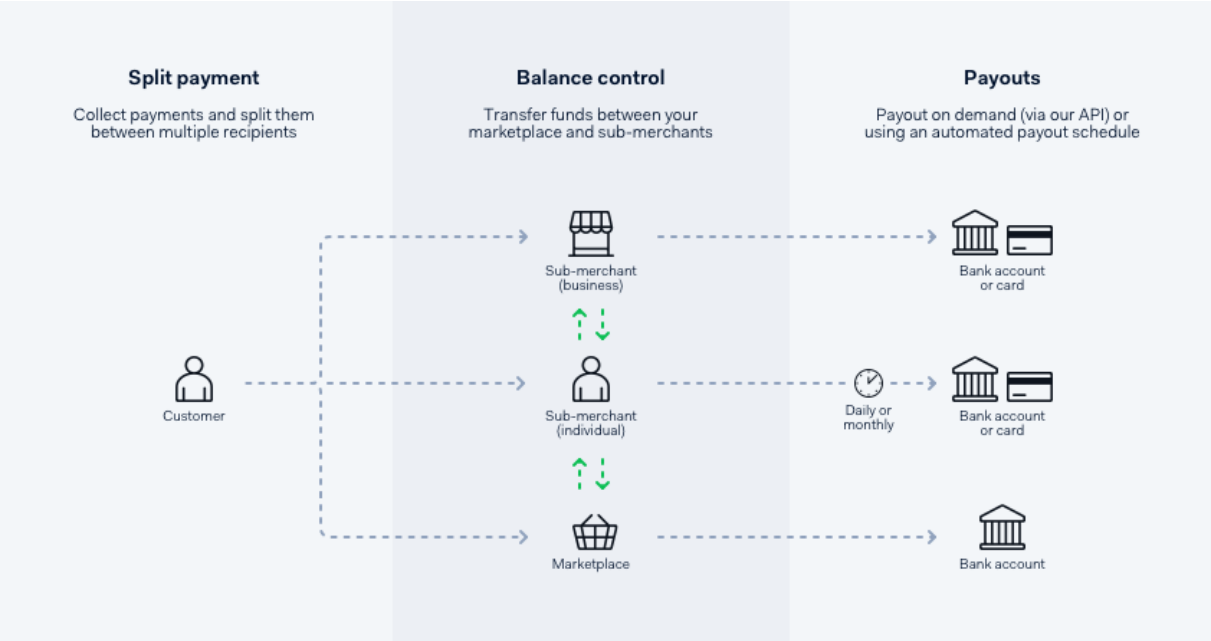
In figuur 21 wordt de integratie van Stripe in een applicatie weergegeven.

1. Door middel van de Stripe-*api* worden de betalingsgegevens vanuit de app naar de Stripe servers gestuurd.
2. Indien de betalingsgegevens correct zijn, stuurt Stripe een token terug. Dit token wordt achteraf gebruikt voor het uitvoeren van een betaling.
3. De app stuurt het token naar de backend van de applicatie. In de backend wordt gebruikgemaakt van de Stripe-*api* om de betaling uit te voeren.
4. Het token wordt naar de Stripe backend gestuurd om de betaling uit te voeren. Stripe geeft een resultaat terug met informatie over de transactie.
5. De backend van de applicatie stuurt deze info door naar de app.

Figuur 21: Schematische voorstelling integratie met Stripe [20]

3.3.4 Adyen MarketPay

Adyen MarketPay biedt, zoals Stripe Connect, de mogelijkheid aan platformen om geld te ontvangen van klanten en uit te betalen aan derde partijen [21]. Door Adyen MarketPay te integreren in een app heeft de ontwikkelaar de controle over de volgende onderdelen: registratie en verificatie van de handelaren, splitsen van betalingen tussen handelaren, hoe en wanneer handelaren uitbetaald worden en het verzenden van geld tussen het platform en een handelaar [21]. In figuur 22 wordt de werking van Adyen MarketPay schematisch weergegeven.



Figuur 22: Schematische voorstelling Adyen MarketPay [21]

De algemene flow van de integratie van Adyen MarketPay met een app is dezelfde als die van Stripe Connect [22]. Deze is te zien in figuur 21.

3.4 Point Of Sale

Point of sale (POS) is het systeem waar een klant betaalt voor zijn goederen. Specifiek voor de situatie in een restaurant houdt dit in dat een ober kan zien hoeveel een klant moet betalen, dit aan de klant kan tonen en dat de klant kan betalen. De klant kan betalen met cash geld en de bankkaart. Het systeem kan ook een kassaticket afprinten [23]. Tegenwoordig wordt voor restaurants de afkorting POS ook vertaald naar *point of service*, omdat in een restaurant POS-systeem er meer moet gebeuren dan enkel verkopen [24]. Er worden ook rapporten bijgehouden en analyses gemaakt. Het systeem is ook in staat om zowel de inventaris als werknemers te controleren [24]. Dit betekent dat het systeem kan tonen welke gerechten het meest verkocht worden, hoeveel de gerechten opbrengen en hoeveel de arbeidskosten bedragen [25]. De systemen hebben ook de mogelijkheid tot het accepteren van cadeaubonnen [24]. Afhankelijk van de fabrikant van het systeem kan er geïntegreerd worden met integraties van derde partijen. Dit zorgt voor nog meer functionaliteit, zoals bijvoorbeeld accounting [25].

Untill is het POS-systeem dat hier onderzocht wordt. Het is een bedrijf dat gevestigd is in Nederland en heeft mogelijkheden voor de volgende branches: restaurants, bars en cafés, hotels, leisure, evenementen en take-away [26]. Aan een kassasysteem van Untill kunnen verschillende andere programma's gekoppeld worden. Er zijn uitbreidingen voor onder andere bestellen, boekhouden en loyaliteit. Deze koppelingen zijn mogelijk, omdat Untill een *api* heeft waar ontwikkelaars gebruik van kunnen maken [27].Toegang wordt enkel verleend na goedkeuring en het ondertekenen van een NDA. Omwille van deze NDA wordt er niet verder ingegaan op de *api* en de concrete implementatie hiervan. Er kan ook gekozen worden voor een ander systeem, bijvoorbeeld Lightspeed of NCR, maar deze opties hebben een strenger beleid over wie toegang krijgt tot hun *api*. Lightspeed laat enkel partners toe tot hun *api* en aanvragers worden grondig gescreend voordat hun aanvraag goedgekeurd wordt [28].

3.5 Resultaten

In de volgende secties worden de resultaten van de verschillende experimenten besproken. Als eerste worden de resultaten van de *speedtest*, de impact op het energieverbruik en de gebruiksvriendelijkheid van zowel de QR-codescanner als de OCR besproken. De gebruiksvriendelijkheid is getest door middel van een *Hallway test.* Vervolgens word de integratie met Untill besproken op basis van snelheid en robuustheid. Er wordt ook gekeken naar enkele mogelijke beperkingen van de integratie. Als laatste wordt een vergelijking gemaakt tussen Stripe Connect en Adyen MarketPay. Deze vergelijking wordt weergegeven door middel van een vergelijkingsmatrix. In deze matrix zijn de volgende punten opgenomen: prijs, snelheid van uitbetalen, veiligheid en verschillende mogelijkheden.

3.5.1 Identificatie van de tafel door middel van een QR-code

In tabel 2 worden de resultaten getoond van de *speedtest* van de QR-code. De begintijd is het moment dat de QR-codescanner opent en een code gescand kan worden. De eindtijd is wanneer de gegevens uit de QR-code gehaald zijn. De tijden worden in het formaat hh:mm:ss.ms getoond en de intervallen zijn uitgedrukt in seconden.

Poging	Begintijd	Eindtijd	Interval
1	11:34:17.1850	11:34:17.7200	0.53563
2	11:35:06.1000	11:35:06.6630	0.56263
3	11:36:01.9100	11:36:02.4610	0.55105

Tabel 2: Resultaten speedtest QR-code

De QR-code wordt altijd correct gelezen. De camera van de smartphone wordt meteen op de QR-code gericht, zodat er geen tijd verloren wordt met het zoeken van de code. Op deze manier zijn de tijdstippen zo nauwkeurig mogelijk.

De impact van de oplossing op de energie werd gemeten met de debugger van XCode. De tool geeft de impact weer op een schaal. De schaal is laag, hoog en heel hoog. In tabel 3 worden de resultaten van verschillende metingen getoond.

Poging	Impact op de energie
1	Laag
2	Laag
3	Laag
4	Laag
5	Laag

Tabel 3: Resultaten impact op energie van QR-code scannen

3.5.2 Identificatie van de tafel door middel van OCR

Aangezien in de praktijk de tafelnummer van een tafel gelezen moet worden, mag het papier waar de nummer op staat niet te groot zijn. Daarom zijn in deze experimenten de tafelnummers op papieren van 5x5cm geprint. Verder is er ook gekeken naar het effect van verschillende lettergroottes.

Hieruit bleek dat de lettergrootte belangrijk is. Indien deze niet groot genoeg is, wordt de tekst niet correct gelezen. Tekst getypt in Word met een lettergrootte van 120 werd correct gelezen. Een bemerking hierbij is dat restauranteigenaars een grote tafelnummer op de tafels moeten hangen wat niet als visueel aantrekkelijk beschouwd wordt. Er is niet geëxperimenteerd met verschillende lettertypes. Hiermee moet ook rekening gehouden worden en extra onderzoek wordt aangeraden. Zoals eerder vermeld, mag er geen zwarte rand rondom de tekst zitten. De OCR gaat deze rand

proberen te vertalen naar tekst. Uit de experimenten bleek dat als er rond het papier een schaduw was, deze schaduw ook geïnterpreteerd werd als tekst. In het algemeen werd de tekst ook niet altijd consistent vertaald. De tekst “9 a” kan vertaald worden naar “9a”, ”9 a”, “_9 a”, “__9 a” en nog andere mogelijkheden. Soms werd de tekst ook fout vertaald, zoals “120 f” naar “15f”. Het papier met de tafelnummer mag ook niet gekreukt zijn, omdat de tekst dan ook foutief gelezen wordt. Ook moest de tekst zo recht mogelijk in de foto liggen om correct gelezen te worden.

De meest waarschijnlijke oorzaak van het foutief lezen van de tafelnummer is de rand rondom het papier. Wanneer de tekst in het midden van een A4-papier staat en hier een foto van getrokken wordt, is de tekst altijd correct.

De snelheid van de oplossing wordt bepaald door het interval tussen het begin en het einde van de tekstherkenning. In figuur 23 is een voorbeeld van de output te zien.

```
Begin extracting text: 2019-04-17 11:11:01.5290
End extracting text: 2019-04-17 11:11:01.8170
Interval: 0.2882939577102661
```

Figuur 23: Een voorbeeld van de output van de speedtest

Tabel 4 toont de resultaten van meerdere experimenten. Het formaat van het tijdstip is hh:mm:ss.ms en het interval is in seconden uitgedrukt.

Poging	Begintijd	Eindtijd	Interval	Correct
1	9:45:41.4060	9:45:41.6440	0.23807	Ja
2	9:46:03.5980	9:46:03.9670	0.36953	Nee
3	9:46:30.7840	9:46:30.9620	0.17820	Ja
4	9:47:07.8540	9:47:08.1440	0.28947	Nee
5	9:47:45.3550	9:47:45.5480	0.19256	Ja
6	9:48:02.1800	9:48:02.3350	0.15435	Ja
7	9:51:29.9350	9:51:30.0830	0.14781	Ja
8	9:53:49.4090	9:53:49.5740	0.16520	Ja
9	9:54:19.0500	9:54:19.2350	0.18496	Ja
10	9:54:55.5220	9:54:55.6830	0.16013	Ja

Tabel 4: Resultaten speedtest OCR

Voor deze uitwerking is ook de impact op de energie gemeten met behulp van de XCode debugger. In tabel 5 zijn de resultaten weergegeven.

Poging	Impact op de energie
1	Laag
2	Laag
3	Laag
4	Laag
5	Laag

Tabel 5: Resultaten impact op energie van OCR

3.5.3 Resultaten gebruiksvriendelijkheid QR-code en OCR

Uit de voorbereiding van de experimenten met OCR in sectie 3.5.2. is gebleken dat indien de tafelnummer in het midden van een A4-papier staat, de tekst bijna altijd correct gelezen wordt. Daarom werd voor de gebruiksvriendelijkheid gebruikgemaakt van A4-papieren in plaats van kleinere papieren.

Voor deze test zijn er 30 willekeurige mensen uitgekozen. In een eerste fase gebruikten de testers een app dat gebruikmaakt van OCR om de tafel te identificeren. Hierna vulden ze een korte enquête in over hun ervaring. Voor de app dat gebruikmaakt van een QR-codescanner om de tafel te identificeren werden dezelfde stappen doorlopen. De gebruikers moesten hun mening aanduiden op een schaal van 1 tot 5, waarbij 1 helemaal niet akkoord is en 5 helemaal akkoord. Uiteindelijk werd er bevraagd welke optie de voorkeur genoot. In tabel 6 worden de gemiddelde scores getoond voor snelheid, duidelijkheid en aantal nodige stappen. Bij 'Voorkeur technologie' worden het aantal stemmen getoond voor beide technologieën.

	QR-code	OCR
Gebruikers ontvingen de rekening snel	4,15	4,08
Gebruikers vonden het proces duidelijk	4,04	3,85
Gebruikers vonden dat ze veel stappen moesten ondernemen	1,27	1,5
Voorkeur technologie	26	4

Tabel 6: Resultaten enquête gebruiksvriendelijkheid QR-code en OCR op een schaal van 1 tot 5

In de resultaten is te zien dat de QR-codescanner beter scoorde op alle onderdelen. Het verschil in scores tussen de QR-codescanner en OCR is echter niet groot. Er is ook aan de testers gevraagd welke technologie de voorkeur heeft. Uit deze vraag blijkt dat de grote meerderheid de QR-codescanner verkoos boven de OCR.

Er zijn enkele bemerkingen bij dit experiment. Een steekproef van 30 personen is niet heel groot. Ook is dit experiment op één dag in één specifiek restaurant uitgevoerd. Dit kan als gevolg hebben dat de steekproef niet gevarieerd is. De verdeling tussen mannen en vrouwen is in de steekproef ook niet gelijk verdeeld.

3.5.4 Integratie Untill

De snelheid van het opvragen van een rekening wordt onderzocht door een rekening aan te maken en vervolgens deze op te vragen. De tijd tussen het opvragen van de rekening met de app waarbij een verzoek verstuurd wordt naar de server van Untill en het ontvangen van de rekening wordt gemeten. De rekening bevat zes items: vier drankjes en twee maaltijden. In tabel 7 worden de resultaten getoond in milliseconden.

Poging	Interval
1	309
2	321
3	321
4	333
5	305

Tabel 7: Snelheidstest opvragen rekening Untill

Het kan ook dat er in korte tijd veel gebruikers hun rekening opvragen. Daarom wordt ook gecontroleerd wat het effect is als gedurende één minuut er iedere seconde een *request* gestuurd wordt. Figuur 24 toont een deel van de output. Uit de test blijkt dat iedere seconde een *request* sturen geen effect heeft. Gegeven de resultaten van de snelheidstest is dit de te verwachten uitkomst. Iedere *request* wordt in minder dan een seconde behandeld. Het blijven sturen van *requests* heeft geen invloed op de responstijd en de server blijft werken.

```
1
Begin extracting text: 2019-05-23 14:23:20.9720
End extracting text: 2019-05-23 14:23:20.9720
Interval: 0.470363974571228
---------------------
2
Begin extracting text: 2019-05-23 14:23:21.9730
End extracting text: 2019-05-23 14:23:21.9730
Interval: 0.3111840486526489
---------------------
3
Begin extracting text: 2019-05-23 14:23:22.9720
End extracting text: 2019-05-23 14:23:22.9720
Interval: 0.29227399826049805
---------------------
```

Figuur 24: Deel output gedurende één minuut iedere seconde een request sturen

Vervolgens wordt getest hoe het systeem reageert als er 60 *requests* gelijktijdig verstuurd worden. In een gewone restaurantflow kunnen er met één kassa niet meerdere rekeningen gelijktijdig opgevraagd worden. Als echter meerdere klanten met hun app de rekening vragen, gebeurt dit wel. Daarom wordt hier getest of dit een effect heeft. Deze test is uitgevoerd door in JMeter 60 threads gelijktijdig te starten met de *request*. Het kassasysteem kan dit aan. In JMeter komen de *responses* per twee binnen. In het kassasysteem worden de *request* asynchroon behandeld. Dit betekent wel dat de responstijden niet hetzelfde zijn voor iedere gebruiker. In dit geval moeten sommige gebruikers langer wachten op hun rekening. De laatste gebruiker moest één seconde langer wachten dan de eerste gebruiker. Dit verschil is nauwelijks merkbaar.

Als laatste wordt ook het effect van de grootte van de rekening gecontroleerd. In tabel 8 wordt de responstijd per aantal items op de rekening getoond. Uit de gegevens wordt duidelijk dat het interval groter wordt naarmate er meer items op de rekening staan. Dit is wat er verwacht werd. Wanneer de rekening uitzonderlijk groot is, 100 items, dan ontvangt een gebruiker zijn rekening na 1 seconde en 190 milliseconden. Dit is een zeer acceptabele tijd.

Aantal items op rekening	Interval
10	556
20	634
30	647
40	691
50	760
60	920
70	1050
80	1080
90	1100
100	1190

Tabel 8: Resultaten snelheidstest met verschillend aantal items op de rekening

Er is ook nagedacht over mogelijke beperkingen van het opvragen en betalen van de rekening via een applicatie die geïntegreerd is met een kassasysteem. De eerste bemerking is dat er weinig controle is op klanten. Klanten kunnen gewoon opstaan zonder te betalen. Dit is iets waar rekening mee gehouden moet worden, maar vormt ook geen essentieel probleem. In de huidige situatie van een restaurant kan een klant immers ook opstaan en weggaan zonder te betalen. Het zou ook kunnen wanneer een klant wilt betalen en zijn rekening opvraagt dat niet alle gegevens er al op staan. Dit zou kunnen als een ober dit niet meteen de orders ingeeft. De kans dat dit gebeurt is echter klein, zoals eerder vermeld zit de kracht van POS-systemen erin dat bijna alle processen geregeld worden. Wanneer een ober de bestelling doorgeeft, wordt ook automatisch de keuken verwittigd van de bestelling. De POS-systemen hebben ook in bijna alle gevallen de mogelijkheid tot mobiele oplossingen. Dit wil zeggen dat de bestellingen opgenomen worden met een mobiele toepassing, zodat de bestelling meteen in het systeem opgenomen is. De belangrijkste bemerking is dat de gebruiker van de app toegang moet hebben tot het internet. Indien de applicatie geen toegang heeft tot het internet kan er geen connectie gemaakt worden met het kassasysteem. Dan moet er terug beroep op de ober gedaan worden. Het is ook mogelijk dat een gebruiker het ticket van een andere tafel kan scannen en aan deze gegevens kan. Dit kan mogelijk gezien worden als een schending van privacy, maar klanten kunnen gewoon zien wat andere tafels aan het eten en drinken zijn. Vanuit dit perspectief is het geen probleem dat gebruikers andere tafels kunnen scannen.

3.5.5 Vergelijking Stripe Connect en Adyen MarketPay

In tabel 9 is een vergelijking te zien van Stripe Connect en Adyen MarketPay op basis van prijs, snelheid van uitbetalen, veiligheid en de verschillende mogelijkheden.

	Stripe Connect	Adyen MarketPay
Prijs	• Maandelijkse kost van €2 per actief account • €0.10 per uitbetaling • 0.5% van het bedrag • Kosten voor betaalmethode: 1.4%	• Verwerkingskosten: €0.10 • Kosten voor betaalmethode: gemiddeld tussen 0.90 en 1.10% • Geen maandelijkse kosten, integratiekosten, opzetkosten of kosten voor stopzetting • Wel maandelijkse factuur van minimum €100 per maand
Snelheid van uitbetalen	• Dagelijks • Wekelijks • Maandelijks • Handmatig	• Dagelijks • Wekelijks • Maandelijks • Handmatig
Veiligheid	• PSD • KYC • PCI DSS Level 1 certificaat	• PSD • KYC • European banking license • PCI DSS Level 1 certificaat
Mogelijkheden	• Betalingen splitsen • Registratie en verificatie • Terugbetalen • Geld overzetten tussen accounts	• Betaling splitsen • Registratie en verificatie • Terugbetalen • Geld overzetten tussen accounts
Verkrijgbaar in	32 landen	17 landen

Tabel 9: Vergelijkingsmatrix Stripe Connect en Adyen MarketPay

Stripe Connect en Adyen MarketPay verschillen voornamelijk in prijs en aantal ondersteunde landen. Eerst worden de gelijkenissen overlopen. Ten eerste de snelheid van betalen. Bij beide *payment service providers* heeft de eigenaar van de marktplaats de keuze wanneer hij accounts wilt uitbetalen. Ten tweede hebben ze een sterke beveiliging. Zowel Stripe Connect als Adyen MarketPay voldoen aan de KYC-vereisten en hebben het PCI DSS Level 1 certificaat. Beide PSP's beschikken ook over dezelfde mogelijkheden.

De PSP's verschillen op valk van prijs en ondersteunde landen. Adyen MarketPay is verkrijgbaar in 17 landen, terwijl Stripe Connect verkrijgbaar is in 32 landen. Beide rekenen €0.10 aan per betaling, maar Stripe rekent daarboven nog eens 0.5% van het bedrag en1.4% voor de betaalmethode. Adyen MarketPay rekent afhankelijk van de betaalmethode tussen de 0.90 en 1.10% van het bedrag bij.

Stripe Connect vraagt per account maandelijks €2. Adyen MarketPay heeft geen prijs per account, maar vraagt wel een maandelijkse betaling van minimum €100. Dit betekent indien de omzet van een gebruiker verhoogt, hij niet extra moet betalen aangezien dit een vaste prijs is. Verder heeft Adyen MarketPay geen kosten voor opzetting, stopzetting of integratie. Voor Stripe Connect is hier geen informatie over gevonden.

Conclusie

Om het betalen in restaurants te automatiseren zijn er verschillende stappen die onderzocht moeten worden. De klant moet zijn tafel kunnen identificeren, daarna de bijbehorende rekening op zijn smartphone ontvangen en in staat zijn om de rekening te betalen. Om de tafel te identificeren is gekeken naar de volgende technologieën: QR-codes en OCR. Er is gekeken welke oplossing de snelste is, het minste energie verbruikt en het gebruiksvriendelijkst is. Om de rekening op de smartphone te ontvangen, is er geïntegreerd met Untill. Via de *api* van Untill wordt de rekening van een tafel opgevraagd. Om de rekening te kunnen betalen moet gebruikgemaakt worden van een *payment service provider*. Er is een vergelijking gemaakt tussen Stripe Connect en Adyen MarketPay. Het gebruik van PSP's is nodig, omdat deze de betalingen afhandelen conform de wetgevingen en op deze manier komen de servers van de applicatie niet in contact met gevoelige gegevens.

Uit de vergelijking tussen QR-codes en OCR is gebleken dat OCR de snelste oplossing. Een belangrijke bemerking hierbij is dat het nemen van de foto niet inbegrepen is in de tijdsmeting. De tijdsmeting van de OCR was de tijd die nodig is om de foto om te zetten naar tekst. Bij de QR-codes zat het scannen van de code in de tijdsmeting. De camera werd wel op de code gericht voor de meting begon, zodat er hieraan geen tijd verloren werd. In energieverbruik verschilden de twee technologieën niet. Ze hadden beiden een laag energieverbruik. De gebruiksvriendelijkheid van de technologieën is ook getest. Hieruit blijkt dat de QR-codescanner het beter doet op de volgende criteria: gebruikers vonden de ze de rekening snel ontvingen, gebruikers vonden het proces duidelijk en gebruikers vonden dat ze niet veel stappen moesten ondernemen. Op al deze vlakken scoorde de QR-codescanner het beste. Het verschil in score tussen de QR-codescanner en de OCR is niet groot. Wanneer testers naar hun voorkeur gevraagd werd, werd de QR-codescanner met grote meerderheid gekozen.

Het gebruik van de Untill-*api* maakt het mogelijk om de rekening van een tafel op de te vragen en te laten weten wanneer de rekening van een tafel betaald is. Er is gekeken naar hoe snel een klant zijn rekening ontvangt en of de grootte van de rekening een effect heeft. Het proces van de rekening opvragen met een applicatie in plaats van manueel via de kassa kan mogelijk problemen geven. Wanneer een kassaticket manueel opgevraagd wordt, kan er één ticket opgevraagd worden op één moment. Het kan gebeuren dat meerdere gebruikers op hetzelfde moment hun rekening opvragen. De aanvragen kunnen ook veel sneller na elkaar gebeuren, omdat nu een gebruiker zijn rekening kan vragen en een andere gebruiker zijn rekening 0.5 seconden later kan opvragen. Om deze redenen is ook gecontroleerd wat er gebeurt als er naar één systeem meerdere *requests* tegelijk verstuurd worden en als er gedurende één minuut iedere seconde een *request* verstuurd wordt. Uit deze testen blijkt dat een gebruiker zijn rekening gemiddeld na 317,8 milliseconden ontvangt. De grootte van de rekening heeft een effect op de snelheid, maar zelfs wanneer het ticket 100 producten bevat, ontvangt de gebruiker zijn ticket ongeveer één seconde na aanvraag. Iedere seconde een *request* sturen gedurende één minuut heeft geen effect. De server blijft antwoorden en de responstijd blijft hetzelfde. Wanneer er 60 *requests* tegelijk gestuurd worden, handelt de server dit asynchroon af. Dit betekent dat het kassasysteem het aankan als meerdere gebruikers tegelijk hun rekening opvragen. Sommige gebruikers gaan in dit geval langer op hun rekening moeten wachten dan anderen.

Er zijn bij het gebruik van een kassasysteem enkele randvoorwaarden en beperkingen. Er is minder controle op de klanten. Daarom kan een klant gemakkelijker opstaan zonder te betalen. In de huidige situatie van restaurants kunnen klanten ook opstaan zonder te betalen, maar het is een punt om rekening mee te houden. Om de correcte rekening te ontvangen is een gebruiker afhankelijk van de ober. Indien een ober de producten niet tijdig ingeeft in het systeem moet de gebruiker wachten. Deze kans is erg klein, omdat om een POS-systeem goed te gebruiken, dient een ober dit meteen door

te geven. Ook moeten de gebruikers toegang hebben tot het internet om connectie te kunnen maken met het kassasysteem. Een laatste bemerking is dat klanten de tafels van andere klanten kunnen scannen. Dit gaat echter geen probleem zijn, omdat klanten ook kunnen zien wat andere klanten eten. Er is dus geen sprake van schending van privacy.

Stripe Connect en Adyen MarketPay zijn vergeleken op basis van de volgende criteria: prijs, snelheid van uitbetalen van geconnecteerde accounts, veiligheid en verschillende mogelijkheden. Op basis van snelheid van het uitbetalen van geconnecteerde accounts en verschillende mogelijkheden zijn Connect en MarketPay identiek. Beide bieden de mogelijkheid om dagelijks, wekelijks, maandelijks en handmatig uit te betalen. Ze bieden verder ook dezelfde mogelijkheden aan. Beide PSP's voldoen aan de vereiste voorwaarden, zoals KYC en hebben een PCI DSS Level 1 certificaat. MarketPay heeft nog een extra licentie, namelijk de European banking license. MarketPay en Connect hebben een andere benadering tot prijsbepaling. Beide rekenen €0.10 per uitbetaling. Stripe rekent daarop 0.5% van het bedrag, 1.4% voor de betaalmethode en maandelijks €2 per geconnecteerd account. MarketPay rekent bovenop de €0.10 afhankelijk van de gebruikte betaalmethode 0.9 tot 1.10% van het bedrag. Verder heeft MarketPay geen maandelijks variabel bedrag, maar wel een vast bedrag van €100 per maand.

Bibliografie

[1] „What is a QR code?," Unitag, [Online]. Available: https://www.unitag.io/qrcode/what-is-a-qrcode. [Geopend 28 Maart 2019].

[2] „What is a QR Code?," QRcode, [Online]. Available: https://www.qrcode.com/en/about/. [Geopend 28 Maart 2019].

[3] „How to read a QR Code?," Unitag, [Online]. Available: https://www.unitag.io/qrcode/how-to-scan-a-qrcode. [Geopend 28 Maart 2019].

[4] M. O'Neill, „What is a QR Code and how does it work?," smallbiztrends, 12 Februari 2019. [Online]. Available: https://smallbiztrends.com/2015/05/what-is-a-qr-code.html. [Geopend 28 Maart 2019].

[5] S. Pagin, „QR Codes: What are they and how do they work?," Fastprint, [Online]. Available: https://www.fastprint.co.uk/blog/quick-response-codes-what-are-they-and-how-do-they-work.html. [Geopend 28 Maart 2019].

[6] „Information capacity and versions of the QR Code," [Online]. Available: https://www.qrcode.com/en/about/version.html. [Geopend 9 Juni 2019].

[7] „Types of QR Code," QRcode, [Online]. Available: https://www.qrcode.com/en/codes/. [Geopend 28 Maart 2019].

[8] M. Maik, „What is OCR and how does it work for data extraction," Arbitrue, 16 Maart 2018. [Online]. Available: https://www.arbitrue.com/blog/what-is-ocr-and-how-does-it-work-for-data-extraction/. [Geopend 11 Maart 2019].

[9] „Tesseract OCR," [Online]. Available: https://github.com/tesseract-ocr/tesseract. [Geopend 15 Mei 2019].

[10] J. Hagen, 13 Mei 2019. [Online]. Available: https://github.com/tesseract-ocr/tesseract/wiki/ImproveQuality. [Geopend 15 Mei 2019].

[11] „How online credit card processing works," [Online]. Available: https://visual.ly/community/infographic/business/how-online-credit-card-processing-really-works-9-steps. [Geopend 4 Maart 2019].

[12] K. K. Acosta, „Online Payment Process," 16 Juni 2008. [Online]. Available: https://webuser.hs-furtwangen.de/~heindl/ebte-08-ss-Online-Payment-Process-Kathleen.pdf. [Geopend 28 April 2019].

[13] Vantiv, „What are payment service providers?," Vantiv, [Online]. Available: https://www.vantiv.com/payment-processing/payment-service-provider-explained-and-players-involved. [Geopend 4 april 2019].

[14] ComplyAdvantage, „What is means to "Know Your Customer"," ComplyAdvantage, [Online]. Available: https://complyadvantage.com/knowledgebase/kyc/. [Geopend 18 April 2019].

[15] Enigma Consulting, „Wat is PSD2?," Enigma Consulting, [Online]. Available: https://psd2.nl/payment-service-directive-2/over/. [Geopend 18 Maart 2019].

[16] GlobalSign, „What is PCI DSS Compliance?," GlobalSign, 25 Oktober 2016. [Online]. Available: https://www.globalsign.com/en/blog/what-is-pci-dss-compliance/. [Geopend 18 April 2019].

[17] Stripe, „The payments platform for platforms," Stripe, [Online]. Available: https://stripe.com/en-be/connect. [Geopend 18 April 2019].

[18] Stripe , „Connect Overview," Stripe, [Online]. Available: https://stripe.com/docs/connect. [Geopend 4 April 2019].

[19] Stripe, „Payments Overview," Stripe, [Online]. Available: https://stripe.com/docs/payments. [Geopend 12 Maart 2019].

[20] Z. Tamim, „A Swift Toturial for Stripe: Taking Credit Card Payments in iOS Apps," Appcoda, 17 Juli 2015. [Online]. Available: https://www.appcoda.com/ios-stripe-payment-integration/. [Geopend 4 April 2019].

[21] Adyen, „Marketplaces," Adyen, [Online]. Available: https://docs.adyen.com/developers/MarketPay. [Geopend 5 April 2019].

[22] Adyen, „iOS SDK," Adyen, [Online]. Available: https://beta-docs.adyen.com/checkout/ios-sdk/. [Geopend 5 April 2019].

[23] M. Brophy, „What is a POS System: Cost, Features & Providers," FitSmallBusiness, 9 Mei 2019. [Online]. Available: https://fitsmallbusiness.com/what-is-a-pos-system/. [Geopend 10 Mei 2019].

[24] Oracle, „What is a Restaurant POS System?," Oracle, [Online]. Available: https://www.oracle.com/industries/food-beverage/what-is-restaurant-pos.html. [Geopend 6 Mei 2019].

[25] V. Cruz, „What ind of Businesses Use POS Software?," Market Business News, 28 November 2018. [Online]. Available: https://marketbusinessnews.com/pos-software/190999/. [Geopend 6 ei 2019].

[26] Untill, „Branches," Untill, [Online]. Available: https://untill.com/branches/. [Geopend 21 Mei 2019].

[27] Untill, „Koppelen met unTill," Untill, [Online]. Available: https://untill.com/koppelingen/. [Geopend 21 Mei 2019].

[28] Lightspeed, „Word een Lightspeed partner," Lightspeed, [Online]. Available: https://www.lightspeedhq.be/partners/. [Geopend 22 Mei 2019].

[29] Apple, „Core Data," Apple, [Online]. Available: https://developer.apple.com/documentation/coredata. [Geopend 13 Mei 2019].

[30] P. Rea, „Geting started with Coe Data tutorial,” Ray Wederlick, 19 September 2018. [Online]. Available: https://www.raywenderlich.com/7569-getting-started-with-core-data-tutorial. [Geopend 13 Mei 2019].

[31] J. Simard, „A Look Into Realm's Core DB Engine,” Realm, [Online]. Available: https://academy.realm.io/posts/jp-simard-realm-core-database-engine/. [Geopend 13 Mei 2019].

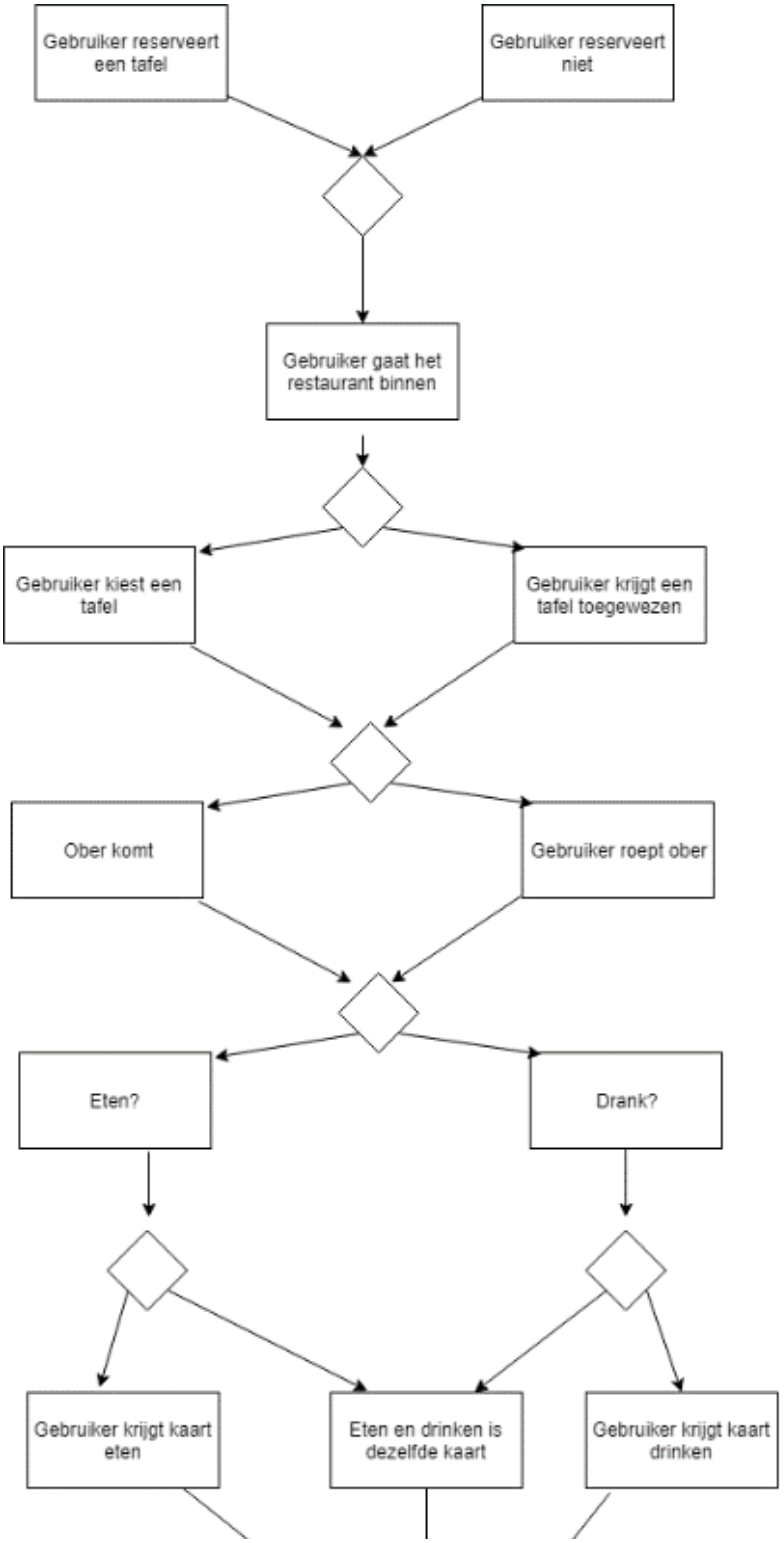
[32] Realm, Realm, [Online]. Available: https://realm.io/. [Geopend 13 Mei 2019].

Bijlagen

A. Schematische voorstelling van de stappen in een restaurant deel 1

B. Schematische voorstelling van de stappen in een restaurant deel 2

A. Schematische voorstelling van de stappen in een restaurant deel 1



B. Schematische voorstelling van de stappen in een restaurant deel 2