

Odisee Hogeschool Gent  
Gebroeders De Smetstraat 1, 9000 Gent

EASYSHOP

Onderzoek en ontwikkeling van een winkelapplicatie

**Auteurs**

Niels De Bruyne, Piet Jacobs, Seppe Snoeck, Daan Zwaenepoel

**Mentoren**

Professionele Opleiding EO-ICT Sabine Martens Academiejaar 2016-2017 Katja Verbeeck



Odisee Hogeschool Gent  
Gebroeders De Smetstraat 1, 9000 Gent

EASYSHOP

Onderzoek en ontwikkeling van een winkelapplicatie

**Auteurs**

Niels De Bruyne, Piet Jacobs, Seppe Snoeck, Daan Zwaenepoel

**Mentoren**

Professionele Opleiding EO-ICT Sabine Martens Academiejaar 2016-2017 Katja Verbeeck

INHOUD

[Figurenlijst 3](#_Toc483243364)

[Tabellenlijst 4](#_Toc483243365)

[Codefragmentenlijst 5](#_Toc483243366)

[Inleiding 6](#_Toc483243367)

[1 Mogelijke oplossingen 7](#_Toc483243368)

[1.1 Marktonderzoek 7](#_Toc483243369)

[1.1.1 Bespreking enquête 7](#_Toc483243370)

[1.1.2 Besluitvorming design webinterface 10](#_Toc483243371)

[1.2 Programmeertalen 10](#_Toc483243372)

[1.2.1 Frontend 10](#_Toc483243373)

[1.2.2 API 11](#_Toc483243374)

[1.3 Betaalmethoden 11](#_Toc483243375)

[1.4 Cloudoplossingen 12](#_Toc483243376)

[2 Gekozen oplossing 13](#_Toc483243377)

[2.1 Programmeertalen 13](#_Toc483243378)

[2.2 Betaalmethoden 13](#_Toc483243379)

[2.3 Cloudoplossing 13](#_Toc483243380)

[3 Technische uitwerking 14](#_Toc483243381)

[3.1 Bespreking frontend 14](#_Toc483243382)

[3.1.1 Routering webinterface 14](#_Toc483243383)

[3.1.2 Opbouw frontend 14](#_Toc483243384)

[3.1.3 API-calls 17](#_Toc483243385)

[3.1.4 Betaalmethode 17](#_Toc483243386)

[3.2 Bespreking backend 17](#_Toc483243387)

[3.2.1 Opbouw API 17](#_Toc483243388)

[3.2.2 Linux-server 18](#_Toc483243389)

[4 Risicoanalyse 19](#_Toc483243390)

[4.1 Risicovolle datatransmissie 19](#_Toc483243391)

[4.2 Onveilige paswoordsleutels 19](#_Toc483243392)

[4.3 Betrouwbaarheid betaalmethoden 19](#_Toc483243393)

[5 Kostenraming en levensduur 20](#_Toc483243394)

[5.1 Totale kosten 20](#_Toc483243395)

[5.2 Levensduur webapplicatie 20](#_Toc483243396)

[5.3 Levensduur hardware 20](#_Toc483243397)

[Conclusie 21](#_Toc483243398)

[Nawoord met kritische reflectie 22](#_Toc483243399)

[Literatuurlijst 23](#_Toc483243400)

[Bijlagenoverzicht 25](#_Toc483243401)

[Bijlage 1: Enquête koopgedrag 25](#_Toc483243402)

[Bijlage 2: Werkplanning 29](#_Toc483243403)

[Bijlage 3: Verslag vergadering 1 30](#_Toc483243404)

[Bijlage 4: Verslag vergadering 2 31](#_Toc483243405)

# Figurenlijst

[Figuur 1: Leeftijd van de bevraagden 7](#_Toc483235616)

[Figuur 2: Gebruik toestellen om te surfen op het internet 8](#_Toc483235617)

[Figuur 3: Afstand naar de winkel (in kilometer) 8](#_Toc483235618)

[Figuur 4: Tijdspanne van 1 boodschappensessie (in minuten) 9](#_Toc483235619)

[Figuur 5: Gebruik onlinebetaalmethoden 9](#_Toc483235620)

[Figuur 6: Routering website Easyshop 14](#_Toc483235621)

# Tabellenlijst

[Tabel 1: Enkele eigenschappen van enkele mogelijke frameworks [3] [4] 10](#_Toc483235629)

[Tabel 2: Mogelijke programmeertalen voor het opbouwen van de API [7] [8] 11](#_Toc483235630)

[Tabel 3: Mogelijke onlinebetaalmethoden [9] 12](#_Toc483235631)

[Tabel 4: Totale kostprijs voor het opzetten van Easyshop 20](#_Toc483235632)

# Codefragmentenlijst

[Codefragment 1: Routering van de website met React-Router-DOM 15](#_Toc483235655)

[Codefragment 2: Opbouw React-component 16](#_Toc483235656)

[Codefragment 3: Gebruik React-component 16](#_Toc483235657)

[Codefragment 4: Glamor-syntax 16](#_Toc483235658)

[Codefragment 5: Koppelen van Glamor aan React-component 17](#_Toc483235659)

# Inleiding

Volgend project wordt uitgevoerd in opdracht van Projecten 1, onderdeel van fase twee in de opleiding elektronica-ICT aan de hogeschool Odisee Gent. Het omvat een webapplicatie genaamd Easyshop die als doel heeft om boodschappen te laten bezorgen door een andere gebruiker. De applicatie heeft twee doelgroepen. De eerste groep zijn de mensen die behoefte hebben aan thuislevering van hun boodschappen. De tweede groep zijn de mensen die de boodschappen aan huis leveren tegen een vergoeding.

Het is mogelijk om een boodschappenlijst te plaatsen, waarna een bezorger de aanvraag kan accepteren en uitvoeren. Wanneer een bezorger een ticket accepteert dan ontstaat er een ‘contract’ met de persoon die het ticket heeft geplaatst. Er zal gewerkt worden met een stappenplan waarbij de boodschappen, winkel en het tijdstip van de levering worden vastgelegd. Nadien levert de bezorger de boodschappen aan huis. Easyshop zal gebruik maken van een onlinebetaalservice. Deze service zal moeten voldoen aan de normen van de Payment Card Industry Data Security Standard.

Easyshop wordt van scratch gemaakt en moet aan een aantal voorwaarden voldoen. Zo is een eerste voorwaarde de financiële veiligheid. Er moet op een veilige en liefst zo goedkoop mogelijke manier geld overgemaakt kunnen worden tussen drie verschillende partijen namelijk de gebruikers, de bezorgers en Easyshop. Een tweede voorwaarde is het vertrouwen in het betaalsysteem. Om dit te bevorderen wordt er gebruik gemaakt van online betrouwbare betaalsystemen van externe services. Een volgende voorwaarde is het opbouwen van een vertrouwensrelatie met de gebruikers. De webinterface moet professioneel ogen om een eerste goede indruk te geven aan potentiële klanten en daarnaast moet er een controle zijn op het correct handelen van alle gebruikers.

De applicatie wordt gerealiseerd door enerzijds duidelijke gesprekken te voeren met de begeleidende technische docent en door anderzijds een literatuurstudie te doen over enkele mogelijke oplossingen. Daarnaast moet er onderzoek verricht worden aan de hand van enquêtes naar de noden van de doelgroepen om deze zo goed mogelijk in te vullen. Dergelijk ontwerp van de webapplicatie ligt in de lijn van de kennis en vaardigheden reeds verworven in user-experience-design. Alvorens het programmeren moeten eerst enkele mogelijke programmeertalen afgewogen worden. Na het maken van de meest gunstige keuze wordt er tijd gespendeerd om de code zo robuust mogelijk te maken. De code zal dan ook vaak getest worden om aan deze voorwaarde te voldoen. Ook zijn er voldoende onlinebronnen voorhanden om tot mogelijke oplossingen te komen.

Er wordt een periode van één maand vanaf 1 februari 2017 voorzien om het onderzoek klaar te hebben. Voor de realisatie van de applicatie worden nog eens drie maanden gerekend.

Het rapport start met een opsomming van de mogelijke oplossingen. Na afweging van de voor- en nadelen wordt er een oplossing gekozen. Deze zal dan stap voor stap beschreven worden in de technische uitwerking. Nadien zal een risicoanalyse volgen die de veiligheid onder de loep neemt. Daaropvolgend wordt de kostenraming besproken. Tot slot wordt het volledige rapport samengevat in de conclusie en volgt er nog een nawoord met kritische reflectie.

# Mogelijke oplossingen

## Marktonderzoek

In dit onderdeel wordt het marktonderzoek besproken. Er wordt gebruik gemaakt van een enquête (bijlage 1) om informatie te verzamelen over de noden van mogelijke eindgebruikers. Deze informatie heeft een impact op de opbouw en mogelijkheden van de webinterface.

### Bespreking enquête

De enquête bestaat uit 16 vragen die het koopgedrag op het internet nagaat van 80 willekeurige personen. Hieronder worden een aantal verkregen antwoorden, die een weerslag kunnen hebben op het eindresultaat, grafisch voorgesteld en besproken.

De enquête is ingevuld door een grote variatie leeftijdsgroepen. Niemand is echter jonger dan 16 jaar (figuur 1). Zo’n 60% van de bevraagden zijn 40 of jonger, de overige 40% is dan vanzelfsprekend ouder dan 40, waarvan maar 3% ouder is dan 70 jaar.

Hieruit kan besloten worden dat boodschappen, al dan niet online, door alle leeftijdsgroepen gedaan worden.

Figuur 1: Leeftijd van de bevraagden

Uit de enquête blijkt ook dat 43% van de mensen gebruik maakt van een computer om te surfen op het internet (figuur 2). Een kleinere groep van 37% maakt vooral gebruik van een smartphone en 20% gebruikt een tablet om te surfen.

Om een zo breed mogelijk publiek aan te spreken, wordt er gekozen voor het ontwikkelen van een webinterface. Op deze manier kunnen gebruikers via alle mogelijke toestellen die over internet beschikken terecht op de webshop.

Figuur 2: Gebruik toestellen om te surfen op het internet

In de enquête is ook duidelijk geworden dat de meeste bevraagden een bakker, slager of een klein buurtwinkeltje beschikbaar hebben binnen een straal van 0 tot 3 kilometer (figuur 3). Om boodschappen in grotere hoeveelheden te kopen, moeten velen meer dan 3 kilometer afleggen.

Figuur 3: Afstand naar de winkel (in kilometer)

Er is ook gevraagd naar hoe lang iemand er over doet om boodschappen te doen (figuur 4). Zo’n 39% doet er minder dan 30 minuten over terwijl 30% er 30 tot 60 minuten over doet. Dan zijn er nog twee minderheidsgroepen, waarvan 22% er 60 tot 90 minuten over doet en 8% 90 tot 120 minuten. Er is ook één gebruiker die één maal per maand boodschappen doet en er meer dan 120 minuten over doet.

Easyshop kan de tijd dat mensen spenderen aan het doen van boodschappen verminderen, zo hoeven gebruikers zich niet meer te verplaatsen naar winkels die vaak op afstanden van meer dan 3 kilometer liggen.

Figuur 4: Tijdspanne van 1 boodschappensessie (in minuten)

Het gebruik van onlinebetaalmethoden is bij de bevraagden zeer eenduidig (figuur 5). Maar liefst 70% maakt gebruik van Bancontact. Alternatieve betaalmethoden worden aanzienlijk minder gebruikt.

Figuur 5: Gebruik onlinebetaalmethoden

### Besluitvorming design webinterface

Ten eerste moet de interface gebruiksvriendelijk zijn en zeker niet complex, het is tenslotte de bedoeling dat mensen van alle leeftijden gebruik kunnen maken van de webinterface om boodschappen te doen of hun diensten als leverancier aan te bieden. Een volgende belangrijke pijler is dat de webinterface op alle toestellen met internettoegang moet kunnen geopend worden zodat er geen potentiële gebruikers uitgesloten worden. Ten derde is boodschappen doen tijdrovend, zeker wanneer de winkels op grotere afstanden liggen, Easyshop moet snel en eenvoudig zijn om tijd te besparen. Tot slot moeten er voldoende betaalmogelijkheden beschikbaar zijn voor de potentiële gebruikers, een minimumvereiste is het implementeren van Bancontact.

## Programmeertalen

Voor deze toepassing wordt er gekozen voor een webinterface zoals hierboven besproken. In dit hoofdstuk worden er een aantal mogelijke programmeertalen vooropgesteld om deze webinterface op te implementeren.

### Frontend

Voor het aanmaken van een webpagina zijn er twee programmeertalen van essentieel belang, namelijk HyperText Markup Language (HTML) en Cascading Style Sheets (CSS). Beide vormen de basis van een eenvoudige webpagina, waarbij HTML zorgt voor de nodige structuur, terwijl CSS de opmaak verzorgt. [1] [2]

Deze basis volstaat echter niet om aan alle nodige functionaliteiten van Easyshop te voldoen. Hiervoor wordt er gebruik gemaakt van JavaScript, dit is een programmeertaal waarmee men het gedrag van een website kan programmeren. De mogelijkheden van JavaScript zijn echter eindeloos en worden snel complex. [1] Om het gebruik van deze krachtige programmeertaal te vereenvoudigen zijn er verschillende frameworks op deze taal gebaseerd, elk met een unieke syntax en functies die bepaalde opdrachten gemakkelijker maken. De webinterface zal dan ook gebruik maken van één of meerdere frameworks die de nodige tools aanbieden om te voldoen aan die doelstellingen. In volgende tabel zijn hiervoor een aantal mogelijke frameworks opgesomd die tegen elkaar worden afgewogen.

Tabel 1: Enkele eigenschappen van enkele mogelijke frameworks [3] [4]

|  |  |  |  |
| --- | --- | --- | --- |
|  | React | Angular2 | Polymer |
| Ontwikkelaar | Facebook | Google | Google |
| Hoofddoel | Dynamische content | Beheren van data en de paginaroutering | Gebruiksvriendelijk en performant |
| Community | Heel groot, veel documentatie en codevoorbeelden | Heel groot, veel documentatie en codevoorbeelden | Groot, voldoende documentatie en codevoorbeelden |

In tabel 1 zijn er een aantal belangrijke eigenschappen opgesomd van enkele populaire frameworks die bruikbaar zijn voor dit project. Naast deze frameworks zijn er nog veel die de nodige tools aanbieden. [5] Hier is de vergelijking beperkt tot de meest populaire frameworks, omdat deze het best gedocumenteerd zijn en voldoende codevoorbeelden bevatten. Ook worden deze in bijna alle moderne browsers ondersteund, wat van belang is om zo’n groot mogelijke doelgroep aan te spreken. Easyshop is een webshop, de pagina-inhoud wordt dan ook het best op een dynamische manier weergegeven. Omwille van deze reden wordt er voor React gekozen binnen dit project.

### API

Een Application Programming Interface (API) zorgt voor de communicatie tussen de verschillende lagen van de website en de achterliggende database. [6] Voor het opbouwen van de API zijn er opnieuw een aantal mogelijkheden, waarvan enkele belangrijke eigenschappen in onderstaande tabel worden weergegeven.

Tabel 2: Mogelijke programmeertalen voor het opbouwen van de API [7] [8]

|  |  |  |
| --- | --- | --- |
|  | Php | Python |
| Laag | Backend | Backend |
| LEEFTIJD | Oud (1995), heel matuur, allomtegenwoordig | Oud (1991) en nog steeds het meest populair |
| Frameworks | Laravel, CodeIgniter… | Django, Flask… |
| Complexiteit | Heel eenvoudig en snel | Eenvoudig en minimalistisch |

In tabel 2 zijn enkele belangrijke eigenschappen van de twee meest gebruikte programmeertalen weergegeven waarmee de API kan opgebouwd worden. Ze zijn beide voorzien van een uitgebreide documentatie met codevoorbeelden. Voor dit project wordt er gekozen voor NodeJS in samenwerking met Python. NodeJS verwerkt data langs de frontend aan de hand van een dynamische pagina. Het is een relatief nieuwe taal dat frequent gebruikt wordt sinds 2009. Hierdoor is het een taal die veel moderne features bevat en een moderne en relatief eenvoudige syntax heeft. De communicatie met de database wordt dan via een Python-oplossing verzorgd. Python is het doorgeefluik die voor de uitwisseling van de data zorgt tussen NodeJS en de database.

## Betaalmethoden

Om betalingen te doen moeten er enkele betaalmethoden voorzien worden. Hieronder worden er enkele mogelijke providers vergeleken met de aangeboden betaalmethoden.

Tabel 3: Mogelijke onlinebetaalmethoden [9]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Paypal | Multisafepay | Ogone | MOLLIE |
| Betaalmethoden | Paypal, Visa, Bancontact… | Visa, MasterCard, iDeal, Bancontact… | Visa, MasterCard, Bancontact… | Bancontact, Visa, Paypal… |
| Prijs | 3,4% + €0,35 per transactie | Opstartkost van €75, maandelijkse kost van €45 en transactiekosten tussen de €0,15 en 2.9% + €0,15 | Opstartkosten van €300, maandelijkse kost van €65 en transactiekost van €0,37 (Fraudedetectie kost extra) | Tussen €0,25 en €15% van de transactie |
| Eenvoud | Gebruikers moeten een account aanmaken | Gebruiksvriendelijk | Gebruiksvriendelijk | Gebruiksvriendelijk |

In tabel 3 zijn er een aantal providers opgesomd die een rijk aanbod aan betaalmethoden verschaffen. De ene biedt al wat meer features aan dan de andere en ook prijzen variëren. Omdat dit project gelimiteerd is in het maken van kosten, wordt er dus gekozen voor de goedkoopste oplossing, namelijk Mollie. Dit zorgt voor een basisimplementatie van enkele betaalmethodes, in een latere fase van het project kan nog steeds gekozen worden voor een provider met meer features indien hier nood aan is. De betaalmethoden die Mollie aanbiedt voldoen ook aan de Payment Card Industry Data Security Standard. [10]

## Cloudoplossingen

Om de website bereikbaar te maken voor iedereen moet deze gehost worden. Hieronder worden er enkele mogelijk cloudoplossingen besproken om de website te hosten.

Een eerste mogelijke cloudoplossing is cPanel, dit wordt door ikdoeict.be aangeboden voor studentenprojecten, zodat ze zelf niet moeten investeren in de nodige hardware. Het heeft een gebruiksvriendelijke webinterface met uitgebreide features en goede ondersteuning. [11]

Een andere mogelijke oplossing is GitHub Pages. Dit is volledig gratis, maar daarom ook beperkt in functionaliteit. Voor dit project is het dan ook niet bruikbaar aangezien GitHub Pages enkel in staat is statische pagina’s te hosten, Easyshop daarentegen maakt gebruik van een backend. [12]

Een laatste mogelijke oplossing die in dit hoofdstuk besproken wordt, is DigitalOcean. Afhankelijk van de benodigde features en hardware kan er gekozen worden tussen verschillende pakketten, elk met een eigen prijskaartje. Het maakt gebruik van een gebruiksvriendelijke webinterface waar alle instellingen terug te vinden zijn. Een groot voordeel van DigitalOcean is dat er heel gemakkelijk internationale servers toegevoegd kunnen worden indien de webinterface internationaal beschikbaar moet worden gemaakt. [13]

# Gekozen oplossing

De gekozen oplossing bestaat uit een combinatie van concepten die betrekking hebben tot de gekozen programmeertalen, de betaalmethoden en de cloudoplossing voor het bouwen van de webinterface. Bij het kiezen van de uiteindelijke oplossing wordt er rekening gehouden met de noden van de eindgebruikers die in de enquête aangehaald zijn, maar ook met de meest gunstige en voordelige oplossing voor de projectleiders.

## Programmeertalen

De programmeertalen zijn essentieel voor het implementeren van de webinterface. Er is gekozen voor HTML om de nodige webstructuur te vormen in combinatie met CSS voor het opmaken van de pagina.

Om het geheel dynamisch en programmeerbaar te maken, wordt er gebruik gemaakt van React, een krachtig JavaScript-framework dat het mogelijk maakt om pagina’s dynamisch bij te werken. Dit verzorgt samen met NodeJS de frontend van de pagina. NodeJS verwerkt dan de data langs de frontend en geeft deze door aan de backend die gebruik maakt van Python.

Python wordt gebruikt voor de implementatie van de API, deze zorgt dan weer voor de communicatie tussen de frontend en de achterliggende database. Deze laatste maakt gebruik van MySQL in combinatie met de InnoDB engine omdat deze perfomant en niet gevoelig aan storingen is. [14]

## Betaalmethoden

De betaalmethoden hangen vast aan een bepaalde provider. Dit project maakt gebruik van Mollie, één van de goedkoopste providers voor de betaalmogelijkheden die Easyshop zal aanbieden. Bovendien zijn er geen opstartkosten aan deze diensten verbonden. [15]

Per Bancontact-transactie rekent Mollie €0,25 + 1,5%, voor PayPal geldt €0,10 + PayPal-aandeel, Mastercard en Visa kosten €0,25 + 1.8% voor Europese betalingen. Mollie voorziet ook een API-implementatie waar de webinterface gebruik van maakt voor betalingen.

## Cloudoplossing

Als cloudoplossing wordt er gekozen voor de oplossing van DigitalOcean, het is een goedkope oplossing die veel te bieden heeft en vooral flexibel is met uitbreidingsmogelijkheden. [13] Het is een zeer respectabel bedrijf dat garant staat voor hoge beschikbaarheid en betrouwbaarheid. Bovendien kunnen studenten gratis gebruik maken van hun services.

Voor dit project wordt er gestart met een eenvoudige Linuxbox die volledig naar eigen hand kan worden gezet door middel van uitgebreide configuratiemogelijkheden en verschillende tools die beschikbaar zijn via de eenvoudige webinterface van DigitalOcean.

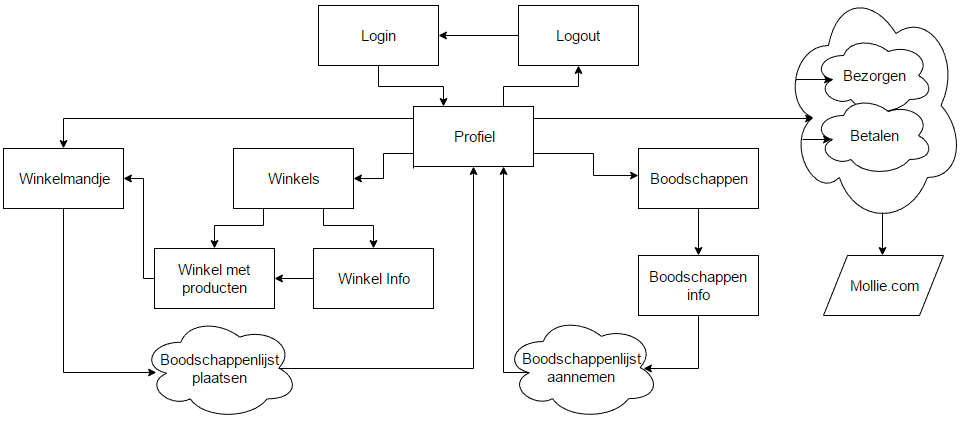
# Technische uitwerking

In dit hoofdstuk wordt de technische uitwerking toegelicht. Er wordt besproken hoe de frontend en backend samen werken als één geheel.

## Bespreking frontend

Om een goed idee te krijgen over de werking van de website wordt eerst de routering van de verschillende webcomponenten besproken. Nadien volgt de technische uitwerking van de frontend.

### Routering webinterface



Figuur 6: Routering website Easyshop

In bovenstaande afbeelding is te zien hoe de routering van de website (figuur 6) precies werkt. Het eerste dat een gebruiker te zien krijgt wanneer die naar de website van Easyshop surft, is een loginpagina. Eenmaal ingelogd krijgt de gebruiker een profielpagina te zien. Hierop kan de persoonlijke informatie, de geplaatste bestellingen en de te leveren bestellingen van de gebruiker bekeken worden. Daarnaast kan de gebruiker vanaf de profielpagina opnieuw uitloggen, het huidige winkelmandje bekijken, winkels bezoeken waarbij men orders kan aanmaken en boodschappen bekijken die de gebruiker dan voor een ander persoon kan leveren. Om boodschappen te leveren voor iemand moet een gebruiker zich inschrijven op de aanvraag van die persoon. Eenmaal de goederen bezorgd zijn, kan er betaald worden met het betaalsysteem dat terug te vinden is op de profielpagina naast het order dat eerder werd aangevraagd.

### Opbouw frontend

De frontend begint bij een eenvoudige HTML-pagina die wat CSS-bestanden en een JavaScript-bestand inleest. Zoals eerder aangehaald wordt er in dit project gebruik gemaakt van React, een JavaScript-framework dat werkt op basis van React-componenten en een dynamische output levert. Dit wil zeggen dat eenmaal de website ingeladen is de webpagina niet meer volledig opnieuw gerenderd wordt. Enkel de componenten die aan wijzigingen toe zijn door interactie van de gebruiker zullen aangepast worden.

Door het gebruik van dit principe is de website snel, maar hierdoor is er ook nood aan routering binnen de webpagina. Deze routering gebeurt aan de hand van de React-Router-DOM die communiceert met de verschillende hoofdcomponenten. Zo zijn er een aantal hoofdcomponenten elk verantwoordelijk voor een deel van de website zoals de profielpagina, de winkelpagina, de boodschappenpagina, de pagina met het winkelmandje enzovoort codefragment 1. Wanneer er bijvoorbeeld een aanvraag gestuurd wordt om het winkelmandje te renderen, dan roept de React-router het React-component aan dat verantwoordelijk is voor het winkelmandje.



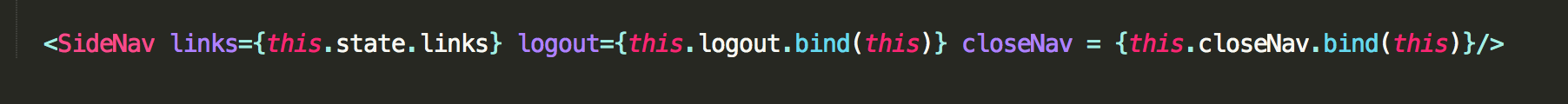
Codefragment 1: Routering van de website met React-Router-DOM

In codefragment 2 is de basisstructuur van een React-component weergegeven. [16] Helemaal bovenaan de code wordt er een import gedaan van enkele externe bibliotheken, waaronder ook de eerder besproken React-Router-DOM. Nadien wordt er een eigen component geïmporteerd, in dit geval de header van de website. Daarna volgt er een constructor van een bepaalde klasse, in onderstaand voorbeeld gaat het over de klasse die verantwoordelijk is voor de profielpagina. Vervolgens kan men verschillende methodes aanroepen die uiteenlopende functies bieden. In dit voorbeeld worden de methodes componentWillMount() en componentDidMount() aangeroepen. De eerste voert stukken code uit alvorens de methode render wordt aangeroepen, terwijl de andere methode pas na het renderen actief is. Tot slot zal er in de methode render HTML-code gereturnd worden om de nodige componenten op de webpagina te renderen.



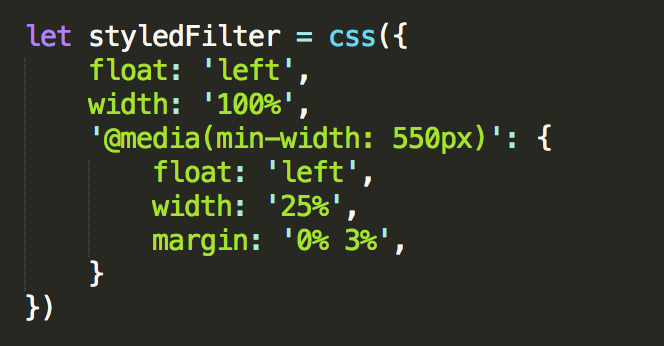
Codefragment 2: Opbouw React-component

Vooraleer er kan gewerkt worden met een React-component moet het eerst geïmporteerd worden. Nadien kunnen er properties aan het component gebonden worden met de nodige data die dan gebruikt kan worden in verschillende componenten. In codefragment 3 is te zien dat het component SideNav bestaat uit de properties links, logout en closeNav.



Codefragment 3: Gebruik React-component

Om de React-componenten te stijlen wordt er gebruik gemaakt van Glamor. [17] In codefragment 4 is er een stuk Glamor-code te zien. De syntax ligt heel dicht bij die van traditionele CSS. Na het aanmaken van de stijl, moet deze nog gekoppeld worden aan een React-component aan de hand van een HTML-tag zoals in codefragment 5.



Codefragment 4: Glamor-syntax



Codefragment 5: Koppelen van Glamor aan React-component

### API-calls

Om API-calls te sturen wordt er gebruik gemaakt van de bibliotheek Axios. [18] De meeste API-calls worden pas uitgevoerd na een eerste render. Op deze manier voelt de website steeds snel aan. Daarnaast biedt de bibliotheek ook de mogelijkheid tot Cross-Site-Access-Control aanvragen. Dit is noodzakelijk om de applicatie te laten communiceren met de verschillende hosts van de frontend en backend. Tot slot worden de API-calls beveiligd met een JSON Web Token dat in het volgende onderdeel verder wordt toegelicht.

### Betaalmethode

Om financiële transacties te verrichten wordt er gebruik gemaakt van de provider Mollie. Aan de hand van hun services is het mogelijk om tal van betaalmogelijkheden aan onze eindgebruikers aan te bieden. De API verzorgt de communicatie tussen Easyshop en Mollie voor de nodige betalingen. Mollie voorziet ook een dashboard waar alle informatie over de verschillende transacties terug te vinden is.

Om gebruik te maken van de services van Mollie moet de server voorzien zijn van het NPM-softwarepakket. Nadien kan je de API van Mollie aanspreken met PHP om API-calls te maken. Het PHP-bestand bestaat uit drie onderdelen. Een eerste onderdeel is het gebruik van de correcte API-sleutel. Mollie biedt een sleutel aan om virtuele transacties te doen, zo kan het gebruik van Mollie getest worden zonder effectieve kosten. Daarnaast is er ook een sleutel voorzien wanneer de website in productie gaat zodat transacties met reëel geld kunnen plaatsvinden. Een tweede onderdeel is het declareren van de nodige variabelen zoals het gebruikte protocol, hostname, path enzovoort. In het laatste onderdeel worden de API-calls effectief uitgevoerd, hier worden de nodige variabelen meegegeven om de betaling mogelijk te maken.

## Bespreking backend

De backend voorziet de nodige API-calls die aangesproken kunnen worden door de frontend om data uit de database te halen of in te database te stockeren. Daarnaast is het ook van belang dat de website gehost wordt op een server zodat deze steeds bereikbaar is op het internet.

### Opbouw API

In het tweede hoofdstuk werd er gekozen om de API te implementeren met Python. Na onderzoek is gebleken dat dit echter niet haalbaar is. Er was geen tool te vinden met de mogelijkheid om de API te genereren, dit is noodzakelijk wegens de vooropgestelde termijn. Een volledige API zelf schrijven voor deze applicatie is zeer tijdrovend. Hierdoor is er gekozen voor een eenvoudigere aanpak aan de hand van een bestaand PHP-project genaamd PHP-CRUD. [19] Deze tool maakt het mogelijk om alle nodige API-calls te genereren op basis van de tabellen in de database. Dit bespaart veel tijd, die gebruikt wordt voor de verdere ontwikkeling van de applicatie. De documentatie van de API is gegenereerd door de tool Swagger, die werkt op basis van een YAML-file die door de API is gegenereerd. [20] Bovendien is het geheel dynamisch, indien er wijzigingen doorgevoerd worden in de database, dan zal de API opnieuw gegenereerd worden en de documentatie bijwerken.

Om API-calls uit te voeren vanaf de frontend is er een extra laag aan beveiliging aanwezig dat gebruik maakt van een twee-staps-verificatie. Eerst moet er een aanvraag gestuurd worden naar de API met de correcte credentials, na verificatie wordt er dan een JSON Web Token gegenereerd en teruggestuurd naar de frontend. Deze token moet dan per sessie opnieuw gepost worden naar de API, die een unieke CSRF-token terugstuurt om API-calls mogelijk te maken. [21]

### Linux-server

De volledige applicatie wordt gehost vanaf een Linux-server die draait op de meest recente versie van Ubuntu op het moment van schrijven. Deze server wordt gehuurd bij DigitalOcean en kan remote geconfigureerd worden door gebruik te maken van een SSH-verbinding.

De server is voorzien van de nodige softwarepakketten om de webapplicatie als geheel te laten draaien. [22] Zo maakt de backend gebruik van een MySQL database om alle nodige informatie te stockeren en PHP 7 om de API-calls te kunnen verwerken. Daarnaast is er voor de frontend Apache 2 geïnstalleerd op de server om de webpagina online beschikbaar te maken. De webinterface is beschikbaar op het domein easy-shop.xyz en de API op het subdomein api.easy-shop.xyz.

# Risicoanalyse

Het is van groot belang om de webapplicatie goed te beveiligen. Dit omwille van het feit dat er financiële transacties gedaan worden en omdat er een database aan de webinterface hangt die persoonlijke informatie bevat. Hieronder worden er een aantal mogelijke problemen beschreven en hoe deze voorkomen kunnen worden.

## Risicovolle datatransmissie

De datatransmissie is een volgend onderwerp dat zeker moet voldoen aan een aantal veiligheidsnormen. Het is niet wenselijk dat een aanvaller in staat is data aan te passen tijdens een financiële transactie. Het is noodzakelijk om alle communicatie tussen de frontend en de API van Easyshop te versleutelen. Dit gebeurt aan de hand van de open standaard van JSON Web Token met behulp van het SHA256-algoritme. [21] De betalingen gebeuren via Mollie en maken gebruik van een HTTPS-verbinding en een SHA1-HMAC-versleuteling. [23] Dit is zeker niet het meeste veilige algoritme en is voor verbetering vatbaar.

## Onveilige paswoordsleutels

Het spreekt voor zich dat paswoorden sterk beveiligd moeten worden. Hoe belangrijker de informatie, hoe sterker het algoritme om de data te versleutelen. In dit project wordt er gewerkt met prijzen en betalingen. Het is dus belangrijk om een sterk algoritme te kiezen dat voldoende bescherming biedt. [24]

Op het moment van schrijven, is alle data in de databank handmatig ingevoerd om een aantal zaken te testen en is er nog geen data geëncrypteerd. Dit is zeker noodzakelijk wanneer de webapplicatie in productie gaat om de gegevens van de gebruikers op een veilige manier te stockeren.

Het systeem zal dan moeten voorzien worden van een encryptie- en decryptiesysteem om logingegevens te valideren. Er zullen een aantal mogelijkheden moeten afgewogen worden om een evenwicht te vinden tussen veiligheid en de snelheid van het algoritme.

## Betrouwbaarheid betaalmethoden

Zoals eerder besproken maakt Mollie gebruik van een minder sterk algoritme om de data te versleutelen. Hierdoor is het zeker aangeraden om te kijken naar alternatieve providers die een betere veiligheid kan garanderen door gebruik te maken van een beter algoritme. Voor dit project was het van belang om de kosten te drukken en dit was mogelijk door het gebruik van een testaccount met virtuele transacties. Andere providers hebben vaak opstartkosten, bieden een minder groot aanbod aan betaalmogelijkheden en hebben vaak niet zo’n uitgebreide documentatie als Mollie. [25]

# Kostenraming en levensduur

Er is geen budget voor dit schoolproject voorzien. Indien er toch onvermijdelijke kosten opduiken wordt er geopteerd voor de goedkoopste oplossing die voldoet aan de noden van de eindgebruiker. De kosten worden dan door persoonlijke bronnen gefinancierd.

## Totale kosten

Tabel 4: Totale kostprijs voor het opzetten van Easyshop

|  |  |  |
| --- | --- | --- |
| **Kostprijsberekening opzetten Easyshop** | | |
| **Item** | **Bedrijf** | **Prijs** |
| Domeinnaam: easy-shop.xyz | Namecheap | €0,90 |
| Huur server: Ubuntu 16.04 | DigitalOcean | €5/maand |
| **Totaal:** | | **€5,90** |

De totale prijs voor dit project bedraagt €5,90 zoals vermeld in tabel 4. Dit zijn enkel de kosten die in rekening zijn gebracht, uiteraard zijn er veel werkuren nodig om dit project te verwezelijken. Het project is dan ook bedoeld als Proof of Concept. Alvorens de applicatie op de markt verschijnt, is er nood aan een verbeterde beveiliging en meer features om te voldoen aan de noden van de eindgebruiker. Om deze twee elementen te implementeren moet er een groter budget en meer tijd voorzien worden.

## Levensduur webapplicatie

Bij het ontwikkelen van een applicatie is het zo goed als onmogelijk om meteen een perfect product te leveren. Het is noodzakelijk om de toepassing regelmatig te voorzien van updates en nieuwe features. Zoals eerder vermeld is de huidige staat van de applicatie zeker niet het eindproduct dat op de markt gelanceerd wordt. De applicatie moet nog evolueren om aan alle vooropgestelde doeleinden te voldoen.

Eenmaal er een eindproduct op de markt gebracht wordt, kan de levensduur van de webapplicatie jaren bedragen indien er genoeg interesse is van gebruikers en de ontwikkeling van Easyshop in een positieve zin evolueert.

## Levensduur hardware

Door gebruik te maken van de services van DigitalOcean voor het leveren van de nodige hardware hoeft er geen rekening gehouden te worden met de levensduur van een server. Hierdoor kan de focus gelegd worden op de eindgebruiker om de levensduur van de toepassing zo groot mogelijk te maken.

# Conclusie

Easyshop maakt het mogelijk om boodschappen te laten leveren door een andere gebruiker. De toepassing voorziet beide doelgroepen van tools om de gewenste activiteiten uit te voeren. Om tot het geleverde eindresultaat te komen, zijn de meest gunstige oplossingen gekozen na het afnemen van een enquête bij potentiële eindgebruikers.

Een eerste aspect is dat de applicatie toegankelijk moet zijn op een breed spectrum van apparaten. Omwille van deze reden is Easyshop een webapplicatie die kan geopend worden in alle populaire browsers indien het toestel over een internetverbinding beschikt.

Een tweede pijler die van groot belang is voor de eindgebruiker is dat de toepassing snel moet zijn. Door het gebruik van React laadt de website dynamisch en worden enkel de componenten die gewijzigd worden opnieuw gerenderd. Hierdoor voelt de website snel aan en hoeft de gebruiker nauwelijks te wachten op het laden van de verschillende webpagina’s. Bovendien beschikt de server waarop de website gehost wordt over voldoende kracht om meerdere gebruikers gelijktijdig te bedienen.

De applicatie moet ook eenvoudig te bedienen zijn zodat mensen van alle leeftijdsgroepen gebruik kunnen maken van de toepassing. Er is gekozen voor een minimalistisch design, waar enkel de nodige tools worden aangereikt om de gewenste acties uit te voeren. Alle informatie wordt doorgespeeld naar de frontend aan de hand van API-calls vanaf de API die op PHP draait.

Een vierde vereiste is het aanbieden van voldoende betaalmogelijkheden. Door het gebruik van de provider Mollie is er voldoende keuze aan betaalmogelijkheden voor de eindgebruiker. Een voordeel hierbij is dat de transactiekost laag is per betaling. Nadelig is dat de versleuteling van de persoonlijke gegevens op een veiligere manier kan dan met het huidige SHA1-HMAC-encryptie. Toch voldoen alle aangeboden betaalmethoden aan de norm volgens de Payment Card Industry Data Security Standard.

Tot slot staat de beveiliging van persoonlijke gegevens centraal. Paswoorden en andere gevoelige data moeten geëncrypteerd in de databank opgeslagen worden. Daarnaast is de datatransmissie versleuteld aan de hand van een JSON Web Token die werkt op basis van het SHA256-algoritme.

Volgens de huidige staat van de applicatie kan deze nog niet gebruikt worden door derden, dit omwille van een aantal gebreken. Op vlak van beveiliging is er nood aan encryptie van persoonlijke gegevens in de database. Ook de betalingen kunnen naar de toekomst toe op een veiligere manier versleuteld worden. Daarnaast is er nood aan de implementatie van een aantal functies zoals het registreren van gebruikers vooraleer gebruikers effectief gebruik kunnen maken van de services van Easyshop. Er zijn ook een aantal uitbreidende functies zoals het aanmaken van een reviewpagina dat meer vertrouwen moet scheppen tussen de gebruikers onderling en het handmatig aanvullen van winkels en producten door winkeleigenaars.

# Nawoord met kritische reflectie

We kunnen zeggen dat we tevreden zijn met het behaalde eindresultaat. Het was technisch een zeer uitdagend project met veel nieuwigheden die voor de nodige problemen zorgden. Ondanks dat het project nog niet klaar is om op de markt gebracht te worden, zijn we blij met de vaardigheden die we hebben verworven tijdens het uitwerken van dit project. Als team hebben we goed gefunctioneerd door de taken onderling te verdelen en kunnen we stellen dat we een goede basis hebben gelegd voor de verdere ontwikkeling van de applicatie.

Het was een leerrijk project. We hebben zowel reeds geziene leerstof uit de richting ICT toegepast om onderzoek te verrichten naar potentiële eindgebruikers als nieuwe programmeertalen geleerd doormiddel van onlinebronnen om de webinterface op te bouwen.

Het opstellen van het wetenschappelijk rapport vroeg veel tijd en was zeker niet makkelijk voor dit onderwerp, maar deze vaardigheden zijn zeker van essentieel belang in de bedrijfswereld.

Ook het in team samenwerken en het halen van vooropgestelde deadlines was een uitdaging. De officiële deadlines werden steeds gehaald, maar de vooropgestelde persoonlijke deadlines werden af en toe overschreden. Er werden ook een aantal ideeën geschrapt om zeker te zijn dat we een werkend geheel zouden kunnen leveren. Hierdoor zijn er enkele geavanceerde functies die we in het begin voor ogen hadden niet aanwezig in de huidige versie van de applicatie.

De samenwerking met onze mentor hebben we heel positief ervaren. In het begin was het niet zo evident om een moment te vinden waarop iedereen beschikbaar was om samen te komen. Eenmaal dit probleem van de baan was, konden we steeds rekenen op de nodige feedback van onze mentor die ons zeker heeft geholpen in het behalen van het eindresultaat. Ook naast de feedbackmomenten, waren er geregeld momenten waarop we reflecteerden over de resultaten die we tot op dat moment hadden behaald, deze werden gerapporteerd in verslagen (bijlage 3, bijlage 4).

Het was zeer belangrijk om van in het begin duidelijke afspraken te maken omtrent deadlines, planning (bijlage 2) en uitwerking van het rapport. Er is gebleken dat dat zeker nodig is om een goede samenwerking te bevorderen.

Alle teamleden vertoonden soms uitstelgedrag, maar anderzijds beschikte iedereen ook over een groot doorzetvermogen onder stress. Bij toekomstige projecten is dit zeker een aspect dat voor verbetering vatbaar is zodat er voldoende tijd is voor uitgebreide functies te implementeren.

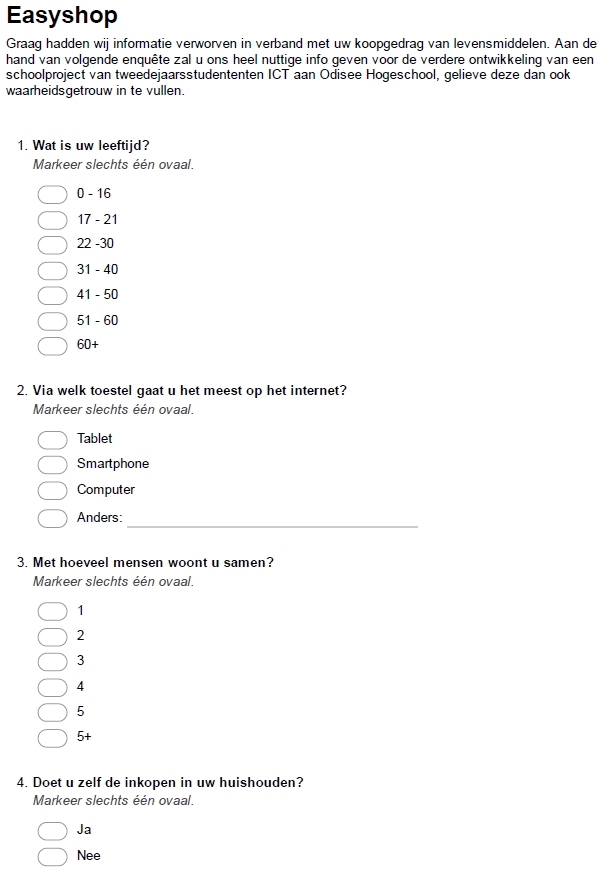
De communicatie werd gedaan via Slack waarin er een chatroom werd opgesteld per categorie. Dit zorgde voor gestructureerde discussies die voor iedereen beschikbaar waren. Daarnaast werd er gebruik gemaakt van GitHub voor het uploaden van het project. Hierdoor kon iedereen op hetzelfde moment aan het project werken zonder dat iemand zich zorgen hoefde te maken over wijzigingen die iemand anders eerder had aangebracht. Wanneer er met meerdere mensen aan één bestand of probleem werd gewerkt, werd er gebruik gemaakt van Skype om zo efficiënt mogelijk te werken.

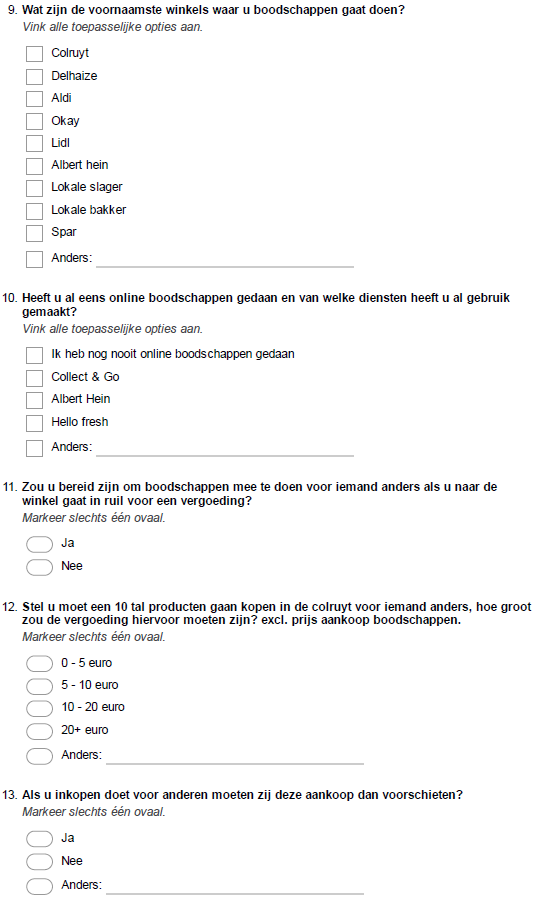
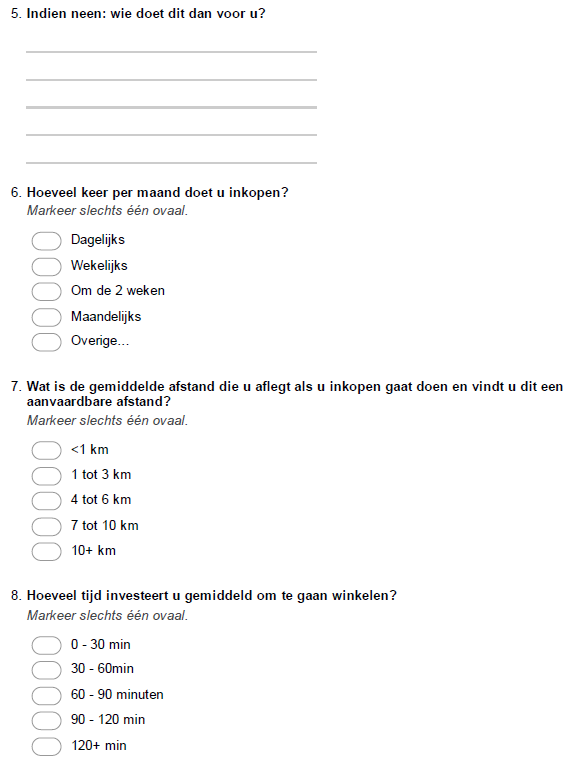
# Literatuurlijst

|  |  |
| --- | --- |
| [1] | W3schools, „W3schools.com,” [Online]. Available: https://www.w3schools.com/. [Geopend 17 maart 2017]. |
| [2] | „HTML5 – Hypertext Markup Language – 5.0,” [Online]. Available: http://www.w3.org/2014/10/html5-rec.html.en. [Geopend 17 maart 2017]. |
| [3] | D. Danailov, „Mentormate,” 10 10 2016. [Online]. Available: https://mentormate.com/blog/polymer-vs-angular-future-web-apps/. [Geopend 17 maart 2017]. |
| [4] | C. Patel, „Bacany technology,” [Online]. Available: http://www.bacancytechnology.com/blog/angular2-vs-react-vs-polymer. [Geopend 17 maart 2017]. |
| [5] | A. Ivanovs, 9 maart 2017. [Online]. Available: https://colorlib.com/wp/javascript-frameworks/. [Geopend 17 maart 2017]. |
| [6] | J. Tomas, 19 september 2015. [Online]. Available: https://conkreetinternet.nl/wat-is-een-api/. [Geopend 17 maart 2017]. |
| [7] | P. Wayner, 9 februari 2017. [Online]. Available: http://www.infoworld.com/article/3166109/application-development/php-vs-nodejs-an-epic-battle-for-developer-mind-share.html. [Geopend 2017 maart 17]. |
| [8] | D. Threlfall, „Worthwhile,” [Online]. Available: https://worthwhile.com/blog/2016/07/19/django-python-advantages/. [Geopend 17 maart 2017]. |
| [9] | Pieter, „Newance,” 27 januari 2016. [Online]. Available: https://www.newance.be/nl/tips/online-betaalsystemen. [Geopend 17 maart 2017]. |
| [10] | Mollie, „Mollie,” Mollie, [Online]. Available: https://www.mollie.com/nl/terms-conditions. [Geopend 2017 maart 17]. |
| [11] | „cPanel,” [Online]. Available: https://cpanel.com/. [Geopend 17 maart 2017]. |
| [12] | „Github Pages,” GitHub, [Online]. Available: https://pages.github.com/. [Geopend 17 maart 2017]. |
| [13] | „DigitalOcean,” DigitalOcean, [Online]. Available: https://www.digitalocean.com/. [Geopend 17 maart 2017]. |
| [14] | MySQL, „MySQL,” Oracle, [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/innodb-benefits.html. |
| [15] | „Mollie,” Mollie, [Online]. Available: https://www.mollie.com/en/. [Geopend 19 april 2017]. |
| [16] | F. O. Source, „React,” Facebook Open Source, [Online]. Available: https://facebook.github.io/react/docs/react-component.html. [Geopend 21 mei 2017]. |
| [17] | threepointone, „Glamor,” [Online]. Available: https://github.com/threepointone/glamor. [Geopend 21 mei 2017]. |
| [18] | nickuraltsev, „Axios,” [Online]. Available: https://github.com/mzabriskie/axios. [Geopend 21 mei 2017]. |
| [19] | mevdschee, „PHP-CRUD,” [Online]. Available: https://github.com/mevdschee/php-crud-api. [Geopend 21 mei 2017]. |
| [20] | S. Software, „Swagger,” SmartBear Software, [Online]. Available: http://swagger.io/. [Geopend 21 mei 2017]. |
| [21] | Auth0, „JWT,” Auth0, [Online]. Available: https://jwt.io/. [Geopend 21 mei 2017]. |
| [22] | T. Brehm, „Linux Tutorials,” HowtoForge, [Online]. Available: https://www.howtoforge.com/tutorial/install-apache-with-php-and-mysql-on-ubuntu-16-04-lamp/. [Geopend 21 mei 2017]. |
| [23] | Mollie, „Mollie Helppage,” Mollie, [Online]. Available: https://help.mollie.com/hc/nl/articles/214016845-Wat-zijn-de-verplichte-parameters-voor-de-reseller-API-. [Geopend 2017 mei 21]. |
| [24] | OWASP, „Cryptographic Storage,” OWASP, [Online]. Available: https://www.owasp.org/index.php/Cryptographic\_Storage\_Cheat\_Sheet. [Geopend 21 mei 2017]. |
| [25] | E. Webshop, „Betaalmethoden en Betaalproviders,” Easy Webshop, [Online]. Available: https://www.easywebshop.be/doc/betaalmethoden-en-betaalproviders. [Geopend 21 mei 2017]. |
| [26] | D. Danailov, „Mentormate,” 10 10 2016. [Online]. Available: https://mentormate.com/blog/polymer-vs-angular-future-web-apps/. [Geopend 17 maart 2017]. |
| [27] | „W3C CSS2.1 specification for pseudo-elements and pseudo-classes,” [Online]. Available: http://www.w3.org/TR/CSS21/selector.html#pseudo-elements. [Geopend 17 maart 2017]. |

# Bijlagenoverzicht

# Bijlage 1: Enquête koopgedrag





# Bijlage 2: Werkplanning

|  |  |  |
| --- | --- | --- |
|  | Uitwerking | Verslag |
| **Week 1** | Projectvoorstel |  |
| **Week 2** | Afspraak technische mentor  Uitwerking projectvoorstel en planning |  |
| **Week 3** |  | Inleiding en basisopmaak |
| **Week 4** | Afspraak technische mentor | Verbeteren inleiding  Mogelijke oplossingen |
| **Week 5** | Publicatie enquête  Design webinterface  Design Databank | Indienen eerste hoofdstuk |
| **Week 6** | Afspraak technische mentor  Homepagina frontend  Aanmaken databank + test Python-API | Verbeteren eerste hoofdstuk  Gekozen oplossing |
| **Week 7** | Overstap naar PHP-API  Opzetten Ubuntu server  Aanmaken alle webpagina’s Frontend  Routering webpagina’s | Indienen tweede hoofdstuk |
| **Week 8** | Odisee infodag |  |
| **Week 9** | API + API-docs operationeel |  |
| **Week 10** | Troubleshooting connectie maken met backend vanuit frontend  Sampledata in database gegenereerd | Verbeteren hoofdstuk 2 |
| **Week 11** | Dynamische winkel- en productpagina |  |
| **Week 12** | Loginpagina, Winkelmandje, Betaling Mollie  Indienen technisch project | Technische uitwerking, Risicoanalyse, Kostenraming en levensduur, Conclusie, Nawoord met kritische reflectie |
| **Week 13** |  | Nalezen op fouten  Indienen volledig document |

# Bijlage 3: Verslag vergadering 1

**Meeting Easyshop verslag**

Datum: 15/02/2017

Tijd: 11u35 - 12u00

Aanwezigen: Seppe Snoeck, Daan Zwaenepoel, Piet Jacobs, Niels De Bruyne & Katja Verbeeck

Volgende meeting: 23/02/2017 om 11u20

**Agenda**

1. Projectvoorstel
2. Inleiding
3. Varia

**Projectvoorstel**

* Aanpassingen (Daan Zwaenepoel)
  + Boodschappen zouden door onze app kunnen gedaan worden.
  + We proberen deze app te bouwen door het verkennen van technologieën die nieuwe zijn voor ons. … kostenbesparend ...
  + Afkortingen voluit schrijven
  + Niet dat maar zeggen wat. bv. Dat  = onze applicatie
  + Realistisch: mensen de app laten testen.
  + Vaag: duidelijk alles beschrijven

**Inleiding**

* Aanpassingen (Piet Jacobs)
  + Duidelijk 2 partijen / doelgroepen opstellen.
  + Alinea 3 en 4 is te vaag
    - Kan voor elk project gebruikt worden
    - Concreet en specifiek
  + alinea 4 -> oplossingen = alternatieven

**Varia**

* Teamnummer vermelden
* Documenten doorsturen voor de vergadering
* We hebben beiden een onderzoek en uitvoerend project
* Taken en functies verdelen
* Tijdsplanning
  + Duidelijk en concreet voor de komende twee weken.
  + Iedereen moet gestart zijn met iets.
  + Alle deadlines
* App laten testen door buitenstaanders/doelgroep
* Werken met git (Seppe Snoeck)
  + Github repo aanmaken
    - Easyshop-Client
    - Easyshop-Backend
  + Katja Verbeeck toevoegen aan GitHub. @odisee of @gmail.
* Elke 2 weken ook reflecteren op de meeting.
* Soms ook in ‘real life’ samen komen.
* Betaalsystemen
  + Bestuderen en Onderzoeken
  + In verslag schrijven
  + Mango peet (?)

# Bijlage 4: Verslag vergadering 2

**Meeting Easyshop verslag**

Datum: 23/02/2017

Tijd: 11u15 - 11u30

Aanwezigen: Seppe Snoeck, Daan Zwaenepoel, Piet Jacobs, Niels De Bruyne & Katja Verbeeck

Volgende meeting: 09/03/2017 om 11u10

**Inleiding**

* trail en error -> er wordt tijd gespendeerd op de code zo robust te maken en testen gemaakt. (Niels Debruyne)

**Planning**

* Grove lijnen al uittypen
* Mijlpalen aka toestand van het project toevoegen (Daan Zwaenepoel)
  + 25/02 DB Design af
  + 09/03 Enquete af
  + 13/03 Design af
  + 31/03 Frontend af
  + 31/03 API Af
  + 22/04 Odisee infodag, af zonder api van winkels en betaalsysteem
  + 25/05 project af
* een individueel kleur of kolom
* toevoegen van onderdeel kolommen en personen (Daan Zwaenepoel)

**Varia**

* Taak specifiek verdelen.
* 09/03 enquete moet af zijn en verdeel
* evaluatiesheet doorsturen naar Mvr. Verbeeck (Seppe Snoeck)

**Taakverdeling**

* Design: Daan Zwaenepoel
* UX: Daan Zweanepoel
* Frontend: Seppe Snoeck
* Backend: Piet Jacobs
* Administratie: Niels Debruyne

**Volgende meeting: 09/03/2017 om 11u10**