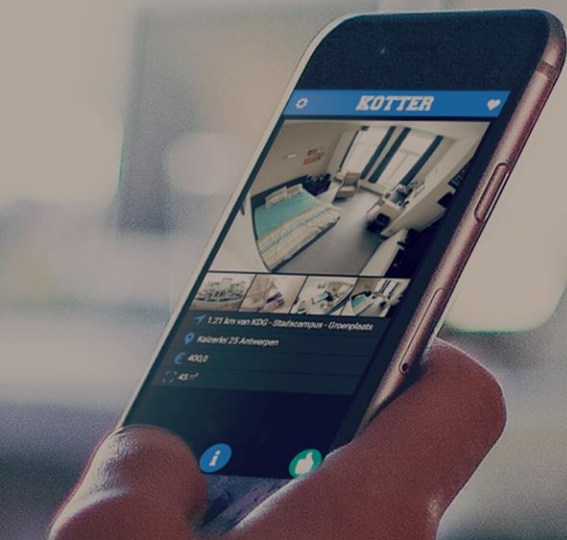




# Eindwerk: Kotter

Voorgedragen tot het behalen van het diploma Multimedia Technology



Maico Paulussen en Matthias Verhoeven

Multimedia Technology

Schooljaar 2014-2015

## Inhoudstafel

1. Omschrijving.....	3
1.1 Concept-doelgroep.....	3
1.2 Functionaliteit .....	4
a) User flow app .....	4
b) User flow website.....	5
1.3 Screenshots .....	6
1.4URL .....	12
2. Planning.....	12
3. Usability test report(s).....	14
3.1 App Website: 5 proefpersonen 16-55jaar .....	14
3.2 Admin paneel .....	15
3.3 App .....	15
4. Technische documentatie .....	16
4.1 Wireframes App .....	16
4.2 Wireframes Website.....	21
4.3 Sitemap.....	24
4.4 Architectuur.....	26
4.5 Database Schema .....	27
4.6 Code.....	28
5. Conclusie .....	35
6. Resources .....	36

# 1. Omschrijving

## 1.1 Concept-doelgroep

Kotter is een app waarmee studenten en ouders van studenten op een leuke manier het ideale kot voor zichzelf of voor hun zoon/dochter kunnen vinden. Bij kotter ben je zeer snel 'up and running'. Zo moet je enkel je login-gegevens, school en prijs ingeven. Hierna kan je al direct onze databank met koten doorlopen. Dit gebeurt via een zeer eenvoudig principe. De gebruikers kunnen elk kot 'liken' of 'disliken'. Dit kan men doen door de icoontjes te gebruiken maar ook door naar links of naar rechts te swipen. De gebruikers kunnen ook specifiek zoeken door de ingebouwde filters. Zo kan men onder meer de minimumoppervlakte bepalen, de maximum afstand tot de campus aangeven en de maximumprijs instellen. De gebruikers kunnen ook via de app eenvoudig de koteigenaars contacteren, zowel via mail als telefonisch.

Wat Kotter uniek maakt is het concept van de totale prijs waarbij geen extra kosten komen op het einde van de maand. Antwerpse studenten en hun ouders gaven immers aan dat de 'extra kosten' één van de onaangenaamste dingen zijn in de zoektocht naar een kot<sup>1</sup>. Sommige verhuurders kiezen ervoor deze kosten uit de advertenties weg te laten, wat de kandidaat-huurder een vertekend beeld geeft over het totaalpakket van de maandelijkse lasten. Andere koteigenaars kiezen er dan weer wel voor om de totale prijs met kosten inbegrepen weer te geven. Dit heeft op zijn beurt dan weer het gevolg dat het vergelijken van koten, en meer bepaald de huurprijzen ervan, alsmaar moeilijker wordt. Hier heeft Kotter op ingespeeld. Ofwel toont de Kotter-app de vaste totale prijs van een kot, ofwel verschijnt er een schatting van de totale prijs. Dit laatste kan gebeuren indien de extra kosten variëren van het gebruik van de huurder in kwestie. Op die manier weet een kandidaat-huurder hoeveel een kot in totaal gaat kosten. Dit voorkomt ook dat een kotbaas een kot kan adverteren voor €450 en dat er dan nog eens €150 maandelijkse kosten bijkomen.

Bij Kotter is er ook een website die als promo dient. Deze website biedt achtergrondinformatie over de Kotter-app. Hier kunnen eveneens geïnteresseerde koteigenaars contact met ons opnemen om hun koten in onze databank te plaatsen. Wanneer we dit accepteren en alle praktische zaken overeengekomen zijn, sturen wij hen inloggegevens door voor het koten-beheerpaneel. Hierop kunnen ze dan hun eigen koten gaan toevoegen. Deze worden altijd in de eerste plaats door het Kotter-team gereviewd. Pas wanneer deze koten goedgekeurd zijn, zullen deze in de app zichtbaar gemaakt worden voor kandidaat-huurders. Van een kot dat al een tijdje in de app zit, kan men bovendien steeds het aantal likes terugvinden in het koten-beheerpaneel.

---

<sup>1</sup> Dit is gebleken uit een (informele) rondvraag naar ergernissen van door ons gekende Antwerpse studenten bij hun zoektocht naar een kot.

## Commercieel aspect (theoretisch)

*“Antwerpen telde vorig academiejaar meer dan 44.000 studenten, van wie er 7500 op kot zitten. Traditioneel start de zoektocht naar een studentenkamer al in mei of juni, maar hoe langer hoe meer studenten stellen die uit. Niet zo in Leuven en Gent, maar dus wel in Antwerpen, waar het aanbod aan koten ruim geworden is. Te ruim, zo blijkt, want honderden koten staan eind augustus, een maand voor de start van het nieuwe academiejaar, nog leeg”. (Gazet van Antwerpen, 2014).*

De markt voor studentenkoten in Antwerpen kent duidelijk een overaanbod. Eigenaars van studentenkoten moeten daarom andere middelen/kanalen aanwenden om hun koten elk academiejaar verhuurd te krijgen. Voor hen kan de Kotter-app een manier zijn dit verschil tussen vraag en aanbod te overbruggen.

Voor dit eindwerk hebben we samengewerkt met één kot-bedrijf. In de toekomst zullen andere kot-bedrijven contact kunnen aanvragen via de website, namelijk [kotterapp.be](http://kotterapp.be). Deze verhuurders kunnen contact opnemen via email, waarop wij onze prijzen zullen doorsturen.

Een kotbedrijf dat bij ons een aanvraag heeft gedaan gaan we steeds evalueren en vervolgens al dan niet accepteren. Indien we een bedrijf accepteren en de financiële kant is achter de rug, sturen we een login en een paswoord door. Dit wilt echter nog niet zeggen dat koten automatisch in de app terecht komen. De kwaliteit van de koten die Kotter aanbiedt moet namelijk steeds gegarandeerd worden. Om dit te garanderen, zal het Kotter-team koten gaan *approven*. Hierdoor sluipen er geen koten in die niet op hetzelfde niveau zitten. We zijn dan ook echt op zoek naar veilige, mooie en verzorgde koten en willen bijvoorbeeld huisjesmelkers eruit filteren.

Tenslotte kiest Kotter ervoor om het contact tussen huurder en verhuurder te vergemakkelijken. Omdat we de gebruikers niet willen hinderen om op een zo gemakkelijk mogelijke manier contact op te nemen met de koteigenaar/verhuurder, zal deze contactinformatie direct beschikbaar zijn.

## 1.2 Functionaliteit

### a) User flow app

De app is ontwikkeld voor tablets en smartphones en kan gedownload worden via de Apple App Store en de Google Play Store. Via de website vindt men ook links naar beide stores. Registratie is eenvoudig en de enige gegevens die we vragen zijn een emailadres en een paswoord. Na de registratie geeft een gebruiker zijn/haar toekomstige school/campus op en een maximum budget.

That's all! De gebruiker krijgt nu koten te zien met het zelfgekozen budget in de buurt van de gekozen school/campus. Hier kan hij/zij telkens ook de extra info bekijken door op de infoknop te duwen. De gebruiker vindt hier vervolgens de locatie van het kot op een kaart, een beschrijving van het kot, al de features, en de contactgegevens van de beheerder.

Door koten browsen is super eenvoudig. Bij elk kot moet de gebruiker 'liken' of 'disliken'. Een kot disliken wil zeggen dat het verwijderd wordt uit de pool waaruit de gebruiker kan kiezen. Een geliket



kot verschijnt in de favorieten lijst. Zo eindigt de gebruiker uiteindelijk met een lijst van koten die hij/zij geweldig vindt. De resterende koten blijven voorgeschoteld worden aan de gebruiker.

Wanneer de gebruiker een bepaald kot wil gaan bezichtigen of nog wat meer info wil, kan hij zeer eenvoudig de kot-beheerder contacteren door het extra infovenster te openen en op het emailadres te duwen. Dit zorgt ervoor dat er een nieuwe mail wordt opgemaakt met als bestemming de koteigenaar en als onderwerp '*Betreffende: kot adres X*'. De kandidaat-huurder kan er ook voor kiezen meteen te bellen aangezien het telefoonnummer zichtbaar is. Wanneer hij/zij op het telefoonnummer duwt, zal de telefoon dialer openspringen met desbetreffende nummer van de koteigenaar.

Indien de gebruiker dat wenst, kan hij/zij de zoekfilters verfijnen. Via de instellingen kan de gebruiker onder meer het budget, de school/campus, eigen badkamer, eigen keuken, minimum oppervlakte aanpassen.

#### b) User flow website

Stel: een kot beheerder wordt naar onze website verwezen of komt er toevallig op terecht. De kot beheerder kijkt op onze website en installeert de app. Hij/zij heeft interesse om zijn koten aan onze database toe te voegen. Via persoonlijk contact met het Kotter-team kan de kot beheerder een account krijgen. Die zal door ons worden opgemaakt. Hij/zij zal vervolgens een email krijgen met zijn login gegevens en kan aan de slag met koten toe te voegen.

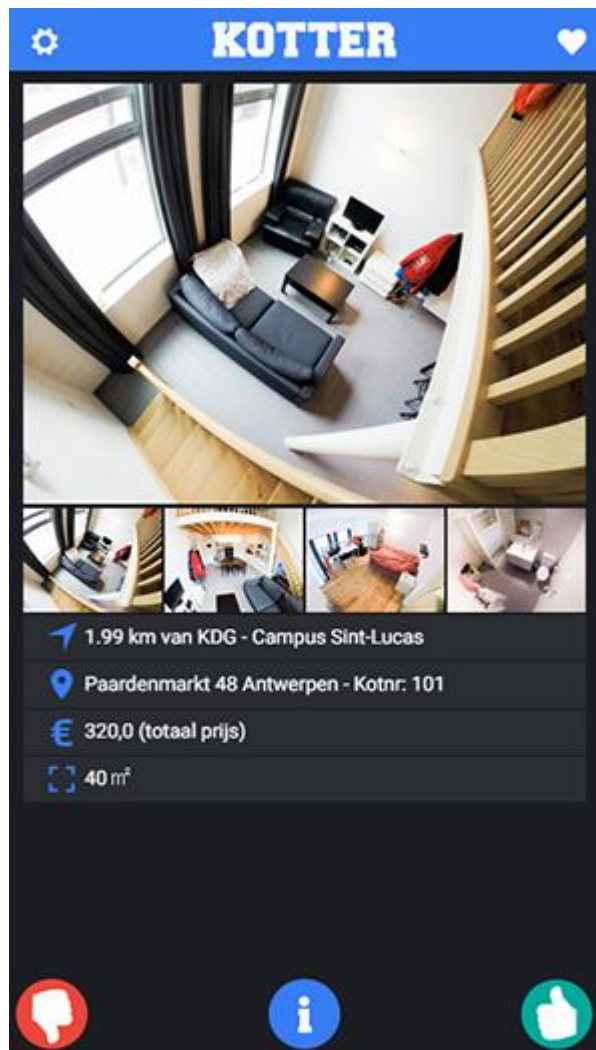
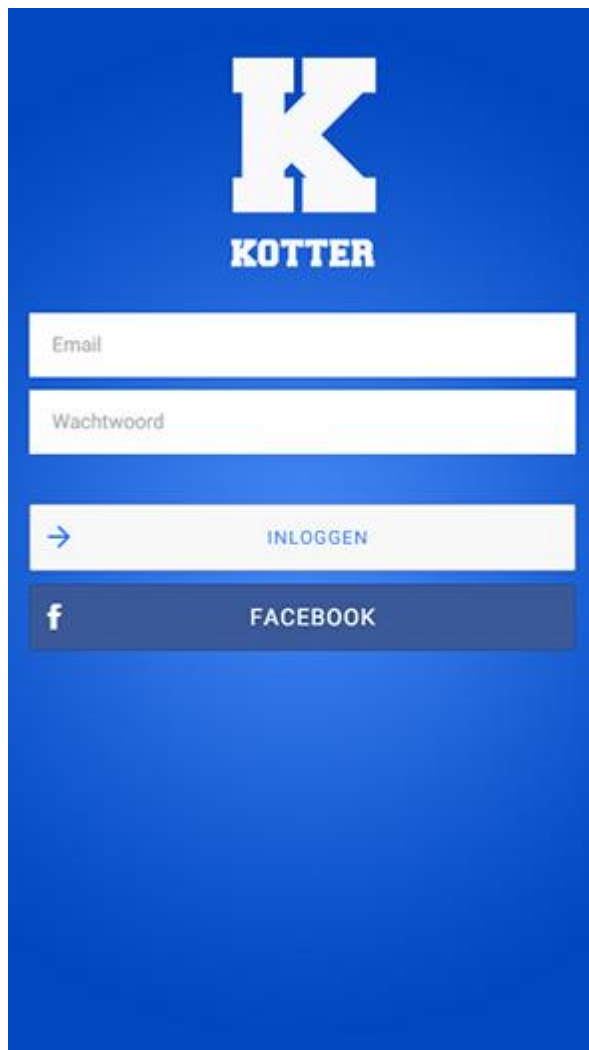
Bij het toevoegen van koten wordt er allerlei data gevraagd zoals onder meer het adres, de contactgegevens, de prijs en de oppervlakte. Bovendien dient de kot beheerder telkens vier afbeeldingen van het kot in kwestie toe te voegen. De prijs moet, zoals reeds vermeld, bovendien de totale prijs zijn. Concreet wil dit zeggen dat de verwarmingskosten, energiekosten en de kosten van het water mee verrekend zitten in deze prijs. Dit geeft een totaalbeeld weer van wat de huurder elke maand dient te betalen.

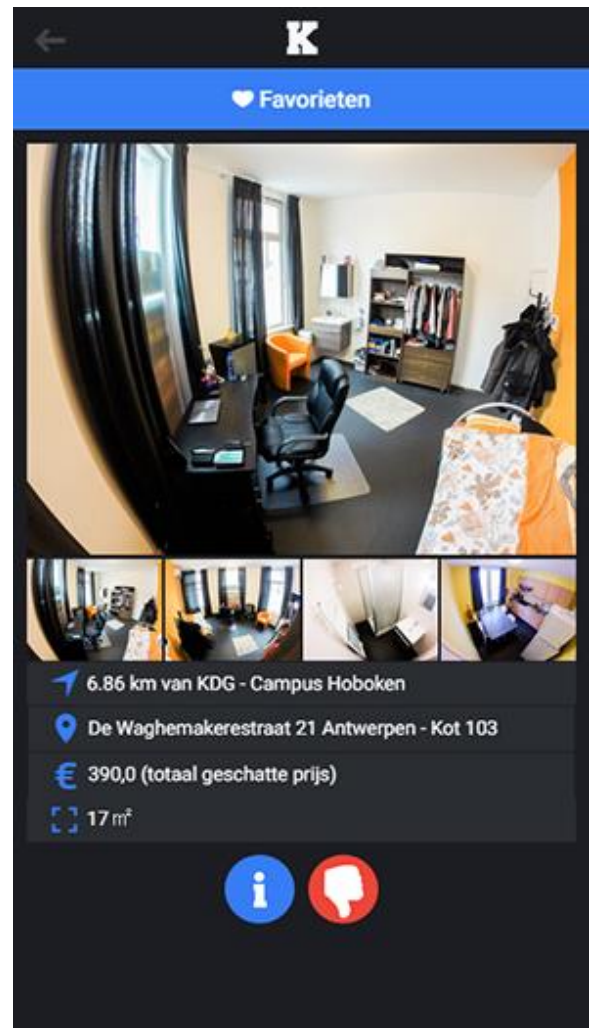
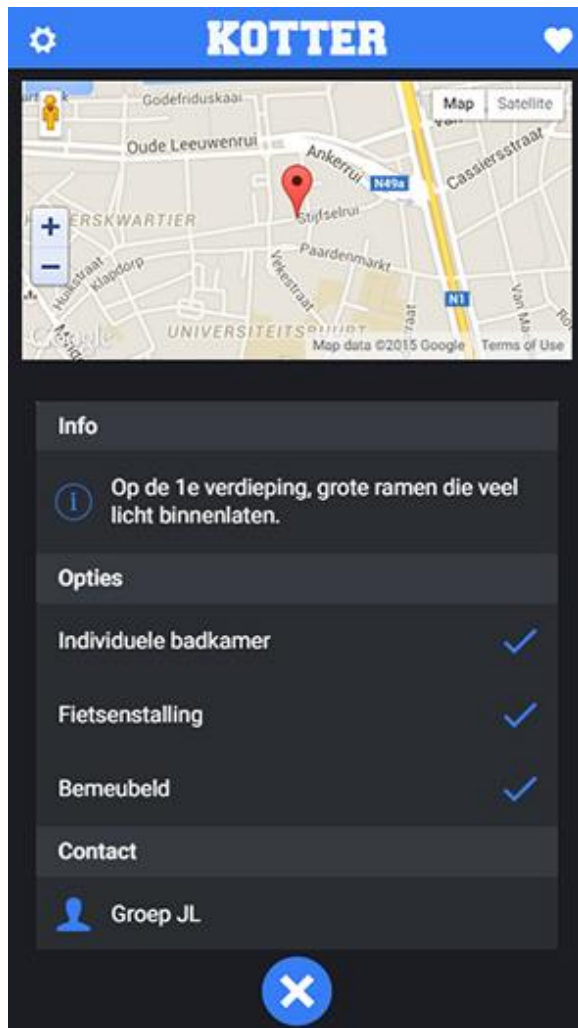
Wanneer deze persoon zijn koten heeft toegevoegd, zullen zij eerst gereviewd worden door het Kotter-team. Pas daarna zullen ze in de app verschijnen. Het Kotter-team zal dan bepalen welke koten worden geaccepteerd of geweigerd. De kot beheerder zal ook steeds zicht hebben op het aantal likes die een kot reeds gekregen heeft. Dit geeft een beeld weer over de populariteit bij de kandidaat-huurders.

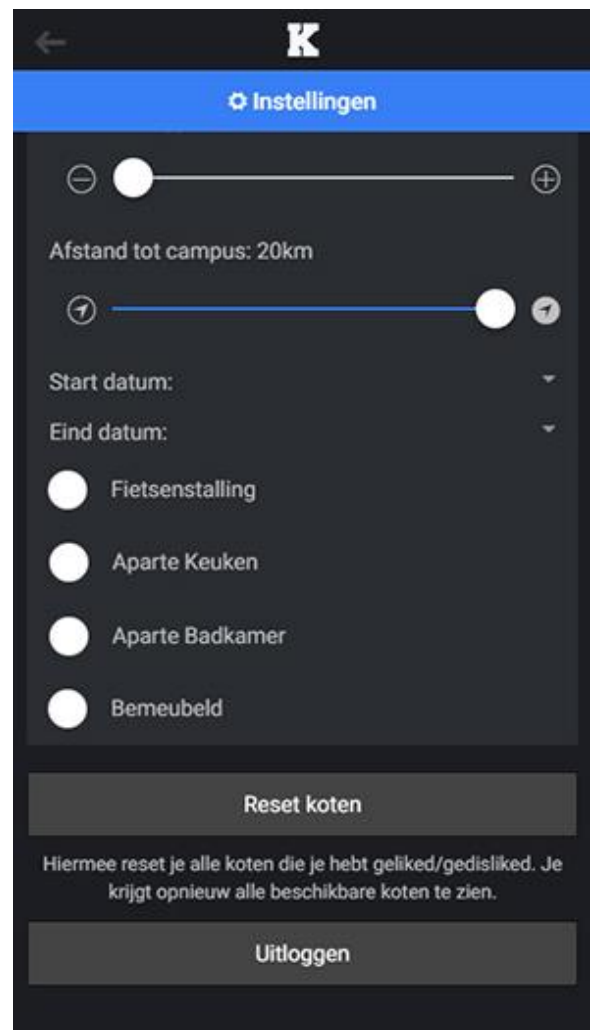
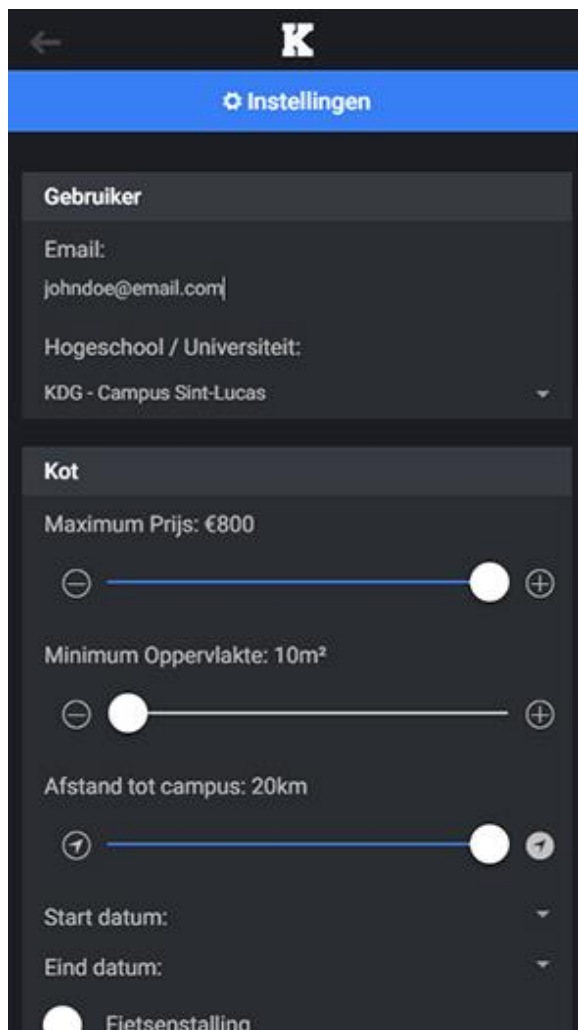
Wanneer de beheerder het kot niet meer wil verhuren kan hij het snel uit onze app verwijderen door het gewoon te deleten uit onze database.

## 1.3 Screenshots

### App









Website KotterApp.be




KOTTER

KOTTER

De app waarmee je je droomkot vindt in Antwerpen.

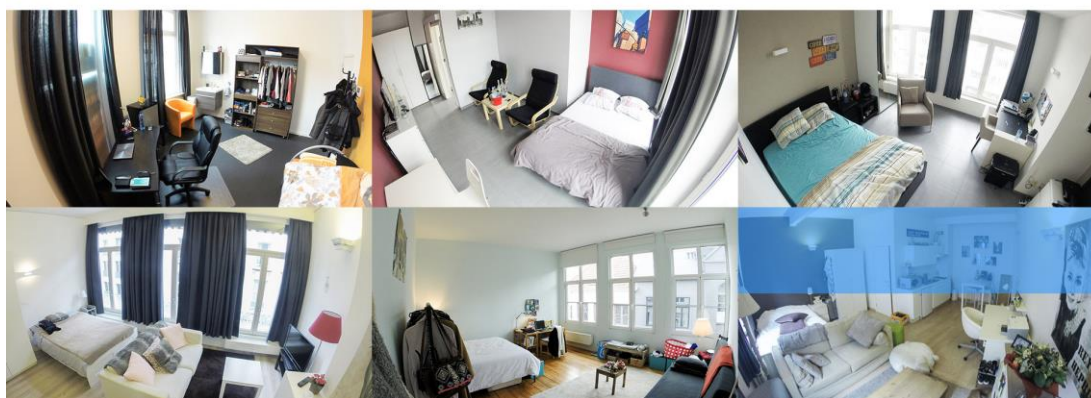


Je droomkot ligt bij ons te wachten.

Kotter is een onafhankelijk medium tussen student en koteigenaar. De kwaliteit van de koten wordt gegarandeerd door manuele selectie. Elk Kotter-kot vermeldt enkel de totaalprijs. Zo kom je niet voor verrassingen te staan en kan je eenvoudiger koten vergelijken.

## Een unieke ervaring

-  Enkel totaalprijzen, alles inclusief
-  Voeg eenvoudig koten toe aan je favorieten
-  Contacteer een kot eigenaar rechtstreeks
-  Meet de afstand naar jou universiteit of hogeschool



## Contacteer ons

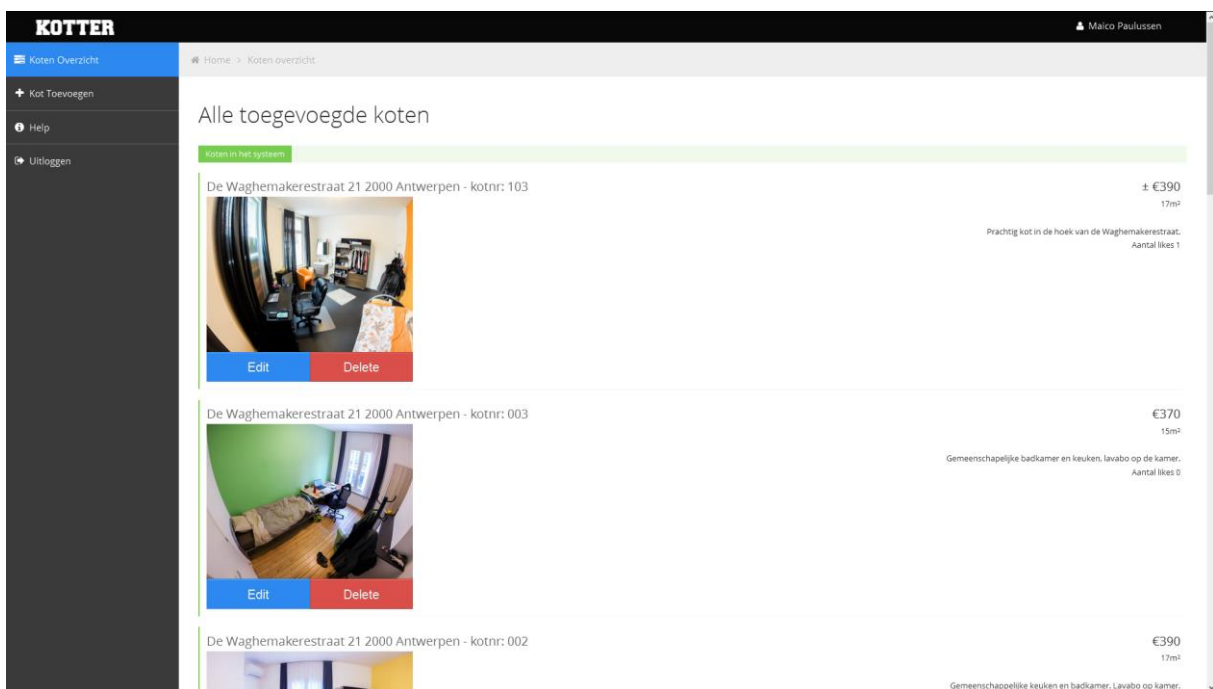
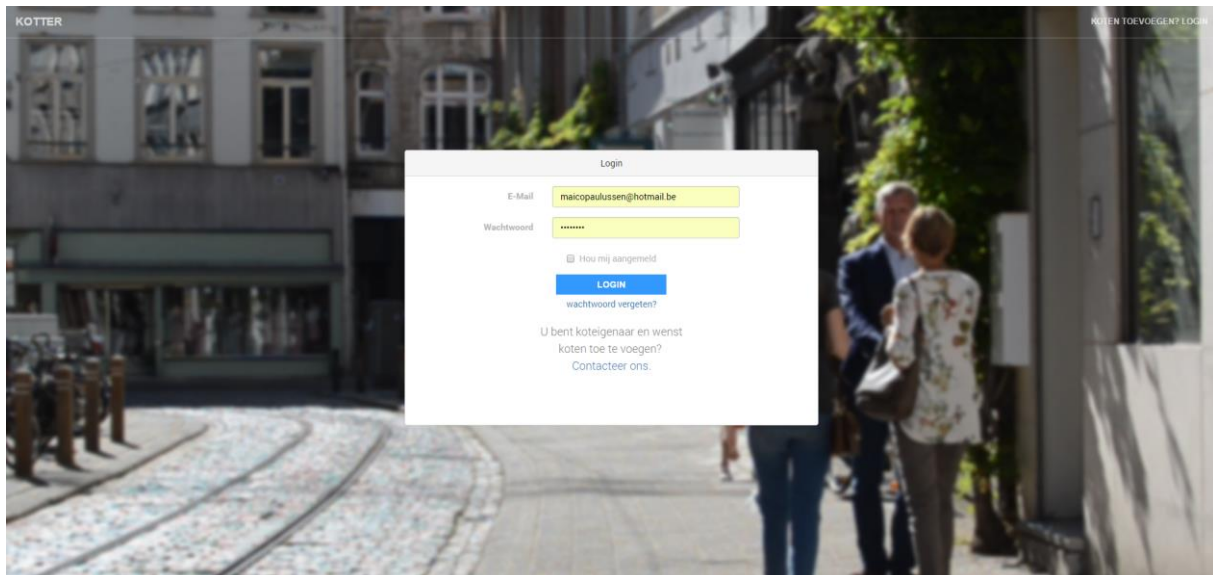
Naam	Boodschap
Email	
Bedrijf	

VERZENDEN



KOTTER © 2013 | Maico Paulussen & Matthias Verhoeven

## Website: Koteigenaar paneel



KOTTER

Maico Paulussen

Koten Overzicht

Kot Toevoegen

Help

Uitloggen

Home > Kot toevoegen

Kot toevoegen

Adres

Gemeente

Straatnaam

Huisnummer

Postcode

Kot Nummer

Contact

Naam

E-mail

Telefoonnummer

Informatie

Oppervlakte (in m²)

Prijs (in €) ☐ Schatting

Beschrijving

KOTTER

Maico Paulussen

Koten Overzicht

Kot Toevoegen

Help

Uitloggen

Home > Help & Contact

Help

Verificatiesysteem

Om de kwaliteit van de koten in de applicatie te garanderen, gaat het Kotter-team koten individueel bekijken vooraleer ze in de app worden toegevoegd. Indien een kot aan alle voorwaarden voldoet, zullen wij het goedkeuren en zal het kot in de app verschijnen. Een verificatie gebeurt binnen 24u. Terwijl een kot gereviewed wordt, zal het opgelijst worden onder 'in review' in het kotenoverzicht. Indien een kot wordt afgekeurd zullen wij een reden geven waarom. Het kot wordt dan opgelijst onder de 'geweerde koten'.

Handleiding

Koten toevoegen

Een kot toevoegen doet u via het menu in de zijbalk genaamd 'kot toevoegen'. Het proces wijst zichzelf uit. Het enige wat we nodig hebben, zijn alle gegevens en 4 kwaliteitsvolle foto's die een beeld geven van het kot. (Een foto met een breedhoeklens/fisheye heeft de voorkeur aangezien deze meer kunnen laten zien binnenin een kamer.)

**Prijs**

Elk kot moet verplicht de totaalprijs laten zien. Indien een vaste totale prijs niet mogelijk is bij een kot, moet er een geschatte totale prijs gegeven worden (vink hiervoor ook het veld 'schatting' aan). Deze geschatte prijs moet zo realistisch mogelijk zijn. Misbruik hiervan zal worden bestraft.

Koten bewerken

Als u een aanpassing wil doen aan een bestaand kot (inclusief foto's aanpassen, contactgegevens aanpassen,...), dan kan u dat zeer eenvoudig door naar het kotenoverzicht te gaan en onder het kot in kwestie op de 'edit' knop te klikken.

Koten deleten

Indien u een kot permanent wenst te verwijderen uit de app, dan kan u dit doen door naar het kotenoverzicht te gaan en onder het bewuste kot op de 'delete' knop te klikken.

Richtlijnen koten

- Het kot heeft een minimum oppervlakte van 10 m²
- De kotprijs is een totaalprijs met alle kosten inbegrepen. Dit mag ook een schatting zijn
- Het kot bevindt zich in zeer goede staat
- Het kot beschikt over 4 duidelijke foto's

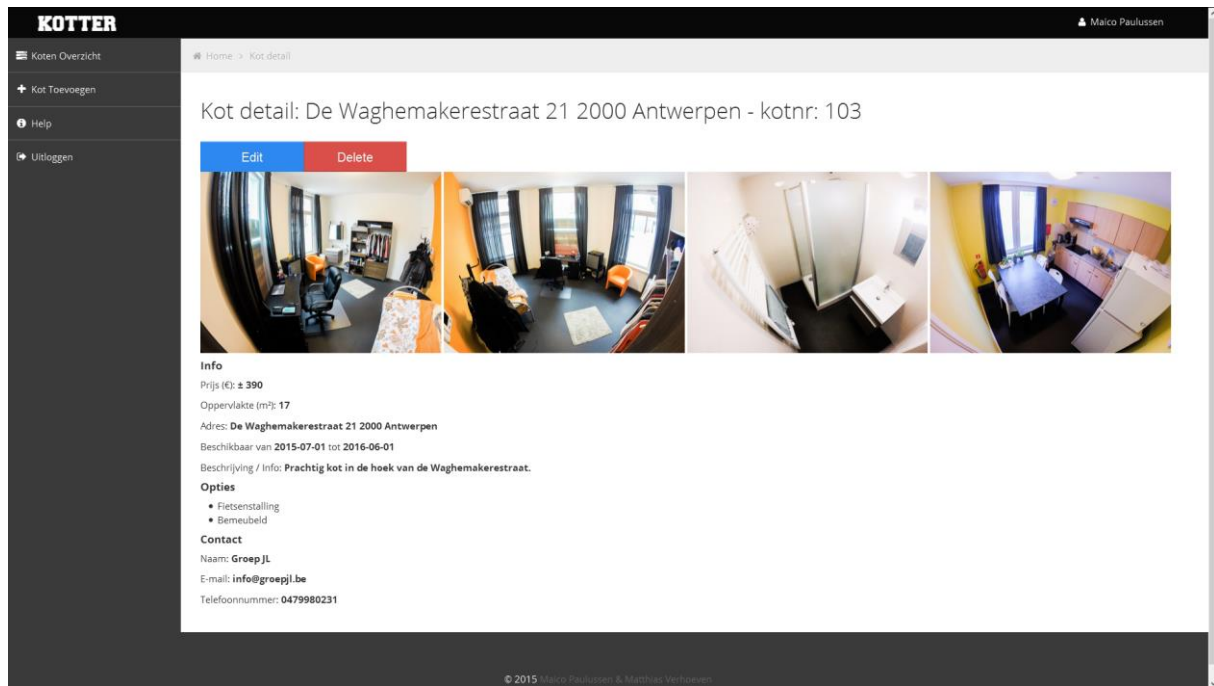
Contact

Voor meer informatie of extra vragen kan u ons steeds contacteren via mail: [info@kotterapp.be](mailto:info@kotterapp.be).

11

KOTTER

MAICO PAULUSSEN & MATTHIAS VERHOEVEN



## 1.4URL

<http://kotterapp.be/>

## 2. Planning

Voorafgaand:

- Brainstormen
- Feedback momenten

Week 1: 9-15 maart

- Database schema (Maico)
- Laravel project aanmaken + login (Maico)
- Laravel koten toevoegen aan db (Maico)
- Contact zoeken kotbedrijf voor fotos (Matthias)

Week 2: 16-22 maart

- Experimenteren met Ionic Framework (Maico)
- Laravel koten overzicht + aanpassen + deleten (Maico)
- Afspraak kotbedrijf (Matthias)

Week 3: 23-29 maart

- IONIC: Tutorial pagina & registratie pagina (Maico)
- Wireframes + design app: 1e versie (Matthias)

Week 4: 30 maart – 5 april

- Wireframes + design app (Matthias)
- Brainstormen naam (Matthias + Maico)
- Facebook register (Maico)
- Info kot ophalen via AJAX (Maico)

#### Week 5: 6-12 april

- Images Kot uitlezen (Maico)
- Basic filters van user (Maico)
- Wireframes + design nieuwe versie (Matthias)

#### Week 6: 13-19 april

- Wireframes + design website (Matthias)

#### Week 7: 20-26 april

- Verschillende kleurencombinaties website (Matthias)
- Afstand berekening van school tot kot (Maico)

#### Week 8: 27 april – 3 mei

- Foto's koten trekken + bewerken (Matthias)
- App images fetchen (Maico)
- Google maps toevoegen app (Maico)
- Filters toepassen bij uitlezen kotten (Maico)

#### Week 9: 4 mei – 10 mei

- Tutorial app (Matthias)
- App design updates (Matthias)
- Filter bug fixes (Maico)
- Favorite pagina + afstand berekening van kotten server-side (Maico)
- Internet connection check (Maico)

#### Week 10: 11-17 mei

- Server aankopen + laravel project online zetten (Maico)
- Extra filters toegevoegd (Maico)
- Google maps bug fix + favorite structuur aanpassen (Maico)
- Campussen antwerpen opzoeken en in db plaatsen (Maico)
- Performance app verbeteren (Maico)
- Front end app (Matthias)
- Nieuw design app (Matthias)

#### Week 11: 18-24 mei

- API in laravel maken voor app (omzetten van basic php files in laravel) (Maico)
- Tinder styled swipe cards toegevoegd (Maico)
- Loading spinner toevoegen (Maico+Matthias)
- Fb login via andere plugin (Maico)
- Feedback website promotor (Maico+Matthias)



- Timelapses shooten (Matthias)
- Front-end fixes website (Matthias)
- Start ontwikkeling specifiek iOS (Matthias)

#### Week 12: 25-31 mei

- Database shema updaten (Maico)
- Tinder swipe card fix (Maico)
- Login + logout toegevoegd app(Maico)
- Reset favorite koten(Maico)
- Timelapses shooten (Matthias)
- Scènes shooten (Matthias)
- Help beheerderspaneel (Matthias)
- Front-end app (Matthias)
- Finale versie promofilm script (Matthias)
- Stand van zaken + to do (Matthias)
- Usability testing website (Matthias)

#### Week 13: 1-7 juni

- Usability testing app (Matthias)
- Laatste timelapses shooten (Matthias)
- Bug fixes (Maico)
- Contact pagina toevoegen website (Maico)
- Admin paneel koten in laravel(Maico)
- iOS bug fixes (Matthias)
- Admin paneel front-end (Matthias)
- Alle HTML valid maken (Matthias)

#### Week 14: 8-14 juni

- Feedback promotoren
- Laatste scenes filmen promofilm (Matthias)
- Monteren promofilm (Matthias)
- Feedback aanpassingen doorvoeren(Maico+Matthias)
- Code nakijken + opruimen (Maico+Matthias)
- Aan bundel werken (Maico+Matthias)
- Afspraak met kotbedrijf (Matthias)

#### Week 15: 15, 16, 17 juni

- Voorbereiding presentatie (Maico+Matthias)
- Final touches bundels (Maico+Matthias)
- Bundels laten bundelen (Maico+Matthias)

## 3. Usability test report(s)

### 3.1 App Website: 5 proefpersonen 16-55jaar

#### Rol student/ouder

Vraag: Is het duidelijk waarover de website gaat? Zo ja, waarover gaat de website?

Alle proefpersonen wisten onmiddellijk waarover de website ging toen ze de homepagina zagen.

Taak: App downloaden

Scenario: Download de app voor jou smartphone

Dit verliep moeiteloos voor alle proefpersonen.

### 3.2 Admin paneel

#### Rol koteigenaar/bedrijf

Taak: Account aanvragen

Scenario: Je bent een koteigenaar en bent geïnteresseerd in onze app. Contacteer ons.

1 van de 5 proefpersonen vond niet onmiddellijk de contact optie, suggestie van proefpersoon:

Achtergrond al direct wit opvullen voor meer visibiliteit.

Suggestie van andere proefpersoon: Wat als een ouder/student een vraag heeft over de app.  
(enkel contactoptie voor koteigenaars)

Taak: Inloggen

Scenario: Je account is aangemaakt en je hebt de registratiedetails via mail ontvangen. Log in op de website.

Dit verliep moeiteloos voor alle proefpersonen.

Taak: Kot aanmaken

Scenario: Je bent ingelogd en wilt je eerste kot toevoegen. (doe dit met je eigen thuisadres)

Dit verliep moeiteloos voor alle proefpersonen.

Taak: Kot editen

Scenario: Je eerste kot is toegevoegd maar er is toch een fout ingeslopen. Corrigeer één van de gegevens.

Dit verliep moeiteloos voor alle proefpersonen.

Suggestie proefpersoon: Eerst straatnaam en nummer, dan pas gemeente en postcode.

Taak: Kot deleten

Scenario: Je gaat je kot verkopen, het is dus niet meer te huur voor studenten. Verwijder het uit de app.

Dit verliep moeiteloos voor alle proefpersonen.

Taak: uitloggen.

Scenario: Log uit.

2 proefpersonen moesten langer zoeken, en 1 vond het helemaal niet.

-> Uitlog knop misschien beter in submenu

### 3.3 App

#### Rol student/ouder

Taak: Registreren op app.

Scenario: De app is geïnstalleerd op jou smartphone. Open de app en registreer jezelf.

Dit verliep probleemloos voor alle testgebruikers.

Taak: Koten browsen

Scenario: Je bent ingelogd in de app, like een paar koten die je leuk vindt, dislike koten die je niet leuk vindt.

Alle proefpersonen voerden dit zonder problemen uit.

Taak: Info

Scenario: Je hebt een kot gevonden waar je helemaal weg van bent. Contacteer de eigenaar.

Dit verliep moeiteloos voor alle proefpersonen.

Taak: Favorieten

Scenario: Bekijk je favoriete koten.

Dit verliep moeiteloos voor alle proefpersonen.

Taak: Filters

Scenario: Je droomkot heeft misschien wel een aparte badkamer, of een aparte keuken.. Zorg ervoor dat je alleen maar koten te zien krijgt naar jou voorkeuren.

Dit verliep moeiteloos voor alle proefpersonen.

Taak: Reset koten

Scenario: Je beseft dat je wat te vluchtig door de koten heen bent gegaan en wilt alle koten terug resetten. Reset alle gelikete/gedislikete koten.

Dit verliep moeiteloos voor alle proefpersonen.

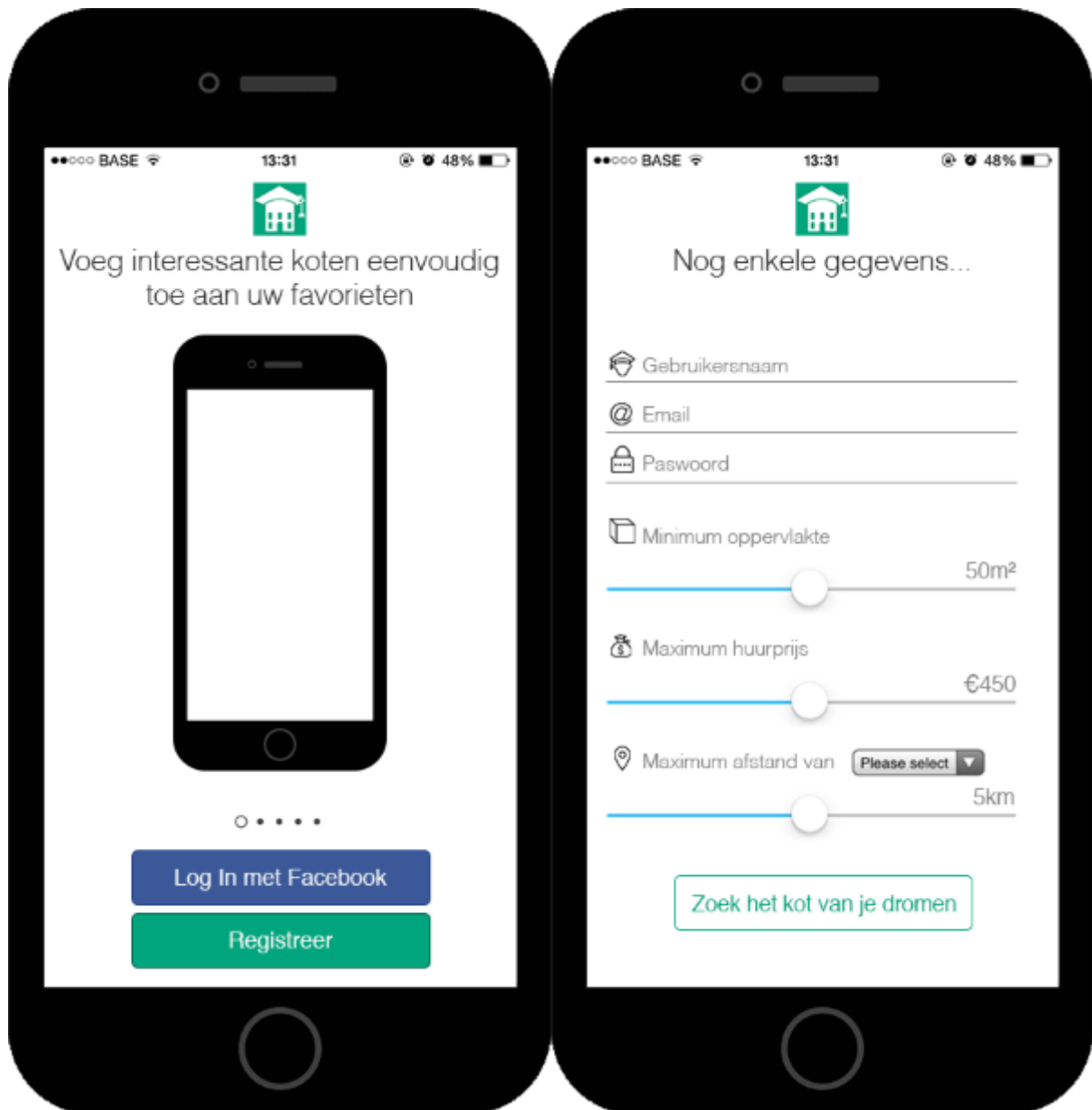
Taak: Uitloggen

Scenario: Meld jezelf af in de app.

Dit verliep moeiteloos voor alle proefpersonen.

## 4. Technische documentatie

### 4.1 Wireframes App

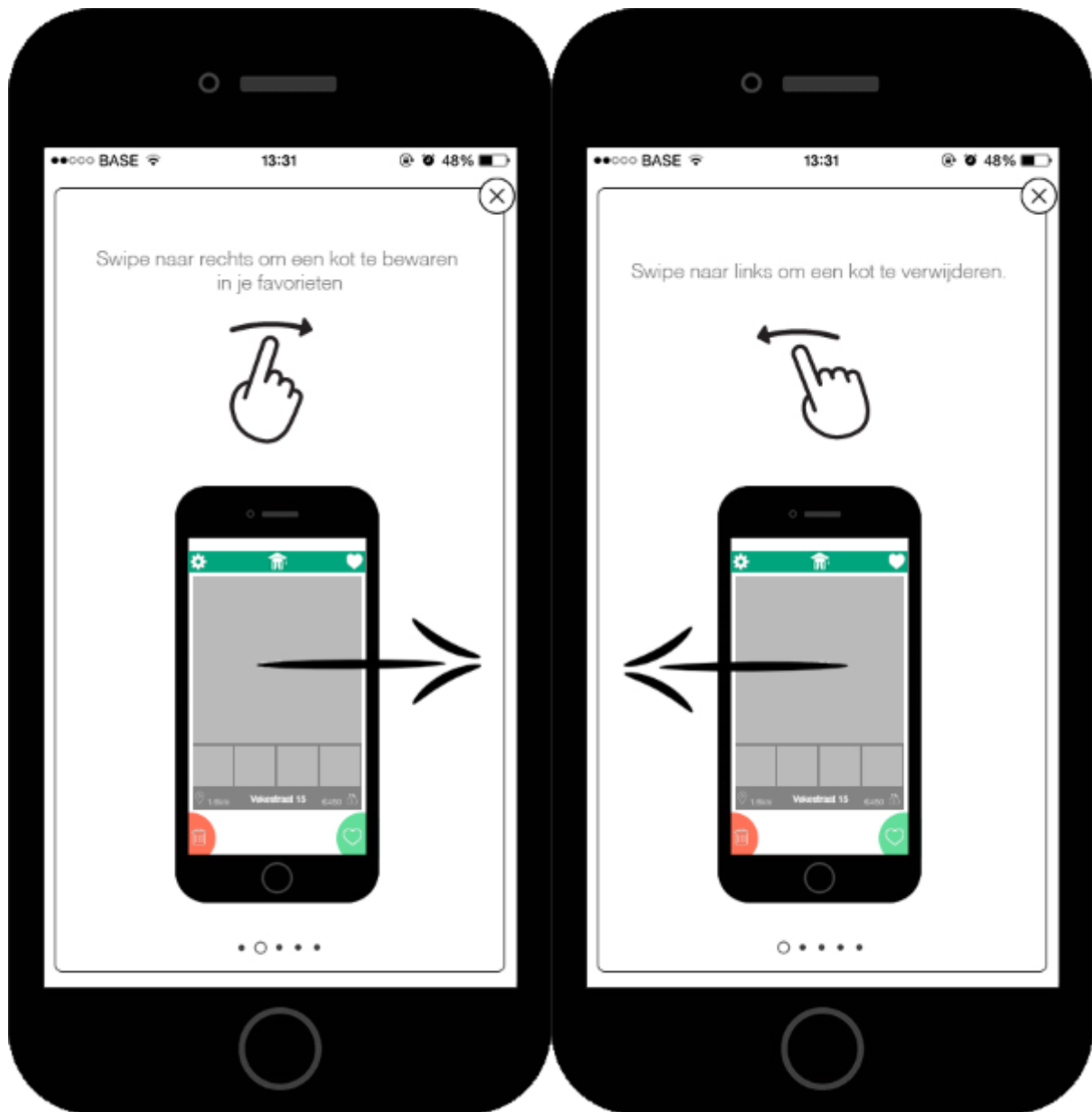


Beginscherm:

- Slogan
- Inlog/registreer opties (+facebook)
- Start tutorial

Registratie gegevens:

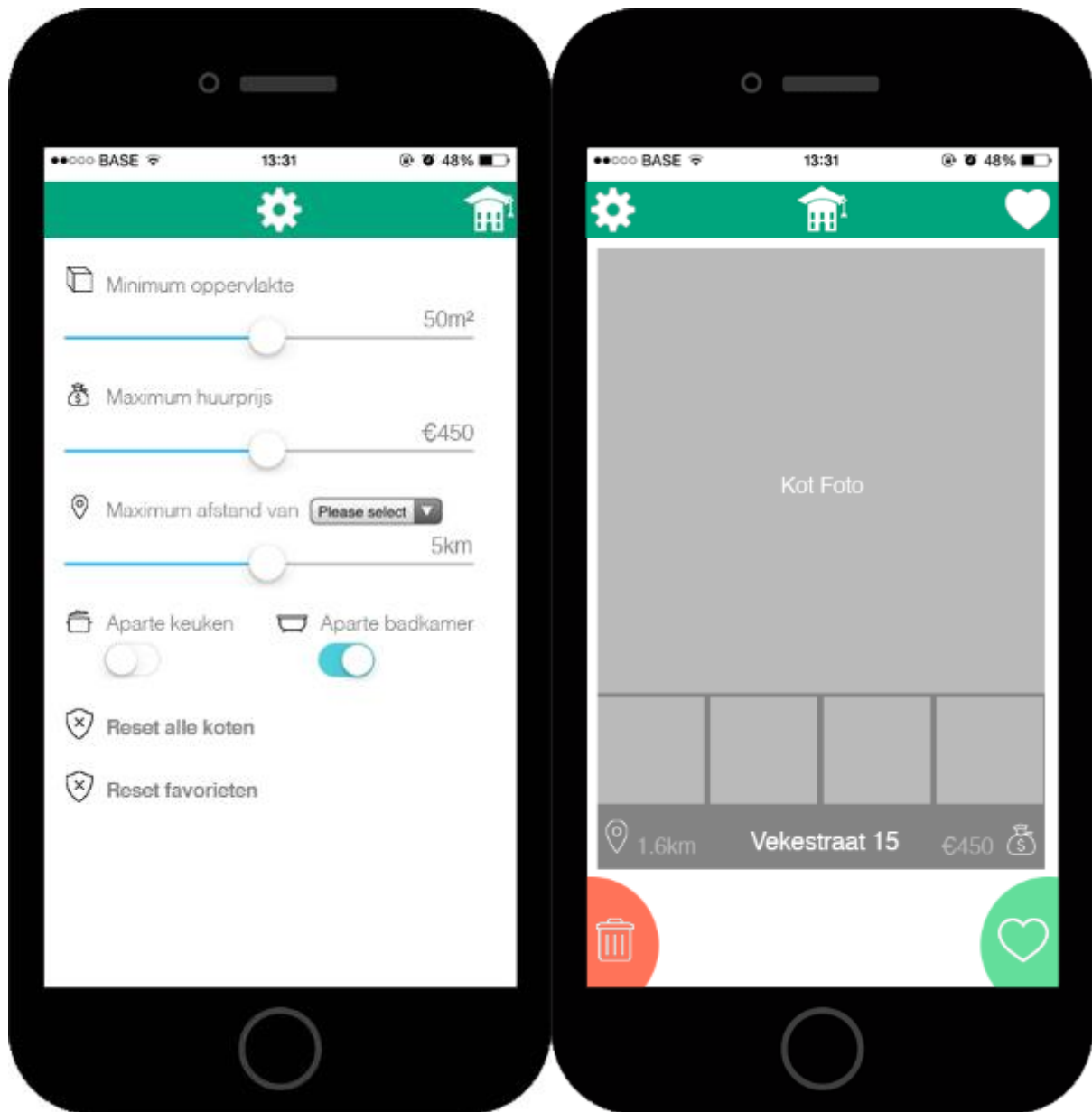
- Gebruikersnaam, email, paswoord
- Minimale oppervlakte
- Maximum huurprijs
- Maximale afstand van ...



Tutorials:

- Basisuitleg applicatie
- Swipe gestures





Instellingen:

- Extra opties zijn hier te zien: Meer opties dan bij initiële registratie
- Mogelijkheid om favorieten te resetten, en alle koten te wissen

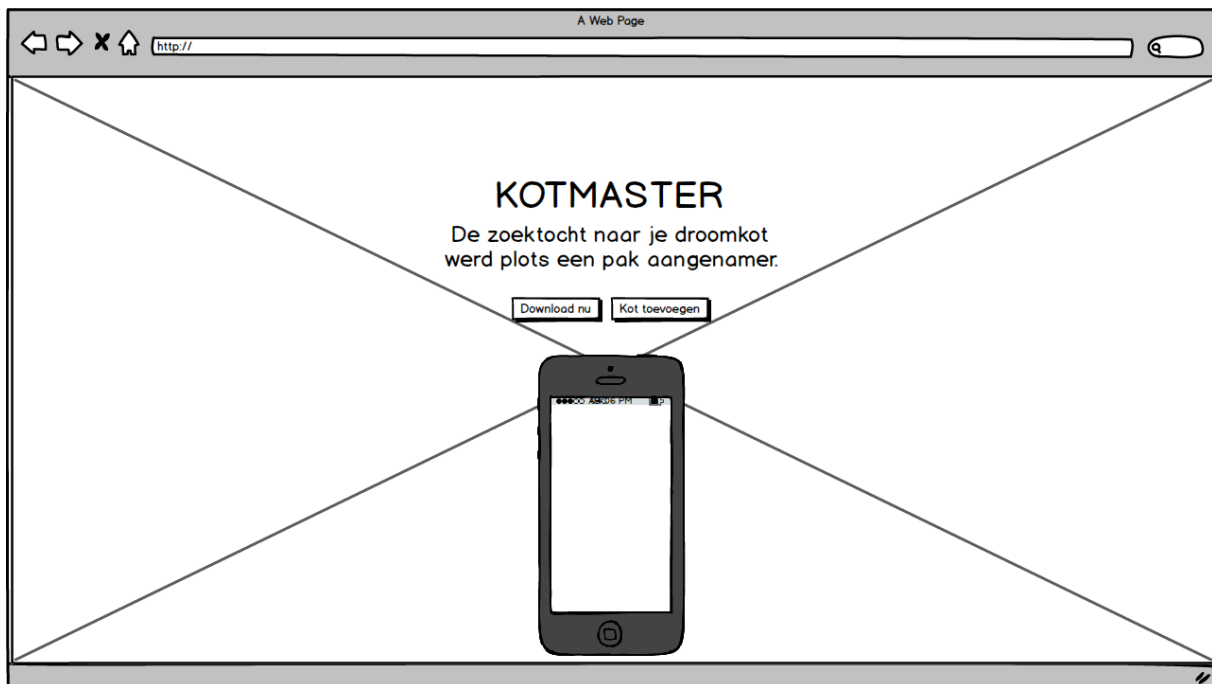


Favorieten:

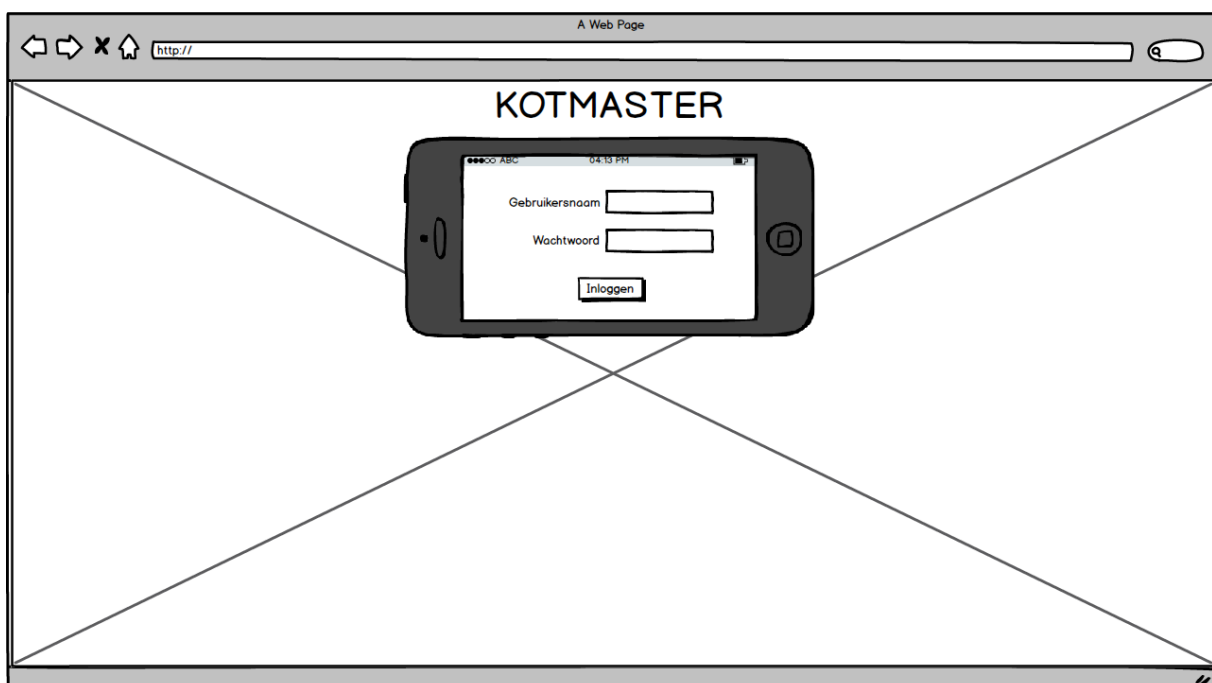
- Beheer favorieten
- Verschillende weergave opties
- Mogelijkheid om te her-selecteren tussen favorieten

## 4.2 Wireframes Website

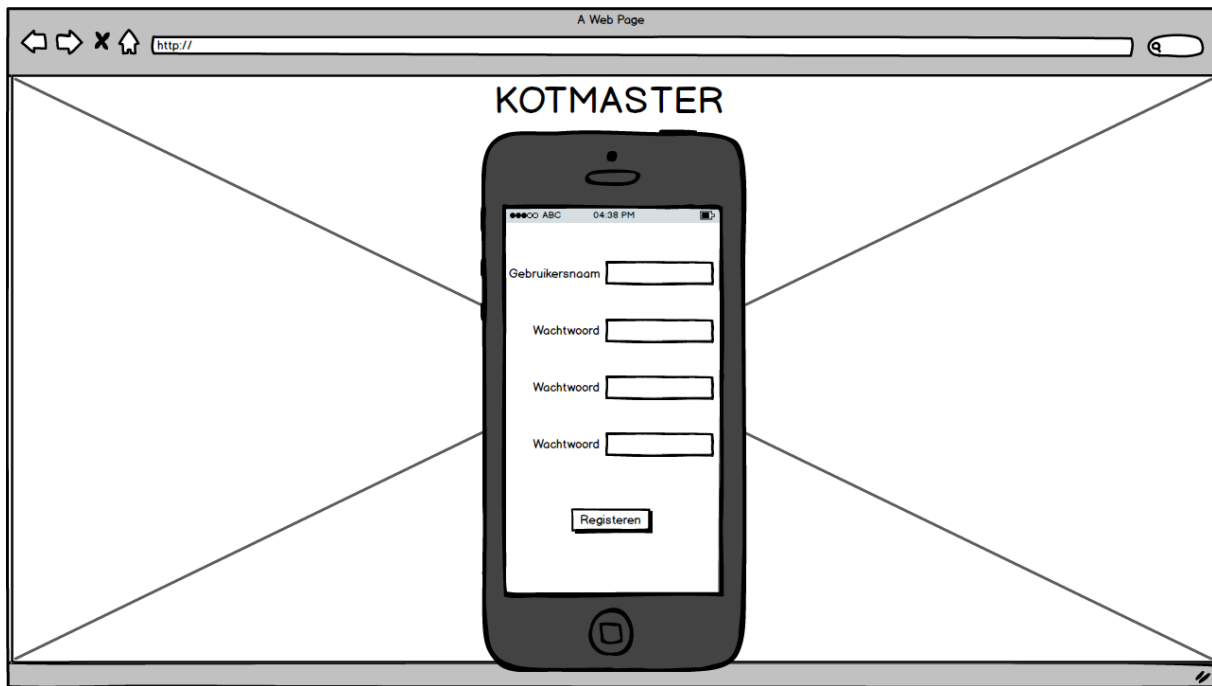
### Oorspronkelijke Balsamiq wireframes



Landing-page: Naam + Slogan + Download button + Login button

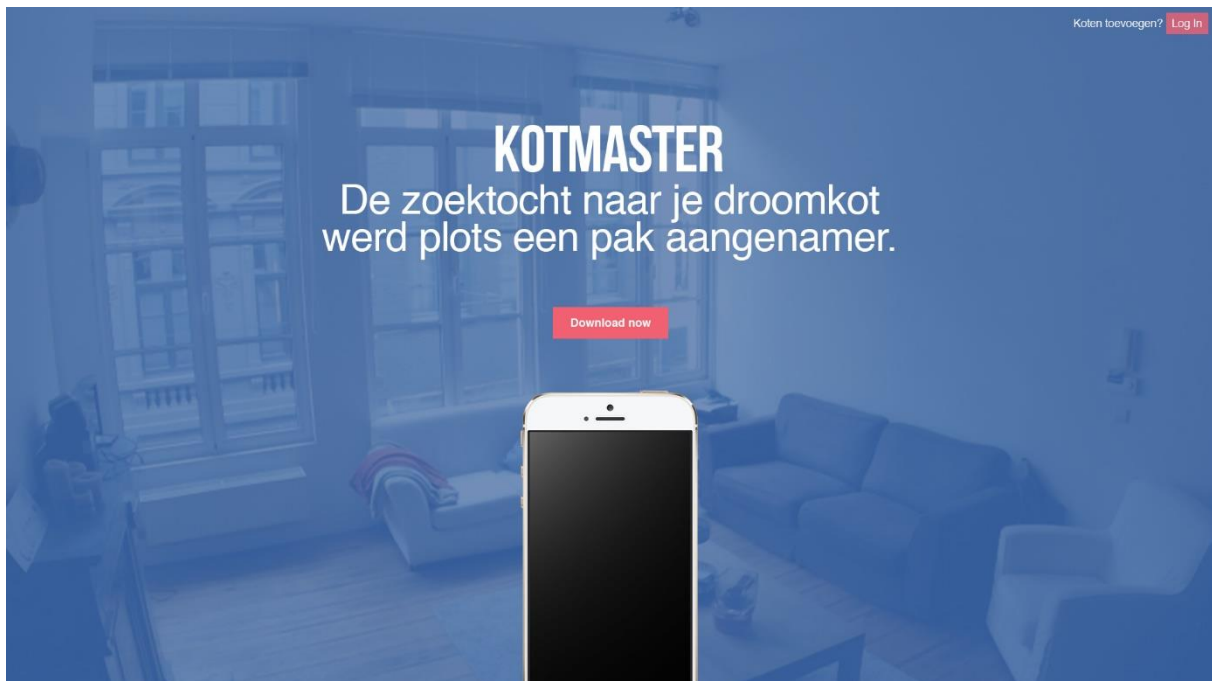


Login scherm in iPhone mockup

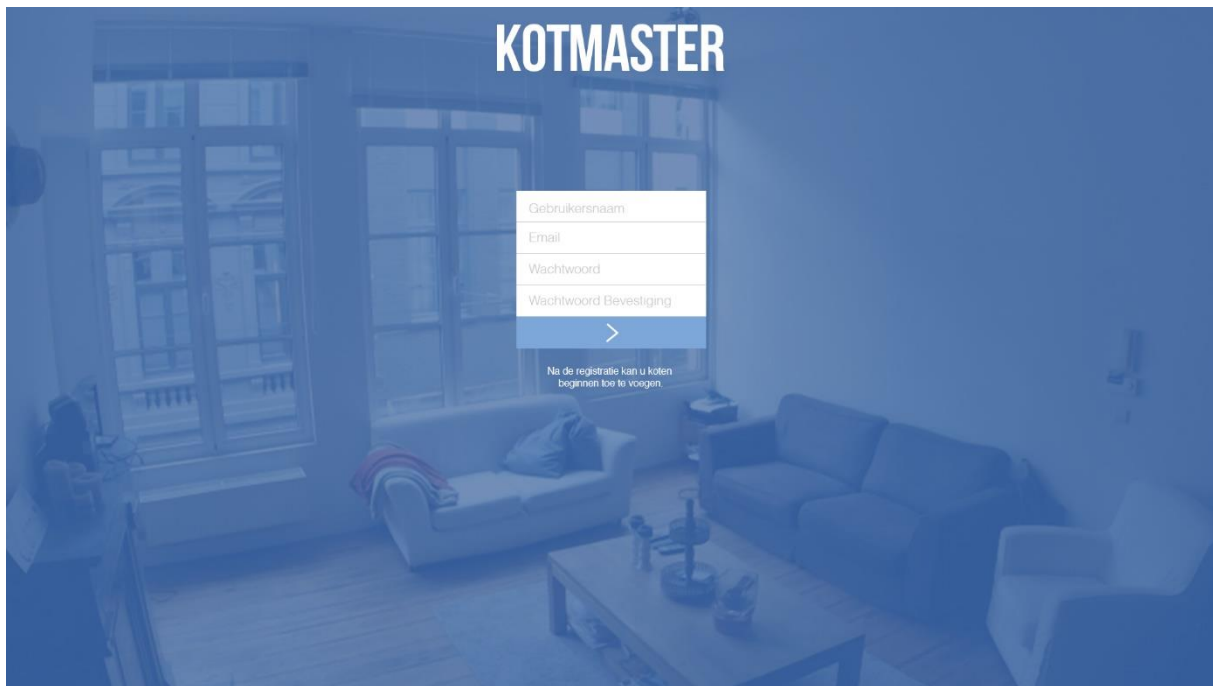


Registratie scherm in verticale iPhone mockup

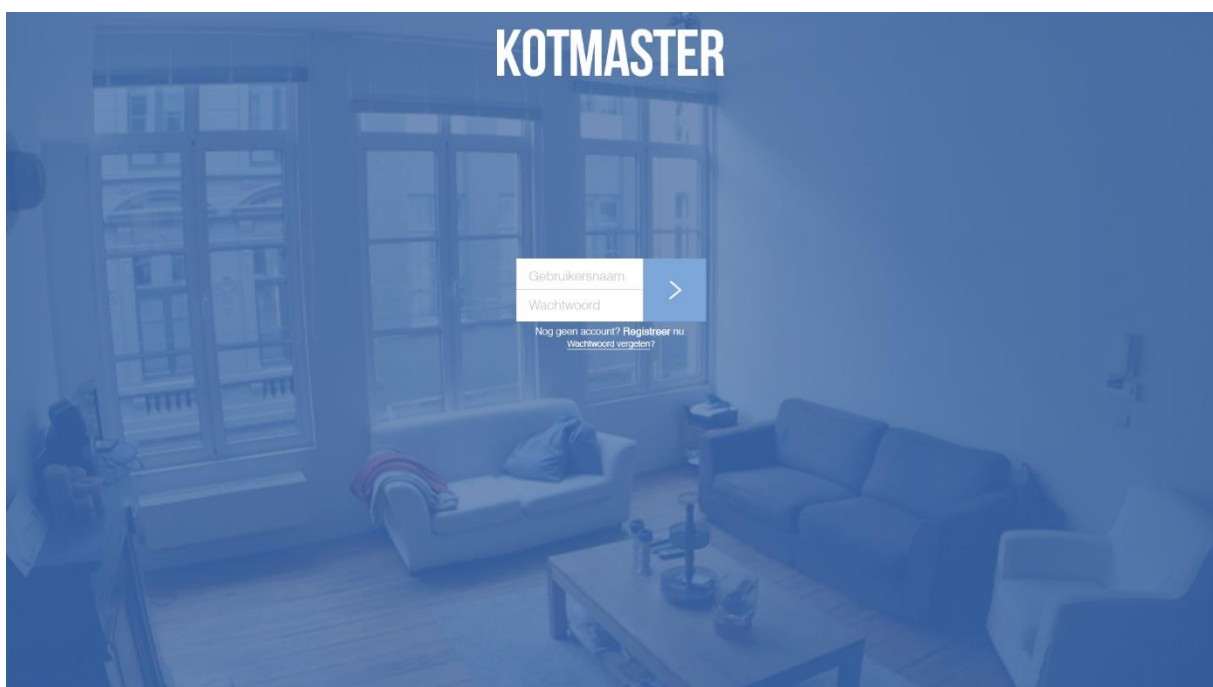
### Photoshop wireframes



Landing-page: Naam + Slogan + Downloadknop + iPhone mockup  
Achergrondafbeelding = luxueus kot



Registratiescherm op zelfde scherm als landing-page  
Eenvoudig clean formulier

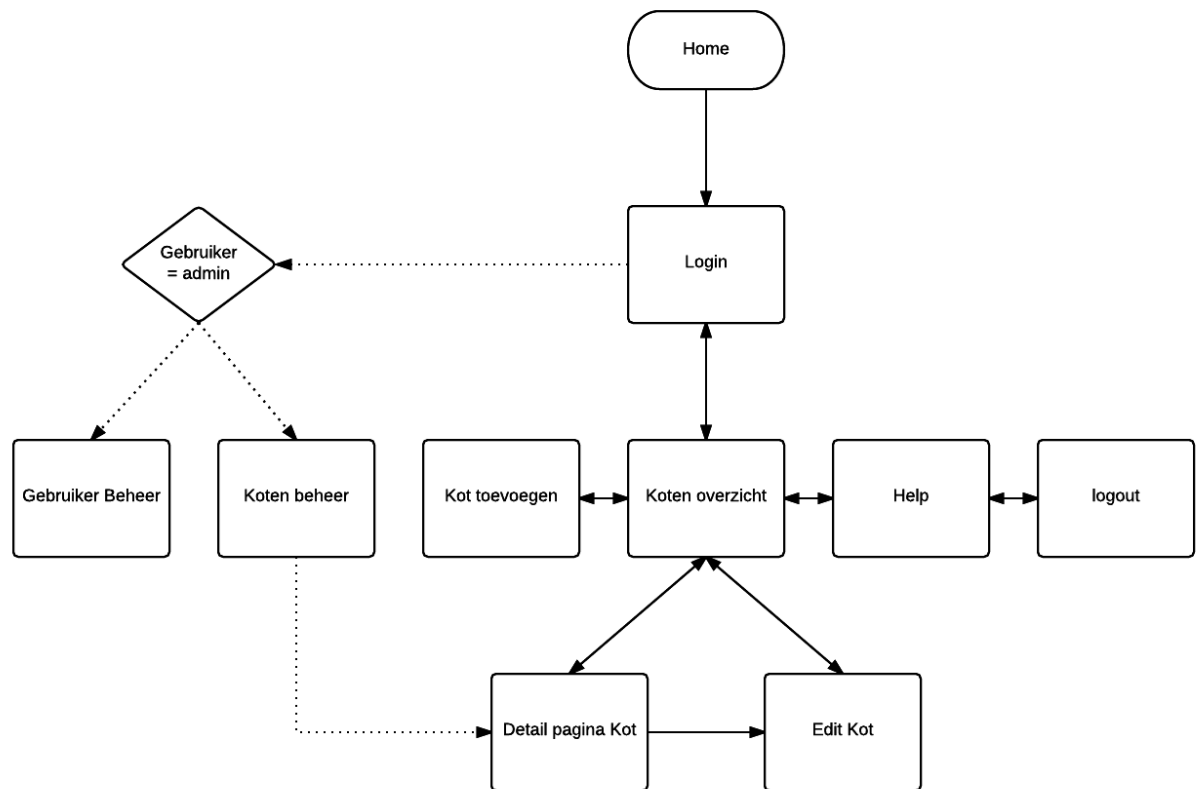


Loginscherm op zelfde pagina als landing-page  
Eenvoudig clean formulier

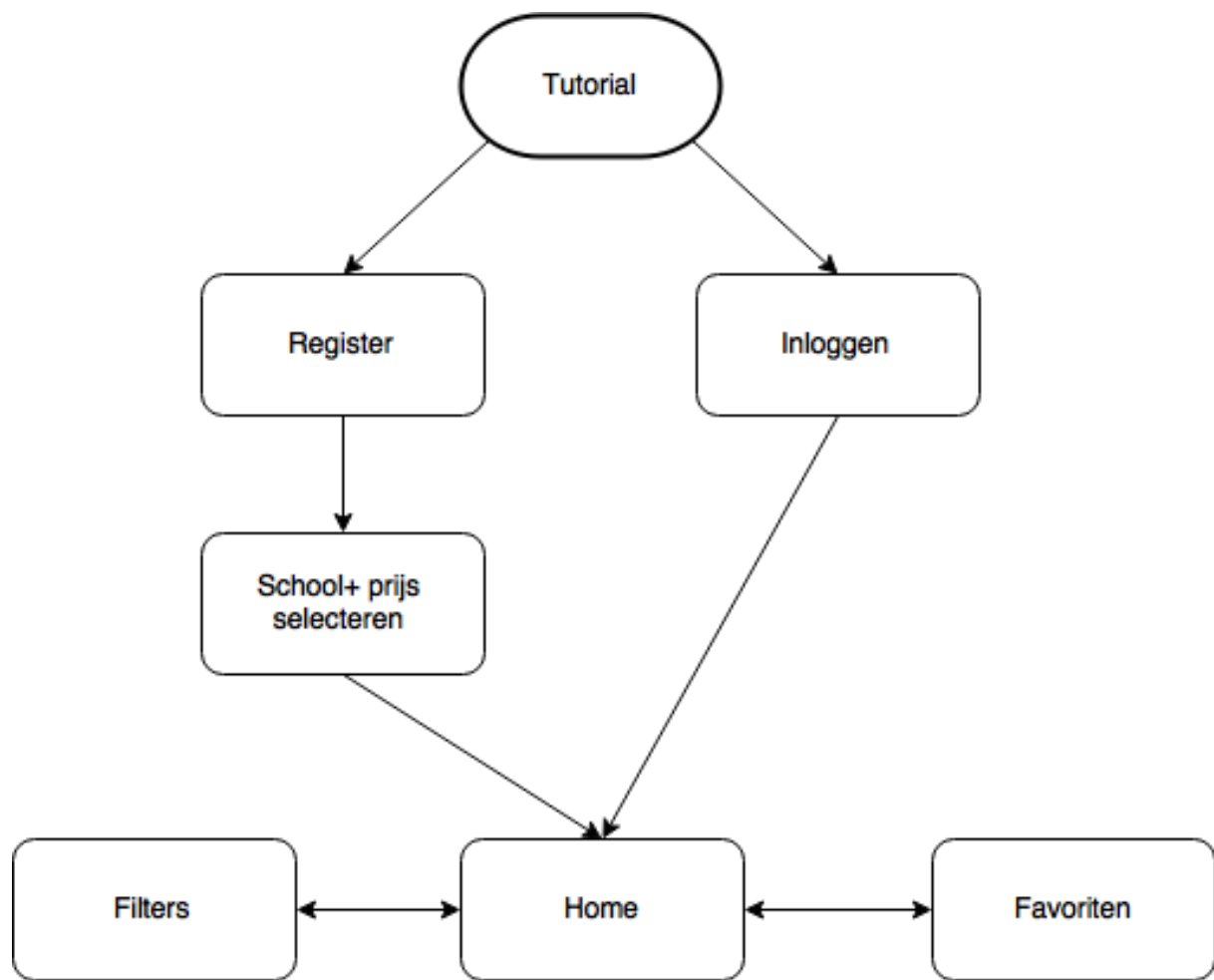


## 4.3 Sitemap

### Website

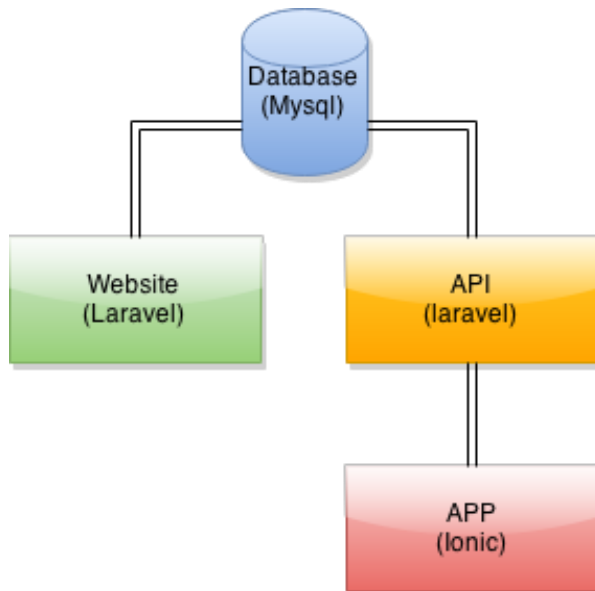


## App



## 4.4 Architectuur

Ons project bestaat uit twee delen: een website en een app. De website gemaakt in laravel. De applicatie is gemaakt in ionic. De app doet telkens ajax calls naar de website namelijk naar de apicontroller deze controller verzorgt alle request van de app.



### Laravel

Laravel is een backend MVC framework of Model-View-Controller framework. Laravel maakt het development process veel eenvoudiger en leuk door gebruik te maken van enkelen geweldige features zo als: Eloquent ORM, Flexible routing, Migrations, Modular packaging system,...

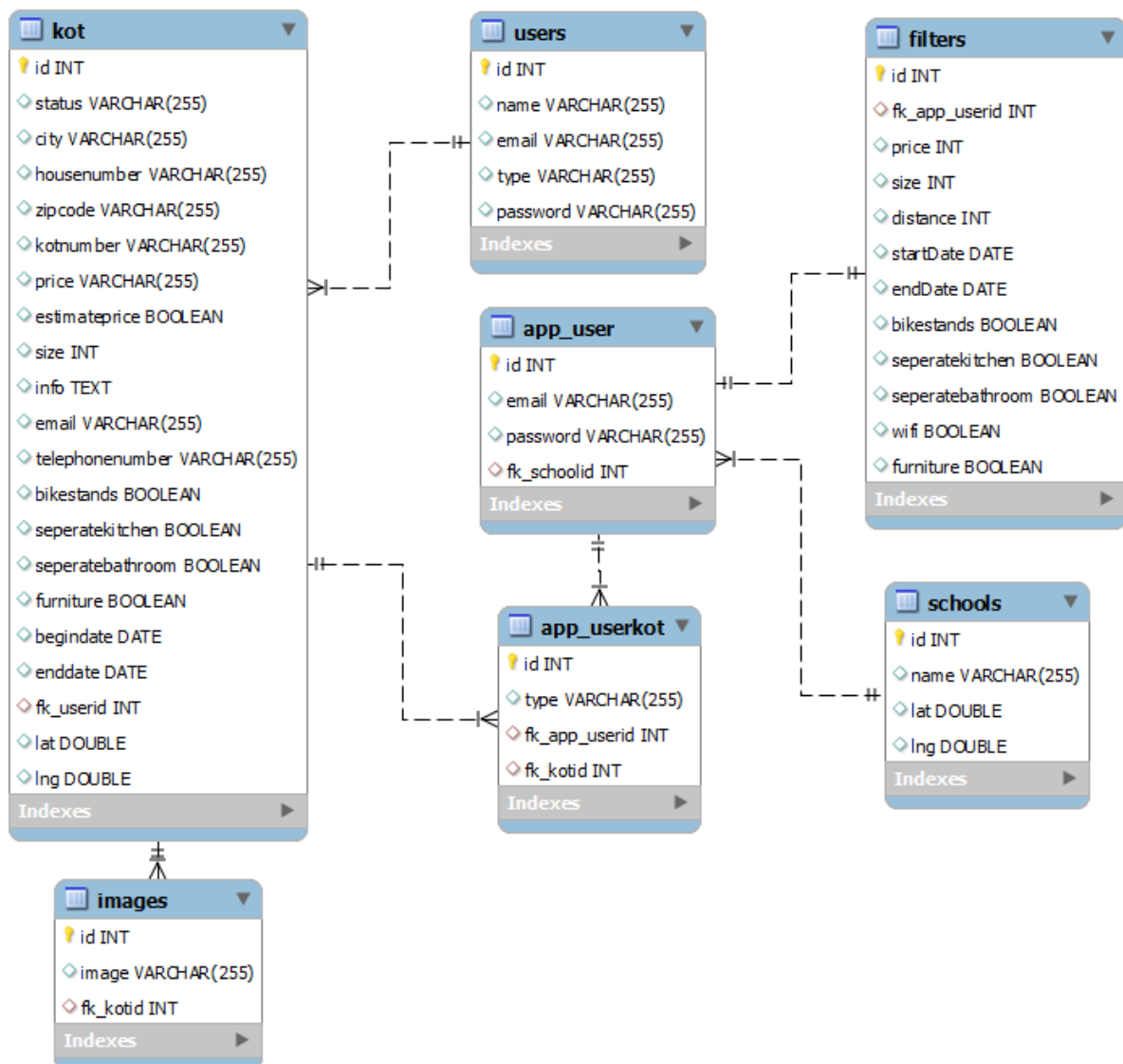
### Ionic

Ionic is Gratis en open source , Ionic biedt een bibliotheek van mobiele - geoptimaliseerde HTML , CSS en JS componenten , gestures , en tools voor het bouwen van zeer interactieve apps. Gebouwd met Sass en geoptimaliseerd voor angularjs .

Ionic maakt gebruik van angularjs , cordova en ionic framework zelf. Angularjs is een MVW framework of Model-View-Whatever framework. Deze wordt dus gebruikt om heel de logica te developen bij een ionic app. Dan hebben we nog cordova dit word gebruikt om de native functionaliteiten van het apparaat te gebruiken zo als de camera, accelerometer, geheugen ...

Het ionic framework zelf is zo als eerder gezegt ene bibliotheek van css, js enhtml componenten. Vergelijkbaar met bv: bootstrap of een foundation.

## 4.5 Database Schema



## 4.6 Code

### Kotten ophalen uit database

Deze code is van de app. Dit is een functie die opgeroepen wordt bij het openen van de view en telkens opnieuw wanneer er gestemd werd. Hier word er gebruik gemaakt van `KotService.getKot` deze zal een get request doen naar de server. Wanneer de data terug komt zal er een marker op de map worden gezet en wordt de data uitgelezen.

```
function getKot()
{
    // get a kot that you haven't voted on and match your current filters
    KotService.getKot(userdata).then(function(response){
        data = response.data;
        if(data['kot'] == null || data['kot'] == false)
        {
            $scope.loading = false;
            $scope.noResult = true;
        }
        else
        {
            $scope.include='templates/card.html';

            currentLocation['lat'] = data['kot']['lat'];
            currentLocation['lng'] = data['kot']['lng'];
            google.maps.event.addDomListener(document.getElementById("map"),
"load", setMap(currentLocation));
            $scope.kot = data['kot'];
            $scope.lenght = data['kot']['distance'];
            if(document.querySelectorAll('td-card')[0] !== undefined &&
document.querySelectorAll('td-card')[0] !== undefined){
                document.querySelectorAll('td-card')[0].removeAttribute("style");
            }
            $scope.loading=false;
        }
    })
};
```

Hier kan je de `KotService` zien. Bij elke functie word er een get request naar de server gestuurd met bepaalde paramaters.



```

.factory('KotService',function($http){
    return{
        getFavKoten: function(userdata){
            return $http({method: "get",dataType:
"jsonp",url:'http://kotterapp.be/api/favkotten',params : {userid:
userdata['id']},headers:{'Access-Control-Allow-Origin': '*'}})
        },
        getKot: function(userdata){
            return $http({method: "get",dataType:
"jsonp",url:'http://kotterapp.be/api/getKot',params: {userid:
userdata['id']},headers:{'Access-Control-Allow-Origin': '*'}})
        },
        reset:function(userdata){
            return $http({method:
"get",dataType:"jsonp",url:'http://kotterapp.be/api/resetkotten',params : {userid:
userdata['id']},headers:{'Access-Control-Allow-Origin': '*'}})
        }
    }
})

```

Bij de getKot request word deze functie aangeroepen in de apicontroller op de server. Hier wordt de functie getkot van het kot model opgeroepen. Na dat deze functie is uitgevoerd zal men het kot als data terug sturen en wordt deze uitgelezen in de app.

```

public function getKot(Request $request)
{
    $userid = $request->get('userid');
    $kot = Kot::getKot($userid,array());
    return response()->json(['kot'=>$kot]);
}

```

Hier kan je de functie zien die het ophalen van het kot behandelt. Er wordt dus een query opgemaakt aan de hand van de gebruiker zijn filters en gestemde koten om het correcte kot te vinden voor hem. Wanneer dit klaar is gaan we nog de afstand van het kot tot de gebruiker zijn campus berekenen aan de hand geotool. Wanneer deze kleiner is dan de gebruiker zijn maximum afstand wordt deze gereturned. Wanneer dit niet het geval is zal deze functie opnieuw worden uitgevoerd tot dat de afstand kleiner is dan de gebruiker zijn maximum afstand tot zijn campus.

```

public static function getKot($userid,$kotid)
{
    $filter = Filter::where('fk_app_userid',$userid)->first();
    $votedKotten = AppUserKot::where('fk_app_userid',$userid)->get();
    $votedKottenId = array();
    foreach($votedKotten as $kot)
    {
        $votedKottenId[] = $kot->fk_kotid;
    }
    $kot = Kot::with('images')->where('status','accepted')->whereNotIn('id',$votedKottenId)->whereNotIn('id',$kotid);
    if($filter->bikestands == true)
    {
        $kot = $kot->where('bikestands',true);
    }
    if($filter->seperatebathroom == true)
    {
        $kot = $kot->where('seperatebathroom',true);
    }
    if($filter->seperatekitchen == true)
    {
        $kot = $kot->where('seperatekitchen',true);
    }
    if($filter->wifi == true)
    {
        $kot = $kot->where('wifi',true);
    }
    if($filter->furniture == true)
    {
        $kot = $kot->where('furniture',true);
    }
    if($filter->price > 0)
    {
        $kot = $kot->where('price','<=',$filter->price);
    }
    if($filter->size > 0)
    {
        $kot = $kot->where('size','>=',$filter->size);
    }
    if($filter->startDate != 0)
    {
        $kot = $kot->where('begindate','<=',$filter->startDate);
    }
    if($filter->endDate != 0)
    {
        $kot = $kot->where('enddate','>=',$filter->endDate);
    }
    $kot = $kot->first();
    if(count($kot) <= 0)
    {
        return false;
    }
}

```

```

        $user = AppUser::with('school')->find($userid);
        $coordA = Geotools::coordinate([$kot->lat, $kot->lng]);
        $coordB = Geotools::coordinate([$user->school->lat,$user->school->lng]);
        $distance = Geotools::distance()->setFrom($coordA)->setTo($coordB);
        $kot->distance = number_format($distance->in('km')->haversine(), 2, '.',
    '');
    if($kot->distance > $filter->distance)
    {
        $kotid[] = $kot->id;
        Kot::getKot($userid ,$kotid);
    }
    else
    {
        return $kot;
    }
}

```

### Kotten opslaan in database

Wanneer beheerders van koten het formulier verzenden om een nieuw kot toe te voegen zal deze naar volgende functie worden gestuurd. Deze het kot zal opslaan en de afbeeldingen in de database.

```

public function store(addKotRequest $request,Kot $kot)
{
    $kot->fill($request->all());

    $param = array("address"=> $request->get('streetname').'
'. $request->get('houzenumber').' ' . $request->get('zipcode').' ' . $request-
>get('city'));
    $response = \Geocoder::geocode('json', $param);
    $response = json_decode($response);

    $kot->fk_userid = \Auth::user()->id;
    $kot->begindate = $request->begindate;
    $kot->enddate = $request->enddate;
    $kot->bikestands = ($request->get('bikestands') == 'on' ? true
:false );
    $kot->seperatekitchen = ($request->get('seperatekitchen') == 'on'
? true :false );
    $kot->seperatebathroom = ($request->get('seperatebathroom') ==
'on' ? true :false );
    $kot->furniture = ($request->get('furniture') == 'on' ? true
:false );
    $kot->wifi = ($request->get('wifi') == 'on' ? true :false );
    $kot->lat = $response->results[0]->geometry->location->lat;
    $kot->lng = $response->results[0]->geometry->location->lng;
    $kot->status = 'new';
    $kot->save();
    foreach( $request->file('images') as $file)
    {
        $manager = new ImageManager();
        $file = $manager->make($file)->widen(800);
        $filename = str_random(40).' .jpg';
        $destination = 'kot_img/';
        $file->save($destination.$filename);
        $image = new Image;
        $image->image = $destination.$filename;
        $image->fk_kotid = $kot->id;
        $image->save();
    }

    return Redirect('/kot');
}

```

Voordat deze code wordt uitgevoerd moet er eerst een validatie gebeuren. Deze gebeurt door addKotRequest. Hieronder ziet u hoe deze eruit ziet. Deze zal er voor zorgen dat er telkens 4

afbeeldingen worden geupload per kot en dat alle nodige velden correct worden ingevu

```
<?php namespace eindwerk\Http\Requests;

use eindwerk\Http\Requests\Request;

class addKotRequest extends Request {

    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        $rules = [
            'city' => 'required',
            'streetname' => 'required',
            'housenumber' => 'required',
            'zipcode' => 'required',
            'price' => 'required|integer',
            'size' => 'required|integer',
            'telephonenumber' => 'required',
            'begindate' => 'required',
            'enddate' => 'required|after:beginndate',
            'images.0' => 'required|image|mimes:jpeg,jpg,bmp,png,gif',
            'images.1' => 'required|image|mimes:jpeg,jpg,bmp,png,gif',
            'images.2' => 'required|image|mimes:jpeg,jpg,bmp,png,gif',
            'images.3' => 'required|image|mimes:jpeg,jpg,bmp,png,gif',
        ];
        return $rules;
    }
}
```

## Stemmen via swipen

Voor de swipe cards hebben we gebruikt gemaakt van een plugin ionic-contrib-tinder-cards die het gedeelte van het animeren van de swipe cards al voor je doet. Je moet alleen zelf nog bepaalde acties vast hangen aan het swipen zelf. Dit is dankzij ionic gestures vrij simpel. Je kan hiervoor custom attributes gebruiken zoals: on-swipe-left , on-swipe-right , on-release,...

```

<td-cards ng-show='!loading'>
  <td-card on-release="onRelease(kot.id)" on-swipe-left="swipeLeft(kot.id)"
on-swipe-right="swipeRight(kot.id)" class="card-{{kot.id}}" id="card-{{kot.id}}">
    <div class='info'>
      
      <div class="kot_thumbs">
        
      </div>
      <div class="kot_icons">
        <a class="item item-icon-left kot_details" href="#">
          <i class="icon ion-navigate"><p>{{lenght}} km van
{{user.school}}</p></i>
        </a>

        <a class="item item-icon-left kot_details" href="#">
          <i class="icon ion-android-pin"><p>{{kot.streatname}}
{{kot.housenumber}} {{kot.city}}</p></i>
        </a>

        <a class="item item-icon-left kot_details" href="#">
          <i class="icon ion-social-euro"><p>{{kot.price}},0</p></i>
        </a>

        <a class="item item-icon-left kot_details" href="#">
          <i class="icon ion-android-expand"><p>{{kot.size}}
&#13217;</p></i>
        </a>
      </div>
      <div class="no-text">NOPE</div>
      <div class="yes-text">LIKE</div>
    </div>
  </td-card>
</td-cards>

```

Deze zullen dan een functie oproepen. Buiten het links en rechts swipen is er ook nog een functie onRelease. Wanneer je niet snel genoeg swiped zal deze anders als een drag gezien worden. Wanneer de drag stopt zal de onRelease functie opgeroepen worden. Hier zal gecontroleerd worden of je naar links of naar rechts geswiped hebt. Aan de hand hiervan wordt er dan gestemd. De timeout zorgt er voor dat er zeker niet 2 keer gestemd wordt, dus voorkomt dat de onRelease functie samen met een

andere functie, zoals bv de swipeLeft functie, samen zullen worden opgeroepen.

```
var timeout;
// when swipe left dislike
$scope.swipeLeft = function(id)
{
    $timeout.cancel(timeout);// cancel the timeout of other functions like
onRelease
    timeout = $timeout(function() {
        $scope.vote('dislike',id);
    }, 800);
}
// when swipe right like
$scope.swipeRight = function(id)
{
    $timeout.cancel(timeout);
    timeout = $timeout(function() {
        $scope.vote('like',id);
    }, 800);
}

// when swipe is slow it wil not register as a swipe
// this function will check on drag release if the users has voted
$scope.onRelease = function(id)
{
    // get with of device
    var w = Math.max(document.documentElement.clientWidth, window.innerWidth ||
0);
    if(document.querySelectorAll('td-card')[0].getBoundingClientRect()['right'] >
(w*1.2133))
    {
        console.log('like');
        $timeout.cancel(timeout);
        timeout = $timeout(function() {
            $scope.vote('like',id);
        }, 800);
    }

    if(document.querySelectorAll('td-card')[0].getBoundingClientRect()['left'] <
(-w*0.2133))
    {
        console.log('dislike');
        $timeout.cancel(timeout);
        timeout = $timeout(function() {
            $scope.vote('dislike',id);
        }, 800);
    }
    $ionicScrollDelegate.scrollTop();
}
```

## 5. Conclusie

### Technologie

De app ontwikkelen was zeker een leuk proces. We hadden allebei geen ervaring met het ionic framework whatsoever, en dat maakte de uitdaging net zoveel interessanter.



Het ionic framework heeft nog veel te bieden dat we nog niet ontdekt hebben, maar door dit project hebben we al heel wat ervaring kunnen opdoen en hebben we ons werkterrein een pak kunnen uitbreiden. We weten nu alletwee dat dit hét framework is om cross-device apps mee te ontwikkelen door middel van technologieën waar we ons helemaal in thuis voelen.

### Time management

Doordat we de paasvakantie hadden overgeslagen tijdens de stage, hebben we twee extra weken tijd gehad om aan dit eindwerk te werken. Die twee weken zijn zeker ten goede gekomen, en dit heeft er ook voor gezorgd dat we alles op tijd hebben kunnen afkrijgen. Uiteraard zijn er een heleboel dingen die we onder –of overschat hadden, maar al bij al viel dit wel mee.

### Samenwerking

In het verleden hebben we al vaak samengewerkt, hierdoor zijn we zeer goed op elkaar afgestemd, en was de taakverdeling zeer eenvoudig gebeurd. Veel samenkomen en bespreken heeft ervoor gezorgd dat we allebei 100% achter alle beslissingen binnen onze app staan.

### Toekomst

Alhoewel de app ‘af’ zou kunnen beschouwd worden, stopt het hier echter niet voor ons. Er staan nog een heleboel dingen te wachten. In de nabije toekomst gaan we namelijk:

- De samenwerking met huidig bedrijf uitbreiden
- Contact zoeken met meerdere andere kotbedrijven
- Kotter uitvoerig beginnen te promoten via social media
- ...

## 6. Resources

<http://ionicframework.com/>

<http://laravel.com/docs/5>

<http://blog.ionic.io/tinder-for-x/>

<https://blog.nraboyn.com/2014/10/implement-google-maps-using-ionicframework/>

<https://blog.nraboyn.com/2014/06/check-network-connection-with-ionicframework/>

<http://ngcordova.com/docs/plugins/facebook/>

<http://ccoenraets.github.io/ionic-tutorial/create-ionic-application.html>

<https://daneden.github.io/animate.css/>

<https://github.com/toin0u/Geotools-laravel>

<http://image.intervention.io/>

<http://stackoverflow.com/>

<http://forum.ionicframework.com/>

<http://specials.han.nl/themasites/studiecentra/verwerken-en-delen/bronnen-vermelden/apa-normen/#comp00004b902de60000000b27453d>