Afstudeerverslag

Gepersonaliseerde mobile Android-app ontwikkelen

Haagse Hogeschool

Student:
Naam: Marc Vivaldi
Nummer: 07000898
Opleiding: Informatica

Bedrijf:
Liones
Polakweg 7
2288 GG Rijswijk
070-3191923

Bedrijfsmentoren:
dhr. M. Olivier

Opdrachtgever:
dhr. G. Wijma

Begeleidend examinator:
mevr. A.M.J.J. Lousberg

Tweede examinator:
dhr. B. Kuiper

Datum:
21-03-2014

Voorwoord

Dit verslag is geschreven naar aanleiding van mijn afstudeerstage bij Liones. Ik heb aan de opdracht gewerkt van 21-10-2013 tot en met 21-03-2014. In het verslag laat ik zien wat ik gedurende deze periode ontwikkeld heb, welke methodes/technieken ik gebruikt heb en wat mijn gedachtes en overwegingen waren voor bepaalde keuzes.

Ik wil Liones graag bedanken voor de stageplaats die zij mij hebben aangeboden. Verder wil ik Gjalt Wijma en Michel Olivier bedanken voor het begeleiden tijdens de afstudeerstage en het verschaffen van de informatie die benodigd was voor het ontwikkelen van het framework en de app.

Tijdens mijn afstudeerstage heb ik het bij Liones erg naar mijn zin gehad. Iedereen was ook bereid om te helpen en vragen te beantwoorden wanneer ik hiermee aankwam. Hiervoor wil ik alle collega's bedanken.

Bij het schrijven van het verslag heb ik van veel mensen hulp gekregen in de vorm van feedback over de structuur en inhoud van het verslag. Hiervoor wil ik mevr. Lousberg en dhr. Kuiper bedanken.

Marc Vivaldi
Rijswijk, 21-03-2014

Inhoudsopgave

Referaat

Dit verslag bevat het ontwerpen en ontwikkelen van een crossplatform framework voor het ontwikkelen van reader apps. Verder is de ontwikkeling van een app beschreven die met behulp van het framework gebouwd is. In de app worden persoonlijke suggesties gedaan op basis van het gedrag van de gebruiker.

Steekwoorden

- Framework voor reader apps
- Android Development
- Xamarin
- Crossplatform
- Personalisatie & Tracking

Inleiding

In dit verslag komen de werkzaamheden bij Liones naar voren die ik heb uitgevoerd tijdens mijn afstudeerstage van de opleiding Informatica op de Haagse Hogeschool.

In het verslag wordt beschreven aan wat voor opdracht er gewerkt is en hoe de ontstane problemen opgelost zijn.

In het hoofdstuk ‘1. Liones’ wordt het bedrijf beschreven waar ik stage gelopen heb. Hierna zal de opdracht en de doelstelling beschreven worden in het hoofdstuk ‘2. Afstudeeropdracht’.

Vervolgens zal het uitgevoerde project chronologisch beschreven worden. Per sprint zal er beschreven worden wat de werkzaamheden waren, welke problemen er ontstonden en hoe dit opgelost is.

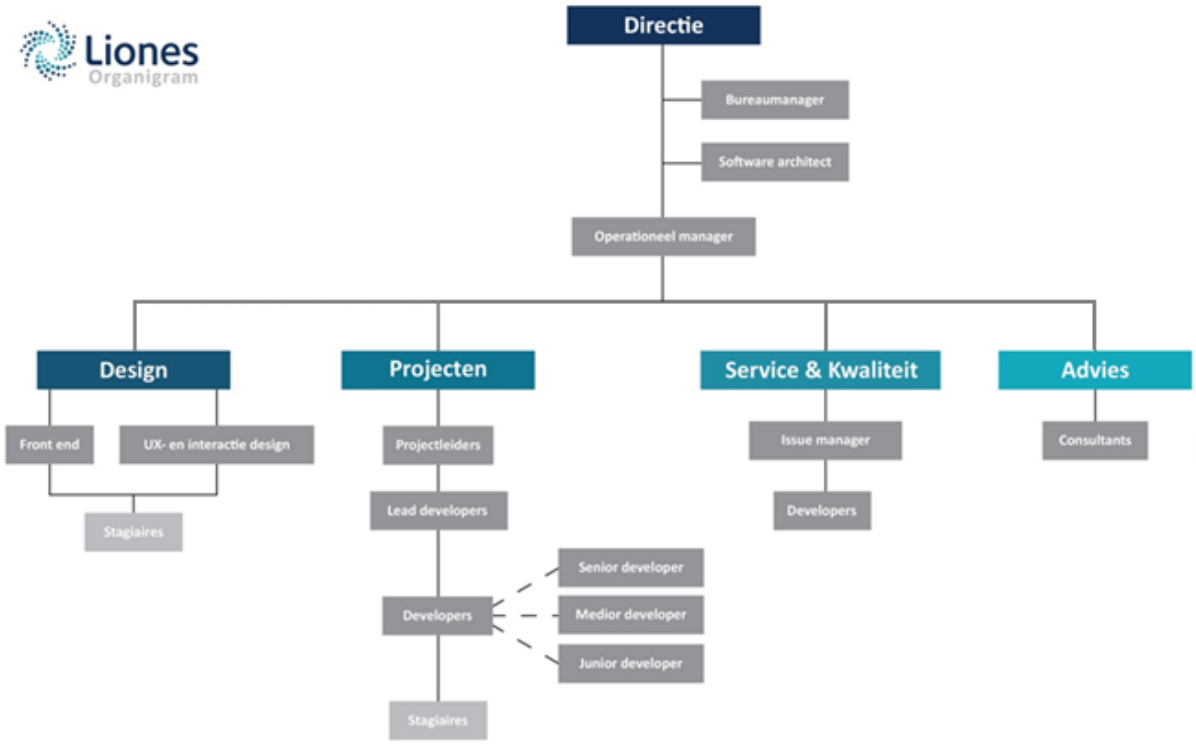
In het laatste hoofdstuk ’12. Reflectie’ wordt er teruggeblikt op het project. Hier worden als eerste de opgeleverde producten geëvalueerd. Ook worden het proces en de leereffecten geëvalueerd. Als laatste wordt er per opgegeven beroepstaak bekeken of het afgesproken niveau behaald is.

1. Liones

Liones is een fullservice internetbureau dat oplossingen bedenkt en realiseert om mensen blijvend te binden en te boeien. De roots van Liones liggen in de mediawereld. Het is in 1999 opgericht. In totaal werken er momenteel 25 personen, veel van deze werknemers zijn jong en hebben een HBO/WO opleiding. De werknemers zijn verdeeld over de volgende afdelingen:

- Development
- Grafisch Design
- Management
- Marketing
- Consulting
- Beheer

De werknemers van Liones zijn breed inzetbaar en goed op de hoogte van de nieuwste technologieën en tools. Liones bouwt complexe websites en applicaties en gaat graag uitdagingen aan.



Afbeelding 1: Organigram Liones

Mijn plaats in de organisatie als stagiair is in de Projecten afdeling. Bij Liones lopen er een aantal Research & Development projecten, waarvan mijn afstuderen er één is. Met Research & Development projecten wil Liones meer ervaring opdoen over bepaalde onderwerpen. Op deze manier wil Liones bekijken hoe, en of, ze met deze onderwerpen verder wil gaan. Ook worden er allerlei knelpunten ontdekt over deze onderwerpen, zonder de druk van een klant erachter.

2. Afstudeeropdracht

2.1. Probleemstelling

Liones heeft een klant die hun fysieke blad online wil hebben, namelijk Binnenlands Bestuur. Liones heeft hiervoor de site en het cms ontwikkeld. De artikelen zijn opgeslagen in een database, met hieraan metadata gekoppeld. Liones heeft het idee om een tablet/mobile Android app te ontwikkelen voor deze klant naast de site. Ze willen weten of een app bevalt voor de klant en of die dit product wil afnemen.

In de app moet een stuk personalisatie worden opgenomen, zodat de gebruiker zijn voorkeuren voor bepaalde artikelen kan opgeven.

Liones wil van de gebruikers weten wat zij interessant vinden en vaak bekijken, om zo hun voorkeuren naar boven te halen. Dit willen ze bereiken door middel van tracking. Dit houdt in dat er bijgehouden wordt wat de gebruiker in de app doet. Deze informatie kan dan weer gebruikt worden om relevante content te tonen in de app.

Het is voor Liones niet duidelijk wat de gewenste, meest soepele en voor de hand liggende manier van navigeren in een app is. Daarom zal er een stuk onderzoek gedaan worden naar dit onderwerp. Dit zal bestaan uit het vergelijken van meerdere apps en de verschillende manieren van navigeren onderwerpen aan een aantal testscenario's.

2.2. Doelstelling

Het ontwerpen en ontwikkelen van een tablet/mobile Android app voor Liones. De app moet personalisatie en tracking bevatten.

Met personalisatie kan de gebruiker aangeven wat zijn interesses en voorkeuren zijn. De gebruiker zal dit instellen bij het profiel dat aangemaakt wordt. Zo kan elke gebruiker zijn voorkeuren voor onderwerpen opgeven.

Door middel van tracking van de gebruiker moeten bepaalde voorkeuren zichtbaar worden. Er wordt gekeken naar wat voor soort artikelen een gebruiker vaak bekijkt. Op basis hiervan moeten er in de app suggesties gedaan worden voor profielverrijking en moet relevante content getoond worden.

Door de combinatie van het profiel en tracking wordt het mogelijk om content te ranken voor de gebruiker. Zo kan er voor elk artikel berekend worden of het relevant is voor de gebruiker.

Tevens moet er vloeiend door de app genavigeerd kunnen worden. De manier van navigeren zal bepaald worden door het uit te voeren onderzoek.

Het grafische aspect van de app heeft geen prioriteit.

2.3. Resultaat

Na het ontwerpen en ontwikkelen van de app wordt deze gepresenteerd aan de klant van Liones. De klant maakt een beslissing of ze verder wil gaan met dit concept. Indien dat het geval is levert Liones de app als volwaardig product. Dit betekent wel dat er grafisch nog veel moet gebeuren. Aangezien dit geen prioriteit is bij de afstudeeropdracht.

De opgeleverde app kan men zien als een gevorderd prototype, waarin het gehele idee duidelijk naar voren moet komen. De onderwerpen als personalisatie, tracking en navigatie moeten in de app zijn opgenomen.

2.4. Uit te voeren werkzaamheden, inclusief een globale fasering, mijlpalen en bijbehorende activiteiten

- Oriëntatie op de huidige situatie (4 dagen)
- Plan van Aanpak opstellen (3 dagen)
- Scrum opzetten (3 dagen)
- Onderzoek naar voorkeuren t.o.v. UI navigeren (5 dagen)
- Requirements achterhalen bij Liones (opdrachtgever) en product backlog opstellen (4 dagen)
- Architectuur ontwerpen voor personalisatie en tracking (5 dagen)
- Schermontwerpen maken (5 dagen)
- Ontwerpen, bouwen, testen volgens Scrum (38 dagen)
    - Basisapp en framework(14 dagen)
    - Navigeren (5 dagen)
    - Tracking (10 dagen)
    - Personalisatie (9 dagen)
- Opleveren (3 dagen)

2.5. Op te leveren (tussen)producten

- Onderzoek naar voorkeuren gebruiker t.o.v. UI navigeren
- Schermontwerpen
- Ontwerpdocument
    - Klassendiagram
    - Sequentie diagram
    - Databaseontwerp (EER diagram)
- Testresultaten
    - Systeemtest
    - Acceptatietest
- App

2.6. Aanpassingen in de afstudeeropdracht

Toen ik met mijn afstudeerstage begon bij Liones heb ik als eerst een aantal gesprekken gehad met mijn opdrachtgever en bedrijfsmentor over de afstudeeropdracht. Uit deze gesprekken kwamen een aantal aanvullende wensen en eisen naar voren, maar ook een aantal wijzigingen in de huidige opdracht. Dit heb ik in mijn planning en ontwerpen opgenomen. Dit is verderop beschreven in dit hoofdstuk.

2.6.1. Aanpassing afstudeeropdracht

De opdracht was aanvankelijk om een gepersonaliseerde app te ontwerpen en ontwikkelen voor Binnenlands Bestuur, waarin ook tracking van de gebruiker opgenomen was. Dit stond beschreven in het originele afstudeerplan.

De klant waarvoor de app ontwikkeld zal worden is veranderd en naast de originele opdracht moet er een framework voor news/content reader apps opgezet worden.

De opdracht is nu het ontwerpen en ontwikkelen van een gepersonaliseerde app voor Locaties.nl geworden. Naast het ontwikkelen van deze app is de opdracht ook om een framework voor news/content reader apps op te zetten. Dit is een uitbreiding op mijn oorspronkelijke opdracht. Van dit framework zal ik de algemene shared core en de Android shared core opzetten (uitleg hoofdstuk sprint 1). De scope van het Android gedeelte blijft daarmee ongewijzigd. Door de toevoeging van het ontwikkelen van het framework is de beroepstaak 3.1 Ontwerpen Softwarearchitectuur opgenomen bij de beroepstaken.

De klant waarvoor de app aanvankelijk ontwikkeld zou worden was Binnenlands Bestuur, nu is dat veranderd naar de klant Locaties.nl. De reden hiervoor was dat Binnenlands Bestuur verkocht is en de verwachting bij Liones was dat ze deze klant kwijt zouden gaan raken.

Binnenlands Bestuur is een nieuwsplatform voor ambtenaren, vergelijkbaar met nu.nl. Locaties.nl is een website voor particulieren en bedrijven om een locatie te plannen voor bijv. een bijeenkomst of bruiloft. Bij Locaties.nl zijn het locaties die opgeslagen zijn in de database. Beiden hebben een database die rijk is aan metadata. Dit was de eis voor de uit te werken case omdat het personalisatie onderdeel hier sterk afhankelijk van is.

Een uitbreiding op de opdracht is dat er ook met Google Maps gewerkt zal worden. Dit was niet opgenomen in de case voor Binnenlands Bestuur.

Een wijziging in de opdracht is het werken met C# in Xamarin. Dit was eerst Java in Eclipse.

2.6.2. Aanpassing op te leveren producten

Doordat de opdracht aangepast werd zijn uiteindelijk de op te leveren producten ook aangepast. De uiteindelijke lijst van op te leveren producten is hieronder getoond.

- **Plan van Aanpak:**
  Algemeen document waarin de opdrachtomschrijving, probleemomschrijving, doelstelling van de opdracht, afbakening van de opdracht, risicofactoren, projectorganisatie, planning en de beschrijving van de mijlpaalproducten in opgenomen zijn.
- **Onderzoek naar voorkeuren t.o.v. UI navigeren:**
  Document waarin het onderzoek naar voorkeuren ten opzichte van navigeren door een app beschreven is.
- **Product backlog:**
  In de product backlog worden alle requirements van de app bijgehouden. Dit bestaat uit alle onderdelen die uitgewerkt moeten worden om een werkende app te kunnen opleveren.
- **Sprint backlogs:**
  Een sprint backlog is een lijst van het werk dat tijdens de betreffende sprint uitgevoerd zal worden. De lijst is opgebouwd uit items die geselecteerd zijn van de product backlog.
- **Schermontwerpen:**
  De schermontwerpen van verschillende belangrijke activities.
- **Ontwerpdocument:**
  Ontwerpdocument met hierin opgenomen de volgende onderdelen:
  - Klassendiagram
  - Sequentie diagram
  - Databaseontwerp (EER diagram)
  - Ontwerp personalisatie/tracking
- **Testresultaten**
  - Systeemtest
  - Acceptatietest

- **Framework:**

Het framework heeft als doel het gemakkelijk op kunnen zetten van reader apps. Opgenomen zijn onderdelen als GridView, ListView en ScreenSliderPager. Deze hebben een volledige implementatie voor de logica die bij deze Views hoort. Geen klant specifieke implementatie.

- **App:**

De app is een gevorderd prototype gebruikmakend van het ontwikkelde framework, waarin het gehele idee duidelijk naar voren moet komen. De onderwerpen als personalisatie, tracking en navigatie zijn in de app opgenomen.

3. Tot stand komen van PvA

Dit hoofdstuk beschrijft het tot stand komen van het Plan van Aanpak. Hierin worden de gebruikte methoden, technieken en planning opgenomen.

3.1. Methode

Bij Liones werken alle projectteams en alle Research & Development opdrachten volgens Scrum. Het werken met Scrum in Research & Development opdrachten is handig omdat er veel factoren zijn die kunnen veranderen tijdens het project. Een agile methode speelt hier goed op in, het is namelijk een flexibele manier van werken. Veranderingen kunnen snel en efficiënt afgevangen worden.

3.2. Scrum

Voor dit project is er gebruik gemaakt van Trello.com. Dit product kan zorgen voor een digitaal scrumboard, waar ook de backlog op bijgehouden kan worden. Dit is een tool waarin met Scrum gewerkt kan worden, het is de sprint backlog. Zo kun je voor elke sprint een board aanmaken en hierin user stories plaatsen. Per sprint zijn er verschillende categorieën waarin de user stories geplaatst kunnen worden. Bijv. To Do, Doing en Done. Er kan met meerdere mensen per board gewerkt worden en user stories kunnen opgepakt worden door gebruikers. Deze zijn dan assigned aan deze gebruiker.

In dit project zal ik samenwerken met een aantal mensen uit andere disciplines binnen Liones, daarom is Trello van toegevoegde waarde.

3.3. Versiebeheer

Versiebeheer voor de source code heeft plaatsgevonden met behulp van SVN. Per sprint is er een aparte branche aangemaakt.

Versiebeheer voor documenten heeft plaatsgevonden met behulp van Dropbox.

3.4. Globale Planning

Bij het opstellen van het afstudeerplan is er een globale planning gemaakt.

- Oriëntatie op de huidige situatie (4 dagen)
- Plan van Aanpak opstellen (3 dagen)
- Scrum opzetten (3 dagen)
- Onderzoek naar voorkeuren t.o.v. UI navigeren (5 dagen)
- Requirements achterhalen bij Liones (opdrachtgever) en product backlog opstellen (4 dagen)
- Architectuur ontwerpen voor personalisatie en tracking (5 dagen)
- Schermontwerpen maken (5 dagen)
- Ontwerpen, bouwen, testen van framework en implementatie van app volgens Scrum (38 dagen)
    - Basisapp en framework(14 dagen)
    - Navigeren (5 dagen)
    - Tracking (10 dagen)
    - Personalisatie (9 dagen)
- Opleveren (3 dagen)

4. High-level vraagstukken

Tijdens de eerste gesprekken met de opdrachtgever en bedrijfsmentor zijn er een aantal High-level vraagstukken naar voren gekomen die voor Liones belangrijk zijn bij het ontwikkelen van het framework en reader apps. De High-level vraagstukken zijn verdeeld over meerdere disciplines binnen Liones.

Liones hoopt antwoord te krijgen op deze vraagstukken door middel van het uitvoeren van Research & Development opdrachten. Mijn afstudeeropdracht is er daar één van.

4.1. Organisatie

Het ontwerpen en ontwikkelen van apps is nieuw voor Liones. Ze willen hier graag mee beginnen, daarom maken ze hier een opzet voor met stagiaires.

- Mobile app met samenwerking tussen sales, User Experience(UX), development.
- Samenwerking testen c.q. ontwikkelen

4.2. Research & Development

Projecten met stagiaires bij Liones zijn vaak Research & Development projecten. Hiermee wil Liones meer inzicht krijgen in die onderwerpen zodat ze hiervoor een basis kunnen opzetten. Deze basis kan dan gebruikt worden voor productie projecten.

- Op welk moment in de Research & Development projecten betrekken we de klant erbij?
- Hoe leggen we een profiel van de gebruiker vast in een mobile app?
- Hoe moeten de serverside services eruitzien?

4.3. Techniek

- Hoe gaan we om met crossplatform?
- Hoe gaan we om met hergebruik?
- Hoe houd je ontwikkeling eenvoudig?
- Welke libraries kunnen we gebruiken?
- Is Xamarin geschikt voor ons?

4.4. Functionele Requirements voor reader apps

- Lijst van content items weergeven
- Support voor categorieën
- Mogelijkheid tot inloggen
- Detail pagina van een item tonen
- Metadata tags tonen van een item

Gemetadateerde tekstuele content evt. Afbeeldingen
Voorbeeld vacatures, locaties, nieuws.

5. Oriëntatie

In de oriëntatie fase van het project zijn er meerdere onderwerpen behandeld.

5.1. Onderzoek naar voorkeuren t.o.v. UI navigeren

In de oriëntatie fase is er gestart met het uitvoeren van een onderzoek naar UI navigeren in bestaande reader apps.

De twee vragen die met dit onderzoek beantwoord moesten worden zijn:

- Welke navigatiestructuren worden er in bestaande reader apps gebruikt?
- Op welke manier tonen de bestaande reader apps hun content?

Op basis van de bevindingen uit het onderzoek zal er antwoord gegeven worden op deze vragen.

De volgende apps zijn hiervoor gebruikt:

- Flipboard
- Zite
- The Guardian
- Material

De keuze was gevallen op deze vier apps omdat deze behoren tot de meest toonaangevende en professionele apps die er zijn op dit moment op het gebied van reader apps.

Deze apps tonen nieuws aan de gebruiker in verschillende categorieën. Vaak zijn deze categorieën door de gebruiker zelf in te stellen. De apps werden geanalyseerd op navigatiestructuren en de manier van het tonen van de content.

Om alle gebruikte navigatiestructuren in kaart te brengen zijn er screenshots gemaakt in elke app bij belangrijke onderdelen. Verder is er gekeken welke content er getoond wordt op welke niveaus en welke verschillen dit met zich meebracht.

De apps zijn geanalyseerd en gedocumenteerd. Dit werd gedaan voor 3 verschillende niveaus. Dit waren categorie-, artikelenoverzicht- en artikelniveau. Hieronder zijn de resultaten getoond van de analyse per niveau.

Het volledige document is opgenomen als bijlage onder 15.4 Onderzoek naar voorkeuren t.o.v. UI navigeren.

5.1.1. Categorieniveau

De navigatie op categorieniveau werkt erg natuurlijk bij Flipboard. Je krijgt een goed overzicht van alle categorieën en je navigeert er gemakkelijk doorheen.

Het selecteren van een categorie is het simpelst bij Flipboard en heeft de meest logische navigatie.

5.1.2. Artikelenoverzichtniveau

Zite heeft de artikelen overzichtelijk onder elkaar geplaatst. Een nadeel van Zite is echter het scrollen door het artikelenoverzicht en dat de previews redelijk groot zijn.

Flipboard en Material werken als het ware met het omslaan van een pagina, waarbij Flipboard één artikel per pagina toont en Material meerdere.

Een nadeel van Material is dat je geen goed beeld krijgt waar de artikelen over gaan, vaak te weinig tekst doordat er meerdere artikelen op 1 pagina staan.

Bij Flipboard krijg je een goed idee waar het artikel over gaat als je door het artikelenoverzicht heen bladert. Er wordt een titel, foto en het begin van het artikel getoond.

Bij The Guardian krijg je een goed overzicht van ongeveer 5 artikelen tegelijk, allen hebben een foto en een koptekst.

	Zite	Flipboard	Material	The Guardian
Overzichtelijkheid artikelen	+	+	-	++
Goede impressie van inhoud artikel	+	++	--	+

Tabel 1: Artikelenoverzichtniveau bevindingen

5.1.3. Artikelniveau

Alle vier de reader apps tonen in een artikel grofweg dezelfde informatie. Het gaat hierbij om:

- Titel
- Tekst
- Auteur
- Bron
- Foto

Flipboard gebruikt het concept van pagina's omslaan voor het navigeren door een artikel. Dit werkt prettig. Het artikelenoverzicht is niet overzichtelijk, er is maar 1 preview per keer zichtbaar.

Zite gebruikt verticaal scrollen voor het navigeren door een artikel.

Material gebruikt verticaal scrollen voor het navigeren door een artikel. Omlaag doorscrollen aan het eind van een artikel zorgt ervoor dat het volgende artikel geopend wordt. Omhoog doorscrollen aan het begin van een artikel zorgt ervoor dat het vorige artikel geopend wordt.

The Guardian gebruikt verticaal scrollen voor het navigeren door een artikel. Omlaag doorscrollen aan het eind van een artikel zorgt ervoor dat het volgende artikel geopend wordt. Omhoog doorscrollen aan het begin van een artikel zorgt ervoor dat het vorige artikel geopend wordt.

	Zite	Flipboard	Material	The Guardian
Navigatie door artikel	+	++	+	+
Presentatie artikel	++	++	-	++
Navigeren naar vorig/volgend artikel	Niet aanwezig	Niet aanwezig	Aanwezig	Aanwezig

Tabel 2: Artikelniveau bevindingen

5.1.4. Conclusie

Op basis van de bevindingen uit het analyse document zijn de volgende beslissingen gemaakt.

We zullen navigeren door de app door gebruik te maken van de navigation drawer in Android, dit is het menu wat vanuit de zijkant opengetrokken kan worden. Dit zal de navigatie tussen de verschillende zoekschermen verzorgen.

Vanuit de verschillende zoekschermen kan een lijst getoond worden in een nieuwe activity die voldoet aan de zoekopdracht.

Als er geklikt wordt op een locatie uit de lijst, dan zal deze locatie in detail getoond worden in een nieuwe activity.

Het terug navigeren zal gebeuren met de back-button van het toestel. Tevens kan er terug genavigeerd worden met de Home-button indien de gebruiker in een activity zit die geen andere activies kan openen. Dit is volgens de Android guidelines opgezet.

Al deze informatie heeft gediend als input voor de navigatiestructuur, het schermontwerp en de flow van de Locaties.nl app. De uiteindelijke flow van de app is opgenomen in een sitemap, welke als bijlage toegevoegd is.

5.2. Locaties.nl

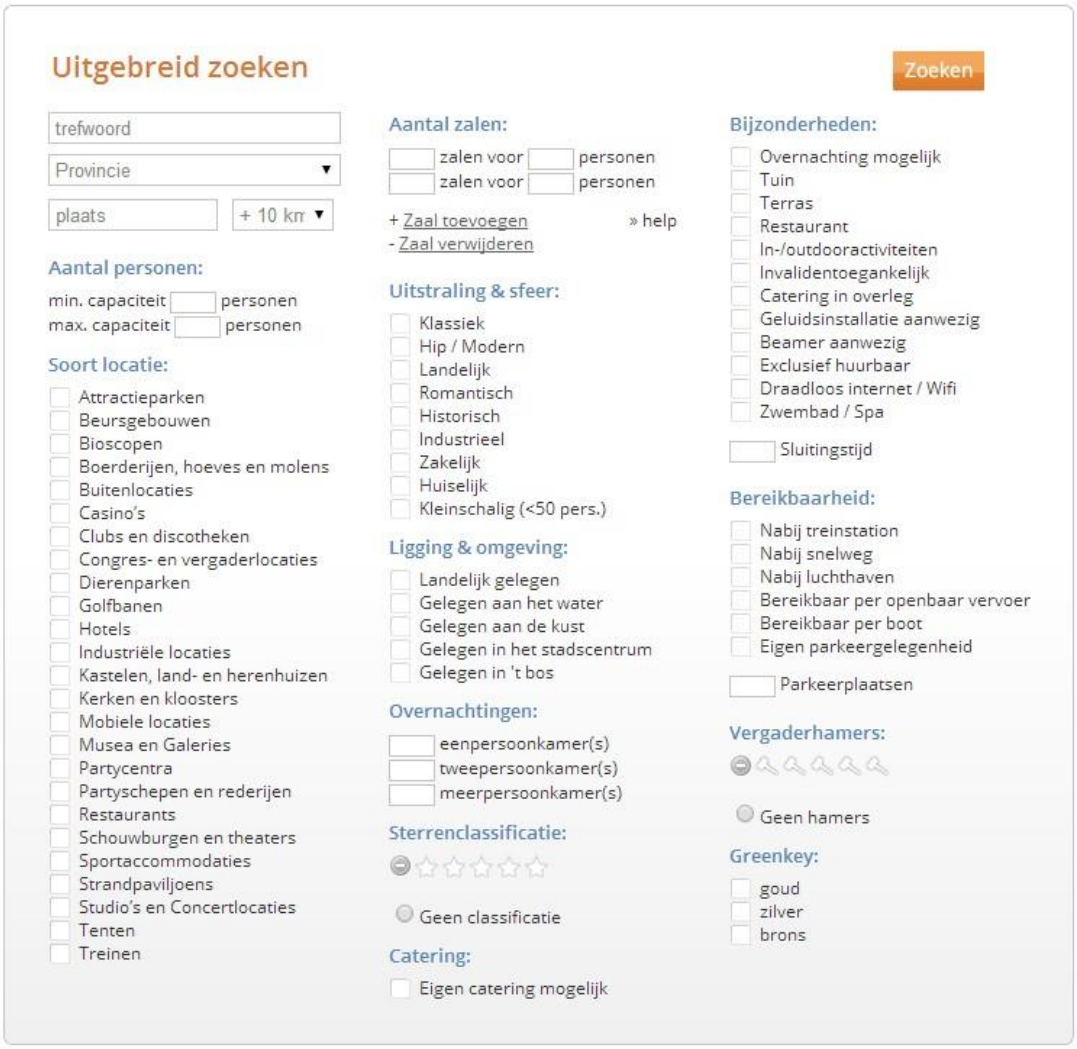
Locaties.nl is een site waar vraag en aanbod bij elkaar komen, met als onderwerp een locatie voor een feest, evenement, meeting of vergadering.

De eigenaar van een locatie kan een locatie gratis of tegen betaling aanmelden op de site, waarna deze geplaatst wordt. Een betaalde vermelding is een stuk uitgebreider dan een gratis vermelding. De eigenaar vult dan alle data en metadata van een locatie in zoals de naam, het soort locatie, NAW-gegevens, bereikbaarheid, bijzonderheden en een beschrijving van de locatie. Ook kunnen er bij betaalde vermeldingen foto’s worden toegevoegd.

Bij Liones wordt metadata van een locatie gezien als de gegevens waarop gefilterd en gezocht kan worden. De andere gegevens wordt gezien als de data van een locatie.

De gebruikers van Locaties.nl kunnen een locatie zoeken door een plaatsnaam of trefwoord in te vullen. Verder kunnen ze ook uitgebreid zoeken. Hierbij kan de gebruiker zijn zoekopdracht nog verder specificeren. Er wordt dan gezocht op de metadata die aan de locaties gekoppeld zit. Denk hierbij aan data zoals de bereikbaarheid en ligging van een locatie.

Als de gebruiker een locatie heeft gevonden dan kan hij deze bekijken. Alle metadata van een locatie wordt getoond op de detailpagina. Hiernaast kan de gebruiker een sfeerimpressie krijgen door de foto’s te bekijken van een locatie.



Afbeelding 2: Uitgebreid zoeken Locaties.nl

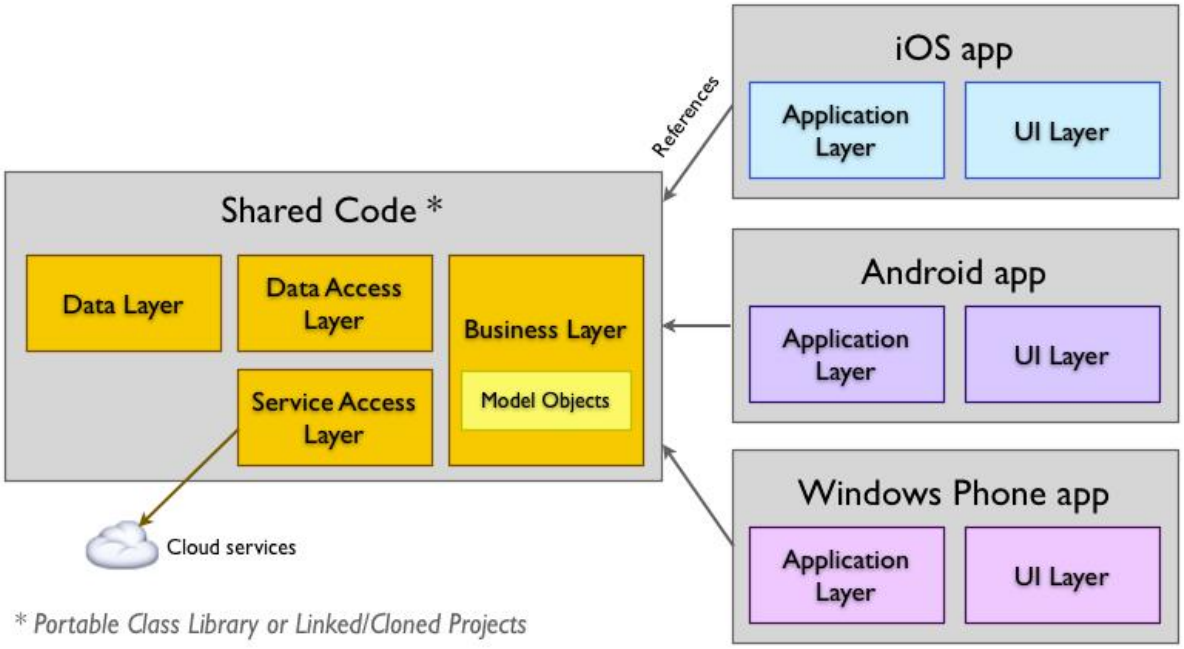
Om een goed totaalbeeld van de site te krijgen heb ik samen met mijn opdrachtgever de backend van Locaties.nl en de webservices hiervan bekeken. Op deze manier heb ik een goed overzicht gekregen van de beschikbare functionaliteiten van de webservices en de gekoppelde metadata bij locaties.

5.3. Xamarin/C#

Toen ik begon aan mijn afstuderen bij Liones zou ik gaan ontwikkelen in Java in Eclipse. Dit is echter veranderd naar C# in Xamarin na een eerste bespreking. De reden hiervoor was dat Liones graag een cross-platform solution wilde waar een groot deel van de code hergebruikt kan worden. Liones heeft niet de bezetting om later voor elk OS een apart team op te zetten, hiervoor willen ze één team opzetten.

Xamarin is een cross-platform solution dat gebruik maakt van een shared C# codebase. Hiernaast wordt er per platform(Android, iOS en Windows Phone) een app gebouwd die gebruik maakt van deze shared codebase. Deze apps worden ook ontwikkeld in C#.

In de app worden platform specifieke onderdelen afgehandeld zoals een lijst view, Google Maps en andere onderdelen. Verder is de UI layer platform specifiek. Doordat deze onderdelen platform specifiek zijn krijg je alsnog een native experience voor elk platform. Alle API's die op iOS en Android beschikbaar zijn, zijn dat ook in Xamarin.



Afbeelding 3: Architectuur Xamarin

Omdat Xamarin en C# nieuw voor mij waren ben ik begonnen met het doornemen van tutorials van Xamarin. Hierin werd de basis van het ontwikkelen in Xamarin in C# uitgelegd. Na het doornemen van de tutorials heb ik als oriëntatie in het platform een aantal kleine apps ontwikkeld. Deze apps focusten zich op elementen die de latere Locaties.nl app ook zou gaan bevatten.
Dit zijn o.a.:

- Lijst met items
- Swipe functionaliteit naar ander scherm
- Images in een grid
- Detail pagina
- Detail pagina met metadata
- Images in het groot te bekijken
- Asynchroon uitvoeren van een taak

Als laatste heb ik een basic RSS reader gemaakt met feeds van NU.nl. Hierna had ik voldoende kennis van het platform en C# om te beginnen aan de echte app en het framework. Het inwerken in heeft ongeveer 10 dagen geduurd.

5.4. Epics & User Stories

Nadat de wijzigingen in de opdracht doorgevoerd waren ben ik begonnen met het interviewen van de opdrachtgever en analyse van Locaties.nl om de user stories te achterhalen. In deze interviews zijn twee soorten user stories naar voren gekomen. Dit waren user stories over het framework en user stories over de app voor Locaties.nl.

De user stories zijn alle functionaliteiten die de app en het framework moeten bevatten. Deze user stories zijn in de bijlagen opgenomen. In het begin van de afstudeeropdracht waren een aantal user stories nog gedefinieerd als epics. Dit zijn user stories die nog opgesplitst kunnen worden in meerdere user stories. Dit kon echter niet altijd gedaan worden omdat de benodigde inzichten nog niet verkregen waren om het te kunnen opsplitsen. Deze werden dan opgesplitst aan het begin van de sprint waarin deze verwerkt waren.

Op basis van de interviews, de analyse van reader apps en de analyse van Locaties.nl is er bepaald welke functionaliteiten er benodigd waren in het te ontwikkelen framework. Dit zijn de basis elementen van reader apps. De lijst met epics is als volgt:

- Lijst van items
- Mogelijkheid om te swipen naar ander scherm
- Images in een grid
- Detail pagina
- Detail pagina met metadata
- Mogelijkheid om images in het groot te bekijken

Op basis van de interviews en de analyse van Locaties.nl is er bepaald welke functionaliteiten er benodigd waren in de te ontwikkelen app. Deze zijn de basis functionaliteiten voor de app. Niet elk onderdeel van de Locaties.nl site zal geïmplementeerd worden in de app. De lijst met epics is als volgt:

- Zoek locatie in de buurt
- Zoek op locatie soort
- Zoek op plaats
- Zoek op locatie naam
- Top 10 locaties
- Uitgebreid zoeken
- Favoriete locaties
- Locaties lijst bekijken
- Locatie in detail bekijken
- Locatie advies aanvragen
- Inloggen/uitloggen
- Registreren
- Tracking
- Locatie suggestie gebruiker

Deze epics zijn uitgewerkt in user stories die opgenomen zijn in de product backlog. De user stories werden geplaatst in verschillende sprints.

Nadat de user stories bekend waren zijn deze geprioriteerd. Dit is in overleg met mijn bedrijfsmentor gedaan. De prioritering geschiedde op de volgende manier: Op basis van complexiteit, prioriteit en tijdsduur werd een user story geplaatst op de sprint backlog. De combinatie van deze factoren bepaalde hoe hoog een user story zou komen te staan. User stories die hoog op de lijst stonden worden als eerst afgehandeld.

5.5. Sprints

In overleg met mijn bedrijfsmentor zijn er een aantal sprints opgesteld voor dit project. Dit is gedaan op basis van de opgestelde epics en user stories. Deze user stories zijn vervolgens in Trello geplaatst in de bijbehorende sprints.

De sprints zijn als volgt:

- Framework basis
- App flow locaties.nl
- Zoek & detailschermen Locaties.nl
- Overige schermen Locaties.nl
- Personalisatie
- Afronding

Elke sprint duurt 2 weken.

De user stories die de sprints bevatten zijn te zien in de sprint backlogs, die opgenomen zijn als bijlagen.

Er is gekozen om te beginnen met de sprint waarin het framework gebouwd wordt. Dit heeft als reden dat het gebouwde framework gebruikt kan worden tijdens het ontwikkelen van de app in de volgende sprints.

Om een duidelijk beeld te krijgen van de navigatie door de app is ervoor gekozen om dit in de tweede sprint op te pakken. Deze flow wordt in de hierna volgende sprints uitgebreid met de echte functionaliteiten die de app moet gaan bevatten.

De derde sprint zal de belangrijkste functionaliteiten van de app bewerkstelligen. Dit heeft als input nodig het gebouwde framework en de flow door de app. Hiermee zullen de zoek & detailschermen van Locaties.nl gebouwd worden.

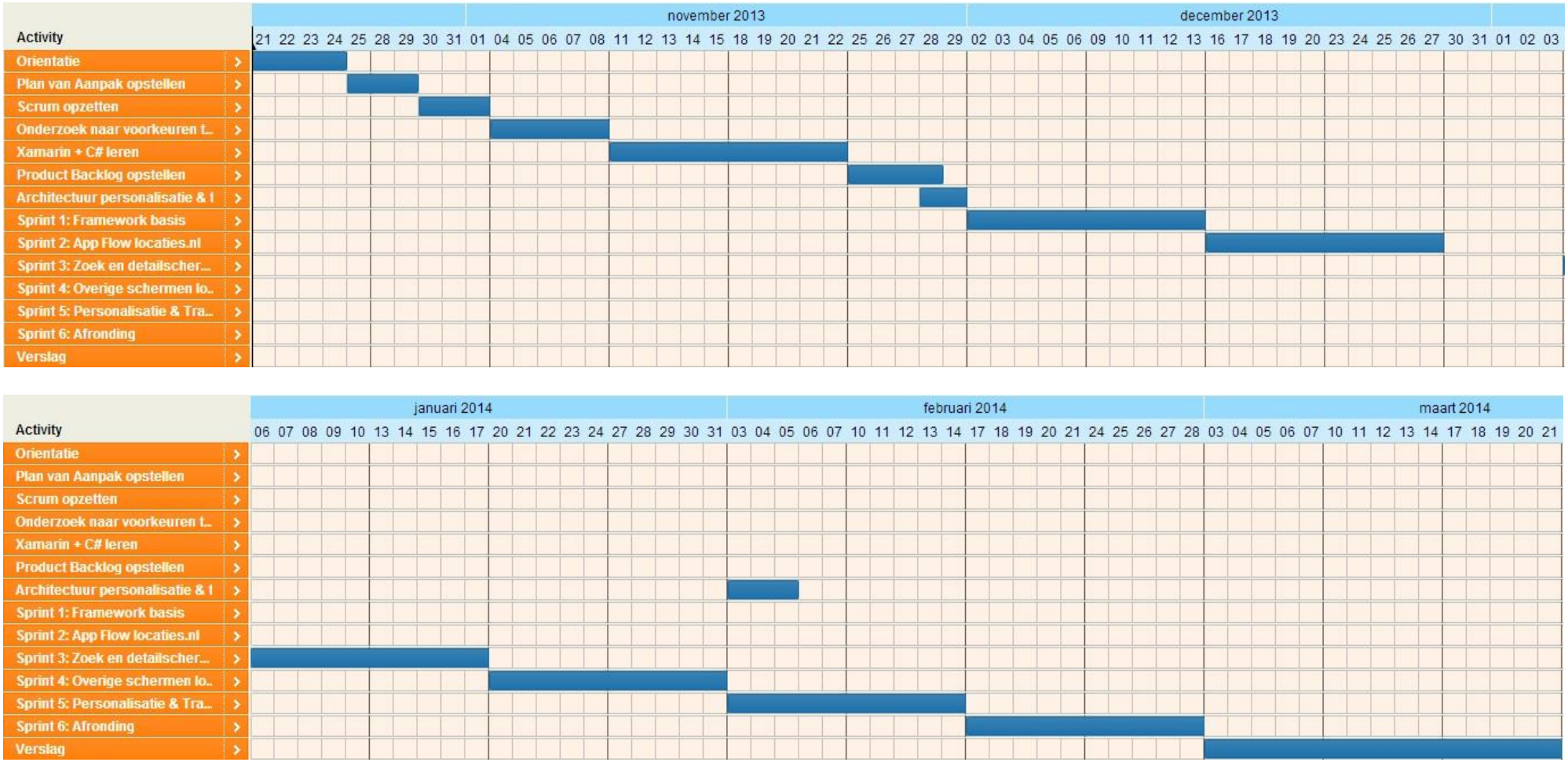
De vierde sprint zal de uitwerking betreffen van de secundaire functionaliteiten.

Om personalisatie toe te kunnen passen is er een werkende app benodigd. Daarom is deze sprint als vijfde ingepland. Hierin wordt in de gebouwde app personalisatie ingebouwd.

De High-level requirements kunnen beantwoord worden wanneer de sprints afgerond zijn. Met de bevindingen die opgedaan zijn tijdens het project kunnen deze vragen beantwoord worden. De antwoorden zijn opgenomen in het hoofdstuk “High-level requirements bevindingen”.

5.6. Detail Planning

Nu de sprints gedefinieerd zijn kan er een gedetailleerde planning opgesteld worden. Van 30 december 2013 tot en met 6 januari 2014 is er een week vrij genomen in verband met de feestdagen.



Afbeelding 4: Detail planning

6. Sprint 1: Framework basis

In de eerste sprint wordt de basis voor het framework opgezet. In dit hoofdstuk zal deze sprint beschreven worden. Aan de hand van het onderzoek naar reader apps en interviews met de opdrachtgever is achterhaald wat deze basis inhoudt.

De user stories van deze sprint zijn opgenomen in de bijlagen onder 15.3.1: Framework basis.

Reader apps bevatten over het algemeen de volgende elementen:

- Lijst van items
- Mogelijkheid om te swipen naar ander scherm
- Images in een grid
- Detail pagina
- Detail pagina met metadata
- Mogelijkheid om images in het groot te bekijken

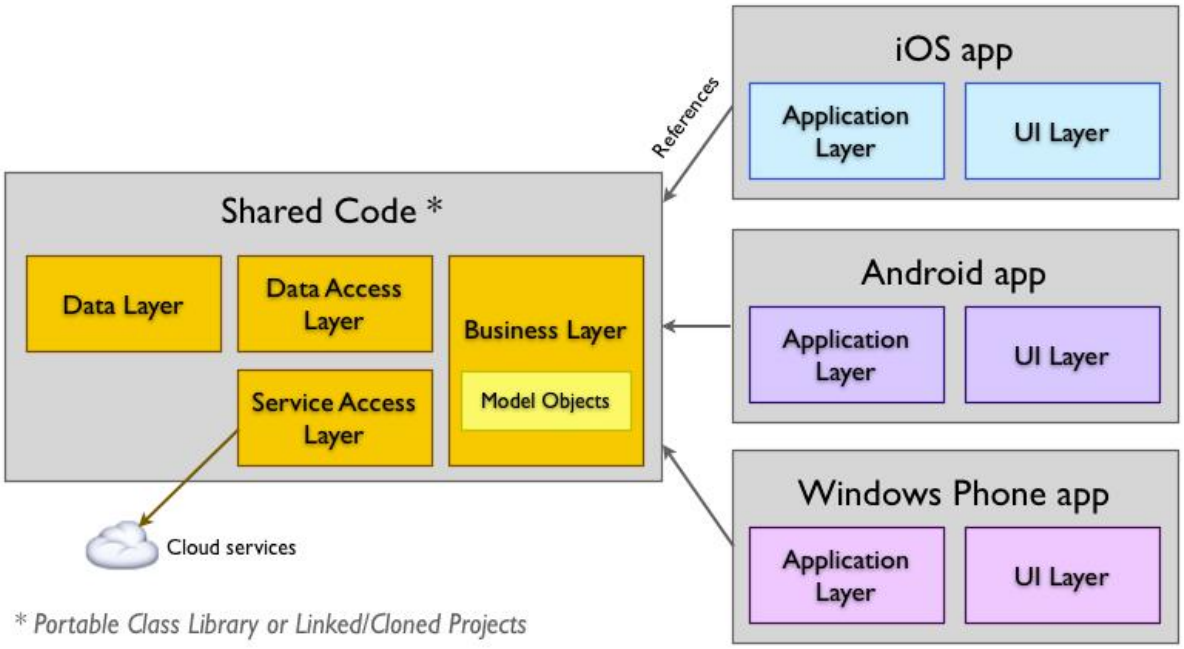
Met het framework moeten er gemakkelijk reader apps ontwikkeld kunnen worden. Het voordeel hiervan is dat er in dit framework gezorgd wordt voor hergebruik van code. Er zal bijvoorbeeld niet in elk project opnieuw een ListView opgebouwd te hoeven worden met alle benodigde logica. Dit zal eenmalig gebeuren in het framework.

Voor de eerste versie van het framework wordt er een algemene shared core en een OS specifieke shared core opgezet. In de algemene shared core staat platform onafhankelijke code die in elk OS gebruikt kan worden. Denk hierbij aan standaarden als RSS etc. om data op te halen en aan standaard classes als Article. De Article class voldoet aan de RSS standaard.

Verder bevat de algemene shared core een View folder waarin interfaces gedefinieerd staan. Dit zijn interfaces voor standaard elementen zoals ListViewController, GridViewController en ScreenSlidePagerViewController, maar ook voor de ViewControllers die geen standaard Android elementen zijn. Voor elk element is er een ViewController ontworpen en ontwikkeld. Elke ViewController implementeert zijn bijbehorende interface. De specifieke implementaties van deze classes zijn opgenomen in de OS specifieke Shared Cores. Hierin staat bijv. gedefinieerd hoe een lijst gemaakt wordt in elk OS of hoe een DetailView voor een locatie opgebouwd is in elk OS.

6.1. Framework architectuur ontwerpen

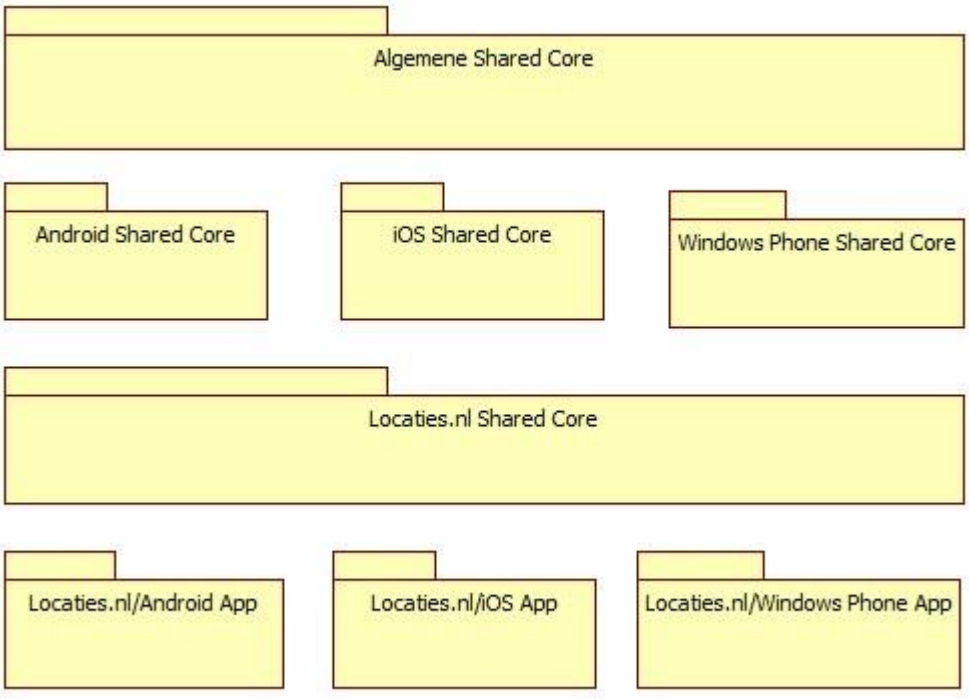
De eerste in deze sprint was het ontwerpen van de architectuur van het framework. Hiervoor heb ik een opzet gemaakt. Deze opzet is gebaseerd op de guidelines van Xamarin.



Afbeelding 5: Architectuur Xamarin

Het idee van Xamarin is om een Shared Code base op te zetten met hierin alle code die niet OS-specifiek is. Echter de rest van de code was wel OS-specifiek en zit daarom in een apart project. Met dit idee van Xamarin wordt er al een deel van de ontwikkelde code hergebruikt op de drie platformen.

Het idee van Xamarin vond ik echter niet optimaal. Indien men een nieuwe reader app zou gaan ontwikkelen, dan zou er veel code verloren gaan die hergebruikt zou kunnen worden. Alle basis elementen van een reader app zouden anders weer opnieuw geschreven moeten worden. Hierdoor ontstond het idee om voor elk OS een Shared Core op te zetten. Dit zou de Application Layer van de apps in de Xamarin architectuur zijn.



Afbeelding 6: Architectuur project

De getoonde architectuur hierboven is een uitbreiding op de Xamarin architectuur. Deze architectuur wordt in het volgende hoofdstuk beschreven.

6.2. Projecten opzetten

In Xamarin Studio zijn 4 projecten opgezet:

- Algemene Shared Core
- Android Shared Core
- Locaties.nl Shared Core
- Android/Locaties.nl App

Deze vier projecten zijn uiteindelijk benodigd om het Android-gedeelte uit te werken voor het gehele project. Voor het ontwerpen en ontwikkelen van het framework zijn de Algemene Shared Core en de Android Shared Core benodigd.

In mijn afstudeerstage werk ik van de OS specifieke projecten de Android versies uit. De iOS en Windows Phone versies worden niet door mij uitgewerkt.

6.2.1. Algemene Shared Core

De algemene shared core bevat niet-OS specifieke code die in elke nieuwe case herbruikbaar is. Het bevat bijvoorbeeld een Service Access Layer waarin een standaard zoals RSS gebruikt wordt. Verder bevat het een View folder met hierin alle ViewController interfaces en de interface IOperatingSystemFactory.

Het bevat de volgende folders:

- Business Layer
- Data Layer
- Data Access Layer

- Service Access Layer
- View

De View folder bevat de interfaces die uiteindelijk geïmplementeerd worden door de ViewControllers en Factories in Android Shared Core. Deze interfaces zijn:

- IOperatingSystemFactory
- IViewController
- IScreenSlidePagerViewController
- IListViewController
- IImageGridViewController
- IDetailViewController
- IContactViewController
- IViewControllerSettings

Door gebruik te maken van deze interfaces wordt het afgedwongen om bepaalde methodes en properties te implementeren in de implementaties van deze Controllers.

6.2.1.1. Strategy Pattern

In de algemene shared core is het Strategy Pattern toegepast. Dit is gebruikt voor datasources in de Service Access Layer. De datasources kunnen als RSS en JSON binnenkomen. De IDataSourceMapper is een interface hiervoor. De RSS en JSON klasses implementeren deze interface. Beide returnen uiteindelijk een Article object. Alleen doen beide dit op een andere manier.

6.2.1.2. References

De Algemene Shared Core heeft alleen een reference naar System. Dit is benodigd om de Algemene Shared Core platform-onafhankelijk te houden.

6.2.2. Android Shared Core

De Android Shared Core bevat Android specifieke code die in elke nieuwe case herbruikbaar is. Het bevat de volgende folders:

- View
- Resources

De View folder bevat de ViewControllers die de interfaces van de Algemene Shared Core implementeren.

- ViewController
- ScreenSlidePagerViewController
- ListViewController
- ImageGridViewController
- DetailViewController
- ContactViewController

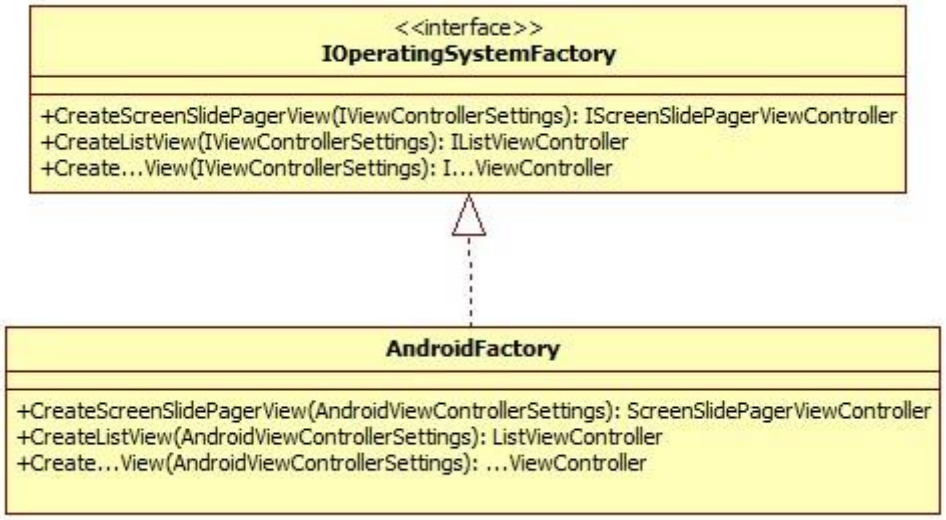
In de ViewControllers zijn de Android implementaties uitgewerkt. Deze bevatten alle logica die benodigd zijn voor de specifieke Controllers.

De Resources folder bevat de layouts, animations, drawables en values die gebruikt worden door de ViewControllers.

Verder bevat de Android Shared Core nog:

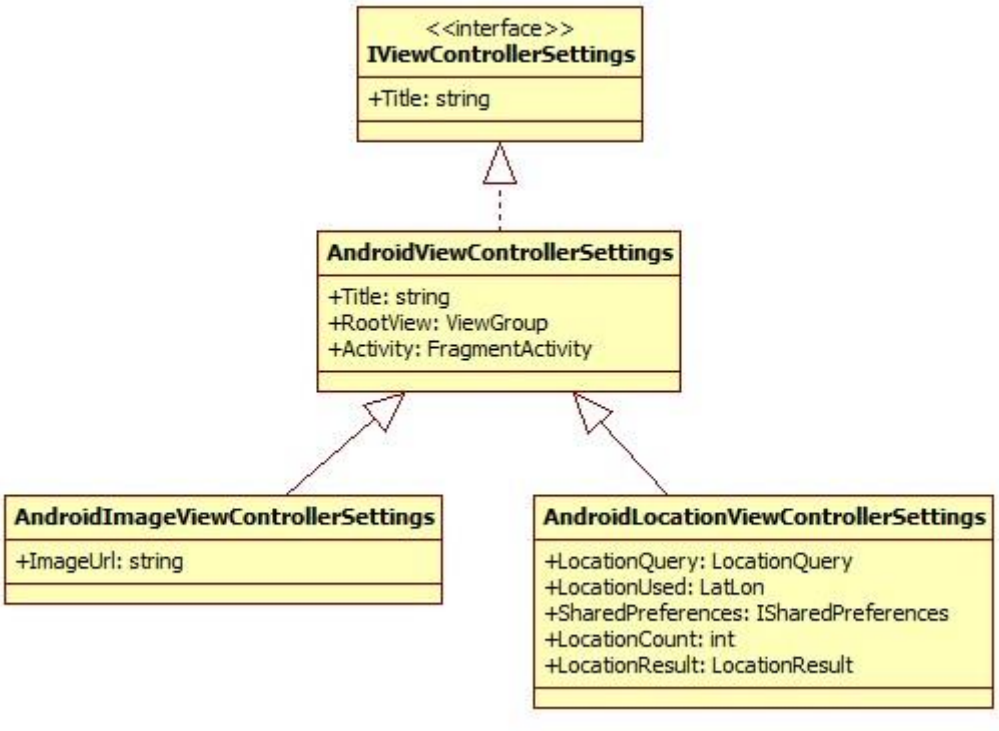
- AndroidFactory
- AndroidViewControllerSettings
- ImageDownloadTask
- ListItem

De AndroidFactory implementeert de IOperatingSystemFactory van de Algemene Shared Core.



Afbeelding 7: AndroidFactory implementeert IOperatingSystemFactory interface

De AndroidViewControllerSettings implementeert de IViewControllerSettings. De AndroidViewControllerSettings bevat de settings die benodigd zijn in de ViewControllers, zoals de ViewGroup, Activity en Title. Maar er is ook specifieke data benodigd in sommige ViewControllers, om deze reden is er overerving toegepast. De subclasses zijn AndroidImageViewControllerSettings en AndroidLocationViewControllerSettings. De AndroidLocationViewControllerSettings wordt bijvoorbeeld gebruikt bij het aanmaken van een activity waarin een locatie gepresenteerd wordt, hier is namelijk locatie data benodigd.



Afbeelding 8: ViewControllerSettings

De ImageDownloadTask download images in een async task. Deze class vraagt in de constructor om een ImageView en een ListItem. Het ListItem bevat een URL van de image die geladen moet worden. Verder bevat het ook een bitmap van de image.

In de ImageDownloadTask worden 3 methodes van de AsyncTask class ge-override. Dat zijn OnPreExecute( ), RunInBackground( ) en OnPostExecute( ). Deze worden tevens in deze volgorde uitgevoerd.

In de OnPreExecute( ) methode wordt de AsyncTask voorbereid. Dit gebeurt op de UI thread. De ViewState van de ImageView wordt geset op Invisible, zodat het niet zichtbaar is.

In de RunInBackground( ) methode worden langdurende taken uitgevoerd op een background thread. Met behulp van de URL uit het ListItem wordt de image gedownload indien deze nog niet gedownload is als Bitmap. Indien het wel gedownload is, dan wordt er niets gedownload.

In de OnPostExecute( ) methode wordt het resultaat van de AsyncTask verwerkt op de UI thread. De Bitmap - die net gedownload is of al beschikbaar was – wordt geset in de ImageView. De Viewstate van de ImageView wordt weer op Visible gezet.

Afbeelding 9: De asynchrone class ImageDownloadTask

De ListItem class wordt gebruikt om data van objecten op te slaan. Bijvoorbeeld om in een lijst de titel, beschrijving en de image te tonen van een locatie, zonder dat het gehele locatieobject in het geheugen gehouden moet worden.

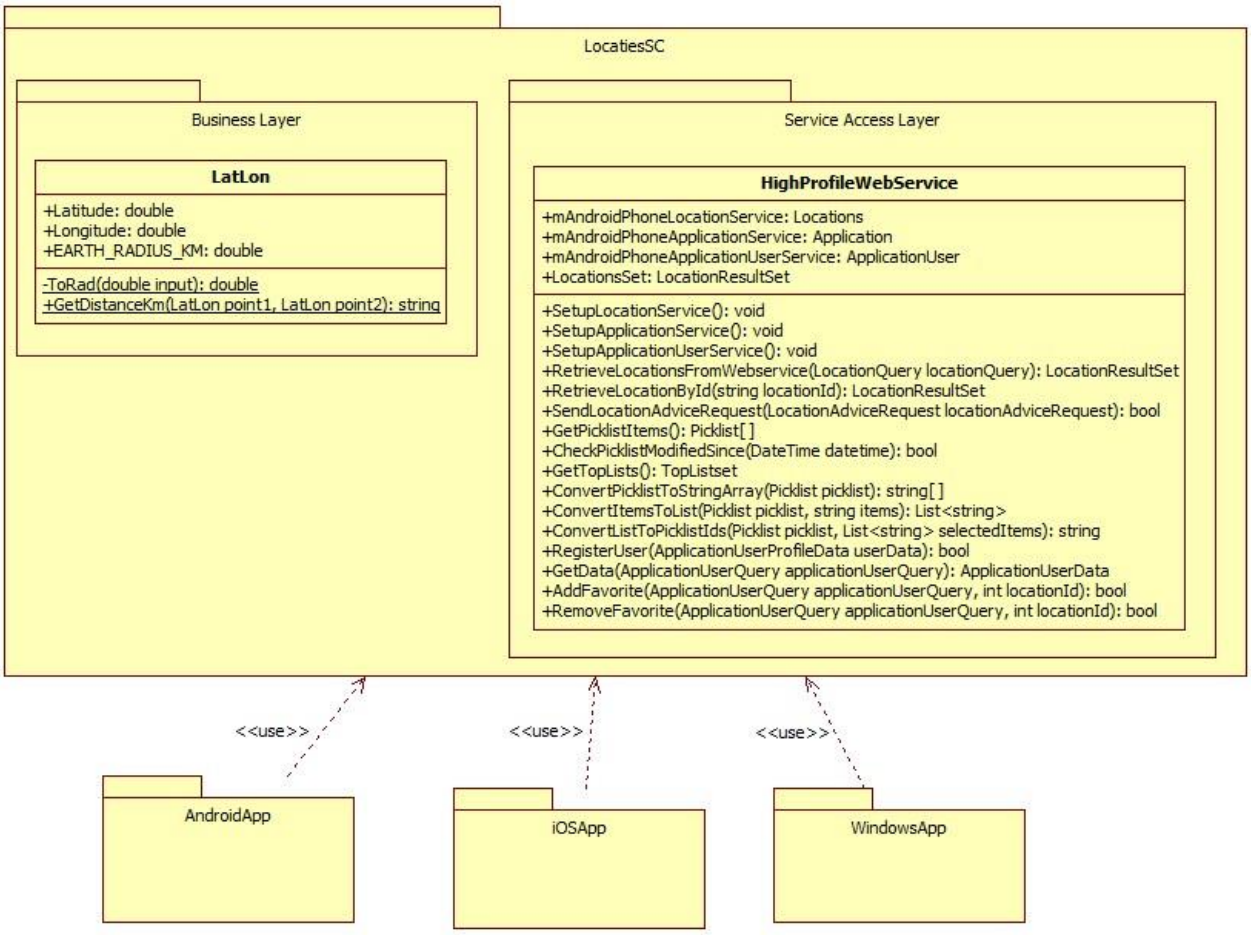
6.2.2.1. References

De Android Shared Core heeft references naar System, Mono.Android, de Algemene Shared Core en de Locaties Shared Core. Dit is benodigd om de Android Shared Core alleen afhankelijk te laten zijn van Android.

6.2.3. Locaties.nl Shared Core

De Locaties.nl Shared Core bevat code die hergebruikt kan worden op alle drie de platforms, maar die wel specifiek is voor de Locaties.nl case. Het bevat de volgende folders:

- Web Reference
- Service Access Layer
- Business Layer



Afbeelding 10: Locaties Shared Core en de toepassing hiervan

De Web Reference folder bevat de webservice van Locaties.nl. Deze webservice wordt gebruikt door de Service Access Layer van deze Shared Core.

In de Service Access Layer wordt de webservice van Locaties.nl geïnitialiseerd en aangesproken. Dit is een SOAP webservice. Deze webservice kan gebruikt worden op alle drie de platformen: Android, iOS en Windows Phone.

De Business Layer bevat de LatLon class, waar de latitude en longitude van een locatie item in opgeslagen kunnen worden. Dit wordt gebruikt in berekeningen voor afstand.

6.2.3.1. References

De Locaties.nl Shared Core heeft alleen references naar System. Dit is benodigd om de Locaties.nl Shared Core platform-onafhankelijk te houden.

6.2.4. Locaties.nl/Android App

Dit is het Android project waarin de app gebouwd zal worden. In deze sprint is dit project minimaal uitgewerkt, alleen om te testen of de gemaakte ViewControllers naar behoren werken. In sprint 2 zal de flow hierin uitgewerkt worden. In sprint 3 zal er een implementatie gemaakt worden voor de Activities met de hoogste prioriteit. In sprint 4 worden de overige pagina's uitgewerkt.

6.2.4.1. References

De Locaties.nl/Android App heeft references naar System, Mono.Android, de Algemene Shared Core, de Locaties.nl Shared Core en de Android Shared Core.

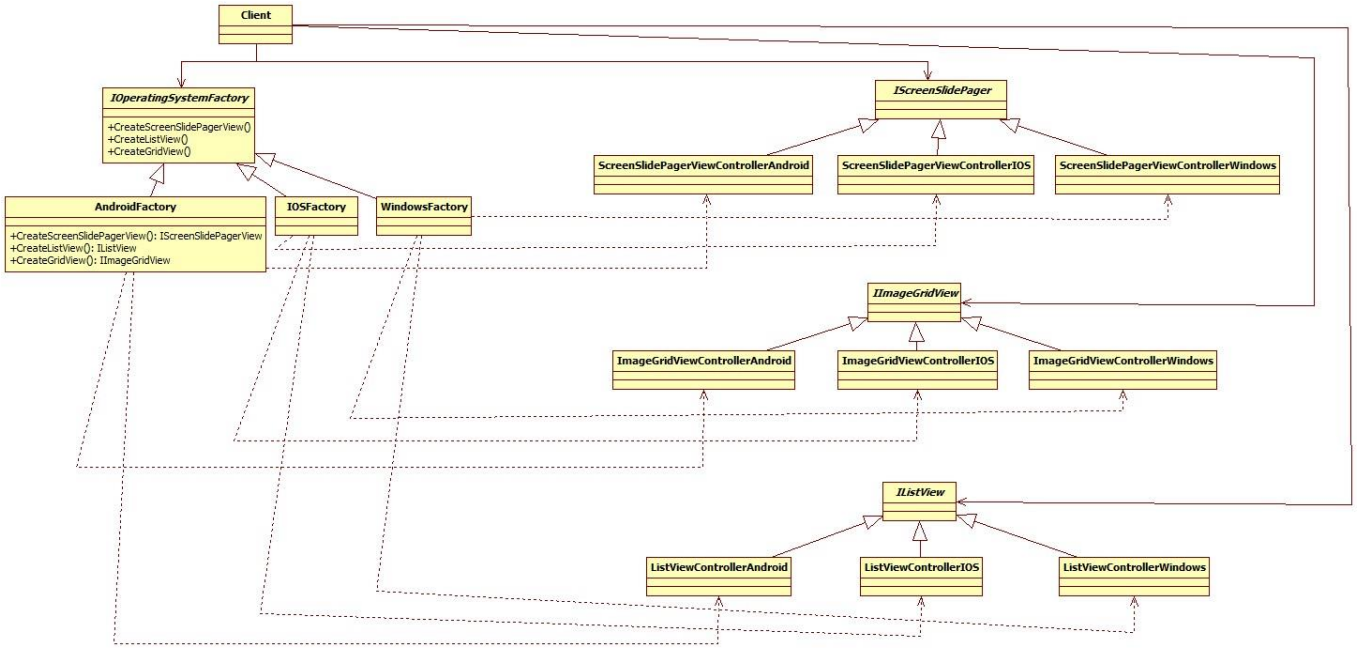
6.3. ViewControllers

De ViewControllers zijn opgezet volgens het Abstract Factory Pattern.

6.3.1. Abstract Factory Pattern

In het framework is het Abstract Factory Pattern toegepast. Hiermee wordt een interface geleverd voor het creëren van families van gerelateerde of afhankelijke objecten zonder hun concrete classes te specifiëren.

Dit is gedaan voor de verschillende OS-en en de verschillende beschikbare ViewControllers. Er is een IOperatingSystemFactory interface en er zijn ViewController interfaces. Elke klasse die afgeleid is van de IOperatingSystemFactory kan zijn eigen implementatie maken van elk type view.



Afbeelding 11: Abstract factory pattern

In het bovenstaande diagram zijn niet alle typen ViewControllers opgenomen. De ViewControllers die niet opgenomen zijn, zouden op precies dezelfde manier weergegeven worden als de getoonde ViewControllers.

Elk OS heeft zijn eigen factory om de benodigde Views aan te maken. Elke OS factory implementeert het IOperatingSystemFactory interface. Dit dwingt af om de vooraf gedefinieerde methodes te implementeren.

Voor elk OS is er een specifieke implementatie voor ListView, omdat dit simpelweg niet anders kan. Deze specifieke implementaties implementeren wel allemaal een interface van de betreffende View.

Uiteindelijk wordt er in de app een Factory aangemaakt en deze maakt de ListView etc. aan.

6.3.2. Uitbreidbaarheid

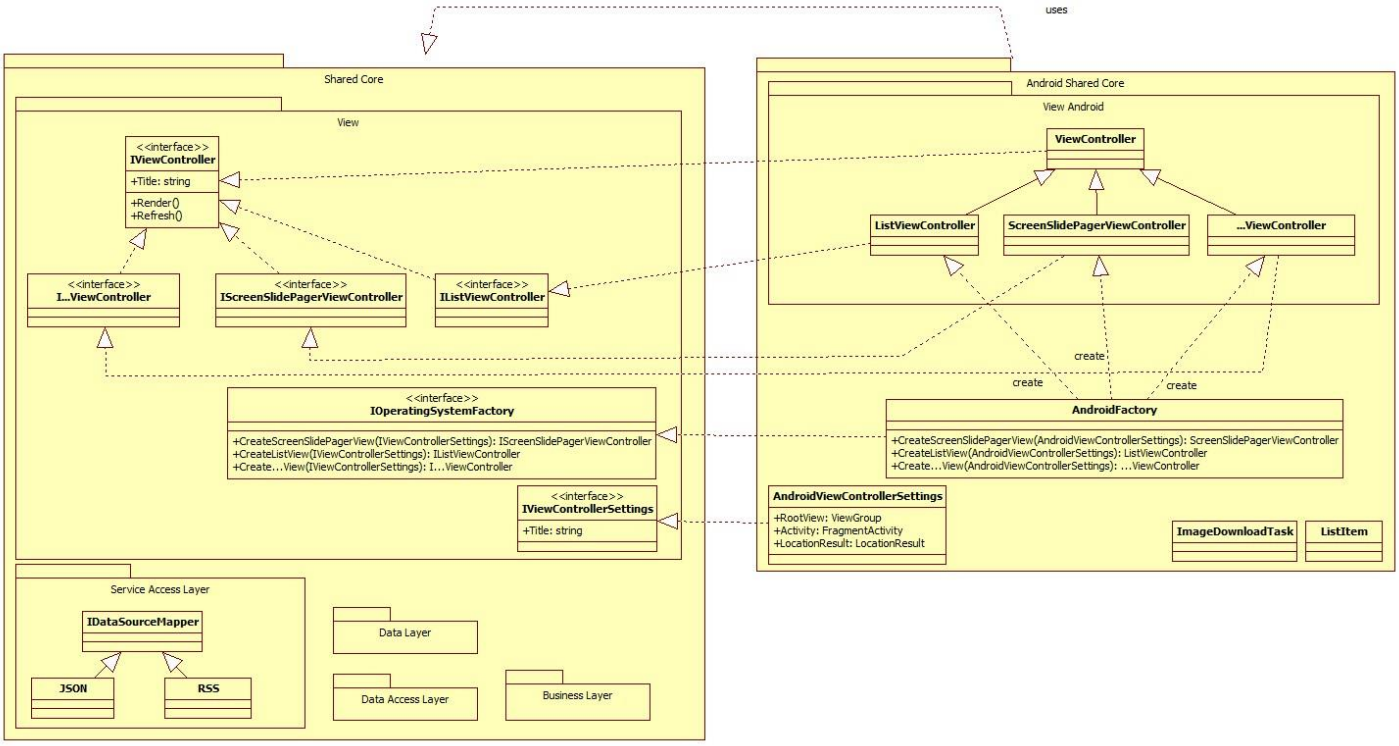
Door de toepassing van het Abstract Factory Pattern is het framework goed uit te breiden. Indien er de behoefte ontstaat naar nieuwe ViewControllers, dan kunnen deze toegevoegd worden in de huidige structuur. De structuur van het framework hoeft dan niet aangepast te worden.

Indien er een nieuw Operating System ondersteund moet worden door het framework, dan is dit mogelijk. Er kan dan simpelweg een nieuwe Factory toegevoegd worden aan de huidige structuur van het framework. Hiernaast zal er een nieuwe Shared Core voor dit Operating System ontwikkeld moeten worden. Deze Shared Core zal de functionaliteiten van de bestaande ViewControllers moeten implementeren voor dat Operating System.

Doordat het framework op deze manier is opgezet is er gedacht aan de uitbreidbaarheid voor eventuele toekomstige veranderingen.

6.3.3. Klassendiagram

In deze paragraaf wordt het klassendiagram van de framework basis getoond, in dit diagram is zijn niet alle Controllers getoond vanwege de leesbaarheid. De functie van het diagram is de werking van het framework uitleggen. Verder zal elke aparte Controller in het framework beschreven worden in een aparte paragraaf. In het klassendiagram zijn geen properties opgenomen, aangezien dit de leesbaarheid van het diagram niet ten goede komt. Op deze manier komt de architectuur beter naar voren.



Afbeelding 12: Framework basis

6.3.4. ViewController

De ViewController is een abstracte class waar alle andere ViewControllers van erven. Het bevat methodes als Refresh() en Render(). Verder bevat het de property Title en de implementatie voor de OnClickListener.

6.3.5. ScreenSlidePagerViewController

De ScreenSlidePager heeft als functionaliteit dat je kan swipen tussen verschillende pages. Deze pages zijn van het type ViewController en zijn opgenomen in een List in de ScreenSlidePager. De lijst accepteert alle Controllers die van ViewController erven.

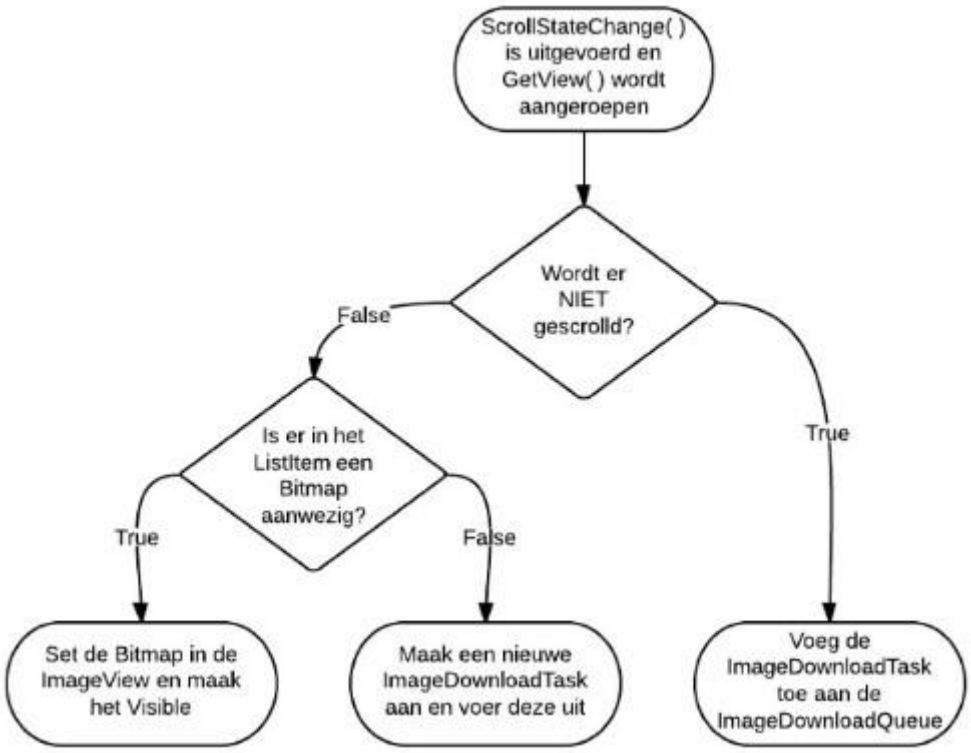
6.3.6. ListViewController

De ListViewController heeft als functionaliteit het weergeven van een lijst. In de ListViewController wordt er een custom ListAdapter toegevoegd waarin een aantal list methodes override worden. Waarvan de belangrijkste GetView() is. Hierin wordt de data op de list gezet. In de customadapter zitten 3 Views. Dit zijn 2 TextViews en een ImageView. De 2 TextViews worden in GetView() altijd geset. De ImageView wordt geset afhankelijk van de ScrollState in de list. Er zijn in het totaal drie ScrollStates:

- Idle
- TouchScroll
- Fling

Idle houdt in dat de lijst niet bewogen wordt. TouchScroll houdt in dat de lijst bewogen wordt met de vinger aan het scherm. Fling houdt in dat de lijst bewogen wordt zonder dat het scherm meer aangeraakt wordt, er is dus een 'gooi' gegeven aan de lijst. Op basis van deze ScrollStates wordt er telkens als de ScrollState verandert een methode aangeroepen. In deze methode, ScrollStateChange( ), wordt er gekeken welke ScrollState actief is. Afhankelijk hiervan worden er met booleans de verschillende states opgeslagen.

In GetView() wordt met behulp van deze booleans het gedrag van de list bepaald.



Afbeelding 13: Flowchart van scroll-logica van de ListViewController

Indien er niet gescrolld wordt, en de bitmap is nog niet geset dan wordt er een nieuwe ImageDownloadTask uitgevoerd. Deze task laadt async de images in.

Als er Fling gescrolld wordt dan worden de ListItems die voorbijkomen toegevoegd aan een ImageDownloadQueue. Deze accepteert een ImageDownloadTask en het totaal aantal zichtbare items op dat moment.

```
public class ImageDownloadTaskQueue : Queue<ImageDownloadTask>
        {
            public void Add (ImageDownloadTask task, int visibleListItemsCount)
            {
                if (!Contains (task)) {
                    Enqueue (task);
                }

                if (Count > visibleListItemsCount + 1) {
                    Dequeue ();
                }
            }

            public void Execute ()
            {
                foreach (var item in this) {
                    item.Execute ();
                }
                this.Clear ();
            }
        }
```

Als er een task wordt toegevoegd, dan wordt er gecheckt of de queue deze binnengekomen task al bevat. Indien dit niet zo is wordt de task toegevoegd aan het eind van de queue. Het is de bedoeling dat alleen listitems in het huidige scherm geladen worden. Dus als de queue groter wordt dan het totale aantal items in het scherm + 1 dan wordt het eerste item van de queue afgehaald. Op deze manier wordt het aantal image downloads beperkt. Images die alweer uit beeld zijn worden niet gedownload.

In de ScrollStateChanged( ) wordt de queue uitgevoerd indien de ScrollState.Idle aan bod is. ImageDownloadQueue.Execute( ) wordt dan uitgevoerd en hierna wordt de queue gecleared.

In de ImageDownloadTask wordt er in de OnPreExecute() de binnengekomen ImageView op Visibility.Invisible gezet.

In de RunInBackground( ) wordt er gekeken of er van het binnengekomen ListItem al een ImageBitmap is. Als dat zo is dan wordt de ImageBitmap van dat ListItem geset. Indien dit niet zo is wordt er gecheckt of het een ImageUrl bevat. Met behulp van de ImageUrl wordt de Bitmap aangemaakt.

In de OnPostExecute( ) van de ImageDownloadTask wordt de ImageView geset met de gedownloade Bitmap item en de visibility hiervan wordt op Visible gezet. Als laatst wordt het met een Animation getoond met een fadein.

6.3.7. ImageGridViewController

De ImageGridViewController bevat dezelfde functionaliteiten als ListViewController. Om deze reden erft ImageGridViewController dan ook van ListViewController.

Het verschil tussen de List- en ImageGridViewController is de manier waarop de data gepresenteerd wordt.

De ImageGridViewController bevat ook een customAdapter.

6.3.8. DetailViewController

De DetailViewController is de detailpagina waar een locatie getoond wordt in zijn geheel. Dit bevat een ScrollView met hierin een ImageView en TextViews. Onder de ScrollView is een bar met like-buttons, favoriet-button en een showOnMap-button.

De DetailViewController maakt gebruikt van de ImageDownloadTask, waarbij async de images geload worden.

6.3.9. DetailMetaDataViewController

De DetailMetaDataViewController is de controller die de metadata van een locatie toont. De controller krijgt een LocationResult binnen en toont deze dan. Afhankelijk van welke metadata de locatie bevat worden er onderdelen getoond of niet. De volgende metadata wordt getoond:

- Locatie soort
- Bereikbaarheid
- Ligging & omgeving
- Uitstraling & sfeer
- Bijzonderheden

Elk van deze onderwerpen kan meerdere items bevatten. Indien er een onderwerp geen invulling heeft, dan wordt deze view niet getoond. Omdat het LocationResult de Ids van de picklist items bevat en niet de Name, moet er een conversie gemaakt worden. Dit wordt gedaan met behulp van een methode die in de webservice class is opgenomen. Deze converteert een string met picklist item ids naar een string met picklist item names.

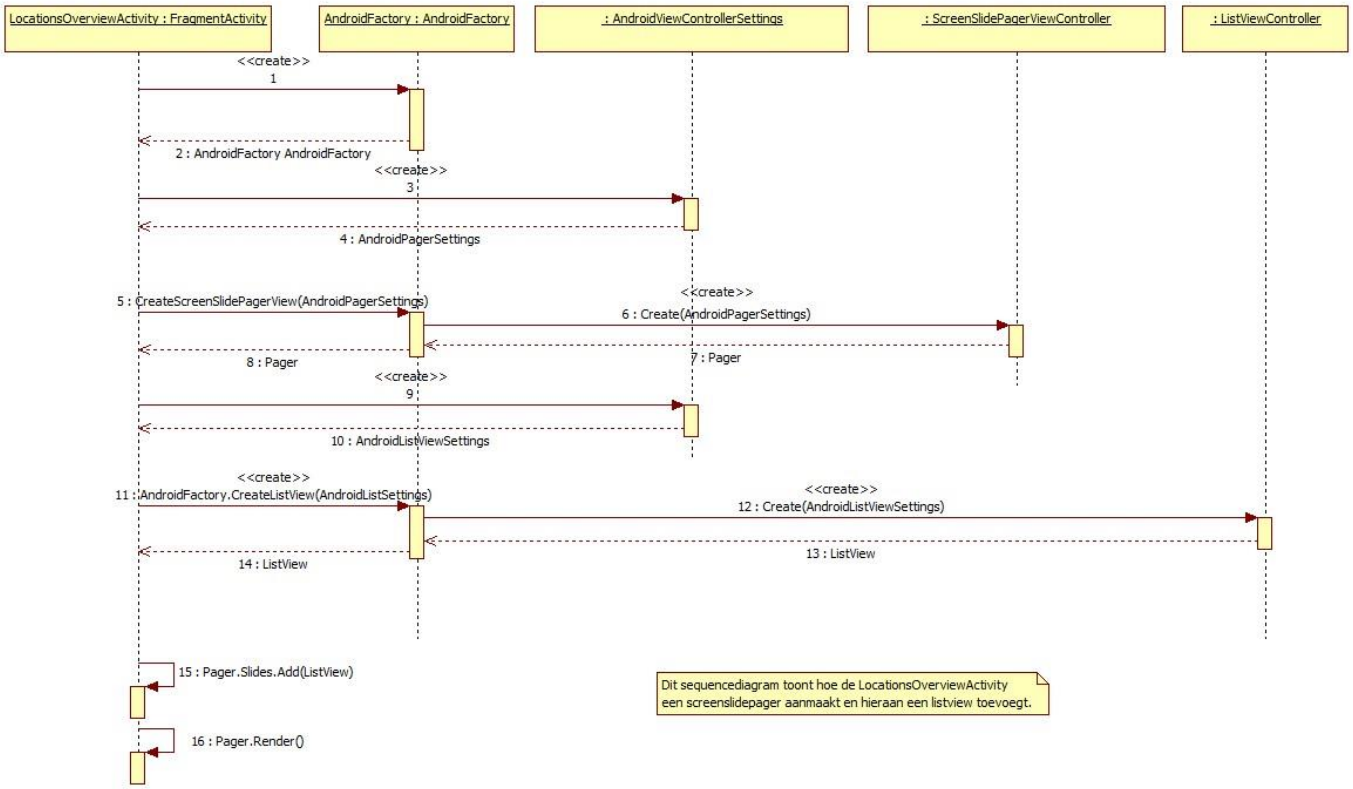
6.3.10. ContactViewController

De ContactViewController is de pagina waar de contactgegevens van een locatie getoond worden.

6.3.11. ImageViewController

In de ImageViewController wordt er een Image asynchroon ingeladen en getoond. Er komt een string binnen met daarin de URL van de image. Indien de image al eerder geladen is en nog in het geheugen zit, dan wordt de image direct getoond zonder het weer te downloaden. Indien de image nog niet gedownload is, dan zal dit worden gedaan met behulp van de asynchrone taak.

6.4. Sequentie diagram Createn van ListView



Afbeelding 14: Sequentie diagram van aanmaken van ListView

6.5. Proces

In deze sprint ben ik begonnen het achterhalen welke ViewControllers er benodigd waren. Dit heb ik gedaan op basis van de functionaliteiten die reader apps over het algemeen bevatten. Dit waren de volgende controllers:

- ViewController
- ScreenSlidePagerViewController
- ListViewController
- ImageGridViewController
- ImageViewController
- DetailViewController
- DetailMetaDataViewController

Deze zijn uitgewerkt en met behulp van het Abstract Factory Pattern opgezet. Ik heb ervoor gekozen om het Abstract Factory Pattern te gebruiken omdat dit goed past in de opgezette architectuur en het doel hiervan.

Abstract factory pattern:

> *The abstract factory pattern provides a way to encapsulate a group of individual factories that have a common theme without specifying their concrete classes. In normal usage, the client software creates a concrete implementation of the abstract factory and then uses the generic interface of the factory to create the concrete objects that are part of the theme. The client doesn't know (or care)*

which concrete objects it gets from each of these internal factories, since it uses only the generic interfaces of their products. This pattern separates the details of implementation of a set of objects from their general usage and relies on object composition, as object creation is implemented in methods exposed in the factory interface.

(Freeman, Robson, Bates, & Sierra, 2004)

Dit was wat ik voor ogen had met de op te zetten architectuur voor het framework.

Tijdens deze sprint was er een kleine complicatie. De behoefte naar een ViewController voor GoogleMaps ontstond. Als dit geïmplementeerd zou worden zou dit het vergemakkelijken om GoogleMap kaarten te tonen in activities. Ik heb geprobeerd deze te ontwerpen en ontwikkelen. Hierbij liep ik echter tegen veel complicaties aan. In overleg met mijn bedrijfsmentor is er besloten om dit niet te implementeren vanwege de vele complicaties die dit opleverde. Uiteindelijk had ik alleen een GoogleMap toegevoegd gekregen, echter zonder hier aanpassingen op te kunnen doen.

Tijdens het gebruik bleek dat er een conflict ontstond voor de gebruiker. Namelijk met het swipen wanneer de MapViewController gebruikt werd. In de Activities wordt namelijk gebruik gemaakt van een Navigation Drawer, deze wordt met behulp van een swipe uitgeschoven. In de ScreenSlidePagerViewController wordt er ook genavigeerd met swipes. De swipe van de Navigation Drawer en ScreenSlidePager werken goed samen. Echter toen de MapViewController toegevoegd werd ontstond er een probleem. De map wordt bewogen door middel van slepen. Dit conflicteerde wanneer men de ScreenSlidePager wilde swipen, dan werd namelijk de Navigation Drawer uitgeschoven. Er kon niet meer terug geswiped worden nu. Dit leverde teveel onduidelijkheden voor de gebruiker op. Daarom is ervoor gekozen om de MapViewController niet te implementeren in het framework.

Gedurende de sprint is er tweemaal overleg geweest met alle betrokkenen van dit project:

- Afstudeerder
- Gjalt Wijma (Opdrachtgever)
- Michel Olivier (Bedrijfsmentor)
- Joël Plas (User Experience & bedrijfsmentor van Arnout Kooij)
- Arnout Kooij (Stagiair CMD)

In het overleg vertelde iedereen waar ze mee bezig waren betreffende het project en wat hun plannen waren voor het vervolg van de sprint. Verder werd de voortgang in de gaten gehouden en werden er ideeën opgedaan/overlegd over eventuele nieuwe functionaliteiten of aanpassingen in bestaande functionaliteiten.

Ik ben begeleid door mijn mentor als ik ergens vastliep in deze sprint. Hij gaf dan tips om me verder op weg te helpen. Verder heb ik veel nagedacht over de architectuur van het framework. Dit legde ik dan voor aan mijn mentor en hier discussieerden/overlegden wij dan over.

6.6. Resultaat

Deze sprint is geslaagd. Het framework is opgezet en de van tevoren gedefinieerde ViewControllers zijn ontworpen en ontwikkeld. Het framework kan nu gebruikt worden om reader apps te ontwikkelen. Dit zal een duidelijke tijdsbesparing opleveren. De sprint is afgerond binnen de opgestelde planning.

7. Sprint 2: App flow Locaties.nl

In deze sprint wordt de flow in de Locaties.nl app opgezet. Dit houdt in dat er genavigeerd moet kunnen worden tussen de verschillende activities en door middel van de navigation drawer. In deze sprint moest ik veel samenwerken met UX designer Arnout Kooij, tevens ook een stagiair. Deze samenwerking gebeurde in de design fase. Ik heb het design gebouwd in de bouw fase.

Van tevoren was de sprint backlog opgezet met stories uit de backlog. Deze user stories zijn opgenomen in de bijlagen onder 15.3.2 Backlog sprint 2: App flow Locaties.nl.

De user stories waren geselecteerd op basis van de belangrijkste functionaliteiten die de app moet bezitten. Dit waren de volgende functionaliteiten:

- Zoek in de buurt
- Zoek op plaats
- Locaties lijst bekijken
- Locatie in detail bekijken

Van deze functionaliteiten was de implementatie niet van groot belang, aangezien de flow van de app centraal stond in deze sprint.

De design en bouw fase werden tegelijk doorlopen, waarin Arnout Kooij leading was in de design fase en ik in de bouw fase. In de design fase heb ik samen met Arnout Kooij nagedacht over het ontwerpen van de app.

Tijdens de sprint is er tweemaal overleg geweest met het gehele team, bestaande uit:

- Gjalt Wijma (Opdrachtgever)
- Michel Olivier (Bedrijfsmentor)
- Joël Plas (Bedrijfsmentor van Arnout Kooij)
- Marc Vivaldi
- Arnout Kooij (stagiair UX Design)

Tijdens dit overleg werden de volgende onderdelen behandeld:

- Voortgang in de gaten houden
- Gemaakte werk bespreken
- Obstakels bespreken
- Brainstormen over moeilijke beslissingen/ontwerpen

7.1. Design fase

Deze sprint had als begin het ontwerpen van de flow door de app. Er was bekend welke functionaliteiten er benodigd waren. Allereerst hebben we de flow tussen deze activities ontworpen. Deze flow is uitgewerkt in een sitemap. De sitemap is als bijlage opgenomen onder 15.2 Sitemap.

De UI van de verschillende schermen is hierna ontworpen. Dit is gedaan met Photoshop en UXPin, dit is een UX design tool. Nadat de afzonderlijke schermen ontworpen waren zijn deze gelinkt aan elkaar in UXPin en zo werd een mockup gecreëerd. In de mockup kan je de gehele flow doorlopen met statische schermen, maar wel met echte clicks. Zo konden er al snel user tests uitgevoerd worden. Arnout Kooij heeft deze user tests afgenomen bij een aantal werknemers/stagiaires.

Uit deze user tests kwamen een aantal onduidelijkheden/fouten naar voren die we weer in de schermontwerpen en flow verwerkt hebben. Er was bijvoorbeeld onduidelijkheid over bepaalde gebruikte iconen. Om deze reden werd er een kleine iconentest opgezet waarbij 3 gebruikers

onderwerpen voorgelegd kregen en daarbij een icoon moesten uitzoeken uit een vooraf opgestelde pool met iconen. Hierdoor zijn een aantal iconen veranderd.

Uiteindelijk heeft Arnout Kooij op basis van onze bevindingen een Android Guidelines document opgesteld voor Liones.

In de design fase voor UI & Interaction Design heb ik vooral een ondersteunende rol gehad, geen uitvoerende rol.

7.2. Bouw fase

Nadat de eerste schermontwerpen klaar waren begon ik met het bouwen van deze activities. Deze activities zijn met behulp van het ontworpen framework opgebouwd. Dit zorgde direct al voor een redelijke tijdswinst. Het bouwen had vele malen langer geduurd indien er geen framework gebruikt werd.

De app bevat naast de flow ook enige functionaliteiten en echte content. Dit komt mede doordat het gemakkelijk op te zetten was met behulp van het framework. Hierin kon met weinig moeite de echte content getoond worden omdat het simpel opgezet was.

Toen uiteindelijk de flow ontworpen was door ons heb ik deze opgenomen in de app. Zo is er uiteindelijk een app met flow en basis functionaliteiten opgeleverd in deze sprint.

7.3. Schermontwerpen

De schermontwerpen van deze sprint zijn in de bijlagen opgenomen onder 15.1 Schermontwerpen.

7.4. Resultaat

Deze sprint is geslaagd. De schermontwerpen zijn gemaakt in samenwerking met Arnout Kooij(UX). Deze zijn opgenomen in de bijlagen onder 15.1 Schermontwerpen. De flow door de app is opgezet in de vorm van een sitemap welke in de bijlage is opgenomen onder 15.2 Sitemap. Verder is er op basis van de flow/sitemap een app ontwikkeld waarin de navigatie is opgenomen. De sprint is afgerond binnen de opgestelde planning.

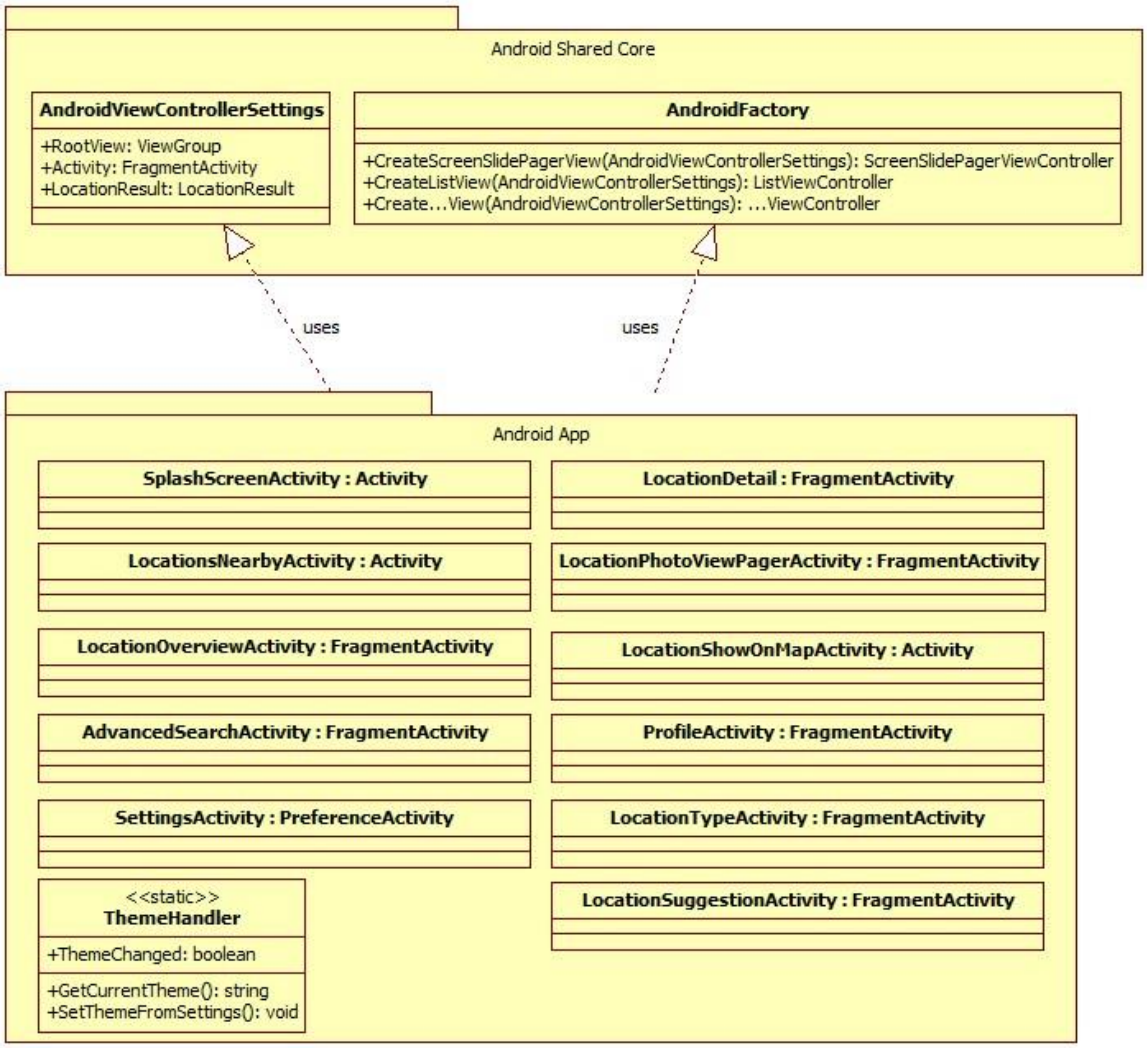
8. Sprint 3: Zoek & detailschermen Locaties.nl

In deze sprint zijn de zoek & detailschermen van de Locaties.nl app ontworpen en ontwikkeld. Dit zijn:

- Splashscreen
- Locatie in de buurt
- Locatie overzicht
- Locatie detail
- Uitgebreid zoeken
- Settings
- Locatie op de kaart tonen
- Foto galerij
- Profiel pagina

Deze activities zijn bedacht op basis van de opgestelde user stories, welke in de bijlagen zijn opgenomen onder 15.3.3 Backlog sprint 3: Zoek & detailschermen Locaties.nl.

De schermontwerpen van deze Activities zijn te zien in de bijlagen, onder ‘Schermontwerpen’.



Afbeelding 15: Structuur Locaties.nl app/Android Shared Core

Hierboven is een deel van de Android Shared Core getoond, er is voor gekozen om niet alle klassen te tonen aangezien dit de duidelijkheid van het diagram zou verminderen. Van de belangrijkste onderdelen is getoond hoe deze gebruikt worden.

8.1. Webservices Locaties.nl

Om de content van Locaties.nl te kunnen tonen waren de webservices van Locaties.nl benodigd. In Sprint 1 was de Locaties Shared Core reeds opgezet. In deze sprint zijn de benodigde methoden ontwikkeld, deze maken gebruik van de webservice.

In de Service Access Layer van de Locaties.nl Shared Core wordt de webservice geïnitialiseerd en gebruikt. Deze webservice bevat drie onderdelen:

- Locations webservice
- Application webservice
- ApplicationUser webservice

De Locations webservice wordt gebruikt om de content(locaties) van Locaties.nl op te halen. Deze webservice verwacht een LocationQuery waarin alle metadata meegestuurd wordt waar de gebruiker op zoekt. Afhankelijk van welke methode gebruikt wordt uit de Location webservice, wordt er een LocationResultset of LocationBaseResultset gereturned. Het verschil tussen deze twee is dat de LocationResultset rijker gevuld is met metadata dan de LocationBaseResultset. Over het algemeen wordt er in de app gebruik gemaakt van de LocationResultset.

Naast de Location webservice is er de Application webservice. Deze webservice wordt gebruikt om alle data op te halen die los staat van de specifieke locaties. De data in de Picklist is de metadata waarop locaties gezocht en gefilterd kunnen worden:

- Locatie soort
- Locatie ligging
- Bijzonderheden
- Landen
- Provincies
- Plaatsen
- Locatie interieur
- Locatie greenkey
- Locatie bereikbaarheid
- Soort event
- Locatie marktsegmenten

Als laatste is er de ApplicationUser webservice. Deze webservice wordt gebruikt om alle user operaties uit te voeren. Dit zijn de volgende methodes:

- Inloggen
- Registreren
- Favoriet toevoegen
- Favoriet verwijderen
- Update user

8.2. SplashScreenActivity

Deze activity toont een splashscreen aan de gebruiker terwijl er data ingeladen wordt. De data die ingeladen wordt, wordt opgeslagen in de SharedPreferences van de app. Deze data is de Picklist die van de applicationwebservice van Locaties.nl opgehaald wordt. Deze Picklist wordt op meerdere plaatsen in de app gebruikt.

Elke keer als de app opgestart wordt, wordt in deze activity met behulp van de webservice gekeken of de data veranderd is. Als de data aangepast is, dan wordt de nieuwe data opgehaald. Indien er niks veranderd is, dan wordt er geen nieuwe data opgehaald en zal de data uit de SharedPreferences verder gebruikt worden in de app.

8.3. LocationsNearbyActivity

Deze activity wordt geopend nadat het Splashscreen getoond is en kan worden gezien als start activity van de app. Hierin wordt als eerste met behulp van GPS de huidige locatie van de gebruiker bepaald, deze locatie wordt dan op de kaart getoond. Indien er geen GPS signaal beschikbaar is zal er een dialog getoond worden aan de gebruiker waarin de GPS settings geopend kunnen worden.

Verder kan de gebruiker in deze activity ook een adres invoeren en deze plaats dan als startpunt te gebruiken voor het verdere zoeken in de app. De kaart wordt dan gecentreerd op de ingevoerde plaats. Het ophalen van de latitude en longitude wordt gedaan met behulp van een request naar de Google Maps API waarin dit adres wordt meegestuurd. Als response worden de latitude en longitude van dit adres teruggestuurd.

De gebruiker kan na het uitvoeren van een van deze twee methodes locaties gaan zoeken. Hiervoor wordt dan een LocationQuery opgebouwd aan de hand van de huidige locatie of de gezochte plaats. Deze LocationQuery wordt hierna geserialized, en in een Intent als Extra gestopt. In de Activity die gestart wordt, wordt deze string weer deserialized naar een LocationQuery.

Verder bevat de activity een Navigation drawer van waaruit andere activities geopend kunnen worden.

8.3.1. Google Maps

Om de locatie op de kaart te tonen moesten er een aantal voorbereidingen getroffen worden. Om met Google Maps aan de slag te kunnen zijn de volgende items benodigd:

- Google Play services component van Xamarin
- Google Map API key
- Manifest file aanpassen

Allereerst is er aan de Locaties.nl app de Google Play services component toegevoegd vanuit Xamarin Studio. Dit is gedaan via de component store die Xamarin Studio beschikbaar heeft gesteld.

Het verkrijgen van de Google Map API key is iets uitgebreider. Hiervoor moet er in de Google API Console een project aangemaakt worden en hierin moet de Google Maps Android API V2 aangezet worden. Vervolgens moet er een nieuwe Android Key aangemaakt worden. Hiervoor is de SHA-1 fingerprint nodig en de package naam van de Locaties.nl app. Hierna wordt er een Google Map API key gegenereerd die gebruikt kan worden.

De Google Map API key moet opgenomen worden in de manifest file van de app, samen met een aantal permissions.

8.4. LocationsOverviewActivity

Deze activity toont de locaties die opgehaald zijn op basis van de LocationQuery die meegegeven is in de Intent. Deze string wordt deserialized naar een LocationQuery object. Hiermee worden de locaties opgehaald door middel van de webservice uit de Locaties.nl package.

De locaties worden op twee manieren gesorteerd getoond. Dit wordt meegegeven in de LocationQuery en dat zijn sortering op Plaats en Afstand.

De lijst waarin de locaties getoond worden, wordt aangemaakt door gebruik te maken van het framework. Er wordt een ListViewController aangemaakt. De eerste set van items in de lijst worden asynchroon ingeladen door een AsyncTask in de LocationsOverviewActivity. Verdere data inladen wordt afgehandeld door de ListViewController.

Verder bevat de activity een Navigation drawer met opties om te navigeren naar andere Activities.

8.5. LocationDetailActivity

De LocationDetailActivity is verantwoordelijk voor het tonen van de gekozen locatie uit de LocationsOverviewActivity. Het krijgt een LocationResult object meegestuurd in de Intent.

Door middel van het framework worden de volgende controllers aangemaakt:

- ScreenSlidePagerViewController
- DetailViewController
- DetailMetaDataViewController
- ImageGridViewController
- ContactViewController

De Detail-, DetailMetaData-, ImageGrid- en ContactViewController worden aangemaakt en aan de slides van de ScreenSlidePagerViewController toegevoegd. De slider bevat hierdoor 4 slides die elk aparte informatie over de locatie tonen.

Verder zijn er in de ActionBar ook functionaliteiten opgenomen. Namelijk de mogelijkheid om deze locatie tot de favorieten toe te voegen, het tonen van de locatie op de kaart, het tonen van de routebeschrijving in de Google Maps app en de mogelijkheid om deze locatie te delen met andere personen.

8.6. AdvancedSearchActivity

De functionaliteit van deze activity is het uitgebreid zoeken naar locaties mogelijk maken. Er kan gezocht worden op de volgende onderwerpen:

- Trefwoord
- Plaats
- Straal
- Provincie
- Locatie soort
- Bereikbaarheid
- Ligging & omgeving
- Uitstraling & sfeer
- Bijzonderheden
- Greenkey
- Sterrenclassificatie

De data die in de SplashScreenActivity gedownload en opgeslagen is in de SharedPreferences wordt hier gebruikt. De items die in de bovenstaande lijst staan maken gebruik van deze Picklist.

Er kan op één onderwerp gezocht worden of op een combinatie van al deze onderwerpen. Wanneer de gebruiker data selecteert of invult, dan wordt dit bijgehouden. Wanneer de gebruiker wil gaan

zoeken, dan wordt een LocationQuery opgebouwd op basis van deze data. Deze LocationQuery wordt dan weer meegegeven in de Intent die geopend wordt, namelijk in LocationOverviewActivity.

Verder kan de huidige zoekopdracht gereset worden.

8.7. SettingsActivity

In de SettingsActivity moet de gebruiker zijn voorkeuren kunnen wijzigen en opslaan. De volgende voorkeuren zijn beschikbaar:

- Notificaties aan/uit
- Thema instellen

Deze instellingen worden bijgehouden in de SharedPreferences van de app.

Met het aan/uit zetten van de notificaties kan de gebruiker aangeven of hij nieuwe en mogelijk interessante locaties wil ontvangen als notificatie op zijn mobiele device.

Met de instelling van het thema kan de gebruiker kiezen tussen twee thema's, namelijk Holo Dark en Holo Light. Indien deze instelling veranderd wordt, dan wordt de app opnieuw opgestart. Dit heeft als reden dat alleen nieuw aangemaakte activities deze verandering zullen overnemen, omdat in de OnCreate( ) van de activity de methode SetContentView( ) wordt aangeroepen. Activities die al gecreate zijn, zouden het oude thema behouden, omdat de OnCreate( ) niet meer uitvoeren.

Om deze reden is er in de SettingsActivity de functie OnSharedPreferenceChanged(ISharedPreferences sharedPreferences, string key) opgenomen. Hierin wordt er gekeken welke shared preference veranderd is. Indien dit de thema instelling is zal de app opnieuw opstarten.

8.8. LocationShowOnMapActivity

In deze activity wordt een LocationResult opgehaald dat meegestuurd is in de Intent die deze activity start. Van dit object worden de coördinaten opgehaald en gebruikt om de locatie op de kaart te tonen.

8.9. LocationPhotoViewPagerActivity

De functionaliteit van deze activity is om foto's van een locatie te tonen. Het navigeren door deze foto's wordt gedaan door middel van swipes naar links en rechts.

Als eerste wordt er uit de Intent een Photo[ ] gehaald. Deze class is afkomstig van de webservice van Locaties.nl. en zit in een LocationResult. Voor elke foto in deze Photo[ ] is er een naam, id en een URL aanwezig.

Hierna wordt er een ScreenSlidePagerViewController aangemaakt en aan de Slides worden ImageViewControllers toegevoegd. In de ImageViewController wordt de foto asynchroon ingeladen.

8.10. ProfileActivity

Deze activity toont het profiel van de gebruiker. Hierin worden de favoriete locaties getoond. Dit gebeurt met behulp van een ScreenSlidePagerViewController met een ImageGridViewController. Op dit moment is deze activity nog gevuld met dummy data. Dit wordt uitgewerkt in Sprint 4.

8.11. ThemeHandler

Deze class is verantwoordelijk voor het setten van het gebruikte thema in de app. Deze class is gedefinieerd als static omdat er geen instantie van deze class nodig is en er maar één thema actief is in de app.

In elke activity wordt voor de OnCreate( ) van deze class de methode SetThemeFromSettings( ) aangeroepen. Hierin wordt dan van de SharedPreferences in de app opgehaald welk thema er door de gebruiker geselecteerd is. Op basis hiervan wordt dan het thema ingesteld in elke activity.

8.12. Resultaat

Deze sprint is geslaagd. De app die ontwikkeld is in deze sprint bevat alle functionaliteiten van de sprint backlog van deze sprint. De sprint is afgerond binnen de opgestelde planning.

9. Sprint 4 – Overige schermen Locaties.nl

In deze sprint zijn de overige schermen van de Locaties.nl app ontworpen en ontwikkeld. Dit zijn:

- Login
- Registreren
- Profiel pagina
- Favoriete locaties
- Top 10
- Locatie advies aanvraag

De user stories die bij deze activities horen zijn opgenomen in de bijlagen, onder 15.3.4 Backlog sprint 4: Overige schermen Locaties.nl.

De schermontwerpen van deze Activities zijn te zien in Sprint 2, onder het hoofdstuk Schermontwerpen.

9.1. Login

Deze activity zorgt ervoor dat de gebruiker met zijn account van Locaties.nl in kan loggen in de app. Het wordt getoond na het tonen van het splashscreen, indien er nog geen gebruiker is ingelogd en is opgeslagen in de SharedPreferences. De gebruiker kan er ook voor kiezen om verder te gaan zonder in te loggen, er zijn dan een aantal functies in de app niet beschikbaar zoals het bekijken en beheren van favorieten.

Voor het inloggen zijn een emailadres en een wachtwoord benodigd waarmee de gebruiker is geregistreerd bij Locaties.nl. De ApplicationUser-webservice van Locaties.nl wordt gebruikt om de gebruiker in te loggen.

De methode wordt uitgevoerd in een AsyncTask. Dit zorgt er namelijk voor dat de UI thread van Android niet vastloopt. De user ondervindt dan geen lag. Indien dit op de UI thread uitgevoerd wordt, dan kan de user lag ervaren indien de internetverbinding niet optimaal is. Het zal dan lijken alsof de app vastloopt.

Er wordt een ApplicationUserQuery object verwacht als parameter, hierin zit de inlogdata, namelijk het emailadres en het wachtwoord. De return value is van het type ApplicationUserData en bevat alle data die van de ingelogde user bekend is. Dit is informatie als Naam, Emailadres, Adres, Favoriete Locaties en Telefoonnummer.

Na het inloggen krijgt de gebruiker feedback of er ingelogd is of niet door middel van een Toast bericht.

Indien er succesvol ingelogd is wordt het ApplicationUserData object geserialiseerd naar een JSON object en opgeslagen in de SharedPreferences van de app. Deze data is vanaf dit punt toegankelijk op elk punt in de app via de SharedPreferences.

Na het succesvol inloggen wordt de gebruiker naar de LocationsNearbyActivity doorgestuurd. Indien er niet ingelogd kon worden dan blijft de gebruiker in de LoginActivity.

9.2. Registreren

Het registreren gebeurt niet in een aparte activity, maar wordt getoond in een Dialog in de LoginActivity.

De ApplicationUser-webservice van Locaties.nl wordt gebruikt om de gebruiker te registreren.

De methode wordt uitgevoerd in een AsyncTask, wederom om de user experience optimaal te houden.

Er wordt een ApplicationUserProfileData object verwacht als parameter, hierin zit alle data die de gebruiker heeft ingevuld bij het registreren. De return value is een bool. True geeft een succesvolle registratie weer, false een onsuccesvolle registratie.

Na het registreren krijgt de gebruiker feedback over het registreren. De gebruiker blijft op de LoginActivity en kan dan inloggen met het zojuist aangemaakte account.

9.3. Profiel pagina

In deze activity kan de gebruiker zijn profiel bekijken van Locaties.nl, indien de gebruiker is ingelogd. In het profiel staat data zoals de naam en het emailadres van de gebruiker. Verder worden alle favoriete locaties van de gebruiker hier getoond.

Bij het opstarten van deze activity wordt er gekeken of er een gebruiker is ingelogd. Dit wordt gedaan door te kijken of er een user is opgeslagen in de SharedPreferences van de app. Indien er geen user is opgeslagen wordt er een button getoond waarmee er naar het loginscherm genavigeerd kan worden. Indien er een user is ingelogd, dan wordt er een button getoond waarmee uitgelogd kan worden. De user zal dan uit de SharedPreferences verwijderd worden.

Indien er een user ingelogd is worden de favorieten getoond in dit scherm. Het tonen van de favorieten gebeurt door middel van een ImageGrid, welke aangemaakt wordt door het gebouwde framework. Het ingelogde user object bevat de favoriete locaties van deze user. Al deze locaties worden getoond in de ImageGrid.

9.4. Favoriete locaties

In de detail pagina van een locatie(LocationDetailActivity) is een button toegevoegd voor het toevoegen en verwijderen van favoriete locaties. Deze functionaliteit is alleen beschikbaar indien de gebruiker ingelogd is. Het toevoegen en verwijderen van favorieten gebeurd door middel van de Locaties.nl webservice. De locaties worden in het profiel van de gebruiker opgeslagen. Dit gebeurt door middel van twee methodes. Respectievelijk de AddFavorite en RemoveFavorite methodes.

Er wordt een ApplicationUserQuery object verwacht als parameter, hierin zit de inlogdata van een profiel, namelijk het emailadres en het wachtwoord. Verder wordt het locationId verwacht van de locatie. Er wordt een boolean gereturned door de webservice, welke aangeeft of het toevoegen/verwijderen succesvol verlopen is.

9.5. Top 10

In de Top 10 Activity wordt de lijst met de 10 meest populaire locaties getoond aan de gebruiker. Deze lijst wordt getoond met behulp van een ListView die aangemaakt wordt door middel van het gebouwde framework. De methode die gebruikt wordt om de data op te halen is hieronder getoond. Er wordt gebruik gemaakt van de ApplicationUser-webservice van Locaties.nl. Hier worden de top locaties opgevraagd.

De webservice geeft een Toplistset terug en deze bevat de 10 meest populaire locaties met al hun data. Deze locaties worden in de ListView getoond aan de gebruiker.

9.6. Locatie advies aanvraag

In de detail activity van een locatie kan de gebruiker ervoor kiezen om een locatie advies aan te vragen. Hiervoor moet de gebruiker dan zijn persoonlijke data opgeven en opsturen via de app. Deze data wordt ingevuld in een Dialog. Hierin vult de gebruiker zijn Naam, Emailadres, Telefoonnummer, Opmerking en een beveiligingscode in. Het id van de locatie wordt ook meegestuurd, dit gebeurt automatisch in de activity.

Voor het aanvragen van het locatie advies wordt de Locations-webservice gebruikt van Locaties.nl. De methode wordt in een AsyncTask uitgevoerd, om de user experience zo optimaal mogelijk te houden. De ingevulde data wordt aan de methode meegegeven als parameter. De methode geeft een boolean terug. True voor een succesvolle aanvraag en false voor een mislukte aanvraag. Op basis hiervan krijgt de gebruiker feedback door middel van een Toast bericht.

De succesvolle aanvraag komt terecht bij de eigenaar van de locatie, door middel van de locatie id. De eigenaar zal deze locatie advies aanvraag verder behandelen.

9.7. Resultaat

Deze sprint is geslaagd. De app die ontwikkeld is in deze sprint bevat alle functionaliteiten van de sprint backlog van deze sprint. De sprint is afgerond binnen de opgestelde planning.

10. Sprint 5: Personalisatie & Tracking

In deze sprint wordt de personalisatie & tracking in de app voor Locaties.nl beschreven. Verder wordt er ook uitgelegd hoe de gebruiker persoonlijke notificaties ontvangt op zijn device. Deze sprint heeft als einddoel dat de gebruiker locatie suggesties krijgt op basis van zijn gedrag in de app.

De user stories van deze sprint zijn opgenomen in de bijlagen onder 15.3.5 Backlog sprint 5: Personalisatie & Tracking.

10.1. Globaal ontwerp Personalisatie & Tracking

Het is de bedoeling om de gebruiker zo goed mogelijke suggesties te doen. Hiervoor zijn een aantal onderdelen benodigd. Het gedrag van de gebruiker in de app zal vastgelegd worden. Dit wordt gedaan op twee manieren.

Als eerste wordt er bijgehouden welke locaties de gebruiker bekijkt. De data van deze locaties zal lokaal opgeslagen worden in een SQLite database. Op deze manier wordt er veel data verzameld van de gebruiker.

Om de gebruiker goede suggesties te kunnen geven zal elke locatie een score moeten krijgen door middel van een algoritme. Dit algoritme zal in detail beschreven worden in hoofdstuk Technisch Ontwerp.

Om inzichtelijk te maken hoe de app de gebruiker voorziet van locatie suggesties, wordt er een dialog getoond waarin beschreven wordt hoe de suggesties tot stand komen. Er wordt getoond op welke metadata de gebruiker veel klikt. Op basis van deze metadata worden nieuwe suggesties voorgelegd aan de gebruiker.

Als alternatief op de hierboven beschreven functionaliteit had het personalisatie ook op een andere manier aangepakt kunnen worden. Dat had met ElasticSearch geweest. ElasticSearch is een krachtige, open source, real-time search en analytics engine. Hierin worden objecten opgeslagen als gestructureerde JSON documenten. De oplossing met ElasticSearch zou veel sneller en schaalbaarder zijn. Echter het opzetten van deze omgeving om te kunnen beginnen met het ontwerpen en ontwikkelen van de personalisatie zou teveel tijd kosten. Dit zou minimaal een week duren. Om deze reden is ervoor gekozen om de oplossing te gebruiken die hiervoor is beschreven.

10.2. Technisch ontwerp Personalisatie & Tracking

Het technisch ontwerp van deze sprint staat beschreven in dit hoofdstuk.

10.2.1. Benodigdheden

Voor het uitvoeren van deze sprint zijn er een aantal benodigdheden. Dit zijn:

- Database met tracking gegevens van gebruiker
- Locaties opgehaald van de webservice van Locaties.nl
- Algoritme om locaties te scoren

10.2.2. Database

Voor deze sprint was het nodig om de metadata van een locatie op te slaan in de database. Hiervoor is gekozen om een component te gebruiken, namelijk SqLite.NET. Deze component is een cross-platform SQLite client en Object Relational Mapping. Hiermee kan een SQLite database aangemaakt en gebruikt worden op de verschillende platforms.

De SQLite component vergemakkelijkt het proces van het aanmaken en gebruiken van de database. Er kan een class gedefinieerd worden in C#, hiervan kan SQLite.NET dan een Table van maken in de database. Verder bevat de component functies om CRUD statements uit te voeren.

Om personalisatie mogelijk te maken moet het gedrag van de gebruiker bijgehouden worden. Er is gekozen om de metadata van de door de gebruiker geopende locaties op te slaan in de tabel Tracking.

Afbeelding 16: De tabel Tracking

De data die bijgehouden wordt in de tabel Tracking staat hierboven beschreven. Een locatie wordt in zijn geheel opgeslagen in de database. In de tabel bestaan 3 soorten velden: Integers, Booleans en Varchars. De data die in de Varchar velden zijn opgeslagen zijn CSV's met ID's van Picklist items vanuit de webservice. De Varchar velden kunnen meerdere items bevatten en kunnen gezien worden als Boolean[ ]. Als een ID van een Picklist item voorkomt in de string, dan staat deze waarde op true. Indien het niet voorkomt staat deze waarde op false. Dit principe geldt voor alle andere Varchar velden. Dit zal later aan de hand van een voorbeeld uitgelegd worden.

Er is gekozen voor deze platte structuur zonder 1…* relaties omdat de webservice de locaties op deze manier aanlevert. Het toevoegen van extra tabellen zou voor onnodige complexiteit zorgen, aangezien deze opgeslagen locaties vergeleken zullen worden met locaties van de webservice. Een aantal velden zouden dan geconverteerd moeten worden.

ID	Naam
69	Attractieparken
139377	Golfbanen
73	Hotels
74	Industriële locaties
179905	Kastelen, land- en herenhuizen
76	Kerken en kloosters
139378	Mobiele locaties
77	Musea en Galeries
78	Partycentra
80	Partyschepen en rederijen
166	Beursgebouwen
79	Restaurants
81	Schouwburgen en theaters
139371	Sportaccommodaties
82	Strandpaviljoens
139380	Studio's en Concertlocaties
83	Tenten
169	Treinen
170	Bioscopen
7103	Boerderijen, hoeves en molens
70	Buitenlocaties
139382	Casino's
139379	Clubs en discotheken
139361	Congres- en vergaderlocaties
139376	Dierenparken

Afbeelding 17: De Picklist items van Locatie Soort

Nu deze data beschikbaar is in de app, kan het gebruikt worden voor het doel van deze sprint, namelijk personalisatie. Het aanbieden van locatie suggesties aan de gebruiker.

10.2.3. Algoritme

Nu de data beschikbaar is in de app kan ermee gewerkt worden.

De bedoeling van het ontworpen algoritme is om een locatie te scoren op basis van de bijbehorende metadata. Deze metadata wordt vergeleken met het opgebouwde profiel van de gebruiker en aan de hand hiervan gescoord. Voor de Integer en Boolean velden zal er een score berekend worden tussen de 0 en 1. Voor de Varchar velden zal er per Picklist item een score tussen de 0 en 1 berekend worden. Deze scores zullen opgeteld worden en de score vormen voor dat veld.

In het totaal zijn er 3 static methodes die geschreven zijn, dit zijn:

- `public static int CountRowsInTracking()`
- `public static double ScoreBool(bool condition, int nrOfTrue, int totalRows)`
- `public static double ScoreInt(double fieldValue, double origin)`

Deze methodes staan gedefinieerd en geïmplementeerd in een static class, DatabaseMethods. Dit heeft als reden dat er geen instantie van deze class benodigd is, alleen de methodes in deze class zijn van belang.

In de CountRowsInTracking methode worden het totaal aantal records in de Tracking database opgeteld en teruggegeven. Deze waarde is later van belang bij het berekenen van de tussenscores.

10.2.3.1. Scoren van Boolean velden

Voor de boolean velden wordt de score van een locatie als volgt berekend. We kijken voor dit voorbeeld naar de database met de volgende records. Dit is het profiel van de gebruiker wat opgebouwd is door locaties te openen.

id	LocationId	LocationName	BeamerAanwezig	EigenCateringMogelijk	GeluidsInstallatieAanwezig
1	133728	Studio's Aalsmeer	1	1	1
2	134092	VVC Adventure	0	0	1
3	134198	BoatHouse	1	0	1

Afbeelding 18: Opgebouwd profiel van de gebruiker, een deel van de velden is weergegeven

De score zal berekend worden per Boolean veld. Dit zijn BeamerAanwezig, EigenCateringMogelijk en GeluidsInstallatieAanwezig. Per boolean veld wordt er geteld hoeveel records true(1) zijn en deze waarde wordt bijgehouden in een variabele. Verder wordt het totaal aantal records van de Tracking tabel gebruikt. Deze waarden worden later in de ScoreBool( ) methode meegestuurd als parameters.

Van alle locaties die in de webservice beschikbaar zijn wordt de score aan de hand van het profiel berekend. Voor elk boolean veld zal met behulp van de ScoreBool( ) methode een score berekend worden.

Na het bepalen van deze waarden wordt er voor elke locatie van de webservice van Locaties.nl, een score berekend. Hierbij wordt de ScoreBool( ) methode gebruikt.

In de ScoreBool( ) methode gebeurt het volgende:

```
public static double ScoreBool(bool condition, int nrOfTrue, int totalRows) {
            double score = 0;

            if (condition && nrOfTrue != 0) {
                score = (double)nrOfTrue / (double)totalRows;
            } else if (condition && nrOfTrue == 0) {
                score = 0;
            } else if (!condition && nrOfTrue != totalRows) {
                score = (double)(totalRows - nrOfTrue) / (double)totalRows;
            } else if (!condition && nrOfTrue == totalRows) {
                score = 0;
```

```
            }

            return score;
        }
```

Als parameters komen binnen:

- De Boolean waarde van het veld van de locatie waarvoor de score berekend wordt
- Het aantal maal dat dit veld op true staat in het gebruikersprofiel
- Het totaal aantal records in het profiel.

Afhankelijk van de condition en de waarde van nrOfTrue wordt er een score berekend.

Hieronder een voorbeeld voor het scoren van de boolean velden.

LocationId	LocationName	BeamerAanwezig	EigenCateringMogelijk	GeluidsInstallatieAanwezig
130342	Test Locatie	1	0	1

Afbeelding 19: Test locatie voor scoren boolean velden

Alle locaties van de webservice komen aan bod en worden gescoord. Voor elk boolean veld wordt een score berekend. Dit gaat als volgt. Voor het veld BeamerAanwezig heeft deze locatie een true waarde. De methode ScoreBool(true, 2, 3) wordt uitgevoerd. Dit staat respectievelijk voor de condition, nrOfTrue en totalRows. Deze methode wordt voor elk boolean veld uitgevoerd.

In de methode wordt het eerste if-statement gecheckt, de condition is true en de waarde van nrOfTrue is geen 0, dus wordt het statement eronder uitgevoerd. De waarde van score is dan nrOfTrue / totalRows = 2/3. Deze score wordt gereturned. Deze berekening wordt voor elk boolean veld gedaan. Uiteindelijk worden al deze scores opgeteld.

Hoe groter het profiel van de gebruiker wordt, hoe nauwkeuriger en veelzeggender deze scores worden.

10.2.3.2. Scoren van Integer velden

Voor de Integer velden wordt de score als volgt berekend. We kijken voor dit voorbeeld naar de database met de volgende records. Dit is het profiel van de gebruiker wat opgebouwd is door locaties te openen.

LocationId	LocationName	MaxParkeerPlaatsen	MaxPersonen
133728	Studio's Aalsmeer	750	3500
134092	VVC Adventure	80	500
134198	BoatHouse	100	150

Afbeelding 20: Opgebouwd profiel van de gebruiker, een deel van de velden is weergegeven

We gaan de score berekenen per Integer veld. Dit zijn MaxParkeerPlaatsen en MaxPersonen. Per Integer veld wordt het gemiddelde berekend en deze waarde wordt bijgehouden in een variabele. Deze waarden worden later in de ScoreInt( ) methode meegestuurd als parameters.

Van alle locaties die in de webservice beschikbaar zijn wordt de score aan de hand van het profiel berekend.

Na het bepalen van deze waarden wordt er voor elke locatie van de webservice van Locaties.nl een score berekend. Hierbij wordt de ScoreInt( ) methode gebruikt.

In de ScoreInt( ) methode gebeurt het volgende:

```
public static double ScoreInt(double fieldValue, double origin) {
            if (fieldValue == origin) {
                return 1.0;
            }

            var scale = origin == 0 ? 1 / 6 : origin / 6;

            var bottomEquasion = (Math.Pow (scale, 2)) * 2;
            var topEquasion = Math.Pow ((fieldValue - origin), (2));
            var score = Math.Exp ((-(topEquasion / bottomEquasion)));
            return score;
        }
```

Als parameters komen binnen de waarde van het veld van de locatie die gescoord wordt en het gemiddelde van het te scoren veld in het gebruikersprofiel.

De volgende formule wordt in de ScoreInt( ) methode gebruikt om de score te bepalen:

$$e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Afbeelding 21: Deel van de Bell Curve formule

De waarde x staat voor parameter fieldValue, de waarde μ staat voor parameter origin, de waarde σ staat voor standaardafwijking. Voor de implementatie van deze basis formule neem ik als vuistregel dat de standaarafwijking 1/6 is van parameter origin. Dit is aan de hand van een aantal testsommen bepaald.

De parameter fieldValue is de waarde van het veld van de locatie die gescoord wordt. De parameter origin is de gemiddelde waarde van dat veld in het profiel van de gebruiker.

Aan de hand van de volgende locatie zal er een voorbeeld getoond worden.

LocationId	LocationName	MaxParkeerPlaatsen	MaxPersonen
130342	Test Locatie	300	800

Afbeelding 22: Test locatie voor scoren integer velden

In de foreach loop waarin alle locaties gescoord worden komt de locatie hierboven aan bod. Voor elk integer veld zal er een score worden berekend. Dit gaat als volgt. Van het profiel wordt het gemiddelde per veld bepaald. Voor MaxParkeerPlaatsen is dat: (750 + 80 + 100) / 3 = 310. Voor MaxPersonen is dat: (3500 + 500 + 150) / 3 = 1383,33333.

Met deze waarden kan nu de test locatie gescoord worden. Deze waarden worden afzonderlijk van elkaar als parameter gebruikt in de ScoreInt( ) methode.

Voor MaxParkeerPlaatsen is de score 0,9814.
Voor MaxPersonen is de score 0,0407.

Op deze manier wordt voor elk integer veld in een locatie een score bepaald. Uiteindelijk worden al deze scores opgeteld.

10.2.3.3. Scoren van Varchar velden

Voor het berekenen van een score voor de Varchar velden kan de methode ScoreBool( ) hergebruikt worden. Hieronder is het deel van het profiel met twee Varchar velden getoond.

LocationId	LocationName	LocationSoort	LocationBereikbaarheid
133728	Studio's Aalsmeer	139361,139379,139380,166,78,81,79,170	140234,166993,166994,166995
134092	VVC Adventure	70,78	140234
134198	BoatHouse	139361,82	140234,166993,166995,166996

Afbeelding 23: Opgebouwd profiel van de gebruiker, een deel van de velden is weergegeven

Er is te zien dat in deze velden CSV waarden opgeslagen staan. Deze Varchar velden kunnen gezien worden als Boolean[ ], als een ID van een Picklist item voorkomt in de string, dan staat deze waarde op true. Indien het niet voorkomt staat deze waarde op false. Welke waarden er voor kunnen komen in LocatieSoort is te zien in de afbeelding van de Picklist van LocatieSoort hiervoor getoond in het hoofdstuk. Met deze gegevens kan er per item in de CSV string de ScoreBool( ) methode uitgevoerd worden.

Indien er alleen een score berekend zou worden voor de items die voorkomen in de CSV string, dan zouden er incorrecte scores ontstaan. Dit komt door het feit dat er dan alleen scores berekend worden voor de aanwezige waarden, wat betekent dat een locatie met meer Locatie soorten gekoppeld ook vaak een hogere score zal krijgen. Doordat alle waarden gescoord worden, ook degene die niet voorkomen in de Locatie soort CSV string, ontstaat er een correcte score.

10.2.3.4. Gewichten toekennen

Een uitbreiding op deze manier van locaties scoren is het toekennen van gewichten aan bepaalde velden. Als bepaalde metadata velden veel belangrijker zijn dan anderen, dan kunnen de scores van deze velden met een bepaalde factor vermenigvuldigd worden. Op deze manier zou de klant aan kunnen geven wat zij de meest belangrijke metadata vinden en een locatie hierop ook meer laten scoren.

Naast het geven van gewichten aan metadata velden is er de mogelijkheid om de tijd een rol te laten spelen in het scoren van locaties. De datum en tijd van het openen van ene locatie wordt namelijk ook opgeslagen in de profieldata. Als een gebruiker een maand geleden een aantal locaties heeft bekeken zullen deze waarschijnlijk van minder belang zijn in het opbouwen van het profiel dan de locaties die onlangs bekeken zijn. Een uitbreiding op het algoritme zou dan kunnen zijn om profieldata ouder dan een maand niet mee te nemen bij het scoren of dit te delen door een bepaalde factor.

10.2.3.5. Eindscore

Aan het eind van het algoritme worden de scores van alle velden bij elkaar opgeteld en tot een totaalscore samengevoegd per locatie. Deze score wordt samen met de locatie opgeslagen in een Dictionary<LocationResult, double>. Deze Dictionary wordt dan gesorteerd aflopend op score, door middel van het uitvoeren van een LINQ statement.

Alle items uit de Dictionary worden hierna met behulp van een ListViewController uit het framework getoond in de app. Om de personalisatie inzichtelijk te maken wordt er in een dialog getoond aan de gebruiker hoe de personalisatie in zijn werk gaat. Er wordt dan tekstueel getoond welke metadata velden doorslaggevend zijn in het vinden van locatie suggesties.

Afbeelding 24: Locatie suggesties dialog

Op deze manier wordt op basis van tracking van de gebruiker locatie suggesties gemaakt.

10.2.4. Unittests

Om te testen of de ontwikkelde onderdelen van deze sprint goed werken zijn er een aantal unittests opgesteld. Dit is gedaan in een Android Unittest Application. Dit is gedaan voor de volgende onderdelen:

- Invoeren tracking data voor het opbouwen van het profiel
- ScoreBool( ) methode
- ScoreInt( ) methode

De code die in deze unittests uitgevoerd wordt is identiek aan de code die in de app geïmplementeerd is.

De verwachtte waardes van de twee methodes zijn handmatig berekend. De unittests werden allemaal succesvol uitgevoerd.

10.2.5. Locaties opgehaald van webservice

Om de personalisatie mogelijk te maken zijn de locaties nodig die gescoord kunnen worden. Een vereiste is echter dat alle locaties beschikbaar moeten zijn op het moment van scoren en sorteren,

indien dit niet zo is kan er geen aflopende lijst opgesteld worden. Er kunnen namelijk nog locaties zijn die nog niet zijn opgehaald, maar die wel hoog zouden scoren voor de gebruiker.

Om deze reden worden alle beschikbare locaties van de webservice opgehaald. Dit kost echter veel tijd en dataverkeer. Het ophalen van deze locaties kan oplopen tot 30 seconden. Voor het huidige prototype app is dit geen probleem, omdat performance geen issue is. Echter als het product in productie genomen zal worden, dan zal dit moeten veranderen.

De oplossing hiervoor is al bedacht, echter kon dit niet tijdig uitgevoerd worden door de opdrachtgever. De oplossing is namelijk om de webservice van Locaties.nl uit te breiden zodat het gehele scoringsalgoritme serverside uitgevoerd zal worden op een ElasticSearch dataset die gerepliceerd is van de database. Op de database worden alle CRUD statements uitgevoerd en de ElasticSearch dataset wordt gebruikt om complexe queries uit te voeren.

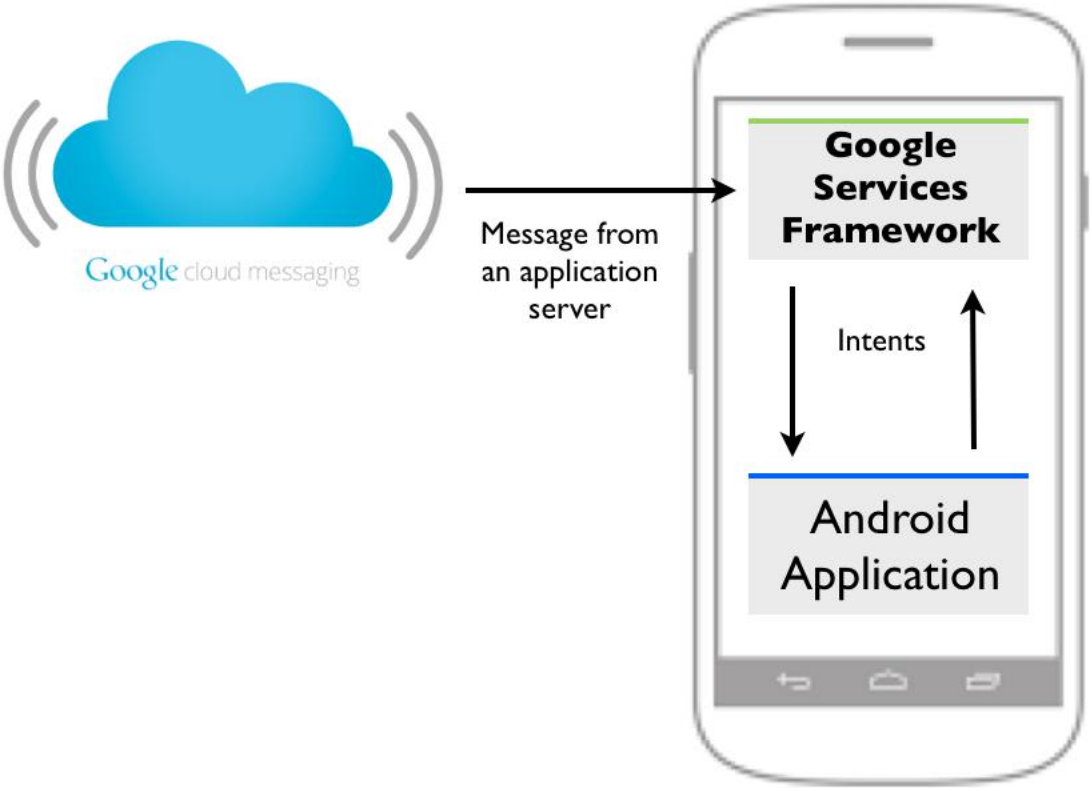
10.3. Persoonlijke notificaties

In deze sprint was er de behoefte om gebruikers persoonlijke notificaties te kunnen sturen. Bijvoorbeeld wanneer er nieuwe locaties aangemeld worden op Locaties.nl, die aan het profiel van de gebruiker voldoen. Om dit mogelijk te maken is er gekozen om Google Cloud Messaging te implementeren.

Google Cloud Messaging is een gratis service van Google. Deze service handelt het verzenden, routing en queuen van berichten van server applicaties af.

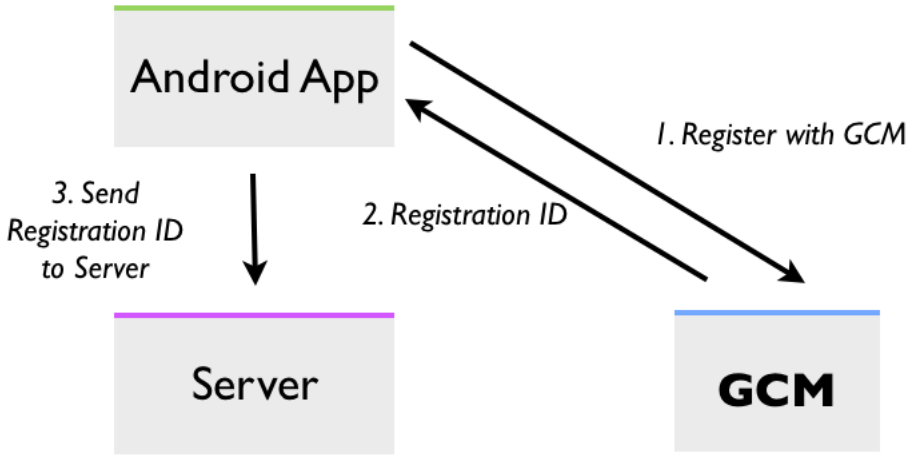
Om berichten te kunnen ontvangen moet het Google Services Framework geïnstalleerd zijn op het toestel. Dit wordt geïnstalleerd wanneer Google Play geïnstalleerd wordt op het toestel. Het Google Services Framework is een proces dat op de achtergrond draait en wacht op berichten van Google Cloud Messaging. Om deze berichten te kunnen ontvangen is er in de app een BroadcastReceiver geïmplementeerd. Deze BroadcastReceiver wacht op berichten van Google Cloud Messaging. Wanneer deze berichten binnenkomen zullen deze gedeserialiseerd worden naar Intents. Deze Intents worden dan gestuurd naar applicaties die hiervoor geregistreerd zijn. Dit wordt door de Google Cloud Messaging IntentService afgehandeld, welke geïmplementeerd is in de app.

Om berichten te kunnen versturen moet er een server opgezet worden. Voor deze sprint is er een standaard PHP server gebruikt die ontwikkeld is voor dit onderwerp, deze draait op de localhost. De server heeft een aantal verantwoordelijkheden. De server moet de registratie ID van de app kunnen opslaan en moet een bericht kunnen versturen naar de Google Cloud Messaging server.



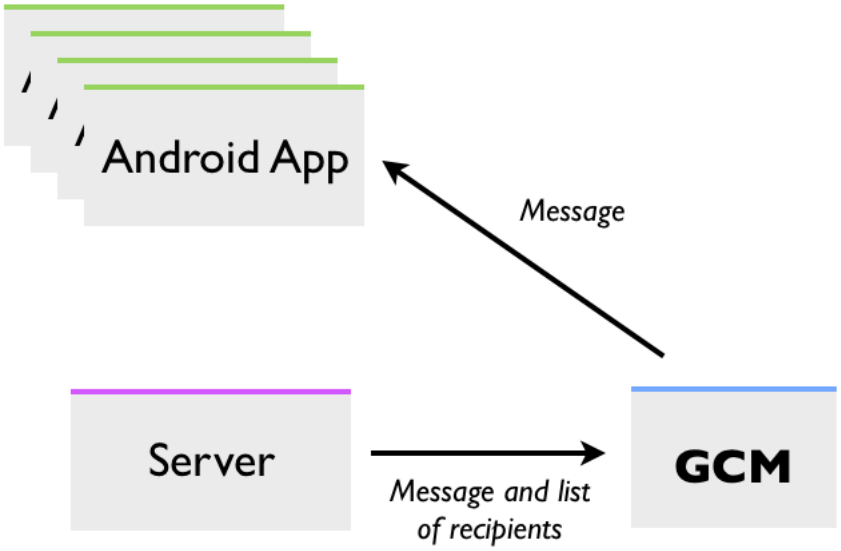
Afbeelding 25: Relaties tussen de verschillende GCM-componenten

Om te beginnen moet een applicatie zich registreren bij Google Cloud Messaging. Wanneer dit gebeurt, zal Google Cloud Messaging een registratie ID terugsturen specifiek voor die applicatie op dat specifieke device. De applicatie zal de registratie ID naar de server applicatie moeten sturen, zodat de server applicatie dit op kan slaan.



Afbeelding 26: Registratie proces van Google Cloud Messaging

Wanneer de registratie klaar is, kan de server beginnen met het versturen van berichten naar de applicatie. De server stuurt een bericht naar Google Cloud Messaging, dit bericht bevat de registratie ID's en het bericht wat verstuurd moet worden.



Afbeelding 27: Bericht versturen met Google Cloud Messaging

Als de server een bericht probeert te versturen naar een device wat offline is, dan zal dit bericht gequeued worden door Google Cloud Messaging. Het bericht wordt dan verstuurd wanneer het device weer online is.

10.4. Resultaat

Deze sprint is geslaagd. Er is personalisatie en tracking ingebouwd in de app die in de voorgaande sprints ontwikkeld was. Verder kunnen er persoonlijke notificaties verstuurd worden vanaf een lokale server. Deze notificaties worden door middel van Google Cloud Messaging verstuurd en hierna ontvangen in de app. De sprint is afgerond binnen de opgestelde planning.

11. High-level Vraagstukken

In dit hoofdstuk wordt antwoord gegeven op de vragen die zijn gesteld in het eerdere hoofdstuk High-level Vraagstukken.

11.1. Organisatie

- **Mobile app met samenwerking tussen sales, User Experience(UX), development.**

Ik kan dit vraagstuk slechts deels beantwoorden. Ik heb niet samengewerkt met de sales afdeling bij Liones. Ik heb wel samengewerkt met Arnout Kooij, die werkzaam was bij Liones als stagiair op de afdeling UX. Deze samenwerking vond plaats in sprint 2 van mijn afstudeeropdracht. Ik kan zeggen dat de samenwerking erg prettig verliep.

Er zijn een aantal raakvlakken tussen het development en UX onderdeel. Bij mijn afstudeeropdracht waren dat de flow door de app en de schermontwerpen. Deze onderdelen heb ik dan ook samen met Arnout Kooij gemaakt. Het was voor beide kanten erg handig om samen te werken aan de flow en de schermontwerpen. Er kon op deze manier direct overleg gepleegd worden. Verder werd er input en feedback geleverd op het gemaakte werk.

De producten die in deze samenwerking ontstaan zijn heb ik gebruikt bij het ontwikkelen van de app voor Locaties.nl.

- **Samenwerking testen c.q. ontwikkelen**

Tijdens het project is er op verschillende manieren getest. Er is een user test opgesteld door UX. Verder heb ik unittests opgesteld in sprint 5 voor het testen van het gebruikte algoritme.

Er zou verder een systeemtest of acceptatietest uitgevoerd kunnen worden.

11.2. Research & Development

- **Op welk moment in de Research & Development projecten betrekken we de klant erbij?**

De klant betrekken in een Research & Development project kan op meerdere momenten, afhankelijk van de intentie waarmee je de klant erbij betrekt. Indien het noodzakelijk is om input te krijgen van de klant is het handig om de klant in een vroeg stadium erbij te halen.

Indien het voor het project niet noodzakelijk is om de klant te betrekken, dan is het handig om de klant pas in een later stadium te betrekken wanneer er bijvoorbeeld een demo gegeven kan worden. Op deze manier kan de klant het project niet een bepaalde richting in duwen en sta je vrijer in de te maken beslissingen.

- **Hoe leggen we een profiel van de gebruiker vast in een mobile app?**

Om het profiel van een gebruiker goed op te kunnen bouwen is het eerst noodzakelijk om te analyseren welke data er benodigd is. Een ander belangrijk aspect is om uit te zoeken wanneer je deze data zal vastleggen, dat wil zeggen op welke punten in de app.

Voor dit project werden de gehele locaties opgeslagen in het profiel. Dit had als reden dat er op de meeste data gezocht kon worden met de webservices.

Wat betreft het bepalen wanneer het profiel vastgelegd wordt kan ik zeggen dat hiervoor twee goede plekken zijn in de app.

Als eerste wanneer de gebruiker een locatie opent in de app. Wanneer hij dit doet blijkt het dat er enige interesse is voor deze bepaalde locatie. Dit is een goede reden om de locatie tot het profiel toe te voegen.

Een andere plek om het profiel op te bouwen is bij het uitgebreid zoeken. Hier toont de gebruiker zijn interesse door op informatie te zoeken die hij belangrijk vind in een locatie.

- **Hoe moeten de serverside services eruitzien?**

De serverside services zullen uitgebreid moeten worden om personalisatie & tracking mogelijk te maken met een goede performance. Op dit moment worden alle locaties ingeladen en wordt er lokaal een algoritme uitgevoerd. Dit is erg inefficiënt en kost veel bandbreedte.

De webservice van Locaties.nl zal dus uitgebreid moeten worden. Het moet mogelijk worden om een locaties toe te voegen aan het profiel van een gebruiker door middel van de webservice. Hiernaast moet het mogelijk zijn om een op basis van het opgebouwde profiel een set van matchende locaties terug te geven aan de gebruiker. Hierbij kan gebruik gemaakt worden van de bestaande webservice. Op de server zal het personalisatie algoritme uitgevoerd moeten worden.

11.3. Techniek

- **Hoe gaan we om met crossplatform?**

Bij Liones was er de behoefte om crossplatform apps te kunnen ontwikkelen. Dit had als reden dat er niet voor elk Operating System een team opgezet kan worden. Daarom werd er gezocht naar een bruikbare oplossing. Deze lijkt gevonden te zijn in het product van Xamarin. Hiermee kan er crossplatform ontwikkeld worden en er kan veel code hergebruikt worden.

Door gebruikt te maken van Xamarin kan er in C# ontwikkeld worden voor alle platformen. Een voordeel hiervan is dat je niet meer drie talen hoeft te kennen om apps te ontwikkelen voor Android(Java), iOS(Objective C) en Windows Phone(C#). Er blijft natuurlijk wel platform specifieke kennis benodigd.

In dit project zijn er twee crossplatform projecten gemaakt. Als eerste de Algemene Shared Core, met hierin de structuur voor het framework. Als tweede de Locaties.nl Shared Core met hierin alle business logica voor deze klant. Deze twee projecten zijn gebruikt bij het ontwikkelen van de Android app. Verder kunnen ze gebruikt worden bij het ontwikkelen van de iOS en Windows Phone versies voor Locaties.nl.

- **Hoe gaan we om met hergebruik?**

In dit project wordt er op drie manieren code hergebruikt.
Dat zijn de volgende onderdelen:

- Algemene Shared Core
- Android Shared Core
- Locaties.nl Shared Core

In de Algemene Shared Core is er een structuur ontwikkeld voor het framework, deze structuur wordt in elke Operating System Shared Core aangehouden.

De Android Shared Core bevat de implementatie van de ontworpen ViewControllers. Deze ViewControllers kunnen in elk nieuw project hergebruikt worden. Op deze manier wordt deze code hergebruikt.

De Locaties.nl Shared Core bevat de business logica voor Locaties.nl. Dit zijn alle webservices die beschikbaar zijn voor Locaties.nl. Deze Shared Core kan hergebruikt worden bij het ontwikkelen van de Locaties.nl apps voor de afzonderlijke platformen.

- **Hoe houd je ontwikkeling eenvoudig?**

Doordat er met Xamarin gewerkt wordt, wordt er op elk platform ontwikkeld in C#. Dit zorgt ervoor dat developers niet drie afzonderlijke programmeertalen hoeven te beheersen voor het ontwikkelen van apps.

Om ontwikkeling van reader apps eenvoudig te houden wordt er gebruik gemaakt van het ontworpen en ontwikkelde framework. Dit zorgt ervoor dat developers niet meer voor elke lijst een nieuwe implementatie hoeven maken hiervan. Ze kunnen met behulp van het framework een lijst aanmaken.

- **Welke libraries kunnen we gebruiken?**

Voor veel development uitdagingen zijn al oplossingen gevonden door andere developers. Xamarin heeft hier goed op ingespeeld met de Xamarin Component Store. Hier kunnen allerlei components toegevoegd worden door developers. Deze componenten kunnen gedownload worden door andere developers die deze functionaliteit willen gebruiken.

De gedownloade components worden direct toegevoegd aan het project en kunnen gebruikt worden. Dit kan gezien worden als het importeren en gebruiken van libraries bij het native developen van Android apps.

In dit project is er gebruik gemaakt van de volgende components:

- Google Play Services
- Xamarin.Mobile
- JSON.Net
- SQLite.Net
- Android Support Library v4

Google Play Services is een component die de Google Play Services API's beschikbaar stelt. In dit project is het gebruikt voor Google Maps en Google Cloud Messaging.

Xamarin.Mobile is een component die de standaard API functionaliteiten beschikbaar stelt van Android, iOS en Windows Phone. In dit project is deze component gebruikt om met GPS te werken.

JSON.Net is een JSON framework voor .Net. In dit project is het gebruikt om serialisatie en deserialisatie van objecten mogelijk te maken.

SQLite.Net is een crossplatform SQLite client en ORM. Deze component is gebruikt voor het aanmaken en gebruiken van een SQLite database voor het Personalisatie & Tracking onderdeel.

De Android Support Library v4 component levert backward-compatible versies van Android framework API's. Deze component is gebruikt om te kunnen werken met de ViewPager en Fragments.

- **Is Xamarin geschikt voor ons?**

Met de uitvoering van dit Research & Development project wilde Liones achterhalen of Xamarin geschikt was voor crossplatform app development.

De reden dat Liones gekozen had voor Xamarin was dat zij niet de capaciteit hebben om een team op te zetten per Mobile Operating System. Xamarin speelt hier goed op in door de mogelijkheid te bieden om veel code te hergebruiken voor Android, iOS en Windows Phone.

Met de ervaring die tijdens het project heb opgedaan in Xamarin, kan ik een uitspraak doen over de geschiktheid van Xamarin. Ik vind Xamarin geschikt als crossplatform app development product voor Liones. Deze uitspraak is echter alleen gebaseerd op het ontwikkelen van een Android app en de shared core. Als het developen van apps voor iOS en Windows Phone op enigszins dezelfde manier zal verlopen, dan kan ik zeggen dat Xamarin in zijn geheel geschikt is voor Liones als crossplatform development product.

Er zitten echter wel een aantal haken en ogen aan, zoals bij elk nieuw product. Xamarin is op een aantal onderdelen afwijkend van de standaard Android SDK. Een aantal classes en methodes zitten op andere plekken dan verwacht of ontbreken. Dit zorgt soms voor wat tragere development, maar is niet onoverkomelijk.

Verder is de documentatie van Xamarin nog niet volledig. Voor sommige onderwerpen bestaat er nog geen documentatie en andere onderwerpen hebben onvolledige informatie. Indien er geen documentatie aanwezig is voor bepaalde onderdelen, dan kan de standaard Android documentatie bekeken worden. De oplossingen van de standaard Android documentatie kunnen vaak herschreven worden naar C#.

11.4. Functionele requirements

- **Lijst van content items weergeven**

Het maken van een lijst in Android is een van de lastigere onderdelen van Android development. Hiernaast is het aanmaken van de lijst in het framework opgenomen, wat de complexiteit verhoogt.

De plaatjes in de lijst worden asynchroon ingeladen, dit verhoogt de complexiteit verder.

Ik denk dat het opnemen van de lijst functionaliteit in het framework innovatief is, er zijn geen voorbeelden van dit concept te vinden.

Ik denk dat dit onderdeel van het framework het meest leerzame deel was om te ontwikkelen. Er kwamen veel complexe constructies aan bod.

- **Support voor categorieën**

Bij Liones willen ze support voor categorieën hebben in de te ontwikkelen apps. Dit is mogelijk door een lijst of grid aan te maken waarin de categorieën geplaatst worden met een plaatje erbij. Dit is niet complex om te gebruiken. De complexiteit ligt in de implementaties van de lijst en grid in het framework.

- **Mogelijkheid tot inloggen**

Dit onderdeel was eenvoudig te maken en bevatte weinig complexe onderdelen. Om in te kunnen loggen moest de webservice aangesproken worden.

- **Detail pagina van een item tonen**

Het aanmaken van een detail pagina in de app is niet complex. De implementatie van de detail pagina in het framework is ook niet complex. Deze implementatie is afgeleid van de andere ViewControllers die reeds gedefinieerd waren in het framework.

- **Metadata tags tonen van een item**

Het aanmaken van een metadata pagina in de app is niet complex. De implementatie hiervan is ook niet complex. Deze is tevens afgeleid van de andere ViewControllers die reeds gedefinieerd waren in het framework.

12. Reflectie

12.1. Productevaluatie

12.1.1. Sprint 1: Framework basis

Het beoogde resultaat van deze sprint was het opzetten van de framework basis voor Android. Met dit framework moeten er reader apps ontwikkeld kunnen worden. De ViewControllers die ontwikkeld zijn in deze sprint maken het mogelijk om de volgende onderdelen aan te maken in een reader app:

- Lijst van items
- Mogelijkheid om te swipen naar ander scherm
- Images in een grid
- Detail pagina
- Detail pagina met metadata
- Mogelijkheid om images in het groot te bekijken

Al deze ViewControllers bevatten hun eigen logica. Doordat de ViewControllers de benodigde logica bevatten, wordt hergebruik van code goed toegepast. Het is namelijk niet nodig om in elke Activity de logica opnieuw te schrijven, de ViewControllers in het framework bevatten deze logica.

Mijn bedrijfsmentor was positief over het eindresultaat van deze sprint. Het voldoet aan de wensen en eisen van de opdrachtgever.

Met het framework kunnen er reader apps ontwikkeld gaan worden voor het Android platform. Hierbij wordt gebruikt gemaakt van de ontwikkelde ViewControllers. Dit framework is door mij gebruikt in de sprints die hierop volgden.

Het framework kan door Liones uitgebreid worden indien de behoefte naar nieuwe ViewControllers ontstaat. Dit komt door de toepassing van het Abstract Factory Pattern in deze sprint. Er kunnen nieuwe ViewControllers ontworpen en ontwikkeld worden. Deze kunnen dan toegevoegd worden aan het bestaande framework.

12.1.2. Sprint 2: App flow Locaties.nl

Het beoogde resultaat van deze sprint was de flow door de app opzetten. Dit houdt in dat er genavigeerd moet kunnen worden tussen de verschillende Activities en door middel van de navigation drawer. Het opbouwen van deze app flow zou gedaan worden met behulp van het ontwikkelde framework uit sprint 1.

In deze sprint waren er twee fases, de design fase en de bouw fase. In de design fase heb ik samen met de UX designer stagiair Arnout Kooij het ontwerp gemaakt voor de verschillende schermen en voor het ontwerp van de flow door de app.

De schermontwerpen hadden als doel het design over te brengen richting de opdrachtgever en om het voor mij inzichtelijk te maken hoe de activities gebouwd zouden worden in de hierop volgende sprints.

De flow door de app was uitgewerkt in een sitemap. Deze sitemap heeft veel nut gehad tijdens de bouw fase, omdat de navigatie door de app op deze manier in kaart is gebracht.

In de bouw fase van deze sprint is er een app gemaakt waarmee er genavigeerd kon worden tussen de verschillende activities die de app zou gaan bevatten. De implementatie van deze afzonderlijke activities is minimaal in deze sprint.

Mijn bedrijfsmentor was positief over het eindresultaat van deze sprint. Het voldoet aan de wensen en eisen van de opdrachtgever.

De app die in deze sprint ontwikkeld is heeft als basis gediend voor sprint 3 en 4, waarin de activities een implementatie kregen.

12.1.3. Sprint 3: Zoek & detail schermen Locaties.nl

Het beoogde resultaat van deze sprint was het uitwerken van de afzonderlijke activities die opgesteld zijn op basis van de user stories in sprint 3. In sprint 2 was er een app ontwikkeld die de flow van applicatie bevatte. Dit is als uitgangspunt gebruikt voor deze sprint.

Ik heb de user stories van deze sprint uitgewerkt per activity op basis van prioriteit. De volgende functionaliteiten zijn geïmplementeerd in deze sprint:

- Google Maps
- Splashscreen
- Zoeken & uitgebreid zoeken van locaties
- Locatie overzicht in een lijst
- Detail pagina van locatie
- Settingsscherm
- Locatie tonen op de kaart
- Foto galerij
- Thema instelbaar

Mijn bedrijfsmentor was positief over het eindresultaat van deze sprint. Het voldoet aan de wensen en eisen van de opdrachtgever.

De app die ontwikkeld is in deze sprint is een basis versie voor de Locaties.nl app. Het bevat de belangrijkste functionaliteiten van Locaties.nl. De app is als uitgangspunt gebruikt voor sprint 4.

12.1.4. Sprint 4: Overige schermen Locaties.nl

Het beoogde resultaat van deze sprint was het uitwerken van de afzonderlijke activities die opgesteld zijn op basis van de user stories in sprint 4. In sprint 3 waren de belangrijkste zoek & detail schermen voor de Locaties.nl app ontwikkeld. Deze app werd als uitgangspunt genomen voor deze sprint.

Ik heb de user stories van deze sprint uitgewerkt per activity op basis van prioriteit. De volgende functionaliteiten zijn geïmplementeerd in deze sprint:

- Login
- Registreren
- Profiel van de gebruiker met favorieten
- Favoriete locaties beheren
- Top 10 locaties
- Locatie advies aanvraag

Mijn bedrijfsmentor was positief over het eindresultaat van deze sprint. Het voldoet aan de wensen en eisen van de opdrachtgever.

Het resultaat van deze sprint is een app waarin alle activities zijn uitgewerkt. De app is als uitgangspunt gebruikt voor sprint 5.

12.1.5. Sprint 5: Personalisatie & Tracking

Het beoogde resultaat van deze sprint was het ontwerpen en ontwikkelen van personalisatie en tracking voor de Locaties.nl app. Het gedrag van de gebruiker moet bijgehouden worden en op basis hiervan moeten er persoonlijke locatie suggesties gegeven worden op basis van dit gedrag. Deze locaties worden op twee manieren aan de gebruiker getoond. De eerste manier is het tonen van deze locaties in een gesorteerde lijst. De sortering is op basis van het opgebouwde profiel gemaakt. De tweede manier is het sturen van een notificatie aan de gebruiker met behulp van Google Cloud Messaging.

Om het gedrag van de gebruiker bij te houden is er een SQLite database aangemaakt. Hierin worden alle locaties die de gebruiker bekijkt opgeslagen. Op deze manier wordt het profiel opgebouwd. De profielopbouw voldoet aan alle wensen en eisen.

Om de persoonlijke locatie suggesties te geven aan de gebruiker is er een algoritme ontwikkeld. In dit algoritme wordt het opgebouwde profiel gebruikt om voor alle afzonderlijke locaties een score te berekenen. Hier wordt gebruik gemaakt van de Bell-curve.

De persoonlijke notificaties worden verstuurd via Google Cloud Messaging. Hiervoor is een lokale server opgezet met database. In de app zijn de benodigde klasses geïmplementeerd om Google Cloud Messaging berichten te kunnen ontvangen.

Mijn bedrijfsmentor was positief over het eindresultaat van deze sprint. Het voldoet aan de wensen en eisen van de opdrachtgever.

Liones kan het personalisatie & tracking onderdeel door ontwikkelen. Hiervoor kunnen ze gewichten gaan toevoegen aan de verschillende velden in de tracking tabel. Op deze manier kunnen ze velden belangrijker maken. Deze wegen dan zwaarder mee in het score algoritme. Deze gewichten kunnen naar eigen inzicht of in overleg met de klant bepaald worden.

Verder kan Liones het score algoritme implementeren op de webservice van Locaties.nl. Op deze manier zal de performance drastisch verbeteren. De app hoeft dan niet tegelijkertijd alle beschikbare locaties te downloaden en dan te scoren. In de webservice zal dan alleen gescoord te hoeven worden op basis van het opgebouwde profiel.

Het opgebouwde profiel wordt op dit moment lokaal opgeslagen op het toestel waar de app is geïnstalleerd. Liones zou de webservice uit kunnen breiden en het mogelijk maken om het opgebouwde profiel op te slaan op de server. Dit zou gekoppeld kunnen worden met het bestaande profiel van de gebruiker. Dit zal de performance ten goede komen.

12.2. Procesevaluatie

Tijdens mijn afstudeerstage heb ik veel zelfstandig gewerkt. Ik was de enige developer die zich constant bezighield met het ontwerpen en ontwikkelen van de app en het framework. Op sommige momenten heb ik echter wel hulp gehad van mijn bedrijfsmentor. Dit was bij onderdelen waarbij ik soms niet in de goede richting zat voor de juiste oplossing. Mijn bedrijfsmentor probeerde mij dan altijd in de goede richting te sturen, zonder het antwoord te geven. Dit vond ik leerzamer dan wanneer ik een antwoord zou krijgen zonder hier over na te hoeven denken.

Tijdens de oriëntatie heb ik de user stories opgesteld voor het project. Deze user heb ik hierna geprioriteerd in samenwerking met mijn bedrijfsmentor. Hierna zijn de user stories in verschillende sprints verdeeld.

In een sprint hanteerde ik altijd dezelfde werkwijze. Voordat ik een sprint begon waren de user stories bekend. Indien een user story te groot was werd deze opgesplitst in meerdere user stories.

Nadat de user stories bekend waren in een sprint begon ik met het ontwerpen van de benodigde functionaliteiten indien dit nodig was. Nadat de ontwerpen klaar waren begon ik met het ontwikkelen.

Ik vond het werken met Scrum erg prettig. Ik kon hierdoor goed inspelen op veranderingen indien deze optraden. Ook had ik na elke sprint een werkend product wat ik aan de opdrachtgever kon tonen.

Omdat ik over het algemeen in mijn eentje werkte in dit project, kon ik geen dagelijkse standup houden. Ik probeerde echter wel de volgende vragen te beantwoorden voor mezelf:

- Wat heb je gisteren gedaan?
- Wat ga je vandaag doen?
- Zijn er ergens problemen ontstaan?

De combinatie van Research & Development project en Scrum vond ik goed werken. In een Research & Development project kunnen er vaak veranderingen optreden. Ook zijn de user stories niet altijd compleet of benodigd. Met een flexibele ontwikkelmethode als Scrum kan hier perfect op ingespeeld worden.

Het werken met C# in Xamarin was nieuw voor mij. Ik heb door de uitvoering van dit project C# en Xamarin goed onder de knie gekregen. Deze twee onderwerpen heb ik vooral in de oriëntatie behandeld, zodat ik in het project direct aan de slag kon gaan. Ik vond dat het leren van C# en Xamarin mij redelijk gemakkelijk afgaan.

12.2.1. Sprint 1: Framework basis

Ik ben deze sprint begonnen met het analyseren van de architectuur die Xamarin gebruikt. Nadat ik hier een goed beeld van had ben ik mijn eigen ontwerp gaan maken voor het framework. Hiervoor heb ik uitgezocht welke Design Patterns ik zou kunnen toepassen in het framework.

Nadat ik een ontwerp had gemaakt voor het framework heb ik dit besproken met mijn bedrijfsmentor. Hierbij hebben we goed discussie gevoerd over bepaalde aspecten van het ontwerp. Hierbij zijn we tot nieuwe ideeën gekomen, deze zijn weer doorgevoerd in het ontwerp.

Nadat er een definitief ontwerp was gemaakt ben ik begonnen met het ontwikkelen van het framework. Ik heb hierbij met complexe onderwerpen gewerkt. Bijvoorbeeld het implementeren van een lijst in het framework, de logica van een lijst implementeren, het asynchroon inladen van plaatjes en het toepassen van het Abstract Factory Pattern in het framework.

Ik kon tijdens deze sprint veel feedback vragen aan mijn bedrijfsmentor. Hierdoor kon ik sneller tot een goed ontwerp en een goede implementatie komen.

Ik heb tijdens deze sprint weinig problemen ondervonden. Echter het toevoegen van een MapViewController in het framework is mij niet gelukt. Dit kwam doordat er veel complicaties optraden bij het ontwikkelen hiervan. Later bleek ook dat er tegenstrijdige swipe gestures

ontstonden indien deze ViewController geïmplementeerd zou worden. Daarom is er besloten om dit niet te implementeren.

Ik zou bij een vergelijkbaar onderwerp het de volgende keer niet anders aanpakken. Ik vond de manier van werken erg prettig. Vooral het overleg met mijn bedrijfsmentor na het ontwerpen van het framework vond ik erg handig. Hierdoor ontstond er een duidelijk beeld van het framework voordat er met de ontwikkeling hiervan begonnen werd. Bij de ontwikkeling van het framework kon ik al gelijk profiteren van de ontworpen structuur. Nadat de eerste ViewControllers ontwikkeld waren, werd het ook makkelijker om de andere ViewControllers te ontwikkelen. Dit kwam doordat elke ViewController aan dezelfde structuur voldoet.

Ik heb veel geleerd tijdens deze sprint. Met name over de architectuur van een systeem en hoe dit goed ontworpen kan worden, zodat dit een goede crossplatform oplossing geeft. Ik heb een veel beter inzicht verkregen in het toepassen en implementeren van complexe design patterns, met name het Abstract Factory Pattern en het Strategy Pattern. Verder heb ik de codeerstandaarden en naamconventies aangehouden voor het programmeren in C#. Ik heb interfaces gedefinieerd en geïmplementeerd in het framework, respectievelijk in de Algemene Shared Core en de Android Shared Core. Deze interfaces kunnen op elk platform geïmplementeerd worden. Deze sprint was erg leerzaam.

12.2.2. Sprint 2: App flow Locaties.nl

Ik ben deze sprint begonnen met de design fase. Hierbij werkte ik nauw samen met Arnout Kooij(UX). We zijn begonnen met het ontwerpen van de flow van de app op basis van de analyse naar navigatie binnen reader apps en de analyse op de website van Locaties.nl. We stonden vrij in het ontwerpen van de flow, dit had als reden dat Liones een verse blik wilde op dit onderwerp.

Nadat de flow opgezet was, zijn we begonnen met het ontwerpen van de schermen voor de app. Hierbij is er veel overleg gepleegd. Dit was ook nodig, omdat we beide veel ideeën hadden over de schermenontwerpen.

Op basis van de flow en schermontwerpen zijn hierna user tests afgenomen bij verschillende werknemers van Liones. Hierbij werden de gebruikte iconen, de flow en de User Interface getest. Hieruit hebben we veel feedback ontvangen en deze feedback is verwerkt in de betreffende onderwerpen.

Nadat de flow en de schermontwerpen af waren heb ik de flow uitgewerkt in een app. Deze app had een minimale implementatie voor Locaties.nl, het doel was dan ook het mogelijk maken om te navigeren door de app.

Doordat we geen beperking opgelegd hebben gekregen konden we vrij ontwerpen. Dit vond ik een prettige manier van werken.

Ik zou deze werkzaamheden een volgende keer niet anders aanpakken. We hebben bij het ontwerpen van de flow en de schermontwerpen geen problemen ondervonden. Ik vond de samenwerking met Arnout Kooij erg prettig verlopen.

Ik heb veel geleerd van de samenwerking met Arnout Kooij. Ik had nog nooit nauw samengewerkt met iemand van zijn vakgebied. Ik heb de samenwerking als zeer prettig ervaren. We hebben een goede flow door de app opgezet. Hiernaast zijn er ook goede schermontwerpen gemaakt, welke ik bij de ontwikkeling van de app kon gebruiken.

12.2.3. Sprint 3: Zoek & detail schermen Locaties.nl

Ik ben deze sprint begonnen met het maken van de implementatie van de Locaties.nl Shared Core. Hierin heb ik de methodes ontwikkeld die de webservice van Locaties.nl aanspreken om zo de benodigde data beschikbaar te hebben in de app.

Nadat de webservice gebruikt kon worden ben ik begonnen met het ontwikkelen van de activities die beschreven staan in sprint 3. Ik heb de activities ontwikkeld op basis van de schermontwerpen uit sprint 2. De functionaliteiten zijn gebaseerd op de user stories van de backlog in sprint 3.

Ik vond deze sprint erg leerzaam. Ik heb veel geavanceerde technieken gebruikt van Android. Onder andere de volgende onderwerpen:

- Asynchroon inladen van plaatjes
- Navigation drawer
- Shared Preferences
- Webservice aanspreken
- Google Maps
- GPS
- Geavanceerde XML structuren voor layout
- Toepassing van JSON

Ik heb mijzelf ingelezen op deze onderwerpen indien dit nodig was. Hierbij volgde ik tutorials als deze aanwezig waren.

De Google Maps implementatie bleek complexer te zijn dan ik van tevoren dacht. Dit kwam doordat de Xamarin tutorials niet van hetzelfde detail niveau waren als de standaard Android tutorials in Java. Dit heeft mij uiteindelijk meer tijd gekost dan ik van tevoren verwachtte.

Ik zou deze sprint in een ander project op dezelfde manier aanpakken. Ik vond de manier van werken erg prettig en efficiënt. Het was nuttig om de schermontwerpen en de flow van de app al ontworpen te hebben zodat er in deze sprint gefocust kon worden op het bouwen van de verschillende functionaliteiten in de activities.

12.2.4. Sprint 4: Overige schermen Locaties.nl

Ik ben deze sprint begonnen met het ontwikkelen van de verschillende activities op basis van de schermontwerpen en flow uit sprint 2. De functionaliteiten zijn gebaseerd op de user stories van de backlog in sprint 4.

In deze sprint heb ik geen nieuwe technieken geleerd. De benodigde informatie had ik al opgedaan in de sprints hiervoor. In deze sprint zijn de benodigde functionaliteiten uitgewerkt in de app.

Ik zou dit in een ander project op dezelfde manier aanpakken. Deze sprint verliep zonder problemen. Het was nuttig om de schermontwerpen en de flow van de app al ontworpen te hebben zodat er in deze sprint gefocust kon worden op het bouwen van de verschillende functionaliteiten in de activities.

12.2.5. Sprint 5: Personalisatie & Tracking

Ik ben deze sprint begonnen met het ontwerpen van het tracking gedeelte van deze sprint. Bij het ontwerpen hiervan dacht ik tevens na over het personalisatie onderdeel van deze sprint. Deze twee onderwerpen moesten namelijk op elkaar afgestemd zijn zodat het tracking gedeelte de input kon leveren aan de personalisatie.

Het ontwerp voor het tracking onderdeel vond ik simpeler dan ik van tevoren had verwacht. Het ontwerp voor het bijhouden van het profiel van de gebruiker besloeg niet meer dan simpelweg de bekeken locaties opslaan in een tabel. Op deze manier werd alle benodigde informatie verzameld. De structuur van deze tabel is identiek als het locatie object wat er vanaf de webservice binnenkomt. Voor dit onderdeel is er gebruik gemaakt van een SQLite database die crossplatform gebruikt kan worden, met behulp van de SQLite.Net component. Deze SQLite.Net component maakt het tevens mogelijk om Object-Relational Mapping(ORM) toe te passen. Door hier gebruik van te maken is dit onderdeel van de sprint aanzienlijk makkelijker geworden. De standaard CRUD statements kunnen met behulp van de component uitgevoerd worden.

Nadat het tracking onderdeel ontworpen was ben ik begonnen met het ontwerp voor de personalisatie in de app. Hiervoor heb ik een algoritme ontworpen waarin alle locaties een score krijgen op basis van het opgebouwde profiel. Dit onderdeel vond ik complex maar tevens erg leerzaam. In dit onderdeel heb ik veel informatie gezocht over personalisatie en hoe dit door andere partijen opgelost werd. Hierbij kwam ik de toepassing van de Bell-curve tegen. Dit onderdeel heeft mij uiteindelijk de meeste tijd gekost in deze sprint.

Om persoonlijke notificaties te kunnen sturen heb ik gebruik gemaakt van Google Cloud Messaging. Hier was vanuit Xamarin geen goede documentatie aanwezig, de Android documentatie was wel van een goed niveau. Door deze twee te combineren ben ik uiteindelijk tot een werkende oplossing gekomen. Dit bleek langer te duren dan ik had ingeschat, doordat ik van beide documentaties een werkende oplossing moest maken.

Indien ik deze sprint nogmaals zou moeten maken dan zou ik dit voor de tracking en het Google Cloud Messaging op dezelfde manier aanpakken en uitvoeren.

Voor het personalisatie algoritme zou ik een andere keuze maken en dit tevens meer uitwerken. Een aanpassing die ik zou willen maken is om het ontworpen score algoritme te implementeren in de webservice. Dit heeft als reden dat de locaties dan niet eerst gedownload hoeven te worden voordat ze gescoord kunnen worden. Tevens zou ik het opgebouwde profiel ook op de webservice willen implementeren. Als deze twee onderdelen beschikbaar zijn op de webservice van Locaties.nl, dan zou dit een grote boost geven aan de performance en schaalbaarheid. Verder zou ik in het algoritme gewichten toekennen aan de verschillende velden. Op deze manier kan de klant zelf bepalen welke onderdelen zij belangrijk vinden bij het geven van persoonlijke suggesties aan de gebruiker.

12.3. Beroepstaken evaluatie

Voor de afstudeeropdracht is aangegeven welke beroepstaken er uitgevoerd zullen worden en op welk niveau dit gedaan wordt. Het bepalen van het behaalde niveau is gedaan aan de hand van de tabel hieronder.

Context		Taakrol		
		Geleid	zelfstandig	Sturend
	simpel	1	2	3
	lastig	2	3	4
	complex	3	4	5

1.4 Uitvoeren analyse door definitie van requirements

Deze beroepstaak heb ik zelfstandig uitgevoerd met de context lastig. De definitie voor deze beroepstaak is hiervoor als volgt.

"Het betreft één applicatie met een groot aantal requirements. Er is sprake van een beperkt aantal stakeholders met tegengestelde wensen en eisen t.o.v. de applicatie. Er is één set aan requirements, die gedurende de analyse kan veranderen."

Deze beroepstaak heb ik uitgevoerd op niveau 3. In het afstudeerplan staat dat deze beroepstaak zal worden uitgevoerd op niveau 3.

Ik ben begonnen met het achterhalen van de user stories door middel van gesprekken met de opdrachtgever.

Ik heb de user stories ingedeeld in verschillende sprints op basis van functionaliteit.

Ik heb de user stories geprioriteerd op basis van complexiteit, prioriteit en tijdsduur. Afhankelijk van de combinatie van deze factoren werden de user stories hoog of laag geplaatst op de sprint backlog. Afhankelijk van de plaats op de sprint backlog werden de user stories opgepakt en afgehandeld.

2.2 Ontwerpen, bouwen en bevragen van een database

Deze beroepstaak heb ik zelfstandig uitgevoerd met de context simpel. De definitie voor deze beroepstaak is hiervoor als volgt.

"Het betreft het ontwerpen, bouwen en bevragen van een database voor slechts één gebruiker. De vragen die gesteld worden op de database zijn slechts op één tabel."

Deze beroepstaak heb ik uitgevoerd op niveau 2. In het afstudeerplan staat dat deze beroepstaak zal worden uitgevoerd op niveau 3.

De reden waardoor het uitgevoerd is op niveau 2 is als volgt. In sprint 5: Personalisatie & Tracking was er geen behoefte was aan een complexe databasestructuur. Het volstond om een enkele tabel te ontwerpen op basis van de locatie objecten die gebruikt worden in de app. Deze tabel slaat de locatie objecten op die de gebruiker bekijkt in de app.

Doordat de benodigde data in sprint 5: Personalisatie & Tracking eenvoudiger bleek dan gedacht waren er ook geen complexe SQL-queries nodig. Met behulp van eenvoudige SQL-queries wordt de data uit deze tabel opgevraagd. Dit zijn queries om een gemiddelde of een telling van een veld op te vragen.

3.1 Ontwerpen softwarearchitectuur
Deze beroepstaak was in de oorspronkelijke opdracht niet opgenomen. Door de toevoeging van het ontwerpen en ontwikkelen van het framework is deze beroepstaak toegevoegd.

Deze beroepstaak heb ik zelfstandig uitgevoerd met de context complex. De definitie voor deze beroepstaak is hiervoor als volgt.

"Het betreft het ontwerpen van de architectuur voor meerdere applicaties. Er wordt rekening gehouden met niet-functionele kwaliteitsattributen en beveiligingsaspecten. De architectuur wordt beschreven vanuit verschillende gezichtspunten (viewpoints) en er wordt rekening gehouden met de belangen van de verschillende stakeholders."

Deze beroepstaak is uitgevoerd op niveau 4.

In sprint 1: Framework basis is een ontwerp gemaakt voor de architectuur van het framework. Dit ontwerp is tevens uitgewerkt in sprint 1.

Het framework maakt het mogelijk om reader apps te ontwikkelen. Er is een architectuur ontworpen voor meerdere applicaties. In deze architectuur is er rekening gehouden met de eis dat het framework crossplatform ondersteuning moest bieden. Dit maakt het ontwerpen van het framework complexer doordat er een deel van de code op drie platformen gebruikt moet kunnen worden.

In het framework is rekening gehouden met de uitbreidbaarheid. Er kunnen nieuwe ViewControllers toegevoegd worden of nieuwe Operating Systems ondersteund worden. Dit is mogelijk gemaakt door de toepassing van het Abstract Factory Pattern. Verder is de architectuur gedocumenteerd en zijn de onderlinge relaties tussen de verschillende onderdelen duidelijk gemaakt.

3.2 Ontwerpen systeemdeel
Deze beroepstaak heb ik zelfstandig uitgevoerd met de context complex. De definitie van complex voor deze beroepstaak is als volgt.

"Het betreft het ontwerpen van een gedistribueerde applicatie of een applicatie met complexe algoritmiek. Er wordt rekening gehouden met niet-functionele kwaliteitsattributen en beveiligingsaspecten en gebruik gemaakt van design patterns."

Deze beroepstaak is uitgevoerd op niveau 4. In het afstudeerplan staat dat deze beroepstaak zal worden uitgevoerd op niveau 4.

Er is gebruik gemaakt van:
- Overerving
- Implementeren van interfaces
- Object georiënteerd ontwerp
- Het ontwikkelde framework
- Functionele decompositie
- Design patterns
- Scheiding tussen Model en View

- UML

Er is rekening gehouden met hergebruik en uitbreidbaarheid in de Algemene Shared Core, Android Shared Core, Locaties.nl Shared Core en de ontwikkelde app voor Locaties.nl.

Voor sprint 5: Personalisatie & Tracking is er een score algoritme ontworpen.

Er zijn schermontwerpen gemaakt voor de app van Locaties.nl. Hierbij is er rekening gehouden met de interactie van de gebruiker met de app.

3.3 Bouwen applicatie
Deze beroepstaak heb ik zelfstandig uitgevoerd met de context lastig tot complex.

De definitie van lastig voor deze beroepstaak is als volgt.

"Het betreft het bouwen van een objectgeoriënteerde applicatie , waarbij geavanceerde concepten van de gebruikte programmeertaal aan de orde komen. Verder wordt rekening gehouden met toekomstige wijzigingen, testbaarheid en hergebruik. Het bouwen gebeurt in een ontwikkelomgeving."

De definitie van complex voor deze beroepstaak is als volgt.

"De applicatie sluit aan op bestaande software of maakt gebruik van een bestaand Framework. Het bouwen gebeurt in een geavanceerde ontwikkelomgeving inclusief testomgeving en versiebeheertool."

Deze beroepstaak heb ik uitgevoerd op niveau 3 tot 4. In het afstudeerplan staat dat deze beroepstaak zal worden uitgevoerd op niveau 3.

Er is gebruik gemaakt van:
- Code standaarden en naamconventies van C#
- Object georiënteerd programmeren in C#
- De ontwikkelomgeving van Xamarin
- Versiebeheertool(SVN) tijdens het ontwikkelen
- Het ontwikkelde framework
- Geavanceerde concepten van C# en Android

Bij het bouwen van het framework is er rekening gehouden met de herbruikbaarheid en uitbreidbaarheid.

13. Verklarende woordenlijst

Intent: Een beschrijving van een uit te voeren operatie. Het meest voorkomende gebruik is het opstarten van activities. In een Intent kan extra data opgeslagen worden, deze data kan gebruikt worden in de te openen activity.

Activity: Een applicatie component wat de gebruiker een user interface levert waarin hij een enkele, gefocuste handeling kan uitvoeren. Een app heeft vaak meerdere activities die losjes verbonden zijn met elkaar.

View: Het basic building block voor user interface componenten. Wordt gebruikt voor het maken van interactieve UI componenten zoals buttons en textfields.

ViewGroup: Een speciale View welke meerdere Views kan bevatten.

SharedPreferences: Een class in het Android framework dat gebruikt wordt voor het opslaan van primitieve data zoals booleans, floats, integers, longs en strings. De data zal blijven bestaan over verschillende sessies, zelfs nadat de applicatie afgesloten is.

Toast: Een Toast verstrekt simpele feedback over een operatie in een simpele popup.

Dialog: Een dialog is een klein venster dat de gebruiker om een beslissing of aanvullende informatie vraagt.

ListViewController: Toont een lijst met scrollbare items aan de gebruiker.

GridViewController: Toont items aan de gebruiker in een tweedimensionaal scrollbaar grid.

ScreenSlidePagerViewController: Een ViewPager welke de gebruiker in staat stelt om te swipen tussen verschillende pages.

DetailViewController: Toont een detail pagina van een locatie. Dit bevat een beschrijving en een plaatje.

DetailMetaDataViewController: Toont alle metadata van een locatie.

ContactViewController: Toont de contactgegevens van een locatie aan de gebruiker.

ImageViewController: Toont een ImageView aan de gebruiker.

Xamarin: Xamarin is een cross-platform solution dat gebruik maakt van een shared C# codebase. Per platform(Android, iOS en Windows Phone) wordt een app gebouwd die gebruik maakt van deze shared codebase. Deze apps worden ook ontwikkeld in C#.

Business Layer: De Business Layer bevat de Model classes die de applicatie nodig heeft en de business logica.

Service Access Layer: In de Service Access Layer bevat code om remote network resources op te halen, zoals web services, images downloaden etc. Het encapsuleert het netwerk gedrag en levert een API aan de Application en UI Layer.

Application Layer: De Application Layer bevat platform specifieke functionaliteit en de conversie tussen de Business Layer objecten en de User Interface.

Data Layer: Deze laag regelt de fysieke opslag van data. De Data Layer wordt over het algemeen alleen gebruikt door de Data Access Layer.

Data Access Layer: In de Data Access Layer zullen de data-operaties voor de applicatie staan als API. Methoden om data op te halen en te bewerken(CRUD) zonder de implementatie hiervan te laten zien aan de caller.

UI Layer: De User Interface Layer bevat de screen-layouts en de interface controls.

User Experience(UX): User Experience betreft het gedrag van een persoon, houding en emoties tijdens het gebruik van een bepaald product. Het bekijkt de bruikbaarheid, het gebruiksgemak en efficiëntie van een product.

14. Bibliografie

Android. (2014). *Android Development*. Opgehaald van Android Development: http://developer.android.com/guide/index.html

Android. (2014). *Google Cloud Messaging*. Opgehaald van Android Developer: http://developer.android.com/google/gcm/gs.html

Freeman, E., Robson, E., Bates, B., & Sierra, K. (2004). *Head First Design Patterns.* O'Reilly Media.

Haagse Hogeschool. (2014). *Afstuderen Academie ICT & Media.* Opgehaald van Blackboard.

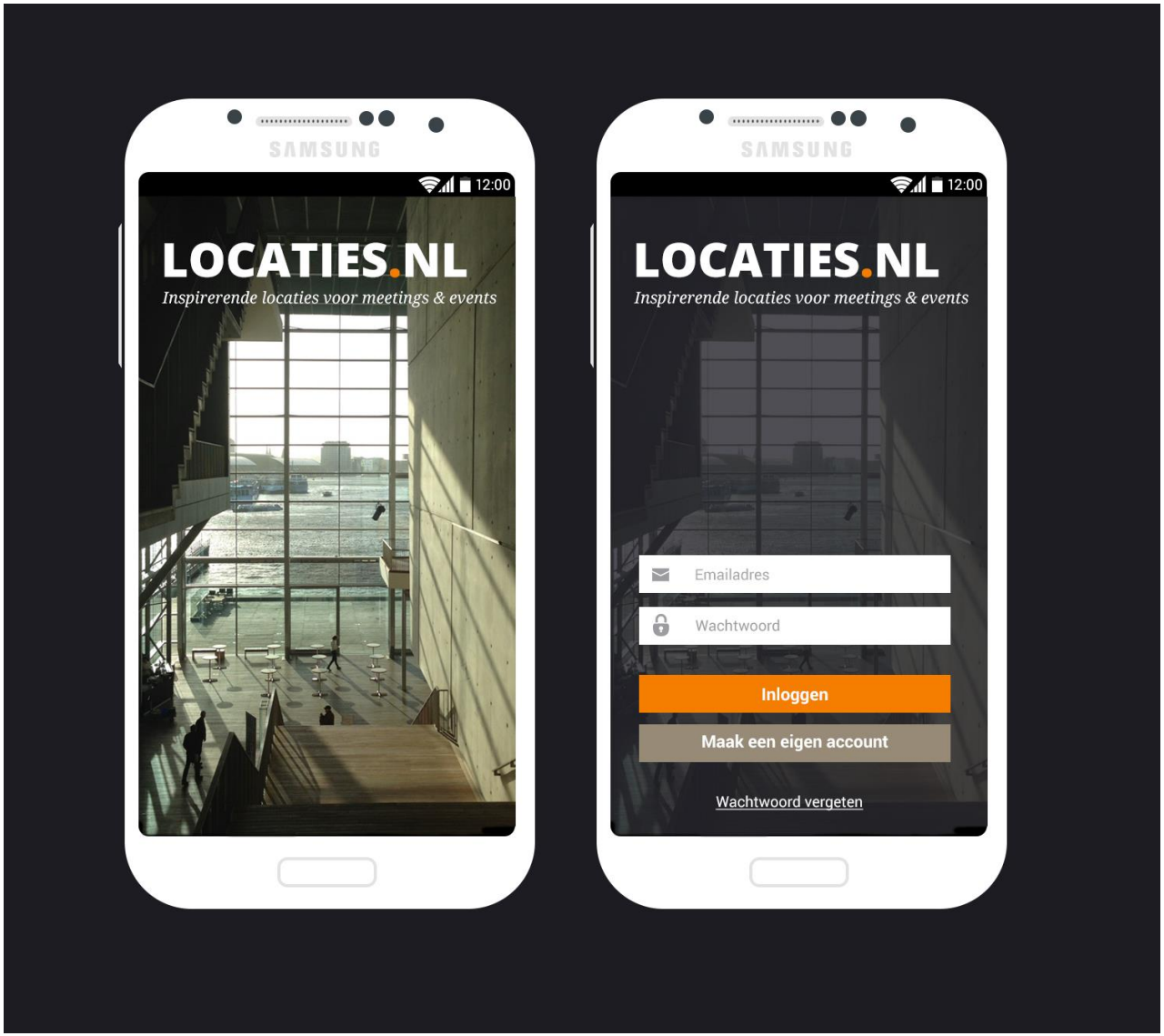
OODesign.com. (2014). *OODesign*. Opgehaald van Object Oriented Design: http://www.oodesign.com/abstract-factory-pattern.html

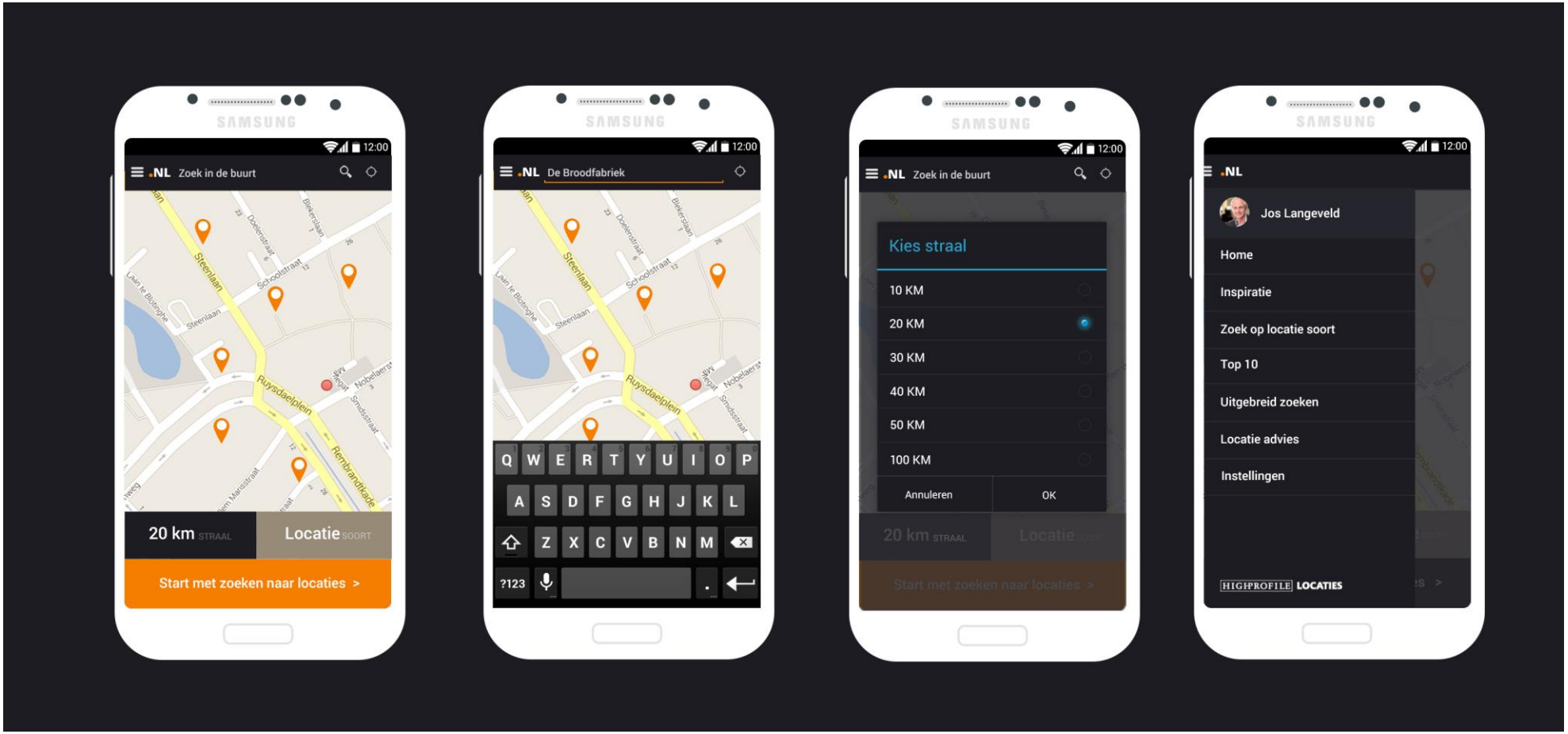
Wikipedia. (2014). *Normal Distribution*. Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Normal_distribution

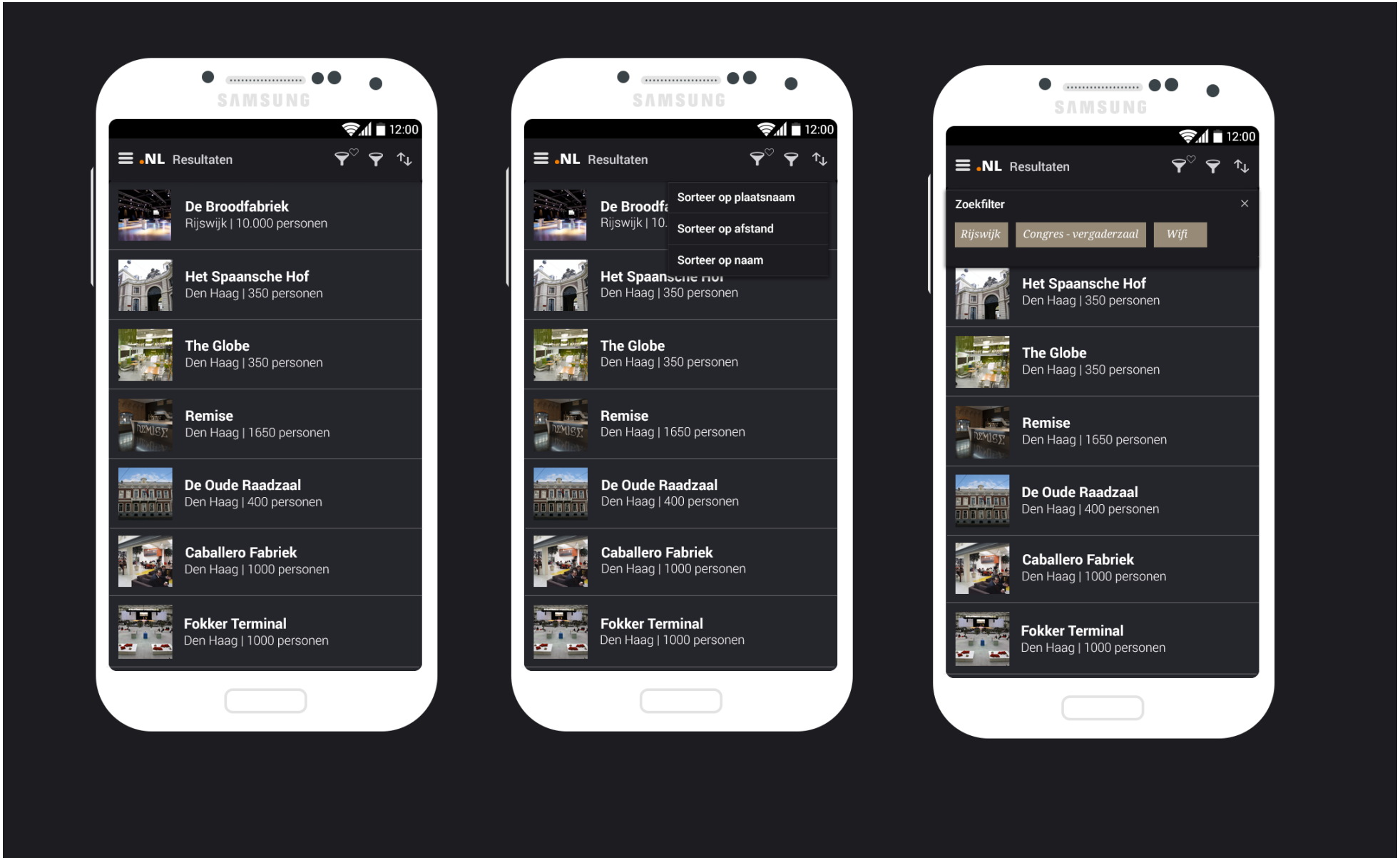
Xamarin. (2014). *Xamarin*. Opgehaald van Xamarin: docs.xamarin.com

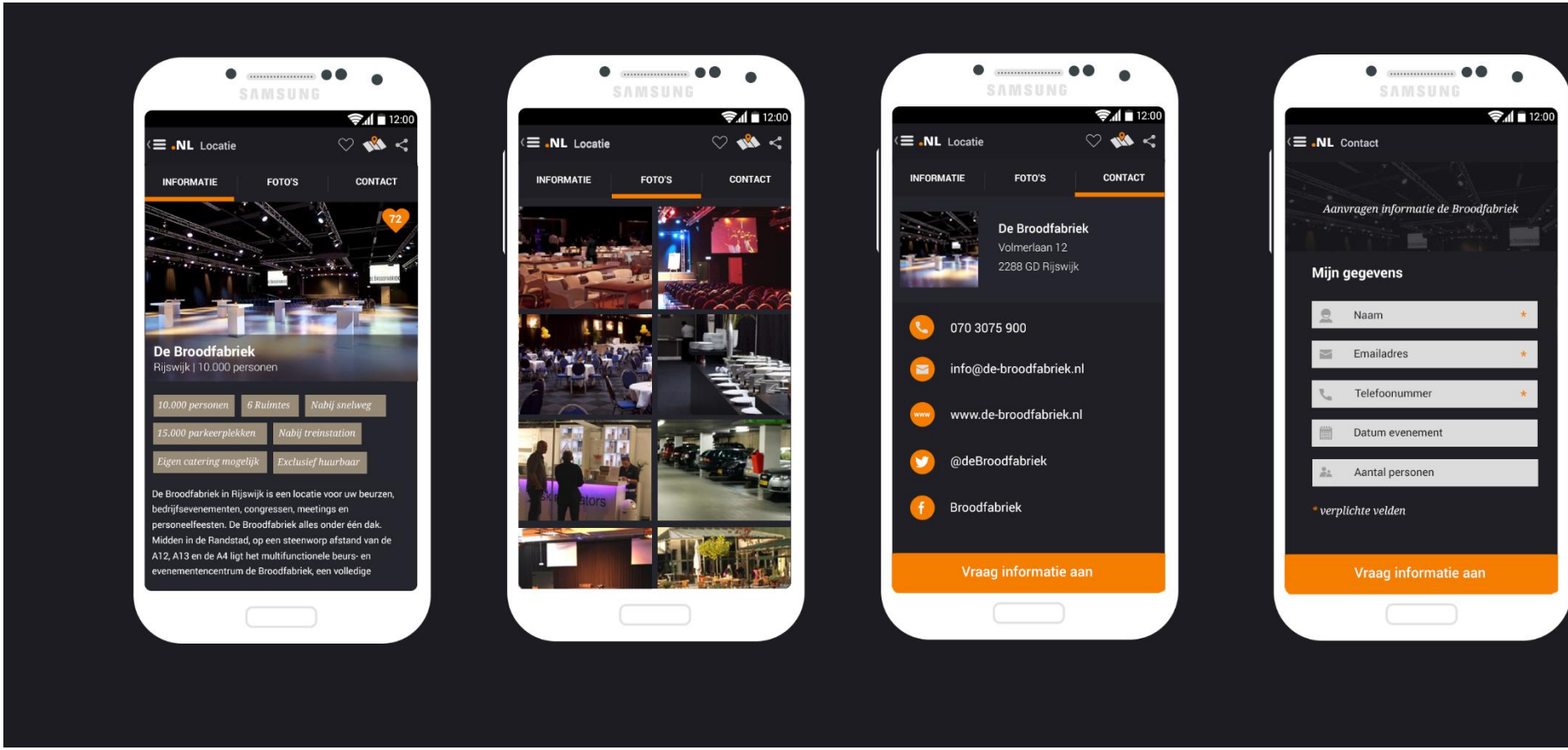
15. Bijlagen

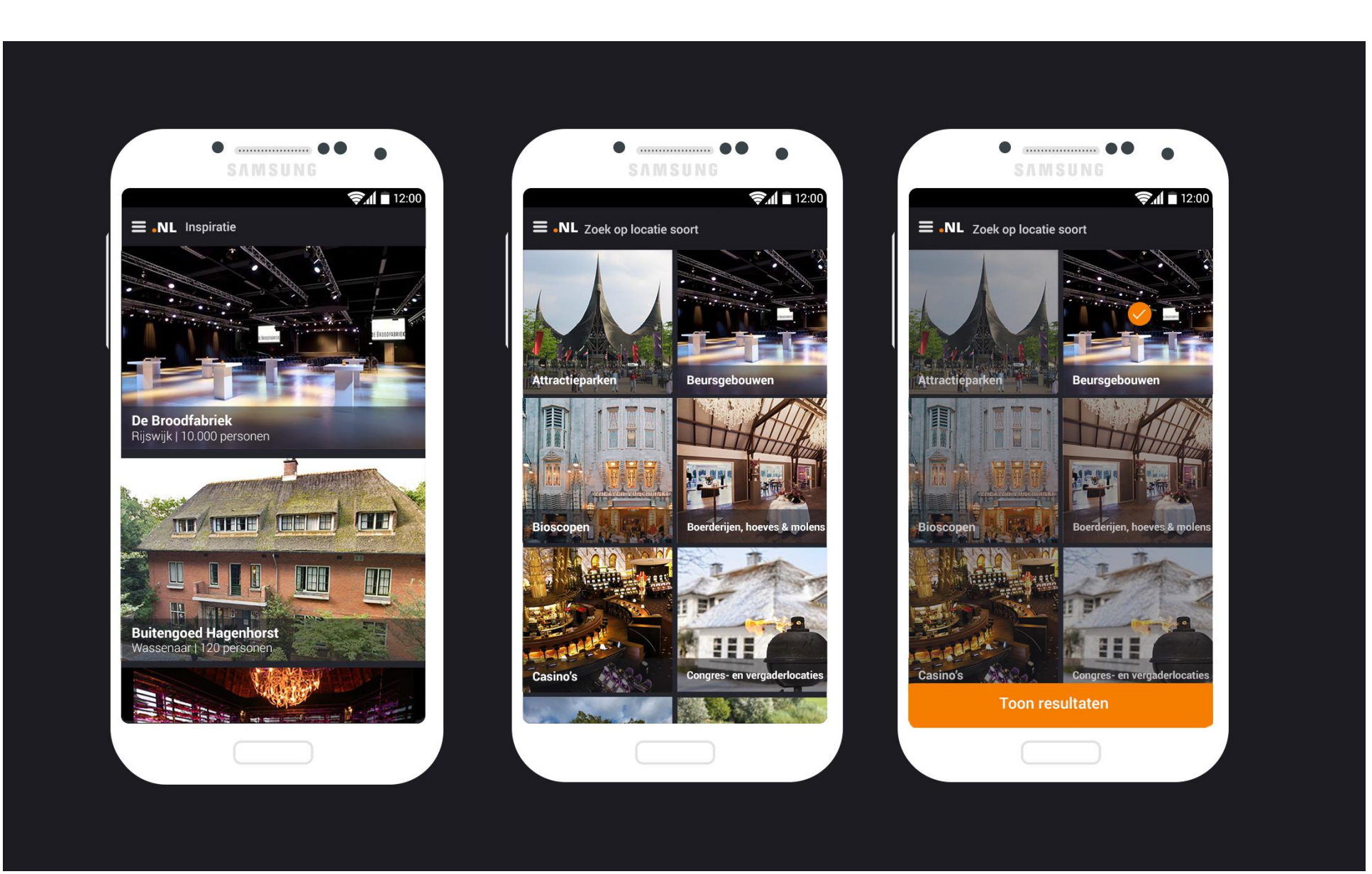
15.1. Schermontwerpen

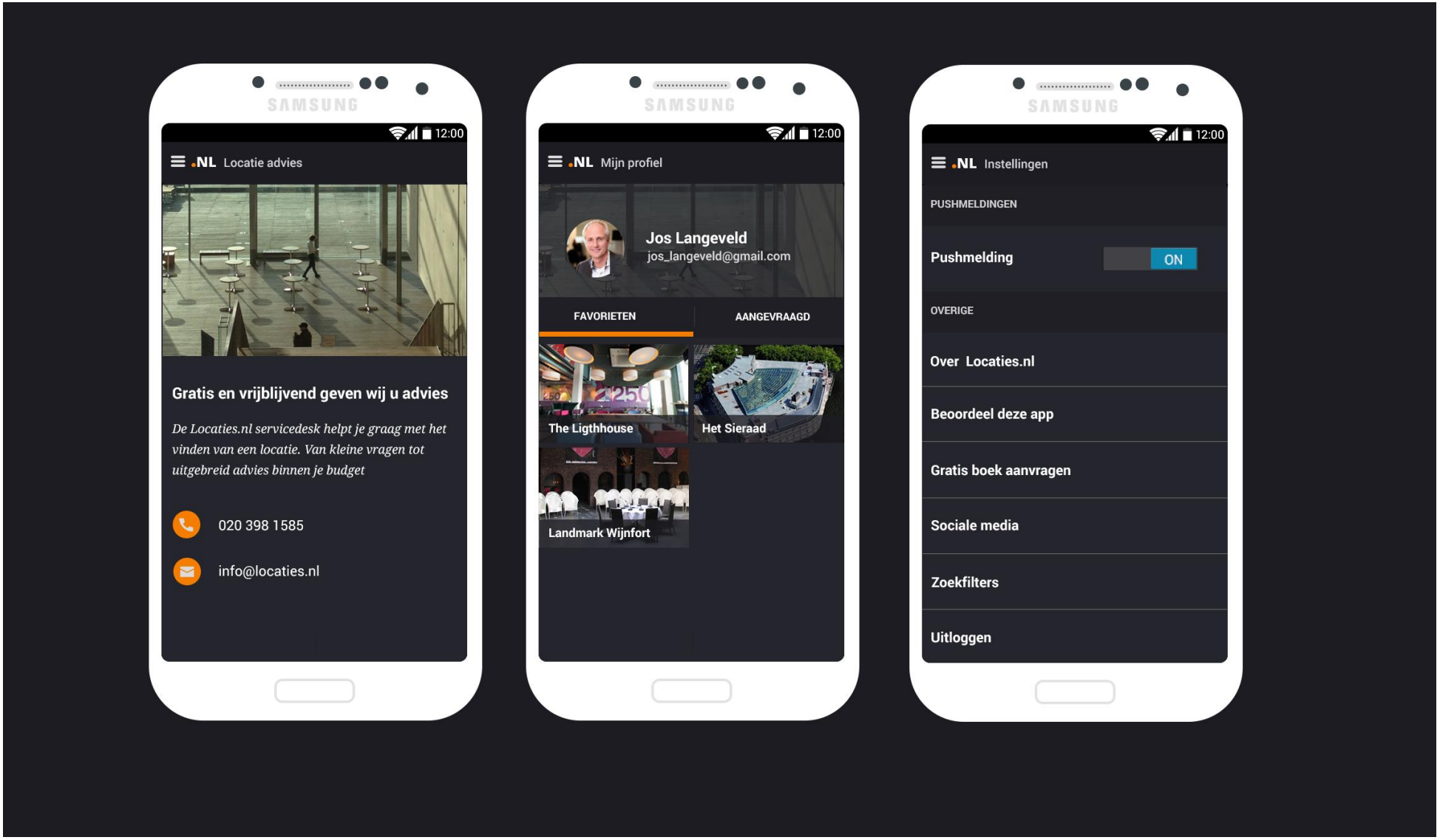


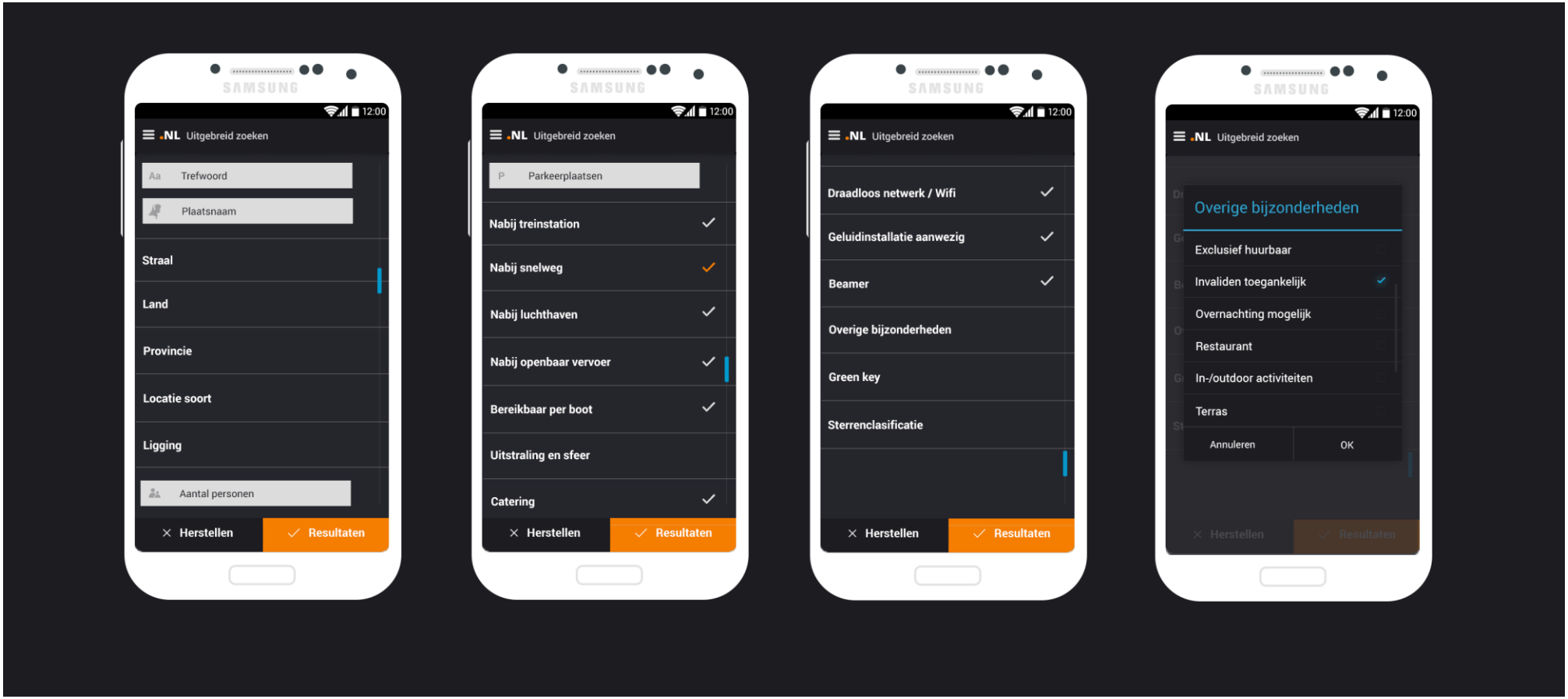

SAMSUNG
12:00
.NL Zoek in de buurt
20 km STRAAL
Locatie SOORT
Start met zoeken naar locaties >
.NL De Broodfabriek
Kies straal
10 KM
20 KM
30 KM
40 KM
50 KM
100 KM
Annuleren
OK
Jos Langeveld
Home
Inspiratie
Zoek op locatie soort
Top 10
Uitgebreid zoeken
Locatie advies
Instellingen
HIGHPROFILE LOCATIES


SAMSUNG
12:00
.NL Resultaten
De Broodfabriek
Rijswijk | 10.000 personen
Het Spaansche Hof
Den Haag | 350 personen
The Globe
Den Haag | 350 personen
Remise
Den Haag | 1650 personen
De Oude Raadzaal
Den Haag | 400 personen
Caballero Fabriek
Den Haag | 1000 personen
Fokker Terminal
Den Haag | 1000 personen
SAMSUNG
12:00
.NL Resultaten
Sorteer op plaatsnaam
Sorteer op afstand
Sorteer op naam
Den Haag | 350 personen
Den Haag | 350 personen
Den Haag | 1650 personen
Den Haag | 400 personen
Den Haag | 1000 personen
Den Haag | 1000 personen
SAMSUNG
12:00
.NL Resultaten
Zoekfilter
Rijswijk
Congres - vergaderzaal
Wifi
Het Spaansche Hof
Den Haag | 350 personen
The Globe
Den Haag | 350 personen
Remise
Den Haag | 1650 personen
De Oude Raadzaal
Den Haag | 400 personen
Caballero Fabriek
Den Haag | 1000 personen
Fokker Terminal
Den Haag | 1000 personen


12:00
.NL Locatie
INFORMATIE
FOTO'S
CONTACT
72
De Broodfabriek
Rijswijk | 10.000 personen
10.000 personen
6 Ruimtes
Nabij snelweg
15.000 parkeerplekken
Nabij treinstation
Eigen catering mogelijk
Exclusief huurbaar
De Broodfabriek in Rijswijk is een locatie voor uw beurzen, bedrijfsevenementen, congressen, meetings en personeelfeesten. De Broodfabriek alles onder één dak. Midden in de Randstad, op een steenworp afstand van de A12, A13 en de A4 ligt het multifunctionele beurs- en evenementencentrum de Broodfabriek, een volledige
12:00
.NL Locatie
INFORMATIE
FOTO'S
CONTACT
12:00
.NL Locatie
INFORMATIE
FOTO'S
CONTACT
De Broodfabriek
Volmerlaan 12
2288 GD Rijswijk
070 3075 900
info@de-broodfabriek.nl
www.de-broodfabriek.nl
@deBroodfabriek
Broodfabriek
Vraag informatie aan
12:00
.NL Contact
Aanvragen informatie de Broodfabriek
Mijn gegevens
Naam
Emailadres
Telefoonnummer
Datum evenement
Aantal personen
* verplichte velden
Vraag informatie aan


SAMSUNG
12:00
.NL Inspiratie
De Broodfabriek
Rijswijk | 10.000 personen
Buitengoed Hagenhorst
Wassenaar | 120 personen
SAMSUNG
12:00
.NL Zoek op locatie soort
Attractieparken
Beursgebouwen
Bioscopen
Boerderijen, hoeves & molens
Casino's
Congres- en vergaderlocaties
SAMSUNG
12:00
.NL Zoek op locatie soort
Attractieparken
Beursgebouwen
Bioscopen
Boerderijen, hoeves & molens
Casino's
Congres- en vergaderlocaties
Toon resultaten


SAMSUNG
12:00
.NL Locatie advies
Gratis en vrijblijvend geven wij u advies
De Locaties.nl servicedesk helpt je graag met het vinden van een locatie. Van kleine vragen tot uitgebreid advies binnen je budget
020 398 1585
info@locaties.nl
SAMSUNG
12:00
.NL Mijn profiel
Jos Langeveld
jos_langeveld@gmail.com
FAVORIETEN
AANGEVRAAGD
The Ligthhouse
Het Sieraad
Landmark Wijnfort
SAMSUNG
12:00
.NL Instellingen
PUSHMELDINGEN
Pushmelding
ON
OVERIGE
Over Locaties.nl
Beoordeel deze app
Gratis boek aanvragen
Sociale media
Zoekfilters
Uitloggen


12:00
.NL Uitgebreid zoeken
Aa Trefwoord
Plaatsnaam
Straal
Land
Provincie
Locatie soort
Ligging
Aantal personen
Herstellen
Resultaten
P Parkeerplaatsen
Nabij treinstation
Nabij snelweg
Nabij luchthaven
Nabij openbaar vervoer
Bereikbaar per boot
Uitstraling en sfeer
Catering
Draadloos netwerk / Wifi
Geluidinstallatie aanwezig
Beamer
Overige bijzonderheden
Green key
Sterrenclasificatie
Overige bijzonderheden
Exclusief huurbaar
Invaliden toegankelijk
Overnachting mogelijk
Restaurant
In-/outdoor activiteiten
Terras
Annuleren
OK

15.2. Sitemap


Legenda
Schermen
Startscherm
Inloggen
Registreren
Zoek in de buurt
Zoek op naam
Menu
Zoekafstand
Locatie soort
Zoekresultaten
Home
Profiel
Favorieten
Aangevraagd
Inspiratie
Lijst locaties
Zoek op locatie soort
Locatie soorten
Top 10
Lijst locaties
Uitgebreid zoeken
Zoeken op metadata
Locatie advies
Instellingen
Pushmeldingen
Over locaties.nl
Beoordeel deze app
Sociale media
Uitloggen
Locatie
Resultaten
Filteren
Sorteren
Informatie
Foto's
Contact
Kaart
Favorieten
Delen
Formulier

15.3. Sprint backlogs

15.3.1. Backlog sprint 1: Framework Basis

Points	Story	Description
	To Do	
	Bobcat-1.0 hoort bij deze sprint	
	Als developer wil ik dat mijn Shared Core, OS-Shared Cores en OS specifieke implementaties in losse csproj opgezet zijn. De Shared Cores zijn dan te gebruiken in elke nieuwe implementatie.	Projecten worden dan: 1. BobcatSC.csproj (= alle non-OS specifieke code, herbruikbaar in elk nieuw project) 2. BobcatSC.Android.csproj (= alle Android specifieke code, zonder klant specifieke implementatie) 3. BobcatSC.IOS.csproj (= alle iOS specifieke code, zonder klant specifieke implementatie) 4. BobcatSC.Windows.csproj (= alle Windows Phone specifieke code, zonder klant specifieke implementatie) 5. Locaties.Android (= De Android specifieke klant implementatie) 6. Locaties.IOS (= De iOS specifieke klant implementatie) 7. Locaties.Windows (= De Windows Phone specifieke klant implementatie)
	Als developer wil ik de applicatielaag van Locaties.nl uitgeschreven hebben in zo eenvoudig mogelijke code, zodat ik een basis heb voor de abstractielaag.	
	Als developer wil ik een lijst kunnen aanmaken met behulp van het framework.	
	Als developer wil ik een imagegrid kunnen aanmaken met behulp van het framework.	
	Als developer wil ik een viewpager aan kunnen maken met behulp van het framework.	
	Als developer wil ik een detailpagina aan kunnen maken met behulp van het framework.	
	Als developer wil ik een detailpagina met metadata kunnen toevoegen met behulp van het framework.	

	Als developer wil ik een image kunnen tonen met behulp van het framework.	
	Als developer wil ik op een eenduidige manier datasources kunnen aanspreken	In eerste instantie volgens standaarden (JSON, RSS, ATOM etc.). Implementatie hoeft nog niet. Wel nadenken over methodes / best practices volgens Xamarin.
	Als developer wil ik de Navigation Drawer gelocked hebben als de ViewController van het type MapViewController is.	
	Als developer wil ik weten of het mogelijk is om views te stylen met CSS, zodat ik het design sneller om kan zetten	
	Als developer wil ik dat er een Google Map getoond wordt van mijn locatie.	
	Als developer wil ik op een eenduidige manier services kunnen aanspreken	Service Access layer. Bijv. SOAP, REST
	Als developer wil ik dat er met de Navigation Drawer genavigeerd kan worden tussen activities.	
	Als stagebegeleider wil ik dat het verslag is bijgewerkt zodat Marc afstudeert.	
	Cast in render() verwijderen in Framework	
	ViewControllers benoemen van wireframe	
	ImageDownloadTask van ListViewController aparte class maken en eventueel interface	
	Op basis van DetailViewController een ContactViewController maken.	
	ImageGridViewController niet meer laten erven van ListViewController of aantal overrides aanpassen (ScrollStateChange)	
	ImageViewController aanmaken op basis van DetailViewController, deze aanmaken in render van DetailViewController	
Doing		

Done

15.3.2. Backlog sprint 2: App flow Locaties.nl

Points	Story	Description
To Do		
	Als stagebegeleider wil ik dat het verslag is bijgewerkt zodat Marc afstudeert.	
	Bobcat-1.3 hoort bij deze sprint	BobcatAndroid = Android Project BobcatSC = Library Project BobcatSC.Android = Android Library Project LocatiesSC = Library Project
	Als klant wil ik dat de flow is uitgewerkt in een interactie ontwerp	
	Als developer wil ik de gehele flow van de applicatie uitgewerkt hebben in een eerste versie met hierin de belangrijkste functionaliteiten.	
	Als developer wil ik de settings meegeven in AndroidViewControllerSettings voor de verschillende controllers	
	in DetailViewController: footer weg, duimpjes weg, showonmap naar ActionBar, favorite naar ActionBar, shareknop erbij in ActionBar	
	in GridViewController: de tekst op het plaatje witte font met border(zwart, schaduw)	
	Als developer wil ik de Soortvermelding implementeren in de LocationQuery	
	Als developer wil ik een splashscreen tonen bij het opstarten van de app	
	load dialog fixen in LocationsOverviewActivity	
	Als developer wil ik in de ListView met gevonden locaties meerdere slides met verschillende sorteringen	met radius: slide 1: naam slide 2: afstand etc. zonder radius: slide 1: afstand slide 2: naam etc. straal uit locationsnearby misschien weg searchview in locationsoverview
	Als developer wil ik de backnavigation met het icon afhandelen	

	Als developer wil ik een settingsscreen met preferences	
	Als developer wil ik een check inbouwen wanneer de huidige locatie nog niet gevonden is	
	Als developer wil ik dat het thema instelbaar is in dark en light	
Doing		
Done		

15.3.3. Backlog sprint 3: Zoek & Detail schermen Locaties.nl

Points	Story	Description
To Do		
	Als stagebegeleider wil ik dat het verslag is bijgewerkt zodat Marc afstudeert.	
	Bij deze sprint hoort Bobcat-1.4	
	Als gebruiker wil ik dat als mijn internetverbinding wegvalt, dat ik een duidelijke melding krijg	
	Als gebruiker wil ik dat als mijn GPS wegvalt, dat ik een duidelijke melding krijg	
	Als gebruiker wil ik de app alleen afsluiten met 2x back drukken	
	Als gebruiker wil ik zien hoeveel zoekresultaten mijn zoekopdracht heeft opgeleverd.	
	Als gebruiker wil ik kunnen zoeken op locatie type	
	Als klant wil ik dat gebruikers een top 10 kunnen bekijken, om de conversie te verhogen	
	Als gebruiker wil ik in de contactpagina van een locatie een koppeling hebben naar de telefoon met het telefoonnummer van de locatie	
	Als gebruiker wil ik in de contactpagina van een locatie een koppeling hebben naar de mail	

	Als gebruiker wil ik in de contactpagina van een locatie een koppeling hebben naar de website van de locatie	
	Als gebruiker wil ik van een locatie de foto's zien.	
	Als gebruiker wil ik de locatie op een kaart zien.	
	Als gebruiker wil ik van een locatie de beschrijving zien.	
	Als gebruiker wil ik van een locatie de naam zien.	
	Als gebruiker wil ik een straal kunnen opgeven wanneer ik in de buurt zoek.	
	Als developer wil ik de content van een locatie meesturen in een LocationResult	
	Als gebruiker wil ik kunnen zoeken op locaties in de buurt.	
	Als gebruiker wil ik verschillende menu's voor de verschillende contexten.	
	Als gebruiker wil ik verschillende soorten menu's in de verschillende activities.	
	Als gebruiker wil ik mijn uitgebreide zoekopdracht kunnen resetten.	
	Als gebruiker wil ik de adresgegevens zien.	
	Als gebruiker wil ik kunnen zoeken op plaats	En mogelijkheid om te filteren op afstand
	Als gebruiker wil ik kunnen zoeken op naam	
	Als gebruiker wil ik een lijst kunnen bekijken van locaties die voldoen aan mijn zoekopdracht.	
	Als gebruiker wil ik een boodschap krijgen indien er geen zoekresultaten zijn na het uitvoeren van een zoekopdracht.	
	Als gebruiker wil ik de bereikbaarheid zien van een locatie.	
	Als gebruiker wil ik de bijzonderheden zien van een locatie.	
	Als gebruiker wil ik de uitstraling en sfeer van een locatie zien.	

	Als gebruiker wil ik de ligging en omgeving van een locatie zien.	
	Als gebruiker wil ik het aantal eenpersoonskamers van een locatie zien.	
	Als gebruiker wil ik het aantal tweepersoonskamers van een locatie zien.	
	Als gebruiker wil ik het aantal meerpersoonskamers van een locatie zien.	
	Als gebruiker wil ik dat de navigation drawer aangepast is op de huidige context.	
	Als gebruiker wil ik de gekoppelde metadata zien.	
	Als gebruiker wil ik een zaaloverzicht zien.	
	Als gebruiker wil ik kunnen aangeven of een locatie relevant is.	
	Als gebruiker wil ik uitgebreid kunnen zoeken	LocationQuery bevat geen capacity, zalen, parkeerplaatsen en vergaderhamer LocationResult bevat geen parkeersuggestie RetrieveLocations met 0 results returnt 0, RetrieveLocationsMinimal met 0 results geeft een error.
	Als developer wil ik dat de Picklist cache maximaal een dag oud is.	
	Als gebruiker wil ik op aantal eenpersoonskamers kunnen zoeken.	
	Als gebruiker wil ik op aantal tweepersoonskamers kunnen zoeken.	
	Als gebruiker wil ik op aantal meerpersoonskamers kunnen zoeken.	
Doing		
Done		

15.3.4. Backlog sprint 4: Overige schermen Locaties.nl

Points	Story	Description
	To Do	
	Als stagebegeleider wil ik dat het verslag is bijgewerkt zodat Marc afstudeert.	
	Als klant wil ik dat gebruikers een top 10 kunnen bekijken, om de conversie te verhogen	
	Als gebruiker wil ik in de app kunnen inloggen met mijn account van Locaties.nl.	
	Als gebruiker wil ik een account kunnen aanmaken met behulp van de app.	
	Als gebruiker wil ik de favorieten van de site ook kunnen bekijken in de app.	
	Als gebruiker wil ik de favorieten van de app ook kunnen bekijken op de site.	
	Als gebruiker wil ik een optie kunnen plaatsten op een locatie op een bepaalde datum.	
	De gebruiker vult bij elke aanvraag zijn persoonlijke informatie in.	
	De gebruiker vult bij elke aanvraag zijn bedrijfsnaam in.	
	De gebruiker vult bij elke aanvraag zijn voornaam in.	
	De gebruiker vult bij elke aanvraag zijn achternaam in.	
	De gebruiker selecteert bij elke aanvraag zijn geslacht.	
	De gebruiker vult bij elke aanvraag zijn e-mailadres in.	
	De gebruiker vult bij elke aanvraag zijn telefoonnummer in.	
	De gebruiker vult bij elke aanvraag eventueel het aantal personen in.	
	De gebruiker vult bij elke aanvraag eventueel de datum van het evenement in.	
	De gebruiker vult bij elke aanvraag eventueel een opmerking in.	

	De gebruiker selecteert eventueel of hij informatie over locatie aanbiedingen wil ontvangen op het ingevulde e-mailadres.	
	De gebruiker vult bij elke aanvraag een security code in.	
Doing		
Done		

15.3.5. Backlog sprint 5: Personalisatie & Tracking

Points	Story	Description
To Do		
	Als stagebegeleider wil ik dat het verslag is bijgewerkt zodat Marc afstudeert.	
	Als developer wil ik weten hoe het profiel lokaal wordt opgebouwd	
	Als gebruiker wil ik favorieten kunnen beheren.	
	Als gebruiker wil ik locatie suggesties krijgen.	
	Als gebruiker wil ik dat er een persoonlijk profiel opgebouwd wordt op basis van mijn gedrag in de app.	
	Als gebruiker wil ik meerdere profielen kunnen beheren in mijn account.	
	Als gebruiker wil ik mijn homescreen kunnen aanpassen.	
	Als gebruiker wil ik de historie van zoekopdrachten kunnen zien.	
Doing		
Done		

15.4. Onderzoek naar voorkeuren t.o.v. UI navigeren

In dit document is er een onderzoek naar UI navigeren in bestaande reader apps.

De twee vragen die met dit onderzoek beantwoord moesten worden zijn:

- Welke navigatiestructuren worden er in bestaande reader apps gebruikt?
- Op welke manier tonen de bestaande reader apps hun content?

Op basis van de bevindingen uit het onderzoek zal er antwoord gegeven worden op deze vragen.

De volgende apps zijn hiervoor gebruikt:

- Flipboard
- Zite
- The Guardian
- Material

De keuze was gevallen op deze vier apps omdat deze behoren tot de meest toonaangevende en professionele apps die er zijn op dit moment op het gebied van reader apps.

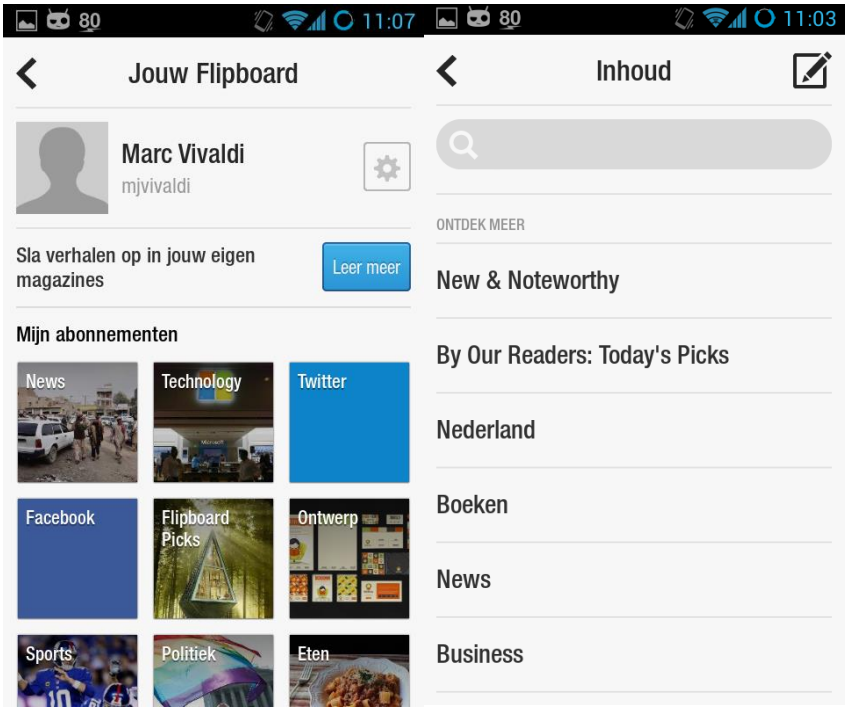
Deze apps tonen nieuws aan de gebruiker in verschillende categorieën. Vaak zijn deze categorieën door de gebruiker zelf in te stellen. De apps werden geanalyseerd op navigatiestructuren en de manier van het tonen van de content.

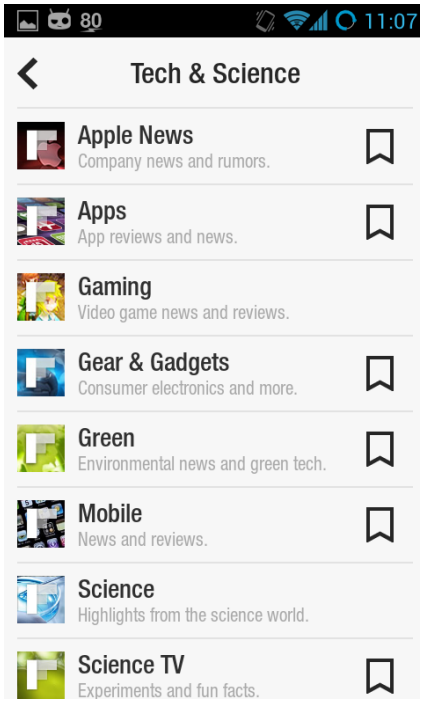
Om alle gebruikte navigatiestructuren in kaart te brengen zijn er screenshots gemaakt in elke app bij belangrijke onderdelen. Verder is er gekeken welke content er getoond wordt op welke niveaus en welke verschillen dit met zich meebracht.

De apps zijn geanalyseerd en gedocumenteerd. Dit werd gedaan voor 3 verschillende niveaus. Dit waren categorie-, artikelenoverzicht- en artikelniveau. Hieronder zijn de resultaten getoond van de analyse per niveau.

Flipboard

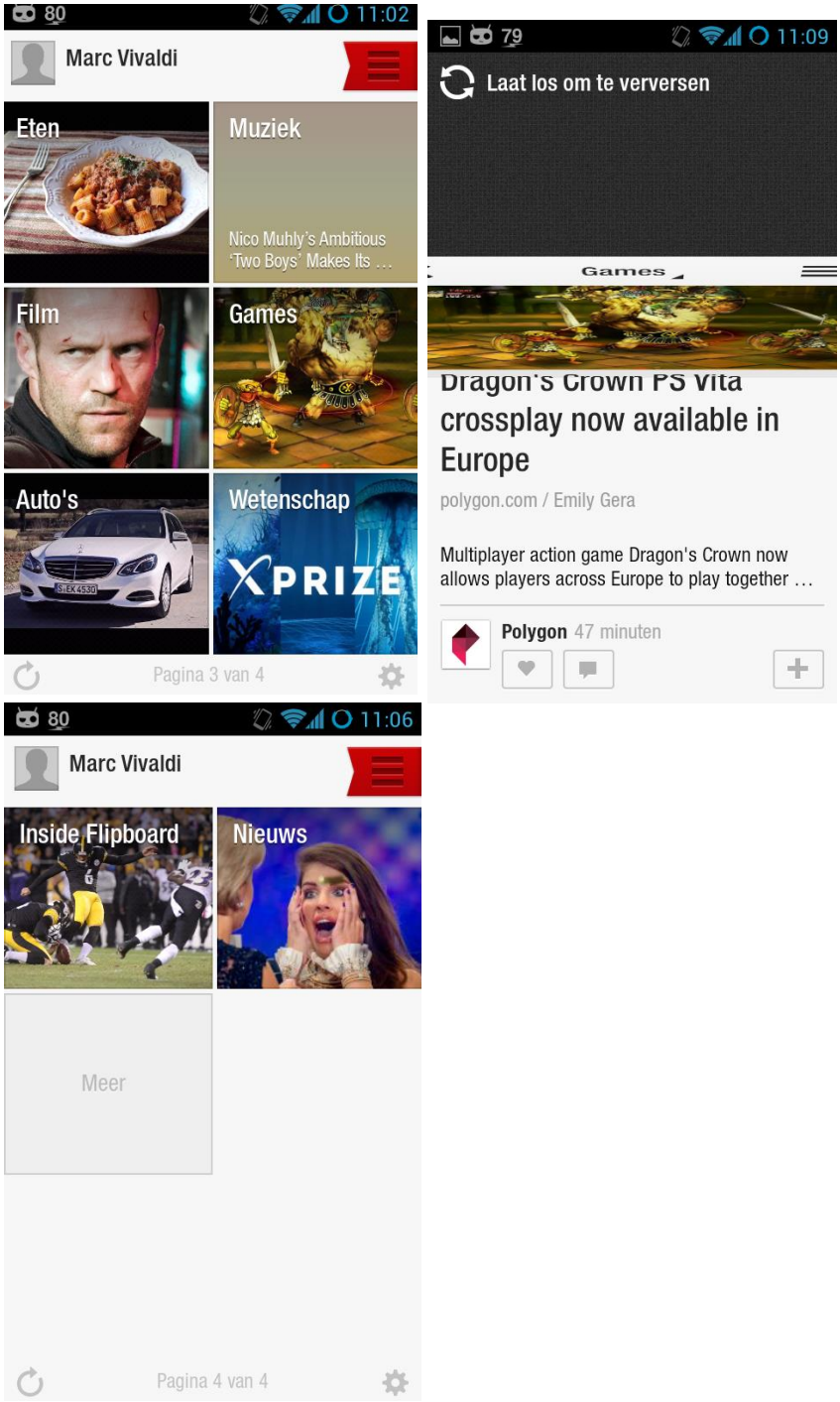
Bij de 1e keer opstarten wordt er gevraagd om categorieën toe te voegen.
Nieuwe categorieën kunnen toegevoegd worden door middel van bij de settings een categorie aanklikken(openen) en daarin op een feed te abonneren, dit komt dan in je persoonlijke lijst te staan.





Categorieën

Alle categorieën worden op de het startscherm getoond, met verticaal swipen blader je door de categorieën. Elke categorie heeft een foto van het meest recente artikel .
Er kunnen nieuwe artikelen ingeladen worden door middel van het scherm naar beneden te trekken.



Artikelenoverzicht

Na selecteren van een categorie wordt er een preview gegeven van het meest recente artikel.
Met een swipe naar rechts navigeert de gebruiker terug naar het categorie overzicht.
Met een swipe naar beneden worden er nieuwe artikelen geladen.
Met een swipe naar boven wordt de preview van het volgende(oudere) artikel getoond.

Van een artikel worden in de preview de volgende items weergegeven:

- Foto
- Koptekst
- Begin van het artikel
- Bron

- Leeftijd van het artikel

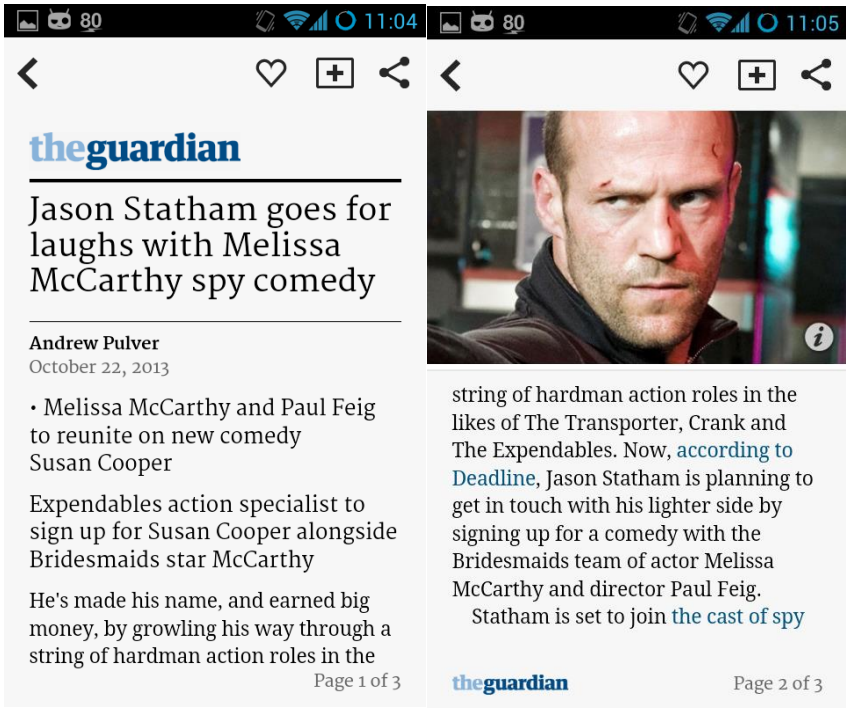
Artikel

Na het openen van een artikel zijn er de volgende mogelijkheden:

- Met een swipe naar links navigeert de gebruiker terug naar het artikeloverzicht
- Met vertical swipen bladert de gebruiker door het artikel

Getoond in het artikel worden:

- Bron
- Titel
- Auteur/datum van publicatie
- Tekst
- Foto



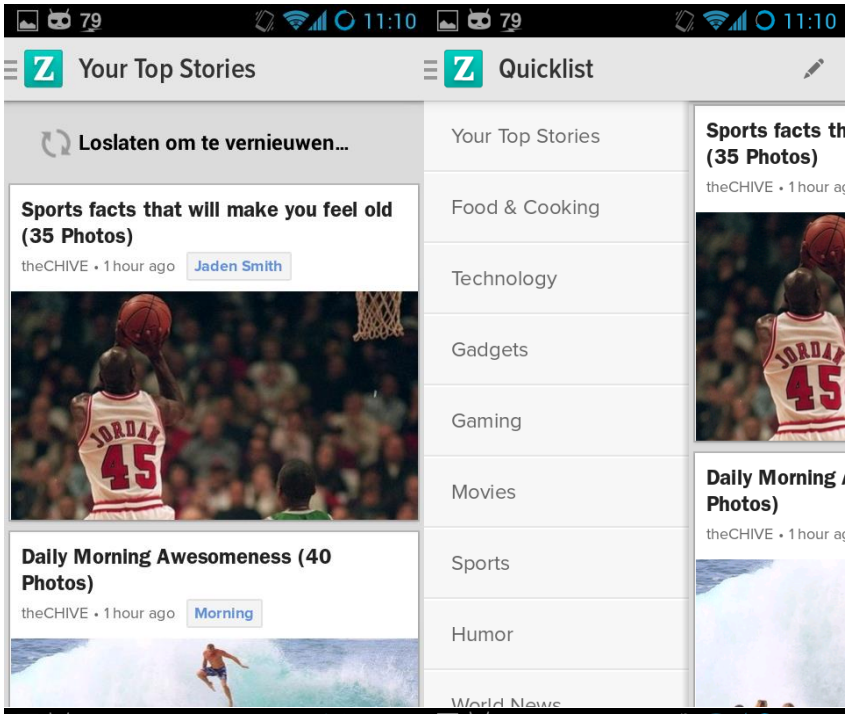
Samenvatting

Flipboard heeft een goede navigatie structuur die erg logisch aanvoelt.
Het categorieniveau ziet er overzichtelijk uit.
Het artikeloverzicht ziet er niet overzichtelijk uit.
In het artikel werkt het als het omslaan van een pagina.

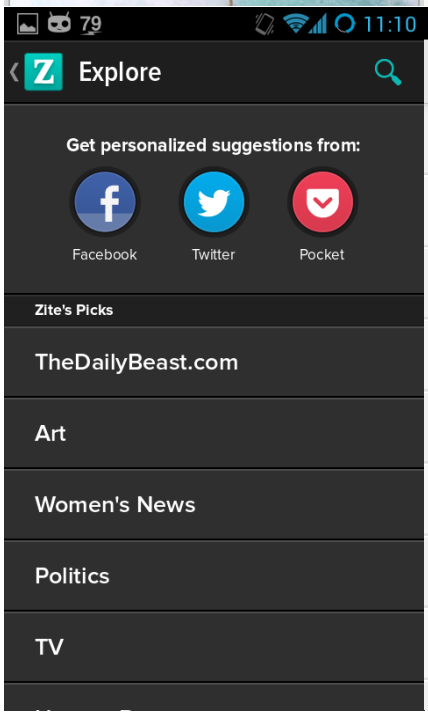
Zite

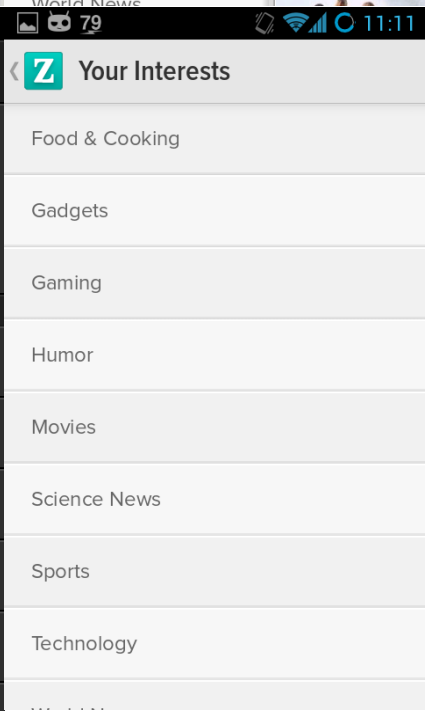
Bij de eerste keer opstarten wordt er gevraagd om een selectie te maken van categorieën die de gebruiker interessant vindt. Op deze manier richt de gebruiker zijn profiel in.
Zite rankt artikelen voor de gebruiker door middel van tracking.
De gebruiker kan categorieën aanvinken wanneer hij deze interessant vindt, Zite zal hiervan artikelen tonen en tevens de categorie toevoegen aan het profiel.
Zite bevat een Quicklist voor veelgebruikte topics

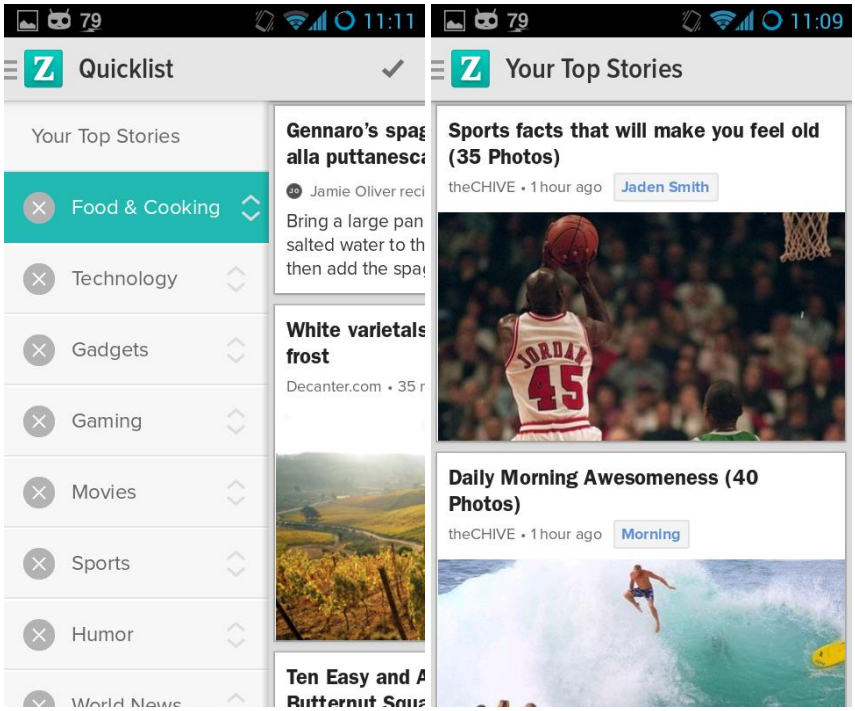
Top Stories worden als eerst getoond.
Swipe naar rechts geeft menu weer met daarin de categorieën.
Met een swipe naar beneden worden er nieuwe artikelen geladen.
Artikelen kunnen per categorie getoond worden.


11:10
Your Top Stories
Loslaten om te vernieuwen...
Sports facts that will make you feel old (35 Photos)
theCHIVE • 1 hour ago
Jaden Smith
Daily Morning Awesomeness (40 Photos)
theCHIVE • 1 hour ago
Morning

11:10
Quicklist
Your Top Stories
Food & Cooking
Technology
Gadgets
Gaming
Movies
Sports
Humor
Sports facts th (35 Photos)
theCHIVE • 1 hour ag
Daily Morning A Photos)
theCHIVE • 1 hour ag


11:10
Explore
Get personalized suggestions from:
Facebook
Twitter
Pocket
Zite's Picks
TheDailyBeast.com
Art
Women's News
Politics
TV


11:11
Your Interests
Food & Cooking
Gadgets
Gaming
Humor
Movies
Science News
Sports
Technology



Categorieën

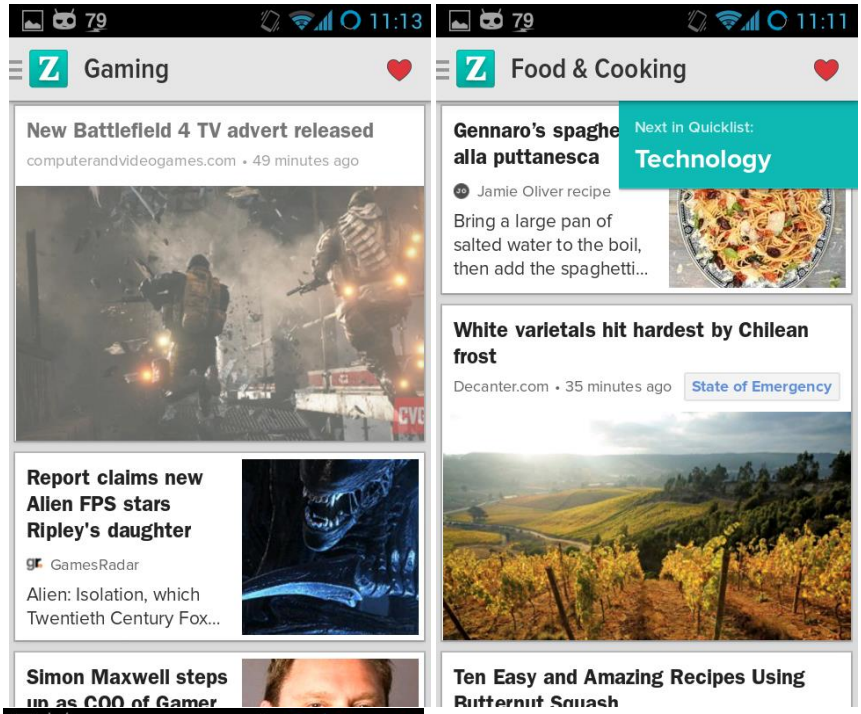
Alle categorieën worden in de navigation drawer getoond, dit heet de Quicklist.

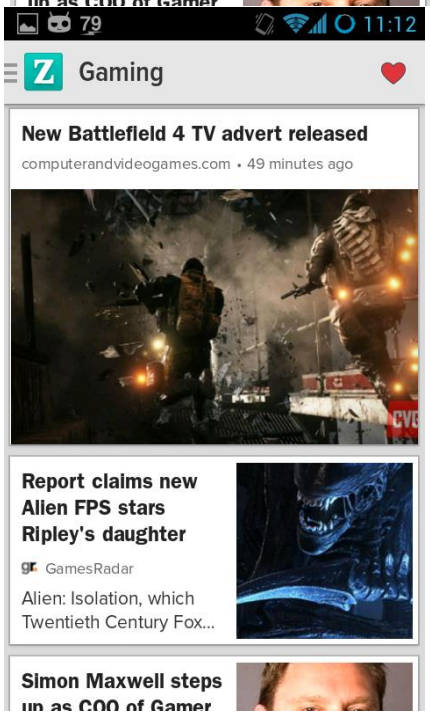
Artikeloverzicht

Zite toont de Top Stories of artikelen per categorie.
De artikelen staan onder elkaar, er kan gescrold worden door de artikelen.
Nadat een artikel bekeken is en de gebruiker keert terug naar het artikeloverzicht, dan worden de bekeken artikelen grijs gemarkeerd.
Met een swipe naar links wordt de volgende categorie van de Quicklist getoond.

Van een artikel wordt getoond:

- Foto
- Koptekst
- Begin van het artikel
- Bron
- Leeftijd van het artikel
- Onderwerp



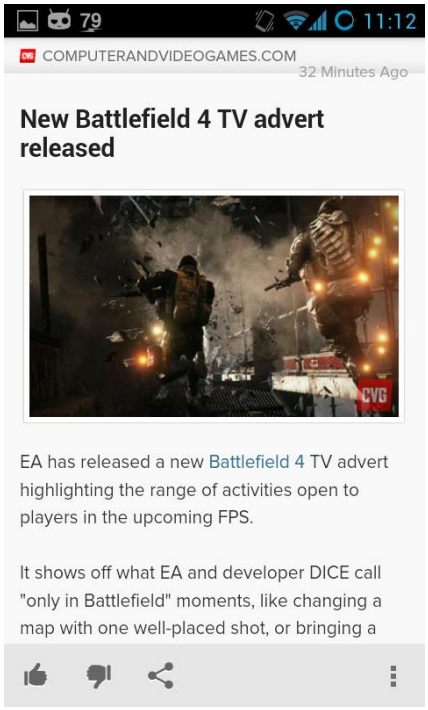


Artikel

Getoond in het artikel worden:

- Bron
- Titel
- Auteur
- Tekst
- Foto

Er wordt door een artikel genavigeerd door middel van verticaal scrollen. Met een swipe naar rechts wordt er teruggekeerd naar het artikeloverzicht.



Samenvatting

Een minpunt is het selecteren van een categorie. Er kan niet op categorieniveau een categorie geselecteerd worden, alleen door middel van de Quicklist.

Material

Bij Material krijgt de gebruiker een editie per dagdeel.

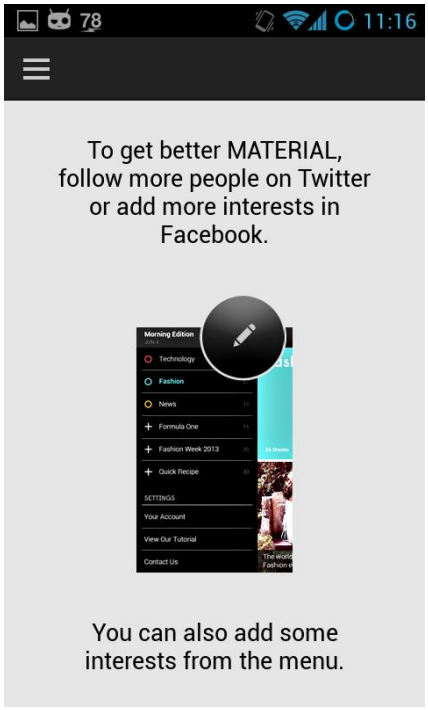
Categorieën

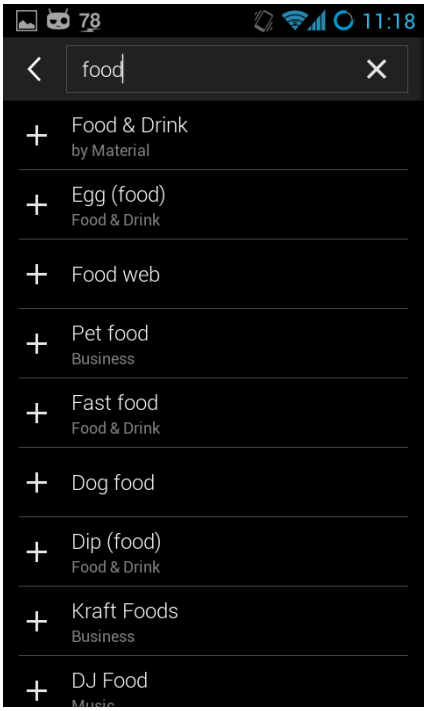
Er kunnen categorieën worden toegevoegd door:

- meer mensen te volgen op twitter
- meer interesses toe te voegen in facebook
- categorieën toe te voegen in het menu

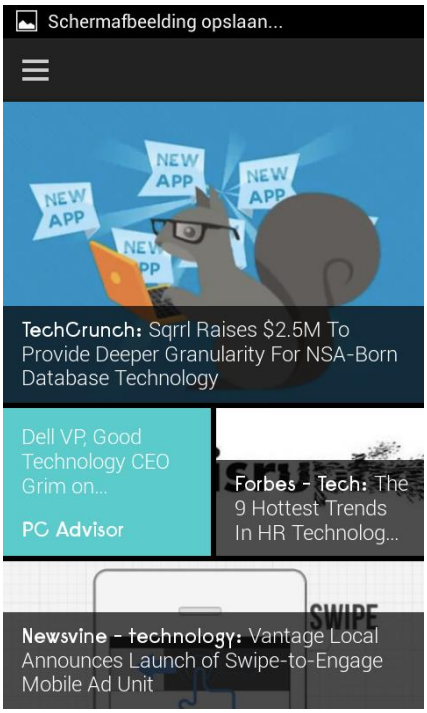
In het menu kan men zoeken naar een bepaalde interesse en deze toevoegen. Deze interesses worden in de volgende editie opgenomen.

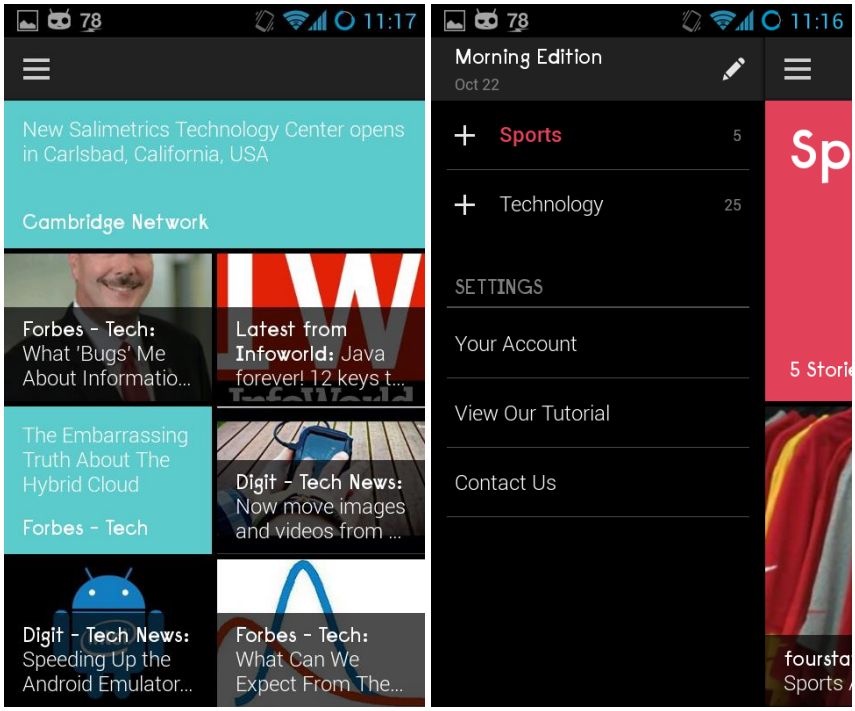
De gebruiker krijgt de artikelen per categorie gepresenteerd, als het ware de pagina omslaan om de volgende set artikelen te zien van deze categorie.


11:16
To get better MATERIAL, follow more people on Twitter or add more interests in Facebook.
You can also add some interests from the menu.


11:18
food
Food & Drink
by Material
Egg (food)
Food & Drink
Food web
Pet food
Business
Fast food
Food & Drink
Dog food
Dip (food)
Food & Drink
Kraft Foods
Business
DJ Food


11:16
Technology
25 Stories
Mashable: None of Your IT Coworkers Understand Your Job


Schermafbeelding opslaan...
NEW APP
NEW APP
TechCrunch: Sqrrl Raises $2.5M To Provide Deeper Granularity For NSA-Born Database Technology
Dell VP, Good Technology CEO Grim on...
PC Advisor
Forbes - Tech: The 9 Hottest Trends In HR Technolog...
SWIPE
Newsvine - technology: Vantage Local Announces Launch of Swipe-to-Engage Mobile Ad Unit



Artikeloverzicht

Horizontaal swipen verandert de categorie.
Als de gebruiker de linker rand van het scherm naar rechts swipen opent de navigation drawer.
Indien de gebruiker hieruit een categorie selecteert worden de artikelen van deze categorie gepresenteerd.
Navigatie door het artikeloverzicht gebeurt door middel van verticaal swipen.

Van een artikel wordt getoond:

- Foto
- Koptekst
- Samenvattende zin
- Bron

Artikel

Door het artikel navigeer je met scrollen (verticaal)

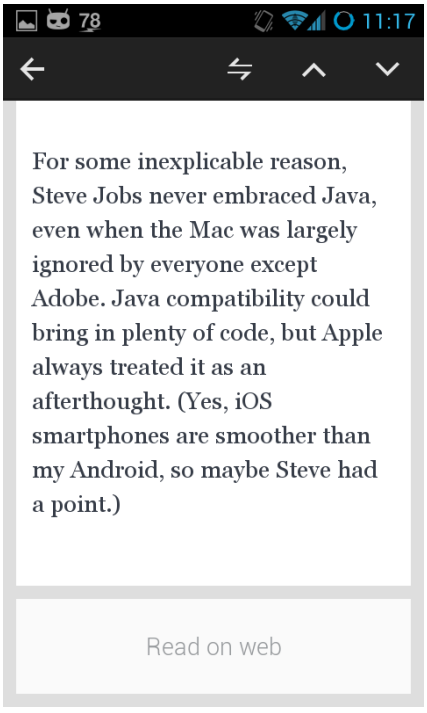
Getoond in het artikel worden:

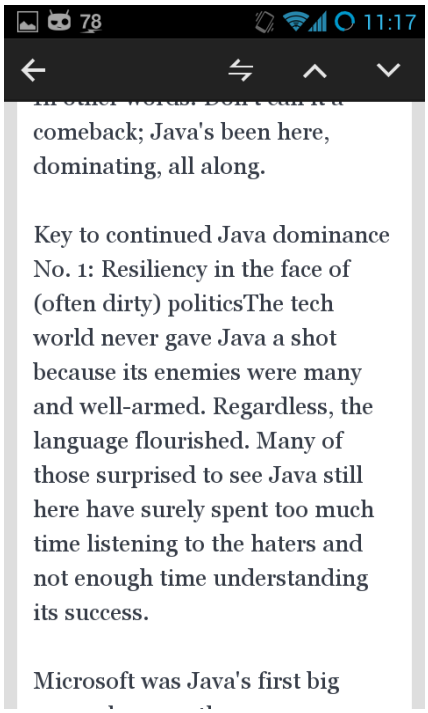
- Bron
- Titel
- Datum van plaatsing
- Tekst
- Foto

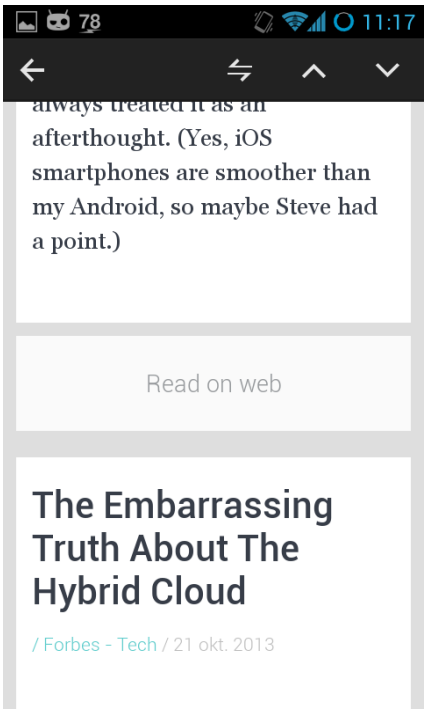
Als het einde van een artikel bereikt is kan je naar het volgende artikel doorgaan door middel van verder scrollen naar beneden. Teruggaan naar het vorige artikel gaat d.m.v. scrollen naar boven

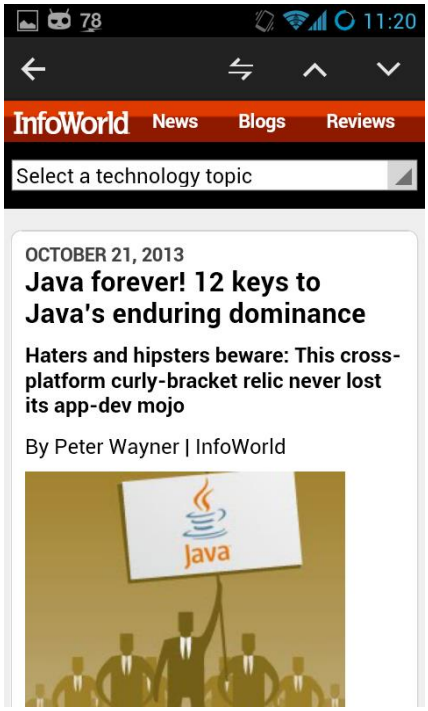
Artikel kan ook via het web bekeken worden via een button onderaan het artikel. Dit heeft geen toegevoegde waarde.


11:17
Java forever! 12 keys to Java's enduring dominance
/ Latest from Infoworld / 21 okt. 2013
InfoWorld
It's easy to forget the value of any given technology once its buzz has arced across our collective


11:17
For some inexplicable reason, Steve Jobs never embraced Java, even when the Mac was largely ignored by everyone except Adobe. Java compatibility could bring in plenty of code, but Apple always treated it as an afterthought. (Yes, iOS smartphones are smoother than my Android, so maybe Steve had a point.)
Read on web


11:17
comeback; Java's been here, dominating, all along.
Key to continued Java dominance No. 1: Resiliency in the face of (often dirty) politicsThe tech world never gave Java a shot because its enemies were many and well-armed. Regardless, the language flourished. Many of those surprised to see Java still here have surely spent too much time listening to the haters and not enough time understanding its success.
Microsoft was Java's first big


11:17
afterthought. (Yes, iOS smartphones are smoother than my Android, so maybe Steve had a point.)
Read on web
The Embarrassing Truth About The Hybrid Cloud
/ Forbes - Tech / 21 okt. 2013



Samenvatting

Een pluspunt is de mogelijkheid van doorscrollen naar volgend/vorig artikel.
Een minpunt is dat de gebruiker maar twee edities per dag krijgt.

The Guardian

The Guardian werkt met een homepage waarop maximaal 8 favoriete categorieën getoond worden. Er kunnen categorieën toegevoegd worden door deze uit een lijst te selecteren.

Andere categorieën, buiten die van de homepage, kunnen bekenen worden door de menu knop in te drukken en “All Sections” te selecteren. Hierin kan dan op basis van categorie, columnisten of zoeken een andere categorie geopend worden.

Categorieën

De categorieën worden onder elkaar getoond, met daarin per categorie de 5 meest recente artikelen. Elke categorie kan apart geopend worden voor meer artikelen.

Swipen links en rechts heeft geen functie.
Verticaal kan er gescrold worden.

Van een artikel wordt getoond:

- Foto
- Koptekst/titel
- Aantal comments (indien comments aanwezig voor dit artikel)

Artikelenoverzicht

Wanneer een categorie geopend wordt, worden indien aanwezig de subcategorieën onder elkaar getoond met hierin een aantal artikelen. Indien er geen subcategorieën aanwezig zijn worden er ongeveer 20 artikelen getoond.

Swipen links en rechts heeft geen functie.
Verticaal kan er gescrold worden.

Van een artikel wordt getoond:

- Foto
- Koptekst/titel
- Aantal comments (indien comments aanwezig voor dit artikel)

Artikel

Getoond in het artikel worden:

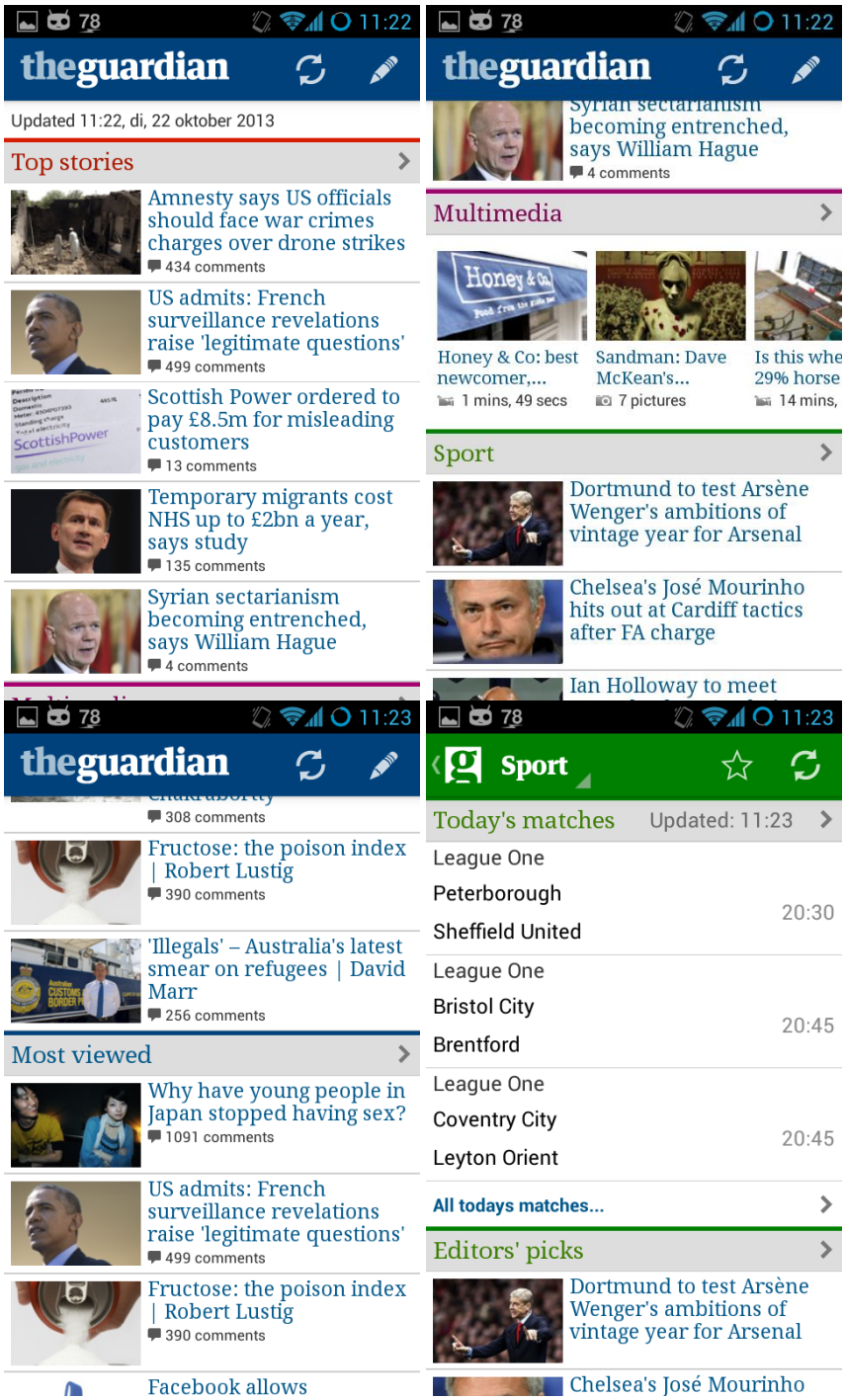
- Koptekst/Titel
- Samenvattende zin
- Auteur
- Foto
- Datum van plaatsing
- Tekst

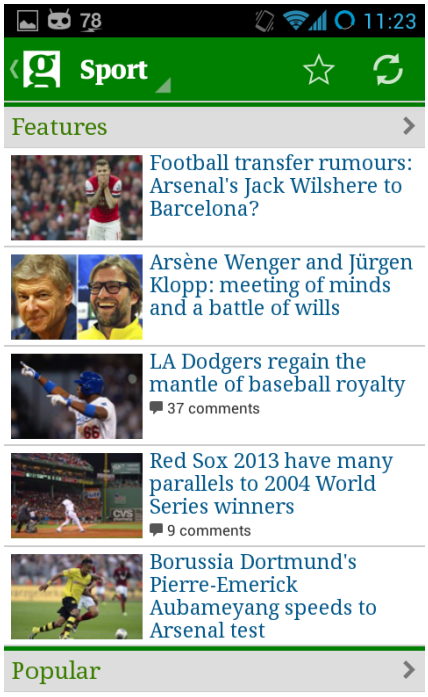
Swipe naar links geeft de gerelateerde onderwerpen(metadata) weer.
Swipe naar rechts geeft de comments weer, indien deze aanwezig zijn voor dit artikel.

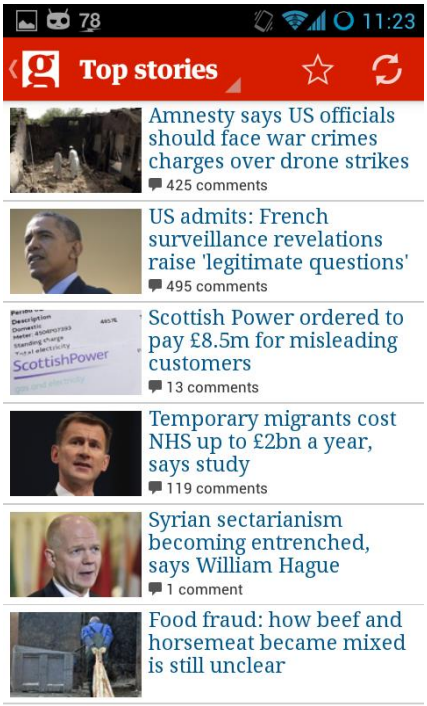
Verticaal kan er door het artikel heen gescrold worden.

Scroll naar boven om naar vorige artikel te bladeren.
Scroll naar beneden om naar volgende artikel te bladeren.

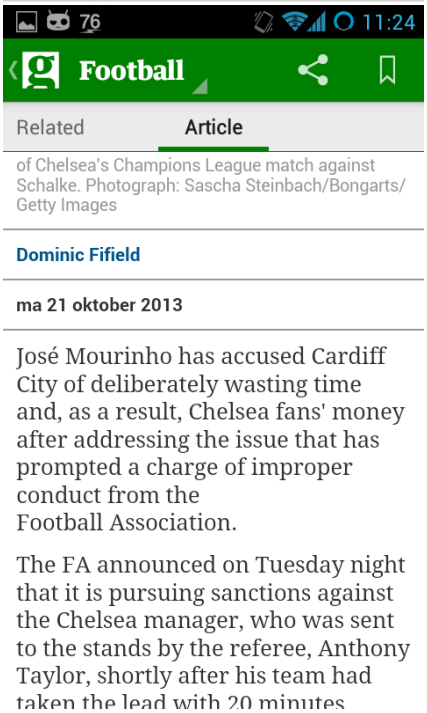
Screenshots

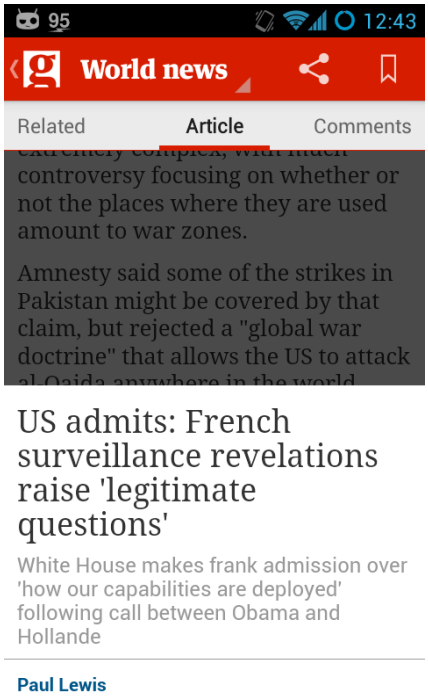



11:23
Sport
Features
Football transfer rumours: Arsenal's Jack Wilshere to Barcelona?
Arsène Wenger and Jürgen Klopp: meeting of minds and a battle of wills
LA Dodgers regain the mantle of baseball royalty
37 comments
Red Sox 2013 have many parallels to 2004 World Series winners
9 comments
Borussia Dortmund's Pierre-Emerick Aubameyang speeds to Arsenal test
Popular


11:23
Top stories
Amnesty says US officials should face war crimes charges over drone strikes
425 comments
US admits: French surveillance revelations raise 'legitimate questions'
495 comments
Scottish Power ordered to pay £8.5m for misleading customers
13 comments
Temporary migrants cost NHS up to £2bn a year, says study
119 comments
Syrian sectarianism becoming entrenched, says William Hague
1 comment
Food fraud: how beef and horsemeat became mixed is still unclear


11:24
Football
Related
Article
Chelsea's José Mourinho hits out at Cardiff tactics after FA charge
• Manager accuses Cardiff of time-wasting in 4-1 Chelsea win
• Mourinho charged with improper conduct by FA
José Mourinho was in combative form on the eve of Chelsea's Champions League match against Schalke. Photograph: Sascha Steinbach/Bongarts/


11:24
Football
Related
Article
of Chelsea's Champions League match against Schalke. Photograph: Sascha Steinbach/Bongarts/ Getty Images
Dominic Fifield
ma 21 oktober 2013
José Mourinho has accused Cardiff City of deliberately wasting time and, as a result, Chelsea fans' money after addressing the issue that has prompted a charge of improper conduct from the Football Association.
The FA announced on Tuesday night that it is pursuing sanctions against the Chelsea manager, who was sent to the stands by the referee, Anthony Taylor, shortly after his team had taken the lead with 20 minutes



Conclusie

Categorieniveau

De navigatie op categorieniveau werkt erg natuurlijk bij Flipboard. Je krijgt een goed overzicht van alle categorieën en je navigeert er gemakkelijk doorheen.

Het selecteren van een categorie is het simpelst bij Flipboard en dit werkt prettig.

Artikelenoverzichtniveau

Zite heeft de artikelen erg overzichtelijk onder elkaar geplaatst. Een nadeel van Zite is echter het scrollen door het artikelenoverzicht en dat de previews redelijk groot zijn.

Flipboard en Material werken als het ware met het omslaan van een pagina, waarbij Flipboard 1 artikel per pagina toont en Material meerdere.

Een nadeel van Material is dat je geen goed beeld krijgt waar de artikelen over gaan, vaak te weinig tekst doordat er meerdere artikelen op 1 pagina staan.

Bij Flipboard krijg je een goed idee waar het artikel over gaat als je door het artikelenoverzicht heen bladert.

Bij The Guardian krijg je een goed overzicht van ongeveer 5 artikelen tegelijk, allen hebben een foto en een koptekst. Dit werkt erg prettig.

Artikelniveau

Flipboard gebruikt het concept van pagina's omslaan voor het navigeren door een artikel. Dit werkt prettig.
Artikelenoverzicht niet overzichtelijk, ziet maar 1 preview per keer.

Zite gebruikt verticaal scrollen voor het navigeren door een artikel.

Material gebruikt verticaal scrollen voor het navigeren door een artikel. Omlaag doorscrollen aan het eind van een artikel zorgt ervoor dat het volgende artikel geopend wordt. Omhoog doorscrollen aan het begin van een artikel zorgt ervoor dat het vorige artikel geopend wordt.

The Guardian gebruikt verticaal scrollen voor het navigeren door een artikel. Omlaag doorscrollen aan het eind van een artikel zorgt ervoor dat het volgende artikel geopend wordt. Omhoog doorscrollen aan het begin van een artikel zorgt ervoor dat het vorige artikel geopend wordt.

Het openen van volgende/vorige artikelen vanuit het huidige artikel werkt erg prettig.