Systeemdossier

Social Media @ Thales

Project Persistent

EIN1Vu2

Erik Sijssens

Wim van der Zijden

Inhoudsopgave

1 Inleiding

Dit is het systeemdossier voor de ontwikkeling van de app voor het project Persistent. Tijdens dit project ontwikkelen we een Android app voor Thales om het haar personeel en bezoekers makkelijker te maken onderling afspraken te maken voor activiteiten zoals lezingen en sportevenementen.

Dit systeemdossier bevat de volgende onderdelen:

- Requirements
- Product Backlog
- Ontwerprapport
- Technische Documentatie

Het testplan en testrapport zijn integraal opgenomen in het product backlog.

2 Requirements

Wat moet er zeker gerealiseerd worden? (Must)

- Must: Er is een grafische representatie van de applicatie.
- Must: Er moet een framework zijn voor de applicatie.
- Must: Een gebruiker moet een account kunnen aanmaken
- Must: Een gebruiker moet kunnen in- en uitloggen.
- Must: Een gebruiker moet een lijst van ruimtes kunnen zien.
- Must: Een medewerker moet een event kunnen aanmaken
- Must Een gebruiker moet een lijst van events kunnen zien waarop de gebruiker zich kan inschrijven.
- Must: een medewerker moet zich kunnen abonneren op event categorieën.
- Must: Een gebruiker moet een notificatie krijgen van een event zodra hij erop geabonneerd is.
- Must: Een gebruiker met een ander zijn/haar account kunnen zoeken.
- Must: Er moet een admin account komen die nieuwe locaties kan toevoegen.
- Must: Een gebruiker moet de gegeven van zijn account kunnen aanpassen en de instellingen kunnen veranderen.

Waar kan het project eigenlijk niet zonder? (Should)

- Should: Bij het maken van een account zou er een verificatie beveiliging in de applicatie moeten zitten.
- Should: Een gebruiker zou een status moeten kunnen aangeven.
- Should: een medewerker zou moeten kunnen aangeven waar hij is.
- Should: een gebruiker zou een andere gebruiker een verzoek moeten sturen voor een zijn locatie
- Should: Er zou een filter optie moeten zijn bij het doorzoeken van de events.
- Should: Gastaccount moet worden verwijderd na meer dan 3 dagen inactief te zijn.

Wat zouden we graag in het project willen zien? (Could)

- Could: Locatie bepaling van mensen door middel van QR-codes.

3 Product backlog

3.1 Stories Sprint 1 (18 Story points)

Story	Als ontwikkelaar wil ik een grafisch ontwerp van de app, zodat op basis hiervan de app vormgegeven kan worden.
Priority	10
Story Points	2
Omschrijving	Een grafisch ontwerp van de app dat aangeeft hoe de navigatie moet plaatsvinden en hoe schermen eruit zien.
How to demo	Laat zien hoe de navigatie werkt en hoe schermen en velden eruit zien.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	21 mei 2014

Story	Als ontwikkelaar wil ik een framework voor de app, zodat er een basis is voor de ontwikkeling van functionaliteit.
Priority	15
Story Points	3
Omschrijving	Het framework bevat de eerder ontworpen UI en basisfunctionaliteit voor de database en de structuur van de Android-activities.
How to demo	Laat een kale werkende app met UI zien die een database gebruikt.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	22 mei 2014

Story	Als gebruiker wil ik een account aan kunnen aanmaken, zodat ik hiermee kan inloggen.
Priority	20
Story Points	3
Omschrijving	Een account bevat altijd een naam en email. Als er een Thales email gebruikt wordt word het een medewerker account, anders wordt het een gast account. Account moet opgeslagen worden in de database.
How to demo	Klik “Register” vanuit het startscherm. Vul een naam, email en paswoord in. Klik OK. Er verschijnt nu een bericht “Registration successful” en je wordt automatisch ingelogd.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	26 mei 2014

Story	Als gebruiker wil ik kunnen in- en uitloggen, zodat ik mijn geregistreerde account kan gebruiken.
Priority	25
Story Points	2
Omschrijving	Een gebruiker logt in met het email en wachtwoord waarmee hij zich eerder heeft geregistreerd (zie story 20). Als hij eenmaal ingelogd is, moet de app dit onthouden.
How to demo	Vul het e-mail en paswoord in in het startscherm en klik ok. De navigatie wordt geactiveerd en je wordt doorgelinkt naar het eerste scherm van de app. Sluit de app af en start hem opnieuw. Je wordt nu automatisch ingelogd.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	3 juni 2014

Story	Als gebruiker wil ik een lijst met ruimtes en capaciteit kunnen zien, zodat ik kan zien welke ruimtes er allemaal in het systeem staan.
Priority	30
Story Points	2
Omschrijving	Ruimtes zijn vergaderzalen en sportruimtes. Elke ruimte heeft een maximale bezetting. De ruimtes zijn opgeslagen in de database.
How to demo	Navigeer naar het scherm waar het overzicht van ruimtes is en constateer dat hier een lijst met ruimtes staat. Voeg met behulp van SQL een ruimte toe aan de database en laat zien dat deze aan de lijst toegevoegd wordt.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	3 juni 2014

Story	Als Thales medewerker wil ik een event aan kunnen maken, zodat ik de app kan gebruiken om events te organiseren.
Priority	40
Story Points	3
Omschrijving	Een event moet een naam en categorie hebben. Categorieën zijn pizza-sessies, sportactiviteiten en overige. Een event heeft een datum en tijd hebben. Een event kan een event een minimum en maximum aantal deelnemers hebben.
How to demo	Klik vanuit het overzichtsscherm voor Events op de “Create Event” knop bovenin het scherm. Er verschijnt een dialoog waarop de benodigde informatie kan worden ingevuld. Als er onvolledige of foutieve informatie wordt toegevoegd, dan komt hier een inline melding van nadat op OK wordt gedrukt. Is alle informatie toegestaan, dan verschijnt de melding: “Event created”.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	3 juni 20014

Story	Als gebruiker wil ik een lijst van events kunnen bekijken, zodat ik op de hoogte ben van alle events in het systeem.
Priority	50
Story Points	3
Omschrijving	Een event heeft een naam, een type, een datum, een begintijd, een eindtijd, een minimum aantal deelnemers en een maximum aantal deelnemers. De events worden opgeslagen in de database en moeten getoond worden in de app.
How to demo	Navigeer naar het scherm waar het overzicht van events is en constateer dat hier een lijst met events staat. Voeg met behulp van de “Create Event”-knop een ruimte toe aan de database en laat zien dat deze aan de lijst toegevoegd wordt.
Status	Done (Test OK)
Opgenomen in	Sprint 1
Testdatum	4 juni 2014

3.2 Stories Sprint 2 (19 Story points)

Story	Als gebruiker wil ik me kunnen aan- en afmelden voor een event, zodat ik aan deze events kan deelnemen of mijn deelname kan annuleren.
Priority	60
Story Points	2
Omschrijving	Er moet een knop komen per event op het scherm “Events”, waarmee een gebruiker zich kan aan- en afmelden voor een event.
How to demo	Ga naar de lijst met events. Klik op “Participate” en constateer dat er nu wordt aangegeven dat je je hebt geregistreerd voor dit event. Klik nu voor hetzelfde event op “Cancel Participation” en constateer dat er nu wordt aangegeven dat je je niet hebt geregistreerd voor dit event.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/22/14

Story	Als gebruiker wil ik de lijst met events kunnen filteren op events die open staan voor inschrijving voor een bepaalde periode, zodat ik kan zien welke events ik nog aan mee kan doen.
Priority	105
Story Points	3
Omschrijving	Er moet een knop komen op het scherm “Events” waarmee een filter ingesteld kan worden waarmee gefilterd kan worden op events die open staan voor inschrijving en bepaalde tijdsperioden. Ook moet het makkelijk zijn in de toekomst extra filteropties toe te voegen.
How to demo	Ga naar het scherm “Events”. Klik op de knop “Filter”. Stel een filter in. Constateer dat de Events beperkt worden op basis van deze filter. Klik nu op “Edit Filter” en werk de filter bij. Constateer dat de wijziging verwerkt wordt. Klik nu op de knop “Delete Filter” en constateer dat de filter verwijderd wordt en alle events weer zichtbaar zijn.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/25/14

Story	Als gebruiker wil ik mijn accountgegevens kunnen inzien en wijzigen, zodat ik controle heb over mijn gegevens.
Priority	110
Story Points	3
Omschrijving	Er komt een scherm “Profiel”. Op dit scherm kan je je profielfoto, je naam en je wachtwoord wijzigen.
How to demo	Navigeer naar de pagina “profiel”. Klik op je foto en wijzig je foto. Klik op het “Edit”-icoontje achter je naam en wijzig je naam. Klik op het “Edit”-icoontje achter je wachtwoord en wijzig je wachtwoord. Constateer dat de wijzigingen persistent bewaard blijven, ook na afsluiten app en opnieuw opstarten.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/05/14

Story	Als gebruiker wil ik kunnen zien wie zich allemaal voor een bepaald event heeft ingeschreven, zodat ik meer informatie heb over dit event.
Priority	115
Story Points	2
Omschrijving	In het scherm “Events” komt een knop waarmee een lijst opgehaald kan worden met de ingeschreven gebruikers.
How to demo	Ga naar het scherm “Events”, klik op de knop ingeschreven gebruikers bij een bepaald Event. Constateer dat hier de juiste informatie wordt weergegeven.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/22/14

Story	Als gebruiker wil ik kunnen zien welke ruimtes beschikbaar zijn op een bepaald moment.
Priority	120
Story Points	3
Omschrijving	In het scherm met ruimte komt een extra veld waarmee de lijst beperkt kan worden op ruimte die beschikbaar zijn voor een bepaald tijdsinterval.
How to demo	Maak een event aan voor een bepaald tijdstip met een bepaalde ruimte. Laat zien dat deze ruimte voor dit tijdsinterval niet meer weergegeven wordt in de lijst met ruimtes.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/12/14

Story	Als Thales medewerker wil ik bij het aanmaken van een event een ruimte kunnen selecteren, zodat deze ruimte gereserveerd wordt voor dit event.
Priority	130
Story Points	3
Omschrijving	Een ruimte kan op één bepaald tijdstip maar door één event gereserveerd worden. Deze reservering heeft invloed op de weergegeven beschikbaarheid van een ruimte.
How to demo	Klik “Create Event” op het scherm “Events”. Selecteer een datum, begintijd en eindtijp. Er verschijnt nu een dropdown met beschikbare ruimtes voor dat tijdsinterval. Selecteer één van die ruimtes. Klik OK. Maak nog een Event aan voor dezelfde datum en tijd en constateer dat de zojuist gekozen ruimte voor dit event niet meer beschikbaar is.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/12/14

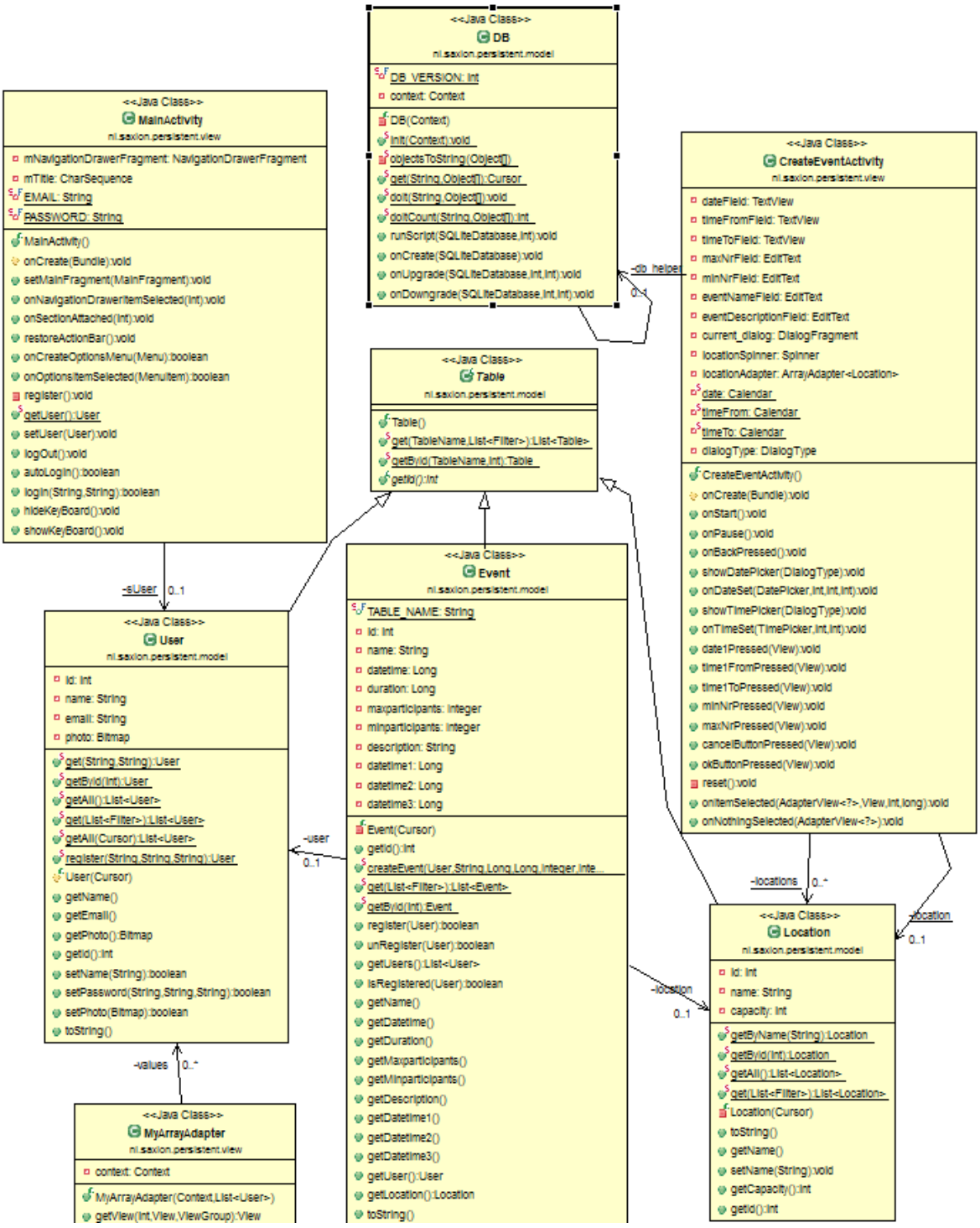
Story	Als gebruiker wil ik een user manual zodat ik kan lezen hoe de app werkt en hoe ik hem kan gebruiken
Priority	140
Story Points	3
Omschrijving	Een document waarin stap voor stap alle functionaliteit van de app wordt uitgewerkt.
How to demo	We kunnen met de app in de ene hand en de user manual in de andere hand de manual stap voor stap doorlopen en controleren of alles klopt en of het een totaalbeeld van de app geeft.
Status	Done (Test OK)
Opgenomen in	Sprint 2
Testdatum	06/25/14

Story	Als gebruiker wil ik alle lijsten in de app kunnen filteren op verschillende waarden.
Priority	150
Story Points	4
Omschrijving	Bij elke lijst moet de knop “Search Filter” leiden naar een scherm waarmee filters ingesteld kunnen worden voor de lijst. Dit betreft de lijsten “Users, Locations en Events”
How to demo	Ga naar het scherm “Events”. Klik op de knop “Filter”. Stel een filter in. Doe “Back” en constateer dat de Events beperkt worden op basis van deze filter. Ga terug naar het filter overzicht en klik nu op de knop “Delete Filter”. Ga weer “Back” en constateer dat de filter verwijderd wordt en alle events weer zichtbaar zijn. Test dit vervolgens ook voor de schermen “User” en “Location” en stel vast dat op alle relevante kolommen gefilterd kan worden.
Status	Done (Test OK)
Opgenomen in	
Testdatum	06/25/14

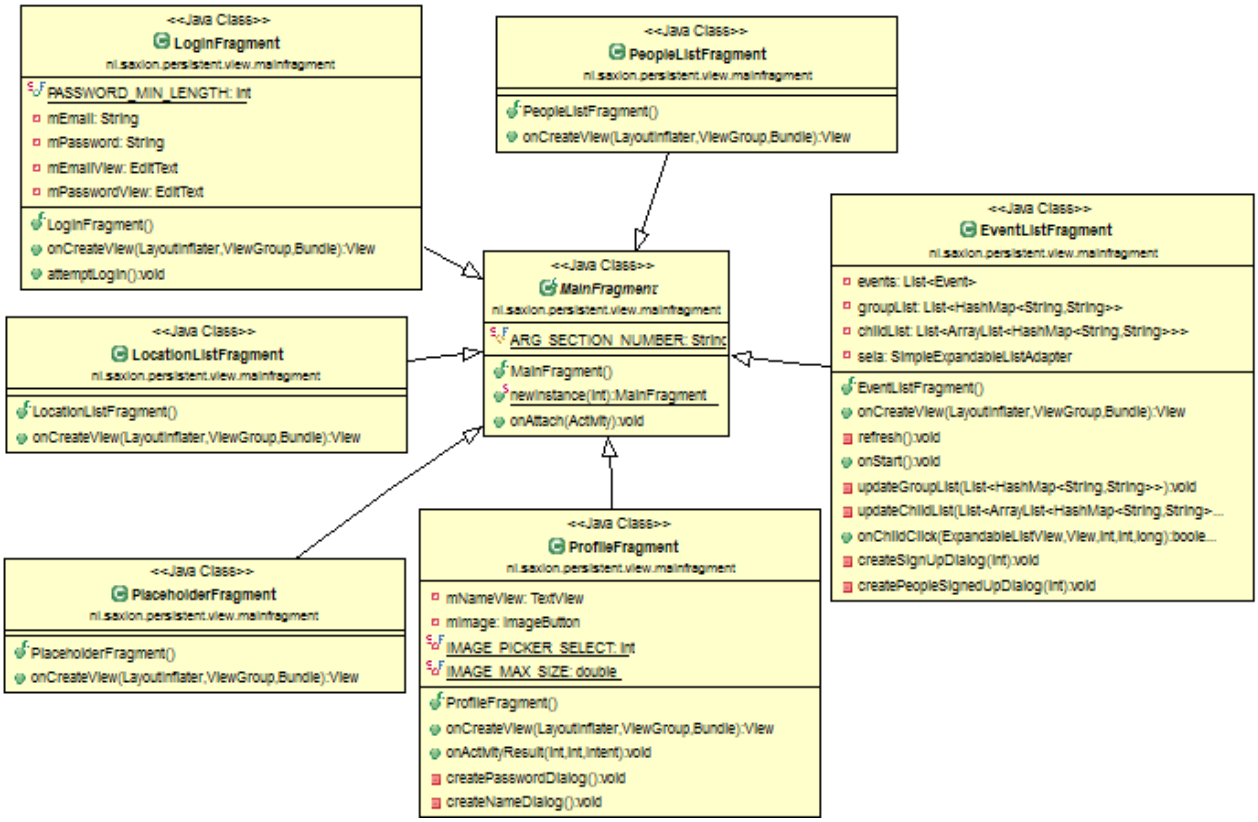
4 Ontwerprapport

In dit hoofdstuk wordt een overzicht gegeven van de klassendiagrammen voor de android code en een datamodel van het ontwerp van de gebruikte database.

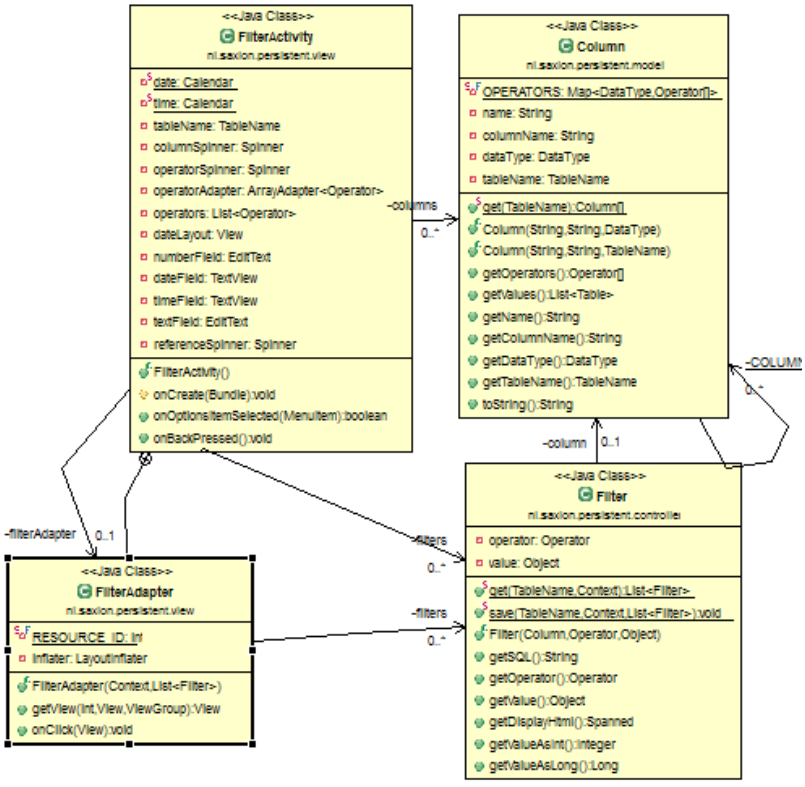
4.1 Basisklassen



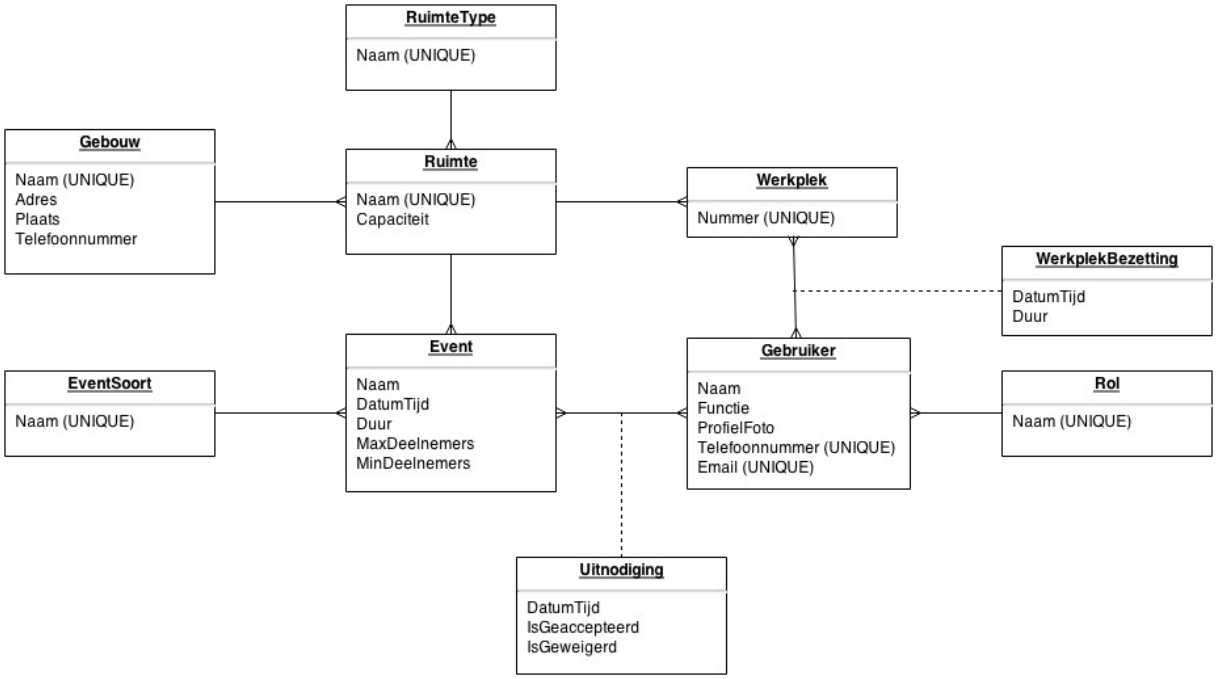
4.2 Fragment Klassen



4.3 Filter klassen



4.4 Datamodel



4.5 Applicatie Architectuur

In de app maken wij gebruik van de standaard in Android aanwezige embedded SQLite database. Omdat deze database een geintegreerd onderdeel is van het android besturingssysteem is het uitvoeren van queries zeer snel. Het nadeel van SQLite is dat het een redelijk primitieve database is met beperkte functionaliteit. Zo is het niet mogelijk om “Full outer joins” te doen en om na het aanmaken van een tabel nog constraints toe te voegen. Hiervoor moet de tabel gedropt en gerecreate worden. Dit is slechts een greep uit de beperkingen van SQLite. Voor meer informatie over de database, zie 5.7 .

Aangezien de app slechts bestaat uit een app en een lokale database, lijkt het ons niet zinvol om hier een model van de applicatie architectuur toe te voegen.

5 Technische Documentatie

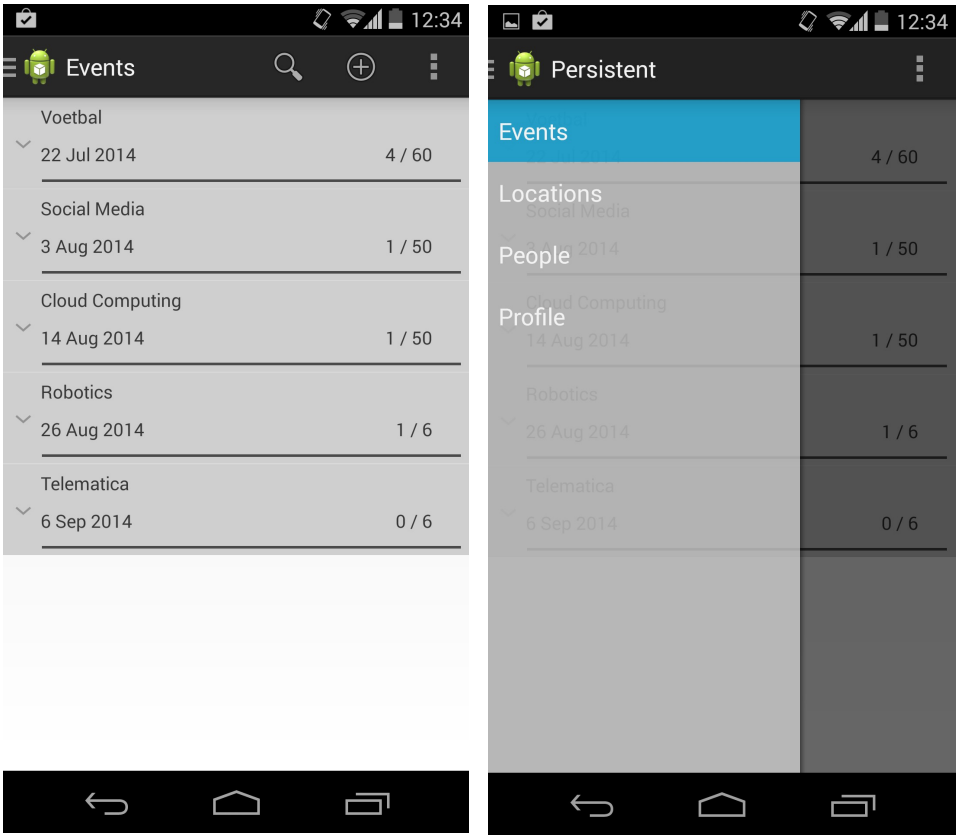
Dit hoofdstuk geeft een overzicht van het technisch ontwerp van de app. Er wordt aandacht besteed aan de gemaakte keuzes op het gebied van de structuur van de android code, de database en de user interface.

5.1 User Interface

Voor de UI hebben we gekozen voor een Navigation Drawer. Hierdoor kan de gebruiker tussen de verschillende onderdelen van de applicatie navigeren, zonder dat de knoppen ruimte innemen op het scherm. We hebben de applicatie onderverdeeld in vier onderdelen:

- Events: de lijst met evenementen
- Locations: de lijst met ruimtes
- People: de lijst met gebruikers
- Profile: De plek waar je instellingen van je account kan wijzigen.

Voor knoppen bij de lijsten die niet direct met een onderdeel van de lijst te maken hebben, gebruiken we knoppen in de action bar. Hierdoor zitten ze de lijst niet in de weg en staan ze altijd boven aan je scherm ook al scroll je naar beneden door de lijst. Verder heb je nog de schermen Login, Register, Create Event, Filter Event en Help.



Screenshot van de Event List. Links zonder Navigation Drawer open en rechts met.

5.2 Manifest

In het bestand AndroidManifest.xml worden een aantal instellingen voor de gehele applicatie geregeld. Zo kan je hier instellen welke rechten de app moet hebben (bijvoorbeeld internet toegang), welke activity moet worden gestart als de app start, en vanaf welke API de app moet werken.

Op deze plek is de keuze gemaakt voor compatibility vanaf API 15, oftwel versie Ice Cream Sandwich (4.0.3). Volgens een bron van Android zelf geeft dit dekking voor zo'n 85% van de Android toestellen. Voor de Galaxy S3 toestellen die bij Thales gebruikt worden is dit zeker genoeg, en ook de meeste gastgebruikers zullen hier genoeg aan hebben. Als launcher activity staat de MainActivity ingesteld, die hieronder nader wordt toegelicht.

Ook is dit de plek waar het launcher icon voor de app ingesteld wordt. Hier hebben wij het logo van Thales voor gebruikt.

5.3 Packages

Er zijn twee hoofdpackages in de source code, nl.saxion.persistent.model en nl.saxion.persistent.view. In het Model package staan de objecten zoals die over het algemeen ook in de database voorkomen, vaak onder dezelfde naam. In het view package staan de views en fragments die erven van de standaard Android classes Activity en Fragment.

5.4 MainActivity

In deze activity zal vrijwel alle functionaliteit worden ingebouwd. Deze activity heeft twee fragments: NavigationDrawerFragment en een dynamisch container fragment. Het NavigationDrawerFragment is een menu dat gebruikt wordt om verschillende container fragments in te laden in het container fragment. Deze container fragments erven allemaal van MainFragment. MainFragment bevat een factory method getInstance(int) die gebruikt wordt om de verschillende implementaties ervan te instantiëren.

De mainActivity bevat ook een Action Bar. Voor de Action Bar zijn verschillende xml layouts: event.xml, global.xml, login.xml en main.xml. Deze layout bestanden worden geladen naar gelang de context. Zo word login.xml geladen als het container fragment een LoginFragment bevat.

5.5 FilterActivity

5.5.1 Structuur

Om lijsten te kunnen filteren is er een FilterActivity gebouwd. Deze Activity is zoveel mogelijk generiek opgezet, zodat het gemakkelijk toe te passen is op alle mogelijke lijsten in de app.

Bij het starten van de FilterActivity wordt de TableName meegegeven. Dit is een enum op Table, de abstracte superclass van alle database objecten. Elke table heeft een aantal searchable columns. Deze worden geïnitialiseerd in een static blok in de class Column. Deze columns worden ingeladen in de FilterActivity en op basis van het datatype wordt een correct inputveld getoond. Er zijn 5 verschillende datatypes gespecificeerd:

- Text
- Number
- Boolean

- Date
- Reference

DataType is een enum op Column. Elk Datatype heeft zijn eigen set van Operators. Ook deze operators worden ingeladen in een static blok op Column. Operator is een enum op de Filter class met twee string-representaties: een sql-representatie en een human-readable representatie.

Met de knop Create op FilterActivity wordt een nieuwe Filter worden aangemaakt en direct weergegeven in het lijstje eronder. Deze lijst wordt opgeslagen en toegepast bij het laden van de lijst in de onderliggende activity. Een filter heeft een getSql() methode die op basis van het datatype de juiste sql geeft om aan de query te plakken. Ook heeft een filter een getDisplayHtml() methode om een filter op een gebruiksvriendelijke manier weer te geven in het lijstje met filters.

5.5.2 Saving as Json

Elke lijst in de app (Event, User, Location),heeft een lijst met filters die gebruikt wordt om de resultaten te filteren. Er is voor gekozen om deze filters op te slaan in de SharedPreferences, zodat de gebruiker niet steeds opnieuw dezelfde filters hoeft in te stellen.

Een complicerende factor hierbij was dat in de SharedPreferences alleen primitieve waarden opgeslagen kunnen worden (String, int, long, etc). Het out-of-the-box opslaan van een lijst met custom objecten is dus niet mogelijk. Daarom is ervoor gekozen om met behulp van de open source library Gson (v2.2.4) de lijst met filters te converteren naar Json en deze vervolgens als String op te slaan in de Shared Preferences. Bij het inladen wordt Gson ook weer gebruikt om het object te creëren vanuit de opgeslagen Json.

Deze methode heeft als nadeel dat er niet altijd vanuit kan worden gegaan dat een object in precies dezelfde staat hersteld wordt. Een voorbeeld hiervan is de instance variabele “Object value” op Filter. In deze variabele kunnen verschillende objecten opgeslagen zijn, afhankelijk van het datatype van de betreffende Column: Long, Integer, etc.. Bij het terugconverteren wordt door Json niet onthouden wat de exacte class van het object was. De volgende code zou dan ook een classcastexception opleveren, zelfs als de oorspronkelijke class van value wel degelijk Integer was:

```
public Integer getValueAsInt() {
	return (Integer) value;
}
```

In plaats hiervan moet de volgende code gebruikt worden voor deze conversie:

```
public Integer getValueAsInt() {
	return ((Number) value).intValue();
}
```

Een ander probleem dat kan optreden is dat wijzigingen aan het de filter-class kunnen leiden tot crashes als er nog opgeslagen instanties zijn van de verouderde versie van deze class. Dit probleem is te verhelpen door een volledige uninstall en re-install van de app.

5.6 CreateEventActivity

Deze activity wordt gebruikt om nieuwe events aan te maken. Als de datum, de begintijd en de eindtijd geselecteerd zijn, verschijnt een lijst met beschikbare locaties. Deze locaties worden beperkt op beschikbaarheid met behulp van de volgende query:

```
SELECT name, capacity, id FROM Locations l
       WHERE NOT EXISTS
              (SELECT * FROM Event
              WHERE location_id = l.id
              AND ((? > datetime AND ? <= datetime_to)
              OR (? >= datetime AND ? < datetime_to)
              OR (? <= datetime AND ? >= datetime_to)));
```

Hierbij zijn de parameters (?) respectievelijk to,to,from,from,from,to. Hierbij is “from” de datum gecombineerd met de vanaf-tijd, en “to” de datum gecombineerd met de tot-tijd.

5.7 Database

We hebben ervoor gekozen de standaard Android SQLite DB te gebruiken. In de uiteindelijke productieversie van deze app zal gebruik gemaakt moeten worden van een externe DB die via een API bereikt kan worden, maar om dit te ontwikkelen was in dit project geen tijd.

De SQLite versie die in Androidtoestellen aanwezig is verschilt per API. Voor de API 15 is dit SQLITE 3.7.4, voor API 16 tot en met 19 is dit 3.7.11. Tussen deze versies zitten subtiele verschillen. Eén van die verschillen is dat in 3.7.4 de volgende syntax níet geldig is, en in 3.7.11 wél:

```
INSERT INTO table1(col1,col2) VALUES
('value','value'),
('value','value');
```

In 3.7.4 mag met een dergelijk statement maar één rij tegelijk inserted worden.

Om vanuit code gemakkelijk de database te kunnen gebruiken is de class DB. Deze class erft van de standaard Android class SQLiteOpenHelper. DB is ontworpen volgens het singleton design pattern. De methode init creëert een instantie welke vervolgens door alle methodes gebruikt wordt. Alle public methods zijn static en gebruiken deze statisch opgeslagen instantie van DB.

Om de Database voor eerste gebruik te initialiseren is er een initialisatiescript. Dit is als raw resource opgeslagen op de volgende locatie: res/raw/db_create.sql. Via de code kan het resource id hiervan opgehaald worden met R.raw.db_create. Dit script bevat create table statements, constraints en insert statements voor testdata. Als er wijzigingen in dit script worden gemaakt, dan dient het versienummer opgehoogd te worden. Dit zorgt ervoor dat dit script opnieuw gedraaid wordt. Dit heeft overigens wel als neveneffect dat alle data in de database verwijderd wordt. Dit is dan ook een tijdelijke oplossing, die niet in de productie behouden moet blijven.

Verder heeft de class DB een aantal convenience methods: “Cursor get(String sql,Object... parameters)”, “void doIt(String sql,Object... parameters)” en “int doItCount(String sql,Object... parameters)”. De eerste is voor het uitvoeren van select statements en geeft een cursor met resultaten terug. De tweede en de derde zijn voor het uitvoeren van insert, update en delete statements, waarbij de derde daarbij als resultaat het aantal “affected rows” teruggeeft.