Bachelorproef AMuRate: Voortgangsrapport
3e Bachelor Computerwetenschappen

David Sverdlov
dsverdlo@vub.ac.be

February 19, 2013

Contents

1 Inleiding

1.1 Doel

De uitdaging van dit bachelorproject is om een applicatie te ontwikkelen voor het mobiele platform Android. De applicatie moet gebruikers in staat stellen om muzieknummers op te zoeken en een beoordeling te kunnen geven. Die scores worden dan opgestuurd naar een data-collection server met een database, waar later recommendeer algoritmen op zullen werken. (De implementatie daarvan valt niet binnen de scope van dit project.)
De promotor van dit project is Dr. Peter Vranckx en mee begeleidt door Maarten Deville.

1.2 Projectbeschrijving

Gebruikers moeten eerst en vooral muziek kunnen opzoeken. Hier zal een scherm voor zorgen, waar men de zoekgegevens kan invullen. Er is een veld voor de titel van een lied in te vullen, en een veld voor de artiest- of groepsnaam. Beide velden hoeven niet tesamen ingevuld te zijn, het volstaat dat er maar een veld ingevuld is. Er zal een melding gegeven worden wanneer er op de zoek-knop geduwd werd zonder dat er iets van zoekgegevens werd opgegeven.

Een hulp-knop informeert mensen over het (correct) gebruik van de applicatie. Dit gebeurt door een pop up en kan gesloten worden door op OK te klikken.

Wanneer er geen werkende data connectie is, zij dat de data-connectie niet aan staat, of dat er geen beschikbaar netwerk in de buurt is, zal de gebruiker een melding krijgen dat hij een werkende data verbinding nodig heeft eer informatie gedownload kan worden.

Wanneer een gebruiker beschikt over een actieve data verbinding en minstens één zoekterm ingevuld heeft, zal de applicatie de gegevens naar de server sturen en een antwoord ontvangen. Het resultaat van de oproep wordt op het scherm getoond en de gebruiker krijgt enkele nieuwe opties.

Wanneer men kiest om de resultaten te bekijken, wordt er een lijst getoond met alle opties. Door op een van de resultaten te klikken, kan er naar het volgende scherm gegaan worden: een gedetailleerde pagina met meer informatie dan de beknopte optie waarop geklikt werd.

Het gedetailleerde scherm van een lied geeft wat triviale informatie over het lied in kwestie, zoals de duratie, album, artiest, ... Daarnaast nog een

afbeelding van het album[1]. Indien er meer album informatie beschikbaar is, zullen de velden die informatie geven over het album een andere kleur krijgen en dienen als knop om naar de gedetailleerde pagina voor een album te gaan.
Als laatste is er een deel van het scherm bedeeld aan het score systeem. Naast de mogelijkheid om zelf eenmalig een score te geven (tussen 0 en 5, met stappen van een halve precisie), wordt ook de gemiddelde score weergegeven van alle scores voor dat lied (en het aantal scores waar het gemiddelde van berekend werd).

Het gedetailleerde scherm voor een album toont enkele informatieve zaken van het album en alle liedjes van het album. De gebruiker kan op elk lied klikken om naar het scherm te navigeren met meer informatie en/of een score te geven.

Wanneer een score toegediend wordt en er een werkende data verbinding is, moet de score niet alleen in de lokale database toegevoegd worden, maar ook naar de server gestuurd worden. De scores worden geidentificeerd via het unieke Android device nummer van het GSM toestel. Dit is een 64-bit hex string.

Als er geen internet connectie zou zijn op het moment dat een score gegeven wordt, kan de lokale database al geupdated worden, maar kan de score niet direct naar de data-collection server gestuurd worden. De (lokaal) unieke ID van de score wordt opgeslagen in een buffer, die geleegd wordt wanneer de verbindingstoestand zou veranderen. Wanneer er terug een data verbinding tot stand zou komen, wordt voor elke ID in de buffer de score uit de lokale database gelezen en naar de server gestuurd.

1.3 Afkortingen en definities

1.3.1 API

API staat voor Application Programming Interface en zorgt voor de communicatie tussen programmas, door de scheiding te vormen tussen verschillende lagen van abstracties.

1.3.2 XML/JSON

XML staat voor Extensible Markup Language en is een van de meest gebruikte opmaaktalen, die gestructureerde gegevens kunnen omzetten in platte tekst. (Om het zo makkelijk(er) door te kunnen sturen.)

[1] Indien er een afbeeldings URL beschikbaar is, anders een standaard blanco afbeelding.

JSON is aan afkorting van JavaScript Object Notation en is een alternatieve simpele manier om objecten voor te stellen als platte tekst.

1.3.3 HTTP

HTTP staat voor HyperText Transfer Protocol en is het medium tussen een webbrowser en een webserver. Die communicatie gebeurt door middel van URLs, die verwijzen naar 'iets' op een of andere webserver.

1.3.4 GUI

GUI staat voor Graphical User Interface en is een visuele vormgeving van een programma, dat door middel van knoppen, afbeeldingen, ... gebruikers toelaat om op een gebruiksvriendelijkere manier met de applicatie om te gaan .

2 Achtergrond

2.1 Last.fm/API: REST

Om informatie (over muziek in dit geval) op te kunnen zoeken, moet er gebruik gemaakt worden van een online database met een openbare en hanteerbare API. *Last.fm* is een muziek recommendation service met een enorme online muziek database, die een gratis (lees: voor geregistreerde gebruikers) API aanbiedt, die iedereen toelaat om mobiele/desktop programmas of web services te bouwen met hun data.
Die procedure verloopt als volgt: om data uit de database te halen moet er een call (oproep) gestuurd worden. Deze gebeurt via HTTP GET naar de Last.fm server die op zijn beurt antwoordt met een XML object (of JSON op aanvraag). De methode van operatie die hier gebruikt wordt heet 'REST'.
De API verschaft tientallen methodes zoals artist.search, artist.getTopAlbums, artist.getTopTracks, track.getInfo, track.search, track.getTags, album.search, album.getBuyLinks, ... De search en getInfo oproepen zullen het meeste gebruikt worden in dit project.

2.2 REST architectuur

REST staat voor Representational State Transfer en is een type van architectuur voor bepaalde gedistribueerde systemen (zoals het World Wide Web), waarin een duidelijke onderscheiding wordt gemaakt tussen client en server. Clienten kunnen oproepen doen naar de server, die de oproepen analyseert, een antwoord formuleert, en dat antwoord terug naar de klant stuurt. REST maakt gebruik van werkwoorden en zelfstandige naamwoorden om de oproepen ook leesbaar voor mensen te maken. De standaard werkwoorden

zijn GET, POST, PUT en DELETE, om data te kunnen verkrijgen, wijzigen of opsturen.

2.3 Website

In dit project wordt er gebruik gemaakt van Git als versiebeheersysteem, omdat het een gratis, eenvoudige en betrouwbare manier is om de broncode te beheren. Om updates in het project naar de repository te sturen wordt er gebruikt gemaakt van Github voor Windows. Nog een voordeel van deze keuze is dat Github de mogelijkheid biedt om snel en simpel websites te maken(en te behoren) voor de bestaande projecten. Zo werd er een kleine website voor dit project opgesteld, die terug te vinden is op: `http://dsverdlo.github.com/AMuRate/`. Hier kan men de open-source broncode bekijken, het logboek of dit document raadplegen en de applicatie (.apk) downloaden voor Android 4.0 toestellen.

2.4 Naam en logo

De naam van een project geeft meestal een kleine beschrijving van wat de applicatie doet/ waar hij voor dient. AMuRate komt van "Android Music Rating". Omdat het project vooral geïmplementeerd zou worden via een Android emulator en simpelweg "AMR" te kort en onduidelijk zou zijn, werd er gekozen voor AMuRate. Het logo (zoals te zien op de voorpagina) is een compositie van een 5-ster in een bol die iets weg heeft van een "dragon ball" (uit de Japanse animatieserie DragonBall Z). In DBZ bestaan er maar 7 bollen op de hele wereld en wanneer die allemaal verzameld zijn, mag de beheerder een wens doen. Een beetje gelijkaardig aan hoe elke artiest hoopt 5 sterren (/ 7 dragon balls) te verzamelen. In de ster zelf zijn drie letters gekleurd ("AMR"), naar de naam van de applicatie.

3 Vereisten en voortgang

Voor de voortgang van dit project besproken wordt, eerst een opsomming van alle vereisten en huidige status.

ID	Side	Beschrijving	Status
1	Client	Titel en artiest kunnen invullen	done
2	Client	Enkel op titel zoeken	done
3	Client	Enkel op artiest opzoeken	in progress
4	Client	Gedetailleerd lied weergeven	done
5	Client	Gedetailleerd artiest weergeven	in progress
6	Client	Gedetailleerd album weergeven	done
7	Client	Scores beheren in lokale database	done
8	Client	Geschiedenis beheren in database	in progress
9	Client	Geschiedenis weergeven op scherm	in progress
10	Client	Scores opsturen naar server	NYI
11	Client	Gem. scores ontvangen van server	NYI
12	Server	Scores ontvangen van client-side	NYI
13	Server	Scores in server database opslagen	NYI
14	Server	Gem. scores halen uit ext database	NYI

Tabel 1: Vereisten van het project met uitleg en implementatiestatus.

Zoals in de projectbeschrijving reeds gezegd werd, bestaat het project uit twee afzonderlijke delen, die verbonden kunnen worden door een internetverbinding. Aan de client-kant is er de applicatie op Android, en aan de server-kant een data-collection server die scores verzameld en geregeld gebruikt voor recommendeer algoritmen. De server-kant is een kleiner packet dan de client-kant, maar niet minder belangrijk.

Bovenstaande tabel toont dat de applicatie langs de client-kant haast volledig af is. Men kan al muziek nummers opzoeken, selecteren en beoordelen. De scores worden opgeslagen in een werkende database. Wat er momenteel aan deze kant nog ontbreekt is een manier om enkel op artiest te zoeken en gebruik geschiedenis te kunnen bekijken.

Nadat deze twee zaken geïmplementeerd zijn, is de applicatie langs de client-side af. Dan moet er aan de server side gewerkt worden. Dit houdt in dat scores naar de data-collection server gestuurd moeten kunnen worden, en occassioneel scores gedownload moeten kunnen worden.

4 Implementatie

4.1 Package onderverdeling

De broncode van het project is opgedeeld in 3 packages: *gui, objects en services*. Hier een voorlopig overzicht van de packages en hun inhoud:

com.dsverdlo.AMuRate.gui
> AlbumActivity.java
> ArtistActivity.java
> MainActivity.java
> SearchResultsActivity.java
> TrackActivity.java

com.dsverdlo.AMuRate.objects
> Album.java
> RatingAdapter.java
> Track.java

com.dsverdlo.AMuRate.java
> DatabaseManager.java
> MyConnection.java

In het *GUI* package zitten alle activities[2]. Elk scherm heeft een aparte activity nodig.
In het *objects* package zitten de objecten, die abstractie brengen in het project. Bijvoorbeeld wanneer een track opgezocht werd en klaar is met downloaden, zal het omgezet worden in een klasse Track. Zo moet de GUI zich niets aantrekken van hoe de representatie eruit zag toen het van de server kwam.
Package *services* bevat de klassen die instaan voor verschillende diensten die moeten gebeuren in het project. Zo is er de MyConnection klasse, die de oproepen doet en antwoorden download van de Last.fm API. De database beheerder klasse is ook een dienst die verleend wordt aan het project, en bijgevolg ook in dit package zit.

4.2 Database

4.2.1 Lokale Database

De lokale database moet data bijhouden die offline geraadpleegd zou willen worden. Hierbij reken ik op een geschiedenis van de laatste zoekopdrachten, bekeken liedjes en de laatst gegeven scores. Aangezien deze applicatie vooral beroep doet op een actieve internetverbinding, zal er niet veel meer dan dat

[2]Dit is de naam van een scherm in Android

opgeslagen worden.
De implementatie van deze database gebeurt in SQLite.

- Ratings table

id	integer	primary key
date	date	
rating	float	
mbid	text	
title	text	
artist	text	

Tabel 2: Het model van de Rating tabel in de lokale database.

- Geschiedenis table

id	integer	primary key
date	date	
rating	float	
mbid	text	
title	text	
artist	text	

Tabel 3: De Geschiedenis tabel in de lokale database.

4.2.2 Externe Database

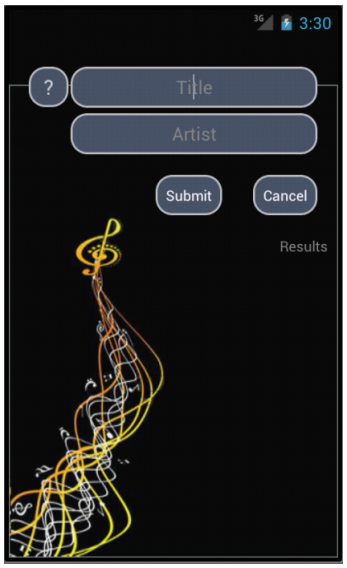
De externe database moet enkel scores van gebruikers verzamelen. Om verschillende gebruikers te identificeren zal de Android device ID mee opgeslagen worden. De tabel zal er dan zo uitzien:

id	integer	primary key
date	date	
rating	float	
mbid	text	
title	text	
artist	text	
userID	text	

Tabel 4: De Rating tabel layout van de database op de server.

5 GUI

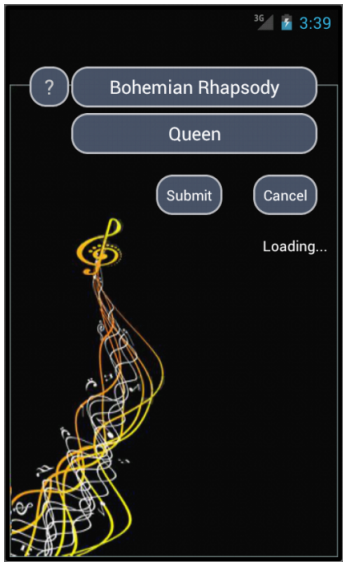
Hieronder volgen enkele schermopnames van de voorlopige GUI. Deze zal normaal gezien niet veel meer veranderen naar het verloop van de rest van het project, maar men kan nooit zeker weten.
Voor een voorbeeld van de huidige GUI volgt hier een voorbeeld van een gebruiker die de applicatie opstart en een lied opzoekt.



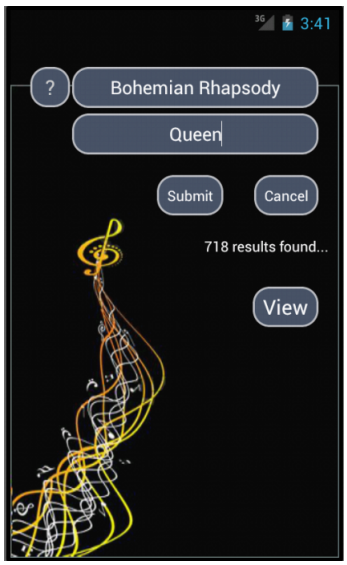
Afbeelding 1: Start scherm.

Dit scherm is het start scherm en het eerste wat de gebruiker ziet wanneer de applicatie gestart wordt. Er zijn twee velden om zoektermen in te geven (een voor artiest en een voor titel). De linkerboven knop met het vraagteken erop toont de instructies van de applicatie. De submit knop start een zoekopdracht, wanneer er zoektermen opgegeven zijn. De cancel knop wist alle ingevulde gegevens (zodat een andere zoekopdracht sneller uitgevoerd kan worden.)

In het naaststaande scherm werden de zoektermen "Bohemian Rhapsody" en "Queen" ingevuld, en op de knop Submit geklikt. De applicatie heeft de zoektermen verwerkt en start het informatie download proces. De text "Loading..." verschijnt op het scherm om de gebruiker te laten weten dat de applicatie ermee bezig is.



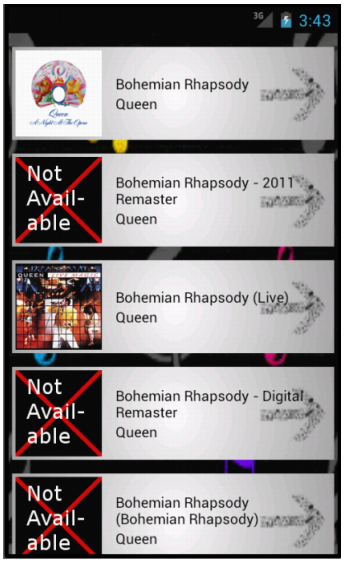
Afbeelding 2: Zoek- en downloadproces.



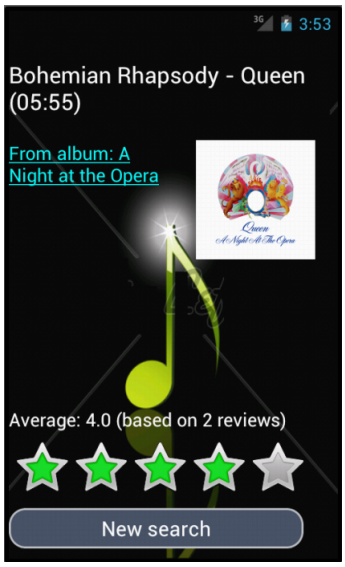
Afbeelding 3: Klaar met downloaden.

Wanneer de applicatie klaar is met het antwoord van de server te downloaden, verandert de text "Loading..." in het resultaat van de oproep, en het aantal gevonden resultaten. Wanneer er een positief aantal resultaten gevonden werd, verschijnt de knop View, waar de gebruiker op kan klikken om de resultaten te bekijken en het gewenste item te selecteren.

De lijst die resultaten weergeeft. Men kan op elke optie klikken om naar een pagina te gaan met meer informatie. Opties die niet beschikken over een afbeelding krijgen de standaard "Niet beschikbaar" afbeelding. De anderen worden asynchroon gedownload.



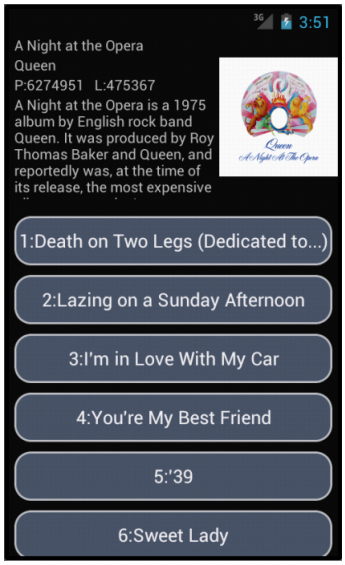
Afbeelding 4: Lijst met zoekresultaten.



Afbeelding 5: Gedetailleerd scherm voor een lied.

Dit scherm geeft de gedetailleerde pagina van een individueel lied weer. De duratie van het lied wordt getoond, het album van het lied, een afbeelding en het score systeem. Het gemiddelde en het aantal van alle stemmen voor dat lied wordt getoond, en 5 sterren bieden de mogelijkheid aan de gebruiker om een eigen score aan het lied toe te dienen. Helemaal onderaan de knop om een nieuwe zoekopdracht te starten.

Wanneer er meer album informatie beschikbaar is en de gebruiker klikt op het album, komt hij op dit scherm terecht. Het bevat informatie over een gegeven album zoals artiest, albumafbeelding, playcount, listeners, een scrollbare album informatie tekst en alle liedjes die op dat album staan. De gebruiker kan op elk lied klikken om naar de gedetailleerde liedjes pagina te gaan.



Afbeelding 6: Gedetailleerd scherm voor een album.

6 Bronnen, tools en referenties

- **Eclipse**
  Programmeerplatform Eclipse (Juno) werd gebruikt voor de implementatie en emulatie.
  `http://www.eclipse.org/`
- **Android Development Toolkit (ADT)**
  Deze plug-in bevat alle benodigdheden om programma's voor Android in Eclipse te bouwen en testen.
  `http://developer.android.com/sdk/index.html`
- **AMuRate homepagina**
  `http://dsverdlo.github.com/AMuRate`
- **Android**
  `http://www.android.com/`
- **Ice Cream Sandwich**
  `http://www.android.com/about/ice-cream-sandwich/`
- **Last.fm**
  `http://www.last.fm/home`
- **Github**
  `https://github.com/`