VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MOBILNÍ APLIKACE PRO PŘÍJEM PODCASTŮ A AUDELIVER.COM

MOBILE APP FOR RECEIVING PODCASTS AND AUDELIVER.COM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MAROŠ JANOTA**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, Ph.D.**
SUPERVISOR

BRNO 2017

Abstrakt

Cieľom tejto práce je implementácia mobilnej aplikácie na príjem podcastov zo služby Audeliver. Využíva moderné grafické uživateľské rozhranie pomocou Material Dizajnu. Aplikácia poskytuje automatickú synchornizáciu podcastov a sťahovanie najnovších epizód. O jednotlivých epizódach sa ukladajú informácie do databáze Realm. V jednotlivých kapitolách sú vysvetlené teoretické informácie o podcastoch a vytváraní aplikácií pre platformu Android. V ďalších častiach je opísaný návrh, testovanie a implementácia celej aplikácie MyPodcasts.

Abstract

The aim of this thesis is to implement a mobile application for receiving podcasts from Audeliver service. It is using modern graphic design for user interface based on Material Design. The application provides automatic synchronization of podcasts and downloading of newest episodes. All information about all episodes are saved into the Realm database. Individual chapters explain theoretical information about podcasts and developing of applications for Android platform. The other chapters describe design, testing, and implementation of the MyPodcasts application.

Kľúčové slová

Android aplikácia, Android Studio, Java, Podcasting, epizóda, hudobný prehrávač, prehrávač podcastov, Audeliver, MyPodcasts

Keywords

Android application, Android Studio, Java, podcast, episode, music player, podcasts player, Audeliver, MyPodcasts

Citácia

JANOTA, Maroš. *Mobilní aplikace pro příjem podcastů a audeliver.com.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Herout Adam.

Mobilní aplikace pro příjem podcastů a audeliver.com

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Prof. Ing. Adama Herouta, Ph.D. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Maroš Janota
27. apríla 2017

Poďakovanie

Na úvod by som sa rád srdečne poďakoval vedúcemu bakalárskej práce Prof. Ing. Adamovi Heroutovi, Ph.D. za poskytnuté pravidelné konzultácie, cenné rady a odborné vedenie.

Obsah

1 Úvod **2**

2 Podcasty a ich používanie **3**
2.1 RSS feed . . . 3
2.2 Aplikácia pre príjem podcastov . . . 3
2.3 Služba Audeliver . . . 4
2.4 Existujúce aplikácie . . . 5

3 Vývoj aplikácií pre zariadenia s OS Android **8**
3.1 Platforma Android . . . 9
3.2 Java a Android SDK . . . 9
3.3 Android Material Design . . . 11
3.4 Databáza Realm Mobile . . . 13

4 Návrh aplikácie **14**
4.1 Špecifikácia požiadaviek . . . 14
4.2 Diagram prípadov použitia . . . 15
4.3 Návrh užívateľského rozhrania . . . 16

5 Implementácia aplikácie **20**
5.1 Práca s databázou . . . 20
5.2 Prehrávanie mediálnych súborov . . . 21
5.3 Získavanie obsahu zo služby Audeliver . . . 23
5.4 Sťahovanie súborov . . . 25
5.5 Synchronizácia podcastov . . . 27
5.6 Nastavenia aplikácie . . . 27
5.7 Nastavenia Podcastov . . . 27
5.8 Ostatné komponenty . . . 28
5.9 Súbor Android Manifest . . . 28

6 Testovanie aplikácie **30**
6.1 Publikovanie v obchode Google Play . . . 31

7 Záver **33**

Literatúra **34**

Kapitola 1

Úvod

Táto bakalárska práca sa zaoberá implementáciou mobilnej aplikácie pre príjem podcastov z webovej služby Audeliver. Určená je pre zariadenia s operačným systémom Android. Služba Audeliver umožňuje vytvárať registrovaným užívateľom svoje vlastné podcasty pomocou vkladania odkazov z rôznych služieb, ako napríklad YouTube, Vimeo, Dropbox a podobne. Aplikácia dokáže vytvorené podcasty spracovať a sprístupniť ich obsah v mobilnom zariadení. Hlavnou výhodou je pridávanie podcastov bez potrebného prihlásenia do služby Audeliver, preto je určená pre cieľovú skupinu užívateľov, ktorý si chcú svoje podcasty spravovať priamo cez webovú službu a v zariadení chcú mať už len pripravený obsah na počúvanie. Synchronizácia so službou bude automaticky zabezpečená a užívateľ si môže prispôsobiť kedy a za akých podmienok sa môže vykonávať.

Práca je rozdelená do siedmych kapitol. V kapitole 2 sú popísané základné informácie o tom, čo to podcast vlastne je, ako sa s ním pracuje a rovnako bude vysvetlená aj služba Audeliver. Kapitola 3 opisuje vývoj aplikácií pre zariadenia s operačným systémom Android, jeho architektúru, databázu Realm Mobile a využívanie Android Material Designu v moderných aplikáciach. Kapitola 4 sa zaoberá návrhom aplikácie zameraného hlavne na grafickú časť. Sú v nej špecifikované požiadavky, ktoré musí aplikácia spĺňať a znázornený je aj diagram prípadov použitia. Najdôoležitejšia kapitola 5 popisuje samotnú implementáciu aplikácie. Sú v nej opísané princípy ako fungujú jednotlivé funkcie. Napríklad získavanie obsahu zo služby Audeliver, práca s databázou, sťahovanie nových epizód, nastavenia aplikácie a podobne. V kapitole 6 je opisané testovanie aplikácie na užívateľoch a čo všetko je potrebné na úspešné zverejnenie aplikácie do obchodu Google Play.

Kapitola 2

Podcasty a ich používanie

V roku 2000 prišiel na svet nový spôsob získavania mediálneho obsahu z rôznych staníc pomocou takzvaného **RSS feedu** [5]. V tej dobe sa táto služba nazývala aj **audioblogging**, no dnes ju poznáme hlavne pod menom **podcasting**. Jej hlavnou úlohou je prinášať vždy najnovší obsah z požadovaných staníc a to pomocou synchronizácie danej stanice so zariadením, na ktorom chceme mediálny obsah počúvať. Výsledkom je zoznam mediálnych súborov, ktoré sa dajú prehrávať online, alebo sa stiahnu do zariadenia a môžu byť prehrané kedykoľvek a to aj v offline režime. Tieto súbory nazývame **epizódy** a vďaka podcastingu si ich môžeme vypočuť kdekoľvek, koľko krát len chceme a hlavne umožňujú aj ich ovládanie, čo znamená pretáčanie zvukového záznamu dopredu alebo dozadu. Epizódy sú zvyčajne zvukové súbory vo formáte **mp3**.

2.1 RSS feed

Súbor formátu **RSS** (Rich Site Summary [5]) sa využíva pri čítaní noviniek z rôznych internetových zdrojov. Najčastejšie je vytvorený a ukladaný vo formáte **XML**. Ponúka získavanie najnovšieho obsahu z danej stránky (**RSS zdroja**), ktorý sa mení veľmi často. Používa sa napríklad pre spravodajstvo poskytujúce najnovšie správy, ale má využitie aj pre sprístupňovanie audio obsahu z rôznych vysielacích staníc alebo podcastových služieb.

V ukážke 2.1 je náhľad, čo všetko obsahuje RSS súbor vygenerovaný službou **Audeliver** ku jednému z vytvorených podcastov. Jedná sa o informácie, ktoré popisujú daný záznam a to napríklad jeho názov, URL adresu, dĺžku, popis alebo čas pridania do podcastu.

2.2 Aplikácia pre príjem podcastov

Názov podcast je vytvorený z dvoch častí a to „POD“ a „CAST“. Anglický význam týchto skratiek je **Portable On Demand** a **Broadcast**, čo v preklade znamená prenosný na vyžiadanie a vysielanie. Z toho vyplýva, že sa jedná o mediálny obsah, ktorý sa často pridáva a užívatelia ho môžu získavať do svojich zariadení. Preto vznikli podcast aplikácie, ktoré získajú daný obsah z vyššie spomínaného **RSS súboru**, spracujú ho a uložia do zariadenia.

Takáto aplikácia musí dokázať spracovať RSS súbory z ktorých sa vytvorí nový podcast obsahujúci najnovšie epizódy. Jedná sa o zoznam zvukových stôp, ktoré si užívateľ môže stiahnuť do svojho zariadenia a vypočuť si ich podľa potreby. Vďaka synchronizácie aplikácie s RSS zdrojom je dosiahnuté získavanie vždy najnovšieho obsahu a užívateľ

```
<rss xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd"
 ↪  xmlns:atom="http://www.w3.org/2005/Atom" xml:lang="de" version="2.0">
    <channel>
        <title>Podcast</title>
        <link>https://audeliver.com/</link>
        <atom:link href="http://audeliver.com/podcast/JrBO7cHXNT3A/feed.xml" rel="self"
         ↪  type="application/rss+xml"/>
        <description>New podcast for TED Talks episodes.</description>
        <generator>Audeliver - http://audeliver.com</generator>
        <lastBuildDate>Fri, 24 Mar 2017 19:28:42 +0100</lastBuildDate>
        <language>en</language>
        <copyright>Copyright 2017 Maros Janota. All Rights Reserved.</copyright>
        <itunes:image href="http://audeliver.com/imagecloud/default.jpg"/>
    </channel>
</rss>
```

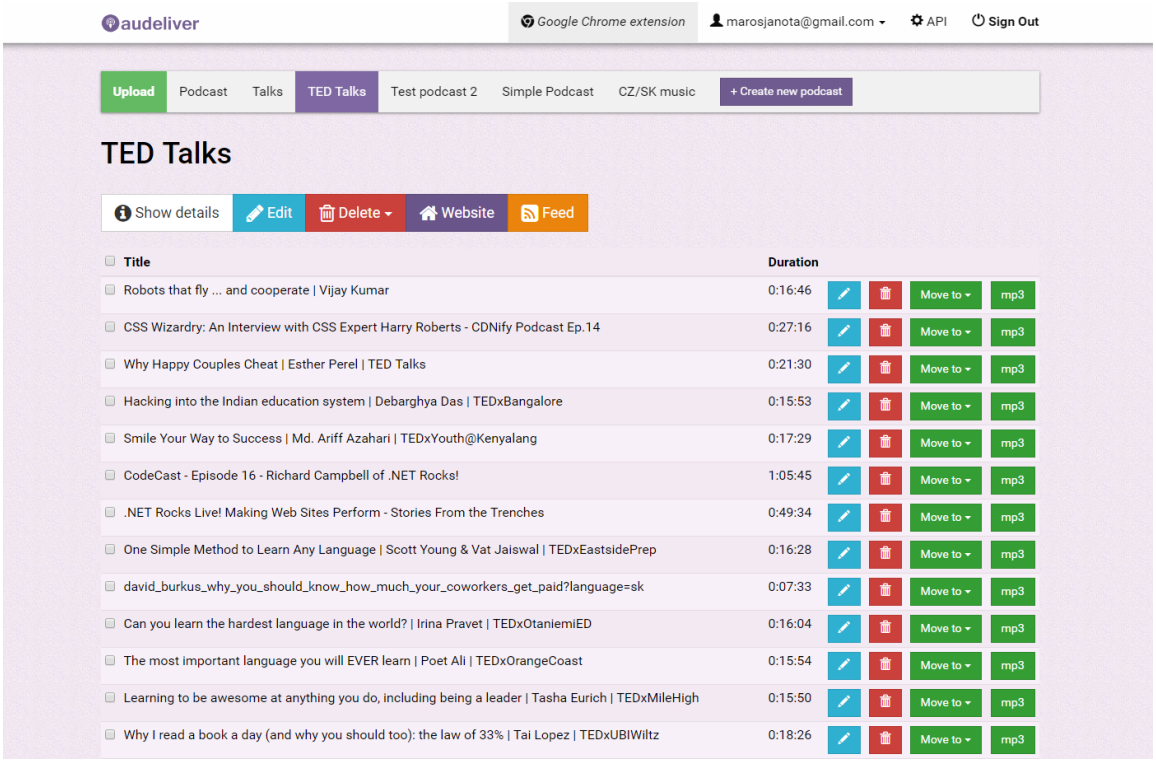
Obr. 2.1: Ukážka ako vyzerá začiatok vygenerovaného **RSS** súboru zo služby **Audeliver** obsahujúci všetky potrebné informácie o vytvorenom podcaste

nemusí manuálne kontrolovať, či boli pridané nové epizódy. Pri synchronizácii môžu byť epizódy automaticky stiahnuté do zariadenia a pripravené na následné počúvanie. Ďalšou funkciou podcastovej aplikácie je aj ukladanie informácií o jednotlivých epizódach. Aplikácia by tak mala vedieť, ktoré epizódy už boli prehrané kompletne alebo kde užívateľ skončil v prehrávaní a umožniť tak pokračovanie prehrávania od daného miesta. V aplikácii MyPodcasts bola implementovaná aj takzvaná fronta prehrávania, kde si užívateľ môže pridávať jednotlivé epizódy, ktoré si chcel vypočuť napríklad po ceste do práce. Vytvorená fronta obsahuje informácie o celkovej dĺžke všetkých zvukových stôp v nej a sprístupňuje túto informáciu aj užívateľom. Nastavenia aplikácie poskytujú rôzne možnosti, ktoré môžu zjednodušiť prácu pri používaní aplikácie takéhoto typu. Nastaviť sa dá napríklad cez aké internetové pripojenie môže aplikácia sťahovať najnovšie epizódy a tak zabráni zbytočnému čerpaniu mobilných dát alebo umožňuje nastaviť, či epizóda má byť po prehratí zmazaná a tak ušetrí aj miesto v pamäti telefónu. Viacej informácií o nastaveniach tejto aplikácie, ale aj o celkovej funkcionalite sa môžete dočítať v ďalších častiach práce.

2.3 Služba Audeliver

Internetová služba **Audeliver** umožňuje registrovaným užívateľom vytvárať svoje vlastné podcasty, ktoré môžu byť súkromné alebo verejné pre všetkých. Do vytvoreného podcastu je možné pridávať odkazy, z viacerých služieb obsahujúcich video alebo audio stopu. Podporované služby sú momentálne **YouTube**, **Vimeo**, **Dropbox**, **Kaltura** a **SoundCloud**. Ďalšia možnosť pridávania je aj vloženie súboru z vlastného počítača. Vo všetkých prípadoch sa vždy detekuje, či sa jedná a čisto audio stopu alebo o video. Ak je vložený obsah video súbor, vytiahne sa z neho audio stopa a uloží ju na úložisko služby audeliver. Vytvorí sa nová epizóda obsahujúca základné informácie ako názov, popis alebo odkaz na úložisko, kde bola nahraná zvuková stopa a následne môže byť priradená do niektorého z vytvorených podcastov. Epizódy sa môžu presúvať medzi všetkými podcastami, ktoré má užívateľ

vytvorené na svojom účte. Ak sa jedná o bezplatný účet, nie je zaručená garancia, že sa epizóda bude dať vždy stiahnuť. Po zaplnení úložiska sa najstaršie epizódy zmažú a vytvorí sa tak miesto pre nové. Audeliver poskytuje možnosť zakúpenia prémium účtu, ktorý zabezpečí, aby sa nikdy žiadne epizódy nezmazali a boli tak vždy dostupné.



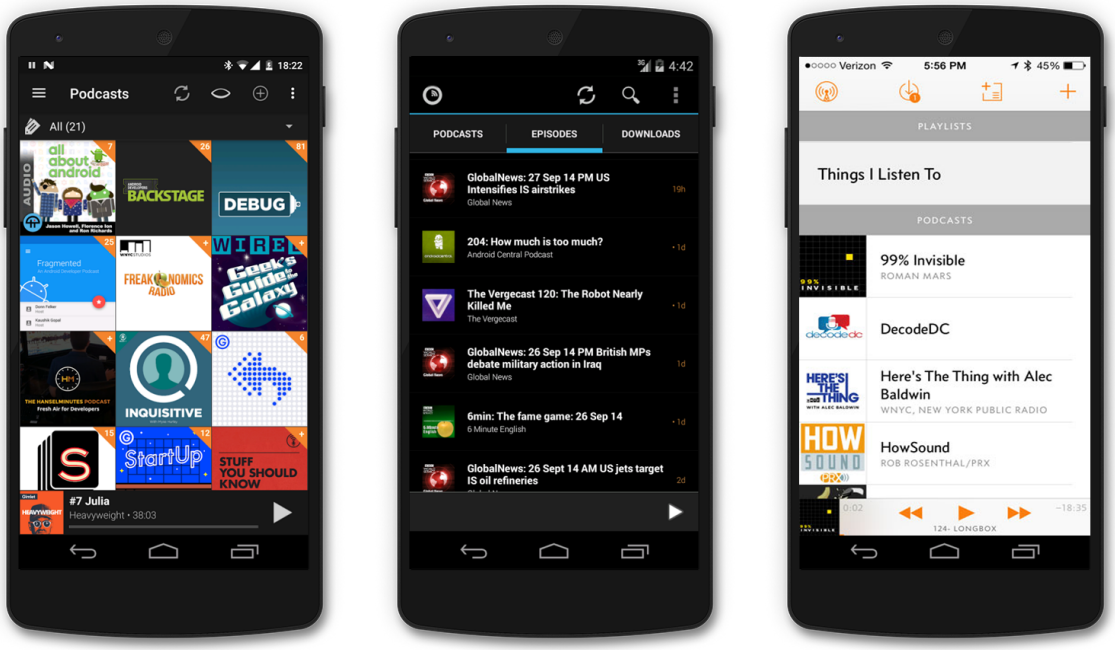
Obr. 2.2: Snímka zo služby Audeliver s ukážkou vytvoreného podcastu. Zobrazené sú aj jednotlivé epizódy, ktoré môžu byť upravené, zmazané alebo presunuté do iného podcastu.

Každý jeden podcast má vygenerované dve samostatné stránky. Jedná je v prehľadnej podobe, kde sú zobrazené všetky pridané epizódy a môžu byť zároveň na nej aj vypočuté a druhá stránka je vo formáte RSS pripraveného pre aplikácie na spracovanie a sprístupnenie obsahu. Názov stránky si môže každý užívateľ zvoliť sám a to v nastaveniach daného podcastu. Tento údaj je veľmi dôležitý a podrobnejší popis nájdete v kapitole 5.3.

2.4 Existujúce aplikácie

Existuje už viacero aplikácií, ktoré dokážu pracovať s podcastami a táto časť sa bude venovať práve nim. Rozoberané budú 3 najznámejšie a najviac využívané aplikácie. Popisovať sa budú ich hlavné výhody a nevýhody oproti ostatným aplikáciám, keďže funkcionalita je skoro vo všetkých podobná. Ďalej sa bude rozoberať ich grafické prostredie a náročnosť používania.

- **Podkicker** – Aplikácia Podkicker disponuje veľmi prehľadným a jednoduchým grafickým rozhraním, na ktoré si každý jeden užívateľ veľmi rýchlo zvykne. Využíva **Material Design** a kontrolku **TabsView** na rýchle prepínanie medzi obsahom. Vďaka tomu má užívateľ všetko pripravené hneď po zapnutí aplikácie a medzi obsahom sa



Obr. 2.3: Ukážky existujúcich aplikácií na porovnanie grafického prostredia. Z ľava môžete vidieť **Pockast Addict**, **Podkicer** a na konci **Overcast**

presúva iba rýchlim pohybom prstov po obrazovke do strán. Samotné ovládanie prehrávania je už trochu zložitejšie a je na to potrebné viacero zbytočných kliknutí. Na spodnej časti obrazovky sa nachádza informačná lišta o aktuálne prehrávanej epizóde a jediné tatíčko na pozastavenie alebo pokračovanie v prehrávaní. Pre zobrazenie celého ovládacieho panelu je potrebné kliknúť na spomínaný názov a tým sa vysunie rozšírená lišta s ovládacími prvkami. Kliknutie na názov epizódy nie je moc intuitívne a nemusí to napadnúť všetkých užívateľov. Pre zavretie ovládacieho panelu je potrebné kliknúť znovu na názov alebo na prázdnu plochu v lište. Hlavnou výhodou tejto aplikácie je dobre štruktúrované menu s nastaveniami v ktorých sa dá rýchlo zorientovať.

- **Podcast Addict** – Druhá testovaná aplikácia je Podcast Addict, ktorá na rozdiel od prvej spomínanej aplikácie **Podkicker** ma veľmi neprehľadné grafické prostredie. Už pri jej zapnutí zostane užívateľ v šoku a je potrebných pár minút na zorientovanie sa v nej. Pri stiahnutí neplatenej verzii sa zobrazí na spodku obrazovky reklama, ktorá je naozaj rušivá. Rovnako ako Podkicker aj táto aplikácia ponúka podobné ovládanie prehrávanej epizódy. V dolnej lište je iba obrázok, názov a tlačidlo na pozastavenie. Pri otvorení ovládacej lišty sa vysunie celá nová obrazovka zo spodnej časti zariadenia a indikuje tak, že sa dá zobraziť aj pomocou rovnakého pohybu ako bola animácia, no bohužiaľ tak to nefunguje. Takéto chovanie je v rozpore s celým **Material Designom**, kde práve animácie majú pomáhať naznačovať, ako sa môže aplikácia ovládať alebo prepínať medzi jej obsahom. Na ľavej časti obrazovky sa nachádza skryté navigačné menu, ktoré je naozaj dlhé a dokonca sa v ňom musí ešte aj posúvať, aby sa odkryl všetok obsah. Aby toho nebolo málo, druhá časť sa skrýva v hornej časti aplikácie a to pod pravým tlačidlom, ktoré zvyčajne slúži pre nastavenia, no v tomto prípade

sa otvorí ďalšie posuvné okno z pravej strany s ďalšími funkciami aplikácie. Na jednu stranu je až prekvapujúce, čo všetko aplikácia dokáže a čo sa v nej dá nastaviť, ale na druhú stranu, je tam toho tak veľa, že radšej by ste mali záujem aplikáciu zatvoriť, odinštalovať a nájsť niečo jednoduchšie. Ukážku môžete vidieť v kapitole 4.3 obrázok 4.3, kde bol tento príklad použitý na zobrazenie neprehľadného grafického prostredia.

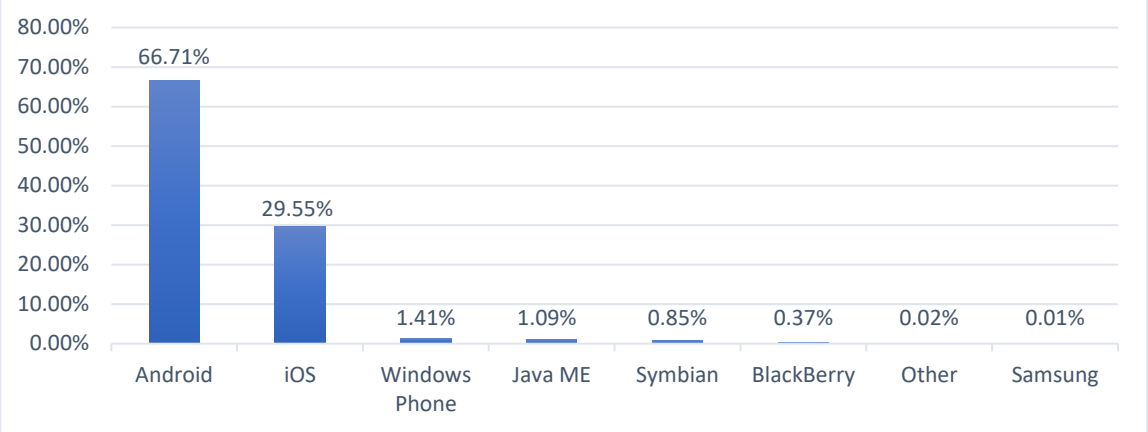
- **Overcast** – Posledná testovaná aplikácia je implementovaná pre zariadenia s operačným systémom (OS) **iOS**. Overcast ponúka veľa užitočných funkcií ako napríklad rýchle filtrovanie epizód, zlepšenie zvukovej kvality prehrávania potlačením okolitého šumu alebo nastavenie rýchlosti prehrávania bez poškodenia zvukovej stopy. Po grafickej stránke aplikácia pôsobí trochu chaoticky. Zoznam podcastov nezobrazuje pomocou ich názvov, ale len obrázkov. Tie niekedy nemusia dať správne najavo o aký podcast sa jedná a musí byť kvôli tomu otvorený. Celková grafika je trochu odlišná od tej na akú sú zvyknutý používatelia zariadení s OS Android. Pre nich môže byť orientácia v nej komplikovanejšia.

Po preskúmaní existujúcich aplikácii som došiel k záveru, že aplikácia MyPodcasts bude bez reklám a navrhnutá bude pomocou Material Designu, na ktorý sú užívatelia OS Android zvyknutý. Ďalej aplikácia bude obsahovať iba najdôležitejšie nastavenia, aby ich nebolo zbytočne veľa a dalo sa rýchlo nastaviť to najpotrebnejšie. Funkcionalita bude zameraná hlavne na pripravenie nového obsahu pre užívateľa bez potrebného prihlasovania do služby Audeliver a možnosť pracovať s jednotlivými podcastami a epizódami. To znamená ukladať o nich potrebné informácie do databáze, prispôsobovať si ich podľa potreby, ovládať prehrávanie epizód a podobne.

Kapitola 3

Vývoj aplikácií pre zariadenia s OS Android

Aplikácia bola implementovaná pre mobilné zariadenia s operačným systémom (OS) Android. Podľa nameraných štatistík ku dňu 28.5.2017 uviedol zdroj **NetMarketShare** [7], že momentálne využíva približne **66.71**% užívateľov práve túto platformu. Preto vývoj aplikácie bol vybraný práve iba pre zariadenia s OS Android. Cieľom bolo využiť všetky dobré vlastností tejto platformy a vytvoriť tak spoľahlivú aplikáciu, čo by sa pri vývoji aplikácie na viaceré mobilné platformy nemuselo podariť. Mimo iné disponuje aj komplexným sprievodcom k vytváraniu aplikácií v dizajnovom jazyku **Material Design** [2], ktorý bude viacej rozoberaný v kapitole 3.3. Spomínaný Material dizajn v tejto dobe využíva väčšina nových aplikácii a práve to pomôže užívateľom aplikácie **MyPodcasts**, rýchlejšie sa naučiť ju ovládať a zorientovať sa v jej obsahu.



Obr. 3.1: Graf znázorňujúci využívanie platforiem na mobilných zariadeniach ku dňu 28.2.2017 z internetóveho zdroja o prieskumoch v marketingu **NetMarketShare** [7]

3.1 Platforma Android

Android [4] je veľmi rozsiahla opensource platforma, založená na Linuxe. Bola vyvinutá hlavne pre mobilné zariadenia a jej jadro bolo navrhnuté tak, aby dokázalo pracovať s veľkým množstvom zariadení, bez ohľadu na ich hardvér alebo veľkosť obrazovky. Architektúra tejto platformy je rozdelená do piatich vrstiev:

1. **The Linux Kernel** (Jadro operačného systému) – tvorí abstraktnú vrstvu medzi hardvérom a softvérom na vyšších vrstvách. Jadro je postavené na Linuxe a využíva veľa jeho vlastností, ako napríklad zabudované ovládače, správa procesov, pamäte alebo sietí. Zároveň poskytuje aj kľúčové bezpečnostné funkcie pre Android.

2. **Hardware Abstraction Layer (HAL)** (Hrdvérová abstraktná vrstva) – poskytuje rozhranie pomocou ktorého vystavuje všetky možnosti hardvéru pre vyššiu vrstvu Java API Framework. HAL pozostáva z viacerých knižníc, ktoré implementujú rozhrania pre prístup k rôznym častiam hardvéru, ako je napríklad kamera alebo Bluetooth modul.

3. **Android Runtime a Native C/C++ Libraries** (Aplikačná vrstva a natívne knižnice) – táto vrstva poskytuje veľké množstvo natívnych knižníc a môžu byť využívané rôznymi časťami systému. Medzi tie najpoužívanejšie knižnice patrí napríklad Media Libraries (prehrávanie video a audio formátov), SQLite (prístup k databázam), OpenGL (vykresľovanie grafiky), LibWebCore (využívaná rôznymi webovými prehliadačmi) a mnoho ďalších.

4. **Java API Framework** – najdôležitejšia vrstva pre vývojárov. Poskytuje prístup k viacerým službám a komponentom ako napríklad

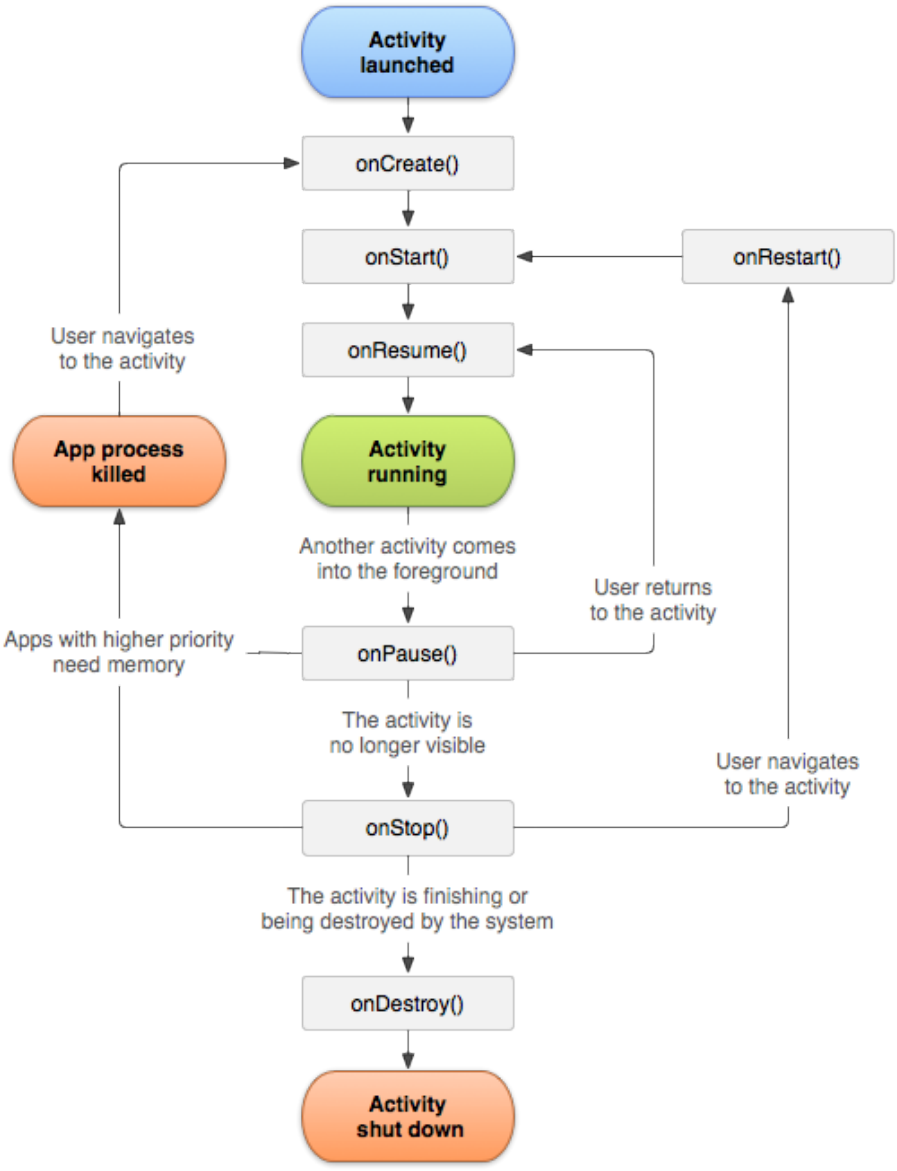
   - **View System** (Zobrazovanie obsahu) – grafické rozhranie
   - **Resource Manager** (Správca zdrojov) – prístup ku grafike, textom, vzhľadom a podobne
   - **Notification Manager** (Správca upozornení) – zobrazovanie notifikácií v lište
   - **Activity Manager** (Správca aktivít) – udržuje životný cyklus vytvorených aktivít
   - **Content Providers** (Poskytovanie obsahu) – umožňuje prístup k dátam z iných aplikácií, ako napríklad Kontakty, Správy a podobne

5. **System Apps** (Systémové aplikácie) – Android poskytuje set základných aplikácií napríklad pre správu emailov, SMS správ, kalendára a podobne. Každý užívateľ si môže stiahnuť do svojho zariadenia aj aplikácie z tretích strán a to pomocou služby Google Play. Tieto aplikácie môžu byť nastavené ako primárne a nahradia tak zabudované.

3.2 Java a Android SDK

Na vývoj aplikácii pre zariadenia Android v tejto dobe už existuje viacero možností no väčšina vývojárov využíva práve programovací jazyk **Java** a **Android SDK**. Java je jeden z mnoho objektovo orientovaných jazykov a na vývoj mobilných aplikácii sa používa trochu upravená verzia **Java pre Android** [9] aplikácie. Tento druh jazyka Java je veľmi podobný

a obsahuje len pár drobných zmien oproti klasickej Jave. Jedná sa hlavne o rozšírenie potrebnými knižnicami a je medzi nimi rozdielne kompilovanie kódu. Zároveň pri vytváraní Android aplikácie musíme počítať aj so životným cyklom každej jednej vytvorenej aktivity, preto bola pridaných viacero nových metód na jeho kontrolu. Celý životný cyklus aktivity je možné vidieť na obrázku 3.2, no medzi tie najpoužívanejšie časti patrí práve **onCreate** (pri vytvorení), **onResume** (pri vrátení do aktivity), **onPause** (pri pozastavení) a **onDestroy** (pri ukonční a zmazaní aktivity), kde sa nastavuje chovanie aktivity v požadovaných stavoch. To znamená, čo sa ma zobraziť na obrazovke pri vytváraní aktivity alebo ktoré prostriedky sa musia uvoľniť pri ukončení aktivity.



Obr. 3.2: Životný cyklus aktivity s pripravenými metódami, ktoré môže vývojár používať na vytvorenie potrebného chovania aktivity, vo všetkých jej životných fázach [3].

3.2.1 Eclipse

Eclipse je open source vývojové prostredie určené na programovanie vo viacerých jazykoch (Java, C++, PHP, a iné). Poskytuje veľmi veľké množstvo rozšírení, ktoré môžu pomôcť pri vývoji pridaním ďalších nových funkcií. Editor zdrojových kódov poskytuje napríklad kontrolu syntaxe, automatické doplňovanie, našepkávanie, vyznačovanie chýb alebo možnosť vytvárania textu do dokumentácie pomocou komentárov.

3.2.2 Android Studio

Android Studio je bezplatný a oficiálny nástroj na vývoj aplikácii pre zariadenia s platformou Android. Predstavený bol v roku 2013 spoločnosťou **Google** a je neustále vo vývoji. Keďže je primárne zameraný iba na vývoj mobilných aplikácií, poskytuje viacej možností a nástrojov ako Eclipse. Medzi jeho hlavné výhody patrí aj rýchly emulátor, na ktorom si môže vývojár testovať svoje aplikácie na rôznych druhoch zariadení, napríklad s inou verziou Androidu alebo veľkosťou obrazovky. Momentálne je Android Studio viacej podporovaný a doladený a preto bol zvolený ako vývojové prostredie pri samotnom vývoji aplikácie MyPodcasts.

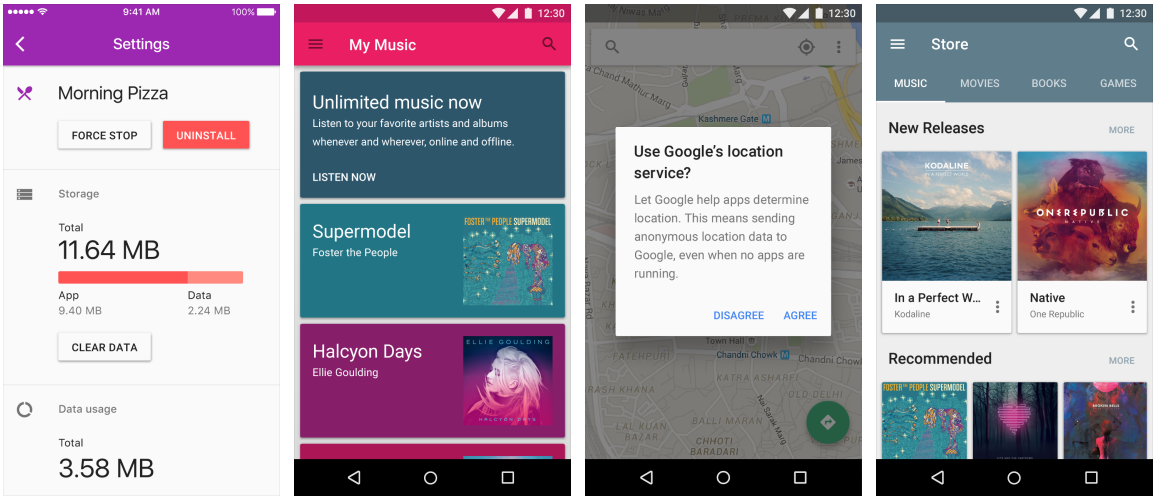
3.3 Android Material Design

Dizajnový jazyk **Material Design** vznikol v roku 2014 a bol vyvinutý spoločnosťou Google. Priniesol nový vzhľad a sadu pravidiel pre aplikácie, ktoré s ním sú implementované. Tento vzhľad bol oveľa jednoduchší, farebnejší a omnoho výraznejší ako predošlí. Priniesol kopu nových animácií a hlavne nové využitie z-osi. Jednotlivé prvky sú tak radené nad seba a je jasne vidieť, s ktorou časťou aplikácie sa práve pracuje. Aktuálne zobrazené elementy sú výraznejšie a zobrazené nad ostatnými, nepoužívanými elementami. Animácie pomocou ktorých sú jednotlivé prvky stránky zobrazované naznačujú chovanie aplikácie. Vďaka nim môže užívateľ ľahko prísť na to, že ak sa daný element presunul pomocou animácie mimo obrazovku, tak sa môže zobraziť potiahnutím obrazovky z tej istej strany, kde bol skrytý.

Spoločnosť Google spolu s novým dizajnovým jazykom priniesol aj webovú stránku, ktorá by mala naučiť vývojárov navrhovať aplikácie v ich novom vzhľade. Jedná sa o **komplexného sprievodcu** určujúci sadu pravidiel, ktoré sa využívali aj pri vývoji aplikcáie MyPodcasts. V sprievodcovi nájdeme ukážky, ako správne navrhovať aplikácie ale aj to, ako by to vyzerať nemalo. Material dizajn prináša zároveň aj kopu nových komponent, ktoré sa môžu využívať pri implementácii aplikácie Android. Na obrázku 3.3 sú zobrazené najviac využívané komponenty v aplikácii. Jednotlivé komponenty sú stručne opísané v ďalších častiach tejto kapitoly. Jedná sa len o krátky popis ako fungujú a v kapitole o implementácii (kap. 5) sa popisuje ich reálne využívanie priamo v aplikácii MyPodcasts.

3.3.1 Tlačidlá

Material dizajn prináša tri základné druhy tlačidiel. Prvé je **Floating action button** (plávajúce tlačidlo), ktoré je v tvare kruhu a zobrazuje animáciu vyplnenia tmavšou farbou pri stlačení. Využíva sa hlavne na pridávanie nového obsahu alebo editáciu. Ďalší typ je **Raised button** (vyvýšené tlačidlo) v obdĺžnikovom tvare. Pri stlačení má rovnakú animáciu ako plávajúce tlačidlo. Obe tlačidlá sú vyvýšené a dávajú tak najavo, že je možné na ne kliknúť. Slúži ako klasické tlačidlo, ktoré sa môže využívať v ktorejkoľvek časti aplikácie. Ako posledné je pripravené na použitie **Flat button** (ploché tlačidlo) v podobe čistého



Obr. 3.3: Ukážky niektorých komponent, ktoré prináša Material dizajn. Jedná sa o najviac využívané komponenty v aplikácii MyPodcasts (z ľava: Tlačítka, Karty, Dialóg, Taby)

textu, bez orámovania. Zvyčajne sa používa pri potvrdzovacích akciách alebo v dialógoch, viz tretí obrázok v ukážkach komponentov 3.3.

3.3.2 Karty

Karty slúžia na zobrazovanie základných informácii o rôznych položkách. Môžu obsahovať napríklad zobrazenú fotografiu/ikonu, názov položky a krátky popis. Slúži aj ako vstupný bod pre zobrazenie detailnejšieho popisu konkrétnej položky. Po kliknutí na kartu sa môže vykonať potrebná akcia, ako napríklad jej odstránenie alebo zobrazenie dialógového okna s podrobnejšími informáciami. Najčastejšie sa používajú v listových zoznamoch a jedná sa v tomto prípade o kolekciu viacerých kariet.

3.3.3 Dialógové okná

Ďalšou komponentnou je dialógové okno, ktoré slúži pre informovanie užívateľa o špecifickej úlohe a môže obsahovať napríklad potvrdzovanie, výstrahu alebo môže zahŕňať rôzne iné akcie. Typické príklady použitia dialógového okná sú:

- **Výstraha** – prerušenie, ktoré informuje užívateľa o konkrétnej akcii pre ktorú je potrebné vybrať odpoveď
- **Jednoduché menu** – v prípade zoznamu môže zobraziť podrobnejšie informácie o konkrétnej položke
- **Potvrdzovacie okno** – hlavné využitie má napríklad pri odstraňovaní prvkov, pri ktorom je potrebné potvrdenie akcie

3.3.4 Taby

Hlavnou úlohou tabov je rýchle prepínanie medzi viacerými obrazovkami. Taby sa dajú prepínať viacerými spôsobmi. Ako prvý je kliknutie na konkrétny tab v zozname, ktorý by mal byť vždy implementovaný do jedného riadku a jednotlivé názvy tabov by mali byť

zobrazené vedľa seba. Pri zmene tabu sa zobrazuje animácia presunutia obsahu mimo obrazovku do strany, ktorá naznačuje užívateľovi ďalší spôsob prepínania. Jedná sa o presúvanie medzi tabmi pomocou potiahnutia obrazovky do jednej zo strán, ako to bolo znázornené v animácii.

3.4 Databáza Realm Mobile

Namiesto klasickej databázy **SQLite**, ktorá sa primárne využívala na platforme Android, bola pri implementácii aplikácie zvolená náhrada pomocou databázy **Realm Mobile** [8]. Jedná sa o prvú mobilnú databázu, ktorá mala nahradiť práve SQLite. Realm bol vytvorený pre mobilnú platformu Android, ale aj iOS. Medzi jej hlavné výhody patrí:

- približne **10x rýchlejšia** ako SQLite
- jednoduchšie používanie
- umožňuje **vytváranie objektov**
- vhodné pre vytváranie a ukladanie dát v reálnom čase

Navzdory kopu ďalším výhodám má bohužiaľ Realm databáza aj pár nevýhod, ktoré je potrebné doladiť:

- stále vo vývoji
- neumožňuje exportovanie/importovanie databázy
- komplikovanejšia migrácia (strata dát pri upravení databázy)

Práca s databázou, štruktúra a vytváranie nových objektov sú detailnejšie opísané v kapitole 5.1.

Kapitola 4

Návrh aplikácie

Nasledujúci text sa zaoberá špecifikáciou požiadaviek, analýzou prípadov použitia a návrhom grafického prostredia aplikácie. Cieľom bolo vytvoriť aplikáciu, ktorá umožní viacero nastavení pre uľahčenie práce s podcastmi a zároveň nebude obsahovať zbytočne veľa nepotrebných nastavení v ktorých sa nejeden užívateľ stratí. Táto kapitola je hlavne zameraná na návrh grafického prostredia a špecifikovanie požiadavkov [6]. Návrh ostatných častí, ako je napríklad databáza, je znázornený v kapitole 5, priamo pri ich implementácii.

4.1 Špecifikácia požiadaviek

Na úplnom začiatku vývoja bolo potrebné nazbierať čo najviac informácii od potenciálnych užívateľov, aby sa presne definovalo chovanie a funkcie aplikácie. Pri získavaní potrebných špecifikácii sa zo začiatku vychádzalo z existujúcich aplikácií, ktoré boli spomenuté v kapitole 2.4. Bral sa ohľad na funkcie a návrh grafických rozhraní, ktoré užívateľov najviac zaujali a zároveň aj tie, ktoré práve rušili užívateľa pri práci alebo boli zbytočné. Väčšina užívateľov s ktorými boli vyhotovené špecifikácie, spravovala podcasty priamo v službe Audeliver a v mobilnom zariadení požadovali iba získavanie obsahu z hotových podcastov. Preto sa počítalo s variantnou, že aplikácia nemusí umožňovať pridávanie nových epizód do podcastu pomocou mobilného zariadenia.

Z nazbieraných poznatkov sa vytvoril základný zoznam požiadavkou, ktoré by mala hotová aplikácia MyPodcasts splňovať. Medzi základné funkcie, ktoré musí zvládnuť každá jedna aplikácia takéhoto typu patrí:

- Pridávanie nových podcastov (v tomto prípade pomocou nastaveného **ID** alebo vytvoreného **QR kódu** službou Audeliver)
- Mazanie a synchronizácia pridaných podcastov
- Sťahovanie, mazanie a prehrávanie epizód

Aby výsledná aplikácia bola zaujímavejšia a malo význam na nej pracovať, museli sa stanoviť aj ďalšie špecifikácie, ktoré ju vylepšia, zjednodušia prácu alebo ponúknu užívateľom niečo nové. Preto boli pridané ďalšie vlastnosti a funkcie ako napríklad:

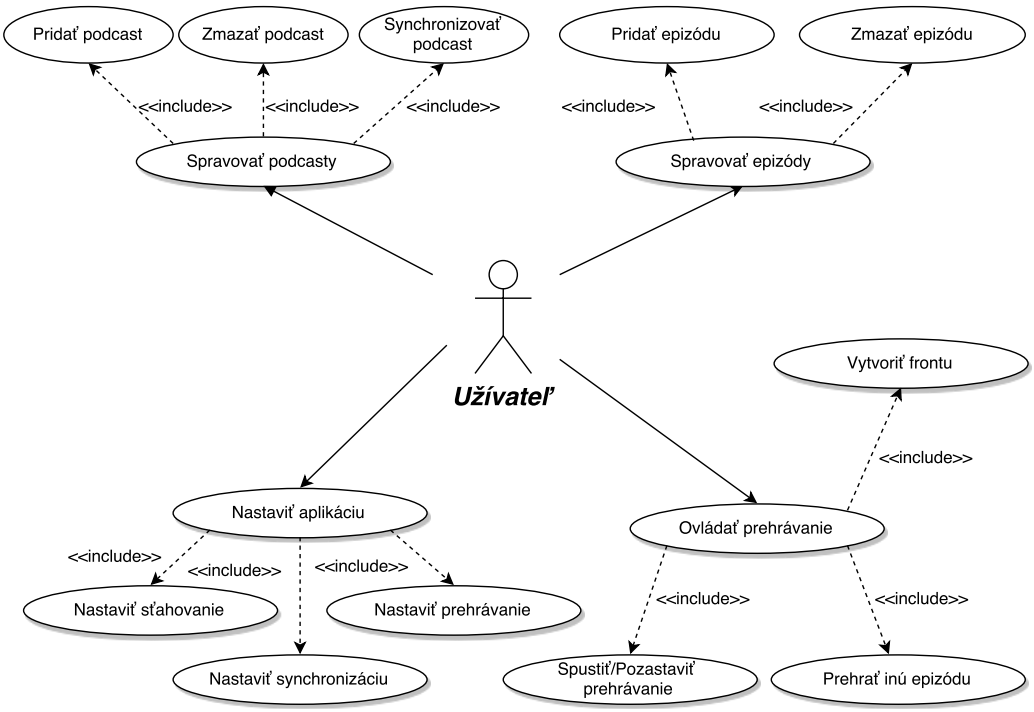
- Nastavenia celej aplikácie (globálne, sťahovanie a synchronizácia)
- Nastavenia pre jednotlivé podcasty

- Prehrávanie všetkých stiahnutých epizód alebo **prehrávanie z fronty**, ktorú si vytvorí užívateľ
- Poskytovanie potrebných informácií o stave aplikácie
- Ukladanie informácií o prehrávaných epizódach (aj po ich zmazaní)
- Rýchle **pretáčanie epizód**
- Prehrávanie od posledného skončeného miesta (pre každú jednu epizódu)

Jednotlivé funkcie a vlastnosti budú viacej rozoberané v ďalších kapitolách a to hlavne v kapitole 5 o ich implementácii.

4.2 Diagram prípadov použitia

Na obrázku 4.1 je znázornený **diagram prípadov použitia** pre aplikáciu MyPodcasts. Obsahuje všetky základné operácie, ktoré musí aplikácia podporovať, aby splnila špecifikované požiadavky.



Obr. 4.1: Znázornený **diagram prípadov použitia** pre aplikáciu **MyPodcasts**. Jedná sa o základné operácie, ktoré musí aplikácia podporovať.

4.3 Návrh užívateľského rozhrania

Na zobrazovanie obsahu sa v celej aplikácii využívajú **fragmenty** (Fragments[1]), ktoré predstavujú vrstvu medzi **aktivitami** (Activity[2]) a **zobrazením** (View[3]). Fragmenty majú podobný životný cyklus ako aktivity (obrázok 3.2), no namiesto metódy `onCreate()` sa využíva `onCreateView()`, ktorá priamo obaľuje to, čo sa má vo Fragmente zobraziť. Hlavná výhoda Fragmentu je, že sa môže instancovať priamo, bez potrebného Intentu (abstraktná operácia potrebná na vyvolanie Aktivity) z ktorejkoľvek časti aplikácie. Prepínanie fragmentov ma na starosti metóda `displayView()` v hlavnej aktivite `MainActivity`. Tá sa stará o zobrazenie správneho fragmentu podľa zadaného id konkrétneho fragmentu. Po jeho zobrazení sa nastaví aj nadpis, ktorý informuje užívateľa o tom, v ktorej časti aplikácii sa práve nachádza.

Obr. 4.2: Ukážka hlavnej obrazovky aplikácie MyPodcasts. Zobrazený zoznam obsahuje všetky stiahnuté epizódy a pri každej z nich sú znázornené dodatočné informácie, ktorých význam je na obrázku vysvetlený.

[1] Android Fragment - https://developer.android.com/guide/components/fragments.html
[2] Android Activity - https://developer.android.com/reference/android/app/Activity.html
[3] Android View - https://developer.android.com/reference/android/view/View.html

4.3.1 Hlavná obrazovka

Pri zapnutí aplikácie sa zobrazí obrazovka, ktorá využíva komponentu **taby** (3.3.4) a zobrazuje v nich všetky stiahnuté epizódy v zariadení. Prvý tab obsahuje úplne všetky stiahnuté epizódy zo všetkých podcastov, ktoré sa môžu prehrať. Pri každej jednej epizóde sa zobrazujú informácie o jej stave, ako napríklad či už bola prehraná alebo koľko z nej už bolo vypočuté. Jednotlivé prvky aj s vysvetlivkami môžete vidieť na obrázku 4.2. V druhom tabe sú zobrazené iba epizódy, ktoré boli pridané do fronty prehrávania a jej vzhľad ostáva úplne rovnaký, ako to bolo v predchádzajúcom prípade. Medzi tabmi sa dá prepínať pomocou klasického preklikávania alebo aj pomocou gest.

Na zobrazovanie všetkých stiahnutých epizód sa využíva **RecyclerView**[4], ktorý poskytuje možnosť zobrazovania väčšieho množstva dát pomocou adaptéru. Adaptér zobrazuje iba taký počet prvkov, aký sa zmestí na obrazovku. Ak zoznam obsahuje napríklad 100 položiek a na obrazovku sa ich zmestí len 5, nemusia sa vykresľovať úplne všetky, ale iba tie potrebné. Pri posúvaní v zozname sa načítavajú nové položky a tie staré sa zase skrývajú.

Navigácia je skrytá v ľavej časti obrazovky a dá sa otvoriť pomocou kliknutia na tlačidlo zobrazujúce menu v hornej časti obrazovky alebo gestami (potiahnutie obrazovky z ľavého kraja do stredu). Menu obsahuje iba základné ovládacie prvky aby bolo prehľadné. Jedná sa o zobrazenie všetkých stiahnutých podcastov, nastavení a prehrávania. Nájdeme v ňom aj doplňujúci odkaz na zobrazenie informácií o aplikácii.

4.3.2 Nastavenia

Ako už bolo spomínané v kapitole o existujúcich aplikáciách 2.4, aplikácia **Podcast Addict** obsahuje veľmi neprehľadné menu, ktoré môžete vidieť na obrázku 4.3 v ľavej časti. Vedľa neho je zobrazené riešenie v aplikácii MyPodcasts, kde sú nastavenia rozdelené do troch základných kategórii a tie obsahujú jednotlivé možnosti nastavenia aplikácie (viz pravá časť obrázku). Týmto spôsobom zobrazovania sú nastavenia oveľa prehľadnejšie a aplikácia sa dá rýchlo a jednoducho ovládať.

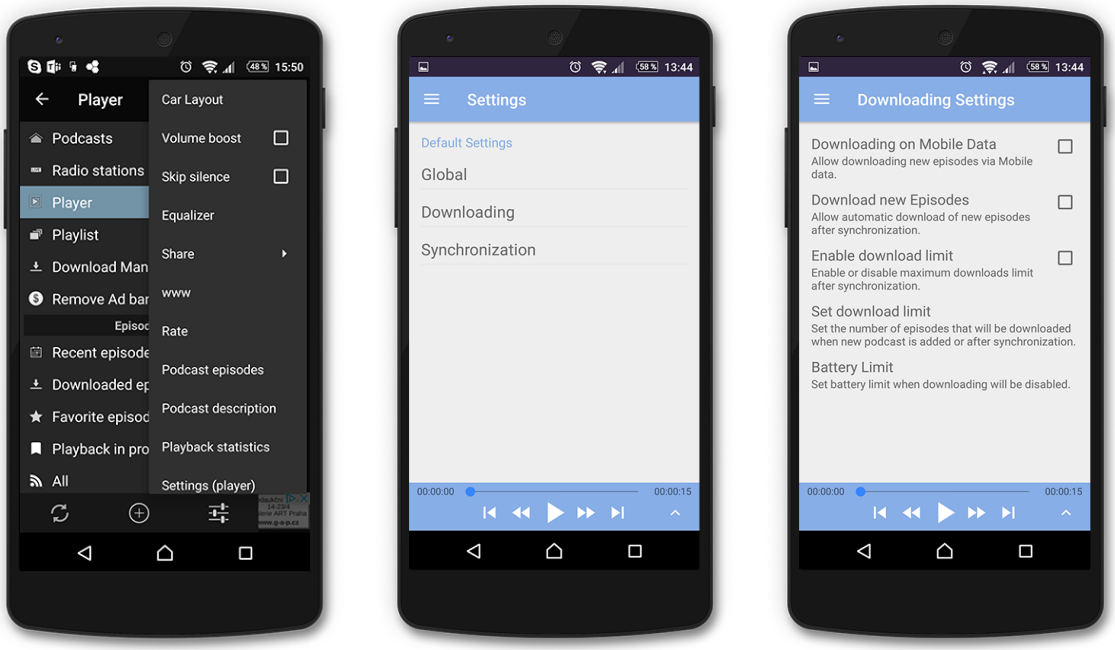
4.3.3 Zoznam podcastov a ich pridávanie

Zobrazovanie všetkých podcastov a epizód je riešené pomocou spomínaného **RecyclerView** s kombináciou **adaptéra**, ktorý zobrazuje iba potrebný počet prvkov. Samotný adaptér využíva zobrazovanie potrebných informácii pomocou ďalšej kontrolky zahrnutej v material dizajne a to konkrétne **karty** (Cards[5]). Slúžia na zobrazovanie stručných informácii o jednotlivých prvkoch. Môžu zobrazovať fotografiu, názov, krátky popis a podobne. Väčšinou sú na ne implementované ďalšie akcie, ako napríklad otvorenie detailnejšieho zobrazenia pomocou **dialógového okna** pri kliknutí na jednu z kariet.

V aplikácii MyPodcasts sa pri krátkom kliknutí na jeden z pridaných podcastov (zobrazených pomocou kariet) otvorí nové okno so zoznamom všetkých epizód, ktoré podcast zahŕňa, viz prvý snímok obrazovky na obrázku 4.5. Po otvorení nového okna sa zobrazí v hornej časti obrazovky obrázok, nadpis a krátky popis o aktuálne otvorenom podcaste. Pod ním su zobrazené všetky epizódy, ktoré podcast obsahuje, pomocou kariet, na ktoré je možné kliknúť a zobraziť tak detailnejšie informácie o epizóde. Vo vyskakovacom okne je

[4] Android RecyclerView - https://developer.android.com/reference/android/support/v7/widget/RecyclerView.html

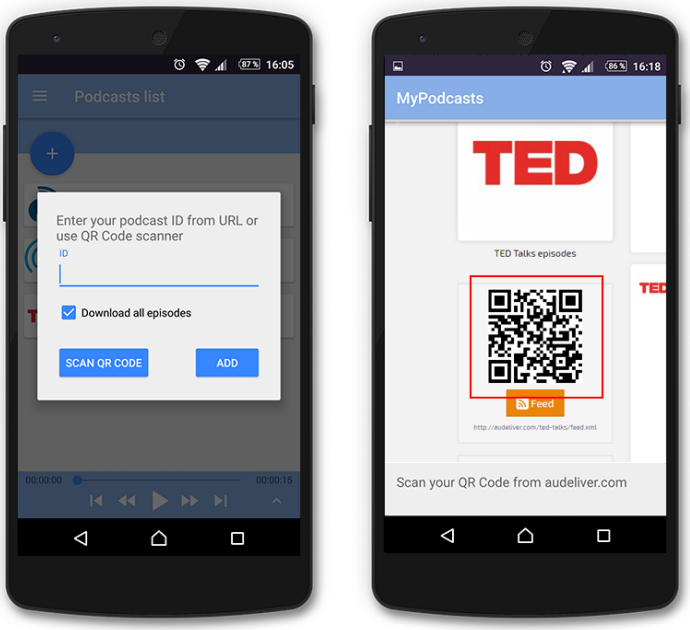
[5] ANdroid Cards - https://material.io/guidelines/components/cards.html



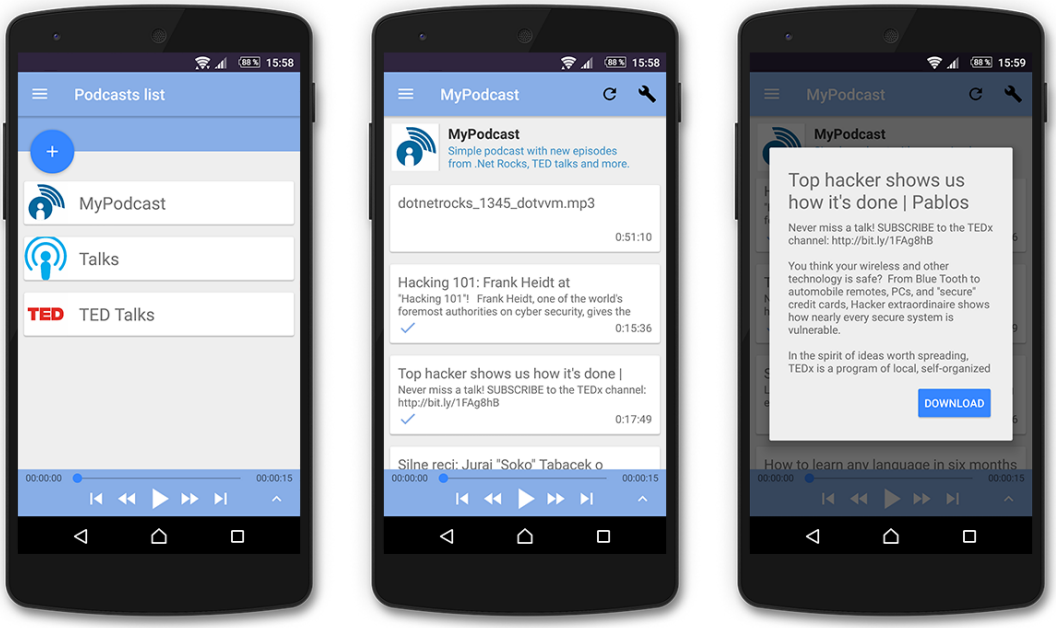
Obr. 4.3: Prvá ukážka znázorňuje zle navrhnuté užívateľské prostredie v aplikácii **Podcast Addict**, ktoré je naozaj neprehľadné a obsahuje až príliš veľa možností v ktorých sa užívateľ môže veľmi rýchlo stratiť. V druhej a tretej ukážke sú snímky obrazovky z aplikácie **MyPodcasts**, kde je znázornené zobrazovanie nastavení v prehľadnejšej a stručnejšej podobe.

aj možnosť stiahnutia konkrétnej epizódy. Ukážky všetkých obrazoviek sú znázornené na obrázku 4.5.

Na prvotnej obrazovke so zoznamom všetkých pridaných podcastov sa nachádza **plávajúce tllačidlo** (3.3.1) pomocou ktorého sa spustí akcia na pridanie nového podcastu. Zobrazí sa vyskakovacie dialógové okno so vstupom pre zadanie ID podcastu, zaškrtávacím tlačidlom na stiahnutie všetkých epizód a dvoma klasickými tlačidlami na potvrdenie akcie alebo pre pridanie nového podcastu pomocou nasnímania **QR kódu**. Toto uživateľské rozhranie bolo navrhnuté tak, aby užívateľ nemusel zbytočne vypisovať celé URL podcastu a mohol si zároveň zapnúť sťahovanie všetkých epizód jediným zaškrtnutím tlačidla. Ušetrí sa tak čas a nie sú potrebné ďalšie vsykakovacie okná. Pridávanie podcastu nasnímaním QR kódu prebiehe veľmi rýchlo a aby užívateľ nebol zmatený, či sa niečo stalo, pridalo sa potvrdenie nasnímania zavibrovaním zariadenia. To naznačí, že kód bol úspešne načítaný a spustí sa pridávanie podcastu.



Obr. 4.4: Ukážka dvoch možností, ako sa dá pridať nový podcast v aplikácii MyPodcasts. Vľavo je prvá možnosť pridania pomocou **ID** a vpravo druhá možnosť, ktorá pridá podcast hneď ako sa naskenuje jeho **QR kód** pridelený službou audeliver.



Obr. 4.5: Snímky obrazoviek jednotlivých častí aplikácie MyPodcasts. Na prvej snímke je znázornené zobrazovanie všetkých stiahnutých podcastov, na vedľajšej sú znázornené všetky epizódy, ktoré daný podcast obsahuje a na poslednom obrázku v pravo je znázornené detailnejšie zobrazenie informácii o epizóde po kliknutí na ňu.

Kapitola 5

Implementácia aplikácie

Celková implementácia aplikácie vychádzala zo špecifikácie požiadavkou a návrhu grafického riešenia popísaného v kapitole 5. V tejto kapitole budú rozpísané jednotlivé časti aplikácie, ich implementácia a funkcionalita. Na začiatku bude vysvetlené, ako aplikácia komunikuje a získava obsah zo služby Audeliver, akým spôsobom sa získaný obsah ukladá do aplikácie a ako je zabezpečená synchronizácia pre získavanie najnovšieho obsahu. Ďalej bude vysvetlené ako funguje prehrávanie stiahnutých epizód a aké komponenty a knižnice sa v aplikácii používajú.

5.1 Práca s databázou

Spracované podcasty a epizódy sa ukladajú pomocou knižnice Realm (3.4) do databázy. Táto knižnica uľahčuje vývojárom prácu s databázou a oslobodí ich od písania zložitých **SQL** príkazov. Pri spustení aplikácie je potrebné databázu inicializovať a ďalej už len pristupovať k tejto vytvorenej instancii. Na uľahčenie bola v hlavnom kontexte `MainActivity` vytvorená metóda `getRealm()`, ktorá sa o to všetko postará. Do Realm databázy sa ukladajú priamo vytvorené objekty (`RealmObject`) medzi ktorými sa môžu vytvárať väzby. Preto každý jeden podcast a aj epizóda majú pripravené svoje vlastné triedy (`PodcastModel` a `EpisodeModel`) reprezentujúce modely, ktoré sa používajú pri ich vytváraní a ukladaní do databáze.

Na obrázku 5.1 je zobrazené získanie Realm instancie, ktorá obsahuje všetky uložené záznamy. Pre uloženie nového záznamu (v tomto prípade vytvorenie nového podcastu) je potrebné vyvolať novú transakciu pomocou metódy `realm.beginTransaction()` za ktorou sa vykonajú všetky požadované operácie a následne sa transakcia ukončí a uloží pomocou `realm.commitTransaction()`.

Databáza Realm poskytuje aj jednoduché vyhľadávanie. Stačí znovu načítať jej instanciu z hlavného kontextu rovnako, ako vo všetkých iných prípadoch a pomocou dotazu s požadovanými parametrami, vráti odpoveď v podobe nájdeného objektu alebo zoznamu viacerých objektov, ktoré spĺňali zadané kritéria vyhľadávania. Ukážku implementácie môžete vidieť na obrázku 5.2.

```
Realm realm = ContextMain.context.getRealm();
realm.beginTransaction();
    newPodcast = realm.createObject(PodcastModel.class, podcastId);
    newPodcast.setName(podcastName);
    newPodcast.setUrl(podcastUrl);
    newPodcast.setXml(podcastXml);
    newPodcast.setDescribtion(podcastDescription);
    newPodcast.setCoverUrl(podcastCoverUrl);
    newPodcast.setAutomaticSynchronization(settingsModel.isAutomaticSynchronization());
    newPodcast.setAutomaticDownloads(settingsModel.isAutomaticDownloads());
    newPodcast.setDeleteFinishedEpisodes(settingsModel.isAutomaticDeletePlayedEpisode());
realm.commitTranscation();
```

Obr. 5.1: Ukážka vytvorenia spojenia s instanciou databáze **Realm** do ktorej sa následne vytvorí nový objekt obsahujúci údaje o epizóde, zísakné z **RSS zdroja**.

```
Realm realm = ContextMain.context.getRealm();
RealmResults<EpisodeModel> realmResults =
   realm.where(EpisodeModel.class).equalsTo("isDownloaded", true).findAll();
```

Obr. 5.2: Vyhľadávanie objektov v databáze Realm pomocou vytvoreného dotazu a zadanými špecifikáciami.

V tomto prípade bude zoznam `RealmResult` obsahovať všetky objekty `EpisodeModel`, ktoré majú nastavený parameter `isDownload` na hodnotu `true`. S takýmto typom zoznamu sa dá pracovať podobne ako s každým iným zoznamom, môžeme v ňom postupne iteráciou vypísať všetky jeho prvky alebo v ňom môžeme rovno vyhľadávať, získať prvý alebo posledný záznam a podobne.

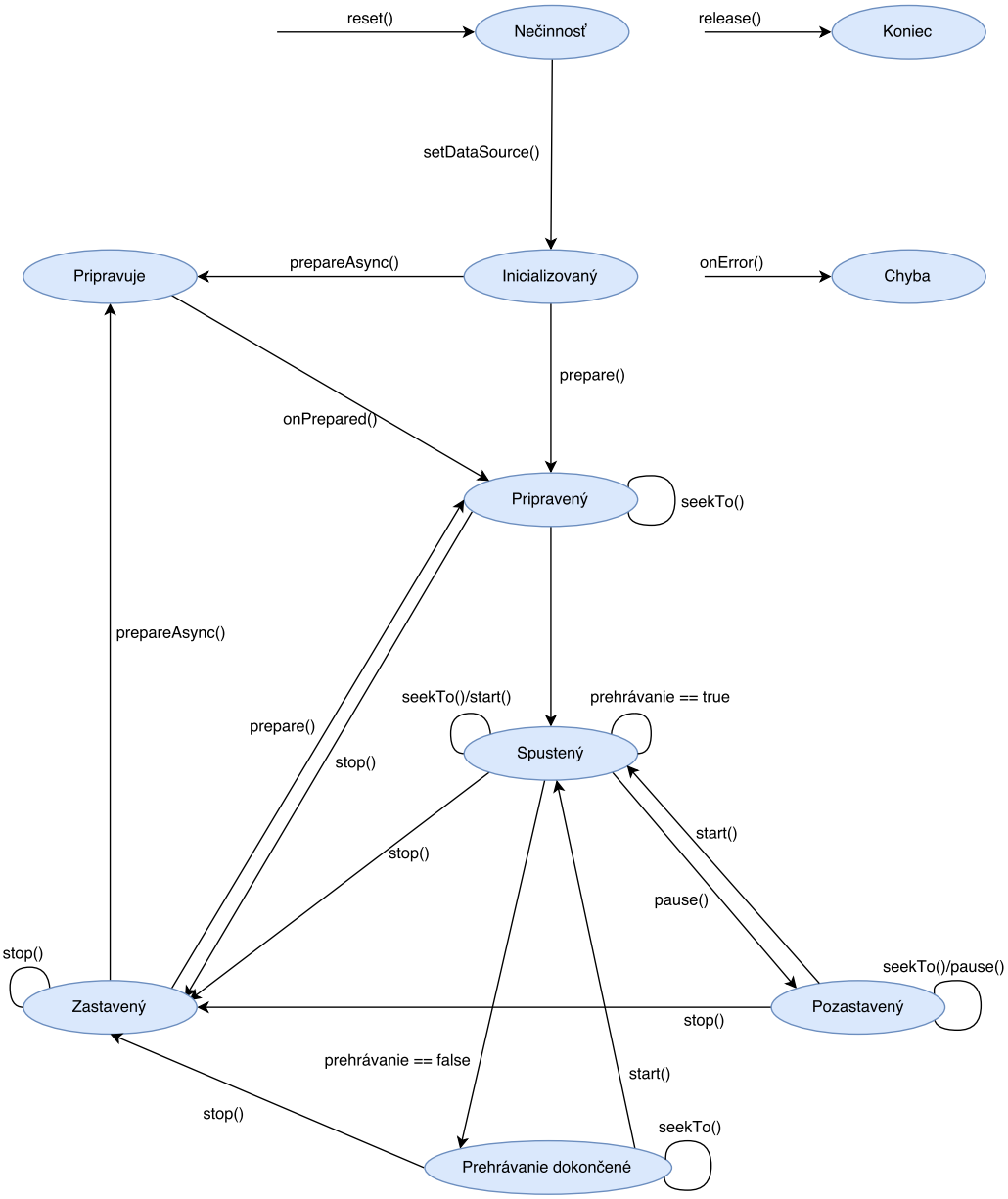
Pri ukončení celej aplikácie je potrebné zatvoriť načítanú instanciu v hlavnom kontexte aplikácie pomocou `ContextMain.context.getRealm().close()`, aby sa všetky zmeny poriadne uložili a pri ďalšom spustení mohli byť znova správne načítané. Táto operácia sa sama vykonáva v metóde `onDestroy()` pri zatvorení alebo skončení hlavnej aktivity.

5.2 Prehrávanie mediálnych súborov

Android ponúka vývojárom možnosť vytvárať jednoduché hudobné aplikácie pomocou triedy `MediaPlayer`[1], ktorá sa postará o kompletné prehrávanie video alebo audio súborov. Celé riadenie prehrávania má na starosti stavový automat, ktorý je znázornený na obrázku 5.3.

Na začiatku musí prebehnúť inicializácia samotného prehrávača, ktorá bola vytvorená v triede `MediaPlayerService`. Tá funguje ako takzvaná **služba** (`Service`) a po jej vytvorení beží po celý čas chodu aplikácie a to na pozadí, čiže aj pri minimalizovaní alebo zablokovaní obrazovky zariadenia.

[1] MediaPlayer.class - https://developer.android.com/reference/android/media/MediaPlayer.html



Obr. 5.3: Stavový automat, ktorý riadi kompletné prehrávanie audio a video súborov pomocou Android knižnice `MediaPlayer`

`MediaPlayerService` sa spustí hneď pri zapnutí aplikácie v hlavnej aktivite, pri samotnej inicializácii. K spustenej službe sa môže pristupovať pomocou vytvorených verejných funkcií a tým sa zabezpečí aj jej ovládanie.

Ak užívateľ spustí prehrávanie epizódy v aplikácii, musí sa najprv v databáze Realm nájsť konkrétna epizóda a následne je potrebné vytvorenému prehrávaču poskytnúť priamu cestu do adresára, odkiaľ ju môže načítať. Po úspešnom načítaní súboru prehrávač prejde do stavu `Prepared` (pripravený) a prehrávanie je možné spustiť.

Až do ukončenia sa nachádza prehrávač v stave `Started` z ktorého môže prejsť do troch ďalších stavov a to `Paused` v prípade pozastavenia prehrávania, `Stopped` ak bolo prehrávanie ukončené alebo `PlaybackCompleted` pri dokončení prehrávania. Pre každý jeden stav je pripravený callback (`onPrepared, onCompletetion` a podobne) v ktorej je nakonfigurované následné chovanie prehrávania. Napríklad v prípade dokončeného prehrávania sa prejde do stavu `onCompletetion()` v ktorom sa ako prvé uloží do databázy informácia, že daný záznam už bol prehraný (parameter `isPlayed`) a v aplikácii sa epizóda zvýrazni ako prehraná. V ďalšom kroku sa nájde nasledujúca epizóda, ktorá sa má prehrať, prehrávaču sa prepošle nová cesta k tejto epizóde a pomocou automatu sa prehrávač znovu dostane do stavu `Prepared` v ktorom sa spustí prehrávanie. Podobne funguje aj pozastavenie alebo stopnutie prehrávania, no epizóde sa nenastavuje parameter `isPlayed` na `true`, ale nastaví sa presný čas pozastavenia/ukončenia do parametru `endTime` v milisekundách. Tento čas je veľmi dôležitý, keďže pri ďalšom prehrávaní rovnakej epizódy sa po jej načítaní v prehrávači vždy pokračuje od nastaveného času. Tento čas môže užívateľ vidieť v percentuálnej podobe pri každej jednej stiahnutej epizóde a má tak prehľad o rozpočúvaných alebo vypočutých záznamoch, viď obrázok 4.2.

5.3 Získavanie obsahu zo služby Audeliver

Služba **Audeliver** ponúka verejné **API** pomocou ktorého môžu vývojári získavať a upravovať obsah. V aplikácii sa táto služba nevyužíva, keďže hlavným cieľom bolo vytvoriť aplikáciu, kde nebude potrebné žiadne prihlasovanie a obsah sa bude dať získavať napríklad pomocou **ID** konkrétneho podcastu alebo naskenovaním **QR kódu**.

Pri pridávaní nového podcastu sa využíva v aplikácii vyskakovacie dialógové okno, ktoré obsahuje informácie pre užívateľa, ako môže vložiť nový podcast. Služba audeliver generuje **URL odkazy**, ktoré sú vždy rovnaké, ale mení sa len ID konkrétneho podcastu, napríklad `http://audeliver.com/ted-talks/feed.xml`. V tomto príklade ID reprezentuje názov `ted-talks`. Pomocou neho je možné vložiť nový podcast do aplikácie. Druhá možnosť je naskenovať QR kód, ktorý obsahuje rovnakú **URL adresu**. Ukážky z aplikácie je možné vidieť na obrázku 4.4. Ak užívateľ potvrdil pridávanie nového podcastu, o všetko ostatné sa už postará trieda `SavePodcast`. Táto trieda obsahuje konštruktor, v ktorom sa nastaví ako hlavný parameter získané URL a doplnková `boolean` hodnota s údajom, či sa majú stiahnuť všetky epizódy po dokončení vytvárania podcastu. Získanie obsahu z URL sa spustí automaticky pomocou metódy `saveIt()`, ktorá využíva asynchrónnu službu `XmlGet` a `XmlDownload`. V nich sa vytvorí nová **HTTP žiadosť** [1] pomocou ktorej sa získa obsah zo zadanej URL a ten sa následne kompletne celý uloží do pomocného `stringu`.

Po získaní celého obsahu sa spustí vytváranie samotného podcastu. Získaný obsah uložený do stringu je potrebné rozparsovať a práve na to slúži trieda `XmlParser`, ktorá dokáže vyhľadávať v celom stringu potrebné elementy identifikujúce napríklad názov alebo popis podcastu. V ukážke 2.1 (str. 4) je možné vidieť, ako sú jednotlivé elementy rozmiestnené. Po získaní všetkých potrebných elementov sa môže vytvoriť nový `PodcastModel` objekt, ktorý sa následne uloží do databázy. Skrátené verzie jednotlivých metód je možné vidieť na obrázkoch 5.4, 5.5, 5.6 a ukladanie do databázy bolo taktiež znázornené na obrázku 5.1.

```
public void saveIt() {
    String uri = "https://audeliver.com/" + podcastUrl + "/feed.xml";
    new XmlGet(uri) {...}.execute();
}
```

Obr. 5.4: Metóda slúžiaca na uloženie nového podcastu. Zadané ID sa pridá do kompletného **URL** a spustí sa získavanie celého jeho obsahu pomocou funkcie `XmlGet()`.

```
public class XmlDownload() extends AsyncTask<String, Void, String> {
    @Override
    protected String doInBackground(String... uri) {
        try {
            HttpURLConnection urlConnection = (HttpURLConnection) uri.openConnection();
            BufferedReader reader =
                    new BufferedReader(new
                     ↪  InputStreamReader(urlConnection.getInputStream()));

            while ((String line = reader.readLine()) != null) {
                builder.append(line + '\n');
            }
            reader.close();
            return builder.toString();
        }
    }
}
```

Obr. 5.5: Získanie **RSS** obsahou pomocou metódy `XmlDownload()` do pomocného stringu, ktorý sa následne odošle na parsovanie.

```
1  public class XmlParser() {
2      // Postupne prechádza všetkými uzlami a vracia ich hodnoty
3      public final String getElementValue( Node elem ) {
4          Node child;
5          if( elem != null){
6              if (elem.hasChildNodes()){
7                  for( child = elem.getFirstChild(); child != null; child =
                    ↪  child.getNextSibling() ){
8                      if( child.getNodeType() == Node.TEXT_NODE  ){
9                          return child.getNodeValue();
10                     }
11                 }
12             }
13         }
14         return "";
15     }
16 }
```

Obr. 5.6: Metóda `XmlParser()` slúži na získanie potrebných informácii o podcaste alebo epizóde. Postupne prechádza všetky uzly a elementy v **RSS** súbore a vyťahuje iba potrebné údaje.

5.4 Sťahovanie súborov

Ak sa užívateľ rozhodne stiahnuť epizódu kliknutím na tlačidlo **Downlaod** (Stiahnúť) v informačnom dialógovom okne epizódy, vyvolá tak akciu, ktorá spustí triedu `DownloadEpisode`. V konštruktore sa podľa zadaného parametru získa z databázy odkaz na objekt, reprezentujúci konkrétnu epizódu. Ak sa epizóda našla, skontroluje sa podľa nastavení v aplikácii, či je zariadenie pripojené na internet a o aký typ pripojenia sa jedná. Ak je v nastaveniach vypnuté sťahovanie nových epizód cez mobilné dáta a momentálne je zariadenie pripojené na internet práve pomocou nich, sťahovanie sa nespustí. Skontroluje sa aj stav batérie v zariadení s minimálnou potrebnou hodnotou zvolenou v nastaveniach. Ak bolo zabezpečené potrebné internetové pripojenie a je dostatok batérie, prejde sa na samotné sťahovanie epizódy. V opačnom prípade sa vypíše chybové hlásenie pomocou `Toast`[2] (vyskakovacie okno) aby informoval užívateľa o chybe ktorá vznikla a mohol ju opraviť.

Metóda `downloadEpisode` obsahuje službu **DownloadManager**[3], ktorá zabezpečuje celý proces sťahovania súborov. Na začiatku je potrebné nastaviť **URI**[4] s odkazom na konkrétnu epizódu, odkiaľ ju je možné stiahnuť. Tento odkaz sa priloží službe DownloadManager a vytvorí sa tak nová žiadosť pre sťahovanie. Nastavia sa všetky potrebné parametre, ako napríklad názov sťahovanej epizódy, popis sťahovania alebo miesto uloženia v mobilnom zariadení. Názov a popis je dôležitý pre notifikačnú lištu, ktorú si služba sama vytvorí a pri spustení sťahovania ju zobrazí v hornej lište zariadenia s informáciami o aktuálnom

[2] Android Toast – https://developer.android.com/guide/topics/ui/notifiers/toasts.html

[3] Android DownloadManager – https://developer.android.com/reference/android/app/DownloadManager.html

[4] URI – https://tools.ietf.org/html/rfc3986

stave sťahovania. Po vyplnení všetkých parametrov sa vytvorená žiadosť odošle a keď bude služba pripravená, spustí sa sťahovanie. Ukážka implementácie je zobrazená na obrázku 5.7.

Aby aplikácia mala prehľad o aktuálnom stave sťahovania, bol vytvorený **BroadcastReceiver**[5], ktorý má na starosti odchytávanie všetkých udalostí o zmene stavu sťahovania. V prípade úspešného sťahovania sa pre danú epizódu upravia parametre v databáze a nastaví sa cesta k stiahnutému súboru.

```
1  public void downloadEpisode() {
2      DownloadManager downloadManager =
3          (DownloadManager) ContextMain.context.getSystemService(Context.DOWNLOAD_SERVICE);
4
5      Uri uri = Uri.parse(episode.getUrl());
6      DownloadManager.Request request = new DownloadManager.Request(uri);
7
8      request.setAllowedNetworkTypes(DownloadManager.Request.NETWORK_WIFI |
       ↪  DownloadManager.Request.NETWORK_MOBILE);
9      request.setAllowedOverRoaming(false);
10     request.setTitle("Audeliver Player downloading");
11     request.setDescription(episode.getName());
12     request.setDestinationInExternalFilesDir(ContextMain.context, "/Episodes",
       ↪  episode.getId() + ".mp3");
13     downloadManager.enqueue(request);
14
15     // Epizóda bola úspešne stiahnutá. Upravia sa jej parametre v databáze
16     BroadcastReceiver receiver = new BroadcastReceiver() {
17         @Override
18         public void onReceive(Context context, Intent intent) {
19            File path  = new File(ContextMain.context.getExternalFilesDir("/Episodes"),
              ↪  episode.getId() + ".mp3");
20            if (path.exists()) {
21                Realm realm = ContextMain.context.getRealm();
22                realm.beginTransaction();
23                episode.setDownloaded(true);
24                episode.setPath(path.getPath());
25                realm.commitTransaction();
26             }
27         }
28     };
29 }
```

Obr. 5.7: Ukážka implementácie sťahovania nových podcastov pomocou Android služby **DownloadManager**. Vytvorí sa nová HTTP žiadosť a získa sa potrebný obsah zo zadaného URL.

[5] Android BroadcastReceiver – https://developer.android.com/reference/android/content/BroadcastReceiver.html

5.5 Synchronizácia podcastov

Na synchronizáciu podcastov bola vytvorená služba na pozadí `SynchronizationService` vďaka ktorej môže užívateľ naďalej pracovať s aplikáciou aj počas celej synchronizácie. Využívajú sa v nej vyššie spomínané funkcie na získavanie obsahu z URL (kap. 5.3) a samotné stiahnutie nových epizód (kap. 5.4).

Jedinou novou funkciou tejto služby je porovnávanie získaného obsahu s uloženým obsahom v databáze. Zisťuje sa, či naozaj ešte existujú všetky epizódy uložené v databáze aj v reálnom podcaste zo služby audeliver. V prípade, že z nej bola niektorá z epizód vymazaná, bude tak vykonané aj v aplikácii. Naopak ak bola pridaná nová epizóda a nie je ešte uložená v databáze aplikácie, spustí sa jej ukladanie a aj sťahovanie, ak je povolené v nastaveniach.

5.6 Nastavenia aplikácie

Významnou časťou aplikácie sú jej **nastavenia**, ktoré majú užívateľom uľahčiť prácu pri synchronizácii, sťahovaní a zároveň aj zabezpečiť, aby aplikácia zbytočne nečerpala mobilné dáta alebo nevybíjala batériu.

Nastavenia obsahujú tri základné kategórie: **globálne nastavenia**, **sťahovanie** a **synchronizácia**. V globálnych nastaveniach je možné povoliť automatické zmazanie epizódy zo zariadenia po jej kompletnom vypočutí. Záznam o epizóde ostane uložený v databáze a bude mať nastavený parameter, že už bol niekedy vypočutý, takže užívateľ nestratí prehľad o jeho vypočutých epizódach. Táto funkcia môže ušetriť miesto v zariadení, ktoré môže byť potrebné napríklad pri ďalšej synchronizácii a sťahovaní najnovších epizód alebo pre iné aplikácie. Ak si užívateľ nezapne automatické mazanie prehraných epizód, globálne nastavenia ponúkajú aj druhú možnosť a to zmazanie všetkých vypočutých epizód v rámci každého jedného podcastu. Poslednou možnosť je nastavenie základného času o koľko sa dá posúvať v rámci prehrávania (**SeekTime**).

V nastaveniach o sťahovaní je možné povoliť sťahovanie nových epizód cez mobilné dáta alebo zapnúť automatické sťahovanie nových epizód pri synchronizácii. Táto možnosť zabezpečí, že užívateľ bude mať vždy na dosah najnovšie epizódy a nemusí ich ručne sťahovať. V prípade, že sa do podcastu pridávajú epizódy veľmi často, dá sa nastaviť maximálny počet stiahnutých epizód pri synchronizácii. To znamená ak sa pridalo do niektorého z podcastu 20 nových epizód, ale užívateľ chce stiahnuť iba najnovších 5, má túto možnosť v nastaveniach a môže si tak znovu ušetriť miesto v pamäti telefónu. Ďalšou výhodou je šetrenie batérie zariadenia, ktoré sa môže zabezpečiť nastavením minimálnej potrebnej úrovne batérie na povolenie sťahovania. Východzia hodnota je nastavená na 20% a ak je stav batérie nižší ako táto hodnota, sťahovanie nebude povolené a užívateľ bude upozornený pomocou **Toast** hlásenia.

Nastavenia synchronizácie ponúkajú rovnako možnosť povolenia pri zapnutých mobilných dátach a nastavenia minimálnej potrebnej úrovne batérie, ako to bolo pri nastaveniach sťahovania. Ďalšou možnosť pre užívateľa je zapnúť automatickú synchronizáciu pridaných podcastov pri zapnutí aplikácie.

5.7 Nastavenia Podcastov

Aplikácia ponúka sadu primárnych nastavení, ktoré sa berú ako východzie, ale niektoré z nich môžu byť nastavené rozdielne pre každý jeden pridaný podcast. Užívateľ si tak môže

zapnúť automatické sťahovanie epizód iba v rámci jedného alebo viacerých podcastov. Ponúka mu tak možnosť pridania viacerých podcastov do aplikácie, ale automatické získavanie nových epizód si môže zapnúť iba na niektorých z nich. Rovnako tak môže byť zapnuté aj mazanie vypočutých epizód iba v rámci niektorých podcastov alebo čas o koľko sa môže presúvať v rámci epizódy (SeekTime) pre daný podcast.

5.8 Ostatné komponenty

- **QrScannerActivity** – Obsahuje implementované rozhranie pre voľne dostupnú knižnicu `MobileVisionBarcodeScanner`[6]. Celá aktivita slúži na získanie prístupu ku fotoaparátu zariadenia a následne získanie požadovaného URL z naskenovaného QR kódu.
- **DeleteFile** – Slúži na korektné odstraňovanie stiahnutých súborov zo zariadenia pomocou ID epizódy. Obsahuje zároveň aj možnosť celkového odstránenia epizódy aj z databázy.
- **GetNextID** – Jedná sa o veľmi jednoduchú, ale dôležitú funkciu, ktorá má na starosti vyhľadávanie všetkých epizód v databáze a vráti unikátne ID pre novú epizódu
- **Picasso** – Knižnica slúžiaca na zobrazovanie obrázkov. Dokáže zobraziť obrázok vložený pomocou URL bez potrebného sťahovania do zariadenia. V prípade, že aplikácia je v režime offline, zobrazí sa primárne nastavený obrázok.

5.9 Súbor Android Manifest

Táto kapitola znázorňuje, s akými službami a oprávneniami pracuje aplikácia MyPodcasts.

Súbor Manifest sa nachádza v koreňovom adresári každej jednej aplikácie a poskytuje o nej najdôležitejšie informácie operačnému systému Android. Všetky informácie musí systém obdržať ešte pred spustením akéhokoľvek kódu aplikácie. Súbor AndroidManifest ďalej poskytuje informácie o:

- Názov balíčka aplikácie (it.janota.app.mypodcasts)
- Všetky komponenty aplikácie (služby, aktivity a podobne)
- Zoznam všetkých používaných knižníc
- Určuje povolenia na využívanie rôznych komponentov zariadenia ako je napríklad fotoaparát, vibrovanie, zapisovanie do pamäte telefónu
- Informácie o minimálnej potrebnej verzie Android API
- Údaj o názve a ikone aplikácie po nainštalovaní do zariadenia

[6] GitHub QR scanner - https://github.com/KingsMentor/MobileVisionBarcodeScanner

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="it.janota.app.mypodcasts">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.VIBRATE"/>

    <application
        android:name="it.janota.app.mypodcasts.MyPodcasterApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name="it.janota.app.mypodcasts.MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name="it.janota.app.mypodcasts.QrScannerActivity" />
        <service android:name="it.janota.app.mypodcasts.MediaPlayerService" />
        <service android:name="it.janota.app.mypodcasts.SynchronizationService" />
    </application>
</manifest>
}
```

Obr. 5.8: Výpis zo súboru `AndroidManifest.xml`, ktorý obsahuje všetky potrebné údaje o aplikácii.

Kapitola 6

Testovanie aplikácie

Celkové testovanie prebiehalo na dvoch skupinách užívateľov. V prvej skupine boli ľudia, ktorý nikdy nevyužívali aplikáciu podobného druhu a nepracovali nikdy ani s podcastami. Táto skupina dostala sadu úloh, ktoré boli zamerané hlavne na užívateľské rozhranie a samotné ovládanie aplikácie. Test nebol nijak časovo obmedzený, ale zaznamenávalo sa, koľko trvalo dokončenie jednotlivých úloh s cieľom zistiť, čo robili najväčší problém. Pozostával z úloh ako napríklad:

- Pridať nový podcast pomocou QR kódu alebo ID z url adresy
- Stiahnuť prvých 5 epizód z tohoto podcastu
- Nastaviť automatické zmazanie epizód po prehraní
- Pridať niektoré zo stiahnutých epizód do fronty prehrávania a skúsiť ich spustiť
- Zmazať ručne poslednú stiahnutú epizódu
- Skúsiť vysvetliť, čo znamenajú jednotlivé prvky zobrazené pri stiahnutom podcaste

Po dokončení testu sa zisťovalo, ktoré úlohy robili najväčší problém, či sa dokázali rýchlo zorientovať v aplikácii a ako sa im s ňou pracovalo. Na teste sa podieľalo celkovo 10 ľudí, v rôznych vekových kategóriách. Najlepšie a najrýchlejšie s aplikáciou pracovali mladý ľudia, ktorý sa dokázali rýchlo zorientovať práve vďaka používanému material dizajnu. Celkovo test dopadol veľmi dobre a jediný problém, s ktorým sa ľudia stretávali, bolo vysvetlenie niektorých grafických prvkov, keďže nikdy nepoužívali podcastové aplikácie a nevedeli ako fungujú.

V druhej skupine boli pravidelný používatelia aplikácii takéhoto typu. Obdržali testovú sadu so zložitejšími úlohami a celý test bol zameraný hlavne na funkcionalitu aplikácie. Úlohy boli typu:

- Zapnúť automatickú synchronizáciu iba pre jeden podcast
- Nastaviť stiahnutie maximálne piatich najnovších epizód a skúsiť pridať nový podcast
- Zmazať všetky vypočuté epizódy v jednom z podcastov
- Nastaviť limit batérie potrebný sťahovanie nových epizód na 50% a pre synchronizáciu na 30%

Test si vyskúšalo 7 ľudí a keďže sa jednalo o skúsenejších užívateľov, narazili pri testovaní aj na pár chýb, ktoré sa následne odstránili. Aplikácia ich zaujala a mali záujem o jej používanie aj v budúcnosti. Dostal som aj pár návrhov na možné vylepšenia, ktoré by som chcel v budúcnosti implementovať. Jednalo sa napríklad o možnosť zapnutia automatického mazania starších epizód (napr. dva týždne staré epizódy) alebo možnosť vypočutia epizódy bez potrebného sťahovania.

Celkové testovanie dopadlo pozitívne a aplikácia zaujala väčšinu užívateľov. Vychvaľovali si hlavne používaný Material Dizajn pomocou ktorého sa rýchlo dokázali zorientovať v aplikácii. Ďalej sa im páčili nastavenia, ktoré aplikácia ponúka. Výber možností bol dostačujúci a dobre kategorizovaný. Ak by sa pridalo viacej nastavení, stratil by sa v nich prehľad a aplikácia by bola náročnejšia na ovládanie.

Niektorý užívatelia mali problém s určením významu niektorých grafických prvkov, preto boli nahradené ikony, napríklad pre zoraďovanie stiahnutých epizód, za lepšie znázorňujúce ikony. Dostal som aj návrh, aby bolo niekde vysvetlené, ako sa má správne pridávať nový podcast. Preto v navigačnej lište bola pridaná sekcia **About**, v ktorej je vysvetlené, ako sa dá pridať nový podcast pomocou nastaveného ID v službe Audeliver.

Pri testovaní skúsenejší užívatelia narazili na nasledujúce chyby, ktoré boli úspešne opravené:

- Automatické mazanie epizód po ich vypočutí niekedy nezmazalo súbor zo zariadenia, ale v aplikácii epizódu ukazovalo ako nestiahnutú
- Celkový čas epizód vo fronte sa zle počítal a ukazoval chybnú hodnotu
- Pri nastavení limitu pre maximálny možný počet epizód na stiahnutie po synchronizácii, prebehlo sťahovanie iba v rámci prvého pridaného podcastu a nie vo všetkých pridaných podcastov.

6.1 Publikovanie v obchode Google Play

Publikovanie aplikácie je možné pomocou webovej služby Google Play developer console, kde vývojári môžu nahrávať svoje vytvorené aplikačné súbory. Pri prvotnej registrácii je potrebné uhradiť poplatok v hodnote 25$. Po prihlásení do systému je možné založiť žiadosť o pridanie novej aplikácie, do ktorej sa vyplnia všetky potrebné údaje, ako napríklad jej názov, verzia, popis, nasnímané obrazovky pre ukážku a samotný aplikačný súbor. Google play začiatkom roka 2017 sprísnil celkové pridávanie nových aplikácii do systému. Do nedávna všetko kontrolovali skripty a automatické testy, no po novom už každú jednu žiadosť prechádzajú stážisti pracujúci pre spoločnosť Google. V mojom prípade bola prvá žiadosť o zverejnenie aplikácie zamietnutá kvôli používaniu loga a mena služby Audeliver. Bolo potrebné dodať licenčnú zmluvu alebo iné potvrdenie o spolupráci oboch strán. Keďže moja aplikácia nevyužíva API, ktoré poskytuje táto služba, došli sme k záveru, že nie je potrebné, aby sa rovnako volala a používala ich grafiku. Preto aplikácia dostala názov MyPodcasts a bola vytvorená nová žiadosť o pridanie do obchodu Google Play, ktorá už bola schválená a aplikácia následne zverejnená vo verzii Beta. V tejto verzii prebiehalo niekoľko testov. Jednalo sa len o uzavretú verziu aplikácie a na testovaní sa podieľali užívatelia, ktorý mali záujem o túto aplikáciu. Po úspešnom testovaní a opravení chýb, na ktoré užívatelia narazili, sa aplikácia zverejnila ako verejná a je možné si ju bezplatne stiahnúť aj pomocou QR kódu na obrázku (6.1).

Obr. 6.1: Vytvorený QR kód s logom aplikácie, pre jej stiahnutie priamo z obchodu Google Play.

Kapitola 7

Záver

Výsledkom tejto práce je mobilná aplikácia, ktorá dokáže spracovať podcasty vytvorené pomocou webovej služby Audeliver. Jednotlivé podcasty je možné pridávať pomocou ich ID alebo naskenovaním QR kódu bez potrebného prihlásenia. Aplikácia bola navrhnutá pomocou sprievodcu Material Dizajn, takže disponuje prehľadným grafickým rozhraním s jednoduchým ovládaním. Ponúka viaceré možnosti nastavení, ktoré sa postarajú o automatickú synchronizáciu podcastov so službou Audeliver. Užívateľ má tak vždy pripravený najnovší obsah, bez jeho zásahu. Hlavnou výhodou aplikácie je aj automatická synchronizácia po pripojení zariadenia do nabíjačky. Ďalej ponúka ukladanie dôležitých informácii o jednotlivých epizódach do databázy Realm. Užívateľ má tak prehľad o vypočutých epizódach alebo čase kde skončilo prehrávanie každej jednej epizódy. Prehrávanie môže byť spustené v rámci všetkých stiahnutých epizód alebo sa môže vytvoriť fronta prehrávania, do ktorej je možné vložiť iba tie epizódy, ktoré si chce užívateľ práve vypočuť. Prehrané epizódy môžu byť automaticky zmazané po prehratí a ušetrí sa tak miesto v zariadení.

Do aplikácie takéhoto typu by sa dalo pridať veľké množstvo funkcií, ktoré by ju mohli ešte viac vylepšiť. Na druhú stranu, ak by obsahovala priveľa možností, ovládanie by bolo náročnejšie. Keďže cieľová skupina, pre ktorú bola aplikácia určená, požadovala jednoduché sprístupnenie najnovšieho obsahu v zariadení, implementovali sa iba tie najpotrebnejšie funkcie. Dosiahlo sa tým zachovanie jednoduchého ovládania a rýchle zorientovanie sa v užívateľskom rozhraní.

Aplikácia MyPodcasts je bezplatná a neobsahuje žiadne reklamy. Do budúcna by som rád umožnil aj pridávanie podcastov vytvorených inou službou ako je Audeliver. Táto funkcionalita je skoro dokončená, no nebola ešte zverejnená kvôli potrebnému doladeniu. Ďalej by som rád pridal aj notifikačnú lištu zobrazenú pri uzamknutej obrazovke, pomocou ktorej by sa dalo ovládať prehrávanie.

Tvorba tejto aplikácie bola pre mňa veľkým prínosom. Lepšie som pochopil ako funguje objektovo orientované programovanie a hlavne jazyk Java. Naučil som sa pracovať s Material Dizajnom, databázou a ďalšími komponentami, ktoré poskytuje OS Android. V budúcnosti by som rád pokračoval vo vývoji aplikácie MyPodcasts. Vďaka veľkému záujmu o vytvorenie podobnej aplikácie aj pre zariadenia s OS Windows Phone, uvažujem o implementácii aj na túto platformu. Získal by som tak prehľad o tom, ako fungujú jednotlivé OS a v budúcnosti by som mohol skúsiť vytvárať už priamo multiplatformové aplikácie.

Literatúra

[1] Fielding, R.: *Hypertext Transfer Protocol – HTTP/1.1*. [Online; navštívené 24.03.2017]. URL https://tools.ietf.org/html/rfc2616

[2] Google: *Material design*. [Online; navštívené 28.03.2017]. URL https://material.io/guidelines/#

[3] Hardy, B.; Phillips, B.: *Android Programming: The Big Nerd Ranch Guide, 2nd Edition*. The Big Nerd Ranch, 2015, ISBN 978-0-13-417145-6, 600 s.

[4] Herodek, M.: *Android: jednoduše*. Computer Press, 2014, ISBN 978-80-251-4298-1, 128 s.

[5] Holzner, S.; J., S.: *RSS: automatické doručovaní obsahu vašich WWW stránek*. Brno : Computer Pres, 2007, ISBN 978-80-251-1479-7, 278 s.

[6] Lacko, L.: *Vývoj aplikací pro Android*. Computer Pres, 2015, ISBN 978-80-251-4347-6, 472 s.

[7] NetMarketShare: *Mobile/Tablet Top Operating System Share Trend*. [Online; navštíveno 28.03.2017]. URL https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=9&qpcustomb=1

[8] Ramel, D.: *Realm Open Sources Mobile Database, Grows It into Enterprise Platform*. [Online; navštívené 22.03.2017]. URL https://adtmag.com/articles/2016/09/27/realm-mobile-platform.aspx

[9] Ujbányai, M.; Vávrů, J.: *Programujeme pro Android*. Praha: Grada, 2012, ISBN 978-80-247-3995-3, 187 s.