FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Vývoj aplikace pro poznámky na platformě Android
Student:	Jiří Groh
Vedoucí:	Ing. Robert Pergl, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Zákazník požaduje aplikaci na zapisování poznámek pro mobilní systém Android z důvodu toho, že ostatní aplikace nabízené na platformním obchodě nesplňují jeho očekávání.

Klíčové funkcionality jsou:
- štítkování poznámek
- rozdělení poznámek na jednotlivé odstavce, které mohou být prohazovány, mazány a formátovány
- možnost přídání odškrtávacích elementů jakožto “checklist”
- text může být formátován, při zvolení odstavce je možné styl aplikovat na celý odstavec
- možnost vlastního seřazení poznámek v seznamu
- šipky na obrazovce umožňující snadné posunování kurzoru v textu

Postup:
- proveďte analýzu požadavků zákazníka
- vyberte nejvhodnější technologii pro dosažení požadovaného výsledku.
- proveďte konceptuální analýzu formou UML modelů a navrhněte architekturu
- aplikaci implementujte, řádně otestujte a nasaďte na platformní obchod
- dokumentujte své řešení

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 23. ledna 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Vývoj aplikace pro poznámky na platformě Android

Jiří Groh

Katedra Softwarového Inženýrství
Vedoucí práce: Ing. Robert Pergl, Ph.D.

13. května 2018

Poděkování

Rád bych vyjádřil poděkování vedoucímu práce Ing. Robertu Perglovi, Ph.D. za jeho čas a cenné rady.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 13. května 2018 ……

Odkaz na tuto práci

Groh, Jiří. *Vývoj aplikace pro poznámky na platformě Android*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Tato práce se zabývá vytvářením aplikace na zapisování poznámek pro zákazníka, který má představu o tom, jak by aplikace měla fungovat a vypadat. Práce se zabývá sběrem a analýzou požadavků, výběrem vhodných technologií, návrhem struktury aplikace a obrazovek, zpracováním UML modelů a nakonec samotnou implementací a testováním. Vývoj je realizován pro platformu Android v jazycích Kotlin a Java.

Klíčová slova aplikace na poznámky, android, textový editor, bohaté formátování textu

Abstract

Goal of this work is to create note-taking application for the customer. This paper describes requirement aggregation, design of architecture and layouts, creating UML diagrams, implementation and testing. The application is developed for operating system Android and is written in Kotlin and Java.

Keywords application for taking notes, android, text editor, rich-text editing

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Často vidíme, že zaneprázdnění lidé si svoje každodenní úkoly píší do poznámkovníku, ať už v papírové či elektronické podobě. V dnešní době nám chytrá zařízení nabízí funkcionality ostatních zařízení a věcí, jako například fotoaparát, přehrávač hudby a poznámkovník není výjimkou.

Zákazník ale není spokojený s nabízenými aplikacemi na platformním obchodě Google Play. Jako člověk, který si každý den zapisuje své myšlenky a své úkoly do poznámkovníku očekává od aplikace více, než jen možnost zapsat prostý text. Při denním použití vzniká velké množství textu, který je potřeba pro přehlednost udržovat formátovaný.

Aplikace může díky elektronické podobě nabídnout mnohem větší možnosti, než-li klasická tužka a papír. Poznámky mohou být formátovány podobnými nástroji, na které je člověk zvyklý z textového procesoru na svém počítači a zároveň pro přehlednost rozděleny do nezávislých odstavců. Odstavce mohou být jednoduše řazeny a mazány.

Poznámky mohou také být zaheslovány, to zamezuje přečtení nevyžádanou osobou a zároveň dovoluje použití aplikace na sdíleném zařízení, kde se obsah aplikace mění v závislosti na zadaném hesle. Nastaveno může být i hlavní heslo, které odemyká vše.

Kapitola 1

Cíl práce

Cílem je vytvořit aplikaci na přání zákazníka. Práce se bude zabývat porovnáním existujících aplikací, analýzou požadavků, návrhem aplikace, tvorbou UML modelů, a poté na základě předchozích kroků vytvořením aplikace pro operační systém Android. K cílu také patří představit aplikaci veřejnosti na platformním obchodě Google Play a umožnit tak její použití více uživateli.

1.1 Požadavky na aplikaci

Text poznámky může být formátován pomocí tzv. „rich-text editing“ ve WYSIWYG[1] editoru. Obsah poznámky může být dělen na odstavce, které lze přesouvat a mazat. Poznámky lze zamykat heslem a tak skrýt poznámku až do momentu vložení příslušného hesla. Poznámky lze vyhledávat a řadit. Jméno poznámky je implicitně tvořeno z obsahu poznámky nebo explicitně zvolené uživatelem. Poznámce lze přiřadit štítek nebo ji označit jako oblíbenou.

[1] What you see is what you get.

Kapitola 2

Analýza

2.1 Srovnání existujících aplikací

2.1.1 Evernote

Evernote je jedna z nejpopulárnějších aplikací na poznámky. Aplikace je vytvořena společností Evernote Corporation, která má jako svůj jediný produkt právě tuto aplikaci.

Tabulka 2.1: Základní informace o Evernote z Google Play [1].

Počet instalací	více než 100 000 000
Počet uživatelských hodnocení	1 482 631
Hodnocení	4.6 z 5 hvězdiček
Placené položky v aplikaci	Ano

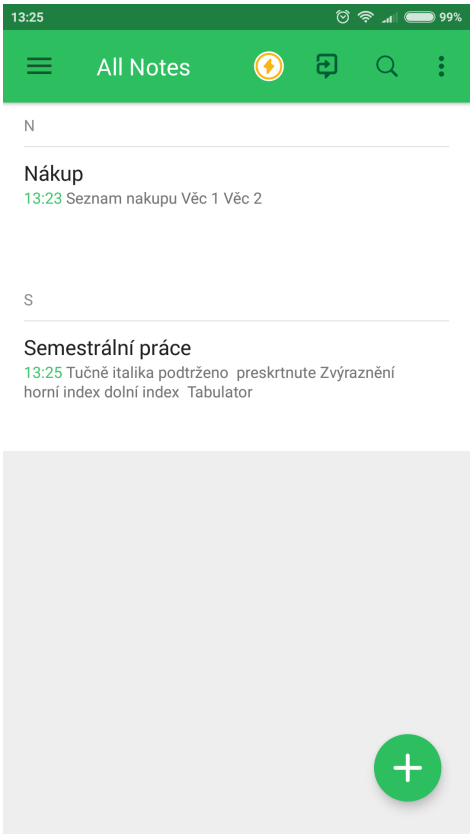
2.1.1.1 Funkce

Podpora více platforem Evernote není pouze aplikací pro mobilní zařízení. Díky webové aplikaci je možné zapisovat poznámky i na stolním počítači. Poznámky se synchronizují automaticky mezi zařízeními. Verze zdarma dovoluje synchronizovat maximálně dvě zařízení a pouze 60 MB dat za měsíc. Maximální velikost jedné poznámky je limitována na 25 MB. Pokud uživatel přesahuje některý z limitů, pak je nutné za službu platit [2].

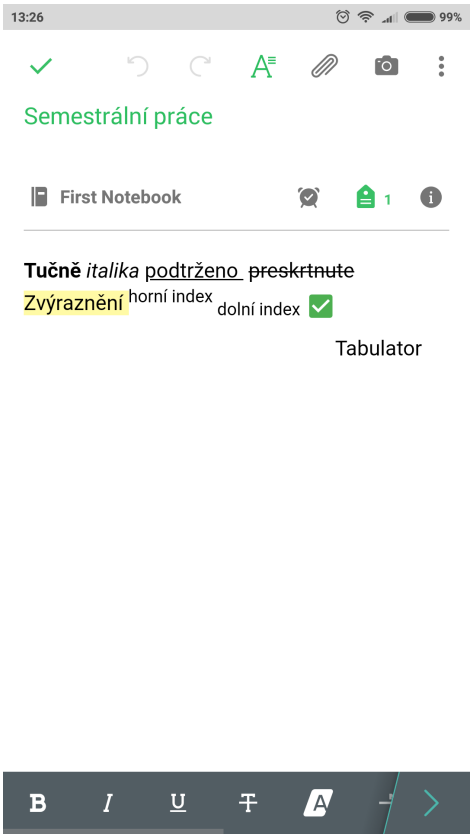
Organizace V aplikaci může mít uživatel více složek. Složka obsahuje poznámky. V poznámkách se dá vyhledávat a lze jim přiřadit štítek.

Editor poznámky Mobilní aplikace nabízí rozsáhlé množství nástrojů na úpravu textu. Nástroje je možné vidět na obrázku 2.2.

Zabezpečení Zabezpečit heslem je možné pouze celou aplikaci formou čtyřmístného PINu. Při každém spuštění je třeba PIN zadávat.



Obrázek 2.1: Seznam poznámek



Obrázek 2.2: Editor s ilustrací nástrojů

2.1.1.2 Shrnutí

Aplikace je velice populární a neustále se vyvíjí. Náročnější uživatel bude s největší pravděpodobností nucen za ní platit.

Oproti požadavkům uvedeným v sekci 1.1 nelze poznámky dělit na samostatné funkční odstavce a zaheslovávat jednotlivé poznámky.

2.1.2 Microsoft OneNote

OneNote od společnosti Microsoft je známá aplikace z balíku Office [3]. Aplikace se dočkala i provedení pro systém Android a je k dostání zdarma na platformním obchodě.

Tabulka 2.2: Základní informace o OneNote z Google Play [4].

Počet instalací	více než 100 000 000
Počet uživatelských hodnocení	441 256
Hodnocení	4.4 z 5 hvězdiček
Placené položky v aplikaci	Ne

2.1.2.1 Funkce

Synchronizace OneNote aplikace nabízí synchronizaci poznámek v rámci Microsoft účtu, to platí i pro OneNote z Office balíku.

Organizace Uživatel má podobně jako v Evernote složky, které mohou být osobní nebo sdílené mezi více lidmi. Složky se dále dělí na sekce. Sekce se nakonec dělí na stránky, ve kterých je obsah poznámek. Vyhledávání je možné jak ve složkách, tak v samotné poznámce.

Editor poznámky Editor podporuje skoro totožné nástroje jako v případě aplikace Evernote (viz obrázek 2.4). Do textu poznámky je možné kombinovat ruční kreslení, obrázky a audio nahrávky s maximální délkou tři minuty. V editoru je možné začít psát kdekoliv, v takovém případě se vytvoří sekce s textem. Pro správu širší poznámky je nutné se gesty navigovat po editoru.

Zabezpečení Samotnou aplikaci, poznámky ani složky není možné zaheslovat. Poznámky jsou spjaty s Microsoft účtem a přihlášení je nutné pouze při prvním spuštění aplikace.

2.1.2.2 Shrnutí

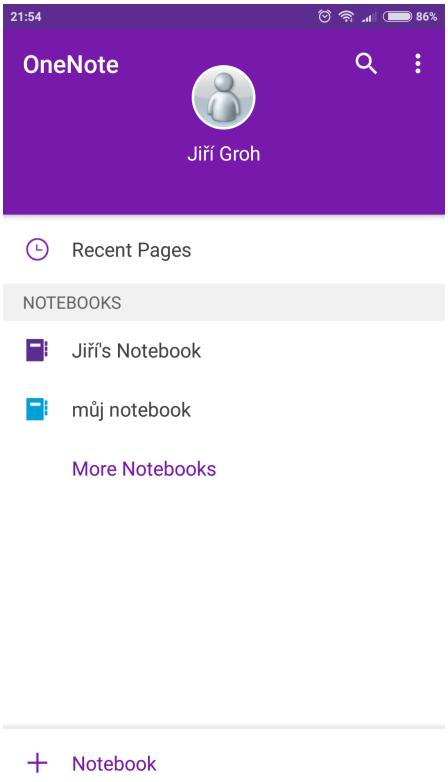
Aplikace kombinuje kreslení a vkládání obrázků kdekoliv do textu. Tím se porušuje uniformita poznámky, a to může vést k nepohodlnému užívání. Aplikaci chybí zabezpečovací prvek.

2.1.3 Google Keep

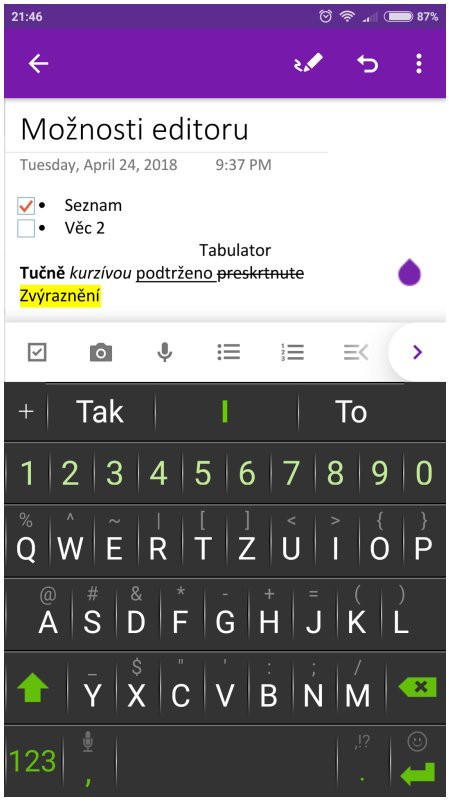
Keep je bezplatná aplikace od společnosti Google, svým provedením velice jednoduchá a přehledná. Aplikace se váže na účet Google a poznámky jsou též spjaty s účtem. Přepínání mezi Google účty je možné přímo v aplikaci. Aplikace je dostupná na webu, v prohlížeči Chrome a ve Chrome OS jako rozšíření, na iOS a Android.

2.1.3.1 Funkce

Synchronizace Google Keep nabízí neomezenou synchronizaci bez poplatků mezi všemi platformami.



Obrázek 2.3: Seznam složek



Obrázek 2.4: Editor s ilustrací nástrojů

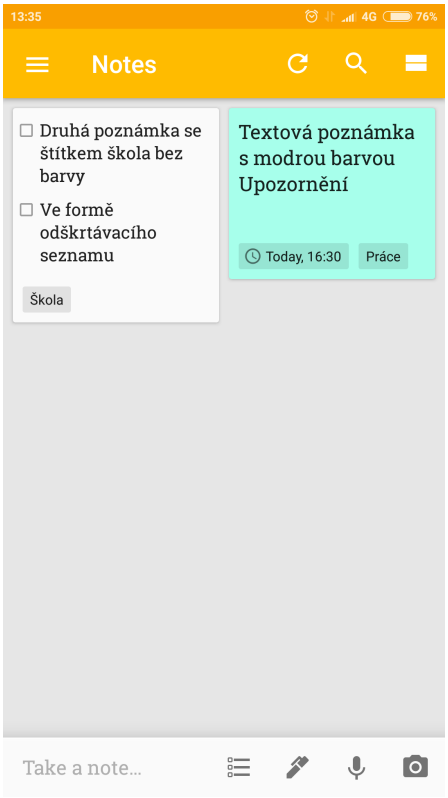
Tabulka 2.3: Základní informace o Keep z Google Play [5].

Počet instalací	více než 100 000 000
Počet uživatelských hodnocení	672 439
Hodnocení	4.4 z 5 hvězdiček
Placené položky v aplikaci	Ne

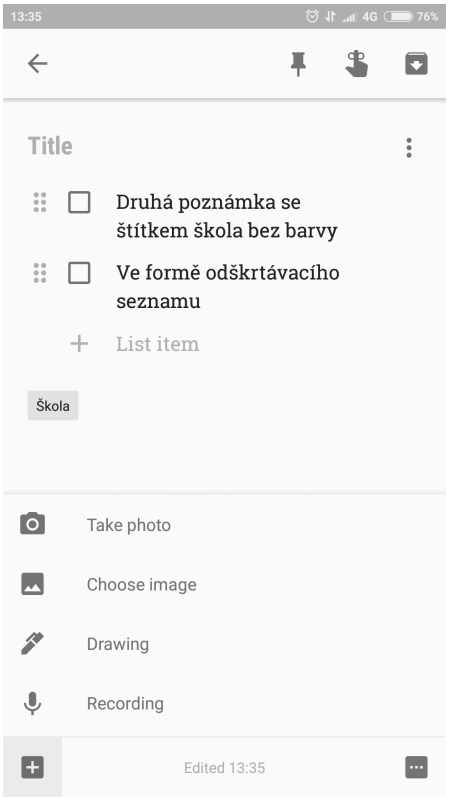
Organizace Poznámky mohou mít přiřazeno více štítků, podle kterých se poznámky dají organizovat. Důležité poznámky mohou být připnuty na začátek seznamu. Poznámky se dají organizovat také pomocí barev. Vyhledávání funguje na text, štítky a barvy.

Editor poznámky Editor nepodporuje žádné formátovací nástroje. V samotné poznámce je nutné vybrat, zda jde o odškrtávací seznam, nebo o textovou poznámku. Lze také přikládat obrázky, ruční nákresy a audionahrávky. Ty se zobrazují vždy na začátku poznámky. Text je možné diktovat hlasem. Do poznámky lze pozvat dalšího kolaboranta pomocí jeho Google účtu. Editor lze vidět na obrázku 2.6.

Zabezpečení Aplikace nemá žádné zabezpečení.



Obrázek 2.5: Seznam poznámek



Obrázek 2.6: Editor s poznámkou

2.1.3.2 Shrnutí

Keep je jednoduchá a přehledná aplikace s plnou synchronizací mezi zařízeními. Aplikace je určena spíše pro poznámky menšího rozsahu. Nelze zkombinovat text a odškrtávací seznam v jedné poznámce.

2.2 Operační systém Android

Android je open-source operační systém určený pro mobilní zařízení, fungující na linuxovém jádře. Společnost Android Inc. založená v roce 2003 začala s vývojem Androidu. O dva roky později byla společnost odkoupena společností Google. V roce 2007 vzniklo konsorcium Open Handset Alliance, které si klade za cíl vytvořit otevřený standard pro mobilní zařízení. Ve stejném roce vyšly první vývojové nástroje. Android je v současné době vyvíjen hlavně společností Google a Open Handset Alliance [6].

2.2.1 Verze

Dle [7] více než 75 % všech chytrých zařízení používá OS Android. Nejnovější verzí je Android 8 - Oreo.
Vyrobená zařízení zpravidla nedostávají aktualizace systému po celou dobu jejich životnosti[2].

[2]Závisí pouze na výrobci, zda aktualizaci poskytne.

Tabulka 2.4: Přehled jednotlivých verzí OS Android a jejich celkový podíl mezi zařízeními s OS Android. Tabulka převzata z [8].

Verze	Název (API level)	Celkový podíl (%)
2.x.x	Gingerbread (9-10)	0.3
4.0.3 - 4.3	Jelly Bean (16-18)	5.4
4.4	Jelly Bean (19)	12
5.x	Lollipop (21-22)	24.6
6.0	Marshmallow (23)	28.1
7.x	Nougat (24-25)	28.5
8.x	Oreo (26-27)	1.1

Při vytváření aplikace vzniká povinnost si vybrat nejnižší podporovanou verzi Androidu. Výběr verze je závislý na charakteru aplikace, protože některé aplikace mohou vyžadovat funkcionality, které byly představeny až v pozdější verzi Androidu.

V tabulce 2.4 je vidět, že drtivá většina zařízení v dnešní době běží na verzi 4 a vyšší.

2.2.2 Vývojové nástroje

Pro OS Android lze vyvíjet na všech populárních operačních systémech: Microsoft Windows, Linux, macOS. Pro vývoj a spouštění aplikace není nutné vlastit fyzické zařízení s OS Android.

Oficiálními programovacími jazyky pro OS Android jsou Java a Kotlin. Jazyky jsou navzájem interoperabilní. Je tedy možné pro jednu aplikaci využít oba tyto jazyky najednou [9].

2.2.2.1 Android SDK

Android SDK je kompletní balíček nástrojů pro sestavování, ladění, optimalizaci a podepisování aplikací [10].

Android Debug Bridge Můstek pro komunikaci s Android zařízením. Dovoluje ovládat připojené zařízení, instalovat/odinstalovat aplikace, pracovat se soubory či spustit příkazový řádek OS Android.

SDK Tools Nástroje pro převod zdrojových kódů na aplikaci.

Android Emulator Virtuální stroj s OS Android. Umožňuje nastavovat verzi OS, velikost obrazovky, simulovat rychlost internetu, simulovat polohu, simulovat senzory...

Apksigner Podepisuje zkompilované aplikace vývojářským klíčem pro zajištění autenticity. Podpis je vyžadován při publikaci aplikace na platformní obchod.

Android Monitor Grafické rozhraní pro monitorování zařízení. Zobrazuje informace o běžících vláknech, paměti, vypisuje žurnál zařízení a umožňuje manipulovat s úložištěm zařízení.

2.2.2.2 Android Studio

Android Studio je oficiálním vývojovým prostředím pro OS Android založené na IntelliJ IDEA od společnosti JetBrains. Android Studio v sobě zakomponovává funkcionalitu SDK a přináší velmi pohodlné využívání nástrojů mimo příkazový řádek.

2.2.3 Základní komponenty

Každá aplikace běží ve svém vlastním virtuálním prostředí, je izolována od ostatních aplikací kvůli bezpečnosti a má své privátní úložiště. Aplikace nemají navzájem přístup do svých úložišť, a proto je sdílení dat mezi aplikacemi možné jen pomocí určených struktur systému.

Aplikace má čtyři základní stavební komponenty [11].

2.2.3.1 Activity

`Activity` má na starosti UI aplikace a zpracovává uživatelské interakce s ním spojené. Jedna `Activity` má zpravidla na starosti jeden typ UI, které zabírá celou obrazovku. Přechody mezi `Activities` jsou tedy běžnou situací, a proto je důležité zabývat se jejich životním cyklem, který je možné vidět na obrázku 2.7.

Příkladem `Activity` může být seznam poznámek, další `Activity` může například obsluhovat úpravu konkrétní poznámky, kterou si uživatel zvolil v seznamu.

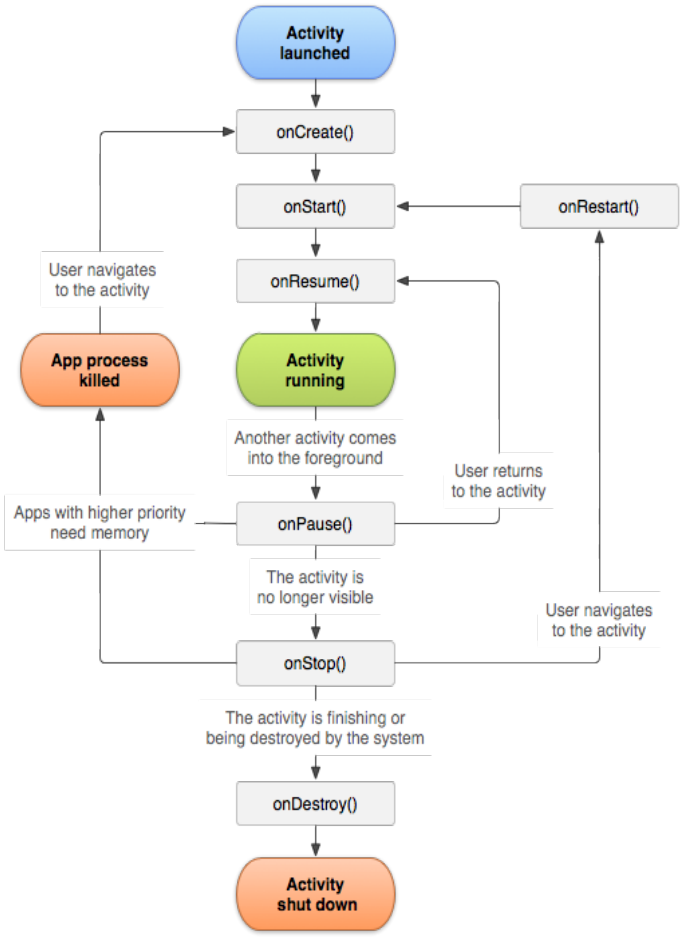
2.2.3.2 Service

`Service` je způsob, jak může aplikace běžet na pozadí a vykonávat operace bez toho, aby uživatel musel mít aplikaci otevřenou.

Typický příklad pro `Service` na pozadí je synchronizace dat se serverem.

2.2.3.3 Content Provider

`Content Provider` nabízí unifikovaný způsob přístupu k datům v databázi. Může být využíván nejen samotnou aplikací, ale i aplikacemi ostatními, pokud



Obrázek 2.7: Životní cyklus Activity [12]

je tak zamýšleno. OS Android tímto způsobem například umožňuje přístup k SMS zprávám uživatele pro aplikace třetích stran.

2.2.3.4 Broadcast Receiver

Tato komponenta umožňuje přijímat upozornění na události, které jsou vyvolány systémem nebo aplikacemi. Systém je schopen doručit událost i aplikaci, která zrovna není spuštěná a zajistit vykonání logiky spojené s událostí, a to i v případě, že je zařízení uspáno. Tímto způsobem se aplikace může dozvědět například o tom, že systém právě dokončil startování.

2.2.4 Komunikace mezi komponenty

`Intent` je komunikační objekt, který slouží ke spouštění aplikačních komponent a přenášení dat do spuštěné komponenty [13]. Data jsou ve formátu

klíč/hodnota.

2.2.4.1 Intent Filters a aplikační manifest

Každá aplikace musí mít `AndroidManifext.xml` soubor, kde jsou zaregistrovány všechny komponenty aplikace. Každá komponenta může mít v manifestu uvedené své `Intent Filters`, které specifikují jaké `Intents` je komponenta schopná přijmout a obsloužit.

Explicitní Intent obsahuje přesný název komponenty, která má být spuštěna. Zpravidla se používá uvnitř aplikace.

Implicitní Intent popisuje pouze typ akce, kterou má být komponenta schopná obsloužit. Typ akce může být například zobrazení webové stránky. Systém poté porovnává typ akce s `Intent Filters` komponent nainstalovaných aplikací. Pokud vyhovuje pouze jedna komponenta, pak je ihned spuštěna. Pokud vyhovuje více komponent, pak je uživatel vyzván k vybrání jedné z nich.

2.2.5 Databáze

Operační systém Android nabízí vestavěnou databázi SQLite. Jde o open-source relační databázi podporující transakce a jazyk SQL [14].

Databáze je uložena ve formě souboru v chráněném úložišti aplikace a není implicitně chráněna heslem ani šifrovaná.

2.2.6 Uživatelské rozhraní

Každá aplikace, která uživateli zobrazuje nějaký obsah, využívá `Activity`. `Activity` očekává právě jeden layout, který bude v danou chvíli zobrazovat. Layout je XML soubor obsahující popis `Views` a jejich atributů [15].

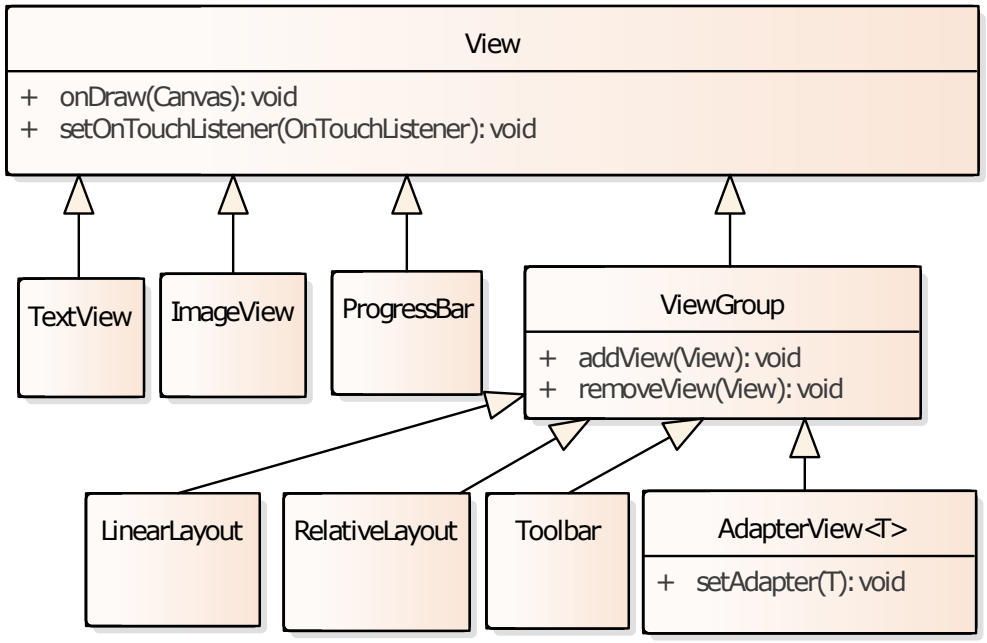
`View` je základní stavební prvek pro tvorbu UI a zajišťuje vykreslování a zpracování událostí. `ViewGroup` je třída dědící z `View`, která může obsahovat ostatní `Views` a `ViewGroups`.

Třídy dědící z `ViewGroup` jsou nazývány kontejnery [19].

Dědičnost mezi `View` a `ViewGroup` připouští následující stavy:

- `ViewGroup` může obsahovat více `ViewGroups` a více `Views`.
- `View` nemůže obsahovat žádný jiný `View`.

Android SDK již nabízí sadu předpřipravených `Views`. Jsou to tlačítka, textová pole, přepínače atd. Je také možné si vytvořit vlastní `View` s vlastní



Obrázek 2.8: Ilustrace hierarchie potomků třídy `View`. Diagram vytvořen podle [16], [17], [18]

logikou. Stejné platí o `ViewGroup`.

Aktivita se zároveň stará o události jednotlivých `Views`. `Views` lze modifikovat pouze z hlavního vlákna aplikace [20]. Procesu, kdy hierarchie layoutu přechází z XML do podoby, kterou je možné vykreslit, se nazývá nafukování (*inflation*). Nafukování převádí elementy zapsané v XML souboru na objekty typu `View` [21].

Čím více se strom layoutu stává hlubším, tím více se nafukovaní stává dražší operací.

K dynamickému přidávání slouží `Adapters` a `Views` dědící z `AdapterView` (viz. obrázek 2.8). Jejich použití zajišťuje plynulé animace při rolování a nedochází ke zbytečnému nafukování `Views`, protože jsou recyklovány.

2.2.7 Fragment

`Fragment` představuje modulární část v `Activity`. `Fragment` může nebo nemusí obsahovat layout, a tedy být nebo nebýt součástí vykresleného UI. Každý `Fragment` má vlastní životní cyklus, který je přímo ovlivněn životním cyklem `Activity`, ke které přísluší. `Activity` může obsahovat více aktivních `Fragments` najednou nebo je střídat na obrazovce. `Fragment` dokáže přežít zničení nadřazené `Activity` (například při otočení obrazovky) a být opět přidán do nově vytvořené instance [22].

2.3 Technologie pro Android

Android se těší velké komunitě schopných vývojářů, kteří přispívají různými knihovnami, které rozšiřují či ulehčují vývoj aplikace.

2.3.1 Programovací jazyk

Jak už bylo zmíněno v sekci 2.2.2, OS Android oficiálně podporuje jazyky Java a Kotlin. První stabilní verze Kotlinu vyšla v roce 2016 [23].

Hlavní výhodou Kotlinu nad jazykem Java je přehlednější kód a automatické generování zapouzdření proměnných a null-safety [24].

2.3.1.1 Ukázka

Následující dvě ukázky třídy `User` jsou ekvivalentní. Zde je vidět generování zapouzdření a celkové zpřehlednění kódu.

```
//Kotlin
data class User(val name: String, val surname: String)
```

```
//Java POJO class
class User
{
    private String name;
    private String surname;
    public User(String name, String surname) {
        this.name = name;
        this.surname = surname;
    }

    public String getName() {
        return this.name;
    }
    public void setName(String newName) {
        this.name = newName;
    }
    public String getSurname() {
        return this.surname;
    }
    public void setSurname(String newSurname) {
        this.name = newSurname;
    }
}
```

2.3.2 Rozšíření Kotlinu pro Android

Kotlin dostal od svých tvůrců ještě další rozšíření pro OS Android.

Anko usnadňuje vytváření layoutů, dialogů a zvyšuje přehlednost kódu při obsluze interakce s UI [25].

2.3.2.1 Ukázka

```
//Kotlin s Anko
button.onClick {
//obsloužení kliknutí na tlačítko
}

//Kotlin bez Anko
val button = findViewById<Button>(R.id.button)
button.setOnClickListener(object : OnClickListener() {
        override fun onClick(View v) {
        //obsloužení kliknutí na tlačítko
        }
    })
```

2.3.3 Databáze

Pro OS Android jsou dostupné knihovny, které umožňují využití ORM[3] nad SQLite (popsaném v sekci 2.2.5).

Jan Bína v [26] uvádí „V objektově orientovaných jazycích, jakým je i Java, pracujeme s daty ve formě objektů. Ty mohou obsahovat proměnné nejrůznějších datových typů, zatímco relační databáze je omezena na několik jednodušších, SQLite konkrétně na čtyři. ORM je technika převodu dat mezi řádky relační databáze a objekty objektově orientovaného jazyka.“

Rozšířené ORM knihovny:

- Room,
- GreenDao,
- ORMLite.

Autor v [26] také uvádí „GreenDAO je dle statistik druhá nejrozšířenější knihovna, která ukládádata do relační databáze. Dle testů je navíc nejrychlejší.“

[3] Objektově relační mapování

2.4 Analýza požadavků

2.4.1 Funkční požadavky

F1 - Seznam poznámek Obrazovka obsahující seznam dostupných[4] poznámek. Každý řádek bude reprezentovat jednu poznámku. Informace v řádku jsou následující:

- titulek (název poznámky),
- datum vytvoření,
- datum poslední úpravy,
- štítek.

Dále je možné provádět následující akce nad poznámkou:

- označení jako oblíbené,
- otevření editoru,
- rychlá editace s novým odstavcem na začátku nebo na konci,
- změna titulku poznámky,
- přidání nebo odebrání štítku,
- zaheslování poznámky.

F2 - Řazení poznámek Uživatel si může libovolně řadit poznámky, a poté si své uspořádání uložit.

Samotný seznam může být dodatečně řazen dle kritérií:

- vlastní uspořádání,
- naposledy upravené,
- naposledy vytvořené,
- abecedně,
- oblíbené.

F3 - Vyhledávání v poznámkách V seznamu dostupných poznámek je možné vyhledávat podle titulku poznámky. Vyhledávání ignoruje diakritiku a nerozlišuje velká a malá písmena.

F4 - Tvoření a editace poznámky Při výběru poznámky ze seznamu, či přidání nové poznámky, je otevřen editor. Základním stavebním prvkem poznámky je odstavec. Poznámka může mít libovolný počet odstavců.

Jednotlivé odstavce jsou nezávislé a obsahují formátovaný text, odrážkový nebo odškrtávací seznam.

[4] Dle platných zadaných hesel.

Poznámka má dva režimy, ve kterých lze upravovat. První z nich dovoluje upravovat text poznámky a druhý režim umožňuje pracovat s odstavci.

Text může být formátován:

- tučně,
- kurzívou,
- podtržením,
- přeškrtnutím,
- zvýrazněním,
- zarovnáním (doleva, na střed, doprava),
- do seznamů (odrážkové, číselné).

Odstavce mohou být označovány a následně nad nimi prováděny tyto operace:

- přesun o pozici nahoru či dolu,
- přesun na začátek či konec,
- smazání.

Editor také umožňuje manuální skrývání a zobrazování softwarové klávesnice a posouvání kurzoru v textu doleva a doprava tlačítky.

Při ručním uložení je možno zvolit vlastní titulek pro poznámku nebo nechat titulek vytvořit automaticky. Automatický titulek je vytvořen z prvních znaků poznámky.

F5 - Štítky Pro přehlednost mohou být jednotlivé poznámky oštítkovány. Nápis štítku si vytváří uživatel sám. Štítek může být z poznámky odebrán.

F6 - Hesla Pro ochranu obsahu může být každá poznámka zaheslována. V takovém případě není v aplikaci zobrazena do té doby, než uživatel zadá příslušné heslo.

V aplikaci je možné nastavit tři typy hesel.

Hlavní heslo dovoluje administrátorský režim a nelze s ním zamknout poznámku, ale dá se použít v případě zapomenutí uživatelského či hostovského hesla. Odemyká všechny poznámky a umožňuje odebrat heslo z jakékoliv poznámky.

Uživatelské heslo slouží k zamknutí soukromých poznámek uživatele (majitele zařízení).

Hostovské heslo je pro případ užití aplikace více lidmi. Poznámky je možno tímto heslem zamknout.

F7 - Nastavování hesel Jaká hesla může uživatel změnit je závislé na tom, která hesla má aktivovaná. S aktivovaným hostovským heslem nemůže uživatel nic měnit. S uživatelským heslem může měnit hostovské a uživatelské heslo. S aktivním hlavním heslem mohou být měněna všechna hesla.

F8 - Vkládání hesel Heslo je vkládáno z hlavního menu aplikace. Heslo je možné nechat zapamatovat, jinak platí jen do úplného vypnutí aplikace. Po vložení správného hesla se odemknou příslušné poznámky.

F9 - Zapomenutí hesel Z hlavního menu lze vynutit zapomenutí všech zadaných hesel, včetně hesel již zapamatovaných.

2.4.2 Nefunkční požadavky

N1 - Aplikace pro Android Aplikace bude vyvinuta pouze pro operační systém Android.

N2 - Automatické ukládání Rozpracovaná poznámka je uložena vždy, když uživatel opustí editor[5]. Tím se předchází nechtěné ztrátě dat.

2.5 Doménový model

Doménový model je způsob, jak popsat entity ze skutečného světa a vztahy mezi nimi, které dohromady jako celek tvoří doménový prostor. Narozdíl od diagramu případů užití, který popisuje chování, doménový model přináší strukturální pohled na věc [27].

2.5.1 Doménový model pro aplikaci

Objekty z realného světa jsou v tomto případě následující:

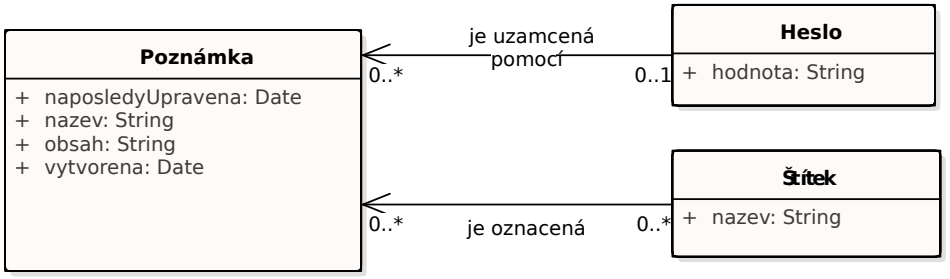
- poznámka,
- štítek,
- heslo.

Vztahy mezi nimi jsou znázorněny na obrázku 2.9.

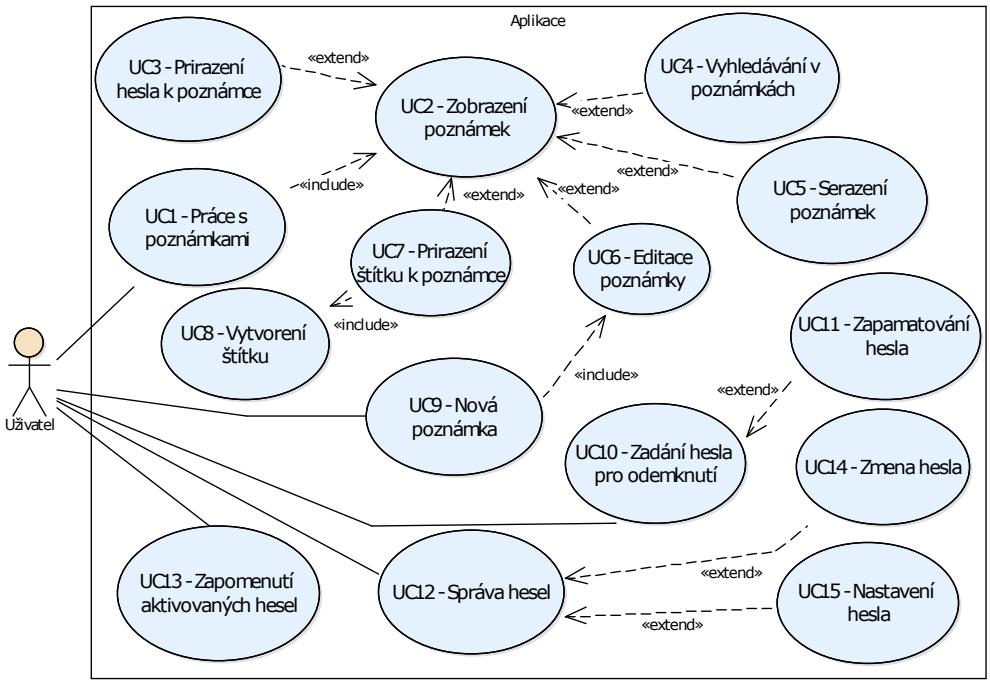
2.6 Případy užití

Účelem diagramu případů užití je vizualizovat možnosti použití aplikace. Základní stavební prvky diagramu jsou vykonavatelé, případy užití a vztahy mezi nimi [28].

[5]Pokud není nastaven titulek ručně, je použit automatický titulek.



Obrázek 2.9: Doménový model



Obrázek 2.10: Diagram případů užití

2.6.1 Popis případů užití

Diagram případů užití je na obrázku 2.10.

UC1 - Práce s poznámkami Aplikace automaticky zobrazí seznam všech dostupných poznámek.

UC2 - Zobrazení poznámek Seznam poznámek obsahující informace a akce nad poznámkami dle požadavku **F1** popsaném v sekci 2.4.

UC3 - Přiřazení hesla k poznámce Uživatel vybere akci nastavení hesla u poznámky a zvolí typ hesla, kterým bude poznámka zabezpečena.

UC4 - Vyhledávání v poznámkách V zobrazených poznámkách uživatel vybere akci hledání, poté píše hledaný výraz. Seznam zobrazuje pouze vyhovující poznámky.

UC5 - Seřazení poznámek Při zvolení akce řažení jsou uživateli představeny možnosti řazení seznamu. Uživatel si zvolí možnost a seznam se seřadí dle vybraného kritéria.

UC6 - Editace poznámky Pro danou poznámku je otevřen editor a její obsah je připraven k editaci.

UC7 - Přiřazení štítku k poznámce Uživatel zvolí přidání štítku u poznámky v seznamu. Následně uživatel zadá název štítku.

UC8 - Vytvoření štítku Pří přiřazování štítku je nutné štítek nejdříve pojmenovat a vytvořit.

UC9 - Nová poznámka Uživatel vybere vytvoření nové poznámky. Vytvoří se poznámka a aplikace ji otevře v editoru.

UC10 - Zadání hesla pro odemknutí Uživatel vybere z menu možnost zadání hesla a vepíše heslo do zobrazeného pole.

UC11 - Zapamatování hesla Při zadávání hesla pro odemknutí může uživatel nechat heslo zapamatovat.

UC12 - Správa hesel Uživatel naviguje na správu hesel a aplikace zobrazí hesla, která uživatel může aktuálně spravovat.

UC13 - Zapomenutí aktivovaných hesel Uživatel touto akcí vynutí zapomenutí všech doposud zadaných hesel, včetně zapamatovaných.

UC14 - Změna hesla Změna hodnoty hesla.

UC15 - Nastavení hesla Prvotní nastavení všech druhů hesel.

2.7 Formátování textu

Jak již bylo zmíněno v sekci 1.1, aplikace bude podporovat určitou množinu nástrojů pro práci s textem.

2.7.1 Nativní řešení v Android aplikaci

Zabudovanou komponentou pro vkládání textu v OS Android je `EditText` [29]. Komponenta reprezentuje vložený text jako rozhraní `CharSequence`, které umožňuje jak vkládat prostý text pomocí třídy `String`, tak také umožňuje

vkládat text, který má připojené formátování k jednotlivým svým úsekům, a to pomocí tříd `SpannedString` a `SpannableString`.

Úseky jsou formátovány pomocí tříd dědících z `CharacterStyle`. OS Android nabízí řadu předpřipravenených tříd, mezi ně například patří:

BulletSpan formát pro odrážkový seznam v úseku,

UnderlineSpan formát pro podtržení úseku,

ClickableSpan formát pro umožnění kliknutí na úsek[6].

Je možné vytvořit vlastní formát poděděním příslušné třídy a implementací příslušných metod [30].

Jeden z funkčních požadavků je možnost vkládat do textu zaškrtávací elementy. V OS Android opět existuje příslušná zabudovaná komponenta s názvem `CheckBox`, která je nepřímým potomkem třídy `View`. [31]

`EditText` není přímý ani nepřímý potomek třídy `ViewGroup` [29], a tudíž nemůže obsahovat žádné objekty třídy `View`. S použitím komponenty `EditText` není možné dosáhnout funkčního požadavku pro zaškrtávací elementy v textu, kvůli vztahům popsaným v sekci 2.2.6.

Řešením této situace bude vytvoření vlastní komponenty, která bude v tomto ohledu nahrazovat komponentu `EditText`.

2.7.2 Editor jako webová stránka

Webové stránky a jejich strukturovací jazyk HTML neklade tak přísné podmínky na strukturu jako OS Android. Například přidání zaškrtávacího elementu do textu v tomto případě není problémem.

Dalším přínosem tohoto řešení je implicitně zajištěná serializace obsahu poznámky ve formátu HTML, který je možno poté uložit do databáze jako prostý řetězec. Při opětovné editaci stačí z databáze přečíst tento řetězec, a poté ho zpět vložit do webové stránky.

2.7.3 Zobrazení webové stránky v aplikaci

Android také nabízí komponentu `WebView`, která umí zobrazit webovou stránku bez obvyklých navigačních a jiných prvků webového prohlížeče. Její použití tedy představuje bezešvý přechod mezi aplikací a webovou stránkou [32].

[6] Například pro URL odkaz.

Kapitola 3

Návrh aplikace

3.1 Perzistence dat

Pro ukládání strukturovaných dat bude sloužit databáze popsaná v sekci 2.2.5. Nad databází bude použita ORM knihovna.

ORM se automaticky stará o vytvoření tabulek v databázi. Generování tabulek probíhá pomocí tříd a anotací. Předlohou budou třídy, které jsou svojí strukturou velmi podobné doménovému modelu.

Nestrukturovaná data, jako jsou zadaná hesla uživatele a nastavení aplikace, jsou ve formátu klíč/hodnota, a proto je pro ně není vhodné vytvářet tabulky v relační databázi. Tato data budou uložena v perzistentní vrstvě `Preferences`, která podporuje právě tento formát. Vrstva je obsluhována systémem.

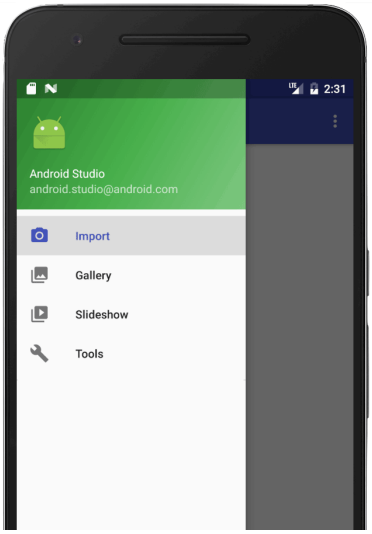
3.2 Design

Design a UI aplikace budou tvořeny v souladu s Material Design a Android guidelines.

Aplikace bude obsahovat hlavní menu (ilustraci hlavního menu je možné vidět na obrázku 3.1), které je implicitně skryté. Menu je vyvoláno gestem přejetí z levé strany obrazovky nebo po kliknutí na ikonu.

3.3 Hesla

Pro zrychlení a zjednodušení práce uživatele budou hesla koncipována formou PINu. PIN bude čtyřmístný a bude obsahovat pouze číslice 0–9.



Obrázek 3.1: Ukázka hlavního menu. Převzato z [33]

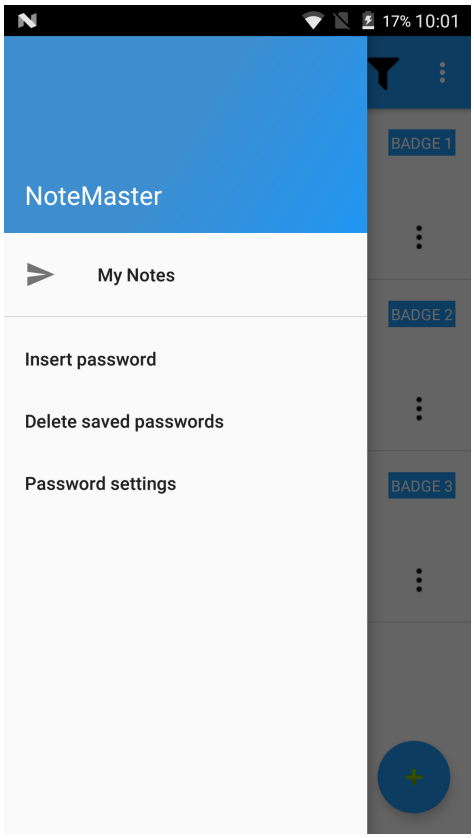
3.4 Události v aplikaci

V aplikaci bude více komponent pracovat se stejnými daty. Pro bezproblémový běh budou jednotlivé komponenty upozorňovat o změnách v datech.

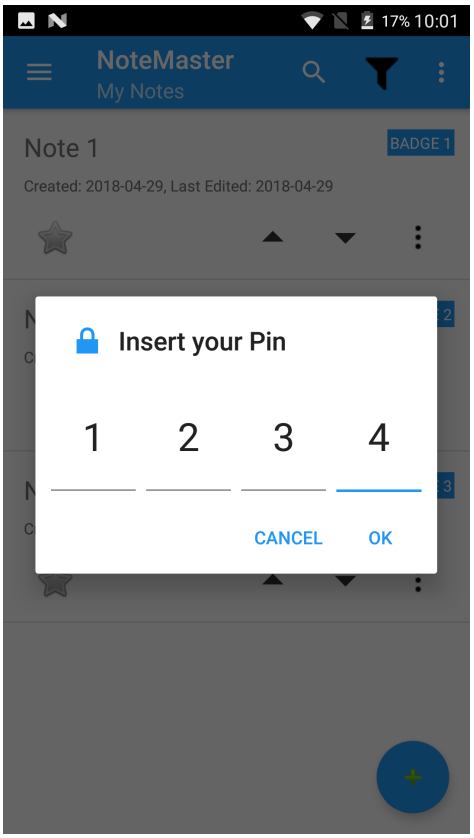
Pro snížení úrovně provázání jednotlivých komponent a zajištění rozšiřitelnosti a udržitelnosti kódu bude pro události použit návrhový vzor *Publish-subscribe*.

Tento návrhový vzor funguje na principu hlášení (*publish*) a naslouchání (*subscribe*). Při změně dat dojde k vyslání hlášení. Všechny komponenty, které na toto hlášení poslouchají, budou moci provést svoji logiku spojenou s tímto hlášením.

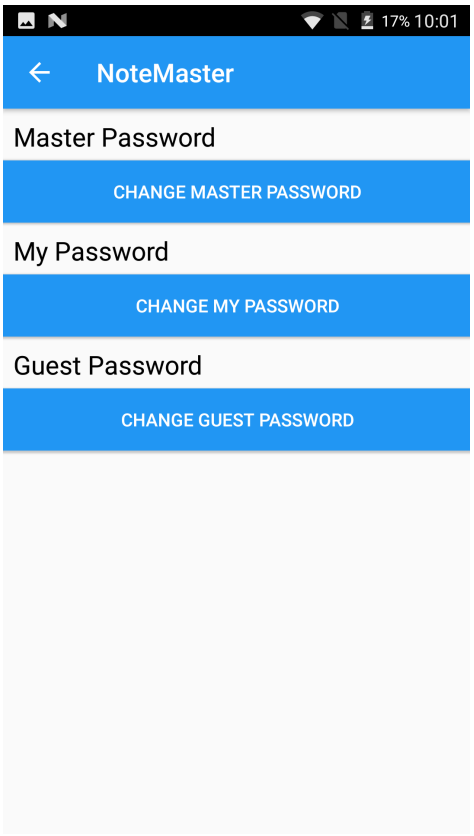
3.5 Návrh obrazovek



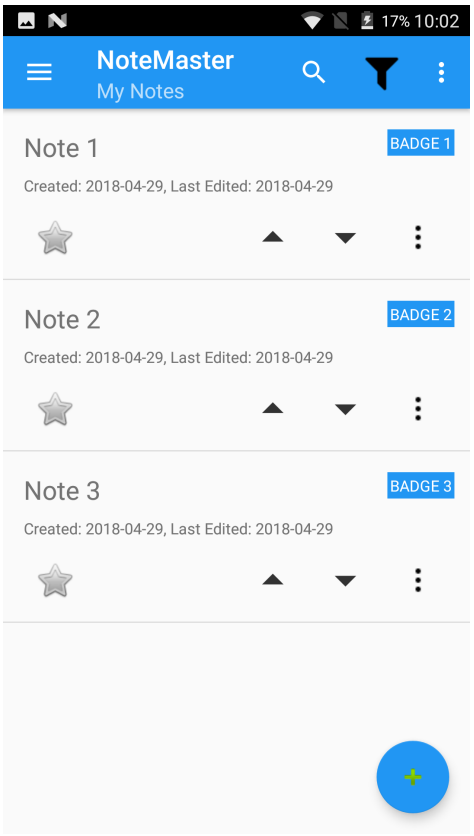
Obrázek 3.2: Hlavní menu aplikace



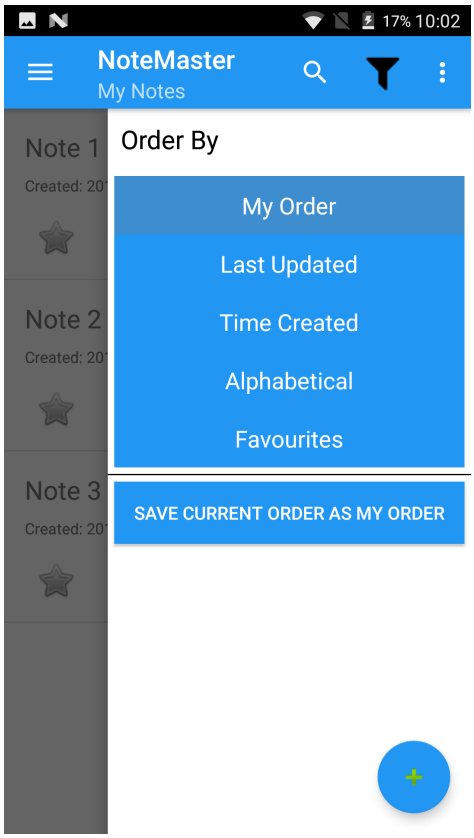
Obrázek 3.3: Dialog pro vkládání hesla



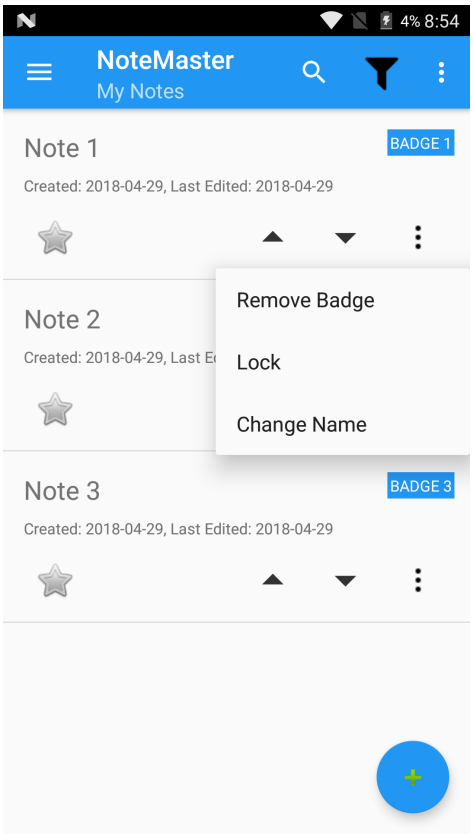
Obrázek 3.4: Správa hesel



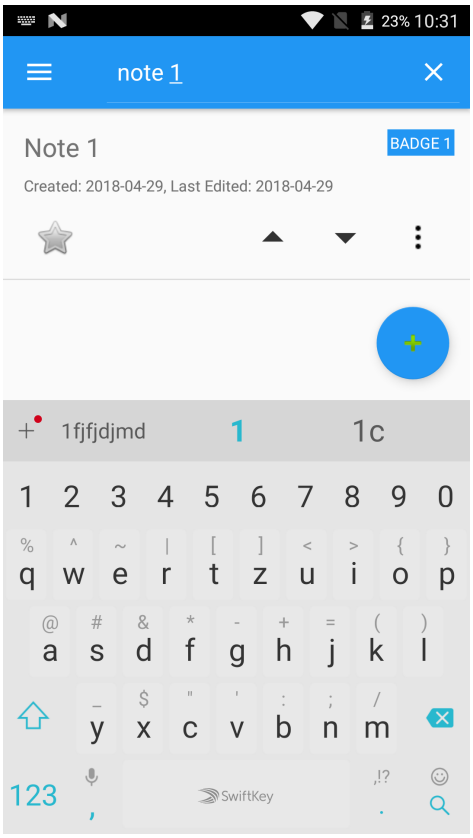
Obrázek 3.5: Výpis poznámek



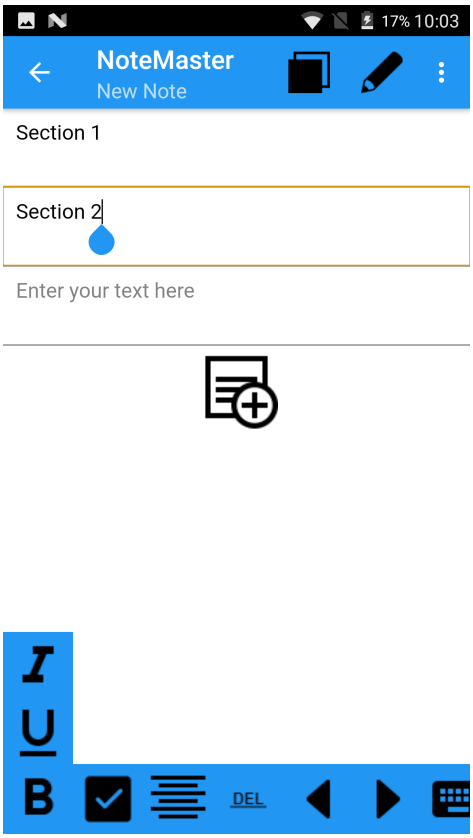
Obrázek 3.6: Řazení poznámek



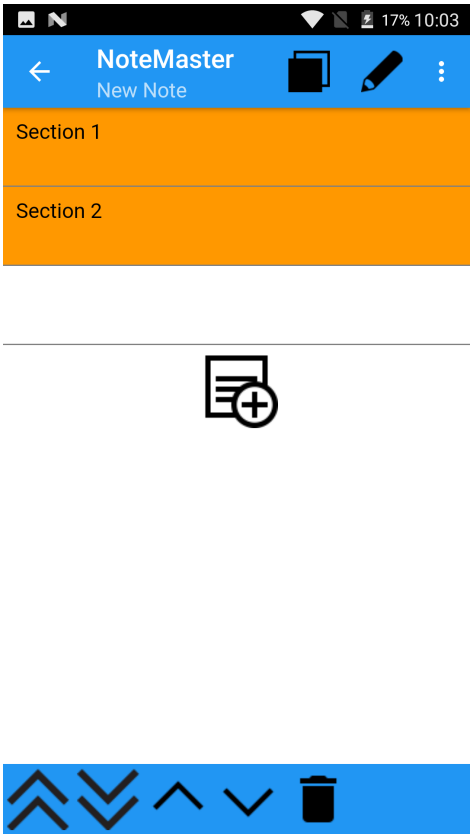
Obrázek 3.7: Akce nad poznámkou



Obrázek 3.8: Vyhledávání poznámek



Obrázek 3.9: Editor textu poznámky



Obrázek 3.10: Editor odstavců poznámky

Návrhy obrazovek nejsou ve formě klasických drátových modelů nebo nákresů, protože Android Studio nabízí WYSIWYG rozhraní pro tvorbu obrazovek a zároveň vytváří zdrojové soubory s layouty, které je možné rovnou využít při vývoji aplikace.

3.5.1 Popis obrazovek

Hlavní menu Hlavní menu je přístupné na hlavní obrazovce aplikace.

Dialog pro vkládání hesla Přehledný dialog sestaven ze čtyř polí po jedné číslici. Důvodem je jednoduchá změna číslice v případě chybného zapsání PINu.

Správa hesel Umožňuje měnit jednotlivá hesla. Obrazovka je spuštěna z hlavního menu, a tedy menu zde není dostupné. Důvodem je zabránění zmatení uživatele a zachování přirozeného větvení aplikace.

Výpis poznámek Výpisem je `Fragment` (viz. sekce 2.2.7) umístěný na hlavní obrazovce aplikace.

Možnosti řazení jsou součástí výpisu poznámek a možnosti jsou, podobně jako hlavní menu, implicitně skryté. Vyvolání nabídky je možné pomocí ikony filtru v horní části obrazovky.

Akce nad poznámkou Rozbalovací menu u každé poznámky nabízející akce k příslušné poznámce.

Vyhledávání je zpřístupněno pomocí ikony lupy v horní části obrazovky. Filtruje poznámky ve výpisu dle zadaného řetězce.

Textový a odstavcový editor Opět jde o samostatnou obrazovku bez hlavního menu. Módy úpravy poznámky se mění kliknutím na ikony v horní části obrazovky. Spolu se změnou módu se také mění paleta nástrojů ve spodní části obrazovky.

3.6 Návrh editoru

Jak již bylo popsáno v sekci 2.7.1, požadované funkcionality na editor textu poznámky nelze docílit použitím nativních komponent OS Android. Proto je nutné vytvořit druhou aplikaci sloužící jako editor. Aplikace bude webová a pro její tvorbu budou využity technologie HTML a Javascript.

3.6.1 Bezpečnost editoru

Potencionální útočník by mohl do editoru napsat škodlivý kód a s jeho pomocí odesílat data mimo aplikaci. Tento problém je známý jako XSS[7] nebo Javascript injection. Součástí tohoto problému je zabránění interpretace uživatelského textu jako HTML elementů.

Nelze však sanitizovat všechny HTML elementy, které poznámka obsahuje, protože by došlo k porušení funkcionalit editoru. Sanitizován bude pouze uživatelský vstup. Konstrukty vytvářené editorem budou brány jako důvěryhodné.

Následující ukázky slouží pouze pro ilustraci útoků

[7] Cross Site Scripting

3.6.1.1 Ukázka XSS

```
// vyvolá dialog pro výběr souboru z úložiště zařízení
<input type="file" />
// tlačítko pro nahrání vybraného souboru
<input type="button" onclick="uploadFile();" />

<script>
function uploadFile()
{
    //logika nahrání vybraného souboru na cizí server
}
</script>
```

Pokud by se tento škodlivý kód nesanitizoval, bylo by možné odeslat soubory ze zařízení na cizí server.

3.6.1.2 Ukázka HTML

```
<input type="button" />
```

Pokud by poznámka obsahovala tento text, tak se nesmí v editoru reprezentovat jako tlačítko, ale musí být sanitizovaná a vždy se zobrazovat jako text.

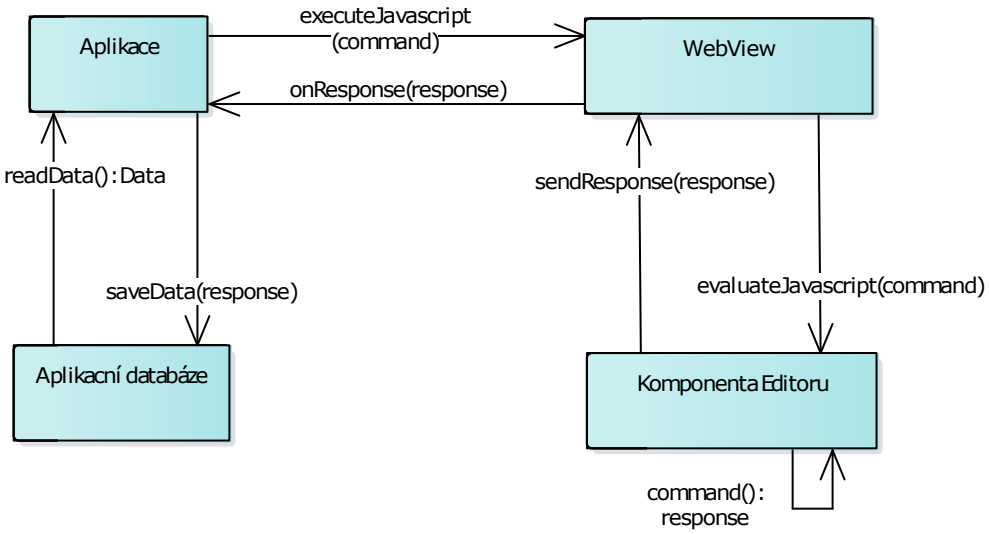
3.6.2 Integrace editoru do aplikace

Jako komunikační můstek mezi aplikací a editorem slouží komponenta `WebView`. Ta umožňuje z aplikace volat přímo Javascriptové funkce a naopak doručovat odpovědi zpět do aplikace.

Zdrojové kódy webové aplikace budou zahrnuty do instalačního balíčku aplikace pro OS Android. Nebude tedy vyžadováno, aby zařízení mělo přístup k internetu pro editaci poznámek.

Aplikace zasílá řetězec s funkcí a jejím parametrem, kterou má editor vykonat. `WebView` dále propaguje řetězec do komponenty editoru, kde je následně funkce vykonána. Odpověď, kterou je návratová hodnota funkce, editor odesílá zpět do aplikace. Data jsou mezi komponentami zasílána v JSON[8] notaci. Aplikace data parsuje a ukládá do databáze. Tento proces je vizualizován na obrázku 3.11.

[8] JavaScript Object Notation



Obrázek 3.11: Komunikační diagram

3.6.3 Komunikace s editorem

Webový editor běží na jiném vlákně než aplikace. Není tedy možné provádět synchronní volání, protože aplikační vlákno by muselo čekat na odpověď a aplikace by neodpovídala na uživatelské akce.

Aplikace tedy bude zasílat požadavky a data do editoru a nebude čekat na návrat volání. Požadavek je dále propagován do editoru, který ho zpracuje. Po zpracování požadavku je výsledek operace, pokud existuje, doručen zpět do aplikace separátním voláním iniciovaným editorem (viz. obrázek 3.12).

3.6.4 Průběh editace poznámky

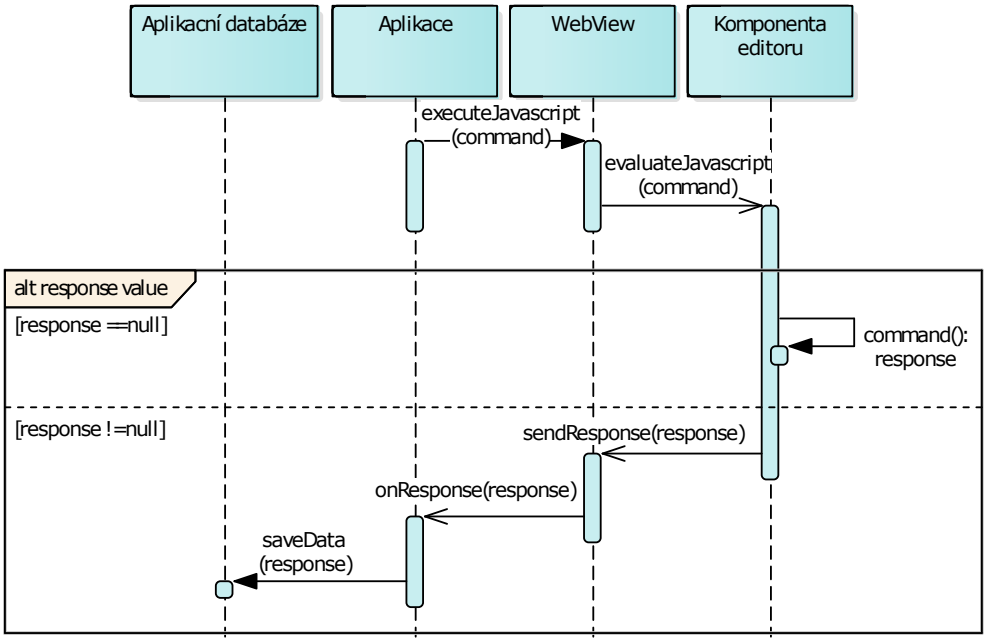
Na obrázku 3.13 je možno vidět průběh otevření poznámky v editoru. Aplikace musí uchovávat data o uživatelských poznámkách v databázi. V případě editace jedné z nich nebo vytvoření nové se její obsah zašle do editoru, který ji zobrazí. Uživatel poté svoji poznámku edituje. Každý jeho úkon se řídí sekvencí, která je popsána v sekci 3.6.3. Po ukončení editace aplikace požádá editor o nový obsah poznámky a uloží ji do databáze.

3.7 Návrh tříd

Diagram na obrázku 3.14 znázorňuje vztahy mezi třídami v aplikaci.

3.7.1 Popis tříd v návrhu

BaseActivity Abstraktní třída, ze které dědí všechny aplikační `Activities`.



Obrázek 3.12: Sekvenční diagram komunikace

BaseFragment Abstraktní třída, ze které dědí všechny aplikační `Fragments`.

MainActivity Hlavní obrazovka aplikace obsahující menu. Obsahuje dynamický layout zobrazující `Fragments`.

EditorActivity Obrazovka pro editaci poznámky. Zobrazuje nástroje a komponentu editoru.

NoteView Zobrazuje webovou stránku s editorem.

PasswordActivity Dovoluje uživateli spravovat hesla na základě právě aktivních hesel.

EntityNote Třída pro ORM popisující strukturu poznámky.

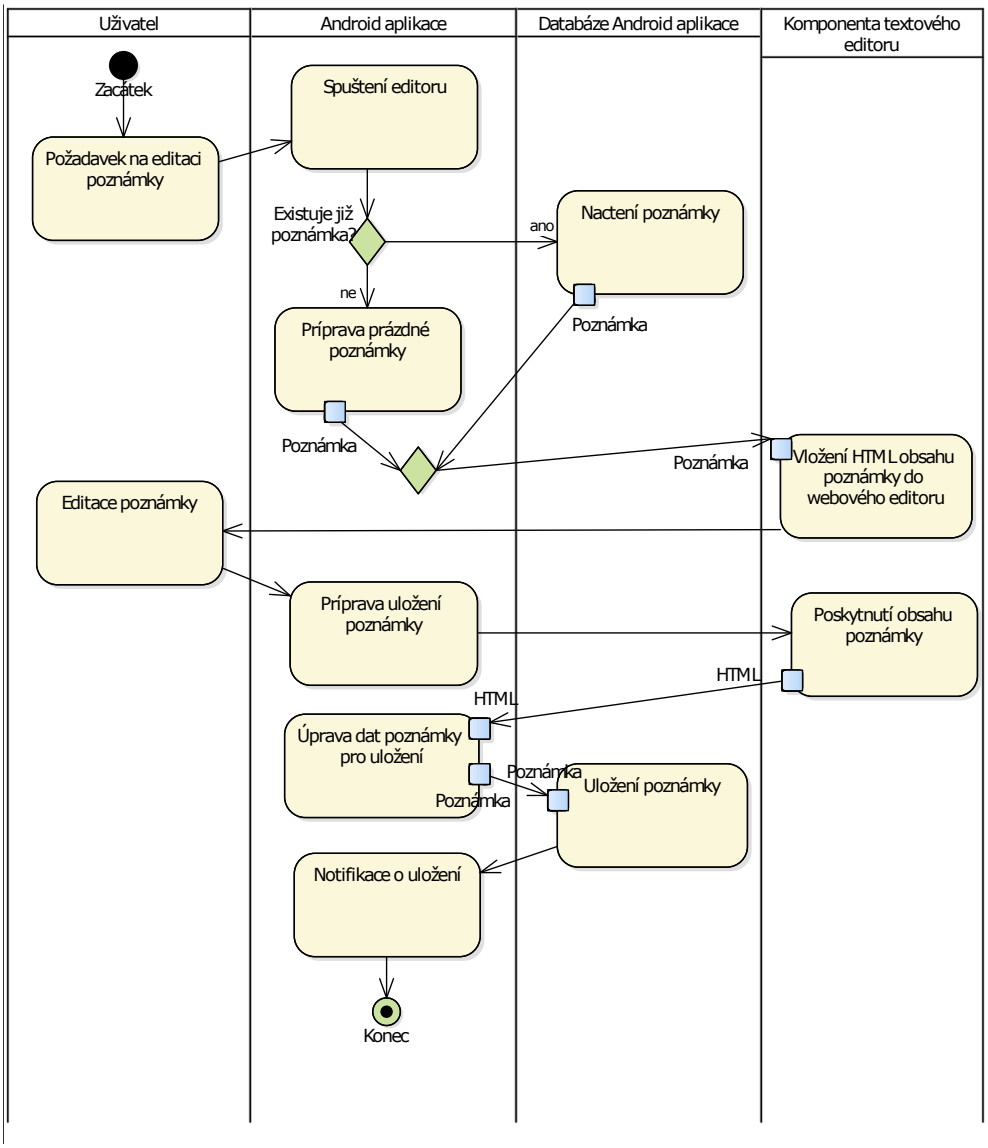
EntityBadge Třída pro ORM popisující strukturu štítku.

KeyboardManager Třída zodpovědná za otevírání a zavírání softwarové klávesnice.

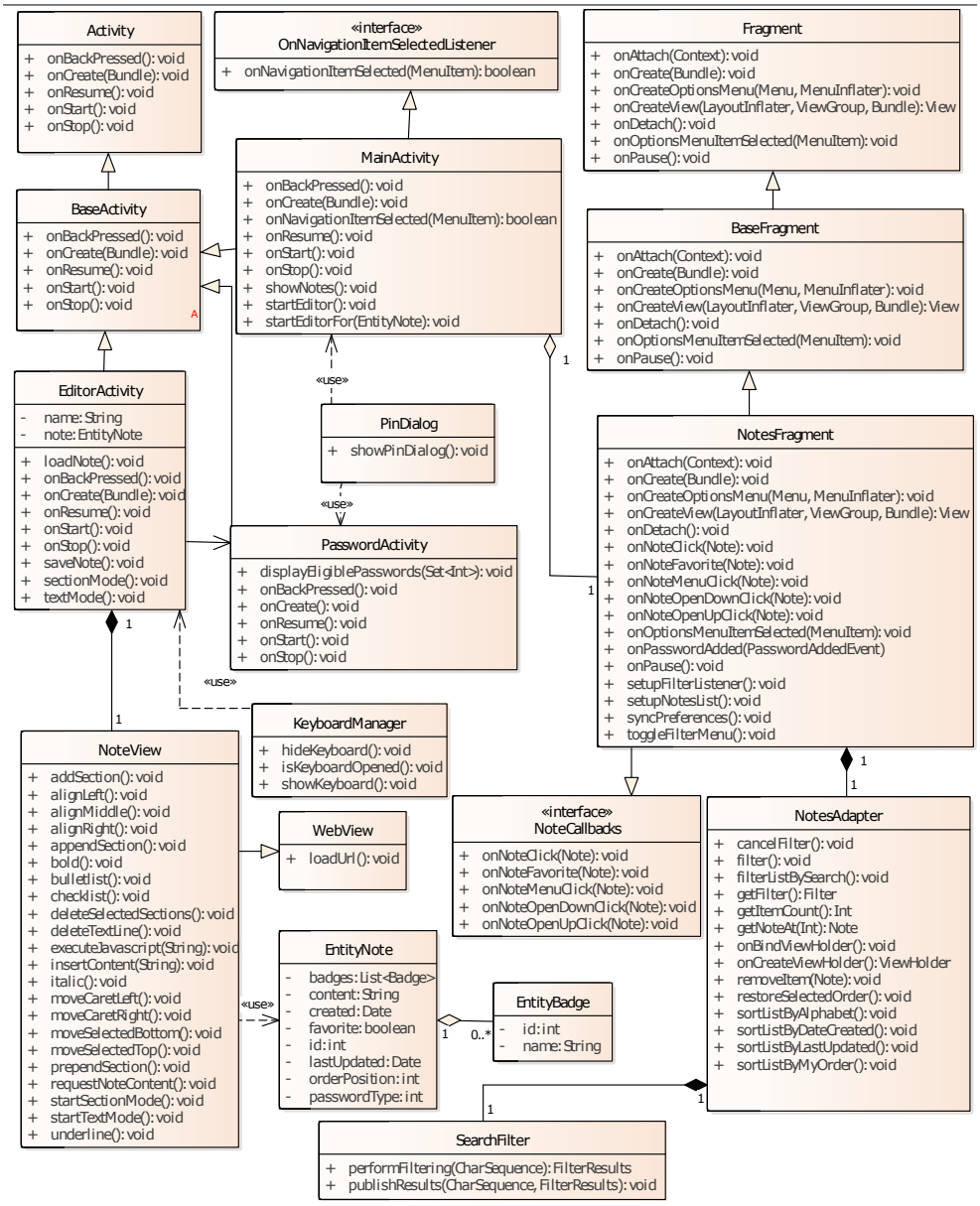
NotesFragment `Fragment` zodpovědný za výpis seznamu poznámek a interakce s ním.

NotesAdapter Adaptér zodpovědný za vykreslení seznamu.

SearchFilter Filtr pro vyhledávání v seznamu poznámek.



Obrázek 3.13: Diagram aktivity editace poznámky



Obrázek 3.14: Diagram tříd

NoteCallbacks Rozhraní pro delegaci uživatelské interakce s poznámkami v seznamu.

PinDialog Dialog pro zadávání hesla.

Kapitola 4

Realizace

4.1 Implementace

4.1.1 Verzování

Pro verzování obou aplikací byl použit verzovací nástroj Git spolu se službou GitLab. Hlavním důvodem verzování bylo předejít ztrátě kódu v případě selhání lokálního pevného disku, a také možnosti vytvoření CI[9] pro automatické spouštění testů.

Pro Android aplikaci a pro webovou komponentu editoru byly zvlášť vytvořeny repozitáře na GitLabu, a to kvůli použití dvou různých vývojových prostředí - Android Studio a JetBrains WebStorm.

4.1.2 Perzistence dat

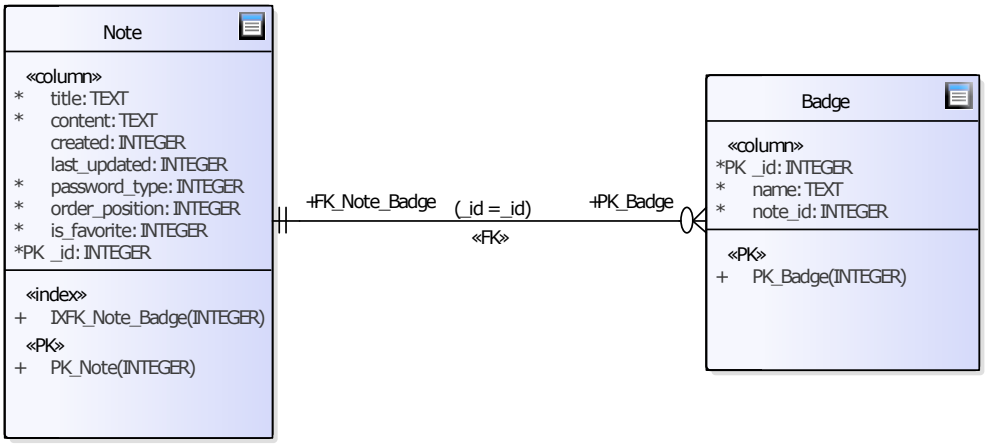
4.1.2.1 Databáze

Pro ORM byla zvolena knihovna GreenDAO kvůli své výkonnostní převaze nad ostatními knihovnami. Další výhodou této knihovny je generování kódů a objektů již v době kompilace, nikoli při běhu aplikace.

Vstupní strukturou pro ORM jsou entitní třídy opatřené anotacemi, které svojí strukturou velmi připomínají doménový model popsaný v sekci 2.9. GreenDAO pak na základě anotací rozeznává vztahy mezi entitamy a generuje tabulky pro relační databázi SQLite spolu s objekty dovolující operace s entitami.

GreenDAO zatím nepodporuje jazyk Kotlin. Vstupní třídy a vygenerovaný kód jsou v jazyce Java.

[9] Continous Integration



Obrázek 4.1: Relační diagram databáze vygenerované knihovnou GreenDAO

4.1.2.2 Ukázka kódu vstupní třídy

```
@Entity
public class Note implements Serializable {

    public static final long serialVersionUID = 42L;
    @NotNull
    public String title;
    @Id
    private Long id;
    @NotNull
    private String content;
    private Date created;
    private Date lastUpdated;
    @ToMany(referencedJoinProperty = "noteId")
    private List<Badge> badges;
    @NotNull
    private int passwordType = PasswordType.NO_PASSWORD;
    @NotNull
    private int orderPosition = 0;
    private boolean isFavorite = false;
}
```

4.1.2.3 Ukázka vygenerovaných tabulek

Relační diagram na obrázku 4.1 znázorňuje vygenerované tabulky relační databáze. Diagram byl vytvořen ze skriptů pro tvorbu tabulek vygenerovanými knihovnou GreenDAO.

4.1.2.4 Ukázka vygenerovaných funkcí

```
//class Note.java
public List<Badge> getBadges() {
    if (badges == null) {
        final DaoSession daoSession = this.daoSession;
        if (daoSession == null) {
           throw new DaoException("...");
        }
        BadgeDao targetDao = daoSession.getBadgeDao();
        List<Badge> badgesNew = targetDao._queryNote_Badges(id);
        synchronized (this) {
           if (badges == null) {
               badges = badgesNew;
            }
        }
    }
    return badges;
}


//class Badge.java
public Note getNote() {
    long __key = this.noteId;
    if (note__resolvedKey == null
        || !note__resolvedKey.equals(__key)) {
        final DaoSession daoSession = this.daoSession;
        if (daoSession == null) {
            throw new DaoException("...");
        }
        NoteDao targetDao = daoSession.getNoteDao();
        Note noteNew = targetDao.load(__key);
        synchronized (this) {
            note = noteNew;
            note__resolvedKey = __key;
        }
    }
    return note;
}
```

V ukázkách vidíme, že vztahy mezi jednotlivými záznamy nejsou načítány při čtení z databáze, ale jsou řešeny tzv. „líným“ načítáním. To znamená, že tyto operace jsou prováděny až v momentě, kdy aplikace tato data doopravdy potřebuje.

4.1.2.5 Hesla

O hesla se stará třída `PasswordManager`, která má na starosti hesla aktivovat, měnit a připravovat k uložení do `Preferences`.

4.1.3 Události

Pro události popsané v sekci 3.4, byla použita knihovna EventBus.

V ukázce kódu můžeme vidět použití knihovny. Komponenta se přihlašuje a odhlašuje v rámci jejího životního cyklu. Druhy událostí, na které komponenta naslouchá, se určují pomocí paramatru metody s příslušnou anotací `@Subscribe`. Lze také specifikovat vlákno, ve kterém má být událost přijata.

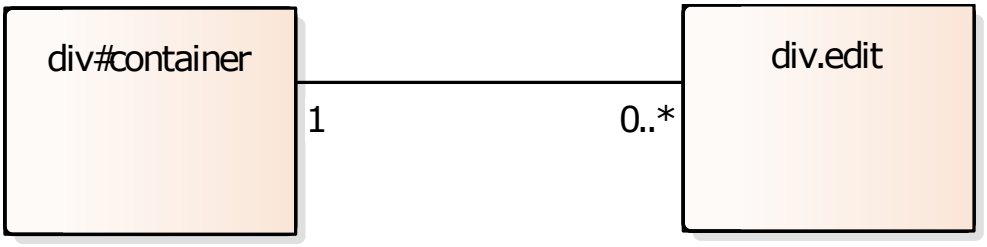
```
override fun onStart() {
    super.onStart()
    EventBus.getDefault().register(this)
}

override fun onStop() {
    super.onStop()
    EventBus.getDefault().unregister(this)
}

@Subscribe(threadMode = ThreadMode.MAIN)
fun onPasswordAdded(event: PasswordAddedEvent) {
    ...
}
```

4.1.4 Editor

Editor je napsaný pomocí jazyků HTML, CSS a Javascript. Struktura poznámky v HTML je znázorněna na obrázku 4.2. Základní dělitel struktury v HTML se nazývá `div`. Prostor, kde vzniká poznámka, je `div` s názvem `container`, který obsahuje další `divs`, které jsou již samotné odstavce poznámky. Editovatelnost je zaručena atributem `contenteditable`.

Javascriptová část obsahuje funkce na úpravu poznámky dle požadavků. Pro formátování se používá funkce `document.execCommand(command)`, která zvládá základní úpravy jako je tučný text, odrážkové menu atd.

Složitější funkce jsou realizovány vkládáním vlastního HTML kódu pomocí již zmíněné zabudované funkce. Konzistentní použití zabudované funkce také umožňuje vrácení se o krok zpět či kupředu.



Obrázek 4.2: Ilustrace HTML struktury

Prohazování a mazání sloupců je řešeno jako prohazování a mazání příslušných elementů v HTML struktuře.

4.2 Testování

Jelikož je platforma Android open-source, je dovoleno všem výrobcům zařízení upravit systém k obrazu svému. Tato skutečnost zapřičiňuje odchylky v chování jednotlivých zařízení.

Odladit aplikaci pro každé zařízení je téměř nemožné, proto OS Android nabízí sadu nástrojů na testování a možnost emulování zařízení.

4.2.1 Statické testy

Android Studio dovoluje automatickou statickou kontrolu kódu pomocí nástroje Android Lint. Lint je nástroj zabudovaný ve vývojovém prostředí Android Studio vytvořený na vyhledání známých chyb, špatného užití, výkonových problémů atd.

Velice užitečná je kontrola tzv. „hard-coded strings“, která vyhledává pevně vložené řetězce v aplikaci. Každý řetězec a hodnota by měla být ve zdrojovém souboru pro správnou lokalizaci aplikace.

Statická kontrola v případě aplikace neodhalila žádné závažné chyby týkající se použitelnosti nebo výkonu aplikace.

Tabulka 4.1: Parametry emulovaných zařízení použitých pro testování aplikace.

verze OS Android	úhlopříčka	rozlišení	RAM
Android 4	3.2”	320x480	512 MB
Android 5	3.4”	240x432	512 MB
Android 6	4.6”	720x1280	1024 MB
Android 7	5.5”	1440x2560	2048 MB
Android 8	7”	2560x1800	2048 MB
Android 8	7”	2560x1800	2048 MB

Tabulka 4.2: Skutečné zařízení použité k testování aplikace.

Zařízení	verze OS Android	úhlopříčka	rozlišení	RAM
HTC One	Android 7	4.7”	1080x1920	2 GB
Xiaomi Redmi Note 4	Android 7	5.5”	1080x1920	3 GB
Lenovo Vibe Z2	Android 5	5.5”	720x1280	2 GB

4.2.2 Unit testy

Předmětem Unit testů jsou třídy pracující s perzistentní vrstvou aplikace. Testované třídy jsou následující:

- PasswordManager,
- Prefs.

4.2.3 Testování programátorem

Základní stavební komponenty OS Android popsané v sekci 2.2.3 a jejich potomci nemohou být instanciovány obvyklým způsobem. O jejich tvorbu se vždy stará systém až při běhu aplikace.

4.2.3.1 Různá zařízení

Pro testování byla využita řada emulovaných a skutečných zařízení pro zajištění správného fungování aplikace na všech podporovaných verzích OS Android a odzkoušení layoutů pro obrazovky s malými úhlopříčkami. Parametry jednotlivých emulovaných zařízení jsou popsány v tabulce 4.1. Skutečná zařízení jsou uvedená v tabulce 4.2.

4.2.4 Automatizované průchody aplikací

OS Android nabízí technologii Espresso pro automatizování testů. Pro nahrání testu stačí aplikaci spustit v režimu nahrávání, a poté naklikat požadovaný

scénář. Espresso zachycuje na jaký objekt bylo při nahrávání kliknuto a vytváří kód, který tento postup simuluje.

Takto vytvořené testy po spuštění jen zopakují scénář. Test je možné doplnit o další podmínky, které musí být v daném kroku splněny, aby byl test úspěšný.

Spolu s využitím všech emulovaných zařízení popsaných v sekci 4.2.3.1 je možné spouštět všechny scénáře na všech zařízeních. To vede k velice efektivnímu testování s okamžitými výsledky.

4.2.4.1 Omezení

Espresso nedokáže zachycovat interakce na webovém editoru, protože obsah není složen z komponent OS Android. Tato část aplikace tedy musí být testována ručně.

4.2.4.2 Scénáře

Scénáře byly tvořeny dle diagramu případů užití, aby co nejlépe simulovaly uživatelskou interakci.

1. **Zamknutí poznámky**

   Předpoklady V aplikaci existuje jedna poznámka a není zamknutá.

   Průběh Aplikace aktivuje hlavní heslo a nastaví uživatelské a hostovské heslo. Poté se poznámka uzamkne hostovským heslem, vynutí se zapomenutí vložených hesel a opět se vloží hostovské heslo.

   Podmínky splnění Poznámka je zamčená hostovským heslem.

2. **Aktivace hesla**

   Předpoklady Přednastavené heslo v aplikaci.

   Průběh Heslo se aktivuje pomocí dialogu vyvolaného z hlavního menu.

   Podmínky splnění Je aktivovaný správný typ hesla.

3. **Vytvoření prázdné poznámky.**

   Průběh Vytvoří se nová poznámka a automaticky se spustí editor. Do poznámky nebude vložen žádný text a editor se ukončí.

   Podmínky splnění Poznámka nebude uložena, protože je prázdná.

4. **Zapomenutí hesel**

   Předpoklady Přednastavené heslo v aplikaci.

Průběh Proběhne aktivace hesla bez zaškrtnutí možnosti zapamatování hesla. Poté se vynutí zapomenutí zadaných hesel. Znovu se zadá stejné heslo a zaškrtne se zapamatování hesla. Opět se vynutí zapomenutí.

Podmínky splnění Po prvním zapomenutí nesmí být heslo aktivní. Po druhém zapomenutí nesmí být heslo uchováno v perzistentní vrstvě aplikace.

5. **Nastavení a aktivace uživatelského a hostovského hesla**

Předpoklady Přednastavené hlavní heslo v aplikaci.

Průběh Aktivuje se hlavní heslo a nastaví se uživatelské a hostovské heslo. Poté se aktivují obě hesla.

Podmínky splnění V aplikaci jsou všechna hesla aktivována.

6. **Vyhledávání**

Předpoklady Existující poznámka v aplikaci.

Průběh Na obrazovce s poznámkami se začne vyhledávat, nejdříve pomocí názvu poznámky, poté se přidá znak, který v názvu poznámky není.

Podmínky splnění Výpis zobrazuje pouze poznámky, jejichž názvy obsahují vyhledávaný termín.

7. **Nastavení štítku**

Předpoklady Existující poznámka, která nemá nastavený štítek.

Průběh Předpřipravené poznámce je nastaven štítek s názvem.

Podmínky splnění Poznámka má nastavený štítek s tím názvem, který byl zadán.

Závěr

Cílem práce bylo vypracovat aplikaci na přání zákazníka. Práce se zabývá analýzou, návrhem, implementací a testováním aplikace.

Na základě výsledku analýzy nemohlo být docíleno požadavků na aplikaci pomocí komponent v OS Android. Bylo tedy nutné vytvořit separátní webovou aplikaci, která je schopná požadované funkcionality docílit. Taktéž bylo nutné navrhnout komunikaci mezi těmito dvěma aplikacemi.

Aplikace umožňuje zapisování poznámek s mnoha nástroji pro formátování textu a správu odstavců. Umožňuje také poznámky chránit heslem, které je ve formě PINu a štítkovat je. Pro snadnou organizaci je možné poznámky řadit dle různých kriterií.

Dle požadavků je aplikace publikována na platformním obchodě Google Play ve fázi beta verze na adrese: `https://play.google.com/store/apps/details?id=cz.cvut.fit.grohjiri.notemaster`

Aplikace je navržena a připravena na rozšiřování své funkcionality do budoucna. Přibudou nové nástroje a díky již implementovanému webovému editoru se nabízí možnost vytvořit nejen mobilní aplikaci, ale i aplikaci webovou.

Literatura

[1] *Evernote | Aplikace na Google Play*. online, [vid. 2018-04-27]. Dostupné z: https://play.google.com/store/apps/details?id=com.evernote

[2] HINE, K.: Evernote. *Journal of the Canadian Health Libraries Association/Journal de l'Association des bibliothèques de la santé du Canada*, 2014, doi:10.5596/c14-001.

[3] TURNER, J.: Using Microsoft OneNote for Collaborative Vocabulary Notebooks in the Academic English Classroom. 2011, [vid. 2018-04-19]. Dostupné z: https://pdfs.semanticscholar.org/beff/ed7a470eb669967ea8590976b7fb5099aebd.pdf

[4] *Poznámky a seznamy Google Keep | Aplikace na Google Play*. online, [vid. 2018-04-27]. Dostupné z: https://play.google.com/store/apps/details?id=com.microsoft.office.onenote

[5] *OneNote | Aplikace na Google Play*. online, [vid. 2018-04-27]. Dostupné z: https://play.google.com/store/apps/details?id=com.google.android.keep

[6] *A Brief History of Android*. online, [vid. 2018-04-20]. Dostupné z: https://visual.ly/community/infographic/technology/brief-history-android

[7] *Mobile Operating System Market Share Worldwide*. online, [vid. 2018-04-20]. Dostupné z: http://gs.statcounter.com/os-market-share/mobile/worldwide

[8] *Distribution dashboard*. online, [vid. 2018-04-20]. Dostupné z: https://developer.android.com/about/dashboards/

[9] *Kotlin and Android*. online, [vid. 2018-04-20]. Dostupné z: https://developer.android.com/kotlin/

[10] *Android SDK tutorial*. online, [vid. 2018-04-20]. Dostupné z: https://www.androidauthority.com/android-sdk-tutorial-beginners-634376/

[11] Application Fundamentals. online, [vid. 2018-04-25]. Dostupné z: https://developer.android.com/guide/components/fundamentals

[12] Activity Lifecycle. online, [vid. 2018-04-14]. Dostupné z: https://developer.android.com/reference/android/app/Activity

[13] Intents and Intent Filters. online, [vid. 2018-04-26]. Dostupné z: https://developer.android.com/guide/components/intents-filters

[14] VOGEL, L.: Android sqlite database and contentprovider-tutorial. ročník 8, 2010, [vid. 2018-04-25]. Dostupné z: http://www.vogella.com/tutorials/AndroidSQLite/article.html

[15] VOGEL, L.: Android development tutorial. 2013, [vid. 2018-04-26]. Dostupné z: https://www.cs.virginia.edu/~cs201/labs/Vogella-Android-Development-Tutorial.pdf

[16] *View*. online, [vid. 2018-04-25]. Dostupné z: https://developer.android.com/reference/android/view/View.html

[17] *ViewGroup*. online, [vid. 2018-04-25]. Dostupné z: https://developer.android.com/reference/android/view/ViewGroup.html

[18] *AdapterView*. online, [vid. 2018-04-25]. Dostupné z: https://developer.android.com/reference/android/widget/AdapterView.html

[19] MORRIS, J.: *Android User Interface Development: Beginner's Guide*. PACKT publishing Ltd, 2011.

[20] *Processes and threads overview*. online, [vid. 2018-04-20]. Dostupné z: https://developer.android.com/guide/components/processes-and-threads

[21] *Android - Inflate (a layout XML to a view UI object)*. online, [vid. 2018-04-20]. Dostupné z: https://gerardnico.com/android/inflate

[22] *Fragments*. online, [vid. 2018-04-20]. Dostupné z: https://developer.android.com/guide/components/fragments

[23] *Kotlin 1.0 Released: Pragmatic Language for JVM and Android*. online, [vid. 2018-04-21]. Dostupné z: https://blog.jetbrains.com/kotlin/2016/02/kotlin-1-0-released-pragmatic-language-for-jvm-and-android/

[24] SAMUEL, S.; BOCUTIU, S.: *Programming Kotlin*. Packt Publishing, 2017, ISBN 9781787126367.

[25] *Kotlin/anko: Pleasant Android application development*. online, [vid. 2018-04-20]. Dostupné z: `https://github.com/Kotlin/anko`

[26] BÍNA, J.: *Android knihovna pro objektově-relační mapování*. Bakalářská práce, České vysoké učení technickév Praze, Fakulta informačních technologií, 2017.

[27] SHUR, G.: *'Coviator' - aplikace pro společné/sdílené plánování zájezdů - OS Android*. Bakalářská práce, České vysoké učení technickév Praze, Fakulta informačních technologií, 2017.

[28] SIAU, K.; LEE, L.: Are use case and class diagrams complementary in requirements analysis? An experimental study on use case and class diagrams in UML. *Requirements engineering*, ročník 9, č. 4, 2004, doi: 10.1007/s00766-004-0203-7.

[29] *EditText*. online, [vid. 2018-04-25]. Dostupné z: `https://developer.android.com/reference/android/widget/EditText`

[30] *Spans, a Powerful Concept.* online, [vid. 2018-04-20]. Dostupné z: `http://flavienlaurent.com/blog/2014/01/31/spans/`

[31] *CheckBox*. online, [vid. 2018-04-25]. Dostupné z: `https://developer.android.com/reference/android/widget/CheckBox`

[32] *WebView*. online, [vid. 2018-04-20]. Dostupné z: `https://developer.android.com/reference/android/webkit/WebView`

[33] *Create a navigation drawer*. online, [vid. 2018-04-25]. Dostupné z: `https://developer.android.com/training/implementing-navigation/nav-drawer`

Příloha **A**

Seznam použitých zkratek

API Application Programming Interface - Aplikační programovací rozhraní

CI Continuous Integration

GUI Graphical User Interface - Grafické uživatelské rozhraní

HTML Hypertext Markup Language

ORM Object Relation Mapping - Objektově relační mapování

OS Operating System - Operační systém

PIN Personal Identification Number - Osobní identifikační číslo

RAM Random Access Memory - Operační paměť zařízení

SDK Source Development Kit - Vývojářský balík

SQL Structured Query Language

UC Use Case - Případ užití

UI User Interface - Uživatelské rozhraní

UML Universal Markup Language

WYSIWYG What You See Is What You Get

XML Extensible Markup Language

XSS Cross Site Scripting

Příloha B

Obsah přiloženého CD

- `install.txt` ..... instalační příručka pro aplikaci
- `notemaster.apk` ..... spustitelný soubor s aplikací
- `src` .....
  - `AndroidApp` ..... zdrojové kódy aplikace pro Android
  - `EditorApp` ..... zdrojové kódy webového editoru
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
  - `uml` ..... zdrojové soubory s UML pro Enterprise Architect
- `paper` .....
  - `thesis.pdf` ..... text práce ve formátu PDF