FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Mobilní Aplikace - Klasifikace
Student:	Bc. Tomáš Havlíček
Vedoucí:	Ing. Zdeněk Balák
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Cílem práce je navrhnout a implementovat mobilní aplikaci, která umožní komunikaci se serverem aplikace "Klasifikace". Aplikace bude podporovat 3 případy užití:
- Učitelům umožní vkládat hodnocení studentů.
- Studentům umožní prohlížet hodnocení.
- Poskytne notifikační službu, která upozorní studenta na změnu v klasifikaci.

1. Seznamte se s dokumentací REST API aplikace Klasifikace.
2. Analyzujte funkcionalitu aplikace z pohledu GDPR a zvolte postupy, které budou v souladu s legislativou.
3. Navrhněte uživatelské rozhraní pro všechny tři případy užití.
4. Zvolte vhodnou implementační platformu. Cílovým prostředím je OS Android. Rozhodněte, zda bude podporován od verze 4 nebo 5 a rozhodnutí zdůvodněte.
5. Vypracujte rešerši notifikačních serverů a zvolte vhodné řešení. Zvolené řešení nechť podporuje platformy Android a iOS.
6. Implementujte všechny tři případy užití.
7. Řešení řádně otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

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

V Praze dne 19. prosince 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Mobilní Aplikace - Klasifikace

Bc. Tomáš Havlíček

Katedra Softwarového inženýrství
Vedoucí práce: Ing. Zdeněk Balák

9. května 2018

Poděkování

Děkuji vedoucímu své práce Ing. Zdeňku Balákovi za cenné rady a připomínky při tvorbě této práce. Dále bych rád poděkovat své rodině za trpělivost a podporu.

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. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 9. května 2018 .....

Odkaz na tuto práci

Havlíček, Tomáš. *Mobilní Aplikace - Klasifikace*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Diplomová práce se zabývá vytvořením mobilní aplikace Klasifikace pro platformu Android, která komunikuje s již existujícím informačním systémem Klasifikace používaným na Fakultě informačních technologií ČVUT. Mobilní aplikace umožní studentům i učitelům zobrazit si své předměty a svou klasifikaci, učitelům dále poskytne možnost klasifikaci zadávat nebo upravovat a studentům pak přijímat notifikace o provedených změnách. Pro fungování notifikací se práce dále popisuje vhodné řešení notifikačního serveru.

V práci jsem nejprve provedl rešerši možných řešení, vytvořil analýzu a návrh aplikace i notifikačního serveru tak, aby byly v souladu s GDPR. Architekturu aplikace, zvolené technologie a uživatelské rozhraní jsem podrobně popsal. Návrh aplikace jsem implementoval a následně otestoval jak automatickými, tak uživatelskými testy. Zdrojové kódy, zkompilované balíčky a protokoly z uživatelského testování lze nalézt v příloze na CD.

Klíčová slova Android, Klasifikace, mobilní aplikace, push notifikace, notifikační server, GDPR

Abstract

This diploma thesis deals with the creation of a mobile application Classification for the Android, which communicates with the already existing web application used at the Faculty of Information Technology of the Czech Technical University. The mobile application allows students and teachers to view their subjects and their classification. Teachers have the possibility to add or change students' scores. The system then sends notifications about new changes. To support notification functionality this thesis describes the most best way to create a notification server.

In this work, I first researched possible solutions, created an analysis and design of the application and of the notification server. The design was made to be in line with GDPR. I have in detail described the application's architecture, selected technologies and user interface. I implemented the application and tested it by both automatic and user tests. Source codes, compiled archives, and user testing protocols can be found in the appendix on the CD.

Keywords Android, Classification, mobile application, push notifications, notification server, GDPR

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Klasifikace v elektronické podobě není v současnosti v českém školství žádnou novinkou, většina vzdělávacích institucích tyto on-line systémy využívá. Přesto se v nich, z pohledu studentů či učitelů, vyskytuje řada nekomfortních drobných nedostatků. Z pohledu dnešní doby asi nejzásadnějším je, že řada systémů není dobře optimalizovaná pro použití v mobilním zařízení. Dalším mínusem je samotný přístup do těchto systémů přes webový prohlížeč v mobilním zařízení, kdy při přihlašování přes externí autorizační systém musí uživatel k úspěšnému vstupu na webovou stránku klasifikace provést řadu dílčích kroků, což bývá zbytečně zdlouhavé. Nelze opomenout i situaci, kdy student napjatě očekává záznam nové známky, a chce-li zjistit, zda je jeho hodnocení již zadáno, musí často aktualizovat webové rozhraní klasifikačního systému, aby přehled průběžně zadaných známek zkontroloval.

Systém zadávání klasifikace v rámci FIT ČVUT je v současné době nejednotný, využívají se systémy Edux nebo Moodle, které však neodpovídají dnešním požadavkům, nepodporují notifikování uživatelů a uchovávají data v podobě, která je špatně strukturovaná pro strojové zpracování. Nyní se plánuje přesun veškeré klasifikace do jednoho informačního systému, kterým je webová aplikace Klasifikace.

Cílem mé diplomové práce je navrhnout a vytvořit mobilní aplikaci Klasifikace pro zařízení se systémem Android, která bude komunikovat s již existující webovou aplikací a která bude řešit výše zmíněné nedostatky. Mobilní aplikace vzniká proto, aby uživatelé, pro které je v dnešní době standardem mít při sobě stále mobilní telefon, měli ke svým datům rychlý a jednodušší přístup a nabídne jim intuitivní a funkční rozhraní pro přístup ke klasifikaci.

V mobilní aplikaci si studenti budou kontrolovat své hodnocení z předmětů, které aktuálně studují nebo studovali v předchozích ročnících. Aplikace bude podporovat push notifikace, které upozorní studenta na nově zapsané hodnocení a na další události. S touto funkcí se pojí další z cílů práce, kterým je analyzovat a navrhnout nejlepší řešení pro notifikační službu umožňující odesílání notifikací ze systému Klasifikace do mobilních zařízení se systémem

Android a iOS.

Učitelé díky této aplikaci ocení, že budou mít možnost upravovat hodnocení svým studentům u předmětů, které vyučují, a to i ve chvíli, kdy nemají aktuálně přístup k počítači, ale mají při ruce pouze mobilní telefon. Odpadá jim zapisování údajů do svých poznámek a jejich následné dopisování v počítači, což může vést ke zdržení, nebo dokonce ztrátě dané informace.

Práce má několik částí, úvodní kapitoly jsou věnované standardům, technologiím a možnostem řešení mobilní aplikace, které se využívají v dalších kapitolách nebo jsou důležité pro samotnou implementaci. Dále je provedena analýza aplikace a notifikačního serveru, porovnání s podobnými programy a vysvětlena směrnice GDPR v kontextu vznikajícího systému. Na základě analýzy je poté proveden návrh architektury aplikace, komunikace s potřebnými systémy, uživatelského prostředí, notifikačního serveru a jeho rozhraní. Závěrečná část je věnována důležitým bodům z realizace aplikace a notifikačního serveru včetně uživatelského a automatického testování.

Kapitola 1

Technologie a možnosti řešení

V této kapitole jsem stručně popsal technologie a standardy, které jsou v práci využívané, dále jsem vyhodnotil možnosti řešení, shrnul jejich výhody a nevýhody a zvolil, jakou z možností pro práci využiji.

1.1 REST

REST (REpresentational State Transfer) je styl, jak navrhovat slabě provázané aplikace (loosely coupled), hlavně jde o webově orientované aplikace, které závisí na nějakém zdroji spíše než na zasílání zpráv. REST není konkrétní standard nebo protokol, je to spíše sada principů či způsob myšlení. [1]

REST je vytvořen jako abstrakce protokolu HTTP a přejímá HTTP principy, ale abstrahuje jeho technické detaily. V současné době je to hlavní (vedoucí) programovací model pro tvorbu Web API [2].

1.1.1 Hlavní principy RESTu

REST pracuje nad protokolem HTTP a je nezávislý na doméně. Pracuje se zdroji pomocí standardních HTTP metod a dodržuje jejich sémantiku. Konkrétní metody a jejich použití:

- GET - získání dat a informací o zdroji
- POST - vytvoření nebo úprava zdroje
- PUT - idempotentní úprava nebo vytvoření zdroje (zdroj je nahrazen přijatým zdrojem)
- PATCH - úprava zdroje editací
- DELETE - smazání zdroje

REST použitý jako architektonický styl definuje dle [3] následující omezení a principy:

- Klient/server - komunikace vždy využívá síťovou architekturu klient / server. Můžeme jasně definovat, který systém reprezentuje server a který systém je klient. Server odpovídá na požadavky klienta.
- Bezstavovost - komunikace neví o předešlém stavu komunikace. Veškeré informace potřebné pro zpracování požadavku v něm musí být obsažené nebo se musí jednat o informace zdroje (princip HATEOAS).
- Možnost uložení do mezipaměti (cache) - lze využít mezipamět pro idempotetní operace.
- Vrstvený systém - systém je tvořen vrstvením různých podsystémů.
- Jednotné rozhraní - konečná množina operací nad zdroji. Je nezávislá na doméně, pouze definuje, jak pracovat se zdroji. Používá výše zmíněné HTTP metody.

Systém, který je v souladu s principy RESTu se nazývá RESTful.

1.2 OAuth 2.0

OAuth 2.0 je standardní průmyslový protokol používaný pro autorizaci. Staví na základech původního OAuth protokolu z roku 2006. Druhá verze protokolu se soustředí na jednoduchost pro vývojáře klientských systémů a poskytuje specifické způsoby autorizace pro webové, desktopové a mobilní aplikace či různá další chytrá zařízení. [4]

Dle způsobu autorizace definuje OAuth 2.0 několik typů udělení přístupu tzv. Grand type. Každý z nich je určen pro specifické použití. Uvádím popis jednotlivých typů udělení přístupu spolu s jejich použitím:

Authorization Code tento typ se používá pro výměnu autorizačního kódu za přístupový token. Poté, co se uživatel přihlásí, je vrácen do aplikace pomocí redirect URL s autorizačním kódem, který aplikace použije vytvoření požadavku na získání přístupového tokenu. [5]

Implicit jedná se o zjednodušený způsob autorizace, umožňující přímo získání přístupového tokenu bez dalšího kroku s autorizačním kódem. V současné době se obecně nedoporučuje tento způsob autorizace používat. Od doby, kdy byl protokol OAuth 2.0 specifikován, se změnila průmyslová praxe a je doporučováno použití autorizačního kódu bez tajného klíče klienta nebo tzv. rozšíření PKCE. [6]

Password neboli heslo se používá pro samotné přihlášení uživatele v primárním systému. Dochází k přímé výměně uživatelského jména a hesla za přístupový token. Teto způsob by neměl být používán aplikacemi třetích stran. [7]

Client Credentials se používá pro získání přístupového tokenu bez kontextu nějakého uživatele. Typicky se používá pro získání informací o samotné klientské aplikaci místo informací o uživateli. [8]

Device Code používá se v zařízeních bez webového prohlížeče nebo v zařízeních bez možnosti jednoduše vkládat text. [9]

Refresh Token tento typ se používá pro aktualizaci přístupového tokenu, který vypršel. To klientům umožní pokračovat s novým validním přístupovým tokenem bez nutnosti uživatelského zásahu v podobě nového přihlášení. [10]

1.3 Možnosti řešení mobilní aplikace

Existuje několik způsobů, jak vytvářet aplikace pro mobilní zařízení. Aplikace může být webová stránka optimalizovaná pro zobrazení ve webovém prohlížeči mobilního zařízení, hybridní aplikace nebo nativní aplikace.

Jednotlivé přístupy se liší podle toho, jakou mají kontrolu nad zařízením. Dle náročnosti různých projektů je třeba se zamyslet, který přístup zvolit.

1.3.1 Optimalizovaná webová stránka

Mobilní verze webu nebo optimalizace webu pro mobilní telefony je vhodný přístup pro malé aplikace, kde se nepředpokládá častá nebo dlouhodobá interakce uživatele, případně jde o aplikace s jednou či několika málo funkcemi, které nejsou výpočetně náročné. Dále je vhodné použít optimalizovanou webovou stránku pro zobrazení statické prezentace. Typickým příkladem, pro který se tento přístup hodí, je prezentace restaurace s rezervačním systémem.

1.3.2 Hybridní aplikace

Vytvoření hybridní aplikace umožňuje vývojáři snadněji pomocí jedné implementace cílit na platformy Android i iOS. Existuje řada frameworků, které toto umožňují. Většinou se pro vývoj používá kombinace HTML, CSS a Javascriptu, tedy technologií, které jsou primárně určeny pro vytváření webů. Framework poté vygeneruje aplikační balíčky pro dané platformy. Výhodou proti webové stránce pro mobily je to, že framework umožňuje lepší přístup k systému, například možnost persistentně uložit a číst data vytvořená aplikací. Nicméně vždy záleží na tom, jak dobře umí použitý framework pracovat se systémovými funkcemi cílových platforem. Funkce jako GPS, push

notifikace nebo fotoaparát jsou zpracovány v jednotlivých frameworcích různě a může být obtížné najít framework, který zvládne pracovat s celou množinou požadovaných funkcí pro danou aplikaci. Další nevýhodou hybridních aplikací je, že aplikace vypadají na všech platformách stejně, což může odporovat standardnímu chování a uživatelskému rozhraní dané platformy. Hybridní aplikace jsou také náročnější na výkon a mohou být pomalejší než aplikace nativní, ale to v současné době většinou není velký problém.

1.3.3 Nativní aplikace

Nativní aplikace umožňují plný přístup a kontrolu nad systémem. Takový způsob řešení je vhodný pro aplikace, které využívají různé systémové prostředky a funkce, potřebují vysoký výkon nebo pro ty, které nechtějí dělat kompromisy ve vzhledu a chování použitých komponent.

Z pohledu platforem Android i iOS existuje obsáhlá dokumentace a podpora pro vývoj nativních aplikací. Systém samotný obsahuje standardní komponenty se standardním chováním pro danou platformu. Využití těchto komponent umožňuje vývojáři jednoduše přistupovat k systémovým funkcím nebo implementovat uživatelské rozhraní dle standardů pro danou platformu.

Mobilní aplikaci Klasifikace budu vyvíjet jako nativní aplikaci.

1.4 Verze Androidu

Android je operační systém založený na linuxovém jádře. Byl vyvíjen od roku 2003 společností Android, kterou v roce 2005 koupil Google. V roce 2007 byla zveřejněna první beta verze systému a v roce 2008 začal prodej prvního telefonu s tímto operačním systémem. [11]

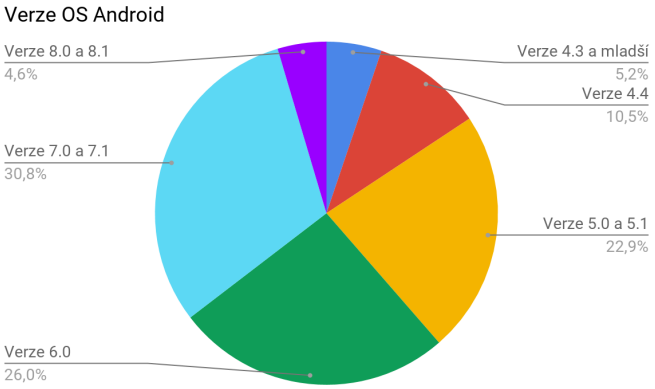
V průběhu vývoje byly doplňovány nové funkce a možnosti, některé verze obsahovaly důležitější změny, jiné nebyly tak zásadní. V současnosti je nejnovější vydaná verze OS Android již 8.1, ale starší verze jsou stále hodně používané. Zastoupení jednotlivých verzí je vidět v tabulce 1.1. To je dáno tím, že výrobci telefonů většinou ukončí podporu svých modelů po jednom až dvou letech od ukončení prodeje daného modelu. Potom už nevydávají nové aktualizace systému. Zároveň si většina výrobců systém Android částečně upravuje nebo rozšiřuje pro použití ve vlastních zařízeních, a proto se telefony s nejnovějším systémem objevují na trhu se zpožděním oproti času vydání nové verze operačního systému.

1.4.1 Zvolená verze

Aplikace Klasifikace bude podporovat zařízení s OS Android od verze 4.4 Kitkat. Tímto krokem se docílí možnosti využití aplikace pro širokou škálu mobilních telefonů, jak je patrné z grafu 1.1, bude to více než 94% zařízení. Navíc, jelikož se systém stále vyvíjí, bude toto číslo časem ještě stoupat.

Tabulka 1.1: Aktuální zastoupení jednotlivých verzí OS Android [12]

Verze	Rok	Jméno verze	API	Zastoupení
2.3.3 - 2.3.7	2010 - 2011	Gingerbread	10	0,3%
4.0.3 - 4.0.4	2011 - 2012	Ice Cream Sandwich	15	0,4%
4.1.x	2012	Jelly Bean	16	1,7%
4.2.x	2012		17	2,2%
4.3	2013		18	0,6%
4.4	2013-2014	KitKat	19	10,5%
5.0	2014	Lolipop	21	4,9%
5.1	2015		22	18%
6.0	2015	Marshmallow	23	26%
7.0	2016	Nougat	24	23%
7.1	2016		25	7,8%
8.0	2017	Oreo	26	4,1%
8.1	2017		27	0,5%



Obrázek 1.1: Poměr jednotlivých verzí OS Android

1.5 Specifika vývoje aplikací v systému Android

Jak již bylo zmíněno, operační systém Android je založen na linuxovém jádře, ale na rozdíl od aplikací pro desktopové operační systémy se aplikace pro Android v různých ohledech liší. V kódu aplikace pro Android například není žádný exaktní vstupní bod, operační systém při spuštění aplikace sám vytvoří její instanci a hlavní aktivitu. Vývojář má možnost vstupovat do chodu jednotlivých komponent pomocí implementace metod pro jednotlivé kroky životního cyklu nebo metod pro vyřízení různých událostí. Komponenty, které jsou pro vývojáře nejdůležitější a nejčastěji se s nimi pracuje, jsou součástí tzv. Android Frameworku.

Dále je třeba se při návrhu a vývoji držet standardů a principů pro tvorbu uživatelského rozhraní. V této kapitole se věnuji klíčovým komponentám, popisuji, k čemu se používají a dále zmiňuji specifika, které je dobré při vývoji aplikací pro operační systém Android znát.

1.5.1 Aktivita

Aktivita je základním stavebním prvkem aplikace, každá aplikace musí mít alespoň jednu aktivitu. Jedná se o třídu, která má definovaný životní cyklus, do jehož kroků má vývojář možnost vstupovat, spouštět vlastní logiku a ovládat aplikaci. Aktivita obsahuje různé komponenty, které se zobrazí uživateli a tvoří uživatelské rozhraní.

1.5.2 Fragment

Fragment se vkládá do aktivity a používá se jako hlavní komponenta uživatelského prostředí. V jedné aktivitě může být vloženo několik fragmentů. Díky tomuto přístupu je možné fragmenty používat vícekrát a jednoduše vytvářet různá rozhraní pro různé účely či velikosti obrazovek. Životní cyklus fragmentu je spjatý s životním cyklem aktivity, která ho obsahuje.

1.5.3 Service

Service neboli služba je komponenta, která se stará o realizaci déle trvající operace na pozadí, která nepotřebuje vstup od uživatele. Typicky se služba využívá pro přehrávání hudby, operace se soubory nebo pro obsluhu síťových požadavků. Spuštěná služba může běžet i v případě, kdy se uživatel nenachází v aplikaci a věnuje se na zařízení něčemu jinému. Pokud má systém málo prostředků, může službu na pozadí omezit či ukončit.

1.5.4 Resources

Každý projekt aplikace pro systém Android má speciální adresář res, do kterého lze vkládat různé resources (soubory a hodnoty), které jsou poté v kódu snadněji přístupné. Mohou to být ikony, textové řetězce pro jednotlivé podporované jazyky, rozložení uživatelského rozhraní, grafika a další hodnoty.

Různé typy položek se v adresáři res nacházejí v různých podadresářích např. values, layout, drawable a další. Kromě daného názvu mohou mít podadresáře ještě příponu, specifikující pro jaké parametry cílového zařízení se mají použít které hodnoty. Příponou lze určit mnoho vlastností, od orientace obrazovky (na výšku nebo na šířku), přes různé velikosti displeje, až po jazykové nastavení systému. Při přístupu k hodnotám není nutné znát konkrétní přípony, vždy se automaticky použije správná hodnota dle konkrétního nastavení zařízení.

1.5.5 Tvorba uživatelského rozhraní v systému Android

Uživatelé operačního systému Andoroid očekávají, že se budou aplikace chovat a vypadat v souladu s platformou. Nejen, že je nutné dodržovat materiální principy pro visuální a navigační vzory, ale je také potřeba se držet pokynů pro kvalitní kompatibilitu, výkon, zabezpečení a dalších. [13]

Visuální a navigační vzory popisuje designový styl Material Design publikovaný Googlem v roce 2014. Jeho cílem je sjednotit vzhled aplikací na různých platformách a zařízeních jako jsou mobilní telefony, tablety, webové aplikace, ale i chytré hodinky nebo televize. Kromě Material designu existují ještě kvalitativní principy, zabývající se vlastnostmi, které jsou specifické pro platformu Android. Zaměřují se spíše na vnitřní fungování a zabezpečení aplikací, dále rozvádí některá pravidla Material designu a přidávají pokyny pro prvky, které Material design nepokrývá.

1.6 Reaktivní programování

Reaktivní programování je paradigma, které se zaměřuje na šíření změn. Pokud program propaguje změny svých dat všem zainteresovaným stranám (uživatelům, jiným komponentám, jiným programům) lze ho nazvat reaktivní. [14] [Learning Reactive Programming with Java 8, Nickolay Tsvetinov] Reaktivní programování je řízené událostmi. Pro implementaci tohoto přístupu se využívá návrhového vzoru „pozorovatel“ (anglicky observer), kde na jedné straně vystupuje objekt pozorovaný (observable či subject) a na druhé objekty pozorující (observer), které na pozorovaném objektu závisí. Pro vytvoření či naopak zrušení závislosti obsahuje pozorovaný veřejné metody, pomocí kterých se k němu pozorovatel registruje, respektive od něj odregistruje. V případě, že se změní stav pozorovaného objektu, informuje o tomto stavu všechny pozorující objekty, které následně vykonají na základě přijaté zprávy požadovanou činnost. [15]

1.6.1 RxJava

Tento způsob programování se v poslední době dostává do popředí a vznikají frameworky pro zjednodušení implementace reaktivních programů. Jedním z nich je i knihovna ReactiveX, která se na Androidu používá ve své podobě jako RxJava.

RxJava obsahuje několik základních typů pozorovaných, kdy nejdůležitější je Observable. Každý z těchto typů obsahuje široké množství operátorů, které umožňují pozorované objekty vytvářet, transformovat, filtrovat či jinak měnit. Dále lze nastavovat výpočetní vlákno, na kterém má konkrétní operace běžet. Kombinováním těchto operátorů lze lehce a čistě dosáhnout komplexního chování, které je dobře čitelné. Mnoho často používaných knihoven je připravených

na využívání RxJavy, což zvyšuje její oblibu a ještě více usnadňuje vývoj aplikací.

1.7 Notifikace v mobilních zařízeních

Aplikace na mobilní telefony často využívají informace z nějakého internetového zdroje. V některých případech je vhodné upozornit uživatele na změnu či vznik takového zdroje. Za tímto účelem je v platformách pro mobilní zařízení možnost zobrazovat uživateli notifikace.

Tato kapitola popisuje, jaké existují varianty pro tvorbu aplikací využívajících notifikace a specifika těchto variant na mobilních zařízeních. Dále zde bude vysvětlen standardní způsob posílání notifikací do mobilních zařízeních se systémem iOS a Android.

1.7.1 Polling, long polling a PUSH notifikace

Zjišťování změn nějakého sdíleného zdroje může být realizováno různými způsoby, které mají svá specifika, výhody a nevýhody. Některé jsou jednoduché, ale více zatěžují síť, jiné vyžadují využití systému třetích stran, či jdou použít jen mezi zařízeními s veřejnou adresou. Nejčastěji se používají tyto metody:

- **Polling**

  Tato metoda se periodicky dotazuje na změnu či vznik zdroje. Nastavená perioda určuje, jaké nejvyšší zpoždění může nastat mezi změnou zdroje a notifikováním uživatele. Jedná se o nejjednodušší způsob pro automatické vyčítání změn, který se dá použít pouze v základních případech, kdy nevadí zpoždění. Nicméně v mobilních zařízeních není tato metoda vhodná, protože automatické, periodické aktualizování na pozadí je značně náročné na spotřebu baterie.

- **Long polling**

  Tento způsob získání aktualizovaných informací se podobá předchozí metodě, avšak server, na který se aplikace dotazuje neodpoví ihned, ale spojení se udržuje. Pokud v této době vznikne událost, která má vyvolat notifikaci, vrátí server tuto informaci jako odpověď na čekající dotaz. Jakmile je vrácena odpověď, nebo je spojení z jiných důvodů ukončeno, vytvoří aplikace spojení nové. Tento postup zajišťuje eliminaci prodlevy mezi vznikem nové informace a jejím doručením do aplikace.

  Tento princip je optimalizován na nižší spotřebu baterie a je použit ve standardní notifikační službě systému Android a iOS, kterými se podrobněji zabývám v následujících podkapitolách.

- **PUSH notifikace**

Poslední způsob doručení nových informací ze serveru je použití takzvaných PUSH notifikací, kdy server sám posílá upozornění na nové informace. Tento princip má omezení v tom, že ne vždy je možné, aby server navázal spojení s klientem. Spojení často může inicializovat pouze klient. Různé implementace PUSH notifikací často obcházejí toto omezení vytvořením spojení ze strany klienta, které je poté udržováno aktivní. Je to obdobný princip jako u předchozího long pollingu. Nicméně, pokud je možné přímé připojení ke klientu ze strany serveru, není nutné udržovat žádné aktivní spojení a server notifikuje klienta v momentě, kdy vzniknou nové informace.

Mobilní zařízení jsou většinou skryta v neveřejných sítích s neveřejnými IP adresami, čímž není možné PUSH notifikace čistě ze strany serveru zasílat.

1.7.2 Notifikace v systému iOS

Funkci notifikací ze vzdáleného zdroje v systému iOS, ale i v dalších operačních systémech společnosti Apple zajišťuje služba zvaná Apple Push Notification service (APNs). Při prvním spuštění aplikace vytvoří operační systém bezpečné spojení mezi zařízením a APNs, přes které si aplikace povolí a nastaví přijímání notifikací. Aplikace dostane z APNs svůj token, který následně sdělí zdroji dat. Zdroj informací poté token využívá pro komunikaci s APNs, přes kterou posílá notifikace do aplikace v zařízení. [16] Graficky je proces znázorněn na obrázku 1.2.

1.7.3 Notifikace v systému Android

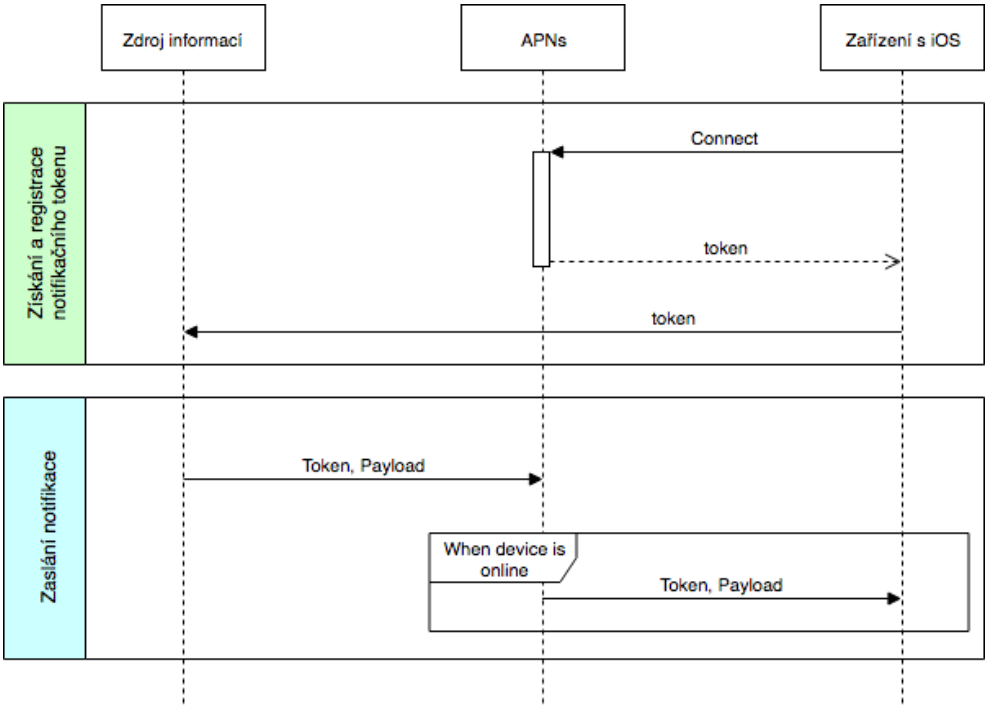
Standardní notifikační službou v systému Android je Firebase Cloud Messaging (FCM), který rozlišuje tři hlavní komponenty. Jsou jimi aplikační server, FCM server a klientské aplikace. Aplikační server posílá zprávy FCM serveru, který tyto zprávy přeposílá klientským aplikacím.

Pro komunikaci mezi aplikačním serverem a FCM serverem je dle [17] možné využít vývojářskou sadu poskytovanou pro programovací jazyky Java, Python, Go a pro systém Node.js nebo jeden ze tří dostupných protokolů:

FCM HTTP v1 API - Nejflexibilnější a nejaktuálnější protokol pro posílání zpráv, doporučuje se využít právě tento protokol. Je specifickým způsobem autorizace aplikačního serveru, který využívá protokol OAuth.

Legacy HTTP protokol - Původní a jednodušší HTTP protokol, s autorizací pomocí klíče posílaného v hlavičce requestů.

Legacy XMPP protokol - Speciální protokol pro posílání zpráv, používá se v případě, kdy je potřeba posílat zprávy nejen z aplikačního serveru



Obrázek 1.2: Graf komunikace při registrování v Apple Push Notification Service [16]

ale i z klientských aplikací. Pro autorizaci se používá standard SASL PLAIN.

Pro zasílání zpráv z aplikačního serveru do klientského zařízení je tedy nejprve potřeba nastavit autorizaci aplikačního serveru, dále je nutné si v klientské aplikaci vyžádat notifikační token, který se odešle aplikačnímu serveru. Aplikační server pomocí tohoto tokenu a svých autorizačních údajů posílá zprávy FCM serveru. V momentě, kdy je cílové zařízení dostupné, odešle FCM server notifikaci. Princip je podobný jako na obrázku v předchozí kapitole, proto nebudu speciální diagram uvádět.

1.8 Notifikační server

Pro oddělení logiky zasílání notifikací přes služby různých platforem a logiky aplikace je vhodné použít notifikační server. Webová aplikace, která obsahuje potřebná data nezasílá notifikace uživatelům přímo, ale notifikačnímu serveru. Ten se stará o jejich rozeslání uživatelům a dále o uchování notifikačních tokenů a jejich mapování na uživatele. Jeden uživatel může dostávat notifikace do více zařízení.

Aplikace v mobilním zařízení obdrží tuto notifikaci a vyčte si podrobné informace z webové aplikace. Tím je zaručeno, že notifikační server je pouze prostředník a nepřechovává žádná uživatelská osobní data.

Dodatečně může webová aplikace informovat notifikační server o tom, že je notifikace označena za přečtenou. Notifikační server následně odešle notifikaci pro schování již zobrazené notifikace.

Kapitola 2

Analýza

V této kapitole je uvedena analýza celého projektu, tedy jak aplikace, tak notifikačního serveru. V kontextu vytvářené mobilní aplikace provádím její porovnání s podobnými existující řešeními, vytvářím seznam funkčních a nefunkčních požadavků a případů užití. Dále se zabývám možnými řešeními pro notifikační server a volbou vhodné varianty. Ke konci kapitoly se věnuji novému nařízení o ochraně osobních údajů GDPR a jeho dopadům na aplikaci a notifikační server.

2.1 Stávající a konkurenční řešení

Jelikož v současné době neexistuje žádná mobilní aplikace používající jako zdroj systém Klasifikace, mohu porovnávat pouze aplikace používající jiné systémy, případně mohu zhodnotit stav toho, jak aktuálně vypadá a funguje webové rozhraní Klasifikace na mobilních zařízeních.

Mezi jednotlivými porovnávanými produkty jsou značné rozdíly, které jsou dané nabídkou možností a funkcionalit, které obsahují.

2.1.1 Školní známky, rozvrh hodin

Aplikace umožňující udržovat lokálně známky, události a rozvrh hodin, v placené verzi i pro více semestrů najednou. Aplikace nekomunikuje s žádným systémem, a proto je nutné všechna data zadávat ručně. Na druhou stranu lze vytvořit zálohu a obnovit ji, takže je uživatelům umožněn celkem jednoduchý způsob přesunu dat na jiné zařízení. Při použití aplikace na více zařízeních, nedochází k synchronizaci dat, tedy změna provedená na jednom z nich se nepromítá do ostatních a je třeba je zadávat úpravy manuálně.

Uživatelské rozhraní je neintuitivní a neodpovídá standardům pro systém Android. Možnost smazání nebo editace známek, událostí nebo položek v rozvrhu není na první pohled dostupná a těžko se hledá. Nicméně pokud

si uživatel na fungování zvykne, bude spokojen s tím, že může mít všechny informace o studiu na jednom místě.

V porovnání s mou aplikací Klasifikace obsahuje aplikace Školní známky, rozvrh hodin navíc možnost vytvořit události a rozvrh hodin. Ze zapsaných hodnocení umí vypočítat součet nebo průměr a přehledně tyto informace zobrazit. Na druhou stranu jsou data zadávána manuálně a aplikaci není možné používat pro vyučující k zadávání známek studentům.

2.1.2 School Marks

Jednoduchá aplikace pro manuální zadávání známek v N úrovních, znázorněných jako složky. V každé složce lze vytvořit podsložku, známku nebo nadpis. V jedné složce je také možnost vytvořit sérii známek pro urychlení zadávání. Pro každou složku se zobrazuje průměr všech známek, které složka přímo obsahuje nebo které jsou obsažené v podsložkách. V nastavení je možnost povolit zadávání váhy známky, která se dá snadno zapnout a vypnout. Nevýhoda je, že aplikace neumí přepnout průměr na součet, takže pokud je směrodatný součet hodnocení a nikoliv průměr, ztrácí své výhody.

Díky systému vnořování známek mohou aplikaci využívat jak studenti pro zapisování vlastní klasifikace, tak i učitelé, kteří ji mohou využít jako jednoduchý zápisník pro udělenou klasifikaci. Oproti aplikaci Klasifikace jsou vkládaná data pouze lokální, nejsou sdílená a jsou tedy dostupná pouze uživateli, který si je zapisuje.

2.1.3 Grade Tracker Pro

Grade Tracker Pro lze vyhodnotit jako propracovanou aplikaci s širokou nabídkou funkcí především pro studenty. Je možné zadávat známky, skupiny známek, stupnice hodnocení, vytvářet si rozvrh, ukládat poznámky o domácích úkolech. Studentům je umožněno zobrazit si tzv. predikci, která spočívá v zadání klasifikace, které chce dosáhnout a systém pak vypočítá, jaké hodnocení musí dostat z příštího testu. Uživatel má možnost vytvořit si zálohu svých dat.

V porovnání s aplikací Klasifikace má Grade Tracker Pro množství funkcí navíc. Zobrazení klasifikace je na stejné úrovni. Data nejsou sdílená, a proto je na uživateli, jak si je bude držet aktuální. Aplikaci není vhodná pro učitele, ale studenti ocení velké množství funkcí jako je rozvrh nebo domácí úkoly.

2.1.4 Power School

Power school je komplexní informační systém pro školy. Umožňuje sdílené zobrazení známek, rozvrhů, úkolů, docházky a mnoho dalšího. Uživatelé jsou rozděleni podle práv a rolí, zahrnujících učitele, studenty i rodiče. Mobilní aplikace je funkční pouze po zadání přihlašovacích údajů uživatele a kódu vzdělávacího zařízení, které instanci systému spravuje.

Část aplikace sloužící pro zadávání a zobrazení hodnocení má podobnou funkcionalitu jako aplikace Klasifikace, umožňuje učitelům zadávat a upravovat hodnocení, což vyvolá odeslání notifikace dotčeným studentům. Klasifikace lze zobrazit i zpětně za minulá školní období. Bohužel jsem neměl možnost otestovat aplikaci v reálném provozu a vycházím pouze z informací a screenshotů z internetu, ale řekl bych, že tento systém je dobře funkční a může sloužit jako inspirace pro další vývoj aplikace Klasifikace. Na druhou stranu velké množství funkcí činí aplikaci místy nepřehlednou a pro nového uživatele se může zdát složitá.

2.1.5 Bigy Klasifikace

Aplikace pro studenty Biskupského gymnázia ve Žďáře nad Sázavou, která umí zobrazit klasifikaci přihlášeného uživatele. Klasifikaci si tedy student nemusí zadávat manuálně a stále vidí aktuální verzi svých známek.

Tato aplikace má z pohledu studenta podobou funkcionalitu jako aplikace Klasifikace, umí zobrazit studované předměty spolu se zadanou klasifikací k danému předmětu, ale neumí při změně obdržet a zobrazit notifikaci. Dále neobsahuje část pro učitele, ve které by mohli zadávat hodnocení studentům. Bohužel ani tuto aplikaci není možné bez přihlašovacích údajů v reálu otestovat, ale dle uváděného popisu na internetu a publikovaných screenshotů je možné ji hodnotit jako pěknou aplikaci pro zobrazení klasifikace s jednoduchým a funkčním uživatelským rozhraním.

2.1.6 Webové rozhraní systému Klasifikace zobrazené v mobilním zařízení

Systém Klasifikace má webové rozhraní, které je částečně optimalizované pro zobrazení ve webovém prohlížeči na mobilním zařízení. Z pohledu studenta je uživatelské prostředí vyhovující, klasifikace u jednotlivých předmětů se zobrazuje bez chyb. Z pohledu učitele je situace horší, na mobilním telefonu pro zadávání hodnot klasifikace nezbývá na displeji místo a jednotlivá vstupní pole jsou překryta jménem studenta. Zlepšení nastává při otevření webové Klasifikace na tabletu, kde už je uživatelské rozhraní opět funkční. Webová aplikace umí také do prohlížeče zasílat notifikace, které jsou však viditelné, pouze pokud má uživatel stránku otevřenou a zobrazenou.

Mobilní aplikace Klasifikace nabízí oproti webovému rozhraní možnost trvalého přihlášení, rychlejší odezvu a plně optimalizované uživatelské rozhraní. Dále umožňuje přijímat notifikace i když je aplikace na pozadí. Na druhou stranu je vyžadována instalace aplikačního balíčku do zařízení a v současné době je podporována pouze pro zařízení se systémem Android.

Tabulka 2.1: Shrnutí srovnání aplikací

	Školní známky, rozvrh hodin	School marks	Grade Tracker Pro	Power School	Bigy Klasifikace	Web Klasifikace	Mobilní aplikace Klasifikace
Přehledné a funkční uživatelské rozhraní	ne	ano	ne	nelze posoudit	ano	ne	ano
Sdílená klasifikace	ne	ne	ne	ano	ano	ano	ano
Možnost pro učitele zadávat klasifikaci	ne	ano	ne	ano	ne	ano	ano
Notifikace	ne	ne	ne	ano	ne	částečně	ano
Další funkce navíc	ano	ne	ano	ano	ne	ano	ne
Placená aplikace	ne	ne	ano	ano	ne	ne	ne
Obsahuje reklamy	ne	ne	ano	ne	ne	ne	ne

2.1.7 Shrnutí

Na základě provedeného srovnání různých aplikací a zhodnocení jejich funkcí jsem vytvořil tabulku 2.1, která porovnává jednotlivé produkty z hlediska různých kritérií, shrnuje klíčové funkce a použitelnost jednotlivých aplikací. Z tabulky nejlépe vychází aplikace PowerSchool, která poskytuje velké množství funkcí spolu se sdílenou klasifikací. Jednoduchostí a dobrou použitelností zaujala i aplikace School marks. V celkovém porovnání vychází mobilní aplikace Klasifikace jako jednoduchá, dobře použitelná a kvalitní aplikace pro studenty i učitele, která má přehledné a intuitivní uživatelské rozhraní, možnost sdílení dat a zasílání notifikací o změnách. Konkurenční řešení v sobě zahrnují i další rozšiřující funkce jako zobrazení a tvorba rozvrhu či ukládání informací o domácích úkolech. Tyto funkce v aplikaci Klasifikace v současnosti implementovány nejsou, ale jsou to dobré nápady na budoucí rozšíření a vývoj.

2.2 Požadavky

V souladu se zadáním práce a na základě technických možností systému Klasifikace v kontextu se srovnáním s podobnými existujícími aplikacemi jsem

definoval seznam funkčních a nefunkčních požadavků.

2.2.1 Funkční požadavky

1. **Přihlášení a odhlášení**

   K identifikaci uživatele pro přístup ke Klasifikaci a pro přístup k dalším datům z KOSu je potřeba, aby se uživatel přihlásil přes autorizační OAuth 2.0 server ČVUT. Po přihlášení dojde k uložení autorizačních dat. Uživatel má možnost se odhlásit, což způsobí vymazání autorizačních dat ze zařízení.

2. **Napojení na rozhraní webové aplikace Klasifikace**

   Webová aplikace Klasifikace poskytuje REST rozhraní pro přístup ke svým datům. Vytvářená aplikace se k tomuto rozhraní po přihlášení uživatele připojí a dle uživatelských akcí může číst nebo měnit data.

3. **Napojení na rozhraní KOSapi**

   Pro získání doplňujících informací o předmětech se aplikace připojuje k REST rozhraní KOSapi, které zprostředkovává přístup k vybrané části dat v databázi KOS.

4. **Zvolení semestru**

   Uživatel má možnost zvolit si semestr, ze kterého chce zobrazit data.

5. **Zobrazení studovaných předmětů**

   Aplikace umožňuje zobrazení předmětů, které přihlášený uživatel ve zvoleném semestru studuje či studoval. Pokud uživatel žádný předmět nestuduje, není možnost zobrazit studované předměty viditelná.

6. **Zobrazení klasifikace uživatele**

   Přihlášený uživatel si zobrazí svou klasifikaci k předmětům, které studuje, či studoval ve zvoleném semestru. Zobrazenou klasifikaci si může manuálně aktualizovat. Některé typy položek se od ostatních vizuálně zvýrazňují, konkrétně to jsou:

   - Zápočet
   - Semestrální test
   - Zkouškový test
   - Ústní zkouška
   - Výsledné hodnocení

7. **Zobrazení předmětů pro klasifikování**

   Aplikace umožňuje zobrazení předmětů, ke kterým má přihlášený uživatel ve zvoleném semestru právo zadávat hodnocení. Pokud uživatel žádný takový předmět nemá, není možnost zobrazit předměty pro klasifikování viditelná.

8. **Zadání a uložení klasifikace**

   Aplikace umožní, uživateli s oprávněním k danému předmětu, zadávat klasifikaci. Uživatel si zvolí, zda chce hodnotit jednu položku klasifikace u studentů podle skupin, nebo zda chce vyplnit kompletní klasifikaci daného studenta. Po ohodnocení a potvrzení se klasifikace uloží.

9. **Notifikace**

   Studenti obdrží notifikaci při změně své klasifikace u předmětů, které studují. Notifikace se po rozkliknutí v aplikaci automaticky označí jako přečtená. Dále má uživatel možnost manuálně označit všechny notifikace jako přečtené.

2.2.2 Nefunkční požadavky

1. **OS Android**

   Aplikace využívá OS Android od verze 4.4 Kitkat (API 19). Tuto nebo vyšší verzi využívá v současné době přes 94% zařízení, což je dostatečná podpora.

2. **OAuth 2.0**

   Pro autorizaci bude využit standard OAuth 2.0, který je implementován autorizačním OAuth 2.0 serverem ČVUT a umožňuje jednoduchý přístup pro autorizaci uživatelů z aplikací třetích stran.

3. **Programovací jazyk Kotlin**

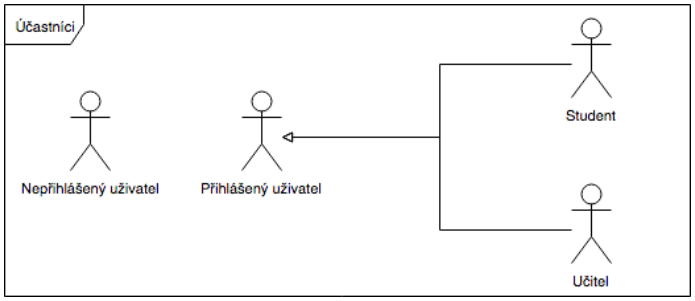
   Při implementaci bude využit programovací jazyk Kotlin. Jedná se o moderní a poměrně mladý jazyk, který je spolu s programovacím jazykem Java oficiálně podporován pro vývoj na platformě Android.

4. **Uživatelské rozhraní**

   Uživatelské rozhraní odpovídá standardům pro platformu Android a dbá na Best practices dle standardu pro uživatelská rozhraní Material design.

5. **Podpora více jazyků**

   Aplikace má dvě jazykové lokalizace - anglickou a českou. Angličtina je výchozí nastavení aplikace a čeština se nastaví automaticky dle jazyku systému.



Obrázek 2.1: Role, kterých mohou uživatelé nabývat

6. **Push notifikace s použitím notifikačního serveru**

   Aplikace bude umět přijímat PUSH notifikace s využitím Firebase messaging systému. Firebase messaging je standardní notifikační služba podporovaná systémem Android.

2.3 Případy užití

Na základě požadavků vytvářím případy užití. Nejprve definuji účastníky, na základě kterých rozděluji případy užití do tří částí podle funkčních celků.

2.3.1 Účastníci

Uživatelé aplikace mohou nabývat několika rolí:

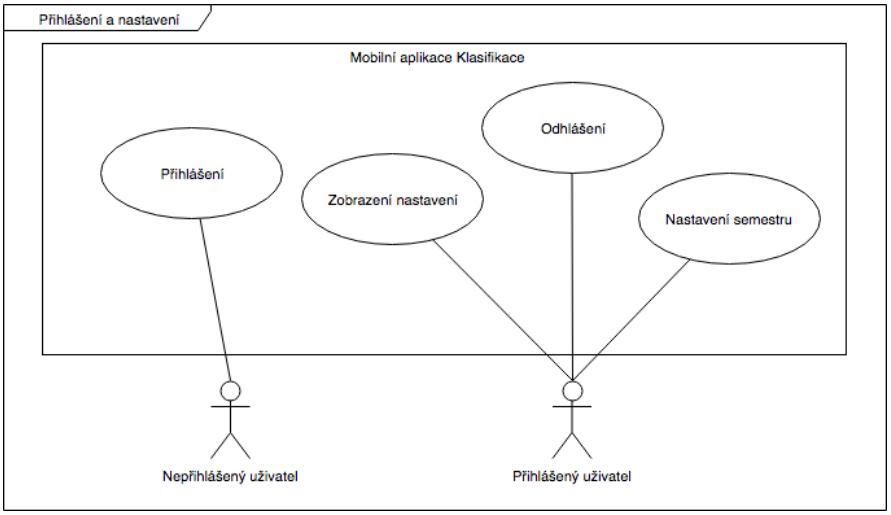
- Nepřihlášený
- Přihlášený
- Student
- Učitel

Každá role má v aplikaci přístup k určitým funkcím. Některé funkce jsou dostupné pro roli učitele i studenta, ty se dědí ze společného účastníka, kterým je Přihlášený uživatel. Aplikace umožňuje i společnou kombinaci rolí student a učitel.

2.3.2 Případy užití týkající se autentifikace a nastavení

Diagram 2.2 a text popisují případy užití týkající se autentifikace uživatele v mobilní aplikaci a nastavení.

- **Přihlášení**



Obrázek 2.2: Případy užití týkající se autentifikace a nastavení

Nepřihlášenému uživateli je umožněno přihlásit se pomocí autorizačního OAuth 2.0 systému, který je poskytován ČVUT.

- **Odhlášení**

  Přihlášený uživatel se může v nastavení odhlásit, což způsobí smazání všech uživatelských dat a cookie, které si aplikace vytvořila.

- **Zobrazení nastavení**

  Uživatel si může zobrazit nastavení, kde je možné se odhlásit a kde je vidět aktuálně nastavený semestr.

- **Nastavení semestru**

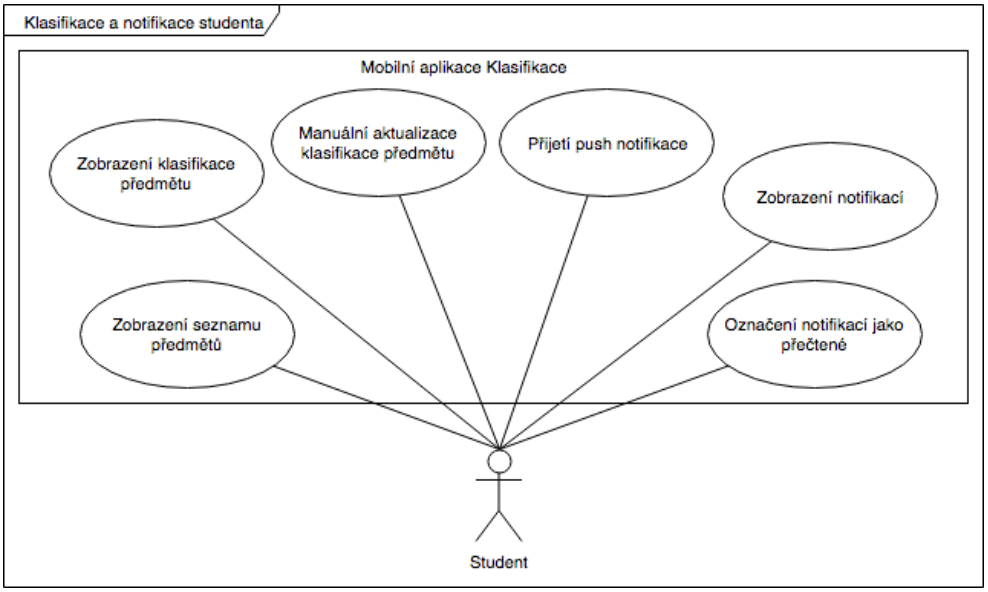
  V nastavení lze měnit semestr aktuálně zobrazených dat. Změna lze provést výběrem kódu semestru z nabídky.

2.3.3 Případy užití týkající se zobrazení klasifikace a notifikací

Dále jsem identifikoval několik případů užití, které se týkají uživatele s rolí student. V následujícím textu a diagramu 2.3 jsou znázorněny a popsány.

- **Zobrazení seznamu předmětů**

  Studentovi je zobrazen seznam předmětů, které studuje nebo studoval ve zvoleném semestru. Má přehledně viditelný název předmětu, kód a počet nových upozornění, které se k danému předmětu vztahují.



Obrázek 2.3: Případy užití týkající se zobrazení klasifikace a notifikací

- **Zobrazení klasifikace předmětu**

  Student si může zobrazit klasifikaci předmětu, který studuje nebo studoval. Důležité položky klasifikace (zápočet, semestrální test, ústní zkouška, zkouškový test a výsledné hodnocení) jsou oproti ostatním barevně zvýrazněné.

- **Manuální aktualizace klasifikace předmětu**

  Zobrazenou klasifikaci si uživatel může manuálně aktualizovat, čímž se seznam znovu načte a zobrazí se nové změny.

- **Přijetí push notifikace**

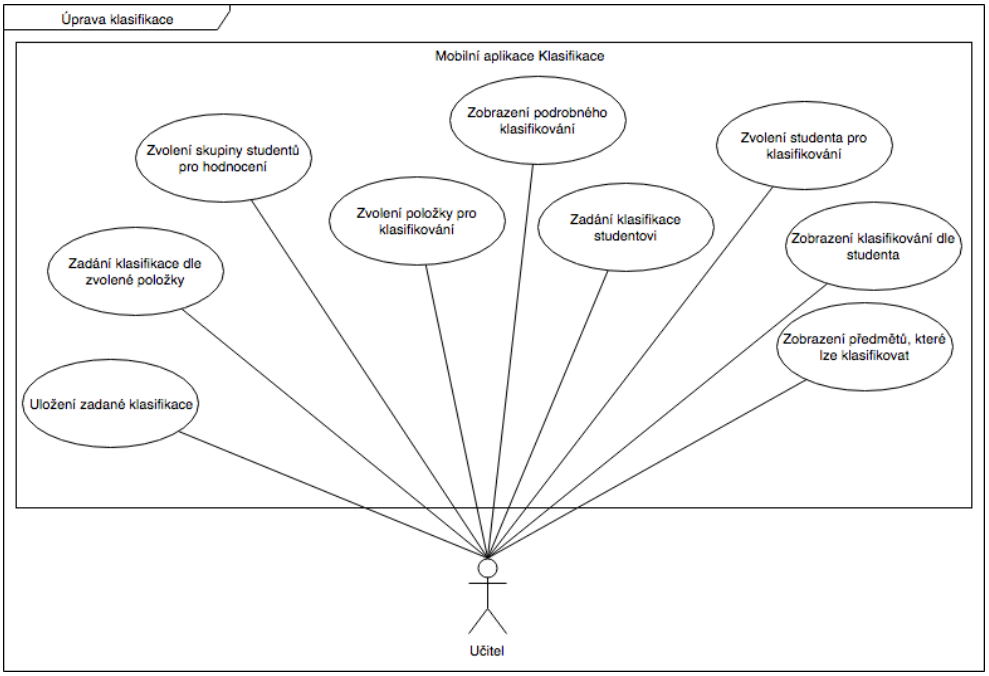
  Uživateli se automaticky zobrazí notifikace týkající se změn v klasifikaci. Lze je rozkliknout a tím zobrazit klasifikaci konkrétního předmětu. Zároveň se tato notifikace označí jako přečtená a skryje se.

- **Zobrazení notifikací**

  Student si může zobrazit seznam všech notifikací, nové notifikace jsou odděleny od přečtených. Ze seznamu lze přejít kliknutím na notifikaci ke klasifikaci konkrétního předmětu. Tím se notifikace označí jako přečtená

- **Označení notifikací jako přečtené**

  Je možné hromadně označit všechny své nepřečtené notifikace jako přečtené.



Obrázek 2.4: Případy užití týkající se úpravy klasifikace

2.3.4 Případy užití týkající se úpravy klasifikace

Uživatel v roli učitel má právo měnit klasifikaci u předmětů, které učí a jsou mu k dispozici tyto případy užití:

- **Zobrazení předmětů, které lze klasifikovat**

  Zobrazuje se seznam předmětů, u kterých má právo přidávat a měnit klasifikaci ve zvoleném semestru. Každý záznam obsahuje kód a název předmětu, informaci s počtem zapsaných studentů a tlačítka pro přechod k zadávání klasifikace.

- **Zobrazení klasifikování dle studenta**

  Tato volba otevře možnost klasifikování dle studenta, kterého si uživatel vyhledá. Zobrazen je seznam všech položek klasifikace, případně jejich vyplněné hodnoty a poznámka.

- **Zvolení studenta pro klasifikování**

  Učitel má možnost zvolit studenta, kterého chce klasifikovat. Seznam všech studentů lze filtrovat dle jména nebo uživatelského jména.

- **Zadání klasifikace studentovi**

  Je možné zadávat nebo měnit hodnoty položek klasifikace vybraného studenta a přidávat nebo upravovat poznámku.

- **Zobrazení podrobného klasifikování**

  Učitel může zobrazit podrobné hodnocení, kde vidí seznam hodnot a poznámky jedné položky klasifikace pro všechny studenty z nějaké skupiny.

- **Zvolení položky pro klasifikování**

  Lze zvolit, kterou položku klasifikace chce v podrobném klasifikování měnit. Výběr provede ze seznamu, který zobrazuje jméno a kód každé položky.

- **Zvolení skupiny studentů pro klasifikování**

  Učitel si vybere, kterou skupinu studentů chce v podrobném klasifikování hodnotit. Výběr provede ze seznamu všech položek, který zobrazuje název skupiny.

- **Zadání klasifikace dle zvolené položky**

  Po zvolení položky a skupiny je umožněné zadávat nebo měnit hodnoty vybrané klasifikace všem studentům z vybrané skupiny. Ke každé může také přidávat nebo upravovat poznámku.

- **Uložení zadané klasifikace**

  Zadanou nebo změněnou klasifikaci musí učitel uložit, čímž se odešle na server. Pokud došlo ke změně v klasifikaci a následné navigaci zpět, zobrazí se učiteli potvrzovací dialog, který na tuto skutečnost upozorní a ze kterého půjde změny dodatečně uložit.

2.4 Možná řešení notifikačního serveru

Jedním z cílů této práce je rešerše notifikačních serverů a volba vhodného řešení. V první části práce jsem se věnoval notifikacím obecně a notifikačním službám operačních systémů iOS a Android. Dále bylo zmíněno, že pro oddělení logiky aplikace od logiky zasílání notifikací je dobré využít tzv. notifikační server, který poskytne jednotné rozhraní pro zasílání notifikací uživatelům a sám poté rozešle notifikace přes notifikační služby daných platforem na všechna zařízení daného uživatele. V této kapitole popisuji existující platformy, které toto fungování umožňují a porovnávám je s variantou vlastní implementace notifikačního serveru.

2.4.1 OneSignal

Platforma pro rozesílání notifikací OneSignal je zdarma a podporuje řadu cílových zařízení. Umožňuje hromadné rozesílání notifikací podle různých nastavení či filtrů ale i na konkrétní zařízení. Dále umí plánovat automatické rozesílání notifikací, rozlišovat odeslaný obsah zprávy dle jazyku cílového zařízení.

Vzhled přijatých notifikací je možné centrálně měnit, přidávat do notifikací různá tlačítka a další objekty.

Na druhou stranu systém nepodporuje posílání notifikace na více zařízení jednoho uživatele. Toto omezení lze lehce obejít nastavením štítku ke každému zařízení daného uživatele a následně notifikaci odesílat pomocí tohoto štítku. Další nevýhodou je nutnost přidávat do aplikace další závislosti na OneSignal SDK.

2.4.2 Urban Airship

Urban Airship je v současnosti jednou z největších platforem pro rozesílání notifikací, která navíc umí mnoho podpůrných funkcí, umožňuje vytvářet různé analýzy a statistiky. Notifikace lze posílat přímo určitým zařízením, ale Urban Airship vyniká hlavně v možnosti tvořit komplexní skupiny příjemců na kombinace širokého množství parametrů cílových zařízení.

Nevýhodou je, že platforma je placená pro projekty s více než jedním tisícem aktivních zařízeních a také vyžaduje přidání závislosti do projektu aplikace.

2.4.3 Carnival.io

Platforma Carnival.io se specializuje na personalizaci a automatizaci notifikací. Umožňuje vytvářet časový rozvrh, kdy a komu se budou notifikace rozesílat. Jednotlivé notifikace lze personalizovat, takže například může být do textu notifikace automaticky doplněno jméno či nějaký stav cílového uživatele. Další specialitou Carnival.io je množství různých druhů notifikací, lze posílat obyčejný text, ale i obrázky či další multimediální obsah. Platforma se postará o logiku i design zobrazené notifikace. Bohužel je však použití této platformy placené.

2.4.4 Catapush

Catapush je dalším příkladem notifikační platformy, která ale navíc umí tzv. obousměrnou komunikaci, kdy je možné odesílat odpovědi na jednotlivé notifikace či zprávy. Každou odeslanou notifikaci lze automaticky personalizovat a následně sledovat její status. V případě, kdy se doručení nepodaří, má systém mechanismy pro automatické znovuodeslání, případně umožňuje odeslat kritické notifikace jako SMS. Využití služby Catapush je zpoplatněno dle počtu aktivních uživatelů za měsíc, za každou odeslanou SMS a další příplatek je účtován v případě využití obousměrné komunikace.

2.4.5 Uniqush

Na rozdíl od ostatních zmíněných platforem je Uniqush volně k využití jako open source. Uživatel má možnost stáhnout sestavený program, který jed-

noduše spustí na vlastním serveru, kde ho také sám spravuje. Komunikace se Uniqush serverem probíhá pomocí HTTP API a ani klientské aplikace, ani systém se zdrojem dat nepotřebují přidávat závislosti, které by umožňovaly fungování notifikací. Systém poskytuje jednoduché API pro nastavení připojení k jednotlivým notifikačním službám platforem, registrací koncových zařízení a posílání notifikací. Neobsahuje žádné další funkce navíc, jedná se pouze o jednoduché odesílání notifikací.

2.4.6 Vlastní řešení

Jelikož notifikace v rámci systému Klasifikace nevyžadují žádnou složitou logiku, není potřeba posílat notifikace podle různých segmentů nebo různých parametrů, je jednou z možností řešení notifikačního serveru vlastní implementace. Použití tohoto přístupu má řadu výhod. Vlastní řešení je vytvořené na míru, umožňuje plnou kontrolu nad odesíláním notifikací do notifikačních služeb a nevyžaduje přidání extra závislostí do mobilní aplikace.

2.4.7 Shrnutí

Všechny uvedené možnosti a platformy oddělující logiku webového systému od logiky zasílání push notifikací dle jednotlivých platforem by pro případ systému Klasifikace dobře fungovaly. Popsaná komerční řešení se kromě jednoduchého odesílání notifikací zaměřují hlavně na automatizaci a personalizaci hromadných notifikací, které cílí na nějakou skupinu uživatel. Kvůli tomu je nutné přidávat do mobilních aplikací závislosti na knihovnách daných platforem. Jelikož takové komplexní řešení není třeba a je dobré držet nízký počet závislostí, tyto systémy nepoužiji.

Vhodnou variantou by mohlo být využití open source software Uniqush, nicméně jsem se nakonec rozhodl pro vytvoření vlastní implementace notifikačního serveru, který bude poskytovat jednoduché rozhraní pro posílání notifikací a bude obsahovat pouze nezbytně nutná, pseudonymizovaná data tak, aby odpovídala principům GDPR, kterému se věnuji v následující kapitole.

2.5 GDPR

Obecné nařízení na ochranu osobních údajů neboli GDPR (General Data Protection Regulation) představuje aktualizovaný právní rámec ochrany osobních údajů v evropském prostoru, který bude od 25. května 2018 přímo stanovovat pravidla pro zpracování osobních údajů, včetně práv subjektu údajů (fyzické osoby). V českém právním prostředí tak obecné nařízení od 25. května 2018 nahradí zákon č. 101/2000 Sb., o ochraně osobních údajů, který v současné době stanovuje povinnosti a práva při zpracování osobních údajů. [18]

Nařízení míří na firmy, instituce i jednotlivce, kteří zacházejí s osobními údaji – zaměstnanců, zákazníků, klientů či dodavatelů, a to napříč segmenty

a odvětvími. Zasáhne i ty, kteří sledují či analyzují chování uživatelů na webu, při používání aplikací nebo chytrých technologií. Cílem GDPR je chránit digitální práva občanů EU. [19]

2.5.1 Osobní údaje z pohledu GDPR

Osobní údaj je z pohledu GDPR nějaká informace o identifikované nebo identifikovatelné osobě (subjektu údajů). Kromě zřejmých osobních dat jako je jméno, příjmení, adresa nebo fotografie jsou osobními údaji i data, která mohou vést k identifikaci osoby nepřímo. Jedná se například IP adresy, emailové adresy nebo dokonce interní vygenerovaná identifikační čísla v databázi. Vyjmenované jsou pouze příklady pro znázornění obecnosti pojmu, co vše je, nebo není osobní údaj, záleží vždy na konkrétní situaci.

Speciální skupinu tvoří navíc tzv. zvláštní kategorie osobních údajů, jsou to citlivé osobní údaje jako rasový původ, náboženské vyznání, zdravotní stav, biometrické údaje např. otisky prstů a další. Tyto údaje podléhají zvláštním opatřením, lze je zpracovávat pouze ve zvláštních, jasně stanovených případech a pro samotné zpracování platí přísnější pravidla a jejich správce má více povinností.

2.5.2 Anonymizace a pseudonymizace

Pseudonimizovaná data jsou data upravená takovým způsobem, že z nich není možné určit konkrétní osobu bez znalosti nějakého klíče. Nejde tedy o nevratný proces, ale pouze o dočasné funkční oddělení určité množiny údajů od ostatních údajů, které však společně identifikaci umožňují.

I interní ID nějaké osoby je tedy osobním údajem, a to i v případě, kdy v daném souboru dat není možné ID ke konkrétní osobě přiřadit, ale někde existuje klíč, který tuto identifikaci umožní. Například správce údajů shromažďuje údaje se jménem, příjmením, věkem a nejvyšším dosaženým vzděláním. Jméno a příjmení nahradí číselným kódem a zvlášť uchovává soubor se jménem, příjmením a tímto číselným kódem. Kdo má přístup k oběma souborům zároveň, je schopen určit, ke komu se informace o věku a vzdělání vztahují, a proto je tyto údaje potřeba chránit výše uvedenými předpisy. Existence dvou oddělených souborů informací jsou pro pseudonymizaci klíčové. [20]

Anonymizace je nevratný proces bez možnosti zpětné identifikace osoby, data osoby se po tomto procesu již nikdy nedají určit či dohledat. Anonymizovaná data, vzniklá odstraněním informací umožňující identifikovat danou osobu, ochraně osobních údajů nepodléhají. Příkladem může být e-shop, který zpracovává pouze informace o věku a obci zákazníků. Ostatní data, která nepotřebuje pro své podnikání, maže.

2.5.3 Označení aktérů v rámci GDPR

GDPR rozlišuje 3 různé aktéry, kterých se týká. Jedná se o:

1. Subjekt údajů, tedy o fyzickou osobu, jejíž osobní údaje mají být zpracovány.

2. Správce údajů, což je nějaká fyzická či právnická osoba nebo státní instituce, která potřebuje zpracovávat osobní údaje subjektu údajů.

3. Zpracovatel údajů, který je pověřen správcem údajů. Správce musí mít se zpracovatelem písemnou smlouvu, kterou lze řetězit (zpracovatel pověří dalšího zpracovatele).

V některých případech je hranice mezi správcem údajů a zpracovatelem tenká. Správci i zpracovatelé mají dle GDPR specifické povinnosti. Musí dodržovat principy nařízení, při vzniku bezpečnostních incidentů musí skutečnost ohlásit Úřadu na ochranu osobních údajů a pokud jde o závažný incident i informovat subjekty údajů. U organizací s více než 250 zaměstnanci je navíc nutnost vést záznamy o zpracování osobních údajů a zajistit si tzv. pověřence na ochranu osobních údajů, který monitoruje, zda je zpracování údajů v souladu s tímto nařízením. [21]

2.5.4 Oprávnění pro zpracování dat

Aby mohl jakýkoli správce zpracovávat osobní údaje musí k tomu být oprávněn. Toho lze docílit několika způsoby [22]:

1. Zpracování informací subjektu ukládá zákon.

2. Zpracování informací je ve veřejném zájmu.

3. Zpracování informací je v životně důležitém zájmu fyzické osoby.

4. Správce má oprávněný zájem na zpracování údajů (např. banka poskytující půjčku fyzické osobě má právo zpracovávat informace o příjmech a pohledávkách dané osoby).

5. Správce potřebuje informace pro plnění smlouvy s danou osobou.

6. Fyzická osoba udělí souhlas se zpracováním svých osobních údajů.

2.5.5 Hlavní principy

Mezi zásady Obecného nařízení patří [23]:

- zákonnost, korektnost, transparentnost – správce musí zpracovávat osobní údaje na základě nejméně jednoho právního důvodu a vůči subjektu údajů transparentně,

- omezení účelu – osobní údaje musí být shromažďovány pro určité a legitimní účely a nesmějí být zpracovávány neslučitelným způsobem s těmito účely,
- minimalizace údajů – osobní údaje musí být přiměřené a relevantní ve vztahu k účelu, pro který jsou zpracovávány,
- přesnost – osobní údaje musí být přesné,
- omezení uložení – osobní údaje by měly být uloženy ve formě umožňující identifikaci subjektu údajů jen po nezbytnou dobu pro dané účely, pro které jsou zpracovávány,
- integrita a důvěrnost – technické a organizační zabezpečení osobních údajů.

2.5.6 Technické aspekty dodržování principů GDPR v IT

Z výše uvedených principů vyplývají některé zásady, které se musí při implementaci a správě IT systémů dodržovat. Uživatel musí být upozorněn na to, jaká data a za jakým účelem jsou o něm zpracovávána, pokud systém shromažďuje více údajů, než je oprávněn, musí si vyžádat od uživatele souhlas. Systém by měl zůstat funkční i v případě, že uživatel zpracování údajů navíc neautorizuje. Pro udělení souhlasu musí být zobrazen samostatný prvek (např. tlačítko nebo checkbox), který nesmí být dopředu označen.

Dalším principem je navržení samotného systému tak, aby bylo zajištěna ochrana osobních údajů. K tomu slouží například již zmiňovaná pseudonymizace, která oddělí informace potřebné k identifikaci osoby od ostatních údajů, nebo lze použít nějaké bezpečné šifrování těchto údajů. Dále by návrh měl počítat pouze se zpracováním těch údajů, které jsou nezbytně nutné.

Dalším aspektem, jímž je nutné se při návrhu informačního systému zabývat, je odvolatelnost souhlasu se zpracováním osobních údajů. Subjekt údajů má možnost svůj souhlas kdykoliv odvolat a to podobným způsobem, jakým souhlas udělil. Pokud je tedy možné udělit souhlas pouhým stisknutím tlačítka, je nutné, aby existovalo obdobné tlačítko, které souhlas odvolá.

2.5.7 Mobilní aplikace Klasifikace a GDPR

Mobilní aplikace Klasifikace komunikuje s API systému Klasifikace, ze kterého získává data o subjektu údajů. Některé získané informace ukládá do vnitřní paměti zařízení. Pokud není systém Android uživatelem upraven nebo napaden nějakým škodlivým software, není možné data jinak než v aplikaci přečíst. Své osobní údaje má tedy uživatel pod svojí kontrolou a může je snadno smazat odhlášením, smazáním dat nebo kompletním odinstalováním aplikace. Pro zachování principu transparentnosti se uživateli po přihlášení zobrazí dialog, který toto chování popisuje.

Při pádu aplikace z důvodu nějaké chyby se odesílají anonymizované údaje o této události do Firebase Crashlitics. Jelikož se jedná o anonymizovaná data, tak se na ně toto nařízení nevztahují. Nicméně i o této skutečnosti je uživatel po přihlášení informován.

Posledním případem, kdy je nutné vyžádat souhlas se zpracováním osobních údajů, jsou push notifikace, protože pro fungování notifikačního serveru je nutné s ním sdílet informace o notifikačním tokenu a typu zařízení spolu s mapováním na daného uživatele. Jelikož push notifikace nejsou pro fungování aplikace jako celku nezbytné, je možnost povolení notifikací spolu se souhlasem o zpracování osobních údajů oddělena a dostupná z obrazovky Nastavení. Pro odebrání souhlasu stačí v nastavení notifikace vypnout, o čemž je uživatel při jeho udělování informován.

2.5.8 Notifikační server a GDPR

Notifikační server zpracovává osobní údaje o uživatelích, kteří si v mobilní aplikaci zapnuli možnost přijímání push notifikací a souhlasili se zpracováním svých údajů na notifikačním serveru. Notifikační server je navržen tak, aby byl v souladu s principy GDPR a neukládal údaje, které nepotřebuje. Mapování uživatele a jeho notifikačních tokenů je pseudonymizované tak, že identifikaci konkrétního uživatele není možné pouze s daty uloženými v databázi notifikačního serveru získat. Databáze obsahuje pouze ID uživatele.

Samotné odesílání notifikací neumožňuje posílat osobní údaje příjemce, obsahuje pouze ID a typ zprávy. Aplikace v notifikovaném zařízení si poté musí sama stáhnout potřebné informace z API systému Klasifikace. Celkový návrh notifikačního serveru je v souladu s principy GDPR.

Kapitola 3

Návrh

V kapitole Návrh se na základě provedené analýzy věnuji návrhu uživatelského rozhraní a architektury mobilní aplikace, popisu potřebných částí API Klasifikace a přihlášení přes OAuth 2.0 server. Z hlediska notifikačního serveru provádím návrh REST rozhraní, které bude server poskytovat a popis API, které bude naopak využívat.

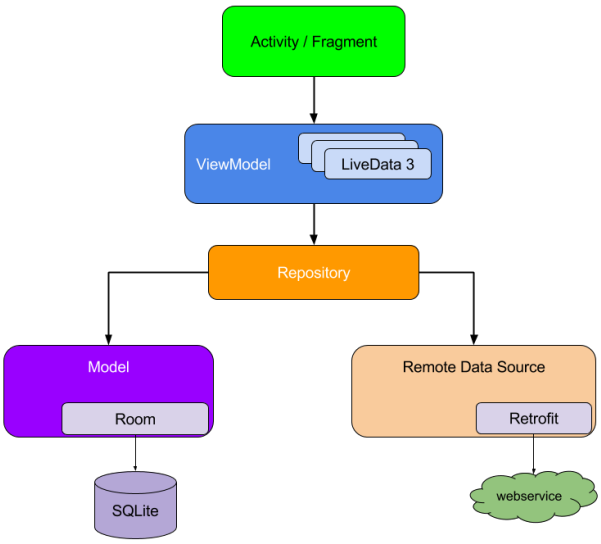
3.1 Architektura

Aplikace bude využívat architekturu MVVC (Model, View, View Model) s tím, že část Model, která obsahuje data a logiku nad nimi, bude rozdělena na další dvě podvrstvy a to na datovou vrstvu a aplikační vrstvu, které jsou známé z třívrstvé architektury. Tato architektura je v dokumentaci pro vývoj aplikací na Android doporučená a je znázorněna na obrázku 3.1).

3.2 Zvolené technologie

O možných technologiích a postupech jsem již psal v předchozích částech práce, zde pouze shrnuji zvolené technologie a knihovny, které podporují navrženou architekturu a které při vývoji použiji. Mobilní aplikaci vyvíjím pro zařízení se systémem Android od verze 4.4 Kitkat v programovacím jazyku Kotlin. Základ aplikace bude tvořit Android App Skeleton. Pro uložení dat využiji pro Android standardní, databázový systém SQLite spolu s knihovnou Room, která zajistí snazší přístup k datům. Požadavky na API různých internetových zdrojů budu tvořit pomocí knihovny Retrofit 2. Vnitřní fungování aplikace bude využívat reaktivních principů za podpory knihovny RxJava 2 a knihovny LiveData.

Notifikační server vyvíjím v programovacím jazyku Java za použití frameworku Spring a Hibernate.



Obrázek 3.1: Architektura aplikace

3.3 Rest API aplikace Klasifikace

Webová Aplikace Klasifikace poskytuje REST rozhraní (REST Api), čímž umožňuje aplikacím třetích stran provádět některé operace, které nabízí její uživatelské rozhraní. Pro přístup k REST Api je nutné získat přístupový token pomocí přihlášení přes OAuth 2.0 poskytované ČVUT. Jelikož cílem práce není implementovat všechny funkcionality webové aplikace, využiji jen část tohoto REST rozhraní. Konkrétně používám tyto části API (uvedený popis je obecný a nejedná se o přesnou dokumentaci, ta je dostupná online z [24].

- **classification-controler** - Soubor endpointů umožňující práci s definicí klasifikace. Pro účely mobilní aplikace využiji pouze požadavek, který vrací definice hodnocení v daném předmětu.
- **login-controller** - Soubor endpointů pro získání informací o aktuálně přihlášeném uživateli. Umožňuje ověřit přihlašovací token, získat informace o uživateli a předměty, ke kterým má vztah jako učitel či jako student.
- **notification-controller** - Rozhraní pro práci s notifikacemi, obsahuje endpointy pro získání všech nebo jen nepřečtených notifikací daného uživatele, označení notifikace jako přečtené nebo nepřečtené a označení všech notifikací jako přečtené nebo nepřečtené. Z uvedených možností využívám všechny, kromě endpointů pro označení notifikací jako nepřečtené.

- **student-classification-controller** - Umožňuje získání a uložení hodnot klasifikace. Klasifikace k danému předmětu lze získat buď pro studenta nebo pro skupinu a konkrétní položku klasifikace, nebo všechna hodnocení všech studentů z nějaké skupiny. Poslední jmenovaná možnost v aplikaci není potřeba, protože při zadávání klasifikace lze vždy využít předchozí dva.

- **student-group-controller** - Obsahuje pouze jediný přístupový bod, ze kterého lze získat skupiny studentů v daném předmětu. Odpověď z tohoto requestu vrací identifikátory názvy a popis jednotlivých skupin. Názvy a popis obsahují proměnné ve složených závorkách, které musí příjemce nahradit správnými textovými hodnotami.

3.4 Přihlášení přes OAuth 2.0

V první části práce se v kapitole OAuth 2.0 věnuji obecnému popisu autorizačního protokolu OAuth 2.0. Na jeho základě provádím návrh realizace přihlášení v mobilní aplikaci Klasifikace. Specifikace a způsob komunikace je standardem definován, důležité je ale zvolit vhodný grant type. Nejlepší variantou je použití Authorization code bez tajného klíče klienta nebo s rozšířením PKCE. Další možností je použít, dnes již méně využívaný, grant type Implicit. Bohužel ani jednu z těchto variant ČVUT OAuth server nepodporuje a tím pádem zbývá standardní grant type Authorization code. V tomto typu musí dle specifikace zůstat Client Secret skryté. K tomu se použije prostředník mezi OAuth serverem a aplikací, který bude mít Client Secret uložené, vyjedná získání nebo aktualizaci tokenu a vrátí ho do aplikace. Implementace prostředníka není předmětem této práce, nicméně je pro fungování aplikace nezbytná, a proto ji provedu, ale nebudu se jí obsáhleji věnovat.

Jednotlivé kroky provedené pro získání OAuth 2.0 tokenu jsou:

1. Otevření autorizačního serveru ve webovém prohlížeči. Otevíraná url obsahuje parametry Client ID, scope, redirect uri a response type.

2. Uživatel se přihlásí.

3. Autorizační server vrátí redirection uri spolu s jednorázovým kódem.

4. Aplikace si vyžádá OAuth token přes prostředníka, kterému v požadavku předá jednorázový kód, Client ID, Redirect uri a grant type.

5. Prostředník přidá k těmto údajům Authorization hlavičku, jejíž hodnota je Basic ClientID:ClientSecret.

6. Autorizační server vrátí odpověď, kterou prostředník předá aplikaci.

7. Aplikace si z odpovědi uloží token a refresh token a je přihlášená.

3.5 Uživatelské rozhraní

Na základě definovaných požadavků a případů užití jsem navrhl uživatelské rozhraní a vytvořil jednoduchý prototyp v programu JustInMind, který umožňuje rychlou tvorbu návrhu a je možné se jednoduše proklikávat. Kompletní prototyp, který lze zobrazit pouze v programu JustInMind, lze nalézt v přiloženém CD.

Při tvorbě uživatelského prostředí jsem se držel standardu pro tvorbu uživatelských rozhraní Material Design a principů v něm uvedeným, návrh v této kapitole popíši a zdůvodním uživatelské rozhraní jednotlivých obrazovek.

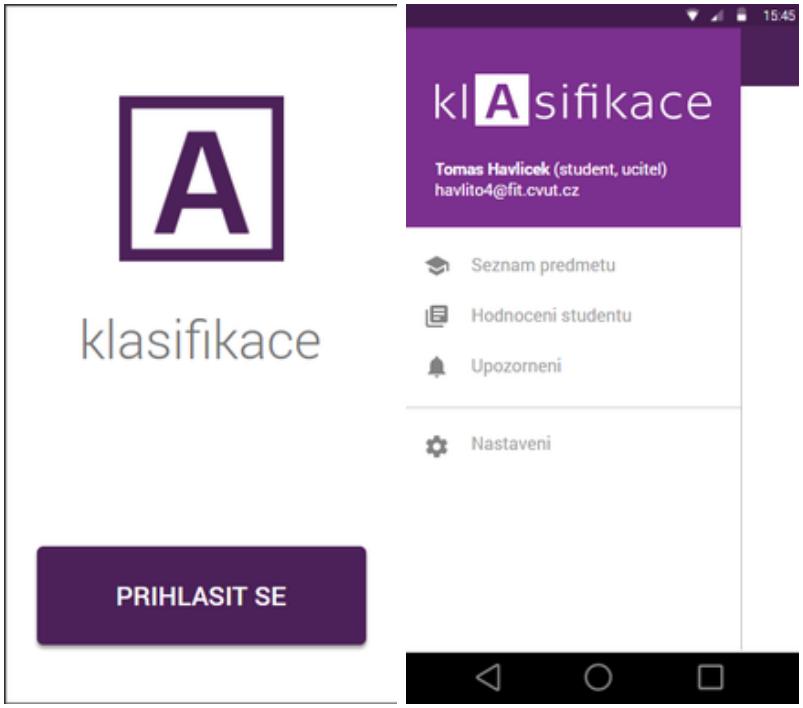
3.5.1 Přihlášení a základní menu

Rozhraní pro přihlášení má dvě jednoduché obrazovky. První z nich, která je vidět na obrázku 3.2 vlevo, využívá jednoduchý a čistý design, obsahuje jen logo a název aplikace s velkým a jasně zvýrazněným tlačítkem pro přihlášení. Po stisknutí tlačítka se zobrazí druhá obrazovka, obsahující webový prohlížeč, ve kterém se načte přihlašovací stránka OAuth 2.0 FIT ČVUT. Nad webovým prohlížečem je v horní části obrazovky zobrazena webová adresa OAuth serveru, na které se uživatel aktuálně nachází. Poté, co uživatel vyplní své přihlašovací údaje a úspěšně se přihlásí, je přesměrován na seznam předmětů.

Po úspěšném přihlášení se zpřístupní menu aplikace na obrázku (...) vpravo, které je dostupné ze všech obrazovek první úrovně a dá se vyvolat kliknutím na ikonku tří čárek v levém horním rohu aplikace, nebo gestem přejetí prstu po obrazovce z levého okraje displeje směrem doprava. Menu obsahuje až čtyři položky a to Seznam předmětů, Hodnocení Studentů, Upozornění a Nastavení. Položka Seznam předmětů je zobrazena pouze pokud má uživatel roli student, položka Hodnocení studentů je naopak dostupná pouze pro roli učitele. Menu dále obsahuje hlavičku, kde je zobrazeno jméno, email a role aktuálně přihlášeného uživatele. Návrh menu je zobrazen na obrázku 3.2 vpravo a využívá jeden ze vzorů pro tvorbu navigace dle standardu Material Design nazývaný Navigation Drawer.

3.5.2 Seznam předmětů

Po úspěšném přihlášení se zobrazí uživateli seznam předmětů dle jeho role. V roli student nebo v roli student a učitel se zobrazí seznam studovaných předmětů, jehož návrh je vidět na obrázku 3.3 vlevo. Každá položka obsahuje název, kód předmětu, semestr a počet nových upozornění. Kliknutím na položku určitého předmětu si uživatel zobrazí svou klasifikaci. V případě, že má uživatel pouze roli učitel, je mu zobrazena obrazovka se seznamem předmětů (na obrázku 3.3 vpravo), u kterých může zadávat hodnocení. Jednotlivé položky seznamu obsahují název a kód předmětu, semestr a dvě tlačítka. Jedno pro zadávání klasifikace dle studenta a druhé pro zapisování detailního



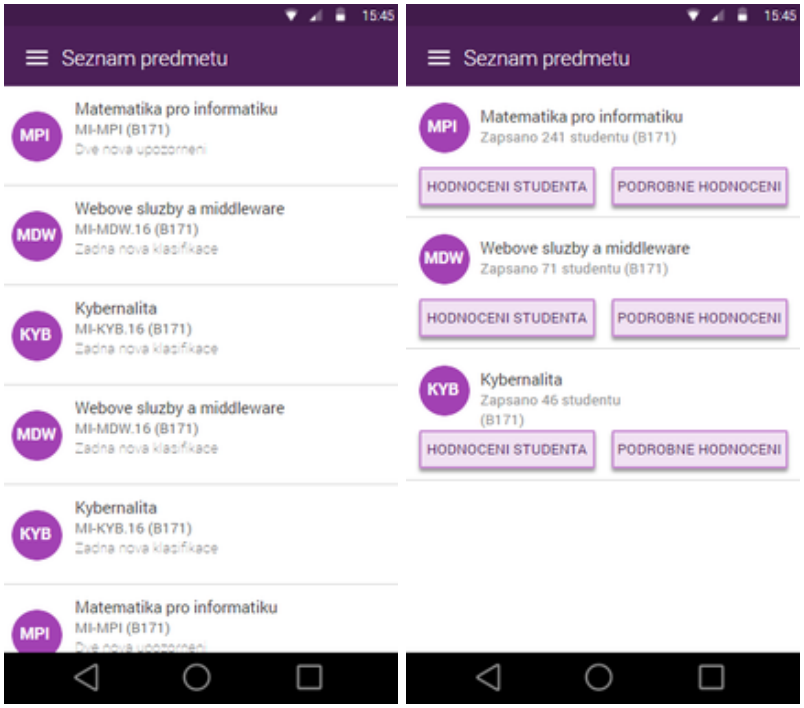
Obrázek 3.2: Přihlášení a základní menu

hodnocení dle vybrané skupiny a položky klasifikace. Oba dva seznamy jsou opět vytvořeny v souladu se standardem Material Design, který uvádí přesné umístění a velikost prvků v jednotlivých položkách. Tlačítka pro hodnocení v seznamu předmětů v roli učitel jsou pro zachování vysoké použitelnosti zobrazeny vedle sebe, aby se při ovládání prsty uživatel dostal tam, kam chce i při méně přesném kliknutí.

3.5.3 Zobrazení klasifikace předmětu z pohledu studenta

Kliknutím na položku v seznamu studovaných předmětů se uživatel-student dostane na konkrétní hodnocení zvoleného předmětu (na obrázku 3.4)), které obsahuje názvy jednotlivých položek, jejich typ a hodnotu. Případně, není-li hodnota zadaná, zobrazí se textový popisek Neohodnoceno. Jednotlivé položky se dělí dle důležitosti svého typu na čtyři druhy. Na základě těchto druhů se dále rozlišuje, jestli a jak se v prostředí zvýrazní. Nejvýraznější jsou položky typu zápočet a výsledné hodnocení, které mají barevný obrys, světle fialové pozadí a tučný text. Dále jsou položky typu Ústní zkouška a Zkouškový test, které mají světle fialové pozadí a tučný text. Posledním zvýrazněným typem je Semestrální test, který má jen tučný text. Ostatní typy položek patří do poslední kategorie, která nemá žádné zvýraznění.

Rozměry a vzhled jednotlivých položek jsou standardní dle Material De-



Obrázek 3.3: Seznam předmětů

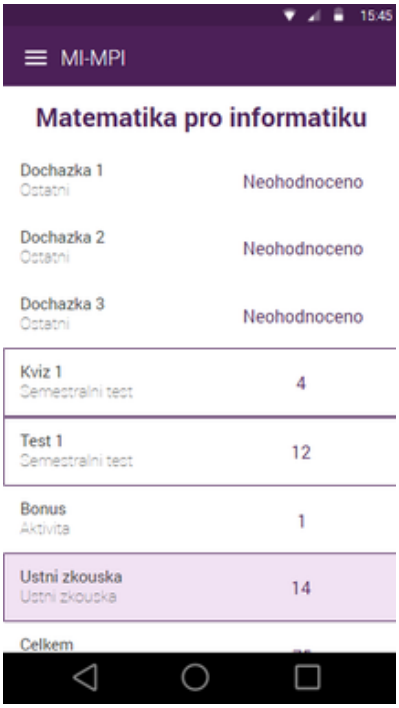
sign. Zvýrazňování některých typů není ve standardu pokryto, ale žádná pravidla neporušuje a je tedy v pořádku.

3.5.4 Zobrazení a zadávání klasifikace k předmětu z pohledu učitele

Zadávání hodnocení může dle požadavků probíhat ve dvou režimech, oba je možné vidět na obrázku 3.5. Buď je možné zadávat všechna definovaná hodnocení jednomu studentovi, nebo jednu vybranou klasifikaci všem studentům z vybrané skupiny. Pro přístup k jednotlivým režimům slouží tlačítka u položek v seznamu předmětů z pohledu učitele, která jsou popsána výše.

V režimu hodnocení dle vybraného studenta je v horní části obrazovky zobrazeno vyhledávací pole pro jeho zvolení. Po kliknutí do tohoto pole se zobrazí seznam studentů, ve kterém má uživatel možnost textově filtrovat. Po nalezení a zvolení studenta se načte a zobrazí jeho aktuální hodnocení s možností ho zadávat nebo upravovat.

Ve druhém, detailním režimu jsou zobrazena dvě vyhledávací pole popsané „vyhledat skupinu“ a „vyhledat klasifikaci“, první pro zvolení skupiny studentů a druhé pro zvolení konkrétní klasifikace. V obou je možnost textově filtrovat. Učitel musí vybrat obě hodnoty, aby se mu načetl seznam studentů ve skupině s aktuálním hodnocením pro vybranou klasifikaci, kterou je možné editovat.



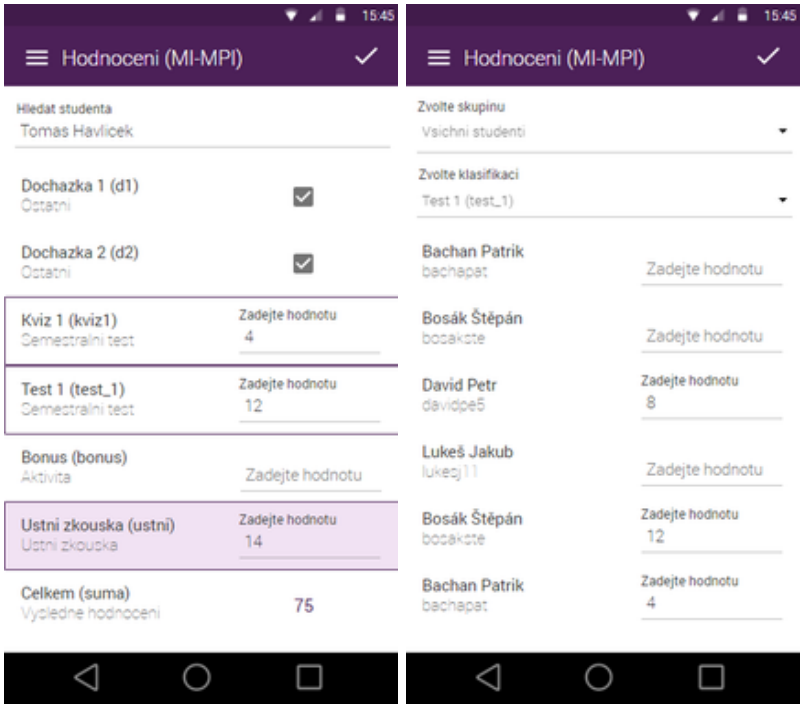
Obrázek 3.4: Zobrazení klasifikace předmětu z pohledu studenta

Položky klasifikace jsou třech druhů - textové, číselné nebo hodnota pravda/nepravda a podle toho se i zobrazují. Buď se jedná o jednoduché textové pole nebo číselné textové pole s tlačítky plus a minus anebo zaškrtávací políčko. Některé položky klasifikace jsou automaticky vypočítané, ostatní se zadávají ručně. Po provedení úprav a zapsání klasifikace je třeba, aby uživatel hodnocení uložil stisknutím tlačítka (v podobě znaku „fajfky“) v pravém horním rohu obrazovky na liště.

Seznam je opět vytvořen s ohledem na Material Design, stejně tak i vyhledávací pole a pole pro zadávání klasifikace. Tlačítko pro uložení je dobře viditelné a stále dostupné v horní liště.

3.5.5 Upozornění a notifikace

DDalší položkou v menu je přístupná obrazovka Upozornění, která, jak je vidět na obrázku 3.6 vlevo, obsahuje seznam notifikací daného uživatele. Nepřečtená upozornění jsou zobrazená nahoře a jsou zvýrazněná světle fialovým pozadím. Kliknutím na upozornění se označí jako přečtené a otevře detail předmětu, kterého se týká. V případě, že vznikne nové upozornění, přijde uživateli notifikace a zobrazí se v horní liště zvané Status Bar. Notifikace obsahuje titulek, popis a ikonu aplikace. Stejně jako v seznamu upozornění otevře kliknutí na notifikaci ve status baru detail předmětu.



Obrázek 3.5: Zobrazení a zadávání klasifikace k předmětu z pohledu učitele

Seznam upozornění i fungování a vzhled příchozích notifikací je navržen dle standardu Material Design.

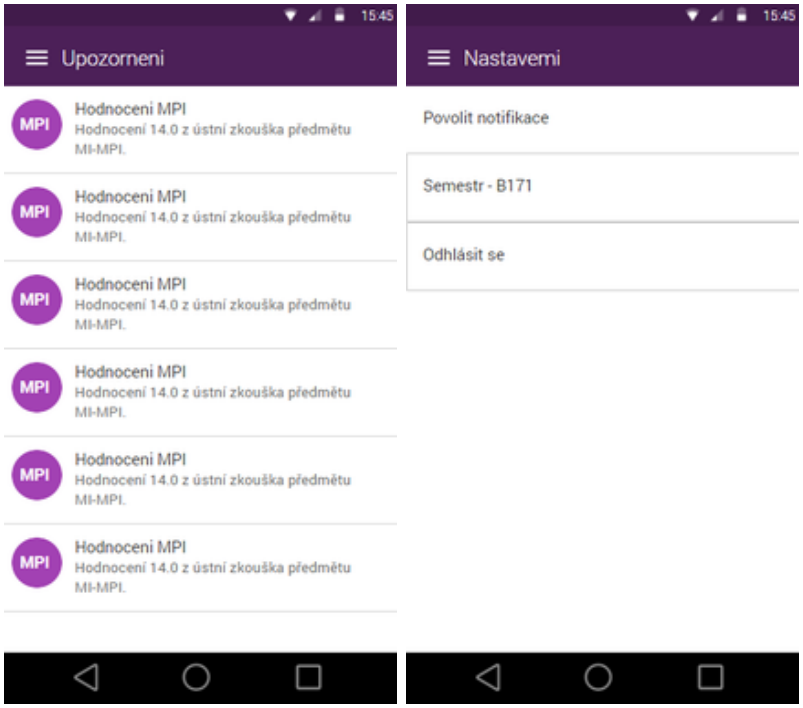
3.5.6 Nastavení

Obrazovka Nastavení (na obrázku 3.6 vpravo) je přehledná a obsahuje pouze tři možnosti. První z nich je zaškrtávací políčko, kde si uživatel zvolí, zda chce zasílání notifikací povolit nebo naopak zakázat. Před samotným povolením notifikací je vyžádán souhlas se zpracováním osobních údajů. Druhý řádek v nastavení otevírá uživateli možnost výběru příslušného semestru, jehož předměty včetně klasifikace požaduje v aplikaci zobrazovat. Výběr semestru se provádí zaškrtnutím radiobuttonu. Poslední řádek obrazovky Nastavení je pro odhlášení z aplikace, kde po zvolení proběhne informační dialog, zda chce uživatel tuto akci skutečně provést.

Obrazovka jednotlivé možnosti nastavení odpovídají standardu Material Design.

3.5.7 Graf obrazovek

Pro lepší názornost a pochopení návaznosti jednotlivých obrazovek aplikace jsem vytvořil graf 3.7. Ten přehledně zobrazuje, jak se dá v rámci aplikace



Obrázek 3.6: Notifikace a nastavení

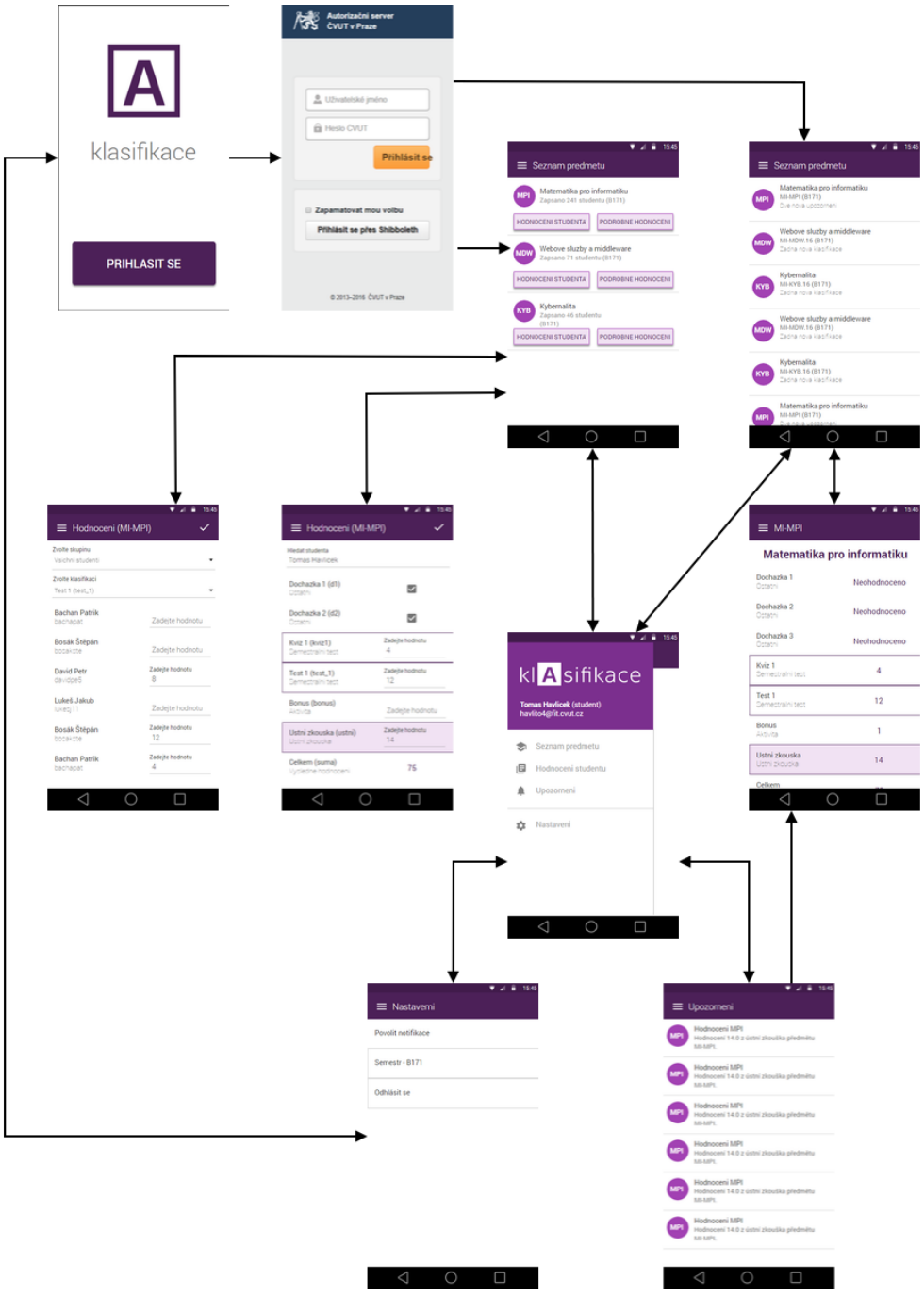
navigovat, jaké jsou zde funkcionality a možnosti. Počáteční bodem tohoto schématu je obrazovka pro přihlášení v levém horním rohu, na kterou navazují ukázky úvodních obrazovek dostupných po přihlášení a to z pohledu učitele a z pohledu studenta. Přibližně uprostřed obrázku uvádím základní menu, které má v aplikaci podobu draweru, je uživateli dostupné ze všech obrazovek první úrovně a dá se vyvolat kliknutím na ikonku tří čárek v levém horním rohu aplikace, nebo gestem přejetí prstu po obrazovce z levého okraje displeje směrem doprava.

3.6 Notifikační server

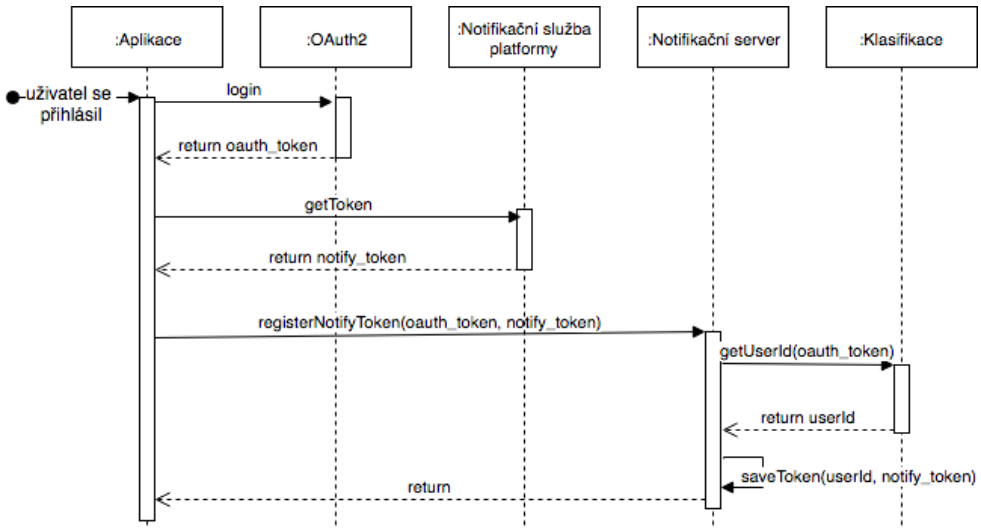
Následující text popisuje se věnuje notifikačního serveru. Nejprve je uveden abstraktní návrh komunikace a fungování, následně konkrétní popis komunikačního rozhraní, které bude server vystavovat a rozhraní, které bude využívat.

3.6.1 Registrace uživatele a notifikačního tokenu aplikace

Tato kapitola uvádí, jak bude probíhat registrace uživatele a jeho notifikačního tokenu na notifikační server. Přihlášení přes OAuth 2.0 a získání notifikačního tokenu z notifikační služby platformy je popsáno v kapitole OAuth 2.0 a v



Obrázek 3.7: Graf obrazovek



Obrázek 3.8: Registrace uživatele a notifikačního tokenu aplikace

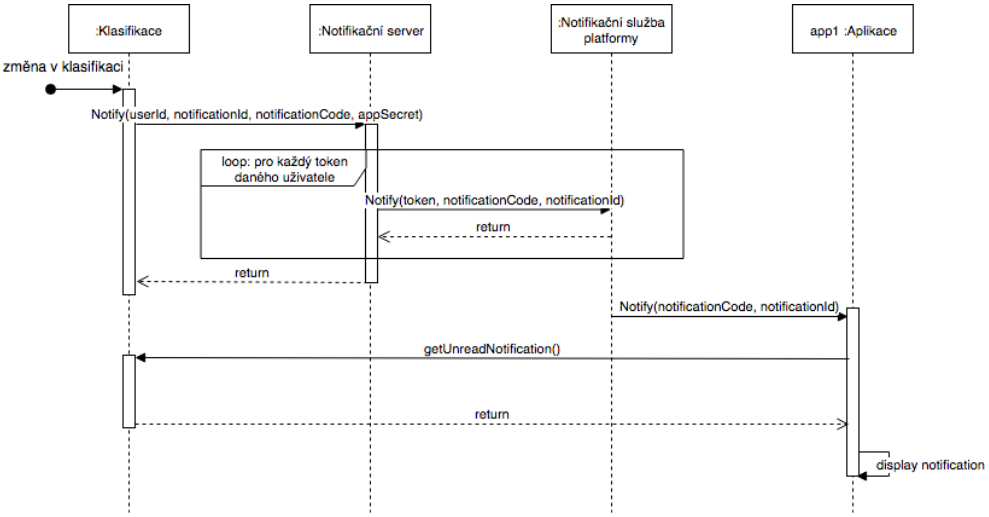
kapitole Notifikace v mobilních zařízeních. V této části se jim nevěnuji a používám je ve zjednodušené formě.

Uživatel se v aplikaci nejprve přihlásí pomocí OAuth 2.0, čímž získá svůj oauth token, dále si aplikace vyžádá notifikační token od notifikační služby své platformy. Následně se aplikace registruje k notifikačnímu serveru, kterému pošle oauth token pro ověření uživatele a notifikační token, který má být registrován pro zasílání notifikací. Notifikační server při zpracování tohoto požadavku získá userId z API aplikace Klasifikace. Vrácené userId je userId uživatele, kterému patří zasílaný token, čímž se celá operace autentifikuje. Notifikační server si získané userId a notifikační token uloží. Pro lepší pochopení jsem vytvořil sekvenční diagram 3.8 modelující celý proces.

3.6.2 Notifikování aplikace přes notifikační server

Při vzniku nové notifikace v aplikaci Klasifikace je informován notifikační server, kterému je doručen identifikátor uživatele userId, identifikátor konkrétní notifikace notificationId, informace o typu notifikace notificationCode, tajný klíč appSecret sloužící pro identifikaci a autorizaci aplikace, která notifikaci posílá. Notifikační server následně vygeneruje notifikaci pro všechny tokeny, které má uloženy pro daného uživatele a odešle je do notifikační služby konkrétní platformy. Notifikační služba platformy se poté postará o doručení notifikace do zařízení. Přehledně je proces znázorněn na sekvenčním diagramu 3.9.

Když aplikace obdrží notifikaci, vyčte si její konkrétní data z aplikace Klasifikace a zobrazí ji. Díky tomu, že má každá notifikace svůj identifikátor, je možné, pokud dojde k označení notifikace v systému Klasifikace jako přeč-



Obrázek 3.9: Notifikování aplikace přes notifikační server

tené, notifikaci skrýt. Využije se k tomu stejný postup jako při posílání nové notifikace s tím, že se změní typ notifikace notificationCode.

3.6.3 Návrh rozhraní notifikačního serveru

Pro fungování komunikace, tak jak je popsáno výše, bude mít notifikační server REST rozhraní.

- Registrace tokenu
  - POST /token
  - Hlavička požadavku:

    Authorization: Bearer oauth_token - autorizační token daného uživatele získaný přihlášením přes službu OAuth 2.0 ČVUT
  - Tělo požadavku (json objekt):

    notify_token - notifikační token získaný z notifikační služby dané platformy

    token_type - platforma Android nebo iOS
  - Návratové hodnoty:

    201 Created - pokud dojde k úspěšné registraci uživatele a tokenu

    400 Bad Request - pokud je požadavek nevalidní, např. neobsahuje správné parametry

    401 Unauthorized - pokud je zaslán nevalidní oauth_token
- Odstranění tokenu

  - DELETE /token
  - Hlavička požadavku:

    Authorization: Bearer oauth_token - autorizační token daného uživatele získaný přihlášením přes službu OAuth 2.0 ČVUT
  - Tělo požadavku (json objekt):

    notify_token - notifikační token, který má být odstraněn

    token_type - platforma Android nebo iOS
  - Návratové hodnoty:

    200 OK - pokud dojde k úspěšnému smazání notifikačního tokenu

    400 Bad Request - pokud je požadavek nevalidní, např. neobsahuje správné parametry

    401 Unauthorized - pokud je zaslán nevalidní oauth_token

    404 Not Found - pokud notifikační server nenalezne notifikační token, který má být odstraněn

- Odeslání notifikace
  - POST /notification
  - Hlavička požadavku:

    Authorization: Bearer app_secret - identifikátor aplikace odesílající notifikaci, sloužící pro ověření
  - Tělo požadavku (json objekt):

    notification_id - jedinečný identifikátor notifikace

    notification_type - typ notifikace, definováno mezi odesílatelem notifikace a mobilní aplikací

    user_id - identifikátor uživatele, kterému je notifikace určena
  - Návratové hodnoty:

    202 Accepted - pokud dojde ke správnému zpracování požadavku

    400 Bad Request - pokud je požadavek nevalidní, např. neobsahuje správné parametry

    401 Unauthorized - pokud požadavek obsahuje nevalidní app_secret

    404 Not Found - pokud notifikační server nezná uživatele s daným user_id

- Skrytí notifikace
  - DELETE /notification
  - Hlavička požadavku:

    Authorization: Bearer app_secret - identifikátor aplikace odesílající notifikaci, sloužící pro ověření

- Tělo požadavku (json objekt):

  notification_id - jedinečný identifikátor notifikace

  user_id - identifikátor uživatele, kterému je má být notifikace skryta

- Návratové hodnoty:

  202 Accepted - pokud dojde ke správnému zpracování požadavku

  400 Bad Request - pokud je požadavek nevalidní, např. neobsahuje správné parametry

  401 Unauthorized - pokud požadavek obsahuje nevalidní app_secret

  404 Not Found - pokud notifikační server nezná uživatele s daným user_id

3.6.4 Rozhraní využívané notifikačním serverem

- Získání ID uživatele
  - GET /public/user-id
  - Hlavička požadavku:

    Authorization: Baerer oauth_token - autorizační OAuth 2.0 token daného uživatele

  - Odpověd:

    user_id - id daného uživatele

  - Návratové hodnoty:

    200 OK - pokud dojde k úspěšné registraci uživatele a tokenu

    401 Unauthorized - pokud požadavek obsahuje nevalidní oauth_token

    404 Not Found

- Odeslání notifikace Firebase
  - POST https://fcm.googleapis.com/fcm/send
  - Hlavička požadavku:

    Authorization: key=server_key - autorizační klíč serveru vygenerovaný přes Firebase konsoli

  - Tělo požadavku (json objekt):

    tělo obsahující json s notifikačním tokenem příjemce a samotná data, tedy typ a id notifikace

  - Návratové hodnoty:

    200 OK - pokud dojde ke zpracování požadavku, v těle odpovědi je v tomto případě informace o skutečném odeslání notifikace

    400 Bad Request - pokud požadavek obsahuje nevalidní tělo

401 Unauthorized - pokud požadavek neobsahuje správný server key

- Odeslání notifikace Apple Push Notification Service
  - POST /3/device/¡device-token¿
  - Hlavička požadavku:

    Authorization: Bearer token - autorizační token serveru
  - Parametry požadavku:

    device-token parametr url obsahuje notifikační token cílového zařízení

    tělo obsahující json s daty notifikace, tedy typ a id notifikace
  - Návratové hodnoty:

    200 OK - pokud dojde ke zpracování požadavku, v těle odpovědi je v tomto případě informace o skutečném odeslání notifikace

    400 Bad Request - pokud požadavek obsahuje nevalidní tělo

    403 Forbidden - pokud se požadavek neautentifikuje

Kapitola 4

Realizace a testování

V této kapitole se věnuji realizaci mobilní aplikace Klasifikace a notifikačního serveru. Popisuji, jak obě komponenty vznikaly, jaké jsem použil technologické nástroje a některé netriviální či jinak zajímavé implementační detaily.

Druhá část kapitoly se zabývá testováním aplikace a to jak automatickým testům pro ověření základní funkčnosti, tak uživatelskému testování použitelnosti a zhodnocením jeho výsledků.

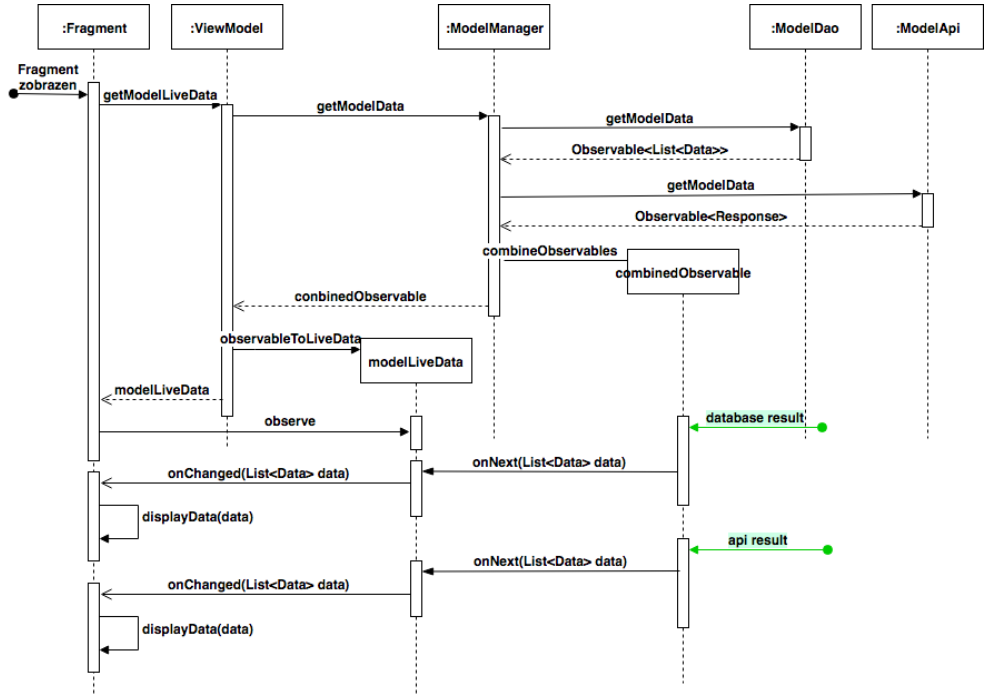
4.1 Společné implementační detaily mobilní aplikace Klasifikace

Aplikace je implementována jako tzn. Single Activity application, což znamená, že obsahuje pouze jednu aktivitu, ve které se pomocí FragmentManageru mění fragmenty. Jako základ pro tento implementační směr používám Andoroid App Skeleton, což je knihovna umožňující jednoduše vytvořit kostru aplikace. Na vývoji knihovny se sám podílím a jsem zvyklý s ní pracovat.

Jak již bylo popsáno z architektonického hlediska, používá aplikace architekturu MVVM, tedy Model, View, ViewModel. Model je reprezentován Manager třídami, se kterými přímo komunikuje View Model, a rozhraními, které obsahují anotované metody. Při kompilaci se vygeneruje implementace těchto rozhraní na základě daných anotací. Anotace a následnou logiku pro generování objektů komunikujících se sítí zajišťuje knihovna Retrofit2 a objekty využívající databázi aplikace knihovna Room.

Část View Model využívá komponenty z relativně nového frameworku Android Architecture Components. Jedná se o komponenty ViewModel a LiveData, které ulehčují práci s životním cyklem aktivit a fragmentů. Hlavně fragmenty, ale i další komponenty Android Frameworku, tvoří třetí část architekury View.

Obecný princip, jak jednotlivé součásti spolupracují, je znázorněn na diagramu 4.1. Zeleně jsou zvýrazněny výstupy z databáze a z API, které se vrátí



Obrázek 4.1: Obecný proces komunikace jednotlivých vrstev architektury aplikace

asynchronně a způsobí aktualizaci zobrazených dat ve fragmentu.

4.2 Přihlášení

Logika přihlášení využívá přihlašovací protokol OAuth 2.0, který je popsán ve stejnojmenné kapitole v části Návrh. Přihlášení je úspěšné, pokud se podaří získat OAuth 2.0 token a refresh token, následně stáhnout informace o uživateli z login-controlleru API Klasifikace. Stažené informace o uživateli se uloží do preferencí a seznam jeho předmětů do databáze. Po úspěšném provedení těchto kroků se uživateli zobrazí seznam předmětů. Pokud je v roli student, nebo student i učitel, zobrazí se seznam studovaných předmětů, pokud je v roli učitel, zobrazí se seznam předmětů, které může hodnotit. Ve speciálním případě, kdy uživatel nemá ani jednu roli, se zobrazí nastavení, kde si případně může zvolit jiný semestr, ve kterém nějaké předměty studoval nebo vyučoval.

4.3 Zobrazení seznamu předmětů

Zobrazení předmětů využívá výše popsaného principu a používají se informace o předmětech stažené při přihlášení. Při zobrazení předmětů je také proveden

pokus o jejich aktualizaci. Pro zobrazení jména předmětu a počtu studentů je nutné získat navíc detailní informace o předmětu z KosApi. Logika na této operaci však není závislá a pokud detail nelze získat, použije se místo něj kód předmětu.

Stejně se u seznamu předmětů studenta zobrazuje počet nových upozornění. I v tomto případě nezávisí zobrazení seznamu předmětů na úspěšném stažení dat o notifikacích.

4.4 Zadávání klasifikace

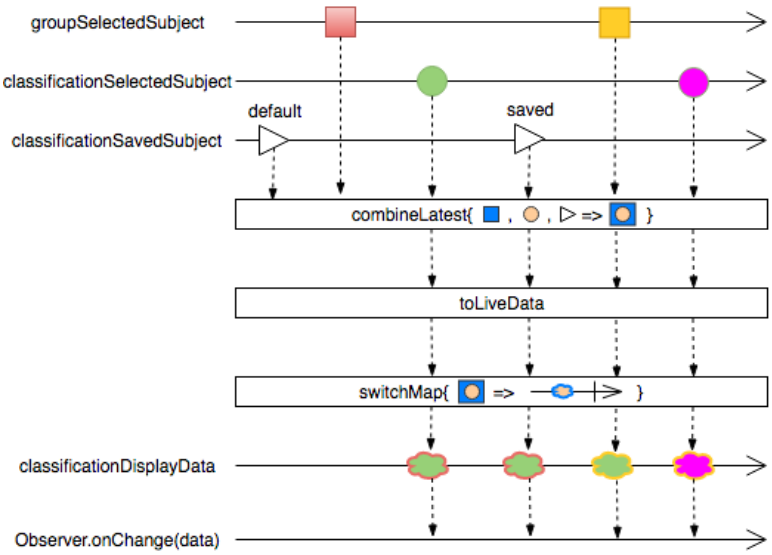
Zadávání klasifikace je umožněno ve dvou režimech, zadávání dle studenta a detailní zadávání. Funkcionalita obou fragmentů je podobná, a proto jsem pro sjednocení společných částí použil dědičnost. Fragmenty mají společného předka BaseClassifyFragment a view modely BaseClassifyViewModel. Společná je hlavně logika zadávání hodnoty a ukládání klasifikace. Udržování změněných hodnot využívá reaktivní principy, kdy změna hodnoty vyvolá akci, která je zachycena a změněná hodnota se zapamatuje. Ukládání se provádí za pomocí mapy, kde klíčem je dvojice student a položka klasifikace. Po kliknutí na tlačítko uložit a úspěšném uložení se tato mapa vymaže a zobrazená klasifikace se znovu stáhne.

Reaktivně je také naimplementována logika vyhledávání studentů, případně skupiny a položky klasifikace. V momentě, kdy uživatel klikne na vyhledanou položku, dojde k vyemitování této hodnoty a spuštění reaktivního řetězce událostí vedoucích ke stažení příslušných dat. Výsledek stahování notifikuje observer, který se postará o zobrazení dat. Jak tento řetězec funguje ve fragmentu pro detailní zadávání klasifikace, je vidět na obrázku 4.2.

4.5 Notifikace

Seznam s upozorněními se vždy stahuje z notification-controlleru z API a neukládá se do databáze. Z toho vyplývá, že bez připojení k internetu nelze seznam zobrazit.

Dále aplikace umožňuje přijímat push notifikace přes Firebase Cloud Messaging. Díky tomu je lze přijímat i když je aplikace na pozadí, nebo dokonce vypnutá. Příchozí notifikace aplikaci probudí a spustí službu, která se o přijetí stará. Jelikož data notifikací neobsahují konkrétní informace, ale pouze typ, stáhne spuštěná služba přesné znění notifikace z API. Pokud se stažení povede, zobrazí se spolu se všemi detaily, pokud ne, zobrazí se generické oznámení o novém upozornění.



Obrázek 4.2: Reaktivní řetězec pro vyhledání a zobrazení detailního klasifikování

4.6 Notifikační server

V rámci práce jsem implementoval jednoduchý notifikační server, který obsahuje navrženou logiku a rozhraní. Na rozdíl od mobilní aplikace nevyužívá notifikační server principy reaktivního programování, ale je implementován objektově. Základ aplikace tvoří framework Spring, který je využíván pro definování rozhraní, posílání požadavků a pro vkládání závislostí (dependency injection). Pro komunikaci s databází je využita knihovna Hibernate. O správu závislostí a celkové sestavení spustitelného balíčku s notifikačním serverem se stará nástroj Maven.

4.7 Uživatelské testování

Při uživatelském testování se ověřuje, jak aplikace funguje v rukách cílových uživatelů. Má ověřit funkčnost a použitelnost uživatelského rozhraní, případně najít nedostatky, které nemusí být na první pohled zřejmé. Uživatelům, kteří se testu účastní, jsou dány předem připravené scénáře, podle kterých provádějí v aplikaci dané činnosti. Testovací scénář popisuje, co má uživatel udělat, jaké prerekvizity jsou potřeba a jaký výsledek je očekáván. Průběh testu je zaznamenáván a je vytvořen protokol, který obsahuje vzniklé poznatky a další informace, které mohou být pro test relevantní.

Není nutné, aby se testování účastnilo velké množství testerů, většina chyb se projeví i při testování několika uživateli. Přesný počet závisí na typu aplikace, počtu jejích funkcí a možností a v neposlední řadě na rozpočtu.

Důraz je kladen především na podrobné zdokumentování průběhu a výsledku jednotlivých testů.

Mobilní aplikaci Klasifikace jsem tomuto typu testování podrobil, vytvořil jsem testovací scénáře a pro účast na testování jsem požádal 3 učitele a 4 studenty FIT ČVUT. Množství testerů v poměru k rozsahu funkcí aplikace lze označit jako dostatečné.

4.7.1 Testovací scénáře

Vzhledem k tomu, že je aplikace určena jak pro studenty, tak pro učitele, bylo nutné vytvořit scénáře pokrývající funkcionalitu obou rolí a při testování je oddělit. V práci však jsou testovací scénáře uvedeny do jedné přehledné tabulky 4.1, v níž jsou na jednotlivých řádcích uvedeny jejich popisy včetně vstupních prerekvizit a očekávaného výsledku. Celkem vzniklo 6 scénářů, z kterých jsou 2 pouze pro studenty, 3 pouze pro učitele a jeden společný pro obě role. Přesto, že jde o nízký počet, pokrývají scénáře hlavní funkcionalitu aplikace a navíc nezatíží oslovené dobrovolné testery časovou náročností.

4.7.2 Protokol z testování

Při každém testování vznikal protokol. Jak již je zmíněno v předchozím textu, bylo nutné připravit dva protokoly. Jeden pro učitele a druhý pro studenty. Nevyplněný protokol pro roli student je na obrázku 4.3, druhý protokol vypadá podobně a je přiložen v příloze.

Při testování vyplňoval uživatel v protokolu své jméno, souhlas s uvedením jména v této práci, jestli je pravidelným uživatelem OS Android a komentáře k výsledkům jednotlivých testů. Zároveň byl pořizován videozáznam pro pozdější podrobnější vyhodnocení testu. Po vykonání testu jsem spolu s uživatelem vyplnil komentář ke každému ze scénářů. Vyplněné testovací protokoly jsou v naskenované podobě na přiloženém CD.

4.7.3 Výsledky testování

Pro účely testování jsem oslovil 3 vyučující a 4 studenty FIT ČVUT. Z řad vyučujících jsem pro testování vybral Ing. Michala Valentu Ph.D., Ing. Tomáše Kalvodu Ph.D. a Ing. Jakuba Průšu. Ze studentů se jednalo o Leoše Glasera, Martina Osvalda, Bc. Lenku Stejskalovou a Vladimíra Vlka. V této části uvádím výsledky testování a jejich podrobné zhodnocení dle pořízeného videozáznamu a protokolu.

V den, kdy testování probíhalo, byla bohužel nasazena testovací instance systému Klasifikace, obsahující chybu při ukládání klasifikace. Tato chyba nebyla způsobená špatným fungováním mobilní aplikace, a proto považuji scénáře, ve kterých je úkolem uložit klasifikaci, za splněné, pokud uživatel našel a kliknul na tlačítko pro uložení.

Tabulka 4.1: Testovací scénáře

Číslo	Prerekvizity	Scénář	Očekávaný výsledek
1	přihlášený uživatel, role student, připojení k internetu	**Zobrazení klasifikace** Zobrazte si seznam studovaných předmětů, vyberte si jeden předmět a zobrazte svoji klasifikaci v něm.	Zobrazená klasifikace vybraného předmětu
2	přihlášený uživatel, role student, připojení k internetu	**Zobrazení upozornění** Zobrazte si seznam svých upozornění, zkontrolujte, zda máte nějaká nepřečtená, označte nepřečtená upozornění jako přečtená.	Zobrazen seznam upozornění, všechna jsou přečtená
3	přihlášený uživatel, připojení k internetu	**Změna semestru** Nastavte si semestr v aplikaci na B171 a poté si zobrazte seznam předmětů. Zkontrolujte, že seznam odpovídá zvolenému semestru.	Zobrazen seznam předmětů v semestru B171
4	přihlášený uživatel, role učitel, připojení k internetu	**Zadání klasifikace dle studenta** Otevřete si zadávání klasifikace dle studenta nějakého předmětu, vyhledejte si studenta, kterého chcete klasifikovat. Vyhledanému studentovi změňte alespoň dvě různá hodnocení. Změny uložte.	Změněná klasifikace daného studenta v systému Klasifikace
5	přihlášený uživatel, role učitel, připojení k internetu	**Zadání klasifikace detailní** Otevřete si detailní zadávání klasifikace u nějakého předmětu, zvolte skupinu a klasifikaci, kterou chcete měnit. Změňte hodnocení alespoň dvěma různým studentům. Změny uložte.	Změněná klasifikace vybraných studentů v systému Klasifikace
6	přihlášený uživatel, role učitel, připojení k internetu	**Zrušení změn v hodnocení** Otevřete si detailní zadávání klasifikace u nějakého předmětu, zvolte skupinu a klasifikaci, kterou chcete měnit. Změňte hodnocení alespoň dvěma různým studentům. Změny zrušte.	Hodnoty klasifikace se v systému nezměnily

Protokol uživatelského testování

Celé jméno:

Souhlasím s uvedením svého jména pro účely diplomové práce Mobilní aplikace - Klasifikace?
ANO / NE

OS Android pravidelně používám (např. ve svém telefonu, tabletu)?
ANO / NE ve verzi:

Prerekvizity

- připojení k internetu
- přihlášen jako student

Testovací scénáře

číslo	Scénář	Očekávaný výsledek
1	**Zobrazení klasifikace** Zobrazte si seznam studovaných předmětů, vyberte si jeden předmět a zobrazte svoji klasifikaci v něm.	Zobrazená klasifikace vybraného předmětu
2	**Zobrazení upozornění** Zobrazte si seznam svých upozornění, zkontrolujte zda máte nějaká nepřečtená, označte nepřečtená upozornění jako přečtená.	Zobrazen seznam upozornění, všechna jsou přečtená
3	**Změna semestru** Nastavte si semestr aplikace na B171 a poté si zobrazte seznam předmětů. Zkontrolujte, že seznam odpovídá zvolenému semestru.	Zobrazen seznam Vašich předmětů v semestru B171

Vyhodnocení

číslo scénáře	Úspěšně dokončen?	Komentář
1		
2		
3		

Obrázek 4.3: Protokol uživatelského testování - student

4.7.3.1 Ing. Michala Valentu Ph.D.

V aktuálním semestru nemá žádné předměty, ve kterých by se systém klasifikace používal. Proto testování začalo scénářem 4. a výběrem semestru B171, ve kterém uživatel takový předmět měl a mohl pokračovat v plnění scénářů 1-3.

Tester úspěšně dokončil všechny scénáře a jednoduše nalezl potřebné ovládací prvky. Při provádění úkolu č. 4 si nicméně nebyl jistý, zda změna semestru již proběhla. Při testování se dále projevil problém s otevřenou klávesnicí i na obrazovkách, kde není možné zadávat text.

4.7.3.2 Ing. Tomáš Kalvoda Ph.D.

Uživatel úspěšně dokončil všechny testovací scénáře, při testování nedošlo k žádným problémům, avšak vyhledávání ve velkém množství studentů v předmětu bylo pomalejší. Při detailním zadávání klasifikace uživatel navrhl přidat možnost zobrazit celou klasifikaci u vybrané skupiny studentů, ne pouze jeden sloupec. Ve výsledku hodnotil aplikaci kladně.

4.7.3.3 Ing. Jakub Průša

Tester začínal scénářem číslo 4, protože v aktuálním semestru nemá žádné předměty, které by mohl hodnotit. Změnu semestru provedl úspěšně, zobrazil seznam předmětů pro hodnocení a rozkliknul hodnocení dle studenta. Po vyhledání studenta změnil hodnocení několika položek. Po chvíli našel tlačítko pro uložení a klasifikaci uložil. Stejně úspěšně pokračovalo plnění dalších dvou scénářů, kdy uživatel správně našel tlačítko zahodit.

4.7.3.4 Leoš Glaser

Uživatel splnil všechny 3 testovací scénáře, bohužel v současné době nestuduje žádný předmět, kde by se systém Klasifikace používal. Výsledek zobrazení detailu byl však korektní, vypsala se hláška \Není co zobrazit". Ke zobrazení detailu s vyplněnou klasifikací nakonec také došlo a to po změně semestru. Při zobrazení notifikací uživatel po krátkém váhání našel tlačítko pro označení všech jako přečtené.

4.7.3.5 Martin Osvald

Tester provedl úspěšně všechny testovací scénáře, zobrazení seznamu předmětů a detailu zvládl rychle, seznam upozornění rychle objevil, ale snažil se označovat notifikace jako přečtené jednotlivě, nakonec našel tlačítko pro označení všech najednou. Při třetím vykonávaném scénáři uživatel navrhuje dát možnost zvolení semestru do menu, nebo na nějaké jiné lépe dostupné místo.

4.7.3.6 Bc. Lenka Stejskalová

Uživatelka kompletně provedla dva ze tří testovacích scénářů, u zobrazení notifikací nenašla možnost označit všechny jako přečtené a přesunula se k dalšímu úkolu. Zobrazení seznamu předmětů i změna semestru proběhla bez váhání. Testerce se aplikace líbila a i přes to, že pravidelně nepoužívá systém Android, se jí hladce podařilo vykonat většinu požadovaných úkonů.

4.7.3.7 Vladimír Vlk

Tento tester zvládl velmi rychle a úspěšně všechny testovací scénáře, nevyskytl se žádný problém s uživatelským prostředím aplikace ani s jejím fungováním. Díky tomu, že aplikace využívá standardů pro tvorbu uživatelského rozhraní v systému Android a tester tento systém pravidelně používá, bylo pro něj provádění scénářů jednoduché.

4.7.4 Shrnutí

Všichni oslovení testeři viděli mobilní aplikaci Klasifikace poprvé, dříve s ní nikdy nepracovali. I přes tento fakt prošli všechny připravené scénáře úspěšně, bez větších problémů a v relativně krátké době ovládání pochopili, což je vidět v tabulkách doby trvání vykonávání jednotlivých scénářů 4.2 a 4.3. Dokonce ani uživatelé, kteří pravidelně nepoužívají operační systém Android, neměli s ovládáním problém. Z toho vyplývá, že uživatelské prostředí aplikace je dobře použitelné, funkční a intuitivní.

Subjektivně hodnotili všichni oslovení testeři aplikaci kladně, přišla jim jednoduchá, přehledná a dobře se ovládala a většinou ji označovali za pěknou či hezkou.

Na druhou stranu testování odhalilo několik drobných nedostatků. Prvním z nich je chybějící hláška či potvrzení při změně semestru. Někteří uživatelé tak po zvolení semestru čekali na stránce s nastavením a nevěděli, jestli změna proběhla. Druhým odhaleným nedostatkem bylo zobrazení klávesnice na obrazovkách, které neumožňují zadávání textu. Uživatel má sice možnost s pomocí tlačítka zpět klávesnici skrýt, ale jedná se o nepotřebné kliknutí navíc. Oba nedostatky byly po ukončeném testování opraveny a doplněny.

Dále jsem od uživatelů dostal jako zpětnou vazbu i několik návrhů na vylepšení. Prvním z nich je přesunutí možnosti zvolení semestru na nějaké lépe přístupné místo. Tento návrh implementován nebude, protože ani jeden z uživatelů neměl problém tuto možnost najít a semestr změnit a zároveň je umístění volby v nastavení logické. Druhým návrhem je pro učitele možnost zobrazit si tabulku s kompletním hodnocením vybraného předmětu, tedy všichni studenti a všechna hodnocení. Tento návrh v rámci diplomové práce realizován není, nicméně je předpoklad, že v budoucnu se tato možnost určitě doplní.

Tabulka 4.2: Doba testování - studenti

Studenti	Scénář č. 1	Scénář č. 2	Scénář č. 3
Leoš Glaser	0:35	0:25	0:22
Martin Osvald	0:32	1:22	0:16
Lenka Stejskalová	0:13	0:22	0:25
Vladimír Vlk	0:15	0:13	0:29

Tabulka 4.3: Doba testování - učitelé

Učitelé	Scénář č. 3	Scénář č. 4	Scénář č. 5	Scénář č. 6
Michal Valenta	1:01	1:05	1:35	0:52
Tomáš Kalvoda	0:53	0:56	0:47	0:31
Jakub Průša	0:37	0:41	0:42	0:28

4.8 Automatické testování

Automatizované testování je jedním ze standardních nástrojů používaných při vývoji software. Přináší výrazné úspory prostředků a samozřejmě zvýšení kvality výsledného produktu. Automatizovat lze však nejen samotnou exekuci vybraných manuálních testů, ale také části nebo dokonce celý proces testování softwaru. Některé nástroje, které porovnávají výsledky provedených testů, instalují a konfigurují aplikaci, nebo vytváří z analýzy testovací případy. [25]

V rámci práce na mobilní aplikaci Klasifikace jsem implementoval jednotkové testy pro ověření funkčnosti jednotlivých metod a testy uživatelského rozhraní, které mají ověřit základní fungovaní uživatelského prostředí aplikace jako celku.

4.8.1 Jednotkové testy

V rámci aplikace se nenachází velké množství funkcí, které by vykonávaly komplexní logiku a bylo je nutné pokrýt testy, nicméně metody, které nějakou logiku obsahují, jsem jednotkovými testy pokryl. Dále jsem jednotkové testy použil pro ověřování správného mapování odpovědí z používaných API na entity v aplikaci.

4.8.2 Automatické testy uživatelského rozhraní

Pro automatické testování uživatelského rozhraní aplikace se na platformě Android používá knihovna Espresso, která obsahuje statické metody pro interakci s uživatelským prostředím a jeho ověřování. Klíčové komponenty Espresso frameworku jsou samotné statické metody ve třídě Espresso, ViewMatchers, které umožňují lokalizovat potřebný prvek UI podle různých kritérií, ViewActions, které umožňují provádět různé akce (např. kliknutí nebo psaní textu)

a ViewAssertions, které se používají pro ověření správného stavu nějaké zobrazené komponenty. [26]

V aplikaci jsem automatické testování uživatelského rozhraní použil pro hlavní funkce aplikace kromě přihlášení, které vyžaduje interakci reálného uživatele. Testuji tedy zobrazení seznamu předmětů z pohledu studenta i z pohledu učitele, detail klasifikace, zadávání klasifikace, uložení nebo zrušení změn, seznam notifikací a nastavení semestru.

Závěr

Cílem mé diplomové práce bylo vytvořit mobilní aplikaci Klasifikace pro zařízení se systémem Android, která bude komunikovat s existujícím informačním systémem Klasifikace, který se již na Fakultě informačních technologií ČVUT využívá. Doposud ale chyběla mobilní aplikace, která by byla dobře optimalizovaná pro použití v mobilním telefonu a díky níž by uživatelé z řad studentů či učitelů měli k informacím týkajícím se hodnocení a klasifikace rychlý a snazší přístup.

Podařilo se vytvořit mobilní aplikaci, která nabízí rychlé, intuitivní a funkční rozhraní pro přístup ke klasifikaci. Aplikace má široké užití, učitelům umožňuje vkládat hodnocení studentům a uživatelům z řad studentů prohlížet hodnocení za celou dobu svého studia a zároveň přijímat notifikace, které upozorňují na nově zapsanou známku či změnu v klasifikaci. Tato funkcionalita bude studenty jistě hojně využívána a oceňována. Pro posílání notifikací ze systému Klasifikace do mobilní aplikace se využívá notifikační server. Analyzoval jsem řadu možných platforem a variant zajišťujících funkci notifikačního serveru a nakonec jsem zvolil možnost vlastního návrhu a implementace, které jsem provedl. Výsledný notifikační server je dělaný na míru systému Klasifikace, má jednoduché rozhraní a umožňuje jak odesílání, tak skrývání notifikací.

V práci je podrobně navrženo a popsáno uživatelské rozhraní. Tyto kapitoly jsou případně využitelné i jako uživatelský manuál pro ty, kteří mají zájem se s aplikací seznámit podrobně a využívat všechny její možnosti a funkcionality. Uživatelské prostředí je pro uživatele přívětivé a intuitivní, neboť bylo navrženo v souladu s designovým principem Material Design a drží se standardů pro tvorbu aplikací pro operační systém Android.

Jelikož se v aplikaci zpracovávají osobní údaje uživatelů, je část diplomové práce věnována problematice nové legislativy EU týkající se ochrany osobních údajů - GDPR. Jsou zde popsány obecné principy nařízení a provedena jejich implementace v aplikaci Klasifikace a notifikačním serveru. Některé získané informace o uživateli si aplikace ukládá do vnitřní paměti zařízení. Ty se

však můžou jednoduše smazat odhlášením, smazáním dat nebo kompletním odinstalováním aplikace. Pro zachování principu transparentnosti se uživateli po přihlášení zobrazuje dialog, který toto chování popisuje.

Každá mobilní aplikace by měla být podrobena uživatelskému testování, stejně tak i vytvořená mobilní aplikace Klasifikace prošla testováním zaměřeným na použitelnost. Do něj bylo zapojeno několik studentů a učitelů Fakulty informačních technologií ČVUT. Výsledkem testování bylo zjištění, že aplikace je plně funkční a nevyskytují se v ní žádné známé chyby. Na základě výsledků testování mohu označit grafické rozhraní jako kvalitní a vysoce použitelné. Pro zajištění včasného odhalení chyb při implementaci, jsem pokryl složitější funkce jednotkovými testy a vytvořil jsem automatizované testy uživatelského rozhraní.

V práci jsem také provedl porovnání mobilní aplikace Klasifikace z různých hledisek s jinými obdobnými programy a je možné konstatovat, že v tomto srovnání lze aplikaci považovat za konkurenceschopný produkt, který bude mít své využití. V koordinaci s týmem vyvíjející webový systém Klasifikace je naplánováno aplikaci v brzké době vydat a průběžně doplňovat o nové funkce. Jednou z nich by mohlo být zobrazení tabulky více položek klasifikace pro učitele, což byl i jeden z návrhů vzešlých z uživatelského testování. V plánu je také možnost vnořování klasifikace, kdy bude mít jedna položka několik podpoložek. Do budoucna by se dala aplikace ještě doplnit zobrazením detailních informací o studovaných předmětech, kterými jsou například počet kreditů, podmínky hodnocení nebo vyučující.

Literatura

[1] GLOVER, A.: *Build a RESTful Web service [online]*. IBM Corp., Červenec 2008, [cit. 2018-02-25]. Dostupné z: https://www.ibm.com/developerworks/java/tutorials/j-rest/j-rest.html

[2] VITVAR, T. s.: Service Concepts and Technologies [online prezentace]. Leden 2018, [cit. 2018-02-25]. Dostupné z: http://mdw.vitvar.com/lecture7.html#/39/v1

[3] VITVAR, T. s.: Service Concepts and Technologies [online prezentace]. Leden 2018, [cit. 2018-02-25]. Dostupné z: http://mdw.vitvar.com/lecture7.html#/40/v1

[4] PARECKI, A.: *OAuth 2.0 [online]*. Březen 2018, [cit. 2018-03-14]. Dostupné z: https://oauth.net/2/

[5] PARECKI, A.: *Authorization Code Request [online]*. Okta, Inc., 2018, [cit. 2018-03-14]. Dostupné z: https://www.oauth.com/oauth2-servers/access-tokens/authorization-code-request/

[6] PARECKI, A.: *OAuth 2.0 Implicit [online]*. 2018, [cit. 2018-03-14]. Dostupné z: https://oauth.net/2/grant-types/implicit/

[7] PARECKI, A.: *OAuth 2.0 Password Grant [online]*. 2018, [cit. 2018-03-14]. Dostupné z: https://oauth.net/2/grant-types/password/

[8] PARECKI, A.: *OAuth 2.0 Client Credentials Grant [online]*. 2018, [cit. 2018-03-14]. Dostupné z: https://oauth.net/2/grant-types/client-credentials/

[9] PARECKI, A.: *OAuth for Browserless and Input-Constrained Devices [online]*. Okta, Inc., 2018, [cit. 2018-03-14]. Dostupné z: https://www.oauth.com/oauth2-servers/device-flow/

[10] PARECKI, A.: *OAuth 2.0 Refresh Token [online]*. 2018, [cit. 2018-03-14]. Dostupné z: `https://oauth.net/2/grant-types/refresh-token/`

[11] CALLAHAM, J.: *The history of Android OS: its name, origin and more [online]*. Android Authority, 2017, [cit. 2018-02-25]. Dostupné z: `https://www.androidauthority.com/history-android-os-name-789433/`

[12] Google LLC: *Distribution dashboard [online]*. 2018, [cit. 2018-05-05]. Dostupné z: `https://developer.android.com/about/dashboards/`

[13] Google LLC: *Design for Android [online]*. Duben 2018, [cit. 2018-05-03]. Dostupné z: `https://developer.android.com/design/`

[14] TSVETINOV, N.: *Learning Reactive Programming with Java 8*. Birmingham: Packt Publishing, první vydání, 2015, ISBN 978-1-78528-872-2.

[15] Vopařil, Vojtěch: *Android mobilní aplikace s GPS trekingem*. Diplomová práce, Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

[16] Apple Inc: *APNs Overview [online]*. Listopad 2017, [cit. 2018-03-16]. Dostupné z: `https://developer.apple.com/library/content/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/APNSOverview.html`

[17] Google LLC: *About Firebase Cloud Messaging Server [online]*. Duben 2017, [cit. 2018-04-08]. Dostupné z: `https://firebase.google.com/docs/cloud-messaging/server`

[18] Úřad pro ochranu osobních údajů: *Základní příručka [online]*. 2013, [cit. 2018-04-24]. Dostupné z: `https://www.uoou.cz/zakladni-prirucka/ds-4744/p1=4744`

[19] ŠKORNIČKOVÁ, E.: *Co je GDPR a jak bude aplikováno v Česku [online]*. [cit. 2018-04-24]. Dostupné z: `https://www.gdpr.cz/gdpr/co-je-gdpr/`

[20] KMOŠKOVÁ, J.; KORNEL, M.: *Co je, co není a co bude osobní údaj podle GDPR [online]*. Květen 2017, [cit. 2018-04-24]. Dostupné z: `http://www.fbadvokati.cz/cs/clanky/541-co-je-co-neni-a-co-bude-osobni-udaj-podle-gdpr`

[21] ŠKORNIČKOVÁ, E.: *Jaké povinnosti ukládá GDPR institucím a firmám [online]*. [cit. 2018-04-24]. Dostupné z: `https://www.gdpr.cz/gdpr/povinnosti/`

[22] HORÁČKOVÁ, J.: *GDPR [prezentace]*. Leden 2018, [cit. 2018-04-24].

[23] ŠKORNIČKOVÁ, E.: *Zásady Obecného nařízení [online]*. [cit. 2018-04-24]. Dostupné z: https://www.gdpr.cz/gdpr/heslo/zasady-obecneho-narizeni/

[24] BALÁK, Z. e.: *FIT Classification REST API [online]*. 2018, [cit. 2018-05-01]. Dostupné z: https://grades.fit.cvut.cz/api/v1/private/documentation

[25] testovanisoftwaru.cz: *Automatizované testování [online]*. [cit. 2018-05-04]. Dostupné z: http://testovanisoftwaru.cz/automatizovane-testovani/

[26] Google LLC: *Espresso basics [online]*. Duben 2018, [cit. 2018-05-04]. Dostupné z: https://developer.android.com/training/testing/espresso/basics

Příloha **A**

Seznam použitých zkratek

API Application Programming Interface

HTTP Hypertextový přenosový protokol

OS Operační systém

REST Representational state transfer

SDK Software Development Kit

UI User interface

Příloha **B**

Obsah přiloženého CD

- `DP_Havlicek_tomas_2018.pdf` ..... tento dokument ve formátu PDF
- `readme.txt` ..... stručný popis obsahu CD
- `build`
  - `Classification`
    - `instalacni_prirucka.pdf` ..... instalační příručka aplikace
    - `Classification-release-1.0.0.apk` . instalační balíček aplikace
  - `notifrication_server`
    - `instalacni_prirucka.pdf` ..... instalační příručka serveru
    - `notification_server-1.0.jar` ..... spustitelný jar notif. serveru
- `src`
  - `Classification` ..... zdrojové kódy mobilní aplikace
  - `notifrication_server` ..... zdrojové kódy notifikačního serveru
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `testing` ..... protokoly a záznamy uživatelského testování