Jihočeská univerzita v Českých Budějovicích

Přírodovědecká fakulta

Webová aplikace pro distribuci mobilních aplikací

Bakalářská práce

Jaroslav Machovec

Vedoucí práce: Ing. Vojtěch Kačírek – Agentes IT s.r.o

Konzultant: Ing. Jiří Jelínek CSc.

České Budějovice 2022

Bibliografické údaje:

Machovec, J., 2022: Webová aplikace pro distribuci mobilních aplikací [Web application for distribution of mobile applications, Bc. Thesis, in Czech] – 52 p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic

Anotace:

The thesis deals with the process of creating a web application for distributing both test and production versions of mobile applications. The first part consists of research and a comparison of the existing solutions. The second part is focused on the design of the application, defining functionality and implementation. During the last part the application is deployed and tested by real users. After the users‘ feedback, the application is modified.

Prohlášení:

Prohlašuji, že jsem autorem této kvalifikační práce a že jsem ji vypracoval pouze s použitím pramenů a literatury uvedených v seznamu použitých zdrojů.

V Českých Budějovicích dne _____ _____

podpis

Poděkování

Tímto bych rád poděkoval Ing. Vojtěchu Kačírkovi za jeho podporu a cenné rady v průběhu tvorby bakalářské práce. Dále bych rád poděkoval Ing. Jiřímu Jelínkovi, CSc. zejména za pomoc při definování rozsahu práce a za zastřešení práce pod Přírodovědeckou fakultu Jihočeské univerzity. Dále všem kolegům z firmy Agentes IT, kteří mi během vývoje aplikace přispívali cennými radami, a také uživatelům, kteří se zúčastnili dotazníkového šetření. Nakonec bych rád poděkoval rodině a kamarádům za jejich podporu.

Obsah

Seznam zkratek

ABB – Android App Bundle

EMM – Enterprise Mobility Management

MDM – Mobile Device Management

MVC – Model, View, Controller

OS – Operační systém

SDK – Software Development Kit

CI/CD - Continuous Integration/Continuous Delivery

1 Úvod

Chytré telefony se v posledních letech staly nedílnou součástí lidského života. Díky svým rozměrům je může každý neustále nosit při sobě. Zdaleka již však neslouží pouze k telefonování a posílání SMS zpráv. Je možné na nich nakupovat na internetu, odesílat platby, komunikovat na sociálních sítích, měřit počet nachozených kilometrů či sledovat výsledky sportovních utkání z celého světa. Stále více firem také používá mobilní telefony k elektronizaci svých procesů. Aplikace, které toto všechno umožní, však musí být do telefonu nejprve staženy a nainstalovány.

Drtivá většina mobilních zařízení využívá operační systém Android nebo iOS. V prosinci roku 2021 společně dosáhli až 99% podílu na světovém trhu. [1] Mobilní aplikace pro veřejnost jsou distribuovány především přes jejich platformy Google Play a App Store. Distribuce testovacích verzí mobilních aplikací tímto způsobem má však některé nedostatky:

- Nahrané verze mnohdy **podléhají kontrole**, která může trvat v řádu dní. To nám znemožňuje aplikace obratně testovat a při vývoji tak vznikají zbytečné prodlevy.
- Google Play navíc neumožňuje **zobrazit historii verzí** a kteroukoliv si nainstalovat, což se může při testování také velice hodit.
- **Výhradní distribuce aplikací třetím stranám** přes Google Play a App Store **vyžaduje**, aby koncová firma používala **nástroj MDM nebo EMM** pro správu zařízení svých zaměstnanců. Některé firmy však tyto nástroje nepoužívají a ani používat nechtějí (např. kvůli vysokým licenčním poplatkům).

K těmto účelům je tedy potřeba najít jiný způsob distribuce. Vhodné řešení by mělo umožňovat distribuci jednotlivých verzí aplikací pro platformu Android i iOS a správu jejich metadat. Uživatel by měl mít možnost si zobrazit historii těchto verzí a každá z nich by měla být po nahrání ihned k dispozici ke stažení. Současně by také mělo obsahovat systém přístupových práv a umožnit zjištění zpětné vazby od zákazníka. Řešení by mělo být možné využít pro testovací verze, ale i pro distribuci produkčních verzí určených třetím stranám bez nutnosti použít nástroj MDM či EMM.

Řešení, která jsou volně přístupná zdarma a dala by se využít, ale výše uvedená kritéria nesplňují. Placená řešení, která by svou funkcionalitou vyhovovala, mohou však znamenat pro menší firmy zbytečnou finanční zátěž, kterou si nemohou dovolit. Proto je potřeba vyvinout řešení nové.

1.1 Cíle práce

1. Analýza již existujících aplikací, technologií a řešení pro distribuci mobilních aplikací, stanovení kritérií pro porovnávání a jejich vah.

2. Definice požadované funkcionality na základě výše provedené analýzy a požadavků firmy Agentes IT. Specifikace případů užití (use cases). Aplikace bude používána jak pro platformu Android, tak iOS.

3. Návrh a vytvoření řešení pro distribuci mobilních aplikací vyvinutých firmou Agentes IT jejím zákazníkům. Bude sloužit k distribuci testovacích i produkčních verzí aplikací. Do aplikace budou nahrávány instalační soubory jednotlivých verzí mobilních aplikací a budou spravována jejich metadata včetně historie. Aplikace bude obsahovat systém přístupových práv a základní CRM nástroje včetně zajištění zpětné vazby od zákazníka.

4. Uvedení aplikace do reálného pilotního provozu, analýza získaných poznatků ze zpětné vazby i pomocí průzkumu spokojenosti. Reakce na ně formou úpravy aplikace.

2 Analýza existujících řešení

2.1 Technologická analýza

Abychom byli schopni porovnat existující řešení, stanovit si požadované funkcionality a navrhnout řešení nové, musíme se nejprve podívat na aplikace obou platforem z technologického hlediska a lépe této problematice porozumět. Zajímat nás bude zejména jakým způsobem je možné vyexportovat instalační soubor poté, co aplikaci naprogramujeme, a s jakými technologickými a licenčními omezeními je možné ho nainstalovat do koncového zařízení.

2.1.1 Android

U Androidu máme dva typy souborů, které můžeme po naprogramování aplikace vyexportovat. Soubor APK (přípona .apk) a nebo soubor AAB - Android App Bundle (přípona .aab).

Soubor **APK** je klasický instalační soubor aplikace, který si uživatel může po stažení do mobilu nainstalovat.

Soubor **AAB** je balíček, který obsahuje veškerý zkompilovaný kód a soubory aplikace. Google Play jej umí použít k vygenerování optimalizovaného souboru APK pro každé jednotlivé zařízení. To znamená, že při instalaci se stáhne pouze kód a soubory nutné k fungování aplikace na daném zařízení, což vede k mírnému snížení velikosti nainstalované aplikace oproti klasickému souboru APK. [2]

U obou typů souborů platí, že je nutné je podepsat klíčem. To umožňuje při instalaci aktualizací ověřit, že nová verze skutečně pochází od stejného autora a nebyla nijak porušena.

Velikou výhodou u Androidu oproti iOS je větší svoboda. Soubor APK nemá žádné další požadavky na způsob distribuce. Stačí jej dostat k uživateli do zařízení a zahájit instalaci. Instalační soubor tedy můžeme například uživateli odeslat v příloze emailu nebo pomocí jiného komunikačního kanálu. Další možností je nahrát soubor APK na web a zpřístupnit uživatelům link ke stažení. Co se týče aplikačního balíčku AAB, tak ten lze nahrát na Google Play či jiné alternativní tržiště aplikací a distribuovat jej tak široké veřejnosti. [3]

Když si to shrneme, tak z technologického hlediska lze distribuovat aplikace pro Android velice jednoduše. Nejsou zde žádná omezení, která by nám bránila vytvořit si vlastní řešení pro distribuci těchto aplikací.

2.1.2 iOS

Distribuce aplikací s operačním systémem iOS je oproti Androidu složitější. Operační systém iOS je vyvíjen společností Apple. Jednou z klíčových hodnot, které tato společnost zastává, je bezpečnost. Nainstalování škodlivé aplikace do mobilního zařízení je nepochybně jednou z bezpečnostních hrozeb a Apple se tedy snaží mít nad distribucí aplikací co největší kontrolu, aby tuto hrozbu eliminoval.

Pokud se rozhodneme vyvíjet aplikace pro iOS, tak máme v zásadě na výběr ze dvou možností, a to stát se členem Apple Developer Programu nebo Apple Enterprise Developer Programu.

Apple Developer Program pokrývá většinu případů užití. Ročně stojí členství v tomto programu 99$. [4] S tímto programem můžeme distribuovat veřejné aplikace přes App Store, ale také lze publikovat aplikace určené pouze určité organizaci či skupině organizací pomocí Apple Business Manageru. K testování aplikací můžeme využít TestFlight. Omezenému okruhu uživatelů je také možné distribuovat aplikace i mimo App Store.

Apple Developer Enterprise Program umožňuje velkým organizacím vyvíjet a distribuovat interní aplikace mezi své zaměstnance. Členství v tomto programu stojí ročně 299$. Zakoupit však lze pouze ve výjimečných případech, a to, když se možnosti distribuce v rámci Apple Developer Programu ukážou jako nedostačující. Organizace navíc musí splňovat hned několik kritérií. Splnit takováto kritéria už není úplně jednoduché. [5]

- Být právnická osoba.
- Mít více než 100 zaměstnanců.
- Využívat program pouze k vývoji interních aplikací a tyto aplikace bezpečně a soukromě distribuovat pouze mezi své zaměstnance.
- Mít zavedený systém, který je schopný zajistit, že si aplikaci mohou stáhnout pouze zaměstnanci.

- Dále také musí úspěšně absolvovat ověřovací pohovor a následně spolupracovat s Applem v rámci procesu průběžného hodnocení.

Pokud se rozhodneme vydat novou verzi naší aplikace, tak musíme vytvořit takzvaný archiv (*archive).* **Archiv** je balíček, který obsahuje naší aplikaci a je podepsán podpisovým certifikátem (*signing certificate*). Archiv také obsahuje profil (*provisioning profile*), který obsahuje některé důležité informace k distribuci (např. seznam zařízení, na které může být aplikace nainstalována).

Podepsání certifikátem umožní operačnímu systému na zařízení zkontrolovat, kdo aplikaci podepsal, a chrání ji před nežádoucími změnami. Pokud by někdo kód aplikace změnil, tak se podpis stane neplatným a aplikaci nepůjde na zařízení nainstalovat.

Poté, co je archiv úspěšně vytvořen, máme na výběr z několika metod distribuce: **App Store Connect**, **Ad Hoc**, **Enterprise**, **Development**. [6]

App Store Connect slouží k distribuci veřejných aplikací, které je možné nejprve stáhnout k otestování přes mobilní aplikaci TestFlight a poté jsou zveřejněny ke stažení přímo v App Store. [56] Taktéž se přes tuto metodu distribuují soukromé aplikace určené pro některou organizaci nebo skupinu organizací. Tato metoda je dostupná pouze členům Apple Developer Programu.

Ad Hoc metoda je určena pro distribuci na omezený počet zařízení. Konkrétně se jedná maximálně o 100 zařízení, jejichž UDID (Unique device ID) musí být přidáno na seznam zařízení v rámci našeho Developer Programu. Tento seznam je uložen v *provisioning profile*, který je součástí archivu aplikace. Při instalaci aplikace do zařízení je zkontrolováno, zda je v seznamu nalezeno UDID tohoto konkrétního zařízení, jinak instalace neproběhne. Pokud se tedy rozhodneme rozšířit distribuci i pro další uživatele, musíme přidat UDID jejich zařízení na seznam v našem Developer Programu a poté vytvořit nový archiv, který bude obsahovat *provisioning profile* s aktualizovaným seznamem zařízení. Tuto metodu distribuce mohou využívat členové Apple Developer Programu i Apple Developer Enterprise Programu. Využít ji lze tedy zejména k distribuci testovacích verzí, ale také k produkční distribuci pro omezený počet zařízení. [6]

Development slouží v průběhu vývoje aplikace k distribuci buildů pouze mezi členy vývojářského týmu. Tak jako u metody Ad Hoc archiv obsahuje profil se seznamem zařízení, na které může být aplikace nainstalována. Tentokrát je to však seznam zařízení členů vývojářského týmu. [6] Opět zde platí, že pokud přidáme nové zařízení v Developer Programu, musíme vytvořit nový archiv s profilem, který obsahuje aktualizovaný seznam, aby bylo možné aplikaci na nové zařízení nainstalovat. Také tuto metodu mohou využít členové obou placených programů.

Enterprise metoda slouží pouze pro distribuci aplikací v rámci Apple Developer Enterprise Programu, které už jsou otestovány a připraveny k užívání koncovými uživateli. [56] Počet zařízení zde není omezen. Musíme ale respektovat pravidla Apple Developer Enterprise Programu a distribuovat aplikace bezpečně pouze mezi své zaměstnance.

Pokud se na to podíváme z hlediska instalačních souborů, tak u metody **App Store Connect** se aplikace nahrávají z vývojového prostředí přímo na TestFlight či do App Store. Ve zbývajících metodách se nám vždy vyexportovalo několik souborů. Jeden z těchto souborů má příponu *.ipa* což je instalační soubor naší aplikace (tzv. IPA soubor). Dále je důležitý soubor *manifest.plist* (tzv. PLIST soubor), což je seznam vlastností ve formátu XML.

Oproti Androidu je distribuce instalačního souboru složitější. Pokud stáhneme do zařízení samotný IPA soubor, tak se instalace nespustí. Proto ho nemůžeme jen tak odeslat uživatelům emailem či obdobným způsobem. Existují zde dvě hlavní možnosti, jak tento IPA soubor zákazníkům oddistribuovat. První možností je k tomu využít **Mobile Device Management** (MDM), což je nástroj pro správu mobilních zařízení v rámci organizace, který přebírá pravomoc instalovat do mobilních zařízení aplikace. Druhou možností je tzv. distribuce **over-the-air** přes webové stránky. [7]

Pokud se rozhodneme distribuovat aplikaci **over-the-air**, je nutné nahrát soubor manifestu (PLIST) a soubor aplikace (IPA) na náš web, který je dostupný prostřednictvím protokolu HTTPS. [7]

Uživatelé, kteří si chtějí aplikaci nainstalovat, si kliknutím na speciální link stáhnou soubor PLIST z našich webových stránek (obr. 1). Stažení souboru PLIST, který v sobě obsahuje uloženou url k IPA souboru, automaticky spustí stažení a instalaci souboru aplikace (IPA). [7]

```
<a href="itms-services://?action=download-
manifest&url=https://example.com/manifest.plist">Instalovat
aplikaci</a>
```

Obrázek 1: Manifest – speciální link ke stažení IPA souboru (Zdroj: autor)

Cesta k IPA souboru se uložila do manifestu poté, co jsme ji vyplnili v průběhu exportování archivu. To může znamenat problém v situacích, kdy předem přesně nevíme, na které adrese bude IPA soubor přístupný. V takovém případě je nutné cestu v souboru PLIST upravit. Struktura souboru PLIST není nijak složitá a i v případě, že bychom jej neměli k dispozici vůbec, si jej můžeme vytvořit a vyplnit správnými údaji sami.

Abychom si to shrnuli, tak distribuování aplikací pro iOS nám přináší mnoho úskalí. Apple se snaží, abychom k distribuci používali oficiální obchod App Store a alternativní způsoby značně omezuje. Z licenčního hlediska největší míru svobody v distribuování přináší účet Apple Developer Enterprise Program, který ale není jednoduché získat a společnost Agentes IT k tomu potřebné podmínky nesplňuje. Pokud bychom si tedy chtěli vytvořit vlastní řešení, tak můžeme distribuovat aplikace pouze pro omezený okruh uživatelů v rámci Ad hoc nebo Development metody distribuce. Případně také pokud by zákaznická firma splnila podmínky a sama si zakoupila členství v Apple Developer Enterprise Programu. V takovém případě by bylo možné jí distribuovat aplikace pro interní účely pomocí našeho řešení bez omezení počtu zařízení. Veřejné aplikace lze distribuovat pouze přes App Store.

Z technologického hlediska to také není úplně přívětivé. Soubor IPA nejde jednoduše nainstalovat jako je to u souboru APK. To lze naštěstí vyřešit nahráním souboru IPA na web společně se správně vyplněným souborem PLIST, který instalaci umožní.

2.2 Analýza jednotlivých řešení

Nyní, když jsme si objasnili technologické a licenční možnosti a omezení u obou platforem, můžeme přejít k analýze jednotlivých řešení.

2.2.1 Google Play

Platforma, která vývojářům umožňuje distribuovat přes Google Play, se nazývá Play Console a jedná se o webový portál. Vývojář si zde musí vytvořit vývojářský účet a uhradit jednorázový poplatek ve výši 25 $.

Na portálu lze přidat další uživatele a spravovat jejich role a práva v rámci celého účtu či pouze u jednotlivých aplikací [8]. Testery či skupiny testerů lze určovat výčtem emailových adres.

Vývojář musí u každé aplikace nejprve vyplnit všechny důležité informace o aplikaci, nahrát grafické soubory, provést hodnocení obsahu aplikace a další potřebné úkony.

Pro distribuci jednotlivých buildů aplikace lze využít několika typů kanálů pro testování v různých fázích vývoje a také kanál produkční: [9]

- Interní testování – 100 testerů
- Uzavřené testování – širší okruh důvěryhodných testerů,
- Otevřené testování – široký okruh uživatelů, volně dohledatelné na Google Play
- Produkce – dostupné komukoliv na Google Play

V rámci uzavřeného testování si můžeme vytvořit více kanálů a v každém kanále distribuovat verzi pro určitou skupinu testerů. Uživatel může být přihlášen k testování pouze v jednom kanálu a každý kanál vždy distribuuje pouze jednu verzi buildu. Uživatel tedy nemá přístup k historii verzí. Při publikaci nového buildu prochází build kontrolou, a tak může trvat několik hodin, než bude k dispozici ke stažení. Pouze v kanále interního testování by to mělo být během několika minut. [9]

Po nahrání buildu se testerům musí odeslat odkaz, na kterém se přihlásí k testování, a odtud jsou přesměrováni do Google Play, kde si mohou aplikaci stáhnout. V zobrazení aplikace v Google Play poté mohou testeři i odeslat soukromou zpětnou vazbu s hodnocením a komentářem.

Kromě aplikací pro veřejnost je možné distribuovat i soukromé aplikace. V nastavení lze zvolit, aby byla aplikace soukromá, a přidat organizace, pro které bude určená. Zákaznické organizaci se poté aplikace zobrazí v sekci *Managed Google Play* a pomocí EMM lze poté centrálně nainstalovat aplikaci koncovým uživatelům. [10] EMM neboli *Enterprise Mobility Management* je nástroj ke správě mobilních zařízení uvnitř organizace.

Výhody:

- Nízká cena
- Automatické aktualizace aplikací
- Možnost vložení zpětné vazby u testovacích verzí a veřejných aplikací
- Správa uživatelů, testerů a jejich pravomocí
- Vysoký rozsah publicity u veřejných aplikací

Nevýhody:

- Chybí historie verzí
- Vyplňování velkého množství informací o aplikaci
- Dlouhý proces schvalování buildů
- Aplikace určené pro třetí strany – distribuce pomocí EMM a bez možnosti zpětné vazby
- Pouze pro Android

2.2.2 App Store

App Store Connect je řešení pro účastníky Apple Developer Programu, které umožnuje distribuovat pouze archiv aplikace, u kterého jsme zvolili metodu App Store Connect. Řešení tedy zahrnuje testování aplikace přes TestFlight a následné zveřejnění v App Store, případně v Apple Business Manageru.

Správa uživatelů a jejich rolí probíhá na webovém portálu **App Store Connect**. Uživatel musí být členem Apple Developer Programu, aby se stal vlastníkem týmu, avšak samotný portál už je zdarma. Do tohoto týmu může přizvat další uživatele a přiřadit jim role, které určí rozsah jejich kompetencí. [11]

K testování můžeme určit až 100 členů našeho týmu jako beta testery. Také je lze přiřadit do více skupin a každé skupině přiřadit konkrétní build k testování. Dále je možné přizvat až 10 000 externích testerů zadáním jejich emailové adresy nebo zasláním odkazu. [12]

Samotné testování probíhá přes **TestFlight,** což je mobilní aplikace, která umožňuje uživatelům instalovat a testovat iOS aplikace. Po spuštění se uživateli zobrazí seznam aplikací, které může testovat. U každé aplikace uživatel nalezne historii buildů s možností instalace.

Každý build je dostupný k testování 90 dní. Je zde také možné odeslat zpětnou vazbu na aplikaci a přiložit k tomu i screenshot obrazovky. [11]

Nahrané buildy podléhají schvalovacímu procesu, ve kterém Apple kontroluje aplikace ve snaze zjistit, zda jsou spolehlivé, fungují dle očekávání, respektují soukromí uživatelů a neobsahují závadný obsah. Taková kontrola může trvat několik hodin, ale i několik dní. [13]

Distribuovat lze také aplikace soukromé pro jednu nebo více organizací. Cílová organizace si nejprve musí založit účet v **Apple Business Manageru**. V App Store Connect poté můžeme nastavit naši aplikaci jako privátní a přiřadit dané organizaci. Aplikace se následně organizaci zobrazí v Apple Bussiness Manageru, odkud ji může zakoupit. [14]

Samotná instalace probíhá pomocí **Mobile Device Managementu** (MDM), což je nástroj ke správě mobilních zařízení v organizaci. Uživatel, jehož zařízení spravuje MDM, si tedy aplikaci nemusí nikde stahovat, jelikož je mu do zařízení nainstalována automaticky. Pomocí Apple Business Manageru a Mobile Device Managementu lze stejným způsobem také objednat a distribuovat i veřejnou aplikaci pro své zaměstnance. [14]

Výhody:

- Zdarma (platí se pouze za Apple Developer Program, který je nutný tak či tak)
- Automatické aktualizace aplikací
- Přehledné testování a historie verzí 90 dní
- Možnost odeslání zpětné vazby u testovacích verzí a veřejných aplikací
- Správa uživatelů, testerů a jejich pravomocí
- Vysoký rozsah publicity u veřejných aplikací

Nevýhody:

- Vyplňování velkého množství informací o aplikaci
- Dlouhý proces schvalování buildů
- Aplikace určené pro třetí strany – distribuce pomocí MDM a bez možnosti zpětné vazby
- Pouze pro iOS

2.2.3 Applivery

Applivery je řešení, které nám umožňuje distribuovat aplikace jak pro Android, tak i pro iOS. U iOS je možné distribuovat aplikace vytvořené metodami Ad Hoc, Development či Enterprise. Aplikace se skládá ze dvou hlavních částí. První částí je **webová administrace** a druhou částí je **obchod aplikací**.

Ve webové administraci lze spravovat uživatele a jejich role buď globálně, nebo pouze pro konkrétní projekt. Dále se zde nahrávají nové buildy a spravují jejich distribuce.

Do obchodu aplikací se dostaneme po naskenování QR kódu nebo pomocí odkazu. Obchod aplikací se nám otevře v prohlížeči, ale je optimalizovaný pro zobrazení na mobilu. V obchodě aplikací se nám zobrazí u každé aplikace seznam distribucí. U každé distribuce nalezneme historii buildů se základními údaji a možností instalace.

Při vytváření nového buildu nahrajeme pro iOS instalační soubor IPA. Pro Android můžeme nahrát instalační soubor APK, případně balíček ABB. Kromě instalačního souboru také vyplníme číslo verze, changelog a můžeme přidat libovolný počet tagů.

Aby se build zobrazil v obchodě aplikací, musíme vytvořit takzvanou distribuci. **Distribuce** zahrnuje podmnožinu buildů aplikace, které chceme v obchodu aplikací zveřejnit. Podmnožinu buildů můžeme určit například vyfiltrováním podle některého z tagů. Buildy lze tedy jednoduše rozdělit do distribucí podle účelu. Některá může obsahovat buildy pro testování a některá pro produkční verze.

Každá distribuce má tři možnosti ochrany: public, private, password. **Public** umožňuje stažení i nepřihlášeným uživatelům. **Private** umožňuje stažení pouze přihlášeným uživatelům, kteří mají oprávnění. **Password** požaduje zadání předem definovaného hesla, aby došlo ke stažení. Také lze nastavit viditelnost distribuce, zda se bude v obchodě aplikací zobrazovat, nebo bude přístupná pouze pomocí odkazu.

Pro odesílání zpětné vazby si musí uživatelé do zařízení nainstalovat **Applivery SDK.** Tento balíček nám také umožní nastavit automatické aktualizace aplikací. [15]

Applivery nabízí několik cenových plánů. Pokud bychom si vybrali plán, ve kterém lze distribuovat až 15 aplikací, tak bychom zaplatili 2 990€ ročně. Pokud by naše firma expandovala a potřebovali bychom plán pro 30 aplikací, tak cena činí 5 990€ za rok. [16]

Výhody:

- Pro iOS i Android
- Automatické aktualizace aplikací
- Přehledné testování a historie verzí
- Možnost odeslání zpětné vazby
- Buildy nepodléhají kontrole a jsou ihned k dispozici ke stažení
- Správa uživatelů, testerů a jejich pravomocí
- Aplikace určené pro třetí strany distribuovány bez nutnosti dalších nástrojů

Nevýhody:

- Vysoká cena
- Automatické aktualizace a odesílání zpětné vazby vyžaduje Applivery SDK
- Nelze distribuovat veřejné aplikace pro iOS

2.3 Porovnání řešení

Nyní přistoupíme k porovnání výše uvedených řešení. Během analýzy jsme zjistili, že každé řešení volí trochu jiný přístup. Jejich silné a slabé stránky se liší v závislosti na scénářích užití. Některé řešení v určitých případech ani nelze využít (např. veřejné iOS aplikace nelze publikovat jinde než na App Store). Ztrací tedy smysl snažit se vyhodnotit univerzálně nejlepší řešení. Aby bylo porovnání lépe vypovídající, rozdělíme si ho do tří kategorií - na distribuce **testovacích**, **produkčních veřejných** a **produkčních soukromých** aplikací. Produkčními soukromými aplikacemi rozumíme aplikace vyvíjené na zakázku pro třetí strany.

U porovnávání nás bude hlavně zajímat, zda v každé kategorii nalezneme mezi současnými řešeními to optimální, nebo zda bychom dokázali vytvořit řešení, které by našim potřebám vyhovovalo více. V analýze jsme se dozvěděli, jaká mají distribuce pro oba operační systémy technologická a licenční omezení. Tyto znalosti zde nyní využijeme a do tabulky si tedy přidáme hypotetické řešení, které bychom teoreticky měli být schopni vytvořit, a pojmenujeme si ho pracovním názvem ***Agenstore***.

K porovnání řešení využijeme metod vícekriteriálního hodnocení. Nejprve je nutné stanovit kritéria pro hodnocení a jejich váhy. Součet vah kritérií se vždy musí rovnat 1.

2.3.1 Stanovení kritérií a přiřazení vah

Stanovení těchto kritérií a vah je konzultováno s managementem firmy Agentes IT. Jedná se o menší firmu s přibližně 20 zaměstnanci, a proto hlavním kritériem s největší přiřazenou váhou je cena. Do ceny započítáváme pouze licenční náklady na samotný systém. Náklady na licence u daných operačních systémů (Apple Developer Program, Apple Developer Enterprise Program) musíme platit nad rámec každého řešení, proto je do srovnání zahrnovat nebudeme. Dále máme u každé kategorie trochu jiné požadavky, a tudíž ostatní kritéria a jejich váhy se liší.

Distribuce testovacích verzí

Pro testování potřebujeme systém správy uživatelů a jejich rolí, abychom mohli určit vývojáře jednotlivých projektů a také testery. V samotném průběhu testování je důležité, abychom byli schopni určeným uživatelům rychle distribuovat nové verze. Pokud build prochází schvalovací kontrolou, tak dochází ke zbytečným časovým prodlevám a také může z různých důvodů dojít k neschválení. Vyplňování detailních informací o obsahu aplikace (věk, cílené publikum, screenshoty, výskyt vulgarismů atd.) je také nežádoucí a při nesprávném vyplnění může akorát způsobit, že build neprojde schvalovací kontrolou. Tester by měl mít přístup k historii verzí s popisem, co se v nich změnilo, a možnost si kteroukoliv nainstalovat pro lepší porovnávání a hledání výskytu chyb. Důležité také je, aby nám po otestování mohl odeslat zpětnou vazbu.

Pro kategorii **Testování** jsme tedy zvolili tyto kritéria a jejich váhy:

- **0,40 - Cena**
- **0,16 - Správa uživatelů a rolí** – určení vývojářů, testerů a jejich pravomocí
- **0,19 - Historie verzí** – možnost instalace, kompletní historie
- **0,13 - Kontrola** – časová prodleva
- **0,04 - Povinné info** – objem povinných informací k vyplnění
- **0,08 - Feedback** – možnost odeslání zpětné vazby, jednoduchost

Distribuce produkčních verzí veřejných aplikací

U publikace veřejných aplikací bývá velkou výhodou aplikaci distribuovat přes oficiální obchod dané platformy, jelikož nám to umožní oslovit širokou veřejnost. Na správu uživatelů a rolí zde není kladen takový nárok, jelikož aplikace si smí nainstalovat kdokoliv a

není takový problém, aby produkční verzi jednou za čas nahrál administrátor ze svého účtu. Výhodou je, pokud se po prvotní instalaci aplikace její aktualizace instalují automaticky. Stejně jako u testování oceníme, pokud nemusíme vyplňovat velký objem informací a pokud aplikace nepodléhá dlouhotrvající schvalovací kontrole. Také je dobré, když nám uživatelé mohou odeslat zpětnou vazbu.

Pro kategorii **Produkční veřejné** jsme tedy zvolili tyto kritéria a jejich váhy:

- 0,40 - **Cena**
- 0,03 - **Správa uživatelů a rolí** – určení vývojářů a jejich pravomocí
- 0,15 - **Aktualizace** – automatická instalace aktualizací
- 0,09 - **Kontrola** – časová prodleva
- 0,03 - **Povinné info** – objem povinných informací k vyplnění
- 0,05 - **Feedback** – možnost odeslání zpětné vazby, přehlednost, jednoduchost
- 0,25 - **Oficiální obchod** – přirozený rozsah publicity

Distribuce produkčních verzí soukromých aplikací

Při publikaci soukromých aplikací je to stejné jako u veřejných aplikací z hlediska správy vývojářů, automatických aktualizací, schvalovací kontroly, vyplňování informací o aplikaci a také odesílání zpětné vazby. Nad rámec toho nás u soukromých zajímá, jestli zákazník musí využívat systému MDM nebo EMM k tomu, aby byl vůbec schopen nainstalovat aplikace na zařízení svých zaměstnanců. Vhodnější řešení by umožňovalo zpřístupnit stažení vybraným uživatelům tak jako u distribuce testovacích verzí.

Pro kategorii **Produkční soukromé** jsme tedy zvolili tyto kritéria a jejich váhy:

- 0,40 - **Cena**
- 0,03 - **Správa uživatelů a rolí** – určení vývojářů a jejich pravomocí
- 0,15 - **Aktualizace** – automatická instalace aktualizací
- 0,09 - **Kontrola** – časová prodleva
- 0,03 - **Povinné info** – objem povinných informací k vyplnění
- 0,05 - **Feedback** – možnost odeslání zpětné vazby, přehlednost, jednoduchost
- 0,25 - **Požadavky na zákazníka** – nutnost MDM či EMM

2.3.2 Vyhodnocení kritérií

Vyhodnocení variant proběhlo metodou váženého součtu. Jednotlivým řešením je pro každé kritérium udělena hodnota v rozmezí 0–1 podle toho, jak dobře dané kritérium splňují. 1 – nejlepší; 0 – nejhorší. Řešení s nejvyšším váženým součtem vyhrává.

Distribuce testovacích verzí

Na obrázku č. 2 vidíme, že v kategorii **testovacích** verzí nejhůře dopadlo Applivery, zejména kvůli vysoké ceně. Správu uživatelů a rolí zvládají dobře všechna řešení. Historie verzí je nejhorší u Google Play, které historii postrádá a pouze umožňuje distribuovat různé buildy v rámci více kanálů. App Store nedosáhl na celý bod z důvodu omezení historie na 90 dní. Kontrolu buildu a vyplňování přebytečných informací vyžadují Google Play i App Store, u kterého však kontrola trvá déle. Ve feedbacku si nejhůře vedlo Applivery, jelikož vyžaduje, aby měl uživatel nainstalovaný Applivery SDK. Část bodu jsme strhli i u Google Play, jelikož neumožňuje ke komentáři přiložit screenshot.

Jako nejlepší řešení se ukázala naše hypotetická aplikace Agenstore, která skvěle vyhovuje ve všech parametrech.

Testování	Váha	Google Play	App Store	Applivery	Agenstore
Cena	0,4	0,99	1	0,05	1
Správa uživatelů a rolí	0,16	1	1	1	1
Historie verzí	0,19	0,1	0,8	1	1
Kontrola	0,13	0,5	0,2	1	1
Info vyplňování	0,04	0,3	0,3	1	1
Feedback	0,08	0,8	1	0,5	1
Vážený součet		**0,716**	**0,83**	**0,58**	**1**
Pořadí		**3.**	**2.**	**4.**	**1.**

Obrázek 2: Porovnání řešení – distribuce testovacích verzí (Zdroj: autor)

Distribuce produkčních veřejných verzí

U veřejného publikování je potřeba říct, že pro Applivery a Agenstore jsme hodnotili publikování aplikací pouze pro Android. Apple neumožňuje distribuci veřejných aplikací jiným způsobem než přes oficiální App Store, a tak v tomto ohledu nelze hledat alternativy.

To nás však nemusí mrzet, jelikož na obrázku č.3 vidíme, že se Google Play a App Store stejně ukázali jako nejlepší řešení, a to zejména proto, že veřejné aplikace většinou

požadujeme distribuovat právě pomocí oficiálních obchodů a také skvěle zvládají automatické aktualizace. Automatické aktualizace umí i Applivery, ale pouze pokud máme nainstalovaný Applivery SDK.

Agenstore se v tomto případě ukázal jako průměrné řešení a Applivery je kvůli své vysoké ceně opět nejhorší.

Produkce - Veřejné	Váha	Google Play	App Store	Applivery	Agenstore
Cena	**0,4**	0,99	1	0,05	1
Správa uživatelů a rolí	**0,03**	1	1	1	1
Aktualizace	**0,15**	1	1	0,5	0
Kontrola	**0,09**	0,5	0,2	1	1
Povinné info	**0,03**	0,3	0,3	1	1
Feedback	**0,05**	0,8	1	0,5	1
Oficiální obchod	**0,25**	1	1	0	0
Vážený součet		**0,92**	**0,907**	**0,27**	**0,6**
Pořadí		**1.**	**2.**	**4.**	**3.**

Obrázek 3: Porovnání řešení – produkční veřejná distribuce (Zdroj: autor)

Distribuce produkčních soukromých verzí (aplikace vyvíjené pro třetí strany)

U soukromého publikování produkčních verzí se ukázalo jako nejlepší řešení Agenstore (obr. 4). Oproti Google Play a App Store zákazník nepotřebuje spravovat zařízení svých zaměstnanců pomocí MDM nebo EMM, ale aplikaci si do zařízení mohou nainstalovat uživatelé sami a poté také odeslat zpětnou vazbu. Jedinou nevýhodou Agenstore je, že není schopné spravovat automatické aktualizace aplikací. Applivery by si v této kategorii vedlo velice dobře, ale kvůli vysoké cenovce se opět umístilo poslední.

Produkce - Soukromé	Váha	Google Play	App Store	Applivery	Agenstore
Cena	**0,4**	0,99	1	0,05	0,9
Správa uživatelů a rolí	**0,03**	1	1	1	1
Aktualizace	**0,15**	1	1	0,5	0
Kontrola	**0,09**	0,5	0,2	1	1
Info vyplňování	**0,03**	0,3	0,3	1	1
Feedback	**0,05**	0	0	0,5	1
Požadavky na zákazníka	**0,25**	0	0	1	1
Vážený součet		**0,63**	**0,607**	**0,52**	**0,81**
Pořadí		**2.**	**3.**	**4.**	**1.**

Obrázek 4: Porovnání řešení – produkční soukromá distribuce (Zdroj: autor)

2.3.3 Závěr srovnání

Porovnání stávajících řešení s naším hypotetickým řešením Agenstore nám přineslo několik poznatků. Dozvěděli jsme se, že pro testování aplikací by bylo nejvhodnější námi vyvinuté řešení. Zbavili bychom se tím časové prodlevy, které způsobuje schvalovací kontrola buildů. Na rozdíl od Google Play bychom také měli k dispozici historii verzí.

Pro publikování veřejných aplikací je nejvhodnější využít oficiálních platforem Google Play a App Store.

Pro distribuci soukromých aplikací vyvíjených pro třetí strany se opět ukázalo jako nejvhodnější naše hypotetické řešení, jak pro Android, tak pro iOS. Je to způsobené zejména tím, že po zákazníkovi nevyžaduje, aby spravoval mobilní zařízení svých zaměstnanců pomocí EMM nebo MDM. Toto kritérium se však může lišit v závislosti na požadavcích zákazníka.

Někteří zákazníci například mohou brát soukromou distribuci přes EMM či MDM jako výhodu, a dokonce ji požadovat. V takovém případě by mohlo být vhodnější využít Google Play a App Store. Také jsme stále limitováni omezeními, které jsou kladené na distribuci iOS aplikací. Je tedy nutné vzít v úvahu, zda je firma, pro kterou vyvíjíme, schopna získat Apple Developer Enterprise Program, či zda pro její účely bude stačit využít Apple Developer Programu s omezením na 100 zařízení. Pokud není ani jedna podmínka splněna, bude lepší využít App Store.

Vidíme, že výsledky porovnání nelze zcela generalizovat a u každého projektu je vždy potřeba se zamyslet zvlášť, které řešení je nejvhodnější. Jako základní přehled výhod a nevýhod nám může posloužit tato tabulka s parametry jednotlivých řešení. (obr. 5)

	App Store	Google Play	Applivery	Agenstore
Distribuce pro platformy	iOS	Android	Android, iOS	Android, iOS
Rychlost dostupnosti instalace nově nahraných verzí	1-2 dny	do 24 hodin	Okamžitě	Okamžitě
Historie verzí	Ano - 90 dní	Ne	Ano - kompletní	Ano - kompletní
Zpětná vazba u testovacích verzí	Ano	Ano	Ano - přes Applivery SDK	Ano
Distribuce veřejných aplikací pro iOS	Ano	Ne	Ne	Ne
Rozsah publicity u veřejných aplikací	Vysoký	Vysoký	Nízký	Nízký
Zpětná vazba u veřejných aplikací	Ano	Ano	Ano - přes Applivery SDK	Ano
Distribuce uzavřenému okruhu uživatelů třetí strany přímo v systému	Ne	Ne	Ano	Ano
Způsob distribuce uzavřenému okruhu uživatelů třetí strany	Přes MDM	Přes EMM	Nevyžaduje další nástroj	Nevyžaduje další nástroj
Zpětná vazba uživatelů třetích stran	Není možná	Není možná	Ano - přes Applivery SDK	Ano
Automatické aktualizace	Ano	Ano	Ano - přes Applivery SDK	Ne
Náklady na licenci samotného systému - počet projektů	Zdarma - neomezeně	25$ jednorázově - neomezeně	2990€ ročně - 15 projektů	Zdarma - neomezeně
Náklady na licence Android - počet zařízení		Zdarma - neomezeně	Zdarma - neomezeně	Zdarma - neomezeně
Náklady na licence iOS - počet zařízení	99$ ročně (ADP) - neomezeně		99$ ročně (ADP) - 100 zařízení **nebo** 299$ ročně (ADEP) - neomezeně	99$ ročně (ADP) - 100 zařízení **nebo** 299$ ročně (ADEP) - neomezeně

Obrázek 5: Přehled parametrů distribučních řešení (Zdroj: autor)

Zkratky: ADP - Apple Developer Program ADEP - Apple Developer Enterprise Program

2.4 Specifikace softwarových požadavků

Na základě provedené analýzy a požadavků firmy Agentes IT jsme stanovili funkční a nefunkční požadavky pro nově vyvíjené řešení.

2.4.1 Funkční požadavky

- Přihlášení do aplikace
- Vyresetování zapomenutého hesla
- Správa společností administrátorem
    - Seznam společností – možnost filtrování, stránkování, volba počtu záznamů na stránce (10, 20, 50)
    - Vytvoření, mazání, editace, detail
- Správa projektů administrátorem
    - Seznam projektů – možnost filtrování, stránkování, volba počtu záznamů na stránce (10, 20, 50)
    - Vytvoření, mazání, editace, detail
    - Nastavení zabezpečení projektu – soukromý/veřejný
- Správa uživatelů administrátorem
    - Seznam uživatelů – možnost filtrování, stránkování, volba počtu záznamů na stránce (10, 20, 50)
    - Vytvoření, mazání, editace, detail
    - Nastavení rolí uživatelů, dle kterých jim jsou funkcionality zpřístupněné
    - Přiřazování uživatelů k projektům
- Správa buildů administrátorem či k projektu přiřazeným vývojářem
    - Vytvoření záznamu a nahrání souboru APK či IPA, mazání, editace
- Zobrazení seznamu projektů, ke kterým má přihlášený uživatel přístup
    - Zobrazení seznamu uživatelů přiřazeným k projektu – omezeno dle role uživatele.
    - Zobrazení seznamu buildů pro Android/iOS, ke kterým má uživatel přístup dle role a dle fáze vývoje (develop/staging/produkce)
        - Možnost filtrování, stránkování, volba počtu záznamů na stránce (10, 20, 50)
        - Zobrazení detailu buildu

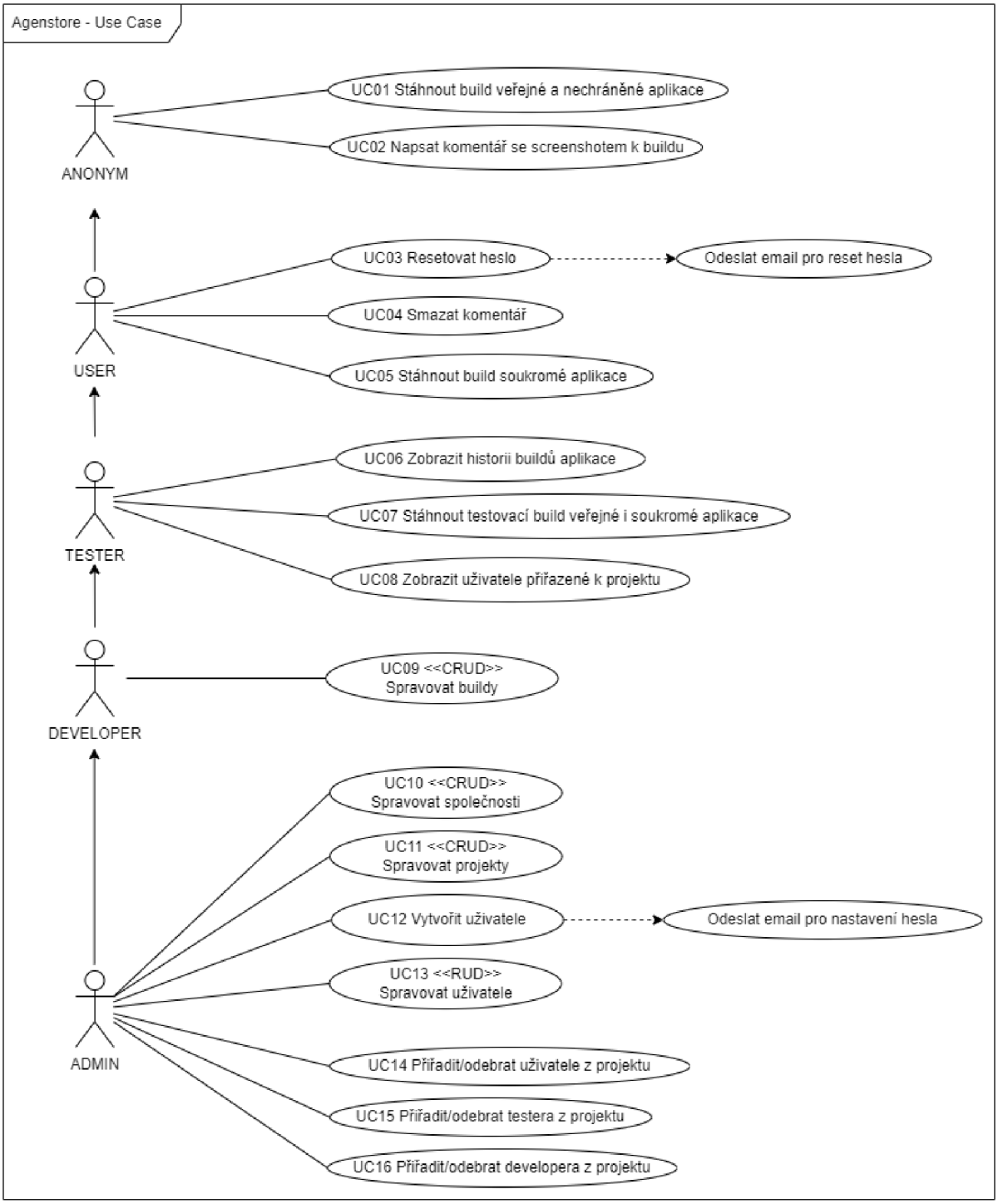
        - Zobrazení textového odkazu, který nás v mobilu přesměruje na stránku, odkud lze build stáhnout
        - Zobrazení QR kódu, který po naskenování mobilem přesměruje na stránku, odkud lze build stáhnout
        - Možnost napsání komentáře k buildu a přiložení screenshotu
        - Možnost smazání svého komentáře
- Možnost stažení buildu, okomentování a přiložení screenshotu, pokud na to má uživatel právo – dle typu zabezpečení projektu, zda je uživatel k projektu přiřazen a dle jeho role

2.4.2 Nefunkční požadavky

- Aplikace s webovým grafickým rozhraním
- Využití programovacího jazyka Java a frameworku Spring Boot
- Víceuživatelský přístup

3 Design

3.1 Případy užití



Obrázek 6: Use Case diagram (Zdroj: autor)

V aplikaci se vyskytuje pět aktorů: **Anonym**, **User**, **Tester**, **Developer** a **Admin.** Funguje zde dědičnost a tak aktéři postupně nabalují use-cases předchozích aktérů.

Aktor **Anonym** představuje jakéhokoliv nepřihlášeného uživatele. Takovýto uživatel si může stáhnout produkční build veřejné aplikace, či kterýkoliv build nechráněné aplikace. K buildu může také napsat komentář a přiložit screenshot, aby dal vývojářům zpětnou vazbu.

Aktor **User** představuje řadového uživatele firmy, který si může stáhnout navíc také produkční buildy soukromých aplikací, ke kterým je přiřazen. K buildům také může přidávat komentáře se screenshotem, a navíc je mu umožněno svůj komentář smazat. Pokud zapomene heslo, tak si ho může vyresetovat. V tom případě se mu odešle email s odkazem pro nastavení nového hesla.

Aktor **Tester** má oproti **Userovi** rozšířenou pravomoc i na stahování a komentování testovacích buildů. Má také možnost zobrazit si historii buildů aplikace, aby si mohl pohodlně vybrat, který si nainstaluje. Dále si také může zobrazit seznam uživatelů přiřazených k projektu.

Aktor **Developer** je vývojář společnosti Agentes IT. U projektů, ke kterým je přiřazen, může nahrávat, editovat a mazat buildy.

Aktor **Admin** má pravomoci nejrozšířenější. Může spravovat společnosti, projekty i uživatele. Pokud vytvoří uživatele nového, tak se mu odešle email pro nastavení hesla. Také může k projektu přiřadit **Usera**, **Testera** či **Developera**, nebo je z projektu naopak odebrat.

3.2 Použité technologie

Použité technologie byly vybrány s ohledem na předešlé zkušenosti autora a také na technologický stack, který firma Agentes IT běžně využívá. Případné rozšiřování a úpravy aplikace v budoucnu by tak vzhledem k know-how uvnitř firmy mělo být jednodušší. Projekt je verzován pomocí gitu. Uložen je ve webovém repozitáři GitLab a je připraven k snadnému nasazování dalších funkcionalit.

Java

Zvolili jsme programovací jazyk Java ve verzi 11. V kombinaci s frameworkem Spring je vhodnou volbou pro tvorbu webových aplikací. Autor s ním má největší zkušenosti a taktéž firma Agentes IT jej primárně používá. Volba tedy vychází i z nefunkčních požadavků. [17]

Spring, Spring Boot

Spring je populární framework, který slouží pro jednoduchý vývoj webových aplikací v Javě. [18] **Spring Boot** je rozšíření Springu. Umožňuje nám na začátku zvolit, co budeme v projektu chtít používat, a podle toho obstará všechny potřebné závislosti, a tím nám zjednoduší počáteční konfiguraci projektu. [19]

PostgreSQL

PostgreSQL je OpenSource objektově-relační databázový systém, který využíváme pro ukládání dat. [20]

Apache Maven

Nástroj pro správu závislostí a sestavování projektů zejména pro projekty v Javě. Díky němu nemusíme knihovny a pluginy stahovat ručně, ale nadefinujeme je v souboru pom.xml. [21]

Thymeleaf

Moderní javovský šablonovací systém na straně serveru pro webová i samostatná prostředí. V aplikaci jej využíváme k vkládání proměnných do HTML šablon, které lze elegantně skládat z více fragmentů, a tvoří nám tedy frontend naší aplikace. [22]

jQuery

jQuery je malá a rychlá Javascriptová knihovna. V aplikaci ji využíváme k oživení statických HTML stránek, například pro zobrazování modálních oken či dynamické změny v struktuře DOM. [23]

Bootstrap

Bootstrap je populární front-endový CSS framework pro vytváření responzivních webových aplikací. Obsahuje v sobě Grid systém a umožňuje nám využívat již nadefinované CSS styly. [24]

Do projektu je také přidán balíček stylů poskytnutý firmou školitele, které firma využívá i u jiných projektů, abychom zachovali firemní vzhled aplikací.

Liquibase

Liquibase je OpenSource projekt, který umožňuje spravovat změny schématu databáze. Aplikaci je díky tomu snadné průběžně dále rozšiřovat. [25]

Git a GitLab

Git je distribuovaný verzovací systém. GitLab je webový repozitář gitu, který nám také pomocí svého CI/CD umožňuje automatické nasazení aplikace. [26] [27]

Docker

Docker slouží pro kontejnerizaci aplikací. Aplikace je zabalená do kontejneru se vším potřebným ke spuštění (kód, knihovny, nastavení). Díky tomu je izolována a nezávisí na prostředí, ve kterém je kontejner spuštěn. V našem případě v jednom kontejneru běží aplikace ve Springu a v druhém kontejneru databáze. [28]

Kubernetes

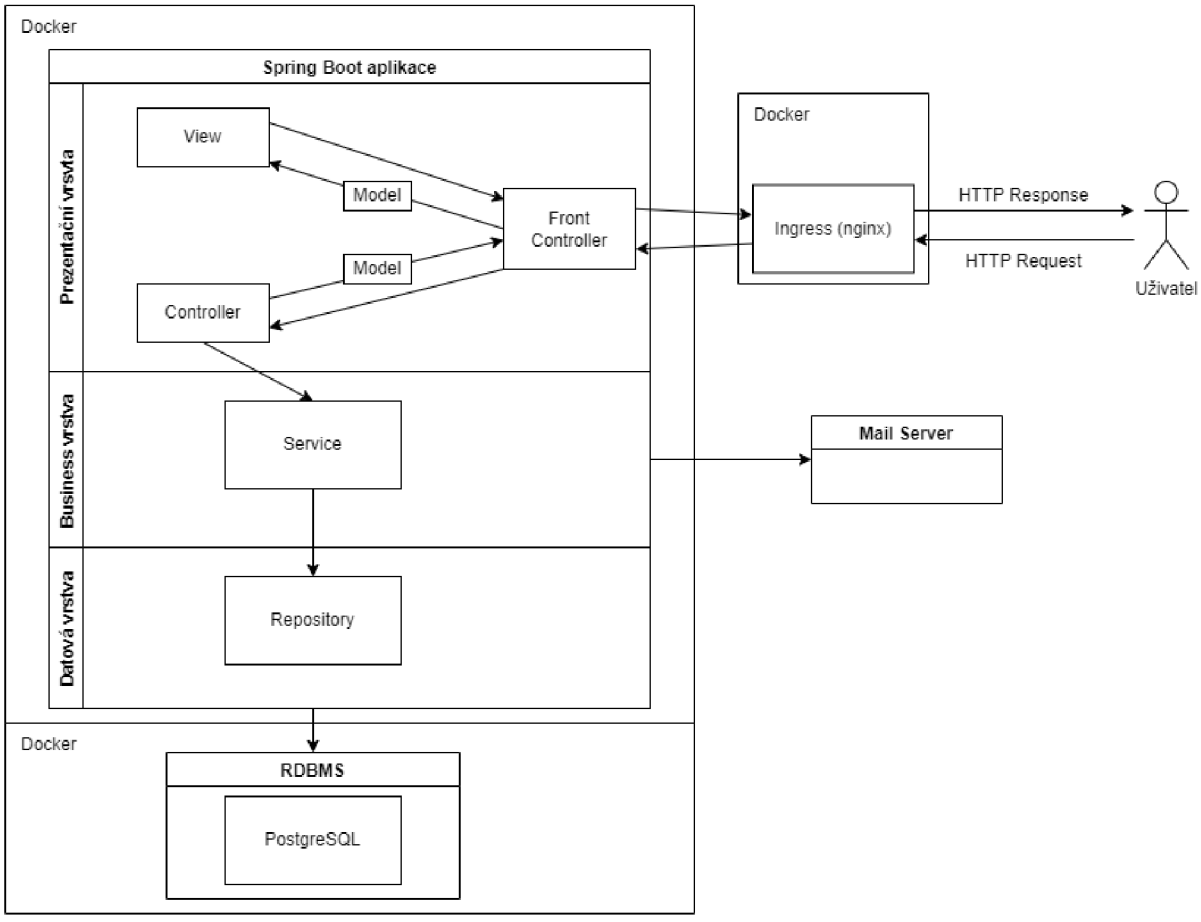
Nástroj pro orchestraci virtualizace na úrovni operačního systému. Stará se o to, aby naše aplikace běžela stabilně bez výpadků. Sleduje zátěž u jednotlivých kontejnerů a lze využít horizontálního i vertikálního škálování. Pokud některý kontejner selže, tak ho nahradí novým. [29]

3.3 Architektura aplikace

Architektura aplikace vychází z technologií, které jsme zvolili. Aplikace funguje jako jeden celek, který obsahuje backend i frontend.

Aplikace je rozdělena do tří vrstev na prezentační, business a datovou vrstvu. Základní pravidlo této třívrstvé architektury je, že prezentační vrstva nikdy přímo nekomunikuje s vrstvou datovou. Komunikace těchto vrstev vždy probíhá přes bussiness vrstvu, která je mezi nimi.

V prezentační vrstvě implementujeme architekturu MVC. Příchozí HTTP žádosti zachytává takzvaný Dispatcher, který implementuje návrhový vzor Front Controller a podle URL směřuje žádosti k příslušným Controllerům. **Controller** je zodpovědný za naplnění **Modelu** daty a výběr **View**. V našem případě v Controlleru zvolíme konkrétní HTML šablonu, která data z Modelu poté uživateli zobrazí.



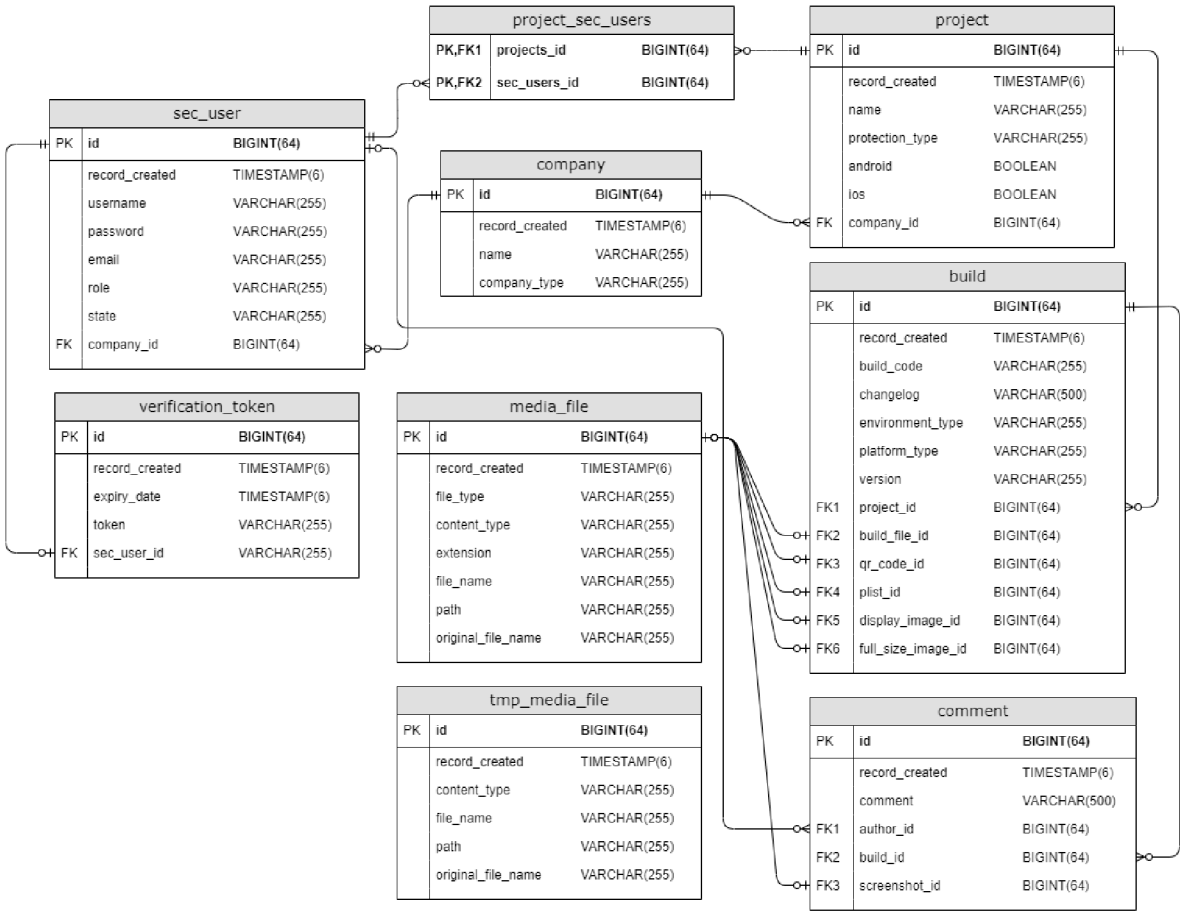
Obrázek 7: Architektura aplikace (Zdroj: autor)

Business vrstva obsahuje **Service** třídy, ve kterých se odehrává logika aplikace. Datová vrstva obsahuje třídy **Repository** a přímo komunikuje s databází pomocí ORM. Využívá se k tomu specifikace JPA, která je ve Springu defaultně implementována pomocí Hibernate.

Aplikace je nasazena do Kubernetes clusteru. Jednotlivé části aplikace (Spring boot aplikace, databáze) jsou zabaleny do Docker kontejnerů. V Kubernetes běží jednotlivé kontejnery v oddělených oblastech – tzv. podech. Pody můžeme chápat jako malinké virtuální počítače. S narůstajícím počtem projektů, a tudíž i uživatelů, může docházet ke zvýšenému provozu. Pokud by stávající zdroje přestávali stačit, tak můžeme zareagovat pomocí horizontálního škálování. V takovém případě se spustí v novém podu další instance naší Spring boot aplikace. Na obrázku architektury aplikace vidíme, že příchozí http žádosti uživatele nejprve putují přes Ingress. Ingress není součástí naší aplikace, ale běží v rámci celého Kubernetes clusteru. Příchozí http požadavky předává jednotlivým instancím naší aplikace a stará se tak o vyrovnávání zátěže.

Instalační soubory IPA a APK, ale i soubor PLIST a všechny obrázky ukládáme na disk. Do databáze vkládáme pouze záznam, který obsahuje cestu k souboru. Vzhledem k povaze dat jsme vyhodnotili, že není nutné tyto soubory na disku šifrovat. Mobilní aplikace po uživateli stejně požadují přihlášení, pokud přistupují k nějakým citlivým datům, a tak by samotné odcizení instalačního souboru aplikace nepředstavovalo vážný problém.

3.4 Datový model



Obrázek 9: Datový model (Zdroj: autor)

Základní tabulkou celé aplikace je společnost (***company***). Společnost může obsahovat více projektů a také více uživatelů, což v obou případech znázorňuje vazba 1:N. Uživatelům může být přiřazena role v závislosti na tom, jakého typu je společnost, ke které patří (administrátorská, zákaznická), což je u obou typů společnosti určeno výčtem rolí (obr. 9).

```
@Getter
@AllArgsConstructor
public enum CompanyType {
    ADMIN(List.of(SecRole.ADMIN, SecRole.DEVELOPER, SecRole.SUPERTESTER, SecRole.USER)),
    CUSTOMER(List.of(SecRole.TESTER, SecRole.USER));

    private List<SecRole> secUserRoles;
}
```

Obrázek 8: Typ společnosti – enum (Zdroj: autor)

Vidíme, že typ společnosti (*company_type*), se na relačním modelu databáze (obr. 8) ukládá jako VARCHAR(255), ale v aplikaci se jedná o výčtový typ *enum*. Datový typ

VARCHAR(255) používáme k ukládání i u všech ostatních výčtových typů *enum* v naší aplikaci, jelikož je ukládáme přímo ve formě textu.

Podle uživatelské role má uživatel mimo jiné přístup k zobrazení buildů a jejich stažení v různých fázích vývoje, což je u každé role určeno výčtem typů prostředí. (obr. 10)

```
@Getter
@AllArgsConstructor
public enum SecRole implements GrantedAuthority {

    ADMIN( authority: "ADMIN", List.of(EnvironmentType.DEVELOP, EnvironmentType.STAGING, EnvironmentType.PRODUCTION)),
    DEVELOPER( authority: "DEVELOPER", List.of(EnvironmentType.DEVELOP, EnvironmentType.STAGING, EnvironmentType.PRODUCTION)),
    SUPERTESTER( authority: "SUPERTESTER", List.of(EnvironmentType.DEVELOP, EnvironmentType.STAGING, EnvironmentType.PRODUCTION)),
    TESTER( authority: "TESTER", List.of(EnvironmentType.STAGING, EnvironmentType.PRODUCTION)),
    USER( authority: "USER", List.of(EnvironmentType.PRODUCTION));

    private String authority;
    private List<EnvironmentType> environmentTypeAccess;
}
```

Obrázek 10: Uživatelské role (Zdroj: autor)

Uživatel má dále klasické atributy jako je uživatelské jméno, heslo a email, pomocí kterého se přihlašuje, a musí být tudíž unikátní. Stav uživatele (state) může nabývat dvou hodnot: VERIFIED, UNVERIFIED. Poté, co administrátor vytvoří uživatele, se uživatel nachází ve stavu UNVERIFIED. Jakmile si uživatel email ověří a nastaví heslo, tak se stav změní na VERIFIED a uživatel se může přihlásit. K tomuto ověření se využívá token, který se ukládá do tabulky ***verification_token***, která je s uživatelem spojená vazbou 1:1. Atribut *expiry_date* určuje, do kdy je token platný.

Projekt má kromě názvu také atributy *android* a *ios*. Oba jsou typu *boolean* a určují, zda bude projekt obsahovat buildy pro daný operační systém. Dalším atributem projektu je typ zabezpečení (*protection_type),* který může nabývat hodnot: UNPROTECTED, PUBLIC a PRIVATE (obr. 11). To určuje, zda bude ke stažení v mobilu nutné přihlášení, či zda bude

```
@Getter
@AllArgsConstructor
public enum ProtectionType {
    UNPROTECTED(List.of(EnvironmentType.DEVELOP, EnvironmentType.STAGING, EnvironmentType.PRODUCTION)),
    PUBLIC(List.of(EnvironmentType.PRODUCTION)),
    PRIVATE(new ArrayList<>());

    private final List<EnvironmentType> environmentTypeAccess;
}
```

Obrázek 11: Typ zabezpečení – enum (Zdroj: autor)

stránka, odkud se build stahuje, přístupná pomocí odkazu i nepřihlášeným uživatelům. Projekty, které jsou PRIVATE, tak vyžadují přihlášení ve všech fázích vývoje. PUBLIC

vyžadují přihlášení u testovacích verzí, ale povolují instalaci produkčních buildů bez přihlášení. UNPROTECTED umožňují stažení všech verzí bez přihlášení.

Mezi projekty a uživateli je realizována vazba M:N pomocí asociační tabulky ***project_sec_users***. Do tabulky se ukládají záznamy o tom, kteří uživatelé jsou přiřazeni k jakému projektu.

Každý projekt může obsahovat více buildů, což je znázorněno vazbou 1:N. Tabulka ***build*** obsahuje atribut *build_code,* což je náhodný kód přiřazený buildu, který je jedinečný a s pomocí kterého se tvoří odkaz ke stažení, aby jej nebylo možné jednoduše odhadnout jako klasické id. Atributy *version* a *changelog* uchovávají číslo verze a popis změn v daném buildu. Atribut *platform_type* nám říká, pro kterou platformu je daný build (ANDROID, IOS). Atribut e*nvironment_type* určuje, zda je build určen k internímu testování během vývoje (DEVELOP), k testování v pokročilejší fázi zákazníkem (STAGING), či zda se jedná o produkční verzi (PRODUCTION).

Do tabulky ***media_file*** se ukládají záznamy o všech souborech, které uživatelé do aplikace nahráli, nebo které byly aplikací vygenerovány a jsou trvale uloženy na disku. Každý build má až 5 různých souborů, které k němu můžeme potřebovat uložit (instalační soubor, QR kód, PLIST a dvě ikony). V tabulce *build* máme tedy 5 cizích klíčů, z nichž nám každý odkazuje na id příslušného souboru v tabulce *media_file.* Tabulka *media_file* obsahuje atribut typ souboru (*file_type*), který je opět typu enum (obr. 12).

```
@Getter
@AllArgsConstructor
public enum FileType {

    ANDROID_BUILD( path: "android-build/", new String[] {"application/vnd.android.package-archive"}, new String[] {"apk"}),
    IOS_BUILD( path: "ios-build/", new String[] {"application/octet-stream"}, new String[] {"ipa"}),
    QR_CODE( path: "qr-code/", new String[] {"image/jpeg"}, new String[] {"jpg"}),
    PLIST( path: "plist/", new String[] {"application/x-plist"}, new String[] {"plist"}),
    SCREENSHOT( path: "screenshot/", new String[] {"image/jpeg"}, new String[] {"jpg"}),
    FULL_SIZE_IMAGE( path: "full-size-image/", new String[] {"image/png"}, new String[] {"png"}),
    DISPLAY_IMAGE( path: "display-image/", new String[] {"image/png"}, new String[] {"png"});

    private String path; // relative path to media root
    private String[] contentType; // supported MIME types, for example image/jpeg
    private String[] extension; // extensions

}
```

Obrázek 12: Typ souboru – enum (Zdroj: autor)

Typ souboru nám určuje část cesty, kam bude soubor uložen na disku. Dále definuje povolené MIME typy pro daný soubor a příponu. Kromě typu souboru ukládáme v tabulce *media_file* atributy:

path – relativní cesta k adresáři se souborem

file_name – název souboru na disku, při ukládání jej vygenerujeme, aby byl jedinečný

original_file_name – původní název souboru v době nahrání

content_type – MIME type souboru

extension – přípona

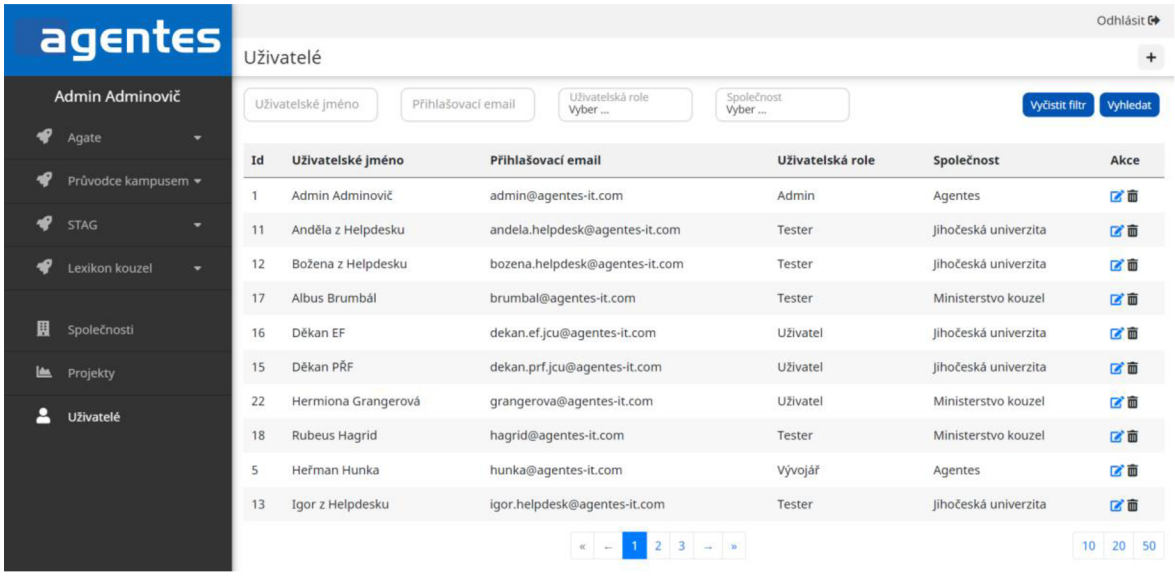
K buildům lze přidávat komentáře, a tak tabulka *build* má vazbu 1:N s tabulkou ***comment***. Ta obsahuje text komentáře (atribut *comment*) a také může obsahovat cizí klíč *author_id* v případě, že komentář odešle přihlášený uživatel. Ke komentáři lze také přidat obrázek, což řeší cizí klíč *screenshot_id* vazbou 1:1 na tabulku *media_file*.

Zcela osamoceně stojí tabulka ***tmp_media_file***, do které se ukládají záznamy o dočasně uložených souborech. Lze do ní například dočasně uložit nahraný instalační soubor, než uživatel vyplní ostatní data. Ve chvíli, kdy uživatel vytvoření buildu dokončí, je soubor přesunut na správné místo na disku a je vytvořen záznam v tabulce *media_file*. Záznam v tabulce *tmp_media_file* v té chvíli zaniká. Tabulka *tmp_media_file* tedy obsahuje atributy *content_type*, *file_name*, *path* a *original_file_name*, které jsou stejné jako v tabulce *media_file*.

4 Implementace

4.1 Grafické uživatelské rozhraní

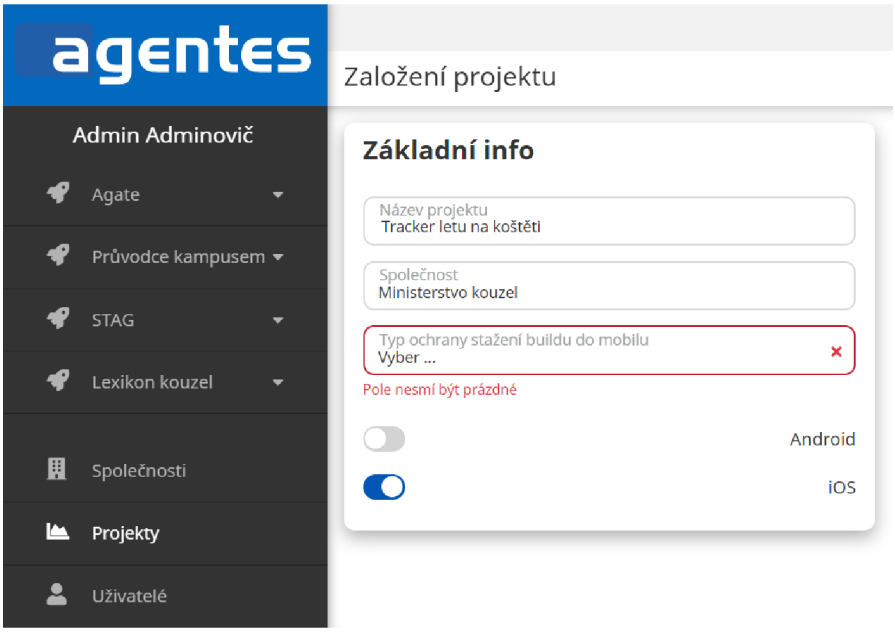
V uživatelském rozhraní nalezneme v levém horním rohu logo firmy Agentes IT. Pod ním se zobrazuje jméno právě přihlášeného uživatele a následuje menu. V horní části menu se zobrazují jednotlivé projekty, ke kterým má uživatel přístup. Pokud je přihlášený uživatel administrátor, tak se mu zobrazuje i spodní část menu, ve které jsou položky: **Společnosti**, **Projekty** a **Uživatelé**. Po rozkliknutí jedné z položek vidíme vždy příslušnou tabulku záznamů. Tabulky mají v aplikaci jednotný design (obr. 13).



Obrázek 13: UI – tabulka uživatelů (Zdroj: autor)

Každý záznam si můžeme rozkliknout pro detailní zobrazení, či využít akčních tlačítek v pravé části pro smazání či editaci záznamu. Pokud je záznamů hodně, můžeme přecházet mezi jednotlivými stránkami pomocí panelu umístěného uprostřed pod tabulkou. V pravém dolním rohu můžeme změnit počet záznamů na stránce. Záznamy lze také libovolně filtrovat pomocí filtrů nad tabulkou. V pravém horním rohu máme tlačítko + pro přidání nového záznamu do tabulky a nad ním je tlačítko pro odhlášení uživatele.

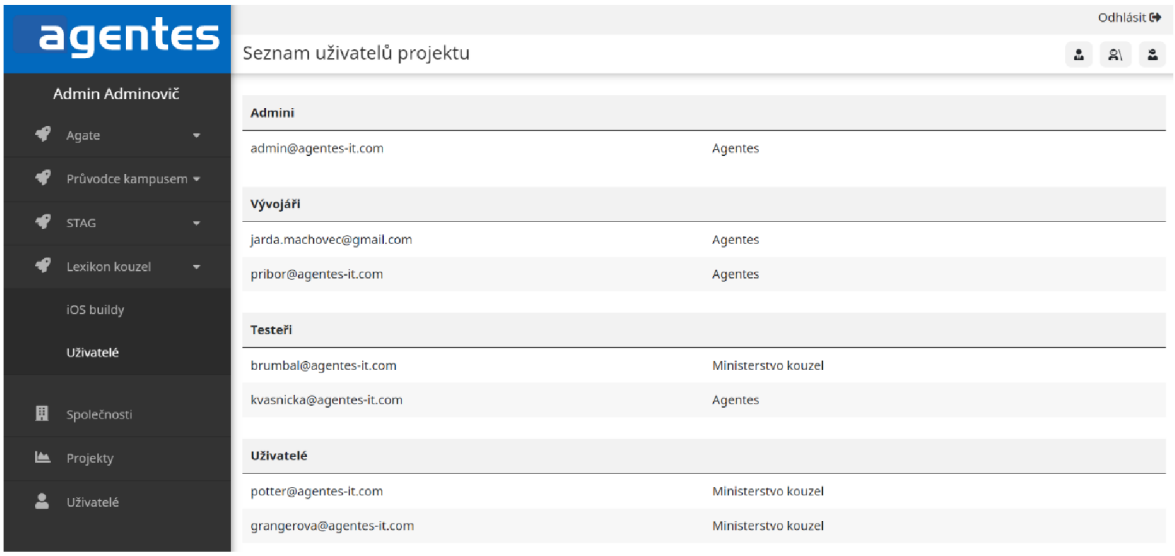
Vytváření záznamů a jejich editace probíhá na nové obrazovce. Pokud uživatel nevyplní všechny potřebné informace či některé pole nevyplní správně, tak se formulář neodešle a zobrazí se mu chybová hláška u problematických polí (obr. 14). Na obrázku vidíme příklad zakládání projektu.



Obrázek 14: UI - zakládání projektu (Zdroj: autor)

Pokud v menu rozklikneme některý z projektů, tak se nám otevře malé podmenu. Můžeme v něm nalézt položky **Android buildy**, **iOS buildy** a **Uživatelé**. Zobrazení položek **Android buildy** a **iOS buildy** závisí na tom, zda je příslušná platforma u daného projektu povolena. Pokud Android buildy či iOS buildy roklikneme, zobrazí se nám tabulka buildů.

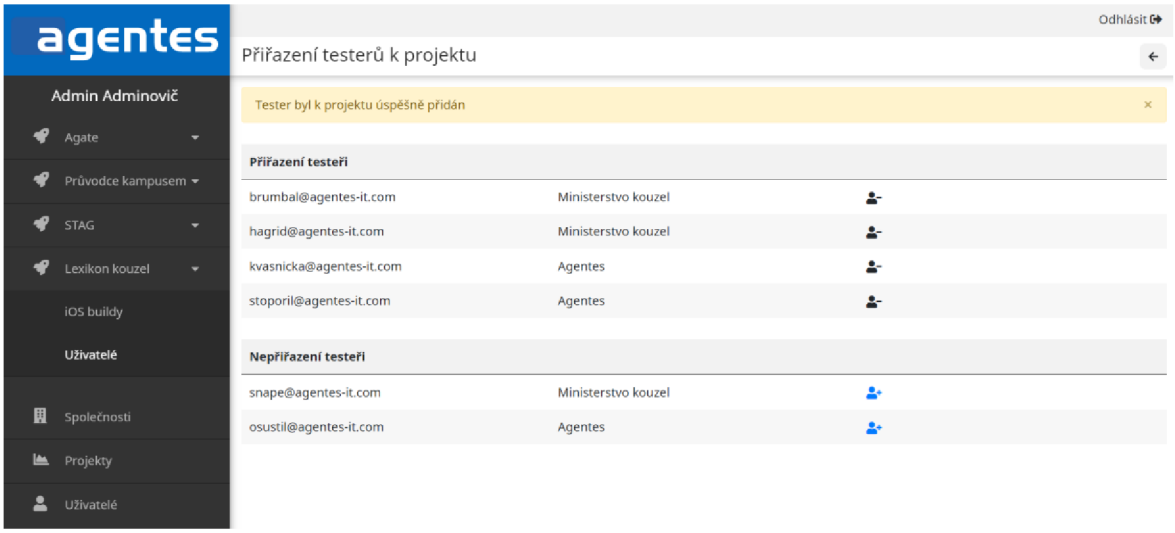
Položka **Uživatelé** v podmenu projektu nám zobrazí seznam uživatelů projektu (obr. 15). Vidíme zde 4 kategorie. **Admini**, **Vývojáři**, **Testeři** a **Uživatelé**. Administrátoři mají přístup ke všem projektům, a tak je k nim nemusíme přiřazovat. V ostatních kategoriích jsou vypsání pouze uživatelé, kteří byli k danému projektu přiřazeni. Kategorie **Testeři** zahrnuje



Obrázek 15: UI - seznam uživatelů projektu (Zdroj: autor)

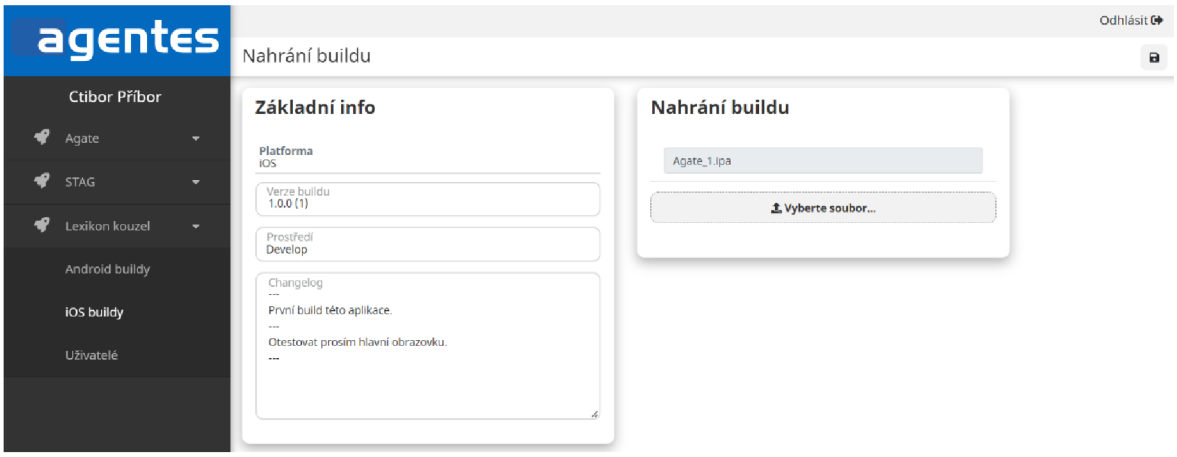
přiřazené testery administrátorské společnosti, kteří mají roli SUPERTESTER, ale také testery společnosti, pro kterou je projekt vyvíjen. Ti mají roli TESTER. Role přihlášeného uživatele určuje, které z těchto kategorií uvidí. ADMIN, DEVELOPER a SUPERTESTER uvidí všechny kategorie. TESTER uvidí pouze uživatele ze své společnosti – tzn. běžné uživatele a testery s rolí TESTER. Administrátor také v pravém horním rohu vidí tři ikonky pro přiřazení vývojářů, testerů i uživatelů.

Po kliknutí se administrátorovi otevře obrazovka pro přiřazení uživatelů dané kategorie (obr. 16). Na obrazovce vidíme dva seznamy - přiřazených testerů a těch, které je možné přiřadit. U každého z nich máme ikonku, pomocí které ho přiřadíme nebo odebereme. Pokud odebrání či přidání proběhlo úspěšně, tak se nám ukáže informační hláška. Tyto hlášky se zobrazují i při úspěšné editaci, vytvoření či smazání záznamu v ostatních tabulkách v administraci.



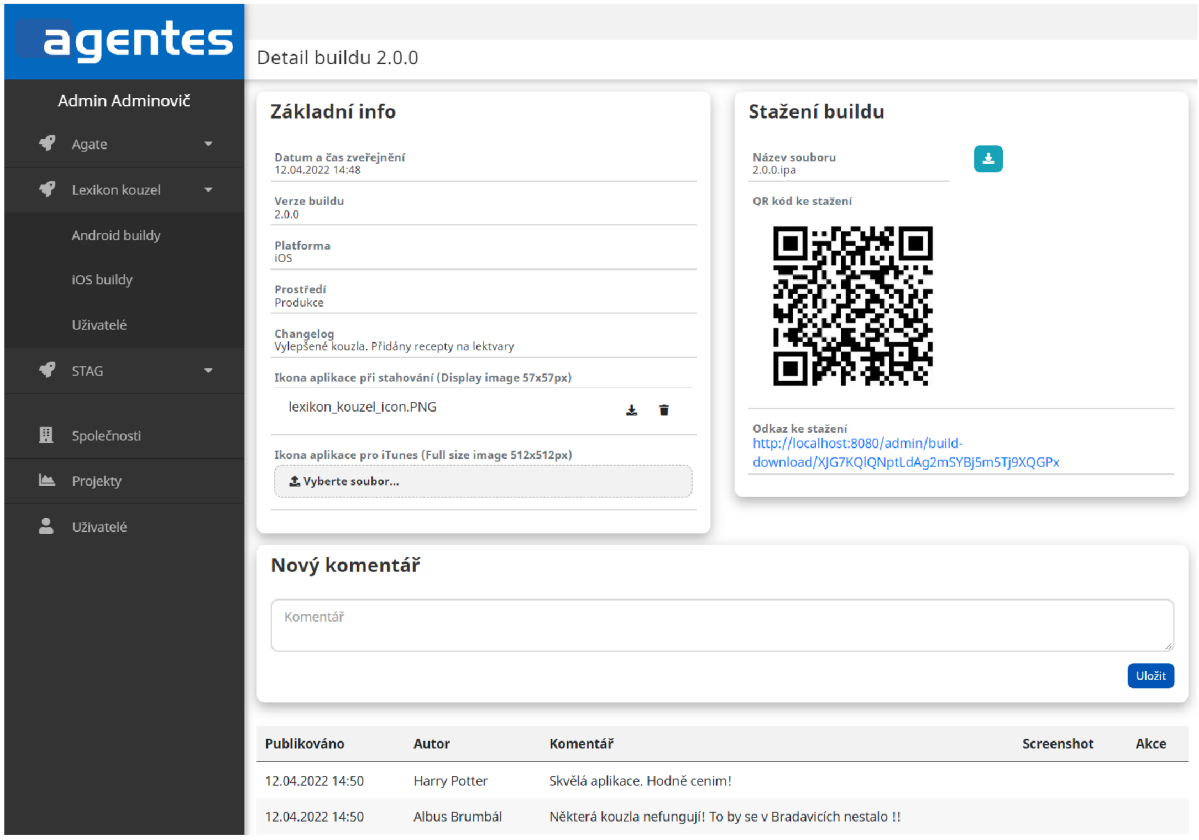
Obrázek 16: UI - přiřazení testerů k projektu (Zdroj: autor)

Pojďme se nyní přesunout ze seznamu uživatelů projektu k buildům. Pokud se vývojář rozhodne nahrát nový build aplikace, tak se mu zobrazí tento formulář (obr. 17). Soubor APK či IPA lze jednoduše nahrát přetáhnutím na tlačítko inputu a po vyplnění informací uložit pomocí ikony diskety vpravo nahoře.



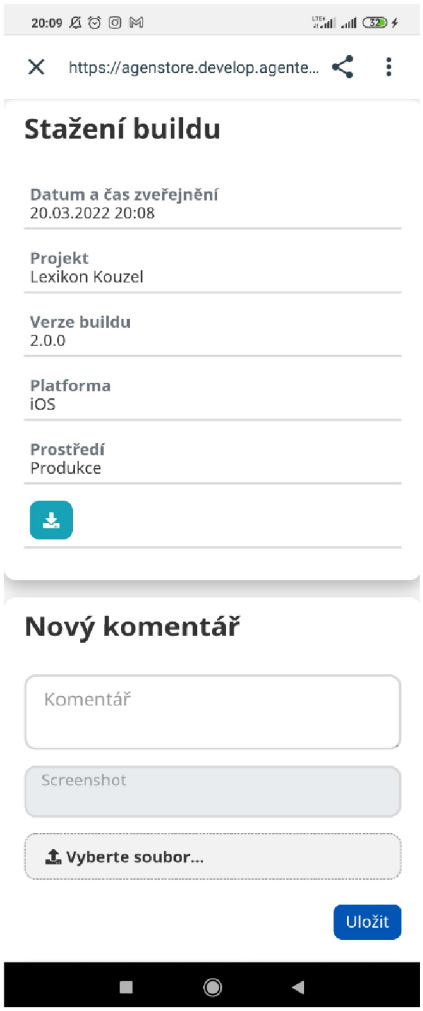
Obrázek 18: UI - nahrání buildu (Zdroj: autor)

Jestliže si některý build v tabulce rozklikneme, zobrazí se nám jeho detail (obr. 18). Pokud se jedná o iOS build, tak zde můžeme nahrát ikonky aplikace, což nám při vytváření buildu umožněno nebylo. Ve spodní části můžeme k buildu napsat komentář a jejich seznam se zobrazuje hned pod tím. Autor zde svůj komentář může smazat. Také může rozkliknutím komentáře zobrazit detail, kde lze nahrát ke komentáři screenshot. Nahrání screenshotu může provést pouze autor. Ostatní, pokud komentář rozkliknou, zde nahraný screenshot uvidí.



Obrázek 17: UI - detail buildu (Zdroj: autor)

V pravé části detailu buildu vidíme QR kód a textový odkaz. Pokud si QR kód naskenujeme mobilem, tak budeme přesměrováni na stránku, odkud lze build stáhnout (obr. 19). Textový odkaz vede na stejnou stránku. Uživatel zde vidí základní informace o buildu a



Obrázek 19: UI - stažení buildu (Zdroj: autor)

samozřejmě tlačítko ke stažení. K zobrazení této stránky může být v mobilním zařízení vyžadováno přihlášení pro zajištění větší bezpečnosti. Záleží to na nastaveném typu zabezpečení projektu a na typu prostředí, pro které je daný build určen. Nechybí zde ani kolonka pro odeslání komentáře a lze přiložit i screenshot obrazovky, aby mohli odeslat zpětnou vazbu u veřejných aplikací i uživatelé, kteří nemají v administraci svůj účet.

4.2 Soubor PLIST

Nejvíce specifickým, ale naprosto klíčovým úkolem, který naše aplikace musí zvládnout, je umožnění instalace iOS buildů. Z pohledu autora se jedná o nejzajímavější část aplikace. Během analýzy jsme zjistili, že k instalaci aplikace do mobilního zařízení nám u iOS

nestačí samotný IPA soubor. Potřebujeme k tomu ještě správně vyplněný soubor typu PLIST. Teoreticky bychom si ho mohli nechat vygenerovat ve vývojovém prostředí xCode, avšak soubor musí obsahovat url adresu, na které bude přístupný soubor IPA. Tuto informaci v době exportování archivu ještě nemáme, proto jsme zvolili jinou možnost. Poté, co uživatel nahraje IPA soubor, tak si PLIST automaticky vygenerujeme a vyplníme správnými údaji sami. Vývojáři tedy stačí nahrát pouze IPA soubor, vyplnit číslo verze, changelog a zvolit prostředí, pro které je daný build určen, stejně jako u buildů pro Android.

Struktura PLISTu

Abychom byli schopni PLIST vygenerovat, musíme se nejprve podívat na jeho strukturu a zjistit, co přesně musí obsahovat. Jedná se o textový soubor strukturovaný pomocí XML a může obsahovat spoustu údajů o instalovaném souboru. Nám bude stačit základní struktura PLISTu, která obsahuje pouze atributy, které jsou povinné (obr. 20). [80]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
  <dict>
      <key>items</key>
      <array>
          <dict>
              <key>assets</key>
              <array>
                  <dict>
                      <key>kind</key>
                      <string>software-package</string>
                      <key>url</key>
                      <string>http://localhost:8080/admin/build-download/rWCqkBEta7KAPWTIx55UbVLtACV8sJov/ios</string>
                  </dict>
                  <dict>
                      <key>kind</key>
                      <string>display-image</string>
                      <key>url</key>
                      <stringhttp://localhost:8080/admin/build-download/rWCqkBEta7KAPWTIx55UbVLtACV8sJov/display-image</string>
                  </dict>
                  <dict>
                      <key>kind</key>
                      <string>full-size-image</string>
                      <key>url</key>
                      <string>http://localhost:8080/admin/build-download/rWCqkBEta7KAPWTIx55UbVLtACV8sJov/full-size-image</string>
                  </dict>
              </array>
              <key>metadata</key>
              <dict>
                  <key>bundle-identifier</key>
                  <string>cz.agentes.Agate2</string>
                  <key>bundle-version</key>
                  <string>1</string>
                  <key>kind</key>
                  <string>software</string>
                  <key>title</key>
                  <string>Agate</string>
              </dict>
          </dict>
      </array>
  </dict>
</plist>
```

Obrázek 20: Příklad struktury souboru PLIST (Zdroj: autor)

Vidíme, že v souboru se nachází dvě hlavní části. V první části jsou definována aktiva a v druhé metadata. Co se týče aktiv, tak zde máme 3 položky představující soubory. U každé z nich musíme vyplnit url, na které je daný soubor přístupný k získání. *Software-package* je

náš instalační IPA soubor. *Display-image* je ikonka naší aplikace velikosti 57x57px používaná během stahování a instalace. *Full-size-image* je velká ikonka 512x512px zobrazovaná v iTunes. Důležitá je především cesta k IPA souboru. Obrázky nejsou k funkčnímu stahování povinné, a tak příslušné hodnoty v PLISTU můžeme nechat prázdné a případně je vyplnit až později. V druhé části, kde jsou definovány metadata, máme kromě atributu *kind*, který zůstává stále stejný, 3 další atributy. *Bundle-identifier* je jedinečný identifikátor aplikace v rámci Apple ekosystému. *Bundle-version* je číslo verze a *title* je název aplikace.

Získání správných hodnot

Hodnoty těchto atributů lze získat přímo z IPA souboru, který funguje jako kontejner, a můžeme jej tudíž rozbalit jako klasický ZIP soubor. Uvnitř jeho struktury lze nalézt soubor Info.plist, který obsahuje mnoho atributů a mezi nimi i ty, které potřebujeme. Pojmenovány jsou však rozdílně než v PLIST souboru, který vytváříme. Naším požadovaným názvům odpovídají takto:

- CFBundleIdentifier = bundle-identifier
- CFBundleVersion = bundle-version
- CFBundleName = title

V průběhu ukládání buildu tedy nejprve uložíme soubor IPA na disk. Poté přejdeme k vygenerování PLISTu. Tato operace se skládá z několika částí. Nejprve si najdeme uložený IPA soubor na disku. Ten poté rozbalíme pomocí metody *unzipPlist (obr. 21)* do cílové složky. Struktura IPA souboru je velice rozsáhlá. Metodu unzipPlist jsme tedy optimalizovali tak, aby

```
public void unzipPlist(String zipFilePath, String destDirectory) throws IOException {
    File destDir = new File(destDirectory);

    if (!destDir.exists()) {
        destDir.mkdir();
    }
    ZipInputStream zipIn = new ZipInputStream(new FileInputStream(zipFilePath));
    ZipEntry entry = zipIn.getNextEntry();

    int stepNumber = 1;
    String plistSuffix = null;
    // iterates over entries in the zip file
    while (entry != null) {
        boolean unzip = false;
        String filePath = destDirectory + File.separator + entry.getName();

        if (stepNumber < 3) {
            if (stepNumber == 2) {
                plistSuffix = entry.getName() + "Info.plist";
            }
            stepNumber ++;
            unzip = true;
        } else if (stepNumber == 3) {
            if (entry.getName().endsWith(plistSuffix)) {
                stepNumber ++;
                unzip = true;
            }
        }

        if (unzip) {
            if (!entry.isDirectory()) {
                // if the entry is a file, extracts it
                extractFile(zipIn, filePath);
            } else {
                // if the entry is a directory, make the directory
                File dir = new File(filePath);
                dir.mkdir();
            }
        }

        zipIn.closeEntry();
        entry = zipIn.getNextEntry();
    }

    zipIn.close();
}
```

Obrázek 21: Rozbalení souboru PLIST (Zdroj: autor)

v několika krocích našla a rozbalila pouze adresáře po cestě k požadovanému souboru a soubor samotný.

Poté, co máme soubor rozbalený, tak z něj parsováním pomocí knihovny *dd-plist* získáme potřebné hodnoty (obr. 22).

```
// parse Info.plist
File plistFile = new File(plistFilePath);

Map<String, String> ipaInfo = new HashMap<>();
try {
    NSDictionary rootDict = (NSDictionary) PropertyListParser.parse(plistFile);

    // get bundle id
    NSString parameter = (NSString) rootDict.objectForKey("CFBundleIdentifier");
    ipaInfo.put("CFBundleIdentifier", parameter.toString());

    // get application name
    parameter = (NSString) rootDict.objectForKey("CFBundleName");
    ipaInfo.put("CFBundleName", parameter.toString());

    // get version
    parameter = (NSString) rootDict.objectForKey("CFBundleVersion");
    ipaInfo.put("CFBundleVersion", parameter.toString());

} catch (IOException | SAXException | ParserConfigurationException | ParseException | PropertyListFormatException e) {
    e.printStackTrace();
}
```

Obrázek 22: Získání hodnot ze souboru PLIST (Zdroj: autor)

Vygenerování nového PLISTu

Tyto získané hodnoty poté společně s url adresou k instalačnímu souboru předáme metodě *toPlist* (obr. 23), která vytvoří celou textovou strukturu souboru. Následně už akorát vytvoříme soubor s tímto obsahem, uložíme jej na disk a záznam o něm vložíme do databáze. Cestu k tomuto souboru poté vložíme v html šabloně do speciálního linku, který jsme si již ukazovali v počáteční analýze (obr. 1). V případě, že k buildu později nahrajeme obrázky ikonek *display-image* nebo *full-size-image*, tak se stávající plist smaže a vytvoří se celý odznova s rozdílem, že při metodě *toPlist* se předají i url adresy k těmto obrázkům.

```
private static String toPlist(String buildUrl, String displayImageUrl, String fullSizeImageUrl, String bundleIdentifier, String bundleVersion, String bundleTitle) {
    String s = "";
    s += "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n";
    s += "<!DOCTYPE plist PUBLIC \"-//Apple Computer//DTD PLIST 1.0//EN\" \"http://www.apple.com/DTDs/PropertyList-1.0.dtd\">\n";
    s += "<plist version=\"1.0\">\n";
    s += "  <dict>\n";
    s += "      <key>items</key>\n";
    s += "      <array>\n";
    s += "          <dict>\n";
    s += "              <key>assets</key>\n";
    s += "              <array>\n";
    s += "                  <dict>\n";
    s += "                      <key>kind</key>\n";
    s += "                      <string>software-package</string>\n";
    s += "                      <key>url</key>\n";
    s += "                      <string>" + buildUrl + "</string>\n";
    s += "                  </dict>\n";
    s += "                  <dict>\n";
    s += "                      <key>kind</key>\n";
    s += "                      <string>display-image</string>\n";
    s += "                      <key>url</key>\n";
    s += "                      <string>" + displayImageUrl + "</string>\n";
    s += "                  </dict>\n";
    s += "                  <dict>\n";
    s += "                      <key>kind</key>\n";
    s += "                      <string>full-size-image</string>\n";
    s += "                      <key>url</key>\n";
    s += "                      <string>" + fullSizeImageUrl + "</string>\n";
    s += "                  </dict>\n";
    s += "              </array>\n";
    s += "              <key>metadata</key>\n";
    s += "              <dict>\n";
    s += "                  <key>bundle-identifier</key>\n";
    s += "                  <string>" + bundleIdentifier + "</string>\n";
    s += "                  <key>bundle-version</key>\n";
    s += "                  <string>" + bundleVersion + "</string>\n";
    s += "                  <key>kind</key>\n";
    s += "                  <string>software</string>\n";
    s += "                  <key>title</key>\n";
    s += "                  <string>" + bundleTitle + "</string>\n";
    s += "              </dict>\n";
    s += "          </dict>\n";
    s += "      </array>\n";
    s += "  </dict>\n";
    s += "</plist>";

    return s;
}
}
```

Obrázek 23: Vytvoření textové struktury PLISTu (Zdroj: autor)

Zabezpečení

Pokud daný build ke stažení nevyžaduje přihlášení, tak toto řešení funguje v pořádku. Během implementace jsme však narazili na problém, který nastal, když bylo potřeba zkontrolovat, zda má přihlášený uživatel oprávnění k danému buildu. Tím, že je URL adresa IPA souboru volána po stažení PLISTu již mimo prohlížeč, tak nejsme schopni podle *session* zjistit, který uživatel je přihlášen. Stejný problém nastal i při stahování souboru PLIST pomocí speciálního linku, kdy jsme také nebyli schopni zjistit přihlášeného uživatele.

Problém jsme tedy vyřešili oproti původnímu návrhu dodatečným přidáním tabulky ***download_token*** do databáze, kam ukládáme bezpečnostní tokeny. Ve chvíli, kdy uživatel zobrazí stránku, odkud se build stahuje, tak zkontrolujeme, zda má oprávnění build stáhnout. Pokud ano, tak vygenerujeme nový token určený pro soubor PLIST a uložíme ho do databáze. Tento token vložíme také v html šabloně do speciálního linku ke stažení souboru PLIST. Poté, co uživatel klikne na tlačítko stáhnout, tak se zavolá url obsahující tento token a na backendu zkontrolujeme, zda je token validní. Pokud ano, tak vygenerujeme nový token, tentokrát určený pro IPA soubor, a uložíme jej do databáze. Také vygenerujeme nový dočasný soubor PLIST, do kterého vložíme url k získání IPA souboru obsahujícího tento nový token. Následně se tedy stáhne soubor PLIST a to spustí stažení souboru IPA. Opět je na backendu zkontrolováno, zda je daný token validní a pokud ano, tak dojde ke stažení a instalaci.

Tokeny mají nastavený čas expirace. Ty, které byly vytvořeny, ale nebyly použity a skončila jim platnost, jsou pomocí cronu pravidelně mazány. Soubor PLIST, který jsme vytvořili a dočasně uložili na disk, je také pomocí cronu smazán.

5 Ostrý provoz a změny

5.1 Nasazení aplikace do ostrého provozu

Všechny stanovené funkcionality se podařilo naimplementovat a aplikace byla ve firmě školitele nasazena do reálného pilotního provozu. Nyní je potřeba vyhodnotit, zda jsou s tímto řešením spokojeni i samotní uživatelé. Abychom od nich získali zpětnou vazbu, rozešleme jim dotazník spokojenosti.

5.2 Dotazník spokojenosti

Hlavním cílem tohoto dotazníku je zjistit spokojenost uživatelů a identifikovat případné nedostatky. Důvodů k nespokojenosti může být více. Uživatelům nemusí vyhovovat členění uživatelského rozhraní, mohou postrádat některou funkcionalitu nebo například naleznou některé chyby ve funkčnosti, které se nám nepodařilo odhalit během průběžného testování. Pokud uživatelé nejsou s něčím spokojeni, musíme zjistit konkrétní důvod, abychom na to mohli zareagovat úpravou aplikace.

5.2.1 Struktura dotazníku

V první části dotazníku jsme se nejprve uživatelů dotázali, jaká je jejich role. Poté jsme přešli k ohodnocení přehlednosti aplikace, zeptali jsme se, zda je napadá, jak přehlednost vylepšit a v otevřené odpovědi mohli tyto nápady uvést.

V další části jsme se dotazovali, zda aplikace obsahuje všechny funkcionality, které od ní vyžadují, a zda by některé nové či úpravu stávajících uvítali. Opět zde mohli nápady uvést v otevřené odpovědi.

Dalším aspektem, na který jsme se ptali, bylo, jestli uživatelé v aplikaci nalezli nějakou chybu a pokud ano, tak aby ji uvedli.

Na závěr jsme se dotázali na zhodnocení aplikace jako celku a nechali prostor uživateli k uvedení kteréhokoliv zlepšení, které nebylo zmíněno v předchozích otázkách.

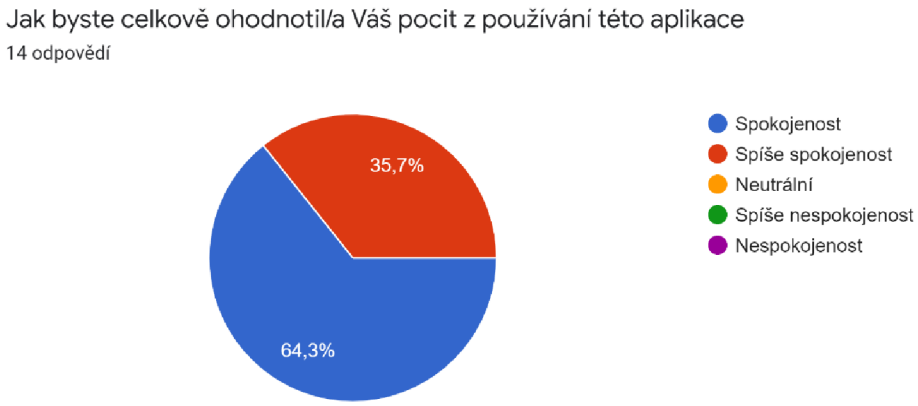
5.2.2 Výsledky dotazníku

Dotazníku se zúčastnilo celkem 14 uživatelů. Z toho 2 administrátoři, 3 vývojáři, 3 testeři a 6 běžných uživatelů. Počet respondentů je adekvátní velikosti firmy a tomu, že se zatím jedná pouze o pilotní provoz.

Z hlediska získání podnětů ke zlepšení aplikace a odhalení chyb dotazník splnil svůj účel. Dostali jsme 4 návrhy na zlepšení přehlednosti aplikace. Jednalo se zejména o přidání popisů jednotlivých rolí, aby bylo zřejmé, jaká mají oprávnění. Dále přesun některých tlačítek či zakrytí stránkování v případech, kdy není potřeba. Ohledně nových funkcionalit jsme dostali 2 návrhy, které oba zmiňovaly chybějící možnost upozornit emailem přiřazené uživatele na nově nahraný build.

Čtyři uživatelé nám také nahlásili, že objevili chybu. Ve třech případech se jednalo o stejnou chybu, kterou byl problém s přihlášením kvůli špatně zadanému emailu v administraci. V posledním případě šlo o nefungující tlačítko pro zobrazení skrytého menu.

I přes tyto podněty a odhalené chyby, které budou zahrnuty do finální úpravy aplikace, ohodnotilo svůj celkový pocit z používání aplikace **9 uživatelů** výběrem možnosti **spokojenost** a **5 uživatelů** možností **spíše spokojenost**. Negativní recenze jsme nezískali žádné. Můžeme tedy konstatovat, že nově implementované řešení hodnotí uživatelé pozitivně.



Obrázek 24: Graf spokojenosti uživatelů (Zdroj: autor)

Kompletní znění otázek a odpovědí je uvedeno v příloze ***Dotazníkové_šetření***.

5.2.3 Provedené změny

Všechny podněty k vylepšení aplikace, které uživatelé uvedli v dotazníkovém šetření, jsme do aplikace doimplementovali. Chyby, které se uživatelům podařilo odhalit, jsme opravili. Zde je uveden kompletní seznam oprav a provedených změn.

Seznam oprav:

- V případě, že byl při založení účtu uživateli vyplněn email a za ním mezera, tak se tato mezera uložila také. Uživatelům se kvůli tomu poté nešlo přihlásit, jelikož email nesouhlasil. Opraveno funkcí trim(), která se použije na vyplněný email při zakládání uživatele.

- Při zmenšení okna prohlížeče se menu schovává, aby zbylo stále dost místa na obsah stránky. Kliknutím na ikonu v levém horním rohu lze menu v případě potřeby zobrazit. Uživatelé odhalili, že tato ikona nefungovala a po kliknutí se menu nezobrazilo. Provedli jsme tedy opravu.

Seznam změn:

- Možnost rozeslání notifikačního emailu přiřazeným uživatelům při nahrání nového buildu.

- Skrytí stránkování, pokud není počet stránek větší než 1.

- Přesunutí tlačítek pro přidání uživatelů, testerů, vývojářů z horního rohu přímo k daným sekcím.

- Přidání tooltipů k jednotlivým rolím v přehledu uživatelů projektu, aby bylo vysvětleno, jaká práva daná role má.

- Přidání popisu rolí na obrazovce vytváření a editace uživatele pro vysvětlení, jaká práva daná role má.

- Zobrazení hlášky, že pokud se po instalaci aplikace sama nespustí, uživatel ji nalezne na ploše (aby si nemyslel, že se instalace nezdařila).

5.3 Plánovaný rozvoj

Během pilotního provozu byla aplikace otestována na aktuálně vyvíjených projektech. S tím, jak budou nové projekty přibývat, bude narůstat i počet uživatelů využívajících naši aplikaci. Lze tedy očekávat, že noví uživatelé budou přicházet s dalšími nápady, jak aplikaci vylepšit.

Z hlediska klíčových funkcionalit by jedním z možných vylepšení bylo zajištění automatických aktualizací pro námi distribuované mobilní aplikace.

6 Závěr

Hlavním cílem práce bylo vyvinout webovou aplikaci, kterou bude firma Agentes IT využívat pro distribuci testovacích i produkčních verzí mobilních aplikací pro Android i iOS. Motivací pro vývoj vlastního řešení namísto oficiálních obchodů Google Play a App Store byl zejména požadavek vyhnout se kontrolám nahraných buildů, které bývají zdlouhavé, a tím zpomalují proces testování a vývoje. Dalším důležitým důvodem bylo, že neveřejné aplikace, vyvíjené na zakázku pro třetí strany, sice lze přes Google Play a App Store distribuovat, ale má to úskalí. Společnost si na dané platformě aplikaci zakoupí, ale samotná instalace na jednotlivá zařízení zaměstnanců probíhá centrálně pomocí nástroje pro správu mobilních zařízení (MDM či EMM). Mnoho společností však takový nástroj nepoužívá a je tedy potřeba najít cestu, jak jim aplikace distribuovat jiným způsobem.

V první fázi jsme provedli analýzu instalačních souborů u Androidu i iOS, abychom zjistili, jakým způsobem s nimi můžeme pracovat a jaká mají omezení. Následně jsme provedli rešerši současných řešení, zvolili jsme vhodná kritéria a provedli jejich porovnání. Na základě této analýzy a požadavků firmy Agentes IT jsme poté specifikovali požadavky na námi tvořenou aplikaci.

V další fázi jsme vytvořili návrh. Určili jsme scénáře užití a provedli jsme také výběr technologického stacku. Navržena byla i architektura celé aplikace a datový model. Následoval samotný vývoj aplikace.

Aplikace byla nakonec spuštěna v reálném pilotním provozu. Pomocí dotazníkového šetření jsme provedli průzkum spokojenosti a získali zpětnou vazbu od uživatelů. Uživatelé hodnotili nové řešení pozitivně, ale získali jsme od nich i mnoho podnětů k vylepšení aplikace. Všechny tyto podněty se nám podařilo vyslyšet a změny naimplementovat. Uživatelům se během pilotního provozu podařilo odhalit také některé drobné chyby, které jsme následně opravili.

Výstupem bakalářské práce je funkční řešení pro distribuci testovacích i produkčních verzí mobilních aplikací pro Android i iOS. Firmě Agentes IT toto řešení přináší zejména možnost rychlejšího a efektivnějšího testování díky tomu, že nahrané buildy nepodléhají kontrole a jsou ihned k dispozici. Testování také usnadňuje možnost zobrazení historie verzí,

stažení kterékoliv z nich a možnost přidání zpětné vazby. Dalším důležitým přínosem je, že firma Agentes IT přes vyvinuté řešení může distribuovat soukromé aplikace, vyvíjené pro třetí strany, i společnostem, které nepoužívají nástroj pro správu mobilních zařízení svých zaměstnanců.

Možným vylepšením do budoucna by bylo zajištění automatických aktualizací mobilních aplikací, které distribuujeme.

Projekt je verzován pomocí Gitu a nahrán do GitLabu, kde je zpřístupněn firmě Agentes IT. Snadno lze tedy pokračovat v implementaci nových funkcionalit a budoucím rozvoji aplikace.

Nakonec zbývá konstatovat, že se nám podařilo splnit všechny cíle bakalářské práce, které jsme si stanovili.

Seznam literatury

[1] Mobile Operating System Market Share Worldwide. *Statcounter Global Stats* [online]. Dublin: StatCounter, c1999–2022 [cit. 2022-02-28]. Dostupné z: https://gs.statcounter.com/os-market-share/mobile/worldwide

[2] About Android App Bundles. *Android Developers* [online]. Mountain View: Google, 2022 [cit. 2022-04-13]. Dostupné z: https://developer.android.com/guide/app-bundle

[3] Alternative distribution options. *Android Developers* [online]. Mountain View: Google, 2020 [cit. 2022-04-13]. Dostupné z: https://developer.android.com/distribute/marketing-tools/alternative-distribution#unknown-sources

[4] Choosing a Membership. *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/support/compare-memberships/

[5] Apple Developer Enterprise Program. *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/programs/enterprise/

[6] Distribution methods. *Xcode Help* [online]. Cork: Apple, c2020 [cit. 2022-04-13]. Dostupné z: https://help.apple.com/xcode/mac/current/#/dev31de635e5

[7] Distribuce vlastních interních aplikací do zařízení Apple. *Apple Support* [online]. Cork: Apple, 2021 [cit. 2022-04-13]. Dostupné z: https://support.apple.com/cs-cz/guide/deployment/depce7cefc4d/web

[8] Add developer account users and manage permissions. *Play Console Help* [online]. Mountain View: Google, c2022 [cit. 2022-04-13]. Dostupné z: https://support.google.com/googleplay/android-developer/answer/9844686?hl=en

[9] Nastavení otevřeného, uzavřeného nebo interního testování. *Nápověda Play Console* [online]. Mountain View: Google, c2022 [cit. 2022-04-13]. Dostupné z: https://support.google.com/googleplay/android-developer/answer/9845334

[10] Publish private apps from the Play Console. *Managed Google Play Help* [online]. Mountain View: Google, c2022 [cit. 2022-04-13]. Dostupné z: https://support.google.com/googleplay/work/answer/6145139#zippy=%2Callow-rd-party-developer-publishing%2Cpublish-to-your-own-organization

[11] Program Roles. *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/support/roles/

[12] Beta Testing Made Simple with TestFlight. *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/testflight/

[13] App Review. *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/app-store/review/

[14] Custom app distribution with Apple Business Manager. *Apple Developer* [online]. Cork: Apple, 2020 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/videos/play/wwdc2020/10667/

[15] Applivery SDK. *Docs | Applivery* [online]. Madrid: Applivery, c2021 [cit. 2022-04-13]. Dostupné z: https://www.applivery.com/docs/mobile-app-distribution/sdk/

[16] Plans & Pricing. *Applivery | MDM, MAM & App Distribution Platform* [online]. Madrid: Applivery, c2021 [cit. 2022-04-13]. Dostupné z: https://www.applivery.com/pricing/

[17] HEROUT, Pavel. *Učebnice jazyka Java*. 5., rozš. vyd. České Budějovice: Kopp, 2010. ISBN 978-80-7232-398-2.

[18] *Spring | Home* [online]. Palo Alto: VMware, c2022 [cit. 2022-04-13]. Dostupné z: https://spring.io/

[19] Spring Boot. *Spring | Home* [online]. Palo Alto: VMware, c2022 [cit. 2022-04-13]. Dostupné z: https://spring.io/projects/spring-boot

[20] *PostgreSQL: The world's most advanced open source database* [online]. Santa Barbara: The PostgreSQL Global Development Group, c1996-2022 [cit. 2022-04-13]. Dostupné z: https://www.postgresql.org/

[21] *Maven – Welcome to Apache Maven* [online]. Wilmington: The Apache Software Foundation, c2002–2022 [cit. 2022-04-13]. Dostupné z: https://maven.apache.org/

[22] *Thymeleaf* [online]. The Thymeleaf Team [cit. 2022-04-13]. Dostupné z: https://www.thymeleaf.org/

[23] *JQuery* [online]. San Francisco: OpenJS Foundation, c2022 [cit. 2022-04-13]. Dostupné z: https://jquery.com/

[24] *Bootstrap · The most popular HTML, CSS, and JS library in the world.* [online]. San Francisco: Bootstrap team [cit. 2022-04-13]. Dostupné z: https://getbootstrap.com/

[25] *Liquibase | Open Source Version Control for Your Database* [online]. Austin: Liquibase, c2022 [cit. 2022-04-13]. Dostupné z: https://www.liquibase.org/

[26] *Git* [online]. [cit. 2022-04-13]. Dostupné z: https://git-scm.com/

[27] *Iterate faster, innovate together | GitLab* [online]. Amsterdam: GitLab, c2022 [cit. 2022-04-13]. Dostupné z: https://about.gitlab.com/

[28] *Home - Docker* [online]. Palo Alto: Docker, c2022 [cit. 2022-04-13]. Dostupné z: https://www.docker.com/

[29] *Kubernetes* [online]. The Kubernetes Authors, c2022 [cit. 2022-04-13]. Dostupné z: https://kubernetes.io/

[80] How to do Over The Air (OTA) distribution? (This tutorial is written for Adhoc Version only). In: *MyAcademic* [online]. 2019 [cit. 2022-04-13]. Dostupné z: https://muaca.blogspot.com/2019/10/how-to-do-over-air-ota-distribution.html

[82] *Android Developers* [online]. Mountain View: Google, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.android.com/

[83] *Xcode Help* [online]. Cork: Apple, c2020 [cit. 2022-04-13]. Dostupné z: https://help.apple.com/xcode/mac/current/#/

[84] *Apple Developer* [online]. Cork: Apple, c2022 [cit. 2022-04-13]. Dostupné z: https://developer.apple.com/

[85] *Docs | Applivery* [online]. Madrid: Applivery, c2021 [cit. 2022-04-13]. Dostupné z: https://www.applivery.com/docs/

[86] *Applivery | MDM, MAM & App Distribution Platform* [online]. Madrid: Applivery, c2021 [cit. 2022-04-13]. Dostupné z: https://www.applivery.com/

Seznam obrázků

Seznam příloh

1. Složka *agenstore* – naprogramovaná aplikace (IntelliJ IDEA projekt)
2. Soubor *Dotazníkové_šetření* – kompletní výsledky dotazníkového šetření