FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Android aplikace pro majitele psů
Student:	Bc. Alina Lagoda
Vedoucí:	Ing. Jiří Hunka
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 diplomové práce je navrhnout a implementovat mobilní aplikaci v OS Android pro majitele psů. Aplikace bude umožňovat sdílení fotografií a událostí, pořizování záznamu trasy procházky se psem a hledání lidí pro společné venčení.

Postupujte dle následujících kroků:
- Analyzujte požadavky cílové skupiny uživatelů.
- Proveďte analýzu alespoň 4 sociálních sítí zaměřených na podobnou tematiku.
- Navrhněte vhodný koncept aplikace na základě předešlých analýz a požadavků zadání.
- Navrhněte vhodné uživatelské rozhraní, vytvořte wireframy aplikace.
- Implementujte funkční prototyp klientské části aplikace.
- Navrhněte a implementujte serverovou část aplikace.
- Android aplikaci včetně serverové části podrobte vhodným testům.
- Zjištěné nedostatky napravte.
- Zhodnoťte přínos aplikace a použitelnost prototypu.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

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

V Praze dne 1. listopadu 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Android aplikace pro majitele psů

Bc. Alina Lagoda

Katedra softwarového inženýrství
Vedoucí práce: Ing. Jiří Hunka

5. ledna 2019

Poděkování

Ráda bych poděkovala vedoucímu práce Ing. Jiřímu Hunkovi za odborné vedení a cenné rady při zpracování práce. Kromě toho bych chtěla poděkovat rodině a přátelům za jejich podporu a trpělivost v průběhu studia.

Prohlášení

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

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

V Praze dne 5. ledna 2019 .....

Odkaz na tuto práci

Lagoda, Alina. *Android aplikace pro majitele psů*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Tato diplomová práce se zabývá kompletním procesem vývoje aplikace skládající se z nativní Android aplikace a serverové části. Aplikace je určena pro majitele psů a zaměřuje se na sociální funkce, jako jsou vedení stránek svých domácích mazlíčků, sdílení fotografií a pořizování záznamu trasy procházky se psem. Součástí práce je analýza existujících řešení, návrh, implementace a testování aplikace. Výsledkem práce je funkční prototyp mobilní aplikace pro majitele psů.

Klíčová slova mobilní aplikace, psi, Android, Kotlin, sociální síť, venčení psů

Abstract

This master's thesis is about complete process of developing system that consists of native Android application and server part. The application is designed for dog owners and focuses on social features such as managing profile pages, sharing photos and tracking walks with pets. The thesis consists of analysis of existing solutions, design, implementation and testing of application. The results of this thesis is functional prototype of the application for dog owners.

Keywords mobile application, dogs, Android, Kotlin, social network, dog walking

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Pes je nejlepší přítel člověka - toto tvrzení slyšel snad každý a málokdo má o tom pochybnosti. V dnešní době je pes nejen věrným přítelem, ale i pomáhá lidem chránit pořádek, zachraňovat životy a dokonce dokáže zbavit člověka negativních emocí. Tyto a ještě mnoho dalších zásluh odůvodňují oblibu psů mezi lidmi, a právě proto se pes stal jedním z nejrozšířenějších domácích mazlíčků.

V dnešní době, kdy skoro každý člověk má v ruce chytrý mobilní telefon s připojením k internetu, rozvoj sociálních sítí spojujících uživatele se stejnými zájmy se výrazně urychlil, což se nemohlo nedotknout i majitelů psů. Každý den v různých sociálních sítích se objevuje velké množství fotografií a příspěvků, které popisují běžný život psů: procházky, hraní, spánek a mnoho dalších. Velké množství domácích mazlíčků dokonce má na sociálních sítích vlastní stránky.

Tak vznikla myšlenka mobilní aplikace, která bude spojovat majitele psů z různých částí zeměkoule a bude umožňovat sdílení fotografií a okamžiků ze života jejich domácích mazlíčků. Tím se vytvoří komunita lidí se stejnými zájmy a myšlenkami, o které se mohou podílet v rámci stránek svých psů. Zároveň tato aplikace by mohla podněcovat lidi chodit na delší procházky se svými domácími mazlíčky tím, že by umožňovala pořízení záznamu trasy procházky se psem a případné sdílení pro všechny členy sociální sítě. V době psaní této práce neexistovala žádná Android mobilní aplikace, která by fungovala jako sociální síť a zároveň umožňovala sledování trasy procházky se psem.

Cílem této diplomové práce je návrh a vývoj výše popsané mobilní aplikace včetně serverové části, která bude ukládat a poskytovat data uživatelů. V rámci této práce budu zkoumat konkurenční mobilní aplikace pro majitele domácích mazlíčků a provedu analýzu uživatelských požadavků. Dalším krokem bude návrh vhodné funkcionality, po čemž bude následovat návrh jednotlivých součástí pomocí metod softwarového inženýrství. Potom bude implementace prototypu serverové a klientské části a následné testování celkového výsledku.

Kapitola 1

Analýza předpokladů

1.1 Současná řešení

Před několika lety ještě nikdo ani nemyslel na to, že chytrý telefon bude mít každý člověk nezávisle na finančním stavu a použití mobilních aplikaci bude samozřejmostí, a proto sociální sítě pro majitele zvířat byly jen ve tvaru webových stránek. V dnešní době většina aplikací určených pro majitele domácích mazlíčků ani nemá webové rozhraní a existuje jen ve tvaru mobilních aplikací. Toto má jednoduchý důvod - člověk tráví víc času se svým telefonem než s počítačem a navíc může jedním kliknutím přidat nebo rovnou pořídit fotografii pro jakoukoliv sociální síť. Analýza současných řešení se hlavně bude zaměřovat na mobilní aplikace se sociálními prvky pro majitele zvířat. V době provedení analýzy žádná z vybraných aplikací neobsahovala funkci sledování trasy procházky se psem, což ale je důležitou částí této diplomové práce.

V průběhu této analýzy největší pozornost bude věnována aplikacím, které jsou určené převážně pro majitele psů, jelikož to odpovídá tématu diplomové práce. Aplikace mohou umožňovat přidávání stránek jiných zvířat, ale toto nebude detailně zkoumané. Shrnutí a porovnání všech analyzovaných mobilních aplikací bude uvedené na konci této sekce v pohodlné formě tabulky.

Podobné aplikace budou analyzované podle kritérií popsaných v následujících odstavcích.

- Založení účtu a přidání zvířat

  Tato část se bude zaměřovat na hodnocení jednoduchosti vytváření účtu. Především je důležitý proces přidávání různého počtu profilových stránek psů.

- Správa profilové stránky

  V této sekci budou především popsané možnosti přizpůsobení stránky uživatele a přidávání příspěvků nebo fotografií.

- Sociální prvky

  Sociálními prvky se v této analýze myslí například přidávání do oblíbených, sledování účtu a různé druhy komunikace.

- Doplňující funkce

  Aplikace mohou obsahovat jiné funkce kromě sociálních, které budou předmětem analýzy v této sekci.

- Uživatelské rozhraní

  Pochopitelné a pohodlné uživatelské rozhraní je jedním z nejdůležitějších prvků libovolné aplikace. Pokud uživatel má na vybranou mezi více konkurenčními řešeními, které nabízí stejné funkce, tak právě podle UX a UI bude většinou rozhodovat, jaký systém má přednost. Při hodnocení uživatelského rozhraní analyzovaných aplikací bude kladen velký důraz na použití standardních prvků pro operační systém, použití logické a pochopitelné navigace a estetický design. Výsledné hodnocení uživatelského rozhraní bude oceněné od 1 do 10, kde 10 je nejlepší výsledek. V hodnocení tohoto bodu není možné vyloučit subjektivní názor.

1.1.1 Dogalize

Mobilní aplikace Dogalize [1] má na Google Play víc než sto tisíc stažení a je dobře hodnocená uživateli, 4,2 body[1]. Je to první aplikace, která se objeví ve vyhledávání sociální sítě pro domácí mazlíčky a jak je možné pochopit z názvu, je primárně určena pro majitele psů. Již na první pohled je vidět, že je to jedna ze starších aplikací, a proto nutně potřebuje aktualizaci uživatelského rozhraní a grafiky. Jako hlavní výhodu své aplikace vývojáři v Google Play uvádí velkou a přátelskou komunitu vlastníků psů, která vznikla pomocí jejich aplikace. Také je potřeba zmínit, že sociální síť Dogalize má nejen aplikaci pro OS Android, ale ještě pro iOS a vlastní webové stránky. Předmětem této analýzy bude jen aplikace pro operační systém Android.

Analýza byla provedena na mobilním zařízení Samsung Galaxy S7 s verzí operačního systému Android 8.

1.1.1.1 Založení účtu a přidání zvířat

Jako libovolnou jinou sociální síť, aplikaci Dogalize není možné použít bez provedení registrace a následného přihlášení. Kromě běžné registrace pomocí elektronické adresy aplikace také umožňuje přihlášení pomocí Facebooku. Při registraci kromě klasických údajů, jako jsou elektronická adresa a heslo, aplikace se ještě zeptá na jméno uživatele. Velkou nevýhodou je to, že po registraci aplikace nenabídne přidání profilu svého psa, protože použití Dogalize

[1] Informace z 18. 3. 2018

není podmíněné vytvářením první stránky. Pro přidání profilu svého domácího mazlíčka uživatel musí přejít do sekce s účtem, dále zvolit správnou záložku a jen potom aplikace umožní přidat psa. Po zadání základních informací psovi, jako jsou plemeno, datum narození, pohlaví a povaha, uživatel může přidat ještě fotografie a videa. Zajímavé je to, že v aplikaci chybí seznam plemen psů a také neexistuje validace uvedeného plemena uživatelem, a proto je možné uvést libovolné slovo. Také aplikace umožňuje přidání více stránek psů stejným způsobem jako vytváření účtu prvního psa.

1.1.1.2 Správa profilu

Aplikace Dogalize umožňuje přidat a upřesnit velké množství informací, ve které se těžko dá vyznat. Profil uživatele se skládá z informací o samotném uživateli, jeho psech a příspěvků. Informace o uživateli kromě adresy také obsahuje fotografie a videa, které se netýkají domácích mazlíčků. Velkou nevýhodou je to, že psy nejsou povinnou součástí účtu uživatele. Náhled této obrazovky stránky uživatele a nepovinné stránky psa je možné vidět na obrázcích 1.1a, 1.1b. Příspěvky v Dogalize jsou zcela nezávislé na psech a jsou spojené jen s účtem uživatele. Právě tyto příspěvky se zobrazují na obrazovce se seznamem událostí, nikoliv fotografie psů nebo uživatele. I když v aplikaci existuje záložka s vlastními příspěvky, v průběhu analýzy se nepodařilo najít způsob jejich přidání kromě sdílení příspěvku již vytvořeného jiným uživatelem.

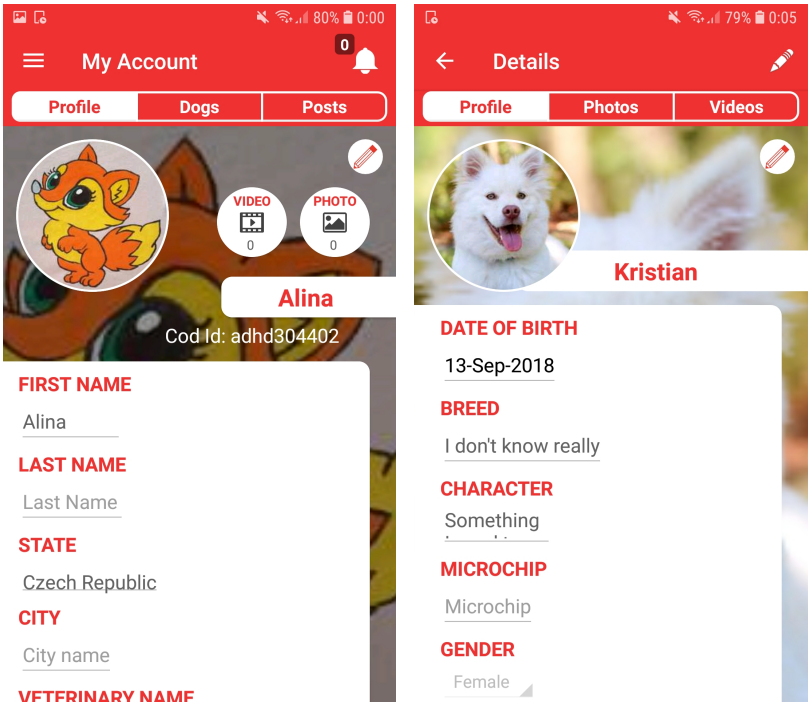
1.1.1.3 Sociální prvky

Tato aplikace je založena na myšlence přidávání do přátel, kde každý uživatel může prohlížet informace a fotografie jiného uživatele jen po schválení žádosti o přátelství. Toto se nevztahuje na příspěvky, které jsou vždy veřejné a dá se je prohlížet v sekci poznávání. Jako velkou nevýhodu se dá označit to, že v sekci poznávání nelze vyhledávat, a navíc počet doporučených příspěvků je omezený. Vyhledávání je umožněno v jiné položce menu určené pro hledání nových přátel, které ale nefunguje podle očekávání. V nápovědě k vyhledávacímu okénku je uvedeno, že vyhledávat je možné například podle plemena psa, ale po jeho uvedení se zobrazí seznam uživatelů aplikace, u kterých není možné se dozvědět nic kromě jména a úvodního obrázku. Velké plus je to, že aplikace umožňuje posílání osobních zpráv přátelům. Kromě toho existuje i funkce skupinové konverzace.

1.1.1.4 Doplňující funkce

Aplikace také obsahuje další funkce, jako jsou posílání osobních zpráv veterinářům a trenérům, adopce psů, hledání ztracených psů anebo partnera pro svého psa.

Obrázek 1.1: Ukázka aplikace Dogalize



(a) Stránka uživatele (b) Stránka psa

1.1.1.5 Uživatelské rozhraní

Hlavním problémem Dogalize je nedodržování standardů pro Android aplikace a zastaralý design. Aplikace nemá dobře promyšlenou roli psů, a proto všechny funkce spojené s domácími mazlíčky působí jako zbytečné a nedokončené. Implementace přidávání a prohlížení psů je nezdařilá, pro přidání psa a jeho fotografií je potřeba provést velké množství kroků a dá se v tom snadno ztratit. Například při přidávání fotografií není jasné, kterému psu ve výsledku budou patřit.

I když Dogalize obsahuje boční menu, navigace v aplikaci je dost komplikovaná. Často se stává, že není možné pochopit, jak udělat potřebnou akci nebo dokončit už započatý scénář.

Aplikace občas používá nelogické prvky jako například existence dvou propisek pro úpravu účtu na jedné obrazovce. Také pokaždé při otevření aplikace je požadované oprávnění pro provedení telefonních hovorů bez žádného vysvětlení. Dále je možné zmínit, že Dogalize obsahuje návod, který ale zřejmě neodpovídá skutečné verzi aplikace.

Vzhledem k výše uvedenému tuto aplikaci hodnotím 4 body z 10.

1.1.2 Yummipets

Yummipets [2] je jedna z novějších sociálních sítí pro majitele různých druhů zvířat a má na Google Play kolem 50 tisíc stažení. Aplikace umožňuje založení stránky nejen pro běžné psy a kočky, ale i například pro křečka nebo želvu. Yummipets má trochu větší hodnocení než předchozí analyzovaný Dogalize, a to 4,4[2]. Navíc tato aplikace je dost často aktualizovaná, například poslední její aktualizace byla zaměřena na vylepšení pro novější Android 8.

Analýza byla provedena na mobilním zařízení Samsung Galaxy S7 s verzí operačního systému Android 8.

1.1.2.1 Založení účtu a přidání zvířat

Yummipets stejně jako předchozí aplikace nabízí dvě možnosti registrace, a to pomocí elektronické adresy a Facebooku. Pokud se uživatel registruje pomocí emailu, tak je donucen uvést svoje jméno, příjmení a datum narození a tento krok není možné přeskočit. Po registraci uživatel musí přidat k svému účtu aspoň jednoho domácího mazlíčka. Pro psa je potřeba ukázat jeho jméno, plemeno a pohlaví, uživatel má také možnost uvést věk a nahrát fotografii.

Po přidání prvního domácího mazlíčka aplikace nabídne zajímavé stránky k sledování, což je skvělý začátek pro vytvoření zajímavého účtu. Další zvíře je možné přidat v rámci nastavení své stránky. Je důležité pochopit, že každý domácí mazlíček v Yummipets má vlastní stránku v této sociální síti. V aplikaci je to řešeno tím, že uživatel se může přepínat v rámci svého účtu mezi různými domácími mazlíčky, a tím naplňovat jednotlivé stránky pro každého z nich.

1.1.2.2 Správa profilu

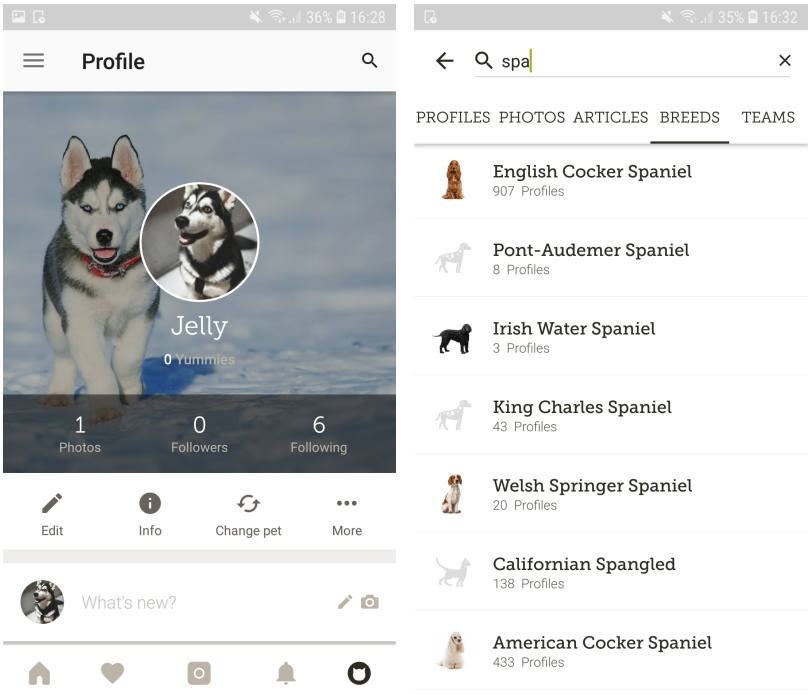
Aplikace poskytuje jednoduché rozhraní pro správu profilů svých zvířat. Uživatel může pro každého domácího mazlíčka přidávat událost, která se skládá z popisu a jedné fotografie, událost je možné přidat i bez fotografie, ale naopak to není povoleno. Je velmi zajímavé to, že dostat se z profilu jednoho domácího mazlíčka do druhého, které jsou propojené s jedním uživatelským účtem, je dost složité, a proto uživatel nemůže snadno vidět, kdo je součástí rodiny tohoto zvířete. Dále v aplikaci není možné vidět žádné informace o vlastníkovi. Náhled profilové stránky je možné si prohlédnout na obrázku 1.2a.

1.1.2.3 Sociální prvky

Aplikace Yummipets je založena na principu sledování, kde uživatel v rámci profilu svého domácího mazlíčka může sledovat události v životě jiných zvířat. Každý domácí mazlíček v rámci jednoho uživatelského účtu má své sledující a sledování, a to znamená, že se jim zobrazují odlišné události v novinkách. Jako

[2] Informace z 25. 3. 2018

Obrázek 1.2: Ukázka aplikace Yummipets



(a) Stránka uživatele (b) Vyhledávání plemen

v každé jiné sociální síti uživatele můžou přidávat fotografie do oblíbených a také psát komentáře. Velkým kladem této aplikace je možnost překladu popisu k fotografiím do svého jazyka, ale na druhou stranu překlad komentářů už není dostupný. Jako velká nevýhoda se dá uznat to, že v novinkách je možné prohlížet jen několik posledních příspěvků, další se nenačítají.

Nové profily ke sledování je možné najít ve speciální záložce, kde aplikace nabídne seznam stránek domácích mazlíčků, které jsou nejvíc populární nebo aktivní. Tato funkce umožňuje uživateli sledovat nejzajímavější profily dostupné v sociální síti. Aplikace také umožňuje vyhledávání stránek psů podle plemen, což je možné vidět na obrázku 1.2b.

1.1.2.4 Doplňující funkce

Yummipets je klasická sociální síť, a proto nemá skoro žádné jiné doplňující funkce. Jediné, co je možné zmínit, to jsou články o domácích mazlíčcích napsané redaktory této aplikace.

1.1.2.5 Uživatelské rozhraní

Sociální síť Yummipets je nová, a proto má o hodně méně nedostatků v uživatelském rozhraní než předchozí analyzovaná aplikace. Díky podobnosti s jinými sociálními sítěmi je aplikace snadno použitelná a má moderní design používající prvky doporučeného standardu pro vývoj Android aplikací. Největší výhoda této aplikace je v tom, že Yummipets není přetížená zbytečnými funkcemi, a proto většina prvků je snadno dohledatelná. Dále je vidět, že aplikace se nachází ve stavu, kdy je už plně použitelná, ale stejně ještě potřebuje malé dodělávky, nad kterými nyní pracují vývojáři.

Z nevýhod je třeba zmínit, že aplikace Yummipets nedovoluje přidávání komentářů bez potvrzení registrace na elektronické adrese, což ale nikde není uvedeno. Také po přidání nebo mazání příspěvků je uživatel navigován na stránku s novinkami, nikoliv na svůj účet.

Vzhledem k tomu, že Yummipets má opravdu dobře zpracované uživatelské rozhraní, tato aplikace dostává 8 bodů z 10.

1.1.3 Bauwow

Bauwow [3] je aplikace, která se nemůže pochlubit vysokým počtem stažení (mezi 5 a 10 tisíci)[3], ale má velmi vysoké hodnocení uživatelů. Tato aplikace je celkově orientovaná na psy a všechny funkce jsou navržené pro jejich majitele. Aplikace je zajímavá tím, že nabízí nejen možnost být členem sociální sítě, ale i databázi rozmanitých služeb pro psy v různých zemích.

Analýza byla provedena na mobilním zařízení Samsung Galaxy S7 s verzí operačního systému Android 8.

1.1.3.1 Založení účtu a přidání zvířat

Tato aplikace jako jediná z analyzovaných nabízí přihlášení nejen pomocí Facebooku a elektronické adresy, ale ještě i pomocí Google. Registrace pomocí emailu nebyla v průběhu analýzy funkční, ale pořád se bylo možné registrovat pomocí Google. Po registraci aplikace požádá o přidání psa, kde povinnými atributy jsou jméno, plemeno a fotografie. Aplikace nemá vlastní seznam plemen, proto uživatel může uvést cokoliv. Přidání jiného psa je jednoduché a probíhá podle stejného scénáře. Přepínání mezi psy je naopak velmi složité a neintuitivní.

1.1.3.2 Správa profilu

Bauwow umožňuje detailní nastavení profilu svého psa, kde je možné přidat fotografie dokumentů (například pas nebo pojištění), historii návštěv veterináře, předpisy, alergie, další návštěvy specialistů a podobně. Tato informace

[3] Informace z 8. 4. 2018

Obrázek 1.3: Ukázka aplikace Bauwow

(a) Stránka uživatele (b) Hlavní stránka (rozcestník)

není viditelná nikomu kromě vlastníka a slouží pro udržování všech informací o domácím mazlíčkovi na jednom místě. Ke každému profilu psa je možné přidat fotografie, které také nejsou viditelné pro ostatní uživatele. Aplikace se liší od běžných sociálních síti a součástí účtu psa nejsou příspěvky, které ale v Bauwow existují. Náhled stránky psa je možné vidět na obrázku 1.3a.

1.1.3.3 Sociální prvky

Aplikace nabízí speciální systém sdílení příspěvků, který je vyčleněn zvlášť od profilu psa. Uživatel sdílí příspěvky v oddělené záložce a příspěvky jsou viditelné pro všechny v rámci obrazovky s událostmi. Také uživatelé můžou spojovat své psy do smeček a vyměňovat si mezi sebou fotografie a zprávy. Zajímavějším prvkem je to, že uživatel může poslat pozvánku na procházku v parku, ale jen uživateli, který se nachází v jeho smečce. I když je to zmíněné jako funkce navíc, toto se ničím neliší od obyčejné osobní zprávy. Jako zajímavou možnost lze zmínit funkci, která dovoluje najít nejbližší psy a poslat jim nabídku připojení ke své smečce.

1.1.3.4 Doplňující funkce

Aplikace Bauwow je bohatá na doplňující funkce, které dovolují uživatelům rychle najít odborníky nebo speciální místa pro svého psa. Přidání míst a specialistů je řešeno přes doporučení přidat do aplikace nějaký prvek jako například místa pro procházku, místa pro oběd se svým psem, zajímavosti, zoo obchody, veterinární kliniky, psí kadeřnictví a podobné. Je velká nevýhoda, že přidání míst záleží jen na uživatelích, aplikace nemá předpřipravenou databázi pro všechny země aspoň s minimálním množstvím těchto míst, tak například pro Českou republiku neexistuje zatím žádné zajímavé místo a nejbližší park se nachází v Německu. Také aplikace nabízí prohlížení kreativních prací spojených se psy, jako jsou například malby nebo sochy. V průběhu analýzy se nepodařilo odhalit, jestli obyčejný uživatel může nahrávat své práce do této záložky.

1.1.3.5 Uživatelské rozhraní

Tato aplikace porušuje standardy pro Android aplikace a je vzorem, jak se aplikace nemají psát. Zaprvé Bauwow obsahuje několik dolních navigačních lišt, které se nesystematicky mění a jsou jen na nějakých obrazovkách. Zadruhé aplikace používá jako hlavní navigační prvek obrazovku s velkými tlačítky, která vedou na různé stránky. Také aplikace nemůže patřit nejen k materiálnímu designu, ale i k žádnému existujícímu. Hlavní stránku aplikace je si možné prohlédnout na obrázku 1.3b.

Uživatel v této aplikaci nemá žádnou kontrolu, příkladem je to, že uživatel nemůže smazat svůj příspěvek anebo aplikace může náhodně odhlásit uživatele z Bauwow. Aplikace občas neukazuje, že se něco provádí a nikdy není zřejmé, jestli nějaká stránka je opravdu prázdná nebo se pořád načítá. Bauwow umožňuje přidat prázdný příspěvek, uvést nesprávné plemeno a většinou se nesnaží předejít chybám.

Při procházení aplikací je velmi snadné se ztratit a vrácení na hlavní obrazovku je občas složité. Také uživatel si musí vždy pamatovat, který pes je v daný okamžik aktivní, protože je to uvedené jen v detailu účtu.

Výsledné hodnocení této aplikace je 0 bodů z 10.

1.1.4 Pack

Sociální síť pro majitele psů pod názvem Pack [4] má velmi dobře zpracovanou webovou verzi a mobilní aplikaci dostupnou jen pro operační systém iOS. Aplikace se nachází ve stavu beta, není ještě dokončena a obsahuje řadu nedostatků, ale už teď vypadá lépe než některé jiné aplikace. Webová verze sociální sítě Pack umožňuje správu účtů svých psů a jejich přidávání do smeček rozdělených podle plemen. Každé plemeno je smečka a má vlastní stránku s fotografiemi, populárními psy v této smečce a nováčky. Pack je zaměřená na sdílení fotografií, a proto ani nedovoluje přidávání příspěvků bez ní.

Analýza byla provedena na mobilním zařízení iPhone 5 s verzí operačního systému iOS 10.3.

1.1.4.1 Založení účtu a přidání zvířat

Aplikace umožňuje jak vytvoření účtu pomocí elektronické adresy, tak i registraci pomocí Facebooku. Po vytvoření účtu aplikace nenabídne vytvoření stránky psa, protože použití této sociální sítě je v nějaké míře možné i bez přidávání domácích mazlíčků. Po registraci uživatel může začít sledovat nějaké profily, přidávat do oblíbených a psát komentáře. Vytvoření profilu psa není složité a provádí se ve formě jednoduchého dotazníku, který se zeptá na jméno, plemeno, pohlaví, bydliště a fotografii. Uživatel si může přidat libovolné množství psů a řídit jejich stránky. Vzhled profilové stránky psa uživatele je možné vidět na obrázku 1.4a

1.1.4.2 Správa profilu

Každý pes v této sociální síti má svoji vlastní stránku a své sledující. V dolní části obrazovky je rychlý přístup k přidání fotografie na stránku psa. Uživatel před přidáním obrázku bude vyzván oříznout ho do tvaru čtverce. Jako velkou výhodu se dá označit to, že uživatel může sdílet tuto fotografii nejen v sociální síti Pack, ale ještě i na Instagramu. Náhled této obrazovky je možné vidět na obrázku 1.4b. Co se týká osobních informací, tak změnit je to možné jen na webových stránkách, aplikace zatím tuto funkci neumožňuje.

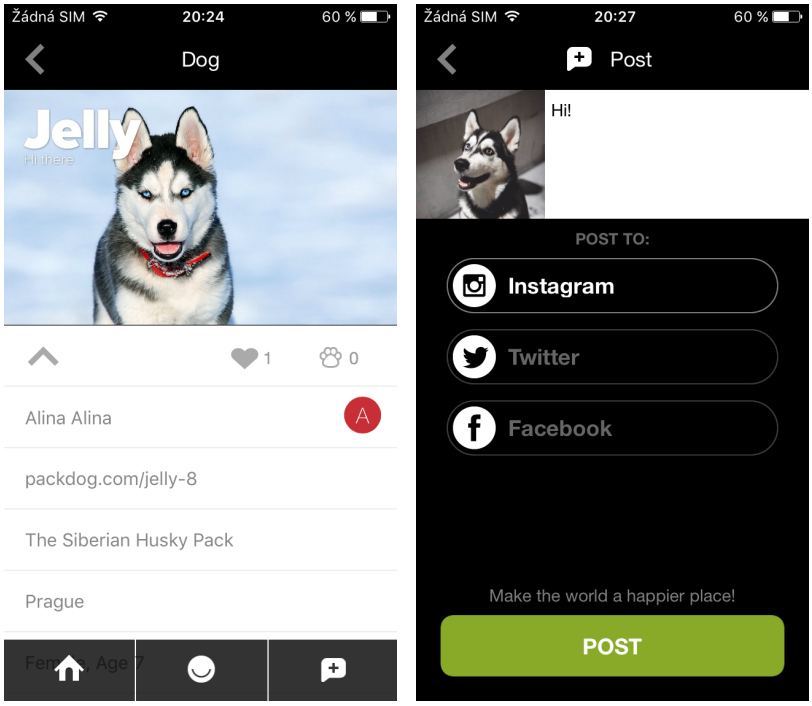
1.1.4.3 Sociální prvky

V této aplikaci zatím není naimplementovaný způsob prohlížení všech účtů uživatelů, kteří sledují profil daného psa. Důležitější je to, že v aplikaci také není způsob prohlížení všech profilů, které sleduje daný uživatel, což v praxi znamená, že přestat sledovat nějaký profil je nemožné. Aplikace obsahuje funkci přidání do oblíbených a přidání komentářů, ale zatím nenabízí možnost prohlížení událostí, které jsou spojené s profily psů uživatele. Toto je velká nevýhoda, jelikož uživatel se například nemůže dozvědět, kdo a kde nechal svůj komentář a nemůže na něho odpovědět. Také aplikace zatím neobsahuje funkcionalitu spojenou se smečkami, která ale je velmi rozsáhlá ve webové verzi.

1.1.4.4 Doplňující funkce

Aplikace Pack je klasická sociální síť zaměřená na sdílení fotografií a mobilní verze nejen neobsahuje žádné další funkce, ale i postrádá nějaké důležité možnosti jako například odhlášení uživatele.

Obrázek 1.4: Ukázka aplikace Pack



(a) Stránka psa uživatele (b) Přidávání příspěvku

1.1.4.5 Uživatelské rozhraní

Hlavním důvodem k absenci větších problémů v uživatelském rozhraní této aplikace je to, že Pack se nachází ve stavu beta, a proto postrádá velké množství užitečných a nezbytných funkcí jako například mazání příspěvků, odhlášení, prohlížení všech sledovaných účtů, vyhledávání ve smečkách a mnoho dalších.

Aplikace obsahuje několik nestandardních prvků, jako je například dolní navigační lišta s neobvyklou animací. Další velká nevýhoda je to, že v několika různých místech tlačítka přidání do oblíbených a začínání sledování jsou v různém pořadí, což je velmi matoucí pro uživatele. Dále ikonka aplikace nahoře obrazovky má nekonzistentní chování, jednou aktualizuje seznam událostí a jednou nabídne udělat svoji smečku, což ve výsledku jen začne sledovat několik náhodných psů.

Pack skoro neobsahuje vnořené obrazovky a všechny akce se provádějí v rámci třech záložek dolní navigační lišty, a proto aplikace má velmi snadnou navigaci.

Vzhledem k výše uvedenému aplikace dostává hodnocení 6 bodů z 10.

1.1.5 Porovnání analyzovaných aplikací

Každá z analyzovaných aplikací se vytvářela jako populární sociální síť pro majitele psů a jiných zvířat. Některé aplikace byly velmi přetížené různými funkcemi, které byly zbytečné a jen mátly uživatele, některé naopak postrádaly důležité věci. V následující tabulce 1.1 je možné vidět porovnání funkcí a vlastnosti analyzovaných aplikací.

Hlavním cílem analýzy konkurenčních řešení bylo zjistit silné a slabé stránky mobilních aplikací pro majitele psů, aby bylo možné převzít dobré zkušenosti a vyhnout se závažným chybám v budoucím návrhu a implementaci této diplomové práce. Společnou chybou všech aplikací kromě Yummipets bylo použití nestandardních navigačních prvků a designu. Také v některých aplikacích chyběla důležitá možnost pohodlného vyhledávání stránek podle plemena nebo jména, což je nepřijatelné pro sociální síť, cílem které je sdílet fotografie a významné okamžiky ze života domácích mazlíčků uživatele. Dalším důležitým poznatkem je to, že i když každá aplikace dovoluje přidání více profilů, pohodlné přepínání mezi nimi a vedení stránky bylo problematické u všech analyzovaných aplikací.

Ze všech analyzovaných aplikací Yummipets a Pack byly zaměřené jen na poskytování funkcí běžných pro sociální sítě a výsledně mají větší přehlednost než Dogalize a Bauwow. Tyto aplikace byly přetížené zbytečnými funkcemi, které se vývojářům nepodařilo dobře znázornit v uživatelském rozhraní a implementovat všechna opatření proti chybám.

1.2 Dotazník

Libovolná aplikace musí být zajímavá a užitečná především pro lidi, kterým je určena, a proto nedílnou částí mé diplomové práce je zjištění skutečných potřeb a zájmu uživatelů. Dnes chytré mobilní telefony jsou samozřejmostí pro každého člověka, a to ani nezáleží na materiálním a sociálním zázemí. Proto je důležité vyvíjet aplikace, které skutečně budou plnit očekávání lidí a přinášet do jejich života nové možnosti a zkušenosti.

Dotazník většinou bude zaměřený na cílovou skupinu budoucí aplikace, a to na majitele psů, ale obecně bude umožňovat zodpovědět lidem, kteří nemají zvíře nebo mají jiné. Cílem otázek z dotazníku bude zjištění celkového zájmu o sociální síť pro majitele psů a také na zkoumání rámcových požadavků na funkčnost aplikace. Důležitá část ankety bude orientována na procházky se psy, jak často se tomu lidé věnují nejen kvůli potřebě vyvenčit psa, ale i vnímají to třeba jako sportovní nebo relaxační zážitek.

Jazykem dotazníku byla zvolena angličtina, jak kvůli možnosti se dozvědět názory lidí z více zemí, tak i kvůli rozšíření na cizojazyčných webových stránkách. Dotazník byl vytvořen na platformě Google Forms[4] a byl rozeslán

[4] www.google.com/forms/about/

Tabulka 1.1: Porovnání funkcí a vlastností

Funkce nebo vlastnosti	Dogalize	Yummipets	Bauwow	Pack
Počet stažení	100 000+	10 000+	5 000+	Nelze určit
Hodnocení	4,2	4,4	4,5	Nelze určit
Zaměřenost jen na psy	✓	✗	✓	✓
Přihlášení přes Facebook	✓	✓	✓	✓
Přihlášení přes Google	✗	✗	✓	✓
Databáze plemen psů	✗	✓	✗	✓
Existence skupin (smeček)	✗	✗	✓	✓
Možnost přidání víc profilů	✓	✓	✓	✓
Přidávání komentářů	✓	✓	✓	✓
Přidávání do oblíbených	✓	✓	✓	✓
Posílání zpráv	✓	✓	✓	✗
Vyhledávání v profilech zvířat	✓	✓	✗	✗
Upozornění na nové události	✓	✓	✗	✗
Další funkce	✓	✗	✓	✗
Sledování trasy procházky se psem	✗	✗	✗	✗
Uživatelské rozhraní	4/10	8/10	0/10	6/10

pomocí sociálních sítí ve skupinách věnujících se psům a také byl zveřejněn na několika cizojazyčných fórech pro majitele domácích mazlíčků.

1.2.1 Struktura dotazníku

Dotazník se skládá ze třech logických celků, kde každý plní svou důležitou funkci. První sekce je určena jen pro lidi, kteří používají na svém mobilním zařízení nějakou sociální síť. Cílem této části je zjistit, jak často uživatelé používají aplikace tohoto typu na svých chytrých telefonech a zájem o přizpůsobení sociálních sítí pro domácí mazlíčky. Druhá část je určena jen pro majitele psů a je úplně zaměřena na procházky a venčení psů. Třetí část je nejdůležitější pro vytváření požadavků na aplikaci a skládá se jen z jedné otázky, která přímo zjišťuje, jaké funkčnosti podle respondentů by měly být nedílnou součástí sociální sítě pro majitele psů.

Existuje několik průchodů tímto dotazníkem a nejkratší zjistí jen uživatelské požadavky na aplikaci. Nejdelší průchod je určen pro majitele psů, jejichž názor je nejdůležitější pro tuto diplomovou práci.

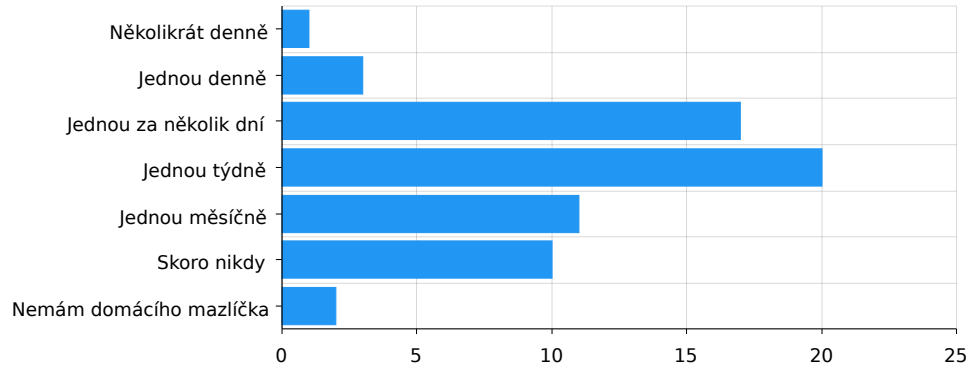
Český překlad zveřejněného dotazníku je si možné prohlédnout v příloze B.

1.2.2 Vyhodnocení výsledků

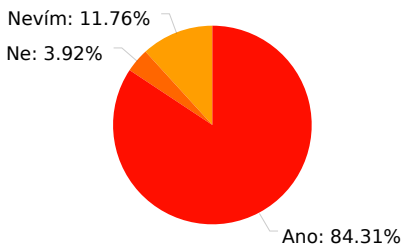
Celkem vyplnilo dotazník 68 respondentů z různých zemí. Tato anketa ukázala, že 64 respondentů z 68 používá sociální sítě na svém mobilním zařízení, což je přibližně 94 %. Je třeba brát v potaz, že výsledky této otázky mohou být ovlivněné faktem, že dotazník byl rozšířen hlavně pomocí sociálních sítí. Na druhou stranu to potvrzuje myšlenku, že vývoj aplikací pro mobilní zařízení v dnešní době je velmi důležitý. Toto také potvrzuje další výsledek, že z 64 respondentů 54 lidí používá sociální sítě minimálně každý den.

Další dvě důležité otázky se zaměřovaly na vliv domácích mazlíčků na sociální sítě uživatele. Otázka, jestli respondent sdílí fotografie nebo důležité okamžiky ze životů svých domácích mazlíčků na sociálních sítích, ukázala, že jen 10 lidí nikdy nezveřejňovalo fotografie svých zvířat. Ostatních 52 respondentů to dělá aspoň jednou měsíčně. Výsledky této otázky jsou znázorněny na obrázku 1.5. Další otázka na tuto tematiku se naopak snažila zjistit, jestli se respondentům líbí sledovat účty jiných lidí, kteří vlastní zvířata. Většina respondentů odpověděla, že aspoň někdy prohlíží účty nebo fotografie s domácími mazlíčky. Z těchto dvou otázek plyne, že uživatelé opravdu mají zájem o zvířecí tematiku a rádi prohlížejí a sdílejí fotografie svých domácích mazlíčků.

Následující blok otázek byl zaměřený na majitele psů, kterých mezi respondenty bylo 75 %. Z výsledků dotazníku plyne, že až 37 lidí z 51 chodí na delší procházku aspoň jednou za několik dní, z nich 10 to dělá každý den. Očekávaným faktem, potvrzujícím zadání této diplomové práce, se také stalo to, že skoro 84 % respondentů, kteří vlastní psa, by mělo zájem o sledování



Obrázek 1.5: Častost sdílení respondenty fotografií svých domácích mazlíčků

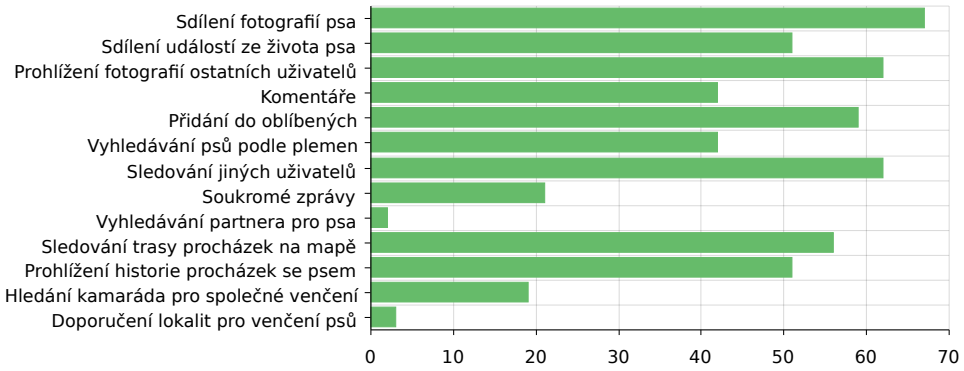


Obrázek 1.6: Výsledky otázky, jestli se respondentovi líbí myšlenka sledování trasy procházky se psem

trasy venčení. Na obrázku 1.6 je možné vidět variantu odpovědi a odpovídající počet respondentů. Na druhou stranu, neočekávaně se objevilo to, že myšlenka hledání kamaráda pro společné venčení nebyla tak úspěšná, jen méně než 50 % respondentů odpovědělo, že to by mohla být užitečná funkce.

Hlavní otázka, výsledky, které mě zajímaly nejvíce, je o požadavcích uživatelů na sociální síť pro majitele psů. Z odpovědí plyne, že všichni by chtěli mít balík standardních funkcí libovolné sociální sítě, a to sdílení fotografií a příspěvků, sledování, přidání do oblíbených a vyhledávání jiných uživatelů. Také velmi velké hodnocení má sledování trasy procházky se psem, což bylo jasné ještě z předchozích otázek. Z výsledků této otázky je ještě jednou vidět, že funkčnost hledání kamaráda pro společné venčení je méně požadována než jako například soukromé zprávy. Podrobné výsledky této otázky jsou znázorněny na obrázku 1.7.

Závěrem analýzy výsledků dotazníku je potvrzení většiny domněnek ohledně požadavků na sociální síť pro majitele psů. Uživatelé nejvíce ocení, pokud



Obrázek 1.7: Rozložení zájmu uživatelů o funkčnosti v navržené aplikaci

aplikace bude umožňovat klasické funkce sociálních sítí a sledování trasy procházky se psem. Funkčnost hledání kamaráda pro společné venčení není tak důležitá pro uživatele, i když je příjemným doplňkem k ostatním možnostem použití aplikace.

Kapitola 2

Analýza

2.1 Výběr platformy

V dnešní době na trhu mobilních operačních systémů bojují dva základní giganti - Android a iOS, kde každý má své výhody a nevýhody. V této části zjistíme, jaké jsou silné stránky každé platformy a také uvidíme základní rozdíly v přístupu k vývoji pro tyto dvě platformy.

Operační systém Android se poprvé objevil v roce 2008 na mobilním zařízení T-Mobile G1 a v té době si nikdo nemyslel, že může získat takovou popularitu. Rychlý růst a rozšíření cílové skupiny uživatelů se odehrálo v půlce roku 2012, kde se na svět dostala nová verze Androidu Jelly Bean, která byla mnohem úspěšnější než předchozí verze. [7]

Pokud budeme mluvit o stavu v červenci roku 2018, Android nyní obsazuje neuvěřitelných 77,32 % světového trhu mobilních operačních systémů. V porovnání s operačním systémem iOS, který ve stejný čas obsahoval jen 19,4 %, je to skoro čtyřikrát větší cílová skupina lidí, která potenciálně může zkoumat a hodnotit novou aplikaci [5]. Je potřeba říct, že pokud budeme uvažovat jen trh Spojených států amerických, tak potom situace bude vypadat úplně jinak a podíl iOS je trochu větší než 52,19 %, Android zabírá jen 47,39 % [6].

Další bod, který není možné opomenout, je rozdíl v rozmanitosti zařízení s operačním systémem Android a iOS. Android může být nainstalovaný na zařízeních libovolného výrobce mobilních telefonů a tabletů, navíc každý výrobce může upravovat jádro systému, protože systém Android je otevřeným softwarem. Toto zařízení může mít libovolný poměr stran, rozlišení a navíc nepodporovat aktualizaci operačního systému. Pokud se aplikace pro Android stává opravdu známou po celém světě, tak se určitě začnou objevovat problémy s podporou nestandardních a neklasických zařízení. Oproti tomu iOS má velkou výhodu, protože aktualizace systému iOS na většině podporovaných zařízení probíhá v průběhu několika měsíců po zveřejnění nové verze. Navíc operační systém iOS může být nainstalovaný jen na omezeném počtu zařízení výrobce Apple.

Tabulka 2.1: Rozdělení verzí operačního systému Android, převzato z [10]

Verze	Označení	API	Podíl
2.3.3 - 2.3.7	Gingerbread	10	0,2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0,3%
4.1.x	Jelly Bean	16	1,2%
4.2.x		17	1,9%
4.3		18	0,5%
4.4	KitKat	19	9,1%
5.0	Lollipop	21	4,2%
5.1		22	16,2%
6.0	Marshmallow	23	23,5%
7.0	Nougat	24	21,2%
7.1		25	9,6%
8.0	Oreo	26	10,1%
8.1		27	2,0%

Také je třeba zmínit to, že dostat aplikaci na Google Play Store je několikrát snadnější než na iOS App Store, a to kvůli nižším poplatkům a rychlejší a mírnější kontrole aplikace.

I když vývoj pro Android je složitější kvůli rozmanitosti zařízení, první verzi aplikace je lepší dělat právě pro tuto platformu kvůli možnosti získat širší zpětnou vazbu a zjistit zájem uživatelů o aplikaci. Výjimkou může být, jen pokud se cílová skupina skládá většinou z občanů Spojených států amerických, pak je opravdu lepší začínat vývoj aplikace z platformy iOS. Jelikož tématem této diplomové práce je návrh a vytvoření prototypu sociální sítě, což předpokládá velké množství uživatelů v budoucnosti, rozhodnutí vyvíjet aplikaci pro operační systém Android vypadá jako nejlepší možná cesta.

2.2 Výběr minimální podporované verze

Již v předchozí sekci bylo naznačeno, že aktualizace starších Android systémů na mobilních zařízeních probíhá velmi pomalu nebo neprobíhá vůbec. Toto staví před vývojáře velký problém - jakou minimální Android verzi bude podporovat aplikace. Podle pokynů od Google je potřeba podporovat aspoň 90 % aktivních zařízení a cílit na nejnovější verzi Androidu [9]. Následující tabulka 2.1 nám uvádí rozdělení různých Android verzí v červenci roku 2018.

Jak je vidět z tabulky, podporovat verze 2.3.3 až 4.3 nemá žádný smysl,

protože celkem obsazují méně než 5 % všech Android zařízení, což je velmi málo a je možné zanedbat. Jako výsledek aplikace bude podporovat Android verzi od API 19 (4.4 KitKat), což bude pokrývat přibližně 95 % všech uživatelů.

2.3 Výběr programovacího jazyka

Ještě před dvěma lety, když se řeklo programování pro Android, tak každý, kdo měl s tím aspoň nějaké zkušenosti, věděl, že pro Android se programuje v jazyku Java. Přelom nastal v roce 2017, kdy na Google I/O byl představený nový jazyk programování pro Android pod názvem Kotlin. Je třeba zmínit, že samotný jazyk se objevil ještě v roce 2012 a původně nebyl určen pro Android. [11]

Kotlin je staticky typovaný programovací jazyk pro JVM, který svým stylem je velmi podobný jazyku Scala. Tvůrci tohoto jazyka se v průběhu vývoje snažili klást důraz na bezpečnost, jednoduchost, kompatibilitu s jazykem Java, a také na krátkost a čitelnost výsledného kódu.

Oznámení Google o oficiální podpoře Kotlinu pro Android v roce 2017 bylo jen odpovědí na žádosti mnoha Android vývojářů, kteří viděli v Kotlinu velký potenciál. Pravděpodobně kvůli vlivu Google se Kotlin z 65. nejpopulárnějšího programovacího jazyka v roce 2017 dostal na 27. místo na začátku 2018. [12]

V následujících podsekcích budou uvedené hlavní rozdíly mezi těmi dvěma jazyky a výhody použití Kotlinu.

2.3.1 Porovnání jazyků Java a Kotlin v Android programování

Další sekce je napsaná pomocí [13], [14], [15].

Hlavní výhoda Kotlinu je v tom, že přináší usnadnění a rysy moderního jazyka bez toho, aby vývojář musel něco obětovat. Kotlin je plně podporován v Android Studiu, což je oficiální IDE pro Android a může bez problémů existovat spolu s Javou v projektu. To se hodí ve starších projektech, které pořád rostou a kde se přidávají nové možnosti a funkce napsané v Kotlinu, aniž by se přepisoval celý projekt. Vývojáři se nemusí bát toho, že výkon Kotlinu bude nižší než u Javy, protože se Kotlin převádí do podobného bytecodu. Některé možnosti Kotlinu jako inline funkce a lambda naopak mohou zrychlit kompilování programů. Dále budou uvedené hlavní výhody Kotlinu oproti Javě.

2.3.1.1 Null-safety

Nejčastějším problémem v aplikacích napsaných v Javě je pád způsobený takzvaným `NullPointerException`, který se stává, když se program snaží přistupovat do objektu, kterým je `Null`. Cílem Kotlinu je eliminovat tento problém

a zavést nový způsob, jak se má zacházet s objekty. Při zavedení nové proměnné je třeba říct, jestli ona může být v nějakém případě `Null`, pokud ne, tak se nikdy nemůže stát `NullPointerException`.

V opačném případě je potřeba s ní zacházet opatrně pomocí operátoru `dog?.name`. Pokud se na tento operátor u `Nullable` proměnné zapomene, kompilátor vyhlásí chybu a sestavení projektu selže. Tím Kotlin může zajistit, že se nemůže stát `NullPointerException` při správném použití všech konstrukcí. V případech, kdy programátor je si jistý, že tato proměnná není `Null`, tak může použít operátor `dog!!.name`, který je vlastně přetypováním na `NonNull` proměnnou stejného typu. Tento operátor může vyvolat `NullPointerException`, proto jeho použití není vítáno.

2.3.1.2 Smart Cast

Další příjemnou vlastností kompilátoru Kotlinu, která je přímo spojena s objekty typu `Nullable`, je chytré přetypování (anglicky Smart Cast). Pokud kompilátor pozná, že `Nullable` proměnná není v daném kontextu `Null`, tak povolí přímý přístup do této proměnné, jako kdyby byla typu `NonNull`.

Toto je vidět na následujícím příkladu, kde objekt `name` je typ `Nullable`, a proto se nedá k němu přistupovat přímo bez speciálního operátoru. Pokud ale v kontextu zjistíme, že tento objekt není `Null`, tak můžeme jednoduše k němu přistoupit.

Kotlin: chyba kompilace

```
var name: String? = "Alice"
print(name.length)
```

Kotlin: smart cast

```
if (name != null)
    print(name.length)
```

Chytré přetypování se používá nejen při práci s `Null` typy. V Kotlinu existuje operátor `is`, který nahrazuje `instanceof` u Javy, a také funguje skvěle s použitím Smart Castu.

Java: abscence smart castu

```
final Object name = "Alice";
if (name instanceof String) {
   print((String) name);
}
```

Kotlin: příklad smart castu

```
val name: Any = "Alice"
if (name is String) {
   print(name)
}
```

Chytré přetypování v Kotlinu usnadňuje psaní aplikací a zvětšuje pochopitelnost kódu.

2.3.1.3 Synthetics

Vývojáři pro Android vždy měli problém s adresováním view v layoutu, protože k tomu v Javě sloužila metoda `findViewById()`, což je velmi náročná operace, která navíc ještě přináší hodně zbytečného kódu a je podvržená chybě kopíro-

vání. Pomocí nové funkce Kotlinu pod názvem synthetics je možné adresovat view přímo podle id. Dále následují dvě ukázky, jak se to dělá v Kotlinu a Javě.

Java: adresování view

```
TextView textView = (TextView)findViewById(R.id.textView);
textView.setText("It is Java code");
```

Kotlin: adresování view

```
textView.text = "It is Kotlin code"
```

2.3.1.4 Lambdas

Jednou z nejdůležitějších výhod Kotlinu je možnost předat anonymní funkci jako parametr jiné funkce. Dále následuje jednoduchá ukázka rozdílu, jak se tento problém řeší v Javě a Kotlinu. Je vidět, že v Kotlinu je to možné napsat na jednu přehlednou řádku, v Javě programátor to musí rozepsat na více řádků.

Java: ukázka onClick()

```
button.setOnClickListener(new OnClickListener() {
   @Override
   public void onClick(View v) {
      doSomething();
   }
});
```

Kotlin: ukázka onClick()

```
button.setOnClickListener { doSomething() }
```

2.3.1.5 Data třídy

Data třídy jsou určené pro implementaci modelových tříd a pomáhají psát jen opravdu užitečný kód. Hlavní výhodou data tříd je to, že vývojář obdrží hodně automaticky odvozených funkcí, aniž by napsal něco kromě definice dat. Kompilátor automaticky odvodí z primárního konstruktoru metody `hashCode()`, `equals()`, `toString()` a `copy()`, což ušetří spoustu mechanické práce programátora.

Kotlin: Data třída

```
data class Dog(val name: String) { var age: Int = 0 }
```

2.3.1.6 Shrnutí

V [16] se uvádí několik hlavních důvodů, proč by Java neměla být primárním programovacím jazykem Android vývojáře.

- Java již není moderní jazyk.
- Android podporuje jenom podmnožinu funkcí jazyku Java 8, které se liší podle verze platformy. Proto Java 8 může způsobovat problémy v programování pro Android.
- Java obsahuje jazykové problémy, jako jsou nekonečné bloky try-catch a NullPointerExceptions.
- Java je pořád procedurální jazyk, i když v poslední verzi přidává podporu lambda funkcí.
- Syntax jazyka je složitý, psaní a údržba většího projektu může vést k chybám. Kotlin kód je mnohem kratší a elegantnější.

Po analýze všech výhod Kotlinu a pochopení toho, že další vývoj Androidu se bude rozvíjet nejspíš jen v tomto směru, je možné říct, že každý Android programátor se musí naučit tento jazyk a používat ho jak v práci, tak i ve svých vlastních projektech. Není možné říct, že Java bude úplně vyloučena z Android programování, ale už za jeden rok oficiální podpory Kotlinu od Google je vidět, že většina programátorů pomalu přechází na Kotlin a Java zůstává jazykem starých projektů a knihoven.

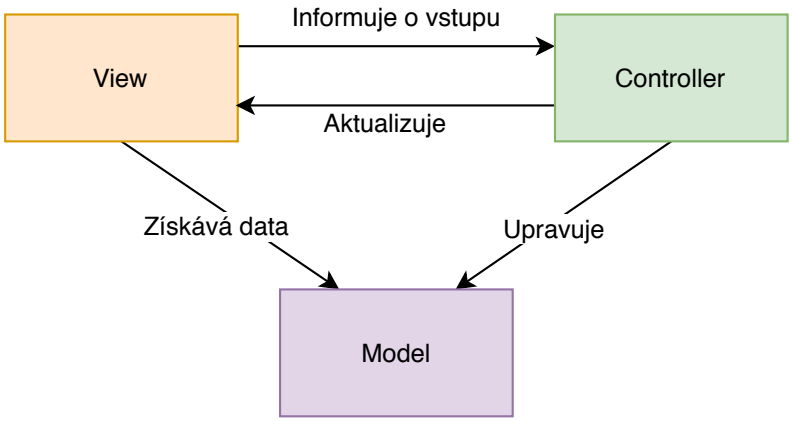
Vzhledem k udělané analýze, pro napsání této diplomové práce byl zvolen programovací jazyk Kotlin.

2.4 Výběr architektury

Architektura je velmi důležitá pro testování, vývoj a údržbu aplikace. Do roku 2017 neexistovala žádná doporučená architektura od Google pro vývoj mobilních aplikací. Proto se často byznys logika aplikace psala do Aktivit (hlavní prvek aplikace), což znemožňovalo znovupoužití kódu a snadnou rozšiřitelnost aplikace. Vzhledem k zjevným nevýhodám tohoto přístupu, pro vývoj aplikací se začaly aplikovat architektonické vzory, jako jsou MVC, MVP, MVVM a další. Existuje velké množství názorů, který z nich je lepší, v následující sekci bude uvedené porovnání nejpopulárnějších z nich.

2.4.1 Porovnání MVC, MVP a MVVM

MVC (Model - View - Controller), MVP (Model - View - Presenter) a MVVM (Model - View - ViewModel) mají dva společné prvky, což jsou Model a View. Model obsahuje byznys logiku a popis procesů, View obsahuje všechno, co se



Obrázek 2.1: Znázornění architektury MVC

týče zobrazení dat. V různých architektonických vzorech se způsob interakce mezi těmito elementy liší. V následujících sekcích čerpám z [17, 18].

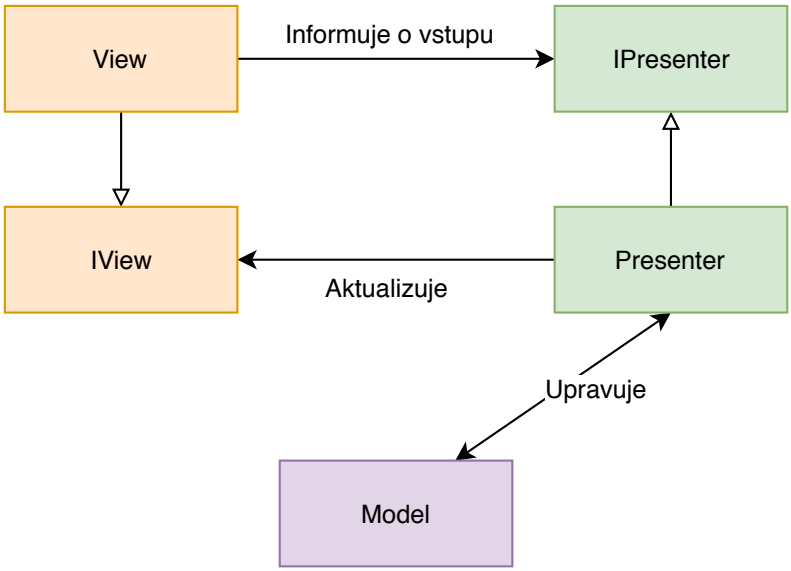
2.4.1.1 MVC

V MVC architektuře Controller získává uživatelský vstup a aktualizuje model. Jakmile je model aktualizován, Controller informuje View, že došlo ke změně modelu, potom View získává data z modelu a zobrazuje je uživatelovi. V implementaci tohoto vzoru pro Android Aktivita nejčastěji vystupuje zároveň jako View a Controller, což je velkou nevýhodou tohoto vzoru, který tímto způsobem porušuje princip oddělení odpovědností. Existuje i modifikace této architektury, kde Aktivita vystupuje jako View, který získává uživatelský vstup a potom informuje o tom speciální třídu Controller. Ale i tato varianta má své problémy, protože jak View, tak i Controller mají přístup k modelu, což velmi znesnadňuje testování kvůli vysoké provázanosti komponent. MVP vzor je znázorněn obrázku 2.1.

2.4.1.2 MVP

V architektuře MVP View komunikuje s Presenterem přes rozhraní, které má název kontrakt mezi Presenterem a View. Oba implementují rozhraní svých kontraktů a jsou přímo propojené, tudíž View má referenci na instanci Presentera a Presenter má referenci na View. Komunikace potom probíhá následovně: View informuje Presentera o uživatelském vstupu, ten upravuje Model a potom aktualizuje View. Klíčová odlišnost architektury MVP od MVC je to, že View nekomunikuje s Modelem přímo, výhodou je také použití rozhraní, a tudíž View a Presenter mohou být nezávisle otestované. Na obrázku 2.2 je znázorněn tento proces.

Ve světě Androidu View (Aktivity a Fragmenty) mají složitý životní cyklus, často zanikají a znovu se vytvářejí, což je jedním z hlavních problémů MVP.



Obrázek 2.2: Znázornění architektury MVP

Proto správné udržování referencí může být problematické, tento nedostatek řeší MVVM.

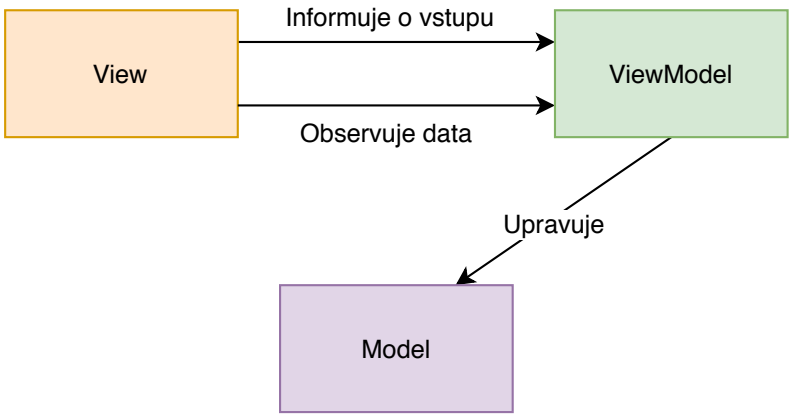
2.4.1.3 MVVM

V následující sekci budu čerpat z [19, 20].

Architektonický vzor MVVM je oficiální doporučená architektura pro OS Android od roku 2017, kdy byla představena na Google I/O. Architektura MVVM se skládá ze třech složek: View, ViewModel a Model. Jako u předchozích popsaných architektur, cílem MVVM je oddělení uživatelského rozhraní od logiky, ViewModel umožňuje opětovné použití jak View, tak i Modelu.

V architektonickém vzoru MVVM, komponenta ViewModel nemá referenci na View, úkolem ViewModelu je úprava Modelu a poskytování dat. View je třeba zaregistrovat na aktualizaci dat z ViewModelu, tím způsobem View vždy bude mít nejčerstvější data k zobrazení. ViewModel může poskytovat data i více zájemcům, například několika view nebo pro účely testování. Na obrázku 2.3 je možné vidět vizualizaci této architektury.

Usnadnit implementaci MVVM je možné pomocí použití doporučené sbírky knihoven Architecture Components pro návrh a vývoj robustních, testovatelných a udržitelných mobilních aplikací. Architecture Components nám umožňuje používat několik nejdůležitějších komponent pro čistou a přehlednou architekturu:



Obrázek 2.3: Znázornění architektury MVVM

- **LiveData**

  LiveData je kontejner pro data, kde libovolné jiné objekty mohou pozorovat změny těchto dat, aniž by se mezi nimi vytvářely závislosti. Stejně jako ViewModel, LiveData nejsou závislé na životním cyklu View. Také LiveData respektují stav životního cyklu komponent aplikace a obsahují logiku čištění, která zabraňuje nadměrné spotřebě a úniku paměti.

- **ViewModel**

  ViewModel poskytuje data pro konkrétní UI komponenty a obsahuje byznys logiku pro komunikaci s Modelem. ViewModel nesmí měnit UI napřímo a také uchovávat referenci na View, protože musí být nezávislý na změně konfigurace, například po rotaci displeje.

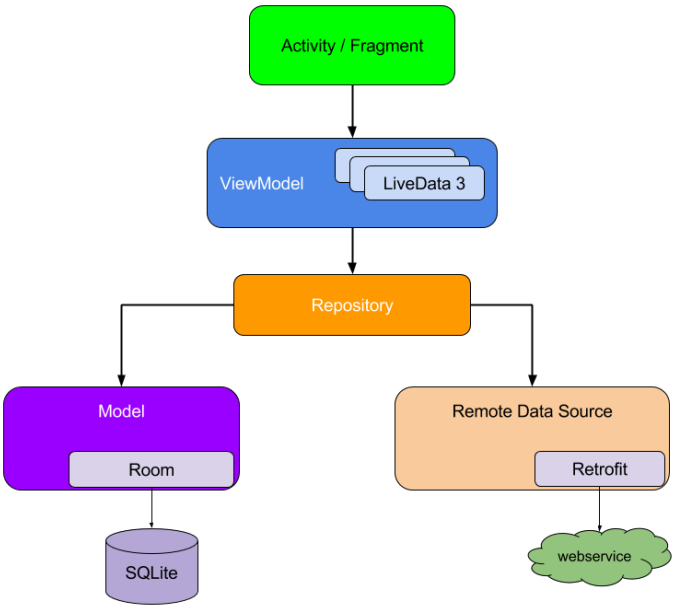
- **Lifecycle**

  Lifecycle umožňuje vytvářet komponenty, které jsou schopny rozpoznat životní cyklus a mohou automaticky přizpůsobit své chování na základě aktuálního životního cyklu Aktivity nebo Fragmentu.

- **Repository**

  Repository slouží pro zpracování datových operací, tento objekt ví, odkud se mají data dostat a jak je aktualizovat. Repository jsou mediátorem mezi různými zdroji dat, jako jsou lokální úložiště a webové služby.

Výsledný náhled architektury je možné vidět na obrázku 2.4. Díky oddělení odpovědností může být každý prvek jednoduše otestován.



Obrázek 2.4: Doporučená architektura pro Android aplikace, převzato z [19]

2.4.1.4 Závěr

V dnešní době rozhodnutí o architektuře mobilní aplikace pro Android je docela přímočaré, protože MVVM a Android Components jsou široce používané v praxi a navíc jsou doporučeným způsobem od Google.

Kapitola 3

Analýza požadavků

3.1 Definice požadavků

Pokud jde o vývoj nového systému nebo aplikace, prvním a nejdůležitějším krokem je přesně definovat požadavky, které budou popisovat funkcionalitu budoucí aplikace a vymezovat její rozsah. Sestavení požadavků na systém probíhalo s použitím předchozí analýzy podobných řešení a také výzkumu preferencí cílové skupiny.

Funkční požadavky se dělí do funkčních a nefunkčních.

3.1.1 Funkční požadavky

Funkční požadavky popisují chování aplikace na vysoké úrovni. Hlavním cílem funkčních požadavků je přesně a stručně popsat funkce systému a evidovaná data, definovat hranice aplikace. [21]

- **FP1. Správa účtu uživatele**

  Registrace bude povinnou podmínkou použití navržené aplikace a přístupu ke všem funkcím. Aplikace umožní registraci uživatele na serveru pomocí elektronické adresy. Již registrovaný uživatel se bude moct odhlašovat v libovolný okamžik a zase se přihlašovat pomocí dříve uvedených údajů.

- **FP2. Správa stránky psa**

  Aplikace umožní vytváření a správu stránky psa, která bude spojena s účtem uživatele. Vytvoření aspoň jedné stránky psa je neoddělitelnou částí použití aplikace. Aplikace bude umožňovat uživateli přidávat více stránek psů a pohodlně přepínat mezi nimi. Také aplikace umožní úpravu a mazání stránky psa.

- **FP3. Evidence příspěvků a fotografií psů**

  Aplikace bude evidovat příspěvky a fotografie přidané uživatelem na stránku psa. Aplikace umožní prohlížet fotografie a události jiných uživatelů.

- **FP4. Sestavení seznamu novinek podle stránek sledovaných uživatelem**

  Aplikace umožní prohlížet souhrn novinek sestavený ze všech nových příspěvků sledovaných uživatelem stránek.

- **FP5. Vyhledávání v profilech a plemenech**

  Aplikace umožní uživateli vyhledávat podle plemen a jména psů. Také aplikace bude umožňovat prohlížení stránek psů seskupených podle jednotlivých plemen.

- **FP6. Vzájemné hodnocení a sledování**

  Aplikace umožní označovat příspěvky jako oblíbené. Také aplikace dovolí sledovat psy pro účely zobrazování novinek z jejich života, které budou sdílené na jejich stránkách. Uživatel také bude moct prohlížet seznam sledujících a sledovaných psů. Dále aplikace bude umožňovat zanechání komentářů k příspěvkům.

- **FP7. Zobrazení nových hodnocení a komentářů**

  Aplikace bude upozorňovat uživatele na nové hodnocení a komentáře přidané k příspěvkům, formou seznamu všech událostí seřazených od nejnovějších. Stejným způsobem aplikace upozorní uživatele na nové sledující.

- **FP8. Sledování trasy procházky se psem**

  Aplikace umožní sledovat a vykreslovat na obrazovku trasu procházky se psem a také uvede základní informaci o ušlé vzdálenosti a času trvání.

- **FP9. Správa již uložených procházek**

  Aplikace bude evidovat minulé procházky a umožní uživateli sdílet trasu jako příspěvek na profil psa.

- **FP10. Vytváření a správa nabídek na společnou procházku**

  Aplikace umožní vytvářet nabídky na společné venčení psů s uvedením lokality a přesného času. Uživatelé budou moct prohlížet všechny dostupné nabídky a zanechávat komentáře. Také pro pohodlné vyhledávání bude moct uživatel filtrovat nabídky podle města.

3.1.2 Nefunkční požadavky

- **NFP1. Nativní Android aplikace**

  Aplikace bude napsána nativně a bude dodržovat obecné standardy pro Android aplikaci.

- **NFP2. Podpora operačního systému Android**

  Aplikace bude podporovat operační systém Android verze 4.4 a výše.

- **NFP3. Internetové připojení**

  Aplikace bude vyžadovat internetové připojení pro provedení všech dostupných akcí.

- **NFP4. Přístup k externímu úložišti**

  Aplikace bude vyžadovat přístup k externímu úložišti pro zobrazení fotografií ze zařízení a následnému nahrávání na server.

- **NFP5. Přístup k poloze**

  Aplikace bude vyžadovat povolení přístupu k poloze pro funkční požadavky FP8 až FP10, které jsou spojené s procházkami.

- **NFP6. Autorizace a autentizace**

  Aplikace musí zajišťovat proces bezpečného přihlášení a použití aplikace.

- **NFP7. Orientace na výšku**

  Aplikace bude podporovat jen orientaci na výšku.

- **NFP8. Jazyk aplikace**

  Jazykem aplikace bude angličtina.

- **NFP9. Uživatelská přívětivost**

  Systém bude intuitivní pro uživatele a nebude potřebovat návod k použití.

- **NFP10. Načítání**

  Aplikace bude správně reagovat na delší načítání dat ze serveru a zobrazovat načítací prvky.

3.2 Model případů užití

Vytváření modelu případů užití je další důležitý krok v návrhu softwarového produktu. Případy užití pomáhají pochopit problémy, které řeší aplikace, přesně popisují funkcionalitu a způsob, jak se výsledný systém bude používat.

Model případů užití nejčastěji zobrazují ve tvaru UML diagramu, který umožňuje vizualizaci případů užití a jejich vztahu. Vytváření modelu případů užití je velmi důležitým krokem v komunikaci s klientem, jelikož přesné definování požadavků je vždy nejtěžším místem většiny projektů. [22]

Model případů užití se skládá z následujících částí.

- Případy užití

  Jednotlivé funkčnosti, které popisují chování systému.

- Vztahy mezi případy užití

  Případy užití můžou mít mezi sebou dva typy vztahů – extend a include. Include se používá, pokud chceme znázornit, že jistý případ užití zahrnuje jiný anebo při popisu společných akcí. Extend naopak volitelně rozšiřuje jeden případ užití pomocí druhého.

- Aktéři

  Aktérem v modelu případů užití je každá entita, která vykonává nějakou roli v daném systému.

- Vztahy mezi aktéry

  Mezi aktéry může existovat vztah generalize.

- Vztahy mezi aktéry a případy užití

  Spojuje aktéry a případy užití.

- Scénáře (volitelně)

  Scénáře popisují podrobné chování při plnění nějakého případu užití. Většinou slouží pro lepší pochopení složitějších případů užití.

Diagram případů užití je znázorněný na obrázku 3.1 a 3.2.

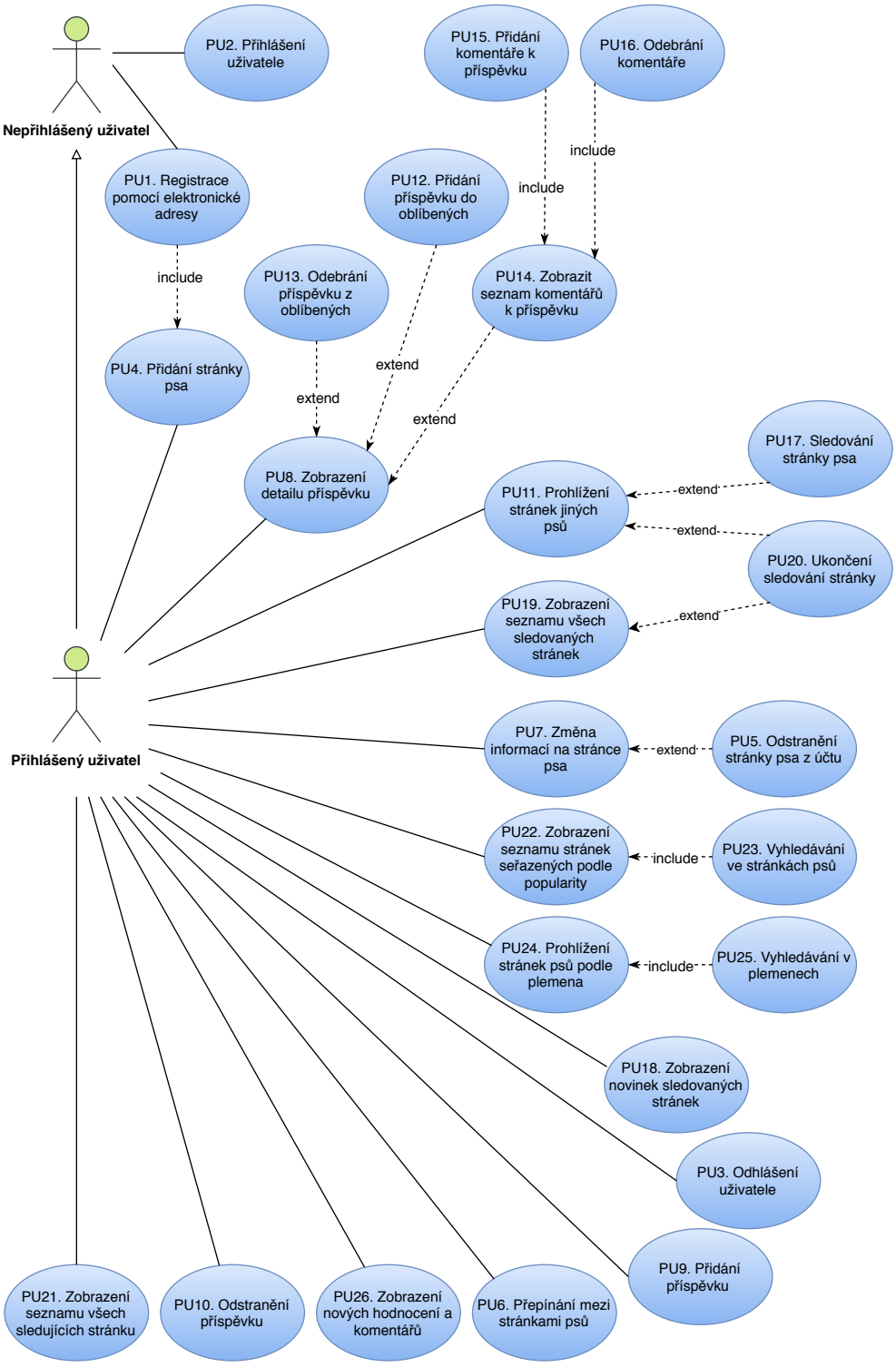
3.2.1 Aktéři

3.2.1.1 Nepřihlášený uživatel

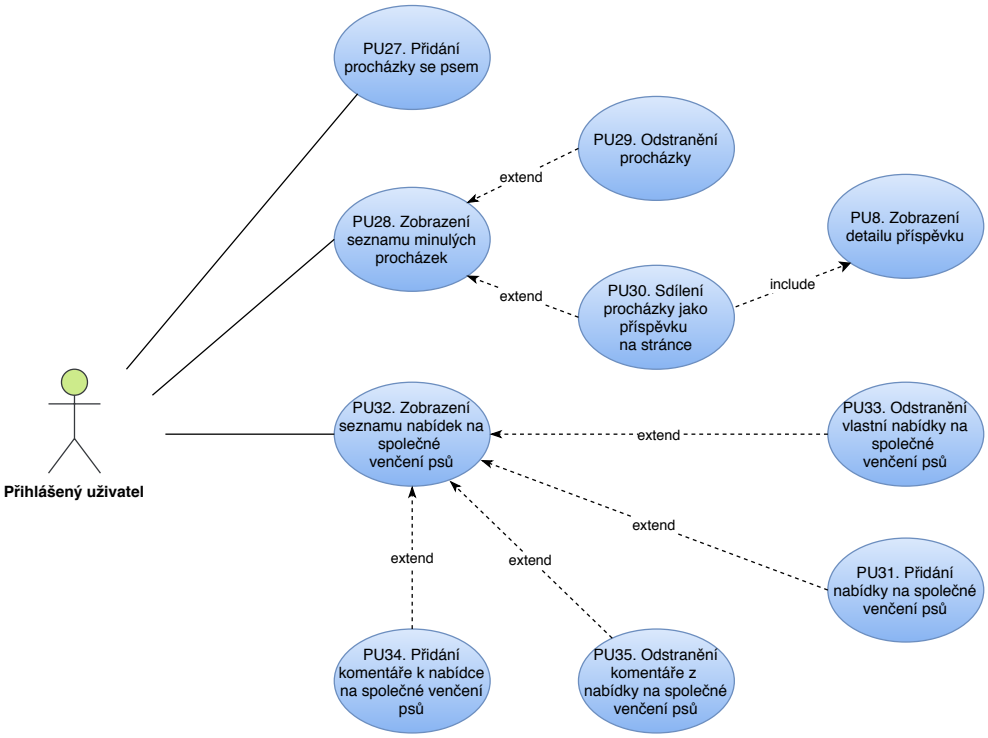
Jelikož používání aplikace je podmíněné registrací, nepřihlášený uživatel v navržené aplikaci se bude moct jen zaregistrovat a následně přihlásit.

3.2.1.2 Přihlášený uživatel

Přihlášený uživatel může provádět libovolnou funkci v navržené aplikaci, včetně všech sociálních funkcí jako například vedení stránky psa, sdílení příspěvků, prohlížení stránek ostatních uživatelů, vytváření procházek a jiné.



Obrázek 3.1: Diagram případů užití - část 1



Obrázek 3.2: Diagram případů užití - část 2

3.2.2 Případy užití

- **PU1. Registrace pomocí elektronické adresy**

  Aplikace umožní vytvořit účet uživatele pomocí elektronické adresy a hesla. Registrace bude umožněna jen tehdy, pokud uvedené údaje budou splňovat základní pravidla a také na serveru nebude registrovaný uživatel se stejnou elektronickou adresou. Nezbytnou částí registrace je přidání první stránky psa, bez tohoto kroku uživatel sice bude zaregistrovaný, ale nebude se moct dostat do aplikace.

- **PU2. Přihlášení uživatele**

  Aplikace umožní přihlášení existujícího uživatele pomocí dříve uvedených údajů (elektronické adresy a hesla).

- **PU3. Odhlášení uživatele**

  Aplikace umožní odhlášení již přihlášeného uživatele.

- **PU4. Přidání stránky psa**

  Po registraci aplikace požádá uživatele vytvořit aspoň jednu stránku psa. Každý pes v aplikaci povinně musí mít uvedené jméno, plemeno, pohlaví a přibližný věk. Pro uživatele, kteří nebudou vědět plemeno,

ke kterému patří jejich pes, bude k dispozici obecné plemeno. Nepovinné údaje jsou datum narození, fotografie a základní informace. Aplikace umožní přidání dalších psů, kteří budou spojeni s účtem uživatele.

Základní cesta:

1. Případ užití se začíná, když uživatel chce přidat stránku psa.
2. Aplikace požádá uživatele o uvedení jména nového psa.
3. Uživatel uvede jméno psa.
4. Aplikace se zeptá uživatele na pohlaví domácího mazlíčka.
5. Uživatel zvolí pohlaví a stiskne pokračovat.
6. Aplikace požádá uživatele o uvedení plemena psa.
   a) Uživatel zvolí požadované plemeno psa.
   b) Uživatel neví přesný název plemena svého psa, a proto zvolí možnost „nevím“.
7. Aplikace požádá uživatele o nahrání fotografie svého domácího mazlíčka.
   a) Uživatel nahraje fotografii.
   b) Uživatel nechce nahrávat fotografii a přeskočí tento krok.
8. Aplikace požádá uživatele uvést datum narození psa.
   a) Uživatel uvede přesný datum narození.
   b) Uživatelé zvolí přibližný věk svého psa.
9. Aplikace požádá uživatele uvést nějakou informaci o svém psovi.
   a) Uživatel uvede informace.
   b) Uživatel přeskočí tuto sekci.
10. Aplikace vytvoří stránku psa podle uvedených údajů a otevře ji uživateli.

- **PU5. Odstranění stránky psa z účtu**

  Aplikace umožní odstranit stránku. Vymazání stránky znamená ztrátu všech dat o psovi.

- **PU6. Přepínání mezi stránkami psů v rámci účtu uživatele**

  Aplikace umožní se přepínat mezi stránkami v rámci rodiny psů spojených se stejným účtem uživatele.

- **PU7. Změna informací na stránce psa**

  Aplikace umožní upravovat informace na stránce svého psa.

- **PU8. Zobrazení detailu příspěvku**

  Aplikace umožní zobrazovat detail události v seznamu ostatních událostí psa.

- **PU9. Přidání příspěvku**

  Aplikace umožní přidání události na stránku psa. Příspěvek se může skládat z kategorie, fotografie a popisu. Uvedení fotografie nebo popisu je povinné.

- **PU10. Odstranění příspěvku**

  Aplikace umožní odstranit příspěvek ze stránky svého psa.

- **PU11. Prohlížení stránek jiných psů**

  Aplikace umožní prohlížet dostupné informace a všechny příspěvky vybraného psa. Také uživatel bude moct vidět stránky psů spojených se stejným účtem.

- **PU12. Přidání příspěvku do oblíbených**

  Aplikace umožní označit příspěvek jako oblíbený.

- **PU13. Odebrání příspěvku z oblíbených**

  Aplikace umožní smazat příspěvek z oblíbených, pokud před tím byl přidaný.

- **PU14. Zobrazit seznam komentářů k příspěvku**

  Aplikace umožní zobrazit všechny komentáře k vybranému příspěvku.

- **PU15. Přidání komentáře k příspěvku**

  Aplikace umožní přidat komentář k příspěvku.

- **PU16. Odebrání komentáře**

  Aplikace umožní smazat vlastní komentář k příspěvku.

- **PU17. Sledování stránky psa**

  Aplikace umožní uživateli sledovat stránku psa. Příspěvky sledovaných psů se budou zobrazovat v novinkách.

- **PU18. Zobrazení novinek sledovaných stránek**

  Aplikace bude zobrazovat všechny nové příspěvky ze stránek, které sleduje uživatel.

- **PU19. Zobrazení seznamu všech sledovaných stránek**

  Aplikace umožní zobrazit seznam všech sledovaných stránek psů. Z tohoto seznamu bude možné snadno řídit své sledované stránky.

- **PU20. Ukončení sledování stránky**

  Aplikace umožní ukončit sledování stránky psa. Události stránky, která byla zrušená ze sledovaných, se dále již nebudou zobrazovat v novinkách.

  Základní cesta:

  1. Případ užití se začíná, kdy uživatel na stránce s novinkami uvidí příspěvky ze stránky psa, kterého už nechce sledovat.
  2. Aplikace zobrazí příspěvek sledovaného psa v seznamu všech novinek.
  3. Uživatel přejde na stránku psa.
  4. Aplikace zobrazí stránku sledovaného psa, včetně všech jeho příspěvků a informací.
  5. Uživatel zruší sledování daného psa.
  6. Aplikace vymaže psa ze seznamu sledovaných a po obnovení smaže všechny jeho příspěvky ze stránky s novinkami.

  Alternativní cesta:

  1. Případ užití se začíná, kdy uživatel chce zrušit sledování psa.
  2. Aplikace zobrazí stránku vlastního psa uživatele.
  3. Uživatel přejde na seznam všech sledovaných stránek a najde psa, kterého chce přestat sledovat.
  4. Aplikace zobrazí seznam všech sledovaných stránek.
  5. Uživatel zruší sledování daného psa.
  6. Aplikace vymaže psa ze seznamu sledovaných a po obnovení smaže všechny jeho příspěvky ze stránky s novinkami.

- **PU21. Zobrazení seznamu všech sledujících stránku**

  Aplikace umožní zobrazit seznam všech sledujících stránku psa.

- **PU22. Zobrazení seznamu stránek seřazených podle popularity**

  Aplikace umožní objevení nových zajímavých stránek psů seřazených podle popularity. Popularita se měří pomocí počtu sledujících.

- **PU23. Vyhledávání ve stránkách psů**

  Aplikace umožní vyhledávání podle jména nebo plemena ve stránkách psů.

- **PU24. Prohlížení stránek psů podle plemena**

  Aplikace bude umožňovat objevovat nové zajímavé stránky ke sledování pomocí filtrování podle plemen.

- **PU25. Vyhledávání v plemenech**

  Aplikace umožní vyhledávání podle plemena ve stránkách psů.

- **PU26. Zobrazení nových hodnocení a komentářů**

  Pro lepší přehled aplikace zobrazí nová hodnocení a komentáře k příspěvkům všech stránek psů uživatele seřazené podle času přidání.

- **PU27. Přidání procházky se psem**

  Aplikace bude umožňovat sledovat procházku se psem a vykreslovat trasu na mapě. Po skončení nahrávání trasy venčení psa aplikace uloží procházku. Aplikace bude v průběhu uvádět základní informace o ušlé vzdálenosti a času trvání.

  Základní cesta:

  1. Případ užití se začíná, kdy uživatel chce vykreslovat na mapu trasu procházky se psem v průběhu venčení.
  2. Aplikace nabídne uživateli začít zpracování nové procházky.
  3. Uživatel začne sledování nové procházky se psem.
  4. Aplikace začne sledování, ukáže mapu a bude vykreslovat trasu. Také aplikace bude uvádět a aktualizovat informace o ušlé vzdálenosti a stráveném čase.
  5. Uživatel zavře aplikaci a bude se věnovat venčení psa. Po skončení procházky uživatel otevře aplikaci a stiskne tlačítko ukončení sledování trasy.
  6. Aplikace ukáže obrazovku se shrnutím procházky.
  7. Uživatel uloží procházku.
  8. Aplikace uloží procházku na účet uživatele na serveru a zeptá se, jestli uživatel chce sdílet procházku jako příspěvek na stránce svého psa.
  9. Uživatel odsouhlasí sdílení.

- **PU28. Zobrazení seznamu minulých procházek**

  Aplikace umožní zobrazit seznam svých minulých procházek. Tyto procházky budou dostupné jen pro vlastníka a nebudou viditelné pro ostatní.

- **PU29. Odstranění procházky**

  Aplikace umožní odstranit již uloženou procházku.

- **PU30. Sdílení procházky jako příspěvku na stránce**

  Aplikace umožní sdílet uloženou trasu na stránce psa jako příspěvek. Tím uživatel zpřístupní trasu procházky pro ostatní uživatele.

- **PU31. Přidání nabídky na společné venčení psů**

  Aplikace umožní přidat nabídku na společné venčení psů. Nabídka bude obsahovat přesný čas a polohu na mapě pro setkání.

  Základní cesta:

  1. Případ užití se začíná, kdy uživatel chce zveřejnit nabídku na společné venčení psů.
  2. Aplikace zobrazí obrazovku, kde uživatel může vyplnit potřebné informace.
  3. Uživatel zvolí polohu na mapě, kde se chce setkat a zvolí přesný datum a čas, také může napsat k nabídce nějaký komentář. Uživatel uloží nabídku.
  4. Aplikace uloží nabídku a zveřejní ji pro všechny uživatele aplikace.

  Alternativní cesta:

  1. Případ užití se začíná, kdy uživatel chce zveřejnit stejnou nabídku jako je nějaká z předchozích, ale s jiným časem.
  2. Aplikace zobrazí všechny nabídky uživatele na společné venčení.
  3. Uživatel zvolí nabídku, kterou chce zopakovat, upraví čas a zveřejní ji jako novou nabídku.
  4. Aplikace uloží nabídku a zveřejní ji pro všechny uživatele aplikace.

- **PU32. Zobrazení seznamu nabídek na společné venčení psů**

  Aplikace umožní zobrazit seznam všech dostupných nabídek na společné venčení psů. Uživatel bude moct zobrazit nabídky jen odpovídající zvolenému městu. Odděleným seznamem uživateli budou zobrazené jeho vlastní nabídky.

- **PU33. Odstranění vlastní nabídky na společné venčení psů**

  Aplikace umožní uživateli odstranit vlastní nabídku na společné venčení psů.

- **PU34. Přidání komentáře k nabídce na společné venčení psů**

  Aplikace umožní přidání komentářů k nabídkám na společné venčení psů.

- **PU35. Odstranění komentáře z nabídky na společné venčení psů**

  Aplikace umožní odstranění již přidaného vlastního komentáře k nabídce na společné venčení psů.

3.2.3 Pokrytí případů užití

Funkční požadavky a model případů užití jsou úzce propojené a je nezbytné, aby všechny funkční požadavky byly pokryté aspoň jedním případem užití. Toto je možné ověřit pomocí tabulky pokrytí funkčních požadavků, kterou je možné vidět v příloze C.

3.3 Diagramy aktivit

Diagram aktivit je dalším druhem UML diagramů, který je určen pro popis procesů v aplikaci a definování toku jednotlivých aktivit a akcí. V navržené aplikaci jedněmi z nejdůležitějších procesů jsou přidání komentáře a sledování trasy procházky se psem.

3.3.1 Přidání příspěvku

Diagram aktivit znázorňující přidání příspěvku je možné vidět v příloze na obrázku D.1. Na tomto diagramu je vidět celkový tok akcí, které uživatel může anebo musí provést pro přidání příspěvku. Například na diagramu je zachyceno, že uživatel nemůže přidat příspěvek bez textové zprávy a zároveň bez obrázku. Také uživatel má na vybranou, jakým způsobem zvolit obrázek pro příspěvek: pomocí galerie, pořídí novou fotografii nebo zvolí nějaký obrázek z nedávných.

3.3.2 Sledování trasy procházky se psem

Diagram aktivit popisující sledování trasy procházky se psem je možné si prohlédnout v příloze na obrázku D.2. Aby mohl začít sledovat trasu procházky, uživatel musí povolit přístup k poloze a zapnout funkci umístění ve svém zařízení, protože aplikace nemůže provádět žádné trekovací činnosti bez těchto povolení. Když mobilní aplikace dostane povolení, tak se spustí služba, která bude zodpovědná za sledování trasy. Po skončení procházky se služba zastaví a aplikace zobrazí shrnutí. Po potvrzení pošle aplikace požadavek na uložení na server a zároveň se zeptá uživatele, jestli chce sdílet tuto procházku jako příspěvek na stránce psa, který se zúčastnil procházky.

Kapitola 4

Návrh

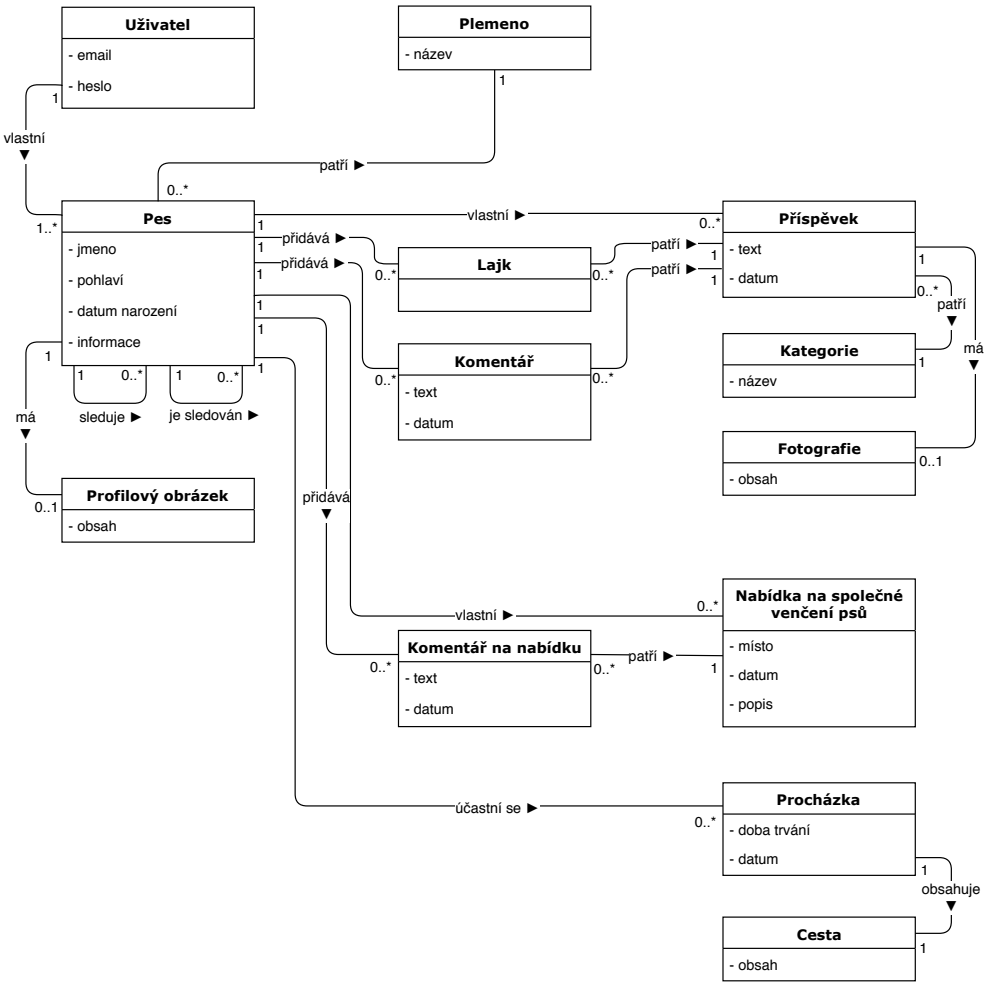
4.1 Doménový model

Modelování domény je dalším krokem, jak přejít od popisu chování systému do návrhu struktury. Doménový model identifikuje relevantní pojmy, myšlenky a koncepty problémové domény navrhované aplikace. Hlavním cílem tohoto modelu je rozložit problémovou doménu na objekty nebo pojmy z reálného světa a popsat vztahy mezi nimi [23]. Tento krok pomůže poskytnout základ pro další návrh aplikace a usnadní návrh databázového modelu.

Doménový model navrhované aplikace je možné vidět na obrázku 4.1. Základní entitou v navrženém systému je pes, který hraje nejdůležitější roli a je terčem všech akcí. Pes není samostatný sám o sobě, jelikož nemůže existovat bez majitele, který bude udržovat stránku a používat aplikaci. Pes vždy patří k nějakému plemenu a může mít profilový obrázek pro hezčí reprezentaci své stránky. Také pes může někoho sledovat a být sledován, což je základní funkcí sociálních sítí a pomáhá tvořit seznam nových příspěvků sledovaných stránek. Pes může přidávat příspěvky do oblíbených a zanechávat komentáře pod nimi. Každý příspěvek může patřit k nějaké kategorii a obsahovat obrázek. Také pes se může zúčastnit procházky a nabízet ostatním vlastníkům se sejít pro společné venčení.

4.2 Návrh uživatelského rozhraní

Návrh uživatelského rozhraní je jedním z nejdůležitějších faktorů úspěchu aplikace, obzvlášť v případě, kdy se mluví o sociální síti. Pro návrh uživatelského rozhraní je důležité především pochopit potřeby a motivaci lidí, kteří budou používat výsledný produkt. Dalším bodem, který je potřeba si uvědomit, jsou doménová omezení a technické požadavky, což spolu s prvním bodem tvoří nutný základ pro vytvoření návrhu uživatelského rozhraní. Hlavním požadavkem je, aby design byl tvořen pro lidi, kteří budou používat výsledný produkt.



Obrázek 4.1: Doménový model

Toto pomůže vytvořit uživatelsky orientované rozhraní, což je příznak kvalitního designu. [26]

Podle [26] hlavními důvody, proč o výsledný digitální produkt lidé nemají zájem, jsou:

- Nezdařilé rozdělení priorit v průběhu vývoje
- Ignorování uživatelských potřeb a požadavků
- Absence nebo nedostatek designového procesu, který umožňuje shromažďovat, analyzovat a využívat znalosti o potřebách uživatele, čímž je možné řídit pro evoluci návrhu uživatelského rozhraní

V předchozích sekcích již byly zkoumané uživatelské potřeby a požadavky, také bylo věnováno dost času analýze problémové domény a technických po-

žadavků, takže je možné začít s návrhem uživatelského rozhraní. Designový proces bude popsaný od začátku, kde budou popsané hlavní principy, kterými byl řízen návrh až po návrh hi-fi prototypů, které budou obsahovat finální design aplikace.

4.2.1 Základní principy použitelného uživatelského rozhraní

V knize Usable Usability [27] byly popsané nejdůležitější principy tvorby použitelného uživatelského rozhraní.

- Funkční design (Functional)

  Především aplikace musí fungovat a splňovat cíle, pro které byla vytvořena. V aplikaci nesmí být nefunkční prvky a odezva aplikace musí být akceptovatelná uživatelem. Také v aplikaci musí být promyšlené i okrajové případy, aby se aplikace nedostala do nekonzistentního stavu.

- Responzivní design (Responsive)

  Responzivní aplikace reprezentuje koncept dvoucestné komunikace, kdy uživatel provede nějakou akci na svém zařízení a okamžitě dostane odezvu, buď výsledek této činnosti nebo zobrazení načítacího prvku. Responzivní design by se měl také přizpůsobovat různým velikostem displeje, pro které je určen.

- Ergonomický design (Ergonomic)

  V tomto principu jde o to, aby použití aplikace bylo co nejvíce jednodušší při minimálním úsilí uživatele, které jsou potřeba k provedení různých akcí. Například tlačítka musí být dostatečně velká, proto, aby uživatel nepotřeboval cíleně klikat na nějakou malou oblast obrazovky. Také často používané scénáře by měly obsahovat jen opravdu nutné kroky.

- Praktický design (Convenient)

  Použití aplikace musí být pohodlné, uživatel by neměl dlouho hledat způsob navigace k potřebné sekci nebo zadávat stejné informace několikrát. Hlavní snahou tohoto bodu je dosažení stavu, kdy použití aplikace není spojené s hledáním potřebných prvků nebo akcí.

- Jednoduchý design (Foolproof)

  Při návrhu uživatelského rozhraní nelze předpokládat, že uživatel bude ideálně ovládat platformu nebo se vyzná v doméně. Proto správný design se musí snažit předejít chybám, a pokud se k ní dostalo, tak nabídnout jednoduchý způsob řešení. Také je známým faktem, že lidé často nečtou návody a nápovědy, a proto je potřeba poskytovat uživateli jen nutné informace, které on může jednoduše pochopit. Také je potřeba mít na mysli, že lidé často zapomínají na různé věci, a proto správný návrh

uživatelského rozhraní musí být intuitivní a nesmí nutit uživatele si pamatovat příliš mnoho informací.

- Viditelný design (Visible)

  Jedna z nejhorších věcí, kterou je možné udělat v návrhu uživatelského rozhraní, je vytvoření nějaké funkčnosti, ke které se bude složité dostat nebo obtížné ji použít. Tento problém je spojen s principem viditelnosti, který může být porušen buď tím, že funkčnost bude schovaná někde v aplikaci nebo nebude rozpoznatelná, i když se bude nacházet přímo před očima uživatele. Také je potřeba klást důraz na to, aby důležité informace byly zvýrazněné a navíc nevypadaly jako prvek, který je možné přeskočit.

- Pochopitelný design (Understandable)

  Je známým faktem, že v případě jazykového vyjádření je možné říct stejnou větu různými způsoby, kde některé mohou být nezdařilé nebo nepochopitelné. Stejně funguje návrh uživatelského rozhraní, kde je možné například vymýšlet nepochopitelné chybové hlášky nebo použít nevhodnou ikonku. V ideálním případě je potřeba docílit toho stavu, kde uživatel, který nemá nic společného s doménou nebo platformou, může snadno pochopit každý prvek navrženého uživatelského rozhraní.

- Logický design (Logical)

  Logický návrh uživatelského rozhraní by neměl stavět před uživatelem otázku „proč“. To se týká jak otázky „proč se to tak stalo“, tak i otázky „proč se to tak nestalo“. Scénáře průchodů aplikací musí dávat smysl, protože pokud se tam vyskytne nějaký nelogický prvek, který například nepřiblíží uživatele k splnění cíle, tak on bude zmatený a nejspíše zavře aplikaci.

- Konzistentní design (Consistent)

  Ve správném návrhu uživatelského rozhraní je potřeba se držet stejných označování, například stejné prvky nesmí odpovídat za různé funkce. Použité prvky by měly být konzistentní se světem, ve kterém žijí uživatelé.

- Předvídatelný design (Predictable)

  Uživatel by měl vždy vědět, co očekávat od akcí, které bude provádět. V žádném případě by nemělo dojít k situaci, kdy uživatel bude náhodně klikat na tlačítka bez tušení, co se má stát. Uživatel by měl vždy vědět, který scénář splní jeho cíl.

Dodržovat v plné míře výše popsaná pravidla je nemožné, protože není možné vyhovět všem a navíc každý uživatel má svůj názor na správný návrh.

Ale je důležité při návrhu uživatelského rozhraní se co nejvíce držet těchto pravidel a snažit se docílit nejlepšího výsledku.

Další návrh uživatelského rozhraní bude probíhat hlavně podle přednášek [28] magisterského předmětu „Návrh uživatelských rozhraní“ přednášených na Fakultě informačních technologií Českého vysokého učení technického v Praze. Při návrhu budou zohledněné principy a znalosti Android platformy, materiálního designu a výše popsaná pravidla.

4.2.2 Základní definice produktu

Základní definice produktu slouží k prohlášení k čemu a pro koho je produkt určen. Definice aplikace musí obsahovat tři základní informace: název produktu, jeho účel a komu je určen. Již na této fázi je potřeba vymyslet název produktu. Proto bylo rozhodnuto, že aplikace bude mít název „Woof!“, což je dostatečně krátké a již v názvu je možné vidět, že tématem této aplikace jsou psy.

„Woof!“ je mobilní sociální síť pro majitele psů, která slouží ke sdílení významných okamžiků ze života svých domácích mazlíčků a sledování trasy procházek.

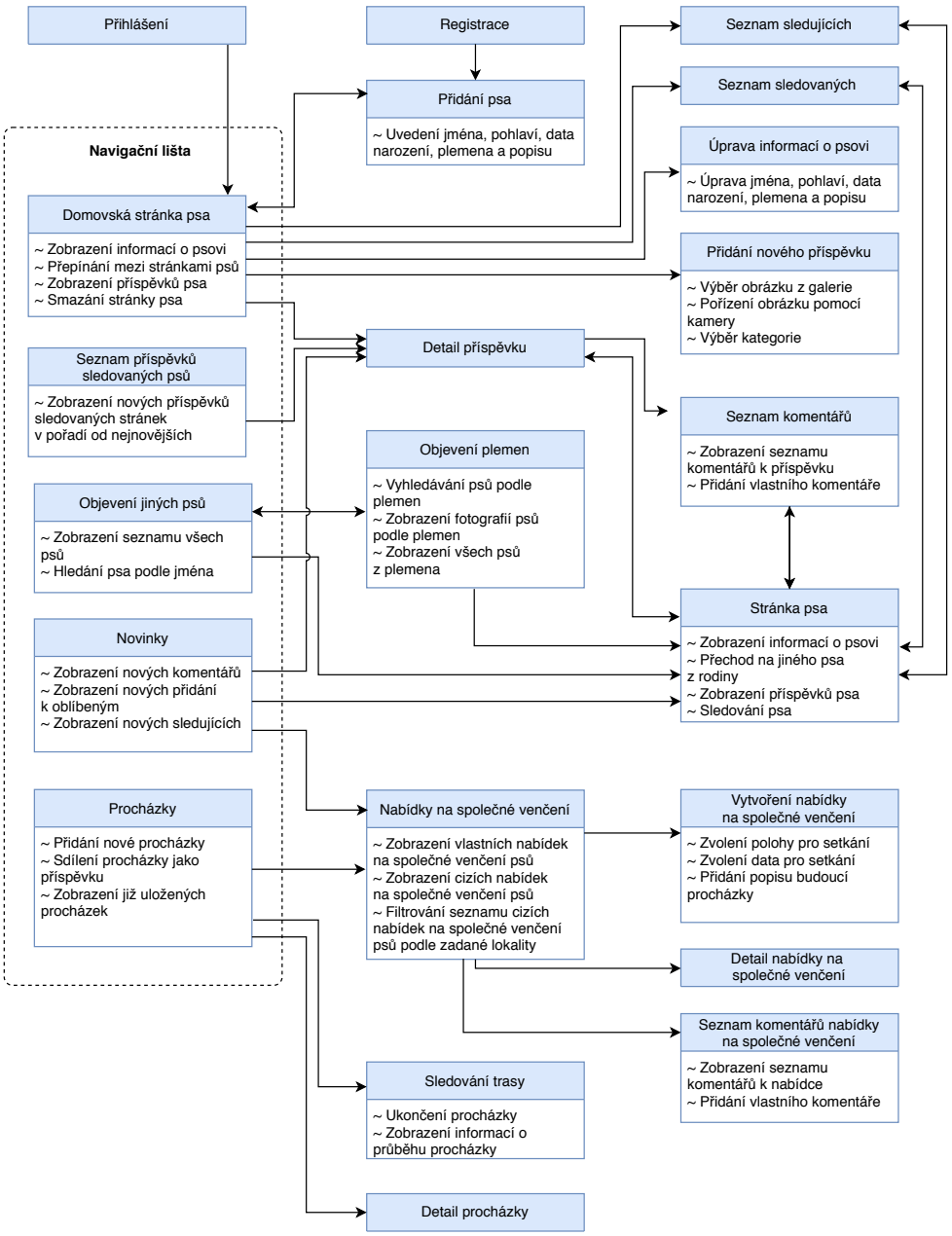
4.2.3 Seznam úkolů aplikace

Seznam úkolů aplikace (anglicky task list) je určen k základnímu definování všech možných činností v aplikaci z uživatelského pohledu. Jednotlivé úkoly vycházejí z předchozí analýzy, návrhu požadavků a případů užití. Úkoly je vhodné rozdělit do logických skupin podle části aplikace, kterých se tato úloha týká, což bude možné použít pro budoucí návrh jednotlivých obrazovek. Dalším krokem je vhodné označit, jestli úloha bude zobrazovací [Z] nebo ovládací [O]. Také je možné ohodnotit častost výskytu těchto úkolů ve výsledné aplikaci, což bude znázorněné pomocí škály od 1 (nejméně používaný) do 5 (nejvíce používaný). Seznam úkolů aplikace je možné najít v příloze E.

4.2.4 Analýza úkolů aplikace

Vytvoření seznamu úkolů aplikace bylo prvním krokem ke správnému návrhu uživatelského rozhraní. Pomocí vytvořeného seznamu je možné určit základní obrazovky a rozložení důležitých prvků. V první fázi analýzy pomocí již sepsaných úkolů aplikace byl vytvořen graf funkčnosti uživatelského rozhraní. Tento graf znázorňuje jednotlivé obrazovky a přechody mezi nimi a zároveň popisuje nejdůležitější funkce, které musí obsahovat daná část aplikace. Graf funkčnosti uživatelského rozhraní je si možné prohlédnout na obrázku 4.2.

Nejdůležitějším rozhodnutím bylo zvolit hlavní obrazovky, které budou mít bod přístupu přes dolní navigační lištu (Bottom Navigation Bar). Tyto obrazovky musí obsahovat nezbytné funkce pro aplikaci „Woof!“. Již od začátku



Obrázek 4.2: Graf funkčnosti uživatelského rozhraní

bylo jasné, že žádná sociální síť se neobejde bez stránky uživatele a seznamu nových příspěvků ze sledovaných stránek, proto tyto dvě obrazovky byly přidané do navigační lišty na začátku návrhu. Další nezbytnou částí aplikace jsou procházky se psem, které také zaslouží vlastní místo v navigační liště.

Volba dalších obrazovek byla složitější, protože ostatní funkce mají menší význam a nemusí mít vlastní tlačítko v navigační liště. První funkce, která připadala v úvahu, byla stránka s nabídkami na společné venčení psů. Ale jelikož podle dotazníku tato funkčnost neměla velký zájem mezi potenciálními uživateli, bylo vyřešeno, že tato část aplikace musí být sloučena se sekcí s procházkami pomocí záložek (Tab Bar).

Jelikož aplikace je tvořena jako nová sociální síť, tak je možné předpokládat, že by uživatelé nejspíš měli zájem o hledání nových stránek pro sledování. Proto bylo vyřešeno, že čtvrtou sekcí v navigačním panelu bude „objevení“, kde si uživatelé budou moct vyhledat stránky jiných uživatelů.

Prvotní návrh obsahoval jen čtyři body přístupu přes navigační lištu, ale pak byl přidán pátý prvek, který byl zodpovědný za zobrazování novinek, jako například přidání příspěvku uživatele k oblíbeným nebo napsání komentáře pod příspěvkem uživatele.

4.2.5 První návrh uživatelského rozhraní

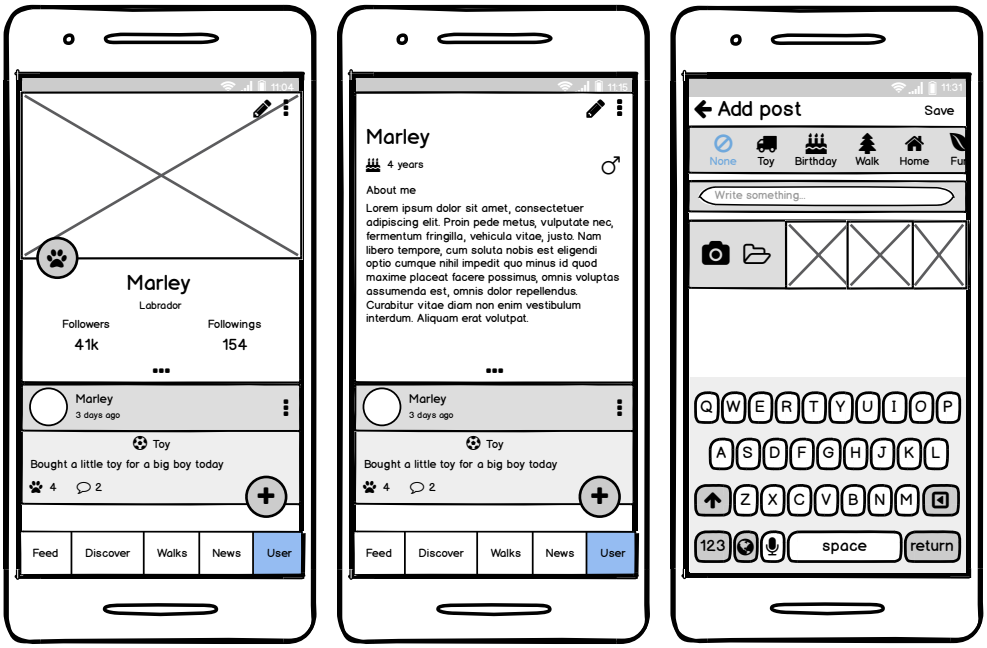
První návrh uživatelského rozhraní vycházel z výstupu analýzy úkolů a grafu funkčnosti uživatelského rozhraní. Tento návrh byl vytvořen ve formě lo-fi prototypu, který reprezentuje důležité koncepty aplikace, obsahuje základní ovládací prvky a jejich rozmístění. Tento prototyp neslouží pro zobrazení grafických částí aplikace a jeho hlavním cílem je zobrazit prvotní návrh uživatelského rozhraní, který může být předmětem další analýzy a vstupem do dalšího kola návrhu uživatelského rozhraní.

4.2.5.1 Profilová stránka psa

Uživatelský profil je hlavní částí libovolné sociální sítě a jeho hlavním cílem je co nejlépe reprezentovat identitu uživatele nebo psa v případě vyvíjené aplikace. Docílit náležité reprezentace je možné pomocí vhodného rozložení prvků obsahujících informaci o psovi jako například jméno, profilový obrázek, plemeno a další.

Obrazovka obsahující profilovou stránku psa se skládá ze dvou částí, a to z hlavičky s informacemi o psovi a seznamu s příspěvky daného psa. Dalšími nezbytnými prvky na této obrazovce musí být akce přidání nového příspěvku, změny psa v rámci účtu, úpravy informací a odhlášení.

Hlavička uživatelského profilu se také bude skládat ze dvou částí: část obsahující nezbytné informace a část s nezávaznými informacemi, které uživatel může přeskočit nebo uvést nepřesně. Při přechodu na stránku psa uživatel vždy uvidí profilový obrázek, jméno psa, plemeno, ke kterému se vztahuje a



(a) Profilová stránka (b) Profilová stránka - 2 (c) Přidání příspěvku

Obrázek 4.3: Lo-fi prototypy: stránka uživatele a přidání příspěvku

také počet sledovaných a sledujících. Počty sledovaných a sledujících jsou zároveň navigačním prvkem k odpovídajícím seznamům. Pokud se bude uživatel chtít dozvědět víc informací o psovi, tak si to může prohlédnout v následující záložce karuselu.

Jelikož uživatel může spravovat v rámci jednoho účtu více stránek psů, tak bylo potřeba přidat prvek, který by byl zodpovědný za přepínání stránek. Tento prvek byl přidán do hlavičky profilové stránky psa, kde na vlastní stránce uživatel může přepínat mezi stránkami svých psů a na cizí stránce si uživatel může prohlédnout rodinu daného psa (stránky, které jsou spojené se stejným uživatelem).

Přidání nového příspěvku bylo navržené ve formě FABu (Float Action Button) v pravém dolním rohu. Na profilové stránce cizího psa je tento prvek zodpovědný za začátek (případně konec) sledování.

První návrh uživatelského rozhraní profilové stránky, včetně dvou stavů hlavičky profilu, je možné vidět na obrázcích 4.3a a 4.3b.

4.2.5.2 Stránka přidání příspěvku

Přidání příspěvku je jedna z potenciálně nejčastějších akcí, kterou uživatel bude potřebovat v rámci používání této aplikace. Jelikož příspěvek může obsahovat kategorii, tak bylo potřeba vymyslet vhodný návrh pro její volbu. První myšlenkou bylo implementovat rozbalovací nabídku kategorií, což ale

není pohodlný prvek pro uživatele v tomto případě, protože vyžaduje klik navíc pro zobrazení všech možností. Proto tento problém byl vyřešen pomocí vertikálního posuvného seznamu, který umožňuje pohodlnou volbu kategorie.

Další zajímavou myšlenkou bylo zohlednění faktu, že uživatelé nejčastěji chtějí sdílet nějakou z posledních fotografií, a proto bylo by vhodné zobrazovat posledních deset až dvacet fotografií z galerie na této obrazovce.

Lo-fi návrh této obrazovky je si možné prohlédnout na obrázku 4.3c.

4.2.5.3 Přidání nového psa

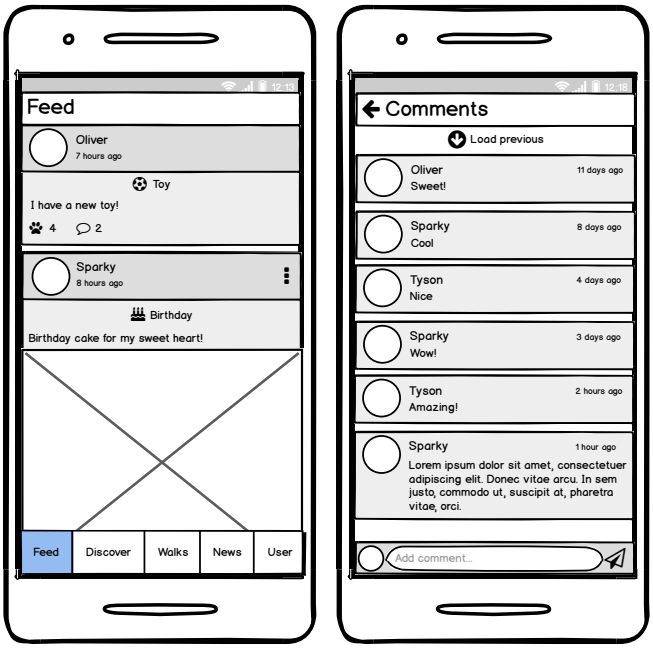
Přidání nového psa je akce, kterou musí provést každý uživatel této aplikace při registraci a pak při přidání dalších psů. Bylo vyřešeno, že uvedení údajů o novém psovi se bude provádět ve formě „onboardingu“, kde se aplikace bude pokrokově ptát na různé důležité údaje o novém členovi sociální sítě. Pro každý nový údaj se objeví nová obrazovka. Tento scénář musí být co nejméně vtíravý a musí umožňovat přeskočit všechno, co opravdu není důležité. Pokud uživatel neví, k jakému plemenu patří jeho pes, může zvolit možnost „nevím“, aplikace si s tím poradí tak, že bude mít speciální typ plemena pro tento případ. Aplikace také bude umožňovat uvést přibližný věk psa, pokud uživatel nebude vědět přesný datum narození. Kroky, jako jsou přidání obecného popisu a profilového obrázku, je možné snadno přeskočit.

4.2.5.4 Stránka se seznamem příspěvků sledovaných stránek

Tento seznam se bude skládat z různých příspěvků, kde každý příspěvek kromě informací o psovi a data přidání může obsahovat popis, obrázek a kategorii. Protože informace o vlastníkovi příspěvku je nezbytná, příspěvek bude rozdělen na dvě části: hlavička, kde bude zobrazena hlavní informace a tělo příspěvku, kde alespoň jeden atribut musí být uveden, protože přidání prázdného příspěvku je zakázáno. Lo-fi prototyp obrazovky se seznamem příspěvků je možné vidět na obrázku 4.4a.

4.2.5.5 Stránka s komentáři

Uživatel může zanechat komentář pod libovolným příspěvkem a nabídkou na společné venčení, kde jejich počet na jedné stránce může být poměrně velký. Proto kromě návrhu samotných komentářů a políčka pro přidání vlastního komentáře bylo potřeba dobře promyslet návrh stránkování. První myšlenka byla implementovat seznam, který se bude začínat nejstarším komentářem a při projíždění seznamu se budou zobrazovat novější prvky. Ale tohle je dost špatný přístup, protože člověk spíš potřebuje vidět nejnovější komentáře a jen pak starší. Druhá myšlenka spočívala v zobrazení seznamu od nejnovějšího komentáře a při projíždění dolů by se zobrazily starší. Tato varianta také nebyla optimální, protože toto chování by nebylo logické pro uživatele, jelikož se



(a) Obrazovka s příspěvky (b) Komentáře k příspěvku

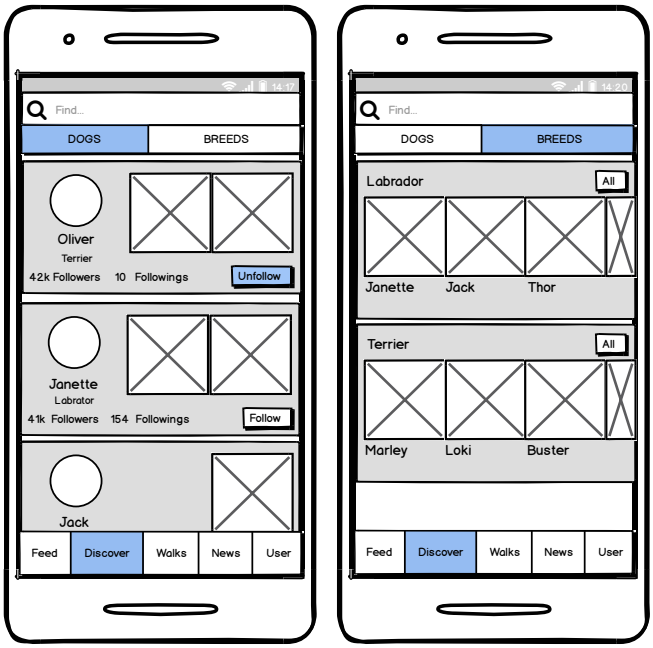
Obrázek 4.4: Lo-fi prototypy: seznam příspěvků a komentáře

vždy očekává, že při projíždění seznamu bude zachované chronologické pořadí komentářů.

Proto byla použita třetí varianta, kterou částečně používají jiné sociální sítě, kde se při načítání seznamu zobrazí posledních deset komentářů, které budou seřazené v přirozeném pořadí (od starších k novějším). Tlačítko načítání dalších komentářů se bude nacházet nahoře obrazovky a seznam se bude dočítat nahoru pomocí starších prvků, což nerozbije ani logiku ani strukturu seznamu. Náhled obrazovky s komentáři je možné si prohlédnout na obrázku 4.4b.

4.2.5.6 Objevení

V této sekci uživatel může najít nové stránky ke sledování. Objevení bude rozděleno na dvě části pomocí záložek (Tab Baru): první část bude věnována samotným stránkám psů, druhá bude zaměřena na plemena. Obě části budou obsahovat vyhledávací políčko pro hledání buď konkrétního psa podle jména nebo plemena. Psy v seznamu budou seřazené podle popularity (počtu sledujících). Dále je potřeba, aby každý prvek v seznamu, který reprezentuje psa, měl vlastní tlačítko pro sledování, jelikož to je cílem této sekce. Náhled návrhu sekcí objevení pro hledání profilových stránek psů je možné vidět na obrázku 4.5a.



(a) Objevení psů (b) Objevení plemen

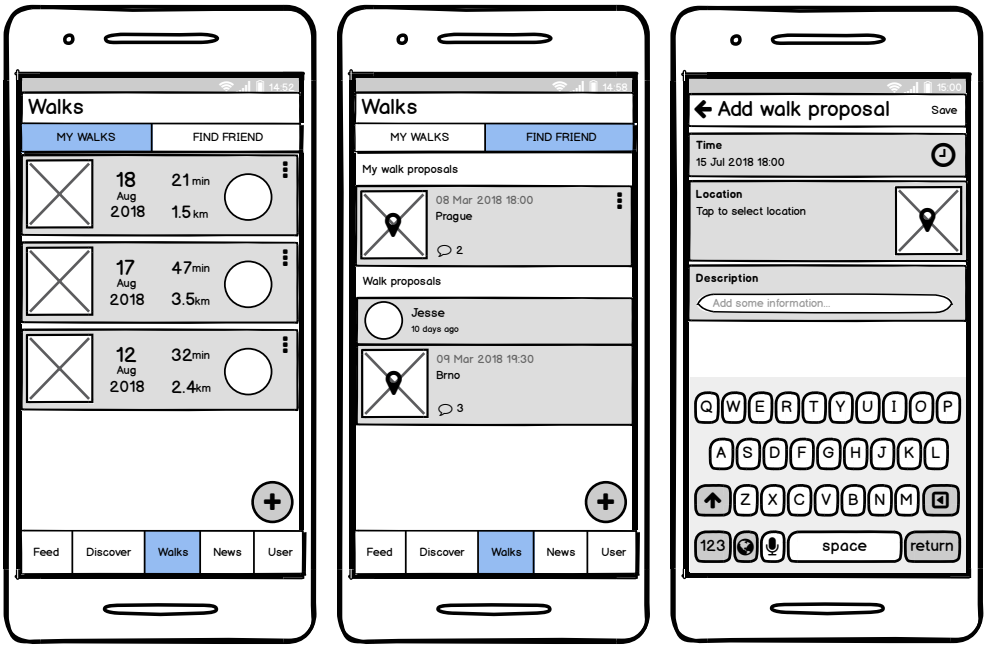
Obrázek 4.5: Lo-fi prototypy: vyhledávání ve psech a plemenech

Druhá záložka bude věnována plemenům, kde každý prvek v seznamu bude obsahovat několik fotografií nejpopulárnějších reprezentantů této skupiny. Možností by bylo také mít v aplikaci fotografie pro každé plemeno, což není flexibilní a navíc by to zabíralo hodně paměti. Zvolené řešení naopak přináší možnost přechodu na stránky nejpopulárnějších psů z daného plemena a obrázky se mohou měnit v průběhu času. Náhled této obrazovky je možné vidět na obrázku 4.5b.

4.2.5.7 Procházky

Sekce s procházkami bude obsahovat všechny uložené procházky pro daného uživatele, v nezávislosti na tom, jestli byly sdílené. Každá procházka musí obsahovat následující důležité informace: doba trvání, ušlá vzdálenost, čas, kdy byla procházka uskutečněna, informace o psovi a snímek mapy s trasou. Přidání nové procházky bylo navrženo jako tlačítko (FAB) v pravém dolním rohu. Lo-fi prototyp této obrazovky je možné vidět na obrázku 4.6a.

Po kliknutí na toto tlačítko se začne automaticky trekování trasy venčení psa, výjimku tvoří případ, kdy uživatel neposkytl aplikaci povolení pro sledování polohy na mapě. Tato obrazovka bude především zodpovědná za poskytování aktuálních informací o průběhu procházky. Po zastavení trekování trasy venčení psa se uživateli ukáže stránka s výsledným snímkem mapy s vykreslenou trasou a výslednými informacemi. V prvním návrhu tato obrazovka



(a) Procházky (b) Nabídky na procházku (c) Přidání nabídky

Obrázek 4.6: Lo-fi prototypy: procházky a nabídky na společné venčení

obsahovala dvě různá tlačítka: jedno pro uložení procházky a druhé pro sdílení procházky jako příspěvku na stránce psa.

4.2.5.8 Nabídky na společné venčení

Nabídky na společné venčení psů se podle návrhu nachází ve druhé záložce vedle procházek. Tato obrazovka obsahuje dva spojené seznamy, kde první část zahrnuje nabídky uživatele na společné venčení a druhá část zahrnuje cizí nabídky. Hlavní součástí každé nabídky je mapa s polohou pro setkání a přesný čas setkání. Také každá nabídka obsahuje vlastní komentáře, kde se uživatelé mohou domluvit o schůzce. Obrazovka s komentáři pro nabídku na společné venčení psů je stejná jako pro příspěvky. Náhled lo-fi prototypu obrazovky s nabídkami na společné venčení psů je možné vidět na obrázku 4.6b.

Také bylo důležité navrhnout obrazovku pro přidání nabídky na procházku, jelikož zadání všech potřebných údajů musí být pohodlné pro uživatele. Proto bylo rozhodnuto, že uživatel nebude uvádět název města nebo nějak pojmenovávat místo pro setkání, jen jednoduše označí potřebné místo na mapě. Výběr data se také bude provádět pomocí jednoho zjednodušeného prvku, kde je možné zvolit všechny potřebné údaje od data do přesného času. Lo-fi prototyp této obrazovky je možné vidět na obrázku 4.6c.

4.2.6 Druhý návrh uživatelského rozhraní

Hlavním nedostatkem, který se objevil v průběhu další analýzy a rozvoje návrhu uživatelského rozhraní, byla absence možnosti změnit psa na jiném místě než na hlavní profilové stránce. Tak vznikaly problémy, že pokud uživatel má více než jednu stránku psa a chce například zanechat komentář, tak může přidat komentář jen od aktuálně vybraného psa. Pokud by chtěl přepnout stránku, tak by musel odejít ze seznamu komentářů, vrátit se na profilovou stránku a změnit psa. Pak návrat do původní obrazovky by byl komplikovaný. Proto lepším řešením je umožnit uživateli měnit stránku psa na skoro všech obrazovkách v aplikaci. Toto bylo vyřešeno pomocí ikony změny aktuálního psa, která se nachází ve všech horních navigačních lištách (Action Bar) v aplikaci.

4.2.6.1 Profilová stránka psa

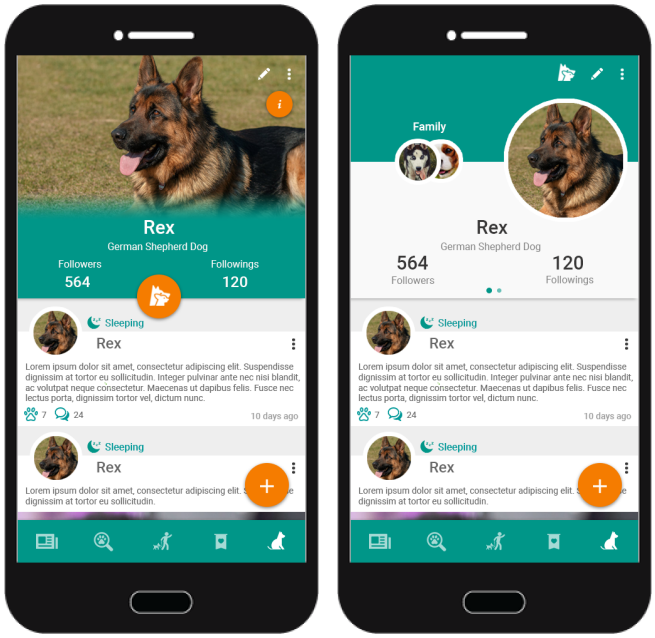
Profilová stránka psa byla podvržena největším změnám v průběhu vývoje této aplikace. Hlavním problémem této obrazovky byla hlavička a správné rozmístění údajů o psovi. První hi-fi prototyp je možné vidět na obrázku 4.7a. Velkou nevýhodou tohoto návrhu je oranžové tlačítko přepínání mezi stránkami psů uživatele. Problém tohoto návrhu spočíval v tom, že se na stránce cizího psa v horní liště nacházela ikona přepínání psů daného uživatele (která se tam objevila po rozhodnutí popsaném v předchozí sekci) a v dolní části hlavičky se nacházelo podobné tlačítko, které již plnilo jinou funkci, a to přepínání mezi členy rodiny psa, na stránce kterého byl uživatel. Toto bylo velmi matoucí pro uživatele a porušovalo princip konzistence a logiky. Proto vznikl další návrh profilové stránky, který je možné vidět na obrázku 4.7b. Tento návrh zachovává konzistenci a zlepšuje celkový grafický dojem oproti předchozímu. V novém návrhu je rodina uživatele znázorněna ve formě malých profilových obrázků, kde se zobrazují maximálně ještě dva členové psí rodiny uživatele. Pokud má uživatel více než tři psy, tak by se vedle popisku rodiny mělo objevit celkové číslo psů, které má uživatel. Po kliknutí na tento prvek se objeví dialog se seznamem psů z rodiny.

4.2.6.2 Hi-fi prototypy ostatních obrazovek

Ostatní obrazovky podlehly menším změnám oproti lo-fi prototypům než profilová stránka. Hi-fi prototypy některých obrazovek, které jsou popsané níže, je možné vidět na obrázcích 4.8.

- Přidání nového příspěvku 4.8a

  Hlavní nevýhoda předchozího návrhu této obrazovky spočívala v tom, že aplikace neupozorňovala na to, kterého psa má uživatel aktuálně vybraného. Toto bylo opravené přidáním profilového obrázku vedle políčka pro přidání textového popisu. Také tlačítka pro přidání fotografií pomocí



(a) První verze (b) Druhá verze

Obrázek 4.7: Vývoj hi-fi prototypu uživatelské stránky

kamery nebo galerie byly přesunuté pod horizontální seznam nedávných snímků.

- Příspěvek 4.8b

  V tomto návrhu byla kategorie přesunuta nahoru pro lepší přehlednost a jednoduché odstranění v případě, pokud uživatel nechce uvést žádnou kategorii. Také datum přidání příspěvku bylo přesunuto do pravého dolního rohu, protože má menší význam než ostatní údaje.

- Stránka s komentáři 4.8c

  Vylepšení této obrazovky se především dotklo grafické strany návrhu. Koncept karet byl změněn na bubliny, aby se to četlo jako jedna konverzace, nikoliv oddělené textové zprávy.

- Objevení psů 4.8d

  Jedna z obrazovek, která se od počátečního návrhu skoro nezměnila.

- Procházky 4.8e

  Změnám podlehla jen grafická stránka této obrazovky.

- Nabídky na společné venčení 4.8f

  Předchozí návrh této obrazovky obsahoval jednu velkou nevýhodu, a to absenci filtrování cizích nabídek podle místa provedení. Tento problém byl vyřešen tak, že byla přidána možnost filtrování cizích nabídek podle města pomocí ikony v horní části obrazovky. Pro pohodlnost uživatele se také přidala funkce automatického doplnění města, ve kterém se nachází, v případě, že má aplikace povolení ke sledování polohy uživatele.

4.2.6.3 Heuristická analýza uživatelského rozhraní

Jedním ze způsobů, jak ohodnotit a otestovat uživatelské rozhraní, je heuristická analýza podle Jakoba Nielsena. Každá aplikace by měla splňovat deset pravidel Nielsenovy heuristické analýzy pro zajištění intuitivního chování a odpovídající použitelnosti. Dále jsou uvedena jednotlivá pravidla [29]:

1. Viditelnost stavu systému

   Systém musí vždy informovat uživatele v rozumném čase o tom, v jakém stavu se nachází a co se právě provádí.

2. Konzistence mezi systémem a realitou

   Systém by měl komunikovat s uživatelem pomocí konceptů a pojmů, kterým uživatel rozumí.

3. Minimální zodpovědnost

   Systém by měl vždy poskytnout uživateli způsob vyřešení problémů jako například opuštění nechtěného stavu, ke kterému se uživatel dostal omylem.

4. Shoda s platformou a standardy

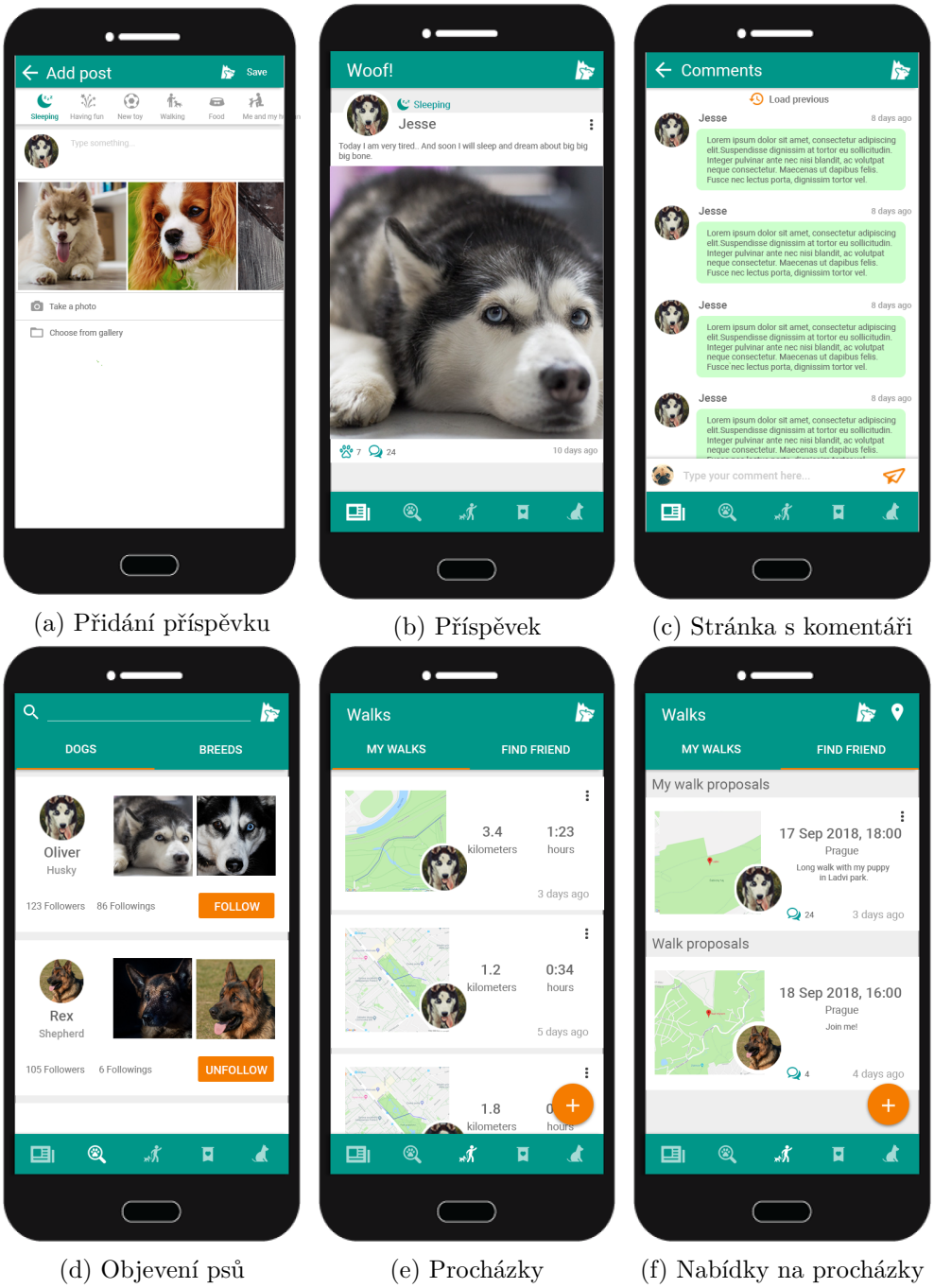
   Systém by se měl dodržovat standardy definované platformou. Systém by měl také dodržovat zvyky definované druhem aplikace, který představuje.

5. Prevence chyb

   Systém musí umět předejít chybám uživatele.

6. Rozpoznání namísto vzpomínání

   Systém by měl poskytnout snadný způsob orientace v systému. Základní koncept tohoto pravidla spočívá v tom, že uživatel nemusí pamatovat příliš mnoho detailů, všechno potřebné musí systém poskytnout na obrazovce.



(a) Přidání příspěvku (b) Příspěvek (c) Stránka s komentáři

(d) Objevení psů (e) Procházky (f) Nabídky na procházky

Obrázek 4.8: Hi-fi prototypy

7. Flexibilita a efektivita

   Systém by měl poskytovat funkce jak pro zkušené, tak i nezkušené uživatele.

8. Minimalita

   Systém by neměl obsahovat irelevantní nebo nepotřebné informace, což velmi snižuje viditelnost důležité informace.

9. Smysluplné chybové hlášky

   Systém by měl obsahovat jen chybové hlášky se srozumitelným popisem problému a případně s postupem jeho řešení.

10. Nápověda a dokumentace

    Systém by měl být použitelný bez příručky, pokud neobsahuje složité koncepty a myšlenky pro uživatele. Nápovědy v složitějším systému jsou jeho nezbytnou součástí.

V průběhu rozvoje návrhu uživatelského rozhraní, byly výsledné koncepty a prototypy vždy ověřené pomocí výše popsaných heuristik. Tento přístup pomohl opravit velké množství nesrovnalostí a menších problémů, jako například absence načítacích stavů na začátku vývoje. Poslední návrh uživatelského rozhraní splňuje všech deset pravidel výše popsané heuristické analýzy, a proto je považován za jeden z nejlepších, který byl navržen v průběhu implementace dané aplikace.

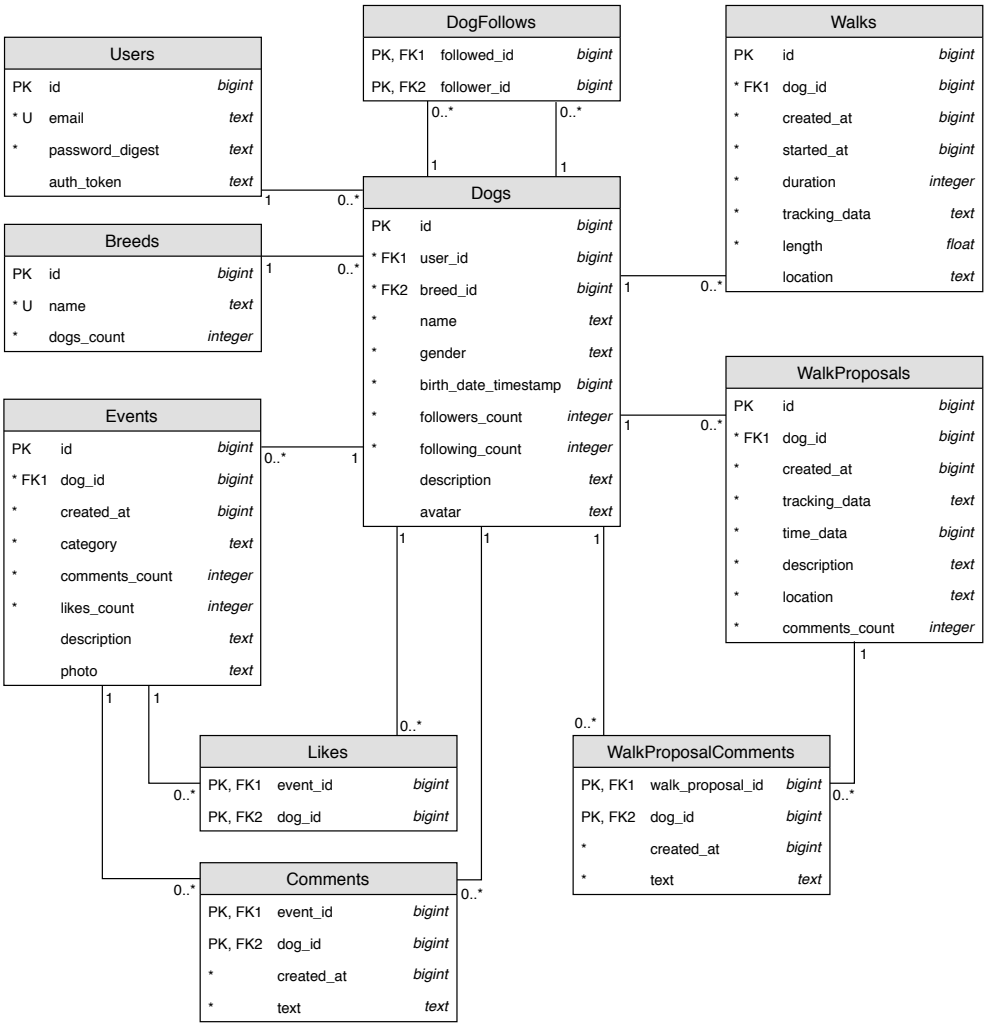
4.3 Databázový model

Serverová část aplikace obsahuje databázi, ve které jsou uložena data uživatelů, návrh této databáze vychází z předchozí analýzy a doménového modelu. Na obrázku 4.9 je možné vidět navržený databázový model. Mobilní aplikace neobsahuje databázi, protože všechna data načítá ze serveru a pro použití libovolné funkce je vyžadováno internetové připojení.

Evidence sledovaných a sledujících psů je navržena pomocí tabulky `DogFollows`. V této tabulce jsou uložené dvojice identifikátorů psů, kteří definují vztah „sleduje“. Například pokud pes s identifikátorem X začne sledovat stránku psa s identifikátorem Y, tak se do tabulky přidá řádek (X, Y). Pokud se pes Y rozhodne sledovat psa X, tak se přidá řádek s hodnotou (Y, X).

Profilové obrázky a obrázky pro příspěvky jsou uložené mimo databázi a do databáze se ukládá jen relativní cesta do těchto souborů.

Z bezpečnostních důvodů se do databáze ukládá jen otisk hesla, nikoliv heslo samotné. Také do tabulky `Users` se ukládá autentizační token uživatele. Více o autentizaci je si možné přečíst v sekci 4.4.3.1.



Obrázek 4.9: Databázový model

4.4 Návrh serverové části

4.4.1 RESTful API

V celé této sekci čerpám z [24].

REST (Representational State Transfer) je architektonický styl, který definuje sadu omezení a dohod. Rozhraní, které splňuje zásady RESTu, se nazývá RESTful. REST je navržen pro optimální využití protokolu HTTP, což je jeho nejdůležitější výhodou. Je potřeba si uvědomit, že REST není ani standardem, ani protokolem, je to pouze architektonický styl, který využívá protokol HTTP.

4.4.1.1 Zdroj

REST používá koncepty, které jsou podobné zásadám protokolu HTTP. Jedním z nejdůležitějších konceptů je zdroj (anglicky resource). Zdrojem se nazývá abstraktní datový model, který může mít hierarchickou strukturu a může obsahovat vnořené zdroje. Také zdroj může být identifikován jedinečným identifikátorem URI (Uniform Resource Identifier). Rozhraní nikdy neposkytuje zdroj napřímo, ale jeden z jeho obrazů, který má název reprezentace. Zdroje jsou podobné objektům v objektově orientovaném programování (OOP), rozdíl je v tom, že metody v RESTu jsou omezeny na množinu metod HTTP (GET, POST, PUT, DELETE), zatímco v OOP mohou být libovolné. Také nelze použít žádné jiné metody kromě HTTP metod k manipulaci se zdroji.

4.4.1.2 Reprezentace zdroje

Zdroje jsou poskytovány formou reprezentace, což je typem serializace zdroje. Pro reprezentaci lze použít různá schémata kódování nebo typy médií (mediatype). Pravidla pro vytváření reprezentace mohou být odlišná, ale všechna zobrazení stejného zdroje obsahují stejné informace. Nejznámějšími reprezentacemi zdrojů jsou například JSON nebo XML.

4.4.1.3 Uniformní rozhraní zdroje

RESTful rozhraní obvykle provádí CRUD operace (Create, Read, Update, Delete), které mohou být jednoduše implementovány pomocí metod HTTP. Vytváření zdroje se provádí pomocí POST metody, číst obsah lze pomocí GET, aktualizovat pomocí PUT a za odstranění zdroje je zodpovědna metoda DELETE.

Metody HTTP mají různé vlastnosti, mohou být bezpečně anebo idempotentní. Bezpečné metody nemění stav zdroje a nemají žádné vedlejší efekty. GET, HEAD a OPTIONS jsou bezpečné HTTP metody. Idempotence znamená, že vícenásobné volání stejné metody vrací vždy stejnou odpověď. PUT, DELETE a všechny bezpečné metody jsou idempotentní metody.

4.4.1.4 Stav v REST

V distribuovaných systémech je potřeba zachovávat informace mezi jednotlivými voláními metod, tato informace se nazývá také „stav“. Vzhledem k tomu, že distribuované systémy se skládají z klientské a serverové části, může být stav zachován v každé z těchto komponent. Podle architektury REST serverová komponenta musí být bezestavovou, a proto je stav uložen na klientské straně. Každé volání API musí být nezávislé na předchozím volání, proto každý požadavek musí obsahovat všechny nutné informace k jeho vykonání. Princip bezestavosti zajišťuje jednoduchou škálovatelnost infrastruktury.

4.4.1.5 Stavové kódy HTTP

Použití stavových kódů HTTP je důležitým principem RESTu. Součástí odpovědi každého API volání musí být stavový kód. Existují různé stavové kódy, které umožňují klientu buď určit typ chyby nebo se dozvědět další kroky po volání API metody.

Kategorie stavových kódů:

- **Informace (kód 1xx).**
- **Úspěch (kód 2xx).** Žádost byla úspěšně zpracována.
- **Přesměrování (kód 3xx).** Klient musí provést jiné volání na základě parametrů přijatých v odpovědi.
- **Chyba klienta (kód 4xx).** Došlo k chybě kvůli datům, která poslal klient.
- **Chyba serveru (kód 5xx).** Došlo k chybě na straně serveru.

4.4.1.6 Závěr

Architektonický styl REST uvádí následující pravidla a omezení pro návrh RESTful rozhraní, které je potřeba dodržovat při návrhu API:

- Využití možností HTTP.
- Použití uniformních rozhraní zdrojů.
- Bezestavová komunikace mezi klientem a serverem.
- Zachování nezávislosti jednotlivých volání.
- Použití stavových kódů HTTP.
- Použití typů médií.
- Návrh názvů zdrojů pomocí podstatných jmen (například users, dogs), nikoliv sloves.

Jak už bylo zmíněno výše, jednou z největších výhod RESTu je škálovatelnost systému. Další výhodou je výkonnost díky použití HTTP cáchování. Jelikož zdroj může existovat v několika reprezentacích, tak API dokáže poskytovat informace v několika formátech, což znamená, že na klientu je možné vybrat vhodný formát.

4.4.2 Stránkování

Stránkování je rozdělení dat do bloků (stránek) nějaké určité velikosti. Nejjednodušší příklad stránkování je vyhledávání v Googlu, kde obvykle počet relevantních výsledků může přesahovat desítky tisíc a zobrazení všech odpovědí najednou není optimální ani pro zátěž serveru, ani pro síťovou komunikaci. Proto server musí umět poskytovat obsah po kouscích pomocí vhodné metody.

Aplikace, která je předmětem této diplomové práce, bude pracovat s velkou množinou dat, kterou bude potřeba zobrazovat uživateli postupně. Příkladem může být seznam příspěvků, kde uživateli budou zobrazené jen nejnovější příspěvky a další část se bude postupně načítat, až pokud se uživatel dostane na konec již načteného seznamu. Existují dva známé přístupy implementace stránkování: pomocí ofsetu (offset pagination) a s použitím klíčů (key based pagination). Znalosti o metodách stránkování čerpám z [25].

4.4.2.1 Stránkování pomocí ofsetu

Stránkování pomocí ofsetu je velmi jednoduchá metoda, která používá `OFFSET` a `LIMIT` SQL příkazy. Tento způsob je velmi neefektivní při použití velkých hodnot ofsetu. Například pokud je potřeba načíst data od 40 000. řádku, tak databáze musí přeskočit prvních 39 999 řádků, což je plýtvání výkonem procesoru. Pro implementaci tohoto způsobu je potřeba, aby API umožňovalo nastavení parametrů `offset` a `limit`, například pomocí URL parametrů.

Ukázka SQL příkazu při použití stránkování pomocí ofsetu

```
SELECT column FROM table LIMIT 10 OFFSET 40000
```

4.4.2.2 Stránkování pomocí klíčů

Stránkování pomocí klíčů je mnohem efektivnější, protože není závislé na velikosti tabulky. Tato metoda je založena na tom, že další záznamy jsou vráceny na základě hodnoty posledního záznamu předchozího kusu dat (stránky). Stránkování pomocí klíčů vyžaduje jedinečnost sloupce pro řazení, pokud sloupec není unikátní, lze ho zřetězit s primárním klíčem k vytvoření jedinečného pořadí. Velkou výhodou tohoto přístupu je umožnění použití databázového indexu, který obsahuje potřebné sloupce. Tím pádem záznamy v předchozích stránkách budou opravdu přeskočeny, což je mnohem efektivnější.

Například seznam příspěvků je potřeba seřadit podle data vytvoření, ale tento sloupec není sám o sobě unikátní, protože několik uživatelů mohou vytvořit událost ve stejný okamžik. Proto klíčem pro řazení bude pár `<timestamp, id>`, kde první prvek bude seřazen sestupně a druhý vzestupně. Potom klient bude mít možnost zavolat endpoint s dvěma nepovinnými parametry: `from` a `limit`. `Limit` funguje stejně jako u předchozí metody, tudíž omezí počet prvků,

`from` je klíč posledního prvku. Pokud `from` není uveden, tak server vrátí data ze začátku seznamu. Aby klient nemusel vypočítávat klíč, server bude vracet klíč posledního prvku v hlavičce odpovědi. Tím pádem ze strany klienta bude potřeba jen předat tento klíč jako parametr do dalšího požadavku.

Tato metoda je mnohem efektivnější než předchozí, proto se bude používat pro stránkování dat v navrženém API.

4.4.3 Návrh API

Rozhraní, které bude poskytovat serverová část, bylo navržené podle pravidel RESTu a doporučení, popsaných v sekci 4.4.1. Výsledné API je rozděleno do několika skupin, kde každá skupina obsahuje metody, které úzce souvisí mezi sebou. Většina rozhraní se skládá ze zdrojů, které jsou odvozené z doménového modelu. Například existují metody pro založení stránky psa, čtení informací o psovi, změnu údajů a také odstranění stránky psa. Ale také existuje několik zajímavějších zdrojů, které budou podrobně popsané dále. Jedním z takových zdrojů je `feed`, který slouží pro akumulaci všech příspěvků, které byly přidané na odebíraných stránkách uživatelem. Dalším zajímavým zdrojem je `recent`, což je seznam informací, které se týkají psů z rodiny uživatele (například nový sledující nebo nový komentář).

4.4.3.1 Skupina USERS

Tato skupina obsahuje metody pro registraci a přihlášení pomocí elektronické adresy a hesla. Validace správnosti formátu elektronické adresy a složitosti hesla probíhá na klientu. Pro jednoduchost se neprovádí dvojité ověření pomocí aktivačního emailu.

Metody ze všech ostatních skupin vyžadují ověření identity uživatele pro každé volání API. Proto registrační proces musí zahrnovat generování autentizačního tokenu, který je unikátní pro uživatele, a proto jednoznačně určuje jeho identitu. Token se odesílá klientu a ten ho používá pro další komunikaci se serverem. Výhodou použití tokenu je to, že klientská část nemusí ukládat přihlašovací údaje, což může předejít jejich odcizení v případě útoku na zařízení uživatele. Pokud požadavek nebude obsahovat token nebo bude obsahovat neplatný token, server nevyhoví požadavku a vrátí chybu. V tomto případě volající musí znovu požádat o vygenerování tokenu.

- `POST /api/v1/users/register`

  Metoda pro založení účtu v aplikaci, která na vstupu dostává elektronickou adresu a heslo. Při pokusu o registraci s již registrovaným emailem vrátí chybu. Kvůli zabezpečení přihlašovacích údajů uživatele se heslo neukládá v otevřené podobě. Proto v průběhu registrace se pomocí hashovací funkce vygeneruje otisk hesla a ten už bude uložen do databáze.

- `POST /api/v1/users/login`

  Slouží pro získání tokenu pro již existujícího uživatele, jako i předchozí metoda na vstupu dostává elektronickou adresu a heslo. Z hesla se vypočítá otisk a ten se bude porovnávat s již uloženým otiskem pro daného uživatele. Pokud otisky jsou stejné, metoda vrátí autentizační token daného uživatele, v opačném případě vrátí chybu.

4.4.3.2 Skupina DOGS

Tato skupina obsahuje metody pro správu psů uživatelů a pro evidenci vazeb sledování. Serverová část neukládá aktuálně vybraného psa uživatele, za toto je zodpovědná mobilní aplikace, a proto všechny metody, které souvisejí s přidáním příspěvků, komentářů a podobné, vyžadují identifikátor vybraného psa.

- `GET /api/v1/dogs`

  Vrací seznam všech psů z rodiny uživatele. Identifikátor uživatele se předává pomocí parametrů URL.

- `POST /api/v1/dogs`

  Slouží pro založení stránky psa. Povinnými parametry jsou jméno, pohlaví a identifikátor plemena.

- `GET /api/v1/dogs/{dog_id}`

  Vrací informace o psovi, jehož identifikátor je součástí URL.

- `PUT /api/v1/dogs/{dog_id}`

  Aktualizuje informace o psovi, jehož identifikátor je součástí URL.

- `DELETE /api/v1/dogs/{dog_id}`

  Odstraní stránku psa, jehož identifikátor je součástí URL.

- `GET /api/v1/dogs/{dog_id}/followers`

  Vrací seznam všech psů, které odebírají stránku psa, jehož identifikátor je součástí URL.

- `GET /api/v1/dogs/{dog_id}/following`

  Vrací seznam všech psů, odebíraných daným psem, jehož identifikátor je součástí URL.

- `POST /api/v1/dogs/{dog_id}/following/{follow_dog_id}`

  Slouží pro přidání vazby sledování mezi dvěma psy. Identifikátor sledovaného psa se předává pomocí parametru `follow_dog_id` a identifikátor sledujícího psa se předává v parametru `dog_id`.

- `DELETE /api/v1/dogs/{dog_id}/following/{follow_dog_id}`

  Ruší vazbu sledování, funguje podobně jako přidání vazby.

4.4.3.3 Skupina PHOTOS

Skládá se z metod, které poskytují obrázky používané v profilech a v příspěvcích. Obrázky nejsou dostupné pro veřejnost, proto tato skupina také vyžaduje autentizační token. Obsahuje také metody, které vrací obrázky v menším rozlišení, například pro zobrazení v seznamech.

- `GET /api/v1/photos/avatars/{dog_id}`
- `GET /api/v1/photos/avatars/{dog_id}/thumbnail`
- `GET /api/v1/photos/events/{event_id}`
- `GET /api/v1/photos/events/{event_id}/thumbnail`

4.4.3.4 Skupina FEED

Obsahuje jedinou metodu, která poskytuje seznam příspěvků ze stránek všech psů sledovaných všemi psy z rodiny uživatele. Příspěvky jsou seřazené sestupně podle časové značky vytvoření, od nejnovějších po nejstarší. Tato metoda podporuje stránkování podle principů popsaných v předchozí kapitole. Jelikož stejného psa může sledovat několik psů uživatele, tak se nesmí stát, že stejné příspěvky se budou vyskytovat ve výsledku vícekrát.

- `GET /api/v1/feed`

4.4.3.5 Skupina EVENTS

Tato skupina obsahuje metody pro správu příspěvků jednotlivých psů.

- `GET /api/v1/events`

  Vrací seznam všech příspěvků psa. Příspěvky jsou seřazené od nejnovějších, také metoda podporuje stránkování.

- `POST /api/v1/events`

  Slouží pro přidání příspěvku, kde povinnými parametry jsou identifikátor psa, kategorie a popis (pokud chybí fotografie) nebo fotografie (pokud chybí popis).

- `GET /api/v1/events/{event_id}`

  Vrací informaci o konkrétním příspěvku.

- `DELETE /api/v1/events/{event_id}`

  Slouží pro odstranění příspěvku ze serveru.

- `POST /api/v1/events/{event_id}/like`

  Přidává příspěvek do oblíbených.

- `DELETE /api/v1/events/{event_id}/like`

  Odstraňuje příspěvek z oblíbených.

4.4.3.6 Skupina COMMENTS

Tato skupina obsahuje metody pro přístup ke komentářům příspěvků.

- `GET /api/v1/comments`

  Slouží pro načítání komentářů pro příspěvek. Komentáře jsou seřazené od nejnovějších, metoda také podporuje stránkování.

- `POST /api/v1/comments`

  Slouží pro přidání komentáře, pro jeho vytvoření je potřeba uvést text, identifikátor psa a příspěvku.

- `GET /api/v1/comments/{comment_id}`

  Získává informaci o daném komentáři.

- `DELETE /api/v1/comments/{comment_id}`

  Slouží pro odstranění komentáře.

4.4.3.7 Skupina BREEDS

Skupina metod, které se týkají plemen psů.

- `GET /api/v1/breeds`

  Vrací seznam všech plemen, které jsou podporované na serveru.

- `GET /api/v1/breeds/{breed_id}/dogs`

  Vrací seznam všech psů, kteří patří do daného plemena, také metoda podporuje stránkování.

4.4.3.8 Skupina DISCOVER

Metody z této skupiny slouží pro hledání zajímavých stránek pro sledování napříč celou sociální sítí. Obě podporují stránkování a vyhledávání.

- `GET /api/v1/discover/dogs`

  Poskytuje stránky psů, kteří jsou seřazené sestupně podle popularity, což je počet psů, sledujících tuto stránku. Vyhledávání se provádí jen ve jménech psů.

- `GET /api/v1/discover/breeds`

  Poskytuje plemena seřazená sestupně podle počtu psů v těchto plemenech. Vyhledává se jen v názvech plemen. Spolu s plemenem se vrací několik nejpopulárnějších psů patřících do daného plemena.

4.4.3.9 Skupina RECENT

Obsahuje jedinou metodu poskytující přehled posledních událostí, které se týkají psů z rodiny uživatele. Události se dělí do několika typů: nový sledující, nové přidání vlastní události do oblíbených, nový komentář na vlastní příspěvek nebo na nabídku na venčení psů. Metoda má jako jediný parametr časovou značku. Pomocí této časové značky server může připravit všechny nové události, které spadají do časového intervalu mezi časovou značkou a aktuálním časem. Počet vrácených událostí každého typu je omezen, aby nezahlcoval uživatele zbytečnými informacemi.

- `GET /api/v1/recent`

4.4.3.10 Skupina WALKS

Tato skupina obsahuje metody pro evidenci procházek se psem.

- `GET /api/v1/walks`

  Vrací seznam procházek uživatele, identifikovaného autentizačním tokenem. Jelikož procházky jsou privátní, tak metoda nevyžaduje identifikátor uživatele. Také metoda podporuje stránkování.

- `POST /api/v1/walks`

  Slouží pro přidání procházky, která musí obsahovat čas začátku, dobu trvání, identifikátor psa a ušlou cestu jako seznam souřadnic.

- `GET /api/v1/walks/{walk_id}`

  Vrací informace o konkrétní procházce.

- `DELETE /api/v1/walks/{walk_id}`

  Slouží pro odstranění dané procházky.

4.4.3.11 Skupina MY WALK PROPOSALS

Tato skupina obsahuje metody pro správu vlastních nabídek uživatele na společné venčení.

- `GET /api/v1/{my_walk_proposals}`

  Vrací všechny nabídky uživatele seřazené podle data vytvoření.

- `POST /api/v1/{my_walk_proposals}`

  Slouží pro přidání nabídky na společné venčení. Nabídka musí obsahovat identifikátor psa, čas setkání a místo jako mapové souřadnice.

4.4.3.12 Skupina WALK PROPOSALS

Metody pro evidence nabídek na společné venčení ostatních uživatelů.

- `GET /api/v1/walk_proposals`

  Vrací seznam nabídek na společné venčení psů jiných uživatelů. Umožňuje filtrování podle města, podporuje stránkování.

- `GET /api/v1/walk_proposals/walk_proposal_id`

  Vrací informace o konkrétní nabídce.

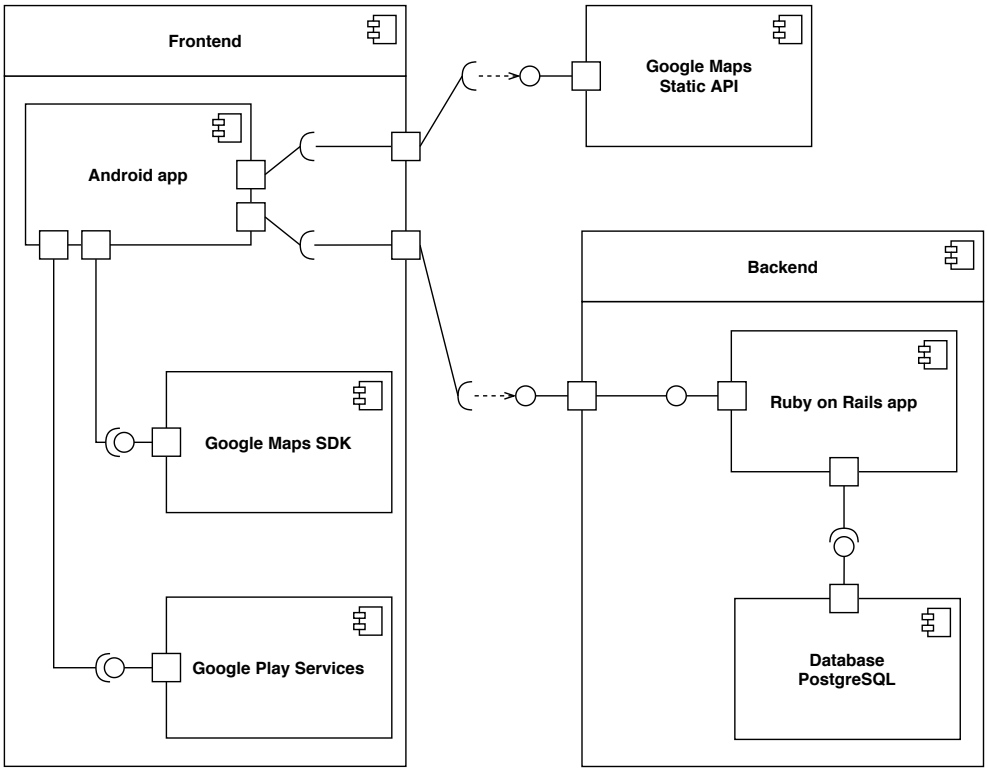
- `DELETE /api/v1/walk_proposals/walk_proposal_id`

  Odstraní nabídku na společné venčení uživatele podle identifikátoru. Jen uživatel, který vlastní nabídku, ji může smazat.

4.4.3.13 Skupina WALK PROPOSAL COMMENTS

Obsahuje metody pro přístup k komentářům nabídky na společné venčení. Tyto metody jsou podobné metodám, které spravují komentáře příspěvků, a proto budou jen uvedené níže.

- `GET /api/v1/walk_proposal_comments`
- `POST /api/v1/walk_proposal_comments`
- `GET /api/v1/walk_proposal_comments/{comment_id}`
- `DELETE /api/v1/walk_proposal_comments/{comment_id}`



Obrázek 4.10: Diagram komponent pro výsledný systém

4.5 Architektura systému

Výsledná architektura systému je znázorněna pomocí diagramu komponent na obrázku 4.10. Cílem tohoto diagramu je ukázat vztah mezi různými součástmi systému. Výsledný systém se skládá z frontendové části (mobilní aplikace pro operační systém Android), backendové části (Ruby on Rails aplikace poskytující RESTful API) a služby třetích stran (Google Maps Static API). Mobilní aplikace používá Google Maps SDK, které umožňuje zobrazení Google map v aplikaci. SDK automaticky spravuje komunikaci s Google Map servery, stahování dat a také zobrazení a interakci s mapou. Aplikace také využívá služby Google Play pro přístup k FusedLocationProviderClientu pro práci s geografickou polohou uživatele. Google Maps Static API se používá pro generování statických snímků mapy pro zobrazení cest z procházek v mobilní aplikaci. Ruby on Rails aplikace slouží pro poskytování obsahu a ukládání dat do PostgreSQL databáze.

Kapitola 5

Realizace

5.1 Serverová část

Serverová část navrženého systému byla implementována jako Ruby on Rails aplikace, která poskytuje RESTful JSON API. Frameworky Ruby on Rails a Grape byly zvolené kvůli rychlosti vývoje, velkému počtu dostupných knihoven a návodů, skvělé podpoře testování a vlastním zkušenostem. Implementovaná aplikace je nasazena na virtuálním privátním serveru (VPS). V průběhu vývoje také byla vyzkoušena možnost cloudového řešení s použitím Heroku [5]. Brzy se objevilo, že výkon bezplatného plánu není dostačující, a proto byl pořízen VPS.

5.1.1 Ruby on Rails

V této sekci čerpám z [30]. Ruby on Rails je framework pro vývoj webových aplikací napsaných v jazyce Ruby. Ruby on Rails aplikace implementují architekturu Model - View - Controller. Daná serverová aplikace nebude mít žádné uživatelské rozhraní (View) a bude jen poskytovat data pomocí API.

Kromě architektury Ruby on Rails také definuje několik principů, které byly dodržované při implementaci:

- Neopakujte se (anglicky „Don't repeat yourself“)

  V aplikaci se nesmí objevovat duplicitní kód a komponenty by měly být znovu použitelné.

- Konvence má přednost před konfigurací (anglicky „Convention over configuration“)

  Aplikace by se měla vyvíjet podle konvencí stanovených Ruby on Rails, nikoliv podle konfigurací definovaných programátorem. Umožňuje rychlé a jednoduché vytváření aplikací při zachování konvencí.

[5] https://www.heroku.com/ruby

5.1.2 Active Record

Tato sekce byla napsaná pomocí [31]. Active Record je architektonickým návrhovým vzorem pro práci s daty. Je to objekt, který obaluje řádek v databázové tabulce nebo pohledu, zapouzdřuje přístup k databázi a přidává doménovou logiku. V této sekci bude popsaná implementace tohoto vzoru v serverové části aplikace „Woof!“. Active Record je jedním ze základních prvků každé Ruby on Rails aplikace a odpovídá Modelu ve vzoru MVC.

Active Record umožňuje:

- Reprezentace modelů a jejich dat, reprezentace vztahů a dědičnosti modelů.
- Validace modelů před uložením do databáze.
- Provedení databázových operací objektově orientovaným způsobem.

Active Record široce využívá princip „Convention over configuration“, zejména definuje konvence pojmenování modelů a vytváření databázových schémat. Databázová tabulka musí být pojmenována s použitím množného čísla a podtržítek mezi slovy. Odpovídající modelová třída musí mít jméno v jednotném čísle ve velbloudí notaci s velkým prvním písmenem. Například instance modelové třídy Dog budou uložené do tabulky `dogs` a pro WalkProposal bude existovat databázová tabulka `walk_proposals`. Active Record umí pracovat i se složitějšími tvary množných čísel, jako je například „person - people“.

Také Active Record má konvence pro pojmenování primárních a cizích klíčů v databázových tabulkách. Velkou výhodou je také to, že Active Record umí generovat a automaticky spravovat některé užitečné databázové sloupce. Tak tabulka může obsahovat sloupec `created_at`, který obsahuje čas vytvoření entity a `updated_at`, který se aktualizuje po každé úpravě entity. Dalším užitečným sloupcem je `(table_name)_count`, který se používá pro cachování počtu entit, které patří instanci. Například tabulka `events` obsahuje vazby 1:N s tabulkami `likes` a `comments`, a proto má sloupce `likes_count` a `comments_count`, které jsou aktualizované po každém přidání (nebo odebrání) řádků do tabulky `likes` a `comments`.

Active Record umožňuje základní operace CRUD (anglicky Create, Read, Update, Delete), přístup k těm metodám je umožněn přes modelovou třídu.

Například na ukázce níže je možné vidět vytváření instance komentáře. Komentář vždy musí mít psa, kterému patří a identifikátor příspěvků, pod kterým se nachází. Při vytváření je potřeba uvést tyto instance, další možností je specifikovat jen jejich identifikátory.

Vytváření komentáře pomocí Active Record

```
attributes = {}
attributes[:dog] = dog
```

```
attributes[:event] = event
attributes[:text] = "Lorem ipsum"

comment = Comment.create! attributes
```

Načítání entit je možné provádět buď podle identifikátoru nebo podle různých podmínek (variace WHERE klauzule z SQL). Použití různých načítacích metod je si možné prohlédnout na ukázce níže. Na první ukázce je uveden příklad vyhledávání podle identifikátoru (`params` jsou parametry API dotazu). Druhá ukázka znázorňuje vyhledávání vazby sledování pomocí entit psů. Třetí reprezentuje složitější dotaz, který vyhledává aktuální cizí nabídky na společné venčení psů pro dané místo.

Různé metody načítání entit pomocí Active Record

```
# První ukázka
dog = Dog.find_by! id: params[:dog_id]

# Druhá ukázka
dog_follow = DogFollow.find_by! followed: @follow_dog, follower: @dog

# Třetí ukázka
user_dog_ids = @current_user.dogs.map(&:id)
items = WalkProposal.where('location ILIKE ?',
    "%#{params[:location]}%") unless params[:location].nil?
items = items.where("time_data > #{now}")
items = items.where.not(dog_id: user_dog_ids)
```

Aktualizace entit probíhá pomocí metody `update`, na vstupu dostává mapu atributů, které je potřeba aktualizovat. Existuje i další způsob pomocí metody `save`. Na následující ukázce jsou představeny příklady použití těchto metod.

Úprava detailu psa pomocí Active Record

```
# Použití update
attributes = {}
attributes[:name] = params[:name] if params.key?(:name)
attributes[:description] = params[:description] if params.key?
    :description
dog = Dog.find_by! id: params[:dog_id]
dog.update! attributes

# Použití save
dog = Dog.find_by! id: params[:dog_id]
dog.name = params[:name] if params.key? :name
dog.description = params[:description] if params.key? :description
dog.save!
```

5.1.3 Helpers

Ruby on Rails poskytuje velké množství pomocníků (helpers) pro práci s daty a čísly. Dobrým pravidlem je extrahovat stejnou aplikační logiku do pomocníků pro znovupoužitelnost napříč různými kontroléry. Tak například byl implementován pomocník pro uložení profilových obrázků a obrázků pro příspěvky. Výsledná metoda a případ použití je na následující ukázce.

Vytvoření profilového obrázku

```
def get_photo(photo)
   attachment = {
      filename: photo[:filename],
      type: photo[:type],
      headers: photo[:head],
      tempfile: photo[:tempfile]
   }
   ActionDispatch::Http::UploadedFile.new(attachment)
end
```

Další užitečnou metodou je `present_error`, která slouží pro vrácení chybových zpráv v jednotném formátu. Implementaci metody je možné vidět na další ukázce. `ErrorCode` je výčet různých chyb obsahujících kód chyby, výchozí zprávu a popis.

Vracení chybové zprávy

```
def present_error(error_code, message: nil, description: nil)
   message ||= error_code.message
   description ||= error_code.description
   error!(
   {
      code: error_code.error_code,
      message: message,
      description: description
   }, error_code.status_code
   )
end
```

Také byla implementovana pomocná metoda pro realizaci key-based stránkování, popsaného v kapitole 4.4.2.2. Tyto metody jsou definované v bloku `helpers` v hlavní API třídě, a proto jsou dostupné všem ostatním kontrolérům, které jsou zodpovědné za různé části API.

5.1.4 Použité knihovny

Serverová aplikace používá následující sadu knihoven (gemů):

- **rails** [38]. Framework Ruby on Rails.

- **pg** [39]. Poskytuje Ruby rozhraní pro práci s PostgreSQL databází.
- **puma** [40]. Aplikační server.
- **bcrypt** [41]. Používá se pro generování otisku hesla uživatele.
- **grape** [42]. Umožňuje jednodušší způsob vytváření API.
- **grape-swagger** [43]. Umožňuje generování dokumentace pro Grape API.
- **paperclip** [44]. Plugin pro ActiveRecord pro práci se soubory.
- **faker** [45]. Slouží pro generování falešných dat pro testování.
- **shoulda** [46]. Používá se pro psaní testů.
- **rspec-rails** [47]. Používá se pro psaní testů.

5.2 Mobilní aplikace

Hlavní důraz v této diplomové práci je kladen na návrh a vývoj právě mobilní aplikace pro operační systém Android, a proto implementaci frontendové části systému bylo věnováno co nejvíce času a úsilí.

5.2.1 Použité knihovny

Žádná Android aplikace se neobejde bez použití vhodných knihoven, zvlášť těch, které se používají skoro v každé mobilní aplikaci pro tento operační systém. V této aplikaci jsou použité následující knihovny:

- **Android support libraries.** Sbírka pomocných knihoven, která má za cíl poskytovat zpětnou kompatibilitu pro novější API, pomocné třídy a utility pro vývoj.
- **Android Architecture Components.** Sbírka knihoven pro návrh robustních mobilních aplikací. Obsahuje prvky LiveData a ViewModel.
- **Anko [48].** Knihovna, která rozšiřuje a usnadňuje vývoj aplikací pro Android. Skládá se z několika částí: Anko Commons, Anko Layouts, Anko SQLite a Anko Coroutines.
- **Retrofit & GSON [49].** Retrofit je REST klient pro Android, GSON je knihovna pro serializaci (případně deserializaci) Java nebo Kotlin objektů do (případně z) JSON reprezentace. Více o těchto knihovnách je si možné přečíst v sekci 5.2.2.
- **Picasso [50].** Umožňuje jednoduchý způsob načítání obrázků v aplikaci.

- **Material Dialogs [51].** Knihovna pro vytváření dialogů v materiálním designu.
- **CircleImageView [52].** Umožňuje efektivní způsob zobrazení kulatých ImageView.
- **uCrop [53].** Poskytuje Aktivitu pro úpravu obrázků (například oříznutí nebo otáčení).
- **MaterialProgressBar [54].** Umožňuje konzistentní zobrazení načítacího prvku (Progress Baru) pro různé verze operačního systému.
- **Stetho [55].** Knihovna pro ladění Android aplikací pomocí Google Chrome.
- **EventBus [56].** Knihovna, která slouží pro výměnu událostí v rámci aplikace. Více je si možné přečíst v sekci 5.2.3.
- **Google Maps SDK [57].** SDK pro zobrazení dynamických map v aplikaci.
- **PlayServices - Location [58].** Knihovna pro zjištění polohy zařízení.
- **SwitchDateTimePicker [59].** Umožňuje zobrazení dialogu v materiálním designu pro výběr data a času.

5.2.2 Retrofit

Retrofit je velmi populární knihovna pro síťovou komunikaci v Android programování. Tato knihovna má skvělou podporu REST API, je snadno testovatelná a integrovatelná, a proto získala oblibu u všech Android programátorů. Po integraci knihovny je potřeba vytvořit modelové třídy a rozhraní. Na ukázce níže je možné vidět, jak vypadá POST požadavek pro přihlášení včetně potřebných modelových tříd.

Příklad deklarace POST požadavku

```
interface Api {
   @POST("api/v1/users/login")
   fun login(@Body userCredentialsEntity: UserCredentialsEntity):
       Call<TokenEntity>
}

data class UserCredentialsEntity(
   @SerializedName("email")
   val email: String,
   @SerializedName("password")
   val password: String
)
```

```
data class TokenEntity(
   @SerializedName("auth_token")
   val authToken: String,
   @SerializedName("id")
   val userId: Int
)
```

U metod jsou důležité uvedené anotace, které ukazují, o jaký typ požadavku se jedná. Parametrem anotace je část cesty bez URL serveru a bez úvodního lomítka.

V parametrech metody se předávají parametry požadavku, na ukázkovém příkladě se předává jen tělo POST požadavku (anotace `@Body`). Navíc se dají specifikovat i URL parametry pomocí anotace `@Query` a parametry cesty (path parameters) pomocí anotace `@Path`.

Modelové entity jsou implementované jako Kotlin data třídy. Anotace `@SerializedName` definuje, jaký název bude mít daný atribut v JSON reprezentaci. Tato anotace není povinná, knihovna Gson umí odvodit názvy JSON atributů z názvů atributů ve třídě, ale po obfuskaci zdrojových souborů se názvy změní a dotazy na API nebudou úspěšné.

Po přípravě rozhraní je potřeba vytvořit instanci Retrofitu a vytvořeného rozhraní a zpřístupnit ji pomocí vzoru Singleton pro zbytek aplikace. Pro vytvoření instance Retrofitu stačí předat základní URL serveru, instanci OkHttpClienta a konvertor, který bude zajišťovat serializaci a deserializaci JSON. Instance `OkHttpClienta` obsahuje nastavení časových prodlev pro zápis a čtení, také má různé Interceptory. Potom pomocí metody `create()`, instance Retrofitu dokáže poskytnout implementaci rozhraní s API metodami. Samotné volání metod může probíhat jak synchronně, tak i asynchronně pomocí Callbacků.

5.2.3 EventBus

Typická Android aplikace se skládá z velkého množství různých komponent, jako jsou například Activities, Fragments, Services nebo Controllers. Efektivní komunikace mezi komponentami může být obtížná, a proto v určitých případech je vhodné použít sběrnici pro účely výměny zpráv. Existuje několik knihoven, které pomáhají v řešení tohoto problému. Jednou z nich je knihovna EventBus, mezi hlavní klady zejména patří::

- Zjednodušení komunikace mezi komponentami
- Oddělení odesílatele a příjemce události (zprávy)
- Dobře funguje s Aktivitami a Fragmenty
- Vyhýbá se komplexním závislostem a předchází chybám v životním cyklu

V implementované mobilní aplikaci sběrnice se používají pro výměnu zpráv mezi komponentami Controller a Repository. V aplikaci existuje celá řada ovladačů, které jsou zodpovědné za úpravu dat: `EventCommentsController` (pro přidání a mazání komentářů), `FollowingController` (pro zahájení a rušení sledování) a další. Z pohledu sběrnice jsou to odesílatele, které přidávají zprávy. Na ukázkovém kódu níže je možné vidět, jak vypadá odeslání zprávy ohledně zrušení komentáře do sběrnice.

Vložení zprávy do sběrnice

```
EventBus.getDefault().post(EventCommentDeleteBusEvent(comment))
```

Třída `EventCommentDeleteBusEvent` je jednoduchá datová třída obsahující jen instanci komentáře, který bude smazán. Přijímači jsou třídy Repository, které poskytují data a obsah, tak například na zprávu `EventCommentDeleteBusEvent` naslouchá `EventCommentsRepository`, který poskytuje seznam komentářů pro určitý příspěvek. Na ukázkovém kódu níže je možné vidět metodu, která se zavolá v okamžiku, kdy se na sběrnici objeví zpráva `EventCommentDeleteBusEvent`. V těle této metody se z aktuálních dat vymaže komentář a nový seznam se nastaví do LiveDat. Tato změna se dostane přes ViewModel do Fragmentu, který provede aktualizaci viditelného pro uživatele seznamu komentářů.

Ukázka implementace pozorování určité zprávy na sběrnici

```
@Subscribe
fun onCommentDeleteBusEvent(event: EventCommentDeleteBusEvent) {
   val data = mLiveData.value?.data?.filterNot { it.id ==
       event.commentEntity.id }
   mLiveData.value?.let { mLiveData.forceSetValue(Resource.copy(it,
       data = data)) }
}
```

5.2.4 Ovládání stavu dat

V průběhu načítání dat se může stát chyba, kterou je potřeba zobrazit uživateli. V této diplomové práci pro účely zobrazení načítacích prvků a chybových hlášek byla použita třída `Resource`. Jedná se o obalovací třídu, která umí rozlišit, v jakém stavu se nachází zdroj dat. ViewModel poskytuje pro UI vrstvu prvek `Resource<List<EventEntity>>`, pomocí kterého dokáže reagovat na stav dat zobrazením načítacích prvků nebo chybových zpráv.

Resource se dá použít spolu s vlastní implementací Observeru, který bude mít pro každý stav svoji vlastní metodu, což je možné vidět na příkladu níže.

Ukázka implementace Resource a ResourceObserver

```
class Resource<T>(private val status: Int, val data: T? = null, val
    message: String? = null)

abstract class ResourceObserver<T> : Observer<Resource<T>> {
   final override fun onChanged(t: Resource<T>?) {
      when {
         t == null -> onUndefined(t)
         t.success -> onSuccess(t.data)
         t.error -> onError(t.status, t.message)
         t.loading -> onLoading()
         else -> onUndefined(t)
      }
   }
   abstract fun onLoading()
   abstract fun onSuccess(data: T?)
   abstract fun onError(status: Int, message: String?)
   fun onUndefined(t: Resource<T>?) {}
}
```

5.2.5 Vytváření rozvržení grafických prvků

Pro vytváření rozvržení grafických prvků (layouts) v této diplomové práci byl použit Constraint Layout, který je poměrně novým způsobem implementace grafické části aplikace a který v posledním roce nabyl popularity, hlavně pomocí podpory komunity a Google. Je to způsob, jak vytvořit responzivní a udržitelné rozvržení grafických prvků v Androidu. V následujících odstavcích čerpám znalosti ze zdrojů [32, 33].

Hlavní myšlenkou Constraint Layoutu je vytváření omezení pro různé grafické prvky uvnitř rozvržení. Omezení definuje vztah mezi dvěma grafickými prvky (Views) v layoutu a řídí, jak budou widgety navzájem umístěny. Omezení se především definují pomocí XML atributu `layout_constraintA_toBOf`, kde `A` a `B` mohou nabývat různých hodnot z množiny: `Start`, `Top`, `End` a `Bottom`. Hodnota daného atributu je buď `id` nějakého view v daném layoutu nebo „parent“. Poslední znamená, že se definuje omezení mezi view a kontejnerem, ve kterém se nachází dané view.

Na ukázkovém kódu níže je možné vidět, že profilový obrázek má omezení vůči kontejneru pro horní a levou hranici. Textová pole musí být po pravé straně profilového obrázku, ale nesmí přesahovat šířku kontejneru, proto oba mají stejná horizontální omezení `layout_constraintStart_toEndOf` a `layout_constraintEnd_toEndOf`. Tato omezení definují, že levá hranice bude zarovnaná na pravou hranici profilového obrázku (s použitím vnějšího okraje) a pravá hranice bude zarovnaná na pravou hranici kontejneru (zase s ohledem na vnější okraj). Aby se tato omezení aplikovala, je potřeba nastavit šířku na `0dp`, tím se řekne, že šířka view je zcela řízená stanovenými omezeními.

Další zajímavou vlastností Constraint Layoutu je řetězení (`chainStyle`). Je to určitý druh omezení, které umožňuje sdílet prostor mezi widgety uvnitř řetězu a řídit, jakým způsobem je tento prostor mezi nimi rozdělen. Je to trochu podobné vahám v LinearLayoutu, ale přináší mnohem více možností. Řetěz má několik režimů: `spread`, `spread_inside` a `packed`.

- Spread

  Je to výchozí režim řetězu, kde widgety (views) budou umístěny v rovnoměrných intervalech v rámci dostupného prostoru.

- Spread Inside

  Zarovnává vnější okraje krajních widgetů (views) v řetězu k vnějším okrajům a poté umístí zbývající widgety v řetězu ve stejných intervalech v rámci dostupného místa.

- Packed

  Shromažďuje dohromady widgety v řetězu. Jejich pozice se dá konfigurovat pomocí atributu `bias`, který má výchozí hodnotu 0,5 (uprostřed dostupného prostoru).

Na ukázkovém kódu je možné vidět, že textová pole jsou uvnitř vertikálního řetězu typu `packed`. Výsledné rozvržení a příslušná omezení podle tohoto kódu je možné vidět na obrázku 5.1.

Ukázka Constraint Layoutu: hlavička s profilovým obrázkem

```
<android.support.constraint.ConstraintLayout
   android:id="@+id/dogBlock"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:paddingTop="@dimen/grid_8"
   android:paddingBottom="@dimen/grid_8">

   <de.hdodenhof.circleimageview.CircleImageView
      android:id="@+id/dogAvatar"
      android:layout_width="@dimen/avatar_size_medium"
      android:layout_height="@dimen/avatar_size_medium"
      android:layout_marginStart="@dimen/grid_8"
      android:src="@drawable/image_avatar_placeholder"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent" />

   <TextView
      android:id="@+id/dogName"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_marginStart="@dimen/grid_8"
```

```
      android:layout_marginEnd="@dimen/grid_2"
      app:layout_constraintBottom_toTopOf="@+id/dogBreedName"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toEndOf="@+id/dogAvatar"
      app:layout_constraintTop_toTopOf="parent"
      app:layout_constraintVertical_chainStyle="packed"
      tools:text="Bubble" />

   <TextView
      android:id="@+id/dogBreedName"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_marginStart="@dimen/grid_8"
      android:layout_marginEnd="@dimen/grid_2"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toEndOf="@+id/dogAvatar"
      app:layout_constraintTop_toBottomOf="@+id/dogName"
      tools:text="Lablador" />
</android.support.constraint.ConstraintLayout>
```

Obrázek 5.1: Ukázka prvku Constraint Layout

Constraint Layout má několik dalších užitečných vlastností, které jsou použité v této práci:

- Group

  Slouží pro logické seskupování určitých prvků (views). Skupina obsahuje pouze odkazy na jiné wigdety a umožňuje měnit viditelnost všech prvků ve skupině tím, že se změní viditelnost skupiny. Skupiny jsou užitečné v případě, kdy obrazovka navíc obsahuje komplexní chybovou hlášku nebo načítací stav, který se skládá z více prvků a kterému je potřeba měnit viditelnost.

- Barrier

  Bariéra je neviditelné view obsahující odkazy na jiné widgety, pro které je vytvořena. Pozice bariéry je vždy určena pozicí krajního widgetu. Bariéry mohou být svislé, nebo vodorovné a mohou být vytvořeny pro horní, dolní, levou nebo pravou stranu referenčních widgetů. Pak ostatní

widgety mohou definovat své omezení vůči bariéře, a tím pádem dosáhnout očekávaného výsledku.

5.2.6 Implementace procházek

Sledování trasy procházky se psem je důležitou součástí výsledné aplikace. Novou procházku je možné zahájit ze seznamu všech procházek, ke kterým je možné se dostat přes hlavní navigační lištu. Samotná obrazovka s mapou je implementována jako Aktivita, která obsahuje `SupportMapFragment` pro zobrazení mapy a elementy uživatelského rozhraní pro zobrazení informací o procházce a také pro zahájení nebo ukončení procházky. `SupportMapFragment` již obsahuje tlačítko pro centrování mapy na polohu uživatele, proto nebylo potřeba přidávat vlastní element pro tuto akci.

Mobilní aplikace musí umožňovat odchod z obrazovky, aniž by se zastavilo sledování trasy procházky. Proto se zahájením procházky se spustí služba (Service), která naslouchá na změny polohy uživatele a poskytuje data o cestě všem zaregistrovaným klientům.

5.2.6.1 Service

Služba (Service) je komponenta Android aplikace, která může provádět dlouhodobé operace na pozadí a poskytuje uživatelské rozhraní. V operačním systému Android se rozlišuje mezi třemi druhy služeb [34]:

- Služby běžící na popředí

  Provádí činnost, která je viditelná pro uživatele, například přehrávání zvukové stopy. Služba běží dále, i když uživatel neinteraguje s aplikací.

- Služby běžící na pozadí

  Provádí činnost, kterou uživatel přímo nevidí, například komprese dat. Ve většině případů mohou být nahrazeny plánovanými úlohami.

- Vázané služby

  Služba je vázána, když se k ní připojí jiná komponenta aplikace pomocí volání `bindService()`. Vázaná služba nabízí rozhraní klient-server, které umožňuje komunikaci se službou, odeslání požadavků a přijetí výsledků. Tato služba běží tak dlouho, dokud je k ní připojen alespoň jeden klient.

Operační systém Android od verze 8 a výše zavádí některé omezení na použití služeb běžících na pozadí. Proto je potřeba používat novou metodu `startForegroundService()`, která spustí novou službu na popředí. Poté, co systém vytvoří službu, má pět sekund na zavolání metody `startForeground()`, čímž se zobrazí notifikace pro uživatele oznamující o běžící službě. Pokud se stane, že tato metoda v časové lhůtě nebude zavolána, systém službu ukončí

a označí aplikaci jako nereagující. Tím vznikne chyba ANR (Application Not Responding).

Služba pro procházky musí běžet na popředí, proto v notifikačním panelu operačního systému musí být zobrazena notifikace o tom, že daná služba běží. Kliknutím na notifikaci se uživatel dostane zpět do aplikace na obrazovku s mapou a vykreslenou cestou.

Služba pozoruje změny polohy uživatele a poskytuje svůj stav každou sekundu. Stavem služby je trojice: čas začátku procházky, seznam souřadnic, pomocí kterých se tvoří cesta a její délka. Každá změna umístění uživatele se přidává do seznamu souřadnic a k délce se připočítává vzdálenost od poslední známé polohy do nové polohy uživatele. Komunikace s registrovanými klienty probíhá pomocí zpráv (`Messenger`).

Pro implementaci `Messengeru` je potřeba:

1. Uvnitř služby vytvořit instanci `Handleru`, který slouží pro přijetí zpráv od klienta.
2. Ve službě vytvořit objekt `Messenger`, který používá předem vytvořený `Handler`.
3. Z `Messengeru` vytvořit `IBinder`, který služba vrátí klientům pomocí příkazu `onBind()`.
4. Klienty používají `IBinder` k vytvoření instance `Messengeru`, kterou pak používají k odesílání zpráv službě a přijetí zpráv ze služby.
5. Služba obdrží každou zprávu v metodě `handleMessage()` `Handleru` vytvořeného v prvním bodě.

Na ukázce kódu je možné vidět periodické odesílání stavu procházky ve službě. Pokud při odeslání zprávy nastane `RemoteException`, tak to znamená, že klient byl odpojen od služby, a je potřeba ho smazat ze seznamu všech připojených klientů. Služba definuje konstantu (například `ServiceMessage.REFRESH`) pro každou zprávu, podle toho klient může rozlišit, o který druh zprávy se jedná.

Ukázka periodického odesílání stavu procházky

```
private fun refresh() {
   sendMessage(ServiceMessage.REFRESH, RefreshEnvelope(mStartTime,
       mPath, mDistance))
}

private fun sendMessage(what: Int, data: Any? = null) {
   for (i in mClients.size - 1 downTo 0) {
      try {
        mClients[i].send(Message.obtain(null, what, data))
```

```
        } catch (e: RemoteException) {
            // The client is irrelevant, remove it from the list.
            mClients.removeAt(i)
        }
    }
}
```

5.2.6.2 Google Map SDK

Pro zobrazení mapy bylo použito Maps SDK od Googlu. Integrace se začíná přidáním příslušné závislosti do projektu a získáním API klíče. Dále stačí přidat prvek `SupportMapFragment` do Aktivity, implementovat rozhraní `OnMapReadyCallback` a metodu `onMapReady(GoogleMap)` a zaregistrovat tento callback pomocí metody `getMapAsync()`, která se pak zavolá na instanci `SupportMapFragmentu`. Potom instance `GoogleMap` může být přiřazena do proměnné pro účely interakce s mapou.

Dalším krokem je potřeba na mapu vykreslit cestu, kterou jde uživatel během své procházky. Pro tyto účely ve třídě `GoogleMap` existuje metoda `addPolyline(PolylineOptions)`. Existují i další metody pro přidání vlastních objektů do mapy, například `addPolygon(PolygonOptions)` nebo `addCircle(CircleOptions)`.

Aktivita, která vlastní instanci `GoogleMap`, dostává aktualizaci cesty přes `ViewModel`, který obsahuje klienta komunikujícího se službu popsanou výše. Po každé aktualizaci souřadnic na mapě bude vykreslena aktuální cesta pomocí metody `redrawPath(List<Location>)`, kterou je možné vidět na ukázce kódu níže.

Ukázka vykreslení cesty na mapě

```
private fun redrawPath(path: List<Location>) {
    mMap.clear()
    val options = PolylineOptions()
        .width(5f)
        .color(ContextCompat.getColor(this, R.color.primary))
        .addAll(path.map { LatLng(it.latitude, it.longitude) })
    mMap.addPolyline(options)
}
```

5.2.6.3 Google Maps Static API

Po ukončení procházky je výsledná cesta uložena jako seznam souřadnic. Pro zobrazení náhledu cesty na mapě (například v detailu procházky nebo události) se používá API, které umí poskytnout snímek mapy s vykreslenou cestou.

Pro použití Google Maps Static API je potřeba získat API klíč a digitální podpis. V oficiální dokumentaci je uvedeno, že digitální podpis je potřeba jen

tehdy, pokud počet požadavků bude převyšovat 25 000 denně, ale při integraci tohoto API se zjistilo, že bez digitálního podpisu mohou dotazy občas selhávat. Tyto atributy se předávají spolu s ostatními atributy v parametrech URL pro získání snímku mapy s potřebnou cestou. Například pro cestu, která obsahuje několik bodů, URL bude vypadat následovně:

Ukázka URL pro získání snímku mapy

```
https://maps.googleapis.com/maps/api/staticmap
   ?center=50.1306775,14.4923994
   &zoom=8
   &size=600x600
   &path=color:0x0000ff|weight:5|50.130769,14.492609|50.130277,14.492695
   &key=API_KEY
   &signature=SIGNATURE
```

V této ukázce `center` znamená souřadnice středu mapy, `zoom` je úroveň přiblížení, `size` je velikost výsledného obrázku v pixelech, `path` obsahuje popis cesty (první parametr je barva, druhý tloušťka čáry a další parametry jsou souřadnice cesty), `key` je API klíč a `signature` obsahuje hodnotu digitálního podpisu. Výpočet středu je relativně jednoduchý, protože je to střed obdélníku ohraničujícího cestu. Úroveň přiblížení musí být správně vypočítána pro každou cestu individuálně, protože cesty mohou být různě velké.

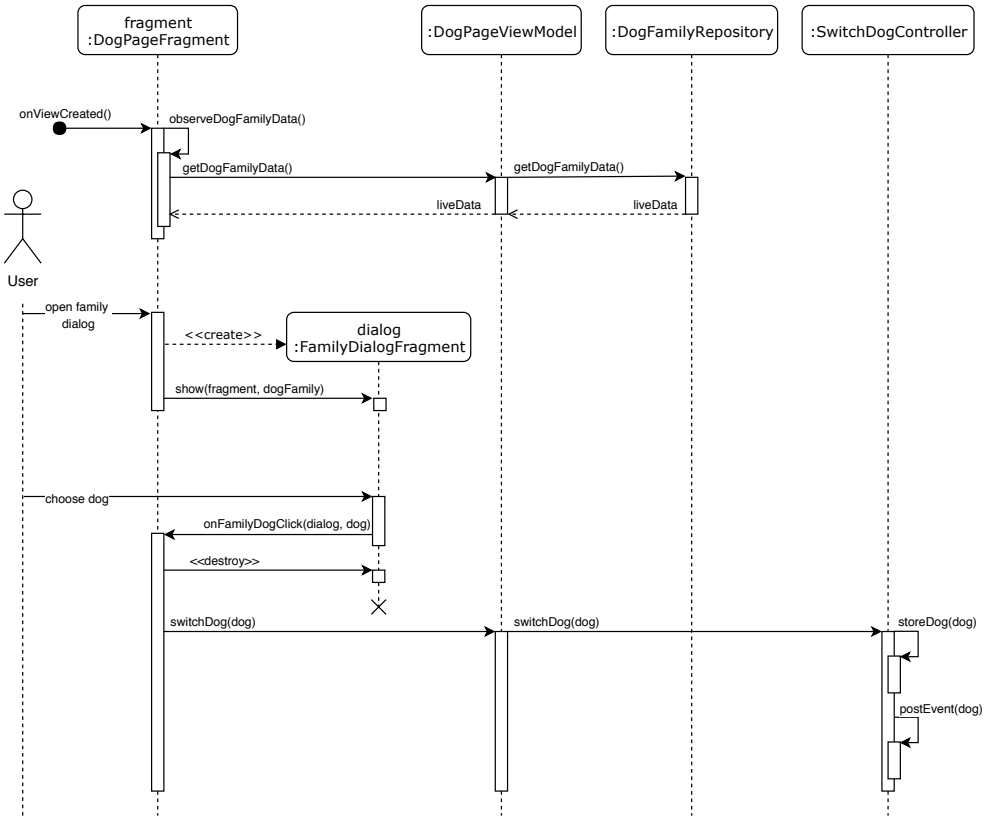
Pro generování digitálního podpisu je potřeba provést následující kroky:

1. Získat v Google Cloud Platform konzoli klíč pro podepisování URL.
2. Sestavit relativní URL požadavek, který již obsahuje API klíč a všechny potřebné parametry.
3. Podepsat výše uvedenou URL pomocí algoritmu HMAC-SHA1 s použitím klíče z prvního bodu.
4. Zakódovat výsledný hash pomocí Base64 a přidat ho jako parametr signature do URL.

5.2.7 Implementace přepínání mezi stránkami psů

Jednou z důležitých funkcí je přepínání mezi stránkami psů, kteří patří do jedné rodiny stejného uživatele. Na obrázku 5.2 je možné vidět sekvenční diagram popisující implementaci přepínání na stránku jiného psa ze své rodiny na domovské obrazovce.

Především po vytvoření obrazovky (Fragmentu) je potřeba začít pozorovat data, která obsahují rodinu psů daného uživatele. Za načítání dat o této rodině je zodpovědný `DogFamilyRepository`, který načítá rodinu uživatele pomocí API, ale navíc umí ukládat výsledek do mezipaměti pro urychlení



Obrázek 5.2: Sekvenční diagram přepínání na stránku jiného psa uživatele

a úsporu síťové komunikace. Když uživatel otevře dialog, tak se vytvoří instance objektu `FamilyDogDialog`, která bude obsahovat pozorovaná data. Po zvolení psa se zavolá Callback na Fragmentu, který uzavře dialog a předá zvoleného psa do ViewModelu. Logika změny psa se používá u několika ViewModelů (na diagramu je uveden jen jeden případ), proto je vhodné vyčlenit tuto logiku do samostatné třídy `SwitchDogController`. Tato třída má metodu `switchDog(dog)`, která perzistentně ukládá volbu uživatele a pomocí sběrnice rozesílá příslušnou událost přepnutí psa všem komponentům, které jsou zaregistrované na tuto událost.

Pro jednoduchost v diagramu není uvedeno načítání detailu a událostí vybraného psa. Jedná se o zavolání `refresh()` metod na dvou instancích Repository, které se nacházejí ve ViewModelu.

5.2.8 Implementace přidání komentáře k příspěvku

Uživatel má možnost přidávat komentáře k příspěvku. Pro odeslání požadavku na přidání komentáře je potřeba mít trojici: identifikátor události, identifikátor autora komentáře (psa) a vlastně text. Sekvenční diagram popisující tento

proces je si možné prohlédnout na obrázku 5.3.

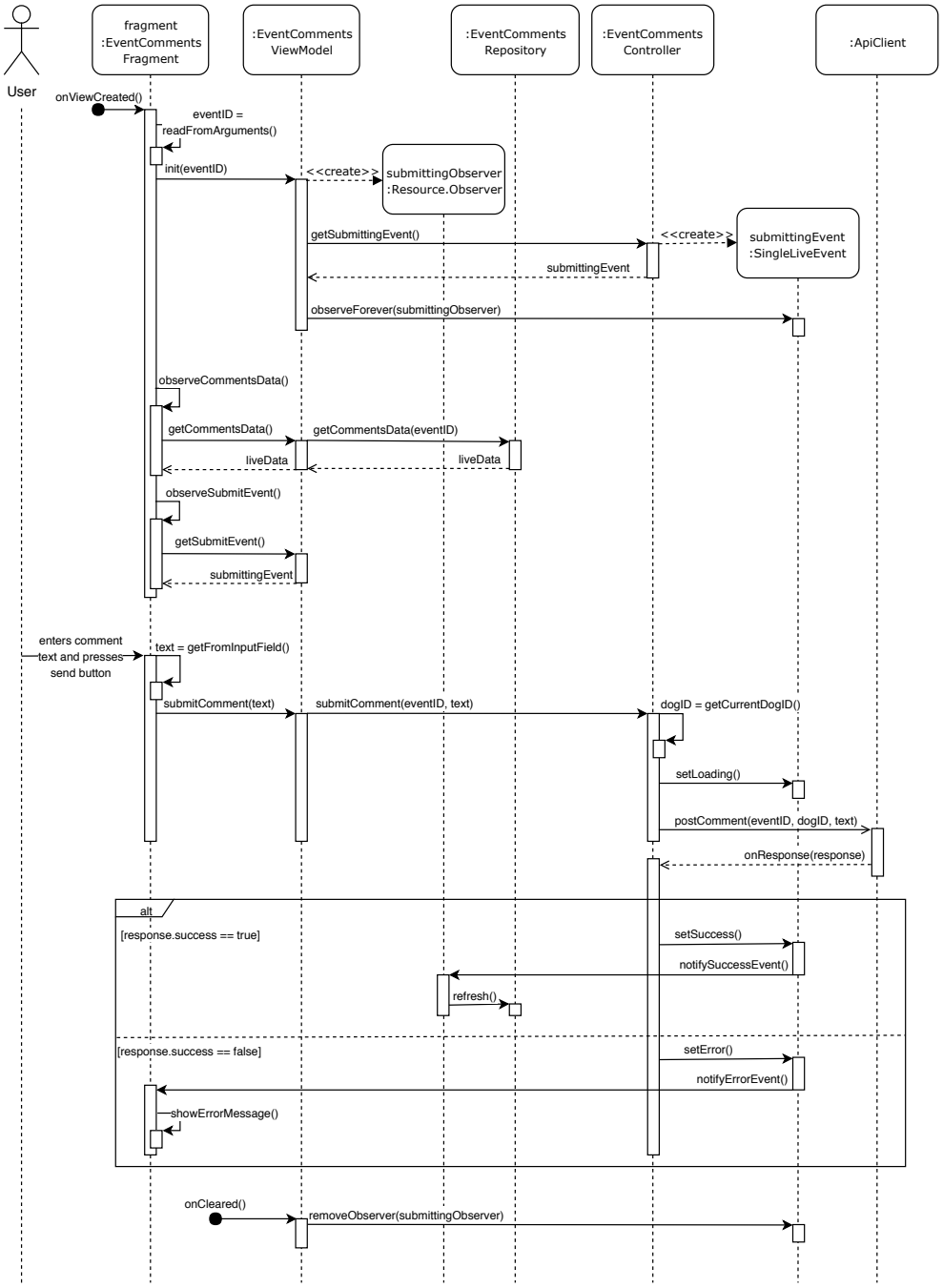
Na začátku po vytvoření obrazovky je potřeba provést inicializaci ViewModelu. V operačním systému Android parametry pro Fragmenty se předávají pomocí argumentů. Tak pro obrazovku se seznamem komentářů se předává identifikátor příspěvku, který slouží pro inicializaci ViewModelu. V průběhu inicializace se vytvoří Observer, který bude pozorovat úspěšné odesílání komentářů. Podobný Observer se pak vytvoří ve Fragmentu a slouží pro zobrazení hlášky o neúspěchu, v případě, že se nepodaří uložit komentář na serveru. Tyto Observery se zaregistrují na změnu hodnoty `SingleLiveEventu`, který je poskytován objektem `EventsCommentController`. Po inicializaci Fragment začne pozorovat LiveData s komentáři daného příspěvku. Za načítání a poskytování těchto dat je zodpovědný `EventsCommentRepository`.

V čase, kdy uživatel vyplní text komentáře a stiskne tlačítko, se zahájí proces odesílání komentáře na server. Fragment přečte text z textového pole a předá ho do ViewModelu. Ten už zná identifikátor příspěvku, a proto předá ho spolu s textem komentáře do objektu `EventsCommentController` pro odeslání požadavku na server. Controller získá identifikátor aktuálně vybraného psa z perzistentního úložiště a provede asynchronní API volání. Controller získá výsledek volání pomocí Callbacku a podle toho, jestli volání bylo úspěšné, se vytvoří zdroj (Resource) se správným stavem a nastaví ho jako hodnotu do `SingleLiveDat`. Na toto zareagují předem zaregistrované Observery a buď se provede aktualizace komentářů pomocí `EventsCommentRepository` anebo se zobrazí hláška o neúspěchu. Po aktualizaci komentářů se změny automaticky projeví ve Fragmentu.

5.2.9 Implementace stránkování na straně klienta

V návrhu serverové části (sekce 4.4.2) již byl popsán použitý způsob stránkování na serveru, což předpokládá, že mobilní aplikace musí být přizpůsobena pro použití stránkování. Podle Architecture Components za načítání dat musí být zodpovědné `Repository` třídy. Protože ve výsledné aplikaci existuje velké množství seznamů, které potřebují implementaci stránkování, tak bylo rozhodnuto, že tuto logiku je vhodné vyčlenit do samostatné třídy, ze které budou zděděné ostatní `Repository`, které podporují postupné načítání. Tato třída je zodpovědná za uchovávání klíče posledního prvku, který bude předán do dalších požadavků. Také tato třída obsahuje pomocnou metodu pro zjištění, jestli na serveru existují další data nebo dotaz již vrátil poslední prvky.

Pro zobrazení seznamu v mobilní aplikaci se používá `RecyclerView`. Jeho hlavní výhodou je to, že `RecyclerView` umí recyklovat položky při skrolování. Používá se spolu s Adapterem, který slouží pro převod modelových tříd do vizuální podoby. Pro implementaci načítacího kolečka na konci seznamu byl použit Adapter, který umí zobrazit několik typů modelů v jednom seznamu. Tak ViewModel bude poskytovat data, které načítá Repository ze serveru a přidává na konec speciální entitu načítacího kolečka. V aplikaci se toto projeví



Obrázek 5.3: Sekvenční diagram přidání komentáře k příspěvku

tak, že při skrolování již načteného seznamu na konci se objeví načítací kolečko, které se v okamžiku ukončení stažení vymění za nově stažená data.

5.2.10 Navigace v aplikaci

Navigace v aplikaci je implementována pomocí `BottomNavigationView`, které bylo nastylované a upravené pro docílení požadovaného chování a vzhledu. Standardní implementace v Android aplikacích pomocí tohoto View předpokládá, že se vytvoří jedna Aktivita, která obsahuje `FrameLayout` a `BottomNavigationView`. `FrameLayout` potom funguje jako kontejner pro jednotlivé Fragmenty, které reprezentují každou navigační záložku. Nevýhodou tohoto řešení je to, že po změně záložky stav předchozí záložky je zahozen, tím pádem, například, uživatel může přijít o pozici v seznamu.

Jelikož standardní chování `BottomNavigationView` je nevyhovující, bylo rozhodnuto, že jedna Aktivita bude obsahovat pět `FrameLayoutů`, jeden pro každou záložku v navigační liště. Při vytvoření Aktivity bude obsah kontejnerů vyplněn odpovídajícími Fragmenty. Výsledná navigace pak funguje jako změna viditelnosti jednotlivých kontejnerů. Toto řešení není paměťově úsporné, ale žádný z kořenových Fragmentů neobsahuje náročné prvky (například mapy), proto tento způsob může být považován za vhodný.

Také pro implementaci navigace byla vyzkoušena i knihovna Navigation Components, která v době psaní této diplomové práce neumožňovala výše popsané chování a navíc obsahovala kritické nedostatky.

Kapitola 6

Testování

6.1 Testování programátorem

Testování programátorem bylo provedeno pokaždé po implementaci dalšího bloku kódu nebo po přidání nové funkčnosti. Cílem tohoto testování bylo ověřit funkčnost implementace a odhalit kritické chyby. Na konci vývojového procesu všechny důležité funkčnosti byly vyzkoušené na různých zařízeních s různými operačními systémy. Seznam těchto zařízení je možné vidět v tabulce 6.1.

Pro testování byla záměrně zvolena zařízení s různými verzemi operačního systému Android, protože způsob implementace pro tyto verze se může lišit. Také pro vyzkoušení a opravu vzhledu na malých obrazovkách byl vytvořen emulátor s velikostí obrazovky 4,5".

6.1.1 Robolectric testy

Robolectric je framework pro psaní jednotkových testů mobilních aplikací pro operační systém Android. Jednou z největších výhod tohoto nástroje je spuštění testů na JVM, což znamená, že pro testování není potřeba reálné zařízení nebo emulátor. Robolectric poskytuje vlastní verzi souboru android.jar kompatibilní s JVM, také zpracovává vložení View, načítání zdrojů a řadu dalších věcí, které jsou implementované v nativním kódu na Android zaříze-

Tabulka 6.1: Technické parametry testovacích zařízení

	LG G3	OnePlus 2	Pixel 2	Emulátor
Verze OS	4.4	6.0	9.0	7.1.1
Velikost displeje	5,5"	5,5"	5,0"	4,5"
Rozlišení displeje	2560 x 1440	1920 x 1080	1920 x 1080	900 x 480
Operační paměť	3 GB	4 GB	4 GB	1,5 GB

ních. Spuštění testů na JVM významně zmenšuje čas trvání testů a umožňuje jejich jednoduché spuštění v integračním prostředí (Continuous Integration) bez doplňujících nastavení.

Robolectric neumí vykonávat nativní kód Androidu a komunikovat se systémovými službami, proto nahrazuje všechny Android třídy takzvanými „stíny“ (anglicky Shadows). Každý stín může modifikovat nebo rozšiřovat chování příslušné třídy v operačním systému Android. Když je nějaká Android třída inicializována, Robolectric hledá příslušnou stínovou třídu a pokud ji najde, vytvoří stínový objekt, který se k ní přidruží. Stínové třídy mohou být použité pro mockování vlastních tříd. [35]

Pomocí daného frameworku byly otestované kontroléry, které jsou zodpovědné za zpracování interakcí uživatele, například přidání příspěvku do oblíbených, odstranění příspěvku, změna aktuálně vybraného psa a jiné.

6.1.2 RSpec

Pro testování serverové části aplikace byla použita knihovna RSpec [47], která je jedním z nejznámějších testovacích frameworků pro jazyk Ruby [36]. Knihovna je primárně určena pro podporu Behavior Driven Development, který popisuje chování z pohledu koncového uživatele. Spolu s knihovnou RSpec byla použita knihovna FactoryBot [60], která pomáhá vytvářet testovací instance potřebné pro provedení testů pomocí továren (anglicky Factory).

Na serverové části byly otestované kontroléry spravující příspěvky a komentáře uživatelů a také byly napsané testy validující správnost modelů.

6.2 Uživatelské testování

Testování použitelnosti (anglicky usability testing) je jednou z nejlepších cest, jak ověřit funkčnost a intuitivnost výsledné aplikace z pohledu obyčejného uživatele.

Testování použitelnosti se dělí na dva druhy: kvantitativní a kvalitativní testování. V kvantitativním testování se provádí analýza chování velkého množství uživatelů. Jednou z nejpopulárnějších technik tohoto testování je provedení A/B testů, které testují několik možných variant stejného prvku a zjišťují, jaká varianta má lepší vliv na uživatele. Dalším známým příkladem je analýza chování uživatelů pomocí různých nástrojů, jako je například Google Analytics. Kvantitativní testování se většinou provádí po vydání aplikace do produkce, a proto nebude součástí této diplomové práce.

Kvalitativní testování použitelnosti se naopak zaměřuje na analýzu chování malého počtu lidí, kteří reprezentují cílovou skupinu uživatelů výsledné aplikace. Cílem tohoto testování je zjistit problémy spojené s použitelností produktu a ocenit spokojenost uživatelů s aplikací. Aplikace byla vyvíjena především pro budoucí uživatele, a proto jejich názor je možností ověřit kva-

litu produktu a vylepšit aplikaci tak, aby odpovídala skutečným očekáváním. Tento druh testování se skládá z následujících fází [28]:

- Hledání vhodných kandidátů pro testování
- Příprava vstupního dotazníku
- Příprava testovacích scénářů nebo úkolů
- Příprava výstupního dotazníku
- Provedení testování
- Analýza výsledků

6.2.1 Příprava a průběh testování

Podle [37] pro odhalení většiny problematických míst v aplikaci, není potřeba, aby se testování zúčastnilo víc než pět lidí. Také je důležité, aby testování použitelnosti bylo provedeno lidmi, kteří reprezentují cílovou skupinu uživatele. Proto jednou z nejdůležitějších podmínek účasti v uživatelském testování aplikace „Woof!“ bylo vlastnictví psa, což výrazně zúžilo počet kandidátů. Ve výsledném testování se zúčastnili čtyři lidí, kde dva z nich vlastní psa a každý používá telefon s operačním systémem Android. Tento počet lidí pomůže odhalit přibližně 75 % problémů s uživatelským rozhraním. Respondenti před začátkem testování vyplnili vstupní dotazník, pomocí kterého byla sestavena krátká charakteristika, kterou je možné vidět níže. Vstupní dotazník a jeho výsledky je možné si prohlédnout v příloze F.

- Respondent A

  Žena, 18-26 let, je majitelkou malého psa, používá operační systém Android, tráví v průměru 5 hodin na internetu, z nich 3-4 hodiny tráví v sociálních sítích, jednou týdně chodí na dlouhé procházky se psem.

- Respondent B

  Muž, 26-34 let, je majitelem velkého psa, používá operační systém Android, tráví v průměru 3 hodiny na internetu, nepoužívá sociální sítě, často chodí na dlouhé procházky se psem.

- Respondent C

  Žena, 26-34 let, nevlastní psa, používá operační systém Android, tráví v průměru 3 hodiny na internetu, z nich 2 hodiny tráví v sociálních sítích.

- Respondent D

  Žena, 18-26 let, nevlastní psa, používá operační systém Android, tráví v průměru 4 hodiny na internetu, z nich 2 hodiny tráví v sociálních sítích.

Testování použitelnosti probíhalo v obyčejné místnosti s výjimkou scénáře „Sledování trasy procházky se psem“, který byl proveden na ulici pro správné fungování GPS. V průběhu provádění scénářů bylo zajištěno správné fungování aplikačního serveru a rychlé internetové připojení, aby uživatelské testování nebylo rušené zbytečnými chybami.

Respondenti měli za úkol splnit následující scénáře.

1. Registrace a přidání psa

   Spusťte aplikaci a zaregistrujte se. Při registraci nemusíte uvádět správnou elektronickou adresu, například můžete použít usertestX@gmail.com, kde X může být libovolné číslo. Po úspěšné registraci vytvořte stránku svého psa podle pokrokového návodu.

2. Přidání příspěvku

   Přidejte první příspěvek na stránku Vašeho psa. Příspěvek by měl obsahovat kategorii, popisek a obrázek.

3. Hledání stránek pro sledování

   Vyhledejte nové zajímavé stránky psů a začněte sledování. Vyhledejte aspoň jednoho psa plemena Husky a začněte ho sledovat.

4. Hledání psa

   Najděte psa se jménem Harley a prohlédněte si všechny jeho příspěvky. Dejte „to se mi líbí“ některým příspěvkům.

5. Přidání komentáře k příspěvku

   Přejděte na libovolnou stránku, kterou sledujete a napište komentář na jeden z příspěvků na této stránce.

6. Sledování trasy procházky se psem

   Představte si, že jdete do parku venčit psa a chcete zaznamenat trasu procházky se psem. Sledujte trasu procházky po dobu jedné minuty, pak uložte procházku. Sdělte tuto trasu jako příspěvek na stránce svého psa.

7. Přidání dalšího psa

   Zaregistrujte dalšího psa. Pokud nemáte ještě jednoho psa, údaje o něm si vymyslete.

8. Přepínání mezi stránkami psů

   Přepněte se zpět na prvního psa.

Po splnění všech scénářů každý respodent vyplnil výstupní dotazník, který měl za cíl zjistit postřehy z testování a celkovou spokojenost s aplikací. Tento dotazník obsahoval jak otázky zaměřené na nějaké konkrétní části aplikace, tak i obecné otázky pro zjištění celkového dojmu. Výstupní dotazník spolu s odpovědí je možné vidět v příloze G.

6.2.2 Analýza výsledků

Testování použitelnosti proběhlo bez významných problémů a respondenti bez větších potíží splnili všechny scénáře. Nehledě na úspěšný průchod scénáři, testování pomohlo zjistit cenné postřehy k vylepšení aplikace, které budou zapracované ve finální verzi aplikace.

- **Problem:** při registraci nového psa se nezobrazoval celkový počet kroků, a proto uživatelé nemohli vědět, kolik otázek zbývá do konce.

  Závažnost: 5/5.

  Řešení: přidání indikátoru pro zobrazování počtu otázek, které zbývají do konce.

- **Problem:** po registraci psa aplikace otevírá stránku s novinkami sledovaných psů, které ještě neexistují na moment začátku používání aplikace.

  Závažnost: 5/5.

  Řešení: po registraci aplikace otevírá profilovou stránku.

- **Problem:** ikona přepnutí mezi stránkami psů se zobrazovala i v případě, kdy uživatel měl jen jednoho psa, což bylo velmi matoucí pro uživatele. Po kliknutí na tuto ikonu se zobrazoval dialog, který neměl žádné použití a jen zobrazoval profilový obrázek jediného psa uživatele.

  Závažnost: 4/5.

  Řešení: skrýt ikonu pro změnu aktivní stránky, pokud má uživatel registrovaného jen jednoho psa.

- **Problem:** na začátku registrace uživatel se snažil kliknout na „log in“, nikoliv na „start now“.

  Závažnost: 4/5.

  Řešení: změnit text „start now“ na „register“.

- **Problem:** po registraci psa bylo potřeba potvrdit přechod na profilovou stránku.

  Závažnost: 3/5.

Řešení: po registraci psa přidat automatický přechod na profilovou stránku.

- **Problem:** notifikace, která se zobrazuje v průběhu sledování trasy procházky se psem, neobsahovala žádnou užitečnou informaci.

  Závažnost: 2/5.

  Řešení: zobrazování délky ušlé cesty a čas trvání procházky. Také byla přidána možnost možnosti ukončit procházku přímo z notifikace.

- **Problem:** při hledání možností přepínání aktivní stránky psa respondent se snažil kliknout na profilový obrázek právě vybraného psa, nikoliv na ikonu rodiny psa.

  Závažnost: 1/5.

  Řešení: kliknutí na profilový obrázek psa v hlavičce profilu také bude zobrazovat dialog pro změnu aktivní stránky.

Jedním z nejlepších zjištění z testování použitelnosti bylo to, že respondenti plnili některé úkoly odlišnými cestami. Například jeden respondent přidal obrázek pomocí galerie, druhý naopak použil možnost volby z posledních pořízených obrázků. Dalším příkladem může být to, že jeden respondent vyhledal psa se jménem Harley pomocí vyhledávacího políčka, a druhý si všiml, že již má tohoto psa ve sledovaných.

Kapitola 7

Výsledky a další rozvoj

Výsledkem této diplomové práce je plně funkční prototyp aplikace pro majitele psů, který splňuje všechny na něj kladené požadavky na začátku práce. Snímky výsledné aplikace je možné vidět v příloze H.

V době psaní této diplomové práce nebyla aplikace dostupná pro veřejnost, ale byla nasazena v alfa kanálu Google Play, ke kterému může přistupovat omezený počet lidí pomocí speciálního odkazu.

Před finálním nasazením aplikace do produkce je potřeba dokončit následující body:

- Testování

  Před finálním nasazením by bylo vhodné provést další testování na různých zařízeních s různými operačními systémy. Také je potřeba sledovat výsledky alfa testování v Google Play konzoli a opravovat případné pády a ANR.

- Přihlášení pomocí Google

  Všechny analyzované na začátku sociální sítě obsahovaly přihlášení pomocí alternativních systémů, jako je například Google. Proto před finálním vydáním by bylo vhodné přidat tento typ registrace a přihlášení.

- Ochrana osobních údajů a GDPR

  Před nasazením aplikace do produkce je potřeba prozkoumat právní podmínky vydání tohoto typu aplikací a přidat požadované prvky do aplikace, jako je například souhlas s ochranou osobních údajů před registrací.

I když návrh a vývoj této aplikace si vyžádal mnoho úsilí a přibližně rok práce, existuje velké množství nápadů a myšlenek pro vylepšení této aplikace. Tato vylepšení jsou dalšími kroky, které mohou být provedené v rámci rozvoje po nasazení aplikace do produkce.

Dále jsou uvedené nejdůležitější kroky, které by mohly vylepšit aplikaci po nasazení do produkce:

- Soukromé zprávy

  Soukromé zprávy jsou nejlepším způsobem pro komunikaci mezi lidmi, což komentáře v plné míře nahradit nemůžou. Proto jedním z prvních kroků po zveřejnění aplikace by mohlo být přidání soukromých zpráv.

- Google Analytics nebo Firebase Analytics

  Jak bylo naznačené dříve, Google Analytics nebo Firebase Analytics jsou jedněmi z nejlepších nástrojů na analýzu chování velkého množství uživatelů. Výstupy této analýzy by mohly pomoct zlepšit aplikaci.

- Push notifikace

  Ve výsledné aplikaci se uživatel může dozvědět o nových komentářích nebo přidání do oblíbených jen pomocí sekce novinek. Jedním z vylepšení by se mohlo stát přidání push notifikací pro oznámení uživatele o nové události.

- Vývoj pro další platformy

  Sociální sítě jsou založené na tom, že pro jejich úspěch je potřeba aktivní účast velkého množství lidí. Omezení na operační systém Android je významným faktorem, který zužuje počet potenciálních uživatelů, proto by bylo vhodné se zamyslet nad rozšířením do dalších platforem.

Závěr

Cílem této diplomové práce bylo navrhnout mobilní aplikaci pro operační systém Android pro majitele psů, která by umožňovala sdílení fotografií a významných okamžiků ze života domácích mazlíčků. Také aplikace by měla umožňovat pořizování záznamu trasy procházky se psem a hledání lidí pro společné venčení. Součástí byl také návrh serverové části pro ukládání a poskytování dat uživatelů. Dále bylo cílem podle návrhu vyvinout funkční prototyp této aplikace, který následně bylo potřeba podrobit vhodným testům.

Všechny cíle této diplomové práce byly naplněny. Před začátkem návrhu byla provedena analýza stávajících konkurenčních řešení a byly zjištěné hlavní problémy, které se vyskytují u tohoto druhu aplikací. Dalším krokem bylo provedení průzkumu cílové skupiny uživatelů, po kterém následovala definice funkčních požadavků na aplikaci. Dále následoval samotný návrh součástí aplikace a jejich komunikace. Při vytváření návrhu mobilní aplikace bylo dbáno na přehlednost a použitelnost uživatelského rozhraní. Následně podle návrhu byly vyvinuty serverová a klientská část aplikace. Při vývoji mobilní aplikace byl hlavní důraz kladen na implementaci udržitelné architektury a použití nových doporučení pro vývoj pro operační systém Android. V závěrečné části této práce byla mobilní aplikace otestována pomocí Robolectric testů a také bylo provedené testování použitelnosti.

Práce na tomto projektu mi umožnila využít mé znalosti ze studia v praxi a zdokonalit své dovednosti v oblasti Android programování, čemu se hodlám profesně věnovat v nejbližší budoucnosti. Také tato diplomová práce mi přinesla mnoho nových zkušeností s technologiemi, které mi dříve nebyly známy, jako jsou například návrh a vývoj RESTful API v Ruby on Rails nebo testování Android aplikací pomocí Robolectric testů.

Výsledkem této diplomové práce je funkční prototyp, který je připraven pro další rozvoj. Na návrhu a vývoji této aplikace jsem strávila velké množství času, a proto bych nerada práci na tomto projektu zanechala. Pevně věřím, že tato aplikace bude dobrým základem pro vznik a rozvoj nové sociální sítě pro majitele psů, která se zalíbí jejím potenciálním uživatelům.

Literatura

[1] Business Competence. *Dogalize - Pet Social Network* [software]. [přístup 12. března 2018]. Dostupné z: https://play.google.com/store/apps/details?id=com.dogalize&hl=en

[2] Octopepper. *Yummypets - Dogs Cats Network* [software]. [přístup 25. března 2018]. Dostupné z: https://play.google.com/store/apps/details?id=com.octopepper.yummypets&hl=en

[3] Bauwow Ltd. *Bauwow The Pet Social Network* [software]. [přístup 6. dubna 2018]. Dostupné z: https://play.google.com/store/apps/details?id=com.bauwowworld.bauwow

[4] Pack. *Pack Dog* [software]. [přístup 16. dubna 2018]. Dostupné z: https://itunes.apple.com/us/app/pack-dog-post-your-dog-photos-meet-dog-owners-by-breed/id890725471?mt=8

[5] Mobile Operating System Market Share Worldwide. *Statcounter* [online]. [cit. 2018-08-15]. Dostupné z: http://gs.statcounter.com/os-market-share/mobile/worldwide

[6] Mobile Operating System Market Share United States Of America. *Statcounter* [online]. [cit. 2018-08-15]. Dostupné z: http://gs.statcounter.com/os-market-share/mobile/united-states-of-america

[7] CALLAHAM, John. The history of Android OS: its name, origin and more. *Android Authorit* [online]. 2018 [cit. 2018-08-15]. Dostupné z: https://www.androidauthority.com/history-android-os-name-789433/

[8] YARMOSH, Ken. Android vs iOS: Which platform to build for first? *Savvy Apps* [online]. [cit. 2018-08-25]. Dostupné z:

https://savvyapps.com/blog/android-vs-ios-which-platform-to-build-for-first

[9] Google Inc. Android Developer. *Support different platform versions* [online]. [cit. 2018-08-26]. Dostupné z: https://developer.android.com/training/basics/supporting-devices/platforms

[10] Google Inc. Android Developer. *Dashboard* [online]. [cit. 2018-08-26]. Dostupné z: https://developer.android.com/about/dashboards/

[11] EMINBAYLI, Suleyman. Kotlin vs Java for Android Developing – 2018. *DZone* [online]. [cit. 2018-10-22]. Dostupné z: https://dzone.com/articles/kotlin-vs-java-for-android-developing-2018

[12] DENISCO RAYOME, Alison. Why Kotlin is exploding in popularity among young developers. *TechRepublic* [online]. [cit. 2018-10-23]. Dostupné z: https://www.techrepublic.com/article/why-kotlin-is-exploding-in-popularity-among-young-developers/

[13] *Kotlin: Kotlin Reference* [online]. [cit. 2018-10-28]. Dostupné z: https://kotlinlang.org/docs/reference/

[14] Using Kotlin for Android Development. *Kotlin: Kotlin Reference* [online]. [cit. 2018-10-28]. Dostupné z: https://kotlinlang.org/docs/reference/android-overview.html

[15] Google Inc. Develop Android apps with Kotlin. *Developer Android* [online]. [cit. 2018-10-29]. Dostupné z: https://developer.android.com/kotlin/

[16] KOTIA, Neeti. Kotlin vs Java: Which will Succeed Android Development in Coming Times? *Konstant Infosolutions* [online]. [cit. 2018-10-28]. Dostupné z: https://www.konstantinfo.com/blog/kotlin-vs-java/

[17] KUMAR, Vipin. Android Architecture Patterns. *Medium* [online]. [cit. 2018-11-05]. Dostupné z: https://medium.com/mindorks/android-architecture-patterns-mv-c-p-vm-4594574eeaa1

[18] SHARMA, Ankit. Why to choose MVVM over MVP. *AndroidPub* [online]. [cit. 2018-11-05]. Dostupné z: https://android.jlelse.eu/why-to-choose-mvvm-over-mvp-android-architecture-33c0f2de5516

[19] Google Inc. Guide to app architecture. *Developer Android* [online]. [cit. 2018-11-05]. Dostupné z: https://developer.android.com/jetpack/docs/guide

[20] CHORDIYA, Akshay. Introduction to Android Architecture Components. *AndroidPub* [online]. [cit. 2018-11-05]. Dostupné z: https://android.jlelse.eu/introduction-to-android-architecture-components-22b8c84f0b9d

[21] LABUNSKIY, Evgeniy. What comes first: Functional or non-Functional Requirements? *Medium* [online]. [cit. 2018-09-15]. Dostupné z: https://medium.com/agiletransformation/what-comes-first-functional-or-non-functional-requirements-b3ee96424742

[22] SILVA, Nishadha. Use Case Diagram Tutorial. *The Creately Blog* [online]. [cit. 2018-09-26]. Dostupné z: https://creately.com/blog/diagrams/use-case-diagram-tutorial/#generalization

[23] EICHBERG, Michael. *Domain Model and Domain Modeling.* Darmstadt, Technische Universität Darmstadt. Department of Computer Science. Přednáška. Dostupné z: http://stg-tud.github.io/eise/WS11-EiSE-07-Domain_Modeling.pdf

[24] MATTHIAS, Biehl. *RESTful API Design: APIs your consummers will love.* Rotkreuz: Api-University Press, 2016. API-University Series. ISBN 15-147-3516-4.

[25] The art of pagination – Offset vs. value based paging. *NovaTec* [online]. 2013 [cit. 2018-12-02]. Dostupné z: https://blog.novatec-gmbh.de/art-pagination-offset-vs-value-based-paging/

[26] COOPER, Alan, Robert REIMANN, Dave CRONIN a Alan COOPER. *About face: the essentials of interaction design.* Fourth edition. Indianapolis, IN: John Wiley, [2014]. s 3-21. ISBN 978-1-118-76657-6.

[27] REISS, Eric L. *Usable usability: simple steps for making stuff better.* Indianapolis, IN: John Wiley, c2012. ISBN 978-1-118-18547-6.

[28] ŽIKOVSKÝ, Pavel. *Návrh uživatelského rozhraní.* Praha, ČVUT, 2011. Nepublikované přednášky.

[29] NIELSEN, Jakob. 10 Usability Heuristics for User Interface Design. *Nielsen Norman Group* [online]. [cit. 2018-10-15]. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/

[30] Getting Started with Rails. *RailsGuides* [online]. [cit. 2018-12-15]. Dostupné z: https://guides.rubyonrails.org/getting_started.html

[31] Active Record Basics. *RailsGuides* [online]. [cit. 2018-12-15]. Dostupné z: https://guides.rubyonrails.org/active_record_basics.html

[32] Google Inc. ConstraintLayout. *Android Developer* [online]. [cit. 2018-10-29]. Dostupné z: https://developer.android.com/reference/android/support/constraint/ConstraintLayout

[33] FRANKS, Rebecca. *ConstraintLayout – guidlines, barriers, chains and groups* [online]. [cit. 2018-10-29]. Dostupné z: https://riggaroo.co.za/constraintlayout-guidelines-barriers-chains-groups/

[34] Google Inc. Services overview. *Android Developer* [online]. [cit. 2018-11-06]. Dostupné z: https://developer.android.com/guide/components/services

[35] *Robolectric* [online]. 2010. [cit. 2018-12-27]. Dostupné z: http://robolectric.org/

[36] Test Frameworks. *The ruby toolbox* [online]. [cit. 2018-12-27]. Dostupné z: https://www.ruby-toolbox.com/categories/testing_frameworks

[37] NIELSEN, Jakob. Why You Only Need to Test with 5 Users. *Nielsen Norman Group* [online]. [cit. 2019-01-05]. Dostupné z: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

[38] Ruby on Rails. *Ruby on Rails gem*. Verze 5.1.4. [software]. [přístup 12. února 2018]. Dostupné z: https://github.com/rails/rails

[39] PG Ruby Gem. *pg*. Verze 0.21.0. [software]. [přístup 12. února 2018]. Dostupné z: https://rubygems.org/gems/pg/

[40] Puma. *Puma: A Ruby Web Server Built For Concurrency*. Verze 3.7. [software]. [přístup 13. února 2018]. Dostupné z: https://github.com/puma/puma

[41] HALE, Coda. *bcrypt*. Verze 3.1.12. [software]. [přístup 16. února 2018]. Dostupné z: https://github.com/codahale/bcrypt-ruby

[42] RUBY GRAPE. *Grape*. Verze 1.0. [software]. [přístup 4. února 2018]. Dostupné z: https://github.com/ruby-grape/grape

[43] RUBY GRAPE. *Grape Swagger*. Verze 0.27.3. [software]. [přístup 25. února 2018]. Dostupné z: https://github.com/ruby-grape/grape-swagger

[44] Thoughtbot. *Paperclip*. Verze 5.1.0. [software]. [přístup 20. února 2018]. Dostupné z: https://github.com/thoughtbot/paperclip

[45] CURTIS, Benjamin. *Faker*. Verze 1.8. [software]. [přístup 25. února 2018]. Dostupné z: https://github.com/stympy/faker

[46] Thoughtbot. *Shoulda*. Verze 3.5. [software]. [přístup 25. února 2018]. Dostupné z: https://github.com/thoughtbot/shoulda

[47] Rspec. *Rspec Rails*. Verze 3.7.2. [software]. [přístup 25. února 2018]. Dostupné z: https://github.com/rspec/rspec-rails

[48] KOTLIN. *Anko*. Verze 0.10.4. [software]. [přístup 15. srpna 2018]. Dostupné z: https://github.com/Kotlin/anko

[49] SQUARE. *Retrofit*. Verze 2.4.0. [software]. [přístup 16. srpna 2018]. Dostupné z: https://github.com/square/retrofit

[50] SQUARE. *Picasso*. Verze 2.71828. [software]. [přístup 20. srpna 2018]. Dostupné z: https://github.com/square/picasso

[51] FOLLESTAD, Aidan. *Material Dialogs*. Verze 0.9.6.0. [software]. [přístup 20. srpna 2018]. Dostupné z: https://github.com/afollestad/material-dialogs

[52] DODENHOF, Henning. *CircleImageView*. Verze 2.2.0. [software]. [přístup 20. srpna 2018]. Dostupné z: https://github.com/hdodenhof/CircleImageView

[53] Yalantis. *uCrop - Image Cropping Library for Android*. Verze 2.2.2. [software]. [přístup 28. září 2018]. Dostupné z: https://github.com/Yalantis/uCrop

[54] HAI, Zhang. *Material Progress Bar*. Verze 1.4.2. [software]. [přístup 28. září 2018]. Dostupné z: https://github.com/Yalantis/uCrop

[55] FACEBOOK, INC. *Stetho*. Verze 1.5.0. [software]. [přístup 15. září 2018]. Dostupné z: http://facebook.github.io/stetho/

[56] JUNGINGER, Markus. *EventBus*. Verze 3.1.1. [software]. [přístup 27. srpna 2018]. Dostupné z: https://github.com/greenrobot/EventBus

[57] Google, Inc. *Google Maps SDK*. Verze 15.0.1. [software]. [přístup 2. listopadu 2018]. Dostupné z: https://developers.google.com/android/reference/com/google/android/gms/maps/package-summary

[58] Google, Inc. *Play Services*. Verze 15.0.1. [software]. [přístup 2. listopadu 2018]. Dostupné z: https://developers.google.com/android/guides/setup

[59] Kunzisoft. *Android SwitchDateTime Picker*. Verze 2.0. [software]. [přístup 27. srpna 2018]. Dostupné z: https://github.com/Kunzisoft/Android-SwitchDateTimePicker

[60] Thoughtbot. *Factory Bot*. Verze 4.8.2. [software]. [přístup 29. února 2018]. Dostupné z: https://github.com/thoughtbot/factory_bot_rails

Příloha A

Seznam použitých zkratek

UX User Experience

UI User Interface

OS Operation System

API Application Programming Interface

JVM Java Virtual Machine

IDE Integrated Development Environment

MVC Model View Controller

MVP Model View Presenter

MVVM Model View Viewmodel

FP Funkční požadavek

NFP Nefunkční požadavek

PU Případ užití

UML Unified Modeling Language

FAB Floating Action Button

Lo-fi prototype Low fidelity prototype

Hi-fi prototype High fidelity prototype

REST Representational State Transfer

HTTP Hypertext Transfer Protocol

OOP Object Oriented Programming

URI Uniform Resource Identifier

JSON JavaScript Object Notation

XML Extensible Markup Language

CRUD Create, Read, Update, Delete

URL Uniform Resource Locator

SDK Software Development Kit

VPS Virtual Private Server

ANR Application Not Responding

GDPR General Data Protection Regulation

Příloha B

Dotazník

1. Používáte nějakou sociální síť na svém mobilním zařízení?
   - Ano
   - Ne

   Poznámka: pokud vaše odpověď je záporná, pokračujte, prosím, otázkou číslo 5.
2. Jak často používáte sociální sítě na svém mobilním zařízení?
   - Několikrát denně
   - Jednou denně
   - Jednou za několik dní
   - Jednou týdně
   - Jednou měsíčně
   - Skoro nikdy
3. Jak často sdílíte fotografie svého domácího mazlíčka na sociálních sítích?
   - Několikrát denně
   - Jednou denně
   - Jednou za několik dní
   - Jednou týdně
   - Jednou měsíčně
   - Skoro nikdy
   - Nemám domácího mazlíčka

4. Líbí se vám prohlížet fotografie jiných domácích mazlíčků?
    - Ano
    - Ne
    - Občas
    - Je mi to jedno
5. Jste majitelem psa?
    - Ano
    - Ne

    Poznámka: pokud vaše odpověď je záporná, pokračujte, prosím, otázkou číslo 9.
6. Jak často chodíte na delší procházku se svým psem?
    - Několikrát denně
    - Jednou denně
    - Jednou za několik dní
    - Jednou týdně
    - Jednou měsíčně
    - Skoro nikdy
7. Líbí se vám myšlenka sledovat trasu procházky se psem na mapě?
    - Ano
    - Ne
    - Nevím
    - Je mi to jedno
8. Co si myslíte ohledně společného venčení psů s jinými majiteli?
    - Dobrá myšlenka, často se nudím, když venčím psa
    - Mohlo by to být dobré
    - Chci venčit svého psa sám
    - Ne, špatná myšlenka
    - Nevím

9. Pokud by existovala sociální síť pro majitele psů, co by měla obsahovat?

    - Sdílení fotografií psa
    - Sdílení událostí ze života psa
    - Prohlížení fotografií nebo událostí psů ostatních uživatelů
    - Komentáře
    - Přidání do oblíbených
    - Vyhledávání psů podle plemen
    - Sledování domácích mazlíčků jiných uživatelů
    - Soukromé zprávy
    - Vyhledávání partnera pro psa
    - Sledování trasy procházek na mapě
    - Prohlížení historie procházek se psem
    - Hledání kamaráda pro společné venčení psů
    - Doporučení lokalit pro venčení psů

Příloha C

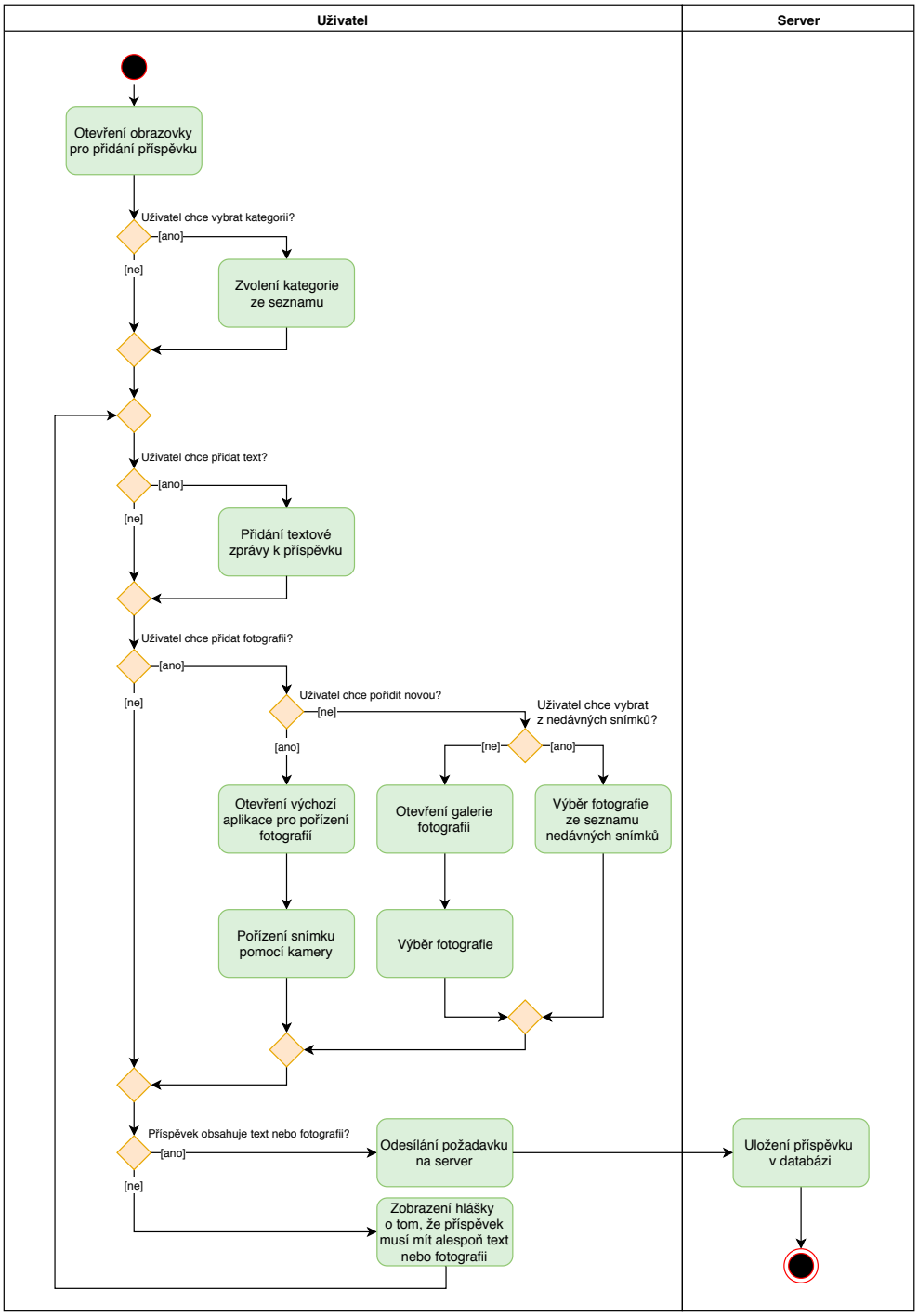
Pokrytí případů užití

Tabulka C.1: Pokrytí funkčních požadavků

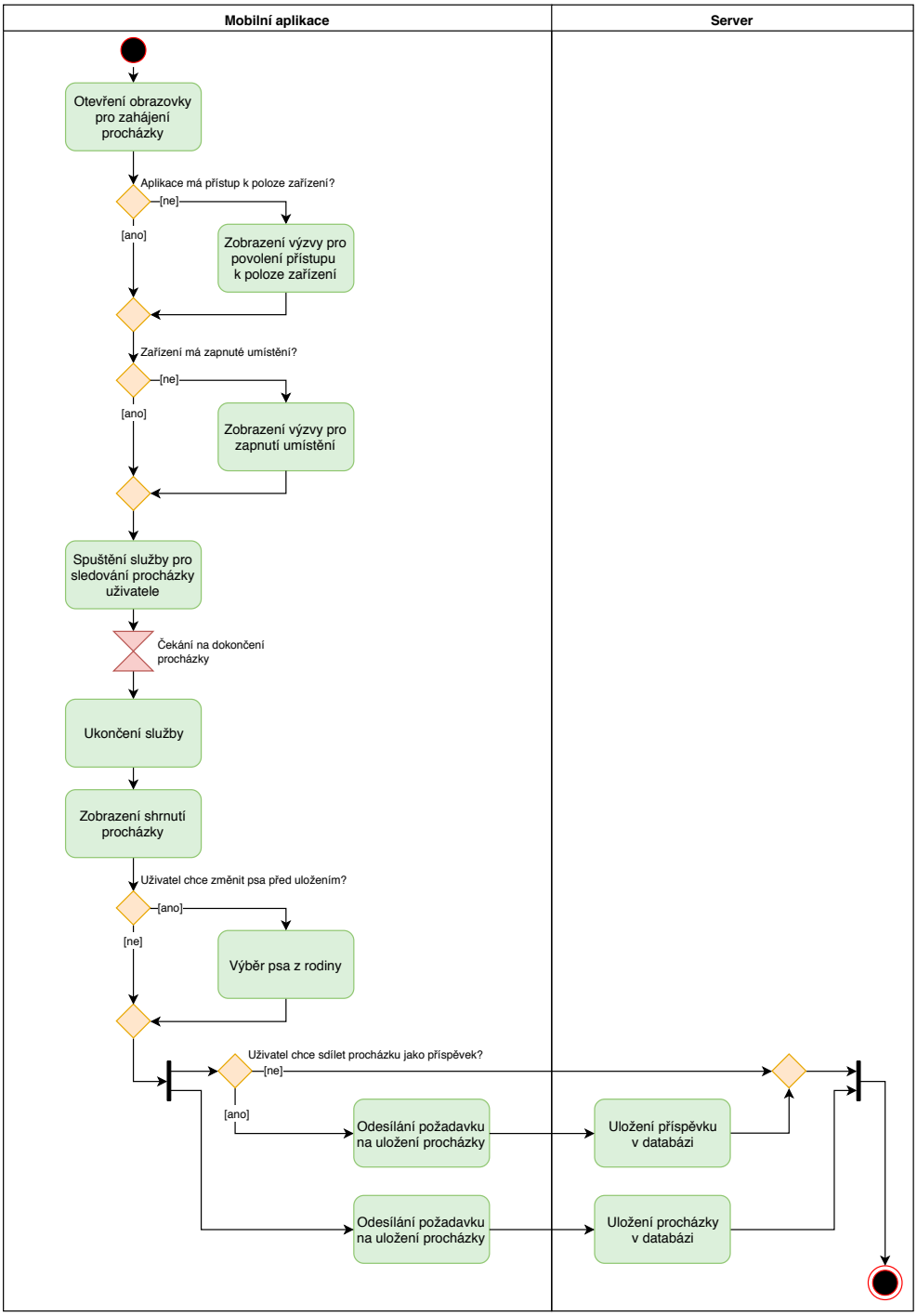
	FP1	FP2	FP3	FP4	FP5	FP6	FP7	FP8	FP9	FP10
PU1	✓									
PU2	✓									
PU3	✓									
PU4	✓	✓								
PU5	✓	✓								
PU6	✓	✓								
PU7		✓								
PU8			✓							
PU9			✓							
PU10			✓							
PU11		✓	✓							
PU12			✓			✓				
PU13			✓			✓				
PU14			✓			✓				
PU15			✓			✓				
PU16			✓			✓				
PU17				✓		✓				
PU18				✓						
PU19						✓				
PU20						✓				
PU21						✓				
PU22					✓					
PU23					✓					
PU24					✓					
PU25					✓					
PU26							✓			
PU27								✓		
PU28								✓	✓	
PU29									✓	
PU30								✓	✓	
PU31										✓
PU32										✓
PU33										✓
PU34										✓
PU35										✓

Příloha D

Diagramy aktivit



Obrázek D.1: Diagram aktivit pro přidání příspěvku



Obrázek D.2: Diagram aktivit pro přidání procházky

Příloha **E**

Seznam úkolů aplikace

- Přihlášení a registrace
  - Zadání přihlašovacích údajů [O; 2]
  - Přihlášení existujícího uživatele [O; 3]
  - Odhlášení uživatele [O; 2]
  - Zobrazení informací o špatných přihlašovacích údajích [Z; 1]
  - Zadání registračních údajů [O; 1]
  - Registrace uživatele [O; 2]
  - Zobrazení informací o existující elektronické adrese [Z; 1]
- Přidání psa
  - Zadání jména psa [O; 3]
  - Zobrazení výběru pohlaví psa [Z; 3]
  - Zadání pohlaví psa [O; 3]
  - Zobrazení seznamu dostupných plemen psa [Z; 4]
  - Zadání plemena psa [O; 3]
  - Přidání profilového obrázku psa [O; 4]
  - Otevření galerie pro výběr obrázku [O; 4]
  - Zvolení přesného data narození [O; 2]
  - Zvolení přibližného data narození [O; 3]
  - Zadání informací o psovi [O; 2]
- Stránka psa
  - Zobrazení profilového obrázku [Z; 5]
  - Zobrazení informací o psovi [Z; 4]

  - Zobrazení počtu sledujících [Z; 5]
  - Zobrazení seznamu sledujících [Z; 3]
  - Zobrazení počtu sledovaných [Z; 5]
  - Zobrazení seznamu sledovaných [Z; 3]
  - Přepínání mezi stránkami psů uživatele [O; 5]
  - Zobrazení příspěvků psa [Z; 5]
    Stránka psa daného uživatele navíc obsahuje
  - Přidání příspěvku [O; 5]
  - Úprava informací o psovi [O; 2]
  - Změna profilového obrázku [O; 3]
  - Mazání stránky psa [O; 1]
    Stránka cizího psa navíc obsahuje
  - Začátek sledování psa [O; 4]
  - Ukončení sledování psa [O; 2]

- Přidání příspěvku
  - Zobrazení seznamu podporovaných kategorií [Z; 4]
  - Výběr kategorie [O; 3]
  - Zobrazení profilového obrázku psa, na stránce kterého příspěvek bude zveřejněn [Z; 4]
  - Přidání popisu příspěvku [O; 4]
  - Výběr obrázku z galerie [O; 5]
  - Pořízení obrázku pomocí kamery [O; 3]
  - Odeslání příspěvku na server [O; 5]

- Příspěvek
  - Zobrazení informací o vlastníkovi příspěvku (jméno a profilový obrázek psa) [Z; 5]
  - Zobrazení kategorie (pokud byla přidána) [Z; 4]
  - Zobrazení popisu příspěvku (pokud byl přidán) [Z; 4]
  - Zobrazení obrázku (pokud byl přidán) [Z; 5]
  - Zobrazení datumu přidání [Z; 5]
  - Zobrazení počtu přidání do oblíbených [Z; 5]
  - Zobrazení počtu přidaných komentářů [Z; 5]
  - Smazání příspěvku (pokud uživatel je vlastníkem) [O; 2]

  - Přechod na vlastníka příspěvku [O; 4]
  - Přechod na seznam komentářů [O; 4]
  - Přidání příspěvku do oblíbených [O; 4]
  - Odebrání příspěvku z oblíbených [O; 2]
  - Zobrazení nových příspěvků ze sledovaných stránek [Z; 5]
- Komentář
  - Zobrazení seznamu komentářů k příspěvku [Z; 4]
  - Zobrazení jména a profilového obrázku psa, který přidal příspěvek [Z; 5]
  - Zobrazení textu komentáře [Z; 5]
  - Zobrazení datumu přidání komentáře [Z; 5]
  - Zobrazení políčka pro napsání vlastního komentáře [Z; 5]
  - Přidání vlastního komentáře [O; 3]
  - Přechod na vlastníka komentáře [O; 4]
  - Smazání komentáře, pokud je vlastníkem [O; 1]
- Objevení
  - Zobrazení seznamu všech psů [Z; 3]
  - Hledání psa podle jména [O; 4]
  - Začátek sledování psa [O; 4]
  - Ukončení sledování psa [O; 2]
  - Vyhledávání psů podle plemen [O; 3]
  - Zobrazení fotografií psů podle plemen [Z; 4]
  - Zobrazení všech psů z plemena [Z; 4]
- Seznam novinek
  - Zobrazení nových komentářů [Z; 3]
  - Zobrazení nových přidání k oblíbeným [Z; 3]
  - Zobrazení nových sledujících [Z; 3]
- Procházky
  - Přidání nové procházky [O; 5]
  - Začátek trekování procházky [O; 5]
  - Konec trekování procházky [O; 5]
  - Zobrazení informací o procházce [Z; 5]

  - Zobrazení informací o psovi [Z; 4]
  - Uložení procházky [O; 4]
  - Sdílení procházky jako příspěvku [O; 3]
  - Zobrazení již uložených procházek [Z; 5]
  - Smazání procházky [O; 1]

- Nabídka na společné venčení psů
  - Vytvoření nabídky na společné venčení psů [O; 3]
  - Smazání nabídky na společné venčení psů [O; 1]
  - Zvolení zeměpisné polohy pro setkání [O; 3]
  - Zvolení data pro setkání [O; 3]
  - Přidání popisu budoucí procházky [O; 2]
  - Zobrazení vlastních nabídek na společné venčení psů [Z; 3]
  - Zobrazení detailu nabídky na společné venčení psů [Z; 3]
  - Zobrazení cizích nabídek na společné venčení psů [Z; 3]
  - Filtrování seznamu cizích nabídek na společné venčení psů podle zadané lokality [O; 3]

Příloha F

Vstupní dotazník

1. Jaká je Vaše věková skupina?
   a) 10-18
   b) 18-26 [**Respondent A, Respondent D**]
   c) 26-34 [**Respondent B, Respondent C**]
   d) 35-42
2. Jak velkého máte psa?
   a) Velmi malý
   b) Malý [**Respondent A**]
   c) Střední
   d) Velký [**Respondent B**]
   e) Nemám psa [**Respondent C, Respondent D**]
3. Jak často chodíte se svým psem na delší procházky například do parku?
   a) Několikrát denně
   b) Jednou denně [**Respondent B**]
   c) Několikrát týdně
   d) Jednou týdně [**Respondent A**]
   e) Jednou měsíčně
   f) Nemám psa [**Respondent C, Respondent D**]
4. Kolik hodin denně průměrně trávíte na internetu?
   a) Méně než jednu hodinu
   b) 1–2 hodiny

c) 3–4 hodiny [**Respondent B, Respondent C, Respondent D**]

d) 4–6 hodin [**Respondent A**]

e) Více než 6 hodin

5. Používáte nějakou sociální síť na mobilním zařízení? Kolik času trávíte v sociálních sítích?

a) Nepoužívám [**Respondent B**]

b) Méně než jednu hodinu

c) 1–2 hodiny [**Respondent C, Respondent D**]

d) 3–4 hodiny [**Respondent A**]

e) 4–6 hodin

f) Více než 6 hodin

6. Jaká je Vaše zkušenost s operačním systémem Android?

a) Nepoužívám vůbec

b) Používám občas pro volání a posílání zpráv

c) Používám mnoho aplikací a umím je ovládat na dostatečně dobré úrovni [**Respondent A, Respondent D**]

d) Jsem expert [**Respondent B, Respondent C**]

Příloha G

Výstupní dotazník

1. Byl pro Vás průběh registrace nového psa pochopitelný a pohodlný?
   a) Rozhodně ano [**Respondent A**]
   b) Spíše ano [**Respondent B, Respondent C, Respondent D**]
   c) Nevím
   d) Spíše ne
   e) Rozhodně ne
2. Jaké další informace byste rádi viděli na profilové stránce psa?
   a) Váha
   b) Výška
   c) Oblíbené jídlo [**Respondent B**]
   d) Město, kde bydlí [**Respondent A, Respondent D**]
   e) Vyznamenání, ocenění
   f) Jiné [**Respondent C**]
3. Líbila se Vám možnost sledování trasy procházky se psem? Chtěli byste něco na ní změnit?
   a) Rozhodně ano [**Respondent A, Respondent B, Respondent C, Respondent D**]
   b) Spíše ano
   c) Nevím
   d) Spíše ne
   e) Rozhodně ne

   Komentář respondenta B: notifikace, která se zobrazuje v průběhu sledování procházky neobsahuje žádné užitečné informace o procházce, kromě slova „Walking. . . “.

4. Jak se Vám líbí vzhled aplikace? (ohodnoťte prosím od 1 do 10)

   [Respondent A] 10

   [Respondent B] 9

   [Respondent C] 10

   [Respondent D] 9

5. Jaký je Váš celkový dojem z používání aplikace? (ohodnoťte prosím od 1 do 10)

   [Respondent A] 9

   [Respondent B] 8

   [Respondent C] 7

   [Respondent D] 8

6. Napadá Vás nějaké vylepšení aplikace, které by zlepšilo Váš dojem z používání aplikace?

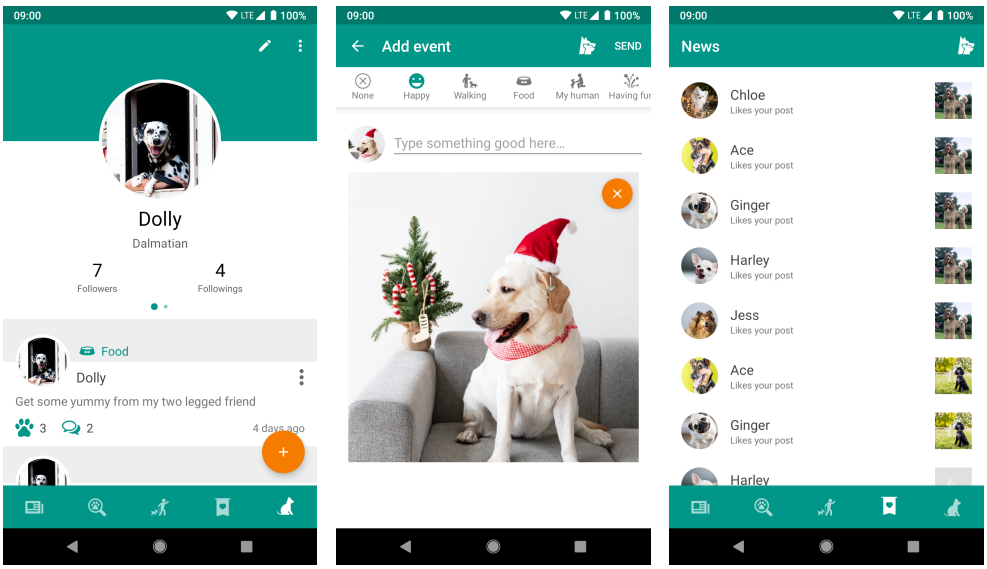
   Komentář respondenta A: na každé obrazovce jsem viděla stejnou ikonu, která vyvolává dialog s profilovým obrázkem mého psa. K čemu to je?

   Komentář respondenta B: při registraci nového psa není vidět, kolik kroků ještě zbývá.

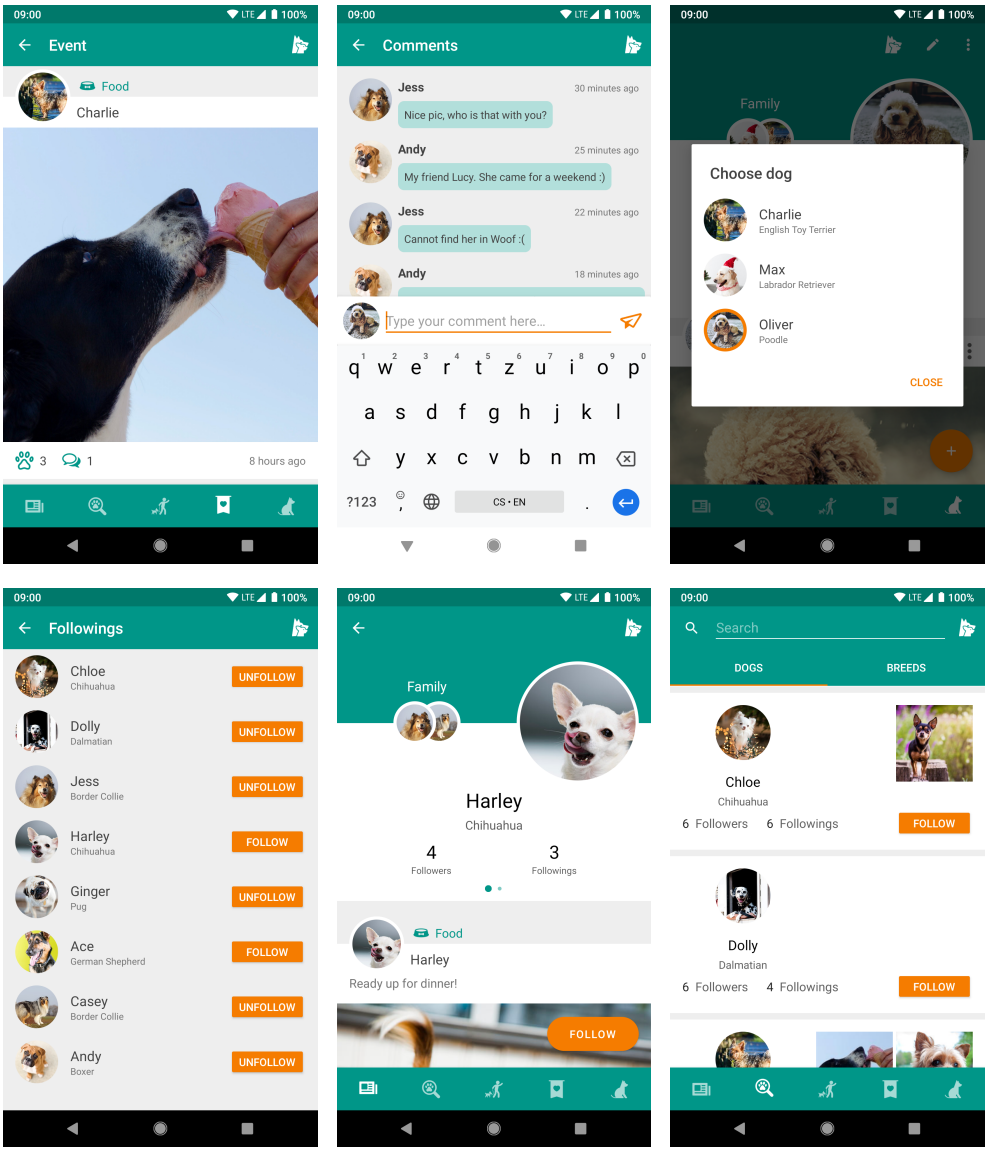
   Komentář respondenta C: po registraci přidat automatické přesměrování na stránku psa.

Příloha H

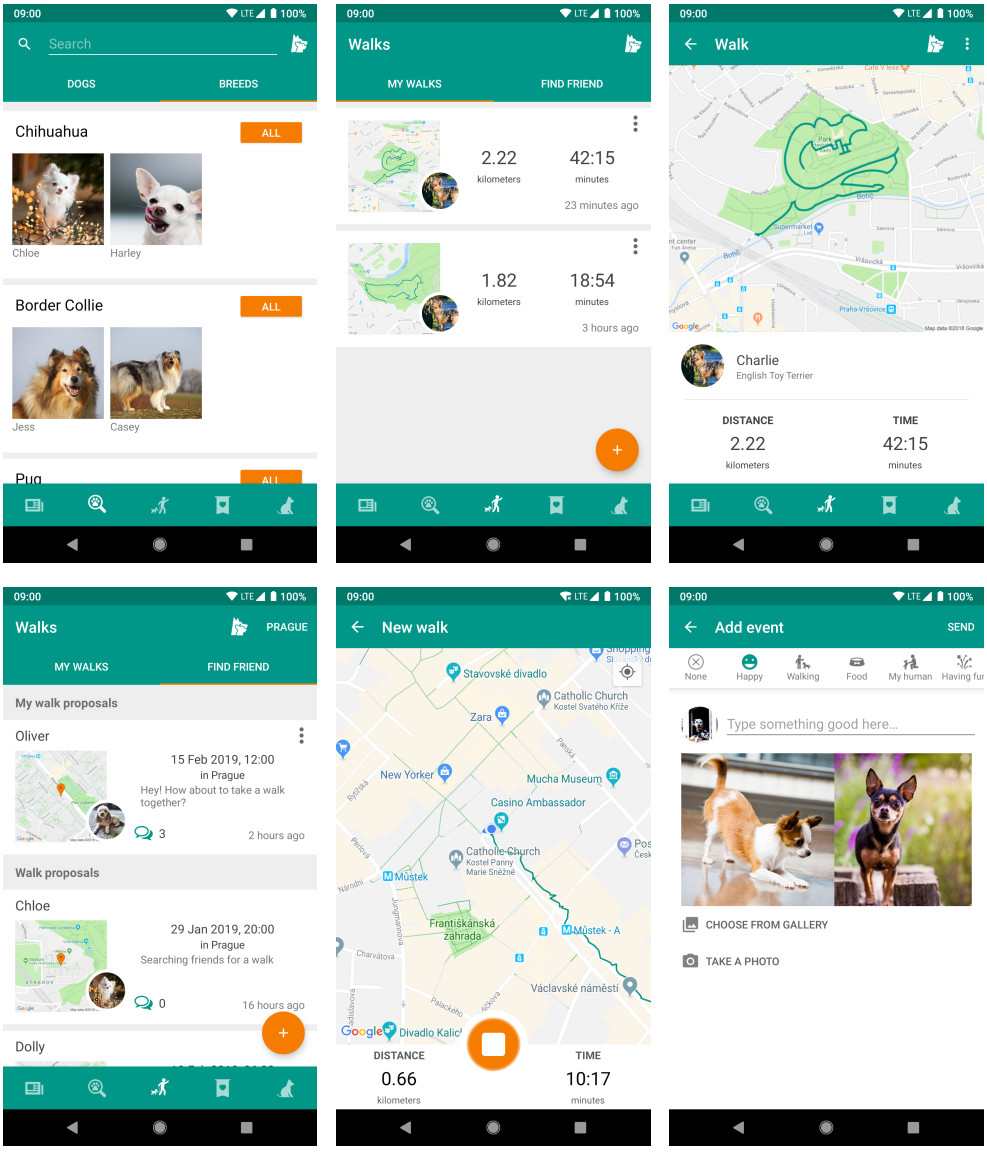
Ukázka výsledné aplikace



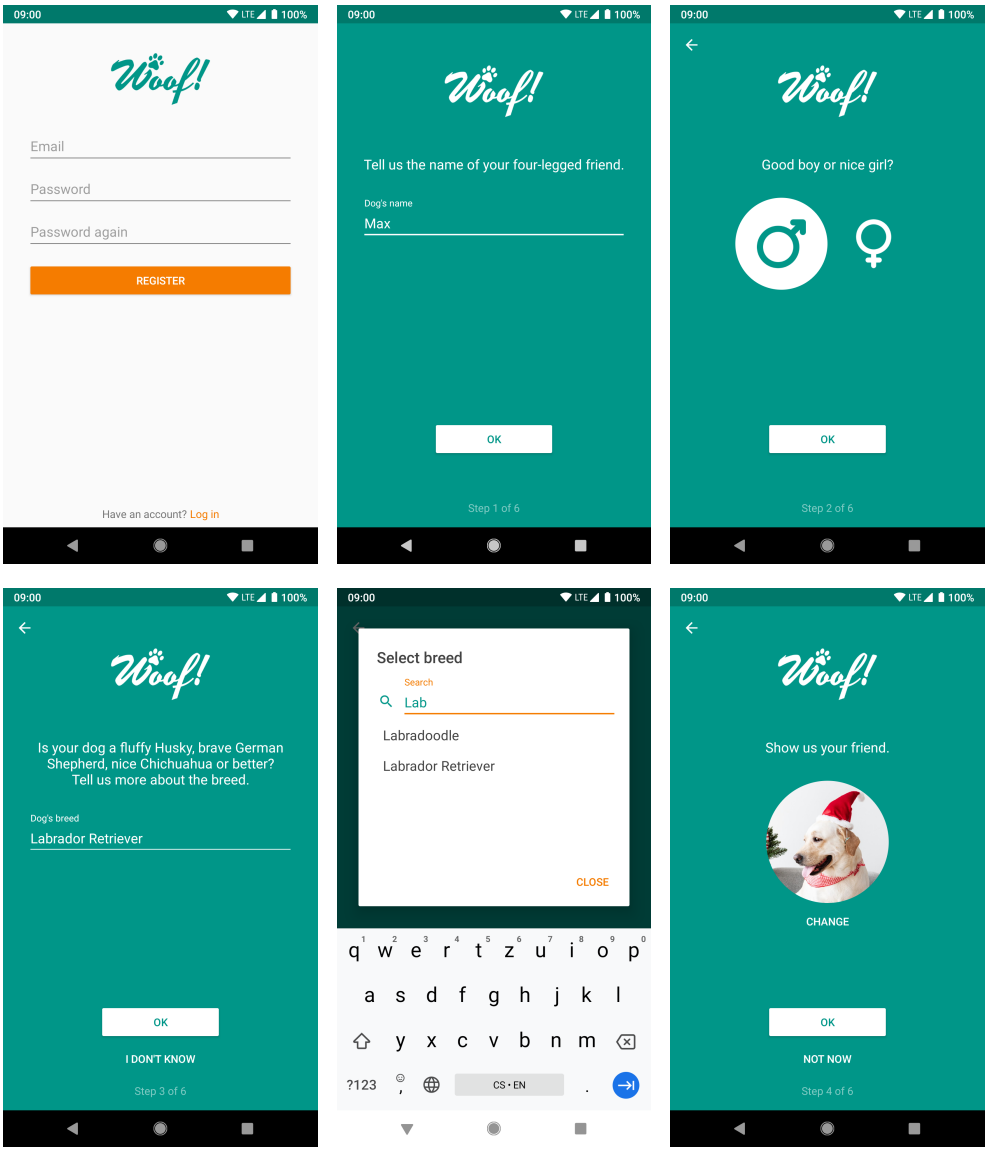
Obrázek H.1: První část ukázky výsledné aplikace



Obrázek H.2: Druhá část ukázky výsledné aplikace



Obrázek H.3: Třetí část ukázky výsledné aplikace



Obrázek H.4: Čtvrtá část ukázky výsledné aplikace: registrace

Příloha I

Obsah přiloženého CD

- `woof.apk` ..... instalační soubor
- `src`
  - `impl` ..... zdrojové kódy Android aplikace
  - `thesis` ..... zdrojová forma práce
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
- `attach` ..... přiložené soubory
  - `swagger` ..... dokumentace API
  - `prototypes` ..... hi-fi prototypy
  - `screenshots` ..... ukázka výsledné aplikace