Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Mobilní aplikace pro majitele psů

Gabriela Pachlová

Bakalářská práce
2022

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Gabriela Pachlová**
Osobní číslo:	**I19130**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Téma práce:	**Mobilní aplikace pro majitele psů**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování

Naprogramujte Android aplikaci zaměřenou na body zájmů pro majitele psů. V teoretické části práce proveďte rešerši podobných aplikací. V praktické části práce vytvořte funkční mobilní aplikaci. Aplikace s využitím Google Maps bude zobrazovat například místa, kde se může pes vykoupat; restaurace, které umožňují vstup i psům; místa, kde můžeme psa pustit na volno a podobně. Jednotlivá místa budou vkládána do lokální databáze s GPS souřadnicemi, s popisem a s případnými fotografiemi.

Rozsah pracovní zprávy: **min 30. stran**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

LACKO, Ľuboslav. Mistrovství – Android. Přeložil Martin HERODEK. Brno: Computer Press, 2017. Mistrovství. ISBN 978-80-251-4875-4.

SPÄTH, Peter. Pro Android with Kotlin: Developing Modern Mobile Apps. Berlin: Springer, 2018. ISBN 978-1484238196.

LACKO, Ľuboslav. Vývoj aplikací pro Android. Brno: Computer Press, 2015. ISBN 978-80-251-4347-6.

Vedoucí bakalářské práce: **Ing. Miroslav Dvořák, Dipl.tech.**
Katedra informačních technologií

Datum zadání bakalářské práce: **17. prosince 2021**
Termín odevzdání bakalářské práce: **13. května 2022**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Jan Panuš, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 28. února 2022

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 14. 8. 2022

Gabriela Pachlová

ANOTACE

Bakalářská práce je zaměřena na evidenci bodů zájmu pro majitele psů pomocí mobilní aplikace. Teoretická část se zabývá porovnáním a analýzou podobných již existujících aplikací. V praktické části je vytvořena mobilní aplikace pro platformu Android, která v lokální databázi eviduje a pomocí Mapboxu zobrazuje body zájmu. Jedná se například o restaurace, které dovolují vstup se psy, místa, kde se může pejsek vykoupat nebo místa, kde je možné pustit pejska z vodítka. O jednotlivých místech jsou evidovány souřadnice, popis a případně fotografie.

KLÍČOVÁ SLOVA

Kotlin, Android, Google mapy, mobilní aplikace, chytrý telefon, Android Studio, Room databáze, lokální databáze, lokace, body zájmu, POI

TITLE

Mobile Application for Dog Owners

ANNOTATION

Bachelor thesis is focused on records of points of interest for dog owners using a mobile application. The theoretical part deals with the comparison and analysis of similar already existing applications. In the practical part a mobile application for the Android platform is created, which records points of interest in a local database and uses Mapbox to display them. Points of interest are, for example, restaurants, that allow entry with dogs, places, where the dog can swim or places, where the dog can run free without a leash. coordinates, descriptions and possibly photos are registered about individual places.

KEYWORDS

Kotlin, Android, Google maps, mobile application, smartphone, Android Studio, database Room, local database, location, points of interest, POI

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM ZDROJOVÝCH KÓDU

SEZNAM ZKRATEK

DAO	Database Access Object
IDE	Integrated Development Environment
JVM	Java Virtual Machine
SDK	Software Development Kit
SQL	Structured Query Language
URI	Uniform Resource Identifier
XML	Extensible Markup Language

ÚVOD

V dnešní době jsou chytré telefony nedílnou součástí našeho života. Počet jejich uživatelů se každý rok zvyšuje. V České republice v roce 2021 používalo chytrý telefon 76,6 % osob nad 16 let, zatímco v roce 2018 pouze 63,1 % [1]. Každý chytrý telefon využívá operační systém. Na českém trhu jsou dva hlavní operační systémy. Operační systém Android s procentuálním zastoupením 79,53 % a operační systém iOS od společnosti Apple se zastoupením 20,27 %, uvedeno k listopadu 2021 [2].

Cílem této bakalářské práce je vytvořit mobilní aplikaci právě pro platformu Android. Mobilní aplikace je určená pro majitele psů, kteří si chtějí v aplikaci evidovat místa, které mohou se svým čtyřnohým mazlíčkem navštívit. Jedná se například o restaurace, které s nimi umožňují vstup, ale také místa v přírodě, která jsou vhodná k procházce. Aplikace všechny místa zobrazuje na mapě a umožní uživateli k těmto místům napsat vlastní popisek a vložit fotografii.

Hlavním důvodem výběru tohoto tématu byla motivace naučit se něco více o vývoji mobilních aplikací, a protože jsem vlastníkem mobilního zařízení s operačním systémem Android, byl i výběr platformy jasný. Moje láska ke psům zapříčinila i výběr určení aplikace, tedy že se jedná o aplikaci určenou pro majitele psů. Nedostatečné množství aplikací zaznamenávající body zájmu pro majitele psů na trhu, ujasnil i její přesné zaměření.

Obsahem bakalářské práce je nejprve popis již existujících aplikací, které evidují body zájmu a jsou určeny pro majitele psů. Dále jsou rozebrány technologie použité při vývoji a obecný popis aplikace: co umožňuje a jaké jsou její vizuální části. Nakonec popis vývoje jednotlivých částí aplikace a její otestování.

1 SROVNÁNÍ EXISTUJÍCÍCH APLIKACÍ

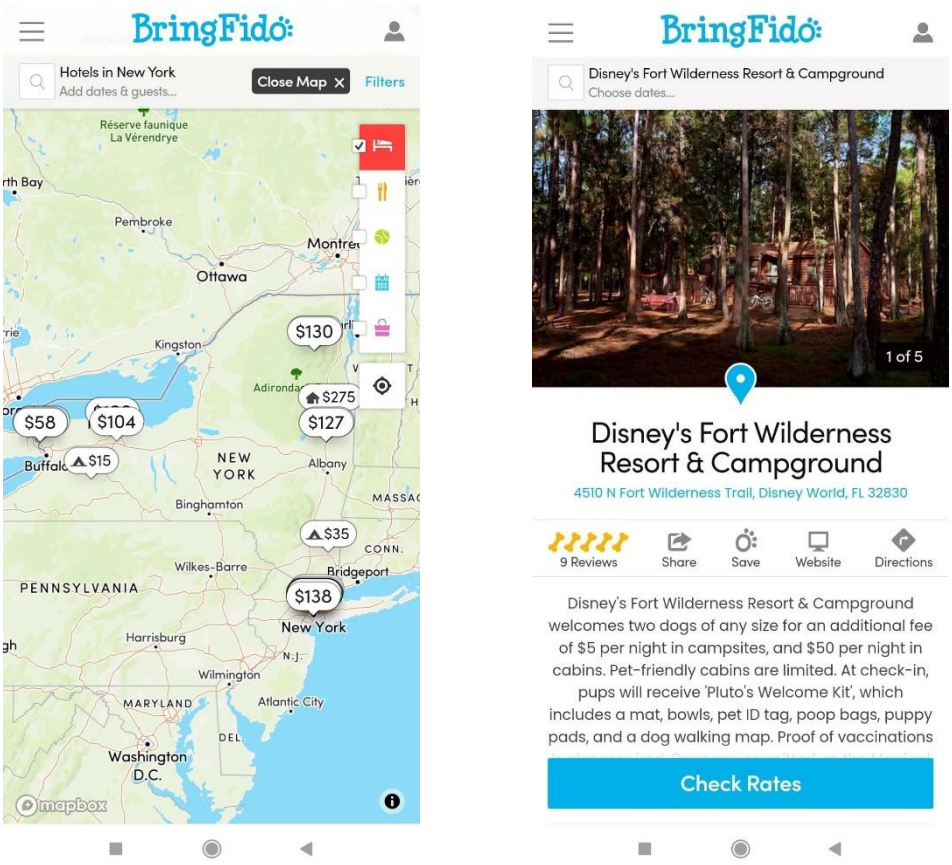
V následující kapitole jsou rozebrány čtyři již existující aplikace. Jsou vybrány takové, které zobrazují body zájmu na mapě a jsou určeny pro majitele psů. Každá je krátce popsána a, pokud je to možné, tak i následně porovnána s mobilní aplikací vyvíjenou v praktické části této práce. Během rešerše bylo zjištěno, že aplikace, které jsou zaměřeny pro majitele psů, jsou určeny spíše pro výcvik a monitorování zdraví psů.

1.1 BringFido

Mobilní aplikace pro platformu Android i iOS, která je dostupná také ve webové podobě, BringFido, je zaměřena především na cestování se psy. Zaujme jejich majitele, kteří s nimi rádi jezdí na dovolené a výlety. Aplikace vznikla jako reakce na nedostatek informací ohledně ubytování se zvířaty, se kterými se majitelé potýkali, pokud chtěli se svými domácími mazlíčky vycestovat. Hotely často neměly na svých webových stránkách uvedeno, zda pobyt dovolují, popřípadě jaká pravidla se při pobytu musí dodržovat [3].

Hlavní funkcí aplikace je vyhledávání ubytování a jejich následná rezervace. Kromě toho je možné procházet restaurace, psí parky, veterináře a mnoho dalších míst, kde jsou psi vítáni. Uživatel zadá název destinace a vybere kategorii, která ho zajímá. Výsledky jsou zobrazeny na mapě, nebo jako dlaždicový seznam se základním popisem. Zaregistrovaní uživatelé mohou hodnotit místa a také přidávat nová místa, psát recenze a sdílet fotografie z výletů. V aplikaci lze najít blog o cestování se zvířaty, který obsahuje články o oblíbených hotelech, restauracích a aktivitách po celém světě [4].

Velkým plusem aplikace je její dostupnost pro platformu Android i iOS a možností jí využít i ve webové podobě. Spíše se ale soustředí na možnost rezervace ubytování při cestování se psem, na rozdíl od vyvíjené aplikace, která má za cíl umožnit uživateli evidovat si body zájmu a jednoduše a rychle si tyto body prohlížet na mapě.



Obrázek 1: Aplikace BringFido

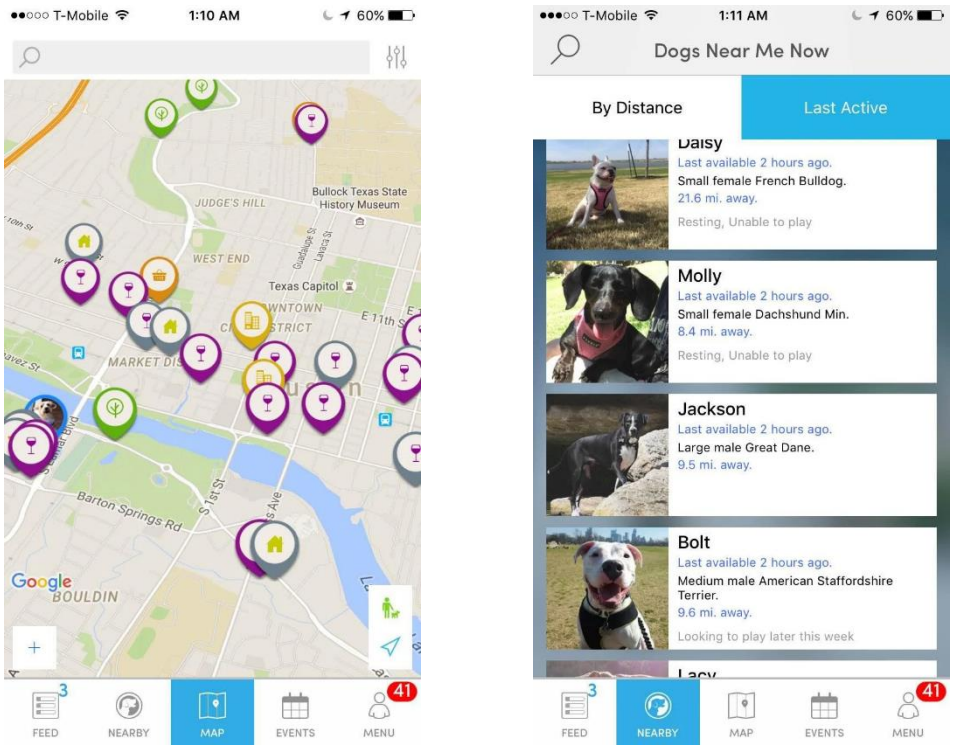
1.2 BarkHappy

Cílem mobilní aplikace BarkHappy je dopřát psům více aktivního a sociálního života. Za vznikem aplikace stál nedostatek informací o místech, která jsou pro ně vhodná. Jeden z hlavních záměrů aplikace je umožnit jim seznamovat se s ostatními psy a zpříjemnit majitelům život se svým mazlíčkem. Zajímavostí je název aplikace, který vyjadřuje pocit obrovského štěstí a nadšení, který vede k nekontrolovatelnému štěkání, například pokud se pes velmi dobře bavil v parku [5].

Aplikace je dostupná pro platformu Android i iOS pouze ve Spojených státech amerických. Umožňuje vyhledávat restaurace, hotely, parky a další místa, která jsou vhodná pro psy. Kromě toho se lze podívat na to jaká pravidla a opatření na těchto místech platí. Cílem aplikace je zpestřit sociální život psů, proto nabízí funkci vyhledávání ostatních psů v okolí, s jejichž majiteli se uživatel může spojit. Majitelé se v ní mohou podívat na nadcházející akce vhodné pro psy a také pořádat své vlastní. Pokud se uživateli mazlíček ztratí, lze vytvořit upozornění o ztrátě s jeho fotografií, díky tomuto upozornění aplikace informuje uživatele ve stejné oblasti

a zvýší tím šanci na jeho nalezení. Stejně jako upozornění o ztrátě, lze vytvářet i upozornění o nálezu psa [6].

Protože je tato aplikace dostupná pouze ve Spojených státech amerických nebylo možné ji v České republice vyzkoušet.



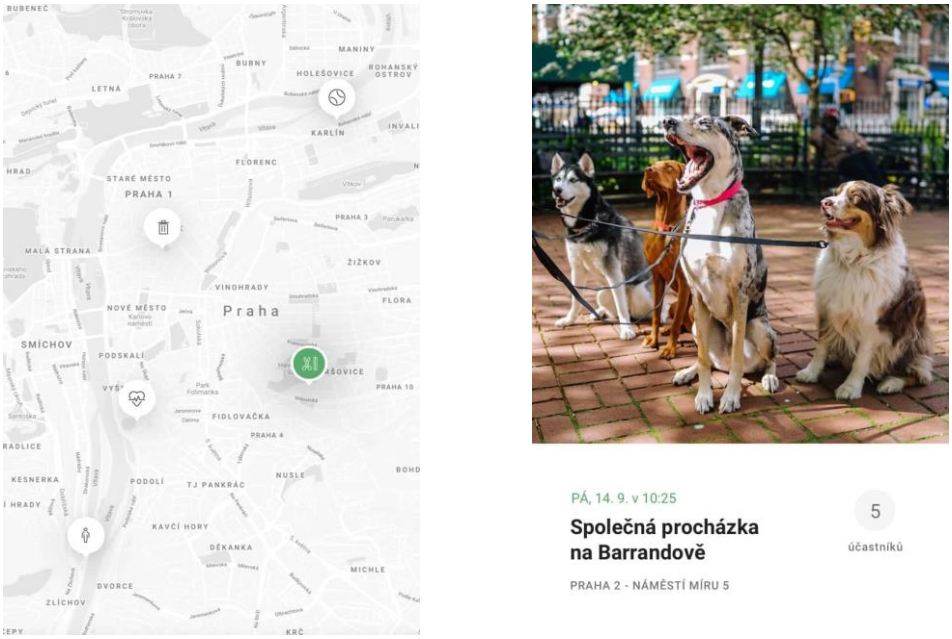
Obrázek 2: Aplikace BarkHappy [7]

1.3 Fiddo

Česká mobilní aplikace Fiddo slouží především ke zkvalitnění psího života. Momentálně není v provozu, protože ukončila svoji činnost k 31.3.2022, ale plánuje svůj návrat na podzim roku 2022.

V aplikaci mají být dostupné články a videa, informace o péči, tréninku, zdraví, ale i rozhovory s osobnostmi z psího světa. Na mapě by se měla zobrazovat místa jako jsou veterinární ordinace, služby pro psy nebo třeba odpadkové koše. Aplikace umožní měření vzdálenosti procházek a také ulehčí seznamování majitelů psů, kteří si spolu budou moci vyjít na společnou vycházku. Další funkcí aplikace by měla být možnost přidávat akce, komentáře a fotografie svých mazlíčků a společně je sdílet s ostatními uživateli [8].

Kvůli ukončení jejího provozu, nebylo ani aplikaci Fiddo možné, během psaní této práce, vyzkoušet.

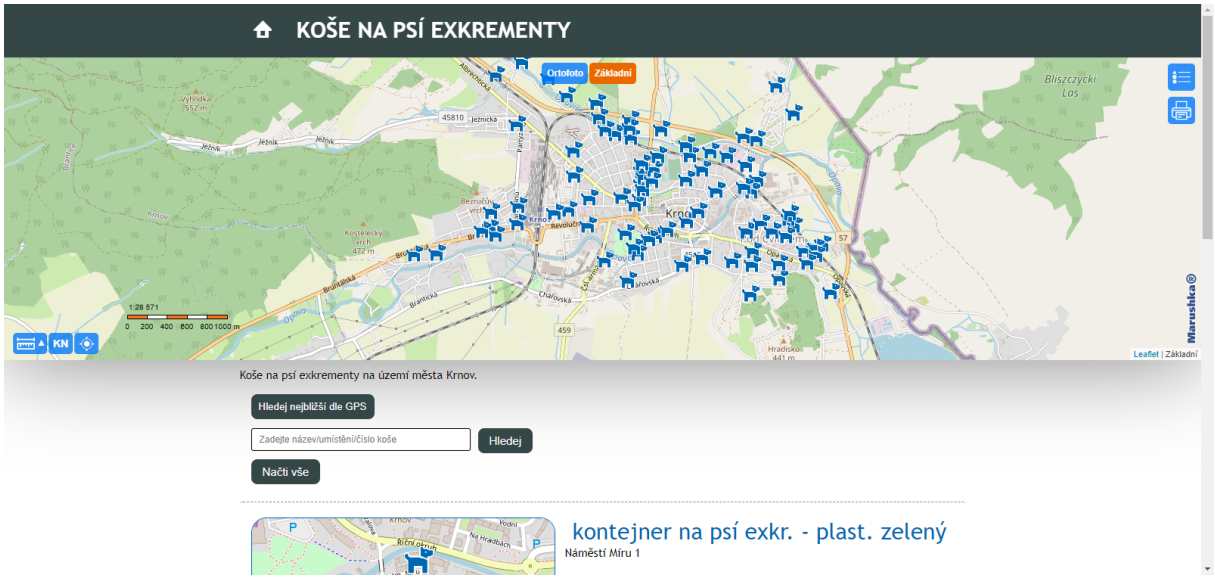


Obrázek 3: Aplikace Fiddo [8]

1.4 Psí koše Krnov

Poslední aplikací je webová aplikace českého města Krnov. Zobrazuje pouze odpadkové koše na mapě města pomocí modré ikonky psa. Po kliknutí na ikonku se zobrazí detailní informace o konkrétním odpadkovém koši, tedy jeho inventární číslo, název, umístění a poznámka. Aplikace umožňuje vyhledávat pomocí zadání názvu, umístění nebo čísla koše. Také umí podle aktuální polohy uživatele zobrazit nejbližší odpadkový koš [9].

Na rozdíl od vyvíjené aplikace se jedná o webovou nikoliv mobilní aplikaci. Aplikace Psí koše Krnov má jediný úkol, kterým je zobrazit občanům a návštěvníkům města pozice odpadkových košů ve městě. Neumožňuje tedy přidávat nové odpadkové koše ani nezobrazuje žádné jiné body zájmu, které by mohly být pro pejskaře zajímavé. Chybí také podpora náhledu fotografie daných odpadkových košů.



Obrázek 4: Aplikace Psí koše Krnov

2 POUŽITÉ TECHNOLOGIE

Následující kapitola obsahuje specifikace technologií, které byly použity při vývoji aplikace. V úvodu některých podkapitol jsou také zmíněny možné alternativy těchto technologií.

2.1 Programovací jazyky

Důležitým rozhodnutím při začátku vývoje nové aplikace je výběr programovacího jazyka. Protože aplikace je vyvíjena pro platformu Android, jednalo se především o výběr mezi programovacím jazykem Java a Kotlin. Nakonec byl zvolen programovací jazyk Kotlin, protože se jedná o preferovaný programovací jazyk pro tuto platformu [10]. Při tvorbě grafického prostředí aplikace je jako výchozí jazyk ve vývojovém prostředí Android Studio zvolen značkovací jazyk XML, který byl v mé aplikaci použit, právě díky podpoře vývojového prostředí. Posledním použitým jazykem v aplikaci je jazyk SQL, který je využit při práci s lokální databází.

2.1.1 Kotlin

V červenci roku 2011 společnost JetBrains, která vyvíjí například známé vývojové prostředí IntelliJ IDEA, představila svůj nový projekt Kotlin, který se v roce 2012 stal open-source. V roce 2017 se Kotlin oficiálně stává programovacím jazykem pro vývoj pro platformu Android [11]. Kotlin byl původně navržen jako náhrada za programovací jazyk Java při vývoji mobilních aplikacích pro platformu Android. Nakonec se v roce 2019, osm let po oznámení začátku jeho vývoje, stává preferovaným programovacím jazykem na této platformě [10].

Kotlin je open-source projekt dostupný zdarma, vyvíjený pod licencí Apache 2.0. Jeho zdrojový kód je volně dostupný na stránce GitHub a primárně je vyvíjen týmem vývojářů firmy JetBrains s přispěvateli ze společnosti Google. Mnoho mobilních aplikací je již napsáno s využitím tohoto programovacího jazyka. Jedná se například o streamovací službu Netflix, výukovou aplikaci Duolingo: Naučte se anglicky nebo mobilní aplikace sociálních sítí Twitter a Reddit [12].

Programovací jazyk Kotlin běží nad JVM (Java Virtual Machine). JVM poskytuje běhové prostředí, ve kterém lze spouštět bytecode jazyka Java, respektive lze spouštět bytecode jazyků postavených nad jazykem Java, například právě jazyk Kotlin [13]. Existují i jiné varianty, například Kotlin/Native technologie, která zkompiluje kód jazyka Kotlin na nativní binární soubory, které lze spouštět bez virtuálního stroje [14]. Při realizaci aplikace je využita varianta běžící nad JVM.

2.1.2 XML

Značkovací jazyk XML neboli eXtensible Markup Language je jeden z nejrozšířenějších textových formátů pro sdílení strukturovaných informací. K zápisu se používají takzvané tagy, které musí být vždy ukončené nebo označené jako prázdné. Díky tomu počítač dokáže zachytit běžné chyby jako například nesprávné vnoření tagů. Jazyk XML je koncipován tak, aby byl snadno čitelný a sebepopisný [15].

V praktické části je tento jazyk využit při návrhu grafického rozložení jednotlivých prvků aplikace. Kromě toho je také použit k uložení textových řetězců zobrazovaných uživateli do jednoho souboru, konkrétně soubor string.xml pro snazší úpravu a dohledání jednotlivých řetězců a soubor colors.xml k uchování barevných konstant. Tyto dva soubory jsou umístěny ve složce res (resources) v podsložce values.

2.1.3 SQL

Jazyk SQL neboli Structured Query Language je jazyk určený pro práci s relačními databázemi. Historie tohoto jazyka sahá až do roku 1970, kdy Dr. Edgar F. Ted poprvé popsal relační model pro databáze. O čtyři roky později se poprvé objevuje SQL a jeho první verze vychází v roce 1989 [16].

Relační databáze se skládá z tabulek, které jsou propojeny vzájemnými vztahy (relacemi). Tabulka obsahuje strukturovaná data a skládá se z řádků, kdy každý řádek má svůj unikátní identifikátor, a sloupců, které označují atributy. Atributem může být například jméno, telefonní číslo, město a podobně. Pomocí SQL můžeme přistupovat k datům, která jsou uložena v databázi, mazat a upravovat je nebo přidávat nová data. Jazyk SQL toho ale nabízí mnohem více. Umožňuje přidávat a mazat tabulky nebo rovnou celé databáze. Pomocí tohoto jazyka můžeme nastavovat oprávnění, využívat funkce, vytvářet procedury a mnoho dalšího [16].

Při vytváření mobilní aplikace v praktické části je tento jazyk využit pro přístup do lokální databáze Room.

2.2 Vývojové prostředí

Zvoleným programovacím jazykem k realizaci mobilní aplikace je jazyk Kotlin, proto se výběr vývojového prostředí zúžil na tři IDEs (Integrated Development Environments): IntelliJ IDEA, Android Studio a Eclipse, pro které společnost JetBrains oficiálně poskytuje doplněk pro podporu vývoje v Kotlinu. Kotlin doplněk pro Eclipse má omezenou podporu, a proto je přímo v jeho dokumentaci doporučeno použít jiné vývojové prostředí. Vybráno bylo vývojové

prostředí Android Studio, které je založené na IntelliJ IDEA a je označeno jako oficiální vývojové prostředí pro platformu Android [17].

2.2.1 Android Studio

Vývojové prostředí Android Studio je, jak již bylo zmíněno výše, oficiální IDE pro vývoj aplikací pro Android. Je založené na vývojovém prostředí IntelliJ IDEA a jeho cílem je zvýšit produktivitu programátora při tvorbě aplikací [17].

Umožňuje automatické formátování kódu, které je v nastavení specifikováno a toto nastavení je možné upravovat dle potřeby. Do automatického formátování patří jednotná konvence pro velikost tabulátoru a odsazení, psaní mezer, prázdných řádků, závorek a mnoho dalšího. Dále umožňuje aplikovat změny v kódu do spuštěné aplikace bez nutnosti restartování. Poskytuje automatické dokončování kódu ve třech úrovních: základní doplňovaní, které zobrazí návrhy proměnných, metod a podobně, chytré doplňování, které zobrazí možnosti na základě kontextu a doplnění příkazu automaticky přidáním chybějících závorek, … [18].

Android Studio podporuje různé verzovací systémy, příkladem může být Git, Subversion nebo Mercurial. Nástrojem pro sestavení projektu je Gradle, který je v Android Studiu zprostředkován pomocí doplňku Android plugin for Gradle. Je přímo integrovaný do vývojového prostředí a není potřeba jeho příkazy provolávat skrz příkazovou řádku [18].

2.3 Mapové podklady

Výběr mapového podkladu je neméně důležitou součástí vývoje této mobilní aplikace. Na trhu jich existuje nepřeberné množství, jedná se například o Google Maps, Mapbox, OpenStreetMap, TomTom, Leaflet a mnoho dalších [19]. Při vývoji jsem vybírala mezi platformou Mapbox a Google Maps.

Platforma Google Maps požaduje vyplnění údajů z platební karty, než Vám bude dovoleno mapy využívat. Na každý měsíc poté poskytne 200 dolarů k využívání map. V případě překročení tohoto limitu, ale společnost začne strhávat příslušný objem peněz z Vašeho bankovního účtu. Konkrétně se jedná o 7 dolarů za každých tisíc požadavků při využívání Maps SDK for Android. Kompletní ceník je uveden na stránkách Google Maps Platform [20]. Proto byla nakonec zvolena platforma Mapbox.

2.3.1 Mapbox Maps SDK for Android

Společnost Mapbox poskytuje knihovnu Maps SDK for Android pro vkládání map do mobilních aplikací. Zkratka SDK znamená Software Development Kit. SDK obecně poskytuje

sadu nástrojů, knihoven, dokumentaci, ukázky kódu a návody, které pomáhají vývojářům vytvářet aplikace na konkrétní platformě [21].

Mapbox na rozdíl od Google Maps nepožaduje vyplnění informací o platební kartě před začátkem používání map a poskytuje výhodnější cenovou nabídku. Omezeni jsou aktivní uživatelé, a ne počty požadavků na mapy. Měsíčně může aplikaci používat až 25 tisíc uživatelů zdarma, poté se platí čtyři dolary za každých dalších tisíc uživatelů, přičemž se cena za tisíc uživatelů snižuje u hranice 125 tisíc a 250 tisíc uživatelů za měsíc. Kompletní ceník je k dispozici na stránkách Mapboxu [22].

Kromě mapových podkladů pro Android poskytuje společnost Mapbox mapové podklady i pro platformu iOS a pro webové rozhraní. Také lze využít možnosti navigace, lokace nebo například využít takzvané playgroundy neboli hřiště, v kterých si obecně lze vyzkoušet některé vlastnosti map přímo na webových stránkách Mapboxu. V playgroundu je možné například vytvořit statické mapové obrázky, které lze následně zobrazit v aplikaci bez nutnosti použití knihovny, jedná se poté pouze o vloženou mapu bez jakýkoliv interaktivních nebo ovládacích prvků [23].

3 O APLIKACI

Cílem práce je vytvořit mobilní aplikaci vhodnou k označování bodů zájmu, které vhodně zobrazuje na mapě a umožňuje s nimi další manipulaci. Tyto body zájmu jsou určeny pro majitele psů.

Vyvíjená aplikace umí zobrazovat body zájmu na mapě a umožňuje jednoduchou filtraci podle čtyř kategorií: odpadkový koš, restaurace, místo k plavání a hřiště. O všech bodech na mapě je možné zobrazit detailní informace: název, popis, kategorii, souřadnice a volitelně fotografii. Zároveň je uživatel schopen přidat na mapu další bod a všechny body případně měnit nebo mazat. Fotografie mohou být vyfoceny fotoaparátem mobilního zařízení nebo mohou být vybrány z úložiště telefonu. Souřadnice jsou automaticky vyplněny podle polohy uživatele s možností další editace. Všechny vstupy jsou před uložením do databáze zvalidovány. Při zapnutí aplikace se mapa vykreslí v místě, kde se daný uživatel zrovna nachází. Pokud se při manipulaci s mapou uživatel dostane na jemu neznámé místo, má možnost využít návratového tlačítka, které ho vrátí na výchozí bod.

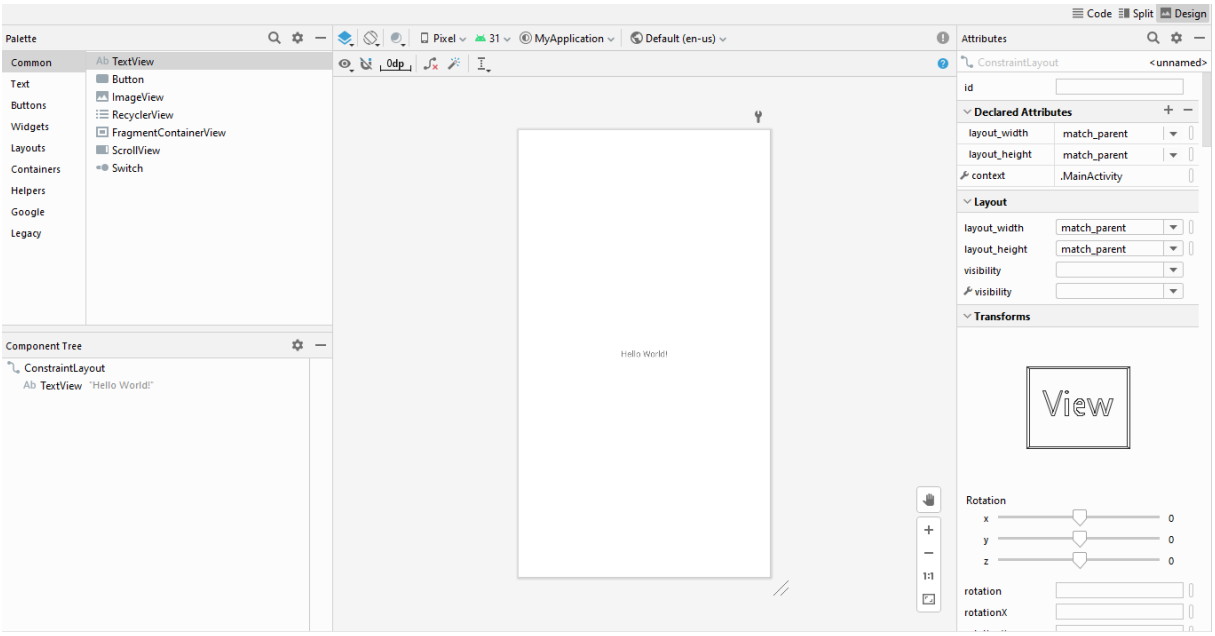
3.1 Vzhled aplikace

Aplikace obsahuje dvě takzvané aktivity. Při návrhu jejího designu je pro nás důležité, že jedna aktivita je reprezentována jako jedno okno aplikace. V našem případě se jedná o aktivitu pro hlavní obrazovku a aktivitu, která je společná pro vkládání a editaci bodů. Kromě nich aplikace obsahuje ještě dialog, který zobrazuje detail bodu zájmu.

Vzhled aplikace je popsán pomocí jazyka XML v jednotlivých souborech pro aktivity, případně dialogy, které se nachází ve složce res (resources) v podsložce layout. Protože aplikace je vyvíjená ve vývojovém prostředí Android Studio, není potřeba psát XML kód ručně, ale stačí použít editor, který nám umožní jednoduše přetahovat dané komponenty na obrazovku aktivity. Těmto komponentám poté můžeme nastavovat a měnit jejich atributy. Pokud tato změna ovlivní vzhled dané komponenty, ihned uvidíme tuto změnu i graficky na náhledu dané aktivity v editoru.

Na obrázku 5 je zobrazena ukázka nově vytvořené prázdné aktivity otevřené v editoru. V levé horní části pod nadpisem Palette můžeme vybírat mezi komponentami, které pomocí drag and drop (táhni a pusť) lze vložit do vytvořené aktivity zobrazené uprostřed. Pod výčtem komponent je zobrazen Component Tree, strom komponent v existujícím návrhu. Protože komponenty nemůžeme vkládat přímo do aktivity, musí být zvoleno nějaké rozložení (layout). V tomto případě se jedná o rozložení Constraint Layout, na kterém je zobrazena komponenta TextView

s nápisem Hello World! Na pravé části obrazovky můžeme měnit atributy daných komponent. Pokud bychom chtěli zobrazení přepnout čistě do kódu XML nebo vidět zároveň kód i návrh můžeme přepínat mezi možnostmi Code, Split a Design v pravém horním rohu.



Obrázek 5: Editor

3.1.1 Hlavní obrazovka

Vzhled hlavní obrazovky, která se zobrazí při zapnutí aplikace, je definován v souboru activity_main.xml. Náhled hlavní obrazovky je zobrazen na obrázku 7. Zvoleným rozložením je Constraint Layout. Pozice komponent v tomto rozložení se určuje pomocí připnutí dané komponenty k nadřazenému objektu. Aby byla pozice jasně určena, musí být připnuta horizontálně i vertikálně. V editoru se daná komponenta zobrazí na místě, na které byla přetažena. Po zapnutí aplikace, v případě, že jsme danou komponentu nepřipnuli k žádnému nadřazenému objektu, bude ale zobrazena na pozici [0, 0] [24].

První vloženou komponentou je komponenta mapView, která zobrazí mapový podklad. Při spuštění aplikace je mapa vykreslena v místě, kde se uživatel právě nachází, toto místo je označeno modrou tečkou. Mapa zabírá celou plochu obrazovky a jsou na ní vykresleny jednotlivé body zájmu. Po kliknutí na bod na mapě se zobrazí detail daného bodu. Tyto body jsou na mapě označeny ikonkou podle kategorie, do které konkrétní bod patří, viz obrázek 6.

Obrázek 6: Kategorie bodů zájmu

Modrá ikonka značí místa, která jsou vhodná ke koupání se psem. Zelená ikonka slouží k označení hřišť, míst v přírodě a podobně. Na místě oranžové ikonky se nachází restaurace, která dovoluje vstup se psy a červená ikonka označuje místo, kde se nachází odpadkový koš. Ikonky byly vytvořeny pomocí Maki Icons Editoru na stránkách společnosti Mapbox. Editor obsahuje již předpřipravené ikonky bodů zájmu, které lze mírně upravovat například změnou barvy [25]. Tyto ikonky jsou open-source a spadají pod licenci CC0. Ikonky pod touto licencí mohou být jakkoliv zpracovávány, upravovány a distribuovány. Přidáním této licence se autor vzdává svých autorských práv [26].

V horní části obrazovky je umístěna komponenta Chip Group, která sdružuje jednotlivé Chip komponenty. V našem případě slouží Chipy k filtrování bodů zobrazených na mapě podle kategorie. Na mapě jsou poté zobrazeny pouze ty body, které jsou pomocí Chipů zaškrtnuty. Každý Chip je označený příslušnou ikonkou viz obrázek 6 a krátkým textem, který udává, o jaké místo se jedná.

Poslední dvě komponenty jsou tlačítka FloatingActionButton v pravém dolním rohu. Horní tlačítko vrátí po kliknutí pohled na mapě na pozici, kde se uživatel právě nachází. Spodní tlačítko umožňuje přidat nový bod a po kliknutí nás přesune na obrazovku vkládání bodu.

Obrázek 7: Hlavní obrazovka

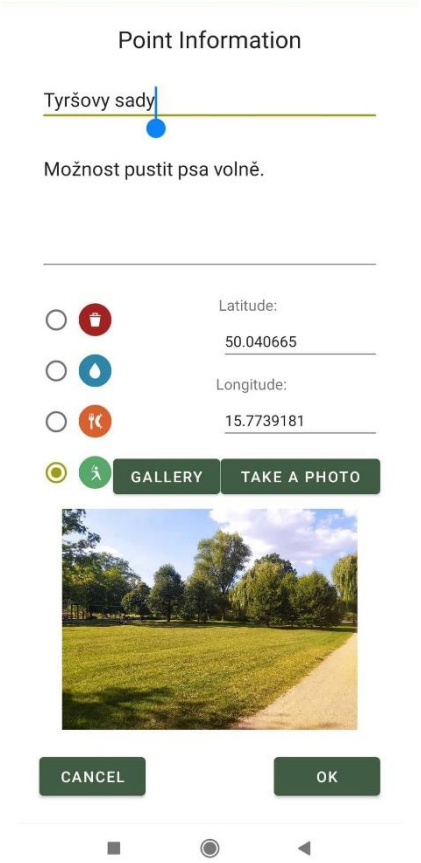
3.1.2 Obrazovka vkládání a editace

Vzhled této aktivity je definován v souboru aktivity_insert_point.xml a je zobrazen na obrázku 8. Na tuto aktivitu je možné dostat se dvěma způsoby. Z hlavní obrazovky kliknutím na tlačítko + v pravém dolním rohu nebo při rozkliknutí detailu bodu po kliknutí na tlačítko Edit. Stejně jako v předchozí aktivitě je zvoleno rozložení Constraint Layout.

První komponentou je TextView s textem Point Information, který je stejný jak v módu editace, tak v módu vkládání. Její text v aplikaci změnit nelze a při vkládání bodu se nevyužívá. Tato komponenta uživateli pouze oznamuje skutečnost, že se jedná o stránku, kde vyplňujeme informace o bodu zájmu. Následují dvě komponenty EditText pro název bodu a popis bodu. V případě, že se nacházíme v módu editace, jsou tyto hodnoty již předvyplněné názvem a popisem konkrétního existujícího bodu. Pro výběr kategorie bodu jsou v aktivitě přidány komponenty RadioButton sdružené v komponentě RadioGroup. Pro jeden bod může být vybrána pouze jedna kategorie současně. O jakou kategorii se jedná je rozlišeno pomocí ikonek viz obrázek 6. Souřadnice se zadávají pomocí dvou komponent EditText, do kterých lze zadávat

pouze čísla a označují zeměpisnou šířku a zeměpisnou délku. Všechny komponenty zmíněné v tomto odstavci jsou povinné a nesmí zůstat prázdné.

Poslední nepovinnou položkou je fotografie. Fotografii lze vložit kliknutím na tlačítko Gallery nebo Take a Photo. Tyto tlačítka jsou umístěna nad komponentou ImageView, která vyfocenou fotografii, respektive fotografii vybranou z galerie, zobrazí. Na dolní hraně obrazovky je umístěno tlačítko Cancel, které nás vrátí na hlavní obrazovku, a tlačítko Ok, kterým potvrdíme vytvoření nebo úpravu daného bodu a také se vrátíme na hlavní obrazovku.



Obrázek 8: Vložení/Editace bodu

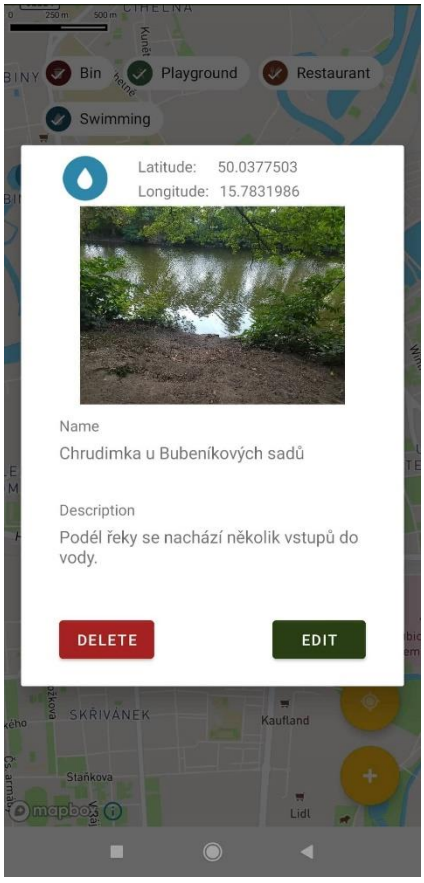
3.1.3 Detail bodu zájmu

Poslední důležitou vizuální částí aplikace je dialog pro zobrazení detailu bodu zájmu znázorněném na obrázku 9. Vzhled dialogu je popsán v souboru custom_point_dialog.xml. Tento dialog se zobrazí po kliknutí na libovolný bod na mapě. Stejně jako aktivita pro hlavní obrazovku a aktivita pro vložení a editaci bodu zájmu využívá rozložení Constraint Layout.

V levém horním rohu je pomocí komponenty ImageView zobrazena ikonka reprezentující kategorii bodu zájmu. Vedle ikonky se nachází dvě komponenty TextView, které slouží k zobrazení souřadnic, konkrétně k zobrazení zeměpisné šířky a zeměpisné délky daného bodu.

Následně je pomocí komponenty ImageView zobrazena fotografie bodu. Název bodu je zobrazen komponentou TextView pod fotografií. Stejně jako název bodu je i jeho popis umístěn v totéž komponentě pod názvem bodu.

Detail bodu zájmu obsahuje dvě tlačítka. Tlačítko Delete pro jeho smazání a tlačítko Edit pro úpravu. Po zmáčknutí tlačítka Delete se smaže bod, jehož detail byl otevřený, detail se zavře a na hlavní obrazovce je pohled na mapě přesunut na místo, kde se odstraněný bod dříve nacházel. Automatickým přesunutím pohledu kamery na mapě na souřadnice odstraněného bodu, je uživateli ukázáno, že se daný bod korektně smazal. Po stisknutí tlačítka Edit je zapnuta aktivita společná pro přidání nového bodu a úpravu bodu. Této aktivitě je předána informace o tom, že se jedná o editaci bodu a také bod samotný, aby bylo možné předvyplnit potřebná pole v aktivitě. Konkrétně pole název, popis, kategorie, zeměpisná šířka, zeměpisná délka a fotografie, budou v aktivitě pro editaci bodu již předvyplněny dle informací, které si o sobě uchovává bod zájmu, který byl zobrazen v detailu. Detail bodu zájmu lze zavřít kliknutím mimo zobrazený dialog.



Obrázek 9: Detail bodu zájmu

4 VÝVOJ APLIKACE

4.1 Struktura aplikace

Ještě před samotným popisem vývoje aplikace je nutné zmínit strukturu aplikace. Soubory potřebné při vývoji aplikace jsou dostupné na následující cestě: app/src/main/java/com/upa/mobileapplicationfordogowners. Jsou zde vytvořeny čtyři složky:

Složka activities, která obsahuje zdrojový kód aktivity pro vložení a editaci obrázku – InsertPointActivity.kt, hlavní obrazovky – MainActivity.kt a dialogu pro zobrazení detailu bodu zájmu – CustomDialogShowPoint.kt.

Složka database, která obsahuje potřebné kódy pro práci databáze. Konkrétně DatabasePointsOfInterest.kt, PointOfInterestDao.kt, PointOfInterestEntity.kt. Součástí adresáře database je také soubor Category.kt, který obsahuje enum sloužící k rozlišení kategorií bodů zájmu.

Složka helpers uchovává soubor PermissionHelper.kt, který obsahuje pomocný kód, jenž pomáhá při vyžadování nutných povolení od uživatele.

A poslední složka validation, v které se nachází soubor PointOfInterestValidator.kt. Kód v tomto souboru slouží ke zvalidování bodu zájmu před vložením do databáze.

4.2 Mapové podklady

4.2.1 Tokeny

Prvním krokem při vývoji aplikace, kromě návrhu vzhledu aplikace, který je popsán v sekci 3.1, je vložit mapové podklady do projektu a dále s nimi pracovat. K tomu, aby bylo možné mapové podklady do projektu vložit, potřebujeme veřejný a soukromý token. Zisku tokenů předchází vytvoření Mapbox účtu na stránkách společnosti. Každý uživatel má vygenerovaný výchozí veřejný token, případně je k dispozici možnost vygenerovat jiný veřejný token. Tento token se vkládá do souboru string.xml pod názvem mapbox_access_token. Dále se vytváří soukromý token s parametrem DOWNLOADS:READ. Zobrazí se pouze jednou a znovu zobrazit nelze. Vkládá se do souboru gradle.properties pod názvem MAPBOX_DOWNLOADS_TOKEN [27].

4.2.2 Přidání závislosti

Dalším krokem je vložení Mapbox Maps SDK jako závislost do projektu. Pro verzi Android Studia Arctic Fox (2020.3.1) a vyšší se kód vkládá do souboru settings.gradle. Tento zdrojový

kód je dostupný na sránkách Mapboxu v návodu k instalaci mapových podkladů pro Android [27].

Pro korektní funkčnost musí být atribut minSdk v souboru build.gradle nastaven na číslo 21 nebo vyšší. Dále se ve stejném soboru přidá nové pravidlo pro sestavení pro Mapbox Maps SDK dle zdrojového kódu uvedeného v instalačním návodu. Nakonec stačí synchronizovat Gradle soubory, což nám samo Android Studio nabídne, protože některé soubory byly měněny [27]. Ve vyvíjené aplikaci je místo verze 10.7.0, která je k 11. 8. 2022 uváděna na stránkách Mapboxu, využita starší verze 10.2.0, protože během implementace map do projektu, byla právě tato verze nejnovější dostupnou verzí.

4.2.3 Vložení mapy

Komponenta map se vkládá do souboru activity_main.xml pomocí zdrojového kódu, který je uveden v již zmiňovaném návodu [27]. V tento moment projekt obsahuje komponentu MapView, s kterou lze dále pracovat.

4.2.4 Práce s mapovým podkladem při vývoji

Při práci s mapovým podkladem je potřeba od uživatele vyžádat oprávnění na získání jeho aktuální lokace. Více o oprávnění je dostupné v sekci 4.7. Ve vyvíjené aplikaci je mapový podklad využit pro zobrazení bodů zájmu a aktuální pozice uživatele. Před implementací těchto funkcionalit je nutné přiřadit id komponenty MapView do proměnné. V tomto případě je nejprve deklarována globální proměnná, do které je následně toto id přiřazeno viz zdrojový kód 1. Funkce initializeMap je volána při spuštění aktivity. Přiřadí id komponenty do proměnné, načte nový styl mapy pomocí funkce loadStyleUri a povolí zobrazení pozice uživatele na mapě.

```
    private var mapView: MapView? = null
    private fun initializeMap() {
        mapView = findViewById(R.id.mapView)
        mapView?.getMapboxMap()?.loadStyleUri(Style.MAPBOX_STREETS) {
            mapView!!.location.updateSettings {
                enabled = true
                pulsingEnabled = true
            }
        }
    }
```

Zdrojový kód 1: Inicializace mapy

Kromě zobrazení pozice uživatele je také nutné přesunout se na mapě na správné místo. Tedy na místo, kde se uživatel právě nachází. Proto se nastavuje pozice kamery viz zdrojový kód 2. Nastaví se zeměpisná šířka, zeměpisná délka a pohled na mapu se přiblíží na požadovanou vzdálenost. Poté se pomocí funkce setCamera(cameraOptions) tyto změny aplikují.

```
    private fun setMapPosition(coordinates: DoubleArray) {
        val cameraPosition = CameraOptions.Builder()
            .center(
                Point.fromLngLat(
                    coordinates.get(1), // longitude
                    coordinates.get(0)  // latitude
                )
            )
            .zoom(13.0)
            .build()

        mapView?.getMapboxMap()?.setCamera(cameraPosition)
    }
```

Zdrojový kód 2: Nastavení pozice kamery na mapě

Zobrazení bodů využívá takzvané anotace. Tato anotace je na mapě zobrazena jako ikona. Použité ikony jsou vyobrazeny na obrázku 6. Aby bylo možné anotaci přidat musí se získat instance třídy PointAnnotationManager. Bod je reprezentován instancí třídy PointAnnotationOptions, které se nastaví ikona, pozice na mapě a velikost ikony. Pomocí PointAnnotationManager a funkce create(pointAnnotationOptions) je tento bod zobrazen. Příklad práce s anotacemi lze nalézt v návodu Mapboxu pro Maps SDK pro Android v sekci markery a anotace [28]. Bod zájmu na mapě má ještě jednu funkci a tou je zobrazení detailu

bodu po kliknutí. Proto se každému bodu musí nastavit listener, který po kliknutí na daný bod zavolá funkci zobrazující detail konkrétního bodu.

4.3 Lokace uživatele

Lokace uživatele je potřebná při práci s body zájmu, protože je o nich evidováno místo kde se nachází, a při práci s mapovými podklady při změně pozice kamery. Získání lokace ve vyvíjené aplikaci je zobrazeno ve zdrojovém kódu 3. Zisku lokace předchází získání oprávnění viz sekce 4.7. Po získání oprávnění je zisk lokace triviální. Pomocí instance třídy LocationManager a funkce getLastKnownLocation(provider) získáme instanci třídy Location. Následně pracujeme s atributy latitude (zeměpisná šířka) a longitude (zeměpisná délka) této třídy.

```
1  @SuppressLint("MissingPermission")
2      private fun getCurrentLocation(): Location? {
3          if (!arePermissionsGranted()) {
4              return null
5          }
6          val locationManager: LocationManager =
7              getSystemService(Context.LOCATION_SERVICE)
8              as LocationManager
9          return locationManager
10                 .getLastKnownLocation(LocationManager.GPS_PROVIDER)
11     }
```

Zdrojový kód 3: Získání aktuální lokace uživatele

4.4 Databáze

Body zájmu jsou ukládány do lokální databáze Room. Pro správnou funkčnost databáze jsou potřeba tři základní komponenty: Datové entity, které v databázi představují tabulky aplikace. Pro vyvíjenou aplikaci je entitou třída PointOfInterestEntity. Databázová třída, která uchovává databázi a skrz níž se přistupuje k datům. V tomto případě se jedná o třídu DatabasePointsOfInterest. A rozhraní PointOfInterestDao, které poskytuje metody potřebné k manipulaci s daty v databázi. Než bude možné databázi využívat je nutné přidat potřebné závislosti do souboru build.gradle, které jsou uvedeny v Google Developers dokumentaci pod záložkou Save data in a local database – Setup a synchronizovat příslušné soubory [29].

Databázová třída DatabasePointsOfInterest musí být opatřena anotací @Database, která označuje pole entit, které budou součástí databáze. V případě vyvíjené mobilní aplikace se jedná pouze o PointOfInterestEntity. Dále obsahuje verzi databáze. Databázová třída musí být

abstraktní a musí dědit ze třídy RoomDatabase. Poslední podmínkou je nutnost deklarovat abstraktní funkci pro každou DAO třídu, v tomto případě pouze jednu funkci pro PointOfInterestDao, která nemá žádné argumenty a vrací instanci příslušné DAO třídy [29]. Ukázka kódu viz zdrojový kód 4.

```
1 @Database(entities = [PointOfInterestEntity::class], version = 1)
2 abstract class DatabasePointsOfInterest : RoomDatabase() {
3     abstract fun pointOfInterestDAO(): PointOfInterestDao
4 }
```

Zdrojový kód 4: Databázová třída

Každá instance databázové entity PointOfInterestEntity představuje jeden bod zájmu. Ze zdrojového kódu 5 vyplývá jakou má bod zájmu strukturu. Tedy, že o něm jsou uchovávány informace o jméně, popisu, kategorii, zeměpisné šířce, zeměpisné délce a adrese obrázku. Zároveň je pro každý z nich automaticky vygenerován primární klíč sloužící k jeho jednoznačné identifikaci.

```kotlin
@Entity
data class PointOfInterestEntity(
    @ColumnInfo(name = "name")
    var name: String,

    @ColumnInfo(name = "description")
    var description: String,

    @ColumnInfo(name = "category")
    var category: Category,

    @ColumnInfo(name = "latitude")
    var latitude: Double,

    @ColumnInfo(name = "longitude")
    var longitude: Double,

    @ColumnInfo(name = "imageUri")
    var imageUri: String,

    @PrimaryKey(autoGenerate = true)
    val id: Int = 0
)
```

Zdrojový kód 5: Databázová entita

Rozhraní PointOfInterestDao poskytuje metody pro manipulaci s daty, které se nachází v databázi. Ve zdrojovém kódu 6 jsou uvedeny příklady implementovaných metod. První metoda slouží k získání všech bodů z databáze, jejichž kategorie je jedna z kategorií na vstupu. Další je metoda pro získání bodu podle id. A poslední pro smazání bodu podle id. Kromě nich jsou ve vyvíjené aplikaci také využity metody pro získání všech bodů, vložení bodu a aktualizaci bodu, které jsou také implementovány v rozhraní PointOfInterestDao.

```
1     @Query
2     ("SELECT * FROM PointOfInterestEntity
3     WHERE category IN (:filterCategories)")
4     fun getPointsByCategory(filterCategories: List<Category>)
5     : List<PointOfInterestEntity>
6
7     @Query("SELECT * FROM PointOfInterestEntity WHERE id=:id ")
8     fun getPointById(id: Int): PointOfInterestEntity
9
10    @Query("DELETE FROM PointOfInterestEntity WHERE id=:pointId")
11    fun deleteByPointId(pointId: Int)
```

Zdrojový kód 6: Vybrané metody pro manipulaci s daty v databázi

Před použitím databáze je nutné vytvořit instanci databáze [29]. Její využití je poté například v aktivitě InsertPointActivity, kdy se po kliknutí na tlačítko OK zavolá buď metoda insert nebo metoda update, definovaná v rozhraní PointOfInterestDao, s bodem zájmu jako parametrem podle toho, zda se aktivita otevřela v módu pro vytváření nového bodu nebo v módu pro editaci. Tedy zda byla aktivita otevřena přes tlačítko + na hlavní obrazovce, nebo přes tlačítko Edit v detailu bodu. Ukázka vytvoření instance databáze a zavolání metody je ve zdrojovém kódu 7, kde proměnná pointForEdit je instancí třídy PointOfInterestEntity.

```
1 val db = Room.databaseBuilder(
2             applicationContext,
3             DatabasePointsOfInterest::class.java,
4             "databasePointsOfInterest"
5         ).build()
6
7 db.pointOfInterestDAO().updatePoint(pointForEdit)
```

Zdrojový kód 7: Vytvoření instance databáze a volání metody update

4.4.1 Coroutines

Aby bylo možné volat metody pro manipulaci s daty z databáze je potřeba implementovat coroutines. Jedná se o knihovnu, která umožňuje zpracovat kód asynchronně. Ve vyvíjené aplikaci je tohoto přístupu využito, protože volání do databáze může trvat delší dobu a mohlo by se stát, že hlavní vlákno aplikace bude zablokované tak dlouho, že se uživateli začne aplikace jevit, jako že přestala reagovat. Aby bylo možné coroutines použít v projektu musí se přidat závislost do souboru build.gradle a změněné soubory synchronizovat. Kód přidané závislosti je dostupný například na stránkách Google developers v Advanced Kotlin guides pod záložkou

Kotlin coroutines on Android [30]. Příklad použití pro smazání bodu z databáze je zobrazen ve zdrojovém kódu 8.

```
1 CoroutineScope(Dispatchers.IO).launch {
2        db.pointOfInterestDAO().deleteByPointId(pointId)
3 }
```

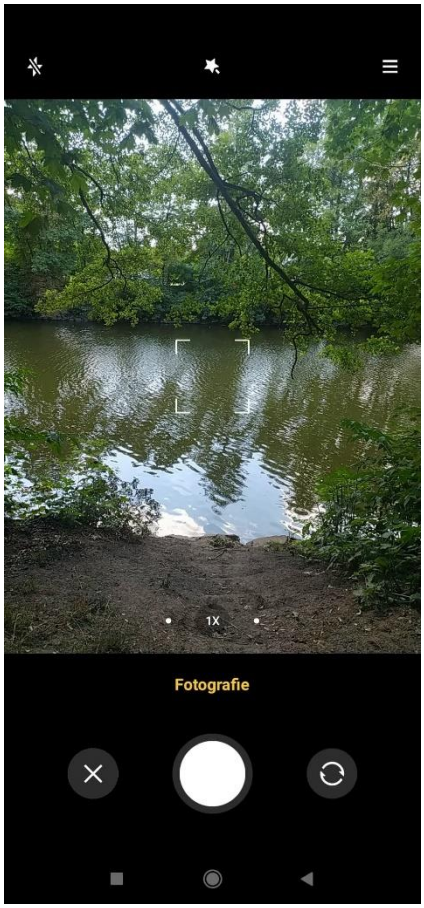
Zdrojový kód 8: Coroutines

4.5 Získání fotografie

Ve vyvíjené aplikaci jsou dvě možnosti vkládání fotografií, z úložiště zařízení nebo pomocí fotoaparátu zařízení. Získání fotografie je volitelná akce při vkládání nebo editaci bodu v aktivitě InsertPointActivity.

4.5.1 Fotoaparát zařízení

K zachycení fotografie z fotoaparátu zařízení je potřeba několik věcí, počínaje získáním oprávnění ke spuštění fotoaparátu zařízení a přístupu k jeho úložišti. Oprávnění jsou popsána v sekci 4.7. Dále takzvaný Intent, který popisuje, jaká aktivita má začít. Kromě toho je schopen přenášet data mezi aktivitami [31]. Pomocí Intentu je například řešeno i rozlišení, zda se jedná o mód editace nebo o mód vkládání bodu při volání aktivity pro editaci nebo vytvoření bodu. Dále je potřeba zavolat samotnou aktivitu, a nakonec zpracovat získanou fotografii [32]. Vzhled zavolané aktivity je zobrazen na obrázku 10. V případě vyvíjené aplikace je zachycená fotografie uložená v lokálním úložišti zařízení. Aby bylo možné fotografii uložit do úložiště zařízení je nutné poskytnout adresu, kde má být výsledná fotografie umístěna a její název, tedy poskytnou URI. Kromě uložení fotografie dojde také k zobrazení fotografie v komponentě ImageView.



Obrázek 10: Fotoaparát

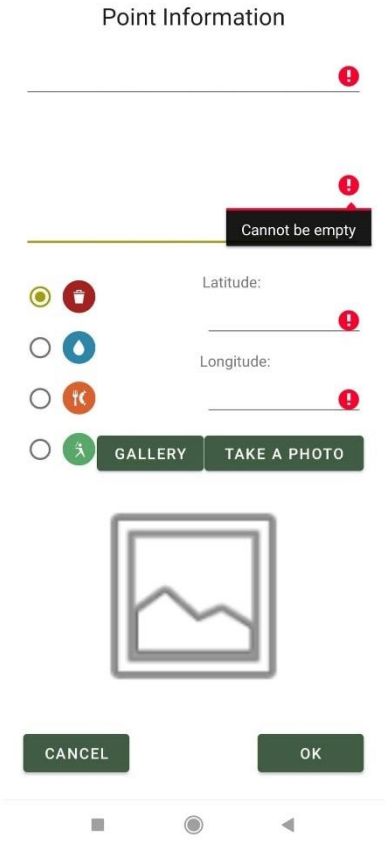
4.5.2 Úložiště zařízení

K získání fotografie z úložiště zařízení dochází obdobným způsobem, ke kterému jsou potřeba oprávnění pro přístup do úložiště. Oprávnění jsou popsaná v sekci 4.7. Dále je potřeba Intent, který specifikuje, o jakou akci se jedná. Tedy, že budeme získávat fotografii. Následně se zavolá samotná aktivita. Stejně jako u zachycení fotografie fotoaparátem je i vybraná fotografie z úložiště zařízení zobrazena v komponentě ImageView.

4.6 Validace

Každý vstup od uživatele je zvalidován, v tomto případě se jedná o všechny komponenty EditText v aktivitě pro editaci nebo vytvoření bodu. K validaci byla vytvořena pomocná třída PointOfInterestValidator, která poskytuje veřejnou funkci validate. Tato funkce je volána před vložením nového bodu do databáze nebo před aktualizací bodu a vrací pouze hodnoty pravda nebo nepravda. Tedy zda je vstup validní nebo ne. Při vkládání nebo aktualizaci bodu nesmí jméno, popis, zeměpisná šířka a zeměpisná délka zůstat prázdné nebo obsahovat pouze bílé znaky. Zároveň zeměpisná šířka musí být číslo datového typu double s hodnotou od -90 do 90.

Pro zeměpisnou délku platí, že hodnota datového typu double musí být od -180 do 180. V případě nedodržení podmínek se příslušná akce neprovede a uživateli je pomocí červeného vykřičníku u příslušné kolonky zobrazen důvod neúspěšného pokusu viz obrázek 11.

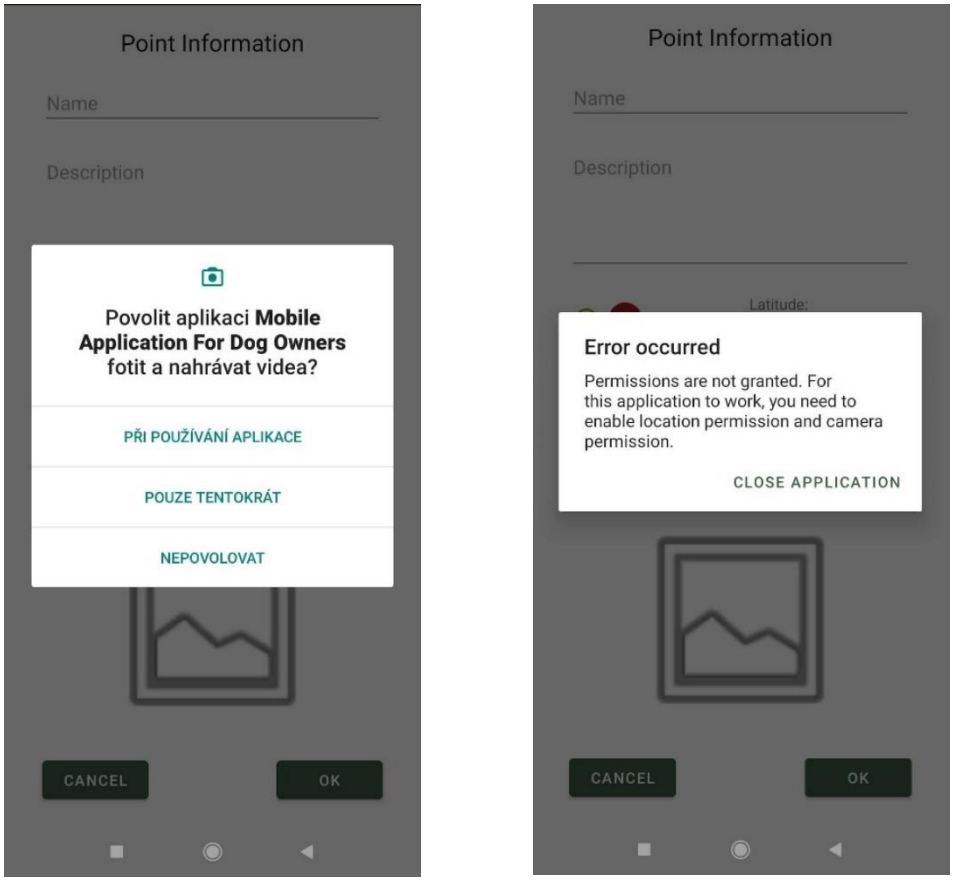


Obrázek 11: Neúspěšná validace vstupů

4.7 Oprávnění

Pro vývoj aplikace jsou důležité dva druhy oprávnění: normální a dangerous neboli nebezpečné. Normální oprávnění je pouze definováno v souboru AndroidManifest.xml a při instalaci aplikace je uděleno automaticky bez nutnosti vyzývat uživatele. K udělení oprávnění označené jako dangerous musí být uživatel vyzván [33]. Ve vyvíjené aplikaci je uživatel vždy vyzván předtím, než je dané oprávnění potřeba, například po kliknutí na tlačítko Take a Photo v aktivitě pro přidání nebo editaci bodu se zobrazí následující dialog viz obrázek 12. V případě jeho neudělení je uživatel vyzván k ukončení aplikace a k jeho udělení, tento dialog je zobrazen na obrázku 12. Aplikaci lze používat i bez nich, ale není zaručená její správná funkčnost a samozřejmě bez oprávnění nebudou příslušné části aplikace fungovat vůbec. K získání nebezpečných oprávnění je v projektu vytvořena třída PermissionHelper. V hlavní aktivitě jsou potřeba pro získání aktuální lokace uživatele. V aktivitě pro vložení a editaci bodu jsou potřeba

pro přístup k fotoaparátu zařízení, přístup do úložiště zařízení a povolení získání aktuální lokace uživatele.



Obrázek 12: Oprávnění

5 TESTOVÁNÍ APLIKACE

Aplikace byla poskytnuta několika osobám s různými znalostmi z oblasti informačních technologií. Jejich úkolem bylo otestovat nejenom funkčnost aplikace, ale i intuitivnost ovládání vyvíjené aplikace. Na základě jejich hodnocení proběhlo během samotného vývoje několik změn. Například potřeba přidávat fotografie nejenom z fotoaparátu zařízení, ale i z úložiště telefonu nebo mírné změny při rozložení prvků.

Při testování finální verze měla aplikace kladné ohlasy. Ovládání bylo dostatečně intuitivní a rozložení prvků bylo vyhovující. Nikdo z dotazovaných neměl problém aplikaci používat a během testování se aplikace chovala korektně.

ZÁVĚR

Cílem bakalářské práce bylo vyvinout aplikaci pro operační systém Android sloužící pro majitele psů, která bude evidovat body zájmu a ukládat je do databáze. Místa jsou ukládána s názvem, popisem, souřadnicemi, kategorií a volitelnou fotografií. Aplikace všechny požadavky splňuje. Kromě toho navíc umožňuje filtrování bodů na mapě, zobrazuje pozici uživatele a poskytuje více možností při získávání fotografie. Aplikace také zobrazuje body pomocí čtyř různých ikonek pro každou kategorii bodu zájmu.

Aplikaci může využít běžný uživatel, který by si rád zaznamenával místa, která se svým psem navštívil. Také má velký potenciál dále se rozvíjet, například přidáním možnosti ukládání bodů do externí databáze a podporu uživatelských účtů a s tím související možnost přidat více fotografií, hodnocení míst, přidávání komentářů a podobně. Po takovýchto úpravách by mohla aplikace začít sloužit například i pro města, jako tomu je u webové aplikace Psí koše Krnov, nebo jako psí sociální síť, kde si uživatelé budou moci navzájem sdílet poznatky a fotografie z daných míst.

POUŽITÁ LITERATURA

[1] ČSÚ [Český statistický úřad]. *Osoby v ČR používající chytrý telefon a telefon bez operačního systému - vývoj v letech 2018 - 2021*. [online] 2021-11-23 [cit. 2021-12-17]. Dostupné z: https://www.czso.cz/documents/10180/142872020/062004210302.pdf/2ce79b67-a493-4cfb-8e3e-5b3a235754d5?version=1.1

[2] GEMIUS RANKING. *Operating systems – families.* [online] © Copyright Gemius 2021 [cit. 2021-12-17]. Dostupné z: http://ranking.gemius.com/cz/ranking/systems/

[3] BringFido. *About BringFido*. [online] © 2005-2022 Kendall Media, Inc. [cit. 2022-08-06]. Dostupné z: https://www.bringfido.com/about/

[4] BringFido. [online] © 2005-2022 Kendall Media, Inc. [cit. 2022-08-05]. Dostupné z: https://www.bringfido.com/

[5] BarkHappy. *The BarkHappy Team*. [online] © 2022 BarkHappy Inc. [cit. 2022-08-06]. Dostupné z: https://barkhappy.com/about/

[6] BarkHappy. [online] © 2022 BarkHappy Inc. [cit. 2022-08-06]. Dostupné z: https://barkhappy.com/

[7] GooglePlay. BarkHappy. [online] 2022-06-02 [cit. 2022-08-14]. Dostupné z: https://play.google.com/store/apps/details?id=com.barkhappy&hl=cs&gl=US

[8] Fiddo. *Aplikace Fiddo pro všechny pejskaře a jejich čtyřnohé parťáky.* [online] [cit. 2022-08-06]. Dostupné z: https://fiddo.cz/

[9] *GEOPORTÁL KRNOV - Koše na psí exkrementy*. [online] (c) Městský úřad Krnov. [cit. 2022-08-06]. Dostupné z: https://portal.geostore.cz/mycitykrnov/psikose

[10] Codecademy Team. *What Is Kotlin Used For?* [online] 2022-03-01 [cit. 2022-08-07]. Dostupné z: https://www.codecademy.com/resources/blog/what-is-kotlin-used-for/

[11] Kotlin. *Past.* [online] [cit. 2022-08-06]. Dostupné z: https://kotlinlang.org/lp/10yearsofkotlin/past/

[12] Google Developers. *Develop Android apps with Kotlin.* [online] [cit. 2022-08-06]. Dostupné z: https://developer.android.com/kotlin

[13] PCMag. *Java Virtual Machine.* [onlne] © 1996-2022 ZIFF DAVIS. PCMAG DIGITAL GROUP [cit. 2022-08-07]. Dostupné z: https://www.pcmag.com/encyclopedia/term/java-virtual-machine

[14] Kotlin docs. *Kotlin Native.* [online] 2022-08-05 [cit. 2022-08-07]. Dostupné z: https://kotlinlang.org/docs/native-overview.html

[15] QUIN, Liam R. E. *XML Essentials – W3C*. [online] Copyright © 2015 W3C ® (MIT, ERCIM, Keio, Beihang) [cit. 2022-08-08]. Dostupné z: https://www.w3.org/standards/xml/core

[16] PETERSON, Richard. *What is SQL? Learn SQL Basics, SQL Full Form & How to Use.* [online] 2022-07-26 [cit. 2022-08-08]. Dostupné z: https://www.guru99.com/what-is-sql.html

[17] Kotlin docs. *IDEs for Kotlin development.* [online] 2022-08-05 [cit. 2022-08-07]. Dostupné z: https://kotlinlang.org/docs/kotlin-ide.html

[18] Google Developers. *Meet Android Studio*. [online] 2022-06-21 [cit. 2022-08-07]. Dostupné z: https://developer.android.com/studio/intro

[19] Storemapper. *Best Google Maps API Alternative.* [online] 2022-06-14 [cit. 2022-08-08]. Dostupné z: https://www.storemapper.com/blog/google-maps-api-alternative/

[20] Google Maps Platform. *Pricing that scales to fit your needs.* [online] [cit. 2022-08-08]. Dostupné z: https://mapsplatform.google.com/pricing/

[21] SANDOVAL, Kristopher. *What is the Difference Between an API and SDK?* [online] 2016-06-02 [cit. 2022-08-08]. Dostupné z: https://nordicapis.com/what-is-the-difference-between-an-api-and-an-sdk/

[22] Mapbox. *Mapbox pricing.* [online] © Mapbox [cit. 2022-08-08]. Dostupné z: https://www.mapbox.com/pricing/

[23] Mapbox | Docs. *Documentation.* [online] [cit. 2022-08-08]. Dostupné z: https://docs.mapbox.com/

[24] Google developers. *Build a Responsive UI with ConstraintLayou*t. [online] 2022-05-27 [cit. 2022-08-09]. Dostupné z: https://developer.android.com/training/constraint-layout

[25] Mapbox. *Maki Icons Editor.* [online] [cit. 2022-08-09]. Dostupné z: https://labs.mapbox.com/maki-icons/editor/

[26] Creative Commons. *CC0 1.0 Universal (CC0 1.0) Public Domain Dedication.* [online] [cit. 2022-08-09]. Dostupné z: https://creativecommons.org/publicdomain/zero/1.0/deed.en

[27] Mapbox. *Installation.* [online] [cit. 2022-08-11]. Dostupné z: https://docs.mapbox.com/android/maps/guides/install/

[28] Mapbox. *Annotations.* [online] [cit. 2022-08-12]. Dostupné z: https://docs.mapbox.com/android/maps/guides/annotations/annotations/

[29] Google developers. *Save data in a local database using Room*. [online] [cit. 2022-08-12]. Dostupné z: https://developer.android.com/training/data-storage/room

[30] Google developers. *Kotlin coroutines on Android.* [online] [cit. 2022-08-13]. Dostupné z: https://developer.android.com/kotlin/coroutines

[31] Google developers. *Intents and Intent Filters*. [online] [cit. 2022-08-13]. Dostupné z: https://developer.android.com/guide/components/intents-filters

[32] Google developers. *Take photos.* [online] [cit. 2022-08-13]. Dostupné z: https://developer.android.com/training/camera/photobasics

[33] Google developers. <*permission*>. [online] [cit. 2022-08-13]. Dostupné z: https://developer.android.com/guide/topics/manifest/permission-element.html#plevel