FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Aplikace pro sběratele turistických známek na OS Android
Student:	Matyáš Frank
Vedoucí:	Ing. Jiří Hunka
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Cílem této práce je navrhnout, realizovat a otestovat aplikaci podporující činnosti sběratele turistických známek.

Postupujte v těchto krocích:
1. Analyzujte problematiku turistických známek.
2. Pokuste se navázat úzkou spolupráci s výrobcem a autorem známek – TURISTICKÉ ZNÁMKY s. r. o.
3. Proveďte důkladný sběr požadavků – jako zdroj se pokuste využít jak TURISTICKÉ ZNÁMKY s. r. o., tak případné uživatele.
4. Na základě analýzy a získaných dat proveďte vhodný návrh.
5. Návrh řádně diskutujte se zadavatelem v případě možností i se společností TURISTICKÉ ZNÁMKY s. r. o.
6. Implementujte navržené řešení.
7. Proveďte vhodné testování.
8. Navrhněte možná vylepšení do budoucnosti.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 4. února 2022 v Praze.

Bakalářská práce

APLIKACE PRO SBĚRATELE TURISTICKÝCH ZNÁMEK NA OS ANDROID

Matyáš Frank

Fakulta informačních technologií
Katedra softwarového inženýrství
Vedoucí: Ing. Jiří Hunka
15. února 2023

Odkaz na tuto práci: Frank Matyáš. *Aplikace pro sběratele turistických známek na OS Android.* Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů kódu

Chtěl bych poděkovat především mému vedoucímu práce Ing. Jiřímu Hunkovi. I když komunikace z mé strany vázla, nezlomil nade mnou hůl a nadále se mi snažil pomáhat. Dále bych chtěl poděkovat respondentům, kteří odpovědli na mé analyzační a testovací dotazníky.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při 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 citovaného zákona

V Praze dne 15. února 2023 .....

Abstrakt

Tato práce se zabývá procesem tvorby klientské aplikace na operační systém Android, pro firmu TURISTICKÉ ZNÁMKY s. r. o. Jedná se o systém, jehož hlavním účelem je správa známek a infomací o nich. Práce ve svém začátku rozebírá informační základnu. Následně analyzuje problematiky TURISTICKÉ ZNÁMKY s. r. o., webové stránky a existujících aplikací, ze kterých vznikají potřebné požadavky. Podle požadavků je sestaven design aplikace, který je posléze implementován, otestován a zdokumentován.

Klíčová slova turismus, sběratelství, plánování, Android, Jetpack Compose, Kotlin

Abstract

This thesis is deals with the process of creation a client-side Android application for company TURISTICKÉ ZNÁMKY s. r. o. This is an system, which main purpose is to maintain stamps and information about them. First, the work describes the knowledge behind. Subsequently, an analysis of TURISTICKÉ ZNÁMKY s. r. o., web and existing apps was performed and out of it were created requirements. With the help of requirements the design could be created, implemented, tested and documented.

Keywords tourism, collecting, planning , Android, Jetpack Compose, Kotlin

Seznam zkratek

API	Application Programming Interface
CRUD	Create, Read, Update, Delete
ČVUT	České vysoké učení technické v Praze
DAO	Data access object
FIT	Fakulta informační technologií
UI	User interface
UX	User experience
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
OAuth	Open Authorization
OS	Operační systém
REST	Representational State Transfer
SQL	Structured Query Language
s.r.o.	Společnost s ručením omezeným
SVG	Scalable Vector Graphics
URL	Uniform Resource Locator
XML	Extensible Markup Language
TZ	Turistické známky

Úvod

S přicházejícím věkem, kdy čím dal více roste zájem o chytré telefony, je otázkou času, kdy jejich průměrný počet na člověka bude roven 1. Většina uživatelů přitom dobrovolně nechce používat internetové prohlížece, pokud existuje aplikace přímo pro daný systém.

Turistické známky, s.r.o., dále už je „TZ“ je firma, která se zaměřuje na výrobu a distribuci sběratelských předmětů. TZ mají jenom v Česku přes 2700 turistických známek. Při takovém počtu známek, už se ve sbírkách orientuje těžce. TZ tedy provozují oficiální webovou aplikaci, ve které uživatel může pracovat s vydanými předměty.

S aplikacemi pro chytré telefony je to složitější. Na IOS do nedávna vůbec neexistovala a na Android existuje pouze neoficiální, nad kterou ovšem TZ nemají kontrolu a navíc už od 27. října 2019 je ve verzi 2.15.1. V dubnu roku 2021 začali vyvíjet aplikaci pro IOS, ovšem oficiální aplikace pro OS Android nebyla v plánu. Netušili ovšem, že na FIT ČVUT se rodí tenhle projekt. Turistické známky, s.r.o. byly zkontaktovány, obeznámeny se záměrem vytvořit aplikaci pro OS Android. Tímto krokem začala spolupráce s firmou.

Motivací k tvorbě této práce bylo především reálné uplatnění projektu. Chuť objevovat nové možnosti, zejména v oblasti designu, analýzy a implementace se vyvinula společně s hlavní motivací.

Práce se skládá z 6. částí. První částí je rešerše, která popisuje informační podklady stojící za vývojem. Druhou částí je analýza, jejímž cílem je dát dohromady požadavky, které budou následně zaneseny do kapitoly třetí s názvem Design. Design vytváří přesnou představu o tom, jak bude vypadat vzhled a funkčnost aplikace. čtvrtá část se zabývá UI implementací designu a vytvořením architektury podle návrhu. Pro ověření funkčnosti se pátá kapitola věnuje funkcionálnímu a uživatelskému testování. Poslední oddíl popisuje dokumentaci a analýzu kódu společně s nasazením aplikace.

Cíle práce

Hlavním cílem této práce je vytvořit aplikaci podporující základní činnosti sběratele turistických známek od firmy Turistické známky, s.r.o. Pro nejkvalitnější výsledek je třeba prostudovat aktuální problematiku a její různá řešení. Podle studie následně navrhnout vlastní řešení. Je pravděpodobné, že se aplikace bude časem rozrůstat o nové funkce. Primárním záměrem je tedy přijít s návrhem, který bude snadno rozšiřitelný. Posléze se bude návrh realizovat a testovat. Samozřejmostí je projekt průběžně dokumentovat.

Kapitola 1
Rešerše

Rešerše je rozdělena 3 na základní oddíly. V první části se nacházejí informace o technologiích a praktikách využitých při tvorbě designu. Ve druhé části jsou popsány designové vzory, ovšem z pohledu architektury a poslední část se zabývá využitými technologiemi při implementaci a jejich srovnáním s možnými alternativami.

1.1 Design

Design je fáze vývoje, ve níchž se z požadavků získaných při analýze vytváří vizuální stránka aplikace. V prvotní části této sekce jsou vysvětleny dva základní pohledy na design. V zápětí jsou rozebrány vzory, které byly použity při tvorbě designu a v samotném závěru sekce je provedeno srovnání editorů používajících se při vytváření designového návrhu.

1.1.1 UX vs UI

Spojením pevných kostí (kód), zdravých orgánů (UX) a příjemného vzhledu (UI) vznikne funkční, smysluplný a estetický organický stroj (web či aplikace)[1], to je jeden z pohledů na rozdělení vývoje aplikace. Hlavní myšlenkou je, že User Experience(UX) neboli uživatelská zkušenost se zabývá funkční stránkou designu a stará se o to, aby aplikace co nejintuitivněji plnila cíle uživatelů. UI se zabývá spíše estetickým dojmem. Dobrý UX design nejde vidět, ale jde cítit.

1.1.2 Heuristiky podle Jakoba Nielsena

Jedná se desatero principů pro tvoření uživatelského rozhraní. [2]

- 1. Zobrazení systémových statusů znamená, že pokud uživatel provede akci, vždy by měl být o provedené akci informován.
- 2. Spojení mezi reálným světem a systémem. Mělo by se využívat stejných slov, frází a konceptů spojených s tématem. Propojení s reálným světem umožňuje přirozenější pocit ze systému.
- 3. Uživatelská kontrola a svoboda. Základní myšlenkou je, že často dochází k nechtěným akcím a je vždy potřeba umožnit uživateli bezpečný návrat.
- 4. Konzistentnost a standardy. V aplikaci by mělo být vše na první pohled jasné a uživatel by neměl polemizovat o tom co, které funkce znamenají.

- 5. Prevence chyb. Zprávy o chybách jsou lepší než nezobrazovat žádné informace, ovšem lepším přístupem je chybám předejít.
- 6. Rozpoznání raději než znovuvolání. Uživatel by neměl být nucen pamatovat si důležité informace pro rozhraní z minulého rozhraní.
- 7. Flexibilita a efektivita. K funkcím by vždy měla vést nejkratší možná cesta, jinými slovy v případě nepotřebných kroků vytvořit cestu bez nich.
- 8. Estetika a minimalismus. Cílem je zobrazovat pouze důležité informace a nezatěžovat uživatele zbytečnými věcmi.
- 9. Nezobrazovat informace v kódovém jazyce.
- 10.Pomoc a dokumentace. Vysvětlit uživateli funkce a pomoci mu s porozuměním systému.

1.1.3 Designové vzory

Zde se nachází podsekce s vybranými designovými vzory, které mají sloužit ke zlepšení uživatelské zkušenosti neboli UX.[3]

1.1.3.1 Navigace

Hlavním prvkem, který uživatele provází celou aplikací je navigace v ní. Pokud se uživatel v aplikaci ztratí nebo mu nejčastější úkony zabírají nejvíce času, má to negativní vliv na UX. V aplikaci jsou využity a různě kombinovány následující vzory:

- SpringBoard - zobrazování navigačních elementů v pomyslné tabulce o několika sloupcích a řádcích
- List Menu - zobrazování navigačních položek v seznamu za sebou
- Tabs - umístění hlavních navigačních destinací na okraj obrazovky

1.1.3.2 Listy

Důležitou otázkou, na kterou nemusí být lehké odpovědět, je jak se budou uživatelovi reprezentovat data. Jedním z možných přístupů je zobrazovat data tabulkově. V této aplikaci jsou data reprezentována pomocí:

- Bezhlavičková tabulka - tabulka neobsahující hlavičku
- Tabulka se seskupenými řádky - tabulka je seskupena podle společného atributu
- Tabulka s vizuálními indikátory - jedná se o tabulku, která kromě textu disponuje i vizuálními prvky, které by měly uživateli přinést dodatečné informace

1.1.3.3 Filtrování,řazení a vyhledávání

V případě, že dat je více než lidský mozek zvládne vstřebat, je potřeba doplnit aplikaci o funkce, jež usnadní prohledávání dat. V této práci jsou použity způsoby vyhledávání podle názvu, dále filtrování a řazení. Pro vyhledávání podle názvu je vždy užité dynamické vyhledávání, které vrací položky okamžitě po zadání vyhledávaného textu. Pro řazení a filtrování lze využít vzor zobrazení kriterií přímo na obrazovce s položkami nebo zobrazení na samostatné obrazovce.

1.1.3.4 Nástroje

Zde jsou popsány vybrané funkce, které by uživateli měli usnadnit integraci s aplikací:

- Vložené akce - Jedná se vždy o viditelné akce, které jsou přiděleny k předmětům a které mohou rozšířit základní navigaci
- Více stavová tlačítka - tato tlačítka mohou měnit vzhled i funkčnost v závislosti na stavu
- Hromadná akce - akce, díky kterým lze provádět operace s více předměty současně

1.1.3.5 Dostupnost

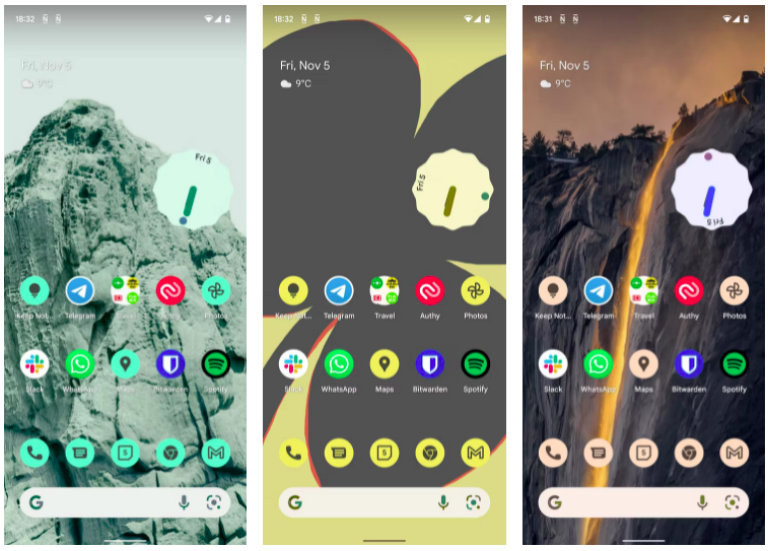
Uživatelé nejčastěji komunikují s aplikací pomocí kliknutí. Ovšem telefony nabízejí více. Dalším způsobem je drag. Jedná se o tažení a posouvání objektu, který je v uživatelově držení. Podobný dragu je flick, ovšem flick bývá instantní akce, jež následně objekt sama posune.

1.1.4 Material Design

Jedná se o designový systém založený společností Google. Material Design je seznam pokynů, komponent a nástrojů, které by měly podporovat nejlepší metodiky tvorby uživatelského rozhraní. Hlavní stavební jednotkou jsou komponenty symbolizující prvky z reálného světa, následně využívající barvy a typografii.[4] [5]

1.1.5 Material You a Material Design 3

Material Design 3 je technický název pro Material You. Material Design 3 je poslední verzí Material Designu. Hlavními změnami oproti Material Designu 2 jsou dynamické barvy a úpravy komponent. Podpora dynamických barev znamená, že vzhled aplikace se může měnit na základě obsahu, může jít o obsah aplikace nebo o obsah tapety na hlavní obrazovce.[6]



1.1.6 Material Theme Builder

Jedná se o technologii, která umí vyprodukovat téma aplikace v systému Material Design 3. Theme Builder umí vygenerovat paletu barev na základě obrázku nebo vybraných barev.

1.1.7 Srovnání editorů pro tvorbu designu

Na trhu je velké množství editorů, ve kterých lze tvořit design pro mobilní aplikace nebo pro aplikace obecně. Ovšem editorů, kterých lze napojit na Material Design 3 už tolik není. Google vytvořil Material Design Kit pro 3 vybrané, jedná se o Adobě XD, Figmu a Sketch.[7]

1.1.7.1 Adobe XD

Adobe XD je vektorový program, který 22.6.2022 stojí přesně 11.49 dolarů.

1.1.7.2 Sketch

Sketch je vektorový program zatím výhradně pro MAC OS, takže byl na začátku automaticky vyřazen.

1.1.7.3 Figma

Figma je vektorový program, který podle oficiálních zdrojů bude navždy zdarma, ovšem s limitem do 3 Figma souborů. Aplikaci lze designovat i v jednom souboru, tedy tento limit nebyl překážkou. Aplikace obsahuje i plugin Material Theme Builder. Z důvodů cenové dostupnosti, obsahu Material Theme Builderu a faktu, že lze pracovat jak v desktop verzi i v prohlížeči, se dále se využívá Figma.

1.2 Designové vzory a možnosti architektury

Christopher Alexander řekl, že každý vzor popisuje problém, který se objevuje znovu a znovu a pak popisuje jádro řešení k onomu problému takovým stylem, že můžete využívat řešení znovu a znovu bez toho abyste to dělali dvakrát stejným způsobem. Alexander mluvil o budovách a městech, ale vztahuje se to i na designové vzory objektově orientovaného programování.[8]

1.2.1 GoF

Jedná s o čtveřici autorů, kterým se často přezdívá Gang of four. Tato skupina autorů popsala 20 návrhových vzorů, které mají pomoci řešit obecné problémy v konkrétním kontextu. V následujících podsekcích jsou vybrány ty, které byly užity při implementaci.[8]

1.2.1.1 Creational Patterns

Hlavní myšlenkou návrhových vzorů je zajištění abstrakce způsobu tvorby jednotlivých objektů. Dále pomáhají odstranit závislosti mezi tím, jak jsou objekty vytvořeny, poskládány a reprezentovány.

1.2.1.1.1 Singleton Zajišťuje, aby třída mohla mít právě jednu instanci a také aby byl implementován globální přístup k ní.

1.2.1.1.2 Builder Jedná o pattern, jehož záměrem je rozdělit konstrukci a reprezentací. Často se řeší vytvořením samostatného objektu, který řeší úlohy konstrukce.

1.2.1.2 Structural Patterns

Tato skupina patternů skládá jednotlivé komponenty do větších struktur.

1.2.1.2.1 Facade Záměrem je tvorba komponent, které můžou sjednotit několik samostatných komplexních komponent v jednu zjednodušenou s konkrétním účelem.

1.2.1.2.2 Composite Hlavním cílem je tvorba stromových struktur. Objekty se označují jako potomky nadřazených rozhraní. Pomocí zaručeného rozhraní poté lze s objekty manipulovat jednotně.

1.2.1.3 Behavioral Patterns

Náplní sekce pro behavioral patterns jsou vzory, které se jsou spojeny s algoritmy a zadáváním odpovědností mezi objekty.

1.2.1.3.1 Observer Zajišťuje 1:N závislost mezi odesílatelem a příjemci. Dává možnost odesílateli notifikovat všechny příjemce.

1.2.1.3.2 Command Myšlenkou je zabalit požadavek do objektu, z důvodu potřeby volaní onoho požadavku bez znalostí podrobností o něm.

1.2.1.3.3 Strategy Umožňuje objektu využít vzájemně nahraditelné algoritmy. Strategie odstraňuje závislost algoritmu na zbytku systému.

1.2.1.3.4 Template Method Vytváří základní kostru algoritmu, ovšem některé kroky se nechávají na podtřídy. Umožňuje podtřídám předělat jisté části algoritmu.

1.2.2 Dependency injection

Jedna z definic dependency injection může znít takto: Je to skupina softwarových designových principů a vzorů, které umožňují tvořit komponenty, které nebudou vzájemně těsně spojené. Myšlenkou je programovací technika, která dělá třídu nezávislou na svých závislostech. Jako závislost lze použít rozhraní a posléze při inicializaci dosadit libovolnou implementaci rozhraní.[9]

1.2.3 Delegation pattern

Delegace umožňuje třídě nechat své rozhraní implementovat pomocí jiného objektu, jenž je podtřídou onoho rozhraní. Jinými slovy deleguje potřeby spojené s rozhraním na dodaný objekt. Využívá se k rozdělení tříd na menší části nebo umožňuje znovu užít delegované objekty. [10]

1.2.4 Model View ViewModel

Podle společnosti Google je doporučenou architekturou rozdělení systému na tři části a to uživatelské rozhraní, doménovou vrstvu a datovou vrstvu, kde je hlavní myšlenkou oddělení zájmů na vrstvy, které jsou spolu co nejméně provázány.[11] Model View ViewModel zkráceně MVVM těmto částem odpovídá. Model drží data, zatímco ViewModel je doménová logika nebo držitel stavu obrazovky. View je uživatelské rozhraní.[12] Základní průběh je takový, že ViewModel získá propojení s daty z datové vrstvy a produkuje k nim přístup pro uživatelské rozhraní jako svůj stav. Události, které se stanou v uživatelském rozhraní a souvisí s doménovou logikou obrazovky, jsou posléze přesměrovány do ViewModelu a jsou řešeny zde.[13] Datová vrstva se může rozdělovat na repozitáře s rozhraním potřebným k získávání dat a samotné komponenty, jenž data dodávají. Tímto způsobem lze abstrahovat způsob, jakým data získáváme.[14]

1.2.5 Data Access Object pattern

Data Access Object Pattern nebo-li DAO se využívá k oddělení nizkoúrovňového získávání dat od vysokoúrovňové doménové logiky aplikace.[15] DAO by se dalo rozdělit na 3 následující sekce:

- DAO rozhraní - jedná se o rozhraní pro operace, které mají být provedeny na modelech
- DAO konkrétní třída - implementace požadovaného rozhraní, která je zodpovědná za získání dat ze zdroje
- DAO model - jedná se o třídu, která slouží k reprezentování vkládaných a získaných dat

1.2.6 Reaktivní programování

Reaktivní programování popisuje designové paradigma, které spoléhá na asynchronní logiku, jenž bude řešit změny v reálném čase na jinak statickém obsahu. Základním konceptem jsou datové streamy, jenž se dají definovat jako časově seřazené sekvence notifikací o událostech. Záměrem je tvorba softwaru reagujícího na události spíše než systém odpovídající na uživatelův vstup. Jedná se o využití Observer patternu a funkce na zpracování příchozích dat.[16]

1.2.7 OAuth2.0

Open Authorization nebo-li OAuth je standard, který je určen pro webové stránky nebo aplikace umožňující přístup ke zdrojům hostovaným na serveru na požádání uživatele. Přináší odsouhlasený přístup a omezené akce, jež uživatel může použít bez sdílení konkrétních informací o uživateli. OAuth2.0 oproti 1.0 využívá přístupový token. Jedná se o informaci symbolizující uživatele, která ovšem s uživatelem nemá nic společného. Tento údaj může být časově limitován. Po uplynutí času již neni validní a nejdou s ním provádět žádné operace pod uživatelovým jménem. Pokud je přístupový token časově omezen, je k němu přidán i obnovující token, pomocí něj poté lze získat nový přístupový token. Pro lepší zabezpečení lze po úvodní autorizování zaslat kód s nutnou podmínkou jeho použití v dalším požadavku pro získání přístupového tokenu. [17]

1.3 Technologie

Jelikož je aplikace cílená pouze na android, nikoliv na více platforem. Využívá tedy nativní způsob implementace pomocí jazyku a funkcí, které mají přístup ke všem funkcím systému.[18]To, že jsou nativní aplikace optimalizovány pro konkrétní platformy přináší řadu výhod. Mezi hlavní patří výkon, bezpečnost nebo přístup.[19]

1.3.1 Framework

Framework by se dal nazvat jako skupina prosředků, které lze využívat k tvorbě systémů.[20] Mezi jeho hlavní výhody patří:

- Bezpečnější kód
- Jednodušší testování a debugování
- Vyhnutí se duplikování kódu
- Čistější a jednoduše adaptovatelný kód
- Možnost zaměřování se na více doménově specifický kód
- Rozšiřitelnost

1.3.2 Srovnání programovacích jazyků pro nativní aplikace na OS Android

Google udává, že aplikace pro operační systém Android lze psát pomocí Javy, Kotlinu a C++. Android Software Development Kit neboli SDK jsou nástroje, které pak aplikaci napsanou pomocí zmíněných jazyků překompilují do nativního jazyka. Jelikož C++ nelze vužít bez pomoci jiného z trojice, bude ze srovnání vyloučen.[21]

Kotlin byl ve spolupráci Jetbrains s Googlem vyvinutý proto, aby nahradil Javu, jakožto hlavní programovací jazyk pro Android. Když už byl ve vývoji nový programovací jazyk, vývojáři se rozhodli kromě přidání nových funkcí vyřešit problémy, které s sebou Java přínášela. Jednou z překážek byl tzv. Billion Dollar Mistake s null hodnotou. V Kotlinu se u každého typu dá určit, zda může obsahovat hodnotu null. V tomto případě se s ním zachází jinak. Dalším vylepšením je možnost využít funkce jako argument. To umožňuje tvorbu Higher-order funkcí. Naopak co Kotlin odstranil, jsou Kontrolované výjimky. Tedy v Kotlinu nemůže být propagována informace o možné výjimce. Malou, ale podstatnou výhodou je ukončení vynucování středníku. [22] Tím nejlepším na tom, že Kotlin vycházel z Javy, je jejich kompatibilita a možnost současného využití.[23] Na Google I/O 2019 bylo oznámeno, že hlavním jazykem, se kterým se bude počítat při vývoji softwaru, se stane Kotlin.

1.3.3 XML

Na Extensible Markup Language nebo-li XML může být pohlíženo jako na protokol pro uchovávání a spravování informací nebo jako na sadu nástrojů umožňující dělat všechno od filtrování dokumentů po filtrování dat, ale stejně tak na to lze pohlížet jako na filozofii pro zpracování informací.[24]

1.3.4 JSON

Zkratka JSON znamená JavaScript Object Notation a jde o způsob zobrazení dat, který má sloužit jako univerzální formát pro výměnu mezi zařízeními. Jedná se o textovou formu dat, jenž je nezávislá na programovacím jazyku, ovšem využívá konvence, které platí v rodině jazyků C. To z něj dělá ideální mezijazykový formát.

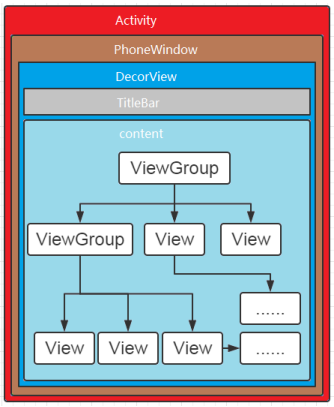
JSON může mít dvě následující podoby, buď se jedná o seřazený list hodnot nebo o kolekci párů jméno/hodnota.[25]

1.3.5 Srovnání nástrojů pro tvorbu nativního uživatelského rozhraní pro systém Android

Od vydání roku 2008 je možné na Android tvořit uživatelské rozhraní pomocí tzv. View systému. Ovšem v červenci roku 2021 vyšel Jetpack Compose, čímž se změnil přístup k tvorbě uživatelského rozhraní.

1.3.5.1 View systém

View systém je systém, ve kterém jsou Views nejmenší stavební kameny uživatelského rozhraní, skládány do ViewGroups neboli layoutů. To jak budou na obrazovce Views a ViewGroups rozvrženy závisí na xml souborech s informací o struktuře zobrazované komponenty. Základním vstupním bodem je aktivita, která dále instancuje jednotlivá Views podle přiloženého XML souboru. Jedná se o imperativní druh UI, protože nejprve musíme element získat i s jeho obsahem, poté modifikovat a následně aktualizovat ten samý element.[26]



Obrázek 1.1 View systém a aktivita

1.3.5.2 Jetpack Compose

Co je to vlastně Jetpack Compose? Je to moderní sada nástrojů pro tvorbu uživatelského rozhraní. Jetpack Compose je oproti View systému naopak deklaritvní styl uživatelského rozhraní, ve kterém je stav předdefinován a rozhraní se aktualizuje v případě změny hodnoty libovolného elementu. Myšlenka za Compose je vytvoření jedné aktivity pro celou aplikaci a tvorbu obrazovek i s navigací mezi nimi, poté už řešit pomocí Compose.[27]

Ovšem Compose nemá pouze kladné vlastnosti. Například rychlost studeného zapnutí aplikace je pořád příznivější pro View systém.[28] Ale View systém kompiluje to, jak bude obrazovka vypadat už při instalaci, zatímco Compose to dělá podle potřeby během aplikace. Tudiž studený start vychází lépe pro XML, avšak Android Runtime dokáže vyhodnotit následující obrazovky a umí si je předkompilovat. Tedy pokud už jsou obrazovky v paměti, Compose v jistých kategoriích předstihuje XML. Například procento zmrazených snímku se během testu změnšilo o více jak polovinu a čas načítání se před kompilováním zlepšil téměr o polovinu.[29]

1.3.6 Backend

Tato část aplikace je často označována jako serverová část, což ovšem není přesné pojmenování. Pro koncového uživatele může být neviditelná, přitom se jedná o životně důležitý orgán systému. Jejím hlavním úkolem je tvorba logiky projektu a manipulace s daty. Zmínil jsem, že pojem serverová část není přesný a to kvůli tomu, že pod pojem back-end může spadat i práce s databázemi a v mobilním vývoji se s databázemi pracuje i v zařízení uživatele. Příkladem backendu může být server zpracovávající vzdálené požadavky od uživatele, s nimiž následně provede operaci ku příkladu na databázi a vrátí výsledek.[30]

1.3.6.1 HTTP

Celým označením Hyper Text Transfer Protocol je základem pro World Wide Web, což je nejrozšířenější služba pro vyhledávání na internetu.[31] Jedná se o protokol, jehož posláním je

přesouvat data síti mezi zařízeními. Typickým průběhem je odeslání požadavku z klientského zařízení na server a vyčkání na odpověď.[32]

1.3.6.1.1 Požadavek HTTP požadavek se skládá z HTTP verze, URL, metody, hlavičky a je možné přidat i tělo. Metody existují 4 a jsou to GET, POST, PUT a DELETE. Hlavička obsahuje například informace o prohlížeči, jazyk nebo o jaký formát dat se jedná. V těle požadavku se nachází konkrétní informace o tom jaká data má server vrátit, například se může jednat o seznam filtrů aplikovaných na sbírky data. [32]

1.3.6.1.2 Odpověď HTTP odpověď se skládá pouze ze 3 částí. Jedná se o kód, hlavičku a volitelné tělo. Kód by se dal označit jako výsledek požavku, například úspěch nebo nedostatečné povolení. Hlavička plní podobnou funkci jako u požadavku a tělo je část odpovědi, kde se nachází požadavkem vyžádaná data, pokud jsou dodána.[32]

1.3.6.2 REST

Representational State Transfer ve zkrace REST je architektonický styl kladoucí si za cíl nastolit standardy pro počítačové systémy na webu, zajištěním jednodušší komunikace mezi zařízeními.[33] Hlavním rysem RESTu je jeho bezstavovost, to znamená, že server nepotřebuje vědět, v jakém stavu se klient nachází a obráceně. Podstatnou výhodou je odstranění závislostí mezi serverem a klientem, což umožní spolupráci aplikací podle obecného rozhraní a každá z nich poté může jít různou cestou implementace.[33]

1.3.6.2.1 REST API Aplication Programming Interface (API)je sada pravidel a protokolů určených k tvorbě a integraci softwaru.[34]

REST API nebo také známé jako RESTful API, je aplikační programovací rozhraní odpovídající architektonickému stylu REST.[34]

API se může nazývat RESTful pokud splňuje následující kategorie:

- Jednotné rozhraní - Znamená, že všechny požadavky na jeden zdroj by měly vypadat stejně. REST API by mělo zajistit, aby stejná požadovaná data vždy odpodídala pouze jednomu URI.
- Client-server separace - Jinými slovy tyto dvě části na sobě musí být kromě spojovacího rozhraní kompletně nezávislé, jediné přes co by měl klient se serverem komunikovat je URI. Žádná jiná interakce, která by mohla ovlivňovat stav systému není povolena, obráceně to platí stejně.
- Bezestavovost - Všechny informace potřebné k vrácení odpovědi na požadavek by měly obsahovat samotný požadavek, server by neměl ukládat žádná data o požadavcích.
- Cachování - Hlavním cílem této kategorie je umožnit cachování na klientské straně a škálovatelnost na straně serverové.
- Vrstvená architektura - Jedná se o generický architektonický styl, který by měl být použit při implementaci API, hlavní myšlenkou je rozdělení systému do vrstev, čímž se zvýší soudržnost a sníží provázanost.
- Kód na požádání - Většina vrácených dat je statická, odpovědi mohou obsahovat také spustitelný kód. V takovém případě by se měl kód spustit jenom na požádání.

1.3.7 SQL

SQL nebo-li Structured Query Language je programovací jazyk určený pro manipulaci s daty v relačních databázích. [35]

1.3.7.0.1 Databáze - organizovaná kolekce stukturovaných informací nebo dat [36]

1.3.7.0.2 Relační databáze Strukturuje data ve formě tabulek, kde jsou řádky a sloupce jednotlivé atributy a jejich vztahy mezi nimi.

SQL systém se dá rozdělit na 3 základní typy komponent, jedná se o:

- SQL Tabulka - tabulky s uloženými daty
- SQL Tvrzení - instrunkce, kterým systém rozumí, říkají co se má s daty stát
- Ukládáná procedůra - kolekce obsahující jedno nebo více SQL tvrzení, která se ukládají přímo do relační databáze

Kapitola 2
Analýza

Cílem této kapitoly je vytvořit potřebné požadavky a případy užití, ze kterých se následně bude dát vytvořit návrh a design aplikace.

2.1 Analýza webové stránky turisticke-znamky.cz

Turistické známky, s.r.o. jsou firmou, která dodává své produkty do více než 10 zemí Evropy. Při prohlížení jejich webu lze tedy vybrat hned z několika jazyků. Při přepnutí jazyka není stránka jen přeložena, ale má kompletně nový design. Tato analýza se zaměřuje na web pro Českou republiku. Tady je výpis hlavních funkcionalit nabízených stránkou:

- přihlášení a registrace
- vyhledávání v seznamu míst:
  - filtrovat podle kategorie místa, lokality, jména známky a podle novějších nebo nejnovějších
  - řadit lze podle čísla, názvu nebo kategorie vzestupně a sestupně
  - obsahuje paginaci
  - u každého známkového místa zobrazena fotka, název a prodejní místa
  - možnost prokliku na konkrétní známkové místo
- konkrétní známkové místo:
  - obsahuje název, popis, prodejní místa, přiblížená a oddálená mapa, poloha, hodnocení, fotografie
  - přidání:
    - názoru na dostupnost sběratelského předmětu v místě
    - komentáře k místu
  - přihlášený uživatel navíc může:
    - prohlížet a vyhledávat v komentářích
    - přidat hvězdičkové ohodnocení na stupnici od 0 do 3
    - přidat místo do své sbírky
- vyhledávání v mapě:
  - filtrovat podle druhu sběratelského objektu nebo podle stavu vyrábění

2.2 Analýza neoficiální aplikace Turistické známky

Neoficální aplikace má 2 varianty. Jedna je zdarma ke stažení a druhá stojí 30 korun českých. Tato sekce bude pojednávat o obou těchto verzích. Zde je výpis hlavních funckionalit:

- lze používat jako anonymní nebo přihlášený uživatel
- přihlášení pomocí zobrazení webové stránky turisticke-znamky.cz, na té samé stránce lze i registrovat a řešit zapomenuté heslo
- nelze prohledávat více sbírek zároveň
- vyhledávání v seznamu míst:
  - filtrovat podle kategorie místa, lokality, okresu, jména známky, stavu vydávání, uživatelova hodnocení, uživatelova statusu k místu
  - řazení předmětů podle čísla, názvu a vzdálenosti
  - u každého známkového místa zobrazen název, číslo a vzdálenost
  - možnost prokliku na konkrétní známkové místo
  - při rotaci displeje je možné na levé straně displeje vyhledávat a na pravé zobrazit detail místa
  - aplikace se při rotaci nebo při přepnutí na pozadí nezvládá vrátit na místo kde uživatel skončil
- import a export sbírky
- ruční nebo automatická oboustranná synchronizace sbírky s webem www.turisticke-znamky.cz přes oficiální API
- konkrétní známkové místo:
  - obsahuje název, popis, prodejní místa, GPS poloha, hodnocení, fotografie
  - funkce "BYL JSEM ZDE"(z detailu známky lze společnosti turisticke-znamky.cz zaslat důkaz o návštěvě místa, vhodné pokud na místě známku neměli nebo byla zavřená pokladna, funkce vyžaduje aktivní určování polohy pomocí GPS), poté lze rovnou z aplikace zaslat email s přehledem odeslaných důkazů a objednat chybějící předměty
  - GPS souřadnice lze zobrazit v externí aplikaci (podporované jsou: Mapy.cz, Google Maps, Sygic navigace, MapFactor navigace, Locus, OsmAnd a Google Earth)
  - možnost automatické aktualizace databáze na pozadí
  - žebříčky a úspěchy, které jsou implementovány jako zobrazení webové stánky v aplikaci
  - obrazovka servis
  - obrazovka o aplikaci
  - příhlášený uživatel navíc může:
    * přidat hvězdičkové ohodnocení na stupnici od 0 do 3
    * přiřadit si k místu jeden z 5 statusů
      - byl jsem zde a mám předmět
      - byl jsem zde a nemám předmět
      - nebyl jsem zde a mám předmět
      - nemám zájem získat předmět
      - plánuji získat předmět

- vyhledávání v mapě:
  - zobrazení známek a nálepek na mapce s barevným odlišením
  - stejné filtrování jako u seznamu míst + lze filtrovat podle data zveřejnění a data návštěvy
- na úvodní obrazovce jsou užitečné informace o počtu předmětů ve sbírce, naposledy uveřejněných známkách a o výročních akcích, které se konají v nejbližší době
- placená verze navíc nabízí:
  - možnost přidávat a editovat u předmětu textovou poznámku
  - zálohu sbírky a postupek na Google Drive
  - řazení předmětů podle data získání
  - jednoduchou statistiku získaných předmětů podle roků
  - upozornění na předměty v blízkém okolí

2.3 Analýza bakalářské práce na téma turistických známek

V roce 2013 byla vytvořena bakalářská práce na stejné téma od Elišky Kuzdasové.[37] Funkcionalita aplikace je následující:

- zobrazení známky na mapě
- přidání známky do sbírky
- odebrání známky ze sbírky
- import sbírky známek
- export sbírky známek
- zadání přihlašovacích údajů
- synchronizace sbírky známek
- možnost nahlášení neplatných odkazů
- statistiky sbírky
- upozornění na nové známky
- upozornění na sérii známek
- zálohování pomocí služby Android Backup

2.4 Tvorba dotazníku a analýza odpovědí

Z důvodu rozšíření informační základny o tématu turistických známek byl vytvořen dotazník, který měl za cíl získat informace od lidí, jak se jim líbí dosavadní aplikace a co by změnili. Celkem na otazník odpovědělo 56 lidí a byl distribuován do facebokových skupin.

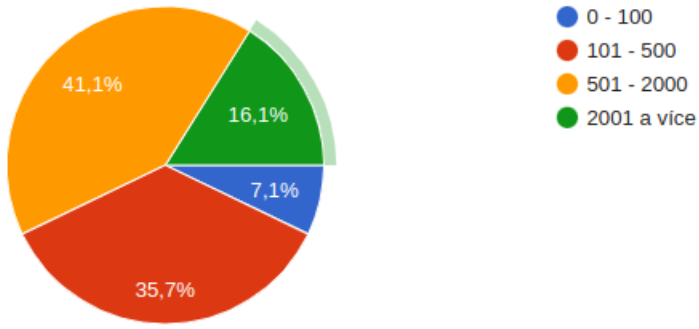
2.4.1 Tvorba dotazníku

První dvě otázky byly zaměřené na zjištění zkušeností odpovídajících uživatelů se sbíráním známek. Jednalo se o otázky zaměřené na dobu sbírání známek a na počet nasbíraných známek. 3. otázka měla za cíl zjistit různé motivace sběratelů pro sbírání známek. Následující 2 otázky pak měly odhalit chyby a špatná rozhodnutí při vývoji androidové a webové aplikace. 5. otázka pak měla umožnit sběratelům podělit se o jejich nápady na funkce do aplikace. Poslední otázkou bylo, s jakým operačním systémem by uživatelé aplikaci používali.

2.4.1.1 Analýza odpovědí na zkušenosti

2.4.1.1.1 1.otázka - "Kolik let sbíráte známky?" Z odpovědí vyplynulo, že 75% odpovídajících, známky sbírá už více než 5 let, tedy většina dotázaných už má poměrně velké zkušenosti se sbíráním předmětů od Turistických známek, s.r.o.

2.4.1.1.2 2. otázka - "Kolik máte nasbíraných známek?" Podle dotázníku má 92.9% lidí aspoň 100 známek, to jenom potvrzuje fakt, že mezi dotázanými není moc nezkušených sběratelů. Díky tomu, že dotázaní jsou z většiny vášniví sběratelé, lze na jejich názory nahlížet jako na problémy, které nejsou pouze teoretické, ale jsou to problémy, se kterými se setkávají. Ovšem to, že se odpovědí zúčastnilo tak málo nezkušených uživatelů znamená, že se mohlo sesbírat málo užitečných informací o funkcích pro méně nadšené sběratele.



Obrázek 2.1 Odpovědi na 2. otázku z dotazníku

2.4.1.2 Analýza odpovědí na motivaci

2.4.1.2.1 3. otázka - "Proč sbíráte známky?" Tato otázka pomohla odhalit hlavní záměry sběratelů. Mezi nejčastějšími odpověďmi bylo objevování míst, tvorba zážitků, sbírání suvenýrů a tvorba vzpomínek. Vyplývá tedy, že hlavní úkoly pro aplikaci jsou objedovávní míst a ukládání informací o výletech, zážitcích a předmětech.

2.4.1.3 Analýza odpovědí na aktuální plusy a minusy

Cílem těchto otázek bylo zjistit, které funkce v aktuálních systémech fungují a uživatelé by je chtěli využívat i nadále a také zjistit, co jim na současných funkcích vadí.

2.4.1.3.1 4.otázka "Používáte webovou stránku turistických známek? Podělte se s námi o důvod, proč ji využíváte nebo naopak nevyužíváte." Mezi mínusy tohoto dotázaní moc důvodů neuvedli a jediné co zmínili bylo, že stránka je chaotická a mapa nepříjemná. Ovšem zmínili hodně důvodů, proč ji využívají. Mezi hlavními byl přehled ve známkách, tip na cestování, plánování a přehled o aktuálních informacích.

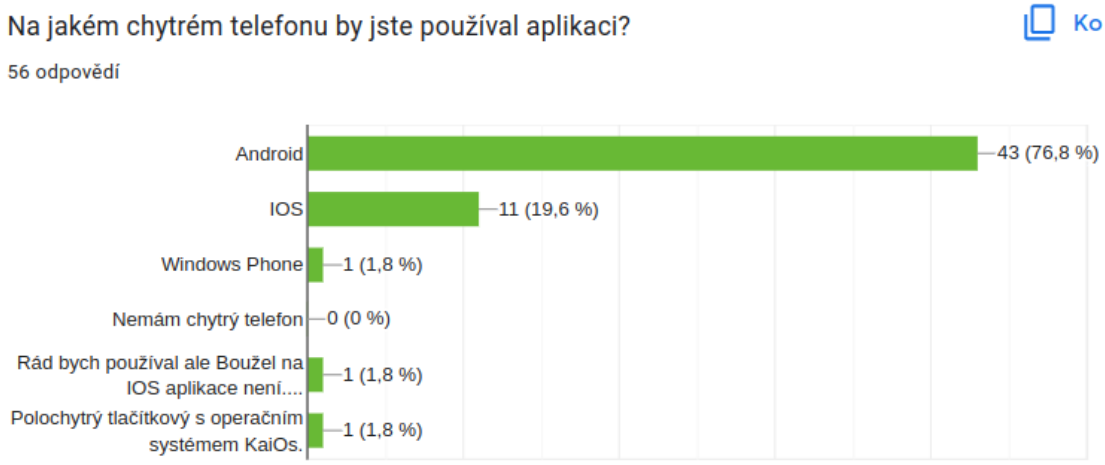
2.4.1.3.2 5.otázka "Pokud využíváte android aplikaci pro sběratele turistických známek, co se vám na ní nelíbí?" Na tuto otázku už odpovědělo pouze 18 lidí, ale zde už mínusů bylo více. Mezi nejčastější patří špatné aktualizace dat a nekompletní data. Další byly zmínky o špatné mapě, chybějících komentářích a skrytých funkcích. Poslední velkou chybou je ztráta uživatelova postupu například při rotaci displeje nebo při navigaci dopředu a zpět.

2.4.1.4 Analýza nápadů

2.4.1.4.1 6.otázka "Co všechno by podle vás aplikace měla umět?" V této kategorii dostali uživatelé možnost být kreativní a přijít s nápady na funkcionality aplikace. Hodně odpovědí zahrnovalo využití Mapy.cz, ovšem z důvodu nedostupnosti SDK tato možnost byla zavrhnuta. Dále zde bylo několik odpovědí, které nejde realizovat z důvodů limitů API, jako uložení různých verzí známek. Dále byly některé nápady, které byly využity. Jedná se o hledání v okolí, tvorba plánů, zobrazení náhledu místa na mapě a rovnou upravovat status k místu z mapy. Také se dotázaní zmínili o filtrování dat a hodnocení míst.

2.4.1.5 Analýza odpovědí na OS

2.4.1.5.1 7.otázka - "Na jakém chytrém telefonu by jste používal aplikaci?" Z odpovědí vyplynulo, že 76% uživatelů by aplikaci používalo na operačním systému Android.



2.5 Analýza dokumentace

Turistické známky zaslaly dokumentaci k jejich API, zde se nachází rozbor důležitých poznatků, které musí být při návrhu a implementaci zohledněny.

2.5.1 Kolekce předmětů

Pro získání obecných sbírek není potřeba být přihlášen a každá kolekce má svůj endpoint, který při zavolání vrátí kolekci zabalenou v XML.

2.5.2 Autorizace

Turistické stránky k autorizaci využívájí standard OAuth2.0. Díky ní pak lze mazat, získávat a ukládat na server předměty jako vlastněné, navštívené a lze k nim přidat komentář. Pro autorizaci uživatele je potřeba udělat několik úkonů. Turistické známky mají autorizaci vyřešenou poměrně netradičně, údaje o uživatelích si zpracovávají sami. To znamená, že pro přihlášení mají turistické známky svůj endpoint, rozšířený o tajný identifikátor, jenž byl vygenerován přesně pro tuto aplikaci. Netradiční je proto, že na endpoint se neposílají data, ale endpoint je zobrazen jako webová stránka v aplikaci. Uživatel se na webové stránce přihlásí a v případě úspěchu server na stránce zobrazí kód, jenž je potřeba zpracovat. Následně se pomocí získaného kódu, tajného identifikátoru a tajného klíče může požádat o přístupový token. V případě úspěchu jsou pvráceny informace o uživateli plus přístupový a obnovující token.

2.6 Požadavky

Podle existujících aplikací, oficiální dokumentace a odpovědí z dotazníků byly vypracovány následující požadavky.

Tabulka 2.1 Funkční požadavky

Požadavek	Popis	Nadřazený požadavek
FP1 Přihlášení a odhlášení	Zobrazit uživatelovi oficiální stánku turisticike-znamky.cz pro přihlášení, na stejné stránce se lze i registovat.	
FP2 Spravování plánů	Vytváření, mazání, editace a vyhledávání plánů.	
FP3 Spravovat oprávnění	Spravovat oprávnění k poloze zařízení a pro přístup k souborům.	
FP4 Vyhledávání míst	Umožnit vyhledávání turistických míst a předmětů.	
FP5 Filtrování dat	Umožnit filtrování míst podle uživatelova zadaného textu, kategorie míst, kategorie předmětů, národnosti sbírky, okresu a lokace. Dále podle toho jestli je místo navštívené. jestli je předmět jako vlastněný a podle toho jestli je předmět v plánu.	FP4

FP6 Filtrování míst podle vzdálenosti	Umožnit uživatelovi při zapnutém oprávnění k poloze také filtrování podle vzdálenosti	
FP7 Řazení míst	Umožnit uživatelovi vzestupně i sestupně řadit místa podle ID, názvu, datumu komentáře a datumu navštívení	FP4
FP8 Řazení míst podle vzdálenosti	Umožnit uživatelovi při zapnutém oprávnění k poloze také řazení míst podle vzdálenosti od zařízení	
FP9 Vyhledávání v listu	Umožnit uživatelovi prohledávat místa v rolovatelném seznamu.	FP4
FP10 Vyhledávání v mapě	Umožnit uživatelovi prohledávat místa v mapě.	FP4
FP11 Poloha zařízení na mapě	Umožnit uživatelovi při zapnutém oprávnění k poloze na mapě zobrazit jeho polohu.	
FP12 Vzdálenost u míst	Umožnit uživatelovi při zapnutém oprávnění k poloze u míst zobrazit jejich vzádelnost k zařízení.	
FP13 Zobrazení jednotlivých předmětů	Systém by měl umožnit zobrazit detail, na kterém by měly být všechny informace o předmětu. Dále by odsud mělo být umožněno přesunout se na místo v extérní mapě, v zabudované mapě, také možnost hodnotit místo a ovládat uživatelův status k němu.	
FP14 Souhrn uživatelových sbírek a plánů	Zobrazit uživatelovy vybrané sbírky a jejich postup.	
FP15 Aktuální informace	Zobrazení webové stránky https://www.turisticke-znamky.cz/aktualni-informace.html	
FP16 Nejnovější známky z vybraných sbírek	Seskupit sbírky, které má uživatel mezi vybranými a zobrazit od nich nejnovější známky.	
FP17 Žebříček sběratelů	Zobrazení webové stránky vhttps://www.turisticke-znamky.cz/zebricek.html	
FP18 Hodnocení míst	Systém by měl umožňovat hodnotit navštívená místa	
FP19 Spravování vlastnictví předmětu	Umožnit spravovat informaci o tom jestli uživatel vlastní předmět nebo ne.	
FP20 Spravování navštívenosti místa	Umožnit spravovat informaci o tom jestli uživatel navštívil místo nebo ne, přídávání je možné pouze pokud je uživatel v dostatečné vzdálenosti od místa.	

FP21 Spravování komentářů k předmětům	V případě vlastnictví předmětu je možné přidat komentář.	
FP22 Synchronizace	Synchronizace vlastněných předmětů, komentářů a navštívených míst se serverem	
FP23 Export a import pro autorizovaného uživatele	Přihlášeným uživatelům umožnit kompletní export a import pouze plánů	
FP24 Export a import pro anonymního uživatele	Anonymní uživatelům umožnit kompletní export a kompletní import	

Tabulka 2.2 Nefunkční požadavky

Požadavek	Popis	Nadřazený požadavek
NFP1 Offline mód	Vyhledávání by po stažení všech potřebných informací mělo fungovat bez potřebného dalšího internetového připojení.	
NFP2 Bez ukončení aplikace neztrácet uživatelův postup	V případě rotace zařízení nebo pokud uživatel přesune aplikaci na pozadí, se aplikace nesmí vrátit do základního stavu a musí pokračovat tam kde uživatel skončil.	
NFP3 Autorizace	Umožnit přihlášeným uživatelům serverové operace jako přidávání známek do sbírek a míst do uložených.	
NFP4 Spodní navigace	Navigace mezi hlavními obrazovkami pomocí spodní navigace.	
NFP5 Náhled místa	Systém by měl umožnit se díky jednomu kliknutí dostat z mapy na náhled místa a zároveň zůstat v mapě.	

2.7 Porovnání mapových poskytovatelů

Z požadavku FP7 vyplinulo, že v aplikace bude potřeba zobrazit mapu. Na trhu je k dispozici několik provozovatelů, kteří mají vytvořený jejich SDK. Mapy, které mají webové stránky by šly přes integrované webové okno, ovšem to by odporovalo s politikou offline first. Kvůli velkému počtu zobrazovaných známek, je potřeba zavést tkz. clusterování, jiný slovy seskupování jednotlivých značek na mapě do skupin.[38] Bude potřeba aby mapy tuto funkci podporovaly.

2.7.1 Mapy.cz

Podle oficiálních zdrojů zatím není SDK dostupný a lze pouze špouštět extérní nainstalovanou aplikaci.[39]

2.7.2 MapBox

Pro Android je k dispozi SDK umožňující zdarma využívání do 25000 měsícně aktivních uživatelů. V případě, že se limit přesáhne platí se poté nejvíce o 4$ za tisíc uživatelů.[40] Avšak podle dokumentace zde neexistuje funkce seskupování značek do skupin.[41]

2.7.3 OpenMaps

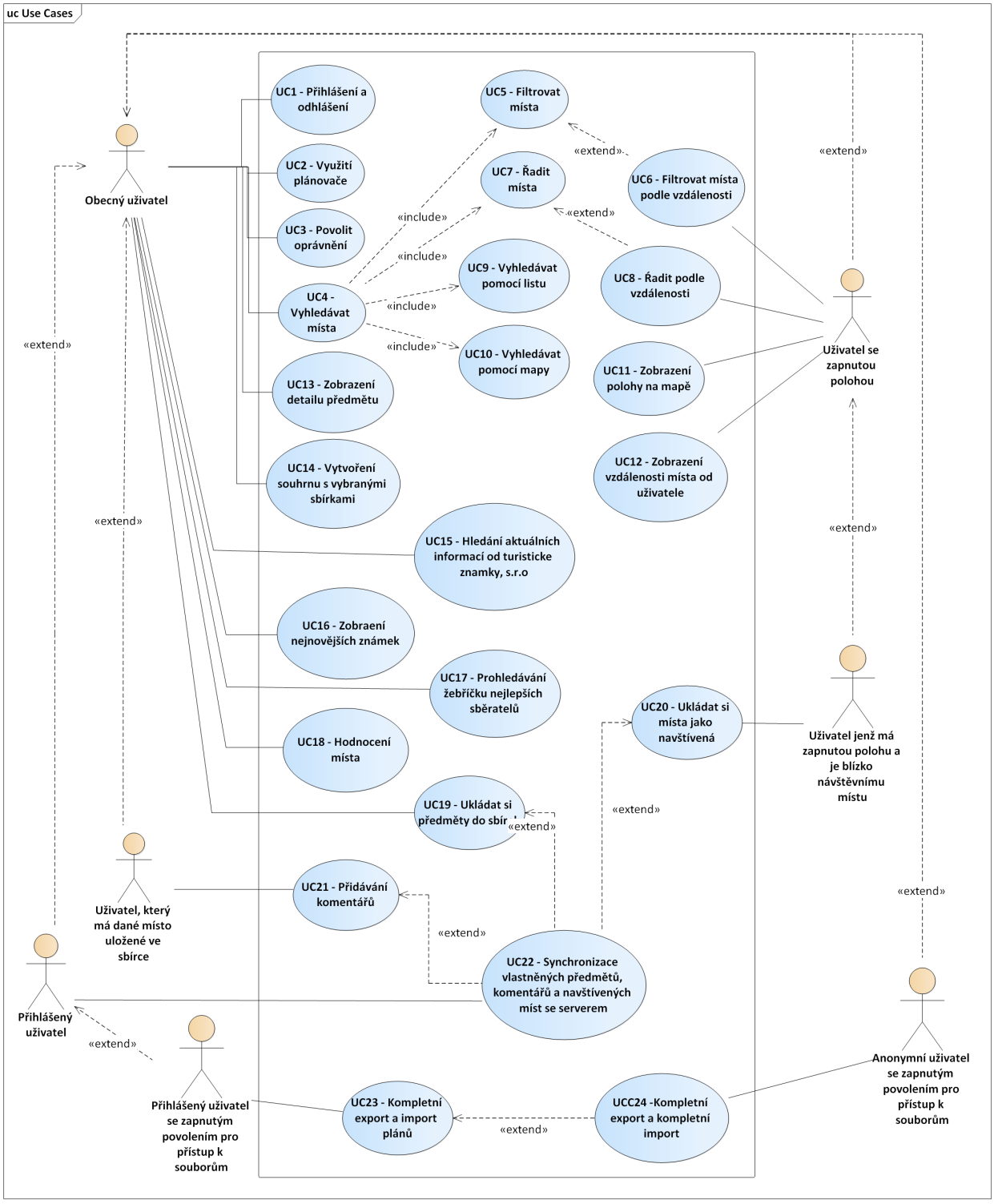
OpenMaps je open source projekt cílící na dodání nového mapového SDK. Využití onoho projektu by tedy nic nestálo, ovšem zase je zde problém absence možnosti clusterování. [42]

2.7.4 Google maps

Google maps má stejně jako MapBox k dispozici kromě různých API i SDK pro Android. Google mírně podporuje mobilní vývoj a jejich mapy jdou tedy při správném nastavení využívat na mobilních zařízeních i zdarma.[43] Pravděpodobně největší výhodou map od Googlu je ale jejich SDK, ve kterém se nacházejí i clusterovací pomocné nástroje.[44]

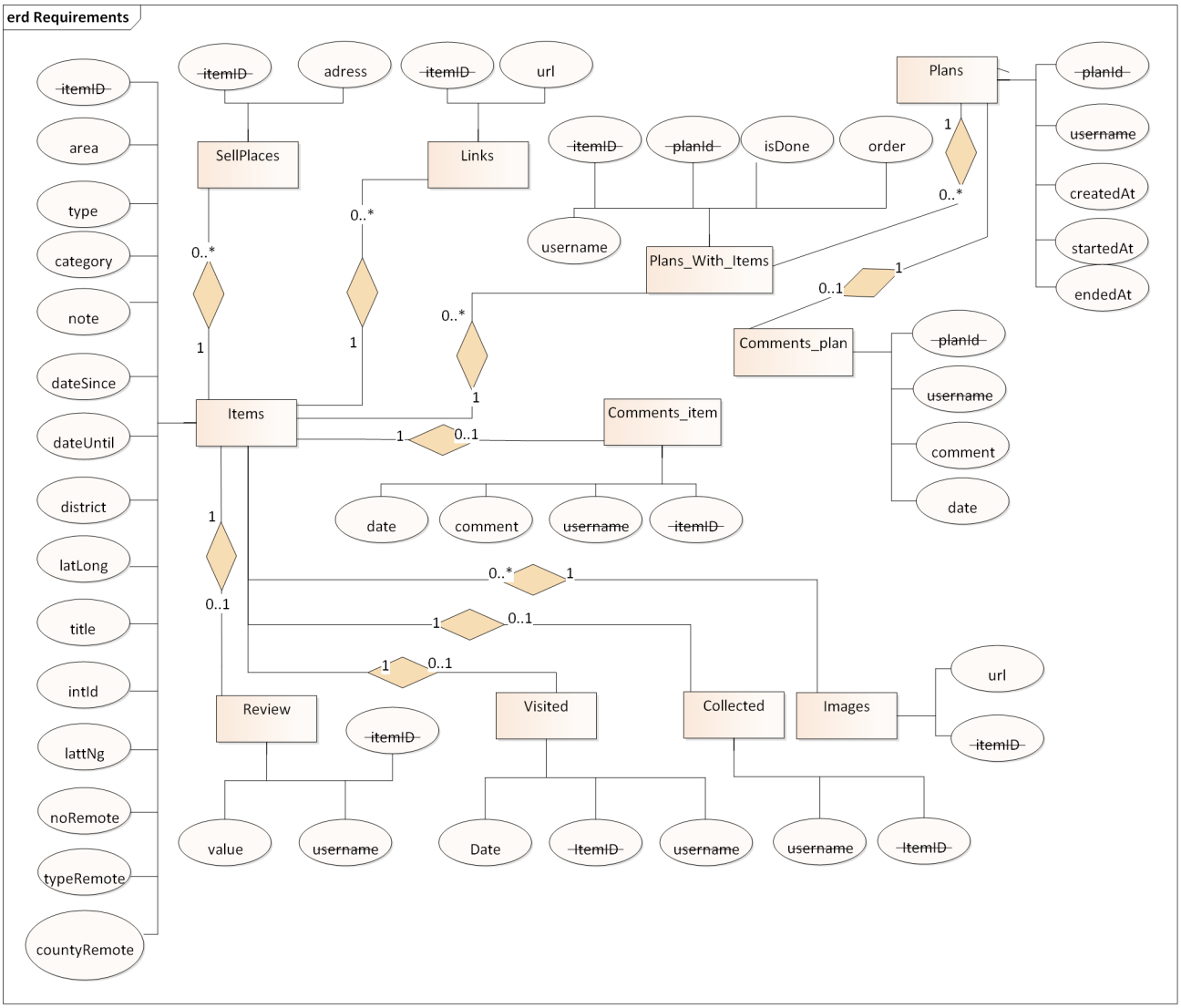
2.8 Případy užití

Vyzobrazení formou Use Case diagramu.



2.9 Databázový návrh

Návrh je formou ER diagramu, neboli Entity Relationship diagramu.



2.10 REST klient

API turistických známek funguje na protokolu HTTP. Vznikla tedy potřeba vytvořit HTTP klienta. Pro možnost tvorby vzdálené komunikace je na Android k dispozici několik HTTP klientů a některé znich přichází ve frameworcích. Mezi ty nejrozšířenější patří Retrofit a Ktor.[45] Ovšem co je rozlišuje? Retrofit je typově bezpečný HTTP klient pro Android a Javu.[46] Ktor je framework určený k tvorbě propojených asynchronních aplikací. Umožňuje implementovat webové aplikace, http služby, mobilní a prohlížečové aplikace.[47] Jako http klient byl z důvodu nativity vybrán ktor. Ne všechno jde vždy podle plánu, to je rozepsáno v části 4.2.2.2.

2.11 Porovnání SQL databází pro Android

Aby aplikace mohla fungovat v offline režimu je potřeba si informace ukládat, k tomu účelu slouží databáze. Z důvodu velkého počtu známek potřebných k uložení a také kvůli velké provázanosti

například mezi předměty a plány se hodí SQL.[48] K dispozici je několik implementací SQL databáze. Zde je popis 3 nejznámějších pro Android.

2.11.1 SQLite

První je SQLite, což je cross-platformí databázový engine, nejpoužívanější na světě.[49] Využívá souborově založený systém, takže je možné jej používat cross-platform. Ovšem neobsahuje možnost spuštění automatických notifikací při změně dat v databázi. Dále je potřeba vždy převádět manuálně jednotlivé objekty na řádkový formát. [48]

2.11.2 ORMLite

Druhou variantou není úplně databáze, ale jedná se spíše o obal okolo sql databází pro mobilní zařízení. ORMLite zjednodušuje způsob dotazování se svým Buildrem pro dotazy, dále je také schopný využít výhod Data Access Objects, konkrétně přivádí základ pro implementaci DAO rozhraní. Nabízí několik užitečných funkcí k dotazování dat, např. ”dotaz podle identifikátoru”. Ovšem data se znovu konvertují na řádkový formát i když automaticky.[48]

2.11.3 Jetpack Room

Třetí variantou je Jetpack Room, což je perzistenční knihovna od společnosti Google. Nabízí abstrakční vrstvu nad SQLite při zachování síly enginu.[50] Umožnuje třídám, označeným jako databázové, tvořit databázi a být hlavním přístupovým bodem pro data.[48] Princip práce s Room je vytvoření DAO rozhraní, pro které je pak automaticky vygenerována implementace. V rozhraní je pro funkce pomocí anotace specifikován sql požadavek a jako výstupní parametr se nastaví nějaká z DAO Model tříd. Dále je možné využívat parametry funkce k dynamickému dotazování na základě hodnot.[51] Poslední zmíněnou výhodou Room knihovny je implementovaný notifikační systém při změně dat. Jedná se o zabalení DAO Modelu do objektu, který je vytvořen podle Observer patternu.[52]

2.12 Porovnání DI knihoven

Pro ulehčení spravování závislosti a díky dalším výhodám DI Patternu, zmíněným výše, dochází k rozhodnutí o využití onoho patternu. Na OS Android jsou k dispozici 2 hlavní kompetitoři, jedná se Koin a Jetpack Hilt. I když provádí stejný úkol, daly by se nazvat jako protějšky.[53]

2.12.1 Koin

Koin je napsaný čistě v Kotlinu, je tedy nativní pro Kotlin a lze jej spustit na všech kompatibilních zařízeních. Tohle z Koinu dělá multiplatformní framework. Dále se jedná o závislost, která se zpracovává za běhu, to znamená že nelze zajistit bezpečnost v čase kompilování, ale také že se negenerují žádné pomocné soubory při kompilaci. Koin jde navíc cestou ServiceLocator patternu, který podle Googlu funguje na základě toho, že třídy si získávají své závislosti na místo přímého vytváření objektů se závislostmi. Někteří lidé tedy namítají, že se nejedná o DI.[53]

2.12.2 Jetpack Hilt

Hilt je knihovna, která jde naopak oproti Koin cestou tvorby závislotí už při kompilaci a v případě nedodané závislosti se o chybě ví už v tu danou chvíli. Ovšem na druhou stranu se při kompilaci generují dodatečné soubory a aplikace se tím zvětšuje.[53] Hilt je nadstavbou

od Google na Dagger 2 knihovně, která je nyní udržována stejnojmennou společností.[54] Díky podpoře Googlu pro implementaci využívá Hilt.

2.13 Porovnání knihoven umožňující reaktivní programování

V případě provádění asynchronních operací je potřeba nástrojů, umožnujících část kódu spustit současně s jinou. Pro Android jsou dvě nejznámější knihovny RxJava a Coroutines.[55]

2.13.1 RxJava

Rx značí Reactive Extensions. Jedná se tedy o rozšíření založené na jazyce Java, jenž mimo jiné umožňuje deklarativní operace převést na asynchronní streamy dat.[56]

2.13.2 Coroutines

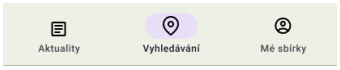
S příchodem Kotlinu došlo k zavedení nového klíčového slova a to ”suspend”, které slouží jako označení funkcí trvající delší dobu a pro které vzniká potřeba spouštět je asynchronně. Coroutine je instance suspendovatelného výpočtu. Coroutines jsou ve své podstatě podobná jako multithreading, s tím rozdílem, že Coroutine není vázána k žádnému konkrétnímu vláknu.[57]

Suspend funkce může asynchronně vrátit hodnotu, ale pro případ opakovaného získávání hodnot je potřeba rozšiřující technologie. K tomu existuje Flow, jenž umožňuje emitovat hodnoty a také je kolektovat. Jinými slovy tedy jde na instanci flow přidat kolektovací funkci spouštějící se kdykoliv se do flow dostane oznámení o nové hodnotě.[58] Flow je postředek k využití Observer patternu.

Coroutines jsou na tom výkonově lépe, protože zvládájí dělat stejné úkoly s méně prostředky za rychlejší čas, také využívají méně operační paměti a vyvíjí kratší zátěž na procesor. To nám tedy dává dohromady, že Coroutines jsou méně náročnější na baterii.[56] Z těchto důvodů se využívá Coroutines.

Kapitola 3
Design

Podle NP4 je hlavní navigace vyřešena pomocí Spodní navigace, pro splnění požadavku lze využít designový vzor Tabs. Základní restrikce Material Designu doporučují omezení na tři až pět hlavních destinací.[59]



Obrázek 3.1 Spodní navigace

3.1 Hlavní destinace

Z důvodu zachování jednoduchosti a pro případ možného rozšíření při důležité změně se využávají 3 hlavní destinace, více jich zatím není potřeba. Pro udržení co největší soudržnosti jsou jako hlavní destinace vybrány Informace, Objevování a Profil. V případě, že se uživatel nachází na jedné z hlavních destinací, tak se vždy zobrazuje spodní navígační panel umožňující změnit destinaci na jinou z hlavních.

3.1.1 Informace

Destinace Informace je zaměřena na zobrazení dodatečných informací o Turistických známkách, s.r.o. V destinaci se nachází horní navigace tvořící destinaci Informace nadřazenou pro Aktuální informace, Žebříček sběratelů a Nejnovější známky. Z toho vyplývá, že Informace pokrývají požadavky FP15, FP16, FP17. Aktuální informace a žebříček sběratelů, jsou pouze webové stránky zobrazené v aplikaci, bez nutnosti využití prohlížeče. Nejnovější známky jsou pak zobrazeny podle 1.1.3.2, kde v hlavičce je název kolekce a u jednotlivých předmětů je následně zobrazeno ID a Název, jedná se o rozšiřitelnou komponentu, která při kliknutí zobrazí tlačítka funkcí spojených s předmětem. Konkrétně jde o navigaci na detail, zobrazení plánovacího dialogu a v případě existence pozice u předmětu i možnost zobrazení v mapě.

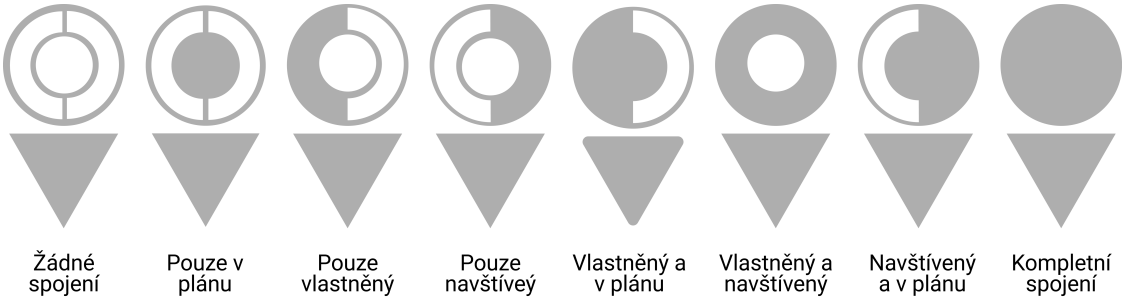
3.1.2 Objevování

Destinace slouží jako základní vyhledávácí bod. Je možné prohledávání v mapě nebo zobrazení překrývající vrstvu obsahující seznam s předměty, filtr a řazení.

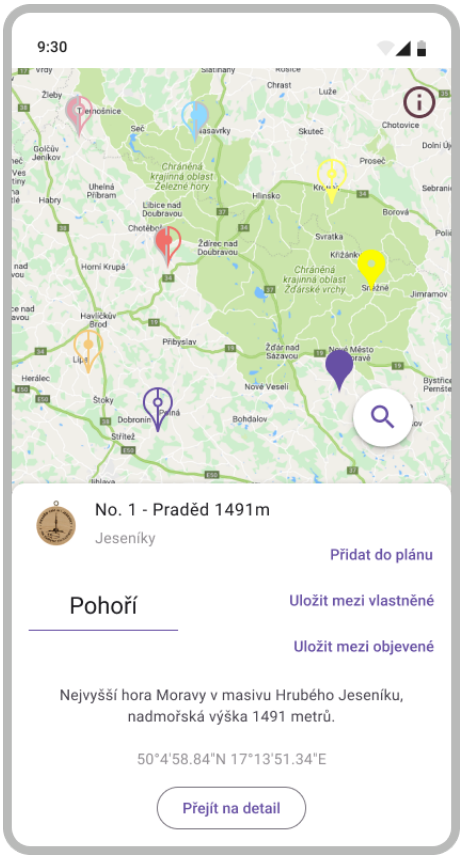
3.1.2.1 Mapa

Pro splnění požadavku NP5 musí u mapy být i komponenta se základními informacemi o předmětu zobrazující se pří kliknutí na značku v mapě, jedná se o náhled místa.

3.1.2.1.1 Značky na mapě Aby se z mapy dalo vyčíst co nejvíce informací bez nutnosti prokliku, jsou vytvořeny speciální značky. Nadesignováno je 8 značek, kde každá značka v sobě nese informaci o tom, jestli je předmět vlastněný, navštívený a zda se nachází v plánu. Tedy tvar značek vypovídá o tom, v jakém je předmět stavu. Dalším atributem značky je barva. Zde bylo čerpáno inspirace přímo od Turistických známek, s.r.o, konkrétně z jejich webu, kde barvou označují kategorie jednotlivých míst. Systém značek pracuje s 8 značkami a 8 barvami, tak tedy pomáhá předměty rozdělit na 64 skupin.



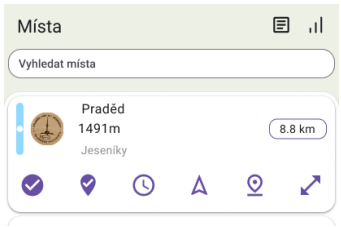
3.1.2.1.2 Náhled místa Komponenta slouží pro zobrazení funkcí a základních informací spojených s konkrétním vybraným předmětem. Při zobrazení náhledu se mapa zmenší a uvolněné místo využije náhled, který malou částí překryje mapu. Horní rohy komponenty jsou zaoblené a díky překrytí se vytváří plynulejší přechod mezi mapou a náhledem. Náhled je při využití gesta možné skrýt. Mezi informacemi se nachází ID, Název, Lokalita, Kategorie, Popis a Poloha a lze přejít na detail a upravovat statusy k předmětu.

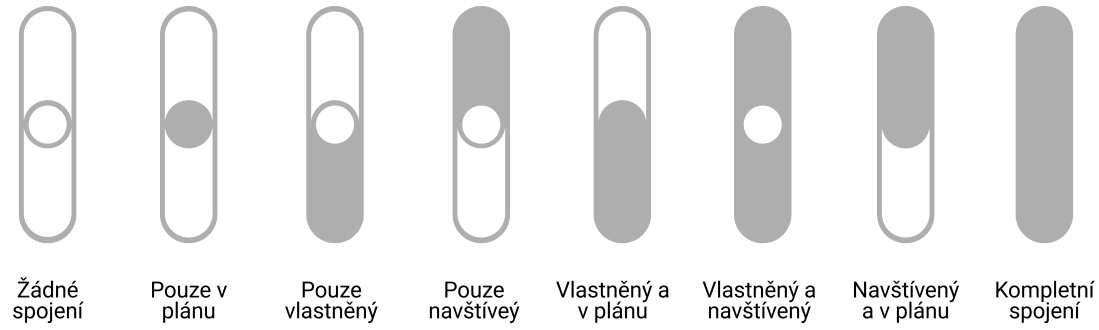


3.1.2.2 Vyhledávací vrstva

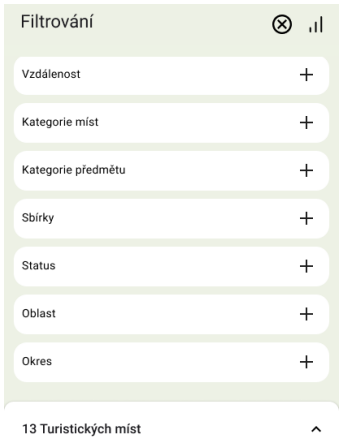
Podle požadavků FP5 až FP9 je potřeba zobrazovat seznamy předmětů, umožnit filtrování a řazení předmětů. Pro tento účel existuje vyhledávací komponenta. Jedná se o 3 stavovou obrazovku umožňujicí přepínání mezi výše zmíněným seznamem, filtrováním a řazením. Tento výběr je možné provést v nástrojovém pruhu, který je umístěn na vrcholu obrazovky. Dále se v pruhu nachází název sekce.

3.1.2.2.1 Místa Nejedná se o pouhý seznam s předměty, ale je možné odsud i filtrovat místa podle názvu. Vstupní pole filtrování podle názvu je mezi nástroji a seznamem. U každého předmětu je zobrazena fotka, název, lokalita a vzdálenost. Dále jsou pro zobrazení statusů vytvořeny speciální vektorové indikátory reflektující značky na mapě, to odpovídá designovému vzoru tabulce s vizuálními indikátory1.1.3.2. Při rozkliknutí předmětu lze vidět, že se zobrazila tlačítka spojená s předmětem, jedná se o upravování statusu k vlastnictví, zobrazení plánovacího dialogu a navigace k detailu. Pokud má předmět i GPS pozici, je možné jej přidávat do navštívených, zobrazit v zabudované mapě a zobrazit v extérních mapách. Tlačítka upravující status Vlastněný a Navštívený s sebou navíc nesou informaci o stavu uživatelova statusu, jde tedy o stavová tlačítka 1.1.3.4. Seznam míst je rolovatelný a využívá se u něj gesta flick.1.1.3.5

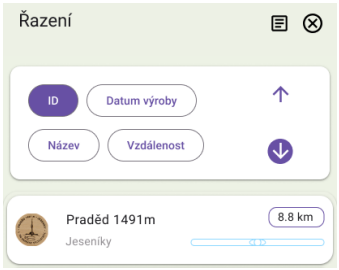




3.1.2.2.2 Filtrování Pro velký počet kategorií je fitlrování kromě názvu řešeno na samostatné obrazovce. Data lze filtrovat podle Vzdálenosti, Kategorie předmětu, Kategorie míst, Statusu, Oblasti a Okresu. Všechny tyto kategorie jsou rozkliknutelné a umožňují vybrat požadavané atributy. Ve spodní části se nachází počet předmětů, jenž prošly filtrem.



3.1.2.2.3 Řazení Pro řazení se z důvodu nízkého počtu kategorií využívá takzvané řazení na obrazovce, to znamená, že v případě zvolení řazení je možné pod řazením vidět i seřazené předměty. Je umožněno řazení podle ID, Názvu, Datumu návštěvy a vzdálenosti. Veškeré zmíněné způsoby řazení lze aplikovat vzestupně i sestupně.

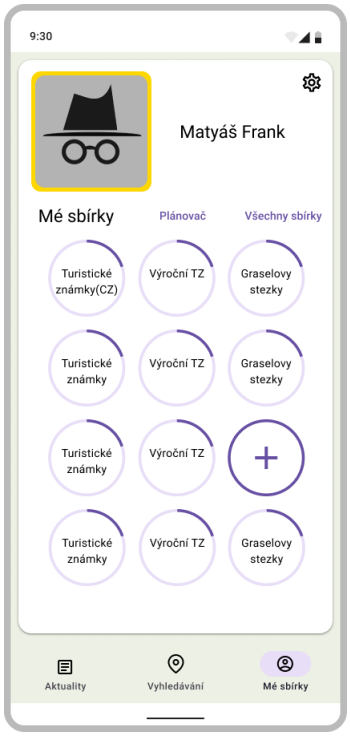


3.1.2.3 Pomocná sekce

Aby se uživatelé v indikátorech a barvách vyznali, mohou využít pomocné informační komponenty, která je rozdělana pomocí Tabs na 3 sekce obsahující barvy, značky na mapě a značky v seznamu. Komponenta splňuje 10. princip od Nielsena.

3.1.3 Profil

Destinace umí zobrazit uživatelovo jméno, obrázek, možnost navigace do plánovače, možnost navigace na výběr stáhnutých kolekcí a přehled stáhnutých kolekcí. Pro lepší informovanost se u kolekcí zobrazí indikátor počtu vlastněných předmětů. Material Design 3 přináší 2 typy indikátorů. První je lineární a druhý je kruhovitý.[60] Aby se na obrazovku vešlo co nejvíce kolekcí a zároveň se zachovala přehlednost, pro zobrazení kolekcí je použitý SpringBoard1.1.3.1, kde jeden předmět je kruhovitá komponenta s názvem kolekce, překrytá kruhovým indikátorem.



3.2 Samostatné destinace

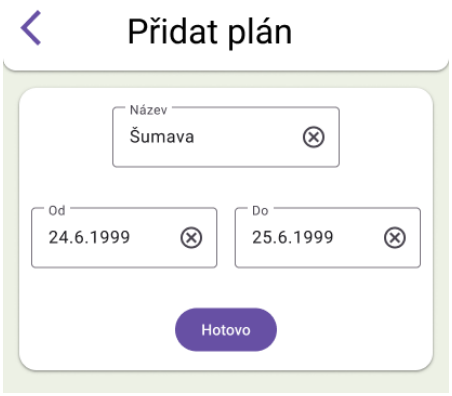
Zde se rozebírají destinace, které nepatří mezi hlavní a tedy na dolní části obrazovky se u nich nenachází navigace. Destinace jsou zařazené jako samostatné, protože je jejich priorita menší a není potřeba mít k nim okamžitý přístup z hlavních destinací.

3.2.1 Plánovač

Jedná se o skupinu destinací, které ve spolupráci tvoří systém na prozkoumávání a správu plánů. Kromě zmíněných destinací se ještě využívá zobrazení plánu na mapě. Nicméně se jedná pouze o znovu využití mapy s náhledem, jejíž funkce jsou rozepsány výše.

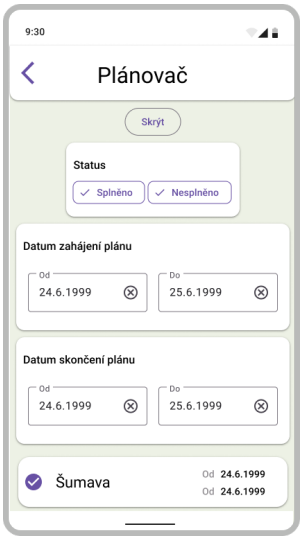
3.2.1.1 Vytvoření plánu

Pro vytvoření plánu se využívá jednoduchá obrazovka, na které se nachází pouze vstupní pole na text s názvem a druhé vstupní pole pro začátek a konec, které po kliknutí zobrazí kalendář a umožní vybrat datum. Poslední funkcí je samotné potvrzovací tlačítko. Na obrazovce jsou zachovány pouze 3 informace a z toho 2 dobrovolné, zbytek plánu se pro zlepšení UX zadává v detailu.



3.2.1.2 Vyhledávač

Aby byla umožněna rozumná práce s plány, samozřejmě je k dispozici filtrace. Na vrcholu můžeme kliknout na filtrovací komponentu, která se posléze rozbalí přes celou obrazovku. Ve filtrování najdeme 2 textové pole pro omezení datumu začátku a další 2 textové pole pro omezení datumu konce. Dále je možné hledat a filtrovat podle statusu, zda je plán dokončen nebo ne a nebo obojí. Plány jsou zobrazeny podle designového vzoru tabulka bez hlavičky1.1.3.2. Na každém řádku je plán obsahující název, status a pokud byly dodány, tak i datumy začátku a konce. Status je zobrazen jako ikonka prázdného kolečka nebo jako vyplněné kolečko s fajfkou.

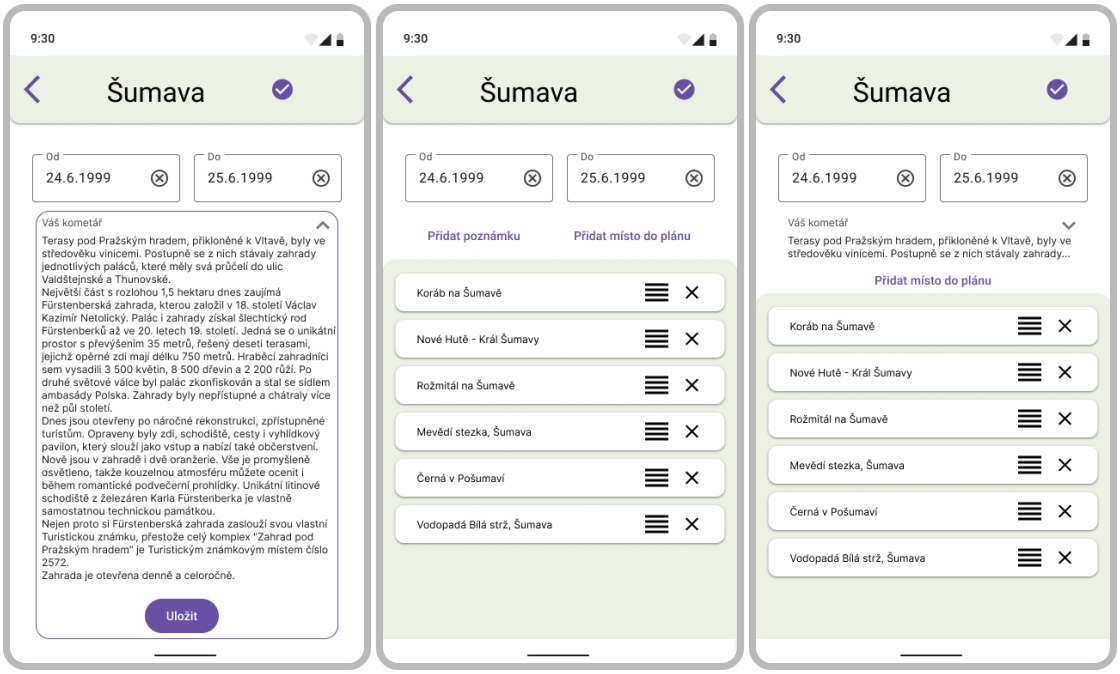


3.2.1.3 Detail plánu

Každý plán kromě zmíněného názvu a datumů dále obsahuje i seznam předmětů a možnost uživatelova komentáře. Název lze najít v nástrojovém pruhu mezi navigační ikonou zpět a ikonkou značící splněnost plánu. Pod nimi se upravují datumy nebo je k dispozici navigace na přidání místa do plánu.

Komentář nese 3 stavy. Jedná se o neexistující komentář, existující zkrácený a existující plně zobrazený. Pokud neexistuje je možné ho přidat pomocí tlačítka nacházejícího se vedle tlačítka pro přidání předmětu, jenž následně po přidání zmizí. Přidání komentáře způsobí vytvoření sekce pod datumy umožňující editaci komentáře, uložení a nebo popřípadě zrušení, v případě zrušení se zobrazí dialog vyžadující potvrzení. Sekce napravo zobrazuje ikonu pro rozbalení nebo pro skrytí umožňující přepínání komentáře mezi těmito stavy. Pokud je komentář zkrácený jsou k dispozici pouze 2 řádky, v případě kliknutí na změnu stavu dojde k rozšíření sekce ke konci obrazovky a komentář je kompletní a v případě nedostatku místa také rolovatelný.

Zobrazení plánovacích míst je realizováno podle designového vzoru tabulka a u každého předmětu lze vidět název předmětu s ikonami pro změnu pozice pro odstranění předmětu z plánu1.1.3.2. Pozice předmětů se může měnit pomocí aplikací gesta drag na předmět a posunutím předmětu na zvolené místo.



3.2.1.4 Dialog

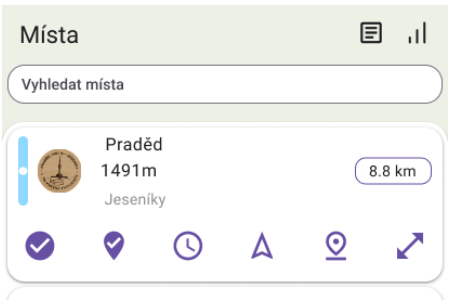
Pro funkce plánovače spojené s předmětem se využívá Dialog. Jedná se o komponentu překrývající část aktuální obrazovky při požadavku o zobrazení. Hlavním účelem je zobrazení rychlých akcí, bez nutnosti navigace. Je možné tedy říct, že se jedná o přechodnou vrstvu mezi předměty a plánovačem. Dialog umožňuje zobrazení navigace do plánovacího vyhledávače, náhled plánů s rychlým přidáním a možnost vytvořit plán.

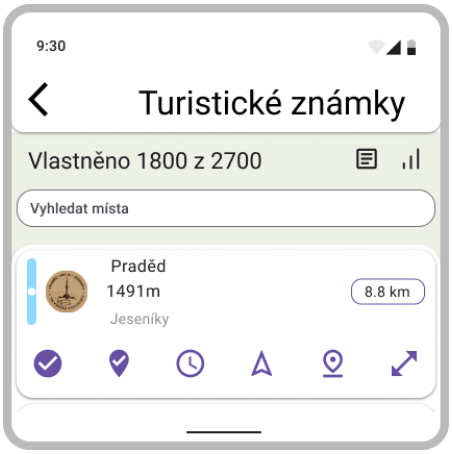
3.2.2 Přidání plánu k předmětu a přidání předmětu k plánu

Aktivity můžeme vykonávat pomocí existujících komponent. Přidání předmětu do plánu je realizovatelné pomocí vyhledávací vrstvy, kdy po kliknutí předmět nerozbalí nabídku s akcemi, ale přidá předmět do plánu. Přidání plánu k předmětu lze realizovat pomocí hlavní obrazovky plánovače, kde po kliknutí nedochází k přesunu do konkrétního plánu, ale k zařazení předmětu do plánu.

3.2.3 Konkrétní kolekce

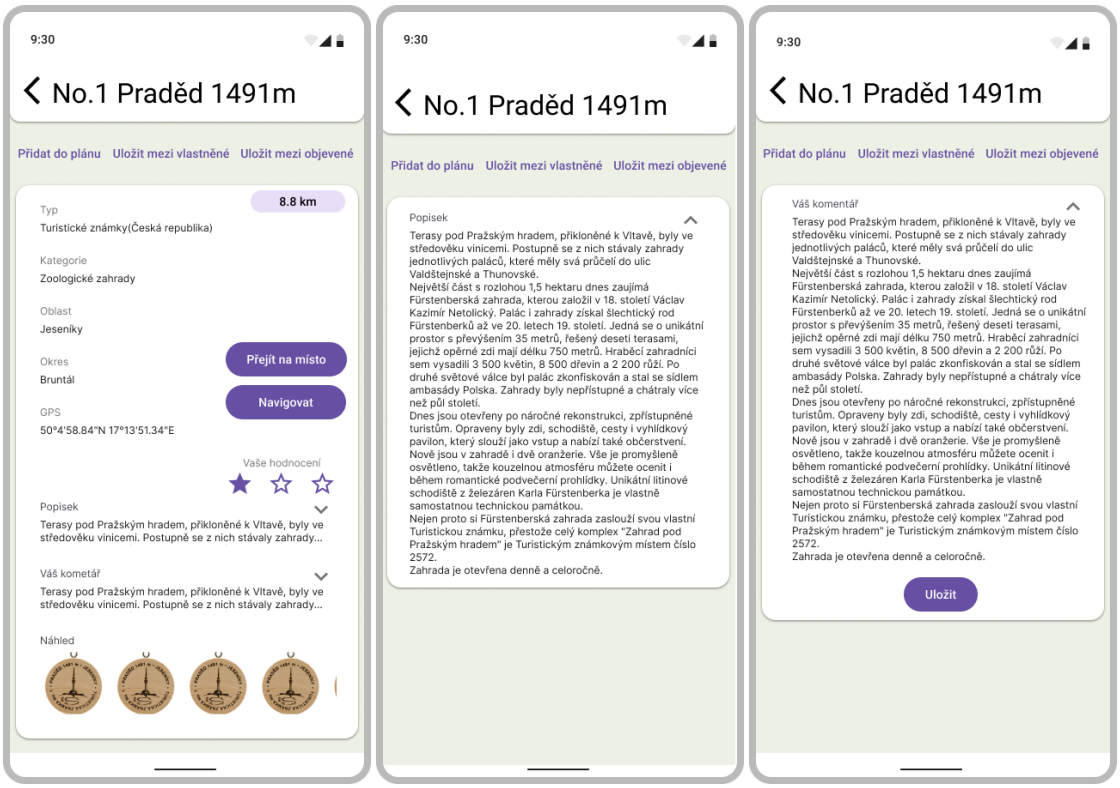
Hlavní vyhledávač má jednu vadu. Jedná se o skrytí vyhledávací překrývající vrsty při omylném táhnutí dolů buď za nástrojový pruh nebo na vrcholu seznamu. Pro umožnění vyhnutí se tomuto problému je možné místa hledat i na samostatné obrazovce. Jedná se o komponentu použitou v překrývající vrstě, ovšem rozšířenou o počet vlastněných předmětů a o nástrojový pruh, jenž obsahuje název kolekce, ve které se nacházíme.





3.2.4 Detail předmětu

V detailu jsou k dispozici všechny existující informace o předmětu. Oproti náhledu zde je navíc Kategorie předmětu, Kategorie místa, okres a seznam všech obrázků k předmětu. Základní funkce jsou k dispozici stejné jako v náhledu, samozřejmě bez tlačítka detailu. Disponuje ovšem i jinými funkcemi. První je spojena s polohou a pokud má předmět GPS lokaci tak pak uživateli, který dal povolení k využívání polohy zobrazí vzdálenost od místa. V případě, že zatím nedal povolení je zobrazeno tlačítko pro vyzvání k potvrzení povolení. Druhá je možnost hodnotit místa a zobrazit hvězdičky hodnocení. Přidávání hvězdiček je formou dialogu. Popisek lze zvěšit a zobrazit ho kompletní. U komentáře se dá otevřít editační mód, zvětšující komentář přes celou obrazovku.



3.2.5 Správa stažených kolekcí a Nastavení

Spravovat stažené kolekce lze pomocí tabulky, která na každém řádku zobrazí název kolekce a a zaškrtávací pole. V nástrojovém pruhu v horní části obrazovky je možné se pomocí navigační ikonky vrátit zpátky a nebo zaškrnout hlavní zaškrtávací pole a všechny kolekce zrušit nebo vybrat.

V destinaci nastavení lze pouze upravovat jazyk aplikace a spravovat nastavení opravávnění k poloze.

Kapitola 4
Implementace

4.1 UI vrstva

Mezi designem a implementací jsou menší změny, pro zlepšení UX designu a pro pokrytí více funkcí byl design upraven. Rozebereme zde jednotlivé komponenty a zjistíme, kde se využívají. Dále zde uvidíme vyzdvihnuté změny a jejich důležitost. Oproti designu zde existuje širokoúhlý mód. Doposud nebyl zohledněn, protože většinou se jedná pouze o změnu uspořádání obsahu.

4.1.1 Accompanist

Jedná se o skupinu knihoven rozšiřující základní funkce Jetpack Compose. Slouží spíše jako laboratorní prostředí pro funkce, jenž ještě nejsou plně připravevny na vydání ve verzi Compose. V aplikaci je využito několik knihoven.[61] Kromě knihoven zmíněných dále se ještě využívá Accompanist Permission, která slouží ke správě oprávnění udělených uživatelem.

4.1.2 Světlé a tmavé téma

Díky spolupráci Jetpacku Compose a Material Designu můžeme namapovat vybrané barvy na sadu barev, následně využitou v aplikaci. To umožňuje nejenom vyměnit barvy v celé aplikaci pomocí editace jednoho souboru, ale i snadné přepínání mezi světlým a tmavým režimem. Téma lze vybrat v nastavení a pokud není vybráno, tak je základním chováním pro Android menší než 10 nastavení tmavého režimu. Pro telefony s novějším Androidem je v základně využito systémové nastavení.[62] Pro úpravu barev v systémovém baru obsahující například čas, využivá se knihovny od Accompanist s názvem System UI Contoller.[63]

4.1.3 Znovupoužitelné komponenty

Základním konceptem Compose je znovupoužitelnost, jejíž cílem je ušetřit čas, zpřehlednit kód a ulehčit testování. Jelikož má většina obrazovek mód profilový i širokoúhlý a občas se jedná o stejný obsah, jenom například rozdělený, má poté každá destinace své znovupoužitelné komponenty. Zde můžeme najít komponenty, které se sdílí mezi destinacemi.

4.1.3.1 Dlouhý text

Umožňuje zobrazení několika řádkových textů. Jedná se o tónově vyvýšenou komponentu, pro kterou lze nastavit funkci, jenž se stane při kliknutí na ni. Obsah se určuje v samotných desti-

nacích.

4.1.3.2 Mapy

Mapy se v aplikaci nachází na celkem čtyřech destinacích. Navíc na dvou z nich je mapa využitá jako lineární prohledávač, jenž umožňuje se pomocí tlačítek navigovat v seznamu míst a na aktuálně vybrané místo v mapě přejít. V případě, že uživatel nedal oprávnění k využití polohy k zařízení, je dialog Povolení k využití zobrazen vždy po startu mapy.

4.1.3.2.1 Náhled V základu je také přidán náhled, jehož viditelnost a zobrazený předmět si komponenta ve svém ViewModelu spravuje sama. Oproti designu zde v náhledu byl přidán 1 stav a to konkrétně zkrácený náhled. Jedná se o zobrazení hlavních informací o předmětu nezabírající moc místa. Dále v rozšířeném náhledu, který svou podobu dostal již v designu, došlo k úpravě tlačítek, zobrazení popisku a rozložení informací. Na vrchol náhledu byly přidány tlačítka ovlivňující stav náhledu a to zmenšit/rozšířit a zavřít, tlačítka slouží k možnosti ovládat náhled bez gesta drag. Nadesignovaná tlačítka byla vyměněna za své ikony, díky kterým se nyní šetří místo a všechny akce jsou přesunuty na spodní čast obrazovky.Pro přidávání do navštívených jsou potřeba dialogy Zapnutí polohy, pro umožnění navštívenosti a Daleko od předmětu. Po stisknutí plánovací ikony je následně zobrazen dialog Plánovač. Ikony mají své barvy podle kategorie místa, které předmětu patří. Popisek může nabývat několika stovek znaků, je tedy k zobrazení v komponentě Dlouhý text a dá se zkrátit do takové míry, aby se předchozí informace zobrazily a nebo byly skryty a popisek zabíral většinu náhledu, v tomto případě je text rolovatelný. Poslední novinkou je ikona sloužící k zobrazení dialogu prodejních míst. Při rotaci displeje zůstává náhled stejný, jen je přesunutý na jiné místo a skrývání mapy funguje odlišným způsobem.

4.1.3.2.2 Zobrazování mapy s náhledem Podle Google pravidel 3.2.2 b) musí všechny značky, které přichází s produktem, být viditelné. [64] Z toho důvodu nejde pro náhled využít překrývající vrstvu. Dalším důvodem by bylo, že střed mapy by už nebyl ve středu zobrazovacího pole, ale pouze uprostřed obrazovky.

Pro profilový mód je možností mít tedy mapu nahoře, nastavit jí určitou výšku a nechat náhled zabrat zbytek. Tohle by fungovalo, ovšem náhled by nepřekrýval mapu. Pro překrytí musí být mapa na jiné vrstvé než náhled. Na vrstvě 0 je tedy mapa o velikosti X a na vrstvě 1 je průhledný box o výšce $BoxH = X - Y$, stím že Y je takové číslo, o které náhled překryje mapu, ale nesmí být tak velké aby překrylo logo Google Maps. Nakonec je náhled na vrstvě 1 pod průhledným boxem rozšířen na zbytek obrazovky.

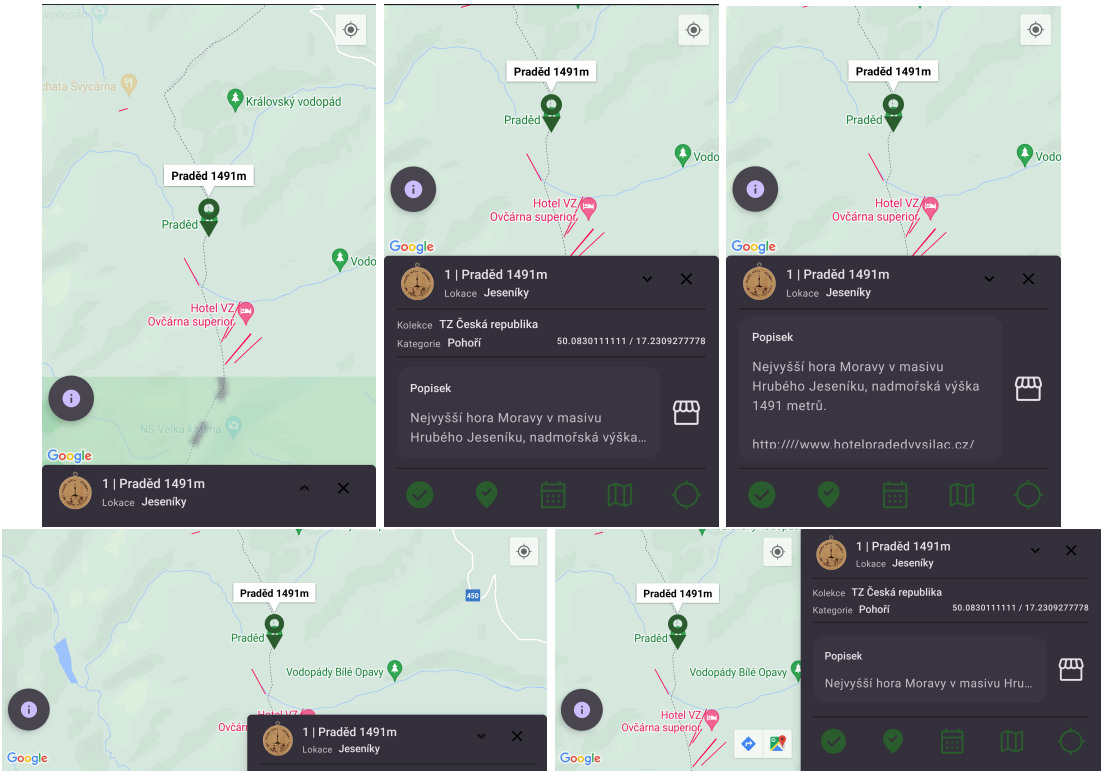
Širokoúhlý mód je také implementován na vrstvy. Na vrstvě 0 je mapa, jejíž šířka se pohybuje od A do šířky obrazovky. Na vrstvě jedna je průhledný box, který zabírá přesnou šířku A a vedle se nachází 1.vrstva z profilového módu. V připadě provedení drag gesta dolů/nahoru se mapa za náhledem buď zkracuje/roztahuje.

Způsob zmenšování a zvětšování náhledu je u obou módu podobný. Je zjištěna celková velikost obrazovky v Dp, což je jednotka míry umožňující podporu zařízení s rozdílnou pixelovou hustotou, s touto velikostí se následně pracuje. V případě profilového módu mapa zabírá při zobrazení od 48 do 100 procent. V případě zobrazení rozšířeného náhledu $X = 0.48 \cdot CelkovaVyska$. Při dragu můžeme zjistit o kolik jsme se posunuli. Při dragu dolů tedy stačí k X přičíst hodnotu, o kolik jsme se posunuli a dosáhneme požadovaného zmenšovacího efektu. Pokud má drag určitou nezanedbatelnou vzdálenost, změní se stav náhledu a buď se roztáhne nebo se z něj stane zmenšený náhled a X se potom nastaví hodnota podle aktuálního stavu. V případě dragu nahoru stačí posunutou vzdálenost odčítat. Drag nahoru je potřeba limitovat minimální velikostí pro mapu, jenž je 48% z celkové velikosti. Pro skrytý náhled poté platí, že $X = CelkovaVyska$ a pro zkrácený náhled se $X = CelkovaVyska - 75.dp$. Jak jsem výše zmínil pro širokoúhlý mód je způsob podobný, avšak s rozdílem procentuálního přepočítání šírky mapy. Například pokud posunu prstem při dragu o 50% dolů, aplikace se rozšíří na 75% za náhled. V případě zkráceného

náhledu se zase jedná o $X = CelkovaVyska - 75.dp$ a je zobrazena celá mapa pod náhledem, naopak u rozšířeného náhledu pak platí $X = CelkovaVyska$.

4.1.3.2.3 Způsob integrace V momentě začátku implementace už Google Maps existovaly jako View i jako Compose verze. Ovšem Compose verze neumožňovala získání map objektu, který je potřebný pro využití clusterování. Byla tedy ještě využita verze View. V aktuálních verzí Google Maps pro Compose už je ale problém vyřešen a přechod na Compose Google maps se nechází v úkolech, které slouží k možnému pokračování. Z důvodu, že mapa je ve View a nereaguje na Compose State změny, je potřeba v incializaci mapy začít asynchronně kolektovat Flow s údálostmi, které pak stav mapy mění. Mezi tyto události patří clusterování, přesunutí se na polohu v mapě, zobrazení tlačítka pro navigaci na uživatelovu polohu.

4.1.3.2.4 Vysvětlivky Umožňují získávání informací o značkách a barvách, neliší se oproti designovému návrhu. Jediná změna je v tom odkud se k nim můžeme dostat. Ikona byla přesunuta z pravého horního rohu, kde místo zabírá ikona pro nalezení uživatelovy polohy, která je od Google maps a nelze ji přesunout. Nově je ikona zobrazena v levém dolním rohu.



4.1.3.3 Tabule

Pro zobrazení rovnoměrně velkých kulatých elementů podle designového vzoru Springboard, bylo potřeba speciální komponenty. Základní komponenta od Compose jménem Grid umí zobrazit elementy v mřížce a zakulatit je, ovšem neumí aby každý element byl stejně velký, to je z důvodu, že jen jedna funkce umí zjistit velikost elementu než se zanese do obrazovky. Jedná se o Layout, jenž umožňuje změřit všechny elementy a jejich minimální a maximální velikost před tím než se elementy vloží na obrazovku, ovšem pokládání na obrazovku je potřeba dělat manuálně a pro každý element je potřeba vypočítat pozici. Podle počtu zadaných sloupců se vypočítá velikost elemetů, tak aby se vyplnilo místo na displeji. Pro umožnění znovupoužitelnosti tabule jenom

poskládá elementy do mřížky a zaručí stejnou čtvercovou velikost, co tabule bude obsahovat a jak se bude chovat už závisí na konkrétním využití. Poslední zárukou je rolovatelnost.

4.1.3.4 Vyhledávání, filtrování a řazení

Komponentu lze využít ze 3 míst a jsou 2 módy, ve kterých ji lze používat. První slouží pro vyhledávání napříč všemi kolekcemi a druhý k prohledávání jedné konkrétní kolekce. Jedná se o spojení 4 destinací pod jednou. Jak seznam míst, tak kolekce, filtry a řazení jsou samostatné destinace, nad kterými je speciální nástrojový pruh. Úvodní destinací je seznam předmětů. K animování přechodu mezi destinacemi je zde využita knihovna od Accompanist Navigation-Animation.[65]

4.1.3.4.1 Nástrojový pruh Některé funkce se mění v závislosti módu a také na aktivní destinaci. Při vyhledávání přes všechny kolekce lze na prvním řádku pruhu vidět pouze navigační ikonu zpět, název aktivní destinaci a pak ikony pro přepnutí aktivní destinace. Přepínací ikony jsou 3, ale zobrazené jsou vždy 2. Levá pracuje s filtry a kolekcemi, pravá pak s řazením. Pokud destinace není aktivní pak je tam její ikona a pokud je aktivní tak je zobrazena ikona kříže, který způsobí návrat na úvodní destinaci. Na první je zobrazen kříž pokud je aktivní destinace kolekce nebo filtry. Pokud je aktivní destinace seznam je pak podle designové návrhu zobrazeno i vstupní pole pro filtrování dat podle názvu, oproti návrhu je pole ovšem v pruhu a né mimo něj. V případě filtrování se využívají filtrovací čipy, jenž jsou vyplněné pokud jsou aktivní, pro úpravu vyhledávaných statusů a s jejich pomocí se vybírá zda předměty mají být navštívené, vlastněné nebo v plánu. Pod statusy se pak ovládá filtrování podle vzdálenosti, pokud uživatel udal povolení, v případě že ne je zobrazen dialog Povolení k využití polohy. Řazení nemá v pruhu žádné speciální funkce.

Pokud se jedná o mód jedné kolekce, je název kolekce napsaný na vrcholu. Vedle něj se v případě, že se u předmětů v kolekci nachází GPS, zobrazí ikona, která má sloužit pro zobrazení kolekce v mapě, tyto 2 funkce se zde pak nachází při jakékoliv aktivní destinaci. Dále pokud se jedná o kolekci bez GPS pak nelze filtrovat podle vzdálenosti.

4.1.3.4.2 Místa Vzhled jednoho předmětu je poměrně razantně změněn. Kvůli výkonostním důvodům je miniatura známky nahradila číslem předmětu v kruhu s barvou podle Kategorie místa. Barva kategorie se také projevuje u ikon a indikátoru stasu k předmětu. Vektorový návrh indikátoru je nahrazen čistě elementy z Compose, otočen o 90 stupňů a roztažen přes celé místo, které předmět zabírá, to by mělo umožnit lepší čitelnost při rychlém rolování. Dále je mezi název a oblast přidána kategorie předmětu. Poslední změnou je v případě řazení podle nějakého datumu zobrazení onoho data u předmětu pod vzdáleností, v případě že není k dispozici datum ani vzdálenost, je zanecháno malé kolečko symbolizující, že na daném místě se může nacházet element. Širokoúhlý mód je pak rozšířen o funkcí náhledu. Při vybrání konkrétního místa se pruh s místy zmenší na polovinu šířky a vedle je zobrazen náhled místa. Náhled obsahuje stejné informace co náhled na mapě plus navíc okres. K tomu je v náhledu k dispozici i funkce pro vyhledání pozice předmětu v seznamu. Kvůli tlačítku pro navštívenost jsou zde potřeba 2 dialogy a to Zapnutí polohy pro umožnění navštívenosti a Daleko od předmětu. Po stisknutí plánovací ikony je pak zobrazen dialog Plánovač.

4.1.3.4.3 Filtrování Velkým rozdílem oproti designu je přidání jedné rozšiřující destinace a jedné samostatné destinace. Po kliknutí na filtry ještě není možné okamžité zobrazení seznamu všech filtrů, ovšem jenom stažených kolekcí. Každá kolekce má totiž jiné filtrovací možnosti, to je z důvodu nekonzistence od Turistických známek. Při jednoduchém kliknutí na kolekci se mohou stát 2 akce. Buď se kolekce pouze přidá mezi prohledávané kolekce a zvýrazní se, to by znamenalo, že kolekce nemá žádné další filtry mimo ty, které jsou v nástrojovém pruhu. Druhá možnost je, že se kolekce přidá mezi prohledávané a následně dojde k navigaci na seznam možných filtrů. Ve

výběru filtrů se kolekce může zadávat nebo odstraňovat z prohledávaných. Dále po klinutí na filtr dochází k přesunu na samostanou destinaci pro vybírání konkrétních předmětů do filtru. Kolekce lze přidávat mezi prohledávané i pomocí hromadné akce nazvané dlouhé kliknutí. Při dlouhém podržení některé kolekce dojde k přepnutí do režimu vybírání a teď lze na kteroukoliv kolekci kliknout, s tím že nedojde k žádné navigaci, pouze se přidá mezi prohledávané nebo popřípadě odebere, k potvrzení výběru dojde až při ukočení režimu pomocí kliku mimo kolekce. Hromadné akce splňují princip efektivity číslo 7 od Nielsena.

Širokoúhlý mód umožňuje zobrazit nástrojový pruh přes levou část displeje a na pravo vložit komponentu k vybírání kolekcí a filtrů.

Mód jedné kolekce umožňuje při filtrování přeskočit výběr kolekce a je buď navigováno rovnou k filtrům a nebo v případě absence filtrů se jenom přepne nástrojový pruh na filtrování a v obsahu zůstane seznam míst.

4.1.3.4.4 Řazení Díky přidání řadícího kriteria Datum komentáře už není možné využít designový vzor na obrazovce a je potřeba použít pro řazení samostatnou destinaci. Při vybrání se kriterium a směr řazení zvýrazní. Pro profilový mód jsou elementy zobrazeny pod sebou a v případě širokoúhlého naopak v mřížce. V případě, že uživatel nemá povolenou polohu není řazení podle vzdálenosti dostupné, v případě kliknutí je zobrazen dialog Povolení k využití polohy.

4.1.3.5 Komentář

Oproti designu je v aplikaci možné editovat komentář pouze na samostatné obrazovce. Kvůli 2 typům komentářům existuje tato komponenta, díky které lze pro každý typ komentáře dodat jinou logiku ale UI zachovat stejné.

4.1.3.6 Dialogy

Celkem lze v aplikaci najít 23 dialogů, sloužících k informovování popřípadě k rozšíření funkcionality. Dialogy mohou fungovat jako povrzování nebo k zobrazení dodatečných funkcí. Následující dialogy mohou obsahovat potvrzovací akce, která se vkládají jako parametr, umožňující znovupoužitelnost.

4.1.3.6.1 Prodejní místa - jednoduchý dialog umožňující zobrazení adres. Očekávaná potvrzovací akce, jenž se má spustí při kliknutí na adresu, je navigace do extérních map, protože využití hledání míst přímo v aplikaci už by zdarma.

4.1.3.6.2 Povolení k využití polohy - Slouží k informování, o tom že aby uživatelé mohli využívat vzdálenost od míst a sledovat svoji polohu na mapě, tak musí aplikaci udělit povolení k využívání polohy. Očekávaná povrzovací akce je informování Androidu o spuštění požadavku na získání oprávnění, o následnou správu oprávnění se již stará samotný Android.

4.1.3.6.3 Plánovač Oproti designovému návrhu je dialog podstatně jednodušší. Hlavní změnou je odstranění seznamu plánu a vyhledávacího pole z plánovače. Pro přidání plánu k předmětu existuje samostatná destinace. Další novinkou je zobrazení tlačítka sloužícího jako vstupní bod pro navigaci na obrazovku se seznamem plánů k danému předmětu.

4.1.3.6.4 Zapnutí polohy pro umožnění navštívenosti - V případě, že uživatel neudělil povolení k využívání polohy a žádá o přidání předmětu mezi navštívené následuje zobrazení dialogu informující o tom, že předměty lze přidat pouze s uděleným oprávněním. Potvrzující akcí by opět mělo být infomování Androidu o spuštění požadavku na získání oprávnění.

4.1.3.6.5 Daleko od předmětu - Pokud uživatel není v dostatečné blízkosti k předmětu, dochází k omezení akce pro přidání a zobrazení informací o nutné minimální vzdálenosti k předmětu.

4.1.4 Destinace

Pro navigaci mezi jednotlivými obrazovkami se využívá Compose Navigation.[66] Navigační komponenta umožnuje držet informace o aktuální destinaci i přes konfigurační změny. V případě, že se nenachází destinace, kterou aplikace obsahuje, pak v ní oproti návrhu nebyly žádné změny.

4.1.4.1 Hlavní

Objevovač je spojením komponent Mapy a Vyhledávání, řazení a filtrování. V případě, že se do mapy vloží funkce pro začátek vyhledávání, pak se vedle ikony pro zobrazení nápovědy zobrazuje i ikona pro zapnutí vyhledávací vrsty s počtem míst, které jsou v seznamu a mapě. Překrývající vrstva vyhledávání je realizována pomocí Accompanist Navigation-Material. Při požadavku zobrazit předmět na mapě z vyhledávací vrsty dojde ke skrytí vyhledávácí vrsty a k zaslání události do mapy o zobrazení značky.[67]

U Informací šlo od návrhu změnit pouze tab s nejnovějšími známkami, protože ostatní 2 taby jsou pouze zobrazené webové stránky, tyto 2 jednoduché taby jsou implementovány pomocí Accompanist Webvíew.[68] U nejnovějších známek je každý předmět tónově vyvýšen pro lepší přehlednost a také má svůj dialog pro plánování.

Na destinaci Profil došlo k odstranění karty, jenž odsazovala celý obsah, to umožňuje využití více místa. V případě, že uživatel není přihlášeny, se na profilu zobrazuje možnost navigovat na přihlašovací stránku Turistických známek.

4.1.5 Přihlášení

První obrazovkou, kterou uživatel po nainstalování uvidí je výběr přihlášení. Zde si může zvolit mezi anonymním přihlášením a přihlášením přes web Turistických známek, na stejné webové adrese je dostupná i registrace. V případě, že uživatel je libovolně přihlášen přeskakuje se na další destinaci. Následně dochází k vybrání kolekcí ke stáhnutí. Po potvrzení kolekcí se přeskakuje už na hlavní destinace.

Pro přihlášení přes webové stránky nejde použít Accompanist komponenta pro WebView, protože je potřeba parsovat odpověď s kódem od serveru. Je teda potřeba využít přímo WebView, stejným způsobem jako u Google maps a zajistit kompatibilitu pomocí AndroidView.

4.1.6 List s předměty pod konkrétním filtrem

Oproti designu, filtrovací kriteria obsahující N kategorii se už při filtrování nevybírají přímo na obrazovce se filtry, ale po kliknutí na filtr dochází k přechodu na tuto destinaci. Zde se nachází možnost vyhledávat kategorie pomocí textu a pomocí zaškrátacího pole si jednotlivé kategorie vybírat. Destinace také obsahuje hlavní zaškrtávací pole, které manipuluje se všemi kategoriemi. Jednolivé kategorie se zobrazují pole vzoru tabulka se seskupenými řádky1.1.3.2, kde v hlavičce je první písmeno kategorii.

4.1.7 Detail předmětu

Společnou úpravou, kterou mají profilový i širokoúhlý mód je využití komponenty Dlouhý text pro zobrazení popisku a komentáře. Pro editaci kometáře se využívá samostatné destinace.

4.1.7.1 Profilový mód

Důležitou změnou je seskupení všech tlačítek z obrazovky do jedné komponenty na spodu obrazovky. To umožňuje lepší přehlednost destinace a navíc je komponenta zkracovatelná to znamená, že při navigaci na destinaci se zobrazuje pouze první řádek tlačítek, k objevení osatních funkcí je potřeba kliknout na rozšíření. Využívá se spodní strana obrazovky, protože zde je lepší přístup pomocí palce. Dále je mezi tlačítka přidané zobrazení dialogu Prodejních míst a dialogu webových adres k předmětu. Vlatněně, Navštivené a Plánované tlačítko je realizováno pomocí více stavových tlačítek, jenž buď zobrazují pouze text symbolizujicí nesplnění kriteria a nebo se jedná o tónově vyvýšené tlačítko symbolizující splění. V případě navštíveného tlačítka je při splění k dispozici i datum navštívení.

4.1.7.2 Širokoúhlý mód

Jedná se pouze o jiné rozložení komponent. Obrázky nejsou na řádku, ale ve sloupci úplně nalevo. Informace bez komentáře a popisku se nacházejí uprostřed vlevo. Popisek s komentářem uprostřed vpravo a tlačítka už nejsou skrývatelná a nachází se ve sloupci úplně vpravo.

4.1.7.3 Obrázky

Informace o předmětu a tlačítka se nachází na spodní vrstvě společně s minatury obrázků. V případě kliknutí na obrázek dochází k zobrazení překrývající vrsty s prohledávačem obrázků. Při prohledávání lze využít buď tlačítek na vrcholu obrazovky nebo gesta drag. Využit je designový vzor Tabs, kde každý obrázek má svůj Tab. Realizace gesta drag, speciální animace při přechodu a zobrazení indikátoru o aktivním obrázku na spodu obrazovky je realizováno pomocí rozšiřující knihovny od Accompanist Pager.[69] Pokud se obrázky nenačtou z internetu je možné požádat o obnovení.

4.1.8 Přidání plánu

Oproti designu je možné narazit na omezení. Prvním je, že každý plán musí mít název, bez toho nejde plán vytvořit a v případě takového pokusu se pouze vybarví ohraničení textového pole pro název červeně. Dalším omezením je, že začátek plánu nemůže být po konci, v takovém případě se pak zase vybarví ohraničení obou polí pro datumy červeně. V plánovacím dialogu existuje funkce vytvořit plán, ta má skoro stejný efekt, ale díky tomu, že se plánovací dialog volá na konkrétním předmětu, je, po vytvoření plánu, do něj automaticky přidán.

4.1.9 Vyhledávač plánů

Vyhledávač zůstal z části nezměněn, ovšem jsou přidány možnosti filtrovat podle názvu a filtrovat podle datumu vytvoření. Dalším nedostatkem designu byla absence řazení. Nyní lze řadit podle názvu a všech datumů. Filtry a řazení mají svůj širokoúhlý režim, při kterém se jejich kriteria zobrazují v mřížce o dvou sloupcích. Poslední změnou je přesunutí datumu u jednotlivých plánů pod název.

4.1.10 Detail plánu

Pro komentář se využívá komponenta pro Dlouhý text a editace komentáře je na samostatné obrazovce, ale důležitější změnou je výměna pořádí u plánových zastávek. V návrhu bylo využito gesto drag. Ovšem kvůli problémové implementaci v Compose verzi 1.3 je změna tvořena pomocí tlačítek nahorů a dolů. Ovšem na list je aplikována animace pozic a tedy v případě, že se u předmětů změní pozice, samy se pak naanimují na své místo. Další změnou u předmětů je možnost

rozkliknutí a zobrazit si u nich rychlé akce. Jedná se o přidání do vlastněných, navštívených, zobrazení předmětu na mapě, extérní mapě, navigaci do detailu a odstranění předmětu, které je nyní zobrazeno až po rozkliknutí. Akce související s polohou jako uložení navštívení a přesun do map je možné využít pouze pokud má předmět polohu. Poslední akcí, která není pro předmět ale pro plán, je znázornit plán na mapě. V případě, že se v plánu nachází předměty bez pozice, je při požadavku o znázornění zobrazen dialog s informacemi o tom, že předměty bez pozice se v mapě nezobrazí. Destinace také obsahuje sekci pro upozornění na vrcholu obrazovky. V případě, že uživatel klikne na přidání předmětu do plánu dojde k navigaci na samostatnou destinaci pro vybírání předmětu. Po přidání dojde k navigaci zpět na detail plánu a zobrazí se upozornění o přidání předmětu do plánu. V případě že uživatel vymaže některý předmět z plánu dojde k zobrazení upozornění o smazání. Novinkou je i možnost smazat plán a zobrazení dialogu při nastavení datum začátků po datumu konce.

Širokoúhlý režim zobrazuje informace na levé straně obrazovky a na pravé pak seznam plánovaných předmětů.

4.1.11 Konkrétní kolekce

Oproti vyhledávání ve všech kolekcích, zde při požadavku zobrazit místo v mapě dojde k navigaci na novou destinaci. V nástrojovém pruhu je jedna změna, která závisi na uživatelově nastavení a jde o zobrazení počtu vlastněných nebo počtu navštívených předmětů v kolekci. V případě, že kolekce nemá předměty s GPS, pak i přes nastavení zahrnující naštívené předměty se zobrazí počet vlastněných.

4.1.12 Kolekce nebo plán na mapě

Pro prohledávání seřazených míst v mapě se využívá lineární prohledávač. Jedná se o rozšířenou komponentu map disponující šipkami pro přešun mezi místy.

4.1.13 Přidávání plánu k předmětu nebo předmětu k plánu

Jedná se o znovuvyužití destinací Vyhledávací vrstva a plánovací Vyhledávač. Dochází pouze k dodání nových funkcí, které se mají stát po kliknutí buď na předmět nebo na plán. Po vybrání předmětu nebo plánu dochází k navigaci zpět a k zobrazení upozornění o přidání na vrcholu obrazovky. Pokud už je předmět s plánem spojen není umožněno znovupřidání a je zobrazen dialog s upozorněním. V prvním fázi implementace této destinace bylo umožněno znovu přidání, následný návrat a informace o existujícím předmětu byly až pak zobrazeny na destinaci, ze které bylo o přidání požádáno, to ale nesplňuje princip číslo 5 od Nielsena.

4.1.14 Stažené kolekce a Nastavení

Stažené kolekce se vyskytují na 2 místech a to buď následně po přihlášení kdy je potřeba zadat kolekce, se kterými se bude pracovat nebo pak se do nich dá navigovvat z hlavní obrazovky Profil, kde je textové tlačítko Upravit. Stažené kolekce jsou dále rozšířeny o povrzovací dialog zobrazující se při požadavku o navigaci zpět. Účelem je vyzobrazení kolekcí ke stáhnutí, smázání a vyžádání potvrzení.

Do Nastavení se naviguje pouze z hlavní destinace Profil. Oproti návrhu zde je přidáno několik možností. Na vrcholu se nachází přepínač mezi světlým a tmavým módem. Dále se nachází možnosti importu a exportu, následované přepínačem pro zobrazení počtu vlastněných nebo počtu navštívených předmětů u konkrétních kolekcí. Na úplném spodu se nachází tlačítko odhlášení. A uprostřed tlačítko pro synchronizaci a tlačítko pro zobrazení dialogu Povolení k využití polohy. V případě, že uživatel neudělil povolení k využití uložiště a chce provádět export

nebo import, je třeba si ho vyžádat. Proces zase začíná dialogem buď pro zápis nebo čtení, který při povrzení pokračuje požadavkem na samotný systém Android. Pokud jsou povolení udělena pak pro anonymního uživatel lze dělat kompletní export a kompletní import, ovšem v případě přihlášeného uživatele zde existuje omezení na využití kompletního exportu a pouze importu plánů, kvůli synchronizaci uživatelových kolekcí se serverem.

4.2 Datová vrstva

Datová vrstvá slouží k odstranění závislostí na tom jak jsou data získávány. V případě potřeby lze zachovat rozhraní datové vrstvy a kompletně předělat například databázový systém.

4.2.1 Offline data

Hlavní myšlenkou je Single source of truth, což zkráceně znamená spojení dat z více zdrojů do jedné lokace.[70] Kvůli možnosti reaktivního přístup je pro uložení data využita Jetpack Room. Většina dat zobrazovaných v aplikaci je získávána reaktivně, tedy data jsou zabalena ve flow, která se pak kolektuje. Výjimkou jsou komentáře při editaci, které stačí načíst pouze při začátku.

Pro ukládání neupravených obecných kolekcí se využívá DatabaseSaver, jedná se tedy o přechod mezi daty obecných předmětů získaných pomoci vzdáleného přístupu a offline databází. Hlavním účelem je správa identifikátorů a rozdělení předmětů do entit spojených relacemi. Ptáte se co znamená správa identifáktorů? Předměty v kolekci mohou mít stejné číslo předmětu, tedy jako identifikátor ho nelze použít. V případě existence Id se k němu pouze přidá písmeno z abecedy a pokud i to je obsazené zvolí se písmeno následující.

Kvůli zlepšení soudržnosti je databáze rozdělana na 4 DAO. Jedná se o 4 zaměření první jsou předměty, druhé plány, třetí import a export a čtvrté je pro kolekce jako celek, sloužící pro úpravu stažených kolekcí.

Pro ukládání informací o uživateli není využita Room, ale je využit DataStore. Podle Google je DataStore ideální pro malé jednoduché data sety.[71] Pomocí něj se ukládají informace o uživateli, konkrétně se jedná samotného uživatele a nastavení specifické pro uživatele, zahrnující například oblíbené kolekce nebo UI téma.

4.2.2 Vzdálený přístup

Při zavolání požadavku na URL, je v případě úspěchu vrácen objekt s HTPP odpovědí. Díky využití způsobů pro automatickou konverzi je získáván rovnou objekt s informacemi o předmětech, v případě neúspěchu při konverzi je pak vrácena výjimka a to jak v případě Ktoru, tak v případě Retrofitu. Dále mají Ktor a Retrofit stejný způsob inicializace a to pomocí vzoru Builder1.2.1.1.2. Pro bezpěčnější získávání dat ze vzdálených zdrojů se využívá komponenty, jenž zabalí výsledek do třídy Resource, v případě úspěchu se pak jedná o Resource.Success obsahující data nebo o Resource.Loading a nebo o Resourece.Failure. Tato komponenta zabraňuje pádu aplikace při vyhozené výjimce.

4.2.2.1 Přihlášení

Pro přihlášení je vytvořena komponenta obsahující pouze jednu metodu, která vytvářít http požadavek pomocí Ktor klienta. Do metody je potřeba zadat secretId, clientId a code. SecretId a ClientId byly vygenerovány speciálně pro tuhle aplikaci a jsou uloženy v kódu. Ovšem ukládat klíče takhle v kódu by byl bezpečnostní prohřešek umožňující provést jednoduché reverzní inženýrství. Dochází tedy k využití SecretPluginu, který klíče zašifruje a vytvoří cestu k získání odšifrované verze, dále se snaží zašifrované verze co nejhlouběji schovat.[72] Poslední potřebnou položkou je code, ten je zparzován při úspěšném přihlášení. Při úspěšném požadavku jsou pak vrácené informace o uživateli společně s tokeny uloženy do DataStoru.

4.2.2.2 Obecné známky

Každá kolekce má svoje URL, to po zavolání požadavku vrátí kolekci ve formátu XML. Původní záměr bylo využít Ktor. Ovšem kolekce obsahují i listy, kterým Ktor nerozumí. Po snažení bylo od spolupráce Ktor a XML upuštěno a vloženo mezi budoucí možné úpravy. Místo toho aplikace

nyní využívá Retrofit se speciálním XML konvertorem. Retrofit klient dodává implementaci k DAO, které disponuje rozhraním pro získání všech kolekcí.

4.2.2.3 Uživatelovy známky

Pro práci s uživatelovými kolekcemi je zapotřebí 2 požadavků. Mít aktuální přístupový token a u každého požadavku vypočítat action hash. V současné době jsou povoleny 4 typy operací a to přidat předmět jako navštívený, přidat předmět jako vlastněný, získat navštívená místa a získat vlastněné předměty. Omezení mazání je z důvodu kyber útoků, které byly provedeny na turistické známky. Útočníci mazali uživatelům známky a tak došlo k vypnutí funkčnosti pro URL určených k mazání.

4.2.2.3.1 Funkce Je možné zažádat o informace z více kolekcí pomocí jednoho požadavku, tedy získat vlastněné nebo navštívené předměty z různých kolekcí. Do požadavku je tedy potřeba zadat seznam kolekcí se svými Action Hashi. Pro přidání míst do uložených je potřeba poslat i aktuální GPS zařízení. Přidání komentáře je možné pokud je předmět uložený jako vlastněný a lze to jak pomocí funkce na přidání místa tak pomocí funkce na přidání předmětu. Pro přidání je potřeba k typu kolekce, státu a číslu předmětu přidat i Action Hash předmětu.

4.2.2.3.2 Obnovovací token V případě, že přístupový token není aktuální, je vrácen kód 401. Pro automatické vyřešení takové situace je vytvořen speciální plugin a speciální třída obsahující funkci na zavolání URL pro obnovu přístupového tokenu. V případě, že dojde k vrácení 401 pak plugin automaticky zavolá funkci pro obnovení přístupového tokenu s posledním obnovovacím tokenem, který byl vrácen. Při úspěšném obnovení tokenu dochází jak k vrácení přístupového tak i obnovovacího tokenu. Tokeny se následně uloží do DataStoru a pak plugin automaticky zavolá požadavek s již platným přístupovým tokenem.

4.2.2.3.3 Action Hash Pokud jde o získávání míst nebo předmětů z konkrétních kolekcí musí být ke každé kolekci vytvořen action hash. Pro ukládání předmětů a míst do uložených je také pro každý předmět potřeba vytvořit action hash. Vytvoří se jako md5 hash md5 hashe, kde je složený řetězec s číslem, státem, typem, požadavkem(jedním ze 4 možných), souřadnicí lat, souřadnicí lng, pevným řetězcem „ZMobilniAplikace“ a aktuálním UTC datem ve formátu dmY. V momentě kdy není nastavena souřadnice lat nebo lng se použije 0, namísto prázdného řetězce.

4.3 Repozitáře

Jedná se o rozšíření MVVM, kdy se mezi Model a ViewModel vloží repozitář. Slouží pak jako přechod mezi Logickou a Datovou vrstvou. Data zde mohou být transformována pro konkrétnější použití. Díky rozhraní repozitářů je pak snadné nahradit datovou vrstvu. Kromě oddělení závislosti dochází také k funkcionálně důležitým akcím. Repozitář je použítý u každého přístupu k datům. Může se jednat o přístup k uživatelových sbírkám, obecným sbírkám, obnovení tokenu nebo pak k offline datům ať se jedná o Room nebo DataStore. Dále pomocí návrhové vzoru Template 1.2.1.3.4 existuje rozšiřující rozhraní zajišťující práci s komentáři, určeno pro repozitář předmětů a pro repozitář plánů. Template vzor je využit kvůli rozdílnému typu identifikátoru, předměty mají text a plány číslo. Repozitáře spojují občas i několik tříd z datové vrstvy, využívájí u toho vzor Facade1.2.1.2.1.

4.3.1 Zobrazení na mapě

V Google Maps nelze na jednom místě zobrazit více značek, navzájem se pak překrývají a viditelná je pouze jedna. Pro eliminaci problému dochází k seskupení značek podle polohy a

následnému rozmístění do kruhu okolo jejich pozice.

4.3.2 Kombinace funkcí

Jednotlivé Flow se dají kombinovat, tedy pod jedním požadavkem na repozitář se může skrývat několik reaktivních požadavků na databázi, které pak projdou transformací a vrací zkombinovaná data.

4.4 Logická vrstva

Při využití MVVM vzoru je logika přesunuta z UI do ViewModelů, ovšem sdílení modelů není nejlepší řešení, protože jednotlivé obrazovky se od sebe můžou lišit. Tedy další způsob je vytvořit na konkrétní účely speciální třídy, které se pak mohou sdílet do ViewModelů, tím si každá obrazovka zachová své rozhraní, ale je umožněno znovu využití komponent. Komponentu lze sdílet více způsoby. Pokud je komponenta nezávislá na ViewModelu, lze komponentu incializovat v samostatném modulu a pomocí DI ji vložit do ViewModelu, ovšem pokud je inicializace závislá pak musí přijít z VM. Pokud je nezávislá na ViewModelu, lze pak využít vzor delegace, kdy se rozhraní ViewModelu rozšíří o rozhraní komponenty a dodaný objekt se pak nastaví jako realizátor daného rozhraní.

4.4.1 Plánování

Téměř všechny destinace spojené s plánováním se někde opakují, kromě detailu a plánu v mapě. Tedy jdou vytvořeny tyto komponenty.

4.4.1.1 Přidání

Přidávat komentáře je možné buď z plánovače nebo z konkrétního místa. Každé přidání má svou destinaci. Logika pro přidání je tedy zabalena v AddPlanHandler a dodávána do VM. Díky nezávilosti na VM, lze incializovat přímo v modulu, využít DI a následně rozhraní ViewModelů určených pro přidávání plánů realizovat pomocí delegace na dodaný objekt. Handler, ale obsahuje data jako například plány a datumy, tedy při využití vzoru Singleton by při každém pokusu o přidání bylo potřeba provést reset data, z toho důvodu zde není využit. Aby bylo možné rozlišit komu plán patří, je potřeba možnost zjištění o jakého uživatele se jedná a tedy zavést propojení na repozitář pracující s uživatelem.

4.4.1.2 Hlavní plánovač

Zobrazit seznamu plánů, lze na 3 destinacích. První je základní prohledávač plánů, druhou je přidání plánu k předmětu a třetí je zobrazení plánů, ke konkrétnímu předmětu. Na všech třech destinacích se plány můžou filtrovat a řadit. Existuje tedy komponenta PlannerHandler, jenž se skládá z PlanFilter a PlanSorter a PlannerSqlGenerator. Princip je jednoduchý filter a sorter drží informace o tom jaká kriteria jsou vybrána a tyto kriteria se pak předají do Sql generátoru, který z nich vygeneruje SQL dotaz, jenž se následně posílá jako argument do PlanRepository. Dao následně vrací plány podle požadavku. Handler navíc disponuje i logikou plánovače a řeší i tedy, která sekce plánovače bude zobrazena, například seznam nebo filtrování. PlanRepository a PlannerSQLGenerator nedrží žádná dodatečná data, na kterých by závisely požadavky, tedy obojí je implementováno jako Singleton a vloženo pomocí DI. Filter, Sorter a SQLGenerator a jsou pak do Handleru zakomponovány pomocí delegace, kdy rozhraní Handleru se skládá z rozhraní těchto 3 komponent a inciliace Filtru a Sorteru dochází přímo při delegaci pomocí

defaultního prázdného konstruktoru. SQL generator ještě disponuje možností získat informace o uživateli.

4.4.1.3 Status

Poslední komponentou pro Plánování je PlanStatusHandler, jenž slouží pro zobrazení výsledků operací týkajících se plánů. Konkrétně jde o přidání předmětu do plánu, odstanění předmětu z plánu, pokus o přidání předmětu do plánu, ve kterém už se předmět nachází a neplatné nastavení datumů pro plán. Tyto akce podle Nielsena vyvolávají reakci. Komponenta pak obsahuje logiku pro zobrazení těchto statusů. Jedná se hlavně o způsob zobrazení a dobu. Zobrazení může být formou pruhu na vrcholu obrazovky nebo pomocí dialogu. Například zobrazit status pro přidání předmětu do plánu by mělo být dostupné, ze všech obrazovek kde se nachází plánovací dialog, protože z něj se jde dostat na destinaci pro přidávání předmětu a po přidání by mělo dojít k návratu zpět a k zobrazení statusu. Handler obsahuje informaci o typu statusu, tedy není incializován jako Singleton.

4.4.2 Filtrování a řazení

Oproti plánovači je zde jedna změna. V případě, že se změní seznam předmětů je potřeba informovat volajícího, tento požadavek vznikl kvůlí mapám aby šlo propagovat změny, dále se pak využívá k rolovaní v seznamu míst na vrchol při změně vyhledávacích požadavků. Vyřešeno je to formou přidání funkce do kontruktoru vyhledávací komponenty, která se při změně zavolá. Delegaci nelze využít z důvodu zavíslosti na funkci volajícího. Jedná se zase o složení Filtru, Sorteru a SQL Generatoru.

Filtr se skládá ještě z komponenty pro řešení specifických filtrů, jenž se zabírá spravováním N konkrétních položek u filtrů jako jsou napříkald Okres a Kategorie míst.

4.4.3 Clusterování

Zobrazení předmětů ze všech kolekcí na mapě bez clusterování způsobí obrovský nárok na výpočetní sílu. Pokud nedojde k dodání výkonu dochází k zamrznutí a následnému pádu aplikace. Google maps disponují možností připojit ClusterManager.[44] Samotný ClusterManager podle vzoru Stategie umožňuje vybrat algoritmus využitý při clusterování. Lze implementovat své vlastní algoritmy, ovšem k dispozici jsou 4 základní. Jedná se o GridBased, ScreenBased, PreCaching a NonHierarchicalViewBased. Jediný algoritmus, který se dá použít bez omezení výkonu je NonHierarchicalViewBased a to z důvodu, že clusteruje pouze předměty nalézající se v zorném poli mapy. Předměty, které se mají zobrazit v mapě jsou předány samotnému algoritmu a o změně předmětů je potřeba informovat clustermanager, který pak už automaticky propaguje změny do mapy. V sekci 4.1.3.2.3 bylo zmíněno, že se události propagují do map pomocí Flow. Algoritmus společně s Flow, konkrétně SharedFlow umožňující jednoduché vkládání a kolektování hodnot, se nachází v komponentě MapClusterAlgorithmHandler. Handler se pak využívá ve ViewModelu hlavního prohledávání nebo u Prohledávače mapy, kde realiazuje rohraní pomocí delegace, při které se pomocí defaultního prázdného konstruktoru inicializuje.

4.4.4 Prohledávač map

Pro umožnění logiky postupného prohledávání mapy pomocí šipek se využivá komponenta, jenž dokáže mezi dodanými místy přepínat. Základem je rozhraní Clusterovací komponenty, které je pomocí delagace realizováno samotnou komponentou pro clusterování. Rozšířeno o možnosti přepínání mezi místy.

4.4.5 Správa pozice

Protože se poloha zařízení využivá v hodně částích aplikace, je logika pro zobrazení dialogu zabalena do samostatné komponenty.

4.4.6 Správa uživatelových statusů k předmětům

Uživatel si může upravovat zda je předmět vlastněný nebo navštívený z náhledu, seznamu předmětů, plánu a detailu. Pro správu těchto požadavků se využívá StatusHandler, jenž obsahuje potřebné funkce. Jedná se o třídu vyhodnocující čistě podle argumentů a neuchovává si žádné dodatečné informace. Je tedy využit vzor Singleton a pomocí DI jednoduše dodávána závislost.

4.4.7 Import a Export

Pro přehlednost je export rozdělen do více souborů podle entit. Každá entitní třída, která tvoří databázi, splňuje rozhraní určené k importu a exportu. Pro jednoduchou manipulaci s jednotlivým poli se k exportu využívá typ souboru csv, ten umožňuje dělit soubor na řádky a sloupce. Jelikož SQL také vrací řádky je tedy csv ideální ke spolupráci. Každá entitní třída umí vypsat svoji hlavičku, utvořit ze sebe seznam hodnot a poskládat se ze seznamu hodnot. Pro správu importu a exportu se využívá speciální komponenta. Základním principem je seznam objektů, kde každý objekt umí získat všechny záznamy ke konkrétní entitě a také ze řádků seznamů hodnot utvořit seznam entint a ty následně uložit do databáze. Seznam objektů tvoří různé třídy ovšem s rozhraním Table, při iteraci se tedy využívá funkcí rohraní a využívá se zde vzor Composite1.2.1.2.2.

Při exportu se nejprve přidá jméno entitní třídy. Posléze se seznam objektů proiteruje. U každé entity se nejprve získají záznamy, následně se znich udělají řádky a uloží do souboru.

Při importu se podle jména entitní třídy najde konkrétní objekt ze seznamu a využije se funkce jenž zpracuje řádky a uloží je do databáze jako záznamy.

Objektový přístup umožňuje snadné rozšíření o další entity. Stačí aby entita splňovala rozhraní a dále přidat do seznamu objekt s funkcemi k získávání a ukládaní dat.

4.4.8 Správa komentářů

Komponenta se stará o získávání komentáře, ukládání komentáře a zajišťuje, aby uživatel omylem neztratil svůj napsaný výtvor. V případě, že nastane pokus o návrat, ale nejsou uloženy změny, dojde k úpravě stavu pro zobrazení potvrzovacího dialogu. Pro možnost generického využití jak pro předměty, tak pro plány je implementováno pomocí vzoru Template.

4.4.9 Internetové požadavky

Základní rozdělení práce v Androidu je popředí a pozadí. V případě, že žádná z aktivit není viditelná uživateli nebo neběží žádné služby spuštěné z popředí dokud byla nějaká aktivita viditelná, potom aplikace běží na pozadí, v opačném případě běží na popředí. Perzistentní práce by se měly provádět na pozadí.[73] Skupina Jetpack přichází s WorkManagerem, který podle nastavení spouští jednotlivé Workery. Před spuštěním Workerů lze nastavit za jakých okolností se má práce spustit, tedy lze nastavit například aby se práce spouštěly pouze v případě připojení k internetu. Dále se dají nastavit vstupní podmínky pro práci, například je možné zadat textový identikátor kolekce, která se má stáhnout. Workerem může být libovolná třída splňující Worker rozhraní, jedná se pouze o funkci „do“ a pár atributů. Z důvodu potřeby volat suspend funkce z „do“ byl Worker nahrazen CoroutineWorkerem, jehož funkce ”do”je označena jako suspend. Funkce „do“

může skončit 3 možnostmi úspěch, chyba, opakování. V případě, že dojde k úspěchu nebo chybě je práce ukončena a při návratu dochází k opakování práce. Do Workerů jsou pomocí DI vkládány repozitáře a jsou posléze volány požadované funkce. To způsobí běh funkcí na pozadí. WorkManager umožňuje jednotlivé workery skládat do front, určovat co bude probíhat paralelně a co ne.[74] Pomocí WorkManageru jsou prováděny operace Přihlášení, Stahování obecných kolekcí a Stahování uživatelových kolekcí.

4.4.9.1 Stahování kolekcí

Pro potřeby práce existuje objekt držící kolekci párů s identifikátorem konkrétní kolekce a funkce pro stažení kolekce, zde se využívá vzor Command1.2.1.3.2. Podle kolekcí, jenž se mají stáhnout se postupně za sebou spouští stahování.

4.5 Dokumentace, Analýza a Nasazení

Pro možnost procházet aplikaci bez Android Studia, ale zároveň efektivněji než pomocí základních textových editorů je udržována dokumentace aplikace v HTML. Dochází k využití dokumentačního enginu jménem Dokka, umožňujicí spolupráci s Kotlinovským způsobem komentářů. [75]

Pro dodržování konvencí se využívá Detekt, nástroj pro statickou kontrolu kódu. Detekt umožňuje výběr pravidel, popřípadě tvorbu vlastních.[76] K rozšíření pravidel o funkční přešlapy při používání Jetpack Compose se využívá sada pravidel Twitter Compose Rules.[77]

Aplikace je dostupná v Obchodu Play. Aby mohla být publikována muselo být splněno několik požadavků. Prvními bylo vyplnit informace o aplikaci zaměřené na obsah aplikace, součásti toho byla i potřeba vygenerovat zásady ochrany soukromí. Dále bylo potřeba zadat informace a grafiku, jenž se zobrazují v samotném obchodě.

Kapitola 5
Testování

5.1 Metody

S přibývajícím počtem různých aplikací přibývá i počet testovacích metod možných k využití. Z důvodu omezených zdrojů byly pro testování vybrány pouze základní.

5.1.1 Funkcionální testování

První je funkcionální testování, jenž bylo z části prováděno manuálně pomocí logování a je rozděleno na 3 typy. První typem je Unit testování sloužící k testování malých částí kódu v jednotlivých komponentech, automaticky je otestována konverze datumů, rozmístění míst na stejném místě do kruhů a počítání vzdálenosti mezi místy. K automatickému testování se využívá Android JUnit 4.[78] Druhým typem je integrační testování, které slouží ke zjištění funkčnosti spolupráce jednotlivých komponent zde jsou vytvořeny automatické testy pro spojení databáze a obou sql generátorů, třetím typem je End-to-end testování, které se zaobírá celkovou funkčností.[79]

5.1.2 Testování použitelnosti

Druhou využitou metodou je testování použitelnosti. Jedná se o interní testování distribuované aplikace mezi skupinu uživatelů. Jedním z cílů je odladit přehlédnuté chyby, ale hlavním cílem této metody je získat informace o použitelnosti aplikace, díky kterým by šly vymyslet návrhy na zlepšení UX. Kvůli rozsáhlé komunitě okolo turistických známek byl původní záměr vzdálené testování se sběrateli turistických známek.[80]

5.1.3 Distribuce

Jednou z otázek při vymýšlení testování bylo, jak získat testery a jak distribuovat aplikaci mezi ně. Ke kontaktu s uživateli již došlo při tvorbě dotazníku v kapitole analýza. Nebyl důvod měnit přístup a pro získání testerů byl napsán příspěvek do několika skupin na sociálních sítích. Z důvodů pravděpodobného produkčního nasazení aplikace na Google play vyplývá, že je moudré testovat aplikaci už přes samotný Google play a následně některé vydání nastavit jako produkční.

5.1.4 Zadání

Posledním bodem je způsob, kterým budou uživatelé aplikaci testovat. Pro větší pokrytí a také díky časovému skluzu byly vybrány testovací scénáře oproti testovacím případům. Zadávání

pomocí případů je oproti scénářům podrobnější. Ovšem správně nastavené scénáře mohou vést ke kompletnímu pokrytí.[81] Zadání je vytvořeno formou google formuláře obsahující úkoly. Ke každému úkolu tester přidá zda proběhl v pořádku nebo naopak nastal problém. První část se zaměřuje na testování funkčnosti a druhá část se zaměřuje na analýzu podle heuristiky od Jakoba Nielsena.

5.1.5 Scénáře pro funkčnost

Tabulka 5.1 Testovací scénáře

Identifikátor	Scénář
TS-F-1	Zkontrolovat přihlášení jako registrovaný uživatel.
TS-F-2	Zkontrolovat stahování kolekcí jako přihlášený uživatel.
TS-F-3	Vyhledávání, filtrování a řazení předmětů přes hlavní prohledávač (prostřední ikonka na dolní navigaci) a s více kolekcemi.
TS-F-4	Vyhledávání, filtrování a řazení předmětů v konkrétní kolekci.
TS-F-5	Zkontrolovat (získávání, přidávání a mazání) jednotlivých předmětů ze sbírek jako přihlášený uživatel. Mazání je aktuálně kvůli problémům v TZ nedostupné. Akce zkuste z náhledu u mapy, ze seznamu míst, z detailu a z konkrétního plánu.
TS-F-6	Zkontrolovat export.
TS-F-7	Jako přihlášený uživatel zkontrolovat import plánů.
TS-F-8	Zkontrolovat odhlášení (přes profil -¿ ozubené kolečko).
TS-F-9	Zkontrolovat přihlášení přes anonyma a následné přihlášení přes profilovou obrazovku.
TS-F-10	Zkontrolovat stahování kolekcí jako anonym.
TS-F-11	Jako anonymní uživatel zkontrolovat kompletní import.
TS-F-12	Zkontrolovat sekci nově přidané známky a funkčnost propojení se zbytkem aplikace.

TS-F-13	Zkontrolovat (získávání, přidávání a mazání) jednotlivých předmětů ze sbírek jako anonymní uživatel. Akce zkuste z náhledu u mapy, ze seznamu míst, z detailu a z konkrétního plánu.
TS-F-14	Zkontrolovat přesun do map třetích stran.
TS-F-15	Zkontrolovat detail předmětu(změna statusu k předmětu, zobrazení na mapě, přidání komentáře, přidání hodnocení.
TS-F-16	Zkontrolovat funkce plánovacího dialogu z konkrétního předmětu.
TS-F-17	Zkontrolovat přidávání plánů přes plánovač.
TS-F-18	Zkontrolovat vyhledávání, filtrování a řazení plánů přes plánovač.
TS-F-19	Zkontrolovat konkrétní plán (přidávání míst, komentáře, práce s předměty, zobrazení na mapě).
TS-F-20	Zkontrolovat synchronizaci jako přihlášený uživatel.
TS-F-21	Zkontrolovat vzhled obrazovek s telefonem otočeným na výšku.
TS-F-22	Zkontrolovat vzhled obrazovek s telefonem otočeným na šířku.

5.1.6 Scénáře pro heuristickou analýzu podle Jakoba Nielsena

V dotazníku jsou uživatelé informování o 10 principech Jakoba Nielsena a jejich úkolem je vypsat prohřešky u konkrétních designových komponent.

Tabulka 5.2 Testovací scénáře

Identifikátor	Scénář
TS-D-1	Obrazovka přihlášení nebo registrace
TS-D-2	Obrazovka výběr kolekcí ke stáhnutí
TS-D-3	Obrazovka mapa na hlavním prohledávači, týká se i náhledu předmětů - režim profil

TS-D-4	Obrazovka mapa na hlavním prohledávači, týká se i náhledu předmětů - režim širokoúhlý
TS-D-5	Obrazovka pro vyhledávání, filtrování a řazení přes hlavní vyhledávač - režim profil
TS-D-6	Obrazovka pro vyhledávání, filtrování a řazení přes hlavní vyhledávač - režim širokoúhlý
TS-D-7	Obrazovka detail - režim profil
TS-D-8	Obrazovka detail - režim širokoúhlý
TS-D-9	Obrazovka profil
TS-D-10	Obrazovka s vyhledáváním, filtrováním a řazením plánů - režim profil
TS-D-11	Obrazovka s vyhledáváním, filtrováním a řazením plánů - režim širokoúhlý
TS-D-12	Obrazovka pro přidání plánu
TS-D-13	Obrazovka konkrétního plánu - režim profil
TS-D-14	Obrazovka konkrétního plánu - režim širokoúhlý
TS-D-15	Způsob přidávání předmětů do plánů z plánu
TS-D-16	Obrazovka prohlížeč mapy buď jako plán na mapě nebo některá konkrétní kolekce na mapě
TS-D-17	Plánovací dialog dostupný z předmětů
TS-D-18	Způsob přidávání předmětů do plánů z předmětů
TS-D-19	Způsob přidávání komentářů
TS-D-20	Obrazovka informace (hlavně nejnovější známky)

5.1.7 Anaýla odpovědí na testovací scénáře

Velká očekávání byla na zapojení sběretelů do destování. Ovšem opak byl pravdou a do testování se zapojili pouze 3 uživatelé. V první sekci dochází k rozboru jejich odpovědí a ve druhé pak analýzu provádím já sám.

5.1.7.1 Uživatelé

V sekci funkcionality toho uživatelé moc nezmínili, jenom u exportu byla zmínka, že rozdělení na více souborů může být nepřehledné. Ovšem o principech podle Nielsena už toho napsali více. K 1. a 2. bodu se žádný s uživatelů nevyjádřil, ale 3. bod byl kladně vyzdvižen u stáhnování kolekcí, konkrétně, že lze stahnuté kolekce upravit né jen po přihlášení. 4.bod uživatelé zmínili u vyhledání a konkrétního plánu.Ve vyhledávání šlo o nejasnost prázdného místa, při otočení do režimu širokoúhlý, do doby než objevili ve volném místě náhled předmětu. U plánu se zmínili, že zobrazení akcí u předmětů je neintuitivní a přidání komentáře pro předmět by mělo spíš jako přidat komentář místo přidat poznámku. Posledním co testeři zmínili byl bod 7 u mapy, kdy jim přišlo, že rozbalování skupin značek by mělo být efektivnější, tedy na míň kliknutí by se měli dostat k jednotlivým předmětům z clusterů.

5.1.7.2 Autor

- 1. Každá akce spouští reakci, tedy bod první je splněn.
- 2. Na některých místech je slovo předměty nahrazeno slovem známka, což zapřičiňuje porušení 2. principu.
- 3. Při startu když uživatel vybere způsob přihlášení není mu umožněno se vrátit zpátky, sice je možné se pak odhlásit, ale už to porušuje princip.
- 4. Zde souhlasím s testery, plus navíc si myslím, že u výběru více kolekcí při filtrování není jasná možnost vybírat kolekce vybírat při dlouhém podržení a to znamená, že 4. bod je porušen.
- 5. Prevenec chyb je využita a za mě nenacházím žádné místo, které by s principem rozporovalo.
- 6. Není potřeba aby si uživatel pamatoval důležité informace mezi obrazovkami. Snad jenom je potřeba zapamatování si indikátorů jak na mapě tak v seznamu, ovšem na to je pomocná obrazovka, u 6. bodu si tedy nejsem jistý zda jej porušuju.
- 7. Snažil jsem se aby aplikace byla efektivní, ovšem testeři našli problém a já s ním souhlasím, tedy bod 7 je lehce porušen.
- 8. Jedním z hlavních rozdilů oproti dosavadním aplikacím měla být jednoduchost a přehlednost. Například u detailu ve srovnání s neoficiální aplikací na android byl tento bod za mě splněn. Celkově se mi zdá, že aplikace umožňuje hodně akcí a zobrazení hodně informací, ale zároveň si drží svůj minimalistický design.
- 9. Žádné informace v kódovém jazyce nejsou zobrazovány.
- 10. Pro porozumění systému je možné využít komponentu s příklady indikátorů a barev, ovšem některé funkce jako například rozšíření předmětu o akce po kliknutí nebo možnost dlouhého podržení nejsou speciálně uvedeny.

Kapitola 6
Závěr

Aplikace splňuje funkční i nefunkční požadavky, které vzešly z analýzy. Design byl původně zamýšlen jako plně fungující prorotyp, ovšem výsledekem byla pouze částečná funkčnost.

Implementace má několik drobných nedostatků. Některé z nich neovlivňují způsob fungování aplikace, ale jedná se pouze o použité technologie, mezi ně patří využití Google Maps jako View místo Compose komponenty a využití Retrofitu pro tvorbu http požadavků k získání obecných kolekcí. Některé nedostatky ovšem funkčnost ovlivňují, jedním z nich je absence možnosti přidat si předmět do navštívených i když se právě nesplňuje vzdálenost od předmětu, to se dá opravit vytvořením dalšího stavu pro naštívení, ovšem tohle TZ zatím nepodporují a je tedy možnost to udělat pouze lokálně jako plány. Dalším nedostatkem je nucení rozkliku předmětu při prohledávání, vylepšením by bylo umožnit přidání například pomocí gesta drag doleva nebo doprava. Dalšími problémy, jenž by se měly v budoucnu opravit a které pomohli odhalit testeři bylo pomalé zoomování na skupinu značek na mapě a nedodržování konvencí u pojmenování.

Testování nedopadlo vůbec podle předpokladů, oproti očekávanému velkému zájmu se k testování přihlásili pouze 3 lidé. Byly připraveny 2 sady scénářů, které měly odhalit všechny nedostatky. První se zaměřovala na správnost funkcí a druhá na správnost designového návrhu. Aplikace z důvodu nízké protestovanosti může obsahovat hodně nedostatků, které nebyly odhaleny. Ovšem i při nízkému počtu testerů se podařilo odhalit chyby v aplikaci.

Aplikace je však úspěně nasazena v obchodě Play a lze ji využívat.

V současné době Turistické známky,s.r.o., zvyšují zabezpečení a kvalitu způsobu distribuce dat pro mobilní aplikace. To znamená, že se v budoucnu změní způsob získávání dat a aplikace bude nucena projít změnami.

Bibliografie

1. LEŠKA, Šimon. *Co je to UX a UI design?* [Online]. 2020 [cit. 2022-06-19]. Dostupné z: https://www.blueghost.cz/clanek/co-je-to-ux-a-ui-design/.
2. *10 Usability Heuristics for User Interface Design* [online]. 2020 [cit. 2023-02-14]. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/.
3. NEIL, There. *Mobile design pattern gallery* [online]. [B.r.] [cit. 2023-01-25]. Dostupné z: https://www.pdfdrive.com/mobile-design-pattern-gallery-2nd-edition-ui-patterns-for-smartphone-apps-d188892628.html.
4. *Material Design* [online] [cit. 2023-01-25]. Dostupné z: https://www.interaction-design.org/literature/topics/material-design.
5. *Material Design* [online] [cit. 2023-01-25]. Dostupné z: https://material.io/.
6. VONAU, Manuel. *Material You: What it is and what we love about it* [online]. 2022 [cit. 2023-01-25]. Dostupné z: https://www.androidpolice.com/everything-we-love-about-material-you/.
7. *Design Resources* [online] [cit. 2023-01-25]. Dostupné z: https://material.io/resources.
8. GAMMA, E; VLISSIDES, J; JOHNSON, R; HELM, R. *Design Patterns Elements of Reusable Object-Oriented Software*. [B.r.]. Dostupné také z: http://www.javier8a.com/itc/bd1/articulo.pdf.
9. DEURSEN, S.V; SEEMAN, M. *Dependency Injection Principles, Practices, and Patterns*. [B.r.]. Dostupné také z: https://www.pdfdrive.com/dependency-injection-principles-practices-and-patterns-d187686374.html.
10. GREENE, Joshua. *4. Delegation Pattern* [online] [cit. 2023-01-25]. Dostupné z: https://www.kodeco.com/books/design-patterns-by-tutorials/v3.0/chapters/4-delegation-pattern.
11. *Guide to app architecture* [online] [cit. 2023-01-25]. Dostupné z: https://developer.android.com/topic/architecture#recommended-app-arch.
12. *ViewModel overview* [online] [cit. 2023-01-25]. Dostupné z: https://developer.android.com/topic/libraries/architecture/viewmodel.
13. *State holders and UI State* [online] [cit. 2023-01-25]. Dostupné z: https://developer.android.com/topic/architecture/ui-layer/stateholders.
14. *Data layer* [online] [cit. 2023-01-25]. Dostupné z: https://developer.android.com/topic/architecture/data-layer.
15. *Data Access Object Pattern* [online] [cit. 2023-02-02]. Dostupné z: https://www.tutorialspoint.com/design_pattern/data_access_object_pattern.htm.

16. *What is Reactive Programming and How Can I Benefit from It?* [Online] [cit. 2023-02-01]. Dostupné z: https://www.codemotion.com/magazine/backend/benefits-of-reactive-programming-codemotion-magazine/.
17. *What is OAuth 2.0?* [Online] [cit. 2023-02-05]. Dostupné z: https://auth0.com/intro-to-iam/what-is-oauth-2.
18. *What is a Native Mobile App Development?* [Online]. 2021 [cit. 2023-01-26]. Dostupné z: https://mdevelopers.com/blog/what-is-a-native-mobile-app-development-.
19. *5 Key Benefits of Native Mobile App Development* [online] [cit. 2023-01-26]. Dostupné z: https://clearbridgemobile.com/benefits-of-native-mobile-app-development/.
20. *What is Framework?* [Online] [cit. 2023-01-31]. Dostupné z: https://www.codecademy.com/resources/blog/what-is-a-framework/.
21. *Application Fundamentals* [online] [cit. 2023-01-25]. Dostupné z: https://developer.android.com/guide/components/fundamentals.
22. *Comparison to Java* [online]. 2023 [cit. 2023-01-25]. Dostupné z: https://kotlinlang.org/docs/comparison-to-java.html#some-java-issues-addressed-in-kotlin.
23. *Is Kotlin compatible with the Java programming language?* [Online] [cit. 2023-02-02]. Dostupné z: https://kotlinlang.org/docs/faq.html#is-kotlin-compatible-with-the-java-programming-language.
24. ERIK, T Ray. *Learning XML* [online]. [B.r.] [cit. 2023-01-26]. Dostupné z: https://repo.zenk-security.com/Programmation/Learning%5C%20XML.pdf.
25. *Introducing JSON* [online] [cit. 2023-01-26]. Dostupné z: https://www.json.org/json-en.html.
26. *4 reasons Jetpack Compose is better than XML* [online]. 2022 [cit. 2023-01-26]. Dostupné z: https://medium.com/@cybercoder.naj/4-reasons-jetpack-compose-is-better-than-xml-ac0efd12db28.
27. *Say Hello to Jetpack Compose and Compare with XML* [online]. 2021 [cit. 2023-01-26]. Dostupné z: https://blog.kotlin-academy.com/say-hello-to-jetpack-compose-and-compare-with-xml-6bc6053aec13.
28. *Measuring Render Performance with Jetpack Compose* [online]. 2021 [cit. 2023-01-26]. Dostupné z: https://engineering.premise.com/measuring-render-performance-with-jetpack-compose-c0bf5814933.
29. *Comparing Jetpack Compose performance with XML* [online]. 2022 [cit. 2023-01-26]. Dostupné z: https://medium.com/okcredit/comparing-jetpack-compose-performance-with-xml-9462a1282c6b.
30. ALMUTTAIRI, Dr. Rafah M. *What's the Difference Between the FrontEnd and Back-End?* [Online]. [B.r.] [cit. 2023-01-27]. Dostupné z: https://www.uobabylon.edu.iq/eprints/publication_4_27425_1402.pdf.
31. *World Wide Web* [online] [cit. 2023-01-27]. Dostupné z: https://www.britannica.com/topic/World-Wide-Web.
32. *What is HTTP?* [Online] [cit. 2023-01-27]. Dostupné z: https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/.
33. *What is Rest?* [Online] [cit. 2023-01-27]. Dostupné z: https://www.codecademy.com/article/what-is-rest.
34. *What is REST API?* [Online] [cit. 2023-01-27]. Dostupné z: https://www.redhat.com/en/topics/api/what-is-a-rest-api.
35. *What is SQL?* [Online] [cit. 2023-01-31]. Dostupné z: https://aws.amazon.com/what-is/sql/.

36. *What is Database?* [Online] [cit. 2023-01-31]. Dostupné z: https://www.oracle.com/database/what-is-database/.
37. ELISKA, Kuzdasová. *Aplikace pro sberatele turistických známek na OS Android*. 2013. Bachelor's Thesis. Fakulta informačních technologií České vysoké učení technické.
38. ROHIT, Sharma. *What is Clustering and Different Types of Clustering Methods* [online]. 2022 [cit. 2023-01-26]. Dostupné z: https://www.upgrad.com/blog/clustering-and-types-of-clustering-methods/.
39. *Mapy.cz API/SDK* [online] [cit. 2023-01-26]. Dostupné z: https://vyvojari.seznam.cz/mapy.
40. *Maps SDK for Android* [online] [cit. 2023-01-26]. Dostupné z: https://www.mapbox.com/pricing#maps-sdks-for-mobile.
41. *Maps SDK for Android* [online] [cit. 2023-01-26]. Dostupné z: https://docs.mapbox.com/android/maps/guides/.
42. *Maps SDK for Android* [online] [cit. 2023-01-26]. Dostupné z: https://github.com/openmobilemaps/maps-core.
43. *Maps SDK for Android Usage and Billing* [online] [cit. 2023-01-26]. Dostupné z: https://developers.google.com/maps/documentation/android-sdk/usage-and-billing#mobile-dynamic.
44. *Google Maps Android Marker Clustering Utility* [online] [cit. 2023-01-26]. Dostupné z: https://developers.google.com/maps/documentation/android-sdk/utility/marker-clustering.
45. HITESH, Chopra. *Ktor-Client Decoded!! What, Why, How and When????* [Online]. 2022-01-17 [cit. 2023-02-02]. Dostupné z: https://proandroiddev.com/ktor-client-decoded-what-why-how-and-when-c6dca7559390.
46. *Retrofit* [online] [cit. 2023-02-02]. Dostupné z: https://square.github.io/retrofit/.
47. *Ktor* [online] [cit. 2023-02-02]. Dostupné z: https://ktor.io/docs/welcome.html.
48. *Android Databases — 6 Important Considerations* [online] [cit. 2023-02-02]. Dostupné z: https://realm.io/best-android-database/.
49. *What Is SQLite?* [Online] [cit. 2023-02-02]. Dostupné z: https://www.sqlite.org/index.html.
50. *Room* [online] [cit. 2023-02-02]. Dostupné z: https://developer.android.com/jetpack/androidx/releases/room.
51. *Save data in a local database using Room* [online] [cit. 2023-02-02]. Dostupné z: https://developer.android.com/training/data-storage/room.
52. *Write asynchronous DAO queries* [online] [cit. 2023-02-02]. Dostupné z: https://developer.android.com/training/data-storage/room/async-queries.
53. IVAN, Garza. *Kotlin dependency injection: Koin vs. Hilt* [online]. 2022-11-11 [cit. 2023-02-02]. Dostupné z: https://blog.logrocket.com/kotlin-dependency-injection-koin-vs-hilt/.
54. *Hilt and Dagger* [online] [cit. 2023-02-02]. Dostupné z: https://developer.android.com/training/dependency-injection/hilt-android#hilt-and-dagger.
55. TIAGO, Fonseca. *RxJava VS Coroutines: Which One Should You Pick? (Part I)* [online]. 2020-05-10 [cit. 2023-02-02]. Dostupné z: https://exaud.com/rxjava-vs-coroutines/.
56. TIAGO, Fonseca. *RxJava VS Coroutines: Which One Should You Pick? (Part II)* [online]. 2020-05-16 [cit. 2023-02-02]. Dostupné z: https://exaud.com/rxjava-vs-coroutines-part-ii/.

57. *Coroutines basics* [online] [cit. 2023-02-02]. Dostupné z: https://kotlinlang.org/docs/coroutines-basics.html.
58. *Asynchronous Flow* [online] [cit. 2023-02-02]. Dostupné z: https://kotlinlang.org/docs/flow.html.
59. *Navigation bar* [online] [cit. 2023-02-06]. Dostupné z: https://m3.material.io/components/navigation-bar/guidelines.
60. *Progress Indicators* [online] [cit. 2023-02-07]. Dostupné z: https://m3.material.io/components/progress-indicators/overview.
61. *Accompanist* [online] [cit. 2023-02-11]. Dostupné z: https://github.com/google/accompanist.
62. *Implement dark theme* [online] [cit. 2023-02-10]. Dostupné z: https://developer.android.com/develop/ui/views/theming/darktheme.
63. *System UI Controller for Jetpack Compose* [online] [cit. 2023-02-11]. Dostupné z: https://google.github.io/accompanist/systemuicontroller/.
64. *Google Maps Platform Terms of Service* [online] [cit. 2023-02-10]. Dostupné z: https://cloud.google.com/maps-platform/terms/.
65. *Jetpack Navigation Compose Animation* [online] [cit. 2023-02-11]. Dostupné z: https://google.github.io/accompanist/navigation-animation/.
66. *Navigating with Compose* [online] [cit. 2023-02-11]. Dostupné z: https://developer.android.com/jetpack/compose/navigation.
67. *Jetpack Navigation Compose Material* [online] [cit. 2023-02-11]. Dostupné z: https://google.github.io/accompanist/navigation-material/.
68. *WebView wrapper for Jetpack Compose* [online] [cit. 2023-02-11]. Dostupné z: https://google.github.io/accompanist/web/.
69. *Pager layouts* [online] [cit. 2023-02-11]. Dostupné z: https://google.github.io/accompanist/pager/.
70. *What is a Single Source of Truth (SSOT)* [online] [cit. 2023-02-12]. Dostupné z: https://www.mulesoft.com/resources/esb/what-is-single-source-of-truth-ssot.
71. *DataStore* [online] [cit. 2023-02-15]. Dostupné z: https://developer.android.com/topic/libraries/architecture/datastore.
72. *Gradle plugin to deeply hide secrets on Android* [online] [cit. 2023-02-12]. Dostupné z: https://github.com/klaxit/hidden-secrets-gradle-plugin.
73. *Background Work Overview* [online] [cit. 2023-02-12]. Dostupné z: https://developer.android.com/guide/background.
74. *Define work requests* [online] [cit. 2023-02-12]. Dostupné z: https://developer.android.com/guide/background/persistent/getting-started/define-work#Overview.
75. *Dokka* [online] [cit. 2023-02-13]. Dostupné z: https://github.com/Kotlin/dokka.
76. *Detekt* [online] [cit. 2023-02-13]. Dostupné z: https://detekt.dev/.
77. *Twitter's Jetpack Compose Rules* [online] [cit. 2023-02-13]. Dostupné z: https://github.com/twitter/compose-rules.
78. *Build local unit tests* [online] [cit. 2023-02-13]. Dostupné z: https://developer.android.com/training/testing/local-tests.
79. ANKIT. *12 Types of Mobile Testing With Real-life Examples (2022 Updated)* [online]. 2022 [cit. 2023-01-23]. Dostupné z: https://testsigma.com/blog/different-types-of-mobile-testing-with-real-life-examples/#What_are_different_types_of_mobile_testing.

80. *Usability testing* [online] [cit. 2023-01-23]. Dostupné z: https://www.optimizely.com/optimization-glossary/usability-testing/.

81. THOMAS, Hamilton. *Test Case vs Test Scenario – Difference Between Them* [online]. 2022 [cit. 2023-01-24]. Dostupné z: https://www.guru99.com/test-case-vs-test-scenario.html.

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `apk` ..... spustilený balíček
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
- `dotazníky` ..... dotazníky v csv formátu