Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Prototyp klientské aplikace pro komunitní překlad textů z kulturních institucí

Plzeň 2022 Jan Pelikán

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2021/2022

Studijní program: Informatika a výpočetní technika
Forma studia: Prezenční
Specializace/kombinace: Informatika (INF18bp)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	**Jan PELIKÁN**
Osobní číslo:	**A19B0157P**
Adresa:	Blatenská 28, Plzeň – Lobzy, 32600 Plzeň 26, Česká republika
Téma práce:	Prototyp klientské aplikace pro komunitní překlad textů z kulturních institucí
Téma práce anglicky:	Prototype of client application for the community translation system of the cultural institution texts
Vedoucí práce:	Ing. Richard Lipka, Ph.D. Katedra informatiky a výpočetní techniky

Zásady pro vypracování:

- Seznamte se se stávajícími technologiemi a aplikacemi mobilních průvodců a poskytovanými službami.
- Seznamte se s problematikou komunitního překladu.
- Navrhněte prototyp klienta mobilního průvodce s možností získávání dat z kulturních institucí.
- Navržený prototyp implementujte pro vybranou platformu.
- Vytvořenou implementaci otestujte.

Seznam doporučené literatury:

Dodá vedoucí práce.

Podpis studenta: Datum:

Podpis vedoucího práce: Datum:

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 25.4.2022

Jan Pelikán

Abstract

This bachelor's thesis aims to design a mobile application that enables museum visitors to interpret and experience information about exhibits in their chosen language. The thesis deals with the development and functionality of the design of this application, developed for mobile phones and devices with the Android operating system. The theoretical part outlines the possible features of the application, e.g., analysis of features, geolocation, augmented reality, and conversion from written to spoken text. The practical part of the thesis is the description of the implementation, i.e., the actual development of the application prototype. The empirical part of the thesis documents user testing of the application design with a summary of individual respondents' reports and findings and their recommendations for the work developed in the thesis.

Abstrakt

Cílem této bakalářské práce je návrh pro vývoj mobilní aplikace, která by měla umožnit návštěvníkům muzeí elektronickou interpretaci informací o exponátech v jich zvoleném jazyce. Práce se zabývá vývojem a fungováním návrhu této aplikace, určena je pro mobilní telefony s operačním systémem Android.

Teoretická část vymezuje popis věcí a úkonů, které aplikace používá nebo by mohla využívat, např. analýza možností, získání GPS polohy, rozšířená realita, převod z textu psaného do mluveného. Nejrozsáhlejší částí práce je popis implementace, tedy samotný vývoj aplikace a její struktura. Třetí část práce se zabývá uživatelským testováním návrhu aplikace. Zde se popisují jednotlivé zprávy respondentů, shrnutí poznatků a jejich případné doplnění do práce.

Obsah

1 Úvod

Pro svou bakalářskou práci jsem si vybral téma Mobilní aplikace pro implementaci komunitního překladu v muzeích. Téma je výstižné vzhledem k aktuální pokrokové a moderní době. Dnes už žije jen hrstka lidí, která by nevlastnila mobilní telefon. Ten, kdo nechce zůstat pozadu, a chce umět používat moderní technologie, musí se přizpůsobit, takzvaně jít s dobou. Společnost si již zvykla na technologická zařízení, jako je počítač či mobilní telefon a těžko by bez nich dokázala žít. Telefon je laicky taková „malá“ přenosná skříňka sloužící pro naše pohodlí, komunikaci, práci, zábavu a mnohdy i vzdělávání. V posledních letech se společnost snaží poznávat svět, cestuje, provozuje turistiku, jelikož jsou tyto činnosti velmi dostupné. Právě mobilní zařízení máme vždy s sebou a používáme ho jako mobilního průvodce. Dalo by se říci, že zcela nahradil papírové mapy, brožury či knižní průvodce. Díky lokalizační funkci nám telefon určí naší přesnou polohu, zobrazí věci kolem nás, jako například památky, restaurace, obchody, galerie a muzea. Cílem této bakalářské práce je co nejvíce přiblížit exponáty v instituci, i když zrovna tištěný projev není v jazyce, kterým mluví.

Bakalářská práce je tedy návrh prototypu mobilní aplikace, která by propojila cestování s poučením. V muzeích jsou popisky k exponátům většinou ve dvou až třech jazycích, v menších muzeích tomu tak být nemusí, jelikož nemají prostor či finanční prostředky na zajištění překladu do cizího jazyka. Představovaná aplikace by měla návštěvníka provést muzeem a představit mu jednotlivé exponáty v jazyce jeho výběru, (pokud je k dispozici.) Aplikace funguje pouze na zařízeních s operačním systémem Android a je doplněním k webové aplikaci. Po stažení se uživatel, pro běžné používání, nemusí přihlašovat ani si vytvářet žádný profil. Pokud však chce uživatel přispívat překlady, profil si musí vytvořit, zároveň samostatný překlad nejde vytvořit v mobilní aplikaci, ale pouze ve webové aplikaci. Získávání překladů funguje na bázi komunitního překladu, do kterého má možnost přispět každý registrovaný uživatel aplikace. Uživatel si může vybrat ze dvou možností získání překladu, QR kód či jiný typ kódu, který se nachází vedle exponátu, nebo ruční volení exponátu, ten lze získat postupem od obecného označení k přesnému. Aplikace nabízí seznam všech podporovaných institucí i s jejich bližšími informacemi.

2 Dostupné technologie pro mobilní aplikace a jejich vývoj

Tato kapitola se zabývá analýzou dostupných knihoven, které by mohli sloužit pro vývoj interaktivního mobilního průvodce. V této kapitole získáte přehled o volně dostupných technologiích, které případně lze využít při vývoji aplikace. Daná znalost nám pomůže při návrhu a programování průvodce.

2.1 Augmented reality

Augmented reality (AR), nebo také česky Rozšířená realita je v dnešní době často využívanou technilogií na poli mobilních zařízení. Pojďme si rozšířenou realitu blíže představit. O co se vlastně jedná a jaké má využití v mobilních aplikacích. Rozšířená realita je způsob zobrazení virtuálně vytvořených objektů do reálného prostoru. Pod virtuálními objekty si můžeme představit pohyblivé i nehybné doplňující informace k realitě, případně i „vylepšené“ objekty z reálného světa, může se jednat i o audio nebo pouze o fotky. Vytvoření těchto objektů závisí na podnětu z reálného světa. Mobilní fotoaparát, respektive kamera, nasnímá reálný objekt, který slouží jako podnět právě pro vytvoření virtuálního objektu.

Dnes můžeme vidět rozšířenou realitu využitou například ve skladech, kde se využívá k automatizaci práce. Rozšířená realita se může využít ke kompletaci objednávek, protože všechna data o položkách v krabici lze získat v díky aplikaci, aniž by byla potřeba krabici otvírat. Díky tomu lze urychlit a zpřesnit práci ve skladech. Jedna z největších logistických firem, DHL, uvádí, že implementace rozšířené reality pomohla jejich efektivitě až o 25 %.
Dalšími příklady využití mohou být: edukativní aplikace (Experience Real History -> využití obrázků a virtuálních karet k zlepšení znalostí z historie), hry (Pokémon GO -> možnost zobrazení pokémonů v reálném prostředí) nebo sociální sítě (Facebook, Instagram nebo Snapchat -> využívají filtry pro vylepšení fotek uživatelů).

Ovšem aby vylepšená realita správně fungovala, je potřeba aby zařízení splňovalo minimální požadavky. Mobilní zařízení musí podporovat lokační služby (senzory náklonu a polohy nebo GPS), fotoaparát a některé aplikace můžou vyžadovat připojení k internetu. Zároveň mobilní telefon musí mít dostatečný výkon, aby byl schopný generovat a využít rozšířenou realitu.

Nyní bych rád přiblížil mnou vybrané knihovny pro AR:

2.1.1 Knihovna ARCore

Jedná se o velkou knihovnu od firmy Google, je dostupná zdarma v celém rozsahu. Minimální požadavky jsou Android verze 7.0 a vyšší a API level 24. Díky tomuto požadavku je tato knihovna spustitelná na 73,7% zařízení s Android operačním systémem [11]. Knihovna má širokou škálu funkcionalit jako například: nastavení správného světla přidaného objektu, tak aby správně „zapadl“ do reality, vylepšit (rozpohybovat) obrazy i obličeje a zajištění správného postavení objektu, to

znamená objekt, který má být v pozadí je překrýván tím v popředí. Knihovna nepodporuje rozpoznávání 3D objektů, ovšem na GitHubu už je toto téma v issues [1]. Díky tomu, že je tato knihovna od Googlu, je zde obsáhlá dokumentace i se zdrojovými kódy pro Android (Java/Kotlin).

2.1.2 Knihovna ARKit

Další velká knihovna pro rozšířenou realitu. Obsahuje dosti podobné funkce jako předešlá knihovna ARCore, knihovna je obohacena o práci s 3D objekty a jejich rozpoznáváním, kterým zatím knihovna ARCore nedisponuje. 3D objekty by se v našem případě hodily, protože bychom mohli potřebovat skenovat například sochu či bustu. Knihovna je vytvořená a vylepšovaná firmou Apple, obsahuje rozsáhlou dokumentaci obohacenou o „best practices“ použití. Tím, že je knihovna od Apple, funguje pouze pro iOS a iPadOS.

2.1.3 Wikitude

Jedná se o all-in-one framework, který podporuje jak Android, tak iOS. Je to hlavní produkt firmy Wikitude. Má stejné funkcionality jako předešlé dvě knihovny, k tomu přidává své další, kterými jsou: Geolokace uživatele, implementace 3D objektů i s jejich skenováním a dokáže sledovat a více objektů zároveň a reagovat na jejich změny. Cena licence začíná od 2490 eur pro jedno použití nebo roční předplatné za 2990 eur [2].

2.1.4 Shrnutí

Z výše vypsaných knihoven, jejich funkcí a ceny vyplývá, že pro implementaci rozšířené reality (AR) by jedinou vhodnou knihovnou pro náš případ bylo ARCore. Samozřejmě existují i jiné knihovny, které nejsou tak populární ani vyspělé. Lze si i zakoupit řešení, které je dělané přímo na míru zákazníkovo požadavkům. Bohužel ARCore neumí rozpoznávat 3D modely, tudíž by muselo být „obohaceno“ o nějaký skener, který by dokázal rozpoznávat objekty. Tento nedostatek by se mohl projevit například, kdyby se v muzeu nacházeli exponáty, které bychom chtěli rozpoznat pouze podle „vzhledu“. Implementace a spojení skeneru a ARCore by bylo složité. Možným využitím rozšířené reality by bylo, že data, která by přišla ze serveru (například po rozpoznání exponátu skener, či naskenování QR kódu) by se zobrazila jako text/fotka do volného prostoru vedle exponátu, tuto funkcionalitu ARCore umí a dal by se případně k tomuto využít.

2.2 QR kód

QR kód je v dnešní době velice rozšířenou technologií pro předávání dat v jednoduché a dostupné formě. Kód může naskenovat a použít kdokoliv, kdo vlastní chytrý mobilní telefon s fotoaparátem pro skenování. Jejich pomocí se dá vytvořit odkaz na webovou stránku nebo předat pouze jen text. Do QR kódu se dá zakódovat až 4296 alfanumerických znaků případně 7089 čistě numerických znaků.

QR kódy by se v našem případě dali využít tím způsobem, že u každého exponátu by se nacházel jeden QR kód, který by se dal snadno a rychle naskenovat. Výsledkem skenu by mohl být nějaký indetifikátor, díky kterému by server jednoznačně věděl, o který exponát se jedná. Mohl by díky tomu rychle a vždy

správně odeslat uživateli data o exponátu, nejspíše se bude jednat o text. Vytvoření QR kódu jako takového nic nestojí, jediné náklady by byli na vytisknutí QR kódů.

Chtěl bych vypsat některé dostupné knihovny, které by se dali využít pro zpracování QR kódu. Většina knihoven, co jsem našel, se nacházejí pouze na GitHubu, a proto k nim je pouze limitované množství informací:

2.2.1 Knihovna Barcode scanner

Jedná se o nejstarší knihovnu z níže vypsaných, která byla vyvíjena od roku 2013. Základ knihovny je postaven na ZXing („Zebra Crossing“) knihovně. Z této knihovny vychází, mnoho nových knihoven pro čtení QR kódu či čárových kódů. Podporuje pouze základní funkcionality. Od roku 2020 již není udržovaná a updatovaná. Výše zmiňovaná knihovna ZXing má podobné funcionality a také již není aktivně vylepšovaná. Použití by bylo podobné. Odkaz na Github s knihovnou Barcode scanner [3]. Odkaz na Zxing [4].

2.2.2 Knihovna Code scanner

Základ knihovny je postaven na předešlé knihovně **ZXing**. Tato knihovna navíc podporuje přední a zadní kameru, přizpůsobitelný snímač QR kódu, orientaci kamery na výšku a šířku a autofocus. Funguje pro Android API 19+, verze 4.4 KitKat. Díky tomuto se dá knihovna využít na 98,1% zařízení. V popisu knihovnu je detailní popis použít, jak v Kotlinu, tak i pro Javu. Odkaz na Github [5].

2.2.3 Knihovna MobileVisionBarcodeScanner

Tato knihovna je hodně podobná té předešlé (Code scanner). Knihovna podporuje autofocus, vykresluje čtverec kolem aktuálně skenovaného kódu. Výhoda, že uživatel ví, zda se daří skenovaný kód zachytit. Odkaz na knihovnu [6].

2.2.4 Porovnání s čárovým kódem

Srovnání s kódem čárovým typu EAN-13 případně EAN-8. QR kód nám dokáže zakódovat mnohem více informací než standardní čárový kód, jenže v našem případě nám stačit pouze identifikační číslo výrobku, které se odešle na server a v databázi se exponát vyhledá. V našem případě se do čárového kódu typu EAN-8 dá zakódovat 7 číslic, které jednoznačně určí exponát, to pro náš případ je dostatečné.

Na druhou stranu, skenování QR kódu je pro uživatele jednoduší, protože se dá skenovat například i vzhůru nohama. To znamená, že po uživateli nechceme, aby musel otáčet telefon do různých úhlů. Zároveň je v dnešní době používání QR kódu moderní, a většině uživatelům tak rychle dojde, že zrovna toto mají naskenovat.

2.2.4 Shrnutí

QR kód by z mého pohledu mohl být dobrým nástrojem pro použití, při implementaci mobilního průvodce pro jednoznačné určení exponátu. Do QR kódu by se dal zakódovat speciální řetězec, který by serveru jednoznačně určil, o který exponát se jedná. Všechny z výše vypsaných knihoven by se dali pro realizaci

využít. Osobně jsem si práci s těmito knihovnami zkoušel a nejlépe se mi pracovali s druhou knihovnou (Code scanner).

V porovnání s čárovým kódem si myslím, že z pohledu uživatele bude více „user friendly“ použití QR kódu, v dnešní době jsou QR kódy na hodně místech použity a uživatel si rychle spojí, že QR kód má naskenovat. Výsledek bude záviset na preferenci Západočeského muzea.

2.3 Text to speech

Česky lze říkat „Text na řeč“ nebo lépe „Syntéza řeči“. Programy používané pro syntézu řeči produkují slova, v lepším případě celé věty přesně tak, jak je psané v textu, který čtou. Využívají k tomu databáze, kde jsou uloženy fóny, difóny nebo celá slova. Díky této interpretaci program dokáže poskládat celá slova a věty v jeden celek.

Implementovat funkci text to speech do mobilního průvodce by šlo tím způsobem, že by uživateli byla nabídnuta možnost, zda daný přeložený text chce přečíst. Ze serveru přijde odpověď v podobě textu k danému exponátu, uživatele bychom se zeptali, zda se chce daný přeložený text přečíst sám, nebo pokud má sluchátka, zda text chce přečíst pomocí „aplikace“.

Podstatným bodem text-to-speech (TTS) u naší aplikace může být potřeba aby více uživatelů v jeden moment využilo tuto funkci. Dobrým způsobem, jak toto vyřešit by byl offline TTS, každému uživateli by se na vlastním telefonu vytvořil zvuk a nemusel by odesílat žádný požadavek pro službu TTS. Ovšem, nepodařilo se mi najít žádnou knihovnu, která by toto podporovala kromě již implementované třídy TextToSpeech, která je implicitně v android telefonech. Knihovna nepodporuje český jazyk, proto by funkce pro češtinu nemohla fungovat. Pro cizince už by TTS mohlo fungovat, protože knihovna podporuje angličtinu, francouzštinu, němčinu, italštinu a španělštinu. Jedná se o implicitně implementovanou technologií, proto není potřeba žádná licence. Odkaz na TTS [7].

Druhým možným způsobem by bylo využít službu TTS online, tím způsobem, že by uživatel obdržel text ze serveru, a poté by odeslal požadavek na službu TTS, že chce daný text přečíst. Problémem tohoto řešení je možná zátěž serveru. V případě, že přijede zájezd cizinců a všichni budou chtít využít tuto službu v cizím jazyce, bude se jednat o desítky požadavků během minuty. Tento případ je pravděpodobný v muzeích a galeriích a pro server není zanedbatelný. Všechny služby podporující online požadavky pro službu TTS mají zpoplatněny zpracování většího objemu dat. Zmíním pouze jednu knihovnu, která zdarma nabízí nejvíce služeb.

Posledním možným způsobem řešení TTS by bylo doporučit uživateli externí aplikaci, která by byla volně dostupná v Google play obchodě. Text, který by se měl číst (informace o exponátu), by se uživateli zkopíroval do schránky a poté by se uživatel sám mohl zvolit, jestli přejde do externí aplikace či nikoli.

2.3.1 Voice RSS

Služba od této společnosti nijak nelimituje počet požadavků za minutu, nebo nelimituje počet uživatelů. Jediným měřítkem je počet požadavků za den, kterým je 350 ve verzi, která je zdarma. Toto číslo není úplně malé a v raných verzích aplikace by určitě stačilo, ale nemůžeme se na toto spolehnout.

Služba nabízí TTS SDK pro Android i jiné jazyky, a komunikuje dále přes webové protokoly HTTP GET a HTTP POST. Nabízí přes 45 světových jazyků včetně češtiny, němčiny a angličtiny. Více možných hlasů u každého z jazyků.

2.3.2 Shrnutí

Funkcionalita text-to-speech by byla uživatelsky přívětivá a dle mého i oblíbená, protože již v dnešní době existují pro muzeích a památkách audio průvodci. Tato featura by mohla z naší aplikace udělat „volnějšího“ audio průvodce, pokud by o to měl uživatel zájem. Výhodou by bylo více možnost, jak možnost čtení textu samotným uživatelem, tak možnost TTS.

Pokud by se funkce TTS implementovala do aplikace, s největší pravděpodobností bude knihovna Androidu. Funkce by fungovala pouze pro turisty, kteří nemluví českým jazykem.

Jinou možností, jak využít text-to-speech by bylo pouhé doporučení již předem vybrané aplikace, kterou by si uživatel stáhl. Naše aplikace by už jen text k přečtení zkopírovala do schránky a poté by si jen uživatel daný text vložil do aplikace a nechal přečíst.

2.4 NFC tagy

Near field communication je technologie pro bezdrátový přenos dat mezi elektrickými zařízeními na krátkou vzdálenost. V dnešní době se využívá například pro bezkontaktní platební terminály. NFC může zprostředkovat obousměrnou výměnu dat, či pouze jednosměrnou. Při použití jednosměrného přenosu dat, se říká že se jedná o pasivní NFC „tag“.

Pro náš účel by byl vhodný právě tento jednosměrný způsob komunikace. NFC tag by sloužil jako identifikátor daného exponátu podobně jako QR kód či čárový kód. Při přiložení telefonního zařízení k NFC tagu by se předal požadavek na server, který by podle identifikátoru věděl, který text z databáze má vrátit.

Výhody NFC tagu:

Přepisovatelnost – data uložená v NFC tagu je možné kdykoli přepsat (pokud se jedná o přepisovatelný tag). Tato funkce se dá využít třeba u expozic, které mají omezenou dobu trvání. NFC tag by se jednoduše po ukončení expozice přepsal a mohl by být přesunut k jinému exponátu. Naproti tomu u QR kódu/čárového kódu tato možnost není a byla by potřebovat vytvořit kód nový.

Skenování – v některých situacích může být pro uživatele jednoduší pouze telefon přiložit k zařízení než správně zaměřit fotoaparát a čekat, než se naskenuje kód. Pro starší uživatele aplikace to může být celkově jednodušší či pokud uživatel nemůže zrovna využít druhou ruku, aby skenování proběhlo jednoduše.

Nevýhody NFC tagu:

Cena – cena jednoho neprogramovatelného NFC tagu se pohybuje od 15 Kč. Zatímco programovatelný stojí od 25 Kč [8] (15.11.2021). Tento NFC tag je nepřepsatelný a svoji hodnotu má již v sobě vloženou od výrobce. Oproti tomu jediný náklad pro QR/čárový kód je papír a tiskárna, přičemž na jeden papír se dá vytisknout více kódů. Z toho vyplývá, že výroba kódu je levnější.

Vytvoření – připravení NFC tagu jako takové je značně obtížnější než třeba QR/čárový kód. Pokud by se zvolil způsob pomocí NFC tagů, samotný tag se musí objednat/koupit, čekat, než tag přijde, a následně do něj zakódovat identifikátor pro server. Při použití QR kódu stačí lze využít internetové generátory, kam se zadá identifikátor a QR kód se vygeneruje a následně jen vytiskne. Tento způsob je méně náročný a rychlejší.

2.4.1 Shrnutí

Použití NFC tagu by bylo výhodné, pokud by muzeum často měnilo expozice a exponáty. Díky tomu, že NFC tag se dá jednoduše přepsat, bylo by to jednodušší než se starat o nové tisknutí kódů. Na druhou stranu, připravit a vytisknout kód není o tolik náročnější než přepsat NFC tag. Při použití NFC tagu by nebyla potřeba implementovat čtečku kódů do aplikace.

2.5 GPS

GPS neboli globální polohový systém je technologie, která umožňuje pomocí elektronického přijímače určit přesnou polohu na povrchu země. Celá technologie funguje na principu přijímání signálu. Elektronické zařízení (mobil, navigace) obsahuje přijímač, který přijímá signál z družic, které obíhají zemi, díky tomu družice dokáží vypočítat pozici přijímače na jednotky metrů.

Pro naši aplikaci by se tento systém dal využít k návrhu muzeí, ve kterém se uživatel nachází. Uživatel bude chtít vybrat ve kterém se nachází muzeu, při tomto požadavku by bylo vhodné seřadit muzea ve vzdálenosti od uživatele od nejbližšího muzea po nejvzdálenější. Tato funkce by ulehčila hledání muzea, protože hledané muzeum by bylo vždy mezí prvními výsledky.

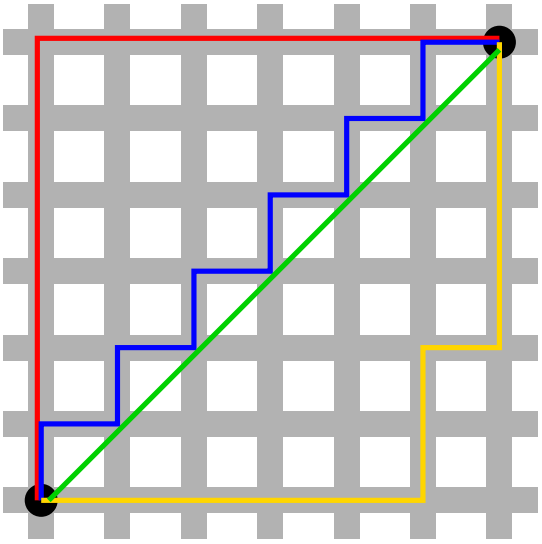
Pokud bude v databázi muzeí málo, můžeme počítat vzdálenost pro každé muzeum při každém požadavku. Ovšem až se databáze rozroste, tento požadavek bude dlouhý, bude se jednat o sekundy až desítky sekund. To pro uživatele není dobré. Níže bych chtěl popsat, jakým způsobem lze tento problém řešit.

Možnosti výpočtu vzdálenosti:

Euklidovská vzdálenost – Jedná se o vzdálenost, kterou bychom mohli nazvat také vzdušnou čarou, jedná se o nejkratší cestu z jednoho bodu do druhého. V prostoru je tato vzdálenost počítána jako odmocnina ze sumy rozdílů druhých mocnin každé souřadnice zvlášť. Pro lepší představu, máme dva body A a B, každý z nich má souřadnici x a y. Obecný vzorec by vypadal takto: $m_e(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$. Prakticky by proběhli dva výpočty, jednou pro souřadnici x a po druhé pro souřadnici y. Tento způsob je výpočetně složitý, protože se počítají druhé mocniny a zároveň odmocniny.

Manhattan vzdálenost – Jméno této vzdálenosti je odvozeno od ostrova Manhattan v New Yorku, kde většina ulic je k sobě kolmá a tvoří takovou mříž. Tato vzdálenost je počítána jako suma absolutní hodnota rozdílu dvou souřadnic. Pokud bychom měli znovu body A a B se souřadnicemi x a y. Obecný vzorec by vypadal takto: $m_m(A,B) = \sum_{i=1}^{n} |a_i - b_i|$. Pro výpočet této vzdálenosti je potřeba pouze absolutní hodnota a odčítání/sčítání. Proto pro velké množství dat, je tento způsob mnohem rychlejší. Na druhou stranu, je také tento způsob méně přesný. Na obrázku

číslo 2.1, lze vidět zelenou barvou nakreslenou euklidovskou vzdálenost o délce přibližně 8,49. Modrá, červená a žlutá jsou vše Manhattenská vzdálenosti a mají délku 12.



2.1 - Ilustrativní obrázek [9]

Jiným, již hotovým řešením se zabývají na následujícím odkaze [10]. Ve zkratce se jedná o vhodné násobení zeměpisné šířky a výšky a využití funkcí sinu, kosinu. Je zde návod, jak využít MySQL příkazy, pro zjištění vzdáleností od bodu v kilometrech. Jedná se o kruhovou vzdálenost od bodu na mapě. Pro nás by bodem na mapě byla poloha uživatele a v okruhu bychom hledali muzea či galerie. V tomto případě je pro získání údajů využit pouze jeden příkaz, který zredukuje počet dat přenášených z databáze. Vzdálenost počítají přímo v příkaze, tudíž eliminují velkou část dat, která by se přenášela zbytečně, pokud bychom vzdálenost počítali mimo databázi.

2.6 Překlad textů

Tento odstavec se bude zabývat možným strojovým překladem ještě nepřeložených textů, jednalo by se spíše o vylepšení pro server. V případě, že by si uživatel zažádal o překlad do jazyka, pro který ještě přeložený text nemáme, server by v tento moment poslal požadavek na překladač (možnost využít DeepL překladač, má vytvořené API rozhraní a zároveň hezké strojové překlady) o překlad daného textu, přeložený text by odeslal uživateli a zároveň by si text uložil do databáze jako překlad.

Využití DeepL je v rozsahu 500 000 znaků za měsíc. Tento limit je pro použití zdarma. Licence DeepL nijak nebrání využití v naší aplikaci. Přidal by se například flag „strojový překlad“, aby bylo jasné, že je stále potřeba udělat překlad, ale již nějaký překlad existuje. Využitím tohoto „on-demand“ přístupu by server nebyl nucený často využívat služby překladače a zároveň měl i překlad pro každého uživatele. Vzhledem k tomuto přístupu by mělo 500 000 znaků stačit. Implementace závisí hlavně na programátorovi serveru, tudíž on rozhodne, zda se využije mnou navržený způsob.

2.7 Programovací jazyk

V zadání je, že výsledná aplikace by měla fungovat na platformě Android. Častými volbami pro Android aplikace jsou: Java, Kotlin, Flutter nebo C#.

Java byla v roce 2021 podle statistik nejrozšířenějším programovacím jazykem pro vývoj mobilních Android aplikací. Postupně je ale nahrazována jazykem Kotlin. Syntaxe Kotlinu může na první pohled být nepřehledná, ale zápis je mnohem stručnější než v Javě.

Další možností je novější Flutter, jedná se o open-source SDK od společnosti Google, podporuje multiplatformní programování, to znamená, že jeden kód může fungovat jak na Android, tak na iOS. Jedná se o framework, pro programovací jazyk Dart.

Poslední z vybraných jazyků je C#. Svojí strukturou a syntaxí je podobný Javě, proto se také využívá pro vývoj mobilních aplikací.

3 Aplikace mobilních průvodců

V rámci této kapitoly, jsem vybral 3 aplikace, které poskytují podobné funkcionality, jako mnou vyvíjená aplikace. Aplikace, které jsem vybral, a níže popsal jsou dle mého nejvíce relevantní pro náš projekt a jejich účelem nejbližší. U každé aplikace popíši základní informace z App store nebo Google play a poté své vlastní zkušenosti z používání. Hodnocení aplikací je využito z App store.

3.1 WikiCompass

Aplikace je vytvořena pouze pro iOS. Podporuje rozšířenou realitu a funkce GPS. Hodnocení 4,8/5 (04.2022). Poslední aktualizace 10.10.2020.

Díky této aplikaci můžeme vyhledávat a zobrazovat články z Wikipedie v reálném čase. Aplikace díky konkrétní poloze uživatele dokáže nabízet informace o zajímavých místech v okolí. Je zde možnost využití rozšířené reality, které rozpozná důležitá místa, na které zrovna koukáte.

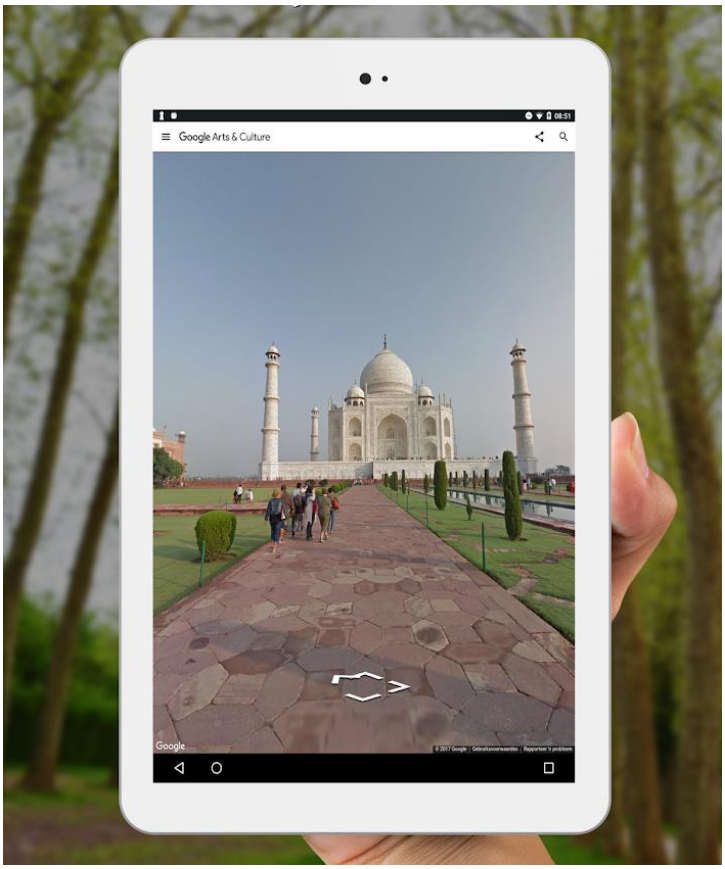
Aplikace má jednoduchý design a ovládání bylo velice intuitivní. Vzhled aplikace se podobá moderním webovým aplikacím a po spuštění se zobrazí mapa, s možnými zajímavostmi, takzvanými „points of interest“ – POI. Lze vidět i na obrázku 3.1. Ke každému takovému místu existuje i článek právě na Wikipedii a pouhým kliknutím na obrázek se článek zobrazí. Aplikace má i druhou možnost používání, a to je pomocí rozšířené reality. Díky této funkcionalitě se návrh článků zobrazí pouze na objekty, na které se v daný moment díváme. Stejným způsobem lze daný článek rozkliknout a zobrazit ho tak v celé podobě. Je zde i možnost vyhledávání článků pomocí volby města, například Plzeň a poté se zobrazí mapa Plzně a zajímavá místa s články.

3.1 - Screenshost z aplikace WikiCompass [13]

3.2 Google Arts & Culture

Aplikace, jak již z názvu vyplívá, je vytvářena pod záštitou Googlu. Funguje na iOS i Android. Umožňuje využívat široké množství funkcí, jako například: 360° videa, rozpoznávač děl, prohlídky ve virtuální realitě, kapesní galerii a mnoho dalšího. Hodnocení 4.6/5 (04.2022). Aktualizováno 14.4.2022.

Tato aplikace od Google je protkaná funkcionalitami, tudíž první orientace v aplikaci není úplně jednoduchá. Ale během krátké chvíle se uživatel zorientuje. Aplikace umožňuje vyhledání kulturních institucí ve vašem městě či blízkém okruhu. Při vyhledání aplikace se zobrazí její informační karta se základními informacemi. Většina funkcionalit aplikace je blíže zaměřená na umělecká díla či historické momenty/osoby. Umožňuje prohlédnutí exponátu přímo ve vašem domově, přetvořit váš obraz v jiný, vytvořený v nějakém uměleckém směru. Díky Google Street view, je zde možnost projít se po více než 2000 muzeím a výstavách po celém světe. Na obrázku 3.2, lze vidět street view v okolí indického Taj Mahal. Při přiblížení k známějšímu exponátu Google nabízí detailní zobrazení. Speciálním a zajímavým zobrazením je časová osa, kde se zobrazují obrazy, historické momenty či jiná kulturní díla.

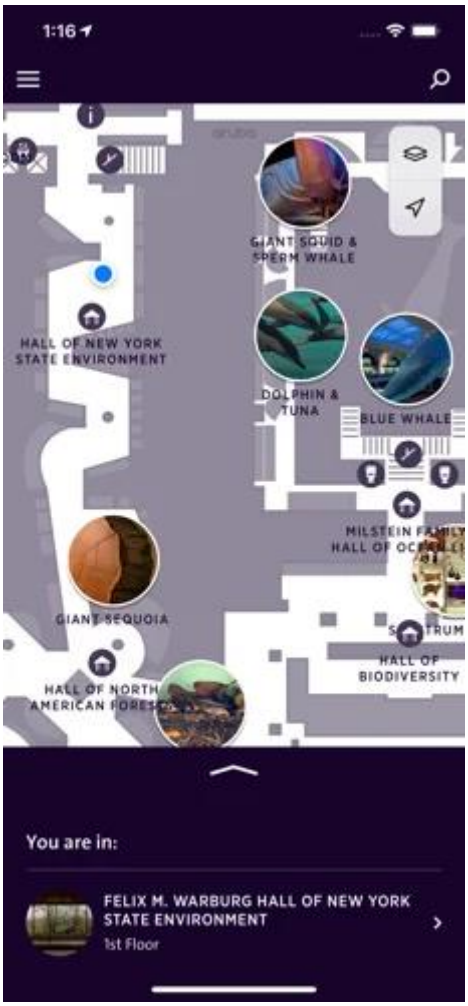


3.2 Screenshot aplikace Google Arts & Culture [14]

3.3 Explorer – AMNH NYC

Aplikace je vytvořena pro iOS i pro Android. Je dělaná speciálně na míru Muzeu přírodní historie v New Yorku. Hodnocení 3.9/5 (04.2022). Aktualizováno 28.2.2022.

Aplikaci jsem měl tu možnost vyzkoušet v reálném prostředí, když jsem byl před třemi lety v New Yorku. Aplikace funguje jako skvělý průvodce pro muzeu, doplňuje informace o aktuálních expozicích, možnost výběru pro vás zajímavého obsahu a přímo nákupu vstupenky do muzea. Díky přesnému určování polohy díky Bluetooth beacons, lze přesně navigovat uživatele, jako klasická navigace například pro automobily. Nabízí taky výběr nejbližších důležitých míst, jako třeba „jídlo“, „toalety“, „obchod se suvenýry“ nebo „východ“. Mapa muzea je doplněna obrázky z expozic, tím pádem se uživatel lépe zorientuje v tom, co by chtěl vidět (viz obr. 3.3).



3.3 Screenshot aplikace Explorer – AMNH NYC [15]

4 Komunitní překlad

Díky stále zvětšující se síti internetových uživatelů, dochází čím dál více ke spojení lidí se stejnými zájmy a koníčky. Internet ale omezil potřebu setkávání se lidí v reálném životě a tímto lidé společně komunikují navzdory tomu, že se od sebe mohou nacházet tisíce kilometrů daleko. Toto umožnilo rozšíření komunitního překladu na novou úroveň. Tím, že jsou lidé většinou spojeni nějakými zájmy či koníčky, komunitní překlad vzniká bet finančních nákladů pomocí dobrovolné práce komunitních členů.

Důležitými aspekty pro překlad textů jsou: kvalita, rychlost a cena. Jak se s těmito aspekty dokáže porovnat komunitní překlad oproti najmutí profesionálních překladačů?

Kvalita: Komunitní překlad se kvalitativně nemůže rovnat překladu profesionálnímu, ale většinou se jedná o užší spektrum překladu, a v tomto ohledu mohou i laici poskytnou kvalitní překlad. Právě díky jejich zálibě, či koníčku

mohou mít v tomto spektru dobré znalosti pro překlad. Vždy může dojít k nepřesnostem v překladu, díky absenci profesionálních překladatelů.

Rychlost: Jedním ze znaků komunitního překladu je, že ve většině probíhá dobrovolně, a proto z pravidla nemá žádný předem stanovený datum dokončení. To může a nemusí být problém.

Cena: V tomto aspektu je komunitní překlad ve velké výhodě. Z pravidla překlad probíhá bez vyplácení odměny a tím se může ušetřit po finanční stránce. Jelikož se jedná o univerzitní projekt, je cenové kritérium jedno z hlavních.

Dalším reálným problémem může být úmyslné překládání s chybami, či přepracování již přeložených textů a jiné podobné nechtěné akce. Je proto důležité vytvořit mechanismy, které tento jev dokážou vyřešit a tím zabránit chybnému překládání. Tyto řešení mohou částečně vést i ke zlepšení celkové kvality překladů. Jedním z možných přístupů je komunitní hlasování. V tomto případě je umožněno celé komunitě hlasovat o kvalitě překladů a tím vybrat nejlepší překlad. Dalším možným přístupem je zavedení uživatelských rolí, ověření překladatelé budou mít právo přijímat překlady, zatímco neověření jenom přeložit text například.

4.1 Volba řešení pro zprávu překladů

Tímto problémem se již zabývala práce, na kterou tato bakalářská práce navazuje, *Analýza existujících mobilních průvodců po muzeích a* památkách [12]. V páté kapitole popisuje čtyři způsoby pro kontrolu překladů. Jimiž jsou:

- Komunitní překlad s validací od jednoho správce nebo skupiny správců
- Wiki přístup
- Komunitní překlad s validací od skupiny uživatelů
- Kategorizace uživatelů na překladatele a normální uživatele

Podle reakcí institucí, které projevili zájem o tento projekt vyplývá, že preferovaným způsobem by byl způsob třetí a čtvrtý, tudíž *Komunitní překlad s validací od skupiny uživatelů* anebo *Kategorizace uživatelů na překladatele a normální uživatele*.

4.1.1 Komunitní překlad s validací od skupiny uživatelů

Toto řešení se zakládá na aktivním přístupu komunitních překladatelů. Uživatelé by byli rozděleni na skupiny překladatelů a moderátorů překladů. Moderátoři by se starali o volbu nejlepšího překladu a mohli by svými hlasy volit ten nejlepší. Zároveň by byli i překladatelé. Díky komunitnímu hlasování by mohl být vybrán nejlepší překlad a zároveň by se zamezilo vandalismu na překladech. [12]

4.1.2 Kategorizace uživatelů na překladatele a normální uživatele

Jedná se o modifikaci předchozího modelu, s tím rozdílem, že by nemohli překládat všichni uživatelé, ale pouze ověření uživatelé. Uživatel se bude moci stát překladatelem poté, co projde procesem ověření. Překladatelé nadále budou komunitně hlasovat pro překlady. Kvalita tohoto modelu znovu závisí na velikosti a aktivitě překladatelů. [12]

4.2.3 Shrnutí

Řešení v našem projektu volil autor serveru a webové aplikace. Vybral první řešení s možným hlasováním o kvalitě překladu. Každá instituce má jednoho či skupinu správců, kteří se starají o překlady, které budou zobrazovány. Dále je implementovaný systém pro hodnocení překladů. Tímto způsobem správci institucí budou mít zpětnou vazbu ke kvalitě překladů.

5 Návrh aplikace

Aplikace byla tvořena za účelem co nejjednoduššího ovládání, zároveň aby uživatel měl co nejkratší cestu k získání překladu k exponátu. Jako interaktivní prvek pro aplikaci byl zvolený QR kód, který byl zmiňován již dříve. Ten umožní uživateli rychlé získání překladu k exponátu. Uživateli bude umožněno i ruční volba exponátu, pokud instituce nebude chtít využívat možnost QR kódů.

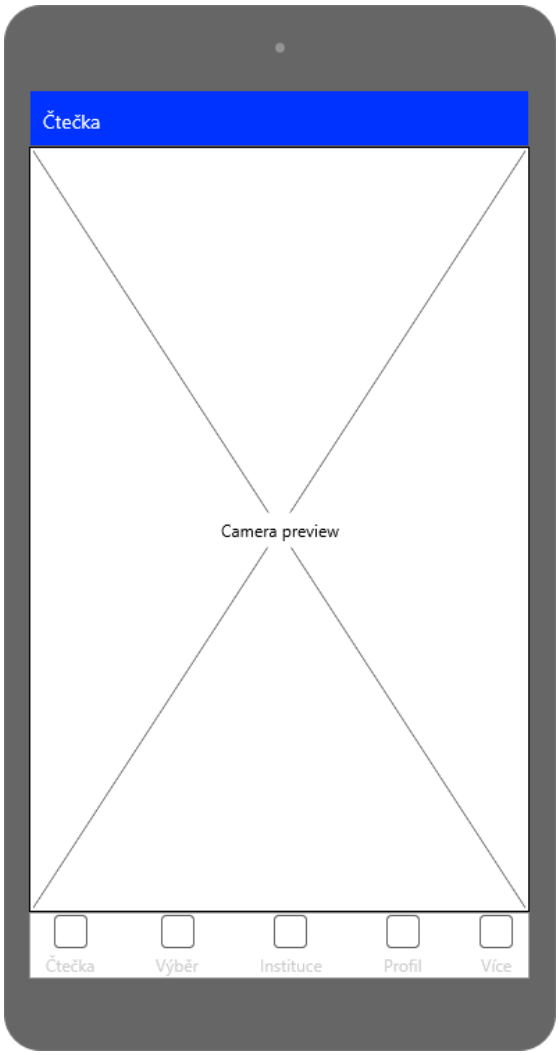
5.1 Návrh obrazovek aplikace

Návrh aplikace je založen na doporučení z předchozí bakalářské práce, na kterou navazuji. [12]. V kapitole *8.4.1* je vyobrazena informační architektura mobilní aplikace. Tento graf jsem využil pro návrh této aplikace, ale usoudil jsem, že nebude v mých silách implementovat celou navrženou architekturu. Pro implementaci jsem vybral ty nejdůležitější prvky, a jedná se o přibližně 90 % z celkové návrhu.

Pro důležité či zajímavé části aplikace jsem vytvořil takzvané wireframy. Wireframe je náhled obrazovky s definicí funkce a obsahu. Pro popis v této kapitole jsem nevytvářel náhled obrazovky s registračním, přihlašovacím formulářem nebo s formulářem pro změnu hesla. Tyto obrazovky mají ve většině aplikací podobný vzhled, a proto není nutné je explicitně zobrazovat.

Celá aplikace je postavená na navigační listě ve spodní části obrazovky. Využil jsem možnosti ikony a krátkého textu, tak aby navigace byla jednoduchá a intuitivní. Spodní navigační lišta dovede uživatele na nejdůležitější obrazovky. V horní části obrazovky je lišta s názvem aktuální obrazovky.

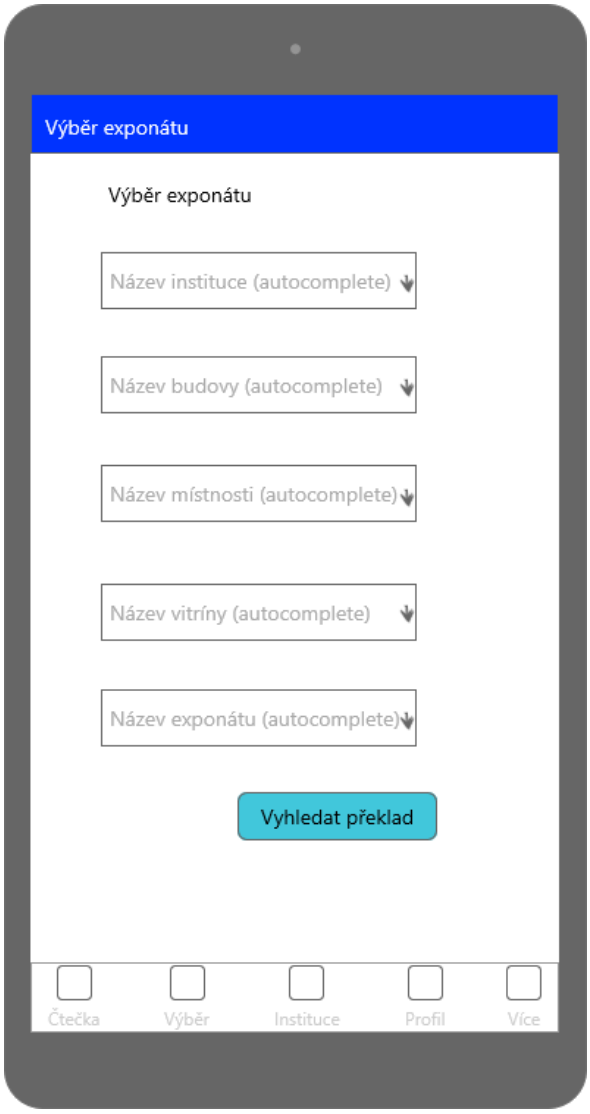
5.1.1 Obrazovka čtečky



5.1 Obrazovka čtečky

Jedná se o úvodní obrazovku při načtení celé aplikace. Důvodem bylo, aby uživatel měl co nejjednodušší a nejrychlejší přístup k získání textu. Pokud uživatel povolí aplikaci přístup k fotoaparátu, většina okna bude obsahovat právě aktuální záběry z fotoaparátu mobilního zařízení (viz obr. 5.1). Uprostřed obrazovky bude ještě menší čtverec, kvůli přehlednosti není vyobrazen, který bude uživateli indikovat, kam zaměřit QR kód, kterým se bude exponát identifikovat. Ve spodní obrazovce je již vidět výše zmíněná navigační lišta. Každá navigační položka bude mít svoji ikonu, která bude co nejlépe vystihovat obrazovku, na kterou odkazuje.

5.1.2 Obrazovka výběru exponátu



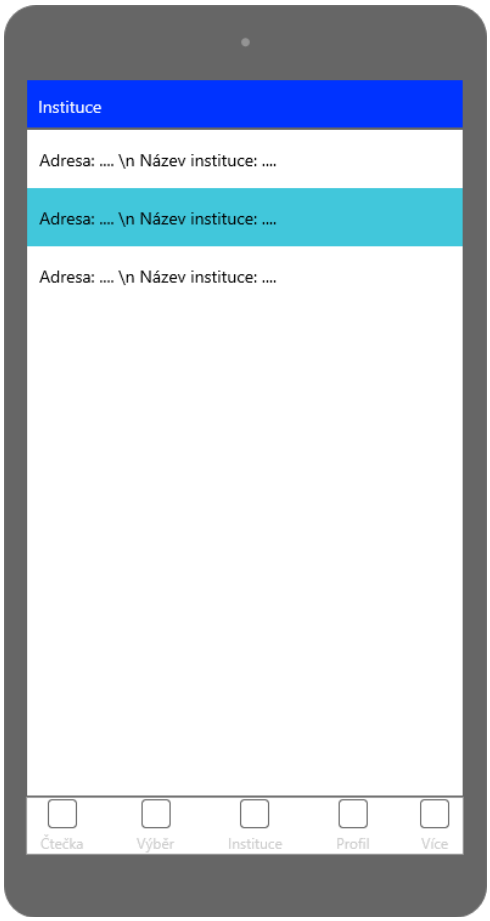
5.2 Obrazovka pro výběr exponátu

Tato obrazovky je pro uživatele druhou možností, jak získat překlad k exponátu. Formulář bude obsahovat 5 podobných polí s jedním tlačítkem pro odeslání formuláře a získání překladu (viz obr. 5.2). Po rozkliknutí pole se uživateli zobrazí nápověda se všemi možnostmi. Tím, jak uživatel bude zadávat text, možnosti se budou filtrovat.

Po vyplnění instituce se podle exponátů naplní další pole. „Název vitríny“ je jediné pole, které nemusí být vyplněno z důvodu, že v muzeu být nemusí. Ostatní pole musí být vyplněné pro zobrazení překladu.

Při správném vyplnění formuláře se zobrazí překlad v jazyce, který si uživatel zvolil.

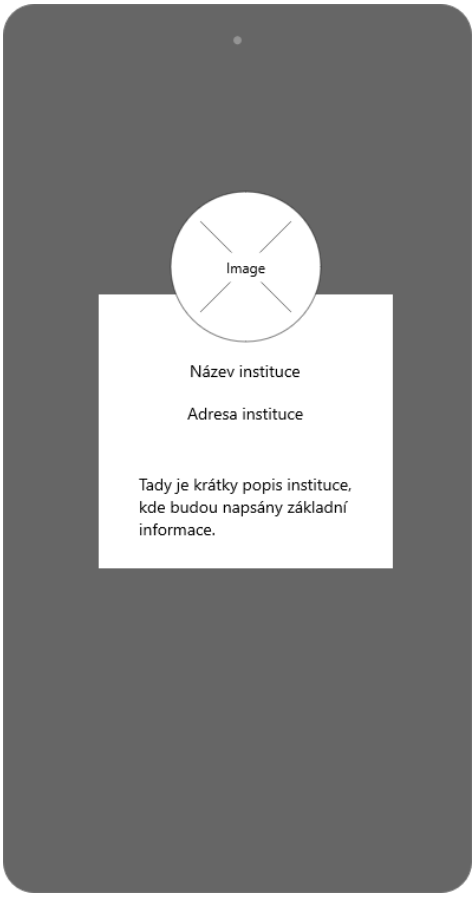
5.1.3 Obrazovka seznam institucí



5.3 Obrazovka seznamu institucí

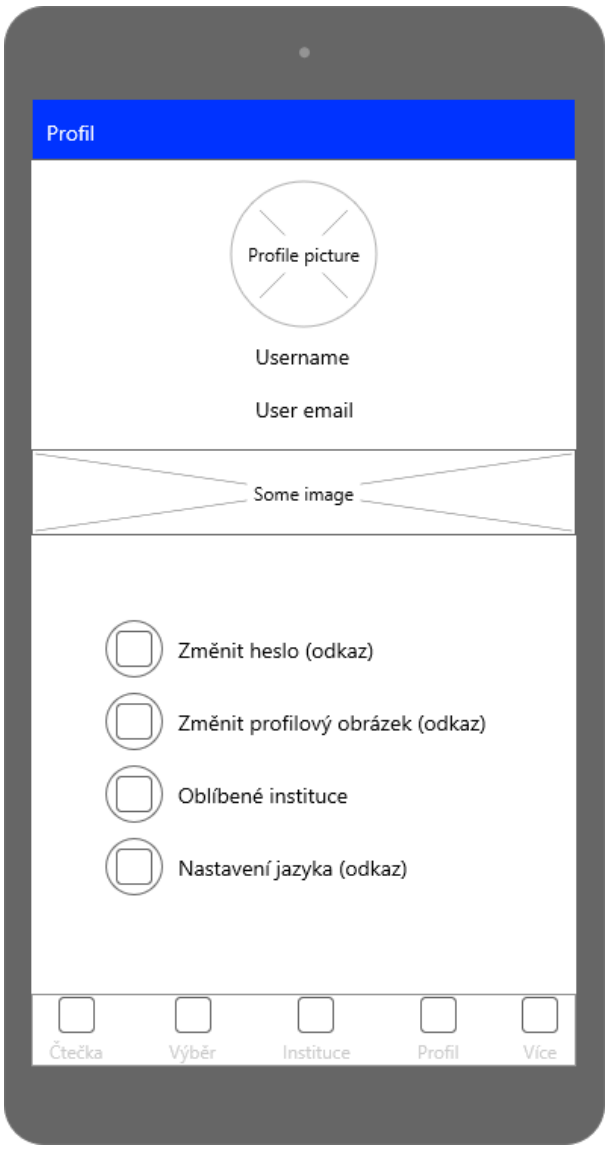
Na této obrazovce se bude nacházet seznam všech institucí, které jsou aktuálně k dispozici. V seznamu bude ke každé instituci uvedena její adresa a název instituce. Toto lze vidět na obrázku 5.3. Na položku v seznamu bude uživatel moci kliknout a tím si zobrazí kartu s institucí. V této kartě se nachází obrázek instituce, název, adresa a krátký popis. (viz obr. 5.4)

Tato karta slouží pouze informativně a nebude možnost žádné větší interakce krom zobrazení karty o instituci. Karta samozřejmě kliknutím půjde zavřít a bude možnost zvolit jinou instituci.



5.4 Karta instituce

5.1.4 Obrazovka profilu

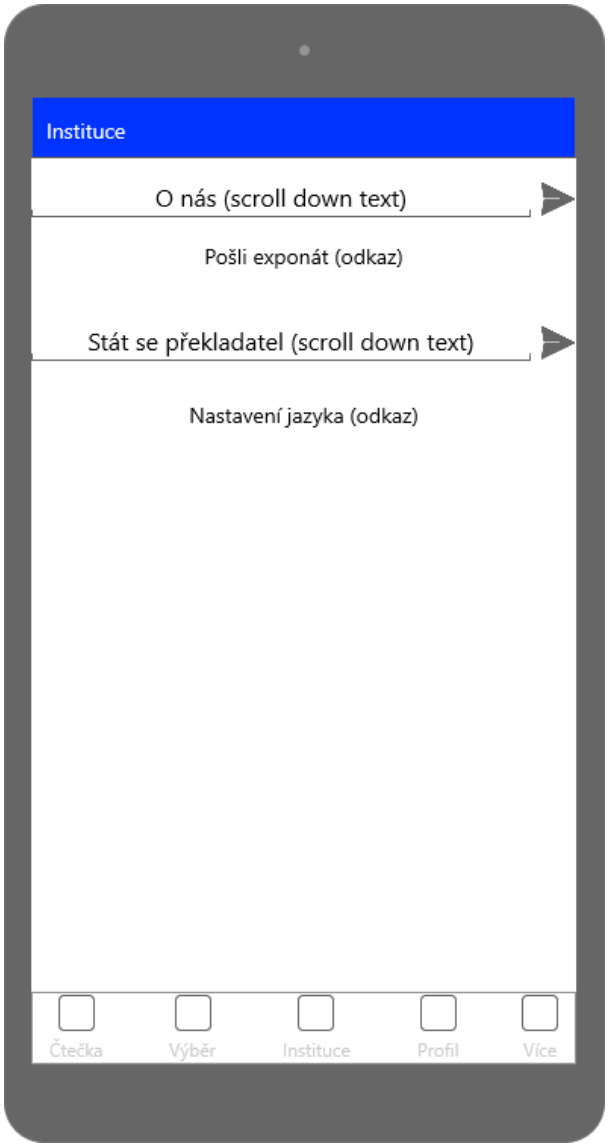


5.5 Profilová obrazovka

Obrazovka se uživateli zobrazí po úspěšném přihlášení do aplikace. V horní části uživatel najde základní informace o svém profilu. Profilový obrázek, jeho jméno a email. Pro oddělení uživatelových dat od zbylé části obrazovky, se využije nějaký neutrální obrázek.

V druhé části se budou nacházet uživatelské odkazy. Formulář pro změnu hesla, dialogové okno pro změnu profilového obrázku, seznam s oblíbenými institucemi a možnost změny jazyka. Náhled obrazovky je na obrázku 5.5. Odkazy budou mít i vlastní ikonku, která by uživateli měla ulehčit orientaci v aplikaci a tím co nejvíce zpříjemnili používání aplikace.

5.1.5 Obrazovka více



5.6 Obrazovka více

Na této obrazovce (viz obr. 5.6) se uživatel může dozvědět všechny užitečné informace, či využít funkce, ke kterým má i nepřihlášený uživatel právo. Uživatel si může přečíst dva texty, **O nás** a **Stát se překladatelem**. Při načtení obrazovky, oba texty jsou schované, uživatel texty nevidí. Při kliknutí na text, či šipku na kraji obrazovky se daný text rozbalí směrem dolů, dalším stisknutím uživatel daný text zase sroluje a schová.

Další dva texty jsou odkazy, na funkce, které uživatel může využívat i jako nepřihlášený uživatel. Je to formulář pro přidání nového exponátu a také možnost nastavení jazyka.

5.2 Uživatelské role

V této podkapitole popíšu všechny role, které jsou využity v celém systému. Tudíž na webové i mobilní aplikaci. Role se v tomto ohledu prolínají, a kvůli tomu, že obě aplikace jsou propojeny, je důležité zmínit všechny role. Uživatelský účet vytvořený v mobilní aplikaci bude fungovat i ve webové a obráceně. Při prvním spuštění aplikace uživatel ještě nemá svůj účet a tím vzniká role **nepřihlášený uživatel**. Tento uživatel se může kdykoli registrovat a poté přihlásit. Tím vzniká role **přihlášený uživatel**. Tímto jsou role pro mobilní aplikaci vyčerpány. Ve webové aplikaci se ale role **přihlášený uživatel**, může rozdělit na **překladatel** a **správce instituce**. Poslední rolí, která je tak trochu spojením ostatních je

návštěvník instituce. Tato role představuje uživatele, pro které byl celý systém vytvářen.

5.2.1 Nepřihlášený uživatel

Nepřihlášený uživatel ve webové aplikaci má velmi omezené pravomoci. Má možnost přečíst si úvodní stránku, a tím zjistit základní informace o celém projektu nebo může využít registrační či přihlašovací formulář. Po registraci a následném přihlášení se uživateli zpřístupní většina webové aplikace.

V případě mobilní aplikace, nepřihlášený uživatel má o mnoho více pravomocí. Uživateli je umožněno zvolit si jazyk, ve kterém chce získávat překlady z našeho systému. Může také získat schválené překlady k exponátům. Zobrazí si seznam všech institucí v systému a případně, pokud najde exponát, který není v systému, má možnost ho do systému přidat.

5.2.2 Přihlášený uživatel

V mobilní aplikaci má přihlášený uživatel stejné pravomoci jako uživatel nepřihlášený, rozšířené o uživatelský profil. Na uživatelské obrazovce má uživatel možnost využít změny hesla, která se projeví i na účtu ve webové aplikaci. Změna profilového obrázku, který se zobrazuje na profilové obrazovce (viz obr. 5.4). Má možnost i změny jazyku, tato funkcionalita je umožněna pro přihlášené i nepřihlášené uživatele.

Ve webové aplikaci se role přihlášeného uživatele dá rozdělit dále do dvou dalších rolí a to: **překladatel** a **správce instituce**. Přihlášený uživatel má možnost být správcem instituce, pokud nějakou vytvoří, či již aktuální správce instituce ho přizval, aby byl také správcem. Díky systému komunitního překladu, přihlášený uživatel se může stát bez jakéhokoli schválení překladatelem a vytvářet překlady. Může také již hotové překlady hodnotit a tím pomoci s výběrem toho nejlepšího.

5.2.3 Překladatel

V případě, že se přihlášený uživatel rozhodne přispívat do komunity svými překlady, tak zprvu si musí vybrat instituci, pro kterou bude aktuálně chtít překládat. Po rozkliknutí instituce se mu zobrazí všechny exponátu, které jsou k instituci přiřazeny. Uživatel má poté možnost zvolit si jazyk, do kterého by chtěl popisek exponátu přeložit. Pokud se mu již vytvořený překlad líbí natolik, že uzná, že není potřeba ho předělávat, může mu dát palec nahoru, a tím ukázat správci instituce, že tento překlad je dobrý.

Překladatel má možnost zobrazení si vlastních již vytvořených překladů. Může vytvářet jejich nové verze a případně je i smazat.

5.2.4 Správce instituce

Správcem instituce se uživatel může stát dvěma způsoby. Jedinou podmínkou je mít uživatelský účet. První způsob, jak se stát správcem instituce je, že si uživatel vytvoří vlastní instituci, kterou dále může spravovat. Druhou možností je, že uživatel bude přizván již existujícím správcem instituce. Tímto získá stejná práva jako správce, který ho přizval.

Správce instituce má poté možnost přidávat jazyky k instituci, pro které bude možnost vytvářet překlady. Dále může také přidávat nové exponáty, a případně editovat data o exponátech, pokud by některé informace byli špatné. V této sekci

mají správci také přístup ke QR kódům, které mohou rozmístit po instituci. Jako správce taky může volit zobrazení překladů. Má možnost zvolit jazyk u exponátu, a ze všech existujících exponátů může vybrat ten, který se bude uživatelům zobrazovat. Pokud si správce není jistý, může využít hodnocení (palce nahoru) komunitou a podle toho vybrat. Aplikace dovoluje taky správci aplikace smazat celou instituci i se všemi informacemi s ní spojenou.

5.2.5 Návštěvník instituce

Tato role je speciální a částečně je popsána již výše. Návštěvník instituce se vůbec nedostane na webovou aplikaci, prakticky vzato ani nemusí vědět, že nějaká webová aplikace existuje. Jeho cílem je využít mobilní aplikaci pro co nejlepší zážitek z návštěvy instituce. V mobilní aplikaci návštěvník instituce může zastávat roli přihlášeného i nepřihlášeného uživatele. Má možnost získávat schválené překlady k exponátům, zobrazit si aktuální instituce v systému či přidat nový exponát, pokud se ještě nenachází v databázi.

5.2.6 Shrnutí

Pro přehlednost a shrnutí uživatelských rolí jsou nejdůležitější práva a akce k rolím shrnutá v tabulce 5.6.

Nepřihlášený uživatel	Přihlášený uživatel	Překladatel	Správce instituce	Návštěvník instituce
Registrace	Překládat texty	Překládat	Přidat nové správce	Zobrazit překlady
Přihlášení	Hodnotit překlady	Spravovat svoje překlady	Editovat přidané exponáty	Zobrazit seznam institucí
	Vytvořit instituci	Hodnotit překlady	Schválit překlady	Přidat exponát
	Přidávat exponáty k instituci		Smazat instituci	

5.6 Tabulka s uživatelskými rolemi

5.3 Zprostředkování překladu

Pro získání oficiálního překladu k exponátu, je potřeba získat ID exponátu a k němu přidat zkratku jazyka, ve kterém bychom chtěli překlad. Server vyhodnotí, jestli k danému exponátu existuje schválený přeložený text ve vybraném jazyce. Pokud ano, odešle překlad v nastylovaném HTML textu a HTTP kód 200 (OK). V případě, že daný text neexistuje, server odešle varovnou hlášku a HTTP kód 400 (Bad Request). V momentě, kdy přijde HTTP kód 400, zkusím na server poslat stejný dotaz, ale změním jazyk na angličtinu. To z důvodu, že angličtina je v dnešní době nejrozšířenější jazyk a je velká šance, že exponát bude do tohoto jazyka přeložen a zároveň návštěvník bude překladu rozumět. V případě, že i v tento moment nepřijde oficiální překlad ale chyba, tak uživateli vyskočí okno, s oznámením, že neexistuje

překlad v jeho jazyce ani angličtině. Poté je už na uživateli, zdali zkusí jiný jazyk, nebo se obejde bez překladu.

Jiný způsob, kterým by mohla probíhat komunikace mezi serverem a klientem v aktuální verzi není implementována a musel by se na ni adaptovat jak server, tak mobilní aplikace. Z mého pohledu by se pro uživatele zpříjemnilo získávání překladu a tím zlepšila jeho zkušenost s aplikací. Tento způsob by na první pohled fungoval stejně, pro získání překladu by bylo potřeba ID exponátu a zkratka návštěvníkova jazyka. Pokud by daný překlad existoval, vše by proběhlo stejně, uživatel by text získal a byl by mu zobrazen. Ovšem, pokud by text neexistoval, server by odeslal seznam všech jazyků, ve kterých daný překlad existuje. Tento seznam by poté mobilní aplikace zobrazila uživateli a ten by se mohl vybrat, který jazyk, mu vyhovuje nejvíce. Pokud by server odeslal zároveň i překlady, ušetřila by se komunikace mezi klientem a server.

6 Popis implementace

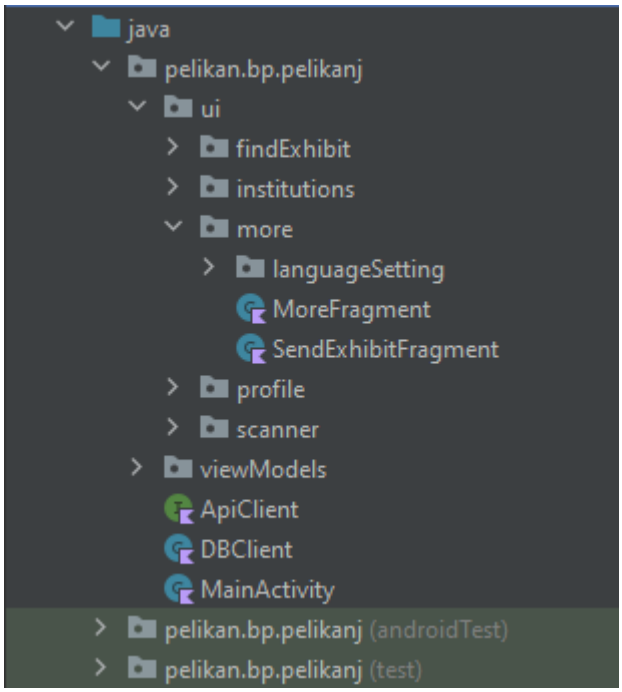
Hlavním cílem aplikace je umožnit uživateli získat překlady k exponátům a tím se nechat provést po muzeu. Pro implementaci mobilní aplikace byl zvolen programovací jazyk Kotlin. Důvodem byl i předmět KIV/MBKZ, který mě naučil základy jazyka Kotlin. Aplikace je programovaná pro API level 23+ neboli Android 6 a více. Tato volba proběhla na základě tabulky [11], tak aby aplikace fungovala na co nejvíce zařízení a zároveň měla přístup k moderním funkcionalitám. Pro vytvoření spustitelného .apk souboru jsem využil nástroj **Gradle**, který se stará i správné sestavení celého projektu a spravuje všechny závislosti na jednom místě. Tímto způsobem byly přidány například knihovny pro čtení QR kódů, či knihovna **Retrofit**, která ulehčuje komunikaci mezi klientem a serverem. Aplikaci jsem se snažil programovat podle výše zmíněných návrhů. Kladl jsem důraz na přehlednost kódů.

6.1 Popis a struktura aplikace

Tato sekce se podrobněji zaměří na zdrojový kód, jeho strukturu do souborů a složek a pomocné soubory a závislosti. Všechny popisované balíky se nachází v defaultním balíku **app/src/main**. Toto se dále rozděluje na složku **java/** se zdrojovými kódy (Kotlin) a složku s externími zdroji aplikace **res/**. V této složce, respektive podsložkách se nacházejí obrázky, ikony, soubory XML s rozložením obrazovek, animace, předdefinované řetězce.

Mimo tyto dvě hlavní složky se nachází ještě nejdůležitější soubor **AndroidManifest.xml**. Jedná se o vstupní bod aplikace, a obsahuje informace o naší aplikaci. Tyto informace se předávají systému ještě před spuštěním aplikace. Soubor obsahuje nastavení celé aplikace. Žádá o právo k užití fotoaparátu, připojení k internetu a zápis a čtení z externího úložiště. Dále obsahuje název aplikace *Průvodce institucemi*, její ikonu, která je dočasně zvolená jako ikona budovy.

6.1.1 Složka java a její podsložky



6.1 Struktura složky java

V této části práce jsou popsány jednotlivé složky ze struktury zdrojových kódů (viz obr. 6.1). Ve složce **java/pelikan.bp.pelikanj/ui** se nachází složky se zdrojovými soubory, které jsem se snažil rozdělit do skupin tak, aby jejich činnost odpovídala obrazovkám aplikace. Tudíž ve složce **ui/more** se nachází zdrojové kódy, které se využívají na obrazovce *Více* a na obrazovkách, na které se uživatel může přesměrovat. Dále složka **viewModels** obsahuje pouze datové třídy, datová třída je jednoduchá třída, která slouží k uchování dat, může obsahovat i standartní funkce. Zpravidla jsou to funkce jako getter/setter nebo toString. Jsou zde ještě dvě třídy a jedno rozhraní, které nezapadá do žádné z výše popsaných balíčků: **ApiClient.kt**, **DBClient.kt** a **MainActivity.kt**.

Soubor MainActivity.kt

Je to vstupní bod aplikace. Při startu aplikace se nejdříve kontrolují data z lokální databáze, kontroluje se, zda už existuje záznam o uživateli. Pokud ano, zkontroluje se uživatelský token (pokud uživatel je přihlášený), načtou se data z uloženého tokenu, pokud nevypršela expirace a token se obnoví. Pokud uživatel ještě nemá žádný záznam v databázi, záznam se vytvoří, s defaultním jazykem „cs“. Kontrolují se také práva aplikace, pro přístup k fotoaparátu a externímu úložišti. Pro lepší fungování aplikace se ve druhém vlákně požádá server o seznam institucí, které se uloží do lokální databáze.

Soubor DBClient.kt

Tato třída představuje kontroler pro práci s lokální databází. Dědí od třídy *SQLiteOpenHelper* a zároveň vytváří její instanci. Tímto způsobem musí třídy implementovat a přepsat dvě metody *onCreate* a *onUpgrade*. Metoda *onCreate*, říká, co se má dít, při vytvoření databáze, metoda *onUpgrade* poté definuje, co se má vykonat, pokud se verze databáze změní. V naší aplikaci při nové verzi databáze se všechny tabulky smažou a poté se zavolá funkce *onCreate*, která tyto tabulky znovu vytvoří.

Další funkce jsou pro zprávu uživatelovo dat. V této tabulce jsou uloženy tři hodnoty: jazyk, token a profilový obrázek. Všechny tři hodnoty jsou reprezentovány typem *string?*, to znamená, že se bude jednat o řetězec ale je povolena hodnota **null**. Existují tedy funkce pro zapsání jedné ze tří hodnot,

vytvoření nového záznamu při prvním spuštění aplikace a potom funkce pro přečtení dat.

Dalším balíčkem funkcí jsou ty, které umožňují ukládání a čtení částí institucí. Ukládají a čtou instituce, budovy, místnosti a vitríny. Tyto funkce se využívají pro ušetření komunikace mezi serverem a aplikací, jsou to data, která se nebudou měnit v době vteřin či minut, proto je možnost uživateli stáhnout tyto data a další stáhnout při dalším startu aplikace například.

Soubor ApiClient.kt

Tento soubor představuje rozhraní pro používání knihovny **Retrofit** a jejích funkcí [17]. Retrofit je REST klient pro Javu a Android, umožňuje načítat a odesílat data ve formátu JSON, ale i v jiném formátu. Zajišťuje velice jednoduše komunikaci mezi serverem a naší aplikací. Kotlin umožňuje vytvořit *companion object* pro rozhraní, tímto můžu využít rozhraní a zároveň využít návrhový vzor *jedináček*. Tímto mám definici navázání spojení mezi serverem a aplikací na jednom místě.

```
@PUT("/users/updatePassword")
fun updatePassword(@Header("Authorization") token: String,
                   @Body password: PasswordModel
                   ): Call<ResponseBody>
```

6.2 Ukázka funkce pro Retrofit klienta

Na ukázce 6.2 bych chtěl popsat základní stavbu funkce, kterou využívá **Retrofit**. Tato funkce posílá na server požadavek pro změnu hesla uživatele. Jako první lze vidět anotace HTTP metody, v tomto případě je to *@PUT*. Tato anotace musí odpovídat té, kterou očekává server, nelze zaměnit *@GET* za *@POST*. V závorkách za anotací následuje cesta k danému koncovému bodu vystaveným serverem. Doména je v tomto případě vynechána, ta se zadává při vytváření spojení. Dále je klasická Kotlin syntaxe pro vytvoření funkce. Ze zápisu lze vyčíst, že funkce bude mít jméno *updatePassword* a bude mít dva parametry. Návratová hodnota funkce bude v tomto případě *ResponseBody*, to z důvodu, že neočekáváme žádná hodnotná data, pouze informaci, zda se akce zdařila, či ne. Nyní k parametrům funkce, ty jsou obohaceny o anotace *@Header* a *@Body*. Anotace *@Header* říká, že tento parametr má přidat při vykonávání od sekce Header a tím vytvořit dvojici **klíč-hodnota**, klíč je *„Authorization“* a hodnota je předaná v parametru. Druhá anotace je *@Body* ta říká, že tento parametr se má interpretovat jako tělo HTTP požadavku.

Takto sestavený požadavek se odešle na server, který ho zpracuje a pošle odpověď, ta bude mít typ *ResponseBody*. V tomto typu odpovědi se můžeme dočíst dvou zajímavých informací *ResponseBody*.**code** a *ResponseBody*.**body**. Proměnná *code* nám říká, jaký HTTP kód má odpověď ze serveru a *body* obsahuje tělo zprávy ze serveru.

V tomto rozhraní jsou takto definovány všechny funkce, které se potom volají z výkonných částí programu.

Složka viewModels

V této složce jsou všechny datové třídy aplikace. Datové třídy jsou určeny, aby uchovávali data, případně k těmto datům doplnili jednoduché funkcionality. Tyto

třídy se definují takto: *data class Translation(val proměnná: Typ, ...)*. Při volání třídy se automaticky volá konstruktor, který je naplněný daty. Deklarace může obsahovat i defaultní hodnoty, které se projeví při nevyplnění parametru. Tyto datové třídy v aplikaci využívám hlavně při komunikaci se serverem, pro návratové hodnoty ze serveru či správné namapování dat v těle požadavku pro server.

Složka ui/

V tomto balíčku se soubory dále dělí do menších částí, které jsou rozdělené podle toho, kde tyto kódy jsou vykonávány.

Složka ui/findExhibit

Obsahem této složky je obsluha pro formulář pro získání překladu k exponátu. Prvky formuláře pro uživatele nabízejí takzvaný *autocomplete*. To je ideální volba pro tuto funkcionalitu, protože vím, že uživatel bude nucen vybrat si nějakou možnost ze seznamu, a tento způsob mu podle jeho psaní filtruje pouze validní možnost. Tento způsob ušetří uživateli čas, tím, že nebude čekat, než napíše celý název, případně ho nenechá udělat chybu. Poté, co uživatel zvolí instituci, aplikace požádá server o exponáty dané instituce a díky tomu, se získá seznam budov, místností, vitrín a exponátů dané instituce. Podle volby budovy se uživateli zobrazí pouze validní místnost, obdobně s místností a vitrínou. Tímto způsobem se uživatel dostane přesně k exponátu.

Implementace proběhla pomocí dvou XML tagů a to <*TextInputLayout*> a vnořený <*AutoCompleteTextView*>. Do vnořeného tagu se pomocí funkce adapter doplní adapter s daty, které má zobrazovat. Tímto se celý formulář inicializuje.

Po stisku tlačítka, zažádá aplikace o překlad, k vybranému exponátu s jazykem, který má uživatel nastavený. Při úspěchu, zobrazí se překlad. V případě, že překlad neexistuje, zkusí se na server poslat žádost, kde se jazyk nahradí angličtinou. V případě úspěchu se text zobrazí, v případě neúspěchu se zobrazí uživateli hláška, že překlad neexistuje v jeho jazyce a angličtině. Po kliknutí mimo zobrazený překlad, formulář zůstane vyplněný až na exponát, tímto způsobem se uživateli ulehčí získání dalšího překladu.

Složka ui/more

V této složce můžeme najít balíček se soubory, které zajišťují volba uživatelského jazyka a poté formulář pro přidání nového exponátu. První zmíněný se nachází v **ui/more/languageSetting**. Realizace proběhla pomocí vlastně vytvořeného adapteru, čili seznamu jazyků k zobrazení. Na obrazovce se nachází pole pro vyhledávání v seznamu a podle zadaných hodnot se v seznamu filtrují jazyky.

Formulář pro nahrání exponátu funguje podobně jako ten, u výběru exponátu. S tím, že se formulář rozšířil o možnost přidat dvě fotografie. Pro uživatelské pohodlí jsem přidal možnost uživateli zvolit, jestli chce fotografii vyfotit nebo vybrat z galerie. Zároveň po vybrání fotografie se zobrazí ve formuláři náhled, aby si uživatel mohl zkontrolovat obsah fotografie. Při odeslání exponátu se uživateli zobrazí animace s tím, že se přidání exponátu povedlo nebo že přidání selhalo. Formulář se po odeslání vymaže a zůstane vyplněna pouze instituce, budova a místnost, aby byla zjednodušena možnost přidávat více exponátů naráz.

Složka ui/profile

V této složce se nachází komponenty pro přihlašovací a registrační formulář, uživatelský profil a formulář na změnu hesla. Formulář pro registraci po uživateli vyžaduje uživatelské jméno, email a heslo. Toto jsem údaje, který musí mít každý uživatel. Pro přihlášení se využívá uživatelské jméno a heslo. Formulář pro změnu hesla požaduje po uživateli pouze heslo a jeho zopakování pro zajištění správného hesla a také že se uživatel nepřepsal. Všechny tyto formuláře mají animaci, které informuje o tom, zda se akce zdařila či nikoli.

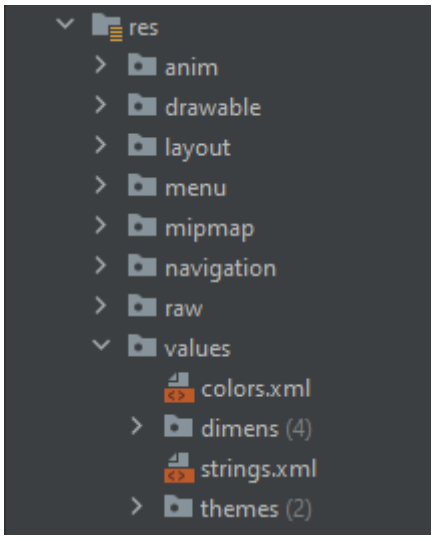
Obrazovka pro přihlášeného uživatele získává data z JWT tokenu. JWT neboli JSON Web Token, je druh tokenu, který umožňuje poslat data, v lidsky nečitelné podobě, ale nejedná se o šifrování. Tento token je uložený v lokální databázi po přihlášení a zároveň server ho odešle při úspěšném přihlášení. V tomto tokenu se nachází informace o uživateli, pro profilovou obrazovku je to uživatelské jméno a email. Uživatel má také možnost zvolit si profilový obrázek, tento obrázek se ukládá pouze u uživatele v telefonu. Server pro profilový obrázek není přizpůsoben.

Je zde také možnost pro *Oblíbené instituce*, bohužel tato funkcionalita z časových důvodů nebyla implementována.

Složka ui/scanner

V této složce je soubor, který se stará o čtečku QR kódů. Využil jsem knihovnu CodeScanner [5], která je již popsána v teoretické části. Aplikace je schopná přečíst i jiné kódy, ale z důvodu intuitivnosti se pro instituce generují zrovna QR kódy. Funkční logika je stejná jako u výběru exponátu. Pokud u načteného exponátu neexistuje překlad v jazyce uživatelem zvoleném, zkusí se vyhledat překlad v angličtině. Poté je buďto zobrazena hláška o tom, že překlad neexistuje anebo se zobrazí oficiální překlad.

6.1.2 Složka res a její podsložky



6.3 Struktura složky res

Obsahem této složky a jejích podsložek jsou všechny externí zdroje aplikace (viz obr. 6.3). Balíček se v adresářové struktuře nachází **app/src/main/res**. V dalších částech práce tyto balíčky více přiblížím. Zde bude jenom stručný popis obsahu balíčků. Balíček **anim** obsahuje definice animací, které jsou popsané *.xml* soubory. Ve složce **drawable** se nachází obrázky a ikony, které se v aplikaci využívají, mohou se zde nacházet i uživatel definované tvary. Jedna z nejdůležitějších složek v této části je **layout**, tato složka obsahuje *.xml* soubory s definicemi vzhledu uživatelských rozhraní po celé aplikaci. Ve složce **menu** se nachází definice menu, které je vyobrazeno jako navigace po aplikaci. Složka **mipmap** obsahuje generované ikony, podobně jako ve složce **drawable**. Složka **navigation** definuje možné přesuny po celé aplikaci. Složka **raw**

obsahuje soubory, které nelze zařadit jinam, jako cizí animace, nebo textové soubory jako například soubor s tabulkou jazyků a jejich zkratek. Balíček **values** obsahuje pro naši aplikaci dva hlavní soubory, a to je **strings.xml** a **colors.xml**.

Obsah složky res/anim

V tomto balíčku se nachází definice animací, které jsem vytvářel, respektive programoval já, na rozdíl od animací ve složce **raw**. Animace v tomto balíčku se využívají pro zobrazení karty s informacemi o instituci a při informování uživatele o úspěchu nebo neúspěchu jeho činnosti (registrace, nahrání exponátu). V těchto souborech se definují základní grafické hodnoty jako například: škálování, průhlednost a transformace. Díky nastavení těchto parametrů poté například obrázek instituce vypadá, že „doskáče“ na obrazovku.

Obsah složky res/drawable

Tento balíček obsahuje všechny ikony, obrázky a programově vytvořené tvary. Jedinou implementačně zajímavou věcí v tomto balíčku jsou programově tvořené tvary. V aplikace je jedná o jednoduché tvary, které jsou definované pomocí *.xml* souborů. Rodičovský prvek je vždy <*shape*> a potomci potom určují tvar daného prvku. Například se zde nastavují hodnoty jako <*corners*>, <*stroke*> nebo <*size*>. Tyto tvary jsou využity v aplikaci jako pozadí pro kartu instituce, velikost obrázku s institucí, nebo zvýraznění prázdného náhledu pro obrázek na obrazovce **Pošli exponát**.

Obsah složky res/layout

V této složce se nachází nejdůležitější soubory v aktuální kapitole. Jsou zde všechny soubory, které definují vzhled uživatelského rozhraní v celé aplikaci. Hlavním souborem je zde **aktivity_main.xml**. Tento soubor definuje hlavní aktivitu (hlavní okno) aplikace. Tento soubor obsahuje pouze definici navigačního panelu a definici fragmentu. Toto lze vidět na ukázce 6.4, ukázka je zkrácena o některé atributy, které nebyli podstatné, ukázka má hlavně ukázat strukturu *.xml* souborů.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >

<com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/nav_view"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:menu="@menu/bottom_nav_menu" />
```

6.4 Ukázka kódu z activity_main.xml

```
    <fragment
        android:id="@+id/nav_host_fragment_activity_main"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:defaultNavHost="true"
        app:layout_constraintBottom_toTopOf="@id/nav_view"
        app:navGraph="@navigation/mobile_navigation" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Na tomto kódu popíšu základní atributy těchto XML tagů. Obalovací tag *<androidx.constraintlayout.widget.ConstraintLayout>*, určuje, že všechny vnořené tagy, budou odpovídat Constraint layoutu. Každý vnořený tag by měl mít svoje *constrainty*. V ukázce jsem nechal constrainty pouze o jednoho tagu a to *BottomNavigationView*. Jak již napovídá jméno, například *constraintBottom_toBottomOf*, říká, že spodní část tohoto tagu bude zarovnaná na spodní část tagu svého rodiče. Úplně stejným způsobem fungují ostatní *constraint* atributy. Tag *<BottomNavigationView>* definuje spodní navigační lištu, důležitým atributem u *BottomNavigationView* je *app:menu="..."*. Tento atribut definuje, které položky se budou v navigační liště vyskytovat (viz **menu**). Druhým tagem je *<fragment>*, z jména lze vyvodit, že v tomto místě se budou nacházet fragmenty aplikace. Zmínit je potřeba atribut *app:navGraph*, díky jeho definici aplikace ví, jak se pohybovat po fragmentech a tudíž načítat správné obrazovky (viz **navigation**). Stejným způsobem jsou definované všechny soubory v této složce, pouze obsahují jiné tagy.

Obsah složky res/menu

Zde je pouze jeden soubor, a ten definuje obsah navigační lišty. Každá jedna položka v tomto souboru je definovaná stejně, a to XML tagem *<item>* a třemi atributy *id, icon, title*. Tímto každá položka bude mít svoji ikonu a název.

Obsah složky res/navigation

V tomto balíčku je jediný soubor, který definuje navigaci mezi fragmenty. V tomto souboru jsou definovány všechny fragmenty aplikace. V hlavní XML tagu *<navigation>* se nachází důležitý atribut *app:startDestination*, ten definuje úvodní obrazovku při startu aplikace. Dále už jsou jen vnořené *<fragment>* tagy (viz ukázka kódu 6.5). Tyto tagy mají svoje atributy, které určují jméno a název fragmentu, jeho definici uživatelského rozhraní obrazovky, pokud se z daného fragmentu má mít možnost někam navigovat, musí mít vnořený tag *<action>*, kde se definuje atribut id a cíl navigace (fragment).

```
<fragment
    android:id="@+id/navigation_profile"
    android:name="pelikan.bp.pelikanj.ui.profile.ProfileFragment"
    android:label="@string/title_profile_fragment"
    tools:layout="@layout/fragment_profile" >
    <action
        android:id="@+id/action_navigation_profile_to_
                        navigation_logged_user"
        app:destination="@id/navigation_logged_user" />
```

```
    <action
        android:id="@+id/action_navigation_profile_to_
                        navigation_registration"
        app:destination="@id/navigation_registration" />
</fragment>
```

6.5 Ukázka definice fragmentu

Obsah složky res/values

Zde zmíním dva soubory: **colors.xml** a **strings.xml**. V prvním zmíněném souboru **colors.xml** jsou definované barvy, které se používají napříč aplikací, jako třeba barva tlačítek. V druhém souboru jsou definované všechny řetězce, které jsou v aplikaci použité. Díky tomuto souboru jsou řetězce definovány na jednom místě, dají se tedy opravit či přepsat chyby v textu. Dalším plusem toho způsobu je přeložení aplikace do jiného nebo dalšího jazyka, stačí poté jen přepsat či přidat tento soubor s jiným jazykem.

7 Uživatelské testování

Testování aplikace proběhlo v rámci uživatelského testování. Uživatelům byla předložena aplikace a sada testovacích QR kódů. Uživatelé měli za úkol aplikaci vyzkoušet v plném rozsahu. Aplikaci testovalo celkem šest různých uživatelů, každý na jiném telefonu. Zprávu jsem dostal pouze od tří testerů a jsou vypsány níže.

7.1 Zprávy od testerů:

Tester A – Mobilní aplikace působí svěžím moderním dojmem. Ovládání uživatelského rozhraní je pohodlné a uživatelsky přívětivé. Líbili se mi hezky zpracované animace (například při otevírání údajů o instituci) a možnost zvolit si svůj preferovaný jazyk. Užitečná vlastnost je i zobrazení anglického překladu, pokud není preferovaný jazyk dostupný. Navigace po muzeu je zpracovaná intuitivně a návštěvník nemá problém jí začít okamžitě používat. Je k dispozici i našeptávač, pomocí kterého lze volit objekty rovnou bez nutnosti přepisování celého názvu. Čtečka QR kódů je rychlá a snadno použitelná. Aplikace poskytuje také možnost přidávat exponáty, což by mohlo usnadnit práci zaměstnancům muzeí. Mezi věci, které se mi nelíbili, můžu zmínit například samotný start aplikace. Hned po prvním spuštění se otevře čtečka QR kódů a to mi nepřipadá přívětivé pro úplně nového uživatele. Uvítal bych nějakou úvodní obrazovku a čtečku QR kódů umístit například doprostředka lišty s tlačítky. Mimo to mě zmátla možnost přidat preferovaný jazyk z obrazovky profilu uživatele. Tato funkcionalita je totiž umístěna v nastavení i zde a nevěděl jsem, která z možností je tedy platná.

Tester B – Aplikace je přehledná a z pohledu uživatele dobře použitelná. Není potřeba složitě bádat nad funkčností ikon. Velmi pěkné je při výběrech exponátů rozbalovací menu. Nemusí se vše vypisovat, což zvyšuje uživatelský komfort. Při spuštění aplikace se objeví čtečka kódu, což samozřejmě urychlí již registrovanému

nebo přihlášenému uživateli použití, nicméně by z mého pohledu mohl být vstup do aplikace přes přihlašovací stránku. Vyhovuje mi možnost výběru jazyka, již předem zobrazený seznam všech jazyků a zároveň filtrování hned po zadání znaku. Líbí se mi také vkládání exponátů do systému. Díky čemuž mohu jako uživatel rozšiřovat databázi exponátů a tím pomáhat muzeu a ostatním návštěvníkům. Vše je hezky graficky vytvořeno, aplikace působí příjemným nevtíravým dojmem. Speciálně animace hezky oživují používání aplikace.

Tester C – Aplikace mě hned zaujala samotnou ikonou, je jednoduchá, ale přesto pěkná a výstižná. Mobilní aplikace působí moderním dojmem, hlavně díky zvoleným barvám v ně aplikace. Přehlednost dominuje, což je pro mě, jako uživatele, velmi důležité. Snadno lze najít instituci i exponát. Líbí se mi nápovědník u výběru exponátu ručně, jelikož to urychlí čas a nemusím se vypisovat s celým názvem, například jménem a číslem místnosti. Profil uživatele také vypadá dobře. Celkově lišta ve spodu aplikace se mi moc líbí, opět přehledné ikony, které jasně určují, k čemu odkazují. Čtečka QR kódu funguje dobře, nezasekává se a reaguje rychle. Líbí se mi také animace. Jediné, co mohu vytknout je, že při spuštění aplikace se zobrazí čtečka QR kódu. Za mě by bylo lepší, kdyby se jako první zobrazil třeba seznam institucí, nebo nějaká úvodní strana.

7.2 Shrnutí reakcí od testerů

Ze zpráv od testerů lze na první pohled vyčíst, že aplikace se uživatelům líbí. Testerům se líbí animace, kterými aplikace disponuje, například při zobrazení karty s institucí. Uživatelé si pochvalují volbu preferovaného jazyka, kde se zobrazuje celý seznam a probíhá aktivní filtrování, dále vyhovuje rozbalovací seznam s možnostmi (instituce, budovy atd.), například při zadávání nového exponátu. Dva ze tří testerů zmínilo, že jim nevyhovuje obrazovky čtečky jako úvodní obrazovka a preferovali by úvodní obrazovku, či volbu jiné obrazovky. Tester B pochopil, že pro fungování aplikace, musí mít založený uživatelský účet a být přihlášený. Důležitým faktorem pro fungovaní je, že ani jednomu z testerů (i těch, co nenapsali zprávu) aplikace nepřestala pracovat, či neukazovala špatné hodnoty.

7.3 Možné vylepšení do budoucna

Vylepšení na základě výsledku testování

Nejčastěji zmiňovaným problém, na který testeři naráží, je úvodní obrazovka. Tou je v nynějším stavu čtečka QR kódů. Podle zpráv by uvítali nějakou úvodní obrazovku, či případnou změnu obrazovky například na seznam institucí nebo přihlašovací formulář. Úvodní obrazovka by mohla například obsahovat seznam aktualit, například formou karty na stejném principu jako je karta o instituci. Tyto aktuality by měli možnost přidávat správci instituce, pro toto vylepšení by se musel rozšířit server, mobilní i webová aplikace. Ale uživatelé by toto ocenili.

Druhým méně závažným problémem, který bych vytknul je, že tester B, si myslel, že pro používání aplikace je potřeba mít uživatelský účet a zároveň být přihlášený. Informace o tom, že uživatelský účet není potřeba, by mohla být také obsažena například na úvodní stránce, nebo při prvním spuštění aplikace.

S případným přidáním úvodní obrazovky by byla možnost předělat spodní navigační lištu. Přidat úvodní obrazovku na začátek navigace a přesunout čtečku na střed a vyvýšit oproti zbytku animace, udělat plovoucí tlačítko. V této době je to hojně využívaný design.

Vylepšení z vlastní hlavy

V aktuální verzi aplikace uživatel nemá možnost vidět jeho aktuálně zvolený jazyk pro překlady. Při návrhu aplikace jsem počítal s tím, že uživatel nebude mít potřebu znát informaci o svém vybraném jazyce, ovšem při používání aplikace by dle mého názoru bylo lepší, kdyby tato informace bylo někde viditelná, například v uživatelském profilu nebo přímo seznamu jazyků, v horní části zobrazit aktuálně zvolený.

Dalším možným vylepšením jsou „oblíbené instituce“. Tato možnost již existuje v uživatelském profilu, v této verzi aplikace není implementovaná. Implementace by byla možná například přidáním ikony srdíčka ke kartě instituce v seznamu institucí a uživatel by mohl dát srdíčko, zároveň by měl například srdíčko červené u již oblíbených institucí. Při volbě z uživatelského profilu by se uživateli zobrazil seznam pouze s oblíbenými institucemi.

Výrazným vylepšením by bylo přidat aplikaci funkcionalitu text-to-speech, kterou jsem z časových důvodů nestihl implementovat. Využít buďto externí aplikaci, nebo využít nějakou již existující knihovnu.

8 Závěr

V první části bakalářské práce se nachází seznámení s možnými technologiemi, které by mohli být pro vývoj aplikace využity. Následně byli analyzovány již existující aplikace, které fungují na podobném principu jako aktuálně vytvářená aplikace. Také proběhla sumarizace možností pro komunitní překlad.

Další částí bakalářské práce bylo vytvořit návrh prototypu aplikace, k tomuto se i úzce vážou uživatelské role. Na základě uživatelských rolí byli navrženy obrazovky a funkcionality aplikace. Podle tohoto návrhu byla aplikace implementována na operační systém Android. Výsledná aplikace byla poté otestována reálnými uživateli, kteří sepsali krátké zprávy o jejich zkušenostech s aplikací.

Hlavním cílem bylo vytvořit prototyp mobilní aplikace, kterou by uživatelé mohli využít v muzeích, či jiných institucích. To se podařilo natolik, že se jedná už o plnohodnotnou aplikaci, která se serverem a webovou aplikací může fungovat pro veřejnost. Cíle práce byly tedy splněny v plném rozsahu.

Možné rozšíření aplikace již byly zmíněny v kapitole **7.3**. Jako hlavní bych asi zmínil přidání text-to-speech funkcionality či vytvoření úvodní obrazovky aplikace s novinkami.

Přehled zkratek

API – Application programming interface

TTS – Text to speech

SDK – Software development kit

NFC – Near field communication

POI – Points of interests

HTML – Hypertext Markup Language

HTTP – Hypertext Transfer Protocol

XML – Extensible Markup Language

REST – Representational State Transfer

URL – Uniform Resource Locator

JWT – JSON Web Token

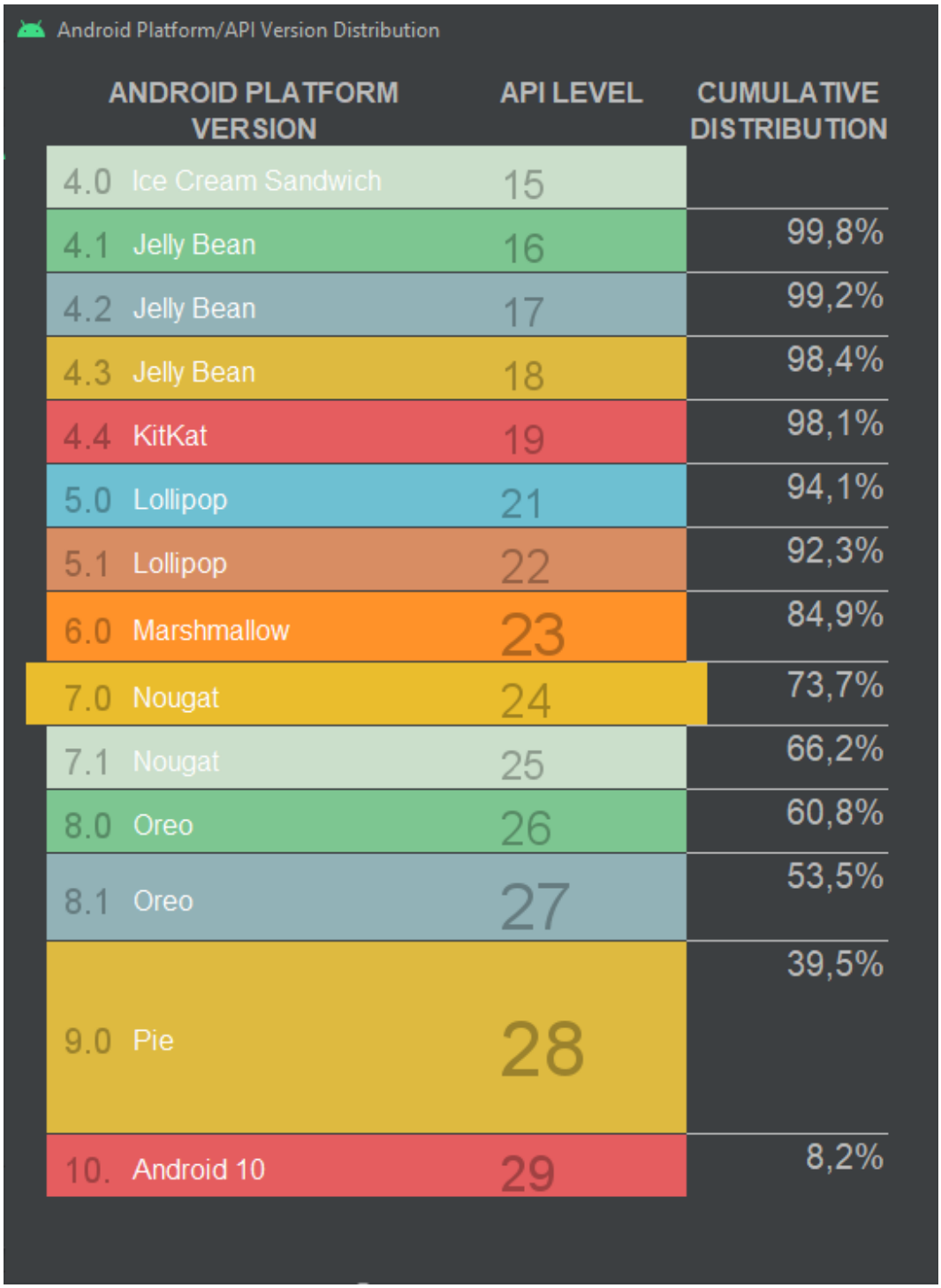
QR kód – Quick Response kód

Literatura

1) Google AR. (2018). Feature request : 3D Object Detection · Issue #418 · google-ar/arcore-android-sdk. [online] Dostupné na: https://github.com/google-ar/arcore-android-sdk/issues/418 [Citováno 15. 2. 2022].
2) Wikitude. (2020). Wikitude Store: Find Best Pricing for your Augmented Reality Experiences. [online] Dostupné na: https://www.wikitude.com/store/ [Citováno 15. 2. 2022].
3) Dushyanth (2022). Barcode scanner library [online] GitHub. Dostupné na: https://github.com/dm77/barcodescanner [Citováno 15. 2. 2022].
4) ZXing. (2020). ZXing library [online] Dostupné na: https://github.com/zxing/zxing [Citováno 15. 2. 2022].
5) Yuriy Budiyev. (2021). Code scanner library for Android, based on ZXing. [online] Dostupné na: https://github.com/yuriy-budiyev/code-scanner [Citováno 15. 2. 2022].
6) KingsMentor (2021). MobileVisionBarcodeScanner. [online] Dostupné na: https://github.com/KingsMentor/MobileVisionBarcodeScanner [Citováno 15. 2. 2022].
7) Android Developers. (2022). TextToSpeech. [online] Dostupné na: https://developer.android.com/reference/kotlin/android/speech/tts/TextToSpeech [Citováno 21. 2. 2022].

8) Drátek.CZ (2021). Sada 6 barevných NFC tagů [online] Dostupné na: https://dratek.cz/arduino/34737-sada-6-barevnych-nfc-tagu.html [Citováno 21. 2. 2022]
9) Wikimedia.org. (2022). Manhattan distance [online] Dostupné na: https://upload.wikimedia.org/wikipedia/commons/0/08/Manhattan_distance.svg [Citováno 21. 2. 2022]
10) Google Geo APIs Team (2017). Creating a Store Locator with PHP, MySQL & Google Maps | Google Maps APIs | Google Developers. [online] Dostupné na: http://web.archive.org/web/20170126150533/https://developers.google.com/maps/articles/phpsqlsearch_v3 [Citováno 21. 2. 2022].
11) Distribuce android zařízení
12) Mikešová, A. (2021) Analýza existujících mobilních průvodců po muzeích a památkách [online]. Bakalářská práce. Dostupné na: https://otik.uk.zcu.cz/bitstream/11025/44241/1/BP_Mikesova.pdf. [Citováno 21.3.2022].
13) Allez Interactive Inc. (2018). WikiCompass. App Store. [online] Dostupné na: https://apps.apple.com/gb/app/wikicompass/id1363789167?l=cs [Citováno 20.2.2022]
14) Google LLC (2011). Google Arts & Culture – Apps on google play, [online] Dostupné na: https://play.google.com/store/apps/details?id=com.google.android.apps.cultural [Citováno 20.2.2022]
15) American Museum of Natural History (2010). Explorer – AMNH NYC. App Store. [online] Dostupné na: https://apps.apple.com/us/app/explorer-amnh-nyc/id381227123 [Citováno 20.2.2022]
16) Karta instituce
17) Retrofit, (2013). Retrofit. Square, [online] Dostupné na: https://square.github.io/retrofit/ [Citováno 20.4.2022]

Přílohy



1 - Distribuce android zařízení podle verzí