FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Mobilní aplikace k určování lokace krajinomalby
Student:	Adam Fišer
Vedoucí:	Ing. Jiří Zoudun
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Cílem práce je vytvořit funkční prototyp Android aplikace k webové aplikaci artopos.net a upravit stávající webovou aplikaci. Aplikace bude umět načítat data z REST API a ukazovat na mapě nebližší krajinomalby z databáze. Dále bude umět nahrát příspěvek ke krajinomalbě s reálnou fotkou krajiny, komentářem a polohou. V aplikaci půjde přidávat nové malby a zobrazit si přidané malby s příspěvky. K tomu bude potřeba rozšířit back-end tak, aby uměl komunikovat přes REST API.

1) Analyzujte aktuální webovou aplikaci.
2) Navrhněte možná vylepšení webové aplikace.
3) Analyzujte požadavky uživatelů pro Android aplikaci.
4) Navrhěte vhodný koncept Android aplikace. Zvolte potřebnou funkcionalitu na základě analýzy.
5) Navrhněte API, přes které bude komunikovat mobilní aplikace s back-endem.
6) Naimplementujte úpravy webové aplikace.
7) Úpravy řádně otestujte.
8) Naimplementujte prototyp Android aplikace na základě návrhu.
9) Prototyp řádně otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

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

V Praze dne 27. listopadu 2017

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Mobilní aplikace k určování lokace krajinomalby

Adam Fišer

Vedoucí práce: Ing. Jiří Zoudun

14. května 2018

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 14. května 2018 .....

Odkaz na tuto práci

Fišer, Adam. *Mobilní aplikace k určování lokace krajinomalby.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Hlavním cílem práce je návrh a implementace Android aplikace, která slouží k lokalizaci krajinomaleb a zobrazování jejich reálných fotografií. Android aplikace bude komunikovat s back-endovou aplikací pomocí REST API. Webová aplikace Artopos.net, ze které vychází Android aplikace, má několik problémů jak v návrhu uživatelského rozhraní, tak v implementaci.

Proto jsem navrhl a vytvořil Android aplikaci, která značně zjednoduší a zefektivní nejčastější scénáře. Přidal jsem uživatelské účty, které zjednoduší přidávání obsahu. Dále jsem vytvořil back-endovou aplikaci v jazyce PHP, která ukládá data do MySQL databáze a poskytuje REST API pro komunikaci s Android aplikací.

Pro implementaci Android aplikace jsem použil framework NativeScript, který nabízí řadu výhod oproti čistě nativnímu vývoji. Mezi tyto výhody patří odstínení od programování nativního UI, správa projektu a možnost vytvoření multiplatformní aplikace (pro Android a iOS) bez duplicity kódu. REST API jsem vytvořil za použití jazyku PHP 7.1 a framworku Symfony 4.

Přínosem této práce je oživení projektu Artopos.net a zjednodušení jeho používání. Díky mobilní aplikaci je mnohem jednodušší a praktičtější využívat aplikaci v terénu a na cestách a přidávat nové příspěvky bez přístupu k osobnímu počítači. Přidání aplikace na novou platformu může vést k rozšíření uživatelské základy a také k oslovení nových uživatelů, kteří díky aplikaci Artopos.net mohou najít zájem o umění.

V příloze lze nalézt zdrojové kódy Android aplikace a back-endové aplikace. Dále příloha obsahuje PDF soubor tohoto textu a jeho zdrojové soubory ve formátu LaTeX, APK balíček s Android aplikací a model původní databáze. Také se zde nachází zadání a výsledky uživatelského dotazníku a export výsledků pre-test a post-test dotazníku použitých při testování použitelnosti.

Klíčová slova Android OS, analýza stávající webové aplikace, návrh aplikace, Artopos.net, umění, geolokace, uživatelská přívětivost, REST API, NativeScript

Abstract

The major goal of this thesis is design and implementation of an Android application which localizes landscape paintings and shows their actual photos. The Android application will communicate with back-end application via REST API. Web application Artopos.net on which is the Android application based on, has several issues in both UI design and implementation.

That is why I designed Android application, which greatly simplifies the most common scenarios. I added user accounts to make it easier for users to add new content. I also created back-end application in PHP that stores data to MySQL database and provides REST API.

For implmenetation of the Android application I used framework NativeScript, which offers a lot of advantages compared to pure native development. These advantages are encapsulaction of programming Native UI, project management and the ability to create multiplatform application without duplicated codebase (even for UI). The REST API was created in PHP 7.1 using framework Symfony 4.

The main asset of this thesis is recovery of the project Artopos.net and simplification of its use. Thanks to the mobile application its easier and practical to use it in outdoor environment and add new posts without accessing a computer. Expanding the application on new platform can lead to increase number of active users and address new users that can discover their interest in art.

There are included source codes of Android application and back-end application in the attachment with PDF file and source code in LaTeXformat of this text. There is also included APK package of the Android application, export of the original database model. Finally the attachment includes questions and answers of the user form and results of the pre-test and post-test form used during usability testing.

Keywords Android OS, analysis of the current web application, software design, Artopos.net, art, geolocation, user-friendly, REST API, NativeScript

Obsah

Seznam obrázků

Úvod

Artopos.net je webová aplikace, která umožňuje lokalizovat krajinomalby a jejich reálné fotografie. Aplikace se snaží usnadnit identifikaci krajinomaleb a zjistit geografickou polohu jejich předlohy. Webová forma existuje již několik let, ale z důvodu technických nedostatků a nedořešeného uživatelského rozhraní není hojně využívána, funguje ovšem jako dobrá databáze krajinomaleb pro historiky a znalce.

Řadu těchto problémů by vyřešilo jednoduché přenesení na mobilní platformu, díky kterému by mohli uživatelé lokalizovat malby a přidávat příspěvky v reálném čase za využití běžných funkcí chytrých mobilní zařízení, jako jsou geolokace nebo fotoaparát. Používání mobilních zařízení se stalo běžnou součástí našeho života a téměř každý dnes vlastní chytrý mobilní telefon, který nosí téměř nepřetržitě u sebe. Proto by uživatelé aplikace mohli nahrávat příspěvky z jakéhokoliv místa, které je pokryto mobilní sítí.

V dnešní době dominují na trhu převážně dva operační systémy na mobilní platformu a to jsou Android OS od společnosti Google a iOS od společnosti Apple. Nejrozšířenějším je právě operační systém Android, který zastupuje přes 80% mobilních zařízení na trhu (viz [1]). Díky tomu, že je Android dostupný pro výrobce mobilních zařízení, vyskytuje se jak na lacinějších, tak na prémiových produktech. Proto jsem se rozhodl vyvíjet právě pro tento Operační systém. Sám vlastním mobilní zařízení se systémem Android, což mi usnadní vývoj a následné testování.

Původní aplikace byla zaměřena spíše na znalce a nadšence do výtvarného umění či samotné umělce, což je velice úzká skupina lidí. V rámci této bakalářské práce se tato skupina může rozšířit také o běžné uživatele, kteří rádi cestují a zajímají se o umění, nebo navštěvují galerie a výstavy. Ještě předtím, než jsem začal pracovat na této bakalářské práci, jsem dostal řadu pozitivních ohlasů na takovou aplikaci a proto si myslím, že by výsledná aplikace tuto novou uživatelskou skupinu mohla zaujmout a tím zároveň přivést nové zájemce o umění, jelikož zájem o výtvarné umění v posledních letech všeobecně klesá.

Téma jsem si zvolil, protože jsem viděl potenciál aplikace Artopos.net a

chtěl jsem se podílet na jejím vývoji. Zároveň bylo na první pohled zřejmé, že aktuální řešení není ideální, má spoustu chyb v návrhu uživatelského rozhraní a je téměř nepoužitelné pro vyhledávání v reálném čase, které by velice usnadnilo používání aplikace. Také se sám zajímám o umění a příležitostně navštěvuji galerie a výstavy.

Prvním cílem rešeršní části práce je analyzovat aktuální webovou aplikace Artopos.net a analyzovat uživatelské požadavky pro návrh aplikace na operační systém Android k této webové aplikace, dále budu analyzovat podobné aplikace které řeší stejné problémy spojené s ovládáním mapy a geolokací. Dalším cílem je prostudovat standard REST API pro komunikaci webové aplikace a Android aplikace. Posledním cílem rešerše je vybrat vhodné nástroje pro implementaci REST API a jeho zabezpečení, nastudovat vývoj Android aplikací a vybrat vhodný programovací jazyk a odpovídající knihovny.

Cílem praktické části práce je vytvořit funkční prototyp Android aplikace k webové aplikaci Artopos.net a upravit stávají webovou aplikaci. Aplikace bude umět načítat data z REST API a ukazovat na mapě nejbližší krajinomalby z databáze. Dále bude umět nahrát příspěvek ke krajinomalbě s reálnou fotkou krajiny, komentářem a polohou. Také si v ní bude moct uživatel zobrazit seznam maleb s možností vyhledávání a filtrování. V aplikaci bude možné zobrazit detail malby s příspěvky. K tomu bude potřeba rozšířit back-end tak, aby uměl komunikovat přes REST API. Posledním cílem je řádně otestovat REST API a prototyp Android aplikace.

Kapitola 1

Současná řešení

V této kapitole se věnuji popisu uživatelského rozhraní a fungování stávající webové aplikace a dalších Android aplikací, které fungují na podobném principu nebo řeší podobné problémy, které budu řešit při návrhu Android aplikace.

1.1 Současná aplikace Artopos.net

Webová aplikace Artopos.net funguje již mnoho let, za tu dobu se podařilo nasbírat a lokalizovat řadu uměleckých děl většinou po České republice ale existují příspěvky i z jiných koutů Evropy. Od té doby ale pomalu skomírala a nikdy nebyla schopná získat si stabilní uživatelskou základnu. Aplikace má na první pohled řadu problémů, jako je pomalé načítání nebo nedořešené UI. Na její vývoj nebyl dostatek prostředků a to se podepsalo celkově na kvalitě a struktuře projektu. Také administrace aplikace byla zanedbávána a byla spíš brána jako občasný koníček správce, jak jsem se dozvěděl z rozhovorů.

Na hlavní stránce aplikace (viz 1.1) je vidět hlavička aplikace, pod kterou se nachází mapa přes celou šířku obrazovky. Mapa je při každém načtení vycentrovaná tak, aby zachycovala Českou republiku a Slovensko. Na mapě jsou vidět červené a zelené piny znázorňující malbu, přičemž zelené značí, že malba již byla nalezena a tudíž je jejich poloha přesná, zatímco červené označují malbu, která zatím nalezena nebyla a její poloha je pouze přibližná. Po kliknutí na pin se zobrazí box s miniaturou a názvem. Kliknutím na něj se uživateli zobrazí stránka s detailem malby.

Pod ní se nachází panel s vyhledávacím formulářem a horizontálně posuvný panel (Slider), ve kterém jsou miniatury maleb. Po kliknutí na miniaturu se zobrazí stránka s detailem malby. Vyhledávat je možné podle názvu malby, autora, instituce,ve které se malba nachází, nebo podle kraje/regionu. Po odeslání formuláře se uživateli zobrazí stránka totožná s úvodní, ve které se ovšem nachází jen výsledky odpovídající vyhledávání.

V detailu malby (viz 1.2) se nachází mapa, na které jsou kromě mapy vyobrazené i lokality příspěvků, větší fotografie a popis malby. Dále je zde jméno autora, kliknutím na něj se uživateli zobrazí detailní informace o autorovi, a název instituce, které může sloužit odkaz na stránky instituce. Také je zde prvek pro sdílení malby na sociálních sítích a tlačítko pro přidání příspěvku.

Pod detailem je seznam příspěvků, které obsahují email a jméno autora příspěvku, datum vytvoření, text a fotografii. Pod nimi je panel se záložkami pojmenovanými podle štítků, které obsahuje daná malba a v nich jsou další malby také označené daným štítkem.

Po kliknutí na `Přidat příspěvek` (viz 1.3) se zobrazí okno s formulářem, které obsahuje pole pro jméno, email, fotografii a text. vedle těchto prvků je mapka s právě jedním pinem, který značí polohu daného příspěvku, pin může uživatel posouvat po mapě a také může zadat zeměpisnou šířku a délku v textové poloze do polí ve spodní části mapy. Kliknutím na tlačítko `Vložit` se příspěvek odešle.

V aplikaci není možné přidávat vlastní malby, autory a instituce. Nové malby lze vytvořit pouze správcem v administraci aplikace. Z rozhovoru se správci aplikace jsem zjistil, že je tato možnost pouze v administraci záměrně, protože chtějí mít kontrolu nad kvalitou přidaných maleb.

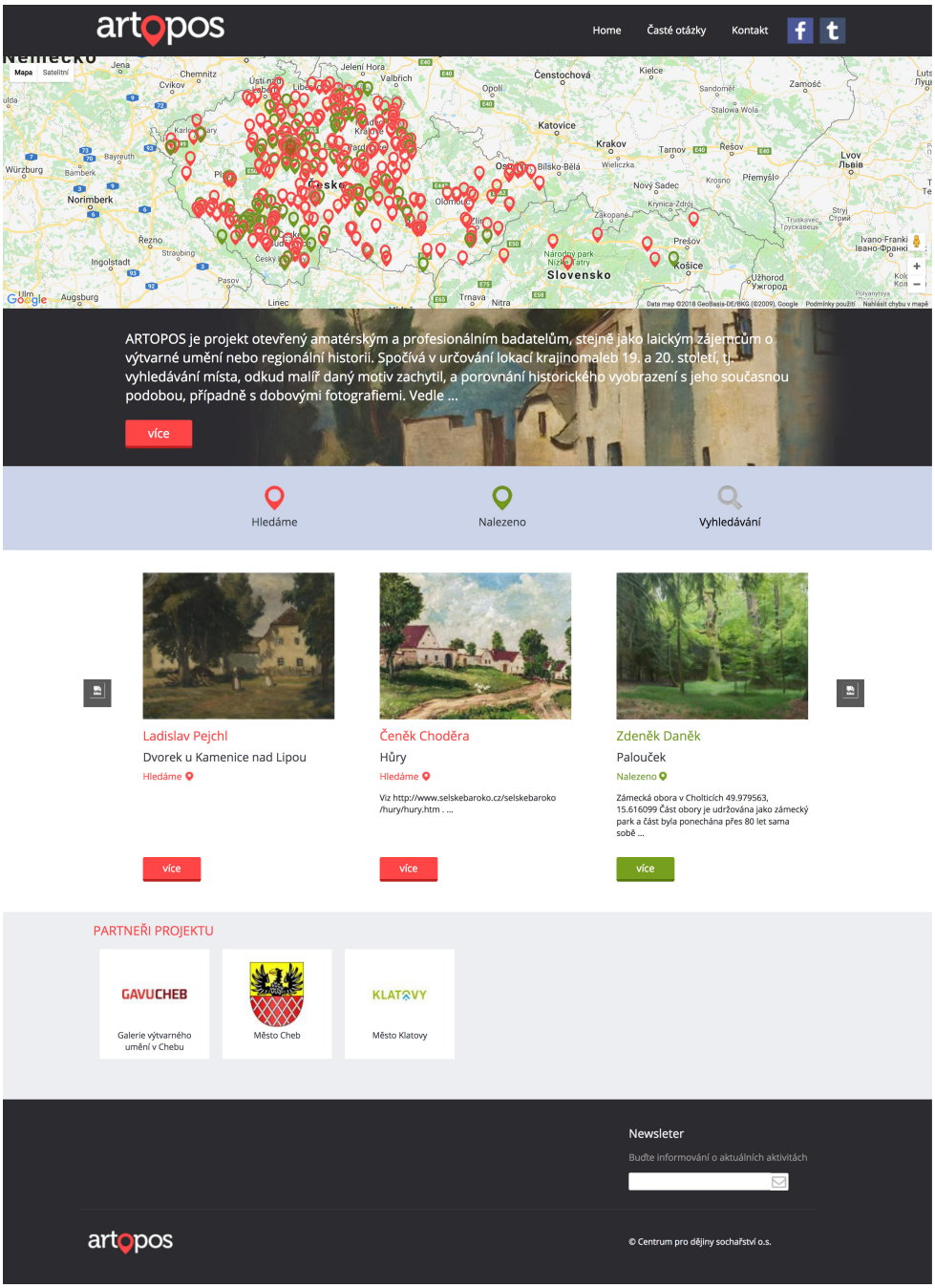
1.2 Aplikace Google Maps

Google Maps je základní aplikace využívající geolokaci pro zobrazování nejbližších určitých míst a objektů a zároveň se jedná o nejpoužívanější aplikaci toho typu. Slouží k lokalizaci jakýchkoliv zajímavých míst, restaurací, obchodů, infrastruktury nebo může sloužit jako GPS navigace. Aplikace je vydaná přímo společností Google, která stojí za vývojem operačního systému Android.

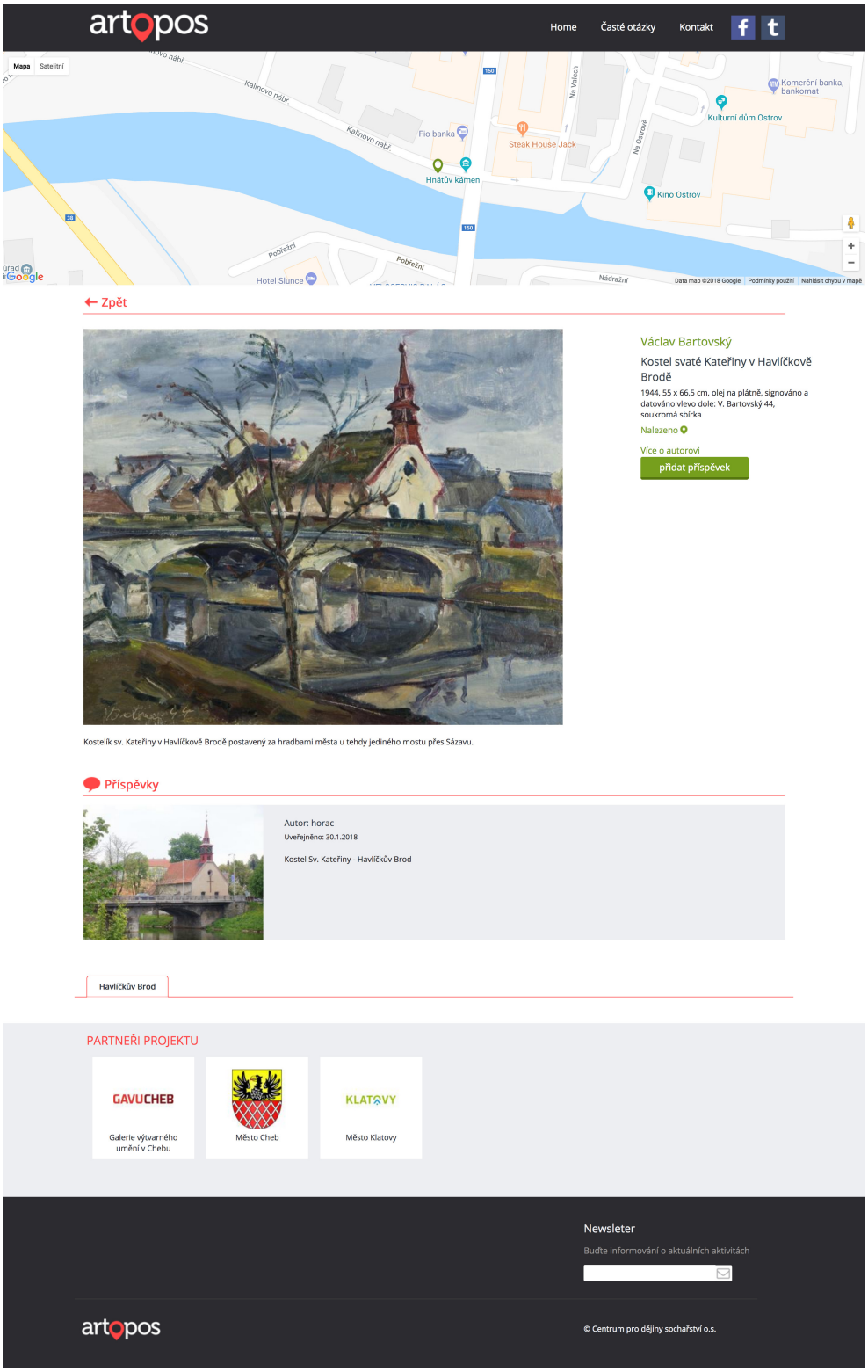
Základní obrazovkou této aplikace je mapa (viz 1.4). Horní lišta je nahrazena textovým polem pro vyhledávání. V dolní části se nachází lišta s tlačítky `Prozkoumat`, `Autem` a `MHD`, která upravují informace a místa viditelná na mapě. Po kliknutí na bod na mapě se v dolní části objeví lišta se základními informacemi (viz obrázek 1.5), ze které se dá kliknutím nebo vytažením dostat na detail místa (viz obrázek 1.6). V detailu se nachází v horní části fotografie nebo posuvná galerie fotografií. Pod ní je název a záložky s přehledem, recenzemi a fotkami.

1.3 Aplikace Geocaching

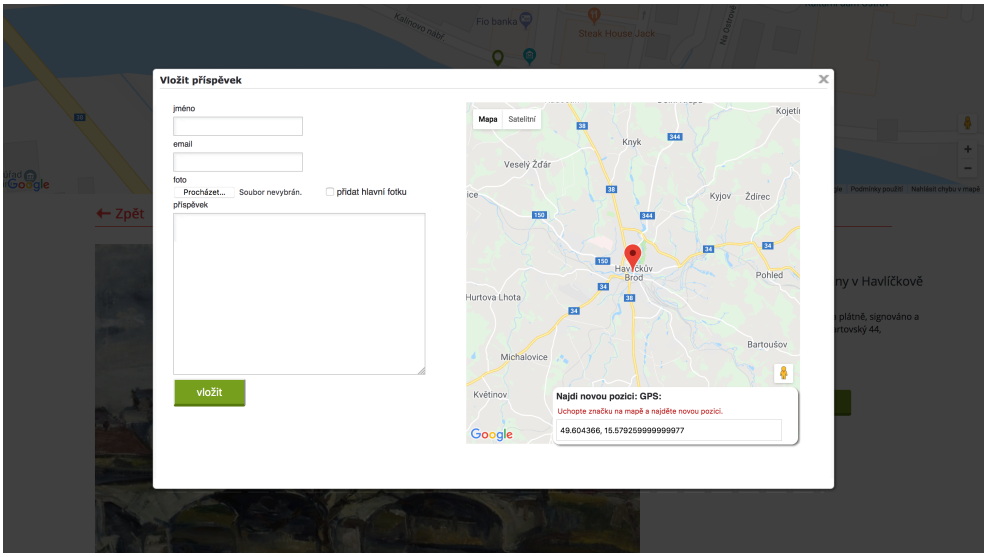
Nejblíže konceptem a designem je aplikace Geocaching podporující stejnojmennou hru. Aplikace slouží k lokalizaci Cache, což jsou reálné objekty většinou zabalené do odolného kontejneru a schované na méně dostupných místech. Smysl hry spočívá v tom, že hráč, který Cache objeví si může prohlédnout



Obrázek 1.1: Artopos.net - hlavní strana



Obrázek 1.2: Artopos.net - detail malby

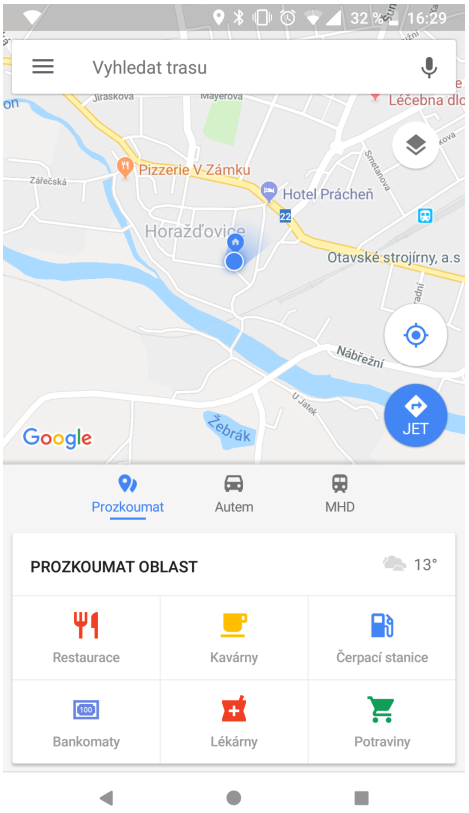


Obrázek 1.3: Artopos.net - přidání příspěvku

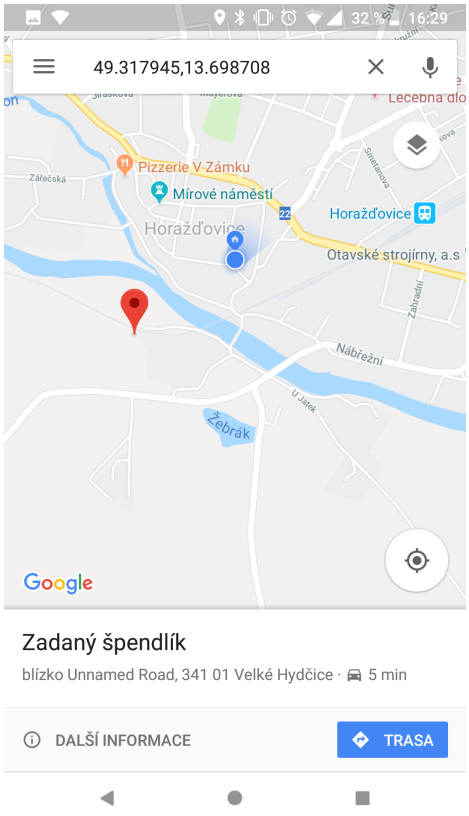
předmět (případně si ho vzít a nechat na jeho místě jiný) a to vše zapsat se do zápisníku, který u Cache nachází.

Úvodní obrazovka (viz 1.7) se skládá z mapy téměř přes celý displej podobně jako u Google Maps. Po kliknutí na značku na mapě označující Cache se objeví spodní panel se základními údaji, který po rozkliknutí vede na detail Cache (viz obrázek 1.8). V horní liště je na levé straně od názvu aplikace ikonka otevírající boční menu a na druhé straně možnost filtrů a vyhledávání. V rozích mapy jsou také ikonky pro její ovládání (volba terénu, aktuální poloha).

Detail Cache (viz 1.9) je rozdělený do záložek, ve kterých jsou další informace. V prvním jsou podrobné textové informace o dané Cache. Ve druhé se nachází Log a ve třetím Aktivita. Text v první záložce se dá rozkliknout na podrobný popis, kde je, podle mého názoru, až příliš mnoho textových informací najednou.

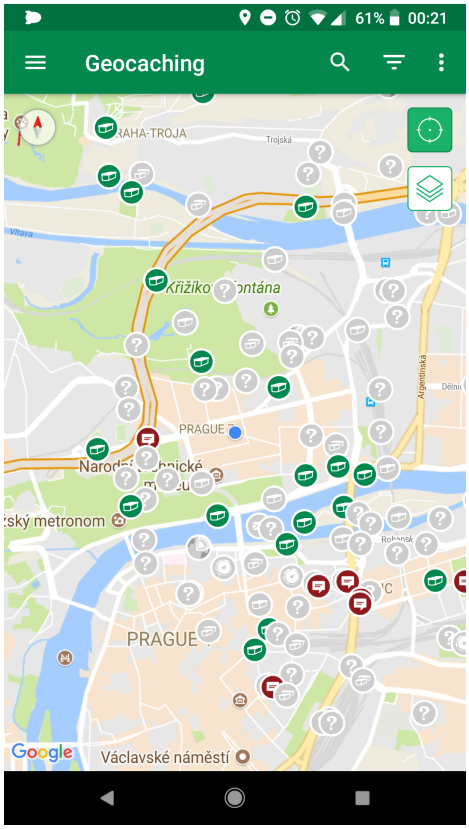


Obrázek 1.4: Google Maps - hlavní obrazovka

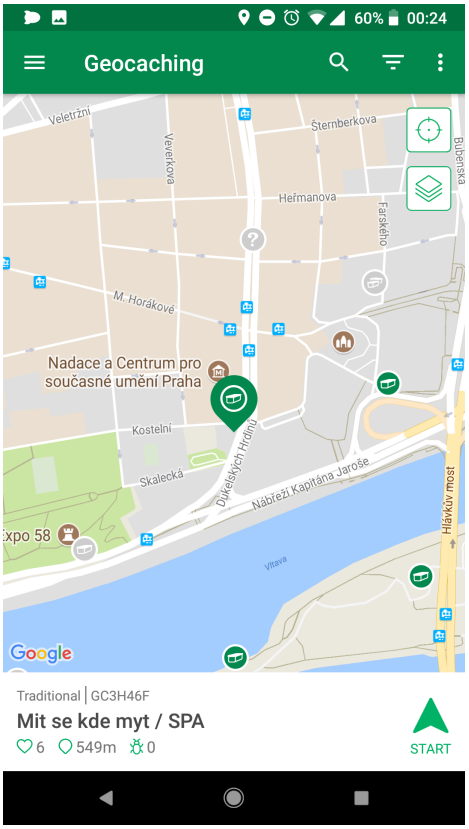


Obrázek 1.5: Google Maps - vybraný pin

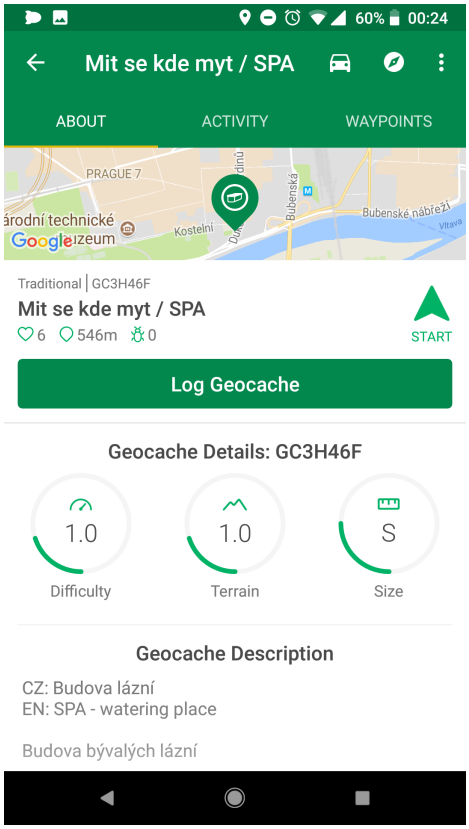
Obrázek 1.6: Google Maps - detail místa



Obrázek 1.7: Geocaching - hlavní obrazovka



Obrázek 1.8: Geocaching - vybraná cache



Obrázek 1.9: Geocaching - detail cache

Kapitola 2

Analýza

V této kapitole se zabývám analýzou uživatelských požadavků, scénáři použití aplikace a požadavky, ze kterých následně vychází návrh aplikace. K vytvoření diagramů jsem používal webovou aplikaci draw.io.

2.1 Analýza uživatelských požadavků

V této sekci jsem použil znalosti získané v předmětu BI-SI. Popisuji zde, jak jsem postupoval při analýze uživatelských požadavků a jaké informace jsem se z ní dozvěděl. Nejprve jsem se zaměřil na uživatelský dotazník, poté jsem vytvořil scénáře, na základě kterých jsem vytvořil funkční požadavky.

2.1.1 Uživatelský dotazník

Vytvořil jsem dotazník rozdělený do několika částí. První část je pro uživatele, kteří již znají aplikaci Artopos.net. Další část je pro uživatele, kteří již použili aplikaci na podobné bázi (konkrétně Geocaching). V poslední části se dotazuji na umění a cestování, abych zjistil, zda je tato aplikace schopná oslovit i nové uživatele. Z dotazníku jsem chtěl zjistit, jestli by cílová skupina, která už projevila zájem o Artopos.net, používala mobilní aplikaci a také, zda by taková aplikace přilákala nové uživatele, zejména ty, kteří projevují zájem o umění a cestování.

Dotazník jsem vytvořil pomocí webové aplikace Google Forms a distribuoval ho pomocí sociálních sítí v digitální podobě. Díky tomu bylo jednodušší vyhodnocení a export výsledků. Jeho zadaní se nachází v příloze.

2.1.1.1 Vyhodnocení dotazníku

Dotazník jsem cílil na uživatele, které již znají Artopos.net a nebo běžné uživatele mobilních zařízení, kteří by mohli hrát jakoukoliv geolokační hru a také na ty, kteří by mohli být zapálení do cestování či umění.

Dotazník byl převážně složen ze zaškrtávacích otázek, kde si uživatel vybere jednu z odpovědí nebo z otázek, kde je možné zaškrtnout více (nebo žádnou) odpověď. V několika otázkách byl prostor pro vlastní textovou odpověď. Sekce o aplikaci Artopos.net nebyla uživateli zobrazena, pokud v předchozí otázce neoznačil odpověď, že aplikaci již použil a stejný princip jsem použil u sekce o aplikaci Geocaching. Dotazník jsem rozesílal prostřednictvím sociálních sítí.

2.1.1.2 Výsledky dotazníku

Z první části, jak můžete vidět na obrázku 2.1, vyplynulo, že větší část uživatelů, kteří již znali aplikaci Artopos.net, ji využívali převážně pasivně a někteří o ní slyšeli, ale nikdy ji nepoužili.

V další sekci o využívání mobilních aplikací založených na geolokaci vyplnynulo, že většina uživatelů má zkušenosti s takovou aplikací a využívá ji občasně nebo na denní bázi. Co se týká odpovědí k aplikaci Geocaching, většina dotazovaných zná hru Geocaching. Co se týká využití mobilní aplikace k této hře, převažuje využití aplikace Geocaching, která se uživatelům používá jednoduše a nebyly k ní připomínky.

Z části zaměřené na cestování vyplynulo, že naprostá většina dotazovaných ráda cestuje převážně po Evropě a zhruba třetina lidí cestuje po České republice(viz obrázek 2.3) a to několikrát do roka (viz obrázek 2.4). Z odpovědí dotazníku vyplynulo, že většina dotázaných ze všech cílových skupin by měla zájem o mobilní aplikaci tohoto typu (viz obrázek 2.4).

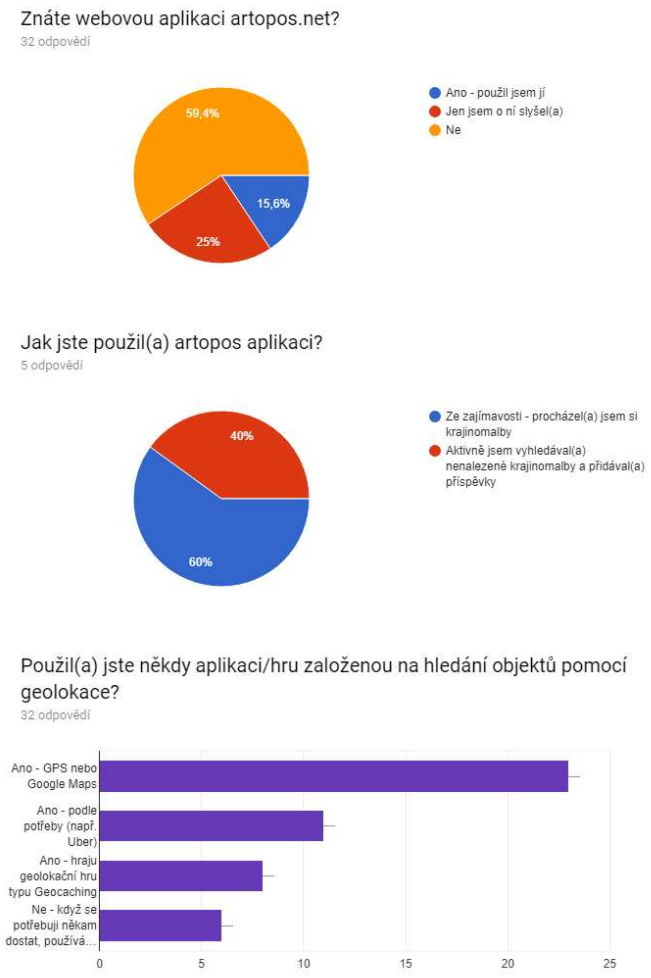
2.1.2 Scénáře

V této sekci najdete uživatelské scénáře aplikace, které částečně vycházejí z webové aplikace. Vytvořil jsem 3 hlavní scénáře na základě cílů uživatele. Prvním cílem je zjistit informace o konkrétní krajinomalbě. Dalším cílem je vyhledat nejbližší krajinomalby. Posledním cílem, kterým se budu zabývat, je nalezení přesné lokace malby a přidat příspěvek.

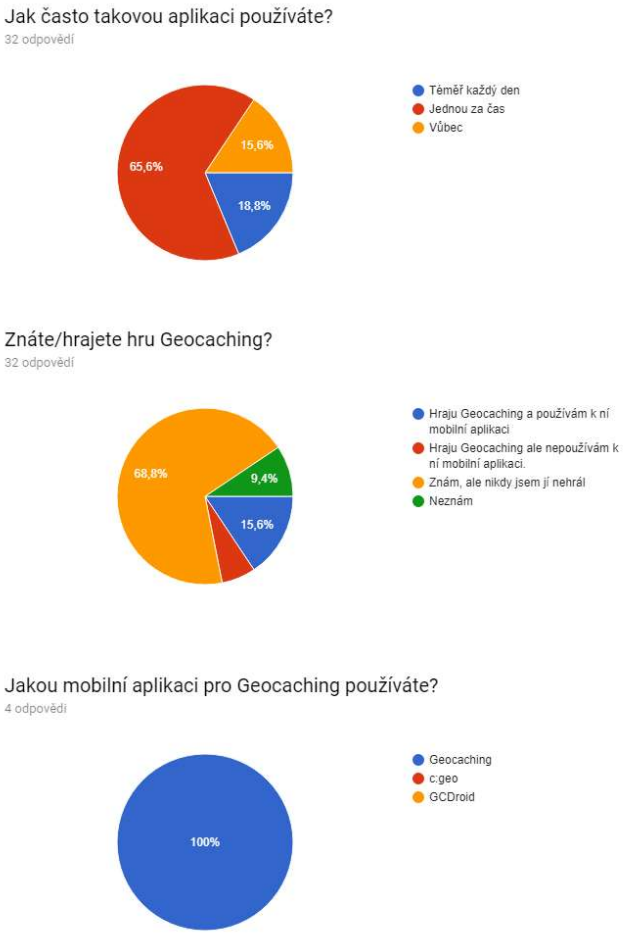
2.1.2.1 Vyhledání nejbližších krajinomaleb

Uživatel nemá konkrétní krajinomalbu, ale chce vyhledat nejbližší malby ve svém okolí, které by mohl lokalizovat nebo navštívit. Ve webové aplikaci tento scénář musí dělat tak, že si pomocí Google Maps prvku na webové stránce ručně (posouváním) najde svojí polohu. Na mapě se zobrazí dané malby, které si uživatel může dál procházet. Není zde žádná možnost zadávání polohy, adresy nebo použití aktuální polohy ze systému.

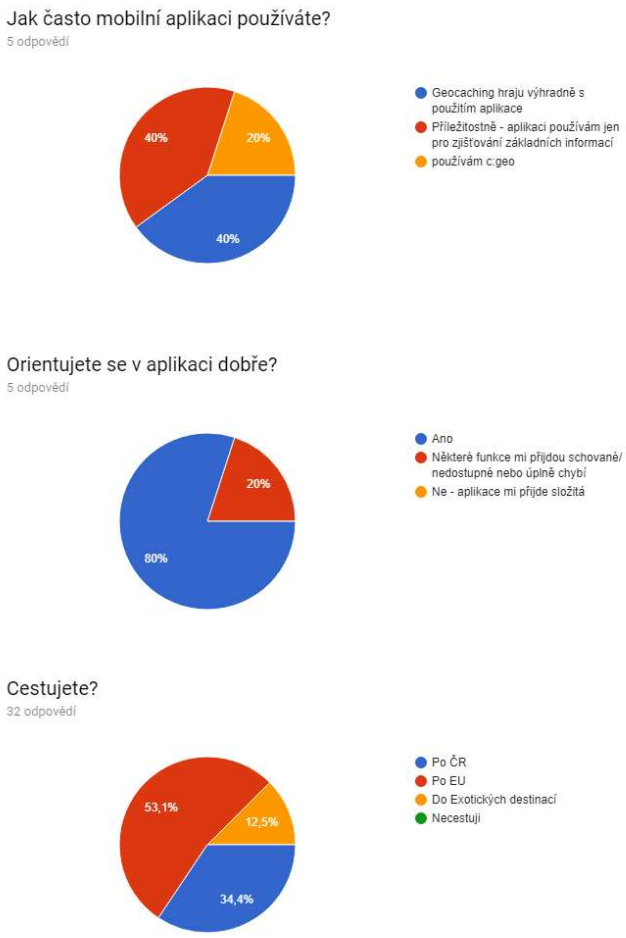
Také lze vyhledávat v seznamu pomocí vyhledávacího formuláře, ve kterém je napevno seznam krajů a několik zahraničních lokalit. Tato varianta je velice omezující, protože malba u sebe musí mít vyznačený kraj, ve kterém



Obrázek 2.1: Výsledky dotazníku 1



Obrázek 2.2: Výsledky dotazníku 2



Obrázek 2.3: Výsledky dotazníku 3

Obrázek 2.4: Výsledky dotazníku 4

se nachází, chybí přesné zadání polohy a je nezbytné vědět, ve kterém kraji/regionu se nacházíte. Tento problém by se dal vyřešit například napojením na Google Geocoding API, které umožňuje získat souřadnice z textové adresy (včetně automatického doplňování).

2.1.2.2 Vyhledání známých krajinomaleb

Uživatel zná název krajinomalby nebo jejího autora a chce o ní získat informace. Ve webové aplikaci si uživatel zobrazí úvodní stránku, kde se níže nachází formulář pro vyhledávání. Známé informace zadá do vyhledávacího pole. V seznamu výsledků si najde odpovídající krajinomalbu a kliknutím na ni si zobrazí její detail.

2.1.2.3 Lokalizace krajinomalby a přidání příspěvku

Uživatel se vyskytuje v okolí krajinomalby, kterou má za cíl lokalizovat. Vyhledávat může podle autora ve formulářovém poli typu Select. To značně snižuje uživatelskou přívětivost pro velké množství autorů. Dále je možné vyhledávat podle Instituce, zde Select tolik nevadí, protože institucí není takové množství. Podobně je možné vyhledávat podle krajů (také Select) a podle částečného nebo úplného názvu nebo jména autora.

Uživatel si vytiskne nebo zapamatuje přibližnou lokalitu, odkud by malba mohla pocházet. Podle svých navigačních a orientačních schopností se dostane co nejblíže danému místu. Označí si místo, ze kterého podle něj autor namaloval dané dílo. Poté si vyhledá krajinomalbu podle názvu ve webové aplikaci a v detailu malby přidá příspěvek.

2.1.3 Funkční požadavky

Funkční požadavky zobrazují vstupy, které by měl systém akceptovat a jaké výstupy by měl produkovat. Dále obsahují informaci o tom, jaká data se ukládají a poskytují dalším systémům. V neposlední řadě zobrazují, jaké výpočty by měl systém provádět.[2]

Zobrazení nebližších maleb Zobrazení nejbližších maleb na mapě pomocí GPS.

Zobrazení informací o malbě Zobrazení detailu malby s fotografií, informacemi, polohou a příspěvky.

Zobrazení příspěvků malby Zobrazení příspěvků s textem, fotografií a polohou.

Zobrazení polohy malby Zobrazení přesné polohy malby na mapě.

Přidání příspěvku k malbě Příspěvek může obsahovat polohu a text. Aplikace také umožní nahrát k příspěvku fotografii nebo ji vybrat z galerie. Uživatel musí být přihlášený, aby mohl přidat příspěvek.

Zobrazení seznamu maleb podle filtru Aplikace umožní filtrovat podle názvu, podle jména autora, podle informace, zda byla již malba nalezena a podle názvu instituce, ve které se nachází.

Vyhledání maleb podle určité polohy Aplikace umožní vyhledat malby nacházející se v určité lokalitě.

Zobrazení přidaných příspěvků uživatelem Aplikace umožní zobrazit příspěvky přidané uživatelem.

Registrace Aplikace umožní uživateli zaregistrovat si účet.

Přihlášení Aplikace umožní uživateli přihlásit se ke svému účtu.

2.1.4 Nefunkční požadavky

Nefunkční požadavky zahrnují obecně požadavky na platformu, spolehlivost, výkon a podporovatelnost.

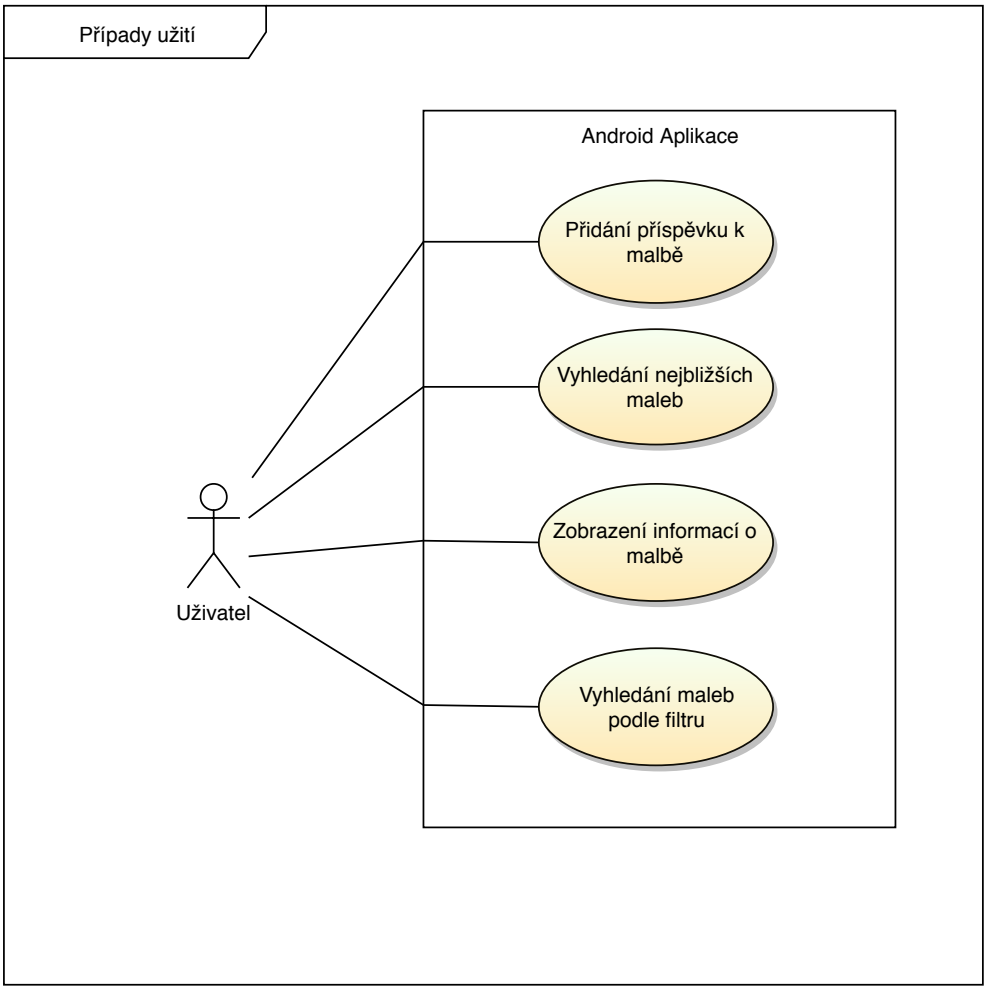
- Android, verze 4.2 a vyšší
- Mobilní data
- GPS
- Fotoaparát
- Přístup k úložišti

2.1.5 Případy užití

Případy užití (use case) inicializuje herec(Actor). Případ užití definuje sekvenci interakcí mezi hercem a systémem. Systém je vyobrazený jako box, případ užití je vyobrazený jako elipsa uvnitř boxu. Asociace komunikace propojují herce s případy užití, kterých se účastní. Typy vztahu mezi případy užití jsou Include a Extend.[3]

Na obrázku 2.5 jsou vidět vyobrazené případy užití Android aplikace. Účely aplikace budou hledání lokací krajinomaleb v terénu, což vychází především z obecných předpokladů k dané aplikaci, aplikace bude také sloužit ke snadnějšímu vyhledávání krajinomaleb za pomocí geolokace a také přidávání nových příspěvků k existujícím malbám.

V původní aplikaci nelze přidávat malby uživateli. Vytvářet nové malby může pouze správce v administraci. Jelikož majitelé požadují kontrolu nad



Obrázek 2.5: Diagram případů užití

přidanými malbami a jejich kvalitou, nebudu tuto část implementovat ani do Android aplikace.

V této práci jsem se nezabýval analýzou, návrhem ani implementací webové aplikace a administrační části a proto nejsou ani v tomto diagramu zahrnuté případy užití spojené s administrací, jako je přidávání a revidování maleb, autorů a institucí.

2.2 Analýza stávající webové aplikace

V této sekci se zabývám analýzou stávající webové aplikace Artopos.net.

2.2.1 Heuristické vyhodnocení uživatelského rozhraní

V této části se věnuji heuristickému vyhodnocení, které slouží k odhalení nedostatků v rozložení prvků a navigace uživatele. Tuto metodu jsem zvolil, protože je použitelná v rané části návrhu poukáže na nedostatky a chyby uživatelského rozhraní za relativně krátkou dobu. Navíc tato metoda nevyžaduje kontakt s klientem, jelikož se jedná o expertní vyhodnocení. Úkolem bylo projít stránky aplikace a odpovědět na následující otázky. [4]

- Ví vždy uživatel, kde se nachází (navigace)? Ví vždy uživatel, v jakém je aplikace stavu?
- Jsou názvy, popisky a jinak používané termíny v souladu s terminologií cílové skupiny?
- Jsou názvy, popisky a jinak používané termíny srozumitelné cílové skupině? Jsou názvy kategorií (například v menu) voleny s ohledem na srozumitelnost pro cílovou skupinu?
- Poskytuje každá akce zpětnou vazbu srozumitelnou pro uživatele?
- Může se uživatel dostat z každé situace? Obejde se takové opuštění bez nutnosti opakovat dlouhou sekvenci akcí? Je použití tlačítek správné?
- Jsou všechny objekty, akce a jiné prvky vidět, kdykoliv je jich potřeba, aniž by si je uživatel musel pamatovat?
- Jsou všechny akce, uspořádání a význam konzistentní s očekáváním cílové skupiny?
- Odpovídá vzhled aplikace a ovládacích prvků cílové skupině?
- Je plocha zobrazení využita přiměřeně (vzhledem k celkovému vzhledu)?

Hlavní stránka aplikace (na obrázku 1.1) vyšla z heuristického vyhodnocení relativně kladně, přesto jsem zde na narazil několik problémů. Prvním z nich je, že `Slider` (prvek pro horizontální rolování s miniaturami maleb) zobrazí pouze 3 položky najednou. Na velkém monitoru (fullHD rozlišení a víc) je tak využitá pouze malá část celkového horizontálního prostoru. Tento problém je zásadní, vzhledem k tomu, že to jediný způsob zobrazení seznamu maleb. Dále jsem narazil nedostatek u komponentu s mapou. Mapa nejde vycentrovat a při každém načtení se vrátí zpět na původní pozici. Poslední problémem na hlavní stránce jsou tlačítka pro vyhledávání, jelikož nevypadají jako prvky, se kterýma by uživatel mohl interagovat.

Hlavní problém stránky s výsledky vyhledávání je ten, že je stejná jako hlavní stránka bez žádného nadpisu nebo indikátoru, že položky ve `Slideru` jsou výsledky vyhledávání. Po načtení stránky se nezobrazuje ani fráze, podle které byli malby vyhledávány.

Detail malby dopadl podle vyhodnocení pozitivně. V přidání příspěvku při posunu malby nelze vycentrovat mapu zpět na jediný pin, který je na mapě a jediný způsob pro jeho přesunutí je jeho přetažení, což je při velkých vzdálenostech frustrující.

2.2.2 Nedostatky uživatelského rozhraní

V uživatelském rozhraní aplikace je několik dalších nedostatků, které popisuji v této části. Z důvodu špatné optimalizace (načítají se veškeré malby včetně miniatur obrázků) trvá načtení zhruba 30 vteřin, což je vysoko nad limitem uživatelské přívětivosti. Mapa, která je nejvýraznější prvek stránky, se při každém načtení přesune na původní místo s původním přiblížením. Na každé stránce jsou odkazy na neexistující obrázky a styly, což způsobuje konzolové oznámení (404 Not Found).

Na hlavní stránce je také vyhledávací formulář, v němž je možné třídit malby v seznamu podle autora, lokace a instituce, ve které se dílo nachází. Po každém odeslání se znovu načte stránka i se všemi malbami na mapě, což vede k dlouhé prodlevě.

Pokud chce uživatel při přidání příspěvku přesunout označenou pozici příspěvku, musí buď tahem myši přesunout základní pin nacházející se uprostřed mapky, která se nachází na poslední lokaci malby (ať je přesná nebo přibližná). V aplikaci není možnost přihlášení a tedy pro nové příspěvky musí uživatel pokaždé zadávat své údaje. Také zde není žádná možnost pro uživatele zobrazit si své přidané příspěvky.

2.2.3 Back-end

Ke zdrojovým kódům jsem se dostal pomocí FTP serveru, přihlašovací údaje mi byly poskytnuty aktuálním správcem administrativní části aplikace a také jsem dostal export stávající databáze ve formátu SQL. Aplikace je uložena na sdíleném hostingu. Back-end aplikace tvoří skripty a šablony v jazyce PHP a data jsou uložená v MySQL databázi.

Back-end aplikace je napsaný velice nepřehledně. Aplikaci chybí objektově orientovaný přístup a použití návrhových vzorů, který zaručí dlouhodobou udržitelnost a snadnou rozšiřitelnost. Nyní je veškerá logika aplikace napsána v jednotlivých skriptech a šablonách bez pevně dané struktury a oddělení jednotlivých vrstev (zobrazovací a datové), například jsou v šablonách přímo SQL dotazy do databáze.

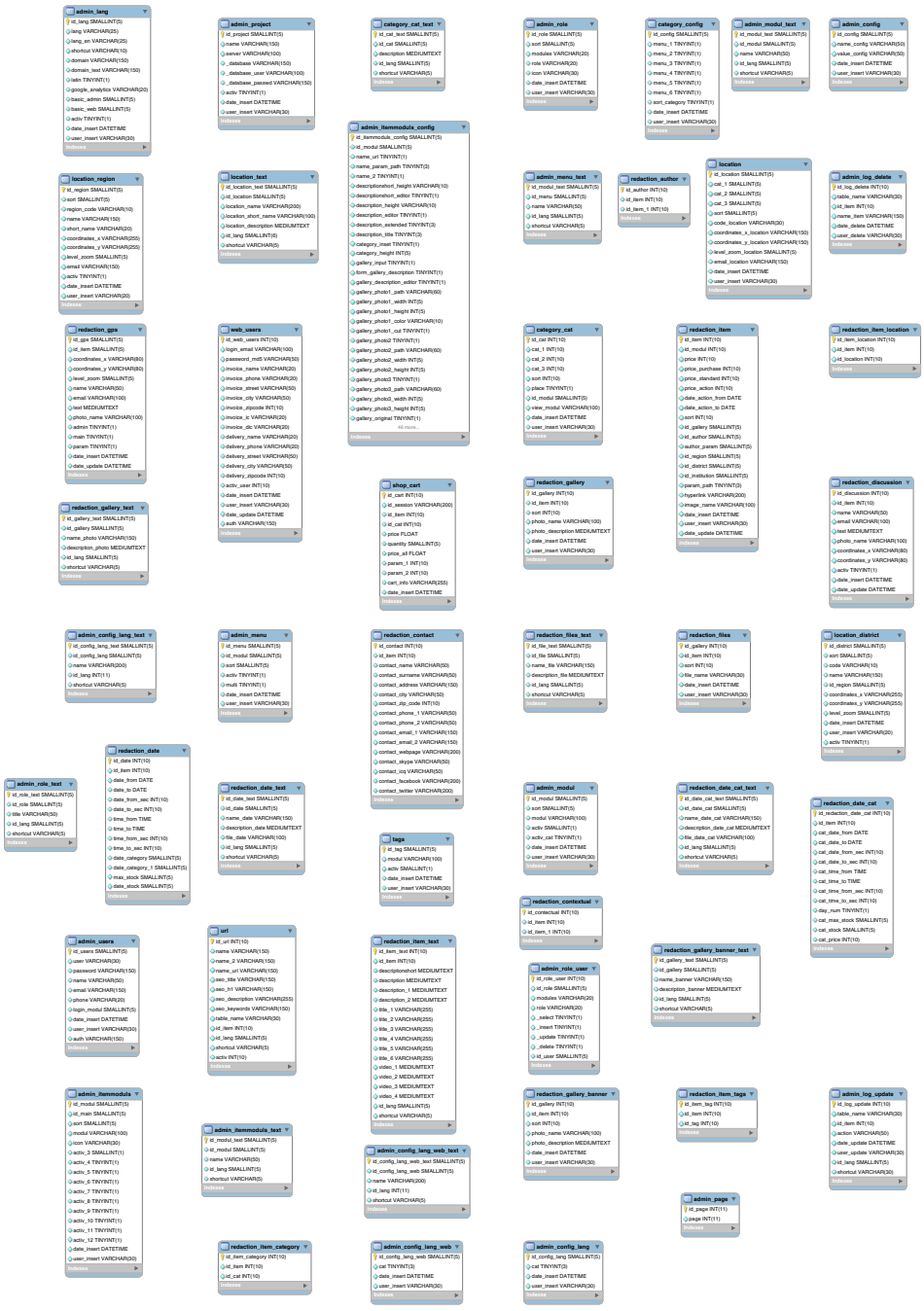
Jak je vidět na obrázku 2.6 (nachází se i v plné velikosti v příloze), samotné databázové schéma je velice nepřehledné. Obsahuje velké množství tabulek, které nemají žádný význam a pravděpodobně byly do schématu zaneseny použitým CMS. Tabulky nejsou pojmenovány sémanticky, takže je obtížné ze samotného schématu poznat, k čemu tabulka slouží. Dále v databázi chybí cizí klíče, takže je v některých případech obtížné odhadnout relaci mezi enti-

tami. PHP skripty v aplikaci nebyly v tomto ohledu příliš nápomocné, jelikož zde není žádné mapování, pouze samotné SQL dotazy.

2.2.4 Vyhodnocení stávající aplikace

Z výše uvedených důvodů vycházejících z problémů současné aplikace jsem se rozhodl vytvořit samostatnou back-endovou aplikaci obsahující API, která bude komunikovat s novým jednodušším databázových schématem, které bude vycházet z doménového modelu, kterému se budu věnovat v další kapitole.

Na tomto back-endu dále bude možné postavit i nový webový front-end, který bude využívat možností API, což je nezbytné pro vytvoření moderní a responzivní aplikace. Data se díky tomu budou načítat dynamicky a to vyřeší jeden z hlavních problémů aktuální aplikace, kterým je dlouhé načítání.



Obrázek 2.6: Vygenerovaný model původní databáze

Kapitola 3

Návrh

V této kapitole se zabývám návrhem Android aplikace a back-endové aplikace s REST API. Pro tvorbu diagramů používám jazyk UML, protože je to používaný standard a byl jsem s ním již seznámen v předmětu BI-SI. Nejprve jsem se zabýval diagramem aktivit, v další části popisuji doménový model a nakonec se věnuji návrhu uživatelského rozhraní Android aplikace.

3.1 Diagram aktivit

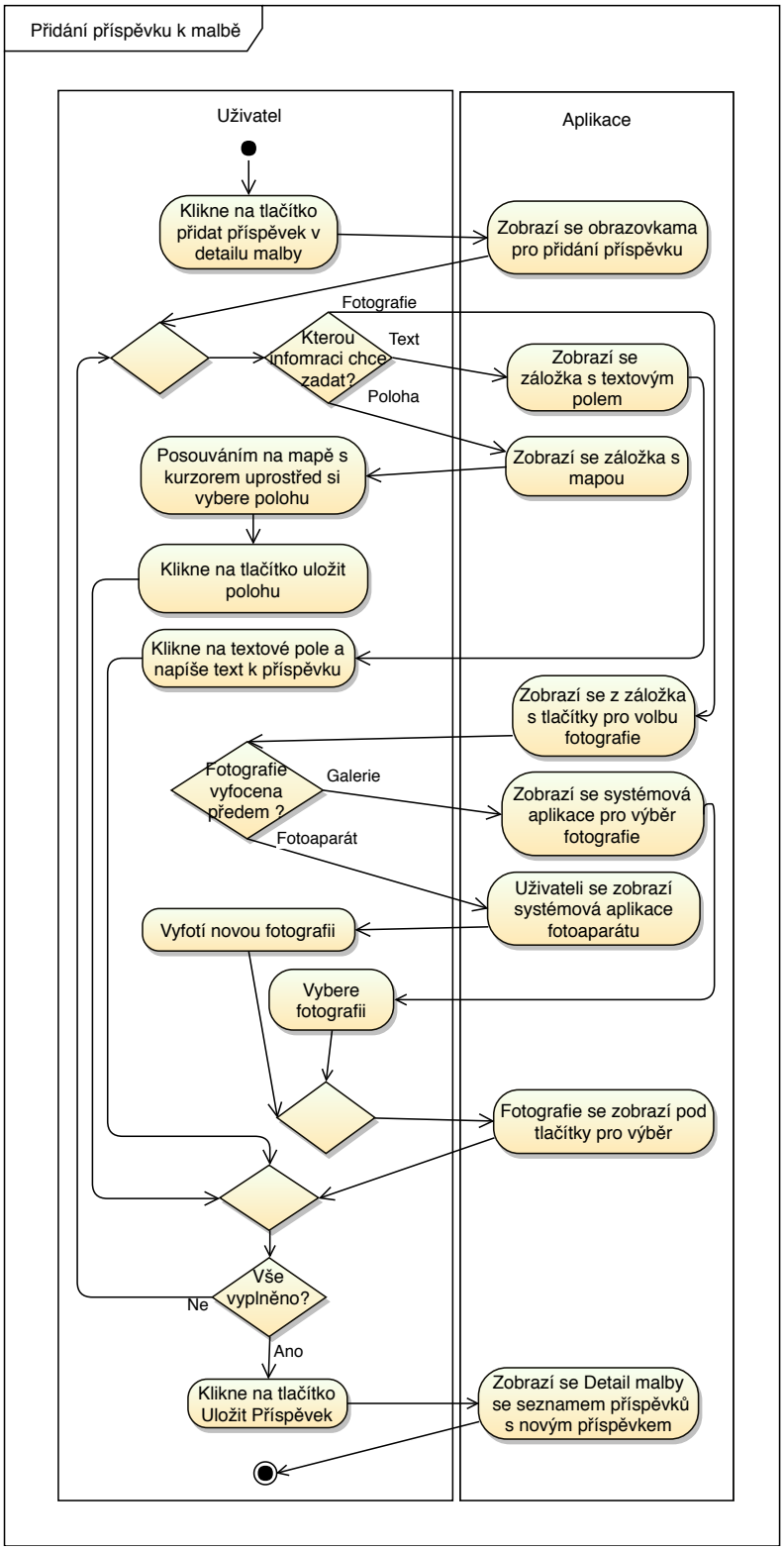
V této kapitole jsem se věnoval diagramu aktivit, který zobrazuje aktivitu uživatele a jeho interakci se systémem. Na obrázcích 3.1, 3.2, 3.3 jsou vyobrazeny nejdůležitější diagramy aktivit - přidání příspěvku k malbě, vyhledání nejbližších maleb a vyhledání informací o malbě.

Nejdůležitější UML diagram pro business modelování je diagram aktivit. Diagramy aktivit jsou často používány ve vývoji softwaru pro zdokumentování průchodu programu například pro ukázání algoritmu použitého k implementaci specifické operace. Diagram aktivit má širokou škálu využití, mohou ukazovat aktivity (paralelní či sekvenční), objekty konzumovány, používány či produkovány aktivitami, zodpovědnost za aktivity a vztahy a závislosti mezi aktivitami. [5]

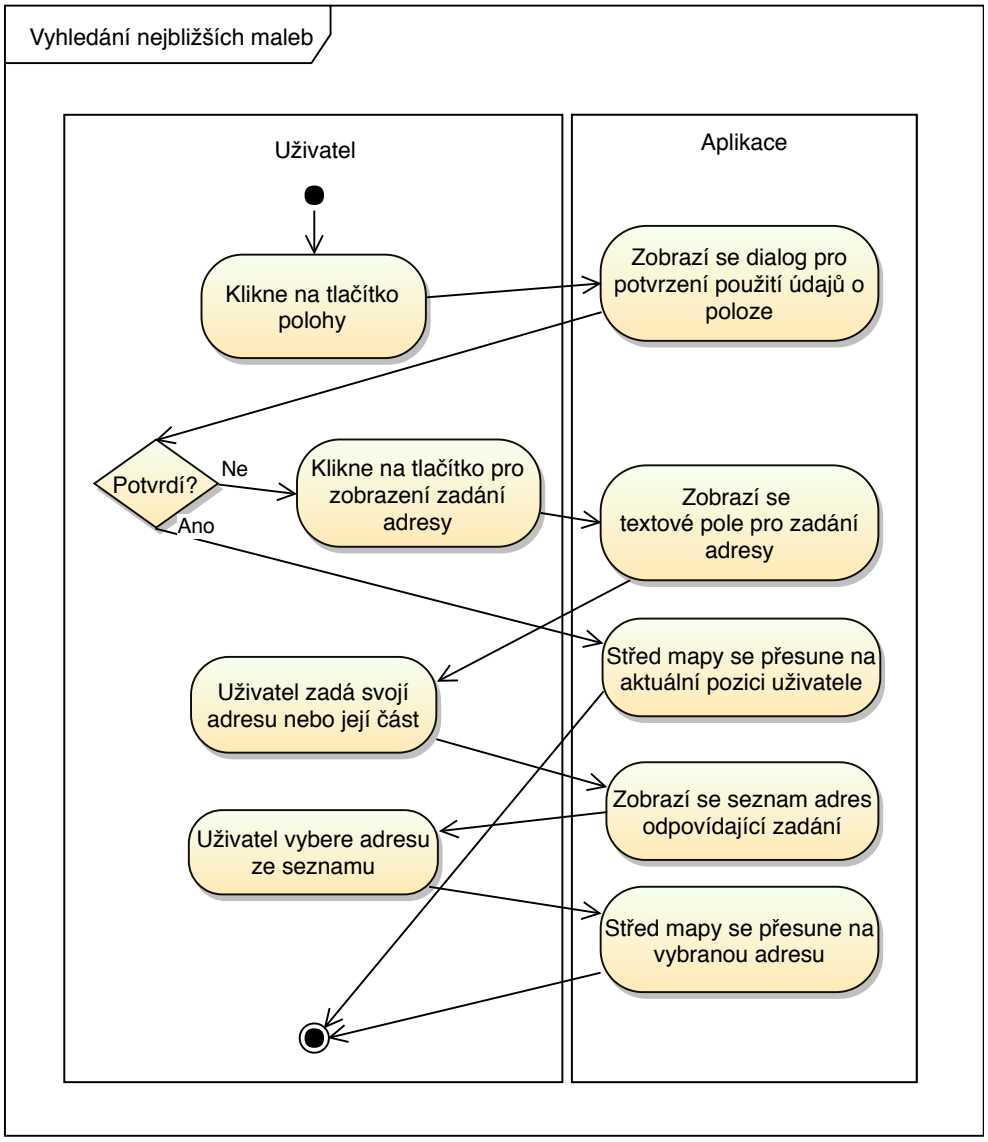
3.2 Doménový model

V této části jsem se věnoval doménovému modelu, který lze najít na obrázku 3.4. Doménový model zobrazuje základní náčrt struktury systému. Obsahuje entity, které reprezentují objekty z reálného světa, ty mají své atributy. Model také vyobrazuje vztah mezi jednotlivými entitami. Z doménového modelu dále vychází návrhový model, který jej rozšiřuje o řadu vlastností.[6]

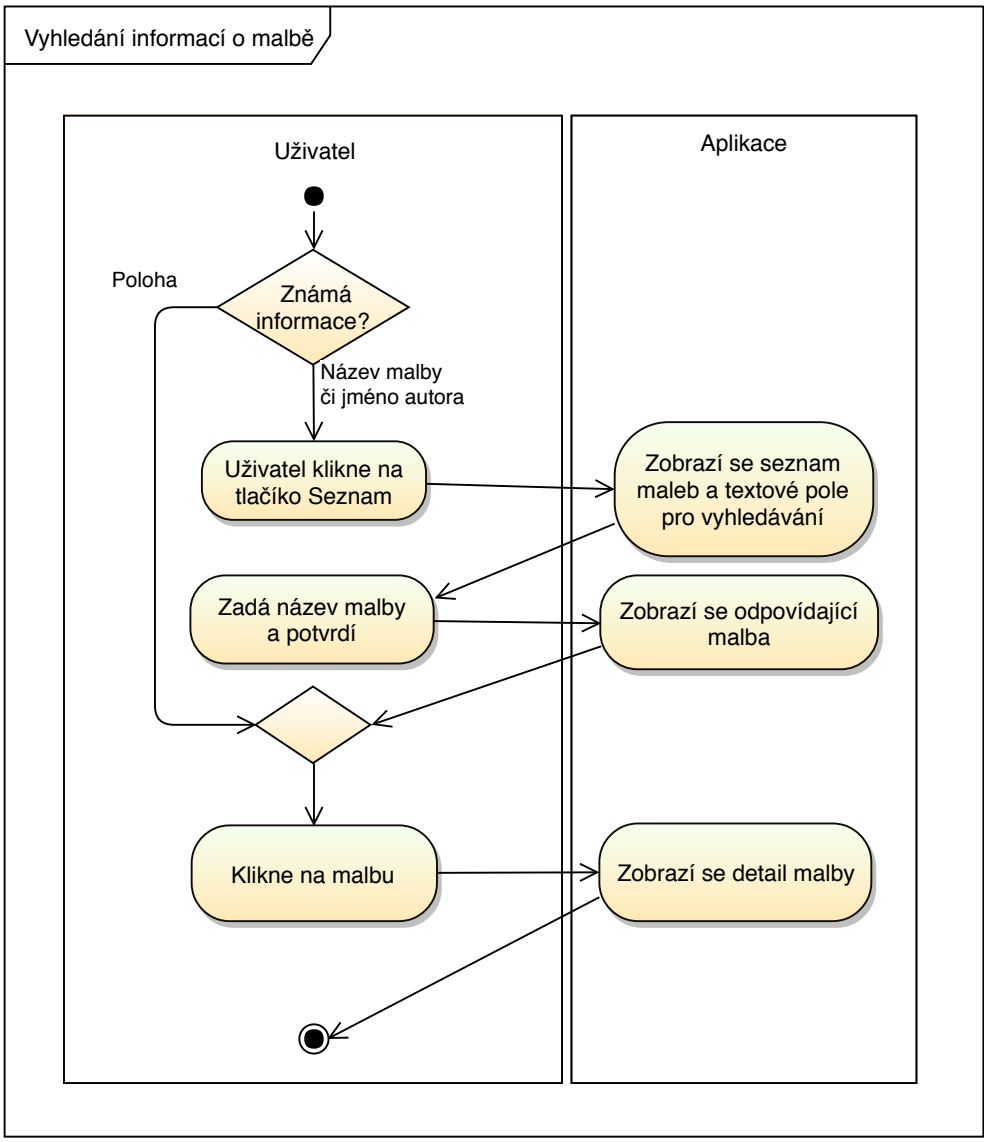
Model je zjednodušená reprezentace (nebo abstraktní popis) části systému, která je pojmenovávána podle reálného světa. Model slouží k lepšímu poro-



Obrázek 3.1: Diagram aktivit - přidání příspěvku k malbě



Obrázek 3.2: Diagram aktivit - vyhledání nejbližších maleb



Obrázek 3.3: Diagram aktivit - vyhledání informací o malbě

zumění systému. V kontextu inženýrství model pomáhá rozhodovat při volbě akcí, které jsou nezbytné k dosažení a udržení cíle systému. Cílem softwaru je automatizovat úkoly z reálného světa. Modely požadavků, struktury a chování na různých úrovních abstrakce pomáhají účastníkům rozhodovat, jak se má cíle dosáhnout a udržet.[7]

Na základě této definice, zdrojový kód je také model, protože je to zjednodušená verze strojových struktur a operací, které jsou nezbytné k automatizaci úkolů v reálném světě. Dokonce by se dalo tvrdit, že správný zdrojový kód je velmi užitečný model, protože říká zařízení, jaké akce mají být vykonány pro splnění cíle systému.[7]

3.2.1 Malba

Malba (Painting) reprezentuje existující krajinomalbu. Informaci o přesnosti polohy určuje atribut resolved. Malba může mít více příspěvků (Post). Malba také může mít instituci (Institution), ve které se nachází a autora (Author), který malbu vytvořil. Pokud ho nemá, znamená to, že je autor neznámý. Malba také může mít libovolný počet štítků (Tag).

3.2.2 Příspěvek

Příspěvek může přidat uživatel (User) k malbě (Painting) v Android aplikaci (musí být přihlášen).

3.2.3 Autor

Autor (Author) je entita, která popisuje výtvarníka, který tvoří nebo tvořil malby. Autor může mít více maleb (Painting). Autora lze vytvořit pouze ve webové administraci.

3.2.4 Instituce

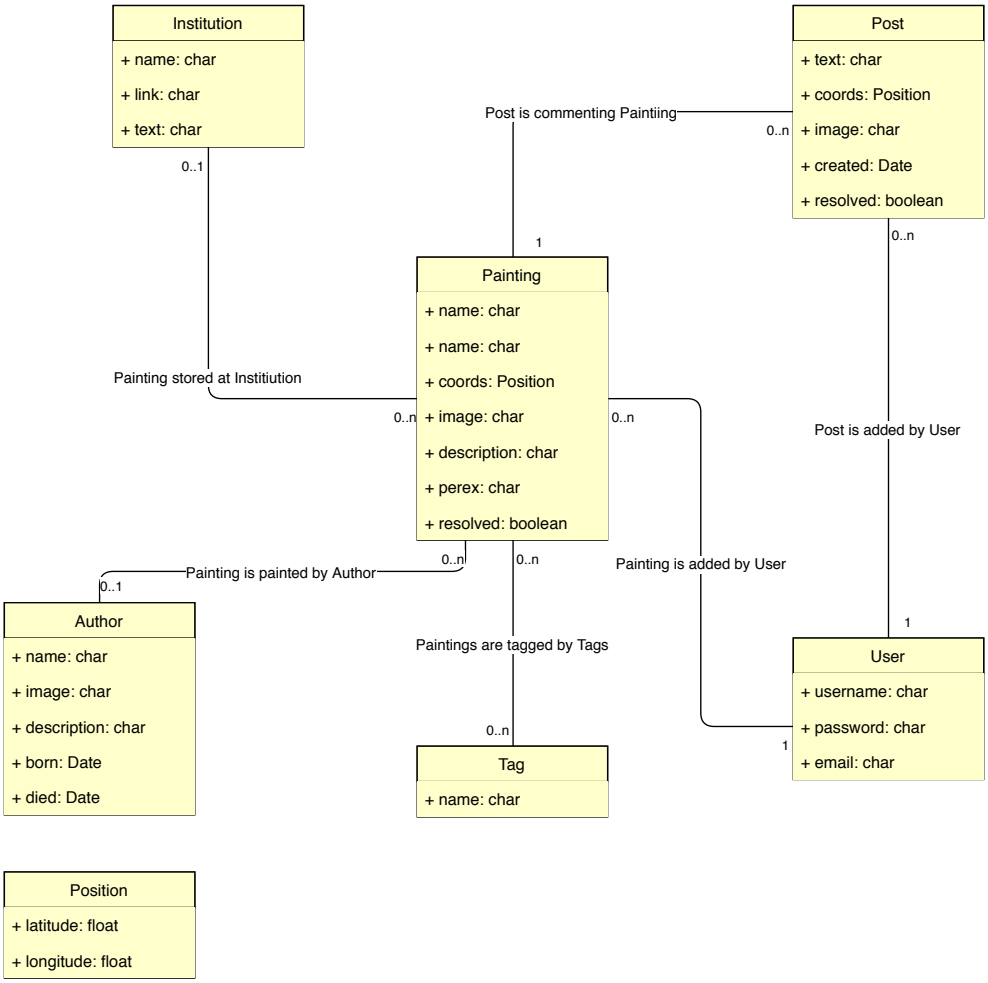
Instituce (Institution) je entita, která reprezentuje instituci, ve které se mohou nacházet malby. Instituce může mít více maleb (Painting).

3.2.5 Uživatel

Uživatel je entita, která může přidávat příspěvky (Post) a malby (Painting). Bez uživatelského účtu není v aplikaci možné přidat příspěvek. V rámci aplikace jej lze vytvořit (registrace). Slouží také k autentizaci a autorizaci.

3.2.6 Tag

Tag je entita, popisují štítek malby a reprezentuje vlastnost či informaci, kterou mohou mít malby společné. Tag může mít libovolný počet maleb (Painting).



Obrázek 3.4: Doménový model

3.3 Návrh uživatelského rozhraní

V této sekci se nachází návrh uživatelského rozhraní doprovázený jeho náčrtem na obrázku 3.5. Dále jsem přiložil export návrhu obrazovek z aplikace Justinmind na obrázcích 3.6 a 3.7.

3.3.1 Návrh obrazovek

V této části se nachází seznam obrazovek Android aplikace s popisem obsažených UI prvků a jejich chování.

3.3.1.1 Hlavní obrazovka

Na hlavní obrazovce bude především mapa, na které budou viditelné piny označující malby, piny budou barevně rozlišovat nalezené a nenalezené malby. V rozích mapy budou menší tlačítka pro ovládání mapy stejně jako v Google Maps (zoom, aktuální poloha). Ve spodní části bude navigační panel s tlačítky dalších obrazovek - seznam maleb, mapa a profil. Aktuální obrazovka bude mít zvýrazněné tlačítko v panelu. Po kliknutí na pin se zobrazí ve spodní části panel s názvem malby, po kliknutí přenese uživatele na detail malby. V horní listě bude tlačítko pro zobrazení textového pole pro vyhledávání adresy. Do něj bude možné zadat adresu, výsledky se zobrazí v seznamu pod textovým polem, kliknutím na ně se mapa přenese na danou adresu a tím se zároveň schová vyhledávací pole a seznam výsledků.

3.3.1.2 Detail malby

Detail malby bude rozdělen do záložek. V první budou informace o malbě a fotografie. Také zde bude jméno autora, pokud je uveden, název instituce, pokud je uveden a informace o tom, zda již byla malba nalezena. Ve druhé bude mapa s malbou a příspěvky a ve třetí se bude nacházet seznam příspěvků s fotografií, textem, datem přidání a jménem uživatele. Po kliknutí na příspěvek se zobrazí detail jeho fotografie. V dolní levém rohu bude FAB tlačítko, které přesune uživatele do obrazovky pro přidání příspěvku.

3.3.1.3 Přidat příspěvek k malbě

V této obrazovce budou také tři záložky. V první bude textové pole pro přidání textu, druhá bude obsahovat mapu pro určení polohy příspěvku. Výběr polohy se bude provádět tak, že uprostřed mapy bude kurzor určující polohu pro zvolení a gestem posouvání bude uživatel hýbat s mapou a tedy i místem, na které kurzor ukazuje. Ve spodní části bude tlačítko `Uložit pozici`. Ve třetí pro pořízení fotografie nebo výběr z galerie. Ve spodní části pod záložkami se bude nacházet tlačítko `Přidat příspěvek`. Pokud uživatel nebude přihlášen při přechodu na tuto obrazovku, zobrazí se dialog, ze kterého se může vrátit zpět nebo přenést na obrazovku pro přihlášení a registraci.

3.3.1.4 Seznam maleb

Seznam, na který se dostane uživatel z úvodní obrazovky pomocí navigačního panelu bude obsahovat pole pro vyhledávání a seznam maleb. V horním panelu bude tlačítko s ikonkou filtru, které přenese uživatele obrazovku s výběrem filtrování seznamu. Seznam bude obsahovat miniatury malby (odpovídající vyhledávání a filtru) s názvem a obrázkem, kliknutím na ně se uživateli zobrazí detail malby.

3.3.1.5 Filtr seznamu

Filtr bude jednoduchá obrazovka obsahující několik prvků pro specifikaci filtru. Bude zde několik UI prvků, pro výběr ze seznamu (výběr instituce, stavu hledání malby). Dále bude tato obrazovka obsahovat vyhledávání adresy, v jejímž okruhu by se měli vyhledané malby nacházet a tlačítko pro resetování filtru.

3.3.1.6 Profil

Profil bude obsahovat jméno a email uživatele a pod ním seznam příspěvků, které uživatel přidal. Po kliknutí na příspěvek se zobrazí detail malby, pod kterou byl příspěvek přidán. Pokud uživatel nebude přihlášen při přechodu na obrazovku, zobrazí se dialog s tlačítky `Zpět` a `Přihlásit`.

3.3.1.7 Přihlášení a registrace

Obrazovka pro přihlášení bude jednoduchá, bude obsahovat pouze pole pro zadání emailu a hesla. Dále zde bude tlačítko `Přihlásit` a tlačítko `Registrace`. Po kliknutí na tlačítko `Registrace` se zobrazí ještě pole pro zadání jména a pro zopakování hesla, místo tlačítka `Přihlásit` se objeví tlačítko `Registrovat`. Pokud přihlášení nebo registrace proběhne úspěšně, objeví se oznámení a uživatel bude přesměrován na původní obrazovku, v opačném případě se zobrazí oznámení o chybě a zároveň se daná chyba vypíše.

3.3.2 Seznam úkolů

V této části popisuji, jak bude uživatel postupovat v průchodu obrazovkami pro splnění daných cílů.

3.3.2.1 Zobrazení detailu malby a přidaných příspěvků

Kliknutím na pin na hlavní obrazovce si uživatel vybere malbu. Tím se zobrazí spodní panel s názvem malby.

Po kliknutí na spodní panel s vybranou malbou přejde uživatel na její detail. Zde je obsah rozdělen do 3 záložek, mezi kterými je možné přecházet klikuntím na tlačítka ve spodním panelu a Swipe gestem (rychlé posunutí prstem po obrazovce).

Zobrazit detail je také možné kliknutím na položku v Seznamu maleb.

3.3.2.2 Přidání příspěvku k malbě

V detailu malby uživatel klikne na tlačítko `Přidat příspěvek`, které ho přenese na obrazovku přidání příspěvku. Pokud není přihlášen, zobrazí se informační dialog, který může uživatele přenést na přihlašovací obrazovku nebo zpět na detail.

Obrazovka pro přidání příspěvku ta bude rozdělena do tří záložek. V první bude možnost zadat text k novému příspěvku, v druhé bude mapa s možností připíchnout nový pin a tím označit lokalitu nového příspěvku a ve třetím se budou nacházet dvě tlačítka. První bude sloužit k výběru fotografie z galerie, které po kliknutí spustí systémovou aplikaci pro výběr z galerie, a druhé bude sloužit k použití fotoaparátu telefonu, které po kliknutí spustí aplikaci Fotoaparát. Ve spodní části bude tlačítko `Uložit příspěvek`, který odešle nový příspěvek. Pokud přidání dopadne úspěšně, přenese aplikace uživatele zpět na detail malby.

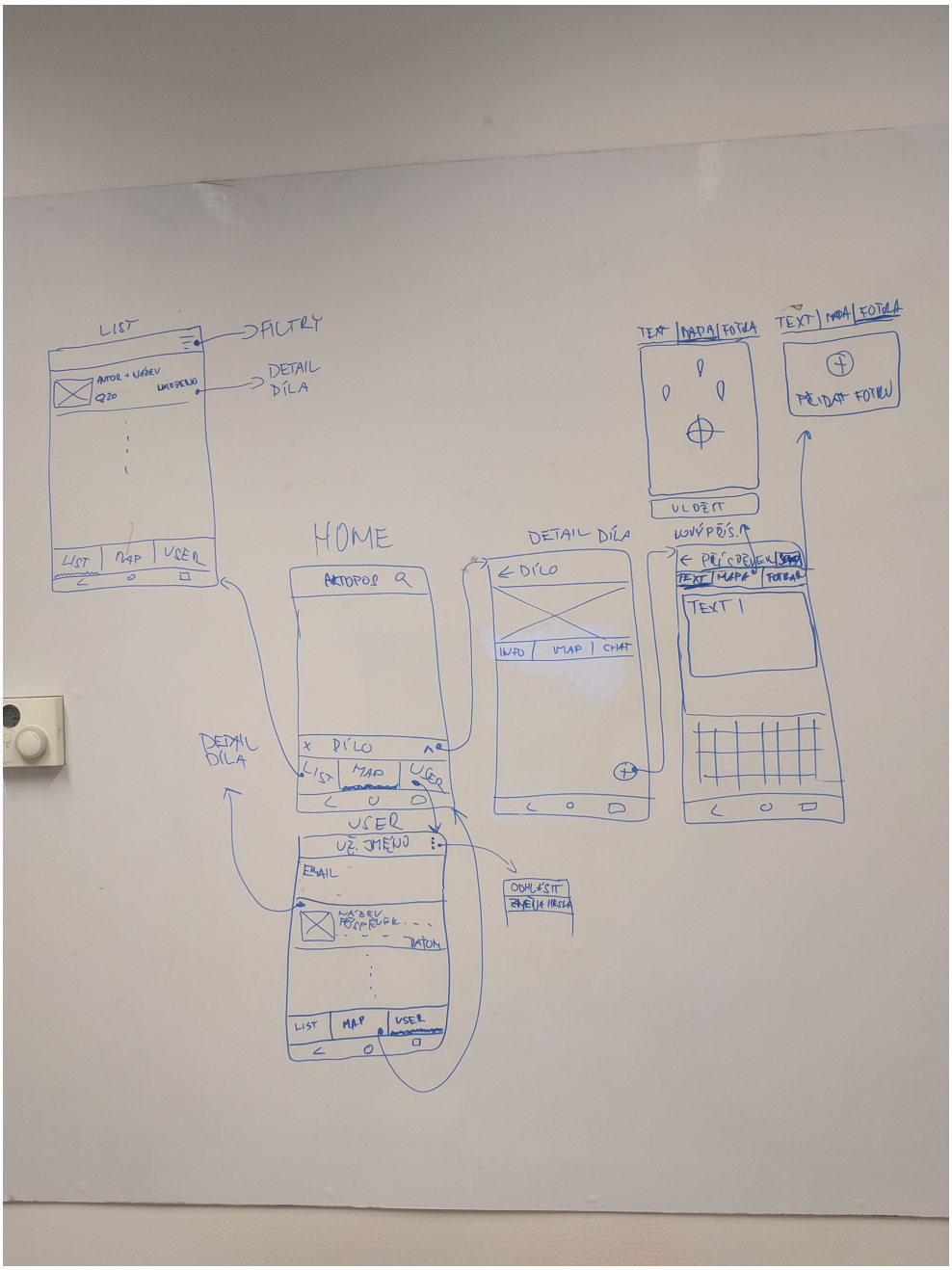
3.3.2.3 Zobrazování maleb podle filtru

Po vybrání levého tabu se uživatel dostane z hlavní obrazovky na seznam maleb. Zde je v horní části textové pole pro vyhledávání, do něj může zadat jméno nebo část jména díla nebo autora. V horní liště je na pravé straně tlačítko pro zobrazení obrazovky s možnostmi filtrování. Zde bude několik aktivních prvků pro zadávání filtrování. Tlačítkem zpět se vrátí na předchozí obrazovku, kde se zobrazí malby, které odpovídají zadaným filtrům.

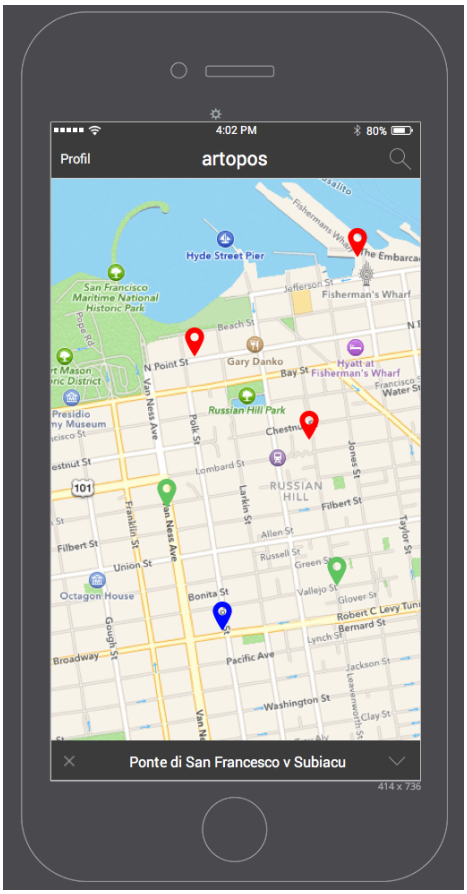
3.3.3 Design

Design aplikace vychází z Material Design pravidel, které oficiálně Google doporučuje. Dodržování těchto pravidel usnadní uživateli orientaci v aplikaci a designérovi návrh jejího uživatelského rozhraní a přesto dává velký prostor úpravám a modifikacím tak, aby všechny aplikace nevypadaly stejně. (odkaz na dokumentaci Material Design [8]) Barevné schéma a logo jsem ponechal původní z webové aplikace aplikace.

Uživatelé Androidu očekávají od aplikací vzhled a chování, které je konzistentní se vzhledem a chováním platformy. Nejen že by měl vývojář dodržovat Material Design pravidla pro vizuální a navigační vzory, ale také by měl dodržovat pravidla kvality, kompatibility, výkonu a zabezpečení. [9]



Obrázek 3.5: Návrh obrazovek



Obrázek 3.6: Návrh obrazovky - mapa

Obrázek 3.7: Návrh obrazovky - detail

Kapitola 4

Výběr technologií pro vývoj

V této kapitole rozebírám možnosti pro vývoj Android aplikace a stručně popisuji některé nejznámější knihovny či frameworky a také zde přibližuji technologie použité pro implementaci REST API.

4.1 Vývoj android aplikace

Android je sice nejrozšířenější operační systém na světě, ale při vývoji aplikace je třeba myslet na fakt, že existují i další platformy, jejichž uživatelská základna není zanedbatelná - aktuálně zejména iOS. Jelikož jsou nástroje pro vývoj na iOS platformu odlišné od těch na OS Android, multiplatformní aplikace by teoreticky zabrala dvakrát tolik času. Navíc se tím komplikuje další rozvoj aplikace, jelikož veškeré změny musí být prováděny dvakrát a verze aplikací musí odpovídat.

Kvůli těmto problémům vznikly frameworky a knihovny, které se snaží vývojáře od těchto problémů alespoň částečně odstínit. Existuje jich celá řada, každý má svoje výhody a nevýhody, které se v následujících odstavcích pokusím přiblížit.

4.1.1 Xamarin

Tento framework je jedním prvních a nejpoužívanějších nástrojů pro multiplatformní vývoj aplikací. Využívá výhod programovacího jazyka C# a pro vývojáře je atraktivní díky jednotnému zdrojovému kódu logiky aplikace. Pro interpretaci využívá Mono, což je open-source implementace .NET frameworku od Microsoftu. [10]

4.1.2 Frameworky používající WebView

Takovéto frameworky, jako je například Cordova, dříve existovaly jako alternativa k nativním aplikacím. Jejich výhoda byla rychlý vývoj aplikace bez

nutné znalosti nativních technologií. Problémy u těchto frameworků nastávaly v otázce výkonu. Vývojář takovéto aplikace musel od začátku vývoje dbát na složitost a efektivitu. Animace v těchto aplikacích byly stěží schopné běžet na 60 snímků za sekundu, což je spodní limit, kdy animace ještě působí plynule pro lidské oko. To je zásadní problém v době, kdy se u aplikací tolik hledí na grafický design a uživatelské rozhraní.

Ačkoliv použití WebView v kombinaci s nativními prvky ukazuje lepší výsledky co se týče výkonu a běhu aplikace kvůli náročným interakcím s DOM ve WebView, byla vyvinuta řešení používající hybridní technologii, ale bez použití WebView. Tato řešení se nazývají nativní Javascriptové frameworky. [11]

4.1.3 Nativní javascriptové aplikace

Tyto frameworky jsou dobrou alternativou k nativnímu vývoji, protože nabízí dobrý výkon a bezproblémový běh aplikace. Zároveň podporují multiplatformní vývoj a nabízí výhody javascriptu. Zde jsem vybral dva nejpopulárnější z této kategorie.

4.1.3.1 React Native

React Native je z této kategorie nejpoužívanější. To je pravděpodobně způsobeno tím, že byl mezi pionýry této technologie a zároveň React je nejpopulárnější javascriptová knihovna pro webové aplikace, za jejímž vývojem stojí Facebook a kterou podporuje velká komunita. React Native podobně jako Xamarin funguje na principu jednotného kódu pro logiku na všech platformách, ale UI aplikace se musí implementovat pro každou platformu zvlášť.

React Native využívá virtuální stroj JavaScriptCore, který je původně použitý pro WebKit (renderovací jádro pro prohlížeče). [12] React Native má díky podpoře komunity velké množství pluginů a doplňujících knihoven a dobrou a obsáhlou dokumentaci plnou příkladů, což velice usnadňuje vývojářům život. React Native používají například aplikace Facebook, Instagram nebo AirBnB. [13]

4.1.3.2 NativeScript

Dalším podobným frameworkem je NativeScript. Tento framework byl vydán přibližně ve stejné době jako React Native a rychle si získal oblibu mezi vývojáři. Proto má také velkou komunitu, která poskytuje řadu doplňků, knihoven a demo aplikací. Výhodou toho frameworku je podpora knihovny Angular (a nově i Vue), které patří mezi nejpopulárnější mezi vývojáři.

Také podporuje nadstavbu nad javascriptem Typescript, který dělá kód přehlednější a čitelnější. Jedna z hlavních předností je jednotný kód pro Android a iOS aplikace a to samé platí pro většinu pluginů. Také je vývojář odstíněn od nativních prvků aplikace, ale díky NativeScript API se jednoduše

dostane ke konkrétním funkcím. Výhodou je i NativeScript Marketplace, což je seznam doplňků s popisem a hodnocením.

Je vyvíjen společností Telerik. Na rozdíl od React Native běží na javascriptovém virtuálním stroji Chrome V8 napsaným v jazyce C++ a vyvíjeným společností Google. Mimo jiné je Chrome V8 také použitý v prohlížeči Google Chrome nebo v serverovém Node.js. [14]

4.1.4 Výběr frameworku

Pro tuto práci jsem si vybral framework NativeScript, protože mě z výše jmenovaných zaujal nejvíce. Vyžaduje minimum zásahů do zdrojového kódu pro multiplatformní vývoj a díky využití správce balíčků NPM je jednoduché instalovaní pluginů, které řeší například použití Google Maps, kamery, galerie, GPS souřadnic a dalších funkcí. Také podporuje úpravu stylů nativních komponent pomocí webového jazyka CSS a vývoj ulehčuje možnost inkrementálního sestavení, které je rychlejší než klasické sestavení aplikace.

Tento framework preferuji i proto, že mám zkušenosti s jazykem javascript a frameworkem Anuglar. Navíc díky dobře napsané dokumentaci, velkému množství demo aplikací a silné komunitě se vývojář lehce dopátrá řešení konkrétního problému.

4.2 Vývoj back-endové části aplikace s API

Po analýze stávající aplikace jsem se rozhodl implementovat API jako samostatnou aplikaci včetně nového databázové modelu. V této sekci popisuji, jaké technologie jsem zvolil a proč.

4.2.1 REST API

Pro komunikaci mezi Android aplikací a back-endovou částí jsem se rozhodl použít REST API, jelikož je to nejpoužívanější a relativně jednoduchý standard, který plně vyhovuje potřebám server-klient aplikací. Využívá http protokol a JSON jako formát dat. Alternativou je formát XML, ale s ohledem na typ aplikace je JSON vhodnější volba.

Alternativou by mohlo být například GraphQL, které je ovšem složitější na implementaci než REST a je vhodnější pro podmíněné a parametrizované dotazy, kterých je v této aplikaci minimum.

Webové služby jsou webové servery vytvořené pro konkrétní účel, které podporují potřeby webové stránky nebo jakékoliv jiné aplikace. Klientské programy používají API pro komunikaci s webovými službami. Obecně řečeno, API vystavuje data a funkce a tím umožňuje interakci mezi programy a také výměnu informací. Webové API je „tvář“ webové služby, která přímo naslouchá a odpovídá na požadavky klienta.

REST architektura je dnes běžně aplikovaná v API designu pro moderní webové služby. Webová služba s REST API se nazývá RESTful. REST API se skládá z navzájem propojených zdrojů (resource). Tyto zdroje tvoří model (resource model).

Dobře navržené REST API může přilákat vývojáře, kteří budou chtít používat dané REST API. V dnešní době volného trhu, kdy webové služby soupeří o pozornost vývojářů, je dobře navržené REST API nezbytná vlastnost. [15]

4.2.2 OAuth

Oauth2 je nejrozšířenějším standardem co se týká zabezpečení API, využívá tokeny pro autentizaci a autorizaci. Často se využívá pro přihlašování pomocí třetích stran. Existuje velké množství dokumentace a návodů, jak implementovat klientskou i serverovou stranu a také existuje řada knihoven, které implementaci tohoto protokolu usnadňují.

Nejprve se uživatel musí autentizovat pomocí emailu a hesla, id klienta a tajného klíče klienta(`client_id` a `client_secret`). Pokud vše proběhne v pořádku server vrátí klientovi JSON objekt, který obsahuje `access_token` , `refresh_token` a dobu v sekundách, za jakou `access_token` expiruje. Po uplynutí této doby si musí uživatel obnovit `access_token` pomocí `refresh_tokenu` (dotaz vrátí nový JSON objekt s tokeny a dobou expirace).

4.2.3 Symfony a Composer

Pro implementaci back-endové aplikace s REST API jsem se rozhodl použít Symfony 4. Symfony je PHP framework vyvíjen společností SensioLabs, která zároveň stojí za správcem knihoven Composer. Symfony se díky dobré struktuře projektu a snadné rozšiřitelnosti jedním z nejpoužívanějších webových frameworků a díky tomu, že je open source, z něj vychází řada dalších knihoven a frameworků (například Drupal nebo Laravel).

Nyní je k dispozici verze 4, která řeší řadu problémů a nedostatků minulých verzí. SensioLabs tvrdí, že je Symfony 4 „micro by default“ a že základní projekt má o 70% méně kódu než Symfony 3 a také je o poznání rychlejší. Nově obsahuje automatizaci instalace a nastavení balíčků a jednodušší správu jejich závislostí (Symfony Flex). To ve výsledku funguje tak, že po instalaci pomocí Composeru se automaticky zaregistruje balíček (a po odstranění se také automaticky vymaže), dále má nyní každý balíček vlastní konfigurační soubory nezávislé na těch ostatních. [16]

Kapitola 5

Implementace

V této kapitole popisuji postup při implementaci. Dále zde popisuji základní informace o vývoji Android aplikace a back-endové části, jako je struktura projektu a použité knihovny.

5.1 Implementace Android aplikace

Vývoj aplikace v NativeScriptu má řadu výhod ve srovnání s nativním Android vývojem. Vývojové prostředí není vázané na Android Studio, ale vývojář může použít téměř libovolné IDE na téměr libovolném operačním systému. V případě vývoje iOS aplikace je nutný operační systém Mac OS X. Pro vytvoření NativeScript projektu jsou nezbytné Android SDK a JVM a NPM.

Instalace NativeScriptu je poměrné náročná, protože má NativeScript řadu závislostí pro sestavování aplikací a vytváření projektů. Proto je nejlepší použít nástroj k instalaci (ke stažení na oficiálních stránkách NativeScriptu), který se o všechny závislosti postará. Instalování pluginů probíhá pomocí NPM (Node Package Manager), což je nepoužívější správce javascriptových knihoven, a většina z nich začíná prefixem `nativescript`.

Build aplikace se spouští pomocí příkazu `tns run Android`, který pustí kompilaci do Android projektu a `Watcher`, který sleduje změny a při každé provede inkrementální sestavení a nainstalování aplikace. Pokud je k připojeno k počítači Android zařízení, které má zapnuté USB ladění a povolené aplikace z neznámých zdrojů, aplikace se nainstaluje a spustí na tomto zařízení. V opačném případě se spustí Android emulátor, pokud je k dispozici.

5.1.1 Knihovny použíté v aplikaci

V této sekci popisuji knihovny a doplňky použité v aplikaci, které značně zjednodušily vývoj aplikace. Všechny tyto balíčky byly nainstalované prostřednictvím správce balíčků NPM.

5.1.1.1 Angular

Angular je jeden ze známějších javascriptových frameworků vyvíjen společností Google, známý svojí univerzálností a škálovatelností. Dnes je vydaná 5 verze toho frameworku a hlavní jádro frameworku doprovází další podpůrné knihovny např. `angular-router`, `angular-http` nebo `angular-forms`. V této aplikaci používám verzi 4.4.

Základní třídou v Angularu je Modul, který zaštiťuje komponenty a služby, které se musí registrovat při jeho inicializaci kvůli Dependency Injection. Komponenty mohout mít vlastní styl (soubor ve formátu CSS) a HTML šablonu pro vytváření jejich vizuální podoby. Ty obsahují řadu direktiv, které usnadňují práci (například ngFor pro vykreslení pole nebo atribut v závorkách značící data-binding proměnné). Do komponentů a služeb lze přidávat pomocí Dependency Injection služby (jejichž třída musí být označená dekorátor `@Injectable`).

V Angularu se často pracuje s návrhovým vzorem `Promise`. Promise je objekt, který slouží jako zástupce pro nějakou hodnotu. Tato hodnota je typicky výsledek asynchronní operace, jako je HTTP požadavek, nebo čtení ze souboru z disku. Zavolání asynchronní operace je okamžitě vrácen objekt promise, který slouží k registraci zpětných volání (callback), které se provedou při dokončení asynchronní operace. [17]

5.1.1.2 NativeScript doplňky

NativeScript poskytuje řadu doplňků také velké komunitě existuje mnoho doplňků třetích stran. Všechny tyto doplňky jsou dostupné prostřednictvím správce balíčků NPM. NativeScript má také Marketplace, což je stránka se seznamem doplňků s popisem a hodnocením. Zde se nachází ty, které jsem použil v aplikaci.

nativescript-sqlite Zaštiťuje a usnadňuje práci s nativní SQLite databází.

nativescript-google-maps-sdk Umožňuje přidat MapView a funkcionalitu s tím spojenou.

nativescript-geolocation Poskytuje GPS souřadnice a také oprávnění aplikace.

nativescript-google-places-autocomplete Využívá javascriptové google places api pro vyhledání informací z adresy.

nativescript-imagepicker Umožňuje jednoduše vybrat obrázek z galerie.

nativescript-camera Umožňuje využívat nativní aplikaci pro fotoaparát.

nativescript-bottombar Navigační panel ve spodní části obrazovky.

nativescript-floatingactionbutton Kulaté tlačítko ve pravém dolním rohu z Material Design pravidel.

nativescript-image-swipe Prohlížeč obrázků s možností zvětšování a podporou gest.

nativescript-securedstorage Přístup k šifrovanému úložišti pro ukládání klíčů nebo hesel.

5.1.2 Výzvy při implementaci

Několik problémů nastalo při implementaci komponentů obrazovek, kdy se vnořené prvky pro záložky (`TabView`) a pro rolování (`ScrollView`) chovaly neresponzivně a neintuitivně. Tyto problémy jsem musel vyřešit úpravou rozvržení prvků na obrazovce. V jednom případě (seznam výsledků ve vyhledávací komponentě) jsem byl nucen prvek pro vykreslování seznamu (`ListView`) změnit na Angular directivu `*ngFor` (slouží pro zobrazení seznamů prvků, ale je pomalejší než nativní `ListView`), protože při změně prvků `ListView` nevykreslovalo prvky správně kvůli omezenému prostoru na obrazovce.

Při situaci, kdy Event Listener poslouchal Event, který se volal příliš často, aplikace se dostávala do situací, kdy se posílalo velké množství požadavků. Proto jsem v několika případech použil funkci `debounce`, která nastavuje maximalní počet zavolání callbacku v určitém časovém intervalu. Tuto funkci jsem použil při změně textu ve vyhledávacím poli a také při změně kamery v mapě.

5.2 Implementace webového back-endu

V této sekci popisuji postup při implementaci back-endové aplikace. Pro implementaci back-endové části jsem použil PHP framework Symfony a správce balíčků Composer pro instalaci dalších knihoven.

Symfony aplikace se může rozdělit do balíčků (Bundle), čímž se mohou oddělit části aplikace, které spolu tolik nesouvisí a díky tomu se projekt stává mnohem přehlednějším. V této aplikaci jsem používal pouze balíčky třetích stran. Symfony pracuje s návrhovým vzorem MVC a používá Dependency Injection. Zároveň má funkci `autowire`, která z hlavní složky automaticky načítá třídy podle toho kam patří a k nim závislosti na služby bez nutnosti jakékoliv konfigurace. Například všechny třídy v podsložce Controller, které dědí třídu Controller, jsou načteny jako služby a jejich metody použité jako akce pro odbavování požadavků. Díky tomu je v Symfony projektech jasně daná struktura a nutí programátora psát přehledný a dobře čitelný zdrojový kód.[16].

Databázový model vychází z doménového a rozšiřuje ho o některé atributy, například obrázky v databázi jsou reprezentovány pomocí `public_id` cloudinary obrázku, url adresy původního obrázku a url adresy miniatury. Schéma

jsem vytvořil pomocí ORM anotací ve frameworku Doctrine (dokumentace viz [18]). Pro práci s Entitami a databází se používají repozitáře, což jsou třídy, které dědí od třídy `Repository` a každá Entita má svou vlastní.

V této práci nebylo prioritou vytvoření webové aplikace a administrace, nicméně Symfony je dobrý základ vytvoření webové aplikace i administrace díky snadné rozšiřitelnosti, například obsahuje šablonovací framework Twig a Security balíček, který řeší správu opravnění a přístupů aplikace. Implementace webové aplikace bude usnadněná také díky možnosti využití části zdrojových kódu z Android aplikace díky NativeScriptu a Angularu.

5.2.1 Použité knihovny a služby

Zde popisuji použité knihovny či služby, které mi usnadnily vývoj aplikace. Všechny knihovny jsem instaloval pomocí nástroje Composer, což mi zjednodušilo jejich správu a instalaci. V kořenové složce projektu je soubor composer.json, ve kterém jsou informace o knihovnách a jejich verzích. Composer zároveň řeší i závislosti knihoven.

5.2.1.1 Doctrine a Twig

Symfony využívá některé knihovny a frameworky třetích stran. Jednou z nich je šablonovací systém Twig, který slouží pro rederování HTML šablon (umí i jiné formáty), díky dobré integraci do Symfony je snadné ho používat a přidávat do šablon nové funkce.

Dále Symfony využívá Doctrine ORM framework, který poskytuje data z databáze. Díky DBAL (Database Abstraction Layer) umožňuje práci s daty jednotně nezávisle na typu databáze (Mysql, Oracle, MSSQL, PostgreSQL, SQLite, Drizzle). Doctrine také podporuje noSQL databáze jako je mongoDB, v tomto případě používá ODM (Object Document Mapping). Doctrine také cachuje dotazy do databáze bez jakékoliv nutnosti nastavování. Pro mapování objektů používá PHP Anotace nebo xml/yaml configurační soubor.

5.2.1.2 OAuth2 Server Bundle

Pro zabezpeční API pomocí OAuth2 protkolu jsem použil OAuth Server Bundle od Friends of Symfony, který je nadstavbou nad `PHP/oauth2-server` knihovnou a zjednoduššuje její použití v Symfony. V první řadě vyžaduje vytvoření potřebných entit a službu poskytující uživatele (`UserProvider`) a jejich nastavení v configuračím souboru.

5.2.1.3 REST Bundle

Pro vytvoření API kontrolerů jsem použil Rest Bundle od Friends of Symfony. Ten se stará o serializaci entit a generuje REST URL z názvu metody kontroleru, tedy například z `PostsController::getAction(id)` vyge-

neruje URL adresu `/api/posts/:id` pro metodu GET. Dále řeší serializaci tříd v tomto případě pouze do formátu JSON. Pro serializaci využívám balíček `jms/serializer-bundle`, který pomocí anotací v třídách Entit serializuje objekty, výchozí hodnoty v tomto balíčku jsou nastavené tak, že je nutné anotovat pouze výjimky. Například ve výchozím nastavení neserializuje objekty a pole objektů, pomocí anotace `@Exclude` se označí atribut jako neserializovatelný, pomocí anotace `@VirtualProperty` nad metodou se serialzuje výstup metody jako atribut.

5.2.1.4 API Doc Bundle

Pro vytvoření dokumentace k REST API používám `nelmio/api-doc-bundle`. Ten se stará o generování dokumentace k API (výchozí adresa dokumentace je `/api/doc`), která popisuje všechny zdroje a jejich metody, které API podporuje. Popis Entit, které slouží jako zdroje v REST API, a kontrolerů se tvoří pomocí PHP anotace.

5.2.1.5 Cloudinary

Pro ukládání obrázků jsem použil službu Cloudinary, která pomocí API ukládá obrázky do cloudového úložiště. Cloudinary API je zabezpečené pomocí klíče a tokenu, veškerá komunikace probíhá přes protokol HTTPS a v tomto případě je přístupné pouze z back-endové aplikace. Obrázky do Cloudinary nahrávám přes formát base64, který se posílá z Android aplikace současně s požadavkem.

Díky tomu nebude webový server zahlcován správou obrázků, protože v případě sdíleného hostingu jsou systémové prostředky nedostatečné na úpravu obrázků(změna velikosti, ořez, vytvoření miniatury). Tyto a další úpravy Cloudinary API podporuje, v databázi aplikace stačí ukládat `public_id`, které vrátí metoda `Cloudinary::upload`. Cloudinary má navíc řadu knihoven pro většinu nejrozšířenějších programovacích jazyků. Existuje Symfony balíček, který obaluje základní Cloudinary knihovnu pro PHP a vytváří Symfony Service z Cloudinary API tříd. Další výhodou je, že je služba zdarma do limitu 10 GB úložiště a 20 GB odeslaných dat měsíčně.

5.2.1.6 Server pro testování

Pro testování back-endu jsem se rozhodl použít https://www.000webhost.com/, což je služba, která zdarma nabízí testovací doménu, sdílený hosting PHP aplikace (podporuje i verze 7.1) i MySQL server.

Kapitola 6

Testování

6.0.1 Testování REST API

Nejprve jsem prošel v prohlížeči všechny adresy REST API a kontroloval, zda jsou data správně strukturovaná a obsahují všechny atributy. Poté jsem napsal jednoduchou webovou aplikaci v Symfony, díky které jsem mohl projít i všechny vnořené zdroje REST API a vyzkoušet registraci uživatele a jeho autorizaci.

Následně jsem API testoval pomocí PHPUnit, které je do Symfony zaintegrované pomocí knihovny `symfony/phpunit-bridge`. V testovacím skriptu jsem prošel entity z databáze a pomocí id jsem vytvořil požadavek na server a porovnával deserializovaná data z JSON odpovědi s hodnotami v objektu dané Entity. Díky tomuto testování jsem objevil pouze několik minoritních chyb, které byly způsobeny překlepy ve zdrojovém kódu.

6.1 Testování aplikace

6.1.1 Testování programátorem

Aplikaci jsem testoval na těchto zařízeních: Google Pixel (Android 8.1), Nokia 6 (Android 8.0) a Samsung S6 (Android 7.1). Nejprve jsem procházel typické scénáře - vyhledání malby, zobrazení detailu, přidání příspěvku. Poté jsem zkoušel ty méně obvyklé scénáře a zároveň krajní situace.

Narazil jsem na několik chyb při komunikaci s API. Použitá knihovna `angular-http` vyhodnocovala odpovědi na požadavky POST a DELETE jako chybné i přesto, že měly být kladné podle obsahu odpovědi. Tento problém vyřešilo správné nastavení typu odpovědi. Dále jsem se setkal s větší odezvou při načítání velkého množství maleb do hlavní mapy. Tento problém jiné aplikace řeší shlukováním bodů při větším oddálení mapy.

Dále jsem narazil na chybu, kdy aplikace spadla po přechodu zpět na obrazovku obsahující `TabView` a v něm záložky s mapou a `ListView` nebo

`ScrollView`. S chybou tohoto typu se setkala řada vývojářů, jak jsem zjistil při vyhledávání na fórech týkajících těchto technologií. Tato chyba byla způsobena balíčkem pro zobrazení mapy a jedno z řešení bylo nastavit v callbacku načtení komponentu jeho id, které je předem uložené v XML souboru.

Dále jsem narazil na pomalejší běh aplikace a horší responzitiva při velkém oddálení hlavní mapy z důvodů velkého množství pinů na obrazovce. Tento problém se může vyřešit shlukováním sousedících maleb do jednoho pinu při větším oddálení mapy.

6.2 Testování použitelnosti

Po dokončení funkčního prototypu aplikace a jejího otestování programátorem jsem provedl testování použitelnosti, o kterém jsem se dozvěděl v předmětu BI-TUR. Jedná se o metodu, kdy dostane tester pre-test dotazník, post-test dotazník a cíle aplikace, které má splnit. Pozorovatel při plnění cílů testera sleduje a zapisuje si poznámky.

6.2.1 Pre-test dotazník

- Věková kategorie?
- Pravák levák?
- Zkušenosti s platformou Android?

6.2.2 Úkoly

- Najít nejbližší malby.
- Vyhledat malby na konkrétní adrese.
- Zobrazit informace o malbě.
- Přidat příspěvek k malbě.

6.2.3 Post-test dotazník

- Bylo vyhledání nebližších maleb jednoduché?
- Bylo vyhledání maleb na konkrétní adrese jednoduché?
- Bylo zřejmé, které malby již byly nalezeny a které ne?
- Bylo zřejmé, že adresa nenalezné malby není přesná?
- Bylo jednoduché získat informace o malbě?
- Bylo snadné přidat nový příspěvek?

- Byla navigace mezi obrazovkami přehledná?
- Odpovídá aplikace standardům Android platformy?
- Jaký je Váš dojem z aplikace?

6.2.4 Výsledky testování

Testování se účastnilo pět testerů, kteří byli převážně z věkové skupiny 20-30, naprostá většina z nich měla již zkušenosti s platformou Android. S prvními třemi úkoly neměl žádný tester větší problémy, objevilo se zde ale několik menších nedostatků.

Z testování jsem se dozvěděl, že nebylo na první pohled dostatečně jednoznačné, co znamená zelené a červené zvýraznění maleb na mapě. Dále nebyl dostatečně výrazný panel ve spodní části mapy na hlavní obrazovce zobrazující vybranou malbu. Jednoho uživatele hned nenapadlo, že může na tento panel kliknout a dostat se na detail (po chvilce to zjistil, ale nepřišlo mu to intuitivní). Dále některým testerům nebylo jasné, jaká je souvislost mezi seznamem příspěvků a jejich pozicí na mapě v detailu malby. Dále se některým testerům přiležitostně stalo, že nebyly vidět texty tlačítek a popisů, dokud neprovedli v obrazovce libovolnou akci, zde se jedná spíš o chybu způsobenou programátorem.

První problém v uživatelském rozhraní jsem vyřešil tím, že jsem panel s vybranou malbou zvětšil a přidal další základní informace o malbě včetně miniatury, stavu hledání. K příspěvkům v seznamu jsem přidal barevné (podle stavu hledání) ikonky s lokací, příspěvky bez lokace nemají tuto ikonku. Chyba s neviditelnými texty byla způsobena knihovnou Angular a šla se vyřešit jednodušše zavoláním metody pro aktualizaci obrazovky při inicializaci komponentu. Také se zde objevila chyba, kvůli které nebylo možné splnit přidání příspěvku, tuto chybu jsem ale před dalším testováním odstranil. Jednalo se o registraci uživatele, která po každém odeslání vrátila chybu registrace.

Na základě tohoto testování jsem některé chyby a nedostatky v uživatelském rozhraní již stihl vyřešit, jelikož se jednalo o drobnosti, které znamenaly malé zásahy do kódu. Další jsem zařadil mezi krátodobé cíle.

Kapitola 7

Další rozvoj aplikace

7.1 Krátkodobé cíle

V krátkodobém horizontu mám v plánu odladit aplikaci, opravit chyby, které byly objeveny na základě posledního testování, a zajistit její stabilitu. Jedná se zejména o vylepšení UI filtrování, přidání úvodní nápovědy a obrazovku s informacemi, aby se snadno nový uživatel dozvěděl, k čemu aplikace slouží a jak se používá. Dále se nabízí možnost přidat nahrávání příspěvků na pozadí. V aplikaci vyšlo z testování i několik optimalizačních nedostatků, které mám v plánu vyřešit. Po těchto úpravách zmenším velikost aplikace pomocí nástroje Webpack a zároveň provedu obfuskaci kódu. Poté může být aplikace vydána na Google Play, kde bude veřejně přístupná uživatelům OS Android.

Další položkou v pořadí je vytvoření webové aplikace a administrace, která bude odpovídat té aktuální. Tato položka je důležitá, jelikož skrze administraci probíhá proces schvalování příspěvků a přidávání maleb. Výhodou je, že webová aplikace může sdílet část zdrojových kódů s Android aplikací díky použité technologii. Back-endová aplikace je také připravená na tato rozšíření a proto by jejich implementace a integrace do projektu neměla být časově náročná. Následně bude potřeba provést migraci stávající databáze na novou a případně převést i doménu.

7.2 Dlouhodobé cíle

Z dlouhodobého hlediska se nabízí možnost vytvořit odpovídající iOS aplikaci, jelikož NativeScript tuto platformu také podporuje. Pravděpodobně bude nutné provést úpravy, aby vzhled a chování aplikace odpovídaly standardům iOS platformy, veškerá logika ovšem zůstane stejná.

Dále by aplikace mohla být vícejazyčná a tím pádem být přístupná i v dalších zemích, což by mohlo vést k rozšíření uživatelské základny. Také by mohla obsahovat více uživatelské interakce, například umožňovat hodnocení

příspěvků dalších uživatelů nebo zobrazovat jejich skóre podle počtu nalezených maleb. Také by aplikace mohla u již nalezených maleb znemožnit přidání příspěvku, pokud uživatel není v okolí daného místa.

Závěr

Cílem práce bylo vytvořit funkční prototyp Android aplikace k aplikaci Artopos.net, která bude komunikovat s back-endem pomocí REST API. Aplikace měla zobrazovat malby a umožňovat uživateli přidávat příspěvky.

V práci jsem se zabýval analýzou webové aplikace Artopos.net a analýzou užitelských požadavků pro Android aplikaci. Také jsem se zabýval analýzou technologií pro vývoj na operační systém Android. Dále jsem analyzoval možnosti vytváření API a jejího zabezpečení. Dalším bodem této práce bylo vytvoření návrhu uživatelského rozhraní a prototypu aplikace a také návrh back-endové aplikace, ve které se budou ukládat data a která bude obsahovat REST API. Poté jsem implementoval prototyp Android aplikace za použití frameworku NativeScript a back-endovou aplikaci v jazyce PHP a frameworku Symfony. Nakonec jsem otestoval REST API pomocí nástroje PHPUnit a prototyp Android aplikace pomocí testování použitelnosti.

Vytvořená aplikace umožňuje prohlížet na mapě nejblížší krajinomalby z databáze a přidávat příspěvky s fotografií, popisem a polohou. Dále umožňuje zobrazovat malby v seznamu a vyhledávat podle názvu, názvu autora. Uživatel může zadat kritéria, podle kterých budou výsledky filtrovány. Back-endová aplikace je spuštěná spolu s MySQL databází na sdíleném hostingu. Adresa REST API je https://artopos.000webhostapp.com/api a jeho dokumentace je k dispozici na adrese https://artopos.000webhostapp.com/api/doc.

Původní webové aplikace byla velice zastaralá a špatně strukturovaná, proto nebylo vhodné její další rozšiřování. Místo toho jsem zvolil řešení napsat novou back-endovou aplikaci s REST API, na které se v budoucnosti může napojit i nová webová aplikace a administrace.

Tato práce také může sloužit jako příručka pro tvorbu multiplatformních aplikací, které zároveň potřebují komunikovat s back-endem prostřednictvím API. Porovnává technologie sloužící pro tyto účely a ukazuje, jak postupovat při vývoji podobné aplikace a čemu se vyvarovat.

V této práci jsem získal spoustu nových zkušeností. Zjistil jsem, jaká úskalí přináší navrhování uživatelského rozhraní pro operační systém Android a v

neposlední řadě jsem se naučil používat framework NativeScript.

Jedním z krátkodobých cílů bude odstranit nedostatky objevené při testování a vydat aplikaci na Google Play. Díky frameworku NativeScript se také nabízí možnost vytvořit v budoucnu odpovídající iOS aplikaci. V plánu mám i vytvoření odpovídající webové aplikace včetně administrace a migraci originální databáze, aby mohla nová aplikace plně nahradit tu původní.

Literatura

[1] Statista: Mobile OS market share 2017. Dostupné z: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

[2] Lethbridge, T. C.; Laganiere, R.: *Object-oriented software engineering*. McGraw-Hill New York, 2005.

[3] Gomaa, H.: *Software modeling and design: UML, use cases, patterns, and software architectures*. Cambridge University Press, 2011.

[4] Heuristic Evaluations and Expert Reviews. Oct 2013. Dostupné z: https://www.usability.gov/how-to-and-tools/methods/heuristic-evaluation.html

[5] Eriksson, H.-E.; Penker, M.: Business modeling with UML. 2000.

[6] URBAN, V.: E-learningové materialy pro výuku jazyka UML [online]. 2014 [cit. 2018-04-29]. Dostupné z: DostupnézWWW<https://is.muni.cz/th/tyzvv/>

[7] Mens, T.; Van Gorp, P.: A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, ročník 152, 2006: s. 125–142.

[8] Material Design. Dostupné z: https://material.io/

[9] Design for Android. Dostupné z: https://developer.android.com/design/

[10] Mono. Dostupné z: https://www.mono-project.com/

[11] Smeets, R.; Aerts, K.: Trends in Web Based Cross Platform Technologies. *International Journal of Computer Science and Mobile Computing*, ročník 5, č. 6, 2016: s. 190–199.

[12] JavaScriptCore. Dostupné z: https://trac.webkit.org/wiki/JavaScriptCore

[13] React Native · A framework for building native apps using React. Dostupné z: http://facebook.github.io/react-native/

[14] Chrome V8 | Google Developers. Dostupné z: https://developers.google.com/v8/

[15] Masse, M.: *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*. "O'Reilly Media, Inc.", 2011.

[16] SensioLabs: Symfony, High Performance PHP Framework for Web Development. Dostupné z: https://symfony.com/

[17] Parker, D.: *JavaScript with Promises: Managing Asynchronous Code*. "O'Reilly Media, Inc.", 2015.

[18] Doctrine, The Open-Source PHP ORM and Persistence Tools Project. Dostupné z: https://www.doctrine-project.org/

Příloha A

Seznam použitých zkratek

API Application programming interface - rozhraní programu pro programátory.

BI-SI Předmět Softwarové inženýrství.

BI-TUR Předmět Tvorba uživatelských rozhraní.

CMS Content managment system - software zajišťující správu dokumentů, často webového obsahu.

DOM Document object model - model, který reprezentuje strukturu dokumentu (webové stránky).

FAB Floating action button - prvek z Material Design, kruhové tlačítko s ikonkou typicky v pravé dolní části obrazovky.

HTML Hypertext markup language - mostly used on web.

IDE Integrated Development Environment - software pro vývoj aplikací.

JSON Javascript object notation - formát zápisu dat vycházející z javascriptového zápisu objektů.

JVM Java virtual machine - program pro spouštění aplikací v jazyce Java.

JWT JSON web tokens - protokol pro autorizaci skrz API.

NPM Node package manager - správce balíčků běžící na Node.js pro vývoj javascriptových aplikací.

MVC Model View Controller - softwarová architektura rozdělující aplikaci do zobrazovací vrstvy,řidící vrstvy a datové vrstvy.

MySQL Typ SQL databáze.

ORM Object-relational mapping - technika používající se pro mapování dat z databáze na objekty.

PHP PHP: Hypertext Preprocessor - programovací jazyk typicky sloužící pro back-end aplikace.

REST Representational State Transfer - architektura rozhraní, navržená pro distribuované prostředí.

XML Extensible markup language - used for Android Views in Java.

SQL Structured query language - programovací jazyk sloužící pro dotazy do databáze.

UI User interface - uživatelské rozhraní aplikace.

UML Unified modeling language - slouží k tvorbě diagramů pro návrh softwaru.

UX User experience - interakce uživatele a jeho dojmy z produktu.

Příloha B

Slovník

Back-end Část aplikace, která zprostředkovává data a řeší logiku nad nimi.

Callback Zpětné volání funkce.

Event Událost - Situace, kdy objekt vyšle signál, že se něco stalo.

Event Listener Funkce, která se provede při spuštění daného Eventu.

Framework Soubor knihoven či softwaru, který řeší typické problémy oblasti za vývojáře a usnadňuje tak vývoj.

Front-end Část aplikace, která zobrazuje data a kterou vidí uživatel.

ListView UI komponenta, která umožňuje zobrazovat rolovací seznam.

Resource Zdroj - reprezentuje zdroj dat v REST API.

ScrollView UI komponenta, která umožňuje rolovat obsahem.

Swipe Gesto rychlého posunutí prstem po obrazovce.

TabView UI komponenta, která umožňuje zobrazovat záložky.

WebView UI komponenta, která umožňuje zobrazit HTML obsah (např. webovou stránku).

Příloha C

Obsah přiloženého flash disku

- `readme.txt` ..... stručný popis obsahu flashdisku
- `apk` ..... adresář s instalačním balíčkem Android aplikace
  - `screenshots` ..... snímky obrazovek Android aplikace
- `db-model` ..... adresář s exportem do PDF modelu původní databáze
- `form`
  - `user-form` ..... uživatelský dotazník
  - `usability-test` ..... pre-test a post-test dotazník
- `src`
  - `impl`
    - `artopos-app-ns` ..... zdrojové kódy Android aplikace
    - `artopos-symfony4` ..... zdrojové kódy back-endové aplikace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF