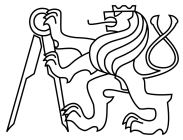
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Mobilni aplikace na hlášení závad ve městě
Student:	Bc. Karel Soukup
Vedoucí:	Ing. Pavel Žikovský, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Cílem práce je návrh a implementace mobilní aplikace na hlášení závad v městech a obcích. Uživatelé pomocí této aplikace budou hlásit nedostatky v městském mobiliáři (např. nefunkční lampy, rozbité cesty atp.) zaměstnancům městského úřadu.

1. Seznamte se s pravidly pro tvorbu uživatelského rozhraní pro Android aplikace.
2. Seznamte se se současnými mobilními aplikacemi, které slouží k hlášení závad ve městech a obcích.
3. Proveďte návrh uživatelského rozhraní mobilní android aplikace s využitím poznatků získaných při testování ostatních aplikací. Navržené rozhraní řádně otestujte na uživatelích ve všech fázích vývoje (lo-fi, hi-fi) - součástí práce tedy budou i uživatelské dotazníky, jejich zhodnoceni a zapracování vzniklých vylepšení.
4. Na základě těchto výzkumů implementujte finální aplikaci. Volba vhodných platforem a technologií je součástí práce.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 8. února 2016

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

Diplomová práce

Mobilní aplikace na hlášení závad ve městě

Bc. Karel Soukup

Vedoucí práce: Ing. Pavel Žikovský, Ph.D

10. května 2016

Poděkování

Děkuji především vedoucímu diplomové práce Ing. Pavlu Žikovskému, Ph.D. za odborné vedení, připomínky a cenné rady. Za podporu při vypracovávání této práce, korekturu a finální úpravy děkuji Janě Morcové.

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 10. května 2016 .....

Odkaz na tuto práci

Soukup, Karel. *Mobilní aplikace na hlášení závad ve městě.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato diplomová práce se zabývá vývojem mobilní aplikace na hlášení závad ve městě. V práci je nejprve stručně popsána webová aplikace se stejnou funkcionalitou, ze které bude daná mobilní aplikace využívat aplikační rozhraní. V další části textu jsou popsána základní pravidla pro tvorbu mobilních aplikací pro operační systém Android a analýza stávajících mobilních aplikací, které slouží k hlášení závad. Velká část práce se zabývá návrhem aplikace včetně testování již od první fáze návrhu až po testování téměř hotové aplikace. V poslední části autor práce popisuje výběr vhodných technologií pro aplikaci a samotnou implementaci aplikace.

Klíčová slova mobilní aplikace, uživatelské rozhraní, Android, uživatelské testování, mapy

Abstract

The diploma thesis deals with development of mobile phone application, which enables citizens to report various problems to city authorities. The first part of the thesis briefly describes a web application with the same purpose. The new mobile phone application uses the application interface of this web version. The next part concerns basic principles and guidelines to create mobile

applications based on the Android operating system guidelines and analyses existing mobile applications. The main part of the thesis deals with design of the application user interface including testing in all development stages. The final part describes a selection of suitable technologies for application development as well as the application implementation itself.

Keywords mobile application, user interface, Android, usability testing, maps

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Před dvěma lety mi volal můj bratr, který je zaměstnancem městského úřadu, že by se se mnou chtěl sejít ohledně aplikace, kterou by rádi měli na jejich úřadě. Byl mi představen nápad, kterým byla webová aplikace kam by občané mohli hlásit různé závady a poruchy, například na veřejném osvětlení.

Tento nápad na úřadě vznikl, protože několikrát týdně museli úředníci řešit telefonáty, emaily nebo dopisy obsahující stížnosti, že něco nefunguje nebo je rozbité. Téměř pokaždé si občané stěžovali zaměstnancům úřadu, kteří danou věc nemohli sami vyřešit. Tito zaměstnanci pak museli přijatou stížnost předat příslušné osobě, což mělo za následek, že řešení daného problému bylo zdlouhavé.

Úřad rozhodl tento problém řešit tím, že se pokusí urychlit jak předání stížnosti odpovědné osobě, tak její rychlé vyřízení.

Po dlouhém hledání se nepodařilo nalézt vhodnou aplikaci, která by splňovala požadavky úřadu. Proto bylo navrženo nové řešení ve formě webové aplikace, kde občané v mapě označí místo problému a problém stručně popíší. Na základě toho, do jaké kategorie občan závadu přidá, se odešle email nebo SMS zpráva osobě, která má danou problematiku na starosti. Například pokud je nahlášena nesvítící lampa, je rovnou ohlášená závada odeslána externí firmě, která danou lampu přijede opravit.

Ne vždy je však nejpohodlnějším způsobem ohlášení závady pomocí webové aplikace. Z tohoto důvodu vznikla další myšlenka, jak by mohl být občanům, ale i zaměstnancům úřadu ušetřen čas. Ke zrychlení by mohlo dojít pomocí mobilní aplikace, kterou by uživatelé mohli využít přímo v ulicích měst a ne jen doma.

Cílem této práce je tedy vytvořit mobilní aplikaci, pomocí které občané měst budou moci hlásit například nesvítící veřejné osvětlení nebo jiný poškozený majetek města. Omezením v této práci je, že mobilní aplikace bude vytvořena pouze pro platformu Android.

Jelikož už podobné aplikace existují, bude obsahem této práce i provedený průzkum těchto aplikací.

Hlavní část práce se bude věnovat návrhu a testování uživatelského rozhraní.

Závěrem bude dle navrženého uživatelského rozhraní vytvořena finální aplikace pomocí vhodných technologií.

Kapitola 1

Popis stávajícího řešení

Tato kapitola se zabývá popisem stávajícího řešení webové aplikace na hlášení závad v městech a obcích, která byla vytvořena autorem této práce a jejíž aplikační rozhraní bude využito při implementaci mobilní aplikace. Nejprve je popsán důvod vzniku samotné aplikace a stručný popis modulů, které celá aplikace obsahuje. Následuje detailnější popis modulu na hlášení závad, který bude využíván při vývoji aplikace v této práci. Na konci této kapitoly se nachází stručný popis technologií využitých při vývoji webové aplikace.

1.1 Vznik

Aplikace vznikla v roce 2013 na základě požadavku Městského Úřadu v Mimoni pro potřeby využití podobné aplikace. Aplikace byla vyvinuta autorem této práce a jedním z modulů byl právě modul na hlášení závad.

První verze aplikace kromě modulu na hlášení závad obsahovala také modul na správu různých objektů ve městě. Do aplikace tak například byli zaneseny objekty lamp veřejného osvětlení, uliční vpusti a odpadkové koše. Modul na správu objektů umožňoval například evidenci provedených úkonů na daných objektech. V praxi to znamená, že zaměstnanci úřadu zaznamenávají do aplikace, například kdy byla provedena nějaká údržba na lampě veřejného osvětlení. Na základě získaných dat, pak mohou zaměstnanci úřadu vyhodnocovat kolik bylo v daném časovém úseku provedeno servisních úkonů a dle toho například požadovat změnu servisních smluv.

Další modul, který aplikace obsahovala, byl modul pro zobrazování informací o plánovaných investicích v daném roce nebo plánovaných dopravních omezeních na území daného města.

1.2 Popis modulu pro hlášení závad

Modulem, který by v této práci měl být detailněji popsán je modul na hlášení závad. Tento modul se rozděluje na dva hlavní typy závad, které mohou občané nahlásit. Jedná se o závadu buď na předem definovaném místě nebo na předem nedefinovaném místě. K obou typům závad je možné i připojit fotografii.

Závada na předem definovaném místě je taková závada, která může být ohlášena na nějakém objektu v mapě (například lampa). Hlášení, například pro veřejné osvětlení, probíhá tak, že se uživateli zobrazí mapa obsahující všechny lampy a uživatel pouze vybere lampu a napíše co je za problém. Takovýto typ závady pak má pro zaměstnance úřadu výhodu zejména v tom, že ví, když je například nějaká lampa neustále porouchaná, že pravděpodobně asi něco není v pořádku. Další výhodou je, že osoba, která bude daný problém řešit (nejčastěji technik) přesně ví o jaký objekt se jedná. Má tak informace o tom jaké jsou parametry daného svítidla a co bude potřeba k jeho opravě.

Závada na předem nedefinovaném místě je taková závada, kdy není předem jasné, kde se může závada nacházet. Nejčastěji se jedná například o poškozenou cestu nebo chodník. Hlášení takové závady probíhá tak, že se uživateli zobrazí mapa a kliknutím do mapy uživatel vybere místo závady a přidá nějaký popis.

Ohlášená závada se může nacházet v jednom ze čtyř stavů. Po ohlášení závady se závada nachází ve stavu *Vytvořeno*. Takový stav dává uživatelům informaci, že závada sice byla ohlášena, ale zatím se s ní nikdo nezabýval. Změnu stavu provádí zaměstnanec úřadu, například technik, který má na starosti opravy lamp veřejného osvětlení. Osoba pověřená ke změně stavu závady po vytvoření závady nastaví závadu do stavu: *v řešení*, *zamítnuto* nebo *hotovo*. Do stavu *v řešení* je závada nastavena v případě, že například úředník objednal opravu dané závady nebo ví, že se daný problém bude řešit v nějakém časovém horizontu. Jako příklad může být ohlášený poškozený chodník, který má být v příštím roce kompletně zrekonstruován. Do stavu *zamítnuto* je závada nastavena v případě, že se jedná o závadu, která se netýká města, nebo není v kompetenci města daný problém řešit. Posledním stavem je stav *hotovo*. Do tohoto stavu se závada nastaví, když je daný problém vyřešen. V některých situacích, kdy je závada vyřešena během jednoho dne, je stav závady změněn rovnou ze stavu *vytvořeno* do stavu *hotovo*. K ohlášené závadě je možné ještě připojit nějakou reakci, například od zaměstnance úřadu. Jako příklad může být, že závada je nastavena do stavu *v řešení* a je přidán komentář, že závada bude opravena do jednoho měsíce.

Aby byla závada co nejrychleji vyřešena, je po jejím vytvoření potřeba, aby byly informovány odpovědné osoby. Na základě kategorie závady je tak odeslána emailová nebo SMS zpráva osobě nebo osobám, které daný problém dokážou zpracovat.

Aby občané tuto aplikaci využívali, bylo potřeba, aby nebyli zbytečně zdržování registrací do aplikace. Proto je možné závadu ohlásit bez nutnosti přihlášení. U závady stačí pouze vyplnit formulář s kontaktními údaji, kdyby

bylo potřeba občana kontaktovat za účelem přesnější definice nebo lokalizace závady.

1.3 Technologie

Webová aplikace je vyvíjena pomocí webového frameworku Play[1] verze 2. Jedná se o webový framework pro jazyk Java a Scala. Aplikace je vyvíjena v jazyce Java a jako přístupovou vrstvu k databázi je využíváno technologie Hibernate[2]. Jedná se o standardní technologii pro objektově relační mapování využívané v jazyce Java. Jako úložiště dat je využívána relační databáze MySQL. Pro úložiště obrázků je využito aplikace Cloudinary[3]. Aplikace Cloudinary slouží mimo úložiště dat, také pro změnu velikosti nahraných fotografií.

Část aplikace, která běží ve webovém prohlížeči, je vyvinuta pomocí frameworku AngularJs[4] využívajícího programovací jazyk Javascript. Pro zobrazení mapových podkladů a samotných objektů je využíváno knihovny OpenLayers 3[5].

Jelikož aplikace byla prvotně navržena pouze pro jedno město, bylo později potřeba vymyslet nejvhodnější způsob, jak umožnit tuto aplikaci využívat i dalším městům a obcím. Dle průzkumu architektury aplikací, které obsluhují více klientů, bylo navrženo řešení, kdy jedna aplikace obsluhuje všechny města a data všech měst jsou uložena v jedné databázi. Takovéto řešení je v dané situaci asi nejlepší, protože nejsou tak vysoké náklady na správu serverů. Snazší je i vývoj dalších verzí, protože novou verzi databáze i aplikace stačí zprovoznit pouze na jednom nebo několika málo serverech. Takovéto řešení má však nevýhodu v tom, že v případě, že by nějaké město chtělo specifické úpravy v aplikaci, je jejich implementace složitější, protože v žádném případě nesmí změny ovlivnit funkcionality ostatních měst.

[1] https://playframework.com/
[2] http://hibernate.org/
[3] https://cloudinary.com
[4] https://angularjs.org/
[5] http://openlayers.org/

Kapitola 2

Analýza

V této částí textu celé práce jsou nejprve popsány základní informace o *Material Design*, což je definice designu, která by měla být využívána u nově vyvíjených Android aplikacích. Po popisu *Material Design* a některých jeho komponentech, jsou shrnuta některá pravidla, která by se měla dodržovat při vývoji nejen mobilních aplikací. Na konci této části textu se nachází analýza stávajících řešení, kde jsou popsány některé mobilní aplikace, se kterými je možné hlásit závady na území měst a obcí. Na základě této analýzy jsou pak shrnuty poznatky pomáhající řešit vývoj mobilní aplikace, která je cílem této práce.

2.1 Material Design

V této kapitole je čerpáno ze zdrojů [4] a oficiálních stránek o Material Design [5].

Material Design je vizuální jazyk společnosti Googl, který spojuje klasické principy dobrého designu s inovací a možnostmi technologií a vědy. Material Design je úzce spojen s Android verzí 5.0 avšak není určen pouze pro Android verze 5.0, ale měl by být využitelný i se staršími verzemi Android případně dokonce i webovými aplikacemi.[4]

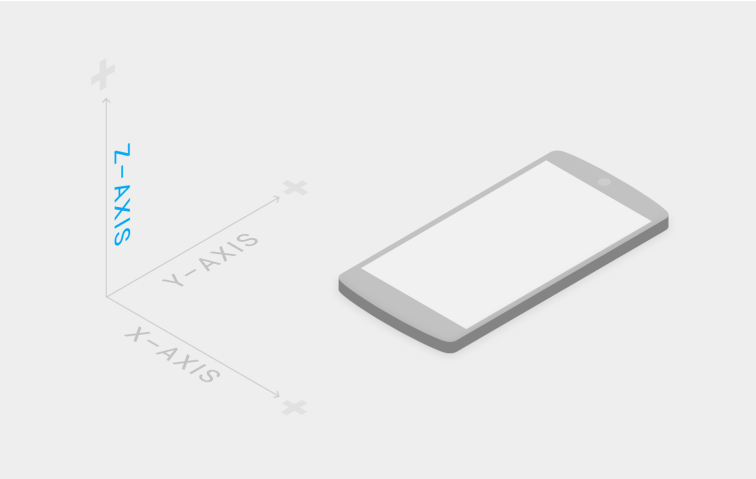
Google popisuje Material Design jako:

A material metaphor is the unifying theory of a rationalized space and a system of motion. The material is grounded in tactile reality, inspired by the study of paper and ink, yet technologically advanced and open to imagination and magic.[5]

Tento popis nebo definice říká, že Material Design je založen na skutečném papíru a inkoustu, ale není limitován pouze tím co může papír a inkoust vytvořit v reálném světě. Papír slouží jako povrch a vše ostatní leží na něm. Papír se může zvětšit nebo zmenšit, což znamená, že například kus papíru se může objevit na obrazovce, jako jeden pixel a poté se zvětšit na velikost celé

obrazovky a na závěr může ještě svůj tvar změnit. Papír se vždy na obrazovce zobrazí pomocí nějakého přechodu (například přechodem ze strany).

Jeden kus papíru může odsunout nějaký další papír pokud dojde ke kolizi jejich hran. Papír je vždy jeden pixel tlustý. Material Design existuje ve 3D prostoru a proto se papíry mohou vyskytovat v několika úrovních na ose Z, viz. obrázek 2.1. Osa Z ovlivňuje vzdálenost mezi jednotlivými papíry, což ovlivňuje výsledný stín. Jelikož zařízení nemají třetí rozměr displeje, hloubka je vytvořena pomocí techniky perspektivy a stínů.



Obrázek 2.1: Osy využívané v Material Design Zdroj: [1]

Stín v Material Design je tvořen pomocí dvou světelných zdrojů: okolního světla a hlavního světla. Jako ukázkový příklad může být pořízení fotografie pomocí fotoaparátu. Hlavním zdrojem světla v tomto případě je blesk fotoaparátu a okolním světlem jsou všechna ostatní. Hlavní zdroj světla vytváří silný stín směřující jasným směrem. Okolní zdroje světla vytváří nevýrazný stín, kde stín je rozšířen do všech směrů. V případě Material Design hlavní zdroj světla směřuje z vrchu do středu obrazovky, čímž je vytvořen stín pod papírem.

2.1.0.1 Typografie

Google Android ve svých posledních verzích využívá písmo Roboto. Roboto je písmo, které bylo speciálně navrženo pro dnešní mobilní zařízení. První verze písma Roboto se vyskytla ve verzi Android 4.0 a ve verzi 5.0 došlo k úpravě některých písmen. Roboto je využíváno pro jazyky, které využívají latinskou, řeckou nebo cyrilickou abecedu. Pro ostatní abecedy je využíváno písmo Noto.

2.1.0.2 Interakce

Jedním s dalších aspektů, který je často ignorován je interakce. Jaká bude interakce v případě, že uživatel klikne na libovolné tlačítko? Je jasné, že se například zobrazí další obrazovka, ale v Material Design je kladen velký důraz také na proces jak se daná obrazovka zobrazí. Zda se tlačítko nějak zmenší, posune nebo změní barvu. Zda následující obrazovka posunem ze strany překryje obrazovku na které bylo tlačítko, nebo zda se vynoří ze stávající obrazovky. V Material Design by měly být všechny přechody plynulé, animace by měly na začátku postupně zrychlovat a před koncem plynule zpomalovat. Nemělo by tedy docházet k tomu, že animace začne na své 100% rychlosti. Animace by neměly sloužit k tomu, aby uživatele ohromily, ale k tomu aby uživateli pomohly porozumět přechodu z jedné obrazovky na druhou. Animace by měla upoutat oči uživatele na důležitý element a ukázat k jaké dochází změně.

2.1.0.3 Metriky a zarovnání

V této kapitole bude využívána zkratka dp, která je zkratkou density-independent pixel. Jedná se o jednotku, která není závislá na rozlišení displeje. V Material Design by měl být každý element na obrazovce zarovnán pomocí hodnoty, která je násobkem 4dp. Například pravá a levá strana obrazovky jsou typicky na telefonech odsazeny 16dp. Pokud je v seznamu zobrazena ikonka na levé straně tak by dle Material Design měl text, který následuje za ikonkou, být odsazen 72db od levého okraje.

2.1.1 Klíčové zásady pro tvorbu mobilních aplikací

V této kapitole budou popsány klíčové principy, které by měly být dodržovány při tvorbě mobilní aplikace. Prvním, čím by vůbec tvůrce aplikace měl začít, je definice cílů aplikace. Jedná se o jedno z klíčových pravidel, které je velice často podceňováno nebo ignorováno.

2.1.1.1 Dělej jednu věc a dělej ji pořádně

Čím detailněji je aplikace zaměřena, tím je snazší zajistit, aby plnila svůj účel. Jakmile vývojář začne vytvářet aplikaci, která toho má umět co nejvíce, tak s jistotou dosáhne pouze toho, že aplikace bude průměrná. Vývojář by si před začátek vývoje aplikace měl nejprve vytvořit seznam, co by aplikace podle něj měla umět. Poté by měl dále postupovat tím, že začne vyškrtávat nejméně důležité věci na seznamu, dokud se nedostane do stavu, že seznam obsahuje pouze absolutní základy nutné pro aplikaci. Doplnit další funkcionality později je vždy možné. Jednoduchou pomůckou pro splnění této zásady je, jakmile si vývojář dokáže odpovědět na otázku: „Proč bude někdo používat tuto aplikaci?“. Na tuto otázku by mělo být odpovězeno jednou větou bez spojek "a"a "nebo".

2.1.1.2 Spolupracuj s ostatními

Přestože by aplikace, dle předešlé zásady, měla dělat pouze jednu věc, neznamená to, že by uživatele omezovala. Jelikož aplikace jsou pouze komponenty, které může uživatel využívat, aplikace by měla detekovat co se snaží uživatel provést. Dle toho co se snaží uživatel vykonat by pak aplikace měla vybrat vhodnou aplikaci z jeho zařízení, která dokončí uživatelův cíl. Například když uživatel potřebuje upravit fotografii, která má být využita v aplikaci, nemá smysl tuto funkcionalitu v aplikaci implementovat. Stačí pouze, když je uživateli umožněno využít jinou aplikaci na úpravu fotografií, kterou uživatel zajisté ve svém zařízení má. Jako další příklad může být to, že uživatel chce sdílet například nějakou poznámku z aplikace. Nemá smysl v této aplikaci implementovat všechny API sociálních sítí. Pouze stačí, když je uživateli umožněno si vybrat aplikaci, se kterou chce sdílet vytvořenou poznámku.

2.1.1.3 Vizualizace

Jednou z hlavních výzev pro mobilní aplikace je to, že je potřeba sdělit velké množství informací, ale máme pouze omezenou velikost displeje. Často se stává, že uživatel pouze rychle koukne na aplikaci, což znamená, že pro něj musí být snadné rychle najít potřebné informace. Proto je potřeba používat krátké texty pro nadpisy, dialogy a instrukce. Je také vhodné využívat obrázky pro rychlé sdělení co například dané tlačítko bude dělat.

2.1.1.4 Jednoduché ale výkonné

Uživatelé, kteří využívají aplikaci poprvé, jsou velice odsuzující a proto je kritické, aby byla aplikace na první pohled snadno použitelná. Základní funkce by měly být jasné. Například když uživatel využívá aplikaci pro správu seznamu kontaktů a na úvodní obrazovce vidí velké tlačítko obsahující plus, je jasné že tímto tlačítkem musí dojít k vytvoření nového kontaktu. Dále by uživateli měl být poskytnut co největší komfort. Pokud tedy uživatel v aplikaci na poznámky chce vytvořit nějakou poznámku, tak mu hned poskytnout předvyplněné údaje s dnešním datem, případně poskytnout okamžité informace o pozici, kde byla poznámka pořízena.

Dalším velice důležitým pravidlem pro snadno použitelné aplikace je, že uživatel má vždycky pravdu i když udělá chybu. Když například uživatel klikne na tlačítko Odstranit tak v 99% případů opravdu chtěl provést odstranění. Proto není nutné uživatele dotazovat dialogovým oknem, zda opravdu chce provést odstranění. Uživatelsky přívětivější je provést ihned odstranění, ale poskytnou uživateli možnost provést tzv. „undo“, které zajistí že se právě provedená akce vrátí zpět. Dialogové okno s dotazem zda má být akce provedena je vhodné zobrazit pouze u akcí, které není možné vrátit zpět, jako je například formátování systému.

2.1.1.5 Platformová jednotnost

Dalším pravidlem je dodržování pravidel, která jsou typická pro cílovou platformu. Pokud tedy neexistuje velice dobrý důvod, neměla by být pravidla pro cílovou platformu porušována. Proto tedy není vhodné navrhovat aplikaci, tak aby se chovala stejně na více než jedné platformě. Příkladem může být aplikace pro platformy Android a iOS, kde obě platformy mají jiná pravidla. Každá platforma má jiné zvyklosti a tak pro uživatele může být matoucí, když aplikace je univerzální pro obě platformy a tak třeba tlačítko zpět se nachází na opačné straně.

2.1.2 Standardní komponenty

V této kapitole budou popsány, některé základní komponenty a funkce důležité na platformě Android.

2.1.2.1 Systémové lišty

Android má dvě systémové lišty: stavová lišta a navigační lišta. Navigační lišta je na dolním okraji displeje a skládá se z tlačítek zpět, domů a přehled. Navigační lišta je zobrazena pouze pokud nejsou na zařízení dostupná stejná hardwarová tlačítka. Stavová lišta je na horním okraji displeje a zobrazuje ikony pro notifikace na levé straně a na pravé straně informace o stavu telefonu.

Ve většině případu by měly být systémové lišty zobrazeny. Vývojář sice může stavovou lištu skrýt, ale takovýto krok dává smysl pouze v případě, že aplikace například chce zobrazit video nebo fotografii přes celou obrazovku.

2.1.2.2 Notifikace

Android již od svých prvních verzí je navržen tak, aby umožňoval víceúlohovost. Právě notifikace je funkcionalita umožňující dát uživateli informaci, že se něco stalo nebo se právě děje v aplikaci, která běží na pozadí. Může se například jednat o upozornění na přijetí nového emailu nebo informace o probíhajícím stahování dat do aplikace.

S Android verzí 4.1 je také možné, aby notifikace obsahovala více informací. V případě přijetí emailu se může jednat, například o informace o předmětu a odesílateli zprávy. V případě, že je zobrazeno více informací o notifikaci, tak je možné, aby se zobrazila například tlačítka na vytvoření odpovědi nebo odstranění emailové zprávy.

2.1.2.3 Aplikační lišta

Aplikační lišta se nachází úplně nahoře pod stavovou lištou. Na levé straně tato lišta nejčastěji obsahuje navigační tlačítko. Navigační ikonka je buď šipka směřující vlevo nebo ikonka skládající se ze třech horizontálních čar (tzv. hamburger ikona). Ikonka šipky směřující vlevo slouží k navigaci o úroveň výše

v hierarchii aplikace. Ikonka skládající se z třech horizontálních čar slouží k otevření hlavního menu. V pravé části aplikační lišty se může nacházet ikona třech teček. Po kliknutí se zobrazí vyskakovací menu, které by mělo obsahovat méně využívané akce nebo akce, které se nehodí na aplikační lištu. Nejčastěji se jedná například o akci, která zobrazí nastavení.

V aplikační liště je sice možné zobrazit i ikonku aplikace, ale obecně to není vhodné, je lepší tuto lištu využít zejména k zjednodušení používání aplikace. Standardní výška aplikační lišty je 56dp na mobilních zařízeních, ale výška může být i větší pokud je to potřeba. Je možné také to, aby aplikační lišta měnila svou velikost, pomocí animace, na základě toho v jakém stavu momentálně aplikace je.

Využívat aplikační lištu má smysl pro většinu aplikací, ale je možné ji skrýt pokud je to potřeba. Skrýt aplikační lištu dává smysl například při sledování videa nebo čtení delšího textu. Běžné chování v Material Design je, že pomocí posouvání obsahu směrem výše aplikační lišta zmizí a opět se objeví zpět, když ji uživatel může opět potřebovat (posunuje obsah směrem dolů).

2.1.2.4 FAB

FAB je výrazné kruhové plovoucí tlačítko, které slouží k vykonání hlavní akce na dané obrazovce. Nejčastěji se jedná o akci jako je vytvoření nové položky, uživatele, poznámky apod. Například, pokud by se jednalo o aplikaci sloužící pro správu kontaktů tak by s velkou pravděpodobností tlačítko FAB sloužilo k vytvoření nového kontaktu.

2.1.3 Navigační komponenty

Google Android umožňuje používat tzv. navigační vzory, které určují, jak se bude moci uživatel v aplikaci pohybovat.

2.1.3.1 Dashboard pattern

Jedná se o jeden z prvních vzorů používaných na Androidu. Aplikace, které využívají tento vzor, zobrazují množinu elementů zobrazených na úvodní stránce jako mřížku ikon. Tento vzhled je dobrý pro aplikace, které mají za cíl zobrazovat omezené množství elementů.

Vzhled se skládá z maximálně dvou ikon na jednom řádku a počet sloupců je omezen podle velikosti obrazovky. Tyto ikony jasně zobrazují symboly hlavních funkcionalit s tím, že všechny možnosti mají stejnou váhu. Tento vzor je ideální pro aplikace, které mají velkou cílovou skupinu osob, která bude aplikaci používat.

Tomuto vzoru je dobré se vyhnout jakmile je potřeba zobrazit vetší množství elementů a dané elementy se nevejdou na obrazovku a bylo by tak nutné pro zobrazení ikon posouvat obrazovku. Stejně tak je dobré tento vzor nevyužívat pokud je třeba na obrazovce zobrazit pouze malé množství ikon. [6]

2.1.4 Sliding panel / Navigation Drawer

Tento vzor je známý díky mobilním aplikacím jako je Facebook a Gmail. Obrazovka s menu se zobrazí po kliknutí do pravého horního (resp. levého horního) rohu na ikonku s třemi horizontálními čarami. Dále je možné menu zobrazit tažením prstu z levého (resp. pravého) okraje na střed obrazovky. Tento vzor se hodí pokud máme velké množství voleb (elementů), které mají všechny stejnou váhu. Tento vzor můžeme také kombinovat se vzorem Tab pattern. [6]

Ovládání probíhá tak, že jakmile klikneme na element v menu, zobrazí se jeho potomek na obrazovce. Z tohoto potomka je možné se například navigovat dále na další potomky (potomky potomka). Nikdy ovšem potomek nesmí odkazovat na element v hlavním menu. Pomocí tohoto vzoru je také možné vytvářet menu, kde položky mohou obsahovat další podpoložky.

Tento vzor není vhodný na vytváření aplikací, které nemají velké množství voleb (možností).

2.1.5 Tabs pattern

Tento vzor zobrazuje pevné menu s komponenty, které všechny mají stejnou úroveň. Jakmile je použit tento vzor tak je menu neustále vidět.

S tímto vzorem je možné využít jeho dvě varianty. První varianta je taková, že všechny položky se vejdou na obrazovku a tak může být menu pevně zobrazeno na obrazovce. V druhé variantě je větší množství položek a tak se všechny nevejdou na obrazovku, v takovém případě je možné s menu posouvat do stran a zobrazit tak ostatní položky. [6]

2.1.6 Bottom Navigation

Jedná se o spodní navigační lištu, která má za úkol přepínat mezi hlavními obrazovkami jedním kliknutím. Obsahem jednotlivých položek navigace je popis a ikonka.

Spodní navigace by měla být použita v případě, že je potřeba zobrazit 3-5 položek. Tato navigace by měla být statická a proto není vhodné zobrazovat více položek, což by zapříčinilo nutnost posouvaní položek.

Když je využívána spodní navigace, tak by se na dané obrazovce v horní části neměly nacházet záložky, protože to může být pro uživatele matoucí. Popis položek by měl být co nejkratší a nemělo by docházet k zalamování textu.[7]

2.1.7 Závěr

Tato kapitola měla za cíl představit *Material Design* a popsat některá pravidla, která by měla být dodržována při návrhu a vývoji samotné aplikace na hlášení závad. V textu se také nachází popis některých základních komponent

využívaných pro vývoj Android aplikací. Cílem této kapitoly nebylo popsat do detailu celý *Material Design*, ale alespoň ve zkratce popsat o co se jedná a ukázat některé komponenty, které budou pravděpodobně využity při vývoji a návrhu aplikace.

2.2 Analýza existujících řešení

Tato kapitola se zabývá analýzou podobných projektů, které se zabývají hlášením závad v městech a obcích. Analyzované jsou pouze aplikace pro platformu Android.

2.2.1 Dej Tip

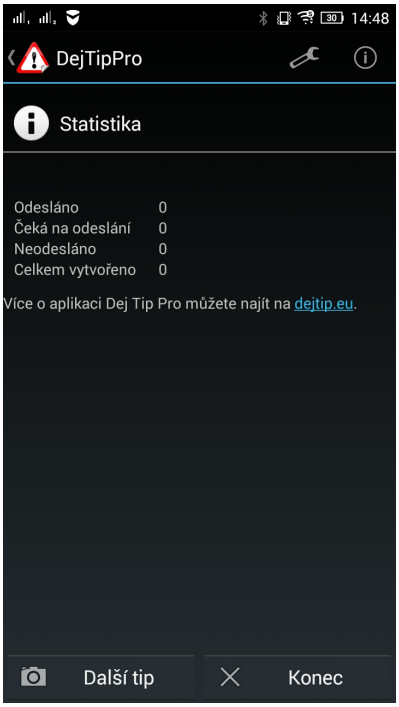
Aplikace Dej Tip je vyvinuta firmou Intergraph CS s.r.o. pro firmu GEFOS a.s. Aplikace slouží k hlášení závad v obcích, které jsou zapojeny do programu Dej Tip. Navíc je možné v aplikaci také hlásit závady, které spadají pod Správu a údržbu silnic, jako jsou závady na mostech, silnicích, svodidlech apod.

V horním menu jsou tři tlačítka. Na pravé straně se nachází tlačítko s názvem aplikace, které má sloužit jako tlačítko zpět. Na úvodní stránce je sice možné na tlačítko kliknout, ale nic se nestane. Na pravé straně se nachází tlačítka pro nastavení a pro zobrazení informací o autorovi aplikace.

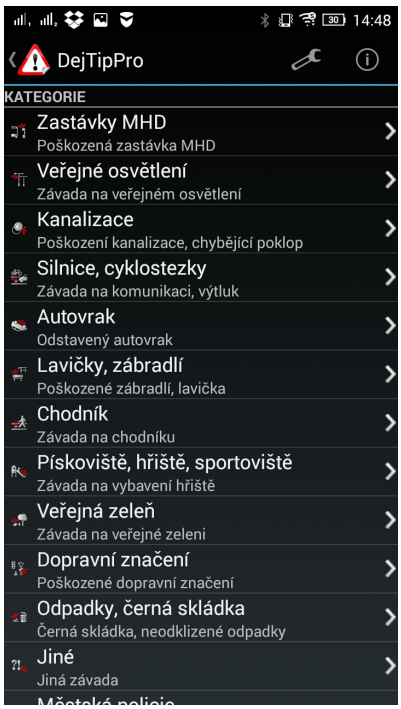
Na úvodní obrazovce aplikace se nám zobrazí statistiky o tom, kolik bylo doposud odesláno závad z našeho zařízení, kolik závad čeká na odeslání, kolik závad nebylo odesláno a kolik závad bylo doposud vytvořeno na našem zařízení. Navíc je zde také umístěna informace s webovými stránkami, kde je možné získat více informací. V dolní části se nachází dvě tlačítka. První tlačíko *Další tip* slouží k nahlášení závady. Druhé tlačítko slouží k ukončení aplikace. Úvodní obrazovku je možné vidět na obrázku 2.2a.

Po kliknutí na tlačítko *Další tip* se spustí fotoaparát a je možné pořídit fotografii, standardním způsobem jako ve všech aplikacích Android. Po pořízení fotografie se zobrazí obrazovka s nadpisem témata, kde je možné vybrat *DEJTIP obce* nebo *DEJTIP SÚS*. U každého tématu je navíc zobrazen krátký popis. U *DEJTIP obce* je popis *Hlášení závad na území měst a obcí*. U *DEJTIP SÚS* se nachází popis *Správa a údržba silnic, hlášení závad*. Zejména nadpis této obrazovky nedává přesnou informaci o tom, co má uživatel na této stránce udělat. Pod nadpisem *Témata* by autor této práce očekával jiný obsah. Popisy položek jsou navíc trochu nekonzistentní. Vhodnější popis pro položku *DEJTIP SÚS* by byl *Hlášení závad pro Správu a údržbu silnic*.

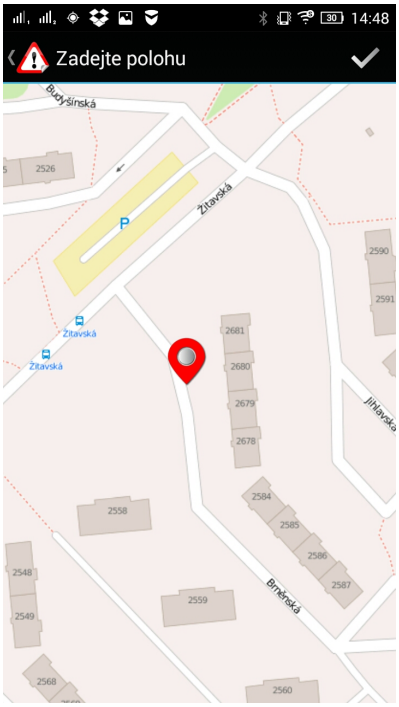
Po vybrání jednoho z témat se zobrazí obrazovka s nadpisem *Kategorie*, kde uživatel vybere kategorii, které se porucha týká. Při výběru tématu hlášení závad na území měst a obcí se zobrazí 14 kategorií, kde je možné vybrat například veřejné osvětlení, kanalizace, silnice, cyklostezky apod. Tato obrazovka je přehledná, jediný problém je, že se zde nachází větší množství kategorií. Obrazovku je možné vidět na obrázku 2.2b.



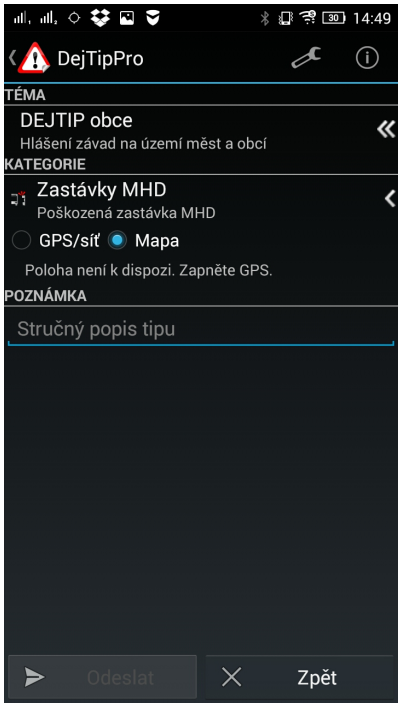
(a) Úvodní obrazovka aplikace



(b) Výběr kategorie problému



(c) Výběr pozice problému



(d) Formulář pro popis problému

Obrázek 2.2: Ukázky obrazovek z aplikace Dej Tip

Po vybrání kategorie se zobrazí obrazovka (viz. obrázek 2.2d), kde je možné ještě změnit téma a kategorii. Dále se na této obrazovce nachází informace o lokalizaci závady. Je zde možné buď využít možnost lokalizace závady pomocí GPS, kdy je po zjištění souřadnic zobrazena také informace o tom jaká je přesnost získaných souřadnic. Další možností je vybrat souřadnice pomocí mapy. Po kliknutí na *radio button* s popiskem *Mapa* se zobrazí mapa (viz. obrázek 2.2c) a poté je možné kliknutím do mapy umístit ikonku na místo, kde se závada nachází. Poté je možné vybranou pozici na mapě potvrdit tlačítkem v pravém horním rohu. Dále se na této obrazovce nachází vstupní pole s popisem poznámka, které slouží k zadání informací o poruše. Odeslání poruchy se provede kliknutím na tlačítko *Odeslat* v levém dolním rohu. V právem dolním rohu se nachází tlačítko *Zpět*. Po kliknutí na toto tlačítko se zobrazí potvrzovací dialog o tom, že závada nebyla odeslána a zda opravdu uživatel chce ukončit tuto akci. Po potvrzení tohoto dialogového okna se aplikace vrátí na úvodní obrazovku. Od tlačítka *zpět* by autor práce očekával vrácení na obrazovku s výběrem kategorie.

2.2.1.1 Shrnutí

Klady

- Informace o tom jaká je přesnost GPS při hlášení závady
- Možnost hlášení závady jak pomocí GPS tak pomocí mapy
- Možnost vytvořit poruchu bez internetového připojení (porucha je odeslána později)

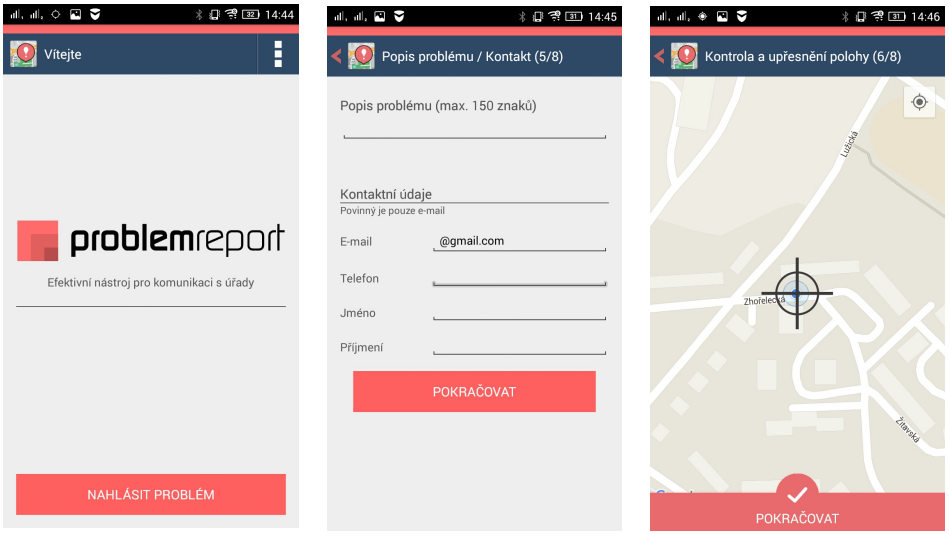
Zápory

- Zbytečné tlačítko *Konec* na úvodní stránce
- Středem pozornosti na úvodní stránce jsou statistiky o tom, kolik bylo nahlášeno závad
- Nutnost pořídit fotografii pro nahlášení poruchy

2.2.2 Problem report

Problem report je aplikace od firmy Hrdlička spol. s r.o. sloužící k hlášení závad v městech. Mobilní aplikaci je zatím možné využít pouze v České Lípě. Webovou aplikaci je možné využít i v dalších městech například v Polsku nebo na Slovensku.

Na úvodní obrazovce aplikace (viz obrázek 2.3a) se nachází logo aplikace a velké tlačítko *NAHLÁSIT PROBLÉM*. V horním menu na pravé straně se nachází tlačítko pro přechod do nastavení. V nastavení je možné změnit jazyk



(a) Úvodní obrazovka aplikace

(b) Formulář pro popis problému

(c) Výběr pozice problému

Obrázek 2.3: Ukázky obrazovek z aplikace Problem Report

aplikace, zadat přihlašovací údaje pro neveřejný přístup, stáhnout offline data a zjistit základní informace o aplikaci. Po stisknutí tlačítka *stáhnout offline data* není vůbec jasné co by se mělo stát nebo co má uživatel očekávat, že se stane. Po stisknutí tohoto tlačítka se pouze zobrazí verze aplikace. Ostatní tlačítka se chovají dle očekávání.

Po stisknutí tlačítka *NAHLÁSIT PROBLÉM* se zobrazí obrazovka pro výběr podatelny. Na obrazovce se zobrazí pouze možnost vybrat město Česká Lípa, proto je tato obrazovka zatím zbytečná a zdržuje tak uživatele. Po vybrání podatelny se zobrazí obrazovka s výběrem typu problému. Nachází se zde 9 typů problémů jako je například dopravní značení, veřejné osvětlení nebo hřiště a lavičky.

Po vybrání typu problému se zobrazí standardní fotoaparát na platformě Android a pro další pokračování je nutné pořídit fotografii. Nutnost pořízení fotografie je dle zkušeností autora zbytečná, protože při hlášení některých typů poruch stačí pouze označit lokaci závady a vše je jasné. Jako příklad může sloužit hlášení závady na veřejném osvětlení, kde tedy uživatel pro nahlášení poruchy musí vyfotit nesvítící lampu, která s pravděpodobností nebude vůbec vidět.

Po pořízení fotografie je zobrazena obrazovka s fotografií a tlačítky pro pořízení nové fotografie a tlačítko pro pokračování. Tato obrazovka je pravděpodobně zbytečná, protože již při pořizování fotografie je možné pořízenou fotografii smazat a pořídit novou fotografii. Tato obrazovka by dávala větší smysl, kdyby bylo možné pořídit větší množství fotografií.

Po kliknutí na tlačítko *POKRAČOVAT* na obrazovce pro potvrzení fo-

tografie se zobrazí obrazovka pro popis problému (viz. obrázek 2.3b). Nyní má možnost uživatel popsat problém a to maximálně 150 znaky. Poté vyplní kontaktní údaje, povinný údaj je pouze email. Jedná se však o nestandardní styl popisu povinných formulářových položek.

Po kliknutí na tlačítko *POKRAČOVAT* se v případě, že nebyl vyplněn email zobrazí dialog obsahující textové pole „Vyplňte email“. Pokud je vše správně vyplněno zobrazí se mapa (viz. obrázek 2.3c), kde je možné vybrat pozici závady. Na obrazovce je zobrazena pozice uživatele a tzv. zaměřovač, který má za úkol uživatel nastavit na pozici, kde chce nahlásit poruchu. Navíc u pozice uživatele je zobrazen kruh zobrazující jakou mají souřadnice GPS, získané pomocí mobilního telefonu, přesnost. Pozici poruchy je možné opět potvrdit pomocí tlačítka *POKRAČOVAT*.

Po potvrzení pozice poruchy se zobrazí obrazovka s názvem *Sumarizace*, kde jsou zobrazeny informace, které budou odeslány o problému. Po kliknutí na tlačítko *POKRAČOVAT* je problém odeslán.

2.2.2.1 Shrnutí

Klady

- Přehledná lokalizace poruchy
- Přehledná úvodní obrazovka
- Možnost vytvořit poruchu bez internetového připojení (porucha je odeslána později)

Zápory

- Uživatel musí projít sedm obrazovek než je problém odeslán
- Zbytečná obrazovka pro výběr podatelny, když zatím existuje pouze jedna podatelna
- Nutnost pořídit fotografii pro nahlášení poruchy
- Zbytečná obrazovka s právě pořízenou fotografií
- Nejasná úloha tlačítka *Stáhnout offline* data v nastavení

2.2.3 ZmapujTo

ZmapujTo je mobilní aplikace a webový portál, který slouží k hlášení podnětů. Autoři aplikace jsou Martin Hujer a Miroslav Kubásek.

Na úvodní obrazovce aplikace (viz. obrázek 2.4a) se zobrazí velké tlačítko *Nové hlášení* a informace o tom v jaké se momentálně uživatel nachází obci tzn. kam má být případné hlášení odesláno. Dále je zde také informace o tom,

že krizové události by měly být hlášeny na linku 112. V horním menu se nachází tlačítko *Poloha*, které ovšem při kliknutí neprovede žádnou reakci. Dále se také v horním menu nachází vpravo tlačítko *Info*, které zobrazí informace o aplikaci a popis jak odeslat nové hlášení. V dolní části na úvodní obrazovce se nachází další menu, kde se nachází položka *Hlášení* a položka *Nastavení*. Položka *Hlášení* zobrazí samotnou úvodní obrazovku. V nastavení je možné přednastavit kontaktní údaje, které jsou použity při hlášení.

Po kliknutí na tlačítko *Nové hlášení* se zobrazí obrazovka, kterou je možné vidět na obrázku 2.4b, kde je možné vyplnit a nastavit veškeré informace o poruše. Pro volbu místa poruchy je možné buď využít tlačítko *zaměřit*, které provede získání pozice pomocí GPS v mobilním zařízení nebo využít tlačítko *Vybrat polohu z mapy*. Po kliknutí na tlačítko *Vybrat polohu z mapy* se zobrazí mapa a po kliknutí do mapy je zde umístěna ikonka znázorňující pozici, na které chceme ohlásit problém. Dále je možné k hlášení přiřadit fotografii a to i více než jednu. Poté uživatel musí vybrat některou z kategorií, které se dané hlášení týká. Nachází se zde pět kategorií:

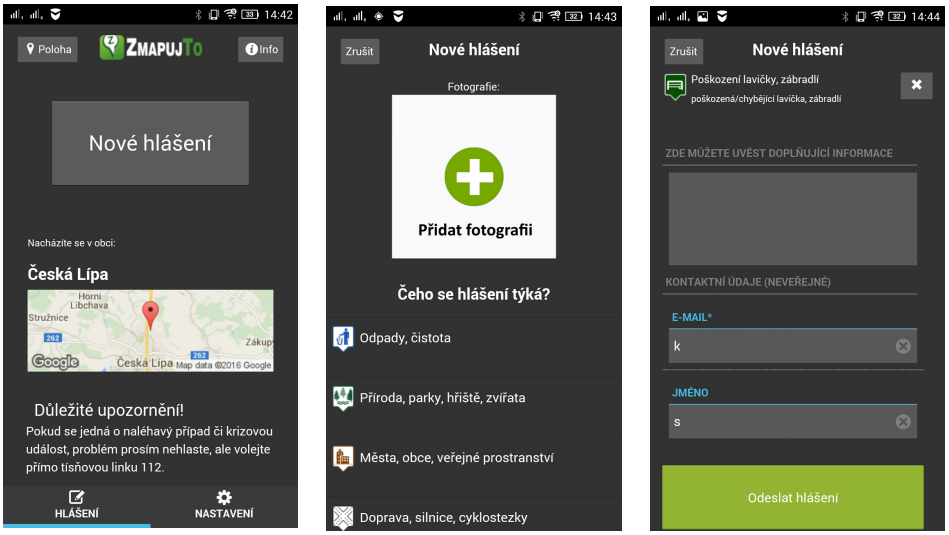
- Odpady, čistota
- Příroda, parky, hřiště, zvířata
- Města, obce, veřejné prostranství
- Doprava, silnice, cyklostezky
- Jiná hlášení

Po výběru jedné z kategorií jsou ještě zobrazeny podkategorie. Po výběru kategorie je možné doplnit doplňující informace o hlášení. Obrazovku je možné vidět na obrázku 2.4c. Na závěr uživatel musí vyplnit kontaktní údaje, kde je povinnou položkou pouze emailová adresa. Vyplněný formulář uživatel odešle kliknutím na tlačítko *Odeslat hlášení*.

2.2.3.1 Shrnutí

Klady

- není nutnost k hlášení přiřazovat fotografii
- možnost přidat i více než jednu fotografii
- nahlášení poruchy se provádí na jedné obrazovce
- přehledná úvodní obrazovka
- není nutnost zadávat do jaké obce má být hlášení odesláno



(a) Úvodní obrazovka aplikace

(b) Výběr kategorie hlášení a přiřazení fotografie

(c) Formulář pro odeslání hlášení

Obrázek 2.4: Ukázky obrazovek z aplikace ZmapujTo

Zápory

- nefungující tlačítko poloha na úvodní obrazovce
- na mapě při lokalizaci poruchy není zobrazena pozice uživatele

2.2.4 Shrnutí

Všechny testované aplikace dokáží splnit svůj cíl, tedy ohlášení závady. Závada v místě, na kterém se uživatel momentálně nachází, je nejsnadněji ohlášena pomocí aplikace ZmapujTo. Pozice uživatele se automaticky načte, uživatel tak pouze vybere vhodnou kategorii a popíše závadu. Ke zrychlení také dojde tím, že uživatel není nucen pořizovat fotografii. V případě, že uživatel chce zvolit jinou pozici než tu, na které se aktuálně nachází, je nejsnadnější tuto pozici nastavit pomocí aplikace Problem Report. Tento způsob uživatel zobrazí tzv. zaměřovačem a pohybem mapy uživatel mění pozici závady (zaměřovač je pořád uprostřed pohledu do mapy). Tento způsob volby pozice závady bude navržen i do mobilní aplikace, která je cílem této práce.

Dle analýzy stávajících řešení je určitě také dobré se v návrhu pokusit o to, aby si uživatel při procesu tvorby závady prošel co nejmenším počtem obrazovek a nebyl tak zbytečně zdržován.

Pro tvorbu závady je také dobré nenutit uživatele k pořízení fotografie, protože v některých případech to opravdu není nutné.

Zajímavou funkcí u aplikace Problem Report a Dej Tip, je možnost vytvořit závadu i bez internetového připojení a závadu odeslat až poté co je internetové připojení dostupné.

Bohužel žádná z aplikací neumožňuje zobrazit v mapě objekty, jako jsou například lampy. Zobrazení objektů může uživatelům pomoci k přesnější lokalizaci a naopak zaměstnancům úřadu k rychlejší opravě závady.

Kapitola 3

Návrh

Cílem této kapitoly je vytvoření návrhu aplikace pro hlášení závad. V této kapitole je potřeba sepsat základní požadavky, omezení aplikace a definovat uživatelské role, které budou aplikaci využívat. Poté je třeba, na základě získaných poznatků, vytvořit základní návrh uživatelského rozhraní aplikace.

3.1 Základní popis aplikace

Mobilní aplikace bude umožňovat hlášení závad, poruch a nedostatků v městech a obcích. Aplikace má za úkol posloužit občanům ke snadnějšímu hlášení závad a úředníkům při jejich řešení. Mezi nejčastější závady patří nesvítící veřejné osvětlení, poškozený odpadkový koš, poškozená vozovka nebo chodník apod.

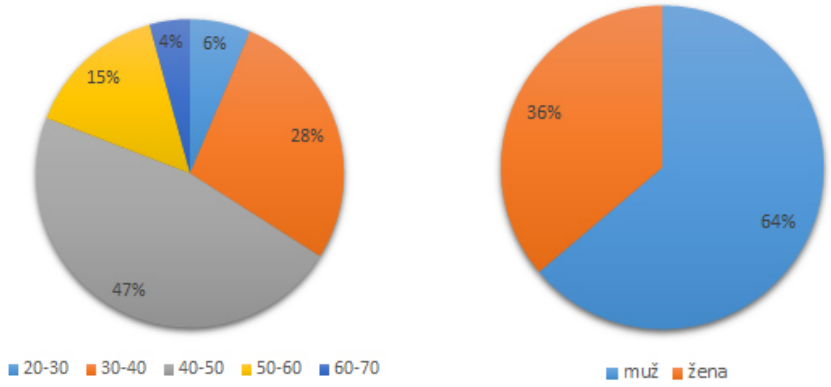
3.2 Omezení

Kvůli složitému návrhu mobilních aplikací pro více platforem, bude aplikace navržena pouze pro platformu Android. Dalším omezením je, že mobilní aplikace bude napojena na stávající aplikační rozhraní webové aplikace (viz. kapitola 1). Díky využití stávajícího API je možné, že vzniknou nové požadavky, které by v případě návrhu úplně nového API nemusely být zohledňovány.

3.3 Uživatelské role

Uživatelské role budou v této kapitole vytvořeny na základě průzkumu o uživatelích internetové aplikace pro hlášení závad (viz. kapitola 1).

Průzkum byl vytvořen na vzorku 100 lidí, kteří využívali webovou aplikaci. Jelikož do aplikace lidé zadávají své kontaktní údaje, bylo možné na základě těchto údajů zjistit pohlaví a věk osob. Díky tomu, že při zadávání závady



(a) Graf: využívání aplikace dle věkových skupin

(b) Graf: využívání aplikace dle pohlaví

Obrázek 3.1: Využívání stávající webové aplikace na hlášení závad

je vyžadován také kontaktní email, bylo možné zjistit, zda se jedná o občana, úředníka či správce veřejného osvětlení. Od systému se očekává snadné ohlášení závady. Dle popisů závad se nejčastěji jedná o závady, které jsou v blízkosti bydliště občanů nebo se vyskytují v místech, kde se daný občan často vyskytuje, jako příklad může být cesta do práce. Jedná se však pouze o odhad, tyto informace by měly být přesněji zjištěny pomocí uživatelských dotazníků (viz. kapitola 4.1).

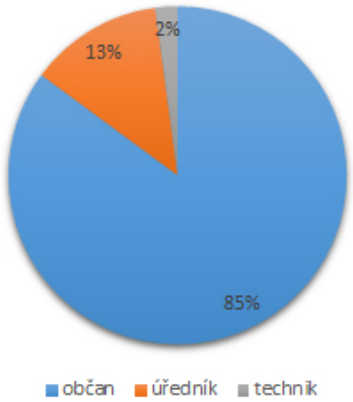
3.3.1 Zpracování uživatelského průzkumu

Na základě uživatelského průzkumu bylo zjištěno, že 47% uživatelů aplikace jsou lidé ve věkové kategorii 40-50 let. Další početnější skupinou jsou uživatelé ve věkové kategorii 30-40 let, kterých bylo v průzkumu 28%, viz. 3.1a. Co se týče pohlaví, 64% uživatelů byli muži, viz. 3.1b.

Uživatelé kteří hlásí závady jsou z 85% občané. 13% nahlášených závad bylo ohlášeno přímo zaměstnanci úřadu, kteří webovou aplikaci využili ke zrychlené komunikaci ohledně hlášení závady na veřejném osvětlení. Technik do aplikace nahlásil závadu ve 2% případů, jednalo se tak zejména o předání závady podřízenému technikovi, viz. 3.2. Dle tohoto zjištění je možné definovat tři uživatelské role: občan, úředník a technik. Z průzkumu bylo také zjištěno, že 81% občanů ohlásí jednu závadu maximálně jednou za půl roku. 19% občanů využívá stávající internetovou aplikaci alespoň jednou za 14 dní.

3.3.1.1 Občan

Role občan zastupuje uživatelskou skupinu lidí, kteří bydlí v daném městě. Jelikož se jedná o roli uživatele, pro kterého by měl být systém primárně určen, návrh by měl být zaměřen co nejvíce na tuto roli. Mobilní aplikace



Obrázek 3.2: Využívání stávající webové aplikace na hlášení závad dle uživatelských rolí

bude určena zejména pro 19% občanů, kteří hlásí závady alespoň jednou za 14 dní. Mobilní aplikace by tímto měla docílit, aby tito uživatelé hlásili více závad. Dle odhadů pro 81% občanů, kteří nehlásí závady tak často, nebude mít mobilní aplikace velký smysl.

Hlavním cílem role občan bude hlášení závad. Tito uživatelé zajisté budou chtít sledovat stav řešení ohlášených závad.

3.3.1.2 Úředník

Jedná se o roli uživatele, který má primárně za úkol spravovat stav řešení závad a případně vyřizovat řešení ohlášených závad. Sice občas může závadu dle uživatelského průzkumu vytvořit, ale není to jeho hlavní činností v systému. Tato role uživatele bude aplikaci využívat odhadem jednou týdně v závislosti na velikosti obce či města. Tato uživatelská role se většinou zúčastní krátkého školení na správu závad, kde mu budou ukázány možnosti hlášení závad.

3.3.1.3 Technik

Jedná se o roli, která má na starosti řešení nahlášených závad. Největší množství závad bude řešit technik spravující veřejné osvětlení. Technik bude aplikaci využívat odhadem jednou týdně v závislosti na velikost obce či města.

3.4 Případy užití

3.4.1 Hlášení závady

- Nahlášení závady na předem definovaném objektu (lampa, odpadkový koš...)

- Nahlášení závady na předem nedefinovaném místě (poškozená vozovka)
- Nastavení pozice závady
- Nastavení pozice závady dle pozice uživatele (GPS)
- Přidání fotografií k závadě
- Změna města pro hlášení závady

3.4.2 Seznam závad

- Procházení uživatelem ohlášených závad
- Procházení všech ohlášených závad
- Zobrazení mapy se všemi závadami

3.4.3 Detail závady

- Zobrazení informací o závadě
- Zobrazení pozice závady
- Zobrazení informací o objektu, na kterém byla závada nahlášena (pokud existuje)
- Zobrazení fotografie závady
- Změna stavu závady
- Vytvoření reakce na závadu

3.4.4 Uživatel

- Přihlášení uživatele (uživatel se může chtít přihlásit do více aplikací - více měst nebo obcí)
- Odhlášení uživatele

3.5 Task group

3.5.1 Výpisy

- zobraz závady, které ohlásil uživatel
- zobraz všechny závady

3.5.2 Formuláře

- vyber kategorii pro hlášení závady
- nahlaš závadu na předem definovaném objektu (lampa, odpadkový koš...)
- nahlaš závadu na předem nedefinovaném místě (poškozená vozovka)
- přidej fotografie k závadě
- změň stav závady
- vytvoř reakci na závadu
- přihlášení uživatele
- odhlášení uživatele

3.5.3 Detaily

- zobraz informace o závadě
- zobraz pozici závady
- zobraz fotografie závady
- zobraz informace o objektu, na kterém byla závada nahlášena (pokud existuje)

3.5.4 Mapa

- zobraz pozici závady
- zobraz mapu se všemi závadami
- nastav pozici závady

3.5.5 Menu

- změň cílové město nebo obec pro hlášení závady

3.6 Nielsenova heuristická analýza

Při návrhu uživatelského rozhraní je dobré držet se deseti následujících pravidel z Nielsenovi heuristické analýzy.

V této kapitole je čerpáno ze zdrojů [8] a [9].

Nielsenova heuristická analýza je jednou ze základních metod pro testování uživatelského rozhraní. Jedná se o seznam pravidel, které by mělo uživatelské rozhraní splňovat. Jakob Nielsen a Rolf Molich v roce 1990 vytvořili heuristiku pro heuristické vyhodnocení a poté v roce 1994 Jakob Nielsen revidoval tuto heristiku na množinu pravidel.[8]

3.6.1 Viditelnost stavu systému

Uživatel by měl být vždy systémem vhodně informován (v rozumném čase) o tom co se zrovna děje. Systém by tak měl reagovat na uživatelský vstup, nebo v případě, že se například provádí nějaký časově náročnější výpočet nebo stahování dat, tak zobrazit *progress bar*.

3.6.2 Shoda systému s realitou

Systém by měl na uživatele mluvit jazykem uživatele se slovy, frázemi a koncepty, které jsou uživateli známé, zná je z reálného světa. Neměly by se využívat pojmy, které jsou například specifické pouze pro daný systém.

3.6.3 Minimální zodpovědnost

Uživatelé se často učí nové funkce systému pomocí chyb, které provedou. Když uživatelé udělají chybu, musí mít možnost provedenou akci vrátit zpět a vrátit tak systém do předchozího stavu. V případě, že se provádí nenávratná akce, je třeba na to uživatele řádně upozornit.

3.6.4 Dodržování obecných a platformových standardů

Uživatel nesmí být zmaten různými termíny v různých situacích, přestože mají dané termíny stejný význam. Systém by měl vždy dodržovat standardy, které jsou na dané platformě. Proto je například potřeba dodržovat standardní chybové hlášky, správné umístění komponent apod. Ideální je používat standardní platformové komponenty.

3.6.5 Prevence chyb

Lepší než dobré chybové hlášky je návrh, který zabraňuje samotnému výskytu chyb. Buď je možné podmínky náchylné k chybám co nejvíce eliminovat, nebo na chyby uživatele upozornit ještě dříve než například potvrdí formulář.

3.6.6 Kouknu a vidím

Je třeba minimalizovat zatěžování paměti uživatele tím, že uživatel vždy vidí potřebné informace a akce, které může provést. Uživatel si tak například nemusí pamatovat informace z jedné části formuláře na další.

3.6.7 Flexibilita a efektivita

Systém by měl v závislosti na jeho možnostech a typu umožňovat dvě verze ovládání. Pro méně zkušené uživatele a pro zkušené uživatele.

Verze pro méně zkušené uživatele by měla obsahovat pouze základní funkce a možnosti nastavení tak, aby „nezkušený“ uživatel nebyl zbytečně zatěžován

funkcionalitami, které stejně nepotřebuje. Naopak pro zkušené uživatele by se měli zobrazit všechny funkcionality, včetně těch složitějších. Zkušenější uživatel by měl mít také případně možnost si potřebné funkcionality přizpůsobit pomocí maker.

Pro oba typy uživatelů je také dobré umožnit využívat klávesové zkratky.

3.6.8 Minimalistický design

Uživatel by měl mít co nejméně možností kam může kliknout, protože každá další možnost soutěží o pozornost uživatele. Čím méně možností uživatel má, tím rychleji je schopen pokračovat. Na obrazovce by také měly být zobrazeny pouze informace, které uživatel v dané situaci opravdu potřebuje.

3.6.9 Smysluplné chybové hlášky

Chybové hlášky by měly být v přirozeném jazyce a neměly by například obsahovat žádné chybové kódy apod. Hlášky by měly přesně popisovat co je za problém a doporučit uživateli jak pokračovat dál. Ideální je, když uživatel nemá možnost dojít do stavu, kdy je potřeba chybové hlášení.

3.6.10 Návody a dokumentace

Přestože je lepší, když je systém použitelný bez jakékoliv nápovědy, nápovědu by měl systém obsahovat. Veškeré informace by v systému měly být snadno vyhledatelné a obsahem nápovědy by měly být názorné příklady.

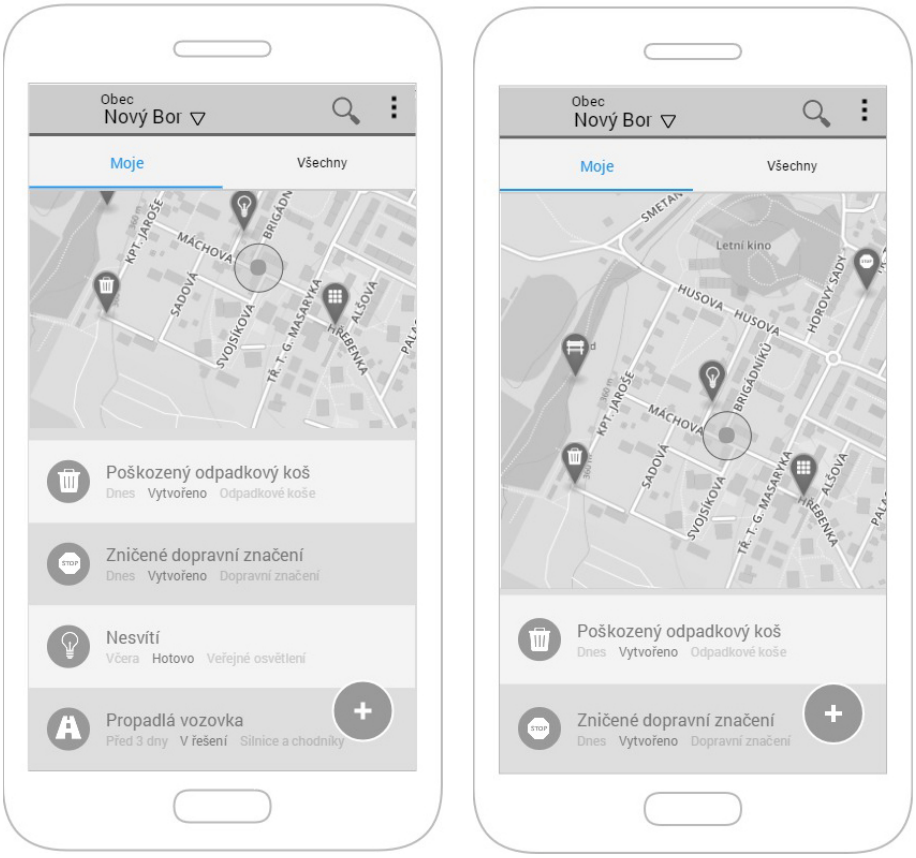
3.7 Lo-Fi prototyp

V této části budou navrženy jednotlivé obrazovky aplikace na základě požadavků získaných v předešlé kapitole. Proces tvorby návrhu obrazovek probíhal nejprve tvorbou velice jednoduché skici tužkou na papír. Na základě takto vytvořeného skici byl poté vytvořen návrh obrazovek za pomoci programu Gliffy[6].

3.7.1 Úvodní obrazovka

Úvodní obrazovka má za cíl uživateli zobrazit aktuální závady a poskytnout možnost ohlásit další závady. Dle požadavků by mělo být možné v aplikaci hlásit závady ve více obcích, proto je v aplikační liště zobrazen název obce, které se aktuální hlášení týkají. Po kliknutí na název obce se zobrazí všechny obce, ve kterých je možné aplikaci využít a je možné zvolit jinou obec, než která je aktuálně vybraná. V pravém okraji aplikační lišty se nachází ikonka „tří teček“, která slouží pro zobrazení menu, kde je možné přejít do nastavení.

[6] https://www.gliffy.com/

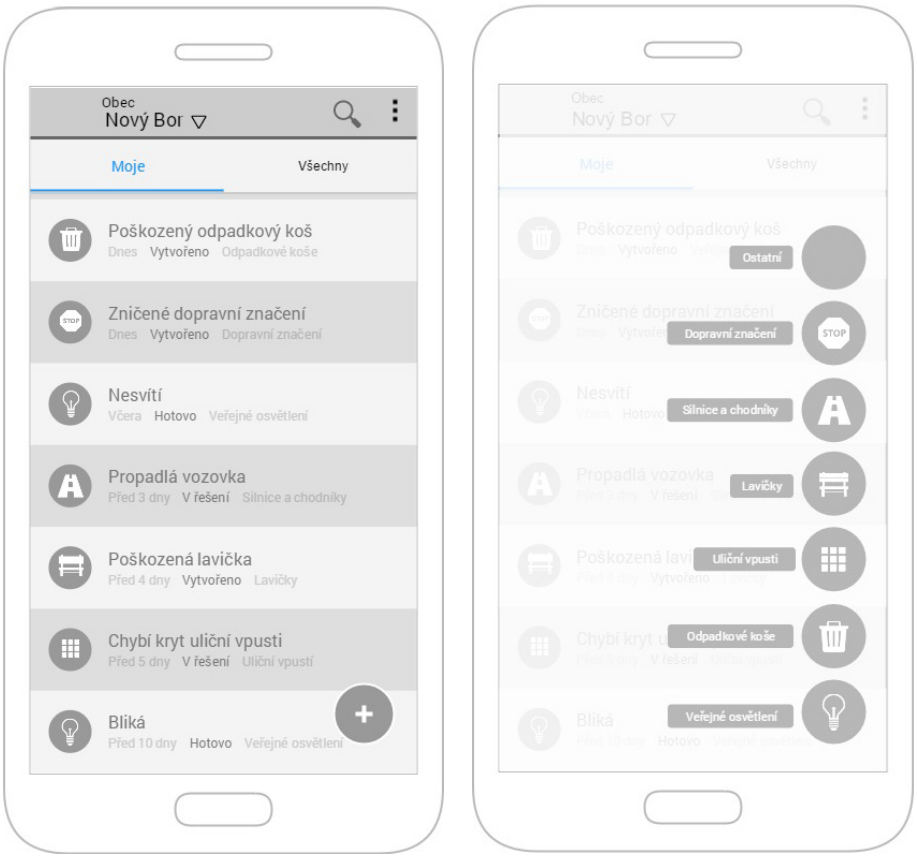


(a) Mapa a seznam (b) Velká mapa

Obrázek 3.3: Úvodní obrazovka

Pod aplikační lištou se nachází menu využívající *Tab Pattern*, viz. kapitola 2.1.5. V menu jsou zobrazeny dvě položky *Moje* a *Všechny*. Obě položky obsahují interaktivní mapu zobrazující pozici a níže seznam závad. Po kliknutí do seznamu na nějakou položku nebo do mapy na nějakou ikonku se zobrazí obrazovka s detailem o závadě, viz. obrázek 3.8. Mapa kromě informace o tom na jaké pozici se závada nachází, zobrazuje také pomocí ikonky to, v jakém se závada nachází stavu a o jakou kategorii závad se jedná. Úvodní obrazovka se může nacházet ve třech stavech:

- 50% obsahu zabírá mapa a 50% zabírá seznam závad (počáteční stav), viz. obrázek 3.3a
- 100% obsahu zabírá seznam závad, viz. obrázek 3.4a
- 66% obsahu zabírá mapa a 34% zabírá seznam závad, viz. obrázek 3.3b



(a) Seznam (b) Výběr kategorie pro přidání závady

Obrázek 3.4: Úvodní obrazovka

Mezi popsanými stavy je možné pohybovat se pohybem prstu směrem nahoru nebo dolů.

V případě že je v menu vybrána položka *Moje*, zobrazí se všechny závady, které byly nahlášeny z daného telefonu. Může tak vznikat otázka proč nejsou zobrazeny všechny závady aktuálně přihlášeného uživatele. Je to z důvodu, že webová aplikace k hlášení závad (které je využíváno API) nevyžaduje po uživatelích, aby byli při hlášení závady přihlášeni. Důvodem je, aby návštěvníci stránek nebyli zbytečně odrazováni od ohlášení nové závady. Proto se jako nejrozumnější řešení jeví mít v telefonu uložené závady, které byly ohlášeny pomocí daného zařízení a ty poté zobrazovat na záložce *Moje*. V případě že je v menu vybrána položka *Všechny* zobrazí se všechny ohlášené závady dle data jejich poslední změny.

V situaci, kdy uživatel zatím nevytvořil žádnou závadu, se na záložce *Moje* nezobrazí mapa a seznam závad, ale pouze informační text, který bude uživatele vyzývat k vytvoření první závady. Stejná situace bude na záložce *Všechny*

(a) Výběr objektu v mapě (b) Objekt vybrán v mapě

Obrázek 3.5: Obrazovka pro výběr objektu v mapě

v případě, že v aplikaci nebudou nahlášeny ještě žádné závady.

V pravé dolní části obrazovky se nachází tlačítko *FAB* (viz. kapitola 2.1.2.4), po jehož aktivaci se zobrazí seznam kategorií do kterých je možné v dané obci nebo městě hlásit závady. Ukázková obrazovka se nachází na obrázku 3.4b. Po kliknutí na tlačítko pro danou kategorii se dle typu kategorie zobrazí některá z obrazovek, které jsou popsány v kapitolách 3.7.2 a 3.7.3.

3.7.2 Obrazovka pro výběr objektu, na kterém má být závada ohlášena

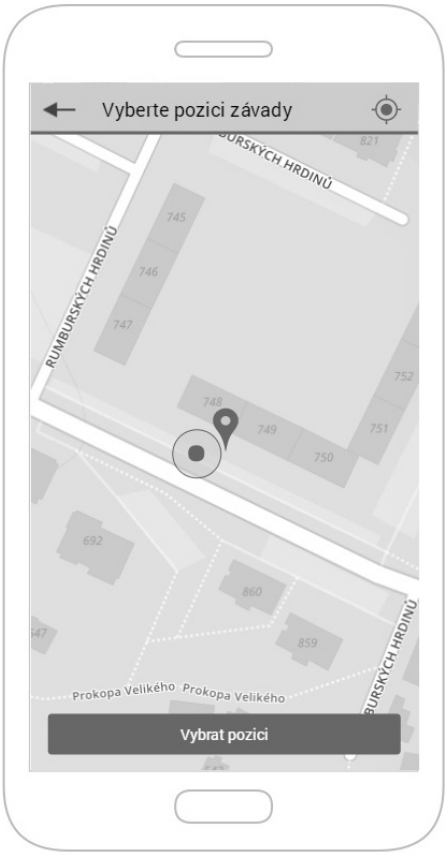
Tato obrazovka je zobrazena v případě, že uživatel chce vytvořit novou závadu a kategorie závady je nastavena tak, že závada má být ohlášena přímo na některém z objektů z dané kategorie. Jako příklad může být kategorie veřejného osvětlení, kdy správce veřejného osvětlení potřebuje vědět přesně o jakou lampu se jedná.

Na aplikační listě této obrazovky se nachází dvě tlačítka. V levé části je tlačítko zpět, které slouží k přechodu na úvodní obrazovku. V pravé části se nachází tlačítko pro nastavení středu mapy na pozici uživatele.

Pod aplikační lištou se nachází mapa, kde jsou zobrazeny objekty, na kterých je možné ohlásit závadu. Kategorie je uživatelem předem vybraná. Na mapě se nachází také ikonka zobrazující pozici uživatele a přesnost, se kterou byla pozice zjištěna. Přesnost je standardně zobrazena pomocí průhledného kruhu, kdy čím je kruh větší, tím větší je nepřesnost.

Po kliknutí na některý objekt v mapě se zobrazí v dolní části panel s názvem objektu a tlačítko na přidání *Nahlásit závadu*. Po kliknutí na tlačítko *Nahlásit závadu* se zobrazí obrazovka pro vytvoření závady 3.7.4.

3.7.3 Obrazovka pro výběr pozice závady



Obrázek 3.6: Obrazovka pro výběr pozice v mapě

Tato obrazovka je zobrazena v případě, že uživatel chce vytvořit závadu pro kategorii, která je nastavena tak, že závada může být ohlášena na jakékoliv

pozici. Jako příklad této kategorie může být kategorie veřejné komunikace a chodníky.

Na aplikační listě této obrazovky se nachází dvě tlačítka. V levé části je tlačítko zpět, které slouží k přechodu na úvodní obrazovku. V pravé části se nachází tlačítko pro nastavení středu mapy na pozici uživatele.

Pod aplikační lištou se nachází mapa. Jelikož má uživatel za úkol nastavit pozici v mapě, kde má být ohlášena závada, byly zvažovány dvě metody jak požadovaného cíle dosáhnout. Jednou z metod je, že uživatel musí kliknout do místa v mapě, kde se umístí ikonka. S využitím této metody nastává problém s citlivostí displeje resp. velikostí prstu uživatele, kdy často nedochází k umístění ikonky na požadované místo. Tato metoda se hodí spíše do webových aplikací. Další metodou je, že se ikonka stále nachází uprostřed mapového okna a při pohybu mapou ikonka neustále zůstává uprostřed okna, ale pohybuje se po mapě. Tímto je velice snadné pro uživatele přesně nastavit pozici. Tato metoda je využívána v mobilní aplikaci Mapy.cz od Seznam.cz a také v aplikaci *Problem Report* popsané v kapitole 2.2.2. Pro návrh aplikace byla využita druhá metoda, tedy metoda, kdy se ikonka neustále nachází uprostřed fragmentu.

Na mapě se dále také nachází ikonka zobrazující pozici uživatele a přesnost, se kterou byla pozice zjištěna. Přesnost je standardně zobrazena pomocí průhledného kruhu, kdy čím je kruh větší, tím větší je nepřesnost.

V dolní částí se nachází tlačítko sloužící pro potvrzení vybrané pozice.

3.7.4 Obrazovka pro vytvoření závady

Tato obrazovka se zobrazí v případě, že uživatel chce vytvořit závadu a už si prošel některou z obrazovek na obrázku 3.6 nebo 3.5.

Na aplikační listě této obrazovky se nachází tlačítko zpět, které podle situace uživatele vrátí buď na obrazovku pro výběr objektu (viz. kapitola 3.7.2) nebo na obrazovku pro výběr pozice závady (viz. kapitola 3.7.3).

Pod aplikační lištou se nachází formulář pro vytvoření závady. Ve formuláři jsou vstupní pole pro definici závady, přesnější definici závady a poté kontaktní údaje na osobu, která danou závadu ohlašuje. Část formuláře pro zadání kontaktních údajů bude nutné vyplňovat pouze při prvním hlášení závady na daném zařízení. Při dalším hlášení se již nebudou vstupní pole pro kontaktní informace zobrazovat, ale zobrazí se pouze jméno osoby a vedle něj ikonka tužky. Po kliknutí na ikonku tužky se zobrazí vstupní pole kde je možné všechny kontaktní údaje změnit.

Pod kontaktními údaji se nachází ještě ikonka pro přidání fotografie. Po kliknutí na tuto ikonku se spustí aplikace fotoaparát v zařízení. Po pořízení fotografie se fotografie zobrazí vedle ikonky pro přidání fotografie. Jakmile je k dané závadě pořízeno více fotografií, než které je možné zobrazit vedle sebe na šířku obrazovky, zobrazí se odkaz s popiskem „+ N FOTEK“. Po kliknutí

(a) Horní část formuláře (b) Horní část formuláře

Obrázek 3.7: Obrazovka pro vytvoření závady

na tento popisek nebo na některou z fotografií se zobrazí všechny fotografie a je také možné některou z fotografií odstranit.

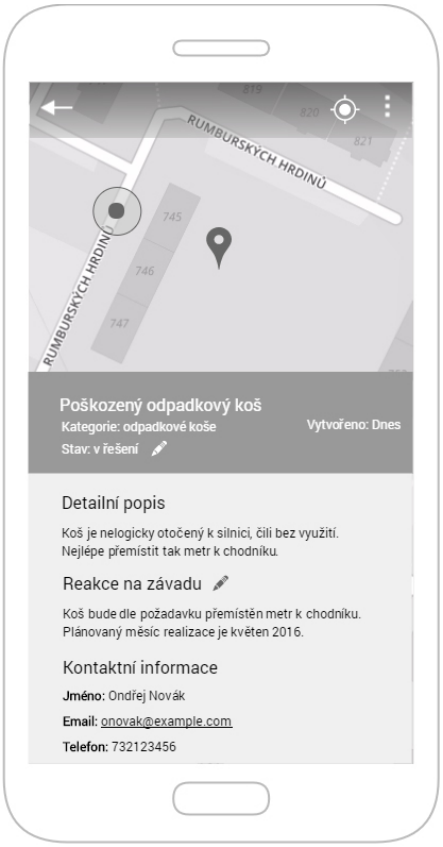
Na konci formuláře se nachází tlačítko *Odeslat*, které odešle vytvořenou závadu.

3.7.5 Obrazovka s detailem o závadě

Tato obrazovka se zobrazí v případě, že uživatel na úvodní stránce klikne v seznamu na název závady nebo v mapě klikne na ikonku dané závady.

V aplikační liště je zobrazeno tlačítko *zpět*, které uživatele vrátí zpět na úvodní obrazovku. V pravé části lišty se nachází tlačítko pro nastavení pozice mapy na pozici uživatele a ikonka pro zobrazení menu. V menu je možné nastavit vrstvu, která má být v mapě zobrazena.

Pod aplikační lištou se nachází mapa, na které je zobrazena pozice závady a pozice uživatele. Pozice uživatele je zobrazena pomocí ikonky, kde se navíc



Obrázek 3.8: Obrazovka s detailem o závadě

ikonka mění podle toho, jakou mají přesnost získané GPS souřadnice zařízení uživatele.

Pod mapou je zobrazen titulek závady obsahující název, kategorii, stav řešení závady a datum vytvoření závady. U stavu řešení závady se nachází ikonka tužky (pouze pokud je přihlášen a má potřebná práva), na kterou když uživatel klikne tak se mu zobrazí možnost změnit stav závady.

Pod titulkem závady se nachází detailní popis závady (pokud je vyplněn), reakce na poruchu (pokud je vyplněna) a kontaktní informace. V případě, že je uživatel přihlášen a má potřebná práva zobrazí se mu u reakce na poruchu ikonka tužky. Po kliknutí na tuto tužku se uživateli zobrazí textové pole, ve kterém může reakci na závadu vytvořit nebo změnit.

3.8 Závěr

V této kapitole byly sepsány požadavky na vytvoření mobilní aplikace na hlášení závad. Pro potřeby definice uživatelských skupin a rolí byl proveden průzkum uživatelů stávající webové aplikace na hlášení závad. Na základě

průzkumu bylo zjištěno, že cílovou skupinou pro mobilní aplikaci by měli být občané, kteří hlásí závady minimálně jednou za 14 dní. Dle výsledků bylo také zjištěno, že se jedná o 19% uživatelů stávající webové aplikace na hlášení závad.

Na základě průzkumu uživatelů a sebraných požadavků byl vytvořen prvotní návrh uživatelského rozhraní aplikace. Tento návrh byl nejprve vytvořen formou skici na papír a poté byl vytvořen detailnější návrh pomocí aplikace *Gliffy*[7].

Provedený návrh obsahuje detailní popis jednotlivých prvků a funkcionalit na navržených obrazovkách.

[7] https://www.gliffy.com/

Kapitola 4

Testování Lo-Fi prototypu

Cílem této kapitoly je provést testování prototypu ve fázi Lo-Fi, které má pomoci ke zlepšení vytvořeného návrhu uživatelského rozhraní, provedeného v kapitole 3.7.

První část kapitoly se bude zabývat tvorbou a vyhodnocením uživatelského dotazníku. Uživatelský dotazník bude vytvořen za účelem nalezení vhodných kandidátů na testování a zjištění některých očekávání, které uživatelé systému na hlášení závad mají.

V další části této kapitoly bude proveden vzdálený test některých navržených obrazovek. Účelem tohoto testu bude otestovat některé obrazovky na co největším počtu osob.

V poslední části kapitoly bude provedeno uživatelské testování. Na základě výsledků z tohoto testování budou provedeny jednotlivé změny návrhu uživatelského rozhraní. Dle zjištěných výsledků budou dle potřeby provedeny další testy.

4.1 Uživatelský dotazník

Uživatelský dotazník byl rozdělen do tří následujících oddílu. V prvním oddílu se nachází obecné informace o respondentovi. Dále následuje oddíl, který má sloužit pro obecné zjištění o využitelnosti systémů na hlášení závad. V poslední oddílu se nachází několik prototypů obrazovky, kde má respondent za úkol dát zpětnou vazbu na návrh některých obrazovek.

4.1.1 Obecné informace o respondentovi

Tento oddíl má za úkol zjistit obecné informace o respondentovi jako je věk, pohlaví, zkušenosti s internetem a o využívaném mobilním telefonu. Tyto informace pak pomohou k výběru vhodných kandidátů na uživatelské testování.

4.1.1.1 Obecné informace o využívání aplikací na hlášení závad

V dalším oddílu následují zjištění obecných informací o využívání aplikací na hlášení závad. Zde se nachází otázky:

- jaký je pro uživatele nejpohodlnější způsob hlášení závad
- co uživatele vede k nahlášení závady
- jaké výhody vidí uživatelé ve stávajících aplikacích na hlášení závad
- chtějí být uživatelé informováni o stavu řešení jimi nahlášených závad

Cílem této části formuláře je zjistit, proč a jak by uživatelé mobilní aplikaci využívali a zda by vůbec měli o danou aplikaci zájem.

4.1.2 Zpětná vazba na prototypy

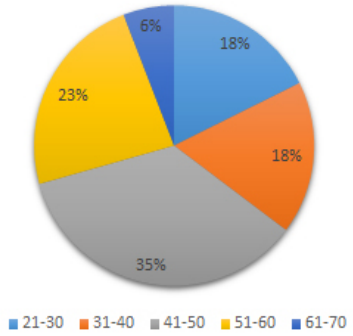
V poslední části uživatelského dotazníku se nachází tři prototypy úvodní obrazovky (viz. obrázky 3.3a, 3.3b, 3.4a), kde má respondent za úkol vybrat, která z obrazovek je dle něj nejvhodnější. Tato otázka má za úkol odhalit zda má smysl implementovat funkcionalitu možnosti přecházení mezi různými druhy zobrazení na úvodní obrazovce, popsané v kapitole 3.7.1. Následuje otázka na prototyp formuláře pro přidání závady. Jedná se o otevřenou otázku, která má za úkol zjistit, zda jsou všechny pole ve formuláři srozumitelně popsány, aby uživatel neměl problém formulář vyplnit. Tato otázka byla do formuláře zařazena, protože při uživatelském testování nebude možné v prototypu vyplňovat formulářová pole.

4.1.3 Výsledky

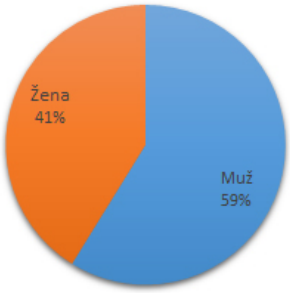
Cílem tohoto průzkumu bylo získat informace o tom, jaké preference a požadavky mají uživatelé. Data získaná z vyplněných dotazníků pak pomohou mimo jiné zvolit uživatele vhodné pro uživatelské testování.

Uživatelský dotazník byl vyplněn 40 osobami. Věkové rozdělení a rozložení dle pohlaví účastníků v testu je možné vidět na grafech 4.1a a 4.1b. Nejpočetnější skupinou účastníků byla skupina uživatelů mezi 41-50 lety. Tento údaj je zejména důležitý, protože uživatelé z této věkové kategorie jsou i nejčastějšími uživateli stávající internetové aplikace na hlášení závad (viz. 3.3.1).

Zajímavostí získanou při průzkumu jsou údaje získané z otázky *Jaký způsob k nahlášení závady je nejpohodlnější.* Největší skupina označila jako nejpohodlnější nahlásit závadu pomocí webové aplikace. Čtvrtina účastníků průzkumu označila jako nejpohodlnější nahlásit závadu pomocí mobilní aplikace, což je zejména dobrá zpráva z důvodu využitelnosti vytvářené aplikace. Okolo 16% účastníků testu označilo, že je pro ně nejsnadnější způsob ohlásit závadu

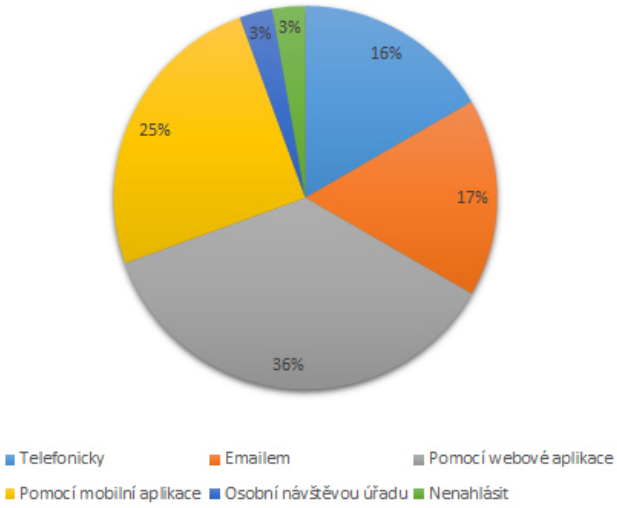


(a) Graf: rozdělení dle věku



(b) Graf: rozdělení dle pohlaví

Obrázek 4.1: Grafy rozdělení dle pohlaví a věku

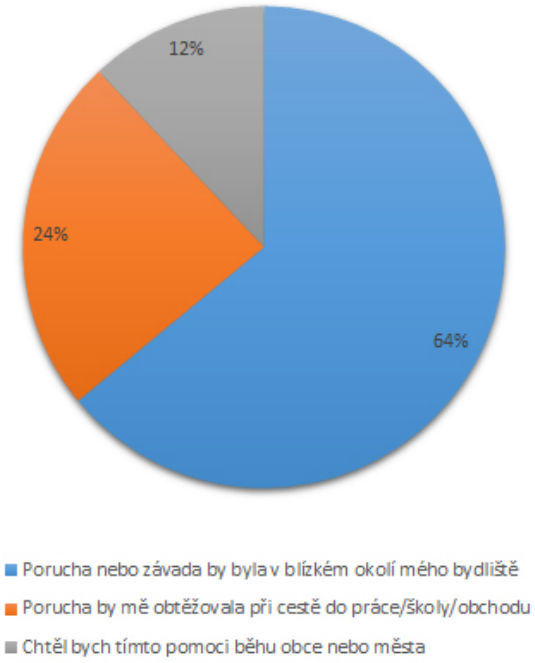


Obrázek 4.2: Graf: Nejpohodlnější způsoby hlášení závad

pomocí mobilního telefonu a stejné procento účastníků označilo za nejsnadnější způsob email. Výsledný graf je možné vidět na 4.2.

Dalšími získanými informacemi z průzkumu je zpětná vazba získaná na otázku *Co by Vás vedlo k nahlášení závady na obecním majetku?*. Bylo zjištěno, že 64% osob zúčastněných v průzkumu, by využilo aplikaci na hlášení závady v okolí svého bydliště. 24% osob v průzkumu by nahlásilo závadu, která by osobu obtěžovala při cestě do práce, školy nebo obchodu. Výsledky je možné vidět na grafu 4.3. Toto zjištění potvrzuje předpoklad, který byl vytvořen v kapitole 3.3.

V průzkumu se také objevila otázka na to, zda by uživatele zajímala informace o tom, jak se jimi ohlášená závada aktuálně řeší. Všichni uživatelé



Obrázek 4.3: Graf: Co by občana vedlo k nahlášení závady

označili odpověď *Ano*. Toto zjištění ukázalo, že bylo vhodné do mobilní aplikace implementovat funkci notifikací, kdy by uživatel byl informován o tom, že se změnil stav řešení závady.

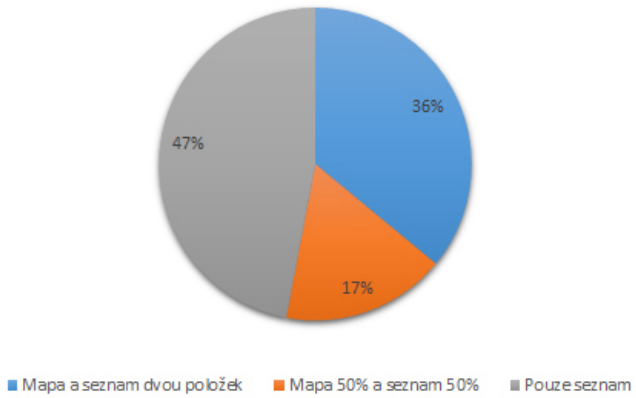
V závěrečné části dotazníku pro uživatelský průzkum se nacházela otázka, zda jsou všechny informace na obrazovce na obrázku 3.7a pro uživatele přehledné, protože tato skutečnost nebyla testována při uživatelském testování Lo-Fi prototypu. Osoby zúčastněné v testu na tuto otázku vždy odpověděli *Ano*.

Další otázkou v závěrečné části formuláře je otázka, které zobrazení by uživatelé preferovali pro zobrazení závad. Uživatelé měli na výběr tři možnosti:

- obrazovka (viz. obrázek 3.3b) - Mapa a seznam dvou položek
- obrazovka (viz. obrázek 3.3a) - Mapa 50% a seznam 50%
- obrazovka (viz. obrázek 3.4a) - Pouze seznam

Nejpreferovanějším způsobem zobrazení je dle průzkumu zobrazení pouze seznamu závad. Tuto možnost označilo 47% osob v průzkumu. Dalším způsobem k zobrazení závad byla možnost zobrazit mapu s dvou položkovým seznamem závad. Tento způsob zvolilo 36% osob. Kompletní výsledky jsou zobrazeny v grafu 4.4. Jelikož jsou výsledky zobrazení mezi mapou se seznamem dvou položek a pouze se seznamem položek rozdílné jen o 11%, je určitě dobré tyto

dvě možnosti v aplikaci ponechat. Uživatel tak bude mít možnost si vybrat to co mu více vyhovuje.



Obrázek 4.4: Graf: Preferovaný způsob zobrazení nahlášených závad

4.2 Klikací test

Na první fázi testování Lo-Fi prototypu byl také navrhnut tzv. klikací test, který má za úkol odhalit, zda uživatelé nebudou mít problém ovládat finální aplikaci. Klikací test je test, kdy je uživateli zobrazen prototyp obrazovky a je mu zadán úkol. Jako příklad úkolu může být: otevřete detail nabídky. Výsledkem tohoto testu je možnost sledovat kam nejčastěji uživatelé klikají, pokud chtějí udělat danou věc. Tento test se provádí vzdáleně a je tak možné, aby se ho zúčastnilo velké množství osob. Pro tento test bylo využito aplikace Verify App[8], protože tato aplikace poskytuje potřebné prostředky pro vzdálené testování. V aplikaci je také možné sledovat například to, jak dlouho uživatelům trvá než kliknou do prototypu. Pomocí této aplikace bylo testováno 6 prototypů obrazovek.

4.2.1 Návrh

V prvním testu se uživateli zobrazí obrazovka detailu závady (viz. obrázek 3.8). Uživatel zde má za úkol kliknout do mapy a označit místo, kde mu mobilní zařízení ukazuje, že se momentálně nachází. Tento test má za úkol odhalit zda ikonka pozice uživatele je dostatečně viditelná a nedochází tak k záměně s ikonkou, která ukazuje pozici závady.

Další test se skládá z posloupností 4 obrazovek a uživatel má za úkol ohlásit novou závadu. Na každé obrazovce se tak uživatel musí rozhodnout jak by pokračoval dál. Úkol má uživatel následující: „Nahlaste závadu, nesvítící

[8] http://verifyapp.com/

lampa, na lampě, která se nachází v ulici Rumburských hrdinů u chodníku před domem s číslem popisným 746.“ V tomto testu je uživateli nejprve zobrazena úvodní obrazovka, poté výběr kategorie hlášení závady a na závěr obrazovka pro výběr lampy a potvrzení výběru lampy v mapě.

Na úvodní obrazovce (viz. obrázek 3.3a) by měl uživatel kliknout na tlačítko FAB (viz. kapitola 2.1.2.4), které slouží pro přidání nové závady. Tento test by měl odhalit, zda je ikonka umístěná v tlačítku FAB dostatečně jasná a uživatel ví co se po kliknutí na tuto obrazovku stane.

Následuje obrazovka pro výběr kategorie (viz. obrázek 3.4b). Zde by pravděpodobně uživatel neměl mít problém s kliknutím na správnou ikonku. Tato obrazovka byla do testu zařazena z toho důvodu, aby si uživatel prošel téměř celým procesem ohlášení závady.

Další obrazovkou je obrazovka pro výběr objektu v mapě (v tomto případě lampy). Obrazovku je možné vidět na obrázku 3.5a. V tomto případě by uživatel měl kliknout na správnou ikonku lampy v mapě. Test by měl rozpoznat zda uživatel vůbec chápe co by měl dělat. U tohoto testu bude velice důležité sledovat zejména čas jak dlouho trvalo uživateli pochopit co má dělat.

Následuje obrazovka (viz. obrázek 3.5b), která je téměř stejná jako předešlá obrazovka, ale již je na ní vybrána požadovaná lampa. Po uživateli je vyžadováno, aby pouze potvrdil svůj výběr kliknutím na tlačítko nahlásit závadu. Uživatel tak musí zaregistrovat, že je již ve stavu, kdy má vybranou správnou lampu a stačí mu tak pouze svůj výběr potvrdit.

Na poslední obrazovce se zobrazí uživateli mapa s označenou pozicí, kde se uživatel nachází (viz. obrázek 3.6) a tlačítko na vybrání dané pozice. Uživatel má za úkol nahlásit závadu na místě, kde se právě nachází. Uživatel navíc má informaci, že již absolvoval kroky pro výběr kategorie závady. Cílem je tedy, aby uživatel pouze potvrdil přednastavenou pozici, kliknutím na tlačítko *vybrat pozici*.

4.2.2 Pilotní test

Navržený test byl nejprve otestován na pěti lidech, aby bylo zajištěno, že je test dobře navržen. Při testování byly zjištěny velké nedostatky při testu, kdy měl uživatel projít čtyřmi kroky. Uživatelům trvalo delší dobu než vůbec pochopili co mají udělat a jelikož úkol byl poněkud delší, trvalo jim dlouho než si ho vůbec pročetli. Často tak uživatelé klikli do míst, kde ani nebylo žádné tlačítko, protože nevěděli jak dál. Problém byl možná způsoben také tím, že aplikace na testování nejprve zobrazí zadání a poté po potvrzení zadání se zobrazí prototyp obrazovky, kde už uživatel nevidí co je jeho úkolem. Proto pro další návrh je určitě potřeba zohlednit, že úkol by se neměl vztahovat na více obrazovek a měl by být co nejkratší. Dalším problémem testu bylo, že obrázky prototypu byly příliš velké a tak se na menších monitorech nevešli celé na jednu obrazovku a uživatel musel rolovat dolů.

4.2.3 Finální test

Na základě poznatků získaných z pilotního testu byly navrženy potřebné úpravy. Nejprve bylo potřeba všechny prototypy obrazovek zmenšit, aby se vešly i na menší monitory. Dalším úkolem bylo zkrátit úkoly, aby byly pro uživatele zapamatovatelné.

Největší změny v popisu úkolů byly provedeny na obrazovkách. Úmyslem bylo, aby si uživatel prošel celým procesem vytváření závady.

Na úvodní obrazovce (viz. obrázek 3.3a) byl úkol přepsán následovně: „Na následující obrazovce klikněte do místa, kde byste očekávali, že bude možné vytvořit novou závadu“. Úkol má pořád stejný cíl jaký byl v prvotním návrhu, ale vztahuje se pouze na jednu obrazovku. Cílem zůstává kliknout na tlačítko FAB pro vytvoření nové závady.

Na obrazovce pro výběr kategorie (viz. obrázek 3.4b) má uživatel následující úkol: „Klikněte na tlačítko pro vytvoření závady na veřejném osvětlení“. Jedná se o velice jednoduchý úkol, se kterým by pravděpodobně neměli uživatelé mít žádné problémy.

Na obrazovce pro výběr objektu v mapě (viz. obrázek 3.5a) musel být opět úkol přepsán tak, aby se daný úkol vztahoval pouze k jedné obrazovce. Tento úkol byl nyní formulován následovně: „Na následující obrazovce vyberte lampu u č.p. 746 v ulici Rumburských hrdinů“. Uživatel by tak měl kliknout na správný objekt umístěný v mapě.

Další obrazovkou je obrazovka pro výběr objektu v mapě, kdy je již nějaký objekt vybrán. Úkol pro tuto obrazovku byl přepsán tak, aby uživatel navázal na úkol, který prováděl o krok dříve. Úkol je tedy následující: „Na předchozí obrazovce jste vybrali lampu u č.p. 746 v ulici Rumburských hrdinů. Nyní svůj výběr potvrďte.“ Cílem testu je zjištění zda si uživatel všiml, že dole na obrazovce přibyl panel obsahující název objektu a tlačítko *Nahlásit závadu*, na které by uživatel měl kliknout.

Poslední obrazovka v testu je obrazovka, která slouží pro výběr pozice závady v mapě. Úkol při prvotní návrhu byl následující: „Nyní chcete ohlásit, že na místě kde se momentálně nacházíte je spadlý strom“. Tento úkol byl přepsán tak, aby uživatel dostal jasný úkol a nebyl tak zbytečně zatěžován nějakou situací, ve které se nachází. Úkol by nyní měl být jasnější: „Na následující obrazovce ohlaste závadu na místě, na kterém se momentálně nacházíte.“

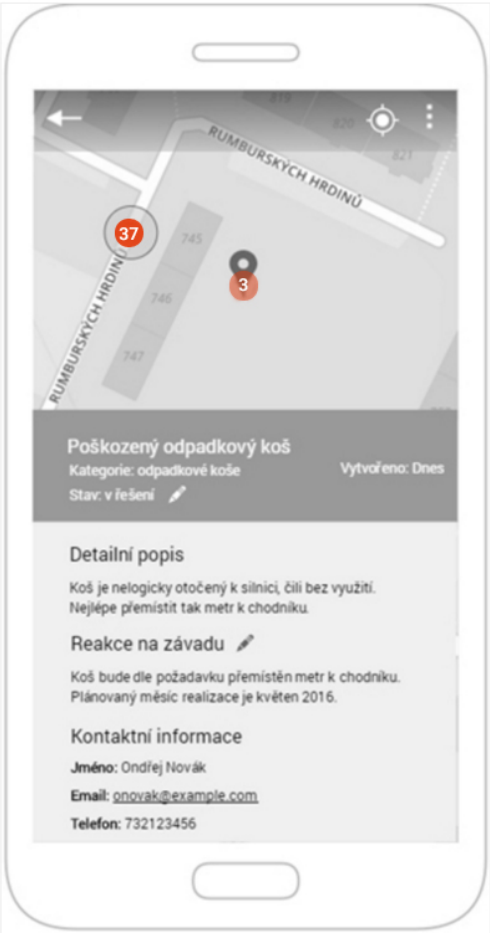
Nově upravený návrh byl otestován na třech osobách a test již proběhl v pořádku a nedocházelo tak k problémům, které se nacházely v předchozím návrhu.

4.2.3.1 Problematika testování

Testování pomocí tzv. klikacího testu je pro uživatele obtížné, zejména proto, že uživateli je zobrazena vždy jedna obrazovka a uživatel tak nezná celý kon-

text, jak by se k dané obrazovce dostal. Dalším problémem je, že navržené obrazovky nejsou barevné, čímž můžou některé prvky zanikat.

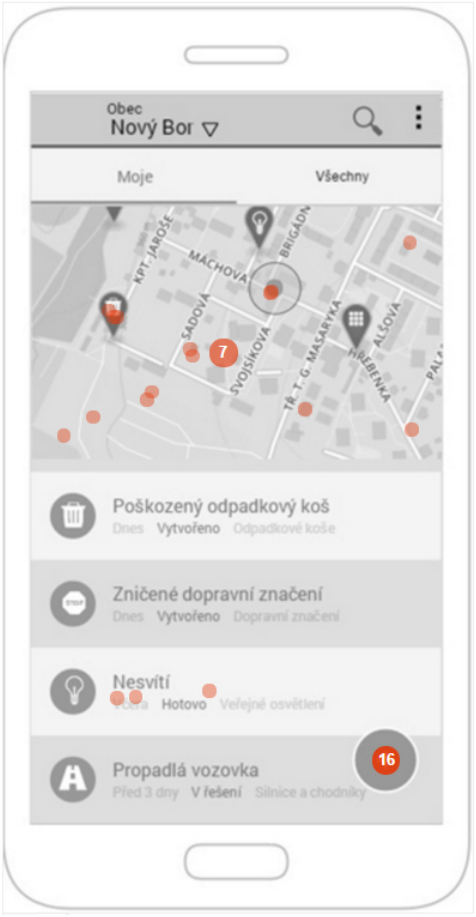
4.2.4 Vyhodnocení



Obrázek 4.5: Obr: Výsledky klikacího testu pro výběr pozice uživatele

Klikacího testu se zúčastnilo 40 stejných osob, kteří vyplnili uživatelský dotazník.

V prvním úkolu měli testování uživatelé za úkol kliknout do mapy, do místa, kde mobilní zařízení ukazuje, že se aktuálně nacházejí. 92% uživatelů kliklo správně, tedy do místa, kde byla opravdu zobrazena ikonka pozice uživatele. 8% uživatelů kliklo na místo, kde se nachází ikonka závady. Tento test ukázal, že ikonka zobrazující pozici uživatele je dostatečně výrazná a většina uživatelů od ní očekává co je i jejím úkolem. Přesné výsledky je možné vidět na obrázku 4.5.

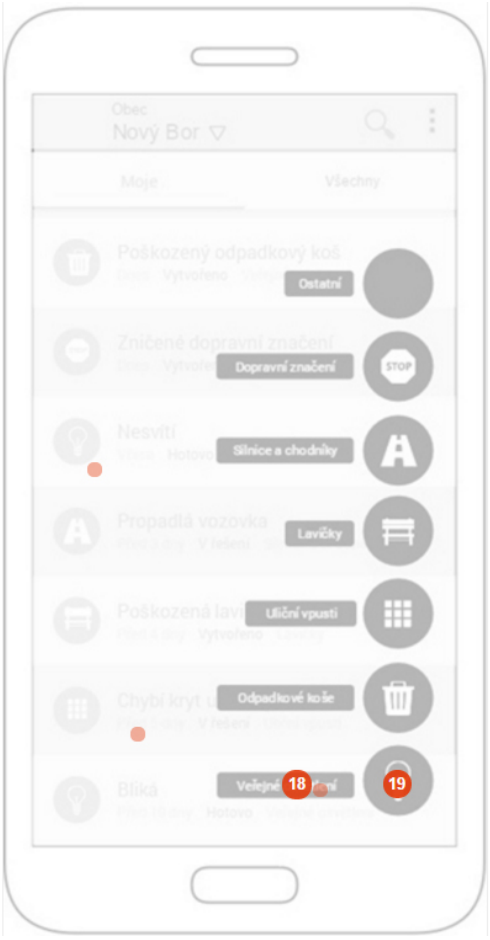


Obrázek 4.6: Obr: Výsledky klikacího testu na úvodní obrazovce

V následujícím úkolu, kdy měl uživatel na úvodní obrazovce kliknout do místa, kde bude možné vytvořit novou závadu, se vyskytly nedostatky návrhu úvodní obrazovky. 40% testovaných uživatelů sice kliklo správně na tlačítko FAB pro vytvoření závady. 52% testovaných osob však pravděpodobně nepochopilo co je obsahem úvodní stránky a tak libovolně klikli do nějakého místa v mapě (nejčastěji do středu mapy). 3 z testovaných uživatelů klikli do seznamu závad. Přestože finální test byl nejprve otestován na třech osobách, je možné, že byl opět špatně navržen zadaný úkol. Některé testované osoby tak pravděpodobně mohly pochopit úkol následovně: *Klikněte do mapy na pozici kde je možné vytvořit závadu.*. Zda byl opět špatně navržen testovací úkol, nebo zda se jedná opravdu o špatný návrh úvodní obrazovky tedy rozhodne uživatelské testování, kde je možné lépe sledovat uživatele a případně také získat lepší zpětnou vazbu. Informaci o tom, kam uživatelé klikali, je možné vidět na obrázku 4.6.

V dalším testu měl testovaný uživatel za úkol vybrat pouze správnou ka-

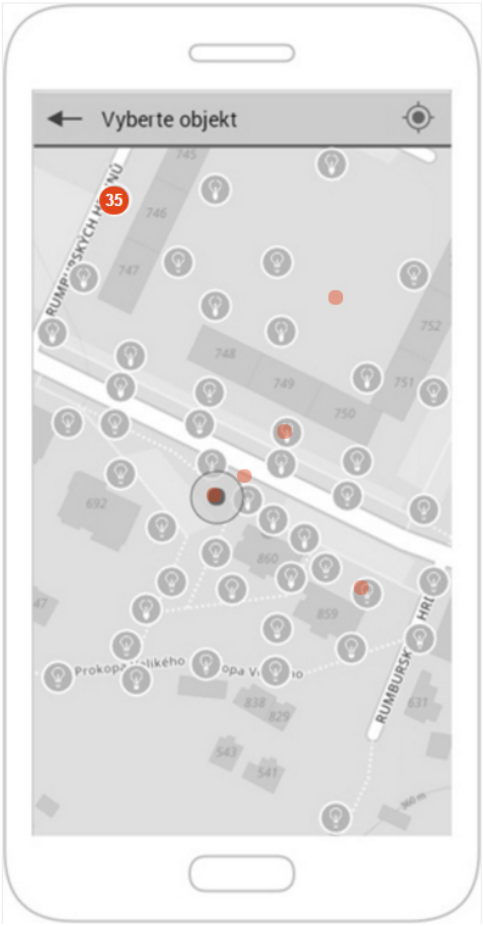
tegorii závady. Uživatelé klikali správně buď na ikonku kategorie závady nebo přímo na nápis závady. V tomto testu došlo pravděpodobně k několika překlepům, protože dva testovaní uživatelé klikli do obrazovky, kde se nic nenacházelo. Tato obrazovka by měla být dle tohoto testu správně navržena. Výsledky testu je možné vidět na obrázku 4.7.



Obrázek 4.7: Obr: Výsledky klikacího testu při výběru kategorie

Při testu na obrazovce pro výběr porouchané lampy, měli uživatelé za úkol vybrat konkretní lampu v mapě. 87% testovaných osob tento úkol zvládlo správně. 13% uživatelů kliklo sice spatně, ale velice často se jednalo o kliknutí do místa, kde se nic nenacházelo, což může znamenat, že se jednalo pouze o nechtěný klik. Tato obrazovka se tedy zdá být podle tohoto testu v pořádku. Výsledky je možné vidět na obrázku 4.8.

Po vybrání porouchaného objektu je uživatelům dle návrhu zobrazena dolní lišta obsahující tlačítko pro nahlášení závady. Na této obrazovce bylo testováno, zda si uživatelé tohoto tlačítka všimnou. Uživatelé v tomto úkolu



Obrázek 4.8: Obr: Výsledky klikacího testu při výběru porouchaného objektu

byli navíc seznámeni s tím, že na předchozí obrazovce již objekt vybrali a nyní mají za úkol pouze svůj výběr potvrdit. 67% uživatelů kliklo správně na tlačítko pro nahlášení závady. Zbytek uživatelů chtělo svůj výběr potvrdit kliknutím znovu na místo porouchaného objektu. I když bylo očekáváno, že tento test zvládne alespoň 90% testovaných uživatelů, procento blížící se 70% může být považováno za dostatečné. Důvodem je zejména to, že uživateli se zobrazila pouze tato obrazovka. V hotové aplikaci se dolní lišta s tlačítkem zobrazí až po vybrání porouchaného objektu, čímž bude uživatel vyzván k tomu, aby svůj výběr takto potvrdil. Výsledky testu je možné vidět na obrázku 4.9.

Poslední obrazovkou v klikacím testu, byla obrazovka pro výběr pozice závady, která se může nacházet na libovolném místě. Uživatel měl zde za úkol vytvořit závadu na místě, na kterém se aktuálně nachází. Dle návrhu by zde měl uživatel kliknout na tlačítko *Vybrat pozici*. Dle výsledků testu, 67% testovaných uživatelů kliklo na ikonku pozice závady. Pouze 33% kliklo na tlačítko *Vybrat pozici*. Tímto je jasné, že toto tlačítko nesplnilo svou funkci.

Obrázek 4.9: Obr: Výsledky klikacího testu při výběru porouchaného objektu

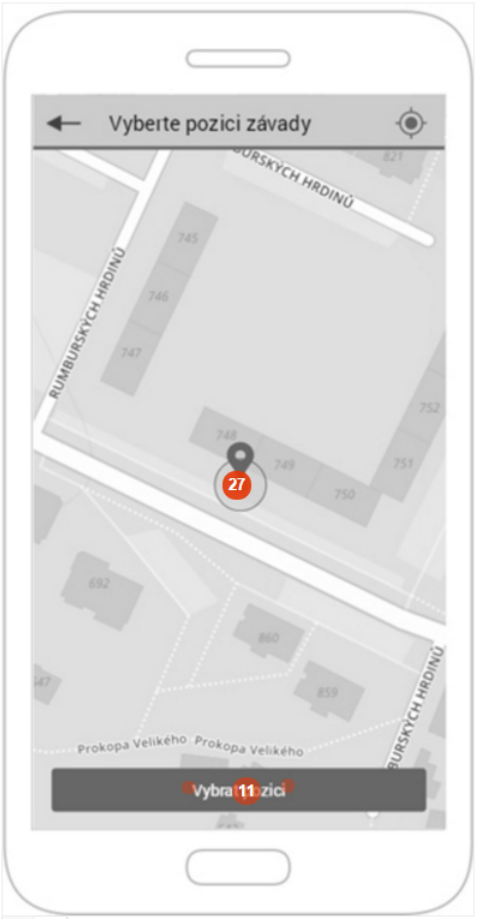
Jako nejjednodušší řešení bude umožnit uživatelům svůj výběr potvrdit jak kliknutím na tlačítko *Vybrat pozici*, tak i na ikonku označující místo závady. Výsledek tohoto testu je možné vidět na obrázku 4.10.

4.2.5 Závěr

Prvotní návrh klikacího testu bohužel absolutně selhal. Naštěstí první návrh testu byl nejprve otestován na třech osobách, takže nedostatky byly odhaleny včas. Druhý návrh klikacího testu již prošel v pořádku. Nejprve byl test vyzkoušen na několika osobách a poté na zbytku účastníků testu.

Na základě klikacího testu byly zjištěny největší nedostatky zejména na úvodní obrazovce, kde velká část účastníků nepochopila význam tlačítka *FAB*. Problém však mohl také být způsoben špatně podaným úkolem. Tato skutečnost by měla však být potvrzena pomocí uživatelského testování.

Dalším zjištěným nedostatkem provedeného návrhu je, že na obrazovce pro



Obrázek 4.10: Obr: Výsledky klikacího testu při potvrzování vybrané pozice závady

výběr pozice závady (viz. obrázek 4.10) testovaní uživatelé klikli pro potvrzení pozice závady přímo na ikonku znázorňující pozici závady. Návrh však byl takový, že by uživatelé měli pozici potvrdit pomocí tlačítka *Vybrat pozici*. Navrhovaná úprava je tedy taková, aby uživatelé mohli pozici závady potvrdit i kliknutím na ikonku závady.

U ostatních testovaných obrazovek testy proběhly v pořádku a na základě výsledků není třeba navrhovat nějaké změny v návrhu.

4.3 Uživatelské testování

V této kapitole bude popsán samotný proces návrhu testovacích úkolů a scénářů, které poté budou otestovány na uživatelích, vybraných na základě uživatelského dotazníku (viz. 4.1).

Na konci kapitoly se pak nachází také samotné zhodnocení uživatelského testování.

4.3.1 Návrh testovacích scénářů

Před samotným uživatelským testováním je nutné nejprve vytvořit úkoly, které bude uživatel plnit. Jako příklad úkolů pro internetový obchod může být:

- Najít zboží v kategorii
- Najít obchodní podmínky
- Objednání produktu
- Změna již vytvořené objednávky

Jelikož seznam může být velice dlouhý a doba testování jednoho uživatele je omezená, je dobré si vybrat jen ty nejdůležitější body. Tyto body poté budou testovány na uživatelích. Ostatní body můžou být testovány například v dalších kolech testování, pokud se tedy koná více kol.

Na základě těchto úkolů jsou poté vytvořeny testovací scénáře. Testovací scénář velice pomůže uživateli vžít se do role uživatele, který potřebuje provést daný úkol.

Úkol: *„Přihlaste se k doktorskému studijnímu programu na Masarykově univerzitě.“*

Scénář: *„Máte titul Mgr. a po spoustě času stráveného výzkumem jste se rozhodli vstoupit do doktorského studijního programu na Fakultě sociálních studií Masarykovi univerzity na Katedře sociologie. Přihlaste se k přijímacímu řízení do tohoto studijního programu.“*

Scénář poskytuje určitý kontext („Vy jste...“, „Chcete provést...“) a uživatelům poskytuje informace, které by měli znát, ale neznají (například uživatelské jméno a heslo k testovacímu účtu). S popisem to není nutné přehánět. Popis by neměl obsahovat jakýkoliv detail, který není potřebný.

Na tom je těžká jediná věc: nenechat ve scénáři žádná vodítka.

Vše by mělo být formulováno tak, aby to bylo jasné, jednoznačné a snadno pochopitelné, a je třeba to provést bez použití běžných nebo jedinečných slov, která se objeví na obrazovce. Jestliže je tak učiněno, je úkol přeměněn v jednoduchou hru na hledání slov.[10]

4.3.2 Testovací úkoly

Níže se nachází seznam testovacích úkolů, které by bylo vhodné na uživatelích otestovat.

- **TU1:** nahlásit závadu na předem definovaném objektu

- **TU2:** nahlásit závadu na předem nedefinovaném objektu
- **TU3:** nastavit jinou obec pro hlášení závady
- **TU4:** zobrazit informace o závadě
- **TU5:** přidat fotografii k závadě
- **TU6:** vytvořit reakci na závadu
- **TU7:** změnit stav řešení závady
- **TU8:** změnit vrstvu mapy na satelitní

Seznam úkolů obsahuje i úkoly, které bude složitější testovat na Lo-Fi prototypu, a proto některé testovací scénáře budou muset být upraveny dle možností prototypu.

4.3.3 Testovací scénáře

Scénář S1 - nahlášení závady na předem definovaném objektu:
„Bydlíte v domě č.p. 746 v ulici Rumburských hrdinů. Před Vaším domem nesvítí lampa veřejného osvětlení, což může ohrožovat bezpečnost Vašich dětí při cestě do školy. Ohlaste závadu na této lampě.“
Testované úkoly: TU1

Scénář S2 - nahlášení závady na předem nedefinovaném objektu:
„Při cestě do obchodu jste narazili na poškozený chodník, přes který je velice obtížné pokračovat s kočárkem. Nahlaste závadu na tomto chodníku. Při vytváření závady pořiďte také fotografii, aby bylo možné problém lépe identifikovat. Nacházíte se na místě poškozeného chodníku.“
Testované úkoly: TU2, TU5

Scénář S3 - volba jiné obce pro hlášení závady:
„Mobilní aplikaci využíváte v místě Vašeho bydliště, v Novém Boru, ale také v místě kde máte chatu, v obci Krompach. Zobrazte závady v místě, kde máte chatu.“
Testované úkoly: TU3

Scénář S4 - vytvoření reakce na závadu a změna stavu závady:
„Jste zaměstnancem úřadu a dostal jste zprávu, že poškozený odpadkový koš byl právě opraven. Nastavte závadu jako vyřešenou a přidejte následující komentář k řešení závady: *Koš byl vyměněn za nový.*“
Testované úkoly: TU4,TU6,TU7

Scénář S5 - změna vrstvy v mapě:
„Nyní chcete nahlásit, že při včerejší vichřici spadl strom v parku. Jelikož v mapě nejsou stromy vyznačeny, je vaším úkolem změnit zobrazenou mapu na satelitní snímky.“
Testované úkoly: TU8

Testovací úkoly byly sdruženy do 5 testovacích scénářů, protože některé úkoly se mohou vázat na další. Tyto testovací scénáře by měly otestovat nejčastější potřeby většiny uživatelů.

4.3.4 Tvorba prototypu

Pro potřeby testování s uživateli je důležité vytvořit prototyp, který je funkčně podobný reálné aplikaci. Pro tyto účely je možné například vytvořit PDF dokument obsahující navržené obrazovky. Poté je v dokumentu možné vytvořit odkazy, například v místě, kde se na navržené obrazovce nachází tlačítko. Po kliknutí na vytvořený odkaz je možné přejít na stránku, kde se nachází jiná obrazovka. Toto je jednoduchý způsob jak simulovat reálnou aplikaci. Další možností je například vytvoření papírového prototypu, na kterém by vždy uživatel označil místo kam by chtěl kliknout a na základě toho by mu byl předložen další papír s návrhem další obrazovky.

Aplikace navrhovaná v této práci má být používána na mobilních telefonech, proto by návrh pomocí prototypu ve formátu PDF nebo na papír nedokázal dobře simulovat veškeré možnosti dotykových displejů. Proto pro tvorbu prototypu bylo využito aplikace InVision[9], která dokáže prototyp spustit na mobilním zařízení a dále také například umožňuje uživateli využívat pohybová gesta (rolování apod). Tímto může být při testování docíleno co největšího napodobení prostředí reálné aplikace.

Prototyp byl vytvořen tak, aby testovaný uživatel mohl provést všechny testovací scénáře, které měly za úkol pokrýt všechny hlavní funkcionality aplikace. Do prototypu testované aplikace bylo nutné vytvořit také prostředí pro pořízení fotografie, která je součástí jednoho z testovacích scénářů. Do prototypu se nepodařilo vytvořit interaktivní formuláře tak, aby je mohl uživatel vyplňovat. S tímto omezením prototypu byl každý účastník seznámen a na konci testu byl dotázán zda byla všechna pole ve formuláři srozumitelně popsány a zda by účastník věděl co má do příslušných polí vyplnit. V uživatelském dotazníku se dokonce nacházela otázka, zda uživatel rozumí všem položkám ve formuláři pro vytvoření nové závady. Všechny testované osoby odpověděli *ano*. Testování srozumitelnosti polí ve formulářích bude navíc ještě otestováno v Hi-Fi prototypu. V případě vyskytnutí problému s poli ve formulářích, nebude se už jednat o tak velký problém, protože při opravě případných chyb v návrhu, nebude potřeba velký zásah do samotného návrhu celé aplikace.

[9] https://www.invisionapp.com/

Tabulka 4.1: Profil 1. účastníka uživatelského testování

Pohlaví	Žena
Věk	34 let
Zkušenost s webovou aplikací na hlášení závad	Ano
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ano

4.3.5 Testování s uživateli

V této kapitole se nachází informace o průběhu uživatelského testování. Do testování byli uživatelé vybráni na základě uživatelského dotazníku. Vybrány byly osoby, které vlastní mobilní telefon se systémem Android a považují jako jeden z nejpohodlnějších způsobů k ohlášení závady hlášení pomocí mobilní aplikace. Z těchto osob byly vybrány osoby napříč dle pohlaví a věkové skupiny.

Testování probíhalo v kanceláři, kde bylo uživateli poskytnuto mobilní zařízení s předinstalovaným prototypem. Aby testování bylo objektivní byl uživatel před testem upozorněn, že pole ve formuláři není možné vyplňovat a že mu během testu nebudou poskytovány žádné rady. Testovaný uživatel vždy dostal na papíře vytištěný pouze jeden úkol, aby nebyl rozptylován dalšími úkoly.

Obrazovka zařízení, na kterém uživatel testoval, byla nahrávána na kameru se zvukem, aby bylo možné například sledovat pohyby rukou uživatele, ale i poslouchat komentáře uživatele při řešení úkolů. Po dokončení testu byl každý uživatel požádán o zpětnou vazbu na práci s prototypem aplikace. Na základě získaných nahrávek, byl pak vytvořen report o každém účastníkovi uživatelského testování.

Účastník 1:

- **Scénář S1:** účastník na úvodní obrazovce (viz. obrázek 3.3a) na mapě hledal místo, kde chce nahlásit závadu (toto místo na mapě na úvodní obrazovce zobrazené není). Účastník se snažil pohybovat s mapou, což v prototypu není možné. Poté se účastník snažil kliknout do seznamu závad na závadu označenou jako závada na lampě (na ikonku této závady). Až poté, když uživatel oznámil, že úkol nezvládne, byl požádán průvodcem testu, aby zkusil s aplikací ještě chvíli pracovat a pokusil se najít ještě další možnost. Poté uživatel klikl na tlačítko FAB, což bylo cílem úkolu a uživatel tak mohl pokračovat dál. Celkem od přečtení úkolu uživateli trvalo dvě minuty než se dostal na další obrazovku a pokračoval dál v úkolu. Účastník po průchodu přes úvodní obrazovku již zvládl úkol bez jakýchkoliv problémů.

Tabulka 4.2: Profil 2. účastníka uživatelského testování

Pohlaví	Muž
Věk	42 let
Zkušenost s webovou aplikací na hlášení závad	Ano
Zkušenost s mobilní aplikací na hlášení závad	Ano
Účastnil se někdy uživatelského testování	Ano

- **Scénář S2:** Uživatel na obrazovce pro výběr pozice závady klikl na místo v mapě, kde se mu zobrazovala pozice jeho a závady, následně klikl správně na tlačítko Vybrat pozici. Úkol byl proveden v pořádku.
- **Scénář S3:** V pořádku
- **Scénář S4:** V pořádku (závadu zobrazil přes mapu)
- **Scénář S5:** V pořádku

Shrnutí diskuze s účastníkem:

Největší problém byl pro účastníka hned na začátku, zorientovat se. V prototypu bylo matoucí to, že mapa měla nastavený příliš velký zoom, proto se účastník snažil nejprve najít místo závady. Účastník tak nevěděl čím má začít - zda například volbou typu závady nebo místem závady. Účastník navrhuje pod tlačítko FAB (pro vytvoření závady) přidat popisek *přidat*, protože na úvodní obrazovce je velké množství kruhových ikonek a textu,tím pádem toto tlačítko zde zaniká. Účastník vidí problém také v tom, že méně zkušený uživatel by asi nedokázal změnit vrstvu mapy, jelikož by tento krok pod ikonkou tří teček nejspíš neočekával (účastník to sám zvládl v pořádku).

Účastník 2:

- **Scénář S1:** Účastník po dvaceti vteřinách klikl do seznamu závad na závadu na veřejném osvětlení (popis závady: nesvítí). Uživatel očekával že si zde zvolí typ závady. Poté se uklepl a zobrazil si seznam měst, kde je možné hlásit závady. Po dalších 40 vteřinách klik na tlačítko FAB pro vytvoření závady. Při klikání na tlačítko FAB účastník prohlásil: „Co je tohle?“ z čehož plyne, že uživatel už nevěděl co zkoušet dál, tak klikl na tlačítko, o kterém nevěděl co by mohlo znamenat. Zbytek scénáře už zvládl v pořádku.
- **Scénář S2:** V pořádku
- **Scénář S3:** V pořádku
- **Scénář S4:** V pořádku (závadu zobrazil přes mapu)

Tabulka 4.3: Profil 3. účastníka uživatelského testování

Pohlaví	Žena
Věk	48 let
Zkušenost s webovou aplikací na hlášení závad	Ne
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ne

Tabulka 4.4: Profil 4. účastníka uživatelského testování

Pohlaví	Muž
Věk	52 let
Zkušenost s webovou aplikací na hlášení závad	Ano
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ne

- **Scénář S5:** V pořádku

Shrnutí diskuze s účastníkem: Účastník z počátku špatně chápal logiku přidávání závady. Seznam závad na úvodní obrazovce považoval jako kategorie závad. Účastník tedy nechápal proč by měl klikat na FAB tlačítko na úvodní obrazovce, když už se mu kategorie zobrazují (místo kategorií se zobrazoval seznam závad). Uživatel prohlásil, že si vůbec obsah textů v seznamu závad nečetl, že se řídil pouze pomocí ikonek.

Účastník 3:

- **Scénář S1:** Účastník testování nejprve začal v mapě vyhledávat místo, kde chtěl závadu nahlásit, poté prohlásil že místo na mapě není. Po 60 vteřinách zkoušení různých možností v aplikaci klikl účastník na FAB tlačítko pro přidání nové závady a zbytek scénáře zvládl už v pořádku.
- **Scénář S2:** V pořádku
- **Scénář S3:** V pořádku
- **Scénář S4:** V pořádku (závadu zobrazil přes seznam závad)
- **Scénář S5:** V pořádku

Shrnutí diskuze s účastníkem: Problém byl pouze první scénář. Účastník se snažil dostat na požadované místo na mapě. Účastník navrhuje možnost nejprve kliknout do mapy, označit místo závady a až poté vybrat typ závady.

Tabulka 4.5: Profil 5. účastníka uživatelského testování

Pohlaví	Žena
Věk	27 let
Zkušenost s webovou aplikací na hlášení závad	Ne
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ano

Účastník 4:

- **Scénář S1:** Účastník začal na úvodní stránce vyhledávat v mapě požadované místo, kde má být nahlášena závada. Byl také upozorněn, že s mapou v prototypu není možné hýbat. Poté klikl do seznamu závad na závadu na veřejném osvětlení s názvem „nesvítí“. Po cca 20 vteřinách od posledního kliknutí uživatel klikl na FAB tlačítko pro přidání závady. Další kroky scénáře již byly provedeny v pořádku.
- **Scénář S2:** Uživatel se snažil potvrdit pozici závady kliknutím na ikonku v mapě, která označuje pozici závady. Při zjištění, že se nic neděje, účastník klikl na tlačítko *Vybrat pozici*, které bylo cílem testu na obrazovce pro výběr pozice závady.
- **Scénář S3:** V pořádku
- **Scénář S4:** V pořádku (závadu zobrazil přes seznam závad)
- **Scénář S5:** V pořádku

Shrnutí diskuze s účastníkem: Jak začít vytvářet závadu se jevilo jako největší problém. O FAB tlačítku na vytvoření závady si účastník myslel, že slouží k přiblížení mapy.

Účastník 5:

- **Scénář S1:** Uživatel nejprve začal hledat místo závady v mapě. Po zjištění, že s mapou nelze v prototypu pohybovat, klikl na závadu na veřejném osvětlení v seznamu nahlášených závad. Ke kliknutí na požadované FAB tlačítko pro vytvoření závady nedošlo a proto musel být účastník posunut do dalšího kroku (výběr kategorie závady), aby mohl otestovat další kroky. Další kroky již byli zvládnuty v pořádku.
- **Scénář S2:** Uživatel se nejprve snažil na obrazovce pro výběr pozice, potvrdit pozici závady kliknutím na ikonku znázorňující pozici závady, poté si všiml tlačítka *Vybrat pozici*, kterým již svůj výběr potvrdil. Zbytek úkolu byl splněn v pořádku.

- **Scénář S3:** Uživatel se snažil požadovanou obec nejprve najít na mapě. Poté klikl do menu pro výběr mapy a všiml si výběrového menu v aplikační liště, která slouží pro výběr obce pro hlášení závad.
- **Scénář S4:** V pořádku (závadu zobrazil přes seznam závad)
- **Scénář S5:** V pořádku

Shrnutí diskuze s účastníkem: Problém s přidáním nové závady. Po proběhnutí celého testu se účastník zeptal, co je obsahem seznamu zobrazeného na úvodní obrazovce.

4.3.6 Zhodnocení

Na základě testování s uživateli byl zjištěn jako největší nedostatek návrh úvodní stránky aplikace. Ani jeden z testovaných uživatelů nedokázal na této stránce začít vytvářet novou závadu. Problémem bylo, že žádný z uživatelů z prvopočátku vůbec nezaregistroval FAB tlačítko na přidání nové závady. Až po delším přemýšlení, nebo když uživatel vyzkoušel kliknout na všechny tlačítka na obrazovce, dokázal vytvořit novou závadu. Někteří uživatelé dokonce úkol nedokázali splnit.

Problémem zaniknutí FAB tlačítka na přidání nové závady může být to, že prototyp byl černobílý. Jelikož například jeden z uživatelů očekával od tohoto tlačítka, že slouží k přiblížení mapy, je možné zmiňovanou možnost se černobílým prototypem zamítnout. Důvodem může být to, že účastníci testu o tlačítku věděli, ale nevyužili ho. Jednou z možností řešení daného problému by mohlo být změnění ikonky ve FAB tlačítku. Po delší úvaze bylo usouzeno, že vhodnější ikonka než *plus* pro vytvoření něčeho nového neexistuje. Někteří uživatelé seznam závad zobrazený na úvodní obrazovce považovali za seznam kategorií závad. Dále většina uživatelů očekávala, že kliknutím do mapy na úvodní obrazovce bude možné vytvořit novou závadu v místě, kam uživatelé klikli.

Jedním z řešení daného problému by mohlo být to, že při první návštěvě aplikace (zatím nenahlásil žádnou závadu), se zobrazí úvodní obrazovka obsahující text: „kliknutím na tlačítko“ *ikonka tlačítka*, vytvoříte novou závadu. Toto řešení bylo již navrhováno v prvotním návrhu, ale do samotného uživatelského testování zařazeno nebylo, jelikož se zdá být pouze částečné a nejedná se o řešení problému s přehledností úvodní obrazovky. Dané řešení by problém pouze oddálilo, protože uživatel by na tuto obrazovku narazil při druhém hlášení závady. Což by mohlo nastat i několik týdnů od prvního nahlášení závady.

Na základě předchozích odstavců bylo usouzeno, že bude potřeba úvodní obrazovku zcela přepracovat. Tato obrazovka dle uživatelského testovaní je nepřehledná a uživatel neví co je jejím obsahem.

Problematika přehlednosti úvodní stránky byla již zjištěna při tzv. klikacím testu popsaném v kapitole 4.2.

Dalším úkolem, který některým uživatelům dělal částečně problém, byl úkol pro výběr pozice závady přímo v mapě (kdy nejsou zobrazeny žádné objekty, ale uživatel vybírá libovolnou pozici). Polovina testovaných uživatelů klikla ihned na tlačítko Vybrat pozici, které posune uživatele o krok dále. Polovina uživatelů ovšem nejprve klikla na ikonku v mapě znázorňující pozici závady a poté až klikla na tlačítko Vybrat pozici. Nejedná se o tak velký problém, protože všichni účastníci testu daný úkol nakonec zvládli. Bylo by vhodné se nad tímto nedostatkem raději pozastavit. Jedním z řešení by například mohlo být provést animaci s tlačítkem *Vybrat pozici*. Při kliknutí na ikonku znázorňující pozici závady by se tlačítko například mohlo „zatřást“ nebo lehce zvětšit a pak zase zmenšit. Řešení tohoto nedostatku bude provedeno až v Hi-Fi fázi, protože v Lo-Fi prototypu je velice obtížné animace simulovat.

Většina účastníků uživatelského testování scénáře číslo 4 zvládli. Občas ale účastníkům delší dobu trvalo všimnout si, že po přidání reakce k závadě je ještě nutné nastavit stav závady. Účastníci testu očekávali, že přidáním reakce je již závada nastavena jako vyřešená. Vhodné bude asi vytvořit novou komponentu, která uživateli lépe znázorní proces zpracování závady a na základě aktuálního stavu umožní uživateli přidat také reakci.

4.4 Změny provedené na základě testování

Na základě výsledků proběhlých testování bylo navrženo, že úvodní obrazovka bude obsahovat spodní navigaci (viz. kapitola 2.1.6). Spodní navigace je specifická tím, že dle pravidel pro vytváření Android aplikací, může tato navigace obsahovat ikonky a ke každé ikonce vždy krátký popis, což by mohlo uživatelům pomoci se lépe zorientovat v aplikaci. Obsahem této navigace by byly následující tlačítka:

- nahlásit - daná položka menu by zobrazila obrazovku pro výběr kategorie pro novou závadu, jednalo by se o úvodní obrazovku
- seznam - daná položka menu by zobrazila obrazovku obsahující seznam všech závad
- mapa - daná položka menu by zobrazila mapu se všemi závadami

4.4.1 Nová úvodní obrazovka

Nová úvodní obrazovka aplikace nyní obsahuje pouze seznam kategorií závad, do kterých je možné nahlásit novou závadu. Seznam kategorií (pro nahlášení nové závady) je na úvodní obrazovce zobrazen, protože se jedná o akci, kterou bude většina uživatelů primárně využívat. Každá položka v seznamu bude obsahovat nadpis, ikonku a jednoduchý popis. Ikonka by mohla pomoci uživatelům rychleji se na úvodní obrazovce zorientovat. V případě, že uživatel

nebude vědět kam novou závadu nahlásit, jednoduchý popis by měl pomoci uživatelům, přesněji vybrat kategorii. Na návrhu této obrazovky (viz. obrázek 4.11) je možné také vidět spodní navigaci, která byla navržena na základě poznatků z uživatelského testování.

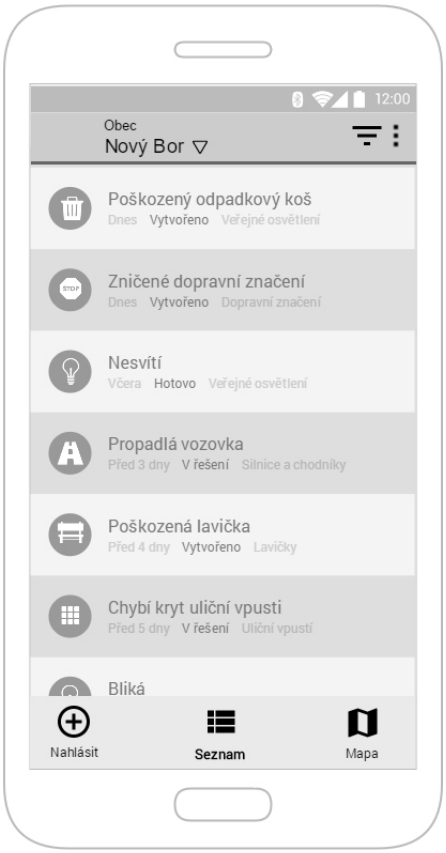
Obrázek 4.11: Úvodní obrazovka - výběr kategorie nové závady

4.4.2 Seznam a mapa se závadami

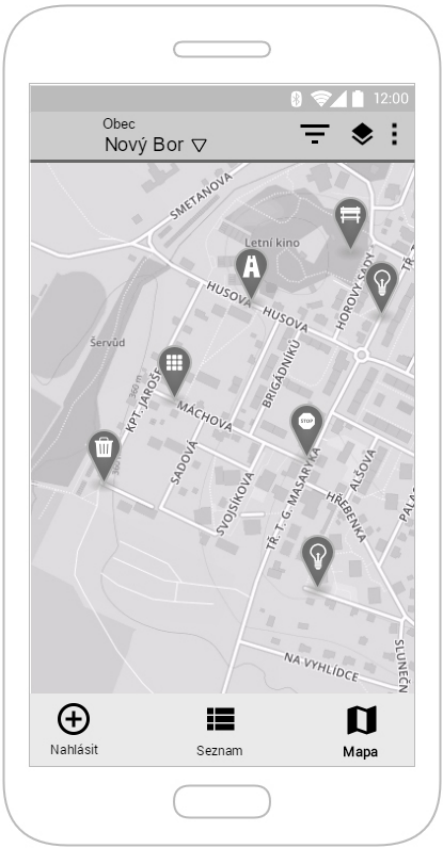
Další dvě záložky, na které je možné se přepnout z úvodní obrazovky, je obrazovka se seznamem závad a obrazovka s mapou nahlášených závad. K rozdělení na dvě obrazovky došlo z důvodu, aby uživatel viděl že je možné snadno se přepínat mezi různými zobrazeními. V prvotním návrhu bylo navrženo, že typ zobrazení bude možné přepínat pomocí pohybu prstu, což nemusí být zřejmé pro všechny typy uživatelů. V uživatelském testování bylo také zjištěno, že velké procento uživatelů preferuje zobrazení v mapě, ale i zobrazení pouze seznamu, proto byla v návrhu ponechána možnost volby. Detailnější informace jsou obsaženy v kapitole 4.1.3.

Na obě obrazovky (se seznamem závad i mapou závad) bylo nově přidáno tlačítko pro možnost nastavení filtrování. Dle návrhu by ve filtru bylo možné nastavit, zda mají být zobrazeny pouze závady odeslané z mobilního telefonu uživatele nebo zda mají být zobrazeny všechny závady. Tato možnost byla dle prvotního návrhu možná pomocí záložek. Filtrování pomocí záložek bylo nakonec odstraněno, protože dle kapitoly 2.1.6, není vhodné kombinovat spodní navigaci se záložkami. Funkcionalita nastavení filtru bude muset být ještě řádně otestována.

Dle získaných poznatků z uživatelského testování, bylo do mapy se závadami přidáno tlačítko pro zobrazení menu s možností nastavení jiné vrstvy mapy. Tímto by se mělo stát pro uživatele snazší zvolit jinou vrstvu mapy, ale také by uživatel měl předpokládat že tato možnost zde je. Možnost, zvolit jinou vrstvu mapy, byla možná doposud pouze pomocí menu zobrazeného po kliknutí na tlačítko „tří teček“ v pravém horním rohu. Možnost změny vrstvy bude fungovat, na všech obrazovkách kde se bude mapa vyskytovat, stejně.



Obrázek 4.12: Obrazovka se seznamem nahlášených závad



Obrázek 4.13: Obrazovka s mapou obsahující nahlášené závady

4.5 Druhé kolo testování

V druhém kole testování bude provedeno pouze uživatelské testování. V uživatelském testování se budou nacházet pouze úkoly týkající se obrazovek, které byly na základě předešlých testů upraveny. Prototyp byl upraven tak, aby umožňoval otestování nově navržených obrazovek.

Nově také byl vytvořen nový úkol:

- **TU9:** zobrazit závady nahlášené uživatelem

4.5.1 Testovací scénáře

Do druhého kola testování byl opět zařazen scénář S2. Scénář S2 pokrývá vytvoření nové závady (otestování nové úvodní obrazovky - viz. obrázek 4.11) a potvrzení vybrané pozice v mapě (viz. obrazovka 3.6). Obrazovka pro potvrzení vybrané pozice totiž v novém prototypu umožňuje uživateli potvrdit pozici kliknutím na ikonku místa závady. Tento scénář také na úvodní obrazovce otestuje, zda uživatele napadne posunovat seznamem závad. Uživatel

má za úkol vybrat kategorii závady, která se nachází až na konci seznamu. Není tedy ihned vidět.

Další scénář, který byl do druhého kola zařazen, je scénář S5. Tento scénář otestuje, zda uživatelé dokáží změnit vrstvu mapy, pomocí nové ikonky umístěné v aplikační listě.

Scénář, který byl také do druhého kola zařazen, je scénář S4. Tento scénář má v tomto kole testování odhalit, zda účastníci dokáží přepnout na jednu z nových záložek (seznam závad nebo mapa závad) umístěných v nové spodní navigaci.

Nově byl vytvořen nový scénář, který má otestovat úkol TU9.

Scénář S6 - zobrazení závad nahlášených uživatelem:
„Před půl rokem jste nahlásili závadu a nyní potřebujete zjistit její stav řešení. Zobrazte na mapě pouze závady, které jste nahlásili vy?“
Testované úkoly: TU9

4.5.2 Testování s uživateli

Druhé kolo testování s uživateli bylo prováděno na třech účastnících, kteří se zúčastnili i prvního kola testování. Podmínky testování byly zachovány stejné jako v prvním kole. Do tohoto testování byly zařazeny scénáře, které jsou popsány v kapitole 4.5.1.

Účastník 1:
Profil účastníka je možné vidět v tabulce 4.1

- **Scénář S2:** Uživateli chvíli trvalo než našel správnou kategorii závady. Uživatel totiž neočekával, že se bude možné na úvodní obrazovce posunovat („scrollovat“). Následně, když vybral kategorii, zvládl již úkol v pořádku.
- **Scénář S4:** Účastník testu klikl na záložce *nahlásit* na kategorii odpadkové koše. Zde se uživateli zobrazila mapa obsahující lampy (prototyp nebyl na tuto situaci připraven - nebyla implementována obrazovka pro vytvoření závady na odpadkovém koši). Uživatel si myslel, že vybral špatnou kategorii a proto se pokusil vybrat kategorii ještě jednou. Po opětovném návratu na úvodní obrazovku vybral ve spodní navigaci položku *seznam*. Zbytek úkolu již zvládl v pořádku.
- **Scénář S5:** Uživatel byl ovlivněn předchozím testováním a klikl tedy nejprve na tlačítko pro zobrazení více nastavení (v tomto menu bylo možné nastavit vrstvu v prvním kole testování). Dále vyzkoušel kliknout na novou ikonku pro filtrování zobrazených závad. A poté již klikl správně na tlačítko změny vrstev.

- **Scénář S6:** V pořádku. Účastník testu zjistil, jak nastavit filtr závad již v předchozím testu.

Shrnutí diskuze s účastníkem: Účastníkovi testování přišla nejasná nová ikonka na změnu vrstvy mapy. Pro změnu vrstvy by účastníkovi testu dávalo větší smysl využít ikonku, která je využita v aplikační liště na nastavení filtrování. Účastník navrhoval, že by bylo vhodnější kdyby se veškeré možnosti jak změn filtrování tak změn vrstvy mapy nacházely přímo v menu pro více nastavení. Na úvodní obrazovce se účastníkovi testování nelíbilo to, že *karty* obsahující kategorie závad jsou zbytečně velké a je pak zbytečně nutné „scrollovat“.

Účastník 2:
Profil účastníka je možné vidět v tabulce 4.4

- **Scénář S2:** Uživateli chvíli trvalo než si všiml, že je možné na úvodní obrazovce „scrollovat“. Zbytek úkolu zvládl v pořádku.
- **Scénář S4:** Účastník testu nejprve vybral kategorii pro hlášení závady (odpadkové koše), poté si ještě jednou přečetl svůj úkol a vrátil se zpět na úvodní obrazovku. Pak klikl ve spodní navigaci na záložku seznam. Na obrazovce kde se nachází detail závady měl účastník problém, nevěděl kde je možné stav závady nastavit.
- **Scénář S5:** V pořádku.
- **Scénář S6:** V pořádku.

Shrnutí diskuze s účastníkem: U detailu nahlášené závady by účastník zlepšil zejména možnost nastavení stavu závady. Pro účastníka byla ikonka pro změnu stavu závady nejasná, proto když chtěl změnit stav závady klikal přímo na text stavu závady. Účastníkovi se provedené změny oproti prvnímu testování líbily a považuje návrh za přehlednější.

Účastník 3:
Profil účastníka je možné vidět v tabulce 4.3

- **Scénář S2:** Účastník nejprve netušil, že bude na úvodní obrazovce možné „scrollovat“. Zbytek úkolu účastník zvládl v pořádku.
- **Scénář S4:** Pro hlášení závady účastník testu nejprve vybral kategorii odpadkové koše. Poté začal v mapě hledat odpadkový koš. Po návratu na úvodní stránku zkusil otevřít rozšířená nastavení (pomocí ikonky v pravém horním rohu). Nakonec klikl na položku *seznam* ve spodní navigaci. Zbytek úkolu už zvládl v pořádku.

- **Scénář S5:** Účastník nejprve klikl na ikonku pro nastavení filtru. Poté klikl na správné tlačítko pro nastavení vrstvy mapy. Úkol tedy zvládl v pořádku.

- **Scénář S6:** V pořádku.

Shrnutí diskuze s účastníkem: Pro účastníka byl největší problém se zorientovat v ikonkách. Účastníkovi také nebylo jasné co si představit pod jednotlivými položkami ve spodní navigaci. Největší problém mu dělala zejména položka *seznam*.

4.5.3 Zhodnocení a návrhy na zlepšení

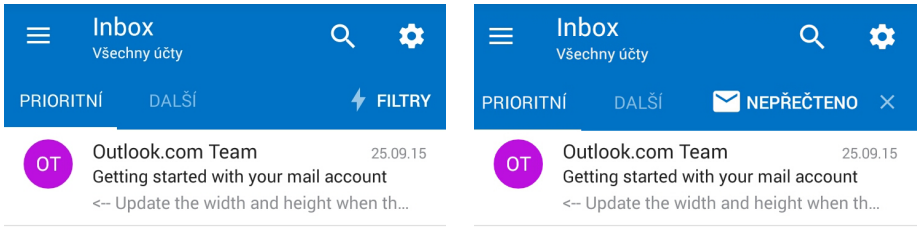
Všichni tři účastníci testování měli problém, že nevěděli o možnosti „scrollovaní“ na úvodní obrazovce. Proto jim chvíli trvalo než vybrali správnou kategorii pro nahlášení nové závady. Je možné, že se jednalo pouze o problém prototypu a účastníci testu neočekávali, že bude možné v prototypu „scrollovat“.

Při tomto kole testování bylo také zjištěno, že uživatelé úvodní obrazovku považují za rozcestník. Proto když jim byl zadán úkol, ať změní stav závady, automaticky všichni tři účastnici klikli na kategorii pro vytvoření nové závady. Závada sice do této kategorie patří, ale uživatel tímto krokem začal vytvářet novou závadu v dané kategorii.

Některým uživatelům také dělalo problém změnit filtrování a změnit typ mapy. Účastníci uživatelského testování nejčastěji nerozuměli jednotlivým ikonkám. Další problém, který doposud nebyl odhalen je, že dosavadní řešení nedává uživateli jasnou informaci o tom, jak je nastaven filtr zobrazených závad. Jako jedno z řešení by mohlo být využití filtrování, jaké je využito například v aplikaci Outlook[10]. Na obrázcích obrazovek z aplikace Outlook je možné vidět jak vypadá aplikační lišta s možností nastavení filtru. Na obrázku 4.14a je možné vidět situaci, když není filtr nastaven. V této situaci je uživateli zobrazena ikonka s popisem, tím pádem by pro něj smysl ikonky měl být jasný. Na obrázku 4.14b je možné vidět situaci kdy je filtr nastaven a vedle nastaveného filtru se nachází ikonka na jeho odstranění.

V tomto kole testovaní bylo pro některé uživatele problémem správně nastavit stav závady. Pro uživatele není jasné kam mají kliknout, aby mohli stav změnit. Často tak klikali přímo na název stavu závady a ne na ikonku *tužky* k tomu určené. Některým účastníkům testů také dělalo problém to, že sice správně nastavili stav závady, ale již nepřidali reakci k závadě (což bylo úkolem).

[10] https://www.microsoft.com/cs-cz/outlook-com/



(a) Filtr nenastaven (b) Filtr aktivován

Obrázek 4.14: Aplikace Outlook - nastavení filtrů

4.6 Závěr

Za účelem otestování vytvořeného Lo-Fi prototypu byl vytvořen uživatelský dotazník, který byl vyplněn 40 osobami. Z dotazníku vyplynulo, že uživatelé nejčastěji chtějí hlásit závady v blízkém okolí jejich bydliště. Na základě dotazníku bylo také vybráno 7 osob vhodných pro uživatelské testování. Z dotazníku bylo také zjištěno, jaké jsou preference o zobrazení již nahlášených závad. 47% uživatelů preferuje zobrazení pouze seznamu závad a 36% uživatelů preferuje zobrazení kombinace obou přístupů.

Pro otestování Lo-Fi prototypu byl také vytvořen tzv. klikací test. Pomocí klikacího testu byly již odhaleny první nedostatky úvodní obrazovky. Zejména pak nepřehlednost úvodní obrazovky a nepochopení smyslu tlačítka *FAB*.

Pro potvrzení získaných výsledků z klikacího testu, bylo provedeno ještě uživatelské testování. Uživatelské testování vyvodilo podobné výsledky jako klikací test.

Na základě získané zpětné vazby byla přepracována úvodní obrazovka. Po úpravě úvodní obrazovka obsahuje spodní navigaci se záložkami na vytvoření nové závady, záložky zobrazující seznam aktuálních závad a mapu aktuálních závad. Záložky pro seznam aktuálních závad a mapu aktuálních závad byly vytvořeny na základě preferencí získaných od uživatelů z uživatelského dotazníku.

Na nově vytvořeném návrhu bylo provedeno opět uživatelské testování a odhaleny byly už ne příliš závažné nedostatky. Jedním z těchto nedostatků je to, že uživatelé neví kde nastavit filtr pro zobrazení nahlášených závad (nejasný význam ikonky). Stejný problém se vyskytl i u změny vrstvy mapy.

Zjištěné nedostatky budou zapracovány při tvorbě Hi-Fi prototypu.

Kapitola 5

Testování Hi-Fi prototypu

Tato kapitola se zabývá testováním prototypu mobilní aplikace ve fázi Hi-Fi. V prototypu určeném k testování v této fázi návrhu, se již nachází veškeré změny provedené na základě výsledků testování v Lo-Fi. V této fázi bude provedeno uživatelské testování za účelem otestování provedených změn a zjištění, zda návrh v Lo-Fi funguje také ve fázi Hi-Fi.

5.1 Provedené změny

V druhém kole testování Lo-Fi prototypu některým účastníkům dělalo problém, že nevěděli o možnosti „scrollování“ mezi kategoriemi pro tvorbu závady na úvodní obrazovce. Z tohoto důvodu byla výška každé položky obsahující kategorii zmenšena tak, aby byl obsah neustále přehledný, ale zbytečně položka nezabírala místo. Bez nutnosti „scrollování“ jsou tak vidět téměř všechny kategorie pro nahlášení závady.

Další problém, který byl odhalen při testování Lo-Fi prototypu, bylo že účastníci druhého kola testování považovali úvodní stránku na záložce s kategoriemi pro tvorbu závad jako rozcestník, jak pro samotnou tvorbu tak i pro vyhledávání již nahlášených závad. Z tohoto důvodu byl do aplikační lišty přidán titulek „Tvorba závady“. Výběrové pole pro možnost změny města nebo obce pro hlášení závady bylo posunuto vpravo.

Na základě druhého kola testování byla do prototypu přidána přehlednější možnost filtrování zobrazených závad v mapě nebo seznamu závad. Nyní se u filtrů nachází také informace o tom, jaký filtr je zrovna nastaven. Uživatel tak bez nutnosti jakékoliv akce má informaci o tom jaké závady mu jsou zobrazeny.

Další vylepšení, které bylo provedeno na základě testování z předchozí fáze, je zobrazení informace o tom jaká je zrovna nastavena podkladová mapa. Jedná se sice o informaci, která je na první pohled uživateli jasná, jelikož rozpozná zda je mapa nastavena na satelitní snímky nebo ne. Ale protože některým uživatelům dělalo problém vrstvu mapy v druhém kole testování

správně nastavit, mělo by takovéto řešení uživatelům pomoci podkladovou mapu změnit.

Poslední úprava, která byla provedena na základě Lo-Fi testování je přidání dialogového okna, které je zobrazeno při změně stavu závady nebo nastavování reakce na závadu. Dialogové okno obsahuje možnost změnit stav závady i rovnou přidat reakci k závadě. Tato změna byla provedena, protože jsou obě změny úzce provázané a provádějí se většinou spolu.

5.2 Uživatelské testování

Pro účely Hi-Fi testování byla využita rozpracovaná mobilní aplikace, která doposud nebyla napojena na API webové aplikace na hlášení závad. V prototypu byly nahrány demonstrační data, které umožnily dostatečné otestování aplikace z pohledu uživatelského rozhraní.

Uživatelské testování probíhalo ve stejných podmínkách jako uživatelské testování Lo-Fi (viz. 4.3.5). Účastníkovi testování byl poskytnut mobilní telefon obsahující prototyp aplikace a postupně mu byly předkládány jednotlivé úkoly.

Testování uživatelského rozhraní v této fázi se zúčastnilo 7 osob, tytéž osoby se zúčastnili i první fáze testování Lo-Fi prototypu (viz. 4.3.5). Jelikož testování Hi-Fi prototypu probíhalo jeden měsíc od testování Lo-Fi prototypu, většina testovaných osob si již nepamatovala jak vypadal prototyp, který naposledy testovali. Tato informace je důležitá zejména proto, aby uživatelé neměli v paměti jak řešili úkoly, které jim během testování dělaly problém.

Do testování byly zařazeny všechny úkoly, které byly součástí jak prvního (viz. 4.3.3) tak druhého kola (viz. 4.5.1) testování Lo-Fi prototypu. Jediný scénář, který byl změněn byl scénář S2 popsaný na straně 53. Tento scénář byl pro každého účastníka změněn tak, že účastník měl označit nějaké místo závady, které bylo blízko jeho bydliště. Tato změna měla za úkol otestovat, zda uživatelé dokáží správně vybrat pozici závady v mapě. Popis zvolené metody pro výběr pozice závady se nachází v kapitole 3.7.3.

5.2.1 Účastník 1

Profil účastníka je možné vidět v tabulce 4.1

- **Scénář S1:** V pořádku
- **Scénář S2:** V pořádku (účastník pozici závady potvrdil kliknutím na ikonku znázorňující pozici závady - jedná se o zapracovanou změnu z Lo-Fi testování)
- **Scénář S3:** V pořádku

- **Scénář S4:** Účastníkovi se nedařilo kliknout na tlačítko pro změnu stavu závady. Tento problém byl způsoben velikostí tlačítka, které tak spatně reagovalo. (Nejedná se o chybu návrhu, ale prototypu.) Dalším problémem bylo, že při vyplňování textového pole pro reakci na závadu, se na klávesnici mobilního telefonu skrylo tlačítko pro potvrzení úprav.
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

5.2.1.1 Shrnutí diskuze s účastníkem:

Účastník testování navrhuje změnit položku s názvem *Nahlásit* ve spodní navigaci na úvodní obrazovce na položku *Hlášení*.

5.2.2 Účastník 2

Profil účastníka je možné vidět v tabulce 4.2

- **Scénář S1:** V pořádku
- **Scénář S2:** V pořádku (účastník pozici závady potvrdil pomocí tlačítka)
- **Scénář S3:** V pořádku
- **Scénář S4:** Při vyplňování textového pole pro reakci na závadu, se na klávesnici mobilního telefonu skrylo tlačítko pro potvrzení úprav.
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

5.2.2.1 Shrnutí diskuze s účastníkem:

Účastníkovi testování nedělal problém žádný úkol.

5.2.3 Účastník 3

Profil účastníka je možné vidět v tabulce 4.3

- **Scénář S1:** V pořádku
- **Scénář S2:** V pořádku (účastník pozici závady potvrdil pomocí tlačítka)
- **Scénář S3:** V pořádku

- **Scénář S4:** Účastník nejprve na úvodní obrazovce klikl na kategorii závady, do které požadovaná závada sice patří, ale uživatel klikl na tlačítko pro vytvoření nové závady. Po otevření mapy, kde měl uživatel vybrat pozici, se vrátil zpět na úvodní obrazovku a přes seznam závad zobrazil požadovanou závadu a zbytek úkolu už zvládl v pořádku.
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

5.2.3.1 Shrnutí diskuze s účastníkem:

Účastníkovi testování chvíli trvalo než se seznámil s funkcionalitou na zobrazení velkého množství bodů v mapě, se kterou se setkal ve scénáři S1.

5.2.4 Účastník 4

Profil účastníka je možné vidět v tabulce 4.4

- **Scénář S1:** V pořádku
- **Scénář S2:** Účastník správně nastavil pozici a přiblížení mapy, poté se ale snažil ikonku znázorňující pozici pro novou závadu umístit na jiné místo kliknutím do mapy. Po zjištění že se ikonka nepřemístila, se uživatel chvíli snažil zjistit, jak posun ikonky funguje. Poté pozici nastavil správně a zbytek úkolu už zvládl v pořádku.
- **Scénář S3:** V pořádku
- **Scénář S4:** Při vyplňování textového pole pro reakci na závadu, se na klávesnici mobilního telefonu skrylo tlačítko pro potvrzení úprav.
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

5.2.4.1 Shrnutí diskuze s účastníkem:

Účastník testování navrhuje, aby položka která je aktuálně vybraná, byla ve spodní navigaci na úvodní obrazovce více zvýrazněná.

5.2.5 Účastník 5

Profil účastníka je možné vidět v tabulce 4.5

- **Scénář S1:** V pořádku

- **Scénář S2:** Účastník správně nastavil pozici a přiblížení mapy. Ikonku pro nově nahlášenou závadu chtěl posunout o malý kousek vedle. Kliknutím těsně vedle ikonky rovnou potvrdil pozici závady. Uživatel sice takto přešel na další krok, ale neoznačil přesné místo, které opravdu označit chtěl. Proto byl účastník požádán o vykonání úkolu znovu. Při druhém opakování účastník již zjistil, že ikonku kliknutím do mapy nepřemístí a po chvilkovém zkoumání toho, jak změna pozice ikonky funguje, již pozici správně nastavil a poté ji potvrdil pomocí tlačítka.
- **Scénář S3:** V pořádku
- **Scénář S4:** V pořádku
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

5.2.5.1 Shrnutí diskuze s účastníkem:

Účastník neměl žádné komentáře.

5.3 Zhodnocení

Při testování Hi-Fi prototypu bylo oproti testování Lo-Fi prototypu pro mnoho testovaných uživatelů mnohem snadnější s prototypem Hi-Fi pracovat. Při testování Lo-Fi prototypu bohužel nebylo možné otestovat zda uživatelé dokáží správně pracovat s mapou. Jednalo se tedy o jednu z funkcionalit, která měla být při testování ověřena. Velkou výhodu v Hi-Fi prototypu viděli účastníci například v tom, že prototyp byl barevný.

Cílem testování v této fázi vývoje uživatelského rozhraní bylo zjistit zda návrh v provedený v Lo-Fi fázi bude fungovat i jako reálná aplikace v prostředí mobilního telefonu.

Při testování se narazilo také na chyby, které přímo nesouvisely s provedeným návrhem. Jednou z takových chyb byla malá plocha tlačítka pro změnu stavu závady. Účastníci testování tak sice klikali do správného místa, ale aplikace nereagovala a účastníci tak svůj klik museli opakovat vícekrát.

Při testování Lo-Fi prototypu v druhém kole, někteří z uživatelů považovali záložku obsahující kategorie pro tvorbu závad na úvodní obrazovce jako rozcestník pro všechny úkony. Na základě tohoto zjištění byl do aplikační lišty přidán titulek s informací, co daná obrazovka obsahuje. Na základě provedených úprav se daná problematika opakovala pouze u jednoho účastníka testu.

Přestože se u dvou uživatelů vyskytl problém s výběrem pozice závady v mapě (scénář S2), nebude tato funkcionalita nijak měněna. Uživatelé po krátkém seznámení s pohybem ikonky znázorňující novou závadu již problémy neměli.

Při testování Lo-Fi prototypu některým účastníkům testování dělalo problém správně nastavit stav závady a reakci na závadu. Na základě tohoto zjištění byly provedeny v Hi-Fi prototypu změny, které přispěly k tomu, že už žádnému uživateli tento úkol nedělal problém. Jediným problémem, který se při testování této obrazovky vyskytl, byl problém s klávesnicí mobilního telefonu, která zakryla některým uživatelům potvrzovací tlačítko.

5.4 Závěr

Na základě testování prototypu v Hi-Fi fázi bylo zjištěno, že do finální aplikace bude potřeba zapracovat pouze malé množství změn. Při testování prototypu v této fázi se ukázalo, že prototyp Hi-Fi se uživatelům ovládá mnohem lépe než prototyp Lo-Fi. Dalším zjištěním bylo, že ve fázi Lo-Fi všechny problémy odhalit nelze. Jako příklad může být otevření klávesnice mobilního telefonu, která zakryla tlačítko pro potvrzení akce. Tato fáze testování byla velice důležitá zejména proto, že v aplikaci se nachází mapa, kterou ve fázi Lo-Fi nelze téměř otestovat. Ovládání mapy testovaní uživatelé zvládli dobře.

Kapitola 6

Realizace

Cílem této kapitoly je vybrat vhodné technologie pro implementaci mobilní aplikace na hlášení závad. Na základě výsledků z podkapitoly o výběru technologie a kapitoly 5, věnující se testování Hi-Fi prototypu, bude provedena samotná implementace.

6.1 Výběr technologie

Jelikož v kapitole omezení (viz. 3.2) je požadavek na to, že aplikace má být pouze pro mobilní platformu Android, nabízí se dvě možnosti jak danou aplikaci vytvořit. Je možné vytvořit nativní aplikaci pro platformu Android. Druhou možností je využít některý z hybridních mobilních frameworků, které umožní vytvořit pouze jednu aplikaci, která bude fungovat na více platformách. V následujících kapitolách budou tyto dvě možnosti rozepsány detailněji.

6.1.1 Nativní aplikace

Nativní aplikace sebou nesou jisté výhody a to je například rychlost aplikace, která je způsobena tím, že programátor přímo přistupuje ke všem funkcionalitám platformy. V případě platformy Android se přistupuje ke všem funkcionalitám pomocí Android SDK. Tvorba nativních aplikací pro platformu Android probíhá v jazyce Java.

Nativní aplikace mají také tu výhodu, že standardní komponenty splňují všechny pravidla pro danou platformu. Toto je velice důležité zejména pro dodržení pravidel Nielsenovy heuristické analýzy (viz. 3.6), hlavně pak pravidla pro dodržování obecných a platformových standardů (viz. 3.6.4).

Vývoj nativní aplikace probíhá pomocí odladěných nástrojů pro vývoj a ladění. V případě platformy Android je vhodné pak využít IDE Android Studio[11]. Android studio umožňuje využít také nástroje pro rychlejší přístup do

[11] http://developer.android.com/sdk/index.html

dokumentace nebo nástroj na profilování.[11]

Nevýhodou vývoje a údržby nativních aplikací je nutnost znát specifický programovací jazyk a knihovny pro jednu mobilní platformu. Proto je cena vývoje větší. Samozřejmě je pak také mnohem dražší údržba více nativních aplikací. V případě nutnosti přidání další funkcionality je nutné požadovanou funkcionalitu implementovat opět na všech platformách.

6.1.2 Hybridní aplikace

[http://www.tecsolsoftware.com/blog/what-is-a-hybrid-mobile-app-development-and-when-should-you-consider-it/]

Hybridní mobilní aplikace jsou webové aplikace, které běží uvnitř nativní mobilní aplikace, která využívá *WebView*. *WebView* je komponenta, která umožňuje zobrazování webových aplikací. Hybridní aplikace navíc může využít funkce mobilního zařízení jako je fotoaparát, GPS nebo například akcelerometr.

Hybridní mobilní aplikace se vyvíjí pomocí JavaScriptu, HTML, CSS. Proto je možné sdílet některé části webových aplikací přímo s mobilními aplikacemi. Finální hybridní aplikace může být pak distribuována pomocí Google Play[12] nebo AppStore[13].

Hybridní aplikace mají výhodu toho, že je možné nasadit stejnou aplikace na více platformách. Toto sebou nese jistou nevýhodu toho, že některé funkcionality mohou být dostupné pouze pro jednu platformu a tak pokud chceme, aby aplikace fungovala správně na všech požadovaných platformách, musíme provést jisté úpravy. Oproti nativní aplikacím jsou hybridní aplikace pomalejší a to z důvodu nutnosti využívat další mezivrstvu, která umožní aplikaci fungovat na všech platformách.

Výhodou je také možnost vývoje v technologiích určených pro vývoj webových aplikací a není tak například pro firmu vyvíjející webovou aplikaci, potřeba další vývojář pro vývoj mobilní aplikace. S možností sdílet kód mezi webovými aplikacemi a webovou aplikací znamená, že je poté údržba mnohem snažší.

Nevýhodou hybridních aplikací je, že vzhled nemusí být stejný jak by byl za využití standardních komponent při vývoji nativní aplikace. Stejně tak hybridní aplikace nemusí fungovat stejně jako nativní aplikace. Problémem je také velice často nedostatečná dokumentace frameworků pro hybridní vývoj.[11]

6.1.2.1 Ionic

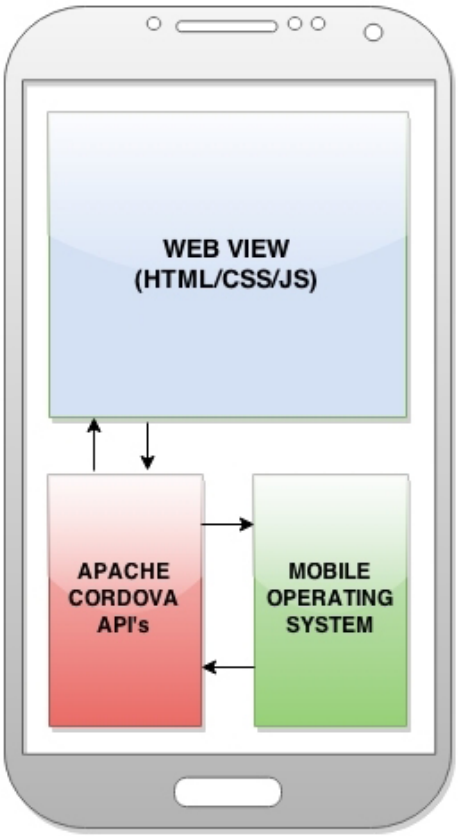
Ionic[14] je opensource SDK pro vývoj hybridních mobilních aplikací pomocí HTML5. Ionic poskytuje HTML, CSS a Javascriptové komponenty optimali-

[12] https://play.google.com/store
[13] https://itunes.apple.com/cz/genre/ios/
[14] http://ionicframework.com/

zované pro mobilní zařízení. Tyto komponenty tak například umožňují využívání gest.

Jako Javascriptový framework je v Ionic využit AngularJs[15]. Což je hodnotná informace zejména proto, že stávající webová aplikace tento framework také využívá a tak by mohly být některé komponenty využity.

Ionic je také velice dobře propojen s Apache Cordova[16]. Apache Cordova API umožňuje využívat nativní funkce zařízení. Aplikace tak například může využívat fotoaparát, Bluetooth, GPS apod.



Obrázek 6.1: Obr: Architektura Ionic frameworku (zdroj: [2])

Na obrázku 6.1 je možné vidět architekturu, kde jako WebView slouží framework Ionic, který pomocí Apache Cordova API využívá funkcionality mobilního operačního systému.

Pro dodržení platformových standardů (viz. 3.6.4) z Nielsenovy heuristiky je potřeba, aby aplikace využívala *Material Design*, který je nyní standardem pro platformu Android. Jelikož Ionic přímo nepodoporuje *Material Design*

[15]https://angularjs.org/
[16]https://cordova.apache.org/

bylo by potřeba při vývoji využít nějakou knihovnu, která by umožnila využít potřebné komponenty.

Jednou z knihoven, které by bylo možné využít pro komponenty z *Material Design* je knihovna *Ionic Material*[17]. Jedná se o knihovnu, která poskytuje základní komponenty z Material Design. Avšak knihovna je zatím pouze ve verzi Alfa. Poslední změna v repozitáři byla navíc provedena před 6 měsíci od tvorby tohoto textu. Může se tak stát, že knihovna už nebude dále vyvíjena což může způsobit problémy s dalším rozvojem aplikace.

6.1.3 Zhodnocení

Z prvopočátku se zdálo jako nejjednodušší řešení jít cestou vývoje hybridní mobilní aplikace za pomoci frameworku Ionic. Tento framework totiž používá k vývoji Javascriptový framework AngularJs. Výhoda využití frameworku AngularJS je zejména v tom, že stávající webová aplikace na hlášení závad využívá také AngularJs a tak by bylo možné některé komponenty využít jak pro webovou tak pro mobilní aplikaci. Ovšem po studiu pravidel pro vývoj aplikací pro platformu Android byla zjištěna potřeba využít *Material Design*. Framework Ionic sice umožňuje sice využít knihovnu pro *Material Design*, ale tato knihovna je zatím pouze v alfa verzi. S knihovnou by se pravděpodobně podařilo úspěšně aplikaci vytvořit, ale je velice pravděpodobné, že mohou nastat problémy s dalším rozvojem a případnými přechody na stabilnější verze knihovny.

Jelikož je cílem této práce vyvinout pouze aplikace pro platformu Android a na základě informací popsaných v předešlých odstavcích se jeví jako nejlepší řešení jít cestou vývoje nativní aplikace. Aplikace tak bude vyvíjena pomocí jazyka Java a Android SDK. Aplikace tak bude snadněji splňovat standardy pro Android aplikace a bude mít větší výkonnost, která bude potřebná při zobrazování velkého množství bodů v mapě (obrazovka 3.5a). Snadnější vývoj by také měl být díky využití Android Studia, které umožňuje využít nástroje k rychlejšímu vývoji. Jedná se například o emulátor, nástroje pro ladění a prostředí pro návrh jednotlivých obrazovek.

6.2 Implementace

V této podkapitole budou nejprve představeny stavební bloky Android aplikace. Poté bude proveden výběr vhodné technologie pro zobrazování map. Na konci této kapitoly bude popsána základní struktura implementované aplikace.

6.2.1 Aplikace Android

Tato kapitola popisuje stavební bloky Android aplikace a popisuje základní pojmy, které pak budou využity při samotné implementaci aplikace.

[17] http://ionicmaterial.com/

6.2.1.1 Soubor manifest

Každá Android aplikace obsahuje manifest soubor. Tento soubor se nachází v kořenovém adresáři aplikace pod názvem *AndroidManifest.xml* a slouží jako centrální bod deklarace aplikace. Manifest obsahuje definice všech aktivit (viz. níže), nastavení práv aplikace, hlavní aktivitu, minimální a maximální verzi Android SDK apod.[12]

6.2.1.2 Aktivita

Základním stavebním prvkem Android aplikací jsou aktivity. Aktivita slouží k reprezentaci jedné obrazovky v aplikaci. Aplikace nejčastěji obsahuje mnoho různých aktivit například pro zobrazování dat nebo získávání dat od uživatele. Jednou z aktivit je také hlavní aktivita, která je použita při spuštění aplikace. Aktivita je Java třída, která obsahuje aplikační logiku Android aplikace.[13]

6.2.1.3 View

Dalším základním stavebním prvkem Android aplikací je *View*. Jedná se o elementy uživatelského rozhraní, které se skládají ze základních stavebních prvků uživatelského rozhraní jako je například textové pole, obrázek, tlačítko apod. Definice *View* je ve formátu *XML*.[14]

6.2.1.4 Fragment

Fragment je komponentou využívanou v případě potřeby rozdělit funkcionalitu v jedné aktivitě. Aktivita pak může obsahovat jeden nebo více fragmentů na obrazovce. Fragmenty jsou nejčastěji využívání k rozdělení funkcionalit na velkých displejích. Fragmenty je také možné využít ve více aktivitách. Jedná se tedy o modulární sekci aktivity s vlastním životním cyklem a možností přijímat vlastní vstupní události.[15]

6.2.1.5 Intent

Intent slouží ke komunikaci mezi aktivitami v Android aplikacích. Mezi aktivitami uvnitř aplikace je možné si například posílat objekty obsahující data. Dále může *Intent* sloužit pro komunikaci s externími aplikacemi. Jako základní příklad využití pro komunikaci s externími aplikacemi je možnost spuštění webového prohlížeče nebo spuštění aplikace na pořízení fotografie nebo videa.

6.2.2 Mapy

Jelikož aplikace vyžaduje využití map pro zobrazování objektů a tvorbu nových objektů je nutné vybrat vhodnou knihovnu, která umožní snadnou implementaci. Při implementaci bylo rozhodováno zda využít Google Maps Android

API[18] nebo Mapbox Android SDK[19]. V této kapitole budou obě technologie stručně popsány a poté bude jedna z technologií vybrána pro implementaci.

6.2.2.1 Google Maps Android API

Pomocí Google Maps Android API je možné zobrazovat v nativních aplikacích mapy zakládající se na datech z Google Maps. API umožňuje zobrazování objektů v mapě jako jsou například body, křivky a mnohoúhelníky. Google Maps API také umožňuje zobrazovat v mapě pozici mobilního zařízení, včetně zobrazení informace o přesnosti získaných dat.[16]

Google maps poskytuje dvě podkladové mapy. Základní mapa obsahuje domy, silnice, řeky, názvy ulic, měst a obcí apod. Základní mapa bohužel neobsahuje čísla popisná domů, což je údaj, který pomáhá velkému množství uživatelů k orientaci v místě jejich bydliště. Další podkladovou mapou je mapa satelitní. Jedná se o satelitní snímky obsahující vyznačené silnice, názvy ulic, měst a obcí. Mapu obsahující satelitní snímky je možné detailně přiblížit a na mapě je tak možné rozpoznat objekty jako jsou stromy nebo zaparkovaná auta. Bohužel při velkém přiblížení je možné v některých místech rozpoznat neaktuálnost snímků. Pomocí aplikace Google Earth[20], která využívá stejné snímky jako Google Maps, bylo zjištěno že některé snímky byly naposledy aktualizovány v roce 2004.

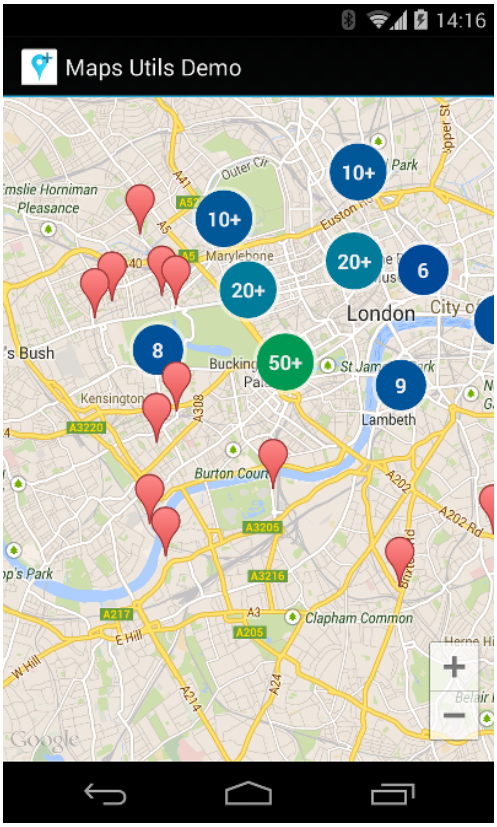
V dokumentaci k Google Maps Android API je popsána detailně každá komponenta s velkým množstvím příkladů. Velké množství příkladů existuje i mimo oficiální dokumentaci.

Důležitou knihovnou, kterou je vhodné využít k zobrazování dat pomocí Google Maps Android API je knihovna Google Maps Android API Utility Library[21]. Jedná se o knihovnu, která umožňuje v mapě zobrazovat data ve formátech GeoJson[22] a KML[23]. Knihovna dále poskytuje funkcionalitu na zobrazení velkého množství bodů v mapě. Na obrázku 6.2 je možné vidět zobrazení velkého množství bodů. V místě, kde je velká hustota bodů při daném přiblížení, se zobrazí informace o počtu a v místě kde je malé množství se zobrazí samotné body. Tato funkcionalita je z pohledu uživatelského rozhraní přehlednější a také není tak výkonnostně náročná na samotné mobilní zařízení.

6.2.2.2 Mapbox Android SDK

Mapbox Android SDK je knihovna pro zobrazování map na mobilních zařízeních Android. Jedná se o mladou knihovnu, která byla uveřejněna v roce

[18] https://developers.google.com/maps/documentation/android-api/
[19] https://www.mapbox.com/android-sdk/
[20] http://www.google.cz/intl/cs/earth/
[21] https://developers.google.com/maps/documentation/android-api/utility/
[22] formát pro zobrazování geoprostorových dat
[23] formát pro zobrazování geoprostorových dat od Google



Obrázek 6.2: Obr: Zobrazení velkého množství bodů v mapě (Zdroj: [3]

2015. Tato skutečnost může být zejména problém z důvodu využití dokumentace a neoficiálních návodů. Mapbox[24] se snaží touto knihovnou konkurovat Google Maps, zejména pak cenami a pravidly využívání. Využívání mapy je do 50 000 mobilních uživatelů zdarma, což by pro potřeby této aplikace mělo být dostačující. Po přesažení zmíněného počtu uživatelů jsou ale ceny neustále velice dostupné.

V mapě je možné zobrazovat standardní prvky jako jsou body, křivky a mnohoúhelníky. Zobrazování formátů ve formátech GeoJson nebo KML zatím možné není. Načíst data z těchto formátů však možné je a to pomocí parsování Json resp. XML formátu a poté i přidání obsažených objektů do mapy.

Mapbox Android SDK poskytuje jako podklad standardní mapu a satelitní mapu. Standardní mapa vychází z pokladů OpenStreetMap[25] a obsahuje domy, silnice, řeky, názvy měst, názvy ulic a také čísla popisná. Satelitní mapa je bohužel zatím v České Republice téměř nepoužitelná, protože velké množství

[24] https://www.mapbox.com/
[25] https://www.openstreetmap.org

snímků je překryto mraky a bohužel není také možné využít velké přiblížení. Například v Německu jsou satelitní snímky mnohem kvalitnější.

Jednou ze zajímavých funkcionalit, kterou Mapbox Android SDK umožňuje je zobrazení mapy v offline režimu, tedy bez připojení k internetu. Jedná se o funkcionalitu, která byla uveřejněna v březnu roku 2016 a zatím se jedná pouze o beta verzi.

Mapbox Android SDK bohužel zatím neposkytuje funkcionalitu pro zobrazení velkého množství bodů v mapě jakou má například Google Maps Android API (viz. 6.2.2.1).

6.2.2.3 Výhody využití Google Maps Android API

Google Maps Android API je prověřenou knihovnou, která je de facto standardem pro zobrazování map na platformě Android. Tato knihovna je velice dobře zdokumentovaná s velkým počtem návodů oficiálních i neoficiálních. Google Maps poskytuje kvalitnější satelitní snímky oproti Mapbox Android SDK, i přesto že v některých místech jsou velice neaktuální. Další výhodou je možnost snadnějšího načítání formátů GeoJson nebo KML, který je i součástí API současné aplikace na hlášení závad. S Google Maps Android API je možné využít také knihovnu Google Maps Android API Utility, která umožňuje využít funkcionalitu na zobrazení velkého množství bodů v mapě, což vyžaduje i charakter vyvíjené aplikace.

6.2.2.4 Výhody využití Mapbox Android SDK

Výhodou využití Mapbox Android SDK je využití podkladových map OpenStreetMap, které obsahují i čísla popisná, která v Google Maps obsažena nejsou. Další výhodou je možnost využít funkcionality na zobrazení map v režimu offline (zatím se funkcionalita nachází pouze o beta verzi).

6.2.2.5 Zhodnocení

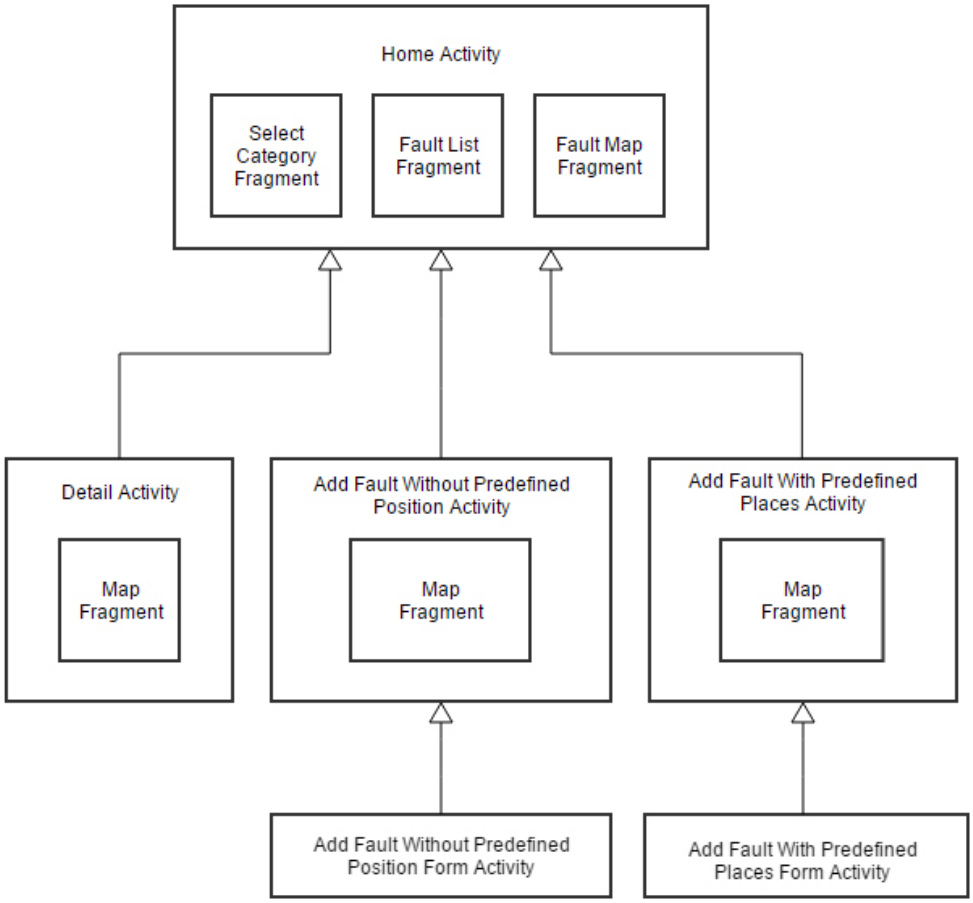
Na základě shrnutých výhod popsaných v kapitolách 6.2.2.3 a 6.2.2.4 vyplývá, že bude vhodnější pro implementaci využít knihovnu Google Maps Android API. Knihovna Mapbox Android SDK však bude do budoucna velice dobře konkurenceschopná, protože na hodně funkcionalitách se v této době pracuje. Jedná se zejména o funkcionalitu zobrazení velkého množství bodů v mapě. Stejně tak se dle oficiální dokumentace Mapbox pracuje na zkvalitnění satelitních snímků. Některé funkcionality jsou již v beta verzi a to zejména zobrazení map v offline režimu.

6.2.3 Struktura aplikace

Aplikace byla rozdělena do několika aktivit na základě návrhu uživatelského rozhraní. Na obrázku 6.3 je možné vidět strukturu celé aplikace, včetně rozdě-

lení některých aktivit do více fragmentů. Směr šipky na obrázku znázorňuje, která z aktivit je rodič. Rodičovská aktivita je aktivita, která v hierarchii aplikace je na vyšší úrovni než její potomek. Tato hierarchie je zejména důležitá při použití tlačítka zpět (na obrazovce nebo jako hardwarové tlačítko), kdy při stisknutí tlačítka je zobrazena aktivita která je rodičem dané aktivity. Pokud tedy je stisknuto tlačítko zpět v aktivitě *Detail Activity*, tak aplikace přejde na *Home Activity*.

Dále v této kapitole budou popsány pouze aktivity, ve kterých se vyskytuje nějaký důležitý nebo zajímavý způsob implementace.



Obrázek 6.3: Obr: Struktura aplikace

6.2.3.1 Home Activity

Home Activity je aktivita, která má za úkol zobrazit úvodní obrazovku. Jelikož úvodní obrazovka obsahuje spodní navigaci a zatím se nejedná o standardní komponentu, která by byla obsažena v Android SDK, bylo potřeba využít kni-

hovnu pro zobrazení této navigace. Na základě provedeného průzkumu byla využita knihovna BottomBar[26]. Jedná se o knihovnu, která obsahuje komponentu spodní navigace s tím, že dodržuje pravidla pro aplikace využívající *Material Design*.

Jelikož úvodní obrazovka obsahuje několik záložek, mezi kterými je možné přecházet pomocí spodní navigace, byly jednotlivé záložky rozděleny do fragmentů.

Select Category Fragment má za úkol zobrazit jednotlivé kategorie pro vytvoření nové závady. Pro zobrazení seznamu kategorií bylo využito komponenty *CardView*.

Fault List Fragment obsahuje seznam již nahlášené závady. Pomocí kliknutí na jednotlivé položky v seznamu je možné rovnou přejít do detailu závady. V seznamu závad je také možné vyfiltrovat pouze závady ohlášené z daného mobilního zařízení.

Fault Map Fragment zobrazuje již ohlášené závady přímo v mapě. V mapě je možné vyfiltrovat pouze závady ohlášené z daného mobilního zařízení nebo také změnit podkladovou mapu na satelitní. Po kliknutí na jednotlivé závady v mapě se rovnou přejde do detailu závady.

6.2.3.2 Add Fault With Predefined Places Activity

Jedná se o aktivitu sloužící k výběru porouchaného objektu. Obsahem obrazovky je mapa obsahující jednotlivé objekty z dané kategorie, na kterých je možné závadu ohlásit. Problémem na této obrazovce je to, že v mapě může být potřeba zobrazit velké množství objektů. Což může být problém zejména v případě, že mapa není příliš přiblížená (je malý zoom) a je potřeba zobrazit všechny objekty v mapě. Objekty pak jsou zobrazeny všechny téměř ve stejném místě což z pohledu uživatele je velice nepřehledné a z pohledu výkonnosti aplikace velice náročné. Pro účely řešení daného problému je v aplikaci využita knihovna *Google Maps Android API Utility* popsaná v kapitole 6.2.2.1. Data jsou z aplikačního rozhraní načteny ve formátu KML a poté jsou všechny objekty předány třídě *Cluster Manager*, která zajišťuje přehledné a rychlé zobrazení i přes velký počet objektů.

6.2.3.3 Add Fault With Predefined Places Form Activity

Jedná se o aktivitu, ve které má uživatel za úkol vyplnit formulář pro vytvoření nové závady. V této aktivitě už je známá informace o tom do jaké kategorie závada patří a kde se závada nachází. Uživatel kromě vyplnění formuláře může také pořídit fotografii. Pořízení fotografie probíhá pomocí aplikace fotoaparát, které je pouze z této aktivity předán *Intent* s požadavkem pro pořízení fotografie a souborem kam má být fotografie uložena. Po pořízení fotografie je

[26] https://github.com/roughike/BottomBar

aktivita o této zkutečnosti informována a může dále pokračovat ve své práci. V případě této aplikace se pořízená fotografie vezme a zobrazí se uživateli.

Stejně jako popisovaná aktivita také funguje *Add Fault Without Predefined Position Form Activity*, která je postavená na stejném základu a jsou v ní provedené jen malé změny.

6.2.4 Proces vytvoření závady

Při vytváření závady je nutné projít přes tři aktivity (výběr kategorie, výběr pozice závady a vyplnění formuláře). Pro přehlednost jak probíhá samotná tvorba závady byl vytvořen obrázek 6.4 zobrazující tvorbu ukázkové závady na lampě. V obrázku je vidět, že nejprve *Home Activity* vytvoří *Intent* obsahující prázdnou závadu pouze s nastavenou kategorií závady. V *Add Fault With Predefined Places Activity* se k závadě doplní informace o pozici závady a vytvoří se vazba na objekt, na kterém má být závada ohlášena. Závada s doplněnými informacemi je v *Intentu* odeslána aktivitě *Add Fault With Predefined Places Form Activity*. V této aktivitě jsou k závadě doplněny další informace jako název a popis závady a kontaktní údaje. Dále je k závadě vytvořena jedna fotografie. Tyto informace jsou odeslány pomoci Json objektu samotnému API webové aplikace na hlášení závad.

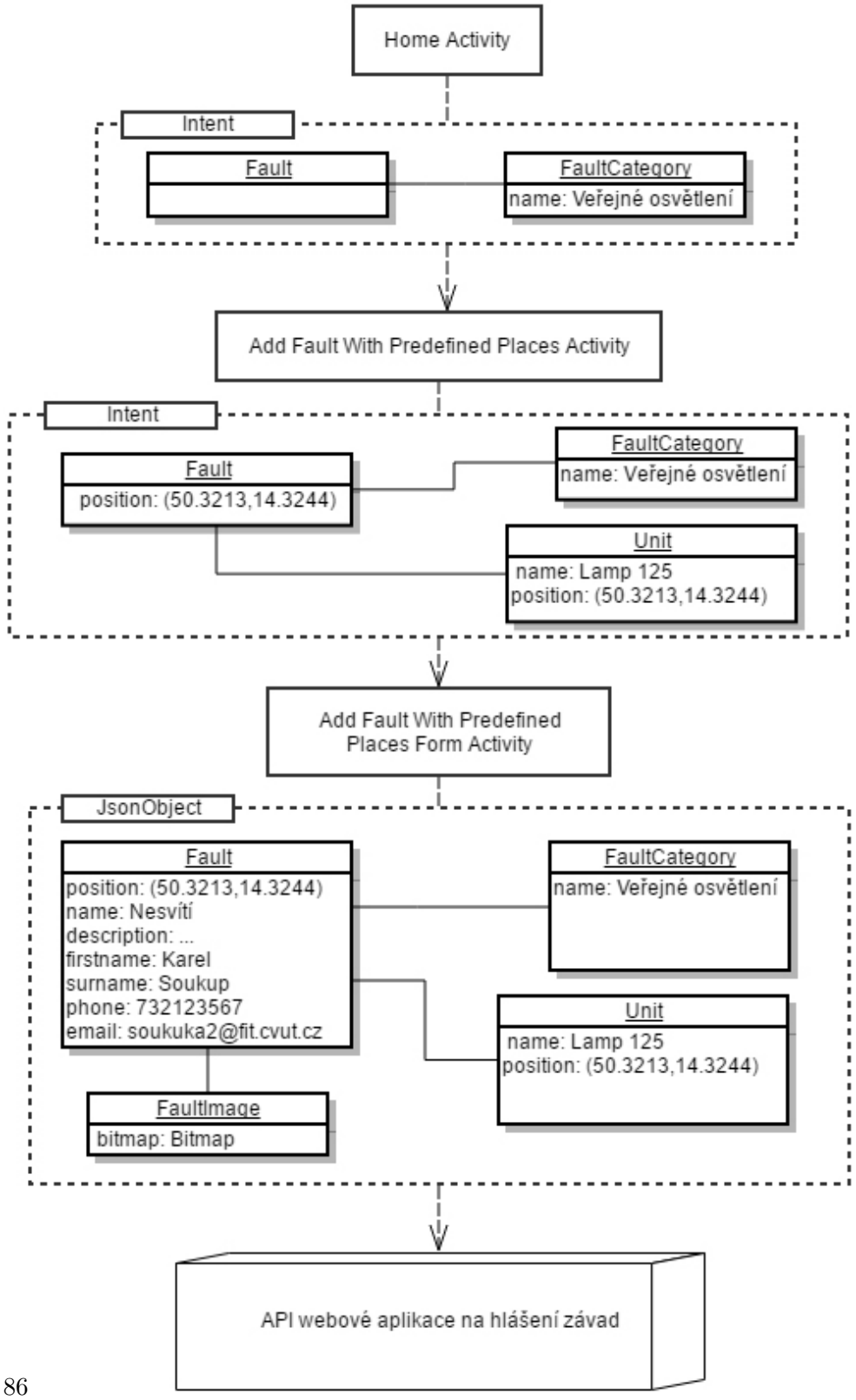
6.3 Závěr

Cílem této kapitoly bylo vybrat vhodné technologie a na základě toho provést samotnou implementaci.

Jelikož aplikace má být vytvořena pouze pro platformu Android, bylo zvoleno jít cestou vývoje nativní aplikace za pomocí Android SDK. Na základě tohoto rozhodnutí jsou pak v této kapitole popsány jednotlivé stavební bloky nativních Android aplikace.

V této kapitole se také nachází podkapitola zabývající se výběrem vhodné technologie pro zobrazování map. Podkapitola se zabývá výběrem mezi standardní knihovnou Google Maps Android API a poměrně mladou knihovnou Mapbox Android SDK. Zvolena nakonec byla knihovna Google Maps Android API, zejména kvůli své dokumentaci, kvalitě mapových podkladů a možnosti využít knihovnu Google Maps Androi API Utility, která umožňuje v mapě zobrazit velké množství objektů.

Na závěr kapitoly se nachází popis struktury aplikace a jednotlivých Aktivit.



Obrázek 6.4: Obr: Ukázka komunikace mezi aktivitami a samotným API

Závěr

Cílem této diplomové práce bylo vytvoření mobilní aplikace na hlášení závad pro platformu Android.

Nejprve je v práci představena webová aplikace na hlášení závad ve městě, která slouží k mobilní aplikaci jako server poskytující aplikační rozhraní pro tvorbu a správu závad.

V analytické části je představen *Material Design*, tedy vizuální jazyk, který by měl být dodržován při vývoji mobilních aplikací na platformě Android. V analytické části se také nachází analýza existujících řešení. V této části byly vyzkoušeny tři aplikace. Na základě průzkumu existujících aplikací bylo například navrženo využití při výběru pozice závady tzv. zaměřovač, který umožní citlivější lokalizaci v mapě.

Pro návrh aplikace byl nejprve proveden průzkum uživatelů stávající webové aplikace. Na základě výsledků průzkumu byly vytvořeny uživatelské role a došlo k definici cílové skupiny. Cílovou skupinou aplikace jsou občané, kteří hlásí závady alespoň jednou za 14 dní. Jedná se o 19% občanů, kteří využívají stávající webovou aplikaci.

Po sepsání případů užití je vytvořen Lo-Fi prototyp obsahující všechny důležité obrazovky aplikace. Některé části prototypu jsou otestovány nejprve pomocí uživatelských dotazníků a poté pomocí tzv. klikacího testu[27]. Již z výsledků klikacího testu bylo jasné, že je něco v nepořádku s navrženou úvodní obrazovkou aplikace. Nedostatky na úvodní obrazovce byly potvrzeny také pomocí uživatelského testování, které bylo provedeno na 7 osobách.

Na základě výsledků provedených testů navrženého Lo-Fi prototypu byly provedeny první změny. Jednou ze změn bylo také odstranění *FAB* tlačítka (viz. kapitola 2.1.2.4). Tyto změny byly poté opět otestovány pomocí uživatelského testování. Lo-Fi prototyp obsahující změny již neobsahoval zásadnější problémy a proto byl dle tohoto návrhu vytvořen Hi-Fi prototyp, obsahující i zapracované připomínky.

[27] uživateli se vzdáleně zobrazí návrh obrazovky a úkol, uživatel na základě úkolu poté klikne do navržené obrazovky

Úkoly pro uživatelské testování Hi-Fi prototypu byly ponechány téměř stejné jako při testování Lo-Fi prototypu. Došlo pouze ke změnám za účelem testování uživatelů, zda dokáží pracovat s mapou. Tato skutečnost totiž nebyla možná testovat v Lo-Fi fázi. Výsledkem Hi-Fi testování bylo, že do finální aplikace bude potřeba zapracovat pouze malé množství změn.

V kapitole realizace byla nejprve pro vývoj aplikace vybrána technologie Android SDK. Důvodem bylo, že cílem této práce je pouze vytvořit aplikaci pro platformu Android a také kvalita poskytovaných nástrojů a dokumentace.

Na základě zvolené technologie bylo následně potřeba ještě vybrat správnou knihovnu pro zobrazování map. Vybrána byla knihovna Google Maps Android API, však konkurenční knihovna Mapbox Android SDK je rovnocenným soupeřem, která v některých odvětvích poskytuje dokonce lepší možnosti. Zejména umožňuje zobrazení map v režimu bez připojení k internetu. Google Maps Android API nakonec byla vybrána z důvodu kvality mapových podkladů a kvůli knihovně umožňující zobrazit velké množství objektů v mapě.

Poslední část kapitoly realizace se zabývá samotnou implementací finální aplikace.

Samotná práce pro mě byla přínosem zejména z důvodu proniknutí do vývoje mobilních aplikací, se kterým jsem doposud žádné zkušenosti neměl. Další cennou zkušeností pro mě bylo také testování návrhu uživatelského rozhraní, které poskytlo velice hodnotnou zpětnou vazbu.

Jako rozšíření pro mobilní aplikaci by bylo vhodné například implementovat možnost práce s aplikací v režimu offline, včetně zobrazení map.

Literatura

[1] What is material? [online]. [Cit. 2016-01-15]. Dostupné z: https://www.google.com/design/spec/what-is-material/

[2] Phan, H.: *Ionic Cookbook*. Birmingham: Packt Publishing, 2015, ISBN 978-1-78528-797-8.

[3] Google Maps Android API Utility Library [online]. Únor 2016, [Cit. 2016-04-10]. Dostupné z: https://developers.google.com/maps/documentation/android-api/utility/

[4] Clifton, I.: *Android user interface design : implementing material design for developers*. New York: Addison-Wesley, 2016, ISBN 978-0-134-19140-9.

[5] Material design Introduction [online]. [Cit. 2016-01-10]. Dostupné z: https://www.google.com/design/spec/material-design/introduction.html

[6] Ruiz, A.: *Mastering Android application development : take your Android knowledge to the next level with this advanced Android application guide, which shows you how to make even better Android apps that users will love*. Birmingham, UK: Packt Publishing, 2015, ISBN 978-1-78588-422-1.

[7] Material design Bottom navigation [online]. [Cit. 2016-01-20]. Dostupné z: https://www.google.com/design/spec/components/bottom-navigation.html

[8] Nielsen, J.: 10 Usability Heuristics for User Interface Design [online]. Leden 1995, [Cit. 2016-03-01]. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/

[9] Žikovský, P.: Návrh uživatelských rozhraní [online]. 2011, [Cit. 2016-03-10]. Dostupné z: https://edux.fit.cvut.cz/courses/MI-NUR/_media/lectures/x09-opakovani.pdf

[10] Krug, S.: *Nenuťte uživatele přemýšlet! : praktický průvodce testováním a opravou chyb použitelnosti webu*. Brno: Computer Press, 2010, ISBN 978-80-251-2923-4.

[11] Ricker, K.: Mobile App Development: How to Decide on Hybrid vs. Native [online]. Prosinec 2015, [Cit. 2016-04-01]. Dostupné z: `http://www.goxuni.com/670364-mobile-app-development-hybrid-versus-native/`

[12] Allen, G.: *Beginning Android*. Berkeley, CA New York, NY: Apress,Distributed to the book trade worldwide by Springer, 2015, ISBN 978-1-4302-4687-9.

[13] Boyer, R.: *Android application development cookbook : over 100 recipes to help you solve the most common problems faced by Android developers today*. Birmingham, UK: Packt Publishing, 2016, ISBN 978-1-78588-619-5.

[14] MacLean, D.: *Pro Android 5*. New York: Apress, 2015, ISBN 978-1-4302-4680-0.

[15] Fragments | Android Developers [online]. [Cit. 2016-02-18]. Dostupné z: `http://developer.android.com/guide/components/fragments.html`

[16] Introduction to the Google Maps Android API [online]. Únor 2016, [Cit. 2016-03-22]. Dostupné z: `https://developers.google.com/maps/documentation/android-api/intro`

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

XML Extensible markup language

dp Density-independent pixel

Lo-Fi Low fidelity

Hi-Fi High fidelity

IDE Integrated Development Environment

SDK Software development kit

HTML HyperText Markup Language

CSS Cascading Style Sheets

API Application Programming Interface

KML Keyhole Markup Language

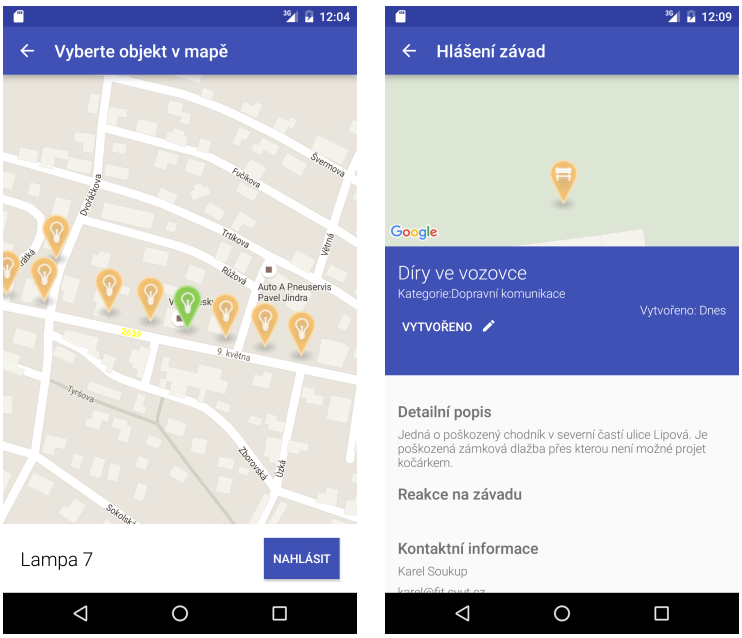
JSON JavaScript Object Notation

GPS Global Positioning System

PDF Portable Document Format

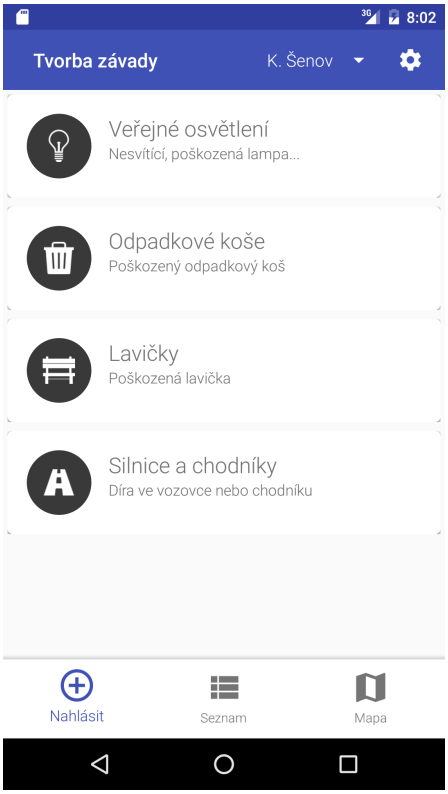
Příloha B

Snímky obrazovek z aplikace



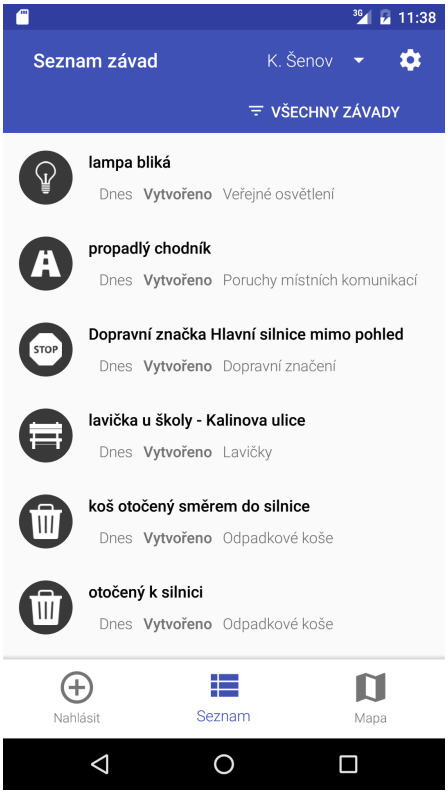
(a) Výběr poškozené lampy (b) Detail závady

Obrázek B.1: Obrazovky aplikace



(a) Výběr kategorie

(b) Mapa závad



(c) Seznam závad

Obrázek B.2: Úvodní obrazovky

Příloha C

Uživatelské testování Lo-Fi (1.kolo)

Účastník 6:

- **Scénář S1:** Uživatel nejprve klikl do seznamu závad na již ohlášenou závadu na veřejném osvětlení. Jelikož prototyp neumožňoval zobrazit právě tuto nahlášenou závadu, testovaný uživatel začal hledat místo závady v mapě. V mapě místo závady nenašel, proto začal zkoušet různá tlačítka, která se nachází na úvodní obrazovce. Během hledání narazil na FAB tlačítko a úspěšně tak překonal úvodní obrazovku. Zbytek úkolu již zvládl v pořádku.
- **Scénář S2:** Uživatel úkol zvládl v pořádku. Pro potvrzení pozice využil správně navržené tlačítko.
- **Scénář S3:** V pořádku.
- **Scénář S4:** V pořádku (závadu zobrazil přes seznam závad). Uživatel myslel, že když přidá reakci na závadu, je závada nastavena jako vyřešená.
- **Scénář S5:** Účastník testu neočekával, že vrstvu bude možné nastavit v nastavení. Očekával spíše ikonku. Úkol ale na konce zvládl.

Tabulka C.1: Profil 6. účastníka uživatelského testování

Pohlaví	Muž
Věk	46 let
Zkušenost s webovou aplikací na hlášení závad	Ne
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ne

Tabulka C.2: Profil 7. Účastníka uživatelského testování

Pohlaví	Muž
Věk	61 let
Zkušenost s webovou aplikací na hlášení závad	Ne
Zkušenost s mobilní aplikací na hlášení závad	Ne
Účastnil se někdy uživatelského testování	Ne

Shrnutí diskuze s účastníkem: Účastník opět konstatoval, že největším problém bylo prvotní vytvoření závady. Úvodní obrazovka působila na účastníka zmateně. Uživateli se nelíbila ani změna vrstvy mapy.

Účastník 7:

- **Scénář S1:** Uživatel po prohlédnutí úvodní obrazovky klikl v seznamu závad na již nahlášenou závadu s popisem *nesvítí*. Po zjištění, že se jedná o detail již nahlášené závady, vrátil se účastník testu zpět na úvodní obrazovku. Účastník prohlásil, že již neví jak dál. Byl proto posunut do následujícího kroku (přepnut na obrazovku kategorií závad). Zbytek úkolu již zvládl v pořádku.
- **Scénář S2:** V pořádku.
- **Scénář S3:** Uživatel nejprve klikl do nastavení v pravém horním rohu. Poté klikl na seznam obcí a úkol tedy zvládl.
- **Scénář S4:** Uživatel nejprve přidal k závadě komentář a myslel si, že tím je úkol splněn. Po upozornění účastníka, že splnil pouze jednu část úkolu, změnil účastník stav závady, čímž úkol dokončil.
- **Scénář S5:** V pořádku.

Shrnutí diskuze s účastníkem: Jednalo se o nejstaršího účastníka testu. Účastník sice má telefon se systémem Android, ale zatím pouze krátkou dobu. FAB tlačítko označil jako tlačítko, od kterého neví co má čekat. Potřeboval by pro přehlednost nějaký popis. Úvodní obrazovku účastník testu označil za zmatenou.

Příloha D

Uživatelské testování Hi-Fi

Účastník 6 Profil účastníka je možné vidět v tabulce C.1

- **Scénář S1:** V pořádku
- **Scénář S2:** Účastník nejprve zapomněl pořídit fotografii, takže musel úkol opakovat.
- **Scénář S3:** V pořádku
- **Scénář S4:** Účastník měl problém s kliknutím na tlačítko pro změnu stavu závady (příliš malá dotyková plocha).
- **Scénář S5:** V pořádku
- **Scénář S6:** V pořádku

Shrnutí diskuze s účastníkem: Účastník měl zejména problémy se změnou stavu závady, protože se mu nedařilo kliknout na požadovanou ikonku.

Účastník 7 Profil účastníka je možné vidět v tabulce C.2

- **Scénář S1:** V pořádku
- **Scénář S2:** V pořádku (pozici závady potvrdil kliknutím na ikonku)
- **Scénář S3:** V pořádku
- **Scénář S4:** Účastník nejprve na úvodní obrazovce zvolil kategorii závady. Na obrazovce obsahující mapu zjistil že se je na obrazovce kde se vytváří nová závada. Použil tlačítko zpět a zobrazil seznam závad, kde zvolil správnou závadu a zbytek úkolu již zvládl v pořádku.
- **Scénář S5:** V pořádku

- **Scénář S6:** V pořádku

Shrnutí diskuze s účastníkem: Tento uživatel se neúčastnil druhého kola testování Lo-Fi prototypu. Podle něj určitě došlo ke změnám k lepšímu a nyní je systém mnohem lépe ovladatelný než byl při první testování.

Příloha E

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `apk` ..... adresář obsahující aplikaci
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF