VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILNÍ APLIKACE PRO SDÍLENÍ POLOHY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE LUKÁŠ TEUER
AUTHOR

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILNÍ APLIKACE PRO SDÍLENÍ POLOHY

MOBILE APPLICATION FOR POSITION SHARING

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE LUKÁŠ TEUER

AUTHOR

VEDOUCÍ PRÁCE Ing. RADEK BURGET, Ph.D.

SUPERVISOR

BRNO 2015

Abstrakt

Tato práce se zabývá návrhem a implementací mobilní aplikace, která umožňuje v reálném čase získat a sdílet informace o poloze uživatele s jinými uživateli. V práci je popsána problematika získávání a sdílení polohy a jsou také popsány důležité technologie a prostředky, které byly použity během implementace. Obsažen je také popis návrhu a implementace webové aplikace pro zobrazení dat získaných z mobilní aplikace. Na konci je také zahrnuto porovnání s již existujícími mobilními aplikacemi.

Abstract

The thesis describes the design and implementation of mobile application, which enables gathering and sharing of user positions with other users. The paper describes the problems of learning and sharing positions and also described important technologies and tools that were used during implementation. Also included is the design and implementation of web application, which displays data gathered from the mobile application. At the end is also a comparison with the already existing mobile applications.

Klíčová slova

Mobilní zařízení, sdílení polohy, Android, GPS, Facebook,Java, JavaScript

Keywords

Mobile device, position sharing, Android, GPS, Facebook, Java, JavaScript

Citace

Lukáš Teuer: Mobilní aplikace pro sdílení polohy, bakalářská práce, Brno, FIT VUT v Brně, 2015

Mobilní aplikace pro sdílení polohy

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Radka Burgeta, Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Lukáš Teuer
17.5.2015

Poděkování

Děkuji mému vedoucímu Ing. Radku Burgetovi, Ph.D. za odborné vedení, cenné rady a trpělivost, kterou mi v průběhu zpracování bakalářské práce poskytl.

©Lukáš Teuer, 2015
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod ... 3
2 Použité technologie ... 5
2.1 GPS ... 5
2.2 OpenShift ... 5
2.3 OpenStreetMaps ... 6
2.3.1 Osmdroid ... 6
2.3.2 OpenLayers ... 7
2.4 Facebook ... 7
2.4.1 Facebook SDK pro Android ... 8
2.4.2 Facebook SDK pro JavaScript ... 8
2.4.3 Facebook SDK pro PHP ... 8
3 Cloud computing ... 10
3.1 Základní vlastnosti cloud computingu ... 10
3.2 Modely pro cloud computing ... 11
3.2.1 IaaS ... 11
3.2.2 SaaS ... 12
3.2.3 PaaS ... 12
4 Android ... 14
4.1 Hlavní Android komponenty ... 14
4.1.1 Aktivita a intent ... 14
4.1.2 Služba ... 15
4.1.3 Broadcast Receiver ... 15
4.1.4 Content Provider ... 15
4.2 Základní grafické prvky ... 16
4.2.1 View ... 16
4.2.2 Layout ... 16
4.2.3 Widget ... 16
4.3 Android Manifest ... 16
4.4 Životní cyklus aplikace ... 17
5 Návrh práce ... 19
5.1 Mobilní aplikace ... 19
5.1.1 Registrace a přihlašování ... 20
5.1.2 Mapa ... 20

5.1.3 Přátele 22
5.1.4 Nastavení 24
5.2 Serverová část 24
5.2.1 Komunikace aplikace se serverem 24
5.2.2 Databáze 25
5.2.3 Webové stránky 25
6 Implementace 27
6.1 Mobilní aplikace 27
6.1.1 Android Manifest 27
6.1.2 Engine.java 28
6.1.3 Datové třídy 28
6.1.4 Komunikace se serverem 29
6.1.5 Spuštění aplikace 30
6.1.6 Registrace a přihlašování 30
6.1.7 Aktivita s mapou 31
6.1.8 Aktivita se seznamem přátel 33
6.1.9 Nastavení 36
6.2 Serverová část 37
6.2.1 Databáze 37
6.2.2 Webové stránky 38
7 Porovnání s existujícími aplikacemi 41
7.1 Glympse 41
7.2 Pathshare 41
8 Závěr 43
9 Literatura 44

A Obsah DVD 45

1 Úvod

Získávání informací o poloze osob pomocí může být v současné době využito různými způsoby. Sdílením těchto informací ve formě pozic a ušlých tras se prakticky vytrácí možnost ztracení i v případě větší skupiny lidí. Velmi se tím také usnadňuje a zrychluje hledání ostatních osob v okolí. Možnost zjištění aktuální pozice bez složitého vysvětlování, nebo sledování trasy někoho, kdo zrovna cestuje či řídí a nemůže volat, jsou jen jedny z mnoha dalších možností, kterýma se dají tyto informace využít. I když některým lidem může být toto shromažďování dat nepříjemné a mohou se obávat zneužití těchto citlivých informací, pozitivní přínosy značně přesahují ty negativní.

Cílem této práce bylo proto vytvořit mobilní aplikaci, která umožňuje získávat informace o poloze a předávat je ostatním lidem, které si uživatel sám vybere a to vše v reálném čase. Bylo zapotřebí nastudovat metody získávání a sdílení dat a jejich problematiku. Dále bylo potřeba vymyslet připojování přátel, a to jak ze sociálních sítí kvůli většímu komfortu, tak bez nich za pomocí vyhledávání. Webové stránky, které umožňují přehled nad polohami a trasami byly také součástí práce.

V následující kapitole se seznámíme s technologiemi použitými při tvorbě této práce. Kapitola bude obsahovat krátké seznámení s těmito technologiemi a také zde bude popsán důvod výběru některých prvků.

Třetí kapitola se zaobírá tématem cloud computingu, což značí poskytování serverových či hostingových služeb přes internet. Proberou se mimo jiné i základní typy těchto služeb.

Ve čtvrté kapitole je přiblíženo programování pro platformu Android. Obsahuje seznámení se základními komponentami Androidu a obeznámení s životním cyklem aplikace. Pro pochopení některých pojmů z kapitoly o implementaci je doporučeno přečíst si tuto kapitolu.

Návrh práce bude obsahem páté kapitoly. Bude obsahovat seznámení s prací z funkčního hlediska, to znamená popsání, jak fungují jednotlivé prvky programů, které jsou dostupné uživateli a jak se používají. Tato kapitola neobsahuje implementační detaily a zaměřuje se pouze na obecný popis.

V šesté kapitole bude popsána implementace prvků zmíněných v předchozí kapitole. Obsahovat bude i popis implementace prvků a funkcí, které nebyly zmíněny, ale přesto jsou důležité pro správné fungování programů.

Sedmá kapitola obsahuje porovnání s vybranými aplikacemi zabírajícími se stejným tématem. Jsou zhodnoceny jejich klady oproti této práci a opačně i klady této práce.

2 Použité technologie

V této kapitole se seznámíme s některými technologiemi či jinými prvky, které jsou využívány k provozu či tvorbě buď mobilní aplikace, webových stránek nebo databáze. Popíšeme, proč byly jednotlivé technologie vybrány a také jejich základní principy a vlastnosti. Technologie sloužící přímo k získávání polohy (jako je třeba GPS) popíšeme přímo v kapitole *Získávání polohy*.

2.1 GPS

GPS je družicový systém, který umožňuje lokalizovat zařízení s přesností od několika metrů. Jeho princip je založen na měření času. GPS satelity neustále vysílají zprávy, které obsahují čas odeslání zprávy, dobu odeslání a pozici satelitu. GPS přijímač umí z těchto informací vypočítat dobu, kterou zpráva cestovala k zařízení, a tedy i vzdálenost od satelitu (zpráva cestuje rychlostí světla). Pokud má přijímač informace od alespoň čtyř satelitů, umí díky jejich pozicím a vzdálenosti vypočítat svoji vlastní. Tento výpočet zjistí zeměpisnou šířku a délku, nadmořskou výšku a umí odhadnout i přibližně jak přesné tyto hodnoty můžou být.

V současné době se ve vesmíru vyskytuje 32 satelitů ve výšce přibližně dvaceti tisíc kilometrů. Satelity krouží po osmi eliptických dráhách okolo Země, kde každý dráha je obsazena čtyřmi satelity, které jsou od sebe rovnoměrně vzdálené. Díky tomuto uskupení je vždy přibližně devět satelitů viditelných z jakéhokoli bodu na Zemi.

2.2 OpenShift

OpenShift je Platform-as-a-Service (PaaS) společnosti RedHat. Umožňuje programátorům hosting jejích aplikací online. Obsahuje podporu pro různé druhy databází a jazyků. Také má přímo zabudované větší množství *frameworků* a jiných různých nástrojů.

OpenShift nabízí podporu PHP a JavaScriptu, což jsou dva hlavní jazyky, které jsou použity pro tvorbu webových stránek této práce. Umožňuje také vytvoření databáze, v našem případě MySQL, a k ní připojit nástroj PHPMyAdmin, který umožňuje snadnou manipulaci s daty v databázi a celkově jednoduchou a intuitivní práci s databází. Povoluje také využít dostatečné množství datové prostoru jak pro webové stránky, tak pro uchování všech dat

(uživatelé, pozice uživatelů atd.). Je také zcela zdarma do určitého limitu současných připojení. Samotná práce s OpenShiftem je také velmi intuitivní a je k ní dostatečné množství materiálu.

Mezi nejznámější alternativy k Openshiftu patří Heroku, Google App Engine nebo Amazon Elastic Compute Cloud. I když Heroku má všeobecně lepší hodnocení, na rozdíl od OpenShiftu ho není možné provozovat zdarma a nemá ani trial verzi. Oproti Amazonu lze OpenShift zase mnohem rychleji zprovoznit a Google App Engine v základu nepodporuje MySQL databáze.

2.3 OpenStreetMaps

Pro základní fungování jakékoli aplikace, která pracuje s pozicemi a trasami, je nutné, aby obsahovala prvek s mapou, na kterém by se tyto pozice zobrazily. Existuje však více druhů map, které se dají používat. Mezi nejznámější pravděpodobně patří Google Maps od společnosti Google a OpenStreetMaps.

Jeden z hlavních rozdílů mezi těmito mapami je, že pro používání Google Maps musíte podepsat jejich smluvní podmínky, zatímco OpenStreetMaps mapy jsou přístupné všem bez dalších potřeb. OpenStreetMaps mapy jsou navíc upravovány v případě změn rychleji, než je tomu u Google Maps.

2.3.1 Osmdroid

Osmdroid je knihovna pro Android, která do určité míry nahrazuje standardní Android prvky pro zobrazení map. Knihovna obsahuje třídu, jež je přímou náhradou standardní třídy v androidu se stejným jménem. Funkce této třídy spočívá v načítání a zobrazování OpenStreetMaps mapy, což je po jejím načtení vykonáváno automaticky, pokud je k dispozici připojení k internetu. Takto načtené části mapy se ukládají do interního úložiště pro pozdější rychlé načtení (nebo pro případ provozu offline). Většina ostatních vlastností této třídy je stejných jako u originální.

Knihovna dále obsahuje další třídy, v této práci se však nepoužívají.

2.3.1.1 OsmdroidBonusPack

OsmdroidBonusPack je nadstavba původního Osmdroidu a přináší sadu nových tříd pro používání s původní knihovnou. V této práci je však použit pouze pro snadněji implementovatelné informační okno, které se dá zobrazit přímo do mapy. Další třídy z knihovny v mobilní aplikaci použity nejsou, v případě budoucího rozšíření by se zvážilo použít další prvky (např. plánovač tras).

2.3.2 OpenLayers

Knihovna OpenLayers slouží k zobrazování map na webových stránkách. Ke knihovně je dostupná dokumentace, která je k nalezení na [8]. Knihovna je psaná pro jazyk JavaScript, který slouží pro manipulaci webových stránek na straně klienta. Není přímo provázaná s OpenStreetMaps, ale umožňuje zobrazovat více druhů map, a to včetně OpenStreetMaps map.

Princip mapy z knihovny OpenLayers vychází z použití vrstev, ve kterých jsou umístěny zobrazitelné prvky. Samotná mapa je základní Layer, v našem případě OSM Layer.

V knihovně jsou kromě základního objektu pro zobrazení mapy i vrstvy s různými vlastnostmi (Vectors, Markers,Layers) nebo i objekty pro umístění do jednotlivých vrstev (Marker, Line).

2.4 Facebook

Pro co nejsnazší přidávání přátel do aplikace bylo navrhnuto použít nějakou sociální síť, na kterou by se uživatel v aplikaci připojil, a následně by se mu načetli všichni jeho přátele. Jako tato sociální síť byl vybrán Facebook kvůli jeho všeobecné známosti a rozšířenosti.

Pro používání Facebooku ve své aplikaci či na webových stránkách je třeba se nejprve zaregistrovat jako Facebook Developer a následně na stránkách Facebooku vytvořit Facebook aplikaci. Tím se vygeneruje číslo aplikace a tajné číslo, které se musí zadat do hlavní souboru Android aplikace nebo při každém přihlášení do Facebooku na webových stránkách, pro umožnění přístupu k funkcím Facebook API.

Od verze Facebook SDK 2.0 požadavek pro získání přátel vrací pouze přátele, kteří mají nainstalovanou tu samou aplikaci jako je ta, ze které požadavek přišel. To způsobilo, že oproti původnímu návrhu aplikace není potřeba filtrovat přátele na ty, u kterých by bylo více možností díky naší aplikaci, zároveň ale zabránilo zobrazování ostatních přátel podle Facebookem

získané poslední pozice. Podrobné dokumentace k programování pro Facebook jsou dostupné na [6].

2.4.1 Facebook SDKpro Android

Facebook SDK (Software Development Kit)pro Android obsahuje předpřipravené třídy a funkce pro inicializování a práci s Facebookem. Obsahuje objekty umožňující přesměrování na webové stránky pro přihlášení do Facebooku a také to samé pro odhlášení z něj. Dále obsahuje třídy pro posílání specifických i obecných požadavků na Facebook server. V Androidu se na rozdíl od PHP či JavaScriptu musí tyto požadavky posílat asynchronně, což způsobuje problémy se zpožděním při zobrazování, ale to podrobněji až v implementaci.

Během implementace mobilní aplikace proběhl přechod Facebook API z verze 3.2.3 na 4.0.0 a veškeré funkce, které předtím pracovaly na principu vytvoření *session* přešly na princip se získáním *AccessToken.*

2.4.2 Facebook SDK pro JavaScript

JavaScriptové SDK poskytuje podobné prostředky jako výše zmíněné SDK pro Android. Na rozdíl od PHP SDK je zaměřena na klientskou stranu webových stránek, umožňuje tedy víc prostředků pro manipulaci s Facebookem na straně uživatele. To znamená, že se lépe implementují dynamické prvky jako je například přihlašování a odhlašování z Facebooku. Na druhé straně se kvůli povaze JavaScriptu hůře používají prvky, které je nutné po získání zobrazit, jako například výpis listu přátel.

V této práci se tedy JavaScriptové SDK používá na webových stránkách pouze na přihlášení a odhlášení z Facebooku.

2.4.3 Facebook SDK pro PHP

Protože PHP kód se zpracovává na straně serveru hned na začátku zpracování stránky a je následně poslán klientovi pro zobrazení, není jej možné na aktuální stránce využívat dynamicky bez znovunačtení. Proto i když toto SDK obsahuje většinu funkcí dříve zmíněných SDK, žádná z nich nelze použit během prohlížení stránky bez použití asynchronního či i synchronního volání pomocí JavaScriptu. Naštěstí SDK obsahuje několik možností pro připojení Facebooku a jednou z nich je získání aktuální *session* z JavaScriptového přihlášení. Ostatní typy nedovolují přihlášení bez přesměrování na jinou stránku, jiným způsobem získaného accessTokenu či programování stránek na Facebook Canvasu. Protože je PHP lepší pro

zobrazování prvků na stránkách, je toto SDK primárně využito na získání dat o přátelích uživatele a následné zobrazení.

Pozn. Přestože v nejnovější API bylo v androidu SDK Session odstraněno, v PHP SDK je stále využíváno u přihlašování.

3 Cloud computing

Obchodní svět je již po dlouhou dobu charakterizován závodem pro nalezení cenově efektivních metod vyvíjení produktů bez nutnosti ve skutečnosti koupit nebo vlastnit všechny potřebné zdroje. I když samo jméno “cloud computing” je poměrně nový termín, tento koncept se používá již značnou dobu.[1].

Cloud computing je tedy pojem pro poskytování hostingových služeb přes internet a nabízí IT manažerům možnosti pro zajištění lepšího cenového poměru pro skladování dat, lepší a rychlejší přístup k softwaru a také optimalizaci ceny a úsilí.

3.1 Základní vlastnosti cloud computingu

Samoobslužný systém na požádání.Zákazník může jednostranně využívat výpočetních vlastností jako je například výpočetní výkon a úložná kapacita, kterou potřebuje automaticky, bez nutnosti lidské interakce s každým poskytovatelem daných služeb.[2]

Široký přístup k síti. Služby jsou dostupné přes síť a přístupné přes standardní mechanismy, které podporují využití heterogenních tenkých či tlustých klientů (např. mobilní telefony, tablety, notebooky a stolní počítače).[2]

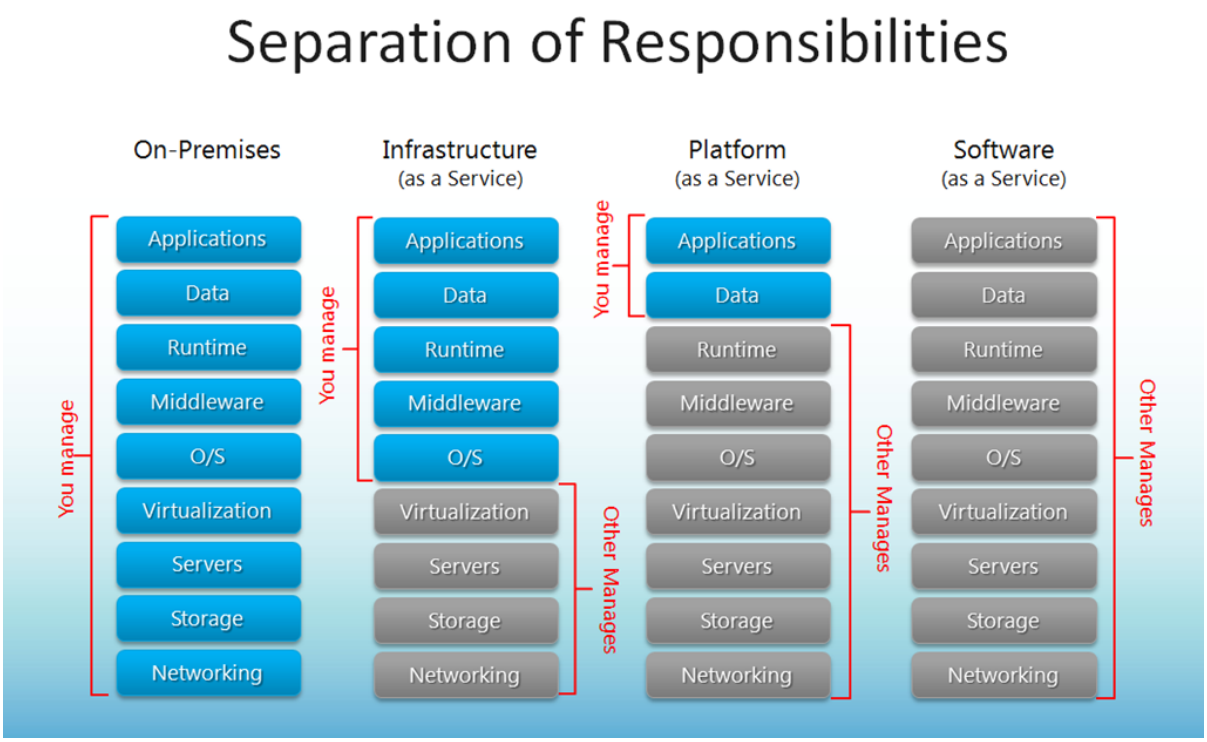
Sdílení zdrojů. Zdroje poskytovatele jsou sdíleny tak, aby mohly sloužit k obsluhování více zákazníků zároveň. Různé fyzické či virtuální zdroje jsou dynamicky přiřazovány a odebírány podle aktuálních zákazníkových požadavků. [2]

Velmi rychlá přizpůsobitelnost. Zdroje mohou být pružně přidělovány a odebírány, v některých případech automaticky, tak aby přizpůsobily aktuální poptávce. Pro zákazníka by zdroje měly často vypadat neomezené a mohou být kdykoli použité v jakémkoliv množství.[2]

Měřená služba. Cloudové systémy automaticky kontrolují a optimalizují použití zdrojů pomocí měření zdrojů na určité úrovni abstrakce vhodné pro daný typ služby. Použití zdrojů může být monitorováno, kontrolováno a ohlášeno, což přináší transparentnost jak pro poskytovatele, tak pro zákazníka.[2]

3.2 Modely pro cloudcomputing

O určitou část služby se stará poskytovatel a zbylou část zákazník. Podle toho jak velká část to je a co obnáší, se dají popsat tři základní modely pro poskytování služeb cloud computingu. Mezi tyto základní modely patří IaaS, SaaS a PaaS.



Obrázek 3 Rozdělení povinností jednotlivých metod [3]

Obrázek zobrazuje všechny tyto modely včetně modelu, kdy se o vše stará samotný zákazník. Jednotlivé položky znázorňují prvky, o které je třeba se postarat. Barva značí, zda se o daný prvek stará provozovatel služby (šedá), nebo samotný zákazník (modrá).

3.2.1 IaaS

IaaS (infrastruktura jako služba) poukazuje na zprostředkování výpočetního výkonu a infrastruktury jako služby. Také známá jako HaaS (Hardware jako služba), IaaS zahrnuje všechny fyzické výpočetní zdroje, které umožňují dodání aplikace jako služby zákazníkem. IaaS zprostředkovává výrazné ušetření peněz organizacím, neboť umožňuje přístup k dalšímu výpočetnímu výkonu na požádání, bez nutnosti pro výrazné investování do dalšího hardwaru, atd. Pro tento důvod existuje míň poskytovatelů IaaS modelu, v porovnání s PaaS (platforma jako služba) a SaaS (software jako služba) poskytovateli, kvůli většímu kapitálu a provozním nákladům, které jsou potřeba pro založení a provoz Iaas podniku.[1]

3.2.2 SaaS

SaaS (software jako služba) je softwarový model, který zahrnuje dodání softwarové aplikace vzdálenému klientovi za pomoci internetu. SaaS spoléhá na centralizovaný hosting softwarové aplikace v "*cloudu*", ke kterému se běžně přistupuje pomocí webového prohlížeče. Může být také nakonfigurován tak, aby umožnil buď veřejný přístup, nebo soukromý přístup, kde pouze určitým uživatelům je povolen přístup do konkrétní softwarové aplikace. SaaS, který je také znám jako software na požádání, nabízí mnoho výhod oproti tradičnímu softwarovému modelu.[1]

3.2.3 PaaS

PaaS (platforma jako služba) je metoda, kdy celá výpočetní platforma může být vzdáleně používána přes internet.PaaS nabízí cestu, jak v podstatě vyřešit všechny infrastrukturní požadavky na vytvoření řešení bez nutnosti zakoupení a instalace nové platformy. Toto umožňuje individuálním klientům sdílet stejné zdroje a vyhnout se nakupování, údržbě, aktualizování a řešení případných problému s hardwarem a softwarem, jako kdyby musely být spravovány samotným klientem. Tento model typicky umožňuje klientům platit pouze za jejich podíl spotřeby zdrojů. Je to zejména výhodné pro společnost, která potřebuje určitou aplikaci, ale používá ji pouze občas.[1]

PaaS typicky umožňuje uživatelům vytvářet, testovat a spustit aplikaci v jednom prostředí. I když se služby během vývojového životního cyklu aplikace mohou měnit, PaaS model umožňuje, aby byly tyto služby umístěny na stejné platformě jako ta, ze které se bude aplikace spouštět, což dovolí dynamické testování uživateli a aktualizace. Nástroje uživatelského prostředí jsou typicky webově orientované, což přináší jednoduchou dostupnost a také rychlé testování koncovými uživateli v různých stupních vývoje.[1]

Tato architektura také rovněž umožňuje nastavovat úroveň oprávnění uživatelů a přístup ke zdrojovému kódu členům vývojového týmu. Vývojové týmy mohou být spravovány skrze dodané prvky, jako jsou oprávnění, úkoly, přidání vlastnictví určitých oblastí projektu. Dále, PaaS služby mohou poskytovat dynamické statistiky o používání, znázorňující vývojáři, kdo co používá a kdy, což dovoluje používat účtování za použití a také které služby se využívají a které ne. [1]

OpenShift patří mezi architekturu využívající metodu PaaS. PaaS architektura byla pro tuto práci vybrána, kvůli tomu že poskytuje možnost vytvářet webové stránky bez potřeby jakékoli další práce. Zároveň Openshift poskytuje MySQL databázi.

4 Android

Android je operační systém pro mobilní zařízení vycházející z jádra Linuxu a je vyvíjen společností Google. Na rozdíl od svého konkurenta iOS, Android je opensource.

Při programování pro Android se používá jazyk Java, a jako hlavní vývojářské prostředí slouží program Eclipse, ke kterému se pro umožnění práce s Androidem musí doinstalovat volně dostupné Android SDK.

Mobilní zařízení s tímto operačním systémem zabírají mezi všemi svými konkurenty největší procento trhu. Zařízení s iOS jsou běžnější převážně v západních zemích a USA. Windows Phone má zatím celkově menší výskyt než zařízení s Android OS.

Jako cílová platforma byl vybrán právě kvůli rozšířenosti a také kvůli tomu, že v programování pro něj je příjemnější. Dále se kvůli běžnějšímu výskytu na trhu snáze hledají osoby, které by byly ochotny testovat aplikaci, což v případě, kdy je například nutné sdílet polohu mezi více lidmi potřeba.

Protože k pochopení Androidu je zapotřebí veliké množství informací, které tato práce nemůže všechny obsáhnout, byly vybrány jen základní informace, které jsou důležité pro pochopení informací obsažených v kapitole o implementaci.

4.1 Hlavní Android komponenty

Android *framework* definuje čtyři hlavní složky: *aktivita*, služba (*service*), *broadcast receiver* a *content provider*. Každá aplikace nepotřebuje použít všechny tyto složky, ale jejich správné používání dovoluje aplikaci plně splynout s platformou.

4.1.1 Aktivita a intent

Aktivita je nejdůležitější složka aplikace, neboť je provázaná s tím, co se zobrazuje na obrazovce. Uživatelé mohou manipulovat s aplikacemi pouze za pomocí *aktivit*. Aplikace může být tvořena jednou, ale i více *aktivitami*. Každá *aktivita* umožňuje uživateli vykonávat určitou úlohu. Abychom zachovali modularitu, předpokládá se, že každá *aktivita* bude dělat pouze jednu úlohu.[4]

Uživatelé zahajují novou aktivitu se záměrem splnit určitý úkol. Tyto záměry jsou v Android frameworku zachyceny jako *intenty*. *Intent* je abstraktní popis operace, která se má provést. Dovaluje tak pozdní vazbu během běhu mezi složkami, i když stále používá pasivní datovou strukturu. [4]

Android si udržuje mapování *intentů* do aktivit a vyvolává správnou aktivitu v závislosti na použitém *intentu*. Pro některé záměry může existovat více než jedna aktivita, která umí splnit daný úkol.

4.1.2 Služba

Služby (*service*) běží na pozadí aplikace. Nenabízejí žádné uživatelské prostředí a nemůže s nimi být přímo manipulováno uživatelem. Samotný Android neomezuje délku jejich trvání, což jim umožňuje pokračovat v provozu na pozadí tak dlouho, dokud má systém dostatek zdrojů pro provoz úkolů na popředí (viz 4.3). Aplikace může poskytovat aktivity, skrze které je možné ovládat služby.[4]

4.1.3 Broadcast Receiver

Aplikace pracuje nejen s uživateli, ale také je ovlivňována přímo platformou a jinými spuštěnými aplikacemi, které vytvářejí události (*event*). V Androidu mají tyto události formu *intentu*.

Pro obdržení určitých typů událostí aplikace se může použít *broadcast receiver*, a zaregistruje určitou sadu *intentů*. Když se v systému vygeneruje shodná událost, Android tuto událost pošle do *broadcast receiveru*. [4]

4.1.4 Content Provider

Content provider dovoluje Android aplikacím vyměňovat data s platformou a s ostatními aplikacemi. Na rozdíl od ostatních složek, práce *content provideru* nezávisí na *intentech*. Místo toho *content provider* používá standardní rozhraní ve formě *contentových* URI, a umožňuje přístup k datům, jakmile je jedna nebo více tabulek podobných k tabulkám v relační databázi.[4]

4.2 Základní grafické prvky

Pro umožnění práce s aplikací se musí aktuální aktivita nejdříve provázat s grafickými prvky, se kterými by mohl uživatel pracovat.

4.2.1 View

View objekty jsou základní jednotkou uživatelského prostředí v Android platformě. Je to datová struktura, v jejíchž datech ukládá parametry *layoutu* a *content* zobrazené obdélníkové oblasti na obrazovce. Navíc poskytuje metody potřebné k vykreslení a měření layoutu.[4]

4.2.2 Layout

Layout se používá k vyjádření hierarchie *view* objektů a jak by měl být každý *view* komponent umístěn na obrazovce. Neboť se velikost, rozlišení a orientace různých Android zařízení hodně liší, *layout* umožňuje vývojáři aplikace dynamicky pokládat *view* komponenty v závislosti na specifikacích zařízení.[4]

4.2.3 Widget

Widget je *view* objekt, který dovoluje uživateli pracovat s aplikací. Android poskytuje bohatou sadu *widgetů*, čímž umožňuje vývojářům jednoduše vytvářet obsáhlé uživatelské prostředí. Vývojáři navíc nejsou limitováni k používání pouze *widgety* nabízené Androidem. Mohou vytvářet zcela nové *widgety*, nebo mohou upravit již existující. [4]

4.3 Android Manifest

Android aplikace jsou popsány v systému pomocí manifest souboru zvanému **AndroidManifest.xml**, což je soubor ve formátu XML. U všech Android aplikací je očekáváno, že budou obsahovat tento soubor v jejich kořenovém adresáři. Tento manifest soubor představuje systému všechny nezbytné informace o aplikaci tak, aby mohla Android platforma správně spustit kód aplikace a udělit nezbytné pravomoci během instalace.

Android manifest soubor poskytuje systému tyto informace:

- Obsahuje jméno, jméno balíčku a číslo verze aplikace.
- Ukazuje minimální verzi API potřebnou k běhu aplikace.
- Popisuje složky aplikace (viz 4.1) a *intenty*, které jsou tyto složky schopné zpracovat.

- Deklaruje, které povolení jsou potřebné pro přístup k chráněným částem Android *runtimu* a také k práci s ostatními aplikacemi běžícími v systému.
- Říká, které povolení potřebují ostatní aplikace pro možnost práce s jejími složkami.
- Obsahuje seznam knihoven, se kterými musí být aplikace provázaná, aby mohla pracovat.[4]

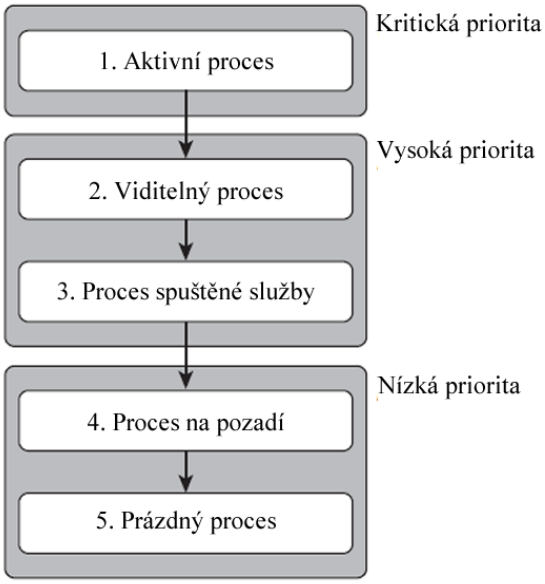
4.4 Životní cyklus aplikace

Životní cyklus Android aplikací je mnohem víc komplikovaný než u aplikací na jiných platformách. U těch je většinou životní cyklus ovládán uživatelem. Uživatelé si mohou vybrat zapnutí či ukončení aplikace, kdykoliv chtějí. V Androidu však platforma spravuje životní cykly aplikací sama, aby mohla co nejlépe využít nedostatkových zdrojů.

Na obrázku číslo 4.1 je vidět jak postupuje životní cyklus aplikace a jak velkou prioritu má ve kterém stavu, a tedy jak velká je šance, že ji systém ukončí v případě nedostatku zdrojů.

Obvyklý postup stavů v Android aplikaci:

Aktivní procesy. Aktivní (v popředí) procesy jsou ty, s jejichž složkami uživatel aktuálně pracuje. Toto jsou procesy, které se Android snaží udržet naživu pomocí získání zdrojů z jiných aplikací. Obecně existuje najednou jen velmi málo těchto procesů a jsou Androidem ukončený jen jako poslední možnost.[5]



Obrázek 4.1 Životní cyklus aplikací

Viditelné procesy. Viditelné, ale neaktivní procesy jsou ty, které obsahují právě viditelné aktivity. Viditelné aktivity jsou viditelné, ale nejsou v popředí a nereagují na události vyvolané uživatelem. Toto nastává, pokud je aktivita částečně zakrytá (například aktivitou, která nevyplňuje celou obrazovku nebo částečně průhlednou aktivitou). Viditelných procesů bývá převážně ještě míň než aktivních procesů a bývají ukončeny jen v případě, že není jiná možnost jak udržet aktivní procesy naživu.[5]

Procesy spuštěné služby. Procesy provozující spuštěné služby (*service*). Protože tyto služby neinteragují přímo s uživatelem, mají proto o něco menší prioritu než viditelné aktivity či služby na popředí. Aplikace se stále běžícími službami jsou považovány jako procesy na popředí a nebudou ukončeny, pokud není potřeba zdrojů pro nějaký aktivní či viditelný proces. Pokud systém ukončí běžící služby, po uvolnění zdrojů se ji pokusí sám od sebe znovu spustit.[5]

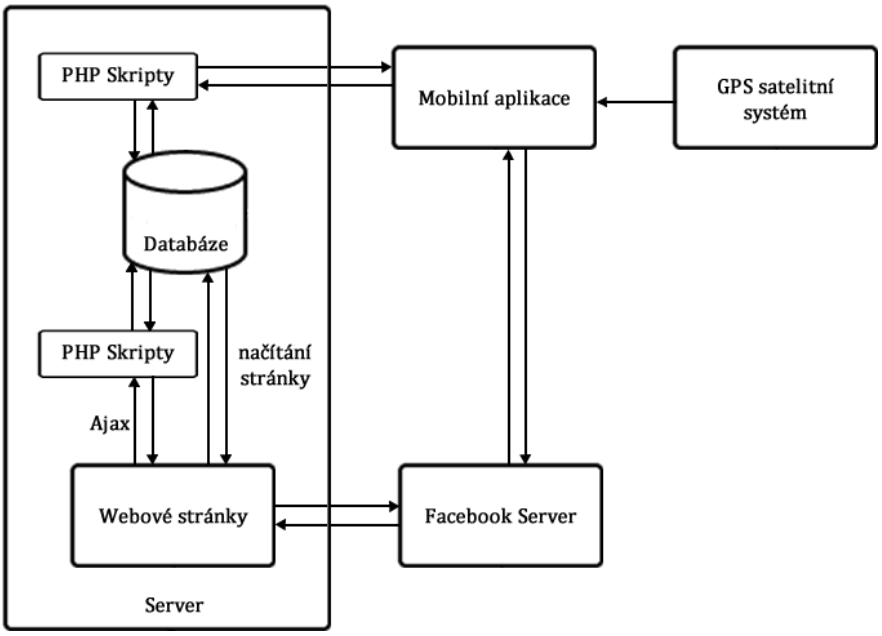
Procesy na pozadí. Procesy, které nejsou viditelné a nemají žádnou aktivní službu. Převážně existuje velké množství procesů na pozadí, které Android ukončuje podle vzoru *last-seen-first-kill*, aby získal zdroje pro procesy na popředí.[5]

Prázdné procesy. Pro zlepšení celkového systémového výkonu, Android často uchovává aplikaci v paměti i potom co dosáhla konce svého životního cyklu. Android si tento *cache* udržuje kvůli zlepšení startovací rychlosti při znovuspuštění této aplikace. Tyto procesy jsou běžně ukončovány, podle potřeby.[5]

Životní cyklus není ani tak potřebný pro pochopení implementace, ale spíše pro pochopení problémů se zdroji při ukončování a přesouvání aplikace na pozadí, které se musí při programování aplikace řešit.

5 Návrh práce

Pátá kapitola popisuje návrh aplikace, webových stránek a všech ostatních důležitých prvků souvisejících s touto prací. Je zde vysvětlena očekávána funkčnost jednotlivých prvků a místo, kde by se tyto prvky měly vyskytnout vzhledem ke zbytku aplikace, není však uveden postup implementace. Popis průběhu implementace následujících prvků bude podrobně probrán v následující kapitole.



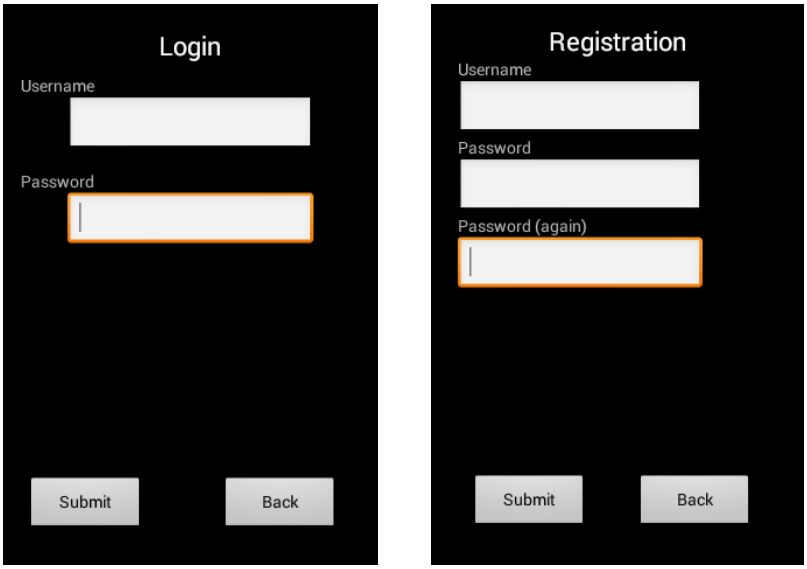
Obrázek 5.1 Struktura celého systému a komunikace v něm

5.1 Mobilní aplikace

Mobilní aplikace je v této práci považována za nejdůležitější část, neboť právě přes ni probíhá největší část manipulace s daty a samotné získávání dat je řešeno pouze přes tuto mobilní aplikaci. Bude v ní možné se zaregistrovat, čím se vytvoří účet v databázi, a následně přihlásit. Kromě mapy pro zobrazování pozic aplikace bude umožňovat pracovat s přáteli a jejich daty. Dále by měla obsahovat možnost měnit určitá globální nastavení týkající se získávání a sdílení dat.

5.1.1 Registrace a přihlašování

Pro možnost uchování dat a pro jejich pozdější využití je nutné, aby se data daly třídit podle jednotlivých uživatelů. K tomu je potřeba, aby každý uživatel šel rozlišit pomocí unikátního jména, a k tomu je nutná registrace. Okamžitě po spuštění aplikace by se tedy měla zobrazit nabídka s možností registrace nebo přihlášení. Pro registraci bude pouze potřeba vybrat si svoje uživatelské jméno a přístupové heslo. Protože hlavní důvod registrace je pouze vytvoření unikátního identifikátoru, nebude potřeba se hned po úspěšné registraci přihlašovat a uživatel bude ihned přihlášen. V případě odhlášení nebo použití svého účtu na jiném mobilním zařízení bude potřeba vybrat v nabídce možnost přihlášení a následně zadat své uživatelské jméno a heslo.



Obrázek 5.2 Přihlašování a registrace

5.1.2 Mapa

Nejdůležitější prvek aplikace, na kterém se budou zobrazovat veškeré poziční data všech osob, tj. jak uživatele, tak jeho přátel. Mezi tyto data bude patřit aktuální nebo alespoň poslední lokace dané osoby. Dále trasa, kterou osoba prošla za určitý čas, a volitelně také cíl cesty nebo přesněji jakékoli místo, které osoba označí na mapě.

Zároveň by obrazovka s mapou měla fungovat jako rozcestník pro zbylé prvky aplikace, jako jsou například přátelé či nastavení. Kromě tohoto by měla obsahovat i prvky, které by uměly manipulovat s daty uživatele, a to alespoň centrování pozice uživatele a jeho cíle. Schování tras v případě, že uživateli překáží, by se zde mělo také objevit. V neposlední řadě možnost odhlášení z aplikace pro případ nutnosti přihlásit se na jiného uživatele.

Obrázek 5.3 Hlavní obrazovka s mapou

Aktuální pozice

Aplikace musí umět zobrazovat nejaktuálnější pozice všech osob, a zobrazuje je tak, aby byly určitým způsobem rozlišitelné od ostatních. Buď rozdílným obrázkem v případě uživatele vůči přátelům, nebo slovním popisem po stisknutí značky. Tím by se mělo zobrazit malé informační okno přímo na mapě, ve kterém bude napsáno uživatelské jméno osoby nebo v případě načtení tohoto přítele přes Facebook, jeho skutečné jméno a navíc i jeho profilová fotka v ikonové velikosti. Dále by zde měl být zobrazen i čas, kdy byla tato pozice zaznamenána.

Trasy

Ušlé trasy uživatelů jsou provázáním všech zaznamenaných pozic osoby za určitou dobu. Měly by se zobrazovat jako spojité čáry o měnitelné barvě. V posledním bodě trasy bude vždy zobrazena značka aktuální pozice.

Cíle

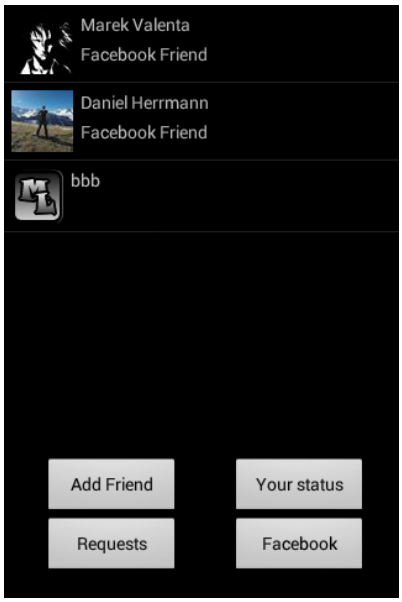
Cíle jsou značky, které se nezískávají pomocí GPS lokalizačního systému, ale uživatel si je bude volit sám dlouhým stiskem místa na mapě. Stejně jako pozice a trasy se cíle budou sdílet mezi přáteli a podobně jako značky se budou rozlišovat obrázkem v případě uživatel-přátelé a stisknutí zobrazí informační okno s podrobnějšími informacemi.

5.1.3 Přátele

Hlavní komponentou pro možnost sdílení polohy jsou samotné osoby, se kterými je možné polohu sdílet. V případě této práce jim budeme pro jednoduchost říkat přátelé. Pro možnost přidat někoho jako svého přítele, je potřeba, aby také měl zaregistrovaný účet v této aplikaci.

Pro manipulaci s přáteli bude potřeba vybrat na obrazovce s mapou nabídku *Friends,* což přepne aplikaci do hlavní obrazovky pro práci s přáteli. Zde budou zobrazeni už aktuálně přidaní přátele jako vertikální seznam, kde každá položka bude obsahovat obrázek, přihlašovací jméno či skutečné jméno pokud se jedná o přítele z Facebooku, a také rozlišení zda je přítel přidán normálně přes žádost nebo pomocí Facebooku. Každá tato položka půjde navíc stisknout pro zobrazení dalších možností týkajících se přímo daného přítele.

Tato obrazovka by měla mít v dolní části možnosti pro zadání žádosti k přidání dalšího přítele, přístup do Facebooku a zobrazení žádostí, které byly poslány uživateli ostatními osobami.



Obrázek 5.4 Obrazovka s přáteli

Přidávání přátel

Jak již bylo zmíněno výše, přátelé půjdou přidávat pomocí žádostí nebo pomocí Facebooku. Zatímco přátelé z Facebooku budou pouze ti, kteří se zaregistrovali v této aplikaci a zároveň se pomocí této aplikace přihlásili do Facebooku, výběr přátel přes žádost nebude mít žádné omezení kromě zaregistrování v aplikaci.

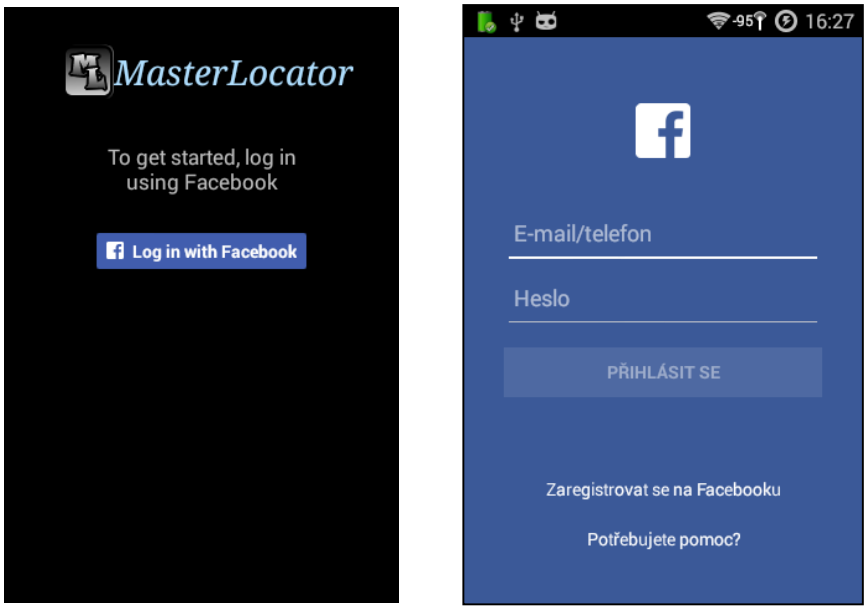
Pro poslání žádosti bude stačit znát pouze uživatelské jméno osoby, kterou si chce uživatel přidat jako přítele. Po vytvoření žádosti se tato žádost objeví v seznamu žádostí u cílené osoby, která ji může buď přijmout, nebo zamítnout. V případě přijmutí se obě osoby stávají navzájem přáteli toho druhého.

Přátelé z Facebooku budou automaticky načteni a přidání po přihlášení do Facebooku.

Přátelé z Facebooku

Kvůli celkem běžné situaci, kdy uživatel má více než jen pár přátel na svém Facebook účtu, a problému, který by nastal v případě, že by většina těchto osob měla nainstalovanou tuto aplikaci, je třeba vytvořit samostatný seznam, do kterého budou přesunuti všichni přátelé načtení z Facebooku. Z tohoto seznamu se budou vybírat pouze osoby, které chce v danou chvíli uživatel sledovat. Tyto osoby se následně přesunou do standardního seznamu přátel.

Přístup k tomuto seznamu bude přes tlačítko Facebook ve spodní části standardního seznamu přátel. Pro jeho zobrazení bude samozřejmě nutné připojení k Facebooku a internetu.



Obrázek 5.5 Přihlašování do Facebooku

Možnosti u přátel

Po stisknutí položky přítele v seznamu přátel se spustí dialogové okno s možnostmi, které bude možné učinit s daným přítelem.Mělo by jít například o přesun v mapě na aktuální pozici přítele, pokud nějaká tato pozice bude existovat. To samé platí i pro cíl cesty a v obou případech se aplikace sama přepne na obrazovku s mapou. Mezi další možnosti bude patřit změna barvy trasy pro lepší přehlednost v případě více tras na stejném místě. Bude také možné vypnout zobrazování přítele na mapě a také zakázat posílání vaší pozice tomuto příteli.

5.1.4 Nastavení

Pro lepší zacházení s aplikací by mělo být uživateli umožněno měnit určité prvky ohledně získávání a sdílení dat. Možnost změny vzdáleností mezi pozicemi zamezí shlukování bodů trasy v případě setrvání uživatele v relativně malé oblasti po delší dobu. Určení doby mezi zaznamenáním polohy se dají vyřešit podobné problémy. Navíc v případě zvolení delších časů, bude během větší část této doby vypnutý GPS modul, čímž se docílí výrazného šetření baterie. Dále bude možné zapnout mód, ve kterém se budou přijímat pouze pozice, které mají maximální odchylku jen pár metrů. Ohledně sdílení lokace si bude uživatel moct vybrat, jak moc staré data chce přijímat od svých přátel.

Nastavení budou přístupná z hlavní obrazovky s mapou stisknutím tlačítka *Settings*.

5.2 Serverová část

Aby mohli jednotliví koncoví uživatelé ukládat data a následně je mezi sebou sdílet, je zapotřebí vytvořit určité centrální body, kde budou tyto data uchovány a odkud budou rozesílány aplikacím. Tomuto místu se říká server a obsahuje databázi, kde se budou ukládat data uživatelů, skupinu skriptů, které se volají mobilní aplikací, a webové stránky, umožňující některé funkce mobilní aplikace přímo na internetu bez nutnosti mobilního zařízení.

5.2.1 Komunikace aplikace se serverem

Protože jsou všechna data uložena v serverové databázi, je nutné, aby aplikace mohla jednoduše komunikovat se serverem. Pro tuto komunikaci je nutné vytvořit množství skriptů, kde každý bude odpovídat a zpracovávat určitý požadavek aplikace. Výběr, který skript bude zavolán, by měl být uskutečněn na straně aplikace.

Skripty budou muset minimálně umět zpracovávat požadavky týkající se registrace nového uživatele, přihlášení již registrovaného uživatele, načtení přátel (pouze normální

přátelé, přátele z Facebooku se zjišťují přes něj), podat žádost příteli, potvrzení žádosti, načtení žádostí, poslání pozice, získání pozic jak svých tak od přátel, získání a zadání cíle cesty, nastavení schování před jiným uživatelem a přidání identifikačního čísla Facebooku po přihlášení se k němu.

5.2.2 Databáze

Databáze je místo, kde jsou uloženy veškeré data týkající se jednotlivých uživatelů. V našem případě se bude jednat o MySQL databázi, kde budou data rozdělena do několika tabulek a navzájem mezi sebou propojena za pomoci relačních odkazů. Databáze musí umět uchovávat informace o uživatelích, historii jejich pozic včetně dob pořízení, přátele uživatelů, cíle jednotlivých uživatelů, a podané žádosti o nové přátele.

Nastavení týkající se individuálního uživatelského prostředí se do databáze nebude ukládat, čímž bude v případě změny mobilního zařízení či odinstalování aplikace ztraceno.

5.2.3 Webové stránky

Na serverové straně bude také umožněno zobrazit databázové data podobně jako v mobilní aplikaci a to pomocí webových stránek. Na webových stránkách se uživatel bude stejně jako v mobilní aplikaci muset přihlásit pro získání přístupu k další činnosti, na rozdíl se však zde nebude moct zaregistrovat. Webové stránky nebudou umožňovat získávání aktuálních pozičních dat přes GPS a budou sloužit pouze k zobrazování těchto dat.

Přihlašování bude umístěno na samostatné stránce, kde po přihlášení bude uživatel automaticky přemístěn na hlavní stránku. Ta bude obsahovat podobné prvky jako mobilní aplikace, které však budou zobrazeny všechny najednou, díky většímu prostoru.

Hlavní stránka

Hlavní stránka bude rozdělena na tři části. První část bude část s mapou, která bude zabírat největší část obrazovky. Na mapě se budou stejně jako v mobilní aplikaci zobrazovat pozice uživatele a jeho přátel, ušlé trasy a jednotlivé cíle cest. Po kliknutí na pozici nebo na cíl se vytvoří, podobně jako v mobilní aplikaci, informační okno s více detaily ohledně majitele dané značky a času kdy byla pořízena.

Druhá část bude panel se seznamem přátel a možností připojení do Facebooku. Protože webové stránky nabízejí více místa a lepší manipulaci s prvky, budou přátele z Facebooku

zařazeni do seznamu přátel, ihned po přihlášení do něj. U jednotlivých přátel bude možné vypínat a zapínat jejich sledování, centrovat na jejich poslední dostupnou pozici, či centrovat na jejich zvolený cíl.

Třetí částí bude horní panel, který bude obsahovat prvky pro manipulaci s uživatelovou pozicí a cílem, možnost výběru z jak velké doby zpátky chceme získávat data a také možnost odhlášení z webových stránek.

6 Implementace

Tato kapitola je zaměřená na popis implementace prvků z předcházející kapitoly. Popisuje však i implementační prvky, které nejsou zobrazitelné uživatelem, jsou však důležité pro zajištění správného fungování ostatních prvků. Obsah je rozdělen do podkapitol pro popis mobilní aplikace včetně skriptů uložených na serveru, se kterými aplikace pracuje, webové aplikace a databáze.

6.1 Mobilní aplikace

Mobilní aplikace se skládá z několika důležitých skupin souborů. Mezi ty, které jsou zajímavé z implementačního hlediska a nejsou pouze automaticky generované, patří Java *class* soubory, které obsahují všechny Android *aktivity*, *dialogy* a objekty, což jsou soubory s kódem zajišťující hlavní funkčnost aplikace. Do další skupiny patří *layouty*, umožňující grafické zobrazení veškerých prvků v aplikaci. Android *Manifest* je také nepostradatelným prvkem aplikace.

Skripty na serverové části aplikace, i když jsou napsány v jazyku PHP na zcela jiné platformě, jsou potřebné pro přenos a uchování dat.

Ke správnému pochopení určitých termínů je nutné mít určitý přehled v programování pro platformu Android, nebo alespoň mít přečtenou čtvrtou kapitolu, kde je popis základních složek této platformy. Tyto prvky jsou popsány také v [7], ze které byly také čerpany informace během implementace.

6.1.1 Android Manifest

Hlavní soubor celé mobilní aplikace, neboť obsahuje všechny složky, které systém potřebuje ke spuštění aplikace. Patří k nim všechny povolení, které aplikace potřebuje k přístupu do dalších částí systému. Nutné jsou povolení k přístupu k lokaci, což umožňuje využívat modul GPS k získání aktuální pozice uživatele. Je požadováno i povolení pro rozšířené příkazy pro lokaci, čímž je získána pravomoc vymazat určitá data ohledně pozic satelitů. Dalšími povoleními jsou přístup k WiFi, internetu a síťovým službám, které jsou potřebné pro kontaktování serveru a posílání dat mezi databází a aplikací. Posledním povolením je přístup k externímu úložišti, což umožňuje uchovávat již stažené části mapy do zařízení pro pozdější použití.

V Android Manifestu je také řečeno jaká je nejmenší verze Android API na kterém může aplikace běžet. V tomto případě je to verze 11, což odpovídá Android verzi 3.0 a výše.

Jsou zde také uloženy inicializační data pro používání Facebook funkcí a prvků. Dále jsou zde popsány také všechny *aktivity*, včetně té, která se spouští jako první po zapnutí aplikace. Tento popis ukazuje systému, jak má s některými *aktivitami* v určitých oblastech zacházet (natočení obrazovky, pamatování si po přechodu na jinou *aktivitu* atd.)

6.1.2 Engine.java

K některým prvkům je potřebné mít přístup na více místech aplikace. Protože posílání dat mezi *aktivitami* není zcela triviální záležitost a často vede k větším problémům než je z ní užitku, je potřeba mít třídu, ke které je přístup z celé aplikace nezávisle na *aktivitě* či procesu, a všechny tyto důležité prvky do ní uložit.

Mezi tyto prvky patří například seznam přátel, seznam facebookových přátel, *context* aplikace či *context aktivity* s mapou.

Dále jsou v *Enginu* uloženy všechny konstanty a proměnné, které se mění a používají ve více *aktivitách* (např. id uživatele, proměnné z nastavení). Zároveň jsou zde i funkce pro použití skrz celou aplikaci jako třeba funkce pro rozpoznání zda je aktuálně přístup k internetu.

6.1.3 Datové třídy

Pro lepší uchovávání a manipulaci s daty, které jsou v aplikaci k dispozici, bylo třeba vytvořit třídy, jejíž objekty by tyto vlastnosti nabízely. Jednalo se hlavně o zaobalení všech dat konkrétní osoby, polohy či seznamu přátel.

Fix

Objekty třídy *Fix* slouží ke shromáždění všech dat týkajících se jedné konkrétní zaznamenané pozice a poskytují metody pro snadnou manipulaci s těmito daty. Objekt obsahuje souřadnice zaznamenané pozice, čas, kdy byla pozice obdržena, a informaci, zda již byla pozice zaslána na server.

Person

Třída *Person* se používá jako abstraktní forma reálné osoby, buď přítele či samotného uživatele. Ta obsahuje všechny dostupné informace, které k dané osobě máme. Objekty této

třídy umožňují uchovávat například uživatelské jméno, identifikační číslo jak z databáze tak z Facebooku, ikonu, odkaz na ikonu, nastavení ohledně zobrazování, seznam pozic (*fixů*), vrstvu mapy s cestou, poziční značku aktuální lokace a cíle, a další. Jako metody poskytují objekty kromě standardního nastavení a vrácení vlastností objektu (*get* a *set*) i metody pracující přímo s prvky na mapě týkající se této osoby (např. změna barvy trasy). Obsažené jsou i metody pro načtení dat z internetu (facebookový obrázek) a serveru (pozice osoby a cíl cesty). Po načtení pozic se ve stejné metodě vytvoří vrstva mapy s trasou.

Protože nová osoba může v aplikaci vzniknout z více různých zdrojů (přidáním přítele z Facebooku, přes žádost či při spuštění aplikace ze starých dat) existuje několik vlastností, které určují, jak byla tato osoba vytvořena a jak se k ní má v určitých případech přistupovat. Například přátelé vytvoření přes Facebook nemají uživatelské číslo, které je uložené v databázi a musejí tedy volat jiný skript pro získání jejich pozic ze serveru.

FriendList a PersonData

FriendList na rozdíl od dříve zmíněných tříd neslouží k seskupení dat určitého prvku, ale je to seznam objektů třídy *Person*, symbolizujících přátele uživatele. *FriendList* nemá žádné vlastnosti týkající se celkového seznamu, nabízí však metody pro práci se seznamem. Mezi tyto metody patří vyhledávání v seznamu podle identifikačního čísla uživatele nebo podle facebookového *id*. Umožňuje také vyvolat načtení pozic pro každou osobu v seznamu.

Kvůli možnosti uložit nastavení, které si uživatel pro jednotlivé osoby v seznamu zvolil, je nutné při vypínání aplikace seznam uchovat a načíst při opětovném spuštění. Jako seznam objektů je nutné nejprve jednotlivé prvky serializovat, což v případě objektů třídy *Person* není možné kvůli cyklickým odkazům. Proto bylo nutné vytvořit pomocnou třídu *PersonData*, která obsahuje pouze data důležitá pro zapamatování, a nemůžou v ní nastat žádné cyklické odkazy. Před vypnutím aplikace je tedy celý seznam převeden na seznam tvořený objekty *PersonData* a ten se následně uloží. Po spuštění aplikace se tento seznam opět převede na normální *FriendList*.

6.1.4 Komunikace se serverem

Android aplikace jsou omezeny tím, že v hlavním vláknu, na kterém běží aplikace, nemůžou přistupovat k internetu. Pro každé volaní skriptu na serveru musí být tedy vytvořen nový asynchronní proces, který musí celou výměnu mezi aplikací a serverem vyřídit sám. Tím vzniká hned několik problémů, které je nutné řešit. Prvním z nich je předávání dat z hlavního vlákna novému při jeho vytváření a naopak vrátit získaná data hlavnímu vláknu při ukončení

nového. Dalším problémem je předem neznáma doba, za jakou se proces vykoná, což způsobuje, že implicitně se nedá zjistit, kdy se proces ukončil a také to může způsobovat zpoždění v zobrazování získaných dat oproti zbylým prvkům.

V této aplikaci je komunikace se serverem řešena přes jedinou třídu **ASYT_NetUpdate.java**, která rozšiřuje třídu *AsyncTask*. Tato třída umožňuje vyvolat nový asynchronní proces, přijímá a vrací však pouze parametry jednoho typu. Příjem parametrů je proto řešen přes datový typ *String*, kde první parametr značí, jaký druh požadavku má být proveden, a další parametry jsou data, které si daná část kódu převede ze stringu do požadovaných datových typů. Rozpoznání, kdy proces ukončí svou práci, je vyřešeno pomocí vytvoření rozhraní nového posluchače událostí, který se spustí při dokončení procesu. Toto rozhraní musí být implementováno při každém použití této třídy. Zároveň je tímto vyřešen i problém i s vracením více hodnot, které se nyní dají poslat přes toto rozhraní.

6.1.5 Spuštění aplikace

Hned po spuštění aplikace se jako první zavolá *aktivita*, která slouží k inicializaci určitých prvků a proměnných. Nejprve je zprovozněn Facebook a následně se načítají data z minulého spuštění aplikace a uloží se do *Enginu*. Jedná se o všechny uživatelské nastavení, seznam přátel včetně nastavení u jednotlivých osob a také uživatelské *id*, což slouží k identifikaci, zda je v aplikaci někdo přihlášen. Všechna tato data jsou uložena v *SharedPreferences*, což je objekt, který umožňuje uchovávat data i po skončení aplikace. V případě nalezeného uživatelského id z minulého běhu aplikace, je zahájena přímo *aktivita* s mapou, jinak je spuštěno úvodní menu pro přihlášení či registraci.

6.1.6 Registrace a přihlašování

Úvodní menu se zobrazí v případě prvního puštění aplikace nebo odhlášení uživatele. Obsahuje rozcestník ve tvaru dvou tlačítek, která spouští *aktivity* pro registraci nového uživatele, či jeho přihlášení.

Registrace potřebuje uživatelské jméno a heslo, které jsou poslány na server pro zpracování. Uživatelské jméno se posílá nezměněno, zatímco heslo se hashuje za pomocí algoritmu *SHA-512*, což je nejsilnější hashovací algoritmus, který Android platforma nativně umožňuje. Po přijmutí serverem se heslo znovu zahashuje pomocí bezpečnějšího algoritmu *WHIRLPOOL* s přidaným 320 bitovým saltem (kryptografická sůl). Pokud uživatelské jméno ještě neexistuje, je přidán záznam do databáze o novém uživateli a na mobilní zařízení se odešle *id* uživatele. Následně se spustí hlavní *aktivita* s mapou.

Princip přihlašování funguje podobně jako registrace, jen namísto ukládání do databáze se kontroluje, zda již daný uživatel existuje. Heslo je stále hashováno jak na straně klienta, tak na straně serveru. V případě úspěchu je opět vráceno id uživatele a spustí se hlavní *aktivita*.

6.1.7 Aktivita s mapou

Hned po spuštění aktivity se provede pokus o zapnutí GPS modulu, začnou se poslouchat příchozí zprávy z GPS modulu a následně se teprve pošle požadavek o příjem lokalizačních dat. Zároveň probíhá inicializace mapy a ostatních prvků na obrazovce (tlačítka).

Následně se aktivita pokusí získat poslední zaznamenanou polohu GPS modulem a nastaví ji jako poslední pozici uživatele. V případě, že si GPS modul nepamatuje žádnou pozici, je nastavena výchozí pozice. Toto se děje nezávisle na serverové části a není k němu potřeba přístup k internetu. V případě, že je k dispozici přístup k internetu, se aktivita v asynchronním procesu pokusí získat historii pozic ze serverové databáze. Pokud tato data existují, přepíši po jejich načtení a zpracování pozici získanou z GPS modulu.

Zároveň s načítáním dat uživatele se aktivita pokouší získat data od přátel, kteří byli získáni během spouštění aplikace. Načítání dat je prováděno v asynchronním procesu, který posílá požadavek na server a opakuje se každých pět vteřin. Původně existovala v nastavení možnost změnit tento interval, bylo však experimentálně zjištěno, že tato funkce způsobuje pouze zanedbatelné zatížení systému a není ji třeba omezovat. Při kratším časovém intervalu však již nastávaly nekonzistence při zpracování dat, neboť některé později spuštěné asynchronní procesy dokončily zpracování dříve než jejich předchůdci. Pětivteřinový interval obsahuje dostatečnou časovou rezervu, aby toto za normálního chodu aplikace nenastalo.

Zároveň se v tomto intervalu spouští i funkce pro aktualizaci mapy, která znovuzíská všechny aktuální pozice, cíle a trasy všech přátel a uživatele a zobrazí je na mapě. Vzhledem k asynchronním procesům, můžou být některé data přátel lehce zpožděná, toto maximálně několikavteřinové zpoždění je však zanedbatelné.

Tlačítka ve spodní části obrazovky mají každé svou funkci, která se spustí v případě jejich stisknutí. V případě tlačítek *Center* a *Center target* se zjistí aktuální poloha nebo poloha cíle a mapa se přesune na tento bod. Při stisknutí *Hidepath* tlačítka se změní přepínač, který v aktualizační funkci řídí vykreslování tras. *Logout* tlačítko zruší v *Enginu* nastavení pro

aktuálního uživatele, vypne aktuální aktivitu a spustí aktivitu pro s rozcestníkem pro přihlášení a registraci. Tlačítka *Friends* a *Settings* spouští aktivitu se seznamem přátel resp. nastavení.

Následující tři body popisují implementaci prvků zobrazených na mapě. Čtvrtý bod ukazuje postup a příjmu a zpracování lokace.

Aktuální pozice

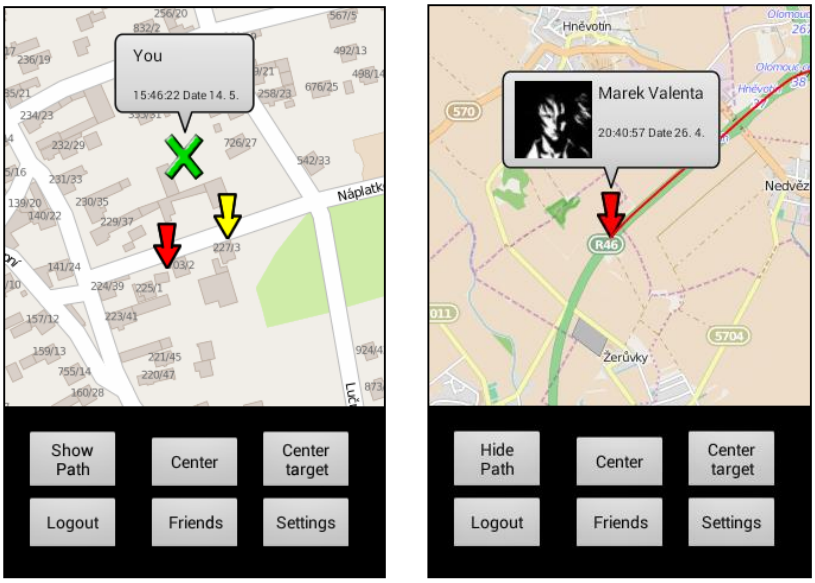
Aktuální pozice je na mapě zobrazena pomocí *marker* objektu (značka), který má každá osoba uložena ve svém objektu. Značka je tvořena obrázkem, pozicí a informačním oknem, které se zobrazí po stisknutí značky. Samotné informační okno je tvořeno časovým údajem zaznamenané pozice, jménem osoby a v případě facebookových přátel i jejich obrázkem. Pozice značky je získána jako poslední známá pozice této osoby. Obrázek značky je šipka, která má barvu v závislosti, zda je osoba uživatel nebo přítel a v případě uživatele zda je to nově získaná lokace nebo ne.

Trasy

Trasy osob jsou tvořeny jako vrstvy mapy, do kterých jsou ve správném pořadí přidávané pozice osoby, a tato vrstva pak nabírá tvar čáry. Vrstvy jsou vytvářeny, jen pokud existují alespoň dvě pozice v seznamu pozic dané osoby. V případě získání nových bodů není třeba vytvářet vrstvu znovu, stačí do ní tyto nové body přidat. Body však není snadné odebírat, proto při změně intervalu, ze kterého chceme pozice získat je vrstva vytvořena znova. Barva trasy patří mezi nastavitelné vlastnosti vrstvy a je pro uživatele a obyčejné přátele nastavená modrá, pro přátele z facebooku červená. Tuto barvu je přes nastavení přátel možno měnit dynamicky. Mezi další vlastnosti vrstvy patří například šířka a zaoblení čáry.

Cíle

Cíle jsou ve většině ohledů stejné jako aktuální pozice, nejsou však součástí trasy a proto potřebují vlastní tabulku v databázi pro jejich ukládání. Mají také odlišné funkce pro jejich získání a uložení. Cíl může uživatel vytvořit pomocí dlouhého stisku mapy, což je umožněno pomocí rozhraní *MapEventsReceiver*, které je implementované aktivitou s mapou. Toto rozhraní umožňuje reagovat na stihnutí mapy a vrátit pozici. Obrázek cíle má tvar křížku s rozdílnou barvou pro uživatele a přítele.



Obrázek 6.1 Ukázka cíle, aktuální pozice, trasy a informačního okna

Příjem pozice

V případě, že GPS modul příjme novou pozici, spustí se nový asynchronní proces pro zpracování této pozice. Pro získávání pozice uživatele se nepoužívá každá pozice samostatně, ale využívá se aproximace z několika pozic. Nejmenší počet takto využitých pozic je pět, během rychlejšího příjmu pozic či většího časového intervalu může být využito více hodnot. V případě, že již uplynul časový interval pro zaznamenání nové pozice do mapy a dostaneme pozici, která má vysokou přesnost, je využita tato pozice, neboť je s větší pravděpodobností lepší než aproximované hodnoty. K této situaci dochází ve většině případů v otevřeném terénu. V případě, že je aplikace v režimu velké přesnosti, využívá se pouze této možnosti (není možné získat pozice uvnitř budov).

V případě úsporného režimu bude po získání lokace příjem nových pozic zastaven a to až do půl minuty před vypršením intervalu pro novou pozici.

Po získání lokace se pozice zapíše do objektu symbolizujícího uživatele (*Person*) a spustí se funkce pro aktualizaci mapy.

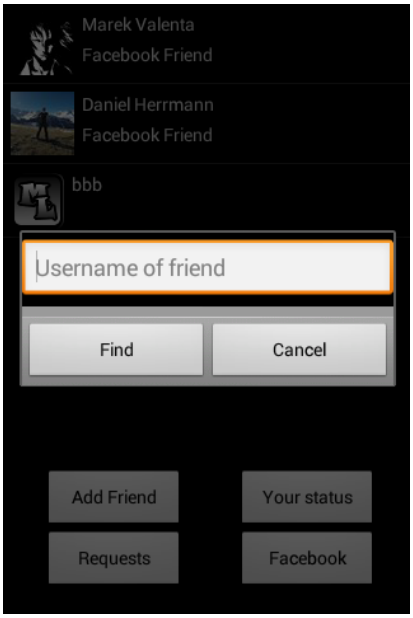
6.1.8 Aktivita se seznamem přátel

V této aktivitě je zobrazen seznam přátel pomocí prvku *ListView* a vyplňuje většinu obrazovky. Ve spodní části jsou tři tlačítka, která slouží k přidávání nových přátel pomocí žádostí a přepnutí se do aktivity s Facebookem.

Po spuštění této aktivity se provede pokus o získání přátel z databáze a po jejich obdržení proběhne přidání nevyskytujících se přátel do *FriendListu*. Mezitím je inicializovaný seznam a pomocí adaptéru se do něj vložili přátele z minulého spuštění aplikace (aktuální FriendList). Při zmáčknutí položky přítele, je zahájen dialog s dalšími možnostmi. Zároveň jsou v této aktivitě implementované funkce, které zpracovávají zpětné volání při výběru některé této možnosti (nastavení na střed mapy, zakrytí apod.). Dále proběhne inicializace tlačítek, kde je ke každému přiřazena vlastní funkce, která pomocí *intentu* spustí další aktivity či dialogy. Při ukončení této aktivity probíhá uschování seznamu přátel do *SharedPreferences*.

Žádosti

Žádosti slouží k přidávání přátel pomocí jejich uživatelského jména. Po stisknutí tlačítka *Add Friend* předchozí aktivity, je spuštěno nové dialogové okno s polem pro zadání jména (Obrázek 6.2). Po vyplnění jména a potvrzení se zašle požadavek serveru o uložení žádosti. Skript na straně serveru vyhodnotí, zda existuje dané jméno a jestli nenastávají jiné problémy a pošle odpověď zpátky aplikaci. V případě kladného vyřízení je žádost uložena do databáze, jinak je poslán popis problému.



Obrázek 6.2 Vytváření žádosti

Seznam žádostí uživateli se dá zobrazit stisknutím tlačítka *Requests*, což spustí novou aktivitu, ve které je zobrazen seznam žádostí. Při stisknutí žádosti se vytvoří dialog s možnostmi pro přijmutí, která odstraní žádost, pošle požadavek na server o přidání záznamu o přátelství do databáze a přidá tuto osobu do *FriendListu*, a pro odmítnutí, což pouze odstraní žádost z databáze posláním požadavku.

Přátelé z Facebooku

Facebook seznam přátel je zobrazen v samostatné aktivitě, neboť při větším množství přátel by se ztrácel přehled mezi ostatními přáteli. Po spuštění aktivity se pošle požadavek na Facebook server, který má získat všechny přátele uživatele, kteří mají tuto aplikaci taky nainstalovanou. Jakmile server odpoví, vytvoří se z těchto získaných dat seznam osob, pro každou osobu se spustí funkce pro získání profilového obrázku a za pomocí adaptéru se vloží do seznamu na obrazovce.

Tento seznam je uložen a při následujících přístupech do této aktivity za aktuálního běhu aplikace se pro vytvoření seznamu na obrazovce využívá uložený seznam osob. Nepředpokládá se přidání nových přátel v sociální síti v průběhu běhu aplikace, a proto není třeba volat server Facebooku znova. Tímto se vyřeší problém se zpožděním při vytváření seznamu v důsledku čekání na přijmutí dat.

Seznam umožňuje vybírat osoby, které se objeví v hlavním seznamu přátel, jednoduše za pomocí stisknutí položky daného přítele. Následně se zjistí, zda již osoba v seznamu přátel není, díky přijmutí žádosti. Pokud není, je přidán nový přítel, jinak se nová data o příteli přesunou k již existujícímu příteli. Rozpoznání zda je uživatel vybrán se řeší *CheckBoxem* na pravé straně položky.

Vzhledem k složitému vztahu mezi normálními a facebookovými přáteli nastává v jejich interakci mnoho kritických situací, které můžou v aplikaci nastat. Mnohé z nich byly při implementaci aplikace odchyceny, určitě však i kvůli propojení Facebook SDK s jinými aplikacemi v mobilním zařízení nebyly otestovány všechny případy.

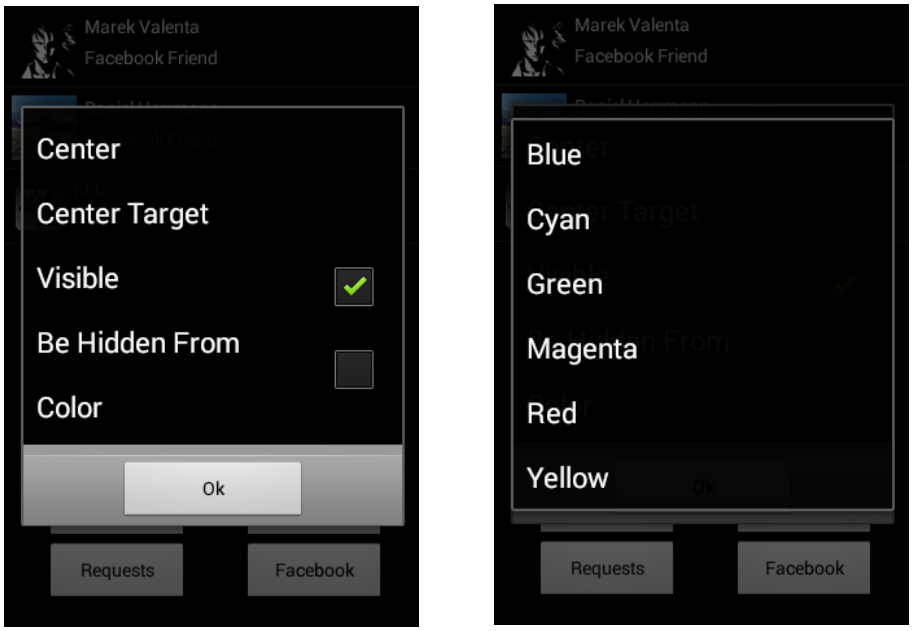
Ve spodní části aktivity je tlačítko, které umožňuje odhlášení z Facebooku.

Možnosti u přátel

Po stisknutí položky přítele v hlavním seznamu přátel se zobrazí dialogové menu s možnostmi pro práci s daným přítelem. V hlavní aktivitě s přáteli je implementované rozhraní, které umožňuje reagovat na stisknutí možnosti v tomto dialogu. Rozhraní obsahuje pět funkcí, jedna pro každou možnost (Obrázek 6.3). V případě vybírání změny barvy je spuštěno nové dialogové okno s výběrem šesti barev (Obrázek 6.3). Po výběru barev se na rozdíl od ostatních možností neposílají žádné informace zpátky do aktivity, ale přeměna barvy trasy je provedena

hned na místě. Při výběru centrování jak pozice, tak cíle přítele, je aplikace přenesena zpátky do aktivity s mapou. Informace o pozici je předána pomocí *Enginu*.

Možnost zakázat sledování uživatelovy pozice daným přítelem, je informace, která se ukládá do serverové databáze. Pro normální přátele se zapisuje přímo do záznamu o přátelství, zatímco u facebookových přátel je vytvořen nový záznam do vlastní tabulky, neboť nejsou ukládány informace o facebookových přátelích do databáze, což je přímo podmínka ve smlouvě pro používání služeb Facebook SDK.



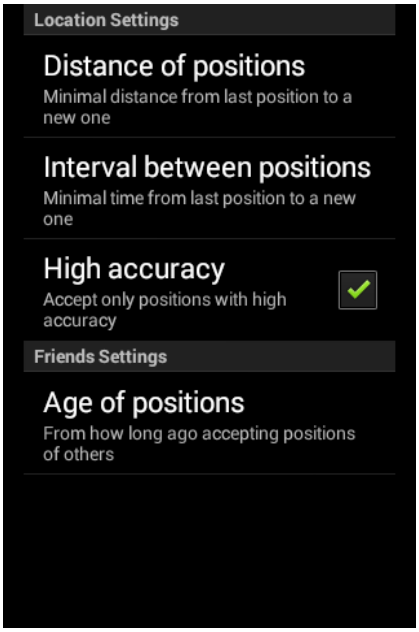
Obrázek 6.3 Dialogové okna s možnostmi přátel a výběrem barvy

6.1.9 Nastavení

Aktivita, která zobrazuje a umožňuje manipulaci s globálním nastavením, je odlišná od ostatních aktivit v aplikaci, neboť její layout obsahuje prvky, jenž jsou přímo propojené s *SharedPreferences*. Aktivita nastaví prvkům jejich hodnoty podle globálního nastavení, a při jakékoli změně v nastavení Android platforma sama uloží dané změny. Všechny prvky jsou implementovány přes seznam možností, ze kterých si může uživatel vybrat, kromě režimu vysoké přesnosti, který se pouze přepíná.

V hlavní aktivitě s mapou je vytvořen posluchač, který okamžitě reaguje na změnu v nastavení. Při zapnutí či vypnutí režimu vysoké přesnosti se změní pouze její proměnná. V případě změny doby nebo vzdálenosti pro novou pozici je třeba kromě změny proměnné i vypnout a zapnout přijímání dat z GPS modulu. U doby se navíc kontroluje, zda není zapnut

úsporný režim, který je řešen za pomocí hodin, které zapínají model GPS (vypíná se v úsporném režimu po získání pozice). Změna intervalu pro data z databáze pouze spustí nové načítání dat.



Obrázek 6.4 Aktivita s nastavením

6.2 Serverová část

Na serverové části je umístěna databáze, ve které jsou uložena všechna data potřebná pro správné fungování celé práce. Dále jsou zde umístěny skripty, které jsou volány z mobilní aplikace a jejichž nejčastější prací je získávání a ukládání dat do databáze. Server obsahuje i webové stránky, které mají podobnou i když částečně omezenou funkci jako mobilní aplikace.

Server běží přes službu OpenShift společnosti RedHat (viz 2.2).

6.2.1 Databáze

Data v databázi jsou rozdělena do šesti tabulek, které jsou navzájem mezi sebou propojeny.

Nejdůležitější tabulka *users* obsahuje data jednotlivých registrovaných uživatelů. Je tam uloženo identifikační číslo uživatele, které se dále používá pro odkazování v tabulkách, přihlašovací jméno, zahashované heslo za pomocí algoritmů SHA512 a WHIRLPOOL, salt (kryptografická sůl), která se používá pro hashování, identifikační číslo mobilního zařízení, které je už získáno pro možné budoucí rozšíření, textový status uživatele a v případě přihlášení do Facebooku i jeho identifikační číslo. Identifikační číslo slouží jako primární klíč.

Tabulka *friends* obsahuje záznamy, které jsou propojeny se dvěma různými záznamy v tabulce *users*. Pro každý takovýto záznam existuje v tabulce jiný s prohozenými uživateli. Záznam také obsahuje položku, která určuje, zda první uživatel dovoluje druhému získávat jeho pozici.

Pro zakázání sdílení pozice mezi přáteli z Facebooku slouží tabulka *hidden_facebook*, ve které je u každého záznamu uloženo identifikační číslo uživatele, který zakázal sdílení pozice, a facebookové identifikační číslo uživatele, který nebude umět přijmout data.

Žádosti se ukládají do tabulky *requests*, která obsahuje pouze dva odkazy na tabulku *users*, značící uživatele, který žádá o přátelství, a uživatele, který je žádán.

Tabulka *locations* uchovává všechny pozice všech uživatelů. Každý záznam obsahuje odkaz na uživatele, geometrický bod skládající se ze zeměpisné šířky a délky a čas, kdy byl záznam pořízen.

Cíle tras jednotlivých uživatelů se zapisují do tabulky *targets*, která má stejnou strukturu jako tabulka *locations*, jen na prvku, který odkazuje na uživatele je umístěn unikátní klíč, neboť každý uživatel může mít najednou pouze jeden cíl trasy.

6.2.2 Webové stránky

Webové stránky slouží k zobrazení dat z mobilní aplikace mimo mobilní zařízení. Obsahuje většinu funkcí, které jsou dostupné v mobilní aplikaci, neumožňuje ale získávání nových pozic a zadávání cílů. Pro implementaci stránek jsou použity jazyky HTML, PHP a JavaScript. Využívá se také JavaScriptové knihovny jQuery a facebookových SDK pro JavaScript a PHP. Pro zobrazení a manipulaci s mapou je použita javascriptová knihovna OpenLayers.

Kromě nejdůležitější hlavní stránky, která poskytuje většinu funkčnosti, je pro přihlášení na webové stránky vytvořena stránka obsahující jednoduchý dvouřádkový formulář pro přihlášení.

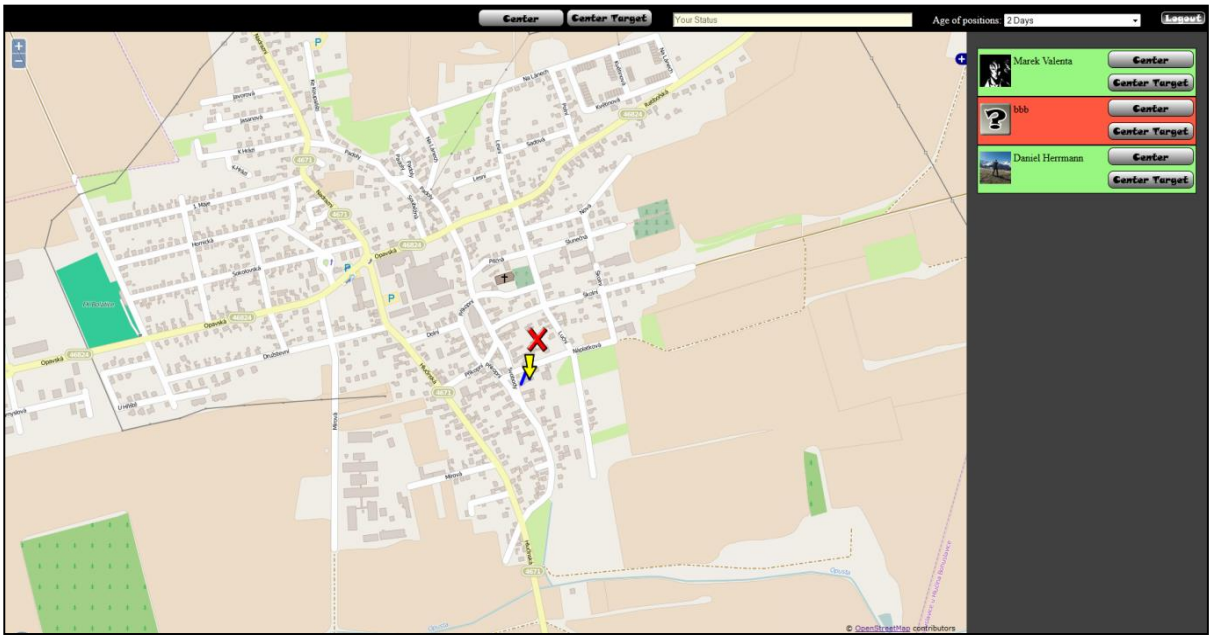
Dále podobně jako skripty pro mobilní aplikaci, jsou potřeba i skripty pro webové stránky. Tyto skripty se volají asynchronně pomocí ajaxu a umožňují načítat data z databáze bez znovunačtení stránky. Jedná se o skripty pro získání pozic a cílů.

Samotná hlavní stránka se skládá ze tří částí, kde první částí je část s mapou, která zabírá většinu obrazovky. Mapa je *javascriptový* prvek knihovny OpenLayers a jakákoli manipulace s ní probíhá pomocí JavaScriptu s využitím funkcí z této knihovny.

Další část stránky je seznam přátel, který je zobrazený na pravé straně obrazovky. Seznam se načte pomocí PHP kódu na straně serveru, a na straně klienta je dále převeden do javascriptové proměnné, aby s ním bylo možné dále manipulovat. Seznam je zobrazen jako tabulka, kde každý přítel má svou vlastní položku, ve které je jméno, obrázek a dvě tlačítka pro vycentrování mapy na jejich pozici nebo pozici jejich cíle. Centrování je prováděno přes javascriptové funkce, které jsou s těmito tlačítky provázané. Samotná položka pak slouží jako tlačítko, které vypíná a zapíná sledování osoby.

Po přihlášení se znovu načte stránka, aby bylo možné načíst přátele na straně serveru pomocí PHP. Pro zobrazení seznamu na obrazovce se normální přátele a facebookový přátele sloučí do jednoho seznamu, v javascriptovém kódu se však pro nutnost používání jiných funkcí udržují rozdělené seznamy.

Třetí část je lišta v horní části obrazovky, která obsahuje prvky pro ovládání uživatele na mapě, nastavení stáří záznamů, status uživatele a možnost odhlášení. V této části je také možné se přihlásit do Facebooku, což je řešeno přes tlačítko javascriptového Facebook SDK, které v případě, že je uživatel v daném prohlížeči přihlášen pouze vrátí *session,* jinak zobrazí přihlašovací okno.



Obrázek 6.5 Hlavní webová stránka s mapou

Po načtení stránky a převedení obou seznamů přátel do javascriptových proměnných, se pomocí ajaxu asynchronně zavolají skripty pro získání dat jednotlivých přátel a samotného uživatele. Toto se opakuje každých pět vteřin, neboť byly experimentálně zjištěny relativně nízké nároky na prostředky systému. Po načtení dat se vytvoří z pozic osoby trasa jako samostatná vrstva mapy. Poslední bod trasy se přidá do vrstvy se značkami, která je nastavena jako nejvyšší vrstva. Je to nutné kvůli zobrazení informačních oken po kliknutí na ikonku poslední lokace nebo cíle. Bez složitějších implementačních zásahů, manipulace s mapou ovlivňuje pouze nejvýše umístěnou vrstvu. Cíle se podobně jako poslední lokace přidávají do vrstvy se značkami.

7 Porovnání s existujícími aplikacemi

Sdílení polohy není téma, které by bylo zcela nové, a proto již existují aplikace, umožňující podobnou funkcionalitu. V této kapitole porovnáme naší aplikaci s dvěma známými aplikacemi zabývajícími se sdílením pozic s jinými uživateli.

7.1 Glympse

Glympse je aplikace zaměřená na sdílení pozice po kratší časové intervaly s menším množstvím osob. Typické bývá sdílení lokace s jednou osobou na dobu kratší jedné hodiny. Aplikace umožňuje sdílet lokaci s více uživateli i po delší časový interval, je to však nutné specificky umožnit. V aplikaci je možné se připojit do sociálních sítí Facebook nebo Twitter a svázat s nimi účet. Umožňuje i úsporný režim a režim jízdy v autě.

Oproti naší aplikaci poskytuje více nastavení nejrůznějších detailů a několika dalších funkcí, jako je třeba sdílení lokace po určitý čas. Na druhou stranu to způsobuje větší složitost ovládání aplikace. V případě globálního sdílení pozice má Glympse problémy, neboť vždy potřebuje uživatelovu akci ke zpřístupnění dat. Glympse navíc nedává tak velký důraz na přístup k mapě. Volná manipulace s mapou a centrování jednotlivých osob je další věc, kterou naše aplikace zvládá jednodušeji. Vybírání cílů také v naší aplikaci umožněna i bez speciálního příkazu, na druhou stranu v Glympse zase funguje vyhledávání na mapě.

Celkově Glympse poskytuje více funkcí, má trochu odlišný cíl zaměření ve sdílení (není globální, ale specifický), ale většinu věcí, které sdílí s naší aplikací, řeší naše aplikace jednodušeji a přehledněji.

7.2 Pathshare

Podobně jako Glympse i Pathshare umožňuje sdílet pozici jen po určitý čas, poskytuje však lepší prostředí pro sdílení s více lidmi. Pathshare pracuje na principu sdílení akcí, kde jeden uživatel vytvoří akci a pomocí sociální sítě či podobného prvku ji rozešle svým přátelům. Tito přátele pak mohou po zvolenou dobu sdílet pozice. Pathshare umožňuje jednoduchou manipulaci s přáteli, jejich vycentrování, schování apod. Instalace a zprovoznění aplikace je také velmi jednoduché.

Oproti naší aplikaci má Pathshare jednodušeji ovladatelné prostředí, není potřeba se nikde registrovat, pouze zvolit jméno a ovládá více druhů map. Stejně jako naše aplikace umožňuje i sledování pomocí webových stránek. Některé prvky naší aplikace se v něm však nevyskytují. Nepodporuje například vytváření tras a tedy ani historii pozic. Akce pracuje na jednorázovém principu, čili není možné mít stálé přátele, ale musí se vždy vytvářet znovu rozesíláním pozvánky. Cíle není taky možné vytvářet a manipulace s mapou není bez vytvoření akce možná.

Pathshare je určen pro příležitosti, kdy není potřeba sledovat pozici lidí po tom, co skončí daná akce. Jeho jednoduchost je vykoupena některými chybějícími funkcemi, které naše aplikace poskytuje.

8 Závěr

Výsledkem této práce je mobilní aplikace, která umožňuje získávat, sdílet a zobrazovat pozici uživatele a jeho přátel. Byla vytvořena i webová alternativa, která rovněž dovoluje zobrazit stejná data jako mobilní aplikace. Tato práce přináší možnost sdílení pozice mezi větší skupiny lidí po neomezeně dlouhou dobu, což je ve většině současných mobilních aplikací stejného typu spíše výjimka. K tomu všemu dovoluje aplikace zobrazovat ušlé trasy a cíle cest uživatelé, propojení se sociální sítí Facebook a upravovat různé funkce aplikace podle přání uživatele.

Tato práce mi osobně přinesla zkušenosti v oblasti programování aplikací, které mají svůj kód rozmístěn po více platformách. Instalování a programování s různými Facebook SDK bylo odměňující, i když místy velmi frustrující. Zlepšení znalostí jazyka Android a Java je také nezanedbatelným přínosem.

Mezi možná rozšíření této práce by mohlo patřit příjemnější grafické prostředí pro uživatele, ne však za cenu zvýšení složitosti interakce s aplikací. Dalším rozšířením by mohlo být umožnění posílání textových či jiných zpráv přímo pomocí této aplikace. Bylo by také vhodné pečlivěji otestovat kritické situace, které mohou v průběhu práce aplikace nastat. Přestože bylo mnoho kritických situací ošetřeno, kvůli složitému komunikačnímu systému aplikace s jinými prvky jako je server nebo Facebook, určitě existují další. Tyto situace není ve většině případů těžké opravit, nejsložitější je je najít nebo si vůbec uvědomit, že můžou nastat.

9 Literatura

[1] Cloud Computing Glossary. Apprenda. [online]. [cit. 2015-05-01]. © 2015-. Dostupné z: http://apprenda.com/library/glossary/.

[2] "The NIST Definition of Cloud Computing" (PDF). National Institute of Standards and Technology. [online]. [cit. 2015-04-29]. Dostupné z: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[3] RENDE, Kevin. SaaS, PaaS, and IaaS. Oh my! ("Cloudy April" - Part 3). *Full of I.T.* [online] 3. duben 2011. Dostupné z: http://blogs.technet.com/b/kevinremde/archive/2011/04/03/saas-paas-and-iaas-oh-my-quot-cloudy-april-quot-part-3.aspx.

[4] CINAR, Onur. Android apps with Eclipse. Berkeley, CA: Apress, 2012. ISBN 9781430244356.

[5] MEIER, Reto. Professional Android 4 application development. [3.ed.]. Indianapolis, IN.: Wiley/[Wrox], 2012, xlii, 817 str.

[6] Facebook Inc.: Documentation for building software with Facebook [online]. https://developers.facebook.com/docs/

[7] Google Inc.: Android Developers [online]. http://developer.android.com/

[8] OpenLayers documentation [online]. http://dev.openlayers.org/releases/OpenLayers-2.13.1/doc/apidocs/files/OpenLayers-js.html

Příloha A

Obsah DVD

- Složka Source_Codes – Obsahuje čistě zdrojové kódy napsané autorem této bakalářské práce
- Složka Applications – Obsahuje všechny soubory nutné pro zkompilování aplikace, nahrání webových stránek a vytvoření databáze.
- Složka Installation – Obsahuje instalační soubor mobilní aplikace. Pro nainstalování přesuňte tento soubor na mobilní zařízení a spusťte ho. Je potřeba minimálně Android 3.0.
- Složka Documentation – Obsahuje tento dokument v PDF verzi