VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Android aplikace pro time-management s automatizací

Automatic Time-Management Application for Android

2016 Petr Macák

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Petr Macák**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Android aplikace pro time-management s automatizací
Automatic Time-Management Application for Android

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je navrhnout a implementovat mobilní aplikaci na platformě Android pro měření a správu času strávených na různých projektech. Aplikace bude umožňovat rozpoznávání práce na jednotlivých projektech díky senzorům a technologiím, které mobilní zařízení nabízí (např. GPS, WiFi, pohybový senzor). V případě potřeby umožní také manuální zápis odpracovaného času. Výsledné řešení bude podporovat export a import pořízených dat a jejich pokročilou prezentaci.

Řešení bude obsahovat:
1. Srovnání aplikací pro time-management na mobilních platformách.
2. Popis možností senzorů a jejich využití pro automatickou klasifikaci činností.
3. Implementaci vlastní mobilní aplikace pro time-management.
4. Otestování navrženého řešení, vyhodnocení spolehlivosti automatického určování činností.
5. Závěr a shrnutí dosažených výsledků.

Seznam doporučené odborné literatury:

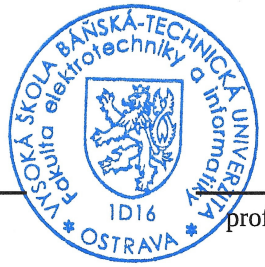
[1] Reto Meier, Professional Android, Wrox; 4 edition, 2015. ISBN 978-1118949528
[2] Joseph Annuzzi Jr. et al., Advanced Android Application Development, Addison-Wesley Professional; 4 edition, 2014. ISBN 978-0133892383
[3] Greg Milette, Professional Android Sensor Programming, Wrox; 1 edition, 2012. ISBN 978-1118183489

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Mgr. Ing. Michal Krumnikl, Ph.D.**

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 25. dubna 2016 .....

Rád bych na tomto místě poděkoval všem, kteří mi pomohli s touto bakalářskou prací a hlavně vedoucímu mé bakalářské práce Mgr. Ing. Michalu Krumniklovi, Ph.D. za jeho rady a připomínky.

Abstrakt

Cílem této bakalářské práce je zaměřit se na návrh a implementaci mobilní aplikace postavené na platformě Android, určené pro měření a správu času stráveného na různých projektech s důrazem na automatizaci rozpoznávání těchto uživatelem zadaných projektů. K dosažení automatizace bude aplikace využívat senzory a technologie, které moderní mobilní zařízení nabízí (GPS, WiFi, pohybový senzor). Aplikace bude navržena na základě výzkumu současného stavu vývoje aplikací pro time-management na všech mobilních platformách a také na obchodě Google Play. Dále proběhne testování funkčnosti a spolehlivosti aplikace. V závěru práce provedu shrnutí a vyhodnocení dosažených výsledků při testování.

Klíčová slova: Time-management, automatizace, Android, GPS, WiFi, pohybový senzor, Google Play

Abstract

The aim of this bachelor thesis is to focus on design and implementation of a mobile application based on Android platform with intention of measuring and management of time spent on various projects with emphasis on automatization of detection of these user-predefined projects. To achieve the automatization the application will use sensors and technology that modern mobile devices offer (GPS, WiFi, motion sensor). The application will be designed based on research of current state of Android application development for time-management applications offered on all mobile platforms and of course also on Google Play catalogue. After implementation, the application will be tested for reliability. In conclusion I will summarize the thesis and present evaluation of the results obtained from testing.

Key Words: Time-management, automatization, Android, GPS, WiFi, motion sensor, Google Play

Obsah

Seznam použitých zkratek a symbolů

AP	–	Access Point
API	–	Application Programming Interface
BTS	–	Base Station Subsystem
GPS	–	Global Positioning Unit
GUI	–	Graphical User Interface
OS	–	Operating System
MAC	–	Media Access Control
SSID	–	Service Set Identifier
UI	–	User Interface
WiFi	–	Wireless Fidelity

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

V dnešní uspěchané době je stále častěji kladen důraz na správnou a efektivní organizaci svého osobního času - ať už v práci, či v osobním volnu. Vznikají stále nové způsoby a postupy, jak zefektivnit své hospodaření s časem (GTD - Getting Things Done, Don't Break The Chain!, Pomodoro technika a další). Většinou jde o dodržování určitých rituálů, které mají ve výsledku za úkol zefektivnit a lépe organizovat naši práci a pomáhají nám si vytvořit určitý systém.
I k tomuto slouží time-management. Ke splnění různých představ, dodržování termínů u projektů či například plnění pravidelných povinností.

Vše se podstatně zjednodušuje s příchodem "chytrých"mobilních telefonů, které umožňují uživateli rychlý přístup k internetu, komunikaci s přáteli prakticky odkudkoli a možnost využívat milióny různých aplikací na cestách. Takovýto "chytrý"mobilní telefon se vyznačuje tím, že je postaven na moderním operačním systému jako je například Android, iOS, Windows Mobile, Windows Phone, Blackberry a například také Symbian OS. Tento systém je pak jádrem telefonu a umožnuje na něm pracovat stejně, či podobně jako s klasickým stolním počítačem.

Uživatel tak dostává možnost si svůj čas organizovat pohodlně a přehledně pomocí aplikací na telefonu kdykoliv a kdekoliv - ať už se jedná o běžný záznam úkolů (To-Do aplikace), pokročilé vedení jednotlivých částí uživatelem zadaných projektů či záznam odpracovaných hodin na jednotlivých projektech. Výhodou je tedy nejen mobilita a pohodlí, které tyto aplikace nabízejí, ale také například možnost sdílení úkolů s kolegy a další funkce pro usnadnění práce.

V následujících kapitolách tato bakalářská práce provede čtenáře celým procesem vytváření aplikace pro mobilní zařízení. Ze začátku porovnám dostupné aplikace pro time-management na mobilních platformách z pohledu uživatele a vymezím, jaké požadavky by měla splňovat "dokonalá"aplikace pro time-management. Dále jsou zde popsány všechny dostupné senzory na moderních mobilních zařízeních, jejich užití a vhodnost pro automatickou detekci a klasifikaci činností. Na základě výzkumu pak následuje vlastní návrh a popis implementace aplikace, jež automatizuje záznam odpracovaného času na jednolivých uživatelem zadaných projektech na základě informací získaných ze senzorů z mobilního zařízení. V neposlední řadě tato práce nabídne dosažené výsledky testování mnou navrženého řešení a vyhodnocení spolehlivosti automatického určování činností.

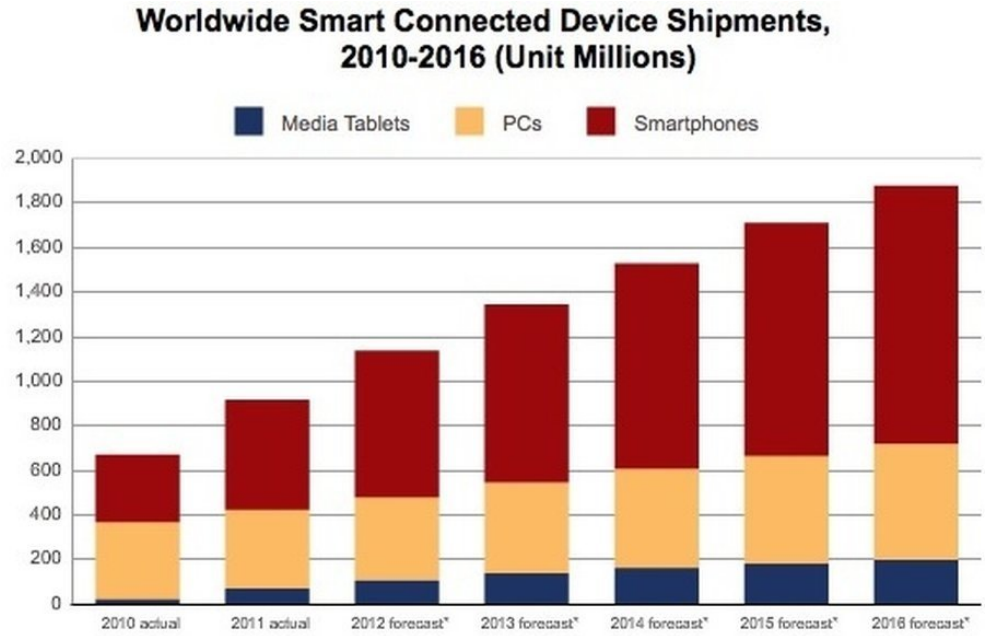
1.1 Motivace

Přestože, jak již bylo zmíněno, existuje celá řada aplikací pro time-management a zaznamenávání času na jednotlivých projektech, téměř vždy je nutný manuální zápis těchto hodnot - vykonaných aktivit, odpracovaného času či záznam úkolů. Mobilní zařízení, ať už telefon či tablet, přitom dnes již běžně nabízí funkce a technologie umožňující získání relevantních informací pro detekci a klasifikaci činností a polohy uživatele a tudíž je možné tento proces automatizovat. Právě na tuto vlastnost se během návrhu a implementace vlastní aplikace zaměřím.

2 Rešerše aplikací pro time-management

Před samotným návrhem a implementací vlastní time-management aplikace bylo zapotřebí provést analýzu aplikací již dostupných na mobilní zařízení. Následující analýza je rozdělena do kategorií dle operačního systému telefonu či mobilního zařízení, pro které jsou dané aplikace nabízeny. Pro samotnou rešerši jsem vybíral ty aplikace, které měly nejblíže k mnou vybrané aplikaci pro time-management (zápis času stráveného na různých činnostech).

Od komerčního uvedení mobilních zařízení v osmdesátých letech 20. století jsme mohli být svědky obrovského a rychlého vývoje těchto zařízení. Ač hlavním úkolem mobilních telefonů bylo právě volání, dnes je toto bráno jako pouze jedna z mnoha jejich funkcí. Moderní smartphone, který máme možnost nyní pořídit, tedy prodělal mnoho změn a neustále probíhá jeho další vývoj[1]. Trh mobilních telefonů stále roste a nynější předpoklady udávají, že v roce 2017 bude více než jedna třetina světové populace vlastnit smartphone[2].



Obrázek 1: Vývoj trhu mobilních zařízení [11]

V roce 2010 pak nastala nová éra tabletů díky uvedení Apple iPad a dalších na trh[3]. Tato zařízení pak dále vyplňují mezeru mezi mobilními telefony a klasickým stolním počítačem a nabízí kompromis mezi mobilitou a komfortem stolního PC. Vedoucí výrobci tabletů již však označují tento boom za ukončený a jejich prodeje zaznamenávají roční poklesy.[4]. Současným

[1] http://www.hongkiat.com/blog/evolution-of-mobile-phones/
[2] http://www.emarketer.com/Article/Worldwide-Smartphone-Usage-Grow-25-2014/1010920
[3] http://techdomino.com/tablets-outsell-pcs-laptops/
[4] http://www.ibtimes.com.au/apple-samsungs-tablet-era-dying-iphone-7-samsung-galaxy-s6-boom-report-1420101

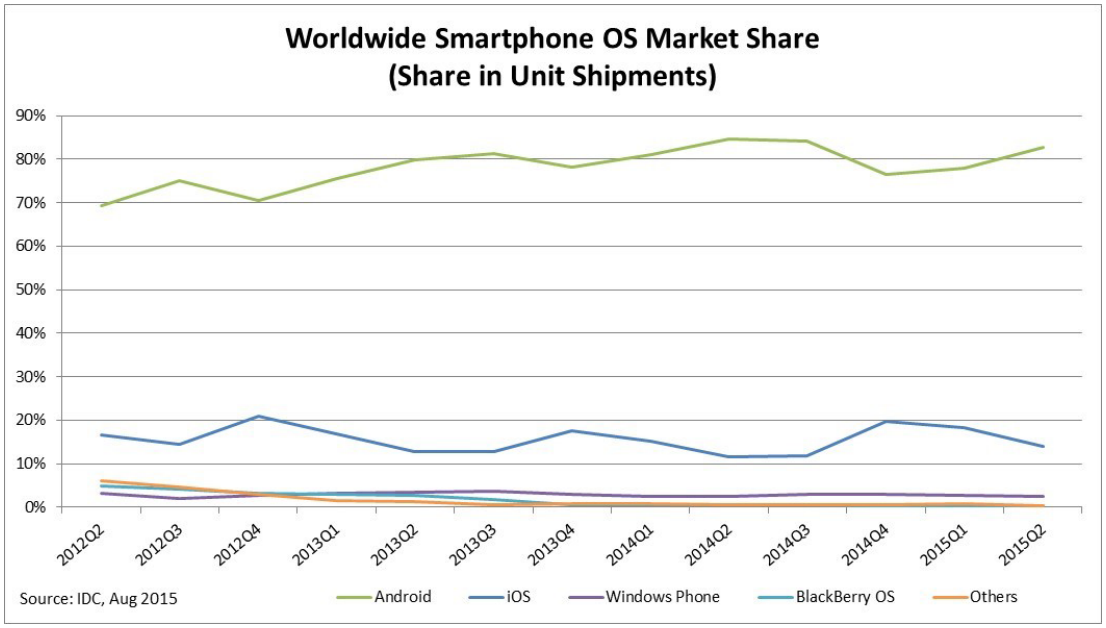
trendem, který pomalu nahrazuje tablety jsou takzvaná 2-in-1 zařízení, která fungují současně jako tablet a po připojení fyzické klávesnice pak nahrazují notebook.

Tento prudký vývoj mobilních zařízení znamenal také možnost pro velké společnosti a jednotlivce vyvíjet software pro tato zařízení. Během několika měsíců od uvedení například operačního systému Android obsahoval jeho aplikační katalog již tisíce aplikací. Mobilní telefony a tablety tak můžeme dále kategorizovat dle operačního systému, na kterém jsou postaveny. Právě tento OS definuje trh aplikací svým online aplikačním katalogem možnou konkurenci a má svůj určitý podíl na trhu. Volba správné platformy, případně platforem je tak pro vývojáře či firmu velmi důležitá.

2.1 Mobilní platformy a jejich aplikační katalogy

Jak již bylo zmíněno výše, operační systém má k dispozici právě jeden aplikační katalog, kde může vývojář svou aplikaci publikovat. Pokud dnešní trh mobilních zařízení rozdělíme dle podílu OS, největší zástupci jsou Android, iOS, Windows Phone a Blackberry.

Jedno z nejtěžších rozhodnutí, jaké musí každý vývojář či firma udělat je výběr platformy, pro kterou danou aplikaci vyvíjet. Problémem je, že aplikace napsaná pro jednu platformu nelze spustit a ani instalovat na platformě druhé. Důvodem je, že například aplikace vyvíjené pro Android jsou psány v Javě. Aplikace pro iOS jsou programovány v jazyce Objective-C a aplikace pro Windows Phone jsou psány v C#.



Obrázek 2: Vývoj podílu trhu OS smartphonů [11]

Každá platforma má také svůj podíl na trhu a tudíž vývojář či firma při své volbě musí brát v potaz i počet potencionálních zákazníků. Když nemůžeme vyvíjet software pro všechny

platformy, logickým krokem je vybrat si platformu s největším zastoupením trhu. K roku 2015[5] je to operační systém Google Android s přibližně 80% podílu na světovém trhu. Za ním následuje Apple iOS, Microsoft Windows Mobile a další. Ačkoli má však Android většinový podíl trhu a dominuje také v počtech stažených aplikací, tak výdělky pro vývojáře a firmy má větší iOS[6]. Důvodů je mnoho, například že uživatelé Androidu nejsou zvyklí za aplikace platit, s čímž také souvisí pořizovací cena Android telefonu oproti Apple zařízením.

2.2 Aplikace dostupné pro Android

Time-management aplikace dostupné pro OS Android jsem vyhledával na online aplikačním katalogu Google Play. Tento katalog je největší mezi mobilními platformami, co se celkového počtu dostupných aplikací týče a tudíž jsem zde nalezl největší zastoupení mezi time-management aplikacemi. Níže jsem rozebral ty nejpopulárnější aplikace pro sledování času.

2.2.1 Jiffy - Time tracker

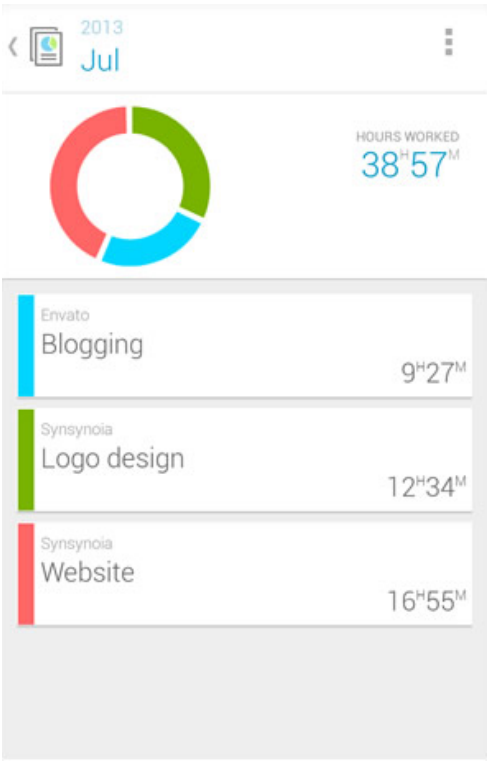
Tato aplikace od firmy Nordic Usability na první pohled zaujme pěkným a přehledným UI. Nabízí uživateli organizaci práce do projektů a manuální zápis hodnot, případně lze spustit měření času v aplikaci, po kterém se objeví v liště notifikace o probíhajícím měření. Dále disponuje přehledným souhrnem a grafy pro odpracované hodiny a módem kalendáře, ve kterém je přehledně vidět odpracovaná doba na projektech a možnosti zálohování dat.

Aplikace je poskytována zdarma, ale nabízí mikroplatby pro odstranění limitu na počet projektů (pro více než 3 projekty), historii záznamů a synchronizaci dat přes internet po vytvoření uživatelského účtu.

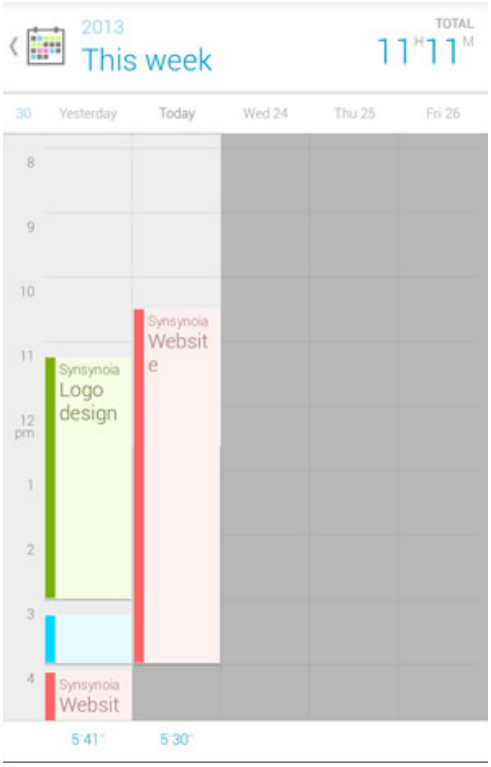
Za příplatek je zde také podpora NFC "JiffyTagů"[7], které umožňují alespoň částečnou automatizaci. Každému takovému tagu v aplikaci uživatel přiřadí projekt a po kontaktu s telefonem se spustí měření času.

Mezi nevýhody aplikace patří například dostupnost pouze v anglickém jazyce, nemožnost exportu dat a nastavení připomenutí.

[5] http://www.idc.com/prodserv/smartphone-os-market-share.jsp
[6] http://bgr.com/2015/04/15/ios-vs-android-developers-revenue-apps/
[7] http://jiffy.nu/jiffytags/



Obrázek 3: Jiffy - seznam projektů

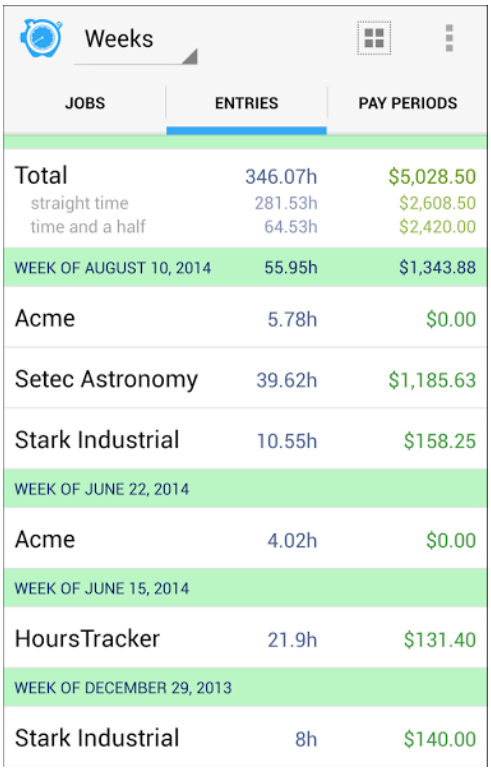


Obrázek 4: Jiffy - pohled kalendáře

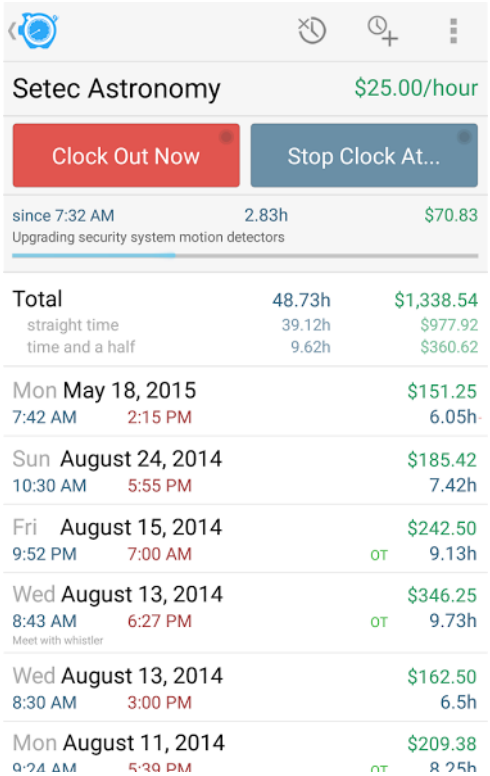
2.2.2 Aplikace HoursTracker: Time Tracking

HoursTracker od stejnojmenné firmy je aplikace dostupná jak pro Android platformu, tak také pro iOS. Její poměrně strohý design je oproti jejím konkurentům poměrně nevýhodou, nicméně aplikace je přehledná. Nabízí rozdělení práce do aktivit a umožňuje manuální zápis času na nich strávených. Tento čas pak automaticky přepočítává dle zadané měny a hodinové mzdy přímo na čistý výdělek. Aplikace kromě výpisu záznamů a aktivit nedisponuje žádným grafickým zobrazením hodnot - tedy žádné grafy a tabulky. Dále je zde možnost exportu dat a také zálohu a synchonizaci dat do cloud služby aplikace.

HoursTracker je nabízena zdarma, ovšem obsahuje vcelku rušící reklamu a omezený počet aktivit a záznamů. Jako jediná time-management aplikace na platformě Android však disponuje částečnou automatizací zaznamenávání času aktivit. Díky GPS umožňuje upozornit uživatele oznámením, či přímo začít zaznamenávat čas, pokud je zařízení v přednastaveném místě pro danou aktivitu. Tato funkce je zde zatím jako funkce experimentální.



Obrázek 5: HoursTracker - shrnutí záznamů



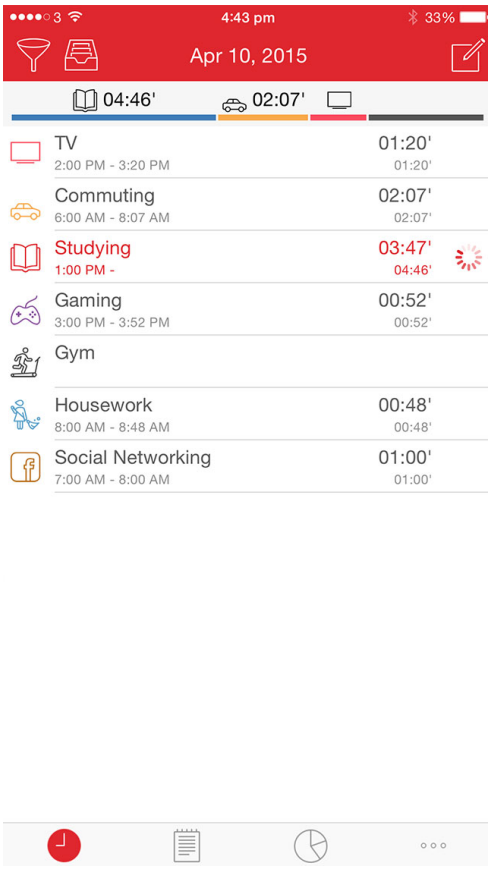
Obrázek 6: HoursTracker - okno aktivity

2.3 Aplikace dostupné pro iOS

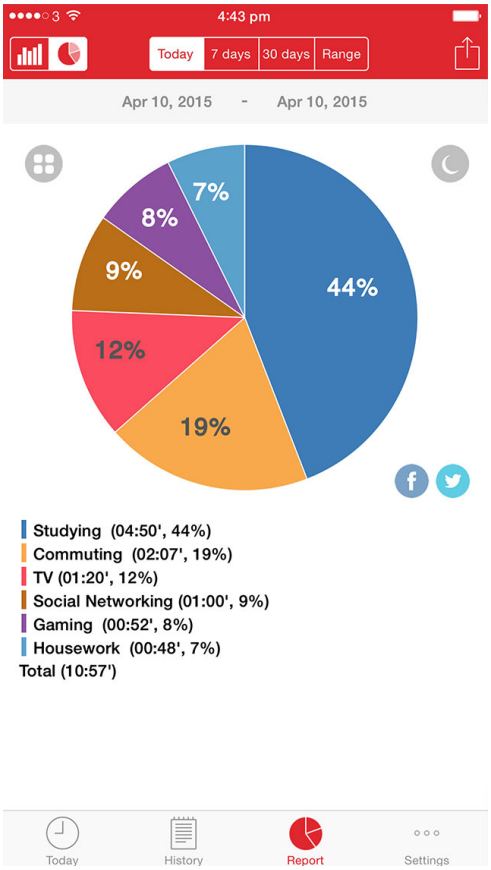
Katalog aplikací pro iOS zařízení - Apple iPhone, iPad, iPod a další je druhým největším dostupným katalogem pro mobilní zařízení. Některé aplikace jsou mutliplatformní (např. HoursTracker), dostupné jak pro Android, tak také pro iOS platformu. Jako aplikace dostupné pouze pro iOS trh zde stojí za to zmínit například ATracker či Tyme.

2.3.1 Aplikace ATracker

ATracker je time-management aplikace švédské firmy, která se chlubí minimální potřebnou konfigurací, elegantním designem a možností spuštění záznamu času jedním dotykem. Kromě klasického zobrazení seznamu aktivit a záznamů disponuje také zobrazením kalendáře či vykreslením grafů pro zadané časové období. Oproti konkurentům nenabízí přepočet odpracovaného času na čistý výdělek v uživatelem zadané měně. Umožňuje exportovat data do souboru .csv, případně vytvářet zprávy pro sdílení na sociálních sítích. Dále nabízí přidání widgetu na plochu zařízení pro snadné spuštění záznamu aktivit, ale kromě tohoto způsobu záznamu dat žádnými dalšími prvky automatizace nedisponuje. Aplikace je dostupná zdarma, nicméně tato verze je omezená počtem možných uživatelských aktivit. V případě, že uživatel vlastní zařízení Apple Watch, je zde možný přístup k aplikaci jednoduše i přes tyto "chytré"hodinky.



Obrázek 7: ATracker - Seznam aktivit



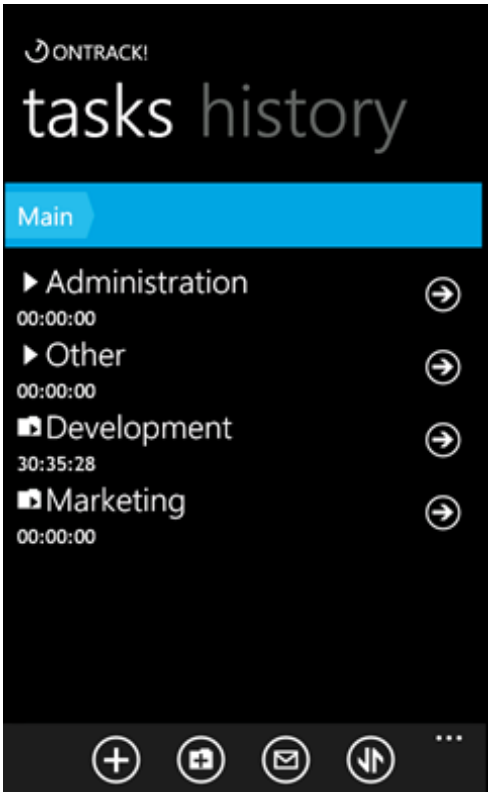
Obrázek 8: ATracker - Zobrazení grafu

2.4 Aplikace dostupné pro Windows Phone

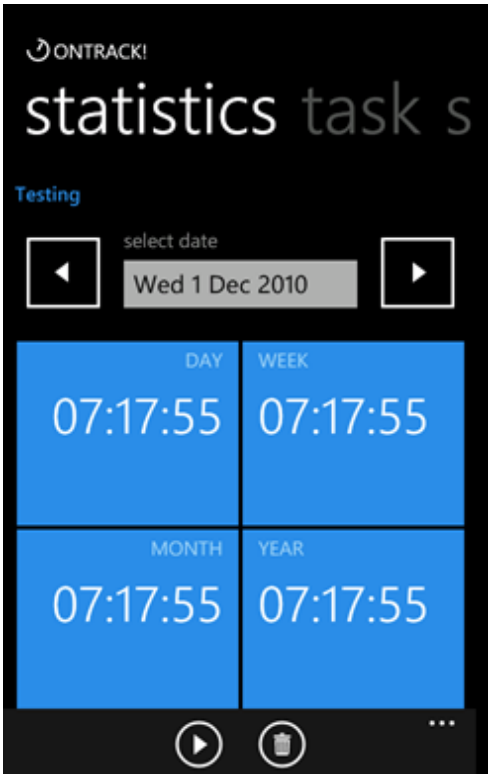
Platforma Windows Phone je poměrně novým trhem pro aplikace a tudíž jse zde nejmenší zastoupení time-tracking aplikací. Jako zástupce této platformy jsem si vybral poměrně jednoduchou aplikaci ONTRACK!.

2.4.1 Aplikace ONTRACK!

Jako většina Windows Phone aplikací má tato svůj specifický, možná až příliš textový design. Aplikace rozděluje aktivity na úkoly (jednoduché aktivity) a projekty (s možností vytváření subprojektů). Vlastní funkci zaznamu času dotekem v programu a také je možný manuální zápis hodnot. Je zde navíc možné zobrazit souhrn pro dané aktivity a záznamy pro určité časové období a možnost exportu dat do .csv souboru. Celkově aplikace nabízí poměrně strohé nastavení, není zde žádné grafické zobrazení hodnot (grafy, mód kalendáře) a aplikace je placená.



Obrázek 9: ONTRACK! - Seznam aktivit



Obrázek 10: ONTRACK! - Zobrazení statistik

2.5 Vyhodnocení výzkumu

Po důkladném průzkumu vlastností aplikací dostupných na všech třech největších mobilních platformách je na řadě jeho shrnutí. Co se počtu aplikací pro jednotlivé platformy týče, není překvapením, že time-tracking aplikací pro Android je nejvíce, následována platformou iOS. Většina aplikací slouží pro jednoduchý zápis odpracovaného času, ale dá se říci, že na každé platformě existuje pár aplikací, které nabízí uživateli i komplexnější funkce - ať už synchonizaci s cloud službou, export dat, propracované UI, pokročilé možnosti zobrazení dat či možnost přidání widgetů na plochu zařízení.

Co se možností automatizace týče, zde byly nejdále aplikace dostupné pro Android. Jiffy nabízí možnost začít či ukončit sledování času na určité aktivitě díky kontaktu zařízení a speciálních fyzických NFC tagů (přívěšků), které lze zakoupit v obchodě aplikace. Samozřejmě je nutné, aby zařízení NFC technologii podporovalo, což je v dnešní době spíše standartem u vyšších tříd mobilních telefonů a zařízení. Nejvíce automatizovaná dostupná aplikace je HoursTracker, která nabízí oznámení či spuštění záznamu času dle aktuální polohy zařízení získané prostřednictvím GPS. Žádná testovaná aplikace pro iOS nevyužívala sensorů pro automatické měření.

Android	Manuální zápis dat	Automatizace	Grafy	Export dat	Online synchronizace
Jiffy - Time Tracker	●	NFC	●	●	●
Mobile Worker - Time Tracker	●	○	○	●	●
HoursTracker: Time Tracking	●	GPS	○	●	●
Toggl	○	○	●	○	●
aTimeLogger	●	○	●	●	●
TimeSheet - Time Tracker	●	GPS	●	●	○
iOS	Manuální zápis dat	Automatizace	Grafy	Export dat	Online synchronizace
Atracker	●	○	●	●	○
Tyme	●	○	●	●	●
aTimeLogger	●	○	●	●	●
Way of Life	●	○	●	●	○
Windows Phone	Manuální zápis dat	Automatizace	Grafy	Export dat	Online synchronizace
ONTRACK!	●	○	○	●	○
timr	●	GPS	○	●	○
My timesheet	●	○	○	●	○

Tabulka 1: Zkrácená tabulka výzkumu

2.5.1 Ideální time-tracking aplikace

Jak dokázal průzkum, time-tracking aplikací je pro všechny mobilní platformy spousta. Z tohoto počtu aplikací jsem získal seznam vlastností, které by měla taková "ideální" time-tracking aplikace mít. Mezi tyto vlastnosti zcela určitě můžeme zařadit možnost tvorby uživatelem zadaných aktivit, manuální zápis hodnot času, možnost spuštění a ukončení časomíry (nejlépe např. z notifikační lišty), možnost exportu dat či jejich synchronizace a také pokročilá prezentace naměřených hodnot (grafy, statistiky, přehledy).

Většina těchto funkcí je obsažena v mé implementaci aplikace. Navíc se ale zaměřuji na automatizaci měření času s využitím senzorů, které mobilní zařízení nabízí. Výpis těchto senzorů a jejich vhodnost pro měření a detekci aktivit následuje v další kapitole této práce.

3 Popis senzorů a možnosti jejich využití

Většina mobilních zařízení dnes disponuje senzory, s jejichž pomocí jsou schopné do určité míry vnímat okolní svět. Jsou schopny získávat informace o poloze, nadmořské výšce, teplotě, zrychlení, zvuku a dalších veličinách, se kterými se můžeme setkat. V této kapitole se budu věnovat senzorům a jejich možnostem využití.

3.1 Senzory

Moderní mobilní zařízení dnes disponuje spoustou možností, jak získávat informace z okolí. Dále se budu věnovat těmi senzory, které mohou být použity pro automatizaci v implementaci mé aplikace pro time-tracking. U každého takového senzoru popíši ve zkratce, jak funguje, jak jej můžeme využít v systému Android a jaká API a možnosti Android pro programátory nabízí.

3.1.1 GPS

Znalost lokace telefonu se stává čím dál více důležitější součástí programování pro mobilní zařízení. Aplikace jsou díky této funkci schopny poskytovat uživateli relevantnější informace, ať už se jedná o pouhé vyhledávání na internetu či například autonavigaci. V mé implementaci je nutné získat lokaci uživatele, aby bylo možné spustit časomíru pro záznam času na aktivitě. Právě jednou z možností, jak lokaci získat, je technologie GPS.

Global Positioning System (GPS) využívá systému satelitů, které krouží okolo planety Země, k získání momentální polohy mobilního zařízení. Systém GPS využívá 27 satelitů s pevně danou trajektorií okolo Země pro vysílání informací o poloze příjímačů na Zemi. Aby bylo možné přijímač přesně lokalizovat, je nutný nejlépe přímý výhled na minimálně 4 takové satelity, s každým dalším se pak zvyšuje přesnost určení polohy.

Jednou z nevýhod GPS na mobilních zařízeních je čas, za který jsou schopny získat svou polohu z vysílání satelitů. Tento proces může trvat i několik minut, a proto zde existují určité technologie, které pomáhají čas pro získání polohy snížit. Jednou z nich je A-GPS, která využívá mobilní sítě, případně internetu, pro získání aktuální polohy satelitů okolo Země (včetně těch, na které je přímý výhled), což může vést k rychlejšímu získání polohy, protože zařízení nemusí získávat tyto informace přímo ze satelitů.

Další technologií vylepšující tento systém je S-GPS. Protože GPS využívá stejný hardware pro komunikaci se satelity a mobilní sítí pro uskutečnění hovorů, tato technologie přidává do zařízení dodatečný hardware, aby bylo možné využívat systém GPS současně se systémem GSM.

Mezi hlavní výhodu technologie GPS patří velmi přesné určení polohy (v ideálních podmínkách i v rámci jednotek metrů). Nevýhod má více - je nutný přímý výhled na satelity, což prakticky znemožňuje využití uvnitř budov a v hustě pokrytých lesech či metropolích. Dále je zde čas nutný k získání polohy, který se může pohybovat v řádech sekund až minut. Další nevý-

hodou je spotřeba baterie, která je na poměry přenosného zařízení poměrně vysoká. Záleží zde tedy na intervalu mezi získáváním poloh, čím menší interval, tím větší je spotřeba baterie.

Většina nástrojů pro získání polohy (včetně pomocí technologie WiFi a GSM) jsou pro systém Android obsažena v balíčku android.location. Tento balíček obsahuje třídy LocationManager, LocationProvider, Location, Criteria a rozhraní LocationListener, které slouží jako API pro přístup k datům poskytovaných senzory pro určení polohy.

V mé implementaci aplikace pak využívám technologii GPS pro určení polohy zařízení pro automatizaci. Pokud bude zařízení v dosahu uživatelem zadaného místa, bude spuštěno zaznamenávání času k této aktivitě. Tato kontrola bude probíhat vždy v určitém intervalu.

3.1.2 WiFi a síť GSM pro určení polohy

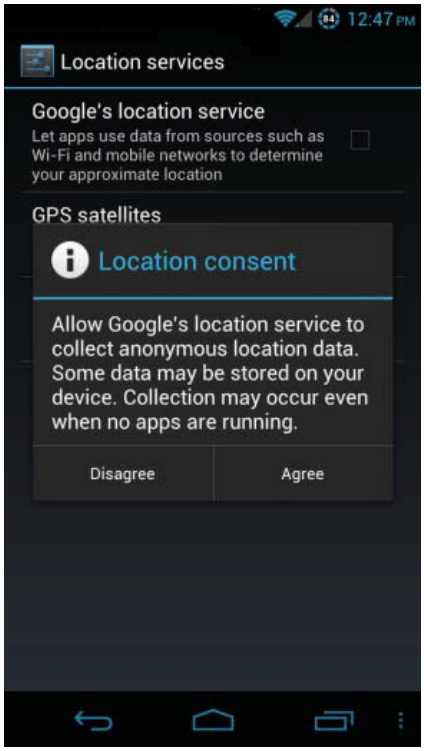
V systému Android je možno jako dalšího poskytovatele polohy použít technologii WiFi či rádiovou síť GSM. Většina chytrých telefonů je vybavena možností připojení k internetu pomocí WiFi a tak společnost Google přišla na způsob, jak takto získat polohu zařízení.

Určení polohy pomocí WiFi funguje tak, že zařízení sleduje přístupové body a jejich sílu signálu. Tento seznam AP pak odešle službě Google Location včetně MAC adres přístupových bodů a služba vrátí zářízení přibližnou polohu díky síle signálů (čili teoretické vzdálenosti) k těmto AP. Aby tento systém získávání polohy fungoval, je nutné, aby služba Google Location měla informaci o poloze těchno AP. Toto je realizováno pomocí funkce "Služba pro zjišťování polohy Google"v systému Android, která umožňuje pravidelně sledovat přístupové body pomocí WiFi a aktuální polohu pomocí GPS a tyto informace pak anonymně zasílá na servery společnosti Google. Takto získaná poloha je tedy relativně přesná, protože tato data jsou neustále aktualizována nespočtem Android zařízeními.

Výhodou této technologie je možnost získání polohy zařízení i uvnitř budov. Navíc vyhledávání AP pomocí WiFi je úspornější, co se baterie týče, než GPS. Stinnou stránkou pak je menší přesnost oproti systému GPS (v rámci desítek až stovky metrů) a nutnost přítomnosti přístupových bodů v blízkosti zařízení. Tyto AP navíc musí vysílat své SSID a také nastávají problémy, když je přístupový bod přemístěn, protože Google nenabízí explicitní nastavení polohy AP a tak data nemusí být vždy úplně přesná.

Na stejném principu pak funguje i získání polohy pomocí sítě GSM. Služba Google Location shromažďuje informace o poloze systému základnových stanic BTS získaných z Android zařízení a pomocí jejich síly signálu pak určuje polohu. Služba tedy ukládá unikátní ID základnových stanic a jejich polohu. Výhodou je nízká spotřeba baterie pro určení polohy a možnost získání polohy kdekoli, kde je dostupná mobilní síť GSM. Nevýhodou je velmi malá přesnost (v rámci až stovek metrů) v oblastech s malým počtem stanic BTS.

Pokud se rozhodneme v aplikaci využívat těchto způsobů získání lokace, v systému Android pak využíváme stejného balíčku android.location jako v případě GPS. V mé implementaci aplikace využiji také technologie WiFi a GSM pro získání polohy uživatele a automatizaci obdobně jako s GPS.



Obrázek 11: Zapnutí služeb zjišťování polohy

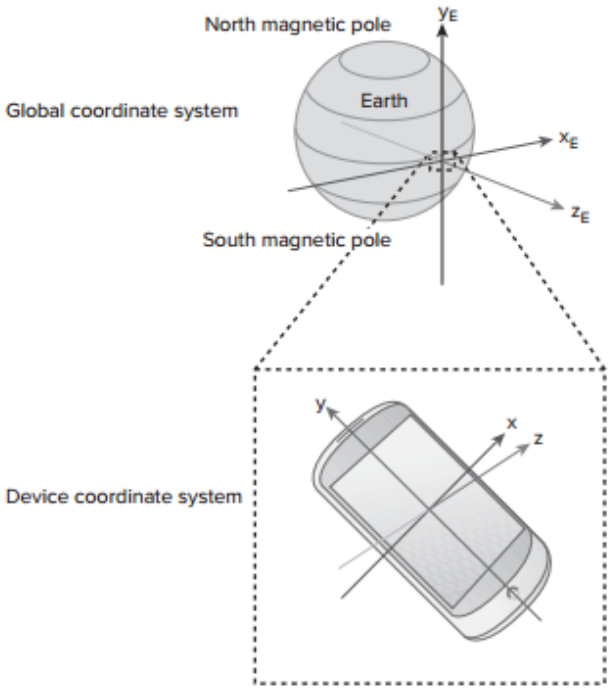
Rovněž jako u GPS v mé implementaci aplikace využívám technologie WiFi a GSM pro určení lokace zařízení v případech, kdy signál GPS není dostupný či pouze za účelem šetření baterie zařízení.

3.1.3 Senzor pohybu

Pohyb a akceleraci zařízení můžeme měřit pomocí zabudovaného akcelometru v telefonu. Akcelometr je senzor, který měří vibrace nebo zrychlení při pohybu. Funguje na principu piezoelektrického jevu, což je schopnost krystalu generovat měřitelné elektrické napětí při jeho deformaci pohybem. S akcelometrem je úzce spojen také gyroskop, který nám umožňuje detekovat pohyb zařízení také v třetí ose a je systémem využíván např. pro určení naklonění či orientace přístroje.

Akcelometr tak měří lineární zrychlení pohybu - je schopen určit směr, kterým se zařízení pohybuje na dvou osách. Díky gyroskopu pak můžeme zaregistrovat i pohyb na třetí ose pomocí úhlové rychlosti otáčení. Tyto senzory se tak navzájem doplňují a lze je využít pro detekci a klasifikaci pohybu zařízení.

Díky akcelometru jsme tedy schopni získat data pro zrychlení na osách x, y a z. Dalším úkolem je tato data správně klasifikovat jako jednotlivé činnosti. Identifikovat zařízení, které není v pohybu (např. je položeno na stole či v kapse sedícího uživatele) není problém, protože jeho hodnota zrychlení bude rovna gravitačnímu zrychlení Země (-9.81 m/s^{-2}). Detekovat různé aktivity typu chůze, běh, řízení automobilu nebo jízda na kole, je možné zkoumáním vzorů dat,



Obrázek 12: Získání zrychlení zařízení [12]

získaných při těchto aktivitách, přičemž tyto hodnoty se budou samozřejmě minimálně lišit zařízení od zařízení a člověka[8].

Google nabízí pro klasifikaci těchto aktivit ActivityRecognitionApi. Pomocí funkce DetectedActivity.getType() jsme schopni získat informaci o tom, zda je zařízení ve vozidle, na jízdním kole a dále chůzi, běh či je zařízení v klidu. V mé implementaci aplikace pak využívám senzoru pohybu k detekci a klasifikaci uživatelem zadaných činností, jako je chůze, běh a kancelářská (sedavá) činnost. Po detekci takové aktivity je spuštěno měření času na dané aktivitě.

3.1.4 NFC

Díky technologii NFC (Near field communication) jsme schopni bezkontaktně komunikovat s objekty podporujícími tuto technologii a také s ostatními telefony v módu peer-to-peer. V rámci této práce popíši NFC tagy, protože se dají použít pro automatizaci a klasifikaci činností. NFC tagy jsou pasivní nálepky či podobné drobné objekty, které nevyžadují žádné přímé napájení. Jsou totiž napájeny přímo telefonem díky elektromagnetické indukce a stačí jim poměrně málo energie. Tagy s skládají obvykle ze dvou částí - antény a čipu. Pomocí antény probíhá přenos energie a dat pomocí krátkých rádiových vln na frekvenci 13.56 MHz a čip pak mimo jiné obsluhuje paměť tagu, kde mohou být uloženy informace pro použití například jako vizitek, jízdenek, občanských průkazů a pasů, docházkových karet a podobně.

Jak již bylo zmíněno v rešerši, například aplikace Jiffy - Time tracker těchto tagů využívá. Uživatel si může na jejich stránkách zakoupit tyto tagy ve formě přívěšků a podložek. V programu

pak označí jednotlivé projekty či aktivity k daným NFC tagům a při přiložení telefonu k tagu je možné spustit měření času na aktivitě.

Obrázek 13: JiffyTags [13]

Nevýhodou tohoto řešení automatizace je nutnost fyzicky vlastnit takovéto tagy a také fakt, že se nejedná o plně automatizované řešení. V dnešní době je také poměrně slabé zastoupení telefonů podporujících technologii NFC.

3.1.5 Ostatní senzory - senzor okolního světla a další

Systém Android a obecně mobilní zařízení dnes nabízí i spoustu dalších senzorů. Přístup k nim je možný přes službu SensorManager, která je součástí Android Sensor API. Právě zde je možné zjistit veškeré informace o dostupných senzorech na zařízení, jejich rozsahy a minimální časy aktualizace dat.

Jedním z nich je například také světelný senzor. Ten je obvykle umístěn na přední straně zařízení a je využíván systémem pro regulaci jasu displeje. Nicméně k těmto datům máme jako programátoři také přístup. V telefonu je tedy umístěna fotodioda a díky SensorManageru dostupném v systému Android jsme schopni získat její hodnoty v jednotkách lux. Typický rozsah těchto hodnot je 1 - 30 000 lux, reálné, senozorem získané hodnoty se však budou lišit dle zařízení.

Dalším senzorem je například barometr, který je ale podporován pouze několika zařízeními na trhu. Jeho hlavním účelem je určení nadmořské výšky tam, kde není možné využít technologie GPS (např. uvnitř budov) nebo pro urychlení získání polohy pomocí GPS. Získané hodnoty hodnoty ze zařízení se pak pohybují v rozsahu 300 - 1 100 mbar.

V neposlední řadě může zařízení disponovat také senzorem příblížení, tzv. proximity senzor. Podobně jako světelný senzor bývá umístěn na přední straně telefonu a je využíván systémem při telefonních hovorech. Pokud při právě probíhajícím hovoru přiblížíme telefon k uchu, dojde k automatickému uzamčení displeje pro zamezení nechtěného kontaktu ucha s ovládáním telefonu. Vzhledem k tomu použití je rozsah tohotu senzoru limitován na 2-3 cm, za tuto vzdálenost pak senzor hlásí maximální vzdálenost. V systému Android je pak tato vzálenost programátorovi dostupná v centimetrech.

4 Vlastní návrh aplikace

V předchozích kapitolách jsme provedli průzkum dostupných mobilních aplikací se zaměřením na time-tracking a popsali jsme senzory mobilních zařízení a jejich možnosti. Dále následuje návrh samotné aplikace včetně realizace jejích požadavků. Aplikace bude navržena tak, aby využila silných stránek zjištěných z rešerše trhu aplikací.

Mimo tyto požadavky bude aplikace implementována za použití nejnovějších metod pro vývoj pro operační systém Android. Důvodem pro tuto volbu je nejvyšší podíl na trhu mobilních aplikací a také nejvyšší počet potencionálních zařízení (uživatelů) aplikace. Mimo jiné je také vyvoj na tuto platformu zcela zdarma.

4.1 Požadavky na aplikaci

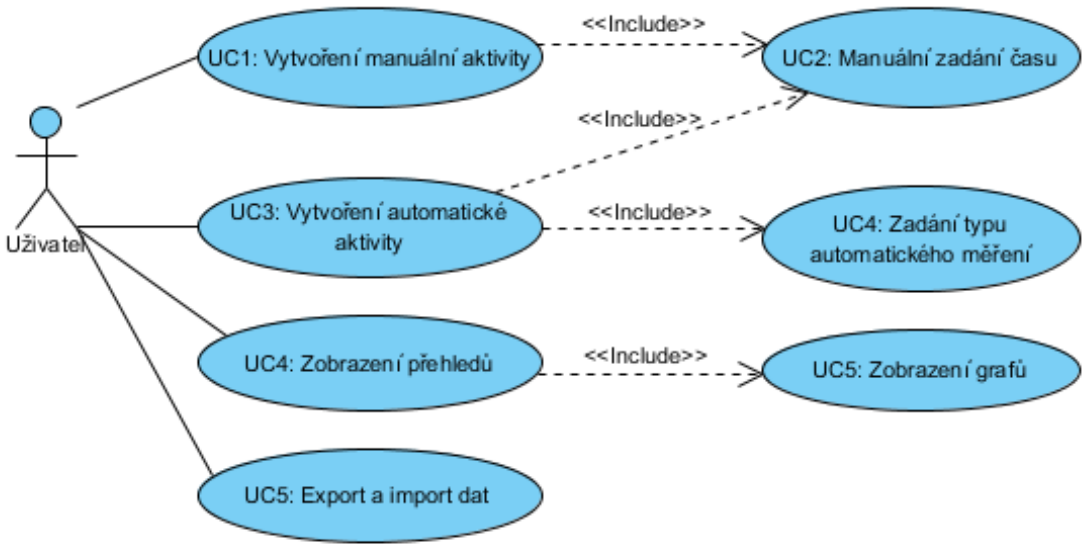
Dalším důležitým krokem při vývoji aplikace je ujasnit si požadavky, které na ni budou kladeny. Následující požadavky na aplikaci vyplynuly z rešerše a budou implementovány v samotné aplikaci. Způsob implementace je popsán v následující kapitole.

1. Možnost tvorby uživatelem zadaných aktivit
2. Možnost manuálního zadávání času k aktivitám
3. Automatizace zadávání času za pomocí senzorů zařízení za pomocí lokace (GPS, WiFi) či typu pohybu(chůze, běh, jízda na kole, jízda ve vozidle)
4. Možnost exportu a importu dat
5. Vizualizace naměřených hodnot - ve formě přehledů, grafů

Za další, neméně důležité požadavky můžeme označit snadné a intuitivní ovládání aplikace a její spolehlivost.

4.2 Use-case diagram

Abychom si přesně vymezili funkčnosti aplikace, je vhodné vytvořit diagram případů užití. Tento use-case diagram pro naši aplikaci AutoTime zjednodušeně popisuje jednotlivé situace, které budou splněny při implementaci této aplikace. Diagram má jediného aktéra - uživatele aplikace.



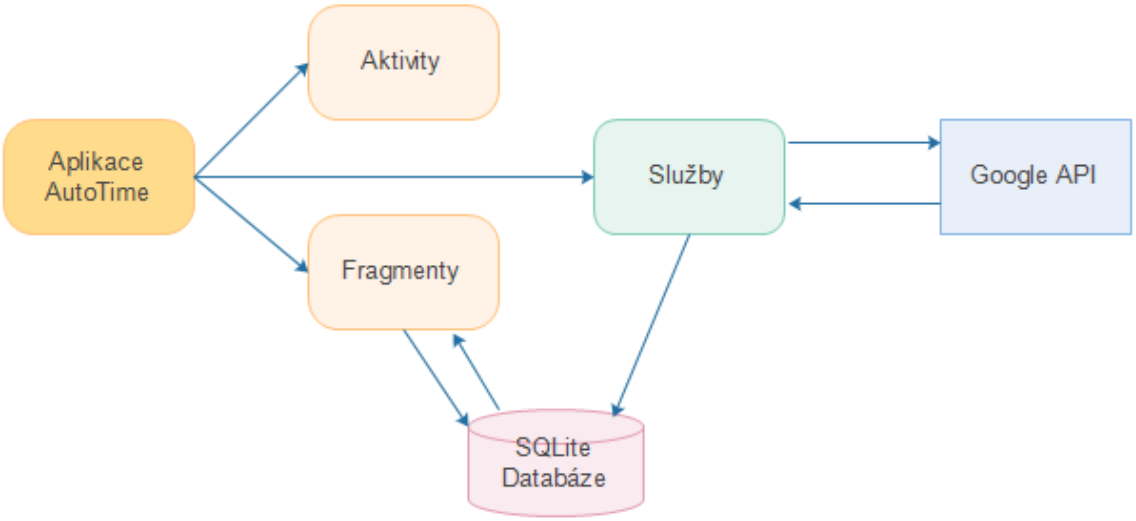
Obrázek 14: Use-case diagram aplikace AutoTime

5 Implementace aplikace

Aplikace je implementována na základě předchozího návrhu a splňuje všechny výše popsané požadavky a funkce. Hlavní scénáře práce s aplikací byly implementovány na základě use-case diagramů.

5.1 Popis stavby aplikace

Při popisu implementace můžeme aplikaci rozdělit a popsat její jednotlivé části. Aplikace se skládá z jedné hlavní aktivity, která přepíná fragmenty aplikace. Tato hlavní aktivita má také za úkol spustění služby geolokace a služby detekce a klasifikace pohybu za pomocí pohybového senzoru zařízení. Tyto služby jsou napojeny na Google API, které poskytuje vyhodnocení aktuální lokace a aktivit. Samotná aplikace a služby mají přístup k databázi, kde jsou všechny aktivity a záznamy uloženy a odkud je k nim přistupováno.



Obrázek 15: Diagram stavby aplikace

Jak již bylo zmíněno, třída *Autotime.class* je považována za hlavní třídu aplikace. Spouští a případně zastavuje služby běžící na pozadí a má za úkol vykreslení bočního navigačního menu aplikace. Prvky tohoto menu pak přepínají jednotlivé fragmenty a takto probíhá změna jednotlivých obrazovek aplikace.

Tyto fragmenty pak obvykle obsahují data, s nimiž pracují, ve formě *ArrayListu* naplněného danými objekty (*MyActivity.class*, *MyRecord.class*). Fragmenty obsahují jako atribut také pomocnou třídu *DBHelper.class*, která má za úkol vytvoření a správu databáze aplikace. Kdykoli fragment potřebuje načíst jednotlivé záznamy či aktivity z databáze, pak zavolá příslužnou metodu této třídy (např. *public ArrayList<MyActivity> getAllAutoActivities()*).

```
public ArrayList<MyActivity> getAllAutoActivities(){
   activities.clear();
   SQLiteDatabase db = this.getReadableDatabase();
   Cursor res = db.rawQuery( "select * from activity where type != ?", new
       String[] {"0"}, null );
   res.moveToFirst();

   while(res.isAfterLast() == false){
      activities.add(new MyActivity(Integer.parseInt(res.getString(res.
          getColumnIndex(ACTIVITY_COLUMN_ID))),
      res.getString(res.getColumnIndex(ACTIVITY_COLUMN_NAME)),
      res.getString(res.getColumnIndex(ACTIVITY_COLUMN_DESC)),
      res.getString(res.getColumnIndex(ACTIVITY_COLUMN_CLIENT)),
      Integer.parseInt(res.getString(res.getColumnIndex(ACTIVITY_COLUMN_PAY)
          )),
      Integer.parseInt(res.getString(res.getColumnIndex(ACTIVITY_COLUMN_TYPE
          ))),
      res.getString(res.getColumnIndex(ACTIVITY_COLUMN_LOCATION))));
      res.moveToNext();
   }
   return activities;
}
```

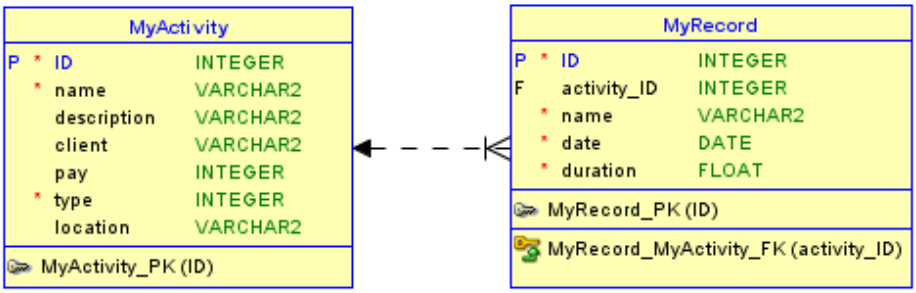
Výpis 1: Metoda pro získání automaticky měřených aktivit z třídy DBHelper.class

5.2 Databáze aplikace

Pro ukládání jednoduchých dat typu klíč-hodnota využívá aplikace *SharedPreferences* a to pro ukládání nastavené aplikace. Pro samotnou databázi využívám v Androidu dostupnou databázi SQLite, kde ukládám všechny uživatelem zadané aktivity a záznamy.

Tato jednoduchá databáze se skládá ze dvou tabulek - seznam aktivit a seznamu záznamů daných aktivit. Z toho také vyplývá jejich vazba 1:N.

Tabulky jsou ekvivalenty, co se vlastností týče, tříd *MyActivity.class* a *MyRecord.class* v aplikaci. Třídy obsahují potřebné get a set metody pro nastavování a získávání atributů daných objektů a dále metody *toString()* a *toCSVString()*, jež využívám pro export dat v aplikaci a logování.



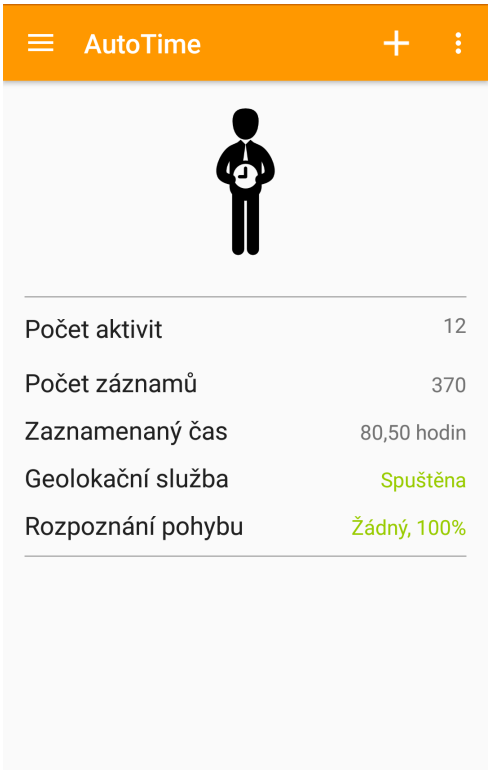
Obrázek 16: Databáze aplikace AutoTime

5.3 Úvodní obrazovka a orientace v aplikaci

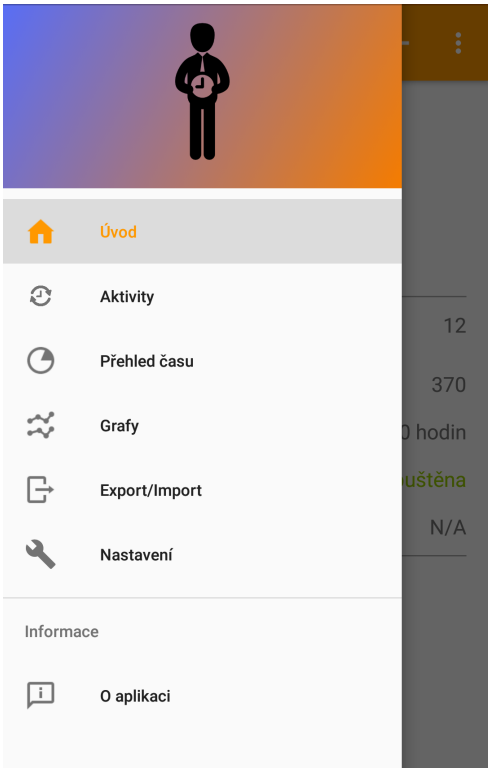
Po spuštění aplikace se objeví aktivita *Autotime.class*, na níž se zobrazují veškeré ostatní fragmenty aplikace. Úvodní fragment aplikace *FragmentHome.class* obsahuje rychlý přehled používání aplikace. Je zde možno vidět počet uživatelem přidaných aktivit, počet zadaných a získaných záznamů včetně jejich doby trvání a informace o právě spuštěných službách sledování lokace a pohybového senzoru a také real-time informace o aktuálně rozpoznaném pohybu.

Navigace v prostředí aplikace je realizována panelem na levé straně obrazovky. Tento prvek obsahuje odkazy, po jejichž stisknutí se přesuneme na další fragmenty aplikace. Je možné jej aktivovat gestem přejetí prstem od z levé hrany obrazovky k pravé či ikonou menu v horní části obrazovky. Zde je také možnost rychlého přístupu k manuálnímu přidání časového záznamu k aktivitě, funkce vypnutí geolokační služby a vymazání databáze aplikace.

Úvodní obrazovku aplikace a navigační panel je možné vidět na obrázcích níže.



Obrázek 17: Aplikace AutoTime - Úvodní obrazovka



Obrázek 18: Aplikace AutoTime - Menu aplikace

5.4 Geolokační služba

Služba pro získávání aktuální lokace zařízení je realizována pomocí třídy *Service*. Tato služba je spuštěna hlavní aktivitou aplikace a je ji možné v případě potřeby zastavit či restartovat. Aby mohla aplikace sledovat geolokaci zařízení, je nutné aby zařízení disponovalo technologií GPS (nejvyšší přesnost), případně WiFi nebo přístupu k síti GSM. Dále musí být v zařízení přítomna klientská knihovna Google Play, pokud možno v co nejaktuálnější verzi.

O přístup k těmto technologiím pak musíme taky zažádat v AndroidManifest.xml souboru aplikace.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

Výpis 2: AndroidManifest.xml a přístup k geolokaci

Dále je zapotřebí inicializace připojení k Google API a jeho LocationService. Toto spojení nám umožní přihlásit se k aktualizacím polohy zařízení v určitém námi zadaném intervalu. Výsledkem je pak objekt *Location*, který obsahuje získané informace o poloze - zeměpisnou šířku, délku, nadmořskou výšku, přesnost a další údaje, se kterými můžeme dále pracovat.

```
mGoogleApiClient = new GoogleApiClient.Builder(this)
.addConnectionCallbacks(this)
.addOnConnectionFailedListener(this)
.addApi(LocationServices.API)
.build();
mGoogleApiClient.connect();
```

Výpis 3: Připojení ke Google API a LocationService

5.5 Služba pohybového senzoru

Stejně jako u geolokace, i pro detekci a klasifikaci pohybu pomocí pohybového senzoru zařízení potřebujeme připojení ke Google API, konktrétně pak k *ActivityRecognition.API*. Tuto službu jsem realizoval pomocí *IntentService*, což je typ služby, který zasílá získané informace aktivitě, která jej spustila za pomocí akcí *Intent*. Pro možnost získávání těchto informací ze senzoru telefonu je nutné také upravit *AndroidManifest.xml* soubor. I zde je možné nastavit interval odebírání aktualizací pohybu, přičemž při mém testování jsem zjistil, že nejnižší možný interval je 5 vteřin.

```
<uses-permission android:name="com.google.android.gms.permission.
    ACTIVITY_RECOGNITION" />
```

Výpis 4: AndroidManifest.xml a přístup k pohybovému senzoru

Pro získání seznamu možností výsledných aktivit musíme extrahovat *ActivityRecognitionResult* z *Intent*. Do objektu *DetectedActivity* pak uložíme z tohoto seznamu nejpravděpodobnější aktivitu. Tento objekt pak obsahuje typ této aktivity a confidence level (důvěru), která se značí od 0 až 100 %. Google aktuálně umožňuje detekci těchto aktivit: chůze, běh, jízda na kole, jízda ve vozidle, rotace zařízení a žadný pohyb (stacionární).

```
ActivityRecognitionResult result = ActivityRecognitionResult.extractResult(
    intent);
DetectedActivity detectedActivity = result.getMostProbableActivity();

int confidence = detectedActivity.getConfidence();
int activityType = detectedActivity.getType();

if (activityType == DetectedActivity.ON_FOOT){
...
}
```

Výpis 5: Získání a detekce pohybu

Na základě takto detekovaného pohybu jsme pak schopni tuto informaci zpracovat a v případě vysoké vyhodnocené důvěry v danou aktivitu záznam uložit do databáze.

5.5.1 ActivityRecognitionApi

Toto API poskytované společností Google pro operační systém Android poskytuje možnost odběru a detekce aktivit získaných ze senzorů mobilního zařízení - akcelerometru a gyroskopu. Pro jeho podporu je nutné vlastnit zařízení s OS Android verze 5.0 a výše (API 21+).

Pomocí metody *requestActivityUpdates(GoogleApiClient client, long detectionIntervalMillis, PendingIntent callbackIntent)* se umožňuje přihlásit k detekci těchto aktivit. Aktivity jsou detekovány periodicky tak, že probudí zařízení a krátce přečte data získaná ze senzorů, přičemž se snaží o minimální spotřebu baterie. Tuto spotřebu ovlivňuje také interval aktualizace těchto dat. Při nižších hodnotách aktualizace je zařízení probouzeno častěji, což zvyšuje spotřebu baterie zařízení. Ideální hodnotu lze získat testováním, o kterém je možno se dočíst dále.

Po přihlášení k těmto aktulizacím získáváme objekty *ActivityRecognitionResult*, jež obsahují seznam *DetectedActivity*. Každá takto detekovaná aktivita s sebou nese typ aktivity (např. *DetectedActivity.IN_VEHICLE*) a confidence level (důvěru) v danou aktivitu, což je hodnota od 0 až 100, přičemž 100 je maximální možná důvěra v aktivitu. Součet důvěry pravděpodobnosti v dané aktivity v tomto seznamu aktivit nemusí být roven stu, protože některé aktivity nemusí být vzájemně exkluzivní - je možná například chůze v autobusu. Je tedy vhodné zpracovávat jednu či více nejpravděpodobnějších aktivit zároveň.

5.6 Přehledy a grafy

Po buď manuálním, či automatickém sběru záznamů pro zadané aktivity bylo nutné tato data vhodně prezentovat. Z rešerše jsme zjistili, že všechny aplikace disponují přehledy aktivit a jejich záznamů a některé z nich tato data prezentují graficky do grafů.

Grafy v Androidu lze realizovat pomocí již existujících knihoven, případně si vytvořit knihovnu vlastní. Tyto knihovny lze rozdělit například na open-source, či placené. Následuje seznam knihoven, které byly brány v potaz a splňovaly níže uvedené požadavky:

- MPAndroidChart
- aiCharts (placené)
- HoloGraphLibrary
- HelloCharts for Android
- WilliamChart

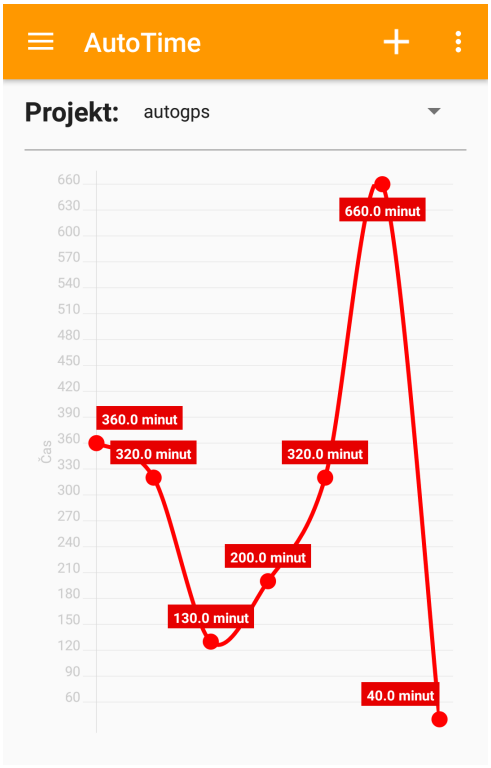
Tyto knihovny se od sebe většinou hodně lišily. Mým cílem bylo vybrat knihovnu, která je aktuální (pro Android 5 a výše), aby umožňovala interakci s grafy a aby podporovala spojnicové a výsečové grafy.

Jako knihovnu pro vykreslování grafů jsem zvolil Hellocharts pro svou barevnost, jednoduchost a pokročilou možnost interakce s grafem. Mimo mnou použitých typů grafů v aplikaci disponuje Hellocharts i další škálou grafů, umoňuje přiblížení a oddálení grafu a animace. Jediná podmínka pro jeho použití je Android 4.0 (API 14+) kvůli použití harwarové akcelerace.

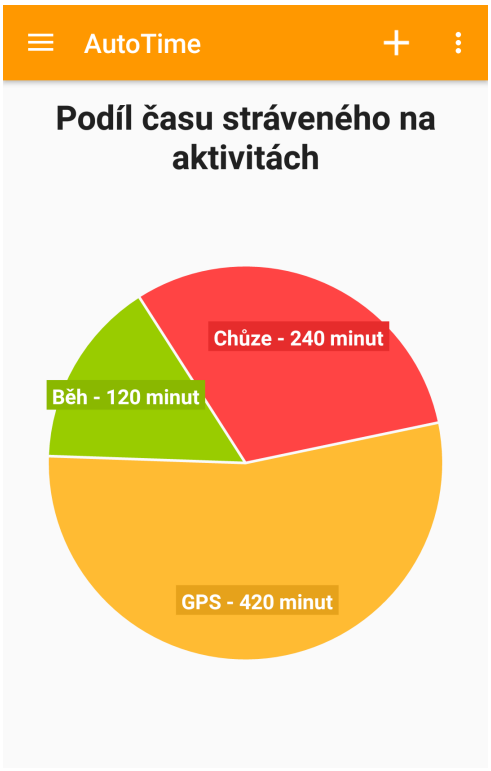
Data získaná z databáze bylo nutné zpracovat pro vykreslení v grafu. Zde jsem využil datové struktury *Map* pro vytvoření struktury aktivita-čas pro výsečový graf a struktury den-čas pro spojnicový graf v aplikaci. Poté již stačilo dané grafy naplnit zpracovanými hodnotami a grafy vykreslit.

```
LineChartView chart = new LineChartView(context);
layout.addView(chart);

List<PointValue> values = new ArrayList<PointValue>();
values.add(new PointValue(0, 2));
values.add(new PointValue(1, 4));
Line line = new Line(values).setColor(Color.RED).setCubic(true);
List<Line> lines = new ArrayList<Line>();
lines.add(line);

LineChartData data = new LineChartData();
data.setLines(lines);
LineChartView chart = new LineChartView(context);
chart.setLineChartData(data);
```

Výpis 6: Vytvoření jednoduchého spojnicového grafu pomocí knihovny Hellocharts



Obrázek 19: Aplikace AutoTime - Spojnicový graf pro vybranou aktivitu



Obrázek 20: Aplikace AutoTime - Výsečový graf pro přehled všech aktivit

6 Testování navrženého řešení

Jakmile byla implementace aplikace hotova, následovalo její testování. Malá skupina testerů testovala stabilitu, přesnost a optimální nastavení měřících parametrů aplikace na více zařízeních. Řada dalších testů byla provedena také autorem této práce. Výsledky tohoto testování jsou uvedeny v této kapitole.

6.1 Testovaná zařízení

Finální verze aplikace byla testována na více fyzických i emulovaných zařízeních. Minimální požadavky na verzi operačního systému Android je 5.0 (API 21) s tím, že zařízení či emulátor musí mít k dispozici přístup ke Google API (Google Play Store a další aplikace).

Co se rozlišení displeje týče, aplikace byla testovanána na rozlišeních HD (1280x720), FHD (1920x1080) a vyšších. Na menších než testovaných rozlišeních je možné, že některé prvky GUI nebudou vykresleny správně.

Seznam testovaných fyzických zařízení:

- Google Nexus 5 (Android 6.0, Android 6.0.1)
- Google Nexus 7 (Android 6.0.1)
- LG G3 (Android 5.0, Android 6.0)
- Asus ME173X (Android 5.0)

Aplikace byla také testována na softwarovém emulátoru systému Android - pomocí emulátoru dostupného v Android SDK a Genymotion emulátoru. Co se grafické stránky týče, nebyly nalezeny žádné problémy. Nutností je však přítomnost Google API v systémovém obrazu a povolena geolokace v systému.

6.2 Testování přesnosti detekce aktivit

Jak již bylo zmíněno, aplikace byla testována na více zařízeních v průběhu několika dnů mimo jiné za účelem testování přesnosti automatické detekce aktivit pomocí geolokace a pohybového senzoru.

Data získaná z aplikace byla porovnávána se skutečně prováděnými aktivitami v daných časech ze všech testovacích vzorků a zařízeních. Následuje tabulka těchto dat z třídenního testování aplikace na zařízení Google Nexus 5 (Android 6.0.1) v terénu.

	Skutečný čas (minut)	Čas naměřený aplikací (minut)
Chůze	344	292
Běh	92	69
Jízda na kole	117	98
Jízda ve vozidle	272	241
GPS a geolokace	2880	2610

Tabulka 2: Porovnání dat získaných z aplikace se skutečností

Vzorek naměřených dat aplikací je přiložen v příloze této práce pro případný import do aplikace. Tento a další vzorky testování sloužily pro optimalizaci hranice "důvěry"v danou detekovanou aktivitu pohybovým senzorem. Jako optimální hranice tohoto nastavení detekce ve výsledné implementaci se ukázalo 60 %, což znamená, že pokud Google API vyhodnotí aktivitu s touto nebo vyšší důvěrou v aktivitu, záznam je zaznamenán do databáze aplikace, jinak je zahozen. S nižší hranicí se často vystytovaly nesprávně detekované aktivity, naopak při nastavení vyšší hranice byla detekce aktivit méně častá. Všechna testovaná fyzická zařízení vykazovala podobnou přesnost, neobjevovaly se zde žádné velké odchylky.

Co se jednotlivých aktivit týče, nejpřesněji pohybem detekovaná aktivita během testování byla jízda ve vozidle a detekce chůze, což může být ale pouze důsledkem největšího počtu takto detekovaných testovacích dat. Způsob vyhodnocování aktivit pomocí pohybového senzoru v Google API není bohužel veřejně znám, a proto je nutné nalézt ideální nastavení samostatně.

U automatických aktivit s geolokací většinou nebyl problém, záznamy byly ukládány do databáze aplikace v pětiminutových intervalech, pokud bylo zařízení v méně než stometrovém dosahu od uživatelem zadaného bodu při vytváření aktivity. Při dostupnosti GPS či WiFi signálu v daném místě byla detekce velmi přesná.

Z testování také vyplynulo, že detekce pohybových aktivit je méně přesná, když je daná aktivita i na malou chvíli přerušena (například náhlé zastavení při chůzi, běhu), což lze ale částečně kompenzovat nastavením nižšího intervalu aktulizace detekce pohybu. Následuje testování za účelem získání ideálního hodnoty tohoto atributu.

6.3 Interval aktualizace polohy a pohybu

Interval aktualizace polohy zařízení byl po testování nastaven na 5 minut, což se zdálo jako dobrý kompromis mezi přesností detekce a úspory baterie kvůli častému přístupu dat z technologií GPS a WiFi.

U detekce aktivit pomocí pohybového senzoru proběhlo rozšířenější testování za účelem získání ideální hodnoty aktualizace dat ze zařízení. Byly testovány hodnoty intervalu aktualizace co 5, 15, 30, 60 a 120 vteřin. Nejnižší možná hodnota byla zjištěna právě pět vteřin, kdy zařízení snímalo v tomto intervalu data získaná ze senzorů a vyhodnocovala dané aktivity. Toto nastavení bylo ideální pro zaznamenání změny aktivit či náhlého přerušení aktivity, ale na druhou stranu zde byla zjištěna největší spotřeba baterie a poměrně hodně nesprávně vyhodnocených

aktivit. Vzhledem k těmto faktům je ve výsledné implementaci nastavena tato hodnota na jednu minutu, což se poté osvědčilo i při testování na více zařízeních mezi testery.

7 Závěr

Cílem této bakalářské práce bylo provést návrh a následně vlastní implementaci aplikace určené pro time-management s automatizací pro zařízení s mobilním operačním systémem Android.

Zpočátku byl proveden průzkum současného stavu trhu aplikací s touto tématikou na všech moderních mobilních platformách. Poté byly prozkoumány všechny možnosti automatizace detekce aktivit, které moderní mobilní zařízení nabízí. Tyto možnosti mimo jiné zahrnovaly automatizaci v rámci geolokace za pomocí GPS, WiFi, sítě GSM a také pomocí dat získaných z akcelometru a gyroskopu zařízení.

S poznatky získaných z rešerše byla následně implementována aplikace AutoTime s vlastnostmi, které se zdály jako nejdůležitější pro tento typ aplikace a které vyplynuly z analýzy. Oproti svým konkurentům zde byla použita určitá míra automatizace pomocí technologií a senzorů dostupných na mobilních zařízeních.

Po implementaci byla aplikace dále testována skupinou testerů za účelem zjištění spolehlivosti automatizace, optimalizaci měřících parametrů a objevení chyb vzniklých při implementaci.

Tato práce mi rozšířila pohled na svět vývoje aplikací pro operační systém Android, díky níž jsem získal představu, co vše obnáší a co je potřeba k vytvoření aplikace pro tuto mobilní platformu. Mým dalším cílem je aplikaci publikovat na katalogu Google Play a postupně ji rozšiřovat a optimalizovat pro to, aby se její používanost dále rozrostla.

Petr Macák

Literatura

[1] MEIER, Reto. Professional Android. 4. vydání. Wrox, 2015. ISBN 978-1118949528.

[2] ANNUZZI JR., Joseph. Advanced Android Application Development. 4. vydání. Addison-Wesley Professional, 2014. ISBN 978-0133892383.

[3] MILETTE, Greg. Professional Android Sensor Programming. 1. vydání. Wrox, 2012. ISBN 978-1118183489.

[4] HELLMAN, Erik. Android Programming: Pushing the Limits. 1. vydání. Wiley, 2013. ISBN 978-1118717370.

[5] VAVRU, Jiri, Miroslav UJBANYAI a Miroslav KAJAN. Android programming: Complete application programming guide [online]. 2. vydání. Grada Publishing a.s., 2014 [cit. 2016-04-08].

[6] CINAR, Onur. Android Quick APIs Reference. 1. vydání. Apress, 2015. ISBN 978-1-484205-24-2.

[7] KWAPISZ, Jennifer R.; WEISS, Gary M.; MOORE, Samuel A. Activity recognition using cell phone accelerometers. ACM SigKDD Explorations Newsletter, 2011, 12.2: 74-82.

[8] YANG, Jun. Toward physical activity diary: motion recognition using simple acceleration features with mobile phones. In: Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics. ACM, 2009. p. 1-10.

[9] YAN, Zhixian, et al. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In: Wearable Computers (ISWC), 2012 16th International Symposium on. Ieee, 2012. p. 17-24.

[10] DERNBACH, Stefan, et al. Simple and complex activity recognition through smart phones. In: Intelligent Environments (IE), 2012 8th International Conference on. IEEE, 2012. p. 214-221.

[11] Smartphone OS Market Share, 2015 Q2. IDC.com [online]. 2015 [cit. 2016-03-05]. Dostupné z: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

[12] Motion Sensors. Android Developers [online]. 2016 [cit. 2016-01-11]. Dostupné z: http://developer.android.com/guide/topics/sensors/sensors_motion.html

[13] JiffyTags. Jiffy [online]. 2015 [cit. 2016-01-11]. Dostupné z: http://jiffy.nu/jiffytags/

A Obsah CD

K práci přiložené CD obsahuje:

1. Text bakalářské práce
   - MAC0345.pdf
2. Projekt se zdrojovými kódy aplikace
   - AutoTime
3. APK aplikace
   - AutoTime.apk
4. Excel dokument popisující tabulku výzkumu
   - pruzkum.xlsx