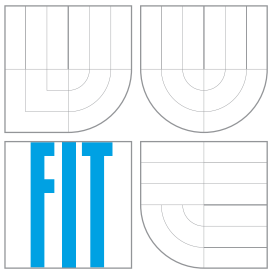
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MOBILNÍ APLIKACE PRO SKENOVÁNÍ A ROZPOZNÁVÁNÍ POKLADNÍCH ÚČTENEK

MOBILE APP FOR SCANNING AND RECOGNITION OF CASHIER BILLS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VLADISLAV BAMBUCH

VEDOUCÍ PRÁCE
SUPERVISOR

prof. Ing. ADAM HEROUT, Ph.D.

BRNO 2016

Zadání bakalářské práce/17983/2015/xbambu03

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Bambuch Vladislav**
Obor: Informační technologie
Téma: **Mobilní aplikace pro skenování a rozpoznávání pokladních účtenek**
Mobile App for Scanning and Recognition of Cashier Bills
Kategorie: Uživatelská rozhraní

Pokyny:

1. Prozkoumejte a popište existující nástroje pro sledování osobních financí.
2. Prozkoumejte a popište existující mobilní aplikace pro skenování a rozpoznání textu.
3. Prostudujte a popište algoritmy zpracování obrazu a rozpoznávání textu relevantní pro tento projekt.
4. Získejte a/nebo vyviňte potřebné algoritmy pro skenování účtenek a rozpoznávání textu na nich.
5. Navrhněte uživatelské rozhraní aplikace pro skenování účtenek a rozpoznání jejich textu, případně pro základní správu osobních financí.
6. Implementujte navrženou aplikaci.
7. Proveďte testování a vylepšování vytvořeného uživatelského rozhraní.
8. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3,
- značné rozpracování bodů 4 až 6.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Herout Adam, doc. Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Cílem práce je vytvořit mobilní aplikaci umožňující focení pokladních účtenek, následně pak správu osobních financí pomocí přehledného grafického uživatelského rozhraní. Aplikace využívá technologie OCR pro digitalizaci cen z účtenek a řídí se novými trendy Android Material Design. Na vyfocené účtence dochází k označování cen jednotlivých položek, které se po kliknutí mezi sebou sčítají do výsledné sumy. Tu pak, společně s dalšími informacemi, může uživatel uložit a vést si statistiky o svých výdajích. Výsledkem je rychlé a velmi snadné přidávání záznamů o nákupech, které jsou k dispozici v přehledných výpisech i ve formě grafů.

Abstract

The aim of this thesis is to develop a mobile application which allows photographing of cashier bills and offers a function of finance management via an intuitive graphical user interface. The app uses OCR technology to digitize the prices of receipts and follows the new Android Material Design trends. The price marking of the particular items is carried out on the taken receipt. When the user indicates particular prices by clicking on them they are summed up into the final amount. Users are able to save such final amounts together with other related information and record statistics about their spending. That results in a fast and very easy way of adding the purchase records which are available both in form of organized summaries and graphs.

Klíčová slova

správa výdajů, skenování účtenek, účetnictví v telefonu, jednoduché uživatelské rozhraní, třídění účtenek, Android OCR, výdaje v telefonu, vím za co platím, domácí účetnictví, BillScanner

Keywords

expense management, expense tracker, receipt scanner, accountancy in phone, simple user interface, receipt mangement, Android OCR, spending on the phone, money tracker, bill scanner, BillScanner

Citace

BAMBUCH, Vladislav. *Mobilní aplikace pro skenování a rozpoznávání pokladních účtenek.* Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Herout Adam.

Mobilní aplikace pro skenování a rozpoznávání pokladních účtenek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana profesora Adama Herouta. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Vladislav Bambuch
16. května 2016

Poděkování

Děkuji vedoucímu práce, panu profesoru Adamu Heroutovi, za předání cenných informací v rámci pravidelných konzultacích a za jeho perfektní přístup.

Obsah

Kapitola 1

Úvod

Tato práce popisuje tvorbu mobilní aplikace *BillScanner* pro operační systém Android, analýzu konkurenčních řešení v oblasti aplikací pro správu osobních financí i aplikací na skenování dokumentů, popřípadě přímo rozpoznávání textu z fotografie.

Důvodem tvorby práce jsou nedostatky, které mají existující řešení umístěné na Google Play. Za hlavní problém, se kterým se autor práce setkal při používání několika aplikací, je označeno přidávání nových záznamů o výdajích. Podrobnou analýzou konkurenčních produktů se zabývá kapitola 2.2.

Uživatelé mobilních zařízení dnes mají jedinečnou možnost si prostřednictvím mobilních aplikací vést záznamy o uskutečněných nákupech a statistiky o celkové finanční bilanci. Mnoho z nich však tuto možnost nevyužívá právě z důvodu komplikovaného manuálního přidávání transakcí. Práce si tedy klade za cíl hlavně zjednodušení přidávání nových záznamů použitím *OCR*[1], a tím se snaží o zefektivnění samotné správy osobních financí.

V práci jsou vysvětleny základní principy tvorby moderních uživatelských rozhraní s využitím *Android Material Design*. Dále je zde popsána nejsložitější část práce, kterou je rozpoznávání textu z fotografie, a použití nezbytně nutných technologií pro předzpracování obrázku.

V neposlední řadě je nastíněna problematika testování vytvořené aplikace — jak v rámci alfa verze pro vybraný okruh testerů, tak veřejné beta verze.

[1] Optical Character Recognition – Optické rozpoznávání znaků

Kapitola 2

Motivace pro zavedení domácího účetnictví

Jedním z problémů dnešního rychlého životního stylu je, že si mnoho lidí neuvědomuje, kolik peněz měsíčně utrácí, a za co přesně [30]. Bezkontaktní platební styky a placení přes internet, trvající jen pár sekund, a k tomu ještě s virtuální měnou, kterou fyzicky neměli nikdy v rukách. Toto má sice nesporné výhody, ale lidé pak ztrácejí kontakt se svými penězi, a to přispívá k nevědomosti o své celkové finanční situaci.

Snadno se takovému člověku může stát, že si z nouze začne peníze půjčovat, ale to jeho problém nevyřeší. Přichází jen další dluhy následované novými půjčkami. Velké množství lidí si totiž nespoří na horší časy [28].

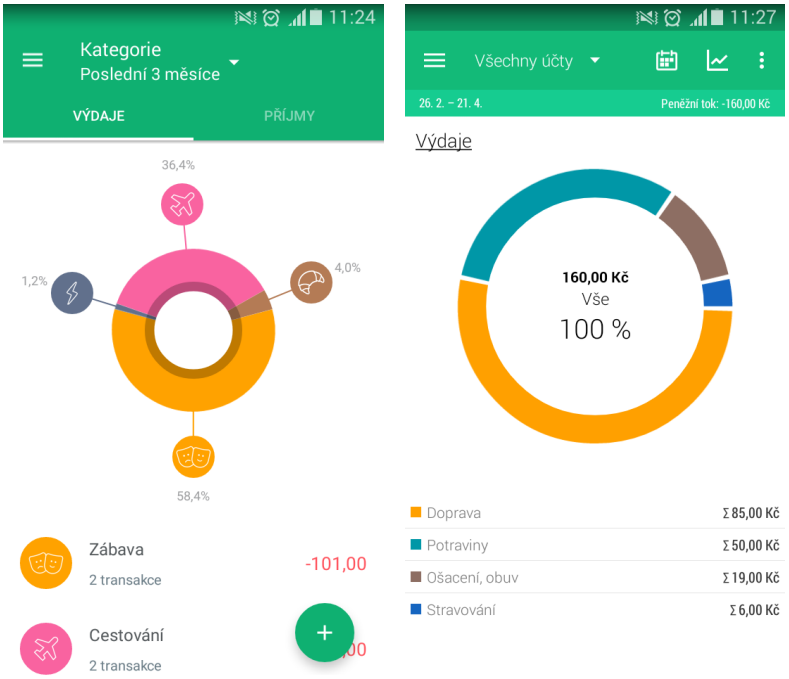
Naučit se finanční gramotnosti a odpovědnosti jde vícero způsoby [29]. Jednou takovou pomůckou mohou být například mobilní aplikace, pomocí kterých si uživatel vede záznamy o svých pravidelných výdajích. S odstupem času se může podívat na přehled svých měsíčních výdajů a zjistit, jak na tom je po finanční stránce. Taková aplikace by mohla být i vhodným doplňkem „*vzdělávacích programů zaměřených na děti, mládež a pedagogy*“ [29].

Chytrá mobilní zařízení jsou dnes velmi rozšířena, a jejich použití pro sledování osobních výdajů je tím pádem vhodnou variantou, jak vylepšit povědomí o problematice finanční gramotnosti. Správně navržená aplikace se snadným ovládáním by tak mohla motivovat uživatele k většímu zájmu o tuto problematiku.

2.1 Třídění účtenek podle kategorií

Vědomí kolik celkově utratím za určité časové období je dobrý začátek, ale není to zdaleka vše. Důležité je vědět, za co konkrétně člověk utratí, a kde by tak mohl potenciálně ušetřit.

Řešením je organizovat své výdaje do kategorií. Ty lze například porovnávat v přehledných grafech, jako je vidět u aplikací *Spendee* a *Wallet* na obrázku 2.1. Každý uživatel má také jiné požadavky na množství a názvy kategorií. Je tedy vhodné, aby aplikace umožňovala přidávání vlastních kategorií.



Obrázek 2.1: Obrázek ukazuje dva příklady, jak můžou být tříděny kategorie v mobilních aplikacích. V levé části obrázku je možné vidět Spendee, vyznačující se povedeným rozhraním i pěknými grafy. Na pravé straně je ukázka z aplikace Wallet, která již tak promyšlenými grafy nedisponuje, a ani legenda nevypadá příliš přehledně.

2.2 Porovnání existujících řešení

Aplikací na správu osobních financí je na Google Play opravdu hodně. To podporuje zdravou rivalitu mezi vývojáři, a tím se i zlepšuje kvalita samotných aplikací. I přes to se dají u konkurenčních řešení najít podobná slabá místa, která poskytují prostor pro vylepšení.

O zajímavých řešeních a výše zmíněných slabinách je psáno ve zbývající části kapitoly, kde jsou rozebrány konkrétní produkty. Zejména je porovnáváno *GUI*[1], možnost OCR po vyfocení účtenky a složitost přidání nového záznamu.

2.2.1 Wallet

Aplikace [23] má pěkné provedení GUI dle standardu Material Design (více v kapitole 3.1), ale je příliš složité a obsahuje hodně informací, které se dají uživateli předat lepším způsobem. Například záznamy lze slučovat po dnech a nepsat datum ke každé položce výpisu — jak je vidět na obrázku 2.2a.

Přidání nového záznamu je zdlouhavé a vyžaduje častou uživatelovu interakci. Skládá se z několika kroků, přičemž je umožněno vyfocení a přiložení fotografie k záznamu. Tato fotografie ovšem slouží pouze jako příloha a né pro OCR. To aplikace neumí vůbec.

2.2.2 Monefy

Monefy [13] nabízí zajímavé a přehledné GUI. Vše se odehrává v jednom okně, což zlepšuje orientaci v aplikaci a budí dobrý dojem při používání. Po vytažení spodní části se úvodní graf překryje seznamem záznamů, což považuji za velmi dobrý nápad. Velká tlačítka přidávají na použitelnosti i ve spěchu, a to se u aplikace tohoto určení jistě hodí. Screenshot z aplikace ukazuje obrázek 2.2b.

Nový záznam o transakcích je možné přidat snadněji, než u předešlé aplikace, ale ani Monefy nedisponuje možností OCR.

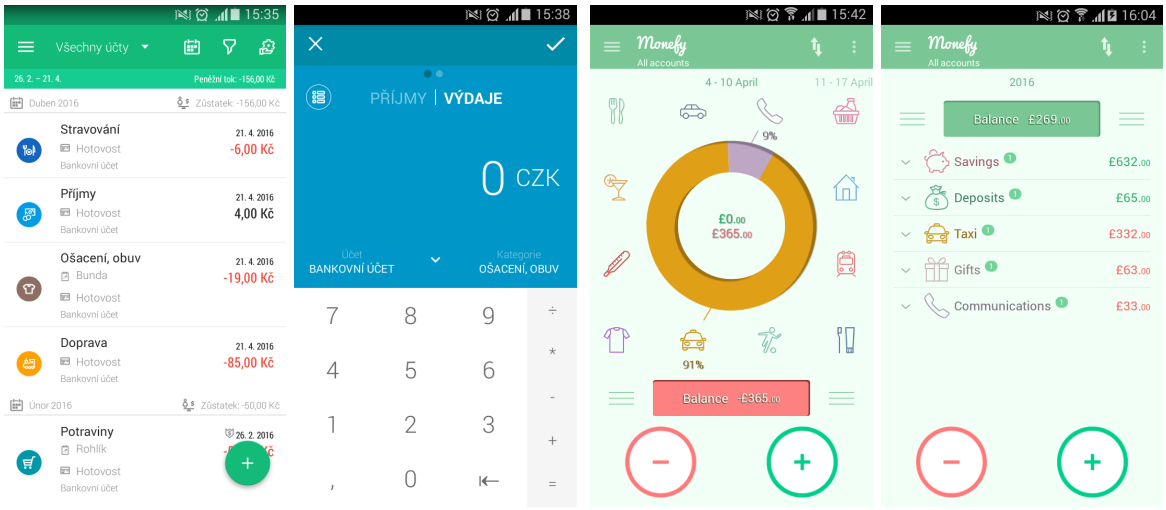
2.2.3 Expensify

Jediným zástupcem v této kapitole, který umí rozpoznávat text na účtenkách je aplikace Expensify [7]. Umožňuje standardně vyfocení nové fotky nebo načtení z galerie. Škoda je, že focení probíhá přes jinou aplikaci, která je v zařízení nastavena jako výchozí. Přechody mezi aplikacemi a jejich komunikace tak celý proces zpomalují. Výraznější prodlevu uživatel pocítí po vyfocení účtenky. Ta je posílána na server a výsledek skenování se uživatel dozví až za několik desítek minut (alespoň tedy pro české účtenky).

Uživatelské rozhraní je spíše navrženo pro používání na výletech nebo služebních cestách, než na sledování pravidelných výdajů. Tomu nasvědčují i další funkce aplikace, jako je sledování ujetých kilometrů nebo času stráveného na cestách.

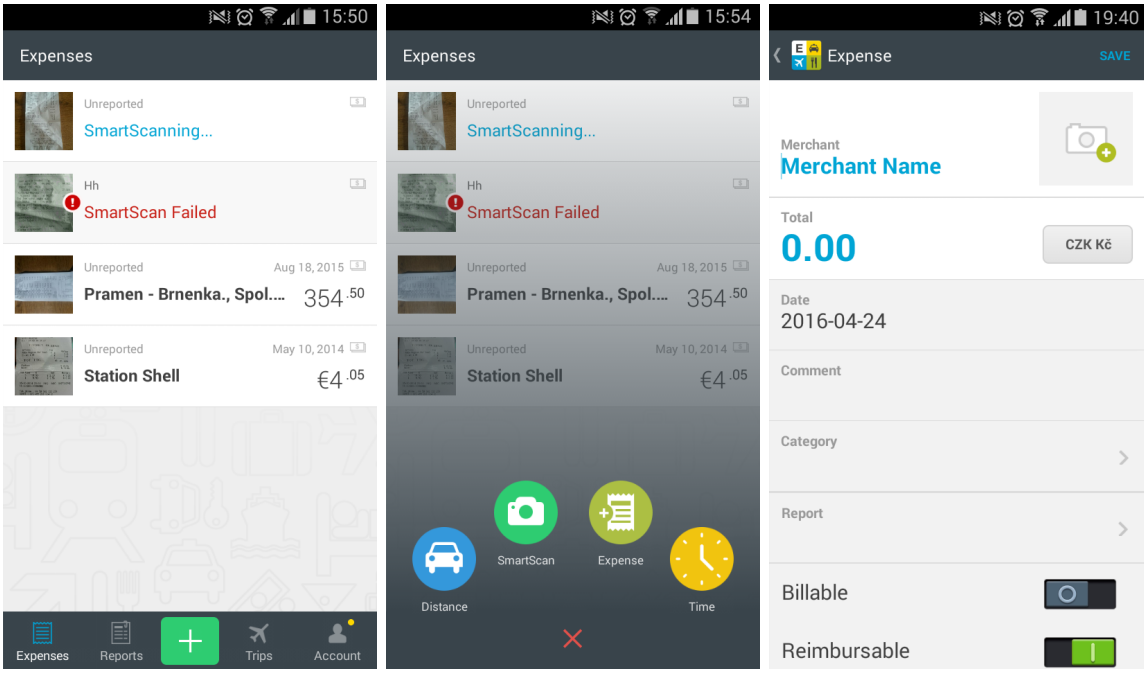
Aplikace je vhodná pro celkový přehled útraty, například za výlet. Už ne tak na třídění nákupů do kategorií. Samotné manuální přidávání záznamů je zdlouhavé a po grafické stránce ne příliš povedené.

[1] Graphical User Interface – Grafické uživatelské rozhraní



(a) Wallet (b) Monefy

Obrázek 2.2: V levé části obrázku je ukázka z aplikace Wallet, která disponuje pěkným, ale místy příliš složitým GUI. Na pravé straně je možné vidět aplikaci Monefy, obsahující netradiční, avšak přehledné uživatelské rozhraní.



Obrázek 2.3: Aplikace Expensify umí zpracovat data z účtenky pomocí OCR a použít je pro automatické přidání záznamu o výdaji. Vývojáři se Material Desingem příliš neřídí, což je vidět například na gradientu ve střední části obrázku. Pravá strana zase ukazuje nepřehledný formulář na manuální přidávání záznamů.

Kapitola 3

Tvorba moderních mobilních aplikací

Nynější trendy v oblasti návrhu mobilních aplikací se řídí novými poznatky *UX*[1]. Je potřeba, aby bylo sjednocené ovládání a design důležitých prvků. Uživatel se tak může spolehnout na to, že stejně vypadající ovládací prvky v jedné aplikaci mají stejný význam a chování i u konkurence. Kromě ovládacích prvků se sjednocují odstíny používaných barev, podoba formulářů a fontů, animace a také rozložení jednotlivých komponent GUI.

Vzniká tak *Flat Design*, který používá Apple, *Modern UI*, jak tomu říká Microsoft a Material Design společnosti Google.

3.1 Android Material Design

Material Design je standard nebo také designový jazyk, pomocí kterého se společnost Google snaží o sjednocení návrhu uživatelských rozhraní napříč všemi Android zařízeními, včetně těch s různými velikostmi displejů. Google tak definuje meze dobrého a špatného návrhu. Dále pak počítá s postupným vývojem a vylepšováním po celou dobu používání standardu [10].

Název *Material* má evokovat fakt, že se jedná o návrh, který se zakládá na zkušenostech a zvyklostech z reálného světa. Uživatel používá prvky rozhraní jako by se jednalo o realitu, protože se tak samy chovají. Důraz je kladen na animace tlačítek a přesunu prvků tak, aby byly zachovány základní pohybové fyzikální zákony. Navíc jsou použity jako prostředek pro udržení uživatelovy pozornosti v případech, kdy dochází k zobrazování/schovávání celých elementů do/mimo viditelnou oblast displeje [11]. Uživatel tak přesně ví, kam daný element zmizel a odkud má očekávat jeho návrat.

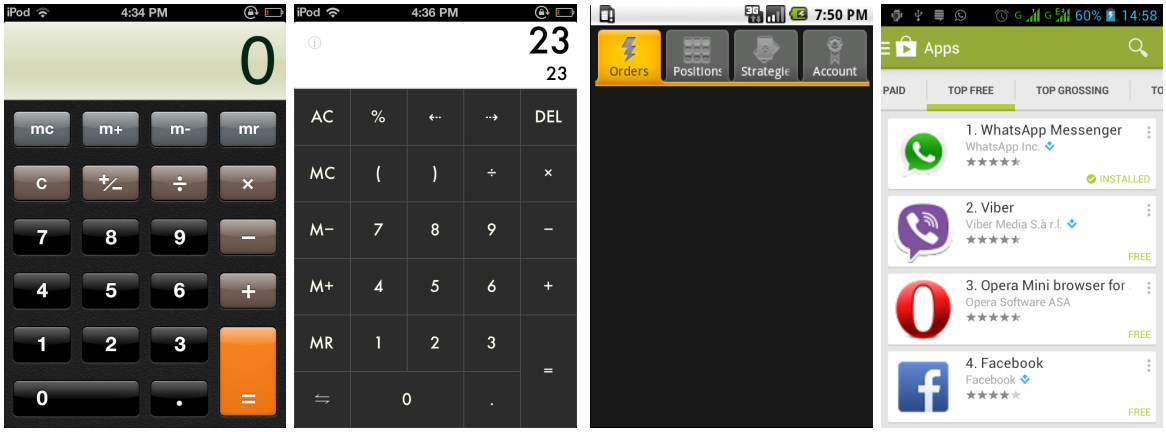
Více informací je možné dohledat na webových stránkách Google Design[2].

3.1.1 Tlačítka

Na obrázku 3.1a je možné vidět rozdíl mezi zastaralým návrhem — anglicky *„skeuomorphic design“* — kalkulačky a novým Material Designem. Tlačítka již nemají plastické provedení se stínováním a zmizely i mezery. Rozhraní tak působí elegantním dojmem a neruší zbytečnými efekty.

[1] User Experience

[2] https://design.google.com/



(a) Změna tlačítek (b) Nový návrh záložek

Obrázek 3.1: Na tomto obrázku je možné vidět rozdíly v prvcích, které přináší standard Material Design. Levá část ukazuje vizuální změnu podoby tlačítek kalkulačky, jež jsou nyní plochá a ničím uživatele neruší. Na pravé straně je zobrazen rozdíl designu záložek, kde zmizelo stínování a tablayout je celkově přehlednější.

3.1.2 Záložky

Obrázek 3.1b porovnává GUI uspořádané do záložek — anglicky *„tab layout“*. Material Design, v pravé části obrázku, stanovuje nový styl přechodů mezi záložkami i vzhled samotných tlačítek. Opět odsud zmizely stíny a prostorové efekty. Co z obrázku není patrné je právě způsob ovládání přechodů. Nově lze použít, kromě klasického kliknutí na tlačítko, gesto *„swipe“* [9].

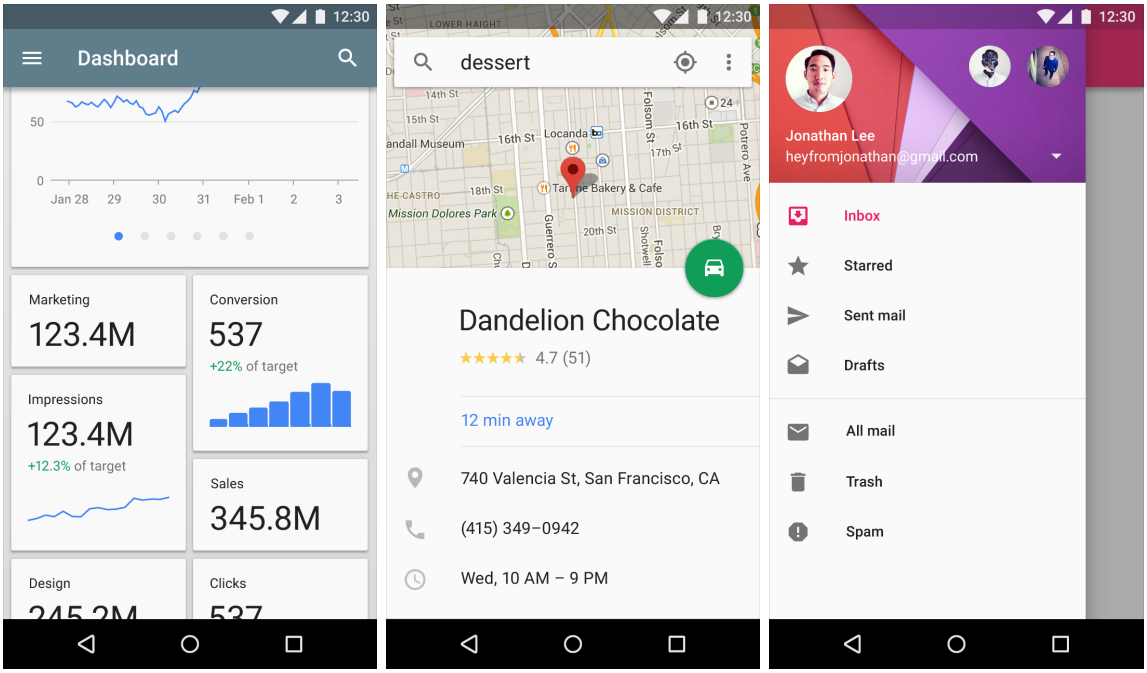
3.1.3 Zcela nové komponenty

Kromě nového designu již dříve používaných komponent, Google vytvořil několik úplně nových, které těží z výhod Material Design.

Cards mohou disponovat různým obsahem (text, obrázky, grafy a další). Jedná se o nový způsob zobrazení dat, která si může uživatel sám modifikovat podle vlastní hierarchie.

Bottom sheet je informační komponenta, zobrazující dodatečný obsah k aktivitě, kterou po vysunutí částečně překrývá.

Navigation drawer se vysouvá z levé nebo pravé hrany displeje, a zejména obsahuje základní navigační prvky a/nebo nastavení.



(a) Cards (b) Bottom sheet (c) Navigation drawer

Obrázek 3.2: Na obrázku je možné vidět tři nové komponenty, které jsou k dispozici ve standardu Material Design. „Cards“ nalevo umožňují vytvořit GUI s hierarchickou strukturou čtyřúhelníků, nesoucí různý textový i grafický obsah. Ve středu obrázku je ukázka „Bottom sheet“, což je výsuvná komponenta, doplňující informace k obsahu za pozadí. Vpravo je pak „Navigation drawer“ obsahující navigační menu.

3.2 Charakteristika systému Android

Google Android je mladý a rychle vyvíjený operační systém. Je postaven na Linuxovém jádře a proto není překvapením, že má otevřený zdrojový kód [2].

Pro vývojáře poskytuje bohatou dokumentaci ke všem dostupným metodám a technologiím, vysvětlení často podává přímo v kódu jazyka *Java*. Google navíc na svých webových stránkách umožňuje ke stažení spousty příkladů hotových aplikací pro konkrétní technologie [1]. Rychlý vývoj a rozsáhlá dokumentace si ale vybírá svou daň v podobě často neaktuálních informací. Aktualizace není při takovém objemu dat vůbec jednoduchá a vývojář se díky tomu může setkat s návodem, který používá zastaralé metody.

Uživatelé mají široké možnosti nastavení a vzhledu jejich grafického rozhraní. Na Google Play najdou velké množství aplikací a snáze si vyberou tu pravou pro své potřeby, než u konkurence, kde jsou na vývojáře vyvíjeny daleko větší požadavky. Zdatnější uživatelé si, díky Linuxovému jádru, mohou dokonce nahrát vlastní upravenou verzi systému, pokud jim nevadí ztráta záruky.

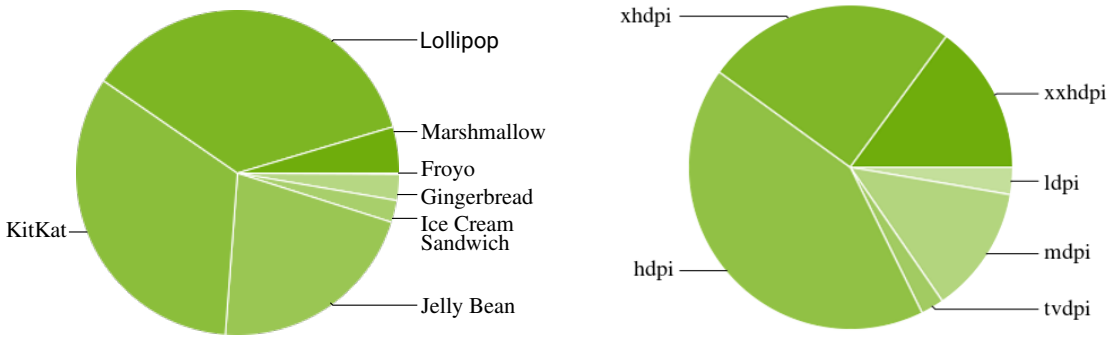
3.2.1 Mnoho různých verzí systému

S častými změnami operačního systému, bez možnosti aktualizace na starých zařízeních, vyvstává problém s roztrouseností používaných verzí. Vývojář používající nové technologie musí brát ohled na verze Android, které tyto funkce nepodporují. Samozřejmě pod podmínkou, že chce vydat aplikaci použitelnou pro starší zařízení. Na obrázku 3.3a je možné vidět v jakém aktuálním poměru jsou používané varianty systému Android [4].

3.2.2 Mnoho fyzických zařízení

Různé velikosti displejů zařízení, která mají nainstalovaný operační systém Android, komplikuje práci vývojářů a nutí je tvořit aplikace s ohledem na tuto skutečnost. V kapitole 7 je vysvětleno, jakými prostředky je možné vytvářet aplikace pro zařízení s různými velikostmi displejů. Obrázek 3.3b zobrazuje množství rozlišení, kterými disponují Android zařízení.

Hodně typů periferií může ovlivnit funkčnost aplikací, které s nimi komunikují. Bezproblémové focení na jednom zařízení může dělat velké problémy u jiného.



(a) Graf verzí systému

(b) Množství rozlišení displejů

Obrázek 3.3: Zde je možné si povšimnout, jaké množství verzí systému Android je uživateli aktuálně (k datu: 18.05.2016) používáno. Dále kolik různých druhů rozlišení Android podporuje a v jakém poměru využití jsou.

Kapitola 4

Strojové čtení pokladních účtenek

Pro úspěšné rozpoznání textu z obrázku je nutné učinit následující kroky. Nejdřív je třeba obrázek upravit, což zahrnuje oříznutí na část, která obsahuje pouze text, eliminaci šumu, vylepšení kontrastu a vyladění jasu. Dalším krokem je narovnání řádků detekovaného textu a úhlu, ze kterého je fotografie pořízena. Až poté je na optimalizovaném obrázku provedena detekce pomocí *neuronových sítí*, jimž je předložena datová sada obsahující potřebné informace o jazyce rozpoznávaného textu. Do větší hloubky jsem v rámci této práci neuronové sítě nestudoval, a zabývám se spíše použitím hotových řešení.

4.1 Knihovna Tesseract v prostředí Android

Pro funkci OCR na platformě Android se dá použít knihovna *tess-two* [31], dostupná přímo z *Gradle* repozitáře nebo také projekt *Tess4J*, psaný v jazyce Java. Obě dvě možnosti využívají knihovny *Tesseract* na rozpoznávání textu z fotky, a *Leptonica* na úpravy obrázků před samotným rozpoznáváním [20].

Pro zlepšení přesnosti rozpoznávání je před použitím knihovny doporučeno předzpracování fotografie. Tesseract sice tuto funkcionalitu interně podporuje, ale dle vývojářů nastávají situace, kdy jeho předzpracování není dostatečné [21].

4.2 Rozpoznávání textu u konkurence

Všechny neplacené testované aplikace používají knihovnu Tesseract, takže výsledky samotného rozpoznávání textu jsou velmi podobné. Rozdíly ale panují u předzpracování obrazu, na kterém je následně spuštěno OCR.

Velmi povedené aplikace umožňují interaktivní vylepšení fotky, při kterém si uživatel vybírá z předvolených módů. Tímto nutným zásahem uživatele se sice na jednu stranu prodlužuje proces samotného skenování, ale výrazně vylepšuje úspěšnost následného rozpoznávání.

Další užitečnou funkcí je automatické ořezávání důležité části fotky. Na té jsou detekovány hrany papíru a výsledný obrázek pak obsahuje pouze požadovaná data. V případě nasazení OCR na neoříznutou fotku může začít probíhat nechtěné zpracovávání elementů, připomínající text, mimo požadovanou část [24]. Tento případ značně prodlužuje proces rozpoznávání a tím i snižuje použitelnost aplikace.

Hrany detekovaného papíru na fotce jsou většinou barevně zvýrazněny. To dělají všechny aplikace, umožňující automatické ořezávání.

Vývojáři aplikace *Quick PDF Scanner* [19] šli ale mnohem dál. Jejich řešení fotí bez nutnosti uživatelova zásahu. Barva zvýrazněných hran se mění podle ostrosti zaměřené oblasti a při zaostření se oblast vyfotí.

4.2.1 OCR Instantly Free

Aplikace [16] umožňuje manuální úpravu fotky pomocí expozice a algoritmu na odstranění šumu. Samotný převod na text není nikterak rychlý, a 4Mpx fotku zpracuje až za 10 sekund. Výsledek je ale výborný a zahrnuje i diakritiku. Screenshoty z aplikace je možné vidět na obrázku 4.1.

Aplikace zpracovaný text nikam neukládá, ani s ním dále nepracuje. Slouží tedy výhradně pro digitalizaci textu za účelem okopírování a například odeslání emailem. Tomu odpovídá i jednoduché GUI, které obsahuje jen několik tlačítek a výrazně ničím neohromí.

4.2.2 TextScanner

Obrázek 4.2a popisuje úpravu vyfocené fotky a následné rozpoznání pomocí aplikace *TextScanner* [17]. Ta umožňuje, podobně jako předešlá aplikace, manuální úpravu pomocí jednoduchého posuvníku, a navíc i mazání textu na obrázku (zřejmě pro účely usnadnění OCR). Výsledný text je sice podobné kvality, ale své konkurenty předčí v rychlosti, za kterou text rozpozná. Stejný obrázek zpracuje již za 6 sekund.

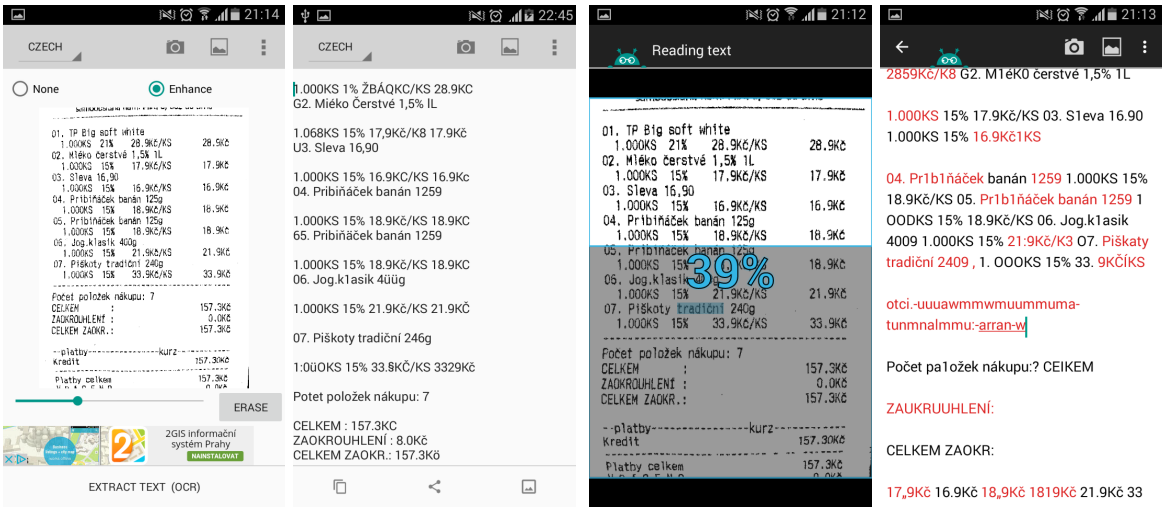
V rámci uživatelského rozhraní je možnost rozpoznaný text zkopírovat nebo rovnou sdílet. To je ale tak vše, co aplikace s textem, popřípadě obrázkem dovede.

4.2.3 Text Fairy

Hlavní výhodou posledního řešení z obrázku 4.2b je automatické vylepšení nasvícení, kontrastu a ostrosti vyfocené fotografie. Aplikace [22] disponuje funkcí automatického narovnání šikmého textu, a také pěknými animacemi, které přesně popisují aktuální stav zpracování. Jenže to trvá celkově více než 30 sekund, a výsledek není zdaleka tak přesný, jako u předešlých dvou aplikací. Uživatelské rozhraní vypadá pěkně a je zde k dispozici seznam všech rozpoznaných obrázků i s výslednými texty.

(a) Původní foto (b) Úprava (c) Rozpoznávání (d) Výsledek

Obrázek 4.1: Na tomto obrázku je možné vidět celý proces rozpoznávání textu z vyfocené účtenky pomocí aplikace OCR Instatntly Free. V levé části je zobrazena zašedlá fotografie po vyfocení, které je hned ve vedlejším kroku manuálně upraven kontrast a jas. Poté je nad upraveným obrázkem provedeno OCR a v posledním pravém kroku je vidět výsledek úspěšného rozpoznání.



(a) TextScanner (b) Text Fairy

Obrázek 4.2: V levé části tohoto obrázku je možné vidět ukázku z aplikace TextScanner, zejména jak je prováděna manuální úprava fotografie a celkový výsledek rozpoznávání. Vpravo pak následuje aplikace Text Fairy s přehledným popisem procesu OCR a výsledným digitalizovaným textem.

Kapitola 5

Návrh intuitivního uživatelského rozhraní

Po rozsáhlé analýze konkurenčních řešení jsem si ujasnil, co by měla má aplikace obsahovat a jakých chyb bych se měl vyvarovat. Již v průběhu testování konkurence jsem přemýšlel o vhodných alternativách řešení problémů, na které jsem u daného produktu narazil.

Cílem je tedy navrhnout GUI snadné na pochopení a ovládání. U aplikace tohoto typu předpokládám nejčastější použití v situaci, kdy jde uživatel z obchodu a chce si rychle uložit informaci o nákupu. Papírovou účtenku již nemusí nadále schovávat, aby si jednou za čas udělal přehled o výdajích, ale jednoduše se podívá do své oblíbené mobilní aplikace.

5.1 Eliminace nadbytečných úkonů

Při návrhu jsem se zaměřil na hlavní funkcionalitu a přemýšlel nad situacemi, ve kterých bude uživatel aplikaci používat. Výsledkem je rozhraní, které umožní primárně rychlé přidání nového záznamu. Ostatně je to věc, kterou jsem vytýkal v kapitole 2.2 hned několika aplikacím.

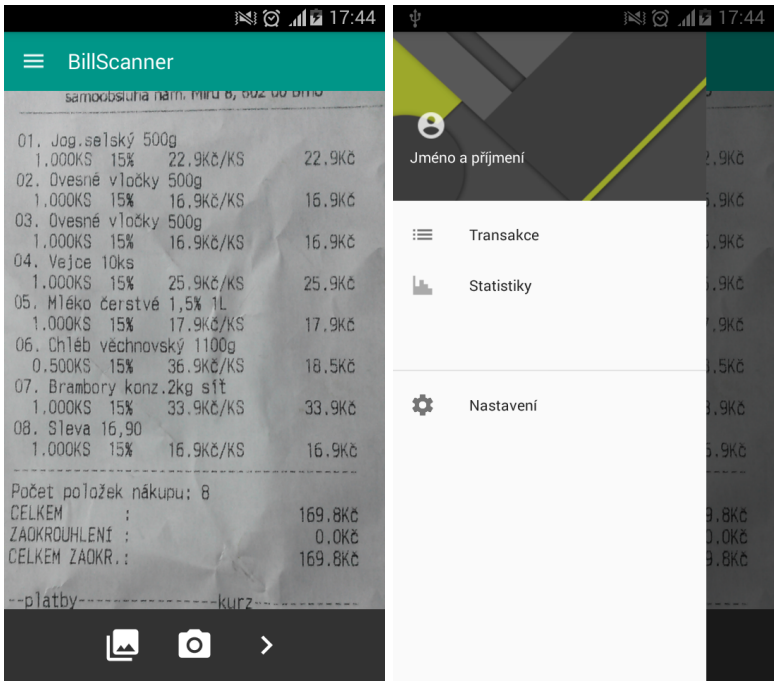
Vedlejší, přesto neméně důležité funkce jsem schoval do bočního výsuvného menu při úvodní aktivitě. Po spuštění aplikace tak může uživatel ihned fotit účtenku bez nutnosti dalších úkonů.

5.1.1 Zaměření na hlavní funkci aplikace

Na obrázku 5.1a je úvodní obrazovka, kterou uživatel uvidí bezprostředně po spuštění aplikace. Není tedy zdržován dalšími nutnými kroky, potřebnými pro zahájení focení. Ve spodní liště jsou pak tři ovládací tlačítka kde:

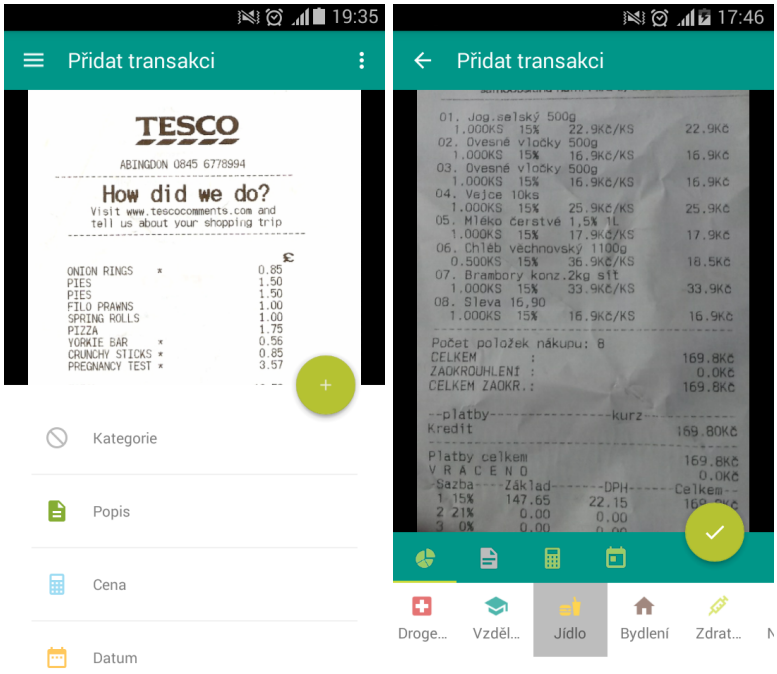
- **levé tlačítko** umožní načíst fotku z galerie telefonu
- **prostřední tlačítko** vytvoří novou fotku
- **pravé tlačítko** přeskočí focení a umožní uživateli přidat záznam manuálně

Po kliknutí na tlačítko v levém horním rohu se objeví výsuvné menu, jak je možné vidět na obrázku 5.1b.



(a) Focení účtenky (b) Aktivace výsuvného menu

Obrázek 5.1: Na levé straně obrázku je vidět úvodní aktivita, která umožňuje focení účtenky ihned po spuštění aplikace. Pravá část ukazuje stav, kdy je z levé hrany displeje vytaženo výsuvné menu.



(a) Původní návrh formuláře (b) S použitím tab layoutu

Obrázek 5.2: Tento obrázek znázorňuje změny, jimiž prošel formulář na vkládání záznamů o výdajích. Změny byly aplikovány na základně recenzí od uživatelů, a také z důvodů nepřesností z hlediska UX.

5.1.2 Formulář na vkládání dat

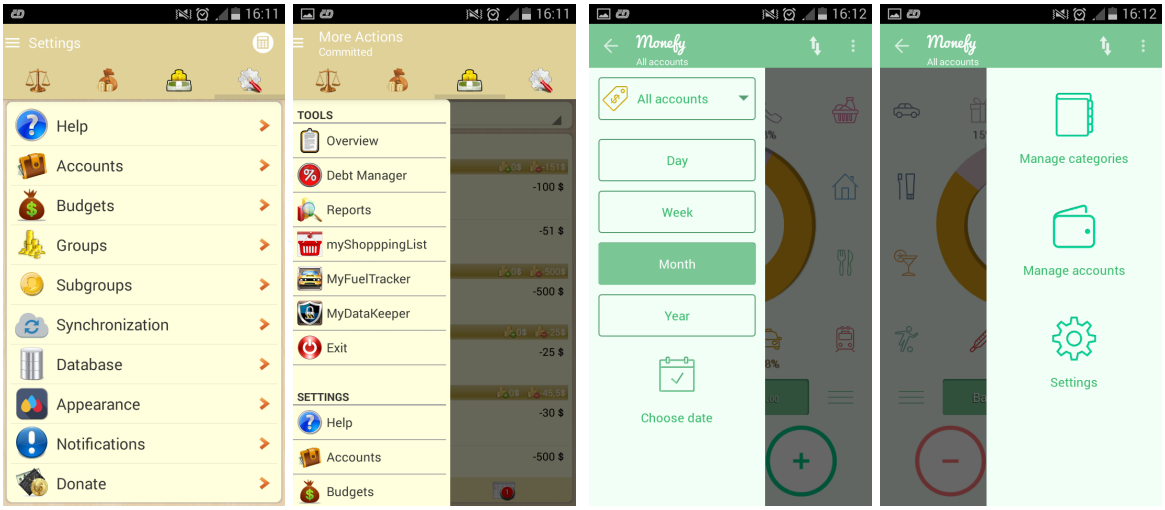
Po kliknutí na jedno z tlačítek úvodní obrazovky se uživateli objeví formulář, ve kterém vyplní nutné položky a po stisku žlutého tlačítka se záznam uloží.

Jak je z obrázků 5.2 patrné, formulář prošel jistým vývojem. Původní návrh pochází z projektu do předmětu ITU[1] na FIT VUT v Brně. V rámci předmětu došlo k testování GUI na několika vybraných uživatelích. Ti si ve svých recenzích stěžovali právě na nedostatky tohoto formuláře.

Na základě recenzí byl návrh kompletně předělán jak po technické, tak i po vizuální stránce. Jednotlivé položky formuláře byly přesunuty do struktury *tablayout*, kde ikonka na každé záložce označuje druh vstupních dat — zleva: *kategorie*, *popisek*, *cena* a poslední je *datum*.
Dále byl změněn symbol na potvrzujícím tlačítku z klasického *plus* na *„fajfku“*. Na první pohled drobná změna má ale velký význam z hlediska UX. Uživatelé zvyklí na chování tlačítka plus, které obvykle přidá nějakou položku do stávajícího seznamu nebo galerie, by mohli být v tomto případě zmateni. Potvrzením se totiž nic takového nestane, pouze se vypíše hláška, že byla nová transakce úspěšně přidána. Proto je vhodnější nově navržený symbol.

Informace o validitě formuláře je znázorněna zelenými barvami ikonek na záložkách. Pokud jsou vyplněny všechny povinné položky, tzn. všechny kromě popisku, vybarví se i samotné potvrzovací tlačítko, které jinak zůstává šedivé.

[1] Tvorba uživatelských rozhraní



(a) myMoneyTracker (b) Monefy

Obrázek 5.3: Na tomto obrázku je vidět porovnání uživatelských rozhraní dvou konkurenčních aplikací. myMoneyTracker na levé straně disponuje velmi složitým GUI, naproti tomu vývojáři Monefy ho vytvořili jednoduché a přehledné.

5.2 Omezení na nejpotřebnější funkce

Mnoho aplikací na Google Play, které jsem při návrhu GUI otestoval, disponuje velkým množstvím funkcí. Jejich spouštěcí tlačítka/ikonky pak zabírají spoustu prostoru a dělají uživatelské rozhraní komplikované a nepřehledné.

Nabízí se tedy otázka, zda opravdu všechny dostupné funkce používá alespoň polovina uživatelů. Při návrhu aplikace se snažím k tomuto problému přihlížet a od konkurence se tímto odlišit. Proto navrhuji jednoduché GUI s tím, že při testování na konkrétních uživatelích zjistím, co přesně by si dále představovali.

5.2.1 Mnoho funkcí neznačí dobrou aplikaci

Když Google začínal se svou první verzí vyhledávače, vytvořil bílou stránku s logem a vyhledávacím polem. Konkurence hýřila všemi možnými informacemi, o nichž si myslela, že uživatelé potřebují, a oni sami si to mysleli též. Google všem ukázal jinou cestu, a od té doby svůj vyhledávač po grafické stránce příliš nezměnil. S tímto vědomím se snažím na návrh aplikace nahlížet.

myMoneyTracker nabízí uživateli množství funkcí a nastavení [15]. Jak je možné vidět na obrázku 5.3a, vývojáři se ale toto kvantum příliš do grafického rozhraní zakomponovat nepodařilo.

Monefy má zato velmi jednoduché GUI [13]. Vývojáři se podařilo elegantním způsobem schovat všechnu doplňující funkcionalitu do postranních menu. Velká tlačítka, na pravé straně obrázku 5.3b, po kliknutí navíc odkryjí roletový seznam dalších nastavení.

5.2.2 Funkce, které chtějí sami uživatelé

Z testování, které proběhlo v rámci již zmíněného projektu předmětu ITU, jsem se dozvěděl o několika funkcích, které v mé aplikace uživatelům nejvíce chybí. Tímto stylem budu nadále pokračovat a postupně zohledňovat jejich potřeby.

Volba rozsahu výpisů a statistik byla nejčastěji zmiňovaným nedostatkem, který se v první verzi aplikace objevil. Na obrázku uvedeném v kapitole 7.5.2 je vidět řešení tohoto problému.

Statistiky podle kategorií se v recenzích objevily taktéž hned několikrát. Uživatelé by ocenili možnost třídit nejen podle data, ale také podle kategorií. Graf by pak obsahoval součet a porovnání výdajů v jednotlivých měsících pro konkrétní kategorii. Navíc by bylo užitečné mít možnost sledovat i měsíční trendy — zda uživatel opravdu šetří tam, kde chce.

Seznam oblíbených produktů je velmi zajímavou funkcí, pomocí které by si uživatel zvolil několik produktů, jejichž cenu by aplikace hlídala přes službu *Kupi.cz*[2] a upozorňovala ho na akce.

Správa vyfocených účtenek se v recenzích objevila sice jen jednou, ale to zdaleka nevypovídá o její důležitosti. Počítám s ní v dalších verzích práce.

Nastavení rozpočtu na měsíc je určitě užitečné, a do budoucna s jeho implementací počítám.

Podpora různých měn by je dle recenze mohla automaticky pomocí OCR rozpoznat a předvyplnit. Určitě zajímavý nápad, ale implementaci nedávám takovou prioritu, jako výše zmíněným.

[2] https://www.kupi.cz/

Kapitola 6

Zjednodušené zadávání položek z účtenky

Tato kapitola popisuje podstatu vzniku aplikace BillScanner. Při používání mobilních aplikací na správu osobních financí jsem si uvědomil, že manuální přidávání položek z účtenky zabírá spoustu času, a tím pádem vzbuzuje nechuť k samotnému používání. Znechucen manuálním přidáváním, přestal jsem si záznamy o útratách vést.

Současně jsem měl zkušenosti s produkty pro mobilní telefony, které umí zpracovávat vyfocený dokument do textového souboru. Napadlo mě tedy, že bych využil rozpoznávání textu k usnadnění práce při vedení záznamů o svých výdajích.

Na úvod bych ještě poznamenal, že konkurence se tento problém snaží samozřejmě také řešit. Napovídá při zadávání kategorií a popisů nebo vytváří nákupní šablony. V tomto je velmi dobrá aplikace *Money Lover* [14], kterou jsem používal nejdéle.

Ke grafickému rozhraní, které umožňuje focení ihned po spuštění aplikace tak přibývají tři velmi podstatné funkce.

6.1 Rozpoznání data

Pomocí technologie OCR rozpoznám datum na účtence a automaticky předvyplním položku formuláře, jak je vidět na obrázku 6.1a. Uživatel již nebude muset hledat údaj o datu pořízení, který bývá často u každého obchodu na jiném místě účtenky.

6.2 Zvýraznění cen

Druhý důležitý údaj, který z fotografie rozpoznám jsou ceny jednotlivých nakoupených položek. Uživatel už není nucen sčítat ceny ručně, po jejich stisknutí to udělá aplikace za něho. Obrázek 6.1b obsahje návrh chování aplikace a barevné zvýrazňování cen.

6.3 Více kategorií na účtence

V situaci, že je na účtence více druhů položek a uživatel je chce roztřídit do několika kategorií, tato aplikace mu to snadno umožní. Při stisknu potvrzovacího tlačítka se vloží nový záznam a uživatel může pokračovat s dalšími kategoriemi. Jak je vidno z obrázku 6.1c, již jednou potvrzené ceny mají jiné barevné označení. Takže by se nemělo stát, že uživatel přidá jednu položku vícekrát.

(a) Rozpoznané datum (b) Označené ceny (c) Více kategorií

Obrázek 6.1: Na tomto obrázku je možné vidět aplikaci BillScanner, používající technologie OCR pro rozpoznávání data a cen na účtence. Na levé části je zvýrazněno rozpoznané datum a ve středu obrázku aplikace označuje ceny a umožňuje na jejich text kliknout. Na pravé straně je pak vidět případ, kdy uživatel přidá záznam s jednou kategorií a aplikace mu umožní zadání nových hodnot pro další záznamy.

Kapitola 7

Programová část vývoje aplikace

Vývoj pro operační systém Android lze vést několika způsoby, v různých programovacích jazycích i vývojových prostředích.

Corona [6] nabízí multiplatformní vývoj v jazyce *Lua*. *Appcelerator* [5] se taktéž nezaměřuje pouze na jednu platformu a podporuje jazyk *JavaScript*. Velmi zajímavým prostředkem je *PhoneGap* [18], umožňující psaní v *HTML*. Vytvoří v lokální síti *Node.js* server, přes který přenáší změny na aplikaci do připojeného mobilního zařízení. To vše v reálném čase. Dále je možný vývoj v *Qt*, *C++* a další.

Já jsem se rozhodl postupovat podle oficiálního návodu na Android Developers [1]. Tedy vyvíjet v jazyce *Java* a grafickou část navrhovat v *XML*. Zprvu jsem používal prostředí *Android Studio*[1], ale později jsem přešel, kvůli lepší podpoře práce s databází, na *IntelliJ IDEA*.

7.1 Prostředky Material Design v praxi

Společnost Google se vývojářům Android aplikací snaží usnadnit práci, kromě dokumentace a návodů, také hotovými komponentami, které může každý jednoduše přidat do svého kódu. Zefektivňuje tím vývoj, ale také pomáhá dodržovat pravidla správného návrhu rozhraní a základní myšlenky UX. Komponenty si může každý vývojář samozřejmě upravit dle libosti, ale základní chování je přednastavené tak, jak to popisuje Material Design. Vývojáři mohli docílit vizuální podoby a chování prvků i před tím, ale Google jim to vytvořením tohoto standardu značně zjednodušil. Některé komponenty jsem zmiňoval v kapitole 3.1.3 a zde popíši i jejich programovou část.

Při použití *Navigation drawer* je nutné zvolit `DrawerLayout` za kořen XML souboru a element komponenty umístit do jeho spodní části tak, aby při aktivaci překrýval všechny elementy v rozhraní. Pro představu je to `NavigationView` na obrázku 7.1.

Dalším pomocníkem je *Floating Action button*, který „plave“ nad všemi elementy, takže musí být taktéž ve spodní části XML souboru. Navíc musí být umístěn v elementu `CoordinatorLayout`. To je v podstatě vylepšený `FrameLayout`, s tím rozdílem, že umí pracovat s novými komponentami Material Design.*Floating Action button* používám například ve formuláři 5.2, kde je to ono „potvrzovací tlačítko“.

Poslední komponentou, kterou tu zmíním, je *Toolbar*. Jedná se o základní navigační prvek, který je umístěn v horní části GUI. Kromě textu a tlačítek, jak je tomu na ob-

[1] V době psaní BP je nově k dispozici verze 2.0, kterou jsem nevyzkoušel. Je ale možné, že v ní vývojáři neduhy první verze opravili.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout>
        <android.support.design.widget.CoordinatorLayout>
                <LinearLayout>
                        <android.support.design.widget.AppBarLayout>
                                <android.support.v7.widget.Toolbar/>
                        </android.support.design.widget.AppBarLayout>
                </LinearLayout>
                <android.support.design.widget.FloatingActionButton/>
        </android.support.design.widget.CoordinatorLayout>
        <android.support.design.widget.NavigationView/>
</android.support.v4.widget.DrawerLayout>
```

Obrázek 7.1: Tento obrázek ilustruje případ použití nových XML elementů pro grafický design Android aplikací. Konkrétně je tu vidět hierarchie jednotlivých elementů, kterou je nutné dodržovat pro plnou funkčnost – NavigationView musí být umístěn jako poslední element v souboru, aby při jeho aktivaci překryl všechny prvky uvedené nad ním.

rázku 5.2, může obsahovat i celé obrázky, roletová menu a sousty dalších prvků [1]. K tomu lze animovat jeho výšku i jiné vlastnosti.

7.2 Knihovny pro zpětnou kompatibilitu

V kapitole 3.2.2 jsem nastínil problém, se kterým se setká každý vývojář aplikací pro operační systém Android. Nyní popíši způsob, kterým se Google snaží zmírnit dopad tohoto problému na samotné vývojáře.

Vydává podpůrné knihovny, zajišťující stejnou funkčnost a design, jak v nových verzích systému, tak i v těch starších. Příkladem může být element `TextView`, který když navrhnu pro verzi 6.0[2], tak může na verzi 3.0 vypadat úplně jinak a nemusí ani fungovat animace. Alternativou je `AppCompatTextView`, který má stejnou funkcionalitu, ale slibuje kompatibilitu se staršími verzemi.

V knihovnách jsou také elementy `AppCompatEditText`, `AppCompatSpinner` a další [1].

7.3 Nástroje pro snadnější vývoj

Kromě knihoven pro zpětnou kompatibilitu vývojáři usnadní práci množství nástrojů, které jsou mu často k dispozici zcela zdarma.

Základním předpokladem pro efektivní vývoj je chytré *IDE*[3]. Já jsem používal *Android Studio*, které je vybudováno na prostředí *IntelliJ IDEA Community Edition* od firmy *JetBrains*. Jedná se ale o neplacenou verzi, a ta nepodporuje tak dokonalou správu databáze v testovacím zařízení, jako to dělá placená *Ultimate Edtition*. Placená verze je ale pro studenty zdarma, a tak jsem přešel na ni.

Při práci s obrázky nebo ikonami se hodí nástroj `Android Drawable Importer`. Existuje jako doplněk do obou výše zmíněných vývojových prostředí, a jak už název napovídá, slouží

[2] Při psaní BP nejnovější verze systému Android
[3] Integrated Development Environment

pro snadnější vkládání různých vizuálních dat do otevřeného projektu.

Při programování v jazyce Java jsem byl vždy zvyklý na vypisování ladících dat na konzoli pomocí `System.out.println()`. IntelliJ nabízí elegantnější řešení v podobě nástroje *logcat*, který ke každému výpisu nastaví vývojářem zvolený identifikátor, a podle něho pak umožní zprávy filtrovat.

Šikovným online nástrojem je *Material Palette* [12], kde si může vývojář interaktivně vybrat barvy svého GUI a nástroj mu zobrazí příklad, jak budou kombinace vypadat. K tomu mu vygeneruje soubor `colors.xml` s konfigurací barev celého vzhledu.

Za nutnost se dá označit možnost spouštět aplikace ve virtuálním stroji, jehož konfigurace a monitorování je k dispozici přímo v IDE. Stroj může mít podobu telefonu, tabletu, televize nebo i hodinek.

7.4 Programování s knihovnou Tesseract

Tesseract je k dispozici ve většině standardních repozitářů Linuxových distribucí. Taktéž existují verze pro Windows i Mac. Po stažení je možné uplatnit *OCR engine* `libtesseract` nebo konzolovou aplikaci `tesseract`.

Knihovna zpracovává text v kódování *UTF-8* a podporuje více než 100 různých jazyků. Pro vývoj vlastní aplikace je možné použít právě knihovnu `libtesseract`, která je napsaná v *C/C++*. V případě programování v jiném jazyce již komunita vývojářů vytvořila množství tzv. *wrapperů*, umožňujících volání C/C++ funkcí z jiného programovacího jazyka [20].

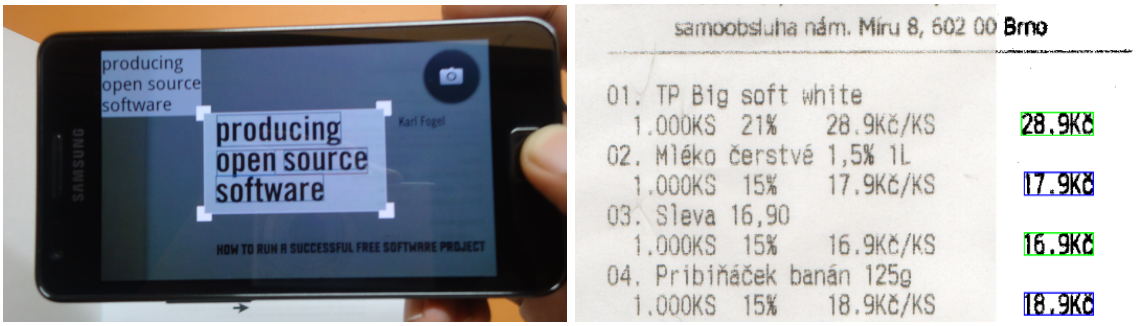
7.4.1 Tesseract v příkazové řádce

Před integrací knihovny do mobilní aplikace jsem vyzkoušel, zda je vůbec reálné rozpoznávat účtenky tímto nástrojem. Použil jsem tedy Linuxovou verzi `tesseract` a účtenku, naskenovanou domácím skenerem při 300 DPI.
Nástroj se používá následujícím způsobem:

```
tesseract vstup.png výstup -l jazyk [nastavení]
```

kde `nastavení` může nabývat jedné z těchto hodnot:

- **hocr** vygeneruje html soubor, který bude obsahovat všechna rozpoznaná slova a navíc jejich souřadnice.
- **pdf** nastaví formát výstupu na pdf, ve kterém umožní snadno vyhledávat nalezená slova.
- **makebox** vypíše do textového souboru tabulku se souřadnicemi levého dolního a pravého horního rohu rámečku, ve kterém rozpoznal nějaký text.



(a) Inspirativní řešení (b) Jednoduché předzpracování

Obrázek 7.2: Na levé straně obrázku můžeme vidět řešení, kterým jsem se nechal při návrhu aplikace inspirovat. Pravá část pak ukazuje vlastní řešení a rozdíl zašedlé fotografie po vyfocení oproti upravené části, na které je prováděno rozpoznávání cen.

7.4.2 Knihovna tess-two

Po ověření, že je tento nástroj opravdu schopen detekovat text a čísla na oskenované účtence, jsem začal hledat příklad reálného použití. Zjistil jsem, že je Tesseract hojně využívaný, ale pouze k aplikaci *DatumDroid* [25] byly dostupné i zdrojové kódy. O ní jsem se dozvěděl v návodu, který stručně a jasně popisuje nutné kroky, vedoucí ke zprovoznění OCR na mobilním zařízení [26]. Navíc mě zaujal obrázek 7.2a, který přesně vystihuje způsob, jak chci označovat ceny na účtence.

Později jsem zjistil, že funkčnost DatumDroid neodpovídá zmíněnému obrázku, a tak mi aplikace posloužila pouze pro pochopení základní práce s knihovnou *tess-two*. Pro představu se jedná o následující kód:

```
TessBaseAPI baseApi = new TessBaseAPI();
baseApi.setDebug(false);
baseApi.init(DATA_PATH, lang); //cesta ke slovniku, jazyk
baseApi.setVariable(TessBaseAPI.VAR_CHAR_WHITELIST, "0123456789.Kkc");
baseApi.setImage(photo); //vyfocena uctenka
baseApi.getUTF8Text();  //spusteni procesu OCR
```

Před zahájením OCR ořízna obrázek, jenž je uložen v proměnné `photo`, na jeho pravou čtvrtinu. Oříznutím dosáhnu rychlejšího zpracování, protože se bude rozpoznávat pouze část účtenky, kde předpokládám existenci cen položek. Samozřejmě se jedná pouze o dočasné řešení, které nemusí fungovat na všech typech účtenek. Poté převedu oříznutý obrázek na černobílý, čímž zvýším šanci na úspěšné rozpoznání ceny. Obrázek 7.2b ukazuje základní předzpracování, které jsem v aplikaci implementoval.

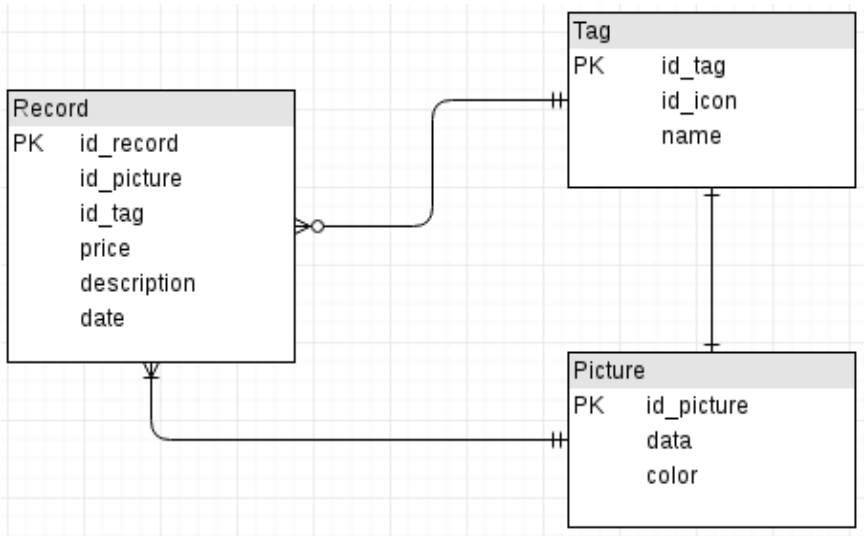
Zavoláním metody `getUTF8Text` se spustí OCR, po jehož dokončení je výsledek uložen v objektu `baseApi`. Pomocí `getResultIterator` a cyklu je pak možné přistoupit k jednotlivým výsledkům.

```
iterator = baseApi.getResultIterator();
iterator.begin();
do {
    lastText =
        iterator.getUTF8Text(TessBaseAPI.PageIteratorLevel.RIL_WORD);
    lastBox =
        iterator.getBoundingBox(TessBaseAPI.PageIteratorLevel.RIL_WORD);

    if (lastText != null) {
        if (isItPrice(lastText)) {
            fullBox = convertToFullSized(lastBox, pivot);
            price = parsePrice(lastText);

            if (price != null) {
                ocrList.add(new OCRItem(fullBox, price));
                drawRect(fullBox, Color.BLUE);
            }
        }
    }
}
while (iterator.next(TessBaseAPI.PageIteratorLevel.RIL_WORD));
```

Cyklus prochází všechna nalezená slova. U každého z nich je rozhodnuto, zda se jedná o cenu, a v tom případě je z něho extrahováno číslo. Důležitým údajem jsou také souřadnice obdélníku, ve kterém bylo dané slovo nalezeno. Oba údaje jsou uloženy a později použity pro identifikaci ceny, na kterou uživatel klikl. Pomocí souřadnic jsou taktéž modrým rámečkem zvýrazněny všechny nalezené ceny.



Obrázek 7.3: Tento obrázek zobrazuje ER diagram databáze, použité v aplikace BillScanner. Schéma obsahuje tři entity, pomocí kterých jsou uchovávány záznamy o výdajích a informace o vyfocených účtenkách v perzistentním stavu.

7.5 Práce s perzistentními daty

Při tvorbě jakéhokoliv programu, který bude při práci nějakým způsobem shromažďovat data, je nutné se zamyslet, jakým způsobem je bude uchovávat. S tím je spjat problém perzistence, neboli zajištění existence potřebných dat i po vypnutí programu, či dokonce celého zařízení.

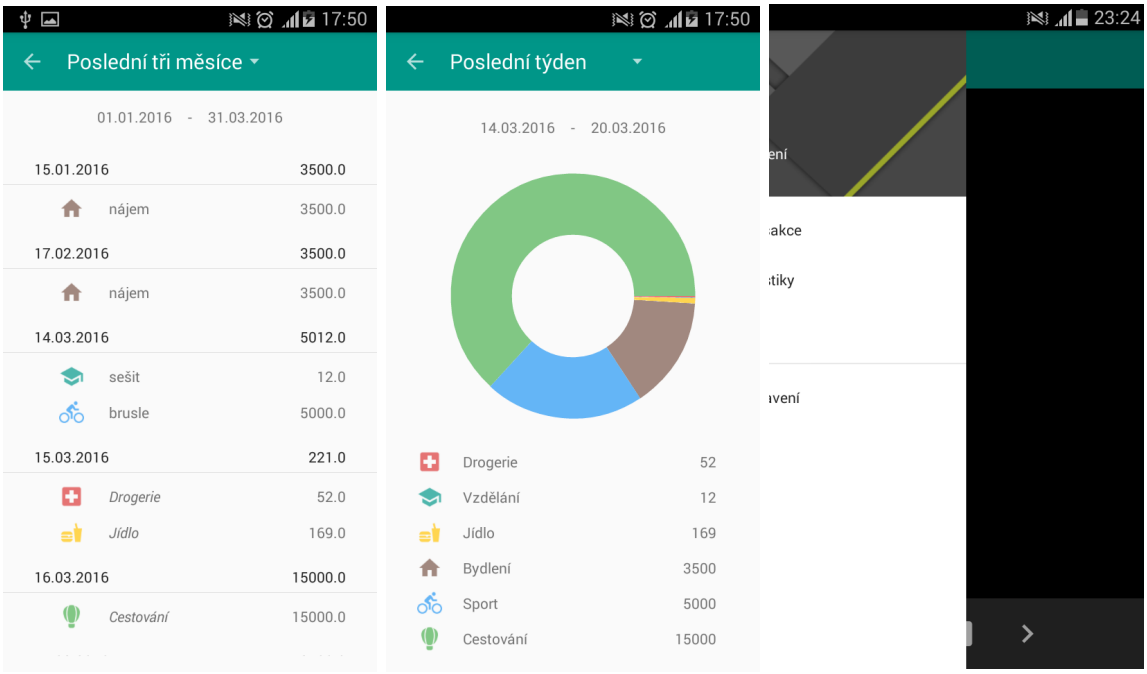
Já jsem se rozhodl ukládat data do databáze. Konkrétně *SQLite*, která má nativní podporu v systému Android a je doporučena i na webových stránkách pro vývojáře [1].

7.5.1 Ukládání dat do databáze

Navrhl a vytvořil jsem tedy databázi podle *ER diagramu* 7.3, do které ukládám informace o přidaných transakcích a fotkách účtenek. V prostředí Android k tomu slouží třída *SQLiteOpenHelper*, umožňující jednoduchou cestou vytvářet tabulky, a do nich přidávat záznamy. Tvorba tabulek se dělá jednoduše metodou `execSQL`, které se předá například následující řetězec:

```
"CREATE TABLE "+ V.T_PIC +"("+
    V.C_ID +" INTEGER PRIMARY KEY AUTOINCREMENT DEFAULT 0, "+
    V.C_DATA +" TEXT NOT NULL, "+
    V.C_COLOR + " TEXT"+
")";
```

Na vkládání, editaci a mazání záznamů slouží metody `insert`, `update` a `delete`, které náleží právě třídě *SQLiteOpenHelper*.



(a) Výpis transakcí (b) Graf kategorií (c) Úvodní obrazovka

Obrázek 7.4: Levá část obrázku zobrazuje možnost vypsání transakcí do přehledného seznamu. Roletovým menu v horním panelu je uživatel schopen měnit rozsah výpisů. Ve středu obrázku je možné vidět druhý typ výpisů, kterým je souhrnný graf všech kategorií za vybrané období. Pravá strana ukazuje případ, kdy se při spuštění aplikace BillScanner zatím nenačetl náhled kamery, ale i přesto může uživatel bez problémů vysunout menu a pokračovat v jedné z uvedených voleb.

7.5.2 Vizuální reprezentace dat

Pro výběr z databáze používám metodu `query`, jejímiž parametry se definuje vybraná tabulka a specifické omezení/podmínky datových hodnot. Metoda vrací objekt `Cursor`, ve kterém jsou uloženy všechny výsledky databázového dotazu odpovídající zvoleným podmínkám. Přes všechny výsledky se dá iterovat třeba následujícím způsobem:

```
if (cursor.moveToFirst()) {  //nastavi kurzor na prvni prvek
  do {
    .
    .
  } while (cursor.moveToNext()); //presune ukazatel na dalsi prvek
}
```

Dolování dat z databáze používám hlavně na výpisy vložených finančních záznamů a na jejich statistiky. To je ostatně možné vidět na obrázcích 7.4a a 7.4b. Dále také do databáze ukládám názvy kategorií a jejich ikonky. Na kreslení grafů v aplikaci pak používám knihovnu *AndroidPlot* [3].

7.6 Implementace vlastní obsluhy kamery

Focení v mobilní aplikaci systému Android jde dvěma způsoby. Buď vývojář použije existující aplikaci v zařízení, která na vyžádání začne obsluhovat kamerovou periferii, nebo vytvoří vlastní obsluhu a integruje kamerové funkce přímo do své aplikace. První metoda je vhodná v aplikacích, jejichž hlavní funkcí není tvorba fotografií, a možnost používání fotoaparátu je jen doplňkovou funkcionalitou. V mém případě aplikace těží z možnosti focení obrázků mobilním telefonem, a tak jsem naprogramoval vlastní obsluhu kamery.

V operačním systému Android je připravená třída `SurfaceView`, starající se o živý náhled z kamery na displej zařízení. Při zahájení komunikace s periferií dochází ke zvýšenému počtu výměn zpráv mezi aplikací a operačním systémem. To může mít za následek zpomalení reakcí na uživatelovy podněty.

Řešením je třída `HandlerThread`, přes kterou je možné spustit počáteční komunikaci a vykreslení náhledu ve vlákně běžícím na pozadí. V mé práci tohoto využívám při každém přechodu na úvodní obrazovku. Do doby, kdy je komunikováno na pozadí, je místo živého náhledu černá obrazovka. Umožňuje to ničím nepřerušené používání aplikace v případě, že si uživatel nechce vyfotit novou účtenku, ale pouze prohlédnout záznamy nebo statistiky. Bez implementace popsaného řešení dochází při používání k časovým prostojům, a to je z hlediska použitelnosti a UX naprosto nevyhovující.

Obrázek 7.4c přibližuje popsanou situaci. Náhled má černou barvu, tzn. že se ještě nenačetl, ale i tak je uživatel schopen aplikaci používat a kliknout na jednu z možností. Následující kód zajišťuje zmíněné řešení:

```
mHandler.post(new Runnable() {
    @Override
    public void run() {
        mCamera = null;
        try {
            mCamera = Camera.open();
            mCamera.setPreviewCallback(null);
            mCamera.startPreview();

        } catch (final Exception e) {
            interrupt();
        }

        MainActivity.this.runOnUiThread(new Runnable() {
            @Override
            public void run() {
                mCamPreview = new CameraPreview(getApplicationContext(),
                    mCamera);
                mFramePreview.addView(mCamPreview);
            }
        });
    }
});
```

V bloku `try-catch` jsou spouštěny ony výpočetně náročné operace, a po jejich úspěšném dokončení se metodou `runOnUiThread` startuje živý náhled z kamery.

Kapitola 8

Zveřejnění aplikace na Google Play

Pro zveřejnění aplikace v obchodě Google Play je nutné vytvořit účet na *Developer Console* [8] a zaplatit registrační poplatek 25$. Poté již vývojář řídí celý proces přidávání aplikace, nahrávání revizí a další náležitosti přímo z Console. Ta nabízí základní statistiky o zařízeních, na kterých je aplikace nainstalovaná, přehled recenzí od uživatelů a hlášení o chybách.

8.1 Alfa verze pro první testování

Alfa verze je dostupná pouze pro úzký okruh uživatelů a vydává se za účelem zjištění prvních chyb v aplikaci. V Consoli se dá snadno nastavit způsob zveřejňování alfa verze. Vývojář si může vybrat z otevřeného a uzavřeného testování. Volbou první možnosti se k testování může připojit libovolný člověk, který klikne na unikátní odkaz vygenerovaný Consolí. Druhým způsobem může vývojář pomocí emailových adres definovat konkrétní uživatele, které chce zapojit do testování. Mezi oběma možnostmi lze kdykoliv přepnout.

Já zvolil otevřené testování. Vygenerovaný odkaz 8.1 jsem sdílel s nejbližšími kamarády, spolužáky a rodinou. Celkově byla aplikace nainstalována na 7 různých zařízení.

8.1.1 Co je třeba před prvním vydáním

Před zveřejněním produktu je třeba ho podepsat privátním klíčem, kterým je každý vývojář identifikován. Podepisování se provádí v rámci kompilace programu. Klíč je vyžadován i při každé nové revizi aplikace, takže se jeho ztráta příliš nedoporučuje.

Dalšími doporučenými úkony jsou: lokalizace aplikace pro více jazyků, popřemýšlení o placené verzi, snížení velikosti obrázků na minimální přípustnou hodnotu, podpora co nejvíce rozlišení a verzí Android, zrušení ladících výpisů, a také použití nástroje *Proguard* pro optimalizaci a minifikaci kódu [27]. Minifikace znesnadňuje dekompilaci, a následnou krádež zdrojových kódů.

8.1.2 Chyby odhalené alfa verzí

Tato verze odhalila vysokou paměťovou náročnost, projevující se ukončením aplikace po vyfocení účtenky. Chyba se objevila u staršího zařízení, které nedisponovalo větším množstvím operační paměti. Problém jsem opravil změnou zacházení s fotografiemi. Původně jsem je převáděl na pole bytů a následně ukládal do databáze.

Podobný problém nastal i u nového zařízení s 2GB RAM. Fotilo totiž příliš velké obrázky, a ty měly na střední rozlišení fotografie velikost větší než 2MB. Chyba nastávala v SQLite

databázi, která standardně nepodporovala tak velké položky. Logicky jsem usoudil, že bude mnohem lepší ukládat do databáze informace o tom, kde jsou obrázky umístěny v lokálním úložišti. To vyřešilo i předchozí problém.

8.2 Veřejná beta verze

Po vyřešení problémů, jež odhalí alfa verze, je vydávána beta verze. Neobsahuje všechnu navrženou funkcionalitu, a je v ní pořád předpokládán výskyt chyb. Tudíž je zvykem, že za ni zákazník stále ještě neplatí. Slouží pro testování veřejností, která může mít kromě odhalování dalších chyb taktéž připomínky k chybějícím funkcím. Očekává se tedy větší množství poznámek a nápadů.

Taková verze je již veřejně k dispozici na Google Play, což přináší další povinnosti pro vývojáře, zejména vytvoření loga a náhledové grafiky. Vhodné je taktéž propagační video, na němž je demonstrováno použití i veškeré možnosti aplikace.

K vydání této verze jsem se bohužel v rámci bakalářské práce nedostal, ale po zhotovení loga a vyřešení formální stránky aplikace, což zahrnuje licence a práva, předpokládám zveřejnění i beta verze.

8.3 Kompletní placená verze

Až dojde k odladění všech chyb základních funkcí, předpokládám zpoplatnění aplikace. Přikláním se spíše k umožnění dokoupit OCR funkci přímo v aplikaci, než uživatele nutit ke koupi před samotným vyzkoušením free verze.

Nabízí se také dvě možnosti, jak implementovat placenou verzi:

Měsíční zkušební verze dovolí uživateli vyzkoušet plný potenciál aplikace, a to může později hrát roli při jeho rozhodování, zda ji koupit natrvalo.

Omezení na počet účtenek za měsíc by mohlo být atraktivní z hlediska větších uživatelských možností. Každý by si vybral tarif podle svých potřeb a frekvence nákupů.

Obrázek 8.1: QR kod nesoucí odkaz pro přístup do alfa testování aplikace BillScanner.

Kapitola 9

Závěr

Výsledkem této práce je mobilní aplikace, jejíž GUI se řídí novými trendy Material Design. Propracované uživatelské rozhraní disponuje z hlediska UX logicky uspořádanými prvky a přehlednými grafy.

Analyzoval jsem množství existujících aplikací dostupných na Google Play a vymýšlel řešení problémů, na které jsem u nich narazil. Při pravidelných konzultacích, u vedoucího práce profesora Adama Herouta, jsem diskutoval nové poznatky a návrhy. Nastudoval a implementoval jsem použití databáze pro ukládání perzistentních dat na mobilních zařízeních, s využitím SQLite. Taktéž jsem vytvořil vlastní komunikaci s rozhraním mobilního fotoaparátu a její výpočetně náročné části přesunul do vlákna, běžícího na pozadí.

Za hlavní úspěch považuji zprovoznění OCR na optimalizované fotografii účtenky, ze které jsem schopen zjistit ceny položek nákupu. K jejich zvýraznění barevným rámečkem jsem musel nastudovat základní možnosti, kterými operační systém Android umožňuje kreslení na obrázky.

Do budoucna plánuji vylepšení předzpracování vyfocené účtenky, dále detekci papíru na fotografii a její ořezávání. V nejlepším případě i automatické focení. Za velmi dobrou funkcionalitu považuji možnost zpracovávání měsíčních výpisů elektronického bankovnictví, které posílají banky klientům zpravidla emailem. Další prostor pro zdokonalování se nalézá v oblasti dolování dat z databáze a vylepšování statistik výdajů.

Literatura

[1] *Android – Developers*. [Online; navštíveno 26.04.2016].
URL http://developer.android.com/index.html

[2] *Android – Open source*. [Online; navštíveno 20.04.2016].
URL https://source.android.com/

[3] *AndroidPlot*. [Online; navštíveno 08.05.2016].
URL http://androidplot.com/

[4] *Android – System versions*. [Online; navštíveno 26.04.2016].
URL http://developer.android.com/about/dashboards/index.html

[5] *Appcelerator*. [Online; navštíveno 08.05.2016].
URL http://www.appcelerator.com/

[6] *Corona Labs*. [Online; navštíveno 08.05.2016].
URL https://coronalabs.com/

[7] *Expensify – Expense Reports*. [Online; navštíveno 08.05.2016].
URL
https://play.google.com/store/apps/details?id=org.me.mobiexpensifyg

[8] *Google Play Developer Console*. [Online; navštíveno 26.04.2016].
URL https://play.google.com/apps/publish/

[9] *Material design components – Tabs*. [Online; navštíveno 20.04.2016].
URL http://www.google.cz/design/spec/components/tabs.html

[10] *Material design – Goals*. [Online; navštíveno 20.04.2016].
URL http://www.google.cz/design/spec/material-design/introduction.
html#introduction-goalsl

[11] *Material design – Principles*. [Online; navštíveno 20.04.2016].
URL http://www.google.cz/design/spec/material-design/introduction.
html#introduction-principles

[12] *Material Palette*. [Online; navštíveno 08.05.2016].
URL http://www.materialpalette.com/

[13] *Monefy – Money Manager*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.monefy.app.lite

[14] *Money Lover*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.bookmark.money

[15] *My Money Tracker*. [Online; navštíveno 08.05.2016].
URL
https://play.google.com/store/apps/details?id=com.ic.myMoneyTracker

[16] *OCR Instantly Free*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.thesimplest.ocr

[17] *OCR - Text Scanner*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.offline.ocr.english.image.to.text

[18] *PhoneGap*. [Online; navštíveno 08.05.2016].
URL http://phonegap.com/

[19] *Quick PDF Scanner FREE*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.mobisystems.mobiscanner

[20] *Tesseract*. [Online; navštíveno 28.04.2016].
URL https://github.com/tesseract-ocr/tesseract

[21] *Tesseract - Expense Reports*. [Online; navštíveno 28.04.2016].
URL https://github.com/tesseract-ocr/tesseract/wiki/ImproveQuality

[22] *Text Fairy (OCR Text Scanner)*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.renard.ocr

[23] *Wallet – Finance a rozpočet*. [Online; navštíveno 08.05.2016].
URL https://play.google.com/store/apps/details?id=com.droid4you.application.wallet

[24] CHASTAGNOL F.: *Building an image processing pipeline with Python*. [Online; navštíveno 28.04.2016].
URL https://speakerdeck.com/pyconslides/building-an-image-processing-pipeline-with-python-by-franck-chastagnol

[25] GUPTA G.: *DatumDroid*. [Online; navštíveno 08.05.2016].
URL https://github.com/DatumDroid/DatumDroid

[26] GUPTA G.: *Making a Simple OCR Android App using Tesseract [online]*. 2011-11-09, [Online; navštíveno 02.05.2016].
URL http://gaut.am/making-an-ocr-android-app-using-tesseract/

[27] Google: *Launch Checklist*. [Online; navštíveno 02.05.2016].
URL
http://developer.android.com/distribute/tools/launch-checklist.html

[28] HRUSOVA, M.: *Češi neumí hospodařit s penězi, i když si myslí opak [online]*. 2014-09-02, [Online; navštíveno 26.04.2016].
URL http://www.nasepenize.cz/cesi-neumi-hospodarit-s-penezi-i-kdyz-si-mysli-opak-12323

[29] GÁFRIK J.: *Finanční gramotnost nadále pokulhává [online]*. 2015-03-24, [Online; navštíveno 26.04.2016].
URL http://www.denik.cz/ekonomika/financni-gramotnost-nadale-pokulhava-20150324.html

[30] Lidovky.cz: *Třetina Čechů neumí hospodařit. Neplánují, nevědí co utratí a vydělají. [online]*. 2011-02-23, [Online; navštíveno 26.04.2016].
URL http://byznys.lidovky.cz/tretina-cechu-neumi-hospodarit\-neplanuji-nevedi-co-utrati-a-vydelaji-1jo-/moje-penize.aspx?c=A110223_120057_moje-penize_nev

[31] THEIS R.: *tess-two*. [Online; navštíveno 08.05.2016].
URL https://github.com/rmtheis/tess-two

Obsah CD

Přiložené CD obsahuje následující adresářovou strukturu a položky:

- `example/` – optimalizované fotografie účtenek, ze kterých aplikace umí rozpoznat ceny
- `poster/` – vektorový plakát aplikace BillScanner
- `report/` – TEX soubory pro překlad této práce
- `report/fig/` – všechna použitá loga a obrázky
- `src/` – zdrojové soubory programové části práce
- `video/` – video s názornou ukázkou používání aplikace BillScanner
- `BP_xbambu03.pdf` – bakalářská práce ve formátu PDF
- `INSTALL.txt` – textový soubor s pokyny pro zprovoznění práce v IDE Android Studio
- `README.txt` – doplňující informace k provozu aplikace