Univerzita Hradec Králové
Fakulta informatiky a managementu

Diplomová práce

2014 Bc. Tomáš Pešek

Univerzita Hradec Králové
Fakulta informatiky a managementu
katedra informatiky a kvantitativních metod

DIPLOMOVÁ PRÁCE

Mobilní aplikace pro webový portál s recepty

Autor: Bc. Tomáš Pešek

Vedoucí práce: Ing. Pavel Kříž, Ph.D.

Hradec Králové, 2014

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a uvedl jsem všechny použité prameny a literaturu.

V Hradci Králové dne 12. srpna 2015 Tomáš Pešek

Poděkování

Rád bych poděkoval vedoucímu mé diplomové práce, panu Ing. Pavlu Křížovi, Ph.D. za obří trpělivost, pochopení, cenné rady a postřehy. Dále bych chtěl poděkovat kolegům z firmy Medoro s.r.o. a hlavně své rodině a přátelům.

Anotace

Cíl práce je navrhnout a implementovat aplikaci pro operační systém Android, která zpřístupní obsah portálu s recepty na mobilních zařízeních (telefony, tablety).

Annotation

Design and implement an application for the Android operating system, which makes the contents of the portal with recipes (including comments, scoring etc.) on mobile devices (phones, tablets).

Obsah

1 Úvod a cíl **1**

1.1 Úvod . . . 1

1.2 Cíl . . . 3

2 Internetový portál **4**

2.1 Analýza portálu . . . 7

2.1.1 User experience . . . 8

2.1.2 Grafika . . . 11

2.1.3 Bezpečnost a technické řešení . . . 12

2.1.4 Optimalizace pro mobilní zařízení . . . 13

2.1.5 Obsah . . . 13

2.2 Existující API a možné problémy při návrhu mobilní aplikace . . . 14

2.3 Shrnutí . . . 14

3 Současné aplikace **16**

3.1 Aplikace s českými recepty . . . 17

3.1.1 Miluji vaření - recepty . . . 17

3.1.2 Recepty bezplatný . . . 19

3.1.3 Apetit sezonní recepty . . . 19

3.1.4 Česká kuchařka . . . 20

3.2 Aplikace s cizojazyčnými recepty . . . 21

3.2.1 CookBook . . . 21

4 Požadavky **23**

4.1 Požadavky na mobilní aplikaci . . . 23

4.2 Požadavky na API . . . 24

5 Realizace **26**

5.1 API . . . 26

5.1.1 Technologie . . . 26

5.1.2 Přenosový formát . . . 28

5.1.3 Úloha controllerů . . . 31

5.2 Mobilní aplikace . . . 34

5.2.1 Mobilní technologie a Android SDK . . . 34

5.2.2 Technologie . . . 38

5.2.3 Realizace . . . 45

5.2.4 Uživatelské rozhraní . . . 46

5.2.5 Sociální sítě 50
5.2.6 Optimalizace pro tablety 52
5.3 Postup rozšíření aplikace 54
5.3.1 Nová verze API 54
5.3.2 Rozšíření mobilní aplikace 56

6 Závěr **58**

Přílohy **I**

Literatura **I**

1 Úvod a cíl

1.1 Úvod

V roce 2013 se dle IDC na světě prodalo na 300 milionů desktop/laptop počítačů, 200 milionů tabletů a 1 800 milionů chytrých telefonů. V roce 2014 už to bylo na 280 milionů desktop/laptop počítačů, 270 milionů tabletů a 1 900 milionů chytrých telefonů. [1] Z tohoto zjištění je jasně vidět nepoměr mezi nově zakoupenými počítači a mobilními zařízeními. Proto se dnes klade důraz na optimalizaci webových stránek pro mobilní prohlížeče a proto jsou dnes odborná fóra přeplněna dotazy ohledně vývoje pro mobilní platformy. [2] [3]

Nikomu z nás zřejmě neunikl fakt, že chytrá mobilní zařízení se stávají nedílnou součástí našich životů. V práci, ve škole, v kavárně, v parku, na ulici. Všude se setkáme s někým, kdo používá mobilní telefon, dnes už většinou chytrý mobilní telefon. Ve vybraných kruzích už se ani nemusí jednat o mobilní telefon, ale o jakékoliv jiné mobilní zařízení, ať už to jsou chytré hodinky nebo chytré brýle. Nemusíme mít po kapsách drobné, když chceme jet městskou hromadnou dopravou, zaparkovat na placeném parkovišti nebo zaplatit regulační poplatek v nemocnici - stačí mobilní telefon a buďto pošleme SMS, nebo již máme na našem chytrém zařízení nainstalovanou aplikaci, která SMS odešle po stisku jednoho tlačítka. S chytrým mobilním zařízením jsme většinou neustále připojeni k internetu, takže nám neunikne žádný pracovní e-mail, když jsme mimo kancelář, neuniknou nám aktuální zprávy z domova i ze světa, nebo nálady našich přátel na sociálních sítích. Chytrý mobilní telefon nám simuluje nejedno další zařízení. Lze ho použít jako hudební přehrávač, fotoaparát, budík, kalendář, nákupní seznam, navigaci do automobilu nebo navigaci do terénu pro oblíbenou hru geocaching, můžeme snadno kontrolovat stav našeho bankovního účtu, plánovat si svůj den v posilovně, pochlubit se s tím, kolik jsme uběhli kilometrů, nebo kam až jsme dojeli na kole. Své fotografie stiskem jednoho tlačítka pošleme rodině a přátelům, stejně tak jako video z našeho podnikového setkání.

Tato zjištění neomylně vedou k závěru, že vývoj a optimalizace pro mobilní zařízení je z dlouhodobého hlediska cesta správným směrem. Podle dat z konce

dubna se podíl mobilních internetových prohlížečů pohyboval kolem 11% [4] a vžijeme-li se do role provozovatel menšího (samozřejmě i většího) e-shopu, mělo by být naším cílem každého desátého návštěvníka, který navštíví náš obchod z mobilního prohlížeče, na našem obchodě udržet a doufat, že vytvoří objednávku a rád se k nám vrátí. Otázkou je, zda optimalizovat současný e-shop, vytvořit e-shop nový, nebo zákazníkovi nabídnout aplikaci ušitou na míru. Zaměříme-li se na mobilní platformu Android a prozkoumáme nabídku aplikací zjistíme, že větší společnosti [1] se vydaly cestou aplikace ušité na míru jejich oboru podnikání, kdežto náš oblíbený dodavatel kávy [2] si vystačil s úpravou své současné prezentace.

Oblíbenost a rozmach mobilních zařízení bezesporu vede ke zvýšené poptávce na dodavatele mobilních aplikací, či zvyšuje poptávku po optimalizaci pro mobilní prohlížeče. Pro podnikatele a investory se zde otevřelo nespočet možností, kam směřovat své budoucí vize nebo investice. Programátorům, vývojářům, analytikům a kodérům to pak přineslo velké množství výzev, kde mohou na novo uplatňovat své zkušenosti a dovednosti a oprostit se tak od zaběhlého stereotypu, nemluvě o nových pracovních příležitostech.

Právě popsání nabytých zkušeností s vývojem aplikací pro mobilní zařízení, konkrétně pro operační systém Android, je součástí praktické části této práce. Jelikož různých příruček, knížek a článků o vývoji pro operační systém Android je bezpočet [2], měla by tato práce upustit od obecného popisu základních programovacích technik, ale měla by být zaměřena již na vyšší cíle. Práce představuje různé knihovny třetích stran, které mohou při vývoji aplikací pro operační systém Android ulehčit jak samotný vývoj, tak hlavně ulehčit následnou správu a případně další rozvoj aplikace.

Bohužel, kvalita aplikací na trhu je různá. Běžnému uživateli je jedno, zda aplikaci programovala firma A, nebo B. Zda u vývoje byly použity agilní metodiky, zda je kód testovaný jednotkovými testy, zda se závislosti mezi objekty předávají podle vzoru Singleton, nebo Inversion of control, zda se data ukládají do souboru nebo do databáze, nebo zda je pro přístup k databázi použito relační mapování. Běžného uživatele zajímá to, zda mu aplikace funguje správně, že má rozumnou odezvu a zda je ovládání intuitivní a nemusí nad aplikací prosedět dlouhé minuty aby zjistil, jak se vlastně ovládá. Zadavatele aplikace na druhou stranu bude zajímat to, kolik aplikace bude stát, kdy bude hotová a zda se uživateli bude dobře používat. Na druhou stranu programátora by všechny

[1] Alza.cz, Mall.cz, CZC.cz, ...

[2] cerstvakava.cz [5]

výše zmíněné aspekty zajímat měly, jestliže to se svým řemeslem myslí vážně a jestli chce zákazníkovi prodat kvalitní produkt, který samotnému programátorovi bude dělat reklamu a naváže díky němu další pracovní příležitosti a zakázky.

1.2 Cíl

Jak bylo naznačeno v úvodu, praktickým obsahem této práce je popis vývoje aplikace pro mobilní zařízení s operačním systémem Android. Aplikace vychází z internetového portálu s recepty a nabízí většinu [3] funkcionalit, které nabízí i internetový portál a některé nové funkcionality přidává [4].

Praktické části předchází zhodnocení dosavadního portálu (které bude obsahovat i část o současném API rozhraní), který je v provozu bezmála patnáct let a s ohledem na jeho stáří nelze říci, že by stávající portál byl přizpůsoben pro mobilní zařízení. Dále praktické části předchází porovnání vytipovaných současných mobilních aplikací, které již pro zařízení s operačním systémem Android existují.

Při návrhu mobilní aplikace práce vychází ze závěrů zjištěných v zhodnocení současného webového portálu a s porovnáním již existujících aplikací. Návrh mobilní aplikace by se měl vyvarovat případným nedostatkům, které zhodnocení webového portálu přinese. Měl by se vyvarovat výtkám ze strany uživatelů současných mobilních aplikací a pokud to bude možné, návrh by měl zohlednit implementaci nebo poupravení funkcionalit, které jsou navrhovány nebo připomínkovány u již existujících aplikací. Tato část bude také pracovat s výsledky hodnocení současného API rozhraní, které sloužilo s laskavým svolením provozovatele portálu pro výukové potřeby v rámci předmětů PRO1 a PRO2 na fakultě informatiky a managementu. Pokud práce ukáže, že je současné API rozhraní nevyhovující, bude se tato kapitola zabývat také návrhem nového API rozhraní.

[3] Po domluvě se zadavatelem je vynecháno fórum a anketa

[4] Po domluvě se zadavatelem napojení na sociální sítě

2 Internetový portál

Internetový portál s recepty, ze kterého tato práce vychází, se nazývá Česká hospodyňka a nalezneme ho na adrese `http://www.ceskahospodynka.cz`.

Portál byl, podle údajů v patičce, vyvíjen v letech 2000 - 2011, čemuž odpovídá celkový vzhled, který v dnešní době může vypadat trochu neaktuálně a zastarale. Portál je horizontálně rozdělen na tři sloupce, kde v levém sloupci můžeme nalézt hlavni menu a informaci o počtu přístupů na portál [1]. Levé menu je neměnné a je uživateli přístupné po celou dobu návštěvy portálu. Pomoci tohoto menu může uživatel snadno udržet orientaci na portále, i když nové trendy rozdělení stránek nedoporučují horizontální rozdělování z důvodu krácení prostoru pro hlavní obsah [13]. Nevýhoda levého menu je v tom, že neumožňuje víceúrovňové zobrazení [2]. Tento detail by ulehčil orientaci na portále, hlavně přepínání mezi recepty, protože portál nedisponuje ani drobečkovou navigaci. Nevhodný prvek v levé části je i informace o přístupu na portál. První problém s tímto indikátorem přístupu je ten, že nezapadá do celkového vzhledu stránek. Je totiž prezentován jako tabulka s modrým záhlaví [3], všechny ostatní tabulky na portále mají ale záhlaví zelené a nejsou ohraničené. Tento detail může vypadat nepodstatně, bohužel v dnešní době produkt prodává grafické zpracování zaměřené na detaily.

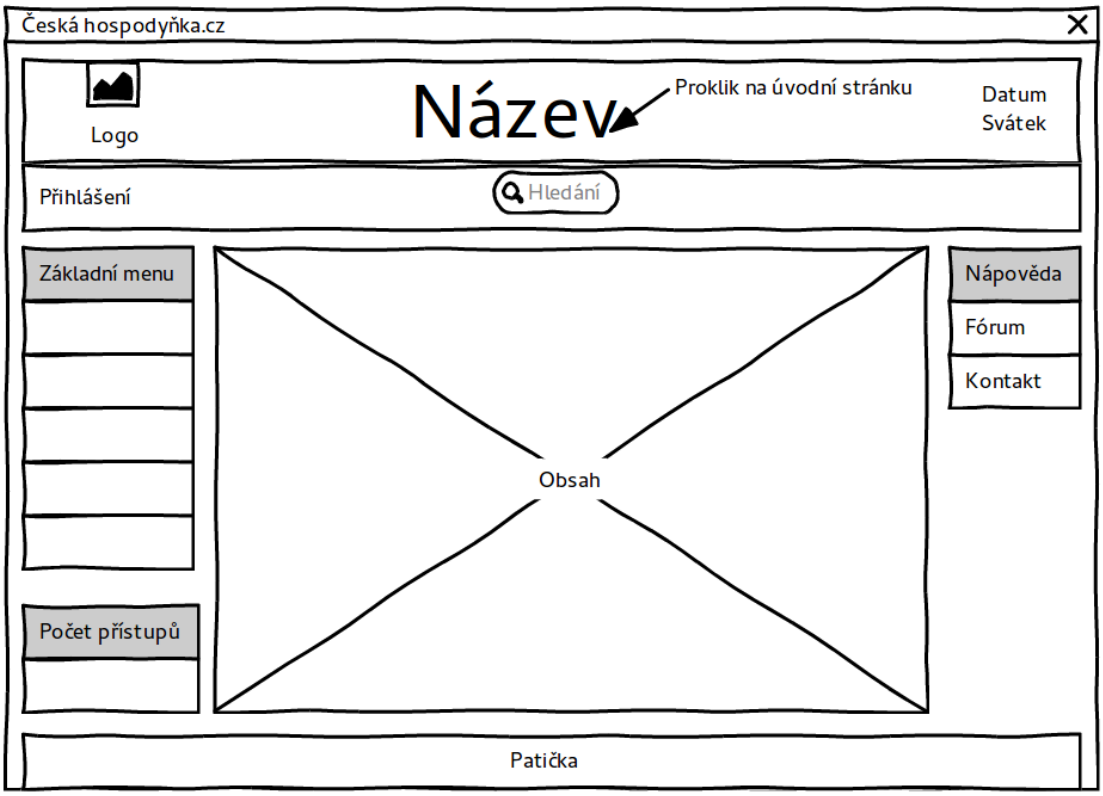
Pravý sloupec portálu trvale obsahuje tři odkazy: Nápověda, Fórum a Kontakt. Lze předpokládat, že tyto odkazy mají zjednodušovat přechod mezi částmi portálu, které mají být pro návštěvníka zajímavé nebo užitečné.

V hlavičce stránek figuruje logo portálu, které je prolinkováno s úvodní stránkou portálu, informace z kalendáře o tom, jaký je den a kdo má svátek, doplněno odkazem na sváteční recept. V liště pod hlavičkou nalezneme odkaz pro přihlášení (popřípadě prolinkem na registraci) a formulář pro základní vyhledávání. Vyhledávání v hlavičce stránky lze dnes považovat za standart a u tohoto portálu je to velice užitečná věc. Při prvním pokusu o vyhledání receptu formulář vyhledává v názvu receptu, ve výsledné stránce ale můžeme hned

[1] Infopanel služby toplist.cz

[2] hlavně u kategorií receptů je tento nedostatek dosti znát

[3] kopíruje totiž styl poskytovatele služeb



Obrázek 2.1: základní kostra portálu.

upravit parametry vyhledávání, kde můžeme vyhledávat v konkrétní kategorii, hlavní surovině nebo ve skupině. Trochu diskutabilní je vyhledávání podle čísla receptu. Nový uživatel nezná z paměti čísla receptů, ale stávající uživatelé třeba čísla receptů mají uvedené v soukromých kuchařkách a třeba jim to pomáhá v hledání.

Výše bylo zmíněno, že portál v hlavičce disponuje možností přihlášení. Po jednoduché registraci (přihlašovací jméno, heslo, reálné jméno, e-mail a kontrolní otázka sloužící k ochraně proti robotům) máme ihned privilegia registrovaného uživatele (nemusíme čekat na potvrzovací e-mail, popřípadě na schválení provozovatelem). Po přihlášení nám portál rozšíří levé menu o možností Moje recepty a Odhlášení a v hlavičce místo odkazu na přihlášení zobrazí naše uživatelské jméno [4]. Registrace na portále nám umožňuje vytvářet si seznam našich oblíbených receptů, které po té nalezneme v nabídce Moje recepty. Dále nám registrace usnadňuje možnost zanechávání komentářů, kde registrovaným uživatelům automaticky předvyplní jméno. Tímto výčet vlastností registrova-

[4] Škoda, že nezobrazí reálné jméno, když je při registraci vyžadováno

ného uživatele končí. Za hrubou chybu by bylo možno považovat fakt, že si uživatel nemůže změnit své heslo (popřípadě jméno nebo e-mailovou adresu, navíc když neslouží jako primární klíč vazby registrovaného uživatele).

Hlavní obsah portálu je závislý na typu prohlížené části portálu. Jednotlivé části mají svá specifika. Blog, Fórum, Obchůdek s tričky návštěvníka zavedou na stránky s jiným rozložením portálu. Názory, Články, Odkazy a Nápověda nás zavedou na textové stránky popisující tématiku odhadnutelnou z názvu sekce. Ostatní části portálu (Recepty, Novinky, Nejoblíbenější recepty, Hledání, Menu a Moje recepty [5]) navádějí návštěvníka na recepty, což je hlavním smyslem portálu. Různé části portálu vedou návštěvníky k receptům různou cestou. V části Recepty se k receptům dostaneme přes kategorie (bohužel seznam receptů v kategorii není stránkovaný, takže u kategorií bohatých na recepty [6] může chvilku trvat, než se stránka načte), část s Novinkami a Nejoblíbenějšími recepty nabízí rovnou tabulku s posledními patnácti (Novinky s deseti) novými, popřípadě nejoblíbenějšími recepty. V části hledání se k receptům dostaneme pomocí zadáním vyhledávacích kritérií, část Menu nabízí seznam receptů rozdělen do ucelených celků, kde v každém celku nalezneme recept na dva obědy, dvě večeře, jeden moučník, jednu polévku a jeden doplňující recept, kupříkladu na speciální nápoj. Část Moje recepty nabízí uživatelem uložené recepty na jednom místě.

Na stránce se samotným receptem uvidíme v horní části fotografii receptu, v pravém sousedství seznam surovin potřebných pro přípravu receptu. Zde je nutné zdůraznit, že seznam surovin je bohužel jen textové pole a tím pádem v pozdějším návrhu mobilní aplikace nebude možné realizovat možnost takzvaného nákupního košíku, kde by nám aplikace vygenerovala seznam surovin pro vybrané recepty. Pod fotografií a seznamem surovin nalezneme postup. Tato část je také jen prosté textové pole, takže při pozdějším návrhu mobilní aplikace nebude možné navrhnout funkci Průvodce přípravy, kde by aplikace vytvořila přesný postup stránku po stránce. Pod postupem můžeme nalézt komentáře uživatelům s možností zanechat vlastní komentář (jak bylo řečeno výše, přihlášení uživatelé mají tu výhodu, že mají předvyplněné své přihlašovací jméno.

[5] V případě přihlášeného uživatele

[6] Maso a zákusky

2.1 Analýza portálu

V cíli práce je uvedeno, že návrhu mobilní aplikace bude předcházet analýza současného portálu, z jejichž závěrů by si měl návrh vzít ponaučení a ty části, které jsou vyhovující, zachovat a naopak části, které se zdají být problémové, vylepšit.

Porovnat portál se nemusí jevit jako nijak těžký úkol, ale co jsou správná kritéria k porovnání? Programátor bude porovnávat kvalitu zdrojového kódu, grafik výsledný efekt stránek, obchodní zástupce bude porovnávat to, kolik je portál schopen vygenerovat peněz, provozovatel zase to, zda portál plní své cíle (osobní uspokojení, výdělek, prohloubení znalostí, ...). Jak tedy hodnotit portál tak, aby hodnocení bylo správně vypovídající, dalo by se porovnat s obdobnými portály a závěry byly použitelné pro následnou realizaci mobilní aplikace? Práce se inspiruje soutěží WebTop100, která hodnotí různé kategorie internetových stránek a portálů (např. kategorie Firemní web, Microsite nebo Mobilní řešení). Dle specifikací jednotlivých kategorií bychom portál zařadili do kategorie Firemní web [7], kde se za kritéria hodnocení považují:

- User experience (váha 25%)
- Grafika (váha 15%)
- Bezpečnost a technické řešení (váha 15%)
- Optimalizace pro mobilní zařízení (váha 20%)
- Obsah (váha 25%)

Díky panu Janu Kaliankovi ze společnosti Trading & Consulting s.r.o., která provozuje internetový obchod SvětBot.cz jsme měli začátkem listopadu 2014 možnost nahlédnout pod pokličku soutěže. Pan Kalianko totiž zveřejnil na svém blogu Jak dělám e-shop [8] své zkušenosti se soutěží. Sice se o soutěži nevyjadřuje úplně lichotivě, spíše naopak, díky tomuto příspěvku si ale čtenář mohl povšimnout alespoň základních principů hodnocení v jednotlivých kategoriích soutěže.

Ačkoliv autor této práce nedisponuje takovými odbornými znalostmi jako porotci soutěže, snaží se alespoň na základě svých dosavadních znalostí objektivně posoudit internetový portál hlavně pro pozdější potřeby vývoje mobilní

[7] Webová prezentace jakékoliv firmy, značky, nebo organizace v českém jazyce
[8] http://jak-delam-eshop.cz

aplikace. Při hodnocení se autor snaží brát v potaz fakt, že internetový portál s recepty není primárně určen k tomu, aby generoval zisk, ale pouze jako zájmový portál s recepty pro náhodného nebo stálého návštěvníka.

2.1.1 User experience

Pojem User experience dnes spojuje metody User experience design (UXD, EUD) [9], Interaction design (IxD) [10] a User interface (UI) [11]. Všechny tyto metody slouží k tomu, aby vývojář a designér navrhli takovou aplikaci, která bude soudružná, předvídatelná a intuitivní pro svého uživatele. Zkrátka to jsou metody, které dělají aplikace dobře použitelné pro své uživatele.

Termín User experience definoval Don Norman, svého času viceprezident pro moderní technologie ve společnosti Apple [14]. Podle jeho slov vymyslel termín User experience z důvodu příliš úzkého pojetí User interface. Ve svém termínu chtěl zahrnout veškeré aspekty zkušeností člověka používajícího softwarové produkty. Jak je uvedeno ve zdroji, v dnešní době se termín rozrostl a už začíná postrádat původní význam. Mnoho lidí termín používá, aniž by věděli, co slovo znamená a jakou má historii. Zvláště na českých fórech [12] se to pojmem User experience jen hemží, ale při detailnějším prozkoumání diskuze si opravdu můžeme povšimnout výše zmíněného názoru ze zdroje, že jen málokdo ví, co to user experience je, nebo dokonce jakou má historii.

Analýzu portálu z hlediska User experience rozebereme z pohledu orientace uživatele na úvodní stránce, na stránce s receptem a zaměříme se na to, zda je orientace na portálu pro uživatele pohodlná a jestli by nešla vylepšit. Předem ale musíme zmínit jako zásadní problém portálu to, že doména ceskahospodynka.cz není propojená se svou subdoménou www.ceskahospodynka.cz. Problém z hlediska UX je v tom, že už uživatel musí přemýšlet nad tím, co je špatně. Proč stránka neexistuje nebo nefunguje? Tento problém by se totiž mohl týkat i stávajících uživatelů portálu. Mohlo by se třeba stát, že se stávající uživatel bude chtít podělit o jeden osvědčený recept na návštěvě u známých, zadá do internetového prohlížeče adresu ceskahospodynka.cz a stránka se nenačte. Přitom před odchodem z domu portál ještě fungoval a po příchodu domů znovu funguje. Při dalším špatném zadání by ale znovu nefungoval a tak pořád dokola.

[9] Zjednodušeně Jak se uživatel cítí [15]
[10] Jak uživatelské rozhraní navrhnout [16]
[11] Samotné uživatelské rozhraní
[12] webtrh.cz, jakpsatweb.cz

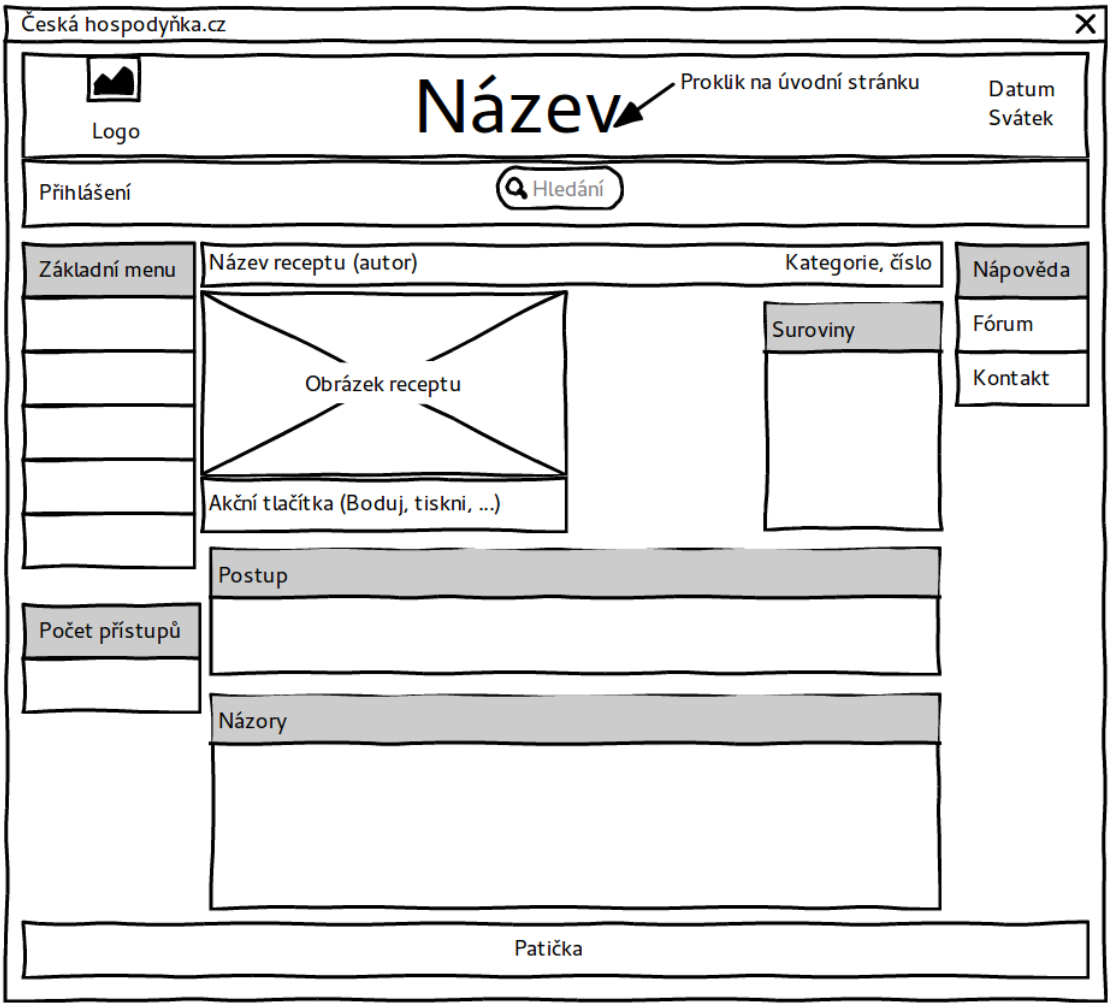
Na Obrázku 2.1 je načrtnuté základní rozložení prvků na portálu. Vidíme tří sloupcový návrh. Popis jednotlivých částí je proveden v úvodní části této kapitoly, proto ho zde nebudeme opakovat. Zde se jen zamyslíme nad využitím úvodní stránky portálu. Úvodní stránka by nám měla poskytnout jasnou vizi o tom, co nám portál nabízí a měla by nám umožnit se k tomu co nejjednodušším způsobem dostat [13]. V tomto směru má portál mezery. Na úvodní obrazovce totiž vidíme neměnný pravý a levý sloupec, ale v hlavní části s obsahem vidíme jen poslední aktualitu. V tomto případě by bylo vhodnější použít pro úvodní stránku buď stránku s kategoriemi receptů, nebo navrhnout úvodní stránku tak, aby stručně zvýraznila a představila jednotlivé položky levého menu.

Menu v levém sloupci nezvýrazňuje stránku, na které se uživatel nachází. Tím se dostáváme k problematice orientace uživatele na portálu: portál nedisponuje drobečkovou navigací. Bez drobečkové navigace se uživateli špatně orientuje na stránce. Když se totiž uživatel prokliká k receptům seřazených podle kategorie (nebo i nejoblíbenější recepty, nejvyhledávanější recepty, menu) vždy při prohlížení detailu receptu ztratí informaci o tom, jak se na konkrétní recept dostal. Uživatel je v tomto případě odkázán pouze na tlačítko Zpět v internetovém prohlížeči. Když si totiž uživatel bude prohlížet recept Pstruh po kanadsku, portál nám žádným způsobem nedá najevo, jaký krok předcházel volbě receptu. Uživatel se mohl na recept proklikat přes menu s názvem Začíná léto, nebo přes kategorii Maso, nebo pomocí vyhledávání. Ve všech třech případech bychom mohli předpokládat, že se uživatel bude chtít vrátit na předchozí krok (prohlédnout si zbylé recepty v menu, jiné recepty v kategorii nebo další výsledky vyhledávání). V momentě prohlížení je uživatel zachráněn zmíněným tlačítkem Zpět, ale v momentě, kdy si recept uloží, nebo někomu pošle už tlačítko zpět nebude fungovat. Další velká škoda je, že levé menu není víceúrovňové. Minimálně u první položky - Recepty by nebylo vůbec na škodu, kdyby v druhé úrovni menu byly kategorie receptů vypsány a ideálně zvýrazněna ta kategorie, ze které máme vybraný recept.

Na stránce s konkrétním receptem, jejíž náčrt si můžeme prohlédnout na obrázku 2.2, trochu zapadá název receptu. U akčních ikon by nebylo na škodu, kdyby obsahovaly atribut title, ve kterém by bylo trochu detailněji popsáno, co ikona dělá. U akce Boduj je podivné, že můžeme jeden recept obodovat vícekrát [14]. U akce Odešli je pozitivní fakt, že uživatele odkaz neodvede do exter-

[13] Bez žádného zbytečného klikání nebo přemýšlení

[14] Při testování nebyla zjištěna žádná maximální hodnota bodů, které může jeden uživatel k jednomu receptu přiřadit



Obrázek 2.2: Detail receptu.

ního e-mailového klienta, ale na stránku, kde můžeme rovnou zadat e-mailovou adresu příjemce, krátký popis a portál recept odešle. Nevýhoda by mohla být v tom, že v odeslaném e-mailu nenajdeme přímý odkaz na odeslaný recept - tím se portál okrádá a potenciální uživatele [15] a příjemci trochu znepříjemňuje možnost prohlédnout si stránku s receptem, kde jsou mimo jiné názory ostatních návštěvníku, které v přijatém e-mailu nenajde. Velmi pěkná vlastnost u akce Tisk je možnost dvou variant tisku - totiž grafickou a černobílou variantu. Jako poslední nedostatek na detailu receptu bychom měli zmínit menší nelogičnost v odesílání komentářů pro přihlášené uživatele. Nepřihlášený uživatel se pod komentářem musí podepsat a dokázat, že není takzvaný robot, který navštíveným stránkám škodí tak, že automaticky vkládá komentáře s reklamou na různé stránky, které navštíví. Tento krok je pochopitelný, co ale pochopitelné není je to, že i přihlášený uživatel musí prokazovat, že není robot [16].

I přes výše zmíněné nedostatky se ale na portálu dá lehce zorientovat a jednoduše se používá. Pro další zkoumání UX by nebylo špatné oslovit přímo návštěvníky portálu a pokusit se zjistit, co jim na portálu činní největší potíže (pokud nějaké mají).

Závěr

Rozložení portálu je jednoduché, neměnné a logické. Portál ale nedisponuje prvky, které by návštěvníkovi zpříjemnily orientaci na stránce (drobečková navigace, zvýrazněná kategorie, kterou si návštěvník prohlíží). Potenciál úvodní stránky je nevyužitý a doména není svázána se svou subdoménou www.

2.1.2 Grafika

Na první pohled je zřejmé, že portál nedisponuje žádnou oslnivou grafikou, která by byla podpořena novodobými trendy. Z grafiky by šlo soudit, že portál je zaměřen na své stálé návštěvníky, kteří jsou už léta zvyklí na jeden styl. Kdybychom se pokusili oprostit od zastaralé grafiky, jako největší problém by mohlo být vytknuto to, že portál žádným způsobem na první pohled nereprezentuje svůj hlavní cíl, tudíž nabídnout uživateli bohatou databázi receptů. Úvodní stránka je v tomto pohledu úplně promarněný potenciál, místo toho,

[15] Dlužno říci, že odkaz na samotný portál v e-mailu je

[16] Minimálně to dokázal už tím, že se přihlásil

aby nabídla zajímavé rozcestí kategorií receptů, obrázky nejoblíbenějších a nejvyhledávanějších receptů nebo náhodné obrázky z menu nabízí pouze jednu poslední aktualitu.

Stejně tak tabulkové seznamy receptů, které obsahují strohé informace o názvu receptu, hlavní surovině, počtu bodů, autorovi, skupině a počet komentářů. Namísto náhledu vidíme v tabulce jen informaci, zda recept obsahuje nebo neobsahuje fotografii.

Samotná stránka s receptem už je, v rámci mezí, zajímavá minimálně tím, že si recept můžeme prohlédnout na fotografii a můžeme se na stránce jednoduše orientovat mezi jednotlivými, pro návštěvníka důležitými, částmi stránky, jako je vedle zmíněné fotografie box s potřebnými surovinami a postup přípravy receptu.

Celkově nelze pochybovat o tom, že by portálu prospěl celkový redesign. Na druhou stranu je nutno pochválit fakt, že se portál drží jednotného stylu a neexperimentuje s žádnými křiklavými barvami, nebo dokonce pohyblivými animacemi u každého nadpisu.

Závěr

Při pohledu na portál je zřejmé, že grafické zpracování nedisponuje žádnými novodobými trendy (stačí porovnat s některými známějšími portály o vaření).

2.1.3 Bezpečnost a technické řešení

Portál bezpečně obstál v jednoduchých i složitějších pokusech o provedení útoku pomocí SQL injection a Cross site scripting. Když ale registrovaný uživatel zapomene heslo, nemá jinou možnost než se obrátit na provozovatele e-mailem. Dále si registrovaný uživatel nemůže změnit heslo a přihlašování není chráněno šifrovaným přenosem. V minulosti se tato problematika dost opomíjela, ale doba se změnila a i když pořád můžeme slyšet názory typu 'Na našich stránkách přece nemáme uložené žádné citlivé údaje o našich klientech' problém je v tom, že už samotné uživatelské jméno a heslo je citlivý údaj a to, že na jednom e-shopu nebo portálu citlivé údaje neukládáme, neznamená to, že návštěvník se pomocí těch samých přihlašovacích údajů nepřihlašuje na jiný portál nebo službu, kde už citlivější údaje (jako například datum narození, čísla platebních karet) uloženy být mohou.

Navíc v poslední době vznikají iniciativy, které přístup k ověřeným certifikátům usnadňují a zlevňují. Společnost Simplia s.r.o. uvádí, že poskytují nejlevnější SSL certifikáty od důvěryhodných autorit [22]. Jestliže nás ale problematika certifikátů opravdu zajímá, tak můžeme nalézt společnosti jako například společnost StartSSL, které nabízejí doménové certifikáty zcela zdarma [23] [17].

Dalším zajímavou iniciativou v rámci snazšího a levnějšího zpřístupnění důvěryhodných certifikátů pro běžné internetové prezentace, obchody a portály může být projekt Let's Encrypt, který nová, stejnojmenná, certifikační autorita plánuje spustit v půli září roku 2015. Autorita by měla nabízet doménové certifikáty zdarma, navíc se softwarem pro jejich snadnou správu. Za touto iniciativou stojí silné společnosti, jako například Cisco nebo Mozilla [24].

Závěr

Portál je odolný proti základním útokům SQL injection a Cross site scriptingu, bohužel ale není provozován na zašifrovaném protokolu SSL, takže kdyby se návštěvník chtěl přihlásit do svého účtu někde na volné síti, kdokoliv by si mohl odposlechnout jeho přihlašovací údaje v čitelné podobě.

2.1.4 Optimalizace pro mobilní zařízení

K této kategorii není co dodat. Portál není nijak optimalizován pro mobilní zařízení. Nápravou tohoto nedostatku se zabývá tato práce.

2.1.5 Obsah

Podle údajů z databáze obsahuje portál s recepty okolo 2 000 receptů, což je opravdu dobrý výkon. Je to jednoznačně nejsilnější stránka portálu. Jako dobrý nápad lze považovat sekci Menu, která návštěvníkům nabízí vybrané recepty na celý víkend, škoda jen, že počet takovýchto menu je vzhledem k počtu receptů tak nízký (okolo dvaceti položek menu). Dalším dobrým nápadem je sekce s Články, kde se návštěvník může dozvědět sem tam nějakou zajímavost. Portál tato sekce obohacuje o obsah, který není nijak spojen s konkrétními recepty.

[17] Jedná se o certifikáty první třídy

Závěr

Obsah je jednoznačně nejsilnější stránka portálu. U většiny receptů je navíc vložena i fotografie.

2.2 Existující API a možné problémy při návrhu mobilní aplikace

Pro potřeby výuky předmětů PRO1 a PRO2 na Univerzitě Hradec Králové bylo historicky vytvořeno jednoduché API, které poskytovalo nejzákladnější informace z portálu v podobě XML výstupu. Toto API je bohužel pro potřeby práce nepoužitelné a v práci musí být implementováno API nové. Současné API totiž neposkytuje základní informace, jako je seznam kategorií, možnost exportu Menu, Článků a jiných částí portálu.

Při zkoumání databázové části portálu bylo zjištěno, že některé databázové tabulky a vazby mezi nimi nesplňují třetí normální formu. Jedná se bohužel i o tabulky, které uchovávají informaci o postupu přípravy receptu a o tabulky, které uchovávají informaci u použitých surovinách. Tento fakt v návrhu mobilní aplikace znemožňuje zahrnout do implementace funkce, které by uživateli z vybraných receptů vygenerovaly nákupní košík, nebo funkce, které by uživatele krok po kroku provedli přípravou receptu.

2.3 Shrnutí

Portál Česká hospodyňka by měl v tomto roce slavit 15 let provozu, což je zajisté úctyhodný výkon. Většina problémů, které byly v této kapitole zmíněny, vychází z pokroku informačních technologií, který není zrovna pomalý. Portál sám o sobě nic neprodává a nenajdeme na něm komerční reklamu. Na druhou stranu na něm nalezneme rozsáhlou databázi receptů s funkcemi, které před patnácti lety nebyly úplně běžné. Výtky ohledně UX a grafiky jsou pro potřeby portálu spíše okrajové. Portál nemá ambice lákat nové masy uživatelů. Kdyby takové ambice měl, spíše by se práce měla věnovat redesignu portálu. Čím by ale portál mohl oplatit důvěru svých stálých zákazníků je to, že by pro ně zařídil šifrovaný provoz. Tento problém je bohužel ještě dnes bagatelizován, ale bohužel rizika dnes jsou a nikdy nevíme, na koho na internetu narazíme [17].

Ze závěrů jednotlivých částí portálu jsou pro další část této práce podstatné hlavně tyto nedostatky:

- Nabídnout lepší orientaci na stránce (zobrazit nebo zvýraznit minimálně prohlíženou část portálu)
- Využít potenciál úvodní stránky
- Zabezpečit přenos dat a hlavně přenos citlivých údajů návštěvníka

3 Současné aplikace

Práce se zaměřuje na vývoj aplikace pro mobilní zařízení s operačním systémem Android.

V přehledu současných aplikací se nyní zaměříme pouze na aplikace, které můžeme najít v databázi Play store, což je jakýsi obchod, kde můžeme jednoduše nakoupit aplikace pro naše zařízení, ať už se jedná o telefon, tablet, hodinky nebo jiný přístroj, který disponuje operačním systémem Android. Nalezneme zde i bezplatné aplikace, kterých je převaha.

Základní dělení aplikací na Play store pro potřeby práce bychom mohli rozdělit na aplikace, které jsou zdarma a na aplikace, které nabízejí české recepty. Ve druhém rozdělení můžeme ale narazit na problém. Při vydávání aplikací na Play store společnost Google přímo nabízí překlad aplikace do konkrétních jazyků, což se na první pohled může jevit jako skvělá vlastnost, po procitnutí ale zjistíme, že aplikace je opravdu česky, bohužel obsah už nikoliv a jediný způsob, jak od sebe rozlišit aplikace s českým a s cizojazyčným obsahem je takový, že to musí uživatel vyzkoušet, popřípadě odhadnout na základě komentářů [1].

Po dalším prozkoumání Play store zjistíme, že by dále šlo aplikace dělit na aplikace podle různých kategorií receptů jako například Recepty pro vegetariány, Recepty pro maminky s dětmi nebo Recepty pro studenty. Z cizojazyčných aplikací například Mexican Recipes, Italian Recipes nebo Diabetic Recipes. První dvě zmíněné cizojazyčné aplikace vytvořila společnost Riafy Technologies a spolu s aplikací s nespecifickými recepty, Cookbook, se jedná opravdu o špičkové aplikace, aplikaci Cookbook se budeme detailněji věnovat v následujících částí této kapitoly, protože při praktické části této práce posloužila jako jakási pomyslná meta, kterou se práce snažila docílit [2].

Jako poslední možností dělení aplikací musí být zmíněno, že aplikace lze dělit podle zařízení, pro která jsou určena (konkrétně verzí SDK, kterým disponuje naše zařízení) a technicky zdatnější uživatele by mohlo zajímat, jak je aplikace

[1] Výjimkou aplikací, které ve svém jméně mají textovou informaci typu české recepty nebo recepty česky

[2] 8.12.2014 společnost El Toro s.r.o. vydala aplikaci Miluji Vaření - recepty, která svými kvalitami v oblasti zpracování aplikace Cookbook dohání a s přihlédneme-li na fakt, že recepty jsou české, tak nejspíš i předhání

velká (kolik zabere místa v zařízení) [3].

Popsat současné aplikace na trhu se může jevit jako lehký úkol, prohlédneme-li si ovšem nabídku aplikací v Play store, názor změníme. Za prvé musíme řešit stejný problém, jako jsme řešili s porovnáním současného portálu - jak určit kvalitu aplikace? A za druhé, které aplikace porovnat? Po chvilce prozkoumávání Play store a zkoušením různých vyhledávacích frází [4] zjistíme, že aplikací zaměřující se na recepty je docela dost. Rozhodně se nejedná o jednotky. Pro zúžení výběru byly použity všechny restrikce, které Play store nabízí, čili byly vybrány jen aplikace, které jsou poskytovány bez poplatku a aplikace, které jsou hodnoceny čtyřmi a více hvězdičkami [5]. I přes tyto restrikce nám Play store nabízí větší množství aplikací, než je možné v jedné kapitole popsat. Proto konkrétní české a cizojazyčné recepty jsou vybrány na zralé úvaze autora. Výběr byl zohledněn počtem hodnocení, počtem instalací a též podle názoru uživatelů, kteří aplikaci používají.

U jednotlivých aplikací se práce zaměřuje na následující faktory:

- Výsledky hodnocení uživatelů na Play store [6]
- Počet receptů
- Rozdělení receptů (jak je složité se proklikat k receptu)
- Forma přístupu k receptům (online nebo offline)
- Obsah reklamy
- Minimální verze Android
- Velikost aplikace

3.1 Aplikace s českými recepty

3.1.1 Miluji vaření - recepty

Už na první pohled [7] je jasné, že se jedná o profesionální aplikaci na vysoké úrovni, navíc s velmi kvalitním obsahem, který vychází z webového portálu

[3] Starší zařízení s operačním systémem Android nabízeli velikost vnitřní paměti v řádu desítek MB
[4] Jako například kuchařka, recepty, vaření, ale i jména známých magazínů apetit nebo gurmet
[5] Maximální počet hvězdiček je pět, minimální jedna
[6] Průměrný počet hvězdiček (1-5) a počet hodnotících
[7] Před nainstalováním

Hodnocení na Play store	4,1 z 480
Počet receptů	téměř 60 000
Rozdělení receptů	Jedna úroveň kategorie
Online x Offline	Online
Obsah reklamy	Ano - pruh ve spodní části aplikace
Minimální verze Android	2.3
Velikost aplikace	4.3 M

Tabulka 3.1: Shrnutí aplikace Miluji vaření - recepty

www.milujivareni.cz. V popisu aplikace na Play store zjistíme, že aplikace disponuje úctyhodným počtem receptů - téměř 60 000, počet instalací se pohybuje v rozmezí od 10 - 50 tisíců, velikost aplikace je 4,3 MB a že uživatelé hodnotí tuto aplikaci průměrným počtem 4,1 hvězdiček. Po letmém prohlédnutí komentářů zjistíme, že uživatelé bez výjimek hodnotí aplikaci kladně, někteří zmiňují jen problém s přihlášením.

Po instalaci se aplikace uživatele zeptá, zda netrpí některou z uvedených diet (jako například dieta pro diabetiky, nebo pro uživatele, kteří trpí celiakií, čili bezlepková dieta) a recepty, které odporují vybrané dietě nenabídne v žádném výsledku vyhledávání. Aplikace seskupuje recepty do sedmi základních kategorií a dále už recepty nerozděluje, stačí si tedy vybrat kategorii a hned vidíme seznam receptů s obrázkem, s hodnocením, s informací o počtu kalorií, s informací o tom, zda recept obsahuje video-návod a autorem receptu. Na stránce s detailem receptu vidíme v horní části fotografii receptu, pod fotografií vidíme detaily přípravy, jako doba přípravy, doba vaření, počet osob, pro který je recept zadán a odkaz na nutriční tabulku. Následuje seznam ingrediencí, postup, hodnocení receptu, možnost přidání fotografie [8] a seznam komentářů. Spodní část aplikace je určena pro reklamní plochu.

Vzhledově i ovladatelností se aplikace dosti podobá výše zmíněné aplikaci Cookbook. Velkou výhodou této aplikace je obsah, který umožňuje vyřadit z výsledků recepty odporující vybrané dietě, rozdělit seznam surovin na položkový seznam a rozdělit postup přípravy receptu na jednotlivé kroky.

[8] Mimo jiné navázáno na zajímavou promo akci - za přidání fotografie k receptu získáváte body, kterými lze uplatnit slevu na smluvním e-shopu

3.1.2 Recepty bezplatný

Aplikace od ruské vývojové společnosti MSV development nás může zmást svým krkolomným názvem i popisem v obchodě Play store. Ne jeden uživatel si také může myslet, že se jedná o cizojazyčné recepty v aplikaci, která obsahuje pouze strojový překlad aplikace. Po prozkoumání hodnocení a komentářů (hodnocení 4,2 hvězdičky) ale zjistíme, že aplikace je vcelku oblíbená a doporučována a až na výjimky je uživateli hodnocena kladně. Počet instalací je obdobný jako u předchozí aplikace, čili v rozmezí 10 - 50 tisíc instalací.

Po prvním spuštění aplikace musíme nejprve stáhnout celou databázi receptů [9]. Aplikace uživateli nabídne možnost, zda data stáhnout na vnitřní, nebo vnější paměť zařízení a doporučí připojení k síti WiFi. Nevýhodou je, že stahování a následné rozbalování stažených dat je i s vysokorychlostním připojením k internetu časově náročné a ani stažení, ani rozbalení nelze spustit v pozadí. Matoucí je, že po instalaci a následném rozbalení nás aplikace nepřepne na aktivitu s recepty, ale nabídne nám znovu stažení dat, což po několikaminutovém čekání na stažení a instalaci může vyvolat nepříjemný pocit bezmoci, když ale stiskneme hardwareové tlačítko zpět, aplikace už nám nabídne hned možnost výběru kategorie (je jich celkem patnáct), po výběru kategorie se nám zobrazí abeceda, kde musíme vybrat začínající písmeno receptu a až po té se nám zobrazí seznam receptů v kategorii, začínající na námi vybrané písmeno. Detail receptu obsahuje v tomto pořadí: Fotografii receptu, možnost přidat recept k oblíbeným, postup přípravy a ingredience. Nenalezneme zde hodnocení receptů ani komentáře, za to aplikace disponuje historií prohlížených receptů.

Vzhledově se jedná spíše o podprůměrnou aplikaci, která své uživatele láká hlavně na obsah, než na grafickou podobu a uživatelsky přívětivé ovládání. Nutnost výběru receptu v kategorii podle abecedy je zbytečný krok navíc, nicméně uživatelé si na něj neztěžují, z čehož bychom mohli soudit, že opravdu důležitý je obsah a ne podoba. Bohužel není nikde uvedeno, kolik receptů aplikace obsahuje, soudě ale podle velikosti stažené databáze se lze domnívat, že se může jednat určitě o množství v řádu tisíců.

3.1.3 Apetit sezonní recepty

Aplikace Apetit sezónní recepty vychází z internetového portálu s recepty apetitonline.cz, který provozuje společnost BURDA Praha, spol. s.r.o., která je, mimo jiné, nakladatelem stejnojmenného měsíčníku s recepty - Apetit. Aplikace na-

[9] kolem 100 MB dat

Hodnocení na Play store	4,2 z 263
Počet receptů	nelze zjistit
Rozdělení receptů	Dvě úrovně, první kategorie, druhá abeceda
Online x Offline	Offline
Obsah reklamy	Ano - pruh pod toolbarem [10]
Minimální verze Android	2.3
Velikost aplikace	8 M (+ 100MB offline databáze receptů)

Tabulka 3.2: Shrnutí aplikace Recepty bezplatný

bízí v základní verzi (bez příplatků) okolo čtyř stovek receptů rozdělených do čtyř sezón (jaro, léto, podzim a zima), kde se v sezóně dále aplikace dělí na již klasické kategorie, které jsou voleny s přihlédnutím na vybranou sezónu (např. v sezóně Jaro nalezneme kategorii Jarní dezerty. Tuto aplikaci si, stejně jako obě předchozí, nainstalovalo od 10 000 - 50 000 uživatelů, 345 uživatelů jí hodnotilo s průměrným hodnocením 3,9 hvězdičky. Aplikace podporuje minimální verzi systému Android 2,1 a její velikost je 13 MB.

Po instalaci aplikace stáhne recepty do paměti telefonu(zmíněných 400 receptů). Při výběru sezóny se aplikace zeptá, zda si uživatel přeje stáhnout obrázky k receptům pro pozdější offline použití. Recepty lze po instalaci používat bez nutnosti připojení k internetu, jen bez obrázků. Po potvrzení tohoto dialogu bude uživatel moci v konkrétní sezóně používat i obrázky k receptům offline. V aplikaci si uživatel může jednoduše zvětšit/zmenšit písmo bez nutnosti hledání nastavení a opouštění tak stránky s receptem a dále si může vytvářet nákupní seznam. U konkrétních receptů chybí komentáře ostatních uživatelů. Vzhled aplikace působí velice profesionálně a aplikace se příjemně používá. Možnost dokupování receptů je navázána přímo na Play store, takže otázka rozšíření receptů není nikterak složitá a zabere zlomek času. Jelikož je aplikace určena k nákupu receptů, je celkem logické, že neobsahuje reklamu.

3.1.4 Česká kuchařka

Zajímavostí aplikace je, že vyšla v momentě, kdy začala vznikat tato diplomová práce. Podle funkcionalit, které aplikace nabízí, lze usuzovat, že aplikace má k dispozici obdobnou strukturu dat, jako Česká hospodyňka. Tato aplikace má jako svůj zdroj dat portál s recepty nezapomenete.cz. Aplikace nabízí jen některé recepty z portálu a soudě podle autora, spíše slouží jako reklama na zmí-

Hodnocení na Play store	3,9 z 345
Počet receptů	400 receptů v základní nabídce
Rozdělení receptů	Dvě kategorie, první sezóna
Online x Offline	Offline (bez obrázků)
Obsah reklamy	Ne (více receptů za příplatek)
Minimální verze Android	2.1
Velikost aplikace	13 M

Tabulka 3.3: Shrnutí aplikace Apetit sezonní recepty

něný portál, jelikož u každého detailu receptu nalezneme odkaz na portál na který když klikneme, otevře se nám portál v komponentě WebView. Aplikaci taktéž nainstalovalo 10 000 - 50 000 uživatelů a 502 uživatelů jí průměrně hodnotilo 3,8 hvězdičkami. Poslední komentáře se o aplikaci nelichotivě vyjadřují kvůli obsahu.

Po instalaci a spuštění aplikace se rovnou otevře levý panel nabízející menu aplikace, kde nalezneme všechny kategorie receptů s informací, kolik receptů si uživatel v dané kategorii ještě neprohlédl. Nikde není popsáno, které recepty aplikace nabízí. Zda jsou to poslední recepty, náhodné recepty nebo recepty určené pro mobilní aplikaci. Podle data zveřejnění, které je u receptu zobrazené, by bylo možno soudit, že se jedná a poslední vložené recepty. Recepty mohou uživatelé sdílet přes různé kanály (např. SMS, GMail, e-mailový klient). Sdílení probíhá tak, že se příjemci odešle odkaz na recept na výše zmíněném portálu. Aplikace také umožňuje ukládat jednotlivé recepty jako oblíbené. V aplikaci ani na portálu není nikde popsán způsob, jak v aplikaci prohlížet více receptů bez nutnosti prohlížet portál v komponentě WebView, což aplikaci dosti znevýhodňuje.

3.2 Aplikace s cizojazyčnými recepty

3.2.1 CookBook

Aplikace Cookbook byla zmíněna v úvodu této kapitoly. Z cizojazyčných aplikací to je jediná aplikace, která je v této práci zmíněna. Google play udává, že si aplikaci nainstalovalo jeden až pět miliónů uživatelů a ohodnotilo jí pře 12 000 uživatelů, průměrné hodnocení je 4,1 hvězdičky. Rozdíl v číslech mezi českými aplikacemi je způsoben tím, že aplikace CookBook je distribuovaná na širším

Hodnocení na Play store	3,8 z 502
Počet receptů	desítky, online celý portál
Rozdělení receptů	jedna úroveň
Online x Offline	Offline jen pár receptů, zbytek online v komponentě WebView
Obsah reklamy	Ve spodní části aplikace, odkaz na sponzora, u jednotlivých receptů
Minimální verze Android	2,3
Velikost aplikace	4 M

Tabulka 3.4: Shrnutí aplikace Česká kuchařka

Hodnocení na Play store	4,1 z 12 126
Počet receptů	?
Rozdělení receptů	Jedna hlavní kategorie
Online x Offline	Online
Obsah reklamy	Panel na spodní části a náhodně reklama přes celou obrazovku
Minimální verze Android	2,3
Velikost aplikace	3,6 MB

Tabulka 3.5: Shrnutí aplikace Cookbook

trhu. I přes to, že aplikace neobsahuje české recepty, několik česky mluvících uživatelů ji používá a hodnotí ji ve směs kladně.

Aplikace funguje pouze online, je rozdělena do základních kategorií, které se v závislosti na období mění [11]. Po výběru kategorie už se rovnou dostaneme na seznam receptů. Seznam receptů obsahuje velkou fotku receptu, název receptu, dobu přípravy, tagy receptu a tlačítko pro uložení receptu mezi oblíbené recepty, které ale nejsou přístupné offline.

Konkrétní recept pak umožňuje ze surovin vytvářet nákupní košík, obsahuje informace o nutričních hodnotách, době přípravy, možnost sdílení receptu a samotný postup přípravy receptu.

[11] Například v únoru se na úvodní stránce s kategoriemi objevila kategorie Valentines day special

4 Požadavky

4.1 Požadavky na mobilní aplikaci

Jak bylo zmíněno v úvodní kapitole, cílem práce je vytvořit mobilní aplikaci pro mobilní zařízení s operačním systémem Android, která bude obsahovat až na dvě výjimky veškeré funkcionality současného webového portálu. Mobilní aplikace po dohodě se zadavatelem nebude veškeré recepty ukládat do paměti telefonu, aby šly recepty načíst i bez nutnosti připojení k internetu, ale bude ukládat jen oblíbené recepty. Ostatní informace budou k dispozici pouze online prostřednictvím webového API.

Mobilní aplikace by měla:

- Nabídnout seznam článků
- Nabídnout konkrétní článek
- Nabídnout seznam kategorií
- Nabídnout seznam odkazů
- Nabídnout seznam menu
- Nabídnout poslední názory
- Nabídnout názory k receptu
- Nabídnout seznam receptů pro vybranou kategorii
- Nabídnout hledání receptů
- Nabídnout konkrétní recept
- Nabídnout nejnovější recepty
- Nabídnout nejnavštěvovanější recepty
- Nabídnout nejlepší recepty

- Umožnit uložení recept
- Umožnit obodovat recept
- Umožnit sdílet recept
- Umožnit uložit názor k receptu
- Umožnit přihlášení do portálu (i pomocí sociálních sítí Facebook a Google+)
- Umožnit synchronizovat oblíbené recepty i mezi přihlašovacími údaji ze sociálních sítí

Ukládání receptu by mělo fungovat tak, že aplikace uloží recept do paměti telefonu pro pozdější použití bez nutnosti připojení k internetu a pokud je uživatel přihlášen (nezáleží na tom, jestli pomocí Facebooku, Google+, nebo pomocí API přímo na portál), odešle požadavek na API s žádostí o uložení konkrétního receptu mezi uživatelovi oblíbené recepty. Tímto aplikace docílí toho, že si přihlášení uživatelé mohou synchronizovat své oblíbené recepty napříč webovým portálem a mobilním zařízením (popřípadě mezi více mobilními zařízeními).

Návrh mobilní aplikace by měl vzít v potaz veškeré nedostatky, které byly shrnuty v kapitole 2 na stránce 4. Hlavně části, které se týkají UX, tedy navrhnout úvodní stránku, která vužije svého potenciálu a nabídne co největší počet funkcí aplikace ihned po spuštění aplikace, a postarat se o to, aby bylo pro uživatele jednoduché se v aplikaci lehce zorientovat.

Poslední požadavek ze strany zadavatele je přizpůsobit mobilní aplikaci pro tablety.

4.2 Požadavky na API

Základní službou API je poskytování dat a provádění úpravy dat bez toho, aby mobilní aplikace přímo přistupovala k datům webového portálu. API musí být schopné poskytnou veškeré informace, které bude chtít mobilní aplikace nabídnout uživateli. Jsou to tyto informace:

- Seznam článků
- Konkrétní článek
- Seznam kategorií

- Seznam odkazů
- Seznam menu
- Poslední názory
- Názory k receptu
- Seznam receptů pro vybranou kategorii, s možností řazení a omezení počtu řádků a přeskočení počtu řádků
- Hledání receptů s možností omezení počtu řádků a přeskočení počtu řádků
- Konkrétní recept
- Nejnovější recepty
- Nejnavštěvovanější recepty
- Nejlepší recepty
- Uložit oblíbený recept
- Obodovat recept
- Uložit názor k receptu
- Přihlášení do portálu
- Synchronizovat oblíbené recepty i mezi přihlašovacími údaji ze sociálních sítí
- Obrázek receptu

Navíc API musí nabídnout stejné služby se sociálními sítěmi, jaké má nabízet mobilní aplikace, takže součástí API je i napojení na sociální sítě Facebook a Google+ a synchronizace účtů.

Poslední vlastnost, kterou práce musela při návrhu API řešit, je snadná rozšiřitelnost a zpětná kompatibilita. V budoucnu by mohl vzniknout požadavek na rozšíření mobilní aplikace, který by s sebou nesl i požadavek na rozšíření API. Případná úprava by samozřejmě zahrnovala úpravu na straně API i na straně mobilní aplikace, může se ale stát, že uživatelé mobilní aplikace z jakéhokoliv důvodu aplikaci nezaktualizují a aplikace bude vyžadovat staré funkce API a API tyto funkce musí být schopné pokaždé nabídnout [1].

[1] Minimálně dokud bude ve statistikách zařízení, které bude staré API využívat

5 Realizace

Konečně se práce dostává do fáze popisu návrhu a realizace aplikace, což je hlavní náplň praktické části této práce. Návrh a realizace probíhaly paralelně ve dvou částech - návrh a realizace mobilní aplikace a návrh a realizace API. V jednotlivých částech této kapitoly jsou popsány jednotlivé kroky realizace.

5.1 API

API je webová aplikace, pomocí které komunikuje mobilní aplikace s portálem. Komunikuje pomocí bezstavového protokolu HTTP, který funguje na principu request - response, čili požadavek - odpověď. Mobilní aplikace pošle API požadavek a API vrátí odpověď v podobě dat, o které mobilní aplikace požádala, nebo výsledek operace, o kterou mobilní aplikace usilovala (například bodování receptu).

5.1.1 Technologie

Při výběru technologie pro vývoj nového API byly hlavním rozhodujícím kritériem technologie, na kterých je postaven webový portál, tedy na technologii Java EE. Pro potřeby tak jednoduchého API, které je potřebné pro mobilní aplikaci, může použití Javy EE budit rozpaky. Bylo by ale nesystematické a na správu náročné starat se o více webových technologií.

Jelikož je potřeba, aby API komunikovalo s databází, je vhodné rozdělit jednotlivé části aplikace na samostatné jednotky, aby zásah do jedné z nich neovlivnil ostatní. Konkrétně se jedná o rozdělení aplikace na část, která se stará o získávání dat, na část, která řídí tok dat a na část, která data prezentuje. Tohoto cíle je dosaženo použitím softwarové architektury MVC[1]. V práci je ovšem část reprezentující data zjednodušená - nabídne pouze výstupní formát přenesených dat[2].

[1] Model - View - Controller

[2] S výjimkou části, kde API pracuje se sociálními sítěmi

V následující části budou krátce představeny dva projekty, které vývojářům usnadňují práci s vývojem pro platformu Java EE.

Spring framework

Spring framework představil v roce 2003 Rod Johnson [21]. Framework usnadňuje vývoj v mnoha ohledech, v této práci je to hlavně výhoda implementace MVC architektury, snazší přístup k datům pomocí JDBC a v neposlední řadě zjednodušení konfigurace enterprise projektu, mimo jiné díky použití návrhového vzoru Inversion of Controll. Spring framework nabízí další možnosti, jak usnadnit vývoj a jeho obrovská výhoda je, že jednotlivé části mohou být použity samostatně. V projektu jsou vloženy závislosti jen na ty části, které projekt potřebuje[3]. Pro použití frameworku v praktické části stačí použít jádro frameworku, podporu MVC architektury a podporu JDBC. Díky Spring frameworku jsou pak jednotlivé části API zpřístupněny v jednotlivých třídách - controllerech, kde samotná třída obsahuje anotaci @Controller a @RequestMapping(”/api”), kde parametr anotace RequestMapping udává adresu, na které jsou reprezentovány požadované metody dané třídy. Jednotlivé veřejné metody pak také obsahují anotaci @RequestMapping(”/categories”), kde parametr anotace @RequestMapping značí adresu, na které se nachází výstup dané metody.

Bude-li potřeba v controllerech přistupovat k databázi, využije controller principu MVC a Inversion of Controll. Bude potřeba nadefinovat v nastavení enterprise aplikace připojení a třídu, která se o připojení stará. Dále musí být vytvořena modelová třída, která pomocí anotace @Autowired před proměnou požádá framework o vytvoření instance zmíněné třídy. V modelu jsou nadefinovány a implementovány obslužné metody a do nastavení aplikace tento model musí být přidán. V samotných controllerech pak opět přes anotaci @Autowired před proměnou controller požádá framework o vytvoření instance požadované modelové třídy.

MyBatis

V předešlé kapitole byl zmíněn datový model vytvářeného API. Klasický způsob získávání dat z databáze pomocí ovladačů JDBC, konkrétně tříd PreparedStatement a ResultSet je zastaralý a pro potřeby projektu nepohodlný, proto

[3] V praktické části tyto závislosti na straně API řeší build manager Maven

zmiňuje tato práce také knihovnu MyBatis, která celý přístup k databázi zjednodušuje. K přístupu k datům pomocí ovladačů JDBC by musel být zadán SQL dotaz, vytvořit instanci třídy PrepareStatement, nastavit parametry dotazu, z instance třídy PrepareStatement vytvořit instanci třídy ResultSet a pomocí iterátoru vytvořenou instanci třídy ResultSet projít a naplnit požadovaný objekt daty, které jsou dále reprezentovány pro potřeby mobilní aplikace. Navíc při plnění objektu daty je třeba dát pozor na datové typy.

Právě tuto problematiku zjednodušuje knihovna MyBatis. Na rozdíl od ORM knihoven, které podobné mapování dat z relačních databází na objekty také řeší [4], knihovna MyBatis řeší především přesně ten problém, který je popsán o odstavec výše, totiž naplnit jednoduše datovou strukturu daty z relační databáze. Oproti jiným knihovnám navíc MyBatis umožňuje uchovat si stoprocentní kontrolu nad sestavenými SQL dotazy, což u jiných ORM knihoven není úplně běžné.

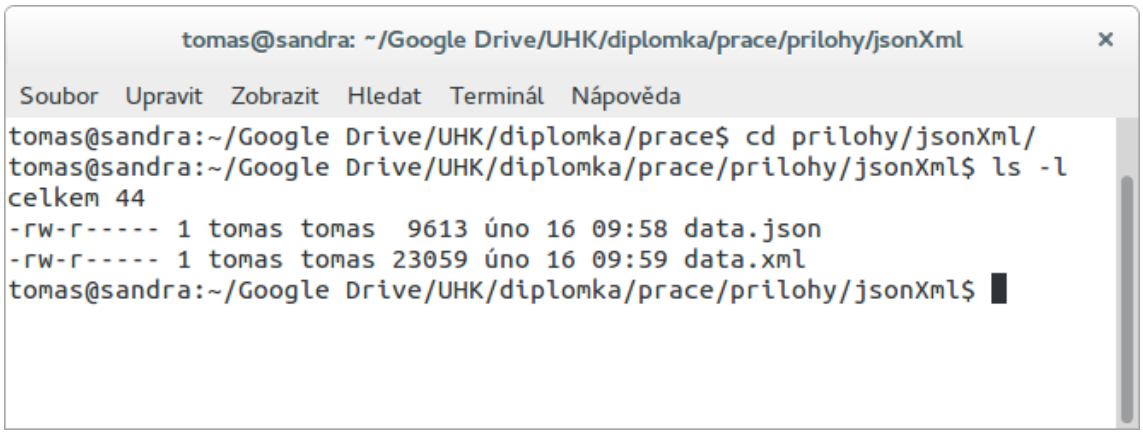
Využití knihovny je následující: Vytvořit interface, ve kterém jsou nadefinovány metody, které budou volány pro jednotlivé požadavky na data. Jednotlivé metody musí být opatřeny anotací @Select a jako parametr této anotace bude konkrétní SQL dotaz, který má být vykonán. Knihovna se pak sama postará o namapování výsledků, které vrátí SQL dotaz do objektu, který je požadován. Po vytvoření tohoto rozhraní musí být rozhraní přidáno do nastavení aplikace a pomocí anotace @Autowired aplikace (controllry) získá přístup k metodám rozhraní. Tím se z rozhraní stává model, který nemusí řešit připojení do databáze a složité metody pro získání dat z databáze pomocí tříd PrepareStatement a ResultSet.

5.1.2 Přenosový formát

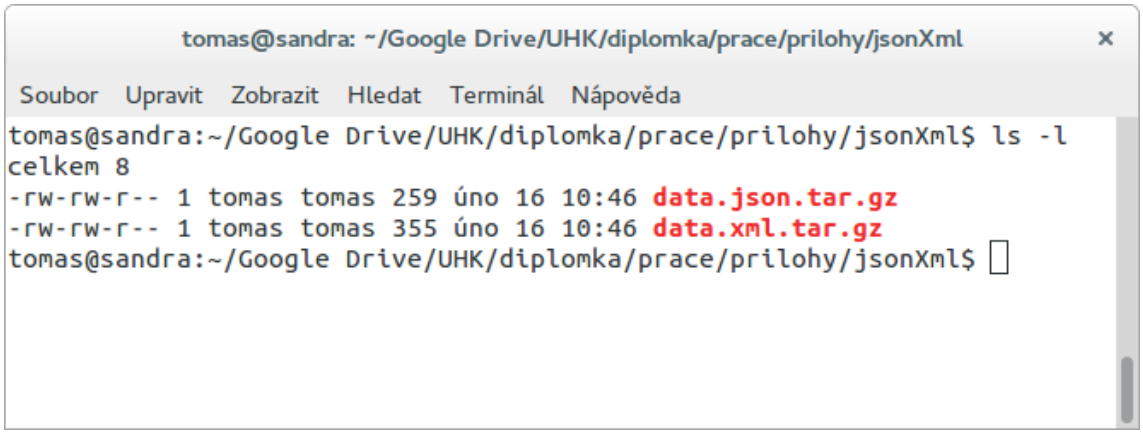
Současné API nabízí data ve formátu XML. Návrh nového API počítá s tím, že data budou poskytována ve formátu JSON. Je to z několika důvodů. První důvod je objem přenesených dat. Protože požadavek na mobilní aplikaci byl takový, aby si veškeré recepty mobilní aplikace aktuálně stahovala sama, tj. bez žádného trvalého úložiště [5], je otázka objemu přenesených dat velice citlivá. Na obrázku 5.1 je konzolový výpis dvou datových souborů, jeden ve formátu JSON a druhý ve Formátu XML. Je vidět, že soubor ve formátu XML je zhruba 2,4 krát větší. Obrázek 5.2 udává totožné soubory, jen zkomprimované. Je vidět,

[4] například knihovna Hibernate ORM

[5] S výjimkou uložených oblíbených receptů



Obrázek 5.1: Porovnání velikostí formátu JSON a XML.



Obrázek 5.2: Porovnání komprimované velikostí formátu JSON a XML.

že xml soubor už není 2,4 krát větší, ale 1,4 krát, přesto je úspora přenesených dat patrná.

Další výhodou formátu JSON je snadné generování a čtení dat pomocí knihovny Gson, kterou lze snadno použít jak pro funkci API, tak pro mobilní aplikaci. Knihovna Gson od společnosti Google jednoduše naplní nadefinovanou datovou strukturu daty pomocí volání jedné metody, která má na svém vstupu řetězec JSON formátu a Type třídy, kterou knihovna naplní daty. Na jednom řádku tak má programátor vyřešené parsování souboru a následně, mnohdy složité, přetypovávání na výsledný datový typ. Z dokumentace knihovny se lze dočíst, že knihovna disponuje těmito vlastnostmi:

- Poskytnout jednoduché metody toJson a fromJson, které se starají o zmíněný převod z java objektů do JSON řetězce a z JSON řetězce do java objektů.

- Povolit vlastní reprezentaci objektů
- Povolit vlastní složité reprezentace objektů s hlubokou dědickou hierarchií a s rozsáhlým využití generických typů
- Rozsáhlá podpora Java generics

V následující ukázce zdrojového kódu je naznačeno, jak knihovna pracuje.

```
public class ExampleController {

   @Autowired         //namísto gson = new Gson();
   private Gson gson;

   public void parseExample(){
      String jsonTxt = "{\"name\": \"Tomáš\", \"surname\":
         \"Pešek\", \"age\": 27}";

      Person person = gson.fromJson(jsonTxt, Person.class);
      System.out.println(person); // Name: Tomáš, surname: Pešek,
         age: 27,000000

      String jsonFromObject = gson.toJson(person);
      System.out.println(jsonFromObject); // {"name": "Tomáš",
         "surname": "Pešek", "age": 27}
   }
}

public class Person{
   private String name;
   private String surname;
   private double age;

   public String getName() {
      return name;
   }
   public void setName(String name) {
      this.name = name;
   }
   public String getSurname() {
      return surname;
   }
   public void setSurname(String surname) {
```

```
        this.surname = surname;
    }
    public double getAge() {
        return age;
    }
    public void setAge(double age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return String.format("Name: %s, surname: %s, age: %f",
            name, surname, age);
    }
}
```

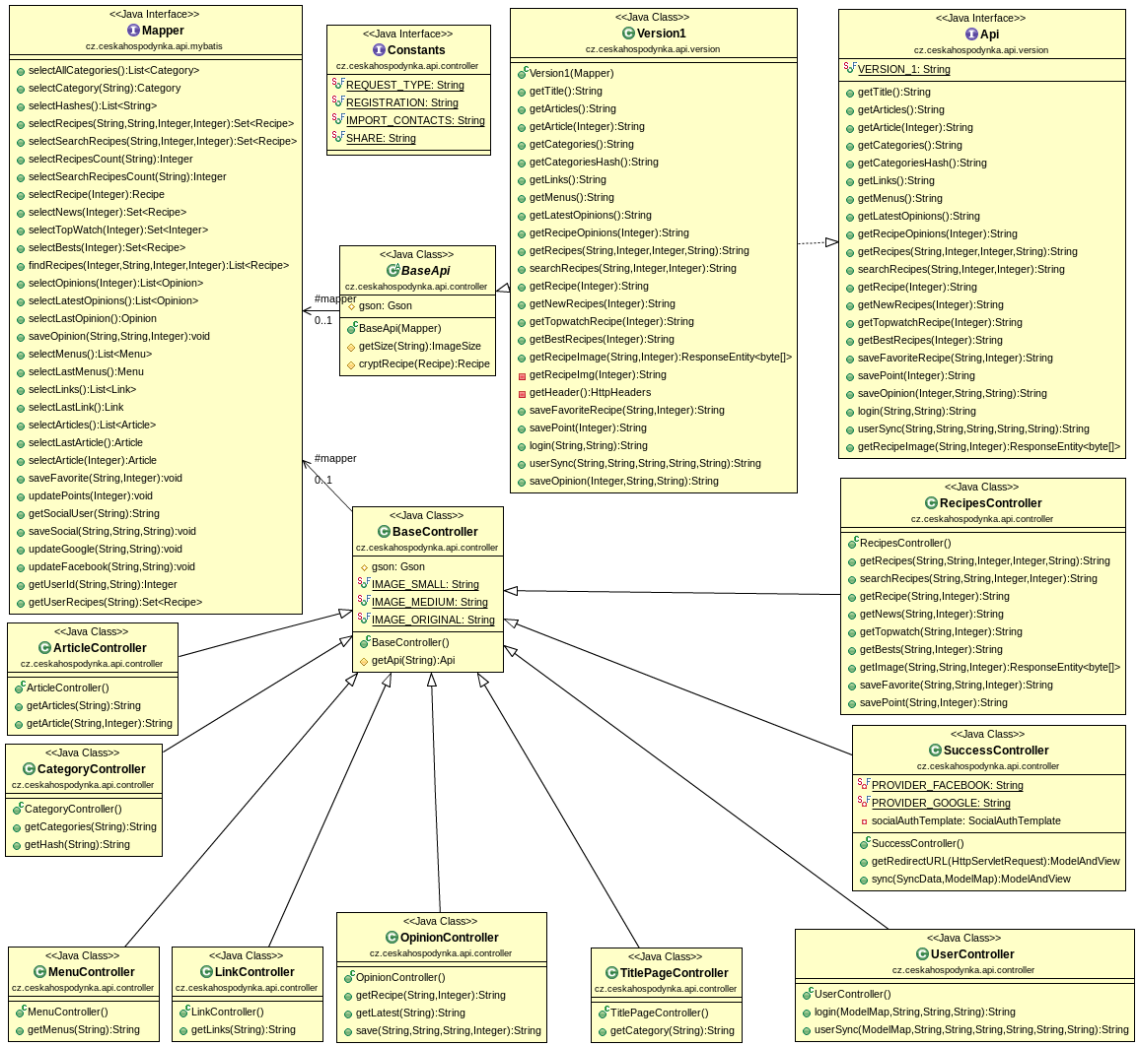
Ukázka kódu 5.1: Použití knihovny Gson

5.1.3 Úloha controllerů

Funkce API jsou popsány výše. Jak bylo zmíněno dříve, API je postaveno na architektuře MVC. Jednotlivé metody API jsou tedy pro přehlednost rozděleny do různých controllerů podle tohoto klíče:

- Články (seznam článků a konkrétní článek)
- Kategorie (seznam kategorií a obrázky k jednotlivým kategoriím)
- Odkazy (seznam odkazů)
- Menu (seznam menu)
- Názory (poslední názory, názory k receptu, přidat názor)
- Recepty (seznam receptů pro danou kategorii, nejnovější, nejhledanější nejbodovanější, bodování a ukládání receptů)
- Úvodní stránka (na jeden dotaz vrátí posledních pár nejnovějších, nejhledanějších, nejbodovanějších receptů, posledních pár názorů, odkazů a menu)
- Sociální sítě (možnost přihlášení a zobrazení profilu)

Na obrázku 5.3 je znázorněn Class diagram controllerů a modelu. Diagramu neobsahuje datové třídy doménového modelu, které jsou využívány k ukládání



Obrázek 5.3: Class diagram controllerů a MyBatis mapperu (modelu)

data a kromě getterů a setterů neobsahují žádné jiné metody. Tyto třídy jsou společné jak pro API, tak pro mobilní aplikaci, mají jednoho společného předka, který nabízí jednu proměnnou s chybovou hláškou, která může být využita při rozšiřování aplikace u případů, kdy už z jakéhokoliv důvodu není žádoucí, aby některá funkce API byla přístupná. Jestliže mobilní aplikace zjistí, že je chybová hláška vyplněná, nedokončí další kroky, které následují po ukončení komunikace s API, ale tuto chybovou hlášku zobrazí.

Controllery se také starají o to, aby data přenesená pomocí API byla chráněna proti jednoduchému vykradení. Jelikož portál je veřejný, nelze nikdy zabránit vykrádání databáze, proto při návrhu zabezpečení bylo bráno v potaz hlavně to, aby potenciální zloděj neměl veškerá data čitelně k dispozici na jedné ad-

rese. Proto v celém API není žádná možnost poskytnutí celé databáze receptů najednou a všechny texty jsou šifrované algoritmem AES. AES je symetrická bloková šifra, používající sdílený klíč. Díky tomuto se případnému vykradači dat práce ztíží. Z principu java byte kódu by útočník mohl získat šifrovací klíč. Otázkou je, zda by pro útočníka bylo jednodušší získání tohoto klíče, nebo získání software, který by získal data ze současného portálu.

Hlavní náplní controllerů je však nabídnout data. V návrhu je zmíněno, že API musí být schopno nabídnout i starší verzi rozhraní. Na obrázku 5.3 je znázorněno, že veškeré controllery mají jednoho společného předka (BaseController) a všechny veřejné metody jednotlivých controllerů vyžadují jako první parametr proměnnou typy String. V této proměnné je API poslána verze API, se kterou je mobilní aplikace schopna komunikovat. Na již zmíněném obrázku je znázorněno rozhraní Api, které je implementováno třídou Version1. Kdyby do budoucna vznikla poptávka po nové funkcionalitě pro API, stačilo by vytvořit novou třídu, například Version2, která bude buď dědit Version1, nebo bude implementovat znovu rozhraní Api a přepíše buď jen konkrétní, nebo veškeré metody. Následně by bylo nutné rozšířit metodu getApi v třídě BaseController (ukázka zdrojového kódu 5.2) o další větev, která by vrátila instanci nově vytvořené třídy Version2.

```
protected Api getApi(String version){
   if(Api.VERSION_1.equals(version)){
      return new Version1(mapper);
   } else {
      throw new IllegalStateException("Api version not
         implemented!");
   }
}
```

Ukázka kódu 5.2: Aktuální podoba metody getApi(String version)

Po té by bylo možno upravit mobilní aplikaci tak, aby mohla pracovat s pozměněnou strukturou dat a na závěr by se v nastavení konstant mobilní aplikace uvedla nová verze API, která se ve všech požadavcích na API automaticky nastaví a odešle. Tím je docíleno toho, že neaktuální aplikace budou žádat se starým identifikátorem API a dostanou starou strukturu dat, nové verze už budou využívat nové struktury.

5.2 Mobilní aplikace

Definice slova Android znamená Kompletní sada softwaru pro mobilní zařízení: operační systém, middleware [6] a klíčové mobilní aplikace (telefonování, ...). Jedná se o platformu, která může být instalována na různých hardwarových konfiguracích. Zatím ještě nejčastěji na mobilních telefonech a tabletech. V poslední době ale může být Android dodáván k hodinkám, televizím, settop boxech nebo ke čtečkám knih [8]. Navíc ho mohou instalovat různí výrobci a dodavatelé.

5.2.1 Mobilní technologie a Android SDK

Na domovské internetové stránce [7] je zmíněn odkaz vedoucí na stránky pro vývojáře [8]. Zde jsou k nalezení detailnější zmínky o softwarové vývojové sadě (SDK) pro Android, pomocí které může kdokoliv vytvářet aplikace pro platformu Android. V citované knížce je kapitola zaměřena na seznámení s touto platformou pojmenována výstižně Java, but not Java Java a je zde vysvětlen rozdíl mezi "Android Javou"a "Sun/Oracle Javou". Rozdíl je v tom, že běhové prostředí pro Javu (JRE) se skládá ze dvou částí. První částí jsou základní balíčky tříd poskytované platformou Java, jako například balíčky pro práci s kolekcemi, se sítí, matematickými operacemi a dalšími. Druhá část JRE obsahuje virtuální stroj (JVM), který se stará o spuštění programů interpretovaných Java bytcodem, který obsahuje svazek class souborů vyrobených Java kompilátorem.

Běhové prostředí pro Android (ARE) vychází ze stejného návrhu, ale nedisponuje všemi základními balíčky jako JRE (nedisponuje například balíčky java.applet, java.rmi, javax.sound nebo javax.swing, naopak balíčky java.io, java.math, java.sql, java.text, javax.net nebo javax.xml disponuje). Na druhou stranu ARE obsahuje některé knihovny navíc. Dále také ARE neobsahuje JVM, která se stará o spouštění aplikací a proto na ARE nelze spustit interpretovaný Java bytecode. ARE disponuje vlastním virtuálním strojem přímo od společnosti Google. Starší virtuální stroj se nazývá Dalvik a od Android verze 4.4 společnost Google představila nový virtuální stroj ART [9] (ve verzi 4.4 si uživatel může vybrat, zda chce používat Dalvik nebo ART, od verze 5.0 už společnost Google Dalvik vyloučila. Tyto virtuální stroje nejsou součástí Android platformy. Lze jej provozovat

[6] Software poskytující služby nad rámec operačního systému [7]
[7] `https://android.com/`
[8] `https://developer.android.com/index.html`
[9] Android Runtime

i na dalších operačních systémech na bázi LINUxu, včetně Vanilla LINUX, BSD nebo Mac OS. Virtuální stroje pro mobilní zařízení jsou přizpůsobeny pro potřeby těchto zařízení. Musí být rychlé, ale nesmí zatěžovat CPU, nesmí mít velké paměťové požadavky a musí fungovat energeticky efektivně. V čem se vlastně Dalvic a ART liší od JVM? Předně, jak bylo řečeno, neinterpretuje Java bytecode, který je tvořen z class souborů, získaných Java compilerem. Dalvik a ART interpretuje vlastní bytecode, který se skládá z jednoho DEX (Dalvic executable) souboru, který má zhruba poloviční velikost než výsledný JAR soubor pro JVM. Další rozdíl mezi JVM a Dalvik/ART je v tom, že JVM je stroj založený na bázi zásobníku, kdežto Dalvik/ART na bázi registrů. Stroj založený na bázi registrů má sice větší instrukční sadu, může ale provádět stejné úkoly s menším kódem, lépe řečeno provede stejný úkol za použití nižšího počtu instrukcí, což snižuje náročnost programu na CPU.

Dalvik/ART se bez Java kompilátoru ale neobejde. Proces kompilace zdrojového kódu totiž probíhá tak, že nejprve se z Java souborů pomocí Java kompilátoru stanou class soubory a po té se vytvořené class soubory "zdexují"[10] do výsledného DEX souboru. [8]

Dalvic proti Art

Výše bylo zmíněno, že Dalvic a ART jsou virtuální stroje od společnosti Google. Starší Dalvic využívá JIT [11] kompilaci, novější ART AOT[12] kompilaci. Rozdíl je v tom, že JIT kompilace dynamicky překládá části programu do strojového kódu před každým spuštěním aplikace, kdežto AOT tento krok provede jednou, při instalaci aplikace. Výhody AOT oproti JIT jsou v tom, že aplikace se rychleji spustí, protože při spuštění vynechává část kompilace do strojového kódu, na druhou stranu AOT musí výsledek kompilace uložit a provést při instalaci, takže se zvyšují paměťové nároky na zařízení a prodlužuje se čas instalace.

Jiné mobilní platformy

V úvodu kapitoly bylo popsáno softwarové prostředí pro vývoj aplikací pro platformu Android. Dnešní trhu však nabízí další mobilní platformy. Mezi další nejznámější mobilní platformy patří zařízení od společnosti Apple s operačním

[10] pomocí dx tool se znovu zkompilují
[11] Just-In-Time
[12] Ahead-Of-Time

systémem iOS [13] a zařízení, které používají operační systém na bázi Windows [14]. Ideální jak pro vývojáře aplikací pro mobilní zařízení, tak pro zadavatele těchto aplikací, by bylo, kdyby vývoj pro tato zařízení byl stejný. Vývojářům by to ušetřilo čas, zadavatelům finanční prostředky. Bohužel tomu tak není. Aplikace pro zařízení od společnosti Apple se vyvíjejí obdobně jako aplikace pro Android zařízení s tím rozdílem, že místo Android SDK musí být použito iOS SDK, které bohužel s Android SDK nelze srovnávat. Aplikace už nevytváříme v programovacím jazyce podobným Javě, ale v jazyce, který je spíše podobný jazyku C++. [9]. Stejně tak je to s aplikacemi pro zařízení s Windows Phone operačním systém, pro který musím pro změnu použít Windows Phone SDK. Zařízení s operační systém Windows Phone obsahují běhové prostředí Windows Runtime (WinRT), které podporuje vývoj aplikací v různých programovacích jazycích, jako například pro Windows typické C++ nebo C#. Umožňuje ale i spouštět aplikace naprogramované v Javascriptu a HTML [10].

Z výše uvedeného lze tedy vyvodit, že pokud chce vývojář vytvořit aplikaci pro tři nejběžnější mobilní platformy, musí ovládat tři mobilní technologie. Existují nástroje, které tuto problematiku řeší. Tato práce uvádí dva příklady takových to nástrojů. První z nich se nazývá PhoneGap. PhoneGap je nástroj, pomocí kterého může být vytvořena nativní aplikace pro mobilní zařízení pomocí technologií HTML, CSS a JavaScript. Jádro PhoneGap je tvořeno open source platformou Apache Cordova [15], která se stará o zpřístupnění nativních funkcí zařízení (například kamera nebo SD karta) pomocí JavaScriptu. Vývoj můžeme vypadat tak, že je vytvářena klasická webová stránka pomocí HTML, CSS a JavaScriptu, ta může být v našem webovém prohlížeči vyzkoušena. Vytvoření nativní aplikace pak zjednodušeně spočívá v tom, že PhoneGap vytvoří jednoduchý webový prohlížeč pomocí nativních komponent (pro Android je to konkrétně android.webkid.WebView) a tento prohlížeč, nemá na starosti nic jiného, než zobrazit aplikaci napsanou pomocí HTML, CSS a JavaScriptu. Zařízení od společnosti Apple s operačním systémem iOS používají jinou komponentu (konkrétně Objective-C UIWebView), takže musí být počítáno s tím, že se výsledná nativní aplikace může zobrazovat jinak na zařízení s operačním systémem Android a jinak na zařízení s operačním systémem iOS. Což je odvěký problém vývojářů webových aplikací [11].

Druhý nástroj, který řeší problematiku multiplatformního vývoje pro mobilní

[13] iPhone, iPod, iPad, ...
[14] Windows Phone, nástupce Windows Mobile
[15] http://cordova.apache.org/

zařízení, se nazývá Xamarin. Xamarin nestaví vývoj aplikací na HTML, CSS a JavaScriptu, ale na C#. Samotná kompilace pak probíhá tak, že výsledný spustitelný soubor pro android jádro Xamarinu vytvoří tak, že přepíše instrukce volané v C# na jejich ekvivalent v Android Javě. Pro iOS pak stejný ekvivalent v Objective C.

Android SDK vs. PhoneGap a Xamarin

Pro praktickou část této práce bylo nakonec zvoleno Android SDK namísto alternativních knihoven, které by aplikaci snáze zpřístupnili pro mobilní zařízení s iOS nebo Windows Phone. V této části tento krok bude ospravedlněn. Na přednášce Temná strana Android API, která se konala 23.10.2013 na Fakultě elektrotechniky a informatiky na Pardubické univerzitě, mluvil Petr Nálevka, člen Urbandroid teamu [16], který vyvinul například aplikace Sleep as droid, která má milióny stažení, o temné straně Android API a mimo jiné zmínil termín Android jungle, který vystihuje skutečnost, že na trhu je nepřeberné množství zařízeních se systémem Android. Android SDK s tímto faktem počítá a různá omezení napříč platformami lze ve zdrojovém kódu podchytit a správná funkcionalita je tak zcela v rukách programátora. Naproti tomu u alternativních způsobů vývoje pro platformu Android se musí programátor spoléhat na konkrétní knihovnu a doufat, že použité funkce budou fungovat všude stejně.

Další nevýhodou alternativních knihoven je fakt, že s nimi nedosáhneme takové efektivity jako u nativního vývoje s Android SDK. Mobilní aplikace, která je součástí praktické části této práce, musí šetrně zacházet s přenesenými daty, čili musí umět efektivně pracovat s přijatými daty tak, že je bude umět efektivně kešovat, že si o data bude říkat postupně a že bude schopna si některá data zpracovávat v pozadí ve vlákně, aby nezatěžovala uživatelské rozhraní a aby nevznikaly zbytečné prodlevy v odezvách aplikace na uživatelský vstup. K tomu všemu musí být aplikace kompatibilní i se staršími verzemi Android platformy. Například řešení ActionBaru tak, aby bylo kompatibilní i se staršími platformamy je náročný úkol i v nativním vývoji pro Android SDK [17], natož pak v knihovnách, ve kterých programátor přesně neví, jak jejich API komunikuje se systémovými prostředky a jak volá systémové funkce.

[16] https://sites.google.com/site/urbandroidteam/
[17] Podrobněji se tímto problémem zabývá 5.2.2 část

5.2.2 Technologie

Dependency Injection, nebo-li řízení závislostí

Dependency Injection [18] (DI) je technika návrhového vzoru Inversion of Controll [19] (IoC). V této práci je tato technika zmíněna z toho důvodu, že obsah praktické části je už tak rozsáhlý, že má smysl se zabývat technikami, které zpřehlední zdrojový kód aplikace a uvolní některé pevné závislosti mezi jednotlivými komponentami. V klasickém modelu programování jedna třída vytváří jiné třídy, které zase vytvářejí nové třídy a tímto modelem vznikají těsné vazby mezi třídami. Proč je to problém? Z jednoho prostého důvodu. Když upravíme třídu, kterou volají jiné třídy, musíme pak v případě změny konstruktoru, nebo parametru konstruktoru, změnit všechny třídy, které vytvářejí instance změněné třídy [19]. Názorně je toto ukázané ve zdrojových kódech API rozhraní pro praktickou část. Ve většině tříd je používána knihovna pro práci s formátem JSON, se kterou pracuje třída Gson, která by mohla být inicializována prostým konstruktorem:

```
class GsonTest {

   Gson gson = new Gson();

}
```

Ukázka kódu 5.3: Původní třída

Takovouto konstrukci můžeme použít kdekoliv v programu. Problém nastane, když bude potřeba volání knihovny upravit, například nastavením formátu data:

```
class GsonTest {

   Gson gson = new GsonBuilder().setDateFormat(DateFormat.LONG);

}
```

Ukázka kódu 5.4: Změna v původní třídě

Tato jednoduchá úprava zajistí, že se datum bude správně přenášet, ale dále musí být veškeré třídy, které používali předchozí kód, otevřeny a upraveny.

[18] česky Vkládání závislostí
[19] česky Obrácená závislost

Řešením je právě využívat vlastnosti DI. DI odebírá třídám veškerou zodpovědnost za vytváření dalších tříd. Když třída potřebuje použít jinou třídu, dostane její instanci na vstupu. Způsobů, jak lze instancí předat, je více, následně jsou představeny tři nejznámější.

```
class GsonTest {

   public Gson gson;

}

class Container {

   public GsonTest provideGsonTest(){
      GsonTest gsonTest = new GsonTest();
      gsonTest.gson = GsonBuilder().setDateFormat(DateFormat.LONG);
      return gsonTest;
   }
}
```

Ukázka kódu 5.5: Property injection

```
class GsonTest {

   private Gson gson;

   public GsonTest(Gson gson){
      this.gson = gson;
   }
}

class Container {

   public GsonTest provideGsonTest(){
      GsonTest gsonTest = new
         GsonTest(GsonBuilder().setDateFormat(DateFormat.LONG));
      return gsonTest;
   }
}
```

Ukázka kódu 5.6: Constructor Injection

```
class GsonTest {
```

```
   private Gson gson;

   public setGson(Gson gson){
      this.gson = gson;
   }
}

class Container {

   public GsonTest provideGsonTest(){
      GsonTest gsonTest = new GsonTest();
      gsonTest.setGson(GsonBuilder().setDateFormat(DateFormat.LONG));
      return gsonTest;
   }
}
```

Ukázka kódu 5.7: Setter injection

O řízení závislostí se v programu stará Dependency Injection Container (DIC), který požadované třídy vytvoří a na vyžádání je předá dál. DIC může mít několik podob a záleží na technologii, která je v projektech použitá. V praktické části jsou použity technologie dvě. Na straně API vytváříme DIC pomocí Spring frameworku a v mobilní aplikaci vytváříme DIC pomocí knihovny Dagger 2.

```
<bean id="gson" class="com.google.gson.Gson">
   <property name="gson" ref="gson" />
</bean>
```

Ukázka kódu 5.8: Spring DIC

```
@Module(library = true)
public class ElementaryModule {

    @Provides @Singleton
    Gson provideGson(){
        return new GsonBuilder().setDateFormat(DateFormat.LONG);
    }
}
```

Ukázka kódu 5.9: Dagger2 DIC

Vytvořením takovýchto kontejnerů zpřístupníme třídu Gson, kde budeme potřebovat s formátem JSON pracovat.

Roboguice Principů DI využívá i, v praktické části použita, knihovna Roboguice, která, slovy autorů, hladí vrásky s vývojem pro platformu Android [27]. Tato knihovna konkrétně řeší problém s předáváním závislostí z XML layoutů, ve kterých programátor definuje vzhled aplikace. Není to nijak závažný problém, spíše se jedná o kosmetickou úpravu kódu tak, aby se dalo v kódu lépe orientovat. Klasicky lze získat vazbu na grafickou komponentu z šablony voláním funkce findViewById(int id), která vrací objekt typu android.view.View, který je předkem všech grafických komponent a programátor při volání výsledek přetypuje na výsledný, požadovaný, typ. Rozdíl je znázorněn v následujících příkladech:

```
public class RecipeActivity extends Activity {

    private TextView recipeName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layotu.recipe_detail);

        recipeName = (TextView)findViewById(R.id.recipe_name);
        recipeName.setText(...);
    }
}
```

Ukázka kódu 5.10: klasické získávání závislostí grafických komponent

```
@ContentView(R.layotu.recipe_detail)
public class RecipeActivity extends RoboActivity {

    @InjectView(R.id.recipe_name)
    private TextView recipeName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        recipeName.setText(...);
    }
}
```

Ukázka kódu 5.11: Získávání závislostí grafických komponent pomocí knihovny Roboguice

Rozdíl je patrný na první pohled. Pomocí knihovny Roboguice je kód přehlednější a snáze se v něm orientuje. Na vyšší úrovni programování by programátorům nemělo jít jen o funkčnost, ale i o zmíněnou přehlednost a s tím spojenou udržitelnost kódu pro budoucí vývoj aplikace.

Zpětná kompatibilita

Problém zpětné kompatibility byl zmíněn v úvodní části této kapitoly. K roku 2015 disponuje Android devíti verzemi. Statisticky nejpoužívanější je verze 4.4 KitKat, kterou používá 39,7% zařízení. Nejnižší sledovaná verze je 2.2, která ovládá 0,4% zařízení, druhá nejnižší sledovaná verze 2.3 pak ovládá 7,4% [20]. Tyto dvě verze pokrývají bez mála 8% všech zařízení a přitom podpora pro ActionBar v horní části aplikace přišla oficiálně až s verzí 3.0. Jelikož praktická práce chce podporovat tyto verze a jelikož chce využívat ActionBar, musela být zpětná kompatibilita řešena. Možnosti, jak tento problém řešit, jsou dvě. První možností je, vytvořit si vlastní komponentu pro ActionBar, která bude vše obhospodařovat, nebo druhá možnost, využít knihovnu, která je již vytvořena, otestována a funkční.

ActionBarSherlock a AppCompact jako náhrada chybějícího ActionBaru Problém s chybějícím ActionBarem ve starších verzích Android platformy řeší Support library přímo od společnosti Google. Tato knihovna však nenahrazuje funkce chybějícího ActionBaru. Support library pouze zařídí to, že když bude spouštěna funkce, které ve starých platformách chybí, tak aplikace nekončí chybou a pádem, jinak ale chybějící funkcionality nenahrazuje. Při podrobném zkoumání bylo zjištěno, že knihovna funkce pouze definuje, vracejí však prázdnou hodnotu, popřípadě nevykonávají žádný funkční kód. Chybějící metody a funkcionality poskytuje právě knihovny ActionBarSherlock nebo AppCompact. Ty jsou schopny rozpoznat verzi Android platformy, na které aplikace běží a podle řídící logiky aplikaci nabídnou nativní knihovny (verze 3.0 a vyšší), nebo vlastní implementaci chybějících (prázdných) metod.

Pro správné fungování knihovny je potřeba, aby programátoři vyvíjeli aplikaci minimálně pro Android 4.0 a aby projekt měl v build path přidanou podpůrnou knihovnu android-support minimálně verze 4. Dále jsou funkce knihovny používány tak, že aktivity (popřípadě fragmenty), obhospodařující chod aplikace, dědí příslušné části od tříd, které upravuje knihovna ActionBarSherlock, respektive AppCompact. Pro příklad, aktivitu zobrazující recept je reprezentována třídou RecipeActivity a v základním stavu dědí od třídy Activity, která

je zahrnuta ve standardním Android API. Bude li tato třída volat metody pro práci s ActionBarem, aplikace spuštěná na nižší verzi Android než 3.0, skončí chybou a ukončí se. Když bude do projektu přidána Support library minimální verze 4.0, aplikace neskončí chybou, ale ActionBar se nezobrazí. Pokud bude projekt obsahovat knihovnu ActionBarSherlock, nebo AppCompact a pokud bude aktivita dědit místo třídy Activity třídu SherlockActivity, respektive ActionBarActivity, aplikace i na starších verzích systému Android zobrazí ActionBar. Pro nižší verze než 3.0 se zobrazí ActionBar implementovaný v knihovně, pro vyšší verze se zobrazí nativní ActionBar.

Výhodou je, že v repozitářích nalezneme rozšíření, které přímo propojuje knihovnu ActionBarSherlock i AppCompact s knihovnou Roboguice. Pokud mají aplikace využívat výhody ActionSherlockBaru nebo AppCompactu a Roboguice, museli by použít vícenásobnou dědičnost, kterou Java (ani Android Java) nepodporuje. Podpůrná knihovna spojuje obě knihovny a ve výsledku by to vypadalo tak, že namísto dědění od SherlockActivity, nebo ActionBarActivity a RoboActivity dědí třídy od tříd RoboSherlockActivity, respektive RoboActionBarActivity, která jsou obsaženy právě v podpůrné knihovně.

Knihovny ActionBarSherlock a AppCompact nabízejí stejné možnosti využití, které jsou pro potřeby této práce postačující. V praktické části je nakonec použita knihovna AppCompact pouze z toho důvodu, že ji oficiálně doporučuje společnost Google jako doplněk zpětné kompatibility se staršími zařízeními. [26]

SQLite a ORM

Základní vlastností aplikace je to, že si uživatel může stáhnout své oblíbené recepty do paměti zařízení a tyto recepty později používat i v momentech, kdy není připojený k internetu.

Android disponuje v základním balíku s databázovým systémem SQLite. Pro potřeby ukládání dat je tento systém dostačující. Nejenže aplikace bude ukládat do této databáze oblíbené recepty, ale využije této databáze i k tomu, že do ní bude ukládat recepty, které uživatel obodoval a tím mu zabrání bodovat jeden recept vícekrát. Navíc ještě aplikace vytvoří mezipaměť pro seznam kategorií, které se nebudou tak často měnit a proto by bylo zbytečné při každém startu aplikace stahovat veškeré kategorie i s obrázkem. Databázový systém SQLite nabízí datové typy pro řetězec, celé číslo, reálné číslo i pro blíže nespecifické binární objekty (BLOB). Databáze funguje tak, že v paměti zařízení vytvoří sou-

bor, do kterého ukládá data. Programátor pak k datům přistupuje pomocí SQL dotazů.

V praktické části je zapotřebí pouze tří tabulek. Jedna pro ukládání oblíbených receptů, jedna jako mezipaměť pro kategorie a jedna pro ukládání bodovaných receptů. V předešlých částech této kapitoly byly představeny různé nástroje, které mohou usnadnit vývoj jak serverové, tak mobilní části této práce, avšak nástroj, který by nabídl dostatečné zázemí pro práci s SQLite a ORM, bohužel není k nalezení. Od takovéhoto nástroje by bylo možno očekávat relační mapování na námi definované objekty. Při vytváření těchto objektů by nástroj mohl nabídnout možnosti pro snadné vygenerování SQL dotazu, který by mohl vytvořit databázovou tabulku přesně podle potřeb, které jsou definovány ve zmíněném objektu. Tato práce nabízí kostru takovéhoto jednoduchého nástroje.

V práci je tedy vytvořen jednoduchý nástroj, který se stará o vytvoření databázových tabulek, o zápis do tabulek a o načtení dat z tabulek a namapování na objekty. S nadsázkou by se dalo říci, že se jedná o velmi základní náznak vlastní implementace ORM nástrojů pro práci s SQLite. Pro reprezentaci databázových tabulek nám slouží takzvané DAO [20] objekty. Ty využívají anotacích @Table a @Column, pomocí jejichž parametrů pomocná třída vygeneruje SQL dotazy pro vytvoření tabulek. Dále slouží DAO objekty jako schránky pro data, přičemž s daty pracují Handlery, ve kterých jsou implementovány funkce, které nad daty jednotlivých tabulek jsou potřeba vykonávat. Příklad jednoho DAO objektu může být viděn na ukázce zdrojového kódu 5.12. Pomocí pomocné třídy Generator vrátí metoda createTable SQL dotaz, který vytvoří tabulku pro uložení dat uchovávaných ve vytvořených DAO objektech.

```
@Table(name = "ch_category")
public class CategoryDAO {

    @Column(name = "name", type = Column.TEXT, primaryKey = true)
    public String categoryName;

    @Column(name = "description", type = Column.TEXT)
    public String categoryDescription;

    @Column(name = "image_url", type = Column.TEXT)
    public String categoryImageUrl;
```

[20] Data Access Object

```
    @Column(name = "image", type = Column.BLOB)
    public Bitmap categoryImage;
}
```

Ukázka kódu 5.12: Ukázka DAO objektu

Ostatní pomocné knihovny

Další knihovny, které jsou v praktické části použity, řeší pouze zjednodušení práce. Knihovna Gson byla představena výše. Další knihovnu, kterou práce zmíní, je knihovna Picasso od vývojářů ze skupiny Square, která mimo jiné stojí za vývojem výše zmíněné knihovny Dagger2 pro řízení závislostí. Picasso je knihovna, která umožňuje aplikacím lépe pracovat s externími obrázky. Díky této knihovně může být zefektivněna odezva aplikace v momentech, kdy si aplikace stahuje obrázkové soubory z API. I když na straně API jsou funkce, které se snaží datový tok omezit na minimum, nelze ho úplně eliminovat a v tomto případě je v programátorově zájmu, aby tato obrázková data přenášela aplikace na pozadí a nevytvářela tak vyšší odezvy na uživatelovy reakce. Knihovna Picasso veškerou obslužnou rutinu již implementuje, takže programátor nemusí sám vytvářet vlákna a starat se o jejich synchronizaci, vše má na starosti Picasso. Příklad použití je znázorněn na ukázce zdrojového kódu 5.13.

```
Picasso.with(getContext())
   .load(getRecipeImagePath())
   .placeholder(R.drawable.ic_launcher_bw)
   .error(R.drawable.ic_launcher)
   .into(recipeImage);
```

Ukázka kódu 5.13: Příklad volání knihovny Picasso

Tímto jednoduchým voláním zajistí knihovna to, že místo obrázku, který chceme získat z API, se nejdříve zobrazí lokální obrázek s identifikátorem R.drawable.ic_launcher_bw, v pozadí pošle požadavek na adresu getRecipeImagePath() a lokální obrázek se při úspěšném získání obrázku nahradí. Při neúspěšném získávání obrázku se nahradí jiným lokálním obrázkem s identifikátorem R.drawable.ic_launcher.

5.2.3 Realizace

V této části nemá smysl zobrazovat class diagram tak, jak byl zobrazený v části této kapitoly pojednávající o API. Mobilní aplikace je rozsáhlejší a class dia-

gram by nebyl moc vypovídající. V následující části bude zmíněna problematika zdrojového kódu. Android SDK nám automaticky rozdělí zdrojový kód na balík, kde jsou uloženy java soubory, které vykonávají funkční část kódu a na balík, kde jsou uloženy xml soubory definující vzhled aktivit, styly, texty, rozměry a obrázky. Při vývoji byl balík s java soubory dále rozdělen na dvě větvě. Na balík, který se stará o uživatelské rozhraní. V tomto balíku jsou definovány jednotlivé aktivity, fragmenty [21] a adaptéry, které se starají o naplnění použitých ListView a GridView daty. Druhý balík se stará o samotné jádro aplikace. Nalezneme v něm balík, který shromažďuje datové třídy pro načtení dat z přeneseného JSON formátu, balík, který obsahuje třídy, které nám usnadňují práci s databází, nebo balík, které shromažďuje třídy s metodami pro komunikaci se sociálními sítěmi

5.2.4 Uživatelské rozhraní

Z hlediska uživatelského rozhraní je v práci znázorněn přístup ke třem stěžejním aktivitám aplikace, se kterými se uživatel bude nejčastěji setkávat. Aktivity, které z hlediska funkcí nevykonávají nic zajímavého, zde nejsou zmíněny.

Hlavní aktivita

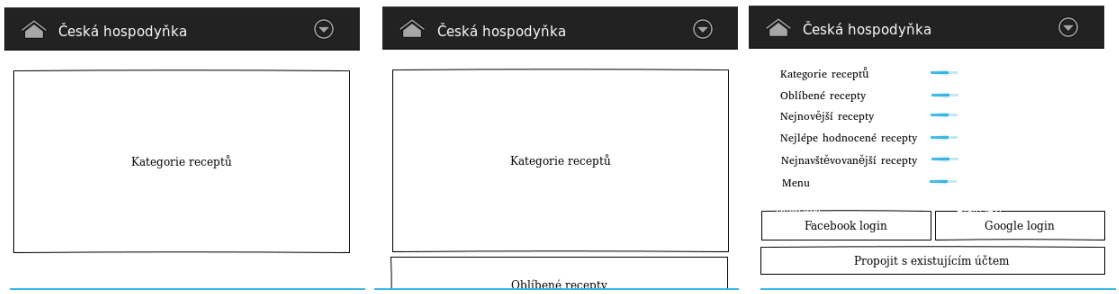
Při návrhu bylo zohledněno nevhodné využití úvodní stránky portálu. Tuto nevýhodu se snaží mobilní aplikace řešit tak, že po spuštění aplikace uvidí uživatel na úvodní aktivitě veškeré možnosti, které portál nabízí. Na obrázku 5.4 je znázorněno, jak návrh tohoto požadavku vypadá.

Tímto bylo docíleno toho, že většina funkcí je uživateli přístupná hned po spuštění aplikace. Bohužel s tímto návrhem je spjat problém v podobě různě velkých obrazovek u jednotlivých zařízení. S tímto rozložením stránky může nastat ta věc, že některým uživatelům se v úvodu zobrazí pouze kategorie receptů a ostatní komponenty budou čekat, až uživatel odroluje obraz níže, čímž vznikl další požadavek v podobě toho, že aplikace musí být schopná uživateli napovědět, aby odroloval obraz níže.

Při návrhu vznikly postupně tři možnosti, jak uživatele upozornit na to, že v úvodní aktivitě je více obsahu, než by se na první pohled mohlo zdát. První z možností má zajiti to, aby poslední položka byla pokaždé zobrazena tak, aby část byla již mimo zobrazenou plochu a část na ní byla.

[21] Fragment je přenositelná část aktivity, která se dá použít ve více aktivitách bez nutnosti kopírovaní nebo dědění aktivit

Obrázek 5.4: Návrh hlavní aktivity mobilní aplikace



Obrázek 5.5: Návrh upozornění na další komponenty na stránce

Další možností je krátká animace, která by využila toho, že jednotlivé komponenty se asynchronně stahují z API. Při prvním kroku by se uživateli zobrazily pouze řádky s nadpisem jednotlivých komponent a pod nimi by byla ikona in progress [22]. Při postupném nahrávání dat by se pak úvodní stránka rozšiřovala na výšku a uživatel by viděl, že na stránce je další obsah, který teď momentálně nevidí, ale na začátku vidět byl. Problém tohoto řešení je ten, že v momentě,kdy uživatel nebude při startu aplikace dávat pozor, je tato funkce k ničemu, protože si postupného načítání nemusí všimnout.

Návrh předešlých dvou možností si můžeme prohlédnout na obrázku 5.5, kde na prvním místě vidíme zobrazení oříznuté stránky, uprostřed návrh na oříznuté zobrazení nejníže položené, viditelné, komponenty a v pravé části návrh s postupným načítáním komponent.

Jako poslední možnost, jak uživatele upozornit na další obsah na stránce, je vytvořením jakéhosi průvodce v aktivitě, kdy úvodní stránka zmodrá, překryje jí průhledný panel, na kterém se zobrazuje animace a text nápovědy. Po dokončení animace je uživatel vyzván, aby klikl na potvrzovací tlačítko, tím se vyvarujeme nevýhodě předešlého řešení. Nápověda by se samozřejmě zobrazila jen při prvním spuštění. Problém by ovšem nastal, kdyby si aplikaci v našem zařízení pustil někdo jiný.

V práci je implementována první varianta v kombinaci s druhou.

Seznam receptů

V seznamu receptů probíhá nejnáročnější zpracování dat z API. Jak bylo zmíněno v kapitole 2 na stránce 4, tak na internetovém portálu jsou recepty pro zvolenou kategorii nahrány všechny najednou. Kdyby toto bylo implementováno v mobilní aplikaci a uživatel by si vybral ku příkladu kategorii Maso, kde

[22] nejčastěji je k vidění točící se kolečko

je uloženo více než tisíc receptů, tak než by aplikace zobrazila uživateli všechny recepty, vyvolalo by to dojem nefunkční aplikace. Proto při návrhu těchto seznamů muselo být myšleno na to, aby načítání receptů bylo plynulé a uživatele neobtěžovalo.

Aby načítání receptů bylo plynulé a neobtěžovalo, implementuje aktivita načítající recepty dvě metody, které se snaží tohoto docílit. První z nich je postupné načítání receptů. Tyto funkce můžou být k vidění u oblíbených sociálních sítí, když jsou prohlíženy příspěvky. Vždy se zobrazí určitý počet příspěvků a na konci se teprve znovu načítají příspěvky další. Tuto funkci tedy implementuje i aktivita obhospodařující zobrazení receptů. Tato aktivita nezobrazuje nativní ListView, ale vlastní komponentu LoadMoreListView, která od nativního ListView dědí a vlastně přidává a obhospodařuje OnScrollListener. Při zachycení události onScroll se na poslední pozici vloží spinner [23] a zavolá další metodu onLoadMore z rozhraní OnLoadMoreListener, které předává našemu rozšířenému ListView aktivita. V této metodě se odešle znovu dotaz na API s žádostí o další recepty a výsledek se opět předá rozšířenému ListView, které odebere položku se spinnerem a připojí vrácená data. Díky této komponentě se nemůže stát, že bylo načítáno najednou neúnosné množství dat.

Druhá metoda, která sice nešetří objem přenesených dat, ale za to dává uživateli dojem plynulé aplikace, je využití knihovny Picasso. Prakticky to znamená, že když se aplikace dotáže API na recepty, API vynechá ve výčtu informací obrázek k receptu, který představuje největší objem dat, i když se na straně API obrázek zmenšuje. Knihovna se pak postará o to, aby se obrázek stáhl ve vlákně a uživatel nemusel čekat.

Návrh layoutu pro seznam receptů si můžeme prohlédnout na obrázku5.6. Ještě zde musí být zmíněno, že aktivita zobrazující recepty je nejvíce využívaná aktivita v aplikaci. Jsou v ní zobrazovany recepty z jedenácti kategorií, nejoblíbenější recepty, nejvyhledávanější recepty, nejnovější recepty i recepty z menu. Proto je zde kladen velký důraz na znovupoužitelnost kódu.

Detail receptu

Aktivita receptu by mohla splňovat podmínky nedůležité aktivity, které byly zmíněny v úvodu této části. Nicméně, přeci jen o recept jde v aplikaci především a proto je zde aktivita zmíněna. Aktivita obsahuje velké množství grafických komponent a víc než kde jinde zde vynikne knihovna Roboguice. Navíc tato

[23] Spiner je označení pro točící se kolečko, signalizující aktivitu v pozadí

Obrázek 5.6: Návrh stránky se seznamem receptů

aktivita je s hlavní aktivitou jediná v aplikaci, která kromě toho, že přijímá data z API, tak na API i data odesílá, konkrétně odesílá informaci o tom, že uživatel recept obodoval, uložil do svých oblíbených receptů, nebo odeslal komentář k receptu.

V této aktivitě by mohla být ideálně řešena funkce nákupního košíku a průvodce přípravou receptu, bohužel portál nedisponuje takovou strukturou dat, která by implementaci těchto funkcionalit umožňovala.

5.2.5 Sociální sítě

Aplikace je, podle zadání, napojena na dvě sociální sítě. První sítí je Facebook, druhou Google +. Pomocí těchto sítí se uživatel může do aplikace pod svým účtem přihlásit a může na sítě sdílet svůj oblíbený recept. Obě sociální sítě disponují vlastním SDK, pomocí kterého zpřístupňují funkce pro práci s jejich API. SDK pro jednotlivé sociální sítě je součástí aplikace a v jádru jsou vytvořeny metody, pomocí kterých aplikace se sociálními sítěmi komunikuje. U obou sociálních sítí jsou v jádře vytvořeny třídy, které umisťují na jedno místo funkce se sítěmi tak, aby šly jednoduše předat pomocí DIC a aby ve funkcích byl přehled.

O vše ostatní se starají SDK jednotlivých sítí. V následující ukázce zdrojového kódu 5.14 je znázorněno, jak se inicializují tlačítka pro přihlášení přes sociální sítě Facebook a Google+.

```
@Inject            // namísto google = new Google();
private Google google;

private void initGoogle(Bundle savedInstanceState){
     google.onCreate(this, googleButton, savedInstanceState, new
        GoogleOnLogin() {
         @Override
         public void onLogin(final Person user) {
             runOnUiThread(new Runnable() {
                 @Override
                 public void run() {
                     if(user != null) {
                         addString(PREF_GOOGLE_USER_UID,
                             user.getId());
                         addString(PREF_GOOGLE_USER_NAME,
                             user.getDisplayName());

                         googleButton.setText(user.getDisplayName());
                         googleButton.setEnabled(false);

                         sync();
                     }
                 }
             });
         }
     });
 }

 private void initFacebook(){

    FacebookSdk.sdkInitialize(this.getApplicationContext());
     CallbackManager callbackManager =
        CallbackManager.Factory.create();

     LoginManager.getInstance().registerCallback(callbackManager,
             new FacebookCallback<LoginResult>() {
                 @Override
                 public void onSuccess(LoginResult loginResult) {
```

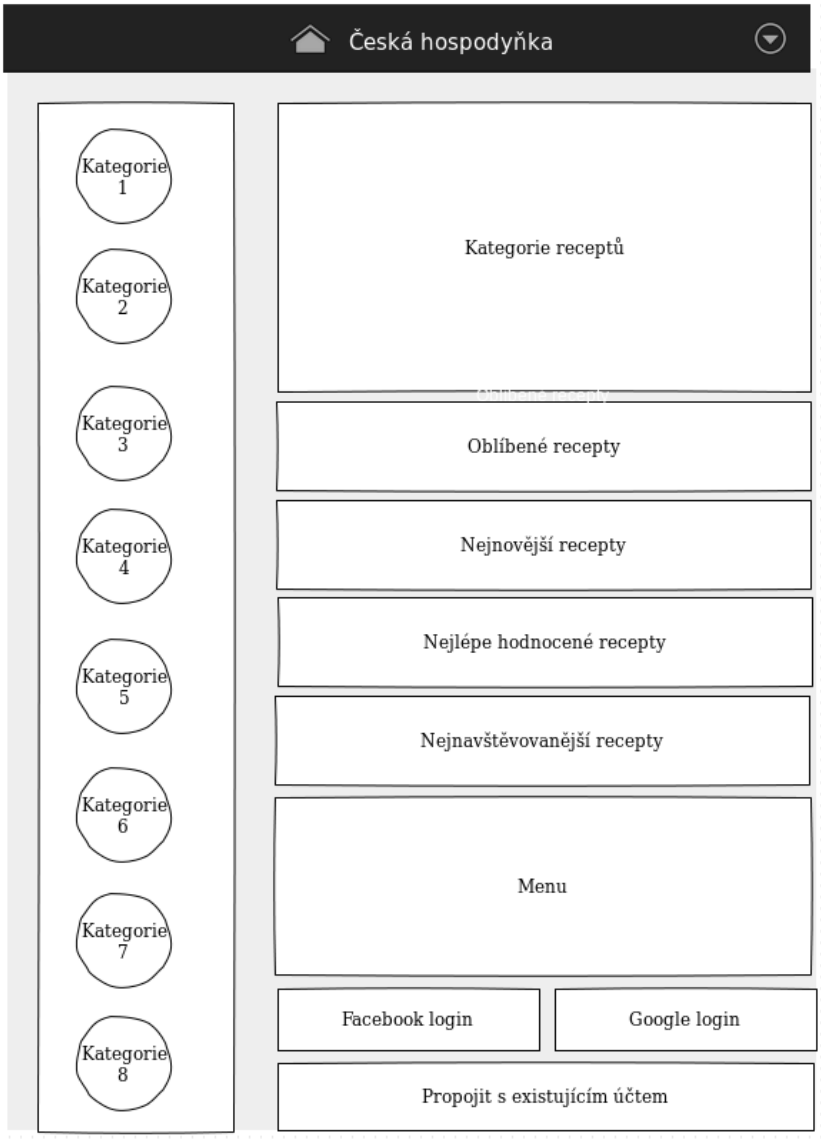
```
                facebookUpdate();
            }
            @Override
            public void onCancel() {
                facebookUpdate();
            }
            @Override
            public void onError(FacebookException exception) {
                facebookUpdate();
            }
        });

}

private void facebookUpdate(){
    boolean enableButtons = AccessToken.getCurrentAccessToken()
        != null;

    Profile profile = Profile.getCurrentProfile();
    if (enableButtons && profile != null) {
        addString(PREF_FACEBOOK_USER_UID, profile.getId());
        addString(PREF_FACEBOOK_USER_NAME, profile.getName());
        facebookButton.setText(profile.getName());
        sync();
    }
}
```

Ukázka kódu 5.14: Příklad inicializace sociálních sítí

5.2.6 Optimalizace pro tablety

V požadavcích na mobilní aplikace je na posledním místě zmíněno, že má být aplikace přizpůsobena pro tablety. Návrh se snaží upřednostnit výhody tabletu - větší prostor pro prezentaci obsahu. Návrh je znázorněn na obrázku 5.7 a je na něm vidět snaha nabídnout uživateli seznam kategorií na levé straně aplikace. Tento prvek je uživateli nabízen po celou dobu spuštění aplikace, tedy i v ostatních částech aplikace. Tento návrh vychází z kapitoly, která hodnotila webový portál a kde byla mimo jiné zmíněna absence druhé úrovně levého menu, která by mohla uživateli nabídnout seznam kategorií receptů.

Realizace tohoto návrhu obnášela vytvořit dvě šablony pro stejnou aktivitu. Android SDK s tímto postupem počítá. Android SDK rozezná rozlišení zařízení



Obrázek 5.7: Návrh úvodní obrazovky pro tablet.

a zkontroluje, zda aplikace nedisponuje speciální šablonou pro zařízení, ze kterého uživatel spouští aplikaci. Šablony pro tablety stačí tedy umístit do složky layout-swXYZdp, kde znaky XYZ značí rozlišení zařízení. Konkrétně jsou tedy ve složce layout-sw600dp uloženy šablony, které se zobrazí na sedmi palcovém tabletu, ve složce layout-sw720dp jsou šablony pro deseti palcový tablet.

5.3 Postup rozšíření aplikace

Při návrhu i vývoji byla brána v potaz skutečnost, že v budoucnu by mohl vzniknout požadavek na rozšíření mobilní aplikace, popřípadě upravit API. S touto skutečností návrh i realizace obou částí počítaly. V jednotlivých předešlých kapitolách byly popsány technologie a postupy při realizaci, přesto vše by pro nezasvěceného programátora bylo těžké se v aplikacích rychle zorientovat. V následujících částech práce popisuje základní možnosti rozšíření jednotlivých částí.

5.3.1 Nová verze API

V části pojednávající o API bylo zmíněno, že k řízení závislostí mezi jednotlivými knihovnami a k sestavování výsledného souboru war pro aplikační server byl využit nástroj Maven. Díky tomuto nástroji nemusí noví programátoři dohledávat, jaké verze podpůrných knihoven byly v této práci použity, stačí si stáhnout zdrojové kódy aplikace, mezi kterými je umístěn i konfigurační soubor maven s názvem pom.xml a ten se o stažení těchto knihoven postará. Tím si noví programátoři zajistí správné podklady pro rozšiřováni API rozhraní bez nutnosti zde vypisovat verze použitých podpůrných knihoven.

Změna neovlivňující strukturu přenesených dat

Příklad takovéto změny může být třeba změna systému v povolování komentářů. Současná funkce portálu všechny komentáře ihned zobrazí a až v momentě problému je administrátor může zakázat. Do budoucna by se mohl například systém změnit tak, že portál bude uchovávat nebezpečné IP adresy, ze kterých nebude chtít po nějaký čas komentáře zobrazovat. Stejná funkce by se očekávala i od mobilní aplikace. Úprava API by po té spočívala v tom, že by stačilo jen vhodně rozšířit SQL dotaz pro získání komentářů (napojit aktuální dotaz na seznam nebezpečných IP adres a rozšířit restrikci v klauzuli WHERE).

Tato změna nijak neupravuje strukturu přenesených dat, tudíž stačí pomocí sestavovacího manageru Maven vytvořit nový war soubor, který správce nahraje na server a po té restartuje aplikační server. Příklad pro sestavení war souboru nalezneme v ukázce zdrojového kódu číslo 5.15.

```
mvn install
```

Ukázka kódu 5.15: Příklad příkazu pro sestavení nové verze API

Změna ovlivňující strukturu přenesených dat

Naproti tomu takovéto úpravy už vyžadují více kroků, protože při špatně zvoleném postupu by mohla nastat situace, kdy starší verze mobilní aplikace přestanou s API komunikovat, protože API by nabízelo jiný formát dat, než kterému mobilní aplikace rozumí. Podrobně byla problematika vysvětlena výše, základním problémem je to, že není v silách ani možnostech kohokoliv, aby zajistil, že veškeré nainstalované mobilní aplikace budou vždy aktuální.

Jako příklad takovéto úpravy může být modifikován předešlý příklad, jen s tím rozdílem, že API odešle s názorem informaci o tom, že názor je zařazen na seznamu nebezpečných serverů a API nechá na mobilní aplikaci, aby tyto komentáře vyfiltrovala, popřípadě na ně jen upozornila uživatele formou například jinak zvolené barvy textu u takovéhoto komentáře. Tento princip popírá šetrnost na přenesená data, ale je zde uveden pouze jako příklad změny, která ovlivní strukturu přenesených dat.

Prvním krokem takovéto úpravy by mělo být vytvoření nové datové třídy doménového modelu, která bude obsahovat proměnnou nesoucí informaci o tom, zda komentář pochází z nedůvěryhodného zdroje, či nikoliv. Dalším krokem by byla úprava modelové části aplikace, která by spočívala v rozšířené tohoto modelu o novou metodu s novým SQL dotazem, která by vrátila nově vytvořenou instanci datové třídy. Následně by následovalo vytvoření nové třídy pro komunikaci, která by byla potomkem předešlé verze a přetížila by metodu pro získávání názorů, kde by byla volána nově vytvořená metoda modelu. Jako poslední krok by bylo nutné rozšířit metodu getApi() tak, jak to bylo již zmíněno na ukázce zdrojového kódu 5.2.

Následovalo by vygenerování war souboru, nahrání na server a restartovaní aplikačního serveru.

5.3.2 Rozšíření mobilní aplikace

Tak, jako u API byl použit nástroj pro řízení závislostí Maven, tak pro mobilní aplikaci byl použit nástroj, který pracuje na podobném principu, který se jmenuje Gradle. Stejně jako u nástroje Maven, tak ani u nástroje Gradle nemusí noví programátoři hledat konkrétní verze použitých knihoven, o vše se postará Gradle, který je navíc plně integrován v doporučeném vývojovém prostředí pro vývoj mobilních aplikací pro operační systém Android - Android studio.

Návrh a realizace mobilní aplikace počítala, stejně jako návrh a realizace API, s možností, že vznikne požadavek na rozšíření. Jestliže se požadavek na rozšíření nebude týkat změny struktury dat, tak lze mobilní aplikaci upravit libovolně. Po úpravách musí být sestaven nový apk soubor a ten následně nahrát na Google Play. Před nahráním na Google Play musí být změněn konfigurační soubor Gradle, ve kterém je uvedené číselné a textové označení verze. Google Play vývojářům neumožňuje nahrát apk soubor se stejným označením, které je již nahráno. Stejně jako v případě API, tak i nástroj Gradle nabízí příkaz, který vygeneruje požadovaný apk soubor. Příkaz je znázorněn na ukázce zdrojového kódu 5.16. Během provádění příkazu bude programátor dvakrát vyzván k zadání hesla. Jedno heslo je pro odemknutí klíčového úložiště a druhé k odemknutí samotného klíče, pomocí kterého je mobilní aplikace podepsána. Cesty k těmto klíčům jsou nadefinovány v konfiguračním souboru Gradle.

```
gradlew assembleRelease
```

Ukázka kódu 5.16: Příklad příkazu pro sestavení nové verze mobilní aplikace

V případě změny mobilní aplikace, která by vyžadovala změnu struktury přenesených dat, musí být nejprve správně změněno API (viz. výše), po té musí být upraveny (popřípadě vytvořeny) datové třídy doménového modelu stejným způsobem, jakým byly upraveny (vytvořeny) v API. Jestliže je vytvořen nový datový tok informací, musí byt vytvořen úkol, který nový datový tok načte. Tyto úkoly jsou shromážděny v balíku cz.ceskahospodynka.mobile.core.tasks. Nejjednodušší způsob vytvoření úkolu je ten, že nově vytvořený úkol bude potomkem třídy AndroidTask a přetíží metodu perform(Class<T> outputType), která vrací požadovaný typ nově vzniklého datového toku. V místě, kde je potřeba získat data z tohoto toku, pak musí být vytvořena nová instance tohoto úkolu, nad kterou musí být zavolána metoda perform. Příklad takovéhoto volání je znázorněn na ukázce zdrojového kódu 5.17.

```
private void loadMenu() {
   new MenuTask().task(Menu.class, new AfterExecutor() {
      @Override
      public void work(Object val) {
         if(val != null && val instanceof Menu){
            menuAdapter = new MenuAdapter(MenusActivity.this,
                ((Menu)val).getMenuList());
            menuView.setAdapter(menuAdapter);
            menuView.setClickable(true);
            menuView.setOnItemClickListener(new
                AdapterView.OnItemClickListener() {
               @Override
               public void onItemClick(AdapterView<?> parent, View
                   view, int position, long id) {
                  goToMenuDetail((Menu.Item)
                      menuAdapter.getItem(position));
               }
            });
            setProcess(false);
         }
      }
   });
}
```

Ukázka kódu 5.17: Příklad příkazu pro sestavení nové verze mobilní aplikace

6 Závěr

Cílem této práce bylo seznámit čtenáře s postupem návrhu aplikace pro mobilní platformu Android, která vychází z internetového portálu s recepty, který má na českém internetu již patnáctiletou historii. Návrh začal analýzou současného portálu. Byla provedena analýza portálu v pěti kategoriích. Díky této analýze se mobilní aplikace mohla vyvarovat nešvarům, které současný portál obsahuje. Hlavní náměty pro zapracování vychází z analýzy UX.

V další kapitole byla práce zaměřena na konkurenci a byly představeny některé aplikace, které může uživatel nalézt ve výpisu aplikací na Google Play. Při prozkoumávání těchto aplikací se práce zaměřila na to, co uživatelé na aplikaci hodnotí kladně a co nikoliv. V této kapitole byl sestaven seznam bodů, které byly u jednotlivých aplikací zkoumány a které byly považovány za důležité z hlediska pozdějšího návrhu vlastní aplikace.

Ve čtvrté kapitole se práce zmiňuje o potřebě vytvořit API rozhraní, se kterým by mobilní aplikace komunikovala a následně jsou popsány funkce, které jsou od API a od mobilní aplikace vyžadovány a které musí být zmíněny jednak v návrhu aplikace a po té v samotné implementaci.

Pátá kapitola je nejrozsáhlejší a věnuje se návrhu a realizaci praktické části. Kapitola je rozdělena do tří částí. První část se věnuje API rozhraní. Zde práce v úvodu představuje technologie, které jsou v praktické části použity a které vývojářům usnadňují práci a pomáhají udržovat přehledný a znovupoužitelný zdrojový kód. Práce takto zmiňuje Spring framework a knihovnu MyBatis. Následuje popis přenosového formátu JSON a jeho výhody oproti XML. V závěru části, která se věnuje API, je popsána úloha controllerů v architektuře MVC a popis případných úprav API rozhraní tak, aby byla zachována zpětná kompatibilita se staršími verzemi mobilní aplikace.

Druhá část páté kapitoly se věnuje mobilní aplikaci. Stejně jako v předešlé části jsou v úvodu zmíněny technologie, které jsou v práci použity. Práce představuje základní princip vývoje pro tři nejrozšířenější mobilní platformy a detailněji se věnuje softwareové sadě pro vývoj aplikací pro platformu Android. Zde práce zmiňuje rozdíl mezi dvěma virtuálními stroji, pomocí kterých plat-

forma Android spouští aplikace a vysvětluje rozdíl mezi JIT a AOT kompilací. Dále jsou představeny dva nástroje, Xamarin a PhoneGap, které usnadňují vývoj napříč mobilními platformami. Práce představuje jejich principy, výhody a nevýhody. Dále práce popisuje návrhový vzor IoC a jeho techniku DI. Jsou zde vysvětleny základní principy DI a jsou představeny jeho výhody a možnosti využití při vývoji pro platformu Android. V závěru úseku věnovaném technologiím pro mobilní aplikace práce zmiňuje problematiku zpětné kompatibility mezi jednotlivými verzemi operačního systému Android a představuje dvě knihovnu, ActionBarSherlock a AppCompact, které tuto problematiku řeší. Po představení technologií následuje návrh a popis realizace samotné mobilní aplikace. Práce popisuje návrh tří základních aktivit, se kterými má uživatel nejčastěji pracovat, jsou to aktivity úvodní stránky, seznamu receptů a detail receptu. Dále práce popisuje možnosti využití a příklad implementace sociálních sítí Google+ a Facebook do aplikací pro platformu Android. Na závěr práce popisuje optimalizaci aplikace pro tablety.

Ve třetí části páté kapitoly pak práce popisuje popisuje postup, kterým lze rozšířit API tak, aby byla zachována zpětná kompatibilita s předešlými verzemi mobilních aplikací. Práce zmiňuje dva typy rozšíření. První rozšíření nevyžaduje změnu struktury přenesených dat a druhé rozšíření tuto změnu vyžaduje. V práci je uveden praktický příklad obou postupů. Dále práce popisuje postup vytvoření nového war souboru. Tento postup je díky použití nástroje Maven automatický. Po popisu rozšíření API rozhraní následuje popis rozšíření mobilní aplikace, kde je v závěru popsáno vydání nového souboru. Tento postup je také automatický, jen s tím rozdílem, že místo nástroje Maven je použit nástroj Gradle.

Zadání práce znělo, navrhnout a implementovat aplikaci pro operační systém Android. Tato problematika byla prezentována v širším pojetí. Krom toho, že se práce zmiňuje i o jiných mobilních platformách, nezanedbatelná část práce je zaměřena na API, bez kterého by aplikace nemohla fungovat.

Možnosti API jsou vzhledem k datovému uspořádání uložených dat, a vzhledem k možnostem mobilní aplikace, celkem vyčerpané, mobilní aplikace má potenciál dalšího vývoje otevřený. Namátkově zde zmíním možnost ukládání vlastních poznámek nebo fotografií k receptům. Jak mobilní aplikace, tak API jsou podle mého svědomí navrženy jednoduše a čistě a žádná další úprava by neměla činit nejmenší problém, dokonce ani kdyby se provozovatel rozhodl zasáhnout do struktury dat na portále a otevřel by tak dveře dalším možnostem, které si uživatel pochvalují u konkurenčních aplikací, jako je třeba možnost vy-

tvářet nákupní seznam.

Literatura

[1] *MobiForge: Global mobile statistics 2014 Home: all the latest stats on mobile Web, apps, marketing, advertising, subscribers, and trends...* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://mobiforge.com/research-analysis/global-mobile-statistics-2014`

[2] *Stackoverflow: Tag info* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://stackoverflow.com/tags/android/info`

[3] *Stackoverflow: Tag info* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://stackoverflow.com/tags/iphone/info`

[4] *Vzhůru dolů: Stav trhu mobilních prohlížečů v Česku* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://www.vzhurudolu.cz/blog/18-mobilni-prohlizece`

[5] *Shoproku.cz: Výsledky soutěže ShopRoku 2013* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://www.shoproku.cz/vysledky-shoproku-2013`

[6] *Nh.ihned.cz: Izraelské ekonomice pomohly hlavně inovativní myšlenky a technologie* [online]. [cit. 2014-10-16]. Dostupný z WWW: `http://archiv.ihned.cz/c1-61319890`

[7] *Ironick: Update on the origin of the term "middleware"*. [online]. [cit. 2014-10-20]. Dostupný z WWW: `http://ironick.typepad.com/ironick/2005/07/update_on_the_o.html`

[8] *COLLINS, Charlie, Michael D GALPIN a Matthias KAPPLER. Android in practice. Shelter Island, NY.: Manning, c2012, xxii, 623 p. ISBN 19-351-8292-7.*

[9] *Apple Developer* [online]. [cit. 2014-10-20]. Dostupný z WWW: `https://developer.apple.com/`

[10] *Windows Dev Center: Get started* [online]. [cit. 2014-10-20]. Dostupný z WWW: `https://dev.windows.com/en-us/getstarted`

[11] *PhoneGap | PhoneGap Explained Visually* [online]. [cit. 2014-10-20]. Dostupný z WWW: http://phonegap.com/2012/05/02/phonegap-explained-visually/

[12] *Introduction to Mobile Development | Xamarin* [online]. [cit. 2014-10-20]. Dostupný z WWW: http://phonegap.com/2012/05/02/phonegap-explained-visually/

[13] *How To Maintain Hierarchy Through Content Choreography* [online]. [cit. 2014-10-20]. Dostupný z WWW: http://www.smashingmagazine.com/2013/04/25/maintain-hierarchy-content-choreography/

[14] *UX Design Defined - User Experience - UX Design* [online]. [cit. 2015-01-20]. Dostupný z WWW: http://uxdesign.com/ux-defined

[15] *What Is User Experience Design? Overview, Tools And Resources* [online]. [cit. 2015-01-20]. Dostupný z WWW: http://www.smashingmagazine.com/2010/10/05/what-is-user-experience-design-overview-tools-and-resources/

[16] *Integrate business modeling and interaction design* [online]. [cit. 2015-01-20]. Dostupný z WWW: http://www.ibm.com/developerworks/library/ws-soa-busmodeling/index.html

[17] *Informace důvěrné jako rozhovor* [online]. [cit. 2015-02-10]. Dostupný z WWW: http://archiv.ihned.cz/c1-63475070-informace-duverne-jako-rozhovor

[18] *Výsledky soutěže WebTop100 v roce 2014* [online]. [cit. 2015-02-10]. Dostupný z WWW: https://vysledky.webtop100.cz/2014/

[19] *Inversion of Control Containers and the Dependency Injection pattern* [online]. [cit. 2015-02-17]. Dostupný z WWW: http://www.martinfowler.com/articles/injection.html

[20] *Dashboards | Android developers* [online]. [cit. 2015-02-17]. Dostupný z WWW: https://developer.android.com/about/dashboards/index.html

[21] *Introduction to the Spring Framework* [online]. [cit. 2015-02-19]. Dostupný z WWW: http://www.theserverside.com/news/1364527/Introduction-to-the-Spring-Framework

[22] *Zabezpečte si web s HTTPS jen za 99 Kč / rok!* [online]. [cit. 2015-07-20]. Dostupné z WWW: `https://crt.simplia.cz`

[23] *StartSSL^{TM} Certificates & Public Key Infrastructure* [online]. [cit. 2015-07-20]. Dostupné z WWW: `http://www.startssl.com/?app=1`

[24] *Let's Encrypt* [online]. [cit. 2015-07-20]. Dostupné z WWW: `https://letsencrypt.org/`

[25] *ART vs Dalvik - Introducing the New Android* x86 Runtime* [online]. [cit. 2015-07-22]. Dostupné z WWW: `https://software.intel.com/en-us/blogs/2014/06/18/art-vs-dalvik-introducing-the-new-android-x86-runtime`

[26] *ActionBarCompat and I/O 2013 App Source* [online]. [cit. 2015-07-22]. Dostupné z WWW: `http://android-developers.blogspot.cz/2013/08/actionbarcompat-and-io-2013-app-source.html`

[27] *roboguice/roboguice - GitHub* [online]. [cit. 2015-07-22]. Dostupné z WWW: `https://github.com/roboguice/roboguice`

Přílohy

Seznam obrázků

2.1 základní kostra portálu. 5
2.2 Detail receptu. 10

5.1 Porovnání velikostí formátu JSON a XML. 29
5.2 Porovnání komprimované velikostí formátu JSON a XML. . . . 29
5.3 Class diagram controllerů a MyBatis mapperu (modelu) 32
5.4 Návrh hlavní aktivity mobilní aplikace 47
5.5 Návrh upozornění na další komponenty na stránce 48
5.6 Návrh stránky se seznamem receptů 50
5.7 Návrh úvodní obrazovky pro tablet. 53

Seznam tabulek

3.1 Shrnutí aplikace Miluji vaření - recepty 18
3.2 Shrnutí aplikace Recepty bezplatný 20
3.3 Shrnutí aplikace Apetit sezonní recepty 21
3.4 Shrnutí aplikace Česká kuchařka 22
3.5 Shrnutí aplikace Cookbook 22

Seznam ukázek kódu

5.1 Použití knihovny Gson 30
5.2 Aktuální podoba metody getApi(String version) 33
5.3 Původní třída 38
5.4 Změna v původní třídě 38
5.5 Property injection 39
5.6 Constructor Injection 39
5.7 Setter injection 39
5.8 Spring DIC 40
5.9 Dagger2 DIC 40
5.10 klasické získávání závislostí grafických komponent 41
5.11 Získávání závislostí grafických komponent pomocí knihovny Roboguice 41
5.12 Ukázka DAO objektu 44
5.13 Příklad volání knihovny Picasso 45
5.14 Příklad inicializace sociálních sítí 51
5.15 Příklad příkazu pro sestavení nové verze API 55
5.16 Příklad příkazu pro sestavení nové verze mobilní aplikace . . . 56
5.17 Příklad příkazu pro sestavení nové verze mobilní aplikace . . . 56

Obsah přiloženého CD

1. CeskahospodynkaAPI - zdrojové kódy API
2. CeskahospodynkaMobile - zdrojové kódy mobilní aplikace
3. Pesek_DP - Textová část

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Rokitanského 62, 500 03 Hradec Králové, tel: 493 331 111, fax: 493 332 235

Zadání k závěrečné práci

Jméno a příjmení studenta: **Tomáš Pešek**

Obor studia: Aplikovaná informatika (2)
Jméno a příjmení vedoucího práce: **Pavel Kříž**

Název práce:
Mobilní aplikace pro webový portál s recepty

Název práce v AJ:
Mobile Application for Web Portal With Recipes

Podtitul práce:

Podtitul práce v AJ:

Cíl práce: Navrhnout a implementovat aplikaci pro operační systém Android, která zpřístupní obsah portálu s recepty (včetně komentářů, bodování atd.) na mobilních zařízeních (telefony, tablety).

Osnova práce:

- Úvod
- Popis existujícího portálu
- Rešerše existujících mobilních aplikací
- Analýza a návrh řešení
- Implementace
- Shrnutí výsledků
- Závěry a doporučení

Projednáno dne:

Podpis studenta

Podpis vedoucího práce