VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Absolvování individuální odborné praxe

# Individual Professional Practice in the Company

2016

Adam Konečný

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Zadání bakalářské práce

Student: **Adam Konečný**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: Railsformers s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

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

Vedoucí bakalářské práce: **Ing. Radovan Fusek**

Konzultant bakalářské práce: Ing. Richard Lapiš

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016


doc. Dr. Ing. Eduard Sojka
*vedoucí katedry*

prof. RNDr. Václav Snášel, CSc.
*děkan fakulty*

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

V Ostravě 8. dubna 2016

Konečný Adam

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 8. dubna 2016


.....

Rád bych poděkoval Ing. Radovanovi Fuskovi za odbornou pomoc, kterou mi poskytnul při zpracování této bakalářské práce. Děkuji panu Ing. Jiřímu Kubicovi za možnost absolvování odborné praxe ve společnosti Railsformers s.r.o. Dále bych chtěl poděkovat Ing. Richardovi Lapišovi, který byl mým konzultantem na odborné praxi.

**Abstrakt**

Bakalářská práce popisuje absolvování odborné praxe ve společnosti Railsformers s.r.o. Odborná praxe byla zaměřena na vývoj mobilní aplikace pro operační systém Android. Aplikace je mobilním klientem pro online kuchařku receptů sRecepty. Úvod bakalářské práce popisuje důvody, které vedly k absolvování odborné praxe. V bakalářské práci je uveden seznam úkolů přidělených praktikantovi. U každého úkolu je detailně uveden postup implementace se zdůrazněním problémů, které se při vývoji objevily. Závěr práce je věnován zhodnocení dosažených výsledků a celkového přínosu praxe.

**Klíčová slova**: bakalářská práce, odborná praxe, mobilní aplikace, Android, Java, Railsformers s.r.o., sRecepty, vývoj mobilní aplikace

**Abstract**

The Bachelor's thesis describes a professional practice at the Railsformers company. The professional practice has focused on mobile application development for Android operating system. The resulting application is a mobile client of a cookbook called the sRecepty. The introduction of the bachelor's thesis includes various reasons of choosing a specialised training. It is followed by a list of tasks that has been assigned to the student. Every task includes a process of implementation in detail emphasising different solutions of issues. The end of this thesis focuses on achieved results and overall point of the practical training.

**Key Words**: mobile application, Android, Java, sRecepty, Railsformers,mobile application development

# Obsah

# Seznam použitých zkratek a symbolů

| | | |
|---|---|---|
| HTTP | – | Hypertext Transfer Protocol |
| HTML5 | – | HyperText Markup Language 5 |
| API | – | Application Programming Interface |
| JSON | – | JavaScript Object Notation |

# Seznam obrázků

# Seznam výpisů zdrojového kódu

# 1 Úvod

Jako téma pro vypracování bakalářské práce jsem si zvolil možnost absolvování odborné praxe. Hlavním důvodem, proč jsem si tuto variantu vybral, byl záměr získat reálné pracovní zkušenosti z IT odvětví. Dalším důvodem, který vedl k výběru této varianty, byl osobní zájem na vývoji mobilních aplikací pro platformu Android.

Odbornou praxi jsem vykonal u společnosti Railsformers s.r.o. U této společnosti jsem začal pracovat již v srpnu 2015. Úvodní část bakalářské práce popisuje společnost Railsformers s.r.o. Větší důraz je kladen na popis technologií, které při své činnosti společnost využívá. V této části je také uveden popis pozice praktikanta.

Druhá část bakalářské práce se zaměřuje na popis jednotlivých úkolů, které mi byly přiděleny a vedly k vytvoření mobilní aplikace pro operační systém Android.

Třetí část práce se zabývá samotnou implementací a realizací jednotlivých úkolů. V jednotlivých částech jsou uvedeny postupy úkolů se zdůrazněním problémů, které se při realizaci objevily. Větší pozornost je věnována právě implementaci vyhledávání a jeho rozšíření o nápovědu při psaní dotazu. Samotné vyhledávání probíhalo ve dvou fázích. První je popis implementace základního vyhledávání. Druhá fáze popisuje implementaci rozšíření, které zobrazuje uživateli nápovědu při psaní dotazu. V práci je také detailně popsána implementace notifikačního centra. Samotný popis této implementace je rozdělen do jednotlivých částí.

V závěru uvádím přehled dovedností a znalostí získaných prostřednictvím studia, které mi pomohly při vykonávání odborné praxe. Dále uvádím dovednosti a znalosti, které mi během absolvování praxe chyběly, a také samotný přínos absolvované praxe.

# 2 Popis odborného zaměření firmy

## 2.1 Popis společnosti

Společnost Railsformers s.r.o. vznikla v roce 2010. Od doby svého založení se zabývá vývojem pokročilých internetových a intranetových aplikací. Společnost se specializuje na projekty se širokým zaměřením, jako jsou pokročilé podnikové systémy, sociální sítě a komunitní portály. Ke své práci využívá nejmodernější webový framework Ruby on Rails, který je postaven na jazyku Ruby. Vývoj systémů ve společnosti je založen na agilním přístupu. V poslední době se společnost orientuje na vývoj mobilních aplikací s využitím multiplatformního HTML5 Ionic frameworku nebo nativního Androidu [2].

## 2.2 Popis odborného zaměření studenta

Ve firmě jsem pracoval jako junior Android programátor. Na této pracovní pozici jsem byl přidělen do týmu, který měl vytvořit mobilní aplikaci pro rozsáhlou kuchařku receptů Srecepty. V rámci této pozice jsem měl analyzovat a navrhovat řešení zadaných úkolů týkajících se mobilní aplikace.

# 3 Seznam zadaných úkolů

## 3.1 Aktivita s výpisem receptů a nekonečným seznamem

Vytvoření aktivity pro zobrazování přehledu nejnovějších receptů. Aktivita bude obsahovat seznam receptů stažených z webového API. Pokud uživatel sjede se seznamem receptů až k poslednímu receptu, dojde k dodatečnému stažení nových receptů – vytvoření nekonečného seznamu.

## 3.2 Vytvoření generické třídy pro stahování dat z webového API

Vhodně navrhnout novou třídu pro stahování obsahu z webu. Tato třída bude postavena na open source knihovně Volley od Google. Data stažená pomocí této třídy budou automaticky převedena do konkrétního typu. Tento typ je převzat z generického parametru.

## 3.3 Vkládání komentářů a recenzí z mobilního klienta

Uživatel mobilní aplikace může u konkrétního detailu receptu vložit komentář nebo recenzi. Nepřihlášený uživatel musí vložit jméno, je-li uživatel přihlášen, posílá se token, který rozpozná uživatele. Pro vložení komentáře a recenze vhodně zvolit a navrhnout komponentu pro vkládání.

## 3.4 Vyhledávání obsahu, vyhledávání s nápovědou

Uživatel může z hlavního menu aktivity vložit dotaz k vyhledání. Výsledky vyhledávání se zobrazí v nové aktivitě. V rámci vyhledávání se prohledávají recepty, ingredience a články, proto je nutné vhodně navrhnout aktivitu s využitím Android Fragmentů. Pokud uživatel píše dotaz do vyhledávacího podle, vhodně zobrazte nápovědu.

## 3.5 Vytvoření notifikačního centra

Cílem je prostudování služeb Google Cloud Messaging, které jsou nutné pro notifikování uživatele. Notifikace je zasílána z webového API, musí být aplikací zachycena a zobrazena uživateli ve formě notifikačního oznámení. Pokud uživatel klikne na tuto notifikaci, spustí se aplikace s aktivitou obsahující detail notifikace.

## 3.6 Vytvoření přehledu kategorií pro recepty

Pro zjednodušení orientace v aplikaci vhodně navrhnout přehled kategorií receptů. Cílem je simulovat chování webu. Aplikace je používána na mobilních telefonech a tabletech, proto je vhodné navrhnout vizuální komponenty pro jednotlivé typy zařízení.

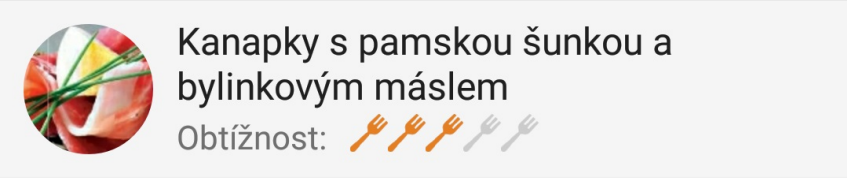
# 4 Zvolený postup řešení zadaných úkolů

## 4.1 Aktivita s výpisem receptů a nekonečným seznamem

### 4.1.1 Základní implementace

První úkol spočíval ve vytvoření aktivity s přehledem receptů. Prakticky se jedná o nejpoužívanější aktivitu v aplikaci sRecepty. Při vytváření jsem musel zjistit, jaký typ komponenty pro zobrazování seznamu použít. Jelikož celý tým od začátku pracoval v Material Designu, rozhodl jsem se použít pokročilou komponentu RecyclerView. RecyclerView slouží k zobrazování seznamu hodnot, které jsou získány z databáze nebo z webového rozhraní.

Prvním potřebným krokem bylo vytvoření základního XML vzhledu pro zobrazování získaných dat (Obrázek 1). XML představovalo položku v seznamu a bylo v souladu s podstatou Material Designu. Aby bylo možné vykreslit získaná data, musela se pro RecyclerView vytvořit vlastní třída RecyclerAdapter.




Obrázek 1: View pro zobrazení receptu

RecyclerAdapter implementuje návrhový vzor adapter a obsahuje pokročilou správu View, která zaručuje, že již vytvořená View jsou využita efektivněji, než kdyby měl toto na starost programátor. Samotný RecyclerAdapter prochází jednotlivé položky v seznamu. Pro každou položku vytvoří novou instanci View, do které jsou vložena data z listu. Adapter implementuje návrhový vzor pozorovatel, proto stačilo vložit nová data a automaticky došlo k vykreslení těchto dat [3].

Pro zobrazení dat v jednotlivých View byl implementován pomocný návrhový vzor ViewHolder. Tento návrhový vzor obsahuje instanci View obsahující reference na jednotlivé objekty potřebné pro vykreslení. Pro každý typ View, které potřebujeme vykreslit v RecyclerView, musíme vytvořit samostatný ViewHolder.

### 4.1.2 Implementace nekonečného skrolování

Aktuálně se v databázi webového rozhraní nachází přes 9000 receptů. RecyclerView je schopen zobrazit všechny tyto recepty najednou, ale největší překážkou je stažení všech receptů do mobilní aplikace. Samotné webové API je postavené tak, aby uživateli poskytovalo nejnovější recepty.

Při spuštění aktivity aplikace se stáhne nejnovějších 25 receptů. Po skrolování uživatele v seznamu na konec se vykreslí nové View indikující načítaní. Zároveň je vytvořen HTTP požadavek na stažení dalších 25 receptů. Nově stažené recepty jsou vloženy do listu v adapteru a vykresleny v RecyclerView.

Samotné nekonečné skrolování bylo postaveno na metodě, kterou implementuje RecyclerView. Jednalo se o vlastní implementaci posluchače na událost addOnScrollListener (Výpis 1). V této metodě bylo potřeba zjistit celkový počet položek v RecyclerView a poslední zobrazenou pozici položky uživatelem. Pokud byla poslední zobrazená položka společně s nastavenou proměnou indikující bod zlomu větší než celkový počet položek v seznamu, došlo ke stažení dodatečných dat.

```
public void onScrolled(RecyclerView recyclerView, int dx, int dy) {
      super.onScrolled(recyclerView, dx, dy);
      //celkovy pocet polozek
      int mTotalCount = mLinearLayoutManager.getItemCount();
      //pozice posledni polozky
      int mLastItem = mLinearLayoutManager.findLastVisibleItemPosition();
      if ((!isloading && mTotalCount <= (mLastItem + mThreshold)) &&
          ApplicationSetting.isConnected()) {
         if (onLoadMoreListener != null) {
            onLoadMoreListener.onLoadMore(paginator);
         }
         paginator++;
         isloading = true;
      }
}
```

Výpis 1: Detekce skrolování na konec seznamu

## 4.2 Vytvoření generické třídy pro stahování dat z webového API

Mobilní klient komunikuje s webovým rozhraním. Webové rozhraní poskytuje data ve formátu JSON. V prvních dnech praxe jsem stažené data v JSON formátu převáděl ručně do specifických tříd. Pro každou třídu jsem musel vytvořit vlastní dekodér, který prošel JSON a naplnil instanci konkrétního objektu. Poté, co bylo webové rozhraní několikrát upraveno, musel jsem také upravit dekodér. Zjistil jsem, že cesta ručního dekódování JSON formátu je značně neefektivní a zabere při změně API hodně času.

Pro výrazné zjednodušení práce a zrychlení vývoje byla vytvořena třída pro stahování a konverzi dat. Základní podstata třídy spočívala v automatickém převodu dat z JSON formátu do objektu uvedeného v parametru generické třídy. Třída je rozšířením Request třídy z knihovny

Volley od Google. Samotný proces konvertování JSON formátu do objektů využívá GSON knihovnu od Google a funguje na pozadí s využitím reflexe jazyka Java.

Tato třída se stala následně základem celé aplikace, zrychlila vývoj a umožnila přidávání nových funkcí za kratší čas.

## 4.3 Vyhledávání obsahu, vyhledávání s nápovědou

Realizace tohoto úkolu probíhala v obou semestrech. V první části jsem implementoval základní vyhledávání receptů, ingrediencí a článků. Pro řešení vyhledávání jsem využil základní komponentu SearchView. K zajištění správné funkčnosti komponenty jsem musel provést následující kroky.

### 4.3.1 Vytvoření rozhraní pro konfiguraci vyhledávání

Jedná se o XML dokument, který slouží pro správné nastavení vyhledávání. V rámci tohoto dokumentu je možné nastavit způsoby hledání, typ nápovědy během hledání, případně konfigurovat vyhledávání prostřednictvím hlasového vstupu. V aplikaci bylo využito základního nastavení obsahující pouze základní titulek a nápovědu pro uživatele. Správně nastavený konfigurační soubor byl následně použit v manifestu aplikace u vyhledávácí aktivity.

### 4.3.2 Zobrazení vyhledávající komponenty v aktivitě

Pro správné vykreslení a zajištění interakce s uživatelem je nutné do konfiguračního souboru aktivity přidat vyhledávající komponentu. Tento konfigurační XML kód definuje pořadí zobrazení konkrétního prvku pomocí atributu orderInCategory. Z uvedeného kódu je také poznat, jakou třídu komponenta využívá. Jedná se o atribut actionViewClass (Výpis 2). Hodnota atributu zaručuje, že vyhledávající komponenta bude využívat nejnovější verzi své implementace.

```
<item
        android:id="@+id/action_search"
        android:icon="@drawable/ic_search_white_24dp"
        android:orderInCategory="100"
        android:title="@string/action_search"
        app:actionViewClass="android.support.v7.widget.SearchView"
        app:showAsAction="ifRoom|collapseActionView"
        />
```

Výpis 2: XML kód pro zobrazení vyhledávací komponenty v aktivitě

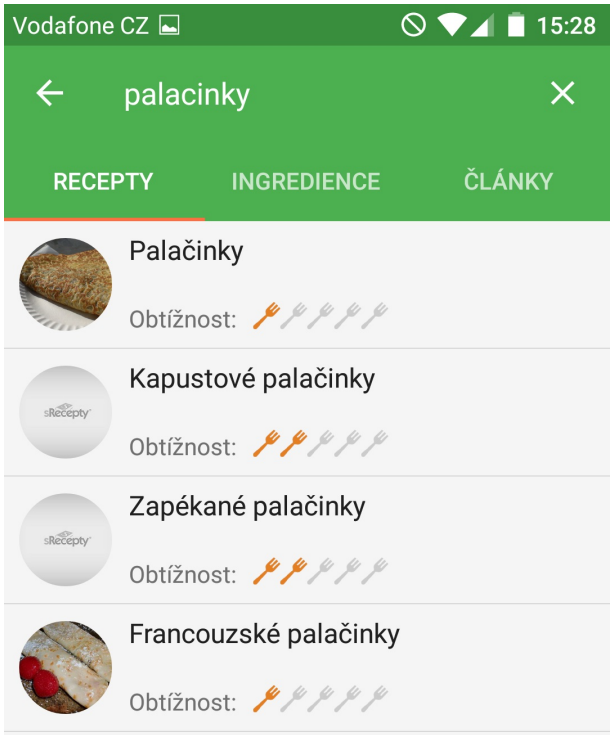
Následovalo propojení vyhledávací komponenty s konfiguračním souborem popsaným výše. Propojení bylo implementováno v aktivitě zajištující vyhledávání. Hlavním krokem bylo získání instance vyhledávací komponenty, která je lokalizovatelná přes ID z konfiguračního menu.

Získáná instance sloužila k nastavení konfigurace hledání v metodě setSearchableInfo. Pro zajištění správné funkčnosti je potřeba získat instanci konfigurace hledání. Instanci vyhledávací konfigurace je možné získat pomocí systémové služby SearchManager, konkrétně pomocí funkce getSearchableInfo.

Správné propojení vyhledávající komponenty s konfigurací vyhledání způsobí, že uživatel může vložit vyhledávací dotaz a získat požadované výsledky.

### 4.3.3 Vytvoření aktivity pro zobrazení výsledku hledání

Uživatel vloží vyhledávaný dotaz, po odeslání dotazu se vytvoří nová aktivita obsahující výsledky hledání (Obrázek 2). Požadavek byl, aby se v jedné aktivitě zobrazily výsledky pro recepty, ingredience a články. Při řešení tohoto požadavku jsem využil fragmentů. Fragment je komponenta v Androidu, která má vlastní životní cyklus. Aktivita může obsahovat několik takových fragmentů [1]. Fragmenty jsou vzájemně nezávislé.




Obrázek 2: Zobrazení výsledku hledání

Jakmile je spuštěna vyhledávací aktivita, systém předává také vyhledávaný dotaz. Vyhledávaný dotaz byl uložen do instanční proměnné v aktivitě. K proměnné se následně přistupovalo z fragmentů za použití reference na aktivitu.

Pro zobrazení výsledků hledání byly vytvořeny samostatné fragmenty. Každý fragment zobrazoval jiný typ výsledků a prováděl HTTP požadavky na webové API. Pokud uživatel kliknul

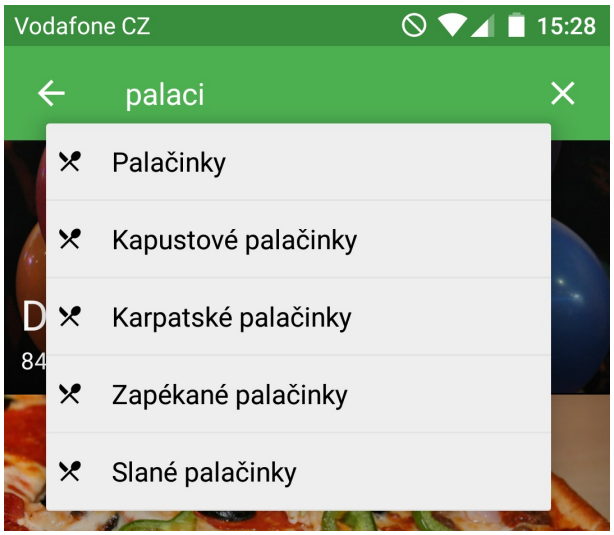
na zobrazenou položku, byl přesměrován na aktivitu obsahující detailní informace o vyhledaném záznamu.

### 4.3.4 Zobrazení nápovědy při vyhledávání

V druhé polovině praxe jsem dostal za úkol rozšířit vyhledávání o nápovědu. Nápověda se uživateli zobrazí během psaní dotazu. Pokud uživatel klikne na položku v nápovědě, bude přesměrován na detail záznamu (Obrázek 3).

Pro tento úkol jsem chtěl využít rozšíření vyhledávací komponenty o nápovědu. Po základní implementaci a zprovoznění tohoto rozšíření jsem zjistil, že uvedená cesta nebude fungovat. Při implementování jsem narazil na problém, který spočíval v tom, že do rozšíření nebylo možné vložit výsledky z webového API. Bylo nutné nalézt řešení vzniklého problému.

Řešením problému bylo využití komponenty ListPopupWindow. Jedná se o komponentu určenou k zobrazení seznamu položek v menu aktivity. Ke správné funkčnosti bylo potřeba implementovat vlastní třídu, která měla na starost zobrazování jednotlivých položek v této komponentě. Jednalo se o rozšíření třídy BaseAdapter z Android SDK.




Obrázek 3: Zobrazení nápovědy při psaní dotazu

Tato komponenta nebyla primárně určena pro tento typ úkolu, protože v základní verzi komponenty nebylo možné nastavit její rozměry a požadované odsazení. Komponenta pracovala pouze v pixelech, přičemž systém Android pracuje v density-pixel. Řešením problému bylo vytvoření funkce, která převáděla vložené density-pixely na klasické pixely.

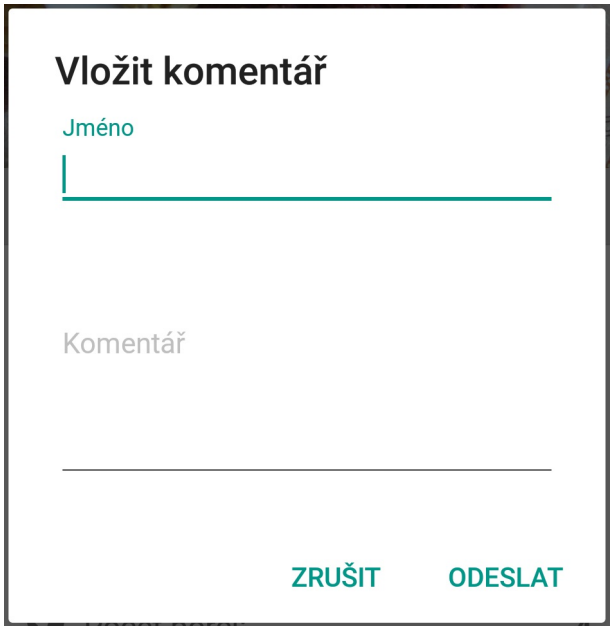
## 4.4 Vkládání komentářů a recenzí z mobilního klienta

Web sRecepty.cz patří mezi největší internetové kuchařky v České republice a má přesah do několika zemí světa. Jedná se o komunitní portál, jehož webové verze umožňují vkládat komentáře

a recenze návštěvníků webu. Stejnou funkcionalitu jsem implementoval v mobilním klientovi.

V prvním kroku jsem se snažil přidávání komentářů implementovat v nové aktivitě. Spouštění nové aktivity jsem následně zhodnotil jako značně neefektivní, rozhodl jsem se využít DialogFragment. DialogFragment představuje plovoucí okno v aktuální aktivitě (Obrázek 4).

V počáteční fázi implementace byla vytvořena třída, která je rozšířením třídy DialogFragment. Následovalo vytvoření základního View obsahujícího formulářové prvky určené pro vstup uživatele. Odkaz na vytvořené View se následně použil jako argument AlertDialogu. Právě AlertDialog umožňuje upravit prvky podle potřeby konkrétní aplikace. Systém následně komponentu po zavolání vykreslil.




Obrázek 4: Vložení komentáře

Při vkládání komentářů je nutné rozlišovat, zda je uživatel přihlášený. Přihlášený uživatel nemusel vkládat do formuláře své jméno. V rámci HTTP požadavku se u přihlášeného uživatele posílal jeho token. Webové rozhraní rozpoznalo, zda se jedná o přihlášeného uživatele a automaticky doplnilo jméno nalezené v databázi.

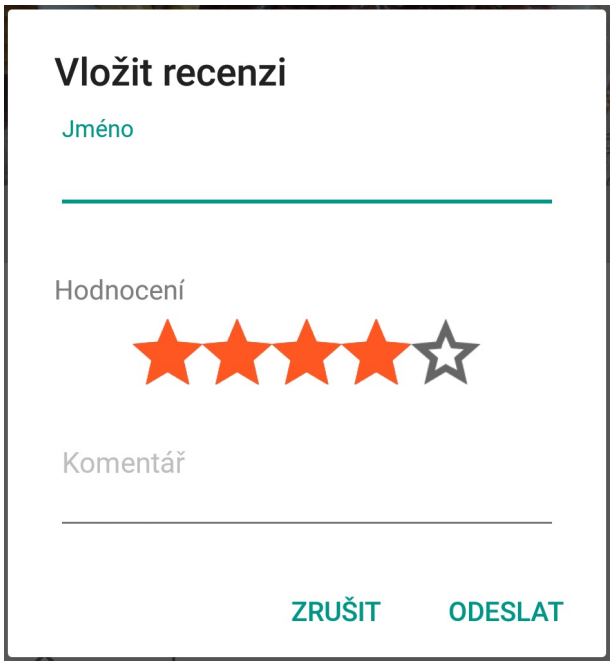
Před odesláním formulářových dat na webové API musela být provedena validace jednotlivých vstupů. Validace vstupů byla provedena nad jednotlivými prvky formuláře. Pokud nebyly položky správně vyplněny, uživatel byl upozorněn textem pod formulářovými prvky. Při implementování jsem narazil na problém. Pokud uživatel nevyplnil formulářový prvek a stiskl tlačítko odeslat, došlo ke zrušení celého dialogu pro komentáře.

Zjistil jsem, že tuto vlastnost způsobují právě základní tlačítka v DialogFragmentu. K odstranění rušení dialogu je potřeba nahradit automaticky vytvářená tlačítka za ručně vytvořená. Pro vytvořená tlačítka spravovat události po kliknutí. Správně implementovaná událost na kliknutí tlačítka po úspěšné validaci odeslala formulářové data na webové rozhraní. Přidaná tlačítka

ovšem nesplňovala Material Design a byla značně nekonzistentní s dosud použitým vzhledem aplikace. Posílání vložených dat na webové rozhraní probíhalo v aktivitě na základě posluchače na událost odeslání.

Podobným způsobem jsem implementoval také přidávání recenzí (Obrázek 5). Recenze obsahovala oproti klasickému komentáři RatingBar. RatingBar je vizuální komponenta Androidu určená právě pro hodnocení na zvolené stupnici.

Webová stránka zobrazuje hodnocení na stupnici od 0 do 5. Během realizace této funkce bylo využito již vytvořeného View, které pocházelo z komentářů. View bylo ovšem rozšířené o komponentu RatingBaru. Validace vstupů probíhala totožným způsobem jako v komentářích. K vytvoření HTTP požadavku bylo potřeba dat z formuláře a hodnocení. Data byla zaslána z aktivity receptů na základě posluchače na událost odeslání recenze.




Obrázek 5: DialogFragment pro vložení recenze

Během implementace obou výše uvedených požadavků bylo nutné také ošetřit jejich zobrazení v aktivitě receptů. Pokud v detailu konkrétního receptu nebyly komentáře, případně recenze, blok pro jejich vykreslení nebyl zobrazen. Zasláním požadavku na webové rozhraní došlo také k lokálnímu výpisu těchto dat.

## 4.5 Implementace notifikačního centra

Celé notifikační centrum bylo postaveno na technologiích Google, konkrétně Google Cloud Messaging. Cílem úkolu bylo umožnit příjem notifikačních zpráv zaslaných prostřednictvím

webové administrace. Součástí notifikačního centra je také schopnost odhlásit uživatele přihlášeného z mobilní aplikace. Celé notifikační centrum se skládá z několika částí.

Před samotnou implementací musela být provedena změna nastavení vývojového prostředí Android Studia, systému Gradle a implementace několika podpůrných knihoven. Součástí bylo také vložení konfiguračního souboru, který vygenerovalo webové rozhraní Google Cloud Messaging.

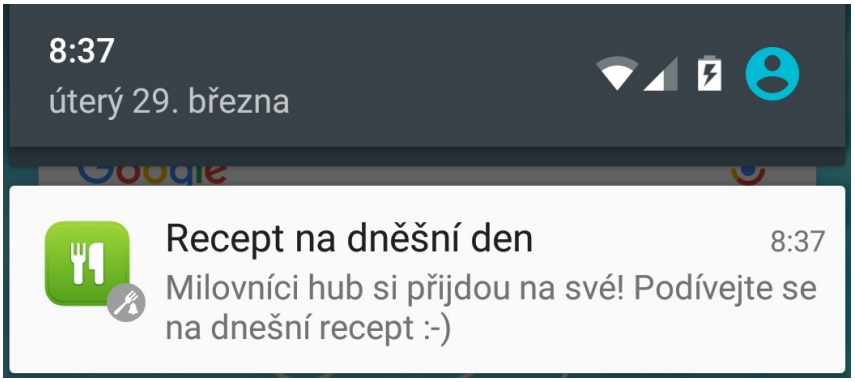
### 4.5.1 Registrační služba

Registrační služba představuje základ celé funkcionality notifikačního centra. Poté, co uživatel spustí aplikaci, dojde na pozadí ke spuštění registrační služby. Služba vygeneruje speciální ID spuštěné aplikace. ID je pro všechny instance jedinečné a umožňuje identifikovat konkrétní zařízení, na kterém aplikace běží. Získané ID bylo následně zasláno pomocí HTTP POST požadavku na webový server. Součástí požadavku byla také MAC adresa zařízení. Pokud byl uživatel přihlášen, zasílal se také token uživatele. MAC adresa je zasílána pro případ, kdyby došlo ke změně ID. Kdyby tento případ nastal, webové rozhraní rozpozná změnu vygenerovaného ID a uloží si nově přijaté.

Posílání notifikací probíhá právě na základě vygenerovaného ID. Administrátor vytvoří notifikační zprávu a odešle ji na servery Google Cloud Messaging. Servery následně podle ID automaticky zajistí poslání notifikační zprávy na konkrétní zařízení.

### 4.5.2 Služba pro příjem notifikací

Aby mobilní aplikace mohla přijímat notifikační zprávy, musela být vytvořena nová služba. Služba pro příjem zpráv byla rozšířením třídy GcmListenerService, která byla v dodávané knihovně Google Services. Po přijetí dat ze serverů Google je zavolána funkce onMessageReceived. Tato funkce slouží pro zpracování přijatých dat a následné zobrazení notifikace. V úvodu je popsáno, že aplikace je schopna vzdáleně odhlásit přihlášeného uživatele. Z tohoto důvodu byla provedena implementace kontroly přijaté zprávy. Pokud by se jednalo o zprávu určenou k odhlašování, aplikace by uživatele vzdáleně odhlásila. Funkcionalita vzdáleného odhlašování uživatele si také vyžádala kontrolu stavu aplikace. V rámci této kontroly se ověřovalo, jestli aplikace je spuštěna na popředí nebo je vypnutá. Na základě stavu aplikace došlo k vyvolání konkrétní akce. Druhou možností bylo právě zobrazení notifikační zprávy (Obrázek 6).




Obrázek 6: Notifikace

### 4.5.3 Zobrazení přijaté notifikace

O zobrazování informačních zpráv se stará funkce sendNotification. Příchozí data z požadavku obsahují pouze titulek, text a ID zprávy.

ID zprávy jsou předány jako argument Intentu. K vytvoření notifikační zprávy je využit objekt NotificationCompat.Builder. U objektu je možné nastavit široké spektrum vlastností. V aplikaci je použita funkce pro nastavení ikonek notifikace, titulku, samotného textu zprávy a základního notifikačního zvuku. Jako poslední argument je předán objekt, který po kliknutí na notifikaci přesměruje uživatele na novou aktivitu.

Při implementování jsem narazil na chybu, která se projevovala následujícím způsobem. Pokud z webového rozhraní bylo zasláno několik notifikací za sebou, v mobilní aplikaci se to projevilo tak, že nově přijatá notifikace překreslila původní. Zároveň ovšem po kliknutí na nově přijatou notifikaci došlo k přesměrování na původní.

Problém byl způsoben tím, že platforma Android si v paměti uchovává všechny Intent události. Pro řešení problému bylo nutné používat označování událostí jedinečným identifikátorem. Zároveň identifikátor intent události sloužil jako identifikátor notifikace. Zavedení identifikátoru způsobilo, že se notifikace nepřekreslily a zároveň každá notifikace měla unikání data pro Intent.

Vytvořený objekt notifikace NotificationCompat.Builder byl předán společně s unikátním identifikátorem službě pro správu notifikací. Systém se následně postaral o notifikaci uživatele grafickým a zvukovým způsobem.

### 4.5.4 Detail notifikace

Uživatelovi se zobrazila v horní liště ikonka signalizující notifikování uživatele. Jestliže uživatel provedl gesto pro zobrazení notifikace, došlo k vykreslení titulku a textu notifikace. Uživatel na tuto notifikaci také mohl kliknout. Kliknutí zapříčinilo přesměrování uživatele na novou aktivitu určenou pro zobrazení detailu notifikace. Detail notifikace byl vytvořen v několika variantách na základě údajů nastavených přes webové rozhraní.

První variantou bylo zobrazení klasické notifikace obsahující pouze titulek a text zprávy. Speciálním požadavkem byla schopnost detailu notifikace vykreslovat HTML značky. Během implementace tohoto požadavku jsem narazil na zajímavou vlastnost. Komponenty na platformě Android umí vykreslit HTML pouze pro formátování textu, odstavců, obrázků a tabulek.

Druhou variantou bylo zobrazení notifikace obsahující stejné prvky jako původní varianta. Nově přidaným prvkem bylo tlačítko. Tlačítko slouží k několika možným scénářům. Primárním účelem tlačítka bylo přesměrování uživatele na webovou stránku, jejíž adresa je získána přes HTTP odpověď. Další účel tlačítka spočíval v přesměrování uživatele na jinou aktivitu. Mezi aktivity, na které mohl být uživatel přesměrován, patří detail receptu, ingredience a detail článků.

Vlastností detailu notifikace je to, že aktivita mohla být spuštěna i když hlavní aplikace neběžela. Pokud uživatel kliknul na tlačítko zpět, došlo k přesměrování na přehled notifikací.

### 4.5.5 Přehled notifikací

Přehled notifikací představuje aktivitu obsahující celou historii přijatých notifikačních zpráv. Charakteristickou vlastností aktivity je, že na základě HTTP požadavku poskytuje jiná data pro přihlášeného a nepřihlášeného uživatele. Primární funkci aktivity plní opět adapter. Adapter převádí přijatá data z HTTP požadavku na jednotlivé položky View. Aktivita obsahuje RecyclerView pro vykreslení těchto položek.

## 4.6 Implementace kategorie receptů

V první veřejné verzi aplikace mohli uživatelé vyhledávat recepty pomocí vyhledávání nebo nekonečného skrolování v přehledu receptů. Na základě těchto poznatků a kvůli zvýšení uživatelské přívětivosti jsem musel v druhé polovině praxe implementovat kategorie receptů. Cílem bylo implementovat podobné chování kategorií z webové verze do mobilní verze.

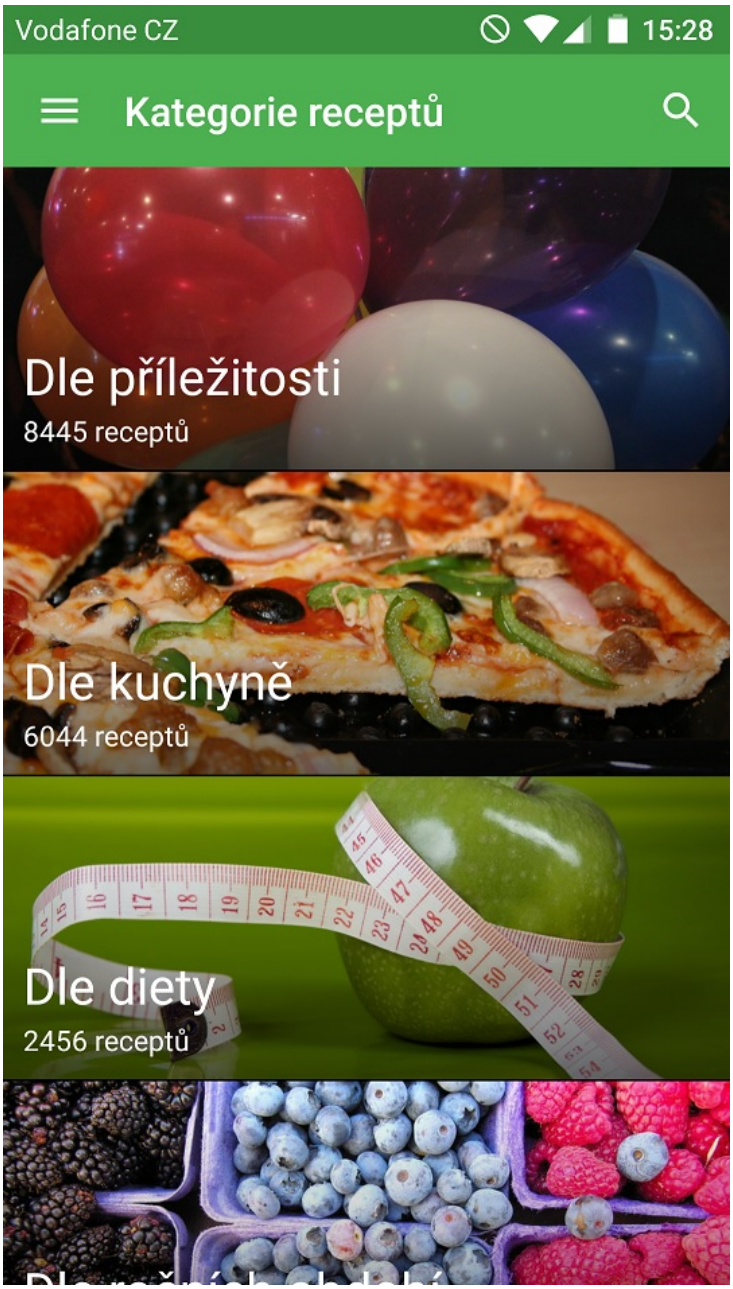
Při realizaci kategorií bylo potřeba vytvořit View (Obrázek 7) představující právě vzhled kategorie. Zajímavostí je, že se jedná pouze o obrázek symbolizující kategorii, titulek popisující kategorii a indikátor počtu receptů. Přes celou plochu vytvořeného View je aplikován průhledný přeliv barev bílé a šedé. Přeliv měl za úkol způsobit dobrou čitelnost textu na případném bílém obrázku.


Obrázek 7: Položka zobrazující kategorii

Poté, co uživatel spustí aplikaci, je vytvořen HTTP požadavek. Po úspěšném stažení dat se vykreslí jednotlivé kategorie receptů (Obrázek 8). Uživatel může kliknout na libovolnou kategorii. Právě událost kliknutí byla implementována tak, aby zpracovala následující scénáře. Z přijaté HTTP odpovědi je možné získat informace o existenci podkategorie. Pokud konkrétní kategorie obsahovala několik dalších kategorií, rekurzivně se zavolala funkce vykreslení nového přehledu kategorií. V druhém případě byl uživatel přesměrován na aktivitu obsahující recepty konkrétní kategorie.

V aplikaci bylo nutné u každého HTTP GET požadavku zasílat v hlavičce informaci, je-li zařízení připojené na WiFi a jedná-li se o tablet. Zařízení, které je aktuálně připojené na WiFi, získá po odeslání HTTP požadavku obrázky s vyšším rozlišením. Nastavení je plně v režii webové administrace.

Další funkčnost, která byla implementována na základě požadavků uživatelů, se týkala automatické změny výšky View. Výška View se měnila podle typu zařízení. Systém automaticky rozpoznal typ zařízení a změnil výšku View podle hodnot nastavených pro konkrétní zařízení. Mobilní telefony mají výšku nastavenou na nižší velikost, tablety na velikost vyšší.




Obrázek 8: Aktivita s přehledem kategorií

# 5 Dovednosti a znalosti uplatněné v průběhu praxe

Při vykonávání odborné praxe ve společnosti RailsFormers s.r.o. jsem využil znalosti z následujících předmětů.

- Programovací jazyky I
  - Tento předmět mi poskytnul základní znalosti jazyka JAVA, kterou jsem využil během programování pro platformu Android.
- Programování 2
  - Programování 2 mi poskytlo základní znalost o objektově orientovaném přístupu při programování aplikací.
- Tvorba aplikaci pro mobilní zařízení 2
  - Souběžně s vykonáváním praxe jsem také absolvoval tento předmět. Pomohl mi získat dodatečné informace o vývoji aplikací pro Android platformu. Osobně hodnotím tento předmět jako nejvíce přínosný pro mou praxi.
- Vývoj informačních systémů
  - Předmět VIS mi poskytl znalosti o použití návrhových vzorů. Znalosti jsem využil pro vhodné navržení struktury komponent aplikace.

# 6 Dovednosti a znalosti scházející v průběhu praxe

Na praxi jsem nastoupil pouze se školními znalostmi platformy Java. V rámci odborné praxe jsem programoval pro operační systém Android, který je na této platformě postaven. Za největší nedostatek považuji právě neznalost systému Android. Zejména architekturu systému a principy vývoje pro tuto platformu.

Další problematickou oblast vidím v neznalosti práce s verzovacími systémy, které jsou důležité pro práci v týmu. Na odborné praxi jsem se setkal s možností verzování souborů poprvé. Na začátku praxe mi byly vysvětleny základní principy tak, abych mohl pracovat v menším týmu.

Poslední oblast, ve které jsem měl omezené zkušenosti, byla práce v týmu. Doposud jsem pracoval na projektech pouze sám.

# 7 Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení

Na odbornou praxi jsem nastoupil bez jakýchkoliv zkušeností s vývojem aplikací. Disponoval jsem pouze znalostmi získanými prostřednictvím studia a sebevzděláváním.

V rámci odborné praxe jsem si prošel kompletním vývojovým cyklem softwaru. Cyklus začínal definováním požadavků, u kterých jsem musel získat jejich bližší specifikaci a končil vydáním první veřejné verze mobilní aplikace. Vývoj na praxi probíhal na základě projektového řízení, které spočívalo v rozdělení projektu do větších úkolů.

Na začátku odborné praxe došlo k seznámení s ostatními kolegy ve společnosti a přidělení konzultanta. Konzultant mne pak seznámil s projektem, jenž byl náplní praxe. Cílem bylo vytvořit mobilní aplikaci pro platformu Android. Aplikace měla být mobilní verzí online kuchařky receptů – sRecepty. Byl jsem součástí tříčlenného týmu složeného ze dvou Android vývojářů a vývojáře spravujícího webové rozhraní.

Vývoj pro platformu Android probíhal relativně bez problému. Největší problémy vznikly primárně díky mé neznalosti platformy Android. Osobně musím říct, že některé části dokumentace Androidu jsou zastaralé a neposkytují správné informace. Komunikace s členy týmu a konzultantem byla bezproblémová.

Bakalářská praxe realizována v první části semestru měla za cíl vytvořit první verzi mobilní aplikace se základní funkčností. Aplikace obsahovala základní přehled receptů, základní vyhledávání a funkcionalitu implementovanou dalším kolegou. Praxe v druhé části semestru měla za úkol rozšířit funkčnost aplikace. Rozšíření bylo primárně zaměřeno na zvýšení přívětivosti pro uživatele. V této fázi byly přidány kategorie, vyhledávání bylo rozšířeno o nápovědu. Součástí této fáze bylo zprovoznění notifikačního centra.

První verze aplikace dosáhla okolo 3000 stažení a měla přibližně 1000 aktivních uživatelů. Vydání druhé verze způsobilo růst aktivních uživatelů na více jak 2000. Aplikace patří ve své kategorii mezi nejlépe hodnocené.

# 8 Závěr

V úvodu bakalářské práce byly popsány důvody, které vedly k absolvování odborné praxe u společnosti Railsformers s.r.o. V části určené pro popis společnosti a pracovní pozice je popsána charakteristika firmy. Větší důraz je kladen na popis technologií, které společnost využívá. Druhá část práce je věnována popisu jednotlivých úkolů přidělených během praxe.

Třetí část práce se zaměřuje na popis implementace jednotlivých úkolů. Při popisování postupu implementace jednotlivých úkolu byly také uvedeny problémy, které nastaly. Následně byly uvedeny příčiny problémů a kroky vedoucí k jejich odstranění. Mezi nejvíce implementačně rozsáhlé úkoly patří notifikační centrum a vyhledávání.

Nejvíce časově náročný úkol představovalo právě notifikační centrum. Implementace spočívala hlavně ve vytvoření služeb běžících na pozadí operačního systému a dodatečných aktivit vykreslujících obsah uživateli. Velmi specifickým požadavkem bylo vzdálené odhlašování uživatelů na základě přijaté notifikace. Jedním z důvodů, proč bylo notifikační centrum implementováno, byla snaha firmy o vynucení spuštění aplikace uživatelem.

Vyhledávání obsahu představovalo také velmi časově rozsáhlý úkol. Realizace probíhala v obou částech praxe. Funkce vyhledávání obsahu patří mezi nejvíce používané funkce právě z důvodu velkého počtu receptů, kterým webové rozhraní disponuje. Základní vyhledávání funguje klasicky na principu vložení dotazu a jeho odeslání. Dodatečně implementovanou funkcí byla nápověda zobrazující výsledky na zadaný dotaz před odesláním.

Absolvováním odborné praxe jsem získal reálné pracovní zkušenosti z IT odvětví. Od základu jsem společně s dalším kolegou navrhnul a vytvořil funkční aplikaci, kterou nyní používají tisícovky uživatelů. Naučil jsem se pracovat v týmu a rozšířil přehled o mobilní platformě Android.

# Literatura

[1] Allen, Grant, *Android 4: průvodce programováním mobilních aplikací,* Brno: Computer Press, 2013. ISBN 978-80-251-3782-6.

[2] Railsformers.com, Railsformers.com [online], Ostrava, © 2016 [cit. 2016-04-06]. Dostupné z: http://railsformers.com/cs

[3] RecyclerView | Android Developers. RecyclerView [online], 2016 [cit. 2016-02-10]. Dostupné z: http://developer.android.com/reference/android/support/v7/widget/RecyclerView.html