FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	EcoTracker pro Android
Student:	David Holkup
Vedoucí:	Ing. Filip Glazar
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce zimního semestru 2021/22

Pokyny pro vypracování

Cílem této bakalářské práce je navrhnout a implementovat aplikaci pro mobilní systém Android. K funkčním požadavkům na aplikaci patří sběr statistik o vyprodukovaném odpadu, jako je například komunální odpad atp. K dalším funkcím aplikace patří správa a přehled událostí o ekologických akcích, jako je například hromadný úklid daného města. Uživatelé mezi sebou mohou sdílet různé nápady pro ekologičtější životní styl.

1. Proveďte analýzu existujících řešení.
2. Na základě provedené analýzy navrhněte vlastní softwarové řešení.
3. Naimplementujte prototyp aplikace.
4. Aplikaci podrobte řádnému testování.
5. Navrhněte potřebné úpravy, které vzešly z testovaní.
6. Připravte produkční verzi aplikace.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 26. února 2020

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

ECO Tracker

David Holkup

Katedra Softwarového inženýrství
Vedoucí práce: Ing. Filip Glazar

28. května 2020

Poděkování

Děkuji svému vedoucímu práce Ing. Filipu Glazarovi za vedení a cenné rady při tvorbě této práce.
Své přítelkyni a rodině děkuji za podporu a trpělivost.
Svému bratrovi Dominikovi děkuji za nápad a inspiraci při při tvorbě tématu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principu při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisu. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisu, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programu, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli zpusobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelum). Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 28. května 2020 . . . . . .

Odkaz na tuto práci

Holkup, David. *ECO Tracker*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

Abstrakt

Cílem této bakalářské práce bylo navrhnout a naimplementovat aplikaci ECO Tracker pro mobilní systém Android. K funkčním požadavkům aplikace patří sběr dat o vyprodukovaném odpadu a následné zobrazování statistik. K dalším funkcím aplikace patří správa a přehled událostí o ekologických akcích. Uživatelé mezi sebou mohou sdílet jak tyto události, tak i různé nápady směřující k ekologičtějšímu životnímu stylu.

Aplikaci jsem naimplementoval. Výsledkem této práce je funkční prototyp s mnoha funkcemi. Tato práce poskytla odpovědi na důležité otázky a ustanovila základ, na kterém staví další fáze projektu.

Klíčová slova Ekologické aktivity, Životní styl, Android, Mobilní aplikace, Kotlin, ECO Tracker

Abstract

The goal of this bachelor thesis was to design and implement mobile Android application ECO Tracker. Functional requirements include data collection of produced waste. Showing user statistics about this data. Another function of ECO Tracker is administration and overview of ecological events. Users can share these events or their various ideas taht should lead to more ecological life style.

I've implemented the app. Result of this effort is functioning prototype with many features. This thesis provided answers to important questions and established the base, which following phases of the project build up on.

Keywords Ecologic Activities, Life-style, Android, Mobile app, Kotlin, ECO Tracker

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Lidstvo ohrožuje mnoho věcí – nejvíce však lidstvo samo.

Svět trpí znečištěním a s touto situací je třeba něco udělat, proto jsem se rozhodl přispět k řešení tohoto problému.
Motivací mi bylo to, že každý den při cestě do školy nebo do práce vidím na zemi kolem sebe množství odpadu. I když ho jeden den po cestě seberu, druhý den je tam znovu. Je potřebné, aby svoje chování změnilo více lidí.
Proto vznikl ECO Tracker, který má uživatele vzdělat tak, aby se sami snažili tuto situaci napravit.

ECO Tracker bude mobilní Android aplikace. Tuto platformu jsem zvolil, protože je velice rozšířená. V květnu roku 2019 Google na své „I/O Keynote“ konferenci uvedl počet aktivních Android zařízení, bylo jich více než 2,5 miliardy kusů [1]. Mám tak možnost oslovit a ovlivnit velkou skupinu lidí.

Tato bakalářská práce je první fází projektu ECO Tracker. Jejím účelem je poskytnout kvalitní základ na kterém bude možné dále stavět. Během tohoto procesu si ujasním priority funkcionalit, design aplikace a celou architekturu projektu.

V práci se budu zabývat následujícím:

1. Nejdříve vytyčím cíle kterých chci dosáhnout.
2. Analyzuji existující ekologické aplikace.
3. Porovnám analyzované aplikace se svým návrhem projektu a případně ho upravím.
4. Představím konečný návrh řešení.
5. Popíši postup implementace jednotlivých částí aplikace. Zejména se budu věnovat jejím problematickým částem.
6. Na závěr zhodnotím jak byly vytyčené cíle splněny a nastíním oblasti, kterými by se mohl zaobírat další vývoj aplikace.

Kapitola 1

Cíl práce

Cílem práce je vytvoření mobilní aplikace pro operační systém Android. Důležité bude nabádat uživatele k zlepšení svého životního stylu a vzdělat ho v oblasti ochrany životního prostředí.

Aplikace bude rozdělená do pěti částí – Formulář (Form), Statistiky (Statistics), Události (Events), Nápady (Ideas) a Profil (Profile).
V sekci Formulář bude uživatel vyplňovat kolik vyprodukoval jakého odpadu. Uživateli budou k dispozici statistiky vytvořené na základě vyplněných údajů. Data o spotřebě budou zobrazena pro každý druh odpadu zvlášť. Statistiky budou uživateli prezentovány formou grafu a textového popisu.
V sekci Události bude přehled nadcházejících ekologických událostí. Uživatel bude moci události vytvářet a sdílet je s ostatními uživateli. Pokud se uživatel rozhodne k události připojit, automaticky se tato událost přidá do jeho výchozího android kalendáře.
Sekce Nápady nabídne přehled ekologických nápadů, postřehů a zajímavostí, které do aplikace vložili ostatní uživatelé. Uživatel na nápady bude moci reagovat, bude je moci sdílet a také vytvářet vlastní nápady. Aplikace bude podporovat i úpravu vzhledu nápadů.
Uživateli budou zobrazeny pouze nápady a události, které jsou napsány v jednom z jazyků, kterým uživatel rozumí. Uživatel si bude moci přidat jazyky, které ovládá, v sekci profil. V té uživatel najde také události a nápady, které vytvořil. Bude je moci upravit nebo smazat. Bude zde mít i možnost přizpůsobit si aplikaci, například si zvolí jednotky, ve kterých budou zobrazovány statistiky.
Aplikace bude zobrazovat jednou denně notifikaci, ve které se uživatele zeptá, zda vynášel odpad a jeho odpověď uloží ke statistickým datům.

Kapitola 2

Analýza a návrh

2.1 Důležité pojmy

2.1.1 Aktivita

Aktivity jsou základní kameny každé Android aplikace. Aktivita je často připodobňovaná k jedné její obrazovce. Stará se o nějakou ucelenou část aplikační logiky [2]. V případě ECO Trackeru je to například aktivita na vytvoření události.
Každá aktivita má svoje okno, ve kterém je definován její vzhled. Při vytváření aktivity dojde i k vytvoření jejího okna a navázání tohoto okna k window manageru (manažeru oken), který ho bude dále spravovat [2].

Dle android dokumentace [2] má aktivita čtyři stavy.

- Pokud je aktivita na popředí, je nazývána aktivní nebo také běžící.
  Toto je aktivita, se kterou uživatel právě interaguje.
- Jestliže je aktivita stále prezentovaná uživateli, ale už nemá focus (pozornost), nazývá se viditelná.
  Do tohoto stavu se dostane běžící aktivita například pokud její část překryla jiná aktivita.
  V této fázi si aktivita stále udržuje svůj vnitřní stav a její okno zůstává připojeno k window manageru.
- Pokud je aktivita kompletně zakryta jinou aktivitou, nazývá se zastavená nebo také skrytá.
  Stále si udržuje svůj vnitřní stav, ale už není viditelná – její okno je skryto.
  Taková aktivita bývá systémem často zničena, pokud systém potřebuje paměť, kterou aktivita okupuje.
- Aktivita se nazývá destroyed (zničená) pokud ji systém uvolnil z paměti.
  Když je aktivita znovu zobrazena, musí být kompletně restartována a obnovena do svého předchozího stavu.

.

2.1.2 Fragment

Fragment je část uživatelského rozhraní, která může být vložena do aktivity.
Vývojař použije fragment, pokud potřebuje jemnější členění aplikace než pouze na aktivity, nebo pokud potřebuje použít stejnou funkcionalitu na více místech.
Fragmenty jsou úzce spojeny s jejich obalující aktivitou. Mají svůj vlastní life-cycle (koloběh života), ten je ale závislý na aktivitě fragmentu.
Pokud je například aktivita zastavena, nemůže v ní být vytvořený žádný fragment nebo pokud je aktivita zničena, všechny její fragmenty jsou zničeny spolu s ní [3].

Fragmenty nemusí být pouze uvnitř aktivity, mohou se také vnořovat. To znamená, že i fragment může obsahovat jiné fragmenty [3].

2.1.3 Shared preferencies

Shared preferencies (sdílené preference) slouží k ukládání dat na Android zařízení ve formátu klíč – hodnota. Jsou vhodná zejména k ukládání malého množství dat.
Aplikace může mít více shared preferencies. Například každá aktivita může spravovat své vlastní shared preferencies, to ale není podmínka.
Shared preferencies mohou být veřejné nebo soukromé. K soukromým má přístup pouze aplikace, která je vlastní a aplikace s root právy [4], naopak k veřejným mají přístup všechny aplikace.

Android nabízí i další formy ukládání dat. Aplikace může svá data také ukládat do souborů nebo do lokální databáze [5].
Tyto způsoby ukládání dat ale ECO Tracker nepoužívá.

2.1.4 Rest a API

API *Application Programming Interface* (aplikační programovací rozhraní) definuje způsob komunikace programů nebo jejich částí. Například pro komunikaci se serverovou částí aplikace bude ECO Tracker používat Rest API [6].

Svoje APIs definuje i operační systém Android [7]. Z těchto budu používat například API pro získání a úpravu shared preferencies, automatické zobrazování notifikací v určený čas a další.

2.1.5 Spring framework

Spring framework činí vývoj enterprise java aplikací jednodužší [8]. Tento framework nabízí velké množství různých modulů. Pro ECO Tracker budu používat modul Spring Boot [9], který předkonfiguruje ostatní spring moduly, činí tak vývoj ještě jednodušší a aplikaci mohu „*prostě spustit*“ [8]. Dále použiji Spring Security [10], pomocí kterého zajistím správný proces autentizace a autorizace.

2.1.6 Object/Relational Mapping (ORM)

„*ORM je proces ukládání objektu do relační databáze. ORM přemosťuje mezeru mezi objektem a relačním prostředím, čímž umožňuje objektově orientovaným aplikacím ukládat objekty jednoduše [...] obsahuje konverzi objektu do a z relačního formátu.*“ [11]

2.1.7 Hibernate

„*Hibernate je jedna z těch implementací ORM, která má vysokou výkonnost a dodatečné funkce jako podpora dual cache, objektově orientovaný dotazovací jazyk (HQL), podpora pro dotazování objektů a podpora transakcí.*“ [11]

2.1.8 Maven

Maven je sestavovací nástroj od společnosti Apache [12]. Byl vyvinut za účelem „*učinit proces sestavování jednodužším, nabídnout uniformní sestavovací nástroj, nabídnout kvalitní informace o projektu a podpořit lepší vývojařské návyky.*“ [13]

2.1.9 Gradle

Gradle je daší nástroj sloužící k sestavování projektu a správě jeho dependencies (závislostí), je to výchozí sestavovací nástroj android aplikací [14].

2.1.10 Android SDK

Zkratka SDK *Sofware Development Kit* je soubor nástrojů nutných pro proces vytváření programů. Android SDK obsahuje potřebné knihovny, debugger, android emulátor a dokumentaci android APIs.
S každou verzí operačního systému je zveřejněna i nová verze SDK [15].

2.1.11 HTTP a HTTPS

HTTP *Hypertext Trasfer Protocol* je protokol používaný webovými servery a prohlížeči ke vzájemné komunikaci přes internet [16].
HTTPS [17] je zabezpečený HTTP protokol (S jako secure). HTTPS šifruje komunikaci mezi servery a prohlížeči pomocí protokolu TLS nebo SSL [18].

2.1.12 Softwarová architektura

Jedna z definic pojmu Softwarová architektura zní: „*Softwarová architektura systému je množina struktur potřebných k uvažování o tomto systému. Zahrnuje softwarové prvky, vztahy mezi nimi a jejich vlastnosti.*“ [19].

2.1.13 Batch server

Batch server je prostředí ve kterém jsou spouštěny batch (dávkové) programy. Tyto programy (nebo skripty) jsou spouštěny pravidelně podle určitého plánu (každé ráno, 1. den v měsíci a podobně) [20].

2.2 Analýza konkurence

V rámci analýzy konkurence jsem mimo jiné vyzkoušel mobilní aplikace Recycle Coach [21], iRecycle [22] a My Little Plastic Footprint [23].

Recycle Coach funguje pouze v některých městech USA. Svým uživatelům radí, co se recyklovat dá a co ne, jaký výrobek patří do jakého kontejneru a kde takový kontejner nalézt. Připomíná jim, kdy se sváží odpad a zobrazuje jim různé tipy a nápady jak více recyklovat a produkovat méně odpadu [21].

Aplikace iRecycle, stejně jako Recycle Coach, radí uživateli při rozhodování jaký odpad patří do jakého kontejneru. Tato aplikace dokonce uživateli zobrazí seznam nejbližších odpovídajících kontejnerů a k vybranému kontejneru uživatele naviguje [22].

My Little Plastic Footprint uživateli prezentuje ekologické alternativy běžných každodenních produktů. Uživatel si může zvolit oblast, ve které se chce zlepšit a později sdělit aplikaci, zda místo běžného neekologického produktu začal používat ekologický. Aplikace na základě těchto dat počítá jakou „*plastovou stopu*“ uživatel má [23].

Většina ostatních aplikací je zaměřená na specifickou zemi. V žádné jsem nenašel možnost sdílet svoje vlastní nápady nebo ekologické události. Tuto funkcionalitu sice nabízí například česká webová stránka Ukliďme Česko [24], ta je ale zaměřena pouze na Českou republiku a v současné době (listopad 2019) nenabízí mobilní aplikaci.

Mnou vyzkoušené ekologické aplikace uživateli radí, jaký odpad patří do jakého kontejneru, kde se takové kontejnery nachází, který den se vyváží odpad a podobně.
To moje aplikace dělat nebude. ECO Tracker má být zaměřen na rozvoj jednotlivců, zobrazovat jim statistiky o tom, jak si vedou v porovnání s ostatními a jak se zlepšují. Má jim dát možnost sdílet své vlastní nápady a tím motivovat i své okolí.

2.3 Název aplikace

Protože zlepšení kvality životního stylu uživatele a následné zlepšování kvality životního prostředí je hlavním účelem této aplikace, rozhodl jsem se ji pojmenovat ECO Tracker. Podobá se totiž různým fitness trackerům, které také – za účelem zlepšení životního stylu uživatele – monitorují jeho stav, zobrazují mu statistiky o jeho činnosti, dávají mu výzvy a hodnotí ho.
ECO Tracker bude činit to samé, pouze v jiné oblasti. Navíc bude obohacen o možnost sdílení nápadů a událostí.

Název ECO Tracker už je ale používán. Našel jsem dvě mobilní aplikace s tímto názvem.

- ECO-Tracker má 5+ stažení z Google Play. Přiložený popis zní „*To track your ECO NFC labeled devices*“ [25]
- Eco-Tracker má 100+ stažení, je stále v režimu early access. Naposledy byl aktualizován 14. listopadu 2017. Jeho popis zní „*An application to help you recycle easier*“ (z Google Play již odstraněn).

Obě aplikace jsem si stáhl. První byla v jazyce, který jsem nerozpoznal, obsahovala textové pole a tlačítko.
Druhá aplikace disponovala obrázkem kontejneru a šesti tlačítky, z nichž pět bylo nefunkčních a jedno otevřelo statický seznam recyklovatelných produktů. Ani jedna z těchto aplikací není aktivní, nepovažuji je proto za konkurenci.

Název ECO Tracker se používá i jako název webových stránek.

- Webová stránka eco-tracker.com společnosti Genesis Now [26] nabízí sběr informací o spotřebě energie, vody a produkci odpadu firmám. Následně jim emailem posílá závěry z těchto dat a grafy.
- Ecotracker.org patří univerzitě v Princetonu a slouží jako propagační stránka mobilní aplikaci Princeton Eco Tracker [27], kterou se mi nepodařilo přes běžné vyhledávání v Google Play najít.
- Stránka s názvem Bentallgreenoak.ecotracker.com patří společnosti BentallGreenOak, která se zabývá obchodem s nemovitostmi [28]. Smysl této stránky se mi nepodařilo dohledat.

Pro získání přístupu k aplikaci Princeton Eco Tracker je třeba se zaregistrovat, vyplnit dotazník, v němž mnoho otázek je specifických pro danou školu (Princeton University). Aplikace samotná funguje tak, že dává uživateli měsíční ekologické výzvy. Ten si splnění úkolů vyfotí, fotky z aplikace odešle a následně se mu přičtou body. Na konci měsíční výzvy může získané body vyměnit za reálné ceny.
Tento nápad se mi líbí. Týdenní a měsíční výzvy chci do aplikace také přidat.

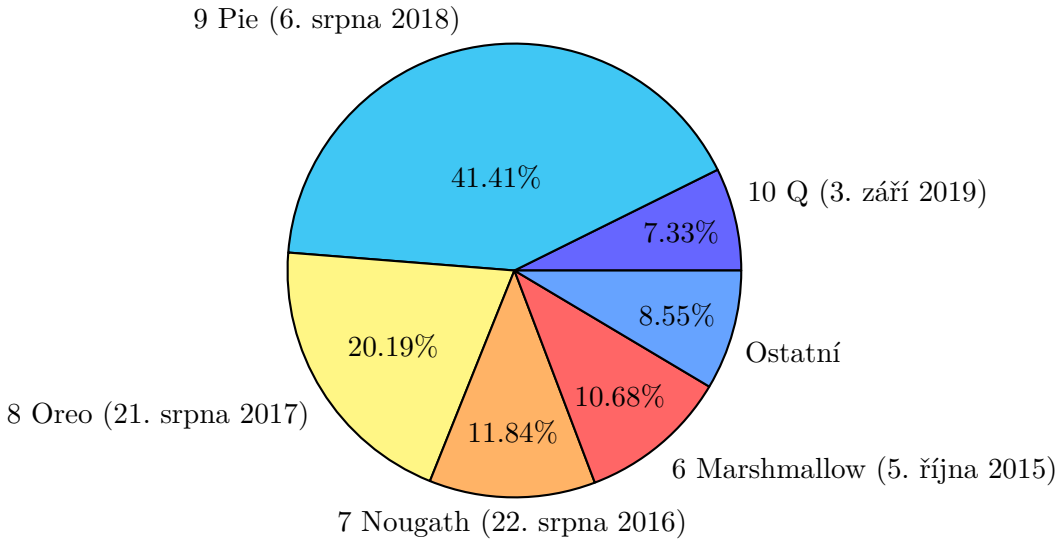
Záměr společnosti Genesis Now a její stránky `eco-tracker.com` se mému projektu podobá nejvíce.
Mezi společné rysy patří sběr dat o spotřebě energií, vody a o produkci odpadu, analýza těchto údajů a následná tvorba statistik z nich. Stejně jako mnou navrhovaná aplikace si i tato stránka klade za cíl zlepšit hospodaření svých klientů.
Rozdíly jsou v cílových skupinách. `Eco-tracker.com` cílí na firmy zatímco ECO Tracker bude sloužit běžným lidem jako jednotlivcům. Dalším rozdílem je cena. Moje aplikace bude přístupná zdarma, naproti tomu služby společnosti Genesis Now jsou zpoplatněné.

Na stránkách World Intelectual Property Organisation [29] vidím, že vlastníci této webové stránky měli zakoupenou i ochrannou známku, ale ta už není aktivní.
Jelikož máme rozdílné cílové skupiny a moje aplikace bude nezisková budu zvolené jméno ECO Tracker používat i nadále. V případě jakýchkoli sporů aplikaci přejmenuji.

2.4 Verze Android SDK

Při tvorbě programu pro Android je třeba brát v úvahu fakt, že velké množství zařízení nemá aktuální verzi systému, jak je vidět na obrázku 2.1 [30]. Největší zastoupení má typicky jeden až dva roky stará verze. Programátor tak musí dělat kompromisy mezi množstvím zařízení, která bude schopen podporovat a mezi funkcionalitou, kterou může použít.

Abych obsáhl větší množství potenciálních uživatelů, rozhodl jsem se podporovat zařízení od verze 6.0 Marshmallow a pokrýt tak zhruba 91% zařízení [30]. Počítám ovšem s tím, že než bude aplikace napsaná a vydaná, toto číslo se ještě zvýší.
Používám tak Android SDK 23,což pro mě znamená absenci podpory Javy 8. Android Marshmallow podporuje nejvýše Javu 7.



Obrázek 2.1: Zastoupení verzí systému Android (březen 2020)

2.5 Java versus Kotlin

ECO Tracker má být mobilní aplikace pro operační systém Android. Ten podporuje aplikace napsané v jazycích Java a Kotlin.

2.5.1 Java

V čase psaní této bakalářské práce (březen 2020) je poslední stabilní verze jazyka Java verze 14 [31]. Protože jsem se ale rozhodl podporovat všechna android zařízení s verzí operačního systému začínající na verzi 6.0 Marshmallow bude ECO Tracker moci používat maximálně Javu verze 7 (také 1.7) [32]. Tato kapitola tedy pojednává o Javě 7 a nepokrývá funkce vydané v pozdějších verzích.

Java je populární, velmi rozšířený a všeobecně uznávaný jazyk. Vznikl již v roce 1995, verze 1.7 byla vydaná v roce 2011 [33]. Je to jazyk, ve kterém je Android napsaný a existuje pro něj velmi široká škála knihoven, návodů a studijních materiálů.

Java je jazyk objektově orientovaný. Zdrojový kód je strukturován do tříd. Instancemi tříd jsou objekty, které spolu navzájem komunikují.
Java je platformě nezávislá [34]. Zdrojový kód je zkompilován do platformě nezávislého bytekódu. Ten je interpretován pomocí Java Virtual Machine (JVM) do platformě závislého bytekódu až v době běhu aplikace. Díky tomuto mechanismu, můžeme program napsaný v Javě spustit na jakékoliv platformě.

Java klade důraz na jednoduchost [34]. Stejně jako například jazyk C++, má Java jednoduchou syntaxi.
Java je *high level language*. To znamená, že programátor se nemusí zabývat nízkoúrovňovými implementačními detaily, jakými jsou například alokace a uvolnění paměti, ukazatele, implementace vláken, procesů a mnoho dalšího. Díky těmto vlastnostem je Java snadno uchopitelná pro programátory, kteří programovat začínají nebo přecházejí z jiného jazyka.

Mezi nevýhody Javy patří takzvaný *boilerplate code*. Takto bývá označován kód, který nepřináší do výsledného programu žádnou hodnotu, je ale nutný, aby program fungoval. Jako příklad uvádím jednoduchou třídu, která obsahuje jednu proměnnou. Na obrázku 2.2 je vidět, že jsem musel jsem napsat 15 řádků kódu, abych vytvořil třídu, která obsahuje pouze jeden atribut.

```java
public class MyClass {
    private MyOtherClass property;

    public MyClass(MyOtherClass property) {
        this.property = property;   // 3x slovo property
    }

    public MyOtherClass getProperty() {
        return property;            //  2x slovo property
    }

    public void setProperty(MyOtherClass property) {
        this.property = property;   // 4x slovo property
    }
}
```

Obrázek 2.2: Příklad boilarplate kódu v Javě.

Další nevýhodou Javy verze 1.7 je omezená možnost funkcionálního programování a předávání částí kódu jako parametru. (Tato problematika byla adresována ve verzi Javy 1.8). Pokud potřebuji předat kód jako parametr, musím k tomuto účelu vytvořit anonymní třídu. Na příkladu 2.3 řadím list obsahující prvky typu MyClass z předchozího příkladu. Potřebuji je seřadit podle jejich atributu property. Abych dosáhl správného výsledku, musím vytvořit anonymní třídu implementující rozhraní Comparator a implementovat metodu compare. Znovu musím pro řešení jednoduchého problém napsat mnoho řádků kódu.

```java
  Collections.sort(myList, new Comparator<MyClass>() {

      @Override
      public int compare(MyClass first, MyClass second) {
          if (first.equals(second)) {
              return 0;
          }
          return first.getProperty().compareTo(second.getProperty());
      }
  });
```

Obrázek 2.3: Příklad řazení listu v Javě.

2.5.2 Kotlin

Kotlin je mladý jazyk. Jeho vývoj započal v roce 2010, první oficiální verze byla vydaná v roce 2016 a Android začal Kotlin podporovat v roce 2017 [35].

Kotlin je objektově orientovaný a zároveň funkcionální jazyk [36]. Kód může být členěn do tříd a může být zároveň i mimo třídy (takzvaně *top-level*).
V Kotlinu je všechno objekt, včetně funkcí, neexistují zde primitivní typy jako například int a boolean v Javě [37].

Kotlin může být zkompilován do stejného bytekódu, jaký používá Java a být interpretován pomocí JVM. Může být také přeložen do JavaScriptu a použit pro vývoj webových aplikací. JetBrains (firma, která Kotlin vyvíjí) pracuje také na možnosti Kotlin kompilovat přímo do platformě specifického bytekódu [36].
Na Androidu funguje kompilace stejně jako v případě Javy. Výsledný bytekód je tedy interpretován při běhu aplikace pomocí JVM.

Kotlin je s Javou navzájem „zaměnitelný“. Třídy napsané v Javě mohou být volané z Kotlinu a třídy napsané v Kotlinu mohou být volané z Javy [36][38]. To znamená, že v Kotlinu mohu použít všechny knihovny napsané pro Javu (Kotlin volá Javu) a mohu také použít všechny Java frameworky, například Spring v serverové části aplikace (Java volá Kotlin). Kotlin tak získává množství funkcí, které by jinak neměl.

Kotlin je moderní jazyk s pokročilou, kondenzovanou syntaxí. Tato zhuštěná syntaxe je velkou výhodou Kotlinu, výrazně snižuje množství boilerplate kódu. Na příkladu 2.4 je třída, která je ekvivalentní třídě z příkladu 2.2. Kotlinovská *data class* [39] má navíc automaticky vygenerované také metody *toString*, *equals* a *hashCode*.

```
1    data class MyClass (var property: MyOtherClass)
```

Obrázek 2.4: Příklad datové třídy v Kotlinu.

Protože Kotlin je i funkcionální jazyk, předávání funkcí jako parametru je velmi jednoduché. Funkce mohou být přiřazovány i do proměnných. Kotlin podporuje takzvané *higher order funkce*, které přijímají funkce jako parametr a vrací funkci jako návratovou hodnotu.
Na příkladu 2.3 je ukázka řazení listu v Javě. Stejná situace řešená v Kotlinu je pro porovnání na příkladu 2.5

```
1    myList.sortBy {it.property}
```

Obrázek 2.5: Příklad řazení listu v Kotlinu.

Další výhodou Kotlinu je jeho způsob práce s tzv. *nullables*, to jsou objekty, které nemusí mít žádnou hodnotu. V Javě jsou všechny objekty implicitně nullable, v Kotlinu je to naopak. Pokud potřebuji v Kotlinu objekt, který může nabývat *null* (žádné) hodnoty, za jeho typ přidám otazník, tím explicitně deklaruji, že tento objekt může nabývat null hodnoty. Pokud při přístupu k němu nekontroluji stav objektu, kód se nezkompiluje. Kotlin tak zabraňuje chybě NullPointerException, jedné z nejčastějších chyb v Java programech.

```
1    // this je v~extension funkci objekt na kterém se funkce volá.
2    // zde to bude instance třídy String
3    fun String.isLongerThan(length: Int) = this.length > length
4
5    ...
6
7    val example = "text"
8    val result  = example.isLongerThan(8) // false
```

Obrázek 2.6: Příklad extension funkce v Kotlinu.

Kontlin nabízí také extension (rozšiřující) funkce. Pomocí nich můžu rozšířit funkcionalitu třídy bez nutnosti z ní dědit. Můžu například přidat metodu třídě, kterou vkládám do programu z nějaké knihovny a nemám k ní přímý přístup. Nebo je požadovaná funkcionalita specifická pro konkrétní kontext a nechci přidávat tuto metodu přímo do rozšiřované třídy. Na příkladu 2.6 je extension funkce, která do třídy String přidává metodu *isLongerThan*.

Na tomto příkladu 2.6 je také patrná další vlastnost Kotlinu a to je type inference (odvození typu). Konstanta *example* je typu String, v kódu to není explicitně napsáno, ale kompilátor je schopen typ odvodit podle hodnoty, kterou do proměnné přiřazuji.
Stejně tak *result* je typu Boolean, protože funkce *isLongerThan* vrací Boolean. Tato funkce ale nemá návratovou hodnotu specifikovanou, i tady kompilátor typ odvodí.

Dále je na příkladu 2.6 vidět, že proměnné jsou v kódu zarovnané. Před jménem proměnné je vždy pouze jedno klíčové slovo.

- **val** pro konstantu (neměnnou proměnnou – *final* v Javě)
- **var** pro proměnnou
- **fun** pro funkci

Všechna tři tato klíčová slova mají tři písmena a jména proměnných jsou tak přímo pod sebou a snadno čitelná.

2.5.3 Porovnání

Oba jazyky jsou si rovnocenné co se výkonnosti týče [40]. Oba dva jsou spouštěny v Java Virtual Machine. Při sestavování aplikace je rychlejší Java, kompiluje se totiž přímo do bytekódu, Kotlin se nejdříve překládá do Javy.

Funkcionalit nabízí Kotlin více. Z části proto, že může pracovat se všemi knihovnami napsanými pro Javu a z části kvůli svým specifickým vlastnostem jako jsou higher order funkce, extension funkce a odvození typu.

Kotlin je bezpečnější. Je to díky jeho práci s nullables. K bezpečnosti také přispívá skutečnost, že v Kotlinu je dobrým zvykem používat konstanty místo proměnných. Parametry metod jsou neměnné, kolekce jsou také neměnné pokud si programátor explicitně nevyžádá měnitelné, programátor je veden k tomu, aby také používal konstanty (val), proměnné (var) by měl použít pouze pokud je to nezbytné [40].

Java oproti Kotlinu nabízí například checked exceptions (výjimky, které je nutné zachytit), ternární operátor a statické proměnné a třídy [37].

2.5.4 Závěr

Rozhodl jsem se použít jazyk Kotlin. Zaujalo mě totiž velké množství funkčností, které Kotlin oproti Javě nabízí, zejména oproti Javě 7. Java 8 přinesla mnoho užitečných změn, například práci se *streamy*. V Kotlinu tato funkcionalita existuje také, nehledě na to, že se v mém projektu kompiluje do Javy 7. Na oficiálních stránkách Androidu je Kotlin také doporučován [41].

2.6 Návrh řešení

ECO Tracker bude mobilní Android aplikace napsaná v jazyce Kotlin. Bude rozdělená do pěti sekcí – události, nápady, formulář, statistiky a profil.

V sekci události si bude moci uživatel prohlížet události vytvořené ostatními uživateli, bude se k nim moci připojit, komentovat je a sdílet. Bude zde také moci vytvořit vlastní události. Když se uživatel připojí k události, ECO Tracker ji automaticky přidá do jeho výchozího android kalendáře. Aplikace uživateli umožní vybrat místo konání pomocí aktivity s mapou. Pokud uživatel tuto možnost využije, k události se uloží i GPS souřadnice její lokace. ECO Tracker potom bude nabízet možnost navigace k místu konání této události.

Sekce nápady bude zobrazovat seznam ekologických nápadů, tipů a zajímavostí. Uživatel i zde bude moci vytvářet své vlastní nápady a přizpůsobovat si jejich vzhled. Nápady bude moci hodnotit (like nebo dislike) a sdílet. Pokud bude mít uživatel, se kterým je nápad (nebo i událost) sdílen, nainstalovaný ECO Tracker, otevře se nápad v něm, jinak se otevře obchod Google Play na stránce s aplikací ECO Tracker.

V sekci formuář bude moci uživatel snadno a jednoduše vyplnit údaje o své spotřebě různých druhů odpadu (dále souhrnně označovány jako zdroje). Pokud uživatel nebude chtít měřit jeho skutečné množství, bude mít možnost k datům přičíst svoji průměrnou spotřebu. Údaje o spotřebě každého zdroje bude moci uživatel odeslat jednou denně. Po odeslání dat zmizí vyplněné zdroje z nabídky formuláře.
ECO Tracker bude jednou denně zobrazovat uživateli notifikaci, připomínající mu vyplnění formuláře. Přímo z notifikace bude moci uživatel odeslat průměrnou hodnotu spotřeby nejčastěji používaných zdrojů.

V sekci statistiky budou uživateli prezentovány jeho výsledky. Statistiky budou zobrazovány pro každý zdroj zvlášť. Budou obsahovat graf, ve kterém bude znázorněn vývoj uživatelovy spotřeby daného zdroje v čase.
Bude zde také textový popis dat, porovnání uživatele s ostatními uživateli aplikace a s průměrným obyvatelem Země. Číselné hodnoty budou uvedeny v běžných jednotkách (kg, l, ...), ale také v jednotkách více přibližujících jejich velikost (bazény pro objem, sloni pro hmotnost apod.).

V sekci profil si bude moci uživatel přizpůsobit jednotky, ve kterých jsou zobrazovány statistiky a formulář. Bude si zde moci změnit email, pod kterým je do aplikace zaregistrován a přidat nebo odebrat jazyky, kterými mluví. Podle těchto jazyků se mu budou zobrazovat příspěvky ostatních uživatelů v sekci nápady a události.
Bude si zde také moci prohlédnout nápady, které vytvořil a události, které buď vytvořil, nebo se k nim připojil.

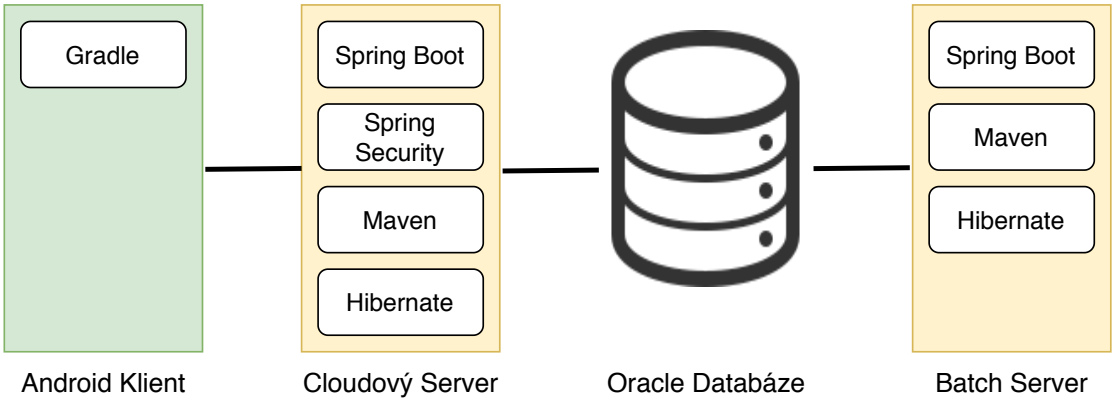
Se serverovou částí aplikace bude android část komunikovat přes Rest API pomocí protokolu Https. Toto rozhraní bude zabezpečeno pomocí technologie Spring Security. Serverová část bude Spring Boot aplikace napsaná také v jazyce Kotlin.

2.7 Architektura

Protože nápady a události musí být dostupné všem uživatelům ECO Trackeru, bude muset mobilní aplikace komunikovat se serverem, který bude zajišťovat správu a synchronizaci dat.
Android aplikace bude pouze klient, který bude odesílat svoje požadavky serverové aplikaci přes Rest rozhraní. Serverová část bude spravovat autorizaci a autentizaci uživatele, ukládání nápadů, událostí, statistických dat a vracení jejich zpracované verze.
Data samotná budou uložena v databázi, do které bude mít server přístup.

ECO Tracker bude mít také batch server. Na něm se na konci každého dne spustí skript zpracovávající statistická data. Pro uplynulý den vypočte například data zobrazovaná v grafech, aby se nemusela počítat opakovaně při tvorbě každého grafu.

Technologie použité v aplikaci jsou znázorněny na obrázku 2.7.



Obrázek 2.7: Diagram architektury

2.8 Vymezení rozsahu

Protože implementace všech funkcionalit popsaných v sekci Návrh řešení je příliš obsáhlá, bude tato bakalářská práce pokrývat následující funkcionality.

- Sběr dat o vyprodukovaném odpadu (i pomocí notifikací).
- Zobrazování statistik.
- Tvorbu, editaci a sdílení nápadů.
- Tvorbu, editaci, sdílení a správu událostí.
- Základní přizpůsobení obsahu uživateli.
- Autorizaci a autentizaci uživatele.

Funkcionality, které do práce naopak zahrnuty nejsou, mám však v plánu je implementovat v pozdějších fázích projektu.

- Hodnotící systém.
- Notifikace na tip dne, výzvu týdne a podobně.
- Přidávání obrázků do nápadů a událostí.
- Některá nastavení přizpůsobení obsahu a formy (například rozložení stránky pro tablety/chromebooky nebo volba zobrazovaných zdrojů).
- Komentáře k událostem.
- Filtrování událostí podle místa konání.
- Obnovení účtu (zapomenuté heslo a pod.).
- Batch server.

Serverovou část aplikace naimplementuji také a bude dostupná v příloze této práce. Zde v textu se však budu zabývat pouze implementací klienta pro Android.

Kapitola 3

Realizace

3.1 Sekce nápady

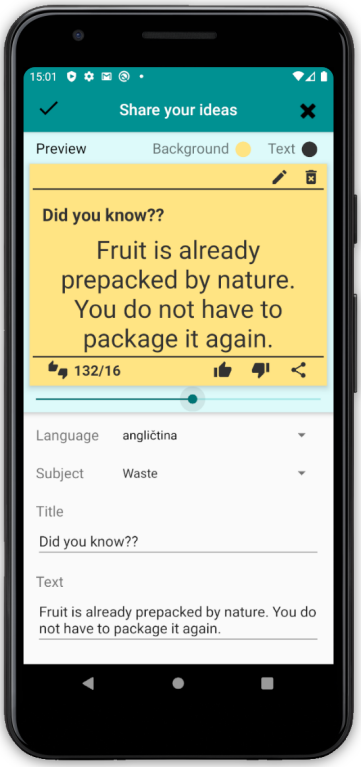
Předpokládám, že sekce s nápady je ta, na které uživatel bude trávit nejvíce času, protože bude obsahovat většinu nového obsahu. Slouží ke sdílení nápadů a myšlenek o ekologii mezi uživateli.

3.1.1 Přehled

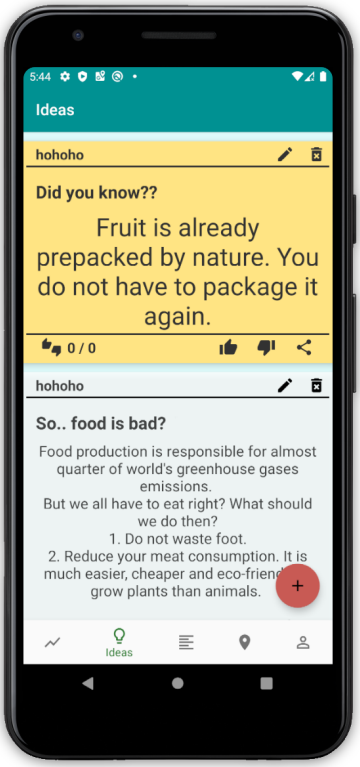
Tato sekce má velmi jednoduchý design. Nahoře na obrazovce je umístěn nadpis sekce, zbytek obrazovky zabírá posuvný seznam nápadů. V pravém dolním rohu je tlačítko startující proces tvorby nového nápadu (snímek obrazovky A.4).

Nápad má svůj název a popis, což jsou jeho hlavní atributy.
Dále má atributy definující jeho vzhled. Těmito jsou velikost a barva písma a barva pozadí.
Nápad má také metadata potřebná pro správné fungování aplikace. Sem patří jazyk, ve kterém je nápad napsaný, téma, kterého se týká a jméno uživatele, který nápad vytvořil. Také sem patří odkaz sloužící k jeho sdílení a informace o tom, kolik uživatelů dalo nápadu palec nahoru a kolik dolů a zda aktuálně přihlášený uživatel dal palec nahoru nebo dolů.

Uživatel si může prohlédnout všechny nápady, kterým rozumí. V aplikaci neexistuje žádné jiné omezení obsahu. Zda uživatel nápadu rozumí nebo ne, rozhodne ECO Tracker podle toho, zda jazyky, které uživatel ovládá (definované v sekci profil), obsahují jazyk, ve kterém je nápad napsaný.



Obrázek 3.1: Tvorba nápadu



Obrázek 3.2: Sekce nápady

Původně jsem chtěl mít 3 druhy nápadů.

- **Věděli jste že..?** sloužící ke sdílení zajímavostí.
- **Tipy** určené na praktické nápady, jak měnit svůj ekologický život k lepšímu.
- **Alternativy** měly nabídnout lepší možnosti ke každodenně používaným předmětům, například plátěná taška namísto plastové apod. (inspirováno aplikací My Little Plastic Footprint [23]).

Toto řešení jsem měl naimplementované, každý druh nápadu měl svojí designovou šablonu a protože nápady z kategorie „alternativy“ měly jiné atributy než ostatní druhy nápadů, měl jsem i pro každou kategorii zvláštní třídu.
Po delším používání aplikace jsem ale zjistil, že toto rozdělení nápadů věci komplikuje, ale nepřidává velkou hodnotu. Při svém testování jsem po čase začal používat pouze výchozí kategorii a došel jsem k závěru, že stejně by se chovali i ostatní uživatelé.
Myslím, že by toto rozlišování kategorií zhoršovalo user experience (uživatelský zážitek), a proto jsem od této myšlenky upustil. Všechny kategorie jsem sjednotil a v aktuálním řešení žádné kategorie nápadů neexistují a všechny nápady mají jednotnou designovou šablonu.

3.1.2 Vytvoření nápadu

Pro vytvoření nápadu uživatel klikne na tlačítko *přidat*, otevře se aktivita CreateIdeaActivity, ve které uživatel vyplní všechna potřebná data. Tato aktivita uživateli již při tvorbě nápadu zobrazuje preview (náhled) toho, jak bude výsledný nápad vypadat (snímek obrazovky A.5).

V okamžiku, kdy je uživatel se svým dílem spokojený, klikne na tlačítko *uložit*. Aktivita zkontroluje, zda má nápad vyplněné všechny povinné atributy a výsledek vrátí třídě, ze které byla zavolána.
Pokud byl vrácený nápad validní, volající třída ho uloží. V opačném případě dostane uživatel možnost svůj nápad opravit a celý proces se opakuje (viz obrázek 3.3).

Uživatel může kdykoli upravit jím vytvořené nápady. Opět se otevře CreateIdeaActivity a proces úpravy nápadu je stejný, jako při jeho vytváření.

3.1.3 Sdílení nápadu

Při sdílení nápadu ECO Tracker nabídne uživateli seznam aplikací schopných nápad sdílet a uživatel si jednu z nich vybere.
Nápad je sdílený jako webová adresa, ta je ve tvaru, kterému ECO Tracker rozumí. Po kliknutí na takovou adresu se ECO Tracker spustí a zobrazí uživateli sdílený nápad.
Pokud uživatel aplikaci nainstalovanou nemá, otevře se místo ECO Trackeru webový prohlížeč a přistoupí na danou webovou adresu. Tu bude obsluhovat serverová část aplikace, obdrží požadavek a přesměruje uživatele na adresu aplikace ECO Tracker v obchodě Google Play, kde si ji bude moci stáhnout (přesměrování není na serveru zatím implementováno). Po jejím stažení si uživatel může sdílený nápad prohlédnout v aplikaci (snímek obrazovky A.14).

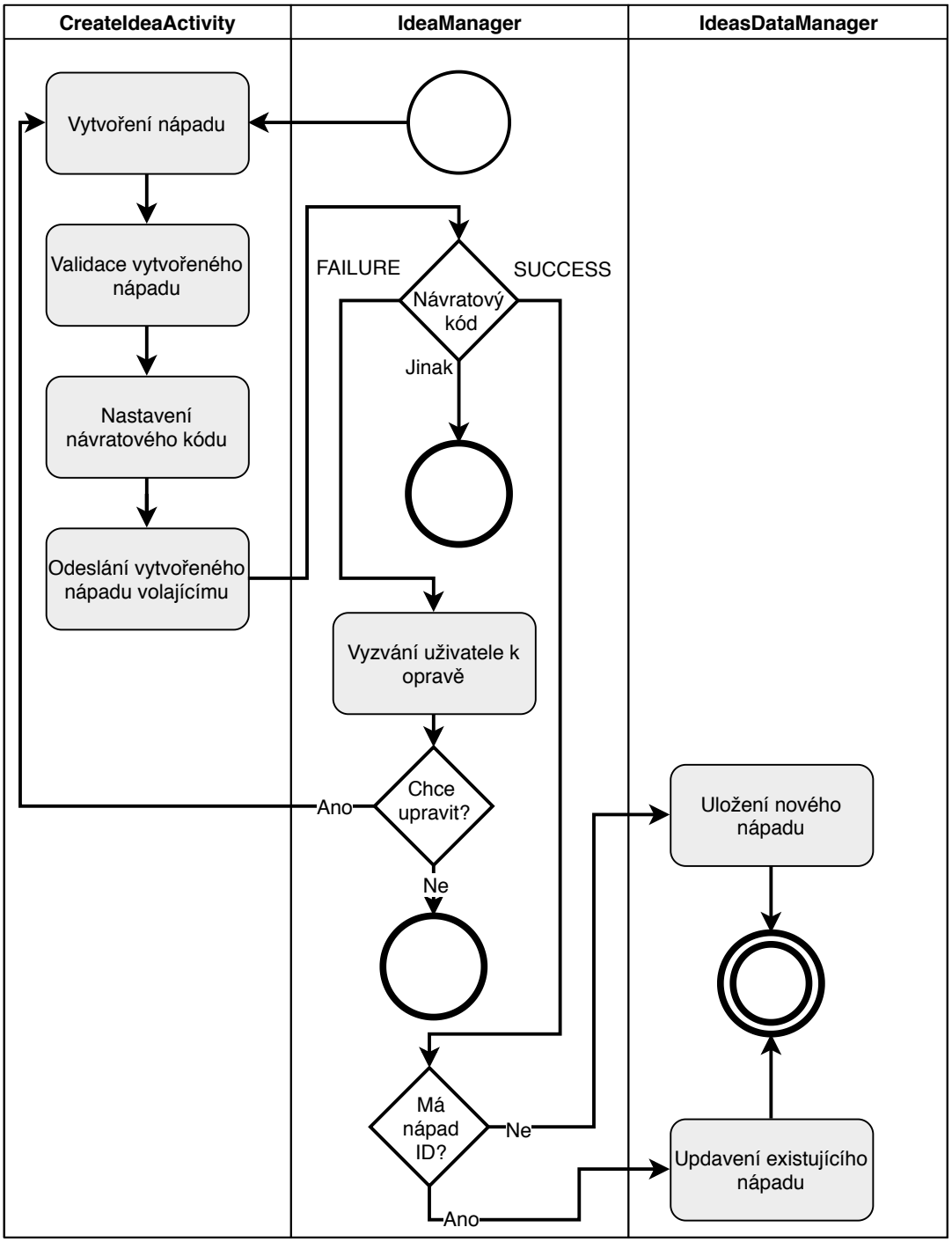
3.2 Sekce události

Sekce s událostmi slouží k tvorbě a sdílení ekologických akcí. Uživatel je tímto vybízen vyjít ven z domu a měnit svět kolem sebe k lepšímu.
ECO Tracker po uživateli vyžaduje změny v chování. Psaní o životním prostředí a ekologii nestačí.

3.2.1 Přehled

Sekce s událostmi je velmi podobná sekci s nápady, také obsahuje seznam událostí a tlačítko pro přidání nové události (snímek obrazovky A.18).

Událost má svůj název, popis, plánovaný datum a čas zahájení a ukončení, místo konání, jazyk, ve kterém je napsaná a volitelně může obsahovat i GPS souřadnice místa konání.



Obrázek 3.3: Diagram procesu vytváření a editace nápadů

Zobrazovány jsou pouze události napsané v jazyce, kterému uživatel rozumí a pouze události, které ještě neproběhly. Do budoucna chci přidat i filtrování na základě vzdálenosti místa konání události a uživatelova bydliště.

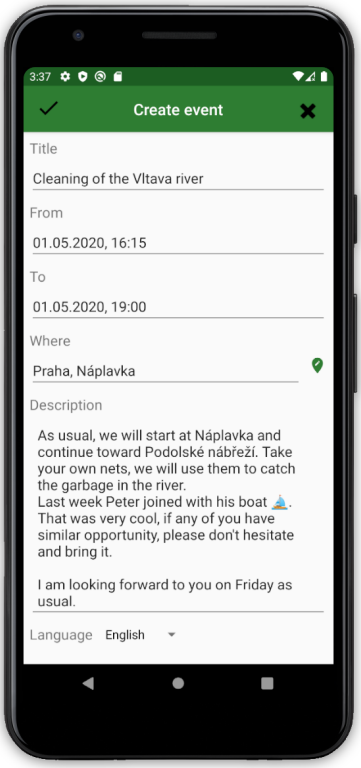
3.2.2 Vytváření událostí

Na obrázku 3.3 je popsán proces vytváření nápadu. Přesně stejným procesem projde při vytváření i událost. Pro vytváření události má ECO Tracker samostatnou aktivitu *CreateEventActivity* (snímek obrazovky A.7). V té uživatel vyplní atributy události. Aktivita následně zkontroluje, zda je událost validní a podle výsledku validace nastaví návratový kód odpovědi a vrátí událost volající aktivitě.
Pokud se vytvořená událost vrátila s návratovým kódem *SUCCESS*, ECO Tracker ho uloží.
Jestliže událost validní nebyla, návratový kód bude *FAILURE* a aplikace dá uživateli možnost událost upravit. Pokud si uživatel přeje událost upravit, celý proces se zopakuje.

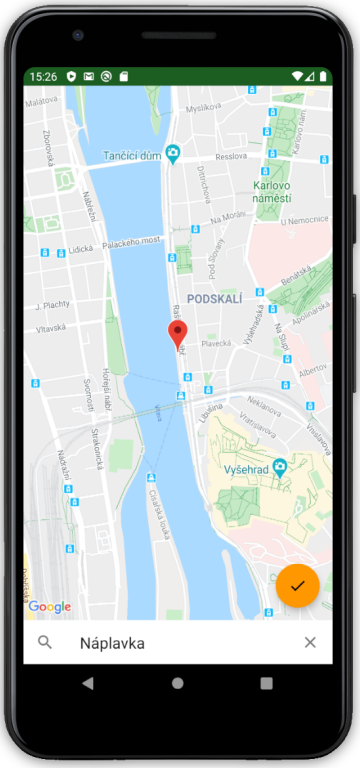
Některé atributy událostí vyžadují sofistikovanější vstupní metody než prosté textové pole.
Pro datum a čas používám kombinaci dvou fragmentů – kalendáře a hodin (snímek obrazovky A.10). Oba tyto fragmenty jsou systémové knihovny [42], nemusel jsem proto stahovat žádnou knihovnu třetích stran.
Použití kalendáře a hodin má oproti textovému poli výhodu v tom, že je jasně daný formát v jakém ECO Tracker datum a čas dostane. Do textového pole může uživatel napsat cokoli. My češi jsme zvyklí na datum ve formátu formátu „dd. MM. yyyy“ – například 15. 12. 2019. Jiné národnosti jsou ale zvyklé na jiné formáty a bylo by proto velmi obtížné až nemožné, získávat datum z textového pole.

Dalším komplexním atributem jsou GPS souřadnice události. Tento atribut je volitelný, pokud ho uživatel nechce specifikovat, nemusí. Jeho přítomnost ale usnadní uživatelům, kteří se k události připojili, její nalezení.
Pokud chce tvůrce události tento atribut uvést, klikne nejprve na ikonu místa ( ) vedle textového vstupu, který specifikuje místo konání události (snímek obrazovky A.7). Tím se spustí aktivita obsahující Mapy Google a vyhledávací pole (snímek obrazovky A.9).
Pokud už měl uživatel vyplněné místo konání události, předvyplní ECO Tracker vyhledávací pole stejným textem a pokusí se ho na mapě najít.
Uživatel může přesnou polohu místa konání události zvolit pomocí vyhledávacího pole (snímek obrazovky A.8) a pomocí gest na mapě samotné (například táhnutí pro posun mapy nebo kliknutí pro výběr místa).

Jakmile je uživatel spokojený se svým výběrem, potvrdí lokaci pomocí tlačítka *OK*. Aktivita s mapou se zavře a volající aktivitě předá vybrané GPS souřadnice a text, který uživatel předtím zadal do vyhledávacího pole.
Pokud uživatel před výběrem souřadnic nezadal textový popis místa konání události, ECO Tracker tento popis vyplní textem, který uživatel zadal do vyhledávacího pole v aktivitě s mapou. V opačném případě uživatelův text nepřepisuje.
Aby si mohl být uživatel jist, že souřadnice jsou vybrané úspěšně, přebarví se ikona místa zeleně.



Obrázek 3.4: Vytváření nápadu



Obrázek 3.5: Výběr místa konání události

Pokud událost disponuje GPS souřadnicemi, zobrazuje se u ní navigační ikonka. Po kliknutí na ni se uživateli otevře nabídka aplikací, ve kterých si může místo konání prohlédnout (mapy nebo navigace). Zvolená aplikace místo nalezne a zobrazí uživateli (snímek obrazovky A.13).

3.2.3 Detailní pohled na událost

Uživatel může na konkrétní událost ze seznamu kliknout. Tím se otevře detailní pohled na danou událost (snímek obrazovky A.11). Tento detailní pohled obsahuje, stejně jako náhled události, všechny její atributy. Navíc obsahuje sekci s komentáři a tlačítka na úpravu a smazání události, ta jsou viditelná a fungující pouze pro tvůrce události.

Událost bude moci okomentovat každý uživatel bez jakéhokoli omezení. Důraz bude kladen pouze na to, aby komentáře byly ve stejném jazyce, v jakém je napsaná událost. ECO Tracker ale nebude obsahovat nic, čím by toto mohl zajistit nebo zkontrolovat, pouze uživateli při psaní komentáře zobrazí žádost o dodržování tohoto pravidla.

Komentáře zatím nejsou implementované, tato funkcionalita nebyla zahrnuta do cílů práce.

3.2.4 Sdílení událostí

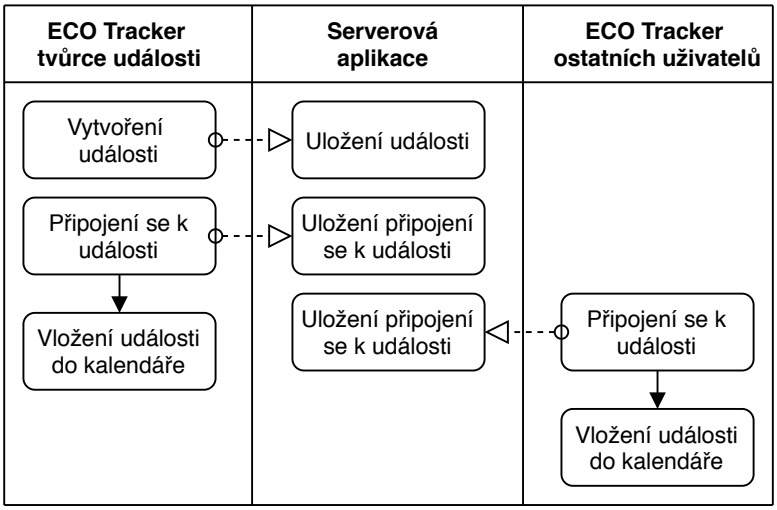
Sdílení událostí funguje stejně jako sdílení nápadů. Aplikace ECO Tracker nabídne uživateli seznam aplikací schopných událost sdílet a uživatel si jednu z nich vybere (snímek obrazovky A.12).
Událost je sdílená jako webová adresa obsahující ID události. Po kliknutí na takovou adresu v android zařízení mohou nastat 2 situace.

1. Uživatel má nainstalovanou aplikaci ECO Tracker.
   Potom se událost otevře v aplikaci (snímek obrazovky A.14) – v detailním pohledu na událost (snímek obrazovky A.11).
2. Uživatel aplikaci ECO Tracker nemá.
   V takovém případě se otevře webový prohlížeč a přistoupí na danou adresu. Tu bude obsluhovat serverová část aplikace, která uživatele přesměruje na stránku ECO Trackeru v obchodě Google Play a nabídne tím uživateli stažení aplikace.
   Po jejím stažení si už může uživatel událost prohlédnout v aplikaci.

3.2.5 Připojení se k události

K události se uživatel připojí kliknutím na tlačítko *Připojit se*. ECO Tracker při tom zároveň vytvoří i událost v jeho výchozím android kalendáři (snímek obrazovky A.15). Aplikace si při tom musí do shared preferencies uložit ID události v ECO Trackeru (jako klíč) a ID vytvořené události v kalendáři (jako hodnotu). Tato informace je nutná pro případné úpravy události nebo její smazání.
Po úspěšném vytvoření události je k ní tvůrce automaticky připojen.

Odpojení od události probíhá stejně jednoduše. Uživatel klikne na tlačítko *Připojen*, které se u události zobrazuje místo tlačítka *Připojit se*, pokud je uživatel připojen.



Obrázek 3.6: Vytvoření a připojení se k události

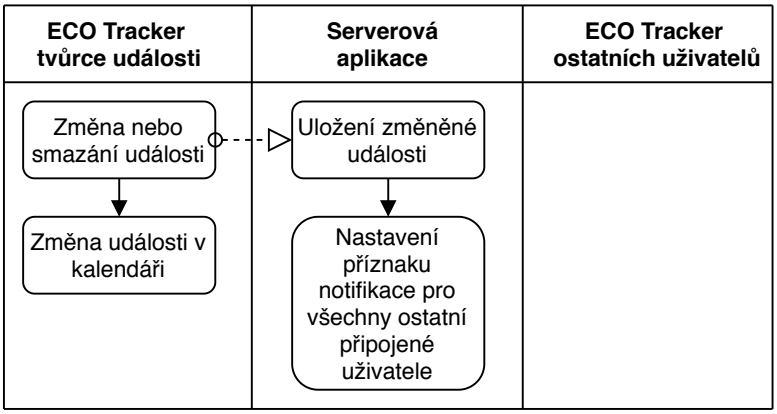
Tím se uživatel od události odpojí, tlačítko změní svůj text zpět na *Připojit se* a ECO Tracker smaže uživateli událost z jeho kalendáře, kterou vytvořil při připojování se k této události.

Komplikace nastává v okamžiku, kdy tvůrce události změní nebo smaže událost, ke které jsou připojeni i další uživatelé. Pro tvůrce je situace jednoduchá, při úpravě události ji ECO Tracker rovnou upraví i v kalendáři a při mazání události ji z kalendáře smaže. Ostatní uživatelé se ale o této změně také potřebují dozvědět a jejich kalendáře musí být rovněž aktualizovány.

Situaci jsem vyřešil tak, že do tabulky, která mapuje připojené uživatele k událostem, jsem přidal boolean (nabývající pouze hodnot pravda / nepravda) atribut, podle kterého rozhodnu, zda je potřeba uživatele o změně informovat nebo nikoli.

Na začátku je tabulka prázdná. Tvůrce vytvoří událost a dojde k jeho automatickému připojení k události. Tím se do tabulky dostane informace o tom, že tvůrce je připojen k vytvořené události a atribut notifikace se nastaví na nepravda.
Následně se připojí druhý uživatel (uživatel B) a do tabulky se, stejně jako v případě tvůrce, přidá informace, že uživatel B je připojený k události a není třeba ho o ničem informovat (obrázek 3.6).

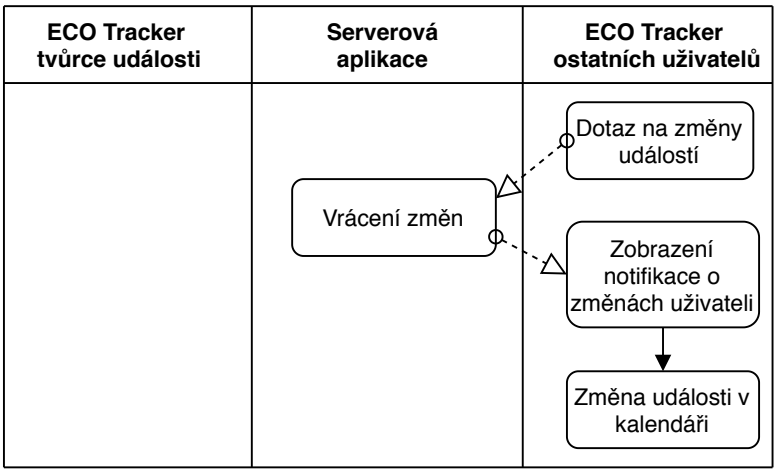
V okamžiku, kdy tvůrce smaže, nebo změní událost, změní se tato událost v databázi a příznak notifikace se nastaví na pravda pro všechny ostatní připojené uživatele. Tímto se do tabulky dostala informace o tom, že je nutné uživatele B informovat o změně události (obrázek 3.7).



Obrázek 3.7: Modifikace události

ECO Tracker zkontroluje, zda je třeba přihlášeného uživatele informovat o změnách, vždy při startu aplikace. Tím se aplikace uživatele B dozví o tom, že je třeba jej informovat. Zobrazí uživateli vyskakovací okno se seznamem změněných událostí (snímek obrazovky A.17) a tyto události upraví i v jeho android kalendáři (obrázek 3.8).

Do budoucna bych chtěl přidat proces, který poběží na pozadí a pravidelně bude kontrolovat, zda se nějaká uživatelova událost nezměnila. V případě že ano, by uživateli zobrazil upozornění.



Obrázek 3.8: Reakce připojených uživatelů na změnu události

3.3 Sekce profil

V této sekci uživatel najde své osobní informace a má možnost je upravit (snímek obrazovky A.19). Je zde jeho emailová adresa, kterou si aplikace uchovává pro případ, že uživatel zapomene své přihlašovací údaje. ECO Tracker tak bude moci uživateli v budoucnu poslat nové údaje na email (zatím neimplementováno).
Dále zde uživatel nalezne seznam jazyků, které ovládá. Při prvním spuštění aplikace se do tohoto seznamu automaticky přidá systémový jazyk (jazyk, ve kterém se zobrazuje android prostředí). Pro přidání dalších jazyků uživatel klikne na tlačítko *přidat jazyk*. ECO Tracker zobrazí vyskakovací okno se všemi existujícími jazyky. Ty jsou opět vypsané v systémovém jazyce (snímek obrazovky A.20). To znamená, že pokud bude mít telefon v českém jazyce, v nabídce jazyků uvidí *angličtina*, *čeština*... Pokud bude systémovým jazykem angličtina, zobrazí se stejné jazyky jako *English*, *Czech*...
Pro seznam vybraných jazyků platí omezení, že musí obsahovat alespoň jeden jazyk.

Pokud se sekce Profil více rozšíří, budou tato přizpůsobení extrahována do samostatné sekce Nastavení.

Možná nejdůležitější část této sekce je část pro správu nápadů a událostí. Smyslem tohoto oddílu aplikace je umožnit uživateli snadný přístup k příspěvkům, které vytvořil. Může je tak upravit nebo smazat, bez toho, aby je musel hledat mezi všemi existujícími nápady a událostmi
Tato část je rozdělená na nápady (snímek obrazovky A.21) a události (snímek obrazovky A.22). U událostí uživatel navíc vidí i události, ke kterým se připojil.

3.4 Sekce Formulář

Tato sekce obsahuje seznam číselných vstupů, seřazených pod sebou. Každý zdroj má jeden číselný vstup, kterým může uživatel specifikovat konkrétní množství vyprodukovaného odpadu (např. pět kilogramů plastového odpadu apod.) U každého zdroje je také tlačítko na použití uživatelovy průměrné hodnoty. Jejím výpočtem se budu zabývat níže.

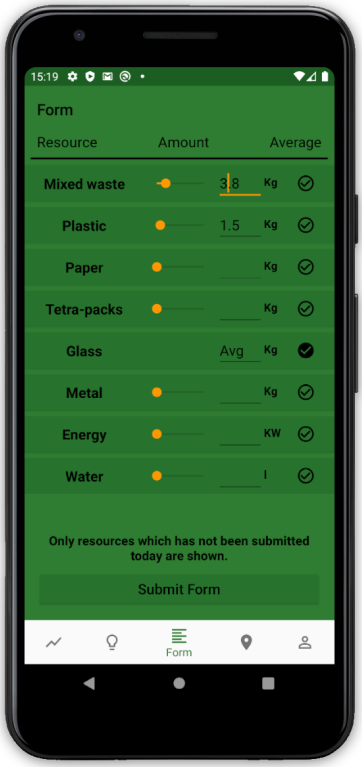
Měl jsem potíže s navrhováním designu této sekce. Myšlenka byla jednoduchá – mít vertikální seznam číselných vstupů, vedle vstupů tlačítka pro průměrnou hodnotu a u každého vstupu nadpis zdroje, ke kterému se váže.
Toto nefungovalo, kohokoli jsem se zeptal na vzhled této obrazovky, řekl mi, že nevypadá dobře. Nikdo mi ale nebyl schopen poradit, jak sekci udělat hezčí.

Dospěl jsem k několika variantám.

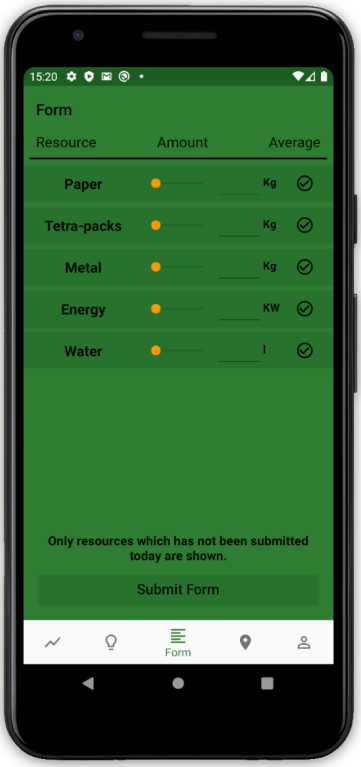
1. Mohl jsem číselný vstup a tlačítko pro průměr přidat pod textový popis v sekci statistiky ke každému zdroji zvlášť a sekci formulář zrušit.

2. Napadlo mě zobrazit formulář pouze jako vyskakovací okno po kliknutí na notifikaci. To by mělo číselný vstup, tlačítko pro průměrnou hodnotu, tlačítko na odeslání hodnoty a horizontálně posuvný seznam zdrojů. Uživatel by vždy našel zdroj, který chce vykázat, zadal by hodnotu a odeslal.

3. Mohl jsem také místo číselného vstupu (zadaný jako text) použít slider (posuvník), kterým by uživatel reguloval odesílanou hodnotu.



Obrázek 3.9: Formulář



Obrázek 3.10: Formulář s některými hodnotami již odeslanými

Nakonec jsem si zvolil třetí možnost, ale místo toho, abych číselný vstup posuvníkem úplně nahradil, pouze jsem vstup zmenšil a posuvník přidal před něj. Uživatel si tak může vybrat, jakým způsobem hodnoty zadá (snímek obrazovky A.23).

Toto řešení má několik výhod. Zdroje zůstanou na jednom místě a uživatel mezi nimi nebude muset přepínat. Také nebude nutné pro zadání hodnot otvírat klávesnici. Ta na obrazovce zabírá mnoho užitečného místa, a když jsou vstupy seřazené pod sebou, je to obzvlášť výrazné.

Každý druh odpadu může uživatel vykázat pouze jednou denně. Po odeslání formuláře se vyplněné položky skryjí a budou přístupné až během následujícího kalendářního dne.

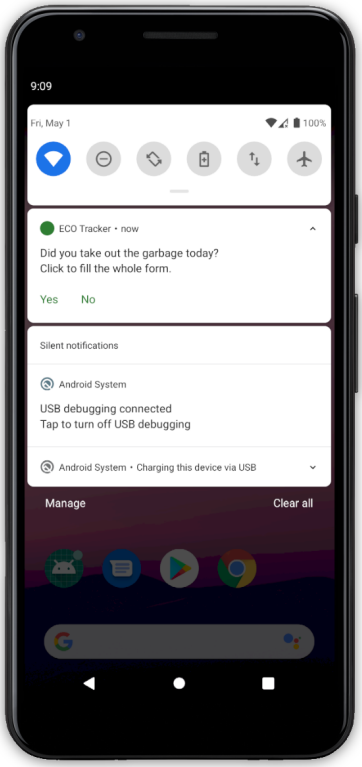
3.4.1 Výpočet průměrné hodnoty a statistik

Pokud se uživateli nechce vážit odpadky pro vyplnění formuláře, má možnost odeslat průměrnou hodnotu.
Co se však stane v okamžiku, kdy uživatel ještě žádnou hodnotu neodeslal? V takovém případě nejsou k dispozici data, ze kterých by se mohl uživatelův průměr vypočítat.
Proto bude průměr uživateli počítán až od okamžiku, kdy odešle pět a více svých vlastních hodnot. Do té doby se místo něj bude používat průměrná hodnota všech uživatelů. Toto řešení není ještě na serveru naprogramováno, jeho implementace ale nebude vyžadovat žádné změny v android aplikaci.

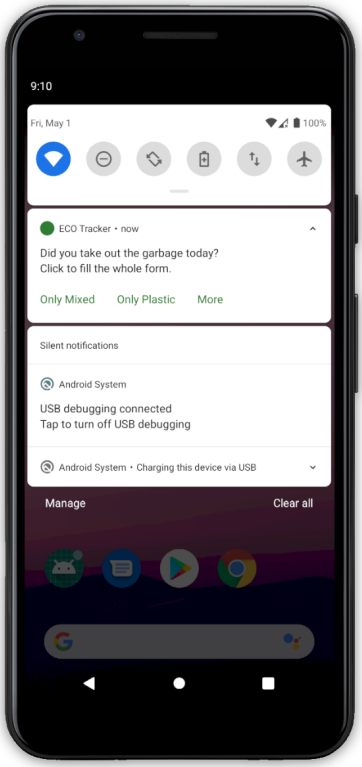
Data jsou ukládána do tabulky obsahující sloupce hodnota, datum, zdroj a uživatelské jméno. V grafech v sekci statistiky zobrazuji vývoj denní spotřeby zdroje. K získání těchto dat musím každou hodnotu v tabulce vydělit počtem dní, které uběhly od posledního vyplnění dat. Tento výpočet se provede při dotazu na hodnoty zobrazené v grafu. Průměr všech uživatelů se bude počítat na konci dne a výsledek se uloží do samostatné tabulky, aby se výpočet nemusel stále opakovat. Tuto proceduru zatím spouštím ručně, později ji však bude spouštět batch server.

3.4.2 Notifikace

Na zadání dat ve formuláři může uživatel snadno zapomenout. Aby měl ECO Tracker spolehlivější data, snažím se tento faktor omezit. Každý den v devatenáct hodin, nehledě na časovou zónu, se uživateli zobrazí notifikace připomínající mu zadávání dat.
Tato notifikace je interaktivní, obsahuje otázku „Vynášel jste dnes odpad?“ a odpovědi „Ano“ a „Ne“.
Kliknutí na notifikaci samotnou otevře formulář. Kliknutí na „Ne“ notifikaci zavře.
Kliknutí na „Ano“ notifikaci upraví. Nyní notifikace obsahuje text „Jaký odpad jste vynášel?“ Notifikace je opět interaktivní, tentokrát obsahuje tři odpovědi – „Směsný odpad“, „Plasty“, „Více“.
Kliknutí na „Více“, stejně jako kliknutí na notifikaci, otevře uživateli formulář.
Směsný odpad, respektive plasty připíšou uživateli jeho průměrnou hodnotu v dané surovině.



Obrázek 3.11: Notifikace formuláře 1. krok



Obrázek 3.12: Notifikace formuláře 2. krok

3.5 Sekce Statistiky

Sekce se statistikami je klíčová. Účelem této sekce je poskytnout uživateli zpětnou vazbu a motivovat ho k zlepšování se.

3.5.1 Design

Po designové stránce je sekce rozdělená do několika částí. Na horní hraně obrazovky je posuvná nabídka se všemi zdroji. Po kliknutí na konkrétní zdroj se zobrazí statistiky uživatelova hospodaření s tímto zdrojem (snímek obrazovky A.27).
Pod nabídkou zdrojů se zobrazuje graf. V něm je vykreslena uživatelova spotřeba daného zdroje a průměrná spotřeba tohoto zdroje všech uživatelů aplikace. Následuje textový popis dat.

3.5.2 Graf

Moje požadavky na knihovnu generující grafy byly následující.

- Open-source licence.
- Pěkný design.
- Možnost vykreslení více křivek do jednoho grafu.
- Možnost snadno měnit data v grafu.
- Možnost přizpůsobení os grafu.
- Možnost upravit vzhled každé křivky v grafu zvlášť.

Vybral jsem knihovnu HelloCharts [43], která nabízí všechny funkce, které jsem od knihovny požadoval, její design ladí s designem zbytku aplikace a je poměrně jednoduché s ní pracovat. Jedinou velkou nevýhodou této knihovny je absence kvalitní dokumentace.

V grafu zobrazuji údaje přihlášeného uživatele a průměr z dat všech uživatelů. Původně jsem chtěl do grafu zanést i průměrnou spotřebu spoluobčanů uživatele a průměrnou spotřebu na světě. Tyto informace jsou ale jenom obtížně dohledatelné. Data, která by se specializovala na konkrétní zdroj v podstatě neexistují. Byl jsem tedy nucen od této myšlenky upustit a zůstat pouze u dat, nad kterými mám kontrolu.
U uživatelovy křivky zobrazuji ještě odhadovaný vývoj jeho spotřeby do budoucna (snímek obrazovky A.27).

3.5.3 Textový popis

Textový popis je rozdělený do několika odstavců.
V prvním odstavci je informace o tom, kolik celkem kilogramů (budu zde v textu používat metrický systém) daného zdroje uživatel spotřeboval od doby, kdy v aplikaci začal tento zdroj vykazovat.

Následující odstavec informuje uživatele o tom, kolik je to kilogramů denně a kolik tun tohoto zdroje by se vyprodukovalo každou hodinu, kdyby měli všichni lidé na světě stejnou spotřebu jako on.
U této části textu se mi velmi líbí to, že uživateli neříkám, jaká je skutečná hodinová spotřeba lidstva. Říkám mu, jakou by mělo lidstvo hodinovou spotřebu, pokud by se všichni lidé chovali stejně jako on. Díky tomu se každá malá změna v chování uživatele v této sekci velmi výrazně projeví. Je totiž násobená počtem všech lidí na světě, což je v čase psaní této aplikace 7,7 miliardy lidí (rok 2019/2020).
Toto chování aplikace je žádoucí, protože by opět mělo uživatele motivovat k lepším výsledkům. Já jsem tuto motivaci pocítil, když jsem viděl své vlastní výsledky.

Pro zdroje typu odpadu, tj. všechny zdroje kromě vody a energie obsahuje popis ještě poslední odstavec, který je pro všechny tyto zdroje totožný.
Informuje uživatele o tom, kolik odpadu lidstvo vyprodukovalo za uplynulý kalendářní rok, podle stránky sensoneo.com [44] a srovnává přihlášeného uživatele s průměrným obyvatelem Země.

3.5.4 Zajímavé jednotky

Jedním z mých požadavků na aplikaci bylo přiblížení množství vyprodukovaného odpadu uživateli tak, aby si dokázal představit, o jak velkém množství se hovoří. Když jsou čísla moc velká, je těžké si pod nimi něco představit. Seznam použitých zajímavých jednotek je níže – v tabulce 3.1.

Tabulka 3.1: Zajímavé jednotky

Zdroj	**Suma**	**Hodinová spotřeba**
Papír	List papíru	Strom
Ostatní pevné zdroje	Horské gazely	Nákladní automobil
Voda	Vany	Olympijský plavecký bazén
Energie	Zvednutí slona do výšky 1m	Cesta na Měsíc a zpět raketou Saturn V (používaná v programu Apollo)

Jednotky jsem vybíral tak, aby celková uživatelova suma daného zdroje vypadala velká i v těchto nových jednotkách. Takže jsem použil relativně lehké věci pro hmotnost (například list papíru, horská gazela), malé věci pro objem a i relativně malou práci pro energii. Když ale uživateli ukazuji, jak velké množství odpadu, vody a energie se spotřebuje každou hodinu, chci opravdu podtrhnout velikost těchto čísel. Nejlépe to asi ukazuje jednotka energie, která je vyjádřená v energii nutné pro cestu Apolla 11 na Měsíc a návratu jeho posádky na Zem zpět.

Prozatím používám jednotky zobrazené v tabulce 3.1. Do budoucna bych ale chtěl přidat více jednotek. Tato změna udělá aplikaci více zajímavou bez toho, aby vyžadovala velké úsilí. Mohl bych například používat různé jednotky v závislosti na velikosti čísla, které chci reprezentovat. Mohl bych mít i množinu jednotek sloužících ke stejnému účelu a vybírat z nich při sestavování textu náhodně. Tím bych mohl prodloužit dobu, než se uživateli výsledky „okoukají“. Chtěl bych jednotky také více rozlišit podle zdroje, ke kterému se váží.

3.6 Autorizace a autentizace

ECO Tracker bude mít potenciálně mnoho uživatelů, každý z nich bude moci příspěvky přidávat, sdílet, hodnotit, editovat a mazat. Je proto nutné, aby aplikace byla schopná mezi uživateli rozlišovat. Každý uživatel musí mít možnost upravit svoje příspěvky, ale nesmí být schopen měnit příspěvky ostatních.

O uživateli vím následující:

- Přihlašovací jméno
- Heslo
- Aktuálně používaný email
- Email, kterým se uživatel zaregistroval

ECO Tracker má dvě aktivity, pomocí kterých se může uživatel přihlásit.

1. LogInActivity (snímek obrazovky A.1)
2. RegistrationActivity (snímek obrazovky A.2)

Pomocí LogInActivity se může přihlásit uživatel s již existujícím účtem, použije své přihlašovací jméno a heslp.

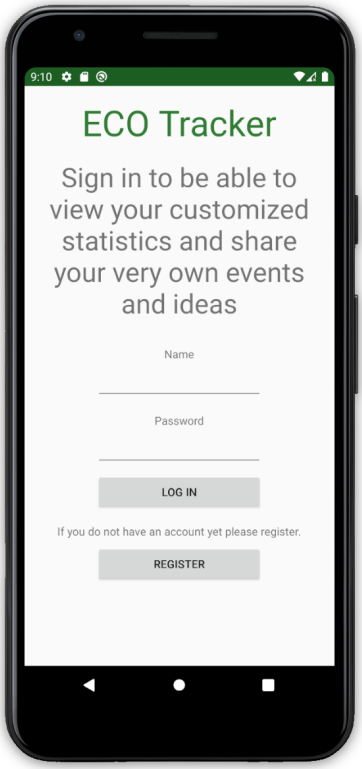
V RegistrationActivity si uživatel účet teprve vytváří. Musí uvést všechny potřebné údaje – přihlašovací jméno, heslo a email. Po vytvoření účtu je uživatel automaticky přihlášen.

Při prvním přihlášení, nebo při registraci si ECO Tracker uloží uživatelovy přihlašovací údaje. Aplikace uživatele automaticky přihlásí pomocí těchto informací při každém dalším spuštění.

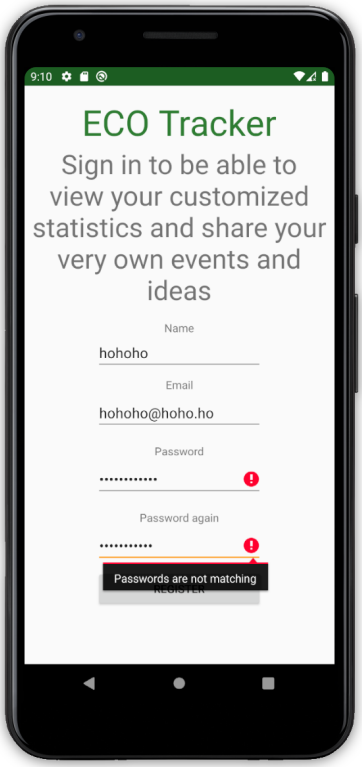
Ověření správnosti přihlašovacích údajů na serveru zajišťuje Spring Security [10]. Pokud byly přihlašovací údaje správné, odešle serverová aplikace android klientovi Json Web Token (JWT) [45], kterým je uživatel autorizován a který serverová aplikace vyžaduje pro další komunikaci.

Protože android aplikace posílá na server citlivá data, jako jsou přihlašovací údaje, JWT nebo email, musí být i tento přenos zabezpečený.
Abych umožnil přenos dat přes Https, musel jsem vygenerovat certifikát, který jsem uložil na server. Serverovou Spring Boot aplikaci jsem nakonfiguroval tak, aby tento certifikát používala při komunikaci s klienty.

V android části aplikace ke komunikaci se serverem používám knihovnu OkHttp [46]. Při vytváření Http klienta jsem přidal do jeho seznamu známých certifikátů i můj vygenerovaný certifikát. Klient certifikátu, a proto i serveru důvěřuje a je ochotný s ním komunikovat.



Obrázek 3.13: Přihlašovací obrazovka



Obrázek 3.14: Ověřování registračních údajů

Pro účely testování, snadného sestavení a použití aplikace ostatními je certifikát na serveru verzován a v android části zapsán přímo do kódu aplikace. Tímto způsobem pouze demonstruji funkčnost řešení. Při nasazení aplikace se bude muset použít jiný certifikát, který nebude veřejně přístupný. Tato změna bude vyžadovat pouze minimální úpravy v konfiguraci aplikací a žádnou změnu zdrojového kódu.

3.7 Testování

Cílem testování softwaru je ověření správnosti kódu a odhalení chyb. Testy jsou také důležité do budoucna. Až se program změní, musím mít jistotu, že všechny jeho části dál fungují správně. V neposlední řadě jsou testy zároveň i druhem dokumentace. Při náhledu do testů vidím, jakým způsobem se má která třída či metoda volat, jaké vstupy vyústí v chybu programu a podobně [47].

Pokrytí kódu testy se nazývá *Code Coverage*. Je to číslo uváděné v procentech a udává množství tříd/metod/řádků kódu, pro které existuje nějaký test. Toto číslo je jenom orientační, stoprocentní code coverage nezaručuje pokrytí všech možností průchodu programem. Nemá vypovídající hodnotu ani o kvalitě testů. Říká pouze, jak velká část kódu je otestovaná aspoň nějakým testem a jak velká část není [48].

Testy softwaru se dají rozdělit do mnoha skupin. Při implementaci ECO Trackeru jsem správnost kódu ověřoval unit (jednotkovými) testy a manuálními testy.
Unit testy jsou malé a rychlé, měly by se zaměřovat jen na jednu jednotku programu, například jednu funkci nebo metodu [49].
Integrační testy jsou větší, pokrývají zpravidla několik tříd a ověřují správnost komunikace mezi nimi [49].

Při tvorbě programu pro Android se testy rozdělují ještě na další dvě kategorie pojmenované Instrumentální testy (také UI testy) a Unit testy [50].
Instrumentální testy potřebují mít ke svému běhu spuštěný systém Android. Na něm se potom spouští testy samotné. Tyto testy se nachází zpravidla ve složce *AndroidTest*.
Druhý název – unit testy – je matoucí. Význam tohoto termínu jsem vysvětlil v předchozím odstavci, zde je však význam jiný. Pokud vyvíjím pro Android, pojmem unit testy myslím takové testy, které pro svůj běh nevyžadují spuštěné android prostředí. Můžu je tedy spustit přímo na počítaci, na kterém aplikaci vyvíjím (Windows/MacOS/Linux ne Android). Tyto testy se nachází ve složce *test* a většinou jsou to skutečně unit testy v pravém slova smyslu. Nic mi ale nebrání v napsání malého integračního testu i zde.

Code coverage EKO Trackeru je 23% řádků kódu pouze ze složky test, která obsahuje zhruba polovinu všech mých unit testů.

Aplikaci jsem testoval během celého vývoje a díky testování jsem odhalil mnoho chyb. Některé z nich uvádím i s příčinou, nebo s řešením, jako příklad níže.

- Nesprávné výsledky v sekci Statistiky.
  Chybou bylo použití celočíselného dělení namísto desetinného při generování statistik.
- Při opětovném spuštění aplikace se ve formuláři zobrazily již odeslané položky.
  Špatně jsem pracoval s datumy a použil = namísto == při porovnávání.
- Používání plovoucího tlačítka v sekcích Ideas a Events působilo selhání testů.
  Řešením bylo místo metody *launchFragmentInContainer* spustit nejdříve obalující aktivitu pomocí *ActivityScenario.launch* a následně v této aktivitě spustit testovaný fragment manuálně.
- Rotace zařízení smazala stav aktuální obrazovky.
  Ukládání stavu v metodě *onSaveInstanceState* situaci vyřešilo.

3.8 Serverová část aplikace a databáze

Serverová část ECO Trackeru je Spring Boot aplikace. K jejímu sestavení používám nástroj Maven [12]. Rozhodl jsem se používat Kotlin i pro vývoj této části projektu.
Serverová aplikace je poměrně jednoduchá, s android aplikací komunikuje pomocí REST rozhraní a poskytuje jí tímto způsobem veškeré informace. Logika, kterou server obstarává navíc (kromě zprostředkování přístupu k databázi) zahrnuje autorizaci a autentizaci, výpočet statistik, logiku týkající se hodnocení nápadů a připojování se k událostem, notifikace uživatelů o změnách jejich událostí a filtrování a řazení zobrazovaných dat (aplikace například nezobrazuje v sekci události takové události, které již proběhly).

Pro komunikaci s databází používám framework Hibernate [11], díky tomu nejsem závislý na konkrétní implementaci databáze a konkrétním dialektu SQL jazyka. V současné chvíli používám pro ECO Tracker databázi Oracle [51], mohu ji však kdykoli vyměnit bez nutnosti zasahovat do zdrojového kódu.

Závěr

Cílem práce bylo navrhnout, implementovat a otestovat mobilní aplikaci pro platformu Android, která by zpracovávala data o produkci odpadu uživatelem a generovala pro něho statistiky na základě těchto informací. Do požadavků také patřila správa událostí a nápadů. Vymezené cíle jsou splněny podle popisu v sekci *Vymezení rozsahu*. V práci jsem se zabýval analýzou existujících řešení, analýzou vlastního řešení, jeho implementací a testováním. Během implementace aplikace jsem několikrát zjistil, že můj původní návrh řešení není ideální a byl jsem nucen ho upravit.

Vzniklá aplikace ECO Tracker je stabilní a obsahuje velké množství implementovaných funkcí. S vývojem ale ještě nekončím, seznam funkcionalit které chci do aplikace v dalších fázích projektu přidat obsahuje například komentáře k událostem, chytré notifikace, cachování příspěvků pro offline použití a mnoho dalšího.

Původně jsem chtěl aplikaci zveřejnit uživatelům v obchodě Google Play již nyní a další funkce přidávat postupně formou aktualizací. Během psaní této práce jsem ale zjistil, že vývoj nativních aplikací pro operační systém Android je příliš nízkoúrovňový a zdlouhavý proces. Přišel jsem také do styku s frameworkem nesoucím název Flutter [52]. Ten je od společnosti Google a klade si za cíl výrazně zjednodušit a urychlit vývoj mobilních aplikací. Výsledná aplikace navíc může být spuštěna jak na Androidu, tak i na IOS. Pro vývoj dalších fází projektu jsem se proto rozhodl přejít k této technologii. Aplikaci tak budu muset napsat znovu, i tak ale vnímám přínos této práce. Pomohla mi rozmyslet si a otestovat vzhled a funkčnost uživatelského rozhraní a identifikovat v aplikaci její klíčové vlastnosti.

Ve své budoucí diplomové práci bych se mohl zabývat automatickou kontrolou obsahu. V této chvíli může do aplikace každý uživatel přidat jakýkoli nápad nebo jakoukoli událost. ECO Tracker nemá schopnost mazat spamy nebo například vulgární či reklamní příspěvky. Tento podprogram by určitě mohl zlepšit kvalitu obsahu nejen aplikace ECO Tracker.

Literatura

[1] Google IO 2019: recap and highlights. *techradar.com [online]*, květen 2019, [cit. 3. 3. 2020]. Dostupné z: `https://www.techradar.com/news/google-io-2019-keynote`

[2] developer.android.com: *Activity [online]*. [cit. 25. 2. 2020]. Dostupné z: `https://developer.android.com/reference/android/app/Activity`

[3] developer.android.com: *Fragment [online]*. [cit. 25. 2. 2020]. Dostupné z: `https://developer.android.com/reference/android/app/Fragment`

[4] developer.android.com: *Save key-value data [online]*. [cit. 25. 2. 2020]. Dostupné z: `https://developer.android.com/training/data-storage/shared-preferences`

[5] developer.android.com: *Data and file storage overview [online]*. [cit. 16. 4. 2020]. Dostupné z: `https://developer.android.com/training/data-storage`

[6] restfulapi.net: *What is REST? [online]*. [cit. 24. 3. 2020]. Dostupné z: `https://www.restfulapi.net`

[7] developer.android.com: *Seznam aplikačních rozhraní systému android. [online]*. [cit. 24. 3. 2020]. Dostupné z: `https://developer.android.com/reference`

[8] Rajput, D.: *Mastering Spring Boot 2.0: Build modern, cloud-native, and distributed systems using Spring Boot*. Packt Publishing, 2018, ISBN 9781787125148, [str. 7-8]. Dostupné z: `https://books.google.cz/books?id=JHxeDwAAQBAJ`

[9] Pivotal Software: *Spring Boot - Reference Documentation [online]*. [cit. 14. 4. 2020]. Dostupné z: `https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/`

[10] Pivotal Software: *Spring Security - Reference Documentation [online]*. [cit. 14. 4. 2020]. Dostupné z: https://docs.spring.io/spring-security/site/docs/current/reference/html5/

[11] Kumar, S.: *Spring and Hibernate*. Tata McGraw-Hill Education, 2013, ISBN 9781259063725, [str. 213-214]. Dostupné z: https://books.google.cz/books?id=jQyeAAAAQBAJ

[12] Apache.org: *Maven – Introduction [online]*. [cit. 15. 4. 2020]. Dostupné z: http://maven.apache.org/index.html

[13] Apache.org: *Maven – Introduction [online]*. [cit. 15. 4. 2020]. Dostupné z: http://maven.apache.org/what-is-maven.html

[14] Kousen, K.: *Gradle Recipes for Android: Master the New Build System for Android*. O'Reilly Media, 2016, ISBN 9781491947302, [str. 15]. Dostupné z: https://books.google.cz/books?id=10pODAAAQBAJ

[15] What does Android SDK mean? *techopedia - elektrotechnický magazín [online]*, květen 2019, [cit. 29. 3. 2020]. Dostupné z: https://www.techopedia.com/definition/4220/android-sdk

[16] Stanford University: *How does the internet work? [online]*. [cit. 23. 4. 2020]. Dostupné z: https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm

[17] cloudflare.com: *What is HTTPS? [online]*. [cit. 23. 4. 2020]. Dostupné z: https://www.cloudflare.com/learning/ssl/what-is-https/

[18] mozilla.org: *SSL/TLS [online]*. [cit. 23. 4. 2020]. Dostupné z: https://www-archive.mozilla.org/projects/security/pki/nss/ssl/

[19] Bass, L.; Clements, P.; Kazman, R.: *Software Architecture in Practice: Software Architect Practice_c3*. SEI Series in Software Engineering, Pearson Education, 2012, ISBN 9780132942782, [kapitola 1.1 What Software Architecture Is and What It Isn't]. Dostupné z: https://books.google.cz/books?id=-II73rBDXCYC

[20] javaee.github.io: *Introduction to Batch Processing [online]*. [cit. 26. 5. 2020]. Dostupné z: https://javaee.github.io/tutorial/batch-processing001.html

[21] recyclecoach.com: Recycle Coach, v. 9.10.1 [online]. https://play.google.com/store/apps/details?id=mobi.recyclecoach.worldster.pack, 10. dubna 2020.

[22] earth911.com: iRecycle, v. 3.0 [online]. https://play.google.com/store/apps/details?id=com.earth911.irecycle, 25. července 2019.

[23] mylittleplasticfootprint.org: My Little Plastic Footprint, v. 2.1.5 [online]. https://play.google.com/store/apps/details?id=com.plasticsoupfoundation.MyLittlePlasticFootprint, 10. dubna 2020.

[24] Kubásek, M.: Ukliďme Česko [online]. https://www.uklidmecesko.cz, [cit. 12. 2. 2020].

[25] eco plugs.net: ECO-Tracker, v. 1.95 [online]. https://play.google.com/store/apps/details?id=com.yu.sheng.ecotracker, 23. listopadu 2019, [cit. 12. 2. 2020].

[26] Now, G.: eco-tracker.com [online]. https://eco-tracker.com, [cit. 12. 2. 2020].

[27] University, P.: Princeton Eco Tracker [online]. https://www.ecotracker.org, [cit. 12. 2. 2020].

[28] BentallGreenOak: bentallgreenoak.ecotracker.com [online]. https://bentallgreenoak.ecotracker.com, [cit. 12. 2. 2020].

[29] [online], W. I. P. O.: https://www3.wipo.int/branddb/en, [cit. 12. 2. 2020].

[30] Mobile & Tablet Android Version Market Share Worldwide [online]. *gs.statcounter.com*, [cit. 24. 3. 2020]. Dostupné z: https://gs.statcounter.com/android-version-market-share/mobile-tablet/worldwide

[31] Oracle Corporation: *Dokumentace jazyku Java 14 [online]*. [cit. 22. 2. 2020]. Dostupné z: https://docs.oracle.com/en/java/javase/14

[32] Oracle Corporation: *Dokumentace jazyku Java 7 [online]*. [cit. 22. 2. 2020]. Dostupné z: https://docs.oracle.com/javase/7/docs/

[33] Java SE version history. *codejava.net - elektrotechnický magazín [online]*, [cit. 29. 3. 2020]. Dostupné z: https://www.codejava.net/java-se/java-se-versions-history

[34] Pecinovskỳ, R.: *Java 7: učebnice objektové architektury pro začátečníky*. Knihovna programátora, Grada, 2012, ISBN 9788024736655, [str. 40-41]. Dostupné z: https://books.google.cz/books?id=gav-1OXareQC

[35] Moskala, M.; Wojda, I.: *Android Development with Kotlin*. Packt Publishing, 2017, ISBN 9781787128989, [str. 27]. Dostupné z: https://books.google.cz/books?id=PJZGDwAAQBAJ

[36] JetBrains s.r.o.: *FAQ – Kotlin Programming Language [online]*. [cit. 22. 2. 2020]. Dostupné z: https://kotlinlang.org/docs/reference/faq.html

[37] JetBrains s.r.o.: *Comparison to Java Programming Language [online]*. [cit. 22. 2. 2020]. Dostupné z: https://kotlinlang.org/docs/reference/comparison-to-java.html

[38] JetBrains s.r.o.: *Dokumentace jazyku Kotlin v. 1.3.6 [online]*. [cit. 22. 2. 2020]. Dostupné z: https://kotlinlang.org/docs/reference/android-overview.html

[39] JetBrains s.r.o.: *Data Classes – Kotlin Programming Language [online]*. [cit. 22. 2. 2020]. Dostupné z: https://kotlinlang.org/docs/reference/data-classes.html#data-classes

[40] Moskala, M.; Wojda, I.: *Android Development with Kotlin*. Packt Publishing, 2017, ISBN 9781787128989, [str. 9]. Dostupné z: https://books.google.cz/books?id=PJZGDwAAQBAJ

[41] developer.android.com: *Kotlin and Android [online]*. [cit. 22. 2. 2020]. Dostupné z: https://developer.android.com/kotlin

[42] developer.android.com: *Pickers [online]*. [cit. 16. 4. 2020]. Dostupné z: https://developer.android.com/guide/topics/ui/controls/pickers

[43] Wach, L.: Hello Charts [online]. "https://github.com/lecho/hellocharts-android", [cit. 5. 3. 2020].

[44] sensoneo.com: Sensoneo [online]. "https://sensoneo.com/sensoneo-global-waste-index-2019/", [cit. 15. 3. 2020].

[45] AuthO: *JSON Web Tokens [online]*. [cit. 16. 4. 2020]. Dostupné z: https://jwt.io/

[46] https://square.github.io/okhttp/: *OkHttp [online]*. [cit. 24. 4. 2020]. Dostupné z: https://square.github.io/okhttp/

[47] Fowler, M.; Beck, K.: *Refaktoring: zlepšení existujícího kódu*. Moderní programování, Grada Publishing, 2003, ISBN 9788024702995, [str. 96-106]. Dostupné z: https://books.google.cz/books?id=hsvzc-XE3nYC

[48] Arnošt, H.; Rudolf, P.: *JUnit 5: Jednotkové testování na platformě Java*. Grada Publishing a.s., 2018, ISBN 9788024712253, [str. 226-227]. Dostupné z: https://books.google.cz/books?id=fBeXDwAAQBAJ

[49] Arnošt, H.; Rudolf, P.: *JUnit 5: Jednotkové testování na platformě Java*. Grada Publishing a.s., 2018, ISBN 9788024712253, [str. 34]. Dostupné z: https://books.google.cz/books?id=fBeXDwAAQBAJ

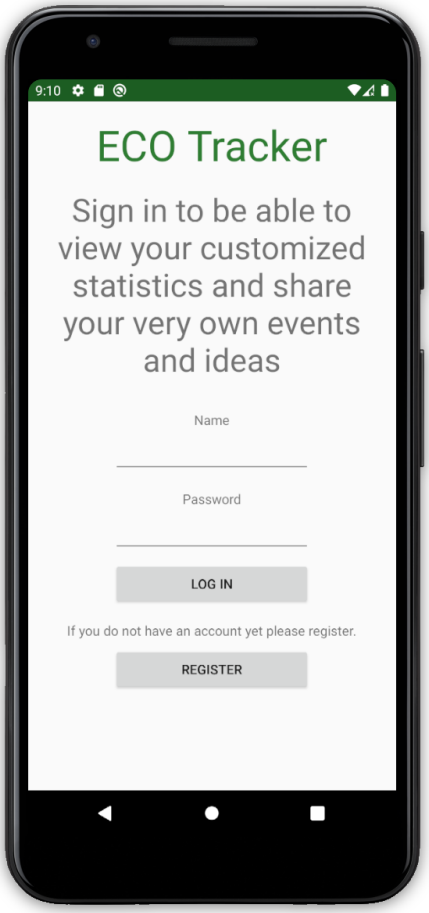
[50] developer.android.com: *Fundamentals of Testing [online]*. [cit. 16. 4. 2020]. Dostupné z: https://developer.android.com/training/testing/fundamentals

[51] Oracle Corporation: *Databáze – Oracle [online]*. [cit. 14. 4. 2020]. Dostupné z: https://www.oracle.com/cz/database/

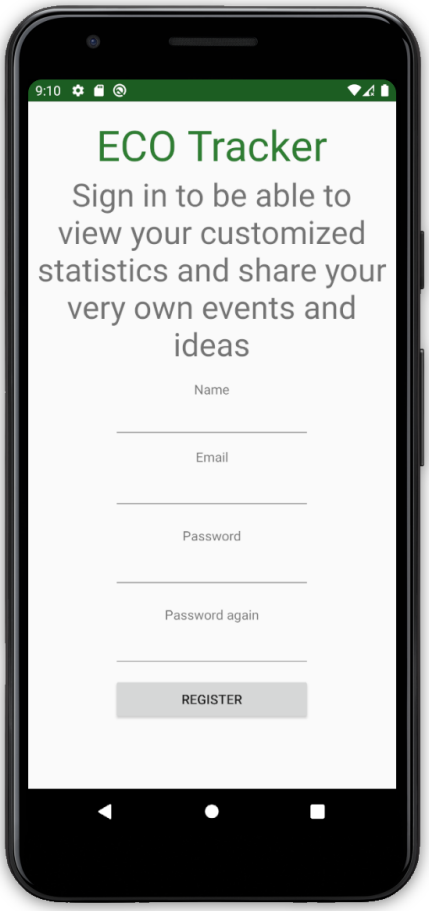
[52] Google.org: *Flutter – Beautiful native apps in record time [online].* [cit. 2. 5. 2020]. Dostupné z: `https://flutter.dev/`

Příloha A

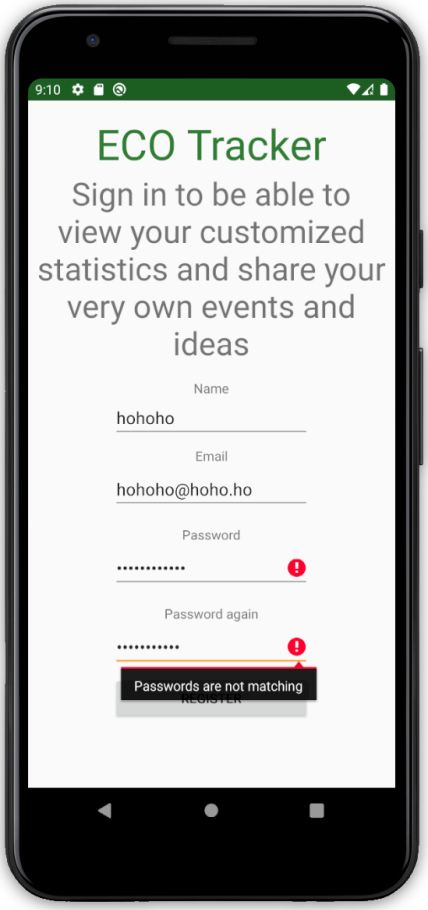
Snímky obrazovek



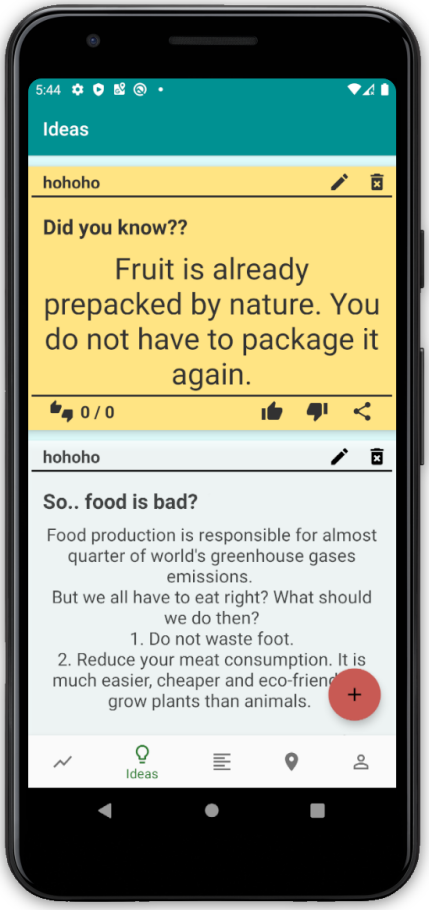
Obrázek A.1: Přihlašovací obrazovka



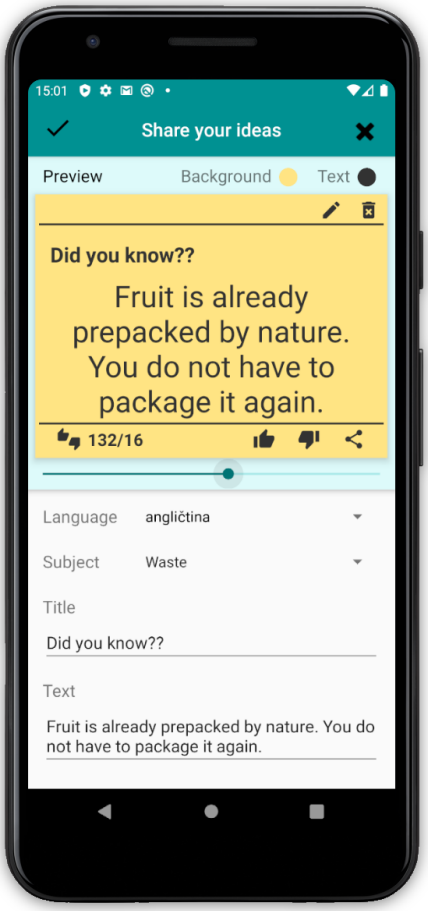
Obrázek A.2: Registrační obrazovka



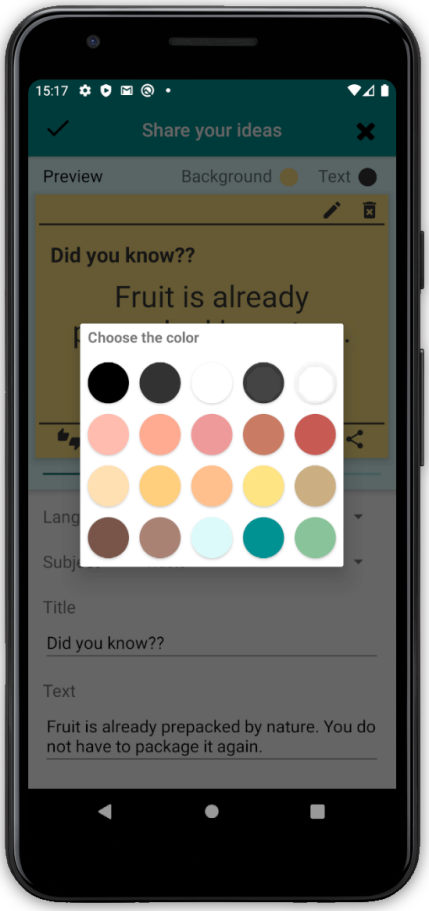
Obrázek A.3: Ověřování údajů



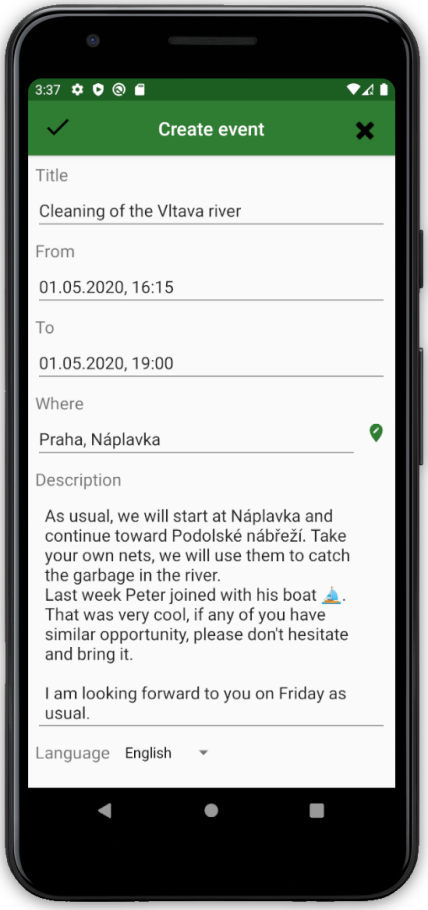
Obrázek A.4: Sekce nápady



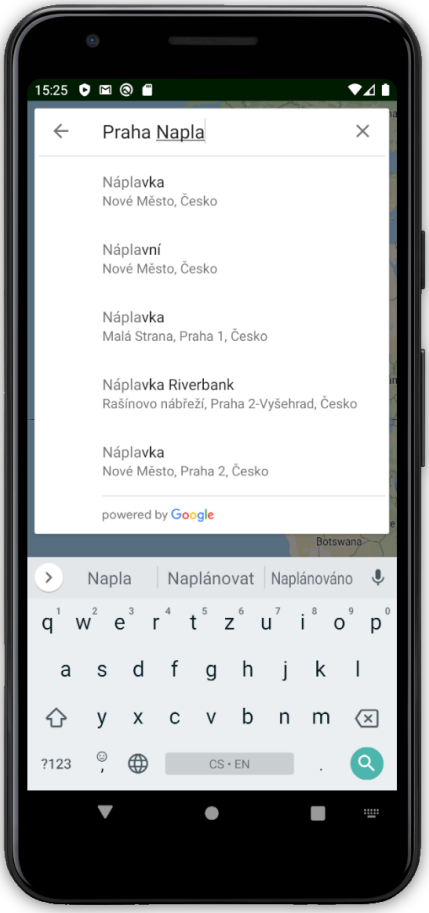
Obrázek A.5: Vytváření nápadu



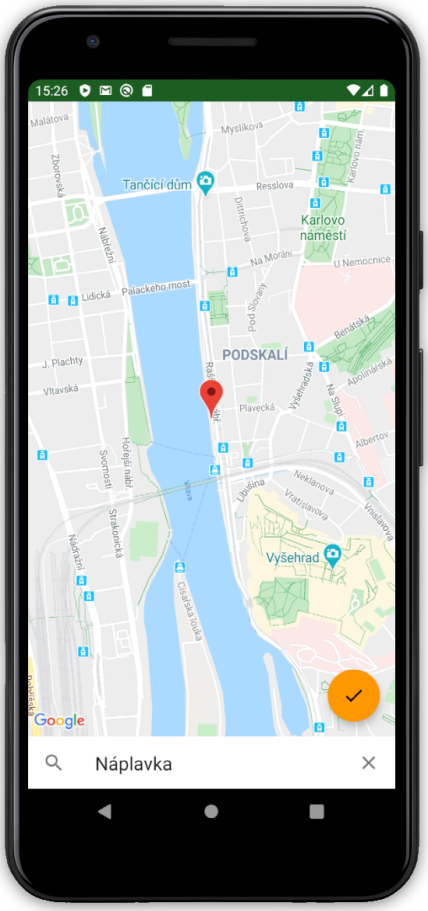
Obrázek A.6: Úprava vzhledu nápadu



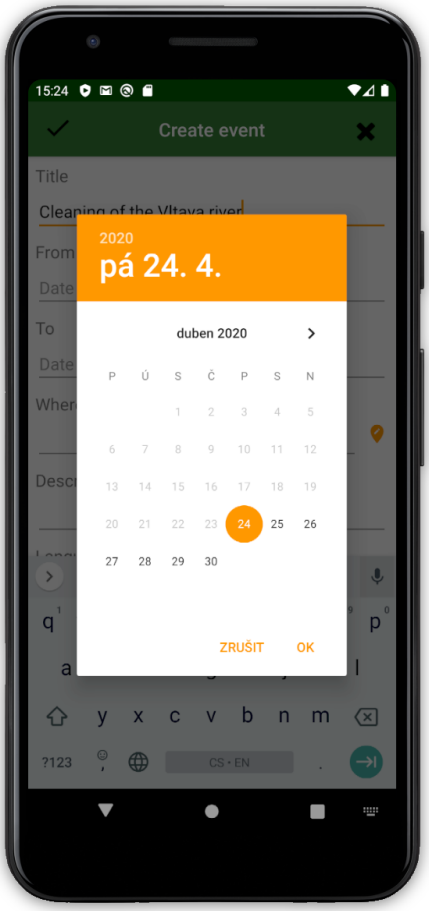
Obrázek A.7: Vytváření události



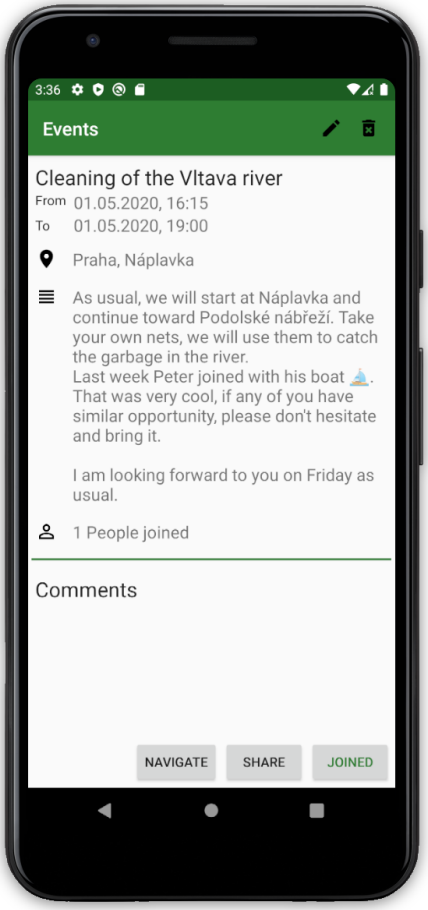
Obrázek A.8: Upřesnění místa konání události



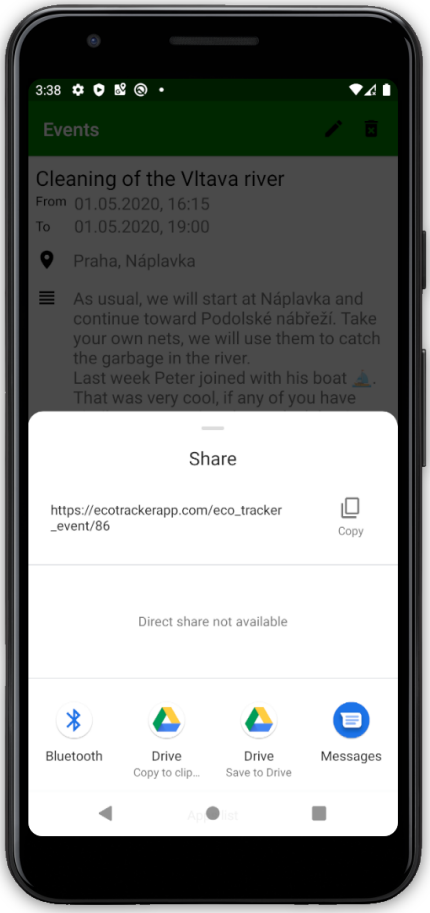
Obrázek A.9: Upřesnění místa konání události



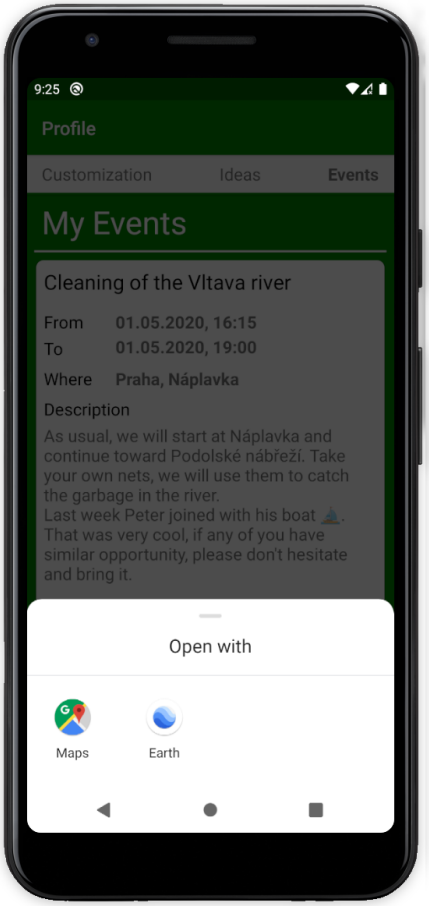
Obrázek A.10: Výběr data konání události



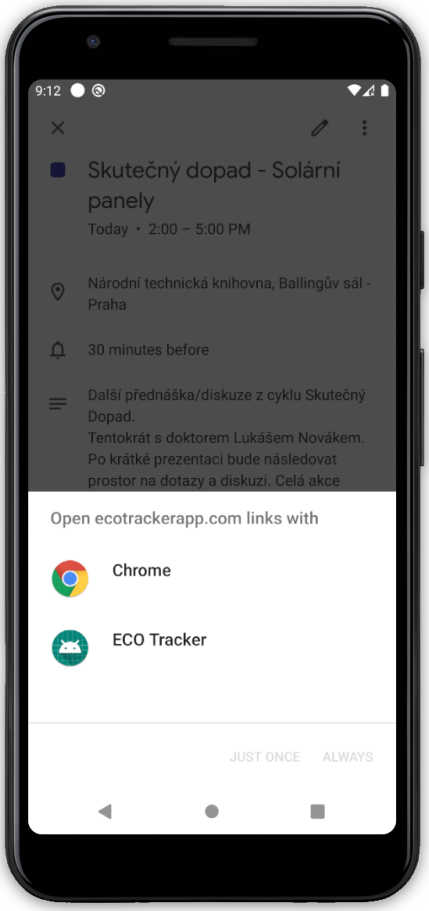
Obrázek A.11: Detail události



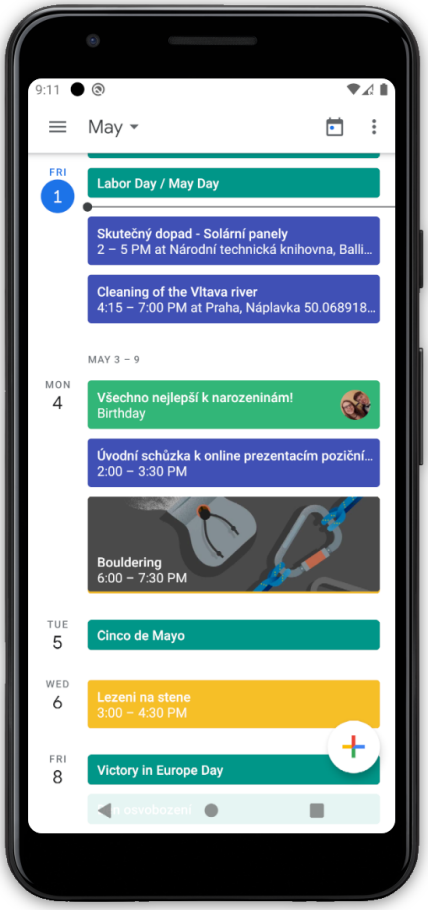
Obrázek A.12: Sdílení události



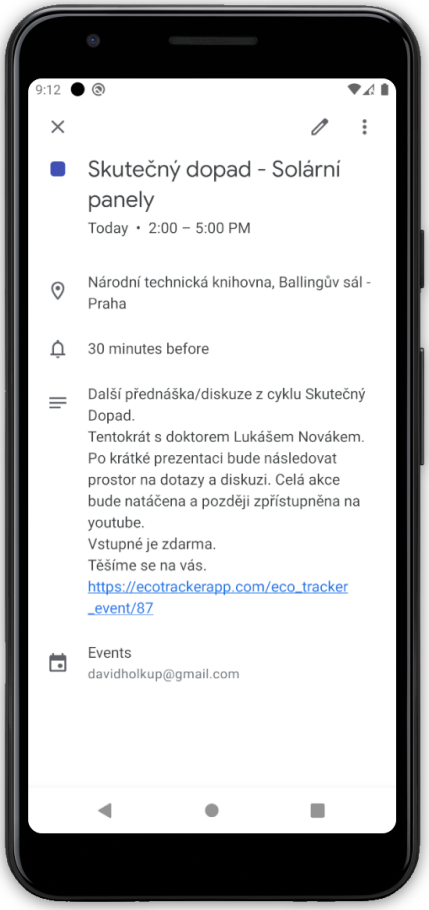
Obrázek A.13: Navigace k místu konání události



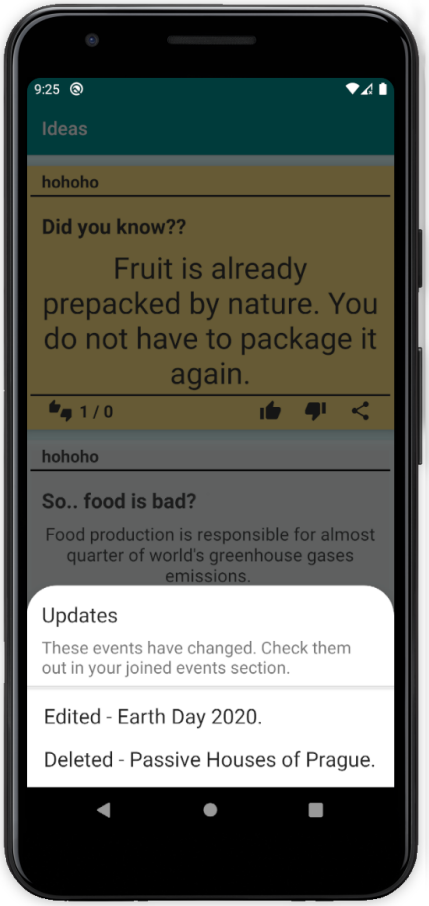
Obrázek A.14: Otevření sdílené události



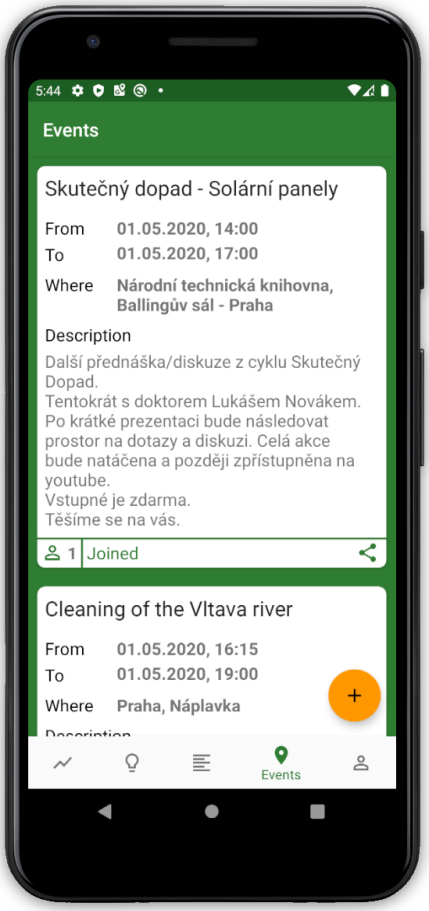
Obrázek A.15: Uživatelovi události přidané do Google kalendáře



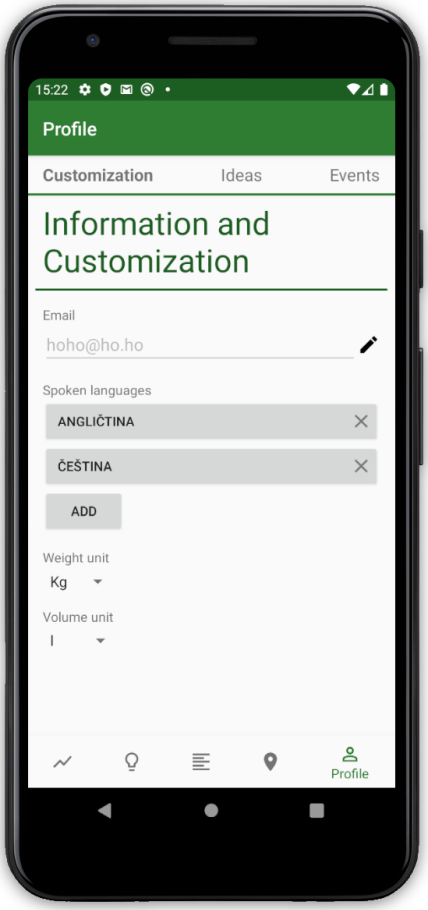
Obrázek A.16: Detail události v Google kalendáři



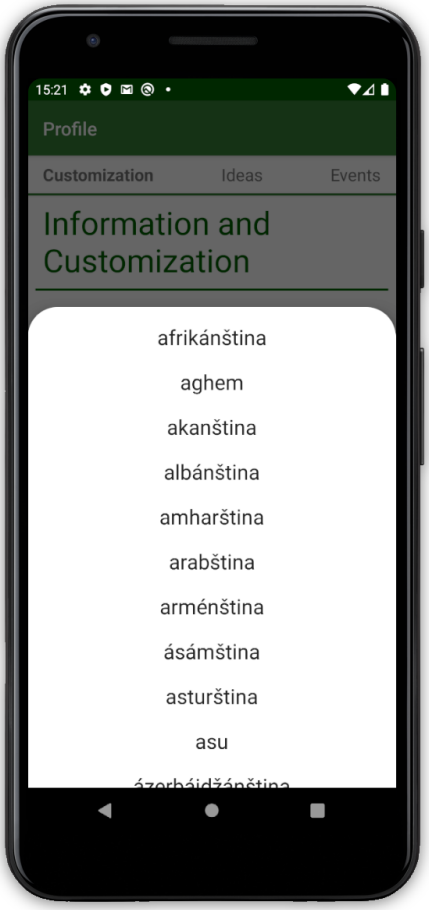
Obrázek A.17: Informace o změněných událostech, ke kterým se uživatel předtím připojil



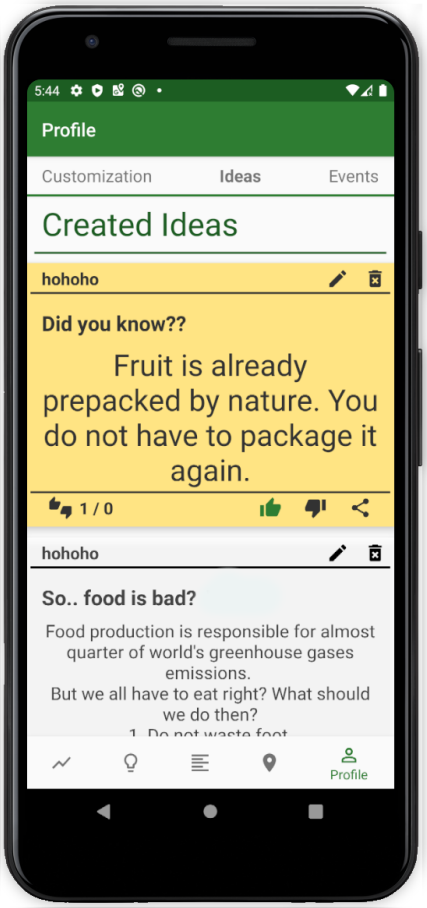
Obrázek A.18: Sekce události



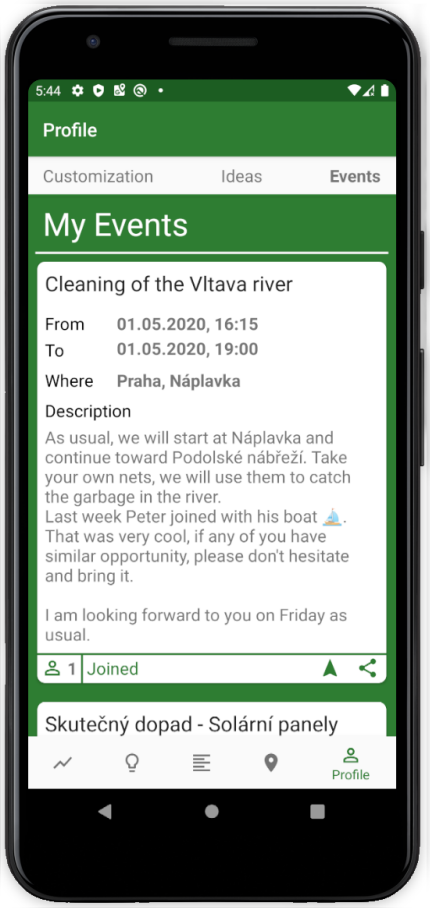
Obrázek A.19: Přizpůsobení aplikace



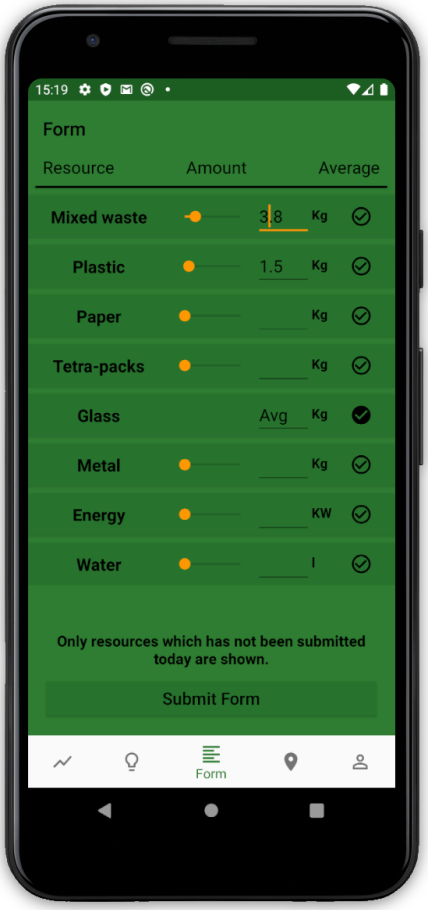
Obrázek A.20: Výběr jazyků



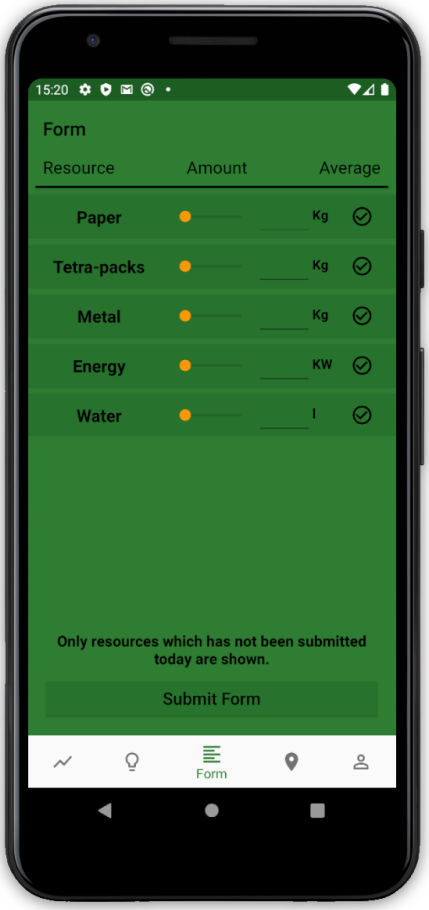
Obrázek A.21: Vytvořené nápady



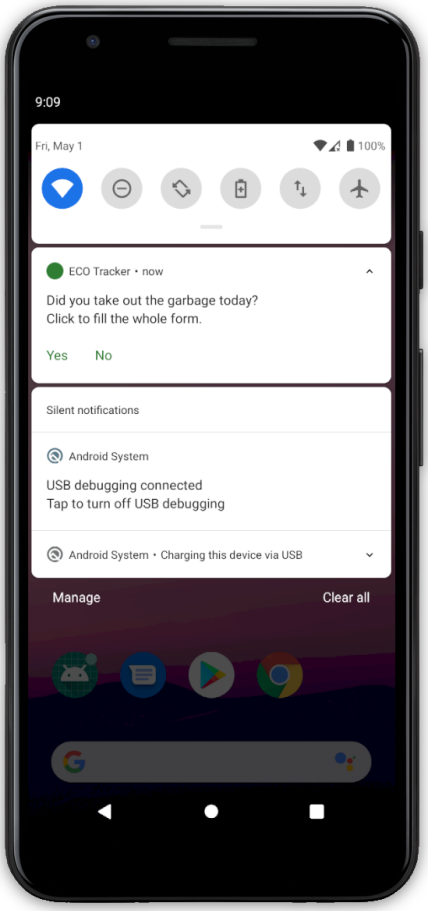
Obrázek A.22: Vytvořené události



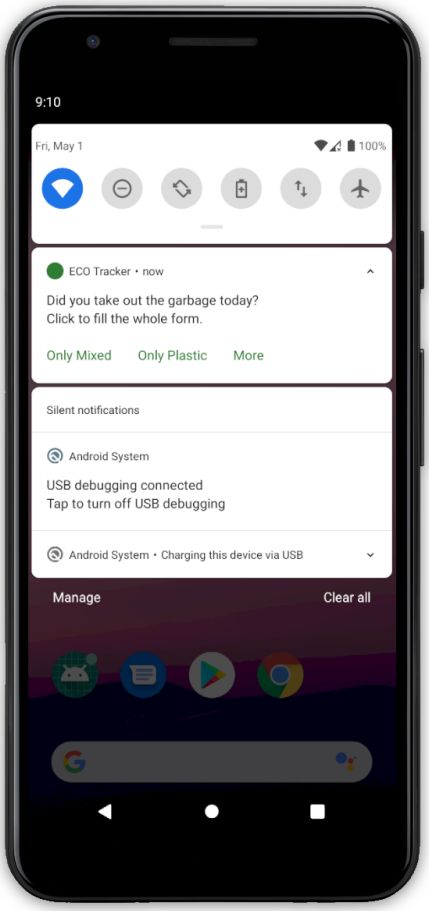
Obrázek A.23: Formulář



Obrázek A.24: Formulář bez již odeslaných položek

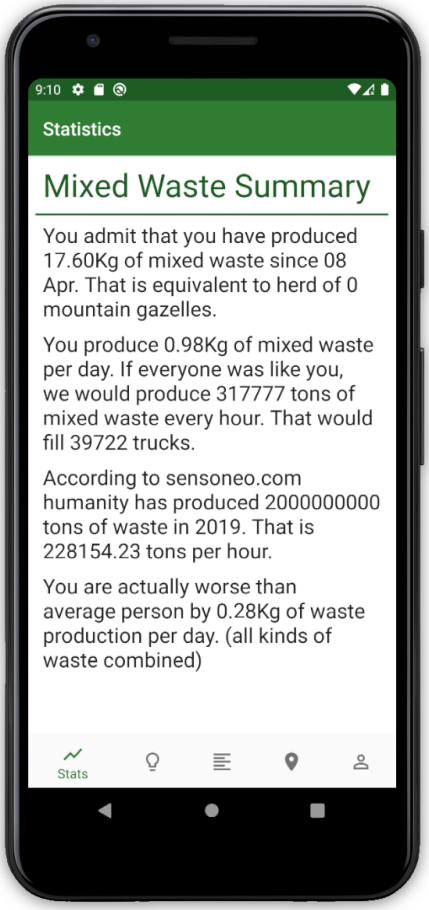


Obrázek A.25: Notifikace formuláře – 1. krok



Obrázek A.26: Notifikace formuláře – 2. krok

Obrázek A.27: Statistiky



Obrázek A.28: Textový popis statistik

Příloha **B**

Obsah přiloženého CD

```
readme.txt ..... stručný popis obsahu CD
bin ..... adresář se spustitelnou formou implementace
    eco-tracker.apk ..... mobilní aplikace ECO Tracker
src ..... adresář se zdrojovými kódy
    eco-tracker ..... zdrojové kódy Android aplikace
    eco-tracker-backend ..... zdrojové kódy serverové aplikace
text ..... text práce
    thesis.pdf ..... text práce ve formátu PDF
    thesis.zip ..... zdrojová forma práce ve formátu LaTeX
```