



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Android aplikace DayWork.cz: návrh UI a aplikace pro firmy
Student:	Kryštof Novotný
Vedoucí:	Ing. Robert Pergl, Ph.D.
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Portál DayWork.cz slouží pro párování nabídek a poptávek brigád. Práce je součástí projektu vytvoření Android aplikace pro pracovníky a Android aplikace pro firmy. Cílem této práce je především vytvoření návrhu UI pro obě aplikace a vývoj aplikace pro firmy.

- 1) Navrhněte UI wireframe modely pro obě aplikace a odsouhlaste se zadavatelem.
- 2) Vytvořte ve spolupráci se zadavatelem grafický návrh.
- 3) Implementujte a otestujte aplikaci pro firmy, která bude postavena na architektuře vytvořené v paralelně běžící bakalářské práci kolegy Michala Mročka "Android aplikace DayWork.cz: základní modul". Zaměřte se zejména na následující funkčnosti:
 - profil (úprava, zobrazení)
 - vytvoření brigády (kompletně nová nebo pomocí předvyplnění dat z jiné brigády)
 - kontinuální ukládání na úložiště při vytváření pro případ pádu aplikace
 - seznam vlastních brigád
 - detail brigády
 - potvrzení / odmítnutí brigádníka
 - zrušení brigády
 - seznam brigádníků u dané brigády

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

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

V Praze dne 10. ledna 2018



**FAKULTA
INFORMAČNÍCH
TECHNOLÓGIÍ
ČVUT V PRAZE**

Bakalářská práce

Android aplikace DayWork.cz: návrh UI a aplikace pro firmy

Kryštof Novotný

Katedra softwarového inženýrství
Vedoucí práce: Ing. Robert Pergl, Ph.D.

27. června 2018

Poděkování

Děkuji svému vedoucímu Ing. Robertu Perglovi, Ph.D. za rady ohledně struktury práce a pozitivní přístup. Děkuji kolegům Michalu Mročkovi a Michalu Sousedíkovi za ochotnou pomoc s překážkami při vývoji. V neposlední řadě děkuji své rodině za materiální i emoční podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické 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 předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, 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 způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 27. června 2018

.....

České vysoké učení technické v Praze
Fakulta informačních technologií

© 2018 Kryštof Novotný. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Novotný, Kryštof. *Android aplikace DayWork.cz: návrh UI a aplikace pro firmu*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

První část mojí práce se zabývá návrhem UI pro Android aplikaci Daywork.cz a obecnými návrhovými doporučeními pro systém Android. V další části mé práce se zabývám vývojem aplikace Daywork.cz pro firemní klienty. Zde krátce pojednávám o vývoji pro Android a seznámím čtenáře se zvolenou architekturou kterou využívá implementační část mé práce.

Klíčová slova Android, UI, grafický návrh, MVVM, Java, Daywork, práce

Abstract

First part of my thesis is about developing UI for Daywork.cz Android application and general Android UI development principles. In the second part I describe development of the Daywork.cz Android application bussiness module. I also refer briefly about the development for Android in general in this part and I make reader familiar with the architecture used in my application.

Keywords Android, UI, graphical design, MVVM, Java, Daywork, jobs

Obsah

Úvod	1
1 Cíle a metodika	3
2 Rešerše	5
2.1 Obecná doporučení pro návrh Android UI	5
2.2 Výběr nástroje pro vývoj wireframů	5
2.3 Nástroje pro vývoj	6
2.4 O Vývoji pro Android	6
2.5 Použitá architektura	9
2.6 Api	10
3 Analýza a návrh	11
3.1 Úvod analytické části	11
3.2 Obrazovky aplikace	11
4 Realizace	17
4.1 Úvod části s realizací	17
4.2 Implementace jednotlivých funkcionalit	17
4.3 Průchody aplikací	19
5 Testování	21
5.1 Jednotkové testy	21
5.2 Testy s využitím Android Frameworku	21
6 Diskuze a práce do budoucna	23
Závěr	25
Literatura	27

A	Obrazovky brigádníka	29
B	Obrazovky firmy	39
C	Slovníček pojmů a zkratk	47
D	Obsah přiloženého flashdisku	49

Seznam obrázků

3.1	Moje brigády	12
3.2	Nabídka brigád	13
3.3	Srovnání UI na webu a v aplikaci	15
A.1	Přihlášení	30
A.2	Zobrazení brigád	30
A.3	Registrace - osobní údaje	31
A.4	Registrace - znalosti	31
A.5	Registrace - zkušenosti a představení	32
A.6	Registrace - konfekční velikost, věk, média	32
A.7	Filtr lokací	33
A.8	Filtr pracovišť	33
A.9	Konverzace	34
A.10	Detail	34
A.11	Nastavení emailů	35
A.12	Nastavení notifikací	35
A.13	Kalendář	36
A.14	Notifikace	36
A.15	Profil pracovníka	37
B.1	Registrace 1	40
B.2	Registrace 2	40
B.3	Registrace 3	41
B.4	Správa brigád	41
B.5	Nová brigáda 1	42
B.6	Nová brigáda 2	42
B.7	Nová brigáda 3	43
B.8	Nová brigáda 4	43
B.9	Nová brigáda 5	44
B.10	Nová brigáda 6	44

B.11 Pracovníci	45
B.12 Profil společnosti	45

Seznam tabulek

4.1 Průchody aplikací	20
---------------------------------	----

Úvod

Portál Daywork.cz funguje jako prostředník mezi brigádníky a firmami, které brigády vypisují. Narozdíl od známých portálů s nabídkami práce (jobs.cz, prace.cz, ...) má Daywork.cz unikátní funkcionalitu. Dokáže spárovat brigádníky s brigádou tak, aby každý brigádník dostával co možná nejrelevantnější nabídky. Díky tomu je výběr pro brigádníka rychlejší a snažší. Tato funkcionalita stojí na povinnosti každého brigádníka vyplnit strukturovaná pracovní data o své osobě do svého profilu v aplikaci. Jedná se o data typu pracovní zkušenosti, IT znalosti, jazyky, a další.

Firmy vyplní stejný typ strukturovaných dat při vytváření každé z brigád. Na základě průniku těchto dat jsou pak brigádníkům zobrazovány brigády s požadavky podle jejich profilu.

V současné době je portál Daywork.cz v ostrém provozu a plně funkční. V době tvorby aplikace však nebylo využívání chytrých telefonů natolik rozšířené, takže je webová aplikace navržena primárně pro užití na desktopu. V současné době provozovatel poptává nativní aplikaci pro systém Android, která využije stávajícího řešení, ale zároveň přinese uživatelům lepší komfort při používání na mobilním telefonu.

Tato aplikace bude výsledkem souběžně probíhajících bakalářských prací z nichž každá implementuje jednu z klíčových funkcionalit aplikace. Práce byla mezi mě a mé kolegy rozdělena takto:

- Michal Mroček: Android aplikace DayWork.cz: základní modul Navrhne a implementuje základní architekturu aplikace a funkcionality sloužící pro přihlašování, odhlašování a registraci
- Kryštof Novotný: Android aplikace DayWork.cz: návrh UI a aplikace pro firmy Navrhne UI jak pro brigádníky, tak firemní část aplikace, implementuje funkcionalitu aplikace pro firmy
- Michal Sousedík: Android aplikace DayWork.cz: modul pro brigádníky Implementuje funkcionalitu pro brigádníky a funkcionalitu chatu mezi

firmami a brigádníky

Implementační část mé práce tedy závisí na základním modulu kolegy Michala Mročka a využívá z něj třídy a funkcionality. Na části kolegy Sousedíka bude moje implementace nezávislá s výjimkou některých entit společných jak pro firmu, tak brigádníky a funkcionalitu zobrazování a nahrávání obrázků, která je identická pro obě části aplikace.

Cíle a metodika

Cílem první části práce je navrhnout uživatelské rozhraní mobilní aplikace Daywork.cz pro Android (dále jen aplikace) jak pro brigádníky, tak pro firmy. Úkol budu řešit takto:

1. Provedu všechny průchody webovou aplikací a zaznamenám všechny usecasey^[C] aplikace.
2. Najdu a nainstaluji vhodný editor wireframů.
3. Vytvořím wireframy^[C] podle usecasů webové aplikace.

Cílem druhé části je implementovat funkcionalitu mobilní aplikace pro firmy s využitím architektury paralelně běžící bakalářské práce kolegy Michala Mročka. Úkol budu řešit takto:

1. Nainstaluji Android studio a lokální Daywork server běžící v prostředí Pharo
2. Seznámím se s architekturou aplikace navrženou kolegou
3. Dle svých návrhů implementuji požadované funkcionality
4. Funkcionalita otestuji

Požadované funkcionality jsou:

1. Zobrazení a úprava profilu
2. Vytvoření brigády (kompletně nová nebo pomocí předvyplnění dat z jiné brigády)
3. Kontinuální ukládání na úložiště při vytváření pro případ pádu aplikace
4. Seznam vlastních brigád

1. CÍLE A METODIKA

5. Detail brigády
6. Potvrzení / odmítnutí brigádníka
7. Zrušení brigády
8. Seznam brigádníků u dané brigády

Rešerše

2.1 Obecná doporučení pro návrh Android UI

Pro operační systém Android existují designová doporučení, která mají zaručit konzistentní vzhled a chování aplikací, ale také přehlednost a dobrou uživatelskou zkušenost. V době tvorby této aplikace je hlavním UI^[C] paradigmatem pro Android Material Design[1]. Nejdůležitější doporučení z hlediska ovladatelnosti a použitelnosti jsou:

- Velikost klikatelných elementů Neměla by být menší než 48x48dp[2]
- Obsah okna: Nahoře je umístěn toolbar^[C] s popisem činnosti která je prováděna, toolbar může mít Overflow menu^[C] vpravo a šipku zpět vlevo[3]
- Navigace prostřednictvím spodního menu[4] obsahuje 3 - 5 ikon, konzistentně zobrazena napříč všemi obrazovkami, sdružuje akce podobné důležitosti. Do tří ikon využívají jak ikonu, tak textový popisek

2.2 Výběr nástroje pro vývoj wireframů

Na návrh wireframů existuje v současné době velké množství programů a obecně možností jak k tomuto úkolu přistoupit. Aplikace budu posuzovat z těchto hledisek: learning curve, efektivita pro účel, cena. Z nepřeberného množství programů na internetu budu vybírat z těchto 3:

- Vektorový grafický editor Inkscape[5] - Výhodou je že v něm již umím pracovat a je zdarma, nevýhodou že je příliš obecný, takže bych šablony obrazovek musel dělat kompletně od začátku
- JustinMind prototyping tool[6] - Výhodou je že je velice profesionální, umí nativně nahrávat prototyp do cloudu a následně ho spouštět na

mobilním zařízení. Má veliké množství nastavení a funkcionalit, z nichž některé jsou placené.

- Pencil project[7] - Freewareový editor který je jednoduchý na učení a je možné do něj snadno stáhnout šablony pro prvky systému Android.

2.3 Nástroje pro vývoj

Vývoj pro Android probíhá buď v jazyce Java nebo Kotlin s použitím Android Frameworku, který je pravidelně zveřejňován společností Google. Pro sestavení a správu dependencies se používá Gradle. Tyto nástroje jsou součástí vývojového prostředí Android Studio[8].

V práci budu používat tyto knihovny / moduly.

- `daywork.core`- modul implementovaný kolegoou Michalem Mročkem. Zajišťuje sestavení aplikace s pomocí dependency injection a poskytuje pomocné metody pro obsluhu `ViewModel`ů.
- `dagger`- zajišťuje dependency injection^[C]
- `room`- zajišťuje práci s databází a implementuje za nás SQL příkazy v `DAO`^[C] objektech
- `android.arch.lifecycle.extensions:1.1.0` - umožňuje používat `View-Modely`
- `com.squareup.retrofit2:retrofit:2.3.0` - generuje nízkoúrovňové HTTP requesty naší aplikace
- `com.squareup.retrofit2:converter-gson:2.3.0` - odpovídá za serializaci objektů do JSONu^[C] a parsování JSONu na objekty
- `ButterKnife` - slouží pro snazší binding `Views` do referencí v aktivitě, nebo přiřazování listenerů k `Views`.

2.4 O Vývoji pro Android

Android framework definuje pro vývojáře základní třídy které slouží jako stavební kameny aplikace. S těmito třídami se pracuje tak, že vývojář zdědí funkcionalitu frameworkem definované třídy a ve své implementaci přepíše (`@Override`) metody k tomu určené. Typicky `onCreate()` a další metody související s životním cyklem(viz 2.4.5).

Vzhled uživatelského rozhraní se definuje v XML^[C] souborech, z nichž Android Framework vytvoří (`inflate`) třídy typu `View`.

2.4.1 View

Základním kamenem uživatelského rozhraní Android aplikace je `View`. `View` je abstraktní třída, od které dědí všechny elementy uživatelského rozhraní (`Button`, `TextView`, `EditText`, `RadioButton`, ...).

Aby bylo možné elementy skládat do skupin a pozicovat, existuje `ViewGroup`, což je předek všech kontejnerů na prvky UI. Sám dědí od `View`, takže je možné `ViewGroup` libovolně zanořovat (Jen teoreticky. Z důvodu výkonu není vhodné vytvářet příliš hluboké struktury).

Příkladem nejčastěji užívaných `ViewGroup` jsou:

- `LinearLayout` - umísťuje svoje zanořené views (children) buď horizontálně, nebo vertikálně, podle svého atributu `android:LayoutOrientation`
- `ConstraintLayout` - umožňuje pozicovat children pomocí omezení (constraints). Constraints jsou pravidla, která definují umístění jednoho `View` v závislosti na jiném `View`, nebo kontejneru samotném (parent). `ConstraintLayout` je velmi užitečný, protože dokáže vytvořit velice komplexní layout, aniž by bylo potřeba zanořovat `ViewGroups` do sebe.
- `FrameLayout` zaujme určený prostor v jeho parentovi. Používá se jako kontejner na `Fragment` 2.4.3

,

2.4.2 Aktivita

`Activity` (Aktivita), je komponenta, která reprezentuje jednu konkrétní akci prováděnou uživatelem (vyplnění dat, výběr položky ze seznamu). Aktivita obsahuje kořenový `ViewGroup` a v něm `Views` reprezentující různé elementy pro interakci. Aktivita je zpravidla fullscreenová (i když nemusí být). Může, ale nemusí obsahovat `Fragmenty`.

2.4.3 Fragment

`Fragment` zapouzdřuje určitou malou část UI funkcionality. Narozdíl od `Activity`, `Fragment` vždy musí být pro svoji funkčnost připojen k aktivitě. Jedna aktivita může mít i více fragmentů současně, což je užitečné při návrhu UI pro tablety, nebo při implementaci navigace v aplikaci. `Fragment` obsahuje stejně jako `Activity` kořenový `ViewGroup` s `Views` uvnitř.

2.4.4 Perzistence dat

Pro perzistenci dat podporuje Android SQLite databázi, možnost ukládání na filesystem zařízení, nebo `SharedPreferences`^[C]. V naší aplikaci volíme

SQLite databázi nad kterou je postaven Room Framework z Lifecycle Components. Ušetří nám to psaní kódu který pracuje s databází. Díky Room frameworku stačí programátorovi definovat abstraktní DAO entitu, její metody a metody označit anotacemi `@Query`. Room Framework pak z entity vygeneruje implementaci DAO objektů.

2.4.5 Životní cyklus aktivit a fragmentů

Běžná desktopová aplikace běží dokud nesplní svůj úkol, nebo ji nezavřeme, či nezhavaruje. Narozdíl od ní, jednotlivé části Android aplikace mohou být systémem při nedostatku zdrojů bez varování ukončeny, v případě že s nimi uživatel právě neinteraguje. Vždy také dochází k ukončení a znovu vytvoření Aktivity při změně konfigurace zařízení (rotace, vysunutí hw klávesnice, připojení do doku, ...) a je třeba s tím při návrhu počítat.

Tomuto chování se říká životní cyklus a nesprávné obslužení životního cyklu mívá za následek ztrátu dat z uživatelského rozhraní, chybné chování aktivity, nebo pád celé aplikace.

Nad životním cyklem Aktivit nemá vývojář žádnou kontrolu. Celý ho řídí systém a odpovědností vývojáře je na jednotlivé události životního cyklu správně reagovat. To se provádí přepsáním metod třídy Activity v potomkovi, kterého implementujeme. Na Lifecycle události se reaguje přepsáním Lifecycle callbacks, což jsou metody, které na sobě Aktivita zavolá při událostech životního cyklu.

Lifecycle callbacks pro aktivitu jsou:

- **onCreate** Volá se při vytvoření aktivity systémem. Zde se očekává, že dojde k nastavení layoutu a načtení dat, pokud se jedná o restart aktivity.
- **onStart** Aktivita se spouští
- **onResume** Při zavolání této metody se mohou spolehnout že aktivita běží a je viditelná uživatelem
- **onPause** V této metodě došlo k částečnému překrytí aktivity jinou aktivitou nebo dialogem. Již v tomto stavu může být již v tomto stavu aktivita ukončena, je tedy nutné uložit data z UI pro případ že se tak stane.
- **onStop** Aktivita není vidět uživatelem, může dojít k ukončení aktivity. Data je ale dobré ukládat již v **onPause**
- **onDestroy** Voláno těsně před tím než je aktivita systémem zavřena. Nemusí být vůbec zavolána, pokud systém již aktivitu ukončil dříve.

Obdobný cyklus implementuje i `Fragment`, který vždy kopíruje cyklus aktivity ke které je připojen. V případě, že se v aktivitě nacházejí data, je potřeba je v `onPause` uložit do systémové instance třídy `Bundle`. Systém `Bundle` zachová pro případ, že bude daná aktivita opět potřeba a následně doručí v metodě `onCreate`, kde je na vývojáři, aby data z `Bundle` prezentoval zpět v UI.

2.5 Použitá architektura

Modul bude záviset na hlavním modulu z bakalářské práce Michala Mročka, který provedl návrh architektury a implementaci funkcionalit přihlášení, odhlášení, registrace.

Jako použitá architektura byla zvolena MVVM^[C][9], která je od uvedení `Lifecycle Components`[10] podporována přímo vývojáři systému.

Důvody pro volbu této architektury jsou především snadná testovatelnost a vyvarování se problémů při nesprávném ošetření životního cyklu aktivit. Narozdíl od standardní obsluhy životního cyklu popsáno v předchozí sekci, komponenta `ViewModel` řeší obsluhu životního cyklu za programátora.

Architektura aplikace vypadá následovně:

- Aktivita, `Fragment` - Stará se o zobrazení dat uživateli a přijímání dat od uživatele sleduje `ViewModely`, ze kterých načítá data pro zobrazení uživateli a ukládá do nich data, která je třeba zachovat
- `ViewModel` slouží jako skladiště dat pro aktivitu, monitoruje životní cyklus aktivit které ho sledují tak, aby se k nim při jejich restartu mohl opět připojit. Drží v sobě referenci na `Repository`
- `Repository` slouží pro načítání dat do `ViewModelu` a to buď z databáze, nebo `Api`. drží si referenci na `Connector`
- `Connector` slouží pro konkrétní obstarání dat, buď z `API`, nebo z databáze. drží si referenci na `TaskManager`
- `TaskManager` spouští asynchronní načítání a ukládání dat
- `Task` reprezentuje vlákno které provede načtení nebo uložení požadovaných dat

Implementace vypadá v praxi tak, že Aktivita si v metodě `onCreate` zaregistruje `Observery` na všechny `ViewModely` od kterých bude zobrazovat data a následně na změnu dat reaguje v zaregistrované metodě.

2.6 Api

Aplikace využívá funkcionalitu stávající webové aplikace s využitím REST API^[C][11], které je implementováno serverem. Api komunikuje ve formátu JSON^[C].

V mojí Aplikaci je Api reprezentováno třídou `CompanyApi`, která má metodu pro každý Api endpoint^[C]. Třída `CompanyApi` je abstraktní, její konkrétní implementaci zajišťuje knihovna `retrofit2`, která na základě deklarace metod vytvoří konkrétní http requesty.

Analýza a návrh

3.1 Úvod analytické části

V této části se hodlám věnovat návrhu podoby aplikace a to jak po stránce funkční, tak grafické. Cílem této části práce je vytvořit wireframy pro firemní i podnikovou část aplikace. Výsledkem bude konkrétní grafická podoba aplikace včetně naznačení interakce s jednotlivými prvky.

3.2 Obrazovky aplikace

Při navrhování aplikačních obrazovek jsem se snažil zachovat podobnost ovládání s webovou aplikací tak, aby prvky byly podobné a zároveň odpovídaly designovým doporučením pro Android.

Aplikace slouží pro dva oddělené typy uživatelů - firmy a brigádníky. Zatímco firmy vypisují brigády a poptávají brigádníky, brigádníci práci hledají. UI se pro oba dva typy uživatelů liší, nicméně některé prvky mají společné.

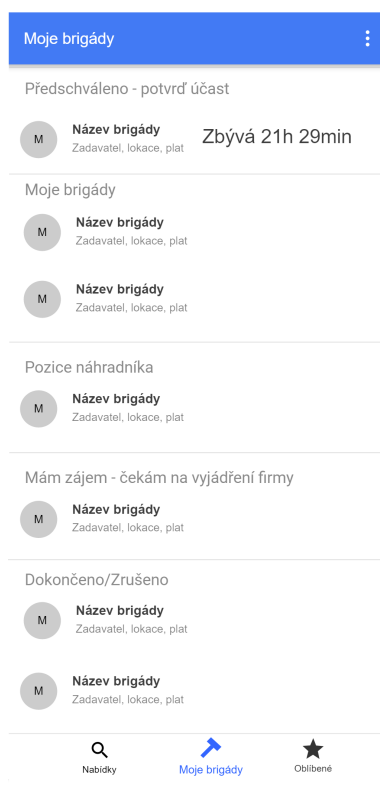
Nákresy důležitých obrazovek jsou k dispozici v přílohách A, B. Kompletní sada všech obrazovek včetně webového prototypu s naznačením přechodu mezi obrazovkami je k dispozici na doprovodném médiu ve složce daywork-wireframes.

3.2.1 Brigádník

Webová aplikace pro navigaci spoléhá na postranní lištu s možnostmi. Při převodu na mobilní aplikaci se nabízí převést lištu na tzv. "hamburger menu"/ Navigation drawer^[C], čímž by navigace byla téměř 100% konzistentní s webovou aplikací. Nicméně Navigation Drawer bývá uživateli často přehlédnut, zvolil jsem tedy variantu s permanentní navigací vespod obrazovky, která obsahuje nejdůležitější akce.

Zároveň jsem sjednotil navigaci na "Mám zájem", "Čekám na potvrzení" a "Moje brigády" pod jednu obrazovku, neboť představují totéž - brigády jed-

3. ANALÝZA A NÁVRH



Obrázek 3.1: Pracovník - moje brigády

noho brigádníka. Rozdíl v těchto brigádách představuje jen stav schválení ze strany firmy a brigádníka, což bude na obrazovce reprezentováno odlišným pořadím v seznamu. 3.1

Akce které nejsou tak důležité jsem přesunul do Overflow menu^[C], kde jsou stále poměrně snadno přístupné ale nezabírají místo.

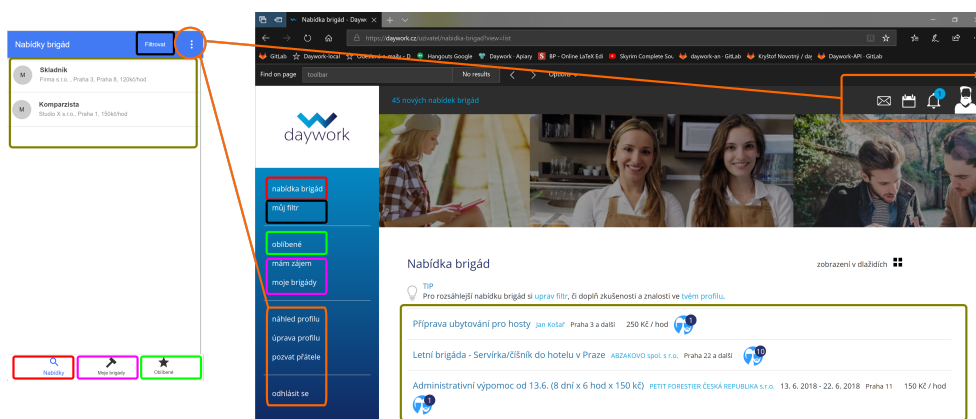
Akce pro filtrování brigád bude v mojí aplikaci dostupná jen z obrazovky "Nabídky brigád", neboť jen pro tuto obrazovku je relevantní a ani ve webové aplikaci nemá na jiné brigády, než na ty právě hledané, žádný vliv.

Úprava webového designu do obrazovky mobilní aplikace na obrázku zde: 3.2

Brigádníkovy nejdůležitější způsoby užití jsem v návrhu zohlednil takto:

- Registrace následuje styl webové aplikace. Brigádník postupně prochází obrazovkami "Osobní údaje"- "Vzdělání a znalosti"- "zkušenosti a představení"- "Média a ostatní". Obrazovky jsou v pořadí podle důležitosti informací od nejdůležitějších k těm nejméně důležitým.

Jediný výraznější rozdíl oproti webové aplikaci je ve způsobu přidávání položek na obrazovce 2 A.4 u "Jazyky" a "IT schopnosti", kde jsem zvolil



Obrázek 3.2: Pracovník - nabídka brigád

pro Android typické přidávání položek přes `FloatingActionButton`.

- Obrazovka přihlášení umožní přihlásit, registrovat a nebo použít Google / Facebook účet. Funkcionalita se neliší od webové verze
- Pro zobrazení profilu jsem zvolil jednu dlouhou obrazovku místo seznamu karet, který je na webu z důvodu zjednodušení navigace. Zároveň jsem přesunul nastavení upozorňování do sekce nastavení, protože v profilu chci zobrazovat jen data relevantní k osobě uživatele a nikoli chování aplikace.
- Obrazovku úprava profilu jsem z návrhu mobilní aplikace odstranil - profil se bude upravovat na obrazovce se zobrazenými údaji bez nutnosti spustit editaci. Jedná se o velmi intuitivní a přirozený způsob práce, který je na mobilních zařízeních častý.
- Zobrazení seznamu brigád (nabídky, brigády kterých se účastní / chce účastnit, oblíbené brigády) jsem pouze zjednodušoval. Chybí přepínač mezi zobrazením v řádcích a dlaždicích, na mobilním displeji bude stačit pouze zobrazení v řádcích. Seznam brigád je reprezentován standardním `RecyclerView`^[C], a na každé z položek je název brigády, provozovatel a částka.
- Zobrazení detailu brigády - zobrazuji stejné informace o brigádě jako na webu, v horní části jsou tlačítka pro obsluhu brigády se stejnou funkcí jako v původní aplikaci.
- Nastavení filtru je přístupné přes obrazovku s nabídkami brigád. Samotný filtr se kvůli velkému množství položek otevírá v samostatné aktivitě, kde je možno nastavovat lokace a pracoviště.

- Obsluha brigády (projevení zájmu, odmítnutí, potvrzení) probíhá tlačítky v horní části náhledu detailu brigády. Tlačítka mají stejný popis i funkcionalitu jako u webové aplikace.
- Návrh nastavení kalendáře spočívá v rozpadu kalendáře z webové aplikace na seznam termínů - celý kalendář včetně událostí v něm by se na displej dost dobře nevešel, takže se období nepřítomnosti definují jako položky typu od - do.
- Komunikace s firmou - tato část se inspirovala od nativní aplikace Zprávy, kde můžeme přes seznam konverzací přejít ke konkrétní konverzaci a mluvit. Seznam konverzací je podobně jako na webu členěn po trojicích brigádník-firma-brigáda a řazen podle firem.

3.2.2 Společnost

U společnosti tvoří podstatnou část webové aplikace nabídka brigád vypsanych danou společností. Z pohledu použitelnosti uživatelského rozhraní mi na ale na navigaci ve webové aplikaci několik věcí nevyhovuje. Jsou to:

- Nutnost překlíkávání mezi obrazovkami které stejně zobrazí jen seznam brigád a liší se pouze tlačítkem pod brigádou (v obrázku označeno červeným rámečkem 3.3). Toto by bylo možné vyřešit třemi tlačítky stále viditelnými pod každou brigádou.
- Při výběru akce s brigádou se UI chová v rozporu s očekáváním. V elementu reprezentujícím brigádu jsou 3 typy prvků - hyperlink s názvem brigády, nevyplněné tělo prvku, tlačítko s názvem akce. Ačkoli hyperlink i tlačítko reagují po svém na akci najetí myši, všechny vedou na stejnou akci - tu která je právě zvolena v panelu nalevo. Hyperlink a tlačítko jsou tedy zbytečné, kdyby se místo zbarvení tlačítka nebo hyperlinku zbarvil celý element, zpětná vazba by byla pro uživatele jasnější.

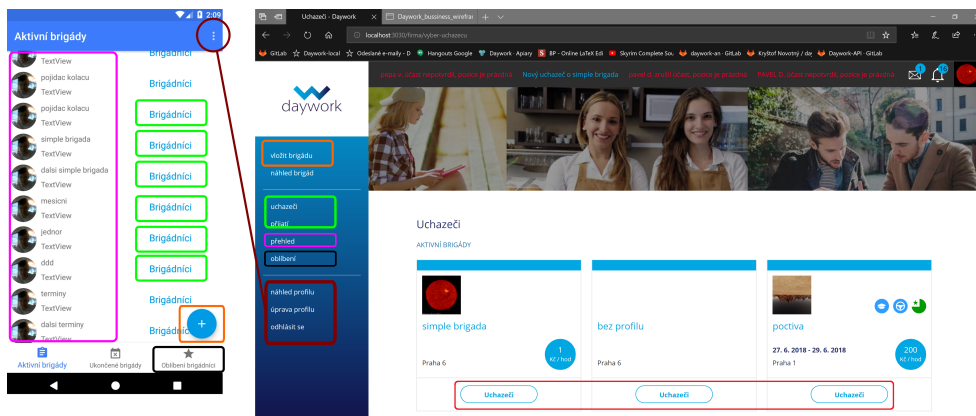
Tyto nesrovnalosti jsem se rozhodl nepřenášet do UI mobilní aplikace a místo toho zvolit jiný systém navigace. Na seznamu brigád se bude kliknutím na brigádu na detail brigády, a klikem na link "brigádníci" se bude přecházet do další aktivity, ve které bude firma spravovat brigádníky od příslušné brigády.

Klíčové use case společnosti jsou:

- Založení nové brigády - buď nové, nebo ze šablony podle stávající brigády. Zde jsem se v pořadí vyplňovaných informací držel webové předlohy s výjimkou posledního kroku kdy jsem oddělil média od mzdy a popisu brigády, protože obrazovka byla příliš dlouhá a informace dost nesourodě na to, abych oddělením zpřehlednil práci.

Podobně jako u brigádníka je přidávání položek zkušeností a jazyků do seznamů řešeno přes `FloatingActionButton`.

3.2. Obrazovky aplikace



Obrázek 3.3: Srovnání UI na webu a v aplikaci

- Zobrazení detailu brigády je ekvivalentem zobrazení detailu brigádníka. Firma zde přehledně vidí všechny aspekty brigády, které uvidí i brigádník.
- Obsluha uchazečů brigády (přijmutí, odmítnutí, náhradník, odvolání) se provádí podobně jako ve webové aplikaci z obrazovky s detailem brigádníkovy profilu. Ke konkrétním akcím slouží patřičně pojmenovaná tlačítka v horní části obrazovky s profilem.
- Zobrazení profilu - Podobně jako u brigádníka se k profilu přistupuje přes Overflow menu v pravém horním rohu obrazovky. Narozdíl od webové aplikace jsem profil sjednotil na jednu obrazovku s informacemi oddělenými standardními Dividery abych zlepšil přehlednost a snížil nutný počet kliknutí.
- Úprava profilu probíhá z obrazovky zobrazení profilu kliknutím na informaci, kterou si uživatel přeje editovat. Podobně jako u brigádníka jsem vynechal obrazovku dedikovanou pro úpravu profilu.
- Zobrazení profilu brigádníka - Zde jsou zobrazeny podobně jako na webu v jediné obrazovce všechny informace o brigádníkovi.
- Zprávy jsou obdobně jako u brigádníka řešeny podobně jako nativní aplikace Zprávy

Realizace

4.1 Úvod části s realizací

V této části se hodlám věnovat implementaci modulu aplikace pro firemní zákazníky. Tento modul bude součástí aplikace, která je vyvíjena souběžně s touto bakalářskou prací. Kolega Michal Mroček vyvíjí architekturu aplikace, Michal Sousedík modul pro brigádníky.

4.2 Implementace jednotlivých funkcionalit

- Zobrazení a úprava firemního profilu - Realizováno aktivitou `CompanyProfileActivity`. Ta sleduje `CompanyProfileGetViewModel` pro načítání dat a `CompanyProfileUploadViewModel` pro změny profilu. Odpověď z `Api` se deserializuje do `CompanyProfileAbout` entity, která obsahuje všechna data profilu společnosti. V databázi je reprezentován entitou `CompanyProfileDao`.
- Náhled detailu brigády - realizován `CompanyJobDetailActivity`, ve které je `JobDetailFragment`. Tento fragment využívám jak pro náhled vytvářené brigády, tak pro detail již aktivní brigády, proto je mu nutné v parametru předat, jestli se jedná o novou brigádu, nebo ne. Podle toho fragment nastaví uživatelské rozhraní (zobrazí / skryje tlačítka pro obsluhu brigády, nebo aktivaci brigády).
Přenos dat do aktivity má na starosti `CompanyJobDetailViewModel`, databázi pak `CompanyJobDetailDao`.
- Vytvoření brigády, včetně možnosti vyplnění dat z jiné brigády - Obsluhuje aktivita `CreateJobActivity`, která při přechodu na další krok vymění `Fragment` s uživatelským rozhraním. Jednotlivé kroky vytvoření brigády reprezentují fragmenty `CreateJob1Fragment`, `CreateJobFragment_2` - `6`). Aby byla zajištěna perzistence dat během vytváření brigády

(jeden z požadavků v zadání), je rozpracovaná brigáda po každé změně uložena do databáze prostřednictvím `JobInTheMakingSaveViewModel` a `JobInTheMakingDao`.

Data rozpracované brigády reprezentuje entita `JobInTheMaking`.

Během vytváření brigády nedochází k žádné komunikaci s API a veškerá data jsou ukládána výhradně do vnitřního úložiště. V případě obrázku nedochází během vytváření brigády k jejich nahrávání na server, pouze se do entity uloží jejich URI^[C] v rámci zařízení. Nahrávání dat nové brigády je řešen naráz až po dokončení celého procesu vytvoření brigády.

Vyplnění dat z jiné brigády je řešen podobně jako náhled detailu brigády s tím rozdílem, že entita `CompanyJobDetail` reprezentující detail brigády je převedena do `JobInTheMaking`. Toto řešení jsem zvolil proto, že entita `CompanyJobDetail` vzniká deserializací JSONU po přijetí odpovědi na detail brigády z API. `JobInTheMaking` obsahuje ta samá data, ale zároveň do ni ukládám stavy přepínačů UI během vytváření brigády, takže ji nelze tak snadno serializovat na JSON a odeslat.

- Kontinuální ukládání vytvářené brigády na úložiště je implementováno v rámci vytváření brigády. Data z předvyplněné brigády je z UI možné také vymazat a začít vytvářet brigádu od začátku.
- Náhled seznamu vlastních brigád je implementován aktivitou `CompanyJobsActivity`, která dle nastavení spodní navigace přepíná zobrazené fragmenty. Ty reprezentují buď aktivní brigády - `ActiveJobsFragment`, nebo ukončené (z jakéhokoli důvodu) brigády - `FinishedJobsFragment`. Tyto fragmenty sledují `GetActiveJobsViewModel` a `GetFinishedJobsViewModel`. Jejich dotaz na API je stejný, k filtrování aktivních a dokončených prací dochází až na úrovni databáze.
- Potvrzení / odmítnutí brigádníka Tyto akce má na starosti `WorkerDetailActivity` která zároveň zobrazuje profil brigádníka. Sleduje `ViewModel`ly pro číselníky, které jsou nutné pro interpretaci dat na profilu pracovníka a pak také `WorkerDetailViewModel`, `GetWorkersForJobViewModel`, `GetFavoriteWorkersViewModel`, `AddFavoriteWorkerViewModel`. Z těchto tříd je zjevná funkcionálnost jen u `WorkerDetailViewModel`, který je zodpovědný za načítání dat o profilu brigádníka.

Abych ale mohl brigádníka přijmout, potřebuji nejprve zjistit, z jaké brigády jsem se na detail pracovníka dostal. Jeden pracovník totiž může mít u stejné firmy poptávku po vícero brigádách, a na této obrazovce firma rozhoduje jen o jedné dvojici brigáda-brigádník. To je vyřešeno argumentem, který je předán fragmentu při jeho vytvoření. Fragment má v argumentu public key brigády, takže díky tomu zjistí z api, jací uchazeči k této brigádě patří.

Toto má na starosti `GetWorkersForJobViewModel`. Na základě jeho dat zjistím status právě otevřeného pracovníka rámci dané brigády. Tato informace ovlivní, jaké tlačítko (přijmout / odvolat / odmítnout) se zobrazí.

`GetFacoriteWorkersViewModel` získá oblíbené pracovníky dané firmy. Přijatá data ovlivní podobu tlačítka pro "oblíbení" brigádníka (odebrat z oblíbených / přidat do oblíbených).

Po provedení akce s brigádníkem - "oblíbení", odmítnutí, přijetí dojde k obnovení UI, aby byla správně zobrazena brigádníkova akční tlačítka.

- Zrušení brigády je implementováno menu položkou v Toolbaru, která je na detailu brigády vždy viditelná. Kliknutí na položku "Zrušit brigádu" otevře sadu dialogů s potvrzením a vyplněním důvodu. Po potvrzení je na API odeslán požadavek na zrušení dané brigády
- Zobrazení seznamu brigádníků u dané brigády Je implementováno `WorkerManagementActivity`, která mění své fragmenty podle `BottomNavigation`. Zobrazuje dvě instance `ApplicantsFragmentu` do kterého jsou načítání buď uchazeči, nebo již přijatí brigádníci. Data zajišťuje `GetWorkersForJobViewModel`, data jsou na přijaté a nepřijaté filtrována databází, kterou obsluhuje `GetWorkersForJobDao`.

4.3 Průchody aplikací

V této sekci popisuji, jak v moji aplikaci dosáhne uživatel funkcionalit ze zadání. Po přihlášení začíná uživatel na obrazovce "Moje brigády".

4. REALIZACE

Tabulka 4.1: Průchody aplikací

Funkcionalita	Postup aplikací
Zobrazení profilu	klik vpravo nahoře na Overflow menu → výběr "Profil"
Úprava profilu	Zobrazení profilu → kliknutí na jakýkoli editovatelný prvek → provedení editace → kliknutím mimo editovanou oblast dojde k uložení
Vytvoření nové brigády	Z obrazovky "Moje aktivní brigády" → klik na <code>\FloatingActionButton</code> vpravo dole → klik "Ne" na dialogu
Vytvoření brigády ze šablony	Z obrazovky "Moje aktivní brigády" → klik na <code>\FloatingActionButton</code> vpravo dole → klik "Ano" na dialogu
Detail brigády	Z obrazovky "Moje aktivní brigády" nebo "Moje ukončené brigády" → klik na obrázek či tělo brigády
Potvrzení brigádníka	Z obrazovky "Moje aktivní brigády" nebo "Moje ukončené brigády" → klik na text "Brigádníci" nebo z "Detail brigády" klik na "uchazeči" → klik na daného brigádníka → klik na "předschválit"
Odmítnutí brigádníka	Z obrazovky "Moje aktivní brigády" nebo "Moje ukončené brigády" → klik na text "Brigádníci" nebo z "Detail brigády" klik na "uchazeči" → klik na daného brigádníka → klik na "předschválit"
Zrušení brigády	Z obrazovky "Detail brigády" klik vpravo nahoře na "Zrušit brigádu" → Odpovědět "Ano" a volitelně vyplnit důvod
Seznam brigádníků u dané brigády	Z obrazovky "Moje aktivní brigády" nebo "Moje ukončené brigády" → klik na text "Brigádníci" nebo z "Detail brigády" klik na "uchazeči"

Testování

5.1 Jednotkové testy

Jednotkové testy slouží k otestování správné funkčnosti vybraných tříd a metod. Vzhledem k tomu, že moje práce využívá pro perzistenční vrstvu framework kolegy Michala Mročka, rozhodl jsem se testovat pouze složitější případy bussiness logiky.

Protože aplikace slouží hlavně ke zobrazování dat která přijdou z API, není příliš potřeba provádět větší množství unit testů - bussiness logiky, nebo algoritmů je v aplikaci málo.

- ProfileValidationTest - Testuje, zda funguje správně validace profilu
- CompanyJobsTest - Testuje, zda seznam brigád funguje správně
- CreateJobTest - Testuje, zda se entita CompanyJobDetail správně převede na JobInTheMaking

5.2 Testy s využitím Android Frameworku

Tyto testy testují komplexní funkcionalitu nejdůležitějších součástí aplikace. Fungují tak, že na spuštěné aktivitě simulují chování uživatele. V těchto testech využívám frameworku Espresso[12].

Výhodou těchto testů oproti jednotkovým testům je, že testují všechny vrstvy aplikace naráz. Nevýhodou je, že běží výrazně déle než jednotkové testy.

Vzhledem k tomu, že nejdůležitější úlohu hraje v aplikaci zobrazování dat do UI, jeho změna a následná perzistence, rozhodl jsem se otestovat funkcionality pracující s daty tímto způsobem:

- Programově vyplním data v dané UI komponentě

5. TESTOVÁNÍ

- provedu rotaci obrazovky, což restartuje danou aktivitu a přinutí aplikaci právě vyplněná data znovu načíst
- Ověřím že vyplněná data odpovídají těm, které si aplikace pamatuje

Vytvořil jsem tyto testovací třídy:

- **TestUtils** - Třída obsahující statické metody které obstarávají vyplnění jednotlivých UI komponent (**EditText**, **RadioButton**, **Spinner**). Zároveň zaznamenává data, která byly do komponenty vyplněny, aby zavoláním **CheckDataPersistence()** mohlo dojít ke kontrole dat. Obsahuje dodatečné **ViewMatchers** a **ViewActions** které jsem si napsal pro účely testování.
- **UpdateProfileTest**- Provede změnu profilu společnosti a následně otestuje, zda byla změna korektně uložena. Změnu provádí přes API, stejně jako v reálném nasazení.
- **CompanyJobsActivityTest**- Tento test netestuje perzistenci dat, ale správnou funkci seznamu s brigádami. Test automaticky prokliká spodní navigaci seznamu brigád a tím ověří že jsou ovládací prvky přítomné, viditelné lze na ně kliknout.
- **CreateJobTest**- Otestuje workflow vytvoření nové brigády od začátku až do konce. Po každé vyplněné obrazovce provede rotaci obrazovky a následnou kontrolu dat.
- **ComplexUseCaseTest**- Tento test otestuje zobrazení seznamu brigád, zobrazení detailu brigádníkova profilu, přijetí brigádníka a zamítnutí brigádníka. Před tímto testem je nutné mít alespoň alespoň 2 brigády a u druhé z nich alespoň 2 uchazeče. Test klikne na druhou brigádu, ověří že se zobrazil její detail. Poté přijme prvního brigádníka a ověří že došlo k přijetí. Následně zamítne druhého a ověří že došlo k zamítnutí.

Diskuze a práce do budoucna

Hlavní nedostatky aplikace souvisí s funkcionalitou, které poskytuje API. V době psaní této práce je API stále ve vývoji a následující funkcionality nedovede obsloužit:

- Založení nové brigády - Neexistuje endpoint kam zaslat data po dokončení vytváření brigády. Moje aplikace vytvoří entitu která by se zaslala na endpoint, ale po vytvoření brigády nic neprovede. Po vytvoření endpointu bude snadné funkcionalitu dodělat.
- Zrušení brigády - Neexistuje endpoint pro zrušení brigády. V UI je vše připraveno.
- Zjištění zda brigáda byla zrušena - V endpointech pro získání seznamu brigád i detailu brigády chybí informace o tom že brigáda byla zrušena (Na detailu je klíč `isDisabled`, ten je ale vždy `false`).
- Obsluha pracovníků - API podporuje nedostatečné množství statusů pro pracovníky. Nelze tak rozlišit nejprve přijmutého a následně zamítnutého brigádníka od přijatého.
- Práce s termíny a kalendářem - API u brigád nevrací jejich termíny, pouze datum ukončení. Je potřeba domluvit s autory API formu jakou budou termíny vraceny, pak bude implementováno v mé aplikaci.

Má aplikace se stále nachází ve stádiu prototypu, proto některé funkcionality z webové verze nejsou dostupné, ikdyž můj návrh s nimi počítá. Tyto funkcionality nejsou pro základní funkci klíčové, ale aby moje aplikace mohla svůj webový protějšek plnohodnotně zastoupit, je třeba je v budoucnu implementovat. Jsou to:

6. DISKUZE A PRÁCE DO BUDUCNA

1. Podpora zpráv
2. Reakce na push notifikace zasílané serverem (tiché i viditelné uživateli)
3. Sekce nastavení - Notifikace a další funkcionality aplikace
4. Sdílení profilu brigádníka přes systémový Share button
5. Zobrazování obrázků - Aby správně fungovalo zobrazování obrázků, je nutné aby server který obrázky poskytuje běžel fyzicky na jiném stroji než Pharo server. To je v testovacím prostředí obtížné zajistit, nicméně v produkční verzi by s tím neměl být problém.

Závěr

Cílem mé práce bylo vytvořit wireframy a grafický návrh aplikace Daywork.cz pro mobilní zařízení a dále implementovat modul pro firmy s využitím architektury vytvořené kolegou.

První část úkolu jsem splnil nákresem wireframů aplikace v nástroji Pencil a následnou konzultací těchto wireframů se zadavatelem. Při návrhu jsem se držel obecných doporučení pro operační systém Android. Konzultace i následné otestování aplikace potvrdilo že návrh je použitelný a splňuje účel.

Druhou část úkolu jsem splnil implementací modulu pro firmy v jazyce Java za použití vývojového prostředí Android Studio. Implementace se zaměřila na usecasey zmíněné v zadání, které jsem po implementaci otestoval.

Požadavky ze zadání jsou v mojí aplikaci uživateli dostupné způsobem zmíněným v kapitole 4.3

Zadání jsem splnil i když aplikace není zcela nasazení schopná kvůli nefunkčnosti některých částí API a absenci funkcionalit s nižší prioritou.

Literatura

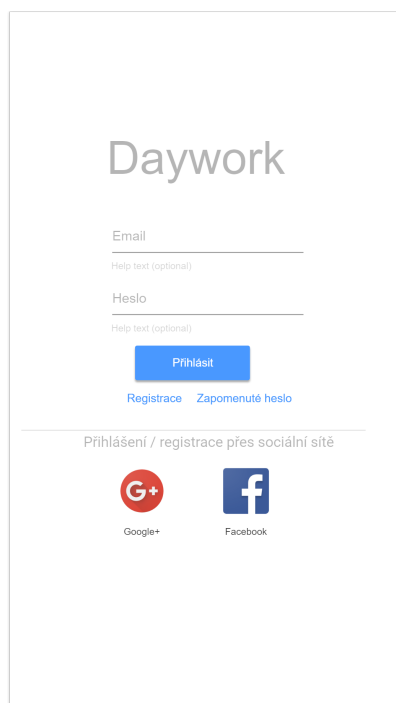
- [1] Google: *Material Design*. [cit. 2018-31-05]. Dostupné z: <https://material.io/design/>
- [2] Google: *Material Design*. [cit. 2018-31-05]. Dostupné z: <https://material.io/design/usability/accessibility.html#layout-typography>
- [3] Google: *Toolbar*. [cit. 2018-31-05]. Dostupné z: <https://material.io/develop/android/components/app-bar-layout/>
- [4] Google: *Bottom Navigation*. [cit. 2018-31-05]. Dostupné z: <https://material.io/design/components/bottom-navigation.html>
- [5] *Inkscape*. [cit. 2018-31-05]. Dostupné z: <https://inkscape.org/en/>
- [6] *Justinmind*. [cit. 2018-31-05]. Dostupné z: <https://www.justinmind.com/>
- [7] *Pencil*. [cit. 2018-31-05]. Dostupné z: <https://pencil.evolus.vn/>
- [8] Google: *Android Studio*. [cit. 2018-31-05]. Dostupné z: <https://developer.android.com/studio/>
- [9] Maynard-Reid, M.: MVVM on Android with the Architecture Components. [cit. 2018-31-05]. Dostupné z: <https://medium.com/@margaretmz/exploring-the-android-architecture-components-117515acfa8>
- [10] *Android Architecture Components*. [cit. 2018-31-05]. Dostupné z: <https://developer.android.com/topic/libraries/architecture/>
- [11] Malý, M.: *REST: architektura pro webové API*. [cit. 2018-31-05]. Dostupné z: <https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/>

LITERATURA

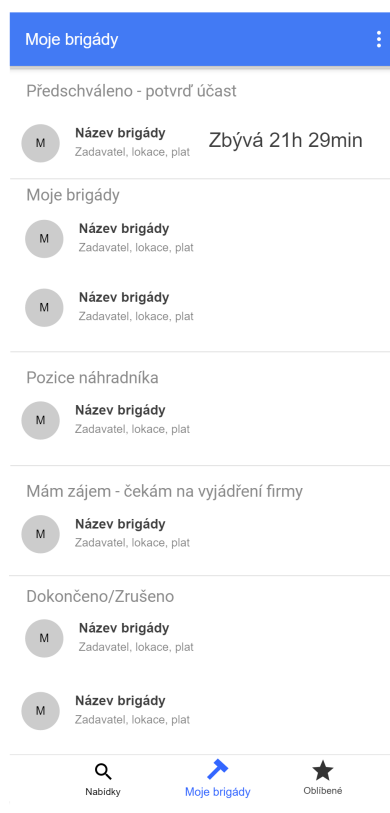
- [12] *Espresso Recipes*. [cit. 2018-31-05]. Dostupné z: <https://developer.android.com/training/testing/espresso/recipes>
- [13] Murphy, M. L.: *Android 2*. Computer press a.s, 2011.

Obrazovky brigádníka

A. OBRAZOVKY BRIGÁDNÍKA



Obrázek A.1: Přihlášení



Obrázek A.2: Zobrazení brigád

← Registrace 1/4

Email

Email - znovu

Heslo

Heslo - znovu

Jméno

Příjmení

Žena
 Muž

Souhlas s [podmínkami](#)

Další

Obrázek A.3: Registrace - osobní údaje

← Registrace 2/4

Vzdělání

Nechci uvést
 Střední
 Vysoké

Studuji Ukončeno

Řidičský průkaz skupiny B

Nechci uvést
 Ano
 Ne

Jazyky

✕ Angličtina ★ ★ ★

★ ★ ★ **+**

IT schopnosti

✕ Word ★ ★ ★

✕ Excel ★ ★ ★ **+**

Další

Obrázek A.4: Registrace - znalosti

A. OBRAZOVKY BRIGÁDNÍKA

← Registrace 3/4

Pár slov o tobě

Long body text - Minantia non modo
formaeque in meis acervo formaeque
gravitate erat indigestaque habentia fixo
mutatas aliud orbis retinebat qui non alta

Pracovní zkušenosti

- ✕ Číšník Rok
- ✕ Taxikář 3 roky

+ (add button)

Další (next button)

Obrázek A.5: Registrace - zkušenosti a představení

← Registrace 4/4

Věk a konfekční velikost

Věk **nad 18**

Výška **150 až 159 cm**

Konfekční velikost **XXL**

Profilová fotografie



Fotogalerie

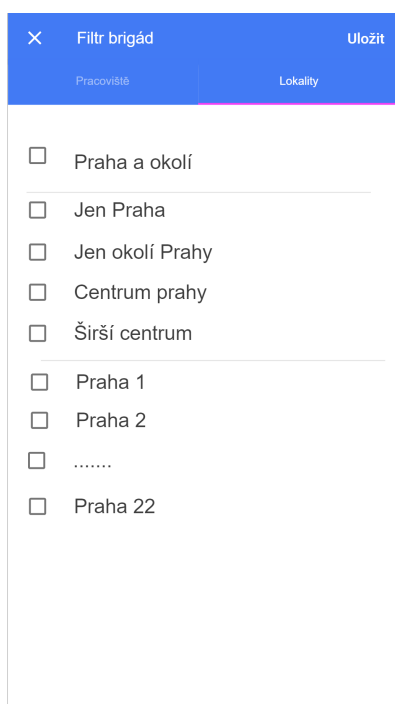
+ (add button)

Videogalerie

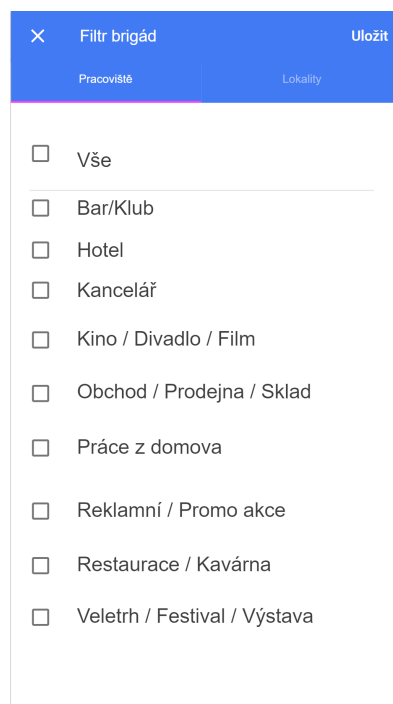
+ (add button)

Dokončit (finish button)

Obrázek A.6: Registrace - konfekční velikost, věk, média

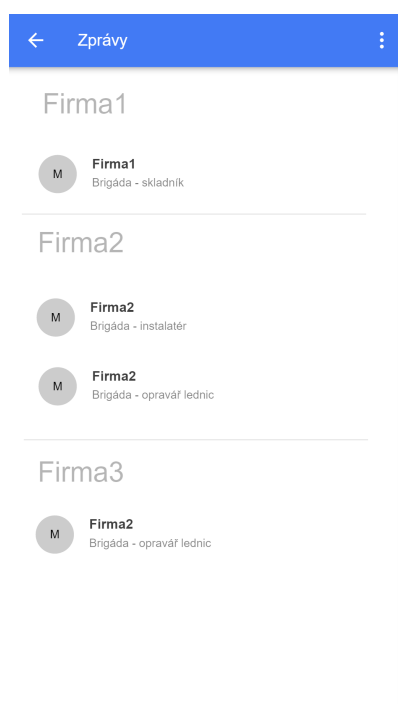


Obrázek A.7: Filtr lokací

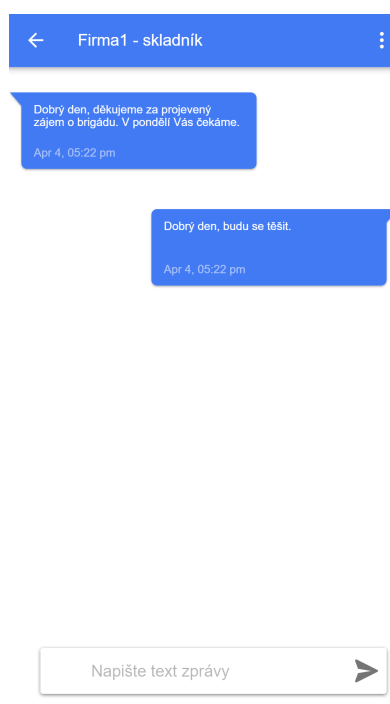


Obrázek A.8: Filtr pracovišť

A. OBRAZOVKY BRIGÁDNÍKA



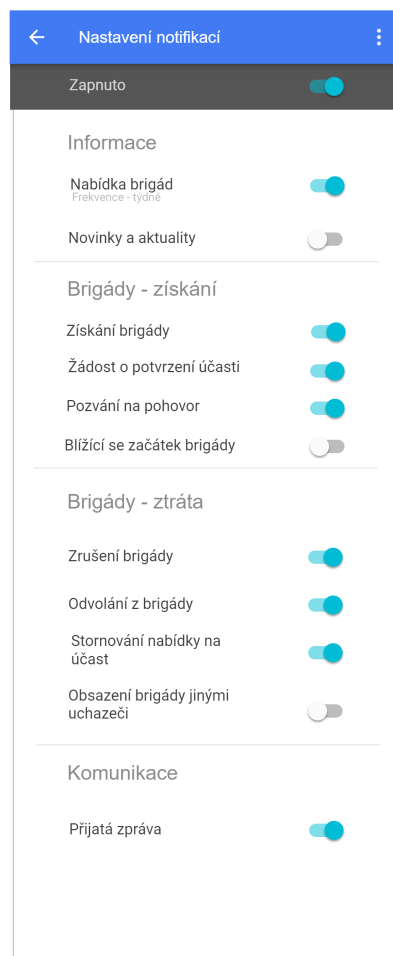
Obrázek A.9: Seznam konverzací



Obrázek A.10: Detail konverzace

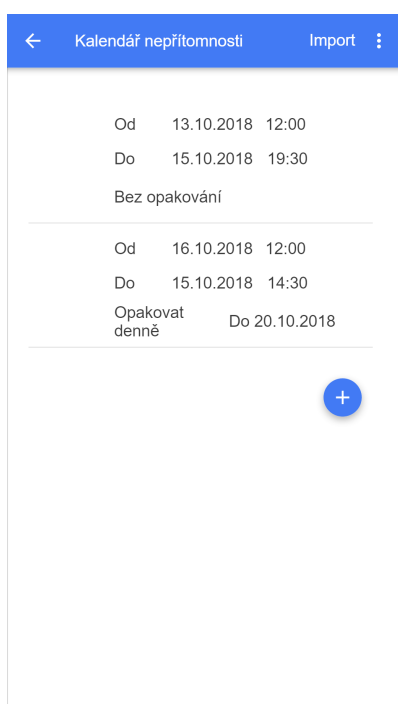


Obrázek A.11: Nastavení emailů

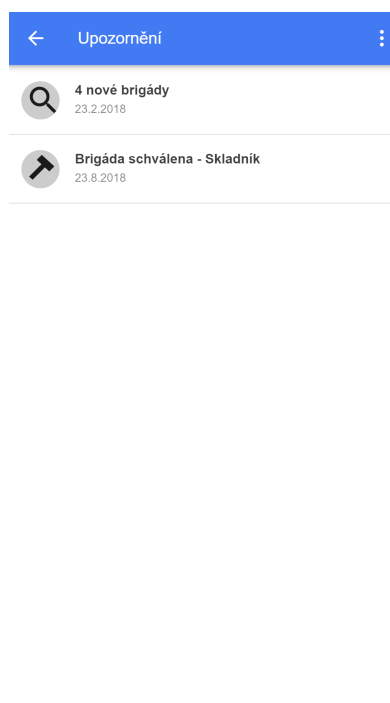


Obrázek A.12: Nastavení notifikací

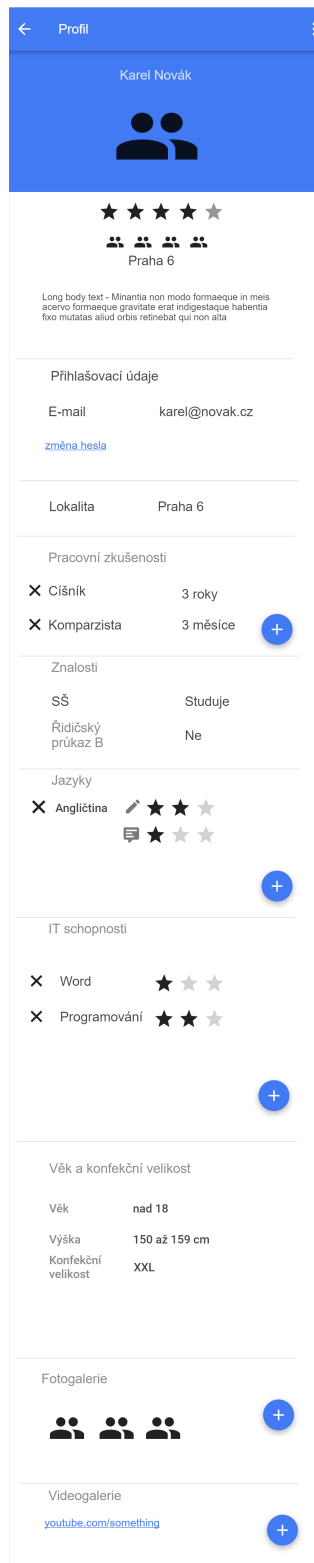
A. OBRAZOVKY BRIGÁDNÍKA



Obrázek A.13: Kalendář nepřítomnosti



Obrázek A.14: Notifikace



Obrázek A.15: Profil pracovníka

Obrazovky firmy

B. OBRAZOVKY FIRMY

← Registrace 1/3

IČ

Název firmy

Zaměstnavatel
 Personální agentura

Sídlo společnosti

Ulice

Obec

PSČ

Česká republika
 Slovenská republika

Kontaktní adresa

Stejná jako adresa sídla

Ulice

Obec

PSČ

Česká republika
 Slovenská republika

[Další](#)

Obrázek B.1: Registrace - firmní údaje

← Registrace 2/3

Kontaktní osoba

Oslovení

Jméno

Příjmení

Telefon

Email

[Další](#)

Obrázek B.2: Registrace - kontaktní osoba

← Registrace 3/3

Přihlašovací údaje

Email bude stejný jako u kontaktní osoby
firma2firma.cz

Heslo _____

Heslo znovu _____

Souhlas s [podmínkami](#)

Dokončit

Obrázek B.3: Registrace - přihlašovací údaje

Moje aktivní brigády

M Dalsí brigáda
lokace, plat

Prehled Přijetí Uchazeči

+

Aktivní Ukončené

Obrázek B.4: Správa brigád

B. OBRAZOVKY FIRMY

Vložení brigády 1/6

Brigáda

Název pozice

Výběr pracoviště

Počet brigádníků

Nezáleží	0
Žena	0
Muž	0

Dalsí

Vložení brigády 2/6

Vzdělání

Nezáleží

Střední

Vysoké

Student Absolvent

Řidičský průkaz skupiny B

Nezáleží

Vyžadován

Jazyky

Angličtina

IT schopnosti

Word

Excel

Praxe

Číšník 3 roky

Věk

Nezáleží

Výška

Od Neuvedeno

Do Neuvedeno

Konfekční velikost

Od Neuvedeno

Do Neuvedeno

Dalsí

Obrázek B.5: Nová brigáda -
název, pracoviště, brigádníci

Obrázek B.6: Nová brigáda
- Schopnosti a zkušenosti
brigádníka

← Vložení brigády 3/6

Typ umístění

Má brigáda přesnou adresu?

Ano - vyplnit adresu

Ne - určím oblast

Adresa

Stejná jako adresa sídla

Ulice _____

Obec _____

PSČ _____

Česká republika

Slovenská republika

[Další](#)

Obrázek B.7: Nová brigáda - umístění

← Vložení brigády 4/6

Termín brigády

Má brigáda přesný termín?

Ano

Ne

[Další](#)

Obrázek B.8: Nová brigáda - termín

B. OBRAZOVKY FIRMY

← Vložení brigády 5/6

Mzda

Jakým způsobem bude brigáda odměněna?

Hodinově

Měsíčně

Jednorázově

Jinak (doplnit do sekce "nabízíme")

Částka _____
V případě že je známa

Popis

Zde vložte popis brigády _____

Požadujeme

Zde upřesněte požadavky na brigádníka _____

Nabízíme

Zde popište co nabízíte _____

[Další](#)

← Vložení brigády 6/6

Obrázek

Logo firmy

Nahrát

Není

Fotky

[+](#)

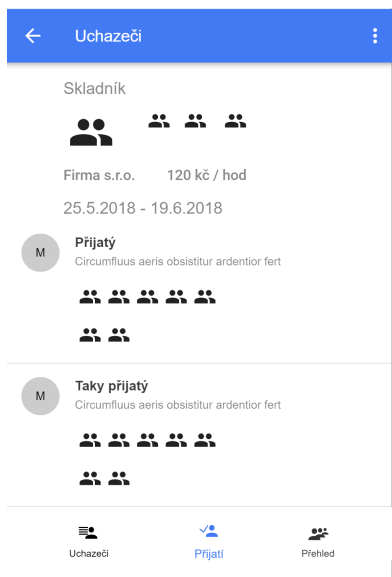
Odkazy na Youtube

[+](#)

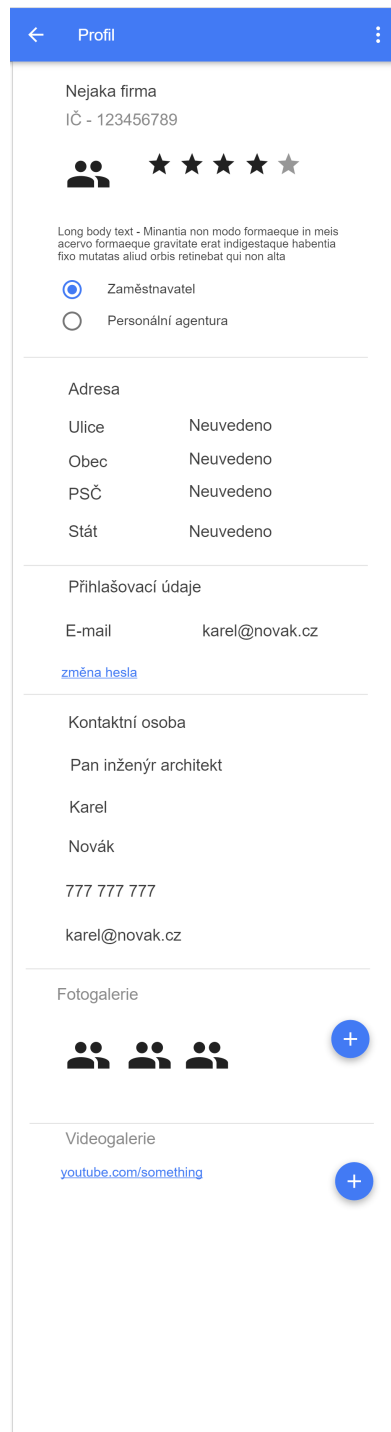
[Náhled](#)

Obrázek B.9: Nová brigáda - mzda a popis

Obrázek B.10: Nová brigáda - média



Obrázek B.11: Pracovníci u dané brigády



Obrázek B.12: Profil společnosti

Slovníček pojmů a zkratk

UI User interface - uživatelské rozhraní

XML Extensible markup language - značkovací jazyk pro popis sémantiky

usecase Konkrétní případ užití aplikace

wireframe Model obrazovky uživatelského rozhraní aplikace

dependency injection Způsob jak sestavit aplikaci aniž by jednotlivé komponenty vytvářely samy komponenty na kterých závisí

dao Data access object. Zprostředkovává zbytku aplikace operace s databází

json Javascript object notation. Textový způsob reprezentace dat využívající strukturu objektů key:val, ... kde key je řetězec a val je buď řetězec nebo další objekt.

SharedPreferences Třída spravující dvojice klíč:hodnota, které se ukládají na filesystem zařízení. Framework poskytuje metody k snadné obsluze SharedPreferences, takže není nutné se zabývat prací s filesystemem při ukládání a načítání. Používá se často pro perzistenci nastavení aplikací.

MVVM Model View ViewModel. Typ architektury aplikace

REST API Api - application programming interface - rozhraní umožňující jedné aplikaci získat data z jiné bez znalosti její vnitřní implementace.
Rest API - bezstavové api

api endpoint Jasně definovaná adresa, která reprezentuje jednu funkcionálnítu api

Navigation Drawer Prvek UI nacházející se typicky vlevo nahoře. Typicky má podobu 3 vodorovných čar pod sebou (odtud se mu přezdívá hamburger menu). Kliknutím na tento prvek vyjede z levého okraje obrazovky kontejner s možnostmi.

RecyclerView Slouží pro zobrazení skrolovatelného seznamu položek na platformě Android

URI Uniform resource identifier - link jednoznačně určující umístění dat

toolbar Horní lišta s odkazem na menu vpravo a popisem prováděné akce. Jedná se o standardní prvek Android UI[3]

Overflow menu Menu přístupné přes 3 tečky typicky umístěné vpravo nahoře. Bývá součástí toolbaru

Obsah přiloženého flashdisku

readme.txt.....	stručný popis obsahu disku
install.....	nástroje nutné pro kompilaci a otestování aplikace
daywork-wireframes	adresář s wireframy
├── worker	wireframy pracovníka
│ ├── pages	Obrázky ve formátu png
│ ├── Resources	Složka s css
│ ├── Daywork_wireframes.epgz ..	Editovatelný soubor programu Pencil
│ └── index.html	Spustí prototyp jako webovou stránku
├── bussiness.....	wireframy společnosti
│ ├── pages	Obrázky ve formátu png
│ ├── Resources	Složka s css
│ ├── Daywork_wireframes.epgz ..	Editovatelný soubor programu Pencil
│ └── index.html	Spustí prototyp jako webovou stránku
daywork-android-project.....	Obsahuje projekt pro Android Studio
├── app	Obsahuje aplikaci
│ └── src	Zdrojové kódy aplikace
└── gradle	
thesis-source	zdrojové kódy textu práce
├── Novotny_Krystof_2018.tex.....	text práce v L ^A T _E Xu
├── images	
│ ├── worker	
│ └── company	
thesis-compiled.....	text práce
└── Novotny_Krystof_2018.pdf	text práce ve formátu PDF