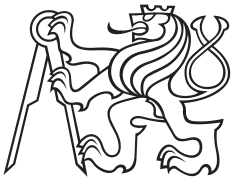
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
katedra počítačů

Kompenzace diabetes mellitus pomocí mobilních technologií

Bc. Tomáš Musílek
Otevřená informatika, Softwarové inženýrství
musilto5@fel.cvut.cz

květen 2017
Vedoucí práce: Ing. Daniel Novák, Ph.D.

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra počítačů

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Musílek Tomáš

Studijní program: Otevřená informatika
Obor: Softwarové inženýrství

Název tématu: Kompenzace diabetes mellitus pomocí mobilních technologií

Pokyny pro vypracování:

1. Seznamte se s problematikou self-managementu diabetes mellitus a edukativních nástrojů v oblasti mhealth. Provedtě rešerši nejrelevantnějších předchozích publikovaných metod se stejným cílem.
2. Navrhněte mobilní aplikaci (client-server), pro inspiraci vycházejte ze systému mobiab.cz.
3. Hru naimplementujte v programovém prostředí Android.
4. Aplikaci otestujte na vzorku 5 pacientů. Porovnejte účinnost edukace vašeho přístupu s publikovanými metodami.

Seznam odborné literatury:

1) Cafazzo, J. A., Casselman, M., Hamming, N, Design of an mHealth app for the self-management of adolescent type 1 diabetes: apilot study, Journal of Internet Medical Research, January 4, 2012.
2) Tatara N, Årsand E, Skrøvseth SO, Hartvigsen G. Long-Term Engagement With a Mobile Self-Management System for People With Type 2 Diabetes, JMIR Mhealth Uhealth 2013;1(1):e1
3) Arsand E, Demiris G. User-centered methods for designing patient-centric self-help tools. Inform Health Soc Care 2008 Sep;33(3):158-169
4) Robroek SJ, Lindeboom DE, Burdorf A. Initial and sustained participation in an internet-delivered long-term worksite health promotion program on physical activity and nutrition. J Med Internet Res 2012;14(2):e43
5) Lyles CR, Harris LT, Le T, Flowers J, Tufano J, Britt D, et al. Qualitative evaluation of a mobile phone and web-based collaborative care intervention for patients with type 2 diabetes. Diabetes Technol Ther 2011 May;13(5):563-569
6) Franklin VL, Greene A, Waller A, Greene SA, Pagliari C. Patients' engagement with "Sweet Talk" - a text messaging support system for young people with diabetes. J Med Internet Res 2008;10(2):e20
7) Katz R, Mesfin T, Barr K. Lessons from a community-based mHealth diabetes self-management program: "it's not just about the cell phone". J Health Commun 2012;17 Suppl 1:67-72

Vedoucí: doc. Ing. Daniel Novák, Ph.D.
Platnost zadání do konce letního semestru 2017/2018

L.S.

prof. Dr. Michal Pěchouček, MSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 20.2.2017

Poděkování / Prohlášení

Chtěl bych poděkovat svojí přítelkyni za její velkou trpělivost a jazykové korektury.

Další dík patří Ing. Václavu Burdovi a Ing. Danielu Novákovi Ph.D. za vedení práce a cenné rady.

Rád bych také poděkoval participantům, kteří se zúčastnili testování, za jejich ochotu a disciplinovanost.

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

V Praze dne 10. 5. 2017

.....

Abstrakt / Abstract

Tato práce seznamuje čtenáře s dostupnými mobilními aplikacemi, které se zabývají dietou nebo diabetem. Práce dále popisuje stav projektu Mobiab. Na těchto základech je poté navržena nová mobilní aplikace Dia Dieta. Tato aplikace je následně otestována na vzorku participantů.

Klíčová slova: inzulín, diabetes, dieta, mobilní technologie

This paper introduces mobile applications focused on diabetes and diet. Furthermore, it shows the state of Mobiab project. Then the new mobile app Dia Dieta is developed. The final part of the thesis deals with testing of the new app by participants.

Keywords: insulin, diabetes, diet, mobile technology

Title translation: Diabetes mellitus compensation by the mobile technology

Obsah /

Tabulky / Obrázky

Kapitola 1
Úvod

1.1 Motivace

Mobiab je mobilní aplikace pro sledování kalorického příjmu a výdeje, glykémie a krevního tlaku. Aplikace má pomáhat jednak při léčbě a prevenci Diabetes mellitus a jednak s udržováním zdravého životního stylu. [1]

Hlavním cílem mé práce je navázat na úspěšnou aplikaci Mobiab [2] [3], vytvořit novou aplikaci, která si bere to dobré z Mobiabu, popřípadě dalších aplikací zmíněných v podkapitole č. 4 a která vše převede do jednoduchého srozumitelného uživatelského prostředí s modernějším vzhledem. Rovněž se v nové aplikaci, jež vytvořím, pokusím o co největší komfort pro uživatele. Od těchto kroků očekávám, že přinesou ještě větší oblibu u uživatelů, než má aplikace Mobiab.

Výsledná aplikace bude určena pro cílovou skupinu diabetiků a zároveň skupinu lidí, která si hlídá své přijmy potravy, drží nějaké diety. Podobně tomu je u aplikace Mobiab. Hlavním přínosem aplikace pro uživatele bude možnost efektivně a rychle přidávat zkonzumovaná jídla, vést si záznamy o glykémii, dávkách inzulinu a pohybu, toto vše snadno a rychle prohlížet v různých typech přehledů a grafů.

1.2 Struktura práce

Práce začíná stručným úvodem do základní problematiky diabetu a stravování (více viz. kapitola č. 2). Další kapitola popisuje stav projektu Mobiab a projektů, které jsou k němu přidružené (více viz. kapitola č. 3). Ve třetí kapitole jsou rozebrány mobilní aplikace zabývající se diabetem a nebo dietou. Jsou zde zdůrazněny jak jejich dobré, tak špatně vlastnosti (více viz. kapitola č. 4). Na základě předchozích kapitol jsou sestaveny požadavky na novou mobilní aplikaci, která vznikne jako výsledek mé diplomové práce (více viz. kapitola č. 5). Dále se věnuji výběru klíčových technologií, na kterých bude mobilní aplikace vystavěna. Na konci kapitoly jsou doplněny nefunkční požadavky pro novou aplikaci (více viz. kapitola č. 6). Kapitola Návrh aplikace se poté věnuje prvotnímu návrhu vzhledu a funkcionality obrazovek aplikace (více viz. kapitola č. 7). V kapitole Implementace je detailně rozebrán výběr architektonického vzoru aplikace a podpůrných knihoven použitých při implementaci (více viz. kapitola č. 8). Následuje kapitola věnující se představení hotových stěžejních obrazovek v nové aplikaci (více viz. kapitola č. 9). V kapitole č. 10 je popsán návrh uživatelských testů, jejich samotné provedení a nakonec vyhodnocení. Závěrečná kapitola shrnuje práci a představuje návrhy na vylepšení do budoucna.

Kapitola 2
Dieta a diabetes mellitus

Moje práce bude pracovat s pojmem dieta, jenž definuji jako správné stravování. Dalším pojmem s kterým bude práce často operovat je diabetes.

2.1 O dietě

Příjmem kalorií a správným složením jídelníčku se zabývá ve své bakalářské práci Václav Burda [2]. Ve své práci představuje vzorce, pro výpočet bazálního metabolismu (BMR). Bazální metabolismus zahrnuje množství vydané energie v klidovém stavu nutné pouze pro fungování životně důležitých orgánů. Hodnotu BMR lze spočítat podle Harris-Benedictova vzorce z roku 1919, který je považován za standard po několik desetiletí a je stále nejpoužívanějším vzorcem pro výpočet BMR. Vstupními parametry tohoto vzorce jsou pohlaví, hmotnost „m“ v kilogramech, výška „h“ v centimetrech a věk „v“ v letech. [2]

Václav Burda dále uvádí vzorce pro výpočet bazálního metabolismu pro muže a ženu.

Vzorec pro výpočet BMR pro ženu:

```
BMR = 655,0955 + (9,5634 * m) + (1.8496 * h) - (4,6756 * v)
```

Vzorec pro výpočet BMR pro muže:

```
BMR = 66,476 + (13,7516 * m) + (5,0033 * h) - (6,755 * v)
```

Pro výpočet výdeje energie za den vynásobíme BMR příslušnou konstantou z následující tabulky, vyjde nám celková hodnota kalorického výdeje za den v kilokaloriích. Pro převod na kilojouly je třeba výslednou hodnotu vynásobit konstantou 4.184. Platí, že energie získaná z potravy by se měla pohybovat okolo spočtené hodnoty výdeje energie.

Koeficienty aktivity	Aktivita	Popis
1,2	Sedavá	Málo nebo žádný pohyb, práce v kanceláři
1,375	Lehká	Málo pohybu nebo sportování 1-3 dny v týdnu
1,55	Střední	Středně pohybu nebo sportování 3-5 dní v týdnu
1,725	Těžka	Hodně pohybu nebo sportování 6-7 dní v týdnu
1,9	Extrémní	Každodenní náročný sport nebo těžká fyzická práce

Tabulka 2.1. Koeficienty pro výpočet spotřebované energie z BMR, zdroj: [2].

2.1.1 Skladba potravin během dne

Aby byla v těle zachována rovnováha platí, že jednotlivé složky potravy (sacharidy, bílkoviny, tuky) by měly tvořit určitou část z přijmuté potravy. 55 - 60% potravy by mělo být složeno ze sacharidů, 25-30% by mělo být tvořeno tuky, 15% by mělo být tvořeno bílkovinami. Jeden gram sacharidů obsahuje přibližně 17kJ, jeden gram bílkovin také přibližně 17kJ, jeden gram tuků pak 37kJ. Máme-li tedy spočtený svůj denní výdej energie, můžeme lehce spočítat, kolik bílkovin, sacharidů a tuků bychom měli za den zkonzumovat. [2]

2.2 O diabetu

Co je diabetes, co způsobuje, jaké jsou jeho typy, jak se projevuje, jak se proti němu bojuje, jsem řešil v rámci své bakalářské práce. Nebudu zde tedy znovu psát obdobný text a s dovolením se zde odkážu na kapitolu 2 v mé bakalářské práci [4], popřípadě na edukační modul, který je součástí aplikace Mobiab. Tento modul je rovněž mým dílem. Modul obsahuje celkem vyčerpávající informace o diabetu.

Kapitola 3
Popis projektu Mobiab

V této kapitole rozeberu projekt Mobiab, ukážu jeho minulost, scope, který obsahuje současný stav projektu. Nakonec ukážu, kde bych na tento projekt chtěl navázat a čeho bych svým snažením chtěl dosáhnout.

3.1 Mobiab počátek

Mobiab začal jako bakalářská práce Václava Burdy v roce 2012. Jednalo se o webový portál a k tomu přidruženou mobilní aplikaci pro platformu Android. Hlavním tématem práce bylo sledování příjmu potravy pomocí mobilní aplikace, dalším podtématem této práce pak bylo kompenzování diabetu. Mobilní aplikace komunikovala s api webového portálu, umožňovala uživateli sledovat svůj denní příjem energie, sacharidů, proteinů a tuků, stanovovala denní limity těchto položek. Uživatel mohl pomocí aplikace vybírat jídla z databáze jídel a přidávat si je mezi zkonzumovaná jídla popřípadě oblíbená jídla. Zkonzumovaná jídla si poté uživatel mohl prohlížet, popřípadě upravovat. Nová jídla šla vytvářet i upravovat.

Webový portál poskytoval podobnou funkcinalitu jako aplikace. [2] Kromě této funkcionality Mobiab obsahoval další moduly a to konkrétně sledování kvality spánku pomocí chytrého telefonu [5] a vyhodnocování stavu maniodepresivních pacientů [6].

3.2 Mobiab pokračování

V roce 2014 autor Mobiabu, Václav Burda, svůj projekt dále rozvinul do podoby diplomové práce. [3] Název práce tentokrát zněl Kompenzace diabetes mellitus pomocí mobilních technologií. Projekt se od sledování příjmu potravy přesunul více ke kompenzaci diabetu. V této fázi vývoje byla mobilní aplikace rozdělena do modulů, některé moduly mohly být v případě potřeby doinstalovány.

Mobilní aplikace umožňovala sledovat kalorický příjem, vykonané aktivity během dne a průběh hladiny glykémie. Aplikace dovedla připojit Bluetooth glukometru a Bluetooth vah pro snadnější a pohodlnější zadávání naměřených hodnot. Díky návrhu klient-server mohl mít lékař k dispozici vždy aktuální údaje o pacientovi a mohl na ně reagovat zasláním zprávy svému pacientovi. [3]

Funkcionalita aplikace byla obohacena o různé další moduly:

3.2.1 Gamifikační modul

Jedná se o modul, který měl za úkol uživatele odměňovat za svědomité vyplňování formulářů aplikace. Za vyplněný záznam (např. konzumaci) uživatel dostal odměnu ve formě bodů, tato odměna byla tím větší, čím častěji uživatel aplikaci používal. Nasbírané body poté bylo možné vyměnit za doplňky pro Avatara, popřípadě za fyzické odměny, které byly součástí e-shopu aplikace. [7]

3.2.2 Upomínkový modul

Modul umožňoval uživateli nastavovat si upomínky, které mu později vyskakovaly jako notifikace v telefonu. Tyto notifikace uživateli připomínaly, že by si měl do aplikace zadat např. večeři.

3.2.3 Sociální modul

Tento modul umožňoval sdílet některé zadané hodnoty na sociální síť Facebook. Uživatelé se s pomocí aplikace přidali do skupiny na Facebooku, kterou zaštiťoval projekt Mobiab, v této skupině poté mohli vzájemně zveřejňovat své dosažené výsledky. [8]

3.2.4 Motivační widget na plochu

Uživateli bylo umožněno přidat si na plochu tzv. avatara, který ho měl zosobňovat. Avatar reagoval na to jak uživatel do aplikace zadával, nebo nezadával údaje. K danému Avatarovi uživatel mohl kupovat za nasbírané body různé kusy oblečení. [9]

3.2.5 Bluetooth komunikace

Autor práce do aplikace zakomponoval moduly pro komunikaci s Bluetooth vahami, s Bluetooth glukometrem a s Bluetooth tlakoměrem. Komunikace s Bluetooth vahami je problematická, nepodařilo se totiž na trhu sehnat vhodné váhy pro integraci do aplikace, byl zvolen prototyp vah, který se na trhu běžně neprodává.

3.3 Rozvoj po odevzdání diplomové práce Václava Burdy

Po odevzdání diplomové práce autor zahájil postgraduální studium a práci nadále udržoval a rozvíjel, do projektu přispívali i další lidé.

3.3.1 Edukační modul

V roce 2014 jsem se do projektu dostal já. V rámci své bakalářské práce jsem vyvinul edukační modul pro aplikaci Mobiab. Práci jsem obhájil v červnu 2015. Edukační modul se skládal z obrazovek popisujících onemocnění diabetes mellitus, vysvětloval co diabetes je, jaké jsou druhy diabetu, co diabetes způsobuje, jak se z diabetem dá bojovat atd. Vytvořil jsem základního průvodce tímto závažným onemocněním.

Další část mé práce se věnovala simulaci působení inzulinu na hladinu glykémie v lidském těle. K aplikaci jsem vyvinul server psaný v Java EE. Tento server pomáhal s generováním simulačních grafů zobrazovaných v aplikaci. [4]

3.3.2 Nový webový portál

Od léta roku 2015 do projektu začal přispívat Václav Dobeš. Ten si vzal za úkol přepsat serverovou část aplikace a vytvořit zcela nového webového klienta. Důvody k přepsání serveru z php na Java EE byly následující:

- Vyměnit procedurální php skripty za objektový návrh s jasně danou dokumentovanou architekturou.
- Umožnit škálování aplikace.

Václav Dobeš při své implementaci vycházel z původní databáze Mobiabu, tuto databázi upravil tak, aby názvy lokalizované do jednotlivých jazyků mohly být součástí jedné databáze. Např. databázová tabulka Food obsahuje pouze číselné údaje o jídle, když chceme zjistit název daného jídla, musíme najít příslušnou tabulku Food_lang, která obsahuje jednak klíč daného jazyka, textovou hodnotu lokalizovaného jména jídla a cizí klíč, který ukazuje na záznam v tabulce Food. [10]

Autor nad danou upravenou databází vystavěl REST API, které umožňuje jednotný přístup ke všem zdrojům (datům z databáze). Autor poté napsal webového klienta, kterým se na toto API připojoval. Webový klient má podobnou funkcionalitu jako jádro aplikace Mobiab. Neobsahuje ovšem tolik modulů.

3.4 Počátek mé diplomové práce

V létě roku 2016 jsem se opět vrátil do projektu Mobiab a začal jsem rozmýšlet, jak nasměrovat jeho pokračování. Původní plán byl přepsat mobilní aplikaci Mobiab tak, aby uměla komunikovat s novým serverem psaným v Java EE. Při této příležitosti jsem plánoval aplikaci upravit tak, aby fungovala offline. Aplikace Mobiab je závislá na internetovém připojení, bez připojení nejde prakticky používat. Tato vlastnost aplikace pravděpodobně odrazuje velké procento uživatelů od jejího používání.

Dalším nedostatkem Mobiabu je nutnost přihlášení se, nepřihlášený uživatel se do aplikace nedostane, nemůže si tedy vůbec vyzkoušet kvůli čemu by měl podstoupit proces přihlášení. Při testování se zároveň jako nedostatek ukázal old-school design aplikace.

Za další překážku pro širší využití aplikace jsem považoval místy složitý návrh aplikace. Na jednu z nejpoužívanějších obrazovek s přidáním konzumace se uživatel dostane přes sérii obrazovek s vyhledáváním jídla. Samotná obrazovka pak obsahuje hromadu editovatelných položek, tlačítek a spinnerů, které jsou poskládány vedle sebe, aby se tam dobře vešly. Tlačítek je tam skutečně mnoho jen na obrazovce s přidáním konzumace jsou tři.

Obrazovka s přidáním konzumace má od sebe oddělené zobrazování a zadávání nových hodnot, to už se ale nedá říct o obrazovce glykémie. Ta tyto dvě činnosti kombinuje. Na jedné obrazovce lze zadat nové měření, zároveň se tam zobrazují měření zaznamenaná v určitý den, je tam možnost přepnout na jiný den, zobrazit si graf s naměřenými hodnotami glykémie, popřípadě editovat starší měření. Tento návrh je modulární, jednotlivé moduly jsou od sebe odděleny, neexistuje mezi nimi žádné propojení, není zde např. obrazovka, jež by ukazovala souhrn zadaných záznamů pro jeden den nebo obrazovka, jež by sloužila jako rozcestník pro zadávání záznamů. Aplikace Mobiab je členěná po modulech a ne po funkcích.

Po dohodě s autorem Mobiabu jsme se dohodli na tom, že nebudu přepisovat Mobiab, který je v této chvíli velký moloch. Implementovat do něj změny, které opravují jeho nedostatky, by znamenalo v podstatě přepsat celé jádro aplikace, navigaci mezi obrazovkami a nakonec i vzhled aplikace. V aplikaci by nezůstal kámen na kameni. Vytvořím novou menší aplikaci, kterou pojmenuji Dia Dieta a která bude obsahovat pouze základní funkcionalitu Mobiabu a bude se snažit o následující:

- Aplikace se bude snažit o jednoduchý user interface.
- Aplikace se bude snažit o design, který sleduje doporučení pro danou platformu (např. Android material design).
- Aplikaci bude možné používat i offline, v offline módu nebude pouze docházet k synchronizaci dat na server.
- V aplikaci bude odděleno zadávání nových hodnot, zobrazování přehledů a zobrazování grafů. Aplikace bude dělená po funkcích.
- Aplikace se bude napojovat na nový server psaný v Java EE.

Kapitola 4
Výzkum projektů podobných Mobiabu

V této kapitole provedu analýzu mobilních aplikací zabývajících se diabetem a nebo dietou.

4.1 Aplikace YAZIO

Počet stažení na Google Play je pro tuto aplikaci řádově 1 milion uživatelů. Aplikace byla naposledy aktualizována 24. dubna 2017.

Jedná se o dietologickou aplikaci, která pomáhá lidem s hubnutím, popřípadě udržením hmotnosti, vylepšením tělesného stavu organizmu. Grafický design této aplikace sleduje designové principy Google pro androidí aplikace. Aplikaci po grafické stránce hodnotím pozitivně.

Aplikace slouží pro zadávání zkonzumovaného jídla, jídlo by mělo být konzumováno a zadáváno tak, aby uživatel plnil svůj plán, který je mu vypočten na základě údajů, které zadal při registraci. Den uživatele je rozdělen na sekce: snídaně, oběd, večeře, svačina. Tyto sekce mají vypočtené limity příjmu kalorií, těchto limitů by se měl uživatel držet. Konzumace lze poté přidávat z několika různých zdrojů, těmi jsou potraviny, jídla a recepty. Jednotlivé potraviny mají různé jednotky pro přidávání, např. 100g, hrníček. Tato vlastnost rozhodně usnadňuje odhadování množství potravin při jejich přidávání.

Pro vstup do aplikace je vyžadováno přihlášení. Běh aplikace v offline modu není možný, aplikace si evidentně nějaká data cashuje, ale nelze v ní plnohodnotně vyhledávat jídla, když už se nám podaří nějaké jídlo najít, nepodaří se nám ho přidat.

Tato aplikace je lokalizovaná do světových jazyků i do češtiny. Obsahuje podrobnou databázi český jídel. Aplikace má v základní neplacené verzi nedostupnou spoustu funkcionalit. [11]

4.2 My Diet Coach - Weight Loss

Počet stažení na Google Play je pro tuto aplikaci řádově 10 milionů uživatelů. Aplikace byla naposledy aktualizována 2. dubna 2017.

Tato aplikace není aplikací pro zaznamenávání zkonzumovaného jídla, jedná se spíše o motivační aplikaci pro hubnutí. Jediné co připomíná zadávání potravin jsou challenge, kdy uživatel zaznamenává množství vypité vody, popřípadě na kolik talířů jídla si dal půl objemu zeleniny, nebo kolikrát zaparkoval od práce dál a šel do ní pěšky.

Aplikace dále obsahuje levelovací systém, ve kterém může uživatel postupovat. Je zde sekce s různými typy, jak se vyhnout přejídání. Dalším motivačním prvkem ke zhubnutí jsou notifikace, které si zde uživatel může přednastavit, aplikace ho např. upozorní na to, aby si do práce vzal tašku s oblečením na cvičení. Je zde i modul s fotografiemi, jenž umožňuje uživatelům se pochlubit se svými úspěchy.

Aplikace je plná animací, celkový design je velmi povedený. Placená verze aplikace obsahuje mnohem více funkcionalit. Jsou zde např. dietní a cvičební plány, přidávání snědeného jídla. [12]

4.3 Diabetes:M

Počet stažení na Google Play je pro tuto aplikaci řádově 100 tisíc uživatelů. Aplikace byla naposledy aktualizována 11. dubna 2017.

Aplikace při vstupu vyžaduje registraci. Pozitivně hodnotím možnost zvolit si jednotky, v nichž bude aplikace pracovat. Jsou zde vyžadovány konkrétní hodnoty jako např. citlivost na inzulin.

Aplikace mi zprvu připadala zmatečná. Jsou zde dva formuláře na vyplňování záznamů. V prvním formuláři se zadává čas, glykémie, sacharidy, inzulin a ještě poznámka. Tyto všechny hodnoty lze zadat najednou, formulář se ukládá tapnutím na ikonou v navigation baru, přičemž v navigation baru je více ikon. Chvíli mi trvalo než jsem přišel na to, jak záznam uložit. V dalším formuláři se zadává najednou tlak, tep, fyzická aktivita, hmotnost a ketony, formulář se opět potvrzuje tlačítkem v navigation baru. V aplikaci je možné si vyhledat potravinu, zadat její množství. Potvrzením vyhledávacího formuláře se předvyplní první formulář pro zadávání, tento formulář pak může uživatel potvrdit, a tím vytvořit záznam o konzumaci potraviny. Ovládání aplikace je místy velmi krkolomné.

Z dat aplikace je možné dělat exporty, data aplikace si uživatel může synchronizovat na různá úložiště.

Aplikace obsahuje sekci s grafy glykémie a s dalšími grafy tvořenými ze zadaných hodnot. [13]

4.4 Cukrovka - Glukóza deník

Počet stažení na Google Play je pro tuto aplikaci řádově 100 tisíc uživatelů. Aplikace byla naposledy aktualizována 19. dubna 2017.

Na první pohled mě překvapilo barevné schéma této aplikace, všechno bylo laděné do tmavě modré, naštěstí to šlo v nastavení aplikace přepnout na bílou, ve které aplikace vypadala mnohem lépe. Aplikace má jen velmi omezenou funkcionalitu, umožňuje zadávat glykémii, tato měření pak aplikace vykresluje do grafu průběhu glykémie. Z aplikace lze generovat exporty, rovněž lze do aplikace exporty importovat. [14]

Aplikace s taktovouto funkcionalitou může sloužit jako diabetologický deník. Tato aplikace je důkazem, že i relativně malá dobře udělaná funkcionalita si své zákazníky najde.

4.5 Diabetes Connect

Počet stažení na Google Play je pro tuto aplikaci řádově 100 tisíc uživatelů. Aplikace byla naposledy aktualizována 5. října 2016.

Tato aplikace mi ze všech diabetologických aplikací připadala jako jedna z nejpoužitelnějších. Aplikace umožňuje uživateli přidávat záznamy, v jednom záznamu může uživatel zadat krevní cukr, počet přijatých výměnných jednotek, počty jednotek jednotlivých druhů inzulínů, krevní tlak, puls, váhu, sportovní aktivitu, poznámku, a další věci. Zadané informace si poté uživatel může prohlížet v přehledu záznamů. Nevýhodou způsobu zadávání této aplikace je, že když uživatel do záznamu zadá např. glykémii, ostatní položky tohoto záznamu zůstanou prázdné. V přehledu se poté u daného záznamu zobrazí hodnota glykémie a všechny ostatní položky zůstanou proškrtnuté, to může působit trochu nepřehledně.

Aplikace dále nabízí možnost vytvářet reporty z naměřených dat. V základní verzi aplikace je možné zobrazit pouze graf průběhu glykémie. Aplikace obsahuje část se statistikami zadaných dat. [15]

4.6 Závěr z výzkumu konkurenčních aplikací

Aplikace, které se specializují především na dietu mají dobré rozsáhlé databáze potravin, o kterých znají jejich výživové hodnoty. Naproti tomu aplikace, které se specializují na diabetes v sobě nemají databáze potravin, zadávání nové konzumace se v nich zredukovalo na zadání počtu jednotek přijatých sacharidů. Diabetes druhého typu souvisí s obezitou, proto podle mého názoru by výsledná aplikace měla mít obojí, tedy část věnující se diabetu, ale i propracovanou dietní složku.

Jednou z pěkných vlastnosti aplikace YAZIO byly volitelné jednotky pro hmotnost jídla, např. jablka se měří na kusy a ne na gramy. S implementací této vlastnosti do výsledné aplikace nepočítám v první verzi. Implementování by zabralo příliš práce, muselo by se hodně měnit jádro aplikace v serverové části.

Aplikace My Diet Coach vyniká propracovanými motivačními prvky, které uživatele vedou k dodržení diety. V první verzi mé aplikace s podobnými motivačními prvky nepočítám. Cílem mé aplikace bude poskytnout základní funkcionalitu pro vedení záznamů diabetika, popřípadě zdravého člověka, který si hlídá své příjmy.

Diabetologické aplikace většinou obsahují mechanizmy exportu dat z aplikace ven, popřípadě importu dat do aplikace. Rozhodl jsem se, že v první verzi nové aplikace tato vlastnost bude chybět, plánuji se soustředit především na základní funkcionalitu, která jak doufám přitáhne uživatele.

Jedna z vlastností, která provázela zkoumané aplikace, byla konfigurovatelnost, uživatelé si mohou při registraci vybrat jednotky, ve kterých budou jednotlivé záznamy zadávat. Možnost změny jednotek bude implementována již v první verzi aplikace.

Možnost dokoupit nějaké vlastnosti pro dané aplikace a nebo alespoň nějaká přítomnost reklamy jsou vlastnosti vyskytující se v testovaných aplikacích. S tímto se v první verzi mé aplikace nebude počítat. Aplikace bude svou funkcionalitou velmi rozsáhlá, bohužel není čas a prostor pro implementaci všech vlastností a funkcionalit.

Další věcí, která provázela snad všechny zkoumané aplikacem, byla nutnost přihlášení pro jakékoliv vyzkoušení aplikace. Této vlastnosti bych se chtěl ve své aplikaci zbavit. Chtěl bych nepřihlášenému uživateli nabídnout podobnou, byť omezenou, funkcionalitu jako je nabídnuta uživateli přihlášenému.

Pro fungování zkoumaných aplikací bylo nutné mít stálé internetové připojení. Tuto potřebu se pokusím ve své aplikaci co nejvíce odbourat. Díky tomu budou moci uživatelé aplikaci používat i na místech, kde není internetové připojení. Budou si tedy moci přidávat záznamy kdykoliv a kdekoliv, např. při jízdě metrem.

Kapitola 5
Funkční požadavky

Na základě obsahu předchozích kapitol jsou zde sestaveny základní funkční požadavky na aplikaci Dia Dieta. V závěru lze zhodnotit, zda jednotlivé požadavky byly a nebo nebyly v konečné aplikaci splněny. Požadavky jsou doplněny o nefunkční požadavky v závěru kapitoly č. 6. Až na konci této kapitoly jsou totiž známy hlavní technologie, na nichž bude celý systém vystavěn.

1. Zadávání záznamů

a) Aplikace umožní zadávat konzumace jídel.
b) Aplikace umožní zadávat hladinu glykémie.
c) Aplikace umožní zadávat dávky inzulinu.
d) Aplikace umožní zadávat vykonané fyzické aktivity.
e) Aplikace umožní zadávat hodnotu krevního tlaku a obvod pasu.
f) Aplikace umožní zadávat uživatelovu hmotnost.
g) Aplikace umožní vytvořit nové jídlo.
h) Aplikace umožní vytvořit nové složené jídlo.

2. Zobrazení denních přehledů

a) Aplikace umožní zobrazit denní přehled všech záznamů.
b) Denní přehled bude obsahovat sekci s denními limity pro energii, sacharidy, tuky a bílkoviny.
c) Denní přehled bude obsahovat agregované hodnoty pro energii, sacharidy, tuky a bílkoviny.
d) Aplikace umožní zobrazit týdenní přehled záznamů, který bude obsahovat agregované hodnoty pro energii získanou z potravin, pro glykémii, pro inzulin a pro sportovní aktivity.
e) V týdenním přehledu budou zobrazeny průměrné hodnoty spočítané přes celý týden.
f) V týdenním přehledu budou zobrazeny hodnoty pro energii získanou z potravin, pro glykémii. Tyto hodnoty budou podbarveny červeně. Intenzita této barvy bude odpovídat tomu, jak moc se daná hodnota vymyká limitům stanoveným pro daného uživatele.

3. Zobrazení grafů

a) Aplikace umožní zobrazit graf průběhu glykémie a graf dávek inzulinu v jednom grafu.
b) Aplikace umožní zobrazit graf průběhu glykémie za celý týden. Celý týdenní průběh bude zobrazen v jednom dni.
c) Aplikace umožní zobrazit graf přijmu energie z potravy a energie vydané cvičením.
d) Aplikace umožní zobrazit graf vývoje uživatelovy hmotnosti a obvodu jeho pasu.

4. O aplikaci

a) Aplikace umožní uživateli zobrazit obrazovku o aplikaci.

5. Přihlášení

a) Aplikace umožní uživateli se přihlásit.
b) Aplikace umožní uživateli vytvořit nový účet.
c) Aplikace umožní uživateli zresetovat zapomenuté heslo.
d) Aplikace umožní uživateli změnit heslo.
e) Aplikace umožní uživateli změnit uživatelské jméno.

6. Uživatelský profil

   a) Aplikace umožní uživateli změnit údaje, které zadal při přihlášení.
   b) Aplikace umožní uživateli změnit hranici hypoglykémie a hyperglykémie.
   c) Aplikace umožní uživateli přepínat zobrazení množství sacharidů mezi výměnnými jednotkami a gramy.
   d) Aplikace umožní uživateli předefinovat hodnotu, která udává kolik gramů sacharidu připadá na jednu výměnou jednotku.

7. Další

   a) Aplikace bude potřebovat internetové připojení pouze pro změnu uživatelského jména, hesla, reset hesla, přihlášení, registraci, úvodní stažení dat. Další akce se obejdou bez připojení.
   b) Jestliže po zapnutí aplikace bude dostupné internetové připojení, aplikace se pokusí ze serveru stáhnout nová data, která ještě nejsou stažena. Aplikace se rovněž pokusí na server nahrát data, která vznikla v aplikaci a ještě nebyla synchronizována.
   c) Aplikace umožní používání i nepřihlášenému uživateli. Ten však přijde o některé výhody: synchronizace dat na server, zobrazení limitů příjmů, možnost přepínat si jednotky, v kterých aplikace pracuje, přenastavovat limity pro hypo a hyperglykémii, možnost předefinovat si, kolik gramů sacharidu připadá na jednu výměnou jednotku, možnost upravovat si položky uživatelova profilu, které zadal při registraci.

Kapitola 6
Výběr technologií

6.1 Výběr mobilní platformy

Období	Android	iOS	Windows Phone	Další
2015Q4	79.6%	18.7%	1.2%	0.5%
2016Q1	83.15%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%
2016Q3	86.8%	12.5%	0.3%	0.4%

Tabulka 6.1. Rozložení podílu na trhu mezi mobilními platformami, zdroj: [16].

Z Tabulky č. 6.1 vyplývá, že mobilní trh ovládají v současné době dva operační systémy, jeden hráč s větším podílem Android a druhý hráč s nezanedbatelným podílem iOS, pro ostatní platformy nemá v podstatě cenu vyvíjet.

K této situaci lze přistoupit několika způsoby, vybrat si jednu platformu a aplikaci vytvořit pouze pro ní, vytvořit aplikace pro obě platformy zvlášť nebo najít nějaký framework, který umí z jednoho kódu vytvořit dvě aplikace pro různé platformy, a pracovat s ním.

6.1.1 Nativní aplikace

Takzvané nativní aplikace jsou vyvinuty specificky pro danou platformu a nainstalovány v zařízení podobně, jako jsou nainstalovány aplikace ve stolním počítači. Mohou využívat hardwarových dispozic telefonu (GPS, fotoaparát, kalendář, kontakty) a díky snadné optimalizaci jsou rychlé, spolehlivé a více šetří baterii zařízení. [17] Nativní aplikace mají uživatelské prostředí poskládané z komponent typických pro danou platformu.

6.1.2 Xamarin

Xamarin je nástroj od společnosti Microsoft, který umožňuje multiplatformní vývoj nativních mobilních aplikací. Jako hlavní vývojový jazyk Xamarin používá jazyk C#. Xamarin umožňuje mezi platformami sdílet až 90% kódu aplikace, zbylých 10% tvoří platformně závislý kód, který je ovšem opět psán v jazyce C#. Pro překlad aplikace pro platformu iOS je potřeba aplikaci buildit na operačním systému Mac ve verzi alespoň OS X Yosemite. [18]

Na jedné straně existuje nativní vývoj pomocí Xamarinu, kde máme společné jádro aplikace a mapování do jednotlivých specifických platforem se řeší pro každou platformu zvlášť. Na druhé straně stojí tzv. Xamarin.Forms. Ty umožňují 100% sdílení kódu mezi jednotlivými platformami, mapování do jednotlivých platforem mají pouze jedno. Tento nástroj, ovšem neumí vyrobit nativní pixel perfect vzhledy pro aplikace na jednotlivé platformy, používá se především na prototypování. [19]

6.1.3 QT

Jedná se o framework pro multiplatformní vývoj v jazyce C++. QT fungují na různém hardwaru, jejich hlavní použití je v embeded zařízeních. QT jdou na multiplatformní vývoj jiným způsobem než Xamarin, místo toho, aby používaly prvky z uživatelských rozhraní jednotlivých platforem, vytváří si na jednotlivých platformách svoje vlastní prvky. To ovšem znamená, že výsledná aplikace bude na všech platformách vypadat více či méně stejně, tím se vytratí nativní vzhled aplikace.

QT jsou distribuovány pod dvěma licencemi. První licence je komerční, ta stojí 295$ měsíčně. Druhá licence GPL & LGPLv3 je zdarma. Použití QT pod touto druhou licencí by znamenalo, že výsledná aplikace by musela být nabízena opět pod touto licencí. Konečná aplikace by musela být open source, navíc by u výsledného produktu musel být implementován mechanismus, pomocí kterého by se daly QT knihovny vyměnit za jiné. [20]

6.1.4 Další hráči

Kromě těchto dvou velkých hráčů se na trhu vyskytují další, kteří využívají html a javascript a nebo se specializují na herní prostředí, např. Unity. Nástroje pracující s javascriptem a html ovšem nevytváří nativní aplikace, aplikace vytvořené pomocí těchto nástrojů budou omezené ve své funkčnosti a vzhledu. [21]

Herní frameworky jsou určené převážně pro tvorbu her, princip který používají na dosažení portace aplikace na různé platformy je podobný jaký používají QT.

6.1.5 Závěr

Vzhledem k různým omezením pro multiplatformní frameworky jsem se rozhodl jít cestou nativní aplikace pouze pro jednu platformu. Vybral jsem si tedy tu platformu, která má v absolutních číslech větší rozšíření. Touto platformou je Android.

6.2 Výběr mobilní databáze

Výsledná aplikace bude fungovat offline, bude pracovat s předem staženými daty. Pro toto stažení dat je nutné přidat do aplikace databázi.

6.2.1 SQlite

Jedná se o knihovnu, která implementuje transakční SQL databázi. Pro svůj běh databáze nepotřebuje žádný další serverový proces. Všechna data databáze jsou uložena pouze v jednom souboru. Tento soubor můžeme volně přenášet mezi jednotlivými platformami. [22]

SQlite databáze je nativně integrovaná do Androidu. Kromě svých výhod, mezi něž patří rychlost a spolehlivost, má i své nevýhody. Jedná se o sql databázi, to znamená, že musíme nadefinovat a spravovat schéma databáze. Dotazováním databáze získáme tzv. Cursor. Z Cursoru pak musíme data ručně mapovat do objektů. Na druhé straně, když chceme daný objekt zapsat do databáze, musíme z něj vyrobit tzv. ContentValues, což je objekt, který obsahuje klíče a hodnoty, jež chceme nahrát do databáze. Práce pouze přímo s SQlite databází je nepohodlná, musíme napsat spoustu kódu, který mapuje data z databázového objektu na ContentValues a z Cursoru na databázový objekt. Naštěstí na SQlite navazuje spousta frameworků, které tuto práci zjednodušují.

Výhodou použití čisté SQlite databáze je to, že databázi můžeme schovat pod ContentProvider, to nám umožní společně s knihovnou CursorLoader dosáhnout toho, že kdykoliv se data v databázi změní, aplikace je na to upozorněna a dojde k znovunačtení dat z databáze. [23] [24]

6.2.2 OrmLite, SugarORM, GreenDAO, Active Android

Knihovny vyjmenované v titulku této podkapitoly používají většinou SQlite databázi, umožňují jednodušší práci s touto databází. Většinou není třeba generovat databázové schéma ručně, schéma se vygeneruje automaticky z anotací databázových tříd. Dále lze objekty přímo vkládat a vytahovat z databáze, není potřeba žádných Cursorů a ContentValues. Rychlost těchto databází je obecně menší než rychlost SQlite. Další nevýhodou je, že dané databáze umí pracovat většinou jen ve vlákně, ve kterém je daný dotaz volaný, volání dotazů na pozadí si musíme naprogramovat sami. [25]

V dokumentacích těchto databází jsem se nedočetl nic o tom, že by podporovaly nějakou podobnou funkcionalitu jako jsou Loadery. Znamená to tedy, že když se data v databázi změní, nedostaneme žádné upozornění. Nenalezl jsem žádnou informaci ani o podpoře transakcí, předpokládám tedy, že transakce tyto databáze neimplementují.

6.2.3 Realm

Realm databáze obsahuje jádro naprogramováné v C++, nevyužívá SQlite databáze. Jedná se o objektovou databázi, můžeme z ní vybírat a zároveň do ní nahrávat přímo objekty. Databáze umožňuje migrace schématu. Všechny změny dat v databázi musí povinně běžet v transakci.

Jednou z klíčových vlastností, které na této databázi oceňuji, je možnost si vybrat, zda daný dotaz či update provedeme synchronně a nebo asynchronně. Za další klíčovou vlastnost lze považovat schopnost notifikovat uživatelské prostředí o změnách v databázi, podobné chování mají Loadery.

```
Realm realm = Realm.getDefaultInstance();
realm.setAutoRefresh(true);

RealmResults<UserBO> userResult = realm.where(UserBO.class).findAllAsync();

userResult.addChangeListener(
    new RealmChangeListener<RealmResults<UserBO>>() {
        @Override
        public void onChange(RealmResults<UserBO> element) {

        }
});
```

Výše uvedený kód najde všechny uživatele. Dotaz proběhne asynchronně. Poté co jsou všichni uživatelé načteni, je volána metoda onChange. Tato metoda má jako parametr kolekci všech načtených uživatelů. Jestliže dojde k nějaké změně v databázi (např. se na pozadí stáhnou ze serveru nová data a uloží do databáze), dojde k dalšímu volání metody onChange, tentokrát s novými daty.

Podle různých neoficiálních benchmarků by tato databáze měla být dokonce rychlejší než SQlite.

6.2.4 Závěr

Po zvážení všech pro a proti jsem se rozhodl pro Realm. Realm implementuje rychlou objektovou databázi. Mezi jeho hlavní výhody patří absence nutnosti implementovat mapování mezi ContentValues, Cursorem a databázovými objekty. Realm z databáze vytahuje objekty a vkládá je do ní přímo. Další výhodou je automatické přenačtení dat v případě, že dojde v databázi ke změně. Nevýhodou Realmu je to, že není odladěný a občas se chová nepředvídatelně.

6.3 Nefunkční požadavky

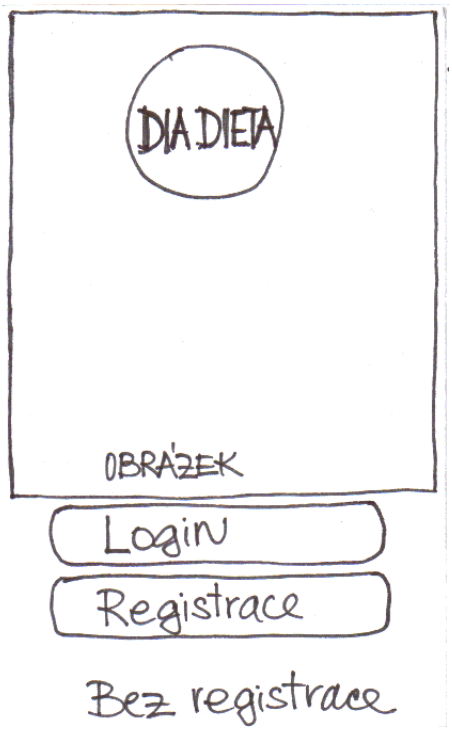
Na základě rozhodnutí učiněných v této kapitole jsou požadavky aplikace Dia Dieta doplněny o následující nefunkční požadavky.

1. Aplikace bude vyrobena pouze pro platformu Android.
2. Aplikace bude fungovat na zařízeních s Androidem 4.0 a výše.
3. Aplikaci bude možné používat v portrait i landscape módu.

Kapitola 7
Návrh aplikace

V této sekci budou zobrazeny a popsány původní návrhy význačných obrazovek aplikace. Finální podoba obrazovek bude k nahlédnutí v dalších kapitolách práce.

7.1 Uvítací obrazovka aplikace



Obrázek 7.1. Uvítací obrazovka aplikace

Obrázek 7.1 zachycuje návrh úvodní obrazovky aplikace. Na obrazovce je zobrazeno logo aplikace a fotografie diabetických pomůcek. V dolní části obrazovky jsou tři tlačítka, první tlačítko slouží pro přihlášení, druhé pro registraci, poslední slouží pro vstup do aplikace bez jakékoliv registrace či přihlášení. Při vstupu do aplikace je tato obrazovka zobrazena pouze uživatelům, kteří nejsou zaregistrováni. Zaregistrovaní uživatelé jsou při spuštění přesměrováni rovnou do aplikace.

7.2 Registrační obrazovka

Obrazovka 7.2 slouží pro zadání nezbytných údajů pro registraci uživatele. V sekci osobní informace uživatel zadává důležité informace o sobě a zároveň informaci o tom, zda trpí nějakým typem diabetu a informaci o intenzitě svého pohybu. Sekce denní limity přijmu slouží pro zadání hodnot denních limitů přijmu pro energii, sacharidy, tuky a bílkoviny. Uživatel si může tyto

Obrázek 7.2. Registrační obrazovka

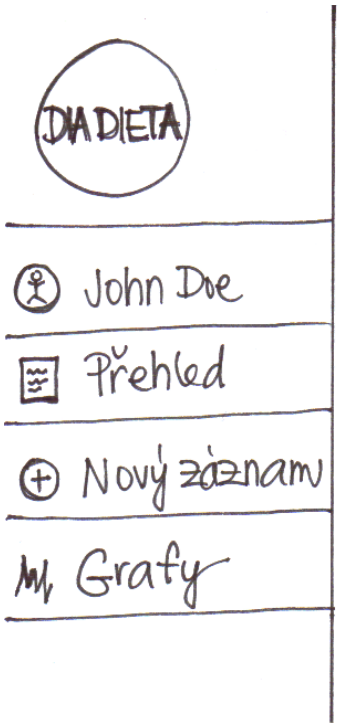
hodnoty vyplnit sám nebo může použít tlačítko `Přepočítat`. Tlačítko `Přepočítat` spočte limity příjmu na základě uživatelova věku, cílové hmotnosti a jeho aktivity. Pro samotný výpočet se použijí vztahy popsané v podkapitole č. 2.1. Poslední sekce registračního formuláře obsahuje pole pro vyplnění hesla a checkbox pro odsouhlasení podmínek používání.

7.3 Přihlašovací obrazovka

Obrázek 7.3. Přihlašovací obrazovka

Obrazovka č. 7.3 slouží pro zalogování do aplikace.

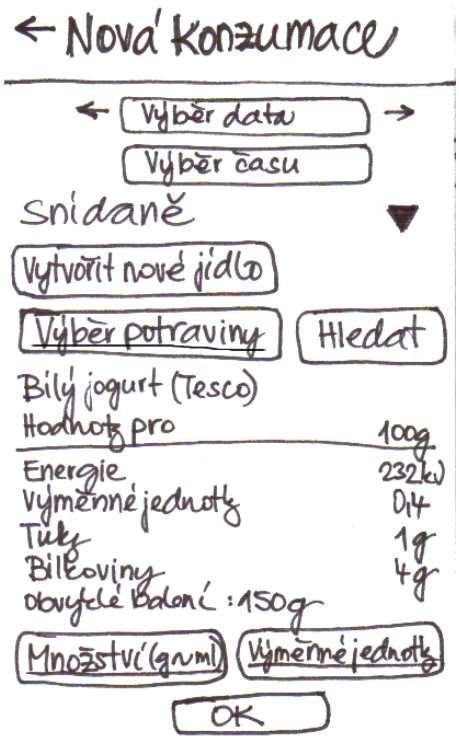
7.4 Hlavní menu



Obrázek 7.4. Hlavní menu

Obrázek č. 7.4 pochází již z vnitřku aplikace. Obrázek zachycuje konkrétně hlavní menu aplikace. Ve vrchní části se nachází logo. Pod logem je seznam obrazovek, na které se uživatel může prokliknout. První položka menu zobrazuje jméno přihlášeného uživatele a zároveň slouží jako odkaz na obrazovku s detailními informacemi o daném uživateli. Následující odkaz `Přehled` vede na obrazovku s denním a týdenním přehledem záznamů. Odkaz `Nový záznam` vede na obrazovku umožňující přidávat záznamy (Jídlo a pití, Glykémie, Inzulin, Aktivita, krevní tlak, Hmotnost). Poslední odkaz `Grafy` vede na obrazovku, která zobrazuje zadané záznamy ve formě grafů.

7.5 Nová konzumace



Obrázek 7.5. Nová konzumace

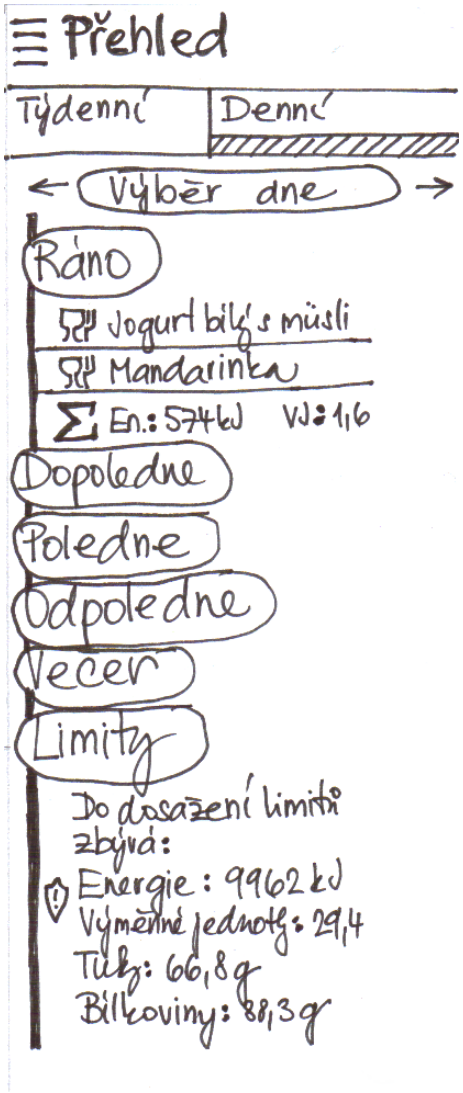
Obrázek č. 7.5 zobrazuje návrh obrazovky pro přidání konzumace jídla. Ve vrchní části obrazovky se nachází prvek pro výběr data a času. Na všech obrazovkách zabývajících se zadáváním, bude pro výběr času použita tato komponenta. Obrazovka dále obsahuje tzv. Spinner, pomocí kterého lze rozhodnout, zda zadaná konzumace bude např. snídaně, nebo oběd. Následuje tlačítko které uživatele přesměruje na obrazovku, na které si může vytvořit svoje vlastní jídlo.

Následující sekce se zabývá výběrem potraviny. V návrhu je tento výběr řešen hybridním řešením. Vyskytuje se zde textové pole pro zadávání, které automaticky filtruje databázi jídel. Jídla, která projdou filtrem, jsou poté zobrazena ve vyskakovacím seznamu pod tímto zadávacím prvkem. Vedle tohoto prvku se v návrhu objevuje tlačítko `Hledat`, toto tlačítko vede na obrazovku vyhledávající jídla pomocí názvu a popřípadě kategorií, do kterých toto jídlo patří. Pod těmito vyhledávacími prvky se vyskytuje sekce zobrazující detailní informace o dané vybrané potravině.

V další části této obrazovky se zadává množství zkonzumované potraviny, množství lze zadat v gramech, popřípadě lze zadat výměnné jednotky. V této sekci stačí zadat pouze jednu z těchto informací, ta druhá se automaticky dopočítá.

Konečný formulář se potvrzuje tlačítkem ve spodní části obrazovky.

7.6 Denní přehled



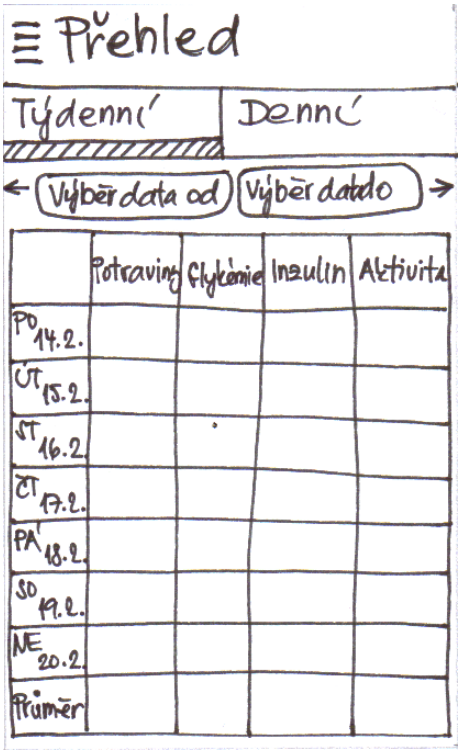
Obrázek 7.6. Denní přehled

Obrazovka č. 7.6 zachycuje přehled všech zadaných záznamů během jednoho dne. Ve vrchní části obrazovky se nachází sub menu, které umožňuje přepínat mezi obrazovkou s denním a týdenním přehledem. Pod tímto sub menu se nachází grafický prvek pro volbu dne, jehož přehled si přejeme zobrazit.

Samotné zobrazované záznamy jsou tříděny do šesti kategorií. Příslušnost k dané kategorii je pro každý záznam spočtena na základě zadaného času. Poslední kategorie se jménem `Limity` neslouží pro zobrazování záznamů, ale pro zobrazení hodnot, jež uživateli říkají, kolik mu zbývá sníst jednotlivých složek potravy do dosažení denních limitů. Jednotlivé kategorie mohou být kliknutím sbaleny, dalším kliknutím rozbaleny. Kategorie zobrazující záznamy (všechny kromě kategorie `Limity`) obsahují jako poslední položku sumu hodnot pro dané denní období.

Všechny položky obsažené v jednotlivých kategoriích se dají rozkliknout, tím se zobrazí jejich detailní obsah, po rozkliknutí se dají opět sbalit.

7.7 Týdenní přehled



Obrázek 7.7. Týdenní přehled

Obrazovka č. 7.7 zachycuje týdenní přehled zadaných záznamů. Nahoře se vyskytuje prvek, díky němuž lze vybrat konkrétní týden, pro který, chceme zobrazovat přehled. V týdenním přehledu jsou zobrazeny jen ty nejdůležitější záznamy a to konkrétně energie a sacharidy přijaté z potravy, hodnoty naměřené glykémie, dávky inzulinu a sportovní aktivita. Jednotlivé řádky tabulky přísluší k jednotlivým dnům v týdnu, poslední řádek tabulky poté ukazuje průměrné hodnoty spočtené ze všech dní ve zvoleném týdnu.

Kapitola 8
Implementace

8.1 Výběr architektonického stylu

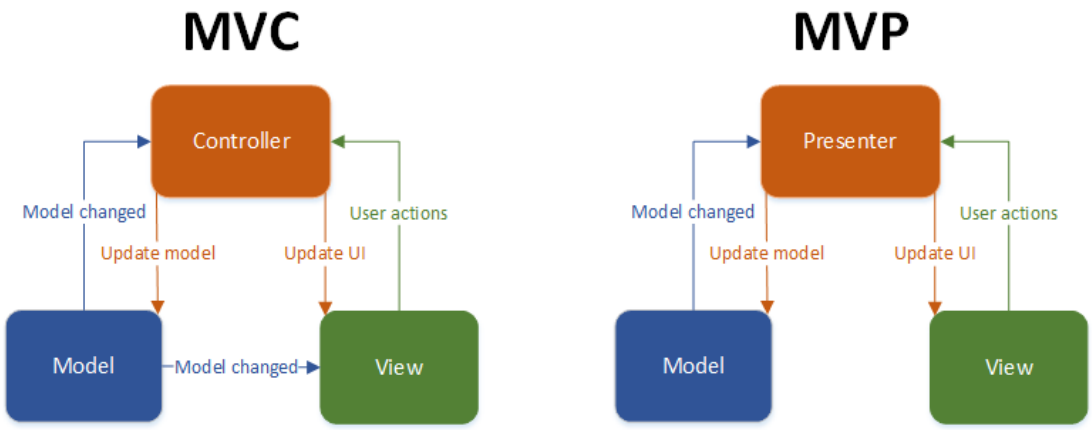
Architektonický styl má zá úkol oddělit od sebe jednotlivé vrstvy aplikace, tak aby všechna logika, načítání dat z databáze, transformace dat a jejich zobrazení nebyly součástí kódu jedné třídy. Kód vytvořený bez dobrého architektonického stylu je neudržovatelný i malá změna znamená přepis kódu na mnoha místech. Kromě toho, že takový kód je těžce udržovatelný, je také netestovatelný a více náchylný k chybám. Kód bez správného stylu vede k tomu, že prvotní vývoj může být rychlejší, skoro každá dodatečná změna ovšem znamená obrovské zásahy a přepsání velké části logiky.

Z těchto důvodů je důležité před začátkem implementace učinit rozhodnutí, jaký architektonický vzor bude použit. Výběr, správná implementace a dodržení zásadně ovlivní testovatelnost, čistotu kódu, rozšiřitelnost, chybovost a další atributy systému.

8.1.1 Model View Controller, Model View Presenter

Myšlenka těchto dvou modelů je podobná, oba se snaží rozdělit aplikaci na tři samostatné části.

- Komponentě udržující stav systému se říká Model, často se jedná o databázi s daty.
- Komponentě zpracovávající vstup a výstup od uživatele se říká View.
- Komponentě obsahující logickou funkcionalitu systému se říká Controller/ Presenter.



Obrázek 8.1. Model View Controller/Presenter

Rozdíly mezi MVC a MVP jsou následující.

1. V architektuře MVC je View notifikováno každou změnou, která se stane v Modelu. V architektuře MVP View neví nic o Modelu. Updatovat View na základě změn v Modelu je úlohou Presenteru.
2. View v architektuře MVC má v sobě více logiky než v architektuře MVP, stará se totiž o zpracování dat z Modelu. V architektuře MVP je tato logika přesunuta z View do Presenteru.

3. Model v architektuře MVC v sobě obsahuje logiku na spolupráci s View.

Základní architektura Androidí aplikace není ovšem nakloněna použití těchto stylů, Model se dá z aplikace lehce oddělit, problém nastane, když se pokusíme od sebe oddělit View a Controller/Presenter. Žádná oficiální knihovna, která by napomáhala s implementací těchto návrhových stylů, není. Na Githubu se ovšem dají najít knihovny, jež tyto návrhové architektonické styly implementují. [26]

Dá se říci, že MVP je evolučním nástupcem MVC, na jednu stranu přináší odlehčení View, větší modularitu systému, lepší možnosti testování, změny se lépe integrují do systému. Na druhou stranu s sebou přináší i některé nevýhody, velký počet interfaců pro komunikaci mezi jednotlivými vrstvami, interfacy mají spoustu metod a velkou rozsáhlost kódu. [27]

8.1.2 Model-View-ViewModel

Jedná se o architektonický styl, který byl přinesen na platformu Android s příchodem databindingu (více viz podkapitola č. 8.1.3). MVVM se skládá ze tří částí:

1. View - Je zodpovědné za vizualizaci dat. Zobrazuje a upravuje data ViewModelu.
2. ViewModel - Obaluje Model a poskytuje rozhraní pro View, které se na něj pomocí databingu napojuje. Změna ve ViewModelu se automaticky promítá do View. Naopak View může komunikovat s ViewModelem, dojde-li k nějaké události, např. ke kliknutí nebo ke změně hodnoty ve formuláři.
3. Model - Komponenta, jež udržuje stav systému, často se jedná o databázi.

Celý systém poté funguje následovně. Na počátku jsou vytvořeny View a ViewModel. View se spojí s ViewModelem pomocí volání příslušné metody na View. ViewModel začne číst data z Modelu a začne je přes databinding zobrazovat do View. Jesliže dojde ke změně v Modelu, ViewModel automaticky načte nová data z Modelu a zobrazí je do View. ViewModel dále reaguje na události, které vznikají ve View (např. kliknutí na tlačítko). Na základě těchto událostí upravuje Model.

Použití MVVM s sebou nese různé nevýhody. Některé části kódu končí v xml souborech, to stěžuje ladění aplikace. V aplikaci pracujeme s některými prvky z uživatelského prostředí přes databinding, někde se však nevyhneme klasickému přístupu. Napsání testů bude mnohem těžší než v případě MVP.

Na druhé straně MVVM přináší i některé výhody. Na implementaci nám stačí použít oficiální knihovny od Googlu pro databinding (více viz pokapitola č. 8.1.3). Vyhneme se klasickému biolerplate kódu při práci s view, např. findViewById, setText atd. Tyto věci se odehrávají pomocí databindingu. [27]

8.1.3 Android databinding

Jedná se o knihovnu napomáhající redukci množství kódu, jenž je potřeba pro propojení aplikační logiky a View. Tato knihovna je oficiálně vydávaná Googlem, pro podporu databindingu stačí do souboru build.gradle přidat následující. [28]

```
    dataBinding {
        enabled = true
    }
```

Následující kód ilustruje použití databindingu.

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
   <data>
       <variable name="viewModel" type="com.example.ViewModel"/>
```

```
    </data>
        <EditText android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{viewModel.firstName}"
            android:onClick="@{viewModel.onFirstNameClickListener}"
            />
</layout>
```

Výše uvedený kód zajistí, že se na obrazovce objeví text, který je přítomen ve ViewModelu a to konkrétně v atributu firstName. Při kliknutí na tento grafický element dojde k notifikaci objektu onFirstNameClickListener o kliknutí.

S touto knihovnou lze dosáhnout následujícího. Kdykoliv změníme ve viewModelu atribut firstName, dojde k automatickému updatu hodnoty v zobrazovaném grafickém prvku. Přidáním další logiky (OnTextChangeListener) lze docílit změny textu v atributu firstName, kdykoli uživatel změní text v grafickém prvku pomocí klávesnice. Tím získáme tzv. Two-way databinding.

Databinding umožní zrušení závislosti ViewModelu na View. Přinese pouze jednostrannou závislost View na ViewModelu.

8.1.4 Závěr

Ve své práci jsem se rozhodl použít MVVM. Tento architektonický styl jsem implementoval sám podle svých potřeb od začátku, využil jsem pouze databindingu v Androidu 8.1.3.

Jako Model mi posloužila databáze, nad kterou jsem postavil servisní vrstvu. Ke každé tabulce v databázi jsem vytvořil servisní třídu. Tyto jednotlivé servisní třídy mají spoustu společné logiky ve svém předkovi BaseDbServise, ve svém těle obsahují pouze implementaci abstraktních metod z BaseDbServise a dále např. nějaké speciální dotazy do databáze.

V Androidu se uživatelský interface zapisuje pomocí xml souborů, s použitím databindingu jsem docílil vygenerování tříd z daných xml souborů, které reprezentují View. S těmito třídami se v aplikaci pracuje pouze tak, že od nich vytvořím instanci a propojím je s ViewModelem. Následuje typická ukázka z inicializace fragmentu.

```
 public View onCreateView(LayoutInflater inflater,
  @Nullable ViewGroup container,
  @Nullable Bundle savedInstanceState) {
        binding = FragmentGraphBinding.inflate(inflater, container, false);
        viewModel = new GraphEnergyViewModel(getContext(),
                getBundleWithSavedData(savedInstanceState),
                GraphDemoFragment
                .createGraphViewModelFragmentI(drawerActivityI));
        binding.setViewModel(viewModel);
        return binding.getRoot();
}
```

Třída FragmentGraphBinding je generovaná z xml souboru, ve kterém je popsáno jak má daná obrazovka vypadat. Instance třídy GraphEnergyViewModel představuje v tomto případě ViewModel. Voláním `binding.setViewModel(viewModel);` dojde k propojení View a VieModelu. Třídy ViewModelu v sobě obsahují rozhraní, na které se napojují jednotlivá View, každé specifické View se napojuje na svůj specifický ViewModel.

Každá třída reprezentující v aplikaci ViewModel dědí od třídy BaseViewModel, která obsahuje logiku pro dvě základní věci.

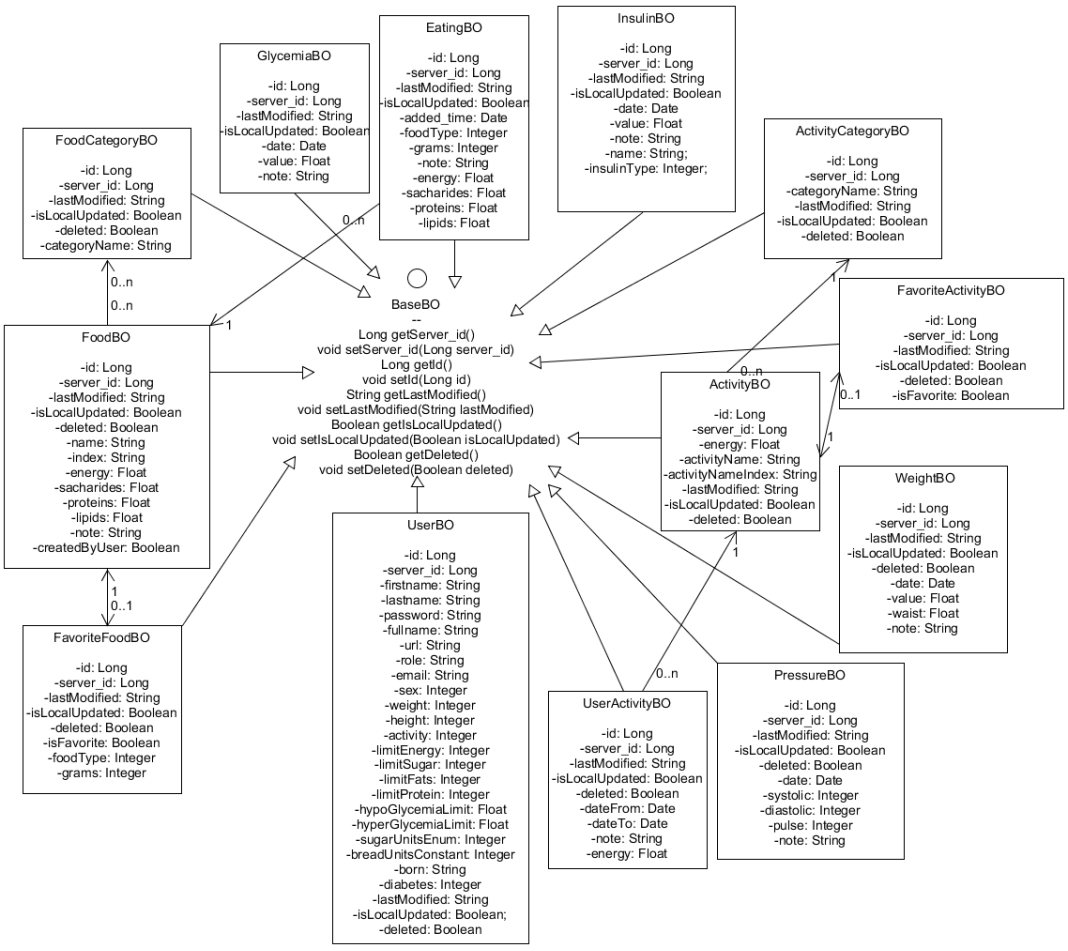
- První věcí je uložení stavu ViewModelu a znovunačtení ViewModelu z uloženého stavu. Tato vlastnost je nezbytná, aby se neztrácel obsah, jenž uživatel vyplní do formuláře a poté rotuje

obrazovku. Defaultní chování Androidu způsobuje zničení všeho a znovulazožení při přetočení obrazovky. Ke stejnému efektu dojde např. i při minimalizaci aplikace.

- Druhou věcí je validace formulářů. Třída BaseViewModel v sobě má metodu public static boolean validate(BaseUIViewModel[] validatedModels)`{...}`. Tato metoda dostane list objektů, které se umí samy zvalidovat. Metoda všechny objekty zvaliduje a vrátí výsledek, který říká, zda jsou všechny objekty validní.

8.2 Databáze

Jako databázi na klientovi používám objektovou databázi Realm. Tato databáze svou strukturou plně neodpovídá databázi na serveru. Při návrhu jsem se rozhodl, že nebudu plně podporovat všechnu funkcionalitu, kterou nabízí rozhraní serveru. Díky tomu, že lokální databáze může obsahovat pouze jednoho uživatele a obsahuje data pouze z jedné jazykové mutace, mohla být struktura lokální databáze opět o něco zjednodušena. Kompletní schéma databáze v mobilní aplikaci lze vidět na obrázku č. 8.2.



Obrázek 8.2. Schéma databáze aplikace

8.2.1 Problémy s Realmem

Realm patří mezi mladé, stále se vyvíjející technologie. Není to tedy léty prověřená odladěná technologie. Z vlastní zkušenosti mohu uvést např. spadnutí aplikace poté co velikost dazabáze dosáhla 1 GB. Tato enormní přetíženost byla způsobena velkým množstvímm transakcí, které

se nestíhaly zapsat do databáze. Při velkém počtu malých transakcí je nutné tyto transakce shlukovat. Shlukování transakcí ovšem přináší další problém, selže-li jedna transakce, jež je součástí velké transakce, selže celá tato velká transakce a nedojde k žádným změnám v databázi.

8.3 Synchronizace dat mezi aplikací a serverem

Před začátkem mé implementace server, napsaný Václavem Dobešem, poskytoval REST api, které umožňovalo CRUD operace nad základními entitami. Tato implementace by umožnila vytvořit aplikaci, která by si neukládala žádná data a všechny změny v datech by se ihned provolávaly na server. Mým cílem bylo vytvořit aplikaci fungující i v offline režimu, a proto jsem musel původní server pozměnit.

8.3.1 Stažení všech dat do aplikace

Při zapnutí, popřípadě po přihlášení, se aplikace pokusí stáhnout všechna data, která na serveru přibyla od té doby, kdy se aplikaci naposledy podařilo synchronizovat. Abych docílil toho, že mi ze serveru nepřijdou všechna data, která tam jsou uložena, přidal jsem ke každé databázové entitě na serveru časovou známku, která určuje, kdy naposledy došlo k úpravě dané entity.

Když se poté serveru ptám např. na všechny konzumace, přidám do dotazu časovou známku, která udává čas poslední úspěšné synchronizace. Jako výsledek dostanu konzumace, vytvořené nebo upravené od poslední synchronizace. Tento přístup hodně ušetří síťový přenos např. při práci s jídlem, v prvním běhu synchronizace server vrátí něco kolem 6000 záznamů, v dalších bězích dostávám typicky 0 a nebo maximálně jednotky záznamů.

Když ze serveru přijdou data, která už jsou uložena v databázi telefonu, porovnávám časové známky dat v databázi a dat co přišla ze serveru. Data v lokální databázi přepíšu daty ze serveru jen tehdy, mají-li data ze serveru větší časovou známku.

8.3.2 Vymazání dat z databáze

Vymazání dat z databáze bylo původně na serveru uděláno tak, že server smazal danou entitu z databáze. Toto nešlo zachovat. Jeden uživatel může mít totiž více zařízení, na kterých bude aplikace nainstalována, když poté z jedné aplikace smaže např. konzumaci, nikdy se do druhé aplikace nedostane informace o tom, že daná konzumace byla smazána. Proto jsem na serveru opět upravil databázi, ke všem entitám jsem přidal příznak udávající, zda je daný záznam smazaný či nikoliv.

Když se poté aplikace zeptá serveru na všechny konzumace, vrátí server i ty konzumace, které byly smazány. S entitami v lokální databázi označenými jako smazané, se dále nepracuje, bylo by možné je smazat z lokální databáze.

8.3.3 Update databáze na serveru

Jestliže uživatel upraví např. konzumaci, tak poté v lokální databázi u té určité upravené konzumace se v kolonce isUpdated objeví informace, že daný záznam byl upraven.

Po úpravě nebo vytvoření záznamu se v aplikaci na pozadí spouští úloha provolávající update dat na server. V první fázi se z lokální mobilní databáze vytáhnou všechny záznamy, které mají nastavený příznak isUpdated. Pro tyto záznamy se následně volají vzdálené úlohy na serveru, jež mají za úkol updatovat dané záznamy v databázi serveru.

8.3.4 Vytvoření nových záznamů

Při vytváření nových záznamů jsem narazil na jeden problém jehož řešení zde popíšu. Každý záznam v lokální i v serverové databázi má svůj syntetický primární klíč. Pro dva různé databázové záznamy stejného entitního typu platí, že se jejich primární klíče nesmí nikdy rovnat.

Když v lokální databázi založím nový záznam, musím mu při jeho založení přidělit jedinečný syntetický klíč (celé číslo). V aplikaci můžu zaručit, že daný syntetický klíč bude jedinečný v rámci aplikace, nemůžu už však zaručit, že daný klíč bude jedinečný i v rámci databáze na serveru. Mohlo by se mi tedy později stát, že až se budu snažit daný záznam založit v databázi na serveru, už tam bude nějaký jiný záznam se stejným primárním klíčem. Jsou v podstatě dvě možnosti jak se vyhnout těmto kolizím.

1. Změnit typ primárního klíče z celého čísla na textový řetězec a generovat tyto klíče na straně mobilní aplikace náhodně např. pomocí UUID.randomUUID(). Toto řešení zaručí, že s velkou pravděpodobností ke kolizi nikdy nedojde.
2. Nechat server generovat primární klíče. V aplikaci používat vlastní sadu primárních klíčů, které se ovšem nebudou žádným způsobem posílat na server. Při založení entity v lokální databázi ji přidělíme lokální primární klíč. Když se poté později entita založí v databázi na serveru, server zpátky pošle primární klíč, jenž této entitě přidělil. Tento klíč si poznamenáme do dané entity v lokální databázi. Pro další úpravy entity v databázi na serveru použijeme primární klíč, který nám vrátil server.

V implementaci aplikace jsem použil druhou možnost. U každé databázové entity si vedu jeden lokální primární klíč a jeden klíč, který slouží jako primární klíč na serveru.

Samotné provolávání vytvoření nového záznamu na server se chová obdobně jako provolávání updatu záznamu (viz pokapitola č. 8.3.3).

8.3.5 Závěr

Díky úpravám na serveru a implementaci synchronizace v aplikaci jsem docílil toho, že na vytváření záznamů v aplikaci není potřeba internetové připojení. Vytvořené záznamy se nasynchronizují na server, až bude připojení dostupné. Aplikaci by mělo být bez problému možné provozovat na více zařízeních přihlášených pod stejným účtem. Záznamy zadané na jednom zařízení se nasynchronizují na druhé zařízení. Toto chování jsem otestoval mnoha manuálními testy.

8.4 Použité knihovny třetích stran

Při implementování aplikace jsem mnohdy řešil problémy, které už někdo vyřešil přede mnou. Řešení těchto problémů jsou publikovány ve formě knihovny. Tato kapitola bude obsahovat výčet zajímavých knihoven, které jsem použil při implementaci.

8.4.1 RetroLambda

Jedná se o plugin do buildovacího nástroje Gradle. Tento plugin umožňuje používat některé konstrukty z Javy 1.8 i v Jave 1.5, 1.6 a nebo 1.7. Používat nejnovější javu 1.8 totiž není při psaní androidí aplikace v současné době možné. Při použití javy 1.8 musíme nastavit minimální Api level na 24, tímto krokem znemožníme využívání aplikace pro naprostou většinu uživatelů se strašími verzemi Androidu.

Retrolambda neumožňuje používat všechny nové věci z javy 1.8, umožňuje používat pouze lambda výrazy a reference na metody. Pomocí lambda výrazu lze některé části kódu výrazně zkrátit. [29]

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.0%
4.1.x	Jelly Bean	16	3.7%
4.2.x		17	5.4%
4.3		18	1.5%
4.4	KitKat	19	20.8%
5.0	Lollipop	21	9.4%
5.1		22	23.1%
6.0	Marshmallow	23	31.3%
7.0	Nougat	24	2.4%
7.1		25	0.4%

Data collected during a 7-day period ending on March 6, 2017.
Any versions with less than 0.1% distribution are not shown.

Obrázek 8.3. Relativní počet uživatelů používající danou verzi Androidu, zdroj: [30].

```
view.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                doSomething();
            }
        });
```

Předešlý kód může být s použitím lambda výrazu zapsán následovně.

```
 view.setOnClickListener( (vieww) -> doSomething());
```

8.4.2 Retrofit 2

Jedná se knihovnu, která podstatně zjednodušuje provolávání serverového api. Stačí si nadefinovat oanotované interfacy, které reprezentují serverová volání. [31]

Následuje ukázka implemetace interface a jeho metody pro stažení všech kategorií jídla.

```
public interface FoodApiClient {

    @GET("rest/food/category")
    Call<FoodCategoryListsDto> getAllFoodCategories(
     @Query("lastModified") String lastModified);

    ...
 }
```

Instanci FoodApiClienta, která nám umožní provolávat metodu getAllFoodCategories, získáme následujícím způsobem.

```
Retrofit retrofit = new Retrofit.Builder()
    .baseUrl("https://diaapp-vsas.rhcloud.com/")
```

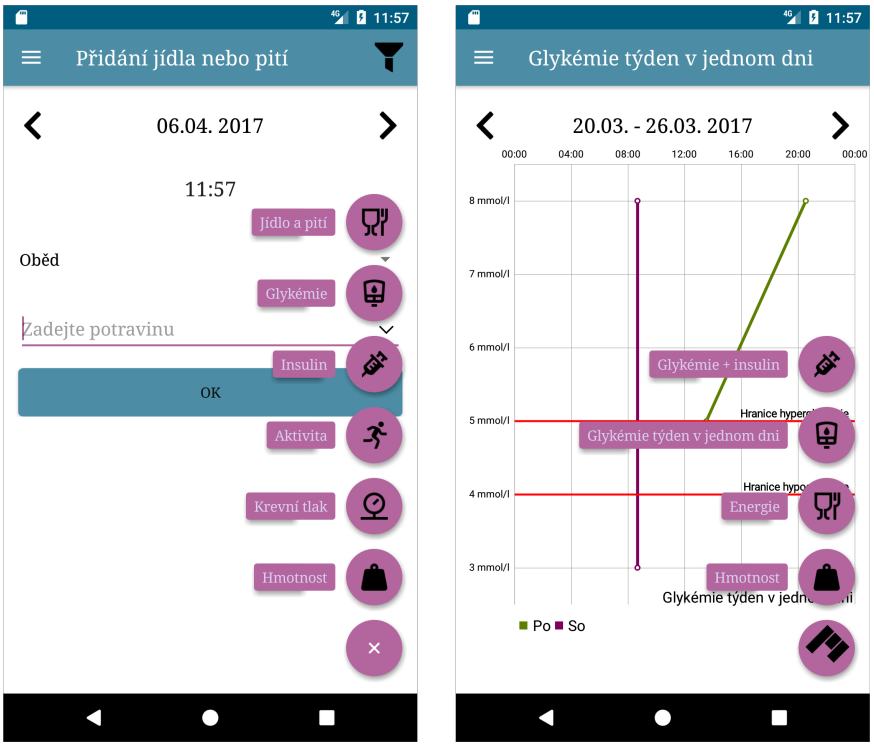
```
    .build();

FoodApiClient foodApiClient = retrofit.create(FoodApiClient.class);
```

8.4.3 FloatingActionButton

Tuto komponentu jsem v aplikaci použil pro navigaci mezi jednotlivými obrazovkami sloužícími pro přidávání záznamů a pro navigaci mezi obrazovkami, které jsou určeny k prohlížení grafů. Díky využití této komponenty jsem nemusel implementovat obrazovku, jež by sloužila jako rozcestník mezi obrazovkami pro přidávání záznamů. A rovněž jsem ušetřil jednu obrazovku, která by tvořila rozcestník mezi obrazovkami s grafy.

Použitím FloatingActionButtonu se navigace v aplikaci značně zjednodušila. [32]



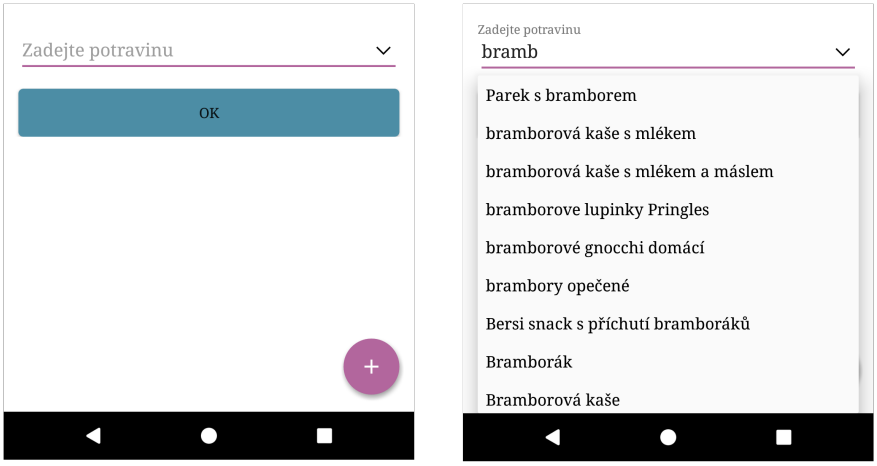
Obrázek 8.4. Ukázka použití FloatingActionButton

8.4.4 EditSpinner

Tuto komponentu jsem využil pro výběr jídla a aktivity. Bohužel mi tato knihovna nevyhovovala v té podobě v jaké ji autor vytvořil a proto jsem si ji pro své účely v aplikaci upravil.

Použitím této komponenty jsem se vyhnul tvorbě extra obrazovky pro vyhledávání jídla. Tato komponenta funguje ve zkratce tak, že podle zadaného řetězce od uživatele se v její drop-down nabídce objevuje seznam položek, z kterých si uživatel může vybrat hledanou položku. Aby se položka objevila v daném drop-down seznamu, musí její jméno zbavené háčků a čárek obsahovat všechna slova, která uživatel zadal (opět zbavená háčků a čárek).

Podobnou funkcionalitu má androidí komponenta zvaná AutoCompleteTextView, tuto komponentu jsem nepoužil, protože se nedokázala vyrovnat s českou diakritikou. Kdyby do této komponenty uživatel zadal např. mleko, nenašlo by mu to mléko. [33]



Obrázek 8.5. Ukázka použití EditSpinneru

8.4.5 HockeyApp

Jedná se o systém, který pomáhá s vývojem mobilních aplikací. Pro přidání základní podpory HockeyApp do androidí aplikace je nutné přidat do Gradlu závislost na knihovně

```
net.hockeyapp.android:HockeySDK
```

a přidat HockeyAppId do manifestu aplikace. HockeyAppId lze získat vytvořením uživatelského účtu na `https://rink.hockeyapp.net`, kde vyplníme informace o aplikaci, kterou budeme spravovat.

V aplikaci poté lze volat příkaz CrashManager.register(this). Ten způsobí, že aplikace bude při svém pádu zaznamenávat logy. Při dalším startu aplikace bude uživateli nabídnuto, aby log z pádu odeslal na server HockeyApp. Správci aplikace bude doručena emailová notifikace, že aplikace někomu spadla. Správce si může inkriminované pádové logy vyhledat, opravit kód aplikace, tak aby k danému pádu již nemohlo dojít.

Kromě této funkcionality HockeyApp nabízí prostředí napomáhající uživatelskému testování. Vývojář aplikace může nahrát build aplikace na HockeyApp a může do projektu přidat testery. Testerům chodí při vydání nové aplikace emailová notifikace o nové verzi. Testeři si přes link mohou tuto novou verzi stáhnout a začít testovat. [34]

HockeyApp má i další funkcionality, ty jsem ovšem ve své práci nevyužíval, např. sledování používání aplikace.

8.4.6 ExpandableRecyclerView

Tuto komponentu jsem použil na zobrazení přehledu dne. V přehledu dne je uživateli umožněno zobrazovat a schovávat záznamy, které paří k jednotlivým částem dne.

V Androidu existuje nativní komponenta, která tuto funkcionalitu umožňuje, jmenuje se ExpandableListView. Třída ListView, od které ExpandableListView dědí, se v nově psaných aplikacích nahrazuje pomocí RecyclerView. Práce s RecyclerView je jednodušší než s ListView. Použitím RecyclerView máme zajištěno, že dochází k tzv. recyklaci grafických prvků. Znamená to, že když např. v RecyclerView zobrazujeme 1 milion prvků, nevytvoří se kvůli tomu 1 milion grafických objektů, vytvoří se jich např. jen 10 a ty se používají dokola. ListView recyklaci umožňuje také, musí se ale explicitně naimplementovat. RecyclerView rovněž umožňuje jednoduše dělat animací přidávání a odebírání prvků. Při použití Listview je implementace animace přidání a odebrání prvků na programátorovi.

V Androidu ovšem neexistuje žádná nativní komponenta ExpandableRecyclerView, proto jsem si zvolil ExpandableRecyclerView z knihovny zvěřejněné na Githubu. [35] Tato knihovna

Obrázek 8.6. Ukázka použití ExpandableRecyclerView

má podobnou funkcionalitu jako ExpandableListView, ovšem má podobné rozhraní a vlastnosti jako RecyclerView.

8.4.7 MPAndroidChart

Jedná se o knihovnu, která umožňuje zobrazování grafů v Androidu. V současné době podporuje až 8 typů grafů. Umí reagovat na uživatelská gesta, animovat zobrazování dat, zobrazovat grafy s odlišnými typy os. [36] Jak vypadají grafy vykreslené pomocí této knihovny, lze vidět na obrázku č. 8.4. Klíčové vlastnosti, kvůli kterým jsem si tuto knihovnu vybral, jsou: pěkný vzhled grafů a možná integrace s Realmovou databází, dokumentace s ukázkami.

Tato grafová knihovna nabízí integraci s Realmovou databází, plánoval jsem využít této vlastnosti, a tím zjednodušit práci s grafy. Bohužel této vlastnosti nakonec nešlo využít. Jestliže chci do grafu pomocí této knihovny vykreslit bod, poskytuje api jen jednu možnost jak tento bod vykreslit, musím zadat jeho souřadnice X a Y ve formě čísla v datovém typu float. V souřadnici Y není žádný problém, skoro všechny hodnoty, jako je např. glykémie nebo energie mám uloženy v tomto typu. Problém je zde s osou X, v databázi jsou v současné chvíli časy přidání jednotlivých položek uloženy ve formě objektu Date, ten v sobě obsahuje počet milisekund od 1. 1. 1970 v datovém formátu long. Problém zde nastane, když bych se pokusil převést toto číslo ve formátu long na float. Datový typ long pro svůj zápis využívá 64 bitů, zatímco float využívá 32 bitů. Převodem takto vysokého longového čísla na číslo ve formátu float by zaručeně nastaly chyby.

Konverzi času z long do float jsem nakonec vyřešil tak, že vezmu nejmenší zobrazovaný čas, ten prohlásím za výchozí bod s hodnotou 0 na ose X. Souřadnice X pro další body se spočítá jako rozdíl v hodinách mezi časem prvního bodu a časem zobrazovaného bodu. Takto dopočítávanou souřadnici X ovšem nelze udržovat v databázi, z tohoto důvodu jsem nakonec nepoužil integraci s Realmem, kterou knihovna MPAndroidChart nabízí.

Při testování aplikace jsem došel k názoru, že tato knihovna není moc stabilní a obsahuje v sobě chyby. Dvakrát se mi stalo, že aplikace začala padat okamžitě po spuštění v důsledku chyby, která vznikla v této knihovně. První pád jsem vyřešil úpravou kódu volání knihovny tak, aby k danému problému nedocházelo. U druhého pádu jsem byl nucen vypnout animování zobrazovaného grafu. Studiem kódu knihovny, jsem došel k závěru, že knihovna obsahuje chybu, tuto chybu jsem reportoval do issue trackeru dané knihovny `https://github.com/PhilJay/MPAndroidChart/issues/3008`.

První a druhý pád aplikace způsobený knihovnou MPAndroidChart, měly spolu společné to, že se projevovaly jen za určitých okolností, když už se ale objevily, aplikace začala padat instantně, což by v ostrém provozu nejspíše způsobilo odinstalování aplikace uživateli.

Dalším problémem, na který jsem v souvislosti s touto knihovnou narazil, je zvýrazňování hodnot, jež uživatel vybere tapnutím v grafu. Občas se nějaké hodnoty nepodaří vybrat a nebo se nepodaří zobrazit jejich popisek.

Tato knihovna přinesla do projektu spoustu problémů. Jestli dojde v budoucnu k přidání dalších grafů, popřípadě k úpravě stávajících, bylo by dobré tuto knihovnu vyměnit za jinou.

8.5 Přizpůsobení pro různé obrazovky

Androidí zařízení mají různé velikosti a rozlišení obrazovek. Pro správné zobrazení na jednotlivých zařízeních je nezbytné v aplikaci udělat jisté kroky.

8.5.1 Hustota displeje

Hustota displeje se udává v jednotce dpi (dots per inch). Tato hodnota udává kolik bodů je zobrazeno na jednom palci displeje. Pro zjednodušení Android zavádí šest skupin, do kterých rozděluje displeje podle jejich hustoty: low, medium, high, extra-high, extra-extra-high a extra-extra-extra-high.

Displej telefonu může patřit do jedné z těchto skupin. Jestliže v aplikaci máme nějaké obrázky, je důležité tyto obrázky poskytnout pro všechny tyto hustoty displeje. Telefon si pak podle své hustoty displeje vybere patřičný obrázek, který bude zobrazovat. Jestliže telefonu neposkytneme obrázky pro všechny tyto hustoty, je schopný si pomocí škálování obrázek upravit tak, aby ho mohl správně zobrazit. Poskytneme-li ovšem obrázky pouze pro malé hustoty displejů, na zařízení s velkou hustotou bude obrázek rozpixelovaný. Použijeme-li pouze obrázky pro velkou hustotu, poté bude mít zařízení s malou hustotou hodně práce se změnou velikosti obrázků. [37]

Podporu různých hustot displeje jsem vyřešil tak, že jsem každý obrázek do aplikace nahrál pro každou hustotu displeje zvlášť.

8.5.2 Density-independent pixel (dp)

Jedná se o virtuální pixelovou jednotku používanou pro pozicování prvků na obrazovce. Jeden dp je rovný jednomu fyzickému pixelu na obrazovce s hustotou 160dpi. Pro přepočet dp na fyzické pixely se dá použít následující vzorec: px = dp * (dpi / 160). Na obrazovce která má hustotu 240 dpi bude jeden dp zabírat 1.5 fyzických pixelů. [37]

Definování rozměrů grafických prvků pomocí dp zaručí, že dané prvky budou mít stejnou absolutní velikosti na všech zařízeních.

8.5.3 Velikost obrazovky

Velikost obrazovek se rozděluje v Androidu do následujících kategorií:

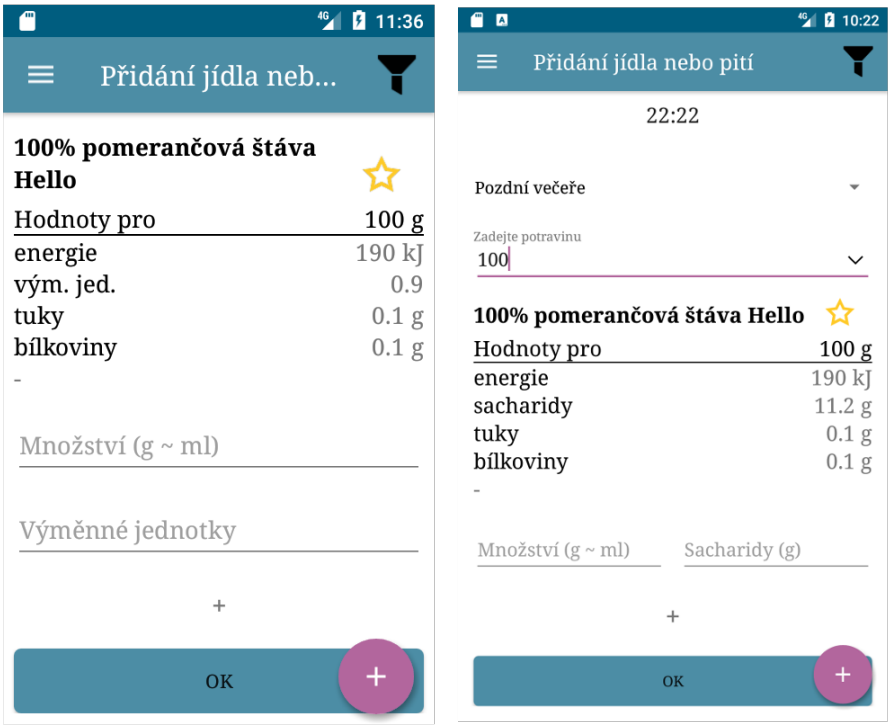
```
xlarge screens - nejméně 960dp x 720dp
large screens - nejméně 640dp x 480dp
normal screens - nejméně 470dp x 320dp
```

```
small screens - nejméně 426dp x 320dp
```

Pro optimalizaci aplikace na různé velikosti obrazovek lze v aplikaci napsat specifické layouty pro různé velikosti obrazovek. Aplikaci Dia Dieta jsem psal primárně pro rozlišení 480dp * 360dp. Tento rozměr odpovídá na displeji s hustotou xx-hdpi následujícímu rozlišení obrazovky: 1920px * 1080px, což je Full-HD rozlišení. Na větších displejích se obsah zobrazí korektně, obsah obrazovek bude však vzdušnější. Grafické prvky okolo sebe budou mít více místa, toto chování lze simulovat v telefonu zobrazením aplikace v landscape módu.

Pro displeje s menší obrazovkou jsem zmenšil velikost písma a odsazení, i přesto se mi nepodařilo vždy správně zobrazit všechny grafické prvky. Prvek pro zadávání hmotnosti jídla toho byl důkazem. Texty `Množství (g\~ml)` a `Výměnné jednotky` se tam stále nevešly. Proto jsem přešel k následujícím úpravám. Na velkém displeji budou prvky vedle sebe, na malém displeji budou pod sebou, takže se tam dané texty pohodlně vejdou.

Dvojice obrázků zachycuje, jak vypadá formulář pro zadávání konzumace na malém a velkém displeji.



Obrázek 8.7. Přizpůsobení na malé displeje

Kapitola 9
Popis hlavních funkcionalit

Tato sekce se zabývá hotovou aplikací. Ukazuje konkrétní podobu a funkcionalitu obrazovek.

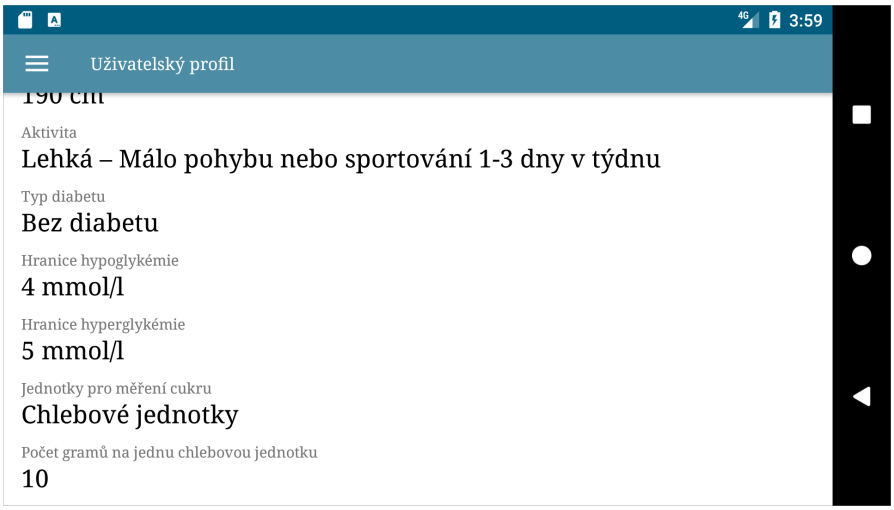
9.1 Hlavní obrazovka

Hlavní obrazovka je zobrazena uživateli při spuštění, jestliže není přihlášený do aplikace. Obrazovka obsahuje fotografii různých diabetických pomůcek, sadu tlačítek. První tlačítko slouží pro přihlášení do aplikace, druhé pro vytvoření nového uživatelského účtu. Poslední tlačítko vpustí do aplikace nepřihlášeného uživatele. Hlavní obrazovku lze vidět na obrázku č. 9.1.

Obrázek 9.1. Hlavní obrazovka

9.2 Uživatelský profil

V této sekci aplikace se zobrazují základní informace o uživateli. Většinu těchto informací zadává uživatel při registraci do aplikace. Sekce umožňuje tyto informace změnit. Kromě informací zadaných při přihlášení je zde možnost předefinovat si hranice hypoglykémie a hyperglykémie, přepínat si zobrazení množství sacharidů mezi gramy a výměnnými jednotkami a nastavit si kolik gramů sacharidů váží jedna výměnná jednotka, jak ukazuje obrázek č. 9.2.



Obrázek 9.2. Uživatelský profil

9.3 Zadávání konzumace

Jedním ze stavebních kamenů této aplikace je zadávání nové konzumace. Celá použitelnost aplikace doslova stojí na tom, jak dobře se bude konzumace zadávat. Není proto divu, že tato obrazovka byla několikrát upravena, než se ustálila v dnešní podobě.

V aplikaci Mobiab se vyhledávání jídla řešilo poměrně složitým způsobem:

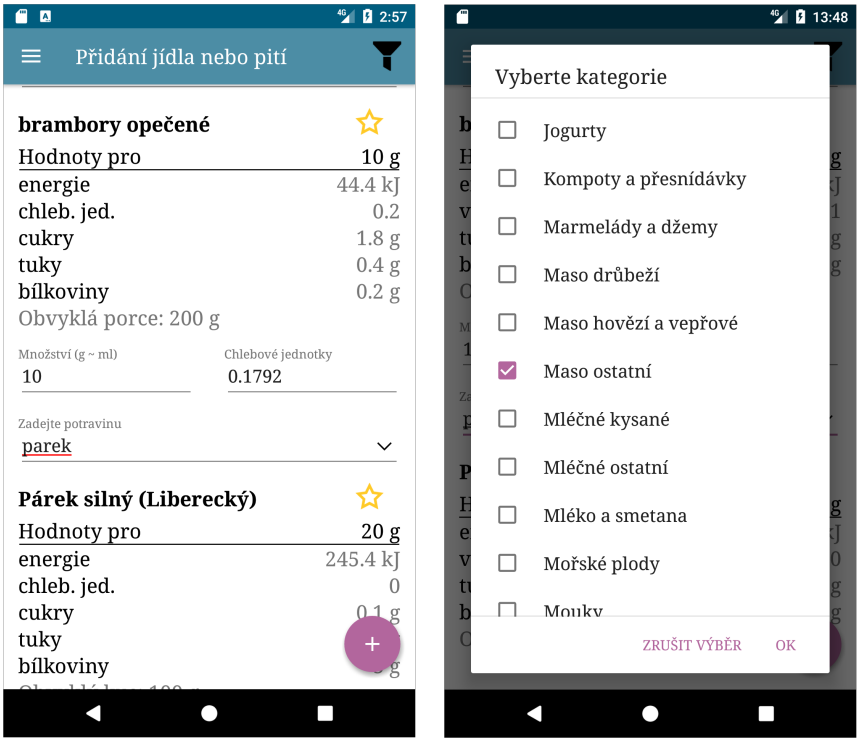
1. Uživatel mohl kliknout na tlačítko `Moje jídla`, poté se dostal na obrazovku s přehledem všech jeho jídel, z kterých si jedno mohl vybrat. Na této obrazovce se nedalo hledat pomocí názvu potraviny.
2. Možnost Procházet potraviny uživatele vedla na obrazovku se všemi kategoriemi jídla. Uživatel si zde mohl vybrat kategorii, ve které chce hledat a poté byl přesměrován na obrazovku, kde hledal pomocí názvu potraviny.
3. Další možností bylo kliknout na Oblíbená jídla. Obrazovka, kam byl uživatel přesměrován, se podobala té z bodu 1). Byly na ní však zobrazená jídla, která si uživatel přidal do oblíbených.
4. Uživatel také mohl zadat název jídla už na hlavní obrazovce přidávání a stisknout tlačítko `Vyhledat`, poté se dostal na obrazovku s výběrem všech jídel, které odpovídaly jeho zadání. Na této obrazovce uživatel už nemohl měnit zadání, mohl měnit pouze kategorie, ve kterých potravinu hledá.

Toto řešení vyhledávání mi připadalo hodně složité, vymyslel jsem si tedy zadávání, které je z mého pohledu jednodušší. Pro celé zadávání mi stačí pouze jedna obrazovka. Ústředním prvkem obrazovky pro přidávání konzumace je komponenta pro vyhledávání jídla z databáze. Komponentu v akci lze vidět na obrázku č. 8.5. Tato komponenta v reálném čase filtruje jídla, na základě uživatelského vstupu.

Pro možnosti dalšího filtrování jsem do hlavičky obrazovky pro přidávání konzumace umístil ikonu filtru. Po jejím stisknutí se otevře dialog pro volbu kategorií jídel, ve kterých bude komponenta pro vyhledávání jídel vyhledávat, jak se možné vidět na obrázku 9.3.

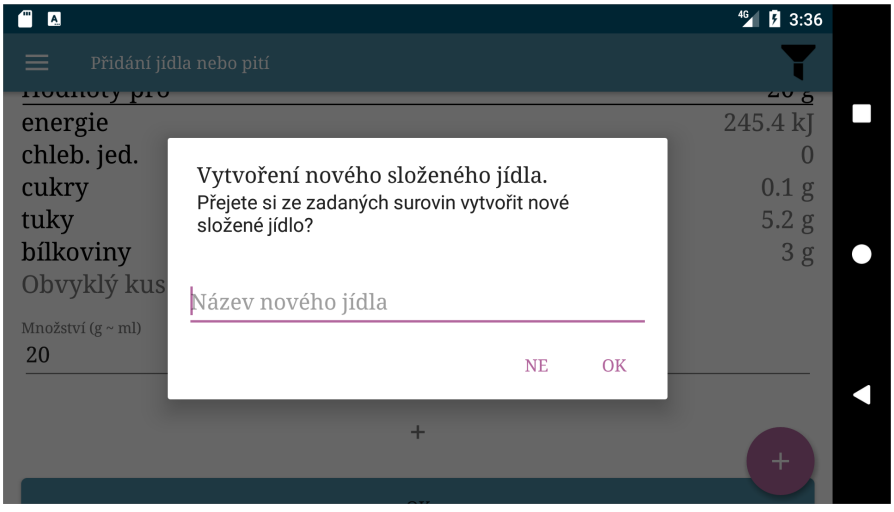
Předpokládám že uživatel bude pracovat nejčastěji s jídly, která označil jako oblíbená a nebo s jídly, která sám osobně vytvořil. Proto jsem pořadí nalezených jídel v komponentě pro vyhledávání upravil tak, že nejdříve se zobrazují jídla přidaná do oblíbených (taková jídla jsou v přehledu označena žlutou hvězdičkou), poté se zobrazují jídla vytvořená uživatelem a nakonec se zobrazí zbytek.

Poté co vybereme jídlo z nabídky komponenty pro výběr, zobrazí se detailní informace o jídle s možností zadat kolik gramů dané potraviny uživatel snědl, jak je vidět na obrázku č. 9.3.



Obrázek 9.3. Přidání konzumace

Při zadávání konzumace je možné zadat více potravin najednou. U každé vybrané potraviny lze poté specifikovat množství. Jestliže uživatel zadá více potravin v jedné konzumaci, je mu poté při potvrzování formuláře nabídnuto, zda chce ze zadaných potravin vytvořit nové složené jídlo (viz obázek č. 9.4) a nebo chce pro každou zadanou potravinu vytvořit samostatnou konzumaci. Jestliže se uživatel rozhodne pro první možnost, tedy pro vytvoření nového složeného jídla, toto nové složené jídlo se vytvoří a následně se vytvoří konzumace tohoto jídla.



Obrázek 9.4. Vytvoření složeného jídla

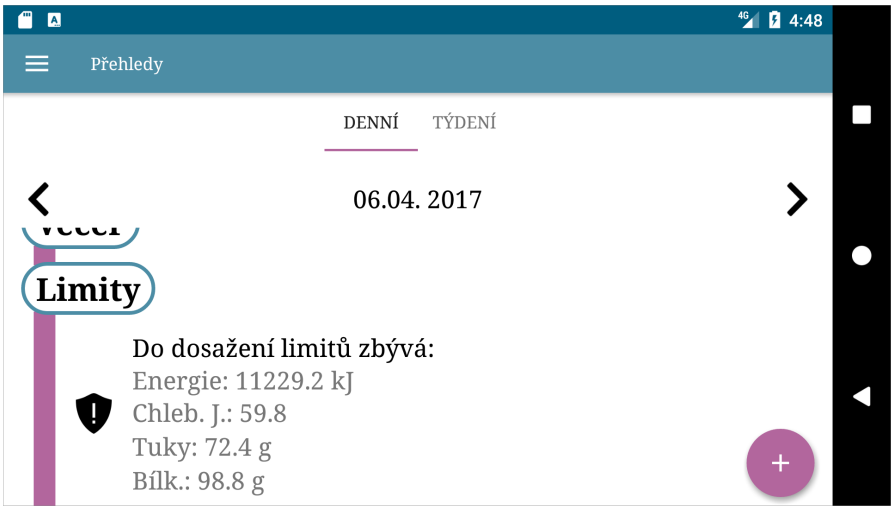
9.4 Zadávání aktivity

Vyhledávání a následné zadávání aktivity jsem řešil analogickým způsobem jako zadávání konzumace, s tím rozdílem že nelze zadávat více aktivit najednou ani tvořit žádné složené aktivity.

9.5 Přehledy zadaných záznamů

9.5.1 Denní přehled

Jak vypadá denní přehled lze vidět na obrázku č. 8.6. Přehled je rozdělen podle denních dob. Každá denní doba má pod sebou záznamy. Jako poslední položka denní doby je zde položka sumy. Položka sumy shrnuje zadané údaje pro danou denní dobu. Denní přehled je zakončen sekcí s limity příjmu. V této sekci může uživatel zjistit, kolik mu zbývá do dosažení limitů a nebo o kolik překročil své limity příjmu pro daný den, jak ukazuje obrázek č. 9.5.

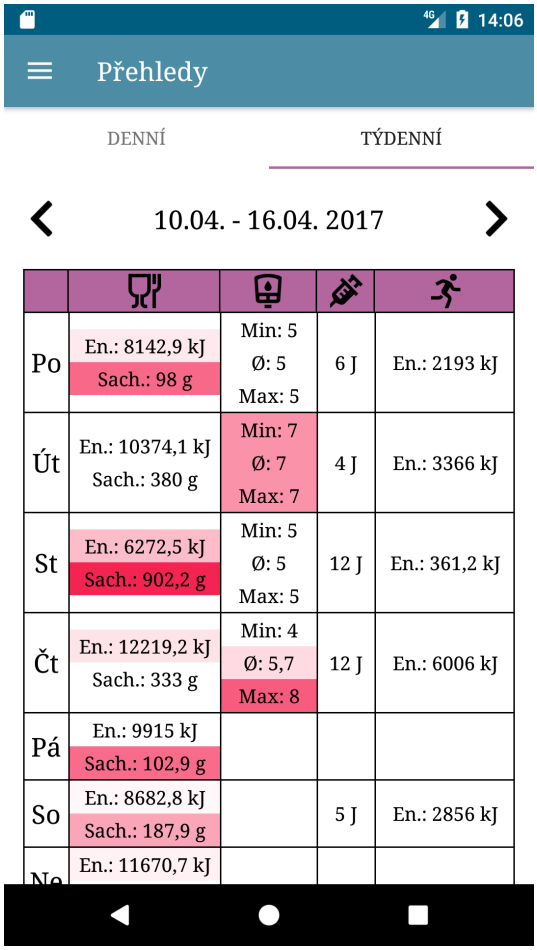


Obrázek 9.5. Denní přehled - limity příjmů

9.5.2 Týdenní přehled

Týdenní sekce sumarizuje energii a sacharidy přijaté z potravy, vykonanou aktivitu a množství vpíchnutého inzulinu. Kromě toho ukazuje nejmenší naměřenou glykémii, nejvyšší naměřenou glykémii a průměrnou hodnotu. Jako poslední řádek v tabulce přehledů jsou zobrazeny průměrné hodnoty přes všechny dny v týdnu, jak lze vidět na obrázku č. 9.6.

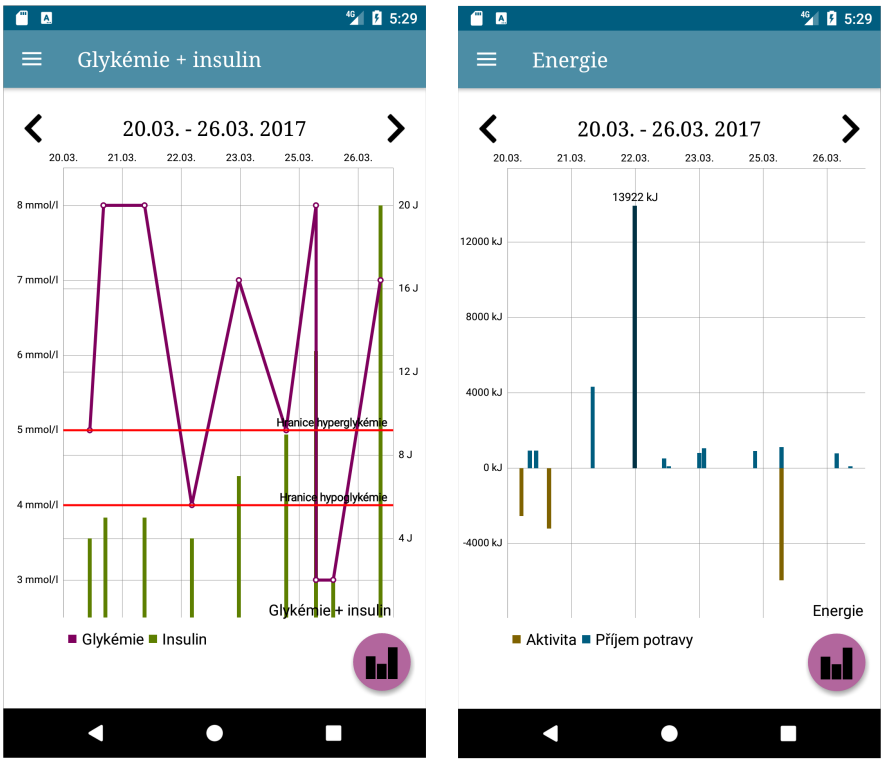
Jestliže je hodnota energie, sacharidů z potravy a nebo glykémie mimo své limity, je poté daná hodnota podbarvena červenou barvou. Čím více daná hodnota porušuje své limity, tím výraznější je její podbarvení.



Obrázek 9.6. Týdenní přehled

9.6 Grafy

V této sekci aplikace si uživatel může prohlížet grafy vytvořené z hodnot, které zadal. K dispozici jsou celkem čtyři grafy. První graf Glykémie + inzulin zobrazuje průběh glykémie a dávky inzulinu během jednoho týdne (viz levý obrázek č. 9.7). Druhý graf ukazuje průběhy glykémií měřených během jednoho týdne, jednotlivé denní průběhy jsou zobrazeny přes sebe. Další graf zobrazuje příjem energie a vykonanou aktivitu během doby jednoho týdne (viz pravý obrázek č. 9.7). Poslední graf ukazuje vývoj uživatelovy hmotnosti za dobu používání aplikace.



Obrázek 9.7. Graf glykémie + inzulin, Graf energie

Kapitola 10
Testování

Tato kapitola zachycuje způsob, kterým bylo provedeno testování aplikace Dia Dieta na participantech. Toto testování si klade za cíl objevit slabiny vytvořené aplikace a navrhnout, jak tyto slabiny eliminovat.

10.1 Screener

Pro účely testování byl napsán tzv. Screener. Tento dotazník má za úkol odfiltrovat pouze relevantní participanty. Nemělo by např. cenu testovat s participanty, kteří nemají a nebo neumí používat mobilní telefon. Cílovou skupinou jsou participanti starší 15 let, kteří vlastní mobilní telefon s Androidem ve verzi alespoň 4.0 a jsou zběhlí v používání mobilních aplikací. Ideální participanti pro toto testování trpí diabetem.

Pro odfiltrování cílové skupiny participantů jsem vytvořil následující dotazník.

1. Je váš věk vyšší než 14 let?
   a) Ano
   b) Ne
2. Vlastníte mobilní telefon s operačním systémem Android ve verzi alespoň 4.0?
   a) Ano
   b) Ne
3. Používáte v mobilním telefonu nějaké aplikace?
   a) Ano
   b) Ne
4. Trpíte nějakým typem onemocnění diabetes?
   a) Ano
   b) Ne

10.1.1 Požadované odpovědi na Screener

Aby byl participant zařazen do testování, požaduji po něm následující odpovědi na Screener:

1. Je váš věk vyšší než 14 let?
   a) Ano
2. Vlastníte mobilní telefon s operačním systémem Android ve verzi alespoň 4.0?
   a) Ano
3. Používáte v mobilním telefonu nějaké aplikace?
   a) Ano
4. Trpíte nějakým typem onemocnění diabetes?
   a) Ano
   b) Ne

10.2 Potestový dotazník

Potestový dotazník má za úkol shrnout výsledky testování, ukázat slabiny aplikace, nasměrovat její další vývoj.

10.2.1 Uzavřené otázky

1. Jak hodnotíte své zkušenosti s mobilním telefonem?
   a) Základní (telefonování a posílání SMS zpráv, používání předem nainstalovaných aplikací)
   b) Pokročilé (instalace a používání nových aplikací)
   c) Expertní
2. Při použití aplikace jsem si vytvořil/a uživatelský účet.
   a) Ano
   b) Ne - aplikaci jsem používal/a bez přihlášení
3. Navigace v mobilní aplikaci mi nedělala problémy.
   a) Rozhodně souhlasím
   b) Souhlasím
   c) Neumím se rozhodnout
   d) Nesouhlasím
   e) Rozhodně nesouhlasím
4. Aplikace se vždy chovala podle mého očekávání.
   a) Rozhodně souhlasím
   b) Souhlasím
   c) Neumím se rozhodnout
   d) Nesouhlasím
   e) Rozhodně nesouhlasím
5. Vkládání zkonzumovaných jídel bylo jednoduché a intuitivní.
   a) Rozhodně souhlasím
   b) Souhlasím
   c) Neumím se rozhodnout
   d) Nesouhlasím
   e) Rozhodně nesouhlasím
6. Využili jste při vyhledávání jídel možnost přidat si jídlo do oblíbených?
   a) Ano
   b) Ne
7. Jak jste využívali možnost přidat více jídel najednou?
   a) Nikdy jsem nepřidával/a více jídel najednou
   b) Přidával/a jsem více jídel najednou, ale nikdy jsem nevytvořil/a složené jídlo
   c) Při zadávání nového jídla jsem využil/a možnosti vytvořit nové složené jídlo
8. Vytvořili jste si své vlastní jídlo?
   a) Ano
   b) Ne
9. Vkládání aktivit bylo jednoduché a intuitivní.
   a) Rozhodně souhlasím
   b) Souhlasím
   c) Neumím se rozhodnout
   d) Nesouhlasím
   e) Rozhodně nesouhlasím
10. Využili jste při vyhledávání aktivit možnost označit si aktivitu jako oblíbenou?
    a) Ano
    b) Ne
11. Využili jste při hledání aktivity a nebo jídla tzv. filtr (ikona nahoře vpravo)?
    a) Ano
    b) Ne

10.2.2 Otevřené otázky

1. Máte zkušenosti s nějakou jinou aplikací zabývající se diabetem popřípadě dietou? Pokud ano, tak jakou?
2. Setkali jste se při instalaci aplikace Dia Dieta s nějakým problémem? Pokud ano, s jakým?
3. Jaký byl největší problém, na který jste při používání narazil/a?
4. Co se vám na aplikaci nejvíce líbilo?
5. Co byste aplikaci naopak vytknul/a?
6. Které funkce aplikace jste nejvíce používal/a?
7. Prostor pro vaši poznámku:

10.3 Způsob testování

Ze znění otázek Screeneru a potestového dotazníku jsem vyrobil Google formuláře. Výhoda použití Google formulářů pro vyplňování dotazníků spočívá zejména v časové úspoře a v tom, že není nutné se s participanty scházet osobně. Zároveň jako hlavní nevýhodu uvedu nízkou míru osobní kontroly nad vyplňováním forlumářů.

Průběh testování byl následující.

1. Participant obdržel odkaz na vyplnění formuláře Screeneru.
2. Vyhodnotil jsem zda participant prošel podmínkami Screeneru.
3. V případě, že participant prošel Screenerem, zaslal jsem mu email s instrukcemi (viz podkapitola č. 10.3.1).
4. V případě, že participant neprošel podmínkami Screeneru, zaslal jsem mu pouze děkovný email (viz podkapitola č. 10.3.2).
5. Vyhodnotil jsem výsledky testování.

10.3.1 Instrukce pro testování

Tato sekce obsahuje text emailu, který byl participantům rozesílán po úspěšném absolvování Screeneru:

Gratulujeme, byl/a jste vybrán/a jako člen testovacího týmu mobilní aplikace Dia Dieta. Aplikace není v současné době distribuována přes Google Play. Musí být tedy stažena z jiného zdroje. Pro stažení aplikace klikněte ve svém mobilním zařízení na následující odkaz:

`https://diaapp-vsas.rhcloud.com/resources/mobileApp/app-release-01.apk`

Po kliknutí na odkaz dojde ke stažení apk souboru do vašeho telefonu. Po stažení tohoto souboru ho zkuste otevřít a danou aplikaci nainstalovat. To se vám pravděpodobně nepovede, telefon zablokuje instalaci aplikace z neznámého zdroje. Telefon vám rovněž umožní přesměrování do nastavení, kde si můžete povolit instalaci aplikací z neznámých zdrojů. Jestliže mi důvěřujete, povolte si instalování aplikací z neznámých zdrojů, aplikaci nainstalujte. Po instalaci si instalování aplikací z neznámých zdrojů znovu zakažte v nastavení telefonu. Jestliže mi nedůvěřujte, ukončete prosím testování a dejte mi vědět o tom, že ukončujete testování. Po instalaci si prosím danou aplikaci projděte, zkuste ji alespoň dva dny používat. Zkuste do ní zadat reálné potraviny, které jste zkonzumovali, reálné sportovní aktivity, popřípadě další položky. Poté prosím vyplňte webový formulář na adrese:

`https://goo.gl/forms/LKXfvmmwmtovclRi1`

Čím déle aplikaci vydržíte používat, než mi odešlete zpětnou vazbu, tím lépe. Prosím vás však abyste na to nezapomněli. Vyplnění potestového dotazníku je velmi důležité, pomůže to vylepšit aplikaci, která by mohla být užitečná pro diabetiky a lidi trpící nadváhou. Bez vyplnění potestového dotazníku je vaše testování bezcenné.

S pozdravem,

Dia Dieta

10.3.2 Poděkování za účast

Tato sekce obsahuje text emailu, jenž byl odesílán participantům, kteří nepřošli Screenerem.

Dobrý den, bohužel nevyhovujete požadavkům na účastníka testování aplikace Dia Dieta. Děkujeme vám za účast v předtestovém dotazníku.

S pozdravem,
Dia Dieta

10.4 Vyhodnocení testování

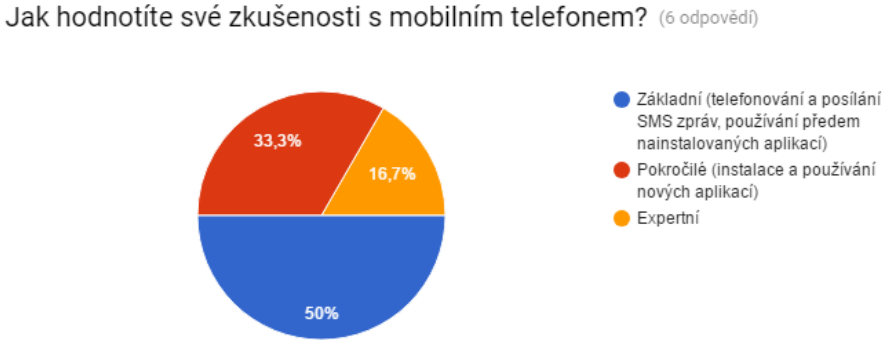
Screener vyplnilo celkem osm participantů. Samotného testování se jich zúčastnilo pouze šest. Jeden participant se nezúčastnil z toho důvodu že neměl k dispozici zařízení s Androidem. Další participant úspěšně prošel Screenerem, nainstaloval si aplikaci, chvíli ji zkoušel testovat, testování však nedokončil. Participantovi v mé aplikaci chyběla jedna klíčová funkcionalita, na kterou byl zvyklý z jiné aplikace. Tento participant používal pro zaznamenávání snědené potravy mobilní aplikaci myFitnessPal. Tato aplikace umožňuje zadávat zkonzumované potraviny pomocí skenování jejich čárového kódu. Vyvinutí této funkcionality by jistě zvýšilo použitelnost aplikace Dia Dieta. Tato funkcionalita bude zařazena do dalších verzí aplikace.

Celým testováním tedy prošlo šest participantů, pouze jeden z nich byl diabetikem a intenzivně do aplikace zadával svoji naměřenou glykémii a dávky inzulinu.

10.4.1 Uzavřené otázky

1. Jak hodnotíte své zkušenosti s mobilním telefonem?

   O úrovni zdatnosti jednotlivých participantů svědčí diagram č. 10.1. Každá zdatnostní skupina měla své reprezentanty. Nejvíce participantů hodnotilo svoje schopnosti jako základní.



Obrázek 10.1. Zkušenosti s mobilním telefonem

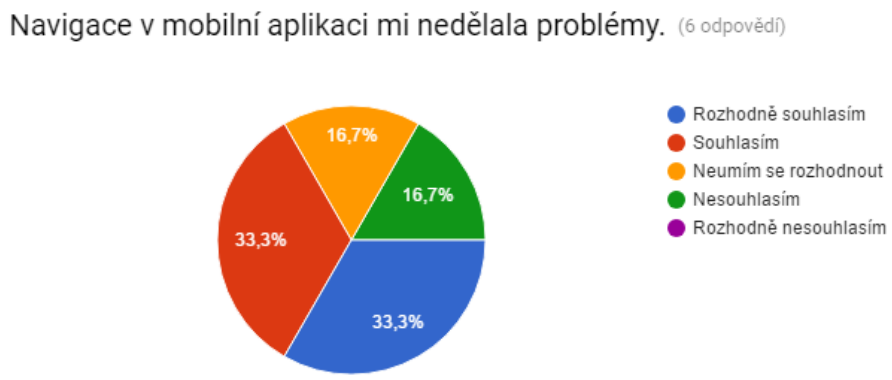
2. Při použití aplikace jsem si vytvořil/a uživatelský účet.

   Ačkoliv participanti nebyli nijak instruováni k tomu, aby si uživatelský účet vytvořili, čtyři participanti z šesti se zaregistrovali.

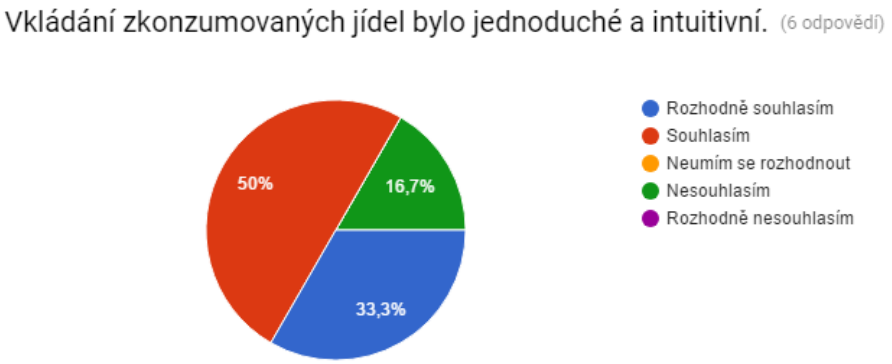
3. Navigace v mobilní aplikaci mi nedělala problémy.
4. Aplikace se vždy chovala podle mého očekávání.

   Tři participanti rozhodně souhlasili, dva participanti souhlasili a jeden se neuměl rozhodnout nad platností tohoto tvrzení.

5. Vkládání zkonzumovaných jídel bylo jednoduché a intuitivní.
6. Využili jste při vyhledávání jídel možnost přidat si jídlo do oblíbených?



Obrázek 10.2. Navigace v aplikaci



Obrázek 10.3. Vkládání jídel

Pouze jeden participant využil možnost přidat si jídlo do oblíbených.

7. Jak jste využívali možnost přidat více jídel najednou?
   Pouze jeden participant využil možnost vytvořit nové složené jídlo. Další dva participanti přidávali do konzumace více jídel najednou, nevytvářeli však nová složená jídla. Zbytek participantů nikdy nepřidával více jídel najednou.

8. Vytvořili jste si své vlastní jídlo?
   Polovina participantů si své jídlo nikdy nevytvořila.

9. Vkládání aktivit bylo jednoduché a intuitivní.
   Tři participanti souhlasili, dva rozhodně souhlasili a jeden se neuměl rozhodnout nad platností tohoto tvrzení.

10. Využili jste při vyhledávání aktivit možnost označit si aktivitu jako oblíbenou?
    Pouze jeden participant využil této možnosti.

11. Využili jste při hledání aktivity a nebo jídla tzv. filtr (ikona nahoře vpravo)?
    Pouze jeden participant využil tohoto filtru.

10.4.2 Otevřené otázky

Tato sekce obsahuje všechny odpovědi participantů, tak jak je napsali, mohou se zde tedy objevovat pravopisné i jiné chyby.

1. Máte zkušenosti s nějakou jinou aplikací zabývající se diabetem popřípadě dietou? Pokud ano, tak jakou?

   Pouze jeden participant měl nějaké zkušenosti s jinou aplikací podobného rázu. Měl zkušenosti s aplikací Kalorické tabulky.

2. Setkali jste se při instalaci aplikace Dia Dieta s nějakým problémem? Pokud ano, s jakým?

   Pouze jeden participant uvedl problém. Jako problém uvedl nutnost povolit instalaci aplikace z neznámých zdrojů. Tento problém se vytratí poté až bude aplikace vydána oficiálně na Google Play.

3. Jaký byl největší problém, na který jste při používání narazil/a?

   a) Vybírání potraviny z rozevíracího seznamu - výběr je nepřehledný, zejména u potravin, které mají název přes 2 a více řádků. Pokud jsem měl oběd až po 13hod, tak mi ho aplikace řadila jako odpolední jídlo, ačkoliv jsem vybral, že to byl oběd.
   b) Zadávání glykémií - pouze celá čísla, nešla mi zadat desetinná čárka pro přesnou hodnotu glykémie.
   c) možnost vkládat jídlo pouze na gramy a ne na kusy, např. jablko...
   d) Při zapnutí mi pokaždé aplikace ukáže hlášku „aplikace přestala pracovat“, ale aplikace běží.
   e) žádný
   f) žádný

4. Co se vám na aplikaci nejvíce líbilo?

   a) svižnost aplikace, bez nutnosti internetového připojení
   b) design, nápad
   c) jednoduchost, grafika
   d) široká databáze potravin, kde s trochou fantazie poskládat jakékoliv jídlo
   e) jednoduchost, uzpůsobené i pro tablety
   f) přehledné energetické grafy

5. Co byste aplikaci naopak vytknul/a?

   a) Přijde mi, že aplikace plýtvá místem na obrazovce - prvky jsou velké a tím pádem se na displeji zobrazuje méně informací.
   b) Nikde není napsáno k čemu aplikace slouží, výpočty jednotek gramů a hodnot na den nejsou nikde vysvětleny, plus jdou změnit takže nechápu jestli si je mám změnit podle sebe nebo nechat vypočtené...
   c) Že má evidentně mnohé zajímavé funkce jako třeba vkládat složené jídlo, ale to jsem nevěděla při testování.
   d) nedostatečné seznámení uživatele s možnostmi aplikace
   e) nic
   f) Mohl by noveho uzivatele podrobneji provest aplikací.

6. Které funkce aplikace jste nejvíce používal/a?

   a) přidávání jídla a pití, přehledy - denní a týdenní
   b) glykemie a grafy
   c) vložit jídlo a aktivitu a grafy
   d) vkládání potravy a aktivit ke hlídání objemu jídelníčku
   e) vkládání jídel
   f) přidat jídlo

7. Prostor pro vaši poznámku:

a) Uvítal bych, pokud by se daly ovládací prvky a text škálovat - pro mě vše působilo příliš velké.
b) Moc pěkná aplikace.

10.4.3 Závěry z testování

Ačkoliv počet participantů byl relativně nízký, dá se z jejich odpovědí alespoň rámcově odhadnout, jak na tom výsledná aplikace je.

Otázky, které měly zhodnotit, jak se participantům s aplikací pracovalo, měly odpovědi většinou v kladné části spektra, pouze jeden participant nesouhlasil.

Možnost přidat si potravinu a nebo aktivitu do oblíbených využil pouze jeden participant. Toto chování participantů by mohlo souviset s tím, co vyšlo najevo v otázce: Co byste aplikaci naopak vytknul/a? Dalším možným vysvětlením je, že přidávání do oblíbených zrychluje práci s aplikací až při dlouhodobém používání, přičemž participanti aplikaci používali pouze krátce. Na základě testování a provozu aplikace Mobiab vyšlo najevo, že tato funkcionalita je uživateli hojně využívána, domnívám se tudíž že při delším používání by participanti tuto funkcionalitu začali používat i v aplikaci Dia Dieta.

Otázka, která zhodnocovala přidávání více jídel najednou, naznačila, že participanti tuto funkcionalitu moc nepoužívali. V jiné části dotazníku vyšlo najevo, že někteří nevěděli, že je to vůbec možné. Databáze jídel obsahuje typicky jednotlivé suroviny, neobsahuje už však hotová jídla, která jsou kombinací jednotlivých surovin. Typická konzumace, kterou uživatel zadává se skládá z více jednotlivých surovin. Z tohoto důvodu je zde naimplementována funkcionalita, která umožňuje zadat více surovin/jídel najednou. Tato funkcionalita zjednodušuje využívání aplikace.

Otázka zjišťující, zda si participanti vytvořili svoje vlastní jídlo ukázala, že zhruba polovina participantů to udělala. Databáze jídel obsahuje zhruba něco přes 6000 jídel. Již při krátkodobém používání se dá zjistit, že některá jídla tam chybí. Chybějící jídlo se dá řešit následujícími způsoby. Buď najdeme podobné jídlo nebo zadáme jednotlivé suroviny, z kterých se jídlo skládá, a nebo vytvoříme úplně nové jídlo. Při testování se ukázalo, že zhruba polovina participantů se vydala třetí cestou a vytvořila úplně nové jídlo. To mě přivádí k možnosti zobrazovat uživatelsky vytvořená jídla všem uživatelům a ne jen pouze jejich autorům. Tím by se mohla hodně rozšířit databáze jídel. Tato myšlenka bude reflektována v kapitole č. 11.

Otázka zabývající filtrováním aktivit a nebo jídel na základě kategorií ukázala, že tato funkcionalita nebyla moc využívána. Tento výsledek si vysvětluji tím, že hledání potravin a aktivit pomocí upraveného EditSpinneru je dostatečně výkonný nástroj na vyhledávání. Participanti většinou nepotřebovali hledat podle kategorií. Další možné vysvětlení opět souvisí s odpovědí na otázku: Co byste aplikaci naopak vytknul/a?

Otázka zabývající se největšími problémy aplikace objevila řadu problémů a námětů na změny. Jeden participant si stěžoval na nepřehledný výběr potravin, v případě že názvy potravin jsou víceřádkové. Tento problém by měl být řešitelný otestováním výběru potravin na různých typech obrazovek, na základě tohoto testování by měly být provedeny patřičné úpravy. Vzhledem k tomu, že si na toto chování stěžoval pouze jeden participant, nepatří tento problém mezi závažné nedostatky. Dalším problémem bylo chování popsané následující výpovědí: „Pokud jsem měl oběd až po 13hod, tak mi ho aplikace řadila jako odpolední jídlo, ačkoliv jsem vybral, že to byl oběd.“ Toto chování je způsobeno tím, že řazení jídel do jednotlivých skupin v přehledu dne probíhá automaticky na základě času přidání záznamu, tudíž to vůbec nesouvisí se zadáním typu jídla uživatelem. Uznávám, že toto je matoucí a v závěru tedy navrhnu změnu.

Další participant si stěžoval na nemožnost zadávat desetinná čísla u glykémie. Možnost zadávat desetinná čísla jsem testoval na svém zařízení. Na mém zařízení se dala desetinná čísla zadávat bez problému. Díky participantově testování je však jasné, že aplikace obsahuje chybu,

která se projevuje pouze na některých zařízeních při určité konfiguraci. Díky tomu, že jsem se snažil jednotlivé zadávací prvky uživatelského prostředí psát modulárně, stačí opravit chybu pouze na jednom místě. S pomocí participanta a screenshotu z jeho telefonu jsem objevil podstatu problému. Participantovi na klávesnici chybělo tlačítko pro desetinnou čárku a nebo tečku. Problém byl způsoben mojí snahou poskytnout českým uživatelům možnost zadávat desetinnou čárku místo desetinné tečky. Problém jsem vyřešil použitím defaultního chování prvku pro zadávání čísel, to má ovšem za následek, že uživatel bude nucen místo desetinné čárky zadávat desetinnou tečku.

Dále tu byla výtka na to, že uživatelé mohli vkládat potraviny pouze v gramech a nikoliv na kusy, případně v dalších alternativních jednotkách. Uvědomuji si, že toto je velkou slabinou v použitelnosti mé aplikace. Přidat funkcionalitu, která by tento problém odbourala bude tedy jedním z prvních úkolů pro rozvoj do budoucna.

Při testování se ukázalo, že jednomu z uživatelů aplikace při spuštění opakovaně padala. Naštěstí pád neshodil celou aplikaci a uživatel mohl pokračovat v testování. Díky integraci HockeyApp jsem byl schopen získat logy z tohoto pádu. Analýzou jsem došel k závěru, že aplikace spadne v knihovně od Googlu při té příležitosti, kdy se snažím animovaně zobrazit tzv. Android Activitu. Použití této animace je podporováno od verze Androidu 21, participant měl verzi Androidu 24. Ve verzích by tedy problém být neměl. Osobně si myslím, že problém je v tom, že participant měl nějakou podivnou verzi Androidu založenou na Androidu 6. Jeho operační systém se jmenoval DOOGEE-X5max_PRO. Tento problém jsem vyřešil vyplnutím zmíněné animace pro všechna zařízení.

Již několikrát zmiňovaná otázka: Co byste aplikaci naopak vytknul/a? přinesla velmi jednoznačnou odpověď, téměř všichni participanti si stěžovali na nedostatečné seznámení uživatele s aplikací. Implementace této funkcionality bude zařazena do dalšího vývoje aplikace. Jeden participant si stěžoval na to, že mu přišly grafické prvky v aplikaci moc veliké, vzhledem k tomu že tato výtka zůstala samotná, nepovažuji tuto stížnost za klíčovou. V budoucnu by bylo dobré design aplikace a velikost prvků probrat s grafikem.

Participanti většinou používali části aplikace, které se zabývají jídlem popřípadě tělesnou aktivitou, jeden participant, používal zadávání glykémie a prohlížení grafů glykémie. V budoucnu by bylo dobré do aplikace přidat Google analytics a s jeho pomocí sledovat, jak uživatelé aplikaci používají. Na základě těchto pozorování pak bude mnohem jednodušší rozhodnout se, která funkcionalita je používána a která v aplikaci jen zabírá místo.

10.4.4 Shrnutí testování

Testování s uživateli odhalilo dva bugy, přičemž ani jeden nezabránil používání aplikace. Participanti většinou aplikaci hodnotili v kladně, zazněla chvála na rychlost a grafický návrh. Našla se však i kritika grafického návrhu. Testování přineslo další podněty pro rozvoj aplikace. Některé podněty byly již známe z analýzy podobných aplikací, nebyly však v Dia Dietě implementovány kvůli nedostatku času. Aplikace je značně rozsáhlá i při vypuštění těchto funkcionalit. Jiné podněty, jako např. nedostatečné seznámení uživatele s aplikací, vyšly najevo až při testování.

Toto testování napomohlo určit směr, kterým by se aplikace Dia Dieta měla v budoucnu vydat.

Kapitola 11
Závěr

Mým úkolem bylo navázat na projekt Mobiab, vytvořit novou aplikaci s podobnou funkcionalitou. Nová aplikace ovšem měla mít oproti Mobiabu nějaké rozdíly (viz podkapitola č. 3.4).

Jedním z hlavních rozdílů byla co největší snaha o jednoduchý user interface. V aplikaci jsem se snažil minimalizovat počet obrazovek. Všechny komponenty pro zadávání jsem psal obecně a využíval je skrz celou aplikaci. Neimplementoval jsem nadbytečnou funkcionalitu, v aplikaci nejsou obrazovky se spoustou tlačítek, které mají rozličné funkce. Každá obrazovka má svůj jasný účel, např. zadat jídlo, upravit aktivitu nebo prohlížet záznamy. Dalším rozdílem je snaha o současný vzhled aplikace podle material designu.

Aplikaci je možné používat offline, což je jeden z mých prvotních plánů. Ačkoliv se to může zdát jako snadné, byl to velice obtížný úkol. Místo jedné databáze člověk musí najednou spravovat databáze dvě a kromě toho se musí potýkat s mapováním objektů, jež se posílají po síti, na objekty obsažené v jednotlivých databázích. Bylo nutné vymyslet jak se zachovat dojde-li ke konfliktům při synchronizaci a implementovat logiku synchronizace.

Dalším předsevzatým bodem bylo rozdělit aplikaci po funkcích a ne po jednotlivých záznamech. Aplikaci mám rozdělenou na část, která se např. zabývá zadáváním záznamů, další část se zabývá zobrazováním grafů, atd. Mobiab byl naproti tomu oddělen po modulech, obsahoval např. modul pro jídlo nebo modul pro glykémii, v těchto modulech se poté odehrávalo vše (zadávání, zobrazování zadaných záznamů, prohlídka grafů atd.).

Posledním bodem bylo napojit aplikaci na nový server psaný v Java EE. To se mi povedlo, nenapojil jsem však novou aplikaci na starou instanci serveru. V kódu serveru jsem udělal změny a server nasadil znovu jinde.

Funkční požadavky vyspecifikované v kapitole č. 5 jsem splnil všechny bez výjimky. Nefunkční požadavky z podkapitoly č. 6.3 jsou taktéž všechny splněny.

Závěrečným testováním s participanty se ověřilo, že aplikace je většinou stabilní. Žádný participant si během testování nestěžoval na podivné chování aplikace. Participanti přijímali aplikaci většinou kladně, oceňovali její rychlost a vzhled. Našlo se pár výhrad a dva bugy, které se projevují na některých zařízeních, ale ani ty nebránily používání aplikace. Během testování se objevila spousta funkcionalit, které by mohly být přidány do dalších verzí aplikace.

Testováním se prokázalo, že nic nebrání vydat první verzi aplikace do obchodu Google Play. Po jejím vydání se uvidí jaký bude mít úspěch, zda bude mít cenu pokračovat v jejím vývoji a nebo se bude jednat o mrtvý projekt.

11.1 Možnosti dalšího vývoje

V této podkapitole je sepsán seznam vylepšení aplikace, která by mohla být implementována v dalších verzích aplikace. Seznam je řazen podle priority od těch nejvíce potřebných věcí až po věci méně významné.

1. Přidat průvodce, který by vysvětloval jak aplikace funguje a jaké jsou její možnosti.
2. Přidat možnost zadávat množství potravin i v jiných jednotkách než v gramech.
3. Odebrat položku typ konzumace (např. oběd, večeře atd.) při zakládání konzumace.
4. Přidat podporu Google analytics.

5. Přidat možnost zadávat potraviny za pomoci skenování jejich čárkových kódů.
6. Vymyslet způsob jak rozšířit počet jídel v databázi, nejlépe za použití uživatelů.
7. Lokalizovat aplikaci a databázi do všech světových jazyků, aplikace i databáze je na to připravena, stačí udělat překlady.
8. Výměnit knihovnu použitou pro genenerování grafů.
9. Přidat možnost exportovat data z aplikace.
10. Důkladně přizpůsobit aplikaci na různé displeje včetně konzulatce návrhu s grafikem.
11. Zpoplatnit některé části aplikace.
12. Přidat do aplikace motivační prvky.

Literatura

[1] Václav Burda. *MPAndroidChart*. https://play.google.com/store/apps/details?id=cz.asleep. 2017-04-16.

[2] bc. VáclavBurda. *Sledování přijmu potravy pomocí mobilní aplikace.* 2012. https://dip.felk.cvut.cz/browse/pdfcache/burdavac_2012bach.pdf. Bakalářská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D.

[3] Václav Burda. *Kompenzace diabetes mellitus pomocí mobilních technologií*. 2014. https://dip.felk.cvut.cz/browse/pdfcache/burdavac_2014dipl.pdf. Diplomová práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D.

[4] Tomáš Musílek. *Návrh výukového modulu pro pacienty trpící cukrovkou.* 2015. Bakalářská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D..

[5] David Slezák. *Sledování kvality spánku pomocí chytrého telefonu.* 2012. Bakalřská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D..

[6] Jiří Mosinger. *Vyhodnocování stavu maniodepresivních pacientů pomocí chytrého telefonu s operačním systémem Android.* 2012. Bakalřská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D..

[7] Ondřej Smrž. *Návrh a tvorba motivačního modulu pro projekt kompenzaci diabetes mellitus.* 2014. Diplomová práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D. .

[8] Věra Okrouhlicová. *Sociální síť pro kompenzaci diabetu.* 2014. Bakalářská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D..

[9] Tomáš Hrstka. *Návrh virtuálního průvodce – avatara pro podporu léčby chronických nemocí.* 2014. Bakalářská práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D..

[10] Václav Dobeš. *Implementace webové aplikace pro kompenzaci diabetes mellitus.* 2016. Diplomová práce. ČVUT v Praze. Vedoucí práce Ing. Daniel Novák, Ph.D. .

[11] YAZIO. *YAZIO*. https://play.google.com/store/apps/details?id=com.yazio.android. 2017-03-30.

[12] InspiredApps (A.L) LTD. *My Diet Coach - Weight Loss*. https://play.google.com/store/apps/details?id=com.dietcoacher.sos. 2017-03-30.

[13] Sirma Medical Systems. *Diabetes:M*. https://play.google.com/store/apps/details?id=com.mydiabetes. 2017-03-30.

[14] Klimaszewski Szymon. *Cukrovka - Glukóza deník*. https://play.google.com/store/apps/details?id=com.szyk.diabetes. 2017-03-30.

[15] SquareMed Software GmbH. *Diabetes Connect*. https://play.google.com/store/apps/details?id=com.squaremed.diabetesconnect.android. 2017-03-30.

[16] IDC. *Smartphone OS Market Share, 2016 Q3*. http://www.idc.com/promo/smartphone-market-share/os. 2017-03-30.

[17] Petr Věžník. *Mobilní aplikace*. http://wiki.knihovna.cz/index.php/Mobilní_aplikace#cite_note-5/. 2017-04-02.

[18] Xamarin. *Introduction to Mobile Development*.
https://developer.xamarin.com/guides/cross-platform/getting_started/. 2017-03-30.

[19] Xamarin. *An Introduction to Xamarin.Forms*.
https://developer.xamarin.com/guides/xamarin-forms/getting-started/introduction-to-xamarin-forms/. 2017-03-30.

[20] QT. *QT*.
https://www.qt.io/. 2017-03-30.

[21] thinkapps. *Cross Platform Mobile Development Tools: Ending the iOS vs. Android Debate*.
http://thinkapps.com/blog/development/develop-for-ios-v-android-cross-platform-tools/. 2017-03-30.

[22] SQlite. *SQlite*.
https://www.sqlite.org/about.html. 2017-04-02.

[23] Google. *Content providers*.
https://developer.android.com/guide/topics/providers/content-providers.html. 2017-04-02.

[24] Google. *Loaders*.
https://developer.android.com/guide/components/loaders.html. 2017-04-02.

[25] Aldo Ziflaj. *5 of the Best Android ORMs*.
https://www.sitepoint.com/5-best-android-orms/. 2014-09-05.

[26] Vasiliy. *MVP and MVC Architectures in Android – part 1*.
http://www.techyourchance.com/mvp-mvc-android-1/. 2015-07-12.

[27] Thinkmobiles. *MVP vs MVVM: A Review of Architectural Patterns for Android*.
https://thinkmobiles.com/blog/mvp-vs-mvvm-android-patterns/. 2017-02-10.

[28] Google. *Data Binding Library*.
https://developer.android.com/topic/libraries/data-binding/index.html. 2017-04-16.

[29] *Retrolambda*.
https://github.com/orfjackal/retrolambda. 2017-04-04.

[30] Google. *Dashboards*.
https://developer.android.com/about/dashboards/index.html. 2017-03-06.

[31] *Retrofit*.
http://square.github.io/retrofit/. 2017-04-04.

[32] *FloatingActionButton*.
https://github.com/Clans/FloatingActionButton. 2017-04-04.

[33] *EditSpinner*.
https://github.com/xyxyLiu/Edit-Spinner. 2017-04-04.

[34] *HockeyApp - The Platform for Your Apps*.
https://hockeyapp.net/#s. 2017-04-06.

[35] *expandable-recycler-view*.
https://github.com/bignerdranch/expandable-recycler-view. 2017-04-06.

[36] Phillipp Jahoda. *MPAndroidChart*.
https://github.com/PhilJay/MPAndroidChart. 2017-04-06.

[37] Google. *Supporting Multiple Screens*.
https://developer.android.com/guide/practices/screens_support.html. 2017-04-14.

Příloha A
Slovníček pojmů

API	Jde o sbírku procedur, funkcí, tříd či protokolů nějaké knihovny (ale třeba i jiného programu nebo jádra operačního systému), které může programátor využívat.
Avatar	Reprezentace uživatele ve virtuální realitě.
Bluetooth	Bluetooth je v informatice proprietární otevřený standard pro bezdrátovou komunikaci propojující dvě a více elektronických zařízení.
Boilerplate kód	Jako boilerplate kód se označují takové části kódu, které musí být vloženy opakovaně a zároveň jsou buď úplně stejné a nebo obsahují drobné změny.
Bug	Chyba v programu.
ČVUT	České vysoké učení technické v Praze
FEL	Fakulta elektrotechnická ČVUT
Github	GitHub je webová služba podporující vývoj softwaru za pomoci verzovacího nástroje Git.
Gradle	Gradle je nástroj pro automatizaci sestavování programu vzniklý původně pro prostředí Javy.
Issue tracker	softwarová aplikace, která je navržena tak, aby napomáhala zajišťovat kvalitu při sledování hlášených softwarových chyb při práci programátorů.
Plugin	Zásuvný modul
Project scope	Znamená souhrn všech dodávaných výstupů, jejichž vytvoření podmiňuje skutečnost, že produkt, nebo služba realizovaná projektem bude dodána se všemi specifikovanými funkcemi a vlastnostmi.
REST	Jedná se o architekturu komunikace mezi serverem a klientem.
User Interface	Grafické uživatelské rozhraní - umožňuje ovládat počítač pomocí interaktivních grafických ovládacích prvků.
Widget	Ovládací prvek počítače. Umožňuje komunikaci uživatele s počítačem a práci s daty. Patří sem například prvky pro výběr a zobrazení (tlačítko pro přepínání, nabídkové menu, ikona programu), textové vstupy (řádek pro napsání adresy, našeptávač) nebo výstupy (indikátor průběhu, balónová nápověda)a dialogová okna.