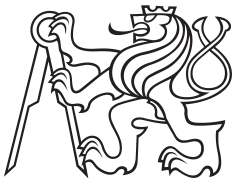
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačů

Mobilní aplikace pro sběr dat z experimentů

Michael Kolář

Školitel: Ing. Ivo Malý, Ph.D.
Obor: Softwarové inženýrství a technologie
Květen 2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kolář** Jméno: **Michael** Osobní číslo: **487232**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Mobilní aplikace pro sběr dat z experimentů

Název bakalářské práce anglicky:

Mobile application for experiment data collection

Pokyny pro vypracování:

Prozkoumejte vývojovou platformu Flutter za účelem tvorby multiplatformních víceuživatelských mobilních aplikací. Dále navrhněte architekturu aplikace pro sběr dat z experimentů. Zaměřte se jak na klasické manuálně vyplňované formulářové dotazy tak i na možnost sběru dat automaticky (např. poloha uživatele). Pro realizaci vyberte vhodné knihovny a datové úložiště splňující požadavky na souběžnou práci více uživatelů. Navrhněte také vhodné datové struktury pro variabilní strukturu otázek a odpovědí. Navržené řešení implementujte ve formě prototypové mobilní aplikace na platformě Flutter pro platformu Android. Řešení ověřte jak na jednoduchém dotazníku např System Usability Scale (SUS) tak i na dlouhodobém experimentu typu deníkové studie.

Seznam doporučené literatury:

1. Flutter, https://flutter.dev/
2. Goodman E., Kuniavsky M., Moed A. Observing the user experience: a practitioner's guide to user research. 2nd ed. Waltham: Morgan Kaufmann, c2012. ISBN 978-0-12-384869-7.
3. Lewis, J. R.: Measuring Perceived Usability: The CSUQ, SUS, and UMUX, International Journal of Human–Computer Interaction, 2018

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Ivo Malý, Ph.D., katedra počítačové grafiky a interakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **24.06.2021** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **19.02.2023**

Ing. Ivo Malý, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

Poděkování

Především bych chtěl poděkovat Ing. Ivu Malému, Ph.D. za jeho ochotu a cenné rady při zpracovávání bakalářské práce již od samotných začátků. Dále bych rád poděkoval všem, kteří se podíleli na návrhu nebo testování mé aplikace. Děkuji, že jste mi věnovali svůj čas a poskytli mi zpětnou vazbu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 20. května 2022

Michael Kolář

Abstrakt

Tato bakalářská práce se zabývá návrhem a implementací mobilní aplikace pro sběr dat z experimentů. Cílem bylo vytvořit fungující mobilní aplikaci pro platformu Android pomocí multiplatformního frameworku Flutter, která by podporovala souběžnou práci více uživatelů najednou. Aplikace byla navržená, aby podporovala jednorázové i opakující se experimenty, které mohou obsahovat různé druhy odpovědí např. textový řetězec, nahrání obrázku, výběr data a další. Pro ukládání získaných dat byla použitá NoSQL databáze Cloud Firestore. Funkčnost navržené implementované aplikace byla otestována pomocí uživatelského testování na reprezentativním vzorku uživatelů.

Klíčová slova: multiplatformní mobilní aplikace, Android, Flutter, sběr dat, experimenty

Školitel: Ing. Ivo Malý, Ph.D.
Katedra počítačové grafiky a interakce

Abstract

This bachelor thesis deals with designing and implementing a mobile application for data collection from experiments. The goal was to create a working mobile application for the Android platform using the multiplatform Flutter framework that would support the simultaneous work of multiple users at once. The application was designed to support non-recurring and recurring experiments, which can contain various types of answers, such as text string, image upload, date selection, and more. The NoSQL database Cloud Firestore was used to store the obtained data. The functionality of the proposed implemented application was examined utilizing user testing on a representative sample of users.

Keywords: multiplatform mobile application, Android, Flutter, data collection, experiments

Title translation: Mobile application for experiment data collection

Obsah

Obrázky

Tabulky

Kapitola 1
Úvod

V současné době jsou mobilní aplikaci velmi populární a není důvod si myslet, že by se jejich popularita v blízké době měla snížit. Využívají se k ovládání bankovních účtů, chytré domácnosti, k navigaci, sdílení zážitků s ostatními lidmi, poslouchání hudby, zkrátka téměř na cokoliv existuje již nějaká mobilní aplikace. Jenom na obchodech Google Play a App store se v tuto chvíli dá stáhnout více než 4,5 mil. aplikací [1].

Celkový vývoj mobilních aplikací mě fascinuje a chtěl jsem do této problematiky více proniknout. Proto se bakalářská práce na vývoj konkrétní aplikace jevila jako skvělá příležitost vyzkoušet si vytvoření mobilní aplikace od samotného návrhu až po funkční aplikaci.

Obsah práce jsem rozdělil do pěti kapitol. Druhá kapitola se věnuje analýze stávajících podobných mobilních aplikací a tvorbě aplikačních požadavků. Třetí kapitola se zabývá návrhem uživatelského rozhraní. Ve Čtvrtá kapitole porovnávám různá implementační řešení. Pátá kapitola detailně popisuje implementaci mobilní aplikace. Poslední kapitola pojednává o provedeném testování.

Kapitola 2
Analýza

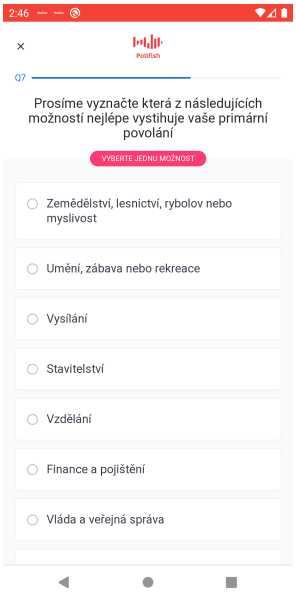
2.1 Podobné mobilní aplikace

Aplikace svými funkcionalitami velmi připomíná různé dotazníkové aplikace, kterých můžeme na každé platformě najít obrovské množství. Rozhodl jsem se vybrat tři nejpopulárnější aplikace z obchodu Google Play, jelikož u aplikací, na rozdíl od podobných obchodů jako třeba App Store, zveřejňuje orientační údaje o počtu stažení. Přestože tyto aplikace jsou dostupné také na ostatních platformách, následující analýzu budu brát zejména z pohledu platformy Android, protože aplikaci budu v rámci bakalářské práce implementovat a zkoušet na této platformě. Konkrétně jsem vybral tyto tři aplikace: AttaPoll, SurveyHeart a YouGov.

2.1.1 Aplikace AttaPoll

Vydavatel: AttaPoll
Počet stažení na Google Play: 1-5 mil.
Dostupnost: Android, iOS

První aplikaci vyvinula britská společnost AttaPoll ltd. Mobilní aplikace je dostupná po celém světě. Vyplňováním dotazníku si uživatelé mohou přijít na menší přivýdělek. U každého z dostupných dotazníků uživatel vidí, kolik za vyplnění dostane peněz a jaká je zhruba časová náročnost. Dotazníky obsahují velmi široké spektrum témat, je možné je i filtrovat podle délky času, který jejich vyplňování zabere. Vydělané peníze je pak možné převést na PayPal účet, nebo do peněženky s kryptoměnami - na výběr jsou v současné době Ethereum a Bitcoin. Poslední možností je darovat vydělané peníze na charitu.

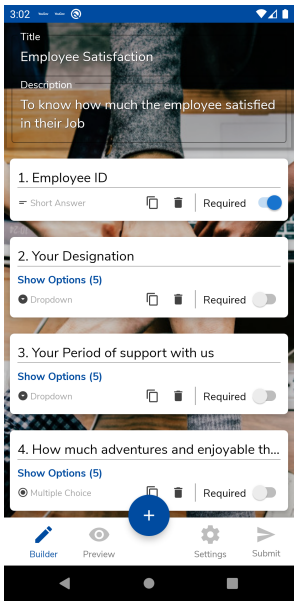


Obrázek 2.1: Podobné aplikace - AttaPoll

2.1.2 Aplikace SurveyHeart

Vydavatel: SurveyHeart
Počet stažení na Google Play: 1-5 mil.
Dostupnost: Android

SurveyHeart vytvořil tým vývojářů z Indie. Aplikace je primárně určená na vytváření dotazníků a formulářů. Uživatel si může vybrat z 9 různých typů otázek. Zajímavou funkcí je možnost vytvářet dotazníky i v offline módu. Pro tvorbu dotazníků je možné si vybrat z více než třiceti různých šablon z několika odlišných kategorií témat. Aplikace také obsahuje funkci náhled dotazníku, kterým si uživatel může vytvořený dotazník zobrazit. Po vyplnění dotazníku obdrží uživatel notifikaci s upozorněním a může se ihned podívat na výsledky dotazníku. Zároveň aplikace nabízí možnost výsledky exportovat do Excelu.

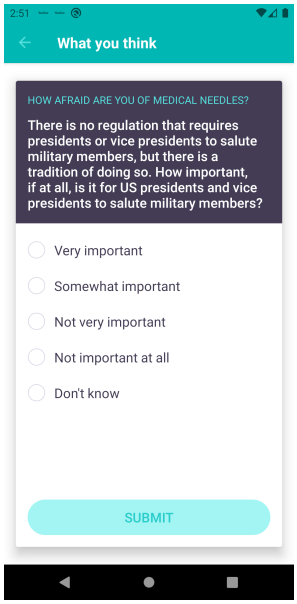


Obrázek 2.2: Podobné aplikace - SurveyHeart

2.1.3 Aplikace YouGov

Vydavatel: YouGov PLC
Počet stažení na Google Play: 1-5 mil.
Dostupnost: Android, iOS

Poslední aplikací je velmi populární YouGov, kterou vyvinula britská společnost YouGov PLC. Uživatelé zde mohou vyplňovat různé dotazníky, většinou politicky zaměřené, za které dostanou odměnu ve formě bodů. Body je pak možné vyměnit za několik různých kupónů, které se liší podle státu. Pokud má někdo zájem vidět výsledky dotazníků, YouGov každý měsíc posílá na e-mail všem svým členům odkazy na výsledky vybraných dotazníků, které zveřejňuje na své webové stránce.



Obrázek 2.3: Podobné aplikace - YouGov

2.2 Funkční požadavky

V sekci níže přikládám funkční požadavky uvedené formou seznamu, které aplikace musí splňovat.

- Aplikace umožní uživateli se zaregistrovat.
- Aplikace umožní uživateli se přihlásit.
- Aplikace umožní uživateli se odhlásit.
- Aplikace umožní uživateli vybrat si, kterých skupin (kategorií) experimentů se chce účastnit.
- Aplikace umožní uživateli zobrazit aktivní experimenty, pouze ze skupin, kterých se chce účastnit.
- Aplikace umožní uživateli zobrazit všechny aktivní experimenty.
- Aplikace umožní uživateli zobrazit všechny aktivní již zodpovězené experimenty.
- Aplikace umožní uživateli prohlížet všechny otázky uvnitř vybraného experimentu.
- Aplikace umožní uživateli odpovídat na všechny otázky uvnitř vybraného experimentu.
- Aplikace umožní uživateli vytvořit nový experiment.
- Aplikace musí umět automaticky sbírat současnou polohu uživatele.
- Aplikace musí umět sbírat data z různých typů otázek (např. textový řetězec, nahrání obrázku atd.).

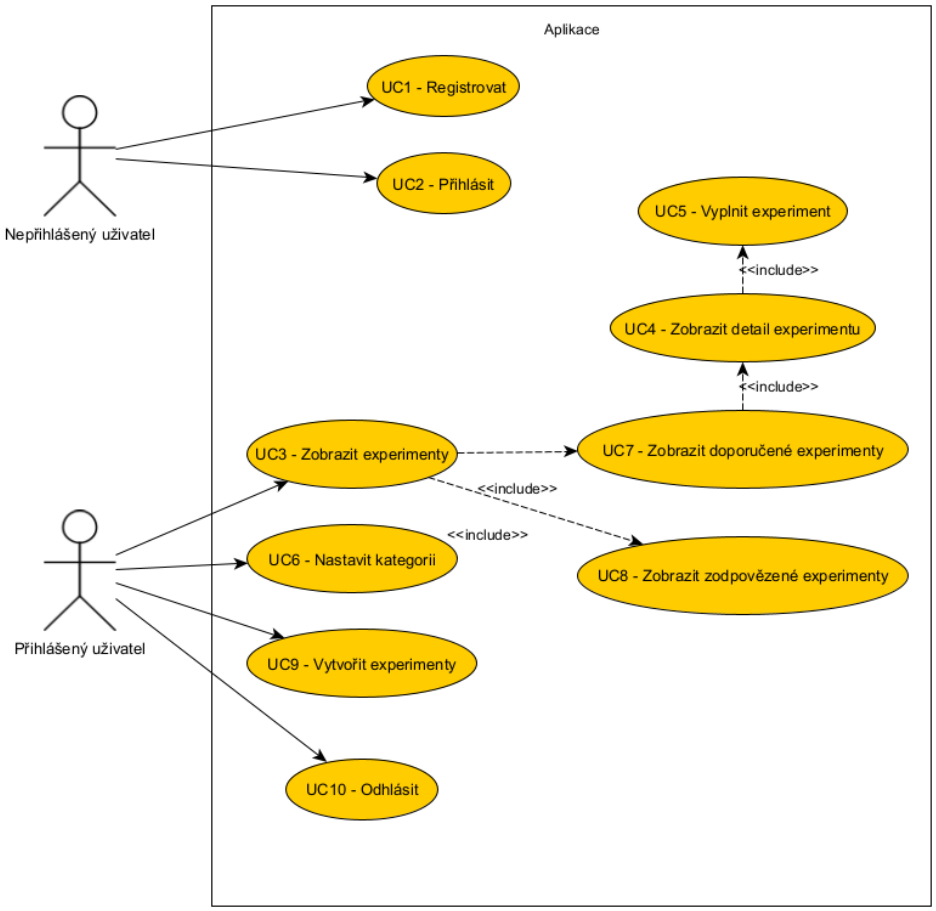
2.3 Nefunkční požadavky

- Aplikace bude zabezpečená a nedovolí neautentizovaným uživatelům prohlížet ani vyplňovat experimenty.
- Aplikace umožní souběžnou práci více uživatelů současně.
- Aplikaci bude v budoucnu možné snadno rozšířit o nové typy sbíraných dat.

2.4 Případy užití

Případy užití (nebo-li *use cases*) specifikují jednotlivé aktivity vycházející z funkčních požadavků, iniciované uživatelem. V sekci níže přikládám textový seznam případů užití. Graficky zpracovaný diagram případu užití je možné vidět v příloze 2.4.

- **UC1 - Registrovat**
  Nepřihlášený uživatel se může zaregistrovat.
- **UC2 - Přihlásit**
  Nepřihlášený uživatel se může přihlásit.
- **UC3 - Zobrazit experimenty**
  Přihlášený uživatel může zobrazit seznam aktivních experimentů.
- **UC4 - Zobrazit detail experimentu**
  Přihlášený uživatel může zobrazit detail experimentu.
- **UC5 - Vyplnit experiment**
  Přihlášený uživatel může vyplnit experiment.
- **UC6 - Nastavit kategorii**
  Přihlášený uživatel může nastavit kategorii experimentů, kterých by se chtěl přednostně účastnit.
- **UC7 - Zobrazit doporučené experimenty**
  Přihlášený uživatel může zobrazit pouze doporučené experimenty na základě kategorií, které si nastavil jako aktivní.
- **UC8 - Zobrazit zodpovězené experimenty**
  Přihlášený uživatel může zobrazit pouze experimenty, které již zodpověděl.
- **UC9 - Vytvořit experimenty**
  Přihlášený uživatel může vytvořit nové experimenty.
- **UC10 - Odhlásit**
  Přihlášený uživatel se může z aplikace odhlásit.



Obrázek 2.4: Diagram - případy užití

Kapitola 3

Návrh

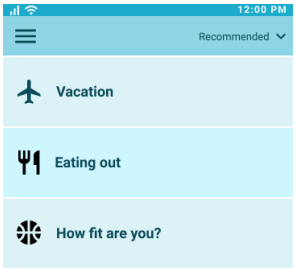
3.1 Uživatelské rozhraní

Pro návrh uživatelského rozhraní jsem použil webovou aplikaci Figma[1]. Aplikaci Figma jsem vybral hlavně z důvodu, že je dostupná přímo z webového prohlížeče a k jejímu používaní postačí pouze internetové připojení. Veškerá práce se pak ukládá do cloudu a je dostupná z jakéhokoliv počítače. Dalším velkým pozitivem je, že aplikaci může kdokoliv používat zcela zdarma.

Obrazovky jsou navrženy pro platformu Android. Při samotném návrhu jsem se velmi inspiroval současnými Material design doporučeními [2]. Každou obrazovku jsem se snažil navrhnout tak, aby byla pro uživatele maximálně komfortní a snadná na užívání.

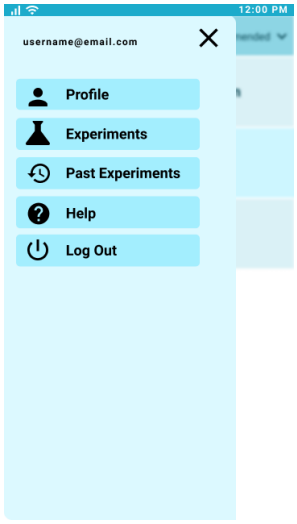
3.1.1 Hlavní obrazovky

K hlavní navigaci v aplikaci slouží navigation drawer, který nalezneme v levém horním rohu. V pravém horním rohu se nachází tři filtry: Recommended, Answered a All. *Recommended* filtr zobrazí experimenty na základě zvolených kategorií, kterých se chceme účastnit. *Answered* filtr zobrazí experimenty, kterých jsme se již zúčastnili a *All* všechny pro nás dostupné experimenty. Ve zbývající části obrazovky pak nalezneme seznam experimentů. Každá položka v seznamu obsahuje ikonku kategorie, ve které se experiment nachází a jeho název.



Obrázek 3.1: Prototyp - Hlavní obrazovka

[1] https://www.figma.com/



Obrázek 3.2: Prototyp - Navigation drawer

Na obrázku **3.2** je znázorněn rozbalený navigation drawer. V horní části je uveden e-mail, pod jakým jsme v aplikaci přihlášeni, napravo od e-mailu se pak nachází tlačítko, kterým se vrátíme zpět, respektive zabalí navigation drawer. Teoreticky by se dal místo navigation drawer použít ještě bottom navigation bar, ale jelikož obsahuje 5 položek, přijde mi tato varianta lepší zejména pro uživatele, kteří mají mobilní telefon s menší obrazovkou.

3.1.2 Experimenty

Do obrazovky s experimentem se dostaneme kliknutím na jakékoliv místo vybrané položky ze seznamu na hlavní obrazovce (viz obrázek **3.1**). Každý experiment může obsahovat libovolný počet otázek. Aplikace bude podporovat následující typy odpovědí:

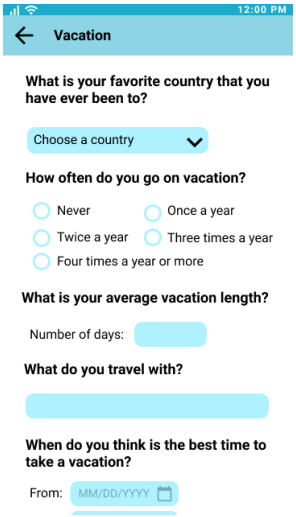
- textový řetězec
- číselný řetězec
- výběr možnosti pomocí skupinového přepínače (radio button)
- výběr z výběrového pole (select option)
- výběr 0 až n odpovědí z n možných odpovědí
- nahrání obrázku
- výběr data, nebo časového období z kalendáře

Vzhledem k diverzitě různých odpovědí, kdy každá zabírá různě velkou část obrazovky, jsem navrhl dvě varianty, jak by obrazovky mohly vypadat:

1. Jednostránková varianta
2. Vícestránková varianta

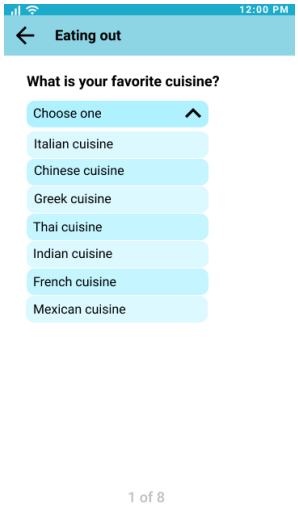
Jednostránková varianta

V jednostránkové variantě se všechny otázky zobrazí na jedné stránce a uživatel bude muset po zodpovězení otázky sám doscrollovat k další otázce. Tato varianta mně osobně připadá lepší pro experimenty s menším počtem otázek, protože uživatel po zodpovězení otázky nemusí dělat gesto rukou navíc a může se rovnou věnovat další otázce. Další výhodu vidím v případě, že se uživatel rozhodne změnit některou z předchozích odpovědí, i když už je skoro u poslední otázky. V této variantě se pomocí scrollování jednoduše dostane tam, kam potřebuje, ale u druhé varianty by musel použít gesto swipe přesně tolikrát, o kolik otázek se chce vrátit zpátky. U otázek, kde jako odpověď vybíráme místo na mapě, se mi tato varianta jeví jako horší, protože plocha mapy není tak velká (ačkoliv mapa jde samozřejmě přiblížit) a některým uživatelům by to mohlo připadat nepřehledné.



Obrázek 3.3: Prototyp - Experiment - Jednostránková varianta

Vícestránková varianta



Obrázek 3.4: Prototyp - Experiment - Vícestránková varianta

Ve vícestránkové variantě se na rozdíl od předchozí varianty každá otázka zobrazí na své vlastní stránce a uživatel se mezi otázkami posouvá pomocí gesta swipe. Tato varianta se mi jeví jako lepší pro některé typy odpovědí, typicky pro ty, které vyžadují více prostoru (např. výběr místa na mapě, zvolení data v kalendáři, náhled obrázku). Problém s návratem zpět o více otázek se dá vyřešit relativně snadno pomocí jedné obrazovky navíc na konci experimentu, kde by se vypsaly všechny odpovědi a kliknutím na ně by se uživatel dostal zpět na konkrétní obrazovku s otázkou a jeho zvolenou odpovědí.

3.1.3 Profil

Poslední důležitá obrazovka pro uživatele je obrazovka s profilem. V této obrazovce má uživatel možnost si nastavit, kterých kategorií experimentů by se chtěl v budoucnu účastnit. Jak můžete vidět na obrázku **3.5**, kategorie jsou uspořádané ve dvou sloupcích. Samotný výběr kategorií je pak jednoduchý - jedním kliknutím kategorii vybereme (položka se poté rozsvítí zelenou barvou) a druhým kliknutím zase můžeme kategorii z výběru odstranit (šedá barva).

Obrázek 3.5: Prototyp - Profil

3.2 Testování prototypu

Posledním krokem bylo mnou navržený interaktivní prototyp aplikace otestovat. Hlavním cílem bylo zjistit, zda je uživatelsky příjemnější jednostránková nebo vícestránková varianta a také jestli průchod obrazovkami je logicky navazující. Testování probíhalo na mobilním zařízení, kde byl spuštěný interaktivní prototyp vytvořený pomocí webové aplikace Figma.

3.2.1 Účastníci

Pro otestování prototypu byli vybráni 3 muži a 1 žena ve věku od 15 do 45 let. Ani jeden z nich nemá žádné větší zkušenosti s testováním aplikací, přesto bych řekl, že vypovídající hodnota je vysoká, protože testování reálně simuluje používaní aplikace běžným uživatelem. Účastníci vlastní chytrý telefon již delší dobu, s používáním aplikací mají různou úroveň zkušeností.

3.2.2 Úkoly

Pro otestování prototypu byly napsány čtyři scénáře. Každý účastník dostane na začátku 30 vteřin na rychlé seznámení s aplikací, poté mu bude předložen scénář, který si přečte a následně se bude snažit zadaný úkol splnit. Zatímco účastníci budou plnit zadané úkoly, já si budu poznamenávat veškeré jejich kroky.

Níže jsou popsány úkoly, které budou přesně takto předloženy účastníkům. Pořadí 2. a 3. úkolu bude prohozené tak, aby dva účastníci měli prvně 2. úkol a dva účastníci měli prvně 3. úkol. To je z důvodu, aby výsledek porovnání jednostránkové a vícestránkové varianty experimentů byl skutečně vypovídající a nebyl ovlivněn např. větší zkušeností s aplikací.

1. Právě jste si nainstalovali aplikaci a chcete si nastavit kategorie, kterých experimentů se chcete účastnit. Nastavte, že se chcete účastnit kategorie *Travel*.
2. Zúčastněte se experimentu *Vacation*.
3. Zúčastněte se experimentu *Eating out*.
4. Podívejte se do historie experimentů, kterých jste se již v minulosti účastnili.

Po splnění všech úkolů jsem se každého účastníka zeptal, zda se mu více líbila jednostránková nebo vícestránková varianta a také na celkové osobní dojmy z aplikace.

1. účastník

Úkol	Výsledek
1.	Úspěch, lehká dezorientace na začátku
2.	Úspěch
3.	Úspěch
4.	Úspěch

Tabulka 3.1: Testování prototypu - Účastník 1

Po testové otázky: Vícestránková varianta byla lepší. Upravil by možnost přepínání na další/předchozí otázku ve vícestránkové variantě experimentu

i pomocí kliknutí (v prototypu to bylo nastaveno jen na gesto swipe) kamkoliv na obrazovku - levá půlka obrazovky by sloužila k vrácení o otázku zpět a pravá naopak k přepnutí na další otázku.

2. účastník

Úkol	Výsledek
1.	Úspěch
2.	Úspěch
3.	Úspěch, menší problém s přepínáním na další otázku
4.	Úspěch

Tabulka 3.2: Testování prototypu - Účastník 2

Po testové otázky: Vícestránková varianta byla o trošku lepší. Při prvním spuštěním aplikace by na začátek umístil krátkou ukázku jako návod, jak aplikaci používat.

3. účastník

Úkol	Výsledek
1.	Úspěch, lehká dezorientace na začátku
2.	Úspěch
3.	Úspěch, problém s gestem swipe
4.	Úspěch

Tabulka 3.3: Testování prototypu - Účastník 3

Po testové otázky: Jednostránková varianta přišla intuitivnější. U vícestránkové varianty by přidal větší nápovědu (např. ve formě šipky na stranách), aby bylo snadnější poznat, že experiment není dokončený a pokračuje otázkou na další straně.

4. účastník

Úkol	Výsledek
1.	Úspěch, lehká disorientace na začátku
2.	Úspěch
3.	Úspěch, problém s gestem swipe
4.	Úspěch

Tabulka 3.4: Testování prototypu - Účastník 4

Po testové otázky: Jednostránková varianta byla lepší z důvodu, že stačí jen scrollovat prstem dolů. U vícestránkové varianty nevěděl na začátku, jak přepnout na další otázku.

3.2.3 Vyhodnocení testování

Testování prototypu se neobešlo bez komplikací. Největším problémem bylo, že v prototypu fungovaly běžně používaná gesta trošku neobratně a to vedlo ke zmatení testerů. Nicméně hlavní cíl testovaní se podařilo naplnit a to zjistit, že vícestránková varianta je pro uživatele příjemnější a přehlednější. Další zjištění byla většinou designového rázu, např. doplnění informativních ikonek k intuitivnějšímu ovládání aplikace.

Kapitola 4

Technická analýza

V následující kapitole porovnávám vývoj mobilních aplikací pomocí Flutter frameworku s ostatními multiplatformními přístupy. Dále zde rozebírám problematiku spojenou se správou stavů ve Flutteru.

4.1 Vývojové nástroje - SDK

Volba implementačního řešení byla pro mě jednoznačná, jelikož ze zadaní bakalářské práce vyplývá, že aplikace musí být naimplementována pomocí platformy Flutter. V následujících podsekcích porovnávám Flutter s nativním přístupem a nejvíce používanými multiplatformními řešeními, tedy React Native a Xamarin.

4.1.1 Nativní přístup

Nativní přístup je bez diskuze nejlepší možné řešení z pohledu optimalizace pro danou platformu. Proto je také nejpopulárnější a nejvíce používaný pro vývoj mobilních aplikací. Díky tomu se těší velké komunitě aktivních vývojářů a kvalitní dokumentaci. Největší nevýhodou nativních řešení je nutnost znát více programovacích jazyků. Pro vývoj mobilních aplikací na platformu Android se využívá Kotlin a Java, pro platformu iOS zase Swift a Objective-C. Důsledkem toho vyžaduje vývoj na obě platformy většinou dva různé týmy a je finančně nákladnější.

- Výhody
  1. Optimalizace pro danou platformu
  2. Snadný přistup k funkcím operačního systému
  3. Konzistence komponent
  4. Nejnovější API pro zařízení
  5. Velká komunita a kvalitní dokumentace
- Nevýhody
  1. Potřeba znát více programovacích jazyků

2. Pouze pro danou platformu
3. Cena a doba vývoje

4.1.2 Flutter

Flutter je open-source framework a mezi multiplatformními řešeními poměrně novinkou. Vyšel v roce 2017 a je vyvíjen společností Google. Využívá programovací jazyk Dart, rovněž od společnosti Google, který je následně kompilován do nativního jazyku dané platformy. V porovnání s ostatními multiplatformními způsoby vývoje je díky tomu rychlejší. Komponenty, pomocí kterých se tvoří uživatelské rozhraní se nazývají widgety [3]. Jako velkou výhodu vidím funkci Hot Reload, která umožňuje aplikaci znovu zkompilovat při zachování současného stavu [4]. Dále disponuje kvalitní dokumentací, která značně přispívá k rostoucímu počtu vývojářů. Na druhou stranu Flutter stále prochází poměrně bouřlivým vývojem, tudíž je dost pravděpodobné, že kód napsaný ve starší verzi nebude v nové funkční.

- Výhody
  1. Funkce Hot Reload
  2. Rychlý a levný vývoj mobilních aplikací
  3. Jednotný kód pro platformy Android a iOS
  4. Kvalitní dokumentace
- Nevýhody
  1. Menší komunita vývojářů
  2. Bouřlivý vývoj
  3. Méně dostupných balíčků a knihoven

4.1.3 React Native

React Native je nejpopulárnější framework pro vývoj multiplatformních mobilních aplikací. Je vyvíjen společností Facebook. Pro vývoj se používá programovací jazyk JavaScript, kvůli tomu vykreslení většího množství dat zabere mnohem delší dobu, než je tomu u nativního vývoje. Disponuje rovněž obdobnou funkcí, jako je Hot Reload ve Flutter, která se nazývá Fast Refresh [5]. Těší se velké komunitě vývojářů.

- Výhody
  1. Funkce Fast Refresh
  2. Velká komunita a počet dostupných balíčků
  3. Jednotný kód pro platformy Android a iOS
- Nevýhody
  1. Horší optimalizace
  2. Mizerná dokumentace

4.1.4 Xamarin

Xamarin je dalším možným řešením vývoje multiplatformních mobilních aplikací. Vyšel v roce 2011, je od společnosti Microsoft. Je open-source a dále rozšiřuje .NET platformu nástroji speciálně určenými pro vývoj aplikací na iOS, Android a macOS [6]. Pro vývoj se používá programovací jazyk C#. Stejně jako tomu je u Flutter a React Native, je zde možnost aplikaci aktualizovat za běhu pomocí funkce Hot Reload. Poměrně velkou nevýhodu může být placená profesionální licence.

- Výhody
  1. Funkce Hot Reload
  2. Velká komunita a kvalitní dokumentace
  3. Jednotný kód pro platformy Android a iOS
- Nevýhody
  1. Velikost aplikace
  2. Placená licence
  3. Opožděná aktualizace API

4.2 State management

Správa stavu je poměrně komplexní téma a zvolení správného přístupu je důležité již na začátku vývoje aplikace. Existuje celá řada různých balíčků, které práci s vnitřními stavy aplikace značně usnadňují. Vnitřními stavy aplikace je myšleno ukládání dat vznikajících interakcí uživatele s aplikací, tedy např. vyplňování různých formulářových políček. Tato data je následně často potřeba sdílet do jiných obrazovek či komponent.

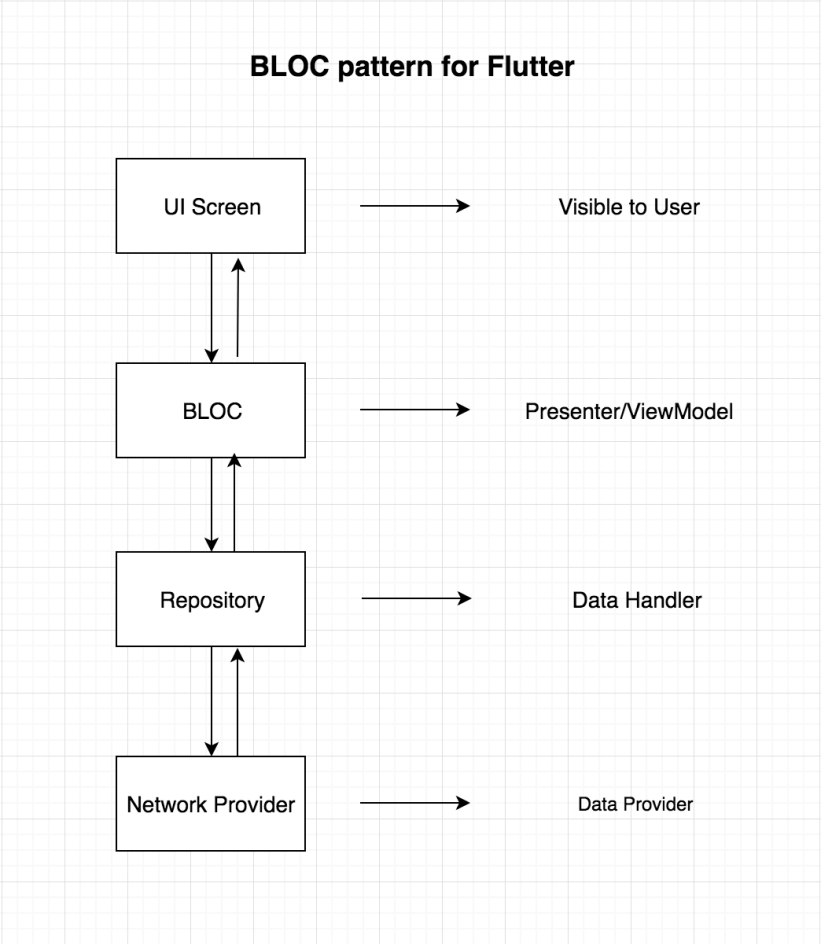
4.2.1 Bloc vzor

Bloc je návrhový vzor, který slouží k oddělení bussiness logiky od prezentační vrstvy. V mnohém se podobá již osvědčeným návrhovým vzorům MVP a MVVM. Jediným rozdílem od MVVM je nahrazení části ViewModel částí Bloc, která zachytává všechny události (interakci uživatele). Tyto události následně ukládá a při každé změně signalizuje zpátky do prezentační vrstvy, že se změnil stav a měla by se překreslit s novými daty, podrobnější vysvětlení je možné nalézt v článku [7].

Na obrázku **4.1** můžeme vidět rozdělení návrhového vzoru na jednotlivé části:

- UI Screen - obsahuje veškeré uživatelské rozhraní (obrazovky a komponenty) viditelné pro uživatele
- Bloc - slouží jako zprostředkovatel komunikace mezi prezentační a datovou vrstvou

- Repository - zajišťuje spojení s Network Provider vrstvou a data následně přetvoří do klasických objektů se kterými se pak snadněji pracuje
- Network Provider - obstarává veškerou komunikaci s databází a dalšími obdobnými službami



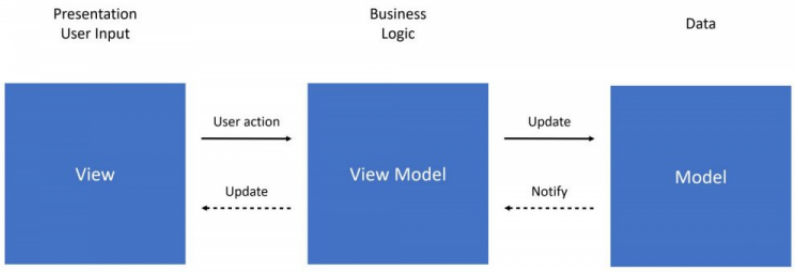
Obrázek 4.1: Návrhový vzor Bloc [8]

4.2.2 MVVM vzor

MVVM je jedním z dalších velmi populárních návrhových vzorů pro vývoj mobilních aplikací. Stejně jako Bloc odděluje prezentační vrstvu od bussiness logiky do vrstev, které se nazývají ViewModel a Model. ViewModel funguje jako prostředník mezi vrstvami View a Model. Veškerá interakce uživatele proběhne přes tuto vrstvu a na View vratí aktualizovaná data. Mezi hlavní výhody patří udržovatelnost, snadné otestování aplikace pomocí unit testů a možnost rozšíření aplikace v budoucnu [9].

Z obrázku **4.2** je patrné, že tento návrhový vzor se dělí na následující části:

- View - obsahuje všechny komponenty a obrazovky, které uživatel vidí při používaní aplikace.
- ViewModel - prostředník mezi View a Model, který zpracovává události od uživatele, pošle je Model části a následně upozorní View, že by se mělo uživatelské rozhraní aktualizovat s novými daty.
- Model - obsahuje objekty s daty.



Obrázek 4.2: Návrhový vzor MVVM [7]

4.2.3 InheritedWidget

InheritedWidget je widget, který umožňuje předávat data do svých potomků. Tento přístup se vyplatí jedině u malých aplikací, kde není příliš potřeba získávat data z ostatních widgetů. Výhodou pak je, že InheritedWidget je součástí Flutter frameworku a není potřeba stahovat žádné další balíčky.

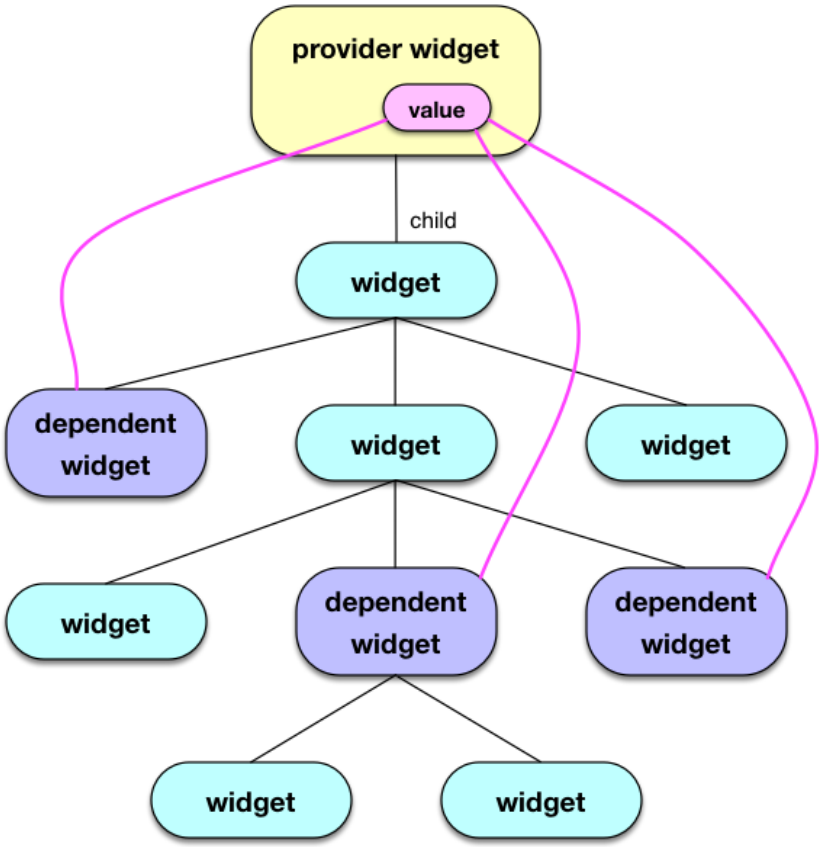
4.2.4 SetState

SetState je rovněž součástí Flutter frameworku a slouží k upravení stavu uvnitř widgetů. Při změně stavu se daný widget překreslí s novými daty. Je vhodný pouze v situacích, kdy není potřeba tato data dále sdílet do ostatních widgetů.

4.2.5 Provider

Provider balíček je velmi populární řešení, které značně usnadňuje správu stavů ve Flutteru. Dle oficiální dokumentace je to wrapper okolo InheritedWidget [10]. Funguje na principu, že se celá aplikace obalí tímto widgetem viz obrázek **4.3** a je pak možné dostat se k datům ze všech ostatních widgetů dle potřeby. Díky tomu se poměrně výrazně sníží nadbytečný kód. Další velkou výhodou tohoto balíčku je možnost použít Consumer třídu, která zjednodušeně popsáno, umožňuje nastavit posluchače na daný Provider a kdykoliv se v něm změní data, upozorní všechny své posluchače, že by se měl tento widget

překreslit s aktualizovanými daty. Mezi hlavni nevýhody bych zařadil větší složitost na pochopení a závislost na BuildContext.



Obrázek 4.3: Provider diagram [11]

Kapitola 5
Implementace

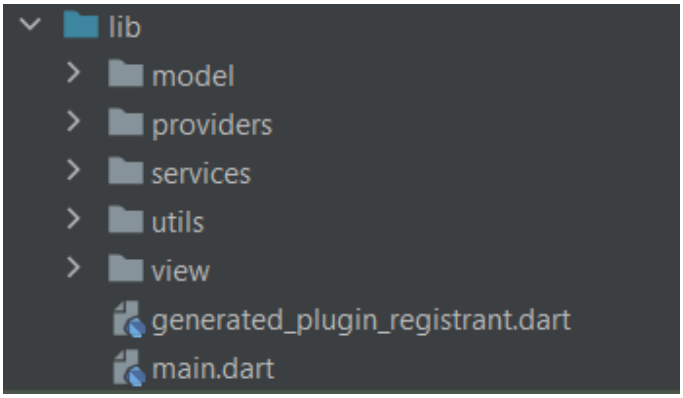
Zadáním této bakalářské práce bylo vytvořit funkční prototyp pomocí frameworku Flutter pro platformu Android. V následující kapitole popisuji, jak jsem postupoval při implementaci mobilní aplikace.

5.1 Zvolený architektonický vzor

Při implementaci jsem se snažil držet architektonického vzoru MVVM doplněného balíčkem Provider pro snadnější správu stavu aplikace. MVVM jsem vybral zejména z důvodu, že je poměrně podobný vzoru MVC, který se často využívá při vývoji webových aplikací, a který jsem si osvojil při studiu. Jelikož jsem s Flutter frameworkem neměl žádné předchozí zkušenosti, ze začátku jsem zkoušel používat InheritedWidget ale s rostoucí složitostí aplikace jsem zjistil, že toto řešení není vhodné. Rozhodl jsem se tedy pro vyzkoušení Provider balíčku, který se mi v pozdější části vývoje velmi osvědčil, ačkoliv byl složitější na pochopení.

Projekt jsem pro snadnější orientaci rozdělil do následujících částí viz obrázek **5.1**:

- Model - zde se nachází všechny modely, které představují data se kterými aplikace pracuje.
- Providers - adresář, ve kterém jsou všechny třídy typu Provider reprezentující část ViewModel z vybrané architektury MVVM.
- Services - složka, kde najdeme všechny služby (např. databáze), které se volají z části ViewModel.
- Utils - zde se ukrývají pomocné služby např. ke směřovaní (routing) chodu mezi obrazovkami.
- View - obsahuje veškeré uživatelské rozhraní, tedy widgety, které uživatel vidí při používaní aplikace.

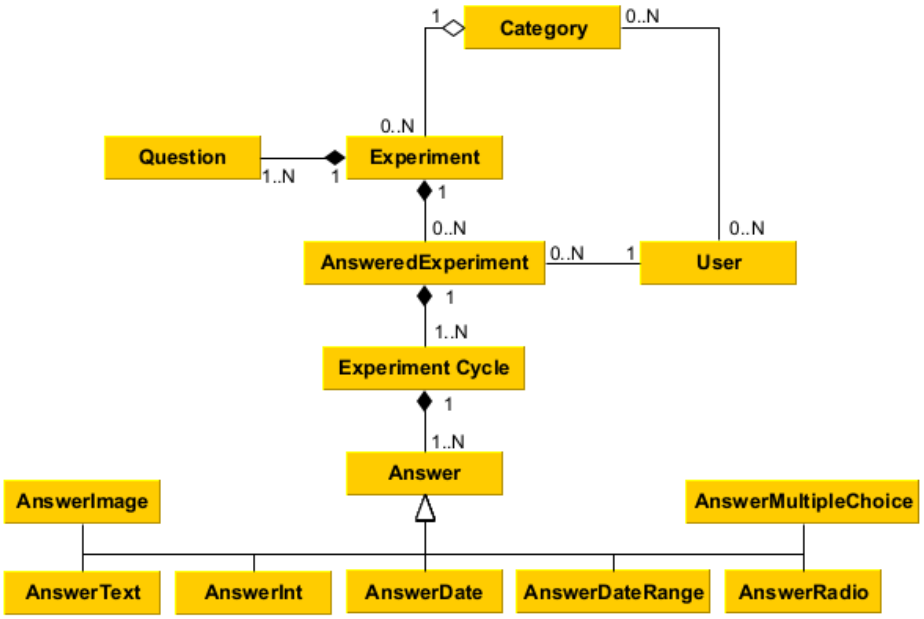


Obrázek 5.1: Struktura projektu

5.1.1 Datový model

Pro správný chod aplikace bylo potřeba vymodelovat vhodné objekty, které by reprezentovaly experimenty obsahující různé typy otázek. Z navrženého prototypu jsem měl poměrně dobrou představu, jak by takové objekty mohly vypadat.

Jelikož bylo potřeba experimenty rozlišovat na opakující se (např. deníková studie) a neopakující se (jednorázový experiment), vytvořil jsem třídu *Experiment*, která obsahuje List objektů typu *ExperimentCycle*. Tento objekt pak reprezentuje jednotlivý cyklus a umožňuje ukládání každého opakovaní do třídy *Experiment*. Poté bylo nutné vytvořit objekty, které by umožnily ukládání různých typů dat. Kvůli různorodému typu sbíraných dat z jednotlivých typů otázek jsem se rozhodl vytvořit abstraktní třídu *Answer*, která tyto objekty zastřešuje a obsahuje společné atributy a metody. Následně jsem vytvořil pro všechny typy otázek samostatný objekt, který abstraktní třídu *Answer* rozšiřuje a doplňuje o potřebné atributy, aby bylo možné získané odpovědi ukládat. Tyto odpovědi bylo poté možné vložit do Listu u třídy *ExperimentCycle* a jednoduše s nimi pracovat. Třídní diagram je možné vidět na obrázku níže 5.2



Obrázek 5.2: Třídní diagram

5.2 Služby

V následující sekci popisuji, jaké služby jsem využil při implementaci aplikace. Nejdůležitějším krokem bylo vybrat správnou databázi, kam by se ukládala získaná data. K tomu jsem využil službu Firebase[1], která je na trhu od roku 2011 a stejně jako Flutter je od společnosti Google. Používá se zejména pro vývoj mobilních a webových aplikací. Mezi hlavní důvody, proč jsem se rozhodl vybrat Firebase, patří snadná integrace databáze do Flutter aplikace a možnost využít autentizační službu.

5.2.1 Autentizace

Většina aplikací potřebuje nějakým způsobem omezit neautentizovaným uživatelům provádět změny v databázi. Firebase má svojí vlastní službu, která přesně toto zajišťuje. Rozhodl jsem se tedy tento autentizační balíček[2] využít. Implementace registrace a přihlašovaní do aplikace byla díky tomu poměrně snadná, jelikož jsem nemusel např. vůbec řešit předávaní tokenů a hashovaní hesla. Balíček umožňuje registraci a přihlašovaní uživatelů do aplikace pomocí klasického uživatelského jména a e-mailu ale i pomocí třetích služeb jako např. přes Google, Facebook a Twitter. Pro přístup do aplikace jsem vybral autentizaci uživatelů pomocí e-mailu a hesla.

[1] https://firebase.google.com/
[2] https://pub.dev/packages/firebase_auth

5.2.2 Cloud Firestore

Firebase nabízí dvě různé služby, které umožňují ukládání dat do databáze - Cloud Firestore a Realtime Database. Cloud Firestore je novější databáze, je postavená na úspěších Realtime Database, ale s jistými vylepšeními, o kterých se zmíním níže. Obě databáze můžeme klasifikovat jako NoSQL.

Na základě podrobného popisu a porovnání obou databází v oficiální Firebase dokumentaci [12] jsem se rozhodl, že pro mé potřeby se více hodí použít Cloud Firestore. Hlavními kritérii pro moji volbu byla jednodušší práce s komplexnějšími daty a možnost získaná data řadit a filtrovat dle potřeby.

Do Cloud Firestore se data ukládají ve formě kolekce dokumentů. Dokumenty jsou objekty, které velmi připomínají klasické JSON objekty. V případě potřeby ukládat komplexnější hierarchická data se dají využít subkolekce, které značně zjednodušují práci s databází.

Jelikož se každá operace čtení či zápisu z databáze počítá a v případě překročení určité hranice je zpoplatněná, bylo nutné práci s daty náležitě optimalizovat. K tomu jsem využil tzv. streamy, které v případě přidání či odebrání dokumentu neprocházejí celou kolekci, ale pouze přečtou změnu a aktualizují předchozí data. To znamená, že pokud se zrovna dívám na experimenty a jiný uživatel v tom samém okamžiku vytvoří jeden nový experiment, stream zaznamená změnu a započítá jako jedno nové provedené čtení, místo aby přečetl celou kolekci a započítal všechny dokumenty znovu.

5.2.3 Cloud Storage

Pro možnost ukládat uživatelem vložené obrázky jsem se rozhodl využít další službu od Firebase - Cloud Storage[3], která je přímo určená k ukládání obrázků, videí a zvukových nahrávek.

Integrace Cloud Storage do stávající aplikace nebyla složitá. Do dokumentu uloženého v Cloud Firestore s vyplněným experimentem jsem vložil jako atribut referenci s veřejnou url adresou na obrázek uložený v Cloud Storage, který bylo následně možné snadno získat.

5.2.4 Lokalizační služba

Pro automatický sběr aktuální polohy uživatele jsem využil balíček Geolocator[4]. Geolocator je speciálně udělaný pro framework Flutter a umožňuje snadný přístup k geolokačním službám na platformách Android, iOS, macOS a dalších. Aby sběr souřadnic proběhl úspěšně, je potřeba potvrdit povolení přístupu k geolokačním službám mobilního zařízení. Vyskakovací okénko jsem naimplementoval, tak aby se zobrazilo pouze při vyplňovaní experimentů, kde je automatický sběr lokace uživatele nastaven jako aktivní.

[3] https://firebase.google.com/docs/storage
[4] https://pub.dev/packages/geolocator

5.3 Implementace uživatelského rozhraní

Poslední důležitou částí implementace bylo vytvořit uživatelské rozhraní, pomocí kterého by uživatel mohl aplikaci ovládat. Jelikož jsem neměl žádné předchozí zkušenosti s Flutter frameworkem, tato část byla pro mě poměrně obtížná a zabrala mi nejvíc času.

Díky předem vytvořenému prototypu jsem jasně věděl, jak by aplikace měla vypadat. Obrazovky jsem se snažil dělat přesně podle návrhu. Nejprve jsem začal tvořit obrazovky pro registraci a přihlášení. Poté jsem vytvořil hlavní obrazovku zobrazující všechny experimenty, ze kterých si uživatel může vybrat a vyplnit libovolný experiment.

Dalším krokem bylo vytvořit widgety, které by reprezentovaly všechny implementované typy otázek. Jelikož každý experiment může obsahovat 1 až N částí, bylo nutné je šikovně poskládat do sebe. V prototypu se mi nejvíce osvědčila jednostránková varianta, to znamená, že každá otázka je na samostatné stránce a uživatel se mezi nimi přesouvá pomocí gesta swipe. K tomu jsem použil dostupný widget *PageView*, který je součástí Flutter frameworku.

Posledním důležitým krokem implementace uživatelského rozhraní bylo vytvořit navigační menu, pomocí kterého by se uživatel mohl snadno přesouvat po aplikaci. K tomu jsem použil dostupný widget *Drawer*, který je rovněž součástí Flutter frameworku.

Nakonec jsem se rozhodl, že ještě předělám informační a chybové hlášky, které se mi moc nelíbily. K tomu jsem našel vhodný widget *SnackBar*, který jsem nastavil tak, aby se zobrazil na 15 vteřin ve spodní části obrazovky s požadovanou hláškou a barvou.

5.3.1 Vytvoření nových experimentů

Vzhledem k tomu, že mobilní telefony nejsou k vytváření takto komplexních dat kvůli své velikosti příliš stavěné, rozhodl jsem se v rámci této bakalářské práce detailní uživatelské rozhraní do mobilní aplikace neimplementovat. Vhodnějším místem pro vytváření nových experimentů by byla webová aplikace, kde by se experimenty vytvářely.

Nicméně experimenty je možné do aplikace alespoň nahrát vložením textu ve formě JSON objektů s potřebnými parametry do textového pole, které se nachází v navigačním menu pod názvem *Add Experiments*. Níže přikládám podrobný návod, jak je možné experiment vytvořit a přidat do aplikace.

Experiment

Návod na vytvoření validní JSON objektu reprezentující třídu Experiment uvnitř aplikace:

- Povinné parametry

  1. name - název experimentu. (String)

2. category - název kategorie. (String, možnosti jsou: Science, Music, Travel, Food, Sports)
3. location - jestli chceme, aby experiment zjistil aktuální polohu uživatele při odeslání vyplněného experimentu. (Boolean)
4. questions - list otázek, podrobněji popisuji níže.
5. repeated - umožňuje experiment opakovat. (Boolean, pokud je true, tak se nepovinné parametry Cycles a NumberOfDays stávají povinné)

- Nepovinné parametry

1. dateFrom - možnost nastavení počátečního data, pokud není vyplněno, automaticky se nastaví na dnešní datum. Poznámka: jednorázové experimenty jsou nastaveny jako aktivní po dobu 30 dní. (String, ve formátu 'rok-měsíc-den')
2. cycles - počet cyklů opakování. (Integer)
3. numberOfDays - počet dnů mezi jednotlivými opakováními. (Integer)

Question

Návod na vytvoření validní JSON objektu reprezentující třídu Question uvnitř aplikace:

- Povinné parametry

1. qType - typ otázky. (String, možnosti jsou: textInput, intInput, dateInput, dateRangeInput, imageInput, radioButtonInput, multipleChoiceInput)
2. question - popis otázky. (String)

- Nepovinné parametry

1. options - povinný parametr pokud qType je radioButtonInput nebo multipleChoiceInput. Reprezentuje možné odpovědi, ze kterých si může následně uživatel vybrat. (List<String>)

Ukázkový validní JSON se všemi požadovanými parametry pro vytvoření jednorázového experimentu je možné vidět v příloze A.1. Opakovaný experiment se nachází v příloze A.2. Do aplikace je možné nahrát více experimentů najednou.

5.3.2 Spuštění experimentu

Nejdůležitější obrazovkou, kterou aplikace obsahuje, je obrazovka, kde si uživatel vybírá z možných experimentů a rozhoduje se, který by chtěl vyplnit. Zde jsem vycházel skoro přesně z navrženého prototypu. Jediným rozdílem bylo upravení vzhledu opakujících se experimentů. Přidal jsem ikonku symbolizující

opakování na pravé straně, aby je bylo možné rozeznat od jednorázových a zároveň jsem za názvem přidal aktuální cyklus a celkový počet (hodnota v závorce) viz obrázek **5.3**.

V hlavní části obrazovky se zobrazuje list experimentů. Kliknutím na libovolný z nich se uživatel dostane do detailního pohledu experimentu a může začít odpovídat na otázky v experimentu.

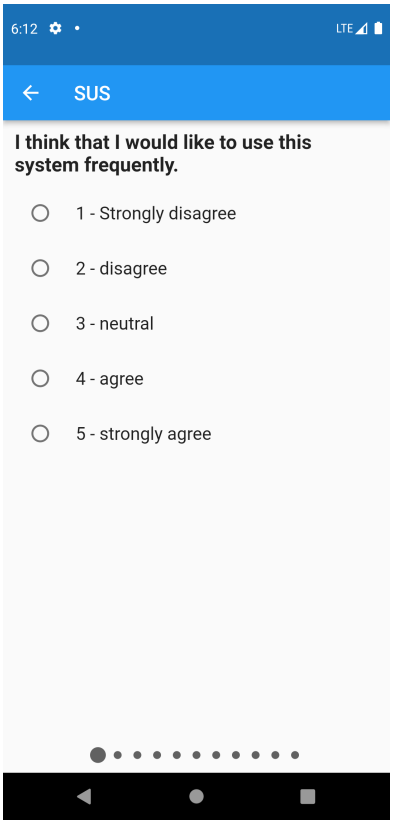
Obrázek 5.3: Uživatelské rozhraní - Výběr experimentu

5.3.3 Vyplnění experimentu

Na obrázku **5.4** je vidět, jak vypadá detailní pohled na experiment po rozkliknutí. V horní části se vždycky zobrazí popis jednotlivých částí. V hlavní části obrazovky se nachází widget, který reprezentuje sbíraná data. Gestem swipe nebo kliknutím na tečku ve spodní části obrazovky se uživatel posune k další části experimentu. Tečky znázorňují počet částí experimentu. Velká tečka značí část experimentu, která je aktuálně zobrazená na obrazovce.

Implementované obrazovky sbíraných dat:

- Textové pole - umožňuje vyplnit text libovolné délky, velikost pole se automaticky přizpůsobuje délce textu.
- Číselné pole - umožňuje vyplnit číslo.
- Vložení obrázku - umožňuje nahrát obrázek z galerie mobilního zařízení.
- Vyplnění data - zobrazí kalendář, kde uživatel vybere datum.
- Vyplnění data od do - zobrazí kalendář, kde uživatel vybere datum od a datum do.
- Výběr 1 odpovědi z N možných - ukázku, jak widget vypadá, je možné vidět na obrázku níže **5.4**.
- Výběr 1 až N odpovědí z N možných

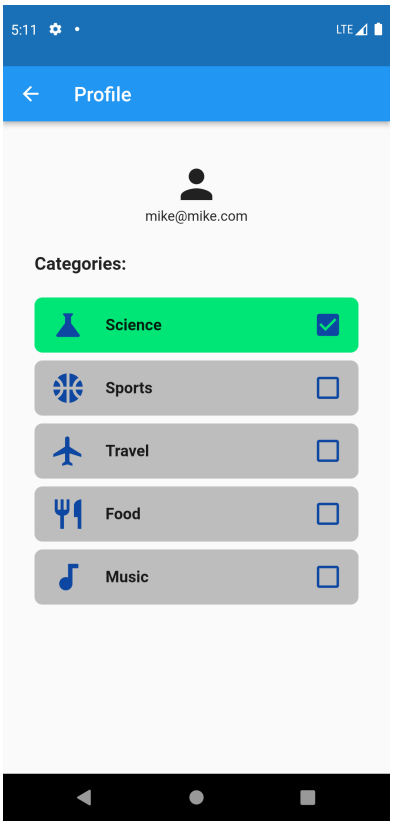


Obrázek 5.4: Uživatelské rozhraní - Vyplnění experimentu

5.3.4 Profil

Při implementaci obrazovky profilu k žádné výraznější změně od navrženého prototypu nedošlo. Uživatel si zde může vybrat kategorie, kterých by se chtěl přednostně účastnit. Pokud je kategorie šedá, značí to, že není momentálně aktivní. Kategorie se aktivuje kliknutím na pole, tím se rozsvítí barva na zelenou a ikonka se odškrtne viz obrázek **5.5**.

Každý experiment je zařazený do jedné z těchto pěti kategorií a je možné je podle těchto kategorií filtrovat. To znamená, že pokud mám na profilu aktivní kategorie např. *Science* a *Travel* a na hlavní obrazovce, kde jsou vidět všechny experimenty, vyberu filtr *Recommended*, zobrazí se pouze aktivní experimenty, které jsou v kategorii *Science* nebo *Travel*.



Obrázek 5.5: Uživatelské rozhraní - Profil

5.4 Možnost rozšíření aplikace o další sbíraná data

Aplikaci jsem se snažil vytvořit tak, aby v případě potřeby rozšíření možností sbíraných dat v budoucnu byla implementace co nejjednodušší. Pokud bychom hypoteticky chtěli aplikaci rozšířit o možnost např. sběru souřadnic z mapy, byl by postup následující:

- Nejprve by bylo nutné vytvořit třídu reprezentující data, která chceme sbírat. V tomto konkrétním příkladě by to byly atributy datového typu Double umožňující uchovat hodnotu vybraného místa na mapě, tedy souřadnice zeměpisné šířky a délky.
- Následně by bylo potřeba vytvořit widget, kde by uživatel mohl vybrat místo na mapě. K tomu bychom museli využít API např. Google Maps Platform[5].
- Poté by bylo nutné přiřadit widgetu atribut typu String (např. locationInput), podle kterého by aplikace dokázala rozpoznat, že se má daný widget vykreslit.
- Posledním krokem by bylo upravit třídu *answerService*, kde by se doimplentovalo ukládání daného objektu do Cloud Firestore databáze.

[5] https://developers.google.com/maps/documentation

Kapitola 6
Testování

Poslední důležitou částí této bakalářské práce bylo mnou naimplementovanou aplikaci otestovat. V následující kapitole detailně popisuji, jak testování probíhalo.

6.1 Uživatelské testování

Aplikaci jsem se rozhodl otestovat na reálném mobilním zařízení s operačním systém Android. Pro důkladné otestování aplikace jsem se rozhodl vytvořit pět vlastních scénářů, kde uživatelé mají za cíl splnit jednoduché úkoly. Každý úkol je zaměřen na jinou část mobilní aplikace. Poslední část testování byla určená ze zadaní bakalářské práce, tedy otestovat aplikaci na deníkové studii a SUS (System Usability Scale) dotazníku. Hlavním cílem testovaní je odhalit, jestli se v aplikaci nevyskytují nějaké chyby, kterých jsem si při implementaci nevšiml a také zda je aplikace uživatelsky příjemná při používaní.

6.1.1 Účastnící testovaní

Pro testování bylo vybráno pět testerů, jelikož podle článku [13] bylo zjištěno, že pět testerů stačí na objevení až 85 % problémů.

Testery jsem se snažil vybrat, tak abych měl různorodé zastoupení věkových kategorií, pohlaví, předchozích zkušeností s používáním a testováním mobilních aplikacích a to z důvodu maximálního přiblížení reálnému prostředí a simulace běžných uživatelů.

Účastník 1

Věk: 47
Pohlaví: Muž
Mobilní zařízení: Samsung Galaxy A20E
Zkušenosti: Běžný uživatel mobilních aplikací, žádné zkušenosti s testováním ani vývojem.

Účastník 2

Věk: 13
Pohlaví: Muž
Mobilní zařízení: Xiaomi Redmi Note 8
Zkušenosti: Nadprůměrný uživatel mobilních aplikací, žádné zkušenosti s testováním ani vývojem.

Účastník 3

Věk: 44
Pohlaví: Žena
Mobilní zařízení: Xiaomi Redmi Note 8 Pro
Zkušenosti: Běžný uživatel mobilních aplikací, žádné zkušenosti s testováním ani vývojem.

Účastník 4

Věk: 26
Pohlaví: Žena
Mobilní zařízení: Samsung A53
Zkušenosti: Nadprůměrný uživatel mobilních aplikací, žádné zkušenosti s testováním ani vývojem.

Účastník 5

Věk: 24
Pohlaví: Muž
Mobilní zařízení: Xiaomi Redmi Mi 8 Light
Zkušenosti: Nadprůměrný uživatel mobilních aplikací, střední zkušenost s testováním i vývojem.

6.1.2 Testovací scénáře

V následujících podsekcích popisuji kroky jednotlivých testovacích scénářů, které jsem přesně ve stejném znění předložil všem testerům. Každému uživateli jsem nejdříve předal prvních pět testovacích scénářů, aby je všechny hned po sobě provedli. Pro všechny testy platí stejná prerekvizita - mobilní zařízení musí být připojeno k internetové síti.

1. scénář

Registrace a přihlášení do aplikace

1. Spustit aplikaci.

2. Klepnout na tlačítko "Sign Up".
3. Vyplnit požadované údaje.
4. Klepnout na tlačítko "Create Account".
5. Po úspěšné registraci je uživatel přenesen na přihlašovací obrazovku.
6. Vyplnit své přihlašovací údaje.
7. Klepnout na tlačítko "Log In".
8. Při úspěšném přihlášení je uživatel přenesen na obrazovku s experimenty.

2. scénář

Nastavení doporučené kategorie

1. Klepnout na ikonku menu v levém horním rohu.
2. Po levé straně se zobrazí navigační menu.
3. Vybrat "Profile" z nabídky navigačního menu.
4. Klepnout na kategorie "Science".
5. Kategorie se rozsvítí zelenou barvou a checkbox na pravé stráně se zaškrtne.
6. Klepnout na ikonku šipky zpět v levém horním rohu.

3. scénář

Filtrování experimentů podle nastavené kategorie v profilu

1. Klepnout na "All" v pravém horním rohu.
2. Zobrazí se rozbalovací menu.
3. Vybrat položku "Recommended" z rozbalovacího menu.
4. Zobrazí se pouze experimenty, kategorie *Science*.

4. scénář

Vyplnění experimentu

1. Klepnout na experiment s názvem "TEST EXPERIMENT" z nabídky všech experimentů.
2. Zobrazí se obrazovka s detailem experimentu.
3. Odpovědět na první část experimentu klepnutím na zvýrazněné (modré) pole.

4. Gestem swipe se přesunout na další část experimentu.
5. Stejným postupem se dostat až na poslední část experimentu.
6. Zobrazí se obrazovka, kde jsou vidět všechny odpovědi uživatele.
7. Klepnout na tlačítko "Save Answers".
8. Zobrazí se hlavní obrazovka se všemi experimenty.
9. Klepnout na "All"v pravém horním rohu.
10. Zobrazí se rozbalovací menu.
11. Vybrat položku "Answered" z rozbalovacího menu.
12. Zobrazí se obrazovka s již zodpovězenými experimenty, obsahující právě vyplněný experiment s názvem "TEST EXPERIMENT".

5. scénář

Odhlášení z aplikace

1. Klepnout na ikonku menu v levém horním rohu.
2. Po levé straně se zobrazí navigační menu.
3. Vybrat "Log Out" z nabídky navigačního menu.
4. Uživatel je odhlášen a zobrazí se úvodní přihlašovací obrazovka.

6. deníková studie

Po dokončení prvních pěti testovacích scénářů jsem dal každému účastníkovi za úkol zúčastnit se deníkové studie. Otázky pro tuto deníkovou studii mají reálný základ, čerpal jsem je výběrem z diplomové práce [14] zabývající se vlivem spánku na pracovní výkon. Deníkovou studii jsem nastavil jako opakující se po jednom dni po dobu pěti dní.

7. SUS dotazník

Poslední částí testování bylo vyplnit System Usability Scale (dále jen SUS) dotazníků. SUS vymyslel John Brooke v roce 1986. Jedná se o jednoduchý dotazník, skládající se z 10 otázek, které uživatel ohodnotí jako ve škole pomocí stupnice 1-5 a má za cíl zjistit vnímanou jednoduchost používaní aplikace [15], což se mi zdá jako jedna z nejdůležitějších vlastností u tohoto typu aplikací.

Do aplikace jsem přidal tento dotazník pod názvem SUS a každého účastníka testovaní jsem požádal, aby dotazník po dokončení deníkové studie vyplnil.

6.2 Výsledek uživatelského testování

Na závěr jsem se všech účastníků zeptal na následující po testové otázky:

1. Co se Vám na aplikaci líbilo?
2. Co se Vám na aplikaci nelíbilo?
3. Jak byste aplikaci zlepšili?

Účastník 1

První účastník testování zvládl všechny úkoly bez problémů viz tabulka 6.1.

Úkol	Výsledek
1.	Úspěch
2.	Úspěch
3.	Úspěch
4.	Úspěch
5.	Úspěch
6.	Úspěch
7.	Úspěch

Tabulka 6.1: Testování - Účastník 1

Po testové otázky:

1. Design aplikace je velmi dobře navržený a uživatelsky přívětivý.
2. Nic mě nenapadá.
3. Nic mě nenapadá, vše funguje dle očekávaní.

Účastník 2

Během vyplňování deníkové studie byla objevená poměrně závažná chyba. Pokud mělo mobilní zařízení při vyplňování experimentů, který sbíral automaticky polohu uživatele, vypnutou službu pro určování polohy, tak se data získaná z experimentu neuložila do databáze. Zbytek úkolů proběhl bez problémů viz tabulka 6.2.

Úkol	Výsledek
1.	Úspěch
2.	Úspěch
3.	Úspěch
4.	Úspěch
5.	Úspěch
6.	Částečný úspěch
7.	Úspěch

Tabulka 6.2: Testování - Účastník 2

Po testové otázky:

1. Vzhled aplikace vypadal hezky. Aplikace běžela plynule a nebyla složitá na používání.
2. U delších experimentů nebyly vidět odpovědi při návratu na předchozí otázky. K úspěšnému uložení experimentu se sběrem lokace byla potřeba zapnutá lokace jinak se experiment neuložil do databáze.
3. U delších experimentů by mohl uživatel vidět svoje předchozí odpovědi. Přidat upozornění, že není zapnutá poloha v nastavení mobilu.

Účastník 3

Při testování byl zjištěn problém s mobilním zařízení od společnosti Xiaomi. Účastník při vyplňování experimentu ve čtvrtém úkolu 6.3 napsal odpověď do textového pole, přesunul se na další otázku gestem swipe, kde již ale neměl možnost schovat klávesnici (u ostatních mobilních zařízení od jiné značky byla vždy možnost šipkou dolů klávesnici schovat).

Úkol	Výsledek
1.	Úspěch
2.	Úspěch
3.	Úspěch
4.	Částečný úspěch
5.	Úspěch
6.	Úspěch
7.	Úspěch

Tabulka 6.3: Testování - Účastník 3

Po testové otázky:

1. Jednoduchost používání, přehlednost.
2. Nenapadá mě nic konkretního.

3. Zapamatováním přihlašovacích údajů. Přidat možnost vyplnit čas výběrem z hodin. Při vyplňování experimentu se neukládaly do jednotlivých dotazových obrazovek již zodpovězené odpovědi. U deníkové studie mi vadilo, že nebylo vidět, jestli jsem vyplnila každé opakovaní. U vyplňovacích polí vyskočí ve spodní části klávesnice, kterou nelze zrušit a překáží u dalších otázek.

Účastník 4

Testování aplikace čtvrtým účastníkem nebyla nalezena žádná chyba viz tabulka 6.4.

Úkol	Výsledek
1.	Úspěch
2.	Úspěch
3.	Úspěch
4.	Úspěch
5.	Úspěch
6.	Úspěch
7.	Úspěch

Tabulka 6.4: Testování - Účastník 4

Po testové otázky:

1. Na aplikaci se mi líbilo, že byla snadno použitelná.
2. Vzhled aplikace by mohl být trošku hezčí.
3. Aplikace by mohla mít víc jazyků.

Účastník 5

Poslední tester navrhl, aby se do aplikace přidala informační hláška při zaškrtávání kategorie v profilu. Při testování nebyla nalezena žádná chyba viz tabulka 6.5.

Úkol	Výsledek
1.	Úspěch
2.	Úspěch, přidal by SnackBar, že se změna uložila
3.	Úspěch
4.	Úspěch
5.	Úspěch
6.	Úspěch
7.	Úspěch

Tabulka 6.5: Testování - Účastník 5

Po testové otázky:

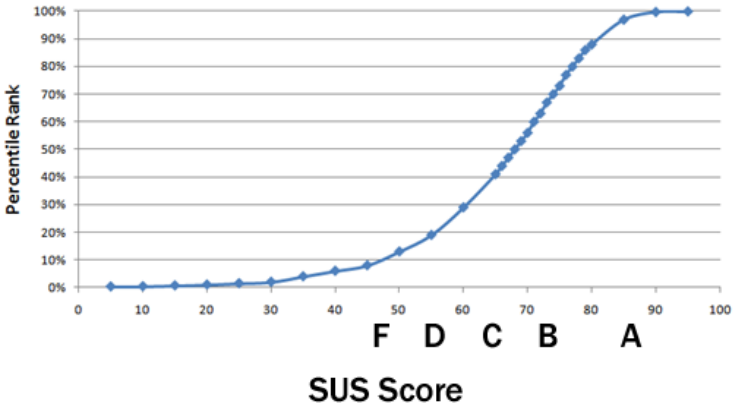
1. Jednotný design, jednoduchost používaní.
2. Ve filtrovacím menu bych zvolil místo slova *recommended* nějaké lépe vystihující slovo.
3. Shromažďování polohovacích údajů bych do experimentu přidal jako další otázku, tak aby bylo vidět, že jsem v tomto experimentu sdílel svojí polohu.

6.2.1 Výsledek deníkové studie

Ačkoliv byla deníková studie nastavená, aby se opakovala po jednom dni po dobu pouze pěti dní, většina účastníků testovaní alespoň jeden den zapomněla na deníkovou studii odpovědět. Přesto hodnotím testování jako úspěšné, jelikož nebyl nalezen žádný problém s opakujícími se experimenty.

6.2.2 Výsledek SUS

Aplikace je považována za nadprůměrnou pokud dosáhne skóre 68 a více, a podprůměrnou pokud dosáhne méně než 68. Na obrázku **6.1** vidíme, že výsledné hodnocení se dá interpretovat pomocí percentilu. Pokud chceme aby naše aplikace byla v horních 10 % všech aplikací, musela by dosáhnout skóre 80,3. To je také hranice, kde je mnohem větší šance, že uživatelé následně aplikaci doporučí ke stažení např. svému kamarádovi nebo kolegovi.



Obrázek 6.1: SUS hodnocení

Po vyhodnocení všech SUS dotazníků, kterým jsem zakončil testování aplikace uživateli, bylo dosaženo průměrného skóre 79,5 viz tabulka níže 6.6. Aplikace by tedy měla být nadprůměrnou s reálnou šancí úspěchu.

Účastník	Body
1	75
2	77,5
3	82,5
4	77,5
5	85
Průměr	**79,5**

Tabulka 6.6: Testování - Výsledek SUS

6.3 Závěr testování

Všichni účastníci testování se shodli, že je mobilní aplikace funkční, přehledná a snadno použitelná. Zároveň většina ocenila design. V rámci testování byly nalezeny drobné chyby v aplikaci. Nejzásadnější z nich byla, že pokud nebyla zapnutá lokace na mobilním zařízení, experiment, který vyžadoval zapnutou lokaci, se i přes vyplnění neuložil a neodeslal do databáze. Zároveň se při vyplňování experimentu nezobrazila žádná chybová hláška, která by uživatele upozornila na zapnutí lokace. Další větší chybou bylo nezobrazování vyplněních odpovědí při návratu na předchozí obrazovky experimentu.

Kapitola 7

Závěr

V rámci bakalářské práce jsem si vyzkoušel několik různých částí z vývoje mobilních aplikací. V první části jsem provedl analýzu podobných mobilních aplikací, které se již na trhu nachází a vytvořil aplikační požadavky. Poté jsem prozkoumal Material Design doporučení a podle nich navrhl interaktivní prototyp, jak by aplikace mohla vypadat. Tento prototyp jsem nechal otestovat dalšími lidmi, kteří mi dali skvělou zpětnou vazbu, co bych měl ještě vylepšit nebo případně udělat jinak, až budu implementovat skutečnou aplikaci.

Ve druhé části jsem započal implementaci navržené aplikace. Implementaci mobilní aplikace poměrně dost ztěžoval fakt, že jsem s frameworkem Flutter nikdy dřív nepracoval. Po prvotních obtížích se aplikaci podařilo naimplementovat. Výsledná aplikace splňuje všechny aplikační požadavky a vzhledem se od navrhovaného prototypu příliš neliší. Nakonec jsem provedl uživatelské testy, které měly za cíl odhalit, zda je aplikace uživatelsky příjemná a neobsahuje chyby.

Výsledkem testování bylo ověřeno, že mobilní aplikace je funkční, přehledná, snadno použitelná s pěkným uživatelským rozhraním. Během testovaní byly zároveň nalezeny chyby, které by bylo v rámci budoucího užití nutno opravit.

7.1 Budoucnost práce

Naimplementovaná aplikace je spíše ve formě prototypu, jak by taková aplikace mohla vypadat. Pokud by se měla aplikace začít více používat, bylo by nutné vytvořit webové rozhraní pro pohodlnější přidávání experimentů a zároveň aplikaci vyzkoušet a otestovat na platformě iOS.

Literatura

[1] Mansoor Iqbal. App Download and Usage Statistics (2022). https://www.businessofapps.com/data/app-statistics/. [Citováno: 2022-04-22].

[2] Material Design guidelines. https://material.io/design. [Citováno: 2022-04-23].

[3] Flutter architectural overview. https://docs.flutter.dev/resources/architectural-overview. [Citováno: 2022-04-26].

[4] Flutter. Hot reload. https://docs.flutter.dev/development/tools/hot-reload. [Citováno: 2022-04-26].

[5] React Native. Fast Refresh. https://reactnative.dev/docs/fast-refresh. [Citováno: 2022-04-26].

[6] Xamarin. What is Xamarin? https://dotnet.microsoft.com/en-us/learn/xamarin/what-is-xamarin. [Citováno: 2022-04-26].

[7] Abhishek Dhanani. Which Pattern to Choose From MVVM and Bloc? https://flutteragency.com/which-pattern-to-choose-from-mvvm-and-bloc/. [Citováno: 2022-04-27].

[8] Sagar Suri. Architect your Flutter project using BLOC pattern. https://medium.com/codechai/architecting-your-flutter-project-bd04e144a8f1. [Citováno: 2022-04-27].

[9] Jitesh Mohite. Flutter: MVVM Architecture. https://medium.com/flutterworld/flutter-mvvm-architecture-f8bed2521958. [Citováno: 2022-04-29].

[10] Provider documentation. https://pub.dev/documentation/provider/latest/. [Citováno: 2022-05-01].

[11] Joseph T. Lapp. Understanding Provider in Diagrams. https://medium.com/flutter-community/understanding-provider-in-diagrams-part-1-providing-values-4379aa1e7fd5. [Citováno: 2022-05-01].

[12] Choose a Database: Cloud Firestore or Realtime Database. https://firebase.google.com/docs/database/rtdb-vs-firestore. [Citováno: 2022-05-02].

[13] NIELSEN, Jakob; LANDAUER, Thomas K. A mathematical model of the finding of usability problems. In: Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems. 1993. p. 206-213.

[14] Šumský, Adam. Deníková studie: vliv spánku na pracovní výkon. Diplomová práce. Masarykova univerzita, Brno, 2018. https://is.muni.cz/th/p7euh/. [Citováno: 2022-05-04].

[15] The System Usability Scale and How it's Used in U. https://medium.com/thinking-design/the-system-usability-scale-how-its-used-in-ux-b823045270b7. [Citováno: 2022-05-04].

Příloha A

Vzorový experiment

A.1 Vzorový jednorázový experiment

Experiment je nastaven, aby nesbíral automaticky polohu uživatele. Obsahuje všechny implementované typy sbíraných dat.

```
[
    {
       "name":"EXPERIMENT NAME",
       "category":"Sports",
       "location": false,
       "repeated":false,
       "questions":[
          {
             "qType":"textInput",
             "question":"Description"
          },
          {
             "qType":"intInput",
             "question":"Description"
          },
          {
            "qType":"dateInput",
            "question":"Description"
         },
         {
            "qType":"dateRangeInput",
            "question":"Description"
         },
         {
            "qType":"radioButtonInput",
            "question":"Description. Note(1-N possible
options)",
            "options":[
               "Option A",
```

```
                "Option B",
                "Option C",
                "Option D"
            ]
        },
        {
            "qType":"multipleChoiceInput",
            "question":"Description. Note(1-N possible
    options)",
            "options":[
                "Option A",
                "Option B",
                "Option C",
                "Option D"
            ]
        },
        {
            "qType":"imageInput",
            "question":"Description"
        }
    ]
}
]
```

A.2 Vzorový opakovaný experiment

Experiment je nastaven, aby sbíral automaticky polohu uživatele. Celkem se experiment 10 zopakuje a jednotlivé opakovaní je nastaveno na délku 5 dní. Obsahuje všechny implementované typy sbíraných dat.

```
[
    {
        "name":"EXPERIMENT NAME",
        "category":"Science",
        "location": true,
        "repeated":true,
        "numberOfDays":5,
        "cycles":10,
        "questions":[
            {
                "qType":"textInput",
                "question":"Description"
            },
            {
                "qType":"intInput",
                "question":"Description"
```

```
          },
          {
            "qType":"dateInput",
            "question":"Description"
         },
         {
            "qType":"dateRangeInput",
            "question":"Description"
         },
         {
            "qType":"radioButtonInput",
            "question":"Description. Note(1-N possible
    options)",
            "options":[
               "Option A",
               "Option B",
               "Option C",
               "Option D"
            ]
         },
         {
            "qType":"multipleChoiceInput",
            "question":"Description. Note(1-N possible
    options)",
            "options":[
               "Option A",
               "Option B",
               "Option C",
               "Option D"
            ]
         },
         {
            "qType":"imageInput",
            "question":"Description"
         }
      ]
   }
]
```

Příloha B

Návod jak aplikaci spustit

Aplikaci je možné spustit po vytvoření instalačního souboru. Instalační soubor vytvoříme ze zdrojového kódů po nainstalování Flutter SDK[1].

Následně se pomocí příkazu *flutter pub get* stáhnou všechny potřebné knihovny a balíčky. Příkazem *flutter build apk* se vytvoří instalační soubor, který je možné následně nahrát na mobilní zařízení.

[1] Podrobný návod, jak Flutter SDK nainstalovat se nachází zde - `https://docs.flutter.dev/get-started/install`