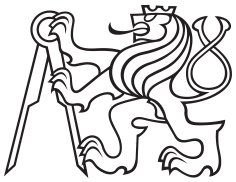
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačů

Zpřístupnění služby Telegram pro nevidomé uživatele

Martin Hruška

Softwarové inženýrství a technologie

Vedoucí: Doc. Ing. Daniel Novák Ph.D.
Školitel–specialista: Ing. Jan Hadáček
Květen 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Hruška** Jméno: **Martin** Osobní číslo: **474759**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Zpřístupnění služby Telegram pro nevidomé uživatele

Název bakalářské práce anglicky:

Making Telegram service accessible to blind users

Pokyny pro vypracování:

Projekt se soustředí na zpřístupnění služby Telegram nevidomým uživatelům. Cílem projektu je vytvoření mobilní aplikace pro Android, která bude komunikovat s API společnosti Telegram. Nesdílnou součástí projektu je i návrh rozhraní, a to společně s uživateli s využitím technologie návrhu zaměřeného na uživatele (tzv. user-centered design).
1. Seznamte se s principy návrhu softwarových řešení pro nevidomé uživatele
2. Zpřístupněte rozhraní aplikace Telegram pro nevidomé uživatele.
3. Rozhraní otestujte na 5 nevidomých uživatelích

Seznam doporučené literatury:

[1] Matt May , Wendy Chisholm , Universal Design for Web Applications, 2008
[2] Universal Design - World Blind Union, 2020

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Daniel Novák, Ph.D., Analýza a interpretace biomedicínských dat FEL

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

Datum zadání bakalářské práce: **13.02.2020** Termín odevzdání bakalářské práce: **14.08.2020**

Platnost zadání bakalářské práce: **30.09.2021**

doc. Ing. Daniel Novák, 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

Rád bych tímto poděkoval Ing. Janu Hadáčkovi za jeho čas a cenné rady při technické implementaci bakalářského projektu.

Prohlášení

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

V Praze, ...

Abstrakt

Tato bakalářská práce se zabývá analýzou požadavků, návrhem a implementací Android chatovací aplikace komunikující s chatovací službou Telegram a uzpůsobené potřebám nevidomým uživatelům. V rámci analýzy jsou vyhodnoceny funkční požadavky, které jsou později využity k návrhu struktury aplikace a její implementaci. Výsledná aplikace je poté otestována zrakově postíženými uživateli za účelem získání poznatků, které budou využity k jejímu budoucímu vývoji.

Klíčová slova: Android, Telegram, chat, nevidomí

Abstract

This bachelor's thesis deals with an analysis of requirements, design and implementation of Android chat application which is communicating with the Telegram chat service and is adapted to the needs of blind users. As part of the analysis, there is an evaluation of functional requirements which are later used to design the structure of the application and to its implementation. The resulting application is then tested by visually impaired users in order to gain knowledge that will be used for its future development.

Keywords: Android, Telegram, chat, blind

Title translation: Making Telegram service accessible to blind users

Obsah

Obrázky

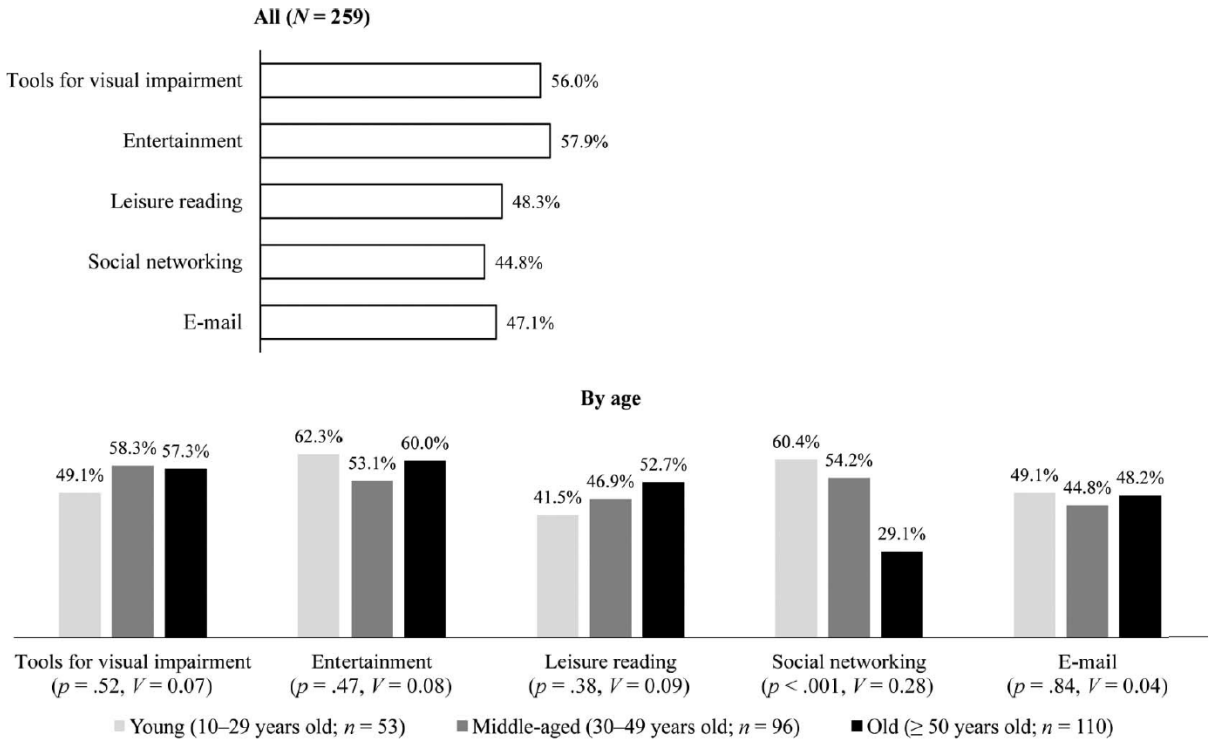
Tabulky

Kapitola 1

Úvod

Statistiky z konce roku 2019 [2] ukazují, že mobilní chatovací služby označované anglickým výrazem *instant messaging apps* mají po celém světě až 2,52 miliard uživatelů. Do roku 2022 by pak tato hodnota měla překročit hranici 3 miliard. Chatovací služby mají přibližně o 20% více uživatelů než sociální sítě a u uživatelů s chytrým mobilním telefonem (smartphonem) je až 75% šance, že aktivně používají minimálně jednu chatovací službu. Jedná se tak s přehledem o nejčastější formu dálkové komunikace, která poskytuje efektivní a především bezplatnou alternativu k mobilní komunikaci pomocí SMS a MMS zpráv. Nutnost připojení k interenetu je v dnešní době už jen malým problémem.

S neustálým rozvojem mobilních technologií roste i zájem zrakově postižených uživatelů o využívání populárních mobilních aplikací, ať už se jedná o aplikace pomahájící jim v jejich každodenním životě, aplikace sloužící pro jejich zábavu nebo právě aplikace využívané k dálkové komunikaci s ostatními uživateli mobilních telefonů. Tento zájem zaznamenává i online průzkum ***A Survey on the Use of Mobile Applications for People Who Are Visually Impaired*** [9] z roku 2017, jehož cílem bylo zjištění míry užívání mobilních aplikací zrakově postíženými uživateli mobilních telefonů včetně jejich celkového dojmu z těchto aplikací. Průzkumu se zúčastnilo 259 zrakově postižených uživatelů mobilních telefonů, z toho 144 (55.6%) bylo zcela nevidomých. Na obrázku 1.1 jsou znázorněny nejpoužívanější aplikace u různých věkových skupin účastníků průzkumu. Z obrázku je patrné, že k většímu rozdílu mezi věkovými skupinami dochází jen u využívání sociálních sítí, kde starší generace výrazně zaostává za tou mladší. Důvodem je jakési upínání starší generace na tradiční způsoby komunikace pomocí SMS zpráv a posílání e-mailů, které je způsobeno především nezájmem o učení se nových technologií, které navenek kvůli velkému množství nabízených funkcionalit vypadají pro tuto generaci příliš komplikovaně. Dále je z obrázku patrné, že téměř polovina účástníku bez ohledu na věk aktivně využívá e-mail, což může indikovat možný zájem o další komunikační aplikace jako jsou právě moderní chatovací služby. Tabulka 1.1 znázorňující počet měsíčně stažených aplikací u jednotlivých účastníků průzkumu pak jen potvrzuje, že mobilní aplikace jsou dnes nedílnou součástí života i u zrakově postižených uživatelů.



Obrázek 1.1: Obrázek znázorňující nejpoužívanější aplikace u 259 účastníků průzkumu s ohledem na jejich věk.

Počet měsíčně stažených aplikací	Počet účastníků	% účastníků
1-2	159	61.4%
3-5	66	25.5%
více než 5	34	13.1%

Tabulka 1.1: Tabulka znázorňující počet měsíčně stažených aplikací u 259 účastníků průzkumu.

1.1 Cíl bakalářské práce

Cílem bakalářské práce je analýza požadavků na chatovací službu pro nevidomé, její návrh a implementace testovatelného prototypu. Tento prototyp, má být následně otestován alespoň třemi [1] nevidomými uživateli. Výsledky testování, které pravděpodobně pozmění požadavky na aplikaci i její strukturu, budou využity pro budoucí vývoj chatovací služby. Prototyp bude přímo implementován do již existující aplikace Petra Svobodníka (dále jen *kmenová aplikace*) sloužící pro zjednodušení používání operačního systému Android nevidomým uživatelům. Prototyp tak bude muset dodržovat určité návrhové

[1] Nemožnost dostatečného testování z důvodu povinné karantény zapříčinilo pokles této hodnoty z hodnoty uvedené v zadání.

a implementační konvence *kmenové aplikace*, které budou blíže popsány v kapitolách **Návrh** 3 a **Implementace** 4.

1.2 Struktura bakalářské práce

V kapitole **Analýza** 2 budou nejprve zmíněny dostupné mobilní technologie pro nevidomé, dále zde budou představeny nejpopulárnější chatovací služby a nakonec zde budou definovány funkční požadavky potřebné k návrhu a implementaci aplikace. Kapitola **Návrh** 3 nejprve rozebírá principy návrhu aplikací pro nevidomé uživatele a následně detailně popisuje strukturu všech částí implementovaného prototypu odvozené od struktury *kmenové aplikace*. V kapitole **Implementace** 4 jsou popsány technologie použité k implementaci prototypu včetně technického popisu *kmenové aplikace* a popisu integrace aplikace s asynchronním API chatovací služby Telegram. Nakonec je zde detailně rozebrána implementace jednotlivých částí navrženého prototypu. Kapitola **Testování** 5 popisuje testování prototypu třemi nevidomými uživateli a jeho výstupy v podobě nalezených chyb, chybějící funkcionality či neintuitivní struktury některých částí prototypu. Součástí této kapitoly je i příloha B obsahující výsledky z testování prototypu. V kapitole **Závěr** 6 jsou pak zhodnoceny celkové výstupy projektu, do budoucna implementovatelné funkcionality a míra naplnění cílů.

Kapitola 2

Analýza

Tato kapitola se nejprve zabývá existujícími technologiemi vyvinutými pro nevidomé uživatele, poté porovnává míru užívání nejpopulárnějších chatovacích služeb a vysvětluje důvod vybrání chatovací služby Telegram k implementaci aplikace. Nakonec jsou vyhodnoceny funkční požadavky na aplikaci, které jsou použity k jejímu návrhu v kapitole **Návrh** 3 a její implementaci v kapitole **Implementace** 4.

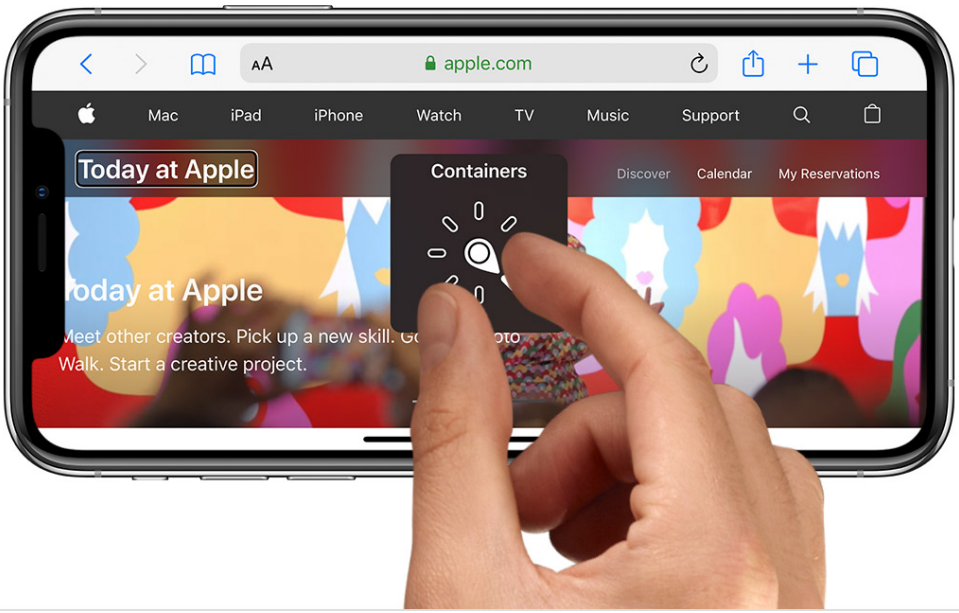
2.1 Dostupné mobilní technologie pro nevidomé uživatele

Před rozšířením tzv. chytrých telefonů se lidé se zrakovým postižením museli spoléhat na řadu technologií sestavených k jedinému účelu. Mezi tyto technologie patří například různá hlasová GPS zařízení, hlasové rozpoznávače měny, zařízení pro čtení Braillova písma, speciální e-book čtečky apod. Koupě všech těchto zařízení však není levnou záležitostí a ne všichni si je tak mohou dovolit. Mnohem levnější a praktičtější alternativou k těmto jednoúčelovým technologiím je právě chytrý telefon (anglicky *smartphone*) využívající jednoho z moderních mobilních operačních systémů (např. Android, iOS, Windows Phone atd.) a aplikačního prostředí pro instalaci nejrůznějších mobilních aplikací. Uživatel si tak namísto pořizování jednotlivých jednoúčelových zařízení pořídí pouze chytrý telefon, do kterého si stáhne aplikace pokrývající funkcionality všech těchto technologií, čímž docílí centralizace všech potřebných funkcionalit do jednoho jediného zařízení.

2.1.1 Mobilní telefony s dotykovým displejem

Nevidomí uživatelé chytrých telefonů s dotykovým displejem většinou využívají tzv. čtečky obrazovky (anglicky *screen reader*) sloužící k prezentování obsahu obrazovky v podobě hlasu či Braillova písma uživateli a nějakého softwaru na rozpoznávání hlasu (anglicky *voice recognition software*) sloužícího k převodu hlasového vstupu uživatele na text. Čtečky obrazovky provádějí různé akce na základě uživatelem provedeného gesta (přejetí prstem, dvojité klepnutí prstem apod.). Software rozpoznávání hlasu pak provádí akce související s významem hlasového vstupu uživatele. Mezi nejpopulárnější mobilní čtečky obrazovky patří VoiceOver pro operační systém iOS, TalkBack pro operační systém Android a Narrator pro operační systém Windows Phone. Mezi

nejpopulárnější softwary na rozpoznávání hlasu patří Siri pro operační systém iOS, Google Now (Google Assistant) pro operační systém Android a Cortana pro operační systém Windows Phone. [11]



Obrázek 2.1: Ukázka fungování čtečky obrazovky VoiceOver. Obrázek stáhnutý z `https://support.apple.com/cs-cz/HT204783` dne 25. 7. 2020

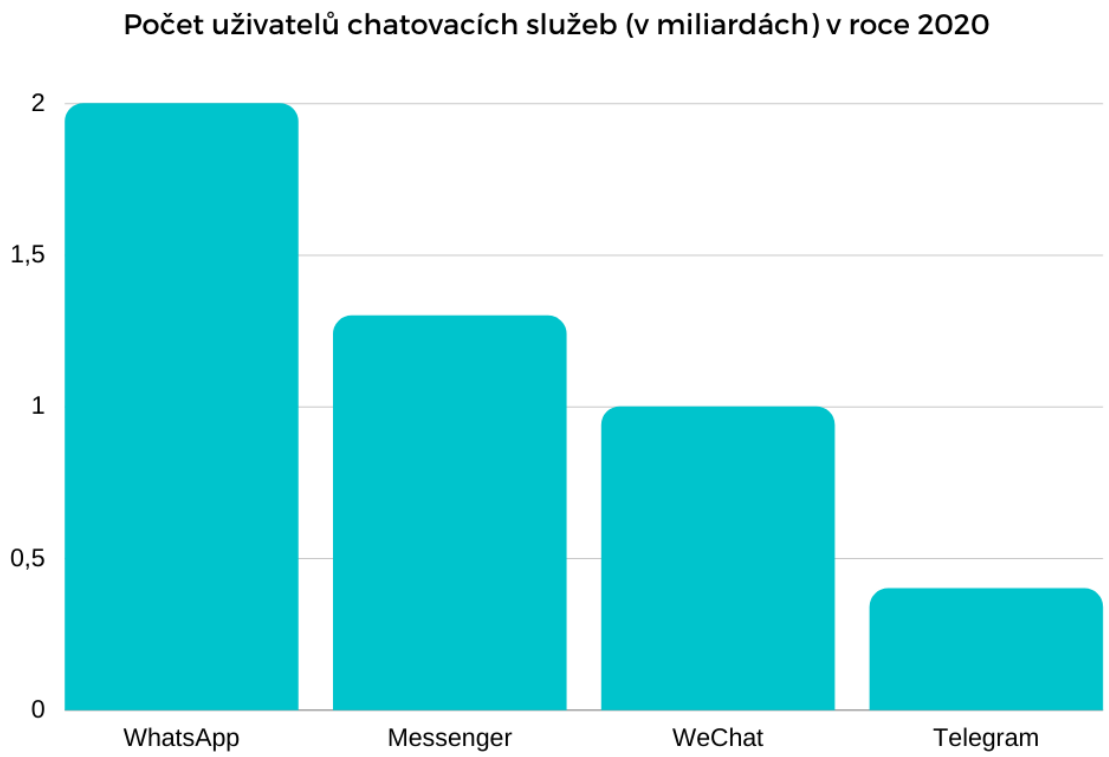
2.1.2 Tlačítkové mobilní telefony se speciálním softwarem

Průzkumy ukazují, že většina uživatelů, včetně těch nevidomých, používá mobilní telefon s dotykovým displejem. Online průzkum ***Smartphones in visual impairment*** z roku 2014 [10] ukazuje, že chytrý telefon s dotykovým displejem používá 101 (77%) ze 131 zrakově postižených účastníků průzkumu. V dalším online průzkumu ***A Survey on the Use of Mobile Applications for People Who Are Visually Impaired*** z roku 2017 zmíněném již v kapitole **Úvod** 1 používá mobilní telefon s dotykovým diplejem až 247 (95.4%) z 259 zrakově postižených účastníků průzkumu. Můžeme tak pozorovat odklon zrakově postižených uživatelů od tlačítkových mobilních telefonů, jejichž uživatelé jsou převážně lidé vyššího věku. Hladká plocha dotykových displejů však může představovat velké problémy i ostatním nevidomým uživatelům, jejichž hlavním smyslem pro komunikaci s mobilním zařízením je hmat, a pro které může být pouhá hlasová podpora nedostačující. Proto se stále vyrábí řada tlačítkových mobilních telefonů s dedikovaným softwarem, který je speciálně upraven pro potřeby zrakově postiženým uživatelům. Za zmínku zde stojí mobilní telefon BlindShell Classic 2.2, na kterém běží *kmenová aplikace* a tím pádem i implementovaný prototyp. Podobné mobilní telefony jsou například SmartVision2, Felix Phone, Lucia a Corvus RG160.

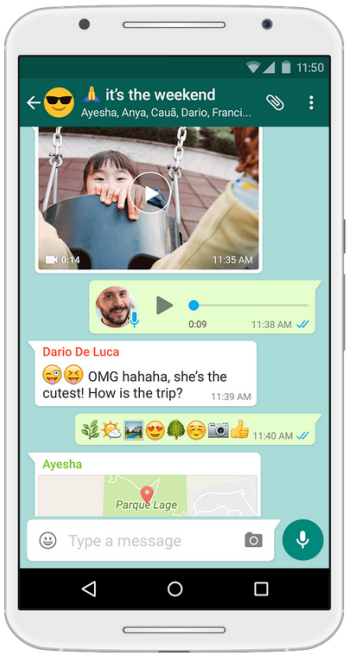
Obrázek 2.2: Mobilní telefon Blindshell Classic. Obrázek stáhnutý z `https://www.blindshell.cz/eshop/blindshell-classic` dne 25. 7. 2020

2.2 Populární chatovací služby

Obrázek 2.3 znázorňuje počet uživatelů tří nejpopulárnějších chatovacích služeb (tzn. těch s nejvíce uživateli), kterými jsou aplikace WhatsApp, Facebook Messenger a WeChat [3]. Na grafu je znázorněna i chatovací služba Telegram, jejíž API je použito k implementaci aplikace. Zmíněné chatovací služby poskytují až na menší rozdíly velmi podobné pole funkcionalit. Umožňují zasílání textových zpráv a multimediálních souborů, vytváření a spravování skupinových konverzací, komunikování s ostatními uživateli pomocí hlasových hovorů atd. I jejich uživatelská rozhraní vypadají velmi podobně, což ukazují obrázky 2.4, 2.5, 2.6 a 2.7 znázorňující uživatelská rozhraní jednotlivých chatovacích služeb. Chatovací služba Telegram, která v počtu uživatelů výrazně zaostává za ostatními chatovacími službami, byla k implementaci aplikace použita z toho důvodu, že se jedná o open-source (tzn. zdrojový kód chatovací služby Telegram je pro vývojáře volně dostupný). Dalším důvodem bylo její jednoduché a dobře dokumentované API, které je blíže popsáno v kapitole **Implementace** 4.

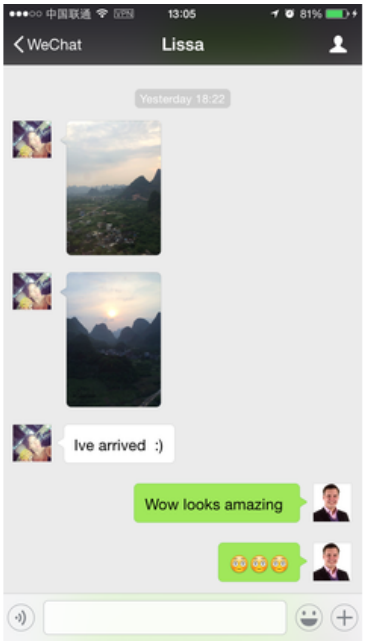


Obrázek 2.3: Obrázek znázorňující počet uživatelů populárních chatovacích služeb v roce 2020

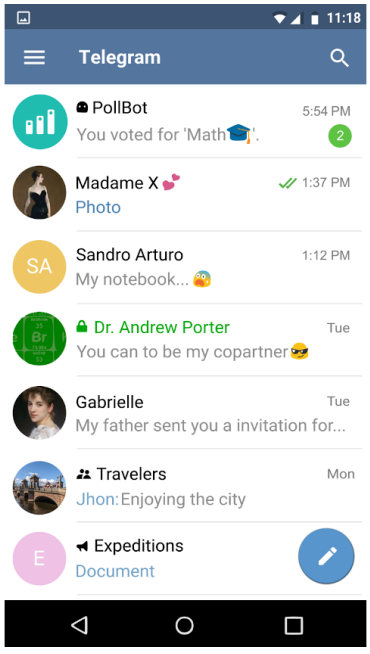


Obrázek 2.4: Ukázka uživatelského rozhraní mobilní aplikace WhatsApp. Obrázek stáhnutý z `https://www.whatsapp.com` dne 25. 7. 2020

Obrázek 2.5: Ukázka uživatelského rozhraní mobilní aplikace Facebook Messenger. Obrázek stáhnutý z `https://www.messenger.com/features` dne 25. 7. 2020



Obrázek 2.6: Ukázka uživatelského rozhraní mobilní aplikace WeChat. Obrázek stáhnutý z `https://en.m.wikipedia.org/wiki/WeChat` dne 25. 7. 2020



Obrázek 2.7: Ukázka uživatelského rozhraní mobilní aplikace Telegram. Obrázek stáhnutý z `https://cs.wikipedia.org/wiki/Telegram_(software)` dne 25. 7. 2020

2.3 Funkční požadavky na aplikaci

Po několika konzultacích s vedoucím projektu a po schůzce s nevidomým expertem Michalem Jelínkem z organizace SONS ČR, byly vyhodnoceny následující funkční požadavky na aplikaci:

1. Kompletní autentizace uživatele u chatovací služby Telegram: aplikace by kromě přihlášení a odhlášení uživatele měla umožňovat i jeho registraci a podporovat dvoufázovou autentizaci.
2. Zobrazení konverzací: uživatel by měl mít možnost zobrazit si nějaký seznam všech svých konverzací.
3. Zobrazení zpráv vybrané konverzace: uživatel by měl mít možnost zobrazit si nějaký seznam všech zpráv vybrané konverzace.
4. Odesílání zpráv do konverzace: aplikace by měla umět posílat a přijímat textové zprávy, hlasové zprávy (včetně jejich přehrávání) a fotografie
5. Vytvoření a správa skupinové konverzace: uživatel by měl mít možnost vytvořit novou skupinovou konverzaci a spravovat její členy (tzn. přidávání a odebírání jejích členů).
6. Zobrazení a změna profilu: aplikace by měla uživateli umožňovat zobrazení a modifikaci jeho jména, jeho globální uživatelského jména a jeho profilové fotografie.

Kapitola 3

Návrh

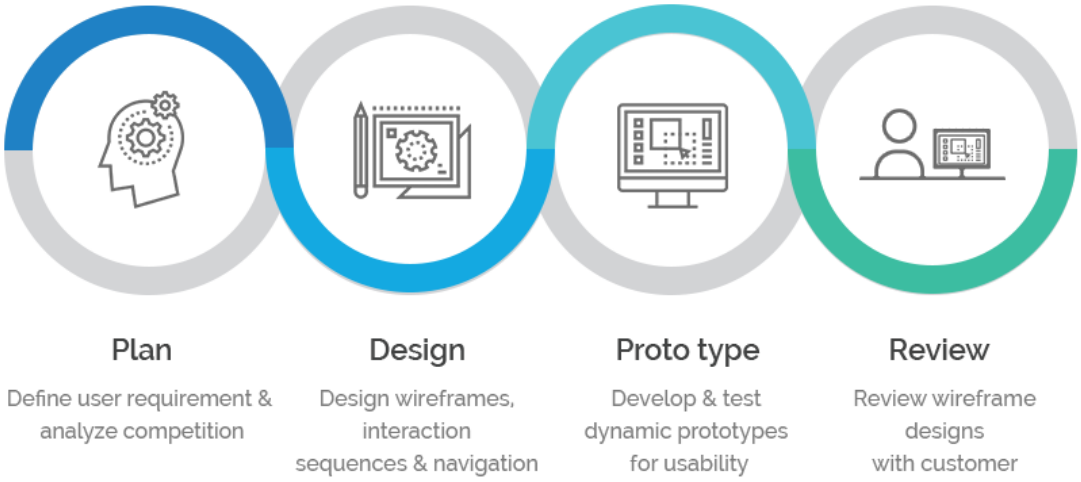
V této kapitole jsou funkční požadavky definované v kapitole **Analýza** 2 použity k návrhu jednotlivých sekcí aplikace.

3.1 Principy návrhu mobilních aplikací pro nevidomé

Univerzální návrh mobilních aplíkací je takový návrh, jehož implementace může být používána všemi lidmi bez ohledu na jejich postižení. Zásadními principy univerzálního návrhu jsou jeho intuitivnost pro uživatele, jeho lehká pochopitelnost a odolnost vůči případným uživatelovým nevalidním vstupům. [8]

3.1.1 User-centered design

User-centered design je návrh plně přizpůsobený potřebám uživatele. Zjednodušený proces návrhu znázorňuje obrázek 3.1. Po spuštění procesu dochází k detailní specifikaci potřeb potenciálních uživatelů výsledného produktu (v případě bakalářské práce se jedná o mobilní aplikaci), velký důraz je kladen na to, kdo bude produkt využívat, k jakému účelu ho bude používat a za jakých podmínek ho bude používat. V další fázi procesu dochází k navržení mobilního aplikace a následnému vytvoření testovatelného prototypu. V poslední fázi procesu je implementovaný prototyp otestován cílovými uživateli a data z tohoto testování jsou použita k budoucímu vývoji produktu. [7]



Obrázek 3.1: Zjednodušený proces user-centered designu. Obrázek stáhnutý z `https://itspresso.com/5-benefits-of-user-centered-design-process-to-use-it/` dne 19. 7. 2020

3.2 Struktura aplikace

V této sekci jsou popsány jednotlivé části aplikace z uživatelského hlediska, jejichž návrh je odvozen od návrhu kmenové aplikace. Pro popis jednotlivých částí jsou použity diagramy, které dodržující náledující konvence:

- Dvě šipky jdoucí z dvou položek přímo proti sobě značí dvě položky stejného menu.
- Položky spouštějící speciální akce jsou označeny zobáčkovitými závorkami <>. Akce <keyboard> prezentuje uživateli klávesnici pro mauální zadání nějaké hodnoty. Akce <list of items> reprezentuje menu stejných položek neznámé velikosti. Akce <audio player> a <audio recorder> pak naznačují, že daná položka slouží pro přehrávání nebo nahrávání audia, kterým je v aplikaci hlasová zpráva.
- Samotné šipky s případným popisem vycházející z položek pak určují, co se stane po potvrzení konkrétní položky (nepopsané šipky většinou značí pouhé otevření nového menu po rozkliknutí dané položky). Někdy jsou pro seskupení šipek se stejným popisem nebo prostě jen pro větší přehlednost jednotlivé šipky znázorněny barevně.
- Někdy je kromě pouhé architektury aplikace nutné zaznamenat i průchod touto architekturou, jedná-li se například o popis procesu autentizace.

K tomu slouží zelená BPMN komponenta pro start procesu a červená BPMN komponenta pro jeho ukončení.

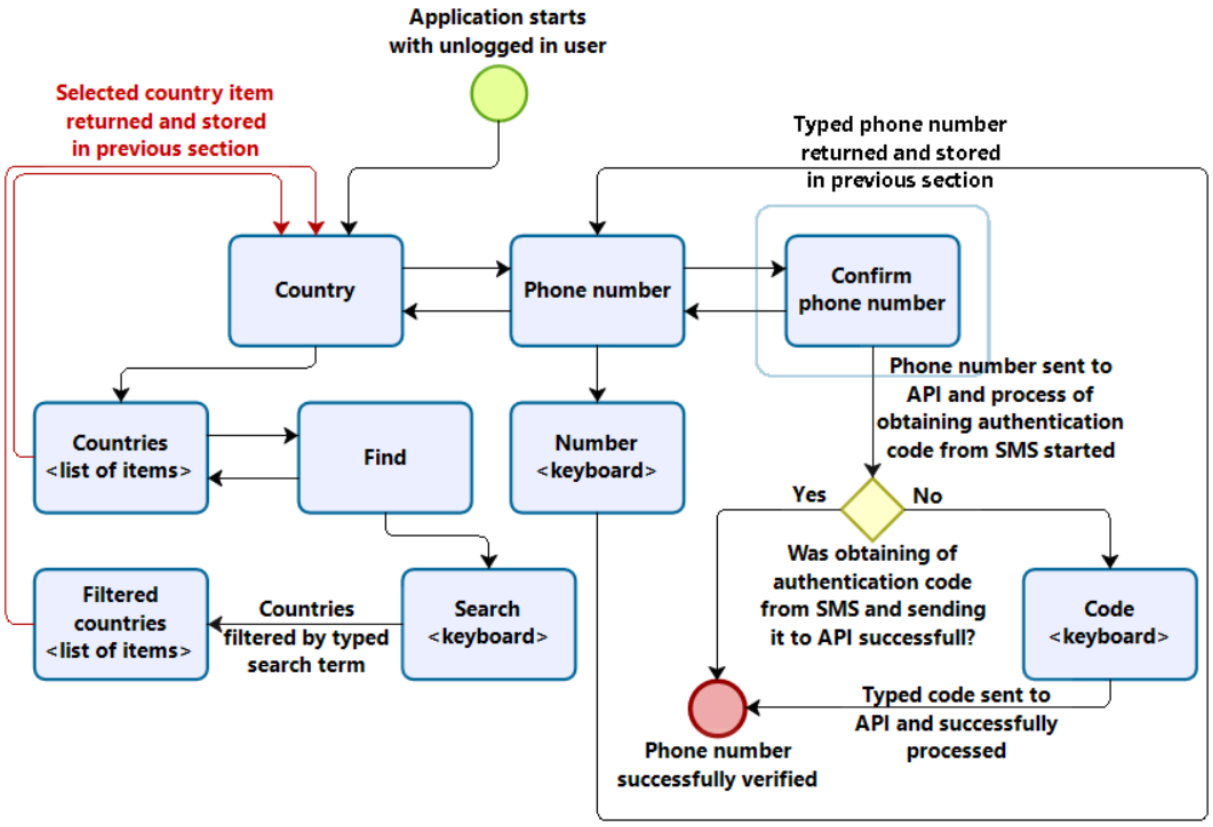
Uživatel se bude v aplikaci pohybovat pomocí navigačních tlačítek na tlačítkovém mobilním telefonu. Tlačítky DOLEVA a DOPRAVA se bude pohybovat uvnitř menu, tlačítkem POTVRDIT rozklikne položku menu nebio potvrdí ukončení nějaké akce a tlačítkem ZPĚT se vrátí do předešlého menu.

3.2.1 Autentizace uživatele

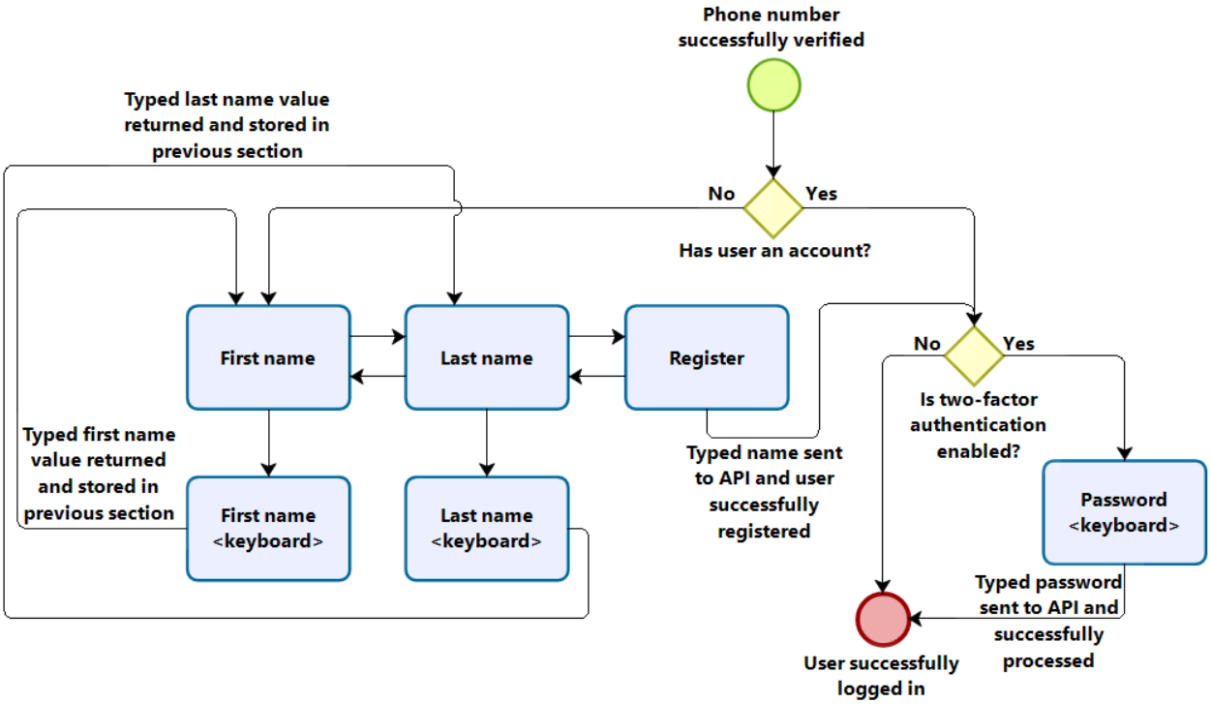
Po spuštění aplikace nepřihlášeným uživatelem proběhne proces autentizace, který je rozdělen na ***proces verifikace mobilního telefonu*** a následný ***proces dodatečné autentizace***.

V ***procesu verifikace mobilního telefonu*** znázorněného diagramem 3.2 je uživateli prezentováno menu o třech položkách. Položka ***Country*** reprezentuje zemi uživatelova mobilního operátora a slouží ke správnému určení jeho telefonní předvolby. Po jejím rozkliknutí je otevřeno menu se zeměmi, z kterých může uživatel vybírat, a s položkou pro filtrování daných zemí. V položce ***Phone number*** uživatel manuálně zadá své telefonní číslo. Položka ***Confirm phone number*** pak slouží k potvrzení a odeslání zadaných hodnot z předchozích dvou položek ke zpracování.

Proces dodatečné autentizace znázorněný diagramem 3.3 umožňuje uživateli bez uživatelského účtu registrovat se do chatovací služby Telegram. Pro registraci je uživateli prezentováno menu s položkami ***First name***, ***Last name*** a ***Register***. Zadané hodnoty ve ***First name*** a ***Last name*** (nepovinná položka) formují jméno uživatele, které je prezentováno ostatním uživatelům chatovací služby Telegram. Položka ***Register*** pak slouží k potvrzení a odeslání zadaných hodnot z předchozích dvou položek ke zpracování. ***Proces dodatečné autentizace*** kromě registrace umožňuje i dvoufázovou autentizaci. Při dvoufázové autentizaci je po uživateli požadováno heslo, které musí manuálně zadat.



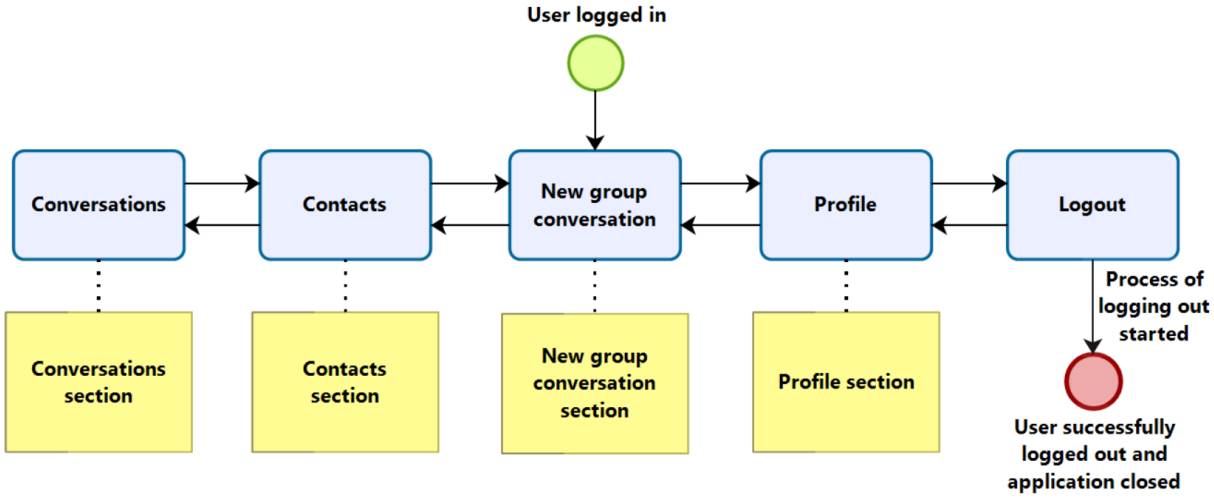
Obrázek 3.2: Diagram znázorňující architekturu jednotlivých obrazovek v procesu verifikace mobilního telefonu.



Obrázek 3.3: Diagram znázorňující architekturu jednotlivých obrazovek v procesu dodatečné autentizace po úspěšné verifikaci mobilního telefonu.

3.2.2 Hlavní menu

Po úspěšné autentizaci je uživateli prezentováno hlavní menu aplikace o pěti položkách znázorněné diagramem 3.4. Položka ***Conversations*** spouští sekci **Konverzace** 3.2.3. Položka ***Contacts*** spouští sekci **Kontakty** 3.2.4. Položka ***New group conversation*** spouští sekci **Nová skupinová konverzace** 3.2.5. Položka ***Profile*** spouští sekci **Profil** 3.2.6. Položka ***Logout*** odhlásí uživatele a ukončí aplikaci.

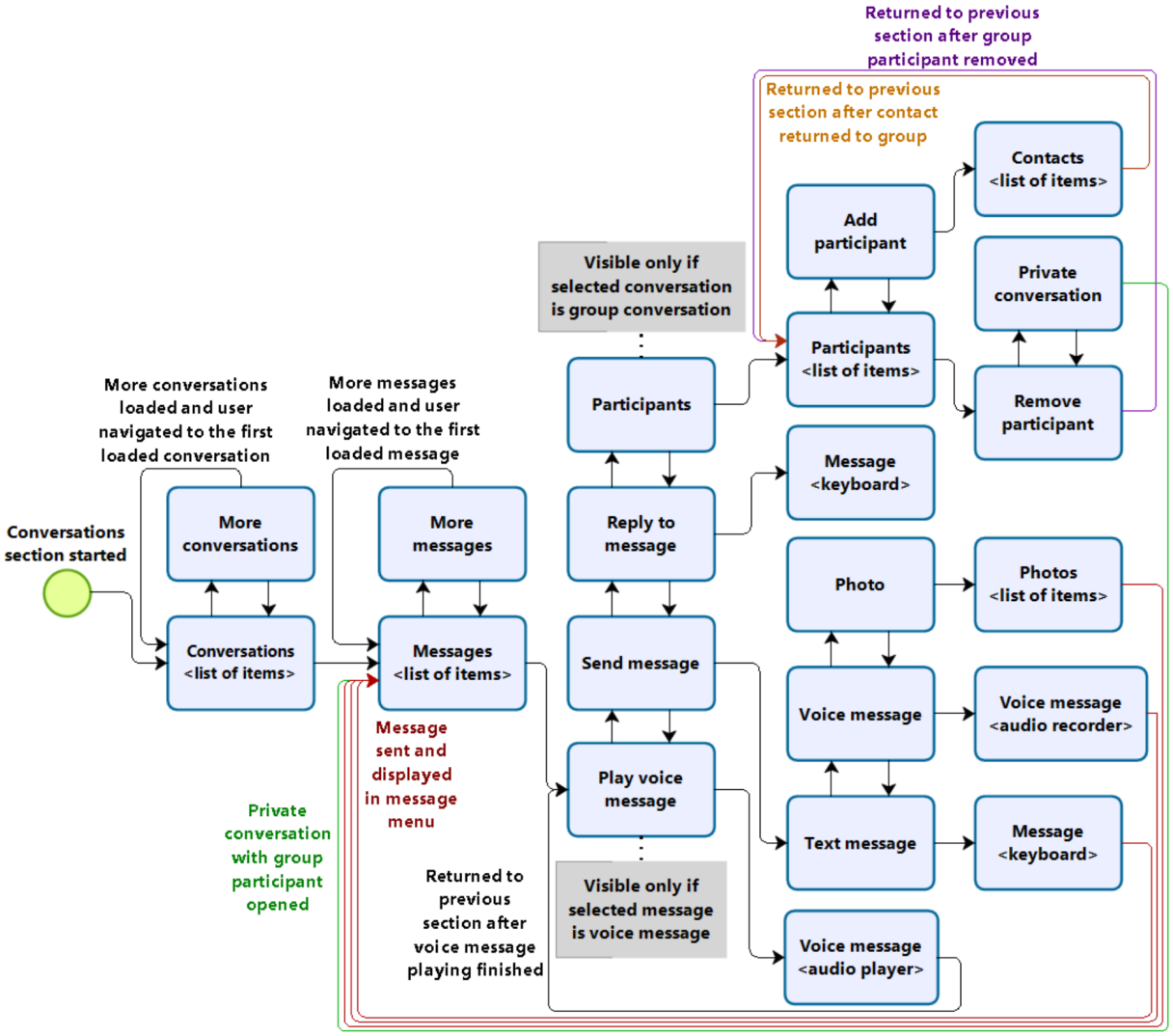


Obrázek 3.4: Diagram znázorňující architekturu jednotlivých obrazovek v hlavním menu přihlášeného uživatele.

3.2.3 Sekce Konverzace

Sekce **Konverzace** zázorněná diagramem 3.5 je nejdůležitější částí aplikace sloužící k zobrazení všech konverzací, zobrazení a posílání zpráv a provádění dalších akcí souvisejících s konverzacemi. Po spuštění sekce je uživateli prezentován seznam konverzací, jejichž počet je kvůli výkonnosti omezen na dvacet. K načtení více konverzací slouží položka ***More conversations***, která načte dvacet dalších konverzací. Konverzace jsou řazeny podle aktivity od nejaktuálnějších k nejdéle neaktivním. Po otevření jedné z konverzací je uživateli prezentován seznam jejích zpráv, jehož počet je stejně jako u seznamu konverzací omezen na dvacet. K načtení více zpráv slouží položka ***More messages***, která načte dvacet dalších zpráv. Zprávy jsou řazeny od nejnovějších k nejstarším a zobrazuje se u nich odesílatel, hlavní obsah zprávy, datum a čas doručení a v případě odchozí zprávy také její stav odesílání. Po rozkliknutí jedné ze práv je uživateli prezentováno menu s akcemi souvícejících s danou konverzací a rozkliknutou zprávou. Akce ***Play voice message*** přehraje aktuálně rozkliknutou hlasovou zprávu s možností pozastavit přehrávání či ho o deset sekund přetočit dopředu nebo dozadu. Položka s touto akcí je viditelná pouze pokud aktuální rozkliknutou zprávou je hlasová zpráva. Akce ***Send message*** přesměruje uživatele na menu s konkrétními typy zpráv k odeslání. Odesílaná zpráva může být textová zpráva, hlasová zpráva nebo vybraný obrázek z mobilního telefonu. Akce ***Reply to message*** umožnuje uživateli textovou zprávou odpovědět na aktuálně rozkliknutou zprávu. Akce ***Participants***, viditelná pouze pokud je daná konverzace skupinovou konverzací, umožňuje uživateli zobrazení členů skupinové konverzace, jejich přidávání a odebírání. Po jejím rozkliknutí je uživateli prezentováno menu se všemi členy skupiny a s položkou ***Add participant***, která otevře menu s kontakty z mobilního telefonu, které používají chatovací službu Telegram a po jejichž

rozkliknutí je daný kontakt přidán do skupinové konverzace. Po rozkliknutí člena skupiny je otevřeno menu s dvěma položkami ***Private conversation*** a ***Remove participant***. Položka ***Private conversation*** otevře, případně předtím vytvoří, pokud neexistuje, soukromou konverzaci s daným členem skupiny. Položka ***Remove participant*** pak daného člena odstraní ze skupiny.

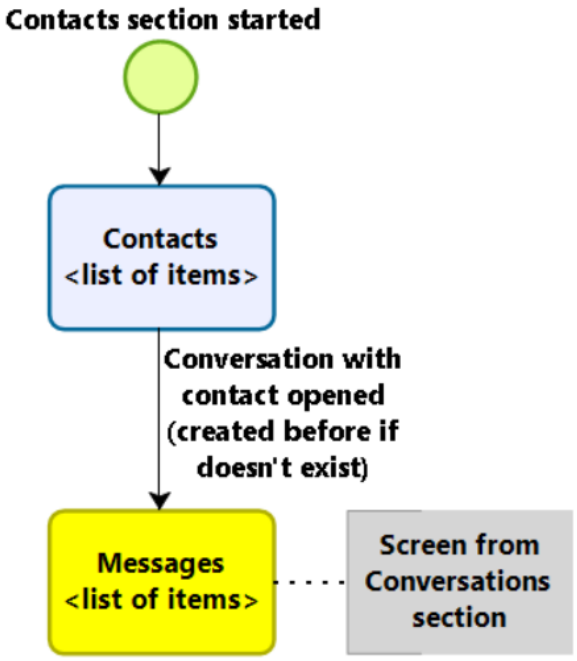


Obrázek 3.5: Diagram znázorňující architekturu jednotlivých obrazovek v sekci Konverzace. Pozn.: Diagram je pro jeho rozumné zobrazení přetočen na bok.

3.2.4 Sekce Kontakty

Sekce **Kontakty** znázorněná diagramem 3.6 slouží ke snadnému přístupu k soukromým konverzacím s kontakty z uživatelova mobilního telefonu využívajícími chatovací službu Telegram. Po spuštění sekce je uživateli prezentováno menu s kontakty, po jejichž rozkliknutí je otevřena soukromá konverzace s

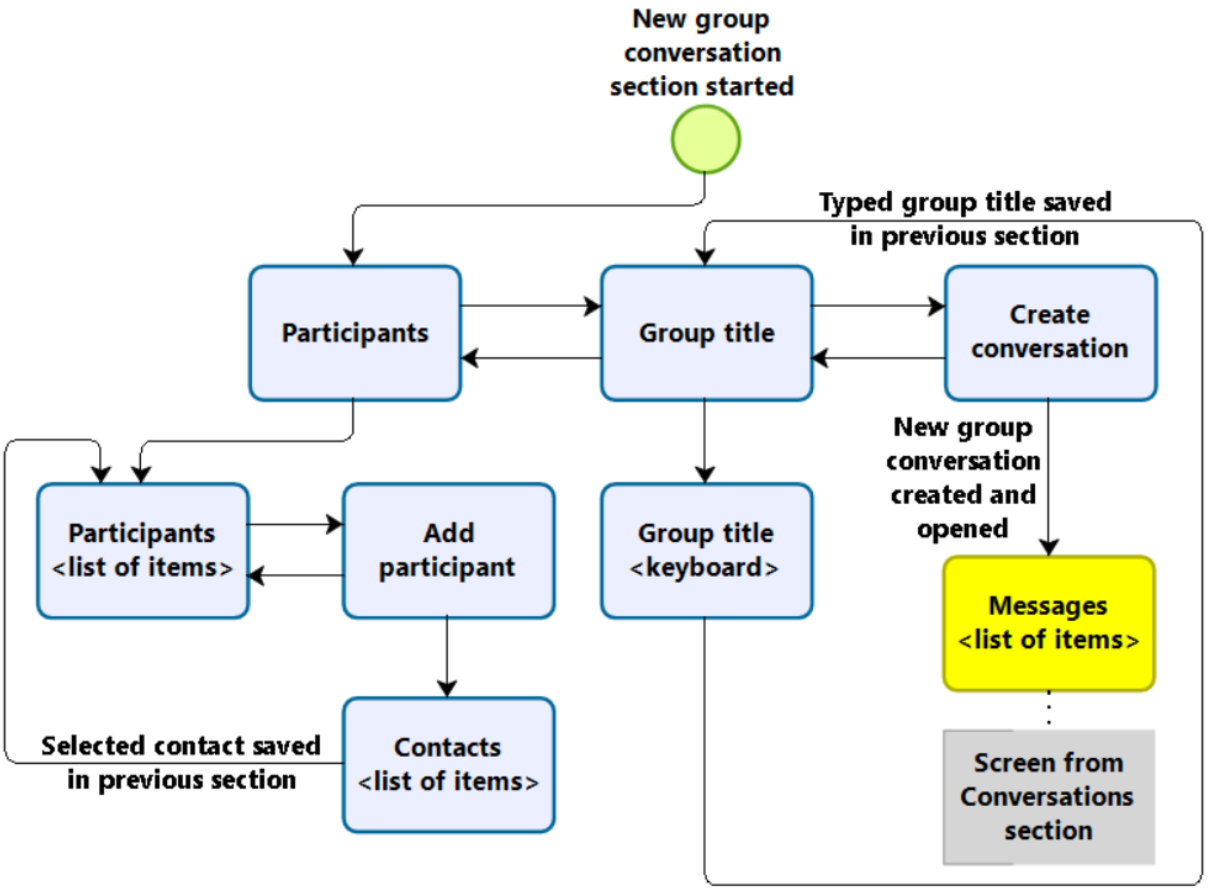
daným kontaktem. Pokud daná konverzace neexistuje, tak je před svým otevřením nejprve vytvořena. Obrazovka, v níž se konverzace otevře, je součástí sekce **Konverzace** 3.2.3.



Obrázek 3.6: Diagram znázorňující architekturu jednotlivých obrazovek v sekci Kontakty.

3.2.5 Sekce Nová skupinová konverzace

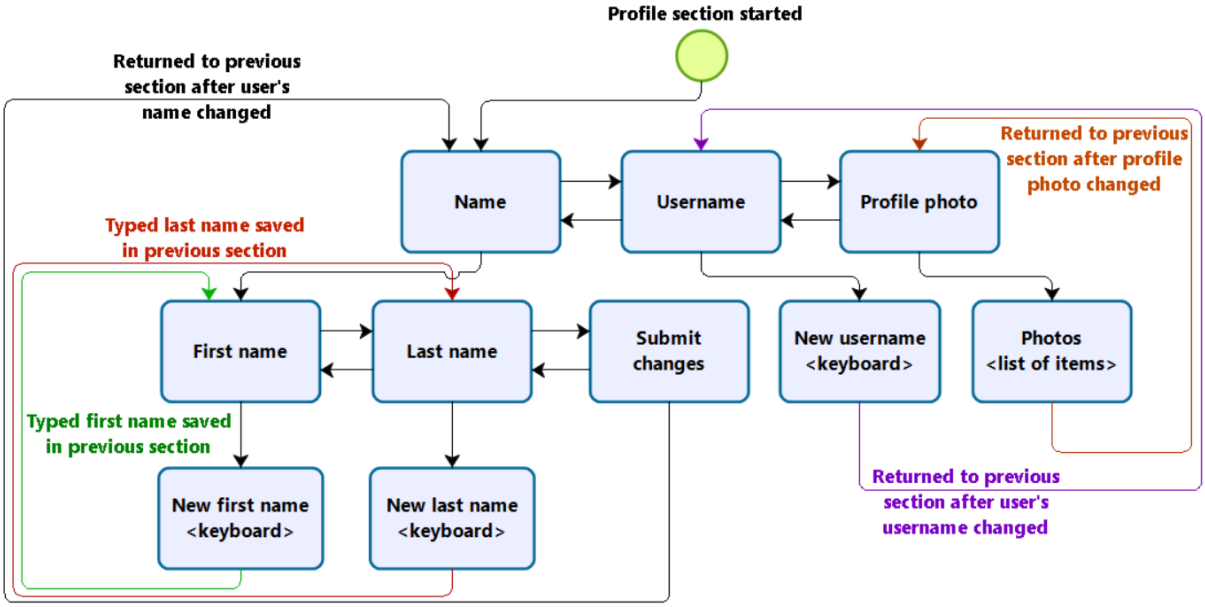
Sekce **Nová skupinová konverzace** znázorněna diagramem 3.7 umožňuje uživateli vytvořit novou skupinovou konverzaci. Po spuštění sekce je otevřeno menu se třemi položkami ***Participants***, ***Group title*** a ***Create conversation***. Položka ***Participants*** slouží pro zvolení členů nové skupiny. Obsahuje menu, v němž je nejprve jen jedna položka ***Add participant***, která zobrazuje menu s kontakty z mobilního telefonu využívajícími chatovací službu Telegram. Po rozkliknutí jednoho z kontaktů je daný kontakt uložen do předchozího menu k položce ***Add participant***. Položka ***Group title*** v hlavním menu pak určuje název skupinové konverzace, který uživatel manuálně zadá, a který bude prezentován ostatním uživatelům skupiny. Nakonec položka ***Create conversation*** zpracuje zadané hodnoty z předchozích dvou položek hlavního menu a otevře nově vytvořenou skupinovou konverzaci. Obrazovka, v níž se konverzace otevře, je součástí sekce **Konverzace** 3.2.3.



Obrázek 3.7: Diagram znázorňující architekturu jednotlivých obrazovek v sekci Nová skupinová konverzace.

3.2.6 Sekce Profil

Sekce **Profil** znázorněna diagramem 3.8 zobrazuje uživatelův profil a umožňuje jeho úpravu. Množství prezentovaných a editovatelných informací bylo omezeno na jméno uživatele, jeho globální jméno (username) a profilový obrázek. Po spuštění skece je uživateli prezentováno menu se třemi položkami ***Name***, ***Username*** a ***Profile photo***. Položka ***Name*** otevírá další menu se třemi položkami ***First name***, ***Last name*** a ***Confirm changes***. První dvě položky ukazují aktuální jméno uživatele a jsou editovatelné. Položka ***Confirm changes*** pak potvrzuje případné změny ve jméně. Položka ***Username*** v hlavním menu této sekce, také editovatelná, pak prezentuje uživateli jeho globální jméno, pod kterým ho může každý uživatel chatovací služby Telegram najít. Položka ***Profile photo*** pak umožňuje uživateli nastavit si či změnit jeho profilový obrázek. Po jejím rozkliknutí je otevřeno menu s fotografiemi z mobilního telefonu a po potvrzení je vybraná fotografie nastavena jako profilový obrázek v chatovací službě Telegram.



Obrázek 3.8: Diagram znázorňující architekturu jednotlivých obrazovek v sekci Profil.

Kapitola 4

Implementace

V této sekci jsou poznatky z kapitol **Analýza** 2 a **Návrh** 3 použity k implementaci testovatelného prototypu.

K porozumění této kapitole je od čtenáře požadována znalost základních konceptů objektově orientovaného programování a technická orientace ve vývoji mobilních aplikací pro operační systém Android, jehož implementační komponenty budou v této kapitole často zmiňovány.

4.1 Použité technologie

Testovatelný prototyp je stejně jako *kmenová aplikace* napsaný v programovacím jazyce Java. Ke komunikaci s chatovací službou Telegram je pak použita knihovna TdLib, která je detailněji popsaná v sekci Knihovna TdLib. K implementaci aplikace bylo využíváno i oficiálních Android tutoriálů od společnosti Google [1]. Další technologie využívány implementovaným prototypem budou zmíněny v následujících sekcích této kapitoly.

4.2 Kmenová aplikace z technického hlediska

V této sekci budou představeny některé důležité komponenty a implementační koncepty *kmenové aplikace*, detailní popis implementace kmenové aplikace zde však nebude.

Hlavním stavebním kamenem *kmenové aplikace* jsou čtyři abstraktní komponenty BlindActivity, BlindMenuActivity, BlindMenuFragment a BlindFragmentAdapter. Komponenta BlindActivity je abstraktní implementací fragmentové aktivity, kterou v *kmenové aplikace* i v testovatelném prototypu rozšiřují aktivity s nějakou speciální funkcionalitou jako je například přehrávání hudby nebo načítání nějakých dat a jejich následné přeposlání jiné aktivitě. Abstraktní implementaci fragmentové aktivity reprezentovanou komponentou BlindMenuActivity pak rozšiřují aktivity, které potřebují uživateli prezentovat nějaké statické menu sloužící pro jeho navigaci v *kmenové aplikace*. Komponenta BlindMenuActivity používá k implementaci statického menu abstrakní implementaci fragmentu BlindMenuFragment a abstraktní implementaci fragmentového adaptéru BlindFragmentAdapter, z čehož vyplývá, že

jednotlivými položkami statického menu jsou fragmenty rozšiřující komponentu BlindMenuFragment, a že tyto fragmenty jsou v menu utvářeny pomocí fragmentového adaptéru rozšiřujícího komponentu BlindFragmentAdapter a pomocí Android komponenty ViewPager. Samotný pohyb ve statickém menu po stisknutí navigačních tlačítek mobilního telefonu je v komponentě BlindMenuActivity realizován implementací listeneru View.OnTouchListener. Kdykoliv od teď bude v této kapitole u nějaké aktivity uvedeno, že prezentuje uživateli statické menu, tak to znamená, že daná aktivita rozšiřuje abstraktní implementaci fragmentové aktivity BlindMenuActivity, v případě neuvedení této skutečnosti to pak znamená, že aktivita rozšiřuje abstraktní implementaci fragmentové aktivity BlindActivity.

4.2.1 Integrace chatovací služby do kmenové aplikace

Aplikace je do *kmenové aplikace* přímo implementovaná jako Java balíček. Kompilace testovatelného prototypu tak nemůže proběhnout samostatně, prototyp musí být zkompilován spolu s celou *kmenovou aplikací*, jíž je přímou součástí. Výhodou tohoto řešení je možnost testovatelného prototypu jednoduše využívat jakékoliv části *kmenové aplikace*.

4.3 Knihovna TdLib

Knihovna TdLib fungující napříč platformami je v podstatě klient chatovací služby Telegram sloužící pro usnadnění vytváření uživatelských aplikací využívajících platformu chatovací služby Telegram. Knihovna může spolupracovat s jakýmkoliv programovacím jazykem, který dokáže spouštět její C funkce. Pro spuštění jejích funkcí v programovacím jazyce Java, který je používán pro implementaci testovatelného prototypu, je použito JNI, které je v případě tohoto projektu již součástí předpřipravené knihovny pro operační systém Android, která je dostupná přímo na stránkách chatovací služby Telegram, a která byla integrována do *kmenové aplikace*. Velikou výhodou knihovny je především její jednoduchost, protože knihovna TdLib se za vývojáře stará o ukládaní dat na lokální úložiště (v případě tohoto projektu se jedná o mobilní zařízení), o šifrování dat a o síťové implementační detaily. [6]

Důležité také je, že knihovna TdLib je plně asynchronní, což znamená, že všechny požadavky do ní poslané budou zpracovávány asynchronně. Způsob, jakým se nevidomí uživatelé orientují v mobilním zařízení, je však velmi synchronní. Nevidomému uživateli musí být jednotlivé kroky, pomocí kterých postupuje při používání mobilního zařízení, prezentovány postupně jeden po druhém s co nejmenší dobou čekání mezi těmito kroky. Této skutečnosti je přizpůsobena i *kmenová aplikace*. Největší výzvou tohoto bakalářského projektu je tedy propojení asynchronní knihovny TdLib s velmi synchronně implementovanou *kmenovou aplikací*.

4.3.1 Komunikace s knihovnou TdLib

Komunikace s knihovnou TdLib probíhá přes knihovnou implementovaného klienta org.drinkless.td.libcore.telegram.Client. Tomuto klientovi je při jeho vytváření předána implementace rozhraní Client.ResultHandler, do jejíž metody onResult budou chodit aktualizace z Telegram API (například aktualizace související s autentizací uživatele, příchod nové zprávy, změna pořadí konverzací apod.). Klientovi mohou být při jeho vytváření předány i dvě implementace rozhraní Client.ExceptionHandler, v implementovaném prototypu jsou však místo nich předány null hodnoty.

```
Client.create(new TelegramUpdatesHandler(), null, null);
```

```
private class TelegramUpdatesHandler implements Client.ResultHandler {
  @Override
  public void onResult(TdApi.Object object) {
    switch (object.getConstructor()) {
      case TdApi.ConcreteUpdate.CONSTRUCTOR:
        // action for concrete update performed
        break;
      ...
    }
```

Implementaci rozhraní Client.ResultHandler přijímají i všechny funkce nabízené knihovnou TdLib, které v ní komponentám aplikace volajícím tyto funkce asynchronně vracejí své výsledky.

```
client.send(new TdApi.Function(parameters),
  new Client.ResultHandler() {
    @Override
    public void onResult(TdApi.Object object) {
      if (object.getConstructor() == TdApi.ConcreteObject.CONSTRUCTOR) {
        // action for returned result TdApi.ConcreteObject performed
      }
      ...
    }
  }
);
```

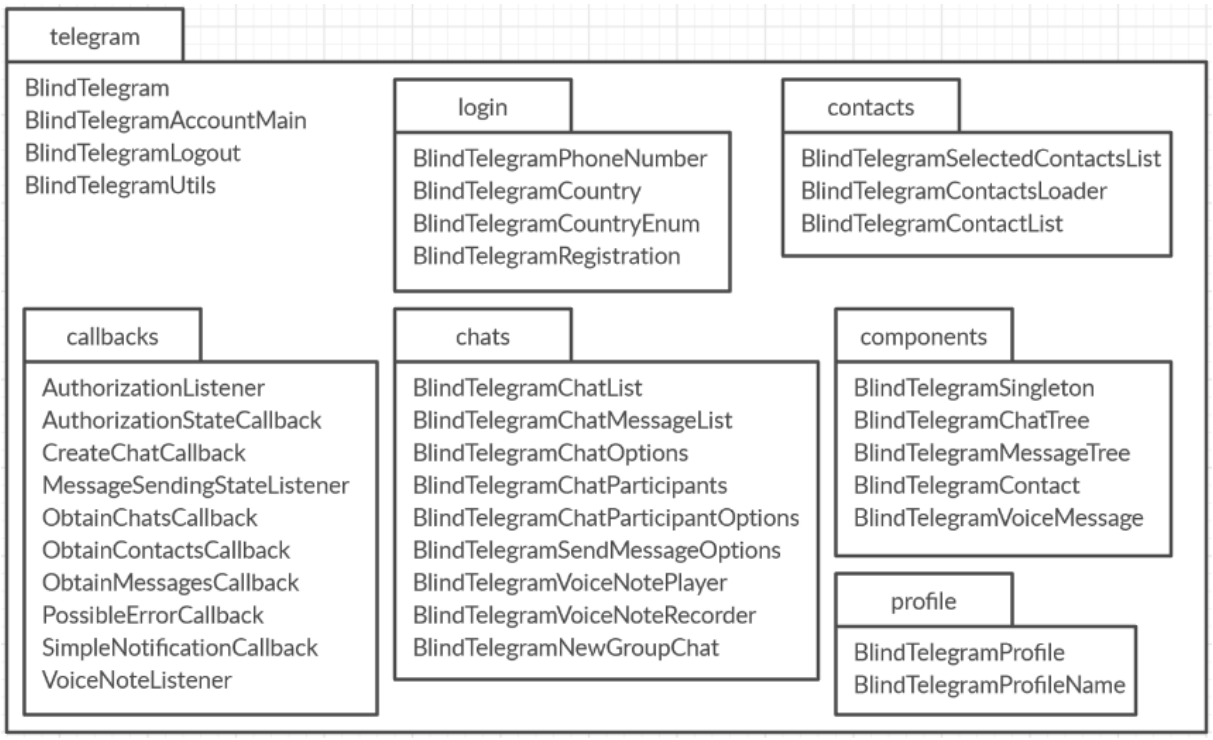
Aktualizace i vrácené výsledky rozšiřují třídu TdApi.Object a jsou identifikovány pomocí číselného identifikátoru označeného jako CONSTRUCTOR.

Singleton

V implementovaném prototypu je ke komunikaci s knihovnou TdLib (tzn. k držení instance dříve zmíněného klienta) využit singleton BlindTelegramSingleton pro jeho jednoduché nastavení a jednoduchou komunikaci s jednotlivými komponentami implementovaného prototypu. Přestože v *kmenové aplikace* nedochází k nečekanému ukončování procesů, která pro lepší výkonnost systému provádí operační systém Android, tak pro produkční aplikaci by bylo zřejmě vhodnější zvolit ke komunikaci s knihovnou TdLib místo singletonu Android komponentu Service, která by byla po případném ukončení dedikovaného procesu opět restartována.

4.4 Implementace jednotlivých částí aplikace

V této sekci je popsána implementace jednotlivých částí aplikace navržených v kapitole **Návrh** 3. Obrázek 4.1 ukazuje všechny třídy aplikace a jejich zařazení do balíčků, ve kterých jsou logicky seskupeny podle jejich účelu.



Obrázek 4.1: Obrázek znázorňující zařazení jednotlivých tříd aplikace do balíčků.

4.4.1 Autentizace uživatele

Po spuštění aplikace je jako první spuštěna aktivita BlindTelegram řídící autentizaci uživatele, kterého přesměrovává do ostatních autentizačních aktivit na základě aktuálního autorizačního stavu asynchronně získávaného z Telegram API přes již zmíněný BlindTelegramSingleton. Získávání aktuálního autorizačního stavu je realizováno pomocí listeneru AuthorizationListener, který je do singletonu registrován z aktivity BlindTelegram v její callback metodě onResume.

```
public interface AuthorizationListener {

  void proceedAfterAuthorizationStateUpdated(int authorizationState);

  void proceedAfterErrorOccurred(int authorizationState, TdApi.Error error);

}
```

Tento listener pak reaguje na dvě události. Jednou je změna aktuálního autorizačního stavu, druhou je nastání nějaké chyby při zpracovávání aktuálního autorizačního stavu způsobené například nevalidním uživatelským vstupem. V prvním případě je aktivita BlindTelegram o události informována pomocí metody listeneru proceedAfterAuthorizationStateUpdated, která vrací nově aktuální autorizační stav. V druhém případě je informována pomocí druhé metody listeneru proceedAfterErrorOccurred, která vrací chybně zpracovaný aktuální stav s objektem TdApi.Error obsahujícím bližší informace o nastané chybě. Při změně autorizačního stavu reaguje aktivita BlindTelegram přesměrováním uživatele do další části autentizačního procesu souvisejícího s novým autorizačním stavem. Při nastání nějaké chyby ve zpracování aktuálního autorizačního stavu pak aktivita reaguje prezentováním nastané chyby uživateli a opakováním části autentizačního procesu souvisejícím s aktuálním autorizačním stavem. Autorizační stavy jsou zpracovávány zasíláním dat, které většinou vyžadují uživatelský vstup, přes singleton do Telegram API. Po úspěšném zpracování autorizačního stavu dochází k jeho změně.

Proces autentizace je rozdělen na proces verifikace mobilního telefonu a proces dodatečné autentizace, jejichž průběh je popsán níže.

Proces verifikace mobilního telefonu

1. Aktivita BlindTelegram registruje v singletonu BlindTelegramSingleton listener AuthorizationListener.
2. Po obdržení autorizačního stavu TdApi.AuthorizationStateWaitPhoneNumber spustí aktivita BlindTelegram aktivitu BlindTelegramPhoneNumber pre-

zentující menu pro manuální zadání uživatelova telefonního čísla, které je nebo bude registrováno v chatovací službě Telegram.

3. Uživatelem vyplněné telefonní číslo je vráceno zpět do aktivity BlindTelegram a posláno přes singleton do Telegram API.

4. Po odeslání validního telefonního čísla obdrží aktivita BlindTelegram autorizační stav TdApi.AuthorizationStateWaitCode, což znamená, že Telegram čeká na autentizační kód zaslaný v SMS zprávě, který je nutný k dokončení procesu verifikace mobilního telefonu.

5. Aktivita BlindTelegram spustí proces automatického získání autentizačního kódu z příchozí SMS zprávy, kdy je vytvořeno vlákno, které po dobu třiceti vteřin každou vteřinu kontroluje přes statické metody pomocné třídy BlindTelegramUtils databázi mobilního telefonu, zda nepřišla nová zpráva od chatovací služby Telegram obsahující onen autentizační kód. V případě nalezení zprávy pak vlákno získaný autorizační kód odešle přes singleton do Telegram API a ukončí svoji činnost. Pokud vlákno do třiceti vteřin onu SMS zprávu nenalezne, tak je ukončeno a uživateli je prezentována obrazovka pro manuální napsání autentizačního kódu.

6. Po úspěšném zpracování autentizačního kódu je proces verifikace mobilního telefonu ukončen.

Proces dodatečné autentizace

1. Pokud uživatel s verifikovaným telefonním číslem není registrován v chatovací službě Telegram, tak aktivita BlindTelegram obdrží autorizační stav TdApi.AuthorizationStateWaitRegistration a spustí aktivitu BlindTelegramRegistration, která uživateli prezentuje menu pro manuální zadání svého jména, pod kterým bude prezentován ostatním uživatelům chatovací služby Telegram.

2. Uživatelem vyplněné jméno je vráceno zpět do aktivity BlindTelegram a posláno přes singleton do Telegram API.

3. Pokud uživatel používá dvoufázovou autentizaci, tak aktivita BlindTelegram obdrží autorizační stav TdApi.AuthorizationStateWaitPassword a prezentuje uživateli obrazovku pro manuální zadání dodatečného hesla.

4. Uživatelem vyplněné heslo je vráceno zpět do aktivity BlindTelegram a posláno přes singleton do Telegram API.

5. Po úspěšném zpracování předchozích kroků nebo po jejich přeskočení, je-li uživatel již registrován a nepoužívá-li dvoufázovou autentizaci, obdrží aktivita BlindTelegram autorizační stav TdApi.AuthorizationStateReady, po kterém přesměruje uživatele do hlavní menu aplikace 4.4.2.

6. Aktivita BlindTelegram je ukončena a s ní i proces dodatečné autentizace.

4.4.2 Hlavní menu

Hlavní menu aplikace reprezentuje aktivita BlindTelegramAccountMain obsahující statické menu s položkami spouštějícími jednotlivé sekce aplikace. Položka LogoutItem pak spouští proces odhlášení uživatele, který je popsán níže.

Proces odhlášení uživatele

1. Položka hlavního menu LogoutItem spustí aktivitu BlindTelegramLogout a ukončí stávající aktivitu BlindTelegramAccountMain, čímž se aktivita BlindTelegramLogout stane jedinou spuštěnou aktivitou celé aplikace.
2. Aktivita BlindTelegramLogout odhlásí přes singleton BlindTelegramSingleton uživatele a poté daný singleton zničí, čímž dojde i k uzavření a zničení Telegram klienta, který je v singletonu používán pro komunikaci s Telegram API.
3. Po dokončení předešlého kroku je aktivita BlindTelegramLogout ukončena, čímž je ukončena i celá aplikace a uživatel je přesměrován zpět do *kmenové aplikace*.

4.4.3 Sekce Konverzace

Z implementačního i uživatelského hlediska se jedná o suverénně nejrozsáhlejší sekci celé aplikace. Pro přehledný popis její implementace je tato sekce rozdělěna do několika logicky oddělených subsekcí.

Dynamické menu s konverzacemi

Po spuštění sekce je dynamické menu s konverzacemi prezentováno uživateli.

Po jakékoliv změně v jakékoliv konverzaci jako je příchod nové zprávy do konverzace, změna názvu konverzace nebo přidání či odebrání člena skupinové konverzace musí být menu s konverzacemi aktualizováno. Klasické statické menu, které je používané v *kmenové aplikaci*, tak není dobrou volbou, protože při každé změně by bylo nutné toto menu celé přetvořit, což by u většího množství položek mohlo vést k výrazným výkonnostním problémům. Pro zobrazení konverzací tak bylo implementováno dynamické menu, které jednoduše reaguje na všechny změny související s konverzacemi.

Dynamické menu s konverzacemi je reprezentováno aktivitou BlindTelegramChatList, která k zobrazení jednotlivých konverzací používá stromovou

strukturu BlindTelegramChatTree. Stromová struktura BlindTelegramChatTree je vytvořena po úspěšném přihlášení uživatele v singletonu BlindTelegramSingleton a je přes singleton přístupná všem částem aplikace. BlindTelegramChatList je pak aktivita, která využívá vlastní implementaci rozhraní View.OnTouchListener pro pohyb (změna atributu currentNode ukazující na aktuálně vybraný uzel v BlindTelegramChatTree) ve stromové struktuře po stisknutí navigačních tlačítek na mobilním telefonu. Jednotlivé uzly stromové struktury (položky dynamického menu) mají v sobě uloženy údaje o konverzacích potřebné k jejich řazení, včetně identifikátoru konverzace chatId, který je v případě rozkliknutí konverzace poslán do nově spuštěné aktivity BlindTelegramChatMessageList prezentující dynamické menu se zprávami vybrané konverzace. Položkou dynamického menu je i položka More conversations, která po svém rozkliknutí zavolá přes singleton metodu z Telegram API, která do stromové struktury BlindTelegramChatTree načte 20 dalších konverzací. Hlavním smyslem takto implementovaného dynamického menu je, že při jeho změně dochází zpravidla k modifikaci pouze jedné položky menu namísto celého menu.

Techničtějším popisem implementace tohoto dynamického menu je, že aktivita BlindTelegramChatList používá layout blind_telegram_chats_activity.xml, který obsahuje textové pole TextView. Obsah textového pole je pak měněn na základě atributu currentNode ukazujícího na aktuálně vybraný uzel ve stromové struktuře BlindTelegramChatTree. Po každé změně atributu currentNode je navíc uživateli přehrána hlasová identifikace aktuálně vybraného uzlu stromové struktury (položky dynamického menu).

Dynamické menu se zprávami vybrané konverzace

Implementace dynamického menu se zprávami vybrané konverzace je téměř stejná jako implementace dynamického menu s konverzacemi. Toto menu je reprezentováno aktivitou BlindTelegramChatMessageList, která k zobrazení jednotlivých zpáv vybrané konverzace používá stromovou strukturu BlindTelegramMessageTree. Změnou oproti dynamickému menu s konverzacemi je zaplnění textového pole TextView obsaženého v layoutu blind_telegram_chat_message_list_activity.xml. Textové pole zde obsahuje odesílatele, stav odeslání, obsah zprávy a čas jejího doručení. Podobně jako dynamické menu s konverzacemi i toto menu obsahuje položku More messages sloužící pro načtění 20 dalších zpráv. Stromová struktura BlindTelegramMessageTree je vytvořena v singletonu BlindTelegramSingleton, když je poprvé zapotřebí a následně je po celou dobu své životnosti redukována v době jejího nevyužívání na maximální počet 1000 zpráv. Po rozkliknutí jakékoliv položky reprezentující konkrétní zprávu je spuštěna aktivita BlindTelegramChatOptions, které je předán identifikátor vybrané konverzace, identifikátor rozklinuté zprávy a případně ještě identifikátor souboru hlasové zprávy, je-li rozkliknutá zpráva hlasovou zprávou.

Možné akce ve vybrané konverzaci

Statické menu s položkami různých akcí souvisejících s vybranou konverzací je prezentováno aktivitou BlindTelegramChatOptions. Je-li vybrána hlasová zpráva, tak je součástí menu i položka Play voice message, která spouští proces přehrání hlasové zprávy. Položka Send message spouští proces odeslání zprávy do konverzace a položka Reply to message proces odpovědi na vybranou zprávu. Nakonec položka Participants spouští subsekci správa členů skupinové konverzace, jejíž implementace je, stejně jako jsou průběhy všech zmíněných procesů, popsána níže.

Proces přehrání hlasové zprávy:

1. Je spuštěna aktivita BlindTelegramVoiceNotePlayer, které je předán identifikátor souboru hlasové zprávy.
2. Aktivita získá podle identifikátoru souboru přes singleton BlindTelegramSingleton z Telegram API soubor vybrané hlasové zprávy.
3. Pokud není soubor lokálně stažen v paměti mobilního telefonu, tak ho aktivita BlindTelegramVoiceNotePlayer stáhne.
4. Aktivita BlindTelegramVoiceNotePlayer používá layout blind_telegram_voice_player_activity.xml, který obsahuje dvě textová pole. Jedno textové pole slouží k prezentování aktuálního stavu přehrávání hlasové zprávy uživateli (Loading, Playing, Paused) a druhé k zobrazení průběhu hlasové zprávy. Uživateli je umožněno díky vlastní implementaci listeneru View.OnTouchListener hlasovou zprávu stopnout nebo ji o 10 sekund přetáčet pomocí navigačních tlačítek mobilního telefonu.
5. Po ukončení přehrávání hlasové zprávy je uživatel vrácen do aktivity BlindTelegramChatOptions.

Pozn.: Protože audio soubory v chatovací službě Telegram používají kompresní formát Opus a jsou obsaženy v Ogg kontejneru, tak je nemožné k jejich přehrání nebo nahrání využít populární Android komponentu pro přehrávání a nahrávání multimediálních souborů zvanou MediaPlayer, jejíž aktuální verze používaná v kmenové aplikaci tuto relativně moderní kombinaci nepodporuje. Je tak využita vlastní, v kmenové aplikaci již obsažená, implementace přehrávače těchto audio souborů používající knihovnu Opus for Android, která pomocí JNI komunikuje s nativní knihovnou Opus codec sloužící pro kódování a dekódování multimediálních souborů kompresním formátem Opus [4]. Konkrétní třída využita pro přehrávání hlasových zpráv se jmenuje BlindOpusPlayer, třídou pro jejich nahrávání je potom třída BlindOpusRecorder. Popis implementace těchto tříd však nebude součástí této bakalářské práce, protože se nejedná o přímo implementované části testovatelného prototypu.

Proces odeslání zprávy do konverzace:

1. Po rozkliknutí položky Send message je uživateli prezentováno menu s položkami Text message, Voice message a Photo.

2. Po rozkliknutí položky Text message je uživateli prezentována obrazovka pro manuální zádání textové zprávy, potvrzením napsané hodnoty je daná zpráva přes singleton BlindTelegramSingleton poslána do Telegram API, čímž dojde k vytvoření nové textové zprávy v konverzaci.

3. Alternativně ke kroku 2, rozkliknutím položky Voice message je spuštěna aktivita BlindTelegramVoiceNoteRecorder, která ihned začne nahrávat hlasovou zprávu, jejíž data jsou ukládána do audio souboru vytvořeným v mobilním zařízením. Po ukončení nahrávání je lokální cesta k audio souboru hlasové zprávy poslána přes singleton do Telegram API, čímž dojde k vytvoření nové hlasové zprávy v konverzaci.

4. Alternativně ke krokům 2 a 3, rozkliknutím položky Photo je spuštěna sekce kmenové aplikace, jejíž implementace zde nebude popsána, a která uživateli prezentuje statické menu s fotografiemi z mobilního telefonu. Po vybrání jedné z nich je cesta k sobouru oné fotografie odeslána přes singleton do Telegram API, čímž je do konverzace poslána nová zpráva reprezentována vybranou fotografií.

5. Po splnění kroku 2, 3 nebo 4 je uživatel přesměrován zpět do dynamického menu se zprávami vybrané konverzace používané v aktivitě BlindTelegramChatMessageList a atribut currentNode (ukazatel na konkrétní položku v dynamickém menu) ve stromové struktuře BlindTelegramMessageTree je nastaven, aby ukazoval na odeslanou zprávu.

6. Aktivity BlindTelegramVoiceNoteRecorder a BlindTelegramChatOptions při vrácení uživatele do aktivity BlindTelegramChatMessageList ukončí svojí činnost.

Proces odpovědi na vybranou zprávu:

1. Rozkliknutím položky Reply to message je uživateli prezentována obrazovka pro manuální napsání textové zprávy.

2. Po potvrzení napsané zprávy je zpráva odeslána přes singleton do Telegram API, čímž je vytvořena nová textová zpráva odpovídající na dříve rozkliknutou zprávu.

3. Uživatel je přesměrován zpět do dynamického menu se zprávami vybrané konverzace prezentované aktivitou BlindTelegramChatMessageList a atribut currentNode stromové struktury BlindTelegramMessageTree ukazuje na položku reprezentující odeslanou zprávu.

4. Aktivia BlindTelegramChatOptions je po vrácení uživatele do aktivity BlindTelegramChatMessageList ukončena.

Správa členů skupinové konverzace

Po spuštění této subsekce je spuštěna aktivita BlindTelegramChatParticipants, která uživateli prezentuje statické menu s položkami reprezentujícími jednotlivé členy skupinové konverzace a položkou Add participant pro přidání nového člena skupiny. Subsekce by se dala z implementačního hlediska rozdělit na tři procesy, na proces přidání nového člena do skupinové konverzace, proces odebrání člena ze skupinové konverzace a proces otevření soukromé konverzace s členem skupinové konverzace. Všechny tři procesy jsou popsány níže.

Proces přidání nového člena do skupinové konverzace:

1. Rozkliknutím položky Add participant spustí aktivita BlindTelegramChatParticipants aktivitu BlindTelegramContactsLoader, která je v kombinací s aktivitou BlindTelegramContactList využita k vybrání kontaktu z mobilního telefonu, který je registrován u chatovací služby Telegram. Vybraný kontakt je poté vrácen do aktivity BlindTelegramChatParticipants. Detailnější fungování aktivit BlindTelegramContactsLoader a BlindTelegramContactList je popsáno v sekci Kontakty 4.4.4
2. Po obdržení vybraného kontaktu je přes singleton BlindTelegramSingleton zavolána metoda z Telegram API, které jsou jako parametry předány identifikátory skupinové konverzace a vybraného kontaktu.
3. Pokud je přidání nového člena do skupinové konverzace úspěšné, tak metoda z Telegram API vrací aktivitě BlindTelegramChatParticipants objekt TdApi.Ok a uživatel je informován o úspěšném přidání nového člena. Pokud je přidání neúspěšné, tak metoda z Telegram API vrací aktivitě BlindTelegramChatParticipants objekt TdApi.Error a uživatel je informován o neúspěšném přidání nového člena do skupinové konverzace.

Proces odebrání člena ze skupinové konverzace:

1. Rozkliknutím položky nějakého člena dojde ke spuštění aktivity BlindTelegramChatParticipantOptions, která uživateli prezentuje statické menu s položkami Remove participant a Private conversation.
2. Po rozkliknutí položky Remove participant je uživatel přesměrován zpět do aktivity BlindTelegramChatParticipants, do které je poslán aktuálně vybraný uživatel reprezentovaný objektem datové třídy BlindTelegramContact.
3. Z aktivity BlindTelegramChatParticipants je přes singleton zavolána metoda z Telegram API, které jsou jako parametry předány identifikátory skupinové konverzace a vybraného kontaktu k odstranění.

4. Pokud je odebrání člena ze skupinové konverzace úspěšné, tak je metodou z Telegram API vrácen aktivitě BlindTelegramChatParticipants objekt TdApi.Ok informující uživatele o úspěšném odebrání člena. Pokud je odebrání člena neúspěšné, tak je metodou z Telegram API vrácen aktivitě BlindTelegramChatParticipants objekt TdApi.Error informující uživatele o neúspěšném odebrání člena ze skupinové konverzace.

Proces otevření soukromé konverzace s členem skupinové konverzace:

1. Po rozkliknutí položky nějakého člena je spuštěna aktivita BlindTelegramChatParticipantOptions prezentující uživateli statické menu s položkami Remove participant a Private conversation.
2. Rozkliknutím položky Private conversation dosáhneme naprosto stejné funkcionality jako u rozkliknutí kontaktu v sekci Kontakty, popis její implementace je tedy již zaznamenán zde.

4.4.4 Sekce Kontakty

V této sekci je uživateli prezentováno statické menu, jehož položkami jsou kontakty z mobilního telefonu vyfiltrované na uživatele chatovací služby Telegram. Po rozkliknutí jakéhokoliv kontaktu je otevřena soukromá konverzace s daným kontaktem.

O vyfiltrování kontaktů se stará aktivita BlindTelegramContactsLoader, statické menu s těmito kontakty pak vytváří aktivita BlindTelegramContactList. Implementace těchto dvou aktivit a proces otevření soukromé konverzace s vybraným kontaktem jsou popsány níže.

Aktivity BlindTelegramContactsLoader a BlindTelegramContactList

Kombinace aktivit BlindTelegramContactsLoader a BlindTelegramContactList je používána i v jiných sekcích aplikace, které potřebují zobrazit vyfiltrovaný seznam kontaktů. Tato kombinace pak nabízí dvě lehce rozdílné funkcionality, které jsou identifikovány pomocí boolean hodnoty newChat, jejíž vychozí hodnota je nastavena na false. Pro jedinečnost těchto dvou aktivit je tak popis jejich implementace oddělen od popisu implementace jednotlivých procesů, jejichž jsou součástí.

Bez ohledu na vybranou funkcionalitu je nejdříve proveden stejný proces zobrazení vyfiltrovaných kontaktů s následujícími kroky:

1. Aktivita BlindTelegramContactsLoader získá po svém spuštění přes statickou metodu pomocné třídy BlindTelegramUtils z databáze mobilního telefonu všechny uložené kontakty.

2. Kontakty získané v předešlém kroku jsou aktivitou BlindTelegramContactsLoader importovány přes singleton BlindTelegramSingleton do Telegram API, kde jsou do kontaktů uživatele chatovací služby Telegram uloženy pouze ty kontakty, jejichž telefonní číslo je registrováno v chatovací službě Telegram.
3. Aktivita BlindTelegramContactsLoader získá přes singleton z Telegram API všechny kontakty, které zde byly uloženy v předchozím kroku (jedná se o kontakty z mobilního telefonu vyfiltrované na kontakty registrované v chatovací službě Telegram).
4. Data z vyfiltrovaných kontaktů jsou aktivitou BlindTelegramContactsLoader uloženy do objektů datové třídy BlindTelegramContact implementující rozhraní Parcelable, které umožňuje posílání objektů dané třídy mezi aktivitami.
5. Objekty datové třídy BlindTelegramContact jsou odeslány do nově spuštěné aktivity BlindTelegramContactList.
6. Aktivita BlindTelegramContactList přijme objekty datové třídy BlindTelegramContact, z jejichž dat následně vytvoří statické menu s vyfiltrovanými kontakty, které pak prezentuje uživateli.

Pokud je boolean hodnota newChat nastavena na true (tzn. hodnota newChat byla do aktivity BlindTelegramContactsLoader explicitně poslána z aktivity, která ji spouštěla), tak kombinace aktivit BlindTelegramContactsLoader a BlindTelegramContactList slouží pro případné vytváření a otevírání soukromých konverzací s vyfiltrovanými kontakty. Funkcionalita kombinace pak vypadá následovně:

1. Aktivita BlindTelegramContactsLoader je ukončena.
2. Po rozkliknutí nějaké položky statického menu prezentovaným aktivitou BlindTelegramContactList dojde ke spuštění procesu otevření soukromé konverzace s vybraným kontaktem, jehož průběh je popsán níže v této sekci.

Pokud je boolean hodnota newChat nastavena na false (tzn. aktivitě BlindTelegramContactsLoader nebyla z aktivity, která ji spouštěla, explicitně poslána žádná hodnota newChat), tak kombinace aktivit BlindTelegramContactsLoader a BlindTelegramContactList slouží pro vybrání určitého kontaktu, který je vrácen do aktivity spouštějící tuto kombinaci. Funkcionalita kombinace pak vypadá následovně:

1. Po rozkliknutí nějaké položky statického menu prezentovaným aktivitou BlindTelegramContactList je objekt datové třídy BlindTelegramContact uložený ve vybrané položce poslán zpět do aktivity BlindTelegramContactsLoader.

2. Aktivita BlindTelegramContactList ukončí po odeslání objektu svoji činnost.

3. Aktivita BlindTelegramContactsLoader pošle obdržený objekt do aktivity, která aktivitu BlindTelegramContactsLoader spustila.

4. Aktivita BlindTelegramContactsLoader ukončí po odeslání objektu svoji činnost.

Proces otevření soukromé konverzace s vybraným kontaktem

1. Uživatel rozklikne jeden z kontaktů ve statickém menu prezentovaným aktivitou BlindTelegramContactList.

2. Aktivita BlindTelegramContactList zavolá přes singleton BlindTelegramSingleton metodu z Telegram API, které jako parametr předá identifikátor vybraného kontaktu již obsažený v rozkliknuté položce kontaktu ze statického menu. Identifikátor vybraného kontaktu je identifikátorem uživatele chatovací služby Telegram, kterého rozkliknutá položka ze statického menu kontaktů reprezentuje.

3. V případě že žádná konverzace s daným kontaktem neexistuje, tak ji metoda z Telegram API vytvoří. Aktivitě BlindTelegramContactList je pak vrácen identifikátor existující či nově vytvořené konverzace.

4. Aktivita BlindTelegramContactList otevře konverzaci s vybraným kontaktem spuštěním aktivity BlindTelegramChatMessageList ze sekce Konverzace 4.4.3, které předá identifikátor dané konverzace získaný v předešlém kroku.

5. Aktivita BlindTelegramContactList ukončí svoji činnost, čímž dojde i k ukončení tohoto procesu.

4.4.5 Sekce Nová skupinová konverzace

Po spuštění sekce je spuštěna aktivita BlindTelegramNewGroupChat, která uživateli prezentuje statické menu s položkami Participants, Group title a Create conversation. Rozkliknutím položky Participants dojde ke spuštění procesu zvolení členů nové skupinové konverzace, jehož průběh je popsán níže. Položka Group title prezentuje uživateli obrazovku pro manuální zadání názvu nové skupinové konverzace. Rozkliknutím položky Create conversation je pak spuštěn proces vytvoření a otevření nové skupinové konverzace, jehož průběh je rovněž popsán níže.

Proces zvolení členů nové skupinové konverzace

1. Po rozkliknutí položky Participants spustí aktivita BlindTelegramNewGroupChat aktivitu BlindTelegramSelectedContactsList, které předá seznam aktuálně zvolených členů nové skupiny, kteří jsou reprezentováni jako objekty datové třídy BlindTelegramContact.
2. Aktivita BlindTelegramSelectedContactsList vytvoří z příjatého seznamu objektů datové třídy BlindTelegramContact statické menu, do kterého navíc přidá položku Add participant, a které pak prezentuje uživateli. V případě prvotního spuštění tohoto procesu se tak statické menu skládá pouze z položky Add participant.
3. Rozkliknutím položky Add participant spustí aktivita BlindTelegramSelectedContactsList aktivitu BlindTelegramContactsLoader, která je v kombinací s aktivitou BlindTelegramContactList využita k vybrání kontaktu z mobilního telefonu registrovaným u chatovací služby Telegram a jeho vrácení do aktivity BlindTelegramSelectedContactsList. Detailnější fungování aktivit BlindTelegramContactsLoader a BlindTelegramContactList je popsáno v sekci Kontakty 4.4.4
4. Vybraný kontakt je přidán do statického menu aktivity BlindTelegramSelectedContactsList (statické menu je celé vytvořeno znovu).
5. Ukončením aktivity BlindTelegramSelectedContactsList jsou pak všichni vybraní členové nové skupinové konverzace poslány do aktivity BlindTelegramNewGroupChat, kde jsou uloženi.

Pozn.: Kroky 3 a 4 mohou být pro zvolení dalších členů nové skupinové konverzace opakovány, kliknutím na již vybraného člena ve statickém menu prezentovaným akitivitou BlindTelegramSelectedContactsList je uživateli prezentován dialog pro jeho případné odebrání.

Proces vytvoření a otevření nové skupinové konverzace

1. Po rozkliknutí položky Create conversation zkontroluje aktivita BlindTelegramNewGroupChat, zda počet členů nové skupinové konverzace zvolených v procesu zvolení členů nové skupinové konverzace je roven alespoň 2. Dále je zkontrolováno, zda uživatel zadal v položce Group title nějaký název pro novou skupinovou konverzaci.
2. Po úspěšné validaci z předešlého kroku zavolá položka Create conversation přes singleton BlindTelegramSingleton metodu z Telegram API, které jsou jako parametry předány pole identifikátorů zvolených členů nové skupinové konverzace a uživatelem zadaný název nové skupinové konverzace.

3. Nová skupinová konverzace je vytvořena a metoda z Telegram API vrací aktivitě BlindTelegramNewGroupChat identifikátor nově vytvořené skupinové konverzace

4. Aktivita BlindTelegramNewGroupChat otevře skupinovou konverzaci spuštěním aktivity BlindTelegramChatMessageList ze sekce Konverzace 4.4.3, které předá identifikátor dané skupinové konverzace získaný v předešlém kroku.

5. Aktivita BlindTelegramNewGroupChat ukončí svoji činnost, čímž je ukončen i tento proces.

4.4.6 Sekce Profil

Po spuštění sekce Profil je spuštěna aktivita BlindTelegramProfile, která uživateli prezentuje statické menu s položkami Name, Username a Profile photo. Průběhy procesů pro změnu jména, uživatelského jména (username) a profilové fotografie jsou popsány níže.

Proces změny jména

1. Po rozkliknutí položky Name je spuštěna aktivita BlindTelegramProfileName prezentující uživateli statické menu s položkami First name, Last name a Submit changes.

2. Položka First name prezentuje uživateli obrazovku pro manuální zadání jeho křestního jména.

3. Položka Last name prezentuje uživateli obrazovku pro manuální zadání jeho příjmení.

4. Položka Submit changes zavolá přes singleton BlindTelegramSingleton metodu z Telegram API, které jsou jako parametry předány vyplněné hodnoty z položek First name a Last name.

5. Pokud je změna jména úspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfileName objekt TdApi.Ok a uživatel je informován o úspěšné změně jména. Pokud je neúspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfileName objekt TdApi.Error a uživatel je obeznámen o neúspěchu při změně jména.

6. Aktivita BlindTelegramProfileName ukončí svoji činnost, čímž je ukončen i tento proces.

Proces změny uživatelského jména

1. Po rozkliknutí položky Username je uživateli prezentována obrazovka pro manuální zadání jeho uživatelského jména.
2. Po potvrzení nového uživatelského jména je zavolána přes singleton BlindTelegramSingleton metoda z Telegram API, které je jako parametr předána vyplněná hodnota z položky Username.
3. Pokud je změna uživatelského jména úspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfile objekt TdApi.Ok a uživatel je informován o úspěšné změně uživatelského jména. Pokud je neúspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfile objekt TdApi.Error a uživatel je obeznámen o neúspěchu při změně uživatelského jména.

Proces změny profilové fotky

1. Po rozkliknutí položky Profile photo je spuštěna sekce z kmenové aplikace, která zde ani v kapitole Návrh 3 není popsána, protože není přímou součástí testovatelného prototypu. Nicméně ona sekce z kmenové aplikace prezentuje uživateli po několika statických menu seznam fotografií uložených v mobilním telefonu.
2. Po vybrání jedné z fotografií je cesta k souboru dané fotografie vrácena do aktivity BlindTelegramProfile.
3. Po obdržení cesty k souboru vybrané fotografie zavolá aktivita BlindTelegramProfile přes singleton BlindTelegramSingleton metodu z Telegram API, které je jako parametr předána ona cesta k souboru vybrané fotografie.
4. Pokud je změna profilové fotografie úspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfile objekt TdApi.Ok a uživatel je informován o úspěšné změně profilové fotografie. Pokud je neúspěšná, tak metoda z Telegram API vrací aktivitě BlindTelegramProfile objekt TdApi.Error a uživatel je obeznámen o neúspěchu při změně profilové fotografie.

Kapitola 5

Testování

V této kapitole je popsán způsob testování implementované aplikace a jsou zde zaznamenány nalezené nesrovnalosti, které nemusí být přímo chybami, během testování aplikace. Výsledky testování aplikace se nachází v příloze X této bakalářské práce.

Testování implementovaného prototypu proběhlo až přibližně dva týdny před termínem odevzdání bakalářské práce, nalezené chyby a návrhy na opravu tak nestihly být do jejího odevzdání realizovány. Místem konání tohoto testování byla pražská centrála organizace SONS ČR, což je organizace složená ze zrakově postižených lidí spolupracujících pro zlepšení kvality života všem zrakově postiženým lidem [5]. Aplikaci pak testovali 3 zkušení zrakově postižení testeři (z toho dva zcela nevidomí) z této organizace.

5.1 Dotazníky

Testerům byly před samotným testováním aplikace i po něm prezentovány dva níže popsané dotazníky.

5.1.1 Dotazník před testem

Dotazník před testem, který byl testerům prezentován před testováním aplikace, slouží pro získání bližších informací o testujících subjektech.

Obsah dotazníku:

1. Jaký je Váš věk?
2. K jakému účelu používáte mobilní telefon?
   a. Pouze na volání, případně psaní SMS zpráv
   b. Využívám i některé další funkce mobilního telefonu (budík, webový prohlížeč, hudební přehrávač apod.)
3. Jaký typ mobilního telefonu používáte?
   a. Tlačítkový
   b. Dotykový

4. Na jakém operačním systému Váš mobilní telefon běží?
5. Jakou technologii používáte pro zjednodušení Vaší orientace na mobilním telefonu?
6. Jste obeznámen/a se stále rozvíjejícím se a dnes velmi populárním trendem online chatovacích služeb využívajících k přenosu zpráv internet?
   a. Ano
   b. Ne
7. Používal/a jste nebo používáte nějakou online chatovací službu?
   a. Ano + název služby
   b. Ne
8. Jaké funkce od testované aplikace očekáváte?

5.1.2 Dotazník po testu

Dotazník po testu je testerům prezentován po testování aplikace a slouží pro zachycení jejich celkového dojmu z aplikace a z jejího testování.

Obsah dotazníku:

1. Jak byste ohodnotil/a náročnost jednotlivých úkolů na škále od 1 (velmi snadné) do 5 (velmi obtížné)? Co Vám dělalo největší problém?
2. Co Vám během plnění testovacích úkolů dělalo největší problém?
3. Jak byste zhodnotil obtížnost orientace v testované aplikaci na škále od 1 (velmi snadná) do 5 (velmi obtížná)?
4. Postrádal/a jste v aplikaci nějakou funkci?
5. Jaký je Váš celkový pocit z testované aplikace? Používal/a byste ji?

5.2 Testovací úkoly

Testerům jsou jeden po druhém prezentovány testovací úkoly, které jsou popsány níže.

1. Jděte do kontaktů a otevřete konverzaci s jakýmkoliv kontaktem.
2. Odešlete do konverzace hlasovou zprávu trvající alespoň 15 vteřin.

3. Přehrajte si odeslanou hlasovou zprávu (zkuste přehrávání pozastavit a přetočit ho dopředu nebo dozadu).

4. Najděte v konverzacích skupinovou konverzaci s názvem „Test group".

5. Odstraňte ze skupinové konverzace jakéhokoliv člena kromě toho s názvem „Blind Phone“.

6. Znovu přidejte odstraněného člena do skupinové konverzace.

7. Přejděte v aplikaci do profilu přihlášeného uživatele.

8. Změňte uživatelského jméno (username) přihlášeného uživatele na „Blindphonetest“.

5.3 Nalezené nesrovnalosti během testování aplikace

5.3.1 Nekonečné načítání hlasové zprávy při jejím přehrání

Na tuto chybu narazil první tester po spuštění položky pro přehrání hlasové zprávy, kdy její načítání trvalo nekonečně dlouhou dobu. Po opětovném spuštění položky se však hlasová zpráva načetla úspěšně. Ostatní testeři na tuto chybu nenarazili.

Chybu se po testování podařilo přehráváním ostatních hlasových zpráv replikovat, objevila se však jenom u některých hlasových zpráv. Chyba je nejspíš způsobena neobdržením aktualizace o staženém souboru hlasové zprávy z Telegram API. Odstranit by se pravděpodobně dala požadavkem na získání souboru hlasové zprávy z Telegram API po vývojářem zvolené době, po kterou se hlasová zpráva nebyla schopná načíst.

5.3.2 Pojmenování položky Send message

Pro zobrazení menu s položkami odesílajícími různé typy zpráv (textové zprávy, hlasové zprávy, fotografie) slouží položka Send message, jejíž pojmenování zmátlo prvního testera, který si myslel, že ona položka slouží pro odeslání pouze textové zprávy. Je tedy na zvážení, zda by se dána položka měla přejmenovat například na „Send“.

5.3.3 Předvyplnění stávajícího uživatelského jména do obrazovky pro jeho změnu

Při změně uživatelského jména v profilu by se prvnímu testerovi líbilo, kdyby se stávající hodnota uživatelského jména předvyplnila do obrazovky pro zadání nového uživatelského jména. Toho se dá velmi jednoduše docílit předáním oné hodnoty do aktivity pro zobrazení obrazovky pro zadávání uživatelského vstupu, která je implementovaná v kmenové aplikaci.

5.3.4 Filtrování již přidaných členů do skupinové konverzace

Pro přidání nových členů do skupinové konverzace je uživateli prezentováno menu s kontakty z mobilního telefonu, které jsou vyfiltrovány na kontakty používající chatovací službu Telegram. Problémem je, že tyto kontakty nejsou dále filtrovány na kontakty, které se ještě ve skupinové konverzaci nenachází. Ačkoliv aplikace neumožňuje opětovné přidání člena, který se již ve skupinové konverzaci nachází, a dokonce o tom uživatele informuje, tak třetí tester by zmíněné dodatečné filtrování uvítal. Dalo by se toho opět celkem jednoduše docílit předáním seznamu stávajících členu skupinové konverzace aktivitě, která vytváří statické menu s vyfiltrovanými kontakty, která by podle předaného seznamu kontakty dále vyfiltrovala.

Kapitola 6

Závěr

Cílem bakalářské práce, který je blíže popsán v kapitole Úvod, bylo zanalyzovat požadavky na chatovací službu pro nevidomé a tyto požadavky využít k návrhu a implementaci testovatelné prototypu. Implementovaný prototyp pak měl být otestován nevidomými uživateli a výsledky těchto testů zpracovány pro budoucí vývoj aplikace.

Přestože všechny tyto cíle byly splněny, tak aplikace nemůže být bez úprav použita v produkčním prostředí. Jak bylo zmíněno v kapitole Implementace, prototyp používá ke komunikaci s Telegram API singleton, který v sobě přes implementaci Telegram klienta komunikujícího s knihovnou TdLib musí držet autentizační stav. V případě ukončení procesu, ve kterém je singleton spuštěn, operačním systémem Android, nedojde k automatickému restartu singletonu, jak by tomu mohlo být u Android komponenty Service. Pro správné fungování singletonu i po této události by v obnovených aktivitách musel být vytvořen nějaký mechanismus pro obnovení v singletonu uloženého Telegram klienta včetně jeho autentizačního stavu. Dále by musely být obnoveny stromové struktury reprezentující dynamické pole konverzací a zpráv vybrané konverzace, které jsou inicializovány a drženy uvnitř singletonu. Tento mechanismus by však musel být implementován i pro aktivity, které nejsou součástí implementovaného prototypu, ale jsou součástí kmenové aplikace, protože se očekává životnost singletonu i po ukončení aplikace. Veliká rozsáhlost implementovaného prototypu ve spolupráci s docela komplikovaným propojením asynchronní knihovny TdLib a synchronní implementací kmenové aplikace pak pravděpodobně v aplikaci zanechala nějaké chyby, které je nutné pro její další vývoj opravit.

Literatura

[1] Android developers. https://developer.android.com/. Accessed: 2020-07-22.

[2] Messaging apps statistics for 2020. https://kommandotech.com/statistics/messaging-apps-statistics/. Accessed: 2020-07-12.

[3] Most popular global mobile messenger apps as of july 2020, based on number of monthly active users. https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/. Accessed: 2020-07-17.

[4] Opus for android. https://github.com/louisyonge/opus_android. Accessed: 2020-07-25.

[5] Sjednocená organizace nevidomých a slabozrakých Čr. https://www.sons.cz/. Accessed: 2020-07-17.

[6] Telegram api a knihovna tdlib. https://core.telegram.org/. Accessed: 2020-07-17.

[7] User-centered design basics. https://www.usability.gov/what-and-why/user-centered-design.html. Accessed: 2020-07-19.

[8] Martine Abel-Williamson. Universal design. Technical report, WBU (World Blind Union), 2015.

[9] Nora Griffin-Shirley, Devender Banda, Paul Ajuwon, Jongpil Cheon, Jaehoon Lee, Hye Park, and Sanpalei Lyngdoh. A survey on the use of mobile applications for people who are visually impaired. *Journal of visual impairment & blindness*, 111:307–323, 07 2017.

[10] Antonio Filipe Macedo, Laura H. Moreno, Rui S. Silva, and Michael D. Crossland. Smartphones in visual impairment. *Investigative Ophthalmology & Visual Science*, 55(13):4150–4150, Apr 2014.

[11] John Morris and James L. Mueller. Blind and deaf consumer preferences for android and ios smartphones. 2014.

Příloha A

Použité zkratky

- API - Application Programming Interface
- BPMN - Business Process Model and Notation
- TdLib - Telegram Database Library
- JNI - Java Native Interface

Příloha B

Výsledky z testování aplikace

V této příloze se nachází výsledky testování aplikace třemi zrakově postiženými uživateli, které je popsáno v kapitole Testování.

B.1 Tester 1

Prvním testerem je zrakově postižená žena, která však není zcela nevidomá.

B.1.1 Dotazník před testem

1. 33
2. Využívám i některé další funkce mobilního telefonu (budík, webový prohlížeč, hudební přehrávač apod.)
3. Dotykový
4. Android
5. Hlasová podpora a zvětšování
6. Ano
7. Ano, WhatsApp a Messenger
8. Možnost odesílání textových zpráv, hlasových zpráv a fotografií

B.1.2 Dotazník po testu

1. Hodnocení náročnosti testovacích úkolů
   a. 1
   b. 4
   c. 1
   d. 1
   e. 1
   f. 1

   g. 1
   h. 1

2. Pro testera bylo největším problémem nalezení položky pro nahrání a odeslání hlasové zprávy obsažené v menu, které je uživateli prezentováno po rozkliknutí položky Send message. Testera zmátla právě ona položka Send message, kterou považoval za položku pro napsání obyčejné textové zprávy.
3. 2
4. Tester uvedl, že mu v aplikaci žádná funkce nechyběla.
5. Celkový pocit z aplikace je ze strany testera velmi dobrý a kdyby prý používal kmenovou aplikaci, tak by používal i tuto v ní zabudovanou aplikaci.

B.2 Tester 2

Druhým testerem je nevidomý muž.

B.2.1 Dotazník před testem

1. 41
2. Využívám i některé další funkce mobilního telefonu (budík, webový prohlížeč, hudební přehrávač apod.)
3. Dotykový i tlačítkový
4. iOS, Android
5. VoiceOver, Talkback, kmenovou aplikaci
6. Ano
7. Ano, Hangout, Skype, WhatsApp a dříve ICQ
8. Možnost odesílání textových zpráv, hlasových zpráv a fotografií, možnost volání, možnost vytváření skupinových konverzací

B.2.2 Dotazník po testu

1. Hodnocení náročnosti testovacích úkolů
   a. 1

b. 1
c. 1
d. 1
e. 2
f. 2
g. 1
h. 1

2. Tester je obeznámen s fungováním kmenové aplikace, a proto mu žádný úkol vetší problém nedělal.
3. 1
4. Tester uvedl, že mu v aplikaci žádná funkce nechyběla.
5. Celkový pocit z aplikace je ze strany testera velmi dobrý a aplikaci by prý používal.

B.3 Tester 3

Třetím testerem je nevidomý muž.

B.3.1 Dotazník před testem

1. 50
2. Využívám i některé další funkce mobilního telefonu (budík, webový prohlížeč, hudební přehrávač apod.)
3. Dotykový
4. iOS
5. VoiceOver
6. Ano
7. Ano, WhatsApp a Skype
8. Možnost odesílání textových zpráv včetně emoticonu, možnost odesílání multimediálních souborů, možnost sdílení své polohy, možnost přeposílání zpráv, možnost vytváření skupinových konverzací, možnost zobrazení kontaktů vyfiltrovaných na uživatele chatovací služby Telegram

B.3.2 Dotazník po testu

1. Hodnocení náročnosti testovacích úkolů
   a. 1
   b. 1
   c. 2
   d. 1
   e. 2
   f. 2
   g. 1
   h. 1
2. Pro testera bylo největším problémem nalezení položky pro nahrání a odeslání hlasové zprávy, který byl způsoben jeho dezorientací po zobrazení seznamu zpráv vybrané skupinové konverzace.
3. 1
4. Tester uvedl všechny nenaplněné funkce již zmíněné v otázce, jaké funkce od testované aplikace očekává, nacházející se v dotazníku před testem. K nim ještě přidal chybějící možnost mazání zpráv.
5. Celkový pocit z aplikace je ze strany testera dobrý a kdyby prý používal kmenovou aplikaci, tak by používal i tuto aplikaci.

Příloha C

Obsah přiloženého CD

- bachelor_thesis_main.pdf - PDF soubor obsahující text bakalářské práce
- telegram/ - adresář obsahující soubory implementované aplikace