FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Diplomová práce

Mobilní aplikace pro podporu komunity

David Bubik

PLZEŇ

2024

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Diplomová práce

Mobilní aplikace pro podporu komunity

Bc. David Bubik

Vedoucí práce
Ing. Ladislav Pešička

PLZEŇ **2024**

Citace v seznamu literatury:
BUBIK, David. *Mobilní aplikace pro podporu komunity*. Plzeň, 2024. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky. Vedoucí práce Ing. Ladislav Pešička.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. David BUBIK**
Osobní číslo:	**A22N0040P**
Studijní program:	**N0613A140040 Softwarové a informační systémy**
Téma práce:	**Mobilní aplikace pro podporu komunity**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Prozkoumejte vybrané systémy pro komunikaci mezi uživateli v rámci dané komunity.
2. Vyberte vhodné technologie pro systém, který umožní v rámci dané komunity správu uživatelů, skupinový chat, sdílení akcí v kalendáři, sdílení videí, sdílení své geografické polohy a další potřebné činnosti. Systém by měl být koncipován pro snadné využití i staršími uživateli bez hlubších znalostí moderních technologií.
3. Navrhněte systém, splňující předchozí bod zadání, který bude tvořen klienty pro mobilní zařízení na platformě Android a serverem na zvolené platformě.
4. Navržený systém realizujte, ověřte jeho funkcionalitu a navrhněte další možná rozšíření.

Rozsah diplomové práce: **doporuč. 50 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

dodá vedoucí diplomové práce

Vedoucí diplomové práce: **Ing. Ladislav Pešička**
Katedra informatiky a výpočetní techniky

Datum zadání diplomové práce: **8. září 2023**
Termín odevzdání diplomové práce: **16. května 2024**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 11. října 2023

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Plzni dne 22. dubna 2024

. . . . . .
David Bubik

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Abstrakt

Cílem této práce je vytvořit nový komunikační systém pro podporu komunity, který umožňuje správu uživatelů, skupinový chat, sdílení akcí v kalendáři, sdílení videí a sdílení geografické polohy. Tento systém je tvořen klienty pro mobilní zařízení Android a serverem na zvolené platformě. Dále je bezpečný a použitelný pro technicky méně schopné uživatele. V první části práce analyzuji již existující komunikační aplikace a porovnávám jejich vlastnosti. Dále provedu návrh komunikačního systému a vyberu vhodné technologie. Následující část práce se zabývá implementací navrženého systému za pomocí vybraných technologií. Poslední část obsahuje otestování systému a další možná rozšíření.

Abstract

The goal of this work is to create a new communication system for community support that allows user management, group chat, calendar event sharing, video sharing and geo-location sharing. This system consists of clients for Android mobile devices and a server on the chosen platform. It is also secure and usable for less technically capable users. In the first part of the thesis, I analyze existing communication applications and compare their features. Next, I perform the design of the communication system and select the appropriate technologies. The following part of the thesis deals with the implementation of the designed system using the selected technologies. The last part contains the testing of the system and other possible extensions.

Klíčová slova

Komunikační systém • Mobilní klient • Android • Realtime databáze

Poděkování

Tímto bych rád poděkoval Ing. Ladislavu Pešičkovi za odborné vedení, cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracovávání diplomové práce.

Bc. David Bubik,
autor práce
(únor 2024)

Obsah

Úvod 1

V dnešní rychle se měnící společnosti je téměř nutné využívat komunikační nástroje. Příkladem může být telefonování, nebo posílání krátkých zpráv prostřednictvím SMS či mobilních aplikací s využitím Internetu. Pro skupinovou komunikaci lze využít některé z dostupných aplikací, které jsou na tuto problematiku zaměřeny.

Jedním z problémů těchto aplikací je začlenění méně technicky zdatných členů komunity do společné on-line komunikace. Tito lidé mají sice zájem o sdílení svých názorů, ale komplikovanost většiny stávajících aplikačních řešení jim to významně ztěžuje. Nejedná se pouze o posílání zpráv, ale také o sdílení videí, kalendářních akcí, zakládání privátních či veřejných chatovacích místností nebo v některých případech sdílení své geografické polohy.

Další potíží je zajištění bezpečnosti technicky méně zdatných uživatelů. Tito uživatelé většinou nejsou informováni o úskalích veřejných komunikačních aplikací. Pokud jim neznámý uživatel zašle odkaz pro vyplnění citlivých údajů, jsou schopni tyto informace poskytnout. Dále je možné, že nedopatřením budou sdílet svá osobní data, jako jsou fotografie či videa. Je tedy nutné zajistit, že všichni uživatelé aplikace jsou členové komunity a poskytnout jim bezpečné prostředí pro komunikaci v rámci komunity.

V této práci budu analyzovat stávající řešení mobilních aplikací pro podporu komunity, jejich přednosti a nedostatky. Následně bude proveden návrh, implementace a otestování mnou vytvořeného produktu.

Všechny služby systému musí být snadno intuitivně přístupné, ovladatelné a spravovatelné. Dále je nutné vzít v úvahu, že ne všichni potencionální uživatelé jsou technicky zdatní. Nemusí tedy mít založeny účty na známých sociálních sítích nebo vlastnit e-mailovou adresu. Z toho vyplývá, že tyto služby nemůže výsledné řešení striktně vyžadovat.

Vytvořený systém bude mít dvě části. První část bude sloužit pro správu a udělování oprávnění jednotlivým uživatelům. Tato součást systému je zamýšlena v podobě webové aplikace. Samotná komunikace se bude odehrávat v mobilním klientu na platformě Android. Běžný uživatel bude pracovat pouze v mobilním klientu, s webovou aplikací nepřijde do styku. Je tedy nutné, aby mobilní aplikace byla přehledná a

uživatelsky přívětivá.

1.1 Cíle práce

Jedním z cílů této práce je seznámit se s již vypracovanými komunikačními aplikacemi, které mají implementovaného mobilního klienta. Dále je nutné tyto aplikace analyzovat a porovnat jejich přednosti a nedostatky. Následující cíl je návrh nového systému, který musí splňovat stanovené požadavky. Pro vytvoření tohoto návrhu budou využity poznatky z prvního cíle práce.

Hlavním cílem práce je vytvoření nového komunikačního systému. Tento systém musí mít implementované následující funkcionality: privátní a veřejný skupinový chat, sdílení akcí v kalendáři, sdílení videí a sdílení své geografické polohy. Aplikace by také měla podporovat určité úrovně oprávnění, například uživatel, vedoucí, správce, kde uživatel nemůže zakládat nové skupinové chaty, ale může se k nim připojit. Nemůže nahrávat videa, nebo vytvářet kalendářní akce, je mu však umožněno přehrávat videa a akcí se zúčastnit. Vedoucímu je navíc umožněno zakládat a spravovat privátní či veřejné skupinové chaty. Dále má oprávnění vytvářet kalendářní akce. Správce má k dispozici všechny funkcionality aplikace, není mu však umožněno vstupovat do privátních skupin bez povolení vlastníka skupiny. Všichni uživatelé mají možnost sdílet svou geografickou polohu.

Posledním cílem práce je otestovat vytvořený systém a navrhnout jeho další možná rozšíření. Této problematice je věnována poslední kapitola práce.

Analýza vybraných aplikací — 2

Tato kapitola se bude zabývat prozkoumáním již vytvořených aplikací. Vybraná řešení musí splňovat následující kritéria: mít implementovaného mobilního klienta a umožňovat uživatelskou nebo skupinovou komunikaci.

Jednotlivé systémy budou popsány a budou zmíněny jejich kladné a záporné vlastnosti. V závěru této části budou porovnána jednotlivá řešení s ohledem na stanovené požadavky systému pro podporu komunity.

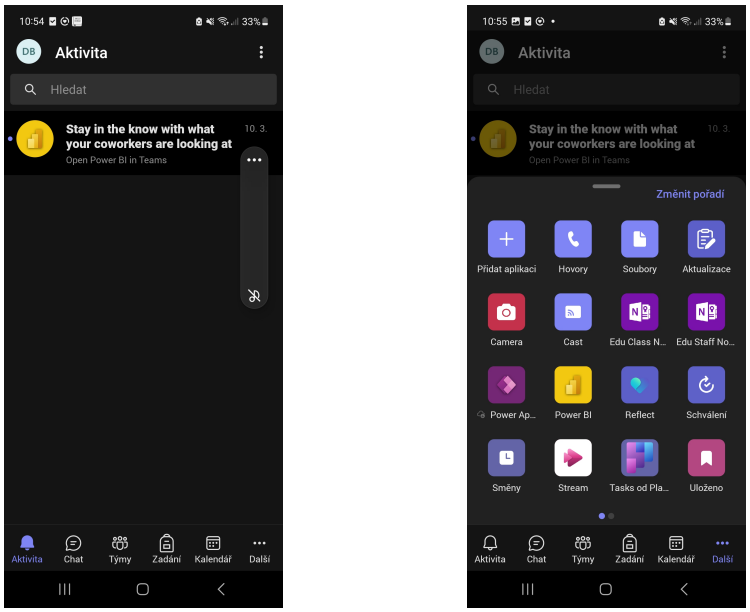
2.1 Microsoft Team (MS Teams)

MS Teams [Mica] je nástroj pro komunikaci a spolupráci vyvinutý společností Microsoft. Je to jeden z nejrozšířenějších nástrojů pro týmovou komunikaci. Jedná se "o komplexní aplikaci, která poskytuje chat, schůzky, video a hlasové hovory, spolupráci na dokumentech, vyhledávání informací, poznámky, integrace s nástroji třetích stran a další služby". [Mel18].

Tento systém je velice komplexní, poskytuje tedy velké množství služeb a funkcionalit pro své uživatele. Pokud se podíváme na mobilního klienta, zjistíme že většina požadovaných služeb je dostupná. Konkrétně služby volání mezi dvěma a více osobami, sdílení kalendáře či videí nebo zakládání chatových místností. Hlavním problémem je ale, nepřehlednost této aplikace (viz obr. 2.1). Podle mého názoru technicky méně zdatný uživatel je schopen aplikaci využívat, ale při správě uživatelů, chatů a zbylých služeb je zde velké riziko, že omylem provede akce, které nezamýšlel. Důvodem je velké množství poskytovaných služeb a následná customizace. Například postup nahraní videa bez návodu je komplikovaný (viz obr.2.2) a ne všechny formáty jsou podporovány.

Mezi další nedostatky (z pohledu využití cílovou komunitou) patří nutnost poskytnout e-mailovou adresu při registraci. Dále je Microsoft Teams licencovaný produkt. Základní funkcionality jsou zdarma, ale pro využití méně rozšířených služeb, jako je sdílení geografické polohy, je nutné zakoupit licenci [Micb].

Microsoft Teams je vyspělá komunikační platforma, která poskytuje velký rozsah služeb a je vhodná pro řízení týmů či organizací. Problémem je, že její mobilní klient je podle mého názoru pro technicky méně zdatné uživatele příliš složitý. Dále je pro správnou funkcionalitu vyžadovaná e-mailová adresa. Tato aplikace je tedy velice vyspělá, ale svou komplexností není vhodná pro využití v menších komunitách s méně technicky zdatnými členy.



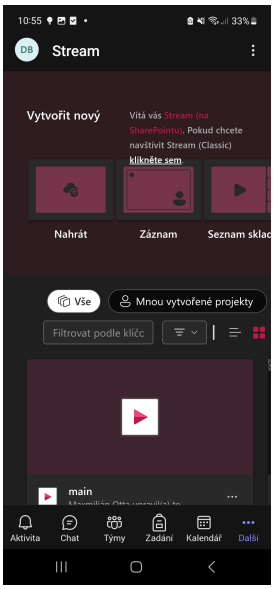
(a) Domovská stránka. (b) Možné nástroje.

Obrázek 2.1: Domovská stránka aplikace MS Teams.

2.2 Messenger

Messenger je komunikační služba, která umožňuje lidem posílat zprávy na Facebooku a Instagramu. Jedná se o samostatnou aplikaci pro lidi, kteří chtějí mít rychlý přístup ke konverzacím se svými přáteli, rodinou nebo komunitami [Faca]. Jedná se o velmi rozšířenou aplikaci, která umožňuje posílání zpráv mezi uživateli, vytváření skupin, posílání videí, zahajování hovorů a mnoho dalších služeb.

I když je k dispozici desktopová aplikace, je tento systém primárně koncipovaný jako mobilní klient. Jeho hlavní zaměření je komunikace buď v textové nebo hlasové podobě. Z těchto důvodů lze snadno zakládat chatovací skupiny nebo zahajovat hlasové hovory mezi dvěma či více uživateli. Je dokonce možné vytvářet komunity. Ty jsou však vázané na facebookové skupiny, je tedy nutné mít pro využití této funkcionality založen facebookový účet. Pro základní funkci však messenger aktivní facebookový účet nevyžaduje, je však nutné mít alespoň deaktivovaný facebookový účet[Facb]. Mezi další méně uživatelsky rozšířené funkcionality patří



Obrázek 2.2: Stránka pro upload videa v aplikaci MS Teams.

posílání financí, možnost komunikovat se známými firmami chatovou formou, sdílení geografické polohy, nebo využití facebookového virtuálního asistenta.

Podle zveřejněných statistik je messenger globálně třetí nejvyužívanější chatovací aplikací[Sta]. Je tedy zřejmé, že se jedná o vyspělou aplikaci, která je vhodná pro využití buď jedinci nebo organizacemi. Tato aplikaci je také poskytnuta uživatelům zdarma. Dále jsou její základní funkce (chatování, volání) intuitivně přístupné a ovladatelné (viz obr.2.3). Dalo by se tvrdit, že mezi slabiny této aplikace patří absence pokročilé správy uživatelů. Všichni uživatelé jsou si rovni a mohou plně využívat všechny funkcionality aplikace. Tento přístup má své přednosti, mohl by být ale nevhodný pro využití organizací s technicky neznalými uživateli. Tito uživatelé se mohou volně pohybovat v celé aplikaci, což jim umožňuje omylem dělat akce, které negativně ovlivní je nebo celou skupinu. Například nedopatřením sdílet osobní videa, zvát neznáme uživatele do skupiny nebo sdílet osobní údaje s neznámým uživatelem.

Dalším problémem je otevřenost aplikace pro všechny uživatele. Každý uživatel může zasílat zprávy jiným uživatelům a to včetně videí, URL odkazů a hlasových zpráv, což vede k případnému phishingu či výskytu uživatelů s falešnou totožností. Toto může být potencionálně nebezpečné pro technicky méně zdatné uživatele, kteří jsou k těmto rizikům náchylnější.

Obrázek 2.3: Domovská stránka aplikace Messenger [Facc].

2.3 WhatsApp

Služba WhatsApp začala jako alternativa k SMS zprávám. Tento produkt v současnosti podporuje odesílání a příjem různých médií: ať už jde o text, fotografie, videa, dokumenty a polohu nebo o hlasové hovory. Pro ochranu soukromí je do aplikace zabudováno koncové šifrování [wha].

Mobilní klient obsahuje mnoho funkcionalit. Ne všechny tyto funkcionality jsou však poskytnuty zdarma. Mezi přístupné funkce patří: posílání zpráv, zakládání skupinových konverzací, volání a sdílení videí či GIFů. Dále je možné doinstalovat různé balíčky, které poskytují další užitečné služby.

Chatovací služba je lehce a intuitivně přístupná (viz obr. 2.4), identifikace uživatele probíhá pomocí mobilního čísla. Veškeré uložené kontakty a media jsou zálohované, pomocí cloudových služeb (vyžaduje povolení od uživatele). Všechny chaty jsou v mnoha ohledech nastavitelné. Mezi tato nastavení patří: zálohování, šifrování, oznámení či nastavení pozadí chatu. Chat dále umožňuje správu jeho uživatelů. Je také možné sdružovat jednotlivé skupinové chaty do komunit. V těchto komunitách je možné zasílat zprávy či oznámení všem členům komunity. Všechny chaty je možné zamknout pomocí otisku prstu, nebo Face ID. Je tedy velice složité, aby se k nim dostala nepovolaná osoba.

WhatsApp dále poskytuje možnost volání za použití aplikace. Lze volat jednomu nebo více uživatelům, nebo zahájit skupinový hovor. Hovory jsou šifrované a poskytnuté zdarma , nezávisle na lokaci. Tyto konverzace je možné provozovat v audio či video podobě.

Aplikace WhatsApp je podle statistik globálně nejpoužívanější chatovací aplikací[Sta]. Z toho důvodu je možné tvrdit, že uživatelské rozhraní základních funkcionalit je velice přívětivé. Možnosti chatování, volání a zakládání skupin jsou velice propracované a mají mnoho zajímavých osobních nastavení. Podobně jako v mnoha

jiných aplikacích je možné v chatových skupinách sdílet videa či jiná média. Možnost volání mezi uživateli nebo ve skupině je také zajímavá a užitečná.

Možné slabiny této aplikace mohou být: omezená správa uživatelů, nemožnost zakládat kalendářní události, pokročilé nastavení může být problémové pro méně technicky zdatné uživatele. Jedná se o velice sofistikovaný systém, který obsahuje mnoho podpůrných funkcí. Jeho největší slabinou je z mého pohledu jeho komplexnost. Pro technicky zdatné uživatele se jedná o dobrou vlastnost. Méně informačně gramotní uživatelé se však mohou v aplikaci ztratit a může oslovit neznámý uživatel, který se může vydávat za někoho známého.

Obrázek 2.4: Chat aplikace WhatsApp [Yak].

2.4 Slack

Slack je komunikační aplikace, která je vytvořena primárně pro firmy. Jedná se o systém, vytvořený pro sdružování lidí, aby pracovali jako jeden tým[Slaa]. Hlavní důraz této aplikace je položen na komunikaci v týmu a správu jednotlivých pracovních projektů.

Jelikož se jedná o systém zaměřený na řízení firemních projektů, poskytuje mnoho zajímavých funkcionalit. V mobilním klientu je možné využívat chat, sdílení videí, provozování video hovorů, správu uživatelů, vytváření událostí a sdílení geografické polohy. Je možné zakládat nová vlákna, ve kterých mohou uživatelé sdílet

své názory na zadané téma nebo zasílat zprávy přímo určitému uživateli. V aplikaci je možné relativně přehledně využívat základní funkcionality. Nastavování pokročilých vlastností profilu či jednotlivých vláken může být pro některé lidi matoucí až složité. Pro většinu uživatelů plné využití mobilního klienta vyžaduje prostudování návodu či instruktážních videí.

Tato aplikace je primárně koncipována pro desktop využití, je tedy pochopitelné, že grafické rozhraní mobilního klientu nepatří mezi nejlepší. Posílání zpráv jednomu uživateli je relativně intuitivní, zakládání skupinových chatů (vláken) je také poměrně snadné. Využívání pokročilých vlastností je však složitější.

Pro spuštění a zaregistrování v této aplikaci je nutné využít e-mail s pracovní doménou. Pomocí pracovní domény je možné provázat jednotlivé uživatele. Tímto způsobem je tedy možné sdružit celou organizaci do jedné aplikace. Jednotlivé služby se dále liší podle zakoupené licence. V základní licenci (poskytnuté zdarma) je poskytnut jeden workspace, volání a posílání zpráv mezi dvěma uživateli v dané organizaci a možnost přidat deset rozšiřujících nástrojů. Je také možné posílat přímé zprávy uživatelům mimo organizaci. V placených licencích je dále garantovaný 99.99% uptime a jsou zpřístupněny další funkcionality. Mezi tyto funkcionality patří: vytváření a sdílení dokumentů, neomezený workspace, zálohování zpráv a mnoho dalších služeb [Slab].

Slack je aplikace vhodná pro firmy. Má vhodně implementované komunikační služby, sdílení médií a umožňuje propojení celé organizace. Hlavním problémem této platformy je nutnost vlastnit e-mailovou adresu ve firemní doméně. Dále pro technicky méně zdatné uživatele může být grafické rozhraní nepřívětivé. Pro využití ve větším rozsahu je nutné zakoupit licenci. Z těchto důvodů bych Slack nedoporučoval jako základní komunikační platformu pro uzavřenou komunitu.

2.5 Telegram

Telegram je aplikace pro zasílání zpráv se zaměřením na rychlost a bezpečnost. Je velmi rychlá, jednoduchá a bezplatná. Telegram můžete používat na všech svých zařízeních současně - vaše zprávy se plynule synchronizují mezi libovolným počtem telefonů, tabletů nebo počítačů[LLP].

Systém Telegram obsahuje desktopového a mobilního klienta. Velkou výhodou tohoto produktu je, že se jedná o opensource, který je již otestovaný mnoha uživateli. Dále klade velký důraz na zabezpečení a rychlost přenosu zpráv či hovorů. Mezi poskytované funkcionality patří posílání zpráv v soukromé nebo skupinové konverzaci, posílání videí a jiných médií, zahajování hovorů a sdílení geografické lokace. Skupinové hovory mohou být nahrávány a jejich účastníkům ztlumen mikrofon. Chaty jsou graficky velice přívětivé a nastavitelné. Posílání videí, GIFů nebo

jiných médií není složité. Je také možné přeposílat zprávy, videa a ostatní média mezi uživateli (pokud to uživatel povolil).

Telegram používá identifikaci uživatele pomocí mobilního čísla. První zaregistrované zařízení musí být s telefonním číslem. Pro připojení dalších zařízení pod stejný účet se využívá již zaregistrované zařízení, na které přijde v aplikaci Telegram ověřovací kód. Komunikace probíhající pomocí Telegramu jsou šifrované pouze pokud si to uživatel přeje. Telegram dále obsahuje mnoho relativně méně užívaných funkcionalit.

Telegram je velice elegantní řešení chatovací aplikace. Obsahuje mnoho užitečných vlastností, které lze velmi podrobně nastavit. Jedná se o spolehlivou aplikaci, která je komunitně testovaná a užívaná ve velkém měřítku. Jelikož jde o opensource projekt, tak jsou všechny licence zdarma. Problémem je, že funkcionalita vytváření událostí není poskytnuta a jelikož se jedná rozsáhlý projekt, který podporuje mnoho různých vlastností, může se stát pro technicky nezkušené uživatele nepřehledným.

Dalším problémem je otevřenost aplikace pro všechny uživatele. Každý uživatel může zasílat zprávy jiným uživatelům a to včetně videí, odkazů a hlasových zpráv. Toto může být potencionálně nebezpečné pro technicky méně zdatné uživatele.

2.6 Viber

Viber je komunikační aplikace zaměřena spíše na mladší cílovou skupinu. Poskytuje služby posílání zpráv, hlasových hovorů, video hovorů a další možnosti. Všechny individuální hovory, osobní chaty a všechny skupinové chaty jsou chráněny vestavěným koncovým šifrováním[Med].

Tento systém obsahuje desktopového a mobilního klienta. V mobilním klientu je možné posílat zprávy mezi jednotlivými uživateli nebo ve skupinách. Dále je možné vytvářet sociální komunity, zakládat video či audio hovory a sdílet svou geografickou lokaci. Celá aplikace je poskytnuta uživatelům zdarma.

Viber identifikuje uživatele pomocí mobilního čísla. Po stažení a registraci je možné najít ostatní uživatele za pomocí aplikace "kontakty"(u některých platforem je nutné specificky povolit tuto službu) nebo přímo v aplikaci Viber. Pokud má daná osoba stažený Viber na svém mobilním zařízení, objeví se u jeho kontaktu speciální ikona (u některých platforem je nutné specificky povolit tuto službu). Kliknutím na tuto ikonu lze vybrat, zda chci volat nebo poslat zprávu. Veškeré konverzace jsou šifrované a zálohovatelné. Dále je možné propojit mobilního klienta s PC nebo tabletem. Všechna tato zařízení fungují pod jedním účtem a jsou synchronizovatelná.

Tato aplikace je zaměřena na posílání zpráv. Z tohoto důvodu je grafické zpracování této části velice vyspělé. V chatu lze posílat zprávy, GIF obrázky, nálepky a videa. Vybrané zprávy lze smazat i po odeslání. Všechny chaty jsou automaticky šifrované za pomocí end-to-end šifrování. Je také možné nastavit typy posílaných

zpráv na běžné či automaticky mizící. Samotný chat má mnoho nastavení. Mezi ně patří možnost skrytého chatu, do kterého se není možné dostat bez pinu nebo nastavit "radar" takovým způsobem, že ostatní uživatelé neuvidí váš status či zda jste si zobrazili příchozí zprávy. Je možné si zablokovat nebo odblokovat jednotlivé uživatele. Pro kreativní skupinu lidí je také přítomna funkcionalita pro vytvoření obrázků, GIFů nebo nálepek. Dále je možné sledovat populární veřejné účty nebo komunity. Tímto způsobem lze získat informace o dění ve světě a kontaktovat své přátele v jedné aplikaci. Zajímavou implementovanou funkcionalitou je možnost hraní her přes tento systém. Díky tomu je možné se spojit s ostatními uživateli a informovat je o svém postupu.

Aplikace Viber je spíše určena pro mladší uživatele. Má velice dobře graficky zpracovaný chat a provázání mezi jednotlivými uživateli. Každý uživatel má možnost vytvářet své grafické prvky a následně je sdílet. Podle mého názoru mezi slabiny této aplikace patří její komplexní možnosti nastavení pro většinu implementovaných prvků. To může být velice matoucí pro méně technicky zdatné uživatele. Dalším nedostatkem je absence možnosti zakládání kalendářních událostí.

2.7 Discord

Discord je bezplatná komunikační aplikace, která je z velké části využívána pro komunitní skupinovou komunikaci. Discord poskytuje prostor pro jednotlivé komunity (nazývají je servery), kde si její členové mohou zasílat zprávy, volat si či sdílet videa a obrázky.[Inc].

Discord je primárně koncipovaný jako desktopová aplikace, má však také mobilního klienta. V tomto klientu je možné se připojit k jednotlivým serverům, posílat zprávy či zahajovat audio nebo video hovory. Tato aplikace přednostně slouží ke sdružování uživatelů do skupin či komunit. Toto sdružování funguje pomocí discord serverů. Jednotlivé servery může každý uživatel vytvářet.

Discord servery fungují jako sdružující místa pro komunity. Na každém serveru lze vytvořit jednotlivé kanály, ve kterých lze diskutovat o vybraných tématech.Dále je možné zakládat hlasové kanály, ty se převážně využívají ke komunikaci během hraní týmových počítačových her. Server samotný obsahuje velké množství nastavení. Některá z nich jsou například přístupnost, chat-bot pro časté otázky, šifrování a ověření věku. Je také možné chránit přístup k jednotlivým kanálům za pomocí hesla.

Registrace a přihlášení uživatelů probíhá za pomoci e-mailové adresy. K identifikaci v aplikaci samotné je využito speciální uživatelské jméno. Při připojení k serveru si může uživatel nastavit pod jakým jménem bude viditelný. Správce serveru má k dispozici správu všech připojených uživatelů.

Primární zaměření Discordu je na sdružování lidí do komunit. V každé komunitě pak existují vlákna pro diskuze o jednotlivých tématech. Systém správy serverů je velice vyspělý. Lze vytvářet veřejná či soukromá vlákna, spravovat uživatele, určovat pravidla nebo vytvořit chatbota, který zodpoví základní otázky. V každém vlákně je pak možné posílat zprávy, sdílet videa, obrázky, nebo jiná média.

Discord byl prvotně vytvořen se záměrem propojení hráčů počítačových her. Nyní však přerostl herní komunitu a je používán různými organizacemi. Problémem je, že se primárně jedná o desktopovou aplikaci a její mobilní klient není hlavním záměrem aplikace. Z tohoto důvodu se jeví mobilní klient jako nepřehledný, až chaotický. Posílání zpráv a případné připojení se k serveru je relativně složité, ale stále proveditelné. Jakékoli nastavení bez návodu či instruktážních videí je pro technicky neznalé uživatele velice obtížné. Dalším problémem je, že tato aplikace nepodporuje vytváření kalendářních událostí a sdílení geografické polohy.

2.8 Porovnání vybraných aplikací

Každá z analyzovaných aplikací představuje systém, s poměrně velkým uživatelským zastoupením a podporou pro mnoho užitečných funkcionalit. Detailní zaměření se může lišit, ale všechny zprostředkují komunikaci mezi uživateli nebo ve skupinách. Následně budou uvedeny vlastnosti jednotlivých aplikací, které mohou být považovány za slabé nebo silné.

MS Teams je nástroj vyvinutý společností Microsoft. Jedná se o komunikační platformu, která je primárně zaměřena na desktop a tablet. Hlavní předností této aplikace je provázání s dalšími produkty společnosti Microsoft. Chat, zakládání video hovorů, vytváření událostí a mnoho dalšího je zahrnuté pod licencí, která je poskytnuta zdarma. Mezi slabé stránky patří relativní uživatelská složitost mobilního klienta, nutnost vlastnění e-mailové adresy, není zde zdarma poskytnuta možnost sdílení geografické polohy a není možné soukromě nasadit tuto aplikaci pouze pro jednu organizaci.

Messenger je jedna z platforem určených pro komunikaci, které vlastní firma Meta (dříve Facebook). Jedná se o masivně rozšířený nástroj, který poskytuje mnoho služeb. Mezi tyto služby patří personální a skupinové chaty, video či audio hovory a sdílení geografické polohy. Využívání tohoto produktu je poskytnuto zdarma. Hlavní slabinou tohoto systému je absence pokročilé správy uživatelů, nemožnost zakládání kalendářních akcí a velké množství ikon bez slovních popisků. Déle není možné tuto aplikaci nasadit pouze pro jednu organizaci.

WhatsApp patří mezi aplikace od firmy Meta (dříve Facebook). Jedná se o alternativu k SMS zprávám nebo volání. Tato aplikace obsahuje relativně velké množství podpůrných funkcionalit, které jsou vhodné pro komunitní chaty. Pro technicky

méně zdatné uživatele může být složité využívat pokročilé nastavení jednotlivých služeb. Tento systém nepodporuje zakládání kalendářních akcí.

Slack je platforma zaměřená na komunikaci uvnitř firem. Její funkcionality jsou přizpůsobené za tímto účelem. Jedním z omezení může být nutnost registrace pomocí e-mailové adresy s firemní doménou. Dalo by se tvrdit, že z toho důvodu tato aplikace není vhodná pro malé komunity. Také grafické rozhraní mobilního klienta může být pro technicky méně zdatné uživatele matoucí.

Telegram je jediná z vybraných aplikací, která je opensource. Toto je její velká přednost, protože je možné ji nasadit samostatně pro jednu komunitu. Pro technicky méně schopné uživatele může být telegram poměrně složitý. Jedním z důvodů je velké množství poskytnutých funkcí. To může být velkou předností pro většinu uživatelů. Uživatelům s menší technickou znalostí to však může být na obtíž.

Viber je aplikace s velice dobře propracovaným chatem. Obsahuje velké množství obrázků, GIFů a jiných médií pro komunikaci. Zahrnuje nástroje pro vytvoření nových médií. Vzhledem k velikosti systému a všech nastavitelných vlastností se jedná o elegantní řešení komunikační aplikace. Pro technicky méně zdatné uživatele může být složité využít a nalézt jednotlivé funkcionality.

Discord je primárně koncipován pro herní komunitu. Je využívaný mnoha dalšími uživatelskými skupinami, není však koncipován pro technicky méně zdatné uživatele.

Vlastnosti	MS Teams	Messenger	WhatsApp	Slack	Telegram	Viber	Discord
Zdarma	Pouze základní funkce	Ano	Ano	Ne	Ano	Ano	Ano
Skupinový chat	Ano	Ano	Ano	Ano	Ano	Ano	Ano
Sdílení videí	Ano	Ano	Ano	Ano	Ano	Ano	Ano
Sdílení akcí	Ano	Ne	Ne	Ano	Ne	Ne	Ne
Správa uživatelů	Omezená	Omezená	Omezená	Ano	Omezená	Omezená	Omezená
Sdílení polohy	Ne	Ano	Ano	Ano	Ano	Ano	Ne
Uživatelsky přívětivé pro cílovou komunitu	Ne	Částečně	Částečně	Ne	Částečně	Částečně	Ne

Tabulka 2.1: Tabulka vlastností vybraných aplikací.

V uvedené tabulce (viz 2.1) jsou vidět vlastnosti jednotlivých aplikací. Z pro-

vedené analýzy lze vyvodit závěr, že žádná z prozkoumaných aplikací plně nepokrývá stanovené požadavky na aplikaci pro podporu komunity. Hlavním problémem těchto aplikací je, že jsou příliš komplexní. Tato vlastnost vede na zvýšení složitosti ovládání aplikace, která může být problémová pro technicky méně zdatné uživatele. Jednou z překážek pro technicky méně zdatné uživatele, která se objevuje ve většině aplikací je používání ikon bez slovního popisu. Někteří uživatelé neznají význam takových ikon, bylo by pro ně tedy lepší využití slovního popisu při ovládání hlavních funkcionalit. Dalším problémem je, že většinu z uvedených systémů není možné nasadit samostatně pro jednu organizaci.

Po konzultaci s komunitou vyšlo jako nejvhodnější řešení vytvoření nového uzavřeného systému, který upřednostní jednoduché ovládání před komplexností nabízených funkcí. Takový systém poskytne jednoduché a bezpečné prostředí pro podporu fungování komunity a redukuje rizika phisingu, falešné identity a dalších hrozeb tohoto typu. Instance tohoto systému by byla následně nasazena separátně pro každou komunitu.

Návrh komunikačního systému 3

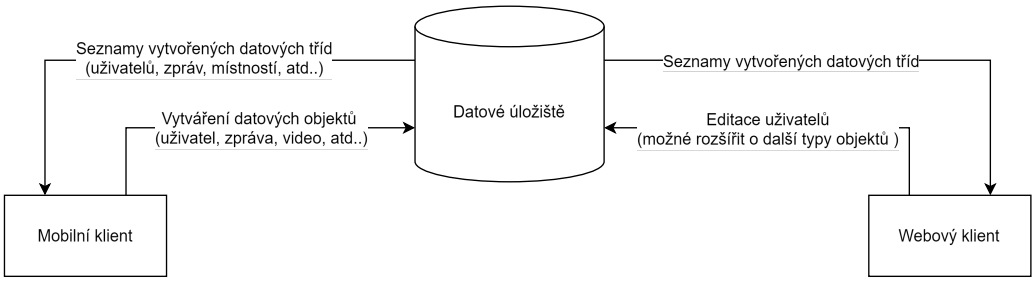
V této kapitole bude proveden návrh pro vytvoření nového komunikačního systému pro podporu komunity. Požadované vlastnosti systému jsou:

- Správa uživatelů.
- Zakládání soukromých a veřejných chatovacích místností.
- Sdílení videí.
- Vytváření akcí.
- Sdílení geografické polohy.

Všechny funkcionality by měly být snadno ovladatelné a intuitivní. Aplikace bude poskytovat čtyři různé úrovně oprávnění. Jmenovitě:

- Běžný uživatel, který může používat chat, zobrazovat videa a události, ale nemůže nic zakládat.
- Vedoucí, který muže zakládat chatovací místnosti a kalendářní události pro jím založené místnosti.
- Admin, který muže provádět správu uživatelů, sdílet videa a zakládat kalendářní akce viditelné pro všechny uživatele.
- Super admin, který může změnit hesla uživatelů a smazat jakoukoli místnost, video či viditelnou kalendářní akci (některé privátní akce pro něho nemusí být viditelné).

Podrobný popis využití služeb v závislostí na oprávnění viz kapitola 3.2. Systém se bude skládat z mobilních klientů, webové aplikace a datového úložiště. Správa uživatelů bude probíhat za použití webové aplikace a komunikace samotná bude probíhat v mobilních klientech. Z uživatelského pohledu bude komunikace mezi webovým a mobilním klientem, či mezi mobilními klienty navzájem, uskutečněna za pomocí datového úložiště (viz obr.3.1)



Obrázek 3.1: Návrh komunikace mezi jednotlivými částmi systému.

3.1 Návrh funkcionalit

Tato podkapitola se zaměřuje na návrh jednotlivých funkcionalit vytvářeného systému. Za pomocí různých typů UML (Unified Modeling Language) diagramů[Mülo4] bude znázorněn architektonický návrh jednotlivých služeb. Hlavním cílem je popsat jejich funkcionalitu a komunikaci mezi sebou a uživatelem. Dále je nutné navrhnout grafické rozhraní webového i mobilního klienta. Jednotlivé funkcionality musí být intuitivně ovladatelné i pro méně technicky zdatné uživatele. Za tímto účelem je nutné provést základní grafický návrh, ze kterého bude moci následná implementace vycházet.

3.1.1 Návrh chatovacích místností

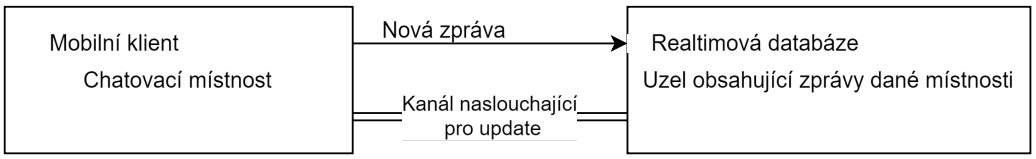
Základní částí tohoto systému je možnost zasílaní zpráv do různých skupin. Jelikož tuto službu budou využívat téměř všichni uživatelé, je nutné aby byla snadno přístupná a ovladatelná. Po konzultaci s cílovou komunitou bylo dosaženo závěru, že systém musí poskytovat dva druhy místností, konkrétně veřejné a privátní. Do veřejných místností se může přihlásit kterýkoli uživatel, avšak pro vstup do privátních místností je nutné zažádat o vstup a získat povolení od zakladatele místnosti (více o vztahu mezi typem uživatele a využití služeb viz 3.2).

3.1.1.1 Funkcionální návrh chatovací místnosti

V navrhovaném systému budou dva druhy chatovacích místností. Konkrétně veřejná a privátní, kde privátní místnost má kontrolovaný vstup uživatelů. Připojení k privátní místnosti bude probíhat tak, že uživatel pošle do místnosti zprávu se žádostí o vstup. Tuto zprávu uvidí pouze zakladatel místnosti a pouze on na ni může reagovat. Po jejím případném schválení bude moci uživatel vstoupit do místnosti. Zakladatel má dále možnost odstranit vybrané uživatele z místnosti a zakládat kalendářní akce viditelné pouze pro členy místnosti.

Pro samotné zasílání zpráv bude využita technologie Realtime Database. Jedná se o databázi zpravidla hostovanou v cloudu. Data jsou uložena ve formátu JSON a

průběžně synchronizována s každým přidruženým klientem [Khe+17]. Výhodou této technologie je, že databáze sama informuje klienty o změně, přidání či odebrání dat. Komunikace bude tedy probíhat oboustranně. Klient bude zasílat data do databáze a bude vytvořen komunikační kanál ve kterém naslouchá, zda přichází zpráva z databáze (viz obr. 3.2).



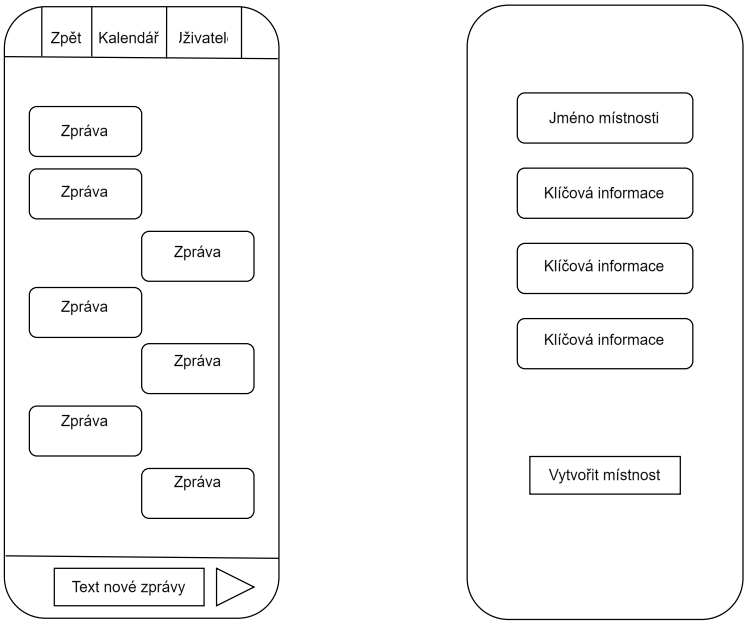
Obrázek 3.2: Návrh komunikace mezi chatovací místností a databází.

Správa místností bude probíhat také za pomoci realtimové databáze, kde uživatel s dostatečným oprávněním může do databáze přidat novou místnost. Seznam místností, získaný z databáze za pomocí komunikačního kanálu, bude k dispozici všem uživatelům, bez ohledu na oprávnění. Důvodem pro toto rozhodnutí je umožnění všem uživatelům zažádat o přístup do jakékoli místnosti.

3.1.1.2 Návrh grafického rozhraní chatovacích místností

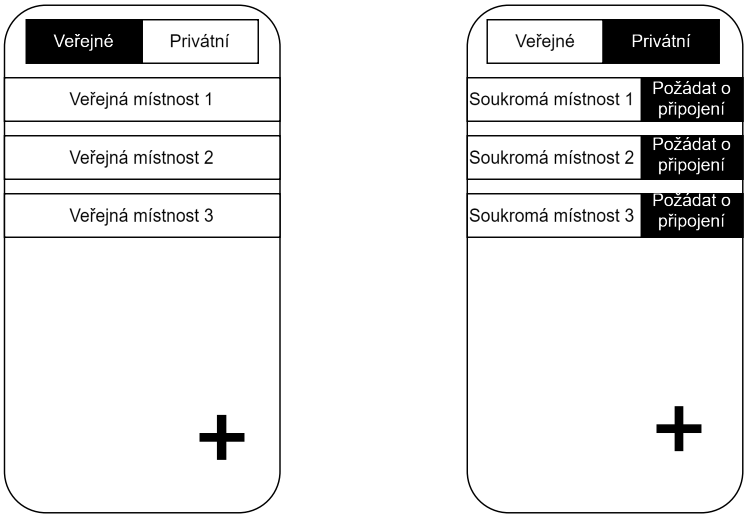
Návrh samotného chatu bude vycházet z poznatků získaných v kapitole Analýza již vytvořených aplikací (viz kapitola 2). Téměř všechny aplikace mají stejný typ grafického rozhraní (GUI) pro posílání zpráv. Tento typ byl zachován, jelikož se jedná o převážně intuitivní řešení, které jsou schopni používat i technicky méně zdatní uživatelé. Specificky se jedná o obrazovku, kde horní část je vyhrazena pro nastavení a administrativní služby (např. odhlášení, tlačítko zpět, seznam uživatelů místnosti, atd.). V prostřední části, která zabírá většinu místa, se nachází zprávy. Dolní část je vyhrazena umístění elementu pro napsání nové zprávy (viz obr. 3.3a). Založení nové chatovací místnosti bude probíhat za pomocí zobrazeného formuláře. Tento formulář bude obsahovat elementy sloužící k zadání klíčových informací o místnosti a také tlačítko s funkcí přidat novou místnost (viz obr. 3.3b).

Grafický návrh pro vybrání chatovací místnosti je z části inspirován aplikací Discord. V horní části obrazovky se bude nacházet menu, za pomocí kterého lze vyfiltrovat veřejné či privátní chatovací místnosti. Prostřední část bude vyhrazena pro seznam místností. V případě privátní místnosti bude u názvu tlačítko s funkcí žádosti o připojení. Dolní část obrazovky bude určena pro ikonu (návrh viz obr. 3.4), sloužící k přesměrování na formulář pro přidání nové místnosti. Tato ikona nebude viditelná všem typům uživatelů (detail viz kapitola 3.2).



(a) Návrh GUI pro chat. (b) Návrh GUI vytvoření místnosti.

Obrázek 3.3: Návrh grafického rozhraní pro chat a vytvoření místnosti.



(a) Návrh GUI pro veřejné místnosti. (b) Návrh GUI pro privátní místnosti.

Obrázek 3.4: Návrh grafického rozhraní pro zobrazení místností.

3.1.2 Návrh sdílení kalendářních akcí

Služba sdílení kalendářních akcí je méně rozšířená mezi analyzovanými komunikačními aplikacemi. Pořádání akcí je důležitou součástí pro správné fungování komunity, a z tohoto důvodu je nutné tuto službu poskytovat.

3.1.2.1 Funkcionální návrh sdílení kalendářních akcí

Sdílet kalendářní akce může uživatel s vyšším oprávněním (viz kapitola 3.2). Tento uživatel vytvoří kalendářní akci pro všechny uživatele, nebo pro členy jedné privátní místnosti. Vytvořená akce se objeví v přehledu akcí, a jednotliví uživatelé si ji mohou přidat do svého libovolného nainstalovaného kalendáře.

Vytvoření události bude probíhat v mobilní aplikaci, ze které se informace o vytvořené události pošlou do realtimové databáze. Seznam akcí se z databáze získá pomocí komunikačního kanálu. Tento seznam se zobrazí v mobilní aplikaci, a poté si uživatel zvolí kterých akcí se chce zúčastnit. Zakladatel události ji také může následně smazat.

Uvedené řešení bylo zvoleno, protože ne všichni členové komunity se chtějí zúčastnit všech pořádaných akcí. Tímto způsobem si každý uživatel může zvolit, kterých událostí se zúčastní.

3.1.2.2 Grafický návrh sdílení kalendářních akcí

Sdílení kalendářních akcí bude mít tři části. V první části bude možné založit událost. V druhé části si uživatel bude moci zobrazit založené události a vybrat si ty, kterých se chce zúčastnit. Poslední část bude zobrazovat detail vybrané události a bude obsahovat funkci pro přidání události do kalendáře.

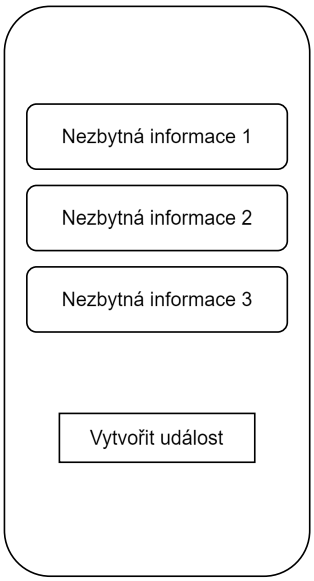
Pro vytvoření události bude sloužit intuitivní formulář, který bude obsahovat elementy pro potřebné informace a tlačítko pro založení akce (viz obr. 3.5a). Podobný formulář bude využit také pro službu přidání akce do kalendáře. Ten však bude předvyplněný a bude obsahovat tlačítko s funkcí přidání akce do kalendáře (viz obr. 3.5b).

Zobrazení všech dostupných událostí bude mít stejné rozvržení jako zobrazení chatovacích místností (viz kapitola 3.1.1.2). Rozdílem bude sloučení horní a prostřední části. Získá se tak více místa pro vytvořené události. Ve spodní části bude pro uživatele s vyšším oprávněním viditelná ikona pro přidání nové události (viz obr. 3.5c).

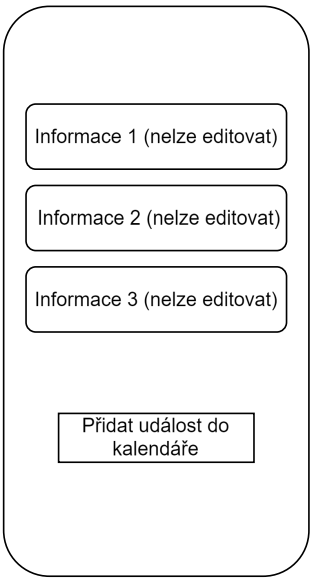
Důvodem pro zachování rozložení prvků je uživatelská přívětivost. Tímto způsobem se uživatel může navigovat ve více službách jedním typem grafického rozhraní.

3.1.3 Návrh sdílení videí

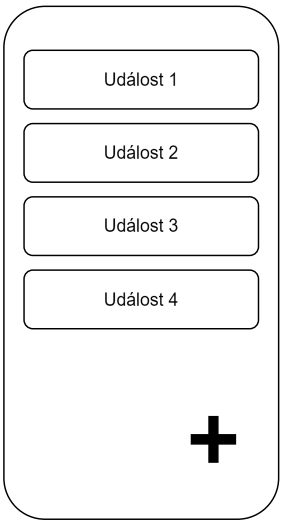
Možnost sdílení videí je důležitou součástí komunikačního systému, kterou podporuje většina komunikačních platforem. Na rozdíl od většiny ostatních aplikací, kde lze videa posílat ve skupinách, v tomto řešení je zvolen jiný přístup, který byl zvolen po konzultaci s představiteli komunity. Videa bude možné sdílet z platformy



(a) Návrh GUI vytvoření události.



(b) Návrh GUI přidání události do kalendáře.



(c) Návrh GUI pro zobrazení událostí.

Obrázek 3.5: Návrh grafického rozhraní pro sdílení událostí.

YouTube přímo do aplikace, kde je pro jejich zobrazení vyhrazen prostor. Sdílet videa nebude umožněno všem typům uživatelů, ale každý uživatel ho bude moci přehrát. Důvodem pro tento přístup je zajištění validity obsahu.

Obsahová validita je definována jako míra, do jaké jsou vybrané prvky relevantní a reprezentativní pro cílový produkt. [Yus19]. Jinými slovy se jedná o metodu, která měří, zda jsou dané informace vhodné k připojení do celkového obsahu. V tomto konkrétním případě se jedná o to, zda je sdílené video vhodné pro všechny uživatele a jakým způsobem reprezentuje či rozvíjí komunitu.

Toto řešení bylo zvoleno po konzultaci s cílovou komunitou. Důvodem je va-

lidace obsahu videí, ochrana uživatelů a úsporné využití datového úložiště. Videa na platformě YouTube jsou již validovaná a relativně bezpečná. Dále tento přístup chrání technicky méně schopné uživatele od toho, aby omylem sdíleli svá osobní videa. Sdílení videí je umožněno pouze uživatelům s vyšším oprávněním. Důvodem je to, že tato služba je zamýšlena pro sdílení organizačních videí, upoutávek či oficiálních záznamů z uspořádaných akcí. Sdílení osobních či tématicky nevhodných videí není součástí navrhované funkcionality. Hlavním důvodem je ochrana uživatelů, kteří si mohou být jisti, že informace ve sdílených videích jsou do určité míry ověřené a relevantní.

3.1.3.1 Funkcionální návrh sdílení videí

Sdílet videa může uživatel s vyšším oprávněním (viz kapitola 3.2). Video bude možné sdílet za pomocí realtimové databáze a platformy YouTube. Uživatel, který chce video sdílet, ve svém mobilním klientu vyplní URL adresu videa a pošle YouTube klíč do realtimové databáze.

YouTube klíč, nebo také YouTube video ID je unikátní ID, které slouží k identifikaci videa nahraného na serveru YouTube. Toto ID se následně využívá k vytvoření URL adresy pro zobrazení videa.

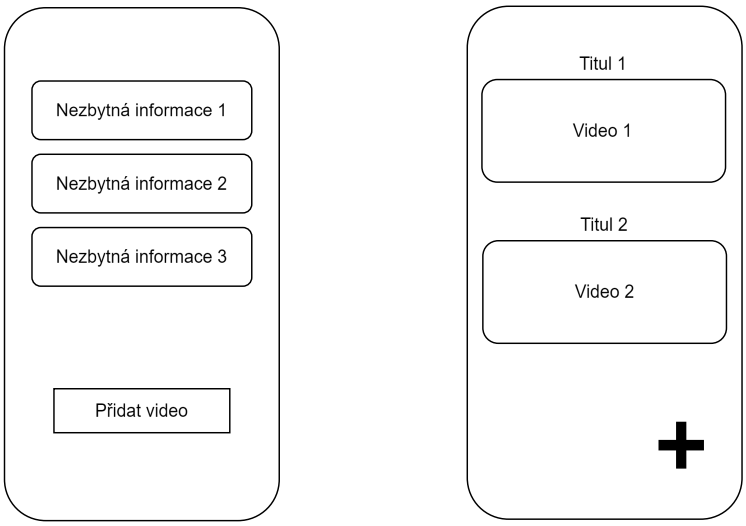
Realtimová databáze informuje mobilního klienta o nově přidaném videu prostřednictvím vzniklého komunikačního kanálu. Videa budou následně zobrazena a přehrávána v mobilní aplikaci z platformy YouTube za pomocí widgetů.

Widgety jsou lehké miniaplikace pro konkrétní úkoly, které využívají místní anebo webový obsah. Widgety se spouštějí v rámci běhového prostředí známého jako widget engine. Spuštění widgetu tedy nevyžaduje spuštění webového prohlížeče. [Lee09]. V tomto případě se jedná o okno, ve kterém je možné spustit video z platformy YouTube, aniž by bylo nutné opustit aplikaci.

3.1.3.2 Grafický návrh sdílení videí

Sdílení videa lze rozdělit na dvě úlohy. První je sdílení vybraného videa, které bude přístupné za pomocí jednoduchého formuláře. Tento formulář bude obsahovat editovatelné elementy, do kterých uživatel vloží nutné údaje o videu a tlačítko s funkcí sdílení videa (viz obr. 3.6a).

Druhou úlohou je zobrazení sdílených videí. Videa budou zobrazena pod sebou v jednotlivých widgetech. Mezi videi bude mezera a každé z nich bude obsahovat titul. Ve spodní části obrazovky bude umístěna ikona pro nahrání nového videa (viz obr. 3.6b). Tato ikona však bude viditelná pouze uživatelům s potřebným oprávněním (detail viz kapitola 3.2).



(a) Návrh GUI pro přidání videa. (b) Návrh GUI pro zobrazení videa.

Obrázek 3.6: Návrh grafického rozhraní pro sdílení videí.

3.1.4 Návrh sdílení geografické polohy

Sdílení geografické polohy je služba, která je využívána napříč mnoha komunitami. Hlavním přínosem této služby je možnost lokalizovat členy komunity v neznámém prostředí. Problémem je, že se jedná o kontroverzní funkcionalitu. Někteří uživatelé mají výhrady s poskytováním své polohy z osobních důvodů. Pokud se jedná o sdílení své polohy s jednou konkrétní osobou je toto riziko přijatelnější. Pokud se však jedná o sdílení mezi více uživateli, nastávají obavy o ochranu osobních údajů.

Jedním z možných řešení tohoto problému, které je vhodné také pro informačně méně zdatné uživatele je následující. Možnost sdílení své polohy bude intuitivní a bude dodržovat tato pravidla: uživatel je viditelný na společné mapě, pouze když se na ni sám dívá (geolokace nepoběží při minimalizaci aplikace) a v databázi nebude uložen žádný historický záznam o pozici uživatele. Tímto způsobem lze relativně bezpečně zpřístupnit tuto službu i technicky méně zdatným uživatelům.

3.1.4.1 Funkcionální návrh sdílení geografické polohy

Sdílení gegorafické polohy bude probíhat za pomocí geolokace samotného mobilního zařízení, realtime databáze a online map. Mobilní aplikace si při spuštění služby vyžádá od uživatele povolení pro získání jeho geografické polohy. Tuto polohu pošle realtimové databázi, se kterou otevře komunikační kanál. Dále v daném časovém intervalu bude posílat do databáze update své geografické pozice. K zobrazení geografické polohy budou využity online mapy. Aplikace zaznamená polohu všech uživatelů, které lze najít v realtime databázi. Pokud dojde v databázi k aktualizaci polohy některého z uživatelů, pošle databáze tato data za pomocí komunikačního

kanálu do mobilního klienta. Klient přijatá data zaznamená do vykreslené online mapy. Pokud uživatel tuto službu ukončí či minimalizuje, smaže se tím záznam o jeho pozici v realtimové databázi.

3.1.4.2 Grafický návrh sdílení geografické polohy

Grafická část sdílení geografické polohy bude odpovídat zvoleným online mapám. Uživatel bude moci takovou službu pomocí tlačítka spustit a následně vypnout.

3.1.5 Návrh služby změny hesla

Služba změny hesla je zásadní pro provoz mobilní aplikace. Pokud uživatel zapomene heslo ke svému účtu, pak musí existovat způsob, jak se k účtu přihlásit. V implementovaném systému je tento problém řešen následovně. Uživatel, který zapomněl své heslo kontaktuje jiného uživatele, který má oprávnění admin, nebo super admin. Uživatel s oprávněním admin nebo super admin, se připojí do webové aplikace, kde změní zapomenuté heslo na zvolenou hodnotu. Následně kontaktuje uživatele, který své heslo zapomněl a sdělí mu nové heslo, pod kterým se může přihlásit. Uživatel se přihlásí a změní si své heslo v mobilní aplikaci na nové.

3.1.5.1 Funkcionální návrh změny hesla

Pro změnu hesla bude zapotřebí vložit stávající heslo a nové změněné heslo. Stávající heslo se ověří a v případě, že ověření selže, zobrazí se uživateli chyba. Pokud je ověření úspěšné, pak je heslo změněno a uživateli se zobrazí potvrzení, že bylo heslo úspěšně změněno.

3.1.5.2 Grafický návrh změny hesla

Grafické rozhraní pro změnu hesla musí obsahovat formulář pro zadání stávajícího hesla, nového hesla a kopie nového hesla. Dále zde bude tlačítko pro potvrzení, že chci heslo změnit.

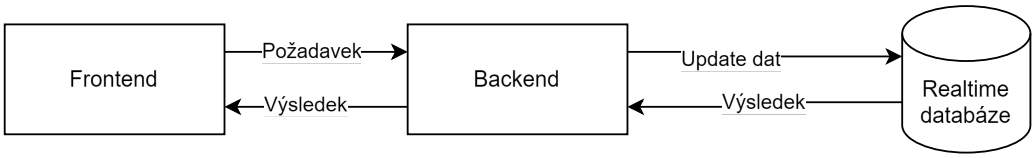
3.1.6 Návrh správy uživatelů

Správa uživatelů je jedna z klíčových služeb tohoto systému. Pro zachování snadné a intuitivní ovladatelnosti byl zvolen přístup využití webové aplikace. V této aplikaci, která je oddělena od mobilního klienta, bude docházet k editaci práv uživatelů. Dále zde bude možnost uživateli zablokovat přístup do aplikace, či jej následně znovu povolit. Poslední poskytovanou službou je změněna hesla vybranému uživateli.

Z provozních důvodů systému budou mít do webové aplikace přístup pouze uživatelé s vyšším oprávněním. Co se ale týká správy uživatelů v jednotlivých místnostech, ta se bude odehrávat přímo v mobilním klientu. Nebude tedy nutné, aby uživatel, který spravuje pouze jednu chatovací místnost, měl do webové aplikace přístup.

3.1.6.1 Funkcionální návrh správy uživatelů

Správa uživatelů bude probíhat ve webové aplikaci. Využit bude frontend aplikace, backend aplikace a realtimová databáze. Komunikace mezi frontendem a databází bude výhradně skrze backend (viz obr. 3.7).



Obrázek 3.7: Návrh komunikace mezi webovou aplikací a databází.

Samotná webová aplikace bude poskytovat tři služby. Připojení se do webového klienta aplikace pomocí uživatelského jména a hesla, zobrazení domovské stránky a zobrazení služby pro správu uživatelů. Editace uživatelů se bude z uživatelského pohledu odehrávat na frontendu. Z frontendu poté přijde požadavek na backend. Tento požadavek může být dvojího typu. Prvním je získání dat o všech uživatelích a druhým je změna dat vybraného uživatele. Backend tyto požadavky zpracuje a vloží data do databáze, nebo je z databáze získá a vrátí frontendu odpověď.

Domovská stránka webové aplikace je zahrnuta v návrhu z důvodu eventuálního rozšíření systému. V základním návrhu webové aplikace se počítá pouze se správou uživatelů. V následujícím rozšíření by se však mohly přidat nové služby (více viz kapitola 6.2 Testování aplikace a možná rozšíření).

3.1.6.2 Grafický návrh správy uživatelů

Správu uživatelů ve webové aplikaci lze rozdělit na tři části. První je přihlášení, které z grafického pohledu bude reprezentováno jednoduchým formulářem.

Druhá část je domovská obrazovka, která slouží k navigaci mezi poskytovanými službami. V tomto návrhu však webová aplikace poskytuje jen jednu službu, z tohoto důvodu je navržena jako stránka s jedním tlačítkem, které odkazuje na službu správy uživatelů.

Třetí částí je samotná služba správy uživatelů. Bude graficky reprezentována tabulkou uživatelů, ve které budou informace o uživateli a tlačítka umožňující editovat jeho údaje. Konkrétně se jedná o zvýšení či snížení oprávnění a zablokování

či odblokování uživatele v aplikaci. Tento návrh byl zvolen pro jeho jednoduchost a intuitivnost.

3.1.7 Návrh autentizační části mobilního klienta

Služba registrace do aplikace bude přístupná pouze skrze mobilního klienta. Podle požadavků na vytvořený systém nebude identita uživatele vázaná na jeho e-mailovou adresu. Byl tedy zvolen přístup unikátního uživatelského jména. Do aplikace se tedy bude přihlašovat za pomocí uživatelského jména a hesla. Heslo bude na straně mobilního klienta šifrované za pomocí technologie hashování hesel.

Hashování hesel je běžný přístup k udržování informací o heslech uživatelů, které se později používají k ověřování. Pro každé heslo se vypočítá hash, který je udržován na straně poskytovatele služby. Když se uživatel přihlásí ke službě, vypočítá se hash daného hesla a porovná se s uloženým hashem. Pokud se oba hashe shodují, je ověření úspěšné [HPM15].

3.1.7.1 Funkcionální návrh autentizační části mobilního klienta

Pro registraci nového uživatele bude nutné vyplnit nezbytná data (uživatelské jméno, heslo, atd..). Při potvrzení zadaných údajů dojde k jejich validaci. Následně se provede kontrola, zda uživatel se stejnými údaji již není v databázi zaregistrován. Pokud je kontrola provedena úspěšně, tak se provede vložení nového záznamu do databáze, který obsahuje vyplněná data. Heslo je uloženo v podobě hashe.

Služba připojení do aplikace bude využita za pomocí uživatelského jména a hesla. Z hesla se na klientské straně vytvoří hash. Následně se z databáze získají data o uživateli se zadaným uživatelským jménem. Součástí získaných dat je také hash hesla, uloženého v databázi. Tyto dva hashe se porovnají a pokud jsou shodné, tak je přihlášení povoleno. Při správném přihlášení jsou data o uživateli uložena v mobilním telefonu. Při novém spuštění aplikace se automaticky provede pokus o připojení s uloženými daty. Opakované vyplňování údajů tak není nutné.

3.1.7.2 Grafický návrh autentizační části mobilního klienta

Autentizační služba mobilního klienta bude mít dvě části. Jmenovitě registrace a připojení. Obě tyto služby budou využívat stejný typ formuláře pro zadávání dat. Rozdíl bude v počtu položek a funkci, která bude navázána na tlačítko.

3.2 Návrh užití funkcionalit pro jednotlivé typy uživatelů

V této podkapitole popíši návrh funkcionalit z pohledu úrovně oprávnění. Za pomocí UML diagramů případů užití popíši, které funkcionality může uživatel používat v závislosti na přiděleném oprávnění. UML je "univerzální jazyk pro vizuální modelování systémů", který se používá převážně v softwarovém inženýrství [Mülo4].

3.2.1 Super admin

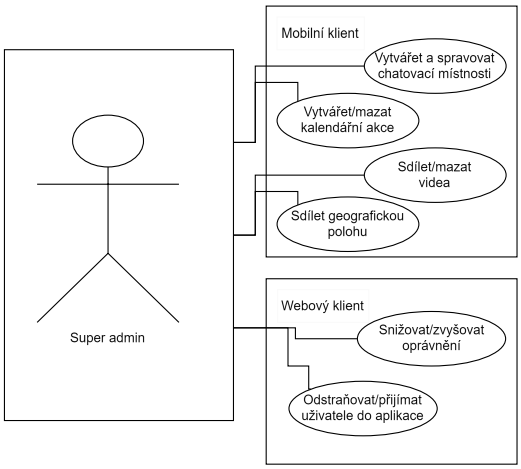
Super admin bude mít k dispozici všechny funkcionality aplikace v plném rozsahu (viz obr. 3.8). Jmenovitě může:

- Zakládat a spravovat veřejné/soukromé chatovací místnosti.
- Vytvářet/mazat kalendářní akce viditelné pro všechny uživatele.
- Vytvářet/mazat kalendářní akce viditelné pro jednotlivé privátní skupiny.
- Sdílet/mazat videa.
- Sdílet geografickou polohu.
- Zvyšovat oprávnění uživatelům až do úrovně super admin.
- Snižovat oprávnění uživatelům, kteří mají oprávnění admin nebo méně.
- Odstranit nebo přijmout zpět do aplikace uživatele, který má oprávnění admin nebo méně.
- Mazat všechna videa.
- Mazat kalendářní akce viditelné všem uživatelům.

3.2.2 Admin

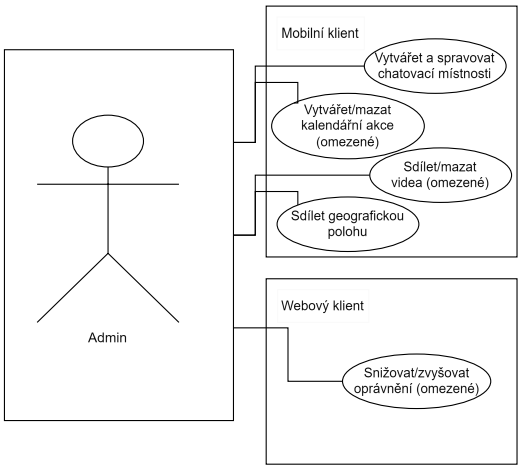
Uživatel s oprávněním admin může využívat většinu funkcionalit (viz obr. 3.9). Jmenovitě:

- Zakládání veřejných/soukromých chatovacích místností.
- Správa veřejných/soukromých chatovacích místností které sám založil.
- Vytváření kalendářních akcí viditelných pro všechny uživatele.
- Vytváření kalendářních akcí viditelných pro jednotlivé privátní skupiny.



Obrázek 3.8: UML diagram případů užití pro uživatele typu super admin.

- Mazání kalendářních akcí, které sám založil.
- Sdílení videí.
- Mazání videí, které sám sdílel.
- Sdílení geografické polohy.
- Zvyšovat oprávnění do úrovně vedoucí.
- Snižovat oprávnění uživatelů typu vedoucí a méně.

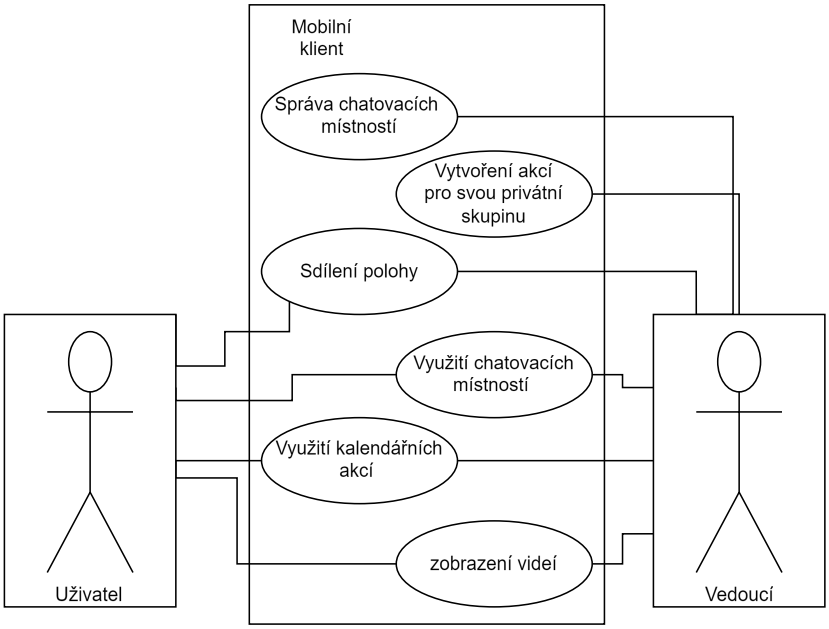


Obrázek 3.9: UML diagram případů užití pro uživatele typu admin.

3.2.3 Vedoucí a uživatel

Vedoucí má k dispozici možnost zakládat a spravovat chatovací skupiny, vytvářet a mazat kalendářní akce pro privátní skupiny, které sám založil a sdílet svou geografickou polohu.

Uživatel typu uživatel může pouze využívat funkcionality aplikace. Nemá žádný přístup ke správě uživatelů, událostí či videí. Uživatel má k dispozici následující funkcionality: vstup a chatování v místnostech, zobrazení aktivit a přidání si jich do kalendáře, zobrazení videí a sdílení geografické polohy.



Obrázek 3.10: UML diagram případů užití pro uživatele typu uživatel a vedoucí.

Výběr použitých technologií

4

V této podkapitole bude uveden seznam technologií, které jsou vhodné pro využití při implementaci navrhované aplikace. Tento systém se skládá ze tří částí. Jmenovitě mobilní klient, realtimová databáze a webová aplikace. Pro implementaci každé z těchto částí jsou k dispozici různé technologie. Jediné omezení, které plyne ze specifikace systému, je funkčnost mobilního klienta na platformě Android.

4.1 Výběr technologie pro implementaci mobilního klienta

Mobilní klient musí být snadno a intuitivně ovladatelný. Podle specifikací je nutné aby mobilní klient byl provozuschopný na platformě Android. Jiné platformy nejsou ve specifikaci zmíněny. Není tedy nutné, aby aplikace na jiných platformách fungovala.

Vývoj aplikace může být nativní, tedy pouze pro platformu Adnroid nebo multiplatformní. Pro nativní vývoj na platformě Android se převážně využívají programovací jazyky Java nebo Kotlin. Pro multiplatformní vývoj se využívají frameworky React Native,Flutter nebo .NET MAUI.

4.1.1 Nativní vývoj

Nativní vývoj je vývoj aplikace pouze pro zvolenou platformu, která je v tomto případě Android. Android je platforma v současnosti vyvíjena skupinou OHA (Open Handset Alliance). Jedná se o sdružení několika společností, které vyvíjejí hardware nebo software, například Google, Intel, NVIDIA, Motorola nebo T-Mobile. [GS15].

Společnost Google vytvořila robustní programovací prostředí Android Studio[Goo24], které je navrženo pro vývoj mobilních aplikací na platformu Android. Toto programovací prostředí mimo jiné obsahuje emulátor pro různé typy zařízení, které běží na platformě Android. Dále plně podporuje programovací jazyky Java a Kotlin.

4.1.1.1 Programovací jazyk Java

Java je programovací jazyk, který je vlastněn společností Oracle. Jedná se o objektově orientovaný programovací jazyk založený na třídách, který je navržen tak, aby měl co nejméně implementačních závislostí. Jedná se o univerzální programovací jazyk, který má vývojářům aplikací umožnit napsat jednou a spustit kdekoli (WORA), což znamená, že zkompilovaný kód v jazyce Java může běžet na všech platformách, které podporují jazyk Java, bez nutnosti rekompilace [GHA05].

Programovací jazyk Java se využívá pro vývoj desktopových, webových nebo mobilních aplikací. Jedná se o robustní nativní řešení pro vývoj komunikační aplikace pro platformu Android.

4.1.1.2 Programovací jazyk Kotlin

Kotlin je programovací jazyk primárně zaměřen na vývoj mobilních aplikací pro platformu Android, ale je možné ho kompilovat i na jiných platformách. Jedná se relativně novější programovací jazyk, který je založený na Javě.

Jedná se o univerzální programovací jazyk podobný jiným programovacím jazykům, jako je Java, Python, C++, JavaScript nebo C#. Společnost Google v květnu 2019 oznámila, že Kotlin je nyní preferovaným jazykem pro vývoj aplikací pro Android. V tomto ohledu je obzvláště dobře podporován v prostředí IDE pro vývoj aplikací Android Studio (které je postaveno nad prostředím JetBrains IntelliJ IDEA IDE) [Hun21].

Mezi jeho hlavní přednosti patří [Hun21]:

- Relativní flexibilita, která pomáhá snazšímu učení jazyka.
- Jeho schopnost běžet na (téměř) jakémkoli operačním systému, ale zejména na třech velkých operačních systémech Windows, MacOS a Linux.
- Dostupnost široké škály knihoven, které lze použít k rozšíření základních funkcí jazyka. To je částečně dáno jeho kompatibilitou s jazykem Java - což mu umožňuje využívat knihovny vytvořené pro tento jazyk.
- Dostupnost verze jazyka Kotlin, která využívá běhové prostředí JavaScriptu poskytované Node.js, což nabízí další alternativu k použitelné platformě.

4.1.2 Multiplatformní vývoj

Multiplatformní vývoj je zaměřen na optimalizaci poměru nákladů a přínosů sdílením stejného kódu mezi různými verzemi vyvinutými pro různé platformy[Del+15]. Velkou výhodou tohoto přístupu je široká cílová skupina. Mezi frameworky, které tento přístup využívají patří: React Native, Flutter, .NET MAUI

4.1.2.1 React Native

React Native je framework JavaScriptu pro psaní skutečných mobilních aplikací s nativním vykreslováním pro systémy iOS a Android. Je založen na Reactu, knihovně JavaScriptu. Díky tomu, že většinu napsaného kódu lze sdílet mezi platformami, umožňuje React Native snadný současný vývoj pro Android i iOS[Eis15].

React Native, podporuje multiplatformní vývoj a je založen na funkčních a otestovaných JavaSriptových knihovnách.

4.1.2.2 Flutter

Flutter je multiplatformní framework, jehož cílem je vývoj vysoce výkonných mobilních aplikací na rozdilných platformách (př. Android, iOS a GoogleOS). Byl vydán společností Google [Wu18]. Pro psaní programu je využit programovací jazyk Dart, který je také vlastněn společností Google.

4.1.2.3 .NET MAUI

.NET Multi-platform App UI (.NET MAUI) je multiplatformní framework pro vytváření nativních mobilních a desktopových aplikací za pomocí jazyka C# a XAML[Mic24]. Jedná se o řešení poskytnuté společností Microsoft. Jelikož využívá programovací jazyk C#, jedná se o řešení které je atraktivní také pro desktopové developery.

4.1.3 Vybraná technologie pro implementaci mobilního klienta

Pro implementaci mobilního klienta lze zvolit nativní nebo multiplatformní přístup. V tomto projektu byl zvolený nativní vývoj s využitím Android Studia a programovacího jazyka Kotlin.

Kotlin je programovací jazyk, který je preferovaným jazykem pro vývoj aplikací pro Android. Jedná se o flexibilní jazyk, který je relativně snadno pochopitelný. Vývojové prostředí Android Studio má implementovanou plnou podporu tohoto jazyka. Obsahuje mnoho použitelných knihoven, které jsou vhodné pro následnou implementaci.

4.2 Výběr vhodné realtimové databáze

Realtimová databáze je jedna z hlavních částí projektu. Jedná se typ datového úložiště. Tento typ musí podporovat následující scénář: Pokud uživatel změní data ve své aplikaci, tato data budou synchronizovaná v cloudu a z něho dále všem uživatelům, kteří používají aplikaci. Je k dispozici několik možných řešení, které splňují

požadavky realtimové databáze. Například Firebase Realtime Database, MongoDB nebo RethinkDB.

4.2.1 Firebase Realtime Database

Firebase Realtime Database je cloudová NoSQL databáze, která synchronizuje data všech klientů v reálném čase. Data jsou v databázi uložena ve formátu JSON a všichni připojení klienti sdílejí jednu instanci, která automaticky rozesílá nejnovější aktualizovaná data.[Mor17].

JSON (JavaScript Object Notation) je datový formát založený na datových typech programovacího jazyka JavaScript. V posledních několika letech si JSON získal obrovskou popularitu mezi webovými vývojáři a stal se hlavním formátem pro výměnu informací na webu. JSON dnes hraje klíčovou roli i ve webových aplikacích [Pez+16].

Firebase neposkytuje pouze službu provozu realtime databáze, ale i mnoho jiných služeb. Některé z nich jsou vhodné pro využití při implementaci vyvíjeného systému, například FCM (Firebase Cloud Messaging). Tato služba je vhodná pro posílání notifikací. Balík služeb Firebase je poskytován firmou Google. Z tohoto důvodu je poměrně jednoduché propojení Firebase a jednotlivých služeb, které Google poskytuje.

4.2.2 MongoDB

MongoDB je databáze orientovaná na dokumenty. Jedná se o datové úložiště, které je často uvedeno ve srovnávacích studiích díky své flexibilitě a snadnému použití, přičemž stále nabízí prostředky pro řešení komplexních problémů[ACP+21].

MongoDB ukládá dokumenty ve formátu BSON, což je binárně kódovaná serializace JSON navržená tak, aby byla efektivní z hlediska úložného prostoru i rychlosti prohledávání. Dokument BSON obsahuje několik pojmenovaných polí a automaticky generovaný identifikátor, který jej jednoznačně identifikuje. Pole se skládá ze tří složek: názvu, datového typu a hodnoty. BSON podporuje složité datové typy JSON, jako jsou pole a vnořené dokumenty, ale také další typy, například binární, celočíselné, s plovoucí desetinnou čárkou nebo datumové. Dokumenty jsou uchovávány v tabulkách bez schémat, které se nazývají kolekce, což umožňuje, aby v jedné kolekci koexistovaly různorodé dokumenty (ačkoli kvůli efektivitě indexování se doporučuje podobnost)[ACP+21].

Databázové vlastnosti MongoDB poskytují efektivní řešení pro uložení dat. Problémem může být že se jedná o licencovaný produkt (viz [Mon]). Z tohoto důvodu se jedná o vhodné řešení pro spíše větší systém, než je ten, který je zde navrhován.

4.2.3 RethinkDB

RethinkDB je úložiště dokumentů JSON s možností dotazování srovnatelnou s MongoDB. RethinkDB je však zcela nezávislý projekt, který se zdá být v některých oblastech ambicióznější než MongoDB a nabízí pokročilé funkce, jako jsou "pull based θ-join" dotazy. Mezi zajímavější speciality RethinkDB patří "changefeeds": dotazy v reálném čase s rozhraním pro dotazování pomocí proudu událostí [Win+19].

Theta join je spojení, které propojuje tabulky na základě jiného vztahu než rovnosti mezi dvěma sloupci. Spojení theta může používat jakýkoli jiný operátor než operátor "rovnost"[SAP24].

Jedná se o opensource škálovatelnou databázi, která poskytuje službu pro realtimovou synchronizaci JSON objektů mezi jednotlivými klienty. Z tohoto důvodu se jedná o vhodné řešení pro vyvíjený systém.

4.2.4 Vybraná realtimová databáze

Všechny zmíněné databázové produkty jsou použitelné pro následnou implementaci vytvářeného systému. Každý má vlastnosti, díky kterým by mohl být určitou skupinou vývojářů upřednostněn.

Pro tento komunikační systém bylo zvoleno řešení Firebase Realtime Database z následujících důvodů: Jedná se o službu poskytnutou zdarma, komunitně je hojně využívána při implementaci komunikačních nástrojů a jedná se o produkt firmy Google. Ve vytvořené aplikaci bude využito více služeb, které firma Google poskytuje. Některé jsou součástí balíku Firebase, například FCM a jiné jsou samostatné služby, jako jsou Google maps. Z důvodu kompatibility a snadné implementace je vhodné, pokud jsou použité služby poskytované jednou institucí.

4.3 Výběr technologie pro implementaci webové části systému

Webová část je navržena jako samostatná aplikace, která bude komunikovat s realtime databází. Je tedy nutné implementovat frontend a backend aplikace. Frontend a backend jsou dva zásadní aspekty každé aplikace. Frontend je to, co uživatelé vidí a s čím interagují, například vizuální prvky, jako jsou tlačítka, zaškrtávací políčka, grafika a textové zprávy. Backend tvoří data a infrastruktura, díky nimž aplikace funguje. Ukládá a zpracovává data aplikace pro její uživatele[AWS24].

Pro implementaci Backendu je možné použít Java SpringBoot framework, Node.js nebo Python Django. Frontend je možné vyvinout za pomocí nástrojů React nebo

AngularJS. Je mnoho dalších nástrojů, které by se daly pro implementaci webové aplikace využít.

4.3.1 Java Spring Boot

Spring Boot je jeden z modulů Spring frameworku, který poskytuje funkce pro rychlý vývoj aplikací (RAD). Je postaven na populárním frameworku Java Spring Framework. Hlavní výhodou tohoto frameworku je, že můžeme aplikaci jednoduše spustit, protože vyžaduje jen minimální konfiguraci Springu, což značně usnadňuje vývoj stand-alone aplikací založených na Springu. Pro své provozní účely využívá Spring Boot vestavěné servery a moduly Spring Frameworku.[MST22].

Jedná se o velice rozsáhlý framework, který obsahuje velké množství detailních nastavení. Je vhodný pro rozsáhlé stabilní projekty, u kterých je potřeba zajistit dlouhou dobu životního cyklu.

4.3.2 Node.js

Node.js je jedním z nejznámějších prostředí, které podporují vývoj JavaScriptu na straně serveru. Komunita vytvořila velké množství knihoven pro Node.js, a nebo knihoven, které jsou s Node.js kompatibilní. Mezi nimi jsou zvláště významné nástroje jako node-mysql nebo node-couchdb, které podporují asynchronní interakci s relačními, respektive NoSQL datovými úložišti. Mnoho frameworků poskytuje plnohodnotný webový stack, například Connect a Express, které jsou svým rozsahem a komunitním využitím, srovnatelné s Rackem a Rails ve světě Ruby. Správce balíčků Node.js, npm, umožňuje instalaci knihoven a jejich závislostí. Poslední uvedenou předností je, že existuje mnoho dostupných knihoven pro JavaScript na straně klienta, které byly napsány tak, aby vyhovovaly systému modulů CommonJS, čímž fungují také v systému Node.js[TV10].

Jedná se o relativně rozšířený nástroj mezi programátory webu a mobilních aplikací. Jeho hlavní výhodou je jeho jednoduchost a velké množství otestovaných knihoven připravených k použití.

4.3.3 Python Django

Jedná se o webový framework, který k vytváření webových stránek využívá jazyk Python. Jeho cílem je rychlý vývoj dynamických webových stránek. Tento framework byl vytvořen pro urychlení vývojové fáze webu, ale obsahuje i další vlastnosti vhodné pro vývoj frontendu. Jednou z nich je využití vzoru MVC, který umožňuje mít při implementaci ucelenou architekturu. V dnešní době je využíván firmami jako Instagram, Openslack.org nebo Mozilla.org[DBR16]. Django poskytuje způsob, jak

poměrně jednoduše vytvořit frontend aplikace za použití programovacího jazyka Python.

4.3.4 React

React je framework založený na JavaScriptu. React původně vytvořili inženýři ve společnosti Facebook, aby vyřešili problémy spojené s vývojem složitých uživatelských rozhraní, kde se využívají komplexní soubory dat, které se v průběhu času mění. React ve skutečnosti vznikl pro implementaci reklam, kde se používal tradiční přístup Model-View-Controller na straně klienta. Takové aplikace se obvykle skládají z obousměrné vazby mezi daty a vykreslovací šablonou. S využitím frameworku React se změnil způsob, jakým se tyto aplikace vytvářejí. [Gac15].

Jedná se o nástroj, který poskytuje relativně snadný způsob jak vytvořit frontend webové aplikace. Má také poměrně velké zastoupení ve skupině webových developerů.

4.3.5 AngularJS

AngularJS se na seznam frameworků MVC na straně klienta zařadil teprve nedávno, přesto se mu podařilo přitáhnout velkou pozornost, především díky inovativnímu šablonovacímu systému a snadnému vývoji. Jeho šablonovací systém je totiž v mnoha ohledech jedinečný:

- Jako šablonovací jazyk používá HTML.
- Nevyžaduje explicitní obnovení DOM, protože AngularJS dokáže sledovat akce uživatele, události prohlížeče a změny modelu, aby zjistil, kdy a které šablony je třeba obnovit.
- Má rozšiřitelný subsystém komponent a je možné naučit prohlížeč interpretovat nové značky a atributy HTML.

Šablonovací subsystém je možná nejviditelnější součástí AngularJS, nejení však jeho jedinou předností. AngularJS je kompletní framework s několika nástroji a službami, které jsou obvykle potřebné při vývoji webových aplikací[KD13].

4.3.6 Vybrané technologie pro implementaci webové aplikace

Pro implementaci webové části systému byla zvolena kombinace React pro frontend a Node.js pro backend. Jedná se o kombinaci, která je často používána při implementaci webových aplikací. Node.js má oproti frameworku Java Spring Boot výhodu, že se nejedná o tak komplexní produkt a je možné se v něm snadněji orientovat.

React byl vybrán hlavně z implementačních důvodů, protože se jedná o framework, se kterým byl vývojář lépe obeznámen. Dále se jedná framework který není příliš komplexní a umožňuje rychlý vývoj webové aplikace.

Implementace systému

5

V této kapitole se budeme zabývat popisem implementace cílového produktu. Systém se skládá ze tří částí, Firebase Realtime Database, webové aplikace a mobilního klienta na platformě Android. Firebase je již implementována firmou Google a poskytuje Mobile-Backend-as-a-Service pro využívání jednotlivých služeb. Hlavní zaměření této kapitoly proto bude na popis implementace webové aplikace a mobilního klienta.

Mobile Backend as a Service (mBaaS) je služba cloud computingu, která umožňuje vývojáři propojení jejich aplikací s databázemi a funkcemi, jako je správa uživatelů, push notifikace a integrace se službami sociálních sítí. Tyto služby jsou obvykle poskytovány s podporou sad pro vývoj softwaru (SDK) a rozhraní pro programování aplikací (API).[Mah+19].

5.1 Připojení k databázi

Součástí systému je Firebase realtime database, se kterou komunikuje mobilní i webová aplikace. V této podkapitole bude popsáno jak jsou jednotlivé části systému mezi sebou propojeny.

5.1.1 Připojení mobilní aplikace k databázi

Mobilní aplikace vyžaduje přístup ke všem datům v databázi a je tedy nutné, aby byl přístup z mobilní aplikace zabezpečený. Firebase realtime database poskytuje několik možností autentikace, mezi které patří: e-mailová adresa s heslem, telefonní číslo a Google nebo Facebook účet. Pro mobilní aplikaci bylo vybráno řešení autentikace za pomocí telefonního čísla a Google účtu.

Důvodem pro zvolení telefonního čísla je, že není nutné provádět další krok mimo aplikaci samotnou. U ostatních typů autentikace je nutné založit e-mailový či jiný účet. I když založení e-mailového či jiného účtu není technicky náročný úkol, jedná se o zbytečný krok navíc.

Dále je také implementována možnost autentikace pomocí účtu Google. Někteří uživatelé nechtějí z osobních důvodů sdílet své telefonní číslo, a proto je implementován další způsob autentikace. I když přihlášení pomocí Google účtu vyžaduje jeho založení, mnoho uživatelů již má tento účet vytvořený. Z těchto důvodů byla zvolena autentikace pomocí účtu Google jako sekundární možnost.

Při prvním spuštění mobilní aplikace bude uživatel přesměrován na obrazovku sloužící pro ověření zařízení (viz obr. 5.1). Na této obrazovce vyplní uživatel své telefonní číslo a případně SMS kód, který mu byl zaslán, nebo zvolí ověření pomocí účtu Google. Po úspěšném ověření má uživatel přístup do aplikace. Uživatel může být vyzván k opětovnému ověření mobilního zařízení v případě, že byl z Firebase účtu odhlášen.

Obrázek 5.1: Ověření mobilního zařízení.

5.1.2 Připojení webové aplikace k databázi

Webová aplikace pro svou funkcionalitu potřebuje přístup pouze k datům o uživatelích. Z tohoto důvodu nemůže číst či editovat data o zprávách, místnostech, videích, akcích a geografické poloze.

Webová aplikace komunikuje s Firebase realtime databází pomocí vygenerovaného API klíče. Může číst data o uživatelích a editovat je. Při editaci uživatelských dat je ze strany databáze vynucena předem nastavená struktura.

5.2 Implementace mobilního klienta

Tato podkapitola se věnuje vytvoření mobilního klienta. Pro jeho implementaci je využit programovací jazyk Kotlin pro platformu Android (důvod pro zvolení tohoto řešení je popsán v kapitole 4.1). Samotný kód je rozdělen do jednotlivých balíčků podle jejich významu. Pro implementaci je využit architektonický vzor MVVM (model-view-viewModel).

Základní myšlenkou je, že *model* je abstrakcí doménového modelu, *views* bude obsahovat uživatelsky prezentovatelné rozhraní a *viewModel* načítá data z modelu a předává je view.

5.2.1 Model mobilního klienta

Pro správnou funkci aplikace je nutné vytvořit model, který se skládá z několika datových tříd. Tyto datové třídy jsou implementovány na straně mobilního klienta, konkrétně v balíčku *Model*. Mobilní klient je následně zapisuje do databáze, odkud je webová aplikace čte a edituje.

5.2.1.1 Datová třída Uživatel

Datová třída Uživatel slouží k přihlášení a registraci uživatele a také je využívána k jeho identifikaci. Nachází se ve třídě *Model/UserModel.kt*. Obsahuje následující atributy:

- **key** - Slouží jako klíč v realtimové databázi a unikátní identifikátor pro uživatele.
- **name** - Jméno uživatele.
- **surname** - Příjmení uživatele.
- **userName** - Uživatelské jméno. Slouží k jednoznačné identifikaci uživatele v aplikaci, musí být unikátní mezi všemi uživateli.
- **password** - Obsahuje hash hesla. Slouží k ověření uživatele při připojení do aplikace.
- **privileges** - Obsahuje úroveň oprávnění uživatele. Úroveň oprávnění specifikuje služby, které jsou uživateli zpřístupněny (detail viz kapitola 3.2).
- **FCMToken** - Obsahuje FCM token. Slouží k zasílání notifikací mobilním klientům (detail viz kapitola 4.2.1).

5.2.1.2 Datová třída Zpráva

Datová třída zpráva je využívána k reprezentaci zaslané či přijaté zprávy. Nachází se ve třídě *Model/MessageModel.kt*. Obsahuje tyto atributy:

- **message** - Textový obsah zprávy.
- **date_time** - Čas odeslání zprávy. Slouží k řazení zpráv.
- **sender_id** - Identifikátor odesílatele zprávy, který slouží identifikaci odesilatele.
- **sender_username** - Uživatelské jméno odesilatele.
- **sender_name** - Jméno odesílatele.
- **sender_surname** - Příjmení odesílatele.
- **message_type** - Typ zprávy, který slouží k odlišení různých typů zpráv. Jsou implementovány čtyři typy zpráv: Příchozí zpráva, odchozí zpráva, ohlášení místnosti a zprávy pro zakladatele místnosti.
- **isAdmin_msq** - Slouží k identifikaci zpráv určených pouze pro správce místnosti.
- **msg_id** - Jedná se o unikátní identifikátor zprávy.

5.2.1.3 Datová třída Místnost

Datová třída místnost se využívá pro reprezentaci chatovací místnosti. Tato místnost může být privátní či veřejná a obsahuje množinu členů. Dále je možné zakládat kalendářní akce specificky pouze pro uživatele v místnosti. Nachází se ve třídě *Model/RoomModel.kt*. Datový model místnost obsahuje následující atributy:

- **groupChatId** - Unikátní klíč, který identifikuje danou místnost.
- **roomName** - Jméno chatovací místnosti.
- **isPublic** - Určuje zda se jedná o privátní či veřejnou místnost.
- **admin_name** - Uživatelské jméno zakladatele skupiny.
- **admin_id** - Identifikátor zakladatele skupiny.
- **members** - Seznam členů skupiny.
- **events** - Seznam událostí ve skupině.

5.2.1.4 Datová třída Kalendářní událost

Tato datová třída slouží k reprezentaci kalendářní události. Nachází se ve třídě *Model/EventModel* a obsahuje následující atributy:

- **key** - Unikátní klíč, který identifikuje danou událost.
- **name** - Jméno události.
- **place** - Místo události.
- **organizer** - Jméno a příjmení pořadatele události.
- **roomKey** - Identifikátor místnosti pro kterou je událost viditelná. V případě, že má být událost viditelná pro všechny uživatele, je využita unikátní konstanta.
- **date_time** - Čas konání události.
- **organizerKey** - Unikátní klíč pořadatele události.

5.2.1.5 Datová třída Video

Tato datová třída se využívá k reprezentaci videa. Nachází se ve třídě *Model/VideoModel.kt* a obsahuje následující atributy:

- **key** - Unikátní klíč, který identifikuje dané video.
- **videoUrl** - Url odkaz na video.
- **title** - Název videa.
- **source** - Platforma, ze které je video streamováno. V této verzi je, v souladu s požadavky komunity na aplikaci, podporována pouze platforma YouTube, avšak v případných rozšířeních by bylo možné zahrnout i další platformy.
- **organizerKey** - Unikátní ID uživatele, který nahrál odkaz na video do aplikace.

5.2.1.6 Datová třída Geografická lokace

Tato třída reprezentuje geografickou lokaci uživatele. Nachází se ve třídě *Model/LocationModel* a obsahuje následující atributy:

- **key** - Unikátní klíč, který identifikuje danou instanci datového modelu geografické lokace.

- **user** - Uživatelské jméno uživatele, který sdílí svou geografickou lokaci.
- **latitude** - Hodnota zeměpisné šířky geografické polohy.
- **longtitude** - Hodnota zeměpisné délky geografické polohy.

5.2.2 Views mobilního klienta

Views část mobilního klienta se nachází v adresáři *res/layout/*. Zde jsou jednotlivé soubory ve formátu XML. Každý soubor reprezentuje grafické rozhraní jednotlivé služby či komponenty (viz obr. 5.2).

5.2.3 ViewModel mobilního klienta

ViewModel mobilního klienta je rozdělen do jednotlivých balíčků. Konkrétně *Views/*, *Adapter/* a *Fragment/*. V balíčku *Views/* se nacházejí podadresáře, do kterých jsou klasifikovány třídy podle svého využití. Samotné třídy v balíčku *Views/* slouží k obsluze jednotlivých elementů, které jsou v layoutu. Tyto třídy jsou rozděleny do skupin Autorizace, Skupinové chaty, Události, Video a Lokace.

V každé třídě z balíčku *Views/* se nachází konektor, který propojí tuto třídu s vybraným layoutem, jenž obsahuje grafické rozhraní pro danou službu. Dále se zde nachází připojení k elementům z tohoto rozhraní. Každý z těchto elementů je podle potřeby obsloužen. Některé elementy vyžadují využití tříd z balíčku *Adapter/* nebo *Fragment/*.

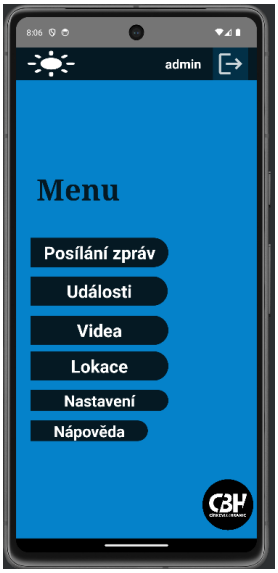
Třídy v balíčku *Adapter* jsou využity pro zobrazení seznamu interaktivních elementů, jejichž počet se mění v reálném čase. Příkladem může být seznam zpráv, seznam chatovacích místností nebo událostí. Třídy v balíčku *Fragment* jsou využity pouze pro zobrazení seznamu chatovacích místností. Zde se využívá element *TabLayout*, kterému jsou předány dva fragmenty, jmenovitě veřejné místnosti a soukromé místnosti.

5.2.4 Implementace zaregistrování nového uživatele

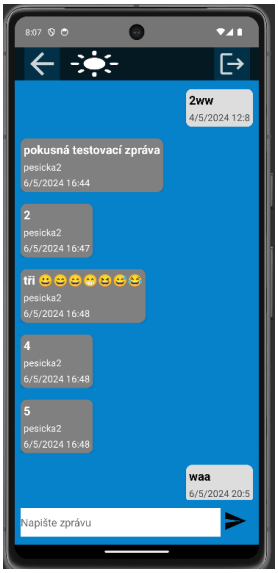
Služba zaregistrování nového uživatele je implementována v souboru *SignIn.kt*. Grafické rozhraní služby je definováno v souboru *activity_sign_in.xml*. Zaregistrování se do mobilní aplikace probíhá za pomocí vyplnění jednoduchého formuláře, který obsahuje elementy pro získání jména, příjmení, uživatelského jména a hesla uživatele. Po vyplnění formuláře a validaci zadaných dat, získá tato služba seznam všech uživatelských jmen z databáze Firebase a zjistí zda je uživatelské jméno obsazeno. Pokud je zadané uživatelské jméno volné, pak je uživatel úspěšně zaregistrován.

(a) Layout registrace.

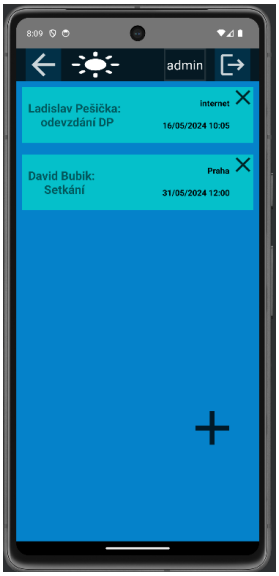
(b) Layout přihlášení.



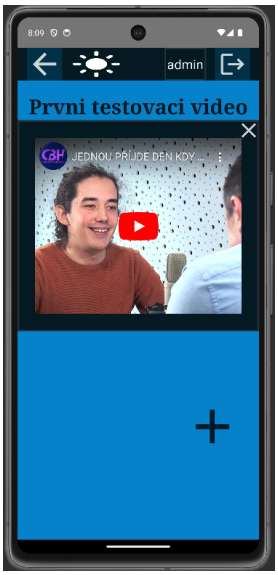
(c) Layout menu.



(d) Layout chatu.



(e) Layout událostí.



(f) Layout videí.

Obrázek 5.2: Layouty jednotlivých služeb.

5.2.5 Implementace přihlášení do mobilní aplikace

Služba přihlášení do mobilní aplikace je implementována v souboru *LogIn.kt*. Grafické rozhraní služby je definováno v souboru *activity_log_in.xml*. Přihlášení do aplikace probíhá za pomocí vyplnění jednoduchého formuláře, který obsahuje elementy pro získání uživatelského jména a hesla uživatele. Na heslo je následně aplikována hashovací funkce algoritmu Bcrypt. Z databáze Firebase služba získá data o uživateli, mezi kterými je také uživatelské jméno a heslo v zahashované podobě. Jednotlivé

hashe hesel se porovnají. V případě, že nejsou shodné, je uživateli zobrazena chybová hláška. Pokud jsou shodné, pak se data o uživateli uloží do preferencí aplikace, a při každém dalším otevření aplikace dojde k automatickému pokusu o přihlášení pomocí údajů z preferencí. Tyto údaje budou smazány, pokud se uživatel odhlásí.

Bcrypt je hashovací funkce vytvořená na základě algoritmu Blowfish dvěma bezpečnostními výzkumníky, Nielsem Provosem a Davidem Mazièresem. Tato hashovací funkce má několik výhod, používá původní náhodnou "salt" (salt je náhodně generovaný řetězec připojený k heslu, aby bylo obtížnější jej prolomit). Náhodné generování "salt" dále také zabraňují vytváření vyhledávacích tabulek. Útoky hrubou silou patří k nejčastěji používaným nástrojům pro získávání údajů o heslech.[BEN21]

5.2.6 Implementace změny hesla

Služba změny hesla je implementována v souboru *ResetPassword.kt*. Grafické rozhraní je definováno v souboru *activity_reset_password.xml*. Pro změnu hesla je nutné vyplnit formulář, který vyžaduje stávající heslo, nové heslo a kopii nového hesla. Po jeho vyplnění a stisknutí tlačítka s názvem "ZMĚNIT HESLO" dojde k validaci poskytnutých údajů. Pokud jsou všechny údaje vyplněny a stávající heslo je korektní, pak se služba připojí k databázi Firebase a pokusí se změnit heslo. V případě, že změna selže, zobrazí se uživateli chybové hlášení. Pokud je heslo změněno, pak se uživateli zobrazí hláška "Vaše heslo bylo úspěšně změněno".

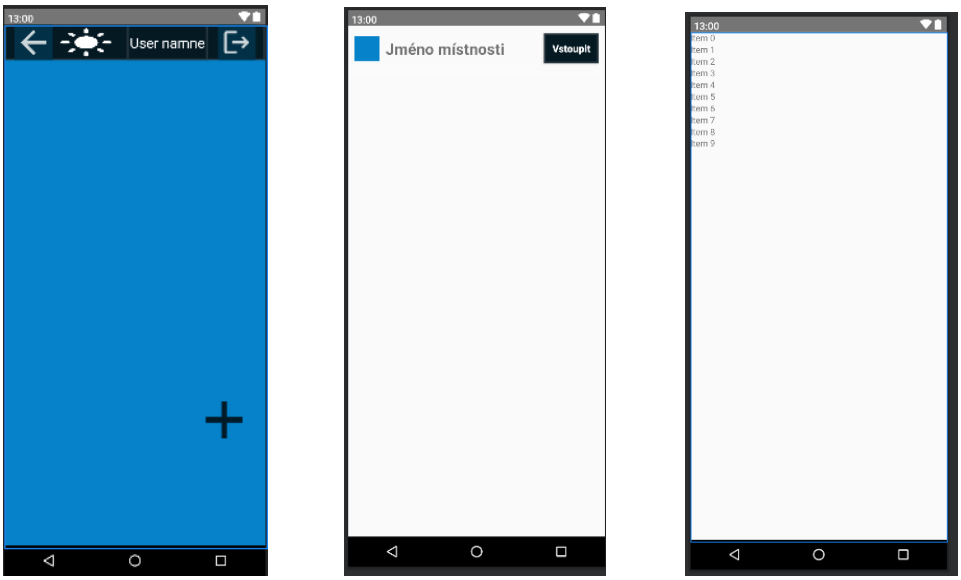
5.2.7 Implementace přehledu chatovacích místností

Služba přehled chatovací místností využívá ke své implementaci následující třídy *Views/GroupViews/ChatOverview.kt*, *Adapter/PrivateAdapter.kt*, *Adapter/PublicAdapter.kt*, *Fragments/PrivateRooms.kt* a *Fragments/PublicRooms.kt*.

Grafické rozhraní je implementováno v souborech *res/layout/activity_chat_overviue.xml*, *res/layout/fragment_private_room.xml*, *res/layout/fragment_public_room.xml* a *res/layout/rv_layout.xml* (viz obr 5.3).

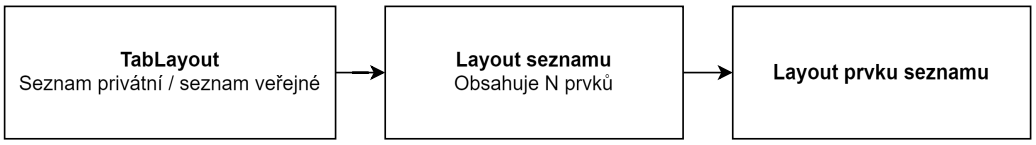
Přehled chatovacích místností je implementován pomocí tří částí. První zobrazuje layout, kde je možné za pomocí TabLayoutu vybrat zobrazení veřejných nebo privátních místností. Druhá část je zobrazení seznamu samotného. Třetí část je zobrazení jednoho prvku ze seznamu. Propojením těchto tří částí vznikne služba přehled chatovacích místností (viz obr. 5.4).

Konkrétně třída ChatOverview.kt obsahuje konektor k layoutu chat_overview. Tento layout obsahuje několik elementů. Hlavním z nich je TabLayout, ve kterém se následně zobrazí seznamy jednotlivých místností. Pro samotné zobrazení je využita třída TabLayoutMediator, které je mimo jiné parametrem předán TabLayout a seznam fragmentů k zobrazení. Instance fragmentů jsou reprezentovány třídami PrivateRooms.kt a PublicRooms.kt.



(a) Chat_overview. (b) Rv_layout. (c) Fragment_room.

Obrázek 5.3: Návrh grafického rozhraní pro zobrazení místností.



Obrázek 5.4: Přehled chatovacích místností.

Tyto třídy obsahují konektory k layoutu fragment_public_room nebo fragment_private_room. Tyto layouty obsahují element *widget.RecyclerView*. Jedná se o element připravený pro zobrazení listu s proměnným množstvím prvků. Tento list je připojený k elementu *widget.RecyclerView* pomocí prvku *widget.RecyclerView.adapter*.

Jednotlivé adaptéry jsou reprezentovány třídami *PrivateAdapter.kt* a *PublicAdapter.kt*. Tyto adaptéry jsou připojeny k layoutům, které obsahují grafické rozhraní jednoho prvku v listu. Seznam prvků je získán z Firebase databáze za pomocí komunikačního kanálu. Pokud přijde upozornění o vzniku nové místnosti, je adaptér za pomocí metody *adapter.notifyDataSetChanged* upozorněn na nutnost obnovení seznamu místností (viz obr. 5.5).

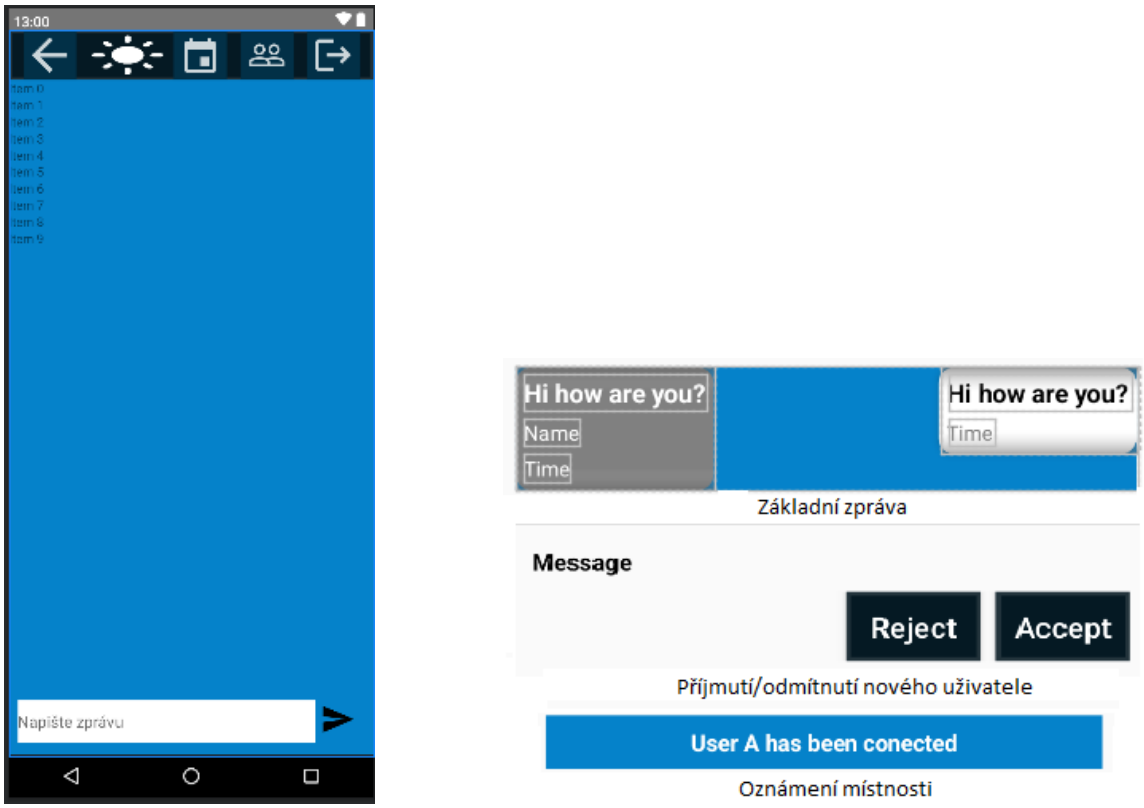
Poslední část přehledu chatovacích místností je možnost založení nové místnosti. Podle zadání bude tato služba zpřístupněna pouze uživatelům s oprávněním vedoucí, admin a super admin. Je implementována v třídě *CreateGroupChat.kt* a grafické rozhraní se nachází v souboru *activity_group_chat.xml*. Jedná se o jednoduchý formulář, po jehož vyplnění je možné založit novou místnost.

Obrázek 5.5: Přehled chatovacích místností.

5.2.8 Implementace služby chatovací místnosti

Služba chat je rozdělena na dvě hlavní části, veřejný chat a privátní chat. Pro implementaci jsou využity třídy *PrivateGroupChat.kt, PublicGroupChat.kt, PrivateChattingAdapter.kt* a *PublicChattingAdapter.kt*. Implementace grafického rozhraní se nachází v souborech *activity_private_group_chat.xml, activity_public_group_chat.xml, rv_messa-ge.xml, rv_other.xml* a *appceptreject_layout.xml* (viz obr. 5.6).

Třídy PrivateGroupChat a PublicGroupchat obsahují připojení ke grafickému rozhraní. V tomto rozhraní se mimo jiné nachází prázdný seznam (*recyclerview*), který je připravený pro vložení prvků. Dále je zde použita komponenta *recyclerview.adapter*, která ukazuje na instanci třídy PrivateChattingAdapter nebo PublicChattingAdapter. Adapter slouží ke zpracování měnícího se seznamu prvků. Tyto prvky jsou získané pomocí komunikačního kanálu z databáze Firebase. Seznam se následně pomocí metody adapter.notifyDataSetChanged předá adaptéru, který celý seznam po jednotlivých prvcích vykreslí. Třídy PrivateChattingAdapter a PublicChattingAdapter slouží k vykreslení jednotlivých prvků seznamu, v tomto případě zpráv. Adaptéry dostanou v parametru předán seznam prvků, který postupně pro-



(a) Layout privátní místnosti.

(b) Layout jednotlivých typů zpráv.

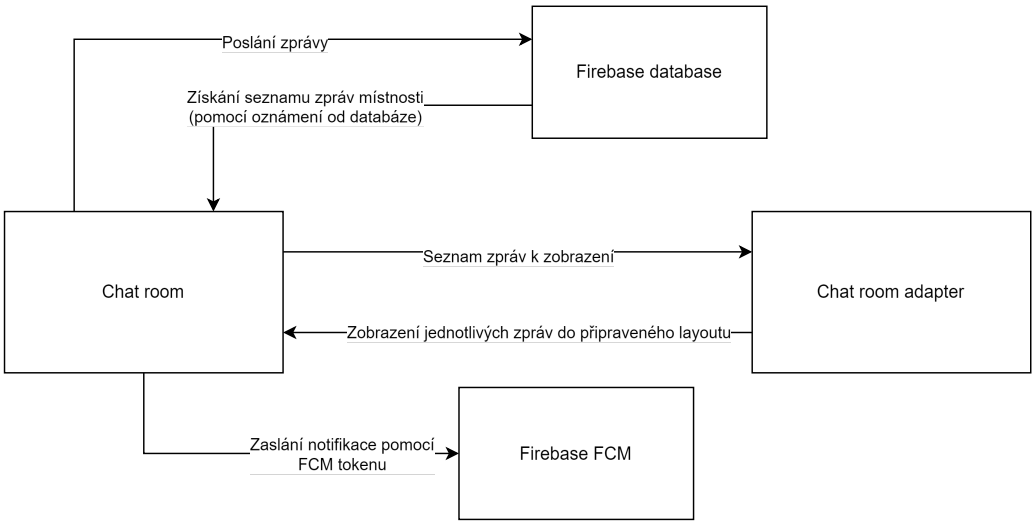
Obrázek 5.6: Návrh grafického rozhraní pro chatovací místnost.

chází a podle typu zprávy vykresluje. Existují čtyři typy zprávy. Odchozí zpráva, příchozí zpráva, informační oznámení místnosti (např. byl připojen uživatel A) a žádosti o připojení do místnosti, které jsou viditelné pouze zakladateli místnosti.

Mimo základní funkcionalitu zobrazení zpráv obsahuje služba chatu možnost zaslání zprávy, v privátních skupinách založení akce pro skupinu a odstranění člena ze skupiny. Pro zaslání zprávy je využita třída PrivateGroupchat, respektive PublicGroupChat, kde je implementována funkce obsluhující stisknutí tlačítka pro poslání zprávy. V této funkci se založí instance datové třídy zpráva, která se následně zašle do Firebase realtime database. Ta poté upozorní všechny aktivní klienty, že došlo ke změně dat. Pokud se jedná o privátní skupinu, pak je navíc zaslána notifikace za pomocí FCM tokenu všem členům skupiny. Aktivní klienti si z komunikačního kanálu přečtou jednotlivé zprávy, které předají adaptéru k vykreslení (viz obr. 5.7).

Implementace odstranění uživatele z místnosti je provedena v souborech *MemberList.kt* a *MemberAdapter.kt*. Definice grafického rozhraní se nachází v souborech *rv_member_layout.xml* a *activity_member_list.xml*. MemberList aktivita je spuštěna po kliknutí na tlačítko s logem members, které se nachází v layoutu privátní skupiny.

MemberList.kt obsahuje konektor k připojení grafického rozhraní *activity_member-*



Obrázek 5.7: Funkce posílání a zobrazení zpráv.

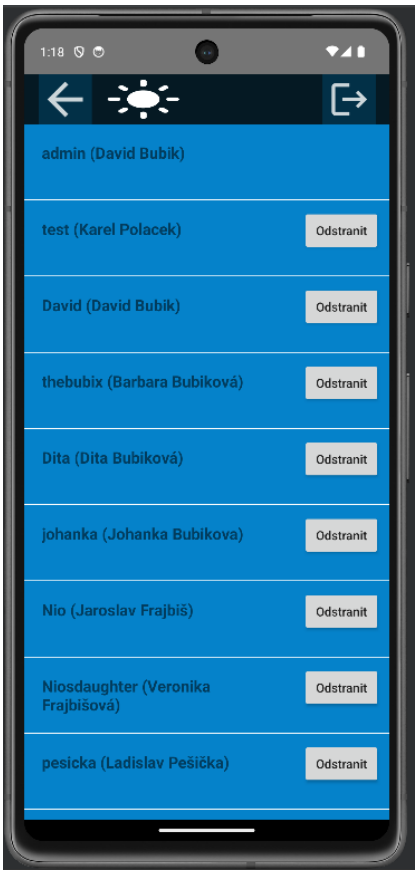
_list.xml. V tomto rozhraní se nachází prázdný seznam (*recyclerview*), který je připravený pro vložení prvků. Pomocí prvku *recyclerview.adapter* je připojena třída *MemberAdapter.kt*, které je v parametru předán list všech členů skupiny. Seznam členů je získán z Firebase Realtime database. Jednotlivé prvky generované třídou MamberAdapter obsahují jméno, příjmení, uživatelské jméno a tlačítko pro odstranění uživatele (viz obr. 5.8). Podle specifikace zadání může odstraňovat uživatele z místnosti pouze zakladatel místnosti.

Služba vytvoření kalendářní události je přístupna přes tlačítko s ikonou kalendáře, které se nachází v layoutu privátní chatové místnosti, nebo z hlavního menu aplikace (implementace služby viz kapitola 5.2.9).

5.2.9 Implementace služby sdílení kalendářních akcí

Služba sdílení kalendářních akcí využívá ke své implementaci následující třídy *CalendarEvent.kt*, *AddCalendarEvent.kt*, *RegisterEvent.kt* a *CalendarAdapter.kt*. Grafické rozhraní se nachází v souborech *activity_add_calendar_event_all.xml*, *activity_calendar_event.xml*, *activity_register_event.xml* a *rv_event_layout.xml*. Tato služba obsahuje tři části. Vytvoření události, zobrazení seznamu dostupných událostí a možnost registrace události do kalendáře.

Vytvoření události je implementováno v souboru *AddCalendarEvent.kt*. Tato funkcionalita je dostupná z přehledu všech dostupných událostí, nebo z privátní místnosti. Pro její využití je nutné mít oprávnění vedoucí nebo vyšší. Důvodem pro toto rozhodnutí je zachování konceptu validation of content. Samotné vytvoření události probíhá za pomocí jednoduchého formuláře. Po jeho vyplnění jsou data zkontrolována a poslána do databáze Firebase. Lze vytvořit dva typy události, udá-

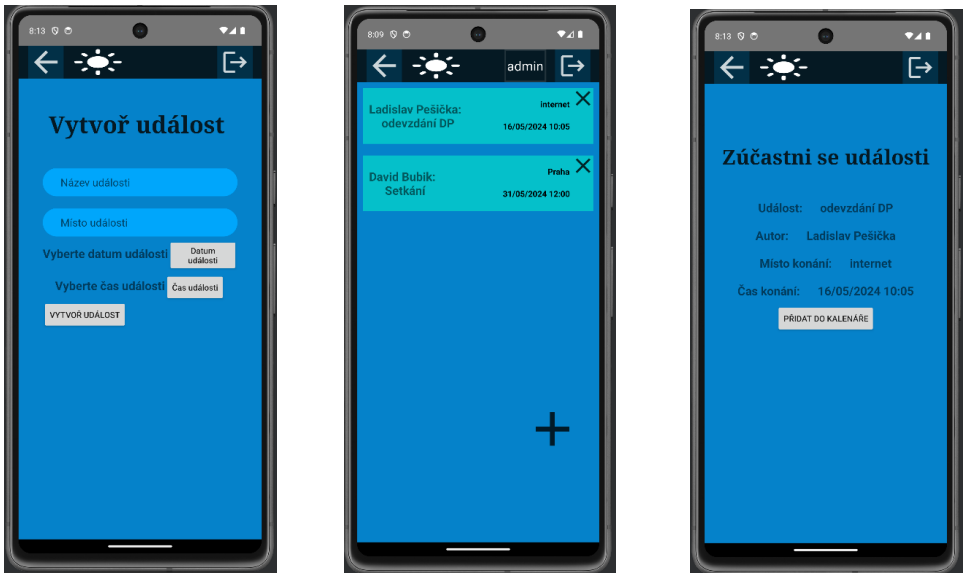


Obrázek 5.8: Odstraňování uživatelů z místnosti.

lost pro specifickou privátní skupinu, nebo pro všechny uživatele. Indikátor, o který typ události se jedná, je předán spolu s intentem pro spuštění aktivity. Pokud je třída zavolána z privátní místnosti, vytvoří se privátní událost. Pokud ne, pak je událost viditelná pro všechny uživatele.

Zobrazení seznamu událostí je implementováno ve třídě *CalendarEvent.kt*, a *CalendarAdapter.kt*. Jedná se o propojení připraveného layoutu pro seznam událostí, který je ve třídě *CalendarEvent.kt* a následného vykreslení prvků seznamu. Tuto část imlementuje třída *CalendarAdapter.kt*. Prázdný seznam je reprezentován komponentou *RecyclerView*, která obsahuje atribut adapter. Tento atribut ukazuje na instanci třídy CalendarAdapter, které byl v parametru předán seznam událostí. Pokud je nutné seznam vykreslit znovu (realtimová databáze upozorní na novou událost), je zavolána metoda *adapter.notifyOnDataSetChanged*, která upozorní adapter na nová data. Po stisknutí vybrané události jste přesměrováni na aktivitu, která implementuje možnost registrace události do lokálního kalendáře. Dále je implementována funkcionalita, která po stisknutí příslušného tlačítka (reprezentovaného ikonou) přesměruje uživatele do služby pro vytvoření nové události. Tato ikona je viditelná pouze pro uživatele s oprávněním admin či super admin.

Služba zapsání vybrané události do lokálního kalendáře je implementována ve třídě *RegisterEvent.kt*. Této třídě se v konstruktoru předají data o vybrané události. Tyto informace se přehledně zobrazí na obrazovce společně s tlačítkem *"Přidat do kalendáře"*. Po stisknutí tohoto tlačítka se spustí nový intent typu ACTION_INSERT, kterému jsou předána všechna potřebná data o události. Spuštěním tohoto intentu dojde k otevření kalendáře, který je v mobilním zařízení již nainstalován. Pokud takový kalendář neexistuje, pak se zobrazí chybová hláška. Jestliže je v mobilním zařízení nainstalováno více kalendářů, může uživatel vybrat, jaký chce použít. Po otevření kalendáře jsou údaje o události automaticky předvyplněny a uživatel musí pouze potvrdit vytvoření události v kalendáři. Po uživatelském potvrzení je uživatel vrácen zpět do aplikace (viz obr. 5.9).



(a) Vytvoření události. (b) Seznam událostí. (c) Registrace události.

Obrázek 5.9: Služba registrace kalendářních událostí.

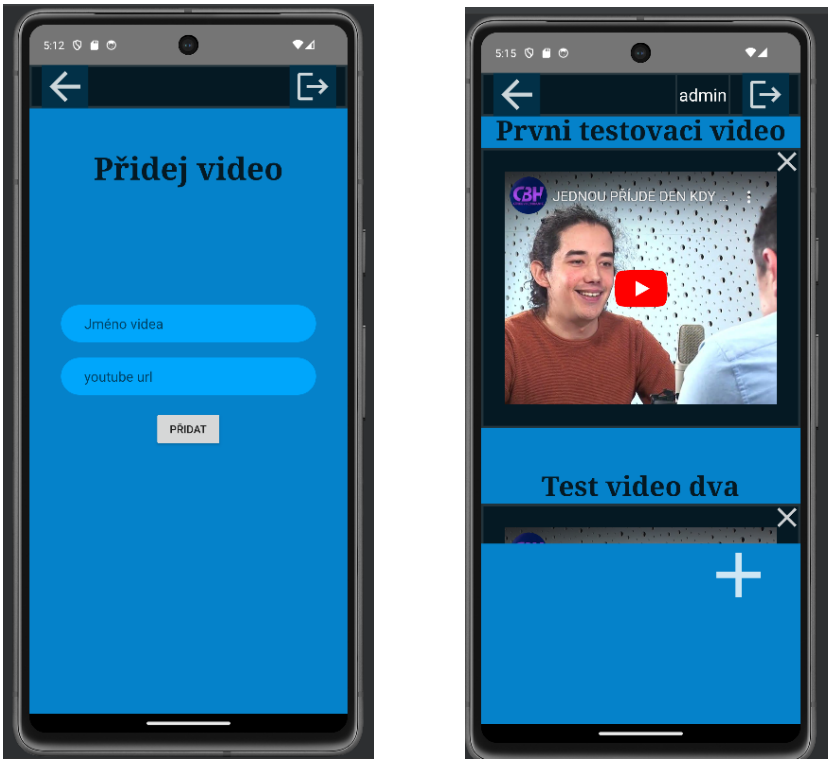
5.2.10 Implementace služby pro sdílení videí

Služba sdílení videí je implementována ve třídách *Video.kt*, *AddVideo.kt* a *VideoAdapter.kt*. Grafické rozhraní je definováno v souborech *activity_video.xml*, *activity_add_video.xml* a *rv_video_layout.xml*. Tato služba obsahuje dvě části. Přidání nového videa a zobrazení seznamu videí.

Přidání nového videa je implementováno ve třídě *addVideo.kt*, která obsahuje propojení s grafickým rozhraním ze souboru *activity_add_video.xml*. Pro sdílení videa je nutné vyplnit jednoduchý formulář, který mimo jiné obsahuje URL adresu videa. Z této adresy je následně extrahován YouTube klíč sdíleného videa. Pokud je adresa chybná, nebo se nepodaří klíč z URL adresy získat, pak se zobrazí hláška

s chybou. Pokud je správná, tak se do Firebase databáze vloží nová instance třídy VideoModel.

Zobrazení seznamu videí je implementováno ve třídách *Video* a *VideoAdapter*. Ve třídě Video je implementován layout seznamu pro zobrazení prvků, které jsou vykreslovány ve třídě VideoAdapter. K propojení adaptéru a seznamu je využit atribut *adapter* elementu *RecyclerView*. Pro zobrazení videa samotného je využit element *WebView*. Jedná se o widget, ve kterém je možné video přehrát bez opuštění aplikace. Uživatel s oprávněním super admin má možnost smazat jakékoli přidané video. Uživatel s oprávněním admin může smazat pouze videa, která sám sdílel. Pouze pro uživatele s oprávněním admin nebo super admin bude viditelná ikona pro přidání nového videa (viz obr. 5.10). Tato restrikce byla implementována po diskuzi s členy cílové skupiny z důvodu validace obsahu videí.



(a) Přidání nového videa. (b) Seznam sdílených videí.

Obrázek 5.10: Služba sdílení videí.

5.2.11 Implementace sdílení polohy

Služba sdílení polohy je implementována ve třídě *Location*. Tato třída mimo jiné implementuje rozhraní *OnMapReadyCallback* a obsahuje atributy typu *GoogleMap* a *FusedLocationProviderClient*.

Při spuštění služby se inicializují potřebné součásti a vyžádá se aktuální poloha uživatele. Aktuální poloha se zobrazí na mapě a uloží do databáze Firebase Realtime Database. Aplikace reaguje na upozornění databáze Firebase o změnách údajů polohy, a následně aktualizuje polohu uživatelů na mapě. Aktualizace polohy jsou vyžadovány a přijímány pomocí *fusedLocationClient*. Při pozastavení aplikace je poloha uživatele z databáze odstraněna a aktualizace polohy jsou zastaveny. Po obnovení aplikace dojde k restartu sledování polohy.

Pro inicializaci mapy je využita funkce *onMapReady*, ve které se dále zkontrolují oprávnění a je spuštěna funkce *getData*. Tato funkce získává data z databáze a následně zakreslí pozice uživatelů do mapy. Jedná se pouze o uživatele, kteří aktivně sdílí svou pozici. Pokud je aplikace minimalizována, je spuštěna funkce *onPause*, která přeruší sdílení polohy a vymaže záznam o geografické pozici uživatele z databáze. Pokud se uživatel navrátí do aplikace, je zavolána funkce *OnRestume*. Zde se znovu spustí proces sdílení polohy a její zaznamenávání do Firebase databáze.

5.3 Implementace webové aplikace

Webová aplikace se skládá z frontendu a backendu. Frontend je implementován v jazyce JavaScript s využitím frameworku React. Skládá se ze tří hlavních částí: login, domovská stránka a stránka pro správu uživatelů.

Backend je implementován v jazyce JavaScript za pomocí prostředí Node.JS a dodržuje následující strukturu. Soubor *index.js* je základní soubor, který spouští backend. Obsahuje základní rozřazení URL adres. Adresář routes obsahuje soubory, které podrobněji rozřazují jednotlivé služby do URL adres. Například soubor index.js definuje URL adresu *"/server/users"*. V souboru *routes/users.js* je tato URL adresa dále specifikována pro jednotlivé služby jako *"/server/users/getUsers"*, *"/server/users/setUsers"* a další. V adresáři controllers jsou soubory, které implementují jednotlivé funkce.

5.3.1 Implementace frontendu

Frontend obsahuje tři hlavní komponenty připojení: domovská stránka a správa uživatelů. Tyto komponenety jsou implementovány v souborech *Home.jsx*, *Login.jsx* a *Users.jsx*. Dále jsou pro implementaci frontendu využity následující adresáře: *Components*, ve kterém jsou sdílené komponenty pro více stránek, a *context*, který obsahuje prostor pro údaje o připojeném uživateli. Tyto údaje jsou přístupné v rámci frontendu webové aplikace.

Služba pro připojení do aplikace je implementována v souboru *Login.jsx*. V tomto souboru je definován formulář, ve kterém uživatel vyplní své přihlašovací údaje. Po stisknutí tlačítka *"Přihlásit se"* pošle tato služba požadavek pro ověření při-

hlašovacích údajů na backend. Pokud je kontrola údajů úspěšná, je uživatel přihlášen a přesměrován na domovskou stránku aplikace. Detaily o uživateli jsou uloženy v atributu LoginContext.currentUser. Pokud je ověření neúspěšné, pak se uživateli zobrazí chyba.

Domovská stránka je implementována v souboru *Home.jsx*. Na této stránce jsou zobrazeny komponenty pro přesměrování na implementované služby a možnost odhlášení. Momentálně je implementována pouze jedna služba, ale je zde připraven prostor pro případné budoucí rozšíření. Možnost odhlášení je implementována ve sdílené komponentě *Navbar*. Kontaktní údaje komunity jsou zobrazeny na komponentě *Footer*.

Stránka pro správu uživatelů je implementována v souboru *Users.jsx*. Na této stránce může uživatel s dostatečným oprávněním editovat oprávnění uživatelů, zablokovat či odblokovat uživatele v aplikaci a změnit hesla uživatelům na novou hodnotu. Zmíněné funkce jsou implementovány za pomocí přehledné tabulky, ve které je možné s využitím tlačítek editovat data o uživatelích. Pro samotnou editaci dat je využit backend aplikace. Tato stránka zašle požadavek pro editaci uživatelských dat na backend a v závislosti na odpovědi zobrazí upravená data, nebo chybu. Dále je zde zobrazena komponenta *Navbar*, která umožňuje odhlášení z aplikace a zobrazuje uživatelské jméno přihlášeného uživatele. Je také zobrazena komponenta *Footbar*.

5.3.2 Implementace backendu

Backend je implementován v jazyce JavaScrip za pomocí prostředí Node.Js. Jeho struktura je rozdělena do tří částí. První část je základní bod pro připojení k backendu. Je implementována v souboru *index.js*. Pro přesměrování na služby, které backend poskytuje je použita knihovna express.

Druhou částí je přesměrování na konkrétní poskytované služby. Soubory, za pomocí kterých je implementována tato část se nacházejí v adresáři *routes*. Pro každou kategorii poskytnutých služeb je vytvořen separátní soubor s názvem *NÁZEV_KATEGORIE.js*. Služby jsou rozděleny do několika kategorií. První je kategorie Uživatelé, která obsahuje služby pro editaci oprávnění a změnu hesla. Druhou kategorií je Připojení, která obsahuje služby pro ověření připojení do aplikace. Dále se zde nacházejí kategorie, které jsou připravené pro případné další rozšíření. Konkrétně správa místností, registrace a správa domovské obrazovky. Tyto tři kategorie jsou připraveny, avšak nejsou implementovány. V souborech kategorií uživatelé a připojení jsou definovány cesty k službám, které jednotlivé kategorie poskytují.

Poslední částí je implementace poskytnutých služeb. Tyto služby jsou implementovány v souborech, které se nachází v adresáři *controllers*. Jedná se o služby připojit,

získat data o všech uživatelích, změnit oprávnění vybraného uživatele a změna hesla vybraného uživatele.

Služba připojení uživatele je implementována s využitím knihovny bcrypt. Službě je při spuštění předáno uživatelské jméno a heslo uživatele. Následně získá z databáze data o uživateli a za pomocí knihovny bcrypt porovná hashe získaných hesel. Pokud dojde k chybě je vrácena chybová hláška *Chyba validace* se statusem 500. Pokud se hesla neshodují je vrácena chyba *Špatné jméno/heslo* se status 401. V případě, že jsou hesla shodná, je vrácena pozitivní zpráva a status 200.

Služba získat data o uživatelích je implementována v souboru *users.js*. Tato služba získá seznam všech uživatelů z Firebase databáze a vloží je do sezname prvků. V případě, že nedojde k chybě při získání dat, je vrácen seznam prvků a status 200. Pokud dojde k chybě, je vrácena chybová hláška se statusem 500. Pokud databáze vrací prázdný seznam uživatelů, je vrácena hláška *Data nenalezena* a status 404.

Služba změny oprávnění uživatele je implementována v souboru *users.js*. Službě je při spuštění předána cílová hodnota oprávnění a id uživatele. Následně se služba pokusí upravit záznam ve Firebase databázi. Pokud je editace úspěšná, je vrácena hláška *Hodnota úspěšně změněna* a status 200. V případě že dojde k chybě je vrácená chybová hláška a status 500.

Služba změny hesla uživatele je implementována v souboru *users.js*. Službě je při spuštění předáno id uživatele, kterému je heslo změněno. Dále tato služba nastaví heslo uživatele v databázi Firebase na nové heslo. Pokud nedojde k chybě, je vrácena hláška *Heslo úspěšně změněno* a status 200. V případě, že dojde k chybě, je vrácena chybová hláška a status 500.

Testování aplikace a možná rozšíření — 6

V této kapitole bude popsáno, jakým způsobem jsem otestoval funkčnost vytvořeného systému a další možná rozšíření, která by mohla být v budoucnu implementována. Tento systém je navržen pro službu komunitě. Z tohoto důvodu, testování a konzultace dalších možných rozšíření probíhalo ve spolupráci s vybranou komunitou.

6.1 Testování systému

Systém byl testován třemi typy testů. Konkrétně jednotkovými testy, testovacími scénáři a uživatelským testováním. Pro testovací scénáře a uživatelské testování byla využita část cílové komunity. Testovací skupina se skládá z 10 členů ve věkovém rozmezí 16 až 58 let.

6.1.1 Jednotkové testy

Jednotkové testování je klíčovým aspektem vývoje moderního softwaru, který se zaměřuje na ověřování jednotlivých částí zdrojového kódu, aby se zajistilo, že fungují tak, jak mají. Jednotka může být v tomto kontextu tak malá, jako jedna funkce nebo tak velká, jako třída. Hlavním cílem testování jednotek je izolovat jednotlivé části programu a ověřit jejich správnost, čímž se zvyšuje kvalita, udržovatelnost a spolehlivost softwaru.

James Whittaker uvádí definici jednotkového testování takto "jednotkové testování testuje jednotlivé softwarové komponenty nebo jejich množinu. Testeři definují vstupní doménu pro dané jednotky a zbytek systému ignorují". V tomto kontextu při využití termínu jednotkový test, je míněna definice, kterou uvedl James Whittaker [Whioo].

Některé výhody tohoto typu testování jsou včasná detekce chyb a zvýšení kvality kódu. Princip jednotkového testování je vysoce rozšířený, protože se často jedná o jednoduché krátké a čitelné části kódu, které testují funkci systému.

Včasná detekce chyb je zásadní, protože čím dříve je chyba detekována, tím méně úsilí a nákladů je nutné vynaložit na její opravu. Z tohoto důvodu jsou jednotkové testy použity při implementaci různých druhů systémů.

Druhým přínosem je, že jednotkové testování vede k čistotě kódu. Pokud má být kód jednoduše testovatelný, měl by být napsán udržitelným způsobem. Čitelnost a snadná orientace v napsaném kódu je velice důležitá pro jeho následnou údržbu. Pokud má být část kódu jednoduše testovatelná, pak by měla být napsána přehledně.

Provedené jednotkové testování je zaměřeno na otestování funkcionality jednotlivých komponent. Tento typ testu je využit pro konstruktory datových tříd a pro jednotlivé adaptéry, které jsou definovány v mobilním klientu. Důvod pro zvolení těchto tříd je, že poskytují jednoznačnou a snadno testovatelnou službu. Datové třídy vytvoří instanci třídy, která se dá poměrně snadno otestovat a adaptéry vytváří jednotlivé prvky z poskytnutého seznamu, které jsou také testovatelné. Tento typ testování byl prováděn výhradně vývojářem, bez vlivu cílové komunity.

6.1.2 Testovací scénáře

Testování systému pomocí testovacích scénářů byl druhým stupněm testování. Pokud verze systému prošla jednotkovým testováním, pak byla předána členům z testovací skupiny společně s testovacími scénáři, které bylo potřeba projít a zpracovat.

Testovací scénář je test založený na scénáři. Takový scénář by měl mít několik charakteristik:

- Test je založen na základě znalostí toho, jak se program reálně používá, včetně informací o motivaci zúčastněných osob.
- Scénář je věrohodný. Nejenže by se mohl stát v reálném světě, ale zainteresované strany by měly uvěřit, že se něco takového pravděpodobně stane.
- Příběh zahrnuje složité použití programu, složité prostředí nebo složitý soubor dat.
- Výsledky testů lze snadno vyhodnotit. To je cenné pro všechny testy, ale pro scénáře je to obzvlášť důležité, protože jsou složité[Cem13].

Vytvořené testovací scénáře, obsahují sadu testovacích případů, které je možné v rámci scénáře provést. Jednotlivé scénáře jsou rozděleny do kategorií podle typu funkcionalit, které testují. Následně budou uvedeny některé scénáře.

6.1.2.1 Testovací scénář registrace nového uživatele

Prvním uvedeným testovacím scénářem je ověření funkce registrace nového uživatele. V tomto scénáři se uživatel s libovolným oprávněním snaží zaregistrovat do systému. Scénář obsahuje následující testovací případy.

Testovací případ vytvoření nového uživatele.
V tomto testovacím případě bude otestována správná funkce zaregistrování nového uživatele. Uživatel vyplní korektní data ve zobrazeném formuláři a zaregistruje se.
Seznam jednotlivých kroků:

- Po otevření aplikace klikněte na tlačítko "REGISTRACE".
- Do pole Jméno vyplňte jakékoli jméno s českou diakritikou.
- Do pole Příjmení vyplňte jakékoli příjmení s českou diakritikou.
- Do pole Uživatelské jméno vyplňte jakékoli uživatelské jméno.
- Do pole Heslo vyplňte své heslo.
- Zkontrolujte, že jsou všechna pole vyplněna.
- Klikněte na tlačítko "ZAREGISTROVAT".

Korektním výsledkem je zobrazení upozornění s textem "Registrace úspěšná", nebo "Uživatelské jméno zabrané". Pokud jste si jisti, že zadané uživatelské jméno je unikátní a stále se vám zobrazuje hláška "Uživatelské jméno zabrané", pak se jedná o chybný výsledek.

Testovací případ vytvoření nového uživatele, se zabraným uživatelským jménem.
V tomto testovacím případě bude otestována správná funkce zaregistrování nového uživatele, který zadává již obsazené uživatelské jméno. Uživatel vyplní korektní data a ve zobrazeném formuláři a pokusí se zaregistrovat.
Seznam jednotlivých kroků:

- Po otevření aplikace klikněte na tlačítko "REGISTRACE".
- Do pole Jméno vyplňte jakékoli jméno s českou diakritikou.
- Do pole Příjmení vyplňte jakékoli příjmení s českou diakritikou.
- Do pole Uživatelské jméno vyplňte uživatelské jméno, které je již zabrané.
- Do pole Heslo vyplňte své heslo.

- Zkontrolujte, že jsou všechna pole vyplněna.
- klikněte na tlačítko "ZAREGISTROVAT".

Korektním výsledkem je zobrazení upozornění s textem "Uživatelské jméno zabrané". Pokud se zobrazí jakékoliv jiné hlášení, jedná se o chybný výsledek.

Testovací případ vytvoření nového uživatele, s chybějícími daty.
V tomto testovacím případě bude otestována správná funkce zaregistrování nového uživatele, který zapomene zadat jeden, či více požadovaných údajů. Uživatel vyplní data a ve zobrazeném formuláři a pokusí se zaregistrovat.
Seznam jednotlivých kroků:

- Po otevření aplikace klikněte na tlačítko "REGISTRACE".
- Do pole Jméno vyplňte jakékoli jméno s českou diakritikou.
- Pole příjmení nechte prázdné.
- Do pole Uživatelské jméno vyplňte uživatelské jméno.
- Do pole Heslo vyplňte své heslo.
- Zkontrolujte, že alespoň jedno pole není vyplněno.
- klikněte na tlačítko "ZAREGISTROVAT".

Korektním výsledkem je zobrazení upozornění s textem "Vyplňte prosím všechny údaje". Pokud se zobrazí jakákoli hláška, jedná se o chybný výsledek.

6.1.2.2 Testovací scénář připojení uživatele

Dalším testovacím scénářem je ověření funkce připojení registrovaného uživatele. V tomto scénáři se uživatel s libovolným oprávněním snaží připojit do mobilní aplikace. Scénář obsahuje následující testovací případy.

Testovací případ korektního připojení uživatele.
V tomto testovacím případě je otestována funkcionalita připojení do mobilní aplikace. Uživatel zadá všechny požadované údaje korektně a pokusí se připojit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Do pole uživatelské jméno vyplňte korektní uživatelské jméno.

- Do pole Heslo vyplňte odpovídající heslo.
- klikněte na tlačítko "PŘIHLÁSIT SE".

Správným výsledkem je zobrazení obrazovky s menu implementovaných služeb a zobrazení správného uživatelského jména v horní části obrazovky.

Testovací případ připojení uživatele se špatnými údaji.
V tomto testovacím případě je otestována funkcionalita připojení do mobilní aplikace. Uživatel zadá všechny požadované údaje, některé nekorektně a pokusí se připojit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Do pole uživatelské jméno vyplňte jakékoli přihlašovací jméno.
- Do pole Heslo vyplňte neodpovídající heslo.
- klikněte na tlačítko "PŘIHLÁSIT SE".

Správným výsledkem je zobrazení upozornění s textem "Špatné uživatelské jméno nebo heslo".

Testovací případ připojení uživatele s nevyplněnými údaji.
V tomto testovacím případě je otestována funkcionalita připojení do mobilní aplikace. Uživatel zadá některé údaje a pokusí se připojit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Do pole uživatelské jméno vyplňte jakékoli přihlašovací jméno.
- Pole Heslo nechte prázdné.
- klikněte na tlačítko "PŘIHLÁSIT SE".

Správným výsledkem je zobrazení upozornění s textem "Vyplňte prosím všechny údaje".

6.1.2.3 Testovací scénář pro založení události, dostupné všem uživatelům

Pro tento scénář je nutné, aby uživatel měl oprávnění admin, nebo super admin. Tento uživatel založí událost, která je přístupná všem uživatelům.

Testovací případ korektního založení veřejně přístupné události
V tomto testovacím případě je otestována funkcionalita založení veřejně přístupné události. Uživatel s oprávněním admin, nebo super admin zadá všechny údaje a pokusí se událost založit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Uživatel se přihlásí pod účtem který má oprávnění admin, nebo super admin.
- Klikněte na tlačítko "Události".
- Klikněte na ikonu +, která se nachází v dolní části obrazovky.
- Vyplňte pole "Název události".
- Vyplňte pole "Místo události".
- Zvolite datum a čas, které se nenachází v minulosti.
- Klikněte na tlačítko "Vytvoř událost".

Správným výsledkem je přesměrování na seznam událostí, ve kterém je nově vytvořená událost viditelná.

Testovací případ založení veřejně přístupné události v minulosti
V tomto testovacím případě je otestována funkcionalita založení veřejně přístupné události v minulosti. Uživatel s oprávněním admin, nebo super admin zadá všechny údaje a pokusí se událost založit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Uživatel se přihlásí pod účtem, který má oprávnění admin, nebo super admin.
- Klikněte na tlačítko "Události".
- Klikněte na ikonu +, která se nachází v dolní části obrazovky.
- Vyplňte pole "Název události".
- Vyplňte pole "Místo události".
- Zvolit datum a čas, které se nachází v minulosti.
- Klikněte na tlačítko "Vytvoř událost".

Správným výsledkem je zobrazení chybové hlášky s obsahem "Nelze vytvořit událost v minulosti".

Testovací případ založení veřejně přístupné události s chybějícími údaji
V tomto testovacím případě je otestována funkcionalita založení veřejně přístupné události s chybějícími údaji. Uživatel s oprávněním admin, nebo super admin zadá některé údaje a pokusí se událost založit.
Seznam jednotlivých kroků:

- Po otevření aplikace uživatel uvidí přihlašovací obrazovku.
- Uživatel se přihlásí pod účtem, který má oprávnění admin, nebo super admin.
- Klikněte na tlačítko "Události".
- Klikněte na ikonu +, která se nachází v dolní části obrazovky.
- Vyplňte pole "Název události".
- Nevyplnit pole "Místo události".
- Zvolite datum a čas, které se nachází v minulosti.
- Klikněte na tlačítko "Vytvoř událost".

Správným výsledkem je zobrazení chybové hlášky s obsahem "Je nutné vyplnit všechna pole".

6.1.2.4 Závěr testovacích scénářů

Během scénářového testování byly vytvořeny scénáře pro otestování následujících služeb: přihlášení, registrace, založení privátní/veřejné chatovací místnosti, sdílení videí, vytvoření kalendářních akcí, snížení/zvýšení oprávnění uživatele a změna hesla. Připomínky uživatelů získané během testování byly zapracovány.

6.1.3 Uživatelské testování

Hlavním cílem uživatelských testů je zlepšit použitelnost testovaného produktu. Dalším cílem je zlepšit proces, kterým jsou produkty navrhovány a vyvíjeny, za účelem vyhnutí se stejným problémům u dalších produktů [DF12].

Ačkoli mohou existovat velké rozdíly v tom, kde a jak se uživatelské testy provádí, většina testů má tyto společné charakteristiky: [DF12].

- Primárním cílem je zlepšit použitelnost produktu. Každý test také může mít konkrétnější cíle, které formulujete při plánování testu.

- Účastníci testu jsou skuteční uživatelé.
- Účastníci plní skutečné úkoly.
- Zaznamenáváte, co uživatelé v systému dělají.
- Analyzujete nalezené problémy.

Uživatelské testování tohoto systému bylo provedeno za pomocí cílové komunity. Testovací skupina se skládala z deseti účastníků ve věkovém rozmezí 16 až 58 let. Hlavním cílem tohoto testování bylo zjistit, zda je systém uživatelsky přívětivý pro nehomogenní skupinu účastníků. V požadavcích pro tento systém je uvedeno, že by měl být snadno a intuitivně použitelný i pro méně technicky zdatné uživatele. Z tohoto důvodu testovací skupina zahrnovala uživatele s různým věkem, dosaženým vzděláním a technickou zdatností. Předmětem tohoto druhu testování bylo mimo jiné zjistit, zda je možné vyvinutý systém v cílové komunitě používat, a zda nechybějí některé vlastnosti, které je nutné implementovat.

Tento test probíhal několik měsíců a systém byl na základě tohoto testování aktualizován. Bylo přidáno několik služeb, které se z počátku nejevily jako zbytné. Mezi klíčové poznatky patří nejednoznačnost sdílení polohy, při zapomenutém heslu se nebylo možné přihlásit, absence hlášky pro chybné přihlášení a zobrazení prázdných widgetů při vložení neexistujícího videa.

Služba sdílení polohy byla matoucí, protože uživatelům nebyly poskytnuty základních informace při startu služby. Někteří uživatelé si nebyli jistí, jak a kdy svou pozici poskytují. Z tohoto důvodu žádali o zobrazení bližšího popisu služby.

Absence hlášky pro chybné přihlášení byla matoucí pro většinu uživatelů. Uživatel, který si nebyl jist, zda jsou uvedené údaje korektní, nebyl informován o neúspěšném přihlášení. Nebyl si tak jist, zda jsou poskytnuté údaje nekorektní, nebo zda proces přihlašování stále trvá.

Vytvořený systém podporuje pouze přehrání videí z platformy YouTube. Třída, která reprezentuje video je však připravena pro následnou implementaci jiných platforem. Z tohoto důvodu bylo v první verzi aplikace možné nahrát URL, které nevedlo na platformu YouTube, obsah tohoto URL samozřejmě nebyl zobrazen.

Celkově uživatelské testování poskytlo cenné poznatky o vlastnostech vyvíjeného systému, které bylo třeba zlepšit. Vyřešením zjištěných problémů a zavedením doporučených zlepšení, byla zajištěna vyšší uživatelská přívětivost, která vede k příjemnější zkušenosti při využívání vytvořeného systému.

6.2 Možná rozšíření systému

Vytvořený systém má relativně velký prostor pro implementaci dalších služeb, které by poskytly lepší uživatelskou zkušenost. Systém byl navržen a implementován způ-

sobem, aby následná rozšíření byla poměrně snadno začlenitelná do zbytku aplikace. Jednotlivá rozšíření jsou rozdělena do skupin podle části systému, do kterého by mohla být začleněna.

6.2.1 Možná rozšíření mobilního klienta

Mobilní klient bude využíván všemi typy uživatelů. Z tohoto důvodu musí být přehledný a jednoznačný. Proto není doporučena implementace většího množství dalších služeb, které by případně zahltily technicky méně zdatné uživatele. Rozšíření již poskytnutých služeb je však vhodné.

Služba sdílení videí by mohla být expandována pro obsažení více platforem než jen YouTube. Vybrané platformy by měly být validované a bezpečné.

Dále by bylo vhodné poskytnout možnost sdílení stažitelných audiozáznamů. Tato služba by rozšiřovala službu sdílení videí, kde by byla přístupná jako jedna položka v Tab menu. Důvod pro vhodnost implementace této služby je, že někteří uživatelé chtějí mít možnost poslouchat audio záznamy z pořádaných akcí v offline prostředí.

Ke skupinovým chatům by bylo vhodné přidat možnost zasílání hlasových zpráv. Tato vlastnost je využívána převážně mladšími uživateli. Jedná se však o službu, která může být atraktivní i pro jiné skupiny uživatelů.

Dalším možným rozšířením mobilní aplikace je služba video nebo audio hovorů. Toto je služba, kterou by bylo nutné implementovat zvlášť a jednalo by se o další službu, která by byla dostupná z menu aplikace.

Bylo by také vhodné rozšířit menu nastavení o další funkcionality. Například customizace velikosti fontu, stylu fontu nebo barevné palety. Další funkcí, která by mohla být implementována v menu nastavení je změna uživatelského jména.

Poslední možné rozšíření je implementace chatbota, který bude odpovídat na základní otázky. Takový chatbot by byl vhodný pro všechny typy uživatelů. Jeho hlavní úlohou by bylo zodpovídat otázky ohledně ovládání aplikace a úrovní oprávnění.

6.2.2 Možné rozšíření webové aplikace

Webová aplikace je určena pro uživatele s vyšším oprávněním. Jedná se o administrativní část aplikace. Z tohoto důvodu rozšíření této části aplikace jsou administrativního druhu.

Jedno z navrhovaných rozšíření je možnost spravovat chatovací místnosti z webové aplikace. V tomto rozšíření by bylo možné spravovat všechny místnosti, které uživatel vytvořil. Konkrétně vytvářet akce a přijímat či odstraňovat členy místností.

Dalším případným rozšířením je získání historie kalendářních akcí. Vytvořené kalendářní akce nejsou viditelné na seznamu akcí, pokud již proběhly. Získání celé historie by mohlo být vhodné pro prezentování akcí, které komunita uskutečnila.

Závěr 7

Hlavním cílem této diplomové práce bylo navrhnout, vytvořit a ověřit funkčnost komunikačního systému určeného pro podporu komunity. Tento systém poskytuje následující služby: Správu uživatelů, skupinový chat, sdílení akcí v kalendáři, sdílení videí a sdílení své geografické polohy. V rámci práce jsem navrhl, implementoval a otestoval komunikační systém.

Vytvořený systém slouží pro komunikaci mezi členy komunity. Obsahuje služby pro posílání zpráv, sdílení videí, zakládání chatovacích místností, zakládaní kalendářních akcí a správu uživatelů. Systém se skládá z mobilní aplikace, ve které je implementována komunikační část s přehledným uživatelským rozhraním a webové aplikace, která obsahuje administrativní část. Dále je grafické rozhraní celého systému navrženo s velkým důrazem na vytvoření bezpečného a přehledného prostředí v rámci komunity i pro technicky méně zdatné uživatele.

V této práci jsem nejdříve prozkoumal vybrané existující systémy určené pro komunikaci v rámci komunity, porovnal jejich vlastnosti a nenalezl jsem žádnou, která by vyhovovala všem požadavkům z cílů práce. V následujícím kroku jsem provedl návrh nového systému z hlediska jeho funkčnosti, architektury a vzhledu. Navržené aplikace (mobilní a webová) společně poskytují služby pro komunikaci v rámci komunity. Navržený systém jsem implementoval a otestoval jeho funkčnost.

Komunikační systém lze snadno rozšířit o další služby a byl otestován ve spolupráci s cílovou komunitou. Tento systém představuje vhodné řešení komunikačního systému pro danou komunitu, které splňuje všechny body zadání diplomové práce.

Instalační příručka příručka

A

Instalační příručka obsahuje pokyny pro nasazení systému.

A.1 Nasazení systému

Systém je nasazen ve čtyřech krocích. Prvním z nich je vytvoření Firebase účtu. V tomto účtu je nutné založit nový projekt a povolit vybrané služby. Druhým krokem je spuštění mobilní aplikace. Třetím krokem je nasazení backendu webové aplikace a posledním krokem je nasazení frontendu webové aplikace.

A.1.1 Vytvoření účtu v platformě Firebase

Pro vytvoření účtu v platformě Firebase je nutné mít e-mailový účet. Po založení účtu je nutné povolit následující služby: realtimová databáze, FCM tokeny, autentizaci anonymních uživatelů a vygenerovat API klíč pro připojení z webového prohlížeče. Posledním krokem je editace pravidel pro připojení do realtimové databáze. V těchto pravidlech je nutné povolit přístup všem autentizovaným uživatelům.

A.1.2 Spuštění mobilní aplikace

Při spuštění aplikace je nutné změnit odkaz do realtimové databáze. Ten lze změnit přímo v aplikaci, za pomocí funkce, která je dostupná pod ikonou nastavení na přihlašovací obrazovce, nebo v kódu v souboru *utils.kt*.

A.1.3 Nasazení backendu webové aplikace

Při nasazení backendu je nutné zadat následující proměnné prostředí.

- DATABASE_URL - URL adresa Firebase Realtime Database.
- API_KEY - Klíč pro připojení k Firebase z webového prohlížeče.
- API_TOKEN - Token pro identifikaci zpráv zasílaných frontendem.

Po jejich zadání backend běží na vybraném serveru.

A.1.4 Nasazení frontendu

Při nasazení frontendu je nutné vytvořit následující proměnné prostředí:

- NODE_VERSION - Verze Node.
- REACT_APP_URL_BACKEND - URL adresa vedoucí na backend.
- REACT_APP_TOKEN_BACKEND - Token identifikující zprávy vedoucí na backend (musí být shodný s API_TOKEN)

Po jejich vyplnění frontend běží na serveru.

Uživatelská příručka B

Uživatelská příručka obsahuje návod pro používání systému.

B.1 Ovládání mobilní aplikace

V této části popíši ovládání mobilní aplikace, která představuje hlavní část systému.

B.1.1 Registrace a připojení

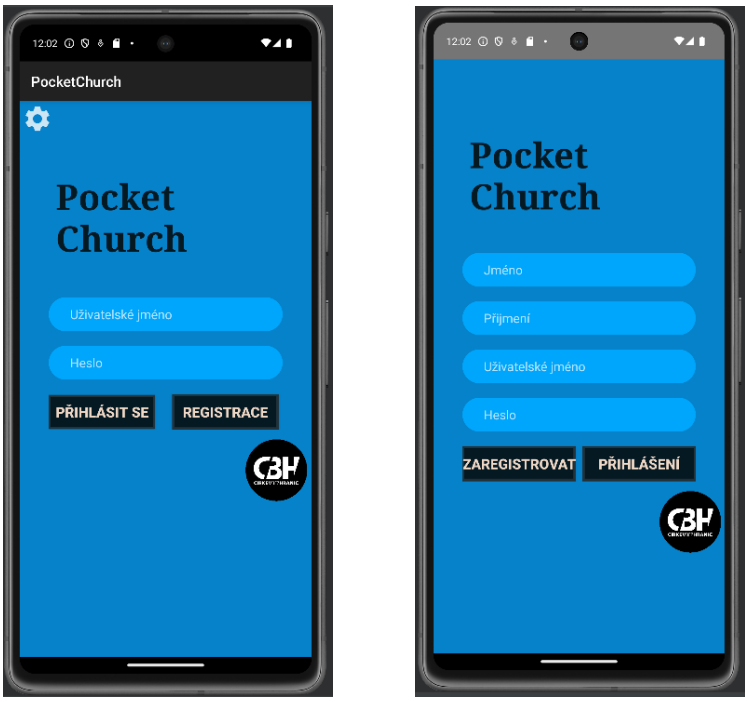
Při prvním spuštění aplikace je uživatel přesměrován na obrazovku pro přihlášení (viz obr. B.1a). Následně je nutné stisknout tlačítko *REGISTRACE*, které přesměruje uživatele na obrazovku pro registraci (viz obr. B.1b). Zde se uživatel zaregistruje a po stisknutí tlačítka *REGISTRACE* je po úspěšném zaregistrování přesměrován zpět na obrazovku pro přihlášení, kde se přihlásí.

B.1.2 Vytvoření chatovacích místností a posílání zpráv

Po úspěšném přihlášení je uživatel přesměrován na obrazovku se základním menu (viz obr. B.2a). Zde stiskne tlačítko *POSÍLÁNÍ ZPRÁV*, které ho přesměruje na obrazovku se seznamem vytvořených chatovacích místností (viz obr. B.2b). V seznamu chatovacích místností uživatel vybere do které místnosti vstoupit a začít psát. Uživatelé s oprávněním vedoucí, admin nebo super admin vidí ikonu *+*, po jejímž stisknutí jsou přesměrováni na obrazovku pro vytvoření chatovací místnosti.

B.1.3 Vytváření a sdílení kalendářních akcí

Po úspěšném přihlášení je uživatel přesměrován na obrazovku se základním menu (viz obr. B.2a). Zde stiskne tlačítko *UDÁLOSTI*, které ho přesměruje na obrazovku s přehledem vytvořených událostí. Pokud si přeje zapsat událost do vlastního kalendáře, musí vybranou událost stisknout. Uživatelé s oprávněním admin a super admin mohou také událost vytvořit. Pro vytvoření události je nutné stisknout ikonu



(a) Připojení. (b) Registrace.

Obrázek B.1: Připojení a registrace.

+, která se nachází v přehledu událostí. Po stisknutí této ikony je uživatel přesměrován na obrazovku pro vytvoření nové události.

B.1.4 Sdílení videí

Videa do aplikace může sdílet pouze uživatel s oprávněním admin, nebo super admin. Sdílené video je streamováno z platformy YouTube. Uživatelé s nižším oprávněním se mohou na videa pouze dívat.

Pro shlížení videí je nutné na obrazovce s hlavním menu (viz obr. B.2a) stisknout tlačítko *VIDEA*, které uživatele přesměruje na obrazovku se seznamem videí. Jednotlivá videa lze následně ve widgetu spustit. Pro přidání nového videa je nutné stisknout ikonu +, která přesměruje uživatele na obrazovku pro zadání nového videa.

B.1.5 Sdílení geografické polohy

Pro sdílení geografické polohy je nutné mít zapnuto získání polohy v nastavení telefonu. Dále je nutné udělit aplikaci oprávnění přístupu k datům o geografické poloze mobilního zařízení.

Pokud má aplikace všechna potřebná oprávnění, pak uživatel stiskne tlačítko *LOKACE*. Po stisknutí tohoto tlačítka se ukáže hláška se základními informacemi o

(a) Hlavní menu. (b) Přehled místností.

Obrázek B.2: Hlavní menu a přehled místností.

sdílení polohy. Po odsouhlasení hlášky uživatel sdílí svou polohu a zobrazí se mapa s polohou všech uživatelů, kteří ji sdílí. Jakmile uživatel obrazovku opustí, sdílení polohy přestane.

B.1.6 Změna hesla

Pro změnu hesla je nutné v hlavním menu stisknout tlačítko *NASTAVENÍ*, které uživatele přesměruje na obrazovku s možným nastavením (viz obr. B.3a). Na této obrazovce uživatel stiskne tlačítko *ZMĚNA HESLA*, které uživatele přesměruje na obrazovku pro změnu hesla (viz obr. B.3b)

(a) Menu nastavení. (b) Změna hesla.

Obrázek B.3: Menu nastavení a změna hesla.

Obrázek B.4: Webová stránka pro správu uživatelů

B.2 Ovládání webové aplikace

Webová aplikace obsahuje nástroje pro správu uživatelů. po připojení může uživatel využívat tyto nástroje za pomocí poskytnuté tabulky (viz obr. B.4). Do této aplikace mají přístup pouze uživatelé s oprávněním admin a super admin.

Struktura přiloženého zip souboru

C

- `Aplikace_a_knihovny` ... Tento soubor obsahuje spouštěcí scripty aplikace, knihovny a zdrojové soubory
  - `client` ..... Obsahuje frontend webové aplikace
    - `PocketChurchFront` ..... Obsahuje frontend webové aplikace
      - `build` ..... Obsahuje manifest, CSS soubory a média
      - `node_modules` ..... Obsahuje externí knihovny
      - `public` ..... Obsahuje fonty a další veřejné podklady
      - `src` ..... Obsahuje zdrojové kódy
    - `Dokumentace` ..... Obsahuje JSDoc dokumentaci
  - `server` ..... Obsahuje backend webové aplikace
    - `PocketChurchBack` ..... Obsahuje backend webové aplikace
      - `controllers` ..... Obsahuje zdrojové kódy
      - `node_modules` ..... Obsahuje externí knihovny
      - `routes` ..... Obsahuje zdrojové kódy
    - `Dokumentace` ..... Obsahuje JSDoc dokumentaci
  - `mobil` ..... Obsahuje mobilní aplikaci
    - `PocketChurch` ..... Obsahuje zdrojové soubory
    - `APK` ..... Obsahuje instalační soubor mobilní aplikace
    - `Dokumentace` ..... Obsahuje Javadoc dokumentaci
  - `readme.txt` ..... Obsahuje pokyny k nasazení systému
- `Text_prace` ..... Text diplomové práce
- `Poster` ..... Obsahuje vypracovaný poster k diplomové práci
- `Vstupni_data` ..... Vstupní datová sada a informace o předpřipravených uživatelích
- `readme.txt` ..... Popis struktury přiloženého zip souboru

Bibliografie

[Mel18] "MELISSA HUBBARD
, "Matthew J. Bailey". *Mastering Microsoft Teams*. Apress, 2018. ISBN 9781484236697.

[ACP+21] ANDREOLI, Remo; CUCINOTTA, Tommaso; PEDRESCHI, Dino et al. RT-MongoDB: A NoSQL database with differentiated performance. In: *Proceedings of the 11th International Conference on Cloud Computing and Services Science-CLOSER*. Science a Technology Publications (SciTePress), 2021, s. 77–86.

[AWS24] AWS. *What's the Difference Between Frontend and Backend in Application Development?* Amazon, 2024-05-04. Dostupné také z: `https://aws.amazon.com/compare/the-difference-between-frontend-and-backend/`.

[BEN21] BATUBARA, Toras; EFENDI, Syahril; NABABAN, Erna. Analysis Performance BCRYPT Algorithm to Improve Password Security from Brute Force. *Journal of Physics: Conference Series*. 2021, roč. 1811, s. 012129. Dostupné z DOI: 10.1088/1742-6596/1811/1/012129.

[Cem13] CEM KANER, JD. An introduction to scenario testing. *Florida Institute of Technology, Melbourne*. 2013, s. 1–13.

[DBR16] DAUZON, Samuel; BENDORAITIS, Aidas; RAVINDRAN, Arun. *Django: web development with Python*. Packt Publishing Ltd, 2016. ISBN 1787121380.

[Del+15] DELIA, Lisandro; GALDAMEZ, Nicolas; THOMAS, Pablo; CORBALAN, Leonardo; PESADO, Patricia. Multi-platform mobile application development analysis. In: *2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS)*. 2015, s. 181–186. Dostupné z DOI: 10.1109/RCIS.2015.7128878.

[DF12] DUMAS, Joseph S; FOX, Jean E. Usability testing. *The human-computer interaction handbook: fundamentals, evolving technologies, and emerging applications*. 2012, s. 1221–1241. ISBN 9780429103971.

[Eis15] EISENMAN, Bonnie. *Learning react native: Building native mobile apps with JavaScript*. "O'Reilly Media, Inc.", 2015. ISBN 1491989149.

[Faca] FACEBOOK. *Messenger*. Facebook. Dostupné také z: https://messengernews.fb.com/about/.

[Facb] FACEBOOK. *Messenger App Basics* [https://www.facebook.com/help/messenger-app/basic/117818065545664]. [B.r.]. Accessed: May 15, 2024.

[Facc] FACEBOOK. *Messenger pictures*. Facebook. Dostupné také z: https://messengernews.fb.com/resources/messenger-product-screens/.

[Gac15] GACKENHEIMER, Cory. *Introduction to React*. Apress, 2015. ISBN 1484212460.

[GS15] GILSKI, Przemyslaw; STEFANSKI, Jacek. Android os: A review. *Tem Journal*. 2015, roč. 4, č. 1, s. 116.

[Goo24] GOOGLE. *Android Studio*. Google, 2024-03-04. Dostupné také z: https://developer.android.com/studio/.

[GHA05] GOSLING, James; HOLMES, David Colin; ARNOLD, Ken. *The Java programming language*. Addison-Wesley, 2005.

[HPM15] HATZIVASILIS, George; PAPAEFSTATHIOU, Ioannis; MANIFAVAS, Charalampos. *Password Hashing Competition - Survey and Benchmark* [Cryptology ePrint Archive, Paper 2015/265]. 2015. Dostupné také z: https://eprint.iacr.org/2015/265. https://eprint.iacr.org/2015/265.

[Hun21] HUNT, John. *Beginner's Guide to Kotlin Programming*. Springer, 2021. ISBN 3030808920.

[Inc] INC., Discord. *WHAT IS DISCORD*. Discord Inc. Dostupné také z: https://discord.com/creators/what-is-discord.

[Khe+17] KHEDKAR, Sonam; THUBE, Swapnil; ESTATE, WI; NAKA, C et al. Real time databases for applications. *International Research Journal of Engineering and Technology (IRJET)*. 2017, roč. 4, č. 06, s. 2078–2082.

[KD13] KOZLOWSKI, Pawel; DARWIN, Peter Bacon. *Mastering Web Application Development with AngularJS.* Packt Pub., 2013. ISBN 1782161821.

[Lee09] LEE, Alison. Mobile web widgets: Enabler of enterprise mobility work. In: *2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), Spain, Page*. Citeseer, 2009. Č. 86-94.

[LLP] LLP, Telegram Messenger. *What is Telegram?* Telegram Messenger LLP. Dostupné také z: https://telegram.org/faq#q-what-is-telegram-what-do-i-do-here.

[Mah+19] MAHALI, Muhammad Izzuddin; SUMARYANTO; PUTRO, Nur Hidayanto Pancoro Setyo; RAHMAT, Baiquni. Android and FIREBASE mBaaS-based Information System Design of Students Activity Unit (SAU) Using the Rational Unified Process (RUP) Method. In: *Proceedings of the 2019 International Conference on Mathematics, Science and Technology Teaching and Learning*. 2019, s. 6–12.

[Med] MEDIA, Viber. *About Viber*. Viber Media. Dostupné také z: https://www.viber.com/en/about/.

[Mica] MICROSOFT. *MS teams*. Microsoft. Dostupné také z: https://www.microsoft.com/cs-cz/microsoft-teams/.

[Micb] MICROSOFT. *MS teams Pricing*. Microsoft. Dostupné také z: https://www.microsoft.com/cs-cz/microsoft-teams/enterprise#pricing.

[Mic24] MICROSOFT. *What is .NET MAUI?* [online]. 2024. [cit. 2024-05-14]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-8.0.

[Mon] MONGO. *MongoDB Pricing*. Mongo. Dostupné také z: https://www.mongodb.com/pricing/.

[Mor17] MORONEY, Laurence. The Firebase Realtime Database. In: *The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform*. Berkeley, CA: Apress, 2017. ISBN 978-1-4842-2943-9. Dostupné z DOI: 10.1007/978-1-4842-2943-9_3.

[Mül04] MÜLLER MIROSLAV, Hana Kanisová a. *UML srozumitelně*. 1. Brno: Computer Press, 2004. ISBN 8025110834.

[MST22] MYTHILY, M.; SAMSON ARUN RAJ, A.; THANAKUMAR JOSEPH, Iwin. An Analysis of the Significance of Spring Boot in The Market. In: *2022 International Conference on Inventive Computation Technologies (ICICT)*. 2022, s. 1277–1281. Dostupné z DOI: 10.1109/ICICT54344.2022.9850910.

[Pez+16] PEZOA, Felipe; REUTTER, Juan L; SUAREZ, Fernando; UGARTE, Martín; VRGOČ, Domagoj. Foundations of JSON schema. In: *Proceedings of the 25th international conference on World Wide Web*. 2016, s. 263–273.

[SAP24] SAP. *Theta joins*. SAP, 2024-05-04. Dostupné také z: https://help.sap.com/docs/SAP_BUSINESSOBJECTS_BUSINESS_INTELLIGENCE_PLATFORM/512fca6758c4495bb6a50fe3e1e4b892/464ee9ff6e041014910aba7db0e91070.html.

[Slaa] SLACK TECHNOLOGIES, Inc. *Slack*. Slack Technologies, Inc. Dostupné také z: https://slack.com/intl/en-gb/help/articles/115004071768-What-is-Slack-.

[Slab] SLACK TECHNOLOGIES, Inc. *Slack ceny*. Slack Technologies, Inc. Dostupné také z: https://slack.com/intl/en-gb/pricing.

[Sta] STATISTA. *Most popular global mobile messenger apps as of January 2024, based on number of monthly active users*. statista. Dostupné také z: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/.

[TV10] TILKOV, Stefan; VINOSKI, Steve. Node.js: Using JavaScript to Build High-Performance Network Programs. *IEEE Internet Computing*. 2010, roč. 14, č. 6, s. 80–83. Dostupné z DOI: 10.1109/MIC.2010.145.

[wha] WHATSAPP. *WhatsUp about*. Dostupné také z: https://www.whatsapp.com/about.

[Whi00] WHITTAKER, James A. What is software testing? And why is it so hard? *IEEE software*. 2000, roč. 17, č. 1, s. 70–79.

[Win+19] WINGERATH, Wolfram et al. Real-Time Databases. *Real-Time & Stream Data Management: Push-Based Data in Research & Practice*. 2019, s. 21–41.

[Wu18] WU, Wenhao. React Native vs Flutter, Cross-platforms mobile application frameworks. 2018.

[Yak] YAKOVLIEVA, Olena. *whatsapp-interface-template*. vecteezy. Dostupné také z: https://www.vecteezy.com/vector-art/13473361-whatsapp-interface-template-on-mobile-phone.

[Yus19] YUSOFF, Muhamad Saiful Bahri. ABC of content validation and content validity index calculation. *Education in Medicine Journal*. 2019, roč. 11, č. 2, s. 49–54.

Seznam použitých zkratek

API Application Programming Interface.

BSON Binary JSON.

DOM Document Object Model.

FCM Firebase cloud messaging.

GUI Grafické Uživatelské Rozhraní

HTML Hypertext Markup Language.

IT Informační technologie.

JSON JavaScript-Object-Notation.

MVC Model-view-controler.

NoSQL Not only SQL.

RAD Rapid Application Development

SDK Service Development Kit

SQL Structured Query Language.

URL Uniform Resource Locator

UML Unified Modeling Language.

WORA Write once, run anywhere.

XML Extensible Markup Language

Seznam obrázků

Seznam tabulek