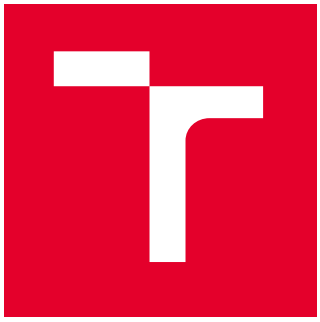


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

BEZPEČNOST MOBILNÍCH APLIKACÍ PRO SYSTÉM ANDROID

ANDROID MOBILE APPLICATION SECURITY

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — **Ing. Jakub Michálek**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Václav Oujezský, Ph.D.**

SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Ing. Jakub Michálek ***ID:*** 186140

Ročník: 2 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Bezpečnost mobilních aplikací pro systém Android

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vývoj ukázkové aplikace pro mobilní operační systém Android, která bude demonstrovat reálná bezpečnostní rizika spojená s poskytnutím práv přístupu uživatelem. Prostudujte a v teoretické části popište systém Android a možná bezpečnostní rizika zaměřená na uživatele. V praktické části pak v jazyce Kotlin naprogramujte výše popsanou aplikaci dle doporučených postupů. Práce vyžaduje jednak znalosti z oblasti bezpečnosti aplikací, dále však znalost programování. Výstupem diplomové práce je zpracovaná teoretická část s návrhy scénářů, logický nákres (schéma) a technický popis výsledné aplikace. Dále je výstupem funkční aplikace v Android Studio.

DOPORUČENÁ LITERATURA:

[1] Android Developers [online]. USA: Google, 2021 [cit. 2021-9-8]. Dostupné z: https://developer.android.com/

[2] Kotlin Programming Language [online]. USA: Kotlin Foundation, 2021 [cit. 2021-9-8]. Dostupné z: https://kotlinlang.org/

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 24.5.2022

Vedoucí práce: Ing. Václav Oujezský, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Obsahem této práce je seznámení s mobilním operačním systémem Android a jeho bezpečnostními riziky pro běžného uživatele. Pro demonstraci těchto rizik byla vytvořena aplikace, která je na praktických příkladech představuje uživateli.
V první části práce je seznámení s operačním systémem Android, jeho architektura, verze a základní komponenty. Druhá část práce popisuje rizika spojená s tímto operačním systémem. Další část práce popisuje oprávnění udělována aplikacím a nebezpečí, která mohou vzniknout jejich odsouhlasením. Poslední část práce se zabývá návrhem a implementací aplikace z programového hlediska.

KLÍČOVÁ SLOVA

Android, Kotlin, mobilní aplikace, oprávnění Android

ABSTRACT

The thesis's objectives are to introduce a computing platform for Android mobile devices and its security risks for the average user. The application that was created presents these risks on practical examples.
The first part of the diploma thesis introduces the Android operating system, its architecture, version, and basic components. Security risks of this operating system are described in a second part. The following section is devoted to Android permission and the danger which can arise from their authorization. The final section of the work delves into design and implementation of the application from the programming point of view.

KEYWORDS

Android, Kotlin, mobile application, permissions on Android

MICHÁLEK, Jakub. *Bezpečnost mobilních aplikací pro systém Android*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 87 s. Diplomová práce. Vedoucí práce: Ing. Václav Oujezský, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Ing. Jakub Michálek
VUT ID autora:	186140
Typ práce:	Diplomová práce
Akademický rok:	2021/22
Téma závěrečné práce:	Bezpečnost mobilních aplikací pro systém Android

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Václavu Oujezskému, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V současné době má u sebe chytrý telefon skoro každý. Přes 70 procent z nich je založeno na operačním systému Android. Jelikož tyto zařízení poskytují uživatelům spoustu různých funkcí, jsou stále více využívány k bezpečnostně kritickým účelům, jako je internetové bankovnictví, potvrzování plateb nebo přihlašování k online účtům. Také na nich dochází neustále ke sběru citlivých uživatelských dat, mezi které patří například textové zprávy, informace o poloze, historie hovorů nebo fotografie a videa.

Operační systém (OS) Android díky jeho zastoupení na světovém trhu je často předmětem útoků a tyto útoky mohou mít velký dopad na mnoho uživatelů. Proto vývojáři tohoto systému kladou velký důraz na jeho bezpečnost a poskytují tvůrcům mobilních aplikací mnoho prostředku jak zabezpečit jejich aplikace proti zneužití a úniku citlivých dat. Jelikož je ale OS Android poskytován jako „open source“ licence a aplikace pro tento systém může vyvíjet kdokoliv, vzniká zde prostor k jejich zneužití. Jednou z možností je zneužití stávající aplikace, která je nedostatečně zabezpečena a útočník nemá problém ji napadnout. Druhou možností je vytvoření účelně závadné aplikace, která bude uživateli škodit.

Cílem této práce je představení bezpečnostních rizik OS Android, které se týkají běžného uživatele. Dále návrh a vytvoření mobilní aplikace, jejíž účelem bude demonstrovat běžnému uživateli reálná bezpečnostní rizika. Konkrétně zneužití systémových oprávnění za účelem zisku a možného dalšího zpracování citlivých uživatelských dat.

V rámci první kapitoly je představen OS Android. Jsou zde popsány jeho verze, architektura a struktura aplikací.

Druhá kapitola se věnuje bezpečnosti aplikací, především rizikům s nimi spojeném. Ty nejzávažnější rizika jsou zde uvedeny a podrobně popsány.

Třetí kapitola se zabývá přímo bezpečnostními riziky spojenými se systémovými oprávněními OS Android a jejich možným zneužitím.

Poslední kapitola je věnována přímo návrhu a realizaci mobilní aplikace sloužící k demonstraci reálných bezpečnostních rizik pro běžného uživatele.

Část I

Teoretická část

1 Operační systém Android

Android je open-source platforma [1], založená na jádře Linux, určená hlavně pro mobilní zařízení, tedy chytré telefony, fotoaparáty a navigace. Nachází se i v televizních přijímačích, autech a různých dalších chytrých zařízeních.

Systém Android vyvíjí organizace Open Handset Alliance, jejíž součástí jsou desítky firem, včetně těch nejznámějších v mobilní branži – Google, HTC, Intel, NVIDIA atd. Jde o jeden z mála systémů, které podporují více platforem, a můžete ho vidět na zařízeních různých značek. [1]

Největší výhodou a zároveň nevýhodou platformy je její otevřenost a možnost úprav, ať už ze strany výrobců nebo uživatelů. Úpravy se netýkají jen konfigurace či widgetů[2], ale i firmwaru[3]. Zařízení s operačním systémem Android dodává na trh mnoho firem. Je to záruka dynamičtějšího vývoje nových zařízení, ale na druhou stranu to představuje problém, protože dané aplikace běží na různých přístrojích s různým rozlišením displeje a různým výkonem procesoru. V praxi to znamená různý komfort uživatelského rozhraní. Zdrojem informací pro tuto kapitolu byla primárně kniha – Vývoj aplikací pro Android. [2]

[1] Platforma s volně přístupným zdrojovým kódem.

[2] Ovládací prvek, který slouží k interakci mezi uživatelem a grafickým rozhraním programu.

[3] Software, který slouží k řízení základních komponent zařízení.

1.1 Verze OS Android

Za dobu, co je systém Android na trhu, bylo vydáno několik jeho aktualizací, které opravují chyby a přidávají nové funkce. Zavedl se standart pro psaní programů pro konkrétní verzi systému tzv. úroveň API[4]. Proto před vývojem aplikace je velice důležité zvolit cílovou verzi systému, resp. volbu úrovně API. Verze systému android je volena tak, aby bylo zacíleno na co největší počet uživatelů.

Google zveřejňuje statistiky zastoupení jednotlivých verzí systémů Android, které přistupují do služby Google Play. Vzhledem k těmto údajům se vývojář může rozhodnout, kterou verzi systému zvolit pro svou aplikaci. Pro svoji aplikaci jsem zvolil úroveň API 19 a vyšší, jelikož funkce obsažené v knihovnách jsou dostatečné a aplikace bude funkční na 98,1 % zařízeních. [3]

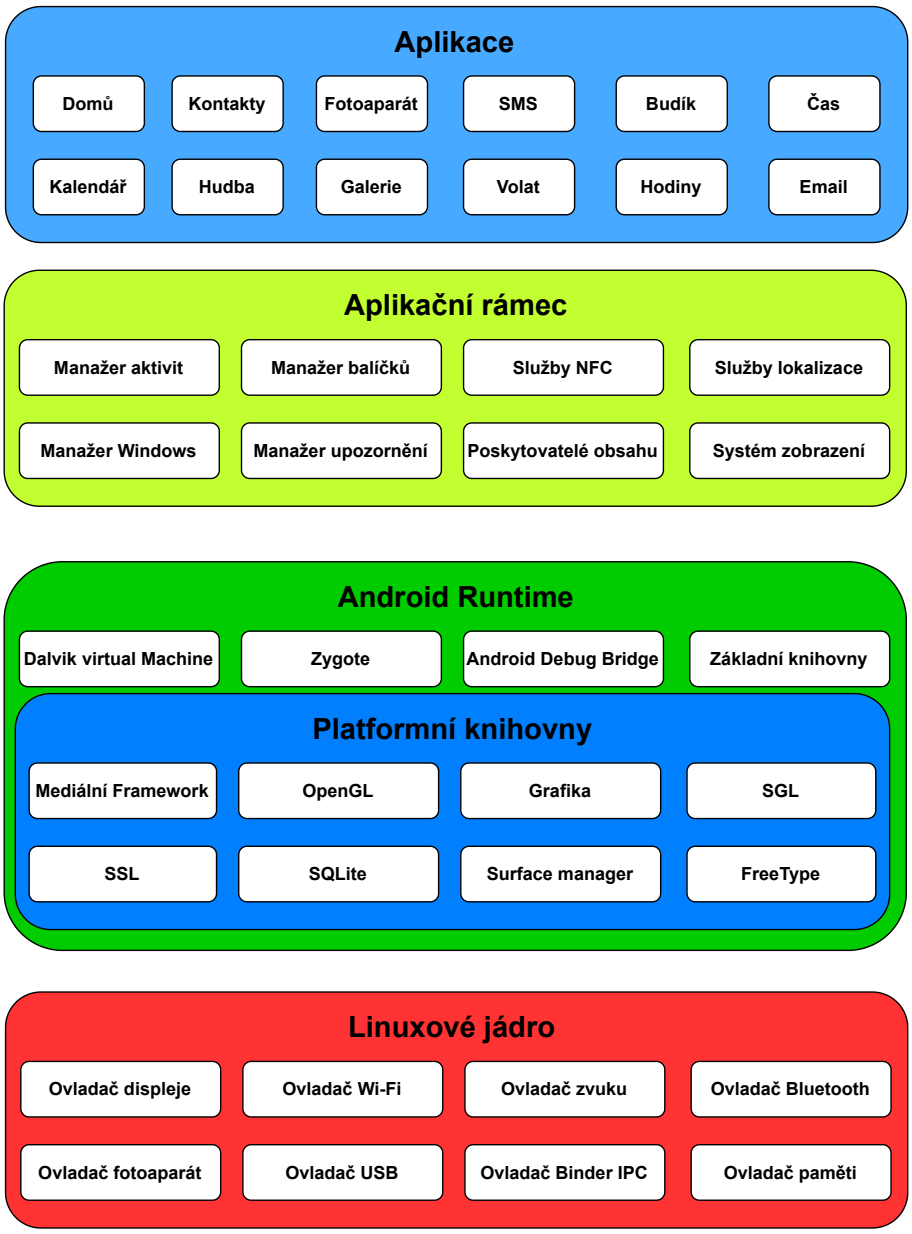
Verze OS Android	Level API	Kumulativní distribuce [%]
4.1 Jelly Bean	16	99,8
4.2 Jelly Bean	17	99,2
4.3 Jelly Bean	18	98,4
4.4 KitKat	19	98,1
5.0 Lollipop	21	94,1
5.1 Lollipop	22	92,3
6.0 Marshmallow	23	84,9
7.0 Nougat	24	73,7
7.1 Nougat	25	66,2
8.0 Oreo	26	60,8
8.1 Oreo	27	53,5
9.0 Pie	28	39,5
10. Android 10	29	8,2

Tab. 1.1: Podíl jednotlivých verzí OS Android na trhu.

[4]API je sada nástrojů a protokolů pro vytváření aplikací. Slouží k usnadnění práce programátorovi.

1.2 Architektura OS Android

Architektura operačního systému Android je rozdělena do pěti vrstev. Každá vrstva má svůj účel a nemusí být přímo oddělena od ostatních. Jeho architektura [4] je zobrazena na obrázku 1.1.



Obr. 1.1: Architektura OS Android. [4]

1.2.1 Aplikace

Aplikace [4] představuje nejvyšší vrstvu architektury. Dělí se na nativní aplikace a aplikace třetích stran. Aplikační vrstva je obsluhována při běhu systému za pomoci tříd a služeb z aplikačního rámce.

1.2.2 Aplikační rámec

Část systému, která je nejčastěji využívána vývojáři aplikací. Poskytuje třídy používané k vytváření aplikací pro Android a také obecnou abstrakci pro přístup k hardwaru. Stará se též o uživatelské rozhraní a prostředky aplikace. [4]

1.2.3 Android Runtime

Obsahuje důležité komponenty, jako jsou základní knihovny a virtuální počítač Dalvik. Android Runtime pohání aplikace společně s knihovnami a tvoří základ aplikačního rámce.

Dalvik Virtual Machine (DVM) je, stejně jako Java Virtual Machine (JVM), registrově založený virtuální stroj. Je speciálně navržen a optimalizován pro Android, aby bylo zajištěno, že zařízení může efektivně spouštět více instancí. Při vytváření vláken a správě paměti na nízké úrovni se spoléhá na Linuxové jádro.

Základní knihovny v Android Runtime umožňují implementovat aplikace pro Android pomocí standardního programovacího jazyka JAVA [4].

1.2.4 Knihovny platformy

Nad jádrem je situovaná vrstva knihoven, která poskytuje přímý přístup aplikaci k různým komponentům systému Android. Jsou to nativní knihovny napsané v jazyce C/C++. Tvoří mezivrstvu mezi různými komponenty vyšších vrstev a linuxovým jádrem [4]. Níže jsou uvedeny některé ze základních knihoven systému Android:

- Mediální knihovna pro přehrávání a nahrávání audio a video formátů.
- Knihovna Surface manager, která poskytuje správu zobrazení.
- Knihovny SGL a OpenGL Graphics pro 2D a 3D grafiku.
- SQLite pro podporu databází a FreeType pro podporu fontů.
- Web-Kit pro podporu webového prohlížeče a SSL pro zabezpečení spojení do internetu.

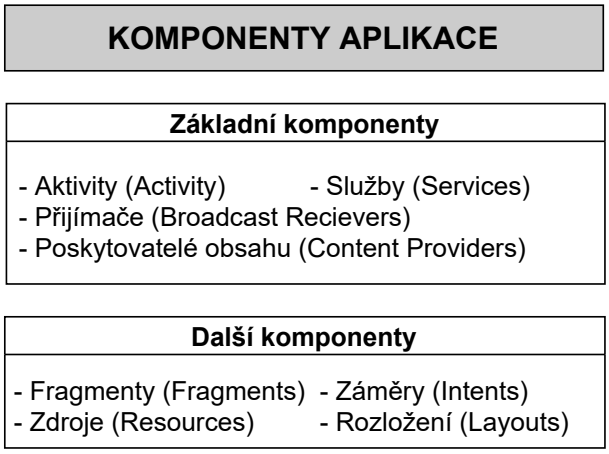
1.2.5 Linuxové jádro

Linuxové jádro je základním pilířem a nejnižší vrstvou architektury systému Android. Jedná se o upravené jádro populárního operačního systému Linux, kde úpravy zahrnují redukce funkcí a jejich přizpůsobení možnostem mobilních zařízení. Tato vrstva zabezpečuje komunikaci mezi hardwarem a softwarem. [4]

Jádro zabezpečuje správu paměti, správu procesů, základní síťovou vrstvu a ovladače. Řízení procesů umožňuje, aby více procesů běželo současně, aniž by se vzájemně ovlivňovaly. Součástí jádra je i správa napájení, která zabezpečuje, aby se energeticky náročnější moduly, tedy procesor a obrazovka, při delší nečinnosti vypínaly.

1.3 Aplikace v OS Android

Mobilní aplikace pro systém Android jsou tvořeny ze základních a dalších komponent, obrázek 1.2. Základní komponenty jsou důležitými stavebními kameny každé aplikace pro Android a deklarují se v souboru `AndroidManifest.xml`. Tento soubor popisuje každou hlavní komponentu aplikace a způsob, jakým mezi sebou jednotlivé komponenty interagují. [5]



Obr. 1.2: Komponenty aplikace pro OS Android. [5]

1.3.1 Aktivity

Základní úlohou aktivity je zobrazovat uživateli údaje z nižších vrstev v požadované formě a umožnit mu aplikaci ovládat pomocí grafického uživatelského rozhraní. Každá aktivita je potomkem třídy `android.app.Activity`. Aktivita také reaguje na události jejího životního cyklu a překrývá příslušné metody.

Aktivita většinou zabírá celou plochu displeje, ale není to pravidlem. Každá aktivita má vlastní životní cyklus a je potřeba počítat s tím, že instance aktivit mohou být v odůvodněných případech odstraněny. Například pokud začnou docházet systémové zdroje, systém se pokusí odstranit nepoužívané komponenty.

Vždy jedna z aktivit je hlavní, a to ta, která se zobrazí uživateli ihned po spuštění aplikace. Pokud je spuštěna další z aktivit, předchozí se pozastaví ,ale zůstane v paměti paměťového prostoru nazývaného Back Stack. Tento soubor obsahuje informace o aktuálně spuštěných aktivitách a umožňuje mezi nimi jednoduše přecházet.

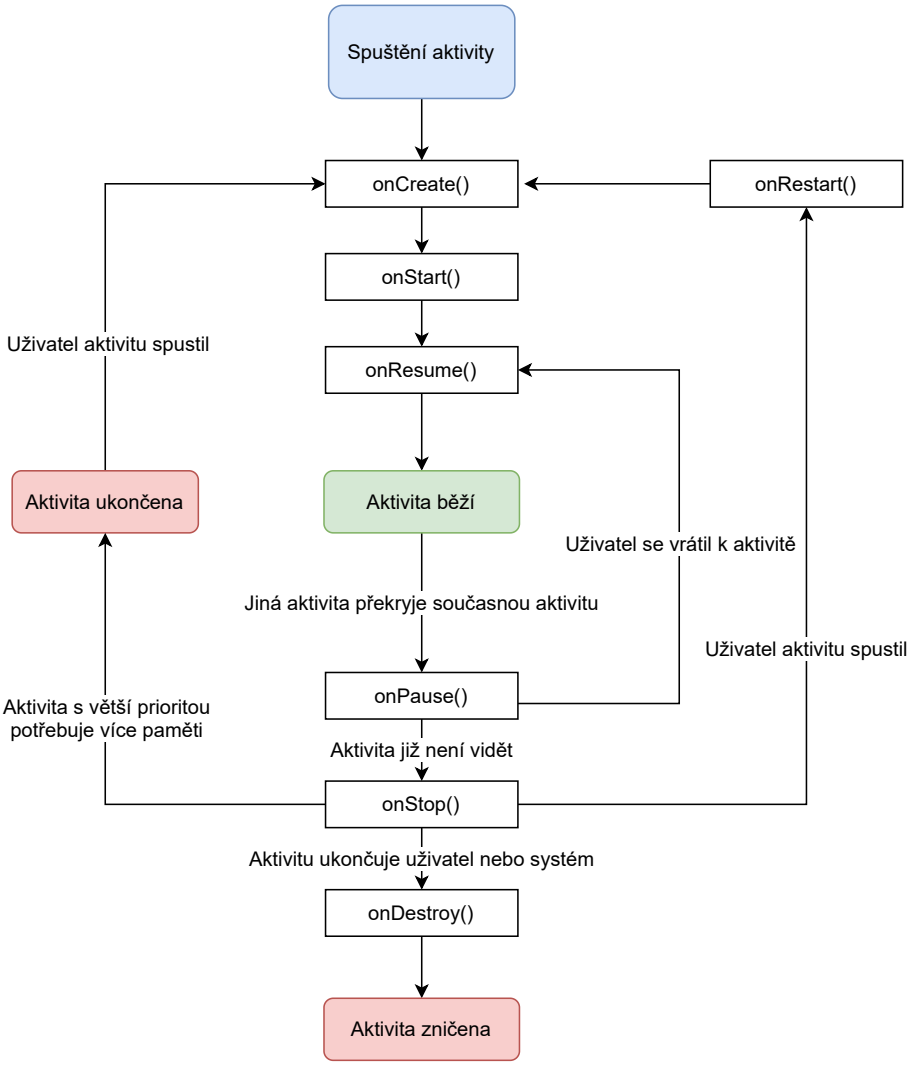
Životní cyklus aktivity

Je definován metodami, které se spouští v přesně daných situacích a v určeném pořadí.

Životní cyklus aktivit je uveden na diagramu 1.3 a skládá se ze tří hlavních fází:

- Aktivita na popředí – je zobrazena na displeji a umožňuje uživateli interakci s aplikací. Aktivity v tomto stavu jsou ve fázi cyklu `Running` nebo `Resumed`.
- Pozastavená aktivita – nachází se stále v paměti a zpravidla je částečně viditelná. Uživatel k ní ale nemá přístup. Typickým příkladem je překrytí aktivity dialogovým oknem.
- Zastavená aktivita – je kompletně překrytá jinou aktivitou. Zůstává stále uložena v paměti Back Stack a lze se tedy k ní rychle vrátit. V případě nedostatku paměti jsou tyto aktivity odstraňovány.

Při vývoji aplikací je nutné vždy brát ohled na životní cyklus aktivity. Jednotlivé fáze tohoto cyklu jsou popsány níže.



Obr. 1.3: Graf životního cyklu aktivity. [6]

OnCreate() – metoda, která slouží ke spuštění aktivity. Definují se v ní prvky grafického uživatelského rozhraní (dále jen GUI) a konfigurují proměnné a objekty potřebné k běhu aktivity. Po zavolání této metody je aktivita stále zastavená, neviditelná a nekomunikuje s uživatelem.

Metoda `onCreate()` se volá:

- Při prvním spuštění aktivity.
- Pokud aktivita již běžela, ale byla překryta jinou aktivitou a následně se uživatel vrátil k té původní.
- Pokud byl původní proces odstraněn v důsledku nedostatku paměti.
- Aktivita běží, ale je nutná výměna zdrojů, například při otočení displeje, odpojení hardwarového prvku nebo změně jazyka aplikace.

V této metodě jsou také získávány informace z předchozích aktivit za pomocí záměrů (intent) a objektu Bundle.

onStart() a onResume() – Při volání této metody aktivita přechází do popředí. Dochází k realizaci činností potřebných k zobrazení uživatelského rozhraní aktivity. Rozdíl mezi metodami `onStart()` a `onResume()` je v tom, že první z nich se volá při spuštění aktivity a druhá, pokud přechází aktivita z pozadí na popředí.

onPause() – V případě, že je spuštěná jiná aktivita, přechází ta předchozí do pozadí. V této metodě je vhodné automaticky uložit změny údajů, se kterými aktivita pracovala.

Obvykle se v ní řeší uložení aktuálních neobnovitelných dat, ukončení snímání senzorů, ukončení všech animací a operací náročných na centrální procesorovou jednotku (dále jen CPU) a podobně.

onStop() – Volá se při zastavení aktivity z jiného důvodu, než je nedostatek paměti. Aktivita je stále uložena v Back Stacku a uživatel se k ní může snadno vrátit. Pokud tak udělá, volá se metoda `onRestart()`.

onDestroy() – S pomocí této metody je aktivita zcela ukončena.

1.3.2 Služby (Services)

Služby se používají pro pravidelné a nepřetržitě vykonávané akce, které nevyžadují grafické rozhraní. Příkladem může být stahování souborů nebo přehrávání hudby na pozadí. Služby se spouštějí, zastavují a kontrolují z jiných komponent aplikace, například z aktivit, Broadcast Receiverů či jiných služeb.

Služby mají dvě hlavní výhody, a to, že umožňují:

- Provádět činnosti na pozadí i v případě, že systém aplikaci služby ukončí.
- Aby kód služby v jednom procesu mohl komunikovat s kódem v jiném procesu.

Výhodou služeb je i to, že jsou méně náchylné k násilnému ukončení systémem při nedostatku paměti, a i když k tomu dojde, tak při dostatku paměti jsou znovu inicializovány.

1.3.3 Přijímače nesměrového vysílání (Broadcast receivers)

Komponenta, díky které aplikace ví o jednotlivých systémových událostech. Dá se také nazývat jako přijímač oznámení, kde oznámení může obsahovat informaci o přijetí krátké textové zprávy (dále jen SMS) či multimediální zprávy (dále jen MMS), o nízké hodnotě energie v baterii zařízení, o změně souřadnic globálního polohového systému (dále jen GPS) nebo vložení SD karty.

Kromě systémových událostí je možné vytvářet události i samotnou aplikací a ty následně zachytávat pomocí broadcastu. Přestože broadcast nemá vlastní uživatelské rozhraní, dokáže vytvářet notifikace, které uživatele přes Notification Manager upozorňují na různé události.

Aplikace můžou mít libovolný počet přijímačů oznámení, které jsou pro danou aplikaci relevantní. Tyto přijímače se vytvářejí jako rozšíření základní třídy `BroadcastReceiver`.

1.3.4 Poskytovatelé obsahu (Content providers)

Systém používá poskytovatele obsahu k usnadnění přístupu ke sdíleným datům, jako jsou mediální soubory, informace o kontaktech a kalendáři. Aplikace tak mohou snadno přistupovat k údajům jiných aplikací, které také vystupují jako poskytovatelé obsahu.

1.3.5 Fragmenty

Komponenty, které běží v kontextu aktivity. Fragment zapouzdřuje kód aplikace, takže se dá snadno opakovaně použít a umožňuje diferencované zobrazování na zařízeních s různou velikostí displeje.

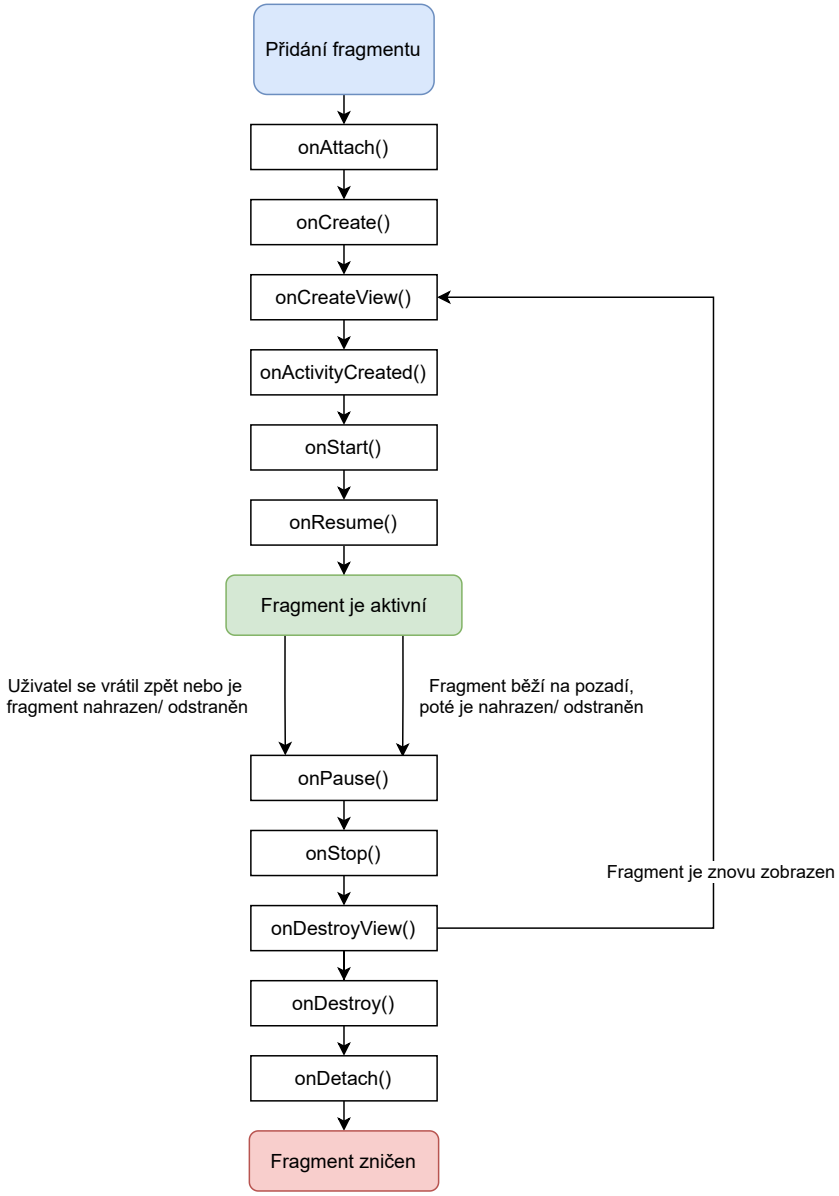
Implementace uživatelského rozhraní je zapouzdřena ve fragmentech. Aktivity fungují pouze jako kontejnery pro fragmenty, přičemž fragmenty v aktivitě se dají dynamicky měnit.

Životní cyklus fragmentu

Z hlediska životního cyklu funguje fragment v rámci aktivity a do značné míry je ovlivněn jejím životním cyklem. Volání metod aktivity koresponduje s voláním příslušných metod ve fragmentu.

Životní cyklus fragmentu je uveden na diagramu 1.4 a skládá se ze tří hlavních fází:

- Resumed (spuštěný) – fragment je zobrazen na popředí.
- Paused (pozastavený) – tento stav nastane, pokud je aktivita v rámci níž je fragment zapouzdřen, částečně překryta jinou aktivitou, například dialogem.
- Stopped (zastavený) – fragment se nezobrazuje, avšak pokud je aktivita uložena v Back Stacku, instance fragmentu stále existuje. Pokud je aktivita odstraněna z Back Stacku, jsou odstraněny i všechny fragmenty zapouzdřené v ní.



Obr. 1.4: Graf životního cyklu fragmentu. [7]

Fragment obsahuje metody, které se v životním cyklu aktivity nenachází:

- onAttach() – metoda, která se aktivuje ještě před onCreate() při asociaci fragmentu s aktivitou, která je fragmentu odevzdávaná jako argument.
- onCreateView() – aktivuje se při vytváření uživatelského rozhraní fragmentu.
- onActivityCreated() – volá se při vytvoření hostitelské aktivity.
- onDestroyView() – volá se při odstranění fragmentu z uživatelského rozhraní.
- onDetach() – volá se při zrušení asociace mezi fragmentem a příslušnou aktivitou.

1.3.6 Záměr (Intent)

Pomocí záměru jsou definovány vstupy a výstupy aktivit. Umožňují vzájemnou komunikaci mezi volně vázanými komponenty aplikace.

Jak již vyplývá z názvu, záměr oznamuje, co má aktivita v úmyslu. Například pokud je potřeba poslat SMS, vyhledat soubor, aktivovat fotoaparát a podobně. V praxi to funguje tak, že systém vyhledá všechny aktivity, které jsou schopné záměr splnit a pokud je jich více, tak si uživatel mezi nimi vybere a následně je daná aktivita spuštěna.

Záměry se dělí na dvě skupiny podle jejich směrování:

- **Explicitní** – je v nich přesně určeno, kterou akci mají provést, kdo a jakým způsobem ji provede a s jakými údaji. V aplikaci je těchto záměru například využíváno ke spouštění jiných aktivit z aktivity hlavní.
- **Implicitní** – je v nich pouze určeno jaká akce se má provést, ale ne jakým způsobem. Systému je pouze předána tato informace, na základě které rozhodne, s pomocí jaké aktivity se záměr uskuteční. Pokud je více možných aktivit je volba na uživateli.

1.3.7 Zdroje (Resources)

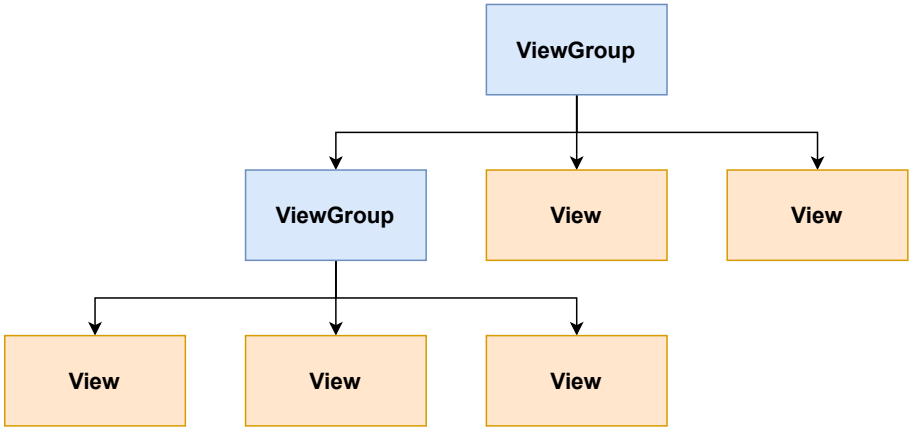
V aplikacích se využívá mnoho definic objektů. Typickým příkladem jsou textové řetězce, definice barev, tvarů a podobně. k ukládání slouží adresář zdrojů `/res`.

V této složce se nachází mnoho různých objektů v závislosti na účelu aplikace. Nejčastěji v ní najdeme tyto vnořené podsložky:

- *Layout* – definice rozmístění prvků uživatelského rozhraní.
- *Menu* – definice struktury menu.
- *Strings* – definice textových řetězců.
- *Colors* – definice barev použitých v aplikaci.
- *Dimens* – rozměry v jednotkách dp (density independent pixel).
- *Drawable* – definice obrázků a tvarů pro vykreslování.
- *Styles* – definice stylů pro prvky uživatelského rozhraní.

1.3.8 Rozložení (Layouts)

Rozložení definuje strukturu uživatelského rozhraní v aplikaci, například v aktivitě. Všechny jeho elementy jsou založeny na hierarchii objektů `View` a `ViewGroup`. `View` obvykle reprezentuje nějaký objekt, se kterým může uživatel interagovat. `ViewGroup` je naproti tomu neviditelný kontejner, který definuje rozložení objektu `View` na obrazovce, jak lze vidět na obrázku 1.5.



Obr. 1.5: Zobrazení hierarchie View pro rozložení UI. [8]

Objekty typu `View` se nazývají „widgety“ a je jich mnoho druhů, například tlačítka nebo textové pole. Objekty `ViewGroup` se nazývají „layouty“ a dělí se na základě struktury rozložení prvků v nich, například `LinearLayout` nebo `ConstraintLayout`.

Rozložení jsou psány v jazyce Extensible Markup Language (dále jen XML) a umožňují přehledně oddělit vizuální prezentaci aplikace od kódu, který řídí její chování. Použití souborů XML také usnadňuje poskytování různých rozložení pro různé velikosti a orientace obrazovky. [8]

2 Rizika spojená s mobilními aplikacemi

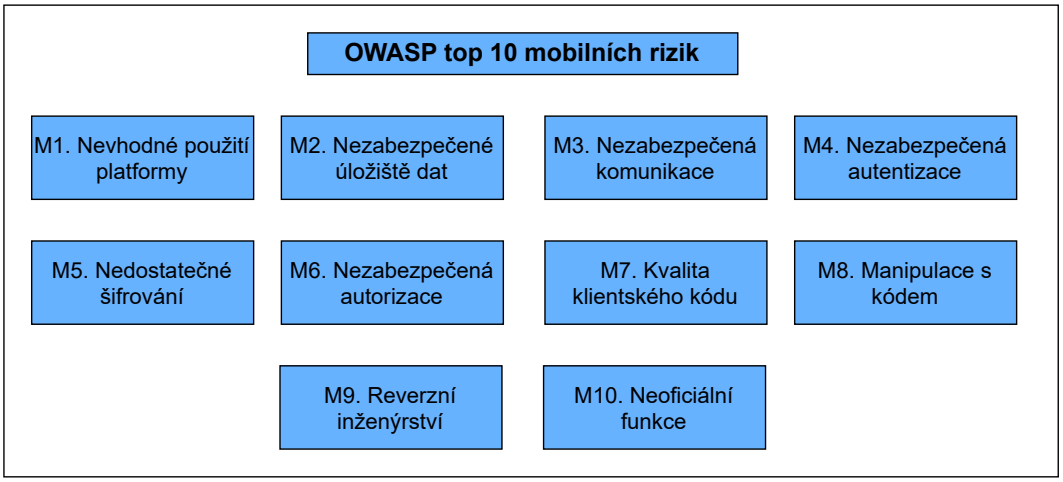
Aplikace pro mobilní zařízení dnes čelí mnohým rizikům. Většina z nich se odvíjí od klasických útoků na webové a desktopové aplikace, nicméně existují i rizika přímo spjatá s oblastí mobilních technologií a jejich využitím:

- Jelikož naprostá většina aplikací pro svou plnou funkcionalitu potřebuje přístup k internetu, tudíž využívá architektury klient-server. Dochází často ke vzájemné komunikaci přes nezabezpečenou bezdrátovou (dále jen Wi-Fi) nebo mobilní síť. Tuto komunikaci může útočník odchytit či modifikovat, a tím se například dostat k citlivým datům uživatele.
- Mobilní zařízení nosíme neustále při sobě, tudíž jedním z rizik je jeho ztráta či odcizení. Pokud k tomu dojde, může se útočník dostat k citlivým údajům uložených v paměti zařízení.
- Velká část rizik pochází přímo ze samotného mobilního zařízení. Například malware stažený z neoriginálních zdrojů či útoky z ostatních nainstalovaných aplikací. Tyto aplikace mohou například shromažďovat citlivá data uživatele, a poté je proti němu využít za účelem zisku či jeho poškození.
- Mobilní aplikace mohou zpracovávat data z mnoha zdrojů, takže hrozí útoky na neobvyklé vstupní body, např. blízkopolní komunikace (dále jen NFC), Bluetooth, fotoaparát, mikrofon, SMS zprávy, univerzální sériová sběrnice (dále jen USB) či Quick Response (dále jen QR) kódy.

2.1 Rizika podle modelu OWASP

Základem pro testování bezpečnosti mobilních aplikací je OWASP Mobile Security Project vytvořený neziskovou organizací Open Web Application Security Project (dále jen OWASP), která je známá především svou prací v oblasti zabezpečení webových aplikací. Tento projekt představuje bezplatný a centralizovaný zdroj informací. Klasifikuje bezpečnostní rizika mobilních aplikací a dokumentuje postupy pro bezpečný vývoj, které snižují dopad či pravděpodobnost výskytu zranitelností. Projekt se zaměřuje především na aplikační vrstvu a nepopisuje bezpečnostní rizika samotných mobilních platforem, nicméně aplikační rizika související s danou mobilní platformou jsou brána v úvahu. [9]

Podobně jako OWASP Top 10 pro webové aplikace, identifikuje a kategorizuje OWASP Top 10 Mobile Risks ta nejzávažnější rizika v oblasti bezpečnosti mobilních aplikací. Jednotlivé kategorie rizik lze vidět na obrázku 2.1 a níže jsou stručně popsány. [10]



Obr. 2.1: Rizika podle modelu OWASP.

M1. Nevhodné použití platformy

Nevhodné použití platformy patří mezi největší chyby zabezpečení mobilních zařízení. Uživatel od aplikace očekává to, že je naprogramována tak, aby dodržovala bezpečnostní doporučení a pokyny dané vývojáři systému Android. Aplikace je však mohou ve většině případů neúmyslně porušovat anebo se může jednat o chybu v jejich implementaci.

Nejčastěji se jedná o problémy týkající se nesprávného používání bezpečnostních ovládacích prvků a funkcí, které jsou součástí mobilního operačního systému, například:

- Požadování nadměrných nebo nesprávných oprávnění.
- Nesprávná implementace objektu Intent, který je daný jako veřejný a při běhu aplikace může odhalit citlivé informace nebo dát neoprávněné povolení k jejímu spuštění.

M2. Nezabezpečené úložiště dat

Další na seznamu je ukládání dat do nezabezpečeného úložiště. Této slabiny může útočník využít, pokud je samotné mobilní zařízení odcizeno, ztraceno a nebo je na něm spuštěn malware, který jedná jeho jménem. Tímto způsobem může dojít k úniku osobních údajů a dalších citlivých dat. Jelikož není možné mít aplikace neukládající žádná data, musíme tyto data uložit na bezpečné místo, které nebude přístupné dalším aplikacím nebo uživatelům. [11]

M3. Nezabezpečená komunikace

Mobilní aplikace si běžně data vyměňují v modelu klient-server a tyto data musí být bezpečně přenesena přes síť operátora a internet. Pokud jsou data odesílána nezašifrovaně, může je kdokoli monitorující síť zachytit a následně zneužít.

M4. Nezabezpečená autentizace

Tato kategorie se zabývá nedostatky v procesu autentizace uživatele vyskytující se buď přímo v samotné mobilní aplikaci, nebo na straně serveru. Pokud nejsou dodrženy bezpečnostní standardy, může docházet k obcházení autentizačních mechanismů a získání přístupu do aplikace. Tyto nedostatky mohou být také zneužity k získání důvěrných dat, jako jsou například finanční informace.

M5. Nedostatečné šifrování

Existují dvě situace, kdy může dojít ke kompromitaci kryptografického zabezpečení aplikace, jejímž důsledkem je odhalení citlivých dat:

- Kryptografický algoritmus používaný pro šifrování a dešifrování je již překonán.
- Samotný kryptografický proces má nedostatky v jeho implementaci.

Je mnoho faktorů, díky kterým může být aplikace nebezpečná z hlediska nedostatečného šifrování:

- Nejsou využívány kryptografické algoritmy zabudované v dané platformě.
- Klíče používané v daných algoritmech nejsou bezpečně uloženy.
- Využívané algoritmy jsou vymyšlené přímo vývojáři dané aplikace nebo jsou zastaralé.

M6. Nezabezpečená autorizace

Každý uživatel vyžaduje jiná práva. Běžnému uživateli stačí základní přístupová práva. Ale například takový správce vyžaduje mnohem více různých oprávnění. Špatně navržená autorizační schémata potom mohou vést k tomu, že běžný klient může získat přístup k informacím nebo funkcím mimo rozsah své úrovně oprávnění.

M7. Kvalita klientského kódu

Vlivem špatně naprogramované aplikace může útočník předat škodlivé vstupy do funkcí, které ji tvoří a sledovat její chování. Takové napadení se může promítnout snížením výpočetního výkonu, zvýšením využití paměti atd.Tyto chyby musí opravit přímo vývojáři dané aplikace a pokud se tak nestane, může to vést k problémům jako jsou:

- Zranitelnost zabezpečené formátů řetězce.
- Přetečení vyrovnávací paměti.
- Integrace s nebezpečnými knihovnami třetích stran.
- Vzdálené spuštění kódu.

Je mnoho aplikací, které spoléhají na knihovny třetích stran, které často obsahují chyby a nejsou dobře testovány. Taková rizika jsou mimo kontrolu vývojáře aplikace a jedinou možností je tyto knihovny nevyužívat. Pokud jsou bezpečnostní zranitelnosti způsobeny chybou v kódu vytvořeného přímo vývojářem, je žádané tyto chyby co nejdříve odhalit a opravit.

M8. Manipulace s kódem

Samotné obchody s aplikacemi někdy obsahují nelegitimní verze mobilních aplikací. Ty jsou útočníkem upravené tak, aby na první pohled vypadaly stejně, a přitom obsahovaly škodlivý kód. Útočníci mohou tyto padělané aplikace znovu podepsat a publikovat je v obchodech s aplikacemi třetích stran. Mohou také využít phisingového útoku, a donutit uživatele si je stáhnout.

M9. Reverzní inženýrství

Jedním z rizik je také zpětná analýza a dekompilace kódu. Pokud se takovým způsobem dostane útočník ke zdrojovému kódu aplikace, není pro něj problém mu porozumět a upravit ho tak, aby obsahoval škodlivé funkce, nebo přenášel nežádoucí reklamy. Jakmile takto aplikaci upraví, může ji znovu zkompilovat a distribuovat dále.[12]

M10. Vedlejší funkcionalita

V některých aplikacích si mohou jejich vývojáři zanechat zadní vrátka, nebo nadbytečné funkce, které mohou později využít. Koncový uživatel není ze svého pohledu schopný takové riziko odhalit. Takový přístup vývojářů vytváří značná bezpečností rizika.

Pokud útočník objeví tyto slabé stránky, může je lehce zneužít bez jakéhokoliv povšimnutí uživatele. Většinou dochází k odhalení těchto slabin pomocí analýzy konfiguračních a binárních souborů dané aplikace.

3 Oprávnění Android

Oprávnění aplikace pomáhají podporovat soukromí uživatelů tím, že chrání přístup k následujícím oblastem:

- **Důvěrným datům**, mezi které patří stav systému a kontaktní informace uživatele.
- **Důvěrným akcím**, ke kterým lze zařadit například připojení ke spárovanému zařízení nebo nahrávání zvuku.

Každá aplikace pro android běží ve vlastním izolovaném prostoru a ve výchozím stavu nemá oprávnění provádět žádné operace, které mohou potenciálně nepříznivě ovlivnit jiné aplikace, operační systém nebo uživatele. Pokud tedy potřebuje používat prostředky nebo informace mimo tento izolovaný prostor, je třeba deklarovat oprávnění pro přístup k těmto zdrojům a nastavit žádosti o tyto oprávnění. Jak správně definovat tyto oprávnění je dáno pracovním postupem pro použití oprávnění. Informace v této kapitole jsou primárně převzaty z oficiálních stránek vývojářů systému Android.[13]

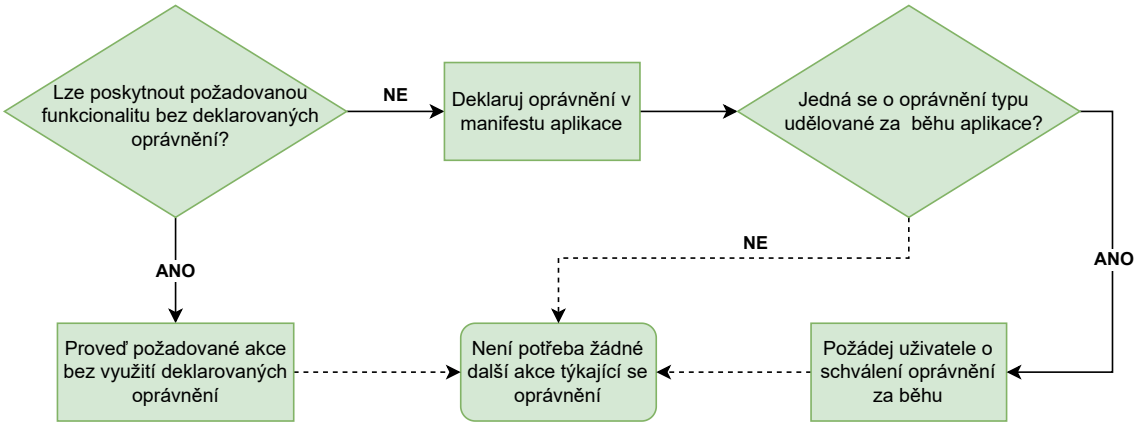
3.1 Postup pro použití oprávnění

Pokud aplikace nabízí funkce, které mohou vyžadovat přístup k důvěrným datům nebo akcím, je nutné určit, zda může informace získat a akce provést, aniž by bylo nutné deklarovat oprávnění. Existuje mnoho případů, kdy není nutné požadovat jakákoliv oprávnění, patří mezi ně fotografování, pozastavení přehrávání médií a zobrazování relevantních reklam.

Pokud ale pro svou plnou funkcionalitu potřebuje získat oprávnění od uživatele, je potřeba je deklarovat. Některá z těchto oprávnění mohou být automaticky přidělena při instalaci aplikace. Ale jiná, známá jako oprávnění za běhu, vyžadují jejich odsouhlasení při běhu aplikace přímo v ní.

3.2 Typy oprávnění

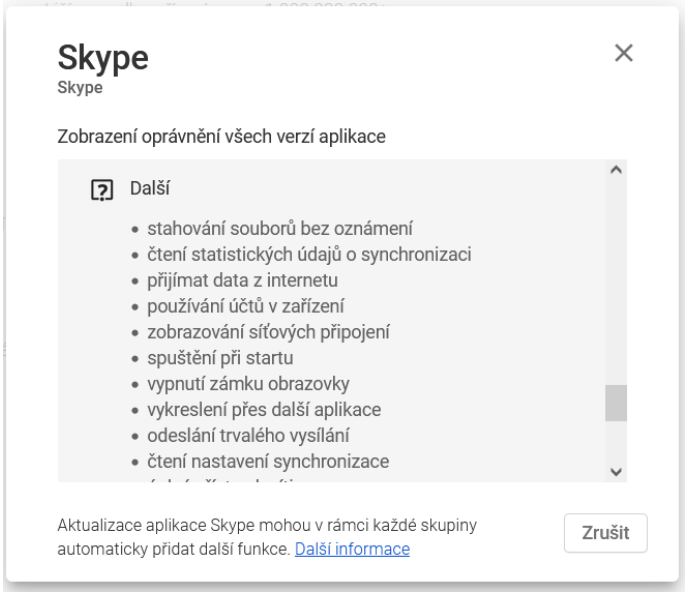
Android definuje tři nejdůležitější skupiny oprávnění, a to oprávnění udělována při instalaci, běhu aplikace a speciální oprávnění. Každá z těchto skupin má omezený rozsah dat, ke kterým může přistupovat a akcí, které může aplikace provádět po jejich udělení.



Obr. 3.1: Postup pro použití oprávnění.

3.2.1 Oprávnění udělována při instalaci aplikace

Tato oprávnění dávají aplikaci omezený přístup k datům a akcím, které mohou bezpečnost uživatele jen minimálně ovlivnit. Pokud jsou v aplikaci deklarovány, systém je automaticky udělí při její instalaci. Uživatel si tato oprávnění může prohlédnout v obchodě s aplikacemi, kde se v podrobnostech vybrané aplikace nachází položka oprávnění aplikace s odkazem „Zobrazit další“. Při jeho rozkliknutí se zobrazí výčet oprávnění a v sekci „Jiné“ udělovaných při instalaci, jak je znázorněno na obrázku 3.2.



Obr. 3.2: Příklad oprávnění udělovaných při instalaci aplikace Skype. [14]

Existuje několik podtypů těchto oprávnění, kde nejpoužívanější jsou normální a podepsaná.

Normální oprávnění

Jsou vyžadována, pokud aplikace potřebuje přistupovat k datům nebo zdrojům mimo její prostor. Tyto oprávnění vytváří velmi malé riziko pro ochranu soukromí uživatele nebo provoz jiných aplikací. Patří sem například oprávnění přístupu do internetu, k blueetoth nebo NFC.

Tyto oprávnění jsou deklarovány v manifestu aplikace a jsou přiděleny při její instalaci. Tudíž systém k jejich odsouhlasení nevyzve a uživatelé nemohou tato oprávnění odebrat.

Podepsaná oprávnění

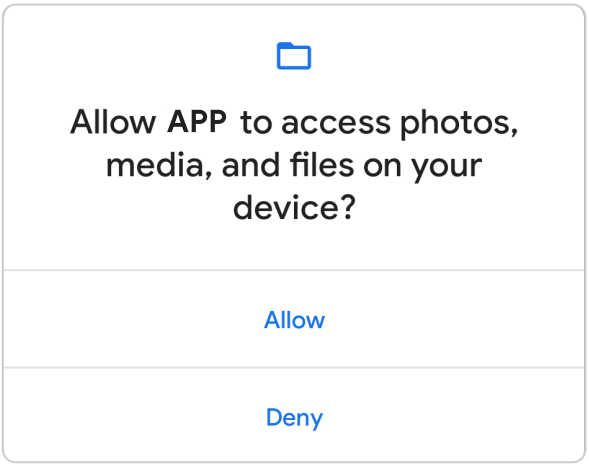
Tato oprávnění jsou také uděleny systémem při instalaci aplikace. K tomu udělení ale dochází jen pokud je aplikace, která se pokouší oprávnění použít, podepsána stejným podpisem, jako aplikace, která je definuje.

3.2.2 Oprávnění udělována při běhu aplikace

Tento typ je také znám jako oprávnění nebezpečná. Poskytují aplikaci přístup k citlivým datům uživatele a umožňují provádět akce, které podstatně ovlivňují systém a další aplikace. Způsob, jakým systém Android tyto oprávnění získá, závisí na jeho verzi:

- **Android 6.0 Marshmallow (API úroveň 23) a vyšší** – pokud jsou tyto oprávnění vyžadována, systém zobrazí za běhu aplikace výzvu, která dává uživateli možnost je povolit a nebo zakázat. Příklad této výzvy je na obrázku 3.3. Dokud nejsou uživatelem schváleny, tak je aplikace nemůže využívat.
- **Android 5.1.1 Lollipop (API úroveň 22)** nebo nižší – systém Android automaticky požádá uživatele o udělení všech deklarovaných nebezpečných oprávnění již při instalaci aplikace do zařízení. Pokud je uživatel přijme, všechna oprávnění jsou udělena v jednom okamžiku. Pokud uživatel žádost o oprávnění zamítne, je instalace aplikace ukončena. Pokud aktualizace aplikace vyžaduje další oprávnění, je uživatel vyzván k přijetí těchto nových oprávnění. [15]

Mezi tyto oprávnění patří například čtení a posílání textových zpráv, přístup k vnitřnímu úložišti nebo poloze zařízení. Všechna tato nebezpečná povolení jsou uvedena v tabulce 3.1.



Obr. 3.3: Příklad dialogu pro udělení oprávnění za běhu aplikace.

3.2.3 Speciální oprávnění

Speciální oprávnění odpovídají konkrétním operacím aplikace. Může je definovat pouze samotná platforma nebo výrobci OEM[1]. Tyto oprávnění se definují za účelem chránit přístup k obzvláště výkonným akcím jako je například zobrazování přes jiné aplikace.

Tyto oprávnění lze nalézt v nastavení zařízení v sekci „Speciální oprávnění“ a každé z nich má své vlastní podrobnosti k implementaci. K těmto oprávněním patří například změny nastavení systému, neomezená data nebo instalování neznámé aplikace.

3.3 Zásady ve vyžadování oprávnění

Oprávnění pomáhají systému Android docílit následujících cílů souvisejících s ochranou uživatele:

- **Kontrola** – uživatel má kontrolu nad daty, které sdílí s aplikacemi.
- **Transparentnost** – uživatel rozumí tomu, jaká data aplikace používá a proč k nim přistupuje.
- **Minimalizace dat** – aplikace přistupuje a používá pouze ta data, která jsou potřebná pro konkrétní úkol nebo akci, kterou uživatel vyžaduje.

Tato část popisuje několik základních doporučených postupů pro efektivní využití oprávnění v aplikacích.

[1]Obchodní termín, který označuje výrobce zařízení, jehož výrobek je prodáván a propagován jinou obchodní značkou.

	Skupina oprávnění	Oprávnění	Popis
1	CALLENDAR	READ_CALENDAR, WRITE_CALENDAR	Čtení v a zápis do kalendáře
2	CALL_LOG	READ_CALL_LOG, WRITE_CALL_LOG, PROCESS_OUTGOING_CALLS	Čtení v a zápis do historie volání
3	CAMERA	CAMERA	Použití fotoaparátu
4	CONTACTS	READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS	Čtení a vytváření kontaktů
5	LOCATION	ACCESS_FINE_LOCATION, ACCES_COARSE_LOCATION	Přístup k lokalizaci zařízení
6	MICROPHONE	RECORD_AUDIO	Nahrávání zvuku
7	PHONE	READ_PHONE_STATE, READ_PHONE_NUMBERS, CALL_PHONE, ANSWER_PHONE_CALLS, ADD_VOICEMAIL, USE_SIP	Čtení stavu telefonu a telefoních čísel Uskutečňování a příjem hovorů
8	SENSORS	BODY_SENSORS	Údaje ze senzorů těla
9	SMS	SEND_SMS, RECEIVE_SMS, READ_SMS, RECEIVE_WAP_PUSH, RECEIVE_MMS	Posílání, čtení SMS a MMS zpráv
10	STORAGE	READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE	Čtení v a zápis do externího úložiště

Tab. 3.1: Oprávnění udělována za běhu aplikace. [16]

Vyžadování minimálního počtu oprávnění

Aplikace by měla vyžadovat pouze oprávnění, které jsou nutné k provedení akce, kterou uživatel zvolí. Vývojář by se měl zamyslet, zda je přístup k citlivým datům nutný a nelze použít alternativní způsob, jak docílit požadované funkcionality.

Vyžadování oprávnění ve správném okamžiku

Oprávnění udělována za běhu by měla být vyžadována až před vykonáním funkce, ke které jsou potřeba. Například pokud aplikace umožňuje odesílání zvukových zpráv ostatním, je ideální, aby se oprávnění pro přístup k mikrofonu požadovalo až při stisknutí tlačítka, které slouží k nahrání této zprávy.

Zvážení použití externích knihoven

U využití externích knihoven si je nutno uvědomit, že mohou od uživatele vyžadovat oprávnění. Od vývojáře by mělo být samozřejmostí ověřit, za jakým účelem knihovny dané oprávnění požadují.

Transparentnost

Každá žádost o oprávnění by měla obsahovat popis s vysvětlením za jakým účelem by měl uživatel dané oprávnění odsouhlasit.

Viditelně informovat o využití oprávnění

Při přístupu k citlivým údajům nebo hardwaru, jako je fotoaparát nebo mikrofon by měla aplikace poskytovat nepřetržitou indikaci, že jsou využívány pokud systém již tyto indikátory neposkytuje. Tato funkcionalita pomáhá uživatelům přesně pochopit, kdy aplikace přistupuje k citlivým datům nebo provádí nebezpečné akce.

3.4 Zneužití oprávnění

Ve většině případů aplikace požadují oprávnění u kterých je jasné, že jsou potřeba. Může se i stát, že některé z nich se na první pohled zdají jako nesmyslné, ale ve skutečnosti jejich požadování má naprosto dobrý důvod.

Naopak existuje taky několik případů zneužití oprávnění aplikace. V této kapitole jsou uvedeny čtyři scénáře, které popisují možnosti jak oprávnění zneužít a poukazují na příklady, které se reálně staly.

3.4.1 Sběr dat

První scénář popisuje případ, který se většině uživatelů vybaví, když po nich aplikace požaduje příliš mnoho oprávnění, které pro její funkčnost nedávají smysl. Taková aplikace s velkou pravděpodobností slouží ke sběru dat o svém uživateli.Tento případ se dá ilustrovat s pomocí známé aplikace pro sociální média Facebook.

Je všeobecně známo, že Facebook data, která sbírá o svých uživatelích využívá pro reklamní účely. Údaje shromážděné touto aplikací se pohybují od osobních údajů uživatele, které platformě poskytl až po historii volání a podrobnostech o zprávách SMS. [18]

Stejně jako jakákoli jiná, musí aplikace Facebook požádat o systémová oprávnění, aby mohla přistupovat k datům a zdrojům mimo aplikační prostor. Pokud se podrobně podíváme na oprávnění, která požaduje starší verze aplikace Facebook Lite, uvidíme mimo jiné tato oprávnění: `READ_SMS` a `READ_CALL_LOG`. První oprávnění umožňuje aplikaci získat podrobnosti o odeslaných a přijatých SMS zprávách a druhé oprávnění umožňuje aplikaci načíst historii hovorů. Zatímco oprávnění `READ_SMS` by mohlo být potenciálně použito k získání vícefaktorových ověřovacích kódů nebo k tomu, aby aplikace mohla fungovat jako chatovací. Tato oprávnění také umožňují aplikaci shromažďovat mnoho osobních údajů.

Pro uživatele dřívějších verzí Androidu bylo velice těžké vyřešit problém s velkým počtem požadovaných oprávnění. Protože jak bylo zmíněno v kapitole 3.2.2, pokud se je uživatel rozhodl neudělit, aplikace se do zařízení ani nenainstalovala a pro její používání bylo tedy nutné udělit všechny požadované oprávnění. V novějších verzích Androidu je už možné takové oprávnění neudělit, ale mnoho uživatelů je slepě udělí kvůli neznalosti důsledků nebo jednoduše z pohodlí. Některé z aplikací také odmítnou vykonávat jakoukoliv funkci bez daných oprávnění.

Kromě výše uvedeného, Facebook využívá také alternativní způsob sběru uživatelských dat, a to jiné aplikace. Existuje mnoho aplikací sdílejících data s Facebookem. Pokud jim uživatel udělí nebezpečné povolení, jeho data mohou být s Facebookem sdílena, i když související oprávnění v aplikaci Facebook byla nejprve pečlivě zakázána. Jak ukazuje zpráva společnosti Privacy International, mnoho aplikací sdílí data, která mají o uživatelích s Facebookem pomocí Facebook SDK.

Shromažďování citlivých dat pro reklamní účely lze považovat za zneužití oprávnění danou aplikací. Lze také tvrdit, že shromažďování takového množství dat i za legitimním účelem vytváří značnou hrozbu pro soukromí uživatele a dokonce lze pomocí nich ovlivňovat i celé komunity. Skvělým příkladem tohoto nebezpečí je skandál Facebook-Cambridge Analytica, kdy unikla citlivá data desítek milionů uživatelů a údajně byla použita k ovlivnění voleb. [19]

Existuje mnoho dalších aplikací typu adware, které fungují podobným způsobem, ale s velkou pravděpodobností s nějakým nekalým úmyslem. Obvykle vyžadují velké množství oprávnění a díky nim shromažďují informace o uživateli. Tyto data poté sdílejí s reklamními sítěmi za účelem zobrazování vysoce cílených reklam. [17]

3.4.2 Aplikace podobné malwaru

Ve druhém scénáři bude popsán příklad, kdy aplikace vyžaduje udělení mnoha oprávnění za účelem zneužití dat uživatele nebo přímo jeho zařízení.

Dobrým příkladem je aplikace využívající malware Joker [20]. Joker přihlásí uživatele k odběru placených služeb bez vědomí uživatele díky odsouhlasení nebezpečných oprávnění. Využívá k tomu oprávnění `READ_PHONE_STATE`, díky kterému získá telefonní číslo uživatele a zneužije ho ke koupení předplatného placených služeb. Jelikož tyto služby obvykle vyžadují potvrzovací kód zaslaný pomocí SMS zprávy, vyžádá si malware také oprávnění `READ_SMS`, které slouží ke čtení SMS zpráv. Poté není problém koupení předplatného potvrdit bez vědomí uživatele.

Dalším příkladem zneužití těchto oprávnění může být aplikace, která požaduje oprávnění `SEND_SMS` pro odesílání SMS zpráv na prémiová čísla nebo aplikace přistupující na SD kartu, která zneužije soubory na ní uložené. Existují také aplikace podobné malwaru, které požadují velmi omezená oprávnění a spoléhají se, že pomocí nich zneužijí jiné legitimní aplikace, které disponují rozsáhlejšími možnostmi přístupu k citlivým datům a akcím.

Pro uživatele, které si aplikace stahují přímo z obchodu Google Play[2] nejsou aplikace popsané výše, velkým nebezpečím [21]. Google takové aplikace v mnoha případech ani nedovolí přidat do svého obchodu a pokud se tak stane, dojde při odhalení nekalých záměrů k jejich rychlému odstranění. Proto jsou největším rizikem pro uživatele aplikace stažené z neznámých zdrojů, například na pochybných webových stránkách.[17]

3.4.3 Zneužití legitimních oprávnění

Třetí scénář popisuje aplikace, které požadují legitimní oprávnění, tedy takové, které dávají uživatelovi smysl pro jejich správnou funkčnost. Ale taktéž při chodu aplikace dochází k jejich zneužití.

Příkladem takové aplikace může být aplikace sloužící ke komunikaci, která vyžaduje oprávnění ke čtení a odesílání SMS. Tato oprávnění poté zneužívá k odesílání SMS na prémiová čísla nebo k zachycení tokenů pro vícefaktorovou autentizaci. Tato oprávnění se pro danou aplikaci zdají legitimní a uživatel neočekává, že by mohlo dojít k jejich zneužití. Taková aplikace poté může lehce uživatele přimět platit za služby, které si nepředplatil a získat například data pro přístup k jeho účtům.

Jedinou možností je aplikaci přestat používat a odinstalovat ze svého telefonu. V praxi takové aplikace nejsou tak běžné, většina aplikací požaduje nelegitimní oprávnění. Útočníkovi se to taktéž nevyplatí, jelikož by musel vynaložit na vytvoření zdání legitimní aplikace značné úsilí. [17]

3.4.4 Zneužití oprávnění jiných aplikací

Posledním příkladem zneužití oprávnění je využívání aplikace, která při svém běhu zneužije oprávnění udělené legitimní aplikaci. Využije toho, že legitimní aplikace využívá nechráněné funkce.

Pokud některá z legitimních aplikací požaduje nebezpečné oprávnění a odhalí systému funkci, která je využívá. Umožní jakékoliv jiné aplikaci nainstalované v zařízení využívat těchto oprávnění legitimní aplikace, bez toho, aby o ně musela škodlivá aplikace žádat.

[2]Oficiální obchod s aplikacemi pro OS Android

Například je možné uvést aplikaci průzkumník souborů, která má udělené oprávnění číst soubory v místním úložišti. Pokud tato aplikace systému odhalí funkci, která slouží pro čtení v úložišti, umožní tím využívat tuto funkci i ostatním aplikacím i když nemají oprávnění `READ_EXTERNAL_STORAGE`. Daný princip zneužití je znám jako Problém zmateného zástupce [23].

Praktickým příkladem toho problému byly aplikace Google a Samsung Fotoaparát, které byly v roce 2019 označeny za zranitelné vůči takovému typu zneužití [22]. Aplikace obsahovali nechráněnou funkci, která jiné aplikaci umožňovala pořizovat snímky nebo videa prostřednictvím aplikace Fotoaparát. Tyto snímky byly zapsány na SD kartu, což pro tyto aplikace je typické. Škodlivá aplikace mohla zažádat o oprávnění pro přístup k SD kartě, což není samo o sobě podezřelé, a poslat Intent do zranitelné aplikace a dané fotografie extrahovat. Při povolení oprávnění využívat fotoaparát byl také povolen přístup ke GPS a díky tomu mohla škodlivá aplikace sledovat uživatele, aniž by potřebovala oprávnění `LOCATION`.

Jedinou možností pro uživatele, jak se vyhnout tomuto zneužití oprávnění, je důvěra ve vývojářské týmy legitimních aplikací, které musí dbát na správnou a bezpečnou implementaci funkcí, aby nevznikaly nechráněné funkce. [17]

Část II

Praktická část

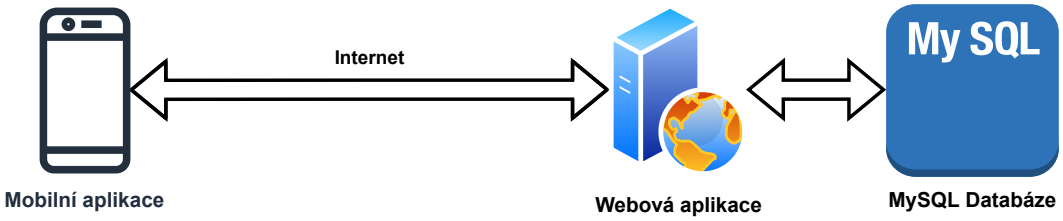
4 Aplikace Rizika bezpečnostních oprávnění

Tato část diplomové práce popisuje strukturu, vzhled a nejdůležitější části vytvořené aplikace, které jsou popsány pohledem programátora a běžného uživatele.

Aplikace, jako celek je tvořena ze tří částí:

- **Mobilní aplikace** – naprogramována s pomocí jazyka Kotlin ve vývojovém prostředí Android Studio. Jazyk Kotlin,[1] je oficiálním jazykem pro vytváření aplikací pro Android a vývojové prostředí (dále jen IDE) Android Studio nabízí mnoho funkcí pro jejich efektivní programování.
- **Webová aplikace** – využívá jazyk Hypertext Preprocessor (dále jen PHP) zejména pro funkční části aplikace a Hypertext Markup Language (dále jen HTML) pro zobrazování dat a interakci s uživatelem.
- **Databáze** – jedná se o MySQL databázi ve formátu InnoDB.

Z důvodu, že zdrojový kód obsahuje několik stovek řádků kódu, nejsou v práci uvedeny zdrojové kódy metod a funkcí, ale spíše vysvětlen jejich princip.



Obr. 4.1: Hlavní části aplikace.

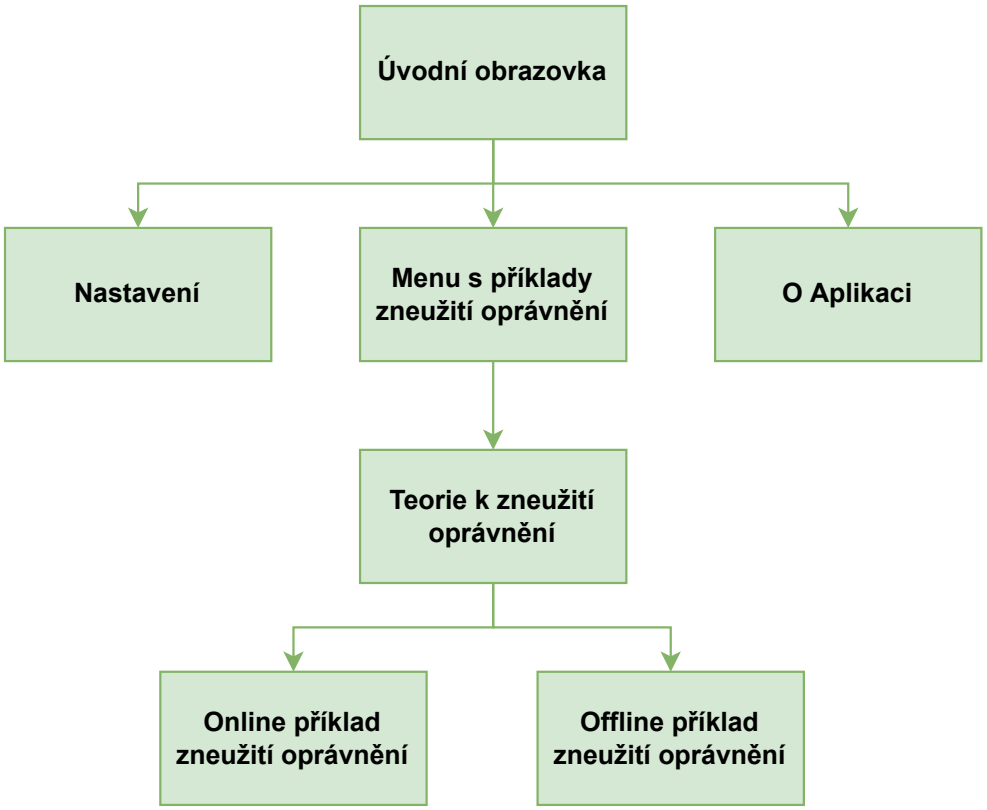
4.1 Mobilní aplikace

Vytvořená aplikace nese jméno „Rizika bezpečnostních oprávnění“. Cílem aplikace je představit rizika udělování bezpečnostních oprávnění pro běžného uživatele. V této verzi aplikace je představeno nebezpečí udělení většiny rizikových oprávnění. Uživateli je nejdříve představeno, co obnáší povolení tohoto oprávnění a poté na praktickém příkladu demonstrováno jeho zneužití. To spočívá v tom, že aplikace přečte citlivá data ze zařízení a odešle je na vlastní server. V zařízení je poté s využitím webové aplikace ukázáno, že se data nachází mimo samotné zařízení a potenciální útočník by je mohl zneužít k získání citlivých údajů uživatele. Praktický příklad může být zobrazen i v režimu offline, kdy jsou uživateli ukázána data, která by byla odeslána z jeho zařízení, pokud by byl nastavený server dostupný. V aplikaci si uživatel může pomocí nastavení také měnit server, na který se budou citlivá data odesílat a jazyk aplikace.

[1] Oficiální stránky jazyku Kotlin: https://kotlinlang.org/.

4.1.1 Struktura

Aplikace využívá architekturu Single-Activity. To znamená, že běh celé aplikace zabezpečuje jedna aktivita a veškeré obrazovky, se kterými uživatel interaguje, jsou tvořeny jednotlivými fragmenty, které aktivita hostuje. Každá z obrazovek slouží k jedné určité funkci. Tato architektura zajišťuje větší stabilitu aplikace, jelikož využívá instanci pouze jedné aktivity. Dále přináší lepší možnosti pro sdílení dat mezi jednotlivými částmi aplikace. Na obrázku 4.2 lze vidět logickou strukturu vytvořené aplikace a návaznost jednotlivých obrazovek na sebe.



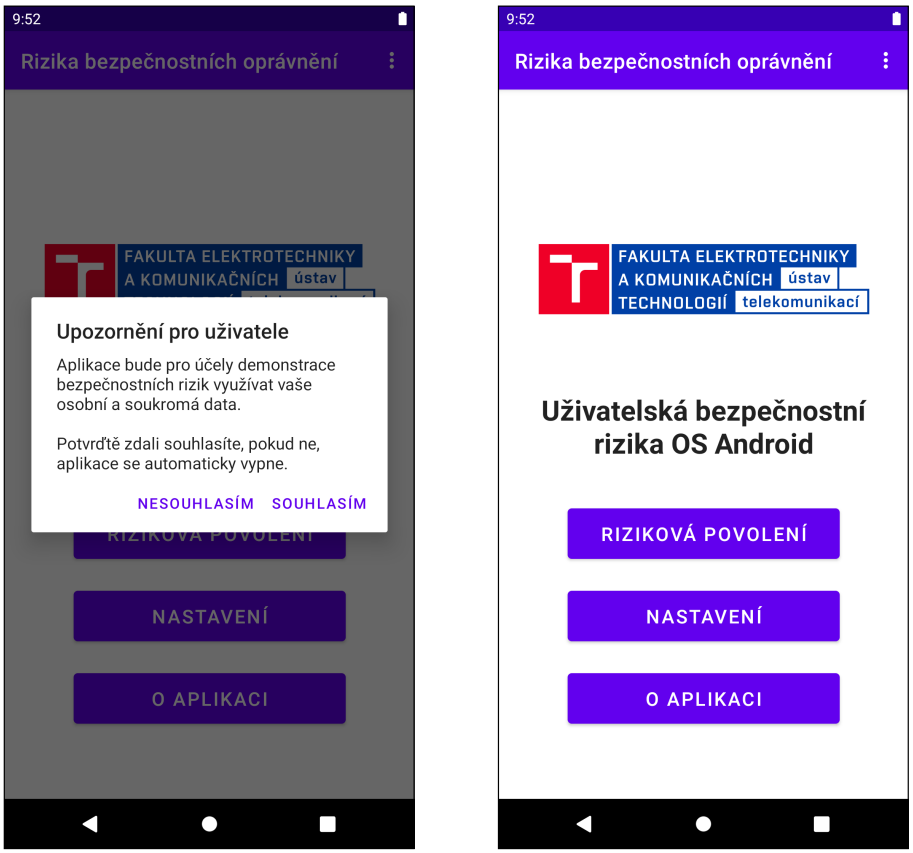
Obr. 4.2: Struktura mobilní aplikace.

4.1.2 Úvodní obrazovka

Po spuštění aplikace se nejdříve uživateli zobrazí dialog upozorňující na to, že aplikace bude při jejím chodu využívat citlivá data ze zařízení uživatele za účelem demonstrace nebezpečí udělování oprávnění. Dialog nabízí výběr ze dvou možností a pokud uživatel:

- Souhlasí – zobrazí se úvodní obrazovka.
- Nesouhlasí – aplikace se ukončí z důvodu nutnosti udělení souhlasu pro její používání.

Úvodní obrazovka obsahuje tři tlačítka, která slouží k dalšímu pohybu v aplikaci. Po jejich stisknutí jsou zobrazeny příslušné obrazovky. Vzhled úvodní obrazovky a dialogu s upozorněním, lze vidět na obrázku 4.3.



Obr. 4.3: Vzhled dialogu s upozorněním a úvodní obrazovky.

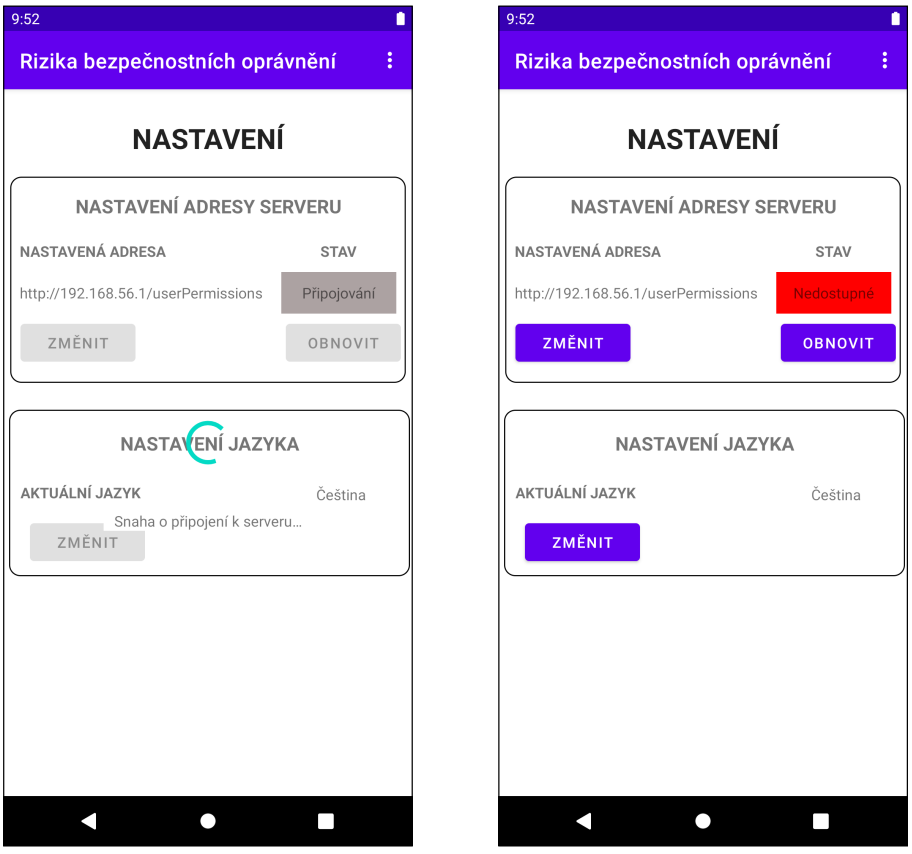
4.1.3 Nastavení

Umožňuje uživateli změnit jazyk aplikace a adresu serveru, na který se budou odesílat data. Do nastavení se lze dostat buď z úvodní obrazovky nebo odkudkoliv z aplikace s pomocí tlačítka NASTAVENÍ, které obsahuje menu umístěné v horní liště aplikace.

Při spuštění je nejprve z úložiště telefonu nahráno poslední uložené nastavení. Poté se otestuje dostupnost aktuálně nastaveného serveru. Aktuální nastavení a i to, zda je server dostupný, či ne, se zobrazí uživateli.

Obrazovka pro nastavení je rozdělena do dvou celků. První z nich slouží k nastavení adresy serveru. Zobrazuje aktuálně nastavenou adresu a její dostupnost. Dále obsahuje tlačítko ZMĚNIT pro otevření dialogu umožňující nastavení nové adresy a tlačítko OBNOVIT s pomocí kterého lze kdykoliv otestovat dostupnost aktuální adresy.

Druhý slouží pro nastavení jazyka aplikace. Zobrazuje uživateli aktuálně nastavený jazyk. Pro jeho změnu slouží tlačítko ZMĚNIT, které zobrazí dialog pro změnu jazyka.

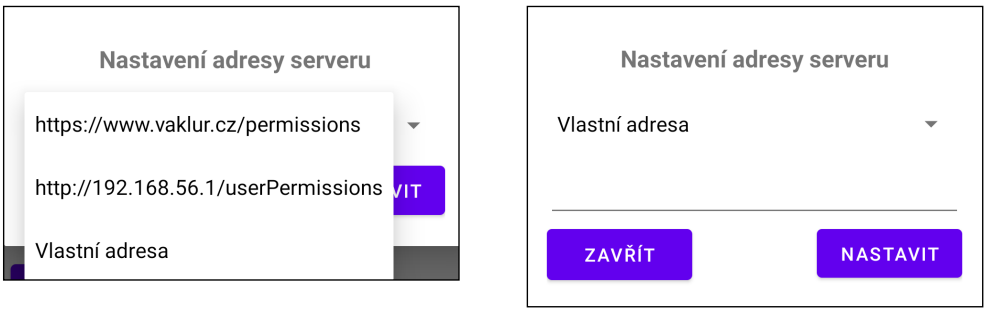


Obr. 4.4: Vzhled obrazovky nastavení.

Dialog pro změnu adresy serveru

Slouží k nastavení již dříve používané nebo zcela nové adresy serveru. K výběru je využit rozbalovací seznam, který obsahuje výčet všech adres, které již byly v aplikaci použity a položku „Vlastní adresa“. Pokud je zvolena vlastní adresa, zobrazí se editační pole pro její zadání.

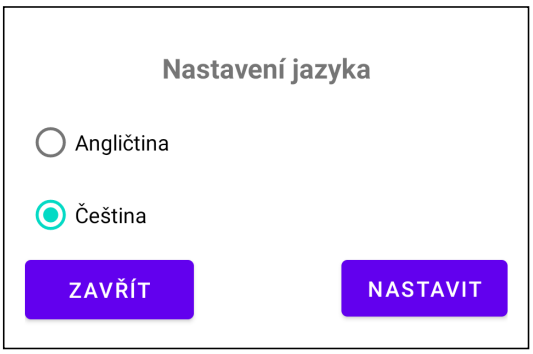
Pro konečné nastavení je nutné stisknout dialogové tlačítko NASTAVIT. Po stisknutí tlačítka se otestuje dostupnost zvolené adresy. Pokud je dostupná, uloží se do nastavení jako aktuální a bude použita pro komunikaci aplikace se serverem. Jestliže ne, v nastavení zůstane jako aktuální adresa ta, která byla jako poslední dostupná.



Obr. 4.5: Vzhled dialogu pro nastavení adresy serveru.

Dialog pro změnu jazyka

Jednoduchý dialog, ve kterém je v aktuální verzi na výběr ze dvou jazyků – angličtina a čeština. Ty lze zvolit s pomocí přepínačů. Pro nastavení je nutné stisknout tlačítko NASTAVIT. Po jeho stisknutí se zkontroluje, zda uživatel nezvolil již nastavený jazyk. V takovém případě dojde pouze k ukončení dialogu. Jestliže je zvolen jazyk jiný než aktuální, dojde k překladu celé aplikace a zavření dialogu.



Obr. 4.6: Vzhled dialogu pro nastavení jazyka aplikace.

4.1.4 Seznam s příklady zneužití oprávnění

Seznam obsahuje výčet všech rizikových oprávnění, ke kterým jsou v aplikaci vytvořeny praktické příklady. Tyto oprávnění lze vidět přímo na obrázku 4.7. Po stisknutí tlačítka s příslušným oprávněním je nejdříve zobrazena teorie, která se ho týká, a ze které se dá dále dostat k samotnému praktickému příkladu.

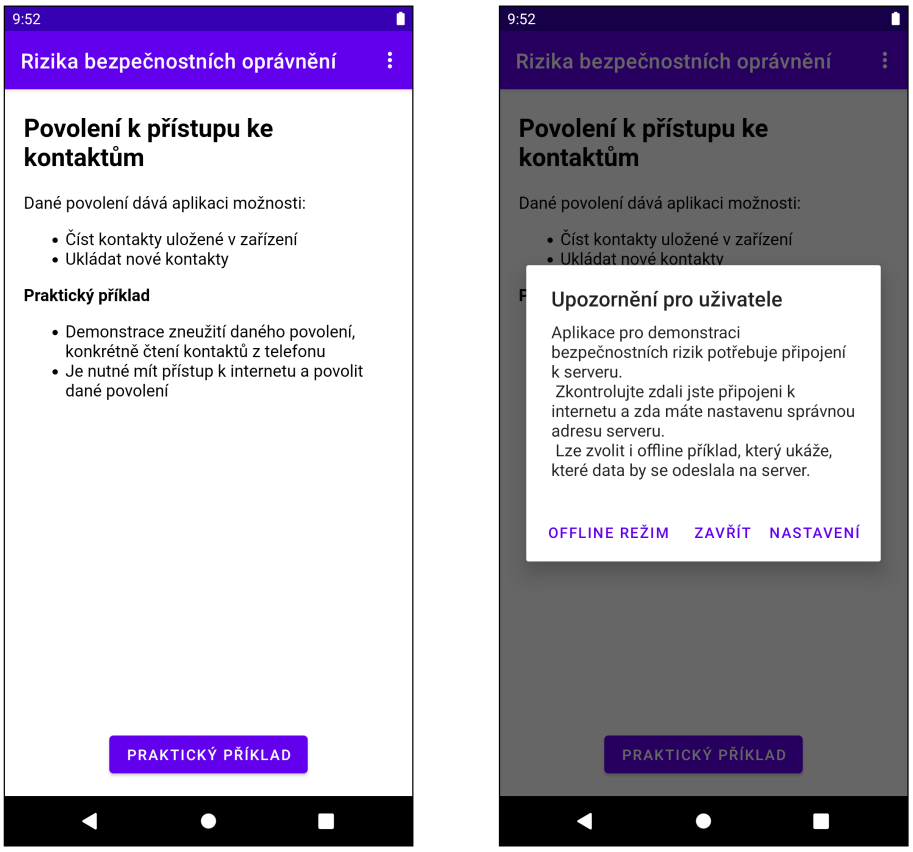
Obr. 4.7: Vzhled seznamu s příklady zneužití.

4.1.5 Teorie k příkladu zneužití oprávnění

Zajišťuje zobrazení teorie k danému oprávnění, které představuje, ke kterým datům se lze s daným oprávněním dostat. Ve spodní části se nachází tlačítko PRAKTICKÝ PŘÍKLAD, které slouží ke kontrole udělení příslušného oprávnění, získání a odeslání citlivých dat na server a pro zobrazení praktického příkladu zneužití oprávnění.

Pokud se nepodaří odeslat data na server, je uživateli zobrazen dialog, který lze vidět na obrázku 4.8. Ten uživatele o nedostupnosti serveru informuje a nabízí mu pomocí tlačítek tři možnosti. Po stisknutí tlačítka:

- ZAVŘÍT – se dialog zavře.
- OFFLINE PŘÍKLAD – se uživateli zobrazí offline praktický příklad, ve kterém jsou zobrazena data, která by byla s daným oprávněním z aplikace odeslána.
- NASTAVENÍ – otevře se nastavení aplikace, kde si uživatel může zvolit adresu serveru, která bude dostupná.



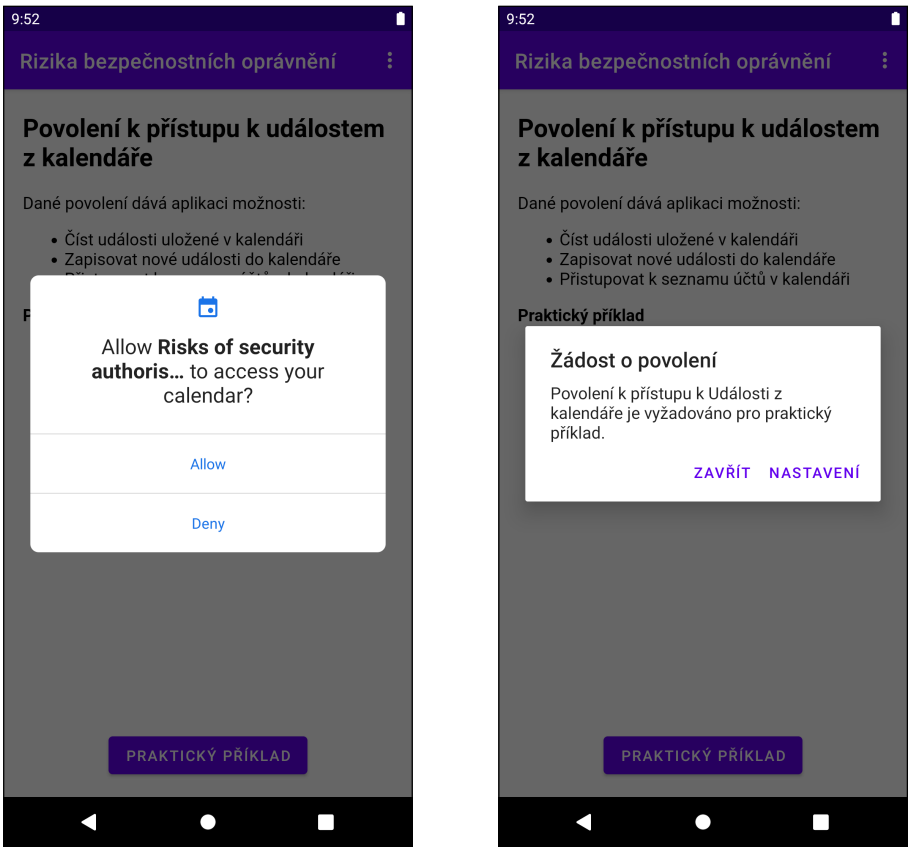
Obr. 4.8: Vzhled teorie a dialogu s upozorněním na nedostupnost serveru.

Kontrola udělení oprávnění

Jelikož je udělení oprávnění nutné pro odeslání dat a zobrazení praktického příkladu, dojde při stisknutí tlačítka ke kontrole, zda je oprávnění uděleno. Mohou nastat dvě možnosti:

- Oprávnění uděleno – ze zařízení jsou získána citlivá data, která jsou následně odeslána na server.
- Oprávnění neuděleno – mohou nastat dvě možnosti v závislosti na tom, zda již bylo dané oprávnění aplikací dříve vyžadováno.
  - Pokud je vyžadováno poprvé, zobrazí se systémový dialog s žádostí o udělení daného oprávnění.
  - Pokud již bylo dříve zamítnuto, zobrazí se dialog upozorňující uživatele, že je nutné oprávnění udělit, a to přímo v nastavení zařízení. Pro usnadnění je v dialogu tlačítko NASTAVENÍ, které uživatele přesměruje do podrobností o aplikaci. Tam lze ručně oprávnění povolit.

Proces, kterým je oprávnění vyžadováno je znázorněn na diagramu v příloze A.1.



Obr. 4.9: Dialogy pro udělení oprávnění uživatelem.

Získání citlivých dat ze zařízení

Po udělení příslušného oprávnění jsou ze zařízení v závislosti na jeho druhu získána citlivá data, mezi které patří:

- SMS zprávy – obsahují telefonní číslo adresáta/příjemce, datum a čas odeslání/příjmu, samotný text zprávy a typ. Typ zprávy nese informaci o tom, zda je SMS zpráva přijatá nebo odeslaná.
- Kontakty – každý kontakt obsahuje telefonní číslo a k němu patřící jméno.
- Záznamy hovorů – obsahují telefonní číslo, datum a čas uskutečnění hovorů, délku hovoru a typ záznamu. Typ značí, zda byl hovor přijatý, odchozí nebo zmeškaný.
- Události z kalendáře – událost obsahuje její název, datum a čas jejího začátku a konce a popis, který k ní vytvořil její zakladatel.
- Poslední známá poloha – informace o poloze obsahují souřadnice, nadmořskou výšku a přesnost GPS.
- Data z externího úložiště – fotky a obrázky, které jsou uložené přímo z v galerii uživatele v zařízení.
- Údaje z účastnické identifikační (dále jen SIM) karty – jedná se o telefonní čísla registrovaná v zařízení, aktuální typ datové sítě, ke které je telefon připojen a kód mobilního operátora.
- Fotky z fotoaparátu – fotka získána s využitím předního fotoaparátu zařízení bez vědomosti uživatele aplikace.

Odesílání dat na server

Data, získaná díky udělení oprávnění jsou z aplikace odeslána na server. Tento proces se skládá z několika kroků:

1. Otestování dostupnosti serveru – Zkontroluje se, zda je adresa aktuálně nastaveného serveru dostupná.
2. Vytvoření uživatele – V databázi na serveru se vytvoří profil uživatele, který slouží pro přihlášení do webové aplikace.
3. Přidání typu oprávnění k uživateli – Do profilu uživatele v serverové databázi se přidá typ aktuálně zvoleného oprávnění.
4. Odeslání citlivých dat do databáze – Do příslušné tabulky v databázi v závislosti na oprávnění jsou poslána citlivá data ze zařízení.

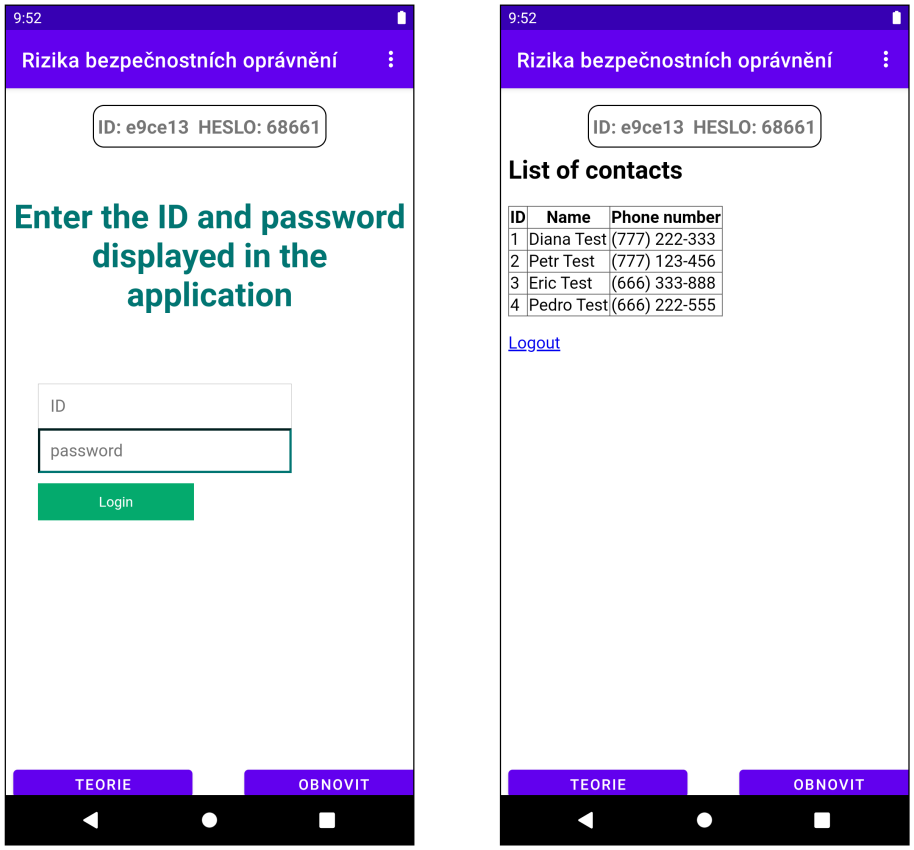
Tento proces je znázornění na diagramu A.2.

K odesílání dat z aplikace na server jsou použity funkce z vlastní třídy `CommunicationFunction`. Pro samotnou komunikaci využívají tyto funkce knihovnu Volley. Jedná se o oficiální Hypertext Transfer Protocol Secure (dále jen HTTPS) knihovnu vyvíjenou společností Google, která slouží k rychlé a snadné komunikaci přes internet v systému Android. Třída obsahuje dvě skupiny funkcí v závislosti na jejich účelu:

- Pro správu uživatelů (přidávání/odstraňování) a odesílání dat do databáze.
- Pro testování dostupnosti serveru.

4.1.6 Praktický příklad zneužití oprávnění

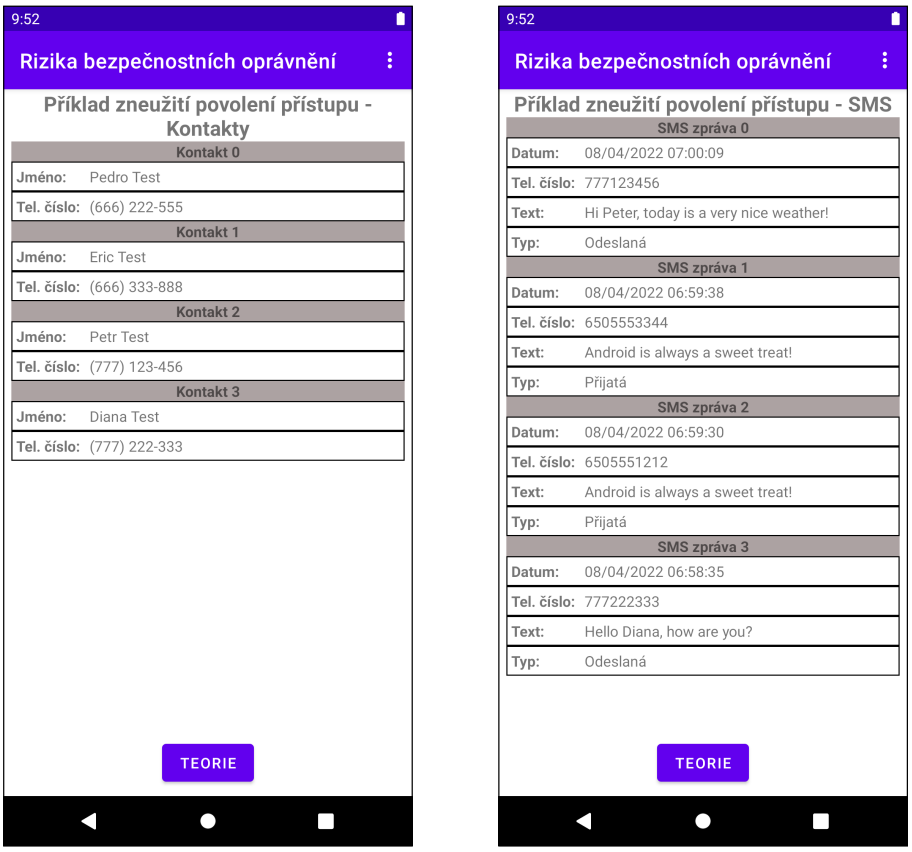
Slouží k zobrazení praktického příkladu zneužití zvoleného oprávnění. Při jeho vytvoření je nejdříve přečteno ID telefonu, ze kterého je prvních šest znaků použito jako unikátní ID a dalších pět znaků jako unikátní heslo pro daného uživatele. ID a heslo jsou zobrazeny uživateli v horní části obrazovky. Pod těmito údaji se nachází webové okno s webovou aplikací, viz kapitola 4.2, která slouží pro zobrazení citlivých dat uživatele na serveru. Pod webovým oknem se nachází tlačítko TEORIE, pomocí kterého se lze vrátit zpět k teorii k danému oprávnění. Vedle něj je umístěno tlačítko OBNOVIT, které slouží k znovunačtení webové aplikace.



Obr. 4.10: Vzhled praktického příkladu s webovou aplikací.

4.1.7 Offline praktický příklad zneužití oprávnění

V závislosti na druhu oprávnění, kterého se příklad týká, zobrazuje citlivá data, která by byla v případě dostupnosti serveru na něj odeslána. Pokud by žádná data odeslána nebyla, je uživatel upozorněn, že v zařízení se taková data nenacházejí. Vzhled offline praktického příkladu pro oprávnění přístupu k SMS a kontaktům lze vidět na obrázku 4.11.



Obr. 4.11: Vzhled obrazovky offline praktického příkladu.

4.2 Webová aplikace

Zajišťuje zobrazení citlivých dat uživatele, která byla získána z jeho mobilního zařízení s pomocí mobilní aplikace. Též obstarává příjem těchto dat a správu uživatelských účtu vztahujících se k jednotlivým zařízením. Aplikace je napsána v jazycích PHP a HTML. V PHP jsou napsány všechny funkční části, které zabezpečují příjem dat, jejich následné dešifrování a komunikaci s databází. HTML je využito pouze pro zobrazení dat ve webovém okně.

Struktura webové aplikace s popisem účelu jednotlivých souborů je zobrazena na obrázku B.1. Z něj je patrné, že v každé složce se nachází soubor .htaccess, ten slouží k omezení přístupu k souborům v dané složce a umožňuje k nim přistupovat jen pomocí funkcí volaných přímo v kódu aplikace. Účel a další podrobnosti k ostatním souborům aplikace jsou popsány v podkapitolách níže, a to na základě funkce, kterou vykonávají.

Aplikaci lze rozdělit do tří logických celků podle funkce, kterou každý z nich vykonává:

- Příjem dat z mobilní aplikace.
- Správu dat v databázi.
- Zobrazení dat z databáze uživateli.

4.2.1 Příjem dat z mobilní aplikace

Samotný příjem dat z aplikace zabezpečují funkce, které se nachází v souboru `index.php`. Pokud aplikace obdrží zprávu, je nejdříve ověřeno, zda hlavička zprávy obsahuje platnou operaci, například `add_user` pro přidání uživatele do databáze. Všechny definované operace jsou uvedeny v tabulce 4.1. Dále je ověřeno, zda se v těle nachází všechny požadované parametry. Pokud ano, je zpráva považována za platnou a hodnoty parametrů jsou dešifrovány. To zajišťuje třída `RSA`, ve které je definována funkce pro dešifrování dat s pomocí algoritmu RSA[2] a soukromého klíče uloženého v aplikaci. Po dešifrování se zavolají funkce, které slouží pro uložení dat do příslušných tabulek v databázi a do mobilní aplikace je odeslán Hypertext Transfer Protocol (dále jen HTTP) stavový kód 200. Ten informuje odesílatele o úspěšném příjmu dat ze strany aplikace. Pokud je hlavička nebo tělo zprávy neplatné, je odeslán stavový chybový kód 400, který informuje odesílatele o tom, že jeho požadavek je neplatný.

[2]Asymetrická šifra, která je založena na Eulerově větě, a která je použitelná jak pro šifrování, tak pro podepisování dokumentů.

4.2.2 Správa dat v databázi

Správu dat zajišťuje třída `DbOperation`, která je definována v souboru `DbOperation.php`. Obsahuje funkce pro přidávání, odstraňování a aktualizaci dat v databázi. Při zavolání jednotlivých funkcí je nejdříve navázáno spojení s databází, které zabezpečuje třída `DbConnect`, definována v souboru `DbConnect.php`. Pokud je databáze dostupná, je provedena zvolená operace. Pokud se objeví jakýkoliv problém, je o tom uživatel informován prostřednictvím chybového kódu. Všechny definované databázové funkce jsou uvedeny v tabulce 4.1.

Operace	**Databázová funkce**	**Popis**
add_sms	addSms	Přidání SMS zprávy
add_event	addEvent	Přidání události z kalendáře
add_call_log	addCallLog	Přidání záznamu hovoru
add_camera_photo	addCameraPhoto	Přidání názvu fotografie
add_contact	addContact	Přidání telefonního kontaktu
add_location	addLocation	Přidání poslední známé lokace
add_phone_state	addPhoneState	Přidání informací ze SIM karty
add_media_photo	addMediaPhoto	Přidání názvu obrázku
add_user	createUser	Vytvoření profilu uživatele
delete_user	deleteUser	Odstranění uživatele a všech jeho dat
add_permission_type	addPermissionType	Přidání typu oprávnění k uživateli
delete_permission_type	deletePermissionType	Odstranění typu oprávnění uživatele
upload_image	/	Nahrání obrázku/fotky do složky na serveru
server_state	/	Otestování dostupnosti serveru

Tab. 4.1: Operace a databázové funkce definované ve webové aplikaci.

4.2.3 Zobrazení dat uživateli

Jedná se část webové aplikace, která zabezpečuje interakci s uživatelem. Slouží pro zobrazení dat z databáze, které byly ze zařízení využívající mobilní aplikaci odcizeny za účelem demonstrace nebezpečí oprávnění. Tvoří ji tři webové stránky, a to:

- Přihlášení uživatele.
- Zobrazení dat z databáze.
- Odhlášení uživatele.

Přihlášení uživatele

Tato stránka obsahuje přihlašovací formulář, který požaduje od uživatele jeho ID a heslo. Tyto údaje jsou vygenerovány samotnou mobilní aplikací a uloženy v databázi umístěné na serveru. Pro uživatele jsou údaje zobrazeny v praktickém příkladě, viz kapitola 4.1.6.

Při vyplnění a potvrzení přihlašovacích údajů nastane kontrola, zda se uživatel se zadaným ID nachází v databázi. Pokud ano, ověří se, zda hash zadaného hesla odpovídá tomu uloženému v databázi. Při úspěšném ověření jsou tyto údaje uloženy do sessions a dojde k přesměrování na stránku se zobrazením dat z databáze. Sessions jsou využívány z důvodu bezpečnosti, jelikož přihlašovací údaje jsou díky tomu uloženy přímo na serveru a do prohlížeče klienta je odesílán jen jejich identifikátor. Pokud uživatel není v databázi nebo nezná správné heslo, je zobrazena chybová hláška o nesprávnosti přihlašovacích údajů.

Přihlašovací formulář uživatele je zabezpečen proti útoku SQL injection. K tomuto zabezpečení je v aplikaci využito takzvané „Escapování“ znaků, které mají speciální význam v SQL. V samotném kódu je volána funkce `mysql_real_escape_string()`, která ještě před odesláním dotazu do databáze ošetří vstupní proměnné, tedy ID a heslo uživatele.

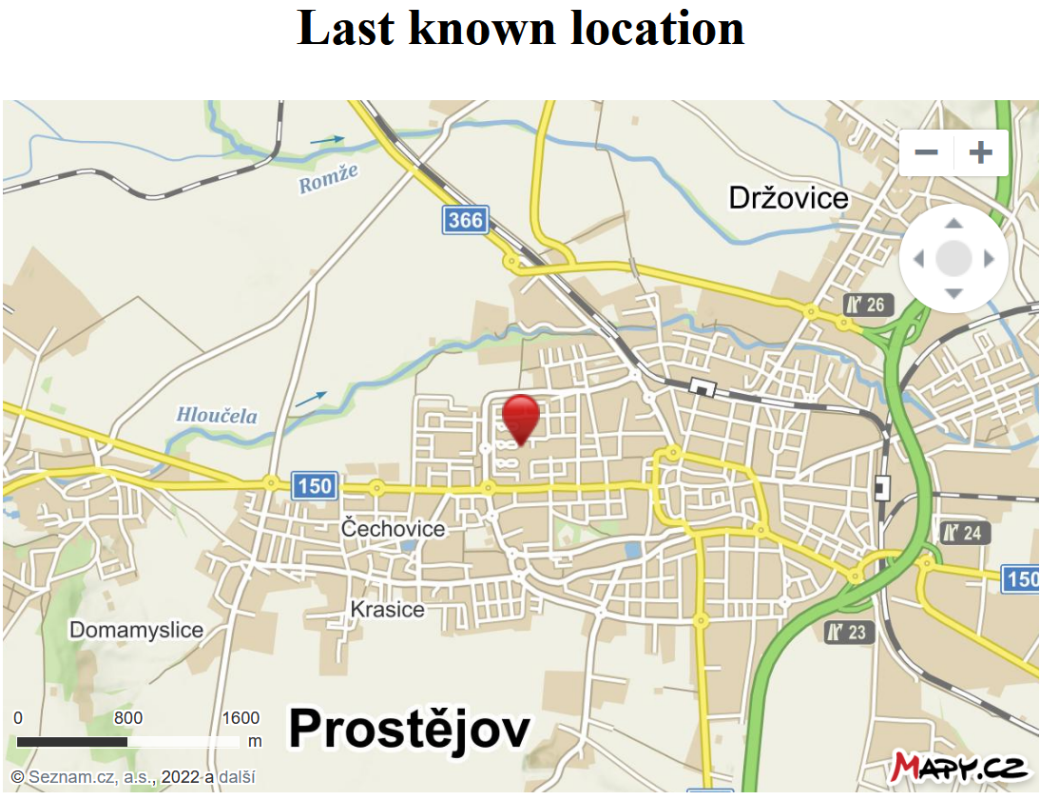
Obr. 4.12: Přihlašovací stránka s chybovou hláškou.

Zobrazení dat z databáze

Po úspěšném přihlášení je na základě přihlašovacích údajů získaných ze sessions vyhledán typ oprávnění, ke kterému se pro daného uživatele nachází v databázi data. Pokud jsou data uložená v databázi ve formě:

- Textových záznamů – Zobrazí se data v přehledné tabulce.
- Cest k obrázkům– Zobrazí se pod sebou jednotlivé obrázky.
- Souřadnic – Zobrazí se mapa s polohou definovanou danými souřadnicemi. Aplikace využívá k zobrazení mapy opensource API od Mapy.cz.

Pod zobrazenými daty se nachází odkaz, pomocí kterého se může uživatel odhlásit.



Obr. 4.13: Zobrazení poslední známé polohy zařízení uživatele.

Odhlášení uživatele

Tato stránka obsahuje pouze PHP skript, který odstraní uživatelovi údaje ze sessions, a poté ho automaticky přesměruje na přihlašovací stránku.

4.3 Databáze

Pro uložení citlivých dat ze zařízení a správu uživatelských účtů je použita MySQL databáze ve formátu InnoDB. Strukturu databáze lze vidět v příloze C.1. Tabulky lze rozdělit dle jejich účelu na:

- Tabulku pro uložení uživatelských účtů.
- Tabulky pro uložení citlivých dat uživatelů.

Tabulka s uživatelskými účty

Tabulka `users` slouží k uložení uživatelských účtů, které jsou pro každé mobilní zařízení unikátní. Obsahuje sloupce pro ID a zahashované heslo uživatele. Další sloupce obsahují informaci o tom, k jakému z nebezpečných oprávnění se v databázi nachází příslušná citlivá data. Na obrázku 4.14 jsou záznamy dvou uživatelů aplikace. První z nich (ID – 79c32c) má v databázi uložené SMS zprávy ze svého zařízení a druhý (ID – d15304) data ze SIM karty.

id	userid	password	event	callLog	camera	contact	location	phoneState	sms	storage
967	79c32c	$2y$10$W /gapy9AD(							sms	
968	d15304	$2y$10$lq						phoneState		

Obr. 4.14: Příklad záznamů z tabulky s uživatelskými účty.

Tabulky s uživatelskými daty

Slouží k uložení citlivých dat, která byla získána z mobilní aplikace. Každá tabulka obsahuje sloupec s ID uživatele, na základě kterého je rozlišeno, ke kterému uživateli příslušné data patří. Zbylé sloupce jsou odlišné. Jejich počet i data, která obsahují, jsou dána oprávněními, ke kterým se vztahují. Tabulku `callLogs`, která obsahuje záznamy hovorů získané ze dvou mobilních zařízení lze vidět na obrázku 4.15.

id	userid	phoneNumber	date	duration	type
50	e9ce13	6505551212	08/04/2022 06:58:51	6	Příchozí
51	e9ce13	6505551212	08/04/2022 06:59:06	0	Neznámý
52	e9ce13	777222333	08/04/2022 06:58:09	5	Odchozí
53	e9ce13	6505551212	08/04/2022 06:59:13	0	Zmeškaný
54	79c32c	+420774981126	25/04/2022 12:41:11	3900	Příchozí

Obr. 4.15: Tabulka obsahující záznamy hovorů uživatelů.

4.4 Testování

Testování z velké části probíhalo bez využití automatických testů, a to především z důvodu vysoké náročnosti pro jejich vytvoření. Jako mnohem efektivnější řešení se jevilo manuální testování systému po částech a až nakonec jako celku. Díky tomu mohly být chyby včas zachyceny a opraveny, což eliminovalo zdlouhavé hledání chyb při závěrečných fázích testování.

Testování probíhalo třemi způsoby, a to s využitím:

- Emulátoru zařízení s OS android a serveru v lokální síti.
- Reálných mobilních zařízení a serveru umístěném na hostingu.
- Předem definovaných testů v prostředí Firebase Test Lab.

4.4.1 Testování v průběhu vývoje

Při vývoji aplikace docházelo k průběžnému testování všech nových a i již implementovaných funkcí.

Mobilní aplikace byla testována s pomocí Android emulátoru, který nabízí IDE Android Studio. Hlavní výhodou emulátoru je možnost testovat aplikaci na různých verzích OS Android, různých rozlišeních displeje a bez nutnosti nahrávání na vlastní mobilní zařízení. Nevýhodou je malý výpočetní výkon a nedostatečná optimalizace. Veškeré bugy a problémy s optimalizací uživatelského rozhraní, které nastaly při testování na emulátoru, byly opraveny. Na konci vývoje tedy aplikace nevykazovala žádné neočekávané chování, byla stabilní a plně funkční.

K testování webové aplikace a komunikace aplikace-server byl využit server umístěný v lokální síti vytvořený pomocí programu XAMPP[3]. Na tomto serveru běžela webová aplikace a MySQL databáze.

Ukládání a zpracování přijatých dat do databáze s pomocí PHP skriptů definovaných ve webové aplikaci bylo nejdříve testováno pomocí programu Postman[4], který umožňuje vytvořit HTTPS požadavky, odesílat je na server a zachytávat jeho odpovědi. Po otestování, že je ze strany serveru vše v pořádku, bylo testování rozšířeno na odesílání požadavků a dat přímo z aplikace a byla doladěna celková funkcionalita.

Webová aplikace byla testována a laděna přímo ve webovém prohlížeči, byla optimalizována a dopracována do plně funkční verze. V rámci testování bylo též ověřeno zabezpečení přihlašovacího formuláře před útokem SQL Injection. K tomuto ověření bylo využito manuálních testů, které jsou navrženy v metodice pro testování zranitelností webových aplikací v projektu OWASP.

[3]Oficiální stránky programu XAMPP: https://www.apachefriends.org/index.html
[4]Oficiální stránky programu Postman: https://www.postman.com/

4.4.2 Testování na reálných zařízeních

Aplikace byla testována na pěti mobilních zařízeních od různých výrobců a s odlišnými hardwarovými parametry. V rámci tohoto testování byla webová aplikace a MySQL databáze umístěna na serveru hostitele. Dané serverové řešení bylo využito, jelikož bylo potřeba, aby aplikace byla dostupná odkudkoliv z internetu. Zařízení i s jejich parametry jsou uvedeny v tabulce 4.2.

Mobilní zařízení	Verze OS	Rozlišení obrazovky [px]	Paměť RAM [GB]
Xiaomi MI9	11	2340 x 1080	6
POCO X3 NFC	11	2400 x 1080	6
Xiaomi Redmi 5 Plus	8.1	2160 x 1080	4
Samsung Galaxy A40	11	2340 x 1080	4
Motorola One Macro	11	2340 x 1080	4

Tab. 4.2: Zařízení, na kterých byla aplikace testována.

Finální verze aplikace na těchto zařízeních fungovala plynule a bez pádů. Komunikace mezi aplikací a serverem byla též funkční a přihlašování do webové aplikace a zobrazování dat nevykazovalo žádné chyby. Z výsledků testování lze aplikaci považovat za plně funkční.

4.4.3 Testování v prostředí Firebase Test Lab

Finální verze aplikace byla též testována pomocí automatických testů v prostředí Firebase Test Lab. Dané prostředí umožňuje vývojáři testovat jeho aplikaci na řadě různých zařízení a jejich konfigurací.

Pro testování byl využit Robo test integrovaný přímo v tomto prostředí. Účelem testu je analyzovat strukturu uživatelského rozhraní aplikace. Tuto strukturu poté metodicky prozkoumávat a simulovat aktivity uživatelů. Výsledkem testování je protokol informující vývojáře o chybách, které při něm nastaly. Dále jsou zobrazeny snímky obrazovky aplikace a video zobrazující průběh provádění testu. Výsledky testu slouží k odhalení hlavních příčin selhání aplikace a mohou pomoci s problémy se špatnou optimalizací uživatelského rozhraní.

Testování vytvořené aplikace s pomocí Robo testu, vždy dopadlo úspěšně. Pro test byly zvoleny různé druhy virtuálních i fyzických zařízení dostupných v prostředí Firebase. Zařízení se také lišila verzí OS Android a minimální verze, na které test probíhal, byla verze 8.1. Výsledky jednoho z testů lze vidět v příloze D.1.

4.5 Možnosti rozšíření

Aplikace lze rozšířit ještě o mnoho funkcí a nabízí také možnosti vylepšení jejího designu, ať už z pohledu komfortu uživatele nebo i z modernějšího a stylového designu. V následujících podkapitolách jsou popsány ty nejzajímavější z nich.

4.5.1 Rozšíření praktických příkladů o další oprávnění

První možností by bylo rozšíření praktických příkladů zneužití oprávnění o všechna nebezpečná oprávnění, která jsou v systému Android definována. Mezi ta, která nebyla v aplikaci využita, patří oprávnění přístupu k bluetooth[5], mikrofonu a senzorům sledující životní funkce uživatele mobilního zařízení.

4.5.2 Zabezpečení zpráv

Aktuální verze aplikace využívá pro zabezpečení komunikace na transportní vrstvě protokol HTTPS a aplikační vrstvě algoritmus RSA, kdy jednotlivé zařízení vlastní veřejný klíč a na serveru je umístěn klíč soukromý. Tento způsob je funkční, ale nepatří mezi best-practice pro takový druh komunikace.

Zabezpečení na aplikační vrstvě by chtělo rozšířit o model, u kterého by bylo asymetrické šifrování pomocí RSA využíváno jen k distribuci klíče, například pro algoritmus AES[6].Šifrování dat by poté bylo prováděno jen s využitím tohoto algoritmu. Díky tomu, by se značně zmenšila velikost přenášených dat mezi aplikací a serverem.

[5]Standard pro bezdrátovou komunikaci propojující dvě a více elektronických zařízení.
[6]Standardizovaný algoritmus využívající symetrickou blokovou šifru.

Závěr

Cílem této práce bylo vytvořit mobilní aplikaci pro demonstraci reálných bezpečnostních rizik vznikajících využíváním operačního systému Android běžným uživatelem. Výsledkem této práce je mobilní aplikace s názvem „Rizika bezpečnostních oprávnění“. Tato aplikace slouží k ukázce bezpečnostních rizik, která vznikají při udělování nebezpečných oprávnění aplikacím v OS android.

Tato rizika jsou demonstrována na vytvořených praktických příkladech, které se skládají z teoretické části a samotného příkladu. V teoretické části je uživateli vysvětleno, jaké akce může aplikace s daným oprávněním provádět. Pro přechod na příklad, je poté oprávnění vyžadováno. Po jeho povolení jsou ze zařízení, na kterém aplikace běží, odeslána na server citlivá data. S pomocí webové aplikace, která je zobrazena v mobilním zařízení v rámci praktického příkladu, je demonstrováno, že se daná data nachází mimo samotné zařízení uživatele a mohla by být dále zneužita.

Tento praktický příklad tedy slouží k tomu, aby si běžný uživatel mobilního zařízení s OS Android uvědomil, že udělování nebezpečných oprávnění aplikacím, může vést k odcizení a následnému zneužití jeho citlivých dat.

Aplikace byla důkladně otestována na reálných zařízeních v provozu a z výsledků testování se jeví jako plně funkční.

Využití aplikace vidím například ve výuce, kdy lze využít jako prostředek pro praktickou demonstraci zranitelností OS Android, ať už pro běžné uživatele, bezpečnostní inženýry nebo programátory mobilních aplikací.

Literatura

[1] RAJNET, Ondřej. Android. Netos.cz [online]. 2014, 2014 [cit. 2021-11-16]. Dostupné z: http://or.jnespor.cz/linux_mob.html

[2] LACKO, Luboslav. Vývoj aplikací pro Android. Dotisk 1. vydání. Brno: Computer Press, 2015. ISBN 978-80-251-4347-6.

[3] Distribution dashboard. Developers [online]. 2021, 30.7.2021 [cit. 2021-11-16]. Dostupné z: https://developer.android.com/about/dashboards

[4] Android Architecture. Tutlane [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://www.tutlane.com/tutorial/android/android-architecture

[5] Android Application Components. Techplayon [online]. 2017 [cit. 2021-11-16]. Dostupné z: https://www.techplayon.com/applications-component/

[6] FRANK, Jiří. Lekce 8 - Android programování - Životní cyklus aktivity [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://www.itnetwork.cz/java/android/zaklady/tutorial-programovani-pro-android-v-jave-zivotni-cyklus-a-novy-projekt

[7] KONEČNÝ, Matěj. Vyvíjíme pro Android: Dialogy a activity. Zdroják.cz [online]. 2012 [cit. 2021-11-16]. Dostupné z: https://zdrojak.cz/clanky/vyvijime-pro-android-dialogy-a-activity/

[8] Layouts. Developers [online]. 2021, 27.10.2021 [cit. 2021-11-16]. Dostupné z: https://developer.android.com/guide/topics/ui/declaring-layout

[9] BLÁHA, Lukáš. Bezpečnost mobilních aplikací. Aec [online]. 2015 [cit. 2021-11-16]. Dostupné z: https://www.aec.cz/cz/ztisku/lukas-blaha-bezpecnost-mobilnich-aplikaci-dsm-2015.pdf

[10] MUKHERJEE, Lumena. OWASP Mobile Top 10. Infosec insight [online]. 2020 [cit. 2021-11-16]. Dostupné z: https://sectigostore.com/blog/owasp-mobile-top-10/

[11] Mobile Top 10 Security Risks. Guardsquare [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://www.guardsquare.com/blog/owasp-mobile-top-10-security-risks-for-app-developers

[12] BASATWAR, Govindraj. Mobile App Security. Appsealing [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://www.appsealing.com/mobile-app-security-a-comprehensive-guide-to-secure-your-apps/

[13] Permissions on Android. Developers [online]. 2021, 11.11.2021 [cit. 2021-11-16]. Dostupné z: https://developer.android.com/guide/topics/permissions/overview

[14] Skype. Google Play [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://play.google.com/store/apps/details?id=com.skype.raider

[15] Lekce 1 - Android - Úvod do oprávnění aplikací. Itnetwork.cz [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://www.itnetwork.cz/java/android/opravneni/android-uvod-do-opravneni-aplikaci

[16] Android Runtime Permissions Tutorial and Example. Camposha [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://camposha.info/android-examples/android-runtime-permissions/#gsc.tab=0

[17] LARDINOIS, Simon. How malicious applications abuse Android permissions. NVISIO labs [online]. 2021 [cit. 2021-11-16]. Dostupné z: https://blog.nviso.eu/2021/09/01/how-malicious-applications-abuse-android-permissions/

[18] GALLAGHER, Sean. Facebook scraped call, text message data for years from Android phones. ArsTECHNICA [online]. 2018, 3.24.2018 [cit. 2021-12-08]. Dostupné z: https://arstechnica.com/information-technology/2018/03/facebook-scraped-call-text-message-data-for-years-from-android-phones/

[19] Facebook–Cambridge Analytica data scandal. Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2021 [cit. 2021-12-08]. Dostupné z: https://en.wikipedia.org/wiki/Facebook%E2%80%93Cambridge_Analytica_data_scandal

[20] KUPRINS, Aleksejs. Analysis of Joker. Medium [online]. 2019, 3.9.2019 [cit. 2021-12-08]. Dostupné z: https://medium.com/csis-techblog/analysis-of-joker-a-spy-premium-subscription-bot-on-googleplay-9ad24f044451

[21] Google Play Protect. Google Developers [online]. 2021 [cit. 2021-12-08]. Dostupné z: https://developers.google.com/android/play-protect

[22] YALON, Erez. How Attackers Could Hijack Your Android Camera to Spy on You. Checkmarx [online]. 2019, 19.11.2019 [cit. 2021-12-08]. Dostupné z: https://checkmarx.com/blog/how-attackers-could-hijack-your-android-camera/

[23] MITRA, Amrita. What is the Confused Deputy Problem? The Security Buddy [online]. 2017 [cit. 2021-11-16]. Dostupné z: https://www.thesecuritybuddy.com/vulnerabilities/what-is- confused-deputy-problem/

Seznam symbolů a zkratek

AES Advanced Encryption Standard

API Rozhraní pro programování aplikací

CPU Centrální procesorová jednotka

DVM Dalvik Virtual Machine

GPS Globální polohový systém

GUI Grafické uživatelské rozhraní

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

IDE Vývojové prostředí

JVM Java Virtual Machine

MMS Multimediální zpráva

NFC Blízkopolní komunikace

OEM Originální výrobek výrobce

OS Operační systém

OWASP Open Web Application Security Project

PHP Hypertext Preprocessor

RSA Šifrovací algoritmus Rivest, Shamir, Adelman

QR Quick Response

SD Secure Digital

SDK Systémový vývojový nástroj

SIM Subscriber Identity Module

SMS Krátká textová zpráva

SQL Structured Query Language

SSL Secure Sockets Layer

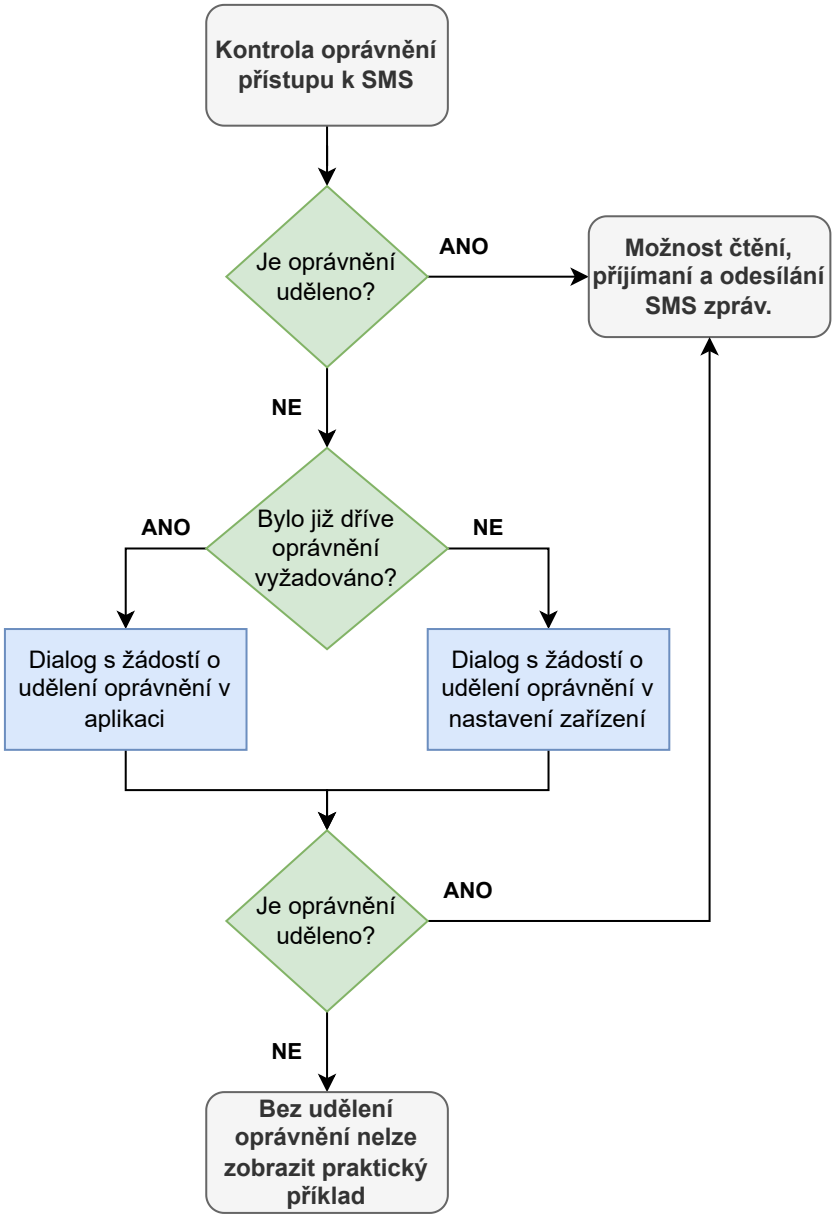
USB Univerzální sériová sběrnice

Wi-Fi Bezdrátová komunikace

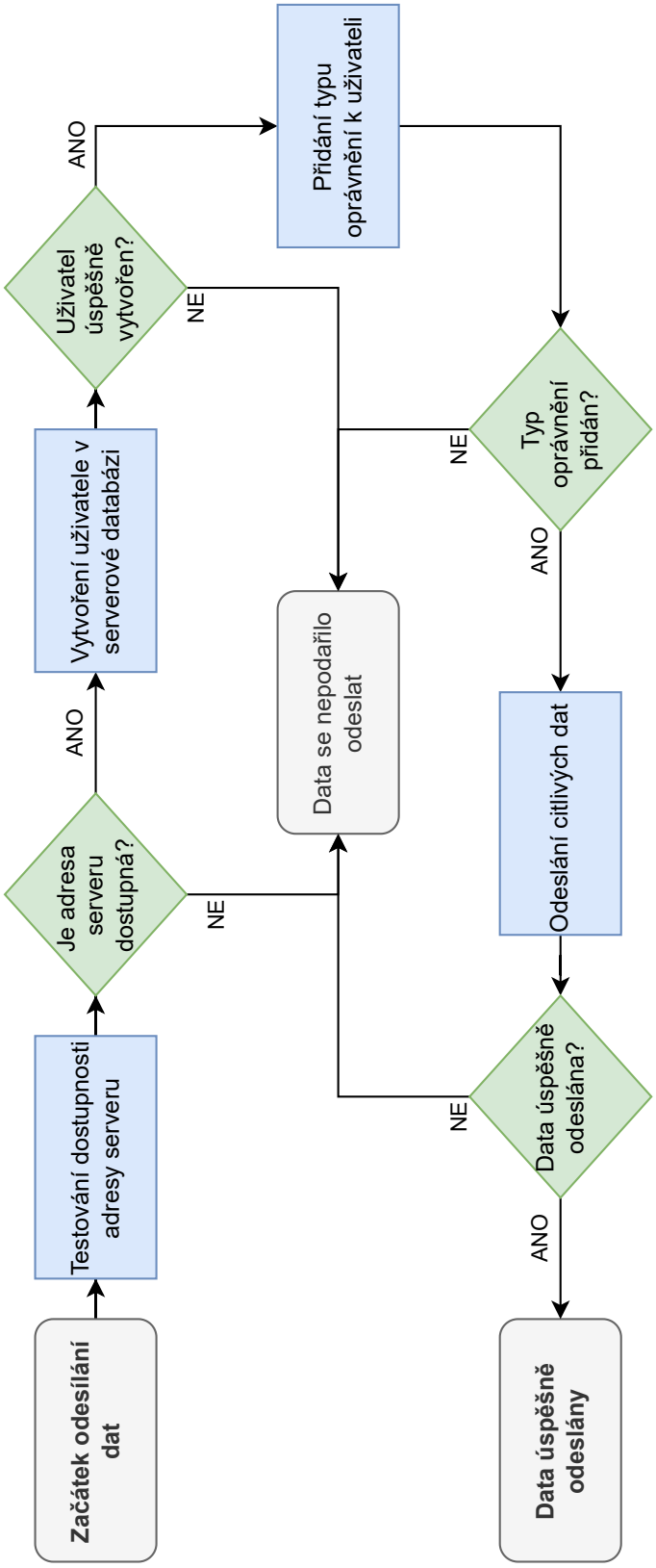
XML Extensible Markup Language

Seznam příloh

A Diagramy procesů mobilní aplikace

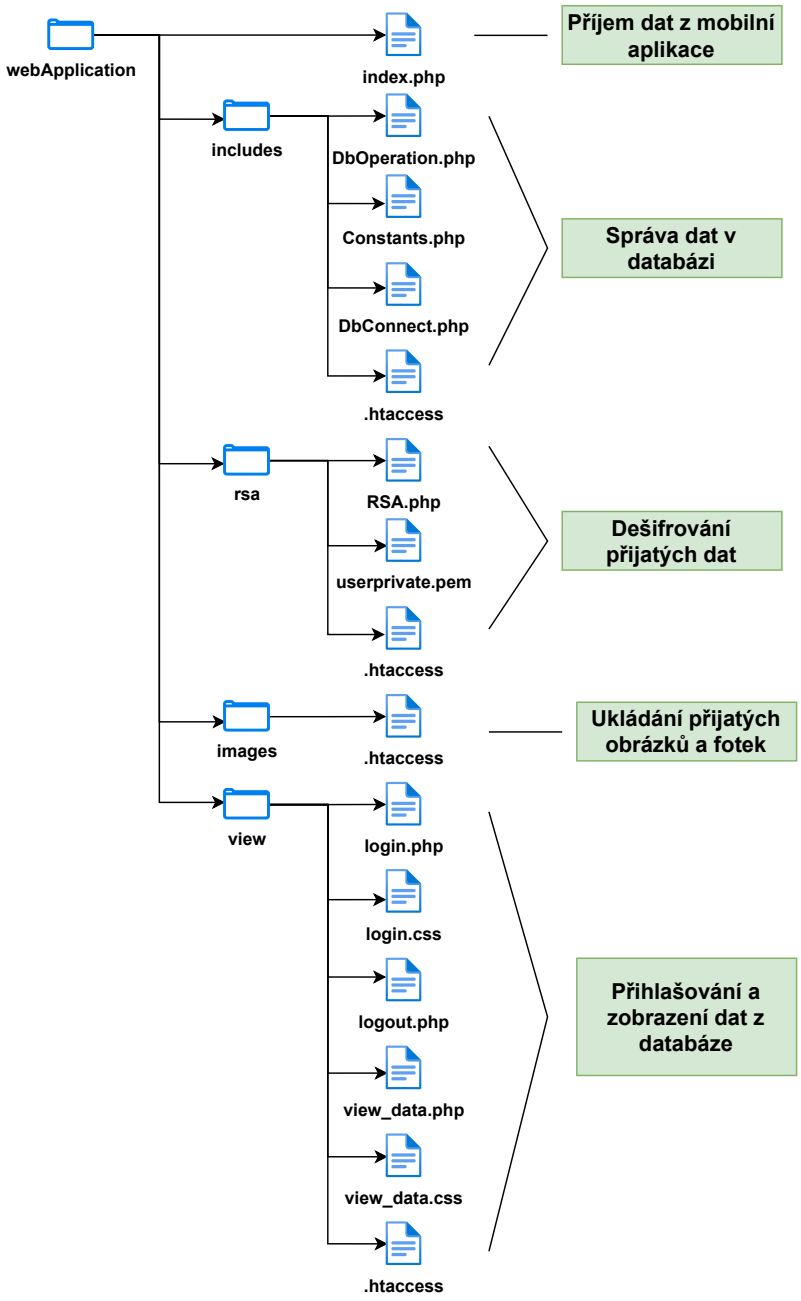


Obr. A.1: Diagram znázorňující proces vyžadování oprávnění.



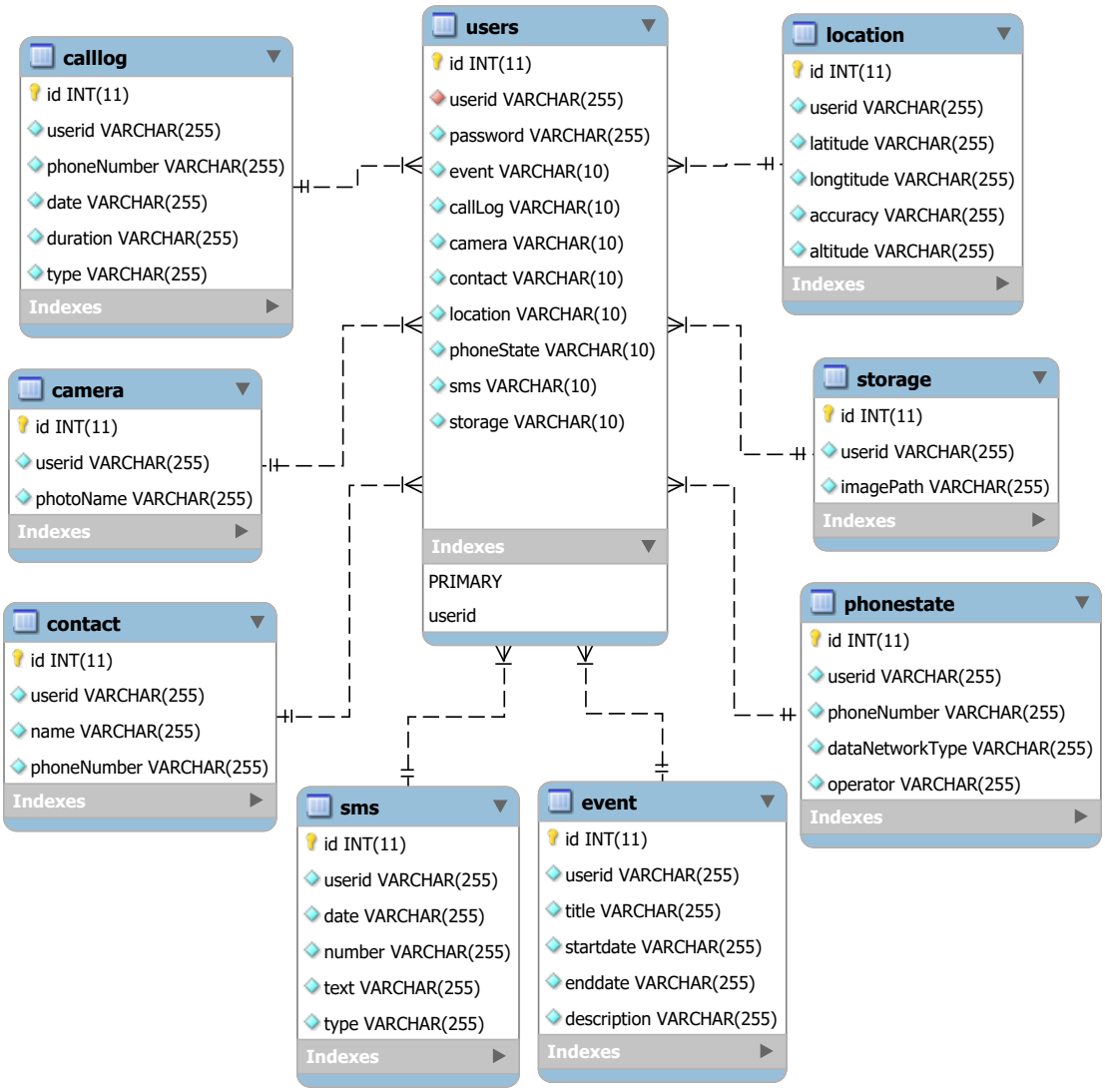
Obr. A.2: Diagram znázorňující proces odesílání dat na server.

B Struktura webové aplikace



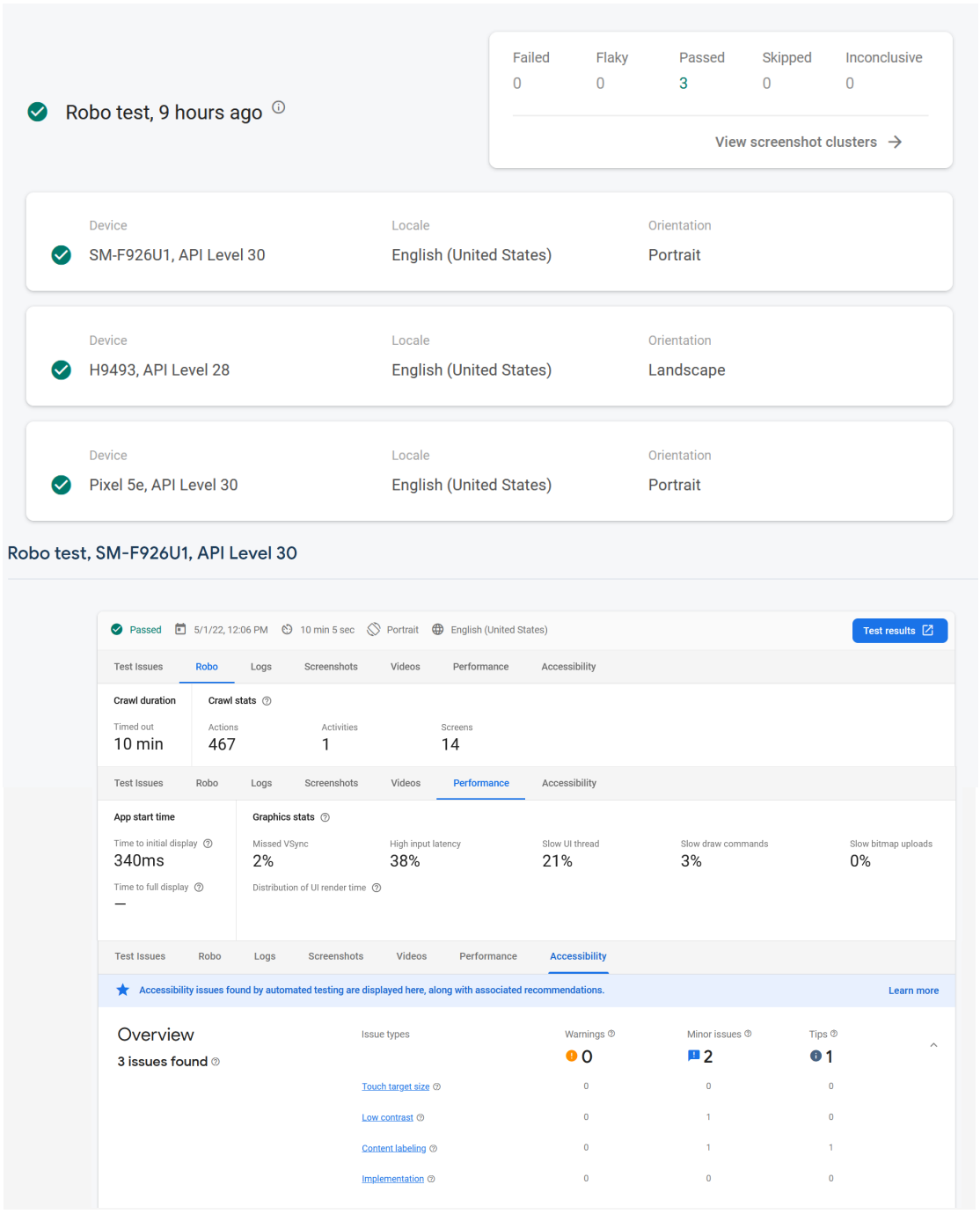
Obr. B.1: Struktura webové aplikace.

C Struktura MySQL databáze



Obr. C.1: Struktura MySQL databáze.

D Výsledky Robo testu



Obr. D.1: Výsledky Robo testu aplikace pro tři různá zařízení.

E Přiložené soubory

- `/` ..... kořenový adresář přiložených souborů
  - `Database` ..... Složka s MySQL databází
    - `insecureappdb.sql` ..... Soubor obsahující databázi ve formátu SQL
  - `Mobile application` ..... Složka se zdrojovými kódy a dokumentací aplikace
    - `Source code` ..... Složka se zdrojovými kódy aplikace
    - `Documentation` ..... Složka s dokumentací k aplikaci
    - `user_permissions.apk` ..... Instalační soubor aplikace
  - `Web application` ..... Složka s HTML a PHP soubory webové aplikace
  - `Diplomová_práce_Michálek.pdf` ..... Diplomová práce ve formátu PDF