



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
BRNO UNIVERSITY OF TECHNOLOGY

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**  
FACULTY OF INFORMATION TECHNOLOGY

**ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ**  
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

**NATIVNÍ MOBILNÍ APLIKACE PRO MONITORING  
VČELNICE**

NATIVE MOBILE APPLICATION FOR BEE SIDE MONITORING

**BAKALÁŘSKÁ PRÁCE**  
BACHELOR'S THESIS

**AUTOR PRÁCE**  
AUTHOR

**JAN OSOLSOBĚ**

**VEDOUCÍ PRÁCE**  
SUPERVISOR

**Ing. FRANTIŠEK GRÉZL, Ph.D.**

**BRNO 2021**

## **Abstrakt**

Cílem práce je vytvořit multiplatformní mobilní aplikaci, která pomáhá včelaři vlastníci zařízení od Forsage.net. Aplikace je vyvíjena pomocí frameworku Nativescript pro operační systémy Android a iOS. Vytvořené řešení poskytuje včelaři možnost přijímat notifikace, zobrazit webovou aplikaci a pomocí Bluetooth získat data ze senzorů. Přínosem této práce je, že poskytuje včelaři přehled o jeho včelstvech a upozornění v případě situace, na kterou je potřeba rychle reagovat.

## **Abstract**

The aim of this work is to create a multiplatform mobile application, that can help beekeepers who own devices from Forsage.net. This application has been developed by using the framework Nativescript for Android and iOS operating systems. The created result provides the beekeeper with the option to receive notifications, to display a web application and to use Bluetooth to obtain data from sensors. The major benefit of this application is that it provides the beekeeper with an overview on his beehives and warns him in a situation which needs to be responded to immediately.

## **Klíčová slova**

včelaření, vzdálený monitoring včelnice, mobilní aplikace, nativescript, multiplatformní programování, push notifikace, Bluetooth Low Energy

## **Keywords**

beekeeping, distanced bee side monitoring, mobile application, nativescript, multiplatform programing, push notification, Bluetooth Low Energy

## **Citace**

OSOLSOBĚ, Jan. *Nativní mobilní aplikace pro monitoring včelnice*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. František Grézl, Ph.D.

# **Nativní mobilní aplikace pro monitoring včelnice**

## **Prohlášení**

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana X... Další informace mi poskytli... Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....  
Jan Osolsobě  
5. května 2021

## **Poděkování**

V této sekci je možno uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc (externí zadavatel, konzultant apod.).

# Obsah

<b>1</b>	<b>Úvod</b>	<b>2</b>
<b>2</b>	<b>Vzdálený monitoring včelnice</b>	<b>3</b>
2.1	Senzory . . . . .	3
2.2	Existující řešení . . . . .	4
<b>3</b>	<b>Návrh mobilní aplikace</b>	<b>6</b>
3.1	Diagram případů užití . . . . .	6
3.2	Přijímání notifikací . . . . .	6
3.3	Zobrazení webové aplikace . . . . .	7
3.4	Získání dat ze senzorů pomocí Bluetooth . . . . .	7
3.5	Grafické rozhraní . . . . .	7
<b>4</b>	<b>Použité technologie</b>	<b>9</b>
4.1	Metody vývoje mobilní aplikace . . . . .	9
4.2	NativeScript . . . . .	12
4.3	Vue.js . . . . .	13
4.4	Firebase . . . . .	13
4.5	Apollo GraphQl . . . . .	14
4.6	NativeScript WebView Interface . . . . .	14
4.7	Bluetooth Low Energy . . . . .	15
<b>5</b>	<b>Implementace řešení</b>	<b>16</b>
5.1	Push notifikace . . . . .	16
5.2	Oboustranná komunikace s Webview . . . . .	17
5.3	Získání dat ze senzorů . . . . .	20
5.4	Grafické rozhraní . . . . .	21
<b>6</b>	<b>Testování</b>	<b>24</b>
6.1	Praktické testování . . . . .	24
6.2	Nalezené chyby . . . . .	26
6.3	Nápady na vylepšení . . . . .	26
<b>7</b>	<b>Závěr</b>	<b>27</b>
	<b>Literatura</b>	<b>28</b>
<b>A</b>	<b>Obsah přiloženého CD</b>	<b>30</b>

# Kapitola 1

## Úvod

Včely jsou pro naši planetu jedním z nejdůležitějších stvoření. Opylují až 80 procent našich pěstovaných plodin[2] a tím poskytují stravu až pro 90% lidské populace [12]. Včelaření se u nás stává populárním koníčkem i u mladší generace.

Technologie se začínají promítat i do včelaření. Díky vzdálenému monitoringu včelstev dokážeme získávat informace o úlu, i když u něj nejsme. Pomocí změny dat dokážeme předpovídат některé události. Díky tomu může včelař zredukovat počet svých návštěv v úlu a tím ušetří včely od stresu a svůj volný čas. Právě vzdáleným monitoringem včelstev se zabývá firma Forsage.net, která poskytuje včelaři informace o jeho úlu pomocí úlové váhy a senzoru v úlu. Ve spolupráci s touto firmou je tato práce vytvořena.

V této práci jsou probrány možnosti vzdáleného monitoringu včelstev a metody vývoje multiplatformní aplikace založené na jazyku JavaScript. Dále je zde popsán návrh, implementace a testování mobilní aplikace.

Cílem této práce je vytvořit nativní mobilní aplikaci pro mobilní platformy Android a iOS pomocí frameworku NativeScript. Tato aplikace zobrazuje webovou aplikaci, ve které se včelaři zobrazují měřené údaje o jeho úle. Toto zobrazení musí spolu s mobilní aplikací komunikovat. Další funkcí aplikace je získání dat ze senzorů pomocí Bluetooth Low Energy. To slouží, když včelař navštíví stanoviště s úly a chce zjistit aktuální situaci v úlu. Důležitou součástí aplikace je příjemání notifikací poslaných ze serveru. Díky notifikacím dokáže včelař zareagovat při situaci, která vyžaduje rychlou reakci. Takovou situaci může být rojení včel, u kterého kdyby rychle nezareagoval, přišel by včelstvo. Hlavním cílem je vytvořit uživatelsky přívětivou mobilní aplikaci, která usnadní včelám i včelaři život.

## Kapitola 2

# Vzdálený monitoring včelnice

Včelaření je pro mnoho lidí koníček, pro některé i práce. Při vlastnění více úlů na různých místech (stanovištích) musí včelař při kontrole navštívit postupně všechny. Vzdálený monitoring včelnice slouží k tomu, aby včelař nemusel svoje stanoviště s úly navštěvovat tak často a tím může ušetřit čas. Důležité je to i pro včely. Včelám každý zásah v úlu včelařem způsobuje stres a to včelám škodí. Díky vzdálenému monitoringu dokážeme lépe plánovat zásahy v úlu, nebo předcházet nechtěným událostem. Pomocí sbírání dat si včelař dokáže zobrazit data v kontextu historického vývoje.

Včelaře zajímá jakou hmotnost má jeho úl. Pomocí hmotnosti jde zjistit například kolik včely nanosily nektaru, nebo v zimě jaký je úbytek jejich zásob. Teplota v úlu nám dokáže v zimě napovědět, jestli včely v úlu stále jsou. Pomocí změny vlhkosti dokážeme zjistit, kdy se přeměňuje nektar na med, nebo kdy začne matka plodovat (klást vajíčka)[13]. Tyto vlastnosti ovlivňuje také počasí, proto je potřeba při vyhodnocování s tím počítat. Pomocí analýzy bzučení v úlu se dá poznat, jestli jsou včely ve stresu, což může být důsledkem rojení, nebo nepřítomnosti matky [16]. Další funkci, kterou jde pomocí monitoringu včelnice poznat a upozornit včelaře je, když někdo úl odcizí, nebo se převrhne.

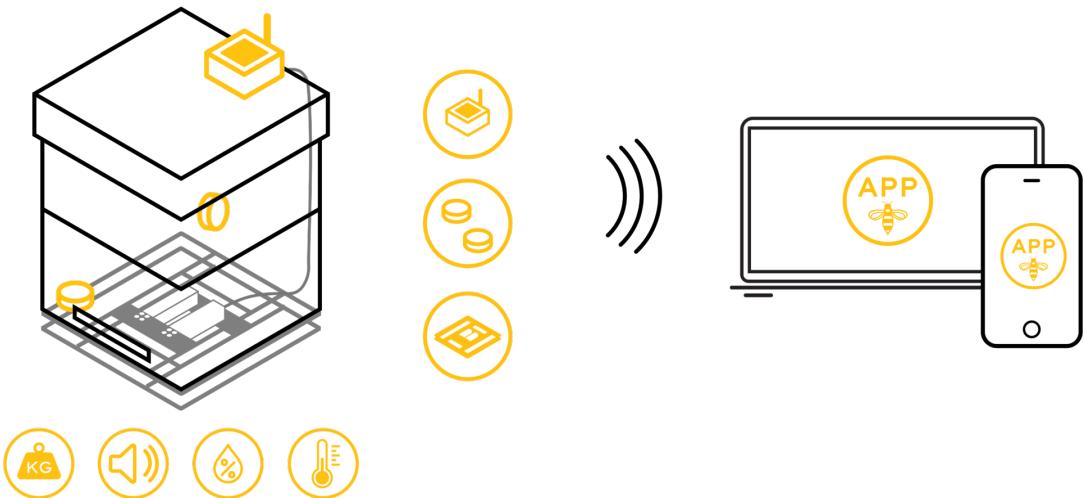
### 2.1 Senzory

Na měření vlastností na trhu existují různé druhy senzorů. Existují senzory v úlu pro měření teploty, vlhkosti a snímání zvuku. Úlové váhy na kterých stojí celý úl snímají jeho váhu. Některé měří i teplotu a vlhkost venkovního prostředí. Některé firmy nabízejí i senzor, který se vloží do česna (hlavního vstupu do úlu) a který scítá přilétající a odlétající včely.

Firma Forsage.net nabízí dva druhy senzorů: úlovou váhu a senzor 2.1, který se vkládá do úlu. Tyto senzory posílají pomocí Bluetooth data na přenašeč, který je v pravidelných intervalech posílá na server. Forsage.net využívá získaná data k následnému výzkumu včel.

#### Úlová váha

Úlová váha je umístěna pod úlem, mezi podstavcem a dnem úlu. Díky ní dokážeme sledovat změny váhy úlu. Můžeme pozorovat, kolik včely denně přináší medu, nebo kolik zásob jim zbývá v zimě. Váha má maximální nosnost 200 kg. Váhy čerpají energii z baterie přenašeče. Pokud je do jednoho přenašeče zapojeno více vah, jsou zapojeny sériově.



Obrázek 2.1: Schéma senzorů [8]

### Senzor v úlu

Senzor v úlu snímá zvuk, měří teplotu a vlhkost v úlu. Tyto senzory jsou od finské společnosti Ruuvi. Měří nejen teplotu a vlhkost, ale i tlak a má v sobě akcelerometr. Tento senzor lze umístit na různá místa v úlu i mimo něj, protože je voděodolný. Když je umístíme blízko česna, dokážeme měřit hodnoty vnějšího prostředí. Pomocí senzoru v medníku (prostor, kam včely mají tendenci ukládat med) dokáže podle změny vlhkosti odhadnout, kdy se tvoří v úlu med. Díky tomu, že vysílá data pomocí Bluetooth Low Energy, baterie vydrží až 3 roky.

### Přenašeč

Přenašeč v pravidelných intervalech sbírá data ze senzorů a ty následně přeposílá na server, kde data zobrazuje webová aplikace. Pro maximalizování výdrže baterie data z senzorů sbírá každých patnáct minut a posílá je na server vždy jednou za hodinu. Během zimy se tento interval prodlužuje na několik hodin, protože včely nejsou tak aktivní. K přenosu dat využívá GSM síť, proto se v přenašeči nachází datová SIM karta. Na jeden přenašeč je možné napojit více vah a senzorů. Podle počtu zařízení se odvozuje trvanlivost baterie. Průměrně vydrží jeden rok.

## 2.2 Existující řešení

V současné době existuje několik řešení monitoringu včelstev. U většiny jsou přístupné demo účty jejich aplikací, které jsem vyzkoušel. Provedl jsem jejich srovnání a vybral klíčové možnosti firem. Srovnání jednotlivých firem je sepsáno v tabulce 2.1. Většina firem je z



Obrázek 2.2: Ukázka senzorů [8]

Firma	Váha	Senzor v úlu	Mobilní aplikace	Webová aplikace	Posílaní notifikací
Bee Hive Monitoring	Ano	Měří teplotu, vlhkost, zvuk	Nepřehledné zobrazení grafů	Nepřehledné uživ. prostředí, z příplatek	Ano
SolutionBee	Ano	Měří teplotu, vlhkost, zvuk	Starší uživ. rozhraní	Přehledné, intuitivní uživ. rozhraní	Ano
BeeSpy	Ano	Ne	Ne	Jednoduché a přehledné uživ. rozhraní	Ne
Forsage.net	Ano	Měří teplotu, vlhkost, zvuk	Ne	Přehledné uživ. rozhraní	Ne

Tabulka 2.1: Přehled existujících řešení

Česka nebo ze Slovenska, protože je tu největší koncentrace včelařů na světě [19]. Nicméně jsem vybral i řešení od firmy z USA SolutionBee, které je velmi propracované. Pokud bych se zaměřil pouze na mobilní aplikace, většina firem je nemá. Aplikace jsou ve špatném stavu, mají nepřehledné uživatelské rozhraní a jednoduše se v nich dá ztratit.

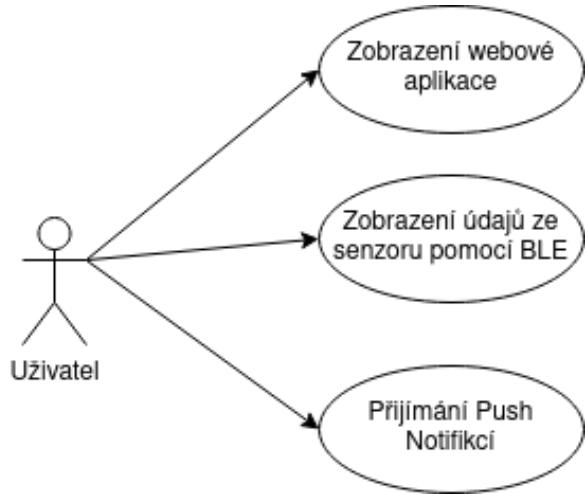
## Kapitola 3

# Návrh mobilní aplikace

V následující kapitole se budu věnovat návrhu aplikace. Od firmy Forsage.net jsem dostal zadání, že aplikace musí umět přijímat notifikace poslané ze serveru, zobrazovat její webovou aplikaci a získávat data přímo ze senzoru pomocí Bluetooth. Následný návrh aplikace, výběr technologií a implementace byla na mé uvážení.

### 3.1 Diagram případů užití

Diagram případů užití zachycuje vnější pohled na systém[1]. Definuje, jaké má uživatel možnosti. Diagram se skládá z aktérů, v tomto případě uživatele, a z případů užití, neboli akcí které uživatel může využít.



Obrázek 3.1: Diagram případů užití

### 3.2 Přijímání notifikací

V úlu mohou nastat situace, u kterých je potřeba rychle reagovat, například když se rojí včely, nebo kdyby nám někdo kradl úly a váha najednou rychle klesla. V takových situacích je potřeba co nejrychleji upozornit včelaře, aby mohl zareagovat. Situace, u kterých je potřeba

Upozornění pro včelaře	Příznaky	Období	Příčina
Úl byl ukraden	Váha klesla na minimum	-	Úl se převrhla, úl někdo ukradl
Zařízení přestalo komunikovat	Nepřichází data z zařízení	-	Možná ztráta signálu, vybití baterie, zničení zařízení
Baterie v senzoru je skoro vybitá	Zmenšení napětí v baterii	-	Baterie je vybitá, potřeba vyměnit / nabít
Úl byl vyloupen jiným včelstvem	Prudký pokles váhy o 2-3 kg během několika hodin	září - říjen	Zásoby slabého včelstva byly vyloupeny jiným včelstvem
Matka začala plodovat	Zvýšení vlhkosti, postupný pokles váhy	prosinec - leden	Včelí matka začala klást vajíčka
Potřeba přikrmít včely	Váha úlu se blíží váze prázdného úlu se včelstvem	únor - duben	Včely spotřebovali většinu zimních zásob
Začala snůška	Váha začala růst a zvýší se vlhkost v úlu	duben - červenec	Včely začaly sbírat nektar
Skončila snůška	Váha úlu se zastavila/klesá	duben - červenec	Včely přestávají sbírat pyl/ nektar
Potřeba přidat nástavek	Váha úlu dosáhne určité hodnoty	duben - červenec	Včely nemají kam ukládat pyl/nektar, hrozí rojení
Nízká teplota v úlu	Stejná teplota v úlu jako venkovní	prosinec - březen	Včely jsou slabé nebo uhynuly

Tabulka 3.1: Přehled upozornění pro včelaře

reagovat, jsou sepsány v tabulce 3.1 i s příznaky, jak situaci poznat, obdobím, ve kterém se daná situace může vyskytovat a příčinou situace.

Mobilní aplikace má tyto notifikace přijímat a zobrazit uživateli. V budoucnu je v plánu přidat události vyhodnocené z analýzy bzučení.

### 3.3 Zobrazení webové aplikace

Druhou část aplikace tvoří zobrazení webové aplikace. V mobilní aplikaci využijeme nativní prvky jednotlivých mobilních platform. Mobilní aplikace ale musí s webovou komunikovat, protože aby se například při odhlášení z webové aplikace uživatel odhlásil i z mobilní. Mobilní aplikace také musí reagovat při změně uživatele.

### 3.4 Získání dat ze senzorů pomocí Bluetooth

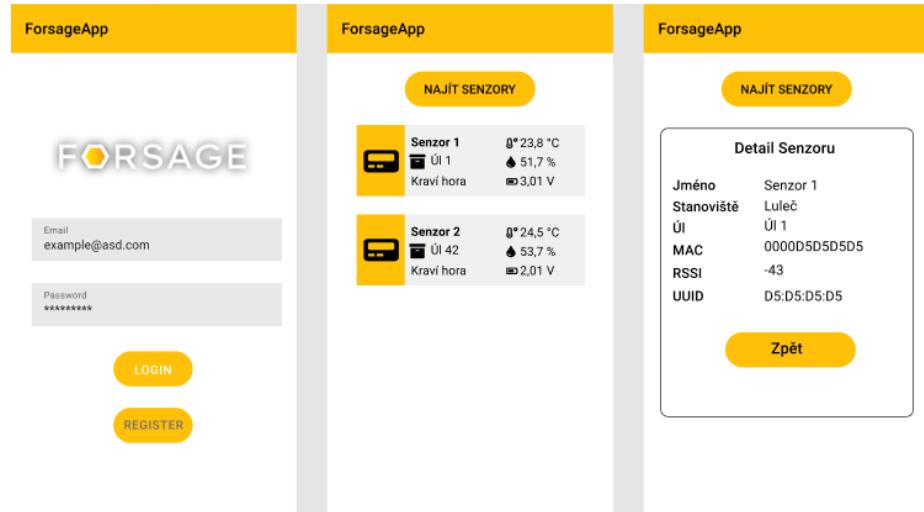
Třetí částí je zjištění dat přímo ze senzorů pomocí Bluetooth. Když chce včelař při návštěvě svého stanoviště zjistit, jaké teplota je v úlu, nače si data ze senzoru uvnitř. Aplikace získá seznam senzorů uživatele, a následně zobrazí data ze senzorů a data o senzorech (úl, stanoviště, atd.).

### 3.5 Grafické rozhraní

Po vyjasnění co má aplikace umět, můžeme přejít k návrhu grafického rozhraní. Naše aplikace, se skládá z zobrazení webové aplikace a samotné části pro práci s Bluetooth. Mým cílem bylo, aby uživatel nepoznal rozdíl, kdy se nachází v zobrazení webové aplikaci nebo v aplikaci samotné. Proto jsem se snažil volit grafické rozhraní podobné webové aplikaci.

#### 3.5.1 Figma

Pro návrh grafického rozhraní jsem využil webovou aplikaci Figma. Figma pracuje s vektorovou grafikou, a používá se k návrhu prototypů. Návrh je občas poměrně zdlouhavý, ale jde pak využít, protože Figma ukazuje CSS jednotlivých prvků.



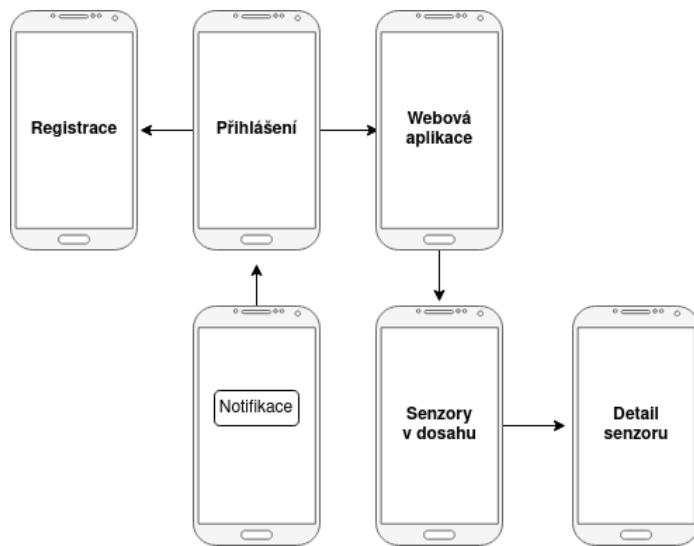
Obrázek 3.2: Grafické návrhy částí aplikace

Vytvořil jsem několik návrhů grafického rozhraní pro přihlášení, stránku skenování pomocí Bluetooth a detailní informace daného senzoru. Příklad je možno vidět na obrázku 3.2.

### 3.5.2 Vztahy mezi obrazovkami

Pro lepší představu jsem vytvořil diagram 3.3, který znázorňuje přechody mezi jednotlivými obrazovkami. V obrazovce "Webová aplikace" se dá přecházet ve webové aplikaci mezi jednotlivými stanovišti, úly a senzory, zobrazit grafy atd.

Přechod je vždy myšlen tak, že se dá z jedné stránky přejít na další, ale i zpět na předchozí.



Obrázek 3.3: Grafické návrhy částí aplikace

# Kapitola 4

## Použité technologie

Mobilní aplikace se dají vyvíjet různými způsoby. Můžeme se specializovat pouze na jednu mobilní platformu, nebo vyvíjet multiplatformně pomocí technologií založených na JavaScriptu. Popíšeme si zde mobilní platformy, pro které budeme vyvíjet a technologie, které byly během vývoje použity.

### 4.1 Metody vývoje mobilní aplikace

Protože budeme vyvíjet multiplatformně, v této sekci se podíváme na jednotlivé mobilní platformy a metody vývoje.

#### 4.1.1 Mobilní platformy

V současné době jsou na trhu mobilní telefony hlavně s dvěma platformami. Je to Android 4.1.1 od společnosti Google a iOS 4.1.1, kterou vyvíjí Apple. Mezi další mobilní platformy patří Windows Phone, BlackBerry OS, nebo některé mobilní telefony běží na systémech založených na Linuxu. Tyto platformy se vyskytují pouze ojediněle, proto se budu zabývat pouze platformami Android a iOS. 4.1 Každá z platform má svoje specifika, nebo také jinak přistupuje k uživatelům.

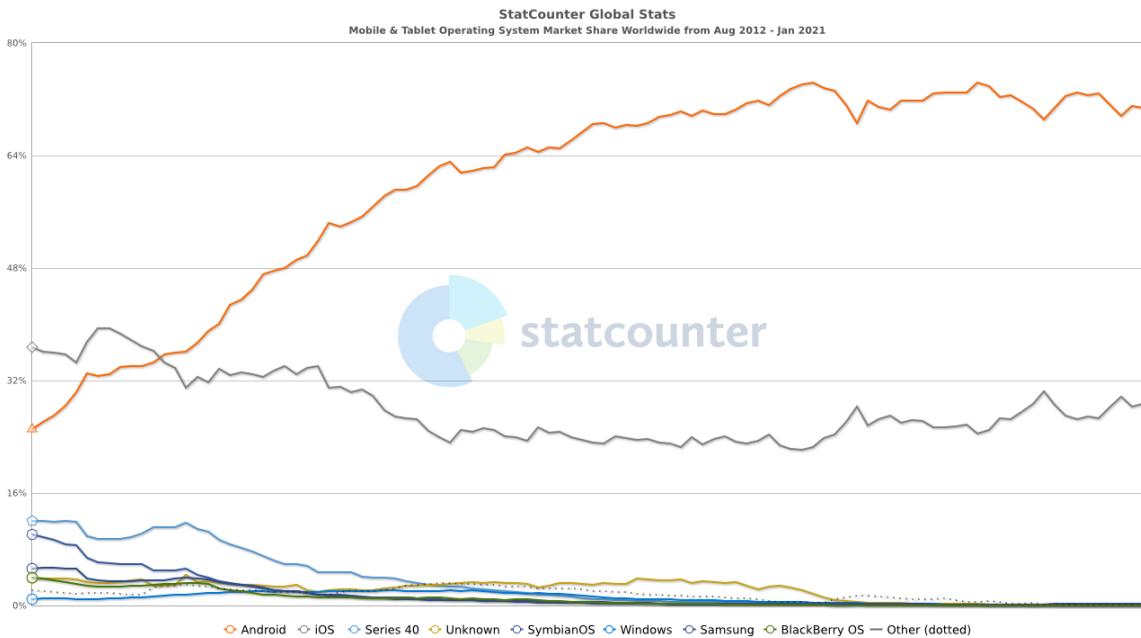
#### Android

Android je nejrozšířenější mobilní platforma. Vývoj vede organizace Open Hand's Alliance, která spadá pod společnost Google. Android je založen na linuxovém jádře[15], které je dostupné jako otevřený software. Výrobci mobilních zařízení si za dodržení stanovených podmínek mohou Android upravovat, proto existuje mnoho různých verzí. Výrobci upravují nejen konfigurace nebo widgety, ale například i firmware.

Společnost Android Inc. vznikla v roce 2003, v roce 2007 ji odkoupila společnost Google a následně založila konsorciump Open Hand's Alliance, které zahrnovalo další společnosti, které působily v oblasti mobilních technologií. V roce 2008 byl uveden na trh první mobilní telefon s operačním systémem Android.

Architektura systému Android je rozdělená do pěti vrstev. Každá z vrstev provádí svoje úkony a v případě potřeby mezi sebou komunikují. K lepší představě slouží obrázek 4.2.

Aplikace se na této platformě šíří hlavně pomocí distribuční služby Google Play. Díky tomu, že služba nemá tak přísné podmínky pro umístění aplikace a existuje mnoho druhů verzí a velikostí displeje, je mnoho aplikací nekvalitních, nebo neodpovídají danému zařízení.



Obrázek 4.1: Vývoj poměru zastoupení mobilních platforem 2012-2021[9]

## iOS

Tato mobilní platforma se nachází na telefonech iPhone, tabletech iPad, nebo i v upravené verzi na hodinkách Apple Watch. iOS je určen pouze pro produkty společnosti Apple. Vývoj a testování aplikací je možné pouze na zařízení od stejného výrobce, kde je pro to vyvinuta aplikace XCode.

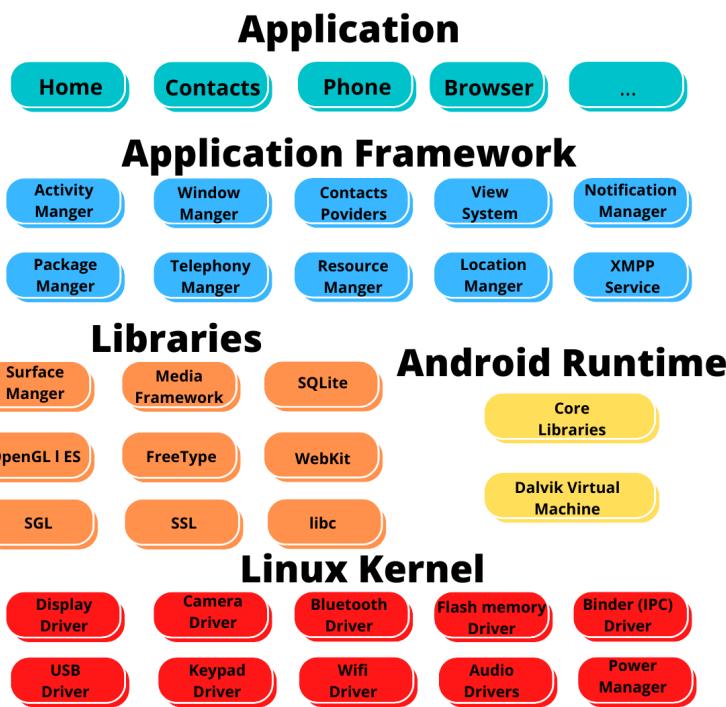
Společnost Apple vznikla v roce 1976. Nejprve se věnovali výrobě počítačů a v roce 2007 byl představen první iPhone, na kterém byla i první verze iOS. Nyní již vyšla dvanáctá řada telefonu iPhone.

Architektura systému iOS je odlehčenou verzí operačního systému macOS, který se používá na počítačích společnosti Apple. Systém je unixového typu a je rozdělen na čtyři vrstvy[14]. Platí, že vrstva vždy využívá pouze služby vrstvy přímo pod ní. Naopak spodní vrstva nemá žádné informace o vrstvách nad ní. Na nejvyšší vrstvě aplikace komunikují s aplikačním rozhraním vrstvy a využívají potřebné frameworky. Vrstvy jsou zobrazeny na obrázku 4.3.

Šířit aplikace se dá pouze pomocí aplikace App Store, která má přísné podmínky, aby se na ní aplikace dala umístit. Aplikace musí být kvalitní a dostatečně otestované.

### 4.1.2 Vývoj multiplatformních mobilních aplikací

Cílem vývoje aplikace je, aby se dostala k co nejvíce uživatelům. Jelikož máme několik mobilních platforem, lze si usnadnit práci vyvíjením aplikace na více mobilních platforem zároveň. Tento vývoj se nazývá multiplatformní a umožňuje vytvořit pomocí jednoho kódu spustitelné aplikace na jednotlivých platformách. Díky tomu není vývoj tak náročný jako kdybychom vyvíjeli aplikaci pro každou platformu zvlášť.



Obrázek 4.2: Struktura operačního systému Android

## Nativní aplikace

Nejstarší metodou vývoje mobilních aplikací je vytvořit aplikaci nativně - samostatně pro jednotlivé platformy. U platformy Android se využívá vývojové prostředí Android Studio a programovací jazyky Kotlin nebo Java. Pro platformu iOS prostředí XCode a jazyky iOS Swift nebo ObjectiveC.

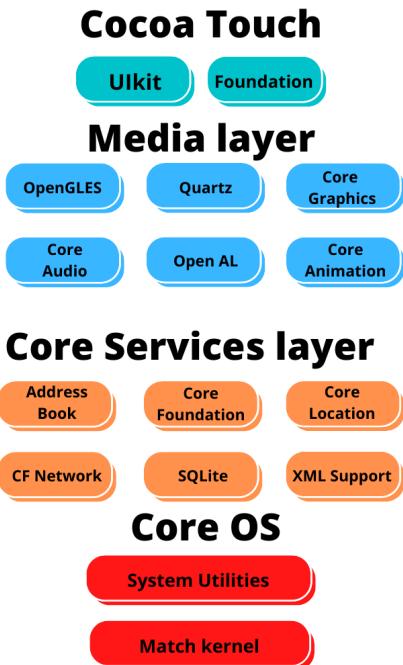
Výhodou nativní aplikace je většinou větší výkon, protože je uzpůsobena právě na onu platformu. Dále dokáže využívat specifické funkce a nativní UI prvky specifické pro danou platformu.

## Multiplatformní nativní aplikace

Multiplatformní nativní aplikace využívají stejný kód. Tento kód je později přeložen do nativního kódu jednotlivých plafotrem[10]. Díky tomu dokážeme jednoduše vyvíjet aplikace pro více plafotrem současně.

Nejnovější trend ve vývoji mobilních aplikací jsou multiplatformní aplikace interpretované za běhu. Tyto aplikace jsou psány v JavaScriptu, a překládány pomocí frameworků na jednotlivé platformy.

Mezi příklady frameworků, které takto pracují jsou NativeScript ,React Native, nebo Titanium. Pro mojí aplikaci jsem si vybral právě NativeScript, který je popsán v sekci 4.2.



Obrázek 4.3: Struktura operačního systému iOS

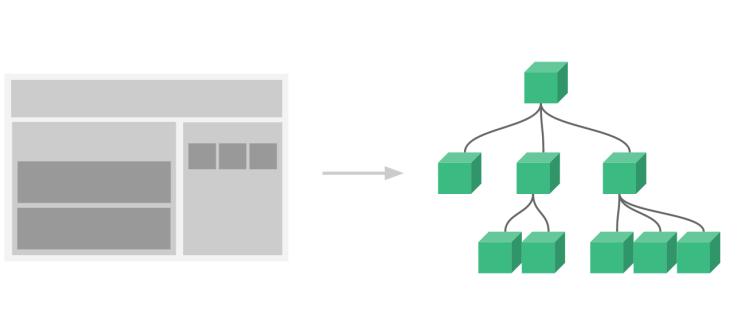
## 4.2 NativeScript

NativeScript je open-source framework, který se používá pro vývoj mobilních aplikací na platformách Android a iOS. NativeScript byl vytvořen firmou Telerik v roce 2014. Tato firma jej dále rozšiřuje. NativeScript využívá JavaScript, CSS a XML[7].

Vývoj mobilních aplikací pomocí NativeScriptu je možný na všech desktopových platformách. Pouze u vývoje pro platformu iOS je k přeložení potřeba XCode, což je vývojové prostředí pro vývoj aplikací pro iOS a Mac OS, které je dostupné pouze na platformě Mac[11]. Aby se dalo vyvíjet aplikace pro iOS i na platformách Linux a Windows vytvořil NativeScript vývojové prostředí NativeScript Sidekick, na kterém můžeme vzdáleně kompilovat aplikace. Kompilace probíhá na stroji Mac Pro. Díky tomuto prostředí není nutné vlastnit zařízení Mac pro vývoj aplikací pro iOS.

NativeScript využívá pro stylování CSS, pomocí kterého lze navíc i vytvářet animace. CSS pochází z webového prostředí, kde slouží k popisu zobrazení jednotlivých elementů. Díky tomu, že jej zná a používá mnoho vývojářů se CSS dostává i do jiných technologií, včetně NativeScriptu.

Původně se dalo vyvíjet pouze pomocí JavaScriptu[18], avšak nyní je možné využít TypeScript, který je nadstavbou JavaScriptu a rozšiřuje jej o statické typování a další atributy, které známe z objektově orientovaného programování. Dále je možné využít frameworky AngularJS a Vue.js. AngularJS vyvíjí společnost Google a je navržen aby pomohl oddělit zobrazovací a aplikační logiku. Frameworku Vue.js se věnuje následující kapitola 4.3.



Obrázek 4.4: Dělení aplikace na komponenty[20]

### 4.3 Vue.js

Při vývoji mobilní aplikace jsem využíval Javascriptového frameworku Vue.js[20]. Vue.js vytvořil v roce 2013 Evan You, a inspiroval se jiným javascriptovým rámcem AngularJS. Tento framework je progresivní, což znamená, že můžeme aplikaci rozdělit na části. Ty vyvíjet nezávisle na sobě. Vue.js je také reaktivní. Díky tomu se při každé změně dat sám překreslí.

Základní knihovna je zaměřena pouze na vrstvu zobrazení a je velmi snadné ji vyzvednout a integrovat s jinými knihovnami nebo stávajícími projekty.

Vue.js komponenty rozšiřují základní HTML prvky a zapouzdřují je do znovupoužitelného kódu. Díky tomu dokážeme rozsáhlou aplikaci skládat z malých, samostatných a opakovaně použitelných komponent. Komponenty poté tvoří stromovou strukturu, kterou můžeme vidět na obrázku 4.4.

#### 4.3.1 NativeScript-Vue

NativeScript-Vue je plugin pro NativeScript, který umožní využívat Vue.js k vytvoření mobilní aplikace. Oproti Vue.js se v některých případech liší v syntaxi, nicméně přebírá veškeré výhody Vue.js.

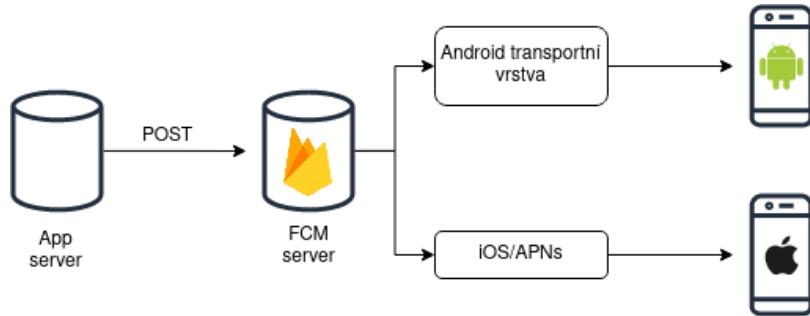
### 4.4 Firebase

Firebase je platforma vyvinutá společnosti Google[6], která slouží pro vytváření mobilních a webových aplikací. Nabízí vývojáři posílání Push notifikcí, Real-time databáze, autentizace, správu uživatelů, cloudové funkce a další možnosti. Některé služby Firebase jsou placené.

#### Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) slouží k zasílání Push Notifikací na telefonní zařízení, nebo webový prohlížeč. Zařízení, které má notifikaci přijmout je nejprve potřeba zaregistrovat. Aplikace se zaregistrouje na webu Firebase. Když chce server poslat notifikaci na dané zařízení, pošle na FCM server identifikační token onoho zařízení a token aplikace. Mobilní platformy se liší v přijímání a zpracování Push notifikací, proto musí FCM server specificky předat zprávu pro jednotlivé platformy. Identifikační token se v průběhu mění podle instalace, proto je potřeba ho při změně aktualizovat.

U zařízení s Android se zpráva předá do transportní vrstvy, kde se notifikace zpracuje za pomocí služeb Google Play. U iOS se notifikace zpracovávají pomocí Apple Push Noti-



Obrázek 4.5: Diagram toku notifikací

fication service (APNs). Tok notifikací je znázorněn v obrázku 4.5. Pro posílání na iOS je potřeba mít aplikaci registrovanou a certifikovanou v Xcode.

## 4.5 Apollo GraphQL

Apollo je sada nástrojů pro práci s GraphQL. Dělí se na Apollo Client a Apollo Server. Apollo Client je knihovna pro JavaScript. Dá se díky tomu využít pro webové, ale i nativní mobilní aplikace. Apollo Server je serverová část pro tvorbu GraphQL aplikačního rozhraní. V aplikaci používám Vue Apollo.

### GraphQL

GraphQL je dotazovací jazyk pro tvorbu API. Využívá se pro komunikaci mezi serverem a aplikací. Vyvinula ho společnost Facebook v roce 2012 pro svou mobilní aplikaci a o tři roky později se objevuje jako open-source. GraphQL je silně typované, proto, když chceme pracovat s nějakou entitou, musíme nejdříve definovat její typ.

GraphQL obsahuje tři základní objektové tipy a to Query, Mutation a Subscription. Typ Query obsahuje veškeré dotazy, které můžeme volat. Mutation zastupují zbylé operace, například mazání, přidávání, aktualizování dat. Pomocí Subscription dokážeme udržovat se serverem aktivní spojení a tím pádem číst aktuální data.

## 4.6 NativeScript WebView Interface

Je plugin pro oboustrannou komunikaci mezi komponentem `WebView` a nativní aplikací na Androidu, nebo na iOS. Přes standardní komponent `WebView` nelze předávat data, proto bylo potřeba využít tento plugin. Pomocí tohoto pluginu můžeme vysílat a příjemat události vyvolané z `WebView`, či z nativní aplikace.

U mobilní platformy Android využívá jeho rozhraní, přes které je možné předávat data[3]. U platformy iOS je to složitější. Na ní nemůžeme posílat data z `WebView` do aplikace. Proto to plugin obchází ve dvou krocích. Nejprve vytvoří dočasný `iFrame`, což je vnořený rámec do kterého je vložena jiná stránka. Do metadat `iFrameu` uloží data, které v druhém kroku data z rámce získá. Díky tomu dokážeme předat jméno události, která se má spustit.

## 4.7 Bluetooth Low Energy

Bluetooth Low Energy je čtvrtá verze Bluetooth, což je standard pro bezdrátovou komunikaci mezi dvěma zařízeními[17]. Oproti ostatním verzím má nízkoenergetickou náročnost, což ale zamezuje většímu toku dat. Proto se využívá hlavně u přenosu dat z mobilního zařízení k chytrým hodinkám nebo chytrým zařízením v domácnosti.

### Bluetooth Advertisement

Bluetooth Advertisement, neboli Bluetooth reklama je metoda mobilního marketingu. Využívá se k dat do mobilních zařízení. Příjemce ale musí výslovně souhlasit s příjmem této reklamy. V našem případě se v Bluetooth Advertisementu posílají data ze senzoru.

# Kapitola 5

## Implementace řešení

Aby byla implementace úspěšná, bylo potřeba zvolit správné technologie a provést dobré návrh aplikace. Nyní se můžeme přesunout k samotné implementaci řešení. Implementace je rozdělena podle funkčních celků, které jsou v aplikaci použity. Je zde popsána implementace mobilní aplikace, ale i komunikace se serverem.

### 5.1 Push notifikace

K posílání Push notifikací používám plugin Firebase. Uživateli se po stažení aplikace vytvoří identifikační číslo jeho instalace. Po té, co se přihlásí přes mobilní aplikaci k účtu na webové aplikaci, uloží se identifikační číslo k jeho profilu.

Pokud webová aplikace vyhodnotí, že je potřeba poslat notifikaci, odešle HTTP dotaz na server, kde v těle dotazu je uložen identifikační číslo uživatele a obsah notifikace. Server následně odešle danou notifikaci na daný mobilní telefon.

#### 5.1.1 Uložení Firebase Cloud Messaging Tokenu

Při instalaci zařízení se zařízení zaregistrouje ve Firebase a získá Firebase Cloud Messaging Token (FCM token), ten se využije pro následnou identifikaci při posílání notifikace.

Abychom mohli propojit uživatele s jeho mobilním zařízením, musí se nejprve přihlásit. Přihlašovací údaje se pošlou na server pomocí mutace v GraphQL, server vrátí JSON Web Token, který vložíme do hlavičky dotazu. Díky tomu jsou všechny naše dotazy na server autentizované. Následně uložíme příslušný FCM token k uživateli na server.

Pokud server vyhodnotí, že je potřeba poslat notifikaci, vezme token zařízení a pošle na server Firebase HTTP volání s posláním notifikace na dané zařízení. Komunikaci mezi mobilním zařízením a serverem ukazuje obrázek 5.1.



Obrázek 5.1: Znázornění komunikace server-mobilní zařízení

### 5.1.2 Posílání notifikací serverem

Pokud server vyhodnotí, že je potřeba poslat notifikaci vytvoří HTTP dotaz, který pošle na REST aplikační rozhraní Firebase serveru. Posílá se pomocí dotazu POST na adresu <https://fcm.googleapis.com/fcm/send>. Do hlavičky je potřeba vložit klíč aplikace, který se vygeneruje ve Firebase.

Tělo dotazu je ve formátu JSON a obsahuje cíl dotazu, prioritu, objekt notifikace, dobu trvání, nebo lze přes něj poslat i data. Do cíle dotazu se zadá FCM token zařízení, který je uložen na serveru. Lze poslat i hromadné notifikace, a to použitím skupin tokenů. Priorita popisuje, jak je notifikace důležitá. U zpráv s normální prioritou může aplikace přijmout zprávu s neurčeným zpožděním. Normální priorita optimalizuje spotřebu baterie klientské aplikace, proto se používá pokud nepotřebujeme doručit okamžitě. Když je zpráva odeslána s vysokou prioritou, je odeslána okamžitě a aplikace zobrazí dané oznámení. Doba trvání specifikuje jak dlouho je notifikace aktivní. Defaultně je nastavena na čtyři týdny.

```
{  
    "to": "f88Kf9X6...",  
    "priority": "high",  
    "notification": {  
        "title": "Bzzzzzzz",  
        "body": "Uletly vcely",  
        "badge": "1",  
        "sound": "default",  
        "color": "#ffc107",  
    },  
}
```

Výpis 5.1: Ukázka těla dotazu při posílání notifikace

Objekt notifikace určuje samotný obsah notifikace. Titulek a tělo určuje textovou část. Lze při notifikaci přehrát nastavený zvuk, změnit barvu notifikace. Badge určuje, jaké číslo se přičte k zobrazení u ikony aplikace v mobilu.

Při poslání těla 5.1, se na cílovém zařízení zobrazí notifikace, která je zobrazena na obrázku 5.2

Pro snazší práci posílání notifikací ze serveru na mobilní aplikaci jsem k tomu vytvořil dokumentaci, která je přítomna v přiložených souborech.

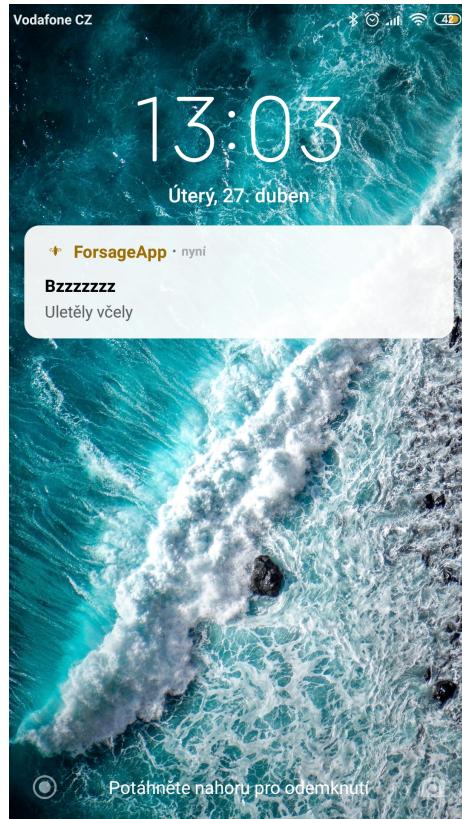
## 5.2 Oboustranná komunikace s Webview

Při zobrazení webové aplikace nemůžeme využít pouze komponent NativeScriptu Webview, protože přes něj se s aplikací nedá komunikovat. Proto využijeme plugin pro oboustrannou komunikaci s Webview.

Potřebujeme zjistit, když se uživatel z webové aplikace odhlásí, aby ho to odhlásilo i z mobilní. Naopak, pro navigaci na stránku se skenováním pomocí Bluetooth, musíme dát webové aplikaci najevo, ať zobrazí v menu i možnost přejít na tuto stránku

### 5.2.1 Přihlášení do webové aplikace

Jelikož kvůli uložení tokenu pro notifikace máme do aplikace vlastní přihlašování, se musí uživatel při otevření aplikace rovnou přihlásit. To zajistíme tak, že do URL adresy webové



Obrázek 5.2: Zobrazení předchozí notifikace na zařízení

aplikace přidáme jeho JSON Web token, který jsme dostali u přihlášení. Tuto adresu<sup>5.2</sup> následně zobrazíme ve WebView.

<https://app.forsage.net/login?jwt=eyJhbGciOiJIUzI1NiJ9...>

Výpis 5.2: Příklad URL adresy

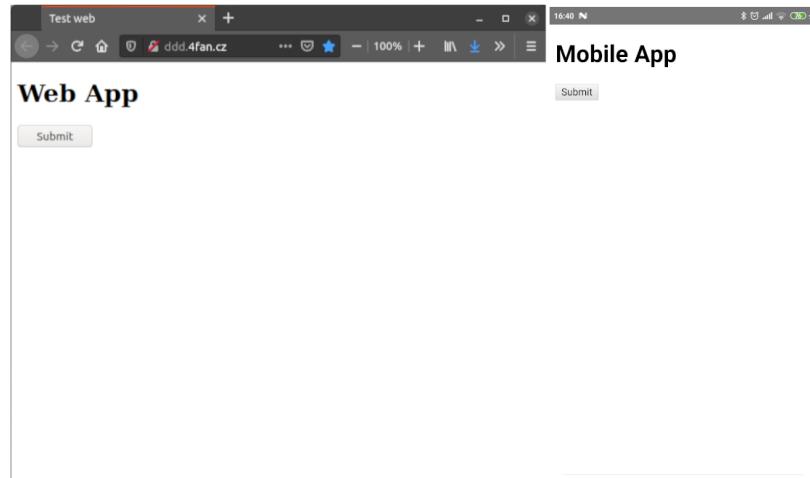
Aby se uživatel nemusel při každém spuštění aplikace přihlašovat, bylo potřeba jeho JSON Web token uložit do lokálního úložiště v zařízení. V NativeScriptu k lokálnímu uložení slouží ApplicationSettings <sup>5.3</sup>, do kterého se dá ukládat jednoduše pomocí hodnoty, definování typu a jména proměnné.

```
appSettings.setString("token", jwt);
```

Výpis 5.3: Uložení tokenu do lokálního uložiště

### 5.2.2 Testovací web

Abych oboustrannou komunikaci nemusel testovat přímo ve webové aplikaci, vytvořil jednoduchý web, na kterém jsem testoval, zdali komunikace funguje. Skládá se z dvou jednoduchých komponentů. Tlačítka, pomocí kterého jsem testoval funkčnost upozornění mobilní



Obrázek 5.3: Rozdíl zobrazení testovacího webu v prohlížeči a mobilní aplikaci

aplikace na stisknutí tlačítka ve webové aplikaci a nadpisu, které by se mělo měnit na základě, jestli je zobrazováno ve webové aplikaci, nebo ve Webview. Zobrazení webu je vidět na obrázku 5.3.

Tento web se skládá ze dvou souborů, které jsou přiloženy v adresáři www.

### 5.2.3 Reakce na tlačítko v Webview

Mobilní aplikace vyčkává, jestli se z Webview nezavolá nějaký spouštěč. Podle názvu spouštěče se spustí následující funkce. V spouštěči lze předávat hodnotu v parametru.

Při stisknutí tlačítka "Odhlásit", by se stalo, že uživatel se odhlásí pouze z webové aplikace, a zůstane přihlášen v mobilní aplikaci. Předcházíme tomu tak, že když se uživatel odhlásí z webové aplikace, pošlu Webview spouštěč logout, který když ho aplikace příjme smaže z lokálního úložiště jeho JSON Web token a vrátí ho na přihlašovací stránku.

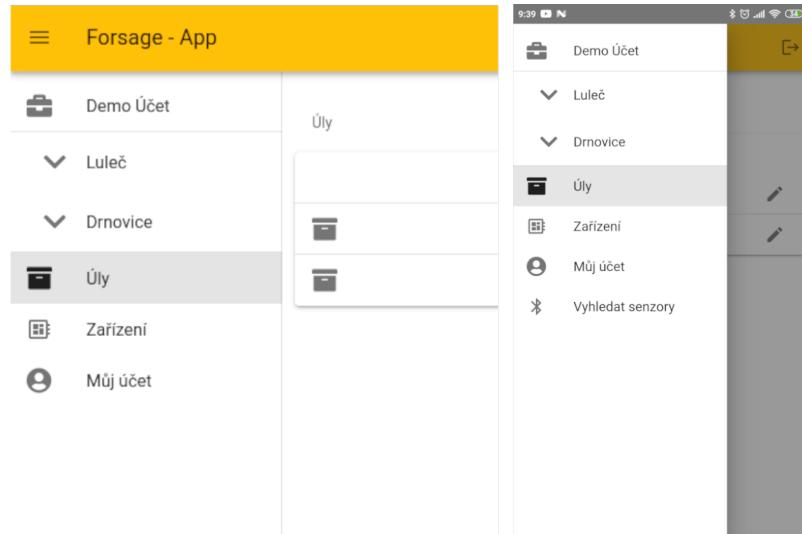
Uživatel může vlastnit více účtů, mezi kterými může přepínat. Klasicky je to demo účet ke kterému přidá později jeho vlastní účet. Každý účet má jiný JSON Web token, proto vždy při změně účtu je potřeba, aby to mobilní aplikace poznala a změnila hodnotu JSON Web tokenu uloženém v zařízení. Kvůli tomu při změně uživatele zavolá webová aplikace spouštěč login, který má v parametru JSON Web token nového účtu. Mobilní aplikace na to zareaguje tak, že změní JSON Web token v lokálním úložišti a aktualizuje Webview.

### 5.2.4 Zobrazení v položky v menu

Aby aplikace byla uživatelsky přívětivá, je potřeba aby obsahovala jen jedno menu. Použil jsem menu webové aplikace, do kterého bylo potřeba přidat přepnutí na stránku skenování pomocí Bluetooth. Bylo potřeba aby tato možnost se zobrazovala pouze v mobilní aplikaci, a ve webové se nezobrazovala. Proto bylo potřeba dát webové aplikace vědět, že se nachází ve WebView mobilní aplikace.

Při spuštění WebView nastavíme hodnotu **User-Agent** na vlastní hodnotu, a to Forsage-App. Když se v mobilní aplikaci spustí zobrazení webové aplikace, zkонтroluje hodnotu **User-Agent** a když odpovídá námi zadané hodnotě, zobrazí v menu možnost přejít na stránku skenování pomocí Bluetooth. **User-Agent** je povinná součást hlavičky a umožňuje serveru identifikovat aplikaci [5]. Webová aplikace při stisknutí této položky v menu pošle

spouštěc "Bluetooth", díky kterému mobilní aplikace pozná, že má přesměrovat na stránku se skenováním. Rozdíl v menu webové a mobilní aplikace je vidět na obrázku 5.4



Obrázek 5.4: Rozdíl menu aplikace v prohlížeči a na mobilní aplikaci

### 5.3 Získání dat ze senzorů

Pro komunikaci se senzorem pomocí Bluetooth Low Energy využívám plugin pro NativeScript. Tento plugin umožňuje skenovat zařízení v okolí a následně se na ně připojit. Mým cílem bylo naskenovat senzory v dosahu a zobrazit o nich informace.

#### 5.3.1 Skenování zařízení

Při vstupu uživatele na stránku, se pošle dotaz na server abychom získali data o senzorech uživatele. Vytvoříme pole objektů. Každý objekt odpovídá senzoru a obsahuje jeho jméno, MAC adresu, jméno úlu, ve kterém se nachází a stanoviště. Zkontroluji jestli je na daném zařízení zapnutá možnost Bluetooth připojení. A následně začne skenovat okolí. Získané identifikační číslo upraví a srovná s MAC adresami uživatelových senzorů.

Pokud zařízení naleze uživatelův senzor, zpracuje z něj údaje. O tom pojednává následující kapitola 5.3.2. K senzoru se není potřeba připojit, stačí pouze senzor naskenovat. Při naskenování získáme jeho identifikační číslo, RSSI - což je hodnot které ukazuje sílu přijímaného signálu. Dále je zde objekt `advertisementData`, ve kterém se nachází `manufacturedData`.

#### 5.3.2 Zpracování dat z Bluetooth advertisements

Když získáme ze senzoru `manufacturedData`, která jsou uloženy v datovém typu `Array Buffer`. Tyto data rozdělíme po bytech. Nejprve převedeme první byte a zkontroluje správnost verze senzoru. Následně převádíme zbytek hodnot tak, že posuneme jeden byte a operací `OR` sjednotíme a předeme podle datového typu.

Z hodnot převedeme teplotu, vlhkost, napětí v baterii podle tabulky 2.1, protože zbytek hodnot je pro nás nevypovídající. Výslednou hodnotu musíme upravit podle toho v jaké

jednotce se vyskytuje. U teploty musíme hodnotu vydělit 200, aby nám vyšla hodnota ve stupních Celsia. Podobně postupujeme i u vlhkosti. U napětí baterie musíme navíc přičíst 1.6 V, protože je to tak definováno[4].

Offset	Popis	Rozsah dat	Formát	Poznámky
0	Verze	5	8 bit unsigned integer	Verze senzoru
1-2	Teplota	-32767 - 32767	16 bit integer	v 0.005 °C
3-4	Vlhkost	0 - 40 000	16 bit unsigned integer	v 0.0025%
5-6	Tlak	0 - 65534	16 bit unsigned integer	v Pa s offsetem -50 000 Pa
7-8	Akcelerace v ose X	-32767 - 32767	16 bit integer	
9-10	Akcelerace v ose Y	-32767 - 32767	16 bit integer	
11-12	Akcelerace v ose Z	-32767 - 32767	16 bit integer	
13-14	Napětí baterie	0 - 2046 0 - 30	11 bit unsigned integer 5 bit unsigned integer	Napětí baterie nad 1.6V v mV Síla signálu nad -40dBm v 2dBm
15	Sčítač pohybů	0 - 254	8 bit unsigned integer	Počet detekcí pohybů
16-17	Pořadové číslo měření	0 - 65534	16 bit unsigned integer	Pořadové číslo měření
18-23	MAC adresa		48 bit	MAC adresa

Tabulka 5.1: Převod hodnot z manufacturedData

Pokud hodnoty zjistíme poprvé, uloží se do pole senzorů, do kterého k nim přidáme i údaje ze serveru. Když daný senzor už v poli je, pouze aktualizujeme jeho hodnoty. Aby hodnoty byly stálé aktuální, voláme každých 5 sekund funkci na skenování okolí.

## 5.4 Grafické rozhraní

Cílem implementace grafického rozhraní bylo předělat návrh uživatelského rozhraní z Figmy 3.5.1. Jelikož jsem nenavrhl všechny obrazovky, vyřešil jsem jednotlivé obrazovky nyní. Grafické rozhraní mělo být co nejvíce podobné rozhraní webové aplikace, aby uživatel nepoznal kdy je ve webové, a kdy v mobilní aplikaci.

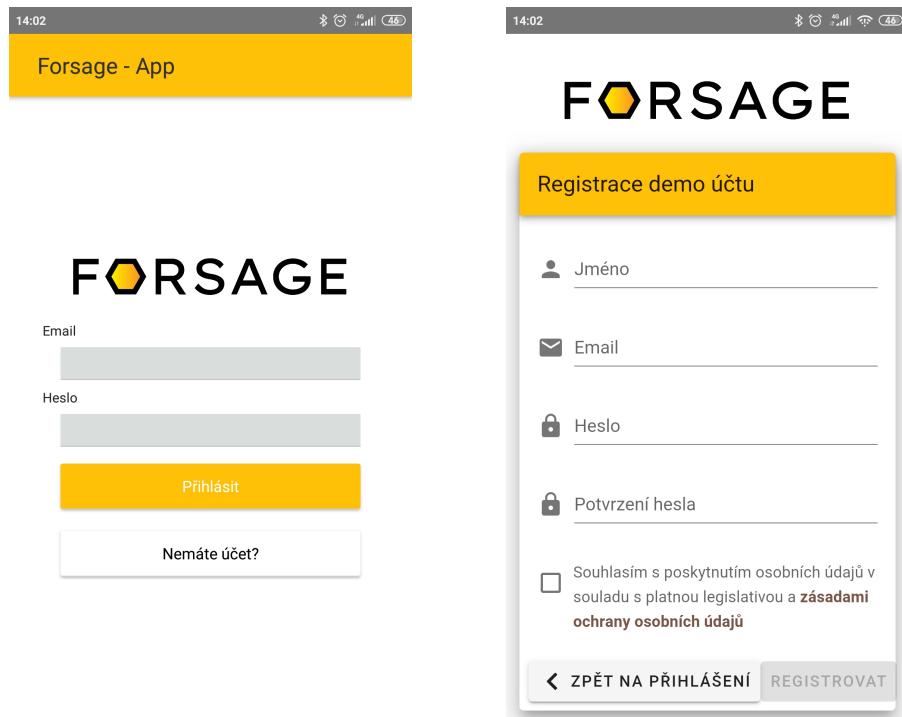
Důležitým aspektem uživatelského rozhraní jsou barvy. Barvy jsem převzal z webové aplikace. Tyto barvy jsou zvoleny podle účelu aplikace do barev včel. Hlavními barvami jsou žlutá, černá a bílá. U některých prvků se výjimečně vyskytuje šedá.

K vytvoření uživatelského rozhraní jsem využil framework NativeScript 4.2 a stylování pomocí CSS.

### 5.4.1 Přihlašovací a registrační stránka

Při spuštění aplikace se zobrazí nativní přihlašovací formulář. DO kterého uživatel zadává svůj email a heslo. V případě že nechá jedno pole prázdné, nebo nemá připojení k internetu, zobrazí se mu dialogové okno, které ho upozorní co je špatně. Součástí této stránky je tlačítko "Nemáte účet", které slouží k přesměrování na registrační stránku.

Stránka s registrací vytvořená pomocí `WebView` a zobrazuje vytvoření účtu ve webové aplikaci. Po registraci získá uživatel přístup k Demo účtu, ve který obsahuje ukázkové data.



Obrázek 5.5: Ukázka stránky přihlášení a registrace

#### 5.4.2 Stránka webové aplikace

Na této stránce je zobrazená webová aplikace pomocí `WebView`. Díky tomu, že webová aplikace má vlastní akční panel, je panel na této stránce skryt. V této stránce se do menu aplikace přidávala možnost přepnutí na stránku skenování senzorů pomocí Bluetooth.

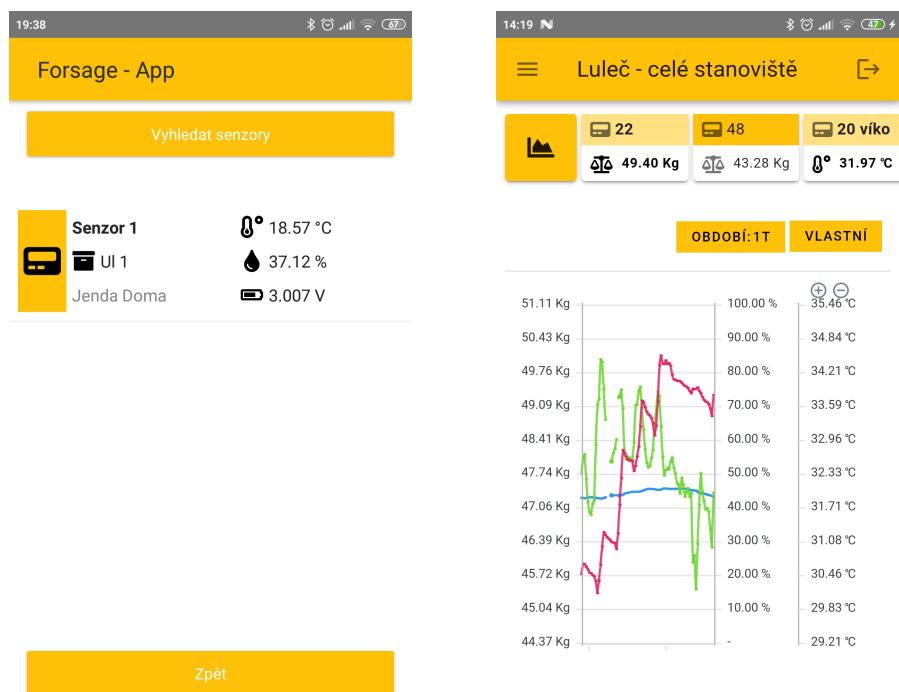
#### 5.4.3 Stránka skenování senzorů

Na stránce skenování senzorů pomocí Bluetooth bylo potřeba, aby se dynamicky přidávali senzory, podle toho jak se budou nacházet. K tomu využíváme komponent NativeScriptu `ListView`, který dokáže zobrazit objekty v poli. Když do pole přibude nový objekt `ListView` ho ihned zobrazí. Pro lepší přehlednost se ke každému senzoru vypíše jméno úlu, ve kterém je a stanoviště na kterém leží. Dále data získaná ze senzoru: teplota, vlhkost a napětí baterie. Způsob zobrazení senzorů ukazuje obrázek 5.6. K hodnotám je přiřazena ikona, která reprezentuje danou položku.

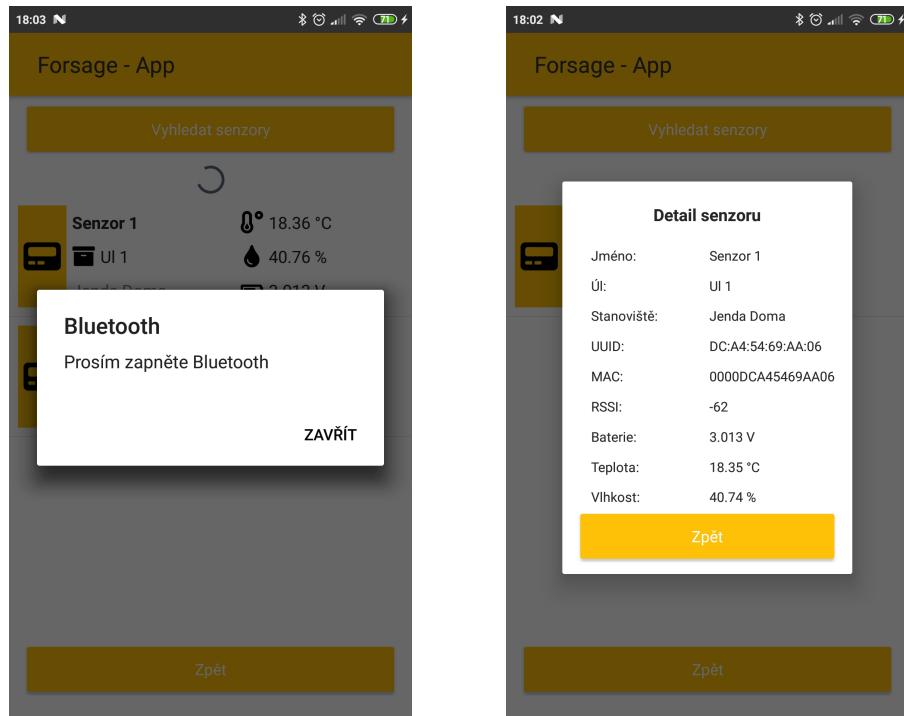
Po kliknutí na daný senzor se zobrazí detaily o něm. Data ze senzorů se aktualizují při každém spojení s senzorem.

#### 5.4.4 Dialogová okna

Pro lepší uživatelské rozhraní jsou součásti grafické aplikace dialogová okna. Tato varianta se vyskytuje u zobrazení detailu senzoru, u upozornění na nezapnutý Bluetooth, nebo u špatných přihlašovacích údajů. Dialogové okno Detailu Senzorů je implementováno jako samostatná stránka, jen je následně zobrazeno jako dialogové. Ostatní dialogová okna jsou zobrazeny pomocí Alert Dialogs, což je jednoduchá a rychlá forma dialogového okna. Ukázka obou dialogových oken je na obrázku 5.7



Obrázek 5.6: Ukázka zobrazení senzoru a zobrazení webové aplikace



Obrázek 5.7: Ukázka rozdílu dialogových oken

# Kapitola 6

## Testování

Testování aplikace je důležitou součástí vývoje mobilní aplikace. Díky testování dokážeme odhalit nejen chyby aplikace, ale i místa, které jsou nevhodně navržena a implementována pro reálné použití.

Testování mělo také za cíl ověřit aplikaci na více různých zařízení. U Android aplikace jsem jí testoval na mobilních zařízení *Samsung Galaxy S3 mini*, *Samsung Galaxy J3*, *Xiaomi Redmi Note 4*, *Xiaomi Redmi Note 7* a *Sony Xperia Z5*. U iOS jsem testoval pouze na *iPhone SE*, protože jsem více zařízení nesehnal. Aplikaci pro iOS jsem otestoval pouze na emulátoru a to na *iPhone 7*, *iPhone 8 plus* a *iPhone X*. Aplikace byla na všech zařízeních funkční. Na iPhone bylo lehce pozměněný stylový přepis. Nebylo potřeba žádné větší úpravy. U aplikací pro iOS nebylo testováno přjímání notifikcí, protože by bylo potřeba aplikaci zaregistrovat, na což nebyli zdroje.

V následující sekci popisují návrh praktických testů, popis jejich provedení a vyhodnocení výsledků.

### 6.1 Praktické testování

Cílem tohoto testování je ověření intuitivnosti a použitelnosti aplikace. Testujeme, zdali je aplikace srozumitelná a dokáží sami vykonat jednotlivé funkční úkony aplikace. V tabulce 6.1 jsou sepsány konkrétní testy, které budou provedeny s uživatelem přímo na včelnici.

K praktickému testování jsem požádal včelařku (dále již uživatel), která vlastní zařízení od Forsage.net. Ta mé žádosti vyhověla a testy byly prováděny u ní na včelnici v katastru obce Luleč. Uživatel má pouze zkušenosti s webovou aplikací Forsage.net. Testy proběhly bez komplikací a vyhodnocení jednotlivých testů je sepsáno v následující sekci.

#### Registrace a přihlášení uživatele

První test proběhl bez problému. Uživatel byl schopen během několika minut sám se zaregistrovat, ověřit na mailu svůj email a přihlásit se.

#### Zobrazení dat ze senzorů

Tento test také proběhl bez problému, uživatel správně začal hledat v menu webové aplikace a dále jej aplikace navedla. Po kliknutí na tlačítko "skenovat zařízení" poznal že má vyčkat dokud se zařízení nenajde. Po zobrazení si hned dovedl spojit dané ikony s vlastnostmi.

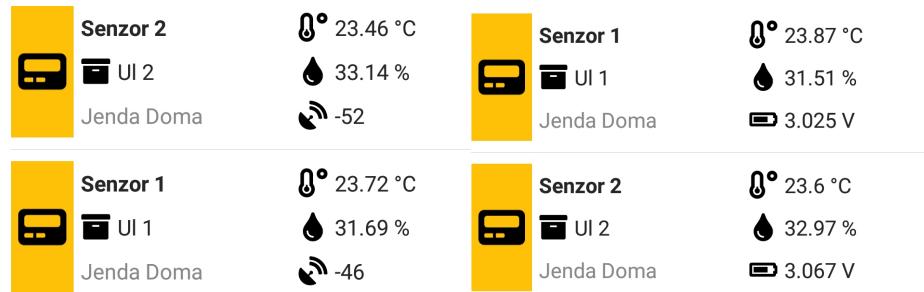
Cíl testu	Způsob testování	Způsob vyhodnocení
Registrace a přihlášení uživatele	Předat mobilní aplikaci uživateli, který ji ještě nepoužíval a má za úkol se registrovat a následně přihlásit do aplikace.	Bylo uživateli hned jasné, jak se registrovat a následně přihlásit?
Zobrazení dat ze senzorů	Uživatel dát za úkol naskenovat data ze senzorů pomocí Bluetooth a zjistit jaká je teplota a vlhkost v úlu. Zároveň ho požádat aby přemýšlel na hlas a popisoval co dělá.	Dokázal uživatel rychle najít možnost zobrazení dat? Poslouchat, u které části se nejvíce zasekl.
Zobrazení detailu senzoru	Uživatel má za úkol zobrazit detailní údaje o senzoru a zjistit jeho MAC adresu.	Dokázal uživatel zobrazit detail senzoru? Chtěl by uživatel zobrazit jiné data na hlavní stránce senzorů?
Zobrazení dat z webové aplikace	Předat uživateli mobilní aplikaci s účtem, na kterém se nachází referenční data, a zadat mu úkol. Má zobrazit váhu úlu ve včelnici za poslední týden.	Dokázal uživatel intuitivně zobrazit data ve webové aplikaci? Dokázal přejít mezi jednotlivými stanovištěmi?
Změna účtu	Předat uživateli mobilní aplikaci s přihlášeným Demo účtem. Nechat ho, ať změní svůj stávající Demo účet na jeho osobní.	Dokázal uživatel intuitivně přejít na jiný účet?

Tabulka 6.1: Návrh testů pro praktické testování

### Zobrazení detailu senzoru

U tohoto testu uživatel intuitivně našel detail senzoru. Při diskuzi, které hodnoty by chtěl vidět na stránce zobrazení senzorů, zhodnotil, že ho více zajímá napětí v baterii, než kvalita přijímaného signálu.

Na základě této zpětná vazby jsem nahradil zobrazení kvality signálu napětím baterie. Rozdíl můžete vidět na obrázku 6.1.



Obrázek 6.1: Rozdíl mezi původním zobrazením a zobrazením po testování

### Zobrazení dat z webové aplikace

Uživatel při tomto testu přešel v aplikaci mezi stanovišti a rychle našel zadání. Krátce ho zdrželo hledání konkrétního úlu, než pochopil, že se mezi nimi přechází tažením. Následovalo rychlé zobrazení hmotnosti za celý týden.

## Změna účtu

U tohoto úkolu se uživatel nejvíce zasekl. Po rozkliknutí menu nejprve hledal přepnutí účtu ve stránce "Můj účet", kde kliknutím na profilový obrázek či jméno se snažil splnit úkol. Posléze již správně našel možnost změny účtu v menu při kliknutí na vlastní jméno.

V důsledku tohoto testu byla přidána ikona ke jménu, díky ní je lépe napovídající, že zde se přepínají účty.

## 6.2 Nalezené chyby

V následující sekci jsou popsány nalezené chyby během testování aplikace.

### Automatické odhlášení z webové aplikace

Během dlouhodobějšího testování se aplikace po delším spuštění odpojila z webové aplikace, avšak v mobilní zůstala přihlášená. Tato chyba byla způsobena vypršením platnosti JSON Web Tokenu po dvanácti hodinách. Tento problém se mi nepodařilo z aplikace odstranit.

### Vyhledávání senzorů

Během testování jsem zjistil, že při opuštění stránky se senzory se dále skenuje okolí. Kvůli tomu měla aplikace větší náročnost na baterii mobilu. Tato chyba byla odstraněna přidáním vypínače skenování při opuštění stránky.

## 6.3 Nápady na vylepšení

V rámci vývoje a testování aplikace, vyvstali nápady na možná vylepšení mobilní aplikace. Tyto nápady jsou zde sepsány a některá jsou naplánovaná na další vývoj.

### Offline přístup

V současném stavu mobilní aplikace funguje pouze, když má připojení k internetu. Je to z důvodu zobrazení webové aplikace a dotazování na server. Skenování senzorů pomocí aplikace by ale šlo i v offline módu. Pro další rozšíření funkčnosti aplikace je potřeba uložit lokálně identifikační číslo a jméno úlu všech vlastněných senzorů, aby při vyhledávání aplikace poznala vlastněné senzory. Pro toto rozšíření je již připraveno uložení dat do lokálního uložiště.

### Komplexní správa včelnice

Ideálním rozšířením aplikace by bylo přidělat částečně automatizovaný včelařský deník. V něm by měl včelař jednak automaticky přidělována data ze senzorů, ale i mohl přidávat vlastní zápisky, jako například jakou barvu má královna, či jak silné včelstvo je. V tomto rozšíření by si včelař mohl založit v aplikaci úly, ve kterých nemá senzory a k nim si vést zápisky. Aplikace by včelaři nabízela podle ročního období návrh záznamů.

Toto rozšíření bude implementováno do webové aplikace a mobilní jej bude pouze zobrazovat.

# Kapitola 7

## Závěr

Cílem této práce bylo vytvořit multiplatformní mobilní aplikaci pro monitoring včelnice, která bude přijímat notifikace, zobrazovat webovou aplikaci a získávat data ze senzorů pomocí Bluetooth. V úvodní části jsou shrnuty možnosti vzdáleného monitoringu včelnice, využití různých senzorů a srovnání existujících řešení.

Další část je zaměřená na návrh mobilní aplikace. Tato část obsahuje případy užití, návrh uživatelského rozhraní a jednotlivých funkčních celků aplikace. Po návrhu následuje část s použitými technologiemi. Na začátku této části jsou rozebrány mobilní platformy a metody vývoje mobilních aplikací. Je zde popsána problematika vyvíjení mobilních aplikací pomocí jazyku JavaScript a frameworku Nativescript, který je v aplikaci používám. Následně jsou zde popsány vybrané technologie a jejich použití.

Nedílnou součástí je implementace navržených částí. V této kapitole je popsán způsob registrace mobilního zařízení pro přijímání Push Notifikací a způsob posílání notifikace ze serveru. Součástí této kapitoly je vytvoření oboustranné komunikace s webovou aplikací a navázání spojení s serverovou částí webové aplikace. Pro lepší otestování funkčnosti byl vytvořen testovací web, na kterém se otestovala oboustranná komunikace, před nasazením do webové aplikace. Úspěšně se podařilo implementovat získávání dat ze senzorů v úlu, které využívá Bluetooth Advertisement k přenosu dat. Dále je zde představena implementace grafického rozhraní aplikace s využitím nativních prvků a dialogových oken. Krátkou ukázkou aplikace je možno shlédnout na <https://youtu.be/EdxFy1i0sow>. Do aplikace se lze přihlásit pomocí vytvořeného účtu ve webové aplikaci, nebo je možné se v aplikaci i zaregistrovat.

Mobilní aplikace byla otestována na několika mobilních zařízení a proběhlo praktické testování, jehož výsledky byly následně zahrnuty do implementace. V této kapitole jsou shrnuty i nápady na budoucí vylepšení.

Podařilo se mi vyvinout mobilní aplikaci v NativeScriptu, která pracuje s různorodými technologiemi. Aplikaci chci publikovat na Google Play i na App Store aby mohla být využívaná včelaři.

# Literatura

- [1] *Use Case Diagram*. 2019. Dostupné z:  
<https://www.smartdraw.com/use-case-diagram/>.
- [2] *Bees as Pollinators: Arkansas Pollinators*. Division of Agriculture - University of Arcansas System, 2020. Dostupné z:  
<https://www.uaex.edu/farm-ranch/special-programs/beekeeping/pollinators.aspx>.
- [3] *Building web apps in WebView*. 2020. Dostupné z:  
<https://developer.android.com/guide/webapps/webview.html#BindingJavaScript>.
- [4] *Data format 5 (RAWv2)*. Ruuvi Developer Documentation, Mar 2020. Dostupné z:  
[https://docs.ruuvi.com/communication/bluetooth-advertisements/data-format-5-rawv2?fbclid=IwAR0l4q2iykX3f0gs\\_ecy-3941vxrNCm7FHE07EXh9RfsFdKC5xWIhMFVTzs#test-vectors](https://docs.ruuvi.com/communication/bluetooth-advertisements/data-format-5-rawv2?fbclid=IwAR0l4q2iykX3f0gs_ecy-3941vxrNCm7FHE07EXh9RfsFdKC5xWIhMFVTzs#test-vectors).
- [5] *Web technology for developers*. 2020. Dostupné z:  
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent>.
- [6] *Documentation / Firebase*. Google, 2021. Dostupné z:  
<https://firebase.google.com/docs>.
- [7] *Empower JavaScript with native APIs*. OpenJS Foundation, 2021. Dostupné z:  
<https://nativescript.org/>.
- [8] *Forsage - Vzdálený monitoring včelstva*. Forsage.net, Mar 2021. Dostupné z:  
<https://www.forsage.net/senzory-do-ulu/>.
- [9] *Mobile Operating System Market Share Worldwide*. StatCounter Global Stats, 2021. Dostupné z: <https://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/#monthly-201208-202101>.
- [10] ANDERSON, N. J. *Getting started with NativeScript: explore the possibility of building truly native, cross-platform mobile applications using your JavaScript skill-NativeScript!* Packt Publishing, 2016. ISBN 978-1785888656.
- [11] APPLE, I. *Xcode*. 2021. Dostupné z: <https://developer.apple.com/xcode/>.
- [12] ECHO24. *Včely byly prohlášeny za nejdůležitější organismus na planetě. Přitom jsou v ohrožení*. Echo24.cz, Oct 2019. Dostupné z: <https://echo24.cz/a/Sytbp/vcely-byly-prohlaseny-za-nejdulezitejsi-organismus-na-planete-pritom-jsou-v-ohrozeni>.
- [13] GRUNA, B., POČUCH, M., PŘIDAL, A. a LSTIBŮREK, J. *Včelařství*. Pracovní společnost nástavkových včelařů CZ, z.s., 2016. ISBN 978-80-270-0776-9.

- [14] LACKO, L. a HERODEK, M. *Vývoj aplikací pro iOS*. Computer Press, 2018. ISBN 978-80-251-4942-3.
- [15] LUBOSLAV, L. *Vývoj aplikaci pro Android*. Computer Press, 2015. ISBN 978-80-251-4347-6.
- [16] PRITCHARD, D. J. a VALLEJO MARÍN, M. Floral vibrations by buzz-pollinating bees achieve higher frequency, velocity and acceleration than flight and defence vibrations. *BioRxiv*. Cold Spring Harbor Laboratory. 2019. DOI: 10.1101/2019.12.17.879981. Dostupné z: <https://www.biorxiv.org/content/early/2019/12/18/2019.12.17.879981>.
- [17] TOWNSEND, K. *Getting started with Bluetooth low energy: tools and techniques for low-power networking*. O'Reilly, 2014. ISBN 978-1491949511.
- [18] VLADIMIROV, R. *The Future of Building NativeScript Apps*. NativeScript, May 2019. Dostupné z:  
<https://blog.nativescript.org/the-future-of-building-nativescript-apps/>.
- [19] VONDRUŠKA, M. *Situace v Čechách a v Evropě*. Včely na střeše, Mar 2013. Dostupné z: <http://www.vcelynastrese.cz/statistika/>.
- [20] YOU, E. *Introduction - Vue.js*. Vue.js, 2021. Dostupné z:  
<https://vuejs.org/v2/guide/>.

## **Příloha A**

### **Obsah přiloženého CD**

CD přiložené k bakalářské práci obsahuje tyto položky: