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 **JAN OSOLSOBĚ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. FRANTIŠEK GRÉZL, Ph.D.**
SUPERVISOR

BRNO 2021

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Osolsobě Jan**
Program: Informační technologie
Název: **Nativní mobilní aplikace pro monitoring včelnice**
Native Mobile Application for Bee Side Monitoring
Kategorie: Informační systémy

Zadání:

1. Seznamte se se senzory používanými pro monitoring včelnice
2. Seznamte se s problematikou návrhu a implementace aplikací na platformě JavaScript.
3. Prostudujte technologií NativeScript pro tvorbu mobilních klientských aplikací přenositelných na různé platformy.
4. Navrhněte architekturu uživatelské aplikace pro správu senzorů na včelnici.
5. Implementujte navrženou aplikaci.
6. Proveďte testování vytvořené aplikace a ověřte přenositelnost řešení.
7. Zhodnoťte dosažené výsledky.

Literatura:

- Žára, O.: JavaScript - Programátorské techniky a webové technologie, Brno: Computer Press, 2015
- Anderson, N. J.: Getting Started with NativeScript. Birmingham: Packt Publishing, 2016

Pro udělení zápočtu za první semestr je požadováno:

- 10 stran technické zprávy
- zprovoznění minimální aplikace virtuální včenlice

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Grézl František, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cílem práce je vytvořit multiplatformní mobilní aplikaci, která pomáhá včelaři vlastnící 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, remote bee side monitoring, mobile application, nativescript, multiplatform programming, 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 Ing. Františka Grézla Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Jan Osolsobě
9. května 2021

Poděkování

Rád bych poděkoval panu doktoru Grézlovi, za vedení práce a dobře mířené rady. Dále bych chtěl poděkovat Filipu Zacharovi za cenné konzultace.

Obsah

Kapitola 1

Úvod

Včely jsou pro naši planetu jedním z nejdůležitějších stvoření. Opylují až osmdesát procent našich pěstovaných plodin [2] a tím poskytují stravu až pro 90 % lidské populace [18]. Včelaření se u nás stává populárním koníčkem i u mladší generace.

Technologie se začínají promítat i do včelaření [17]. Díky vzdálenému monitoringu včelstev dokážeme získávat informace o úlu, i když u něj nejsme. Pomocí změny dat dokážeme předpovídat některé události. Pomocí využití technologií 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 [9], která poskytuje včelaři informace o jeho úlu pomocí úlové váhy a senzoru v úlu. Práce je vytvořena ve spolupráci s touto firmou.

V této práci jsou probrány možnosti vzdáleného monitoringu včelstev a metody vývoje multiplatformní aplikace založené na jazyku JavaScript. Je zde srovnán rozdíl frameworků NativeScript a React Native. Dále je zde popsán návrh aplikace. Je zde popsána implementace, komunikace se serverem a následné 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 ke zjištění aktuální situace v úlu, když včelař navštíví stanoviště. Důležitou součástí aplikace je přijímání notifikací poslaných ze serveru. Díky notifikacím dokáže včelař zareagovat při situaci, která vyžaduje rychlou reakci. Takovou situací může být rojení včel, u kterého kdyby rychle nezareagoval, přišel by o včelstvo. Hlavním cílem je vytvořit uživatelsky přívětivou mobilní aplikaci, která usnadní včelám i včelaři život.

Kapitola 2

Vzdálený monitoring včelnice

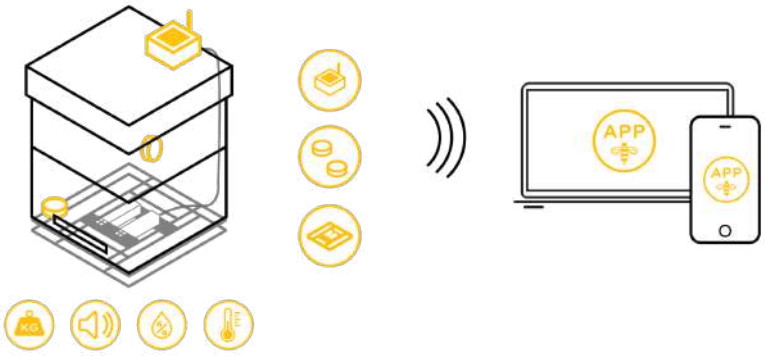
Včelaření je pro mnoho lidí koníček, pro některé i práce. Při vlastnění více úlů na různých místech (stanovištích) musí včelař při kontrole navštívit postupně všechny. Vzdálený monitoring včelnice slouží k tomu, aby včelař nemusel svoje stanoviště s úly navštěvovat tak často a tím může ušetřit čas. Důležité 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. Dále monitoring včelnice umí rozpoznat, pokud se úl převrhne nebo je odcizen, a upozornit na to včelaře.

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) [19]. Tyto vlastnosti ovlivňuje také počasí, proto je potřeba s tím počítat při vyhodnocování. Pomocí analýzy bzučení v úlu se dá poznat, jestli jsou včely ve stresu, což může být důsledkem rojení, nebo nepřítomností matky [22]. 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 hmotnost. Některé měří i teplotu a vlhkost venkovního prostředí. Některé firmy nabízejí i senzor, který se vloží do česna (hlavního vstupu do úlu) a který sčítá přilétající a odlétající včely. Existují i senzory, kteří měří přítomnost oxidu uhličitého, ale není uvedeno, jakým způsobem to na včely působí [14].

Firma Forsage.net nabízí dva druhy senzorů: úlovou váhu a senzor, který se vkládá do úlu. Tyto senzory posílají pomocí Bluetooth data na přenašeč ten je shromažďuje a následně je v pravidelných intervalech posílá na server. Jednotlivé senzory jsou popsány v následujících sekcích. Forsage.net využívá získaná data k následnému výzkumu včel. Schéma senzorů je zobrazena na obrázku 2.1 a jejich reálná podoba na obrázku 2.2.



Obrázek 2.1: Schéma senzorů [9]

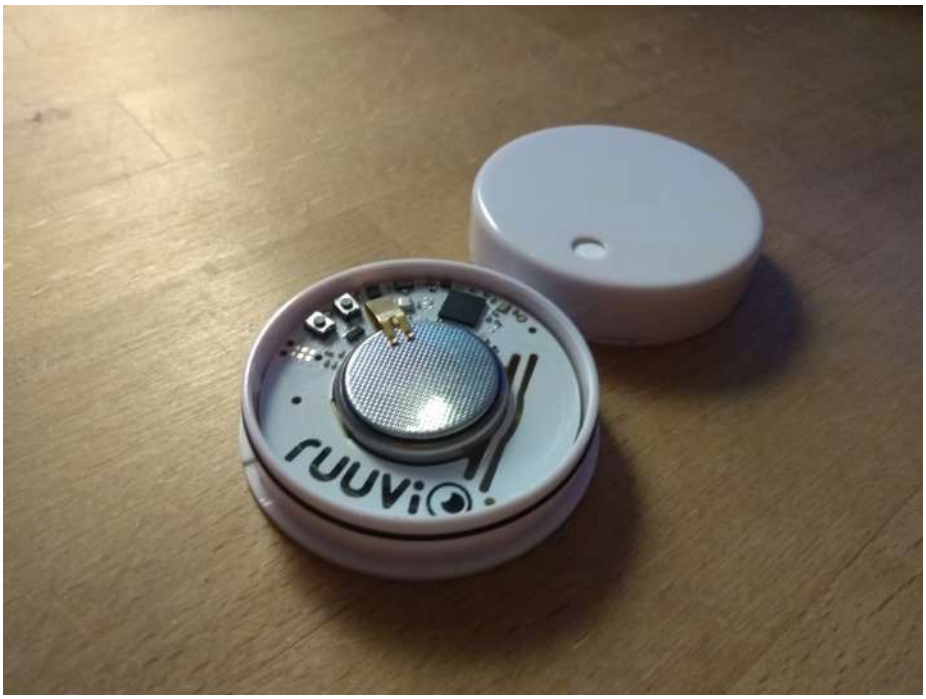
Úlová váha

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. Na váhu se dá umístit libovolná velikost úlu. Váha má maximální nosnost 200 kg a dá se vybavit i venkovním teploměrem. Přesné vážení ale závisí na rovnosti podkladu a pevnosti dna. Váhy čerpají energii z baterie přenašeče. Pokud je do jednoho přenašeče zapojeno více vah, jsou zapojeny sériově.

Obrázek 2.2: Ukázka senzorů [9]

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 mají 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. Včely si senzoru v úlu nevšímají. Díky tomu, že vysílají data pomocí Bluetooth Low Energy, baterie vydrží až 3 roky. Tento senzor je vidět na obrázku 2.3.



Obrázek 2.3: Detail senzoru v úlu

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 ze senzorů sbírá přenašeč 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. Většinou stačí jeden vysílač na jedno stanoviště s úly. Podle počtu zařízení se odvozuje výdrž baterie. Průměrně vydrží jeden rok.

2.2 Existující řešení

V současné době existuje několik řešení monitoringu včelstev. U většiny jsou přístupné zkušební 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 Česka nebo ze Slovenska, protože je tu největší koncentrace včelařů na světě [26].

Firma	Váha	Senzor v úlu	Mobilní aplikace	Webová aplikace	Posílání 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

Vybral jsem i řešení od firmy z USA SolutionBee, které je velmi propracované. V tabulce hodnotím i zdali firma posílá notifikace při událostech na které je potřeba reagovat, jaké senzory daná firma využívá. Dále hodnotím, jak vypadá jejich webová aplikace a jestli mají mobilní aplikaci. Pokud bych se zaměřil pouze na mobilní aplikace, většina firem je nemá. Existující aplikace jsou ve špatném stavu, mají nepřehledné uživatelské rozhraní a jednoduše se v nich dá ztratit.

2.3 Výhody a nevýhody monitoringu včelnice

V této sekci se podíváme na výhody a nevýhody vlastnění vzdáleného monitoringu včelnice.

Výhodou vzdáleného monitoringu včelnice je zobrazení aktuálních údajů o úlu, díky tomu včelař ví, co se v úlu děje. Jde poznat, kdy včely začaly nosit nektar, nebo zda úl vykradlo jiné včelstvo. Dokáže díky tomu plánovat a minimalizovat návštěvy v úlu. Podle údajů z úlu je možné včelaře upozornit nebo mu poradit, co by měl dělat. Největší nevýhodou je cena vybavení pro monitoring včelnice. Ta se pohybuje v řádu několika tisíců korun. Taková cena je pro obyčejného hobby včelaře, který vlastní jen pár úlů vysoká. U včelařů, kteří včelaření mají jako svoje povolání, a mají několik stanovišť daleko od sebe, dokáže vzdálený monitoring včelnice ušetřit hodně času.

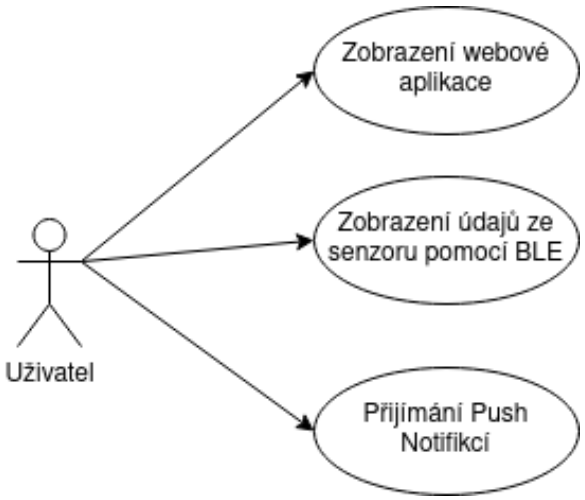
Kapitola 3

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 senzorů pomocí Bluetooth. Následný návrh aplikace, výběr technologií a implementace byla na mém uvážení. V původním návrhu měla aplikace navíc umět ještě konfiguraci zařízení, ale to nakonec firma vyřešila pomocí administrátorského účtu, proto z tohoto návrhu sešlo.

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 provést. Na obrázku 3.1 můžeme vidět, že uživatel může provádět tři hlavní akce a to zobrazení webové aplikace, přijímání Push Notifikací ze serveru webové aplikace a zobrazení dat ze senzorů pomocí Bluetooth.



Obrázek 3.1: Diagram případů užití

Upozornění pro včelaře	Příznaky	Období	Příčina
Úl byl ukraden	Váha klesla na minimum	-	Úl se převrhl, úl někdo ukradl
Zařízení přestalo komunikovat	Nepřichází data ze zařízení	-	Možná ztráta signálu, vybití baterie, zničení zařízení
Baterie v senzoru je skoro vybitá	Zmenšení napětí v baterii	-	Baterie je vybitá, potřeba vyměnit nebo 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 přikrmit včely	Váha úlu se blíží váze prázdného úlu se včelstvem	únor-duben	Včely spotřebovaly 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 nektar
Potřeba přidat nástavek	Váha úlu dosáhne určité hodnoty	duben-červenec	Včely nemají kam ukládat pyl nebo 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

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 reagovat, jsou sepsány v tabulce 3.1 i s příznaky, jak situaci poznat, obdobím, ve kterém se daná situace může vyskytovat a příčinou situace. Pro posílání a přijímání notifikací využijeme službu Firebase, která je detailněji popsána v sekci 4.5.4.

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 platforem. Mobilní aplikace, ale musí s webovou komunikovat, aby se například při odhlášení z webové aplikace uživatel automatiky odhlásil i z té mobilní. Mobilní aplikace také musí reagovat při změně uživatele.

Pro tuto část aplikace nemůžeme využít standardní komponent NativeScriptu WebView, ale musíme použít plugin NativeScriptu pro oboustrannou komunikaci, který je popsán v sekci 4.5.6.

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, jaká teplota je v úlu, načte si data ze senzoru uvnitř. Aplikace získá seznam senzorů uživatele ze serveru webové aplikace a následně zobrazí data ze senzorů a data o senzorech (úl, stanoviště, atd.).

Senzory vysílají data pomocí Bluetooth Low Energy, proto musíme při implementaci využít plugin, pomocí něhož dokážeme se senzory komunikovat.

3.5 Komunikace se serverem webové aplikace

Při přihlášení do aplikace, registrování aplikace pro Push Notifikace a získání seznamu senzorů uživatele budeme muset komunikovat se serverovou částí webové aplikace. Tento server využívá aplikační rozhraní GraphQL, které je blíže popsáno v sekci 4.5.5. Pro práci s tímto rozhraním v Nativesriptu využijeme Apollo GraphQL.

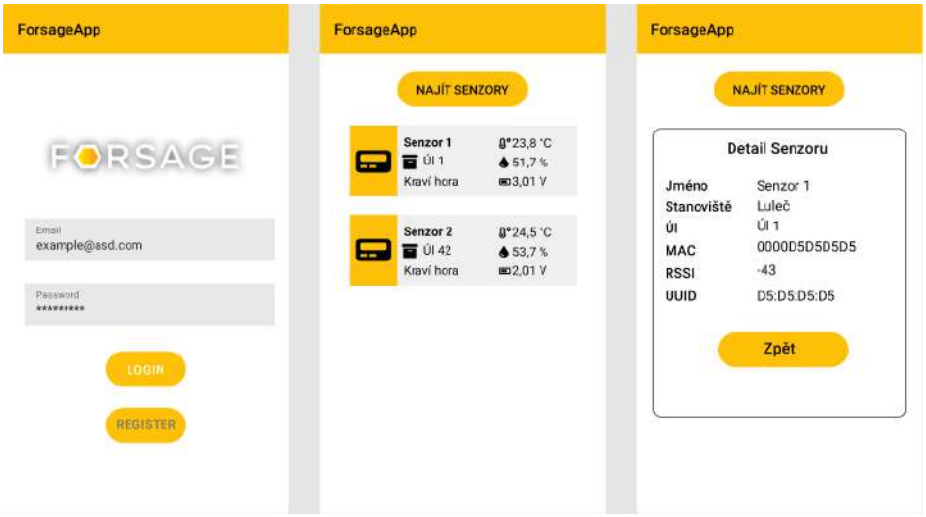
3.6 Grafické rozhraní

Po vyjasnění toho, 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 aplikace nebo v aplikaci samotné. Proto jsem se snažil volit grafické rozhraní podobné webové aplikaci.

3.6.1 Návrh uživatelského rozhraní

Pro návrh grafického rozhraní jsem využil webovou aplikaci Figma [11]. Figma pracuje s vektorovou grafikou a používá se k návrhu prototypů. Návrh je občas poměrně zdlouhavý, ale jde pak využít, protože Figma generuje CSS jednotlivých prvků.

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.

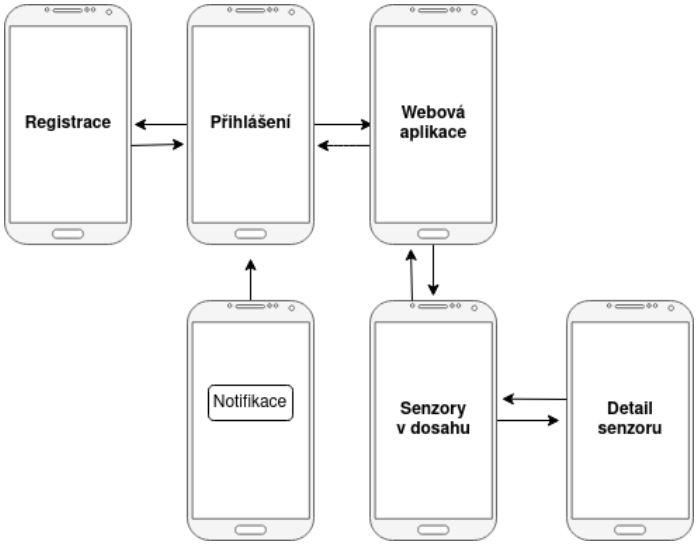


Obrázek 3.2: Grafické návrhy částí aplikace

3.6.2 Vztahy mezi obrazovkami

Pro lepší představu jsem vytvořil obrázek 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: Zobrazení přechodů mezi jednotlivými stránkami

Kapitola 4

Vývoj mobilních aplikací

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 multiplatformě, v této sekci se podíváme na jednotlivé mobilní platformy pro které budeme aplikaci vyvíjet. Zhodnotíme si různé způsoby vývoje aplikace a srovnáme frameworky Nativescript a ReactNative, které se dají využít při vývoji mutiplatformní aplikace, která je interpretována za běhu a staví na jazyku JavaScript.

4.1.1 Mobilní platformy

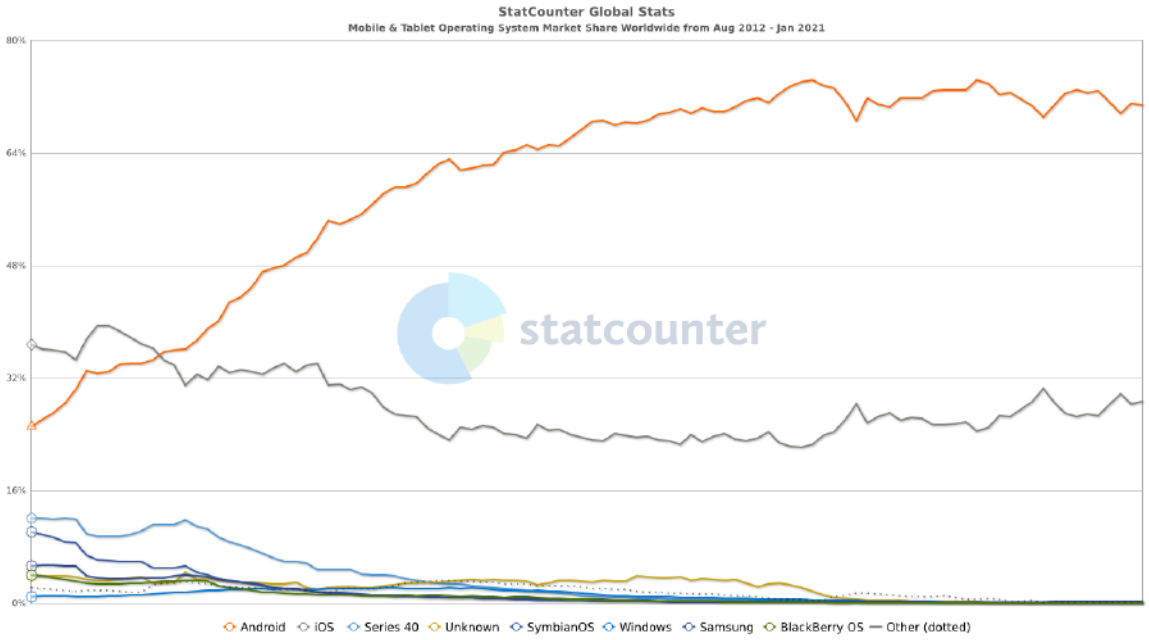
V současné době jsou na trhu mobilní telefony hlavně s dvěma platformami. Je to Android od společnosti Google a iOS, kterou vyvíjí Apple. Mezi další mobilní platformy patří Windows Phone, BlackBerry OS, nebo některé mobilní telefony běží na systémech založených na Linuxu. Tyto platformy se vyskytují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

Android

Android je nejrozšířenější mobilní platforma. Vývoj vede organizace Open Hands Alliance, která spadá pod společnost Google. Android je založen na linuxovém jádře [21], 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.

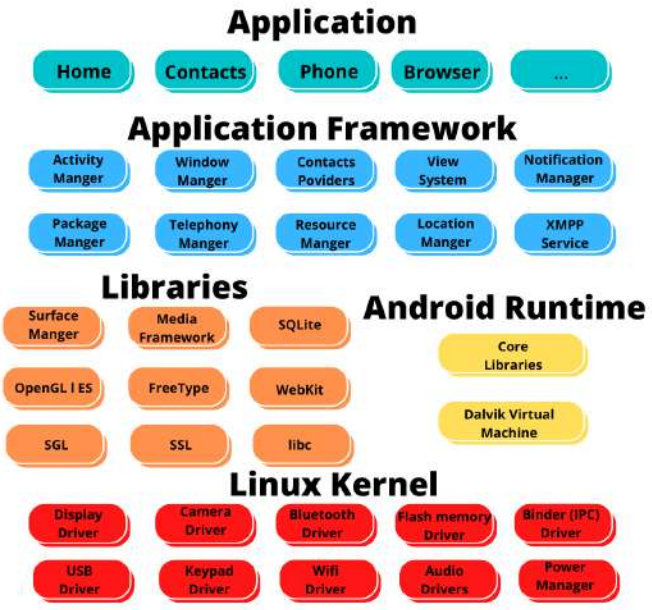
Společnost Android Inc. vznikla v roce 2003, v roce 2007 jí odkoupila společnost Google a následně založila konsorcium Open Hands Alliance, které zahrnovalo další společnosti, které působily v oblasti mobilních technologií. V roce 2008 byl uveden na trh první mobilní telefon s operačním systémem Android.

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.



Obrázek 4.1: Vývoj poměru zastoupení mobilních platforem 2012–2021 [12]

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.



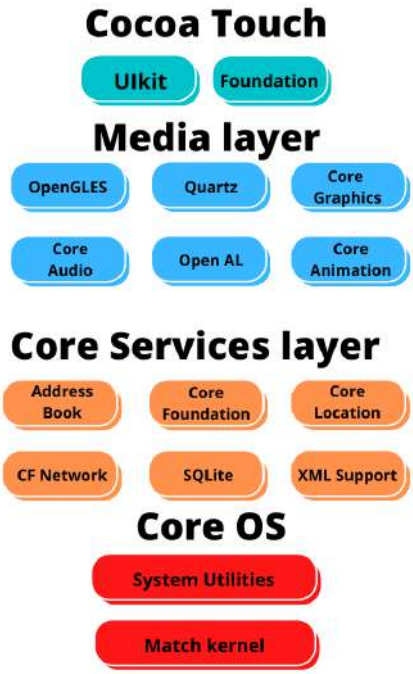
Obrázek 4.2: Struktura operačního systému Android

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 společnost 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[20]. 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.



Obrázek 4.3: Struktura operačního systému iOS

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

4.1.2 Přístupy vývoje 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. V následující části si srovnáme různé přístupy vývoje mobilní aplikace.

Nativní vývoj 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 funkce a nativní prvky uživatelského rozhraní specifické pro danou platformu. Využívají se také u složitějších aplikací, například u těch, které využívají 3D grafiku.

Nevýhodou je nepřenositelnost řešení na více platforem. To je hlavním důvodem vzniku nových přístupů při vývoji mobilních aplikací.

Multiplatformní vývoj aplikace založená na webových technologiích

Tato metoda je nejjednoduší pro multiplatformní vývoj. Princip této metody je, že aplikace je v podstatě okno prohlížeče. Vývojář proto používá stejné technologie jako při vývoji webu a to JavaScript, HTML a CSS. Pro tento vývoj jsou uzpůsobeny frameworky Ionic nebo PhoneGap.

Výhodou je snadný vývoj a testování za pomocí webového prohlížeče a jelikož využívají webové technologie jsou jednodušší na naučení.

Nevýhodou je větší náročnost na výkon telefonu a také nelze využít nativních prvků mobilních platforem.

Multiplatformní vývoj aplikace překládané do nativního kódu

Pomocí tohoto vývoje aplikací jsou aplikace napsány v jednom programovacím jazyku a následně jsou přeloženy pro jednotlivé platformy. Příkladem pro vývoj v této technologii je platforma Xamarin.

Velkou výhodou je, že aplikace vyvinuté tímto způsobem jsou skutečně nativní a využívají všechny prvky uživatelského rozhraní jednotlivých mobilních platforem. Aplikace mohou být díky tomu výkonné a zároveň málo náročné na výkon telefonu.

Nevýhodou je, že při vývoji je při každé změně potřeba celou aplikaci přeložit. Také, že nelze všechen zdrojový kód sdílet mezi jednotlivými platformami. A je potřeba využít jazyky jednotlivých platforem.

Multiplatformní vývoj aplikace překládané za běhu

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 [15].

Výhodou je, že aplikace překládané za běhu mohou zobrazovat nativní prvky mobilních platforem a není díky tomu potřeba zasahovat do úprav uživatelského prostředí jednotlivých platforem. Další výhodou je, že při vývoji aplikace při změně části kódu není potřeba

překládat celou aplikaci, ale přeloží se pouze změněná část. To je způsobeno povahou interpretovaného jazyka JavaScript.

Nevýhodou je, že technologie pro tento vývoj jsou stále nové, a proto nejsou vyladěny všechny detaily. Další nevýhodou je pomalejší start při spuštění aplikace a větší nároky na výkon.

Mezi nejznámější frameworky, které takto pracují jsou NativeScript a React Native. Tyto technologie si představíme a následně v části 4.4 srovnáme.

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 [8].

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 [16]. 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 [25], 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 vyvijí společnost Google a je navržen tak, aby pomohl oddělit zobrazovací a aplikační logiku. Frameworku Vue.js se věnuje část 4.5.2.

Výhodou NativeScriptu je, že se dá zkombinovat s dalšími frameworky a nadstavbami JavaScriptu. Další výhodou je, že při stylování používá CSS, popřípadě jeho nadstavby, SASS nebo LESS. Velkou výhodou je, že existuje velké množství pluginů, které rozšiřují funkčnost NativeScriptu.

Nevýhodou je, že je stále ve vývoji, a proto dokumentace neobsahuje detailnější popis problémů a složitějších použití.

4.3 React Native

React Native je framework, který pomocí HTML, JavaSriptu a frameworku React, uzpůsobený pro vývoj multiplatformních aplikací. Poprvé se objevil v roce 2013 na mezinárodním Facebook hackatonu a následně byl v roce 2015 vydán firmou Facebook [23].

Z počátku byl vývoj možný pouze pro mobilní platformu iOS, ale následně se přidala podpora pro Android a Windows Phone. Vývoj je možný na všech desktopových platformách, ale u vývoje na platformu iOS je stejně jako u Nativescriptu potřeba XCode a tím i platforma Mac. Díky vývojovému prostředí Expo je možné aplikaci přeložit v cloudu a tím vyvíjet pro platformu iOS i ve Windows nebo v Linux.

React Native využívá pro stylování aplikace vlastní přístup, který vychází z principů CSS. Syntaxe je trochu odlišná a není zde použité klasické stylování pomocí tříd a identifikátorů. V React Native je možné využít různé animační efekty jako posunutí nebo rotace, či je možno využít různých režimů jako například nastavitelnou délku trvání.

Aplikace lze vyvíjet pouze za použití webového framworku React, který je také od společnosti Facebook a nelze využít jiné nadstavby jako u NativeScriptu.

Výhodou je velká popularita a tím velké množství rozšíření a dobře zpracovaná dokumentace včetně mnoha návodů. Další výhodou je možnost vyvíjet aplikace i pro mobilní platformu Windows Phone.

Nevýhodou tohoto frameworku je, že se nedá zkombinovat s dalšími frameworky. Další nevýhodou je složitější učení, kvůli jinému stylování a jinému přístupu. Kvůli pozdějšímu přidání mobilních platforem Android a Windows Phone jsou některé komponenty funkční pouze na některé platformě.

4.4 Srovnání NativeScript a React Native

Tyto frameworky mají mnoho společného, v této sekci srovnáme jejich společné výhody a naopak, co mají rozdílného.

Architektura

Oba frameworky pro vývoj multiplatformních mobilních aplikací toho mají hodně společného. Pracují s podobnou architekturou, která přistupuje k aplikačnímu rozhraní pomocí JavaScriptu, a která je zobrazena na obrázku 4.4. Při kompilaci je aplikace překládaná jako nativní aplikace, kde hlavní část tvoří jádro frameworku. Toto jádro obsahuje základní funkcionalitu a také kontejner pro kód v JavaScriptu. Tento kontejner funguje jako hlavní komunikační prvek mezi nativním a JavaScriptovým kódem. Díky tomu je možné volat funkce v JavaScriptu, funkce frameworku a knihovny třetích stran.

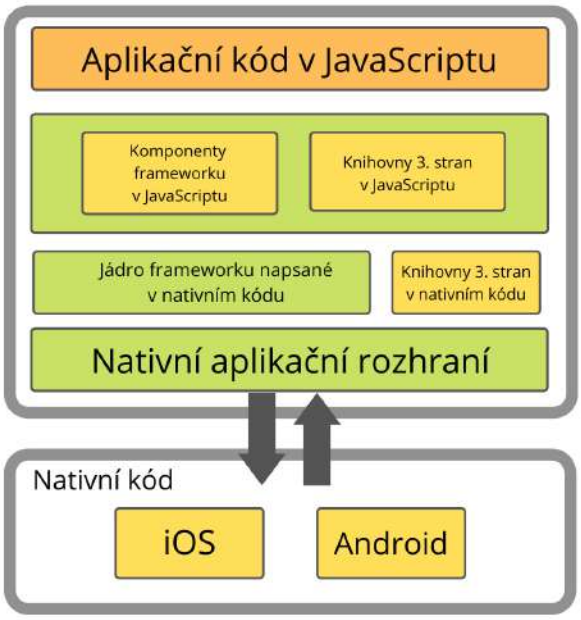
JavaScript je také použit při vývoji velké části frameworků NativeScript a React Native. Za běhu aplikace komunikuje JavaScript s nativním kódem oběma směry. Při volání je požadavek přeložen do nativního kódu a odpověď je zpátky přeložena do JavaScriptu. Výhodou této architektury je, že při vývoji není potřeba překládat celou aplikaci, ale framework za nás rozhodne, kterou část přeloží znovu.

Komunita

Srovnejme komunitu, která frameworky využívá.Větší zastoupení má framework React Native. Díky tomu má větší množství návodů, internetových diskuzí, dostupných komponentů a modulů třetích stran. Nutno podotknout, že NativeScript má také velké zastoupení oproti jiným frameworkům na vývoj mobilních aplikací.

Rozdíly

Výhodou NativeScriptu je, že může využívat i jiné frameworky jako Angular nebo Vue.js, nebo lze využít jazyk JavaScript, nebo jeho nádstavbu TypeScript. U React Native lze použít pouze framework React, který má ale zase velkou popularitu, ze které těží. U stylování aplikace nabízí oba frameworky dobré podmínky. U NativeScriptu je výhodou, že se vývojář nemusí učit nic nového a může používat CSS. Výhodou React Native je, že se na něm dá vyvíjet i pro mobilní platformu Windows Phone. Nicméně tato platforma má čím



Obrázek 4.4: Základní architektura frameworků

dál menší zastoupení. Při vývoji multiplatformní aplikace neuděláte při výběru mezi těmito frameworky chybu. Oba jsou moderní a mají vysoký potenciál. Pro vývoj aplikace jsem si zvolil NativeScript s frameworkem Vue.js.

4.5 Použité technologie

V této části jsou popsány vybrané technologie, které jsou použity při implementaci mobilní aplikace.

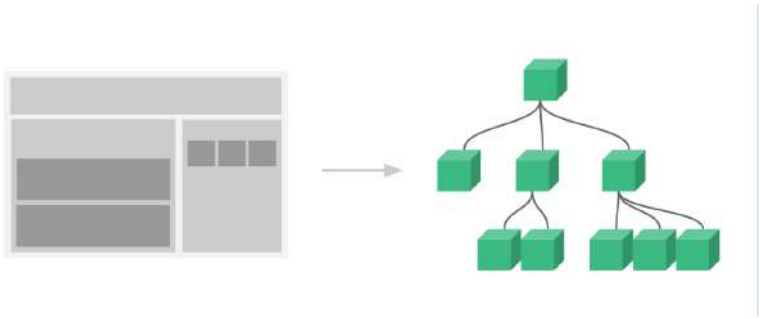
4.5.1 Nativescript

Pro implementaci multiplatformní aplikace byla zvolena technologie Nativescript, která je detailněji popsána v předchozí sekci 4.2, kde je srovnávána s podobným frameworkem React Native. Hlavním důvodem výběru byla možnost využít framework Vue.js, který je popsán v následující sekci.

4.5.2 Vue.js

Při vývoji mobilní aplikace jsem využíval JavaScriptového frameworku Vue.js [27]. Vue.js vytvořil v roce 2013 Evan You, při vývoji se inspiroval jiným JavaScriptovým rámcem AngularJS. Tento framework je progresivní, což znamená, že můžeme aplikaci rozdělit na části. Ty lze 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.



Obrázek 4.5: Dělení aplikace na komponenty [27]

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 4.5.

4.5.3 NativeScript-Vue

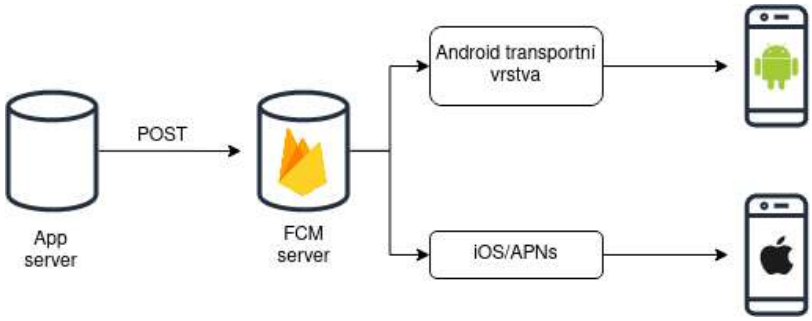
NativeScript-Vue je plugin pro NativeScript, který umožní využívat Vue.js k vytvoření mobilní aplikace. Oproti Vue.js se v některých případech liší v syntaxi, nicméně přebírá veškeré výhody Vue.js.

4.5.4 Firebase

Firebase je platforma vyvinutá společnosti Google [7], 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 zaregistruje 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.



Obrázek 4.6: Diagram toku notifikací

U zařízení s systémem Android se zpráva předá do transportní vrstvy, kde se notifikace zpracuje za pomocí služeb Google Play. U iOS se notifikace zpracovávají pomocí Apple Push Notification service (APNs). Tok notifikací je znázorněn v obrázku 4.6. Pro posílání na iOS je potřeba mít aplikaci registrovanou a certifikovanou v Xcode.

4.5.5 Apollo GraphQl

Apollo je sada nástrojů pro práci s GraphQL [6]. 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 [13]. Využívá se pro komunikací 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 typovaný, proto, když chceme pracovat s nějakou entitou, musíme nejdříve definovat její typ.

GraphQL obsahuje tři základní objektové typy 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.

4.5.6 NativeScript WebView Interface

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 příjimat 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 `iFramu` uloží data, které v druhém kroku data z rámce získá. Díky tomu dokážeme předat jméno události, která se má spustit.

4.5.7 Bluetooth Low Energy

Bluetooth Low Energy je čtvrtá verze Bluetooth, což je standard pro bezdrátovou komunikaci mezi dvěma zařízeními [24]. 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 k chytrým zařízením v domácnosti.

Bluetooth Advertisement

Bluetooth Advertisement neboli Bluetooth reklama je metoda mobilního marketingu [10]. Využívá se k přesunu 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 dobře návrh aplikace. Nyní se můžeme přesunout k samotné implementaci řešení. Implementace je rozdělena podle funkčních celků, které jsou v aplikaci použity. V této kapitole je popsána nejen 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. Poté, co se přihlásí přes mobilní aplikaci k účtu na webové aplikaci, se uživateli uloží k jeho profilu identifikační číslo.

Pokud webová aplikace vyhodnotí, že je potřeba poslat notifikaci, odešle HTTP dotaz na server, kde je v těle dotazu uloženo 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 zaregistruje ve Firebase a získá Firebase Cloud Messaging Token (FCM token), ten se využije pro následnou identifikaci zařízení 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 přes něj lze poslat i data. Do cíle dotazu se zadá FCM token zařízení, který je uložen na serveru. Lze posílat 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 notifikaci 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. Při notifikaci lze přehrát nastavený zvuk či změnit barvu notifikace. Badge určuje, jaké číslo se přičte k zobrazení u ikony aplikace v mobilu.

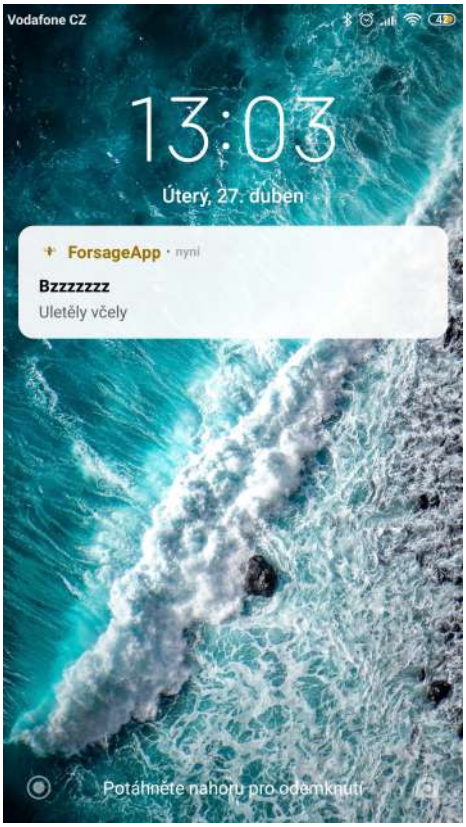
Při poslání těla 5.1 se na cílovém zařízení zobrazí notifikace, která je zobrazena na obrázku 5.2.

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 zajistit, v případě, že se uživatel odhlásí z webové aplikace, aby byl odhlášen 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.



Obrázek 5.2: Zobrazení předchozí notifikace na zařízení

5.2.1 Přihlášení do webové aplikace

Protože kvůli uložení tokenu pro notifikace musí se uživatel při otevření aplikace rovnou přihlásit. To zajistíme tak, že do URL adresy webové aplikace přidáme jeho JSON Web token, který jsme dostali u přihlášení. Tuto adresu 5.2 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 5.3, do kterého se dá ukládat jednoduše pomocí hodnoty, definování typu a jména proměnné.

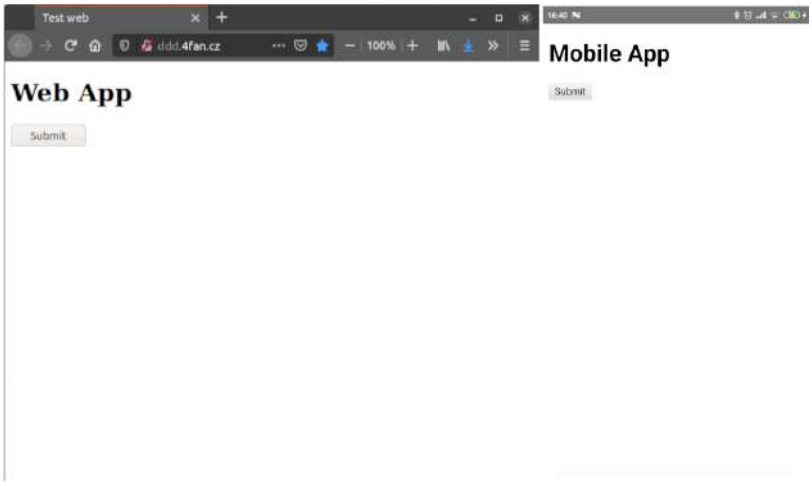
```
appSettings.setString("token", jwt);
```

Výpis 5.3: Uložení tokenu do lokálního uložiště

5.2.2 Testovací web

Aby se oboustranná komunikace nemusela testovat přímo ve webové aplikaci, vytvořil jsem jednoduchý web, na kterém jsem testoval, zda-li komunikace funguje. Skládá se ze dvou

jednoduchých komponent. Tlačítka, pomocí kterého jsem testoval funkčnost upozornění mobilní aplikace na stisknutí tlačítka ve webové aplikaci, a nadpisu, které by se mělo měnit na základě toho, zda je zobrazováno ve webové aplikaci, nebo ve Webview. Zobrazení webu je vidět na obrázku 5.3.



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

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. Ve spouštěči lze předávat hodnotu v parametru.

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

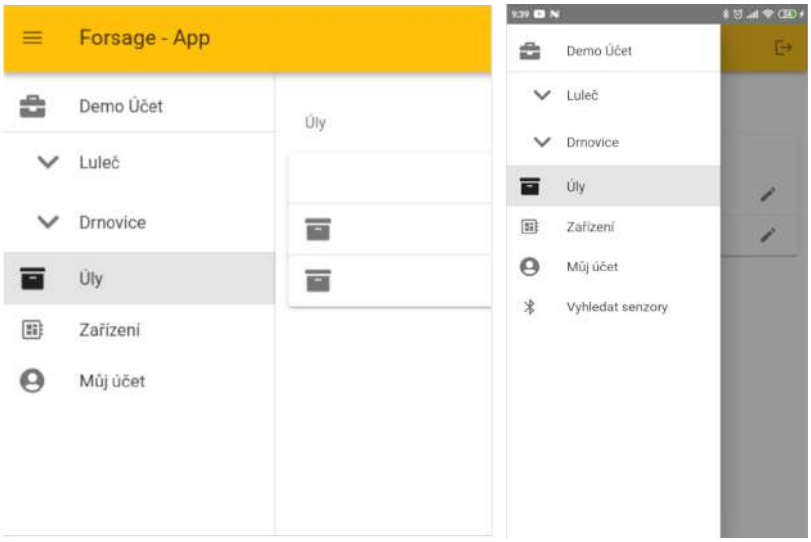
Uživatel může vlastnit více účtů, mezi kterými může přepínat. Klasicky je to demo účet, ke kterému přidá později jeho vlastní účet. Každý účet má jiný JSON Web token, proto vždy při změně účtu je potřeba, aby to mobilní aplikace poznala a změnila hodnotu JSON Web tokenu uloženém v zařízení. Kvůli tomu 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 se tato možnost 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 ForsageApp. Když se v mobilní aplikaci spustí zobrazení webové aplikace, zkontroluje 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

spouštěč “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 včelaře v dosahu a zobrazit o nich informace. Postup při skenování senzorů a zpracování dat popisují následující sekce.

5.3.1 Skenování zařízení

Při vstupu uživatele na stránku, se pošle dotaz na server abychom získali data o senzorech uživatele. Server nám vrátí seznam uživatelových senzorů s jejich jménem, jménem úlu, jménem stanoviště a identifikačním číslem. Z tohoto seznamu vytvoříme pole objektů. Každý objekt odpovídá senzoru a obsahuje jeho jméno, MAC adresu, jméno úlu, ve kterém se nachází a stanoviště. Zkontroluje, jestli je na daném zařízení zapnutá možnost Bluetooth připojení. A následně začne skenovat okolí. Získané identifikační číslo upraví na tvar uložený ve webové aplikaci a srovná s MAC adresami uživatelových senzorů.

Pokud zařízení nalezne uživatelův senzor, zpracuje z něj údaje. O tom pojednává následující část 5.3.2. K senzoru se není potřeba připojit, stačí pouze senzor naskenovat, díky tomu že data jsou uložena v Bluetooth Advertisement. Při naskenování získáme jeho identifikační číslo, RSSI – což je hodnota, které ukazuje sílu přijímaného signálu. Dále je zde objekt `advertismentData`, ve kterém se nachází `manufaturedData`. O zpracování dat z `manufaturedData` pojednává následující sekce.

5.3.2 Zpracování dat z Bluetooth advertisements

Když získáme ze senzoru manufacturedData, která jsou uložena v datovém typu `Array Buffer`. Rozdělíme tyto data po bytech. Nejprve převede první byte a zkontroluje správ-

nost verze senzoru. Následně převádíme zbytek hodnot tak, že podle formátu vezmeme daný počet bitů. U šestnácti-bitového formátu posuneme prvních osm bitů a operací `OR` sjednotíme s dalšími osmi bity. Výsledné číslo převedeme podle datového typu na celé číslo.

Z hodnot převedeme jen teplotu, vlhkost, napětí v baterii podle tabulky 2.1, protože zbytek hodnot je pro nás nevypovídající a v monitoringu včelnice nejsou tak důležité. Výslednou hodnotu musíme upravit podle toho, v jaké jednotce se vyskytuje. U teploty musíme hodnotu vydělit číslem 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, aby se pokrylo rozpětí baterie[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 1,6 V v mV Síla signálu nad -40 dBm v 2 dBm
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 manufaturedData

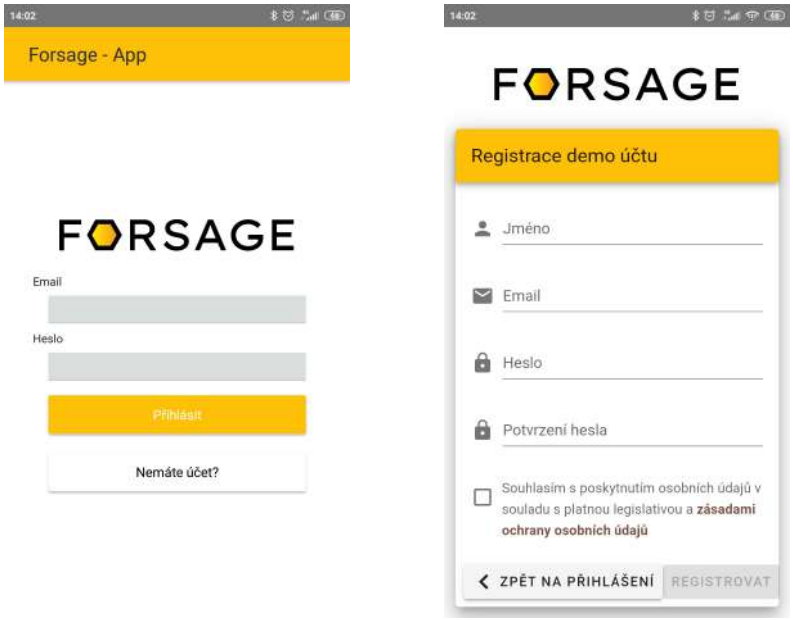
Pokud hodnoty zjistíme poprvé, uloží se do pole senzorů, do kterého k nim přidáme i údaje ze serveru. Když daný senzor už je načtený a uložený v poli, pouze aktualizujeme jeho hodnoty. Aby hodnoty byly stále aktuální, voláme každých 5 sekund funkci na skenování okolí. Grafické zobrazení zjištěných senzorů popisuje sekce 5.4.3.

5.4 Grafické rozhraní

Cílem implementace grafického rozhraní bylo předělat návrh uživatelského rozhraní z Figmy 3.6.1. Při implementaci jsem jednotlivé stránky vytvářel podle návrhů. Grafické rozhraní mělo být co nejvíc 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 a to 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. Implementaci jednotlivých stránek popisuji v následujících sekcích.



Obrázek 5.5: Ukázka stránky přihlášení a registrace

5.4.1 Přihlašovací a registrační stránka

Při spuštění aplikace se zobrazí nativní přihlašovací formulář. Do toho 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 je vytvořená pomocí `WebView` a zobrazuje vytvoření účtu ve webové aplikaci. Během registrace musí uživatel potvrdit registraci na svém emailu. Po registraci získá uživatel přístup k demo účtu, ve kterém obsahuje ukázková data. Když uživatel následně zakoupí zařízení od firmy Forsage.net, zařízení se mu ukážou v jeho účtu aplikace.

5.4.2 Stránka webové aplikace

Na této stránce je zobrazená webová aplikace pomocí komponentu `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. V akčním panelu webové aplikace je tlačítko pro zobrazení menu a tlačítko pro odhlášení z aplikace. Při kliknutí na toto tlačítko se aplikace odhlásí a vrátí na přihlašovací stránku. Na obrázku 5.6 je vidět rozdíl v zobrazení webové a mobilní aplikace.

5.4.3 Stránka skenování senzorů

Na stránce skenování senzorů pomocí Bluetooth bylo potřeba, aby se dynamicky přidávaly senzory podle toho, jak se budou nacházet. K tomu využíváme u komponent NativeScriptu ListView, který dokáže zobrazit objekty v poli. Při kliknutí na tlačítko "Vyhledat senzory"se zobrazí komponent Nativescriptu ActivityIndicator, který slouží k upozornění uživatele, že se na pozadí aplikace něco děje. V našem případě aplikace vyhledává pomocí Bluetooth nové senzory. Tento indikátor aktivity se vypne, jakmile aplikace zhodnotí, že našla všechny možné senzory. Když do pole přibude nový objekt, ListView ho ihned zobrazí. Pro lepší

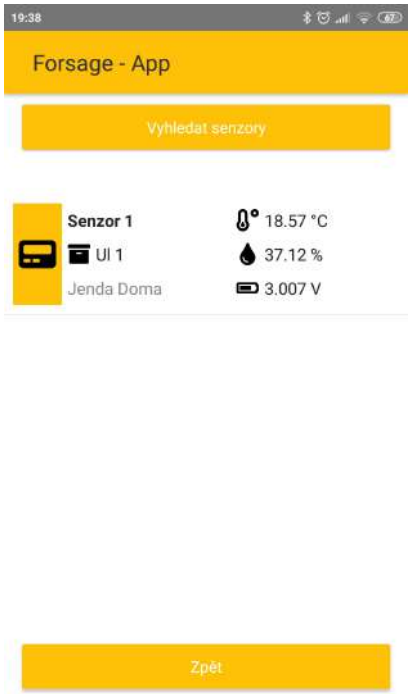
Obrázek 5.6: Ukázka rozdílu webové aplikace v prohlížeči a v aplikaci

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 5.7. K hodnotám je přiřazena ikona, která reprezentuje danou položku.

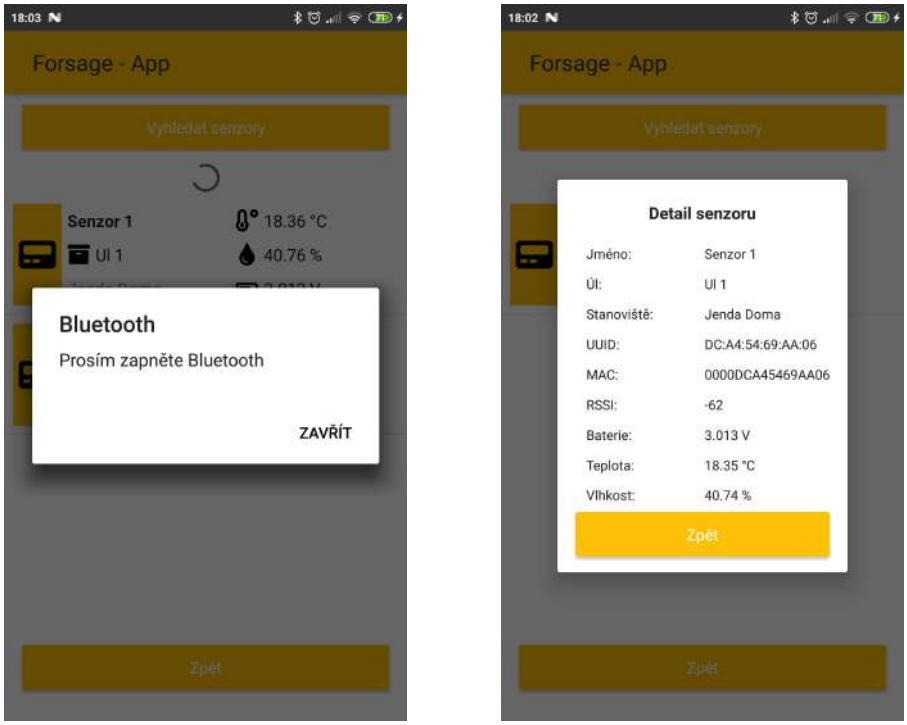
Po kliknutí na daný senzor se zobrazí detaily o něm. Data ze senzorů se aktualizují při každém spojení se senzorem.

5.4.4 Dialogová okna

Pro lepší uživatelské rozhraní jsou součástí grafické aplikace dialogová okna. Tato varianta se vyskytuje u zobrazení detailu senzoru, u upozornění na nezapnutý Bluetooth nebo u špatných přihlašovacích údajích. Dialogové okno Detailu Senzorů je implementováno jako samostatná stránka, jen je následně zobrazeno jako dialogové. Ostatní dialogová okna jsou zobrazena pomocí Alert Dialogs, což je jednoduchá a rychlá forma dialogového okna. Ukázka obou dialogových oken je na obrázku 5.8.



Obrázek 5.7: Ukázka stránky zobrazení senzorů



Obrázek 5.8: Ukázka rozdílu dialogových oken

Kapitola 6

Testování

Pomocí testování dokážeme odhalit nejen chyby aplikace, ale i místa, která jsou nevhodně navržena a implementována pro reálné použití.

Testování mělo také za cíl ověřit aplikaci na více různých zařízení. U Android aplikace jsem ji testoval na těchto mobilních zařízeních:

- Samsung Galaxy S3 mini,
- Samsung Galaxy J3,
- Xiaomi Redmi Note 4,
- Xiaomi Redmi Note 7,
- Sony Xperia Z5

U aplikace pro iOS jsem testoval na následujících zařízení:

- iPhone SE,
- iPhone 7 – emulátor,
- iPhone 8 plus – emulátor,
- iPhone X – emulátor

Většinu jsem musel testovat pomocí emulátoru, protože jsem nesehnal jiné zařízení.

Aplikace byla na všech zařízeních funkční. Na iPhone byly některé komponenty jinak odsazené, ale nebylo potřeba žádné větší úpravy. U aplikací pro iOS nebylo testováno přijímání notifikcí, protože by bylo potřeba aplikaci zaregistrovat v Apple vývojářském programu, na což nebyly zdroje.

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

6.1 Praktické testování

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

Cíl testu	Způsob testování	Způsob vyhodnocení
Registrace a přihlášení uživatele	Předat mobilní aplikaci uživateli, který ji ještě nepoužíval, a dát mu za úkol se registrovat a následně přihlásit do aplikace.	Bylo uživateli hned jasné, jak se registrovat a následně přihlásit?
Zobrazení dat ze senzorů	Uživateli 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? Pozorovat, 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. Uživatel 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

O 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 zkušenosti pouze 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 e–mail a přihlásit se.

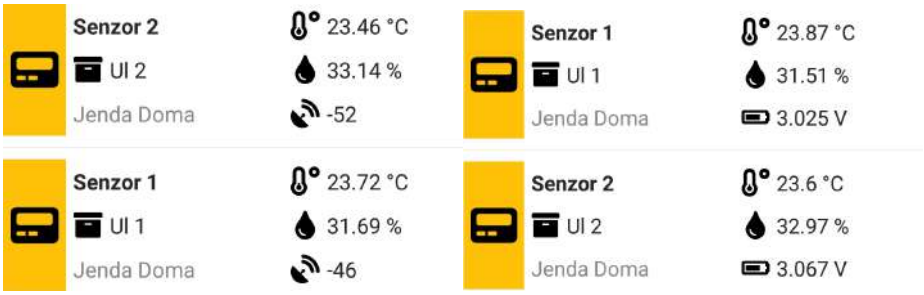
Zobrazení dat ze senzorů

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

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 zpětné vazby jsem nahradil zobrazení kvality signálu napětím baterie. Rozdíl můžete vidět na obrázku 6.1, kde se místo kvality signálu zobrazuje napětí v baterii.



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 zadané. 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 se snažil splnit úkol kliknutím na profilový obrázek či jméno. 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 vyvstaly nápady na možná vylepšení mobilní aplikace. Tyto nápady jsou zde sepsány a některé jsou naplánovany 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 mohl by zde přidávat i 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 ním 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. V práci jsou srovnány frameworky využívající JavaScript, a to NativeScript, který v aplikaci používám a React Native. 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 se 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 ukázku aplikace je možno shlédnout na https://youtu.be/lUiA5uBtqdg. 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 využívána včelaři.

Literatura

[1] *Use Case Diagram* [online]. SmartDraw, 2018 [cit. 2020-12-07]. Dostupné z: https://www.smartdraw.com/use-case-diagram/.

[2] *Bees as Pollinators: Arkansas Pollinators* [online]. Division of Agriculture - University of Arcansas System, 2020 [cit. 2021-05-05]. Dostupné z: https://www.uaex.edu/farm-ranch/special-programs/beekeeping/pollinators.aspx.

[3] *Building web apps in WebView* [online]. Android Developers, 2020 [cit. 2021-05-03]. Dostupné z: https://developer.android.com/guide/webapps/webview.html#BindingJavaScript.

[4] *Data format 5 (RAWv2)* [online]. Ruuvi Developer Documentation, 2020 [cit. 2021-04-04]. Dostupné z: https://docs.ruuvi.com/communication/bluetooth-advertisements/data-format-5-rawv2?fbclid=IwAR0l4q2iykX3fOgs_ecy-394lvxrNCm7FHEO7EXh9RfsFdKC5xWIhMFVTzs#test-vectors.

[5] *Web technology for developers* [online]. Mozzilla, 2020. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent.

[6] *The Apollo Data Graph Platform* [online]. Apollo GraphQL, 2021 [cit. 2021-05-01]. Dostupné z: https://www.apollographql.com/.

[7] *Documentation | Firebase* [online]. Google, 2021 [cit. 2021-05-04]. Dostupné z: https://firebase.google.com/docs.

[8] *Empower JavaScriptwith native APIs* [online]. OpenJS Foundation, 2021 [cit. 2021-03-29]. Dostupné z: https://nativescript.org/.

[9] *Forsage - Vzdálený monitoring včelstva* [online]. Forsage.net, 2021 [cit. 2021-05-04]. Dostupné z: https://www.forsage.net/senzory-do-ulu/.

[10] *Intro to Bluetooth Advertisements* [online]. Bluetooth® Technology Website, 2021 [cit. 2021-05-01]. Dostupné z: https://www.bluetooth.com/bluetooth-resources/intro-to-bluetooth-advertisements/.

[11] *Minds meeting minds is how great ideas meet the world* [online]. Figma, 2021 [cit. 2021-05-01]. Dostupné z: https://www.figma.com/.

[12] *Mobile Operating System Market Share Worldwide* [online]. StatCounter Global Stats, 2021 [cit. 2021-01-15]. Dostupné z: https://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/#monthly-201208-202101.

[13] *A query language for your API* [online]. GraphQL, 2021 [cit. 2021-05-06]. Dostupné z: https://graphql.org/.

[14] *Senzory* [online]. Bee Hive Monitoring, 2021 [cit. 2021-05-06]. Dostupné z: https://www.beehivemonitoring.com/cs/.

[15] 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.

[16] Apple, I. *Xcode* [online]. 2021 [cit. 2021-02-22]. Dostupné z: https://developer.apple.com/xcode/.

[17] Autofakta.cz. *Včely, včelaři a internet věcí: 3 pól - Magazín plný pozitivní energie* [online]. 2016 [cit. 2021-05-01]. Dostupné z: https://www.3pol.cz/cz/rubriky/bez-zarazeni/2016-vcely-vcelari-a-internet-veci.

[18] Echo24. *Včely byly prohlášeny za nejdůležitější organismus na planetě. Přitom jsou v ohrožení* [online]. Echo24.cz, 2019 [cit. 2021-05-01]. Dostupné z: https://echo24.cz/a/Sytbp/vcely-byly-prohlaseny-za-nejdulezitejsi-organismus-na-planete-pritom-jsou-v-ohrozeni.

[19] Gruna, B., POČUCH, M., PŘIDAL, A. a LSTIBŮREK, J. *Včelařství*. Pracovní společnost nástavkových včelařů CZ, z.s., 2016. ISBN 978-80-270-0776-9.

[20] Lacko, L. a Herodek, M. *Vývoj aplikací pro iOS*. Computer Press, 2018. ISBN 978-80-251-4942-3.

[21] Luboslav, L. *Vyvoj aplikaci pro Android*. Computer Press, 2015. ISBN 978-80-251-4347-6.

[22] 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* [online]. 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.

[23] Shoutem. *A brief history of React Native* [online]. React Native Development, 2016 [cit. 2020-12-01]. Dostupné z: https://medium.com/react-native-development/a-brief-history-of-react-native-aae11f4ca39.

[24] Townsend, K. *Getting started with Bluetooth low energy: tools and techniques for low-power networking*. O'Reilly, 2014. ISBN 978-1491949511.

[25] Vladimirov, R. *The Future of Building NativeScript Apps* [online]. NativeScript, 2019 [cit. 2020-11-28]. Dostupné z: https://blog.nativescript.org/the-future-of-building-nativescript-apps/.

[26] Vondruška, M. *Situace v Čechách a v Evropě* [online]. Včely na střeše, 2013 [cit. 2020-11-04]. Dostupné z: http://www.vcelynastrese.cz/statistika/.

[27] You, E. *Introduction - Vue.js* [online]. Vue.js, 2021 [cit. 2021-11-04]. Dostupné z: https://vuejs.org/v2/guide/.

Příloha A

Obsah přiloženého DVD

DVD přiložené k bakalářské práci obsahuje tyto položky:

- `readme.txt` – Soubor ve kterém jsou pokyny, jak nainstalovat a spustit systém.
- `/thesis_pdf` – Adresář s technickou zprávou bakalářské práce ve formátu pdf.
- `/thesis_latex` – Adresář s technickou zprávou bakalářské práce ve formátu LaTeX.
- `/app` – Adresář se zdrojovými kódy mobilní aplikace
- `/www` – Adresář se zdrojovými kódy testovacího webu