

ŽILINSKÁ UNIVERZITA V ŽILINE
FAKULTA ELEKTROTECHNIKY A INFORMAČNÝCH
TECHNOLÓGIÍ

PRÁCA ŠVOS

MARTIN TIMKO

**IoT platforma pre komunikáciu súkromných
dopravných prostriedkov s používateľskou aplikáciou
prostredníctvom cloudu**

Vedúci práce: prof. Ing. Róbert Hudec, PhD.

Registračné číslo:

Žilina, 2021

ŽILINSKÁ UNIVERZITA V ŽILINE
FAKULTA ELEKTROTECHNIKY A INFORMAČNÝCH
TECHNOLÓGIÍ

PRÁCA ŠVOS

ŠTUDIJNÝ ODBOR:

Informatika

MARTIN TIMKO

**IoT platforma pre komunikáciu súkromných
dopravných prostriedkov s používateľskou aplikáciou
prostredníctvom cloudu**

Žilinská univerzita v Žiline

Fakulta elektrotechniky a informačných technológií

Katedra multimédií a informačno-komunikačných technológií

Žilina, 2021

Čestné vyhlásenie

Vyhlasujem, že som zadanú prácu vypracoval samostatne, pod odborným vedením vedúceho práce prof. Ing. Róberta Hudeca, PhD., a používal som len literatúru uvedenú v práci.

V Žiline, dňa 21. februára 2021



.....
Podpis

ABSTRAKT V ŠTÁTNYM JAZYKU

TIMKO, Martin: *IoT platforma pre komunikáciu súkromných dopravných prostriedkov s používateľskou aplikáciou prostredníctvom cloudu*. [ŠVOS práca]. – Žilinská univerzita v Žiline. Fakulta elektrotechniky a informačných technológií; Katedra multimédií a informačno-komunikačných technológií. – prof. Ing. Róbert Hudec, PhD. – Stupeň odbornej kvalifikácie: iný. – Žilina: FEIT UNIZA, 2021. 28s

Práca sa venuje návrhu univerzálnej platformy, ktorá umožní sprostredkovanie informácii z inteligentných osobných dopravných prostriedkov ich majiteľovi, v čase keď sa pri nich bezprostredne nenachádza. Súčasťou je tiež vytvorenie mobilnej aplikácie, ktorá slúži ako klient prístupu k tejto službe. Platforma využíva možnosti moderných technológií, ako napríklad komunikačné siete pre Internet of Things a umiestnenie výpočtových kapacít služby do cloudu, čím je dosiahnutá geografická nezávislosť od lokality, kde sú služby platformy využívané.

Kľúčové slová: IoT, Sigfox, inteligentné vozidlá, elektromobilita, cloud

ABSTRAKT V CUDZOM JAZYKU

The thesis is devoting to designing a universal platform, which will mediate an exchange of information from smart personal transportation vehicles to their owner at a time, when he is not immediately present near them. The part of it is also the creation of a mobile application, that serves as a client to access this service. The platform uses the possibilities of modern technologies, such as communication networks for the Internet of Things and placement of the computing capacity of the service into the cloud, thus achieving an geographical independence from the locality, where the service is being used.

Key words: IoT, Sigfox, Smart vehicles, Electromobility, Cloud

Obsah

Zoznam skratiek a pojmov	6
Úvod	7
1 Telematická jednotka	9
2 Sigfox.....	13
3 Sigfox Cloud a Amazon Web Services	16
4 Android aplikácia	19
5 Firebase Cloud Messaging	21
Záver	23
Zoznam použitej literatúry	24
Zoznam príloh	26
Prílohy.....	27
Príloha A: Snímky obrazovky z používateľskej aplikácie pre Android	28

Zoznam skratiek a pojmov

API	Application Programming Interface
backend	infraštruktúra, ktorá umožňuje poskytovanie nejakej služby
callback	mechanizmus na volanie nejakej entity pri nastaní nejakej udalosti
CAN	Controller Area Network, dátová zbernica
cloud	serverová infraštruktúra v internete poskytujúca služby klientom
cluster	zoskupenie serverov v logickom alebo geografickom zväzku
endpoint	koncový, resp. začiatkový bod služby pre komunikáciu s inou
FaaS	Function-as-a-Service
GPS	Global Positioning System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IoT	Internet of Things
LPWAN	Low-Power Wide-Area Network
MVVM	Model-View-Viewmodel, návrhová architektúra
OS	Operating System
RAM	Random Access Memory
REST	Representational State Transfer
UI	User Interface
UID	Unique identifier
VPN	Virtual Private Network
watchdog	mechanizmus dohliadajúci na funkčnosť zariadenia alebo služby
wi-fi	bezdrôtová komunikačná technológia

Úvod

Je nepopierateľné, že v súčasnej dobe sledujeme v oblasti moderných technológií 2 veľké fenomény – prvým, v oblasti informačných technológií, je mohutné začínanie používania cloudových technológií, teda presun výpočtových aj úložiskových kapacít najrôznejších služieb na serverové farmy vlastnené a prevádzkované spoločnosťami tretích strán, a druhý, prevažne v automobilovom priemysle, postupná zmena dominantného spôsobu pohonu dopravných prostriedkov zo spaľovacích motorov na elektrické motory, ktoré neprodukujú žiadne emisie, poskytujú často lepšie výkonové parametre a v neposlednom rade sú oveľa jednoduchšie. Keďže svet moderných technológií je veľmi pestrý, jednotlivé odvetvia sa navzájom prelínajú, kombinujú a podporujú, za cieľom priniesť náročným používateľom čo najlepší možný zážitok, nemali by sme zabúdať ani na kombináciu spomenutých dvoch fenoménov a možnosti, ktoré sa tým otvárajú. Všetky významné technologické spoločnosti sa dnes snažia pridávať do svojich produktov nejakú pridanú hodnotu, funkcie, ktoré činia zariadenia a služby „chytrými“, ktoré im umožňujú integrovať sa s inými službami či zdrojmi údajov, komunikovať spolu, vyhodnocovať rôzne situácie a reagovať na ne.

Do tohto konceptu perfektne zapadá aj myšlienka ponúknuť osobným dopravným prostriedkom funkcionality jednoducho a efektívne komunikovať s ich majiteľom a poskytnúť mu rôzne informácie o sebe, v dobe keď sú mu užitočné. Medzi tieto dopravné prostriedky môžeme zaradiť nielen osobné autá, ale tiež motocykle, bicykle, či v poslednej dobe veľmi na popularite rastúce elektrické kolobežky. Túto myšlienku ešte viac podporuje zmienený nárast elektromobility, kedy všetky tieto zariadenia prechádzajú na elektrický pohon so zdrojom energie z batérií, ktoré treba dobíjať. Používateľ často necháva svoj dopravný prostriedok nabíjať aj mimo domova, či už na rôznych verejných nabíjačkách na strategických miestach, v práci, alebo napríklad na klasických čerpacích staniách vybavených aj nabíjacími stanicami. Je pritom predpoklad, že majiteľ nieje po celú dobu nabíjania v bezprostrednej blízkosti vozidla, ale môže si ísť napríklad do budovy vypiť kávu. V tejto chvíli nastáva situácia kedy potrebuje vedieť ako nabíjanie pokračuje – koľko času zostáva, alebo aký dojazd bude mať k dispozícii ak by nabíjanie ukončil teraz. Medzi ďalšie možnosti môžeme zaradiť prezeranie si informácií o doterajšej spotrebe, o teplote vo vozidle alebo jeho okolí a následne vzdialené ovládanie klimatizácie alebo vykurovania, upozornenie na nebezpečne vysokú teplotu a súčasnú prítomnosť domáceho zvieratá, či

možno polohu prostriedku v prípade krádeže. Kľúčová požadovaná vlastnosť je jednoduchosť tohto procesu pre používateľa, čo znamená najmä odstránenie/potlačenie problémov s konektivitou zariadení – najmä vozidla. Nemôžeme očakávať, že majiteľ bude po pripojení nabíjačky riešiť ešte pripájanie vozidla na nejakú blízku wi-fi sieť. Taktiež sa žiada zabezpečiť funkčnosť systému aj pri cestovaní do zahraničia. To značne sťažuje aj použité bežných mobilných sietí, pretože by bolo potrebné riešiť medzinárodný roaming a veci s tým spojené. Skvelé riešenie ponúkajú moderné IoT komunikačné siete, z ktorých vybrané vedia fungovať nezávisle od operátora ktorý ich prevádzkuje, sú lacné, spoľahlivé a poskytujú dostatočné pokrytie väčšiny zaľudnených miest v krajinách rozvinutého sveta.

Cieľom práce je porovnať dostupné technológie a na základe týchto požiadaviek vytvoriť systém, ktorý bude možné integrovať do rôzne typov dopravných prostriedkov a ktorý umožní jednoduchý, spoľahlivý prenos želaných informácií do cloudu, kde môžu byť vyhodnotené a uschované, a následne do smartfónu majiteľa a ich prezentovanie prostredníctvom mobilnej aplikácie.

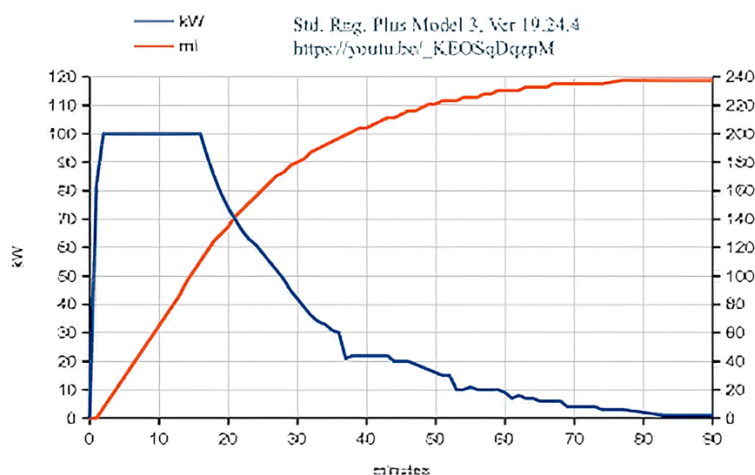
Práca je rozdelená na niekoľko celkov podľa logickej postupnosti, ako vytvorená platforma funguje. Začína prvou kapitolou, ktorá sa venuje telematickej jednotke v dopravnom prostriedku, ktorá slúži na zber údajov o danom prostriedku. Pre účely tejto práce je telematická jednotka nahradená simulátorom postavenom na mikropočítači. Druhá kapitola sa venuje prenosovej technológii Sigfox, ktorá je určená pre prenos údajov z IoT zariadení do cloudového prostredia. V tretej kapitole je rozobraná problematika cloudov, počnúc backendom technológie Sigfox a pokračujúc na verejný cloud AWS na ktorom beží logika služby našej platformy. Štvrtá kapitola je o mobilnej aplikácii, ktorú má používateľ – majiteľ dopravných prostriedkov nainštalovanú vo svojom smartfóne, a ktorá mu prezentuje z nich prenesené údaje. Posledná piata kapitola je o technológií push pomocou ktorej sú údaje proaktívne prenášané z cloudu na zariadenie používateľa.

1 Telematická jednotka

Ako zdroj informácií na strane dopravného prostriedku, ktorá by sa v produkčnom riešení nachádzala integrovaná v danom vozidle od jeho výroby, prípadne by mohla byť dokúpená ako doplnkové príslušenstvo a inštalovaná dodatočne. Takéto riešenie by mohlo dávať zmysel najmä pri jednoduchších zariadeniach ako napríklad e-bicykel, alebo nie-inteligentné dopravné prostriedky, kde by nebolo nutné/možné pripojiť sa na systém daného prostriedku a čerpať informácie z neho. Ako príklad neinteligentného zariadenia môžeme uviesť napríklad klasický bicykel, kde môže riešenie slúžiť na zber a prenos štatistík o priemernej rýchlosti, ktoré je možné získať jednoduchým senzorom rovnakým spôsobom ako bežné tachometre pre bicykle. Druhým údajom by mohla byť poloha, ktorú by bolo možné získať z integrovaného GPS modulu. Celá telematická jednotka by mohla byť inštalovaná ako malá, odolná krabička prichytená na rám vozidla a napájaná prostredníctvom dynamu, prípadne záložnej batérie.

Integrácií do systému zložitejších vozidiel sa táto práca nevenuje, ale je možné predpokladať nutnosť oficiálnej integrácie výrobcom, nakoľko prístup k informáciám z palubného systému moderných automobilov, napríklad prostredníctvom zbernice CAN je veľmi dobre zabezpečený a závislý na systéme konkrétneho modelu vozidla alebo jeho výrobcovi.

Pre účely tejto práce a jej prezentácie sme sa rozhodli vytvoriť simulátor takejto telematickej jednotky, založený na mikropočítači Raspberry Pi. Použili sme jeho najnovšiu verziu 4B, v konfigurácii so 4GB pamäte RAM a s prídavným príslušenstvom v podobe krabičky s dotykovým kapacitným displejom. Výkon tohto mikropočítača je ale oveľa vyšší ako je pre účel takejto telematickej jednotky potrebný a pri návrhu reálnej jednotky by postačoval výkon na úrovni 8bitových mikrokontrolérov a niekoľko kB pamäte. Na nami vytvorenom simulátore beží kód, ktorý simuluje zmenu niekoľkých parametrov v situáciách ako nabíjanie vozidla, jazda, alebo nečinnosť. Medzi tieto parametre patria: aktuálny percentuálny stav nabitia, aktuálne napätie batérie, doterajšia dĺžka trvania aktuálneho stavu, v prípade nabíjania odhad zostávajúceho času a dojazd vozidla pri aktuálnom nabití a spotrebe. Základným údajom pre dopočet ďalších je percentuálny stav nabitia, ktorého hodnota je na základe časového priebehu a zvoleného nabíjacieho prúdu vypočítavaná tak, aby simulovala exponenciálnu krivku priebehu nabíjania reálnych batérií, ktorej príklad môžeme vidieť na obrázku č.1.



Obrázok 1: Ukážka priebehu nabíjania batérie elektromobilu [1]

Niekoľko ďalších parametrov je posielať konštantne, len pre účely ukážky. Vnútrotná teplota vo vozidle je vyčítavaná z tepelného senzora BMP280 pripojeného k Raspberri Pi prostredníctvom zbernice I2C.

Aplikácia simulátora je napísaná v programovacom jazyku C++ a pozostáva z viacerých komponentov, bežiacich v samostatných vláknach: samotný simulátor údajov, komponent “broadcaster“ zabezpečujúci komprimovanie a odosielanie údajov, komponent pre UI a ovládanie simulátora a komponent “watchdog“ sledujúci stav ostatných komponentov, ktorý v prípade ich zlyhania resetuje službu. Aplikácia beží v operačnom systéme Raspberry Pi OS ako služba, čo umožní jej automatické spustenie po naboťovaní zariadenia a prípadné resetovanie v prípade chyby. Simulátor vieme ovládať prostredníctvom príkazového riadku, pričom vieme zmeniť simulovaný stav vozidla, nabíjací prúd a maximálnu požadovanú úroveň nabitia. Cieľom práce je vytvoriť aj jednoduchú grafickú nadstavbu rozhrania pre ovládanie simulátora, založenú na knižnici Qt, ktorá by využila potenciál dotykového displeja ktorým je zariadenie vybavené.

```
[>] State: Charging (with current: 40,0A with target charge: 85%)

[i] Entering main loop.

--- time ----- voltage ----- charge ----- elapsed | remaining ----- range ---
* 16:17:05      665,1V      65,1% of 85%      (0 mins / 3 hours, 32 mins)      325km
* 16:17:10      665,8V      65,8% of 85%      (5 mins / 3 hours, 27 mins)      329km
* 16:17:15      666,5V      66,5% of 85%      (10 mins / 3 hours, 22 mins)      332km
* 16:17:20      667,2V      67,2% of 85%      (15 mins / 3 hours, 17 mins)      335km
* 16:17:25      667,8V      67,8% of 85%      (20 mins / 3 hours, 12 mins)      339km
* 16:17:30      668,4V      68,4% of 85%      (25 mins / 3 hours, 7 mins)       342km
[>] Broadcasting message (at 16:17:33) successful
* 16:17:35      669,1V      69,1% of 85%      (30 mins / 3 hours, 2 mins)       345km
* 16:17:40      669,7V      69,7% of 85%      (35 mins / 2 hours, 57 mins)      348km
* 16:17:45      670,3V      70,3% of 85%      (40 mins / 2 hours, 52 mins)      351km
* 16:17:50      670,8V      70,8% of 85%      (45 mins / 2 hours, 47 mins)      354km
```

Obrázok 2: Ukážka aktuálneho UI simulátora

Nasimulované údaje spolu s aktuálnym stavom vozidla sú v pravidelnom časovom intervale z telematickej jednotky odosielané prostredníctvom IoT technológie Sigfox do centrálnej databázy prevádzkovateľa tejto technológie (viď kapitola č.2), alebo prostredníctvom wi-fi priamo do cloudu, kde sa nachádza backend našej platformy. Interval tohto odosielania nie je konštantný, ale závisí od aktuálneho stavu vozidla – pri nabíjaní je interval častejší ako pri iných stavoch, kedy nie je predpoklad že užívateľ potrebuje čo najaktuálnejšie informácie, a tiež od dostupnej konektivity (wi-fi/sigfox). Dôvod tohto nastavenia bude vysvetlený v ďalšej kapitole. Proces odosielania pozostáva z komprimácie údajov do 2 typov paketov o dĺžkach 12 bajtov, nakoľko toto je maximálna veľkosť jedného dátového paketu, ktorý je možné preniesť pomocou technológie Sigfox [2]. Kompresia je dosiahnutá presne stanoveným bitovým posunom pri jednotlivých premenných s údajmi, nakoľko väčšina z nich nevyužíva celý rozsah hodnôt poskytnutý použitými dátovými typmi, respektíve pri údajoch s reálnym číselným typom sú premenné pretypované na neštandardný dátový typ half [3], ktorý oproti typu float (32 bitov) zaberá len 16 bitov, na úkor zníženej presnosti, ktorá je ale pre naše potreby úplne dostačujúca. Prvý typ paketu (regulárny) sa posiela s väčšou periodicitou ako druhý typ (rozšírený), ktorý obsahuje menej premenlivé údaje. Tabuľku zobrazujúcu konkrétne údaje enkapsulované do oboch typov paketov, spolu s počtom bitov ktoré obsadzujú môžeme vidieť na obrázku č. 3:

Regular message content:

attributes:	length:	data type:	unit:
- state	(3 bits)	enum	-
- current charge	(7 bits)	uint8_t	perc
- target charge	(7 bits)	uint8_t	perc
- current	(10 bits)	uint16_t	amp
- elapsed time	(13 bits)	uint16_t	minutes
- remain. time	(13 bits)	uint16_t	minutes
- current range	(11 bits)	uint16_t	km
- out. temperature	(16 bits)	half	°C
- ins. temperature	(16 bits)	half	°C
	96 bits		
	=12 bytes		

Extended message content:

attributes:	length:	data type:	unit:
- location lat	(26 bits)	float	coordinates
- location long	(26 bits)	float	coordinates
- des. temperature	(16 bits)	half	°C
- max. current	(10 bits)	uint16_t	amps
- elec. consumption	(16 bits)	half	amp
- reserve	(2 bits)	-	-
	96 bits		
	=12 bytes		

Obrázok 3: Rozdelenie prenášaných údajov do paketov

Po pripravení paketu aplikáciu vyhodnotí dostupnú konektivitu zariadenia. Ak je k dispozícii pripojenie na internet prostredníctvom wi-fi (napríklad vozidlo je na oficiálnej nabíjacej stanici výrobcu alebo na stanici partnera, ktorá poskytuje wi-fi sieť na ktorú sa vie dopravný prostriedok automaticky pripojiť), odošle sa paket cez internet priamo na cloud platformy (viď. kapitola č.3). Ak wi-fi nieje dostupné, odošle sa paket na sériový port Raspberry Pi, na ktorom je pripojený Sigfox modul, čo je samostatný modul obsahujúci všetku potrebnú elektroniku na komunikáciu so sieťou Sigfox. Tieto moduly musia byť pre prístup do siete Sigfoxu oficiálne certifikované, čím majú pridelený jedinečný UID kód, ktorým sa v sieti identifikujú [4]. Ich cena na trhu sa pohybuje v rozsahu od 10 do 50eur. Na účely tejto práce sme použili modul LPWAN SigFox node od českého výrobcu LPWAN/cz, ktorý je založený na modeme SFM10R1. Konfigurácia, aj všetka komunikácia s modulom prebieha pomocou jednoduchého protokolu AT príkazov.

2 Sigfox

Sigfox je jednou z moderných komunikačných technológií kategórie LPWAN (Low-Power Wide-Area Network), určených pre využitie v IoT (Internet of Things) svete. Jej kľúčovými vlastnosťami sú extrémne nízka prenosová rýchlosť, len do 600b/s, čo ale vo svete IoT nieje nežiadúce (nepotrebujeme prenášať veľa údajov), na oplátku to ale prináša veľmi nízku energetickú náročnosť, čo je veľkou výhodou najmä v batériovo napájaných zariadeniach. Technológia poskytuje vysokú spoľahlivosť prenosu - každá správa je pri štandardnom nastavení vysielaná 3-krát, na mierne odlišných frekvenciách a s niekoľkosekundovým odstupom. Týmto prístupom sa dosiahne značné zvýšenie pravdepodobnosti, že ju z dôvodu väčších vzdialeností alebo vo viac zarušenom prostredí zachytiť aspoň nejaká základňová stanica v okolí, respektíve že sa správu podarí úspešne doručiť aj ak pri niektorom z vysielaní nastala kolízia s vysielaním iného zariadenia v okolí, čo by inak nebolo možné odhaliť. Základňová stanica a sieť potvrdenie príjmu správy nepotvrďuje, ale je zodpovedná za riešenie duplicit ktoré z princípu vznikajú prijatím rovnakej správy viacnásobne, často aj viacerými základňovými stanicami. Dosah signálu k základňovej stanici je vo voľnej krajine približne do 50km, v zastavanej oblasti cca 10km, pričom pri dobrej sile signálu, alebo v pokrytí viacerých staníc súčasne nie je problémom ani komunikácia z interiérov budov alebo dokonca z podzemných priestorov.

Technológiu vyvíja a prevádzkuje za pomoci partnerov rovnomená, francúzska firma Sigfox, ktorá vlastní patenty na jej komunikačné princípy. Siete samotné sú prevádzkované partnerskými spoločnosťami, pričom platí pravidlo, že v každej krajine prevádzkuje sieť len jeden operátor. Sieť je postavená na single-hop hviezdicovej topológii, kedy každá základňová stanica má (prostredníctvom VPN) priame pripojenie do centrálného Sigfox cloudu, kde sa vykonáva kontrola integrity, autorizácie, zjednotenie duplicit, metrika, filtrovanie a archivovanie, a odkiaľ sú prostredníctvom callbackov správy ďalej distribuované adresátom. Základňové stanice nekomunikujú navzájom.

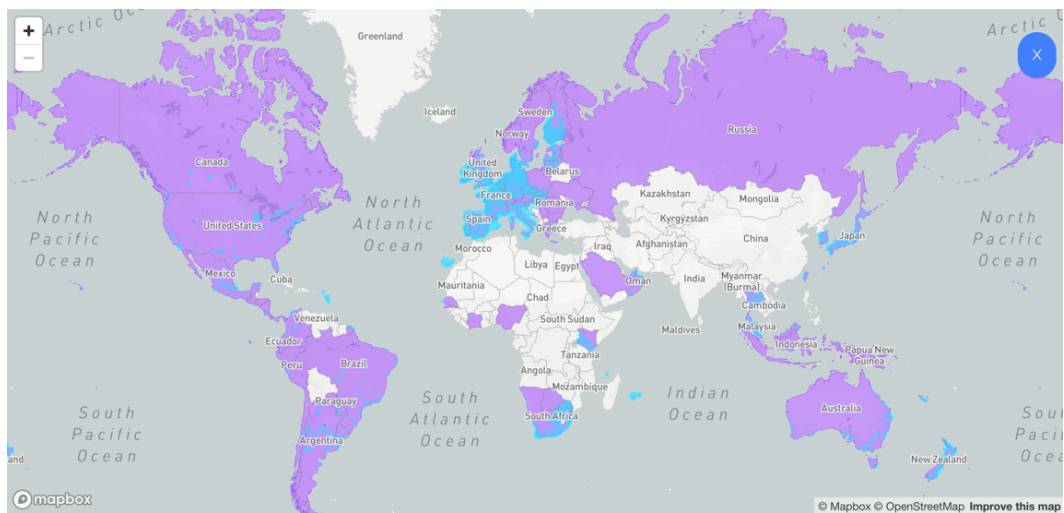
Komunikácia v éteri prebieha vo voľne dostupnom, nelicencovanom frekvenčnom pásme 868MHz (pre Európu), z čoho ale legislatívne vyplývajú isté obmedzenia, vid' nižšie. Vysielanie je zabezpečené šifrovaním AES128, pričom všetky správy šifruje modem automaticky, pomocou nastaveného kľúča. Ako bolo spomenuté už skôr, každý vysielateľ sa v sieti identifikuje vlastným unikátnym UID. Po prijatí správy základňovou stanicou je táto odoslaná na centrálny Sigfox cloud, pričom tento prenos je zabezpečený sadou protokolov

IPsec, ktorá šifruje a autentifikuje každý IP datagram prenášaný sieťou. Táto kombinácia zabezpečení robí technológiu Sigfox veľmi bezpečnou a zneužitie je takmer nemožné. Maximálna dĺžka obsahu jednej správy vo vzostupnom smere (uplink, teda od vysielateľa na základňovú stanicu) je 12 bajtov, pričom hlavička tvorí ďalších 26 bajtov. Sigfox umožňuje aj komunikáciu v zostupnom smere (downlink, teda zo základňovej stanice na klienta), kedy jedna správa môže mať maximálne 8 bajtov, no jej použitie je menej časté.

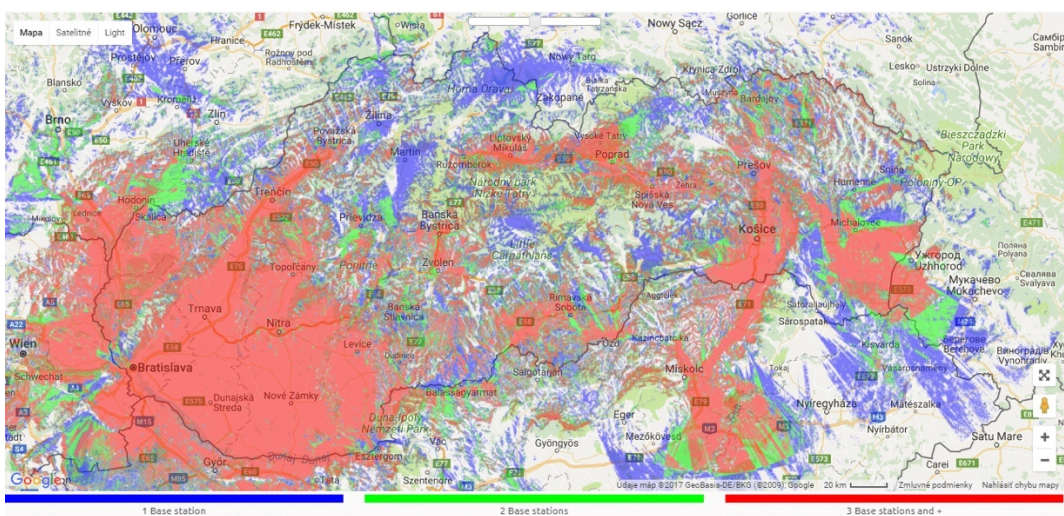
Dôležitým faktorom je spomenuté legislatívne obmedzenie vysielania vo voľnom pásme 868MHz. Európsky inštitút pre telekomunikačné normy stanovuje pre zariadenia využívajúce toto frekvenčné pásmo tzv. pracovný cyklus (duty cycle) vo výške 1%, čo znamená, že zariadenie môže v tomto pásme vysielat' maximálne 1% času. To je 864 sekúnd denne, čo po prepočte s prenosovou rýchlosťou 100b/s, plnou dĺžkou správy a štandardným opakovaním vysielania jednej správy 3-krát to vychádza na odvysielanie do 144 správ denne. Sigfox oficiálne stanovuje limit 140 správ za deň vo vzostupnom smere, a len 4 správy denne v zostupnom, hoci tieto limity nijak aktívne nekontroluje ani neobmedzuje ich prekročenie. Treba podotknúť, že rovnaké obmedzenie sa týka aj iných komunikačných technológií pracujúcich v pásme 868MHz, napríklad LoRa. Nastavenie periodicity vysielania telematickej jednotky prostredníctvom Sigfox siete by teda bolo na zväžení výrobcu a konkrétneho použitia, pričom stále je k dispozícii neobmedzená komunikácia prostredníctvom wi-fi. Vypočítali sme ale doporučenú schému, ktorá rešpektuje potrebu čo najaktuálnejších informácií pri nabíjaní dopravného prostriedku, a počíta s nenutnosťou častého prenosu, ak je prostriedok nečinný, alebo v jazde. Doporučené nastavenie je teda počas nabíjania bez dostupnosti wi-fi pripojenia vysielat' aktualizáciu každých 5 minút, čo činí 20 správ za hodinu, a pri nabíjaní 6 hodín denne mimo domu (dosahu domácej wi-fi siete) by došlo k odoslaniu 120 správ, a okrem toho odvysielat' aktualizáciu okamžite pri každej zmene stavu alebo mimoriadnej udalosti (začiatok/ukončenia nabíjania, upozornenie na prekročenie bezpečnej teploty a prítomnosť osoby/zvierat'a vo vozidle, a podobne). Pri takomto nastavení je nepravdepodobné, aby došlo k prekročeniu požadovaných limitov. Komunikáciu v zostupnom smere vieme vzhľadom na výraznú limitáciu využiť na účel vyžiadania o okamžité zaslanie aktuálneho stavu, napríklad kvôli získaniu aktuálnej informácie o polohe. Jediným spôsobom, ako sa vyhnúť týmto limitáciám by bolo použitie komunikačnej technológie pracujúcej v licencovanom pásme, napríklad NB-IoT alebo LTE-M, ktoré sú však príznačné vyššou zložitnosťou, cenou, nutnosťou vlastniť SIM kartu a komplikovaným roamingom, ktorého

nutnosť je vo vozidle predpokladom. Jednou z najväčších výhod Sigfoxu a aj dôvodom jeho výberu na použitie v tejto práci je jeho schopnosť úplne automatického roamingu, ktorý nie je potrebné nijako konfigurovať a nie je nijako navyše spolplatený. Táto vlastnosť teda zabezpečuje kontinuálnu konektivitu dopravného prostriedku ku cloudu kdekoľvek na svete v dosahu Sigfox siete.

Sigfox aktuálne dosahuje celosvetovo pokrytie 5,8 miliónov km² a 1,3 miliardy obyvateľov, pričom jeho rozširovanie je veľmi rýchle a medziročne stúpa v desiatkach percent. Mapy pokrytia vo svete a na Slovensku môžeme vidieť nižšie, na celosvetovej mape sú fialovou farbou aktuálne potvrdené pokryté územia a modrou farbou krajiny s ktorými spoločnosť rokuje, respektíve už prebieha budovanie sietí, ale mapy pokrytia ešte nie sú k dispozícii. [2][4][5][6][7]



Obrázok 4: Pokrytie Sigfox siete celosvetovo [8]



Obrázok 5: Pokrytie Sigfox siete na Slovensku [9]

3 Sigfox Cloud a Amazon Web Services

Po tom, čo je správa prijatá a spracovaná sieťou Sigfox, sa ocitne v centrálnom cloude spoločnosti Sigfox, ktorý sa nazýva Sigfox Backend. Vo prihlásení vieme vo webovom rozhraní tohto cloudu vieme sledovať množstvo informácií o našich zariadeniach, vytvárať hierarchiu podradených prístupových účtov, rozdeliť zariadenia do skupín a tak ďalej, najdôležitejšou úlohou je ale preposlanie správy ďalej, na cloud prevádzkovateľa konkrétnej služby. To je riešené mechanizmom callbackov, t.j. “spätných“ volaní API cieľového cloudu pri dostupnosti novej správy. V rámci tohto procesu Sigfox Backend umožňuje vykonať dekodovanie prijatého obsahu na jednotlivé údaje, a to aj v prípade ich neštandardného rozloženia do jednotlivých bitov. Následne je z údajov zložená samostatná správa v podobe JSON objektu, do ktorej môžu byť pridané aj ďalšie údaje ako časová značka, ID zariadenia, sila signálu, atď. K tomuto JSON objektu ktorý sa odosiela ako telo HTTP správy sa môžu pripojiť HTTP hlavičky potrebné napríklad k autentifikácii na cieľovom API. HTTP správa je následne (predvolene) POST metódou odoslaná na cieľovú URL adresu. Prenos je zabezpečený SSL/TLS šifrovaním. Podobným princípom je možné vytvoriť aj tzv. bi-directional callbacky, ktoré umožňujú volanie aj z opačnej strany a vyvolajú komunikáciu na zariadenie v zostupnom kanáli. [10]

```
Url pattern https://fe6ea208.eu-gb.apigw.appdomain.cloud/ecar-iot-kit-api/v1/sigfox/uplink
Use HTTP Method POST
Send SNI [x] (Server Name Indication) for SSL/TLS connections
Headers X-Debug-Mode true
X-AWS-Client-Id 74a9df9b-9b32-49e0-8252-860871a18e01
contentHeader {time}
X-AWS-Client-Secret f8c630b5-3acd-456c-991f-da61f350daa4
myheader skuska
header value
Content type application/json
Body
{
  "time": "{time}",
  "deviceID": "{device}",
  "seqNumber": "{seqNumber}",
  "data": "{data}",
  "state": "{customData#state}",
  "current_charge": "{customData#current_charge}",
  "target_Charge": "{customData#target_charge}",
  "current": "{customData#current}",
  "elapsed_time": "{customData#elapsed_time}",
  "remain_time": "{customData#remain_time}",
  "range": "{customData#range}",
  "elec_consumption": "{customData#elec_consumption}",
  "indoor_temp": "{customData#indoor_temp}"
}
```

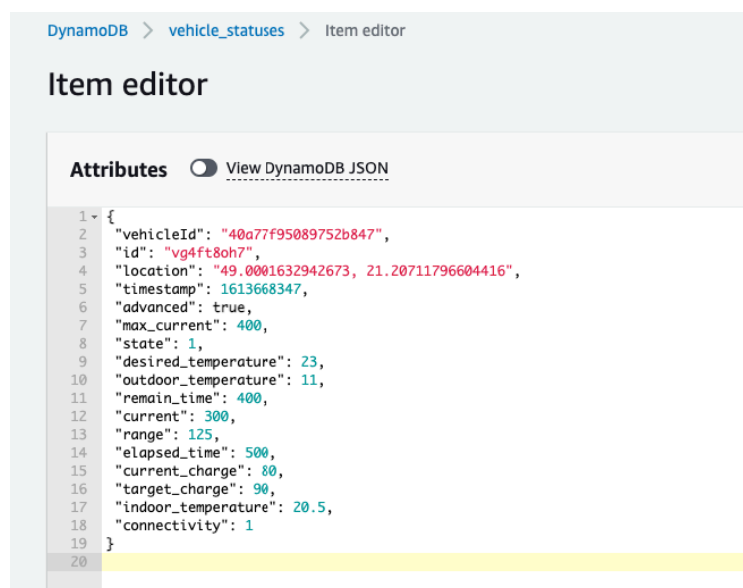
Obrázok 6: Ukážka callbacku v Sigfox Backend

Ako druhý cloud, ktorý platforma využíva na uskladňovanie a spracovanie údajov sme sa v našej práci rozhodli použiť cloud Amazon Web Services (ďalej AWS), ktorý je v súčasnosti na trhu cloudových poskytovateľov jednoznačnou dominantou. AWS poskytuje veľké množstvo rôznych typov služieb, ktoré sú implementované ako samostatné komponenty, tzv. resources, ktoré vedia navzájom komunikovať. Vzniká tak modulárny systém, kde sú od seba jednotlivé časti prevádzkovo nezávislé, účtovanie za ich používanie je vyhodnocované samostatne, je možné ich škálovať, vytvárať redundantné riešenia a integrovať službu s rôznymi inými, aj externými cieľmi. Hlavnými komponentmi ktoré potrebujeme v našom riešení my sú (v zátvorke sú uvedené ich názvy v rámci platformy AWS): API endpoint (API Gateway), FaaS runtime (Lambda), NoSQL databáza (DynamoDB) a služba push komunikácie (SNS, Simple Notification Service). Všetky tieto komponenty patria do tzv. always-free free tier ponuky, t.j. balíka komponentov, ktoré sú do istého limitu využívania poskytované zadarmo. Tieto limity sú dostatočne vysoké na to, aby poskytli slobodu vývoja a prezentácie produktov, alebo aj ich prevádzkovanie pri malej záťaži, bez obáv z vysokých poplatkov, s ktorými sú často spájané obavy z využívania cloudových služieb. Napríklad limity, ktoré sa týkajú nášho riešenia sú: 25GB kapacita NoSQL databázy, 1 milión spustení Lambda funkcií mesačne, 1 milión volaní na API Gateway mesačne a 1 milión publikovaných notifikácií cez službu SNS mesačne. Po ich prekročení by sa používanie konkrétneho komponentu začalo automaticky účtovať (po včasnom upozornení) podľa štandardnej cenovej schémy. [11][12]

Mechanizmus fungovania je nasledujúci – API Gateway je slúži ako HTTP endpoint pre príjem správ zo Sigfox Backendu, respektíve priamo z dopravného prostriedku (pri prenose cez wi-fi). Prijatá správa sa posunie na spracovanie Lambda funkcii, ktorá vykoná dekompresiu, prípadne ďalšie úkony s údajmi, napríklad vyhodnotenie, . Správa je následne uložená do dokumentovej databázy DynamoDB, čo vyvolá ďalšiu Lambda funkciu, ktorá v databáze overí, či na notifikácie o zmene stavu daného dopravného prostriedku je prihlásený nejaký klient (mobilná aplikácia). V prípade že áno, posunie túto správu na odoslanie danému klientovi službe SNS, ktorá ich odošle na servery spoločnosti Google (Firebase), ktorá prostredníctvom technológie push aktívne sprostredkuje jej doručenie na cieľové zariadenie. Službe Firebase Cloud Messaging, ktorá je na to použitá sa venuje 5. kapitola.

API Gateway má nastavené okrem endpointu pre príjem aktualizácií stavu vozidiel aj ďalšie endpointy - na registráciu/odregistráciu klienta na záujem prijímať notifikácie

o zmene stavu vozidla s želaným id, manuálne vyžiadanie posledného známeho stavu nejakého vozidla, vyžiadanie skupiny záznamov z vyžiadaného časového obdobia (pre zobrazenie histórie), alebo pre spätné ovládanie dopravného prostriedku (zmena želanej interiérovej teploty, požiadavka na zaslanie momentálneho stavu, atď.) (je cieľom ďalšej implementácie). DynamoDB databáza ukladá aktualizácie stavov všetkých dopravných prostriedkov pripojených na platformu vo formáte založenom na JSON objektoch. Každý záznam predstavuje jeden “dokument“, pričom tieto dokumenty sú rozdelené do tzv. partícií, na základe id vozidla. Takéto rozdelenie značne uľahčuje indexovanie a vyhľadávanie v NoSQL databáze, hlavne ak by už obsahovala milióny záznamov. Interne tiež databázový systém riadi distribúciu fyzického uloženia údajov na servery nachádzajúce sa v rovnakom clustri, čo značne zvyšuje rýchlosť ich vyhľadávania na rozdiel od scenáru, kedy by sa čiastočné bloky údajov nachádzali na rôznych serveroch v úplne iných geografických lokalitách.



The screenshot shows the AWS DynamoDB console's 'Item editor' for the 'vehicle_statuses' table. The 'Attributes' tab is active, displaying a JSON object with the following data:

```
1 {
2   "vehicleId": "40a77f95089752b847",
3   "id": "vg4ft8oh7",
4   "location": "49.0001632942673, 21.20711796604416",
5   "timestamp": 1613668347,
6   "advanced": true,
7   "max_current": 400,
8   "state": 1,
9   "desired_temperature": 23,
10  "outdoor_temperature": 11,
11  "remain_time": 400,
12  "current": 300,
13  "range": 125,
14  "elapsed_time": 500,
15  "current_charge": 80,
16  "target_charge": 90,
17  "indoor_temperature": 20.5,
18  "connectivity": 1
19 }
```

Obrázok 7: Ukážka záznamu v databáze DynamoDB

Funkcie Lambda sú založené na princípe cloud-computingu zvanom FaaS (Function-as-a-Service), kedy poskytovateľ cloudu zabezpečuje celé prostredie pre beh daných funkcií, a vývojár poskytne len samotný kód funkcie, ktorá sa spustí a vykoná pri každom zavolaní iným komponentom. Sama môže synchronne alebo asynchrónne spustiť ďalšie funkcie a svoj výsledok vráca späť entite, ktorá ju spustila. Tento moderný prístup ponúka vysokú efektívnosť, jednoduchosť a výhodnú cenovú politiku založenú len na čase kedy funkcia reálne beží, a množstve pamäte, ktorú alokuje. Kód týchto funkcií je možné poskytnúť v rôznych programovacích jazykoch, v našom riešení sme použili jazyk JavaScript. [12]

4 Android aplikácia

V rámci práce sme si ako cieľovú mobilnú platformu vybrali operačný systém Android. Pre tento operačný systém sme vytvorili aplikáciu, ktorá používateľovi sprostredkuje všetky informácie a možnosti interakcie so svojimi sparovanými dopravnými prostriedkami. Aplikáciu sme písali v programovacom jazyku Java a za pomoci nástrojov ktoré poskytuje oficiálne IDE vývojové prostredie Android Studio. Pri návrhu a programovaní aplikácie sme dbali na využitie moderných mechanizmov ktoré nám Android SDK a Android Studio ponúka, ako napríklad knižnice Android Jetpack [13], ktorá poskytuje množstvo komponentov navrhnutých a vyladených pre uľahčenie a skvalitnenie vývoja aplikácií pre Android, platformy Firebase [14] ktorá ponúka nástroje na často riešené problémy pri vývoji aplikácií, využili sme napríklad mechanizmy pre prihlasovanie používateľov, cloudovú databázu pre uloženie údajov o používateľových vozidlách, či mechanizmu push komunikácie medzi cloudom a aplikáciou. Taktiež sme sa držali Googlom odporúčaných návrhových vzorov, ako napríklad single-activity aplikácia, kedy celá aplikácia pozostáva len z jedného komponentu aktivity a jednotlivé obrazovky sú implementované ako fragmenty. V rámci interného rozdelenia funkčných častí aplikácie sme použili model MVVM (Model-View-ViewModel), ktorý logicky oddeľuje grafickú (prezenčnú) časť aplikácie, logiku a zdroje informácií do samostatných celkov.

Pri návrhu používateľského rozhrania nám pomohol nástroj designer implementovaný v Android Studiu. Ako GUI komponenty sme použili štandardné Android prvky (views) ako tlačidlá, textviews, editboxy, ale aj dodatočné externé knižnice napr. pre zobrazenie grafov, či mapy. Views sú v rámci GUI rozmiestnené pomocou viacerých rôznych layout komponentov, a to LinearLayout, ConstraintLayout, GridLayout, ale aj moderných CardView a CoordinatorLayout. Pre jednoduchú komunikáciu grafických komponentov s logikou aplikácie sme použili mechanizmus View-binding. Snímky obrazoviek z aplikácie sa nachádzajú v prílohe A.

Aplikácia pozostáva z hlavnej obrazovky, kde používateľ vidí informácie dostupné o aktuálne zvolenom prostriedku prehľadne rozdelené do dlaždíc. Tieto sú interaktívne, a po kliknutí na ne sa používateľovi zobrazí buď dialógové okno (napr. pre úpravu nabíjacieho prúdu alebo želanej teploty vo vozidle) alebo je presmerovaný na obrazovku s detailnejším zobrazením (napr. po kliknutí na dlaždicu polohy sa zobrazí fragment s mapou). Druhou obrazovkou aplikácie je karta histórie, kde si používateľ vie vo forme interaktívneho grafu

prezerat' časový priebeh rôznych údajov v minulosti. Treťou obrazovkou sú nastavenia aplikácie, kde používateľ nájde možnosti na zmenu svojej prezývky, odhlásenie sa z účtu, zmenu jazyka aplikácie, zmenu niekoľkých preferencií správania sa aplikácie, nastavenie názvu, obrázka a údajov aktuálneho dopravného prostriedku, a niekoľko informácií o aplikácii. Táto obrazovka je na rozdiel od ostatných vytvorená pomocou komponentu PreferenceScreen a jeho podkomponentov pre jednotlivé typy nastavení, ako napr. EditTextPreference, SwitchPreference a tak ďalej. Zvolené nastavenia sú ukladané do úložiska SharedPreferences, lokálnej databázy vozidiel, alebo aj do cloudovej databázy Firestore. Zo spodnej časti obrazovky vieme vysunúť zoznam všetkých vozidiel spárovaných s účtom daného majiteľa, kde vidíme ich posledný známy stav a vieme si ich zvoliť pre zobrazenie na hlavnej obrazovke. Poslednou obrazovkou je prihlasovacia obrazovka, ktorá sa zobrazí len pri prvom spustení aplikácie, resp. ak sa používateľ odhlási. Táto obrazovka nám ponúka 2 spôsoby ako sa prihlásiť – buď pomocou emailu a hesla, alebo pomocou Google účtu.

Prihlasovanie je riešené pomocou služby Firebase Authentication. Po prihlásení sa z cloudovej databázy Firebase Firestore stiahnu údaje daného používateľa o jeho spárovaných dopravných prostriedkoch, ich obrázky z Firebase Cloud Storage a prípadne uložené preferencie aplikácie. Tieto sú načítané do lokálnej SQLite databázy, prípadne do SharedPreferences. Následne si po prihlásení (a každom ďalšom spustení aplikácie) aplikácia vyžiada zo servera (cloudu AWS) najaktuálnejšie údaje o zvolenom vozidle, a to prostredníctvom HTTP požiadavky na REST API na cloude. Táto HTTP komunikácia je implementovaná s použitím knižnice Retrofit. V závislosti od nastavenia v preferenciách aplikácie sú ďalšie updaty stavu vozidla prijímané už pomocou mechanizmu push správ (viď kapitola č.5), alebo sa na ne aplikácia periodicky dopytuje na REST API rovnakou metódou ako prvýkrát po spustení. Vždy po prijatí nového stavu ľubovoľného vozidla je tento stav najprv uložený do lokálnej databázy aplikácie, a následne, ak ide o aktuálne zvolené vozidlo, sú prekreslené dlaždice s informáciami, prípadne iné prvky. Lokálna databáza je ako medzi prvok použitá z dvoch dôvodov. Po prvé sa do nej aplikácia pozrie hneď po spustení, a v prípade že poslednú update vozidla nie je starší ako nastavená hranica, hneď ho zobrazí, čo potenciálne ušetrí čas načítavania aktuálnych údajov zo siete po každom spustení aplikácie. Po druhé, je využitá ako prvotný zdroj údajov pri požiadavke na vykreslenie grafu na obrazovke história, až následne sú údaje prípadne doplnené ďalšími, prijatými z cloudu. To je týmto spôsobom implementované z rovnakého dôvodu (rýchlejšie načítavanie).

5 Firebase Cloud Messaging

Posledným dielom skladačky je technológia push správ, respektíve notifikácií. Ide o riešenie na medziplatformovú komunikáciu medzi viacerými cloudmi alebo cloudom a zariadením, spôsobom, že komunikácia s klientom je iniciovaná zo strany servera. To pri väčšine ostatných sieťových komunikačných protokolov nie je možné, minimálne nie bez prvotného vytvorenia spojenia klientom a jeho udržiavania. Interne technológia push samozrejme stále využíva princíp dopytovania sa servera zo strany klienta, avšak deje sa to neviditeľne pre služby a aplikácie, ktoré ju využívajú. Tento proces zabaľuje vyššia vrstva nad týmito procesmi, ktorá zároveň v komunikácii vystupuje ako prijímateľ správ, a sama ich následne distribuuje konkrétnym procesom, ktorým sú finálne určené. V prípade mobilných zariadení túto funkcionalitu implementuje samotný operačný systém, a teda je nutné aby bola implementovaná priamo jeho vývojárom. V prípade systému Android je to spoločnosť Google, pričom jej pomenovanie pre implementáciu technológie sa volá Firebase Cloud Messaging (FCM) a funguje ako súčasť platformy Firebase, a v prípade systému iOS spoločnosť Apple, pričom jej implementácia sa volá APNs – Apple Push Notification service. Táto technológia je tiež známa z prostredia webu, kde webové stránky vedia poslať používateľovi notifikácie, pričom úlohu príjemcu v tomto prípade zastáva webový prehliadač. Hlavnou výhodou a dôvodom vzniku tejto technológie je jej efektívnosť – namiesto toho aby sa x procesov pravidelne dopytovalo rôznych serverov na dostupnosť nového stavu, alebo s nimi udržiavalo otvorené spojenie (ako to rieši napr. technológia WebSockets) čo je veľmi energeticky náročné, všetku komunikáciu medzi procesmi v zariadení a servermi v internete zabaľuje operačný systém na jednom konci, a centrálny cloud na druhom konci, ktorý zbiera a drží všetky správy určené cieľovému zariadeniu. Vytvorí sa tým lievnik, kedy operačný systém na základe vlastného uváženia spravuje spojenie, a cloud zasiela správy pre všetkých adresátov na danom zariadení súčasne. Pokiaľ je mobilné zariadenie prebudené, táto komunikácia sa javí ako v reálnom čase, pokiaľ je v uspanom režime, doba doručenia záleží na nastavenej politike. [15][16]

Zo strany záujemcu o využitie technológie je potrebné, aby odosielateľ vedel ID adresáta, tzv. token, a aby ho spolu s obsahom správy odoslal na centrálny server, v prípade FCM na endpoint <https://fcm.googleapis.com/fcm/send>, spolu s autorizačným kľúčom v HTTP hlavičke. Aby sme to mohli docieľiť v našom riešení, pri spustení klientskej aplikácie našej platformy táto “zaregistruje“ na našom AWS cloude svoj záujem prijímať

aktualizácie o stave daných dopravných prostriedkov, teda odošle naň svoj pridelený token spolu s ich identifikátormi. Následne AWS cloud preposiela nové aktualizácie z týchto prostriedkov na majiteľove mobilné zariadenie. Firebase Cloud Messaging, rovnako ako Apple Push Notification service sú free of charge, teda poskytované bezplatne.

Záver

Cieľom práce bolo navrhnúť a vytvoriť univerzálnu platformu, do ktorej by bolo možné pripojiť dopravné prostriedky rôzneho typu a značiek, či pomocou integrácie služby od výroby, alebo dodatočnou inštaláciou telematickej jednotky dokúpenej ako samostatného zariadenia. Tento cieľ sa nám podarilo splniť a projekt je vo vysokom štádiu rozpracovania, vrátane ukázkového riešenia pre prezentáciu platformy. Súčasťou je simulátor telematickej jednotky osobného elektrického automobilu, prenosová časť založená na technológií Sigfox, umiestnenie logiky platformy do verejného cloudu AWS a mobilná aplikácia pre OS Android. Systém je modulárny, a teda ak by niektorá z častí nevyhovovala požiadavkám konkrétneho použitia, nie je problém vymeniť ju za inú, napríklad použiť inú prenosovú technológiu, alebo umiestniť logiku do cloudu iného poskytovateľa, alebo do svojho vlastného. Práca sa venovala hlavne vysvetleniu princípov a poskytnutiu ukážky. Ako celok môže slúžiť ako dobrý príklad moderného IoT riešenia nejakého problému a pomôcť ako sprievodca z čoho všetkého sa takýto systém skladá, ako jednotlivé časti spolupracujú a ako postupovať pri jeho návrhu. Obzvlášť prínosné to môže byť v súčasnej dobe, kedy je svet IoT riešení a cloudových technológií na raketovom vzostupe, no zároveň sú tieto oblasti tak rozsiahle, že vývojár alebo firma, ktorá by chcela pracovať na vývoji nejakého produktu, no nemá v tejto oblasti skúsenosť, potrebuje venovať veľmi veľké množstvo času len zorientovaniu sa v problematike.

Zoznam použitej literatúry

- [1] Standard Range Plus Supercharging Speed [online] URL:
<https://teslamotorsclub.com/tmc/threads/standard-range-plus-supercharging-speed.146816/page-9>
- [2] SIGFOX. What is Sigfox? [online] URL: <https://build.sigfox.com/sigfox>
- [3] The Institute of Electrical and Electronics Engineers (2008). IEEE 754-2008 [online] URL: <https://standards.ieee.org/standard/754-2008.html>
- [4] SIGFOX. Development informations [online] URL:
<https://build.sigfox.com/development>
- [5] SOS.SK (2017). Technológie na bezdrôtový prenos dát [online] URL:
<https://www.sos.sk/articles/sos-supplier-of-solution/internet-of-things-3-cast-technologie-na-bezdrotovy-prenos-dat-2044>
- [6] SIGFOX (2021). Introducing 0G network [online] URL:
https://www.sigfox.com/sites/default/files/og-guide/Sigfox%20-%20Introducing%200G_Jan2021.pdf
- [7] AMR.SK. Technológia IoT Sigfox [online] URL: <http://www.amr.sk/?s=tech-sigfox>
- [8] SIGFOX. Coverage information [online] URL: <https://www.sigfox.com/en/coverage>
- [9] SIGFOX SLOVAKIA (2020). Aktuálne pokrytie Sigfox na Slovensku [online] URL:
https://sigfoxslovakia.com/technologie-sigfox/aktualne_pokrytie_sigfox_na_slovensku/
- [10] SIGFOX. Sigfox developers guides [online] URL: <https://www.sigfox.com/en/sigfox-developers>
- [11] AMAZON. AWS Free Tier [online] URL: <https://aws.amazon.com/free/?all-free-tier>
- [12] AMAZON. AWS Documentation and developers guides [online] URL:
<https://docs.aws.amazon.com/>
- [13] GOOGLE. Android Jetpack documentation [online] URL:
<https://developer.android.com/jetpack>
- [14] GOOGLE. Firebase documentation [online] URL: <https://firebase.google.com/docs>

[15] KAMATH A. (2020). What are Push Notifications? An In-depth 2020 Guide [online]

URL: <https://www.moengage.com/blog/what-are-push-notifications/>

[16] WIKIPEDIA.ORG. Push Technology [online] URL:

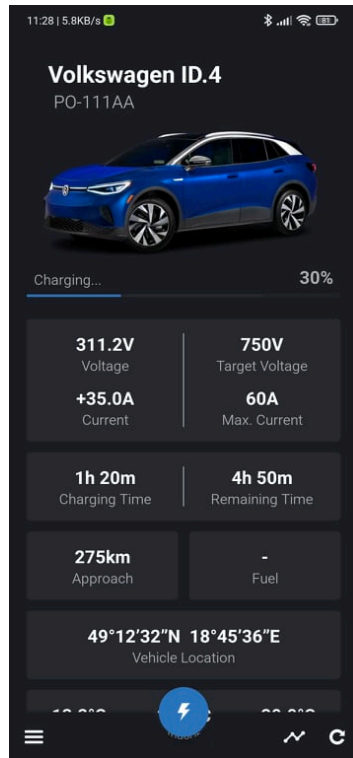
https://en.wikipedia.org/wiki/Push_technology

Zoznam príloh

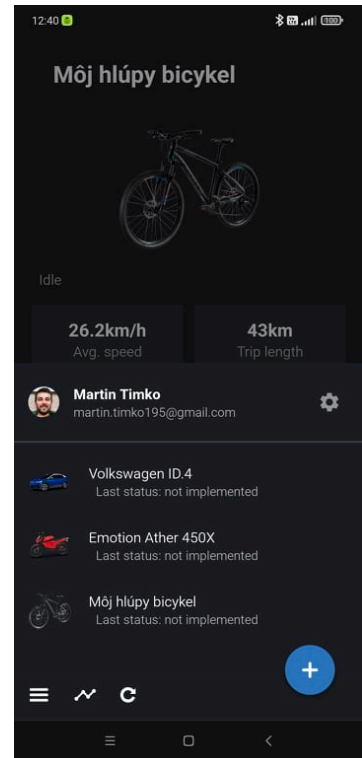
Príloha A Snímky obrazovky z používateľskej aplikácie pre Android

Prílohy

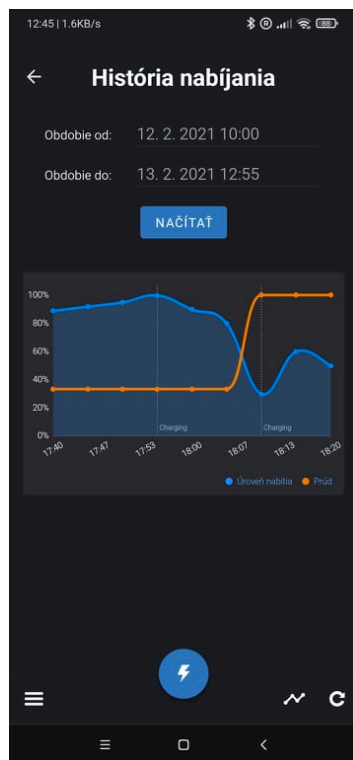
Príloha A: Snímky obrazovky z používateľskej aplikácie pre Android



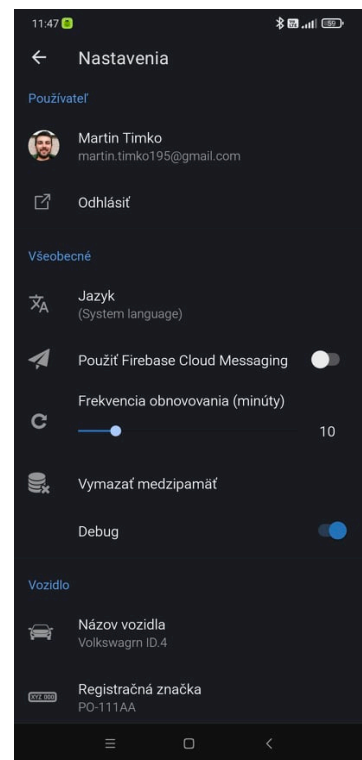
Domovská obrazovka



Selektor aktuálneho vozidla



História



Nastavenia