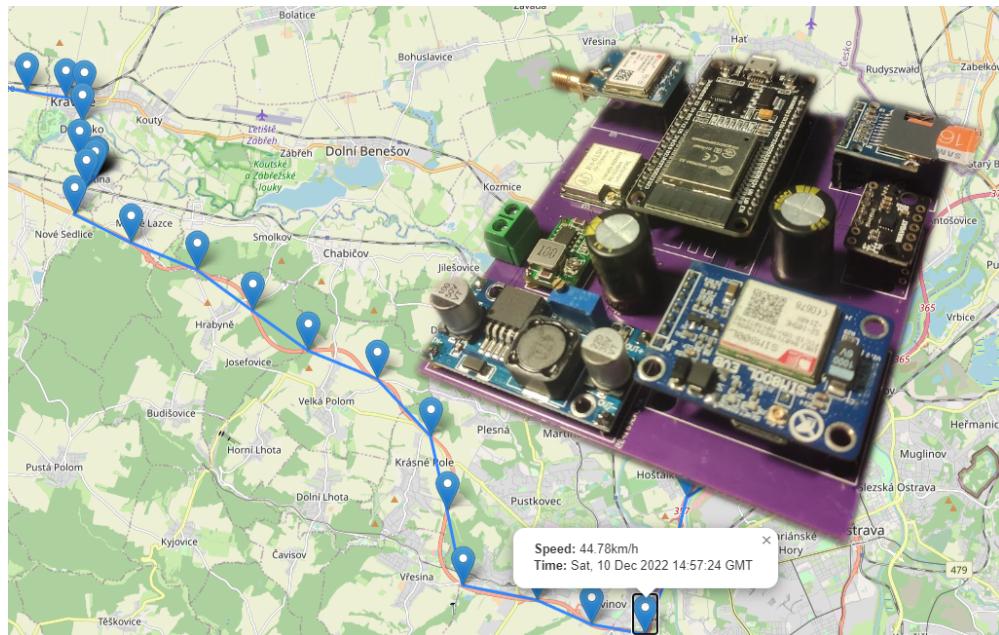


ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

CarStatsTracker

Jakub Heisig



Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2022/2023

Poděkování

- *Rád bych poděkoval panu učiteli Ing. Petru Grussmannovi za jeho připomínky pro vylepšení funkčnosti projektu*

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2022

podpis autora práce

ANOTACE

Výsledkem projektu je hardwarové zařízení a potřebný software, pomocí kterého je možno sledovat vozidlo či objekt v jeho pohybu díky bezdrátové komunikaci prostřednictvím LoRaWAN a WiFi. Jsou přijaty a uloženy údaje jako: GPS pozice, rychlosť a přesný čas na lokální SD úložiště a zároveň jsou posílány přes LoRaWAN síť. Tyto údaje lze zobrazit jako body na interaktivní mapě přímo na web serveru sledovacího zařízení nebo na veřejně dostupném webovém serveru, chráněném pomocí HTTPS a přihlašovacími údaji. Na tento server používající Node.JS jako backend, jsou data přijata, zpracována a uložena do JSON souborů. Také jako přídavnou možností je schopnost měření zrychlení mezi předem zadanými rychlostmi a vzdálenostmi.

ANOTATION

The result of the project is a hardware device and the necessary software that can be used to track a vehicle or object in its movement through wireless communication via LoRaWAN and WiFi. Data such as: GPS position, speed and exact time are received and stored on local SD storage while being sent over the LoRaWAN network. This data can be displayed as points on an interactive map directly on the tracker's web server or on a publicly accessible web server, protected by HTTPS and login credentials. On this server using Node.JS as a backend, the data is received, processed and stored in JSON files. Also as an additional option is the ability to measure acceleration between pre-specified speeds and distances.

KLÍČOVÁ SLOVA

LoRa, GPS sledování, vzdálená komunikace, zobrazení v mapě, měření zrychlení

OBSAH

Úvod	5
1 Teoretická a metodická východiska	6
1.1 Zjišťování zeměpisné polohy	6
1.2 Bezdrátová komunikace	6
2 Využité technologie	8
2.1 ESP32	8
2.2 LoRa & LoRaWAN	8
2.3 Debian VPS	8
2.4 JavaScript	9
2.5 Node.JS	9
2.6 JSON	9
2.7 Leaflet & OpenStreetMap	9
2.8 GPS	9
2.9 PlatformIO / VSCode / Atom.io	10
2.10 MQTT	10
2.11 SD úložiště	10
3 Způsoby řešení a použité postupy	11
3.1 Hardware a kód zařízení	11
3.1.1 ESP32 a přídavné moduly	11
3.1.2 Napájení	12
3.1.3 Finální zhotovení	13
3.1.4 ESP32 Program	14
3.2 Software a serverový backend	16
3.2.1 Zpracovávání dat	16
3.2.2 Uživatelské rozhraní	17
4 Výsledky řešení a výstupy	19
4.1 Funkce práce	19
4.2 Splněné a nesplněné cíle	19
Závěr	21
Seznam použitých informačních zdrojů	22

Úvod

Cílem práce bylo vytvořit a vyrobit zařízení schopné zaznamenávání pohybu auta a odesílání těchto dat pomocí LoRaWAN či GSM sítě. Dále vytvořit webový server pro přijímání a ukládání dat a umožnit jejich zobrazení na veřejně zabezpečené stránce pomocí Node.JS.

Mou inspirací byly zařízení s jménem Dragy od Dragy Motorsports a MTrack od Automatrics Limited. Dragy pracuje jako výkonný GPS přijímač, který po propojení s mobilem je schopen měřit zrychlení auta, brzdění a více funkcí na přesnost 1/100 sekundy. MTrack je zařízení pro záchranu ukradeného vozidla pomocí GSM sítě, ale také pomocí vysílaného RF beaconu.

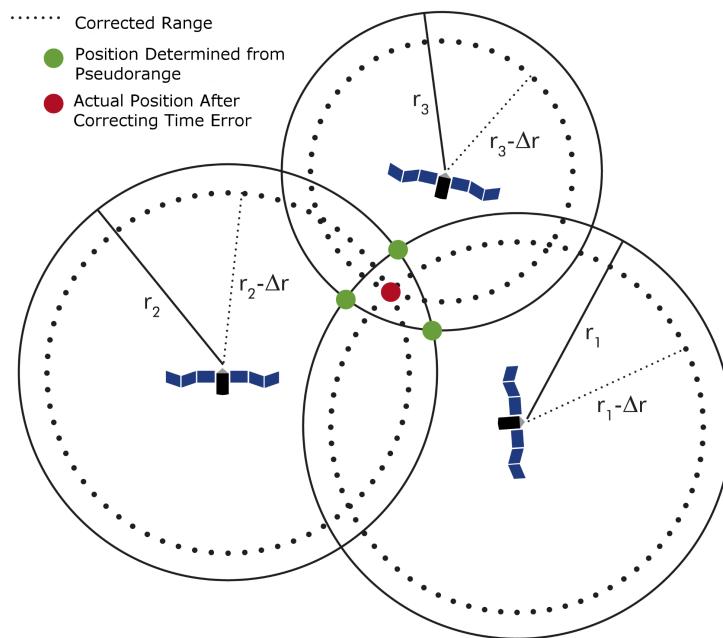
V dokumentaci se zaměřuji na teorii LoRa komunikace a GPS. Dále na jednotlivé využité technologie, finální hardwarové a softwarové zhodnocení a možné vylepšení v budoucnosti.

1 TEORETICKÁ A METODICKÁ VÝCHODISKA

1.1 Zjišťování zeměpisné polohy

Jedna z elektronických možností získání aktuální zeměpisné polohy je GPS, tzv. Global Positioning System. Jedná se o radionavigační družicový systém pro určování polohy a času na zemském povrchu a v přilehlém prostoru se začátky vracející se až do roku 1973.

Systém pracuje na principu jednosměrného dálkoměru, kde se měří doba šíření signálu mezi satelitem a zařízením. Pomocí výpočtu ze 3 satelitů, je možné určit přibližnou polohu. Čím více satelitů jste schopen zachytit a zaznamenat, tím větší je přesnost určení polohy až na 1 metr. Pracuje se tedy s velmi krátkými délkami času, kde špatná synchronizace času v satelitech může způsobit chybu v měření při 1ns o 30cm. Proto existuje přes 20 řídících a monitorovacích stanic, které kontrolují a spravují více než 24 satelitů.

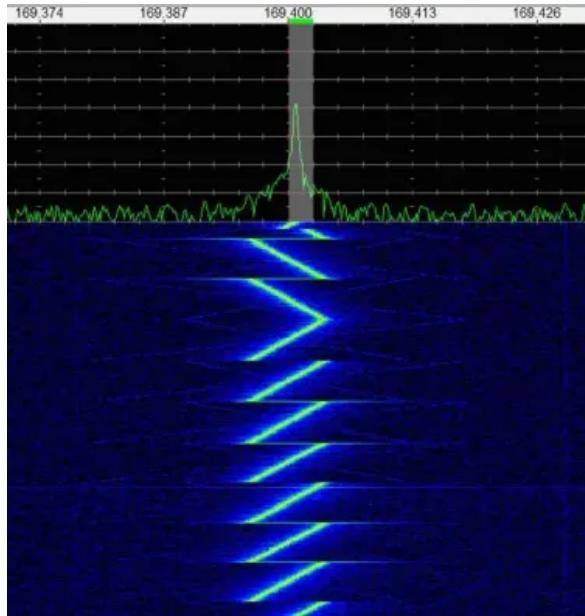


Obrázek 1: Jak názorně funguje triangulace GPS (zdroj: globalspec.com)

1.2 Bezdrátová komunikace

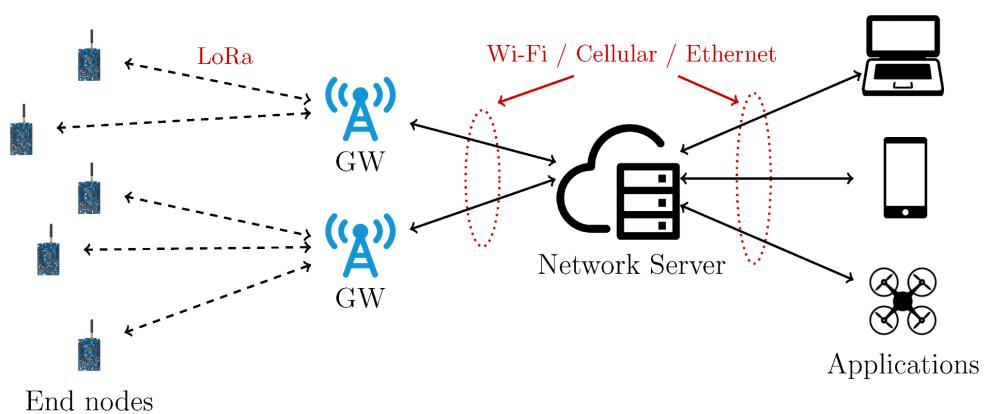
Jedna z hlavních bezdrátových komunikací využita na tomto projektu je LoRa, která je postavena na CSS (chirp spread spectrum) modulaci, což znamená že signál navazuje na pohybující se sinusoidu a tím využívá celé ohraničené spektrum protokolu. Díky tomuto není náchylná Dopplerovému efektu a zaniknutí signálu u nízkovýkonových vysílačů, což LoRa je s maximálním výkonem 25mW. Díky této modulaci a spolu s nízkou přenosovou rychlostí, je LoRa

schopna přenést data na velké vzdálenosti mezi zařízeními na frekvenci 868-928MHz. Tato vzdálenost může dosáhnout od 5km v zastavěné části města až do 16km při přímém dohledu mezi zařízeními.



Obrázek 2: Waterfall LoRa modulace (zdroj: link-labs.com)

LoRaWAN je jedna ze síťových protokolů pro LoRa. Zajišťuje správnou komunikaci mezi hlavními stanicemi (LPAN Gateways) a koncovými zařízeními, z kterých jsou data přijata, zpracována na serverech různých firem (v méém případě TheThingsNetwork) a přeposlána uživatelským aplikacím. Tato technologie je nejvíce využívána pro IoT zařízení po celém světě.



Obrázek 3: Diagram principu LoRaWAN (zdroj: mdpi.com)

Tuto technologii jsem použil kvůli její zajímavosti a škále možností využití v jakémkoliv zařízení potřebující přenos dat z zařízení kdekoliv na světě na internetový server, pokud je v dosahu gateway použité LoRaWAN infrastruktury.

2 VYUŽITÉ TECHNOLOGIE

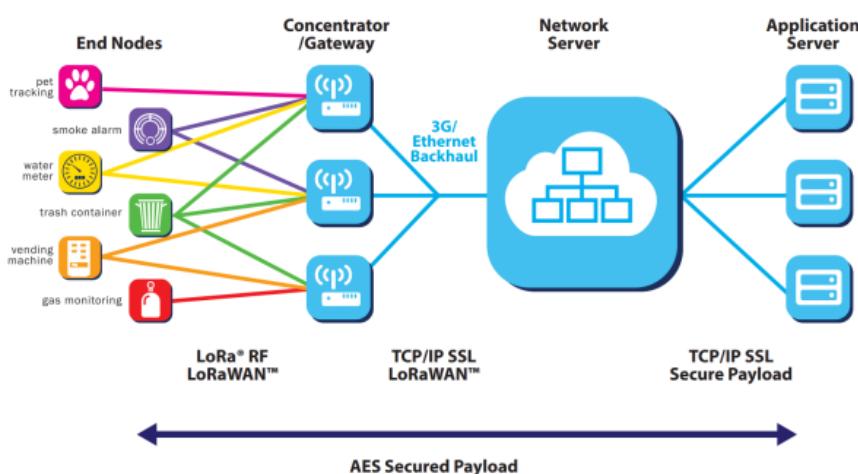
2.1 ESP32

ESP32 je programovatelný jednočipový mikrokontrolér (MCU) s vestavěnou WiFi a Bluetooth připojení od výrobce Espressif. Tento výkonný 2 jádrový čip s frekvencí až 240MHz můžeme naprogramovat a dynamicky využívat všechny jeho přístupné GPIO piny k propojení s ostatními hardwarovými prvky. Funguje jako mozek celého fyzického zařízení.

2.2 LoRa & LoRaWAN

LoRa (Long Range) je bezdrátové řešení komunikace pro velké vzdálenosti při malé spotřebě. Využívá bezlicenční pásmo a díky vysílání v krátkých impulsech v decimetrových vlnách potřebuje jenom már mW výkonu pro dosah až 10km.

LoRaWAN (Long Range Wide Area Network), je protokol postavený na LoRa, kde se jednotlivé IoT zařízení mohou připojit na gateway zařízení a komunikovat takto s servery na internetu (TheThingsNetwork, Helium, etc..).



Obrázek 4: Diagram principu LoRaWAN (zdroj: medium.com)

2.3 Debian VPS

Pro hostování veřejné webové aplikace používám VPS (Virtual Private Server), což je privátní virtualizovaný server, provozovaný cizí firmou, za který platíte určitou sumu záležící na zvolených parametrech. Já jsem zvolil VPS s operačním systémem Debian s 1GB RAM, 1 vláknem Xeon procesoru a 40GB SSD úložiště.

2.4 JavaScript

Jedná se o multiplatformní, objektově orientovaný, událostmi řízený skriptovací jazyk, jehož autorem je Brendan Eich. Je možné ho využít na webových stránkách, ale také na serverové části projektu.

Všechn kód, kromě kódu použitého přímo pro mikrokontrolér, je napsaný v JavaScriptu.

2.5 Node.JS

Node.JS je softwarový program, který umožňuje psát aplikace v JavaScript jazyce. Tyto aplikace ale nejsou spuštěny na zařízení klienta, ale na serveru. Většinou se používá pro psaní webových serverů a API díky svému asynchronnímu přístupu ke kódu.

2.6 JSON

JavaScript Object Notation je nezávislý datový formát a způsob zápisu dat. Může obsahovat objekty, pole a jednotlivé hodnoty které se mohou dále rozvíjet. Využil jsem ho kvůli mé vlastní zkušenosti a jeho jednoduché implementaci v JavaScript jazyce a Arduino prostředí díky knihovně ArduinoJSON.

2.7 Leaflet & OpenStreetMap

OpenStreetMap je *Open Database* projekt (podobně jako Wikipedie) pro tvorbu volně dostupných geografických dat a následně jejich přenesení do topografických map.

Leaflet je *Open Source* JavaScript knihovna umožňující vytvářet interaktivní mapy na webu pomocí prostředků jako OpenStreetMap. Umožňuje pohyb mapou, vytváření bodů, čar, ohrazení objektů, návodů a navíc běží čistě na klientském prohlížeči oproti konkurenci.

2.8 GPS

Díky GPS (Global Positioning System), jsme schopni pomocí malé antény a desítky satelitů ve vesmíru zjistit s velkou přesností naši polohu spolu s nadmořskou výškou a přesným UTC časem. Tyto GPS čipem zpracované data můžeme získávat v rázech milisekund a dále tato data zpracovávat a zjistit naši pozemní rychlosť a více.

2.9 PlatformIO / VSCode / Atom.io

Atom.io byl (2011-2022) open source textový/zdrojový editor vyvinut společností GitHub pomocí Node.JS a webového prohlížeče Chromium. Používal jsem ho k vývoji projektu než jsem přešel na VS Code.

Visual Studio Code je také open source textový/zdrojový editor, ale vyvinut společností Microsoft. Je více populární a vyvinutý než Atom.io a obsahuje více kvalitních modulů pro usnadnění práci, např. s mikrokontroléry jako PlatformIO.

PlatformIO je zásuvný modul pro VSCode, umožňující programování pro podporované mikrokontroléry. Obsahuje databázi použitelných knihoven, mikrokontrolérů, frameworků a více. Použil jsem ho později pro programování a debugování mého projektu, protože dokázal rozpoznat změny v kódě a kompilovat jenom nalezené změny a tím ušetřit čas o více než 80% oproti ArduinoIDE.

2.10 MQTT

Jedná se o jednoduchý zprávový protokol, kde přijímače (tzv. subscribers) dají najev o které zprávy mají zájem, tyto zprávy jsou poté schopni přijmout od tzv. publishers, kteří posílají zprávy právě jenom přijímačům, kteří mají o jejich zprávy zájem. MQTT bylo vyvinuto pro zařízení s omezeným výkonem nebo rychlostí připojení. Je nutné ho provozovat přes TCP/IP protokol.

2.11 SD úložiště

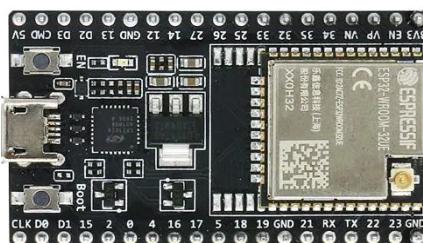
Secure Digital je malá paměťová karta využívající flash paměť vyvinutou pro přenosné zařízení. SD karty využívají různé způsoby komunikace, SPI mód pracující v 1 bitové sběrnici, pomocí které mohou mikrokontroléry komunikovat s SD kartami a SD mód pracující až s 4 bitovou sběrnicí pro zvýšení propustnosti a rychlosti. V mému projektu používám formát karet MicroSD SDHC, který přinesl podporu 32GB karet a nutnost podpory FAT32

3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

3.1 Hardware a kód zařízení

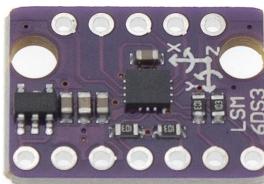
3.1.1 ESP32 a přídavné moduly

Samotné fyzické zařízení je založeno na čipu ESP32, které funguje jako hlavní mozek propojující všechny ostatní moduly jako akcelerometr, LoRa modul, GPS modul, SD kartu a provádí všechny naprogramované funkce zapsané v jeho EEPROM paměti. Tento čip jsem zvolil kvůli potřebě WiFi připojení, ale také kvůli jeho výkonu s dvěma jádry na 240MHz.



Obrázek 5: ESP32 Vývojová deska

Pro akcelerometr jsem zvolil čip LSM6DS3. Je více než dostatečný, navíc s zabudovaným gyroskopem. Výhoda tohoto čipu je že máme na výběr mezi I2C nebo SPI komunikací.



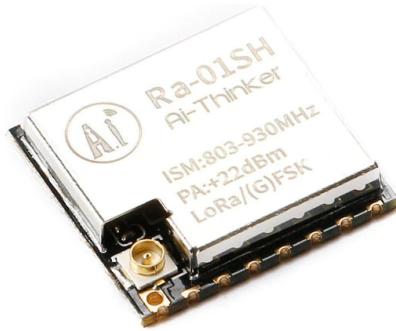
Obrázek 6: LSM6DS3 modul

Pro GPS modul jsem zvolil u-blox NEO-6M, jelikož tento modul je schopen UART komunikace na 3.3v, stejně jako ESP32. Zvolil jsem ho hlavně kvůli možnosti naprogramovat tento modul na vyšší UART baudrate, možnost ovládat informační GPS zprávy pro více informací k dekódování a k možnosti zvýšení frekvence odesílání dat až na 10Hz.



Obrázek 7: u-blox GPS modul

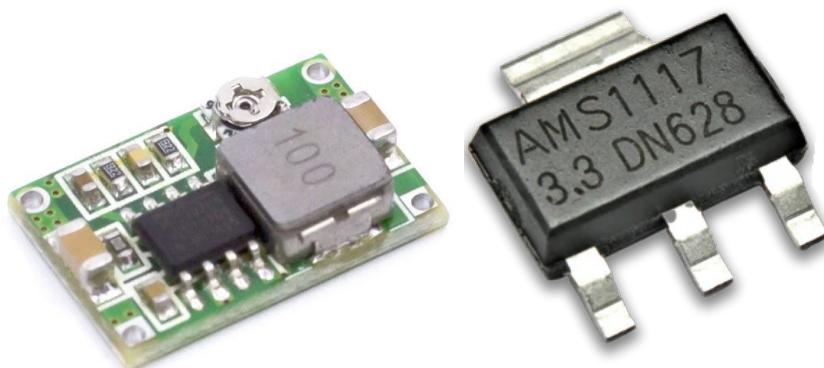
Jedna z hlavních funkcí projektu je komunikace pomocí LoRa. Kvalitní čipy nejsou zatím pro mě cenově přijatelné a proto jsem zvolil samostatný modul RA-01SH, který používá SX1262 čip pomocí SPI komunikace. Tyto moduly mají většinou dobrou podporu knihoven, ale je zapotřebí přečíst celý datasheet a zjistit druh zapojení, protože každý výrobce je vyrábí jinak. Tento model je 803-930MHz, což znamená, že jej můžeme použít jak v Evropě na 868MHz, tak i v USA na 915MHz.



Obrázek 8: Ai-Thinker LoRa modul

3.1.2 Napájení

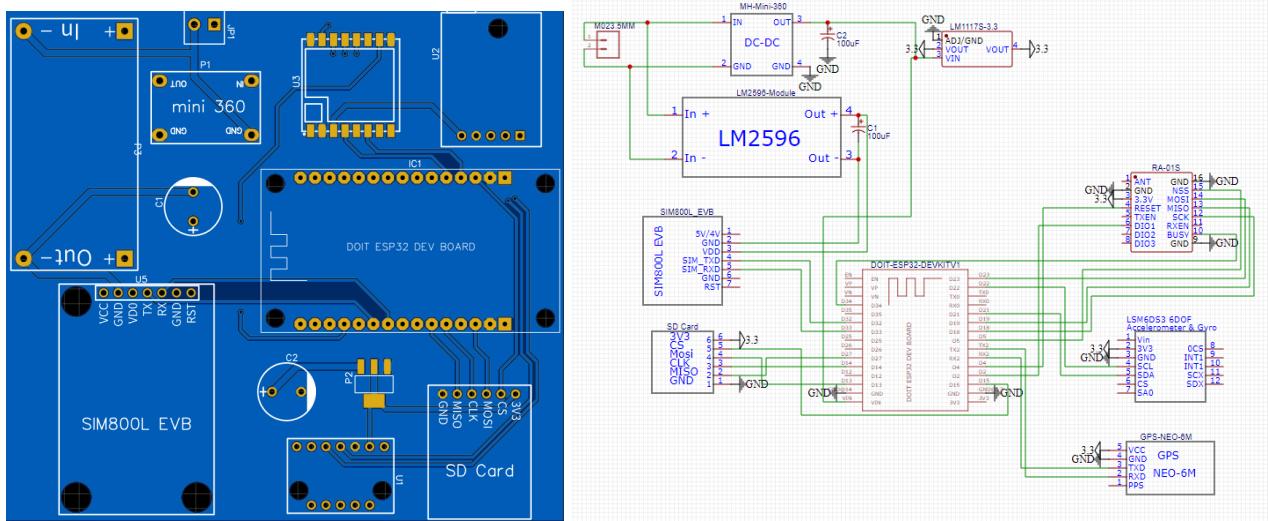
Pro napájení jsem zvolil step-down konvertory, které snižují napětí na předem nastavené hodnoty, takže zařízení může být napájeno voltáží od 6 V do 30 V bez strachu z nevratného poškození. Na finální desce se nachází celkem 4 regulátory, jeden napájí GSM modul, který bude někdy v budoucnu dodělán a druhý napájí další dva menší lineární regulátory. Tyto dva hlavní regulátory mají na svých výstupech kondenzátory pro vyrovnání náhodných proudových špiček nebo rušení způsobené regulátory. Zbývající dva lineární regulátory snižují napětí na 3.3 V, z nichž jeden napájí zvlášť samostatné ESP32 a druhý všechny ostatní moduly.



Obrázek 9 a 10: Mini-360 spínací regulátor a AMS1117 lineární regulátor

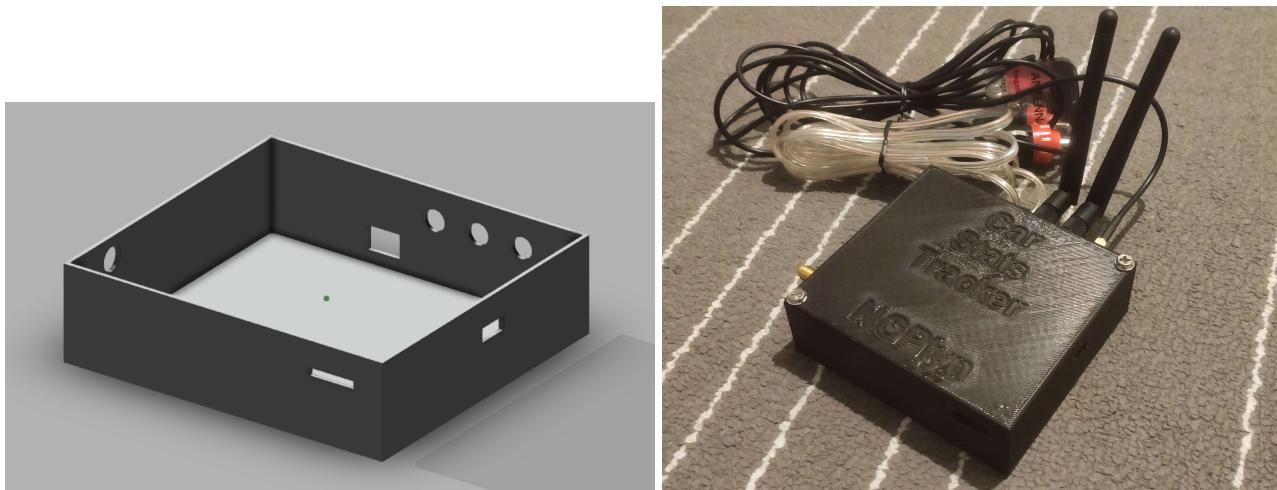
3.1.3 Finální zhotovení

Všechny moduly jsou napájené na vlastně navržený plošný spoj pomocí programu EasyEDA a vyroben firmou JLCPCB. Zvolil jsem je kvůli vlastní zkušenosti a preference. Designovací program je jednoduchý a nabízí pestré nástroje k vývoji elektronických schémat a poté z nich vytvořit plošné spoje. Výroba je velmi rychlá s perfektní kvalitou a příznivou cenou.



Obrázek 11 a 12: Design plošného spoje a jeho elektrické schéma v EasyEDA

Pro finální vzhled jsem vymodeloval box, pomocí programu Fusion360 podle naměřených rozměrů, s otvory pro umístění antén, konektor ESP32 a otvoru pro zasunování SD karty. Následně jsem obě části boxu vytiskl na 3D tiskárně, vložil plošný spoj, přišrouboval vývody antén, přilepil matky, přišrouboval kryt a připojil antény spolu s napájecím kabelem.

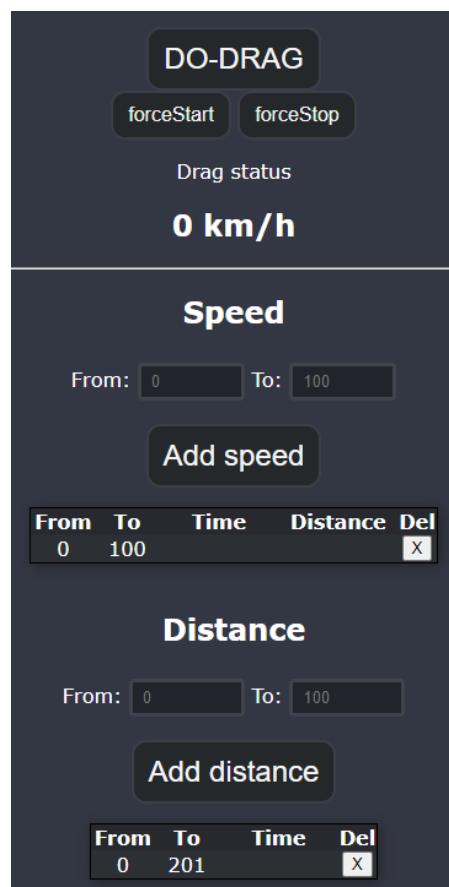


Obrázek 13 a 14: Navržený box v Fusion360 a finální zařízení

3.1.4 ESP32 Program

Jsou využívány obě dvě jádra procesoru, z čehož druhé jádro zpracovává TCP komunikaci a webový server pro plně asynchronní funkci. První hlavní jádro je využíváno zbylým kódem. ESP načte nastavení z SPIFFS paměti pomocí LittleFS, nastaví výchozí hodnoty proměnných a inicializuje všechny potřebné moduly jako WiFi, webový server, akcelerometr, SD kartu a LoRa. Poté se procesor dostane do loopu, kde dekóduje zprávy z GPS modulu a pokračuje v kódu pro zvolený pracovní mód.

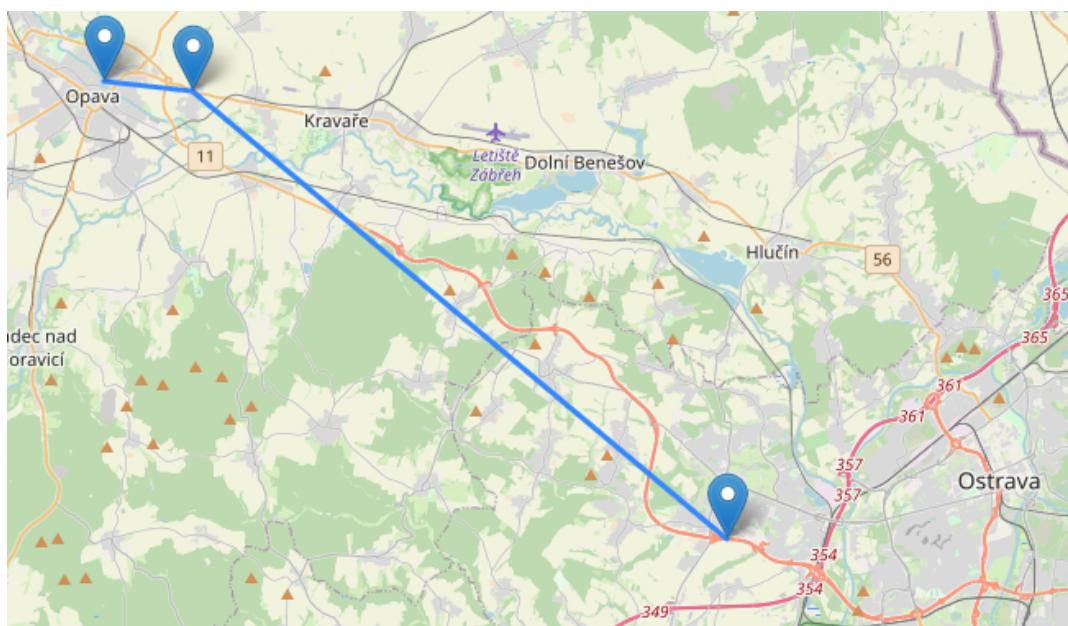
DRAG mód je zpřístupněn až poté, co jsou získány a zpracovány data od uživatele. Tento mód čeká na pohyb pomocí akcelerometru a při detekci pohybu spustí funkci pro měření. Tato funkce zaznamená startovní čas a pozici, poté pro každou rychlosť a vzdáenosť zaznamená uplynulý čas a vzdáenosť potřebnou k dosažení dané rychlosti. Tento *while()* cyklus je možné také přerušit manuálně pokud nebyly dosažené nastavené cíle. Po ukončení je vyvolána další funkce, která zpracuje data z polí a upraví přijatý JSON od uživatele, který je potom odeslán zpět pro zobrazení dat a také uložen na SD kartu. Tato funkce se také používá pro živé zobrazení dat na webu uživateli pomocí parametru funkce, který jenom aktualizuje daný JSON bez uložení.



Obrázek 15: Ovládání DRAG módu

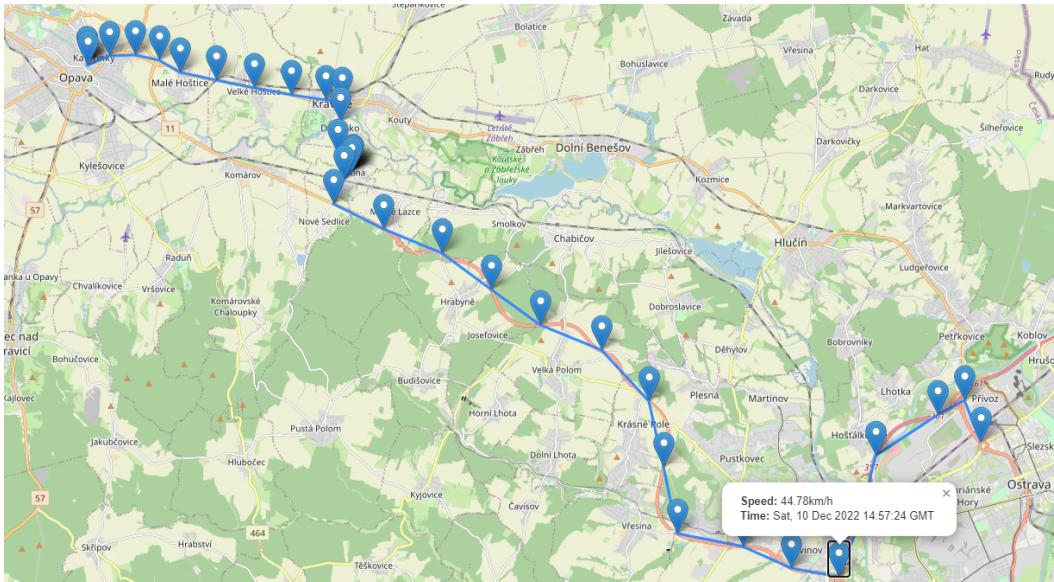
Sledovací mód je při každém zapnutí zvolen jako výchozí. Pokud není detekován žádný pohyb a jsou dostupné GPS data, jsou potom v intervalu 6 minut data uložena na SD kartu a v intervalu 30 minut odeslána přes dostupnou LoRaWAN síť. Jakmile je zaznamenán pohyb, je tento pohyb ověřen zda se nejedná o kalibrační chybu či krátký pohyb objektu a pokud dojde k závěru, že se objekt pohybuje, vytvoří se nový záznam a začnou se ukládat dostupné data v intervalu jedné minuty na SD kartu a v intervalu 5 minut se také odešlou přes LoRaWAN síť. Pokud ale je zaznamenáno, že GPS rychlosť je skoro nulová a po dalších 4 minutách se nemění, je aktuální záznam ukončen a opětovně se čeká na pohyb.

Pokud se zařízení dostane do dosahu nastavené WiFi sítě, je proveden pokus o připojení a při úspěšném připojení jsou všechny neodeslané data poslány serveru pomocí SSL zabezpečeného POST dotazu pro zpracování a uložení. Také jsou prováděny pokusy o připojení k LoRaWAN síti, protože ne kdekoli se zařízení nachází v dosahu funkční gateway.



```
No data available
Data from TTN: 2;n;49.94263;17.90756;1.24;1670671593
Data from TTN: 3;y;49.93854;17.95380;94.23;1670672042
Data from TTN: 4;y;49.91701;18.01102;47.34;1670673123
Data from TTN: 4;y;49.83014;18.13374;120.16;1670673843
No data available
No data available
Data from TTN: 5;y;49.81897;18.16788;118.88;1670684364
Data from TTN: 5;y;49.94027;17.94388;52.74;1670685805
Data from TTN: 5;y;49.94299;17.90676;0.13;1670686165
```

Obrázek 16 a 17: Přijaté data přes LoRaWAN síť



Obrázek 18: Přijaté data potom co se zařízení připojilo k internetu pomocí WiFi

3.2 Software a serverový backend

Pro backendový runtime jsem zvolil Node.JS, jedná se o aplikaci, která pracuje s JavaScript jazykem, kde jste schopni napsat backend webových aplikací, potřebující asynchronní funkce.

Server po startu zprovozní SSL zabezpečený webový server s přihlašovacími údaji a komunikaci s TTN pomocí MQTT.

```
root@ns89485:~/WebServer# node main.js
HTTP server listening on port 80
HTTPS server listening on port 433
Connected to TTN MQTT
```

Obrázek 19: Nastartovaný Node.JS server

3.2.1 Zpracovávání dat

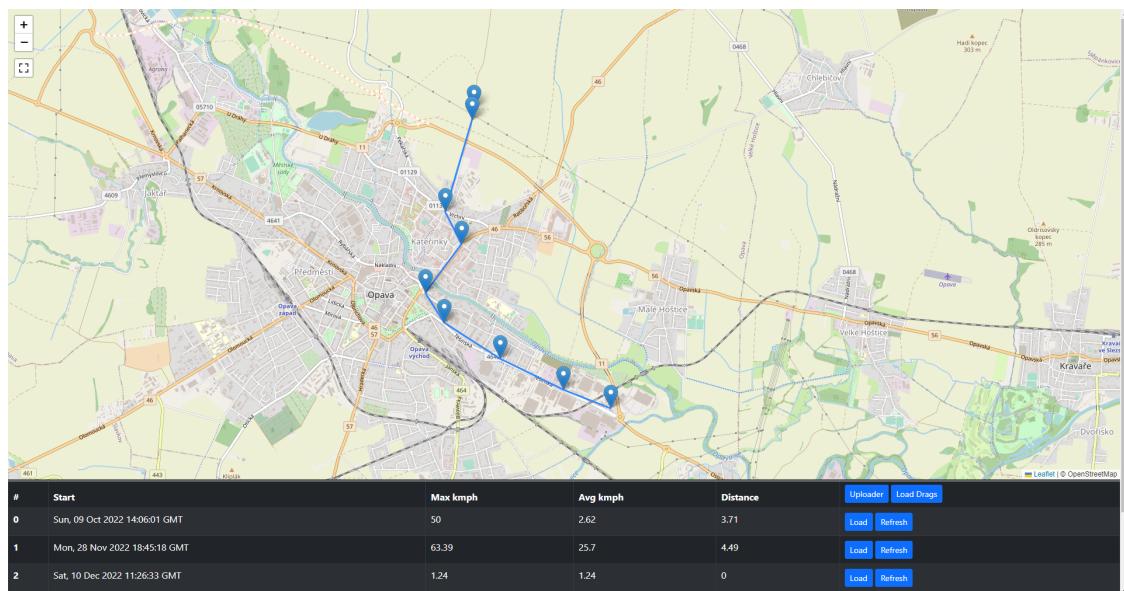
Server dále čeká na přijetí dat pomocí MQTT nebo POST dotazu, přijaté data dekóduje a zjistí zda se jedná o drag nebo sledovací data. Drag data se jednoduše vloží do příslušného JSON souboru, ale sledovací data po dekódování jsou vložena do dočasného pole z kterého jsou postupně předávány do funkce pro zpracování pomocí `setInterval()`. Tato funkce zformátuje dostupné data a vloží je podle timestampu do správné pozice v patřičném JSON souboru. Pokud záznam o potřebném souboru neexistuje, soubor je vytvořen a záznam o jeho existenci také do indexačního JSON souboru obsahující data jako maximální a průměrná rychlosť a ujetá vzdálenost. Tyto data jsou vypočítána ze všech zatím dostupných bodů vždy po zpracování nových dat.

3	Sat, 10 Dec 2022 11:34:02 GMT	94.23	17.68	3.19	Load	Refresh
4	Sat, 10 Dec 2022 11:49:03 GMT	130.84	65.35	34.23	Load	Refresh

Obrázek 20: Data z indexačního JSON souboru obsahující informace jako počáteční čas, maximální a průměrnou rychlosť a celkovou vzdálenost trasy.

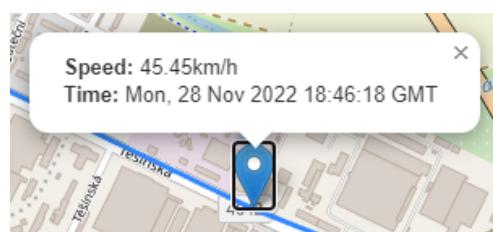
3.2.2 Uživatelské rozhraní

Uživatelské GUI dostupné na veřejné webové adrese je velmi jednoduché, obsahující jenom interaktivní mapu a seznam všech tras s informacemi spolu s tlačítky umožňující načtení trasy nebo přepočítání informačních dat.



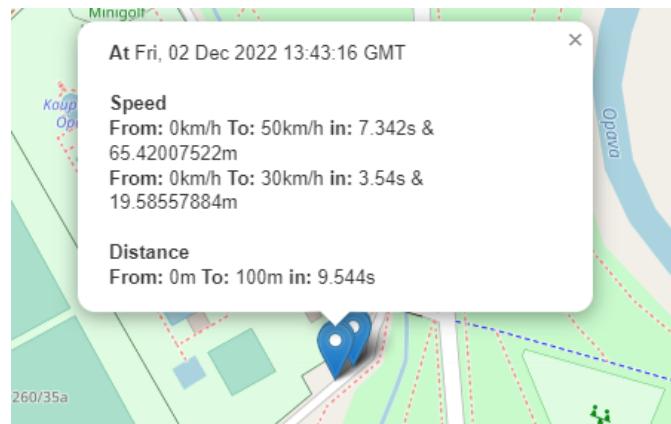
Obrázek 21: Webové rozhraní projektu

Po načtení trasy se získané body dekódují a zobrazí na mapě. Pokud body nejsou nijak propojeny modrou čarou, což znamená, že vozidlo stálo na místě dostatečně dlouho po ukončení minulé trasy. Naopak pokud jsou body propojené modrou čarou, znamená to, že bylo vozidlo v daném bodě v pohybu a upřesňuje směr jízdy s ostatními body. Při kliknutí na jakýkoliv bod se zobrazí nápoveda obsahující specifické údaje o daném bodu.



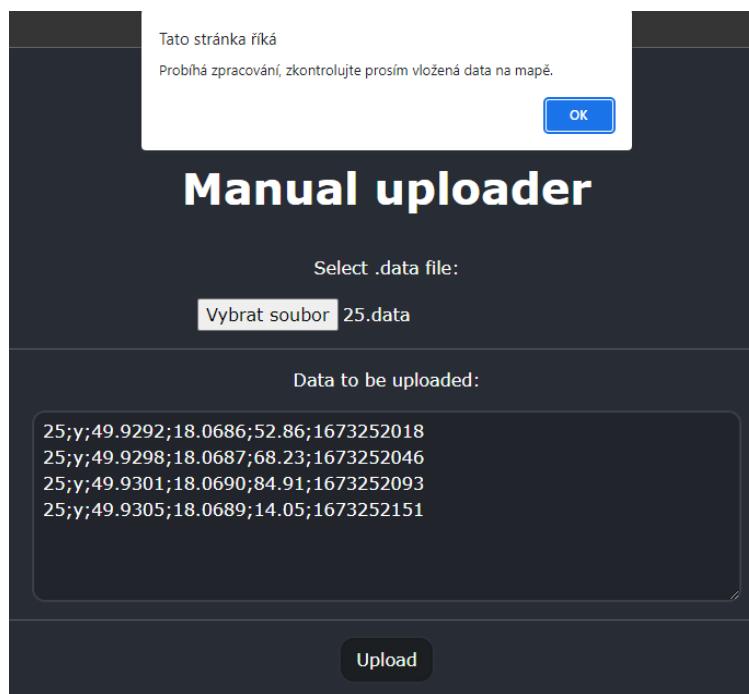
Obrázek 22: Přidružené data k bodu na mapě v sledovacím módu

Je možné také načíst body všech zaznamenaných testů zrychlení pomocí tlačítka "Load Drags". Na mapě se objeví jednotlivé body určující startovní body. Po kliknutí na jednotlivé body se zobrazí návod se všemi dostupnými informacemi.



Obrázek 23: Přidružené data k bodu na mapě v DRAG módu

Pokud nastane problém s nahráváním dat na server, je možno použít manuální nahrávací stránku, která přečte data ze souboru (je možné je také ručně napsat, vložit či upravit) a pošle je pro zpracování na server.



Obrázek 24: Stránka pro manuální vložení dat na server

4 VÝSLEDKY ŘEŠENÍ A VÝSTUPY

4.1 Funkce práce

Zařízení v sledovacím módu pracuje plně samostatně až na první zapnutí, kdy uživatel musí nastavit určité parametry jako jméno a heslo WiFi ESP hotspotu a routeru s přístupem k internetu, také adresu veřejného webového serveru. Přístupové údaje LoRa jsou zabudované do kódu při komplikaci, do budoucna je v plánu je zakomponovat do stávajícího konfiguračního souboru. Zařízení nyní při každém spuštění zkalibruje pohybový senzor a čeká na pohyb, přitom se snaží připojit k LoRa a WiFi síti v předem určeném intervalu. Zařízení čeká na pohyb a pokud zaznamená tento pohyb pomocí senzoru nebo rychlosti GPS, ověří zda se jedná o delší trasu než jenom např. přeparkování. Začne zaznamenávat GPS data do nového souboru na SD kartě a také je občas vysílat prostřednictvím LoRaWAN sítě. Pokud je vozidlo zastaveno po delší dobu než několik minut, je momentální relace ukončena a opětovně se čeká na pohyb.

Zařízení v DRAG módu vypne veškeré funkce sledovacího módu a čeká na vstup dat od uživatele. Po zpracování poskytnutého JSON dotazu začne čekat na pohyb nebo ruční start a poté zařízení spustí časomíru a zpracovává dosažené body do upraveného originálního JSON dotazu. Tento upravený JSON s aktuálními daty je zasílán v intervalu zpátky uživateli pro zobrazení. Po ukončení posledním rychlostním bodem, vzdáleností nebo manuálně, je vytvořen závěrečný JSON a ten je uložen na SD kartu.

Serverová aplikace zpracovává přijaté data ze zařízení a umožňuje jejich libovolné zobrazení. Naslouchá na MQTT brokeru od TTN pro přijaté LoRa zprávy, které server dekóduje, zpracuje a uloží do příslušného JSON souboru. To samé se stane s daty přijatými na určité adresu při POST dotazu, což je využito zařízením, které po připojení k internetu nahraje všechny nezaznamenané data z SD karty na server. Aplikace také hostuje zabezpečenou stránku pro interaktivní zobrazení všech dostupných dat na mapě jako propojené body, podobná verze se také nachází přímo na zařízení, ale zobrazuje dostupné data z SD karty a funguje jenom pokud máte také přístup k internetu.

4.2 Splněné a nesplněné cíle

Hlavní cíl projektu bylo vytvořit sledovací zařízení zaznamenávající GPS data, tento cíl jsem splnil nad své očekávání spolu s dalšími prvky jako měření zrychlení, což byl vedlejší cíl.

Kód sám o sobě není perfektní a určitě by se dal vylepšit jak backend, tak i frontend a usnadnit každodenní používání zařízení (QOL), např. nativní aplikací pro mobilní telefon. Další budoucí cíle vhodné zhotovení jsou zakomponování GSM komunikace do celého projektu, díky které vznikne hlavní druh komunikace a LoRa se stane záložní nebo také přechod z TTN na Helium, které má větší pokrytí. Vylepšení by si taky zasloužil server zpracovávající data, aby místo JSON souborů používal databázi MySQL pro ukládání dat a také nahradit některé funkce využívající POST dotazu MQTT brokerem.

ZÁVĚR

Cílem práce bylo vytvořit a vyrobit zařízení schopné zaznamenávání pohybu auta a odesílání těchto dat pomocí LoRa sítě, dále také měření zrychlení auta a ujeté vzdálenosti. Navíc napsat server pro přijímání a ukládání dat vzdáleně na veřejně přístupné stránce pomocí Node.JS.

Závěrečné zařízení je možné schovat kdekoliv v sledovaném objektu, např. v A sloupu osobních a firemních aut. Zaznamenávají se údaje jako GPS čas, pozice a rychlosť na SD kartu lokálně, ale také se odesílají na vzdálený server pomocí LoRaWAN nebo WiFi připojení. Zaznamenané data je uživatel schopen interaktivně zobrazit na mapě buď lokálně na ESP webu nebo na veřejném serveru.

S prací jsem spokojen, ale chybí mi nedodělané věci jako GSM komunikace pokud v blízkosti není LoRa gateway nebo možnost využití Helium místo TTN.

Zdrojový kód a PCB design se nachází na GitHub adrese: <https://github.com/ncplyn/carstatstracker>

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

- [1] Benoît Blanchon, *ArduinoJson Documentation v6* [online] [cit. 2022-12-30]
Dostupné z: <https://arduinojson.org/v6/doc/>
- [2] NC Plyn, *Protoprotocol-Helmet-ESP32* [online] [cit. 2022-12-30]
Dostupné z: <https://github.com/NC Plyn/Protoprotocol-Helmet-ESP32>
- [3] NC Plyn, *Lidl4chan* [online] [cit. 2022-12-30]
Dostupné z: <https://github.com/NC Plyn/Lidl4chan>
- [4] GyverLibs, *UnixTime* [online] [cit. 2022-12-30]
Dostupné z: <https://github.com/GyverLibs/UnixTime/blob/main/src/UnixTime.h>
- [5] Nina Scholz, *Insert json object to ordered array in javascript* [online] [cit. 2022-12-30]
Dostupné z: <https://stackoverflow.com/a/38418979>
- [6] Espressif Systems contributors, *WiFiClientSecure* [online] [cit. 2022-12-30] Dostupné z:
<https://github.com/espressif/arduino-esp32/tree/master/libraries/WiFiClientSecure>
- [7] bcardarella, *Generating a self-signed cert with openssl that works in Chrome 58* [online]
[cit. 2022-12-30] Dostupné z: <https://serverfault.com/a/845788>
- [8] Anh Quân Tông, *Create a NodeJS server to get data from TTN via MQTT* [online]
[cit. 2022-12-30] Dostupné z:
https://drive.google.com/drive/folders/1StPhZQ4ArbOQpxu7d_64jj_qod6Y3bn
- [9] Volodymyr Agafonkin, *Leaflet Tutorials* [online] [cit. 2022-12-30]
Dostupné z: <https://leafletjs.com/examples.html>
- [10] me-no-dev, *ESPAsyncWebServer Documentation* [online] [cit. 2022-12-30]
Dostupné z: <https://github.com/me-no-dev/ESPAsyncWebServer>
- [11] beegee-tokyo, *SX126x-Arduino* [online] [cit. 2022-12-30]
Dostupné z: <https://github.com/beegee-tokyo/SX126x-Arduino>
- [12] Mikal Hart, *TinyGPS++* [online] [cit. 2022-12-30]
Dostupné z: <http://arduiniana.org/libraries/tinygpsplus/>