



# LiveMetro

Webové aplikace

Martin Chmelík, Jan Lukeš, Petr Valenta, Timothy Laurich

2025

# Obsah

## Obsah

Obsah .....	2
Úvod.....	3
Rozbor .....	3
System.....	3
Programování: .....	3
<b>Backend</b> .....	4
<b>Frontend</b> .....	5
Testování .....	7
Spuštění/Nasazení .....	7
Nasazení: .....	7
Spuštění: .....	7
Github .....	7

# Úvod

Cílem této práce bylo vytvořit webovou aplikaci, která zobrazuje živou polohu souprav pražského metra na mapě. Aplikace umožňuje sledovat jednotlivé linky metra A, B a C, jejich stanice a aktuální pohyb vlaků v reálném čase. Data jsou získávána z otevřeného rozhraní Golemio a jsou uživateli zobrazována pomocí webového rozhraní s využitím technologie WebSocket.

## Rozbor

Projekt je rozdělen na serverovou a klientskou část. Serverová část se stará o získávání dat z externího API, jejich zpracování, převod geografických souřadnic na souřadnice odpovídající mapě a odesílání těchto dat připojeným klientům. Klientská část má na starosti vykreslení mapy metra, zobrazení stanic a animované zobrazení pohybu jednotlivých vlaků. Uživatel může pomocí myši získat doplňující informace o vlacích a stanicích.

## System

Aplikace je provozována jako webová aplikace běžící na Node.js serveru. Klient k aplikaci přistupuje prostřednictvím běžného webového prohlížeče.

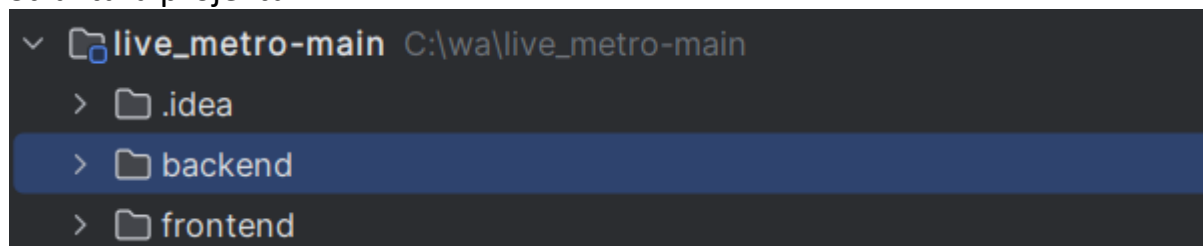
Použité technologie:

- Node.js
- Express.js
- WebSocket (knihovna ws)
- HTML
- CSS
- JavaScript
- API Golemio (API hlavního města Prahy)

Server pravidelně aktualizuje data o poloze vlaků a pomocí WebSocket spojení je rozesílá všem připojeným klientům.

## Programování:

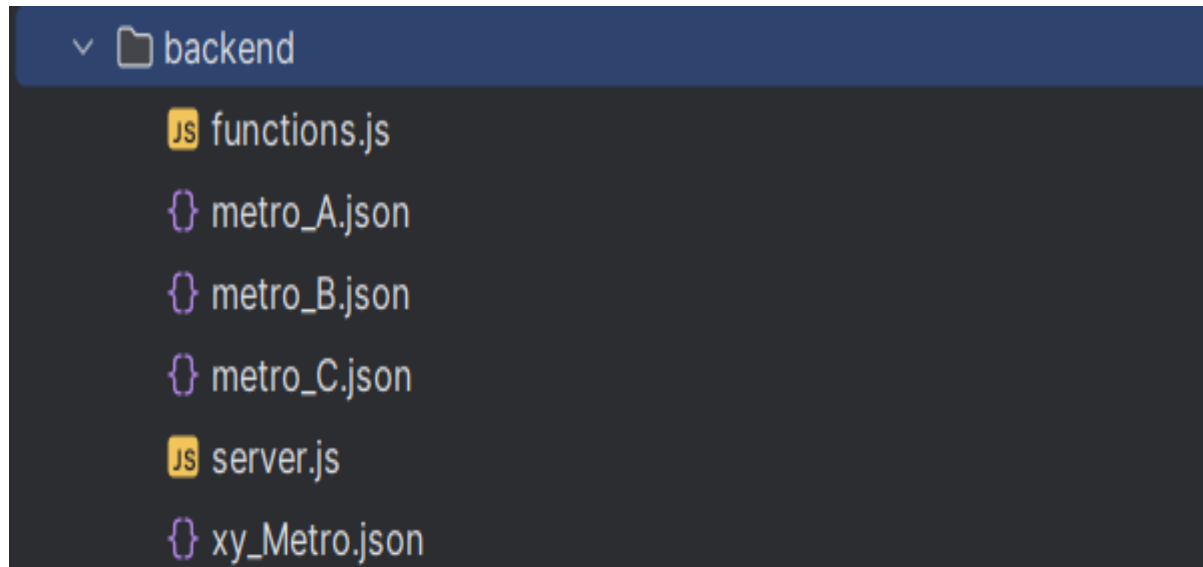
Struktura projektu:



*Rozdělení na backend a frontend*

Projekt je rozdělen na backend a frontend.

## Backend



Struktura backendu

Do souborů metro\_A.json, metro\_B.json, metro\_C.json se ukládají údaje o aktuálních pozicích meter podle každé linky v souřadnicích. Do souboru se xy\_Metro.json ukládají souhrnné údaje o všech linkách metra v pixelech (x a y). Pro všechny soubory platí, že metra se do souborů zapisují jen v momentě, kdy jsou vidět (někam jedou), když dorazí na konečnou přestanou se ukládat. (API už neposkytuje data).

### function.js

```
/** Tento skript zajišťuje funkce pro pravidelné stahování dat o poloze vlaků pražského metra z API, převod jejich souřadnic na pixely a ukládání do JSON souborů pro další vizualizaci. */
```

```
function coordToXY(lat, lon)
```

```
/** Funkce převádí zeměpisné souřadnice (šířku a délku) na souřadnice v pixelech odpovídající obrázku mapy. Výsledek je objekt obsahující x a y hodnoty pro vizualizaci na mapě. */
```

```
export function all_metro()
```

```
/** Tato funkce načítá data o poloze vlaků z lokálních JSON souborů a převádí je na pixelové souřadnice pro všechny linky metra A, B a C. Výsledná data ukládá do souboru xy_Metro.json a zároveň je vrací pro další použití. */
```

```
export async function fetchMetroCoordinates()
```

```
/** Asynchronní funkce získává aktuální polohu vlaků z API Golemio pro všechny tři linky metra. Data uloží do lokálních JSON souborů (metro_A.json, metro_B.json, metro_C.json) pro další zpracování. */
```

## server.js

```
/**Tento skript spouští HTTP a WebSocket server, rozesílá je všem
připojeným klientům a spouští loop ve kterém se spouští jejich data.
**/

function broadcast(data)

/** Na vstup přichází JSON a posílá je všem klientům připojeným přes
WebSocket. Zajišťuje, že pouze aktivní klienti dostanou aktualizace.
**/

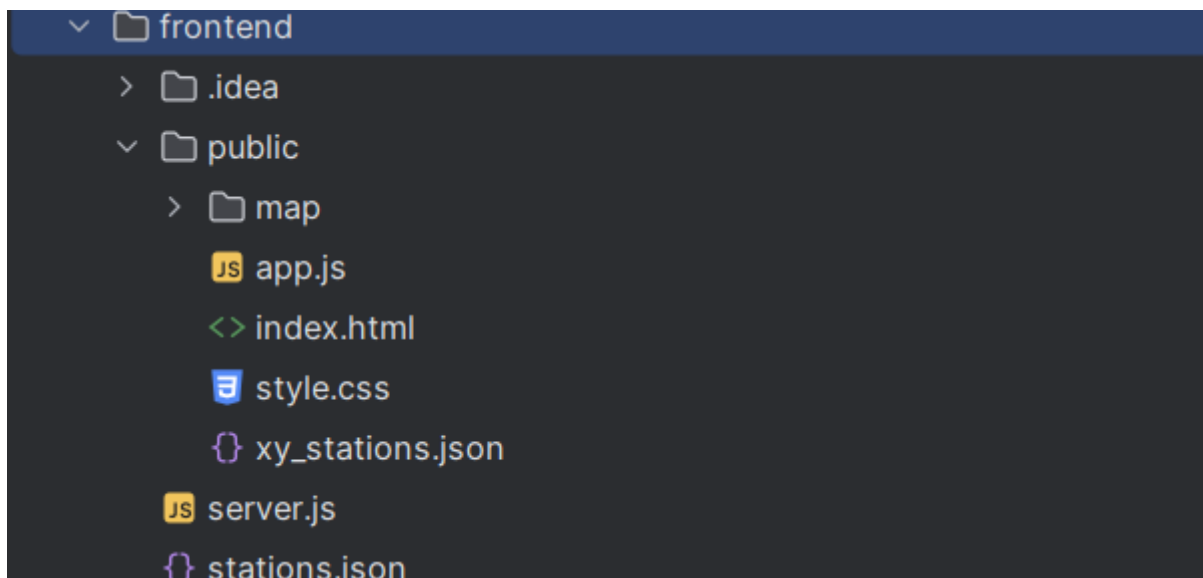
async function updateLoop()

/** Asynchronní funkce, která pravidelně volá fetchMetroCoordinates a
all_metro a následně aktualizovaná data rozesílá klientům. Cyklus se
opakuje každých 10 sekund. **/

wss.on('connection', ws => { ... })

/** Obsluha nového připojení klienta přes WebSocket. Pokud existuje
soubor xy_Metro.json, pošle jeho obsah okamžitě novému klientovi, aby
měl aktuální data ihned po připojení. **/
```

## Frontend



Struktura frontendu

V souboru stations.json jsou uloženy všechny stanice s jejich souřadnicemi. V souboru xy\_stations.json jsou uloženy všechny stanice s x a y polohou v pixelech. Style.css obsahuje styly a v index.html jsou dvě vrstvy. Spodní vrstva je plánec pražských linek metra a v horní vrstvě se vykreslují vlaky a stanice.

## app.js

```
/** Tento skript zajišťuje vizualizaci polohy vlaků a stanic pražského metra na webové stránce, přijímá data přes WebSocket a animuje pohyb vlaků na mapě. */
```

```
function updateMetroPositions(data)
```

```
/** Funkce zpracovává nová data o poloze vlaků pro všechny linky A, B a C a předává je funkci updateLine, která provede aktualizaci vizualizace. */
```

```
function updateLine(lineName, newPositions, lineClass)
```

```
/** Funkce aktualizuje pozice vlaků na konkrétní lince, animuje pohyb existujících vlaků a přidává nové vlaky do mapy. Vlaky, které se již nevyskytují, jsou odstraněny. */
```

```
function renderStations(data)
```

```
/** Funkce vykresluje všechny stanice metra podle dat z JSON souboru a ukládá jejich DOM elementy do mapy pro pozdější interakci. */
```

```
function createStationElement(station, lineClass)
```

```
/** Funkce vytváří HTML element pro jednotlivou stanici, nastavuje její pozici a styl a přidává interakci s tooltipem při najetí myši. */
```

```
function createTrainElement(coord, lineClass)
```

```
/** Funkce vytváří HTML element pro vlak, nastavuje jeho pozici, vzhled a interakci s tooltipem, a přidává ho do mapy metra. */
```

```
function animateTrainTo(element, newCoord)
```

```
/** Funkce animuje pohyb vlakového elementu z aktuální pozice na novou pozici pomocí CSS přechodu a aktualizuje jeho tooltip s aktuálními souřadnicemi. */
```

## server.js

```
/** Tento skript spouští proxy WebSocket server, který přijímá data z externího WebSocket serveru metra a přeposílá je všem připojeným klientům. */
```

```
metroWS.on("open", ...)
```

```
/** Událost se spustí při úspěšném připojení k externímu WebSocket serveru metra a vypíše potvrzení do konzole. */
```

```
metroWS.on("message", ...)
```

```
/** Událost se spustí při příjmu dat od externího WebSocket serveru, zpracuje JSON a rozesílá jej všem klientům připojeným k tomuto proxy serveru. */
```

```
wss.on("connection", ...)
```

```
/** Událost se spustí při připojení nového klienta k proxy WebSocket serveru a pošle uvítací zprávu. */
```

# Testování

Testování aplikace probíhalo manuálně během vývoje. Byla ověřena správná funkčnost připojení k API, WebSocket komunikace mezi serverem a klientem, vykreslení stanic a pohyb vlaků.

Aplikace byla testována v běžných webových prohlížečích.

## Spuštění/ Nasazení

### Nasazení:

Pro spuštění aplikace je nutné mít nainstalované prostředí Node.js. Po stažení projektu je potřeba nainstalovat závislosti pomocí příkazu `npm install`. Potom v obou složkách jak backendu tak frontendu v terminalu zadat `node server.js`

### Spuštění:

Aplikace se spustí pomocí příkazu `node server.js`. Následně je možné aplikaci otevřít ve webovém prohlížeči na adrese <http://localhost:3001/>.

## Github

[https://github.com/Bendy545/live\\_metro](https://github.com/Bendy545/live_metro)

