

RailPlot

Vasút rendszer tervezői program



Készítette:

Körtélyesi Bálint Máté

Zarapkar Joshua Anand

RailPlot - Felhasználói Dokumentáció.....	3
1. Bevezetés	3
2. Rendszerkövetelmények.....	4
3. Telepítés és Indítás.....	5
4. Regisztráció és Bejelentkezés	6
5. A Program Használata (Fő Felület - main.blade.php).....	9
6. Hibajelzések és Problémamegoldás.....	16
7. Adminisztrátori Felület (index.blade.php - User & Train Management).....	18
8. Elérhetőség, Támogatás.....	20
RailPlot - Fejlesztői Dokumentáció (Integrált és Bővített Változat)	21
1. Bevezetés	21
2. Fejlesztői Környezet.....	22
3. Technológiai Választások Indoklása.....	23
4. Adatszerkezet (Migrációk és Feltételezések Alapján).....	26
5. Tipikus Algoritmusok és Adatfeldolgozási Folyamatok.....	28
6. Tesztelés.....	31
7. Tesztelői Hozzáférés Biztosítása	35
8. Fejlesztési Lehetőségek.....	36
9. Összefoglalás.....	38
10. Irodalomjegyzék	

RailPlot - Felhasználói Dokumentáció

1. Bevezetés

- **A program bemutatása:** Üdvözljük a RailPlot alkalmazásban! Ez egy innovatív, böngészőben futó szoftver, amelyet arra terveztek, hogy intuitív és hatékony eszközt biztosítson vasúti hálózatok vizuális megtervezéséhez és kezeléséhez. A RailPlot segítségével könnyedén létrehozhat és módosíthat vasútvonalakat, elhelyezhet állomásokat egy interaktív, nagyítható és mozgatható térképen (vászon), és nyomon követheti a hálózatának főbb jellemzőit a beépített statisztikai panelen. Az alkalmazás célja, hogy a komplex tervezési folyamatot egyszerűbbé és átláthatóbbá tegye.
- **A témaválasztás:** A fő motiváció az volt, hogy a piacon elérhető professzionális szoftverek gyakran túlságosan komplexek, drágák, vagy platformfüggőek lehetnek a hobbi felhasználók, diákok vagy kisebb léptékű tervezési feladatok számára. A RailPlot egy könnyen hozzáférhető, intuitív felületet kíván biztosítani, amely lehetővé teszi a felhasználók számára, hogy gyorsan és hatékonyan hozzanak létre, módosítsanak és elemezzenek hálózati struktúrákat.
- **Célközönség:** Kinek ajánljuk a RailPlot-ot? Elsősorban vasútmodellezőknek, akik saját terepasztalukhoz vagy virtuális világhoz terveznek hálózatot; közlekedésmérnök-hallgatóknak és szakembereknek, akik tanulmányaikhoz vagy kisebb projektekhez keresnek egy könnyen használható vizualizációs eszközt; várostervezéssel foglalkozóknak, akik a közösségi közlekedési hálózatok tervezésének korai fázisában szeretnének koncepciókat kidolgozni; valamint minden olyan hobbi felhasználónak, aki érdeklődik a vasutak és a hálózattervezés világa iránt.

2. Rendszerkövetelmények

Annak érdekében, hogy a RailPlot zökkenőmentesen fusson, az alábbiakra lesz szüksége:

- **Hardver:**

- **Eszköz:** Egy asztali számítógép vagy laptop stabil internetkapcsolattal. Bár az alkalmazás reszponzív lehet, a tervezési funkciók kényelmes használatához nagyobb képernyő javasolt.
- **Processzor és Memória:** Nincsenek szigorú megkötések, de egy modern, legalább Intel Core i3 vagy azzal egyenértékű AMD processzor, és minimum 4 GB RAM ajánlott a gördülékeny működéshez, különösen nagyobb hálózatok tervezése esetén.
- **Perifériák:** Egér használata kifejezetten ajánlott a vásznon történő precíz navigációhoz (nagyítás, mozgatás) és az elemek (állomások, vonalak) pontos elhelyezéséhez és szerkesztéséhez. Érintőképernyőn a használat korlátozott lehet.

- **Szoftver:**

- **Operációs rendszer:** Bármilyen modern asztali operációs rendszer megfelelő (Windows 10 vagy újabb, macOS, Linux disztribúciók).
- **Webböngésző:** A legfrissebb verziója az alábbi böngészők egyikének: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari. A böngészőnek támogatnia kell a HTML5 Canvas technológiát és a JavaScriptet (ezek általában alapértelmezetten engedélyezve vannak).
- **Internetkapcsolat:** Aktív internetkapcsolat szükséges az alkalmazás betöltéséhez és használatához, valamint az adatok mentéséhez és betöltéséhez.
- **Egyéb szoftver:** Nincs szükség további szoftverek telepítésére a felhasználó gépére az alkalmazás használatához.

3. Telepítés és Indítás

1. Szükséges eszközök

A RailPlot alkalmazás használatához a következő szoftverek szükségesek a számítógépen:

- **PHP** (minimum 8.1-es vagy újabb verzió),
- **Composer** (PHP csomagkezelő),
- **Node.js** és **npm** (Node Package Manager),
- **XAMPP** vagy más webservert és **MySQL** adatbáziskezelő, ha adatbázis is szükséges.

2. Előkészületek

Először **töltse le** vagy **klónozza** a projektet a számítógépére.

Ezután navigáljon a projekt **backend** mappájába az alábbi paranccsal:

```
cd RailPlot2/backend
```

3. Backend telepítése

A backend mappában telepítse a szükséges **PHP csomagokat** a következő paranccsal:

```
composer install
```

```
php artisan migrate
```

4. Frontend telepítése

Telepítse az **npm csomagokat**:

```
npm install
```

Ezután indítsa el a **Vite fejlesztői szerverét**:

```
npm run dev
```

Ez a parancs elindítja a **frontend assetek** (CSS, JS) kezelését.

5. Laravel szerver indítása

Indítsa el a **Laravel belső szerverét**:

```
php artisan serve
```

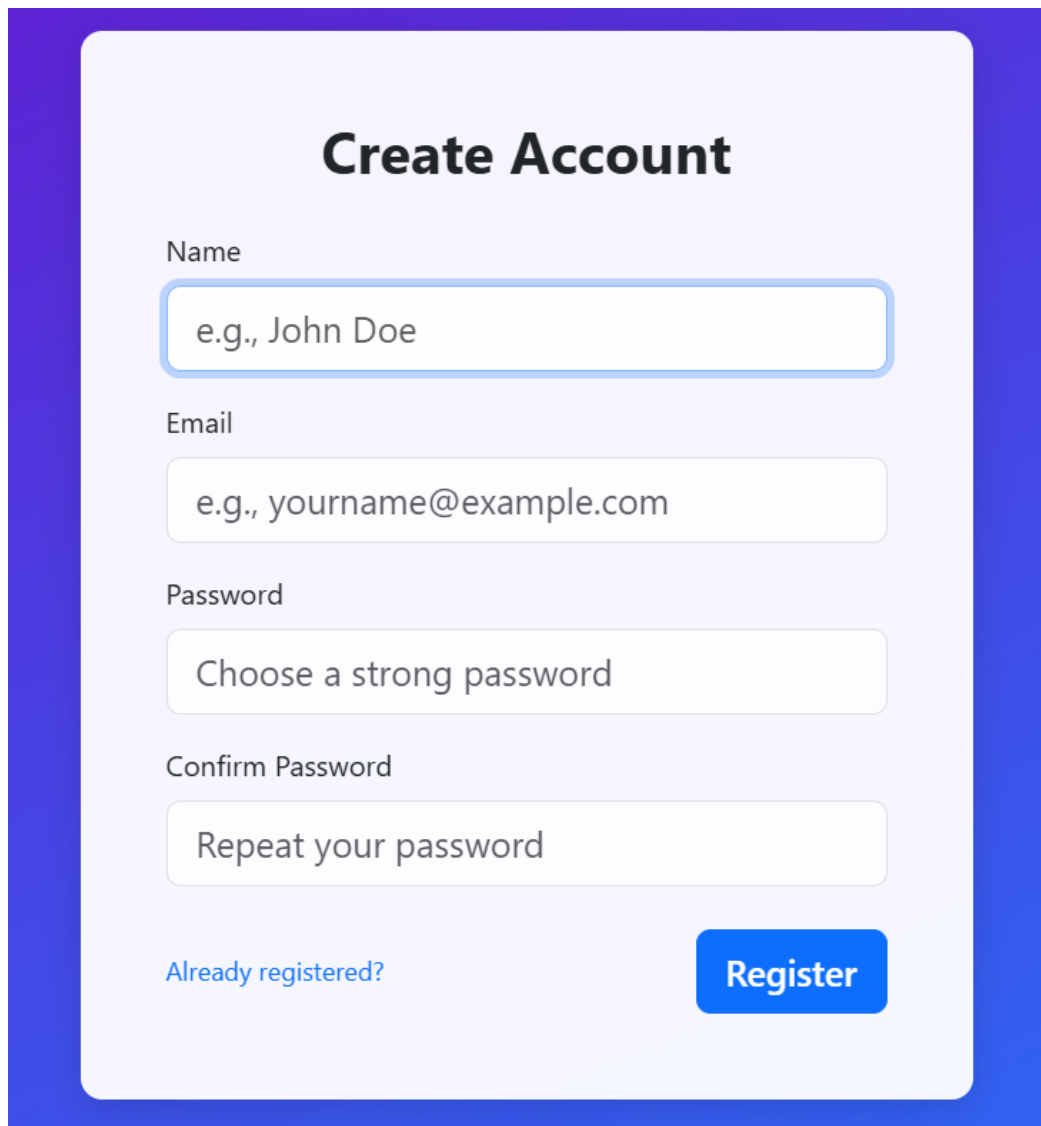
A szerver elindítása után az alkalmazás elérhető lesz itt:

```
http://localhost:8000
```

4. Regisztráció és Bejelentkezés

A RailPlot legtöbb funkciójának eléréséhez felhasználói fiókra van szükség.

- **Szabadon elérhető funkciók:** Elképzelhető, hogy bizonyos alapvető információk, bemutató oldalak vagy példa hálózatok megtekinthetők bejelentkezés nélkül is, de a saját hálózatok létrehozása, szerkesztése és mentése regisztrációhoz/bejelentkezéshez kötött. (A pontos korlátozások a szerveroldali logikától függnék).
- **Regisztráció (register.blade.php):**
 1. Kattintson a bejelentkezési oldalon található "Sign Up" linkre, vagy navigáljon közvetlenül a regisztrációs oldalra.
 2. Töltse ki a regisztrációs űrlapot:
 - **Név (Name):** Adja meg a teljes nevét. Általában betűket, számokat és szóközöket tartalmazhat. Kerülje a speciális karaktereket, hacsak nincs másképp jelezve.
 - **E-mail (Email):** Adjon meg egy valós, működő e-mail címet. Erre a címre kaphat fontos értesítéseket, és ezt használhatja a fiókjához való hozzáféréshez vagy jelszó-visszaállításhoz. Az e-mail cím formátuma szabványos (felhasznalo@domain.hu). Ékezetes karakterek általában nem megengedettek.
 - **Jelszó (Password):** Válasszon egy erős, nehezen kitalálható jelszót. Javasolt kis- és nagybetűk, számok és speciális karakterek kombinációját használni (pl. !@#\$\$%^&*). A rendszer általában elvár egy minimális jelszóhosszt (pl. 8 karakter), de ez a konkrét implementációtól függ.
 - **Jelszó megerősítése (Confirm Password):** Írja be újra pontosan ugyanazt a jelszót, amit az előző mezőbe írt, az elgépelések elkerülése végett.
 3. Kattintson a "Register" vagy "Create Account" gombra.
 4. Sikeres regisztráció esetén a rendszer általában átirányítja a bejelentkezési oldalra vagy automatikusan bejelentkezteti. Előfordulhat, hogy e-mailben kap egy megerősítő linket, amelyre kattintva aktiválnia kell a fiókját (ezt a users tábla email_verified_at mezője sugallja).

A 'Create Account' form with a blue border. It contains four input fields: 'Name' with placeholder 'e.g., John Doe', 'Email' with placeholder 'e.g., yourname@example.com', 'Password' with placeholder 'Choose a strong password', and 'Confirm Password' with placeholder 'Repeat your password'. At the bottom left is a blue link 'Already registered?' and at the bottom right is a blue 'Register' button.

Create Account

Name

Email

Password

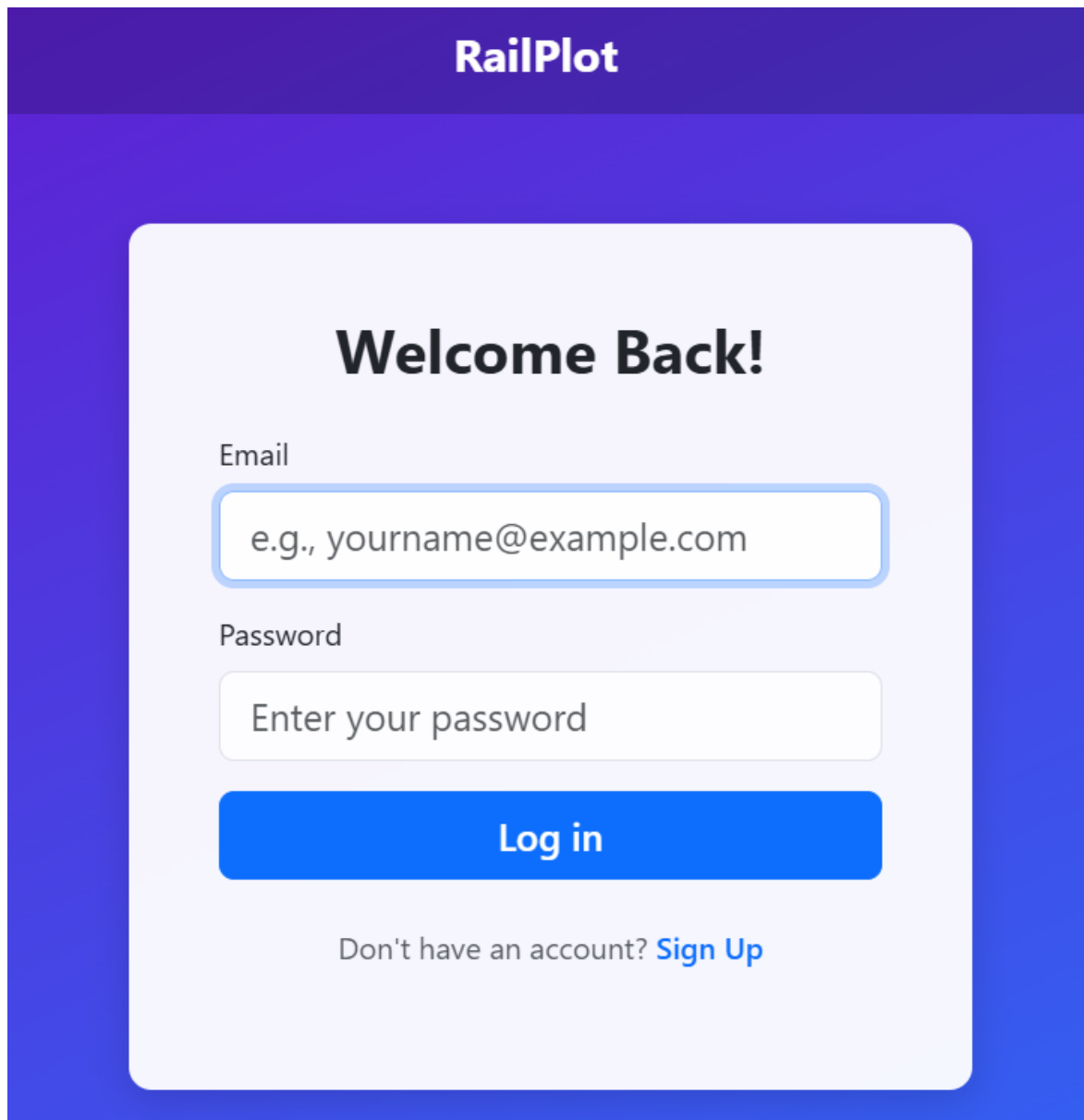
Confirm Password

[Already registered?](#)

- **Bejelentkezés (login.blade.php):**

1. Navigáljon a bejelentkezési oldalra.
2. Adja meg a regisztrációkor használt e-mail címét (Email) és jelszavát (Password). Ügyeljen a kis- és nagybetűk megkülönböztetésére a jelszóban.
3. **"Remember me" opció:** Ha bejelöli ezt a négyzetet, a böngésző egy ideig megjegyzi a bejelentkezési állapotát (egy remember_token segítségével), így legközelebb nem kell újra beírnia az adatait (csak ha kijelentkezik, vagy a munkamenet lejár). Ezt a funkciót csak saját számítógépen használja.
4. Kattintson a "Log in" gombra.
5. Sikeres bejelentkezés után a rendszer a fő alkalmazás felületére (main.blade.php) vagy a műszerfalra irányítja.

6. **Elfelejtett jelszó:** Ha elfelejtette jelszavát, kattintson a "Forgot your password?" linkre (ha elérhető). Ez általában egy folyamatot indít el, ahol az e-mail címére kapott utasítások segítségével új jelszót állíthat be (a password_reset_tokens tábla használatával).



The image shows a login interface for 'RailPlot'. It features a dark blue header with the 'RailPlot' logo. Below the header is a light blue box containing the login form. The form has a 'Welcome Back!' heading, followed by 'Email' and 'Password' labels. The email input field contains the placeholder 'e.g., yourname@example.com'. The password input field contains the placeholder 'Enter your password'. Below the password field is a blue 'Log in' button. At the bottom of the form, there is a link that says 'Don't have an account? Sign Up'.

RailPlot

Welcome Back!

Email

Password

Log in

Don't have an account? [Sign Up](#)

5. A Program Használata (Fő Felület - main.blade.php)

Bejelentkezés után a fő tervezőfelület fogadja, amely több részből áll:

- **Navigációs Sáv (Felső menüsor):** Gyors hozzáférést biztosít a legfontosabb funkciókhoz.
 - RailPlot (Logóval): Visszatérés a kezdő nézethez vagy a főoldalra (ha van ilyen).
 - Create Line: Megnyitja a "Create Line" modális ablakot (felugró ablakot) egy új vasútvonal létrehozásához és konfigurálásához.
 - Export: Lehetővé teszi az aktuálisan szerkesztett hálózat adatainak lementését a saját számítógépére. Az exportálás valószínűleg JSON formátumban történik, ami egy szöveges adatcsere formátum, így a hálózat könnyen megosztható vagy később visszatölthető.
 - Import: Megnyit egy fájlválasztó ablakot, ahol kiválaszthat egy korábban exportált JSON fájlt (.json kiterjesztésű). A kiválasztott hálózat betöltődik a vászonra, felülírva az aktuális tartalmat.
 - Clear Grid: Eltávolít minden állomást és vonalat az aktuális vászonról, lényegében tiszta lappal indulhat. Ez a művelet általában nem vonható vissza, ezért körültekintően használja.
 - Refocus: Visszaállítja a vászon nézetét az alapértelmezett nagyítási szintre és középre igazítja a tartalmat (vagy egy előre definiált kezdőpontra). Hasznos, ha "elveszett" a nagyítás vagy mozgatás során.
 - Show Lines: Megnyit egy oldalsó panelt (offcanvas menüt), amely listázza az aktuális hálózatban szereplő összes vonalat. Innen valószínűleg kiválaszthat vonalakat szerkesztésre vagy áttekintésre.
 - Log Out: Kijelentkezik az alkalmazásból, visszairányítva a bejelentkezési oldalra. A nem mentett változások elveszhetnek!
- **Vászon (canvas):** Ez az alkalmazás szíve, egy nagy, interaktív terület, ahol a vasúti hálózat vizuálisan megjelenik.
 - **Navigáció:** Az egeret használva (vagy érintőképernyőn gesztusokkal) mozgathatja (panning) és nagyíthatja/kicsinyítheti (zooming) a térképet a kényelmesebb szerkesztés érdekében.

- **Koordináták:** A vászon felett mozgó egérkurzor aktuális X és Y koordinátái valós időben megjelennek egy kis lebegő ablakban (cursorCoords), segítve az elemek pontos elhelyezését.
- **Nagyítási szint:** Az aktuális nagyítás mértéke (pl. 1.00x, 2.50x) szintén kijelzésre kerül (scaleDisplay).
- **Elemek elhelyezése:** Új állomásokat valószínűleg a vászonra kattintva lehet elhelyezni (esetleg egy eszköztár kiválasztása után). A vonalak az állomások összekötésével jönnek létre.

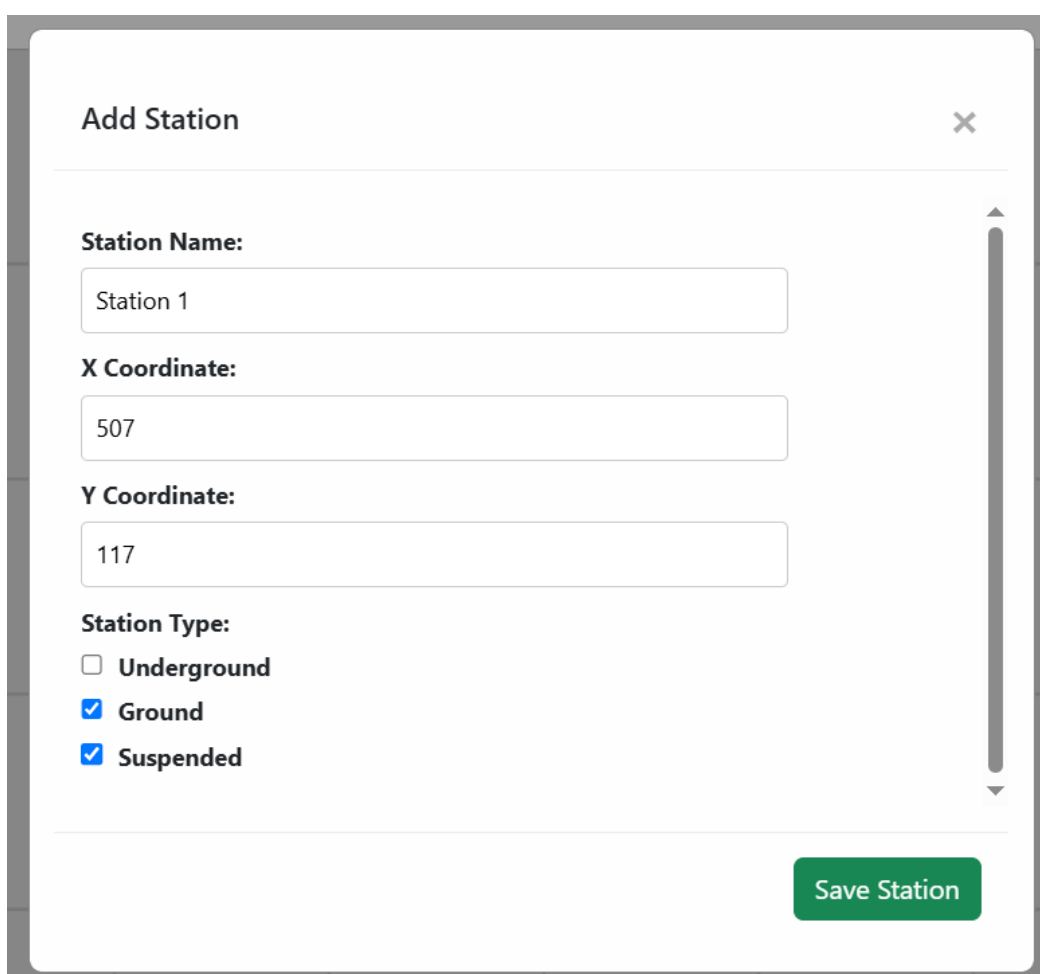


- **Állomások Kezelése (stationModal - Modális Ablak):** Amikor egy új állomást hoz létre, vagy egy meglévőre kattint, ez a felugró ablak jelenik meg a részletek szerkesztéséhez.
 - **Cím:** "Add Station" (új állomásnál) vagy "Edit Station" (meglévőnél).
 - **Station Name:** Szöveges mező az állomás nevének megadására. Célszerű egyedi és informatív neveket választani. Bármilyen szöveg elfogadott lehet, beleértve az ékezetes betűket és szóközöket (pontos megkötések nincsenek a fájlban). Hosszkorlát lehetséges.
 - **X Coordinate, Y Coordinate:** Számbeviteli mezők az állomás pontos vízszintes (X) és függőleges (Y) pozíciójának beállítására a vásznon. Ezek az értékek a vászon belső koordináta-rendszerére vonatkoznak. Lehetnek egész vagy tizedes számok.
 - **Station Type:** Jelölőnégyzetek (checkboxok) az állomás típusának kiválasztásához. Egy állomás lehet egyszerre többféle típusú is (pl. egy felszíni

állomásnak lehetnek földalatti peronjai is). Választható típusok: Underground (földalatti), Ground (földfelszíni), Suspended (függesztett/magasvezetésű).

- **Gombok:**

- **Save Station:** Elmenti az állomás adatait (új létrehozása vagy meglévő módosítása) és bezárja az ablakot.
- **Delete Station:** Véglegesen törli az állomást a hálózathoz. Ez a gomb csak meglévő állomás szerkesztésekor jelenik meg, és általában megerősítést kér.
- **Close (X ikon):** Bezárja az ablakot mentés nélkül.



The image shows a modal window titled "Add Station" with a close button (X) in the top right corner. The form contains the following fields and options:

- Station Name:** A text input field containing "Station 1".
- X Coordinate:** A text input field containing "507".
- Y Coordinate:** A text input field containing "117".
- Station Type:** A section with three radio button options:
 - ☐ Underground
 - ☒ Ground
 - ☒ Suspended

A green "Save Station" button is located at the bottom right of the form.

- **Vonalak Kezelése (addLineModal - Modális Ablak):** Új vonal létrehozásakor (Create Line gomb) vagy egy meglévő vonal szerkesztésekor (pl. a Show Lines listából kiválasztva) ez az ablak nyílik meg.
 - **Cím:** "Create Line" vagy "Edit Line".
 - **Line Name:** Szöveges mező a vonal nevének megadására (pl. "Fővonal", "Kék metró"). Bármilyen szöveg elfogadott lehet, hosszról és karakterektől függően.

- Line Code (2 letters): Rövid, általában 1 vagy 2 karakterből álló azonosító kód a vonalhoz (pl. "M2", "K1"). Csak betűk vagy számok lehetnek engedélyezettek, maximum 2 karakter hosszúságban.
- Line Color: Színválasztó eszköz, amellyel kiválasztható a vonal színe, ahogyan az a vásznon megjelenik. Ez segít a különböző vonalak vizuális megkülönböztetésében.
- Line Type: Rádiógombok (egyszerre csak egy választható) a vonal domináns típusának beállítására: Underground (földalatti), Ground (földfelszíni), Suspended (függesztett). Ez befolyásolhatja a vonal vizuális megjelenítését vagy a hozzárendelhető vonatok típusát.
- Stations: Ez a szekció szolgál az állomások vonalhoz való hozzáadására és sorrendjük beállítására.
 - A + gombokkal lehet újabb állomás-kiválasztó mezőket hozzáadni. Valószínűleg egy legördülő listából lehet kiválasztani a már a vásznon létező állomásokat, vagy esetleg a vásznonról közvetlenül is hozzá lehet adni őket.
 - Az állomások sorrendje általában húzással (drag and drop) vagy nyilakkal módosítható (a fájl ezt nem részletezi).
- Train Assignment: Vonatok hozzárendelése a vonalhoz.
 - Assign Train: Legördülő lista a rendszerben elérhető (az admin által felvitt) vonattípusok közül való választáshoz.
 - Train Quantity: Számbeviteli mező, ahol megadható, hány darab vonat közlekedjen a kiválasztott típusból ezen a vonalon. Az elérhető maximális darabszám (maxTrainQuantity) kijelzésre kerülhet.
- **Gombok:**
 - Save Line: Elmenti a vonal adatait (létrehozás/módosítás) és bezárja az ablakot.
 - Delete Line: Véglegesen törli a vonalat és annak kapcsolatait az állomásokkal. Csak meglévő vonal szerkesztésekor látható, és megerősítést kérhet.
 - Create Express Service: Speciális funkció, amely valószínűleg egy újabb modális ablakot nyit meg (expressServiceModal), ahol a vonal meglévő állomásai közül kiválaszthatók azok, amelyeken az expressz

járat megáll. Ez létrehoz egy új, "expressz" típusú vonalváltozatot.
Csak mentett vonal esetén érhető el.

- Close (X ikon): Bezárja az ablakot mentés nélkül.

Create Line

Line Name: Line_A

Line Code (2 letters): AA

Line Color: [Blue color bar]

Line Type:

- ☐ Underground
- ☒ Ground
- ☐ Suspended

Assign Train: Yamanote (Qty: 60)

Save Line

Create Line

☐ Suspended

Assign Train: Yamanote (Qty: 60)

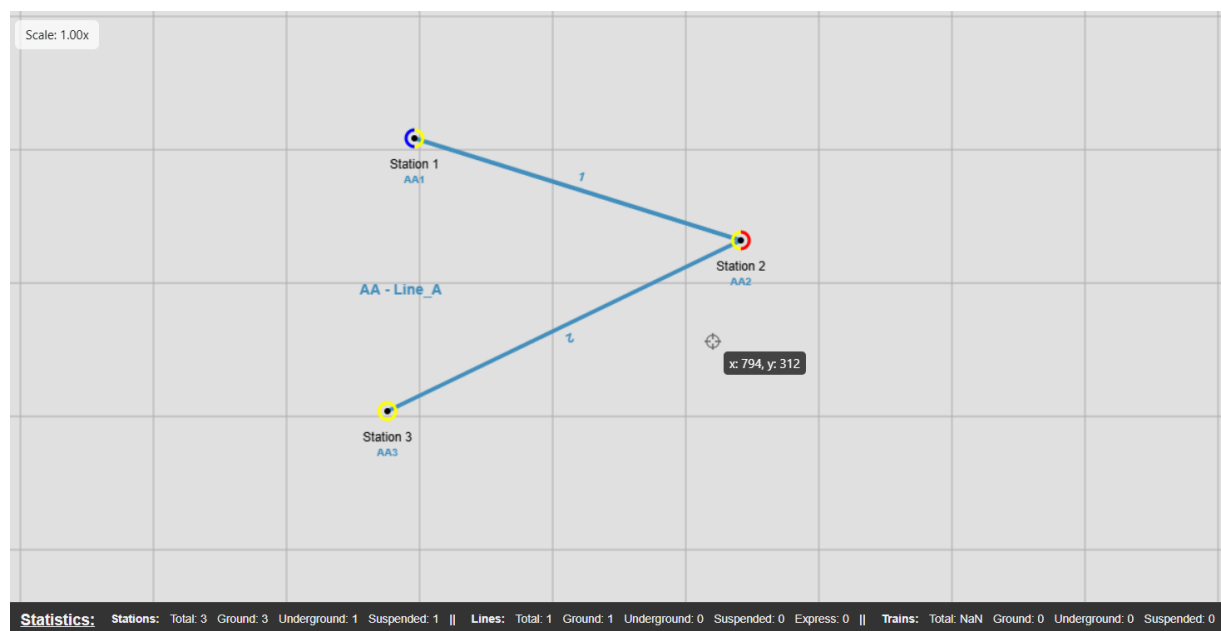
Qty: 4 (Max: 60)

Stations:

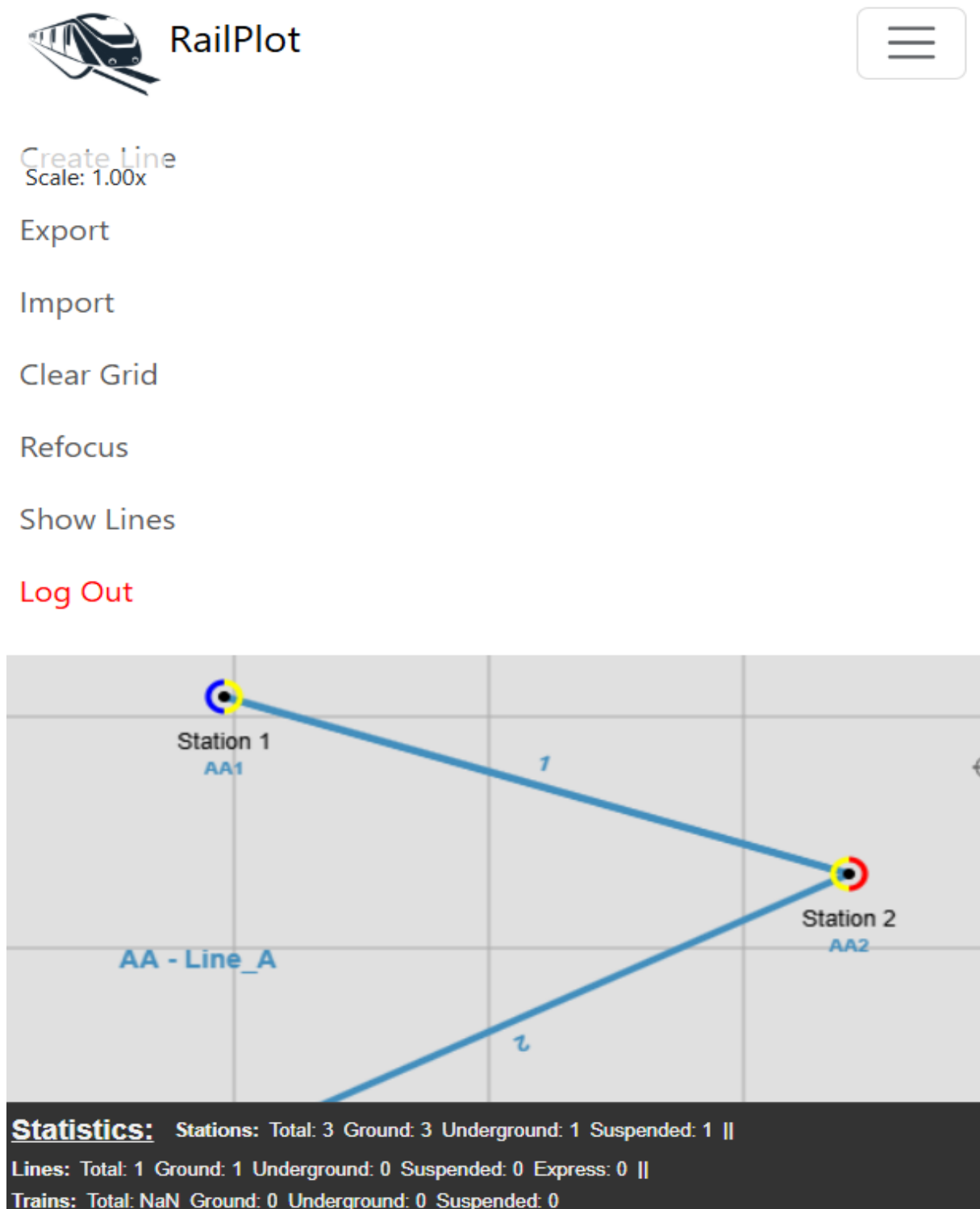
Station 1	↑	↓	×
Station 2	↑	↓	×
Station 3	↑	↓	×
+			

Save Line

- **Statisztikák (statsPanel):** Egy oldalsó vagy alsó panel, amely valós időben frissülő összesített adatokat mutat a létrehozott hálózatról.
 - **Állomások (Stations):** Összesen (Total), Földfelszíni (Ground), Földalatti (Underground), Függesztett (Suspended).
 - **Vonalak (Lines):** Összesen (Total), Földfelszíni (Ground), Földalatti (Underground), Függesztett (Suspended), Expressz (Express).
 - **Vonatok (Trains):** Összesen (Total), és típusonkénti bontásban (Ground, Underground, Suspended).



- **Telefonos/kis képernyős nézet:** A Bootstrap formázásnak köszönhetően ez az oldal telefonon vagy vagy kisebb képernyőjű eszközökön is tökéletesen használható a reszponzív tulajdonságai miatt.



6. Hibajelzések és Problémamegoldás

A program használata során különféle hibaüzenetekkel találkozhat. Ezek célja, hogy segítsenek a problémák azonosításában és javításában.

- **Űrlap validációs hibák:** A leggyakoribb hibatípus. Akkor jelennek meg, ha egy űrlapon (pl. regisztráció, bejelentkezés, állomás/vonal szerkesztése) valamelyik mezőt helytelenül vagy hiányosan tölti ki.
 - *Példák:* "Ez a mező kötelező.", "Érvénytelen e-mail formátum.", "A jelszónak legalább 8 karakter hosszúnak kell lennie.", "A két jelszó nem egyezik.", "A vonalkód maximum 2 karakter lehet."
 - *Megjelenés:* Általában a hibás beviteli mező alatt vagy mellett jelenik meg piros színű szöveg, és maga a mező is piros keretet kaphat.
- **Szerveroldali hibák:** Olyan problémák, amelyek a háttérrendszerben (szerveren) történő adatfeldolgozás során lépnek fel.
 - *Példák:* "Hiba történt a mentés során.", "Az importált fájl formátuma érvénytelen.", "Adatbázis kapcsolati hiba.", "Nincs jogosultsága ehhez a művelethez."
 - *Megjelenés:* Ezek általában egy felugró üzenetben vagy egy külön hibaoldalon jelennek meg. Ritkábban előfordulhatnak általános hibaüzenetek is, mint pl. "500 Internal Server Error".
- **Bejelentkezési hibák:**
 - *Példa:* "Hibás e-mail cím vagy jelszó."
 - *Megjelenés:* Közvetlenül a bejelentkezési űrlapon jelenik meg.
- **Importálási hibák:** Ha a feltölteni kívánt JSON fájl sérült, nem megfelelő formátumú, vagy olyan adatokat tartalmaz, amelyek ütközést okoznának a rendszerben.
 - *Példa:* "Hiba a JSON fájl feldolgozása közben.", "Nem létező állomás ID a fájlban."
 - *Megjelenés:* Felugró üzenet vagy hibaüzenet a felületen.
- **Problémamegoldási tippek:**
 - **Validációs hiba:** Olvassa el figyelmesen a hibaüzenetet, és javítsa a beviteli mező tartalmát az üzenetnek megfelelően.
 - **Általános hiba:** Próbálja meg frissíteni az oldalt (F5 vagy Ctrl+R). Ha a hiba továbbra is fennáll, próbáljon meg kijelentkezni és újra bejelentkezni.

- **Import hiba:** Ellenőrizze az importálni kívánt fájl épségét és formátumát. Próbálkozzon egy korábbi, biztosan működő exportfájlal.
- **Ha semmi sem segít:** Jegyezze fel a hibaüzenetet (akár képernyőképet is készíthet), és vegye fel a kapcsolatot a rendszer üzemeltetőjével/fejlesztőjével (lásd 8. pont).

7. Adminisztrátori Felület (index.blade.php - User & Train Management)

Az adminisztrátori jogosultsággal rendelkező felhasználók számára egy külön felület érhető el, ahol a rendszer alapvető beállításait és adatait kezelhetik. Ennek két fő része van:

- **Elérés és Navigáció:**

- Az admin felület általában egy külön URL-en érhető el, vagy bejelentkezés után egy speciális menüpontból.
- A felhasználó- és vonatkezelő nézetek között az oldalak alján található "Switch to User Management" és "Switch to Train Management" gombokkal lehet váltani.
- Mindkét admin nézetből elérhető a "Log Out" gomb a kijelentkezéshez.

- **Felhasználók Kezelése (User Management Panel):**

- **Áttekintés:** Egy táblázat listázza az összes regisztrált felhasználót.
- **Megjelenített adatok:** Név (Name), E-mail cím (Email), Szerepkör (Role - 'User' vagy 'Admin').
- **Műveletek:**
 - **Törlés:** Minden sor (felhasználó) mellett megjelenik egy törlés gomb (kuka ikon), amellyel az adott felhasználói fiók véglegesen eltávolítható.
 - **Korlátozások:** Az adminisztrátori szerepkörű felhasználókat (Admin) biztonsági okokból nem lehet ezen a felületen törölni. Továbbá, a bejelentkezett adminisztrátor saját magát sem törölheti. A törlés előtt a rendszer mindig megerősítést kér egy felugró ablakban (Are you sure...?).

- **Vonatok Kezelése (Train Management Panel):**

- **Új Vonat Hozzáadása:**
 - Egy különálló szekcióban található űrlap segítségével lehet új vonattípusokat felvinni a rendszerbe.
 - **Űrlap mezői:**
 - **Train Name:** A vonat azonosító neve (pl. "Stadler FLIRT", "BHÉV MXA").
 - **Train Quantity / capacity:** Az adott típusból rendelkezésre álló teljes darabszám (készlet) megadása. Ez korlátozza, hogy összesen hány darabot lehet kiosztani a különböző vonalakra.

- **Select Type:** Legördülő menü a vonat alapvető típusának kiválasztására (Ground, Underground, Suspended), ami meghatározza, milyen típusú vonalakhoz rendelhető hozzá.
 - Az "Add Train" gombra kattintva az új vonat mentésre kerül az adatbázisba.
- **Meglévő Vonatok Kezelése:**
 - Egy táblázat listázza a rendszerben már szereplő összes vonattípust.
 - **Megjelenített adatok:** Név (Name), Rendelkezésre álló darabszám (Quantity/capacity), Típus (Type).
 - **Műveletek (soronként):**
 - **Szerkesztés:** A név, darabszám és típus mezők közvetlenül a táblázatban szerkeszthetők. A beírt módosítások a sor végén található pipa ikonra (Update Train) kattintva menthetők el.
 - **Törlés:** A sor végén található kuka ikonra (Delete Train) kattintva a vonattípus véglegesen eltávolítható a rendszerből. Figyelem: Ez a művelet hatással lehet azokra a vonalakra, amelyekhez ez a típus hozzá volt rendelve! A rendszer törlés előtt megerősítést kér.

8. Elérhetőség, Támogatás

Ha kérdése van az alkalmazás használatával kapcsolatban, hibát észlel, vagy fejlesztési javaslata van, kérjük, vegye fel velünk a kapcsolatot az alábbi email címen:

- **E-mail:** railplot1@gmail.com

Kérjük, hiba bejelentésekor a lehető legrészletesebben írja le a problémát, a lépéseket, amelyek a hiba előidézéséhez vezettek, és ha lehetséges, csatoljon képernyőképet a hibaüzenetről vagy a problémás képernyőrészletről.

RailPlot - Fejlesztői Dokumentáció (Integrált és Bővített Változat)

1. Bevezetés

- **A Projekt Célja:** A RailPlot projekt azzal a céllal jött létre, hogy egy modern, webalapú, nyílt forráskódú (vagy annak szánt) alternatívát kínáljon vasúti és egyéb kötöttpályás hálózatok egyszerűsített vizuális tervezésére.
- **Szükségesség és Egyediség:** Míg léteznek általános rajzolóprogramok és komplexebb közlekedésmodellező szoftverek, a RailPlot specifikusan a vonalak, állomások és azok kapcsolatainak vizuális megjelenítésére és alapvető attribútumokkal (típus, szín, kód) való ellátására koncentrált. Egyediségét a következők adhatják (a konkrét implementációtól függően):
 - **Webalapú és platformfüggetlen:** Bármilyen modern böngészőből elérhető, operációs rendszertől függetlenül.
 - **Interaktív vászon:** HTML5 Canvas technológia használata a dinamikus és reszponzív megjelenítésért.
 - **Egyszerűsített adatmodell:** Könnyen érthető struktúra az állomások, vonalak és vonatok kezelésére.
 - **Import/Export funkció:** Adatok egyszerű megosztása és mentése JSON formátumban.
 - **Beépített statisztikák:** Alapvető hálózati mutatók gyors áttekintése.
 - **Adminisztrációs felület:** Felhasználók és alapvető erőforrások (pl. vonatok) központi kezelése.
- **Technológiai Célkitűzések:** A projekt célja volt a modern, széles körben használt és jól dokumentált technológiák alkalmazása a fenntarthatóság és továbbfejleszthetőség érdekében. Ez magában foglalja a Laravel PHP keretrendszer használatát a szerveroldali logikához, a HTML5 Canvas és JavaScript (valószínűleg egy keretrendszerrel vagy anélkül) használatát a frontend interaktivitáshoz, valamint a Bootstrap keretrendszert a reszponzív felhasználói felület kialakításához.

2. Fejlesztői Környezet

A projekt fejlesztése és karbantartása során az alábbi hardver- és szoftverkönyezet használata javasolt vagy szükséges:

- **Hardver:**

- **Fejlesztői Gép:** Legalább egy közepkategóriás laptop vagy asztali számítógép (pl. Intel Core i5/AMD Ryzen 5 vagy jobb processzor, 8 GB+ RAM, SSD a gyorsabb build és futtatási idő érdekében).
- **Operációs Rendszer:** Windows, macOS vagy Linux. A Laravel fejlesztés mindhárom platformon támogatott (pl. a Laravel Sail Docker-alapú környezettel).
- **Monitor:** Nagyobb felbontású monitor (vagy több monitor) megkönnyíti a kód és a böngésző egyidejű áttekintését.

- **Szoftverek:**

- **Webszerver:** Helyi fejlesztéshez a Laravel beépített szervere (php artisan serve) általában elegendő. Éles környezetben Apache vagy Nginx webszerver szükséges, megfelelően konfigurálva (pl. URL újraírás).
- **PHP:** A projekt által megkövetelt PHP verzió (pl. 8.1 vagy újabb, a composer.json fájl alapján ellenőrizendő). Szükségesek lehetnek bizonyos PHP kiterjesztések (pl. pdo_mysql, mbstring, xml, gd - ha képezelés is van).
- **Adatbázis-kezelő:** MySQL (ajánlott) vagy PostgreSQL. A fejlesztéshez SQLite is használható az egyszerűség kedvéért. Szükség van egy adatbázis-kliensre is (pl. phpMyAdmin, DBeaver, TablePlus).
- **Composer:** A PHP függőségek telepítéséhez és kezeléséhez elengedhetetlen (composer install).
- **Node.js és NPM/Yarn:** A frontend függőségek (pl. Bootstrap JS, esetleges JS keretrendszerek, build eszközök mint a Vite) telepítéséhez és kezeléséhez (npm install vagy yarn install, npm run dev/build).
- **Kódszerkesztő/IDE:** Egy modern kódszerkesztő vagy integrált fejlesztői környezet (IDE), amely támogatja a PHP, JavaScript, HTML, CSS nyelveket és a Laravel keretrendszert (pl. Visual Studio Code PHP Intelephense/Laravel Extension Pack kiegészítőkkal, vagy PHPStorm).
- **Verziókezelő:** Git a kód verziókövetéséhez és a közös munkához (pl. GitHub, GitLab, Bitbucket használatával).

- **Böngésző Fejlesztői Eszközök:** Chrome DevTools, Firefox Developer Tools nélkülözhetetlenek a frontend debuggolásához, hálózati kérések elemzéséhez, teljesítményprofilozáshoz.

3. Technológiai Választások Indoklása

A projekt technológiai alapjainak kiválasztásakor több szempontot vettünk figyelembe:

- **Programnyelv (PHP) és Keretrendszer (Laravel):**

- **PHP:** Az egyik legelterjedtebb szerveroldali szkriptnyelv, különösen weben. Hatalmas közössége van, rengeteg dokumentáció, könyvtár és eszköz érhető el hozzá. Könnyen telepíthető és futtatható a legtöbb tárhelyszolgáltatónál.
- **Laravel:** Egy elegáns szintaxisú, fejlesztőbarát PHP keretrendszer, amely követi a MVC (Model-View-Controller) tervezési mintát és számos beépített eszközt kínál, amelyek jelentősen gyorsítják a fejlesztést:
 - *Eloquent ORM:* Intuitív, Active Record implementáció az adatbázis-műveletekhez. Leegyszerűsíti a CRUD (Create, Read, Update, Delete) műveleteket és a táblák közötti kapcsolatok kezelését.
 - *Blade Sablonmotor:* Egyszerű, mégis erőteljes sablonrendszer, amely lehetővé teszi a tiszta PHP kód és HTML elválasztását, öröklődést, komponenseket biztosít.
 - *Routing:* Könnyen definiálható útvonalak a HTTP kérések Controller metódusokhoz való irányítására.
 - *Middleware:* Lehetőség HTTP kérések szűrésére és módosítására (pl. autentikáció, CSRF védelem).
 - *Artisan Console:* Parancssori eszköz adatbázis-migrációk futtatásához, kódgeneráláshoz, feladatütemezéshez stb..
 - *Beépített Autentikáció/Autorizáció:* Gyorsan implementálható bejelentkezési, regisztrációs, jelszó-visszaállítási funkciók és jogosultságkezelés (pl. a role mező alapján).
 - *Biztonság:* Beépített védelem gyakori webes sebezhetőségek ellen (pl. CSRF, XSS, SQL injection - Eloquent használatával).

- **Frontend Technológiák:**

- **HTML5 Canvas:** A main.blade.php-ban található <canvas> elem ideális választás a dinamikus, 2D grafikai megjelenítéshez, mint amilyen a vasúti térkép kirajzolása, nagyítása és mozgatása. Hardveresen gyorsított lehet, ami komplexebb hálózatok esetén is jó teljesítményt biztosít.
- **JavaScript:** Elengedhetetlen a kliensoldali logika megvalósításához: a Canvas API vezérlése (rajzolás, eseménykezelés - kattintás, mozgatás, nagyítás), a









felhasználói felület elemeinek (modális ablakok, statisztikák) dinamikus frissítése, AJAX kérések küldése a Laravel backend felé az adatok lekérdezéséhez és mentéséhez (a main.js fájl felelős ezért, ami nem áll rendelkezésre). Valószínűleg Vanilla JS vagy egy kisebb könyvtár/keretrendszer (pl. Alpine.js, Vue.js, vagy akár jQuery) használatos.






- **Bootstrap:** Egy népszerű, reszponzív CSS és JavaScript keretrendszer. Előre elkészített komponenseket (navigációs sáv, gombok, kártyák, modális ablakok, táblázatok, űrlap elemek, offcanvas menü) és egy grid rendszert biztosít, ami jelentősen megkönnyíti és gyorsítja az esztétikus, mobilbarát felhasználói felület kialakítását.
- **Adatbázis (MySQL - Feltételezett alapértelmezett):**
 - A MySQL széles körben használt, megbízható, jól skálázható relációs adatbázis-kezelő rendszer. A legtöbb tárhelyszolgáltató támogatja, és a Laravel is kiválóan integrálódik vele. Nagy közössége és rengeteg dokumentációja van. Alternatívaként a PostgreSQL is jó választás lehet, ha specifikusabb adatbázis-funkciókra van szükség.












4. Adatszerkezet (Migrációk és Feltételezések Alapján)









A rendszer adatbázis-struktúráját a Laravel migrációk definiálják. Az elérhető migrációs fájlok és a felhasználói felület (main.blade.php, index.blade.php) alapján a következő táblák és kapcsolatok azonosíthatók vagy feltételezhetők:

- **users tábla:** Felhasználói fiókok kezelése.
 - id (PK): Egyedi azonosító.
 - name: Felhasználó neve.
 - email (Unique): Felhasználó e-mail címe, bejelentkezéshez használatos.
 - email_verified_at: E-mail cím megerősítésének időpontja.
 - password: Hash-elt jelszó (fontos a biztonságos hash algoritmus használata, pl. Bcrypt - Laravel alapértelmezett).
 - remember_token: "Emlékezz rám" funkcióhoz.
 - role (Default: 'user'): Jogosultsági szint (pl. 'user', 'admin').
 - created_at, updated_at: Időbélyegek (Laravel Eloquent automatikusan kezeli).
- **password_reset_tokens tábla:** Elfelejtett jelszó funkcióhoz.
 - email (PK): Kapcsolódó felhasználó e-mail címe.
 - token: Biztonságos, véletlenszerű token.
 - created_at: Token lejáratí idejének kezeléséhez.
- **sessions tábla:** Aktív felhasználói munkamenetek tárolása.
 - id (PK): Session azonosító.
 - user_id (FK): Bejelentkezett felhasználó ID-ja.
 - ip_address, user_agent: Biztonsági és nyomkövetési információk.
 - payload: A session változók (pl. bevásárlókosár, felhasználói beállítások) tárolása.
 - last_activity: Munkamenet lejáratí idejének kezeléséhez.
- **trains tábla:** A rendszerben elérhető vonattípusok definíciója.
 - id (PK): Egyedi azonosító.
 - name: Vonattípus neve.
 - capacity: Az adott típusból a rendszerben elérhető teljes darabszám (készlet).
 - type: Vonat típusa ('ground', 'underground', 'suspended'), meghatározza a kompatibilitást a vonalakkal.
 - created_at, updated_at: Időbélyegek.
- **Kapcsolati Ábra:**

 	rail_db trains
	id : bigint(20) unsigned
	created_at : timestamp
	updated_at : timestamp
	name : varchar(255)
	capacity : int(11)
	type : varchar(255)

 	rail_db password_reset_tokens
	email : varchar(255)
	token : varchar(255)
	created_at : timestamp

 	rail_db users
	id : bigint(20) unsigned
	name : varchar(255)
	email : varchar(255)
	email_verified_at : timestamp
	password : varchar(255)
	remember_token : varchar(100)
	created_at : timestamp
	updated_at : timestamp
	role : varchar(255)

 	rail_db sessions
	id : varchar(255)
	user_id : bigint(20) unsigned
	ip_address : varchar(45)
	user_agent : text
	payload : longtext
	last_activity : int(11)

5. Tipikus Algoritmusok és Adatfeldolgozási Folyamatok

A rendszer működése során számos algoritmus és folyamat fut le a háttérben és a felhasználói felületen. Néhány kulcsfontosságú példa:

- **Vászon Rajzoló Ciklus (main.js - feltételezett):**

1. **Adatlekérés:** Az oldal betöltésekor vagy frissítéskor AJAX kéréssel lekéri a szerverről az aktuális felhasználóhoz tartozó összes állomás, vonal, és azok kapcsolatainak adatait (pl. egy JSON objektumban).
2. **Állapotkezelés:** Nyilvántartja a vászon aktuális állapotát: eltolás (pan offset X, Y), nagyítási szint (zoom factor).
3. **Megjelenítés (Renderelés):** Egy rajzoló ciklus (pl. requestAnimationFrame használatával optimalizálva) felelős a vászon tartalmának kirajzolásáért:
 - **Tisztítás:** A vászon teljes tartalmának törlése minden rajzoló ciklus előtt (clearRect).
 - **Transzformáció:** A vászon kontextusának eltolása és skálázása az aktuális pan/zoom állapotnak megfelelően (translate, scale). Ez biztosítja, hogy a koordináták helyesen jelenjenek meg a nagyított/eltolt nézetben.
 - **Rács rajzolása:** Halvány segédrács rajzolása a tájékozódáshoz.
 - **Állomások rajzolása:** Végigiterál az állomás adatokon. Minden állomást a transzformált x_coord, y_coord alapján rajzol ki, figyelembe véve a típusát (type_ground, type_underground, type_suspended) – eltérő alakzatok, színek vagy ikonok használatával.
 - **Vonalak rajzolása:** Végigiterál a vonalakon. Minden vonalhoz:
 - Lekéri a hozzá tartozó állomásokat a line_station kapcsolat és az order alapján sorrendbe rendezve.
 - Az állomások transzformált koordinátáit összeköti a vonal color színével és a type típusának megfelelő stílussal (pl. folytonos, szaggatott, vastagabb/vékonyabb vonal). Lehetnek egyszerű egyenes szakaszok (lineTo) vagy komplexebb görbék (bezierCurveTo, quadraticCurveTo) a szebb vonalvezetésért.
 - **Feliratok és Kiemelések:** Állomásnevek, vonalkódok kiírása az elemek mellé (figyelve a nagyításra, hogy olvasható maradjon). Az egér alatt lévő vagy kiválasztott elemek vizuális kiemelése.

4. **Eseménykezelés:** Figyeli az egér (kattintás, mozgatás, görgő) és billentyűzet eseményeket a vásznon a pan/zoom, elem kiválasztás, új elem létrehozás stb. funkciókhoz.
- **Új Vonal Mentése (Laravel Controller/Service):**
 1. **Kérés Fogadása:** Fogadja a POST kérést az addLineModal űrlap adataival (JSON vagy form data formátumban). A kérés tartalmazza a vonal nevét, kódját, színét, típusát, a hozzáadni kívánt állomások ID-ít és sorrendjét, valamint a hozzárendelt vonat ID-ját és darabszámát.
 2. **Validáció (Laravel Request Validation):** Szigorú validációs szabályok alkalmazása a beérkezett adatokra (pl. name kötelező, code max 2 karakter, color érvényes hex kód, type létező típus, station_ids tömb létező állomás ID-kkal, train_id létező vonat ID, quantity pozitív egész szám és nem lépi túl a készletet).
 3. **Autorizáció:** Ellenőrzi, hogy a bejelentkezett felhasználónak van-e joga új vonalat létrehozni.
 4. **Adatbázis Műveletek (Tranzakcióban):**
 - DB::beginTransaction();
 - Új Line rekord létrehozása és mentése az adatbázis lines táblájába.
 - Végigiterál a kapott állomás ID-kon és sorrendeken, és létrehozza a megfelelő rekordokat a line_station pivot táblában (attach vagy sync Eloquent metódusokkal).
 - Létrehozza a rekordot a line_train pivot táblában a vonat hozzárendeléséhez. (Itt ellenőrizni kell, hogy a kért quantity nem lépi-e túl a trains.capacity-t, figyelembe véve az esetlegesen más vonalakhoz már hozzárendelt mennyiségeket is!).
 - DB::commit();
 5. **Válasz Küldése:** Sikeres mentés esetén JSON választ küld a kliensnek (pl. a létrehozott vonal adataival, hogy a frontend frissíthesse a nézetet). Hiba (pl. validációs hiba, adatbázis hiba) esetén a tranzakció visszavonásra kerül (DB::rollBack();), és hibaüzenetet tartalmazó JSON válasz kerül visszaküldésre megfelelő HTTP státuszköddal (pl. 422 Unprocessable Entity, 500 Internal Server Error).
 - **Hálózat Exportálása (Laravel Controller/Service):**

1. **Adatlekérdezés:** Lekérdezi az adatbázisból az aktuális felhasználóhoz tartozó összes stations, lines adatot, valamint a kapcsolataikat (line_station, line_train) az Eloquent relációk (with([...])) segítségével, hogy minimalizálja az adatbázis-lekérdezések számát (N+1 probléma elkerülése).
 2. **Adatstrukturálás:** Az Eloquent modellekből/kollekciókból egy tiszta PHP tömböt vagy objektumot hoz létre, amely tartalmazza a hálózat összes releváns adatát egy könnyen feldolgozható struktúrában (pl. állomások listája, vonalak listája, ahol minden vonal tartalmazza az állomásainak ID-it sorrendben és a hozzárendelt vonatok adatait).
 3. **JSON Konverzió:** A PHP tömböt JSON formátumú sztringgé alakítja (json_encode).
 4. **HTTP Válasz:** Egy Response objektumot hoz létre a JSON sztringgel. Beállítja a Content-Type fejléct application/json-re és a Content-Disposition fejléct attachment; filename="railplot_export.json"-ra, hogy a böngésző fájlletöltést kezdeményezzen.
- **Vonat Törlése (Admin Controller - index.blade.php Train Managementhez):**
 1. **Kérés Fogadása:** Fogadja a DELETE HTTP kérést a törlendő vonat ID-jával (pl. /admin/trains/{train_id}).
 2. **Authentikáció és Autorizáció:** Middleware segítségével ellenőrzi, hogy a kérést küldő felhasználó be van-e jelentkezve ÉS rendelkezik-e 'admin' jogosultsággal.
 3. **Adatmodell Betöltése:** Megkeresi a Train modellt az adatbázisban a kapott ID alapján (Train::findOrFail(\$train_id)). Ha nem található, 404-es hibát dob.
 4. **Kapcsolatok Kezelése (Fontos!):** Mielőtt a vonatot törölné, meg kell határozni, mi történjen azokkal a line_train bejegyzésekkel, amelyek erre a vonatra hivatkoznak. Lehetőségek:
 - Kaszkádolt törlés (ha az adatbázis séma így van beállítva): A vonat törlése automatikusan törli a kapcsolódó line_train rekordokat is.
 - Hiba dobása: Megakadályozza a vonat törlését, ha még hozzá van rendelve vonalakhoz, és üzenetet küld az adminnak, hogy először távolítsa el a hozzárendeléseket. (Ez általában biztonságosabb.)
 - Hozzárendelések nullázása (nem alkalmazható itt, mivel a train_id valószínűleg kötelező a line_train táblában).
 5. **Törlés:** A vonat törlése az adatbázisból (\$train->delete()).

6. **Válasz/Átirányítás:** Sikeres törlés után visszairányítja az admint a vonatkezelő listára (`redirect()->route('admin.trains.index')`) egy sikerüzenettel (`with('success', 'Vonat sikeresen törölve.')`). Hiba esetén (pl. jogosultsági probléma, vagy ha a törlés le van tiltva a kapcsolatok miatt) visszairányítás hibaüzenettel.

6. Tesztelés

A szoftver minőségének biztosítása érdekében átfogó tesztelési stratégia alkalmazása szükséges. Mivel a kódbázis nem tartalmaz dedikált tesztfájlokat, itt egy általános áttekintés a javasolt tesztelési szintekről és típusokról, amelyeket a RailPlot esetében alkalmazni lehetne:

- **Egységtesztek (Unit Tests):**

- **Cél:** Kisebb, izolált kódrészletek (osztályok, metódusok) helyes működésének ellenőrzése.
- **Eszköz:** PHPUnit (Laravel beépített támogatással).
- **Példák:**
 - Egyedi validációs szabályok tesztelése (pl. LineCode szabály).
 - Segédfüggvények, Service osztályok logikájának tesztelése (pl. egy koordináta-transzformációs függvény, export adatelőkészítő logika).
 - Eloquent modellek scope-jainak, metódusainak tesztelése (adatbázis interakciók mockolásával vagy in-memory adatbázis használatával).
 - Custom Casts vagy Attribute Mutator/Accessorok tesztelése.

- **Integrációs Tesztek (Feature Tests):**

- **Cél:** Több komponens együttműködésének tesztelése, jellemzően egy teljes HTTP kérés-válasz ciklus szimulálásával.
- **Eszköz:** PHPUnit Laravel kiterjesztéseivel (HTTP tesztelési segédfüggvények, adatbázis interakciók).
- **Példák:**
 - Felhasználói regisztráció végpont tesztelése: POST kérés küldése valid és invalid adatokkal, adatbázis állapotának és a válasznak (átirányítás, session üzenetek) ellenőrzése.
 - Bejelentkezési végpont tesztelése: Helyes/hibás kredenciákkal, autentikációs állapot ellenőrzése.
 - Vonal létrehozása API végponton keresztül: Kérés küldése, adatbázisban a lines, line_station, line_train táblák tartalmának ellenőrzése.
 - Export funkció tesztelése: GET kérés az export végpontra, válasz státuszkódjának, fejlécének és a JSON tartalom struktúrájának ellenőrzése.

- Adminisztrátori műveletek (felhasználó törlése, vonat létrehozása/módosítása/törlése) tesztelése megfelelő autentikációval és autorizációval.
- **Felhasználói Felület (UI/End-to-End) Tesztek:**
 - **Cél:** A teljes alkalmazás működésének tesztelése egy valós böngészőn keresztül, a felhasználói interakciók szimulálásával.
 - **Eszköz:** Laravel Dusk (böngészőautomatizálási eszköz). Alternatívák: Selenium, Cypress, Playwright.
 - **Példák:**
 - Regisztrációs űrlap kitöltése, beküldése, hibaüzenetek ellenőrzése, sikeres regisztráció utáni átirányítás ellenőrzése.
 - Bejelentkezés, majd navigálás a fő tervezőfelületre.
 - Vázon interakciók tesztelése: Új állomás létrehozása kattintással, állomás adatainak módosítása a modális ablakban, vonal létrehozása több állomással, szín beállítása, mentés.
 - Export/Import gombok kattintása, fájl feltöltés/letöltés szimulálása (amennyire az eszköz engedi).
 - Statisztikai panel adatainak ellenőrzése egy új elem hozzáadása után.
 - Admin felületen egy felhasználó vagy vonat törlése a gombra kattintva, a megerősítő ablak elfogadása után, majd a lista frissülésének ellenőrzése.
- **Hibakezelési Tesztesetek:** Kifejezetten a váratlan helyzetek és hibák kezelésének ellenőrzése.
 - Érvénytelen adatok küldése az API végpontokra (pl. nem létező ID, formátumnak nem megfelelő érték).
 - Jogosulatlan hozzáférési kísérletek (pl. sima felhasználó próbál admin végpontot elérni).
 - Direkt URL manipuláció (pl. nem létező erőforrás ID megadása).
 - Szerveroldali hibák szimulálása (pl. adatbázis kapcsolat megszakítása) és a felhasználói felület reakciójának ellenőrzése (pl. barátságos hibaüzenet jelenik meg?).
 - Nagy méretű vagy hibás formátumú fájl importálása.
- **Teljesítmény és Terheléses Tesztelés (Opcionális):**

- **Cél:** Az alkalmazás válaszképességének és stabilitásának mérése nagy számú adat vagy egyidejű felhasználó esetén.
- **Eszközök:** ApacheBench (ab), JMeter, K6.
- **Példák:** Sok állomást és vonalat tartalmazó hálózat rajzolási sebességének mérése, API végpontok válaszidejének mérése növekvő terhelés mellett.

7. Tesztelői Hozzáférés Biztosítása

A tesztelés megkönnyítése és a különböző felhasználói szerepkörök ellenőrzése érdekében általában dedikált tesztfiókokat hozunk létre:

- **Normál Felhasználói Fiók:**

- **Cél:** A standard felhasználói funkciók (regisztráció, bejelentkezés, hálózat tervezés, export/import) tesztelése.
- **Létrehozás:** Regisztráció a normál felületen, vagy manuális létrehozás adatbázis seedeléssel.
- **Példa Kredenciálok:** E-mail: user@railplot.test, Jelszó: Password123 (Ezeket biztonságos módon kell kezelni és kommunikálni a tesztelőkkel).

- **Adminisztrátori Fiók:**

- **Cél:** Az adminisztrátori felület funkcióinak (felhasználókezelés, vonatkezelés) tesztelése.
- **Létrehozás:** Manuális létrehozás az adatbázisban vagy egy dedikált seeder futtatásával, a role mezőt 'admin'-ra állítva.
- **Példa Kredenciálok:** E-mail: admin@railplot.test, Jelszó: AdminPass123 (Ezeket különösen körültekintően kell kezelni).

- **Hozzáférés:** A tesztelőknek meg kell adni az alkalmazás tesztkörnyezetének URL-jét és a megfelelő tesztfiókhoz tartozó bejelentkezési adatokat. Fontos, hogy a tesztkörnyezet adatbázisa elkülönüljön az éles környezettől, hogy a tesztelés ne befolyásolja a valós felhasználói adatokat.

id	name	email	email_verified_at	password	remember_token	created_at	updated_at	role
5	Admin	admin@admin.com	NULL	\$2y\$12\$g2wXanOKphRW.cdgClqRguyUyW3SG5LoZz5VcEtBkiG...	NULL	2025-04-28 09:49:53	2025-04-28 09:49:53	admin
6	User	user@user.com	NULL	\$2y\$12\$sDjQYTESMBg7QYrrV3TLWO5fvLsD5/MfbaAR8Emd7MQ...	NULL	2025-04-28 09:51:21	2025-04-28 09:51:21	user

8. Fejlesztési Lehetőségek

A RailPlot jelenlegi funkcionalitása szilárd alapot képez, de számos irányba továbbfejleszthető a felhasználói élmény és a funkcionalitás bővítése érdekében:

- **Fejlettebb Vizuális Szerkesztés:**

- **Görbe vonalak:** Lehetőség íves pályaszakaszok rajzolására (pl. Bézier-görbékkel) a realisztikusabb megjelenítésért.
- **Váltók és Kereszteződések:** Dedikált objektumok váltók, kereszteződések, jelzők elhelyezésére a vásznon.
- **Rétegek:** Különböző típusú elemek (pl. állomások, vonalak, háttértérkép, feliratok) külön rétegeken való kezelése a jobb átláthatóság és szerkeszthetőség érdekében.
- **Háttértérkép/Térkép Integráció:** Lehetőség saját kép (pl. terepasztal tervrajza) vagy online térképszolgáltatás (pl. OpenStreetMap) háttérként való használatára, amelyhez a hálózatot igazítani lehet.

- **Funkcionális Bővítések:**

- **Valós idejű Együttműködés:** Több felhasználó egyidejűleg szerkeszthesse ugyanazt a hálózatot, a változások azonnali megjelenítésével (WebSockets használata, pl. Laravel Echo + Pusher/Soketi).
- **Részletesebb Vonat/Jármű Kezelés:** Vonatokhoz menetrend, sebesség, kapacitás (utas/teher) hozzárendelése. Egyszerű szimuláció futtatása a vonatok mozgásáról a hálózaton.
- **Útvonaltervezés:** Két állomás közötti legrövidebb/leggyorsabb útvonal megkeresése a hálózaton (gráfalgoritmusok, pl. Dijkstra vagy A* implementálása).
- **Költségvetés és Anyaglista:** Az elemekhez (pálya hossza, állomások típusa, váltók száma) költségek rendelése és automatikus anyaglista/költségbecslés generálása.
- **Több Hálózat Kezelése:** Lehetőség több, különálló hálózati projekt létrehozására és kezelésére egy felhasználói fiókon belül.

- **Integráció és API:**

- **API Réteg:** RESTful API biztosítása a hálózati adatok programozott eléréséhez és módosításához, lehetővé téve más alkalmazásokkal való integrációt.

- **Formátumok Támogatása:** Más elterjedt formátumok (pl. GeoJSON, GPX) importálásának és exportálásának támogatása.
- **Felhasználói Élmény és Adminisztráció:**
 - **Verziókövetés/Visszavonás:** Szerkesztési előzmények mentése és a visszavonás/ismétlés (Undo/Redo) funkció implementálása.
 - **Sablonok és Előre Definiált Elemek:** Gyakran használt állomástípusok, pályaelemek sablonként való mentése és újrafelhasználása.
 - **Fejlettebb Jogosultságkezelés:** Részletesebb jogosultsági szintek bevezetése (pl. csak olvasás, szerkesztés, megosztás kezelése).

9. Összefoglalás

- **Elkészült munka értékelése:** A RailPlot projekt sikeresen teljesítette a kezdeti célkitűzéseket, létrehozva egy működőképes, webalapú eszközt vasúti hálózatok vizuális tervezésére. Az alkalmazás a Laravel keretrendszer stabil alapjaira épül, modern frontend technológiákat használ a felhasználói felülethez, és biztosítja az alapvető funkciókat, mint az állomások és vonalak kezelése, import/export, valamint adminisztrációs lehetőségek. A fejlesztés során kihívást jelentett a Canvas alapú rajzolófelület teljesítményének optimalizálása és a különböző böngészők közötti kompatibilitás biztosítása. Az adatszerkezet a migrációknak köszönhetően jól definiált, de a jövőbeli bővítésekhez további átgondolást igényelhet. Összességében a projekt jó alapot teremtett a további fejlesztésekhez.
- **A program hasznosulása a továbbiakban:** Úgy gondolom, hogy a RailPlot a jelenlegi formájában is hasznos lehet diákok és hobbi felhasználók számára. A 'Fejlesztési Lehetőségek' pontban vázolt funkciók implementálásával az alkalmazás vonzóbbá válhatna szélesebb körben, akár félprofesszionális felhasználók számára is. Potenciálisan beilleszthető lenne oktatási anyagokba szemléltető eszközként, vagy közösségi projektek tervezési fázisában nyújthatna segítséget. A nyílt forráskódúvá tétele (ha ez cél) tovább növelhetné az elterjedését és a közösségi fejlesztés lehetőségét.
- **A munkamegosztás a fejlesztés során:** Megfelelő kommunikációval és koordinációval sikeresen tudtuk egymás között felosztani a feladatokat és szükség esetén segíteni egymásnak. Mindkettőnknek része volt a frontend és backend fejlesztésében is.

10. Irodalomjegyzék

A fejlesztés és tesztelés során felhasznált források, referenciák és segédletek:

Weboldalak/Dokumentációk:

- Laravel dokumentáció
- Bootstrap dokumentáció
- Canvas dokumentáció