LazarExpressz – Fejlesztői dokumentáció

Tartalomjegyzék

1. Projekt áttekintése

A LazarExpressz egy webes alkalmazás, amely … (ide kerül a projekt rövid leírása).

2. Követelmények

- PHP
- MySQL/MariaDB
- Webszerver (pl. XAMPP)
- Böngésző

3. Projekt szerkezete

```
assets/
  css/         # Stíluslapok
  docs/        # Dokumentációk
  js/          # JavaScript fájlok
```

```
  php/          # Backend PHP szkriptek
  sql/          # Adatbázis sémák
index.php       # Fő belépési pont
```

4. Telepítés és futtatás

1. Klónozd a repót vagy másold a fájlokat a webszervered gyökerébe.
2. Importáld az `assets/sql/db.sql` fájlt az adatbázisodba.
3. Állítsd be az adatbázis elérhetőségeit az `assets/php/db.php` fájlban.
4. Indítsd el a webszervert, majd nyisd meg a böngészőben az `index.php`-t.

5. Főbb funkciók

- Adatok mentése és lekérdezése
- Proxy funkciók
- ...

6. Adatbázis séma

Az adatbázis szerkezete az `assets/sql/db.sql` fájlban található.

7. Fejlesztői tippek

- A PHP szkriptek az `assets/php/` mappában találhatók.
- A front-endhez szükséges JS és CSS fájlok az `assets/js/` és `assets/css/` mappákban vannak.

8. index.php – Fájl leírása

Az `index.php` a projekt fő belépési pontja, amely a felhasználói felületet jeleníti meg. Az oldal HTML5 szabvány szerint épül fel, és tartalmazza a szükséges metaadatokat, stíluslapokat, valamint JavaScript könyvtárakat.

Főbb elemek:

- **Fejléc** (`<head>`): Itt találhatók a metaadatok (karakterkódolás, leírás, kulcsszavak, szerző, reszponzivitás), valamint a stíluslapok betöltése:
  - `leaflet.css` : A Leaflet térképkönyvtár stíluslapja CDN-ről.
  - `assets/css/styles.css` : Saját stíluslap.
- **Törzs** (`<body>`): Egyetlen fő elemet tartalmaz:
  - `#map` div: Ebben jelenik meg a térkép.

- Scriptek:
  - `leaflet.js` : A Leaflet JavaScript könyvtár CDN-ről, amely interaktív térképek megjelenítését teszi lehetővé.
  - `assets/js/script.js` : Saját JavaScript fájl, amely a térkép működését és egyéb front-end logikát valósítja meg.

Fő funkciók:

- A felhasználó egy interaktív térképet lát, amelyet a Leaflet könyvtár kezel.
- A további funkciók (pl. vonatok megjelenítése, adatok betöltése) a `script.js` -ben valósulnak meg.

Megjegyzések:

- Az oldal reszponzív, mobil eszközökön is használható.
- A szerveroldali logika nem az `index.php` -ben, hanem az `assets/php/` mappában található PHP szkriptekben van.

9. assets/php/db.php – Fájl leírása

Ez a fájl felelős az adatbázis-kapcsolat létrehozásáért a MySQL szerverrel. A `$conn` változóban egy `mysqli` objektum jön létre, amelyet a többi PHP szkript használhat az adatbázis-műveletekhez.

Fő funkciók:

- Létrehozza a kapcsolatot a `localhost` szerveren futó `lazarexpress` nevű adatbázissal.
- Hibakezelés: ha a kapcsolat sikertelen, a szkript leáll és hibaüzenetet ír ki.

Megjegyzések:

- A felhasználónév és jelszó jelenleg alapértelmezett (`root` , jelszó nélkül), ezt éles környezetben módosítani szükséges.
- A kapcsolatot a többi szkript a `require_once 'db.php';` paranccsal tudja használni.

10. assets/php/proxy.php – Fájl leírása

Ez a szkript proxyként működik: továbbítja a beérkező JSON kéréseket a MÁV GraphQL API felé, majd a választ visszaküldi a kliensnek. Emellett, ha a válasz tartalmaz járműpozíciókat, azokat elmenti egy helyi adatbázisba.

Fő funkciók:

- Bejövő JSON kérés továbbítása a MÁV API felé cURL-lel.
- A válasz HTTP státuszkódjának és tartalmának visszaküldése a kliensnek.
- Ha a válaszban van `vehiclePositions` adat, azokat elmenti a `positions` táblába (PDO-val, külön adatbázis-kapcsolattal).
- Hibakezelés: ha nincs input, vagy adatbázis-hiba lép fel, megfelelő hibaüzenetet ad vissza vagy naplóz.

Megjegyzések:

- A proxy lehetővé teszi a front-end számára, hogy közvetlenül ne a MÁV API-t hívja, hanem ezen keresztül, így elrejthetők a kulcsok, illetve kezelhető a CORS.
- Az adatbázis-kapcsolat paramétereit (felhasználó, jelszó, adatbázisnév) szükséges testre szabni.
- A mentéshez PDO-t használ, míg máshol mysqli-t – egységesítés javasolt.

11. assets/php/save_train_data.php – Fájl leírása

Ez a szkript POST kéréseket fogad, és a kapott vonatadatokat elmenti a `trains` táblába. A bemeneti adatokat JSON formátumban várja.

Fő funkciók:

- Csak POST kéréseket engedélyez.
- Betölti az adatbázis-kapcsolatot a `db.php`-n keresztül.
- Ellenőrzi a kötelező mezők meglétét a JSON-ban.
- Előkészített SQL-lel (`prepare` és `bind_param`) menti az adatokat a `trains` táblába.
- Hibakezelés: hiányzó mezők, hibás JSON, vagy adatbázis-hiba esetén megfelelő választ ad vissza.

Megjegyzések:

- A bemeneti mezők neveit és típusait a front-endnek követnie kell.
- A szkript végén bezárja a statement-et és az adatbázis-kapcsolatot.
- A hibák magyar nyelvű üzenetekkel térnek vissza, ami segíti a fejlesztést, de éles környezetben érdemes lehet angolra váltani.

12. assets/js/script.js – Fájl leírása

Ez a JavaScript fájl felelős a LazarExpressz alkalmazás front-end térképes megjelenítéséért, a vonatok adatainak lekérdezéséért, vizualizációjáért, valamint a szerveroldali adatmentés indításáért. A fájl a Leaflet könyvtárat használja interaktív térkép megjelenítésére, és AJAX hívásokkal kommunikál a backenddel.

Főbb funkciók és logikai egységek

1. Térkép inicializálása

- A `map` változó egy Leaflet térképet hoz létre, amely Magyarország középpontjára (47.0, 19.0) van pozícionálva, 7-es zoom szinttel.
- Az OpenStreetMap csempéit használja alapértelmezett térképrétegként.

2. Vonat markerek kezelése

- A `vehicleMarkers` egy Map objektum, amely a térképen megjelenített vonatok markerjeit tárolja, kulcsként a jármű azonosítójával.
- A markerek színe a késés mértékétől függ (`getColorByDelay`), így vizuálisan is azonnal látható a vonat aktuális helyzete.
- A marker egyedi SVG ikonnal jelenik meg, amely tartalmaz egy irányt mutató nyilat (a vonat haladási iránya szerint).

3. Adatlekérés a backendről

- A `fetchGraphQL` függvény POST kérést küld a `proxy.php` -nak, amely továbbítja azt a MÁV GraphQL API felé.
- A `fetchAndDisplayTrains` lekéri az összes, Magyarország területén található vonat pozícióját, majd minden járműhöz külön lekéri a részletes menetrendi adatokat (`fetchTripDelay`).
- A lekérdezések GraphQL formátumban történnek, így rugalmasan bővíthetőek.

4. Menetrendi adatok feldolgozása

- A `fetchTripDelay` lekéri az adott vonat teljes útvonalát, menetrendi és valós idejű érkezési/indulási időkkel.
- A következő megállóhoz tartozó késést, megálló nevét, tervezett és tényleges érkezési időt is meghatározza.
- Ezeket az adatokat átadja a marker megjelenítéséért felelős függvénynek.

5. Marker és popup megjelenítés

- A `displayMarker` függvény felelős a marker létrehozásáért, frissítéséért, valamint a hozzá tartozó tooltip és popup tartalomért.
- A popupban egy táblázat jelenik meg, amely a vonat teljes útvonalát, minden megálló tervezett és tényleges érkezési idejét, valamint a késést mutatja.
- A marker popupjának megnyitásakor egy piros vonal (polyline) jelenik meg a vonat útvonalán.

6. Adatmentés a szerverre

- A `saveTrainData` függvény minden marker frissítésekor POST kérést küld a `save_train_data.php`-nak, amely elmenti az aktuális vonatadatokat az adatbázisba.
- Ez lehetővé teszi a vonatok mozgásának későbbi elemzését, statisztikák készítését.

7. Automatikus frissítés

- A vonatok adatai percenként automatikusan frissülnek (`setInterval(fetchAndDisplayTrains, 60000)`), így a térkép mindig naprakész.

Felhasznált könyvtárak

- **Leaflet**: Interaktív térképek megjelenítésére, markerkezelésre, polylinera.
- **OpenStreetMap**: Térképcsempék forrása.
- **Fetch API**: AJAX kommunikáció a backenddel (proxy.php, save_train_data.php).

Fejlesztői megjegyzések, bővítési lehetőségek

- A marker színezés és ikon testreszabható, akár vonattípus vagy egyéb paraméterek alapján is.
- A popup tartalma bővíthető további adatokkal (pl. vonat kategória, szerelvény hossza, stb.).
- A jelenlegi implementáció minden marker frissítésekor adatot ment a szerverre – nagy forgalom esetén érdemes lehet optimalizálni (pl. csak változás esetén menteni).
- A hibakezelés a frontenden minimális, érdemes lehet bővíteni (pl. ha a backend nem elérhető, vagy hibás adatot ad vissza).
- A térkép jelenleg fix középpontú, de bővíthető felhasználói pozícióval, kereséssel, szűréssel.
- A kód jól strukturált, de további modulokra bontható (pl. markerkezelő, API-kezelő, UI modulok).

Összefoglalás

Az `assets/js/script.js` a LazarExpressz alkalmazás egyik legfontosabb komponense, amely a valós idejű vasúti forgalom vizuális megjelenítését, a menetrendi adatok feldolgozását és a szerveroldali adatmentést valósítja meg. A kód jól bővíthető, átlátható, és a Leaflet könyvtárnak köszönhetően modern, interaktív felhasználói élményt nyújt.

13. assets/sql/db.sql – Adatbázis séma leírása

Ez a fájl tartalmazza a projekt alapvető adatbázis sémáját, amely a vonatok mozgásának és állapotának naplózására szolgál.

Főbb táblák és mezők

- **trains**: A vonatok aktuális és történeti adatait tárolja.
  - `id` : Egyedi azonosító (auto_increment, elsődleges kulcs)
  - `vehicle_id` : A jármű egyedi azonosítója
  - `train_name` : A vonat neve vagy száma
  - `headsign` : A vonat célállomása vagy iránya
  - `latitude` , `longitude` : Földrajzi koordináták
  - `speed` : Sebesség (km/h)
  - `delay_seconds` : Késés másodpercben
  - `stop_name` : Következő megálló neve
  - `scheduled_time` : Tervezett érkezési idő (TIME típus)
  - `actual_time` : Tényleges érkezési idő (TIME típus)
  - `timestamp` : A rekord mentésének időpontja (alapértelmezett: aktuális idő)

Megjegyzések

- A séma könnyen bővíthető további mezőkkel (pl. szerelvény hossza, vonattípus, stb.).
- A `timestamp` mező lehetővé teszi a vonatok mozgásának időbeli visszakövetését.
- A mezők típusai igazodnak a PHP backend által várt adatokhoz.

14. assets/css/styles.css – Stíluslap leírása

Ez a fájl tartalmazza az alkalmazás alapvető stílusait, amelyek a térkép és a popup táblázatok megjelenését szabályozzák.

Főbb stílusok

- **body, html**: Teljes magasság, margó nélküli elrendezés a térkép teljes képernyős megjelenítéséhez.
- `#map`: 100% magasság és szélesség (viewport), így a térkép kitölti a teljes böngészőablakot.
- **.popup-table**: Kis betűméret, összecsukott táblázat, görgethető, ha túl magas.
- **.popup-table td, .popup-table th**: Vékony szegély, kis padding, balra igazított szöveg.

Megjegyzések

- A stíluslap minimalista, de könnyen bővíthető további elemekkel (pl. gombok, oldalsáv, szűrők).
- A popup táblázat kialakítása segíti a menetrendi adatok áttekinthetőségét.
- A reszponzív elrendezés biztosítja, hogy mobilon is jól használható legyen az alkalmazás.

15. Szerző, cél és felhasználás

Készítette: Csontos Kincső

Cél:
Ez a projekt tanulási célzattal készült, a webfejlesztés, adatvizualizáció és valós idejű adatok feldolgozásának gyakorlására.

Felhasználás:
A forráskód szabadon felhasználható, módosítható és terjeszthető oktatási, tanulási vagy nem kereskedelmi célokra. Kereskedelmi felhasználás vagy átdolgozás esetén kérjük, tüntesd fel az eredeti szerzőt.

Kapcsolat:
Bármilyen kérdés vagy észrevétel esetén keresd a szerzőt!