

ZÁRÓDOLGOZAT

Végi Dániel Márk
2025

RallySync (Admin)

Végi Dániel Márk

**SZÁMALK-Szalézi Technikum és
Szakgimnázium**

Szoftverfejlesztő

Nyilatkozat

a záródolgozat eredetiségéről

Alulírott Végi Dániel Márk (név) { Lengyel Anikó (anyja neve) 326794IE (szem. ig. szám)} büntetőjogi és fegyelmi felelősségem tudatában kijelentem és aláírással igazolom, hogy a záródolgozat saját munkám eredménye. A felhasznált irodalmi és egyéb információs forrásokat az előírásoknak megfelelően kezeltem, a záródolgozat készítésre vonatkozó szabályokat betartottam.

Kijelentem, hogy ahol mások eredményeit, szavait vagy gondolatait idéztem, azt a záródolgozatomban minden esetben, beazonosítható módon feltüntettem, a dolgozatban található fotók és ábrák közzétételével pedig mások szerzői jogait nem sértem.

Kijelentem, hogy a záródolgozatom elektronikus változata teljes egészében megegyezik a nyomtatott formával.

Hozzájárulok ahhoz, hogy az érvényben lévő jogszabályok és a Számalk-Szalézi Szakgimnázium belső szabályzata alapján az iskola saját könyvtárában megtekinthető (olvasható) legyen a záródolgozatom.

A záródolgozat titkos/nem titkos.

Budapest, 2025. április. 30.



Tanuló aláírása

Tartalom

1	Bevezetés	3
1.1	Témaválasztás	3
1.2	Tématörténet	3
2	Tervezés, drótvázak.....	4
2.1	A program célja, lényegesebb funkciói	4
2.2	Futtatási környezet	5
2.3	A program fő funkcióinak leírása	6
2.3.1	Adminisztrátor	6
3	Felhasználói dokumentáció	7
3.1	Futtatási környezet	7
3.1.1	Böngészők.....	7
3.1.2	Operációs rendszer	7
3.1.3	Képernyőfelbontás.....	7
3.2	A program fő funkcióinak leírása	8
3.2.1	Admin:.....	8
4	Fejlesztői dokumentáció	9
4.1	Fejlesztői környezet	9
4.2	Program struktúra	10
4.3	Adatbázis	11
4.3.1	Táblák	11
4.3.2	Egyed-Kapcsolati diagramm	17
4.3.3	Kapcsolati ábra	17
4.4	Backend	18
4.4.1	Modellek és Controllerek	18
4.4.2	API végpontok.....	21
4.5	Frontend	23
4.5.1	Jogosultságkezelés	23

4.5.2	Útvonalak (Routing)	23
4.5.3	Menüpontok	24
4.5.4	Komponensek részletes leírása	24
	Példakomponens: <code>UserController.js</code>	24
4.5.5	Context-ek részletes leírása.....	24
	<code>AuthContext.js</code>	24
4.5.6	CSRF token kezelés.....	25
4.5.7	Reszponzivitás.....	26
5	Tesztelés	27
6	Fejlesztési lehetőségek	28

1 Bevezetés

1.1 Témaválasztás

Kezdetben a csapatunk közösen úgy döntött, hogy egy weboldalt készítünk, amely rally versenyautókat és kapcsolódó adatokat követ nyomon. A csapat egyik tagja jól ismeri ezt a világot, mivel évek óta aktívan figyelemmel kíséri a versenyeket, így sok értékes háttérinformációval tudott hozzájárulni a projekthez.

A fejlesztés során azonban kisebb konfliktus alakult ki, ami végül a csapat szétválásához vezetett. Mivel ez a fordulópont már a munka előrehaladott szakaszában történt, nem láttam értelmét új témát választani. Ehelyett úgy döntöttem, hogy az eddigi elképzelések alapján továbbviszem a projektet, és minden energiámat arra fordítom, hogy egy jól működő adminisztrációs felületet hozzak létre.

A célom az volt, hogy egy egyszerűen kezelhető, átlátható, felhasználóbarát admin felület készüljön el – olyasmi, ami nem bonyolítja, hanem megkönnyíti az adminisztrátor munkáját.

1.2 Tématörténet

A projekt egyik kiindulópontja csapattársunk tapasztalata volt, miszerint a rallyversenyekről ritkán kerül nyilvánosságra részletes, rendszerezett információ. A legtöbb esemény nem jelenik meg hivatalos platformon, legfeljebb utólag, nézők által feltöltött videók formájában válik elérhetővé. Ezt a hiányosságot felismerve merült fel bennünk az ötlet, hogy egy olyan weboldalt hozzunk létre, amely képes strukturáltan nyomon követni a versenyeket, és amely – megfelelő fejlesztés és tartalomfeltöltés után – akár a szervezők számára is egy használható, hivatalos felületté válhat.

Az adminfelület kialakításánál elsődleges szempontként a felhasználóbarát működésre törekedtem. A cél az volt, hogy egy olyan kezelőfelület jöjjön létre, amelyet akár technikai háttérrel kevésbé rendelkező személyek is magabiztosan tudnak használni. A funkciók elrendezését, az űrlapokat és az adminfolyamatokat úgy terveztem meg, hogy azok logikusak, egyértelműek és könnyen elsajátíthatók legyenek.

2 Tervezés, drótvázak

2.1 A program célja, lényegesebb funkciói

A program elsődleges célja, hogy megbízható és könnyen elérhető információforrást biztosítson a rallyversenyekkel kapcsolatban. Az alkalmazás olyan adatokat gyűjt és jelenít meg egy helyen, mint a versenyek dátumai, helyszínei, eredmények, versenyzői helyezések és rekordok. A cél, hogy ezek az információk egy központi felületen kereshetően és áttekinthetően álljanak a felhasználók rendelkezésére.

Jelenleg az elérhető online platformok jellemzően kizárólag részleges információkat szolgáltatnak – például csak versenyidőpontokat vagy csak eredménylistákat. Emellett a versenyszervezés folyamata sem egységesített: a szervezők jellemzően személyes kapcsolatokon vagy e-mailes egyeztetésen keresztül veszik fel a kapcsolatot a pályatulajdonosokkal, ami sokszor nehézkes és átláthatatlan.

Az általunk fejlesztett rendszer hosszú távú célja, hogy ezt a folyamatot is digitalizálja, lehetővé téve a szervezők számára a versenyek online benyújtását és menedzselését, ezáltal hatékonyabbá és átláthatóbbá téve a teljes eseményszervezést.

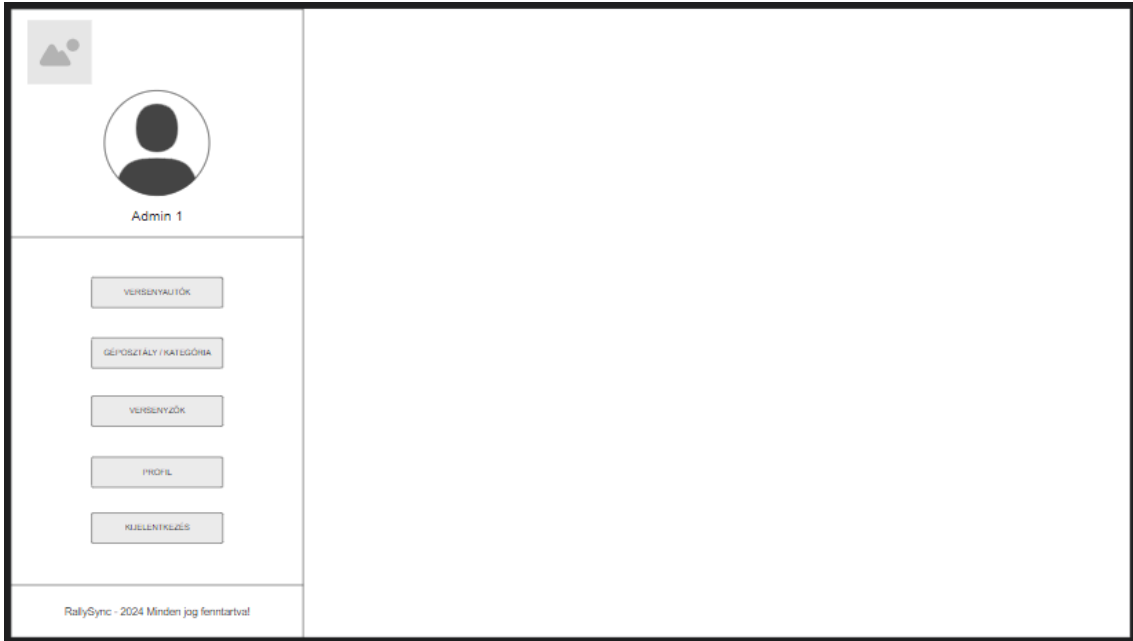
Főbb Funkciók:

ADMIN:

- Új Pálya, Státusz, Jogosultság, Márka típus és Kategória felvitele, módosítása és törlése
- Új versenyautó felvitele, módosítása és törlése
- Verseny módosítása és törlése
- User names módosítása

2.2 Futtatási környezet

-- Admin --



Admin interface mockup. The left sidebar contains a user profile section with a placeholder image and the name "Admin 1". Below the profile are five buttons: "VERSENYAUTÓK", "GÉPOSZTÁLY / KATEGÓRIA", "VERSENYZŐK", "PROFIL", and "KIELENTKEZÉS". The main content area is empty. The footer text is "RallySync - 2024 Minden jog fenntartva!".



User management interface mockup. The left sidebar is identical to the Admin interface. The main content area has a title "Felhasználó" and a table with three columns: "Szervező", "Versenyző", and "Néző". The table contains one row with the following data:

Szervező	Versenyző	Néző
Lakatos Amígő	Lakatos Amőba	Arkana Lakatos

The footer text is "RallySync - 2024 Minden jog fenntartva!".

2.3 A program fő funkcióinak leírása

2.3.1 Adminisztrátor

- Versenyautók felvitele / módosítása / törlése géposztály alapján.
- Kategória létrehozása / módosítása / törlése.
- Verseny helyszínek létrehozása / módosítása / törlése.
- A megszervezett versenyek listázása.
- Versenyeken részt vett versenyzők listázása.
- Versenyzők és kategóriánként az autók listázása.

3 Felhasználói dokumentáció

3.1 Futtatási környezet

A weboldal futtatásához nincs szükség speciális környezetre, ugyanakkor bizonyos platformok használata javasolt az optimális élmény érdekében.

3.1.1 Böngészők

A rendszer működése a legtöbb modern böngészőben támogatott. Az ajánlott böngészők a következők:

- **Google Chrome** (ajánlott)
- **Brave**
- **Microsoft Edge**
- **Mozilla Firefox**

A weboldal JavaScript alapú, és a **React** könyvtárat használja az interaktív felület megvalósításához, valamint **Bootstrap** keretrendszert a reszponzív megjelenés biztosításához.

3.1.2 Operációs rendszer

A webalkalmazás platformfüggetlen, de fejlesztés és tesztelés során az alábbi rendszerekre lett optimalizálva:

- **Windows 10**
- **Windows 11**

3.1.3 Képernyőfelbontás

Nem szükséges konkrét képernyőméret vagy felbontás, mivel a felület a **Bootstrap** keretrendszernek köszönhetően teljes mértékben **reszponzív**, azaz automatikusan alkalmazkodik mobiltelefonok, táblagépek és asztali számítógépek kijelzőjéhez.

3.2 A program fő funkcióinak leírása

3.2.1 Admin:

Az adminfelületen az alábbi főbb műveletek elvégzésére van lehetőség:

- Új pálya, státusz, jogosultság, márkatípus és kategória hozzáadása, módosítása és törlése.
- Új versenyautók rögzítése az adatbázisban, valamint meglévő autók adatainak módosítása vagy törlése.
- Versenyek adatainak szerkesztése és szükség esetén törlése.
- Felhasználónevek módosítása az adminisztrációs felületen keresztül.

Az adminfunkciók kialakításánál szempont volt, hogy minden művelet egyszerűen, néhány kattintással elvégezhető legyen, a hibalehetőségeket pedig minimalizáljuk.

4 Fejlesztői dokumentáció

4.1 Fejlesztői környezet

A projekt fejlesztése során a React könyvtárra építettem, mivel jól kezelhető komponensalapú rendszert biztosít, és korábban is volt már vele tapasztalatom. A felülethez a Bootstrap keretrendszert használtam, mivel gyorsan lehet vele reszponzív elemeket létrehozni, így mobilra és asztalra egyaránt jól alkalmazkodik az oldal.

A fejlesztést Windows 10 és 11 operációs rendszeren végeztem, főként **Visual Studio Code** szerkesztőben. A verziókezeléshez **Git**-et használtam, a kódot **GitHub**-on tároltam. A projekthez **npm**-mel telepítettem a szükséges csomagokat.

Fejlesztés közben rendszeresen teszteltem a weboldalt több böngészőben is (Chrome, Edge, Firefox), és figyeltem arra is, hogyan viselkedik különböző képernyőméreteken.

Összességében egy egyszerű, átlátható fejlesztői környezetet alakítottam ki, ahol kényelmesen lehetett dolgozni, tesztelni, és bármikor bővíteni az oldalt új funkciókkal.

React Verzió: 19.0.0

PHP Verzió: 8.2

Laravel Verzió: 11.31

4.2 Program struktúra

A projekt egy React-alapú alkalmazás, amely komponensekre épül. A cél az volt, hogy az egyes részek jól elkülöníthetők legyenek, ezáltal könnyen bővíthető és átlátható maradjon a kód.

A mappastruktúra a következő főbb részekből áll:

- **/src** – Az alkalmazás fő mappája, itt található minden fejlesztéssel kapcsolatos fájl.
 - **/components** – Itt helyezkednek el az újrahasználatos React komponensek, például gombok, űrlapmezők, navigációs sávok stb.
 - **/pages** – Az egyes oldalak (pl. Admin, Szervező stb) külön-külön komponensként szerepelnek.
 - **/css** – Minimális css stílus változtatások.
 - **/api** – Itt kapnak helyet az API-hívások és az adatkezeléshez kapcsolódó logikák.
 - **/navigation** – Itt vannak a különböző navigációk a különböző felhasználóknak.
 - **/layouts** – Itt vannak az alap layout oldalak a különböző felhasználóknak.
 - **/context** – Itt helyezkednek el contextek különböző dolgokhoz (pl. APIContext.js, AuthContext.js)
 - **/App.js** – Az alkalmazás főkomponense, itt állnak össze az oldalak és komponensek egyetlen működő alkalmazássá.
 - **/index.js** – A belépési pont, ahol a React dom renderelése történik.

Az adminfelület külön figyelmet kapott, mivel itt történik a versenyek, versenyzők és rekordok kezelése. Minden fontos funkció külön komponensben szerepel, így később könnyen cserélhetők vagy továbbfejleszthetők.

4.3 Adatbázis

4.3.1 Táblák

PERMISSION(id, permission)

A felhasználók szerepköreinek meghatározása.

CATEGORY(id, category)

A versenyeken induló járművek kategóriáinak tárolása.

BRANDTYPE(id, brand_type)

Az autók márkájának és típusának nyilvántartása.

PLACE(id, place)

A versenyek helyszíneinek és azok hosszának tárolása.

USER(id, name, email, permission, password) | jogos: versenyző, szervező, admin.

A rendszerben regisztrált felhasználók adatainak tárolása.

CAR(id, brand_type, category, status)

A versenyautók adatainak tárolása, beleértve a márkát, típust, kategóriát és az állapotot.

VERSENY(id, helyszín, szervező, leírás, mettől, meddig)

A versenyek alapvető adatainak tárolása.

VERS-KATEG(id, verseny, kategória, min_létszám, max_létszám)

Az egyes versenyekhez tartozó kategóriák és azok minimális és maximális létszámának tárolása.

BENEVEZ(verseny, versenyző, kategória)

A versenyzők nevezésének nyilvántartása egy adott verseny és kategória szerint.

VERSENYZIK(verseny, versenyző, autó, érkezik, rajt_idő, cél_idő)

A versenyzés eseményeinek tárolása.

Palya			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
helyszin	varchar	Kötelező	Egyedi

Jogosultsag			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
jogosultsag	varchar	Kötelező	egyedi

Marka-tipus			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
marka_tipus	varchar	Kötelező	

User			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
nev	varchar	Kötelező	
email	varchar	Kötelező	
jogosultság	bigint	KK	default(1)
password	varchar	Kötelező	

Auto			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
marka_típus	int	KK	
kategória	int	KK	
status	int	KK	

Kategoria			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
kategoria	int	Kötelező	

Verseny			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
verseny_nev	varchar	Kötelező	
helyszin	int	KK	
szervezo	int	KK	
leiras	varchar	Kötelező	
kezdet_datum	date	Kötelező	
vege_datum	date	Kötelező	Null or vd > kd

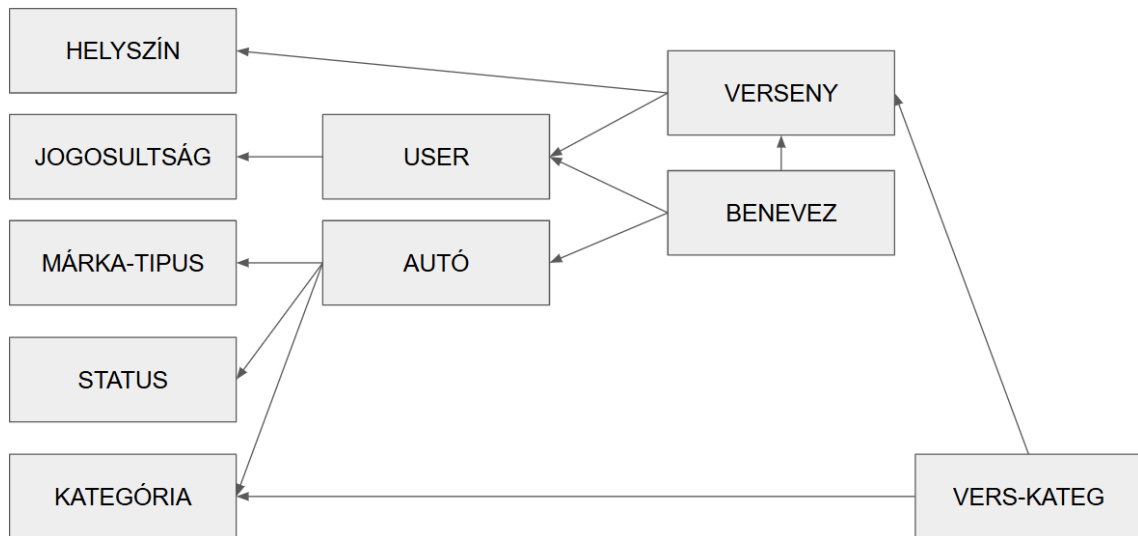
Versenyzik			
Mező	Tipus	Szerep	Korlátozás
verseny	int	KK	
versenyző	int	KK	
auto	int	KK	
érkezik	date	Kitölthető	
rajt-ido	date	Kitölthető	
cel-ido	date	Kitölthető	rajt-ido < cel-ido

Benevez			
Mező	Tipus	Szerep	Korlátozás
versenyző	int	KK	
verseny	int	KK	
kategoria	int	KK	

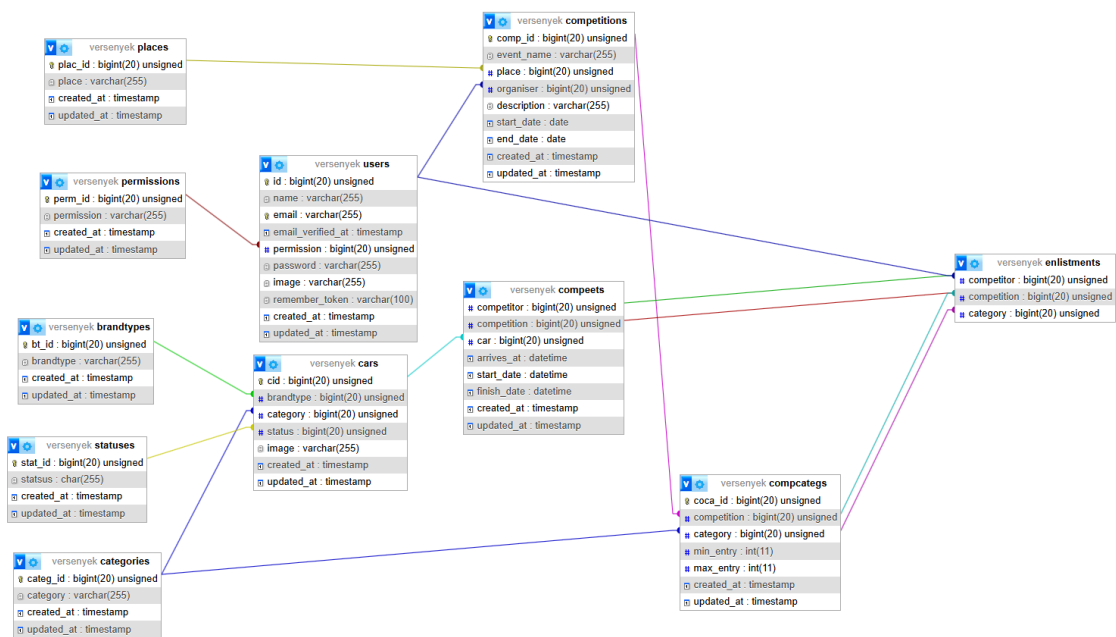
Verseny_kateg			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
kategoria	int	KK	
verseny	int	KK	
min_resztv	int	Kötelező	
max_resztv	int	Kötelező	

Status			
Mező	Tipus	Szerep	Korlátozás
id	int	K(generált)	
status	varchar	Kötelező	

4.3.2 Egyed-Kapcsolati diagramm



4.3.3 Kapcsolati ábra



Artista Windows

XI. ábra: Adatbázis kapcsolati ábrája.

4.4 Backend

4.4.1 Modellek és Controllerek

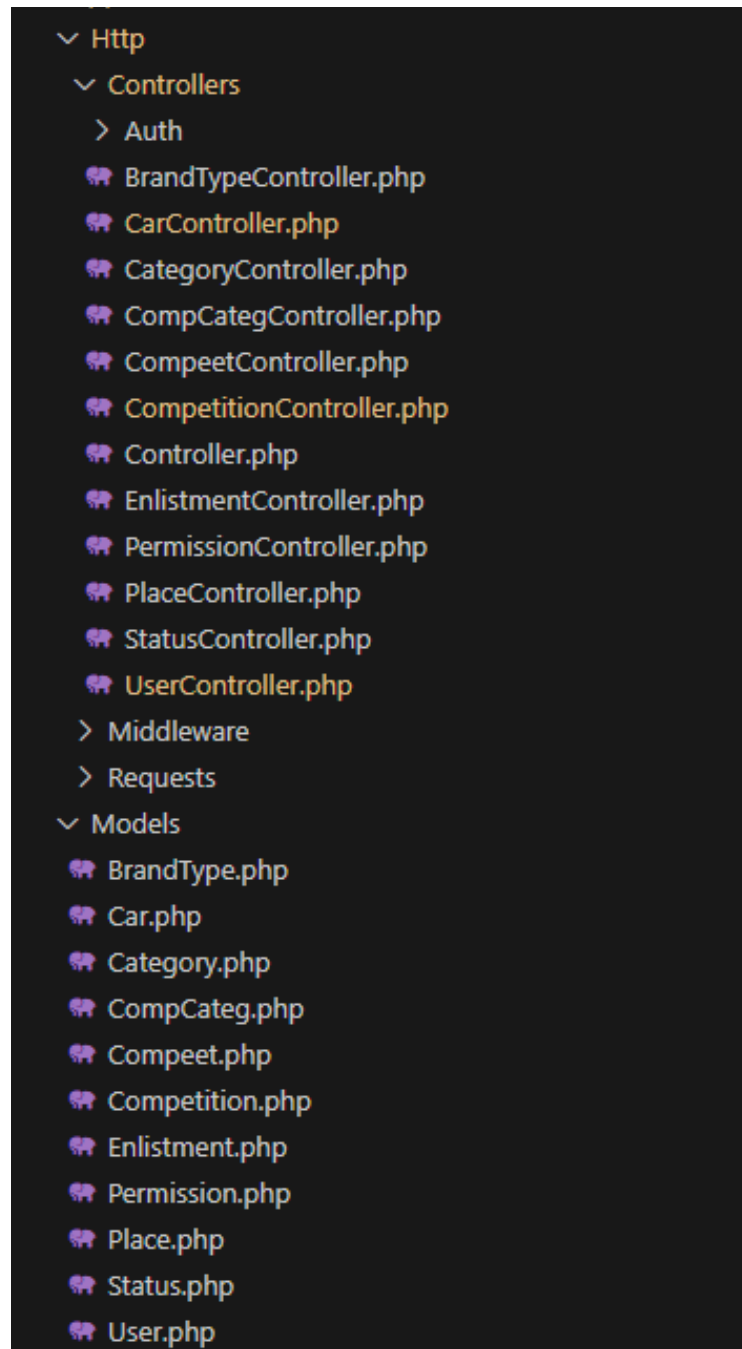
A Backend írása során számtalan Model és Controller file került létrehozásra, amelyek azonnal felhasználódtak. A Modellek feladata az táblákhoz tartozó adatok elérésének segítése. Ezek a php kiterjesztésű filek a Laravel backend oldalon, az “app/Models/” mappán belülre jönnek létre. Programunk írása során a következő Modellek kerültek létrehozásra:

- **Car:** ./app/Models/Car.php
- **Place:** ./app/Models/Place.php
- **Compeet:** ./app//Models/Compeet.php
- **Competition:** ./app/Models/Competition.php
- **Status:** ./app/Models/Status.php
- **Permission:** ./app/Models/Permission.php
- **CompCateg:** ./app/Models/CompCateg.php
- **Category:** ./app/Models/Category.php
- **User:** ./app/Models/User.php
- **BrandType:** ./app/Models/BrandType.php
- **Enlistment:** ./app/Models/Enlistment.php

Ezek mellett Controllerekre is szükség volt a Backend írása során, amelyek az adatokat szolgáltatja a Frontend és az adatbázisban szereplő táblák között. Ezek felhasználásával az adatokat el tudjuk érni, módosítani és akár a törölni is. Hasonlóan a Model mappához, ezt is az app mappán keresztül van lehetőség elérni, viszont itt további mappák szükségesek a Controllerhez jutáshoz. Az app mappán belül, a http/Controller mappa alapértelmezettként szolgál az újonnan létrejött Controllereknek, így azok tárolására szolgálnak. Minden Modelhez készítettünk egyet-egyet, hogy minden táblában szereplő adatok elérhetőek lehessenek:

- **Car:** ./app/http/Controllers/CarController.php
- **Place:** ./app/http/Controllers/PlaceController.php
- **Compeet:** ./app/http/Controllers/CompeetController.php
- **Competition:** ./app/http/Controllers/CompetitionController.php
- **Status:** ./app/http/Controllers/StatusController.php
- **Permission:** ./app/http/Controllers/PermissionController.php
- **CompCateg:** ./app/http/Controllers/CompCategController.php
- **Category:** ./app/http/Controllers/CategoryController.php
- **User:** ./app/http/Controllers/UserController.php
- **BrandType:** ./app/http/Controllers/BrandTypeController.php
- **Enlistment:** ./app/http/Controllers/EnlistmentController.php

Mappa struktúra:



4.4.2 API végpontok

4.4.2.1 Admin:

Kérés típusa	URI	Leírás
Get	/competitionGet	Az összes verseny lehívása
Get	/competition/{id}	Egy megadott verseny lehívása
Patch	/competitionModify/{competitionId}/{categoryId}	Egy megadott verseny megváltoztatása
Delete	/competitionDelete/{competitionId}/{categoryId}	Egy megadott verseny törlése
Get	/compcategs	Az összes verseny kategória lehívása
Get	/compcateg/{id}	Egy megadott verseny kategória lehívása
Patch	/compcategUpdate/{id}	Egy megadott verseny kategória megváltoztatása
Get	/carGet	Az összes auto lehívása
Post	/carCreate	Egy új auto létrehozása
Patch	/carModify/{id}	Egy megadott autó módosítása
Delete	/carDelete/{id}	Egy megadott auto törlése
Post	/carUploadImage	Egy kép felvittele autókrol
Get	/categGet	Az összes kategória
Post	/categCreate	Egy új kategória létrehozása
Patch	/categModify/{id}	Egy megadott kategória megváltoztatása
Delete	/categDelete/{id}	Egy megadott kategória törlése
Get	/placeGet	Az összes hely megjelenítése
Post	/placeCreate	Egy új hely létrehozása
Patch	/placeModify/{id}	Egy megadott hely megváltoztatása
Delete	/placeDelete/{id}	Egy megadott hely törlése
Get	/brandtypeGet	Az összes márka típus lehívása
Post	/brandtypeCreate	Egy új márka típus létrehozása
Patch	/brandtypeModify/{id}	Egy megadott márka típus megváltoztatása
Delete	/brandtypeDelete/{id}	Egy megadott márka típus törlése
Get	/statusGet	Az összes státusz lehívása
Post	/statusCreate	Egy új státusz létrehozása
Patch	/statusModify/{id}	Egy megadott státusz megváltoztatása
Delete	/statusDelete/{id}	Egy megadott státusz törlése
Patch	/userAdminModify/{id}	Az adminnak egy külön megváltoztatás ahol egy

		usernek meg tudja változtatni a user jogosultságát
--	--	---

4.4.2.2 Public:

Get	<code>/userGet</code>	Az összes user lekérdezése
Get	<code>/users/{id}</code>	Egy megadott user megjelenítése
Patch	<code>/userModify/{id}</code>	Egy megadott user megváltoztatása
Delete	<code>/userDelete/{id}</code>	Egy megadott user törlése
Patch	<code>/users/{id}/update-password</code>	Egy megadott user jelszójának megváltoztatása
Post	<code>/userUploadImage</code>	Egy új kép felvitele a usernek
Post	<code>/register</code>	Egy új user regisztrálása
Post	<code>/login</code>	Egy user bejelentkezése

4.5 Frontend

4.5.1 Jogosultságkezelés

Három alapvető szerepkör van definiálva:

- **Versenyző:** Csak saját adatok elérése.
- **Szervező:** Versenyek létrehozása, módosítása.
- **Admin:** Teljes jogosultság minden adat kezelésére.

A jogosultságok meghatározzák, hogy a felhasználó milyen menüpontokat és funkciókat lát és használhat.

4.5.2 Útvonalak (Routing)

Az alkalmazás főbb útvonalai az alábbiak szerint kerültek kialakításra:

- **/login – Bejelentkezés oldal**
Itt történik a felhasználók hitelesítése. A sikeres bejelentkezés után a felhasználó a jogosultsága alapján kerül átirányításra a saját felületére.
- **/register – Regisztrációs oldal**
Új felhasználók itt tudnak fiókot létrehozni. Regisztrációkor alapértelmezetten "versenyző" jogosultságot kapnak, magasabb szintű jogosultságokhoz adminisztrátori jóváhagyás szükséges.
- **/ – Központi átirányító (Landing Page)**
Ha a felhasználó nincs bejelentkezve, egy publikus kezdőlapra kerül. Bejelentkezett felhasználó esetén a rendszer a jogosultság szintjétől függően automatikusan a saját felületére irányítja:
 - **Admin** – Admin felület
 - **Szervező** – Szervezői felület
 - **Versenyző** – Versenyzői felület
- **Publikus kezdőlap**
Azok a látogatók, akik nincsenek bejelentkezve, itt tájékozódhatnak az oldal céljáról, a regisztrációs lehetőségről, valamint általános információkról a versenyekről.

4.5.3 Menüpontok

Menü dinamikusan generálódik a jogosultság alapján:

- **Admin:** Felhasználók kezelése, Versenyek kezelése, Autók kezelése, Kategóriák kezelése, új adatok felvitele.
- **Szervező:** Versenyek létrehozása, saját versenyek kezelése.
- **Versenyző:** Saját nevezések megtekintése, autók regisztrációja.

4.5.4 Komponensek részletes leírása

Példakomponens: `UserController.js`

- **Függvények:**
 - `adminUpdate(id)`
Paraméter: `id` (felhasználó azonosító).
Feladata: Felhasználó módosítása admin API végponton keresztül.
- **State változók:**
 - `users` – A jelenlegi felhasználók listáját tárolja.

4.5.5 Context-ek részletes leírása

`AuthContext.js`

- **State-ek:**
 - `user` – A bejelentkezett felhasználó adatai.
 - `isAuthenticated` – Be van-e jelentkezve a felhasználó.
- **Függvények:**
 - `login(email, password)`
Feladata: Bejelentkezteti a felhasználót, menti a token-t és beállítja a státuszt.
 - `logout()`
Feladata: Kijelentkezteti a felhasználót, törli a token-t és a státuszokat.

4.5.6 CSRF token kezelés

A biztonságos adatküldés érdekében az alkalmazás CSRF (Cross-Site Request Forgery) tokeneket használ. A CSRF tokenek célja, hogy megvédjék az alkalmazást az olyan támadásoktól, ahol egy rosszindulatú oldal megpróbál jogosulatlan műveleteket végrehajtani a felhasználó nevében.

Frontend oldalon a CSRF tokenek kezelése a következő módon történik:

- **Token átvétele:**
A felhasználó bejelentkezésekor vagy az oldal betöltésekor a szerver egy egyedi CSRF tokenet generál és visszaküld. Ez a token általában HTTP Only sütiben vagy API válaszban érkezik.
- **Token tárolása:**
A frontend nem közvetlenül olvassa ki a HTTP Only sütiben tárolt token (mert a böngésző automatikusan csatolja a kérésekhez), de ha a token külön fejlécben vagy bodyban érkezik, azt ideiglenesen a memóriában (pl. context-ben vagy state-ben) tároljuk.
- **Token továbbküldése:**
Minden olyan API-hívásnál, ami adatot módosít (POST, PUT, PATCH, DELETE), a frontend automatikusan csatolja a CSRF tokenet egy HTTP fejlécben (pl. X-CSRF-Token) vagy a kérés törzsében.
- **Sikertelen hitelesítés kezelése:**
Ha a szerver azt jelzi, hogy a CSRF token érvénytelen vagy hiányzik, a frontend automatikusan újratöltheti az oldalt, új tokenet kérhet, vagy átirányíthatja a felhasználót a bejelentkezési oldalra.

4.5.7 Reszponzivitás

Reszponzivitáshoz egyedül Bootstrap keretrendszert használok. Ez automatikusan reszponzívvá teszi ha az különböző elemei jól vannak felhasználva. A Bootstrap egy olyan front-end keretrendszer amit legfőképpen weboldalakon és webalkalmazásokban használnak fel. Alapvetően HTML, CSS és JavaScript használja fel. Itt a Bootstrap előre definiál különböző stílusokat és különböző komponenseket is felajánl. Ilyenek például a reszponzív rácsrendszer, előre elkészített gombok, úrlaapok, navigációs sávok. Ezen túl a bootstrap alapból mobilbarát. Legegyszerűbb felhasználása a bootstrapnek a CDNen keresztül megy.

Pl:

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
```

Alapértelmezett törésvonalak:

Méretosztály	Eszköz típusa	Szélesség (min-width)
xs	Extra kicsi (mobil)	< 576px
sm	Kis (mobil)	≥ 576px
md	Közepes (tablet)	≥ 768px
lg	Nagy (laptop)	≥ 992px
xl	Extra nagy (asztali)	≥ 1200px
xxl	Nagyon nagy	≥ 1400px

5 Tesztelés

AZ ÖSSZES TESZTESET

Teszteset / Bemeneti paraméterek	Várt eredmény	Kapott eredmény	Státusz
Permission::create -> teszt	DatabaseHas: teszt	teszt	Pass
Brandtype::create -> Lada 2107	DatabaseHas: Lada 2107	Lada 2107	Pass
Place::create -> Hungaroring	DatabaseHas: Hungaroring	Hungaroring	Pass
User::create -> testelek@example.com ...	DatabaseHas: testelek@example.com	testelek@example.com	Pass
Car::create -> bt_id ...	DatabaseHas: bt_id	Bt_id	Pass
Migrations created	Creates all	Creates all	Pass

6 Fejlesztési lehetőségek

Az elkészült rendszer már jelenlegi formájában is jól használható az adminisztrációs feladatok ellátására, ugyanakkor számos irányban továbbfejleszthető:

- **További felhasználói felületek kibővítése:**
A jövőben ki lehet bővíteni az admin oldalt a szervező és versenyző felhasználói felületekkel.
- **Élő versenykövetés funkció:**
A versenyek alatt élőben frissülő adatok (pl. indulási és célidők) megjelenítése növelné az oldal vonzerejét és hasznosságát a nézők számára is.
- **Email értesítések és automatikus értesítőrendszer:**
Például sikeres nevezésről, verseny módosításáról vagy új eseményekről automatikus e-mail vagy push értesítések küldése.
- **Statistikák és eredménykimutatások:**
Lehetőség lenne részletes statisztikák megjelenítésére a versenyekről, versenyzőkről és autókról, grafikonok és kimutatások formájában.
- **Többnyelvű támogatás:**
A rendszer többnyelvűvé tétele (például angol, német) segítené a nemzetközi versenyek támogatását és a külföldi felhasználók bevonását.