Rentaruro

Autókölcsönző

Készítette:

Kántor Krisztián Péter

Sebestyén Milán Gábor

Városi Bendegúz Álmos

Nyilatkozat

Kijelentjük, hogy a programkód teljes szövege, valamint a dokumentáció teljes egésze saját szellemi termékünk, melyet önállóan készítettünk.

Tartalomjegyzék

[I. Feladatspecifikáció 4](#_Toc197938002)

[II. Alkalmazott szoftverek és technológiák 4](#_Toc197938003)

[III. Projekt telepítése 4](#_Toc197938004)

[IV. A felület funkciói, felépítése és használata 4](#_Toc197938005)

[V. Adatbázis 4](#_Toc197938006)

[VI. API végpontok 4](#_Toc197938007)

[VII. Tesztelés 4](#_Toc197938008)

[VIII. Források 5](#_Toc197938009)

# Feladatspecifikáció

Az általunk fejlesztett alkalmazás segíti egy autókölcsönző mindennapjait, a felhasználónak egy egyszerűbb és könnyebben használható felületet biztosít a jobb élményért.

A Rentaruro több felhasználói szinttel rendelkezik: ügyfél, dolgozó és admin. A célja az autófoglalás, illetve az adminisztratív feladatok hatékony és precíz elvégzése.

Célunk, hogy minél több embernek nyújtsunk elérhető áron autókat, azok naprakészségét, igényes felszereltségét biztosítsuk, hogy minden felhasználó zökkenőmentesen és biztonságosan járja az utakat.

# Alkalmazott szoftverek és technológiák

* 1. Szoftverek:
* XAMPP:

phpMyAdmin 5.2.1

MySQL 10.3.38

Apache 2.4.56

MariaDB 10.4.28

* Google Chrome Verzió: 123.0.6312.86
* Microsoft Word 2021 LTSC
* Adobe Photoshop 2023
* GitHub Desktop v3.3.8
* Visual Studio Code 1.87.2
* Android Studio 2024.3.1
* Visual Studio Community 2022 17.12.4
* IntelliJ IDEA 2025.1
  1. Programnyelvek, keretrendszerek és modulok

**Frontend**

* Nyelv: JavaScript
* Keretrendszer: React.js
* Routing: React Router DOM
* React: 18.3.1
* React Query (TanStack Query): 5.56.2
* Vite
* Lucide React
* Tailwind CSS
* Shadcn/UI
* Nyelv: C#
* Keretrendszer: .NET Framework
* NuGet: Newtonsoft.Json

**Backend**

* Nyelv: JavaScript (Node.js)
* Web Framework: Express.js
* Adatbázis Kezelő: MySQL
* Authentikáció és Autorizáció: bcrypt, jsonwebtoken
* HTTP Kliens: axios
* CSV Fájlkezelő: csv-parser
* Környezeti változók kezelése: dotenv
* Middleware: body-parser, cors, multer
* Egyéb: assert(beépített Node.js modul)

# Projekt telepítése

A projekt működéséhez elengedhetetlen, hogy a node és az xampp le legyen töltve.

Először ki kell választani a mappát, ahová telepíteni akarjuk a github repositorit. Ha ezt kiválasztottuk, akkor be kell írni a következő parancsot a parancssorba:

git clone https://github.com/Tsikkk/13-projekt.git

Miután letöltött, a server mappában a parancssorba be kell írni a következő parancsokat:

npm i express

npm i sequelize

npm i cors

npm i jsonwebtoken

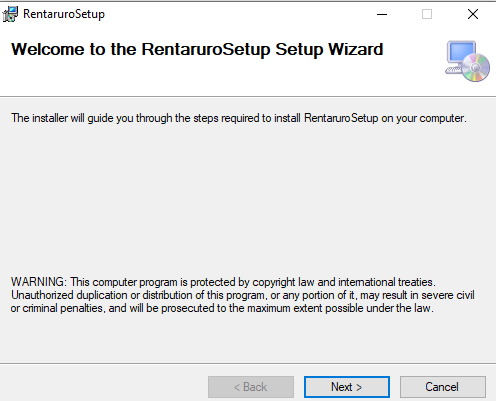
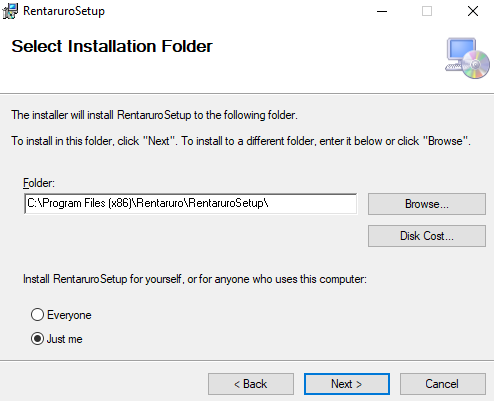
npm i dotenv

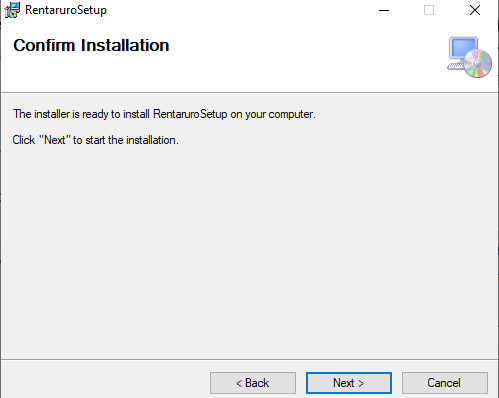
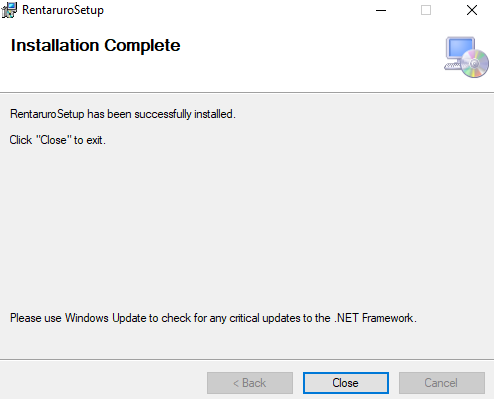
npm i mysql2

Ahhoz, hogy elinduljon a server előtte a xampp alkalmazában el kell indítani a MySql-t.

Ha ez megtörtén, akkor a server mappában a parancssorba be kell írni a következő parancsot:

node server.js

Az asztali alkalmazás letöltéséhez a RentaruroSetup.msi fájlra duplán kattintva elindul a telepítés. A Next> gombra kattintva felhoz egy új ablakot, ahol ki kell választani, hogy hová szeretné telepíteni az alkalmazást és ki tudjon hozzáférni a számítógépén. Majd szintén a Next> gombra kattintva egy új ablakon megkérdezi, hogy biztosan akarja-e telepíteni az alkalmazás. Ha, újra a Next> gombra kattint, akkor elindul a telepítés. Engedélyezni kell a telepítést. Ha, letöltött bezárhatja az ablakot.



Erre az ikonra kattintva elindul az alkalmazás:

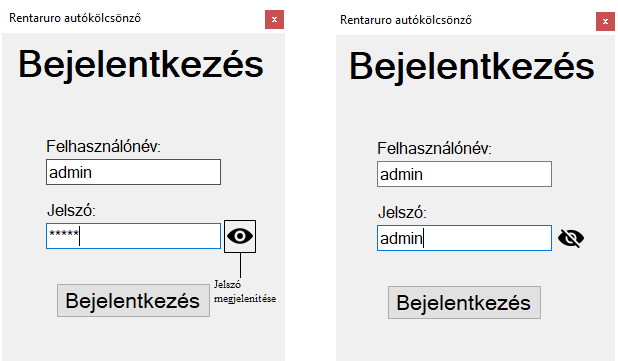


# A felület funkciói, felépítése és használata

A felületek felépítésénél számos dologra törekedtünk, de legfőképp az egyszerűségre és az átláthatóságra.

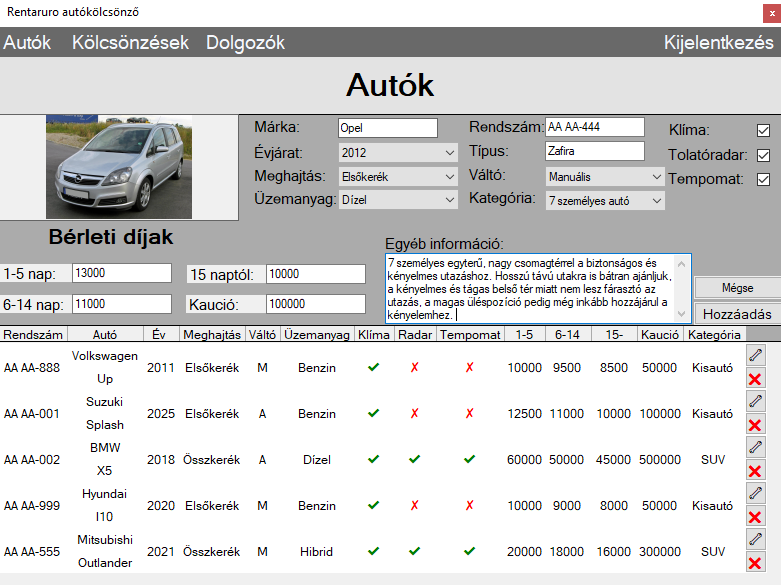
1. Natív asztali alkalmazás

Ezen a felületen lehet kezelni az autókat, kölcsönzéseket és a dolgozókat. Ezekhez azonban csak belépéssel lehet hozzáférni, ezért a szerver létrehoz egy új felhasználót admin névvel és admin jelszóval.



* 1. Autók

Bejelentkezés után megjelenik egy új ablak, ahol egy listában egymás alatt láthatók az autók adatai: rendszám, márka, modell, évjárat, meghajtás, váltó, üzemanyag, klíma, tolatóradar, tempomat, árak, kaució, kategória. Ezeknek az adatoknak az alapján lehet sorrendet alkotni. Autót hozzáadni, törölni és módosítani kizárólag admin jogosultsággal rendelkező felhasználó képes. Egy rendszám csak egyszer szerepelhet az adatbázisban.

Admin jogosultsággal rendelkező felhasználói felület:

Módosítás gomb

Törlés gomb

Lista elem

Lista

Lista fejléc

Autó hozzáadása

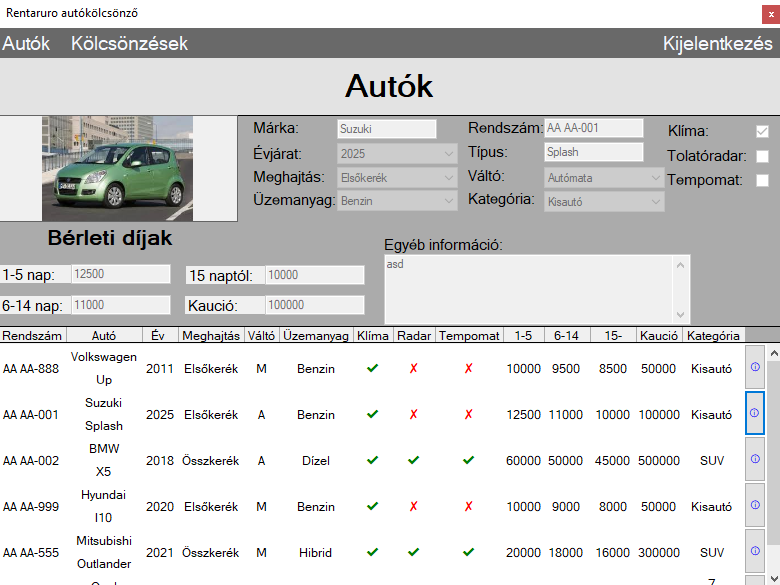
Fejléc

Hozzáadáshoz szükséges adatok: rendszám, márka, modell, fénykép az autóról, évjárat, meghajtás, váltó, üzemanyag, klíma, tolatóradar, tempomat, ár 1 és 5 nap között (Ft/nap), ár 6 és 14 nap között (Ft/nap), ár 15 nap fölött (Ft/nap), kaució, kategória, egyéb információ az autóról. A hozzáadás gombra kattintva hozzáadja az autót az adatbázishoz, a mégse gombra kattintva üresre állítja az adatokat. Minden adatot ki kell tölteni, hogy sikeresen hozzáadja az autót az adatbázishoz.

Az elemek jobb oldalán látható 🔧-gombra kattintva visszatölti az autó adatait, és a Hozzáadás gomb szövegét átírja Módosítás-ra. A gomb lenyomásával frissíti az autó adatait. Módosítani a rendszámon kívül minden adatot lehet.

A 🔧 -gomb helyett dolgozó hozzáféréssel rendelkezőknek a ⓘ -gomb jelenik meg. A gomb lenyomásával megtekinthetik az autó további információit, amelyet a lista nem tartalmaz.

A ✖ -gombra kattintva megjelenik egy ablak, amely megkérdezi, hogy biztosan szeretné-e törölni az autót az adatbázisból. Az igen gombra kattintva az autót törli.

Dolgozói jogosultsággal rendelkező felhasználói felület:

További információk gomb

sta

Lista elem

Lista

Lista fejléc

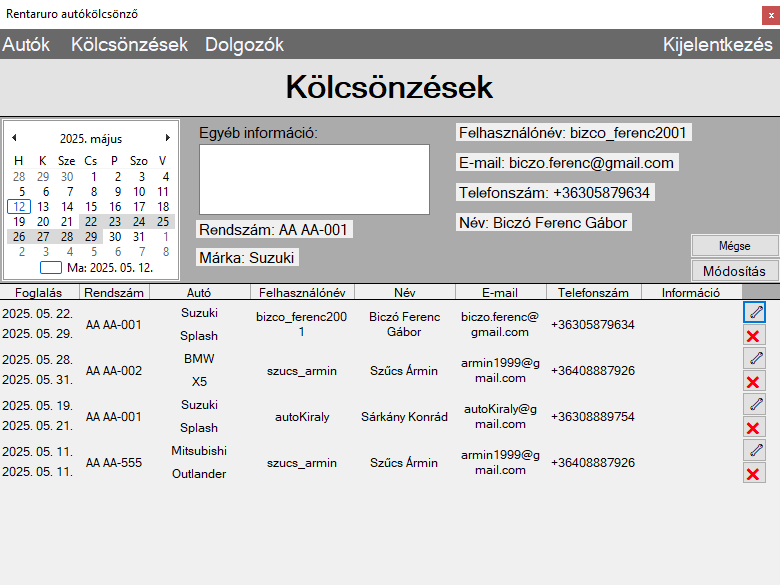
Autó adatai

Fejléc

* 1. Kölcsönzések

A fejlécben lévő Kölcsönzések gombra kattintva megjelenik a kölcsönzések listája. Ez a lista a kölcsönzés kezdetét, végét, az autó rendszámát, márkáját, modelljét, a felhasználó nevét, e-mail-címét, telefonszámát, felhasználónevét és egyéb kérését tartalmazza. Az elemek egymás alatt jelennek meg. Új kölcsönzést ezen a felületen nem lehet létrehozni, mindössze a meglévők hosszát és egyéb információit lehet módosítani, vagy a kölcsönzést törölni.

Az elemek jobb oldalán látható 🔧-gombra kattintva visszatölti a bérlés kezdetét, végét és egyéb információját. Ha ugyanarra az autóra van több foglalás, akkor a foglalást nem lehet úgy módosítani, hogy azok időpontjai bármilyen módon egybe essenek. A módosítás gomb lenyomásával frissíti a szoftver az autó adatait.

A ✖ -gombra kattintva megjelenik egy ablak, amely megkérdezi, hogy biztosan szeretné-e törölni a foglalást az adatbázisból. Az igen gombra kattintva törli a foglalást.

Lista elem

Lista

Lista fejléc

Módosítás gomb

Törlés gomb

Kölcsönzés információ

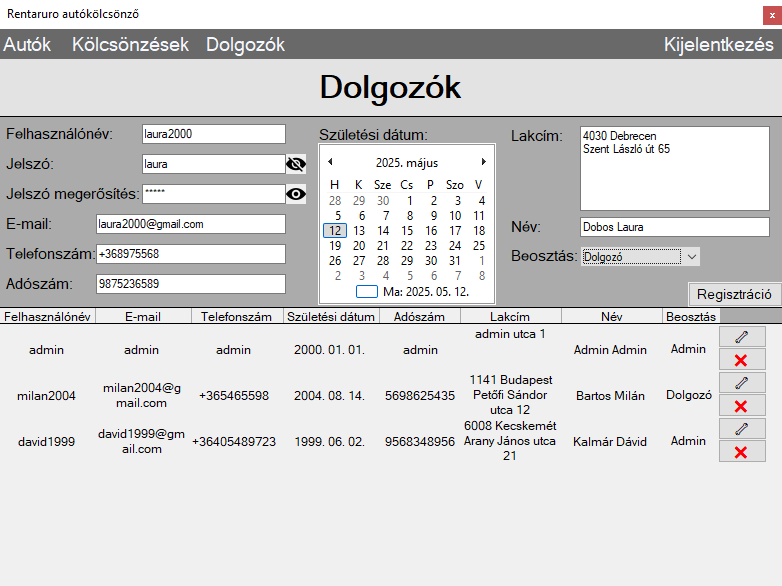
Fejléc

* 1. Dolgozók

A fejlécben lévő Dolgozók gomb kizárólag admin jogosultsággal rendelkező felhasználóknak jelenik meg. A gombra kattintva egy listában látható az összes dolgozó felhasználóneve, e-mailje, telefonszáma, születési dátuma, adószáma, lakcíme, neve és beosztása.

Új dolgozó megjelenítéséhez szükséges adatok: felhasználónév, jelszó, e-mail, telefonszám, adószám, születési dátum, lakcím, név, beosztás (admin/dolgozó). Két ugyanolyan felhasználónévvel rendelkező dolgozót nem lehet létrehozni.

Az elemek jobb oldalán látható 🔧-gombra kattintva visszatölti a felasználó adatait, és a Hozzáadás gomb szövegét megváltoztatja Módosítás-ra. A felhasználónéven kívül minden adatot lehet módosítani. A gombra kattintva frissíti a dolgozó adatait. A felhasználónéven kívül minden adatot meg lehet változtatni.

A ✖ -gombra kattintva megjelenik egy ablak, amely megkérdezi, hogy biztosan szeretné-e törölni az adott felhasználót. Az igen gombra kattintva törli a felhasználót.

Lista

Törlés gomb

Módosítás gomb

Felhasználó adatai

Lista fejléce

Lista elem

Fejléc

A jobb felső sarokban található Kijelentkezés gombra kattintva bezárja az ablakot, és újra megnyitja a bejelentkezési felületet.

1. Kliens oldali komponens

Ezen a felületen lehet felhasználói fiókot létrehozi, és abba bejelentkezni, új foglalást létrehozni. Foglalni csak akkor lehet, ha már be van jelentkezve a felhasználó.

* 1. Felhasználó fiók létrehozása

Új felhasználói fiók létrehotásához szükséges adatok: felhasználónév, jelszó, telefonszám, e-mail, név.

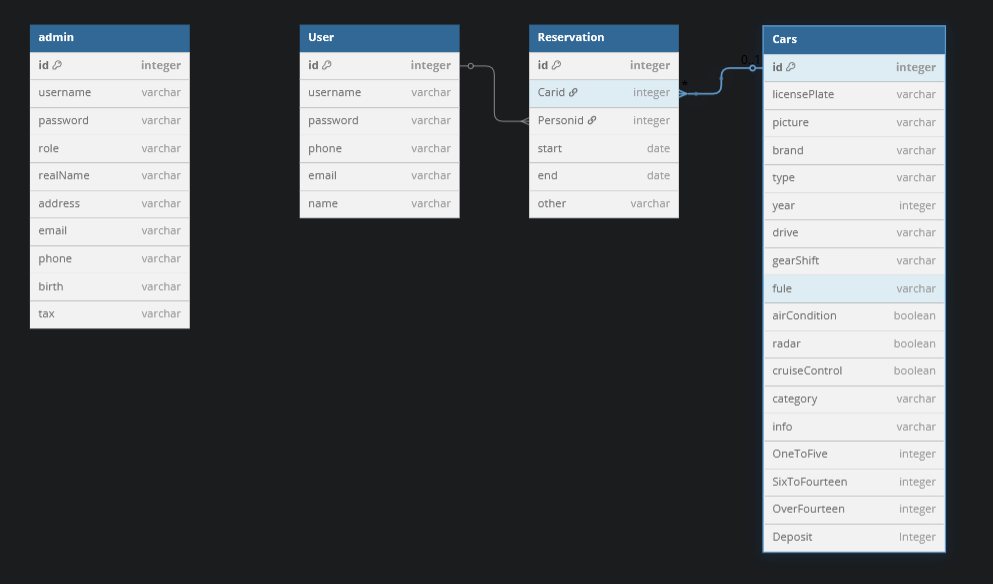
1. Natív mobil alkalmazás

Ezen a felületen lehet felhasználói fiókot létrehozi, és a meglévő felhasználói fiókba bejelentkezni, új foglalást létrehozni. Foglalni csak akkor lehet, ha már be van jelentkezve a felhasználó.

* 1. Felhasználó létrehozása

Új felhasználói fiók létrehotásához szükséges adatok: felhasználónév, jelszó, telefonszám, e-mail, név.

# Adatbázis

A képen látható az adatbázis megjelenése és, a táblák kapcsolódása.

.

Összesen négy táblánk van, és a Reservation tábla az egyetlen, amelyhez adat hozzáadása esetén szükséges más tábla adata is. Az admin táblát kizárólag az asztali alkalmazás felületen használjuk és, csakis arról a felületről érhető el.

# API végpontok

Autók:

1. Autó hozzáadás

* Végpont: /AddCar
* Metódus: POST
* Leírás:
  + Ez a végpont létrehoz egy új autót
* Paraméter:
  + Fénykép, rendszám, márka, modell, évjárat, meghajtás, váltó, üzemanyag, tolatóradar, tempomat, egyéb információ, kategória, 1-5 nap (Ft/nap), 6-14 (Ft/nap), 15- (Ft/nap), kaució, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es (unauthorized) hibaüzenetet küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Ha már létezik ezzel a rendszámmal autó, akkor 409-es hibakódot (conflict) és azt küldi vissza, hogy "Már létezik ezzel a rendszámmal autó!"

1. Autók lekérdezése

* Végpont: /ListCars
* Metódus: GET
* Leírás:
  + Ez a végpont lekérdezi az összes autót
* Hibakezelés:
  + Sikertelen lekérdezés esetén 400-as hibakódot (bad request) küld vissza

1. Autó törlése

* Végpont: /DeleteCar/:id
* Metódus: DELETE
* Leírás:
  + Ez a végpont törli a paraméterkén megadott id-val rendelkező autót
* Paraméter:
  + id: ez alapján találja meg a program, melyik autót törölje, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Ha sikertelen a lekérdezés, 400-as hibakódot (bad request) ad vissza
  + Ha nem találja az autót, "Nem létezik ezzel az id-val autó!" üzenetet és 404-es errort (not found) küld

1. Autó módosítása

* Végpont: /UpdateCar/:id
* Metódus: PUT
* Leírás:
  + Ez a végpont tudja módosítani az autók adatait, kivéve a rendszámot
* Paraméter:
  + id: ez alapján tudja módosítani a program az autó adatait, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Ha nem találja az autót, 404-es errort (not found) és "Nincs ilyen autó!"üzenetet küld vissza
  + Ha sikertelen a lekérdezés, 400-as hibakódot (bad request) ad vissza

Kölcsönzések:

1. Új kölcsönzés

* Végpont: /NewRent
* Metódus: POST
* Leírás:
  + Ezzel a végponttal lehet új kölcsönzést létrehozni
* Paraméter:
  + carId: a kiválasztott autó id-je, personId: a felhasználó id-je, kezdete, vége, egyéb kérés, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza

1. Kölcsönzések lekérdezése

* Végpont: /ListAllRents
* Metódus: GET
* Leírás:
  + Ez a végpont lekérdezi az összes kölcsönzést (id, kezdete, vége, egyébb kérés, autó id, rendszám, márka, model, felhasználó id, felhasználónév, név, email, telefonszám)
* Paraméter:
  + autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza

1. Kölcsönzés törlése

* Végpont: /DeleteRent/:id
* Metódus: DELETE
* Leírás:
  + Ezzel a végponttal lehet törölni a kölcsönzést
* Paraméter:
  + id: ez alapján keresi meg a kölcsönzést, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha nem találja a foglalást, akkor 404-es hibakódot (not found) és "Nem található ilyen foglalás!" üzenetet küld vissza

1. Kölcsönzés módosítása

* Végpont: /DeleteRent/:id
* Metódus: PUT
* Leírás:
  + Ezzel a végponttal lehet a kölcsönzések kezdetét, végét és egyéb kérést módosítani
* Paraméter:
  + id: ez alapján keresi meg a kölcsönzést, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha nem találja a foglalást, akkor 404-es hibakódot (not found) és "Nem található ilyen foglalás!" üzenetet küld vissza

Dolgozók:

1. Regisztráció

* Végpont: /AdminRegistration
* Metódus: POST
* Leírás:
  + Ez a végpont hoz létre új dolgozói felhasználót
* Paraméter:
  + felhasználónév, jelszó, beosztás, név, lakcím, e-mail, telefonszám, születési dátum, adószám, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha már létezik ezzel a felhasználónévvel fiók, akkor 409-es errort (conflict) és 'Már létezik ilyen felhasználó!' üzenetet küld vissza

1. Bejelentkezés

* Végpont: /AdminLogin
* Metódus: POST
* Leírás:
  + Ezzel a végponttal lehet bejelentkezni a dolgozói felületre
* Paraméter:
  + felhasználónév, jelszó
* Hibakezelés:
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha rossz a felhasználónév vagy jelszó, akkor 409-es error kódot (conflict) és "Hibás felhasználónév vagy jelszó" üzenetet küld vissza

1. Dolgozók kilistázása

* Végpont: /ListAllWorkers
* Metódus: GET
* Leírás:
  + Ezzel a végponttal lehet kilistázni a dolgozókat
* Paraméter:
  + autentikáció
* Hibakezelés:
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza

1. Dolgozó törlése

* Végpont: /DeleteWorker/:id
* Metódus: DELETE
* Leírás:
  + Ezzel a végponttal lehet törlölni egy dolgozó felhasználóját
* Paraméter:
  + id: ez alapján keresi meg a dolgozót, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha nem találja a dolgozót, akkor 404-es error kódot (not found) és "Nincs ilyen dolgozó!" üzenetet küld vissza

1. Dolgozó módosítása

* Végpont: /UpdateWorker/:id
* Metódus: PUT
* Leírás:
  + Ezzel a végponttal lehet a dolgozók adatait módosítani
* Paraméter:
  + id: ez alapján keresi meg a dolgozót, autentikáció
* Hibakezelés:
  + Ha nincs token, akkor egy üzenetet "Nem létező token!" és 401-es hibaüzenetet (unauthorized) küld vissza
  + Ha rossz a token, akkor "Hibás token!" üzenetet és 498-es hibakódot (invalid token) küld vissza
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha nem találja a dolgozót, akkor 404-es error kódot (not found) és "Nincs ilyen dolgozó!" üzenetet küld vissza
  + Ha már létezik ilyen felhasználónév, akkor 409-es errorkódot (conflict) "Már létezik ilyen felhasználó!" üzenettel ad vissza

Felhasználók:

1. Regisztráció:

* Végpont: /UserRegistration
* Metódus: POST
* Leírás:
  + Ez a végpont hoz létre új felhasználói felhasználót
* Paraméter:
  + felhasználónév, jelszó, név, e-mail, telefonszám
* Hibakezelés:
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha már létezik ezzel a felhasználónévvel fiók, akkor 409-es errort (conflict) és 'Már létezik ilyen felhasználó!' üzenetet küld vissza

1. Bejelentkezés:

* Végpont: /UserLogin
* Metódus: POST
* Leírás:
  + Ezzel a végponttal lehet bejelentkezni a felhasználói felületre
* Paraméter:
  + felhasználónév, jelszó
* Hibakezelés:
  + Sikertelen lekérdezésnél 400-as error kódot (bad request) küld vissza
  + Ha rossz a felhasználónév vagy jelszó, akkor 409-es error kódot (conflict) és "Hibás felhasználónév vagy jelszó" üzenetet küld vissza

# Tesztelés

1. Natív asztali alkalmazás

|  |  |  |
| --- | --- | --- |
| Tesztelt funkció | Elvárt eredmény | Tesztelés |
| Bejelentkezés | A jelszó megjelenítése gomb lenyomásával megjelenik a jelszó; hibás felhasználónév, vagy jelszó esetén nem enged be a rendszerbe. | [tests\login.wmv](tests/login.wmv) |
| Autó hozzáadás | Üres autót nem lehet hozzáadni, két azonos rendszámmal rendelkező autó nem létezhet | [tests\createCar.wmv](tests/createCar.wmv) |
| Autó törlése | A ✖ gombra kattintva megkérdezi az alkalmazás, hogy biztosan törölni akarja-e az autót, és csak igenre kattintva törli. | [tests\delete.mp4](tests/delete.mp4) |
| Autó módosítása | A 🔧 gombra kattintva sikeresen visszatöltődnek az autó adatai, és változtatásnál a listában az új adatok láthatók. | [tests\updateCar.mp4](tests/updateCar.mp4) |
| Kölcsönzés módosítása | A 🔧 gombra kattintva sikeresen visszatöltődnek a kölcsönzés adatai, és nem lehet ugyanarra az időpontra ugyanazzal az autóval módosítani a kölcsönzést. | [tests\updateRent.mp4](tests/updateRent.mp4) |
| Kölcsönzés törlése | A ✖ gombra kattintva megkérdezi az alkalmazás, hogy biztosan törölni akarja-e a kölcsönzést, és csak igenre kattintva törli. | [tests\deleteRent.mp4](tests/deleteRent.mp4) |
| Dolgozó hozzáadása | Ha bármelyik mező üres, akkor az alkalmazás jelzi, hogy minden mezőt ki kell tölteni, a két jelszónak meg kell egyeznie. | [tests\createWorker.mp4](tests/createWorker.mp4) |
| Dolgozó módosítása | A 🔧 gombra kattintva sikeresen visszatöltődnek a kölcsönzés adatai, és nem lehet a felhasználónevet változtatni. | [tests\updateWorker.mp4](tests/updateWorker.mp4) |
| Dolgozó törlése | A ✖ gombra kattintva megkérdezi az alkalmazás, hogy biztosan törölni akarja-e a dolgozót, és csak igenre kattintva törli. | [tests\deleteWorker.mp4](tests/deleteWorker.mp4) |

# Források

* Adatbázis tervezés: <https://dbdiagram.io>
* Felület tervezés: <https://www.figma.com>
* Feladatok kiosztása: <https://trello.com>
* GitHub repository: <https://github.com/Tsikkk/13-projekt>