

SZENT ISTVÁN KATOLIKUS  
TECHNIKUM ÉS GIMNÁZIUM

**balogh dávid**

**füredel valentin szilveszter**

**tarnóczi levente tibor**

**gépjárműkölcsönzés**

**SZOFTVERFEJLESZTÉS ÉS -TESZTELÉS VIZSGAREMEK**

Sátoraljaújhely, 2022

**TARTALOMJEGYZÉK**

Tartalom

[Bevezetés 4](#_Toc125627264)

[Felhasználói Dokumentáció 5](#_Toc125627265)

[Rendszerkövetelmény 5](#_Toc125627266)

[Hardver követelmények: 5](#_Toc125627267)

[Szoftver követelmények 5](#_Toc125627268)

[Telepítés menete 6](#_Toc125627269)

[Adatbázis hozzáadása 6](#_Toc125627270)

[Weboldal telepítés menete: 7](#_Toc125627271)

[Java alkalmazás telepítése: 7](#_Toc125627272)

[Szoftver használata 9](#_Toc125627273)

[Web alkalmazás használata 9](#_Toc125627274)

[Java alkalmazás használata 15](#_Toc125627275)

[Fejlesztői dokumentáció 23](#_Toc125627276)

[Az alkalmazott fejlesztői eszközök 23](#_Toc125627277)

[Trello projektkezelő 23](#_Toc125627278)

[Git verziókezelő 23](#_Toc125627279)

[XAMPP 24](#_Toc125627280)

[Microsoft Visual Studio Code 24](#_Toc125627281)

[IntelliJ IDEA 25](#_Toc125627282)

[Microsoft Office Word 25](#_Toc125627283)

[Adatmodell leírása 26](#_Toc125627284)

[user tábla 26](#_Toc125627285)

[vehicles tábla 26](#_Toc125627286)

[booking tábla 27](#_Toc125627287)

[message tábla 27](#_Toc125627288)

[Táblák közötti kapcsolatok 28](#_Toc125627289)

[Részletes specifikáció 29](#_Toc125627290)

[Weboldal részletes specifikációi 29](#_Toc125627291)

[Java alkalmazás részletes specifikációi 33](#_Toc125627292)

[Tesztelési dokumentáció 40](#_Toc125627293)

[Java alkalmazás tesztelése 40](#_Toc125627294)

# Bevezetés

Azért választottuk ezt a szakmát, mert régóta érdeklődünk szoftverfejlesztéssel kapcsolatos témák iránt. Vannak olyan kihívások, amik felkeltik a figyelmünket például összetettebb rendszerek készítése és működése. Később szeretnénk ebben a szakmában elhelyezkedni, tevékenykedni vagy továbbtanulni.

Az utóbbi években kis városunkban megnőtt a turizmus és ennek köszönhetően megnőtt a gépjármű bérlésre az igény.

A mi projektünk célja, hogy a hétköznapi emberek számára könnyebbé és egyszerűbbé tegye a gépjármű bérlés lehetőségét, a cégek számára könnyebbé és átláthatóbbá tegye a gépjárművek nyilvántartását és a bérlések kezelését.

Azért választottuk ezt a témát mivel érdeklődünk a személy-gépkocsik iránt. Korábbi autóbérlésünk során találkoztunk néhány nehézséggel, így szeretnénk megkönnyíteni a városba érkező turisták vagy helyiek, vagy környékbeliek közlekedési problémáit, egyúttal egy vállalkozás munkáját, ami bérbeadással foglalkozik.

A projekt elkészítése során a következő programozási nyelveket használtuk: Java, PHP, MySQL, HTML, CSS, JavaScript.

# Felhasználói Dokumentáció

## Rendszerkövetelmény

### Hardver követelmények:

Minimális rendszerkövetelmény:

**Operációs rendszer:** Windows 10/Linux/MacOS

**Processzor:** 1 GHz vagy gyorsabb

**RAM:** 1 GB

**Merevlemez**-**terület:** 1 GB

**Videókártya:** A DirectX 9-es vagy újabb verziójával kompatibilis.

**Kijelző:** 800x600

Ajánlott rendszerkövetelmény:

**Operációs rendszer:** Windows 10/Linux/MacOS

**Processzor:** 2 GHz vagy gyorsabb

**RAM:** 2 GB

**Merevlemez-terület:** 1 GB

**Videókártya:** A DirectX 9-es vagy újabb verziójával kompatibilis.

**Kijelző:** 1280x1024

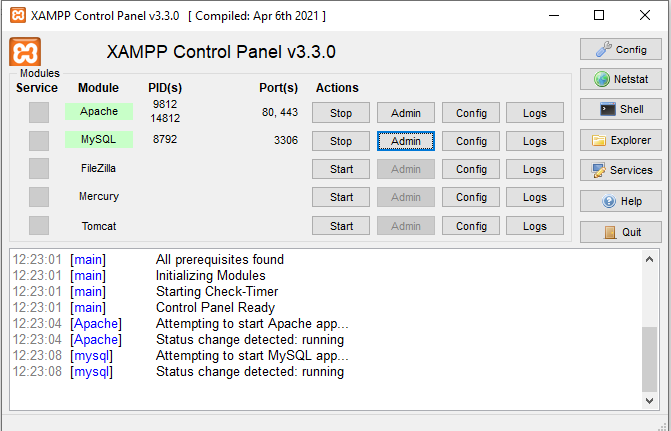
### Szoftver követelmények

* XAMPP v3.3.0
* PHP v8.1.6
* Java Runtime Enviroment(JDK) 11.0.14

## Telepítés menete

### Adatbázis hozzáadása

Indítsa el a xampp-ot, ha nincs feltelepítve a számítógépre, akkor telepítse a v3.3.0 verzió számú vagy újabb programot, amely 8.1.6 vagy újabb php-val rendelkezik. A telepítésének folyamata a telepitő állomány megnyitásával kezdődik, majd ha végzett a telepítés indítsa el az Apache és a MySQL szervert. Ez az adatbázis FONTOS a weboldal és a java alkalmazás futtatásához is!



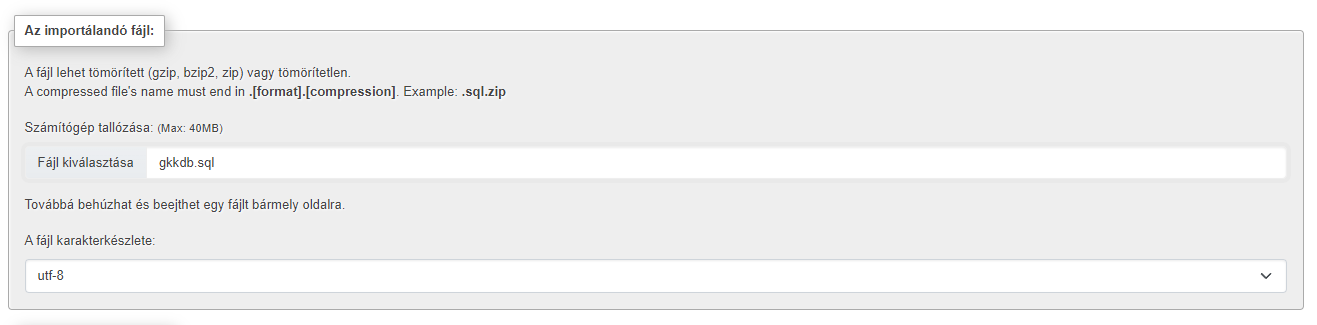
‑. XAMPP Control Panel v3.3.0

Nyisson meg egy böngészőt (pl.: Google Chrome) és lépjen be a phpMyadmin szerverére a <http://localhost/phpmyadmin/> címmel. Hozza létre a következő adatbázist ’***gkkdb***’ néven, a karakter kódolása utf8\_hungarian\_ci legyen.



‑. Adatbázis létrehozása

Nyissa meg a létrehozott adatbázist (gkkdb) és kattintson az ***import*** fülre. Kattintson a fájl kiválasztására, majd válassza ki a *GKKAPP/adatbázis* mappából a *gkkdb.sql* fájlt és kattintson az **indítás** gombra.



‑. Adatbázis importálása

### Weboldal telepítés menete:

A weboldal feltelepítése előtt keresse meg a *xampp/htdocs* nevű mappát, majd törölje annak minden tartalmát, ha ez megtörtént helyezze a *GKK WEB* mappa tartalmát a törölt fájlok helyére. Írja be a keresőmezőbe a *localhost* címet és élvezze weboldalunk nyújtotta szolgáltatásokat.

### Java alkalmazás telepítése:

#### Windows indítás:

Töltse le a git reposytort (<https://github.com/Florex001/GKKApp>). Majd csomagolja ki egy tetszőleges helyre. Indítsa el a „***GKKApp\_win.exe***” fájlt és próbáljon meg bejelentkezni. Ellenőrizze, hogy elérhető-e az adatbázis, ha nem akkor hajtsa végre az „Adatbázis hozzáadása” című fejezetben leírtak szerint az adatbázis létrehozását. Ha nincs, létrehozva az adatbázis ezt a hibaüzenetet fogja látni:



0‑4. Hiba üzenet Java

#### Arch Linux indítása:

Töltse le a git reposytort(<https://github.com/Florex001/GKKApp>). Majd csomagolja ki egy tetszőleges helyre. Indítsa el a „***Gkkapp\_linux.jar***” fájlt és próbáljon meg bejelentkezni. Ellenőrizze, hogy elérhető-e az adatbázis, ha nem akkor hajtsa végre az „Adatbázis hozzáadása” című fejezetben leírtak szerint az adatbázis létrehozását. Ha nincs, létrehozva az adatbázis ezt a hibaüzenetet fogja látni:



‑. Hiba üzenet Java

#### Debian Linux indítása:

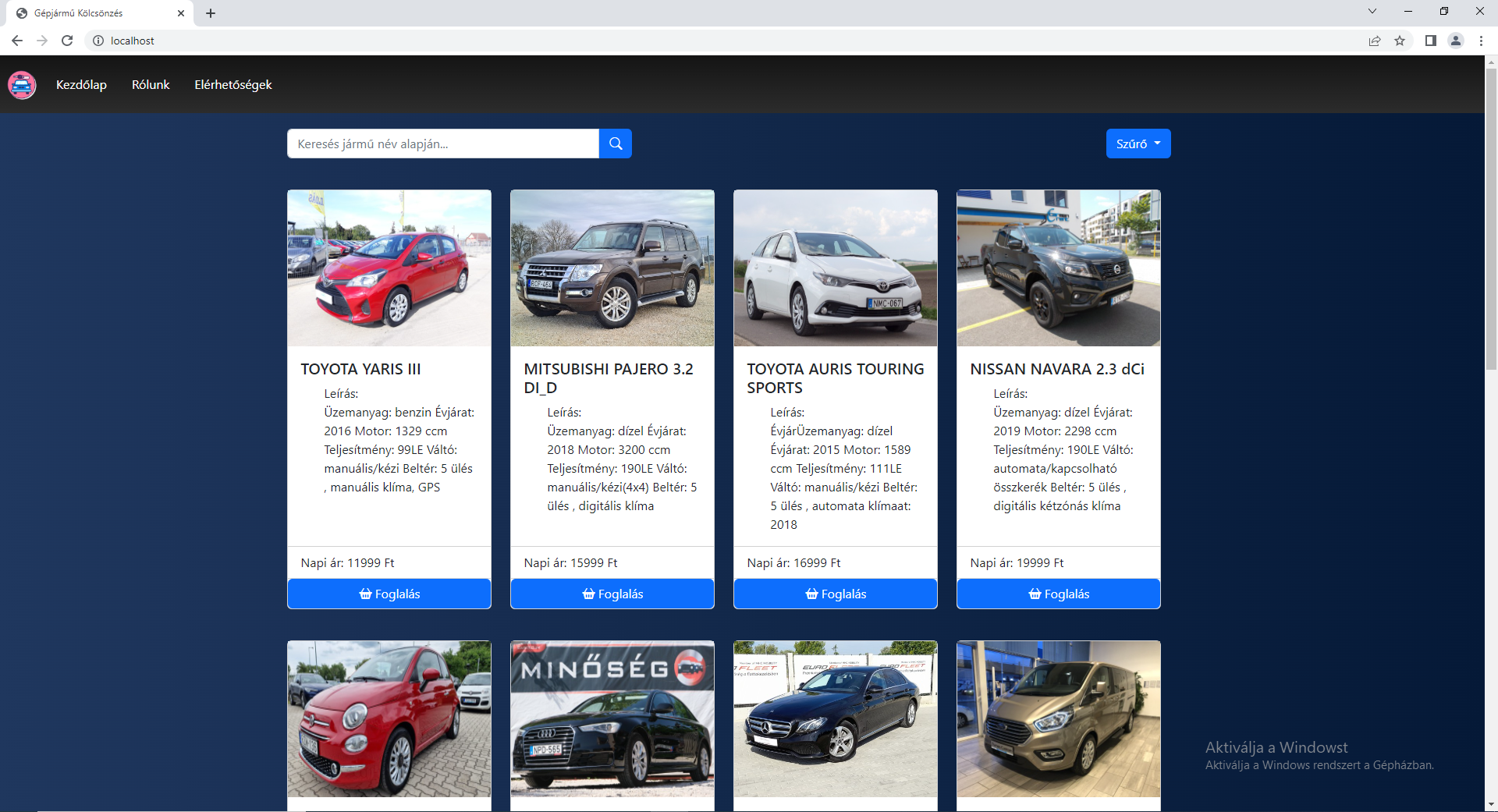
Töltse le a git reposytort(<https://github.com/Florex001/GKKApp>). Majd csomagolja ki egy tetszőleges helyre. Majd a projekt mappájában nyisson meg egy terminált. A terminálban írja be a következő parancsot: „*java –jar Gkkapp\_linux.jar”*. Nyomjon egy entert és már el is indul a program, Próbáljon meg bejelentkezni. Ellenőrizze, hogy elérhető-e az adatbázis, ha nem akkor hajtsa végre az „Adatbázis hozzáadása” című fejezetben leírtak szerint az adatbázis létrehozását. Ha nincs, létrehozva az adatbázis ezt a hibaüzenetet fogja látni:



. Hiba üzenet Java

## Szoftver használata

### Web alkalmazás használata



‑. Kezdőlap

#### Általános információk

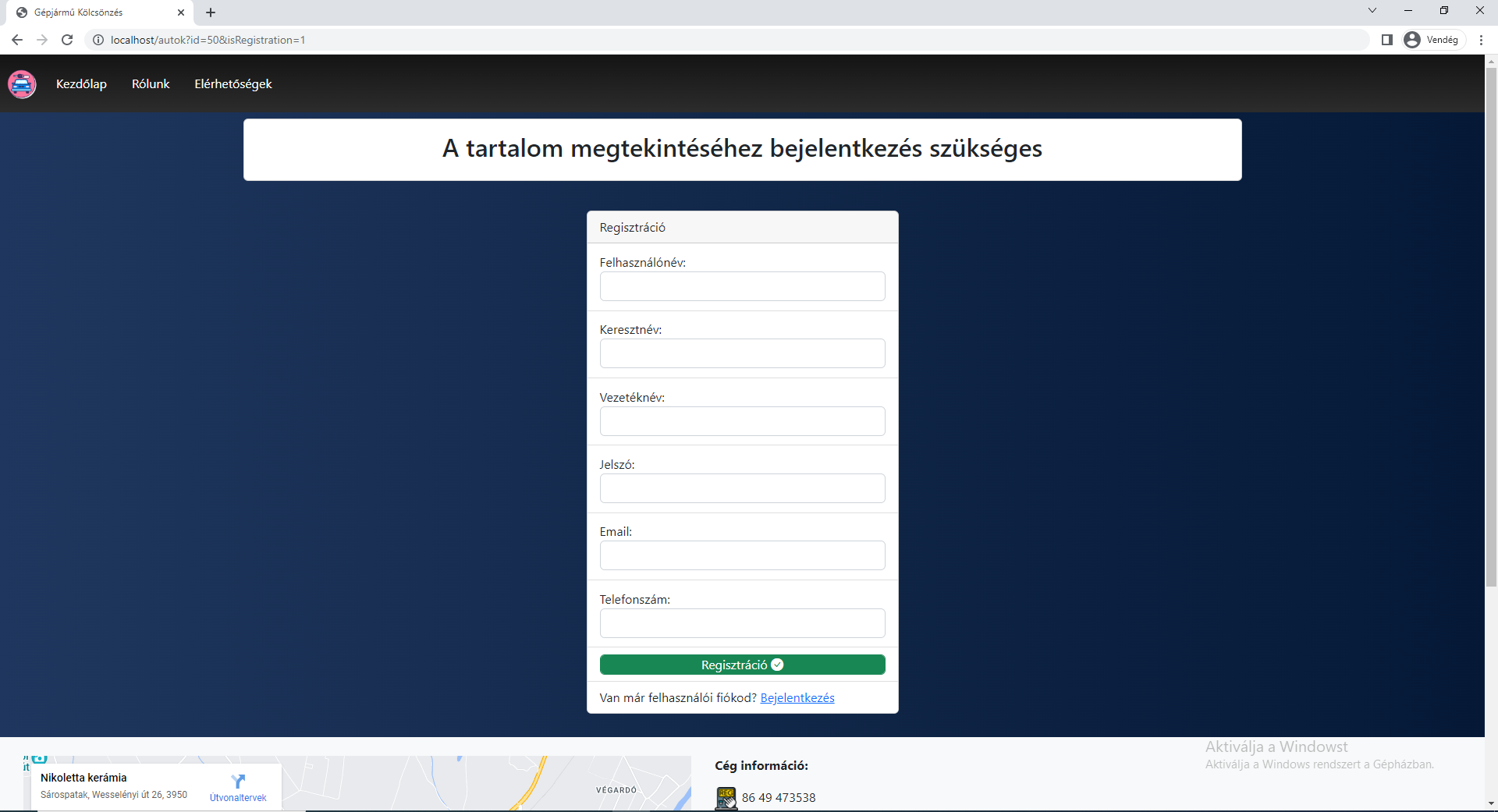
A weboldalunk olyan szolgáltatást nyújt, amelyen keresztül személygépjárműveket lehet bérelni. Az oldal fejlesztéséhez Google Chrome böngészőt használtunk, de más böngészőkkel is működik. A gépjárművekhez általános leírás is tartozik valamint a bérlés napi díjáról is kap tájékoztatás a felhasználó. A gépjárművek között lehet keresni, erre szolgál a weboldalba beépített keresőmotor. Ezen felül a felhasználó tudja rendezni a gépjárműveket *„Ár szerint növekvő”* és *„Ár szerint csökkenő”* sorrendbe a szűrő legördülő menü segítségével. A bérlés megkezdéséhez felhasználói fiókra van szükség.

#### Gépjárművek elrendezése

A gépjárművek kártyákban vannak elhelyezve a kezdőlapon. Ezek elrendezése négyzetrács szerűen van megoldva. A kártyák tartalmaznak egy képet a gépjárműről, egy címet, ami a gépjármű neve, egy leírást a gépjárműről és a gépjármű napi árát. Ezek mellett megtalálható egy foglalás nevű gomb is, amely átnavigálja a felhasználót a gépjármű foglalás oldalra, amelyhez bejelentkezés szükséges.

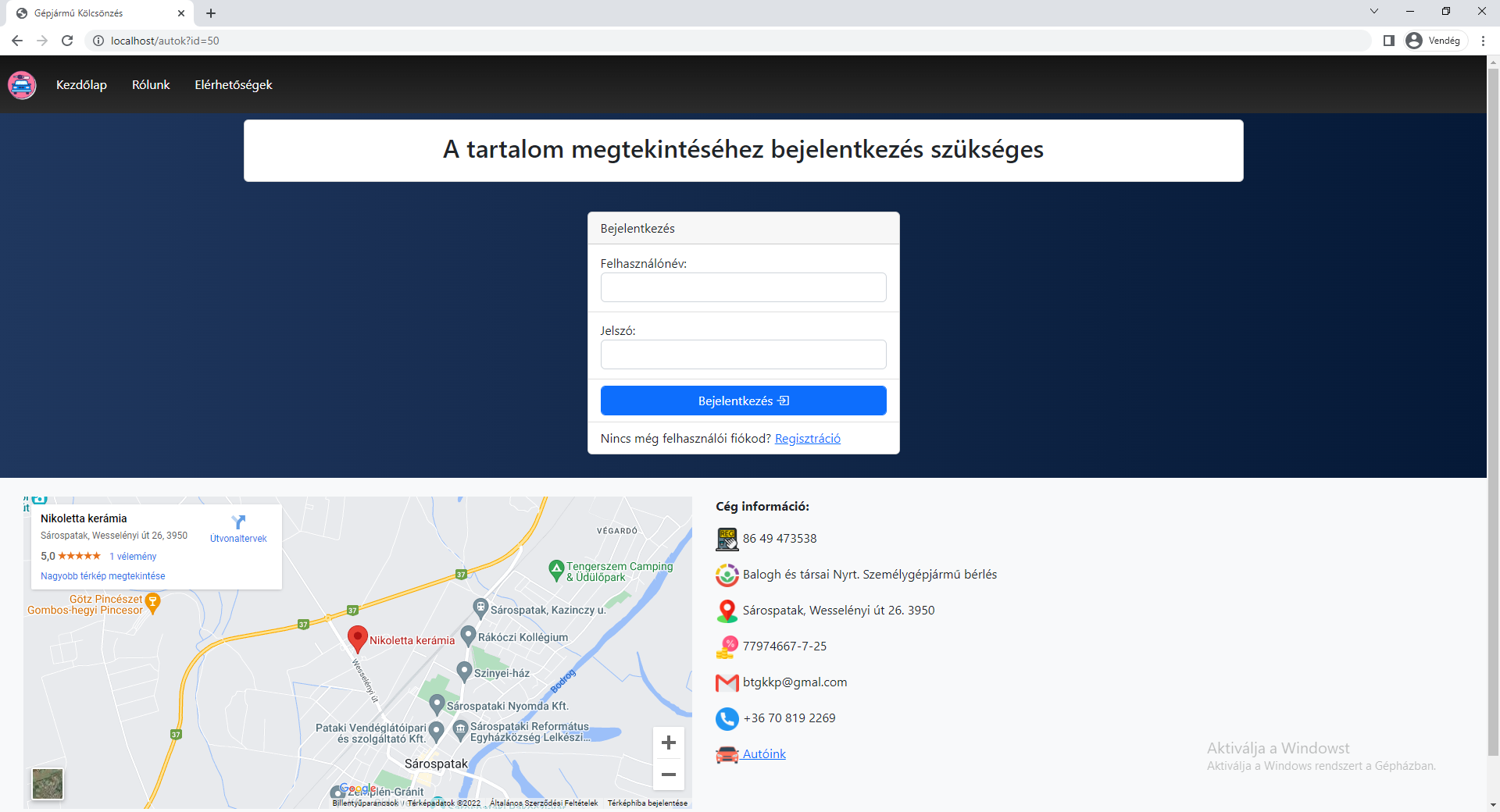
‑. Gépjármű kártya

#### Regisztráció és Belépés



‑. Regisztráció

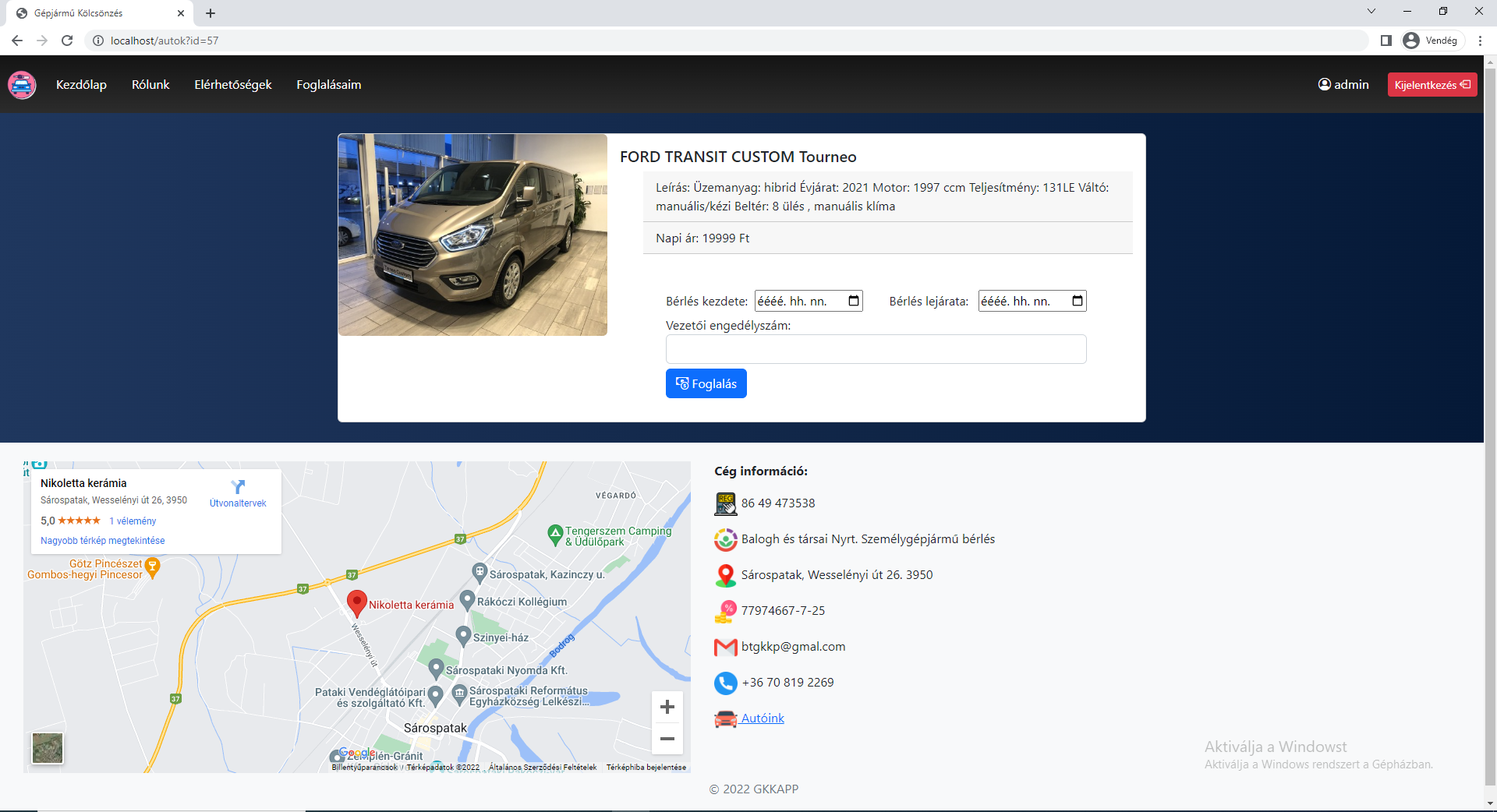
Ahhoz hogy a felhasználó regisztrálni tudjon, ki kell választania az egyik gépjárművet majd a foglalás gombra kattintani (Regisztráció vagy Bejelentkezés esetén nem kerül foglalásra az adott gépjármű!). Majd át navigálja a weboldal a felhasználót hogy *„A tartalom megtekintéséhez bejelentkezés szükséges”*. Ha rendelkezik felhasználói fiókkal jelentkezzen be. Ha nincs még felhasználói fiókja, akkor kattintson a *„Nincs még felhasználói fiókod? Regisztráció”* Regisztráció linkre. Ha rákattintott, akkor az oldal automatikusan átnavigálja a regisztráció oldalra, ahol minden adat megadása kötelező.



‑. Bejelentkezés

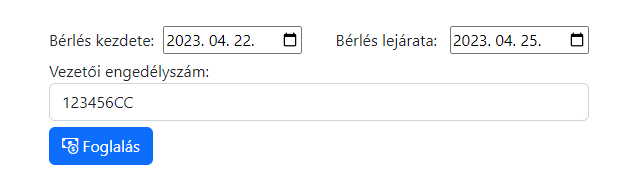
A bejelentkezésnél Felhasználónév és Jelszó megadása kötelező. Ha sikeres a bejelentkezés egy sütit fogunk eltárolni a böngészőjében. Ha sikeres a bejelentkezés, akkor a kiválasztott gépjárműnek a foglalás oldala fog megjelenni. Illetve a navigációs sávban megjelenik a kijelentkezés gomb, amire ha rá kattintunk, akkor törlődik az eltárolt süti és új bejelentkezésre lesz szüksége és még megjelenik a foglalásaim fül is ahol a felhasználó meg tudja tekinteni a gépjármű foglalásait. Profil oldal is elérhetővé vállik onnan ismerhető fel, hogy a felhasználónév lesz a neve annak a menünek.

#### Gépjármű foglalása

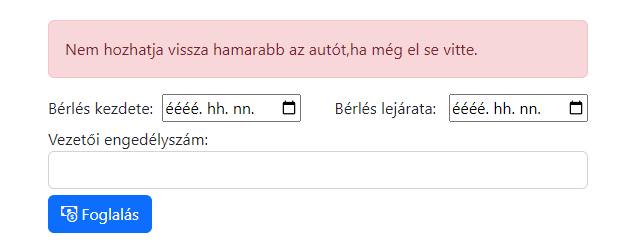


‑. Gépjármű foglalás

Ha sikeresen bejelentkezett, akkor foglalás oldal fog megjelenni a kiválasztott autójával. Ha biztos benne hogy azt az autót szeretné lefoglalni, akkor töltse ki az alábbi mezőket valós adatokkal:



‑. Adatok megadása foglaláshoz

Ha biztos benne hogy arra az időpontra szeretné kibérelni az adott gépjárművet és a vezetői engedélyszáma valós, akkor kattintson a ***Foglalás*** gombra. Ha sikeres a foglalás, akkor át navigálja a ***foglalásaim*** oldalra. Ha nem megfelelő a foglalás, akkor hiba üzenetet fog kapni: Például.:

13. Foglalási hiba üzenet

#### Foglalásaim oldal



‑. Foglalásaim menüpont

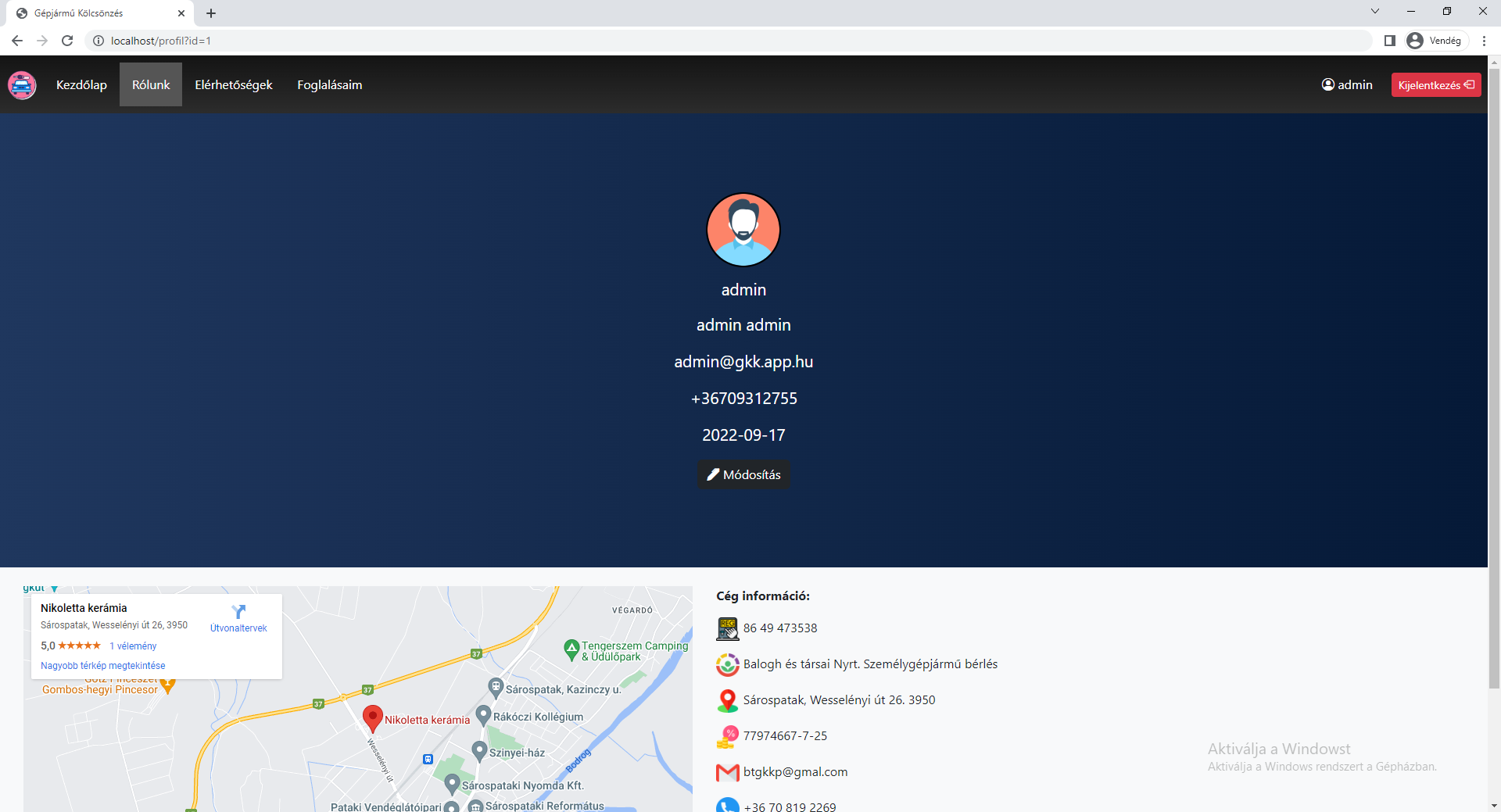
A foglalásaim menüpontra kattintva, vagy ha éppen foglalt egy gépjárművet és átnavigálásra került, akkor a saját foglalásait fogja látni. A foglalását csak a gépjármű átadásának idejéig fogja látni. Ha már a gépjárművet használja, amit bérel, abban az esetben már nem fogja látni foglalását. Fizetés gépjármű átadása előtt történik a telephelyen készpénzzel vagy bankkártyával. Amennyiben módosítani akarja foglalását akkor a navigációs menüben kattintson az ***Elérhetőségek*** menüpontra. Azt fogja tapasztalni, hogy az oldal aljára, fogja navigálni ahol, megjelennek a céggel kapcsolatos elérhetőségek. Módosításhoz hívja fel az ott található telefonszámot vagy írjon emailt. Foglalás lemondására az oldalon van lehetőség.



‑.Foglalás megjelenése

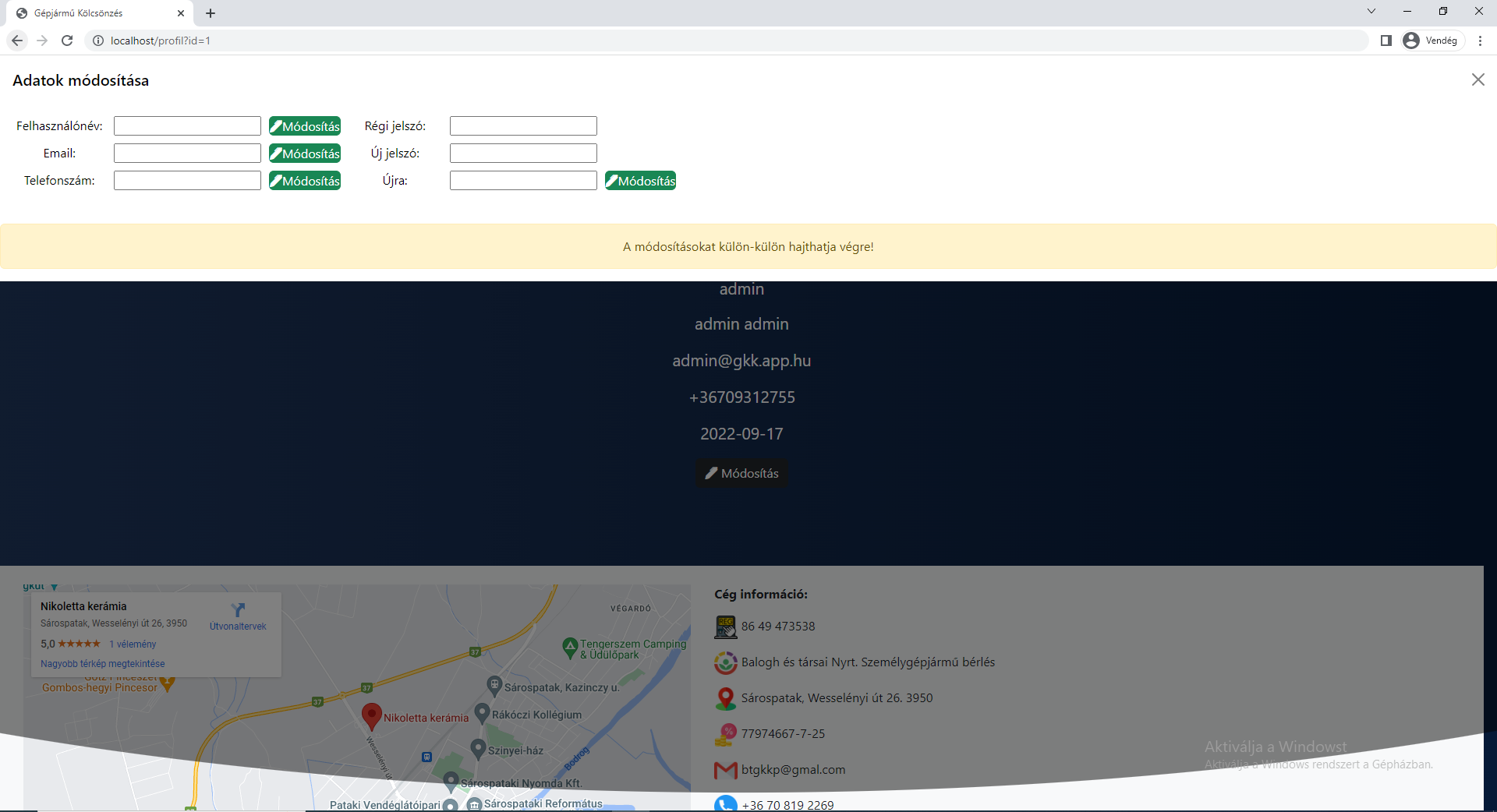
A saját foglalása a *0-15.Foglalás megjelenése* ábrának megfelelően fog megjelenni. Ebben látni fogja a foglalás azonosítóját, amit jegyezzen meg, mert fontos a gépjármű bérlése során. Ezen kívül megjelenik még a gépjármű neve, Foglalás kezdete, Foglalás vége illetve a teljes ár. A teljes ár a két megadott dátum közötti napok száma szorozva napi árral. A ***Foglalás lemondása*** gomb is itt jelenik meg, ha le akarja mondani a foglalását nincs más dolga csak rá kattintani.

#### Profil oldal



‑. Profil oldal

A profil oldalon megjelenik egy profilkép, a felhasználónév, a felhasználó vezetékneve és keresztneve, az e-mail címe illetve a regisztráció időpontja. Láthatunk egy módosítás gombot is amely arra a célra szolgál, hogy ha módosítani szeretnénk az adatokat, akkor erre ad lehetőséget.



‑. Adatok Módosítása

Az adatok csak külön-külön változtathatók meg. A felhasználónév, az e-mail cím, a telefonszám és a jelszó módosítható. A jelszót csak a régi jelszó tudatában tudja módosítani és az új jelszónak és az újra mezőnek egyeznie kell. Ha a vezetékneve és keresztneve változott vegye fel velünk a kapcsolatot e-mail címen.

### Java alkalmazás használata

#### Általános Információk

A Java alkalmazásban a dolgozók a foglalásokkal és a gépjárművekkel kapcsolatos információkat tudják kezelni. Ha a felhasználó jogosultsága admin akkor megtekinthető számára az admin felület is. Ezen felül a dolgozó és az admin számára is megtekinthető az céginformációk, üzenet küldés és a saját profiljának adatai.

#### Bejelentkezés

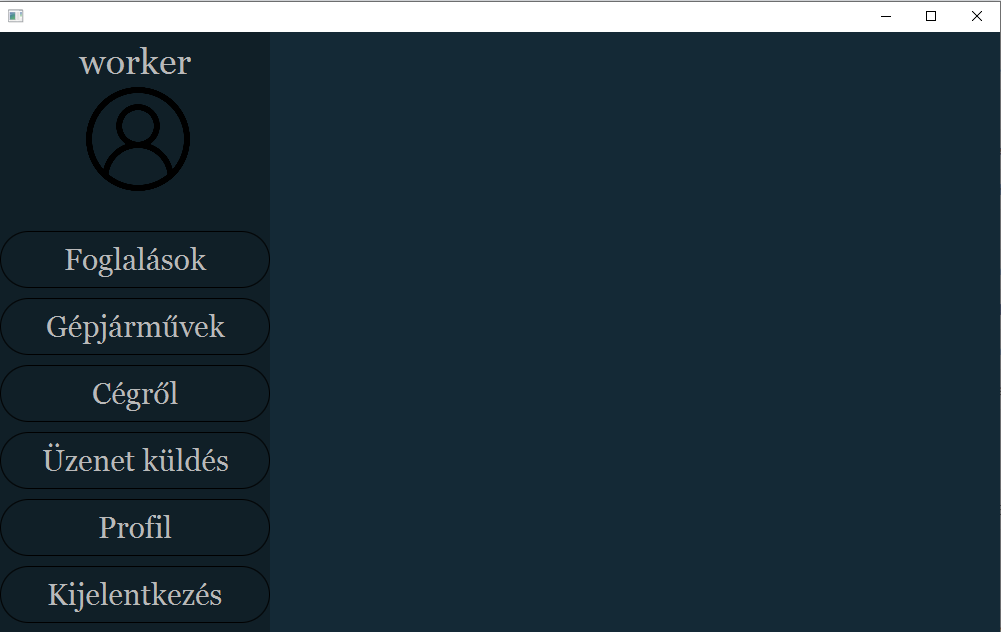
A bejelentkezéshez már egy létező felhasználó szükséges aminek *„worker”* vagy *„admin”* jogosultsága van. Írja be a felhasználónevét és jelszavát majd kattintson a **Bejelentkezés** gombra. Ha nincs megfelelő jogosultsága kérjen meg egy dolgozót, hogy vegye fel a kapcsolatot az adminisztrátorral.

****

. Bejelentkezés

#### Főoldal

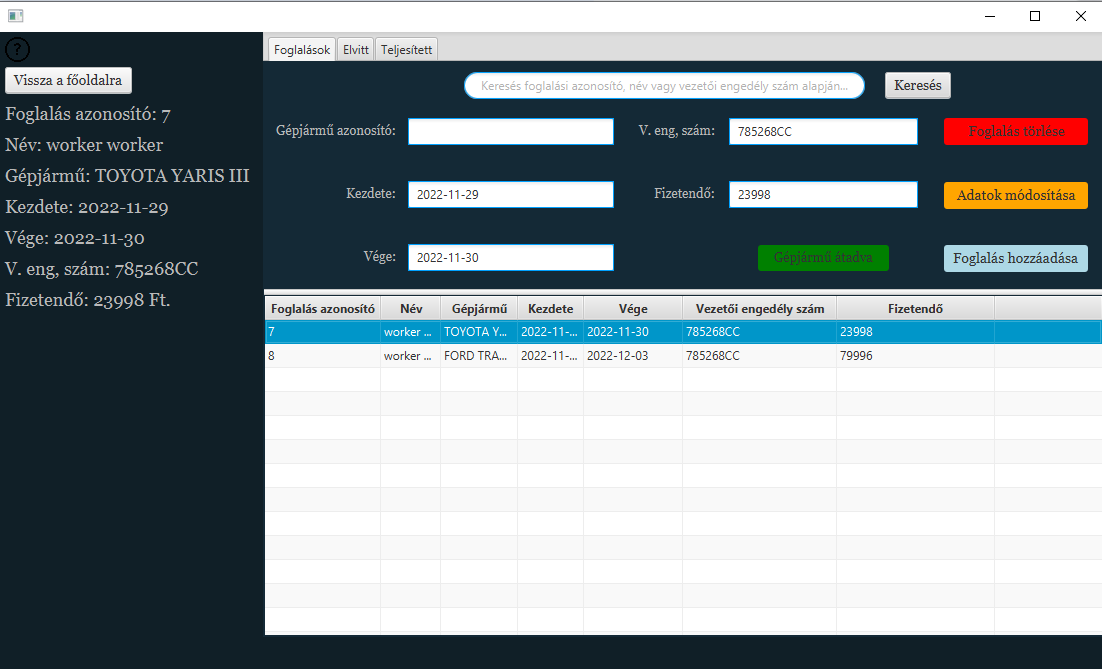
A főoldalon megtalálható a felhasználónév, **Foglalások**, **Gépjárművek**, **Cégről**, **Üzenet küldés**, **Profil** és egy **Kijelentkezés** gomb.



. Főoldal

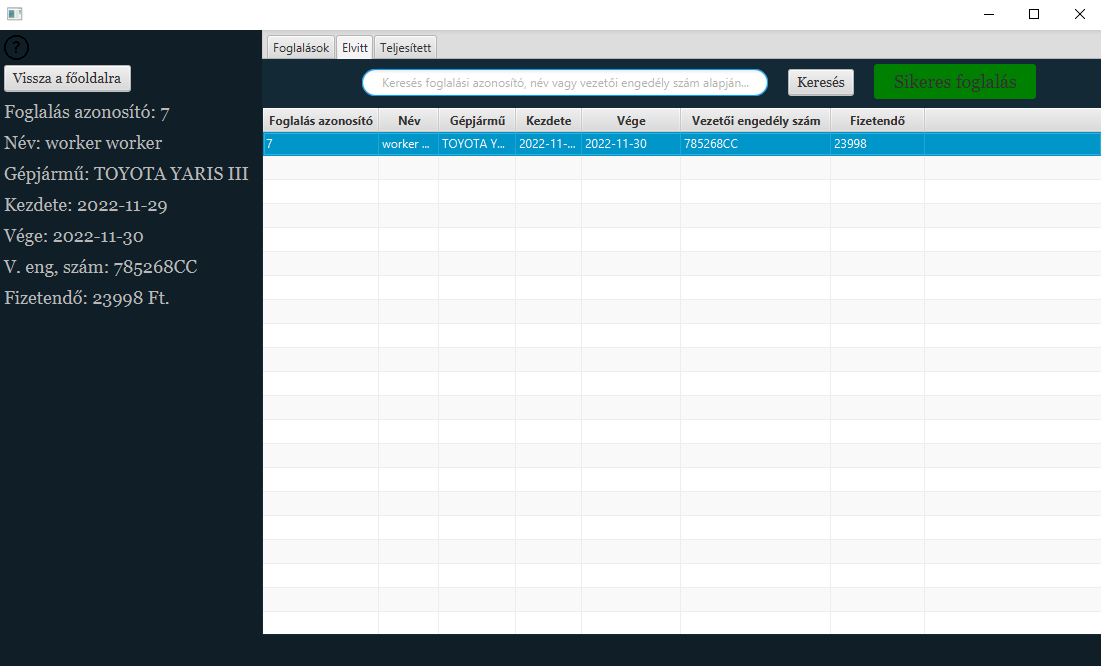
#### Foglalások

A foglalások menüpontban megtalálható az összes foglalás, a foglalásokról az összes adat, egy **Vissza a főoldalra** egy gomb. Ezek között megfigyelhető egy táblázat is ahol megjelenik az összes foglalás. Illetve a foglalások között tud keresni is, továbbá tud foglalást törölni, módosítani és hozzáadni. Tudja a kiválasztott foglalás státuszát is módosítani a **Gépjármű átadva** gombbal.



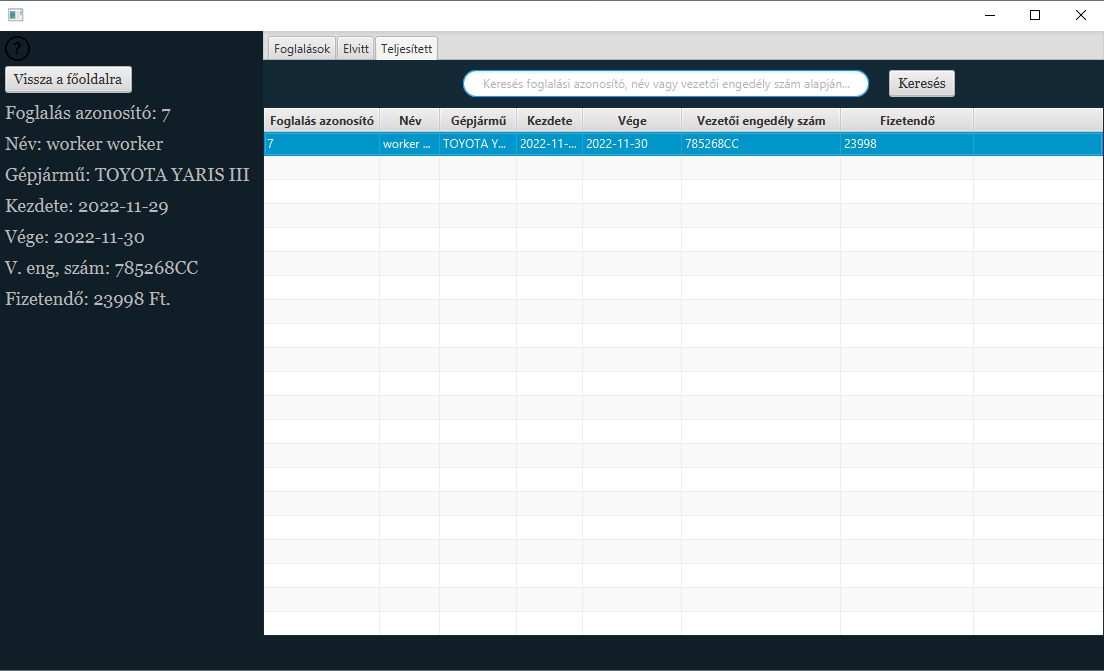
. Foglalások oldal

A státuszát akkor kell módosítani, hogyha átadásra került a bérlő számára a gépjármű. Ezek után az elvitt menüpontba át fog kerülni a kiválasztott foglalás. A **Sikeres foglalás** gombra akkor kattintson, ha az adott gépjármű foglalása végbe ment.



. Elvitt menüpont

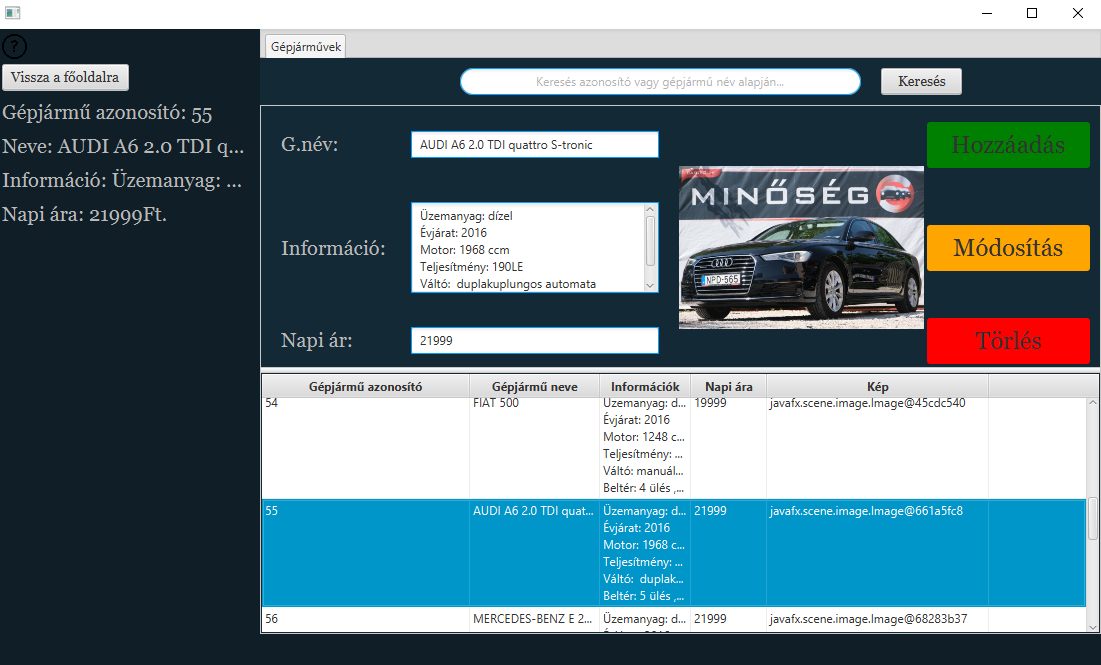
A Teljesített menüponton megtalálható az összes végbement foglalás és annak összes adata.



. Teljesített menüpont

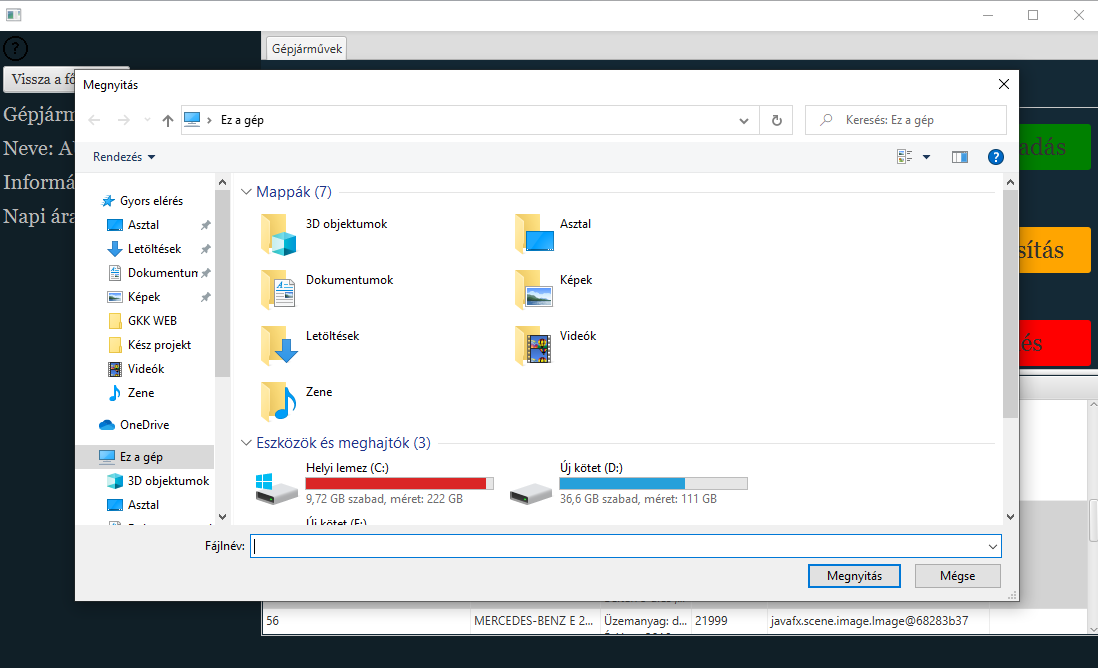
#### Gépjárművek

A gépjárművek oldalon megtalálható egy táblázatban az összes gépjármű amelyek között lehet keresni és a kiválasztott gépjárművet lehet módosítani vagy törölni. Módosításnál csak a gépjármű nevét, információját, és napi árát lehet módosítani.



. Gépjárművek oldal

Ahhoz hogy gépjárművet tudjon hozzáadni, írja be először a gépjármű nevét, információját és a napi árát a beviteli mezőkbe majd kattintson a **Hozzáadás** gombra. Ezután egy fájlkezelő ablak fog megnyílni ahol kiválaszthatja, a gépjárműről a képet majd kattintson a **Megnyitás** gomra. Fog kapni egy felugró üzenet, ha sikeres a hozzáadás.



. Fájlkezelő

#### Cégről

A **Cégről** gombra kattintva egy felugró ablak jelenik meg, amelyben megtalálható a cégről az összes információ, mint például Regisztrációs szám, Cégnév, Székhely és Adószám.



. Céginformációk

#### Üzenet küldés

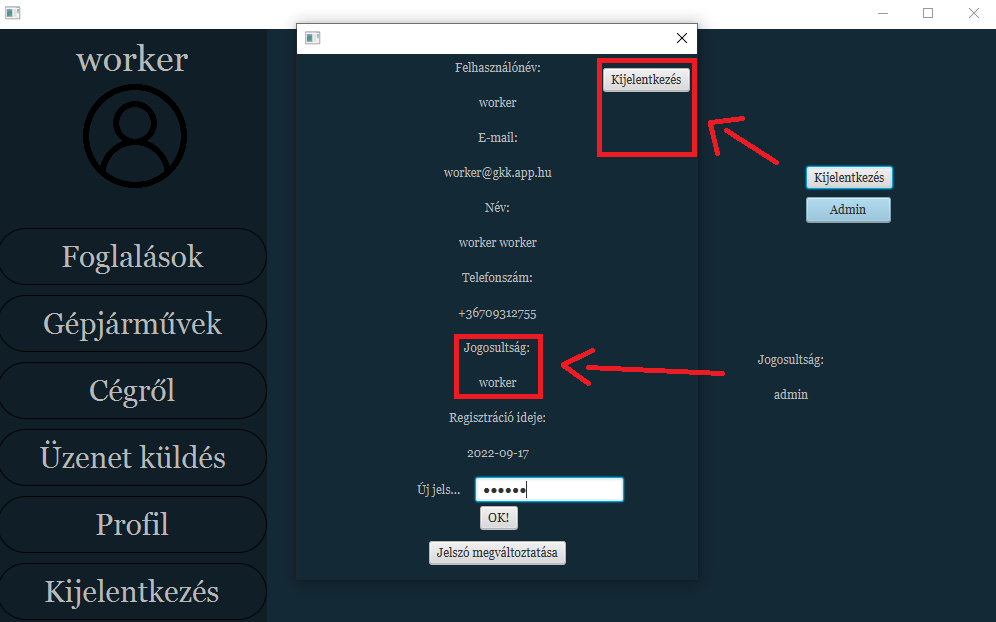
Az üzenet küldés oldalon írhat üzenetet az adminisztrátornak, hogyha valamelyik felhasználónak jogosultságot kell adni vagy elvenni illetve különböző profilmódosításokat lehet kérvényezni. Ha megírta az üzenetét akkor kattintson a **Küldés** gomra. Ezeket az adminisztrátor el tudja fogadni vagy el tudja utasítani az üzentet. Az elfogadásról vagy az elutasításról nem fog értesítést kapni.



. Üzenet küldés ablak

#### Profil

A profil oldalon meg tudja tekinteni az összes felhasználói adatát, mint például felhasználónév, e-mail cím, telefonszám, nevét, regisztráció dátumát illetve jogosultságát. Ha rákattint a **Jelszó megváltoztatása** gombra megjelenik egy beviteli mező, amelybe beírhatja új jelszavát, ha beírásra került kattintson az **OK!** gombra.

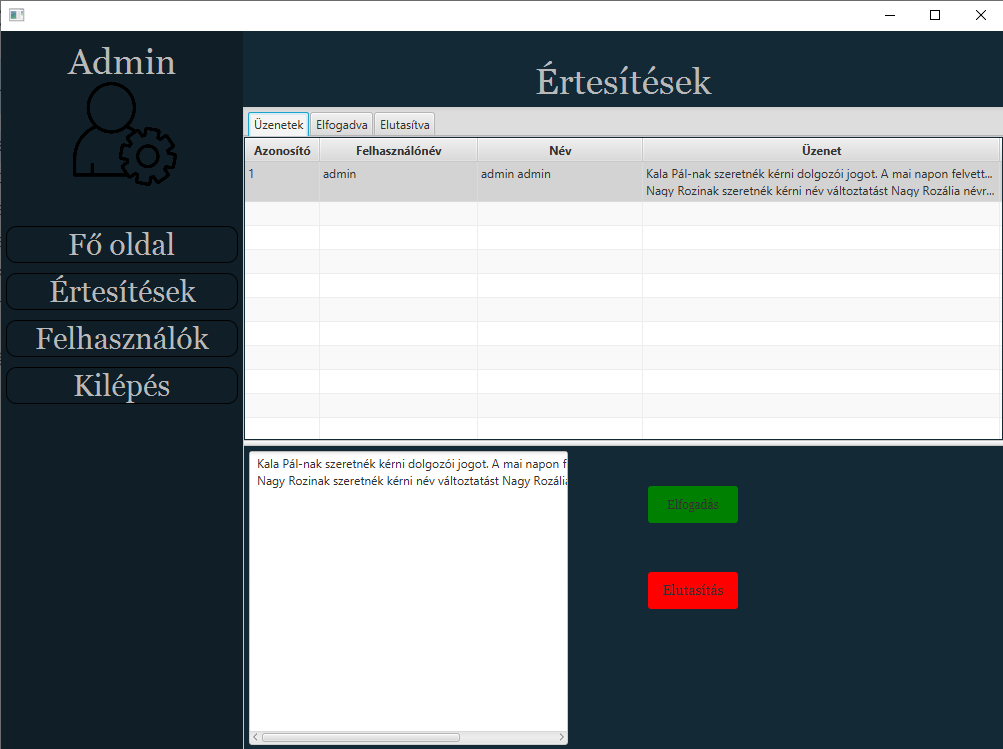


. Profil oldal

Abban az esetben, ha jogosultsága *admin* akkor az oldalon a **Kijelentkezés** gomb alatt megjelenik egy **Admin** gomb, amely átirányít az alkalmazás adminisztrátori felületére.

Az adminisztrátori felület hasonlít a **Főoldalhoz** . Itt megtekinthetőek az **Értesítések**,

**Felhasználók** és egy **Kilépés** gomb.



. Értesítések

Az Értesítések fülön megtekinthetők azok az üzenetek amiket a dolgozók írnak. Az Értesítéseknél olyan üzeneteket fogadhatnak el amelyek felhasználó módosítással kapcsolatosak(Jelszót az adminisztrátor se tud változtatni). Amely kérés nem felel meg ennek azt az adminisztrátornak el kell utasítania. Az **Üzenetek** fülön kívül megtalálható egy **Elfogadva** fül ahol az elfogadott üzenetek találhatóak és egy **Elutasítva** fül ahol az elutasított üzenetek találhatóak.



. Felhasználók

A Felhasználók fülön egy táblázatban fel van sorolva az összes felhasználó. A táblázat alatt egy kereső mező amelyben kereshet felhasználónév, név vagy email alapján a felhasználók között .A kereső mező alatt vannak beviteli mezők. Itt tudja beírni a kívánt adatokat. A beviteli mezők alatt található egy **Törlés** gomb, amely a táblázatban kiválasztott felhasználót tudja törölni, egy **Módosítás** gomb, amivel a táblázatban kiválasztott felhasználó adatait tudja módosítani kivéve az azonosítót, a jelszót és a regisztrációs idejét. Felhasználót is lehet létrehozni , ezt a **Hozzáadás** gombbal tudja megvalósítani miután az összes beviteli mező kitöltésre került kivéve az azonosítót és a regisztrációs időt ugyanis ezeket a rendszer automatikusan tölti ki. A **Kilépés** menüpont vissza lépteti a **Főoldara**. Kijelentkezni a felhasználójából a Főoldalon a Kijelentkezés menüpont segítségével tud kilépni a fiókjából vagy **Profil** menüpontban a jobb felső sarokban a **Kijelentkezés** gombra kattint. Ez vissza irányítja a bejelentkezés felületre.

# Fejlesztői dokumentáció

## Az alkalmazott fejlesztői eszközök

### Trello projektkezelő

A csapatunk a Trello nevű projektkezelőt használta. A Trello megkönnyíti a projektek és feladatok kezelését a csapatok számára. A Trello olyan vizuális eszköz, amely segítette a csapatunk projektjének munkafolyamatát és segített a feladatkövetés kezelésében.



.Trello Ikon

### Git verziókezelő

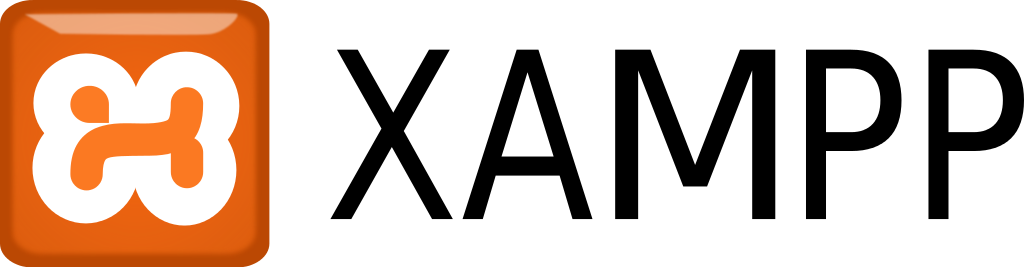
A Git egy ingyenes és nyílt forráskódú elosztott verziókezelő rendszer, amely a kicsitől, a nagyon nagy projektig mindent gyorsan és hatékonyan kezel. A csapatunk a Github rendszerét választotta. Alapvetően is ezt a verziókezelőt használtuk volna. Saját funkcióin felül a Git elosztott verziókövetését és forráskódkezelését (SCM) teszi elérhetővé. Hozzáférés-kezelést és számos együttműködési funkciót nyújt, mint például bug követés, szolgáltatáslekérés, feladatkezelés, valamint wikiket minden projekthez.



. GitHub

### XAMPP

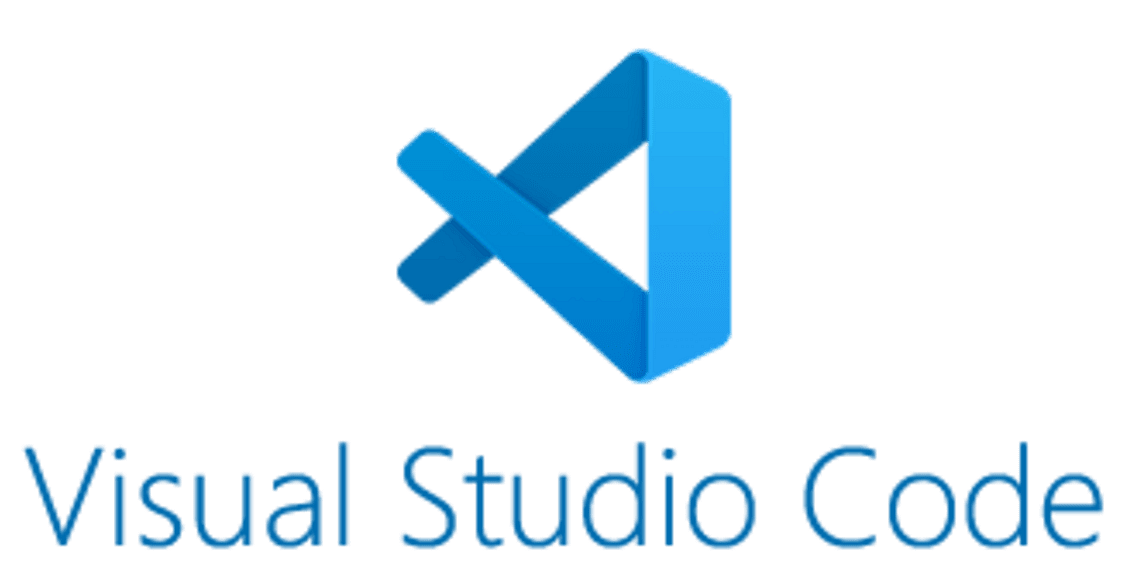
Azért a XAMPP adatbázis kezelőt választottuk, mert a PHP weboldalunknál nagy segítséget nyújtott ennek a használata. Ez egy szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomag, amelynek legfőbb alkotóelemei az Apache webszerver, valamint a PHP programozási nyelv értelmezője.



. XAMPP

### Microsoft Visual Studio Code

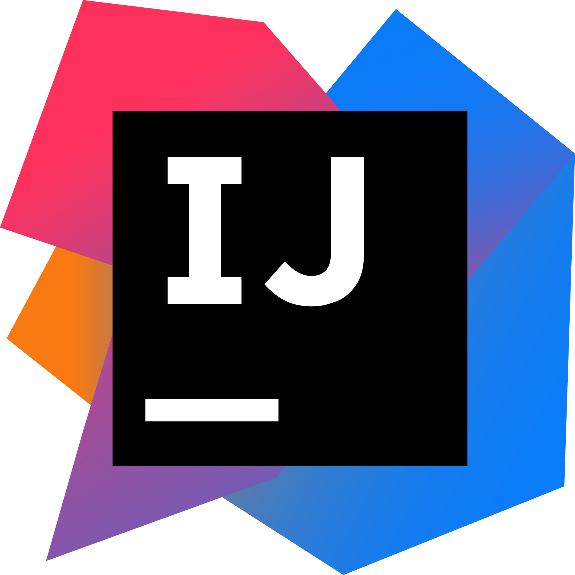
Azért a Visual Studio Code-ot használtuk, mert ezt találtuk a legalkalmasabbnak mivel, sokféle programnyelvet és a Git verzió kezelését is támogatja. Ezen kívül számtalan hasznos bővítmény elérhető hozzá. A Visual Studio Code ingyenes, nyílt forráskódú kódszerkesztő, melyet a Microsoft fejleszt Windows, Linux és OS X operációs rendszerekhez. Támogatja a hibakeresőket, továbbá képes az intelligens kódkiegészítésre az IntelliSense segítségével.



. Microsoft Visual Studio Code

### IntelliJ IDEA

Azért az IntelliJ IDEA-t használtuk, mert ez nyújtja a legnagyobb segítséget a JAVA alkalmazás fejlesztése közben. Az alkalmazást 11.0.14-es verzió számú JDK-val fejlesztettük. Az IntelliJ IDEA egy intelligens, a Java programkód írására alkalmas IDE a Java és más JVM-nyelvek, például a Kotlin, a Scala és a Groovy használatához mindenféle alkalmazáson. Az IntelliJ IDEA bővíthető a JetBrains által fejlesztett ingyenes bővítményekkel, amelyek lehetővé teszik más programozási nyelvekkel való együttműködést. Az IDE (Integrált fejlesztői környezet), olyan szoftver alkalmazás, amely átfogó létesítményeket biztosít a számítógépes programozók számára a szoftverfejlesztéshez.



. IntelliJ IDEA

### Microsoft Office Word

A dokumentáció megírásához a Microsoft Office csomagban található Microsoft Word-öt használtuk. A megbízható Word alkalmazás segítségével gyorsan és egyszerűen létrehozhatja, szerkesztheti, megtekintheti és megoszthatja fájljait másokkal.



. Microsoft Office Word

## Adatmodell leírása

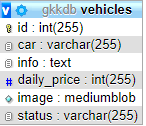
A projektünkhöz használt adatbázisnak a neve „gkkdb”. Ez az adatbázis 4 táblából áll, ezeknek a neve user, vehicles, bookings és message. Ezek között a táblák között kapcsolatok lettek kialakítva. Az adatbázis szerver futtatásához XAMPP szervert használtunk.

### user tábla

36. user tábla

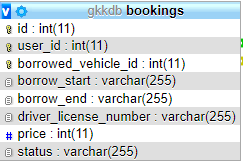
A(z) user táblába a felhasználó adatit tároljuk. A táblában 9 féle adatot tárolunk. Az első az **id**, amely egy azonosító, ami automatikusan kapja, az értékét minden egy regisztrációnál egyel nő a kapott érték. Ez az elsődleges kulcs a táblában. A(z) **username** a felhasználó fantázia neve. Ez a tábla másodlagos kulcsa. A **first\_name** és a **last\_name** a felhasználó keresztneve és vezetékneve. A **password** a felhasználó belépésekor az azonosításhoz szükséges jelszó. Az **email** és a **phone\_number** a felhasználó elektronikus levél címe és telefonszáma. A **registration\_date** (éééé-hh-nn) tárolja, hogy mikor regisztrált a felhasználó. Ez automatikusan tölti fel az adatot az aktuális dátummal. A **rank** a felhasználó jogosultságát tárolja, ezek lehetnek a(z) *user*, a *worker* és az *admin*.

### vehicles tábla

A vehicles táblában a gépjárművek kerülnek tárolásra. Az id az elsődleges kulcs és ez az azonosítója az eltárolt gépjárműveknek. A **car** oszlopban tároljuk a gépjárművek pontos nevét. Az **info** nevű oszlopban tároljuk az adott autó leírását. A **daily\_price** nevű oszlopban van eltárolva a gépjármű napi ára forintban. Az **image** nevű oszlopban tároljuk a képeket amely ***mediumblob*** formátumban van eltárolva. A kép maximális mérete 16 MB lehet. A **status** a gépjármű aktuális állapotát tárolja el melyek a következők lehetnek, *foglalva* és *elérhető*.

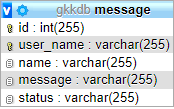
37. vehicle tábla

### booking tábla

 A bookings táblába a felhasználók gépjármű foglalásait tároljuk. Az első az **id**, amely egy elsődleges kulcs foglalás azonosítója. Ez az érték minden foglalásnál egyel nagyobb lesz . A(z) **user\_id** egy idegen kulcs amely a(z) **user** táblából kapja meg a felhasználó azonoítóját. A **borrowed\_vehicle\_id** is egy idegen kulcs ami a **vehicle** táblából kapja meg a gépjármű azonosítóját. A **borrow\_start** és a **borrow\_end** azok a dátumokat tárolja amelyet a felhasználó megad a foglaláskor , innentől kezdődik meg a gépjármű bérlése és fejeződik be. Ezeknek a formátuma éééé-hh-nn. A **driver\_license\_number** a felhasználó vezetői engedély száma amit a foglalásnál ad meg. A **price** az a bérlés megkezdése és a bérlés vége közötti különbség szorozva gépjármű napi árával forintba. A **status** a foglalásnak a státusza amelyek a következők lehetnek, *foglalva*, *elvitte* és a *teljesítve*.

38.bookings tábla

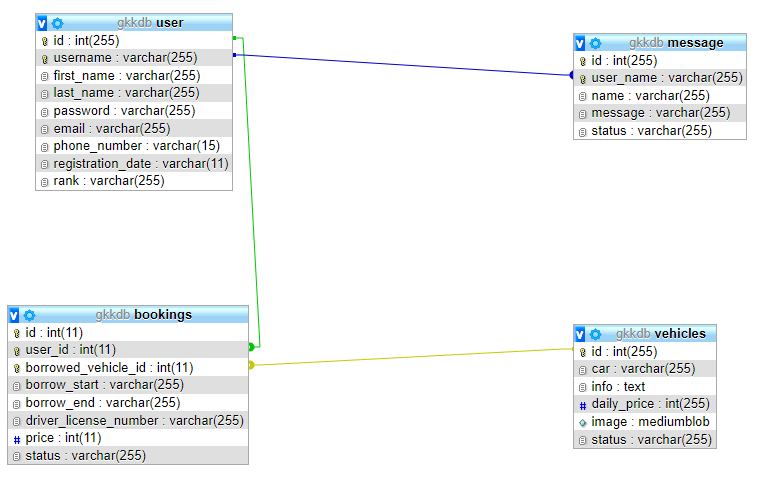
### message tábla

A message tábla azért jött létre, mert az alkalmazásban a worker jogosultsággal bíró felhasználók üzenetet tudjanak küldeni az admin jogosultsággal rendelkező felhasználónak. Az **id** az elsődleges kulcs, ami az üzenet azonosítója. A(z) **user\_name** egy idegen kulcs, ami a(z) user tábla username oszlopával kapcsolódik. A **name** a felhasználó vezetékneve és keresztneve. A **message** oszlopban tárolódik el az üzenet tartalma. A **status** az üzenet státusza, amelyek a következők lehetnek, *elküldve*, *elfogadva* vagy *elutasítva*.

39. message tábla

### Táblák közötti kapcsolatok

A(z) *user* tábla **id** oszlopa csatlakozik a *booking* tábla **user\_id** oszlopához mivel a foglaláskor a felhasználót az **id** alapján azonosítjuk. Illetve a(z) **username** kapcsolódik a *message* tábla **user\_name** oszlopához mivel a küldött üzenetnél megjelenítjük azt a felhasználót, amelyik küldte az üzenetet. A *vehicle* tábla **id** oszlopa kapcsolódik a *bookings* tábla **borrowed\_vehicle\_id** oszlopához mivel tároljuk a foglalásnál a felhasználót, aki foglalta és a foglalt gépjárművet is.



. Táblák közötti kapcsolat

## Részletes specifikáció

### Weboldal részletes specifikációi

#### Adatbázis csatlakozása

function getConnection()

{

    return new pdo(

        'mysql:host=' . $\_SERVER['DB\_HOST'] . ';dbname=' . $\_SERVER['DB\_NAME'],

        $\_SERVER['DB\_USER'],

        $\_SERVER['DB\_PASSWORD']

    );

}

A **getConnection()** függvényünk vissza ad egy pdo-t, amely kapcsolódik a szerverhez. Az adatokat egy ***.htaccess*** fájlból kéri le. A későbbiekben ezzel a függvénnyel lehet csatlakozni az adatbázishoz.

#### Regisztráció

function registrationhandler()

{

    $rank = "user";

    $pdo = getConnection();

    $statement = $pdo->prepare("SELECT username FROM user WHERE username = ?");

    $statement->execute([$\_POST["username"]]);

    $user = $statement->fetch(PDO::FETCH\_ASSOC);

*if* ($user) {

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=regisztracioSikertelen');

*return*;

    }*else* *if*($\_POST["username"] === "" ||  $\_POST["keresztnev"] === "" || $\_POST["vezeteknev"]==="" || $\_POST["jelszo"]==="" ||$\_POST["email"] ==="" || $\_POST["telefonszam"]===""){

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=regisztracioSikertelen');

*return*;

    }*else*{

        $statement = $pdo->prepare("INSERT INTO `user` (`id`, `username`, `first\_name`, `last\_name`, `password`, `email`, `phone\_number`, `registration\_date`, `rank`)

        VALUES (NULL, ? , ? , ? , ? , ? , ? , ? , ?)");

        $statement->execute([

            $\_POST["username"],

            $\_POST["keresztnev"],

            $\_POST["vezeteknev"],

            $\_POST["jelszo"],

            $\_POST["email"],

            $\_POST["telefonszam"],

            date("Y-m-d"),

            $rank

        ]);

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=regisztracioSikeres');

    }

}

A regisztráció csak a weboldalon végezhető el. Létrehozzuk a kapcsolatot az adatbázissal a **getConnection()** függvény segítségével, majd az adatbázisból lekérdezzük a(z) *username* adatot a(z) ***user*** táblából és összehasonlítjuk a beviteli mezőbe beírt felhasználónévvel. Ha létezik már olyan felhasználónév, akkor hiba üzenetet kapunk és ellenőrzi, hogy a beviteli mezők ki lettek-e töltve. Ha nem talált hibát, akkor feltöltésre kerül a(z) ***user*** tábla adatokkal. A dátumot és a rankot a program automatikusan határozza be. Sikeres regisztráció esetén az oldal átnavigálja a bejelentkezéshez.

#### Bejelentkezés

function loginhandler()

{

    $pdo = getConnection();

    $statement = $pdo->prepare("SELECT \* FROM user WHERE username = ? ");

    $statement->execute([$\_POST["username"]]);

    $user = $statement->fetch(PDO::FETCH\_ASSOC);

    if (!$user) {

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=hibasadatok');

        return;

    }

    if (!($\_POST['password'] === $user["password"])) {

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=hibasadatok');

        return;

    }

    $\_SESSION['userID'] = $user['id'];

    $\_SESSION['username'] = $user['username'];

    header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']));

}

A **getConnection()** függvénnyel kapcsolatot létesítünk az adatbázissal, majd lekérdezzük a(z) ***user*** táblából a(z) *username*-t , amit a felhasználó megadott. Ha nem talál olyan felhasználót, akkor hibaüzenetet kapunk. Ha talált olyan felhasználót de, a megadott jelszó és a tárolt jelszó nem egyezik, akkor is hiba üzenetet fogunk kapni. Megegyező adatok esetén cookie-t tárolunk el a böngészőjében, amely tartalmazza az adott felhasználó *id*-át és a *username*-t. Majd ahonnan indította a bejelentkezést, oda fogja navigálni az oldal de, már bejelentkezve.

#### Gépjármű foglalás

function bookingcarHandler()

{

    $user\_id = $\_SESSION['userID'];

    $singlecarid = $\_GET['id'] ?? '';

    $borrowed\_start = $\_POST['start\_booking'];

    $borrowed\_stop = $\_POST['end\_booking'];

    $\_driverlicense = $\_POST['v\_eng\_szam'];

    $status = "foglalva";

    $localdate = date("Y-m-d");

    $pdo = getConnection();

    $statement = $pdo->prepare('SELECT \* FROM vehicles WHERE id = ?');

    $statement->execute([$singlecarid]);

    $car = $statement->fetch(PDO::FETCH\_ASSOC);

    $daily\_price = $car['daily\_price'];

    $\_start = new DateTime($\_POST['start\_booking']);

    $\_stop = new DateTime($\_POST['end\_booking']);

    $interval = $\_start->diff($\_stop);

    $kulonbseg = $interval->format('%a');

    $napszam = $kulonbseg + 1;

    $allprice = $napszam \* $daily\_price;

*if*($borrowed\_start < $localdate || $borrowed\_start === ""){

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=invaliddate');*//nem lehet hamarabb foglalni mint a mai dátum*

*return*;

    }

*if*($borrowed\_start > $borrowed\_stop || $borrowed\_stop === ""){

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=invaliddate1');*//nem lehet - napra foglalni*

*return*;

    }

*if*($\_driverlicense === ""){

        header('Location: ' . getPathWhitId($\_SERVER['HTTP\_REFERER']) . '&info=invaliddriverlicense');*//muszáj megadni a vez eng számot*

*return*;

    }

    $pdo = getConnection();

    $statement = $pdo->prepare("INSERT INTO `bookings` (`id`, `user\_id`, `borrowed\_vehicle\_id`, `borrow\_start`, `borrow\_end`, `driver\_license\_number`, `price`, `status`) VALUES (NULL, ?, ?, ?, ?, ?, ?, ?);");

    $statement->execute([

        $user\_id,

        $singlecarid,

        $borrowed\_start,

        $borrowed\_stop,

        $\_driverlicense,

        $allprice,

        $status

    ]);

    $statement =  $pdo->prepare("UPDATE `vehicles` SET `status` = 'nem\_elerheto' WHERE `vehicles`.`id` = ?");

    $statement->execute([

        $singlecarid

    ]);

    header('Location: ' . '/foglalasaim');*//sikeres foglalás*

}

Foglaláskor bekérjük a felhasználótól a foglalás megkezdésének időpontját és a foglalás befejezésének időpontját illetve a vezetői engedély számát. A sütiben eltárolt *userID*-t is lekérdezzük. A **getConnection()** függvénnyel kapcsolatot létesítünk az adatbázissal, majd lekérdezzük az adott gépjármű adatait es azon belül is a napi árat. Majd árat kiszámítjuk úgy, hogy a kezdés időpontja és a befejezés időpontja közti különbség szorozva az adott gépjármű napi árával. Ezek után a *booking* táblába feltöltjük a foglalás adatait és módosítjuk az adott gépjármű státuszát.

### Java alkalmazás részletes specifikációi

#### Adatbázis csatlakozás

public class DBConnector {

    public static Connection getConnection() throws SQLException {

        Connection con =  DriverManager.getConnection("jdbc:mysql://localhost/gkkdb", "root", "");

*return* con;

    }

}

Az adatbázishoz a DBConnector osztályon belül csatlakozunk. A csatlakozáshoz JDBC drivert használtunk, amellyel lokálisan lehet csatlakozni az adatbázishoz. Ez visszatér egy con paraméterrel és ezt tudjuk használni az osztályoknál, ha valamilyen adatbázissal kapcsolatos műveletet, szeretnénk végrehajtani.

#### Bejelentkezés

*try* {*//létrehozzuk a kapcsolatot az adatbázissal és le kérdezzük az adatokat*

                Connection con = DBConnector.getConnection();

                ResultSet login = con.createStatement().executeQuery("SELECT \* FROM user WHERE username = ('"+ username +"') AND password = ('"+ password +"');");

*if* (login.next()){

                    ResultSet rank = con.createStatement().executeQuery("SELECT rank FROM `user` WHERE (username = '"+ username +"')");

*if* (rank.next()){

                        global\_rank = rank.getString("rank");

*if* (rank.getString("rank").equals("worker") || rank.getString("rank").equals("admin")){

                            FXMLLoader welcome\_view = *new* FXMLLoader(GKKApp.class.getResource("welcome-view.fxml"));

                            Scene welcome\_scene = *new* Scene(welcome\_view.load(), 1000, 600);

                            Stage welcome\_stage = *new* Stage();

                            welcome\_window = welcome\_stage;

                            welcome\_stage.setScene(welcome\_scene);

                            welcome\_stage.setMinWidth(1000);

                            welcome\_stage.setMinHeight(600);

                            welcome\_stage.show();

                            uname\_textfield.setText("");

                            pass\_textfield.setText("");

                            uname\_textfield.requestFocus();

                            login\_window.close();

                        }

                    }*else* {

                        Alert no\_permission\_alert = *new* Alert(Alert.AlertType.CONFIRMATION);*//ha az accountnak nincsen jogosultsága akkor megjelenik egy alert*

                        no\_permission\_alert.setTitle("Hiba");

                        no\_permission\_alert.setHeaderText("Nincs megfelelő jogosultsága!");

                        no\_permission\_alert.initOwner(login\_window);

                        no\_permission\_alert.show();

                        uname\_textfield.setText("");

                        pass\_textfield.setText("");

                        uname\_textfield.requestFocus();

                    }

                }*else* {

                    Alert bad\_data\_alert = *new* Alert(Alert.AlertType.INFORMATION);

                    bad\_data\_alert.setTitle("Hiba");*//hiba ablak jelenik meg a rosszul megadott információk miatt*

                    bad\_data\_alert.setHeaderText("Hibás adatok.");

                    bad\_data\_alert.setContentText("Próbálja újra.");

                    bad\_data\_alert.initOwner(login\_window);

                    bad\_data\_alert.show();

                }

            } *catch* (SQLException e) {

                Alert no\_connection = *new* Alert(Alert.AlertType.ERROR);

                no\_connection.setHeaderText("Sajnos nem tudunk az adatbázishoz csatlakozni.");

                no\_connection.setContentText("Ellenőrizze az internet kapcsolatot.");

                no\_connection.initOwner(login\_window);

                no\_connection.show();

            }

Létrehozzuk a kapcsolatot az adatbázissal és lekérdezzük a megadott felhasználónevet és összehasonlítjuk a hozzá tartozó jelszóval, és ha az is megegyezik a beírt jelszóval, akkor lekérdezzük a hozzá tartozó *rankot* is. A(z) *ranknak* *worker* vagy *admin* jogosultsággal kell rendelkeznie különben ha *use*r jogosultsággal jelentkezik, be a rendszer nem fogja tovább engedni be. Ha a bejelentkezési feltételek sikeresek voltak, akkor tovább engedi a *welcome* ablakra. Ha a jogosultság *user* akkor hibaüzenetet fog kapni, hogy „Nincs megfelelő jogosultsága!”. Ha a felhasználó és a jelszó nem egyezik, akkor is hibaüzenetet fog kapni, hogy „Hibás adatok. Próbálja újra”. Illetve ha az adatbázis szerver nem fut, akkor is hibaüzenetet kap, hogy „Sajnos nem tudtuk az adatbázishoz csatlakozni.”.

#### Gépjármű hozzáadása

private void addcars\_btn(){

        String name = g\_name\_tf.getText();

        String info = Info\_update\_TA.getText();

        String price = Price\_update\_TF.getText();

        String sql = "INSERT INTO `vehicles` (`id`, `car`, `info`, `daily\_price`, `image`, `status`) VALUES (NULL, ? , ? , ? , ?, ?); ";

        PreparedStatement pst;

        FileChooser fc = *new* FileChooser();

        File file = fc.showOpenDialog(open\_folder.getScene().getWindow());

*try* {

*if* (name.equals("") || info.equals("")){

                    Alert error\_alert = *new* Alert(Alert.AlertType.CONFIRMATION);

                    error\_alert.setTitle("Hiba");

                    error\_alert.setHeaderText("Töltse ki az összes mezőt!");

                    error\_alert.initOwner(WelcomeCon.reservation\_window);

                    error\_alert.show();

                }*else* {

*try* {

                        Integer.parseInt(price);

                        Connection con = DBConnector.getConnection();

                        FileInputStream fis = *new* FileInputStream(file);

                        System.out.println(fis);

                        pst = con.prepareStatement(sql);

                        pst.setString(1, name);

                        pst.setString(2, info);

                        pst.setInt(3, Integer.parseInt(price));

                        pst.setBinaryStream(4, fis, fis.available());

                        pst.setString(5, "elerheto");

                        pst.execute();

                        Image image = *new* Image(fis);

                        image\_iv.setImage(image);

                        cars\_TW.getItems().clear();

                        carstable();

                        Alert apply\_del\_alert = *new* Alert(Alert.AlertType.CONFIRMATION);

                        apply\_del\_alert.setTitle("Sikeres gépjármű hozzáadás..");

                        apply\_del\_alert.setHeaderText("Ellenőrizze a hozzáadott elemet.");

                        apply\_del\_alert.initOwner(WelcomeCon.cars\_window);

                        apply\_del\_alert.show();

                    }

*catch* (NumberFormatException e) {

                    Alert error\_alert = *new* Alert(Alert.AlertType.ERROR);

                    error\_alert.setTitle("Hiba");

                    error\_alert.setHeaderText("A napi árnak számot adjon meg!");

                    error\_alert.show();

                }

            } }*catch* (SQLException | IOException e) {

                Alert error\_update\_alert = *new* Alert(Alert.AlertType.CONFIRMATION);

                error\_update\_alert.setTitle("Hiba");

                error\_update\_alert.setHeaderText("Az adatbázis nem tud csatlakozni!");

                error\_update\_alert.setContentText("Próbálja újra!");

                error\_update\_alert.initOwner(WelcomeCon.cars\_window);

                error\_update\_alert.show();

                e.printStackTrace();

            }

        g\_name\_tf.setText("");

        Info\_update\_TA.setText("");

        Price\_update\_TF.setText("");

        image\_iv.setImage(null);

    }

Mielőtt rákattintunk a hozzáadás gombra töltsük ki a beviteli mezőket a gépjármű nevével, a gépjármű leírásával, a gépjármű árával. Ezután kattintson a hozzáadás gombra és ilyenkor megnyílik egy fájlkezelő, amiben ki tudja választani a hozzáadni kívánt képet. Előtte ellenőrzi a kód, hogy ki lettek e töltve a beviteli mezők és kapcsolatot létesít az adatbázissal és feltölti a *vehicles* táblába. Ha sikeres a gépjármű hozzáadása, akkor üzenetet fog kapni, ami így szól: „Ellenőrizze a hozzáadott elemet.”. Amennyiben nem sikerült az adatbázissal csatlakozni hibaüzenetet fog kapni, hogy „Nem sikerült csatlakozni az adatbázishoz.”.

#### Foglalások közötti keresés

private void searchbooking\_btn(){

        String keres = searchbooking\_tf.getText();

        boolean volt =false;

*for* (Cars elem *:* carslist){

*if* (elem.id.equals(keres) || elem.car.equals(keres)){

                index = elem.sorszam;

                volt = true;

            }

        }

*if* (!volt){

            Alert error\_alert = *new* Alert(Alert.AlertType.ERROR);

            error\_alert.setTitle("Hiba");

            error\_alert.setHeaderText("Nem található ezekkel az adatokkal foglalás!");

            error\_alert.show();

            searchbooking\_tf.setText("");

            searchbooking\_tf.setPromptText("Keresés azonosító vagy gépjármű név alapján...");

        }*else* {

            CID\_lab.setText("Gépjármű azonosító: " + id\_TC.getCellData(index).toString());

            name\_lab.setText("Neve: " + CarName\_TC.getCellData(index).toString());

            g\_name\_tf.setText(CarName\_TC.getCellData(index));

            dailyprice\_lab.setText("Információ: " + info\_TC.getCellData(index).toString());

            info\_lab.setText("Napi ára: " + dailyprice\_TC.getCellData(index).toString() + "Ft.");

            Info\_update\_TA.setText(info\_TC.getCellData(index).toString());

            Price\_update\_TF.setText(dailyprice\_TC.getCellData(index).toString());

            image\_iv.setImage((Image) image\_tc.getCellData(index));

            searchbooking\_tf.setText("");

            searchbooking\_tf.setPromptText("Keresés azonosító vagy gépjármű név alapján...");

        }

    }

Bekérjük a felhasználótól a foglalás azonosítóját vagy a gépjármű nevét és végig járjuk az elemeket, és ha az értékük egyezik a felhasználó által megadott adatokkal, akkor beállítjuk a bevitelik mezők értékét a talált adatokkal. Abban, ez esetben, ha nem talál ilyen adatot, akkor egy üzenetet fog kapni, hogy „Nem található ezekkel az adatokkal.”.

## Tesztelési dokumentáció

### Java alkalmazás tesztelése

A Java alkalmazás tesztelését egy segédprogrammal hajtottuk végre, aminek a neve **GS Auto Clicker**. Ez a program önműködően kattint, ahová szeretnénk, és annyiszor amennyiszer szeretnénk. A **Scenebuilderbe** kitöltöttük az összes beviteli mezőket és beállítottuk a programban, hogy vigye végig az összes lehetséges funkciót. A segédprogram 50 milliszekundumos késleltetéssel hajtotta végre a kattintásokat és 15 percen keresztül futtattuk a tesztet, futtatás közben nem tapasztaltuk adatvesztést és a program se panaszkodott illetve a számítógépet se terhelte le. Az adatbázisban az összes adat feltöltésre került a teszt során.

### Webalkalmazás tesztelése

A webalkalmazás tesztelését egy segédprogrammal hajtottuk végre, aminek a neve **GS Auto Clicker**. Ez a program önműködően kattint, ahová szeretnénk, és annyiszor amennyiszer szeretnénk. A Google Chrome automatikus kitöltőjével kitöltöttük a szükséges mezőket. A segédprogram 50 milliszekundumos késleltetéssel hajtotta végre a kattintásokat és 15 percen keresztül futtattuk a tesztet, futtatás közben nem tapasztaltuk adatvesztést és a program se panaszkodott illetve a számítógépet se terhelte le. Az adatbázisban az összes adat feltöltésre került a teszt során.

## Továbbfejlesztési lehetőségek

A projektünkben megannyi továbbfejlesztési lehetőség van. Amikre mi gondoltunk és felhasználók számára hasznos és informatív lehet. Első sorban egy adott gépjárműről több fénykép tárolása és megjelenítése, ezen kívül a gépjárművek kategorizálása és részletesebb információ a gépjárművekről. Ezzel megkönnyítenénk a felhasználót a fölösleges keresésektől és több (pl. hengerűrtartalom, ajtók száma, csomagtartó mérete, fogyasztás) információt tudna az adott gépjárműről. Illetve ha a gépjárművet az aktuális dátumtól későbbi időpontra foglalják, akkor csak a foglalás időpontjától ne legyen elérhető az autó. Ezzel több foglalást tudnánk elérni. Továbbá lehessen értékelni a gépjárműveket illetve véleményt írni róluk, így a következő felhasználó már tapasztalt személyek véleménye alapján tudja kiválasztani azt a gépjárművet, amit szeretne. Gépjármű foglalása esetén online fizetési lehetőség is legyen elérhető a weboldalon például egy **SimplePay**-en keresztül. Ha lefoglalásra kerül az autó, akkor a felhasználó, aki foglalja a gépjárművet, férjen hozzá a gépjármű saját oldalához, ahol problémát vagy műszaki hibát tud bejelenteni a cégnek. Illetve regisztrációnál, foglalásnál, foglalás lejártakor, adatok módosításánál és fizetésnél felhasználó megadott email címére küldjön egy automatikus üzenetet. A felhasználónak lehetősége legyen beállítani a két faktoros hitelesítést, ez által a felhasználó adatai nagyobb biztonságba vannak. Ezt a két faktoros hitelesítést három féle módon oldanánk meg, az email címre, SMS-ben vagy az **Authenticator** nevű alkalmazáson keresztül küldött kód alapján. Regisztrációnál egy reCaptcha figyelje, hogy aki be akar regisztrálni az nem robot. A Java alkalmazásban a gépjárművek kezelése oldalon amikor a felhasználó új járművet szeretne feltölteni akkor a kép feltöltésnél bármilyen formátumú fájlt fel tud tölteni és ezzel sebezhetővé válik a program. Amely képet a felhasználó fel szeretne tölteni azokat a képeket előnézetbe jelenjenek meg még feltöltés előtt és azokat el tudja fogadni vagy utasítani.

## Irodalomjegyzék