**Váci SZC Petőfi Sándor Műszaki Technikum,  
Gimnázium és Kollégium**



# SZOFTVERFEJLESZTŐ ÉS TESZTELŐ

# Záródolgozat

**Cím: Luxus autókölcsönző weboldal**

**Készítette: Morvai Máté**

2025Alulírott ………………… tanuló ezúton nyilatkozom, hogy a szakdolgozatot teljes egészében saját munkámból készítettem el. A felhasznált forrásokat, mint például tudományos írások, online anyagok, személyes beszélgetések stb., megfelelő forrásmegjelöléssel és idézőjelekkel alkalmaztam, minden egyes hivatkozást pontosan feltüntetve. Az elkészült dolgozatban szereplő eredmények a képzőintézmény és a feladatot kiíró intézmény számára, díjazás nélkül, saját célra felhasználhatóak, figyelembe véve a titkosítással kapcsolatos esetleges korlátozásokat.

**Tartalom jegyzék**

[SZOFTVERFEJLESZTŐ ÉS TESZTELŐ 1](#_Toc193741173)

[Záródolgozat 1](#_Toc193741174)

[Bevezetés 5](#_Toc193741175)

[Felhasználói dokumentáció 6](#_Toc193741176)

[A program általános specifikációja. 6](#_Toc193741177)

[Program rendszerkövetelményei 6](#_Toc193741178)

[Program telepítése 6](#_Toc193741179)

[Google Chrome telepítése 6](#_Toc193741180)

[XAMPP telepítése 7](#_Toc193741181)

[Node.js telepítése 8](#_Toc193741182)

[Visual Studio Code telepítése 10](#_Toc193741183)

[Fejlesztői dokumentáció 11](#_Toc193741184)

[Témaválasztás indoklása 11](#_Toc193741185)

[Alkalmazott fejlesztői eszközök 11](#_Toc193741186)

[Adatmodell leírása 13](#_Toc193741187)

[cars (autók) tábla 13](#_Toc193741188)

[orders (rendelések) tábla 14](#_Toc193741189)

[users (felhasználók) tábla 15](#_Toc193741190)

[Kapcsolatok és adatkapcsolati modell 15](#_Toc193741191)

[Összes tábla 16](#_Toc193741192)

[Adatbázis összegezve 16](#_Toc193741193)

[Szerkezetek 16](#_Toc193741194)

[Részletes feladatspecifikáció, algoritmusok 17](#_Toc193741195)

[Backend, Frontend szerveroldali technológiák 17](#_Toc193741196)

[Express.js 17](#_Toc193741197)

[Használata a szakdolgozatomba 17](#_Toc193741198)

[mysql2-promise használata 18](#_Toc193741199)

[Használata a szakdolgozatomban 18](#_Toc193741200)

[dotenv használata 19](#_Toc193741201)

[Használata a szakdolgozatomban 19](#_Toc193741202)

[Biztonság és autentikáció 19](#_Toc193741203)

[Argon2 19](#_Toc193741204)

[Használata a szakdolgozatomban 19](#_Toc193741205)

[JWT használata 20](#_Toc193741206)

[Használata a szakdolgozatomban 20](#_Toc193741207)

[CORS használata 21](#_Toc193741208)

[Használata a szakdolgozatomban 21](#_Toc193741209)

[Nodemailer használata 21](#_Toc193741210)

[Használata a szakdolgozatomban 21](#_Toc193741211)

[React használata 22](#_Toc193741212)

[Használata a szakdolgozatomban 22](#_Toc193741213)

[Összegzés a technológiák kihasználásáról 23](#_Toc193741214)

[Forráskód 24](#_Toc193741215)

[Fontosabb forráskódok és programozási modell bemutatása 24](#_Toc193741216)

[GET, POST, PUT 24](#_Toc193741217)

[Hookok 24](#_Toc193741218)

[Cars (autók) mappa bemutatása 24](#_Toc193741219)

[Bentley.jsx 24](#_Toc193741220)

[Komponens létrehozása és állapotkezelés 25](#_Toc193741221)

[Kedvenc gomb kezelése 25](#_Toc193741222)

[Kölcsönzés gomb kezelése 26](#_Toc193741223)

[Reszponzív Masonry Grid beállítása 26](#_Toc193741224)

[Összegzés 27](#_Toc193741225)

[Components (Komponensek) mappa bemutatása 28](#_Toc193741226)

[Footer.jsx 28](#_Toc193741227)

[Navbar.jsx 29](#_Toc193741228)

[Profile.jsx 31](#_Toc193741229)

[Profil szerkesztés 32](#_Toc193741230)

[E-mail cím és jelszó módosítása 33](#_Toc193741231)

[Rent.jsx 34](#_Toc193741232)

[Felhasználói adatok lekérdezésének folyamata 34](#_Toc193741233)

[Rendelés leadásának folyamata 35](#_Toc193741234)

[Formázás 35](#_Toc193741235)

[Tesztelés 36](#_Toc193741236)

[Hibák 36](#_Toc193741237)

[Továbbfejlesztési lehetőségek 37](#_Toc193741238)

[Összegzés 37](#_Toc193741239)

# Bevezetés

Ezt az adott koncepciót az autók iránti szeretetünk inspirálta, és partnerként Kucsik Gábor diáktársamat választottam. Egy luxusautó-kölcsönző weboldalt készítünk, amelynek célja, hogy mindenki számára elérhetővé tegyük a prémium életstílust. Tervünk, hogy a felhasználók betekintést nyerhessenek a luxus világába, megszabadulva a hétköznapok szürkeségétől, miközben élvezhetik a prémium autók adta exkluzív élményeket.

A dizájn során a luxus és kényelmes stílusjegyeket követtem, hogy az oldal elegáns és letisztult

legyen, de ugyanakkor figyelemfelkeltő is. A fejlesztés során kiemelt figyelmet fordítottunk a

felhasználói élmény folyamatos javítására, így biztosítva, hogy minden interakció

zökkenőmentes és élvezetes legyen. A szakdolgozatban alkalmazott technológiák és

programozási nyelvek közé tartozik a JavaScript a rugalmasság érdekében, CSS a

megjelenésért, MySQL adatbázis a könnyebb kezelhetőség miatt, valamint Node.js, mivel a

PHP kettőnk véleménye szerint elavult és nem elég rugalmas a modern webfejlesztéshez.

Az adatbázis és ezekhez hasonló technológiák biztosítják a dinamikus tartalom kezelését és a gyors működést, míg a frontend biztosítja a felhasználóbarát és esztétikus felületet. Célunk, hogy az oldal minden eszközön és böngészőben jól működjen, így a felhasználók számára bárhol, bármikor könnyedén hozzáférhető legyen a prémium autók bérlésének élménye, Emellett különös figyelmet fordítunk a reszponzív dizájnra, hogy az oldalunk mobilon, tableten és asztali számítógépen egyaránt kifogástalan felhasználói élményt nyújtson. Hosszú távú célunk, hogy a weboldal folyamatos fejlesztésével és új funkciók, választható opciók bevezetésével tovább növeljük a felhasználói élményt, valamint hogy szolgáltatásunk kiváló legyen és lényegre törő, egyszerűbb.

# Felhasználói dokumentáció

## A program általános specifikációja.

## Program rendszerkövetelményei

Oldalunk eléréséhez kezdetben szükségünk van egy böngészőre, melynek "up to date"-nek lennie, tehát tudják használni az általunk használt program nyelveket.

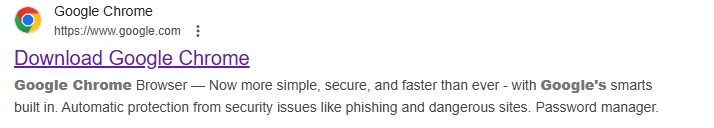
Böngészőnek személy szerint a Google Chrome böngészőt tudom ajánlani, ez egyszerű és letisztult, illetve használata is kézre esik.

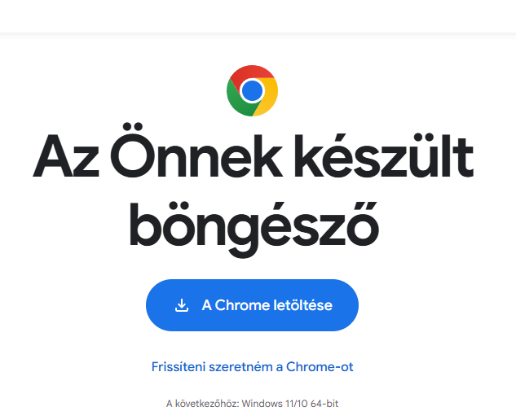
## Program telepítése

A programok telepítése több tényezőből fog állni, szükségünk lesz a „Google Chrome” nevezetű böngészőre. Backend végett a node.js-re mely egy szoftverrendszer amit skálázható internetes alkalmazások, tehát webszerverek készítésére hoztak létre. Szükségünk lesz a XAMPP nevezetű szabad és nyílt forrású webszerver és szoftvercsomag kezelőre az adatbázis érdekében. Ezen felül le kell töltenünk a Visual Studio Code nevű nyílt forráskódú kódszerkesztő, itt fogjuk elindítani a weboldalunkat.

## Google Chrome telepítése

Operációs rendszerünktől függ, hogy az alap böngészőnk pontosan micsoda, számomra a Microsoft Edge. Ez esetben, a Microsoft Edge böngészőnket elinditjuk, majd beirjuk, hogy „Google Chrome letöltése” majd rákattintunk a következőre.

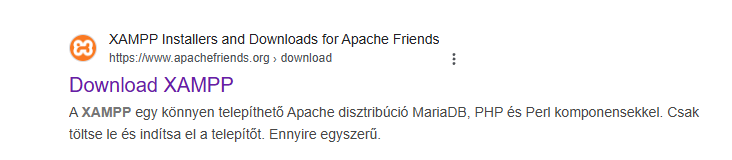


Ez átirányit minket egy oldalra, melyen rákattintunk a „Chrome letöltése” opcióra, a telepitő a fájlkezelőnkbe fog letöltődni.

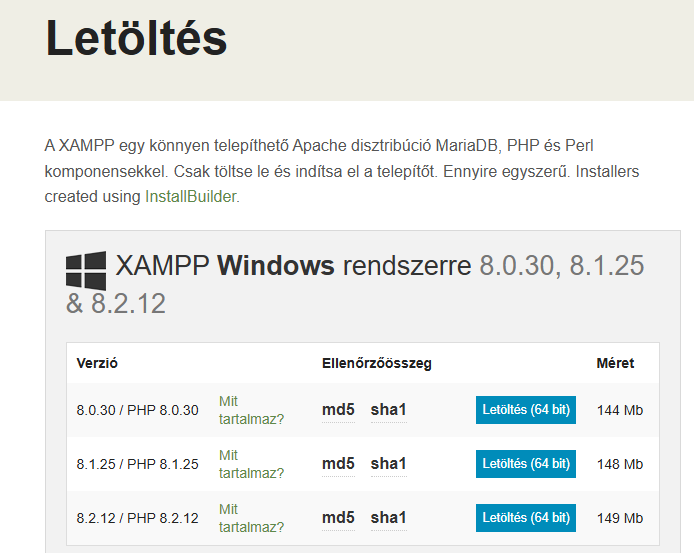
Nyissuk meg a „Chrome Setup” névre hallgató telepitőnket

## XAMPP telepítése

Meglévő böngészőnkkel keressünk rá a XAMPP telepítése címszóra , majd kattintsunk erre:



Ha ezzel készen vagyunk, akkor megjelennek előttünk különböző verziók, ahol fentről a legelsőt válasszuk ki, majd kattintsunk a letöltés gombra.

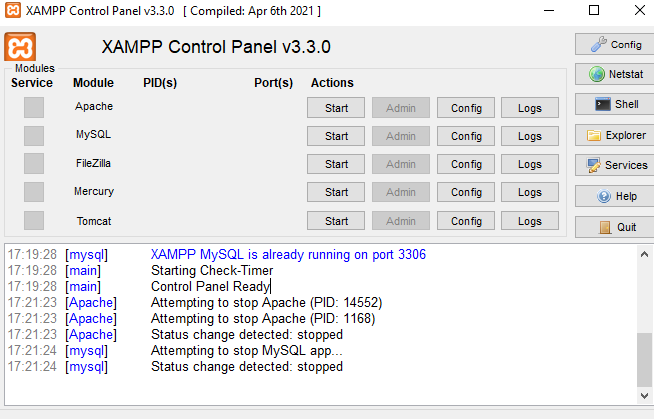


A letöltés magától el kell induljon ha viszont ez nem következik be akkor kurzorunkkal kattintsunk az alábbi zöld „Kattintson ide” szövegre.

Igy biztosra megyünk, hogy a telepitő letöltsön, miután ezt véghez vittük menjünk bele a letöltésekbe és indítsuk el a telepítőt.



Miután elindítottuk, figyelmesen vigyük végig a telepitést majd indítsuk el a programot. Ha az asztalunkon nem jelenik meg az ikonja. Akkor a Windows ikonra kattintsunk rá bal alul s írjuk be, hogy „XAMPP” majd indítsuk el, ezek végrehajtása után meg kell, hogy jelenjen előttünk a XAMPP.



Kattintsunk a „Start” gombokra a MySql illetve az Apache mellett, ehhez rendszergazdai jogosultságokra lesznek szükségünk. Ha elindítottuk, akkor ezzel is készen vagyunk.

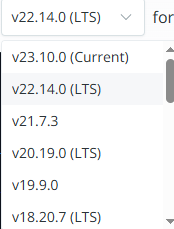
## Node.js telepítése

A node.js telepitéséhez újból a böngészőnk segítségére lesz szükségünk. Nyissuk meg azt, és írjuk be, hogy „Node.js letöltése”, ha eredményként ezt az oldalt kaptuk, akkor kattintsunk rá.

Ezt követően egy letöltési oldal fog a szemünk előtt lebegni.



Nagyon figyeljünk arra, hogy a „Get Node.js” mellett verziók találhatóak, kattintsunk a v22.14 LTS fülre nyissuk le majd kattintsunk erre a 20.19. 0 (LTS) verzióra.



Mindezek után töltsük le a .msi telepítőt, amely úgyszintén a letöltéseinkbe fog megjelenni.



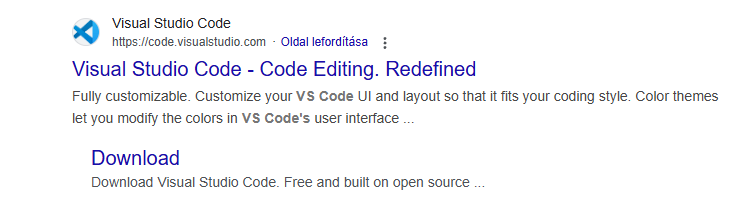
Indítsuk el a telepítőt, és haladjunk tovább mindaddig amig fel nem telepiti a node.js-t



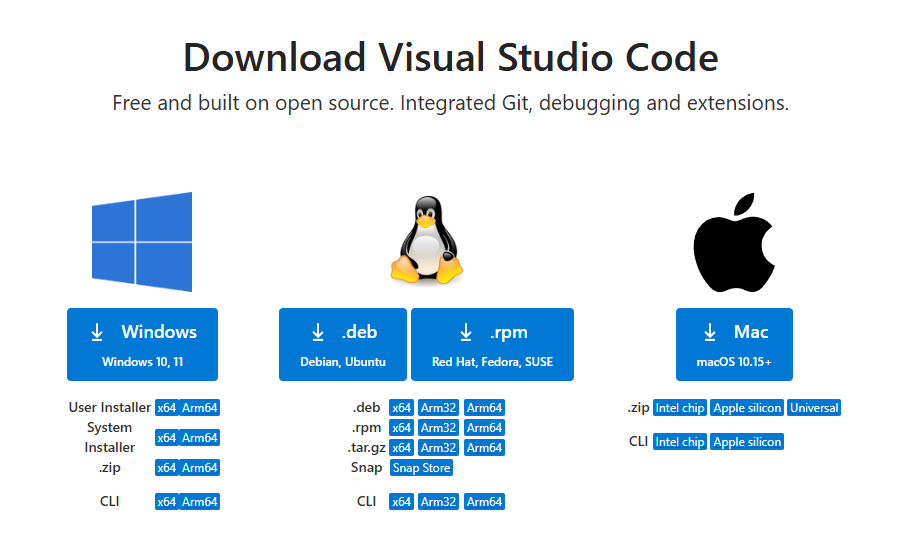
Ha ezzel készen vagyunk akkor már csak a Visual Studio Code van hátra.

## Visual Studio Code telepítése

Nyissuk meg a böngészőnket újból és írjuk be, hogy „Visual Studio Code letöltés”



Kattintsunk a „Download” tehát a letöltés opcióra, majd válasszuk ki az operációs rendszerünket.



Az én esetemben ez Windows. Kattintsunk a kék gombra, majd indítsuk el a telepítőt melyt letöltés után a letöltéseinkbe találunk.



# Fejlesztői dokumentáció

## Témaválasztás indoklása

Számunkra a gépjárművek felé irányuló szeretetünk és érdeklődésünk partneremmel kiskorunk óta fontos szerepet játszik az életünkben. Megoldást kerestünk két meg nem oldott problémára, ez két fő pontból áll, amely a luxusautók élményének átérzése, ezért is van feltüntetve kezdőlapunkon a következő: Luxus | Elegancia | Életérzés.

A második pont az autók kölcsönzésének és menetének le egyszerűsítése. Ez két lényeges tulajdonságból adódik össze. Az egyik a gyorsaság, a célunk az, hogy Magyarországon belül ha bárki luxus autót kölcsönöz, akkor ezt gyorsan tehesse és megbízzon a szolgáltatásba amit nyújtunk. Oldalunk le egyszerűsíti a menetét, mi meg teszünk érte, hogy a gépjármú minél előbb az ügyfélhez kerüljön.

A weboldalunk lefejlesztését teljesmértékben élveztük mert szerettünk és szeretünk is gépjárművekhez kapcsolódó problémákkal foglalkozni. Nekünk ez fontos, és a közeljövőben tervezünk több hasonló munkákkal is dolgozni.

## Alkalmazott fejlesztői eszközök

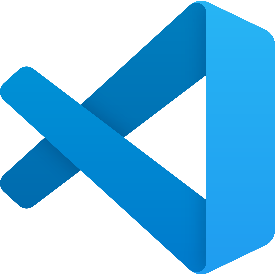
Az eszközök számunkra szoftveralapú alkalmazások, ezekből valamennyire a felhasználóknak is szüksége van.

* **XAMPP**: A webszerver létrehozására és fenntartására.
* **Visual Studio Code:** Kódszerkesztés, illetve tesztelések, hibák feltárása
* **Google Chrome:** Weboldal futtatása, weboldal tesztelése.

A fejlesztéshez és az adatbázis kezeléséhez az **XAMPP** szoftvercsomagot használtuk. Ez az eszköz lehetővé tette számunkra az Apache, MySQL és PHP futtatását, így a webalkalmazásunk számára biztosítottuk a szükséges és kihagyhatatlan háttérrendszert. A XAMPP segítségével könnyedén létrehoztuk és fenntartottuk az adatbázisunkat, amely az oldalunk autóit, rendeléseit, profiljait egyben autentikációs rendszerünket és szolgáltatásainkat kezelte.



A projekt fejlesztésének központi eszközeként **Visual Studio Code**-ot használtunk amely számomra a legtökéletesebb kódszerkesztő program ami jelenleg a marketen található. Ez a kódszerkesztő program támogatja a különféle programozási nyelvek használatát ami nagyon nagy előny volt számukra. A kódírás során egyszerűen dolgozhattunk a különböző fájlokkal és mappákkal, és élvezhettük a beépített hibajavítást, valamint bővítményeket is lehet telepíteni ami különösen kiemeli a felhasználói élményt.



A teszteléshez a **Google Chrome** böngészőt használtuk. Az alkalmazás megfelelő működését folyamatosan teszteltük, hogy biztosak legyünk abban, hogy a felhasználói élmény hibamentes és nincs olyan nagy hiba ami ezt károsítsa. Fontos volt, hogy az oldal minden funkciója megfelelően működik a különböző eszközökön és képernyőméreteken.



**Photoshop és Google Chrome**: A dizájn megalkotásához, valamint az illusztrációk és vizuális elemek létrehozásához **Adobe Photoshop**-ot és **Google Chrome-ot** használtunk.



## Adatmodell leírása

Az adatbázis neve „**racedatabase”**, amely egy autókereskedés online rendelési és felhasználói adatait tárolja. A rendszer lehetőséget biztosít az ügyfelek számára, hogy különböző autómárkák és modellek közül válasszanak, majd online rendelést adjanak le. A rendszer az autók, a rendelések és a felhasználók adatait három fő táblában tárolja, amelyek között logikai kapcsolatok is létrejönnek, ennek felépítéséhez a XAMPP szoftvercsomag járult hozzá.

Az adatbázis minden szükséges lehetőséget tartalmaz mint például lehetővé teszi az autók adatainak kezelését, a rendelések nyomon követését és a felhasználók adatainak tárolását. A táblák közötti kapcsolatok biztosítják az adatintegritást és az összekapcsolhatóságot, így egy hatékony, gyors, egyszerű és jól használható adatbázisstruktúrát hoznak létre amely szeretettel bővíti a felhasználói élményt.

## cars (autók) tábla

* **id** (INT, PRIMARY KEY, AUTO\_INCREMENT) – Az autó egyedi azonosítója.
* **brand** (VARCHAR(255)) – Az autó márkája (pl. BMW, Mercedes, Audi).
* **name** (VARCHAR(255)) – Az autó pontos modellneve.
* **year** (INT) – Az autó gyártási éve.
* **engine** (VARCHAR(255)) – A motor típusa és specifikációi.
* **image\_url** (VARCHAR(255)) – Az autó képének URL-je.
* **price** (VARCHAR(255)) – Az autó ára, amely szöveges formátumban van tárolva.

Az első tábla, a "cars", az elérhető járművek adatait tartalmazza. Minden autó egyedi azonosítóval (id) rendelkezik, amely biztosítja az egyértelmű beazonosítást, mindegyik Auto Increment, ezzel van megoldva az egyediség. A táblában tárolt információk közé tartozik a márka (brand) és a modellnév (name), amelyek alapján a vásárlók könnyen azonosíthatják az elérhető autókat. A gyártási év (year) szintén szerepel az adatok között, mivel ez fontos szempont lehet az autó állapotának és értékének meghatározásában. További jelentős adat a motor típusa (engine), amely részletezi az autó hajtásláncát, például benzines vagy dízel motorral rendelkezik-e. A táblázat tartalmazza az autók árát (price), amely a vásárlók számára kulcsfontosságú információ. A vizuális megjelenítés érdekében az autókhoz tartozik egy kép URL-címe (image\_url), amely lehetőséget biztosít a járművek online megjelenítésére.

## orders (rendelések) tábla

* **id** (INT, PRIMARY KEY, AUTO\_INCREMENT) – Egyedi rendelési azonosító.
* **fullname** (VARCHAR(255)) – A vásárló teljes neve.
* **email** (VARCHAR(255)) – A vásárló e-mail címe, amelyre visszaigazolást kaphat.
* **phonenumber** (VARCHAR(255)) – Telefonszám a kapcsolattartáshoz.
* **address** (VARCHAR(255)) – Szállítási cím.
* **comment** (VARCHAR(255)) – Egyéb megjegyzés a rendeléshez.
* **carId** (INT, FOREIGN KEY) – Az autó azonosítója, amelyre a rendelés vonatkozik.
* **userId** (INT, FOREIGN KEY) – A rendelést leadó felhasználó azonosítója.

A második tábla, az (orders), a leadott rendeléseket kezeli. Minden rendeléshez tartozik egy egyedi azonosító (id), amely biztosítja az egyértelmű nyomon követhetőséget és a félreértések elkerülését. A tábla tartalmazza a megrendelő nevét (fullname), amely elengedhetetlen a regisztrációhoz és a rendelésekhez. Az ügyfél e-mail címe (email) és telefonszáma (phonenumber) lehetővé teszi a gyors kapcsolatfelvételt, akár visszajelzést egy regisztrációról. A rendelés kiszállítása az (address) tehát a lakcím alapján történik. Emellett egy megjegyzés (comment) mező biztosítja, hogy a vevő a problémáit egyszerűen tudja jelenteni. A tábla fontos kapcsolatai közé tartozik a carId mező, amely a megrendelt autóra mutat, valamint a userId mező, amely a rendelést leadó felhasználóhoz kapcsolódik.

## users (felhasználók) tábla

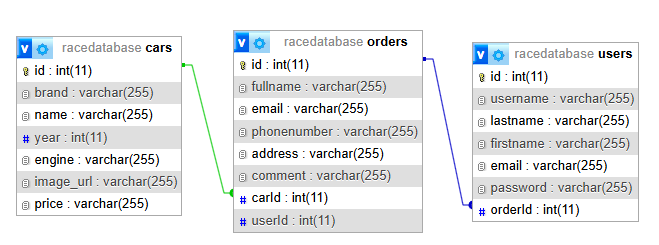
* **id** (INT, PRIMARY KEY, AUTO\_INCREMENT) – Egyedi felhasználói azonosító.
* **username** (VARCHAR(255)) – A felhasználó bejelentkezési neve.
* **lastname** (VARCHAR(255)) – A felhasználó vezetékneve.
* **firstname** (VARCHAR(255)) – A felhasználó keresztneve.
* **email** (VARCHAR(255)) – E-mail cím a regisztrációhoz.
* **password** (VARCHAR(255)) – A titkosított jelszó.
* **orderId** (INT, FOREIGN KEY) – A rendelés azonosítója, amely a felhasználóhoz kapcsolódik.

A harmadik tábla, a "users", a rendszerben regisztrált felhasználók adatait tartalmazza. Minden felhasználó egyedi azonosítóval (id) rendelkezik. Az azonosításhoz és a személyes adatok kezeléséhez a tábla tartalmazza a felhasználó vezetéknevét (lastname), keresztnevét (firstname), e-mail címét (email) és felhasználónevét (username). Az adatbiztonság érdekében a jelszó (password) mező is szerepel a táblázatban, amely titkosított formában tárolható. A "users" tábla kapcsolatban áll a "orders" táblával az orderId mezőn keresztül, amely rögzíti, hogy az adott felhasználó milyen rendeléseket adott le.

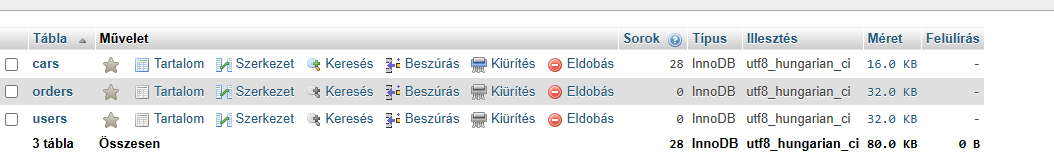
## ****Kapcsolatok és adatkapcsolati modell****

* Az **orders** tábla **carId** oszlopa egy idegen kulcs a **cars** tábla **id** oszlopára, amely azt jelzi, hogy a felhasználó melyik autóra adta le a rendelést.
* Az **orders** tábla **userId** oszlopa egy idegen kulcs a **users** tábla **id** oszlopára, amely azt jelzi, hogy az adott rendelést melyik felhasználó adta le.
* A **users** tábla **orderId** oszlopa idegen kulcsként hivatkozik az **orders** tábla **id** oszlopára, amely azt mutatja, hogy egy felhasználóhoz tartozik rendelés vagy sem.

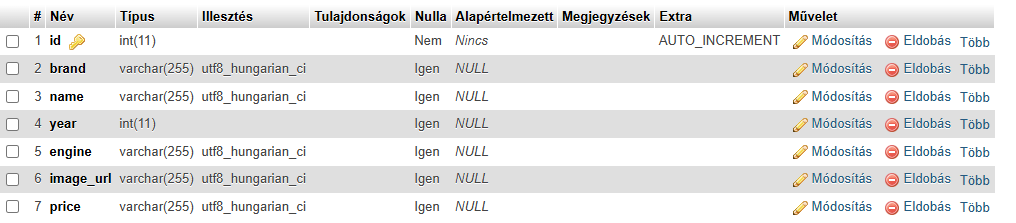
## Összes tábla



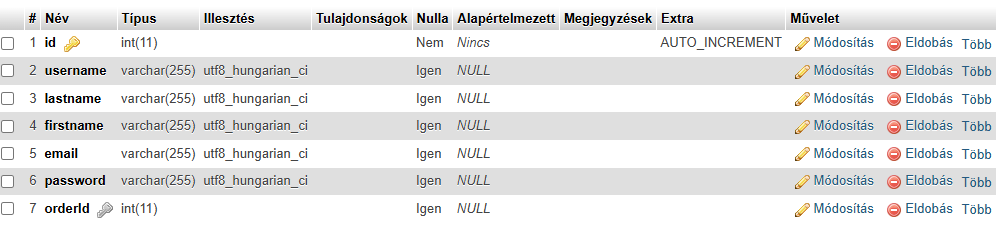
## Adatbázis összegezve



## Szerkezetek







# Részletes feladatspecifikáció, algoritmusok

## Backend, Frontend szerveroldali technológiák

A jobb átláthatóság végett szeretném bemutatni milyen technológiákat és csoportokat, könyvtárakat használtam.

## Express.js

Express.js egy minimalista és rugalmas webkeretrendszer a Node.js számára, amelyet szerveroldali alkalmazások fejlesztésére használnak. Segítségével könnyedén létrehozhatók REST API-k, kezelhetők HTTP-kérések és válaszok, valamint middleware-ek alkalmazhatók az adatok feldolgozására.

# Használata a szakdolgozatomba

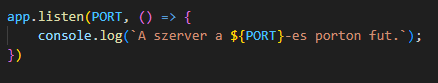
Weboldalam backendjének fejlesztéséhez az Express.js keretrendszert használtam. Az alkalmazásban middleware-eket alkalmaztam a kérések kezelésére, illetve integráltam az adatbáziskezelést és az autentikációért felelős beléptető rendszert ezzel a maximumot nyújtva az oldalnak.

Első fontos lépés a middleware-ek használata volt. A cors() middleware biztosítja, hogy a szerver engedélyezze a domainhez küldött kéréseket, míg az express.json() lehetővé teszi a JSON adatok kezelését.

API végpontok kezelése szintén kulcseleme volt a projekt során. Például a regisztrációs funkció egy POST végpont segítségével történik, amely ellenőrzi a felhasználónév egyediségét, titkosítja a jelszót, és elmenti az adatokat az adatbázisba.



Szerver indítása elengedhetetlen része az alkalmazás működésének. Az Express.js segítségével ezt az alábbi módon oldottam meg:

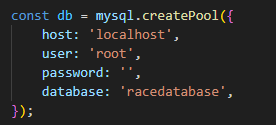


## mysql2-promise használata

A mysql2/promise egy olyan csomag, amely lehetővé teszi az eltérő időben történő adatbázis-kapcsolatot és lekérdezéseket a Node.js környezetben. A promise hozzájárul, hogy a hagyományos callback-alapú megközelítés helyett async/await használatával könnyebb és átláthatóbb és jobb kódot írjunk.

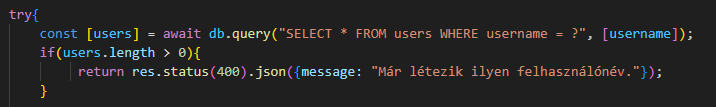
## Használata a szakdolgozatomban

Projektem során a mysql2/promise csomagot használtam az adatbázis kezeléshez, mivel ez lehetővé teszi az aszinkron SQL lekérdezéseket. Az adatbáziskapcsolat beállítása így történt:



createPool() metódus lehetővé teszi egy kapcsolatcsoport létrehozását, ami egyértelműen könnyebbé teszi az adatbázis-műveleteket, mivel a kapcsolatok újra felhasználhatóak, és nem kell minden egyes lekérdezésnél új kapcsolatot nyitni.

Példa arra, hogyan használtam a mysql2/promise csomagot egy felhasználó lekérdezésére:



Ez a kód egy SQL lekérdezést hajt végre az users táblán, és visszaadja az összes olyan felhasználót, akinek a felhasználóneve megegyezik a keresett kritériumokkal. Ez azért lesz fontos, mivel ez alapján döntjük majd el, hogy létezik-e az adott user.

## dotenv használata

A dotenv egy többek közt használt Node.js csomag, ami elérhetővé teszi a környezeti változók kezelését .env fájl segítségével. A dotenv a token könnyebb kezelését oldja meg.

## Használata a szakdolgozatomban

Szakdolgozatomban a dotenv csomagot használtam az érzékeny adatok, például az adatbázis hitelesítési adatai védelme érdekében .env fájlban definiáltam ezeket a változókat:



Dotenv csomag betöltése és a változók beolvasása így történt



Ez a megoldás biztosítja, hogy az érzékeny adatok ne kerüljenek be közvetlenül a forráskódba, és könnyebben, gyorsabban módosíthatók legyenek különböző helyzetekben, pl;. kitudódik a .env fájl.

## Biztonság és autentikáció

Ezek az eszközök segítenek a jelszavak biztonságos tárolásában és a felhasználók azonosításában.

## Argon2

Az argon2 egy modern és biztonságos jelszó védelmet nyújtó algoritmus, amelyet elég sok helyen használnak titkosítás szempontjából.

A bcrypt alternatívájaként fejlesztették ki, leginkább támadások ellen nyújt előnyt.

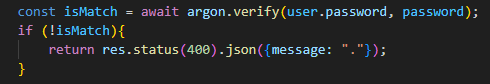
## Használata a szakdolgozatomban

Szakdolgozatom kódolása során az argon2 csomagot használtam a felhasználói jelszavak biztonságos tárolására. A jelszó hashelése az alábbi módon történt:



A metódus egy erős, egyirányú hasht generál a megadott jelszóból, amely biztonságosan tárolható az adatbázisban, ezek általában számok, betűk és speciális karakterek összessége ami randomizált..

Bejelentkezés során a felhasználó által megadott jelszó ellenőrzése az alábbi módon történt

:

A függvény ellenőrzi, hogy a felhasználó által megadott jelszó megegyezik-e az adatbázisban tárolt hashelt jelszóval.

## JWT használata

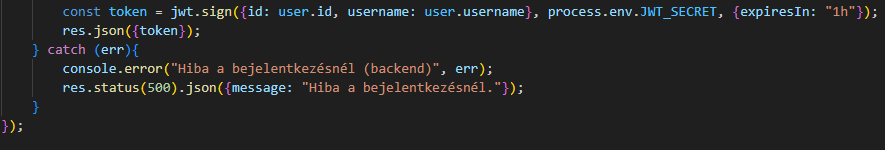
A JSON Web Token (JWT) egy nyílt szabvány (RFC 7519), amelyet az információk biztonságos továbbítására használnak.

A JWT lehetővé teszi az autentikációt és sok mást.

## Használata a szakdolgozatomban

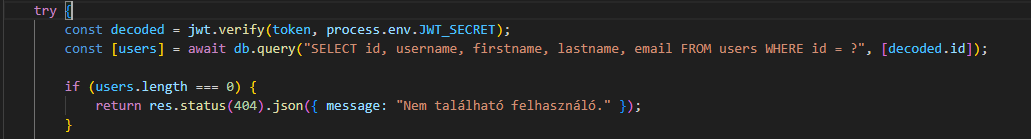
A projekt során a JWT-t az autentikációhoz használtam, hogy a felhasználók biztonságosan bejelentkezhessenek és ellenőrizve, illetve száz százalékosan hitelesítve legyenek a rendszerben.

A tokeneket igy generáltattam:



Ez a kód egy JWT tokent hoz létre, amely tartalmazza a felhasználó azonosítóját, és egy óráig érvényes csak, ez nem minden esetben jó, sokan rövidebb időtartamot adnak meg de van ennek fordított verziója is, amikor több időt adnak meg.

A szerver-oldali hitelesítés során a JWT ellenőrzése így történt



Biztosítja, hogy a beérkező kérésekben lévő token érvényes és megbízható legyen.

## CORS használata

A CORS (Cross-Origin Resource Sharing) egy biztonsági mechanizmus, amely lehetővé teszi a különböző domain-ek közötti erőforrás-megosztást. Alapértelmezés szerint a böngészők blokkolják a különböző domain-ek közötti kéréseket, de a CORS middleware segítségével ezt fel lehet oldani.

## Használata a szakdolgozatomban

CORS middleware-t az Express.js segítségével alkalmaztam, hogy lehetővé tegyem a frontend és a backend közötti kapcsolatot, ez például a headernek az összekötésében mutatkozik meg.:





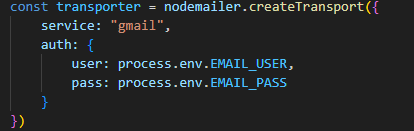
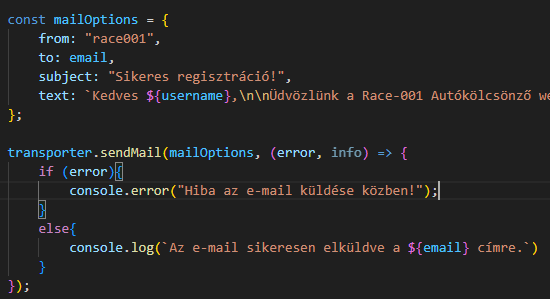
Ezzel biztosítottam, hogy a frontend képes legyen kommunikálni a backenddel, anélkül hogy a böngésző megakadályozná a requesteket.

## Nodemailer használata

A Nodemailer egy népszerű Node.js csomag, amely lehetővé teszi e-mailek küldését SMTP protokollon keresztül.

Segítségével könnyedén konfigurálhatunk e-mail szolgáltatást az alkalmazásunkban.

## Használata a szakdolgozatomban

Weboldalamban a Nodemailer segítségével valósítottam meg az e-mail értesítések küldését a felhasználóknak. Ez a kódrészlet mutatja, hogy hogyan állítottam be az e-mail küldést:

## React használata

A React egy JavaScript könyvtár, amelyet interaktív és dinamikus felhasználói felületek létrehozására használnak.

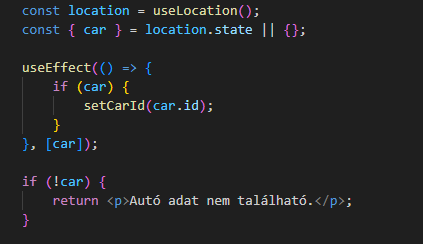
Komponensalapú architektúrája lehetővé teszi az újra felhasználható UI elemek kialakítását, ami megkönnyíti a fejlesztést és a karbantartást.

## Használata a szakdolgozatomban

A weboldalam frontendjének fejlesztéséhez a Reactot használtam, mivel hatékonyan kezeli az állapotváltozásokat és gyors felhasználói élményt biztosít.

Az alkalmazás fő komponensekre van bontva, amelyek felelősek az egyes UI elemek és funkciók megjelenítéséért.

Ebben az adott példában egy könnyebb React komponens látható, amely egy autó adatait vizsgálja.



A React használata lehetővé tette számomra, hogy egy gyors, dinamikus és élményt nyujtó frontend rendszert építsek a weboldalhoz.

## Összegzés a technológiák kihasználásáról

A háttér fejlesztéséhez az Express.js keretrendszert használtam, amely lehetővé teszi API végpontok létrehozását, adatok kezelését és ügyfél kéréseinek kiszolgálását. Az adatokat MySQL adatbázisban tárolom és ehhez a mysql2/promise csomagot használom a lekérdezések és műveletek aszinkron és gyors kezelésére. A bejelentkezések és az adatkezelés biztonságának megőrzése érdekében argon2 algoritmussal titkosítom a jelszavakat, és JSON Web Tokens (JWT) segítségével hitelesítem a felhasználókat, biztosítva, hogy csak az arra jogosult (adminok) személyek férhessenek hozzá a különböző funkciókhoz. Az API megfelelő működése és az adatok biztonságos kezelése érdekében más eszközöket is használok, mint például a CORS, amely lehetővé teszi a kapcsolatok közötti requesteket, és a dotenv csomag adatok tárolására a változókban.

A másik kulcseleme a frontend, amelyet React.js segítségével fejlesztettem. A React egy komponens alapú JavaScript-könyvtár, amely lehetővé teszi újra felhasználható felhasználói felület-elemek létrehozását és a felhasználói élmény optimalizálását. A front-end felelős a tartalom megjelenítéséért a felhasználó számára, valamint a szükséges információk lekéréséért és a háttérbe való elküldéséért. Az alkalmazás reszponzív és rugalmas működését a React állapotkezelés és a React Router hozzájárulásával biztosítottam, lehetővé téve az oldalak közötti gyors navigációt és az adatok valós idejű frissítését.

E-mail értesítésekhez és a fiók megerősítéséhez a Nodemailer csomagot használtam, amely lehetővé teszi az automatikus e-mailek küldését a regisztrációs és foglalási folyamat során.

Összességében a háttér- és az előtér-technológiák kombinációja biztosítja, hogy a weboldal gyors, biztonságos legyen. Az Express.js és a MySQL segítségével hatékonyan kezeltem az adatokat, míg a React.js felhasználói felületek létrehozását tette lehetővé. Különféle biztonsági intézkedések és szoftverek alkalmazásával adatvédelmet és hitelesítést biztosít, hogy a felhasználók biztonságosan használhassák a weboldal.

# Forráskód

## Fontosabb forráskódok és programozási modell bemutatása

Számunkra fontos szerepet játszott a tisztakód elve , eszerint terveztük és valósítottuk meg az oldalunkat. Mindennek ott a helye, ahol kell, hogy legyen.

## GET, POST, PUT

**GET**: Adatok lekérése a szerverről (pl. egy lista betöltése).

**POST**: Új adat küldése a szerverre (pl. egy új felhasználó regisztrálása).

**PUT**: Meglévő adat módosítása a szerveren (pl. egy felhasználói profil frissítése).

## Hookok

**useState**: Állapot kezelése egy komponensen belül.

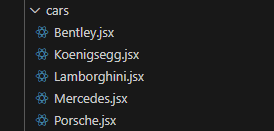
**useEffect**: Oldalhatások kezelése (pl. adatlekérés, eseménykezelés).

**useContext**: Globális állapot megosztása a komponensek között.

**useRef**: DOM-elemek vagy egyéb értékek hivatkozása.

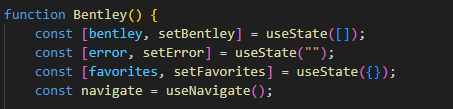
## Cars (autók) mappa bemutatása

A mappa elhelyezkedése a fejlesztői oldalon található, ehhez csak a fejlesztők férhetnek hozzá.

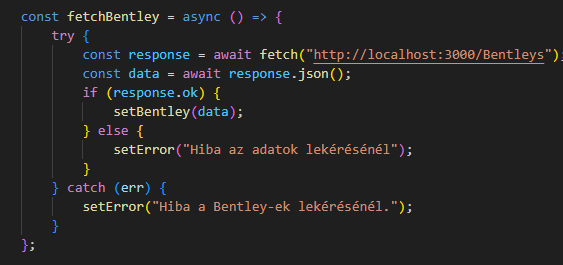


Az összes brand saját névre szóló JSX fájlt kapott ezzel könnyítve a hivatkozásokat és az átláthatóságot.

## Bentley.jsx

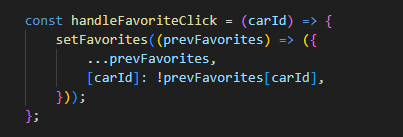
Ez a React-komponens egy Bentley autókat listázó oldalt valósít meg, ahol a felhasználók megtekinthetik az elérhető járműveket, kedvencekhez adhatják őket, és bérlést kezdeményezhetnek.

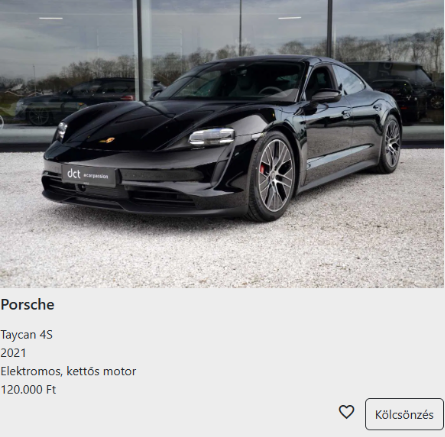
## Komponens létrehozása és állapotkezelés



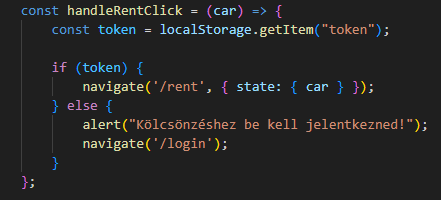
Ez a függvény lekéri az autókat a szerverről. Ha sikeres, akkor a **setBentley(data)** segítségével beállítja az autókat, ha pedig hiba történt a **setError()** függvény egy hibaüzenetet tárol el.

## Kedvenc gomb kezelése



Ez a függvény kezeli a szív ikonra való kattintást, amely egy autót kedvencként jelöl meg vagy távolít el a kedvencek közül.

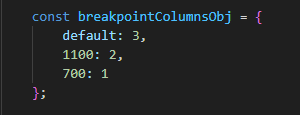
## Kölcsönzés gomb kezelése



Ha a felhasználó a **Kölcsönzés** gombra kattint, az alkalmazás először ellenőrzi, hogy van-e bejelentkezett felhasználó (localStorage.getItem("token")).

Ha igen, akkor a kölcsönzési oldalra dob az adott autó adataival. Ha nincs bejelentkezve, egy figyelmeztető üzenetet kap, és átirányítja a bejelentkezési oldalra.

## Reszponzív Masonry Grid beállítása



Ez határozza meg, hogy különböző képernyőméreteken hány oszlopban jelenjenek meg az autók. Ez attól függ, hogy milyen eszközt használunk.

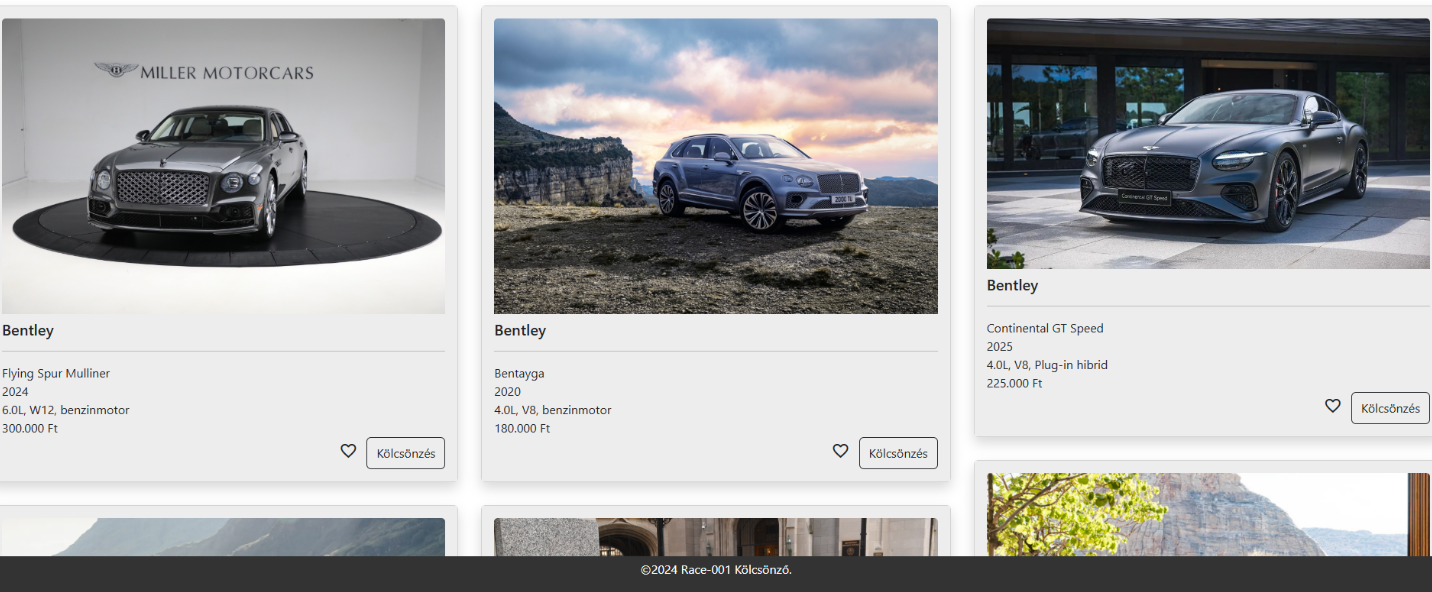
**Renderelés**



Ez jeleníti meg az autók listáját, a kedvenc gombot és a kölcsönzési lehetőséget.

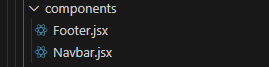
## Összegzés

Ez a React-komponens egy **Bentley autókat listázó oldal**, amely tartalmaz adatlekérést, kedvencek kezelését, kölcsönzési navigációt és reszponzív elrendezést. A felhasználók könnyen böngészhetik és kezelhetik az autókat.



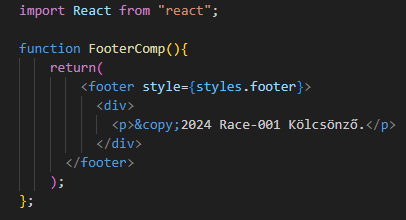
## Components (Komponensek) mappa bemutatása

Ebben a mappában találhatóak azok a komponensek melyek a navbart és a footert kezelik.

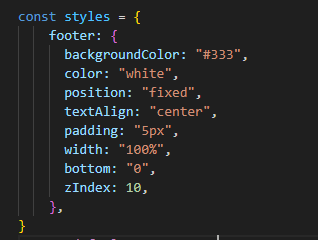


## Footer.jsx

A **FooterComp** egy egyszerű React-komponens, amely az oldal lábrészét biztosítja. A komponens egy <footer> elemet tartalmaz, amely sötétszürke háttérrel és fehér szöveggel rendelkezik. A lábléc fixen helyezkedik el az oldal alján ami görgetéskor is látható marad.

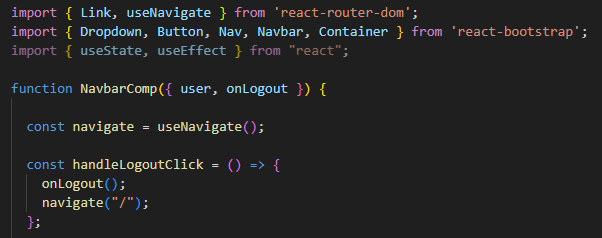


A következő CSS-stílusokat alkalmazza:

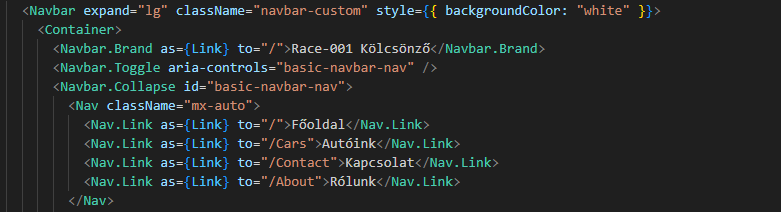


## Navbar.jsx

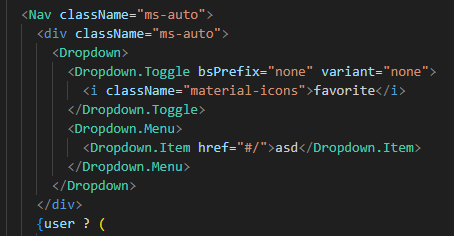
A **NavbarComp** egy komplexebb React-komponens, amely egy Bootstrap alapú navigációs sávként működik. A react-router-dom csomag useNavigate hookját alkalmazza az oldalak közötti navigációhoz.



A kódrészlet (NavbarComp komponens) tartalmaz egy nevet (Race-001 Kölcsönző), amely kattintható linkként működik, valamint több menüpontot: **Főoldal**, **Autóink**, **Kapcsolat**, **Rólunk**.



Továbbá, a navigációs sáv tartalmaz egy kedvencek ikont (<i className="material-icons">favorite</i>), amely egy elnyíló menüt nyit meg, ahol a felhasználók kedvenc, (tehát amiket besziveztek) autóikat jeleníti meg.



Ha a felhasználó be van jelentkezve, egy további ikon (inventory) is elérhető, amely a kölcsönzött autókat jeleníti meg.

A profil elérése egy másik Nav.Link elemként van megvalósítva, amely a bejelentkezett felhasználóknak elérhető csak.

Ha a felhasználó nincs bejelentkezve, a navigációs sáv a **bejelentkezés** és **regisztráció** buttonokat mutatja.

A kijelentkezési folyamat egy handleLogoutClick függvény segítségével történik, ami törli a felhasználói adatokat és visszaküldi a főoldalra (navigate("/")). A bootstrap navbar, dropdown, container és button elemek biztosítják a reszponzivitást és az esztétikus megjelenést, így a navigációs rész minden eszközön megfelelően működik és alkalmazkodik a különböző kijelzőméretekhez.



## Profile.jsx

Ez egy React komponenst használ, amely a felhasználói profil szerkesztését és megtekintését teszi lehetővé, lehetőséget biztosít a felhasználónevek, e-mail címek és jelszavak frissítésére.

A fetchUserData függvény a felhasználói adatok lekérésére szolgál, és az alábbi módon történik:



A fetchUserData függvény a jwtből veszi a tokent, és ha az érvényes, egy GET kérést küld a backendnek. Ha a kérés sikeült, a választ (felhasználói adatokat) a setUser segítségével tároljuk, majd a form mezőket töröljük. Ha valami hiba történik, a hibát dobó állapotot állítunk be.



## Profil szerkesztés

A profil frissítése akkor történik, amikor a felhasználó rákattint a "Mentés" gombra. Az updateProfile függvény nézi a profil adatainak változtatását, szerkesztését.

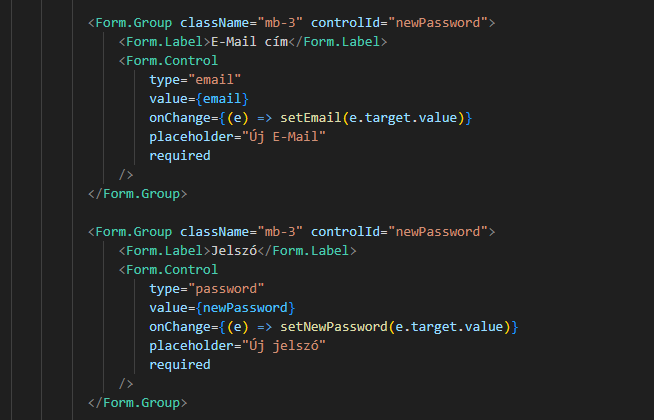
Ez a függvény elküld egy PUT kérést a backendnek, amely tartalmazza a saját igényekre változtatott felhasználónevet, e-mail címet és jelszót.



Az updateProfile függvény először megnézi, hogy létezik-e érvényes token. Ha nem, hibaüzenetet küld. Ezután egy PUT kérést küld a backendnek, amely tartalmazza a módosított adatokat. A válasz függvényében a „React” visszajelzést ad a felhasználónak: ha a kérés sikeres akkor sikeres üzenetet jelenítünk meg, ha nem viszont akkor hibát.

## E-mail cím és jelszó módosítása

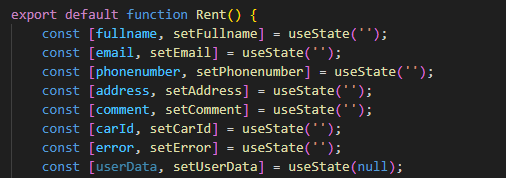
Az e-mail cím és jelszó módosítása a form mezők segítségével történik. Az e-mail cím az email állapotban tárolódik, a jelszó pedig a newPassword állapotban. Az alábbi kódrészletek mutatják be, hogy hogyan kezeljük a jelszó és e-mail cím frissítését:



A felhasználó az e-mail cím és jelszó mezőbe írja be az új adatokat. Az onChange metódus updateli az email és newPassword állapotokat, és a form elküldésekor az új adatokat a backendnek továbbítjuk.

## Rent.jsx

Több useState hook található, amelyek a felhasználói adatokat legfőképp a név, email cím, telefonszám, cím és megjegyzés, a hibákat és a rendeléshez szükséges adatokat tárolják.



## Felhasználói adatok lekérdezésének folyamata

A fetchUserData függvény letölti a felhasználó adatait a backendből. Az érvényességet a localStorage-ban tárolt jwt token biztosítja.

Ha a jwt token nem található, a felhasználó nem lesz bejelentkezve, és hibaüzenetet fog kidobni, erről a másik fél is tudomást fog kapni.



## Rendelés leadásának folyamata

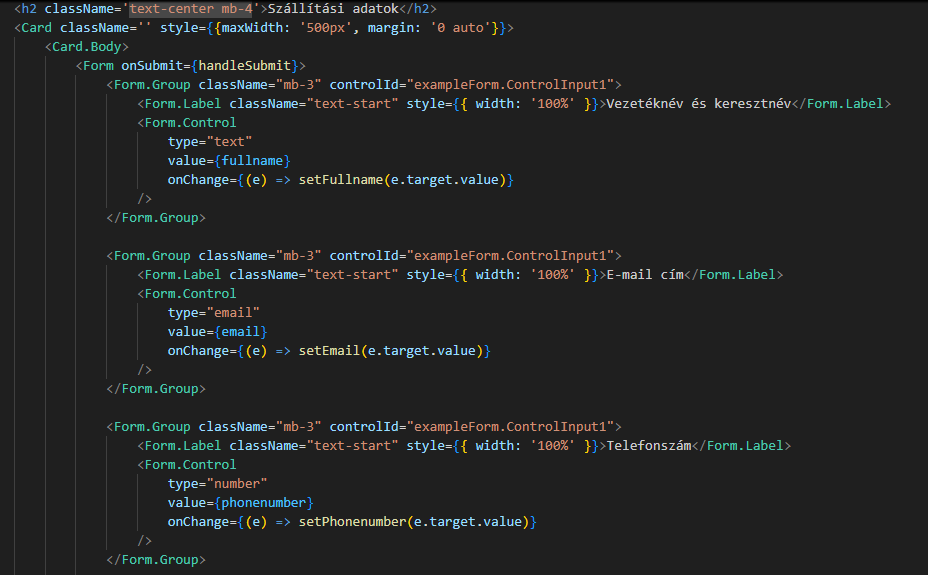
A rendelési form kitöltése után a felhasználó rákattint a "Küldés" gombra, amely hivatkozik a handleSubmit függvényre, ami ha sikeres, és bevan jelentkezve a felhasználó akkor a rendelését letudja adni.

Ez a függvény ellenőrzi, hogy minden kötelező mező ki van-e töltve, és ha igen, akkor a rendelési adatokat egy POST kérést küld.



## Formázás

A jobb oldali rész tartalmazza a rendelési formot, amely elérhetővé teszi a felhasználók számára, hogy megadják a szükséges szállítási adatokat, mint például nevüket, e-mail címüket, telefonszámukat és címüket



# Tesztelés

Társammal a weboldal fejlesztése közben sok leginkább kisebb de igen érzékeny hibába ütköztünk, az én perspektívám szerint sikerült sok hibát kijavítanunk viszont biztosan maradt pár hiba ami egy hosszabb idő folyamán amikor az oldal használva van egyszerre húsz avagy harminc ember által akkor jelennének meg akár terhelés akár bármilyen betöltési és backend hiba végett.

A tesztelésekhez a fekete dobozos illetve a fehér dobozos tesztelési módszereket használtam.

## Hibák

A tokenek érvényességeinek ellenőrzése szempontjából a fentebb említett „fetchUserData” függvény nem annyira tökéletes, csak egyszer van ellenőrizve hogy a token lejárt-e vagy sem, illetve a backend-ben ez nincs megvalósítva, tehát a backend-ben nem ellenőrzi, hogy lejárt avagy még érvényes-e.

Ez a hiba kihat a biztonságra, és jogosulatlan hozzáférést is eredményezhet.

A mezők (form) érvényesítése szempontjából amikor egy felhasználó leadna egy rendelést akkor különböző mezőket kell, hogy kitöltsön viszont ha nem tölti ki az egyiket akkor is le tud adni rendeléseket.

Ez a hiba túl tudja terhelni az oldalt ennek következtében összeomolhat az oldal.

Regisztrációkor illetve amikor a felhasználó változtatni szeretné profiljának jelszavát akkor nincs ellenőrizve, hogy érvényes karaktereket adott-e meg, vagy esetleg a mező nem üres.

Az adott hiba azért veszélyes mivel a felhasználónak nem kell, hogy megadjon jelszót azért, hogy be tudjon jelentkezni az oldalra.

A rendelés oldalon fennáll a placeOrder hiba ami azt jelenti, hogy ha két ugyan olyan keresztnevű és vezetéknevű (tehát a validáció csak a fullname alapján működik) akkor az adatbázis különféle hibákat dobhat.

A placeorder hiba az átláthatatlanságot illetve félreértések vonhat maga után. Kihatással lehet az adatbázisban történő adatok feldolgozásában, és a rendszer false információkat kaphat.

# Továbbfejlesztési lehetőségek

A weboldalunk fejlesztésekor mindig ahogy haladtunk előre úgy jöttek a következő ötletek párat megvalósítottunk párat a sorsra bíztunk mert nem tudjuk előre, hogy mit hoz a jövő. A weboldal szerintünk elérte azt a pontot, hogy megállja a helyét egy megbízásnál pár apróbb simítással és fix konkrétumokkal.

Hogyha a sors úgy hozza akkor a következőket szeretnénk lefejleszteni a jövőben;

A fizetési rendszer véglegesítése akár PayPal vagy bármilyen fizetőeszköz segítségével.

Angol és francia nyelv lefejlesztése, hogy a felhasználó tetszés szerint választhassa ki, hogy éppen milyen nyelven szeretne oldalunkon tájékozódni. A jobb, avagy a bal felső sarokban erre egy legördülő lista jelenne meg és ott tudná kiválasztani az adott nyelvet, ezzel nemzetközivé tenni az oldalt, ez bővíthető akár.

Utánvét elérhetővé tétele és ennek a rendszernek megálmodása mivel sokan nem online fizetés orientáltak.

## Összegzés

A kezdet könnyű volt, és izgalmas de akadálymentes is, ezt sokak nem tudják elmondani. Számomra a partnerem kiválasztása a legegyszerűbb volt, a téma kiválasztásával együtt.

Kucsik Gábor partnerem elképesztő munkát végzett, mint a frontend és a backend részét beleértve, nem volt felosztva hogy ki mit csinál, mindkettőnk ott járult hozzá ahol kellett, élmény volt vele együtt dolgozni és egy mondhatni álmunkat megvalósítani. Szerintem elmondhatom, hogy mind a kettőnk az apait és anyait beletette az elejétől fogva. Egyetlen egy nagyobb akadályunk volt amikor az oldalunkat teljesen át kellett kódolni és dolgozni viszont egy ilyen partner mellett ez eltörpült. Sokat tanultam, mint a kódolás, a tervezés és megvalósítás szempontjából. Amit tanultam az a munkamegosztás, az hogy hogyan kell együtt dolgozni egy csapattárssal, illetve, hogy kitartónak kell legyél és amit megálmodsz azt meg kell valósitani.

A programozásnak számos új oldalát ismerhettem és tanulhattam meg, a node.js használata könnyebbé tette a webszervereknek létrehozását, a React előre segítette a zökkenőmentességet a felhasználói felületek terén. A CSS és a Bootstrap nagy szerepet játszott a design megtervezésében és megvalósításában, és a reszponzivításban. A MySQL használata bővítette az adatok lekérdezése és feldolgozása terén a tudásomat.

# Forrás

## Képek

 <https://www.bentleyofgreenwich.com/imagetag/7165/main/l/New-2024-Bentley-Flying-Spur-Mulliner-W12.jpg>

 <https://d3jvxfsgjxj1vz.cloudfront.net/news/wp-content/uploads/2020/07/01101658/Image-1-The-New-Bentayga-Front.jpg>

 <https://carsguide-res.cloudinary.com/image/upload/f_auto,fl_lossy,q_auto,t_default/v1/editorial/review/hero_image/2025-Bentley-Continental-GT-Speed-Anthracite-Satin-1001x565.jpg>

 <https://media.autoexpress.co.uk/image/private/s--X-WVjvBW--/f_auto,t_content-image-full-desktop@1/v1611675481/autoexpress/2021/01/Bentley%20Flying%20Spur%20V8.jpg>

 <https://www.goldcoastautogallery.com/galleria_images/11271/11271_p2_l.jpg>

 <https://www.amlu.com/wp-content/uploads/2020/08/2021-bentley-bentayga-speed-keeps-fastest-suv-crown9-730x450.jpg>

 <https://editorial.pxcrush.net/carsales/general/editorial/bentley-continental-gt-mulliner-08.jpg?width=1024&height=682>

 <https://hips.hearstapps.com/hmg-prod/images/2022-bentley-flying-spur-hybrid-545-1659030974.jpg?crop=0.910xw:0.681xh;0.0897xw,0.297xh&resize=1200>:\*

 <https://hips.hearstapps.com/hmg-prod/images/2023-bentley-continental-gt-s-coupe-101-1654526518.jpg?crop=0.767xw:0.648xh;0.0651xw,0.220xh&resize=2048>:\*

 <https://da4dkroembtou.cloudfront.net/wp-content/uploads/2020/03/BL4A9053-NEW00-Custom.jpg>

 <https://i.pinimg.com/originals/29/4a/d0/294ad097c1f9a1b0431e0b818e907691.jpg>

 <https://img.hvg.hu/Img/b2dea50fcee14f6eb810034566fbfb2e/4cdb0991-e1b3-4259-a2ad-680dc0a225e5.jpg>

 <https://elektromosauto.hu/wp-content/uploads/2020/03/Koenigsegg-Gemera-01.jpg>

 <https://www.supercars.net/blog/wp-content/uploads/2019/04/Jesko2.jpg>

 <https://www.supervettura.com/blobs/Cars/48/5a8c9adf-653b-49ba-b3c4-9d03f0f96158.jpg?width=1920&height=1080&mode=crop>

 <https://www.europeanprestige.co.uk/blobs/stock/343/images/6bfb3b2d-4f88-47ad-9d2d-c63bc36cbd0d/hi4a1842.jpg?width=2000&height=1333>

 <https://media.autoexpress.co.uk/image/private/s--X-WVjvBW--/f_auto,t_content-image-full-desktop@1/v1620898674/autoexpress/2021/05/Mercedes%20S-Class%202021%20UK-18.jpg>

 <https://www.topgear.com/sites/default/files/images/news-article/2020/11/86d1c7742a1e093a83487debca97eecd/20c0535_179.jpg>

 <https://media.ed.edmunds-media.com/mercedes-benz/eqs/2022/oem/2022_mercedes-benz_eqs_sedan_amg-eqs_fq_oem_1_1600.jpg>

 <https://media.ed.edmunds-media.com/mercedes-benz/g-class/2021/oem/2021_mercedes-benz_g-class_4dr-suv_amg-g-63_fq_oem_1_1600.jpg>

 <https://www.motortrend.com/uploads/sites/5/2020/12/2021-Mercedes-Maybach-GLS-600-4Matic-11.jpg>

 <https://fotografias.larazon.es/clipping/cmsimages01/2020/07/31/26E852AC-2051-454C-96BB-AA368DC33DBB/98.jpg?crop=787,443,x0,y74&width=1900&height=1069&optimize=low&format=webply>

 <https://cdn.elferspot.com/wp-content/uploads/2021/01/porsche-taycan-1726-xxl_kfz69307501_dsc_8071..jpg?class=xl>

 <https://hips.hearstapps.com/hmg-prod/images/2020-porsche-panamera-turbo-mmp-1-1579209005.jpg?crop=0.521xw:0.694xh;0.00651xw,0.240xh&resize=768>:\*

 <https://player.hu/uploads/2021/07/porsche-cayenne-coupe-turbo-gt-1.jpg>

 <https://robbreport.com.au/application/assets/2021/05/1-13-e1621472555244.jpeg>

 <https://static0.carbuzzimages.com/wordpress/wp-content/uploads/gallery-images/original/674000/700/674788.jpg>

 <https://images-porsche.imgix.net/-/media/4824F0B3C7B742CE8E672F804BE2BDFB_8B83A23F3C8B4626AE3AB02E47D1E84A_CM21N3KOX0005-cayman-gts-40-driving?w=2560&h=1441&q=45&crop=faces%2Centropy%2Cedges&auto=format>

 <https://www.lamborghinigoldcoast.com/imagetag/9874/2/l/New-2022-Lamborghini-Aventador-LP-780-4-Ultimae-1647974877.jpg>

 <https://media.autoexpress.co.uk/image/private/s--X-WVjvBW--/f_auto,t_content-image-full-desktop@1/v1632415640/autoexpress/2021/09/Lamborghini%20Huracan%20STO%202021%20review-5.jpg>

 <https://cdn-ds.com/blogs-media/sites/94/2022/12/18031729/Front-view-of-the-Lamborghini-Sian-FKP-37-parked-A_o.jpg>

 <https://www.europeanprestige.co.uk/blobs/stock/160/images/05866226-b060-40b7-b027-93d2bf928528/1j6a9063-edit.jpg?width=2000&height=1333>

 <https://imageio.forbes.com/blogs-images/kbrauer/files/2019/07/2020-Lamborghini-Huracan-Evo-Spyder-Driving-Front.jpg?format=jpg&height=900&width=1600&fit=bounds>

 <https://i.insider.com/605a2b7a106eb50019d052a0?width=700>

 <https://img.hvg.hu/Img/b2dea50fcee14f6eb810034566fbfb2e/8804156a-b4ac-4d04-a48c-f73d9fab27fa.jpg>

 <https://hips.hearstapps.com/hmg-prod/images/2020-bmw-x5-m-104-1582911121.jpg?crop=0.721xw:0.662xh;0.256xw,0.217xh&resize=2048>:\*

 <https://hips.hearstapps.com/hmg-prod/images/2022-bmw-i4-m50-114-1633973115.jpg?crop=1.00xw:0.846xh;0,0&resize=2048>:\*

 <https://media.carsandbids.com/cdn-cgi/image/width=2080,quality=70/9004500a220bf3a3d455d15ee052cf8c332606f8/photos/3BkQBnZ5-MmdMArb6Ld-(edit).jpg?t=166922317584>

 <https://img.autotrader.co.za/13324616>

 <https://cdn.bimmertoday.de/wp-content/uploads/2021/09/Fahrbericht-BMW-iX-2022-Elektro-SUV-01-750x500.jpg>

 <https://hips.hearstapps.com/hmg-prod/images/pagani-huayra-roadster-bc-101-1634741760.jpg?crop=0.655xw:0.983xh;0.160xw,0&resize=1200>:\*

 <https://news.dupontregistry.com/wp-content/uploads/2021/11/Pagani-main.jpg>

## Backend

https://www.npmjs.com/

https://expressjs.com/

<https://sidorares.github.io/node-mysql2/docs>

## Frontend

https://react.dev/

https://getbootstrap.com/

https://react-bootstrap.netlify.app/

## Adatbázis

https://mariadb.com/

https://www.w3schools.com/mysql/mysql\_sql.asp

https://dev.mysql.com/doc/mysql-tutorial-excerpt/5.7/en/examples.html

<https://www.interviewbit.com/blog/mysql-commands/>

## Egyéb források

Denkinger Ákos tanárúr, Molnár Máté tanárúr, Ruga Rebeka tanárnő órai munkáit felhasználtam a szakdolgozatban.

## Fordítók

https://translate.google.com/

https://www.deepl.com/en/translator