**1. Bevezetés**

A **B&K Autókereskedés** egy webes alkalmazás, amelynek célja, hogy megkönnyítse a használt autók adásvételét a felhasználók számára. Az oldal segítségével a látogatók böngészhetnek a jelenlegi autókínálatban, részletes szűrőfeltételek segítségével megtalálhatják a számukra legmegfelelőbb járművet, és akár regisztrációt követően kedvencek listáját is összeállíthatják. A platform lehetővé teszi az autók lefoglalását, amelyhez mindössze a gépjármű árának 10%-át kell kifizetni előlegként.

Ez a dokumentáció átfogó képet ad a projekt felépítéséről, működéséről, valamint a használt technológiákról és komponensekről. A dokumentáció elsősorban a fejlesztők és az üzemeltetők számára nyújt információkat, de a jövőbeni fejlesztésekhez és karbantartáshoz is útmutatóul szolgál.

**2. Áttekintés és Architektúra**

A projekt két fő részből áll:

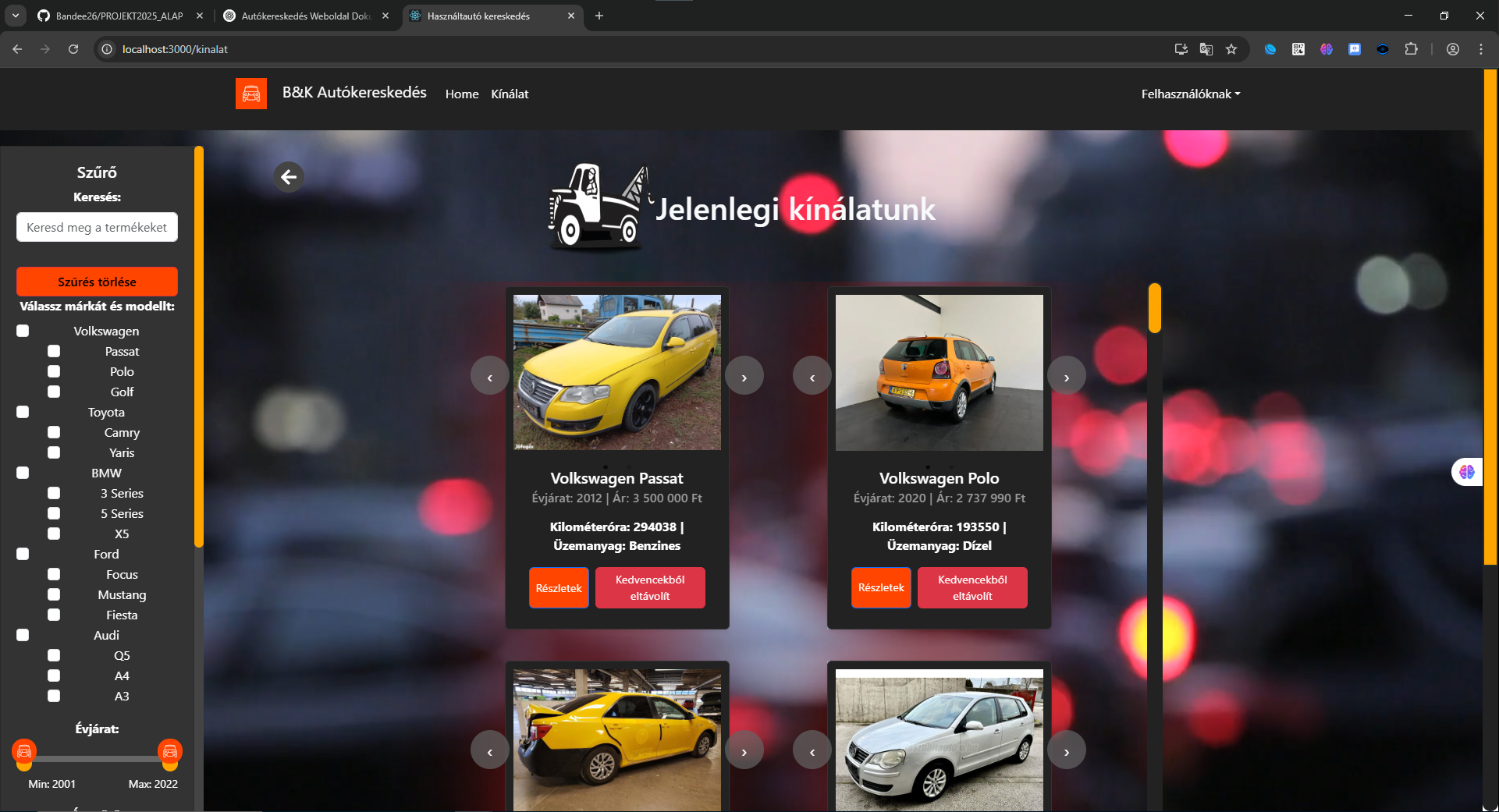
1. **Frontend** – React keretrendszerben íródott, modern JavaScript/JSX technológiákat alkalmazva.
2. **Backend** – Express.js alapú Node.js alkalmazás, amely REST API-t biztosít a frontend számára.

A két réteg közötti kommunikáció JSON formátumú üzenetekkel történik. A biztonság és a skálázhatóság érdekében a backend JWT (JSON Web Token) alapú hitelesítést használ, míg a frontend Axios segítségével küldi a kéréseket az API végpontjaira.

**2.1. Főbb Funkcionalitások**

* **Autók böngészése**: A felhasználók áttekinthetik a rendelkezésre álló autók listáját, beleértve a képeket, árakat, évjáratot és egyéb információkat.
* **Részletes szűrés**: Kereshetnek márka, ár, évjárat, futásteljesítmény és egyéb paraméterek alapján.
* **Autó adatlap**: A részletek oldalon további információk találhatók (például eladó adatai, jármű extrái, motor típusa).
* **Regisztráció és bejelentkezés**: A regisztrált felhasználók kedvenc autókat gyűjthetnek össze, foglalhatnak, és módosíthatják személyes adataikat (például jelszó, e-mail).
* **Foglalás**: A kiválasztott autók 10%-os előleg befizetésével lefoglalhatók, akár személyesen, akár átutalással.
* **Kapcsolatfelvétel**: Akik eladni szeretnének autót, a weboldalon megadott e-mail címen vagy telefonszámon kereshetik fel a kereskedést.





**3. Backend Dokumentáció**

**3.1. Technológiák és Függőségek**

A backend a következő főbb függőségekre támaszkodik:

json

MásolásSzerkesztés

"dependencies": {

"cookie-parser": "~1.4.4",

"cors": "^2.8.5",

"debug": "~2.6.9",

"dotenv": "^16.4.7",

"express": "^4.21.2",

"googleapis": "^146.0.0",

"jsonwebtoken": "^9.0.2",

"morgan": "~1.9.1",

"mysql2": "^3.12.0",

"nodemailer": "^6.10.0",

"nodemon": "^3.1.9"

}

* **Express**: Alap webkeretrendszer a Node.js környezetben, amely segít az útvonalak és middleware-ek kezelésében.
* **cors**: A böngészők közötti erőforrás-megosztást teszi lehetővé, így a React alkalmazás biztonságosan tud kommunikálni a backenddel.
* **dotenv**: A környezeti változók (.env fájl) kezelésére, például adatbázis-hozzáférés vagy titkos kulcsok tárolására.
* **jsonwebtoken**: A JWT alapú hitelesítés megvalósításához szükséges.
* **mysql2**: MySQL adatbázis kapcsolat kezeléséhez használjuk, aszinkron módon.
* **nodemailer**: E-mailek küldését teszi lehetővé (például visszaigazolások, jelszóemlékeztetők).
* **nodemon**: Fejlesztés közbeni automatikus szerverújraindítás.

**3.2. Fájlszerkezet**

* **app.js**: A fő szerverkonfiguráció, ahol beállítjuk a middleware-eket (morgan, cookie-parser, cors stb.) és regisztráljuk az útvonalakat.
* **routes/index.js**: Alapértelmezett útvonal, egy egyszerű ellenőrző végpont, amellyel tesztelhető, hogy a szerver fut-e.
* **routes/termek.js**: Az autókhoz kapcsolódó útvonalak, pl. GET /termek, POST /termek/filter, GET /termek/page/:pageNo stb.
* **routes/user.js** (feltételezett vagy dboperations.js részeként): A felhasználókhoz kapcsolódó műveletek, pl. regisztráció, bejelentkezés, adatmódosítás.
* **db/dboperations.js**: Az adatbázis műveletek, például autók lekérdezése, felhasználók létrehozása, foglalások kezelése, bejelentkezés hitelesítés.

**3.3. Fontosabb API Végpontok**

1. **GET /termek**
   * Leírás: Visszaadja az összes elérhető autót.
   * Válasz formátum:

json

MásolásSzerkesztés

{

"products": [

{

"Auto\_ID": 1,

"Marka": "BMW",

"Modell": "X5",

"Evjarat": 2020,

"Ar": 15000000,

"Kilometerora": 20000,

"Motortipus": "Diesel",

"Szin": "Fekete"

},

...

]

}

1. **GET /termek/page/:pageNo**
   * Leírás: Az autók lapozott lekérdezése. Segít nagyobb adatbázis esetén kezelni a megjelenítendő adatmennyiséget.
   * Paraméter: :pageNo – a kért oldal sorszáma.
2. **POST /termek/filter**
   * Leírás: Autók szűrése a kérés törzsében megadott paraméterek alapján (márka, árintervallum, évjárat stb.).
   * Kérés példa:

json

MásolásSzerkesztés

{

"Marka": "BMW",

"Ar": [1000000, 20000000]

}

* + Válasz: A feltételeknek megfelelő autók listája.

1. **POST /bookings**
   * Leírás: Az autó lefoglalása. Elmenti az adatbázisba, hogy a felhasználó melyik autót és mikor foglalta le.
   * Megjegyzés: A 10%-os foglaló összege is rögzíthető a booking adatokban, valamint a felhasználó adatai.

**3.4. Hitelesítés és Engedélyezés**

A rendszer **JWT (JSON Web Token)** alapú hitelesítést használ:

* **Regisztráció**: A felhasználó megadja az adatait (név, e-mail, jelszó), amelyet a szerver eltárol.
* **Bejelentkezés**: Sikeres azonosítás esetén a szerver JWT tokennel válaszol, amelyet a frontend a böngészőben tárol (általában localStorage vagy sessionStorage).
* **Védett útvonalak**: Minden védett végpont megköveteli, hogy a kérés Authorization fejlécében szerepeljen a korábban kapott JWT token. A szerver ellenőrzi a token érvényességét, és csak ezután engedélyezi a műveletet (pl. foglalás).

**4. Frontend Dokumentáció**

**4.1. Technológiák és Függőségek**

A frontend React keretrendszerre épül, és a következő függőségeket használja:

json

MásolásSzerkesztés

"dependencies": {

"@headlessui/react": "^2.2.0",

"axios": "^1.7.9",

"bootstrap": "^5.3.3",

"cra-template": "1.2.0",

"flowbite-react": "^0.10.2",

"headlessui": "^0.0.0",

"mdb-react-ui-kit": "^9.0.0",

"query-string": "^9.1.1",

"react": "^19.0.0",

"react-bootstrap": "^2.10.7",

"react-dom": "^19.0.0",

"react-responsive": "^10.0.1",

"react-router": "^7.1.5",

"react-router-dom": "^7.1.5",

"react-scripts": "^5.0.1",

"react-slick": "^0.30.3",

"react-slider": "^2.0.6",

"react-spring": "^9.7.5",

"react-use": "^17.6.0",

"slick-carousel": "^1.8.1",

"web-vitals": "^4.2.4"

}

* **React**: A felhasználói felület (UI) megjelenítéséért felelős.
* **Axios**: A backend API-k meghívására, aszinkron kérések kezelésére.
* **Bootstrap / React-Bootstrap**: Egységes, reszponzív dizájnelemek és komponensek használata.
* **React-Router / React-Router-Dom**: Kliensoldali útvonalkezelés a többoldalas élmény biztosításához.
* **React-Spring**: Animációkhoz és látványos átmenetekhez.
* **react-slider**, **react-slick**, **slick-carousel**: Csúszkák, képgalériák és lapozható slider komponensek.

**4.2. Fájlszerkezet és Fő Komponensek**

* **app.js**: A fő belépési pont, ahol az útvonalak (router) konfigurálása történik, és összekapcsoljuk a különböző oldalakat, mint a *Home*, *Kinalat*, *Order* stb.
* **index.js**: A React alkalmazás indításáért felelős fájl (ReactDOM.render vagy createRoot), amely csatlakoztatja az alkalmazást a public/index.html-hez.
* **card.jsx**: Egyedi kártyakomponens az autók megjelenítéséhez. Tartalmazza a kép, cím, ár, gombok stb. megjelenítését.
* **creditcardmodal.jsx**: Egy felugró ablak, amely például a foglalásnál vagy fizetésnél használható, ha online tranzakcióval szeretnénk lefoglalni az autót.
* **footer.jsx**: Az oldal alján található lábléc komponens, ahol általában a kapcsolati információk, jogi nyilatkozatok és közösségi média linkek találhatók.
* **home.jsx**: A főoldal, ami rövid bemutatást ad a B&K Autókereskedésről, kiemelt ajánlatokkal vagy akciókkal.
* **imageslider.jsx**: Képek váltogatására (slider funkció), pl. főoldalon kiemelt autók, akciók megjelenítéséhez.
* **keplapozas.jsx**: Olyan komponens, amely segít a több képből álló galéria lapozásában (vagy oldalszám kezelésében).
* **kinalat.jsx**: A kínálat oldal, ahol listázva jelennek meg az elérhető autók. Axios-szal hívja meg a backend /termek végpontját, és kártyák formájában jeleníti meg az adatokat.
* **menu.jsx**: A navigációs menü (header), amely lehetőséget nyújt a felhasználónak különböző oldalak közötti váltásra (pl. Home, Kínálat, Profil, Kedvencek stb.).
* **order.jsx**: A foglalással és megrendeléssel kapcsolatos funkciók oldala, ahol a felhasználó véglegesítheti a kiválasztott autók foglalását, és kiválaszthatja a fizetési módot (átutalás vagy személyes befizetés).
* **szuro.jsx**: Szűrő komponens, amely lehetővé teszi, hogy a felhasználó ár, évjárat, márka, modell, futásteljesítmény stb. alapján leszűrje a kínálatot. A beállított paramétereket a backend felé küldi POST /termek/filter útvonalra.

Minden komponens saját **CSS** fájllal rendelkezik, amelyek a stílusokat és az esetleges reszponzív beállításokat tartalmazzák.

**4.3. Állapotkezelés és Adatlekérdezés**

* **React Hookok** (pl. useState, useEffect) segítségével kezeljük a komponensek állapotát, és frissítjük a megjelenítést, amikor változik a backendtől érkező adat.
* **Axios**: Az adatok lekérdezésére és küldésére szolgál. Például:

jsx

MásolásSzerkesztés

useEffect(() => {

axios.get("http://localhost:3001/termek")

.then(response => {

setAutok(response.data.products);

})

.catch(error => {

console.error("Hiba történt a termékek lekérdezése közben: ", error);

});

}, []);

* **React-Router**: A különböző oldalak (útvonalak) közötti navigációt oldja meg. Például:

jsx

MásolásSzerkesztés

<BrowserRouter>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/kinalat" element={<Kinalat />} />

<Route path="/order" element={<Order />} />

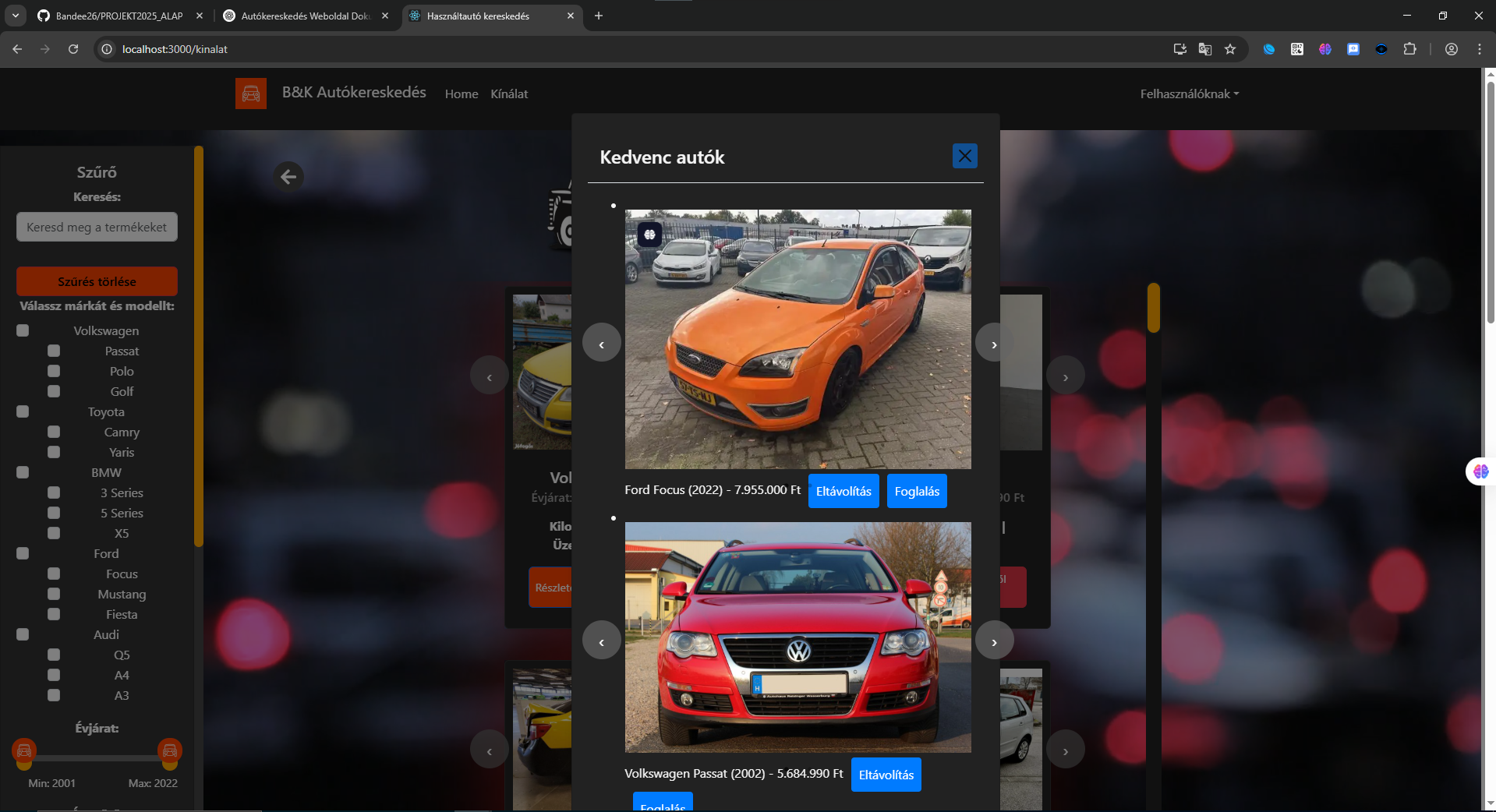
{/\* ...egyéb útvonalak \*/}

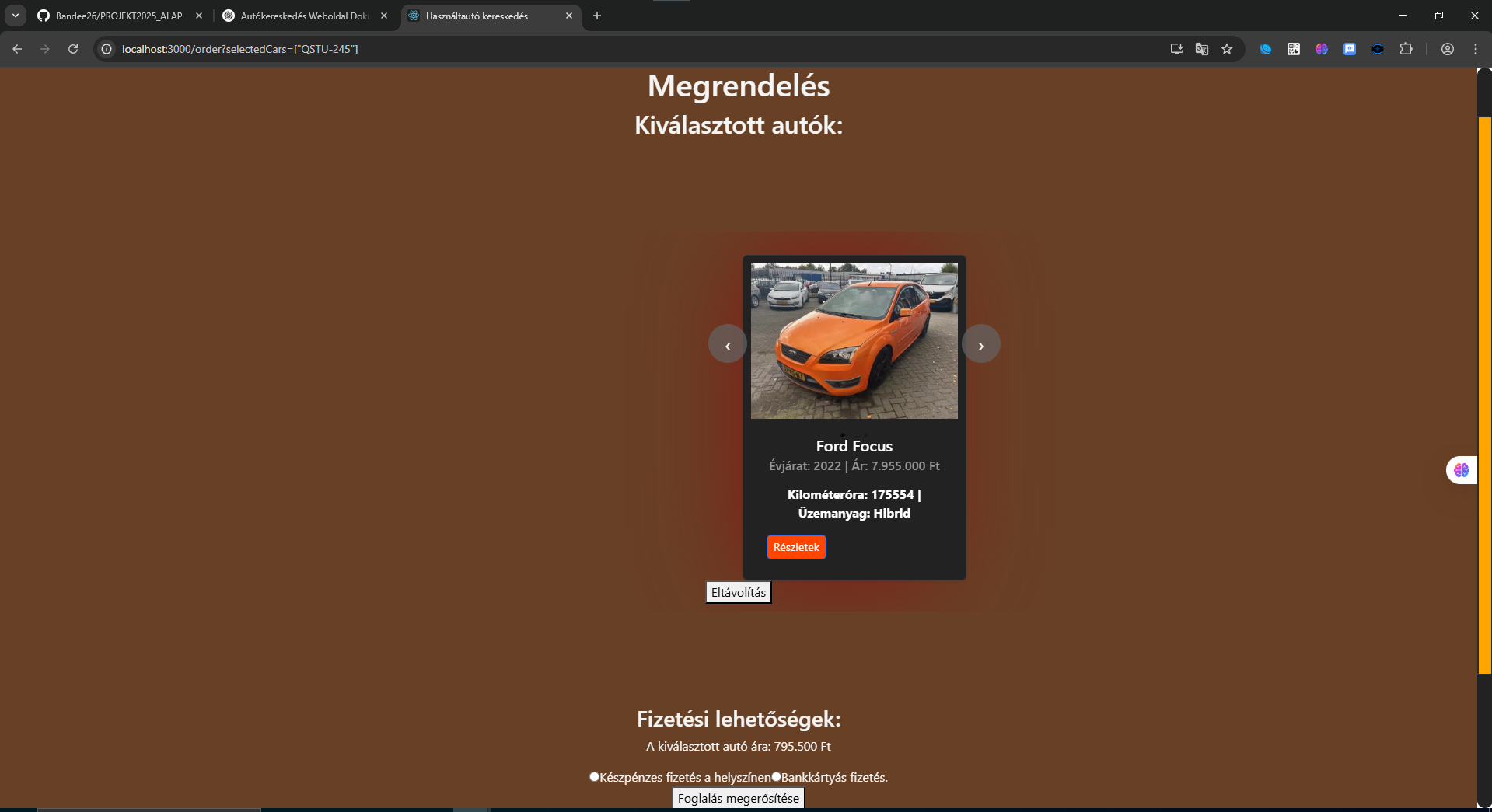
</Routes>

</BrowserRouter>

**4.4. Felhasználói Felület és Élmény**

* **Reszponzív Dizájn**: A Bootstrap és a saját CSS szabályoknak köszönhetően a weboldal mobilon, tableten és asztali gépen is kényelmesen használható.
* **Szűrőfelület**: Az ár- és évjáratcsúszkák (react-slider), a márka/model lekérdezés (axios) és egyéb paraméterek intuitív beállítását teszik lehetővé.
* **Kedvencek Lista**: A bejelentkezett felhasználó a kártyákon található gombbal hozzáadhatja az autót a kedvenceihez, amit később a profiloldalán megtekinthet.
* **Foglalási Folyamat**: A felhasználó a kiválasztott autót/kat lefoglalhatja, amelyről e-mail értesítést is kaphat, ha azt a backend oldalon Nodemailer segítségével konfiguráltuk.





**5. Regisztráció, Bejelentkezés és Felhasználói Fiókkezelés**

* **Regisztráció**: A felhasználó megadja a szükséges adatokat (név, e-mail, jelszó), majd a backend ellenőrzi, hogy létezik-e már az adott e-mail, és ha nem, létrehozza a felhasználót az adatbázisban.
* **Bejelentkezés**: Sikeres belépéskor a backend JWT tokennel válaszol. A frontend elmenti a tokent, és innentől kezdve a felhasználó jogosult a védett funkciók használatára (pl. foglalás, kedvencek, adatváltoztatás).
* **Adatmódosítás**: A bejelentkezett felhasználó megváltoztathatja a jelszavát, e-mail címét vagy más személyes adatát. Ehhez a frontend egy űrlapot biztosít, amelyet a backend PUT vagy POST metódussal kezel.

**6. Foglalási Rendszer**

A foglalás lényege, hogy a felhasználó 10% előleg befizetésével biztosíthatja, hogy a kiválasztott autót senki más ne vásárolja meg előtte. A fizetés történhet:

1. **Személyesen**: Ekkor a foglalási tranzakció rögzítve lesz az adatbázisban, és a felhasználó átveheti a gépjárművet a telephelyen.
2. **Átutalással**: A felhasználó banki átutalás útján küldi el az összeget. A backend ezután frissíti az autó státuszát “Foglalt” státuszra, amíg a felhasználó teljes egészében ki nem fizeti az autó árát.

A foglalás adatai (felhasználó, autó, foglalási idő, fizetési mód stb.) a **bookings** táblában vagy ennek megfelelő adatstruktúrában kerülnek tárolásra.

**7. Adatbázis Felépítés (Vázlatosan)**

Az alkalmazás MySQL adatbázist használ, és a következő táblákat (példa) tartalmazhatja:

1. **users**
   * id (PRIMARY KEY)
   * name
   * email (UNIQUE)
   * password
   * created\_at
   * updated\_at
2. **cars** (vagy termek)
   * Auto\_ID (PRIMARY KEY)
   * Marka
   * Modell
   * Evjarat
   * Ar
   * Kilometerora
   * Motortipus
   * Szin
   * KépURL (opcionális)
   * Egyéb jellemzők (pl. extrák)
3. **bookings**
   * booking\_id (PRIMARY KEY)
   * user\_id (FOREIGN KEY -> users.id)
   * auto\_id (FOREIGN KEY -> cars.Auto\_ID)
   * deposit (a 10% előleg összege)
   * payment\_method (pl. “átutalás”, “készpénz”)
   * status (pl. “foglalt”, “vásárlás alatt”)
   * created\_at
4. **favorites** (kedvencek)
   * fav\_id (PRIMARY KEY)
   * user\_id (FOREIGN KEY -> users.id)
   * auto\_id (FOREIGN KEY -> cars.Auto\_ID)

Természetesen az adatbázis-struktúra projektenként eltérhet, de a fenti váz segít a logika megértésében.

**8. Kapcsolatfelvétel és Eladási Lehetőség**

A weboldalon feltüntetett **e-mail** cím vagy **telefonszám** segítségével bárki felveheti a kapcsolatot a B&K Autókereskedéssel. Ha valaki eladni szeretné a saját használt autóját, a megadott elérhetőségeken keresztül tudja felvenni a kapcsolatot, és egyeztetni az autó átvizsgálásának és megvásárlásának részleteit.

A rendszer jelenleg nem tartalmaz külön online űrlapot az autófelajánlásra, de ez a funkció a jövőben beépíthető, hasonlóan a foglalási rendszerhez.

**9. Fejlesztési és Üzembe Helyezési Útmutató**

**9.1. Fejlesztői Környezet Előkészítése**

1. **Adatbázis létrehozása**: Hozz létre egy MySQL adatbázist, és futtasd a megfelelő SQL scriptet (ha rendelkezésre áll) a táblák generálásához.
2. **Backend telepítése**:
   * Klónozd vagy töltsd le a projektet.
   * Nyisd meg a parancssort a projekt gyökerében, majd futtasd:

bash

MásolásSzerkesztés

cd backend

npm install

* + Töltsd ki a .env fájlt a szükséges adatokkal (DB felhasználó, jelszó, host, JWT\_SECRET, stb.).
  + Indítsd el a szervert:

bash

MásolásSzerkesztés

npm run dev

vagy

bash

MásolásSzerkesztés

nodemon app.js

1. **Frontend telepítése**:
   * Lépj át a frontend könyvtárba:

bash

MásolásSzerkesztés

cd frontend

npm install

* + Indítsd el a fejlesztői szervert:

bash

MásolásSzerkesztés

npm start

* + A böngészőben a http://localhost:3000 címen érheted el az alkalmazást.

**9.2. Éles Környezetbe Történő Telepítés**

* **Build**: A React alkalmazást “production” buildre fordítsuk, amely optimalizált, kicsomagolt verziót hoz létre:

bash

MásolásSzerkesztés

npm run build

* **Szerver**: Az elkészült build mappát kiszolgálhatja egy statikus szerver (pl. serve csomag), vagy be lehet integrálni az Express szerverbe, ha a Node.js is kezeli a statikus fájlokat:

js

MásolásSzerkesztés

app.use(express.static(path.join(\_\_dirname, 'build')));

app.get('\*', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'build', 'index.html'));

});

* **Adatbázis és környezeti változók**: Ügyeljünk arra, hogy az éles környezetben a .env fájl ne legyen publikusan elérhető. Biztonságos környezetben tároljuk a DB hitelesítési adatokat és a JWT kulcsot.

**10. Jövőbeli Fejlesztési Lehetőségek**

* **Autófelajánlási űrlap**: Egy online felület, ahol az eladni kívánt autók adatait be lehet tölteni, fotókat csatolni, és közvetlenül beküldeni a rendszerbe.
* **Értékelési rendszer**: Az eladók és vásárlók közötti tranzakciók minősítése, visszajelzések megjelenítése.
* **Fejlettebb keresőmotor**: Keresés kulcsszavak alapján, például “automata váltó” vagy “metálfény”.
* **SMS értesítések**: A nodemailer mellett SMS gateway integráció is lehetséges, így a foglalásról vagy megerősítésről SMS-ben is értesülhet a felhasználó.
* **Teljesítményoptimalizálás**: Caching réteg bevezetése, CDN használata a képekhez, nagy mennyiségű adatnál pagináció és indexek finomhangolása az adatbázisban.

**11. Összefoglalás**

A **B&K Autókereskedés** weboldal egy átfogó, felhasználóbarát és reszponzív megoldás a használt autók adásvételére. A React alapú frontend könnyen bővíthető és karbantartható, míg az Express.js alapú backend stabil és biztonságos kiszolgálást nyújt. Az autók előzetes minőségi ellenőrzése, a részletes szűrő, valamint a kedvencek és foglalási rendszer lehetővé teszi, hogy a vásárlók a legjobb döntést hozzák meg.

A dokumentációban bemutatott architektúra, komponensek és funkciók együttesen biztosítják a gördülékeny felhasználói élményt, a jövőbeli fejlesztési ötletek pedig lehetőséget kínálnak arra, hogy a rendszer tovább fejlődjön, és még több szolgáltatást nyújtson a felhasználók számára.

A **B&K Autókereskedés** célja, hogy egyetlen online felületen egyesítse mindazokat a funkciókat, amelyekre a modern autóvásárlóknak és -eladóknak szükségük lehet. A jelen dokumentáció segít a fejlesztőknek és üzemeltetőknek megérteni a rendszer felépítését és működését, valamint megkönnyíti a további bővítéseket és karbantartási feladatokat.

Az oldal további fejlesztéseknek néz elébe.