

**DEBRECENI SZC BEREGSZÁSZI PÁL TECHNIKUM**

**ÉS KOLLÉGIUM**

Munkalapkezelő alkalmazás

Készítette: Kiss Csaba Lajos,

Kopasz Béla

**Debrecen, 2024**

**Tartalomjegyzék**

[Bevezetés 3](#_Toc165041107)

[Felhasználói dokumentáció 4](#_Toc165041108)

[Feladatspecifikáció 4](#_Toc165041109)

[Rendszerkövetelmények 4](#_Toc165041110)

[Program telepítése, elérése 4](#_Toc165041111)

[Program használatának leírása 5](#_Toc165041112)

[Webapplikáció leírása 5](#_Toc165041113)

[A kezdőlap és a menüpontok 6](#_Toc165041114)

[Megrendelők menüpont 6](#_Toc165041115)

[Új munkalap menüpont 7](#_Toc165041116)

[Aktív munkalapok, lezárt munkalapok és összes munkalap menüpontok 8](#_Toc165041117)

[Munkalapok részleteinek megtekintése 9](#_Toc165041118)

[Fejlesztői dokumentáció 10](#_Toc165041119)

[Témaválasztás indoklása 10](#_Toc165041120)

[Alkalmazott fejlesztői eszközök 10](#_Toc165041121)

[Backend Rendszerterv 11](#_Toc165041122)

[Adatmodell leírása 11](#_Toc165041123)

[API-végpontok 14](#_Toc165041124)

[Backend api végpontokat Django Rest Framework segítségével hoztam létre. 14](#_Toc165041125)

[Nézet osztályok(Views) 16](#_Toc165041126)

[Következő lépés a nézet osztályok definiálása volt. 16](#_Toc165041127)

[URL-ek beállítása: 20](#_Toc165041128)

[További fejlesztési lehetőségek 21](#_Toc165041129)

[Összegzés 23](#_Toc165041130)

[Irodalomjegyzék 24](#_Toc165041131)

# Bevezetés

A vizsgamunka követelményének megvalósításához egy webalkalmazás elkészítését tűztük ki célul. A webalkalmazás egy adott autószervíz munkalapjainak kezelésére alkalmas applikáció. A modern technológia fejlődése szinte megköveteli a munkalapok digitális kezelését. Erre a problémára szeretnénk megoldást kínálni a webalkalmazásunkkal. Ez a dokumentáció áttekintést nyújt a webalkalmazás elkészítésének folyamatáról. A React frontend és a Django backend fúziójával létrejött alkalmazás REST API-kat használva kezeli a munkalapokat. A dokumentáció feltárja a webalkalmazás egyes összetevőit, és tárgyalja azok funkcionalitását, valamint a REST API fontosságát az alkalmazásban.

Amint megnyitjuk a webalkalmazást egy menüsor fogad mindenkit. Innen egyszerűen lehet a menük között navigálni. A webalkalmazás teljes mértékben reszponzív alkalmazás, illetve a weblap tetején egy menüsor van, ahonnan több menüpont közül tud választani a felhasználó. A cég logójára kattintva a kezdőlapot érhetjük el. A menüsor első elemének segítségével egy új megrendelő adatait rögzíthetjük az adatbázisban, majd egy új munkalapot vehetünk fel. Továbbá a menüsort használva választhatunk a megjelenítendő adatok közül: megtekinthetjük az aktív munkalapokat, a lezárt munkalapokat, vagy az összes munkalapot. A munkalapok csempék formájában jelennek meg a legfontosabb adatokkal és egy megtekintés gombbal is el vannak látva, melyre kattintva részletesebb információhoz juthat a felhasználó az adott munkalappal kapcsolatban. A munkalap adatai módosíthatók, vagy törölhetjük is az adott munkalapot.

Úgy gondoljuk, hogy a webalkalmazás további fejlesztése, illetve kismértékbeni átalakítása esetén, hasznos lehet nemcsak autószerelő műhelyek számára.

# Felhasználói dokumentáció

## Feladatspecifikáció

Weblap funkciói:

* új megrendelő hozzáadása
* megrendelő módosítása/törlése
* új munkalap felvétele
* munkalap módosítása/törlése
* aktív munkalapok megjelenítése
* lezárt munkalapok megjelenítése
* az összes munkalap megjelenítése
* kiválasztott munkalap részleteinek megjelenítése

## Rendszerkövetelmények

Hardver követelménye: a legtöbb eszközön minden gond nélkül működik. Internet elérés nem szükséges a webalkalmazás futtatásához, működéséhez. Mindenféle böngészővel tökéletesen működik.

A webalkalmazás teljes mértékben reszponzív, ezért - esetlegesen - mobilon és tableten is ugyan úgy használható, minden funkcióját megőrzi.

## Program telepítése, elérése

Webes felület Frontend elérése:

[https://github.com/KopaszB/munkalap\_alkalmazas\_frontend.git](vizsgaremek_dokumentácio.docx)

Webes felület Backend elérése:

https://github.com/KCsaba75/vr\_Munkalap.git

A webalkalmazás eléréséhez bejelentkezési adatra nincs szükség, mivel a munkalapok kezelését egy ember (adminisztrátor) végzi.

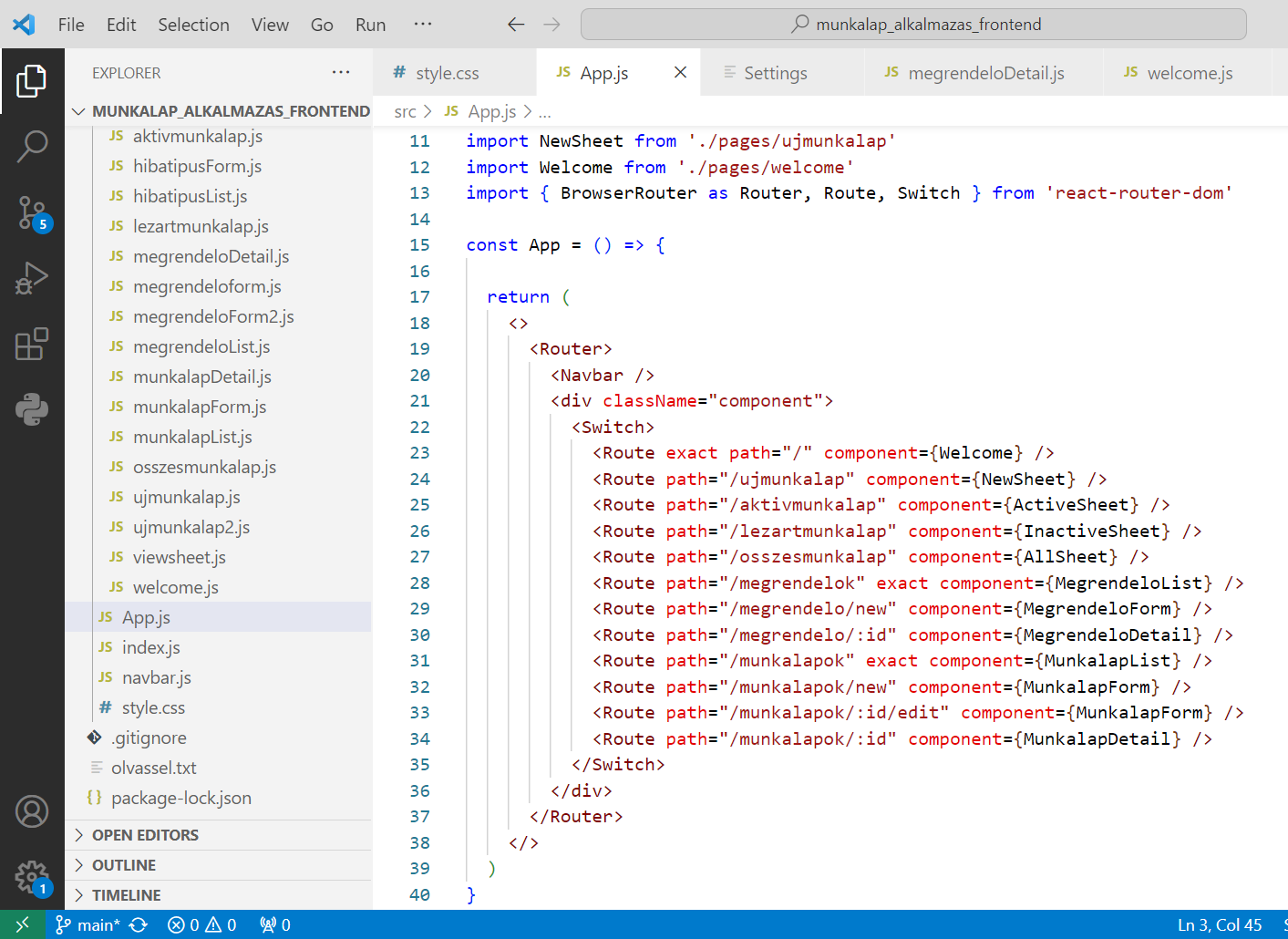
## Program használatának leírása

### Webapplikáció leírása

A webapplikáció felhasználói felületének programozása a React keretrendszerben készült. A React egy JavaScript könyvtár, amelyet általában felhasználói felületek (UI-k) készítésére használnak webalkalmazások fejlesztése során.

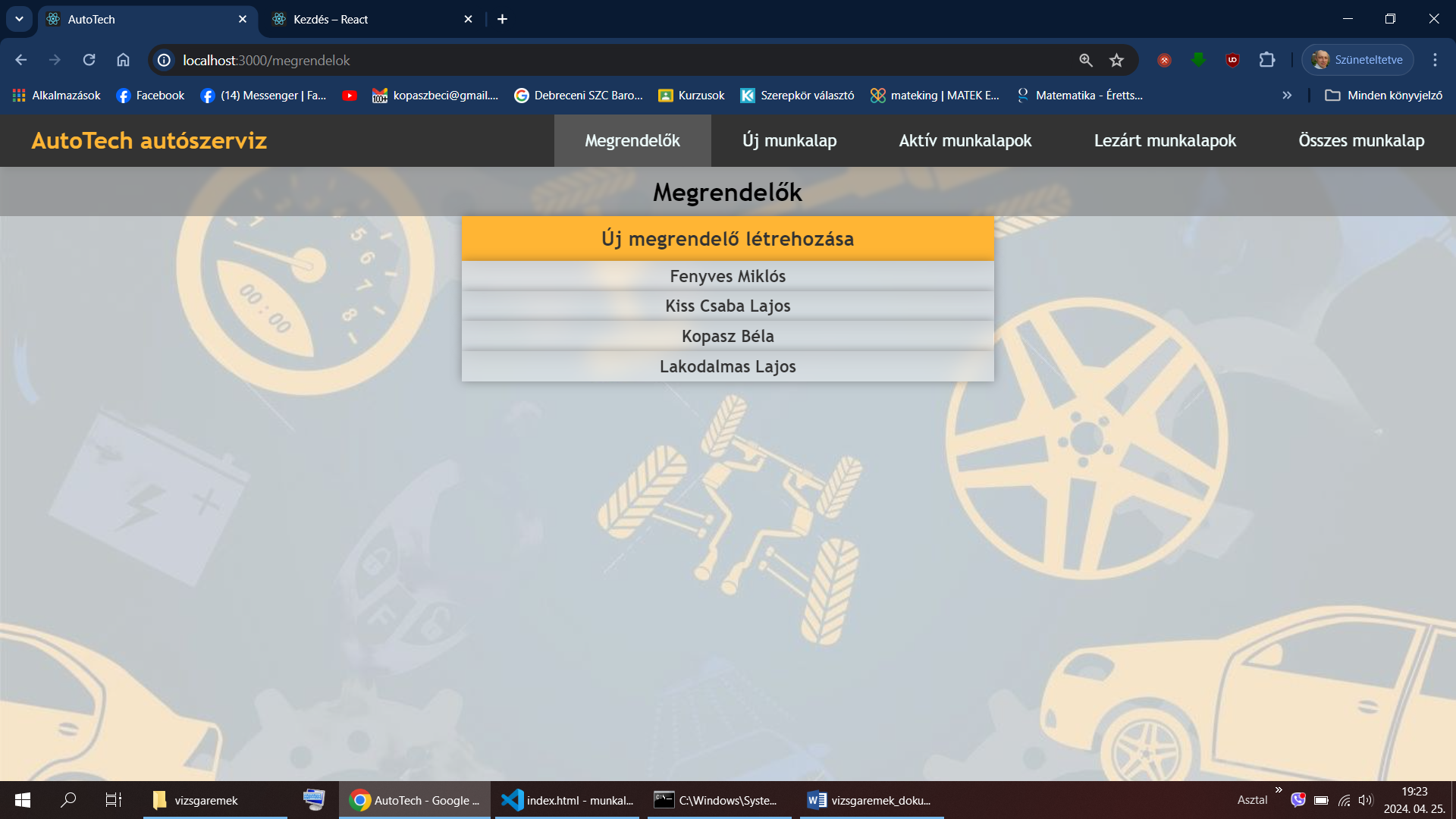
Az egyik legfontosabb tulajdonsága, hogy komponens alapú fejlesztési modellt használ, amely lehetővé teszi a kód újrafelhasználását és a nagy projektek könnyebb kezelését.

A másik fontos tulajdonsága a jsx, ami egy olyan nyelvi kiterjesztés, amely lehetővé teszi a JavaScript és a HTML kombinálását. Ezen kívül számos kiegészítőt és könyvtárat tartalmaz, amelyek további funkciókat és lehetőségeket kínálnak a fejlesztőknek, például a React Router a navigációhoz.



### A kezdőlap és a menüpontok

A kezdőlapon a menüsor öt gombját láthatjuk. Mivel ez egy egyoldalas alkalmazás (single-page-application, későbbiekben: SPA) a gombok nem gyors elérésként működnek. Ha bármelyikre rákattintunk, akkor nem átirányít a kiválasztott oldalra, hanem betölti a megfelelő komponenst. A React egyik nagy előnye, hogy bármilyen interakció az oldallal vagy alárendelt oldalakkal nem feltétlenül igényel a szerver felé kérést, ami azt jelenti, hogy az oldal nem töltődik újra.



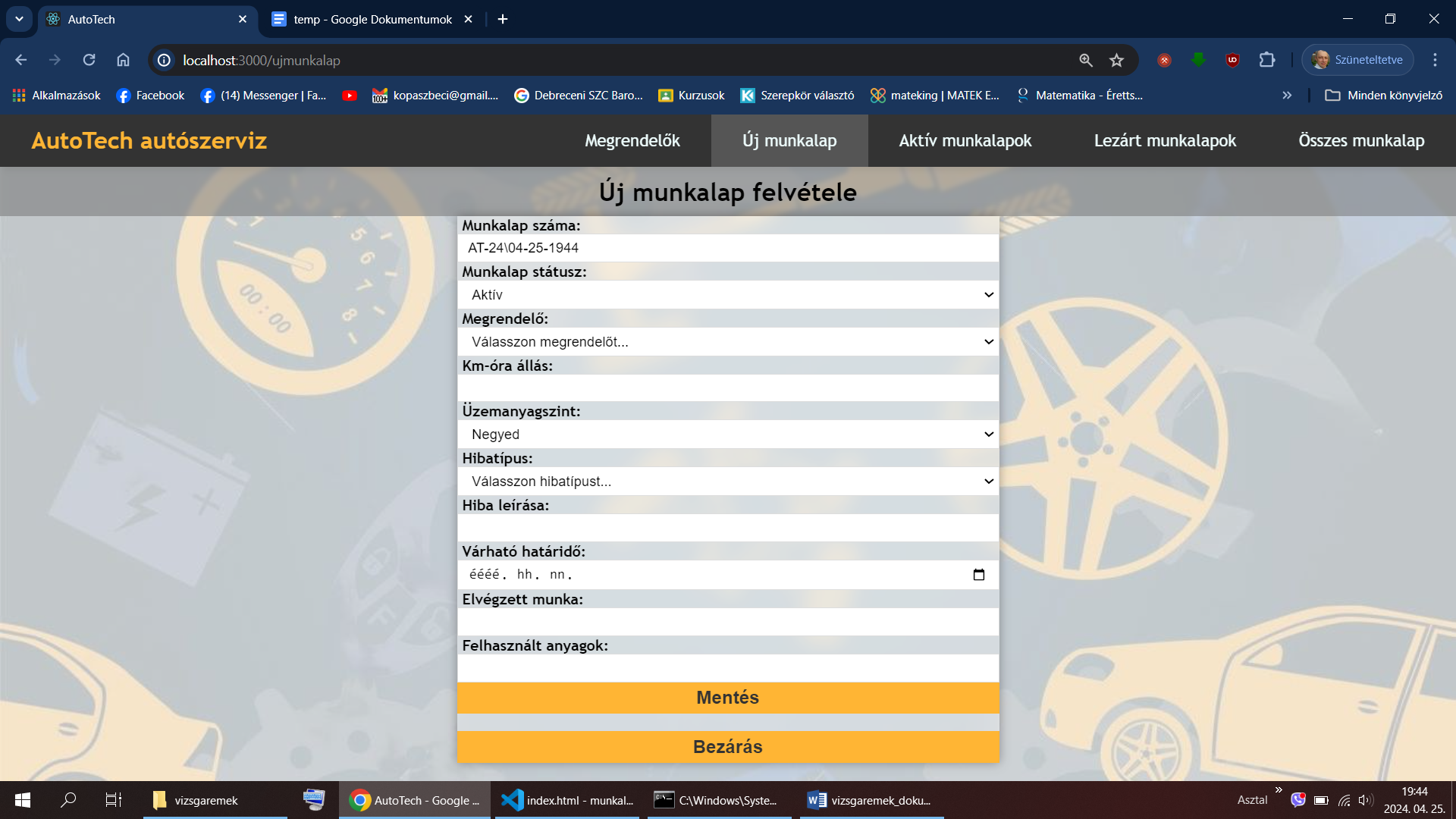
### Megrendelők menüpont

A megrendelők menüpont alatt új megrendelőket adhatunk hozzá az adatbázishoz, illetve a már meglévő megrendelőket láthatjuk. A megrendelő nevére kattintva részletes információkat kaphatunk a megrendelőről. Módosíthatjuk a megrendelő adatait illetve törölhetjük a megrendelőt az adatbázisból.



### Új munkalap menüpont

A menüpont alatt új munkalapot tudunk felvenni az adatbázisba. A munkalap száma automatikusan generált így azt nem kell kitölteni. Aktív- és lezárt munkalapstátuszok közül választhatunk, jelezvén a munkafolyamatot. Itt megrendelő felvételére nincs lehetőség, a már hozzáadott megrendelők közül lehet választani.



### Aktív munkalapok, lezárt munkalapok és összes munkalap menüpontok

A menüpontok alatt, nevüknek megfelelően, az aktív munkalapokat, a lezárt munkalapokat, valamint az összes munkalapot megtekinthetjük. A munkalapok kártyákon vannak elhelyezve, megjelenítve a legfontosabb adatokat a teljes munkalapról. Továbbá minden kártyán elhelyezésre került egy megtekintés gomb, amelyre rákattintva a munkalaphoz tartozó összes adatot meg tudjuk jeleníteni.



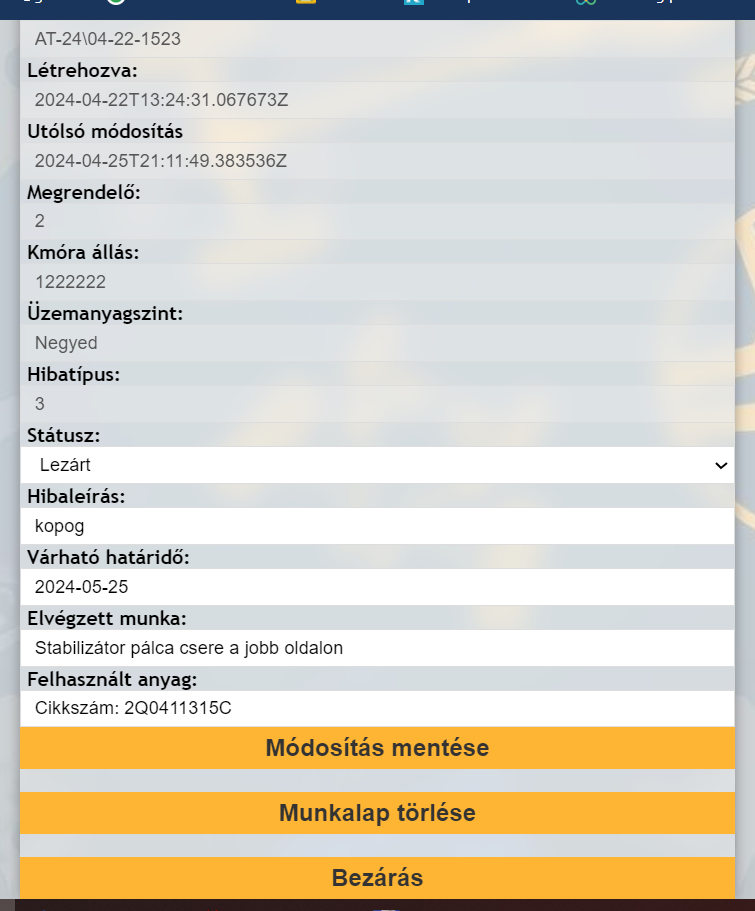


Könnyebb eligazodás végett az aktív és lezárt munkalapstátusz pirossal lett kiemelve. Módosításakor az ennek megfelelő helyre kerülnek a munkalapoki is.



### Munkalapok részleteinek megtekintése

A kiválasztott kártya Megtekintés gombjára kattintva részletes információt kaphatunk a munkalapról. A munkalap nem minden mezője módosítható, mivel a szerviz elvégzése után csak a következő mezőket kell/lehet változtatni: státusz, hiba leírása, várható határidő, elvégzett munka, felhasznált anyag. Itt van lehetőség a munkalap törlésére is, ami az adatbázis összes táblájából törli az adott munkalaphoz tartozó rekordokat.

# Fejlesztői dokumentáció

## Témaválasztás indoklása

A témaválasztás közös döntés eredménye, valós autószerelő műhely által támasztott digitalizációs igény alapján döntöttünk a pilot alkalmazás megvalósítása mellett. A konkrét kérésük az volt, hogy egy olyan platformot hozzunk létre, ami alkalmas a szerviz munkalapjainak digitális formában való kezelésére. Egyeztetések után úgy láttuk jónak, hogy a Django backend és a React frontend vonalon fogjuk megvalósítani egy webalkalmazás formájánam. Ez nekünk nagyszerű tesztfeladat egy a későbbiekben értékesítendő szoftver létrehozásához.

## Alkalmazott fejlesztői eszközök

Frontend:

Munkánk során mindketten a Visual Studio Code -ot használtuk, ami egy jól kezelhető és sokoldalú kódszerkesztő és fejlesztői környezet. Kifejezetten arra tervezték, hogy kényelmes és hatékony legyen a fejlesztők számára. Számos programozási nyelvet és technológiát támogat, beleértve a JavaScriptet, a Python-t, és a HTML-t is. A felhasználói felület kialakítására a React keretrendszert használtuk.

Backend:

A backend egy Django REST framework alapú alkalmazás, amely API-kat biztosít a **Megrendelo**, **Hibatipusok** és **Munkalap** modellek kezeléséhez. Lehetővé teszi a modellek lekérdezését, létrehozását, frissítését és törlését, valamint az aktív, inaktív és az összes munkalap JSON formátumban történő listázását.

## Backend Rendszerterv

### Adatmodell leírása

Adatok tárolása a Django részét képező SQLite adatbázisban történik.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

Adatbázis Létrehozva Django Osztálymodelben:

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

Táblák mezőinek leírása funkciójuk szerint

**Megrendelő Tábla**

* **nev (név)**: Ez a mező a megrendelő nevét tárolja szöveges karakterláncként, maximum 255 karakter hosszúsággal.
* **cim (cím)**: Ez a mező a megrendelő címét tárolja szöveges karakterláncként, maximum 255 karakter hosszúsággal.
* **email (email cím)**: Ez a mező a megrendelő email címét tárolja szöveges karakterláncként, maximum 255 karakter hosszúsággal.
* **telefon (telefonszám)**: Ez a mező a megrendelő telefonszámát tárolja egész számként.
* **rendszam (rendszám)**: Ez a mező a megrendelő járművének rendszámát tárolja szöveges karakterláncként, maximum 10 karakter hosszúsággal.
* **gyartmany (gyártmány)**: Ez a mező a megrendelő járművének gyártmányát (pl.: Ford, Toyota) tárolja szöveges karakterláncként, maximum 255 karakter hosszúsággal.
* **tipus (típus)**: Ez a mező a megrendelő járművének típusát (pl.: Mustang, Corolla) tárolja szöveges karakterláncként, maximum 255 karakter hosszúsággal.
* **gyartási\_ev (gyártási év)**: Ez a mező a megrendelő járművének gyártási évét tárolja egész számként.
* **alvazszam (alvázszám)**: Ez a mező a megrendelő járművének alvázszámát tárolja szöveges karakterláncként, maximum 50 karakter hosszúsággal.

**Hibatípusok Tábla**

* **hiba (hiba)**: Ez a mező a hiba típusát tárolja (pl.: motorhiba, elektromos probléma) szöveges karakterláncként, maximum 255 karakter hosszúsággal.

**Munkalap Tábla**

* **megrendelo\_id (megrendelő azonosító)**: Ez a mező egy idegen kulcs, amely a ***Megrendelo*** tábla elsődleges kulcsára hivatkozik.
* **datum (dátum)**: Ez a mező a munkalap létrehozásának dátumát és időpontját tárolja, és a rekord létrehozásakor automatikusan kitöltődik.
* **utolsomodostas (utolsó módosítás)**: Ez a mező a munkalap utolsó módosításának dátumát és időpontját tárolja, és a rekord mentésekor automatikusan frissül.
* **munkalapstatus (munkalap státusz)**: Ez a mező a munkalap állapotát (pl.: aktív, lezárt) tárolja előre definiált választási lehetőségek használatával.
* **munkalapszam (munkalapszám)**: Ez a mező a munkalap egyedi azonosítóját tárolja szöveges karakterláncként, maximum 20 karakter hosszúsággal.
* **kmoraallas (kilométeróra állás)**: Ez a mező a jármű kilométeróra állását tárolja a munkalap létrehozásakor egész számként.
* **uzemanyagszint (üzemanyagszint)**: Ez a mező a jármű üzemanyagszintjét tárolja a munkalap létrehozásakor előre definiált választási lehetőségek használatával (pl.: negyed, fél, háromnegyed, tele).
* **hibatipus\_id (hibatípus azonosító)**: Ez a mező egy idegen kulcs, amely a ***Hibatipusok*** tábla elsődleges kulcsára hivatkozik.
* **hibaleírás (hiba leírás)**: Ez a mező a szöveges mező a megrendelő által jelentett hiba leírását tárolja.
* **varhato\_hatarido (várható határidő)**: Ez a mező a munka várható befejezési dátumát tárolja.
* **elvegzettmunka (elvégzett munka)**: Ez a szöveges mező a járművön elvégzett munka leírásának tárolására szolgál. Alapértelmezetten üres mező.
* **felhasznaltanyag (felhasznált anyag)**: Ez a szöveges mező a javítás során felhasznált anyagok felsorolására szolgál. Alapértelmezetten üres mező.

### API-végpontok

### Backend api végpontokat Django Rest Framework segítségével hoztam létre.

Serialiser osztályok:

A képen szöveg, képernyőkép, szoftver látható

Automatikusan generált leírás

1. **MegrendeloSerializer**

Ez a serializer osztály a **Megrendelo** modell példányainak szerializálására és deszerializálására szolgál. A Django REST keretrendszer **ModelSerializer** osztályából örököl.

A **MegrendeloSerializer** osztályon belüli **Meta** osztály a következő tulajdonságokat definiálja:

* **model**: Ez a tulajdonság meghatározza azt a modellt, amelyhez a serializer társítva van, ebben az esetben a **Megrendelo** modellt.
* **fields**: Ez a tulajdonság meghatározza a modell azon mezőit, amelyeket bele kell foglalni a szerializált reprezentációba. Ebben az esetben a **\_\_all\_\_** meta opciót használjuk, ami azt jelenti, hogy a modell összes mezője szerepelni fog.

1. **HibatipusokSerializer**

Ez a serializer osztály a **Hibatipusok** modell példányainak szerializálására és deszerializálására szolgál a **Hibatipusok** modellhez van társítva.

A **HibatipusokSerializer** osztályon belüli **Meta** osztály a következő tulajdonságokat definiálja:

* **model**: Ez a tulajdonság meghatározza azt a modellt, amelyhez a serializer társítva van, ebben az esetben a **Hibatipusok** modellt.
* **fields**: Ez a tulajdonság meghatározza a modell azon mezőit, amelyeket bele kell foglalni a szerializált reprezentációba. Ebben az esetben a **\_\_all\_\_** meta opciót használjuk, ami azt jelenti, hogy a modell összes mezője szerepelni fog.

1. **MunkalapSerializer**

Ez a serializer osztály a **Munkalap** modell példányainak szerializálására és deszerializálására szolgál. Hasonlít az előző osztályokhoz, de tartalmaz néhány további funkciót is.

A **MunkalapSerializer** osztályon belüli **Meta** osztály a következő tulajdonságokat definiálja:

* **model**: Ez a tulajdonság meghatározza azt a modellt, amelyhez a serializer társítva van, ebben az esetben a **Munkalap** modellt.
* **fields**: Ez a tulajdonság meghatározza a modell azon mezőit, amelyeket bele kell foglalni a szerializált reprezentációba. Ebben az esetben a **\_\_all\_\_** meta opciót használjuk, ami azt jelenti, hogy a modell összes mezője szerepelni fog.

1. **MunkalapListSerializer**

Ez a serializer osztály a **Munkalap** modell példányainak szerializálására és deszerializálására szolgál. Hasonlít az előző osztályokhoz, de tartalmaz néhány további funkciót is.

A **MunkalapSerializer** osztályon belüli **Meta** osztály a következő tulajdonságokat definiálja:

* **model**: Ez a tulajdonság meghatározza azt a modellt, amelyhez a serializer társítva van, ebben az esetben a **Munkalap** modellt.
* **fields**: Ez a tulajdonság meghatározza a modell azon mezőit, amelyeket bele kell foglalni a szerializált reprezentációba. Ebben az esetben a **\_\_all\_\_** meta opciót használjuk, ami azt jelenti, hogy a modell összes mezője szerepelni fog.

Ezen tulajdonságok mellett a **MunkalapListSerializer** osztály két egyéni mezőt is definiál:

* **munkalapstatus**: Ez a mező egy CharField, amely a munkalap állapotát reprezentálja. A **get\_munkalapstatus\_display()** metódus segítségével kerül kitöltésre, amely visszaadja az **enum** állapot ember által olvasható megjelenítését.
* **uzemenyagszint**: Ez a mező egy CharField, amely a jármű üzemanyagszintjét reprezentálja. A **get\_uzemenyagszint\_display()** metódus segítségével kerül kitöltésre, amely visszaadja az üzemanyagszint **enum** állapot ember által olvasható megjelenítését.

Valamint:

* **megrendelo\_id**: Ez a mező egy beágyazott serializer, amely a munkalaphoz társított megrendelőt reprezentálja. A **MegrendeloSerializer** osztállyal van kitöltve.
* **hibatipus\_id**: Ez a mező egy beágyazott serializer, amely a munkalaphoz társított hibatípust reprezentálja. A **HibatipusokSerializer** osztállyal van kitöltve.

Ennek a beállításnak eredményeként a **MunkalapListSerializer** osztály használható a **Munkalap** modell példányainak listájának JSON formátumba való konvertálására, beleértve a társított megrendelő és hiba típus információit is.

### Nézet osztályok(Views)

### Következő lépés a nézet osztályok definiálása volt.

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

**1. Megrendelő nézet osztályok:**

* **MegrendeloListCreate(generics.ListCreateAPIView):**
  + Ez az osztály lehetővé teszi a Megrendelo modellek listázását és létrehozását az API segítségével.
  + A queryset attribútum meghatározza a lekérdezendő összes Megrendelo objektumot.
  + A serializer\_class attribútum meghatározza a szerializáló osztályt, amely a Megrendelo objektumokat JSON formátumba alakítja át.
* **MegrendeloRetrieveUpdateDestroy(generics.RetrieveUpdateDestroyAPIView):**
  + Ez az osztály lehetővé teszi a meglévő Megrendelo modellek lekérdezését, frissítését és törlését az API segítségével.
  + A queryset és serializer\_class attribútumok ugyanúgy működnek, mint a MegrendeloListCreate osztályban.

**2. Hibatipusok nézet osztályok:**

* **HibatipusokListCreate(generics.ListCreateAPIView):**
  + Ez az osztály lehetővé teszi a Hibatipusok modellek listázását és létrehozását az API segítségével.
  + A queryset és serializer\_class attribútumok hasonlóan működnek, mint a MegrendeloListCreate osztályban, de a Hibatipusok modellre vonatkoznak.
* **HibatipusokRetrieveUpdateDestroy(generics.RetrieveUpdateDestroyAPIView):**
  + Ez az osztály lehetővé teszi a meglévő Hibatipusok modellek lekérdezését, frissítését és törlését az API segítségével.
  + A queryset és serializer\_class attribútumok ugyanúgy működnek, mint a HibatipusokListCreate osztályban.

**3. Munkalap nézet osztályok:**

* **MunkalapListCreate(generics.ListCreateAPIView):**
  + Ez az osztály lehetővé teszi a Munkalap modellek listázását és létrehozását az API segítségével.
  + A queryset és serializer\_class attribútumok hasonlóan működnek, mint a MegrendeloListCreate osztályban, de a Munkalap modellre vonatkoznak.
* **MunkalapRetrieveUpdateDestroy(generics.RetrieveUpdateDestroyAPIView):**
  + Ez az osztály lehetővé teszi a meglévő Munkalap modellek lekérdezését, frissítését és törlését az API segítségével.
  + A queryset és serializer\_class attribútumok ugyanúgy működnek, mint a MunkalapListCreate osztályban.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

**4. Egyéni Munkalap nézet osztályok**

* **AktivMunkalapokJsonView(APIView):**
  + Ez az osztály egy egyéni nézet, amely JSON formátumban ad vissza egy listát az összes aktív munkalapról (ahol a munkalapstatus értéke "enum\_value\_1").
  + A get metódus lekéri az összes aktív munkalapot, majd a MunkalapListSerializer osztályt használja azok JSON-ba való konvertálásához.
* **InAktivMunkalapokJsonView(APIView):**
  + Ez az osztály egy egyéni nézet, amely JSON formátumban ad vissza egy listát az összes inaktív munkalapról (ahol a munkalapstatus értéke "enum\_value\_2").
  + A get metódus hasonlóan működik az AktivMunkalapokJsonView osztályhoz, de inaktív munkalapokat keres.
* **OsszesMunkalapokJsonView(APIView):**
  + Ez az osztály egy egyéni nézet, amely JSON formátumban ad vissza egy listát az összes munkalapról.
  + A get metódus hasonlóan működik az AktivMunkalapokJsonView osztályhoz, de minden munkalapot lekérdez.

### URL-ek beállítása:

A képen szöveg, képernyőkép, szoftver látható

Automatikusan generált leírás

A urlpatterns lista határozza meg az URL-mintákat a Django projekthez. Ezek a minták megmondják a Django alkalmazásnak, hogy melyik nézetet kell meghívnia egy adott URL kérés esetén.

* **path('admin/', admin.site.urls)**: Ez a minta az adminisztrációs felületre irányít minden olyan kérést, amely az /admin/ elérési úttal kezdődik. A django.contrib.admin modul által biztosított admin.site.urls attribútum tartalmazza az adminisztrációs felület URL-mintáit.
* **path('api/megrendelok/', views.MegrendeloListCreate.as\_view())**: Ez a minta az API-hoz tartozó /api/megrendelok/ végpontra irányít. A views.MegrendeloListCreate.as\_view() metódus meghívja a MegrendeloListCreate osztály nézetét, amely lehetővé teszi a Megrendelo modellek listázását és létrehozását.
* **path('api/megrendelok/<int:pk>/', views.MegrendeloRetrieveUpdateDestroy.as\_view())**: Ez a minta az API-hoz tartozó /api/megrendelok/<int:pk>/ végpontra irányít. A <int:pk> rész egy egész számot (primer key) fogad el, amely a lekérdezni kívánt Megrendelo modell azonosítója. A views.MegrendeloRetrieveUpdateDestroy.as\_view() metódus meghívja a MegrendeloRetrieveUpdateDestroy osztály nézetét, amely lehetővé teszi a meglévő Megrendelo modellek lekérdezését, frissítését és törlését.

A következő minták (api/hibatipusok/, api/hibatipusok/<int:pk>/', api/munkalapok/, api/munkalapok/<int:pk>/) hasonlóan működnek az előző Megrendelo példákhoz, de a Hibatipusok és Munkalap modellekre vonatkoznak.

* **path('api/munkalapAktiv/', views.AktivMunkalapokJsonView.as\_view())**: Ez a minta az API-hoz tartozó /api/munkalapAktiv/ végpontra irányít. A views.AktivMunkalapokJsonView.as\_view() metódus meghívja az AktivMunkalapokJsonView osztályt, amely JSON formátumban ad vissza egy listát az összes aktív munkalapról.

A következő két minta (api/munkalapInAktiv/, api/munkalapOsszes/) hasonlóan működnek az előző példához, de az inaktív (InAktivMunkalapokJsonView) és összes (OsszesMunkalapokJsonView) munkalapot listázzák ki.

Összességében az urlpatterns lista egy olyan útválasztási mechanizmust biztosít, amely a bejövő URL-kéréseket a megfelelő nézetekhez irányítja a kívánt műveletek végrehajtása érdekében.

## További fejlesztési lehetőségek

Az eddig leírt funkcionalitáson kívül számos továbbfejlesztési lehetőség is nyitott a felhasználói felülethez és backendhez. Íme, néhány ötlet:

**Funkcionalitás bővítése:**

* Felhasználói felületen adatkezelési opciók bővítése úgy, mint keresés és szűrés a megrendelőknél és a munkalapoknál.
* További modellek hozzáadása az API-hoz, mint például a felhasználók, jogosultságok, beállítások stb.
* Komplexebb szűrési és rendezési lehetőségek bevezetése az API lekérdezésekhez.
* Műveletek végrehajtásának lehetősége az API-n keresztül, mint például e-mailek küldése, fájlok feltöltése, push értesítések küldése stb.
* Automatizálási funkciók bevezetése a feladatok automatizálására.

**Teljesítmény optimalizálása:**

* Cache-elési mechanizmusok bevezetése a lekérdezések gyorsítására.
* Skálázhatóság javítása a terhelés növekedésének kezelésére.
* Teljesítményfigyelési és elemzési eszközök integrálása.

**Biztonság javítása:**

* Továbbfejlesztett hitelesítési és engedélyezési mechanizmusok bevezetése.
* Adatvédelmi funkciók javítása.
* Behatolási tesztek és sebezhetőségvizsgálatok végrehajtása a biztonsági kockázatok azonosítására és javítására.

**Felhasználói élmény javítása:**

* Dokumentáció és oktatóanyagok javítása a fejlesztők és a felhasználók számára.
* Hibakeresési és naplózási funkciók javítása a problémák gyorsabb azonosításához és megoldásához.
* Végfelhasználói felület (UI) fejlesztése az API-hoz, például egy adminisztrációs felület vagy egy fejlesztői konzol.

# Összegzés

A kitűzött cél egy olyan platform létrehozása volt létező autószerviz számára, amelyen keresztül munkalapokat lehet digitalizálni. A projektünk végtermékében ez a cél megvalósult. Sikerült elkészíteni egy olyan webalkalmazást, amely alkalmas egy szerviz munkalapjainak kezelésére. Értjük ez alatt a munkalapok létrehozását, módosítását, törlését, a megrendelők adatainak adatbázisban való tárolását, módosítását, törlését, a munkalapok aktív és lezárt kategóriák szerinti megjelenítését. A projekt elkészítése a személyes fejlődés, önképzés szempontjából is hasznos volt. Mind a frontend, mind a backend fejlesztésénél igyekeztünk jó gyakorlatokat alkalmazni, elsajátítani. Köszönetet mondunk tanáraink felé, akik áldozatos munkájukkal és elkötelezettségükkel új ismereteket, használható tudást adtak, megtanítva bennünket az önálló- és projektben való munkára, együttműködésre. A webalkalmazásunkat továbbfejlesztés szempontjából mi még nem tekintjük késznek. Erről fentebb bővebben írtunk.

A backend API-ja, amelyet a Django REST framework-kel hoztunk létre, számos platformfüggetlen frontend fejlesztési lehetőséget nyit meg a webes és mobilalkalmazások számára.

# Irodalomjegyzék

Frontend:

<https://www.w3schools.com/html/default.asp>

<https://www.w3schools.com/js/default.asp>

https://hu.legacy.reactjs.org/docs/glossary.html

Backend:

<https://www.djangoproject.com/>

<https://www.django-rest-framework.org/>