

DSZC Mechwart András Gépipari és Informatikai Technikum



Raktárkezelő alkalmazás

Projektmunka dokumentáció

Készítette: Kiss Róbert és Szatmári Gábor

Debrecen

2025

Tartalomjegyzék

Bevezetés.....	3
Szakirodalmi áttekintés	3
Használt technológiák	6
A projekt bemutatása.....	8
Tervek a további fejlesztésekre	13
Összegzés	14
Irodalom	16

Bevezetés

Sokan szembesülnek napjainkban azzal a problémával, hogy a digitalizáció ellenére sok területen még tartja magát a papíralapú dokumentáció. Ennek van is helye bizonyos szférákban, ilyen például a pénzügyi- és adóügyi terület, ahol a bizonylatok kiállítása jelentőséggel bír. Azonban sok munkakörben jelentősen csökkenti a munka hatékonyságát, sebességét, emellett növeli a hibalehetőségeket is.

A logisztikában, raktározásban a papíralapú nyilvántartások, leltárak, bevételezési- és kiadási bizonylatok kezelése sem egyszerű feladat. Ezzel a problémával mindketten szembesültünk korábbi munkavégzésünk során. Erre megoldásként néhány évvel ezelőtt Kiss Róbert önképzéssel és kísérletezéssel Java programozási nyelven fejlesztett egy egyszerű asztali alkalmazást, a hétköznapi munkát megkönnyítendő. A raktárban bent lévő anyagok és alkatrészek nyilvántartása, azok kiadása és bevételezése, a beszállítók kezelése, a leltározás mind-mind nagyon nehézkes volt papíron vezetve. Ezt a napi működés során végbemenő anyagmozgások olykor már kaotikussá tették a helyzetet. A továbbiakban részletesen kifejtjük a program működését, de ezt a Java-alkalmazást vettük alapul, a Mechwartban két év alatt tanultakkal kiegészítve, fejlesztve egy webes felületre készítettük el a projektünket.

Szakirodalmi áttekintés

A raktározás, a logisztika elavultságáról már több tanulmány is született. Magyarországon problémaként említik többek között a raktározási kapacitások alacsony volumenét, valamint a szakképzett raktári dolgozók alacsony számát, ami ágazati szinten jelen van sajnos.¹

A lehetőségek hatékonyabb kiaknázására ideális irány lehet a meglévő kapacitások fejlesztése, a humánerőforrás fejlesztése, és a munka gyorsabbá, hatékonyabbá tétele a digitalizálódásba való bekapcsolódással. Az automatizált raktárkezelés nem csak a robotikában érhető tetten. A nyilvántartások és az adminisztráció digitalizációja, hálózatosítása jelentősen növelheti a létszámú munkaerő hatékonyságát, és csökkenti az emberi hibalehetőségeket. Erre a gondolatra

¹ Plötz Anita-Varga János: Digitalizáció hatása a logisztikára

felfűzve készítettük el az alkalmazásunkat, hiszen ha lokálisan, mikroszinten eszközlünk fejlesztéseket, akkor makroszinten is eredményesebbé válik a vállalat.

A raktárkezelő felületünk backend háttére Django környezetben készült. A Django kiváló választás raktárkezelő rendszer backend fejlesztéséhez, mivel beépített ORM-je (Object-Relational Mapping) egyszerűsíti az adatbázis-műveleteket, ami elengedhetetlen a készletnyilvántartás, bevételezés és kiadás pontos kezeléséhez. A framework "batteries included" filozófiája miatt számos beépített funkciót kínál, mint például a felhasználókezelés, jogosultságkezelés és munkamenet-kezelés, amelyek kritikusak egy raktári rendszerben, ahol különböző szintű hozzáférésre van szükség.

A Django admin felülete azonnal használható adminisztrációs panelt biztosít, amely gyorsan testreszabható a raktári adatok kezelésére. Az MVC architektúra (Model-View-Template) tiszta kódstruktúrát eredményez, megkönnyítve a karbantartást és a bővítést. A beépített biztonsági mechanizmusok védenek az SQL injection, XSS és CSRF támadások ellen, ami különösen fontos érzékeny készletadatok kezelésénél.

A Django REST Framework segítségével könnyen építhető API, amely lehetővé teszi mobilapplikációk vagy vonalkódolvasók integrálását. A framework skálázhatósága biztosítja, hogy a rendszer több raktár vagy nagyobb adatmennyiség kezelésére is alkalmas legyen. A migrations rendszer automatizálja az adatbázis-séma változásait, minimalizálva az emberi hibákat a rendszer fejlesztése során.

A Python egyszerűsége és olvashatósága megkönnyíti a csapatmunka koordinálását, míg a gazdag ökoszisztéma számos külső könyvtárat kínál speciális funkciókhoz, például riportgeneráláshoz vagy adatelemzéshez. A Django dokumentációja kiváló, és a nagy közösség gyors segítséget nyújt felmerülő problémák esetén, ami felgyorsítja a fejlesztési folyamatot.²

A megjelenést illetően egyszerűen működő lehetőségeket kerestünk, tekintve, hogy jellemzően ipari felhasználásra szánjuk a programunkat, így a JavaScript, Bootstrap és CSS felhasználásával készültek az oldalak. Ennek számos előnye ismert.

² Django hivatalos oldala, W3schools hivatalos oldala

A JavaScript a modern webalkalmazások gerincét alkotja, lehetővé téve dinamikus, interaktív felhasználói felületek létrehozását. Egy raktárkezelő rendszerben elengedhetetlen a valós idejű adatfrissítéshez, űrlapok validálásához és AJAX-alapú kommunikációhoz a backend-del. A JavaScript segítségével azonnal reagálhatunk a felhasználói műveletekre anélkül, hogy az oldalt újra kellene tölteni, ami jelentősen javítja a felhasználói élményt és a munka hatékonyságát.

A Bootstrap a világ legnépszerűbb CSS keretrendszere, amely előre elkészített komponenseket és reszponzív grid rendszert biztosít. Legnagyobb előnye, hogy gyorsan és konzisztensen fejleszthető vele professzionális megjelenésű felület minimális CSS tudással is. A Bootstrap automatikusan optimalizálja a megjelenést különböző képernyőméretekre, így a raktárkezelő felület tableteken és mobileszközökön is tökéletesen használható, ami a raktári környezetben különösen fontos.

A CSS (Cascading Style Sheets) felelős a vizuális megjelenésért és a felhasználói élményért. Modern CSS3 funkciókkal, mint a flexbox és grid layout, komplex elrendezések valósíthatók meg egyszerűen. A CSS animációk és átmenetek élénkebbé teszik a felületet, vizuális visszajelzést adva a felhasználónak műveletei sikerességéről.

A három technológia együttes használata optimális eredményt hoz: a JavaScript biztosítja a funkcionalitást, a Bootstrap a gyors fejlesztést és reszponzivitást, míg a CSS a testreszabott, márkaidentitásnak megfelelő megjelenést. Ez a kombináció cross-browser kompatibilitást garantál, széles körű közösségi támogatást élvez, és rengeteg ingyenes erőforrás, sablon és plugin áll rendelkezésre. A moduláris felépítés miatt könnyen karbantartható és bővíthető a kód, ami hosszú távon csökkenti a fejlesztési költségeket és időt.³

³ Bootstrap hivatalos oldala, JavaScript hivatalos oldala, W3schools hivatalos oldala, Webiskola

Használt technológiák

A raktárkezelő felületünk backend hátterét egy Django keretrendszerben készült szerver biztosítja. A működtetéséhez szükséges bővítményeket a mellékelt dokumentum (requirements.txt) és a 2. ábra tartalmazza, alapvetően a Django, Cors-headers, a Pillow, és a Restframework telepítésére van szükség. Az automatikusan létrejött fájlstruktúra fejlesztésével az ábrán látható szerkezettel dolgozunk.

Az oldalak betöltése a Leltár kivételével szerveroldali rendereléssel történik, így minden oldal template-ekként készült.

A frontend oldalon a megjelenést Bootstrap felhasználásával CSS és JS fájlokra hivatkozva a template-ek biztosítják. A leltár oldalon használtunk egyedül REST API elérést.

Alapvetően asztali gépre szánt alkalmazás, tekintve az adatbázis jellegére és a nagyobb méretű táblázatos

```
backend
├── config
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   ├── wsgi.py
│   └── __init__.py
├── raktar
│   ├── migrations
│   │   ├── 0001_initial.py
│   │   ├── 0002_rename_gebizid_bizonylat_genbizid.py
│   │   ├── 0003_alter_bizonylat_rendszam_alter_bizonylat_szallito.py
│   │   ├── 0004_alapadat.py
│   │   ├── 0005_alapadat_szorzo.py
│   │   └── 0006_alter_bizonylat_rendszam_alter_bizonylat_szallito_and_more.py
│   │   └── __init__.py
│   ├── static
│   │   ├── assets
│   │   │   ├── images
│   │   │   │   ├── logo.png
│   │   │   │   └── raktar.jpg
│   │   ├── css
│   │   │   ├── alkatresz-csoportok.css
│   │   │   ├── alkatreszek.css
│   │   │   ├── bebizonylat.css
│   │   │   ├── bebizonylatsorok.css
│   │   │   ├── beszallitok.css
│   │   │   ├── dashboard.css
│   │   │   ├── edit-alkatresz.css
│   │   │   ├── gepjarmuvek.css
│   │   │   ├── kivbizonylat.css
│   │   │   ├── leltar.css
│   │   │   ├── lezart_bizonylatok.css
│   │   │   ├── mennyiseg-egysegek.css
│   │   │   ├── metallic-theme-dashboard.css
│   │   │   ├── metallic-theme.css
│   │   │   └── style.css
│   │   ├── js
│   │   │   ├── alkatreszcsoport.js
│   │   │   ├── alkatreszek.js
│   │   │   ├── beszallitok.js
│   │   │   ├── leltar.js
│   │   │   ├── mertekegység.js
│   │   │   └── rendszam.js
│   │   └── raktar
│   │       └── style.css
│   └── templates
│       ├── partials
│       │   └── sidebar.html
│       ├── registration
│       │   └── login.html
│       ├── alkatresz.html
│       ├── alkatreszcsoport.html
│       ├── base.html
│       ├── bebizonylat.html
│       ├── bebizonylatsorok.html
│       ├── beszallito.html
│       ├── dashboard.html
│       ├── edit_alkatresz.html
│       ├── home.html
│       ├── index.html
│       ├── kivbizonylat.html
│       ├── kivbizonylatsorok.html
│       ├── lekerdezes_be.html
│       ├── lekerdezes_ki.html
│       ├── lekerdezes_ossz.html
│       ├── leltar.html
│       ├── lezart_bizonylatok.html
│       ├── login.html
│       ├── mertekegység.html
│       └── rendszam.html
├── admin.py
├── apps.py
├── forms.py
├── models.py
├── serializers.py
├── settings.py
├── tests.py
├── urls.py
├── views.py
├── __init__.py
└── manage.py
```

1. ábra : A fájlstruktúra

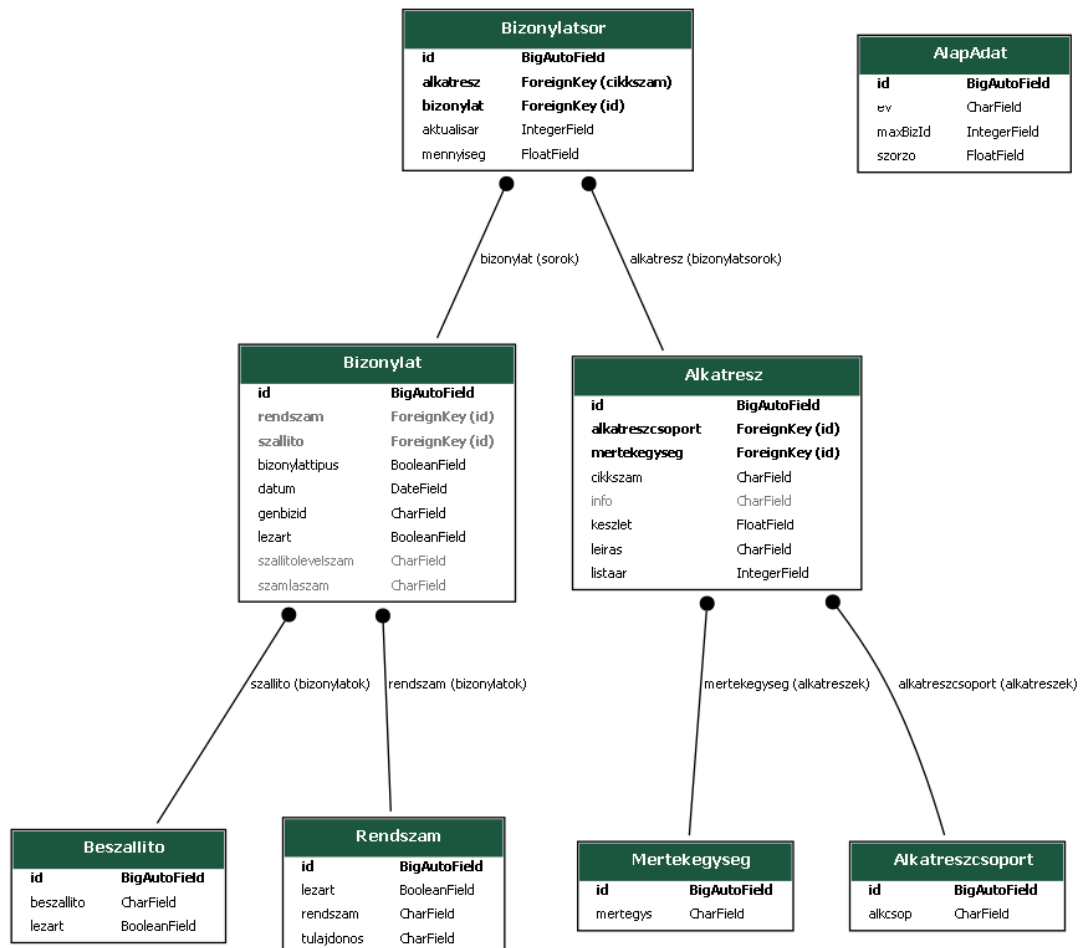
megjelenítésekre, így a reszponzivitást megoldottuk, de nem nyújt teljes mértékű felhasználhatóságot bizonyos képernyőméret alatt, de ez egyébként sem fő szempont a program céljából kiindulva.

A szervergépet egyáltalán nem terheli meg a felület futtatása. A teljes mappa 2,5 MB, melyből a képeket kivéve maga a program 0,5MB-t foglal el.

```
asgiref==3.8.1
charset-normalizer==3.4.4
Django==5.2
django-cors-headers==4.7.0
djangorestframework==3.16.0
pillow==11.1.0
reportlab==4.4.7
sqlparse==0.5.3
tzdata==2025.2
```

2. ábra: A futtatáshoz szükséges keretrendszerek

Az alkalmazás adatbázisa egy sqlite-fájl. A táblák kapcsolótábláját a 3. ábra tartalmazza.



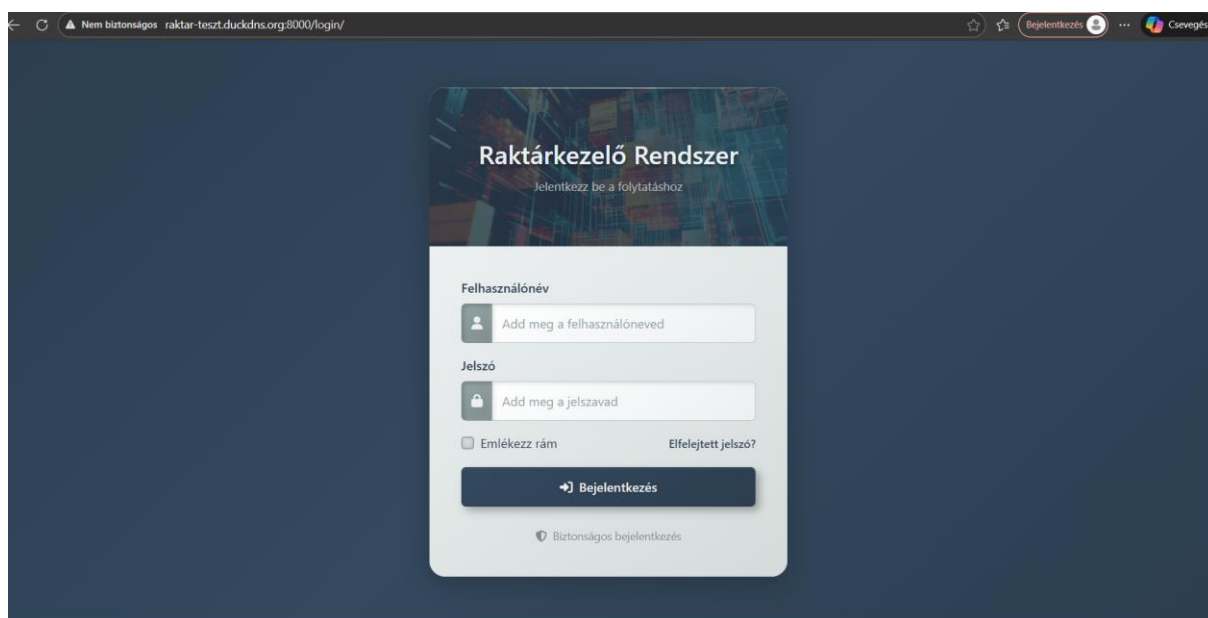
3. ábra: Az adatbázis kapcsolótáblája

A projekt bemutatása

A projektünkkel a fő célunk az volt, hogy egy való életben előforduló problémára adjunk választ, a papíralapú adminisztráció helyett egy könnyen kezelhető felületet biztosítsunk az anyagmozgások, a leltározás, a bizonylatok generálásának napi intézéséhez. A program már működés közben is tesztelve lett az elődjeként elkészített Java-s asztali alkalmazásként.

Alapvetően ipari környezetre szánjuk, ahol helyi hálózaton elérve egy szerveren fut az alkalmazás, itt van tárolva az adatbázis. A munkavégzéshez a rendszergazdák/üzemeltetők által létrehozott felhasználói fiókkal a kezdő oldalon létrehozott login-felületen lehet belépni.

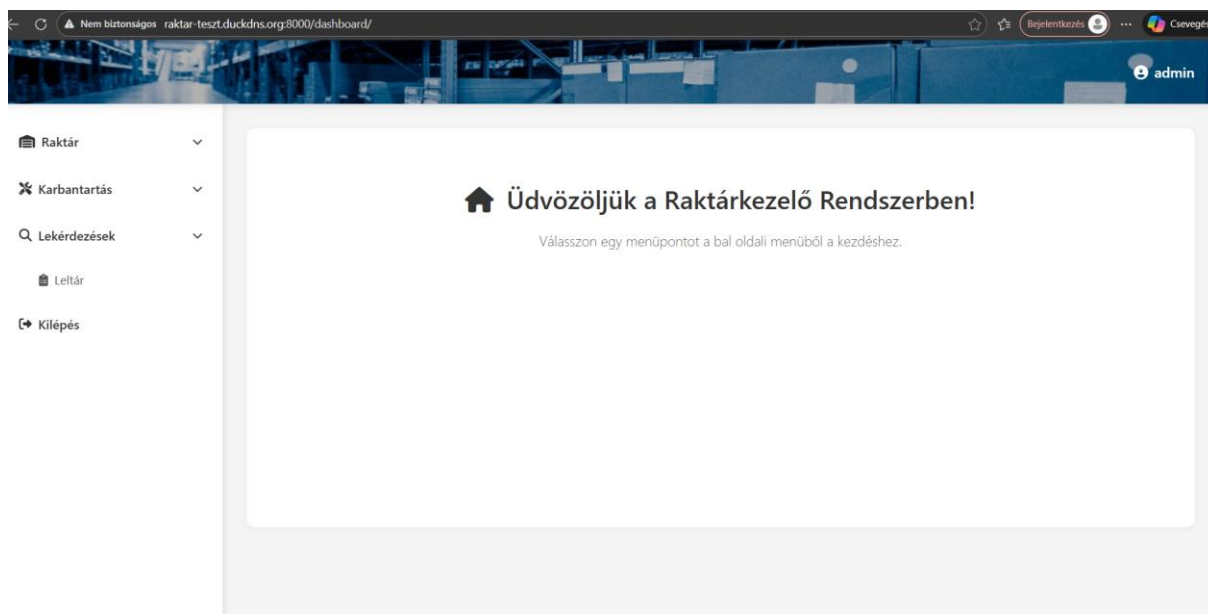
Jelenlegi állapotában egy virtuális szerveren futtatva, valamint egy általunk ideigenesen üzemeltetett Linux-szerveren is elérhető a felület.⁴



4. ábra: A bejelentkezési felület

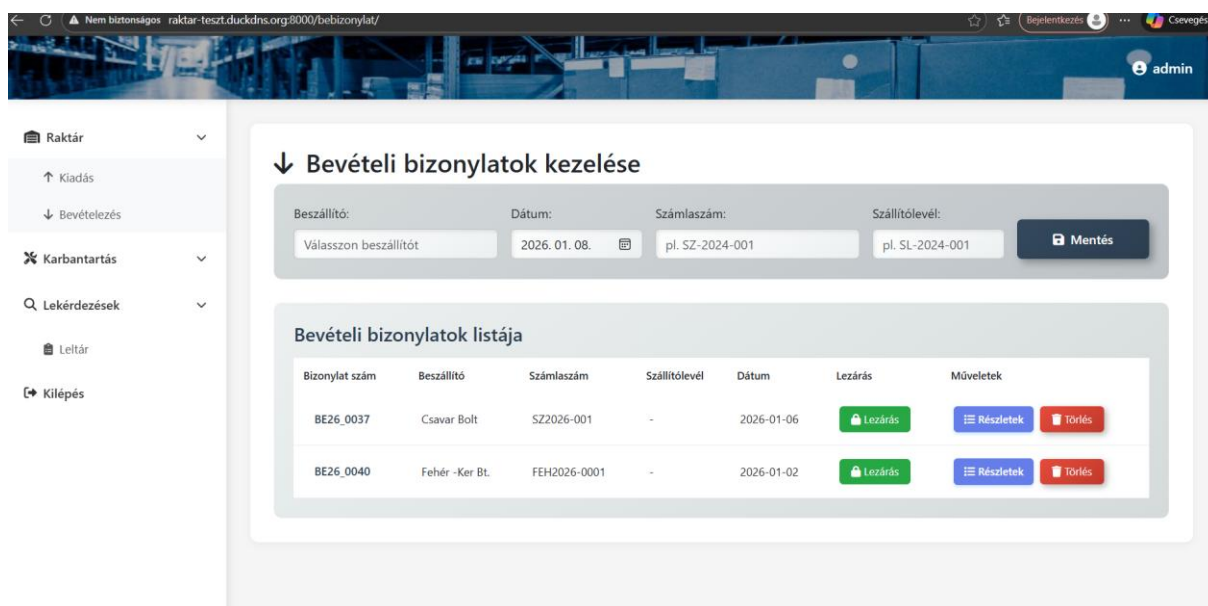
A bejelentkezést követően a főoldalra érkezünk, ahol a baloldalon elhelyezett menüsorral lehet megkezdeni a munkát. A menü parciális sidebar-ként minden oldalon megjelenik, és elérhető.

⁴ A kezdőoldal elérhetősége: <http://raktar-teszt.duckdns.org:8000/login/>



5. ábra: A kezdőfelület összezárt menüpontokkal

Alapvetően négy nagy menüpontba soroltuk az egyes funkciókat. A Raktár pontban a bevételezés és a kiadás műveletekhez köthető felületek kaptak helyet. Itt mind a bevételi, mind a kiadási oldalon tudunk bizonylatot készíteni, amik egyúttal aktív státuszt kapnak annak lezárásáig. Minden létrehozott bizonylat lezárható, törölhető, illetve a Részletekre kattintva hozzá lehet adni tételeket, melyek szerepelnek az adatbázisban.



6. ábra: A bevételi bizonylatok kezelése

A Karbantartás menüpont főként az adatbázis egyes tábláinak kezelését teszi lehetővé. Itt hozzáadhatók alkatrészek, alkatrészcsoporthoz, beszállítók, gépjárművek és mennyiségi

egységek, amelyek mentés után eltárolódnak, így a bizonylatok kezelésénél kiválaszthatók lesznek.

Alkatrészek kezelése

Cikkszám: pl. AB-12345 Mértékegység: Válasszon... Alkatrész-csoport: Válasszon...

Leírás: Rövid leírás További információ: Kiegészítő információk

Készlet: 0 Listaár (Ft): 0

Mentés

Alkatrészek listája

7. ábra: A Karbantartás menüpontjai

A Karbantartás menüben megtalálható a Bizonylatok listája is, amely a lezárt bizonylatokat listázza, ezekkel további műveletek végezhetők. Hiba esetén visszanyithatóvá tettük őket, de ez a funkció természetesen kivehető, ha a vállalat úgy kéri. Emellett bizonylatonként le lehet menteni azokat PDF illetve CSV fájlba is további felhasználásnak megfelelően.

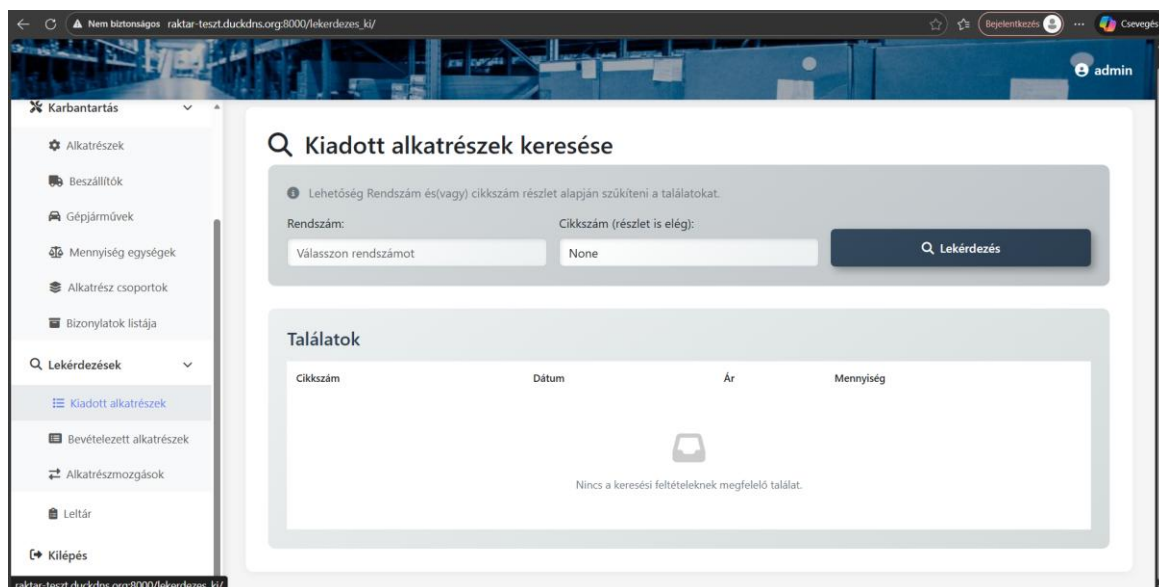
Bizonylatok listája (Lezárt)

Lezárt bizonylatok listája

Típus	Bizonylat száma	Rendszám	Beszállító	Dátum	Műveletek
↓ Bevétel	BE26_0036	-	AutóVillamosság Bt.	2026-01-07	PDF CSV Újranyitás
↓ Bevétel	BE26_0042	-	Fehér -Ker Bt.	2026-01-07	PDF CSV Újranyitás
↑ Kiadás	KI26_0038	AUDI-80	-	2026-01-05	PDF CSV Újranyitás

8. ábra: A lezárt bizonylatok listája

A Lekérdezések menüpontban a kiadott alkatrészekre rendszám alapján, a bevételezettekre beszállító alapján, valamint az alkatrészmozgásokra cikkszám alapján lehet keresni, ezek listázhatók ki itt.



9. ábra: A lekérdezések menüpont

A Leltár menüpont egy összegzése a készleten lévő alkatrészeknek, amely nagyban megkönnyíti az átláthatóságot, a korábbi, papíralapú leltárívekhez képest. A leltározási folyamathoz a lista exportálható Excel-be illetve PDF formátumba, ami az alapértelmezett mentési útvonalra a kívánt fájlt létrehozza egy egyszerű kivitelben.

Alkatrész	Alkatrész-csoport	Cikkszám	Mennyiség	Mértékegység	Egységár	Összár
vékonyfalu	vasanyag	zártszelvény	18.0	Kg.	3000 Ft	54000 Ft
rakodó kapaszkodó	tipus nélkül	kapaszkodó	34.0	db.	18000 Ft	612000 Ft
225-ös csavar	kötőelem	szerkezeti csavar	298.0	db.	30 Ft	8940 Ft
A10-es polc	tipus nélkül	fékbetét (27801)	5.0	db.	500 Ft	2500 Ft
B3 polc	elektromos	gépkocsi izzó (H7)	8.0	db.	1500 Ft	12000 Ft
B2 polc	elektromos	gépkocsi izzó (H4)	35.0	db.	65 Ft	2275 Ft
3-as lemezszekrény	kötőelem	csavar (lemezszekrény)	500.0	db.	9 Ft	4500 Ft

10. ábra: A Leltár menüpont

Végezetül a kilépés menüpont kijelentkezteti a felhasználót a login oldalra visszanavigálva.

A backend rész működését tekintve több részre bonthatjuk. Az adatbázis tábláit a `models.py` fájlban hoztuk létre, az ezekkel kapcsolatos műveleteket a `migrations` mappában található `python` fájlokban definiáltuk. Az `urls.py` fájlban rögzítettük az oldalak elérési útjait, a `views.py` pedig a `function`-okat adtuk meg. A frontend részről az oldalak tartalma `HTML` fájlokban `template`-ekként kerültek létrehozásra, melyekben a vonatkozó oldalakhoz tartozó `JavaScript` és `CSS` fájlok kerültek meghívásra.

A projektmunka során a két fős csapatunk munkamegosztása egyszerű feladat volt: Kiss Róbert a backend rész, Szatmári Gábor pedig a frontend részt vállalta el. Mindketten az őszi időszakban láttunk munkához, egy-egy alap létrehozásával. Nehézséget talán a két szál összefésülése jelentette. Menet közben is találtunk ki hozzá újabb funkciókat, például a szinte kész állapothoz lett hozzáadva a bizonylatok `CSV`-be történő exportálási lehetősége, addig csak `PDF`-ben gondolkodtunk.

Aminek a megoldásával több időt kellett töltenünk az az adatbázis-műveletekhez kötődő műveletek logikája. Alapvetően könnyen működnek az egyes funkciók: mentéssel hozzáadunk, törlésekkel kitöröljük. Azonban ha az adatbázisból kitörölünk egy mértékegységet, de bizonylatok már generálva lettek azok felhasználásával, az például helytelen működéshez vezetett. Ebből kiindulva lépésről lépésre végig kellett haladni az egyes funkciókon, hogy az ilyesféle rendellenességeket kiküszöbölhessük.

Ezek megoldására felugró `messagebox`-okat készítettünk. A véletlen törlések elkerülése érdekében a frontend oldalon kerültek bele felugró megerősítő üzenetek, a backend részről pedig a `CASCADE` törléseket kellett végig tekinteni, hogy jelezze, ha már felhasznált elem kerülne törlésre a Karbantartás menüpont valamely alpontjánál.

```
54 mertkegyseg = models.ForeignKey(Mertekegyseg, on_delete=models.PROTECT, related_name="alkatreszek")
55 alkatreszcsoport = models.ForeignKey(Alkatreszcsoport, on_delete=models.PROTECT, related_name="alkatreszek")
```

11. ábra: A `CASCADE` törlés elkerülése

```
66 <td>
67 <form method="POST" action="{% url 'raktar:deleteBeszallitoById' beszallito.id %}" style="display: inline;">
68   {% csrf_token %}
69   <button type="submit" class="btn btn-delete" onclick="return confirm('Biztosan törli?')>
70     <i class="fas fa-trash me-2"></i>Törlés
71 </button>
```

12. ábra: A véletlen törlésre figyelmeztető üzenet

A két fejlesztés összefésülésére a Github verziókövető felületet használtuk munkánk során, hogy összefésüljünk a saját munkánkat a Master verzióval. Itt a Commit-ok és Pushok-on kívül megtanultuk a Branch-ok és a Pull request-ek

Tervek a további fejlesztésekre

A felületet tervek szerint további módosítások után Kiss Róbert használni tervezi majd, így fontos a működőképesség és a funkcionalitás.

A projektben több ismétlés felfedezhető, ezek egyszerűsítése, összevonása mindenképpen megoldandó feladat. Itt lehet említeni az egyes oldalakon a címek és táblázatok címeinek hasonlóságát, ezt egyszerűen orvosoljuk majd. A duplikációknál említhető az is, hogy az aloldalakhoz mindhez készült külön CSS fájl, a formázásokat meg lehetne majd úgy írni, hogy 2-3 db CSS-re hivatkozzon csak az adott oldal.

A nagyobb lélegzetvételű feladatok egyike az, hogy az alkatrészekhez képeket társítsunk. A táblázatban egy oszlopban egy kis méretű kép jelenne meg thumbnail-ként, rákattintva, vagy a kurzort fölé helyezve pedig nagyban is megtekinthető lenne. Ezek eltárolása az adatbázisban izgalmas munka lesz majd. Ez a funkció azonban nagyon hasznos lenne, ugyanis vannak olyan speciális alkatrészek, amik név, vagy cikkszám alapján sem feltétlenül egyértelműek, így azok fizikai felkeresését nagyban segíteni ez az újítás.

Szintén hasonló okokból, de talán még bonyolultabb irány lehetne egy interaktív térképes felület, ami a tárolóegységeket felülnézetből vizuálisan is mutatná, hogy mely tárolóegység hol helyezkedik el a raktárban. Ennek kibővítéseként az ott tárolt alkatrészek, anyagok felsorolása is megjelenne a „térképes” nézeten, ami a Leltár menüpont egy funkciója lenne. Ez kimondottan hasznos egy külsőszakember által végzett leltározásnál.

További fejlesztési irányként jelöltük ki, hogy a lezárt bizonylatok listája menüpontnál legyenek csoportosan letölthetők a CSV fájlok. Ehhez egy csúszka vagy dátumválasztók segítségével időintervallum lesz megadható, amely időszakban keltezett bizonylatokra külön rá lehet szűrni, azok CSV fájljai csoportosan lesznek letölthetők. Ez a funkció a vállalat pénzügyi részlegének munkáját segíteni, ugyanis a csoportos CSV fájlok a könyvelés által használt programokkal beolvashatók, így lényegesen nőne a hatékonyság illetően is.

Terveink között szerepel egy olyan egyszerűen kivitelezhető lehetőség, ami a készletgazdálkodásnak nyújt segítséget. Az egyes anyagok, vagy alkatrészek esetében a felvételkor beállítható lesz egy kritikus mennyiség, amit elérve jelzést küld akár a felületen,

akár e-mail-ben, hogy a beszerzés felé időben jelezhető legyen az igény azok pótlására. Így elkerülhető az, hogy a bizonylatok menüpontnál olyan mennyiséget adnánk ki, amivel fizikailag nem is rendelkezünk.

Az e-mail-es értesítés szóba hozásával egy alapvető fejlődési irány is megfogalmazódott: a userek hozzáadása a rendszergazdai oldalon. Ehhez külön oldalt tervezünk létrehozni, amelyen a rendszergazda felhasználói fiókokat tud létre hozni, jogosultságokat tud kiosztani, törléseket tud végezni. Erre jelenleg is van lehetőség a Django keretrendszer beépített admin felületén keresztül.

Összegzés

A projektmunka során egy olyan raktárkezelő alkalmazást valósítottunk meg, amely egy valós, a mindennapi munkavégzés során tapasztalt problémára ad gyakorlati és korszerű megoldást. A papíralapú adminisztráció kiváltásával célunk egy átláthatóbb, gyorsabb és megbízhatóbb rendszer létrehozása volt, amely hatékonyan támogatja az anyagmozgások, bizonylatok és készletadatok kezelését ipari környezetben.

fejlesztés során tudatosan építettünk a Mechwart András Gépipari és Informatikai Technikumban elsajátított ismeretekre, és azokat egy már korábban létező, Java-alapú asztali alkalmazás tapasztalataival ötvöztük. A Django-alapú backend, valamint a JavaScript–Bootstrap–CSS technológiákra épülő frontend lehetővé tette egy stabil, jól strukturált és továbbfejleszthető webes rendszer kialakítását. A funkciók megvalósítása közben kiemelt figyelmet fordítottunk az adatbiztonságra, a felhasználói hibák minimalizálására és a raktári folyamatok logikájának helyes leképezésére. A csapatunk meglátásunk szerint jól működött, konstruktívan álltunk az akadályok leküzdéséhez. Ezzel valamelyest belekóstolhattunk, hogy milyen egy munkán „team”-ként dolgozni. A projekt készítése során az agilis fejlesztés elvei mentén dolgoztunk. Létrehoztunk egy működő programmagot, majd ezt bővítettük, kapcsolunk hozzá új tartalmakat és megjelenéseket. Ehhez a Github verziókövető-rendszer alapvető megismerésére volt szükség, ami későbbi, esetleges IT-s projektek megvalósításához elengedhetetlen tudást jelent. Összesen 37 Commit-on keresztül született meg a vizsgamunkánk.

A munka során volt lehetőségünk elmélyülni a webfejlesztés útvesztőiben, felismertük, hogy csak a felszínt kapargatjuk: minél több ismeretre teszünk szert, annál jobban rálátunk arra, hogy mennyi továbblépési lehetőség van. Így a képzésünk során megszerzett tudás mellett betekintést

nyertünk a jövőbeni fejlődésünk tárházába, amennyiben próbálunk lépést tartani a hihetetlenül gyorsan fejlődő IT-szektorral, és annak dinamikus változásaival.

Összességében a projekt nemcsak egy működőképes raktárkezelő alkalmazás elkészítését eredményezte, hanem szakmailag is megerősítette tudásunkat a webes alkalmazásfejlesztés, az adatbázis-kezelés és a csapatban történő munkavégzés területén. Meggyőződésünk, hogy az elkészült rendszer jó alapot biztosít a későbbi bővítésekhez, és valós környezetben is hatékonyan alkalmazható megoldást nyújt.

Irodalom

- Bootstrap hivatalon oldala

<https://getbootstrap.com>

(Utoljára megnyitva: 2026.01.08.)

- Django hivatalos oldala

<https://www.djangoproject.com>

(Utoljára megnyitva: 2026.01.08.)

- JavaScript hivatalos oldala

<https://www.javascript.com>

(Utoljára megnyitva: 2026.01.08.)

- Plötz Anita-Varga János: Digitalizáció hatása a logisztikára

https://kgk.uni-obuda.hu/wp-content/uploads/2025/08/VF_TK_25_2_Digitalizacio_hatasa_a_logisztikara.pdf

(Letöltve: 2026.01.08.)

- W3schools hivatalos oldalán a vonatkozó részek

<https://www.w3schools.com/django/index.php>

https://www.w3schools.com/bootstrap/bootstrap_ver.asp

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

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

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

(Utoljára megnyitva: 2026.01.08.)

- Webiskola

<https://webiskola.hu>

(Utoljára megnyitva: 2026.01.08.)