**Vizsgaremek**

Baczur Zsolt, Békés Zoltán és Magyar Tamás

**Pécs, 2025**

**Baranya Megyei Szakképzési Centrum**

**Baranya Megyei SzC Radnóti Miklós Közgazdasági Technikum**

**sERPa vállalatirányítási rendszer**

Készítették: Baczur Zsolt, Békés Zoltán és Magyar Tamás

**PÉCS**

**2025**

Tartalomjegyzék

[I. Bevezető, a feladat rövid ismertetése 4](#_Toc196304552)

[II. A fejlesztői dokumentáció 5](#_Toc196304553)

[1. Fejlesztőkörnyezet ismertetése 5](#_Toc196304554)

[2. Adatszerkezet ismertetése 5](#_Toc196304555)

[3. Az adatbázis tervezése 6](#_Toc196304556)

[4. Normalizált (Logikai) ERD létrehozása 8](#_Toc196304557)

[5. Algoritmusok bemutatása 8](#_Toc196304558)

[6. Tesztelési dokumentáció 13](#_Toc196304559)

[III. API végpontok 16](#_Toc196304560)

[IV. A felhasználói dokumentáció 17](#_Toc196304561)

[1. A program általános specifikációja 17](#_Toc196304562)

[2. Rendszerkövetelmények 18](#_Toc196304563)

[3. A program telepítése 18](#_Toc196304564)

[4. A program használatának részletes leírása 18](#_Toc196304565)

[V. Továbbfejlesztési lehetőségek 27](#_Toc196304566)

[VI. Összegzés 28](#_Toc196304567)

[1. Forrásmegjelölés 28](#_Toc196304568)

# Bevezető, a feladat rövid ismertetése

A vizsgaremek témájaként egy saját fejlesztésű ERP rendszer (vállalatirányítási rendszer) létrehozása mellett döntöttünk. A választásunkat az motiválta, hogy a jelenlegi piacon elérhető megoldások többsége nem biztosít modern, átlátható felületet, illetve kezelhetőségük bonyolult. Szerettünk volna ezekre a problémákra egy korszerű, minimalista és felhasználóbarát alternatívát nyújtani.

A fejlesztés során felhasználtunk szinte minden korábban tanított fejlesztői eszközt és programnyelvet, illetve új ismeretekre is szert tettünk. A projekt gyakorlati szempontból is hasznosnak bizonyult, mivel nemcsak a technikai tudásunkat mélyítette el, hanem a csapatmunkában és a problémamegoldásban is sokat fejlődtünk. Célunk volt olyan modern, bővíthető rendszert megalkotni, amely akár továbbfejlesztve, később kiadásra is kerülhet.

A projekt során számos olyan területtel foglalkoztunk, amelyek lehetőséget adtak arra, hogy gyakorlatban is alkalmazzuk az iskolai tanulmányaink során megszerzett tudást. Emellett több új technológiát is kipróbáltunk, és gyakran kerültünk olyan helyzetbe, ahol önállóan kellett megoldást találnunk különböző hibákra vagy problémákra. Ezek a kihívások hozzájárultak szakmai fejlődésünkhöz, és megerősítettek bennünket abban, hogy képesek vagyunk összetett informatikai rendszerek megtervezésére és kivitelezésére.

A fejlesztés során a csapatmunkára is különös figyelmet fordítottunk, a feladatokat felosztottuk frontend, backend és adatbázis részre, miközben folyamatosan egyeztettünk egymással. A közös tervezés, ötletelés és a felmerülő problémák közös megoldása nagyban hozzájárult a projekt sikeres megvalósításához. A munka során különféle eszközöket és platformokat használtunk – például Discord-ot a kommunikációra és megbeszélésekre, GitHub-ot a verziókezeléshez, Visual Studio Code-ot a fejlesztésre, valamint phpmyadmin-t az adatbázis kezelésre.

# A fejlesztői dokumentáció

## Fejlesztőkörnyezet ismertetése

Használt hardver:

A fejlesztést a csapattagok saját otthoni számítógépein végezték.

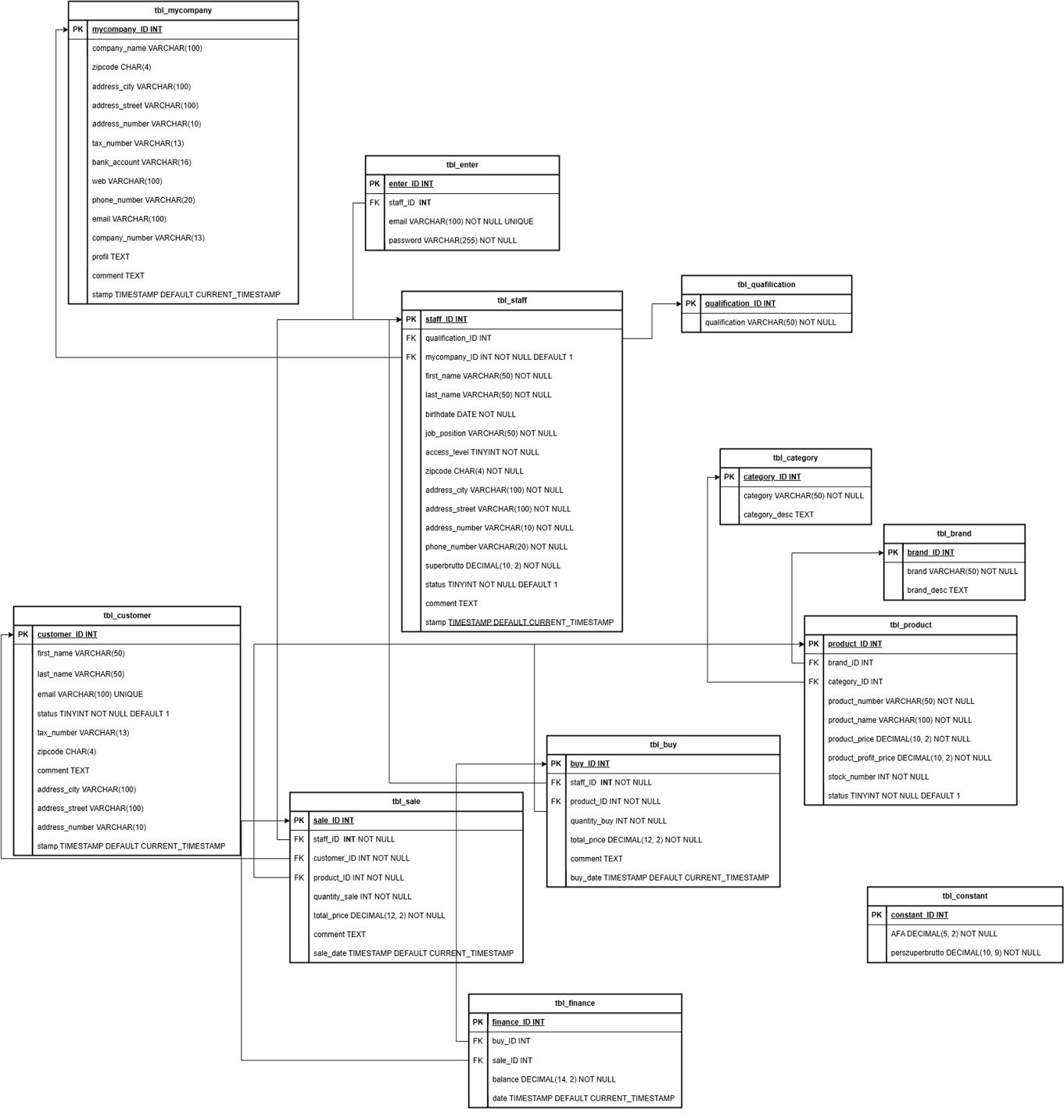
Használt szoftverek:

* Frontend: Visual Studio Code, HTML, CSS, JavaScript, TailwindCSS
* Backend: XAMPP (Apache, PHP 8.x), Visual Studio Code
* Adatbázis: MySQL, phpMyAdmin
* Verziókezelés: Git, GitHub
* Kommunikáció: Discord
* Tesztelés: POSTMAN

A fent említett eszközök kiválasztását a következő szempontok indokolták: ingyenes elérhetőség, platformfüggetlenség, tanulhatóság, valamint a korábbi tanulmányaink során megszerzett tapasztalat.

## Adatszerkezet ismertetése

Az alkalmazás MySQL alapú relációs adatbázist használ. A rendszer adatmodellje normalizált szerkezetet követ, a redundancia csökkentése érdekében. Az alábbi főbb táblák kerültek kialakításra:

* Tbl\_brand: a termékek márkája
* Tbl\_buy: a vásárlások összege
* Tbl\_category: a termékek kategóriája
* Tbl\_constant: az ÁFA és perszuperbruttó állandó értékei
* Tbl\_customer: a partnerek és adataik
* Tbl\_enter: a felhasználók e-mail címe és hash-elt jelszavai
* Tbl\_finance: az adott cég pénzügyi adatai
* Tbl\_mycompany: az ERP rendszert használó cég adatai
* Tbl\_product: a termékek nevei, beszerzési és eladási árai
* Tbl\_qualification: végzettségek
* Tbl\_sale: az értékesítés bevételei és kiadásai
* Tbl\_staff: az alkalmazottak adatai, többek között név, email, belépési szint stb.

draw.io

Ezek közül nem minden került felhasználásra, később részletesebb leírás található, hogy milyen szerepeük lesz a jövőben.

## Az adatbázis tervezése

Az eltárolt adatok: munkavállalók adatai, raktár adatai, vásárlók adatai határozták meg a modell fő tábláinak a létrehozását. A tbl\_customer a vásárlók adatait a tbl\_staff a munkatársak adatait és a tbl\_product a raktározott áruk adatait, mennyiségét tárolja. A tranzakciókat kétfelé bontottuk, így jött létre a tbl\_sale amely idegenkulccsal kapcsolódik a vásárlók, munkatársak, raktározott áruk tábláihoz és így rögzíti ,hogy melyik munkatárs , melyik vevőnek, melyik árut adta el és hány darabot. Külön trigger számolja az eladásból származó bevételt.

A tbl\_buy tábla köti össze a munkatársak és raktározott áruk tábláit így rögzíti hogy, melyik munkatárs, melyik árut rendelte meg a cég számára, raktár feltöltés céljából és itt is külön trigger számolja a kiadást.

Az áruk darabszámát, közvetlenül a tbl\_product tábla: stock\_number nevű mezője tárolja, értékének a változásait eladás vagy rendelés esetén külön trigger számolja. A vállalat pénzének (forgó-tőkéjének) a nyomonkövetését a tbl\_finance tábla tárolja, rekordok formájában, minden egyes a tbl\_sale táblábaban rögzített eladás egy triggert vált ki amely fizikailag kiszámítja és tárolja a pénz pillanatnyi értékét a balance attribútumban. A tbl\_buy táblában rögzített rendelés hasonlóképpen.

A balance származtatott attribútum értéke kiszámolható hiszen az előzetesen végrehajtott tranzakciók, előjeles összege lesz. (eladás: pozitív, rendelés/vásárlás: negatív). Mivel bármikor kiszámolható, ezért a fizikai letárolása nem szükséges és redundanciát okoz.

Ugyanakkor ha, nem tárolnánk le a pénzmennyiséget fizikailag a balance attribútumban, akkor a balance érték lekérésekor azt újra és újra ki kellene számolnia a programnak, konkrétan esetleg sok ezer-százezer tbl\_buy és tbl\_sale ben tárolt tranzakció végigszámolása, minden egyes lekérésnél csökkentené a lekérdezés sebességét és így az egész vállalatirányítási rendszer program sebességét.

Ezért a gyorsabb sebesség érdekében a tbl\_finance tábla balance mezőjében, ténylegesen fizikailag is letároljuk azt, így az egyszerűen és gyorsan lekérdezhető vagy visszanézhető az értéke, minden tranzakciónál.

Egy további táblát is létrehoztunk: tbl\_constant néven, amiben a számításokhoz szükséges értékeket tároltuk pl: áfa értéke, a szuperbruttó és bruttó számolásához szükséges konstanst. Érdekesség ,hogy a tbl\_constant nem kapcsolódik közvetlen idegenkulccsal a többi táblához, hanem a számítás során kell hivatkozni pontosan a számításhoz szükséges konstansra.

A koncepcionális ERD tervezésekor fontos szempont volt változatos kapcsolatok és attribútumok bemutatása a konkrét projektben, ezért összetett és származtatott tárolt illetve származtatott nem tárolt attribútum is megjelent a modellben.

A származtatott attribútumok értékét , triggerrel vagy lekérdezéssel a már meglevő adatokból ki lehet számolni. Szempont volt az is ,hogy az adatbázis ne legyen feleslegesen túl bonyolult , ne tartalmazzon feleslegesen sok attribútumot, táblát hanem inkább a vállalat irányítási rendszerhez szükséges adatokat tárolja el, ugyanakkor az általánosan előfordulható fő tábla, attribútum, kapcsolat típusokat mutassa be.

## Normalizált (Logikai) ERD létrehozása

A koncepcionális ERD normalizálása, során a fellépő redundanciákat megszüntettük, vagy az ésszerűség határain belül csökkentettük, új táblák , kapcsolatok és attribútumok létrehozásával.

A tbl\_staff bizalmas adatai átkerültek a tbl\_enter-be amely 1:1 kapcsolatban van a tbl\_staff táblával, a szétválasztás nem a redundancia csökkentés miatt kell, csak a logikusabb csoportosítás miatt került a munkavállalók email és password attribútuma a tbl\_enter-be amely adatokra csak a megfelelő jogosultsággal rendelkező munkatársak látnak rá. Mivel a qualification attribútum sokszor ismétlődő értékeket: “Alapfokú”, “Középfokú”, “Felsőfokú” értékeket tartalmaz ezért a redundancia csökkentés érdekében új táblát lehet létrehozni, ami nem kötelező ,de gyakran ismétlődő értékeknél érdemes megtenni, ezért létrehoztuk a tbl\_qualification táblát. A kapcsolat N:1 (több az egyhez) ezért a tbl\_qualification tábla elsődleges kulcsa (PK) kerül át a tbl\_staff -ba idegenkulcsként (FK).

[Az ERD diagramm ide kattintva érhető el](https://github.com/baczur-zsolt/serpa/blob/master/Database/cms_staff_20250410.png)

## Algoritmusok bemutatása

A rendszerben több algoritmus is megtalálható, melyek kulcsfontosságú szerepet játszanak a program biztonságos működésében.

#### Bejelentkezési rendszer algoritmusa

A bejelentkezési logika az AccessController osztály login() metódusában valósul meg.

A bejelentkezéskor a felhasználónak meg kell adnia az e-mail címét és jelszavát. A jelszavakat bcrypt algoritmussal hash-eltük.

Főbb funkciói:

* Jelszó hash-elés SHA2-256 algoritmussal
* Többszintű hozzáférés-kezelés (access\_level)
* Session-azonosító és időzített munkamenet-figyelés
* JSON válaszkódok a frontend kommunikációhoz

#### Algoritmus lépései:

1. A felhasználó megadja e-mail címét és jelszavát a bejelentkezési űrlapon.
2. A rendszer ellenőrzi, hogy az e-mail és jelszó mezők szerepelnek-e a POST kérésben.
3. Az SQL lekérdezés SHA2 függvényt használ a jelszó hash-eléséhez (SHA2(jelszó, 256)).
4. A Db::Select() metódus lekérdezi a tbl\_enter táblából a felhasználót.
5. Ha a felhasználó létezik:
   * Elindul a munkamenet (session\_start()).
   * A staff\_ID és az access\_level eltárolásra kerül a session változókban.
   * Rögzítésre kerül a munkamenet azonosítója és az időbélyeg (session\_id(), time()).
   * A szerver 202-es HTTP státuszkóddal válaszol (Accepted).
6. Ha nincs találat: 401-es hibakód (Unauthorized), hibaüzenet visszaküldése JSON formátumban.
7. Ha hiányzik az e-mail vagy jelszó mező: 400-as hibakód (Bad Request).

PHP kód részlet a login() metódushoz:



ábra 1

#### Űrlapvalidáció algoritmusa

A rendszer két szinten végzi az űrlapadatok ellenőrzését: kliensoldalon JavaScript-tel, illetve szerveroldalon PHP-val, ezáltal biztosítva a felhasználói élményt és a biztonságot egyaránt.

Kliensoldali validáció (login.js)

A loginForm elküldésekor az alábbi ellenőrzések történnek meg a JavaScript által:

Ellenőrzési lépések:

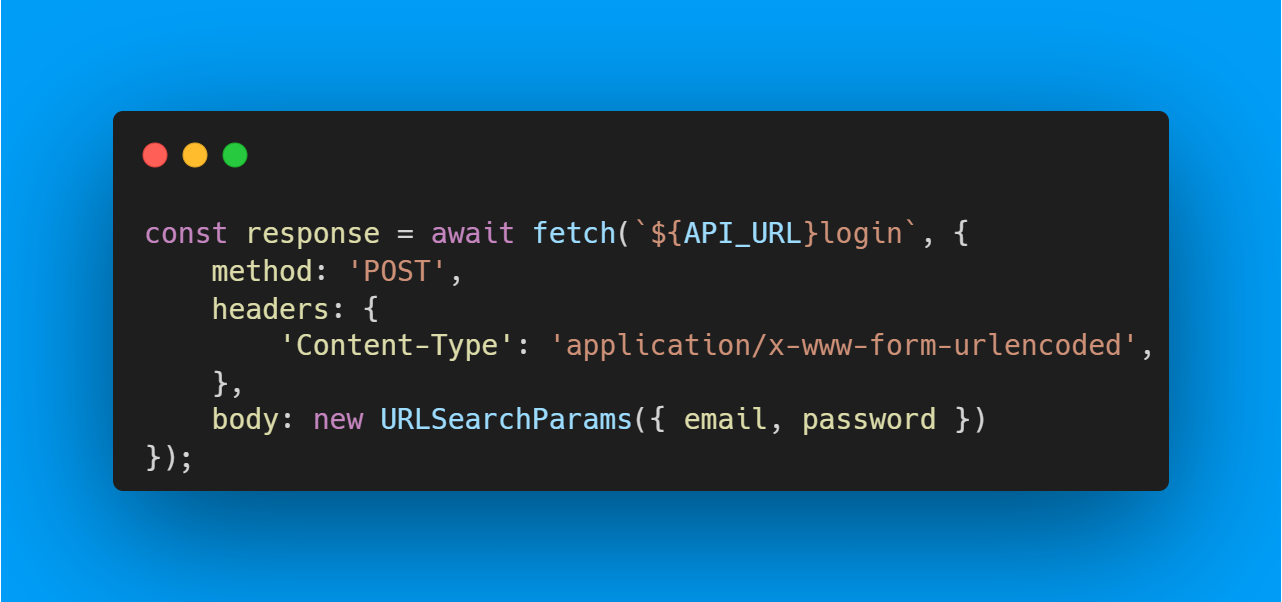
1. Az email és password mezők értékei kiolvasásra és trim()-el megtisztításra kerülnek.
2. Ellenőrzi, hogy a mezők nincsenek-e üresen hagyva.
3. Ha hiba van, megjeleníti a kapcsolódó hibaüzenetet, és megszakítja az adatküldést.
4. Ha nincs hiba, az adatok fetch segítségével továbbításra kerülnek a backendre POST kérésként.

Javascript kód részlet a bejelentkezéshez:

ábra 2

Hálózati kérés és válaszkód-kezelés:

* A fetch kérés application/x-www-form-urlencoded formátumban küldi az adatokat.
* A válasz alapján történik a további működés (átirányítás vagy hibaüzenet).



ábra 3

Szerveroldali validáció

Az AccessController::login() PHP metódusban a szerver oldalon is történik ellenőrzés:

* Ellenőrzi, hogy az email és password mezők be vannak-e küldve.
* Ha hiányoznak, 400 Bad Request választ ad vissza.
* A jelszó SHA2-256 hash-elésen esik át, így a szerver csak titkosított formában kezeli azt.



ábra 4

Ez az algoritmus biztosítja, hogy a felhasználók ne tudjanak üres mezőkkel bejelentkezni, és hogy az érzékeny adatokat a backend csak validált és titkosított formában fogadja. Az aszinkron adatküldés és válaszkód-kezelés révén a felhasználói élmény gördülékeny, a hibák azonnal visszajelzésre kerülnek.

#### Reszponzivitás kezelése algoritmus

A rendszer reszponzív kialakítású, vagyis különböző eszközméretekhez alkalmazkodik. Ezt Tailwind CSS keretrendszerrel valósítottuk meg, amely segíti az elemek különböző képernyőméret szerinti megjelenítését (sm:, md:, lg:, xl: osztályok segítségével).

Lépések:

1. Az oldalelemek külön Tailwind osztályokat kapnak az eszköz méretétől függően.
2. Az elemek egymás alá kerülnek kisebb képernyőn, és egymás mellé nagyobbon.
3. A menüsáv mobilon lenyitható, míg asztali verzióban oldalsávként jelenik meg.

#### Frontend komponensek újra töltése algoritmus, dinamikus frissítés

A rendszer támogatja a dinamikus tartalomfrissítést oldal újra töltés nélkül, így gördülékenyebb és reszponzívabb felhasználói élményt nyújt. Ezt JavaScript és a fetch() API segítségével érjük el, amely aszinkron módon kommunikál a backenddel, és a válaszként kapott JSON-adatok alapján frissíti az érintett komponenseket a DOM-ban.

Folyamat lépései:

Űrlap beküldése gombra kattintáskorA gombhoz egy eseményfigyelő van csatolva, amely megakadályozza az alapértelmezett űrlapküldést (event.preventDefault()), így nem történik teljes oldalfrissítés.

Adatok begyűjtése:

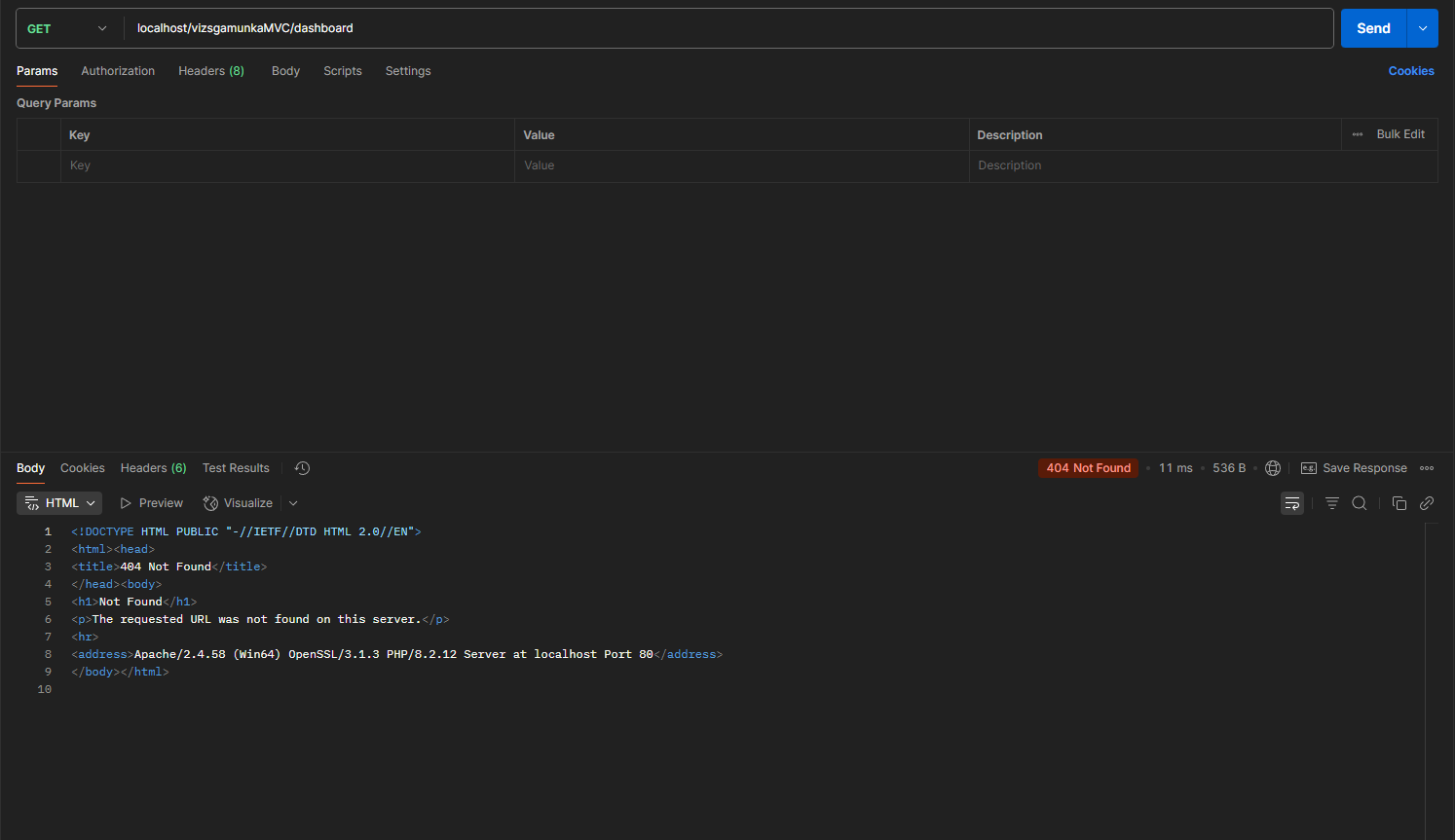
* A JavaScript lekéri az összes mező értékét a DOM-ból.
* POST kérés a szerver feléAz összegyűjtött adatok JSON formátumban kerülnek elküldésre a backend felé egy fetch() POST kérés segítségével. A válasz egy JSON objektum, amely visszaigazolja a sikeres mentést.
* Frontend frissítésA válaszként kapott adatok hozzáadásra kerülnek a meglévő táblázathoz, majd a renderTable() függvény meghívásával a felhasználói felület frissül – az új rekord azonnal megjelenik a táblázatban.

## Tesztelési dokumentáció

A tesztelést POSTMAN segítségével végeztük

Több tesztesetet is megvizsgáltunk, hogy miként regál a program.

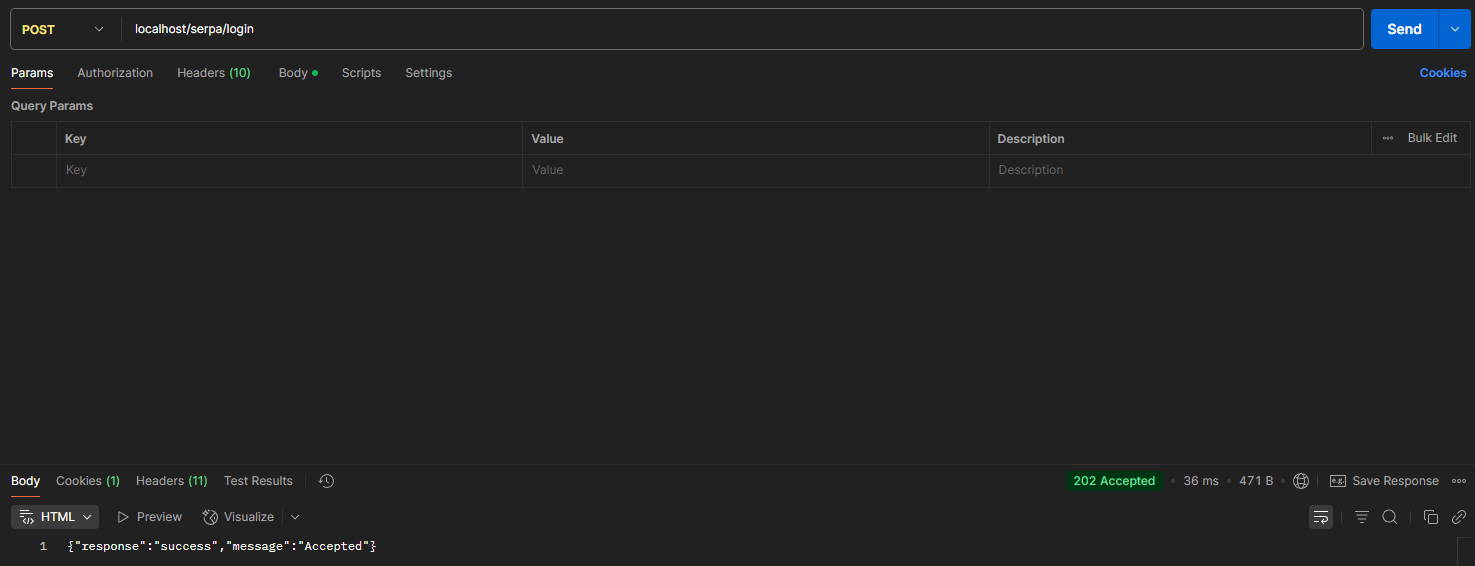
Első tesztesetünknél azt vizsgáltuk, hogy mi történik akkor, ha valaki cask simán az URL beírásával próbálja meg elérni mondjuk az irányítópultot:



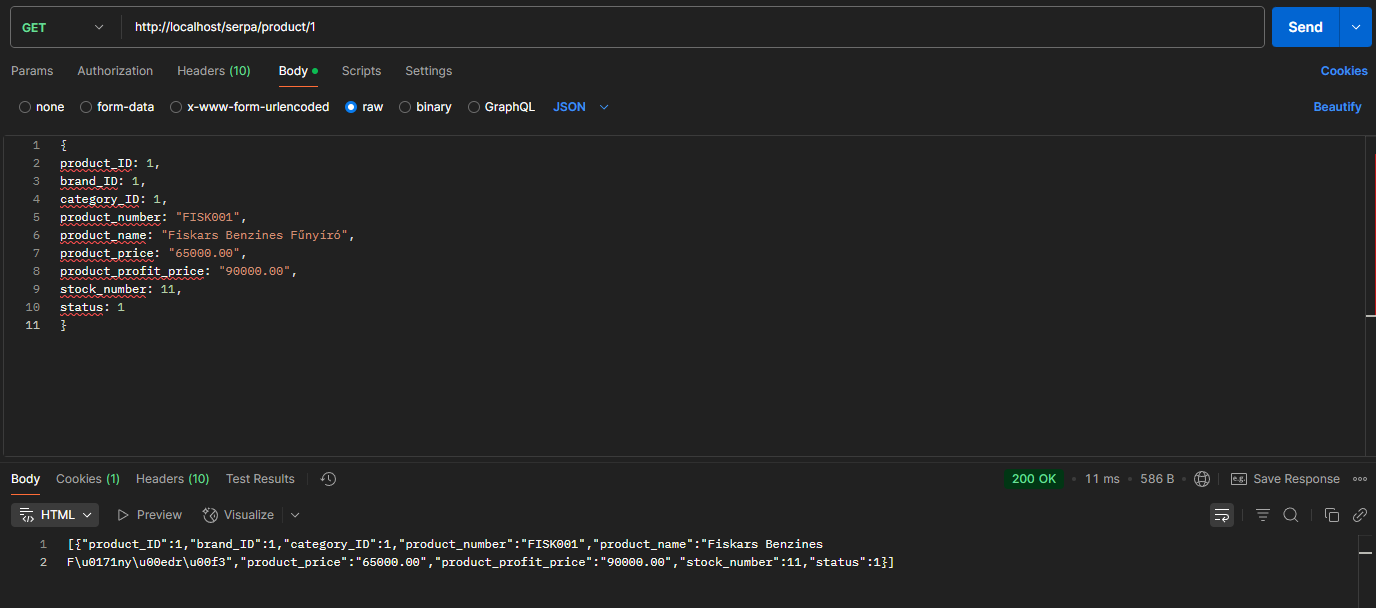
ábra 5

Ez a teszt egy 404 Not found hibakóddal tér vissza, ami azt jelenti, hogy bejelentkezés nélkül nem elérhető az irányítópult, így meggátolva az illetéktelen behatolást a programba.

Sikeres bejelentkezést követően már megtudjuk nyitni az adott oldalt akár az URL beírásával is, a sikeres bejelentkezés esetén egy 202 Accepted státusszal tér vissza a kérés.



ábra 6

Egy adott termék lekérése esetén is megvizsgáltuk a programok, a következőképpen:  


ábra 7

Itt jól láható módon a válasz egy200 OK válasszal, valamin az adott termék (jelen esetben 1-es ID-val rendelkező) adataival tér vissza.

# API végpontok

Az alábbi táblázat a rendszerben használt legfontosabb API-végpontok közül tartalmaz néhányat példaként, metódusukkal, bemeneti adatokkal, válaszformátummal, céljukkal és a szükséges jogosultsági szinttel együtt.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Végpont URL | Metódus | Adatok (input) | Válasz (output) | Funkció / Cél | Jogosultsági szint |
| /login | POST | email, password | 202 / 400 / 401 | Bejelentkezés | – |
| /logout | GET | - | Átirányítás / státuszkód | Kijelentkezés | 1+ |
| /username | GET | - | JSON (felhasználónév) | Felhasználónév lekérése | 1+ |
| /access\_level | GET | - | JSON (jogosultság szintje) | Jogosultság lekérése | 1+ |
| /sale | GET | id | JSON (eladás adatai) | Eladás lekérdezése | 1+ |
| /sale | POST | JSON (vevő, termék, ár...) | JSON, státuszkód | Új eladás | 1+ |
| /sale | PUT | id, JSON frissítés | JSON, státuszkód | Eladás módosítása | 1+ |
| /sale | DELETE | id | 204 No Content | Eladás törlése | 1+ |
| /product | GET | id | JSON (termék adatok) | Termék lekérése | 2+ |
| /product | POST | JSON (név, ár, készlet...) | JSON, státuszkód | Új termék | 2+ |
| /product | PUT | id, JSON | JSON, státuszkód | Termék módosítása | 2+ |
| /product | DELETE | id | 204 No Content | Termék törlése | 2+ |
| /partner | GET | id | JSON (partner adatok) | Partner lekérése | 2+ |
| /partner | POST | JSON (név, cím, típus...) | JSON | Új partner | 2+ |
| /partner | PUT | id, JSON | JSON | Partner módosítása | 2+ |
| /partner | DELETE | id | 204 No Content | Partner törlése | 2+ |
| /employee | GET | id | JSON (alkalmazott adatok) | Alkalmazott lekérése | 3 |
| /employee | POST | JSON (név, státusz, cím...) | JSON | Új alkalmazott | 3 |
| /employee | PUT | id, JSON | JSON | Alkalmazott módosítása | 3 |
| /employee | DELETE | id | 204 No Content | Alkalmazott törlése | 3 |

# A felhasználói dokumentáció

## A program általános specifikációja

Az elkészült program egy webes felületen elérhető ERP rendszer. A rendszer funkciói a felhasználó jogosultsági szintje alapján érhetők el, így biztosított a megfelelő adatvédelem és jogosultságkezelés.

A felhasználók az alábbi főbb modulokat érhetik el:

* Értékesítés kezelése – új rendelések rögzítése, meglévő rendelések nyomon követése, valamint számlák kiállítása. A számlázás során a felhasználó kiválaszthatja, hogy bejövő vagy kimenő számlát szeretne létrehozni, ezzel jelezve, hogy a termék beérkezik-e a céghez vagy értékesítés történik.
* Termékek nyilvántartása – a raktárkészlet naprakészen tartása, új termékek felvétele, meglévők módosítása vagy archiválása.
* Partnerek nyilvántartása – ügyfelek, beszállítók és egyéb üzleti partnerek adatainak kezelése, naprakészen tartása. Lehetőség van új partner felvételére, illetve a meglévők szerkesztésére.
* Alkalmazottak kezelése – dolgozók adatainak nyilvántartása, beosztások kezelése, felhasználói fiókok létrehozása és szerepkörök beállítása, ezzel biztosítva a megfelelő hozzáférési szinteket.
* Statisztikák – bevételek és kiadások grafikus és számbeli megjelenítése
* Pénzügyi modul – a cég bevételeinek és kiadásainak nyilvántartása, a kassza aktuális egyenlegének megjelenítése.

Az alkalmazás modern, letisztult felhasználói felülettel rendelkezik, amelyet úgy terveztünk, hogy még kevés informatikai tapasztalattal rendelkező felhasználók számára is könnyen kezelhető legyen. A menürendszert logikusan építettük fel, hogy könnyen érthető legyen, valamint megfelelő ikonokkal is elláttuk azokat, ezzel segítve a menük átláthatóságát és a könnyebb navigálást az oldalon.

A rendszer célja, hogy egy korszerű, felhasználóbarát és megbízható ERP megoldást nyújtson akár kis- és középvállalkozások számára pl. kis kereskedelmi cégek. A rendszer hosszú távon is jól alkalmazkodik, és lehetőséget biztosít további modulok integrálására is.

## Rendszerkövetelmények

Szerveroldali követelmények:

A webes ERP rendszer működéséhez az alábbi szerverkörnyezet szükséges:

* Webszerver: Apache
* Backend: PHP 7.4 vagy újabb verzió
* Adatbázis-kezelő: MySQL 5.7 / MariaDB vagy újabb
* Operációs rendszer: Windows 10 vagy újabb

Felhasználói (kliensoldali) követelmények:

Az alkalmazás használatához a felhasználóknak az alábbi feltételeknek kell megfelelniük:

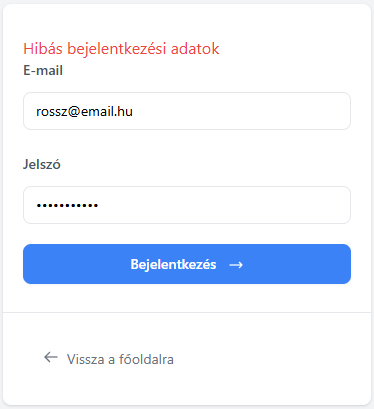
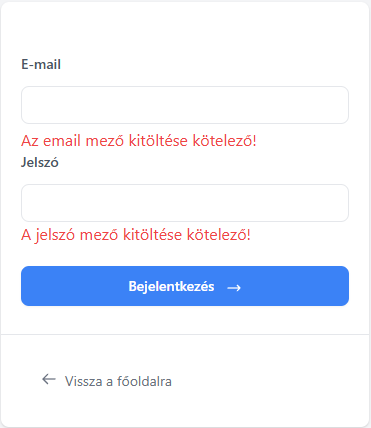
* Eszköz: Asztali számítógép, laptop, táblagép vagy okostelefon
* Internetkapcsolat: Stabil, böngészésre alkalmas kapcsolat
* Böngésző:
  + Google Chrome vagy bármilyen Chromium alapú böngésző
  + Mozilla Firefox
  + Microsoft Edge
* Képernyőfelbontás: Minimum 1280x720 (HD) javasolt asztali eszközökön, de az alkalmazás reszponzív kialakításának köszönhetően mobiltelefonon és táblagépen is kényelmesen használható.

## A program telepítése

A program böngészőben fut, így telepítést nem igényel.

## A program használatának részletes leírása

A dokumentációban bemutatott képernyőképek és funkciók az admin felhasználói jogosultságot tükrözik. Az admin felhasználó rendelkezik az összes modul elérésével, új felhasználók létrehozásával, törléssel és módosítással.

A program induló felülete egy „landing page”, amely rövid, átfogó ismertetést nyújt az alkalmazás funkcióiról és felhasználói felületéről. Innen a felhasználó a „Belépés” gombra kattintva jut el a bejelentkezési felületre.

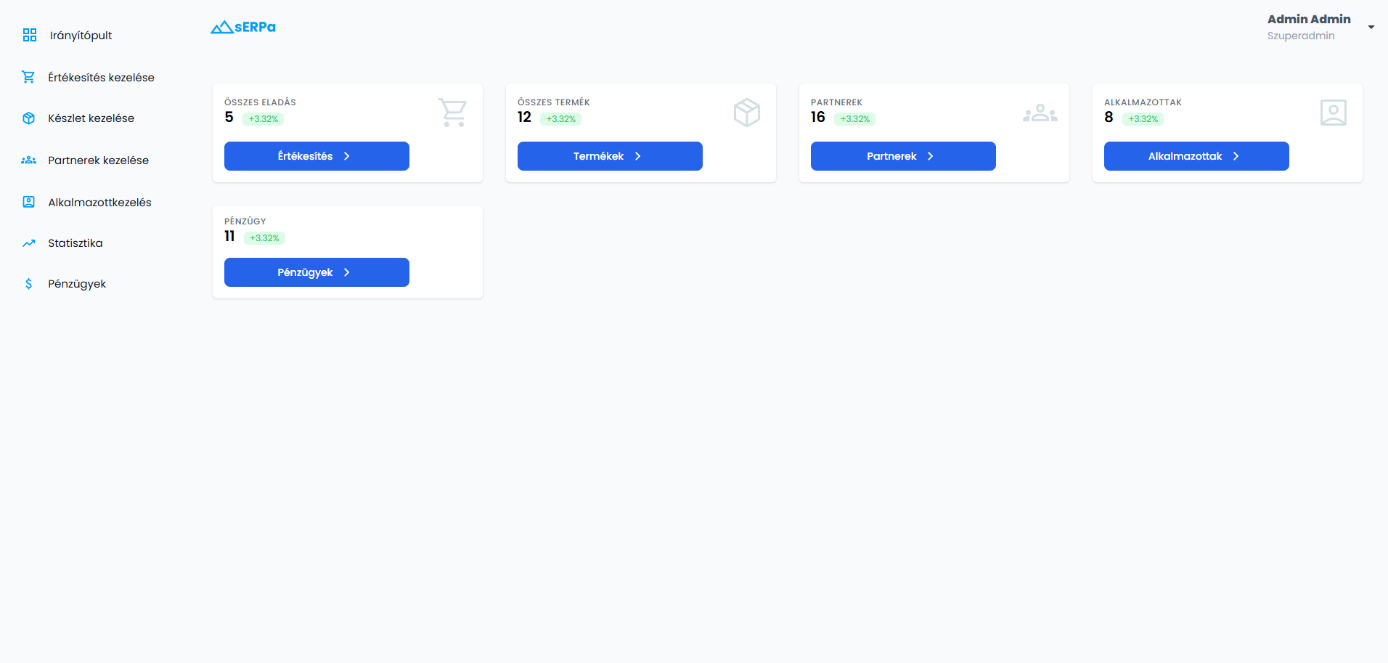
ábra 10

ábra 9

ábra 8

A rendszer kizárólag előre létrehozott felhasználói fiókokkal használható, azaz nincs lehetőség önálló regisztrációra. Új felhasználókat kizárólag az adminisztrátor jogosultságú felhasználók tudnak hozzáadni a rendszerhez, az alkalmazáson belül, a megfelelő adminisztrációs felületen keresztül. A bejelentkezési felületen a felhasználónak meg kell adnia az adminisztrátor által létrehozott fiók e-mail címét és jelszavát. A „Bejelentkezés” gombra kattintva a rendszer ellenőrzi a megadott adatok helyességét. Amennyiben az adatok hibásak, vagy hiányosak, a rendszer figyelmeztető üzenetet jelenít meg, amely egyértelmű visszajelzést ad a felhasználónak a sikertelen bejelentkezési kísérletről, melyek az alábbi képeken láthatóak.

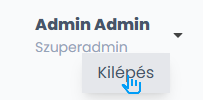
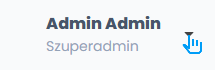
Helyes adatok megadása esetén a következő felület (Irányítópult) fogad minket:

Innen a bal oldali menüsáv, valamint az irányítópult oldalon megtalálható gombok segítségével navigálhatunk át a további oldalakra.

8.ábra

ábra 11

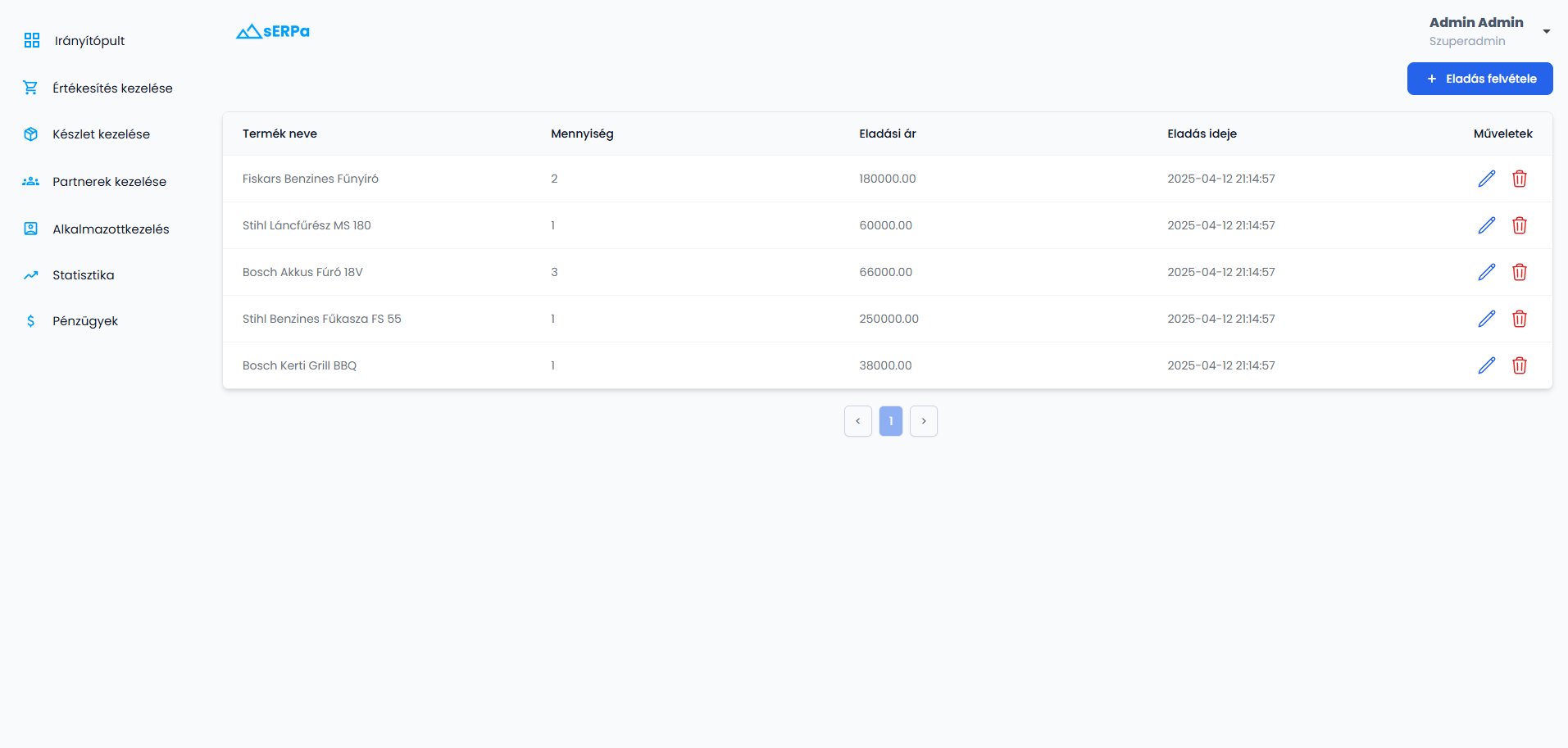
Ahogy bejelentkezésre, úgy kijelentkezésre is lehetőségünk van, melyet a jobb felső sarokban, a nyílra kattintva egy menüből érünk el.



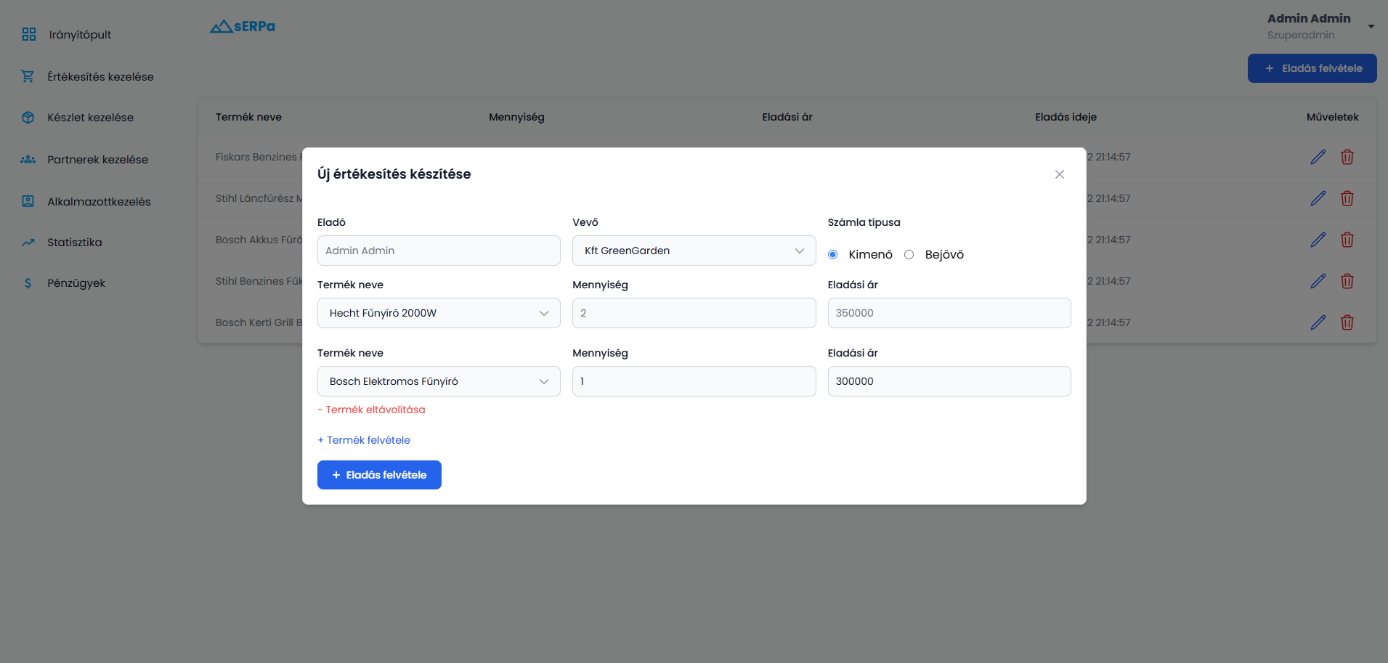
ábra 12

ábra 13

Ugyanitt láthatjuk az éppen bejelentkezett felhasználó nevét és jogosultságát is.

Az értékesítés tartalmazza az összes eladást, vásárlást, valamint az ezekhez tartozó számlát. Az „Eladás felvétele” gombra kattinva jegyezhetünk fel újabb rendelést a rendszerbe.

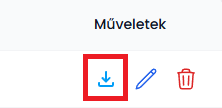
ábra 14

Ekkor egy felugró ablak fogad minket, ahol részletesen kiválaszthatjuk a vásárlás adatait, többek között a vevőt, a számla típusát, milyen terméket szeretnénk értékesíteni, a termékből mekkora mennyiséget és, hogy az adott terméket milyen áron áruljuk. Lehetőségünk van több, különféle terméket is hozzáadni, valamint törölhetünk is terméket. Az eladó kiválasztására nincs szükség, mivel a rendszer automatikusan az éppen bejelentkezett felhasználót veszi alapul eladóként.

ábra 15

Mezők típusa és hosszai a következőféleképpen néznek ki:

* Eladó neve: Az éppen bejelentkezett felhasználó adataiból nyeri ki a program
* Vevő: legördülő menöből választható, az adatbázisban található vevők közül
* Számla típusa: Kimenő vagy Bejövő érték válaszható
* Termék neve: adatbázisban található termékek közül válaszható egy legördülő menüből
* Mennyiség: number típusú
* Eladási ár: az adatbázisból kinyert termék árát automatikusan hozzárendeli a kiválasztott termékhez

Ha elvégeztük az adatok bevitelét, az „Eladás felvétele” gombra kattintva a program elküldi az adatokat a backendnek, ahol az tárolásra kerül az adatbázisba és a mentett adatok alapján egy PDF számlát generál. Amely az ábrán pirossal bekeretezett ikon megnyomásával akár is le is tölthető.

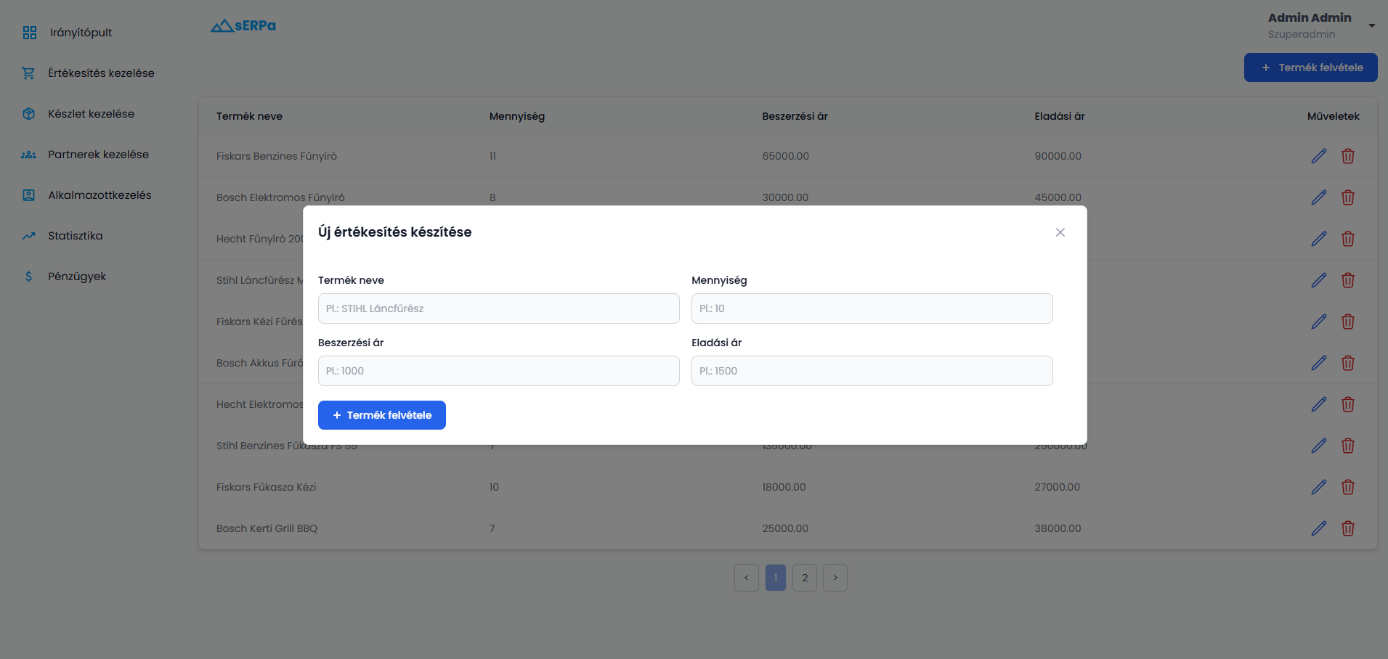
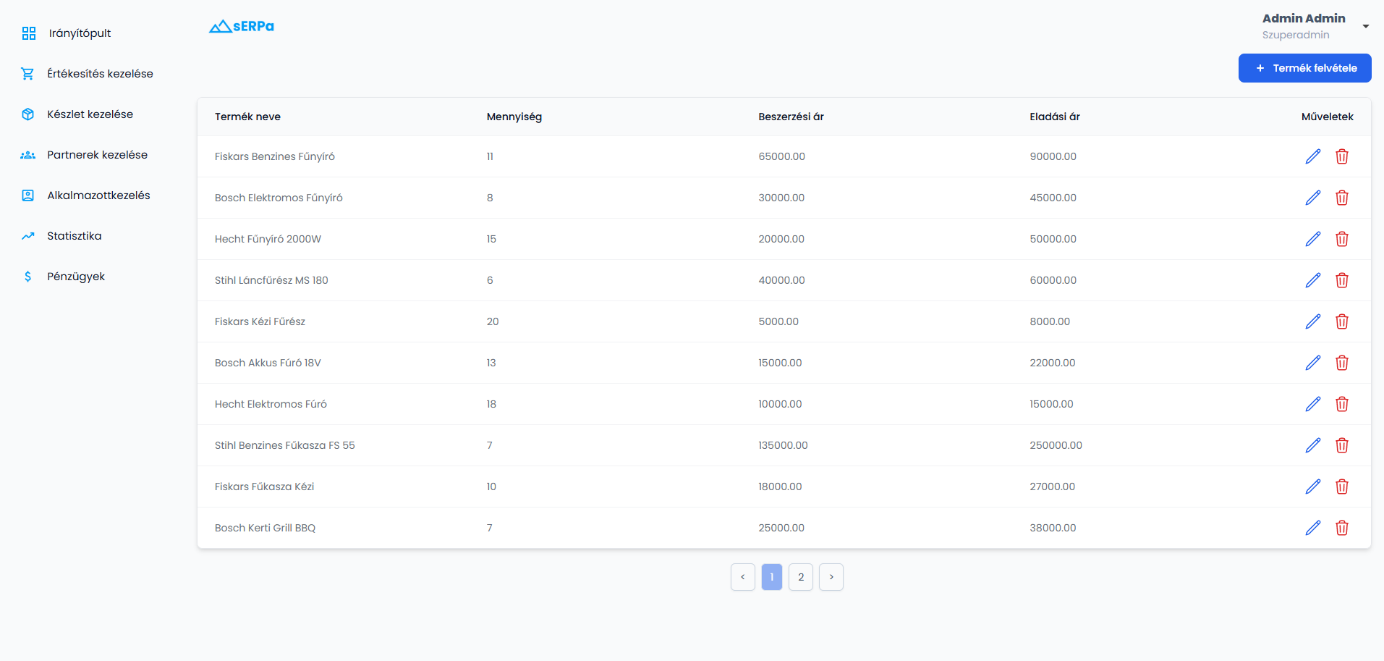
ábra 16

A műveleteknél található ikonoknak is megvan a saját szerepük. A ceruza ikon az adatok szerkesztésére szolgál ha esetleg elgépeltünk valamit vagy helytelen adat került rögzítésre. A szemetes ikonnal pedig a nem kívánt sorokat törölhetjük az oldalról. Fontos megjegyzés, hogy a számlák esetén csak a weboldalon kerül sor törlésre, az adatbázisban a számla mentésre kerül.

A táblázatokban oldalanként 10 sor fér el, így a táblázat alatt elhelyezkedik egy lapszámozás amivel könnyedén lapozhatunk a táblázatok oldalai között. A lapozást megtehetjük az oldalszámra, illetve a balra, jobbra nyilakra kattintva.

ábra 17

Tovább haladva a „Készlet kezelése” menüpontra, a felépítés ugyanaz, csak az adatok másak.

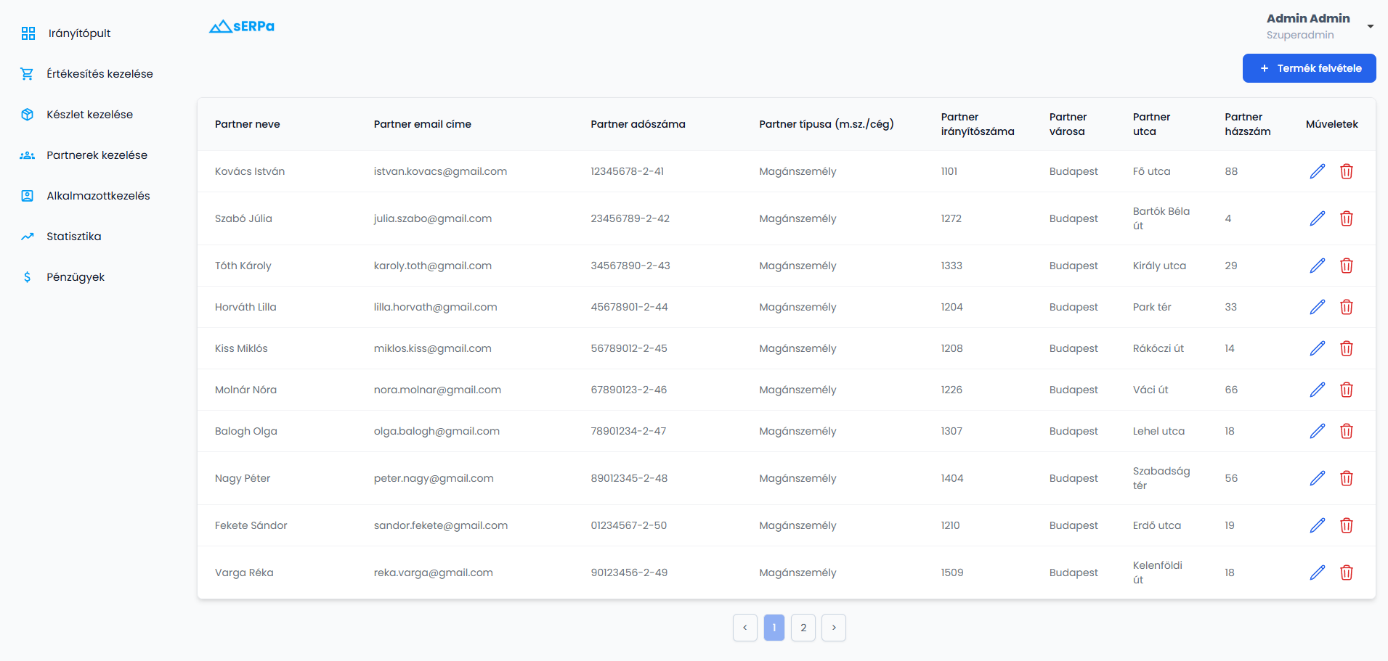
Szintén táblázatos megoldást használ , viszont ezen az oldalon a termék neve, mennyisége, beszerezési ára és eladási ára kerül rögzítésre.

ábra 19

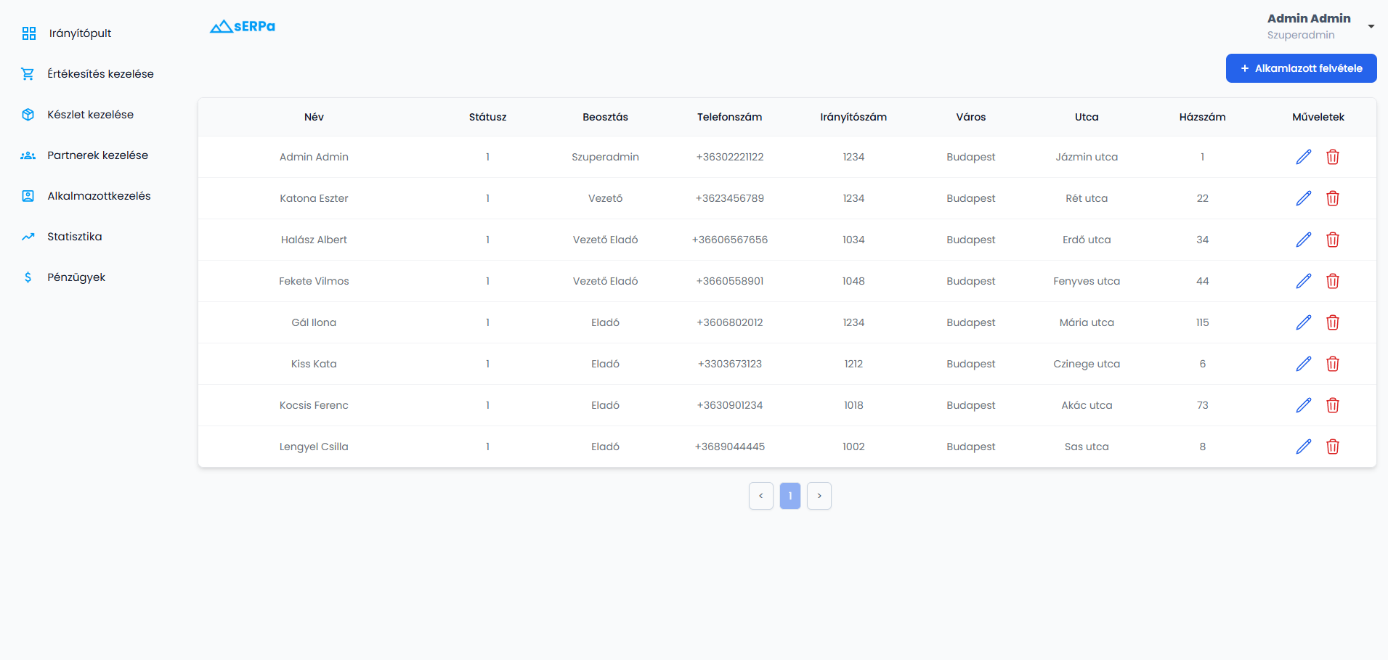
ábra 18

A „Termék felvétele” gombra kattinva, szintén egy modal fogad minket, ahova a megfelelő adatok beírásával elküldhetjük az űrlapot és az mentésre kerül az adatbázisban, azonban itt már nem készül külön PDF fájl.

A következő menüpont a „Partnerek kezelése” ahol a cég partnereinek, avagy a vásárlók és beszállítók adatait adhatjuk meg és tárolhatjuk le. Ahogy a képen is látszik, itt szükségünk van a partner nevére, e-mail címére, adószámára, a partner típusára (amely lehet magánszemély vagy cég), irányítószámára, lakhelyére és címére.



ábra 20

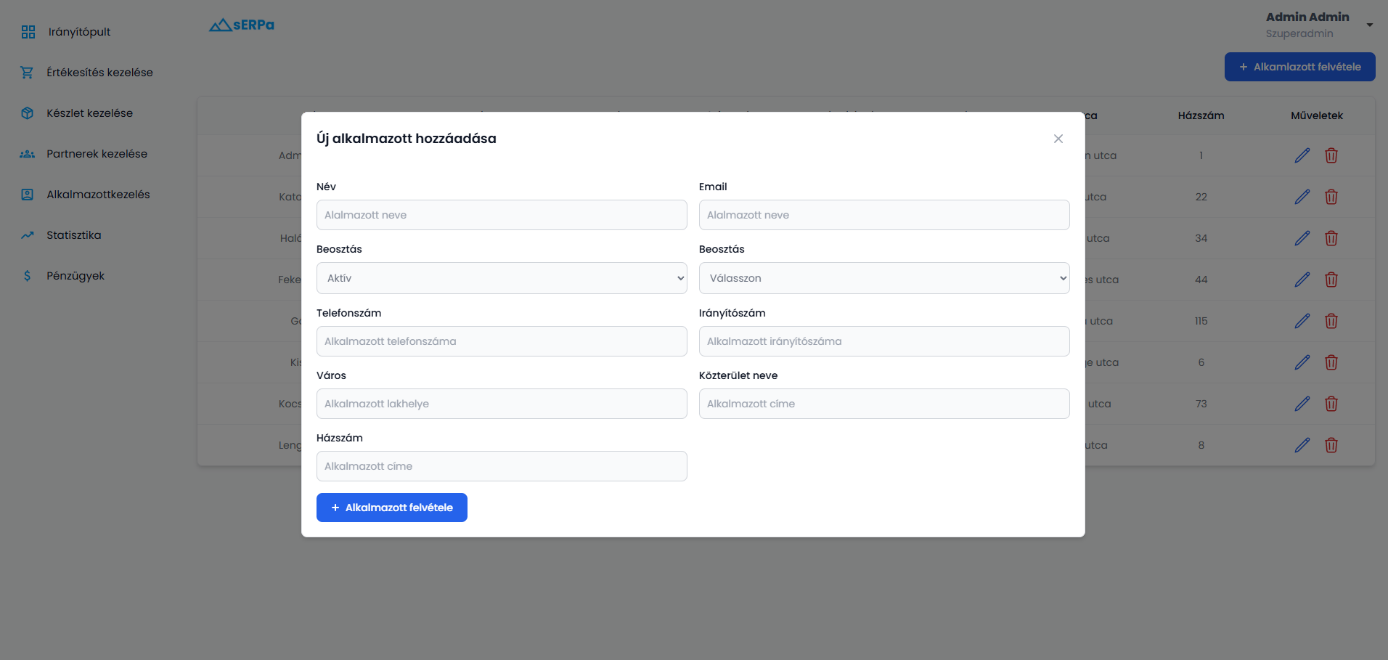
A soron következő menüpont az „Alkalmazottak kezelése”, ahol az alkalmazottak adatait vihetjük fel, módosíthatjuk vagy törölhetjük. Az adatok amelyekre szükségünk van a sikeres alkalmazott felvételhez, az az alkalmazott neve , státusza, beosztása, telefonszáma, irányítószáma, városa, utca és házszám.

ábra 21

Mezők típusa és hosszai a következőféleképpen néznek ki:

* Alkalmazott neve: text típusú, hosszérték nincs meghatározva
* Státusz: legördülő menöből 2 opció közül (Aktív/Inaktív) válaszható, adatbázisban 0 vagy 1
* Beosztás: legördülő menüből válaszható
* Telefonszám: number típusú, maximálisan 15 szám megengedett
* Irányítószám: number típusú, maximálisan 6 szám megengedett
* Utca: text típusú, nincs hosszérték meghatározva
* Házszám: number típusú, maximálisan 3 szám megengedett

A hosszértékek meghatározásánál figyelembe vettük azt is, ha esetleg valaki külföldről kívánja használni az alkalmazást, ahol esetleg eltérő hosszúságú adatot kíván meg egy-egy mező. Az alkalmazottak felvétele vagy hozzáadása szintén egy modal segítségével történik, ahol a fentebb írt adatok megadása szükséges.

A felhasználói felületek jogosultságtól független szinte majdnem megegyeznek, annyi különbséggel, hogy minél nagyobb szintű egy felhasználó, annál több menü érhető el a számára és annál több adat bevitelére és módosítására valamit törlésére van jogosultsága, a jogosultságok menüelosztása az alábbi ábrán látható:

ábra 22



ábra 23

# Továbbfejlesztési lehetőségek

A program fejlesztési folyamata során számos ötlet merült fel, melyeket a csapat gondos átbeszélés után közös megegyezéssel – legalábbis az aktuális fejlesztési szakaszban – elvetettünk. Ezek az ötletek ugyanakkor továbbra is részét képezik a hosszú távú terveinknek, és a jövőbeni verziókban való megvalósításuk mindig számításba kerül.

Célunk a felhasználói élmény folyamatos javítása, ezért egyik fontos tervezett fejlesztés a sötét mód (dark mode) bevezetése. Manapság a szoftverek elvárható alapfunkciói közé tartozik a sötét téma lehetősége, amely nemcsak esztétikai szempontból, hanem a kényelmes használat és a szemkímélő megoldások miatt is nélkülözhetetlen. A sötét mód implementálásával szeretnénk megfelelni a felhasználók igényeinek, és modern, felhasználóbarát környezetet biztosítani.

Ugyancsak fontosnak tartjuk a program többnyelvű támogatását, hogy nemzetközi szinten is elérhetővé és versenyképessé váljon. Elsősorban az angol és német nyelvi verziók kidolgozása áll a terveink középpontjában, de a jövőben további nyelvek hozzáadásával szeretnénk bővíteni a lehetőségeket.

Ezen fejlesztések mellett folyamatosan figyelemmel kísérjük a technológiai trendeket és a felhasználói visszajelzéseket, hogy a program mindig a legmodernebb és legfelhasználóbarátabb megoldásokat kínálja.

# Összegzés

Összegzésképpen, ez a projekt kiváló lehetőséget biztosított számunkra, hogy kipróbáljuk magunkat egy csapatban végzett fejlesztési munka során. Megtapasztalhattuk, milyen kihívásokkal jár a közös munka, és hogyan tudjuk együtt megoldani a felmerülő problémákat, illetve hatékonyan megvalósítani a kitűzött célokat.

A projekt során szerzett tapasztalatok nemcsak elméleti tudásunkat mélyítették el, hanem gyakorlati készségeinket is jelentősen fejlesztették, ami véleményünk szerint az egyik leghatékonyabb módja a programozás elsajátításának.

Emellett lehetőségünk nyílt több olyan fejlesztőeszköz és programozási nyelv megismerésére is, amelyekkel az iskolai keretek között korábban még nem dolgoztunk. Ez a tapasztalat nagyban hozzájárult szakmai fejlődésünkhöz és jövőbeli munkavállalási lehetőségeinkhez.

## Forrásmegjelölés

<https://app.diagrams.net/> adatbázis ERD és táblák megrajzolása

<https://www.creative-tim.com/twcomponents> táblázatok

<https://fonts.google.com/icons> svg ikonok

<https://tailwindcss.com/> TailwindCSS

<https://carbon.now.sh/> Kód részletek