

VIZSGAREMEK

Baczur Zsolt, Békés Zoltán és Magyar Tamás

Pécs, 2025

BARANYA MEGYEI SZAKKÉPZÉSI CENTRUM

BARANYA MEGYEI SZC RADNÓTI MIKLÓS KÖZGAZDASÁGI TECHNIKUM

sERP a vállalatirányítási rendszer

Készítették: Baczur Zsolt, Békés Zoltán és Magyar Tamás

PÉCS

2025

Tartalomjegyzék

I. BEVEZETŐ, A FELADAT RÖVID ISMERTETÉSE	4
II. A FEJLESZTŐI DOKUMENTÁCIÓ	5
1. FEJLESZTŐKÖRNYEZET ISMERTETÉSE	5
2. ADATSZERKEZET ISMERTETÉSE	5
3. AZ ADATBÁZIS TERVEZÉSE	6
4. NORMALIZÁLT (LOGIKAI) ERD LÉTREHOZÁSA	8
5. ALGORITMUSOK BEMUTATÁSA	8
6. TESZTELÉSI DOKUMENTÁCIÓ	13
III. API VÉGPONTOK.....	16
IV. A FELHASZNÁLÓI DOKUMENTÁCIÓ	17
1. A PROGRAM ÁLTALÁNOS SPECIFIKÁCIÓJA	17
2. RENDSZERKÖVETELMÉNYEK	18
3. A PROGRAM TELEPÍTÉSE	18
4. A PROGRAM HASZNÁLATÁNAK RÉSZLETES LEÍRÁSA.....	18
V. TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK	27
VI. ÖSSZEGZÉS	28
1. FORRÁSMEGJELÖLÉS	28

I. 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.

II. A fejlesztői dokumentáció

1. 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

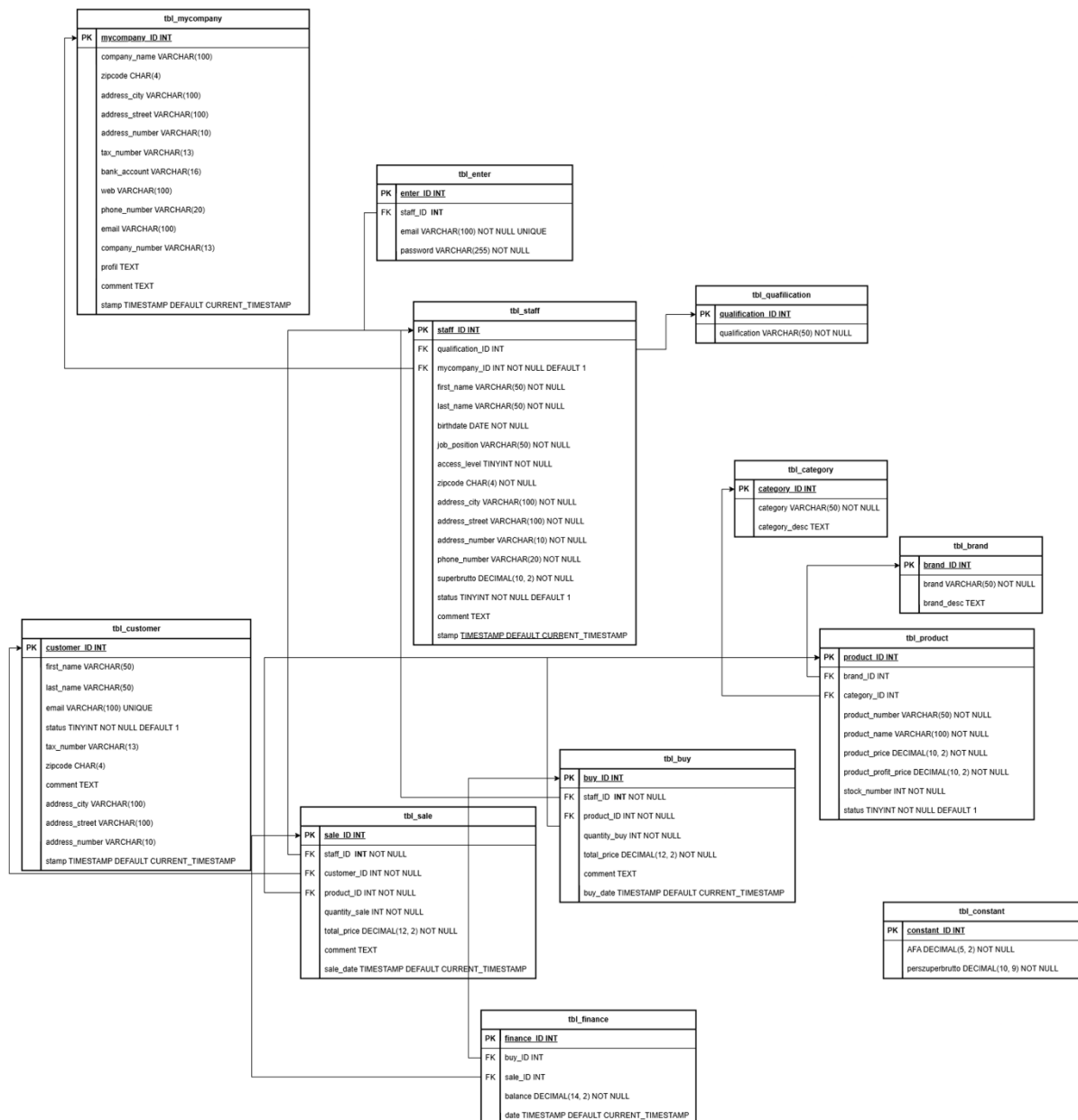
A fent említett eszközök kiválasztását a következő szempontok indokolták: ingyenes elérhetőség, platformfüggetlenség, tanulhatóság, valamint a korábbi tanulmányaink során megszerzett tapasztalat.

2. 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.

3. 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 pénzájának (forgó-tőkéjének) a nyomonkövetését a `tbl_finance` tábla tárolja, rekordok formájában, minden egyes a `tbl_sale` táblában rögzített eladás egy triggerrel 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 visszanezhető 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 konstans. 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.

4. 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](#)

5. Algoritmusok bemutatása

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

Bejelentkezési rendszer algoritmus

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
- Többosztályú 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 változókbán.
 - 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:

```
public static function login(){
    if(isset($_POST['email']) && isset($_POST['password'])){
        $where='email="'.$_POST['email'].'" AND password=SHA2("'.$_POST['password'].'", 256)';
        $user=Db::Select("tbl_enter", "*", $where);
        if(!$user==null){
            session_start();
            $_SESSION['staff_ID']=$user[0]['staff_ID'];

            $level=Db::Select("tbl_staff", "access_level", "staff_ID=".$_user[0]['staff_ID']);

            $_SESSION['access_level']=$level[0]['access_level'];
            $_SESSION['session_id']=session_id();
            $_SESSION['time']=time();
            http_response_code(202);
            echo json_encode(['response' => 'success', 'message' => 'Accepted']);
        }
        else{
            http_response_code(401);
            echo json_encode(['response' => 'error', 'message' => 'Unauthorized access']);
        }
    }
    else{
        http_response_code(400);
        echo json_encode(['response' => 'error', 'message' => 'Bad Request']);
    }
}
```

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:

```
if (!email) {  
    emailErrorElement.textContent = 'Az email mező kitöltése kötelező!';  
    hasError = true;  
}  
  
if (!password) {  
    passwordErrorElement.textContent = 'A jelszó mező kitöltése kötelező!';  
    hasError = true;  
}  
  
if (hasError) return; // A beküldés leáll, ha bármelyik mező üres
```

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).

```
const response = await fetch(`${API_URL}login`, {  
    method: 'POST',  
    headers: {  
        'Content-Type': 'application/x-www-form-urlencoded',  
    },  
    body: new URLSearchParams({ email, password })  
});
```

ábra 3

Szerveroldali validáció

Az ActionController::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 nagyobbban.
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 responszívabb 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:

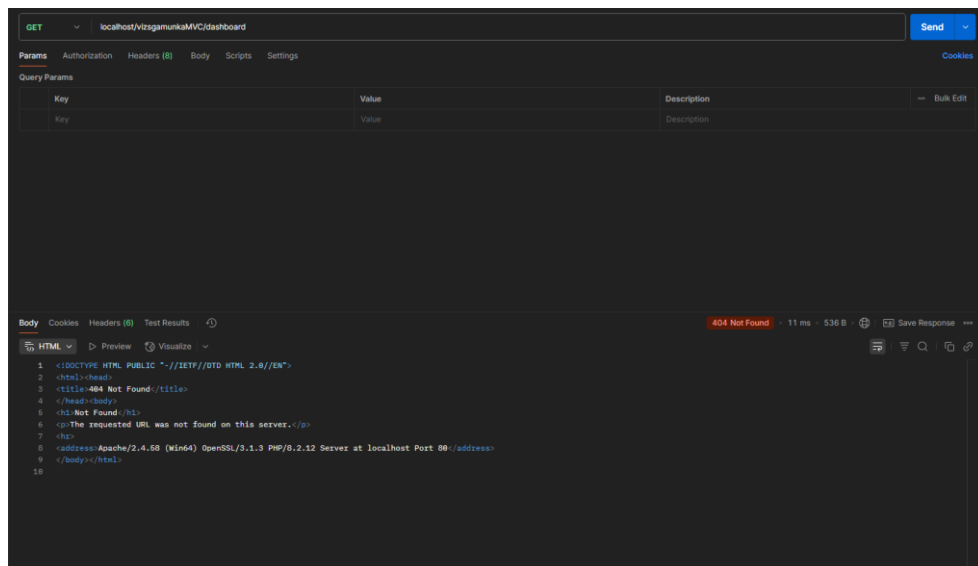
- A JavaScript lekéri az összes mező értékét a DOM-ból.
- POST kérés a szerver feléAz összegyűjtött adatok JSON formátumban kerülnek elküldésre a backend felé egy fetch() POST kérés segítségével. A válasz egy JSON objektum, amely visszaigazolja a sikeres mentést.
- Frontend frissítésA válaszként kapott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

6. Tesztelési dokumentáció

A tesztelést POSTMAN segítségével végeztük

Több tesztet is megvizsgáltunk, hogy miként reagál a program.

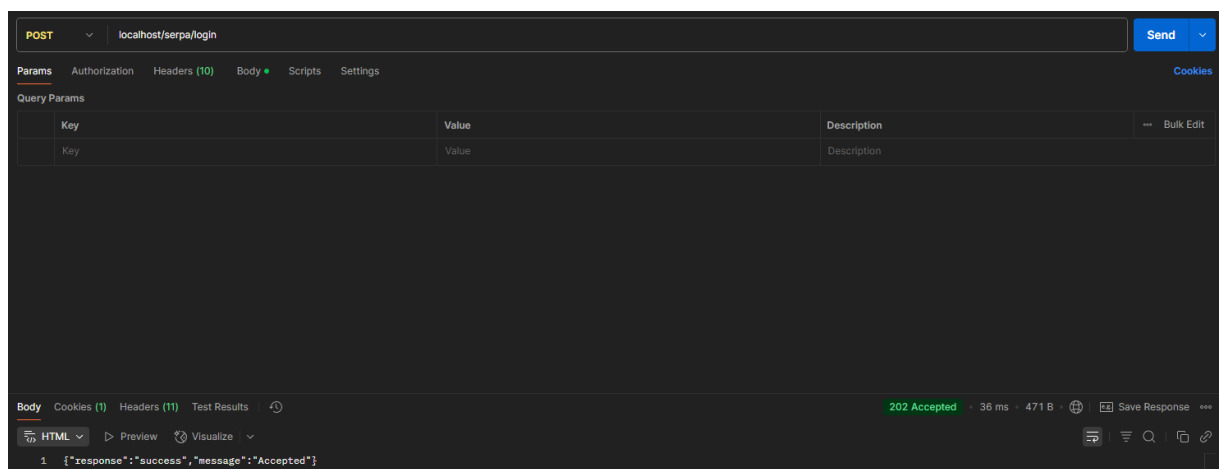
Első tesztetünknel azt vizsgáltuk, hogy mi történik akkor, ha valaki csak simán az URL beírásával próbálja meg elérni mondjuk az irányítópultot:



ábra 5

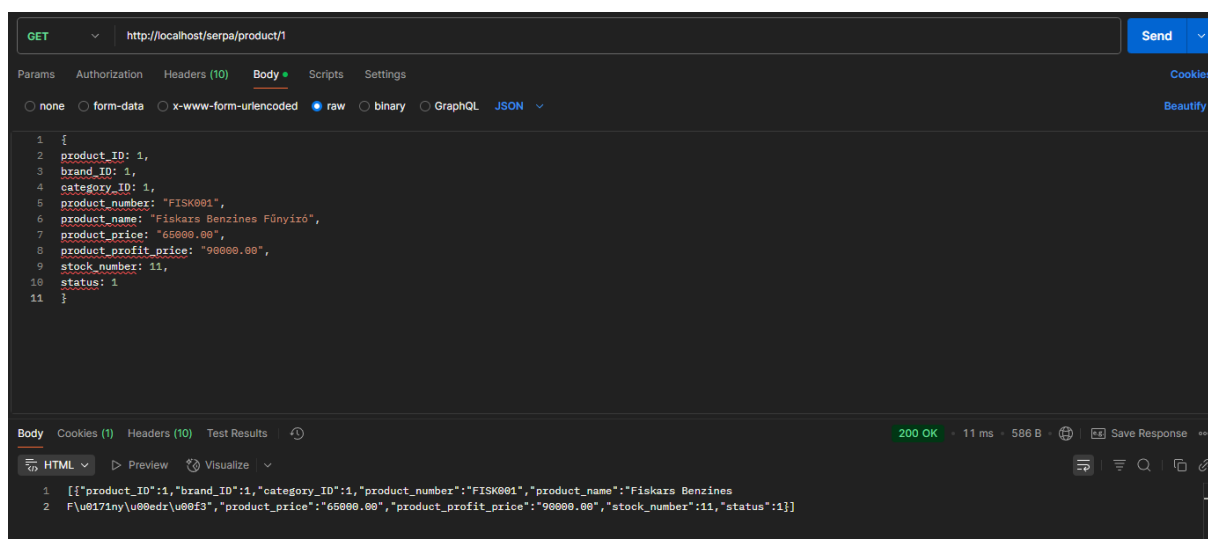
Ez a teszt egy 404 Not found hibakóddal tér vissza, ami azt jelenti, hogy bejelentkezés nélkül nem elérhető az irányítópult, így meggátolva az illetéktelen behatolást a programba.

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 látható módon a válasz egy `200 OK` válasszal, valamint az adott termék (jelen esetben 1-es ID-val rendelkező) adataival tér vissza.

III. 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

IV. A felhasználói dokumentáció

1. 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 kassa 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.

2. 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.

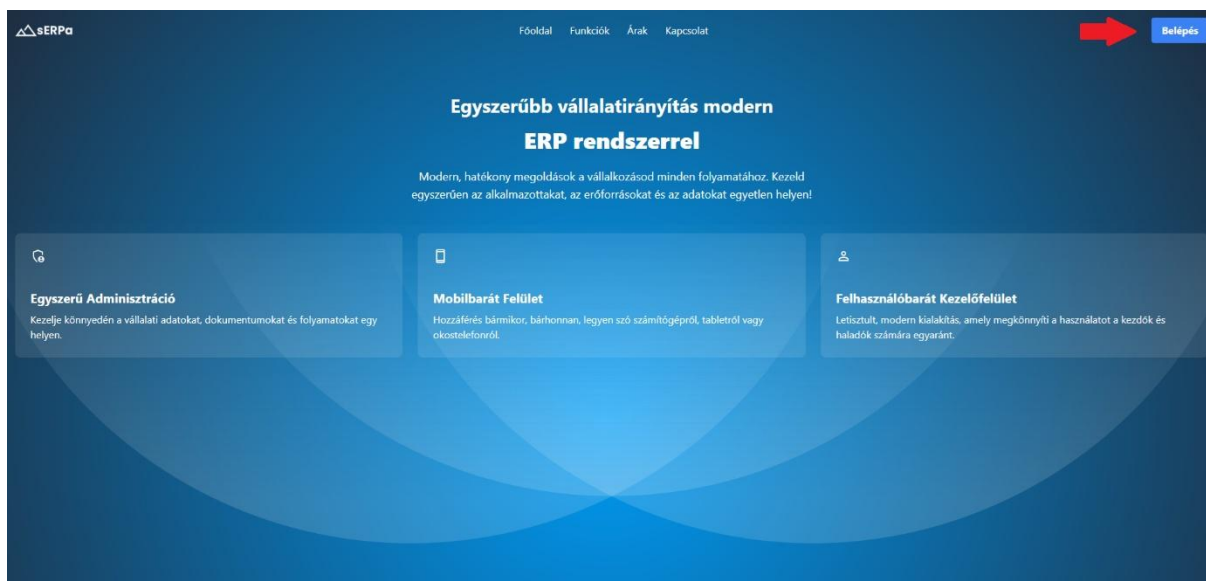
3. A program telepítése

A program böngészőben fut, így telepítést nem igényel.

4. 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 8

ábra 9

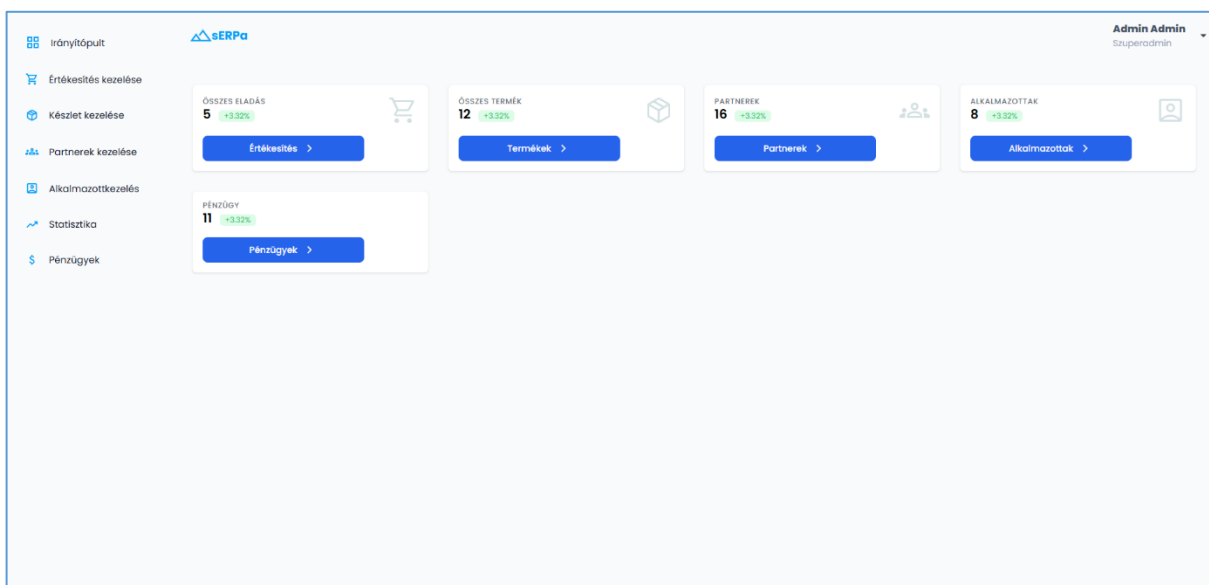
ábra 10

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.

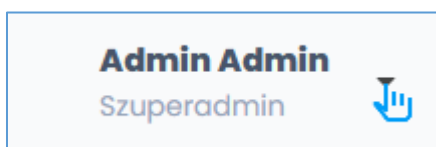
Helyes adatok megadása esetén a következő felület (Irányítópult) fogad minket:

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

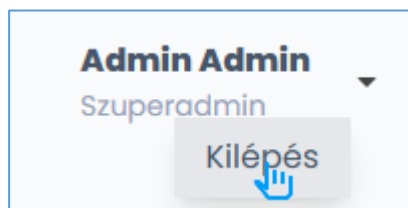


ábra 11

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



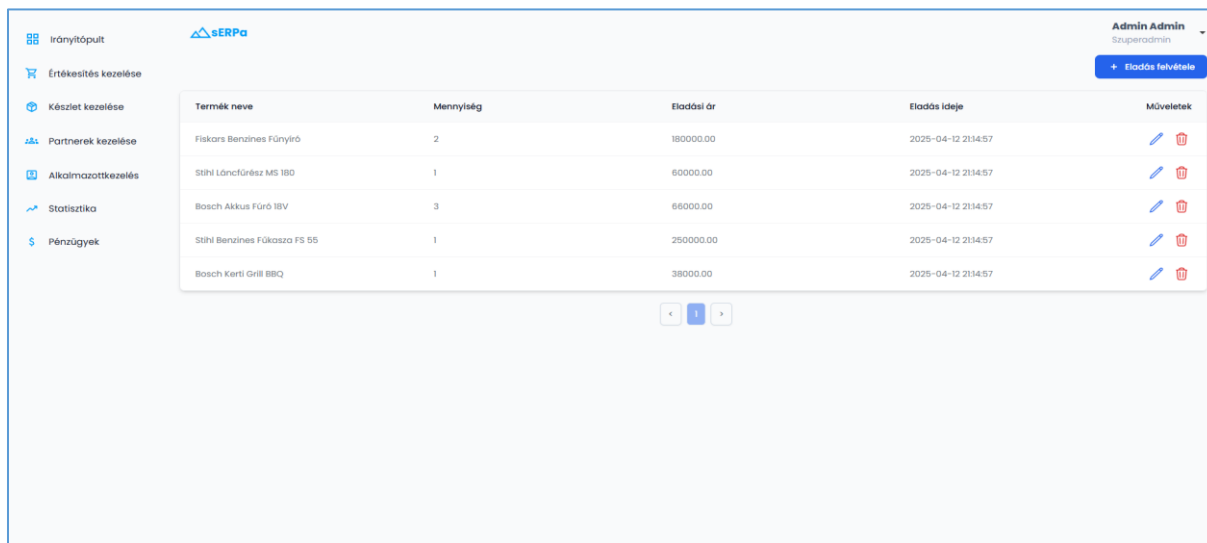
ábra 12



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.

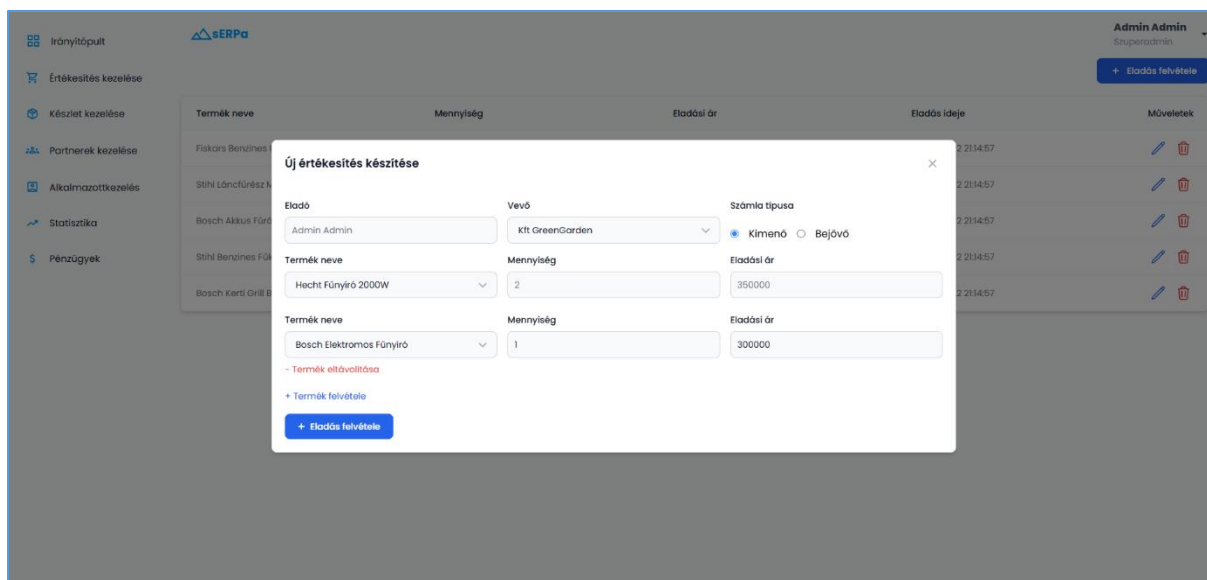


The screenshot shows the 'Eladás felvétele' (Sales Entry) page in the eERP system. It features a sidebar with navigation options: Irányítópult, Értékesítés kezelése, Készlet kezelése, Partnerek kezelése, Alkalmazottkezelés, Statistika, and Pénzügyek. The main area displays a table of sales transactions with columns for Termék neve (Product name), Mennyiség (Quantity), Eladási ár (Sales price), Eladás ideje (Sales date), and Művelet (Action). The table lists five transactions, including Fiskars Benzines Fűnyíró, Stihl Lánzfűrész MS 180, Bosch Akkus Fűró 18V, Stihl Benzines Fűkaszó FS 55, and Bosch Kerti Grill BBQ. Each transaction has a corresponding 'Művelet' column with edit and delete icons.

Termék neve	Mennyiség	Eladási ár	Eladás ideje	Művelet
Fiskars Benzines Fűnyíró	2	180000.00	2025-04-12 21:14:57	Edit Delete
Stihl Lánzfűrész MS 180	1	60000.00	2025-04-12 21:14:57	Edit Delete
Bosch Akkus Fűró 18V	3	66000.00	2025-04-12 21:14:57	Edit Delete
Stihl Benzines Fűkaszó FS 55	1	250000.00	2025-04-12 21:14:57	Edit Delete
Bosch Kerti Grill BBQ	1	38000.00	2025-04-12 21:14:57	Edit Delete

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.



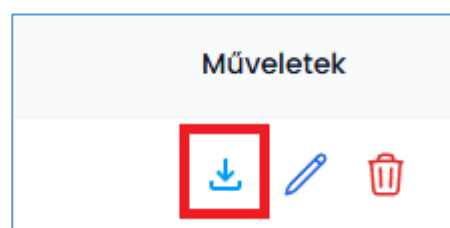
The screenshot shows the 'Új értékesítés készítése' (Create new sale) dialog box in the eERP system. The dialog box contains fields for Eladó (Seller), Vevő (Buyer), Számla típusa (Invoice type), Termék neve (Product name), Mennyiség (Quantity), and Eladási ár (Sales price). The 'Eladó' field is pre-filled with 'Admin Admin'. The 'Vevő' field is set to 'Kft GreenGarden'. The 'Számla típusa' field has two options: 'Kimenő' (selected) and 'Bejövő'. The 'Termék neve' field is set to 'Hocht Fűnyíró 2000W' with a quantity of '2' and a sales price of '350000'. Below this, there is a section for adding more products, with 'Bosch Elektromos Fűnyíró' selected, a quantity of '1', and a sales price of '300000'. The dialog box also includes a 'Termék eltávolítása' (Remove product) link and a 'Termék felvétele' (Add product) link. The background shows the same sidebar and table as in the previous screenshot.

ábra 15

Mezők típusa és hosszai a következőféleképpen néznek ki:

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

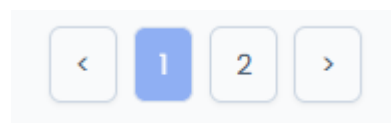
Ha elvégeztük az adatok bevitelét, az „Eladás felvétele” gombra kattintva a program elküldi az adatokat a backendnek, ahol az tárolásra kerül az adatbázisba és a mentett adatok alapján egy PDF számlát generál. Amely az ábrán pirossal bekeretezett ikon megnyomásával akár is le is tölthető.



ábra 16

A műveleteknél található ikonoknak is megvan a saját szerepük. A ceruza ikon az adatok szerkesztésére szolgál ha esetleg elgépeztünk 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 lapozósorozat amivel könnyedén lapozhatunk a táblázatok oldalai között. A lapozást megtehetjük az oldalszámmra, 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, beszerzési ára és eladási ára kerül rögzítésre.

Termék neve	Mennyiség	Beszerzési ár	Eladási ár	Műveletek
Fiskars Benzinés Fűnyíró	11	65000.00	90000.00	
Bosch Elektromos Fűnyíró	8	30000.00	45000.00	
Hecht Fűnyíró 2000W	15	20000.00	50000.00	
Stihl Lánzfűrész MS 180	6	40000.00	60000.00	
Fiskars Kézi Fűrész	20	5000.00	8000.00	
Bosch Akkus Fűró 18V	13	15000.00	22000.00	
Hecht Elektromos Fűró	18	10000.00	15000.00	
Stihl Benzinés Fűkasza FS 55	7	135000.00	250000.00	
Fiskars Fűkasza Kézi	10	18000.00	27000.00	
Bosch Kerti Grill BBQ	7	25000.00	38000.00	

ábra 18

Új értékesítés készítése

Termék neve:

Mennyiség:

Beszerzési ár:

Eladási ár:

+ Termék felvétele

ábra 19

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.

Partner neve	Partner email címe	Partner adószáma	Partner típusa (m.sz./cég)	Partner irányítószáma	Partner városa	Partner utca	Partner házszám	Műveletek
Kovács István	istvan.kovacs@gmail.com	12345678-2-41	Magánszemély	1101	Budapest	Fő utca	88	✎ ✖
Szabó Júlia	julia.szabo@gmail.com	23456789-2-42	Magánszemély	1272	Budapest	Bartók Béla út	4	✎ ✖
Tóth Károly	karoly.toth@gmail.com	34567890-2-43	Magánszemély	1333	Budapest	Király utca	29	✎ ✖
Horváth Lilla	lilla.horvath@gmail.com	45678901-2-44	Magánszemély	1204	Budapest	Park tér	33	✎ ✖
Kiss Miklós	miklos.kiss@gmail.com	56789012-2-45	Magánszemély	1208	Budapest	Rákóczi út	14	✎ ✖
Molnár Nóra	nora.molnar@gmail.com	67890123-2-46	Magánszemély	1226	Budapest	Váci út	66	✎ ✖
Balogh Olga	olga.balogh@gmail.com	78901234-2-47	Magánszemély	1307	Budapest	Lehel utca	18	✎ ✖
Nagy Péter	peter.nagy@gmail.com	89012345-2-48	Magánszemély	1404	Budapest	Szabadság tér	56	✎ ✖
Fekete Sándor	sandor.fekete@gmail.com	01234567-2-50	Magánszemély	1210	Budapest	Erdő utca	19	✎ ✖
Varga Réka	reka.varga@gmail.com	90123456-2-49	Magánszemély	1509	Budapest	Kelenföldi út	10	✎ ✖

ábra 20

A soron következő menüpont az „Alkalmazottak kezelése”, ahol az alkalmazottak adatait vihetjük fel, módosíthatjuk vagy törölhetjük. Az adatok amelyekre szükségünk van a

Név	Státusz	Beosztás	Telefonszám	Irányítószám	Város	Utcá	Házszám	Műveletek
Admin Admin	1	Superadmin	+36302221122	1234	Budapest	Jászmin utca	1	✎ ✖
Katona Eszter	1	Vezető	+3623456789	1234	Budapest	Rét utca	22	✎ ✖
Halász Albert	1	Vezető Eladó	+36906567856	1034	Budapest	Erdő utca	34	✎ ✖
Fekete Vilmos	1	Vezető Eladó	+36606568901	1048	Budapest	Fenyves utca	44	✎ ✖
Gál Ilona	1	Eladó	+3606802012	1234	Budapest	Mária utca	115	✎ ✖
Kiss Kata	1	Eladó	+3303673123	1212	Budapest	Czínege utca	6	✎ ✖
Kocsis Ferenc	1	Eladó	+3630901234	1018	Budapest	Ákác utca	73	✎ ✖
Lengyel Csilla	1	Eladó	+3689044445	1002	Budapest	Sas utca	8	✎ ✖

ábra 21

sikeres alkalmazott felvételhez, az az alkalmazott neve, státusza, beosztása, telefonszáma, irányítószáma, városa, utca és házszám.

Mezők típusa és hosszai a következőféleképpen néznek ki:

- Alkalmazott neve: text típusú, hosszérték nincs meghatározva
- Státusz: legördülő menüből 2 opció közül (Aktív/Inaktív) választható, adatbázisban 0 vagy 1
- Beosztás: legördülő menüből választható
- 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.

The screenshot shows a web application interface with a sidebar on the left containing menu items like 'Irányítópult', 'Értékesítés kezelése', 'Készlet kezelése', 'Partnerek kezelése', 'Alkalmazottkezelés', 'Statistika', and 'Pénzügyek'. The main content area is partially visible in the background, showing a table with columns for 'Házszám' and 'Művelet'. Overlaid on this is a modal window titled 'Új alkalmazott hozzáadása'. The modal contains the following fields:

- Név:** A text input field labeled 'Alkalmazott neve'.
- Email:** A text input field labeled 'Alkalmazott neve'.
- Beosztás:** A dropdown menu with 'Aktív' selected and a 'Válasszon' button.
- Telefonszám:** A text input field labeled 'Alkalmazott telefonszáma'.
- Irányítószám:** A text input field labeled 'Alkalmazott irányítószáma'.
- Város:** A text input field labeled 'Alkalmazott lakhelye'.
- Község/település:** A text input field labeled 'Alkalmazott címe'.
- Házszám:** A text input field labeled 'Alkalmazott címe'.

At the bottom of the modal is a blue button labeled '+ Alkalmazott felvétele'.

ábra 22

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 valamint törlésére van jogosultsága, a jogosultságok menüelosztása az alábbi ábrán látható:

Admin	Vezető	Vezető eladó	Eladó
 Irányítópult	 Irányítópult	 Irányítópult	 Irányítópult
 Értékesítés kezelése	 Értékesítés kezelése	 Értékesítés kezelése	 Értékesítés kezelése
 Készlet kezelése	 Készlet kezelése	 Készlet kezelése	 Készlet kezelése
 Partnerek kezelése	 Partnerek kezelése	 Partnerek kezelése	
 Alkalmazottkezelés	 Alkalmazottkezelés		
 Statisztika			
 Pénzügyek			

ábra 23

V. 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 versenyképesé 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.

VI. Összegzés

Összegzőképpen, 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.

1. 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