**Projektmunka dokumentálása**

**Valai Pékség webalkalmazása**



Premontrei Szakgimnázium és Technikum

Keszthely

Szoftverfejlesztő és tesztelő szak

5-0613-12-03

Készítette:

Szentes Mirtill és Nagy Máté János

Tartalom

[Bevezetés 4](#_Toc195177809)

[A projekt célja 5](#_Toc195177810)

[Projekttervezet 7](#_Toc195177811)

[Felhasználói felület (UI) tervezés 7](#_Toc195177812)

[Regisztráció felület 9](#_Toc195177813)

[Bejelentkezés felület 9](#_Toc195177814)

[Termékek felület 10](#_Toc195177815)

[Rendelés felület 11](#_Toc195177816)

[Felhasznált programok 13](#_Toc195177817)

[Adatbázis 17](#_Toc195177818)

[Táblák 17](#_Toc195177819)

[ER Adatmodell 19](#_Toc195177820)

[Kapcsolatok és integritási szabályok 19](#_Toc195177821)

[Frontend 21](#_Toc195177822)

[Komponensek 21](#_Toc195177823)

[Reszponzivitás és dizájn 23](#_Toc195177824)

[Backend 24](#_Toc195177825)

[Tesztelés 26](#_Toc195177826)

[Funkcionális tesztelés 26](#_Toc195177827)

[Reszponzív tesztelés 26](#_Toc195177828)

[Terhelési tesztelés 26](#_Toc195177829)

[Biztonsági tesztelés 26](#_Toc195177830)

[Fejlesztések 27](#_Toc195177831)

[Reflektálás 28](#_Toc195177832)

[Források 30](#_Toc195177833)

**Projekttervezet**

# Bevezetés

**Előszó**

Projektünket Szentes Mirtill és Nagy Máté János közösen készítettük el, amely a Valai Pékség nevet viseli. A projekt témájának kiválasztásakor fontos szempont volt, hogy egy olyan területtel foglalkozzunk, amelyhez személyesen is kötődünk, és amelyben szívesen mélyedünk el. A pékség, mint téma közel áll hozzánk, mivel mindketten szeretjük a friss pékárukat, és érdekesnek találtuk a helyi vállalkozások digitalizációs lehetőségeit.

A név kiválasztásánál több lehetőséget is megfontoltunk, azonban a Valai Pékség név volt az, amely leginkább visszaadta a projekt által képviselt barátságos és autentikus hangulatot.

**Feladat megosztás**

A projekt során a feladatokat és a munkafolyamatokat egyenlően osztottuk el, hogy mindketten maximálisan hozzájárulhassunk a Valai Pékség webalkalmazásának megvalósításához. Az alábbiakban részletezem, hogy ki melyik feladatot végezte el a projekt során:

**Szentes Mirtill feladatai:**

* Design: Az alkalmazás felhasználói felületét és a felhasználói élményt, figyelve arra, hogy az könnyen navigálható és vizuálisan vonzó legyen.
* Weboldal responsive dizájn: A különböző eszközökön való optimális működés.
* Dokumentáció: Tartalmazta a fejlesztés technikai leírásait, a használt technológiai stack-et és az alkalmazás működésének részletes bemutatását.
* Frontend fejlesztés: A webalkalmazás megjelenéséért és felhasználói élményéért felelt. Termékek bemutatása, az online rendelési felület és a vásárlók számára fontos információkat tartalmazó oldalak.

**Nagy Máté János feladatai:**

* Backend fejlesztés: A webalkalmazás szerveroldali logikájának megvalósítása. A rendelési folyamat, a vásárlói adatok és a rendelésekkel kapcsolatos egyéb funkciók kezelése.
* Adatbázis tervezés és fejlesztés: Az alkalmazás adatbázisának struktúrája. Az adatok hatékony és biztonságos tárolása.
* API fejlesztés: Lehetővé tették a frontend és a backend közötti kommunikációt, biztosítva a zökkenőmentes adatcserét a webalkalmazás különböző részei között.

**Közös feladatok:**

* Projekt menedzsment és ütemezés: A projekt ütemezését és menedzselését közösen végeztük. Mindkettőnk feladata volt a határidők betartása, az egyes feladatok előrehaladás nyomon követése.
* Tesztelés: A projekt végső fázisában közösen teszteltük az alkalmazást, hogy minden funkció hibátlanul működjön. A tesztelés során külön figyelmet fordítottunk a rendelési folyamatra és a felhasználói élményre.

Ezzel a feladatmegosztással biztosítottuk, hogy a projekt minden részlete precízen kidolgozott legyen, és mindenki maximálisan hozzájárulhasson a sikeres megvalósításhoz.

## A projekt célja

A projekt célja, hogy a Valai Pékség számára egy modern és felhasználóbarát webalkalmazást hozzunk létre, amely lehetővé teszi a pékség online jelenlétének biztosítását és üzleti működésének digitalizálását. Az alkalmazás elsődleges funkciói között szerepel a pékség termékeinek részletes bemutatása, az online rendelési lehetőség biztosítása. Az online rendelési felület lehetővé teszi, hogy a vásárlók kényelmesen, akár otthonról, vagy útközben rendeljenek friss pékárut, miközben a pékség munkatársai könnyedén kezelhetik és nyomon követhetik az érkező rendeléseket.

**A webalkalmazás elsődleges célcsoportjai:**

* Magánszemélyek, akik szeretnének gyorsan és kényelmesen friss pékárut rendelni anélkül, hogy el kellene látogatniuk a pékségbe. Az online rendelés egy lehetőséget ad számukra, hogy személyre szabottan választhassanak a termékek közül, és otthonról vegyék át a megrendelt pékáruikat.
* A pékség munkatársai, akik az online rendelések kezeléséért és az adminisztrációért felelnek. A rendszer segíti őket az online rendelési folyamatok hatékony kezelésében, az egyes rendelésekkel kapcsolatos információk gyors elérésében és nyomon követésében, valamint a készletek és szállítási időpontok könnyű kezelésében.

A projekt hosszú távú célja, hogy bemutassa a digitalizáció előnyeit a helyi vállalkozások számára, különösen olyan kis- és középvállalkozások esetében, mint a Valai Pékség, amelyek számára az online jelenlét és az automatizált rendszerek lehetővé teszik a gördülékenyebb működést és a versenyképesség növelését. Emellett a célunk, hogy az alkalmazás révén az online vásárlási élmény egyszerű és élvezetes legyen, miközben megőrzi a pékség autentikus és barátságos hangulatát. Továbbá, a projekt lehetőséget ad arra, hogy a pékség személyre szabott szolgáltatásokat kínáljon a vásárlók számára, például előre rendelést, házhoz szállítást, vagy egyedi termékajánlatokat, amelyek a helyi közösség számára egyedülálló értéket kínálnak. Az alkalmazás további bővítésekkel és frissítésekkel is segítheti a pékség folyamatos fejlődését és a vásárlói élmény javítását. Ezen kívül a projekt során egy stabil és biztonságos platformot is létrehozunk, amely képes kezelni a növekvő rendelési igényeket és biztosítani a vásárlók adatainak védelmét.

# Projekttervezet

## Felhasználói felület (UI) tervezés

A felhasználói felület célja, hogy a látogatók számára egyszerű, mégis esztétikus és funkcionális élményt nyújtson. Az első benyomás kulcsfontosságú, ezért a főoldal színes, letisztult dizájnnal rendelkezik. Fontosnak tartottuk, hogy passzoljon a Pékség témájához, de ne legyen zsúfolt, így bézs-aranybarna árnyalatok kerültek előtérbe.

**A fő elemek között megtalálható:**

* Pékség logója, amely az oldal vizuális azonosítója.
* Felhasználói értékelések, hogy növeljék a bizalmat az új látogatókban.
* Termékekről készült képek, amelyek az ajánlatok minőségét és változatosságát emelik ki.

**A webalkalmazás fő funkciói:**

* **Termékek böngészése és részletezése:**
  + Az érdeklődők megtekinthetik a termékek listáját képekkel és leírásokkal.
  + A termékek nevére kattintva részletes információkat kapnak az adott áruról, például összetevőkről, árakról és elérhetőségről.
* **Kosár és rendelési folyamat:**
  + A kiválasztott termékek a kosárba kerülnek, ahol az árak automatikusan összeadódnak.
  + A rendelési folyamat végig vezeti a vásárlókat a szállítási és fizetési információk megadásán.
* **Reszponzív navigáció:**
  + A navigációs sáv a reszponzív dizájnnak köszönhetően hamburger menüként jelenik meg kisebb eszközökön.

**Navigációs sáv tartalma:**

* **Főoldal:** Részletesebb bemutató a Valai Pékségről.
* **Bejelentkezés:** A rendeléshez szükséges, de a böngészéshez opcionális.
* **Termékek:** Megjeleníti a termékek oldalát.
* **Admin:** Átvisz minket az admin bejelentkezéshez.

**Bejelentkezés után, egy legördülő menüben:**

* **Kosár:** Áttekintést nyújt a vásárlásra kiválasztott termékekről, összegzéssel és megrendelési lehetőséggel.
* **Rendeléseim:** Megjeleníti előttünk az eddigi rendeléseinket.
* **Adatok módosítása**: Saját adatainkat tudjuk módosítani
* **Kijelentkezés**: Kijelentkezik a fiókból.

**Főoldal**

A főoldalon láthatjuk a termékeinket, melyeket képekkel egészítettünk ki. A termékekre rákattintva láthatjuk azoknak részletes leírásait is. A bal oldali sávban tudunk kategóriákra szűrni, mely megkönnyíti dolgunkat, ha egy adott terméket, vagy kategóriában keresünk.

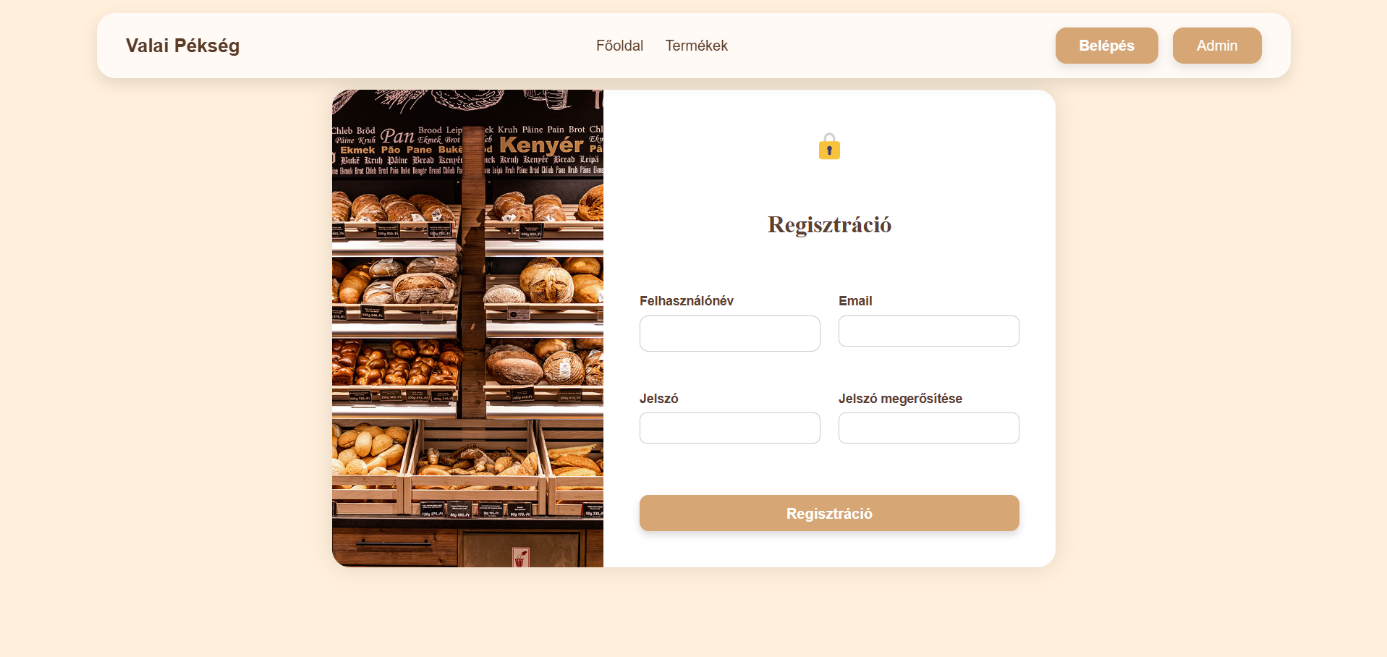
**A fő elemek között megtalálható:**

* Pékség logója, amely az oldal vizuális azonosítója.
* ****Felhasználói értékelések, hogy növeljék a bizalmat az új látogatókban.
* Termékekről készült képek, amelyek az ajánlatok minőségét és változatosságát emelik ki.

## Regisztráció felület

A regisztráció során a felhasználó egyszerűen és biztonságosan megadhatja adatait. Az adatokat a program az adatbázisba feltölti, a jelszó titkosítással. A már beregisztrált adatokkal tudunk bejelentkezni az oldalunkra. A termékeket regisztráció és bejelentkezés nélkül is látjuk, de rendelni csak ezek teljesítése után lesz lehető. Amennyiben ezt nem teljesíti, hiba üzenetet kapunk, ami felszólít a bejelentkezésre.

**A következő információkat kérjük be:**

* **Felhasználónév**
* **Email cím**
* **Jelszó:** kétszer kérjük be az egyezés ellenőrzéséhez

## Bejelentkezés felület

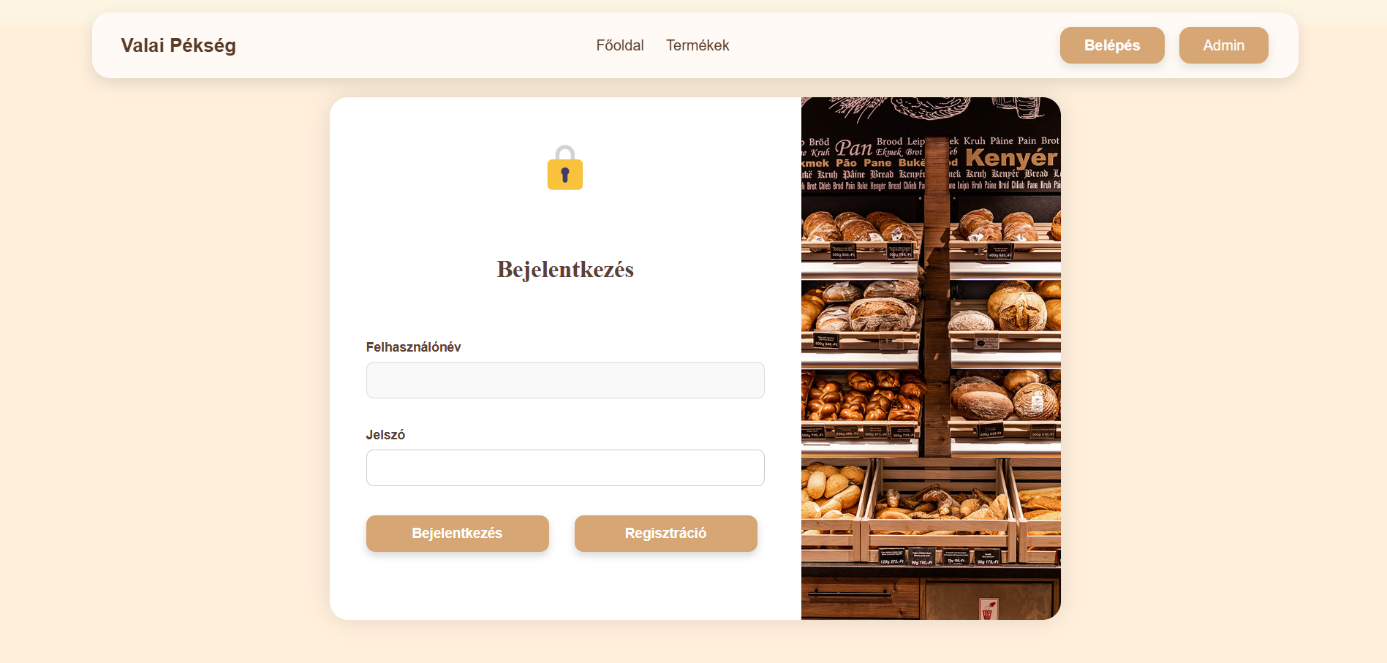
**A bejelentkezés során a felhasználó az alábbi adatokat adja meg:**

* Felhasználónév
* Jelszó

**Hibakezelés:**

* Ha az email cím nem található az adatbázisban, figyelmeztetést jelenítünk meg: „Ez az email cím még nem került regisztrálásra.”
* Ha az email cím helyes, de a jelszó nem egyezik, egy újabb hibaüzenet tájékoztatja a felhasználót.

**Sikeres bejelentkezés után:**

* A felhasználó a kezdőlapra kerül.
* A navigációs sávban a bejelentkezés és regisztráció opció helyett egy kijelentkezési lehetőség jelenik meg.

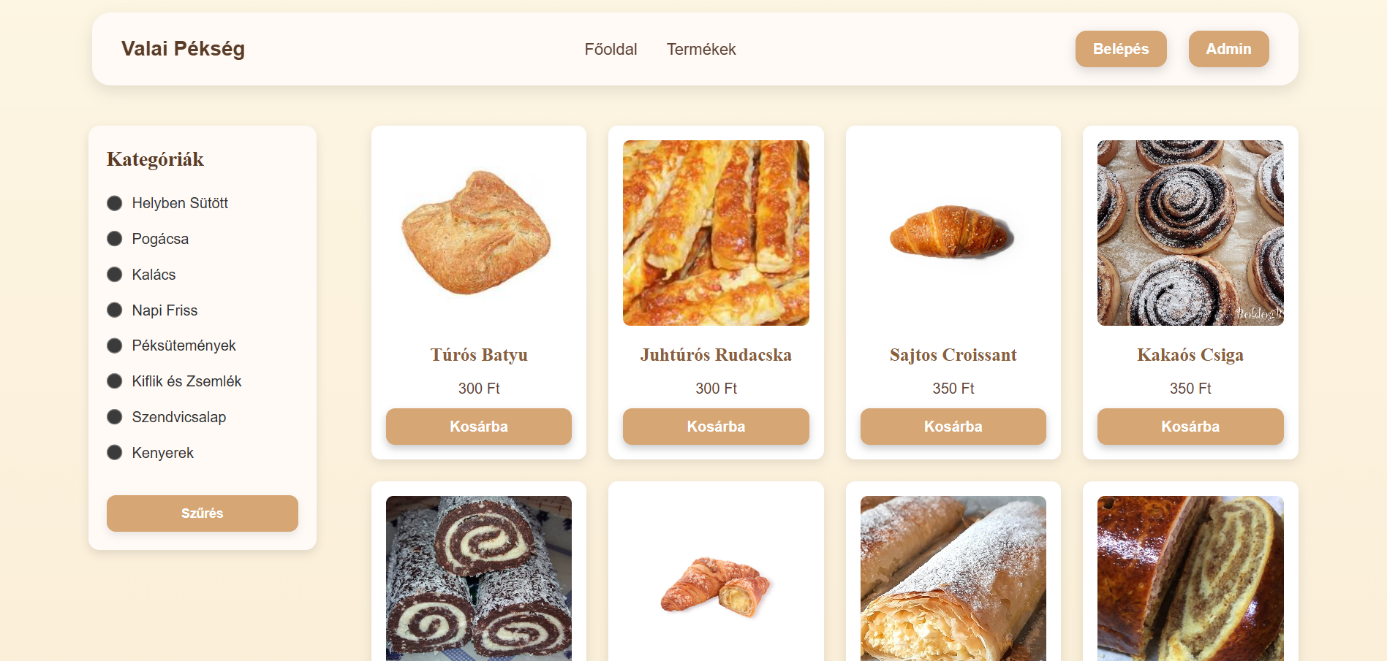
## Termékek felület

A termékek oldalán a felhasználók könnyen és gyorsan böngészhetnek a pékség különböző termékei között, amelyeket kategóriákra bontva láthatnak. A kategóriák segítenek a gyors keresésben, és lehetővé teszik, hogy a felhasználók a kívánt termékeket szűrjék. A termékek minden fontos információt tartalmaznak: a termék nevét, árát és egy képet, hogy vizuálisan is vonzó legyen a választás.

**A termékek oldalán az alábbi funkciók találhatóak:**

* Kosárba gomb: Minden termékhez tartozik egy „Kosárba” gomb, amellyel a felhasználók egyszerűen hozzáadhatják a kiválasztott terméket a kosarukhoz. A gomb színe és stílusa megegyezik a regisztrációs oldalon található gombokkal, hogy egységes élményt nyújtson.
* Kategória szűrők: A bal oldali oldalsávban a felhasználók szűrhetik a termékeket különböző kategóriák szerint, mint például „Pékáru”, „Kenyerek”, vagy „Friss termékek”. Ezáltal könnyen navigálhatnak a termékek között, és gyorsan megtalálhatják a kívánt termékeket.

A felhasználó a termékeket regisztráció és bejelentkezés nélkül is megtekintheti, de a rendeléshez kötelező bejelentkezni. Ha egy felhasználó nem jelentkezett be, és próbál rendelni, akkor hibaüzenetet kap, amely felszólítja a bejelentkezésre.

A vásárlás a regisztrációs felületen keresztül történik, és az ott megadott adatokat a rendszer biztonságosan tárolja.

## Kosár felület

A rendelési folyamatot a kosárban található „Megrendelés” gomb indítja el.

**Az alábbi elemek találhatók a rendelés oldalon:**

* **Rendelés összesítése:**
  + A termékek ára, valamint a teljes fizetendő végösszeg.
* **Vásárlási adatok:**
  + Név
  + Szállítási cím

## Rendelés felület

Itt látható az összes eddigi megredelés.

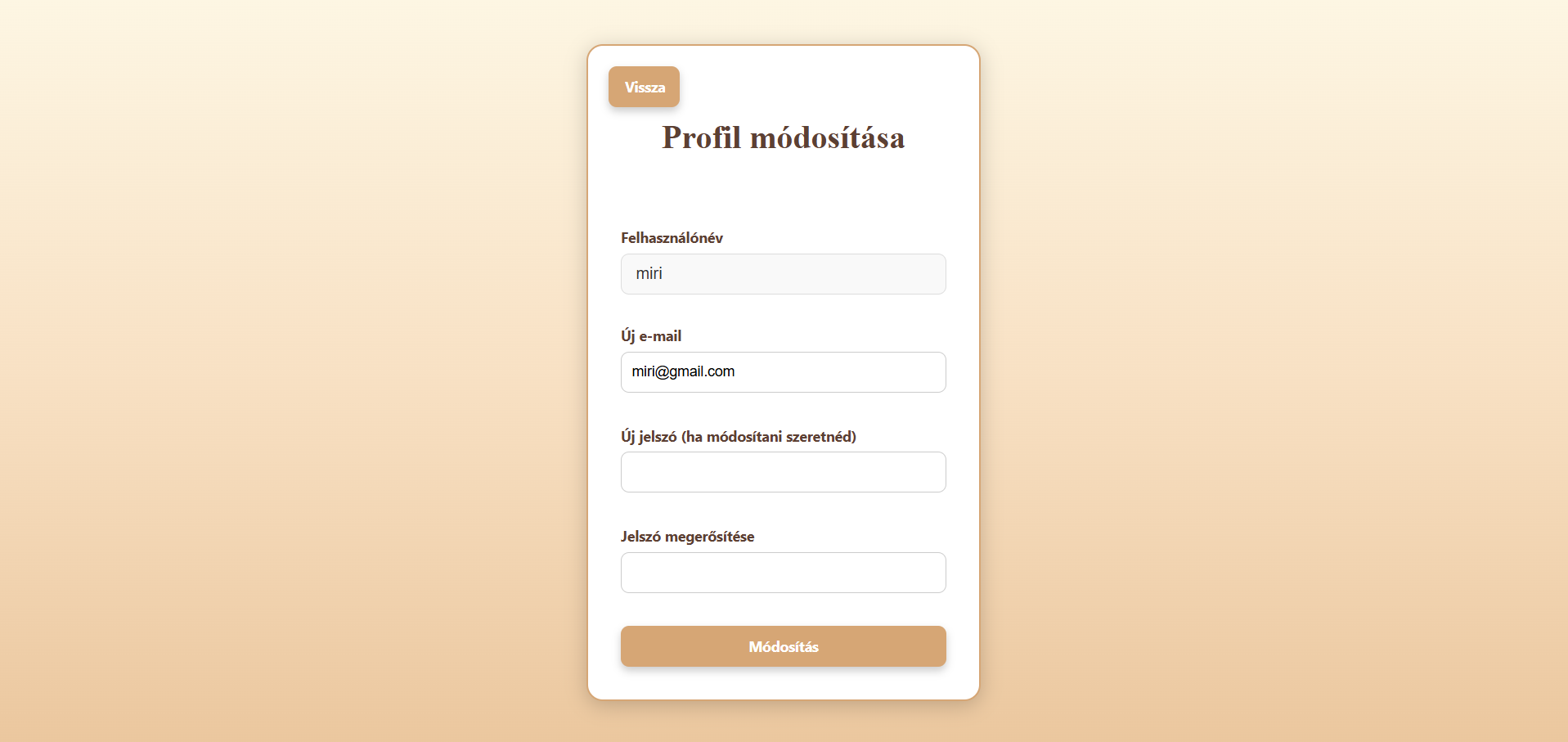
**Az alábbi elemek találhatók a rendelés oldalon:**

* **Rendelés összesítése:**
  + Az összes rendelés dátuma és állapota.

## Adatok módosítása felület

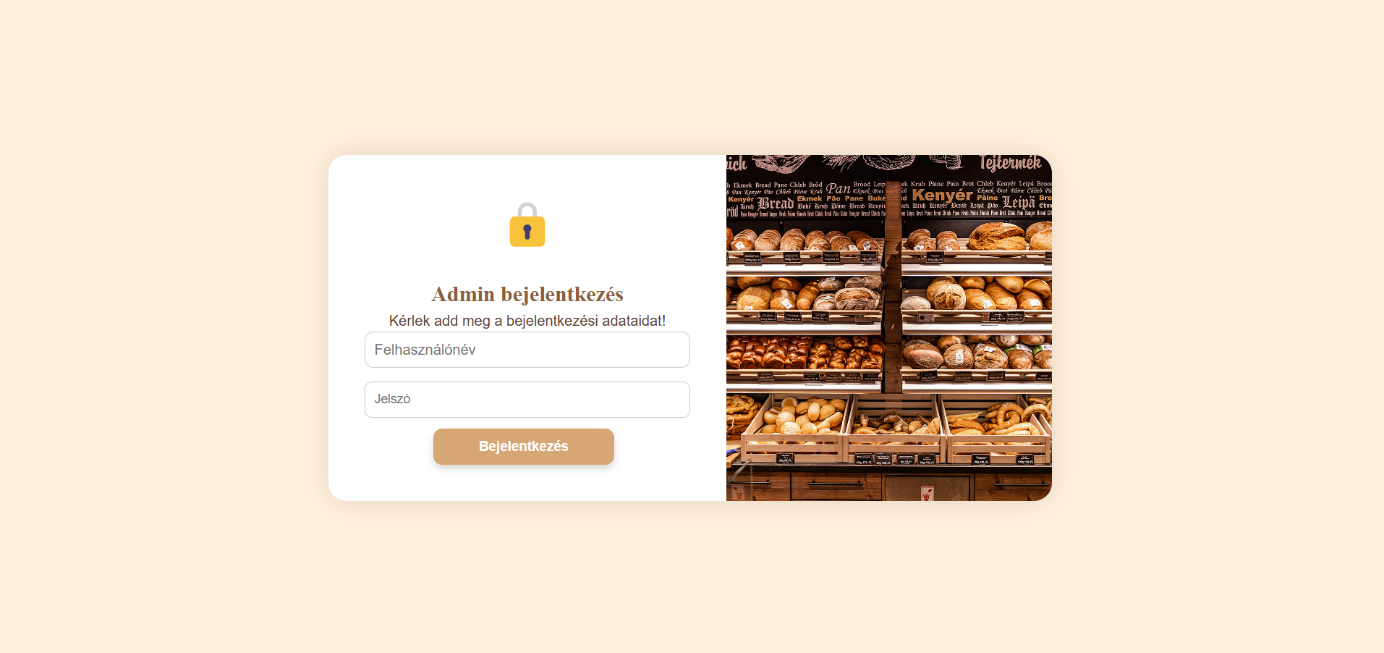
Itt lehet módosítani a felhasználó adatait.

**Az alábbi elemek találhatók a rendelés oldalon:**

* Felhasználónév
* Új e-mail
* Új jelszó és annak megerősítése

## Admin felület

**Login**

* Admin felhaszálónév
* Admin Jelszó

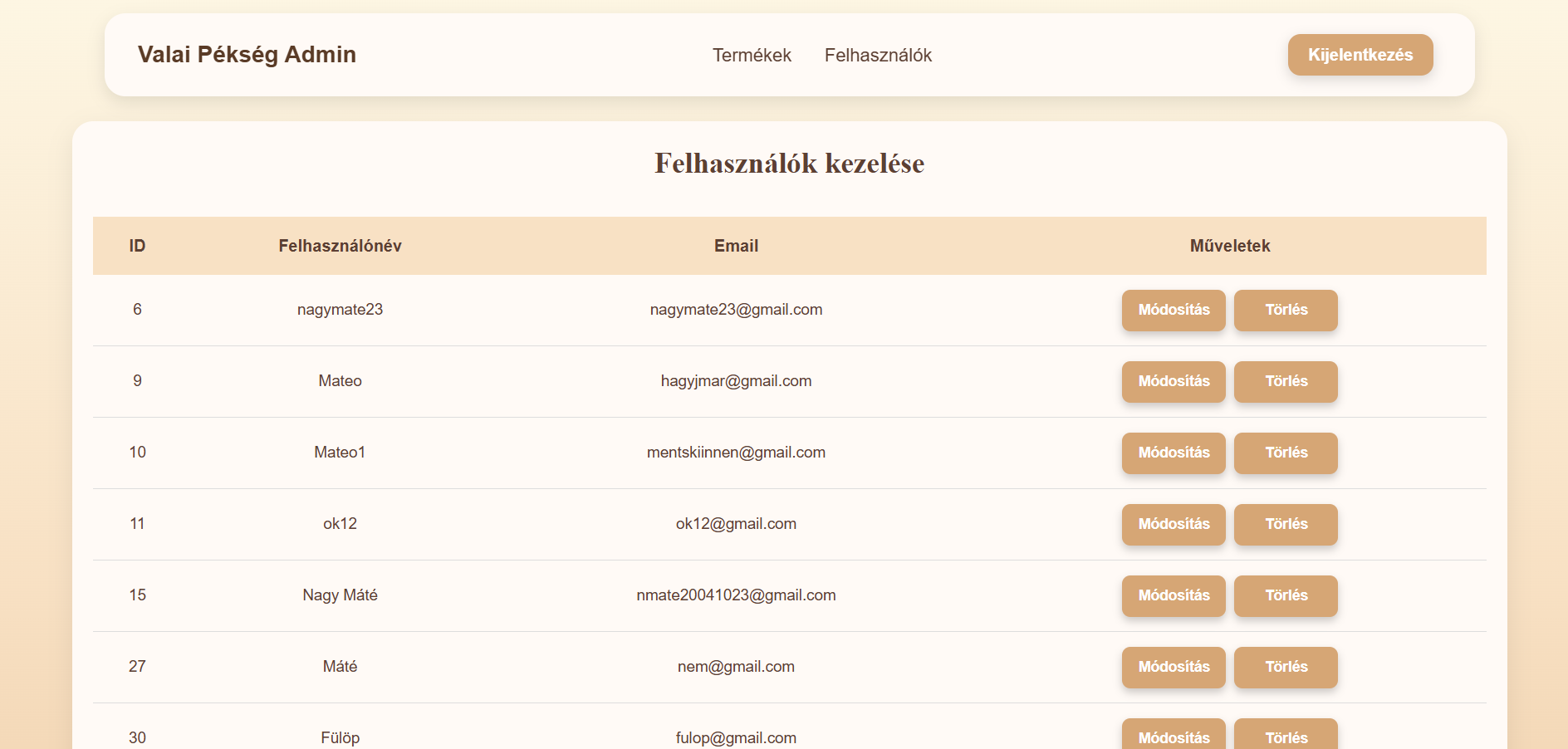
**Vezérlőpult**

A jövőben itt lehet látni a statisztikákat

**Termékek kezelése**

Itt lehet az árakat módosítani és a termékeket törölni. Amennyiben nem adunk meg új árat, és úgy akarunk módosítani kapunk egy figyelmeztető üzenetet.

**Felhasználók kezelése**

Itt lehet új jelszót létrehozni a felhasználók számára, és itt lehet törölni adott felhasználókat.

# Felhasznált programok

**Discord**

A Discord egy sokoldalú kommunikációs platform, amelyet a projekt során elsősorban az egymás közötti kommunikációra használtunk.

**Az alkalmazás lehetővé tette, hogy:**

* Hívásokat bonyolítsunk le, amelyek során részletesen megbeszélhettük a projekt különböző részeit és megoldásait.
* Részfeladatok screenshot-jainak tárolását és megosztását végezzük el, így mindkét fél könnyedén hozzáférhetett a releváns információkhoz és képekhez.
* Az archiválási lehetőség révén visszakereshetővé váltak a korábbi beszélgetések és megosztott anyagok, amelyek fontos szerepet játszottak az együttműködés folytonosságában.

A Discord intuitív felülete és stabil teljesítménye nagyban hozzájárult a hatékony csapatmunkához.

**TeleportHQ**

A TeleportHQ egy online, kollaborációra optimalizált design és fejlesztési eszköz, amelyet a projekt vizuális tervezési és fejlesztési szakaszában használtunk.

**A TeleportHQ-ban:**

Weboldalterveket készítettünk, beleértve a főoldal, a regisztrációs és bejelentkezési oldal, valamint a rendelési felület vizuális megjelenésének kidolgozását. Az eszköz lehetővé tette számunkra, hogy az alkalmazás összes felhasználói felületét részletesen megtervezzük és könnyen testre szabhassuk.

Ez a program kiemelkedő rugalmasságot biztosított számunkra, mivel platformfüggetlen, és bárhonnan elérhető, amíg van internetkapcsolatunk.

**Visual Studio Code**

A Visual Studio Code volt az elsődleges fejlesztési környezetünk a React frontend fejlesztéséhez.

**A program több szempontból is előnyös volt számunkra:**

* Technológiai támogatás: A projekt React, TypeScript, HTML és CSS alapú, amelyekhez a VS Code hatékony eszköztárat biztosított.
* Bővítmények és integrációk: Használtunk automatikus kódkiegészítést, fejlett hibakereső eszközöket és GitHub integrációt.
* Verziókövetés: A GitHub támogatás révén a kódot közvetlenül a VS Code-on belül kezelhettük és verziózhattuk.

A Visual Studio Code intuitív felülete és könnyű használhatósága jelentősen megkönnyítette a fejlesztési folyamatot.

**Visual Studio 2022**

A Visual Studio 2022 volt az elsődleges fejlesztési környezetünk, amelyben az ASP.NET Core backend fejlesztését végeztük.

**A program előnyei:**

* Korszerű fejlesztői eszközök: Beépített hibakereső, automatikus kódkiegészítés és Entity Framework Core támogatás.
* Projektmenedzsment támogatás: Könnyen kezelhető megoldásokat biztosított az ASP.NET Core API fejlesztéséhez.
* GitHub és Git integráció: Verziókövetési és együttműködési lehetőségeket kínált közvetlenül a fejlesztői környezetben.

A Visual Studio 2022 biztosította a stabil és hatékony fejlesztési környezetet a backend számára.

**Xampp**

Az XAMPP egy könnyen telepíthető szoftvercsomag, amely a webes fejlesztésekhez szükséges szervereket és eszközöket tartalmazza.

* Adatbáziskezelés: A PhpMyAdmin felületén hoztuk létre és kezeltük az adatbázisunkat, amely tartalmazza a felhasználók, termékek és rendelések adatait.
* Helyi szerver környezet: Az XAMPP lehetővé tette, hogy helyi környezetben futtassuk és teszteljük az alkalmazást, így gyorsabban azonosíthattuk és javíthattuk az esetleges hibákat.
* Kompatibilitás: A program kiválóan működik több operációs rendszeren is, és egyszerűen kezelhető grafikus felülete megkönnyítette a használatát.

**GitHub**

A GitHub volt a projektünk verziókövetési és tárolási platformja.

**Az alkalmazás számos előnyt nyújtott:**

* Kód tárolása: A projekt forráskódja biztonságosan tárolva volt, így bármikor hozzáférhettünk.
* Verziókezelés: Az egyes fejlesztési szakaszok pontos nyomon követésére és a korábbi verziók visszaállítására is lehetőséget biztosított.
* Csapatmunka támogatása: A pull request-ek és merge request-ek segítségével egyszerűen egyesíthettük az egyéni fejlesztési munkáinkat.
* Nyilvános hozzáférés: A projekt végén a GitHub platformja biztosította, hogy az elkészült munka megosztható legyen másokkal is.

A GitHub nemcsak a kód tárolásában segített, hanem a csapatmunka szervezésében és a fejlesztési folyamat dokumentálásában is kulcsszerepet játszott.

**PowerPoint**

A Microsoft PowerPoint egy prezentációkészítő szoftver, amelyet a projekt során az eredmények és a fejlesztési folyamat bemutatására használtunk.

**A PowerPoint lehetőségei:**

* Diavetítések létrehozása: A projekt során készített tartalmak vizuális és strukturált bemutatása.
* Multimédiás elemek beágyazása: Képek, videók és animációk alkalmazása a bemutató érthetőségének növelésére.
* Együttműködés: Lehetővé teszi a csapat számára a közös szerkesztést és visszajelzések megosztását.
* Professzionális dizájn: Előre elkészített sablonok és testre szabható elrendezések segítették a vizuális megjelenés egységességét.
* Könnyű exportálás: Az elkészült bemutatók megoszthatók különböző formátumokban (PDF, PPTX, videó).

A PowerPoint segítségével a projekt összegzését és bemutatását hatékonyan tudtuk megvalósítani, biztosítva az érthetőséget és a professzionális megjelenést.

# Adatbázis

Az alkalmazásunk adatkezeléséhez MySQL-alapú adatbázist használunk, amelyet az ASP.NET backendben az Entity Framework Core ORM és a Visual Studio 2022 segítségével kezelünk. Fejlesztés során az adatbázist egy helyi MySQL szerveren futtattuk, míg az éles környezetben egy távoli adatbázis-kiszolgálóra kerül. Az adatkapcsolatot és a lekérdezések végrehajtását az ASP.NET Web API biztosítja, amely a React frontend számára továbbítja az adatokat.

**Az adatbázis főbb jellemzői:**

* Struktúra: Az adatokat külön táblákban tároljuk, amelyek a felhasználók, termékek, rendelések és kategóriák adatait kezelik.
* Kapcsolatok: Az adatbázis kapcsolatokkal van optimalizálva, így biztosítva az adatok konzisztenciáját és gyors elérését.
* Kezelés és karbantartás: Az adatbázis-műveleteket az Entity Framework Core ORM segítségével végezzük, amely egyszerűsíti a migrációkat és a lekérdezéseket.
* Biztonság: A felhasználói adatok védelméről titkosítási és jogosultsági szintek gondoskodnak.
* Hatékonyság: Az adatbázis indexelése és optimalizált lekérdezései biztosítják a gyors válaszidőt még nagy mennyiségű adat esetén is.

## Táblák

1. Adminok tábla
   * Tárolja az adminisztrátorok adatait (id, felhasználónév, jelszó).
   * Az id mező az egyedi azonosító (PRIMARY KEY), míg a felhasznalonev UNIQUE kulcs
2. Kategoriak tábla
   * Tárolja a termékkategóriákat
   * Az id egyedi azonosító, a nev pedig UNIQUE kulcs, biztosítva, hogy ne legyenek duplikált kategórianevek.
3. Termekek tábla
   * Tárolja a termékek adatait (id, név, leírás, ár, kategória\_id, kép URL).
   * Az id mező az egyedi azonosító. A kategoria\_id itt is idegen kulcs, ami a kategoriak táblára mutat. A kategória törlésekor a mező értéke szintén NULL lesz.
4. Rendelesek tábla
   * Tárolja az egyes rendelések adatait (id, vevő\_id, rendelés dátuma, állapot).
   * Az id mező az egyedi azonosító. A vevo\_id idegen kulcs a vevo táblára. Ha egy vevőt törölnek, a hozzá kapcsolódó rendelések is törlődnek (ON DELETE CASCADE).
5. Keszlet tábla
   * Tárolja a termékek készletinformációit (id, termék neve, mennyiség, ár, kategória\_id).
   * Az id egyedi azonosító. A kategoria\_id idegen kulcs, amely a kategoriak táblára mutat. Amennyiben egy kategória törlődik, az ON DELETE SET NULL szabály érvényesül, így a kategória törlésével a kapcsolódó mezők NULL értéket kapnak.
6. Rendeles\_tetelek tábla
   * Tárolja a rendelésekhez tartozó termékeket és azok mennyiségét (id, rendelés\_id, termék\_id, mennyiség, összeg).
   * Az id egyedi azonosító. A rendeles\_id a rendelesek táblára mutat, míg a termek\_id a termekek táblára. Ha egy rendelést vagy terméket törölnek, az ide kapcsolódó sorok is törlődnek (ON DELETE CASCADE).
7. Vevo tábla
   * Tárolja a vásárlók adatait (id, felhasználónév, email, jelszó, regisztráció dátuma).
   * Az id mező az egyedi azonosító. A felhasznalonev és az email mezők UNIQUE kulcsok, amelyek megakadályozzák a duplikált regisztrációkat.

## ER Adatmodell

A kapcsolatok kialakítása miatt, ha az egyik táblában változik valami, akkor a másik tábla is módosításra kerül. Ha például egy táblában törlünk egy rekordot, ami más táblában hivatkozásokkal rendelkezik, az adatbázis beállításaitól függően az adott hivatkozásokat is automatikusan frissíteni, törölni vagy újra értelmezni kell. Ez segít elkerülni az inkonzisztens adathalmazokat, és biztosítja, hogy az adatok közötti kapcsolatok mindig megfelelőek maradjanak.

Kapcsolatok és integritási szabályok:

**Kategóriák és termékek/készlet:**

A kategóriák tábla azonosítóit idegen kulcsként használják a termékek és készletek táblái. Így egy termék vagy készlet sor egy adott kategóriához tartozik. A kategória törlése esetén a kapcsolódó sorok kategória mezői NULL értéket kapnak.

**Vevők és rendelések:**

A vevő tábla azonosítóját használják a rendelések tábla idegen kulcsaként. Ha egy vevőt törölnek, az összes rendelése automatikusan törlődik (CASCADE).

**Rendelések és rendelés tételek:**

Egy rendeléshez több rendelés tétel tartozhat, amit a rendelés\_id oszlop határoz meg. Ha egy rendelést törölnek, a hozzátartozó tételek is automatikusan törlődnek (CASCADE).

**Rendelés tételek és termékek:**

A rendelés tételek tábla termék\_id oszlopa a termékek tábla azonosítójára mutat. Ha egy terméket törölnek, az ehhez tartozó rendelés tételek szintén törlődnek (CASCADE).

Ez az adatbázis-struktúra biztosítja az alkalmazás hatékony adatkezelését, lehetővé téve a termékek, rendelések és készletek precíz nyomon követését. Az adatok biztonságos tárolása és az optimalizált lekérdezések révén a rendszer gyors és megbízható működését támogatja.

# **Frontend**

Az alkalmazásunk felhasználói felülete a modern és reszponzív React.js keretrendszer segítségével készült. A React lehetővé teszi az újrafelhasználható komponensek kialakítását, a gyors renderelést és az egyszerű állapotkezelést. A fejlesztést a Visual Studio Code környezetben végeztük, külön frontend mappában. A React alkalmazás külön Node.js környezetben fut, és az ASP.NET backenddel kommunikál HTTP-kérések útján.

**A frontend főbb jellemzői:**

* Modularitás: Minden funkció saját komponensben került megvalósításra, ezzel is támogatva az átláthatóságot és karbantarthatóságot.
* Navigáció: A react-router-dom segítségével biztosítottuk a többoldalas böngészést (SPA).
* Dizájn: Saját CSS-alapú reszponzív stílusokat alkalmaztunk, amelyek biztosítják a mobil- és asztali megjelenítést.
* Állapotkezelés: Lokális állapot (useState, useEffect) és localStorage segítségével oldottuk meg a bejelentkezési információk és kosáradatok tárolását.
* Képfeldolgozás: A backendből kapott BLOB-formátumú képeket URL.createObjectURL() segítségével alakítjuk megjeleníthetővé.

## Komponensek

1. **Login komponens**

* Lehetővé teszi a felhasználók bejelentkezését.
* Az email és jelszó mezők kitöltése után az adatok POST kérésként kerülnek továbbításra a /api/Auth/login végpontra.
* Sikeres válasz esetén a felhasználót átirányítjuk a főoldalra, és az azonosító adatokat elmentjük a localStorage-ba.

1. **Registration komponens**

* Lehetővé teszi új felhasználók regisztrációját (felhasználónév, email, jelszó).
* Az adatok POST kérés formájában kerülnek továbbításra a /api/Account/register végpontra.
* Sikeres regisztráció esetén a felhasználó a bejelentkezési oldalra kerül.

1. **MainPage2 komponens**

* A főoldal, ahol az összes termék megtekinthető.
* A termékek GET kérés segítségével töltődnek be a backendből.
* A termékek megjelenítése kártyás elrendezésben történik: név, ár, kép, kategória.
* A kosárba helyezéshez egy number típusú mezőt használunk, ahol a mennyiség 1 és 10 közötti szám lehet.
* A kiválasztott termékek a kosárba kerülnek, ahonnan rendelés indítható.

1. **Cart komponens**

* A kosár tartalmát jeleníti meg.
* A felhasználó itt láthatja és módosíthatja a termékek mennyiségét.
* A "Rendelés leadása" gomb POST kérésben továbbítja az adatokat a backend felé, ahol a rendeles és rendeles\_tetelek táblákba kerülnek bejegyzésre.
* A rendelés automatikusan „Feldolgozás alatt” állapottal kerül mentésre.

1. **EditProfile komponens**

* A felhasználói profil módosítását teszi lehetővé.
* Az email és jelenlegi jelszó megadásával történik a hitelesítés.
* Sikeres azonosítás után lehetőség nyílik az új email és jelszó megadására.
* Az adatok PUT kérés formájában frissülnek az adatbázisban.

1. **AdminDashboard komponens**

* Az admin számára készített áttekintő felület.
* Statisztikai kártyákon jelennek meg a legfontosabb mutatók: aktív rendelések száma, top termékek, bevétel.
* A diagramokat a Recharts könyvtár segítségével jelenítjük meg.

1. **AdminOrders komponens**

* Megjeleníti az összes aktív („Feldolgozás alatt”) rendelést.
* A rendelés állapota módosítható „Kiszállítva” vagy „Teljesítve” értékre.
* Ha az állapot ezek valamelyikére változik, a rendelés automatikusan törlődik a felületről és az adatbázisból (a backend ezt kezeli).

1. **AdminProducts komponens**

* Megjeleníti az összes terméket egy táblázatban.
* Minden terméknél lehetőség van az ár módosítására (PUT kérés).
* A termékek kereshetőek, és gombbal törölhetőek (DELETE kérés).

A navigáció feltételekhez kötött: az admin oldalak csak belépett admin felhasználó számára érhetők el.

## Reszponzivitás és dizájn

* Az alkalmazás mobilon és asztali eszközökön is egyaránt használható.
* A CSS fájlokat külön kezeltük minden komponenshez a Style/ mappában.
* Rugalmas flexbox és grid elrendezéseket használtunk.
* Saját hamburger menü biztosítja a kényelmes navigációt mobilon.
* A termékek kártyás elrendezése automatikusan átrendeződik kis kijelző esetén.

**Felhasználói élmény és biztonság**

* A felhasználói adatokat csak sikeres hitelesítés után lehet módosítani.
* A kosárkezelés valós időben történik.
* Az admin felületen csak a nem teljesített rendelések jelennek meg.
* A képek biztonságos módon, Blob formátumban töltődnek be a React komponensekbe.

Ez a frontend-struktúra biztosítja a felhasználóbarát működést, a megbízható adatkezelést, valamint a professzionális adminisztrációs funkciókat. A moduláris React architektúra lehetővé teszi a projekt bővítését és hosszú távú karbantartását.

# Backend

Az alkalmazás backendje ASP.NET Core Web API technológiával készült a Visual Studio 2022 fejlesztői környezetben. A szerveroldali logika biztosítja az adatbázis-kezelést, a felhasználói hitelesítést, a rendelések feldolgozását és az adminisztrációs műveleteket. A backend az Entity Framework Core ORM segítségével kommunikál a MySQL adatbázissal, így az adatkezelés hatékony, biztonságos és jól karbantartható.

**A backend főbb jellemzői:**

* REST API architektúra: minden funkció önálló végponton keresztül érhető el.
* Biztonság: a felhasználói adatok kezelése titkosítással és belépési ellenőrzéssel történik.
* Modularitás: külön controllerek kezelik az egyes funkciókat (regisztráció, bejelentkezés, termékek, rendelések stb.).
* ORM: az adatbázis-kezelést az Entity Framework Core biztosítja migrációkkal és LINQ lekérdezésekkel.
* Képtárolás: a termékképek BLOB-ként tárolódnak a MySQL adatbázisban, a backend bináris adatként kezeli ezeket.

**Fő komponensek**

* AuthController
  + Funkció: felhasználók bejelentkezése
  + Végpont: POST /api/Auth/login
  + Bemenet: email, jelszó
  + Működés:
    - Ellenőrzi az e-mail és jelszó egyezését.
    - Sikeres belépés esetén visszaad egy sikerüzenetet (vagy tokent, ha szükséges).
* AccountController
  + Funkció: új felhasználók regisztrálása és adatmódosítás
  + Végpontok:
    - POST /api/Account/register – új felhasználó mentése
    - PUT /api/Account/update-profile – email, jelszó módosítása
  + Működés:
    - Regisztráció során ellenőrzi, hogy az email vagy felhasználónév már létezik-e.
    - Profilmódosítás előtt hitelesíti a jelenlegi jelszót.
* VevoController (Admin)
  + Funkció: felhasználók listázása, módosítása, törlése (csak admin részről)
  + Végpontok:
    - GET /api/Vevo – összes felhasználó
    - PUT /api/Vevo/{id} – adatmódosítás
    - DELETE /api/Vevo/{id} – felhasználó törlése
* TermekController
  + Funkció: termékek kezelése (listázás, módosítás, törlés)
  + Végpontok:
    - GET /api/Termek – összes termék lekérdezése
    - PUT /api/Termek/{id} – ár módosítása
    - DELETE /api/Termek/{id} – törlés
  + A képek a backend által BLOB formátumban kerülnek a frontendhez, application/octet-stream MIME típussal.
* RendelesController
  + Funkció: rendelések kezelése és állapotváltás
  + Végpontok:
    - POST /api/Rendeles – új rendelés létrehozása
    - GET /api/Rendeles/user/{id} – felhasználó rendelései
    - PUT /api/Rendeles/status/{id} – rendelés állapotának módosítása
  + Működés:
    - Alapértelmezett állapot: „Feldolgozás alatt”
    - Ha az állapot „Kiszállítva” vagy „Teljesítve”, a rendelés automatikusan törlődik a kapcsolódó tételekkel együtt.
* AdminDashboardController
  + Funkció: admin statisztikák adatainak előállítása
  + Végpont: GET /api/AdminDashboard/statistics
  + Visszatér: diagramokhoz szükséges adatok (eladások, kategóriaeloszlás stb.)

# Tesztelés

## Frontend teszt

A frontend felület működését manuális és automatizált teszteléssel is ellenőriztük. Az automatizált teszteléshez Selenium keretrendszert használtunk, amely lehetővé tette a valós felhasználói interakciók szimulál

**Automatizált tesztelés (Selenium)**

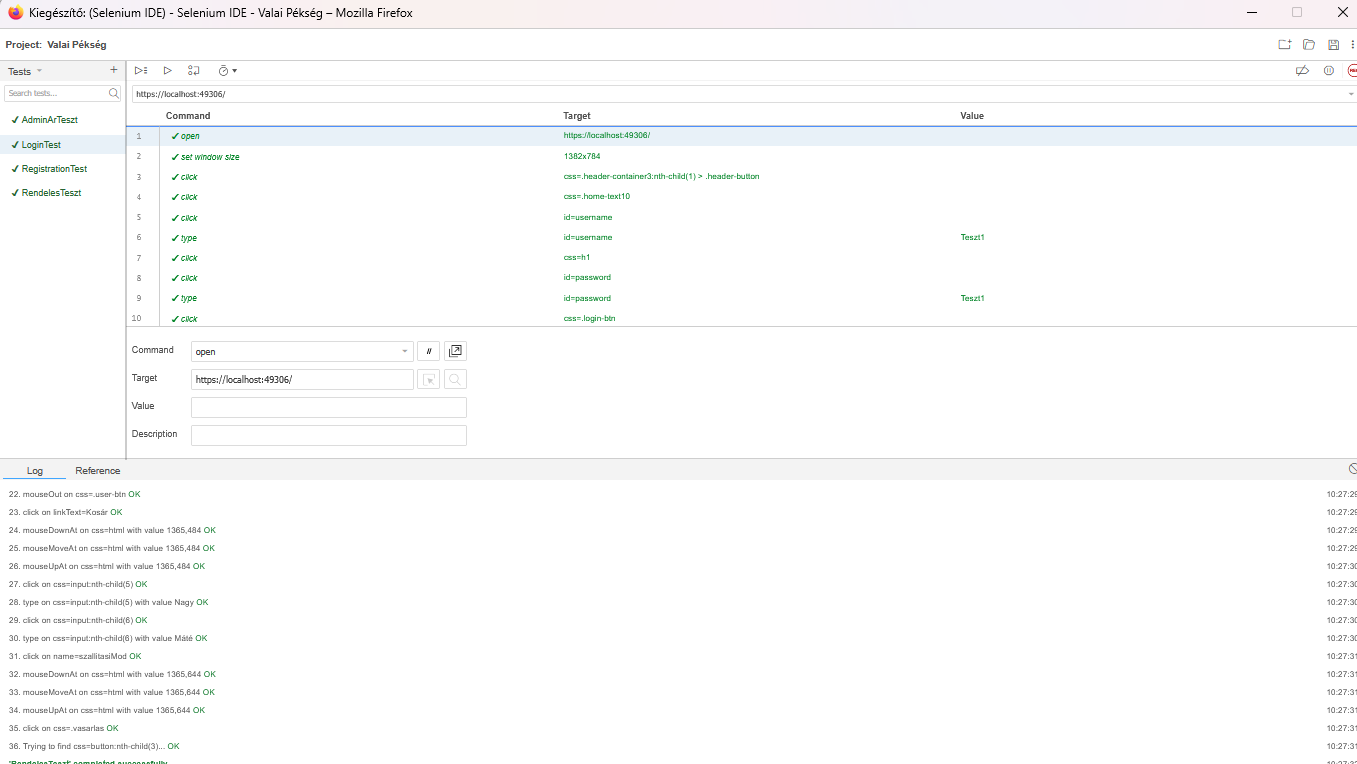
* A Selenium segítségével végigteszteltük a legfontosabb felhasználói folyamatokat: regisztráció, bejelentkezés, termék böngészés, vásárlás, valamint admin oldali műveletek.
* Teszteltük az űrlapmezők helyes működését (pl. validációk), a gombok reakcióját, a navigációt és a visszajelző üzenetek megjelenését is.
* Ellenőriztük, hogy helytelen bejelentkezési adatok esetén hibaüzenet jelenik meg, illetve, hogy a sikeres bejelentkezés után a felhasználó a főoldalra kerül.

**Reszponzív tesztelés**

* A reszponzivitást manuálisan teszteltük különböző eszközökön és képernyőméreteken (asztali, tablet, mobil).
* A tesztek során figyeltük a navigációs menük, termékkártyák, űrlapok és visszajelző elemek mobilos megjelenését.

**Felhasználói tesztelés**

* A rendszer kipróbálása valós felhasználókkal is megtörtént. A tesztelők visszajelzései alapján finomítottuk a kezelőfelületet és a hibakezelést.
* Kiemelt figyelmet fordítottunk a használhatóságra és a hibamentes működésre.

A Selenium alapú automatizált tesztelés jelentősen hozzájárult a rendszer megbízhatóságához, és segített a hibák gyors felismerésében a fejlesztési ciklus során.

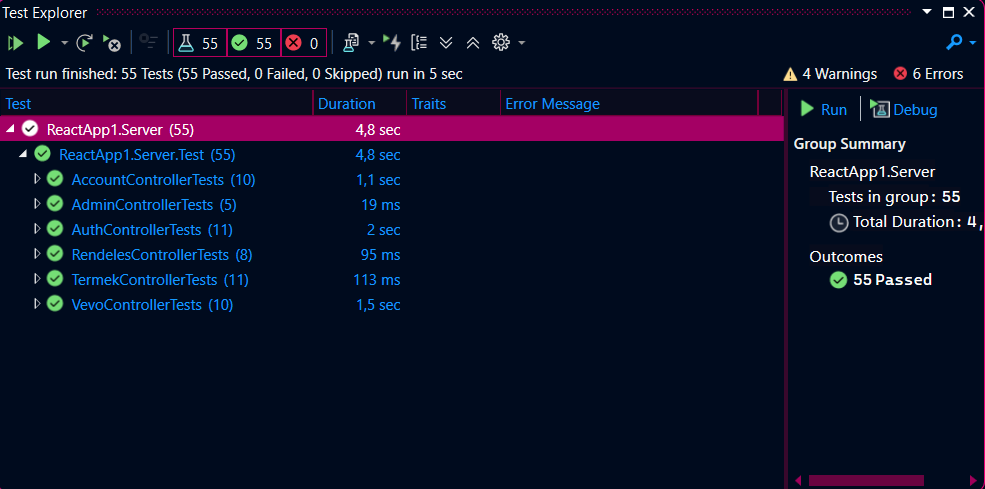
## Backend

A backend oldal stabilitásának és üzleti logikájának megbízhatóságát NUnit segítségével végzett automatizált egységtesztekkel biztosítottuk. A cél az volt, hogy a szerveroldali funkciók különböző helyzetekben is kiszámíthatóan és hibamentesen működjenek.

**Automatizált tesztelés (NUnit)**

* Az egységtesztek az NUnit keretrendszert használják.
* A tesztek futtatása gyors, környezetfüggetlen és megbízható.

**Tesztelt funkciók**

* Felhasználói regisztráció és bejelentkezés
  + Teszteltük, hogy sikeres regisztráció történt.
  + Hibás vagy hiányos adatok.
  + Duplikált e-mail vagy felhasználónév esetén megfelelő figyelmeztetés történik.
* Rendelések kezelése
  + Rendelés létrehozása helyes adatokkal.
* Adatmodell validáció
  + Ellenőriztük, hogy a kötelező mezők hiányában megfelelő hibaüzenet keletkezik.
  + Érvénytelen e-mail cím vagy túl rövid jelszó esetén a szerver elutasítja a kérést.
* Kivételkezelés
  + Ellenőriztük, hogy kivételek esetén naplózás és megfelelő válasz történik.

A tesztelések összességében biztosítják, hogy a \*\*Valai Pékség\*\* alkalmazás \*\*megbízható, biztonságos és felhasználóbarát\*\* módon működjön minden platformon.

# Fejlesztések

**Kiterjesztett termékkategóriák**

További termékkategóriák hozzáadása, mint például gluténmentes vagy vegán termékek.

**Felhasználói profil**

A regisztrált felhasználók számára részletes profiloldal létrehozása, ahol láthatják korábbi rendeléseiket, módosíthatják személyes adataikat.

**Értékelések és vélemények**

Lehetőség biztosítása a felhasználók számára, hogy értékeljék a termékeket és véleményt írjanak.

**Push értesítések**

Bevezetjük az értesítések funkciót, hogy a vásárlók értesülhessenek az új termékekről, akciókról vagy a rendelésük állapotáról.

**Mobilalkalmazás**

A jövőben a webalkalmazás mellé natív mobilalkalmazás készítése iOS és Android platformokra.

# Reflektálás

**Csapatmunka és feladatmegosztás**

* Megtanultuk, hogyan lehet hatékonyan együtt dolgozni egy fejlesztői csapatban.
* A feladatok megfelelő kiosztásával és felelősségi körök meghatározásával gördülékenyebbé tettük a munkafolyamatot.
* A rendszeres kommunikáció és a közös problémamegoldás segített a hatékony együttműködésben.

**Technikai fejlődés**

* Mélyebb ismereteket szereztünk a reszponzív designról, az adatbázis-kezelésről és a biztonsági alapelvekről.
* Az Entity Framework Core és a MySQL használatával gyakorlati tapasztalatot szereztünk az adatkezelésben.
* Az ASP.NET Web API és a React integrációjának megvalósítása során elmélyítettük tudásunkat a modern webfejlesztési technológiákban.

**Problémamegoldás és rugalmasság**

* A projekt folyamán felmerült nehézségeket sikerült kreatívan és hatékonyan megoldani.
* Hibakezelési stratégiákat alkalmaztunk, hogy minimalizáljuk a rendszerösszeomlásokat és az adatvesztéseket.
* A váratlan technikai akadályok ellenére rugalmasan tudtunk reagálni, és gyorsan alkalmazkodtunk az új kihívásokhoz.

**Felhasználói élmény és teljesítményoptimalizálás**

* Törekedtünk arra, hogy a felhasználói felület intuitív és könnyen kezelhető legyen.
* Az alkalmazás teljesítményét optimalizáltuk a gyors betöltési idő és a gördülékeny működés érdekében.
* Teszteléssel és visszajelzések alapján folyamatosan finom hangoltuk a rendszert.

**Összegzés és jövőbeli fejlesztési irányok**

* A projekt során megszerzett tapasztalatok segítenek minket a jövőbeli fejlesztések során.
* Az eddigi munkánkat alapul véve tovább fejleszthetjük az alkalmazást új funkciókkal, például mobilalkalmazás-támogatással vagy mesterséges intelligencia alapú ajánlórendszerrel.
* A következő lépések között szerepel a felhasználói visszajelzések beépítése és a biztonsági rések további csökkentése.

Ez a projekt nemcsak technikai tudásunkat bővítette, hanem hozzájárult a csapatmunkában és problémamegoldásban való fejlődésünkhöz is.

# Források

**Fejlesztési dokumentációk és segédanyagok**

* HTML és CSS dokumentáció: MDN Web Docs – Részletes leírás és példák a modern webfejlesztéshez.
* JavaScript és React dokumentáció: React hivatalos dokumentáció – A React könyvtár hivatalos útmutatója és dokumentációja.
* ASP.NET Core dokumentáció: Microsoft Learn – Az ASP.NET és Entity Framework használatához szükséges források.
* MySQL dokumentáció: MySQL Reference Manual – Az adatbázis-kezelés alapjai és haladó technikák.

**Tervezési és fejlesztési eszközök**

* TeleportHQ: Weboldal tervezési eszköz – UI/UX tervezéshez és prototípus készítéshez használt eszköz.
* GitHub: Verziókövetés és projekt tárolás – A kód verziókövetése és csapatmunka támogatása.
* Visual Studio 2022: Fejlesztőkörnyezet az ASP.NET Core backend és az Entity Framework Core kezelésére.
* Visual Studio Code: Könnyűsúlyú kódszerkesztő a frontend fejlesztéshez.

**Könyvtárak és keretrendszerek**

* React: Komponensalapú JavaScript könyvtár a frontend fejlesztéshez.
* Bootstrap / Tailwind CSS: Stílusok és reszponzív megjelenés biztosítására.
* Axios: HTTP kérések kezelésére a frontend és a backend között.
* Entity Framework Core: ORM rendszer az adatbázis kezeléséhez.

**Tesztelési és hibakeresési eszközök**

* Postman: REST API tesztelésére és hibakeresésére használt eszköz.
* Google Chrome DevTools: Böngészőbe épített fejlesztői eszközök a weboldal optimalizálásához.
* Jest és React Testing Library: Egység- és integrációs tesztelések elvégzésére.

**Adatbázis és szerverkezelés**

* MySQL Workbench: Grafikus felület az adatbázis-kezeléshez.
* Docker: Konténerizált környezet létrehozására és futtatására.

**Segédprogramok**

* ChatGPT
* Cody
* W3Schools

Ezek a források és eszközök biztosították a projektünk gördülékeny fejlesztését és megvalósítását.