**ZÁRÓDOLGOZAT**

**Hegyi gergely**

**SZOFTVERFEJLESZTŐ**

**BAJAI SZC TÜRR ISTVÁN TECHNIKUM**

**SZOFTVERFEJLESZTŐ**

ZÁRÓDOLGOZAT

PIZZÉRIA RENDELÉS Asztali alkalmazás

**Hegyi Gergely**

2024

**NYILATKOZAT A ZÁRÓDOLGOZATRÓL**

Alulírott *..........................................................................* (név) tanuló

**kijelentem, hogy**

*......................................................................................* című záródolgozatomat (nyomtatott és elektronikus formában) a Bajai SZC Türr István Technikumának pedagógusai és tanulói:

* + felhasználhatják (pl. hivatkozás alapjául, olvasótermi használatra) későbbi munkájukhoz a szerzői jogok tiszteletben tartása mellett).
  + nem használhatják fel (titoktartási nyilatkozat csatolása mellett).

Ugyanakkor kijelentem, hogy a záródolgozat ***saját munkám eredménye*.**

Baja, 20.......................

aláírás

1. Bevezetés 5

1.1. A téma ismertetése 5

1.2. Rendszerkövetelmények 5

1.3. Github elérés: 5

2. Felhasználói dokumentáció 6

2.1. Bejelentkezés és Regisztráció 6

2.2. Vásárlói szemmel 7

2.2.1. Kosár 8

2.2.2. Rendelések 9

2.3. Üzemeltetői szemmel 9

2.3.1. Általános 10

2.3.2. Pizzák kezelése 11

3. Fejlesztői dokumentáció 12

3.1. Felhasznált szoftverek, technológiák 12

3.1.1. Szoftverek 12

3.1.2. Technológiák 13

3.2. Adatbázis 14

3.2.1. Adatbázis táblák, mezők 14

3.2.2. Adatbázis kapcsolatok 16

3.3. Funkciók 16

3.3.1. Regisztráció és Belépés 17

3.3.1. Vásárlás 19

4. Továbbfejlesztési lehetőségek 23

5. Ábrajegyzék 25

# Bevezetés

A pizzéria alkalmazást választottam projektmunkának, mert szerettem volna egy olyan témát, amelyben számos kihívást találhatok. Egy ilyen alkalmazás fejlesztése sokféle funkciót és részletet igényel, például felhasználói regisztrációt és bejelentkezést, a pizzák rendelési folyamatát, egyéni összeállítási lehetőségeket, valamint a fizetési folyamatok kezelését. Emellett fontos szempont volt, hogy a téma széles körben releváns legyen, hiszen sok ember rendszeresen rendel pizzát. Úgy gondoltam, hogy az alkalmazásfejlesztés során számos különböző területen, például programozásban és tervezésben is tudok fejlődni, valamint valós gyakorlatot szerezhetek. Összességében úgy éreztem, hogy a pizzéria alkalmazás egy izgalmas és tanulságos projekt lesz számomra.

## A téma ismertetése

Manapság egyre elterjedtebb a házhozszállítás. Az időhiány és a kényelem miatt egyre többen veszik igénybe ezt a szolgáltatást, és a technológia fejlődésével ez a trend csak erősödni látszik. Külföldön és hazánkban, különösen nagyobb városokban már lehetőség van hipermarketekből is rendelni, és csak a kiszállításkor kell az üzletlánc alkalmazottjánál fizetni. Az interneten pár kattintással leadott rendeléseink „pár perc” múlva már a bejárati ajtónkon kopogtathatnak. Az étel házhozszállítása is rohamosan növekszik, mivel ma már a kínálat szinte végtelen. A vendéglátás területén is jelentős fejlődés történt, és a házhozszállítás nagy divat lett. Különösen vonzó lehet azok számára, akik speciális étrendet követnek vagy ételintoleranciával küzdenek, hiszen számos diétás és alternatív fogás közül válogathatnak.Az éttermi vacsorázásnak számos hátránya lehet, mint például az anyagi terhek, az időigény és az otthoni készülődés. A házhozszállítás viszont sok előnnyel jár, hiszen a várakozás alatt hasznosan vagy szórakozva tölthetjük az időt, és a kényelmi szempontok is kiemelkedőek.

## Rendszerkövetelmények

* Windows 7 vagy újabb
* minimum 1 GHz-es processzor
* minimum 512 MB RAM

## Github elérés:

https://github.com/Kay65/PIZZERIA\_RENDELES\_HG.git

# Felhasználói dokumentáció

A felhasználói dokumentáció keretein belül részletesen kívánom bemutatni mind a vásárlók, mind pedig az adminisztrátorok számára az alkalmazás működését és funkcionalitását. Célom, hogy minden egyes pont alatt átfogó és könnyen érthető útmutatást nyújtsak a felhasználóknak arról, hogyan használják a programot az egyes szerepkörökből kiindulva. A vásárlói szempontból kiemelten tárgyalom majd a rendelési folyamatot, a termékek böngészését, a kosár kezelését és a fizetési lehetőségeket, míg az adminisztrátori szempontból részletesen ismertetem az adminisztrációs felület funkcióit, az ügyfél- és rendeléskezelést, valamint az adatok kezelését és karbantartását. A dokumentáció célja, hogy a felhasználók teljes körűen megismerjék az alkalmazást, és hatékonyan tudják azt használni a mindennapi tevékenységeik során, legyen szó vásárlásról vagy az üzleti működés irányításáról.

## Bejelentkezés és Regisztráció

A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás

1. ábra – Bejelentkezés és Regisztráció (saját szerkesztésű ábra) [1]

Az alkalmazás megnyitásakor az első megjelenő képernyő a belépési ablak. Az alkalmazás fő funkciói eléréséhez elengedhetetlen a bejelentkezés. Ehhez szükséges a felhasználó E-mail címének és jelszavának megadása. Amennyiben még nem rendelkezünk felhasználói fiókkal, lehetőség van regisztrációra is, melyet a regisztrációs részen keresztül tehetünk meg. Itt szükséges megadni az E-mail címet, valamint kétszeresen rögzíteni a jelszót.

Ha nem töltjük ki az összes mezőt vagy nem létező felhasználó adatait írjuk be akkor hibaüzentet kapunk.

## Vásárlói szemmel

A képen szöveg, elektronika, képernyőkép, képernyő látható

Automatikusan generált leírás

2. ábra – Főoldal (saját szerkesztésű ábra) [2]

Termékek kiválasztása és kosárba helyezése egyszerű folyamat, amelyet a bejelentkezést követően a főképernyőn végezhetünk el. Itt, egy áttekinthető felületen, lehetőségünk van böngészni a rendelkezésre álló pizzák között, és megadni, hogy mennyit szeretnénk belőlük. Amint kiválasztottuk a kívánt termékeket és megadtuk a mennyiséget, egyszerűen csak rá kell koppintanunk a "Kosárba" gombra, és az alkalmazás automatikusan hozzáadja a kiválasztott elemeket a kosárhoz. Ez a zökkenőmentes folyamat biztosítja, hogy gyorsan és hatékonyan tudjuk összeállítani a rendelésünket, anélkül, hogy felesleges kattintásokra vagy bonyolult menükre lenne szükségünk.

### Kosár

A képen szöveg, képernyőkép, képernyő, szoftver látható

Automatikusan generált leírás

3. ábra – Kosár (saját szerkesztésű ábra) [3]

A kosár oldal egy átfogó nézetet nyújt az összes olyan tételről, amelyet a vásárló a kosárba helyezett. Itt lehetőség van egyenként vagy akár egyszerre is eltávolítani őket a kosárból, csak egyszerűen koppintani kell a "Törlés a kosárból" vagy a "Kosár ürítése" lehetőségre. Ezenkívül az oldal összegzi az összes tétel árát is, így könnyen nyomon követhetjük, hogy mennyibe kerül a rendelésünk. Ha elégedettek vagyunk a kosár tartalmával, és készen állunk a vásárlás véglegesítésére, akkor csak egyszerűen meg kell nyomnunk a "Rendelés véglegesítése" gombot. Fontos megjegyezni, hogy ezzel a lépéssel a rendelés véglegesítésre kerül, és nem lesz több lehetőségünk módosítani a rendelést. Ezenkívül a kosár tartalma is automatikusan törlődik, így biztosítva a gördülékeny és hatékony vásárlási élményt.

### Rendelések

A képen szöveg, szoftver, számítógép, Számítógépes ikon látható

Automatikusan generált leírás

4. ábra – Rendelések (saját szerkesztésű ábra) [4]

Miután a rendelésünket véglegesítettük, a "Rendelések" menüpont alatt kereshetjük fel azt. Itt a rendszer egy 8 karakterből álló azonosítóval látja el, megjelenítve az összegző árat és tájékoztatást nyújt a várható kiszállítási időpontról. Ezenkívül a "Rendelés elemei" opciót választva lehetőségünk nyílik részletesen megtekinteni a rendelésbe tartozó egyes tételeket. Ez az opció segít abban, hogy áttekintsük és megbizonyosodjunk arról, hogy minden szükséges elemet megfelelően hozzáadtunk a rendeléshez.

## Üzemeltetői szemmel

**Az alapértelmezett belépési adatok: felhasználónév: Admin@Admin.hu, jelszó: Admin123.**

Ezekben a pontokban részletesen bemutatom, hogyan működik a program adminisztrációs része. Az oldal legfelső jogkörét az adminisztrátor birtokolja, aki lehet az üzlet vezetője, tulajdonosa, vagy más vezető beosztású alkalmazott.

### Általános

Amennyiben a bejelentkezési oldalon az adminisztrátori felhasználó adatait adjuk meg, nem a megszokott felhasználói felületet fogjuk látni, hanem az Admin felületet. Itt lehetőségünk van elérni a megszokott felhasználói felületet, vagy választhatunk az adminisztrációs lehetőségek közül is. Ez az opció egyfajta kaput nyit az adminisztrációs funkciókhoz, amelyek segítségével szélesebb körű beavatkozást és irányítást biztosíthatunk az alkalmazásban.

A képen szöveg, elektronika, képernyőkép, szoftver látható

Automatikusan generált leírás

5. ábra – Adminisztrátori felület [5]

### Pizzák kezelése

A pizzák kezelésére szolgáló oldalon számos lehetőség áll rendelkezésre:  
Pizzák megtekintése: Ezen az oldalon lehetőség van a már felvett pizzák listájának megtekintésére. Itt láthatók a pizzák nevei és egységárai.

Új pizza felvétele: Új pizza felvételéhez meg kell adni a pizza nevét és egységárát. Ezt követően a pizza automatikusan hozzáadódik a listához.

Meglévő pizza törlése: Ha egy már felvett pizzát törölni szeretnénk, ki kell választani azt a listából, majd rá kell kattintani a "Pizza Törlése" gombra. Ezt követően a pizza véglegesen eltávolításra kerül a rendszerből. A pizzák kezelésének ezen lehetőségei biztosítják, hogy a rendelkezésre álló pizzakínálat mindig naprakész és pontosan tükrözze a valós helyzetet. Így a felhasználók mindig a legfrissebb információkhoz férhetnek hozzá.

A képen szöveg, elektronika, képernyőkép, képernyő látható

Automatikusan generált leírás

6. ábra – Pizzák kezelése [5]

# Fejlesztői dokumentáció

Ebben a dokumentációs részben részletesen szeretném bemutatni a programot a fejlesztői szemszögből. A programot a Visual Studio fejlesztői környezetben készítettem el, és C# programozási nyelven írtam. A program különböző form-okat használ a nézetek megjelenítésére, amelyek segítségével a felhasználók interakcióba léphetnek az alkalmazással. A form-ok lehetővé teszik az adatok megjelenítését és kezelését, valamint az alkalmazás különböző funkcióinak elérését és használatát.

## Felhasznált szoftverek, technológiák

A kódolás kezdetén sokan egy egyszerű jegyzettömböt használnak. Azonban a programozás során a tudás megszerzése és fejlesztése sokkal fontosabb, mint a köré épült, igényesen designolt segédprogramok használata. Bár egy jegyzettömb használata lehetővé teszi a kódírás megkezdését, a kényelem és a hatékonyság szempontjából általában célszerűbb olyan profibb szoftvert választani, amely specifikus funkciókkal és eszközökkel rendelkezik a fejlesztés támogatására. Ezek a szoftverek segítik a fejlesztőt a kódolási folyamat során, és optimalizálják a munkafolyamatot, így időt és energiát takarítva meg

### Szoftverek

A mai szoftverekben található segédeszközök sokféle módon tudnak segíteni a fejlesztőknek. Nem csupán azáltal, hogy könnyebbé teszik a kódolást, hanem számos egyéb funkcióval is rendelkeznek, amelyek felgyorsítják és hatékonyabbá teszik a fejlesztési folyamatot. Ezek az eszközök például segítenek átlátni és strukturálni a kódokat, különféle színekkel jelölik az egyes kódrészleteket, és lehetővé teszik a gyorsbillentyűk használatát a szöveg formázásához és rendezéséhez. Emellett hibakereső és debug funkciókkal is rendelkeznek, amelyek segítségével könnyebben azonosíthatjuk és javíthatjuk a hibákat a kódunkban. Továbbá, ezek a szoftverek biztosítanak egy futtatókörnyezetet, amely lehetővé teszi a megírt kódok tesztelését és futtatását, hogy megbizonyosodjunk a helyes működésükről. Összességében ezek az eszközök jelentősen hozzájárulnak a fejlesztés hatékonyságához és eredményességéhez.

**Visual Studio**

A Visual Studio egy integrált fejlesztői környezet (IDE), amelyet elsősorban szoftverfejlesztők használnak különféle platformokon, például Windowson, macOS-en és Linuxon. A Microsoft által fejlesztett szoftver segítségével lehetőség van számos programozási nyelv, például C#, C++, JavaScript, Python, és még sok más nyelv fejlesztésére és szerkesztésére. A Visual Studio számos funkciót kínál, beleértve a kódszerkesztést, a hibakeresést, a verziókezelést, a tesztelést és a fordítást is, ami jelentősen megkönnyíti a fejlesztők munkáját és hozzájárul a hatékonyabb kódolási folyamatokhoz.

**XAMPP**

XAMPP egy nyílt forráskódú csomag, amely tartalmazza az Apache web szervert, a MySQL adatbázis-kezelőt, a PHP-t és más szükséges szoftvereket. A célja, hogy egyszerűsítse és gyorsítsa fel a webszerver és az adatbázis-kezelő beállítását és használatát, különösen fejlesztési vagy tesztelési környezetben.

### Technológiák

**MySQL (relációs adatbázis-kezelő)**

MySQL egy ingyenes és nyílt forráskódú relációs adatbázis-kezelő rendszer, amelyet a svéd cég, a MySQL AB fejlesztett ki, majd később a Sun Microsystems felvásárlása révén a Sun, majd az Oracle tulajdonába került. A MySQL adatbázis-kezelő nagy népszerűségnek örvend a fejlesztők körében, mivel könnyen telepíthető, egyszerűen használható, megbízható és skálázható.

A MySQL-t gyakran használják webes alkalmazások és más dinamikus webes tartalmak tárolására és kezelésére. Az adatbázis-kezelő támogatja az SQL (Structured Query Language) szabványt, és számos fejlett funkciót kínál, például tranzakciókezelést, indexelést, felhasználókezelést és sok más adatbázis-kezelési funkciót.

A MySQL azért is népszerű, mert széles körben támogatja különféle platformokat, például Linuxot, Windows-t és macOS-t. Emellett számos nyelvhez és környezethez elérhető klienskönyvtárakat és interfészeket is biztosít, amelyek lehetővé teszik az alkalmazások számára, hogy könnyen kommunikáljanak az adatbázissal.

**C#**

A C# egy modern, objektumorientált programozási nyelv, amelyet kifejezetten a .NET keretrendszerhez terveztek. A C# erőteljes nyelvi eszközökkel rendelkezik a különböző típusú alkalmazások fejlesztéséhez, beleértve a webes alkalmazásokat, asztali alkalmazásokat, adatbázis-kezelést, játékokat és még sok másat. A C# tiszta szintaxissal rendelkezik, ami segíti a fejlesztőket az olvasható, karbantartható kód írásában. Az erős típusosság, a kivételkezelés, az automatikus szemétgyűjtés és más fejlett funkciók teszik hatékony és megbízható választássá a szoftverfejlesztésben. A C# integrált a Microsoft Visual Studio fejlesztői környezettel, ami lehetővé teszi a gyors és hatékony fejlesztést, valamint a széleskörű eszközök és erőforrások használatát. A .NET C# alkalmas kezdőktől a tapasztalt fejlesztőkig mindenki számára, és népszerű választás a vállalati és az egyéni projektfejlesztésben is.

## Adatbázis

Az adatbázis létrehozásakor az utf8\_hungarian\_ci szöveges kódolást választottuk, mely tökéletesen kezeli a magyar ábécé karaktereit. Ez biztosítja, hogy az adatbázis megfelelően kezelje a magyar nyelvű adatokat. A táblák létrehozásakor fontos az előzetes tervezés alapján strukturáltan felépíteni azokat, figyelembe véve az oszlopokat és azok tulajdonságait. Ha nincs még kész tervünk, későbbiekben lehetőség van oszlopok hozzáadására vagy törlésére. Ajánlott néhány adatot felvenni a már teszt üzemmód alatt futó táblákba, hogy a programozás során könnyen ellenőrizhető legyen az adatbázis kapcsolatunk és tesztelhessük, hogy a kódunk helyesen működik-e. Ez segít a fejlesztés során a hibák korai felismerésében és javításában.

### Adatbázis táblák, mezők

**ID mezők:** Minden egyes táblának van egy elsődleges mezője, amely az azonosítója, és ezen keresztül hivatkozhatunk az adott rekord értékeire a munka során. Ezek az azonosító mezők általában nem láthatók a végfelhasználók számára. Az azonosító mezők általában integertípusú változókban tárolódnak, mivel ezek csak számértékeket tartalmaznak. A "Auto Increment" beállítást alkalmaztam ezekre a mezőkre, így az értékeket automatikusan generálja és növeli egy egységnyivel az előző rekord értékéhez képest. Ennek köszönhetően soha nem fog ugyanolyan azonosító előfordulni egy adott táblában. Ez a mező kiemelkedő fontossággal bír a táblák közötti kapcsolatok létrehozásánál és fenntartásánál.

**Megnevezés mezők**: A pizza táblához hozzá lett rendelve ez az azonosító mező. A termék megnevezésében a rövid, tömör név került elmentésre, ami lényegre törő módon írja le a pizza fajtáját vagy összetevőit. Amikor a vevő először találkozik a megrendelési felülettel, ez a kiemelt szövegrész jelenik meg először számára, ami lehetővé teszi számára, hogy könnyen és gyorsan azonosítsa és válassza ki a kívánt pizzát. Ez a rövid, de hatékony megnevezés segít abban, hogy a vásárló gyorsabban megtalálja a számára vonzó terméket a rendelési folyamat során.

**Ár mezők**: A pizza és a rendelés táblák mindkettőjében megtalálhatóak ezek a mezők, amelyek az egyes tételek árát tartalmazzák. A pizza táblában az egységárat rögzítik, míg a rendelés táblában az összár kerül rögzítésre. Ezek a mezők alapvető fontosságúak a rendszer működése szempontjából, hiszen az árak rögzítése és nyomon követése elengedhetetlen a rendelések feldolgozásához és az üzleti műveletekhez. A megfelelő árazás lehetővé teszi a vevők számára, hogy átláthatóan és pontosan lássák a rendeléseik végösszegét, miközben a vállalkozás számára is biztosítja a pontos bevételi adatokat és a termékekkel kapcsolatos költségek átláthatóságát. Ezáltal hozzájárul a hatékony és átlátható üzleti működéshez.

**E-mail és Jelszó mezők:** Az E-mail és Jelszó mezők az adatbázisban a Felhasználó táblában kerülnek alkalmazásra. Ezeknek a mezőknek kiemelt szerepük van a bejelentkezés és a regisztráció folyamatában. A bejelentkezés fázisában az e-mail cím és a jelszó mezőket használjuk fel azonosításhoz, ahol a rendszer ellenőrzi a megadott adatokat és összehasonlítja az adatbázisban tárolt információkkal. A regisztráció során ezeket a mezőket új felhasználói rekordok beillesztésére használjuk. Ebben az esetben a rendszer menti az új felhasználó által megadott e-mail címet és jelszót a Felhasználó táblába. Emellett ezek a mezők egyben egyedi kulcsként is funkcionálnak, amelyek segítségével elkerülhetjük az azonos e-mail címek vagy jelszavak megismétlődését az adatbázisban. Ezáltal garantálva van, hogy minden felhasználónak egyedi azonosítója van, és biztosítva van az adatok egyértelműsége és biztonsága a rendszerben.

### Adatbázis kapcsolatok

A képen vázlat, rajz, diagram, Vonalas grafika látható

Automatikusan generált leírás

7. ábra – ER modell (saját szerkesztésű ábra) [6]

Az adatbázisban rengeteg táblakapcsolatot hoztam létre, melyek összekapcsolása nélkül sok tábla nem tudná teljesíteni a kívánt funkciókat. Például, egy rendeléseket jegyző táblánál elengedhetetlen a felhasználók táblájának használata annak érdekében, hogy azonosíthassuk, melyik vásárló adta le a rendelést. Ebben az esetben a rendelés táblában létrehoztam egy UserID nevű mezőt, amelyet összekötöttem a rendelés azonosítójával, így egyértelműen láthatóvá válnak a megrendelő adatai, mivel a táblák kapcsolatban állnak. Minden egyes táblában az első sorban az azonosító mezőt hoztam létre, melyet ID néven azonosítottam. Ezek az ID-k lehetővé teszik a legkönnyebb hivatkozást az adatokra. Az ID-ket Auto Increment funkcióval hoztam létre, így az egyes sorok értékei automatikusan növekednek egy egységgel az előző adathoz képest. Ez egyfajta sorszámként funkcionál, garantálva, hogy két ugyanolyan szám sosem jelenik meg.

## Funkciók

A program többféle formból áll, amelyek különböző nézeteket jelenítenek meg és különböző interakciós lehetőségeket kínálnak a felhasználók számára. Ezek a form-ok általában különféle grafikus elemeket tartalmaznak, mint például szövegmezőket (textbox), listákat (listbox), legördülő menüket (combobox) és gombokat (button). A felhasználó által bevitt adatokat mindegyik ilyen inputmezőből változókba mentem a nevük alapján. Ezeket a változókat aztán a program felhasználja az SQL beillesztéseknél és lekérdezéseknél. Ez az adatok átvitelének módja a felhasználótól az adatbázisba és onnan vissza, ami lehetővé teszi a felhasználók által bevitt adatok tárolását, feldolgozását és kezelését a program által. Ez a struktúra lehetővé teszi a felhasználói interakciókat és az adatok hatékony kezelését a programban.

### Regisztráció és Belépés

Az adatok felvétele többnyire megegyező attól függetlenül, hogy vásárlót regisztrálunk vagy alkalmazottat. Az űrlap általános adatokra vonatkozó rubrikák kitöltése után az adatbázisba létrehozzuk a fiókot és bejelentkezésre kész. Ezzel egy időben létrehozzuk a SESSION változót mely végigkísér minket böngészés közben csak az unset paranccsal inaktiválódik. A két eljárás hasonló. Amennyiben az oldal a módosítás szerepét töltötte be, lekérdeztem az adatbázisból a jelenlegi adatokat és feltöltöttem velük a mezőket, így ha csak egy adatot szeretnénk módosítani, akkor a többihez nem kell hozzányúlni.

**Regisztráció**

A regisztrációs folyamat egy viszonylag rövid űrlapból áll, amelyben csak az e-mail címet és a jelszót kell kétszer megadni. Ha a vásárló valamelyik mezőt üresen hagyja, akkor az if feltételvizsgálat egy hibaüzenetet generál, és a program megáll. Ez a funkció biztosítja, hogy a felhasználó minden kötelező mezőt kitöltsön, mielőtt a regisztrációt befejezné, és megakadályozza a hibás vagy hiányos adatok bekerülését az adatbázisba. A felhasználói interakció ezen a ponton teljesen automatizált, hogy egyszerűbbé és hatékonyabbá tegye a regisztrációs folyamatot, valamint, hogy biztosítsa az adatok pontosságát és teljességét a későbbi használat során.

|  |
| --- |
| if (Email.Text != "" && Password.Text != "")  {  ...  }  else  {  MessageBox.Show("Tölts ki minden mezőt!");  } |

A következő ellenőrzés, amit végzek, az a két jelszó egyezőségének vizsgálata. Amennyiben a felhasználó által megadott két jelszó nem egyezik meg, ezt a felhasználónak egy üzenetablak segítségével jelzem. Ez a funkció különösen fontos a regisztrációs folyamat során, mivel a jelszók egyezőségének ellenőrzése biztosítja, hogy a felhasználó helyesen adta meg a jelszavakat, és megelőzi a helytelen vagy eltérő jelszavak bekerülését az adatbázisba. A MessageBox.Show függvény használata lehetővé teszi a program számára, hogy egy érthető és szemléletes üzenetet jelenítsen meg a felhasználónak, amely segít neki az esetleges hibák vagy téves adatbevitel felismerésében és azok korrigálásában. Ezáltal javítja a felhasználói élményt és növeli a regisztrációs folyamat hatékonyságát és sikeres lezárásának esélyeit.

|  |
| --- |
| if (PassReg.Text == PassRegRep.Text)  {  ...  }  else  {  MessageBox.Show("A két jelszó nem egyezik!");  } |

Az e-mail cím ellenőrzése nem szükséges a felhasználó által megadott adatoknál, mivel az adatbázis maga kezeli az e-mail címek egyediségét. Ha a felhasználó egy már meglévő e-mail címet próbál megadni, az adatbázis automatikusan érzékeli ezt, és Hibaüzenetet dob ki. Amennyiben a felhasználó minden egyéb szűrőn és ellenőrzésen sikeresen átesett, azaz a jelszavak egyeznek, a szükséges mezők ki vannak töltve, és a felhasználói adatok megfelelnek a követelményeknek, akkor az adatbázisba az értékek sikeresen bekerülnek. Ezzel együtt a felhasználói fiók elkészült és készen áll a használatra. Ez a folyamat gyors és hatékony, mivel minimalizálja a felhasználó beavatkozását az adatok ellenőrzésében, és automatizálja a regisztrációs folyamatot, biztosítva a felhasználók számára a könnyű és zökkenőmentes regisztrációt és azonnali fiókelérést.

|  |
| --- |
| string email = Convert.ToString(EmailReg.Text);  string password = Convert.ToString(PassReg.Text);  string updateQuery = $"INSERT INTO `felhasználó` (`Email`, `Jelszó`, `Admin`) VALUES ('{email}','{password}','0')";  MySqlConnection databaseConnection = new MySqlConnection(connStr);  MySqlCommand commandDatabase = new MySqlCommand(updateQuery, databaseConnection);  commandDatabase.CommandTimeout = 60;  MySqlDataReader reader;  try  {  databaseConnection.Open();  reader = commandDatabase.ExecuteReader();  databaseConnection.Close();  MessageBox.Show("Sikeres Regisztráció!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  } |

**Belépés**

A belépés folyamata hasonló módon működik, mint a regisztrációé. A felhasználónak először meg kell adnia az e-mail címét és a jelszavát. Ezt követően a program ellenőrzi, hogy mindkét mező ki lett-e töltve. Ha mindkét mező kitöltése megtörtént, akkor két bevitt adatot ellenőrzünk: az e-mail címet és a jelszót. Amennyiben az e-mail cím nem létezik az adatbázisban, vagy a megadott jelszó hibás, a belépést megtagadjuk, és hibaüzenetet jelenítünk meg a felhasználónak.

Sikeres bejelentkezés esetén a program elmenti a felhasználó azonosítóját egy globális változóban annak érdekében, hogy a többi form is képes legyen elérni azt. Ezt követően a program ellenőrzi, hogy a felhasználó adminisztrátori jogosultságokkal rendelkezik-e vagy sem. Ha a felhasználó adminisztrátor, akkor megnyitja az Admin felület formot, ahol az adminisztrációs funkciókat érheti el. Ha a felhasználó nem rendelkezik adminisztrátori jogosultsággal, akkor a felhasználói felület formot nyitja meg, ahol a szokásos felhasználói műveleteket végezheti el. Ezáltal a belépés folyamatának végén a program dinamikusan vált a felhasználó jogosultságainak megfelelő felületre, biztosítva ezzel a felhasználói élményt és a megfelelő hozzáférést a rendszer funkcionalitásaihoz.

|  |
| --- |
| string email = Convert.ToString(Email.Text);  string password = Convert.ToString(Password.Text);  string sql = $"SELECT `ID`,`Admin` FROM `felhasználó` WHERE `Email`='{email}' AND `Jelszó`='{password}'";  MySqlConnection conn = new MySqlConnection(connStr);  MySqlCommand cmd = new MySqlCommand(sql, conn);  MySqlDataReader rdr;  Dictionary<string, string> comboSource = new Dictionary<string, string>();  bool admin;  try  {  conn.Open();  object obj = cmd.ExecuteScalar();  if (Convert.ToInt32(obj) != 0)  {  rdr = cmd.ExecuteReader();  rdr.Read();  admin =Convert.ToBoolean(rdr[1]);  if (admin == true)  {  Form4 Admin = new Form4();  Admin.UserID = Convert.ToInt32(rdr[0]);  Admin.ShowDialog();  }  else  {  Main main = new Main();  main.UserID = Convert.ToInt32(rdr[0]);  main.ShowDialog();  } |

### Vásárlás

A vásárlás folyamatának kezdeti lépése a kosárba gyűjtés. Ebben a szakaszban a vásárló kiválasztja azokat a termékeket, amelyeket szeretne megvásárolni, és megadja a kívánt mennyiséget. Fontos megjegyezni, hogy a program itt is végzi az adatbevitel ellenőrzését annak érdekében, hogy minden mezőt megfelelően töltöttek-e ki. Emellett figyelembe veszi, hogy a megadott mennyiség ne lehessen 0 vagy negatív értékű. Ez a lépés garantálja a vásárlás során megrendelt termékek pontos és megfelelő rögzítését, valamint segít megakadályozni a hibás vagy érvénytelen adatokat a rendszerben. A mennyiség negatív vagy nulla értékű megakadályozása biztosítja, hogy a vásárlók csak valóságos és elfogadható mennyiségeket adjanak meg a termékekből, ezáltal elősegítve a vásárlási folyamat zökkenőmentességét és a rendelési pontosságot.

|  |
| --- |
| if (comboBox1.SelectedItem == null || textBox1.Text == "")  {  MessageBox.Show("Tölts ki minden mezőt!");  }  else  {  if (Convert.ToInt32(textBox1.Text) <= 0)  {  MessageBox.Show("A mennyiség nem lehet 0 vagy negatív");  }  else  {  ...  } |

Ezután, miután a vásárló befejezte a választást, a "Kosárba" gombra kattintva az összes releváns adat, mint például a termékek nevei és a kiválasztott mennyiségek, bekerülnek a kosár nevű táblába az adatbázisban. Ez a lépés jelenti a vásárló által kiválasztott termékek előzetes összegyűjtését és tárolását a rendszerben, hogy aztán a fizetési és teljesítési folyamat során könnyen hozzáférhetők legyenek. A kosárba helyezés után a vásárló tovább léphet a fizetési és rendelési folyamatokhoz, és a kiválasztott termékeket ténylegesen megvásárolhatja. Ez az előkészítő lépés lehetővé teszi a vásárlók számára, hogy könnyedén kezeljék és módosítsák a kosárban lévő elemeket, mielőtt véglegesítik a vásárlást.

|  |
| --- |
| string pizza = Convert.ToString(comboBox1.SelectedValue);  int db = Convert.ToInt32(textBox1.Text);  string updateQuery = $"INSERT INTO `kosár`(`PizzaID`, `Mennyiség`, `UserID`) VALUES ('{pizza}','{db}',{UserID})";  MySqlConnection databaseConnection = new MySqlConnection(connStr);  MySqlCommand commandDatabase = new MySqlCommand(updateQuery, databaseConnection);  commandDatabase.CommandTimeout = 60;  MySqlDataReader reader;  try  {  databaseConnection.Open();  reader = commandDatabase.ExecuteReader();  databaseConnection.Close(); |

A következő lépésben, a kosár megjelenítő felületen a felhasználó megtekintheti az előzőleg tárolt elemeket, és ha nem elégedett velük, lehetősége van azok törlésére is. Ehhez a program egy olyan SQL parancsot használ, amely törli a kiválasztott elemeket az adatbázisból. Ez a folyamat lehetővé teszi a vásárlónak, hogy áttekintse a kosárban lévő termékeket, és szükség esetén módosítsa vagy törölje azokat, hogy a végleges rendelés megfeleljen az igényeinek és elvárásainak. A törlés funkció segítségével a vásárlók rugalmasan kezelhetik a kosár tartalmát, hogy az pontosan tükrözze az aktuális vásárlási szándékaikat és igényeiket. Ezáltal javul a vásárlói élmény és nő az elégedettség a vásárlók részéről.

|  |
| --- |
| string updateQuery = $"DELETE FROM `kosár` where `UserID`={UserID}";  MySqlConnection databaseConnection = new MySqlConnection(connStr);  MySqlCommand commandDatabase = new MySqlCommand(updateQuery, databaseConnection);  commandDatabase.CommandTimeout = 60;  MySqlDataReader reader;  try  {  databaseConnection.Open();  reader = commandDatabase.ExecuteReader();  databaseConnection.Close();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  } |

Ha a felhasználó elégedett a kosárban található adatokkal, lehetősége van véglegesíteni a rendelést. Ebben az esetben a program létrehoz egy új rekordot a rendelések táblában, amelyet ellát egy 8 karakteres véletlenszerűen generált kóddal, így biztosítva az egyedi azonosítást és nyomon követhetőséget. Ezenkívül a program áthelyezi a kosárban lévő adatokat a rendeléselemek táblába, és ezeket a rendelés azonosítójával látja el. Ezáltal a rendeléshez tartozó összes termék egyértelműen azonosíthatóvá válik a rendszerben. Végül a program eltávolítja ezeket az adatokat a kosár táblából, hogy a felhasználó kosara üres legyen a további vásárlásokhoz. Ez a folyamat biztosítja a vásárlási tranzakciók pontos és megbízható rögzítését, valamint megkönnyíti a felhasználók számára az áttekinthetőséget és a rendelések kezelését a rendszerben.

|  |
| --- |
| var chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";  var stringChars = new char[8];  var random = new Random();  for (int i = 0; i < stringChars.Length; i++)  {  stringChars[i] = chars[random.Next(chars.Length)];  }  var finalString = new String(stringChars);  string kiszállítás = DateTime.Now.AddHours(2).ToString("yyyy-MM-dd hh:mm:ss");  int oszar = osszar();  MessageBox.Show(Convert.ToString(oszar));  string updateQuery = $"INSERT INTO `rendelés`(`Állapot`, `Várható kiszállitás`,`UserID`, `Osszár`, `kód`) VALUES ('felvéve','{kiszállítás}',{UserID},{oszar},'{finalString}'); ";  MySqlConnection databaseConnection = new MySqlConnection(connStr);  MySqlCommand commandDatabase = new MySqlCommand(updateQuery, databaseConnection);  commandDatabase.CommandTimeout = 60;  MySqlDataReader reader;  try  {  databaseConnection.Open();  reader = commandDatabase.ExecuteReader();  updateQuery = $"INSERT INTO `rendeleselem` (`RendelésID`, `PizzaID`, `Menyiség`) SELECT rendelés.ID, kosár.PizzaID, kosár.Mennyiség FROM rendelés JOIN kosár ON rendelés.kód = '{finalString}' AND kosár.UserID = '{UserID}'; ";  commandDatabase = new MySqlCommand(updateQuery, databaseConnection);  databaseConnection.Close();  try  {  databaseConnection.Open();  reader = commandDatabase.ExecuteReader();  try  {  torles();  MessageBox.Show("Rendelés sikeresen rögzítve!");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  databaseConnection.Close();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  databaseConnection.Close();  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  } |

# Továbbfejlesztési lehetőségek

**Kalória, ételintolerancia**

A továbbfejlesztés során fontolóra veszem az étel alapanyagok glikémiás indexének megjelenítését is. Ez azt jelentené, hogy a vevő számára részletes információkat nyújtunk az általa választott termék szénhidrát, zsír és fehérje tartalmáról, valamint megjelenítjük az étel alapanyagok glikémiás indexét is. Ennek segítségével a vendég könnyebben dönthet arról, hogy az adott étel mennyire illeszkedik az aktuális diétájába vagy étrendjébe.

Továbbá, fontolgatom az ételintoleranciában szenvedő ügyfelek számára kifejlesztett megoldást is. Ez magában foglalná az ételintoleranciával kapcsolatos érzékenységek, például a glutén vagy laktóz intolerancia, megjelenítését a termékek részletes leírásában. Ezzel lehetővé tennénk számukra, hogy könnyebben válasszanak olyan ételeket, amelyek megfelelnek az ételintoleranciájuknak.

Ezen felül, nagyobb egységeknél alternatív megoldásokat kínálnánk az étel olyan elkészítésére, amely garantálja, hogy nem tartalmazza az adott alapanyagokat, így lehetővé téve az ételintoleranciával küzdő vendégek számára is biztonságos ételválasztást. Ezáltal a vendégek számára lehetőség nyílik arra, hogy tudatosabban és biztonságosabban válasszanak ételt, figyelembe véve az egészségügyi szempontjaikat és ételintoleranciájukat is.

**Személyre szabható pizzák**

Távolabbi terveim között szerepel a program továbbfejlesztése, melyben kiemelt figyelmet fordítok a grafikus rendelés összeállító fejlesztésére. Ennek célja, hogy olyan élményt nyújtsak a vendégeknek, amelyben aktívan részt vehetnek az étel elkészítésében. A grafikus rendelés összeállító segítségével azt szeretném elérni, hogy a vendégeknek legyen lehetőségük részletesen testre szabni az általuk rendelt ételt, és az elkészítés folyamatában érezhessék magukat részeseinek. Ezáltal célom, hogy az ételválasztás ne csupán egy egyszerű feladat legyen, hanem interaktív élménnyé váljon, amelyben a vendégek személyes ízlésüknek megfelelően alakíthatják az ételüket, és egyfajta alkotóként érezzék magukat az étel elkészítésében. Ez a funkcionalitás új dimenziót adna a vásárlói élménynek, hiszen lehetővé teszi számukra, hogy aktívan részt vegyenek a rendelési folyamatban, és személyre szabottan alakítsák ki az ételüket, miközben mélyebben belemerülnek az ételek világába és kreativitásukat is kifejezhetik.

**További Admin funkciók**

A jelenlegi verzióban az adminisztrátornak korlátozottak a lehetőségei és funkciói. Azonban szándékozom ezen változtatni azzal, hogy további funkciókat vezetek be az adminisztrátor számára. Ezek közé tartozhatnak például a felhasználók kezelése, azaz a felhasználói fiókok létrehozása, módosítása vagy törlése, valamint a rendelések kezelése, ami magában foglalhatja a rendelések nyomon követését, módosítását vagy éppen törlését. Ezek az új funkciók lehetővé teszik az adminisztrátornak, hogy teljeskörűen és hatékonyan kezelje a program működését és ügyvitelét. A bővített adminisztrációs lehetőségek segítenek a felhasználók jobb kiszolgálásában és az üzleti folyamatok hatékonyabb irányításában, ami hozzájárul a rendszer funkcionalitásának és hatékonyságának javításához.

# Ábrajegyzék

[1. ábra – Bejelentkezés és Regisztráció (saját szerkesztésű ábra) [1] 6](#_Toc165732079)

[2. ábra – Főoldal (saját szerkesztésű ábra) [2] 7](#_Toc165732080)

[3. ábra – Kosár (saját szerkesztésű ábra) [3] 8](#_Toc165732081)

[4. ábra – Rendelések (saját szerkesztésű ábra) [4] 9](#_Toc165732082)

[5. ábra – Adminisztrátori felület [5] 10](#_Toc165732083)

[6. ábra – Pizzák kezelése [5] 11](#_Toc165732084)

[7. ábra – ER modell (saját szerkesztésű ábra) [6] 16](#_Toc165732085)