

Introcafe

dokumentáció

Készítette:

Lukovics Marcell, Czézár Krisztián

**Tartalomjegyzék**

**Bevezetés**

1.1. Projekt célja

1.2. Felhasználói igények és célcsoport

**Adatbázis**

2.1. Adatbázisban áttekintése

2.2. Táblák és adattípusok

**Backend fejlesztés**

5.1. Használt technológiák és keretrendszerek

5.2. Alapvető API végpontok

5.2.1. Menü lekérdezése

5.2.2. Rendelés kezelése

5.2.3. Felhasználói jogosultságok

5.3. Biztonsági megoldások

**Frontend fejlesztés**

6.1. Használt technológiák és keretrendszerek

6.2. Interaktív elemek és funkciók

6.3. Reszponzív design megvalósítása

**Jövőbeli fejlesztési lehetőségek** 9.1. Új funkciók javaslatai6

**Bevezetés**

**1.1.: Projekt célja**

A Bomboclat Kft. felkeresett minket weboldalunkon feltűntetett e-mail címünkön keresztül egy kávézó szoftveres megvalósításának igényével. Skibidi János és Gyatt Anna, a cég tulajdonosai egy házaspár, kik friss cégükkel szeretnének belevágni egy vendéglátói vállalkozásba, és megalkotni egy minimalista, relaxáló és munkaorientált kávézót a kornak megfelelő modern megoldásokkal.

**1.2.: Felhasználói igények és célcsoport**

Egy csöndes relaxáció, illetve munkaorientált kávézó megalkotása. A kávézó célja, hogy a legkevésbé szociális ember is nyugodtan fogyaszthasson ezen a helyen, és elmenekülhessen a világ zajaitól egy fárasztó nap után. A kávézó tematikájának legfontosabb eleme a rendelés, illetve kiszolgálás maximális automatizálása az emberi munkaerő leváltása nélkül. Ezen cél eléréséhez elengedhetetlen a kontakt nélküli rendelésleadás megvalósítása, az üzlet belsejének tudatos felépítése és elrendezése, illetve a házirend szabályzatának megfontolt megfogalmazása.

**Adatbázis**

**2.1.: Adatbázis áttekintése**

A kávézó adatainak kezelésére „introcafe” néven hoztunk létre adatbázist, a cégtulajdonos igényeinek megfelelően. A megvalósításhoz a MySQL (MariaDB) adatkezelő nyelvet használtuk, a környezet futtatásához illetve az adattároláshoz a XAMPP program került használatra (<https://sourceforge.net/projects/xampp/>), illetve a táblák grafikus kezeléséhez a devart dbForge Studio Express (<https://www.devart.com/dbforge/mysql/studio/>) változatát használtuk.

Az adatbázisban tervezett elemek:

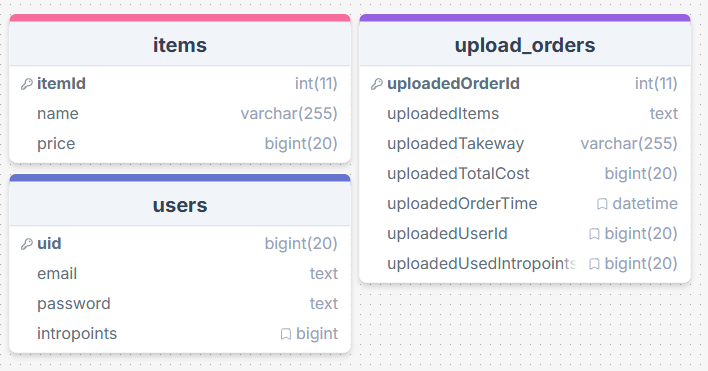
- A kávézó által forgalmazott termékek adatai: Adott termék azonosítója, az ahhoz tartozó terméknév, illetve az eladási ára.

- Az teljesítendő rendelések táblája: Tartalmazza az adott rendelés sorszámát, az sorszámhoz tartozó rendelt termékeket illetve azok mennyiségét, a fogyasztási típust (helyben fogyasztás/elvitel), a végösszeget, a rendelés rögzítésének idejét, amennyiben a rendelés leadása weboldalon keresztül történik, a felhasználó azonítóját, illetve a felhasznált intropontokat.

- A regisztrált felhasználók táblája: Tartalmazza a felhasználó azonosítóját, regisztrációkor megadott e-mail címét, illetve jelszavát, valamit a vásárlások után szerzett intropontjait.

**2.2.: Táblák és adattípusok**

A létrehozott adatbázis táblái, illetve oszlopai az alábbi ábrán láthatók:



Az „**items**” tábla tartalmazza:

- a termék azonosítóját, ezt manuálisan kell megadni termék felvételekor, típusa INT(11) (szám) értéke nem lehet üres (NOT NULL).

- a termék nevét, típusa VARCHAR(255) (szöveg), értéke nem lehet üres (NOT NULL).

- illetve a termék árát. típusa BIGINT(20), értéke nem lehet üres (NOT NULL).

Az „**upload\_orders”** tábla tartalmazza:

- a teljesítendő rendelés sorszámát, típusa INT(11) (szám), értéke nem lehet üres (NOT NULL), AUTO\_INCREMENT típusú, így értékét az SQL adja.

- a rendelt termékek listáját, típusa VARCHAR(255) (szöveg), értéke nem lehet üres (NOT NULL).

- a rendelés fogyasztási típusát (helyben/elvitel), típusa VARCHAR(255) (szöveg), értéke nem lehet üres (NOT NULL).

- a fizetendő végösszeget, típusa BIGINT(20) (szám), értéke nem lehet üres (NOT NULL).

- a rendelés rögzítési idejét, típusa DATETIME (dátum, idő), alapértelmezett értéke az adott pillanatbeli rendszeridő a CURRENT\_TIMESTAMP metódussal, értéke nem lehet üres (NOT NULL).

- a rendelést leadó vásárló azonosítója, értéke nem lehet üres (NOT NULL), alapértelmezett értéke 0.

- az adott felhasználó rendeléshez felhasznált intropontjai, értéke nem lehet üres (NOT NULL), alapértelmezett értéke 0.

A „users” tábla tartalmazza:

- a felhasználó azonosítóját, típusa BIGINT(20), értéke nem lehet üres (NOT NULL), AUTO\_INCREMENT típusú, így értékét az SQL adja.

- a felhasználó e-mail címét, típusa TEXT, értéke nem lehet üres (NOT NULL).

- a felhasználó jelszavát, típusa TEXT, értéke nem lehet üres (NOT NULL).

- a felhasználó hűségpontjait, típusa BIGINT(20), értéke nem lehet üres (NOT NULL), alapértelmezett értéke 0.

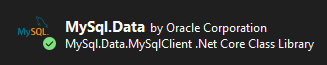
**Asztali alkalmazás**

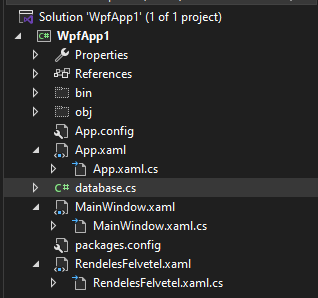
**5.1.: Asztali alkalmazás leírása**

A projekt ezen szegmense a kávézó dolgozóinak szánt felület. Használatával a foglalkoztatottak megtekinthetik az adatbázisban lévő teljesítendő rendeléseket, elkészítés után késznek jelölhetik őket, így törölve az adatbázisból, illetve abban az esetben, ha a vendég bármilyen oknál kifolyólag nem képes leadni rendelését a web felületen, abban az esetben a dolgozó manuálisan is fel tudja venni a rendelést.

**5.2.: Asztali alkalmazás szoftveres megvalósítása**

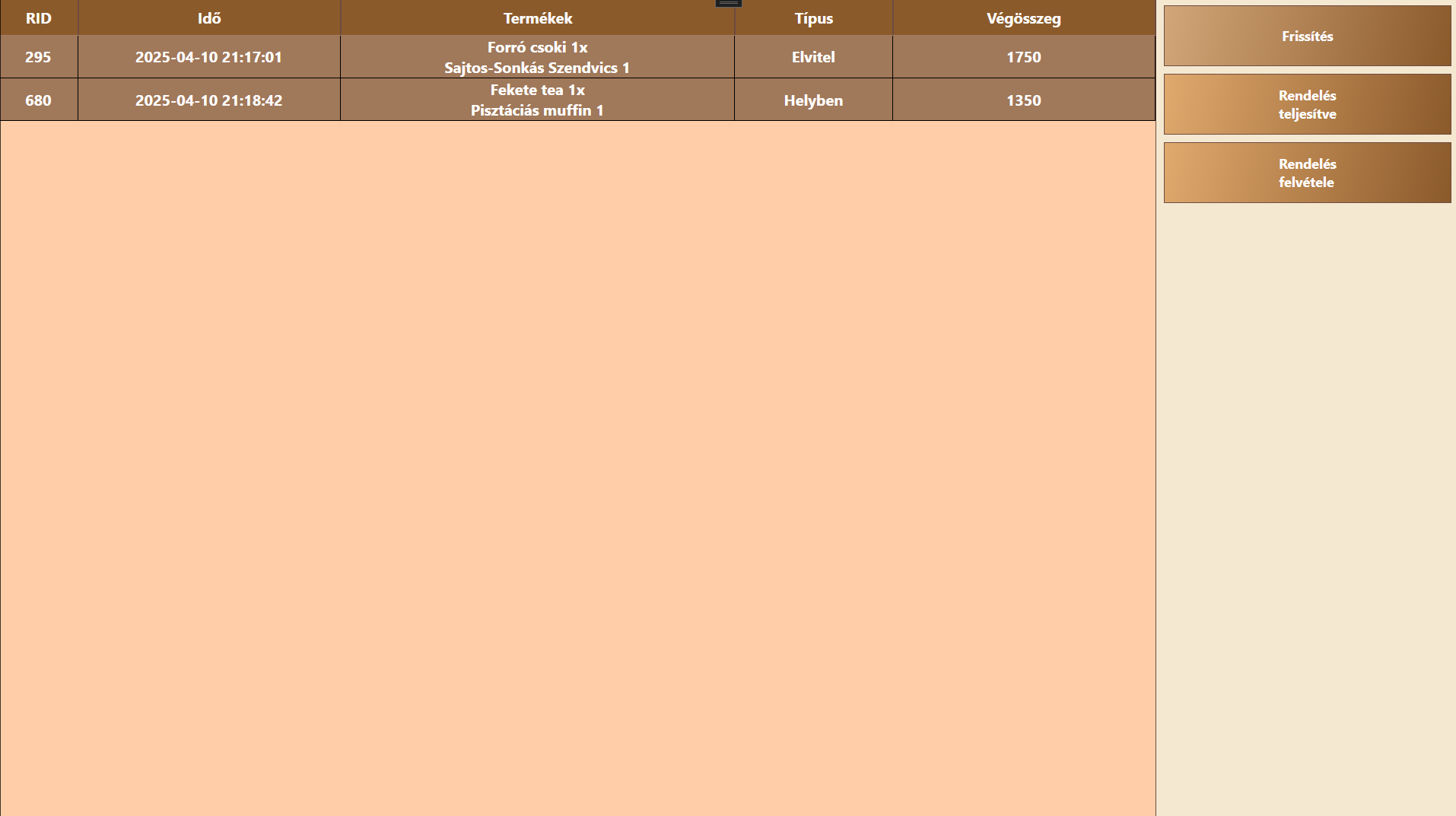
Az applikáció létrehozásához a Microsoft Visual Studio 2022 program ingyenes verzióját használtuk. Ez egy natív WPF Application, illetve C# és XAML programnyelveken lett megírva. A projekt két fő szegmensre bontható, a rendelés kezelő, illetve a rendelés felvétel felületre. Mindkét felülethez tartozik egy-egy dizájn fájl (XAML), ami a szoftver grafikus megjelenésének kódját tartalmazza, ezzel interaktál a felhasználó futtatás során, illetve egy-egy a XAML fájlhoz tartozó C# osztály, ami a mükődéshez szükséges függvényeket tartalmazza, mint például a gombok, szövegdobozok funkciói, stb. Ezen kívül a projekt része egy “database” nevezetű C# osztály is, ami az adatbázisból való lekérdezésekért/feltöltésekért felelős, így lehetővé téve a rendelések kezelését. Az applikáció teljes mértékben reszponzív, így bármilyen felbontású illetve méretű kijelzőn futtatjuk, a tartalma helyesen és olvashatóan fog megjelenni. Az adatbázis közti kommunikációhoz elengedhetetlen a MySQL.Data 9.2.0 bővítmény letöltése:



Az alábbi képen a projekt fájljai, illetve azok kapcsolatai láthatók:  


**5.3.: A rendelés kezelő:**

Az applikáció grafikus megjelenése az alábbi képen látható:

****

A felület érintőképernyős monitorokra lett elsősorban tervezve dizájn szerint. Az ablak két fő elemből áll, egy DataGrid objektumból, amiben a beérkezett rendelések kerülnek kilistázásra, illetve egy StackPanel objektumból amiben három gomb objektum található.

A DataGrid táblázat eleme a rendelés azonosítója, a felvétel időpontja (a rendelések eszerint vannak sorrendbe rakva, a legkorábban felvett lesz legfelül), a rendelt termékek, és azok mennyisége, a fogyasztás típusa, ami lehet elvitel vagy helyben fogyasztás, illetve a fizetendő végösszeg. Ezeket az adatokat az „introcafe” adatbázis „upload\_orders” táblájából kéri le a következő oszlopok segítségével: uploadedOrderId, uploadedOrderTime, uploadedItems, uploadedTakeway, uploadedTotalCost. A DataGrid tartalma automatikus frissül egy adatbázis lekérdezés segítségével, ami minden tizedik másodpercben fut le. Ha kattintással kijelölünk egy sort a DataGrid-ben, kékre változik a színe, és az adott rendelés lesz a kiválasztott elem. Az elem újboli megnyomásával, vagy egy másik rendelésre való kattintással vonhatjuk vissza a kijelölést, utóbbi esetlen az újonnan megnyomott rendelés lesz a kiválasztott.

A második rész egy StackPanel objektum, aminek 3 gomb típusú elem a tartalma. A „Frissítés” gomb a DataGrid manuális frissítésére szolgál, ha esetleg bármilyen okból nem működne az automatikus lekérdezés. Amennyiben egy rendelés ki van választva a DataGrid-ben, a rendelés teljesítése gomb megnyomásával törölhetjük mint a táblából, mint az adatbázisból. Ha a rendelés leadása weben keresztül történt, a rendelést leadó felhasználó intropontjaihoz hozzáadódik a teljesített rendelés végösszege osztva tízzel. A rendelés felvétele gomb megnyomásával új ablakban megnyílik a rendelés felvétele menü.

**5.4.: A rendelés felvétel ablak:**

Az applikáció grafikus megjelenése az alábbi képen látható:



Az ablak 3 részre sorolható:

- Bal szegmens:

- Egy DataGrid objektumból áll, ami az “introcafe” adatbázis “items” táblájából listázza ki az adatokat. Megjeleníti a termék azonosítóját, nevét, illetve árát. A rendelés kezelőben megismert kijelölő rendszer itt is érvényes.

- Középső szegmens:

- Tartalma egy label objektum, ami a “Keresés…” felirat megjelenítésére szolgál, további funkciója nincs.

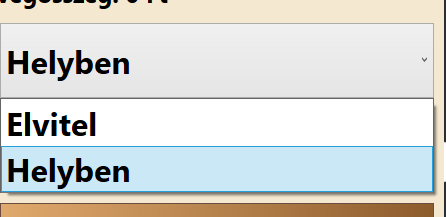
- Egy TextBox objektumból, ami a bal szegmensben való keresésre szolgál, a könnyebb használhatóság érdekében. Bele kattintással, és utána az abba való gépeléssel kereshetünk a táblázat “Név” oszlopában tárolt adatokra.

- Az utolsó objektum egy label, ami a rendeléshez eddig hozzáadott termékek listájának megjelenítése szolgál. Kiírja a termék nevét, illetve ha hozzáadás során érzékeli a program, hogy az adott termék már szerepel a listában, akkor szorzóként hozzáadja az adott termék mennyiségéhez. Feltöltéskor ez lesz az adatbázis uploadedItems mező értéke.

- Jobb szegmens:

- Tartalma egy végösszeg label, ami az eddig felvett termékek összesített árát tárolja. Tartalma minden hozzáadott termék után automatikusan frissül.

- Egy ComboBox objektum, ami a fogyasztás típusának kíválasztására szolgál. Rákattintással egy legördülő menü nyílik meg, ahol a kívánt típus megnyomásával választhatunk:



- Egy hozzáadás gomb, aminek megnyomásával adhatjuk hozzá a rendelt termékek listájába a DataGrid-ben aktív kijelölt elemet. Kijelölés hiányában a gomb nem funkcionális.

- Egy felvétel gomb, aminek megnyomásával feltölthetjük a rendelés adatait az adatbázis “upload\_orders” táblájába. Az elküldött elemek a következők:

- A rendelés azonosítója: Ez lesz az adatbázis uploadedOrderId mezőjének értéke.

- A rendelt elemek listáját, aminek tartalma lesz az adatbázis uploadItems mezőjének értéke.

- A fogyasztás típusát amit a ComboBox kiválaszott eleméből választ ki, ez lesz az uploadedTakeway mező értéke az adatbázisban.

- A fizetendő végösszeget, ez lesz az uploadedTotalCost mező értéke az adatbázisban.

-A MÉGSEM gomb lenullázza az összes beviteli mezőt és bezárja az ablakot, ezzel megszakítva az új rendelés felvételének folyamatát.

**5.5.: MainWindow.xaml.cs működési leírása:**

Változók és példányosítás:

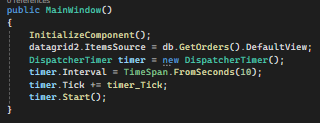


- db: Egy database osztály példánya, amely az adatbázis műveleteket végzi (pl. rendelés lekérdezés, törlés).

- elmentettid: A kiválasztott rendelés azonosítóját tárolja (kezdetben -1).

- sor **/** selectedRow: A DataGrid kiválasztott sorát tárolják különböző formában (DataRow és DataRowView).

Konstruktor („MainWindow()”):

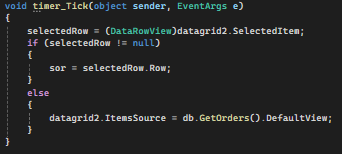


- Inicializálja a komponenst.

- Betölti a rendeléseket a datagrid2 nevű adatgridbe.

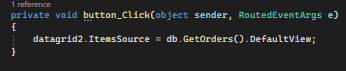
- Elindít egy 10 másodperces időzítőt (DispatcherTimer), amely automatikusan frissíti a rendeléslistát.

Időzítő eseménykezelő (timer\_Tick):



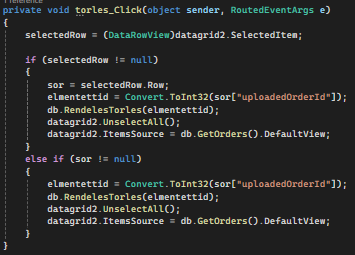
- Automatikusan ráfrissít a datagrid tartalmára, kivéve ha van aktív kiválasztott rendelés, ilyenkor a GetOrders() függvény nem fut le.

Frissítés gomb:



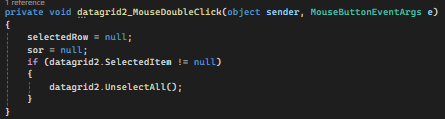
Manuálisan újratölti az adatokat a datagrid2 objektumba.

Törlés gomb:



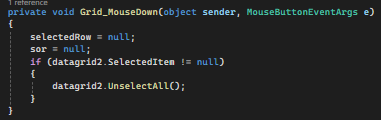
Ellenőrzi, hogy van-e kijelölt sor. Ha van, végrehajtja a törlést, ha nincs, ignorálja a gomb lenyomását.

Újboli kattintás a táblázatban:



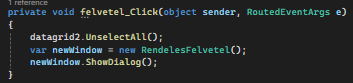
Törli az aktuális kiválasztást (selectedRow, sor), majd megszünteti a kijelölést a táblázatban.

Üres területre kattintás:



A duplakattintáshoz hasonlóan megszünteti a sor kiválasztását, ha a felhasználó a táblázaton kívül kattint.

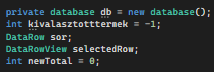
Új rendelés felvétele:



Megnyitja a rendelés felvétel ablakot, olyan módon, hogy ne lehessen interaktálni a rendelés kezelő ablakkal, amíg meg van nyitva.

**5.6.: RendelesFelvetel.xaml.cs működési leírása:**

Változók és példányosítás:



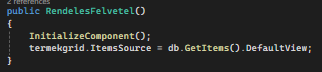
db: A database osztály példánya, amely az adatbázis műveleteket végzi.

kivalasztotttermek: Egy kijelölt termék azonosítója.

sor / selectedRow: A kiválasztott sor adatát tárolják.

newTotal: A rendelés aktuális végösszege.

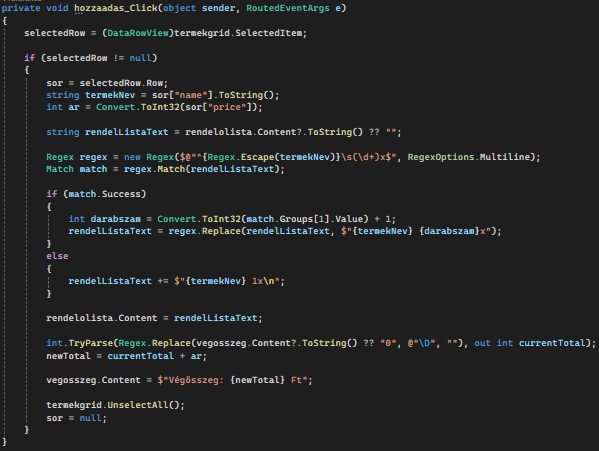
Konstruktor („RendelesFelvetel()”):



Inicializálja az ablak komponenseit.

Betölti a termékeket az adatbázisból, és megjeleníti azokat a termekgrid táblázatban.

Termék hozzáadása a listához:



Ellenőrzi, van-e kijelölt termék a táblázatban.

Ha van:  
 - Lekéri a termék nevét és árát.

- Megnézi, hogy a termék már szerepel-e a rendelés listájában.

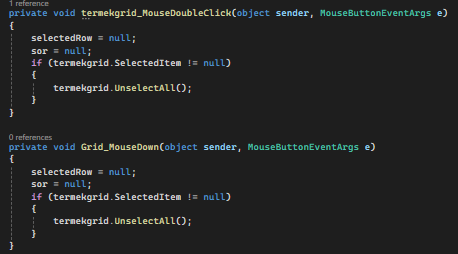
- Ha igen: növeli a darabszámot (pl. Kávé 1x -> Kávé 2x).

- Ha nem: újként hozzáadja a listához.

- Frissíti a végösszeg label-t.

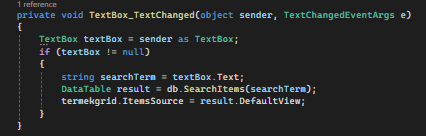
- Törli a kijelölést a táblázatban.

Kijelölés törlése:



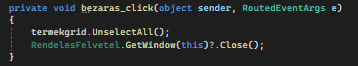
Újboli kattintás, vagy üres részre való kattintás esetén megszüntetik a kiválasztást a DataGrid-ben.

Keresés a termékek között:



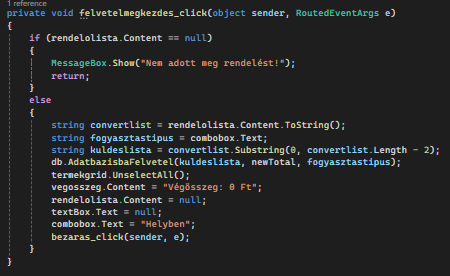
Amikor a keresés mező tartalma változik gépeléssel, meghívja az adatbázis kereső függvényét, majd ráfrissít a táblázatra.

Ablak bezárása:



Törli a kijelöléseket, nullázza az értékeket és bezárja az ablakot.

Rendelés leadása:



Ha üres a rendelő lista, figyelmeztet erre a program és kilép a függvényből.

Ha van tartalma:

- Összegyűjti a listát (rendelolista), a végösszeget és a kiválaszott fogyasztás típusát (comboBox)

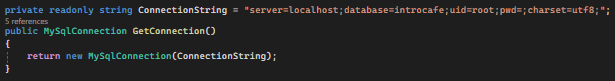
- Meghívja a database.cs AdatbazisbaFelvetel metódusát, amely elmenti a rendelést.

- Kiüríti a mezőket, visszaállítja az alapértékeket.

- Bezárja az ablakot.

**Database.cs működési leírása:**

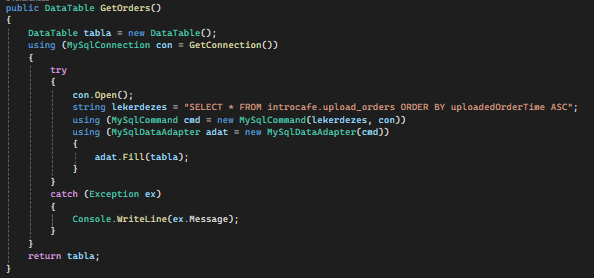
Adatbázis kapcsolat:



Beállítja az adatbázis-kapcsolat alapértelmezett paramétereit (szerver, adatbázisnév, felhasználó, karakterkódolás).

A GetConnection() metódus új kapcsolatot ad vissza a szerverhez.

Rendelések lekérdezése:



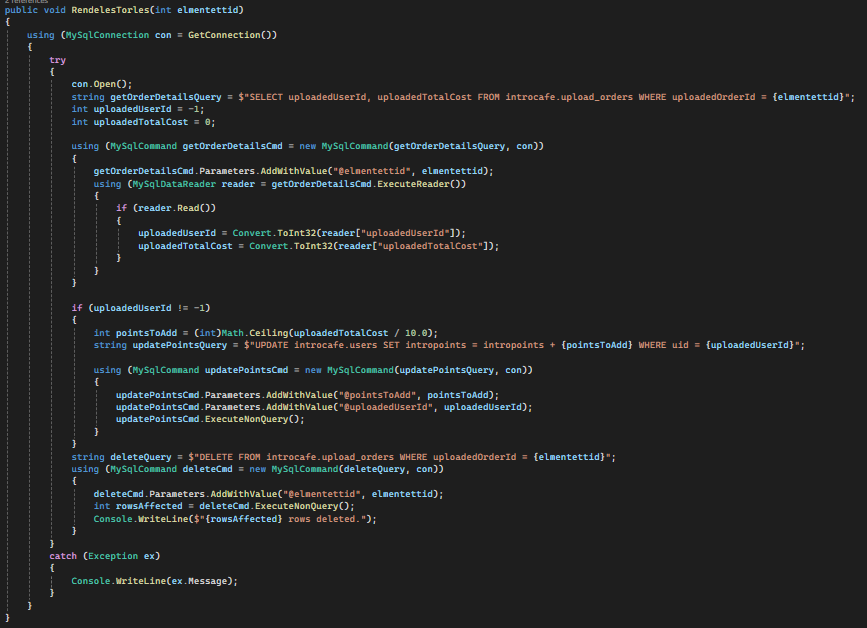
Lekérdezi az összes rendelést az upload\_orders táblából.

A lekérdezés az uploadedOrderTime mező szerint időrendbe rendezi az eredményeket.

Az eredményeket DataTable objektumba tölti be.

A „tabla” változót visszaküldi, ami a rendelések DataGrid értéke lesz.

Rendelés teljesítés:



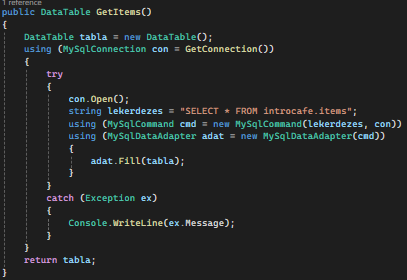
Lekérdezi a törlendő rendeléshez tartozó felhasználó azonosítóját és a rendelés összegét

Kiszámítja a jóváírható pontokat: végösszeg/10, felfelé kerekítve.

Frissíti az intropoints mezőt az adatbázisban.

Végül törli a rendelést az upload\_orders táblából a kijelölt sor azonosító értéke alapján.

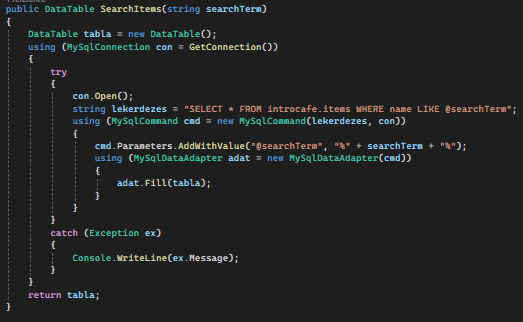
Termékek lekérdezése:



Lekérdezi az összes terméket az items táblából.

A találatokat a „tabla” objektumban visszaadja, ami a termékek DataGrid-ben fog megjelenni.

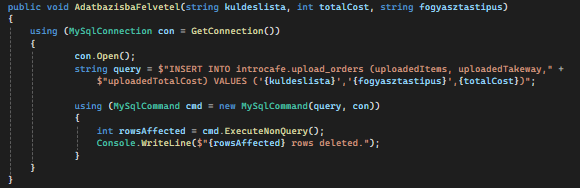
Termékek közötti keresés:



Lehetővé teszi, hogy a felhasználó keresést végezzen a termékek között névrészlet alapján.

A lekérdezés tartalmaz egy LIKE operátort (%keresett\_szöveg%), ami részleges egyezést keres a termék nevében.

Rendelés felvétele:



Egy új rendelést szúr be az adatbázis upload\_orders táblájába.

Három adatot ment:

- uploadedItems: a termékek listája szöveges formában.

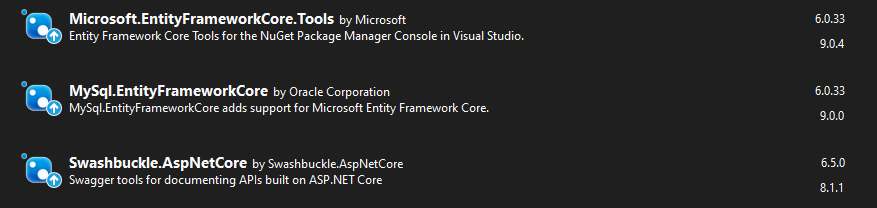
- uploadedTakeway: a fogyasztás típusa ("Helyben", "Elvitelre").

- uploadedTotalCost: a rendelés végösszege.

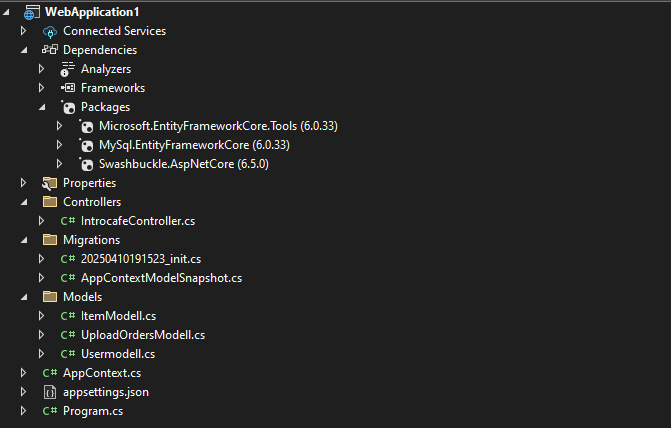
**Web API**

**6.1.: A Web API leírása:**

A projekt ezen részének feladata, hogy lehetővé tegye a kávézó weboldala és az adatbázis közötti kommunikációt. A Web API a Microsoft Visual Studio 2022 programban lett megírva. Létrehozásához a .NET keretrendszer 6.0-ás verzióját használtuk. A projekt létrehozása után elengedhetetlen az alábbi NuGet csomagok letöltése:



A projekt fájljai és azoknak kapcsolatai az alábbi képen láthatók:



**Program.cs:**

A kapcsolat létrehozása a feladata, illetve ezen belül állítjuk be a Swagger grafikus környezet feladatát, illetve a hozzáférési engedélyeket Cors szabványok segítségévél.

**appsettings.json:**

Alapértelmezetten létrehozott fájl, egyedüli módosítás benne, az alapértelmezett adatbázis kapcsolat deklarálása.

**AppContext.cs:**

Feladata az adatbázis beállítása az általunk megadott tábla modellek, illetve az adatbázis elérési útvonalának megadásával.

**Modells mappa:**

Az adatbázisban szereplő táblák, illetve azok adattípusait hozzuk létre benne, hogy adatmegosztáskor a megfelelő értékek kerüljenek megfelelő típussal feltöltésre. A mappában található három C# osztály, a táblákat reprezentálja:

- ItemModell.cs = items tábla

- UploadOrdersModell = upload\_orders tábla

- Usermodell = users tábla

Ezeken a fájlokon belül adjuk meg az oszlopok alapértelmezett értékeit is, illetve az elsődleges kulcsokat.

**IntrocafeController.cs:**

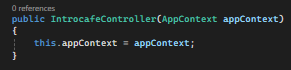
A müködéshez szükséges HttpRequesteket tartalmazza, ilyen például a lekérés, feltöltés, törlés metódus. Bármilyen kommunikáció az adatbázis és a weboldal között ezen keresztül történik. Függvényeket tárol, amik szükség esetén meghívhatóak. Esetünkben egy Get kérést tartalmaz, ami csupán tesztelés céljából létezik a jövőre való tekintettel, illetve egy Post kérést, ami a felhasználó által leadott rendelést küldi el az adatbázisnak.

**Migrations mappa:**

Az adatbázis létrehozásához szükséges fájlokat tartalmazza, további szerepe nincs.

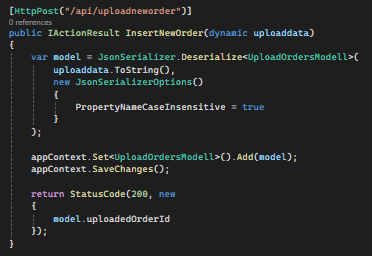
**6.2.: Controllerek leírása:**

Konstruktor:



A kontroller a DbContext példányát (AppContext) injektálja, amely az adatbázishoz való hozzáférést biztosítja.

[HttpPost("/api/uploadneworder")]:

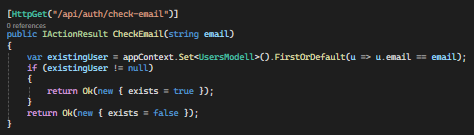


Egy új rendelés beszúrására szolgál az adatbázisba. JSON típusú adatot vár.

Példa értékek:  


Tartalma a rendelést leadó felhasználó azonosítója, a fogyasztás típusa, a megrendelt termékek, a végösszeg, illetve, ha történt ilyen, a rendelés végösszegét csökkentő felhasznált intropontok.

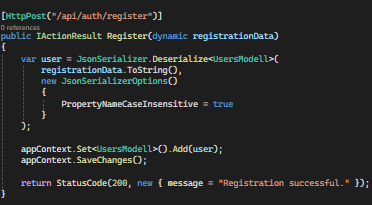
[HttpGet("/api/auth/check-email")]:



Ellenőrzi, hogy a regisztrálni kívánt e-mail cím szerepel-e már az adatbáziban.

Példa válasz:  
{ "exists": true }

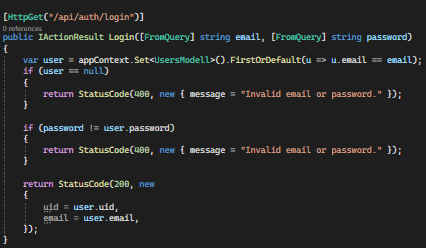
[HttpPost("/api/auth/register")]:



Új felhasználót regisztrál az adatbázisba. JSON típusú adatot vár.

Példa adatok:  
{  
 „email”: „teszt@teszt.teszt”,  
 „password”: „jelszo”  
}

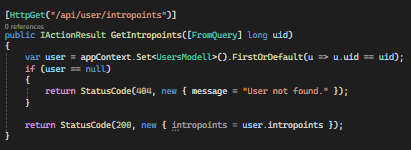
[HttpGet("/api/auth/login")]:



Kapott email és jelszó adat alapján bejelentkezteti a felhasználót.

Példa válasz:  
{  
 "uid": 1,  
 "email": "teszt@teszt.teszt"  
}

[HttpGet("/api/user/intropoints")]:

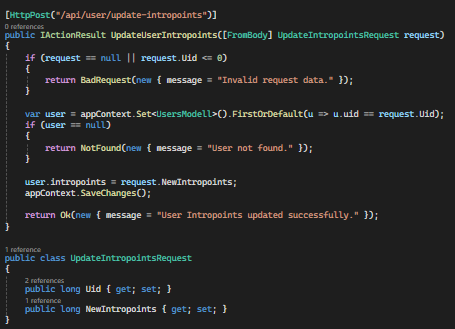


Visszaadja az adott felhasználó intropoints értékét.

Példa válasz:

{  
 "intropoints": 40  
}

[HttpPost("/api/user/update-intropoints")]:



Frissíti a felhasználó intropont számát.

Példa adat:  
{"Uid": 1,”NewIntropoints": 100}

Ehhez tartozik egy belső modell, az UpdateIntropointsRequest.

**Weboldal**

**7.1.: A weboldal célja:**

A projekt ezen része a vásárlók felé irányul. Akár számítógépes, akár mobilos eszközön müködőképes a felhasználó számára a felület. Ezen keresztül tekinthetik meg a látogatók a kávézó promotálására szolgáló oldalt, a bejelentkező panelt, illetve a rendelés leadásához szükséges felületet.

**7.2.: A weboldal szoftveres megvalósítása:**

A weboldal struktúráját illetve alapvető elemeit a HTML5 nyelven készítettük, ennek vizuális díszítéséhez a CSS nyelvet használtuk, a funkciókat, illetve az adatbázissal való kommunikáciért felelős függvényeket pedig JavaScript-ben hoztuk létre. A programfájlok szerkesztéséhez a Microsoft Visual Studio Code applikációt vettük igénybe.

Az index.html fájl az alapweboldal, ezt éri el a felhasználó alapértelmezetten. A login.html a bejelentkező felület, amit az index oldal navigációs sávjából érünk el, az ordersite.html pedig a rendelés leadás felülete, amit bejelentkezés után szintén a navigációs sávból lehet elérni. Ezen felül része a fájloknak egy “img” mappa, amiben az összes felhasznált képet tároljuk, egy css mappa, amiben a html dizájnolásához szükséges fájlok találhatók, illetve egy js mappa, amiben pedig a funkcionalitáshoz szükséges függvényekért felelős JavaScript fájlokat találjuk. A reszponzivitás érdekében a következő automatikus méretezőket csatoltuk a programhoz:

- https://code.jquery.com/jquery-3.5.1.slim.min.js

- https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js

- https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js

- https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css

Az JavaScript-ben lévő alertek díszítésére a „<https://sweetalert2.github.io/#examples>” bővítményt használtuk.

**7.3.: Index oldal bemutatása:**

Az oldal legfelső eleme egy navigációs sáv. Ezen a kávézó logója, illetve a navigáció gombok kapnak helyet. Utóbbiak a főoldal, bejelentkező felület, illetve a rendelő felület közti váltásra szolgálnak.

A fejsávban egy kép látható, illetve előtte a cég neve.

Lejjebb tekerve található egy “Rólunk” szegmens, ahol a cég mutatkozik be röviden.

Ez alatt található a kávézóban dolgozok listája képpel, névvel és pozícióval ellátva.

A következő elem egy “Termékek” konténer, ahol a kávézóban árult kávékat és desszerteket láthatjuk. A típusok közötti váltáshoz a fenti “Italok” vagy “Desszertek” gombot kell megnyomni a felhasználónak. A szeparáció a letisztultság és rendezettség érdekében jött létre.

A footer szegmens a weblap vége, amiben a kávézó információ láthatók.

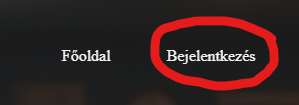
- Ilyen az elérhetőségek: Telefonszám, Cím, E-mail

- A nyitvatartási idő,

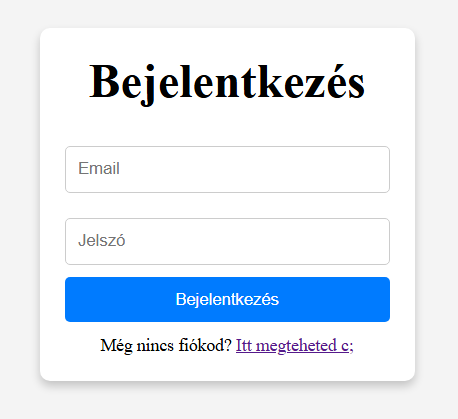
- Illetve a kávézó közösségi portáljaira irányuló megnyomható képek.

**7.4.: Bejelentkezés oldal bemutatása:**

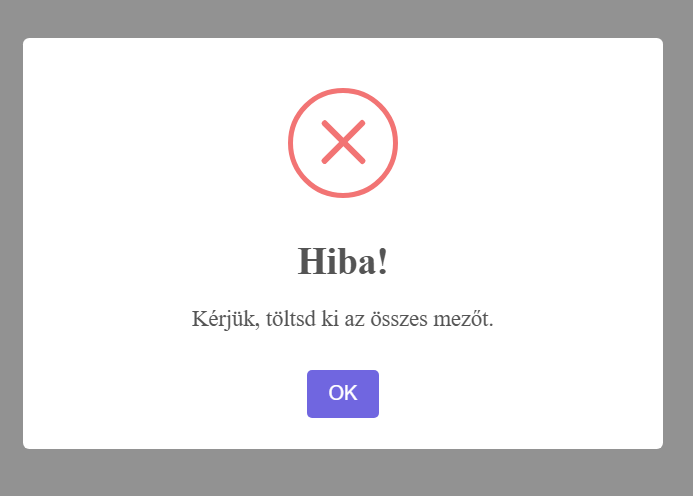
Az index oldal navigációs sávjában található menüponttal érhető el.

****

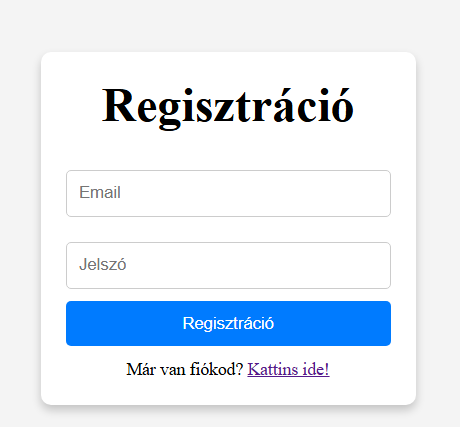
Ilyenkor megnyílik az új felület, amin a felhasználó bejelentkezhet meglévő profiljába.



A megfelelő adatok beírása, majd a bejelentkezés gomb megnyomása után a program betölti a felhasználó azonosítóját, eltárolja egy helyi változóba, majd visszavissz az index oldalra, amin így megnyílik a kijelentkezés, illetve a rendelés opciója. Amennyiben a két mező bármelyike kitöltetlen, a program, jelez a felhasználónak.



Amennyiben a látogatónak még nincs fiókja az adatbázisban, a “Még nincs fiókód? Itt megteheted” szövegre kattintva léphet be a regisztrációs felületre.

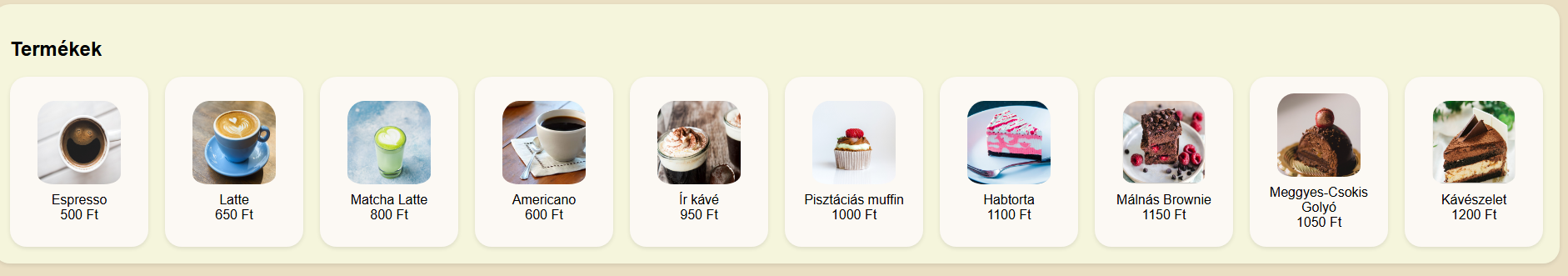


Regisztrációkor az email mezőbe kizárólag email formátumú szöveget fogad el a program (valami@valami.hu pl.). A login felülethez hasonlóan, a program itt sem engedi, hogy bármelyik mező kitöltetlen maradjon. Kitöltés után a kód végez egy ellenőrzést, hogy a megadott email nem-e szerepel már az adatbázisban. Ha igen, hibával tér vissza, és újat kell megadni. Ha minden feltételnek megfelelnek az adatok, regisztrálja a felhasználót, majd visszaviszi a bejelentkezés képernyőre, hogy új fiókjával beléphessen.

**7.5.: Rendelés oldal bemutatása:**

Az index.html navigációs sávjában lévő rendelés gomb megnyomásával kezdődik meg a rendelés folyamata, ám ez a gomb kizárólag akkor elérhető, ha adott felhasználó be van jelentkezve az oldalon. 

Ilyenkor megnyitva az új html oldalt, itt a felhasználó egy “termékek” listából kiválaszthatja kattintással az általa felvenni kivánt termékeket.



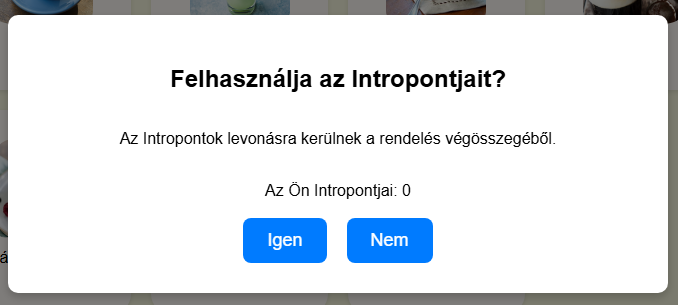
A felvett termékeket egy kosár modal ablakhoz adja hozzá, ami a felület jobb felső sarkában lévő kosár ikonnal nyitható meg. Egy termék egyszeri megnyomásával 1 db-ot ad hozzá a listához az adott termékből. Amennyiben olyan termékre kattint a felhasználó, ami már egyszer bekerült a listába, nem új sorként veszi azt fel, hanem darabszámlálóként hozzáadja az adott termék nevéhez. Minden felvett termék neve és mennyisége mellett megjelenik két automatikusan generált gomb. Ezekkel tudja a felhasználó adott termék mennyiségét csökkenteni vagy növelni. Ennek megnyomásával az adott termék mennyiség számlálójából levesz a program egy darabot. Amennyiben a számláló eléri a 0-át, a terméket teljesen kitörli a listából. A lista alatt található egy végösszeg számláló, amelynek értéke minden termék hozzáadásakor illetve törlésekor automatikusan frissül. A bezárás gomb elrejti a kosár felületét.



A kosár ikon bal alsó sarkában található egy darab számláló, ami minden termék hozzáadása, vagy levonása után automatikusan frissül.



A rendelés leadása gomb megnyomása után megnyílik egy intropont nevezetű modal ablak, amin arról dönthet a felhasználó, hogy a fiókjához tartozó hűségpontjait szeretné-e felhasználni a végösszeg csökkentéséhez.



Bármelyik választási lehetőség után megnyílik egy rendelés típus modal, amin az válaszható ki, hogy helyben szeretnénk fogyasztani, vagy elvitelre csomagoltatni.  


A rendelés típus kiválasztása után a program összegyűjti az eddigi adatokat:

- Rendelt termékek lista: uploadedItems,

- Rendelés végösszege: uploadedTotalCost,

- Rendelés típusa: uploadedTakeway,

- Rendelést leadó felhasználó azonosítója,

- Felhasználó által elhasznált intropontok

majd ezeket a Web API-n keresztül feltölti az adatbázisba, mint egy új rendelést, majd a javascript oldalon minden adatot nulláz az azonnali új rendelés felvétele lehetőségének érdekében.

Az oldal alján található egy rendelés törlése gomb, aminek megnyomásával minden JavaScript változót kinulláz a program, majd visszaviszi a felhasználót az index.html oldalra, így megszakítva a rendelési folyamatot.

**7.6.: JavaScript funkciók müködésének bemutatása:**

**auth.js működése:**

Ez a fájl biztosítja a regisztráció, bejelentkezés, kijelentkezés, valamint az űrlapok közötti váltás logikáját.

**register() funkció:**

Ez a funkció felelős az új felhasználók regisztrációjáért. A működése a következő lépésekből áll:

Lépések:

**-** Adatok lekérése az űrlapból:

- Az e-mail címet és a jelszót a register-email register-password mezőkből kiolvassa a program.

- E-mail cím validálása:

- Egy reguláris kifejezéssel ellenőrzi, hogy az e-mail cím formátuma helyes-e.

- Ha nem megfelelő, egy alertel jelzi a felhasználónak a hibát.

- Mezők kitöltésének ellenőrzése:

- Ellenőrzi, hogy az e-mail és jelszó mezők tartalma nem-e üres.

- Ha valamelyik mező üres, alertel jelzi a hibát.

- E-mail cím meglétének ellenőrzése:

- A web api check-email végpontját használva ellenőrzi, hogy a megadott email cím szerepel-e már az adatbázisban.

- Ha már szerepel, alertben jelzi a felhasználónak a hibát.

- Regisztráció végrehajtása:

- Ha az adatok minden feltételnek megfeleltek, a web api register végpontjával rögzíti a felhasználót az adatbázisba. Hiba esetén hibaüzenetet küld, sikeres regisztráció esetén pedig alertben értesíti a felhasználót a sikeres adatrögzítésről

- Hiba kezelése:

- Ha bármilyen hiba történik a regisztráció során, egy általános hibaüzenetet jelenít meg.

**login() funkció:**

Ez a funkció kezeli a felhasználók bejelentkezését.

- Adatok lekérése az űrlapból:

- Az e-mail címet és a jelszót a login-email és a login-password mezőkből beolvassa.

- Mezők kitöltésének ellenőrzése:

- Ellenőrzi, hogy az e-mail és jelszó mezők nincsenek-e üresen.

- Ha valamelyik mező üres, alertel hibaüzenetet jelenít meg.

**-** Bejelentkezési kérelem:

- A web api login végpontján keresztül bejelentkezteti a felhasználót az e-mail és jelszó paraméterekkel.

- Ha a bejelentkezés sikeres, a felhasználói adatokat elmenti egy helyi adattárolóba, és átirányítja a főoldalra.

**-** Hiba kezelése:

- Ha a bejelentkezés sikertelen, alertel hibaüzenetet jelenít meg.

**logout() funkció:**

Ez a funkció törli a felhasználói adatokat, majd visszairányít a vendég főoldalra.

**showRegister() és showLogin() funkciók:**

A regisztrációs és bejelentkezési űrlapok közötti váltást kezelik.

**orderjs.js működése:**

Ez a fájl biztosítja a kosár kezelését, a rendelési típus kiválasztását, az Intropontok felhasználását, valamint a rendelési adatok elküldését a backend API-nak.

**addToCart(itemName, itemPrice):**

Ez a funkció felelős egy termék hozzáadásáért a kosárhoz.

Lépések:

- Értesítés megjelenítése:

- Egy alert üzenet jelzi, a termék sikeres hozzáadását a kosárhoz.

- Kosár frissítése:

- Ha a termék már szerepel a kosárban, növeli annak mennyiségét és árát.

- Ha a termék még nem szerepel a kosárban, egy egységni árral és mennyiséggel a listához adja azt.

- Kosár számláló frissítése:

- Az updateCartCounter függvény meghívásával frissíti a kosár ikon mellett lévő mennyiség számlálót.

**updateCartCounter() működése:**

Ez a funkció frissíti a kosárban lévő termékek számát a felhasználói felületen.

Lépések:

- Összeadja a kosárban lévő termékek mennyiségét.

- A kosár számláló (cart-counter) elem szövegét frissíti az összesített mennyiséggel.

**showCartModal() működése:**

Ez a funkció megjeleníti a kosár tartalmát egy modal ablakban.

Lépések:

- A kosárban lévő termékeket listázza, megjelenítve a nevüket, mennyiségüket és árukat.

- Hozzáad egy „+” és egy „-„ gombot minden termékhez, amelyekkel a felhasználó növelheti vagy csökkentheto a termék mennyiségét.

- Kiszámítja és megjeleníti a kosár teljes árát.

- A kosár modal ablakot láthatóvá teszi.

**closeCartModal() működése:**

- Bezárja a kosár modal ablakot.

**cancel() működése:**

- Kinulláz minden értéket az oldalon, majd visszaviszi a felhasználót az index.html oldalra.

**showIntropointsModal() működése:**

- Az intropontok felhasználásának kiválasztására használandó modal ablakot láthatóvá teszi.

**checkout() működése:**

Ez a funkció kezdi meg a rendelés leadásának folyamatát.

- Ellenőrzi, hogy van-e termék a kosárban. Ha nincs, alertel hibaüzenetet jelenít meg.

- Ha van termék a kosárban, megjeleníti az intropontok modal ablakot.

**useIntropoints() működése:**

Ez a funkció lehetővé teszi a felhasználók számára, hogy Intropontokat használjanak a rendelés végösszegének csökkentésére.

Lépések:

- Lekéri a felhasználó intropontjait a web api-n keresztül

- Ha az intropontok fedezik a teljes összeget, a végösszeg 0 lesz. Ha nem, az intropontokat levonja a rendelés végösszegéből.

- Frissíti a felhasználó intropontjait.

- Bezárja az intropontok modal ablakot, majd megnyitja a rendelés típus kiválasztására szolgáló modal ablakot.

**skipIntropoints() működése:**

Ez a funkció lehetővé teszi a felhasználó számára, hogy ne használjon Intropontokat a rendelés során. Ebben az esetben a rendelés teljes összege változatlan marad, és az Intropontok nem kerülnek levonásra.

- A kosár tartalmát feldolgozza, és kiszámítja a végösszeget.

- Bezárja az intropoint modal ablakot, majd megnyitja a rendelés típus kiválasztására szükséges modalt.

**updateUserIntropoints(uid, newIntropoints) működése:**

- a web api user/update-intropoints végpontját használva frissíti az új számolt értékre a felhasználó intropontjait.

**fetchAndDisplayIntropoints() működése:**

- Lekéri a bejelentkezett felhasználó adatait a helyi tárolóból. Ha a felhasználó nincs bejelentkezve, egy üzenetet jelenít meg, miszerint az intropontok nem elérhetők.

- A web api user/intropoints végpontjára kérést küld a felhasználó uid azonosítójával.

- Ha a válasz sikeres, a lekért intropontokat megjeleníti az #intropoints-modal-text azonosítójú szövegben. Ha a válasz sikertelen, hibaüzenetet jelenít meg.

- Ha az api hívás során hiba történik, hibaüzenetet jelenít meg.

**showOrderTypeModal() működése:**

A rendelés típus kiválasztására szolgáló modal ablakot láthatóvá teszi.

**setOrderType(type) működése:**

Ez a funkció kezeli a rendelési típus kiválasztását (pl. helyben fogyasztás vagy elvitel), és elküldi a rendelési adatokat a backend API-nak.

Lépések:

- Lekéri a bejelentkezett felhasználó adatait a helyi adattárolóból. Ha a felhasználó nincs bejelentkezve, egy alert üzenetet jelenít meg, amely figyelmezteti, hogy be kell jelentkeznie a rendelés leadásához.

- A kosár tartalmát feldolgozza, és egy szöveges változót készít a termékekről és azok mennyiségéről.

- Összeállítja a rendeléshez szükséges adatokat:

- a felhasználó azonosítóját (uploadedUserId).

- a rendelés típusát (uploadedTakeway), ezt az adatot örököltetjük a függvénybe a type változóval.

- rendelt termékek (uploadedItems).

- rendelés végösszegét (uploadedTotalCost), ami az intropontok felhasználás modal döntésétől változhat.

- a felhasznált intropontokat (uploadedUsedIntropoints), ez lehet 0 is, ha úgy dönt a felhasználó.

- A web api uploadneworder végpontján keresztül feltölti a rendelést az adatbázisba. Ha sikeres a leadás, a felvétellel együtt a felhasználó megmaradt intropontjait is frissíti. Sikertelen leadás esetén hibaüzenettel jelez a konzolban.

- Kiüríti a kosár teljes tartalmát, frissíti a kosár számlálót, bezárja a rendelés típus kiválasztás modal-t, illetve alertel jelzi a sikeres leadást.

**Tesztelés**

**8.1.: A tesztelés menete:**

A teszteléshez kizárólag manuális felhasználói teszteket végeztünk. Ezek menete a kövezkező volt.

Regisztráció:

Az email cím mezőbe hiába írtunk be a formátumnak megfelelő címet, a program egyiket sem fogadta el. Ilyen volt például az

- intro,

- intro@.com,

- intro.com,

- intro.,

- intro@,

- intro@intro.c,

- 1234,

így majdnem biztosan jelenthető a mező hibamentesnek.

Bejelentkezés:

Bármilyen nem létező adat kombináció került beírásra, a program elutasította azt, mondván nem szerepel az adatbázisban, így a program ezen része is majdnem hibamentesnek nyílvánítható.

Rendelésleadás:

A termékekre kattintáskor a termék sikeresen hozzáadódott a kosárhoz, a megnyitáskor a kosárban helyesen megjelennek a termékek, illetve az azokhoz generált gombok, bármilyen módosítás során helyesen frissült a végösszeg, illetve a betöltött intropontok, illetve az elmentett felhasználó azonosítója is minden esetben helyesen volt tárolva. A leadott intropontok is mindkét esetben helyesen lettek felvéve, legyen a rendelés végösszegénél kevesebb, egyenlő vagy több intropontja a felhasználónak.

Rendelés törlés:

A teljesített rendelés csak akkor került törlése, ha az adott sor aktívan ki volt választva. Helyes használat esetén teljesen ki lett törölve az adatbázisból a rendelés, illetve a kiszámolt intropontokat is hozzáadta a felhasználóhoz.

Pincér rendelés felvétel:

Hibamentes felület, a felület tökéletesen felveszi az adatokat.

**Fejleszthetőségi lehetőség**

**9.1.: Lehetőségek:**

1. Azonnali bankártyás fizetés lehetőségének létrehozása webes rendeléskor, akár az Introcafe szerverén tárolt kártyaadatok segítségével, akár külső fizetés kezelőn keresztül (pl. Barion).

2. Újdonságok menü hozzáadása a weboldalhoz.

3. Telefonra telepíthető applikáció, ami a rendelés leadás, illetve profiladatok megtekintésére szolgál.

4. Profil menüpont a bejelentkezett felhasználóknak, ahol módosíthatják jelszavuk, email címük, megtekinthetik intropontjaik számát, illetve törölhetik a fiókukat.

5. Rendeléskor a termékek szortírozása kategória szerint (pl. kávék, desszertek, stb.).

6. AI asszistens implementálása.

**Források**

Képek:

https://unsplash.com/photos/selective-focus-photography-of-latte-in-teacup-jn-HaGWe4yw

https://unsplash.com/photos/clear-drinking-glass-on-brown-wooden-table-nDdiili1xV0

https://unsplash.com/photos/green-and-white-floral-cup-on-white-ceramic-saucer-C63YZ33DdvY

https://unsplash.com/photos/baked-cake-uG1jwfpCRhg

https://unsplash.com/photos/brown-and-black-sliced-cake-on-white-ceramic-plate-395IoDgBxQI

https://unsplash.com/photos/clear-short-stem-glass-on-tabletop--BlcyBSdBXo

https://unsplash.com/photos/selective-focus-photo-of-glass-of-brown-liquid-X6aY\_j6JD\_Y

https://cdn.borsonline.hu/2022/04/NmAP8JQjglrf039fTJC-G5KHLJLhcp7OhUN\_E0BUEuk/fill/1347/758/no/1/aHR0cHM6Ly9jbXNjZG4uYXBwLmNuYXBwLmNvbnRlbnQ0ZS9jb250ZW50LzBkODYyYmJjZmNkMjRhODRhNDI2OTI0YjBiZYjBiZD2.jpg

https://www.pexels.com/photo/photograph-of-chocolate-cupcake-with-red-strawberry-toppings-1055272/

https://kr.pinterest.com/pin/538883911671470107/

https://tesztek.tudatosvasarlo.hu/wp-content/uploads/2022/07/kavezok-tesztje-768x512.jpg.webp

Logó és main background image:

A Microsoft Bing AI képgeneráló program segítségével készültek.

Ikonok:

https://www.iconpacks.net/

# NYILATKOZAT a vizsgaremek eredetiségéről és az elektronikus feltöltésről

Alulírottak……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………... (név, oktatási azonosító),

a Pápai SZC Faller Jenő Technikum, Szakképző Iskola és Kollégium tanulói, ezúton nyilatkozunk, hogy az Informatikai rendszer és alkalmazás üzemeltető technikusi diákjai büntetőjogi és fegyelmi felelősségünk tudatában kijelentjük és aláírásunkkal igazoljuk, hogy a portfólió/vizsgaremek saját, önálló munkánk eredménye.

A hivatkozott irodalmi és egyéb információs források felhasználása a szerzői jogok általános szabályainak, a forráskezelés szabályainak megfelelően történt, a készítésre vonatkozó szabályokat betartottuk.

Tudomásul vesszük, hogy a portfólió/vizsgaremek esetén plágiumnak számít:

 szószerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;

 tartalmi idézet hivatkozás megjelölése nélkül;

 más publikált gondolatainak saját gondolatként való feltüntetése;

 más által készített rajzok saját rajzként való feltüntetése.

Alulírottak kijelentjük, hogy a plágium fogalmát megismertük, és tudomásul vesszük, hogy plágium esetén a portfólió/vizsgaremek visszautasításra kerül.

Várpalota, ……………………………………….

……………………………………….……………………………………….

……………………………………….……………………………………….

aláírás