

Ceglédi SZC Mihály Dénes Szakgimnáziuma és Szakközépiskolája

2230 Gyömrő, Fő tér 2/B

OM Azonosító: 203068

SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ TECHNIKUS SZAKMA
AZONOSÍTÓ SZÁMA: 5 0613 12 03

Könyvklub Webshop



Készítette: Trepák Attila, Nagy Viktória és Király Krisztián

Gyömrő, 2023

Tartalomjegyzék

1. Bevezetés.....	3
1.1. Témaválasztás.....	3
1.2. A projekt folyamata	3
2. Fejlesztői dokumentáció	4
2.1. Alkalmazott fejlesztői eszközök.....	4
2.2. Adatbázis	5
2.3. A weboldal felépítése és lényegesebb függvényei	7
2.4. Az asztali alkalmazás felépítése és lényegesebb függvényei.....	15
3. Tesztelés	20
3.1. PHP tesztelés.....	20
3.2. PHP tesztesetek.....	21
3.3. API végpont tesztelés	21
3.4. API végpont tesztesetek.....	21
3.5. Frontend tesztelés	23
4. Asztali alkalmazás felhasználói dokumentáció	24
4.1. Program telepítése	24
4.2. Program használata.....	26
5. Továbbfejlesztési lehetőségek.....	30
5.1. Weboldal továbbfejlesztése	30
5.2. Adminisztrációs felület továbbfejlesztése	30
6. Összegzés	30

1. BEVEZETÉS

1.1. Témaválasztás

Manapság már saját honlap nélkül szinte lehetetlen eredményeket elérni, ezért sorra készülnek a honlapok minden területen. Csatatunk ezért is döntött úgy, hogy projektünk témájául a weboldal készítést választja, így született meg a „Könyvklub webshop” ötlete.

Törekedtünk arra, hogy amikor a látogató megérkezik weboldalunkra, egyedi, minőségi honlappal találkozzon. A fejlesztés során fiktív adatokkal, termékekkel dolgoztunk, ez nagyban megnehezítette a munkánkat, mivel sok időt vett igénybe az adatok, képek gyűjtése és feldolgozása.

1.2. A projekt folyamata

A webfejlesztés meghatározott lépések egymásutániségéből felépülő folyamat, ahol maga a programozás önmagában nem elégséges a siker szempontjából, éppen ezért projektünket az alábbi folyamatpontokon sorba haladva készítettük el.

1. Információ gyűjtés
2. Kutatás és tervezés
3. Design
4. Programozás
5. Tesztelés és élesítés
6. Karbantartás és üzemeltetés

A csapatmunkában hárman veszünk részt, így feladatokat három részre osztva oldottuk meg.

- Frontend
- Backend
- Tesztelés

A webfejlesztők között széleskörűen elterjedt technológia a Git, mely arra való, hogy a nagyobb projekteken való munka – amin egyszerre többen is dolgoznak – egyszerűbbé, átláthatóbbá és visszakereshetővé váljon, ezért választottuk mi is ezt a verziókezelő rendszert.

A projektünket egy GitHub repository-ba, azaz egy GitHub könyvtárba helyezzük, ami a GitHub szerverén tárolódik. Ezt minden csapattag elérheti, a fejlesztői munka megkezdése előtt letölti az aktuális állapotot a könyvtárból, dolgozik rajta, majd, ha elkészült az aktuális feladattal, visszatölti azt, így aki tovább dolgozik majd, az a már továbbfejlesztett verziót fogja elérni.

2. FEJLESZTŐI DOKUMENTÁCIÓ

2.1. Alkalmazott fejlesztői eszközök

Leíró és programozási nyelvek:

- PHP
- HTML
- CSS
- JavaScript
- C#
- XAML

Fejlesztői környezetek (IDE):

- Microsoft Visual Studio Code
- Microsoft Visual Studio 2022

Adatbáziskezelő rendszer:

- MySQL

Verziókezelő rendszer:

- GitHub

Lokális webszerver:

- Apache

Webszerver:

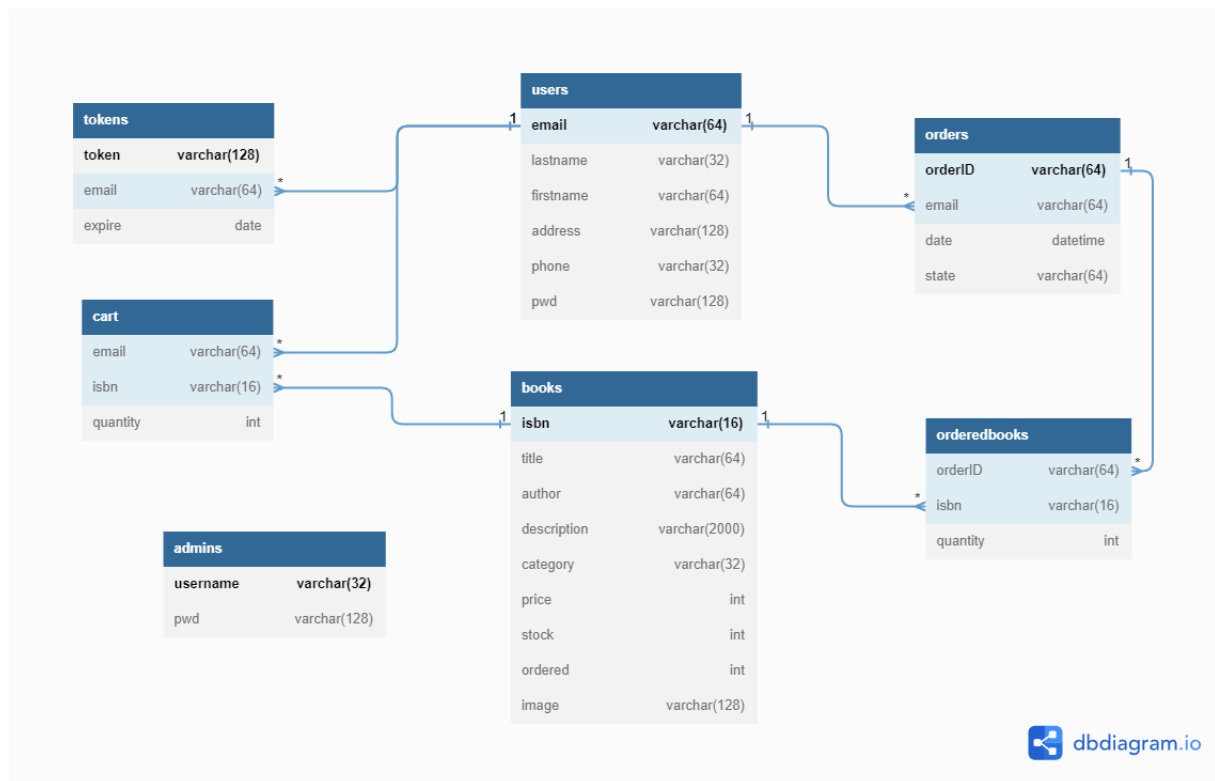
- Rendszer: Ubuntu 20.04 LTS
- Használt szolgáltatások: MySQL, FTP

Egyéb használt programok:

- XAMPP
- Postman
- Figma
- NSIS
- Notepad++
- Google Chrome
- Mozilla Firefox
- Microsoft Excel

2.2. Adatbázis

Adatbázis felépítése:



A **users** tábla szolgál a felhasználók adatainak tárolására:

- email (PK) – a felhasználó e-mail címe
- lastname – a felhasználó vezetéknéve
- firstname – a felhasználó keresztnéve
- address – a felhasználó címe
- phone – a felhasználó telefonszáma
- pwd – a felhasználó jelszavának titkosított karakterlánc (hash)

A **books** tábla szolgál a könyvek tárolására és a készlet követésére:

- isbn (PK) – a könyv egyedi ISBN száma
- title – a könyv címe
- author – a könyv írója
- description – a könyv leírása
- category – a könyv kategóriája
- price – a könyv ára
- stock – a könyvből készleten lévő mennyiség
- ordered – a könyvből eddig megrendelt mennyiség
- image – a könyvről eltárolt és megjelenített kép neve

Az **orders** tábla szolgál az elküldött rendelések tárolására és ezek aktuális állapotának követésére:

- orderID (PK) – a rendelés egyedi azonosítója
- email – a rendeléshez tartozó felhasználó e-mail címe
- date – a rendelés dátuma
- state – a rendelés állapota

Az **orderedbooks** tábla segítségével kapcsoljuk össze a rendeléseket a hozzájuk tartozó könyvekkel:

- orderID (FK) – a rendelés egyedi azonosítója
- isbn (FK) – a könyv egyedi ISBN száma
- quantity – a rendelt mennyiség

A **cart** táblában tároljuk a felhasználók által kosárhoz adott termékeket, hogy ezek ne csak lokálisan, hanem bárholnan elérhetőek legyenek:

- email (FK) – a felhasználó e-mail címe
- isbn (FK) – a könyv egyedi ISBN száma
- quantity – a kosárban lévő mennyiség

A **tokens** táblában tároljuk a jelszó visszaállításához szükséges egyszerhasználatos karakterláncokat:

- token (PK) – egy egyedi karakterlánc
- email – a felhasználó e-mail címe
- expire – a token érvényességének dátuma

Az **admins** tábla tartalmazza az adminisztrátorok belépéshez szükséges adatait:

- username (PK) – az admin felhasználóneve
- pwd – az admin jelszavának titkosított karakterlánca (hash)

2.3. A weboldal felépítése és lényegesebb függvényei

Általános függvények:

- **var_check(array, key, check_value = false, value = „”)**

Paraméterek:

- array: array – globális változókat tartalmazó tömb (\$_POST, \$_GET)
- key: string – globális változó elnevezése
- check_value: bool (opcionális) – globális változó ellenőrzése (nem lehet üres string)
- value: string (opcionális) – a változó elvárt értéke

Leírás:

- a megadott paraméterektől függően bool típusú értékkel tér vissza

- **get_random_string(length = 32)**

Paraméterek:

- length: int (opcionális) – a random generált string hossza

Leírás:

- visszaad egy random karakterekből álló stringet, aminek a hossza a length paraméterben megadott érték

Adatbázis kapcsolat:

Az adatbázis kapcsolatért a DbConnection objektum felel, amely létrehozásakor automatikusan kapcsolódik az előre meghatározott adatbázishoz. Rendelkezik egy privát „connection” nevű változóval, ami egy „PDO” objektum. Ezen felül pedig tartalmazza az alábbi metódust.

- **query(sql)**

Paraméterek:

- sql: string – az sql lekérdezés teljes szövege

Leírás:

- sikeres lekérdezés esetén egy „PDOStatement” objektummal tér vissza, aminek a kezeléséről már a meghívás helyén kell gondoskodnunk
- sikertelen lekérdezés esetén „false” lesz az értéke

Modellek:

Az OOP (Objektum Orientált Programozás) elvek alapján különböző objektumokat hoztunk létre, amik segítenek pontosabban meghatározni egy valós elem (pl. könyv) állandó tulajdonságait. Az objektumok egy része az adatbázisban eltárolt táblákra utal és a mezői is megegyeznek a tábla mezőivel. Ezek közül a lényegesebbek a következők: Book, User, Order, CartItem, Admin, Token. Az „orderedbooks” tábla nem kapott külön osztályt, ennek a megvalósítása az **Order**-en belül történt, egy **Book** objektumokból álló tömb formájában.

Ezekén kívül létrehozott további modellek:

- **Encryption**

Leírás:

- ez az objektum felel a jelszavak titkosításáért és ellenőrzéséért

Metódusok:

- **encrypt(pwd)** – titkosítja a paraméterként kapott szöveget és vissza is adja azt
- **decrypt(pwd, hash)** – egy beépített függvény segítségével ellenőrzi, hogy a megadott szöveg titkosított változata egyezik-e a „hash” paraméterként megadott karaktersorozattal, és ez alapján egy bool értékkel tér vissza

- **EnvReader**

Leírás:

- ebben az objektumban tároljuk az adatbázis eléréséhez szükséges IP címet, felhasználónevet, jelszót és az adatbázis nevét

Metódusok:

- **read_env()** – beolvassa az adatokat tartalmazó „env” fájlt és visszaadja azokat egy asszociatív tömb formájában

- **Email**

Leírás:

- ezzel az objektummal határozzuk meg a kiküldött e-maileket, amik lehetnek regisztrációt vagy rendelést visszaigazoló e-mailek, vagy tartalmazhatják a jelszó megváltoztatásához szükséges linket
- létrehozásakor át kell adnunk egy **User** objektumot, az e-mail tartalmát, és opcionálisan megadhatjuk a tárgyat is, de ez alapesetben „Könyvklub” lesz
- a megszólítást és az aláírást alapból tartalmazza az objektum

Metódusok:

- **send()** – az objektum létrehozása után ezzel a metódussal küldhetjük el az e-mailt a felhasználónak

Alapvető adatbáziskezelő függvények:

Az adatbázis tábláinak kezelésére 1-1 objektumot hoztunk létre, melyek mind tartalmazzák az **IDbHandler** interface-ben meghatározott alapvető- és ezen felül az aktuális táblára vonatkozó specifikus függvényeket. Emellett pedig megvalósítanak egy **DbConnection** objektumot is.

- **select(key)**

Paraméterek:

- key: string – a lekérdezni kívánt rekord egyedi azonosítója

Leírás:

- lekérdez egy rekordot az adatbázis adott táblájából a paraméterként megadott azonosító alapján
- a visszatérési értéke a táblától függ, de legtöbb esetben egy, a modellek között meghatározott objektumot ad vissza

- **select_ordered(order_field, order_type)**

Paraméterek:

- order_field: string – a rendezés alapjául szolgáló mező neve
- order_type: string – a rendezés típusa, ami lehet növekvő (ASC) vagy csökkenő (DESC)

Leírás:

- visszaadja a tábla összes elemét egy tömbben, amelyet a paraméterként megadott értékek alapján rendez sorba

- **select_contains(search, order_field, order_type)**

Paraméterek:

- search: string – a keresendő elem/elemek meghatározása
- order_field: string – a rendezés alapjául szolgáló mező neve
- order_type: string – a rendezés típusa, ami lehet növekvő (ASC) vagy csökkenő (DESC)

Leírás:

- ezzel a metódussal tudunk keresni a megadott táblában
- visszaadja a keresésnek megfelelő elemeket egy tömbben, amelyet a paraméterként megadott értékek alapján rendez sorba
- ha nincs a keresésnek megfelelő elem, akkor egy üres tömböt ad vissza

- **insert(object)**

Paraméterek:

- object: var – ez a paraméter a táblától függ, de legtöbb esetben egy, a modellek között meghatározott objektumot vár paraméterként

Leírás:

- ezzel a metódussal tudunk új elemet rögzíteni az aktuális táblába
- a visszatérési értéke a legtöbb esetben bool típusú és a hozzáadás sikerességétől függ

- **update(object)**

Paraméterek:

- object: var – ez a paraméter a táblától függ, de legtöbb esetben egy, a modellek között meghatározott objektumot vár paraméterként

Leírás:

- ezzel a metódussal módosíthatjuk egy létező elem valamelyik értékét az aktuális táblában
- a visszatérési értéke a legtöbb esetben bool típusú és a módosítás sikerességétől függ

- **delete(object)**

Paraméterek:

- object: var – ez a paraméter a táblától függ, de legtöbb esetben egy, a modellek között meghatározott objektumot, vagy a törölni kívánt elem egyedi azonosítóját várja paraméterként

Leírás:

- ezzel a metódussal törölhetünk egy bizonyos elemet az aktuális táblából
- a visszatérési értéke a legtöbb esetben bool típusú és a törlés sikerességétől függ

Eszközök:

A használt függvényeket és ellenőrzéseket tematikusan külön fájlokba rendeztük. Ezek a fájlok „_tools” végződést kaptak, a fájlnev első része pedig a benne lévő függvények/műveletek feladatára utal (pl. login_tools).

Az legtöbb fájl elején ellenőrzések vannak, amelyek az oldal betöltésekor/frissítésekor minden esetben lefutnak és végrehajtják a bennük meghatározott utasításokat. Ezek az folyamatok az esetek nagy részében POST és GET változók ellenőrzését végzik és ez alapján hajtanak végre módosításokat, vagy irányítják át a felhasználót a megfelelő oldalra. Itt hajtjuk végre azokat az ellenőrzéseket is, amelyek meggátolják a felhasználót abban, hogy olyan oldalra látogasson, amihez nincs jogosultsága.

A fájlok feladatai az alábbi módon oszlanak meg:

- **login_tools**

Leírás:

- a bejelentkezésért és az ehhez kapcsolódó ellenőrzésekért felel

Fontosabb függvények:

- **login(email, pwd)** – ellenőrzi az e-mail címet és a jelszót, ha mindent rendben talál, akkor a szükséges adatok sütikben való eltárolása után bejelentkezteti a felhasználót
- **logout()** – kijelentkezteti az aktuális felhasználót
- **login_display(), logout_display()** – ezek segítségével állítjuk a bejelentkezés és kijelentkezés ikonok láthatóságát, 1-1 „session” változóban eltárolt „display” paramétert adnak vissza, melyek később a HTML kódban vannak meghívva
- **is_logged_in(), is_logged_out()** – a legtöbb oldal elején ezekkel a függvényekkel ellenőrizzük, hogy a felhasználónak van-e jogosultsága az oldal megnyitására, és, ha nincs, akkor visszairányítjuk a főoldalra

- **signup_tools**

Leírás:

- a regisztrációért és az ehhez tartozó ellenőrzésekért felel
- az elején meghatározott elágazásban, a POST változókon keresztül megkapott értékekből létrehozunk egy **User** objektumot

Fontosabb függvények:

- **signup(user, pwd)** – a korábban létrehozott **User** objektumot és a beírt jelszót kell átadnunk ennek a függvénynek, ezután hozzáadjuk az adatbázishoz az új felhasználót, és, ha minden sikeres, akkor be is jelentkeztetjük

- **send_verify_email(user)** – a korábban létrehozott **User** objektumot kell átadnunk ennek a függvénynek, és ezt a függvényt a regisztráció folyamán hívunk meg, hogy megerősítő e-mailt küldjünk a felhasználónak a sikeres regisztrációról

- **book_tools**

Leírás:

- a könyvek és a kategóriák lekérdezéséért felel

Fontosabb függvények:

- **get_books()** – lekérdezi a könyveket az adatbázisból, ha van keresés, akkor csak a keresésnek megfelelő könyveket adja vissza, különben pedig az összes készleten lévő könyvet visszaadja egy **Book** objektumokat tartalmazó tömb formájában
- **get_top_books()** – lekérdezi a top 10 leggyakrabban megrendelt könyvet az adatbázisból és visszaadja egy **Book** objektumokat tartalmazó tömb formájában
- **get_categories()** – lekérdezi a könyveket az adatbázisból és visszaad egy listát a könyvek kategóriáiból, amelyben minden kategória csak egyszer szerepel (ezt használjuk a könyvek szűréséhez)
- **get_image()** – az adatbázisban tárolt név alapján megkeresi a könyv borítóját a szerveren, ha megtalálja, akkor visszaadja az elérési útját, különben pedig egy általános borítót ad vissza, ezzel oldjuk meg azt, hogy minden könyvnek legyen borítója az oldalon

- **order_tools**

Leírás:

- a rendelések elküldéséért felel

Fontosabb függvények:

send_order()

- ebben a függvényben hajtjuk végre a rendelés teljes folyamatát, feltöltjük a rendelést az adatbázisba, csökkentjük a készletet a rendelt mennyiség alapján, kiürítjük a felhasználó kosarát, és küldünk egy visszaigazoló e-mailt a felhasználónak
- a folyamat végén a felhasználót mindenképp átirányítjuk a message oldalra, ahol hiba esetén figyelmeztetjük, hogy a rendelés sikertelen, sikeres esetben pedig megköszönjük a rendelést
- **send_verify_email()** – ebben a függvényben rakjuk össze az e-mailt, melyet sikeres rendelés esetén küldünk a felhasználónak

- **message_tools**

Leírás:

- az üzenetek kiírásáért felel

Fontosabb függvények:

- **set_message(message)** – eltárolja a paraméterként kapott üzenetet egy „session” változóban
- **get_message()** – ha van valami a „session” változóban, akkor azt adja vissza, különben pedig egy üres szöveget (string-et)

- **cart_tools**

Leírás:

- a kosárért és annak teljes tartalmáért felel
- a fájl elején a POST változók ellenőrzésével növeljük, illetve csökkentjük a kosárban lévő elemek számát

Fontosabb függvények:

- **update_local_cart()** – az adatbázis alapján létrehoz egy „session” változót, amiben lokálisan is eltároljuk a felhasználó kosarának tartalmát
- **get_cart_count()** – ha be van jelentkezve a felhasználó, akkor visszaadja a lokális kosárban lévő elemek számát, különben pedig nullával (0) tér vissza
- **check_stock(cart_item)** – egy kosárban lévő terméket vár paraméterként és ellenőrzi, hogy a termékből van-e még készleten a kosár tartalmán felül, mivel, ha nincs akkor nem enged többet hozzáadni a kosárhoz az adott termékből
- **add_to_cart(isbn)** – hozzáadja a kosárhoz a paraméterként kapott ISBN számmal rendelkező könyvet
- **change_quantity(isbn, operation)** – egy kosárban lévő könyv ISBN számát várja paraméterként, valamint egy **Operation** (enum) típusú változót, ami lehet összeadás (Sum), vagy kivonás (Subtract), és ezek alapján növeli vagy csökkenti a termék mennyiségét a kosárban
- **remove_from_cart(isbn)** – eltávolítja a paraméterként kapott ISBN számmal rendelkező könyvet a kosárból (ezt a függvényt hívjuk meg akkor is, amikor a kosárban egy termék mennyisége eléri a nullát)

- **error_tools**

Leírás:

- a hibaüzenetek kiírásáért felel

Fontosabb függvények:

- **set_error_msg(message)** – beállítjuk egy „session” változó értékét a paraméterként kapott üzenetre
- **get_error_msg()** – ha van értéke akkor visszaadja a „session” változóban eltárolt hibaüzenetet, különben pedig egy üres szöveggel tér vissza
- **get_error_state()** – amennyiben van értéke az eltárolt hibaüzenetnek, akkor láthatóra állítja az üzenetet megjelenítő „div” elemet, különben pedig elrejt (ez a függvény a HTML kódban van meghívva)

- **forget_tools**

Leírás:

- az elfelejtett jelszó módosításáért felel
- a fájl elején ellenőrizzük azokat a POST és GET kéréseket, amikkel eljuttatjuk a felhasználót az elfelejtett jelszótól, az jelszó megváltoztatásáig, illetve itt végezzük el azokat az ellenőrzéseket, amik meggátolják azt, hogy bárki hozzáférjen a jelszó megváltoztatás oldalhoz jogosultság nélkül

Fontosabb függvények:

- **create_token()** – létrehoz egy egyedi random karaktersorozatot, amit hozzáad az adatbázishoz, majd a függvény ezzel a karaktersorozattal tér vissza, hiba esetén pedig „false” lesz a visszatérési értéke
- **create_URL(token)** – felépítjük a linket, amit majd később elküldünk a felhasználónak; ez a link tartalmazza a paraméterként kapott token-t és a felhasználó e-mail címét
- **send_forget_email(user, message)** – a paraméterként kapott **User** objektum és üzenet felhasználásával létrehozunk egy **Email** példányt és ezt elküldjük a felhasználónak, mely tartalmazza a jelszó megváltoztatásához szükséges linket
- **modify_password()** – a felhasználó által megadott új jelszót titkosítjuk és hozzárendeljük a profiljához, a felhasznált token-t töröljük az adatbázisból és végül átirányítjuk a **message** oldalra, ahol tájékoztatjuk a jelszó megváltoztatásának eredményéről

Navigáció:

Az oldalak közti navigációért a **navigation.php** felel. Ebben meghatároztunk egy **page** nevű változót, amiben mindig az aktuális oldal nevét tároljuk. Az oldal nevét az url-ben megadható (GET) változó alapján állítjuk (pl.: **p=login**). Ezután újra betöltjük az oldalt, aminek a tartalma dinamikusan megváltozik a **page** változó alapján.

Biztonsági okokból minden fájl elején meg van hívva egy ellenőrző függvény, ami nem engedi be a felhasználót az oldalra, ha nincs hozzá jogosultsága. Ilyen esetben visszaküldjük a főoldalra.

2.4. Az asztali alkalmazás felépítése és lényegesebb függvényei

Általános függvények:

- **Confirm(message, title = „Megerősítés”)**

Paraméterek:

- message – egy eldöntendő kérdés, amire választ várunk a felhasználótól
- title (opcionális) – a megjelenő ablak fejlécében található cím

Leírás:

- egy MessageBox-ot jelenít meg a paraméterként megadott értékek alapján
- a megjelenő ablakon egy „Igen” és egy „Nem” gomb található

- **Alert(message, title = „Könyvklub”, img = MessageBoxImage.None)**

Paraméterek:

- message: string – az üzenet, amit közölni szeretnénk a felhasználóval
- title: string (opcionális) – a megjelenő ablak fejlécében található cím
- img: MessageBoxImage (opcionális) – a megjelenő ablakon lévő ikon, ami utal az üzenet jellegére (információ, hiba stb.)

Leírás:

- egy MessageBox-ot jelenít meg a paraméterként megadott értékek alapján
- a megjelenő ablakon csak egy „OK” gomb található

- **IsNumberInput(input)**

Paraméterek:

- input: string – a szöveg, amit ellenőrizni akarunk

Leírás:

- arra szolgál, hogy ha csak számokat akarunk elfogadni a felhasználótól egy beviteli mezőben, akkor ezzel a függvénnyel ellenőrizhetjük a beírt szöveget
- amennyiben csak számokat tartalmaz a paraméterként kapott szöveg, akkor „true” értéket ad vissza, különben pedig „false” lesz a visszatérési értéke

Adatbázis kapcsolat:

Az adatbázissal történő kommunikációért szerver oldalon egy API felel, kliens oldalon pedig egy **ApiRequest** nevű objektum, ami **RestClient** segítségével csatlakozik az API-hoz.

Ezen felül pedig létrehoztunk még egy **RequestType** nevű enum-ot, amivel a későbbiekben az API felé küldött kérések jellegét határozzuk meg. Ez lehet „Lekérdezés”, „Hozzáadás”, „Módosítás” és „Törlés”.

A kérések (request) sikerességét és a visszakapott eredményt az **ApiRequest** mezőin (property) keresztül érhetjük el.

Az **ApiRequest** objektum GET és POST kérések küldésére alkalmas. A létrehozásakor a konstruktorban megadott paraméterek alapján dönti el, hogy éppen milyen típusú kérést szeretnénk küldeni. Ezeket az alábbi metódusok használatával hajtja végre.

- **ProcessResponse(response)**

Paraméterek:

- response: RestResponse – a RestRequest kérés után kapott válasz

Leírás:

- a válaszként kapott státuszkód és eredmény alapján beállítja az **ApiRequest** objektum mezőinek értékét

- **GetRequest(key, value)**

Paraméterek:

- key: string – a GET változó elnevezése, ahogy az API-ban hivatkozunk rá
- value: string – a GET változó értéke

Leírás:

- küld egy GET kérést az API-nak, majd meghívja a **ProcessResponse** metódust

- **PostRequest(body)**

Paraméterek:

- body: object – egy általános objektumot vár, amely tartalmazza a kéréshez szükséges kulcs-érték párokat

Leírás:

- küld egy POST kérést az API-nak, majd meghívja a **ProcessResponse** metódust
- paraméterként kapott objektumot a következő módon határozhatjuk meg:
var obj = new { key1 = value1, key2 = value2 }

Modellek:

Ebben az esetben is az adatbázis tábláit vettük alapul az objektumok létrehozásánál, így ez könnyebb konvertálást is biztosít az API kérések eredményeiből, mivel az adatbázisban, a szerveren és a kliens oldali alkalmazásban is ugyanazon a néven hivatkozhatunk az azonos értékekre. Annyiban tér el a weboldal objektumaitól, hogy ebben az esetben létrehoztunk egy **OrderedBook** objektumot is, mivel itt külön is kezeljük a rendelésekhez tartozó könyveket.

Ezen kívül az alábbi modelleket hoztuk létre:

- **ImportFile**

Leírás:

- ez az objektum a **.csv** fájlok kezelésére szolgál, melyek segítségével egyszerre több terméket rögzíthetünk az adatbázisban

Metódusok:

- **ReadFile()** – beolvassa a kiválasztott fájlt, beállítja az **ImportFile** objektum mezőinek értékeit a fájl alapján, és feltölt egy **Product** objektumokat tartalmazó listát a fájlban található termékek adataival; egy bool típusú eredménnyel tér vissza a sikerességtől függően

ApiTools:

Az **ApiTools** egy statikus osztály, ami arra szolgál, hogy gyors kéréseket végezhessünk az API felé. Ez az osztály az alábbi publikus függvényt tartalmazza.

- **QuickRequest(values, requestType)**

Paraméterek:

- values: object – egy általános objektum, amelyet POST kéréshez használhatunk
- requestType: RequestType – a lekérdezés típusa

Leírás:

- egy POST kérést küld az API felé és egy bool értéket ad vissza a kérés sikerességétől függően

Frontend felépítése:

A program legnagyobb része 1 ablakon belül jelenik meg, aminek dinamikusan változtatjuk a tartalmát. Ez az ablak különböző komponensekből épül fel.

Az első komponens a menüsor, ahol kiválaszthatjuk, hogy az adatbázis melyik tábláját szeretnénk kezelni. A második pedig egy **Frame** elem, aminek a menü alapján változtatjuk a tartalmát.

A **Frame**-en belül mindig egy **Page** objektumot jelenítünk meg, ami mindig tartalmaz egy általunk létrehozott (szintén Page típusú) **Header** elemet, amiben megtalálható a **Keresés** és az adminisztrátorok kezelésére szolgáló gomb.

A **Frame**-en belül megjelenő elem minden esetben rendelkezik egy **interface** típusú ösztállyal, melyet **ITablePage**-nek neveztünk el. Ez az interface felelős azért, hogy minden oldal rendelkezzen egy **UpdateSource** és egy **Search** metódussal.

Az **UpdateSource** metódussal az aktuális oldal tartalmát frissíthetjük. A **Search** metódussal pedig a paraméterként kapott szöveg alapján kereshetünk az oldalon megjelenő rekordok között.

A bejelentkezés is ezen az ablakon történik, majd ezután állítjuk be az ablak méreteit a megjelenő táblázatokhoz.

Backend felépítése:

A backend nagy része az API irányába küldött kérésekből áll. Ezeket külön osztályokba rendeztünk, csakúgy, mint a weboldalnál. Az osztályok elnevezése mindig utal a tábla nevére, amiből lekérdezést szeretnénk végrehajtani az API-on keresztül, a végződése pedig minden esetben „Handler” (pl.: **UserHandler**).

Minden osztály tartalmaz egy privát **ObservableCollection<T>** típusú objektumot, amely elemeinek típusa a táblától függ. Ezek mellett pedig tartalmaznak egy publikus függvényt, ami visszaadja az **ObservableCollection<T>** tartalmát. Ez a függvény mindig a „Get” szóval kezdődik, a vége pedig szintén a táblára utal (pl.: **GetUsers()**).

A 3 osztály – amely oldallal rendelkezik – tartalmaz 1-1 keresés függvényt, amit minden esetben az aktuális oldal **Search** függvényében hívunk meg (amelyet az **ITablePage** interface-től örökölt).

A felhasználó (**User**) esetében ezeken kívül csak egy **törlés** és egy **módosítás** függvényre volt szükségünk.

A rendeléseknél (**Product**) létezik egy külön **ObservableCollection**, melyben **OrderedBook** objektumokat tárolunk, az aktuálisan kiválasztott termék alapján. Szintén tartalmaz egy **törlés** és egy **módosítás** függvényt. A módosítás itt annyiban más, hogy csak a rendelés állapotát módosíthatjuk. Valamint tartalmaz egy **UpdateQuantity** nevű függvényt, ami a rendelt könyvek mennyiségének módosítására szolgál és még egy törlés függvényt **DeleteOrderedBook** néven, amivel a rendelt könyvek közül törölhetünk.

A termékek esetében a **törlés** és a **módosítás** mellett van egy **hozzáadás** függvény **AddNewProduct** néven, amivel új terméket rögzíthetünk az adatbázisban. Valamint van egy **UploadProductFromFile** névre hallgató függvény, melyben létrehozunk egy **ImportFile** objektumot, a fájl alapján feltöltjük az új termékeket az adatbázisba, és végül visszatérünk egy bool típusú értékkel a sikerességtől függően. A hiba jellegéről felugró ablakkal tájékoztatjuk a felhasználót.

Az adminisztrátorok kezelésére szolgáló **AdminHandler** osztály tartalmazza az adminisztrátorok listáját, az ehhez tartozó lekérdezést végrehajtó **GetAdmins** függvényt, valamint egy **törlés** függvényt. Ezen kívül itt tároljuk az aktuálisan bejelentkezett admin, valamint itt küldjük el a bejelentkezéshez és a regisztrációhoz szükséges kéréseket az erre létrehozott 1-1 függvény segítségével.

A kérések eredményéül kapott JSON objektumok kezelésére a **JsonConvert** osztályt használtuk, amellyel könnyen átalakíthattuk ezt a megfelelő típusú listává.

Külön figyelmet fordítottunk arra, hogy bármilyen fennakadás esetén értesítsük a felhasználót a probléma jellegéről, illetve minden visszavonhatatlan művelet (pl.: törlés) előtt megerősítést kérünk a felhasználótól, a véletlenek elkerülése érdekében.

3. TESZTELÉS

3.1. PHP tesztelés

A PHP teszteléséhez a **PHPUnit** nevű külső könyvtárat használtuk, amellyel 2 fontosabb objektumot teszteltünk: a **DbConnection**-t, ami az adatbázis-kapcsolatért felel, és az **Encryption**-t, amely a jelszavak titkosításánál van segítségünkre.

Mivel az adminisztrációs felülethez készített API az adatbázis-kezelő objektumokat használja, ezért azok ellenőrzését az API végpontok tesztelésénél hajtottuk végre.

PHPUnit használatánál minden objektum tesztelésénél egy új osztályt hozunk létre, ami megvalósítja a **TestCase** absztrakt osztályt, és ezen keresztül érjük el a teszteléshez szükséges metódusokat.

A tesztjeinknek próbáltunk olyan neveket adni, amikből a futtatáskor pontosan tudni fogjuk, hogy melyik teszt sikerült és melyik ütközött valamilyen hibába. Ezt az alábbi 2 kép jól szemlélteti. A felső képen az egyik teszt kódja látható, az alsón pedig a teszt eredménye.

```
public function testQueryWithValidSql()
{
    $conn = new DbConnection();
    $sql = "SELECT * FROM admins WHERE username LIKE 'Trepyn';";
    $result = $conn->query($sql)->fetchObject();

    $this->assertEquals("Trepyn", $result->username);
}
```

```
Db Connection
✓ Query with valid sql
```

3.2. PHP tesztesetek

Művelet	Eredmény	Teszt eredmény
Lekérdezés létező adatokkal	Érkezett válasz az adatbázistól, létezik ilyen felhasználó	Sikeres
Lekérdezés nem létező adatokkal	Érkezett válasz az adatbázistól, nem létezik ilyen felhasználó	Sikeres
Jelszó ellenőrzés megfelelő jelszóval	A jelszó egyezett, igaz értéket kaptunk vissza	Sikeres
Jelszó ellenőrzés hibás jelszóval	A jelszó nem egyezett, hamis értéket kaptunk vissza	Sikeres

3.3. API végpont tesztelés

Az API végpontok tesztelésére a **Postman** nevű alkalmazást használtuk. Ez segít a tesztesetek rendszerezésében, csoportosításában és tárolásában.

A teszteseteket 4 nagyobb részre bontottuk: Admins, Users, Book és Orders.

3.4. API végpont tesztesetek

Admins:

Művelet	Bemenet	Kapott eredmény	Teszt eredmény
Összes admin lekérdezése	POST változó „admins” néven	Lekérdezés sikeres, visszakaptuk az összes admint	Sikeres
Admin bejelentkezés hiteles adatokkal	Felhasználónév: Test Jelszó: 12345	Bejelentkezés sikeres, visszakaptuk a felhasználó adatait	Sikeres
Admin bejelentkezés hibás adatokkal	Felhasználónév: Test Jelszó: 54321	Bejelentkezés sikertelen, hibaüzenetet kaptunk vissza	Sikeres

Users:

Művelet	Bemenet	Kapott eredmény	Teszt eredmény
Összes felhasználó lekérdezése	POST változó „users” néven	Lekérdezés sikeres, visszakaptuk az összes felhasználót	Sikeres
Létező felhasználó keresése	Email: ta@trepy.hu	Keresés sikeres, visszakaptuk a keresett felhasználó adatait	Sikeres
Rendeléssel nem rendelkező felhasználó törlése	Email: test@trepy.hu	Törlés sikeres	Sikeres
Rendeléssel rendelkező felhasználó törlése	Email: ta@trepy.hu	Törlés sikertelen, hibaüzenetet kaptunk vissza	Sikeres

Books:

Művelet	Bemenet	Kapott eredmény	Teszt eredmény
Összes könyv lekérdezés	POST változó „books” néven	Lekérdezés sikeres, visszakaptuk az összes könyvet	Sikeres
Konkrét könyv keresése (feltételezzük, hogy csak 1 van belőle)	Keresés: „normális”	Keresés sikeres, visszakaptuk a „Normális vagy” című könyv adatait	Sikeres
Könyvek keresése (több is lehet belőle)	Keresés: „el”	Keresés sikeres, visszakaptunk minden könyvet, aminek a címében szerepel a keresett szöveg	Sikeres
Rendeléshez tartozó könyv törlése	ISBN: 9789636040284	Törlés sikertelen, hibaüzenetet kaptunk vissza	Sikeres

Orders:

Művelet	Bemenet	Kapott eredmény	Teszt eredmény
Összes rendelés lekérdezése	POST változó „orders” néven	Lekérdezés sikeres, visszakaptuk az összes rendelést	Sikeres
Létező rendelés módosítása	Azonosító: TA-20230515173241 Státusz: Elküldve	Módosítás sikeres	Sikeres
Nem létező rendelés módosítása	Azonosító: TA-202305151 Státusz: Elküldve	Módosítás sikertelen, hibaüzenetet kaptunk vissza	Sikeres
Rendeléshez tartozó könyvek keresése	Azonosító: TA-20230515173241	Keresés sikeres, visszakaptuk az összes könyvet, ami ehhez a rendeléshez tartozik	Sikeres

3.5. Frontend tesztelés

A frontend tesztelését folyamatosan végeztük, minden kisebb-nagyobb változtatás után ellenőriztük, hogy a weboldal minden állapotában úgy működik-e, ahogy szeretnénk.

Az oldal végső változatát leteszteltük több különböző böngészőben is, hogy mindenhol megfelelően jelenjen meg. Tesztelve lett Google Chrome, Mozilla Firefox, Microsoft Edge, Opera és Brave böngészőkben.

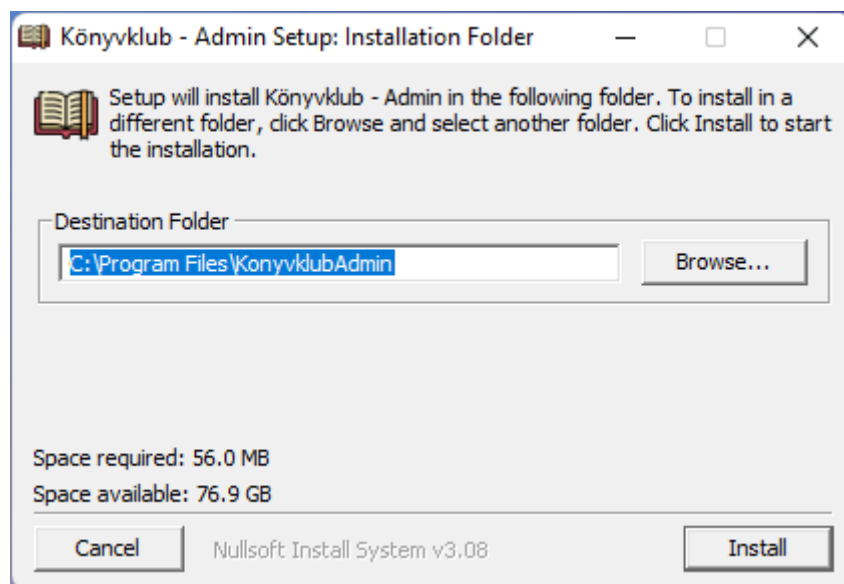
Emellett nagy hangsúlyt fektettünk a reszponzivitásra is, mivel ez a mai világban már elengedhetetlen. Ennek fényében teszteltük az oldalt több, különböző méretű és típusú eszközön, többek között asztali számítógépen, laptopon, tableten, valamint Android- és IOS rendszerű okostelefonokon.

4. ASZTALI ALKALMAZÁS FELHASZNÁLÓI DOKUMENTÁCIÓ

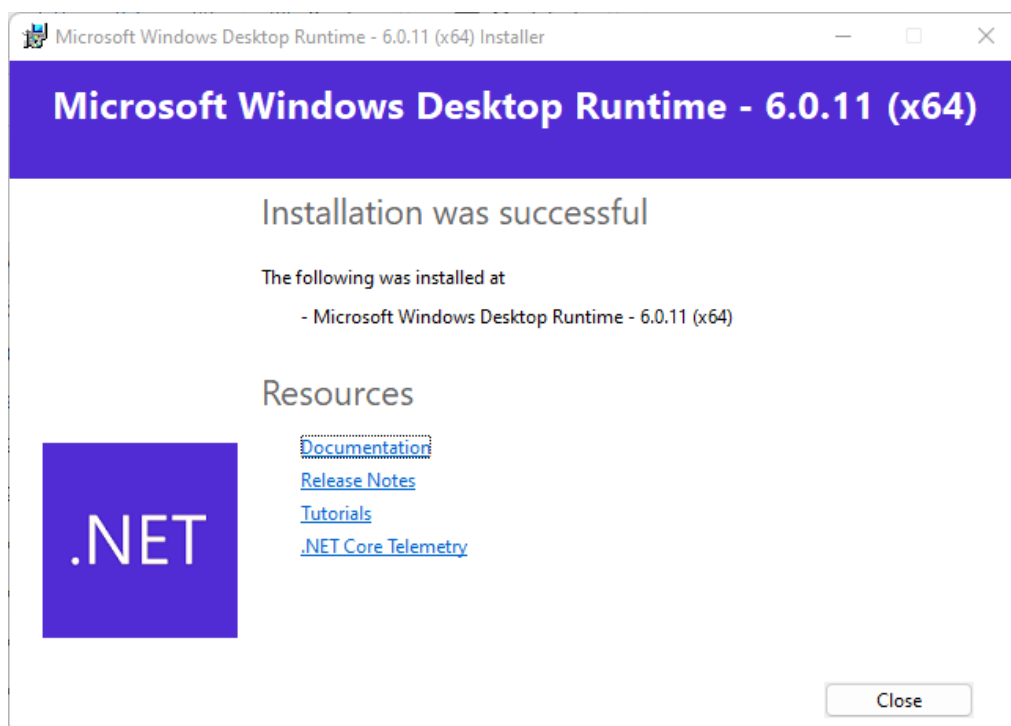
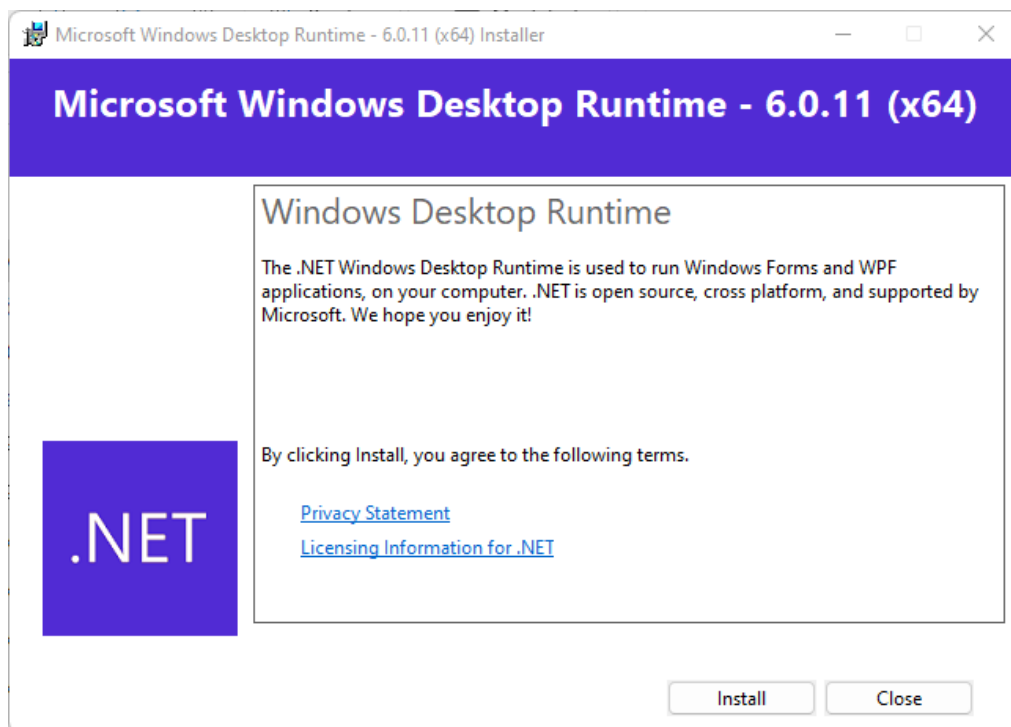
Rendszerkövetelmények		
	Minimum gépigény	Ajánlott gépigény
Processzor	Intel Pentium Dual-Core	Intel Core i3-7100
Memória (RAM)	4GB	8GB
Szabad tárhely	100MB	300MB
Rendszer	Windows 7	Windows 10, Windows 11
Internetkapcsolat	szükséges	szükséges

4.1. Program telepítése

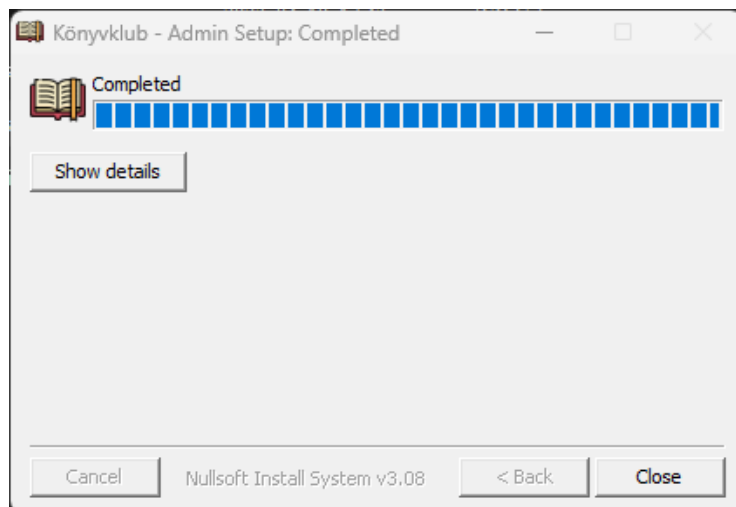
A telepítő elindítása után a következő ablak jelenik meg. Itt kiválaszthatjuk, hogy hova szeretnénk telepíteni a programot.



A telepítés elindítása után abban az esetben, ha nincs telepítve a .NET szükséges verziója, akkor megjelenik az alábbi ablak, amin az „Install” gombra kattintva telepíthetjük azt. Amint ezzel végezett a telepítő, a „Close” gombbal bezárhatjuk a .NET telepítő ablakát.



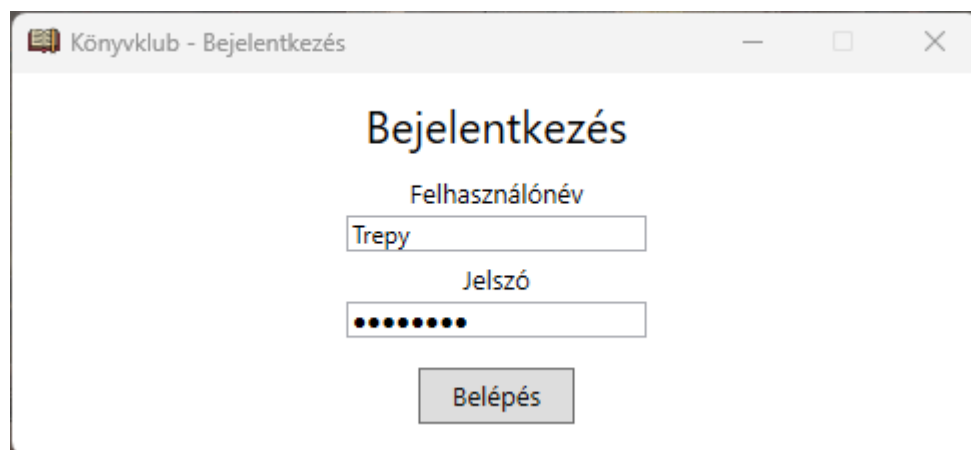
Végül amint befejeződött a program telepítése, bezárhatjuk a telepítőt a „Close” gomb megnyomásával.











A telepítő egy parancsikont is létrehoz az asztalon „Könyvklub - Admin” néven, amivel akár rögtön el is indíthatjuk a programot.

4.2. Program használata

























A program elindítása után a bejelentkezés képernyő fogad, ahol a megfelelő adatok megadásával beléphetünk az adminisztrációs felületre.












A program 3 fő részből áll, ebből az első a **Felhasználók** oldal, ahol a felhasználók adatait tekinthetjük meg és ezeket módosíthatjuk, vagy törölhetjük.

Könyvklub - Felhasználók					
Felhasználók			Rendelések		Termékek
<div>Keresés</div> <div>Reset</div>			Admin kezelés		
Email	Vezetéknév	Keresztnév	Cím	Telefon	
info@nytta.hu	Faludi	Anita	1234 Település, Utca	06123456789	 
minta@konyvklub-shop.hu	Minta	Márton	2230 Gyömrő, Táncsics utca 2.	06701234567	 
ta@trepy.hu	Trepák	Attila	2235 Mende, Andrássy utca 1	06704234569	 
viktorianagy77728@gmail.com	Nagy	Viktória	2230 Gyömrő, Andrássy utca 1	06701234567	 

A második nagyobb rész a **Rendelések** oldal, ahol a bal oldali táblázatban a rendeléseket, a jobb oldali táblázatban pedig a kiválasztott rendeléshez tartozó termékeket kezelhetjük. A módosítás gombbal csak a rendelés állapotát módosíthatjuk. A teljesítés gombbal a rendelés állapota teljesített állapotra vált és ezután az ehhez tartozó termékek már nem módosíthatók és nem törölhetők.

Könyvklub - Rendelések					
Felhasználók			Rendelések		Termékek
<div>Keresés</div> <div>Reset</div>			Admin kezelés		
ID	Email	Dátum	Rendelés állapota		
TA-20230508003239	ta@trepy.hu	2023-05-08 00:32:00	Elküldve	  	
VIKTORIANAGY77728-2023	viktorianagy77728@gmail.	2023-05-08 22:07:00	Feldolgozás alatt	  	
VIKTORIANAGY77728-2023	viktorianagy77728@gmail.	2023-05-09 00:19:00	Teljesítve	  	
MINTA-20230510201612	minta@konyvklub-shop.hu	2023-05-10 20:16:00	Feldolgozás alatt	  	
MINTA-20230510202519	minta@konyvklub-shop.hu	2023-05-10 20:25:00	Teljesítve	  	
TA-20230510233551	ta@trepy.hu	2023-05-10 23:35:00	Feldolgozás alatt	  	
VIKTORIANAGY77728-2023	viktorianagy77728@gmail.	2023-05-11 13:16:00	Feldolgozás alatt	  	
VIKTORIANAGY77728-2023	viktorianagy77728@gmail.	2023-05-12 12:01:00	Feldolgozás alatt	  	

ISBN	Mennyiség		
9789636357986	- 4	 	
9789636142162	- 2	 	
9789636040284	- 1	 	

A harmadik nagyobb rész pedig a **Termékek** oldal, ahol új termékeket rögzíthetünk akár manuálisan, akár egy pontosvesszővel elválasztott **.csv** fájlból. Meglévő termékek adatait és készleten lévő mennyiségét módosíthatjuk, valamint törölhetjük a kifutott termékeket.

Könyvklub - Termékek									
Felhasználók				Rendelések			Termékek		
Hozzáadás				Keresés			Reset		
							Admin kezelés		
ISBN	Cím	Szerző	Leírás	Kategória	Ár	Készleten	Rendelések	Kép	
9789636040284	A csend ereje	Sylvia Löhken	DR. SYLVIA LÖHKE	életmód	3570	9	6	sylvia-a_csend_ere	✖
9789636357986	A jakfarkas zászló	Leslie L. Lawrence	LESLIE L. LAWRENCE	kaland	3650	9	11	leslie-a_jakfarkas_z	✖
9789636142162	A kenyér lelke	Vajda József	"Meg kell adni az	gasztronómia	7590	7	7	vajda-a_kenyer_lel	✖
9789634862918	A két Lotti	Erich Kästner	A gyermekregény	ifjúsági	2850	4	1	kastner-a_ket_lotti	✖
9786155514333	A marsi	Andy Weir	"Hat nappal ezelőt	sci-fi	4240	16	4	weir-a_marsi.jpg	✖
9789630794688	A remény rabjai	Stephen King	"Mit tehet a fogoly	krimi	3740	12	2	king-a_remeny_rai	✖
9789635289981	A Titok	Rhona Byrne	A Nagy Titok törec	életmód	5690	12	0	rhona-a_titok.jpg	✖
9789635182336	Ahol megszakad	Závada Péter	A Moszkva tér koc	irodalom	1899	9	3	zavada-ahol_megs	✖
9786156471062	Az eltűnt remény	Dr. Bagdy Emőke	Jó ideje szinte egy	életmód	3290	9	1	bagdy-az_eltunt_n	✖
9789635660933	Az elveszett jelkép	Dan Brown	"Robert Langdönt,	krimi	5090	10	0	brown-az_elveszet	✖
9789636357559	Az utolsó hvárezm	Lőrincz L. László	A hvárezmi biroda	kaland	3420	14	1	lorincz-az_utolos_l	✖
9789634476054	Bábel	Frei Tamás	"Frei Tamás a színf	krimi	3995	15	0	frei-babel.jpg	✖
9789636358150	Bolondok kolostor	Leslie L. Lawrence	LESLIE L. LAWRENCE	kaland	4390	12	0	leslie-bolondok_kk	✖
9786150165912	Boron innen és túl	Varga Péter	A kis pincészetek j	gasztronómia	3420	18	0	varga-boron_innei	✖
9789635873760	Bot Benő	Julia Donaldson	Bot Benő egy bot-	ifjúsági	2840	15	0	donaldson-bot_be	✖
9789635663491	Dűne	Frank Herbert	Az univerzum legfő	sci-fi	5090	12	0	herbert-dune.jpg	✖
9789634068778	Én, a robot	Isaac Asimov	"A robotika három	sci-fi	2550	15	0	asimov-en_a_robo	✖
9789635096787	Féyn	Szendrói Csaba	Szendrói Csaba, di	irodalom	4690	20	0	szendroi-feyn.jpg	✖

A termék hozzáadása és módosítása ablakok az alábbi állapotban nyílnak meg.

Termékek

Új termék felvétele

ISBN:

Cím:

Író:

Leírás:

Kategória:

Ár:

Készleten:

Rendelések: 0

Kép:

OK

Termékek

Termék módosítása

ISBN:

Cím:

Író:

Leírás:

Kategória:

Ár:

Készleten:

Rendelések: 1

Kép:

OK

Az adminisztrátorokat az **Admin kezelés** gomb megnyomása után tekinthetjük meg. Itt törölhetünk meglévő adminisztrátorokat (kivéve az aktuálisan bejelentkezett admin), valamint regisztrálhatunk újat is.

Admin kezelés

Adminisztrátorok

Felhasználónév	
Anita	X
Trep	X
Viki	X
vizsga	X

Új admin hozzáadása

Regisztráció

Regisztráció

Felhasználónév

Jelszó

Jelszó megerősítése

Felvétel

Mind a 3 oldalon található egy **Keresés** mező. A **Felhasználók** között e-mail cím alapján, a **Rendelések** között a rendelés azonosítója alapján, a **Termékek** között pedig a könyvek címe alapján kereshetünk. A **Reset** gombbal újra lekérhetjük a teljes listát.

5. TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

5.1. Weboldal továbbfejlesztése

A weboldal terén 2 olyan funkció van, amik tervben voltak, de végül nem kerültek megvalósításra.

Ebből az egyik, hogy a főoldalon lévő könyveket lehessen rendezni ár, név, vagy esetleg népszerűség szerint. Ez a funkció kisebb-nagyobb átalakításokat igényelne az oldal navigációját illetően, ezért nem mertünk már belekezdeni az idő rövideje miatt.

A másik ilyen funkció pedig egy teljes profil oldal, ahol a felhasználó bármikor módosíthatná az adatait. Itt beszélhetünk akár címről, akár telefonszámról, vagy akár a fiókhoz tartozó jelszóról is. Továbbá ezen az oldalon lehetősége lenne a felhasználónak a profilja törlésére is, ha esetleg nem szeretné többet igénybe venni az oldal szolgáltatásait.

5.2. Adminisztrációs felület továbbfejlesztése

Az adminisztrációs felületen több továbbfejlesztési lehetőség is felmerült. Ilyen például a képek feltöltése. Jelenleg csak a weboldal FTP szerverén keresztül tudunk feltölteni képeket (pl. borítókat) az oldalra. Ezen lehetne változtatni, esetleg úgy, hogy új könyv rögzítésekor kiválaszthatnánk egy képet a számítógépünkről, illetve több könyv együttes feltöltése esetén többet is.

Ezen kívül még egy kijelentkezés gomb elférne az alkalmazásban, ha esetleg erre lenne igény, bár ez a fejlesztés során egyszer sem merült fel, mint tényleges hiányosság. Némi átalakítást igényelne, de ez egy viszonylag könnyen megvalósítható funkció lenne.

6. ÖSSZEGZÉS

A vizsgamunkát csapatban kellett elkészíteni és ez számunkra nagyon tanulságos volt, megtapasztaltuk milyen társainkkal együtt dolgozni. Ez azért is hasznos mert a későbbi munkáink során elengedhetetlen a csapatmunka egy sikeres projekthez.

A fejlesztés során felmerültek problémák, amiket volt, hogy csak napokkal később, hosszas utánajárás után sikerült megoldanunk. Ennek köszönhetően számos tapasztalattal gazdagodtunk, ami nagyban hozzájárult szakmai fejlődésünkhöz.

VIZSGÁZÓI NYILATKOZAT

Alulírott Trepák Attila, Nagy Viktória és Király Krisztián nyilatkozunk,
hogy a vizsgaremekünk saját, önálló munkánk eredménye.

Aláírás
Trepák Attila

Aláírás
Nagy Viktória

Aláírás
Király Krisztián

Gyömrő, 2023.