SZEGEDI SZC VASVÁRI PÁL GAZDASÁGI ÉS INFORMATIKAI TECHNIKUM

Az 5 0613 12 03 számú Szoftverfejlesztő és -tesztelő szakképesítés

záródolgozata

SkiRent

Sífelszerelés-kölcsönző és -kezelő szoftver

Készítette:

KOVÁCS JÓZSEF MIKLÓS

VARGA GÁBOR

Szeged

2025

Tartalomjegyzék

1. BEVEZETÉS

1.1 TÉMAVÁLASZTÁS INDOKLÁSA

A síelés és snowboardozás népszerű sportok, különösen a téli szezonban. A síkölcsönzőknek hatékony rendszerre van szükségük a felszerelések nyilvántartására, bérlésére és a foglalások kezelésére.

1.2 CÉLKITŰZÉS

Célunk egy könnyen kezelhető weboldal, platform létrehozása, amely lehetővé teszi a felhasználók számára a termékek böngészését és lefoglalását az általa kívánt időintervallumra és a foglalás részleteinek, állapotának megnézésére és követésére online.
Az asztali alkalmazást az admin felhasználóknak szánjuk, ahol egyszerűen és könnyen kezelheti a termékeket, termékkategóriákat, foglalásokat, felhasználókat és a termékekhez kapcsolódó képeket és számlákat.

1.3 KIKNEK SZÁNJUK A SZOFTVERT

A szoftvert minden olyan bérbeadónak szánjuk, akik szeretnék modern módon kezelni a foglalásokat és az elérhető sífelszereléseket és szeretnének a felhasználóknak egy kényelmes és egyszerűen kezelhető webesfelületet nyújtani arra, hogy leadják az igényüket egy-egy termékek kölcsönzésére.

2. FELHASZNÁLÓI DOKUMENTÁCIÓ

2.1 ASZTALI ALKALMAZÁS ÚTMUTATÓJA

Az asztali alkalmazás az admin (bérbeadók) számára készült. Az alkalmazásba vásárlóként nem lehet belépni, erre az alkalmazás fel is hívja a figyelmet, ha az illető megpróbálja.

2.1.1 Minimális Hardverkövetelmények

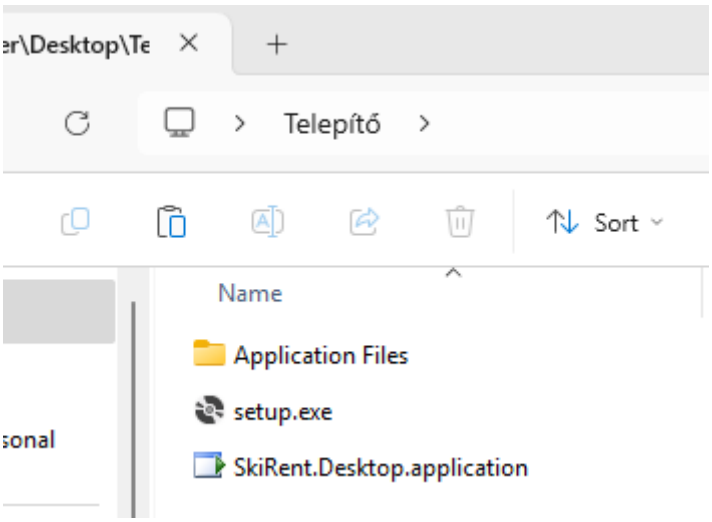
A kényelmes használathoz egy Windows 11-et futtatni képes számítógépet javaslunk.

- Processzor: 1 GHz vagy gyorsabb processzor, 2 vagy több maggal, 64 bites architektúra
- RAM: 4 GB
- Tárhely: 64 GB vagy több
- BIOS/Firmware: UEFI, Secure Bootra képes
- TPM: Trusted Platform Module (TPM) 2.0-ás verziója
- Videókártya: DirectX 12 vagy későbbit támogató kártya
- Képernyő: Nagyfelbontású (720p) vagy nagyobb képernyő

2.1.2 Telepítés

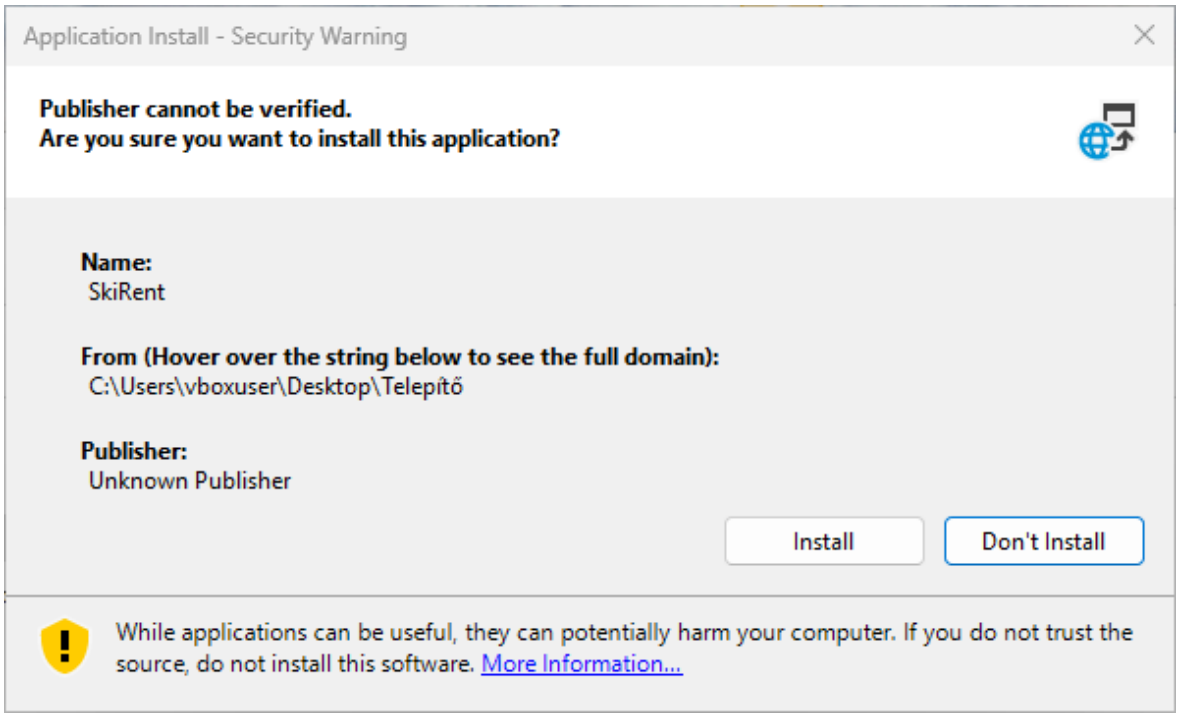
A felhasználó számára átadott telepítő mappában található setup.exe fájl futtatásával elindítható az asztali alkalmazás telepítése (1. kép).

1. KÉP
TELEPÍTŐ FÁJLOK



A telepítő elindítása után a telepítés (install) gombra kattintva elindíthatja a telepítést (2. kép).

2. KÉP
A TELEPÍTŐ ABLAKA

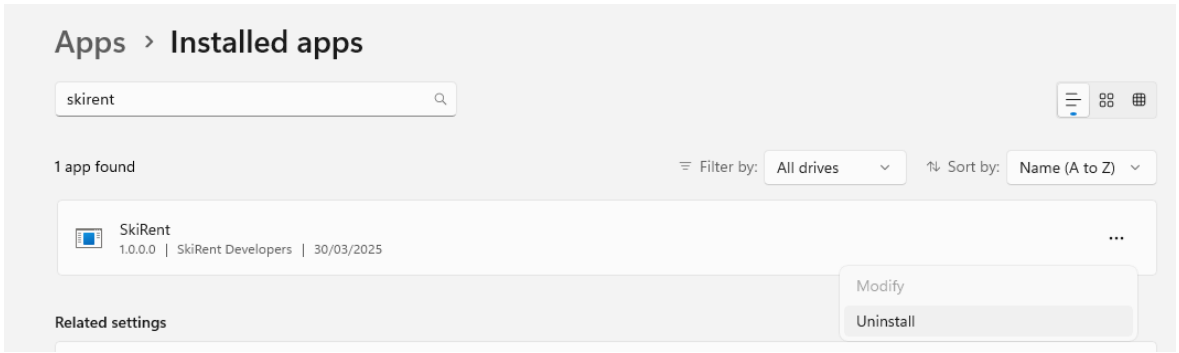


Ezután az alkalmazás sikeresen feltelepült és automatikusan el is indult. Az asztalon megtalálható egy SkiRent elnevezésű parancsikon is.

2.1.3 Az alkalmazás eltávolítása

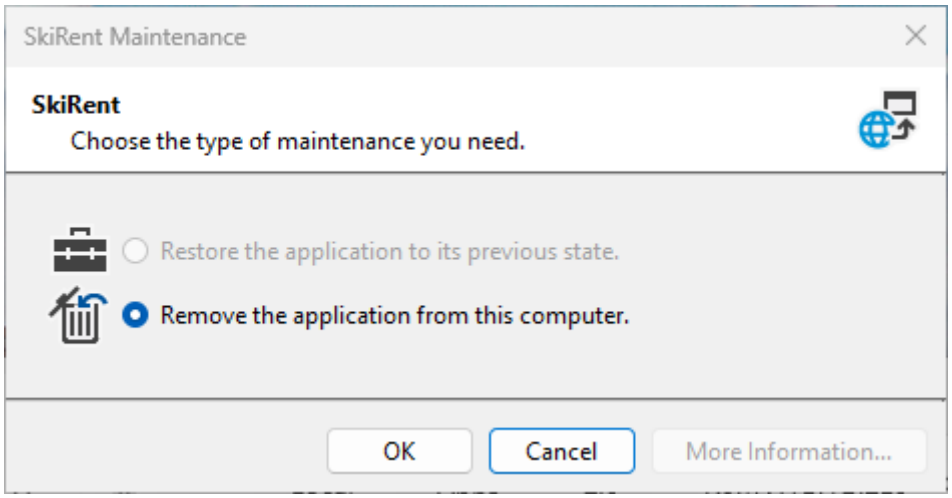
Ha valamilyen oknál fogva az alkalmazásunk eltávolítása mellett dönt, akkor ehhez a szokásos folyamatot kell alkalmaznia: beállításokban a telepített alkalmazásoknál az eltávolítás (uninstall) gombra kell kattintania (3. kép).

3. KÉP
SKIRENT ALKALMAZÁS ELTÁVOLÍTÁSA



Ezután elindul a SkiRent Maintenance program, ahol az OK gomb megnyomásával elindíthatja a törlést (4. kép).

4. KÉP
SKIRENT MAINTENANCE



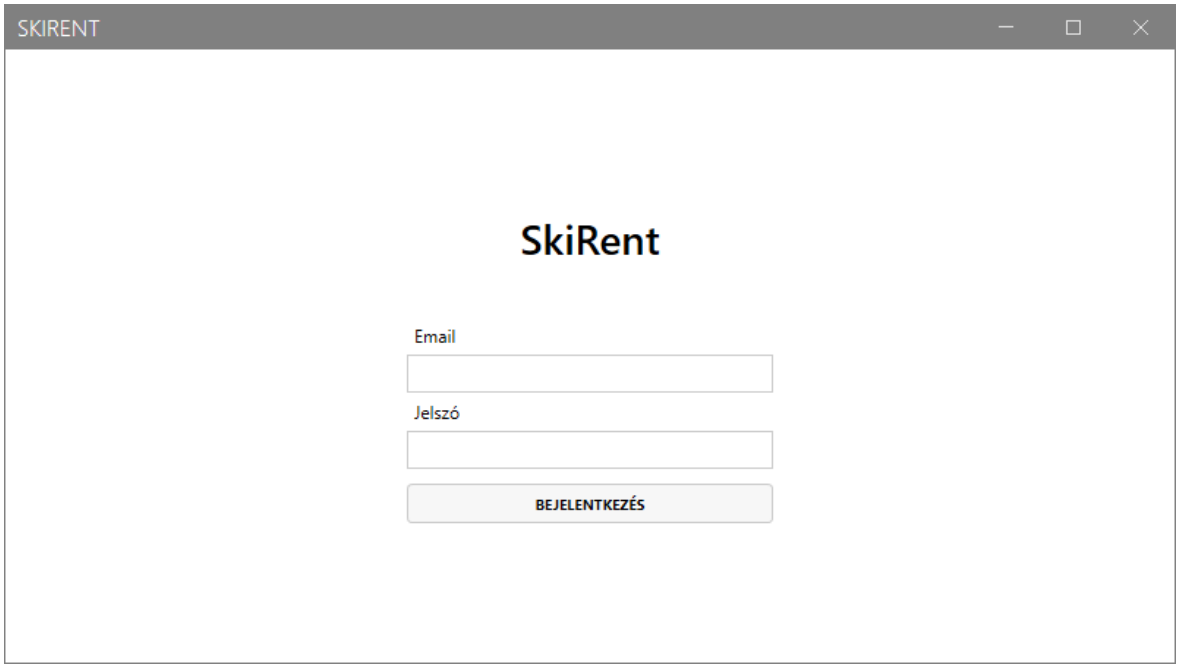
2.1.4 Szervercím módosítása

Ha a telepítő hibás szervercímet tartalmazna, nyissa meg a program mappáját, ahova feltelepült, majd egy szövegszerkesztővel az appsettings.json fájlt. Módosítsa a BaseUrl értékét a szerver IP-címére vagy domain nevére.

2.1.5 Asztali alkalmazásba való bejelentkezés

Ha elindítja az alkalmazást, akkor a bejelentkező felület fogja fogadni, ahol bejelentkezhet az admin felhasználói fiókba (5. kép). Amennyiben ez nem lett megváltoztatva másra az emailcím az admin@example.com, míg a jelszó Admin1234. A jelszó megváltoztatása erősen javasolt az első bejelentkezés után.

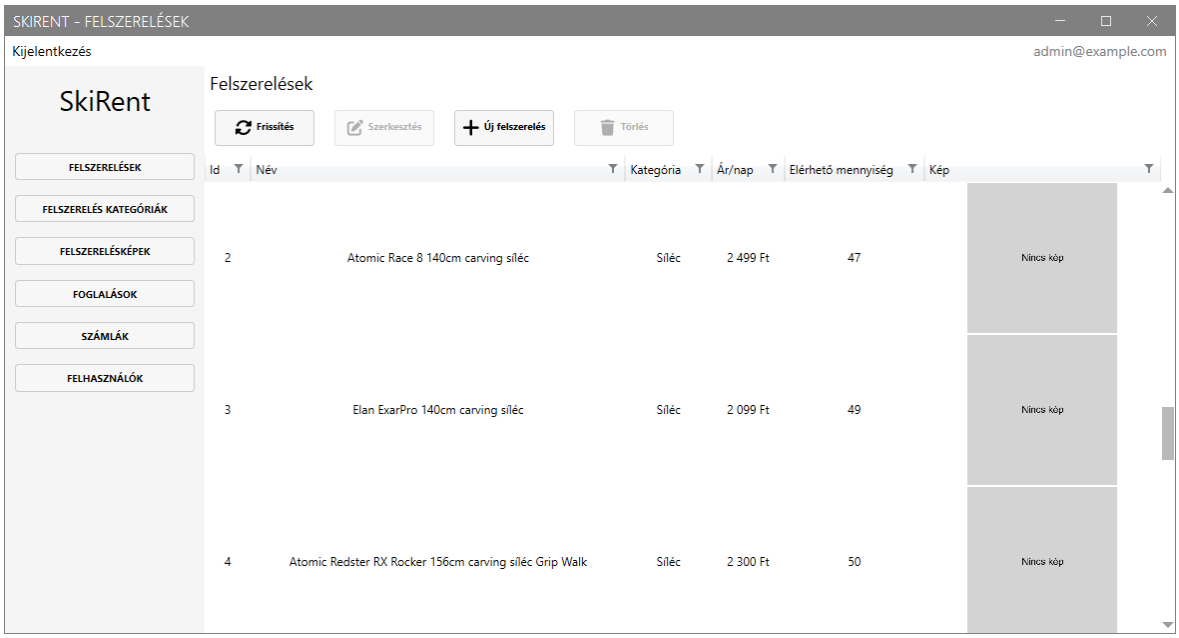
5. KÉP
ADMIN BEJELENTKEZŐ FELÜLET



Sikeres bejelentkezés után a felszerelések menüpontba kerül (6. kép).

Baloldalisávban találhatja a menüpontokat, amik közt válthat attól függően mit szeretne kezelni. A balfelső sarokban találhatja a kijelentkezés gombot, míg a jobbsarokban a bejelentkezett felhasználó emailcíme jelenik meg.

6. KÉP
FELSZERELÉSEK MENÜPONT

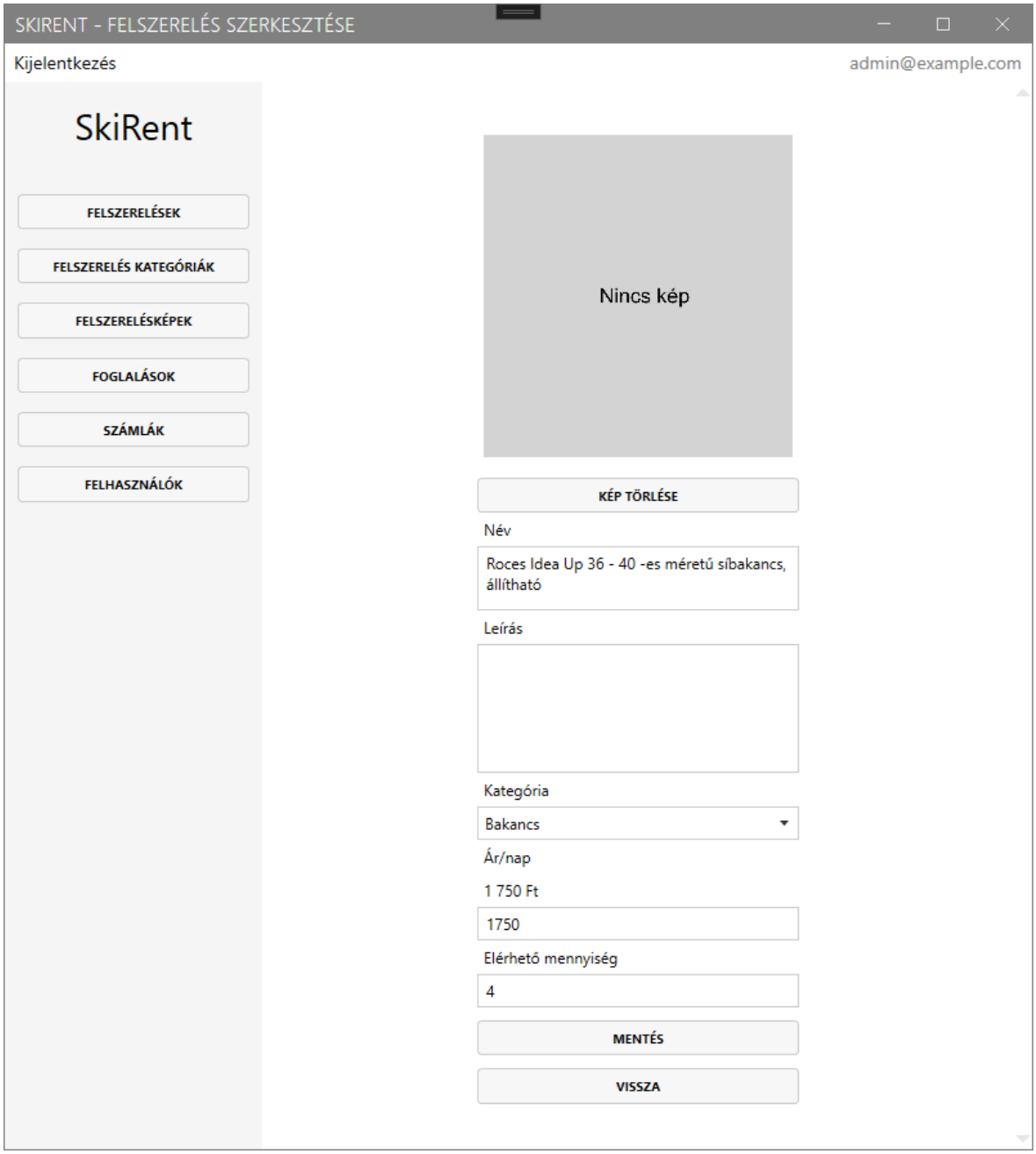


A felső középsősávban találhatóak az adott menühöz tartozó főbb gombok. Jelenesetben a frissítés gomb, amivel a szerverről letöltheti a legfrissebb felszerelés listát. A menüpontok váltásakor mindig leszedi az alkalmazás az adott legfrissebb listát, így evvel önnek nem kell foglalkoznia, csak akkor, ha manuálisan szeretné ezt a folyamatot elindítani.
A listában való egyik elem kijelölése után törölheti vagy szerkesztheti az adott elemet a megfelelő szerkesztés és törlés gombokkal. Az új felszerelés gombbal pedig egy új sífelszerelést adhat hozzá.
A lista tetején, az oszlopok címeinél található egy tölcsér (szűrő) ikon. Ha az egérmutatót fölé viszi, megjelenik egy kis fehér mező, ahová beírhat egy tetszőleges szöveget vagy számot a lista szűréséhez. A kis x ikonra kattintva törölhető a szűrési feltétel.
Az alkalmazásnál törekedtünk a hasonlókinézetre és elrendezésre, ezáltal gyorsabban elsajátíthatóvá válik. Majdnem minden menüpontnál megtalálhatóak a frissítés, szerkesztés, új és törlés gombok, a szűrési lehetősséggel együtt.

2.1.6 Új felszerelés létrehozása vagy szerkesztése

A 7. képen látható a szerkesztés felhasználói felülete.
Ha nincs kép hozzárendelve a felszereléshez, akkor a nincs kép felirat jelenik meg. Erre a feliratra/képre rányomva kilehet választani a feltöltöttképek közül, hogy melyik legyen a termék képe. Egy kép több termékhez is tartozhat.

7. KÉP
A FELSZERELÉS SZERKESZTÉSE



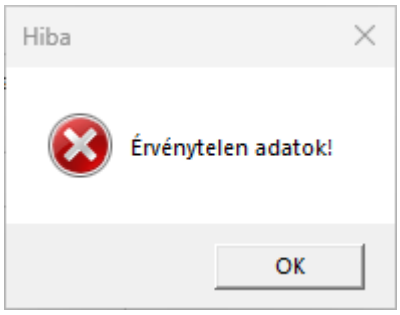
Az alábbi ellenőrzések történnek a tényleges mentés előtt, mikor a felhasználó a mentés gombra kattint:

- Név: minimum 1 karakter, maximum 100 karakter hosszúságú lehet.
- Leírás: maximum 4000 karakter hosszú lehet.
- Kategória: A felszerelésnek kötelezően tartoznia kell egy adott kategóriába.

- Ár/nap: Az ár 0 vagy ennél nagyobb szám lehet.
- Elérhető mennyiség: 0 vagy nagyobb szám lehet

A mentés gombra kattintva elmenthetők a módosítások, míg a vissza gombbal a változtatások nem kerülnek mentésre. Ilyenkor az alkalmazás figyelmeztető üzenetet jelenít meg, hogy a felhasználó biztosan mentés nélkül szeretne-e visszalépni. Ha valamelyik adat helytelenül van megadva, az alkalmazás hibaüzenetet jelenít meg (8. kép).

8. KÉP
ÉRVÉNYTELEN ADATOK HIBAÜZENET



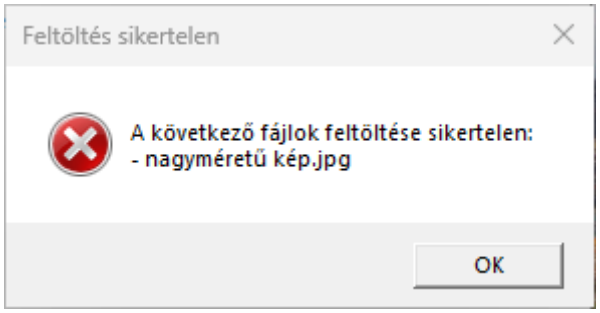
2.1.7 Felszerelés kategóriák

Kategória létrehozásánál vagy szerkesztésénél a kategória neve maximum 100 karakter hosszúságú lehet és nem szerepelhet már az adatbázisban.
Egy kategória csak akkor törölhető, ha már nem kapcsolódik hozzá egyetlen egy felszerelés sem.

2.1.8 Felszerelésképek

Az új kép gombra kattintva kiválaszthatóak a képek, amiket felszeretne tölteni a szerverre, ha netán valamelyik kép nem lenne megfelelő, akkor az alkalmazás egy hibaüzenetben megjeleníti mely fájlokkal volt gond (9. kép).

9. KÉP
KÉPFELTÖLTÉS HIBAÜZENET



A képnek megkell felelnie az alábbi követelményeknek:

- Csak a JPG fájlformátum támogatott.
- A fájlnév nem lehet hosszabb 255 karakternél.
- A fájl mérete nem haladhatja meg a 100 kilobyte-ot.
- A kép mérete nem lehet kisebb 150×150 pixelnél, és nem haladhatja meg a 750×750 pixelt.

A feltöltött fájl neve alapból a feltöltött fájlnévvel lesz megegyező, ez utólag módosítható tetszőleges névre a szerkesztés gombbal.
Egy kép csak akkor törölhető, ha már nem kapcsolódik hozzá egyetlen egy felszerelés sem.

2.1.9 Foglalások

A foglalások szerkesztésénél csak az állapot módosítható.
Ha az állapot „Kifizette” állapotban van, akkor módosítható „Kiszállítás alatt” vagy „Átvette” állapotra.
Ha a jelenlegi állapot „Kézbesítés alatt”, akkor csak az „Átvette” állapotra módosítható.
Ha a jelenlegi állapot „Átvette”, akkor csak a „Visszahozva” állapotra módosítható.
A tételek megtekintése gombra kattintva megtekinthetőek, hogy a felhasználó milyen tételeket foglalt le, milyen mennyiségben és ezek egyenként mennyi összegbe került.
Egy foglalást csak akkor törölhető, ha az állapota „Törölve” vagy „Visszahozva”.

2.1.10 Számlák

A számlák menüpontban megtekinthetőek a számla részletei és maga a számla, ami PDF formátumban van. A megtekintés gombra kattintva letöltődik és automatikusan megnyílik az alapértelmezett PDF olvasóval, ami a felhasználó számára bevan állítva.

Amennyiben a felhasználó törölte a fiókját a számla továbbra is elérhető és megtekinthető marad.

2.1.11 Felhasználók

Új felhasználó létrehozásakor csak egy véletlenszerű, biztonságos jelszó generálható. Ha a felhasználó egy konkrét jelszót szeretne beállítani, azt a szerkesztés gombbal tehető meg. Egy felhasználói fiókot addig nem lehet törölni, amíg van folyamatban lévő foglalása. Ez minden olyan foglalást jelent, aminek az állapota nem „Törölve” vagy „Visszahozva”.

2.2 BACKEND BEÁLLÍTÁSA, ÚTMUTATÓ

A backend biztonságos beállítása összetett feladat és az egyéni igények is sokfélék lehetnek. Ezért ez az útmutató csak röviden foglalja össze a legfontosabb lépéseket.

2.2.1 Szoftverkövetelmények

- Tetszőleges HTTP szerver, pl. Apache 2.4.58
- MariaDB 10.4.32
- PHP 8.2.12
- .NET 8
- Egy tetszőleges adatbáziskezelő kliens, pl. phpMyAdmin 5.2.1
- Windows vagy Linux operációsrendszer

A fenti követelmények többségét a XAMPP szoftvercsomag teljesíti, azonban a .NET környezetet külön kell telepíteni.

A SkiRent.Web tartalmát a *C:\xampp\htdocs* mappába kell helyezni, miután a tartalmát töröltük. Ha ez probléma, akkor egy tetszőleges almappában is el lehet helyezni.
Az adatbázis fájlokat a SkiRent.Database tartalmazza. A schema.sql fájlt futtatva létrehozhatjuk a táblákat és a SkiRent elnevezésű adatbázist.
A seed.sql tartalmaz némi példa, demo adatokat, ez tetszőlegesen módosítható.
Tartalmaz két felhasználót:

- Email: admin@example.com, jelszó: Admin1234
- Email: teszt@example.com, jelszó: Teszt1234

Az admin felhasználó admin jogosultságokkal rendelkezik, míg a másik csak egy általános felhasználói fiók (vásárló).

A SkiRent.Api appsettings.json-ban módosítsuk a ConnectionStrings:Default értékét, ha nem felelne meg az adatbázisunknak megfelelően.

Az AppSettingsben a BaseUrl a szerver elérési útvonala. Ezt módosítsuk a használni kívánt domain névre, ha van ilyen, ellenkező esetben pedig a szerver IP-címére, amelyen a Web API fut. Az API ezt a címet használja a FakePay-nél a callback elérési útvonalának megadására.

A DataDirectoryPath-ot pedig állítsunk be egy mappa elérési útvonalára, amiben az adatokat tárolni kívánjuk. Visszaperjel esetén duplázzuk meg őket.

A PaymentGateway-ben a BaseUrl az a cím, amin a SkiRent.FakePay szolgáltatásunk fut, ezt is módosítsuk a kívánt domainre vagy IP címre.

3. FEJLESZTŐI DOKUMENTÁCIÓ

3.1 ALKALMAZOTT FEJLESZTŐI ESZKÖZÖK

A sífelszerelés-kölcsönző és -kezelő szoftverünkhöz különböző eszközöket használtunk, amelyek segítenek a gyors fejlesztésben és a lehetséges hibák korai észlelését és kiküszöbölését. A backendhez és az asztali alkalmazásunkhoz a C#-ot választottuk nyelvnek a .NET keretrendszert használva, fontos szempont volt, hogy ezt mindketten ismerjük különböző mélységekben. A C# egy objektum-orientált, erősen típusos nyelv, amely segíti a fejlesztőt abban, hogy mindenhol a megfelelő típust használja. Ennek köszönhetően a hibák egy részét már a kód írásakor vagy fordításakor fel lehet fedezni, amelyek máskülönben csak futásidőben derülnének ki.

A .NET és a C# jelentős múltra tekint vissza, széleskörű fejlesztői közösség támogatja, és számos közösség által fejlesztett csomag érhető el hozzá.

Adatbázisnak a MariaDB-t választottuk, míg a frontend PHP-t használ, ezért szükség van egy tetszőleges HTTP szerverre is. A XAMPP alkalmazáscsomag tartalmazza a MariaDB-t, a PHP-t és az Apache HTTP Servert, így az alkalmazásunkhoz szükséges környezetet biztosítja.

A forráskódot Git verziókezelővel követjük, míg a kódok megosztására és hosztolására a GitHubot használjuk. A szoftverfejlesztéshez és a hibakereséshez a Visual Studio-t és a Visual Studio Code-ot használtuk, kiegészítve a SonarQube IDE (korábbi nevén SonarLint) kiegészítővel, amely segít a kód minőségének és biztonságának ellenőrzésében.

A dokumentáció elkészítéséhez és szerkesztéséhez a Microsoft Word dokumentumszerkesztő programot használtuk, míg a diagramok és ábrák létrehozására a Draw.io (Diagrams.net) alkalmazást vettük igénybe.

3.2 ADATBÁZIS

A src/SkiRent.Database útvonalon található a schema.sql és seed.sql fájl. Az előbbi az adatbázis táblákat, triggerek létrehozását tartalmazza, míg az utóbbi példa, demo adatokkal tölti fel az adatbázist. Az adatbázis neve SkiRent, UTF8 Unicode kódolással.

A táblák tartalma:

- Users: A regisztrált felhasználókat tartalmazza, beleértve az adminokat és a vásárlókat is.

- EquipmentCategories: A termékkategóriákat tartalmazza.
- Equipments: A termékeket tartalmazza. A nevét, a termékleírását, a naponkénti árát és az elérhető mennyiséget és a hozzákapcsolódó termékkép.
- EquipmentImages: A termékekhez kapcsolódó képek nevét tartalmazza, Maguk a képek a fájlrendszeren vannak tárolva, és a fájlokat az egyedi Id azonosítóval megegyező néven kerülnek mentésre.
- Bookings: A foglalásokkal kapcsolatos adatokat tartalmazza, mint a teljesár, a foglalás státusza, a kezdő és vég dátum, fizetésiazonosító és melyik felhasználóhoz tartozik
- BookingItems: A foglalt termékek adatait tartalmazza, beleértve a foglalás időpontjában érvényes nevet, árat és a lefoglalt mennyiséget.
- Invoices: A számlához kapcsolódó azonosítókat, összeköttetéseket tartalmaz. Melyik foglaláshoz és melyik felhasználóhoz tartozik az adott számla. Maguk a generált PDF formátumú fájlok (számlák) a fájlrendszeren tárolódnak az egyedi Id azonosítóval megegyező néven kerülnek mentésre.

A táblák között idegenkulcs-kapcsolatok vannak kialakítva, és törekedtünk az adatbázis normalizálására és arra, hogy az adatok optimális tárhelyet használjanak, elkerülve a felesleges kapacitásfoglalást. Figyeltünk arra, hogy ne lehessen olyan adatot törölni, amelyre más táblák hivatkoznak.
A táblákról részletesen:
Users tábla:

- Id: A felhasználó egyedi azonosítója.
- Email: A felhasználó emailcíme.
- PasswordHash: A felhasználó jelszava sózva és hashelve.
- UserRole: A felhasználó szerepköre vagy jogköre.
    - Admin: Admin jogosultságok.
    - Customer: Vásárlói jogosultságok.

EquipmentCategories tábla:

- Id: A felszerelés kategória egyedi azonosítója.
- Name: A felszerelés kategória neve.

Equipments tábla:

- Id: A felszerelés egyedi azonosítója.
- Name: A felszerelés neve.
- Description: A felszerelés termékleírása.
- CategoryId: Idegenkulcs a felszerelés kategória táblára.
- PricePerDay: A felszerelés napi bérleti díja, amelyért a termék bérelhető vagy kölcsönözhető.
- AvailableQuantity: A jelenleg elérhető mennyiség az adott felszerelésből.
- MainImageId: Idegenkulcs a felszereléskép táblára.

EquipmentImages tábla:

- Id: A kép egyedi azonosítója.
- DisplayName: A kép címe, neve.
- CreatedAt: A kép feltöltésének időpontja.
- UpdatedAt: A kép adatainak utolsó módosításának időpontja.

Bookings tábla:

- Id: A foglalás egyedi azonosítója.
- UserId: Idegenkulcs a felhasználók táblára.
- StartDate: A foglalás kezdete.
- EndDate: A foglalás vége.
- TotalPrice: A foglalás teljes ára, végösszege.
- PaymentId: A fizetés egyedi azonosítója.
- Status: A foglalás aktuális állapota.
    - Pending: A foglalási igény beérkezett.
    - Paid: A foglalást kifizették.
    - InDelivery: A termékek kiszállítása megkezdődött.
    - Received: A bérlő átvette a termékeket
    - Cancelled: A foglalás meghiúsult (pl. lejárt a fizetési határidő).
    - Returned: A bérlő visszavitte a termékeket a bérbeadónak.
- CreatedAt: A foglalás létrehozásának időpontja.
- UpdatedAt: A foglalás adatainak utolsó módosításának időpontja.

BookingItems tábla:

- Id: A foglaláshoz tartozó termék egyedi azonosítója.
- BookingId: Idegenkulcs a foglalások táblára.
- EquipmentId: A foglaláshoz tartozó felszerelés egyedi azonosítója (nem hivatkozik közvetlenül a felszerelések táblára).
- NameAtBooking: A foglalás időpontjában érvényes név az adott felszerelésre.
- PriceAtBooking: A foglalás időpontjában érvényes ár (napi bérleti díja) az adott felszerelésre.
- Quantity: Az adott termékből lefoglalt, bérelt mennyiség.

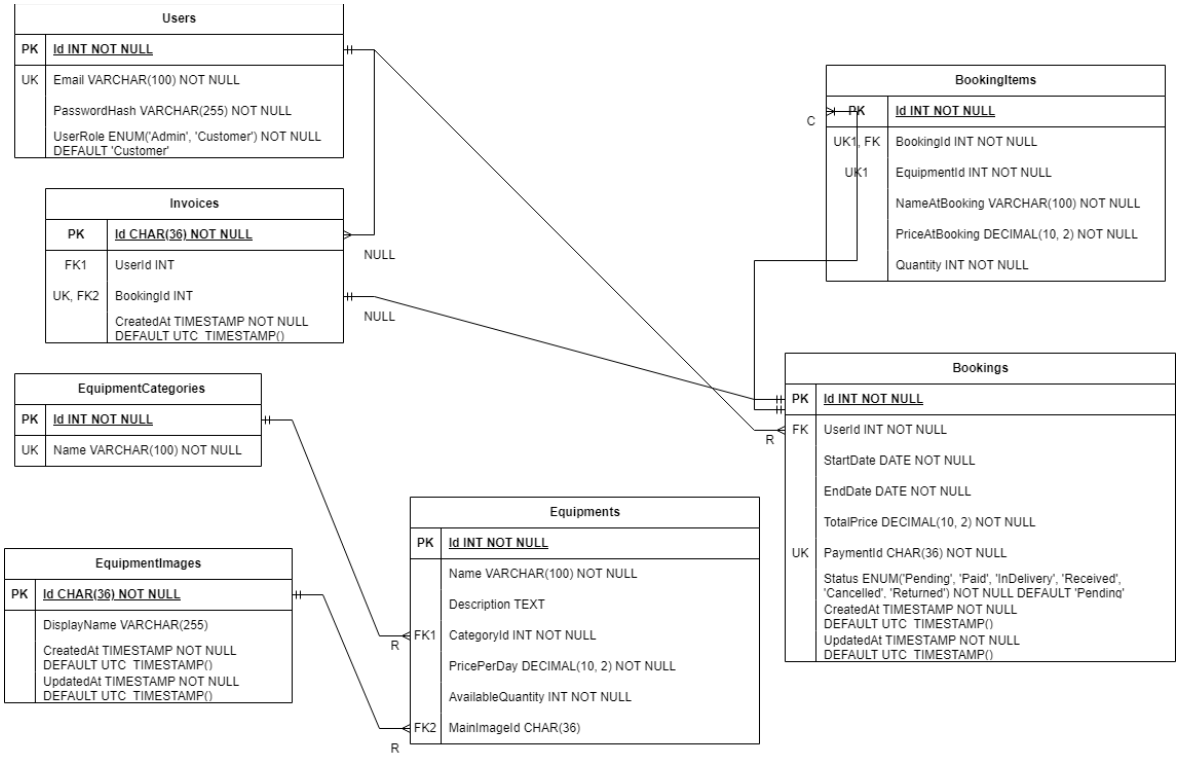
Az BookingId és a EquipmentId egy összetettkulcsot alkotnak, ezáltal biztosítva, hogy egy foglaláson belül csak egyszer szerepelhet az adott termék.

Invoices tábla:

- Id: A számla egyedi azonosítója. Egyszerűsítés céljából a PaymentId szolgál számlaazonosítóként, amely egy GUID/UUID érték.
- UserId: Idegenkulcs a felhasználók táblára.
- BookingId: Idegenkulcs a foglalások táblára.
- CreatedAt: A számla létrehozásának időpontja.

Az adatbázishoz készült egy adatbázisterv a fejlesztés folyamán ez az ábrán látható (1. ábra).

1. ÁBRA
SKIRENT ADATBÁZISTERV



3.3 TESZTEK

A tests mappa tartalmazza az automatizált és manuális teszteket. Az automatizált tesztekhez NUnit tesztkeretrendszert használunk, a tesztadatok létrehozásához pedig az AutoFixture csomagot, ezzel lerövidítve az Arrange szakasz előkészítéséhez szükséges időt.

Az integrációs tesztek futtatása előtt létre kell hozni az Images mappát abban a könyvtárban, amely az adatokat, például képeket és számlákat tároljuk. A másik lehetőség az, hogy az alkalmazást egyszer el kell indítani, így a szükséges mappák automatikusan létrejönnek.

Jelenleg csak az úgynevezett happy path esetek vannak tesztelve. A Stryker Mutator csomag mutációs teszt reportja alapján a tesztek a jövőben további bővítésre kerülnek.

3.4 FELTÖLTÖTT, LÉTREHOZOTT ADATOK HELYE

A src/SkiRent.Api/appsettings.json-ban az AppSettings-en belül a DataDirectoryPath-al lehet beállítani, hogy hova mentse az adatokat, mint például feltöltött képek, generált számlák. Ez alapértelmezettként null értékű, ami azt jeleni, hogy a felhasználó ideiglenes mappáját fogja felhasználni (a TEMP környezetiváltozó értéke). Ez fejlesztés alatt megfelelő lehet, míg éles helyzetben a DataDirectoryPath-ot célszerű beállítani egy olyan mappára,

amihez csak bizonyos személyek férhetnek hozzá, például a rendszergazda. Az értéknek a teljesútvonalnak kell lennie, visszaperjel használata esetén megkell kettőzni a perjelet.

3.5 KONFIGURÁCIÓS FÁJLOK, APPSETTINGS

A SkiRent.Api/appsettings.json fájlban 3 fontos fő rész található: ConnectionStrings, AppSettings, PaymentGateway.

A ConnctionStrings:Default beállításrészben lehet beállítani az adatbázis kapcsolati adatokat: adatbázisszerver címe, felhasználónév, jelszó, portszám.

AppSettings rész:

- MerchantName: A kereskedő (bérbeadó) neve, aki a szolgáltatást nyújtja.
- BaseUrl: Az API címe és portja.
- DataDirectoryPath: A számlák és képek mentési helye. Teljesútvonal formátumban. A null (hiányzó) értéknél az aktuális felhasználó ideiglenes mappája lesz használva.

PaymentGateway rész:

- BaseUrl: A PaymentGateway címe és portszáma.
- SharedSecret: Egy véletlenszerű karaktersorozat, amely a HMAC üzenet hitelesítéséhez használt titkos kulcs, amely biztosítja, hogy csak a jogosult felek tudják ellenőrizni az üzenet integritását és hitelességét. Ebben az esetben a PaymentGateway és a SkiRent.Api.

  A SkiRent.FakePayben levő appsettings.json fájlnál a Clients:SharedSecret-el megkell, hogy egyezzen!

A SkiRent.Desktop/appsettings.json-ben AppSettings:BaseUrl-ben a Web API címét lehet beállítani, amihez az asztali alkalmazás kapcsolódni fog.

3.6 WEB API (SKIRENT.API)

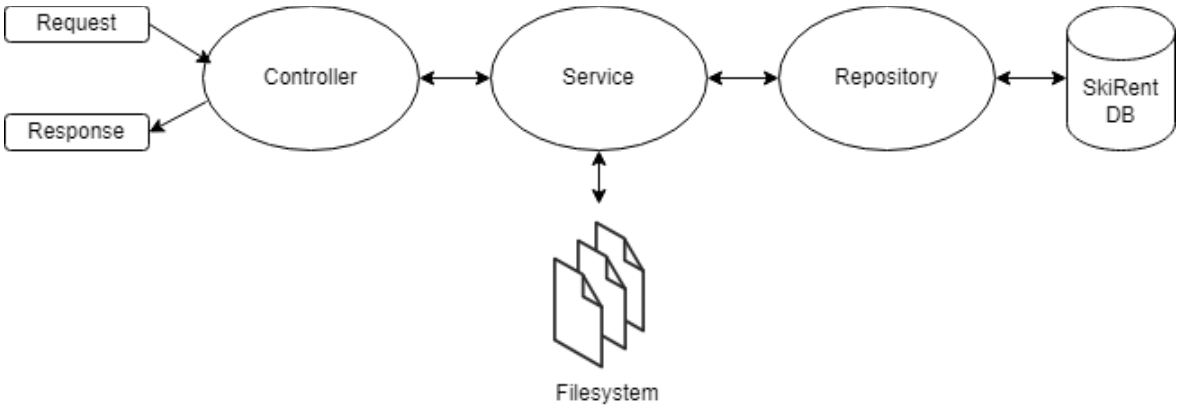
Az API elkészítésekor törekedtünk a RESTful architektúra elveinek betartására, követésére. A program az adatbáziskapcsolathoz a Pomelo.EntityFrameworkCore.Mysql csomagot használja. A PDF formátumú számlák generálásához a QuestPDF csomagot vettük igénybe. A kód tartalmaz egy globális exception handlert a nem várt kivételek elkapására és 500-as HTTP-kóddá alakítására. A hiba naplózásra kerül, amelyet a fejlesztő fel tud használni a javításhoz. A felhasználói jelszavak a Microsoft.AspNetCore.Identity névtér alatt található

PasswordHasher osztállyal kerülnek hashelésre és ellenőrzésre, PBKDF2-t használva. Az osztály automatikusan gondoskodik a jelszó sózásáról is, így ezzel külön nem kell foglalkoznunk, és nincs szükség a só külön tárolására az adatbázisban.
A szerver az időket UTC formátumban tárolja és úgy is várja. A kliens oldali adatok esetében az időpontokat a felhasználó helyi időzónájából UTC-re kell konvertálni küldés előtt. A szerver a visszaküldött időket automatikusan a felhasználó helyi időzónájába alakítja. Erre azért van szükség, hogy a fejlesztőknek ne kelljen foglalkozniuk azzal, hogy a szerver milyen időzóna-beállításokkal rendelkezik, illetve hogy a kliensek földrajzilag hol tartózkodnak.

3.6.1 Kódstruktúra

A kód követi az egy osztály per fájl elvet, a névterek pedig igazodnak a mappastruktúrához. A Controllers mappa a kontrollereket tartalmazza, törekedve arra, hogy ne tartalmazzanak üzleti logikát. A Base mappában található BaseController Problem metódusa alakítja át a service-ekből visszatérő hibákat (amennyiben vannak) a megfelelő HTTP-válasszá, részletes hibaszöveggel, amely jelzi a probléma okát.
Az API végpontjaihoz különböző jogosultságok és policy-k tartoznak. A policykhez kapcsolódó kódok az Authorization/Handlers mappában találhatók.
A Data mappa tartalmazza a Unit of Work-öt, a repository-kat és az adatbázismodelleket.
A repository-k egy generikus BaseRepository-ból öröklődnek, így a gyakran használt funkciókat nem kell újraimplementálni.
Az üzleti logika különböző service-ekre van bontva a Service mappán belül. A CRUD műveleteket támogató service-ek esetében törekedtünk az egységes metóduselnevezésekre (pl. CreateAsync, GetAsync, GetAllAsync, UpdateAsync, DeleteAsync), amennyiben az adott service-ben relevánsak.
A service-ek a Result mintát használják, a FluentResults csomagot alkalmazva. A várt hibák Error-ként kerülnek vissza a kontrollerekhez, ezáltal elkerülve a kivételek okozta teljesítményproblémákat és a felesleges stack trace generálást.
A Web API architektúrájáról egy áttekintés az ábrán látható (2. ábra).

2. ÁBRA
A WEB API ARCHITEKTÚRÁJA



3.7 ASZTALIALKALMAZÁS (SKIRENT.DESKTOP)

Az asztali alkalmazás követi a Model-View-ViewModel (MVVM) architektúrát, WPF-et és CommunityToolkit-et használva. A felhasználói felület alapstílusához és ikonjaihoz a MahApps.Metro csomagot használjuk. Az alkalmazás elsősorban DataGrid-ekben jeleníti meg a szervertől érkező adatokat, törekedve az oldalak közötti egységes megjelenítésre.

3.7.1 Kódstruktúra

A Services mappa tartalmazza a navigációval kapcsolatos service-t. Az alkalmazás egyetlen ablakot használ, amelyben minden megjelenő oldal egy UserControl, és a ContentControl jeleníti meg őket. A ContentControl tartalmának váltásáért a NavigateToAsync metódus felel, míg a ContentControl-ok közötti váltásért a SwitchTo metódus.

Ha egy ViewModel-t aszinkron módon szükséges inicializálni, akkor az IInitializeAsync generic interfészt kell implementálni, megadva a paraméterek típusát. Ha nem szükséges paramétert átadni, akkor navigáláskor az InitializeAsync alapból meghívódik a NavigationService-ben, egyéb esetekben egy aszinkron inicializáló függvényként kell átadni.

Az Utils mappa néhány segédosztályt tartalmaz, például az enumok felhasználó számára érthető formára alakítását a felhasználói felületen való megjelenítés előtt és visszafelé. Emellett itt találhatók osztályok a dátumok és a pénznem formázására, valamint egy véletlenszerű jelszógenerátort.

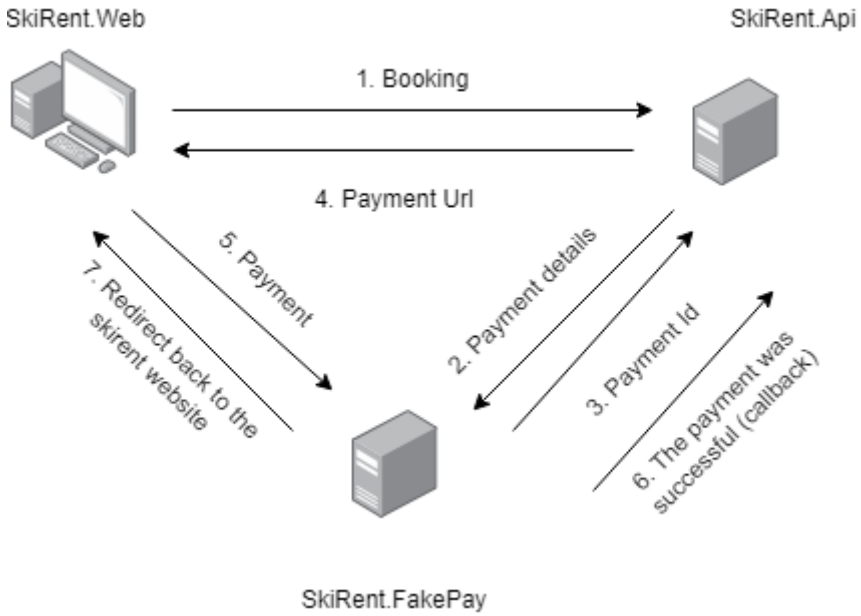
A ViewModels és Models mappák: A szervertől érkező válaszokat az asztali alkalmazás saját igényeihez igazított modellekké alakítjuk. A MainViewModel-ben a bejelentkezés során ellenőrzésre kerül, hogy a felhasználó rendelkezik-e a megfelelő admin

jogosultsággal. Amennyiben igen, a felhasználó adatai eltárolódnak az alkalmazás tulajdonságai között a memóriában (Application.Current.Properties[nameof(CurrentUser)]), így bármelyik osztályból elérhetők, ha később szükség van rájuk.
Az adatok elküldés előtt validáláson mennek keresztül, azonban jelenleg a felhasználói felületen nincs pontos visszajelzés arról, hogy mely mezőkkel van probléma és mi annak az oka. Az App.xaml.cs-ben egy globális exception handler (Current.DispatcherUnhandledException) kezeli a kivételeket. Amennyiben a hiba a FluentValidation-höz tartozik, az alkalmazás egy általános hibaüzenetet jelenít meg arról, hogy az adatok érvénytelenek. Ugyanez a handler felel azért, hogy az alkalmazás ne álljon le vagy omoljon össze váratlan hibák, például hálózati problémák miatt, függetlenül attól, hogy a hiba a kliens vagy a szerver oldalán jelentkezik.

3.8 FIZETÉS (SKIRENT.FAKEPAY)

A fizetéshez jelenleg egy demo fizetési oldal van létrehozva, mivel egy iskolai projektről van szó. A valódi fizetéshez integrálni kellene egy megfelelő fizetési vagy banki szolgáltatást, amelyhez adott esetben szerződéskötés is szükséges a szolgáltatóval. Ilyen szolgáltatók például a K&H, OTP Bank, Stripe és Barion. Fel kell mérni, hogy az adott szolgáltató biztosít-e SDK-t a szolgáltatásához és az nekünk megfelelő-e, vagy a fejlesztőknek kell azt az elérhető dokumentáció alapján implementálni.
A fizetés folyamatához készült egy ábra is a könnyebb érthetőségért (3. ábra).

3. ÁBRA
A FOGLALÁS (BÉRLÉS) ÉS FIZETÉS FOLYAMATA



3.9 MEGOSZTOTT OSZTÁLYKÖNYVTÁR (SKIRENT.SHARED)

A fejlesztés felgyorsítása és a felesleges duplikáció elkerülése érdekében az asztali alkalmazás és a Web API között megosztásra kerülnek a kérés- és válaszmodellek, a kérések validálására használt validátorok, valamint egy kliens a Web API eléréséhez.

3.9.1 Kódstruktúra

A Clients mappa tartalmazza a Web API-hoz tartozó klienst. A Refit csomag használatával elegendő csak az interfészeket megírni és néhány attribútummal ellátni a Refit automatikusan előállítja a szükséges implementációt.

A Contracts mappa tartalmazza a kérésekhez és válaszokhoz használt modelleket, funkciók szerint kategóriákra bontva. A Common mappa kivételével minden fájl követi a Create/Get/GetAll/Update elnevezési sémát. A kérések record típusúak, ami azt jelenti, hogy két record egyenlőnek számít, ha azonos típusokat és értékeket tárol. Emellett automatikusan generálódik hozzájuk egy ToString metódus, amely megkönnyíti a tesztelést: lerövidíti az Assert részt, és hiba esetén a fejlesztő számára olvasható formátumban jeleníti meg az adatokat.

A Validators mappa a kérésekhez tartozó validátorokat tartalmazza, funkciók szerint csoportosítva. A FluentValidation csomagot használva az osztályok elnevezése igazodik

a modellekhez, például CreateValidator és UpdateValidator. A képek validálásához az ImageSharp csomagot alkalmazzuk. Biztonsági okokból a maximálisan lefoglalható memória 32 MB-ra van korlátozva, hogy megelőzzük a szerver túlzott erőforrás-felhasználását képfeldolgozás során, és biztosítsuk a felhasználók zavartalan kiszolgálását.

4. TOVÁBBFEJLESZTÉSI TERVEK

Szeretnénk a teszteket mindenképp bővíteni, hogy a későbbi fejlesztések, karbantartások során is megbizonyosodhassunk az alkalmazás megfelelő működéséről. A dokumentáció további bővítése, különösen a kódok dokumentációját.

A jövőben kuponrendszert szeretnénk bevezetni, amely marketing szempontból is jelentős előnyt nyújthat.

Mivel a síelés nemcsak a helyiek, hanem a turisták körében is népszerű, célszerű lenne angol nyelvű verziót biztosítanunk a szoftverünkből - mind a webes, mind az asztali alkalmazásunkból, hogy más országokban is értékesíthető legyen. Ez a funkció magával vonná a több pénznem támogatását is.

A bérbeadók számára hasznos információ lehet a felhasználók és foglalások száma, valamint ezek napi alakulása. Emellett érdemes lenne kiemelni a legnépszerűbb termékeket, és ezeket egy vagy több diagramon megjeleníteni, hogy a bérbeadók a jövőbeli üzleti döntéseikhez felhasználhassák. Az adatok marketing szempontból is értékesek lehetnek.

Továbbá lehetőséget szeretnénk biztosítani olyan termékek eladására, amit csak egyszeri értékesítésre alkalmas.

Az asztali alkalmazásunkban a felhasználói élmény növelése érdekében pontosabb és részletesebb hibajelzésekkel szeretnénk bővíteni a jelenlegi, általános hibaüzeneteket. Emellett értesítésekkel és jobb átláthatósággal segítenénk a felhasználókat abban, hogy könnyebben nyomon követhessék, mely termékek elérhető mennyisége közelít a nullához. Továbbá kiemelnénk azokat a foglalásokat, amelyek fokozott figyelmet igényelnek.

A szoftverhez megfelelő fizetési szolgáltatás használata, mint például a Stripe vagy a Barion.

Manapság egyre inkább elterjedt a konténerizáció a szervereknél, mivel elősegíti a könnyebb telepíthetőséget és szállítást, ezért célszerű lenne Docker támogatással bővíteni.

A felhasználói adatok védelme fontos, ezért érdemes lehet rendszeresen átfogó biztonsági auditot végeztetni olyan cég által, amely erre szakosodott. Természetesen ettől függetlenül érdemes tisztában lenni a témával kapcsolatban, evvel kapcsolatban adhat némi segítséget az OWASP top 10-es listája és egyéb ajánlásai, javaslatai a szoftverfejlesztés terén.

5. ÖSSZEGZÉS

A főbb funkciókat sikeresen megvalósítottuk a rövid rendelkezésre álló idő ellenére. Néhány részlet apró vizuális finomításra szorul, és fejlesztés közben több továbbfejlesztési ötlet is felmerült. A teszteket mindenképpen érdemes bővíteni, mivel hosszú távon ez kifizetődő, és a fejlesztők számára is megkönnyíti a kód karbantartását. Így visszajelzést kaphatnak arról, hogy a módosítások befolyásolták-e más funkciókat vagy sem.

6. KÖSZÖNETNYILVÁNÍTÁS

Köszönjük tanárainknak, hogy felkészítettek bennünket a vizsgára, és sok új dolgot tanítottak nekünk ebben a két évben. Hálásak vagyunk azért is, hogy mindig készséggel válaszoltak minden kérdésünkre és segítettek eloszlatni aggodalmainkat.

7. IRODALOMJEGYZÉK

C Sharp (programming language). (2025). Forrás: Wikipedia: https://en.wikipedia.org/wiki/C_Sharp_(programming_language)

C Sharp. (2025). Forrás: Wikipedia: https://hu.wikipedia.org/wiki/C_Sharp

Windows 11 - System requirements. (2025). Forrás: Microsoft: https://www.microsoft.com/en-us/windows/windows-11-specifications

HALLGATÓI NYILATKOZAT

Alulírott ..... (hallgató neve)
a Szegedi Szc Vasvári Pál Gazdasági és Informatikai Technikum hallgatója kijelentem, hogy a ..... (szakdolgozat címe) című záródolgozat a saját munkám.

2025. április 1.

.....
aláírás