VizsgaMunka

Szoftverfejlesztő- és tesztelő képzésen

Vizsgamunka

Készítette: Jancsurák Napsugár,

Varga Antónia,

Katona Alexandra

Konzulens: Németh Bence

Tartalomjegyzék

1. Fejlesztői környezet
2. Frontend fejlesztői környezete
3. HTML5
4. CSS3
5. HTML5 és CSS3 keretrendszer együttműködése
6. Bootstrap
7. WPF
8. Backend fejlesztői környezete
9. C#
10. .NET
11. ASP.NET
12. SQL Server
13. Entity Framework (EF)
14. REST API
15. Első implementációk
16. Áttekintés
17. Az implementáció fő funkcionalitása
18. Felhasználók kezelése
19. Ingatlanok kezelése
20. Foglalások és kommunikáció
21. Adatbázis és kapcsolatok
22. Drótváz és vizuális terv
23. REACT alap weboldal
24. Második implementációk
25. Áttekintés
26. Bejelentkezés és regisztráció
27. Profiloldal
28. Adatok kilistázása adatbázisból
29. Működési logika és technikai megoldások
30. Backend fejlesztések
31. Frontend fejlesztések
32. Harmadik implementációk
33. Áttekintés
34. Új tábla az adatbázisban a sikeres szűrés miatt
35. Részletes szűrés
36. Megyére vagy városra
37. További szolgáltatásokkal
38. WPF karbantartó app fejlesztése adatbázis használata nélkül
39. Ingatlanok tábla, teljeskörű CRUD kérések
40. Tesztelés
41. Negyedik implementációk
42. Áttekintés
43. Adatbázis fejlesztése (bérlő vagy kiadó opció)
44. WPF karbantartó app további fejlesztése
45. Felhasználók és Foglalások tábla CRUD kérései
46. Tesztelés

Fejlesztői környezet

1. **Frontend fejlesztői környezete:**
2. A HTML5 (Hypertext Markup Language 5) az internetes tartalom egyik alapja, amely a weboldalak struktúráját és tartalmát határozza meg. Az előző verziókhoz képest a HTML5 számos új elemmel és funkcióval bővült, amelyek jelentősen javítják a weboldalak működését, hozzáférhetőségét és interaktivitását.

Az egyik legfontosabb újítás, hogy a HTML5 már támogatja a multimédiás elemek (például audio és video) közvetlen lejátszását HTML kódból, anélkül, hogy külső pluginokra lenne szükség, így egyszerűbbé válik a tartalmak integrálása. Emellett a HTML5 lehetőséget biztosít a fejlettebb grafikai megjelenítésre is a és az SVG (Scalable Vector Graphics) technológiák bevezetésével. Ezek segítségével dinamikus grafikák és animációk hozhatók létre közvetlenül a böngészőben, amely korábban csak harmadik féltől származó bővítményekkel volt lehetséges.

A HTML5 által bevezetett szemantikus HTML elemek, mint például a <header>, <footer>, <article>, <section>, <nav> nemcsak a weboldal szerkezetének tisztábbá tételét segítik, hanem hozzájárulnak a SEO (keresőoptimalizálás) javításához is. Ezek az elemek segítik a keresőmotorokat abban, hogy jobban megértsék és indexeljék az oldal tartalmát, így elősegítik a weboldalak jobb helyezését a keresési eredmények között.

Összességében a HTML5 egy jelentős előrelépés a webfejlesztés területén, amely számos új lehetőséget biztosít a modern, interaktív és reszponzív weboldalak fejlesztéséhez. A szemantikus elemek, a multimédiás támogatás, az új API-k és az offline működési lehetőségek mind hozzájárulnak ahhoz, hogy a webfejlesztők hatékonyabb és felhasználó barátibb weboldalakat készíthessenek.

1. **CSS3**: A **CSS3** (Cascading Style Sheets, Level 3) a HTML5 mellett a weboldalak megjelenésének és dizájnjának alapvető szabályait határozza meg. A CSS3 az előző verziókhoz képest számos új funkcióval bővült, amelyek lehetővé teszik a fejlesztők számára, hogy dinamikus, interaktív és reszponzív felületeket készítsenek. A **reszponzív dizájn** azt jelenti, hogy az alkalmazás automatikusan alkalmazkodik a különböző képernyőméretekhez és eszközökhöz, így a felhasználók élménye minden eszközön (asztali gépen, mobilon, tableten) optimalizált marad. A **flexbox** és **CSS grid layout** technológiák a CSS3-ban lehetővé teszik az összetett elrendezések könnyed kezelését, miközben javítják a weboldalak teljesítményét is. A CSS3 segítségével a fejlesztők finomhangolhatják a **vizuális elemeket**, mint például a gombokat, navigációs sávokat, képeket, háttérszíneket és animációkat. Az új CSS funkciók, mint a **@keyframes** és **transitions**, lehetővé teszik a weboldal animációinak, áttűnéseinek és egyéb dinamikus hatásainak implementálását, amelyek az oldalon belépő és eltűnő elemek mozgásának és vizuális változásainak kezelését is biztosítják.
2. **HTML5 és CSS3 együttműködése**: A **HTML5** és **CSS3** együtt biztosítják a weboldalak alapvető felépítését és megjelenését. A HTML5 biztosítja az oldal tartalmát és szerkezetét, míg a CSS3 a megjelenést, a vizuális stílusokat és az elrendezéseket szabályozza. A CSS3 lehetőséget biztosít az oldalak **reszponzív** megjelenítésére, így az alkalmazások automatikusan alkalmazkodnak a különböző kijelzők és eszközök méreteihez. A HTML5 új elemzései és struktúrái együttműködnek a CSS3 által kínált vizuális funkciókkal, lehetővé téve komplex felhasználói felületek és interaktív weboldalak gyors létrehozását. Emellett a CSS3-es animációk és a **flexbox** vagy **grid** alapú elrendezések segítenek a fejlesztőknek a dinamikus és vizuálisan gazdag felületek megvalósításában.
3. **Bootstrap**: A Bootstrap egy nyílt forráskódú front-end keretrendszer, amely célja, hogy megkönnyítse a webalkalmazások és weboldalak fejlesztését, különösen a gyors és egyszerű felépítésük terén. A Bootstrap legnagyobb előnye, hogy előre definiált dizájn elemeket és egy rugalmas grid rendszert biztosít, amely jelentősen felgyorsítja a fejlesztési folyamatot. Ennek köszönhetően a fejlesztők könnyedén létrehozhatnak reszponzív, szép és jól karbantartható weboldalakat, anélkül, hogy az alapvető dizájn és elrendezés részletezésére kellene koncentrálniuk.

A **grid rendszer** az egyik alapvető komponens, amely segít az oldal elrendezésében. Ez lehetővé teszi a fejlesztők számára, hogy egyszerűen hozzanak létre rugalmas, több oszlopos elrendezéseket, miközben biztosítja, hogy az oldalak jól nézzenek ki különböző képernyőméreteken. A grid rendszernek köszönhetően nem szükséges bonyolult CSS kódot írniuk, mivel az előre megadott oszlopok és sávok segítenek az elrendezés gyors kialakításában.

A **Bootstrap** emellett számos előre megtervezett és testreszabható **UI komponenset** tartalmaz. Ezek közé tartoznak például a gombok, navigációs menük, űrlapok, kártyák, értesítési rendszerek és sok más hasznos elem, amelyek már beépített stílusokkal és funkciókkal rendelkeznek. Ez jelentős időmegtakarítást eredményez, mivel nem kell minden egyes komponens stílusait és interakcióit külön-külön megírni.

1. **WPF (Windows Presentation Foundation)**: WPF egy Microsoft által kifejlesztett technológia, amely Windows alapú asztali alkalmazások fejlesztésére szolgál. A WPF az asztali alkalmazások felhasználói felületét kezeli, és számos fejlett grafikai elemet, animációt és interaktivitást kínál. A WPF lehetővé teszi a fejlesztők számára, hogy dekoratív, dinamikus és gazdag felhasználói élményt biztosítsanak a felhasználók számára, miközben teljes mértékben kihasználja a modern grafikus kártyák és a Windows operációs rendszer képességeit.

A WPF XAML (Extensible Application Markup Language) szintaxist használ a felhasználói felületek deklaratív leírására, amely segíti a logika és a megjelenítés elválasztását, ezáltal tiszta és karbantartható kódot biztosít. A WPF támogatja az MVVM (Model-View-ViewModel) architektúrát, amely a kód tisztán tartására és a tesztelhetőség növelésére szolgál. A WPF erőteljes adatbinding mechanizmusa segíti a felhasználói felületek dinamikus frissítését, amikor az alkalmazás adatmodellje változik.

A WPF fejlett grafikai lehetőségeket biztosít, beleértve a vektorgrafikát, 3D megjelenítést és hardveresen gyorsított renderelést, amely lehetővé teszi a nagy teljesítményű vizuális elemek és animációk használatát. A WPF támogatja az eseményvezérelt programozást és a stílusok, sablonok alkalmazását, amelyek segítségével testreszabható és újrafelhasználható UI-komponenseket lehet létrehozni.

A WPF további előnyei közé tartozik az adatvizualizáció támogatása különböző vezérlők, például diagramok és táblázatok segítségével, valamint a multi-touch és a stylus eszközök kezelése. Az elosztott rendszerekkel való kompatibilitás érdekében a WPF képes kommunikálni WCF (Windows Communication Foundation) szolgáltatásokkal, lehetővé téve az adatok hatékony feldolgozását és megjelenítését komplex üzleti alkalmazások esetén.

1. **Backend fejlesztői környezete:**
2. **C#**: A **C#** egy objektum-orientált, **erőteljes programozási nyelv**, amelyet a **Microsoft** fejlesztett ki, és a **.NET** platformon való fejlesztéshez használják. A C# modern programozási nyelvként számos fejlett funkcióval rendelkezik, amelyek segítenek a fejlesztőknek tiszta, karbantartható és hibamentes kódot írni. A C# erős típusellenőrzést, **garbage collection**-t (automatikus memória kezelés), valamint **LINQ** (Language Integrated Query) támogatást biztosít, amely megkönnyíti az adatkezelést és a komplex lekérdezések írását. A C# szoros integrációban áll a **.NET** platformmal, így a backend logika, adatkezelés és API fejlesztés ideális eszköze. A C# nyújtja az eszközöket, amelyek lehetővé teszik a fejlesztők számára a komplex üzleti logikák megvalósítását, miközben biztosítják a kód könnyű karbantarthatóságát.
3. **.NET (vagy .NET Core/5/6/7)**: A .NET egy fejlett keretrendszer, amely lehetővé teszi a fejlesztők számára, hogy különböző típusú alkalmazásokat hozzanak létre, mint például webalkalmazások, asztali alkalmazások, mobil alkalmazások, és szolgáltatások. A .NET-et a Microsoft fejlesztette ki, és az évek során jelentős fejlődésen ment keresztül, hogy megfeleljen a modern fejlesztési követelményeknek.

A .NET Core egy nyílt forráskódú és cross-platform verziója a hagyományos .NET Frameworknek, amely támogatja a Windows, Linux és macOS rendszereket, így rugalmasan alkalmazkodik a fejlesztési igényekhez. A .NET Core 3.1 volt az utolsó verziója a .NET Core sorozatnak, ezt követően a Microsoft egyesítette a különböző .NET implementációkat, és bevezette a .NET 5-öt, amely az egységes platformként szolgál a további fejlesztések számára.

A .NET platform gyors és skálázható alkalmazások fejlesztésére lett optimalizálva, amely különösen előnyös nagy terhelésű és adatintenzív alkalmazások esetén. Az újabb verziók, mint a .NET 5, 6 és 7, számos új funkciót és fejlesztési lehetőséget kínálnak, így a fejlesztők még gyorsabban és hatékonyabban tudják elkészíteni a kívánt alkalmazásokat.

1. **ASP.NET Core**: Az ASP.NET Core egy rendkívül erőteljes és rugalmas webfejlesztési keretrendszer, amely a .NET Core környezetre épül, és számos előnnyel rendelkezik. A legfontosabb jellemzője, hogy cross-platform, így az alkalmazások futtathatók Windows, Linux és macOS rendszereken is, biztosítva a fejlesztők számára a különböző platformokon való munkavégzést.

Az ASP.NET Core az MVC (Model-View-Controller) architektúrát követi, amely segít a fejlesztőknek abban, hogy jól szervezett, könnyen érthető és karbantartható kódot írjanak. Az MVC elválasztja az adatokat (Model), a felhasználói felületet (View), és az üzleti logikát (Controller), ezáltal javítva a kód tisztaságát és skálázhatóságát.

Az Entity Framework Core (EF Core) egy ORM (Object-Relational Mapping) eszköz, amely lehetővé teszi az adatbázisok kezelését közvetlenül a C# nyelven keresztül. Ezzel az eszközzel a fejlesztők könnyebben dolgozhatnak az adatbázisokkal, mivel nem szükséges SQL kódot írniuk, és az adatkezelés is egyszerűbbé válik.

Az ASP.NET Core kiemelkedik a teljesítményével és skálázhatóságával is. A keretrendszer optimalizált, így ideális választás a nagy teljesítményű, skálázható alkalmazások számára. Ez különösen fontos lehet olyan projektek esetén, ahol nagy adatforgalom vagy gyors válaszidő szükséges.

Továbbá, az ASP.NET Core teljes mértékben támogatja a modern webtechnológiákat, például a RESTful API-kat, WebSocketeket és SignalR-t, valamint a legújabb JavaScript és CSS könyvtárakat. Ez biztosítja a fejlesztők számára a legújabb eszközöket és funkcionalitásokat, így mindig naprakészen dolgozhatnak.

1. **SQL Server / MySQL / PostgreSQL / SQLite**: Az **adatbázisok** elengedhetetlenek a backend fejlesztés során, mivel az alkalmazásoknak strukturált módon kell tárolniuk és lekérdezniük az adatokat. Az adatbázis-kezelők közül a **SQL Server**, **MySQL**, **PostgreSQL** és **SQLite** mindegyike különböző funkciókat és előnyöket kínál. A **SQL Server** egy robusztus, vállalati szintű adatbázis-kezelő rendszer, amelyet a Microsoft fejlesztett ki, és kifejezetten nagy terhelésű alkalmazásokhoz alkalmas. A **MySQL** egy nyílt forráskódú, elterjedt relációs adatbázis-kezelő rendszer, amely az egyszerűbb alkalmazások számára kiváló választás. A **PostgreSQL** egy másik nyílt forráskódú adatbázis-kezelő, amely támogatja a bonyolultabb adatstruktúrákat és a transzakciók kezelését. Az **SQLite** egy könnyű, beágyazott adatbázis, amely ideális kisebb alkalmazásokhoz vagy helyi adatkezeléshez, mivel nincs szükség egy különálló szerverre a működéséhez. A választott adatbázis-kezelő rendszer az alkalmazás követelményeitől és méretétől függően változhat, de mindegyik biztosítja az adatok hatékony tárolását és gyors lekérdezését.
2. **Entity Framework (EF)**: Az Entity Framework (EF) egy objektum-relációs leképezési (ORM) keretrendszer a .NET környezetben, amely segíti a fejlesztőket az adatbázisok és a programozási nyelvek közötti kapcsolat kezelésében. Az EF lehetővé teszi, hogy a fejlesztők C# osztályokat és objektumokat használjanak az adatbázis-táblák reprezentálására, így elkerülhetik a közvetlen SQL lekérdezések írását. Ezáltal a kód tisztábbá válik, és az adatbázis-kezelés is egyszerűbbé válik.

Az EF két fő megközelítést kínál az adatbázis-kezeléshez. A **Code First** megközelítés lehetővé teszi, hogy a fejlesztők az adatbázist a kód alapján generálják. Ezzel a módszerrel a modellek (C# osztályok) először elkészülnek, és az EF automatikusan létrehozza az adatbázist. Ezzel szemben a **Database First** megközelítés során a már létező adatbázisból hozhatjuk létre a szükséges kódot, azaz az adatbázis struktúrája először készül el, és az EF ezt használja a kód generálásához.

Az Entity Framework megkönnyíti az adatbázis műveletek kezelését, mint az adatok lekérdezése, frissítése és törlése. A fejlesztők komplex lekérdezéseket és tranzakciókat is végezhetnek anélkül, hogy kézi SQL kódra lenne szükségük. Az EF tehát jelentős idő- és munkaerő-megtakarítást jelent a fejlesztési folyamat során.

Az **EF Core** az Entity Framework könnyebb, gyorsabb és platformfüggetlen verziója, amely kifejezetten a .NET Core környezethez készült. Az EF Core számos új funkcióval és fejlesztéssel érkezett, így gyorsabb működést biztosít, miközben lehetővé teszi az alkalmazások futtatását különböző operációs rendszereken, például Windows, Linux és macOS platformokon.

1. **REST API**: A REST (Representational State Transfer) egy architektúra-stílus, amely az interneten keresztül történő adatkommunikációra összpontosít. A RESTful API egy olyan alkalmazásprogramozási felület (API), amely a HTTP protokollt használja az adatok átvitelére a szerver és a kliens között. A REST alapú API-k széles körben elterjedtek, mivel lehetővé teszik a különböző platformok és rendszerek közötti egyszerű és hatékony kommunikációt.

A REST API-k **stateless** működnek, ami azt jelenti, hogy minden egyes kérés teljes körű információt tartalmaz, így a szerver nem tárol semmilyen adatot az előző kérésekről. Mivel nincs szükség állapotkezelésre a szerver oldalán, a REST API-k képesek hatékonyan kezelni a nagy forgalmat, ami különösen fontos nagy skálázhatóságú alkalmazások esetén. Az egyes kérések önállóan is feldolgozhatók, függetlenül a korábbi kérésektől, így javítva a rendszer válaszidejét és teljesítményét.

A RESTful API-k lehetővé teszik, hogy különböző alkalmazások és rendszerek kommunikáljanak egymással, és gyors adatátvitelt biztosítanak. A REST API-k az **HTTP** metódusokat használják, például a GET, POST, PUT, DELETE, PATCH műveleteket, amelyek az adatok lekérdezésére, létrehozására, módosítására és törlésére szolgálnak. Ez az egyszerű és intuitív megközelítés biztosítja, hogy a fejlesztők gyorsan megértsék és implementálják a RESTful API-kat.

A **JSON** (JavaScript Object Notation) formátumot széles körben támogatják a REST API-k, mivel könnyen olvasható és írható, valamint könnyen integrálható a legtöbb programozási nyelvbe. A JSON formátum előnye, hogy kis helyet foglal, gyorsan feldolgozható, és emberi olvasásra is alkalmas, így ideális választás az adatok közötti kommunikációhoz.

A RESTful API-k továbbá könnyen integrálhatók más rendszerekbe, mivel az HTTP protokoll az internetes kommunikáció alapja, így a REST API-k bármely eszközről vagy programozási környezetből elérhetők. A REST API-k által használt stateless és egyszerűségüknek köszönhetően gyors adatcserét tesznek lehetővé, és rugalmasan alkalmazhatóak a különböző típusú webes alkalmazásokban és szolgáltatásokban.

A REST API-k emellett biztonságosak is lehetnek, ha megfelelő autentikációs és autorizációs mechanizmusok, például az OAuth2, JWT (JSON Web Token) vagy API kulcsok alkalmazásával biztosítjuk, hogy a kommunikáció biztonságos maradjon.

Összességében a RESTful API-k az internetes alkalmazások és szolgáltatások közötti adatcserét egyszerűsítik, miközben lehetővé teszik a nagy teljesítményű, könnyen skálázható és platformfüggetlen rendszerek létrehozását.

Első implementáció

1. **Áttekintés:**

Az első implementáció célja az ingatlanbérlési platform alapjainak lefektetése, egy olyan webes alkalmazás kialakítása, amely egyszerű és intuitív módon szolgálja ki a felhasználók két fő csoportját: a tulajdonosokat és a bérlőket.

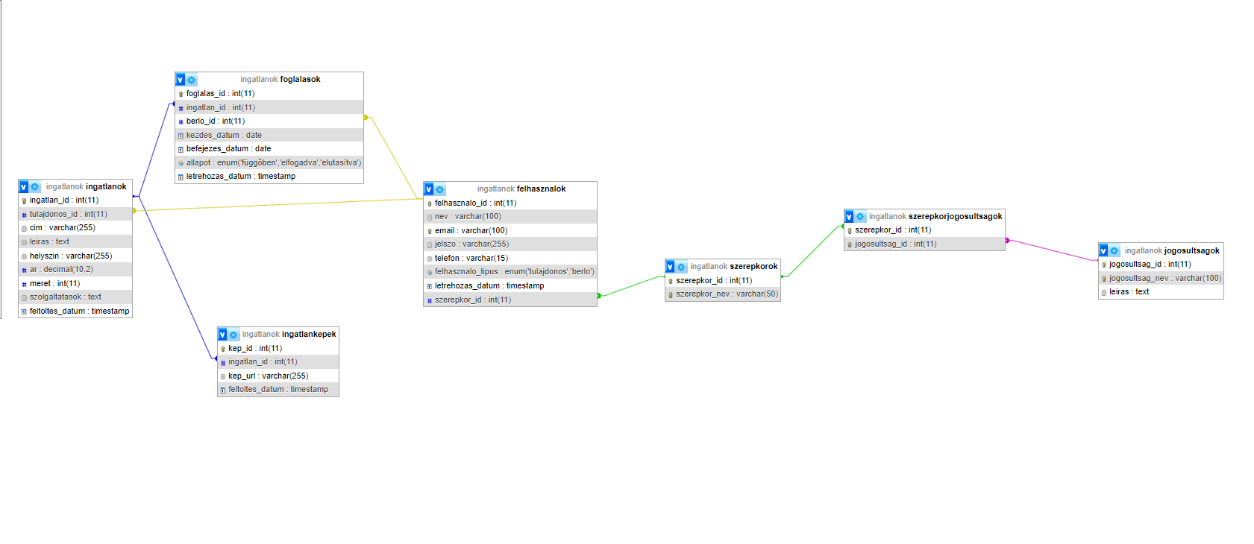
1. **Célfelhasználók:** Magánszemélyek vagy kisebb ingatlanügynökségek, akik ingatlanokat szeretnének bérbe adni, illetve azok, akik kiadó lakásokat keresnek.
2. **Hasonló rendszerek:** Airbnb, Ingatlan.com, Rentingo. Ezekhez képest az első implementáció egyszerűbb, a felhasználók alapvető igényeit célozza meg.
3. **Innováció és alkalmazhatóság:** A projekt könnyen bővíthető. A jelenlegi alapok lehetőséget biztosítanak további fejlesztésekre, például fizetési rendszer, értékelési rendszer vagy mobilalkalmazás integrálására.
4. **Az implementáció fő funkcionalitása:**

Főbb komponensek:

1. Felhasználók kezelése:
   * Regisztráció (tulajdonosként vagy bérlőként).
   * Bejelentkezés és profilkezelés.
   * Szerepkörök és jogosultságok dinamikus kezelése.
2. Ingatlanok kezelése:
   * Tulajdonosok feltölthetik ingatlanjaikat képekkel és leírással.
   * Az ingatlanok listázása, szűrése (pl. ár, helyszín, méret alapján).
   * Egyedi ingatlan részleteinek megtekintése (ár, méret, képek).
3. Foglalások és kommunikáció:
   * Bérlők foglalási kérelmeket küldhetnek a tulajdonosoknak.
   * Tulajdonosok foglalási kérelmeket kezelhetnek (elfogadás/elutasítás).
4. **Adatbázis és kapcsolatok:**

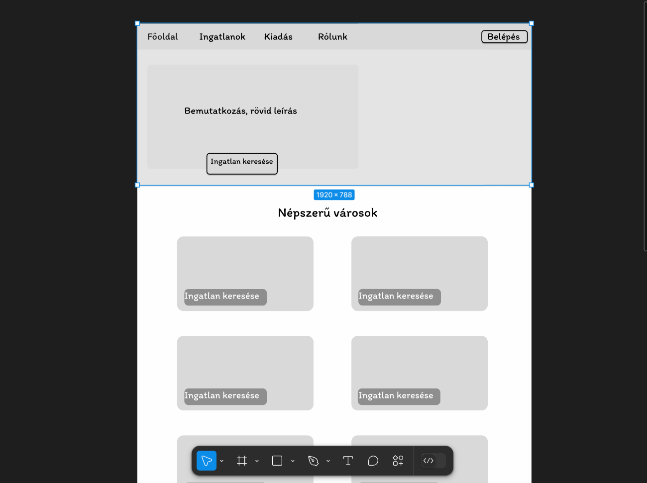
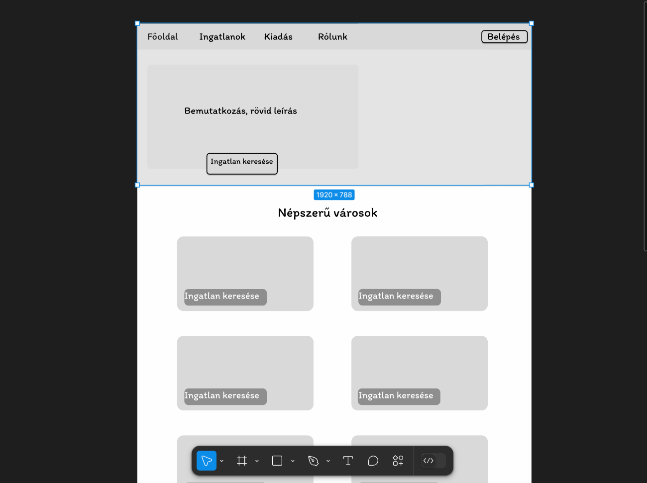
Adatbázis terv: Az adatbázis logikai felépítése biztosítja az adatkezelés biztonságát és rugalmasságát.

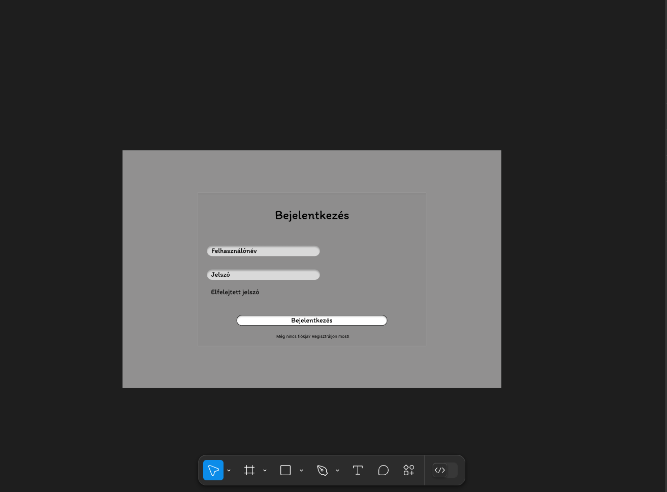
1. Kapcsolatok:
   * A Felhasznalok tábla központi, innen indulnak a kapcsolatok az Ingatlanok, Foglalasok táblák felé.
   * Az Ingatlanok a tulajdonosokhoz kötődnek, és egyedi foglalásokkal, képekkel kapcsolódnak.
   * A Szerepkorok és Jogosultsagok táblák biztosítják a felhasználók jogosultsági szintjeit

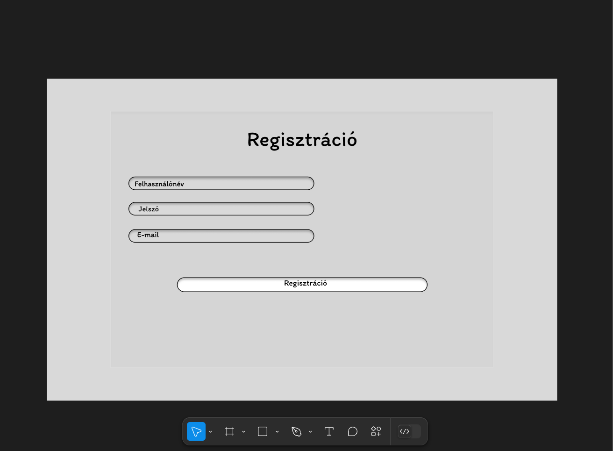


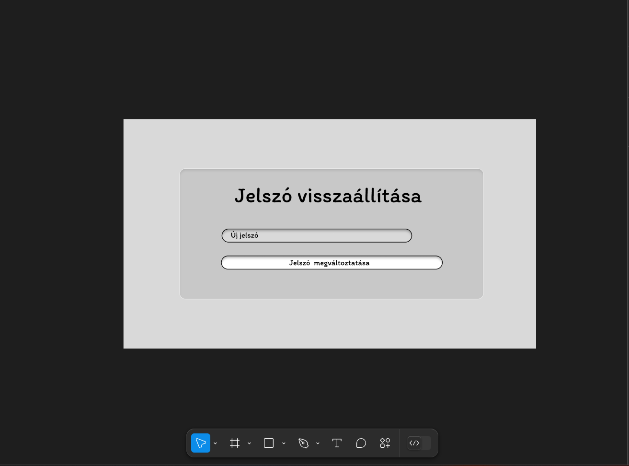
1. **Drótváz és vizuális terv:**

Drótváz**:** A weboldal alapvető elrendezése így épül fel:

1. Főoldal**:**
   * Keresőmező: Ár, helyszín, méret szűrési lehetőségekkel.
   * Legújabb ingatlanok listája.
2. Felhasználói profiloldal:
   * Regisztrációs/Bejelentkezési űrlap.
   * Felhasználói adatok módosítása.

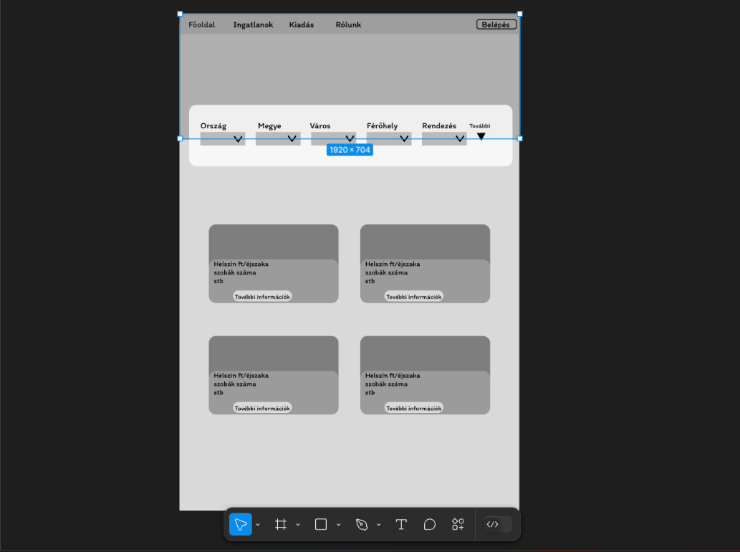






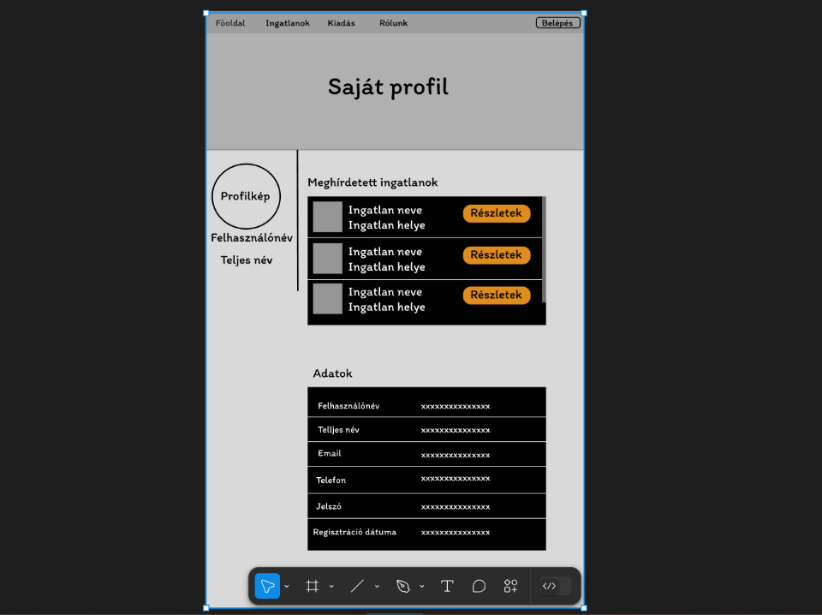
1. Ingatlan részletek, oldala:

* Teljes leírás, képgaléria.
* Foglalás gomb lehetőség.



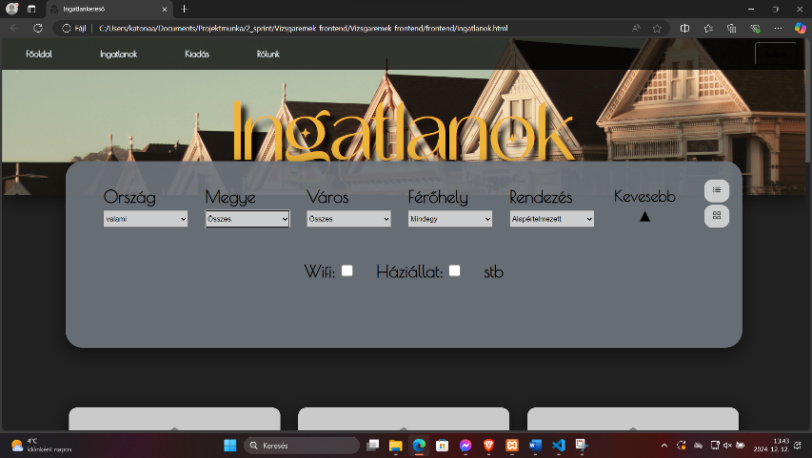
1. Adminisztrációs felület (Tulajdonosoknak):

* Ingatlanok listázása, új ingatlanok feltöltése, meglévők szerkesztése.

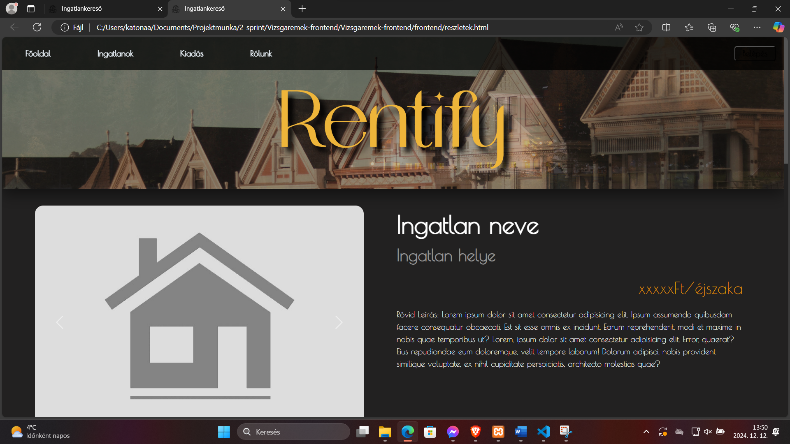


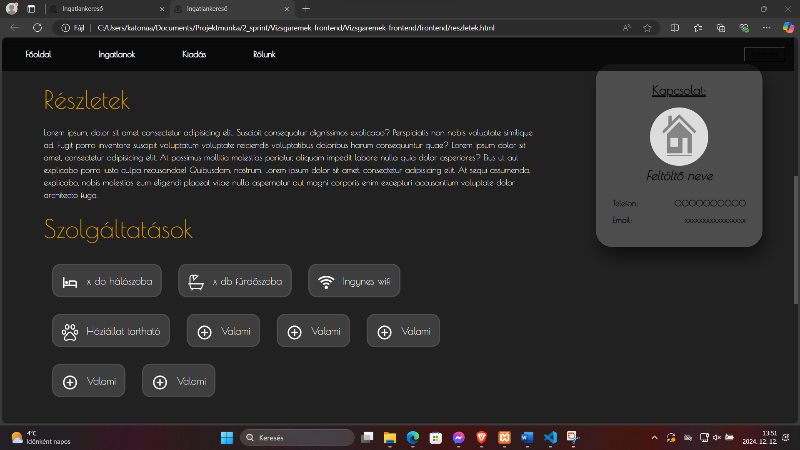
1. **REACT alapú weboldal:**
   1. Keresőmodul:

* Felhasználók bizonyos feltételek szerint ingatlanokra kereshetnek rá, legördülő menükkel.



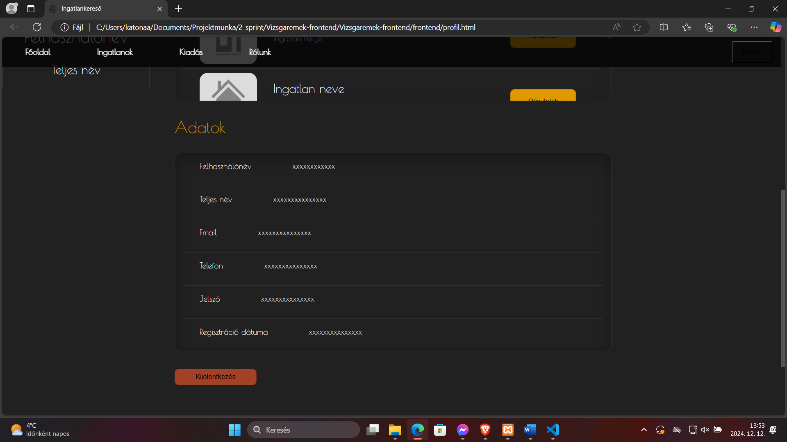
* 1. Részletező oldal:
     + Egy adott ingatlan részletes adatai jelennek meg, például képek, ár, leírás.





* 1. Admin felület:
* Tulajdonosok által kezelhető panel: új ingatlan hozzáadása, meglévő adatok szerkesztése.





Második implementáció

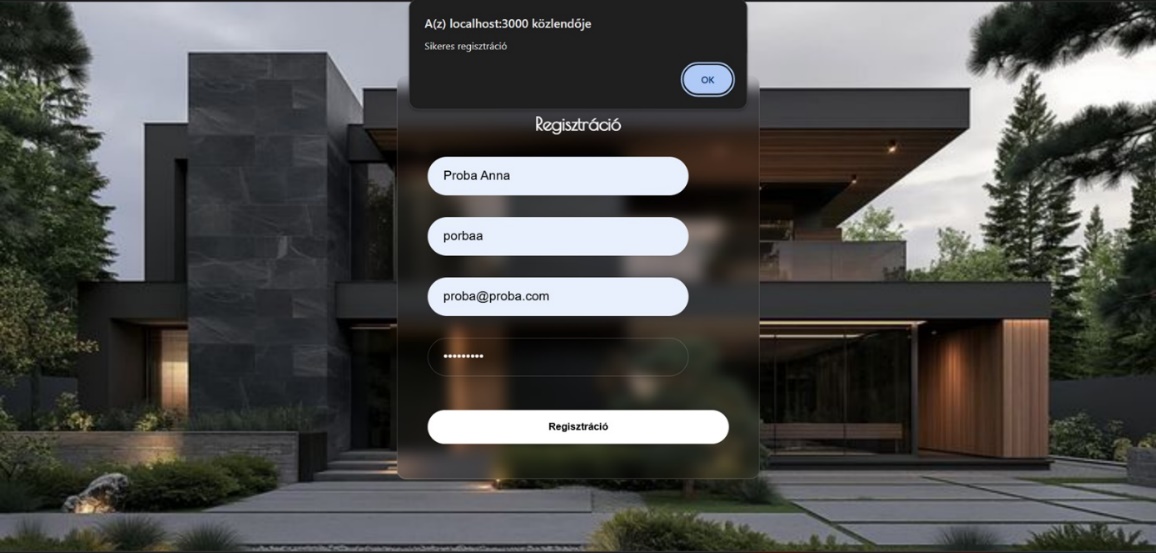
* + 1. **Áttekintés:**

A második iteráció során az ingatlanbérlési platform fejlesztése elsősorban a felhasználói élmény javítására és a működés központi funkcióinak további kidolgozására összpontosított. Ennek keretében kiemelt figyelmet kapott a **bejelentkezési, regisztrációs funkciók**, a **profiloldal** kialakítása, valamint az adatbázis-alapú **adatok kilistázása** és kezelése.

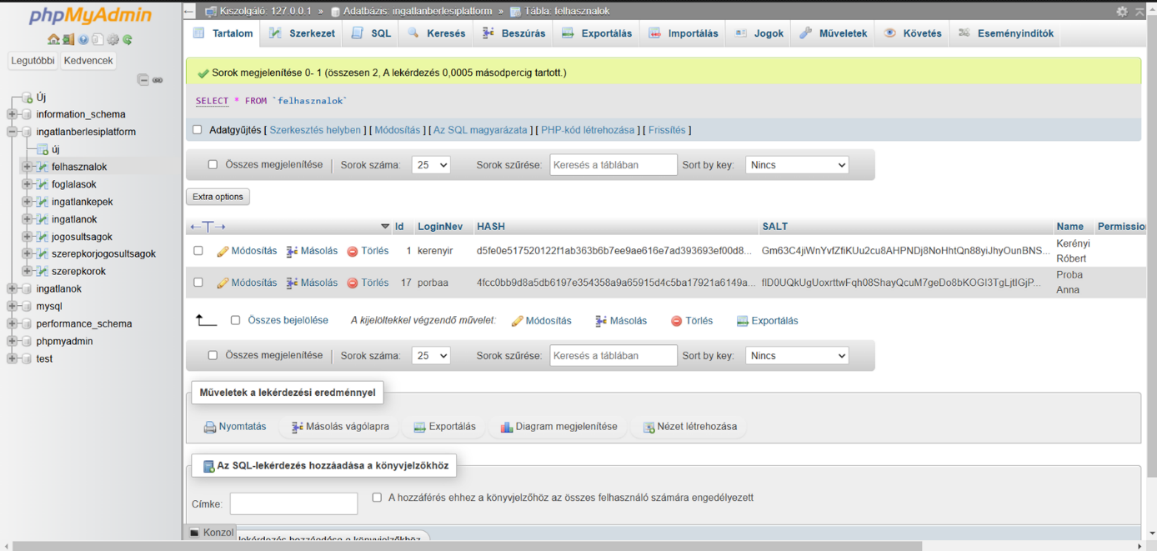
* + 1. **Bejelentkezés és regisztráció:**

1. Regisztráció funkció:

* A regisztráció során a felhasználók kétféle szerepkör közül választhatnak (tulajdonosok vagy bérlők).
* A regisztráció folyamata a következőképpen épül fel:
* Űrlap kitöltése: név, felhasználónév, email cím, jelszó (kötelezően kitöltendő mezők).



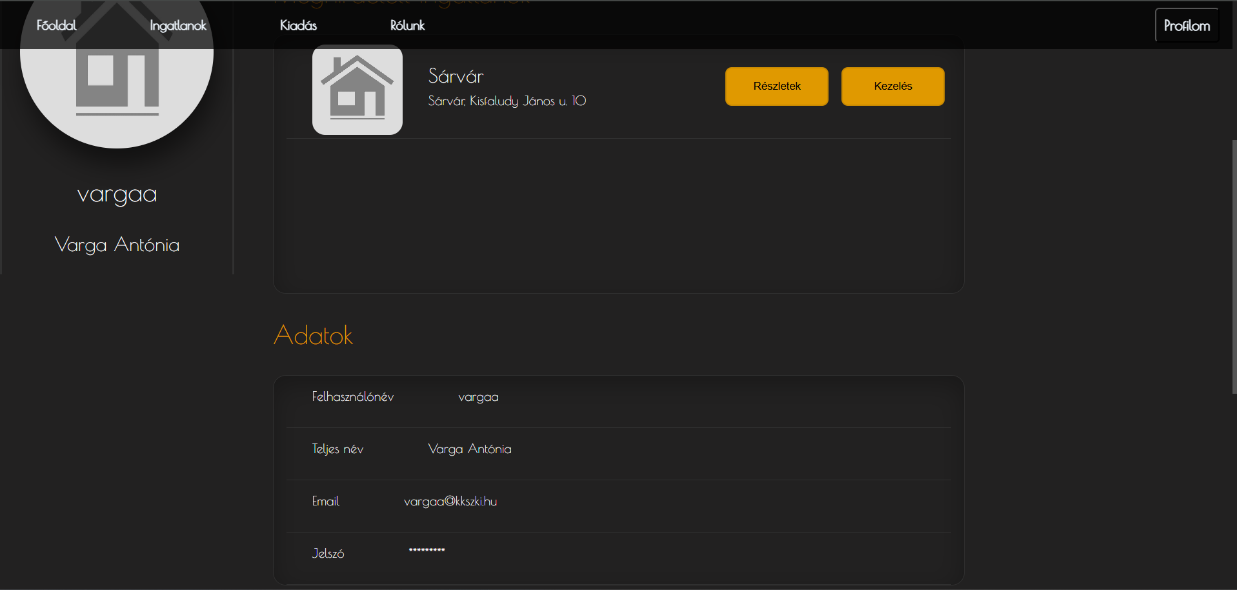
* Szerepkör választás: a felhasználók megjelölik, hogy tulajdonosként vagy bérlőként kíván regisztrálni (későbbi sprintek része).
* Validáció és mentés: az űrlapon megadott adatok validálása (email cím formátum ellenőrzése)
* Az adatbázisban a felhasználók tábla frissítése.



1. Bejelentkezés funkció: A bejelentkezés egy egyszerű felületen keresztül történik, ahol a felhasználók felhasználónevük és jelszavuk megadásával férhetnek hozzá fiókjukhoz. A következő fejlesztéseket implementáltuk:

* Hitelesítés: A megadott adatok ellenőrzése az adatbázisban tárolt értékek alapján.
* Hibakezelés: Hibás adatok esetén értesítés a felhasználónak („Érvénytelen email vagy jelszó” üzenet).
* Token alapú munkamenet-kezelés: Sikeres bejelentkezés után a felhasználó egy időkorlátos munkamenet-tokent kap, amely biztosítja az alkalmazás biztonságos használatát.
  + 1. **Profiloldal:**

A profiloldal funkciójának célja, hogy a felhasználók kezelhessék saját adataikat és áttekintsék a tevékenységüket.

* 1. Felhasználói adatok megjelenítése: A profiloldalon a regisztráció során megadott adatok jelennek meg.
  2. Szerepkör specifikus tartalom:

o Tulajdonosok: Listázhatják és kezelhetik feltöltött ingatlanjaikat (új ingatlan hozzáadása, meglévők szerkesztése)

* Bérlők: Áttekinthetik korábbi foglalásaikat, kommunikációs előzményeiket.
  + 1. **Adatok kilistázása adatbázisból:**

A második iteráció során kiemelt figyelmet kapott az adatbázisból történő adatlekérés optimalizálása és megjelenítése.

* + - * 1. Az ingatlanok megjelenítésére szolgáló modul többféle szűrési lehetőséget biztosít:

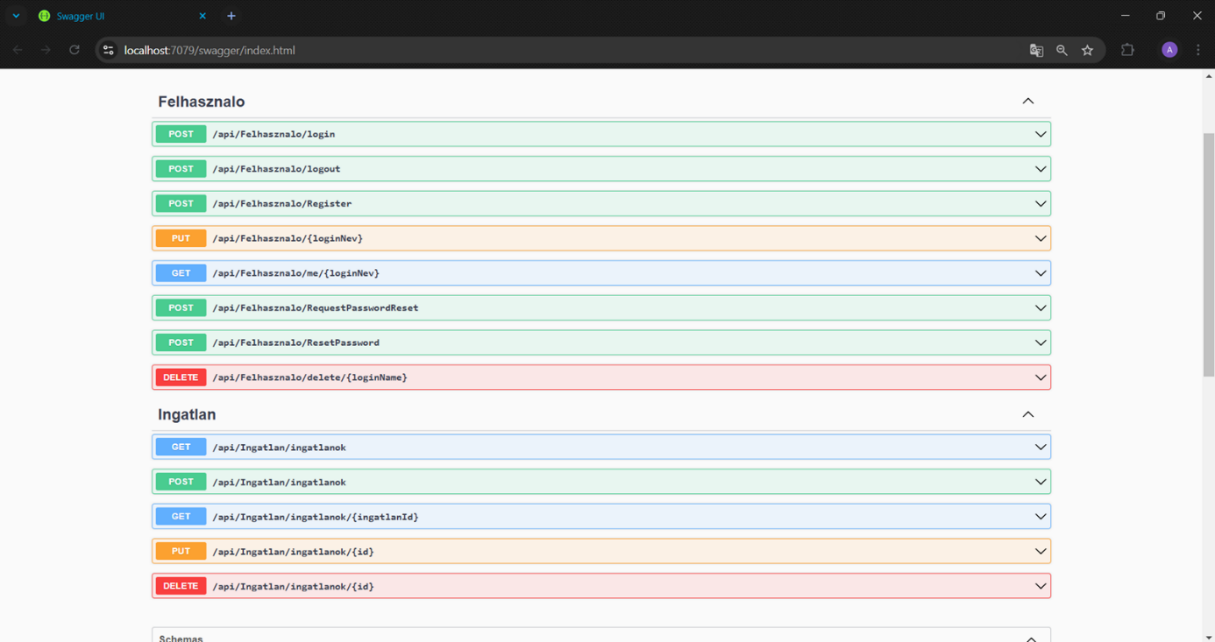
o Alapvető szűrők: helyszín, méret, ár.

o Speciális szűrők.

* + 1. **Működési logika és technikai megoldások**

1. Backend fejlesztések:

* REST API bővítések: az adatlekérések REST API végpontokon keresztül történnek, melyeket C# és ASP.NET Core technológiával implementáltunk.
* Adatbázis műveletek: az Entity Framework Core segítségével biztosítottuk az ORM kapcsolatot, amely lehetővé tette a dinamikus lekérdezéseket és az adatok validációját.



1. Frontend fejlesztések:

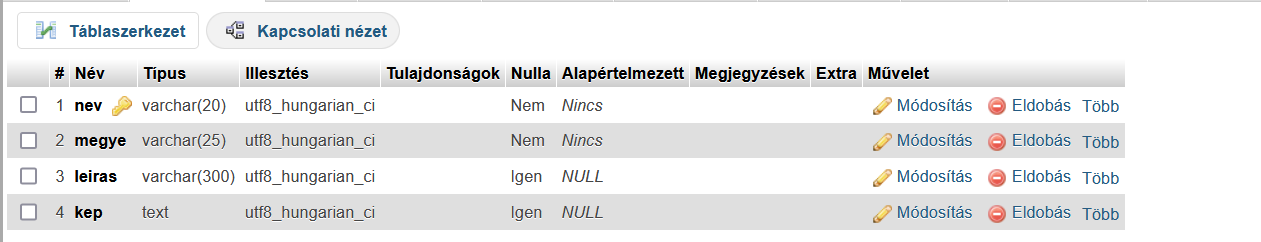
* REACT komponensek: az egyes funkciókat, például a bejelentkezést, regisztrációt és adatlistázást különálló REACT-komponensekbe szerveztük, hogy a kód könnyen karbantartható legyen.
* Reszponzív dizájn: A Bootstrap grid rendszere és egyedi CSS3 megoldások biztosítják az alkalmazás zökkenőmentes működését minden eszközön.

Harmadik implementáció

* + 1. **Áttekintés:**

Ebben az iterációban a platform fejlődése a keresési és szűrési funkcionalitás javítására összpontosított. Az új fejlesztések célja, hogy a felhasználók gyorsabban és pontosabban találjanak releváns ingatlanokat. Az adatbázis bővítésével és a szűrési lehetőségek fejlesztésével a rendszer hatékonyabbá és felhasználóbarátabbá vált. Emellett a karbantartási folyamatokat támogató WPF alkalmazás fejlesztése is megkezdődött, amely az adminisztrációs feladatok kezelését segíti.

* + 1. **Új tábla az adatbázisban a sikeres szűrés miatt:**

Az új szűrési funkció támogatása érdekében egy új tábla került létrehozásra az adatbázisban, amely a városok nevet viseli. Ez a tábla Magyarország összes települését tartalmazza, lehetővé téve a pontosabb keresést és szűrést a rendszerben. A tábla mezői: nev, megye, leiras, kep.

Ez az új adattábla biztosítja, hogy a felhasználók pontosabban és gyorsabban tudjanak keresni a kívánt lokáció alapján.

* + 1. **Részletes szűrés:**

A keresési élmény javítása érdekében fejlesztettük a részletes szűrési funkciót. A felhasználók az alábbi szempontok szerint szűrhetnek:

1. Szobák száma
2. Rendezés szerinti szűrés
3. Extra szolgáltatások (pl. légkondicionáló, erkély, lift, stb.)

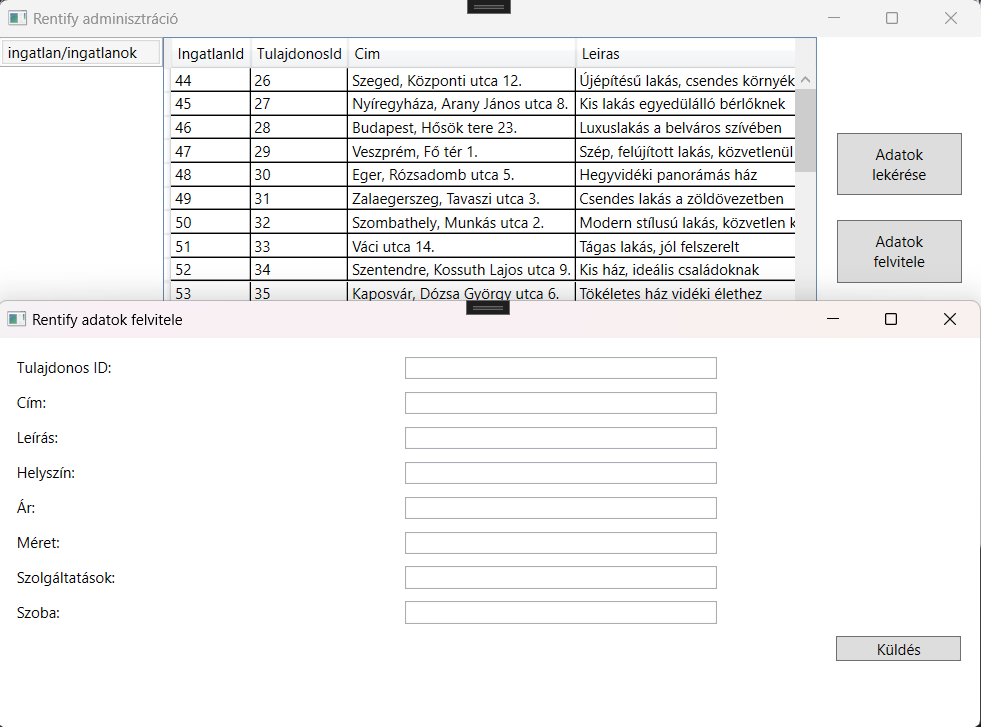
Ezek az új szűrési opciók jelentősen megkönnyítik az ingatlanok közötti navigálást és a megfelelő ingatlan gyorsabb megtalálását.

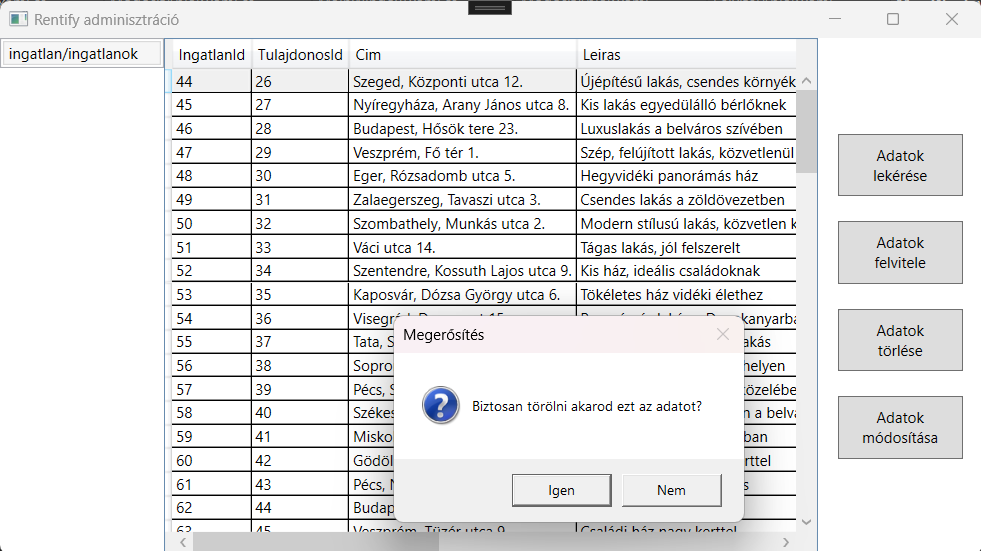
* + 1. **WPF karbantartó app fejlesztése adatbázis használata nélkül:**

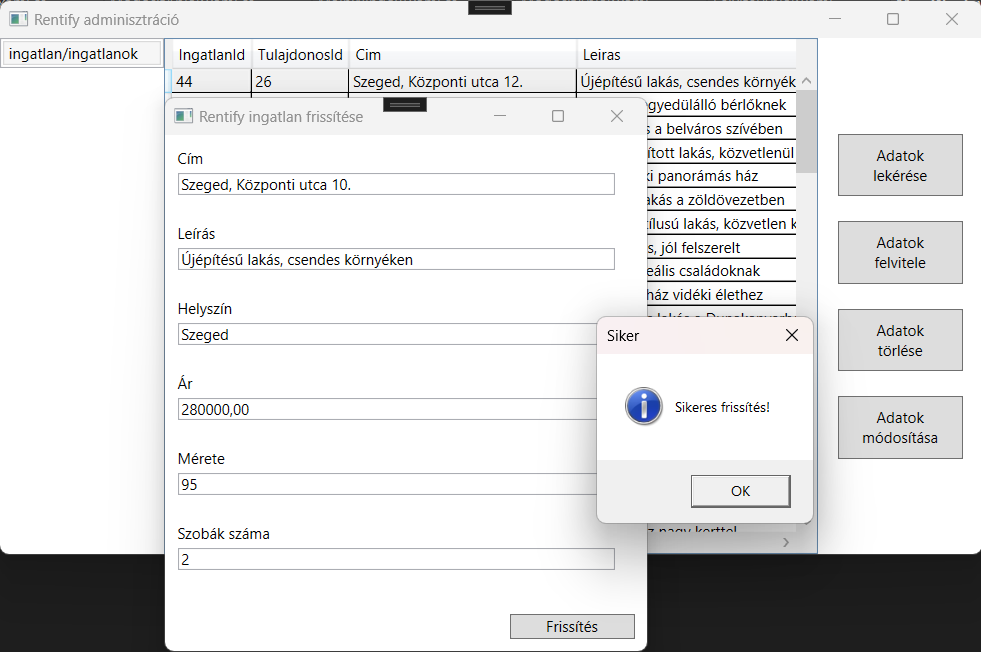
A karbantartási feladatok kezelésére egy külön WPF alapú alkalmazást fejlesztettünk, amely lehetővé teszi az adminisztrátorok számára a rendszer fontos beállításainak és funkcióinak nyomon követését táblákra bontva. Eddig megvalósítottuk az összes CRUD kérést az ingatlan táblára

1. Az alkalmazás eddig az alábbiakat biztosítja:
   * cím,
   * leírás,
   * helyszín,
   * ár,
   * méret,
   * szobák számának

 megjelenítése, módosítása, törlése és teljesen új ingatlan felvitele.







* + 1. **Tesztelés:**

A rendszer minőségének biztosítása érdekében külön tesztelési dokumentáció készült. A tesztelési folyamat magában foglalja az automatikus és manuális teszteket, beleértve:

1. Automatizált tesztelést Selenium WebDriver-rel
2. Egységtesztelést Jest keretrendszerrel a frontend komponensek ellenőrzésére

A részletes tesztelési dokumentáció külön elérhető és tartalmazza a tesztelési forgatókönyveket, eredményeket és hibajavításokat [(https://github.com/JNapsugar/Vizsgaremek.git).](https://github.com/JNapsugar/Vizsgaremek.git)

Negyedik implementáció