2. beadandó feladat

Webes alkalmazások fejlesztése:

Aukciós oldal Webszolgáltatás és asztali kliens WPF-ben

Készítette:
Mikus Márk
NK-kód: CM6TSV
email: kyussfia@gmail.com
2018.05.21.

Feladat:

A cél egy webszolgáltatás ASP.NET Core-ban, MVC architektúrával, amely az adminisztratív funkciókat biztosítja, és Entity Framework segítségével kapcsolódik az adatbázishoz.

2. Aukciós portál

Készítsünk egy aukciókkal foglakozó online rendszert, ahol különböző tárgyakra licitálhatnak a felhasználók.

2. részfeladat: az asztali grafikus felületet az akciók hirdetői használhatják új tárgyak felvitelére, valamint a licitálás helyzetének lekérdezésére.

- Hirdetőként bejelentkezhetünk az alkalmazásba a felhasználónév és a jelszó megadásával, majd ezt követően lehetőségünk van a meghirdetett tárgyak követésére, új tárgy felvitelére, valamint kijelentkezésre.
- Új tárgy felvitelekor megadjuk az elnevezést, a kategóriát, a leírást, a kezdő licitösszeget, a licitálás lezárásának dátumát, továbbá feltölthetünk egy képet is a tárgyhoz.
- A meghirdetett tárgyak (a licitálás lezárási ideje szerint) listázódnak, és minden tárgynál megjelenik, hogy jelenleg mennyit ajánlottak érte. A tárgyat kiválasztva megkapjuk az összes információt képpel együtt, valamint az összes addigi licitet (dátum, név, összeg).
- Lehetőségünk van a licit azonnali lezárására (csak ha valaki már licitált rá, ekkor az aktuális licitáló viszi a tárgyat).

Az adatbázis az alábbi adatokat tárolja:

- Az adatbázis az alábbi adatokat tárolja:
- hirdetők (név, felhasználónév, jelszó);
- felhasználók (név, felhasználónév, jelszó, telefonszám, e-mail cím);
- kategóriák (név);
- tárgyak (név, kategória, leírást, kezdő licit, lezárás dátuma, kép);
- licitek (tárgy, felhasználó, összeg, dátum).

Elemzés:

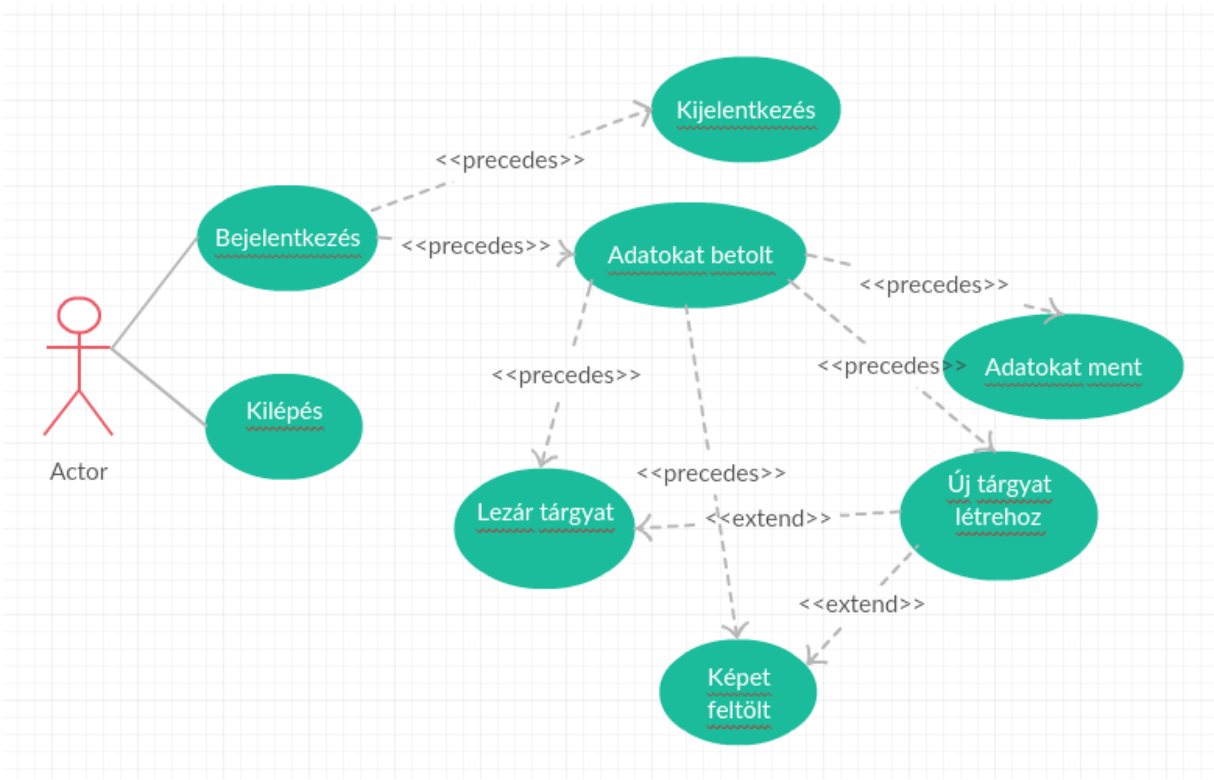
- A feladatot 4 projekt összességeként oldjuk meg:
  - Egy webszolgáltatás ASP.NET Core-ban
  - Egy adat közvetítő alkalmazásréteg a kliens és a webszerviz között (DTO)
  - Egy WPF-ben megvalósított kliensalkalmazás
  - (xUnit) Tesztesetek implementálása a webszerviz teszteléséhez

A webszerviznek lehetőséget kell nyújtani bejelentkezésre a hirdetőknek, adott hirdető tárgyainak lekérdezésére, azok licitjeinek lekérdezésére továbbá új tárgy felvitelére, valamint a tárgyak licitálásának lezárására. A szerviz MVC architektúrában valósul meg. A program lényegi részének dokumentációja ezen feladat 1. részfeladatát képezi.

A kliens alkalmazás MVVM architektúrában kerül megvalósításra, illetve ezen projekt feladata a webszervizen keresztül a DTO objektumok segítségével adatok begyűjtése, majd kezelőfelület szolgáltatása a felhasználónak.

A közvetítőrétegben helyezkednek el az Entitások DTO reprezentációi. Ezek olyan objektumok, amelyek szabadon szinkronizálhatók a kontextusban lévő információkkal.

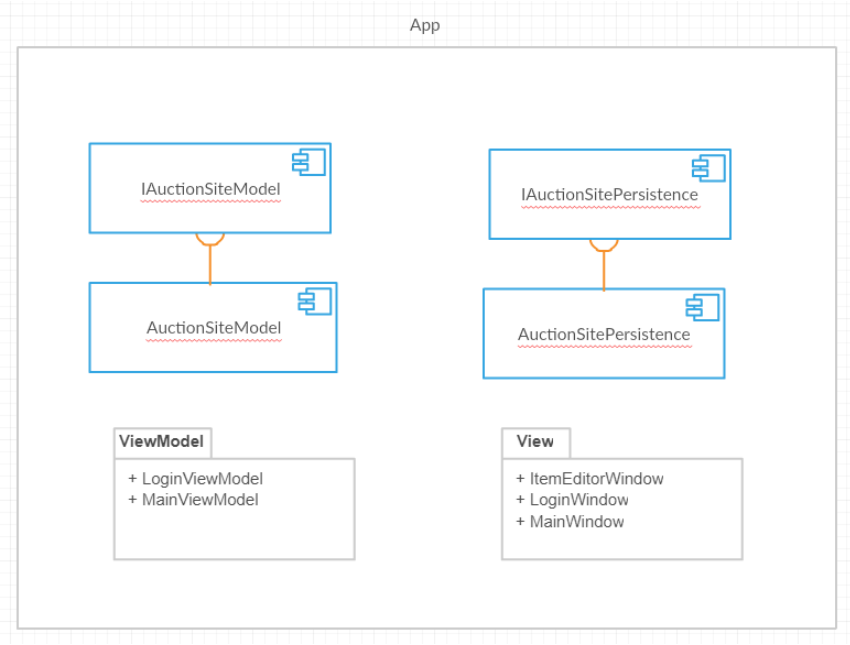
Tervezés és Implementáció:



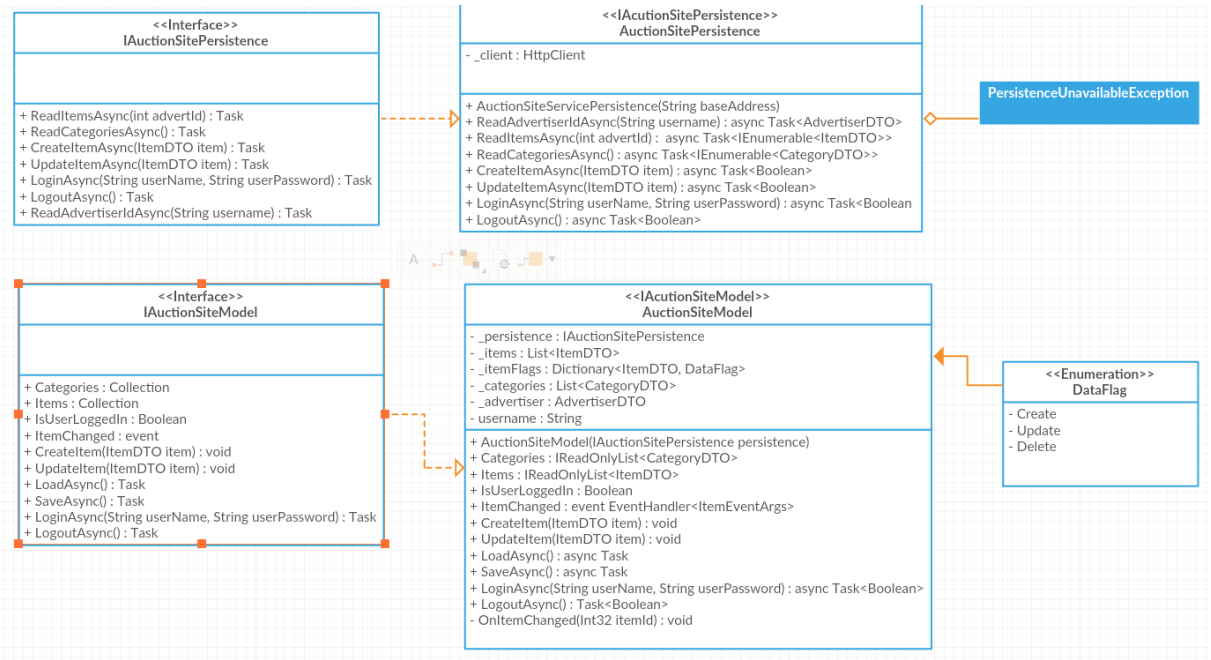
A felhasználói eset diagramon láthatók a kliensen keresztül elérhető funkcionalitások, amelyek tükrözik a webszerviz funkcióit is.

Programszerkezet:

Bővebben a kliens alkalmazásról esik szó, mert az MVC webszerviz részletei a már említett 1. dokumentációban találhatók. Ettől sok eltérés nincs, a megoldás során csupán a megfelelő *Api controllerek* kerültek implementálásra.



A kliens alkalmazás WPF-re tervezett alkalmazás, amely MVVM architektúrában 4 rétegre bomlik. A megfelelő perzisztencia az adatok szolgáltatásáért, a modell azok alkalmazásáért, a nézetmodell a terítéséért, a nézet a megjelenítéséért felel. A modell, illetve a perzisztencia a megfelelő interfészeket implementálva valósulnak meg.



Az alkalmazás magját a program gyökér könyvtárában található App.xaml.cs osztály adja. Ezen osztály köti össze a vezérlőfelületet a nézetmodell eseményeivel. Az App_Startup metódus az alkamazás

indulásakor fut le, itt a program a LoginWindow nézetet jeleníti meg, amin a felhasználó bejelentkezhet az alkalmazásba, illetve kiléphet. Ha bejelentkezett, akkor ViewModel LoginSuccess eseményére kötött, ugyanezen nevű metódusban a MainWindow nézetet kapja a felhasználó ,aki itt menedzselheti a hozzá tartozó tárgyakat. Regisztrálni csak a weboldalon lehet (1.részfeladat)! A program lehetőséget biztosít a tárgyak licitálásának lezárására, illetve kép feltöltésére egy tárgyhoz. Az előbbi az ItemDTO objketum ClosedAt, utóbbi a Picture attribútumát állítja a felhasználó által megadott értékre. A helytelen adatbevitel esetén az App MessageBox-okon keresztül tájékoztat. Az alkalmazás a tényleges adatokat csak a Menüben megtalálható Mentés opció segítségével tárolja el. Ekkor a webszerviz a megfelelő Post, illetve Put metódusokat hívva tárolja el az adatokat.

- Modell

A program Model könytárában találhatóak a réteghez tartozó források. Az IAuctionSiteModel interfész definiálja a megfelelő metódusokat, amelyeket az ezen interfészt megvalósító AuctionSiteModel osztály valósít meg. Tárolja a lehetséges kategóriákat, a bejelentkezett felhasználó termékeit, illetve azt is, hogy a felhasználó be van-e jelentkezve. Az ItemChanged esemény hatására a MainViewModel objektum módosítja a megfelelő kollekciót. A LoginAsync illetve a LogoutAsync segítségével elvégzi a megfelelő autentikációt, míg a CreatedItem illetve UpdateItem segítségével létrehozást, illetve módosítást vált ki a perzisztencián keresztül.

- Perzisztencia

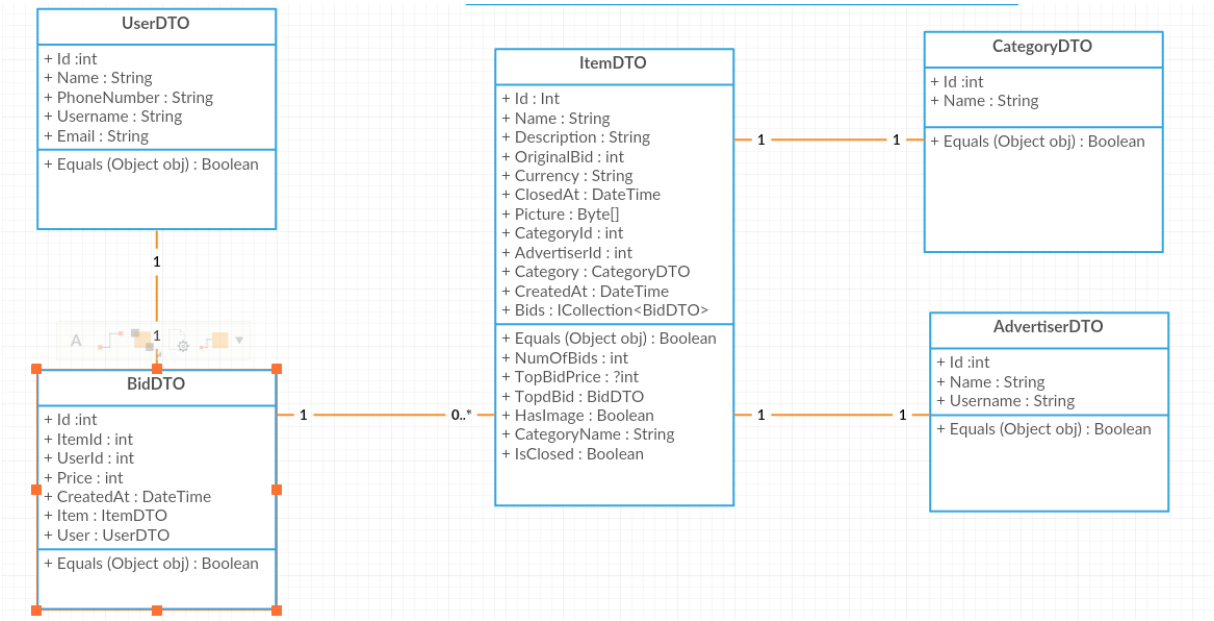
Az IAuctionSitePersitence interfészben találhatóak a megfelelő adatelérési műveletek, illetve az alkalmazáson belül az adatok frissítését szolgáltató metódusok (ReadItemsAsync, readCategoriesAsync). A readAdvertiserIdAsync megadott Username alapján ad vissza egy AdvertiserDTO objektumot. A CreateItemAsync, UpdateItemAsync, LoginAsync és a LogoutAsync a már taglalt működéseket továbbítják a webapi felé.

- Nézetmodell

A ViewModel könyvtárban találhatók a nézetmodell objektumok, illetve az ezekhez szükséges segédobkejtumok. A DelegateCommand osztály a megfelelő nézetre Bindolt Commandok őse. A Converters könyvtárban találhatóak a nézet által használt érték átalakítók, amelyek segítségével a kívánt formátumra hozhatók a megfelelő entitás Property-k.A ViewModelBase definiálj azokat az ősmetódusokat, amelyek segítségével üzenetek közvetíthetőek a Nézetrétegre a MessageEventArgs osztály segítségével. Két főbb nézetmodell komponenst határozhatunk meg. A LoginViewModel a bejelentkezési nézet, a MainViewModel a főablak nézete mögött logikát valósítja meg, illetve tartja a kapcsolatot a modell objektummal.

- Nézet

A View mappában találhatók a nézetet képző xaml állományok. Itt 3 fontos komponens valósult meg. A LoginWindow a bejelentkezési ablakot reprezentálja, amelyen a megfelelő felhasználónév és jelszó pároshoz tartozó szöveges bevizeli mezők találhatók, tovább a egy Ok, és egy Mégse gomb. A MainWindow adja a főablak nézetét, amelyen megtalálhatóak a lezáráshoz, létrehozáshoz és képfeltöltéshez tartozó gombok, egy sáv a megjelenítendő képnek, táblázat a termékekhez és a Menü, amelyen keresztül, frissíthető/menthető az alkalmazásban tártolt adathalmaz és lehetőséget is ad a Kijelentkezésre, a program bezárására. Az ItemEditorWindow új termék létrehozásakor vagy meglévő megtekintésekor jelenik meg, ezen találhatóak az input mezők, amelyeken definiálható egy - egy termék. (Megtekintési módban ezek az inputot le vannak tiltva, csak a jelenlegi értéket mutatják.) Megtalálható továbbá egy táblázat, amelybe a megfelelő terméken lévő licitek kerülnek betöltésre, amennyiben van ilyen.



Tesztelés:

A tesztelési project egy olyan xUnit tesztprojekt, amelynek az AuctionSIteTest állományában találhatóak a teszteléshez szükséges segédmetódusok és a tesztesetek is. Az tesztesetek több kisebb tesztesetekre bonthatók az egyes Assert-ek elkülönítésével. A tesztesetek mindegyike lefutott, illetve az összességük lefedi a webszerviz funkcionalitását. Az alábbi tesztesetek valósultak meg:

- GetCategories
- GetItemsOfAdvertiser
- PostItem
- PutItem
- GetAdvertiserByUserName
- GetBidsOfItem
- GetBids
- GetItem
- GetCategory
- GetUser