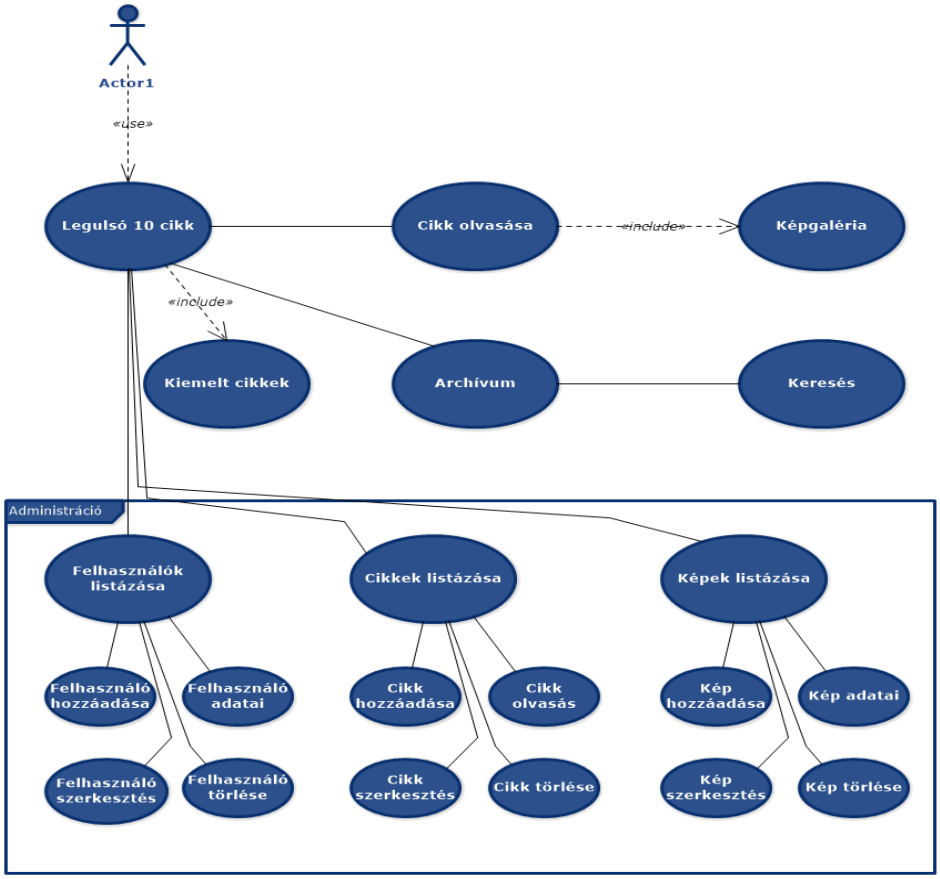
FELADAT:

Készítsük el egy online portál hírkezelő rendszerét, ahol a munkatársak feltölthetik cikkeiket, amelyek megjelennek egy webes felületen. A webes felület tartalmazza magát a hírportált, ahol az olvasok tetszőlegesen böngészhetik a híreket.

- A főoldalon megjelennek a legfrissebb hírek (cím és összefoglaló, dátum szerint csökkenő sorrendben, legfeljebb 10), illetve a lap tetején kiemelten avezető cikk (cím és összefoglaló) képpel (amennyiben több kép tartozik a cikkhez, az első jelenik meg, kicsinyített méretben).
- A címet kiválasztva megjelenik a teljes tartalom (beleértve a szerző nevét, illetve a bevitel, vagy utolsó módosítás dátumát) képpel (ha van kép a cikkhez rendelve, amennyiben több kép tartozik a cikkhez, az első jelenik meg, kicsinyített méretben). A képet kiválasztva megjelenik a cikkhez tartozó képgyűjtemény, ahol egyenként lapozhatunk a képek között, illetve visszaléphetünk a cikkhez.
- A hírportál tartalmaz egy archívumot, ahol dátum szerint csökkenő sorrendben listázódnak a hírek (cím és összefoglaló). Egy oldalon legfeljebb 20 hírt láthatunk, a többiért lapozni kell. Az archívumban lehet keresni is, megadott dátumra, cím(részlet)re, vagy tetszőleges szóra a cikk tartalmából.

ELEMZÉS:

- A weblapot ASP.NET Core MVC keretrendszerben lett megvalósítva, kihasználva az általa adott funkcionalitást (állapotkezelés, validáció).
- Az adatok adatbázisban vannak tárolva, és Entity Framework Core segítségével vannak kezelve.
- A felület több részből tevődik össze. Az egységes elrendezést Layout segítségével valósítjuk meg. Ami tartalmazza a fejlécet, láblécet, beilleszti a szükséges szkripteket.
- A fejléc egy menüsorból áll, mely tartalmazza az oldal nevét valamint egy hivatkozást a kezdőlapra és az archívumra.
- A kezdőlap a legfrissebb 10 hír felsorolása. A kiemelt hírek legelöl jelennek meg.
- A hírek címeire kattintva eljutunk egy oldalra, ahol a teljes cikket elolvashatjuk. Itt megjelenik a teljes szöveg, a cikkhez tartozó kép (a képet kiválasztva megjelenik a cikkhez tartozó képgyűjtemény, ahol egyenként lapozhatunk a képek között, illetve visszaléphetünk a cikkhez), az író neve, utolsó módosítás dátuma.
- Az archívum menüpontra kattintva a lapozható hírarchívumba navigálhatunk. Itt fel van sorolva az összes cikk címe és összefoglalója dátum szerint csökkenő sorrendben. Oldalanként maximum 20 jelenik meg. A kereső mezőt kitöltve szűkíthetjük a megjelenő híreket. Keresni megadott dátumra, cím(részlet)re, vagy tetszőleges szóra lehet.



1. ábra: Felhasználói diagramm

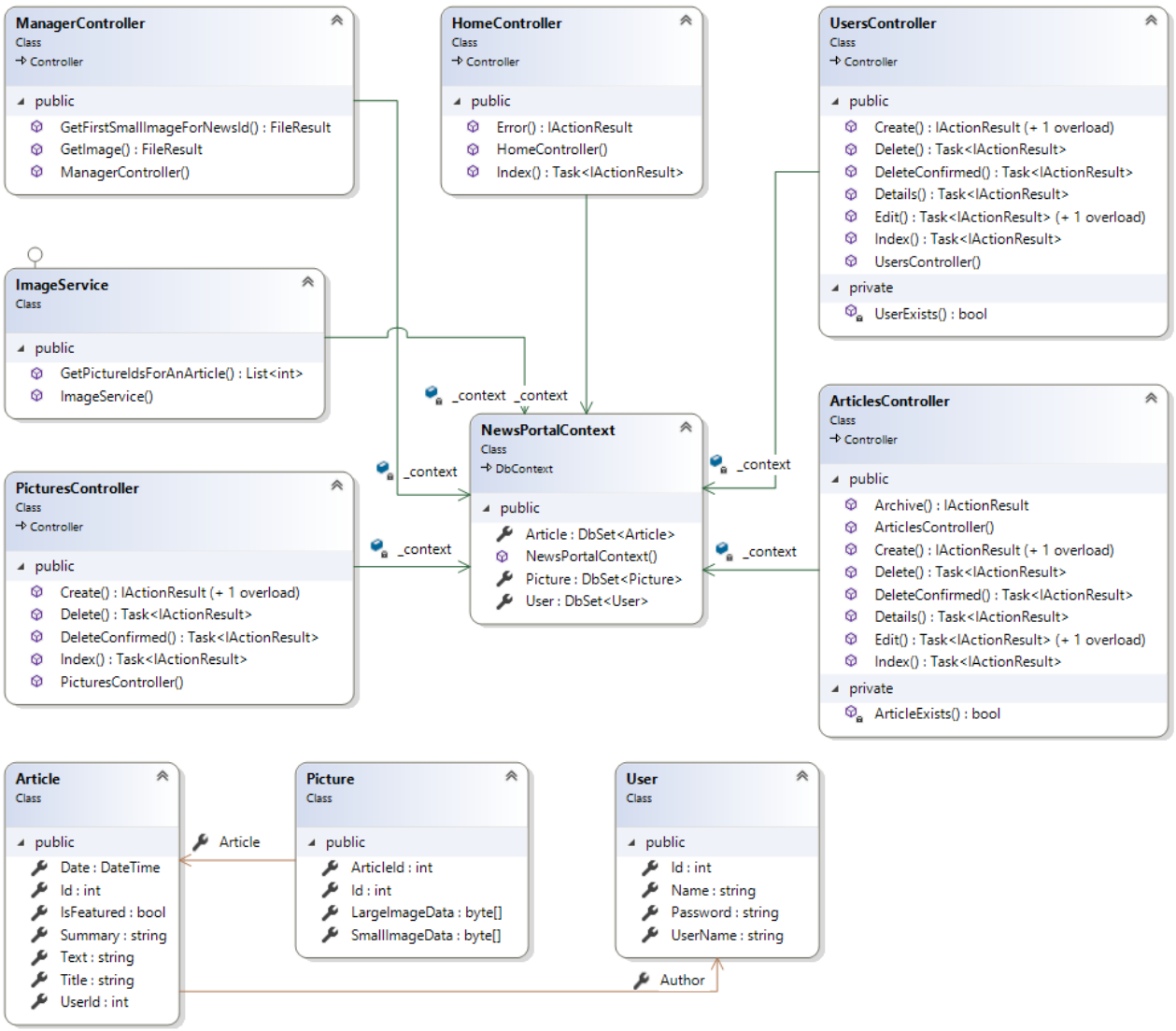
Tervezés:

Programszerkezet:

A programot .NET Core MVC architektúrában valósítjuk meg, ennek megfelelően View, Model névtereket valósítunk meg az alkalmazáson belül. A Controller névtérben van definiálva a felhasználó által kezdeményezett és végrehajtható akciók.

MODELL:

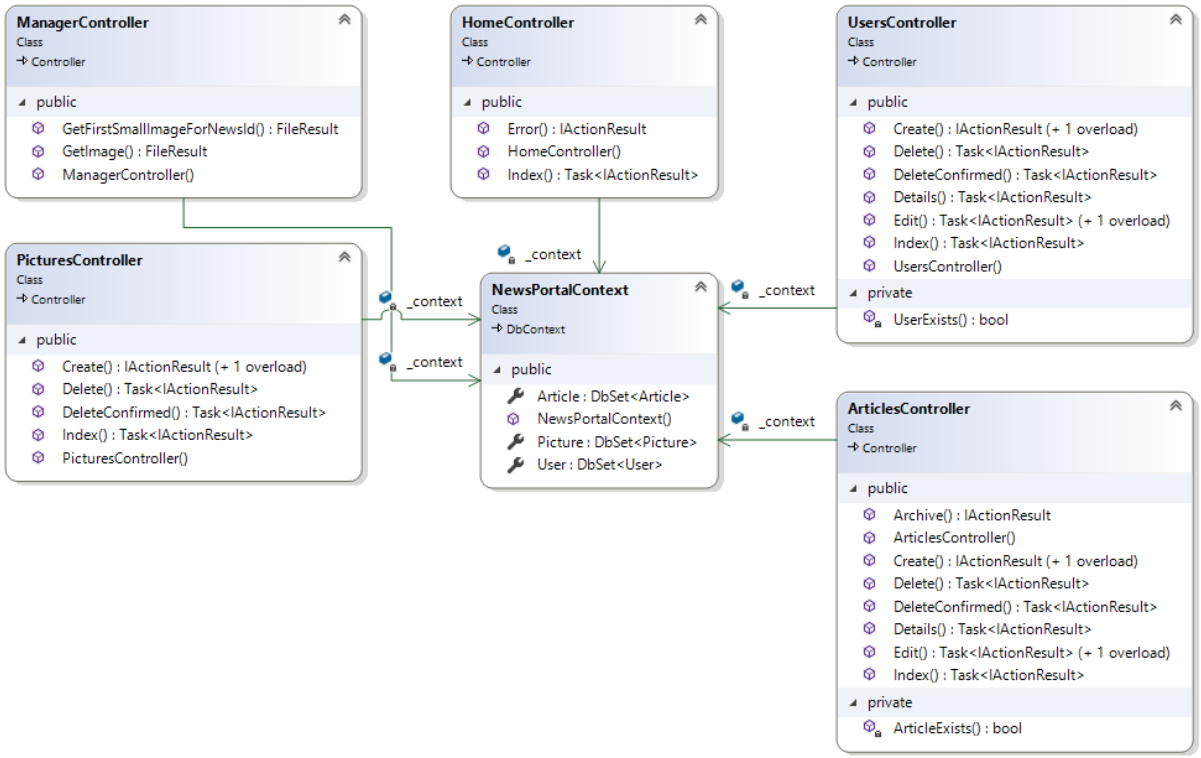
- Az adatbázist az Entity Framework Core által nyújtott entitás modell segítségével egy objektumrelációs adatbázisként reprezentáljuk. Ezt az NewsPortalContext osztály valósítja meg.
- Az entitás modellel a Controllerek közvetlenül tartják a kapcsolatot. Tehát a Controllerek egyes metódusai kérik le és adják tovább a nézetnek az összes adatot az adatbázisból.



2. ábra: Osztálydiagramm

VEZÉRLŐ:

- A HomeController osztály biztosítja a felhasználó számára az Index akción keresztül a fő oldal megjelenítését.
- Az ArticleController a cikkekkel kapcsolatos akciókat valósítja meg. A Details akció egy meghatározott ID-val rendelkező cikk teljes tartalmának kiírására alkalmas. Az Archíve akció valósítja meg cikkek lapokba rendezett kiírását és a keresést.
- A ManagerController segítségével fájlokat (képeket) kérhetünk le az adatbázisból.
- (A PicturesController a képekkel kapcsolatos akciókat valósítja meg)
- (A UsersController a felhasználókkal kapcsolatos akciókat valósítja meg)



3. ábra: Osztálydiagramm (vezérlés)

NÉZET:

A _Layout elrendezés felel az oldal egységes szerkezetű kialakításáért. Beállítja a címsort melyben a weboldal neve és az aktuális oldalra vonatkozó információ szerepel. Beilleszti a működéshez szükséges szkripteket és a szép megjelenítéshez elengedhetetlen stílusokat. Definiál egy menüsort és egy láblécet is mely egy keretet alkotnak az oldalnak.

HOME/[INDEX]

Az index nézet valósítja meg a fő oldalt, mely megjeleníti az utoljára felvitt 10 cikk címét, hozzá tartozó képet, rövid összefoglalóját. A cikkek címei linkek, melyek re kattintva megjelenik a cikk teljes tartalma.

ARTICLES/[CREATE|EDIT|DELETE]

Cikkek listázására/feltöltésére/szerkesztésre/törlésére alkalmas nézetek.

ARTICLES/[DETAILS]

Egy adott cikk teljes tartalmának megjelenítésére alkalmas nézet. Megjeleníti többek között a cikk címét, rövid összefoglalóját, teljes szövegét, szerzőt, utolsó módosítás dátumát, képet (ha van kép a cikkhez rendelve, amennyiben több kép tartozik a cikkhez, az első jelenik meg, kicsinyített méretben). A képet kiválasztva megjelenik a cikkhez tartozó képgyűjtemény, ahol egyenként lapozhatunk a képek között, illetve visszaléphetünk a cikkhez.

ARTICLES/[ARCHIVE]

Az archívum megjelenítésére és keresésre ad lehetőséget. Oldalakra tagolja a cikkek megjelenítését. Oldalanként legfeljebb 20 cikket listáz. A listában a cikkekek címét és rövid összefoglalóját dátum szerint csökkenő sorrendben jeleníti meg. Továbbá három keresési mezőt jelenít meg, melyekkel dátumra, címre, szövegre kereshetünk.

PICTURES/[INDEX|CREATE|EDIT|DELETE]

Képek listázására/feltöltésére/szerkesztésre/törlésére alkalmas nézetek.

USERS /[INDEX|CREATE|EDIT|DELETE]

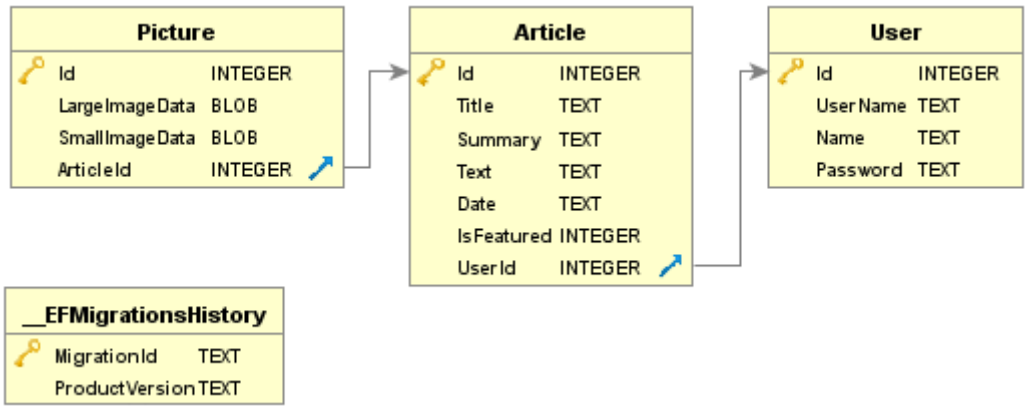
Felhasználók listázására/feltöltésére/szerkesztésre/törlésére alkalmas nézetek.

4. ábra: Komponensdiagramm

ADATBÁZIS:

Az adatbázis 3 táblából áll:

- **Articles:** A cikkek adatait tárolja (id, cím, összefoglaló, dátum, stb..).
- **Pictures:** A képek adatait tárolja (id, kis kép, nagy kép, stb..).
- **Users:** A felhasználók adatait tárolja (id, név, usernév, jelszó).



5. ábra: Osztálydiagramm (Adatbázis)

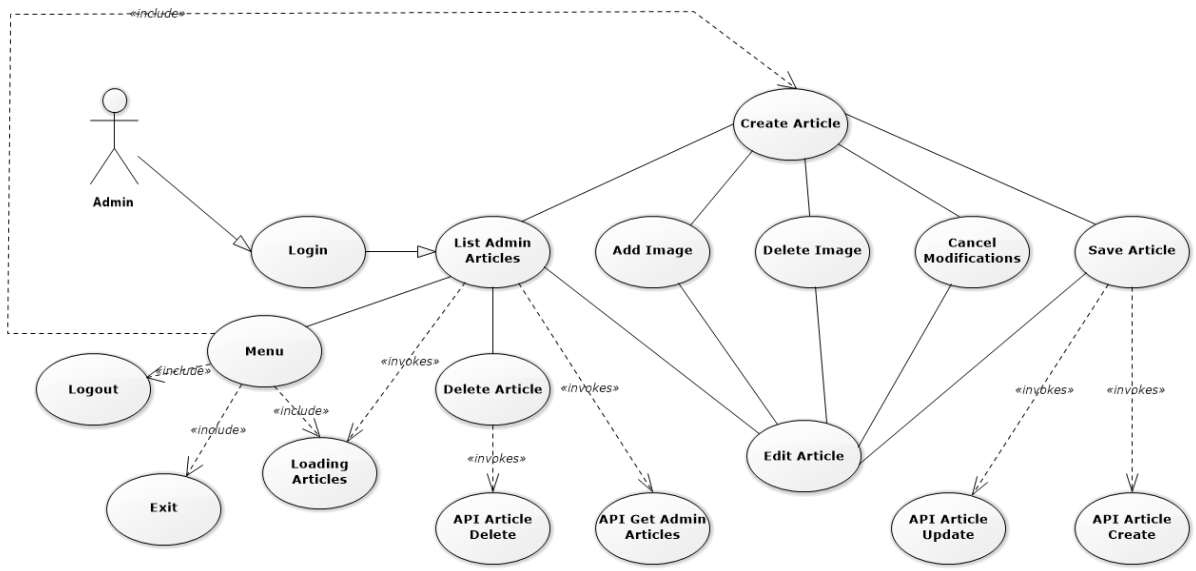
FELADAT:

Készítsük el egy online portál hírkezelő rendszerét, ahol a munkatársak feltölthetik cikkeiket, az asztali grafikus felületet.

- A program használatához először be kell jelentkeznie a munkatársnak a felhasználónév és a jelszó megadásával.
- Ezt követően válnak elérhetővé a szerkesztési funkciók (illetve a kijelentkezés).
- A fő ablakban a saját cikkek listázódnak dátum szerint (cím, író, dátum), amelyeket módosíthatunk, illetve törölhetünk is.
- Új cikk felvitelénél meg kell adnunk a címet, az összefoglalót (max. 1000 karakter), valamint a teljes szöveget. Ezek kitöltése kötelező.
- A cikk beállítható vezető cikknek, ekkor azonban kötelező legalább egy képet feltölteni hozzá. Ezen felül feltölthetnek tetszőleges számú képet a cikkhez.
- Cikk módosításánál is ugyanezt a felületet kapjuk vissza, de már előre kitöltve.
- Cikk törlésénél a program megerősítést kér a felhasználótól.

ELEMZÉS:

- Három alapvető nézet elkészítése szükséges. Ezek a bejelentkező, szerkesztő és fő ablak nézetek.
- A program elindulásakor a bejelentkező ablakot inicializálja a program. Majd bejelentkezés után a program automatikusan lekéri az adatokat az API segítségével. A további funkciók ezek után válnak elérhetővé.
- A fő ablakban dátum szerint csökkenő sorrendben listázódnak a szerverről az API által lekért saját cikkeink adatai. Ezen adatokat egy Datagrid reprezentál.
- A fő ablak menüpontjaiban megtaláljuk a kijelentkezés, kilépés és az adatbázis frissítése menüpontokat.
- A fő ablak három további gombot jelenít meg, melyek az New Article, Edit Article és a Delete Article. Melyekkel rendre új cikket hozhatunk létre, szerkeszthetünk egy cikket illetve törülhetünk egy cikket.
- A New Article gombra kattintva megjelenik az új cikk létrehozására alkalmas nézet. Itt meg kell adnunk a cikk címet, az összefoglalót (maximum 1000 karakter), valamint a teljes szöveget. Ezek kitöltése kötelező. Továbbá a cikk beállítható vezető cikknek, ám ekkor legalább egy kép feltöltése kötelező.
- Egy cikkhez független attól, hogy vezető cikk vagy sem, tetszőleges számú kép feltölthető. A Save-re kattintva az új cikk bekerül az adatbázisba. A Cancel visszavonja a véghezvitt változtatásokat.
- Cikk módosítása során ugyanazt a felületet nyitjuk meg, mint az új hozzáadása során, de már előre kitöltve.
- Cikk és kép törlésénél a program megerősítést kér a felhasználótól.



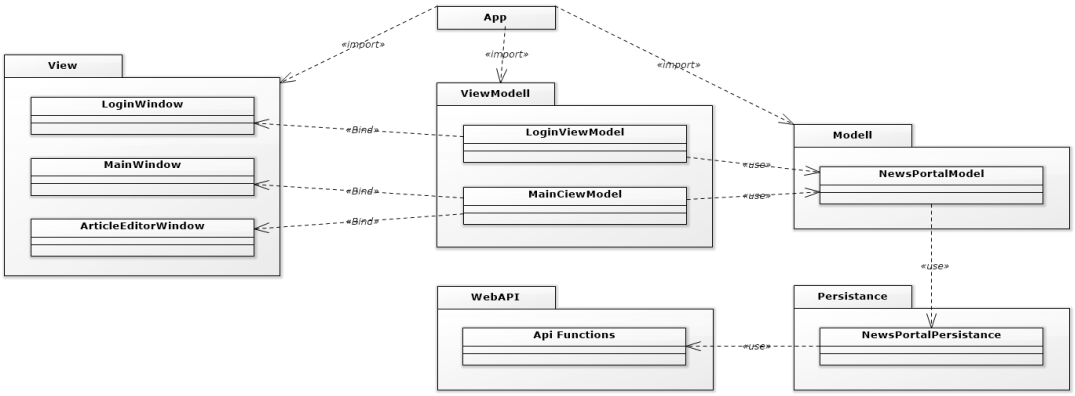
6. ábra: Felhasználói diagramm

TERVEZÉS:

PROGRAMSZERKEZET:

Az adminisztrációs programot MVVM architektúrában valósítjuk meg, ennek megfelelően View, Model, ViewModel névtereket valósítunk meg az alkalmazáson belül. A program környezetét az alkalmazás osztály (App) végzi, amely példányosítja a modellt, a nézetmodell és a nézetet, biztosítja a kommunikációt.

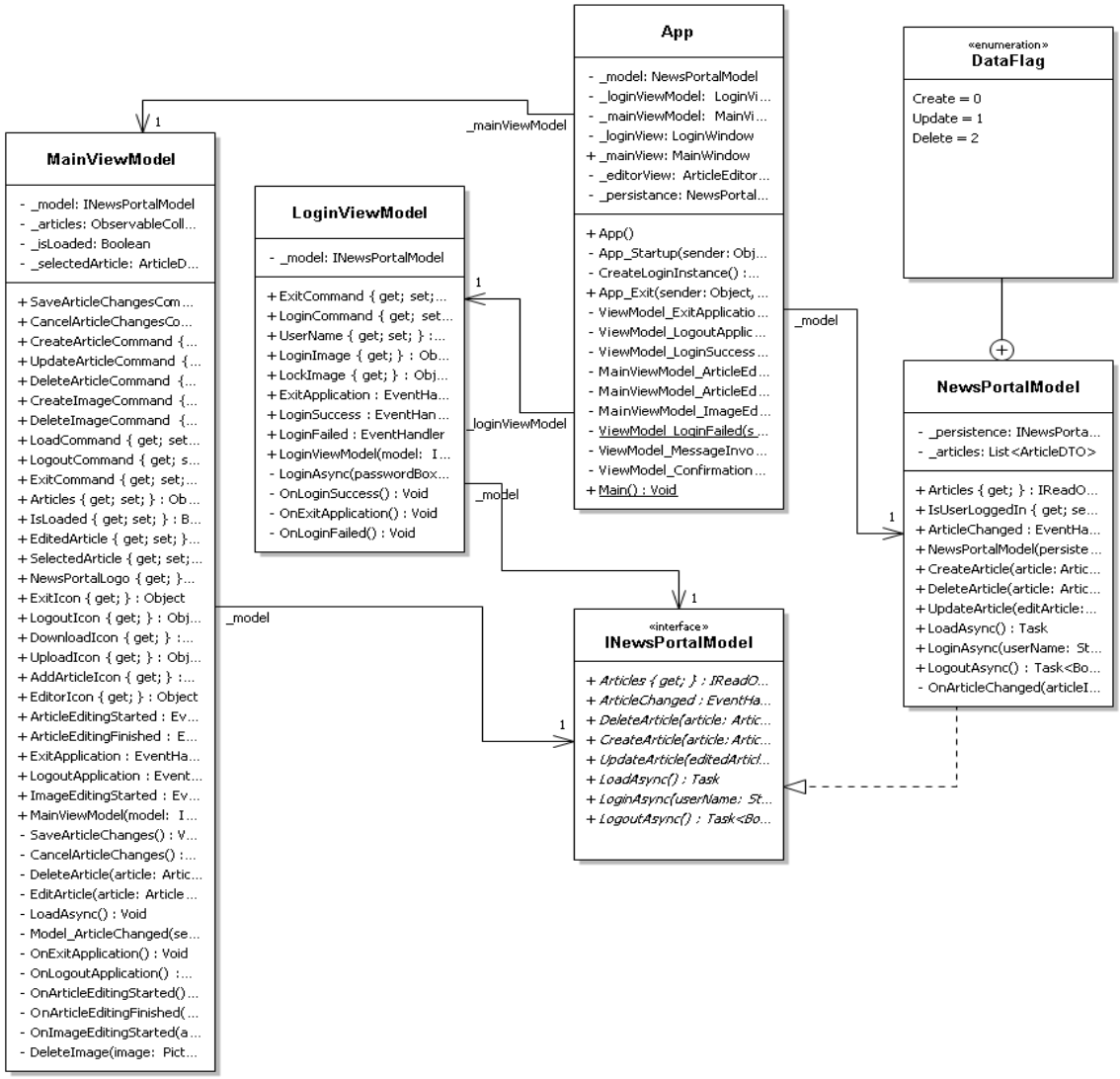
Az WebAPI szolgáltatást .NET Core MVC architektúrában valósítjuk meg, ennek megfelelően View, Model névtereket valósítunk meg az alkalmazáson belül. A Controller névtérben van definiálva a felhasználó által kezdeményezett és végrehajtható akciók.



7. ábra Csomagszerkezet

MODELL:

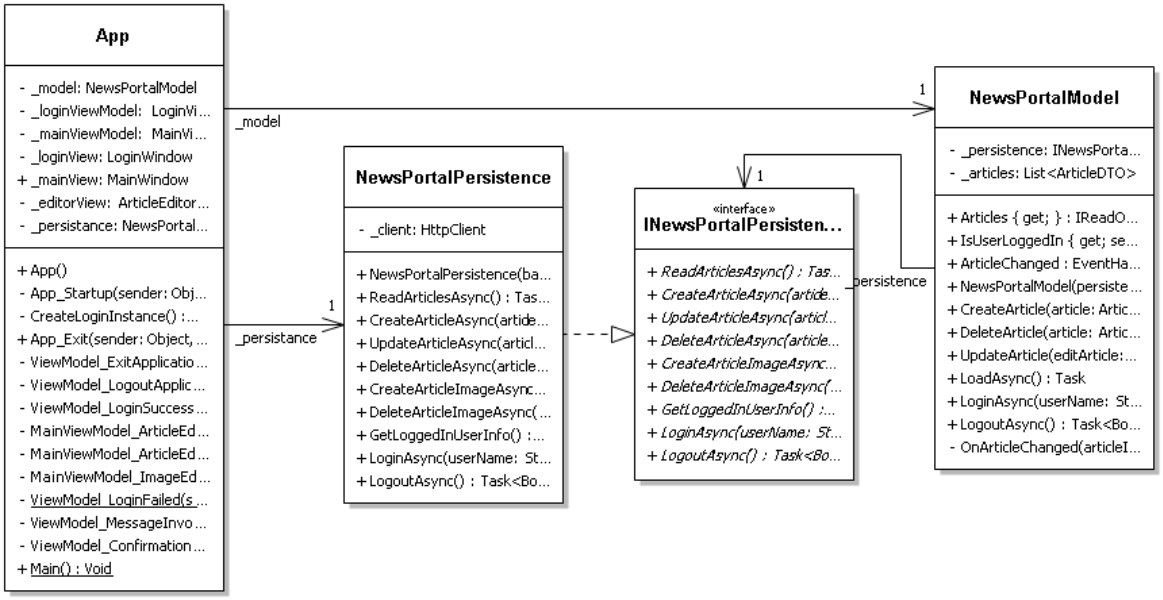
- A modell egy lokális leképezése az adatbázisban levő adatoknak. Az adatok letöltése után, felépít egy lokális adatszerkezetet, melyet a nézet modellek fognak használni és a nézetek fogják megjeleníteni. Data Transfer Objects formátumokat használunk a helyi reprezentációhoz.
- Itt is végzünk ellenőrzéseket, valamint még vizsgáljuk, hogy a lokális adatbázis reprezentáción érvényes-e.



8. ábra: Osztálydiagramm (Modell)

PERZISZTENCIA:

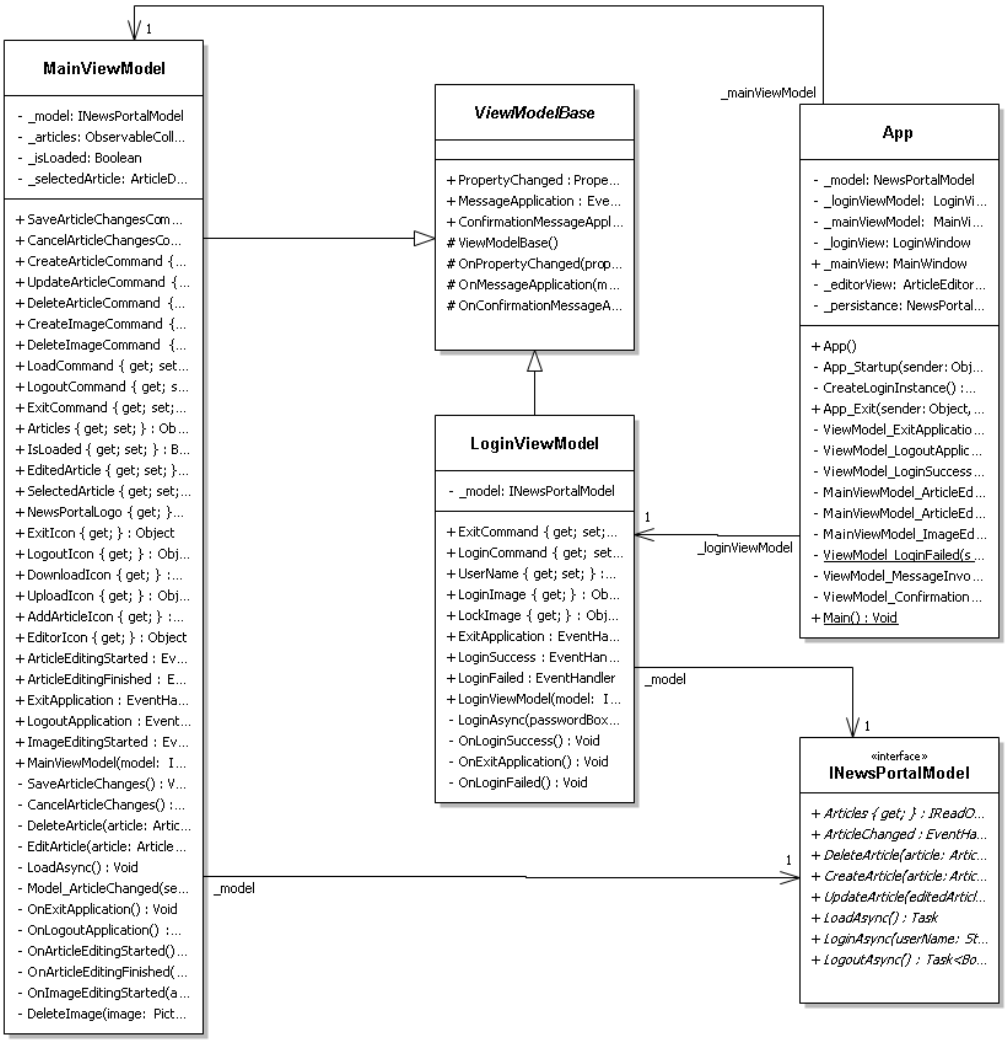
A perzisztencia valósítja meg a WebAPI-val történő kommunikációt, mellyel szinkronban tudjuk tartani a helyi adatbázis reprezentációnkat a valódival. Data Transfer Objects formátumokat használunk a kommunikációhoz.



9. ábra Osztálydiagramm (Perzisztencia)

NÉZETMODELL:

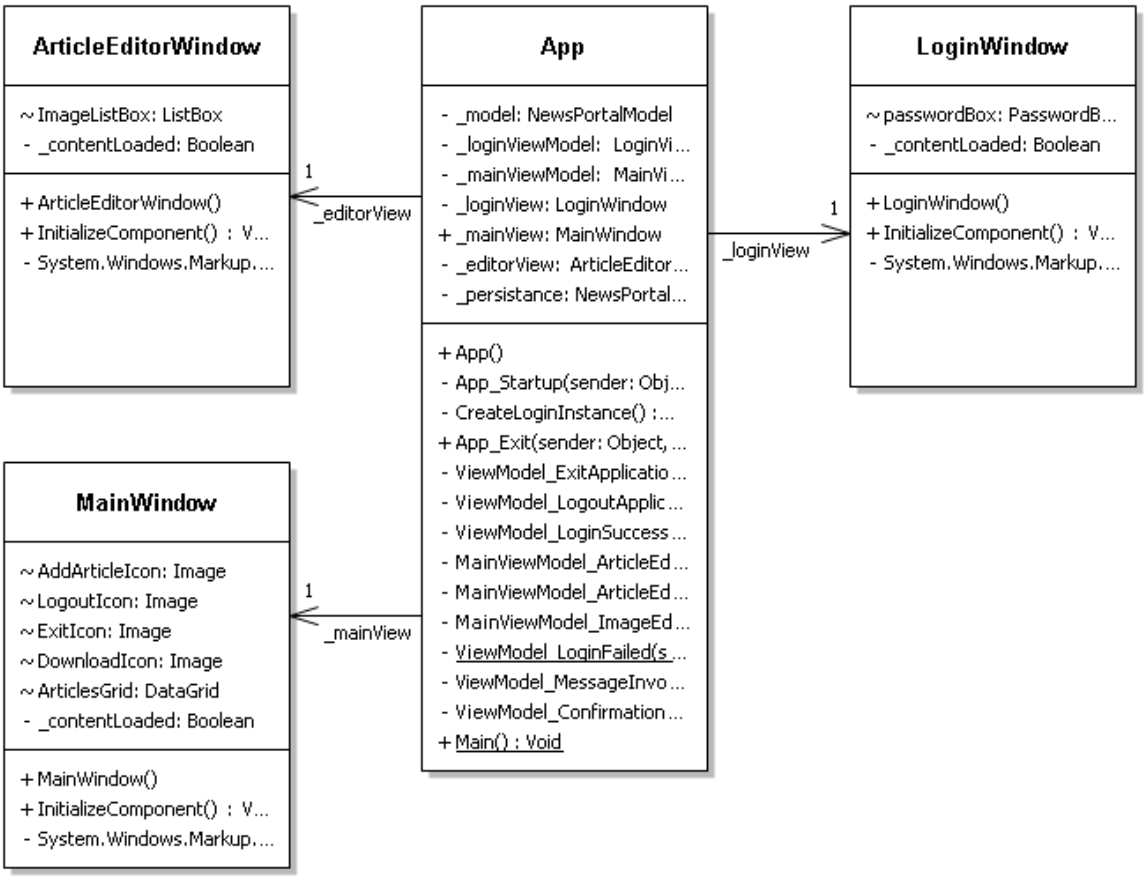
- A nézetmodell megvalósításához felhasználunk egy általános utasítás (DelegateCommand), valamint egy ős változásjelző (ViewModelBase) osztályt.
- A nézetmodellek az adatokat kötik a hozzájuk tartozó nézethez, valamit események hívásával a modell egyes műveleteit hívják meg.
- A bejelentkező ablakhoz tartotó nézet modell ellenőrizteti a bejelentkezési adatokat a Web API-val a modellen és a perzisztencián keresztül és ezek válaszának függvényében küld üzenetet a kontrollernek. Sikertelene esetben, a felhasználónak van további lehetősége próbálkozni. Sikeres bejelentkezés esetén a kontroller megnyitja a fő ablakot és bezárja a bejelentkezést lehetővé tevő ablakot.
- A fő ablakhoz tartozó nézet modell kezeli az összes adatmanipulációt, cikkek megjelenítését. Felsorolható listában tárolja az a saját cikkeinket. Továbbá lehetőséget biztosít adatbázis frissítésére, kijelentkezésre és kilépésre. Az adott akciók során eseményeket indít, melyeket majd a hozzá csatlakoztatott osztályok kezelnek le.
- Cikk és kép törlésénél a program megerősítést kér a felhasználótól.



10. ábra Osztálydiagramm (Nézetmodell)

NÉZET:

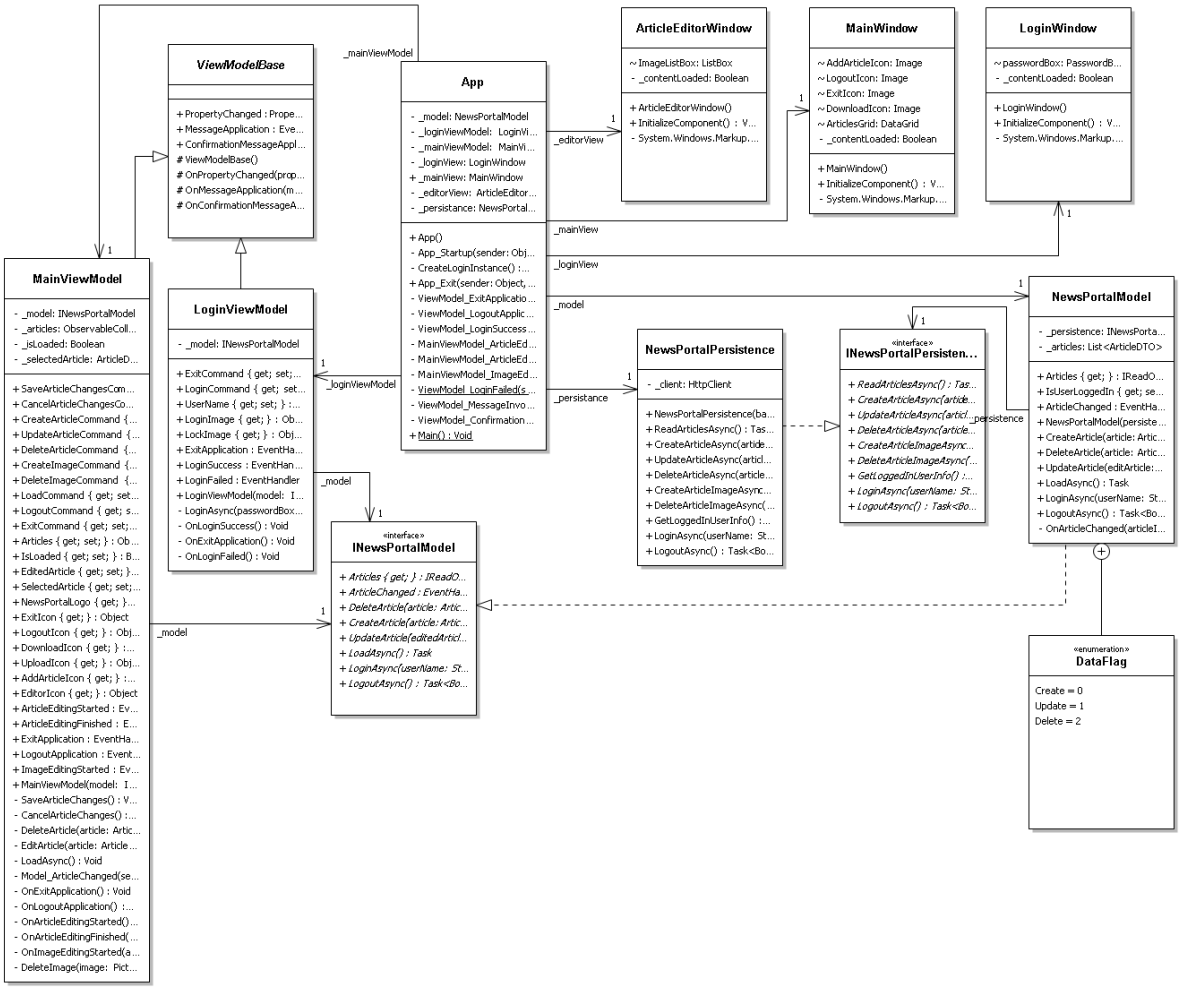
- Három nézet került megvalósításra. Ezek a bejelentkezéshez, fő ablakhoz és szerkesztőablakhoz tartozó nézetek.
- A bejelentkező nézeten felhasználónév és jelszó bevitele után a bejelentkezésre kattintva jelentkezhetünk be az alkalmazásba
- A fő ablakban a saját cikkek egy datagridben kerülnek felsorolásra dátum szerint (cím, író, dátum), amelyeket módosíthatunk, illetve törölhetünk is.
- A fő ablak menüpontjaiban megtaláljuk a kijelentkezés, kilépés és az adatbázis frissítése menüpontokat.
- A fő ablak három további gombot jelenít meg, melyek az New Article, Edit Article és a Delete Article. Melyekkel rendre új cikket hozhatunk létre, szerkeszthetünk egy cikket illetve törülhetünk egy cikket.
- A New Article gombra kattintva megjelenik az új cikk létrehozására alkalmas nézet. Itt meg kell adnunk a cikk címet, az összefoglalót (maximum 1000 karakter), valamint a teljes szöveget. Ezek kitöltése kötelező. Továbbá a cikk beállítható vezető cikknek, ám ekkor legalább egy kép feltöltése kötelező.
- Egy cikkhez független attól, hogy vezető cikk vagy sem, tetszőleges számú kép feltölthető. A Save-re kattintva az új cikk bekerül az adatbázisba. A Cancel visszavonja a véghezvitt változtatásokat.
- Cikk módosítása során ugyanazt a felületet nyitjuk meg, mint az új hozzáadása során, de már előre kitöltve.



11. ábra Osztálydiagramm (Nézet)

VEZÉRLÉS:

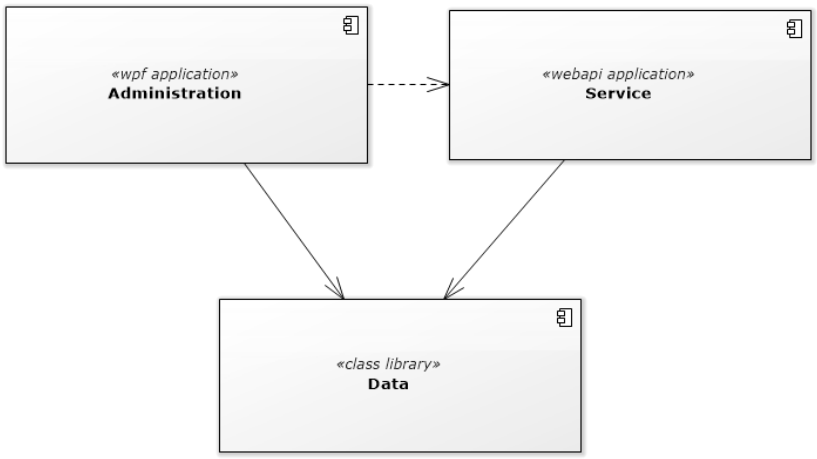
Az App osztály feladata az egyes rétegek példányosítása (App_Startup), összekötése, a nézetmodell, valamint a modell eseményeinek lekezelése, és ezáltal a játék, az adatkezelés, valamint a nézetek szabályozása.



12. ábra Osztálydiagramm (Teljes)

WEBAPI

Az API 3 fő kontrollert tartalmaz. A bejelentkezésen kívül minden akció authentikációhoz kötött. A bemenő paraméterek Data Transfer Objects és megfelelő típusú ID. A visszaküldött Data Transfer Objects-ek mindig csak minimális adatokat tartalmaznak.



13. ábra Kombonens diagramm

Tehát az alábbi a pontokon kommunikál az API.

ACCOUNTCONTROLLER

Get:

- api/Account/Login/{userName}/{userPassword}:
  - Bejelentkezést biztosító akció
- api/Account/Logout
  - Kijelentkezést biztosító akció

ARTICLESCONTROLLER

Get:

- api/Articles
  - Visszaadja a bejelentkezett felhasználó által írt cikkeket
- api/Articles/{id}
  - Visszaadja az id-hez tartozó cikket, ha azt a bejelentkezett felhasználó írta

Post

- api/Articles
  - Új cikk felvételét biztosító akció

Put:

- api/Articles/{id}
  - Adott cikk módosítását biztosító akció

Delete:

- api/Articles/{id}
    - Adott cikk törlését biztosító akció

PICTURESCONTROLLER

Get:

- api/Pictures
    - Visszaadja az összes képet
- api/Pictures/{id}
    - Visszaadja az id-hez tartozó képet

Post

- api/Pictures
    - Új kép felvételét biztosító akció

Put:

- api/Pictures/{id}
    - Adott kép módosítását biztosító akció

DELETE:

- api/Pictures/{id}
    - Adott kép törlését biztosító akció

TESZTELÉS:

A WebAPI funkcionalitása egységtesztek segítségével lett ellenőrizve a NewsPortalWebAPITest osztályban. A teszteléshez teszt adatokat használok és memóriában tárolt adatbázist. Egy MOQolt UserManager és SingInManager került felhasználásra, hogy az API számára úgy tűnjön, hogy egy bejelentkezett, adminisztrátortól jönnek a kérések. Az alábbi tesztesetek kerültek megvalósításra:

- ArticleTest
    - Cikkek lekérdezésének tesztelése
- CreateArticle
    - Új cikke létrehozásának tesztelése
- UpdateArticle
    - Adott cikk frissítése az API-n keresztül
- GetArticleById
    - Adott cikk lekérése
- DeleteArticle
    - Cikkek törlésének tesztje