Horváth Milán **2.Beadandó/4.Feladat**
2019. november 8.
B1V655

1. Feladatleírás

Az asztali grafikus felületet az alkalmazottak használják a rendelések, illetve a weblap tartalmának adminisztrálására.

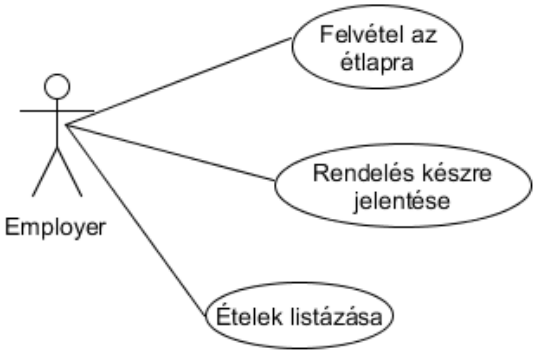
- Az alkalmazott bejelentkezhet (felhasználónév és jelszó megadásával) a programba, illetve kijelentkezhet.
- Bejelentkezve listázódnak a leadott, illetve teljesített rendeléseket (leadás időpontja, teljesítés időpontja, név, cím, telefonszám, összeg), egy rendelést kiválasztva pedig listázódnak a tételeket. A leadott rendelés teljesítettnek jelölhető, ekkor a rendszer rögzíti a teljesítés időpontját is. A lista szűrhető csak teljesített, illetve csak leadott rendelésekre, továbbá a rendelő nevére, illetve cím(részlet)re.
- Lehetőség van új étel, illetve ital hozzáadására (név, ár, illetve étel esetén leírás, csípős/vegetáriánus tulajdonságok megadásával). Az egyértelműség miatt nem engedélyezett több ugyanolyan nevű étel/ital felvitele.

Az adatbázis az alábbi adatokat tárolja:

- kategóriák (név);
- ételek és italok (név, kategória, leírás, ár, csípős-e, vegetáriánus-e);
- munkatársak (teljes név, felhasználónév, jelszó);
- rendelések (név, cím, telefonszám, megrendelt ételek és italok, teljesített-e).

2. Elemzés

- Az alkalmazás perzisztencia részét egy külső C# könyvtárként kezeljük. A weblap részt MVC segítségével hozzuk létre. A grafikus felületet WPF-el MVVM architektúrában megvalósítva.
- Több ablakra is szükségünk van:
  - Bejelentkező felület
  - Egy menüablak, ahol kiválaszthatjuk melyik funkciót szeretnénk használni
  - Rendelések listázására szolgáló ablak.
  - Új étel felvételére szolgáló ablak.
- Az alkalmazáshoz szükséges egy adatbázis is. A feladatban leírt adatbázis jól megfogalmazható SQL adatbázisként.



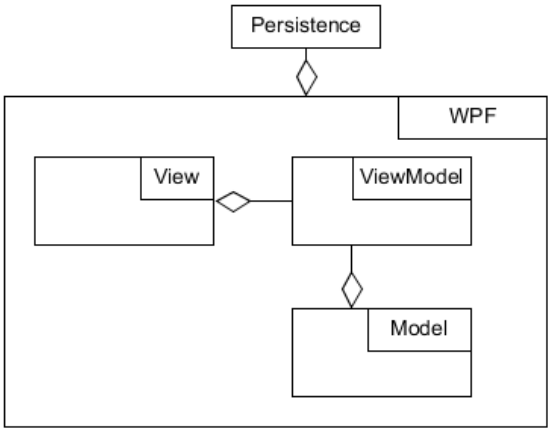
1. ábra. Felhasználói esetdiagram

3. Tervezés

3.1. Programszerkezet

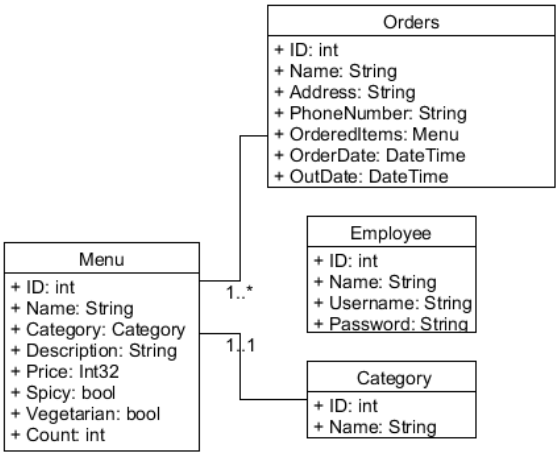
Programszerkezet

- A programot MVVM architektúrával valósítjuk meg.
- A feladathoz a programot két projektre osztjuk. Egy a Order.Persistence, mely az adatbázisát és a WebAPI-nak átadott típusait kezeli.
- A másik projekt az Order.WPF felelős az app futtatásáért, illetve a funkcióiért. Itt találhatóak a View-ViewModel párok, melyekkel megvalósítjuk a különböző oldalakat.



2. ábra. A WPF app és a perzisztencia szerkezete

3.2. Perzisztencia



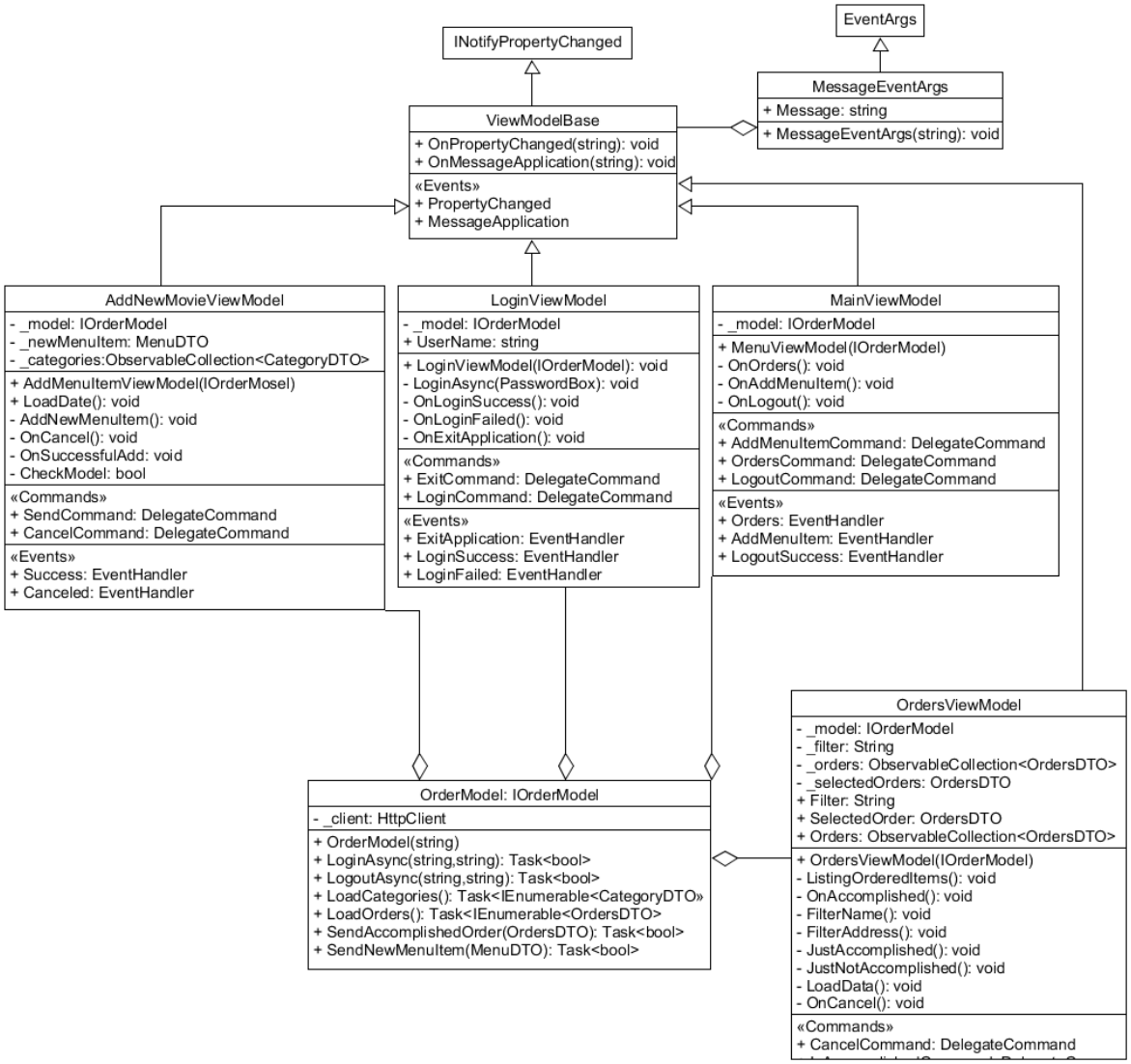
3. ábra. Az adatbázis kapcsolatok, és táblák

3.3. WPF alkalmazás

- WPF alkalmazás MVVM architektúrában megvalósítva
- Model, mely a szerver oldali hívásokat végzi
- Szükséges nézetek:
  - Login - Bejelentkezés
  - Menu - Főmenü
  - AddNewMenuItem - Új étel felvétele az étlapra
  - Orders - A megrendelések listázása, azok lezárása
- Minden nézethez tartozik egy ViewModel

3.4. WebAPI

- api/Account/{username}/{password}:
  HttpGet - Bejelentkezés
- api/Account/Logout:
  HttpGet - Kijelentkezés
- api/Menu
  HttpPost - Új étel felvétele a Menüre
- api/Category/CategoryList
  HttpGet - A kategóriák listázásához szükséges lekérés
- api/Orders/OrdersList
  HttpGet - A megrendelések listázásához szükséges lekérés
- api/Orders
  HttpPost - Megrendelés készre jelentése.



4. ábra. WPF alkalmazás osztálydiagramja

4. Tesztelés

A webszolgáltatás tesztelése Unit teszteken keresztül történt, egy memóriában létrehozott adatbázis használatával.

- *GetOrdersTest:* Megrendelések lekérdezése.
- *GetCategoryTest:* Kategóriák lekérdezése.
- *AddNewMenuItem:* Új étel hozzáadásának tesztelése.
- *IsAccompliched:* Készre jelentés tesztelése.