**Progtech beadandó**

## Készítette: Szalóki Dávid, Seres Péter , Kerepesi Gergő

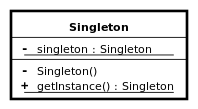
# Projetröl röviden:

Szeretnénk készíteni egy Vatera alkalmazást , ami több Raktárat , Felhasználót , és Terméket fogunk kezelni.

# Csomagok:

C# singleton + Diszítő tervezési minták

Terv : Singleton használata adatbázis kapcsolat létrehozásához



# Osztályok:

**Interfészek**: A különböző szinteken található interfészeket csak azon osztály implementálják, amiknek szükségük van a bennük lévő metódusokra.

**Absztrakt osztályok**: A legutolsó gyermek osztály kivételével, minden szülő absztrakt. Ennek célja, hogy a közös tulajdonságok összegyűjtve legyenek, így megspórolásra kerülnek a fölösleges ismétlések. Ennek lényege, hogy az azonos tulajdonsággal rendelkező dolgok kódjai csak egyszer legyenek megírva, viszont ha a tárgyak különböznek (például a leírásaik), akkor azt a legalsó szinten lehessen kifejteni.

Az osztályokhoz kapcsolódóan felhasználtuk az **OOP alapelveket**

**-Egységbezárás (Encapulation) :** Az adattagokatés a hozzájuk kapcsolódó műveleteket egy egységbe zártuk (class)

**-Öröklődés (Inheritance) :** A gyermek osztály az ős minden mezőjét és metódusát megörökli

**-Többalakúság (Polymorphism) :** A tervezési minták szinte mindegyike ezen alapszik , maga a többalakúság az öröklődés következménye. Mivel a gyermek örökli az ős felületét igy megkapja azok típusait is , igy az objektum több alakban is használható

# Felhasznált tervezési minták:

**Singleton:** Olyan osztályok, amelyeket csak egyszer van szükségünk példányosítani mert csak a metódusokra van belőlük szükség. Nem hozható belőlük létre új példány PL: különböző storagek vagy factoryk. Kiegészítés : többszálú alkalmazás esetén le kell védeni a péndányosítást.