# Zadanie 2

Jesteś poproszony o refaktoryzację klasy UserService i poprawienie metody AddUser. Zakładamy, że obecny kod realizuje konkretną logikę biznesową, która nie powinna ulec zmianie. Jedynym celem zadania jest poprawnie jakości kodu zgodnie z zasadami SOLID, KISS, DRY i YAGNI. Pamiętaj, że logika aplikacji nie powinna się zmienić.

**Ograniczenia**

Klasa Program.cs w projekcie LegacyApp.Consumer NIE MOŻE BYĆ MODYFIKOWANA. Zakładam, że interfejs metody UserService nie może ulec modyfikacji, ponieważ klasa ta jest częścią większego, złożonego systemy. Musimy zachować pełną kompatybilność wsteczną.

Możesz modyfikować dowolne klasy w projekcie LegacyApp z pominięciem klasy **UserDataAccess**. Klasa ta I metoda AddUser powinny pozostać statyczne. Nie możemy modyfikować tej klasy.

# Task 2

You are asked to refactor the UserService class and more specifically its AddUser method. Assume that the code in sound in terms of business logic and only focus on applying clean code principles. Keep in mind acronyms such as SOLID, KISS, DRY and YAGNI.

**Limitations**

The Program.cs in the LegacyApp.Consumer shall NOT CHANGE AT ALL. This includes using statements. Assume that this codebase is part of a greater system and any non-backwards compatibile change will break the system.

Any sort of change in that class will result in you instantly failing this test.

You can change anything in the LegacyApp project except for the UserDataAccess

class and its AddUser method. Both NEED to stay static.