CRUD приложение с ORM

Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

CRUD операции

Create, Read, Update, Delete

Какво са CRUD операциите?

- **C**RUD – Създаване /**Create**/ - пример: добавяне на нов запис в таблицата
- C**R**UD – Прочитане /**Read**/ - пример: прочитане на запис от таблицата
- CR**U**D – Променяне /**Update**/ - пример: промяна на един или няколко елемента от запис в таблицата
- CRU**D** – Изтриване /**Delete**/ - пример: изтриване на запис в таблицата

Защо ORM?

Вече създадохме едно просто CRUD приложение без ORM. Една погрешно написана заявка би създала доста проблеми.
Нека сега да разгледаме няколко предимства на ORM:

- **ORM** прави работата с БД по-лесна и бърза – не пишем собствен SQL код за стандартни заявки
- **По-лесен** за промяна и поддръжка код
- **Защита** от SQL инжекция, чрез филтриране на данните

Архитектура на приложението

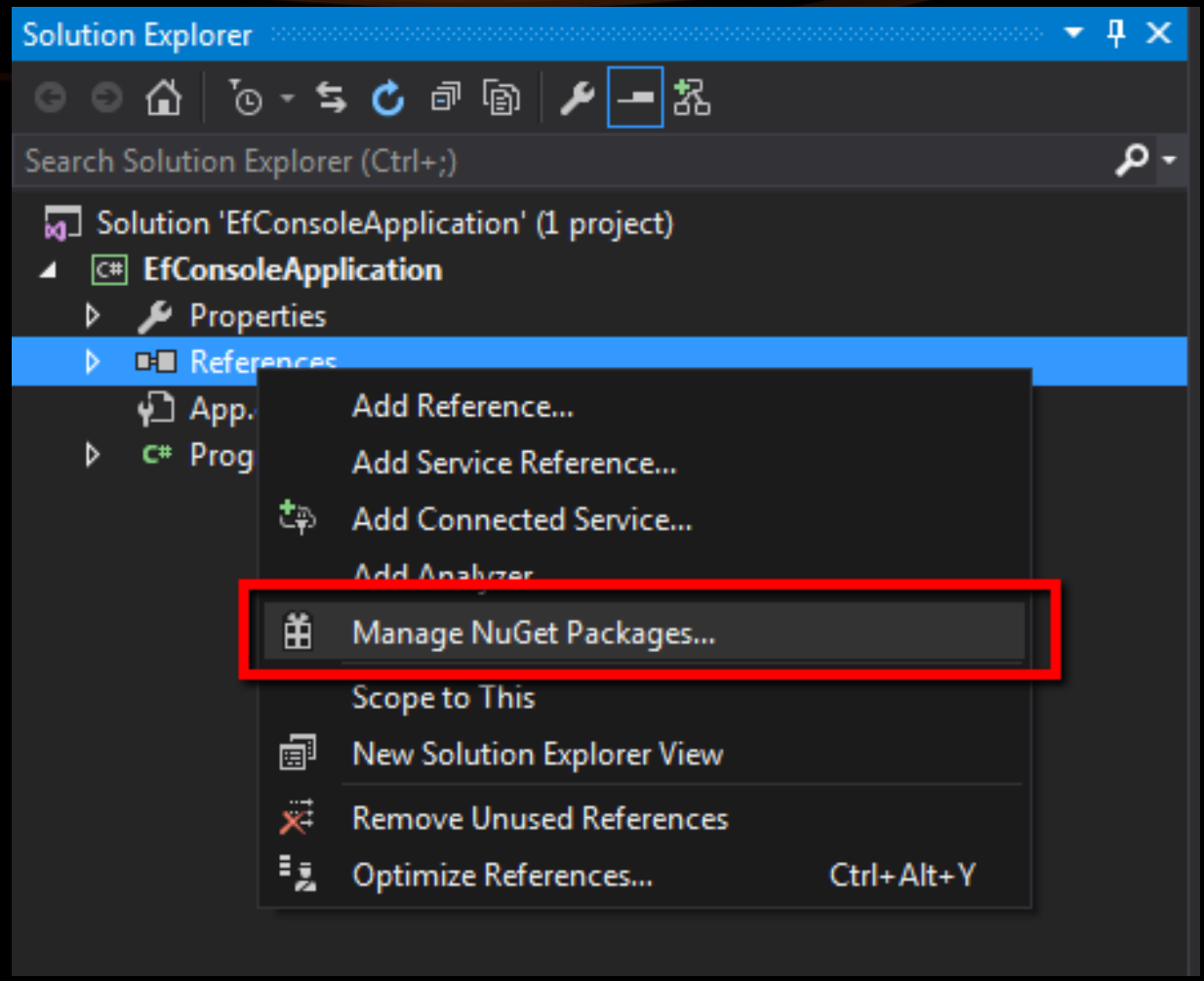
- Ще създадем **CRUD** приложение, което ще управлява **продукти**
- Трислойна архитектура /**3-tier architecture**/
  - Слой за данни /**Data access layer**/
  - Бизнес слой /**Business layer**/
  - Презентационен слой /**Presentation layer**/

Структура на проекта

- Business
  - **ProductBusiness.cs** – описва бизнес логиката свързана с продуктите
- Data
  - **ProductContext.cs** – описва контекста за достъпване на данните
  - Model/**Product.cs** – описва модела данни за продукт
- Presentation
  - **Display.cs** – клас, в който ще реализираме конзолно управление на приложението
- **Program.cs** – Място, където ще създадем обект от класа Display

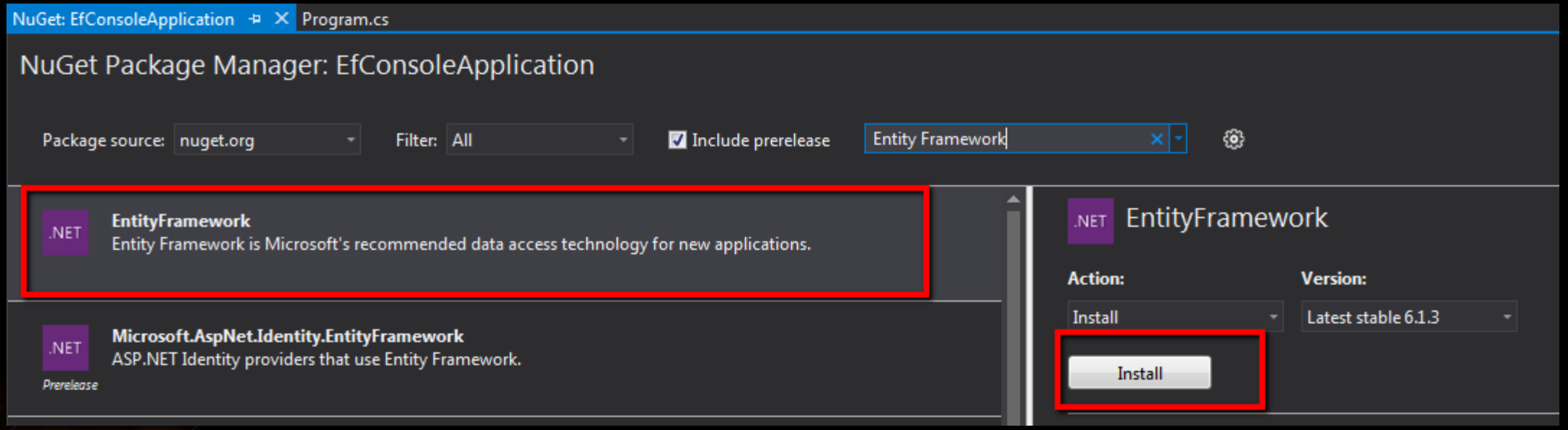
Entity Framework

- Тук ще ползваме Entity Framework
  - Трябва да го добавим към проекта десен бутон на [References] -> Manage NuGet Packages...:



Entity Framework (2)

- Изберете EntityFramework и инсталирайте:



База данни

- Тук ще действаме по Code-first принципа и няма да създаваме предварително база данни. Само ще зададем в App.config файла следният низ за връзка към нея:

```
<connectionStrings>
   <add name="ProductContext" connectionString="Data Source=.;
Initial Catalog=ProductDb; Integrated Security=true"
providerName="System.Data.SqlClient" />
  </connectionStrings>
```

- Поставете този код непосредствено преди </configuration>

Data/Model/Product.cs

- Създайте подпапка Model в Data, а в нея модел клас **Product.cs**

```
public class Product
{
        public int Id { get; set; }
        public string Name { get; set; }
        public decimal Price { get; set; }
        public int Stock { get; set; }
}
```

Data/ProductContext.cs

- Създайте ProductContext клас, който наследява DbContext. Ще има нужда да добавите и using директива:

```
using System.Data.Entity;

public class ProductContext : DbContext
{
        public ProductContext()
            : base("name=ProductContext")
        {
        }
        public DbSet<Product> Products { get; set; }
}
```

Business/ProductBusiness.cs

- Създаваме управляващо поле от клас ProductContext:

```
private ProductContext context;
```

- Създаваме методи, които извършват желаните операции, извиквайки свойството Products и методи върху него, познати ни от LINQ:

```
public List<Product> GetAll(){
  using (productContext = new ProductContext())
  {
    return productContext.Products.ToList();
  }
}
```

- По аналогичен начин реализираме другите методи

Реализиране на презентационен слой

- В рамките на класа Display.cs реализирайте презентационен слой, който да предлага различни операции на потребителя:

```
------------------------------------
                MENU
------------------------------------
1. List all entries
2. Add new entry
3. Update entry
4. Fetch entry by ID
5. Delete entry by ID
6. Exit
```

Реализиране на презентационен слой (2)

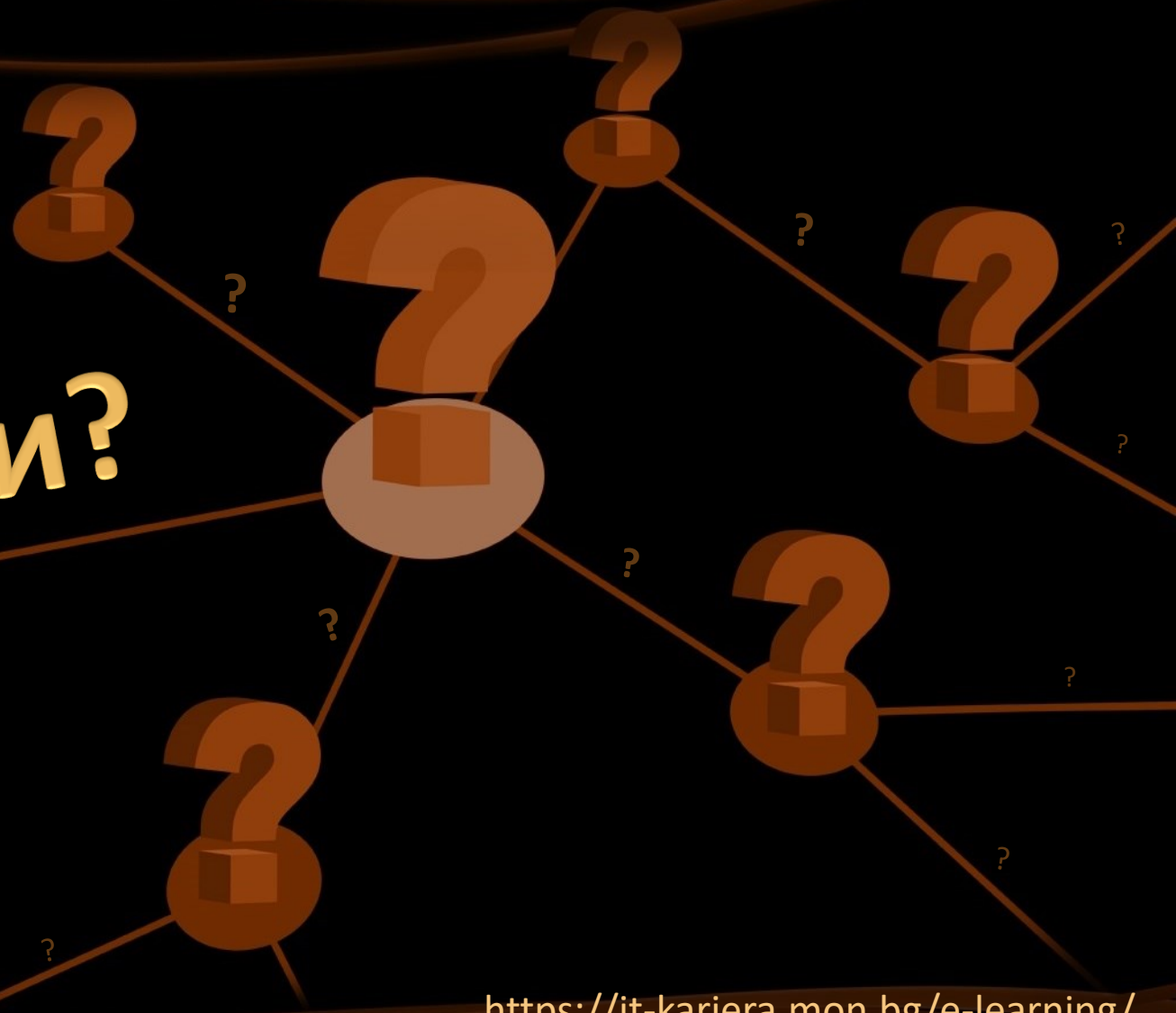
- private ProductBusiness productBusiness = new ProductBusiness() – създава се обект от бизнес класа, който се използва за извикване на съответната бизнес логика
- private void ShowMenu() – метод, който визуализира възможните операции за избор
- private void Input() – метод, който въвежда желаната операция от потребителя и извиква някой от останалите методи, реалзиращи отделните операции

Реализиране на презентационен слой (3)

- void ListAll() – визуализира всички данни
- void Add() – въвежда информация за продукт и я предава за съхранение
- void Update() – приема id, въвежда информация за него и предава за съхранение променената информация
- void Fetch() – извлича информация за продукт по id
- void Delete() – изтрива информация за продукт по id

CRUD приложнеие с ORM

Въпроси?



https://it-kariera.mon.bg/e-learning/

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**