Szkolenie techniczne 2
ASP.NET MVC

Mgr Tomasz Marek

Zajęcia 3

Zadanie do wykonania

- Połączenie modelu aplikacji z bazą danych za pośrednictwem Entity Framework

Kroki do wykonania

- Utworzenie lokalnej bazy danych
- Dodanie do projektu za pomocą NuGet Entity Framework
- Dodanie kontekstu bazy danych
- Utworzenie repozytorium danych
- Uruchomienie aplikacji

Utworzenie bazy danych

- Baza danych powinna zostać utworzona w oparciu o VS. Menu View/SQL Server Object Explorer
- Do bazy danych dodajemy tabelę Products

```
CREATE TABLE [dbo].[Products] (
[ProductId] INT           NOT NULL,
[Author]    VARCHAR (100) NOT NULL,
[Album]     VARCHAR (200) NOT NULL,
[Price]     DECIMAL(16, 2)         NOT NULL,
[Category]  VARCHAR (100) NOT NULL,
PRIMARY KEY CLUSTERED ([ProductId] ASC) );
```

- Następnie uzupełniamy tabelę naszymi danymi testowymi (menu kontekstowe na tabeli i opcja View Data

Utworzenie kontekstu bazy danych

- W folderze **Infrastructure** tworzymy klasę SklepMuzycznyDbContext. Klasa będzie kojarzyła nasz model danych z bazą

```
using System;
using System.Collections.Generic;
using System.Linq; using System.Web;
using System.Data.Entity;
using SklepMuzyczny.Models;

namespace SklepMuzyczny.Infrastructure
{
        public class SklepMuzycznyDbContext : DbContext
        {
                public DbSet<Product> Products { get; set; }
        }
}
```

- W pliku web.config modyfikujemy sekcję connectionStrings. Jako nazwę połączenia podajemy nazwę naszego kontekstu (dodanego powyżej, natomiast w connectionString wstawiamy dane odczytane z właściwości bazy danych

Utworzenie repozytorium produktów

- W folderze Infrastructure tworzymy klasę ProductRepository dziedziczącą po naszym interfejsie IProductRepository. W klasie tworzymy obiekt naszego kontekstu bazy danych

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using SklepMuzyczny.Models;

namespace SklepMuzyczny.Infrastructure
{
	public class ProductRepository : IProductRepository
	{
		SklepMuzycznyDbContext context = new SklepMuzycznyDbContext();
		public IQueryable<Product> Products
		{
			get { return context.Products; }
		}
	}
}
```

Powiązanie repozytorium z kontrolerem

- W dodanej wcześniej klasie ProductControllerFactory modyfikujemy metodę AddBindings tak, aby teraz przekazywała do kontrolera dane z bazy

```
void AddBindings()
{
        ninjectKernell.Bind<IProductRepository>().To<ProductRepository>();
}
```