

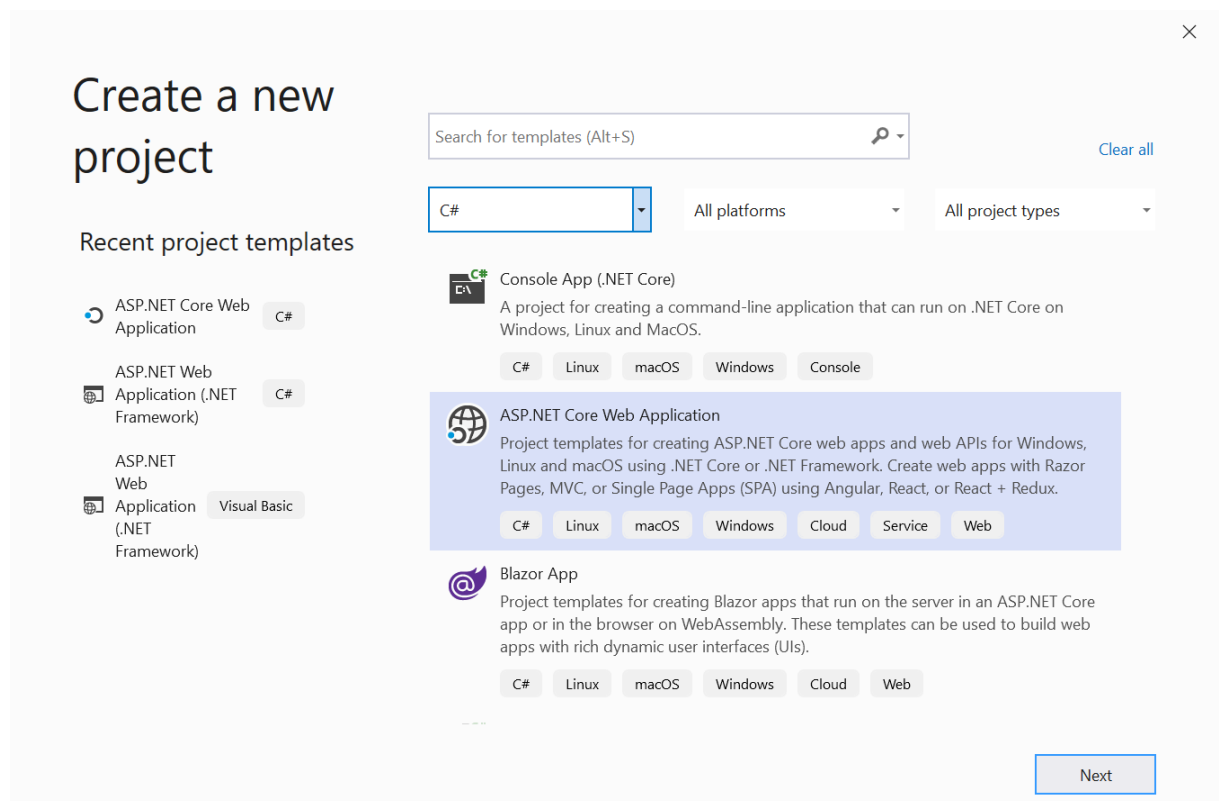
Microsoft .NET: ASP.NET MVC + Entity Framework (Code First)

Do realizacji projektu potrzebne jest zintegrowane środowisko programistyczne Microsoft Visual Studio 2019.

W ramach projektu budowana jest prosta aplikacja wykorzystująca framework MVC, bazująca na modelu bazy danych utworzonym zgodnie ze strategią Code First. Aplikacja zostanie wygenerowana kreatorem MVC scaffolding.

1. Utworzenie nowego projektu.

- a) Uruchom narzędzie Microsoft Visual Studio.
- b) Z menu głównego wybierz File→New Project. Wybierz szablon ASP.NET Core Web Application dla języka C# (szczególną uwagę zwróć na wybór właściwego języka). Kliknij przycisk Next.



- c) W drugim kroku kreatora jako nazwę projektu podaj „MvcNews”. Pole Place solution and project in the same directory pozostaw niezaznaczone. Pozostałe opcje pozostaw domyślne i kliknij przycisk Create.

×

Configure your new project

ASP.NET Core Web Application C# Linux macOS Windows Cloud Service Web

Project name

MvcNews

Location

C:\Users\marek\source\repos

...

Solution name ⓘ

MvcNews

☐ Place solution and project in the same directory

Back

Create

- d) W oknie wyboru frameworków wybierz Web Application (Model-View-Controller). Odznacz pole wyboru Configure for HTTPS. Pozostałe opcje pozostaw domyślne i kliknij przycisk Create.

Create a new ASP.NET Core web application

.NET Core ASP.NET Core 3.1

Empty

An empty project template for creating an ASP.NET Core application. This template does not have any content in it.

API

A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.

Web Application

A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.

Web Application (Model-View-Controller)

A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

Angular

A project template for creating an ASP.NET Core application with Angular

React.js

Get additional project templates

Authentication

No Authentication

[Change](#)

Advanced

☐ Configure for HTTPS

☐ Enable Docker Support

(Requires Docker Desktop)

Linux

☐ Enable Razor runtime compilation

Author: Microsoft

Source: .NET Core 3.1.4

Back

Create

- e) Obejrzyj strukturę projektu w panelu Solution Explorer zwracając uwagę na utworzone przez kreator foldery.
- f) Obejrzyj kod startowy aplikacji w pliku Program.cs. Zawiera on konfigurację domyślnego serwera do uruchomienia aplikacji.
- g) Obejrzyj kod odpowiedzialny za inicjalizację aplikacji w pliku Startup.cs. Odszukaj metodę zawierającą konfigurację usług wymaganych przez aplikację. Odszukaj metodę odpowiedzialną za konfigurację obsługi żądań HTTP, a w niej definicję reguły routingu z charakterystycznym dla ASP.NET MVC schematem adresów URL.
- h) Uruchom projekt kombinacją klawiszy Ctrl+F5 (Start Without Debugging).
- i) Przetestuj nawigację po stronach aplikacji zwracając uwagę na zawartość paska adresu w przeglądarce.

2. Modyfikacja obsługi strony z polityką prywatności wygenerowanej przez kreator, w celu zapoznania się ze współpracą kontrolera z widokiem.

- a) Otwórz do edycji klasę kontrolera HomeController i wstaw poniższą linię kodu na początku ciała metody Privacy():

```
ViewData["Policy"] = "Please do not enter any sensitive information.";
```

- b) Otwórz do edycji stronę widoku wyświetlającą politykę prywatności i zastąp w niej statyczną zawartość elementu <p> poniższym kodem:

```
@ViewData["Policy"]
```

- c) Odśwież stronę w przeglądarce i sprawdź czy tekst polityki został poprawnie przekazany do widoku i wyświetlony.
- d) Zmień kod zarówno kontrolera jak i widoku tak, aby przekazanie tekstu polityki było realizowane poprzez obiekt ViewBag. Odśwież stronę w przeglądarce by upewnić się, że aplikacja działa tak samo jak wcześniej.

Komentarz: ViewBag stanowi wygodną abstrakcję nad ViewData, ale trzeba pamiętać, że nie jest dostępny w modelu programistycznym Razor Pages.

3. Utworzenie obiektowego modelu danych w projekcie zgodnie ze strategią Code First.

- a) W panelu Solution Explorer wywołaj prawym klawiszem myszy menu kontekstowe dla folderu Models projektu i wybierz opcję Add -> Class. Jako nazwę pliku podaj NewsItem.cs.
- b) W utworzonej klasie zdefiniuj jako publiczne poniższe właściwości (właściwości w sensie składni C#, a nie pola!):

typ	nazwa
int	Id
DateTime	TimeStamp
string	Text

4. Dodanie do projektu pakietów Entity Framework i dostawcy usług dla SQL Server.

- a) Uruchom konsolę menedżera pakietów opcją Tools -> NuGet Package Manager -> Package Manager Console z głównego menu.
- b) Z poziomu konsoli uruchom następującą komendę:

```
Install-Package Microsoft.EntityFrameworkCore.SqlServer
```

Komentarz: Powyższa komenda instaluje pakiet dostawcy EF Core dla serwera SQL Server, ale spowoduje to zainstalowanie również pakietu EF Core jako jego zależności.

5. Utworzenie klasy kontekstu bazy danych.

- a) Utwórz w strukturze projektu folder Data opcją Add -> New Folder z poziomu węzła projektu w Solution Explorer.
- b) W utworzonym folderze utwórz klasę o nazwie NewsDbContext, a następnie podmień jej definicję na poniższą (z zachowaniem utworzonej przez kreator przestrzeni nazw).

```
public class NewsDbContext : DbContext
{
    public NewsDbContext(DbContextOptions< NewsDbContext > options) :
        base(options)
    {}

    public DbSet<NewsItem> News { get; set; }
}
```

Zaimportuj niezbędne przestrzenie nazw.

Komentarz: Konstruktor z parametrem DbContextOptions (zamiast domyślnego bezargumentowego) umożliwi ustawianie opcji konfiguracyjnych kontekstu podczas jego rejestracji w aplikacji.

- c) Przejdź do edycji pliku konfiguracyjnego aplikacji appsettings.json. Dodaj na końcu (przed ostatnim nawiasem klamrowym, oddzielając przecinkiem od poprzedniej opcji) poniższą sekcję z łańcuchem połączenia z bazą danych:

```
"ConnectionStrings": {
  "NewsDbContext":
    "Server=(localdb)\\mssqllocaldb;Database=News;Trusted_Connection=True"
}
```

Komentarz: Podany łańcuch połączeniowy zakłada wykorzystanie instancji serwera SQL Server w wersji LocalDb, odpowiedniej dla zastosowań deweloperskich, instalowanej wraz z Visual Studio. Należy zauważyć, że wskazana w parametrach połączenia baza danych jeszcze nie istnieje. Zostanie ona utworzona później zgodnie ze strategią Code First.

Konieczna jest jeszcze rejestracja kontekstu bazy danych jako jednej z usług wykorzystywanych w aplikacji. W tym celu w klasie Startup na końcu metody ConfigureServices dodaj poniższy kod:

```
services.AddDbContext<NewsDbContext>(options =>
    options.UseSqlServer(Configuration
        .GetConnectionString("NewsDbContext"));
```

- d) Po dodaniu powyższej instrukcji, do listy instrukcji using dodaj instrukcję importującą przestrzeń nazw klasy kontekstu bazy danych oraz poniższą instrukcję importującą przestrzeń nazw EF Core:

```
using Microsoft.EntityFrameworkCore;
```

- e) Zapisz wszystkie zmiany.
f) Przebuduj projekt, aby upewnić się, że nie zawiera błędów (Build).

6. Wygenerowanie kontrolerów i widoków do obsługi modelu.

- a) Z menu kontekstowego węzła Controllers w panelu Solution Explorer wybierz opcję Add -> Controller (ewentualnie poprzez opcję Add New Scaffolded Item). W kreatorze kontrolera wybierz szablon scaffoldingu MVC controller with views, using Entity Framework. Kliknij przycisk Add i w kolejnym kroku kreatora wybierz utworzoną wcześniej klasę modelu i kontekstu bazy danych oraz popraw nazwę klasy kontrolera na NewsController. Pozostałe opcje pozostaw domyślne i kliknij przycisk Add.

The screenshot shows a dialog box titled "Add MVC Controller with views, using Entity Framework". It contains the following fields and options:

- Model class:** A dropdown menu showing "NewItem (MvcNews.Models)".
- Data context class:** A dropdown menu showing "NewsDbContext (MvcNews.Data)" with a "+" button to the right.
- Views:** A section with three checked checkboxes:
 - ☒ Generate views
 - ☒ Reference script libraries
 - ☒ Use a layout page:
- Below the checkboxes is an empty text box with a "... button" and the instruction "(Leave empty if it is set in a Razor _viewstart file)".
- Controller name:** A text box containing "NewsController".
- At the bottom right are "Add" and "Cancel" buttons.

- b) Obejrzyj wygenerowaną klasę kontrolera i związane z nim widoki. W klasie kontrolera zwróć uwagę na to, że metody akcji, które realizują operacje na bazie danych są asynchroniczne (i wykorzystują asynchroniczne metody Entity Framework). Zwiększa to skalowalność aplikacji, gdyż wątki usługowe serwera nie będą blokowane oczekiwaniem na odpowiedź bazy danych, a będą mogły być wykorzystane do obsługi innych żądań.
- c) Odszukaj w strukturze projektu stronę wzorcową _Layout.cshtml i dodaj w menu, które wyświetla się na każdej stronie u góry, link wywołujący akcję Index kontrolera News.
- d) Przebuduj projekt (Rebuild).

7. Utworzenie bazy danych przy użyciu migracji.

- a) Przejdź do konsoli menedżera pakietów (lub otwórz ją ponownie, jeśli została zamknięta: Tools->NuGet Package Manager->Package Manager Console).
- b) Wykonaj w oknie Package Manager Console kolejno poniższe komendy.

```
Add-Migration InitialCreate
```

```
Update-Database
```

- c) Obejrzyj treść metod Up i Down w wygenerowanej klasie migracji, służących w tym wypadku odpowiednio do utworzenia i usunięcia tabeli w bazie danych.

8. Uruchomienie aplikacji i przetestowanie jej działania.

- a) Uruchom aplikację (Ctrl+F5)
- b) Przetestuj działanie aplikacji dodając, edytując, przeglądając i usuwając dane.

9. Ustawienie domyślnej daty dodawanych newsów na bieżącą.

- a) W metodzie Create() kontrolera NewsController (wywoływanej metodą GET HTTP) utwórz nowy obiekt NewsItem. Ustaw w nim datę na bieżącą (System.DateTime.Now), a następnie przekaz go do widoku.
- b) Uruchom aplikację i przetestuj dodawanie nowych newsów.

10. Dodanie obowiązkowości i weryfikacji długości tekstu newsa.

- a) Dodaj w klasie encji NewsItem następujące adnotacje: (1) włączającą obowiązkowość tekstu newsa ([Required]), (2) weryfikującą, że jego długość wynosi od 5 do 140 znaków ([StringLength]), (3) wskazującą że etykieta czasowa newsa jest typu daty ([DataType]), aby nie wyświetlały się składniki czasu. Nie zapomnij o zaimportowaniu przestrzeni nazw adnotacji.
- b) Uruchom aplikację aby przetestować działanie dodanych adnotacji.

Zadanie do samodzielnego wykonania

1. Wejdź na stronę aplikacji w dwóch przeglądarkach i sprawdź zachowanie aplikacji w przypadku współbieżnej edycji treści tego samego newsa.

2. Włącz mechanizm optymistycznego zarządzania współbieżnością dla encji NewsItem z wykorzystaniem dedykowanej do tego celu właściwości (rowversion):

- dodaj w klasie encji NewsItem publiczną właściwość RowVersion typu byte[] i oznacz ją adnotacją [Timestamp]

- dodaj migrację AddRowVersionMig i uaktualnij bazę danych

- zmień adnotację Bind metody kontrolera do edycji newsa wołanej przez POST, tak aby uwzględniała dodaną do encji właściwość wersji

- spraw aby właściwość wersji była uwzględniona na stronie do edycji newsa jako ukryte pole formularza, analogicznie do identyfikatora encji (bez tego oryginalna wartość właściwości wersji byłaby tracona w procesie edycji danych w aplikacji ASP.NET MVC!)

- uruchom aplikację w dwóch przeglądarkach i ponownie przetestuj współbieżną edycję danych

3. Dodaj obsługę wyjątku optymistycznego zarządzania współbieżnością.

Uwaga: Kreator kontrolera wygenerował instrukcję try/catch łapiącą stosowny wyjątek, ale w przypadku gdy konfliktową operacją była również modyfikacja, a nie usunięcie, wyjątek jest rzucany ponownie.

- odszukaj instrukcję throw rzucającą ponownie wyjątek po jego złapaniu

- zamiast ponownego rzucenia wyjątku, za pomocą metody ModelState.AddModelError, dodaj komunikat informujący o tym że edytowane dane zostały zmodyfikowane lub usunięte przez innego użytkownika. Uwaga: Jako klucz (key) błędu podaj pusty łańcuch znaków, gdyż błąd nie jest związany z żadnym konkretnym atrybutem modelu

- pamiętaj, aby po przekazaniu komunikatu o błędzie za pomocą odpowiedniej instrukcji return sprawić aby nastąpiło ponowne wyświetlenie strony do edycji danych, na której ma być wyświetlony ten komunikat wraz z prezentacją edytowanych danych

- ponownie uruchom aplikację w dwóch przeglądarkach i ponownie przetestuj współbieżną edycję danych zwracając uwagę na własny komunikat o błędzie