Create a new ASP.NET Web Application

×

Empty
An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

Single Page Application
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication
No Authentication
Change

Add folders & core references
☐ Web Forms
☑ MVC
☑ Web API

Advanced
☑ Configure for HTTPS
☐ Docker support
   (Requires Docker Desktop)
☐ Also create a project for unit tests
lab7.Tests

Back   Create

Adaugarea Bazei de Date

In **Web.config** adaugati ConnectionString-ul bazei de date dupa instalare EF si crearea bazei de date.

```
<connectionStrings>
    <add name="DbConnectionString" providerName="System.Data.SqlClient" connectionString=" Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Andreea\source\repos\DAW2020\lab7\lab7\App_Data\La
b7.mdf;Integrated Security=True"/>
</connectionStrings>
```

Modelul Book

In **Models** creati clasa **Book.cs**.

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

namespace lab7.Models
{
    public class Book
    {
        public int BookId { get; set; }
```
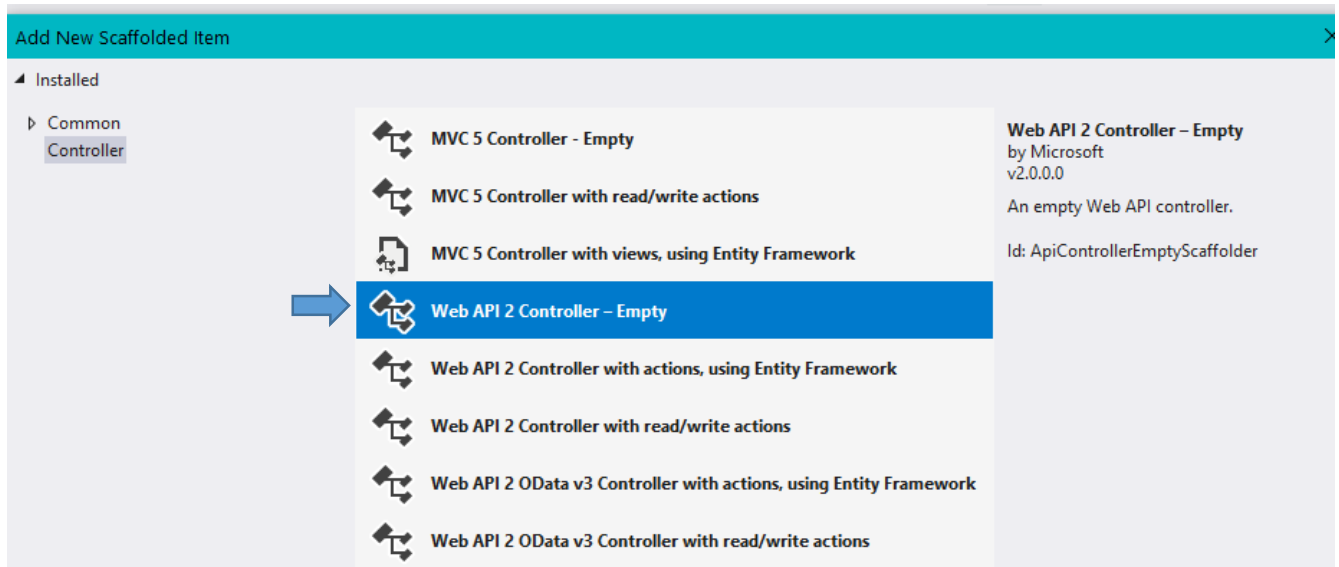
```csharp
        public string Title { get; set; }
        public string Author { get; set; }
    }
    public class DbCtx : DbContext
    {
        public DbCtx() : base("DbConnectionString")
        {
            Database.SetInitializer<DbCtx>(new DropCreateDatabaseIfModelChanges<DbCtx>());
        }
        public DbSet<Book> Books { get; set; }
    }
}
```

Api Controller

In folder-ul **Controllers** creati **BooksController**. La crearea sa selectati 'Web Api 2 Controller – Empty'.

[!] Observati ca **BooksController** nu mai mosteneste clasa **Controller**, ci **ApiController**!

BooksController.cs

```csharp
using lab7.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;

namespace lab7.Controllers
{
    public class BooksController : ApiController
    {
        private DbCtx ctx = new DbCtx();

        // asemanatoare cu actiunea Index pe care o foloseam in proiectele MVC
        public List<Book> Get()
        {
            return ctx.Books.ToList();
        }

        // asemanatoare cu actiunea Details pe care o foloseam in proiectele MVC
        public IHttpActionResult Get(int id)
        {
            Book book = ctx.Books.Find(id);
```

```csharp
            if (book == null)
                // returneaza codul 404
                return NotFound();

            // returneaza codul de succes 200, iar in body vor fi memorate datele obiectului book
            return Ok(book);
        }

        // asemanatoare cu actiunea New pe care o foloseam in proiectele MVC
        public IHttpActionResult Post([FromBody] Book book)
        {
            ctx.Books.Add(book);
            ctx.SaveChanges();

            // helperul Created contine, pe lânga obiectul nou-creat, si adresa la care el va fi gasit
            var uri = new Uri(Url.Link("DefaultApi", new { id = book.BookId }));
            return Created(uri, book);
        }

        // asemanatoare cu actiunea Edit pe care o foloseam in proiectele
        public IHttpActionResult Put(int id, [FromBody] Book b)
        {
            Book book = ctx.Books.Find(id);

            if (book == null)
                return NotFound();

            book.Title = b.Title;
            book.Author = b.Author;
            ctx.SaveChanges();
            return Ok(book);
        }

        public IHttpActionResult Delete(int id)
        {
            Book book = ctx.Books.Find(id);

            if (book == null)
                return NotFound();

            ctx.Books.Remove(book);
            ctx.SaveChanges();
            return Ok(book);
        }
    }
}
```