LABORATOR 8 – Andreea Guster

Pasul 1

Copiem proiectul din laboratorul 7 intr-un folder nou.

Pasul 2

Modificam **Connection String-ul**.

Pasul 3

Ne asiguram ca exista urmatorul cod pentru modelul **Book**.

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

namespace lab7_vineri.Models
{
    public class Book
    {
        public int BookId { get; set; }
        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; }
    }
}
```

Pasul 4

Ne asiguram ca exista urmatorul cod pentru controller-ul **Books**.

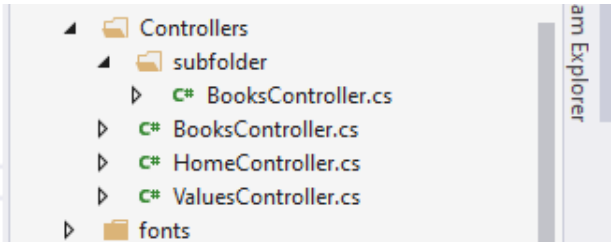
```
using lab7_vineri.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace lab7_vineri.Controllers.subfolder
{
    public class BooksController : Controller
    {
        private DbCtx db = new DbCtx();

        [HttpGet]
        public ActionResult Index()
        {
            List<Book> books = db.Books.ToList();
            ViewBag.Books = books;
            return View();
        }
```

```
public ActionResult Details(int? id)
{
    if (id.HasValue)
    {
        Book book = db.Books.Find(id);
        if (book != null)
        {
            return View(book);
        }
        return HttpNotFound("Couldn't find the book with id " + id.ToString() + "!");
    }
    return HttpNotFound("Missing book id parameter!");
}

[HttpGet]
public ActionResult New()
{
    Book book = new Book();
    return View(book);
}

[HttpPost]
public ActionResult New(Book bookRequest)
{
    try
    {
        if (ModelState.IsValid)
        {
            db.Books.Add(bookRequest);
            db.SaveChanges();
            return RedirectToAction("Index");
        }
        return View(bookRequest);
    }
    catch (Exception e)
    {
        return View(bookRequest);
    }
}

[HttpGet]
public ActionResult Edit(int? id)
{
    if (id.HasValue)
    {
        Book book = db.Books.Find(id);

        if (book == null)
        {
            return HttpNotFound("Coludn't find the book with id " + id.ToString() + "!");
        }
        return View(book);
    }
    return HttpNotFound("Missing book id parameter!");
}

[HttpPut]
public ActionResult Edit(int id, Book bookRequest)
{
    try
    {
        if (ModelState.IsValid)
        {
            Book book = db.Books
                .SingleOrDefault(b => b.BookId.Equals(id));

            if (TryUpdateModel(book))
            {
                book.Title = bookRequest.Title;
```

```
                    book.Author = bookRequest.Author;
                    db.SaveChanges();
                }
                return RedirectToAction("Index");
            }
            return View(bookRequest);
        }
        catch (Exception)
        {
            return View(bookRequest);
        }
    }

    [HttpDelete]
    public ActionResult Delete(int id)
    {
        Book book = db.Books.Find(id);
        if (book != null)
        {
            db.Books.Remove(book);
            db.SaveChanges();
            return RedirectToAction("Index");
        }
        return HttpNotFound("Couldn't find the book with id " + id.ToString() + "!");
    }
  }
}
```

Pasul 4

In folder-ul **Controller**, vom crea un **subfolder** ce va contine tot un **BooksController**, doar ca acest controller mosteneste clasa **ApiController**.




```
using lab7_vineri.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;

namespace lab7_vineri.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);

            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);
        }
    }
}
```

Pasul 5 – Instalam jquery.DataTables

Pentru a încarca date prin AJAX în tabel, vom folosi biblioteca DataTables. O vom instala prin NuGet Package Manager, ca pe Entity Framework (pachetul se numeste jquery.datatables).

Pasul 6 - BundleConfig

In clasa **BundleConfig** adaugati liniile de cod ingrosate.

```
using System.Web;
using System.Web.Optimization;

namespace lab7_vineri
{
    public class BundleConfig
```

```
    {
        // For more information on bundling, visit https://go.microsoft.com/fwlink/?LinkId=301862
        public static void RegisterBundles(BundleCollection bundles)
        {
            bundles.Add(new ScriptBundle("~/bundles/jquery").Include(
                        "~/Scripts/jquery-{version}.js",
                        "~/scripts/datatables/jquery.datatables.js",
                        "~/scripts/datatables/datatables.bootstrap.js"
                        ));

            // Use the development version of Modernizr to develop with and learn from. Then, when you're
            // ready for production, use the build tool at https://modernizr.com to pick only the tests
you need.
            bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(
                        "~/Scripts/modernizr-*"));

            bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(
                      "~/Scripts/bootstrap.js"));

            bundles.Add(new StyleBundle("~/Content/css").Include(
                      "~/Content/bootstrap.css",
                      "~/Content/site.css",
                      "~/content/datatables/css/datatables.bootstrap.css"
                      ));
        }
    }
}
```

Pasul 7 – Adaugare de Views

In **Views/Books** creati view-ul **Details.cshtml**.

```
@model lab7_vineri.Models.Book

<h2>@Model.Title</h2>

@Html.Label("Author", "Author:")
<br />
<p>@Model.Author</p>
```

In **Views/Books** creati view-ul **Index.cshtml**.

```
@{
    ViewBag.Title = "Books";
}

<h2>@ViewBag.Title</h2>

<table id="books" class="table table-bordered table-hover">
    <thead>
        <tr>
            <th>Title</th>
            <th>Details</th>
            <th>Delete</th>
        </tr>
    </thead>
    <tbody>
    </tbody>
</table>

@section scripts
{
    <script>
        $(document).ready(function () {
```

```
            var table = $("#books").DataTable({
                ajax: { url: "/api/books", dataSrc: "" },
                columns: [
                    { data: "Title" },
                    {
                        data: "Title",
                        render: function (data, type, row) {
                            return "<a href='/books/details/" +
                                row.BookId + "'>More about " +
                                row.Title + "</a>";
                        }
                    },
                    {
                        data: "BookId", render: function (data) {
                            return "<button class='bt-delete' data-id=" +
                                data + ">Delete item</button>";
                        }
                    }
                ]
            });

            $("#books").on("click", ".bt-delete",
                function () {
                    var but = $(this);
                    if (confirm("Are you sure?")) {
                        $.ajax({
                            url: "/api/books/" + $(this).attr("data-id"),
                            method: "DELETE",
                            success: function () {
                                console.log("Success");
                                table.row(but.parents("tr")).remove().draw();
                                but.attr("data-id");
                            }
                        });
                    }
                });
        });
    </script>
}
```

In **Views/Books** creati view-ul **New.cshtml**.

```
@model lab7_vineri.Models.Book
@{
    ViewBag.Title = "Create";
}

<h2>@ViewBag.Title</h2>


@Html.Label("Title", "Title:")
<br />
@Html.TextBoxFor(b => b.Title, null, new { placeholder = "Type in the book's title", @class = "form-
control" })
@Html.ValidationMessageFor(b => b.Title, "", new { @class = "text-danger" })
<br />

@Html.Label("Author", "Author:")
<br />
@Html.TextBoxFor(b => b.Author, null, new { placeholder = "Type in the book's author", @class = "form-
control" })
@Html.ValidationMessageFor(b => b.Author, "", new { @class = "text-danger" })
<br />

<button class="btn btn-primary" id="bt-add">Create</button>

@section scripts
{
```

```
    <script>
        $(document).ready(function () {
            $("#bt-add").on("click", function () {
                if (confirm("Are you sure?")) {
                    $.ajax({
                        url: "/api/books",
                        method: "POST",
                        data: JSON.stringify({
                            Title: $("#Title").val(),
                            Author: $("#Author").val()
                        }),
                        contentType: "application/json; charset=utf-8",
                        dataType: "json",
                        success: function () {
                            window.location.href = "/Books/Index";
                        }
                    });
                }
            });
        });
    </script>
}
```

In **Views/Books** creati view-ul **Edit.cshtml**.

```
@model lab7_vineri.Models.Book
@{
    ViewBag.Title = "Edit";
}

<h2>@ViewBag.Title</h2>

@Html.HiddenFor(b => b.BookId)
<br />
<br />

@Html.Label("Title", "Title:")
<br />
@Html.EditorFor(b => b.Title, new { htmlAttributes = new { @class = "form-control" } })
@Html.ValidationMessageFor(b => b.Title, "", new { @class = "text-danger" })
<br />

@Html.Label("Author", "Author:")
<br />
@Html.EditorFor(b => b.Author, new { htmlAttributes = new { @class = "form-control" } })
@Html.ValidationMessageFor(b => b.Author, "", new { @class = "text-danger" })
<br />

<button class="btn btn-primary" id="bt-update">Update</button>


@section scripts
{
    <script>
        $(document).ready(function () {
            $("#bt-update").on("click", function () {
                if (confirm("Are you sure?")) {
                    $.ajax({
                        url: "/api/books/" + $("#BookId").val(),
                        method: "PUT",
                        data: JSON.stringify({
                            Title: $("#Title").val(),
                            Author: $("#Author").val()
                        }),
                        contentType: "application/json; charset=utf-8",
                        dataType: "json",
                        success: function () {
```

```
                        window.location.href = "/Books/Index";
                    }
                });
            }
        });
    });
</script>
}
```