Laborator 3 – Aplicatii Web cu ASP.NET Core si Entity Framework Code First

1. In laboratorul curent, vom dezvolta aplicatia web creata la laboratorul 3. Deschidem Visual Studio si apoi alegem optiunea **Open a project or solution** si cautam proiectul **Nume_Pren_Lab2,** creat anterior. Pentru a putea fi evaluata activitatea aferenta fiecarui laborator, laboratorul curent il vom dezvolta pe un branch nou, diferit de cel master care se afla deja creat. Utilizand pasii indicati in Lab 1, pct. 22 vom crea un branch nou pe care il vom denumi Laborator3
2. Pentru a gestiona categoriile in care se afla fiecare carte cream doua noi entitati: Category si BookCategory, in ferestra Solution Explorer, facem click dreapta pe folderul Models->Add->New Item si alegem Class. Denumim noua entitate Category

3. In clasa creata adaugam urmatoarele proprietati:

```
public class Category
    {
       public int ID { get; set; }

       public string CategoryName { get; set; }

       public ICollection<BookCategory>? BookCategories { get; set; }
    }
```

4. Facem click dreapta pe folderul Models->Add->New Item si alegem Class. Denumim noua entiate BookCategory si adaugam urmatoarele proprietati:

```
public class BookCategory
    {
        public int ID { get; set; }

        public int BookID { get; set; }

        public Book Book { get; set; }

        public int CategoryID { get; set; }

        public Category Category { get; set; }
    }
```

5. Actualizam apoi modelul Books conform codului de mai jos:

```
public class Book
    {
        public int ID { get; set; }

        [Display(Name = "Book Title")]
        public string Title { get; set; }
```

```
        [Column(TypeName = "decimal(6, 2)")]
        public decimal Price { get; set; }

        public DateTime PublishingDate { get; set; }

        public int? AuthorID { get; set; }

        public Author? Author { get; set; }

        public int? PublisherID { get; set; }

        public Publisher? Publisher { get; set; }

        public ICollection<BookCategory>? BookCategories { get; set; }
    }
```

6. In PMC introducem urmatoarele instructiuni

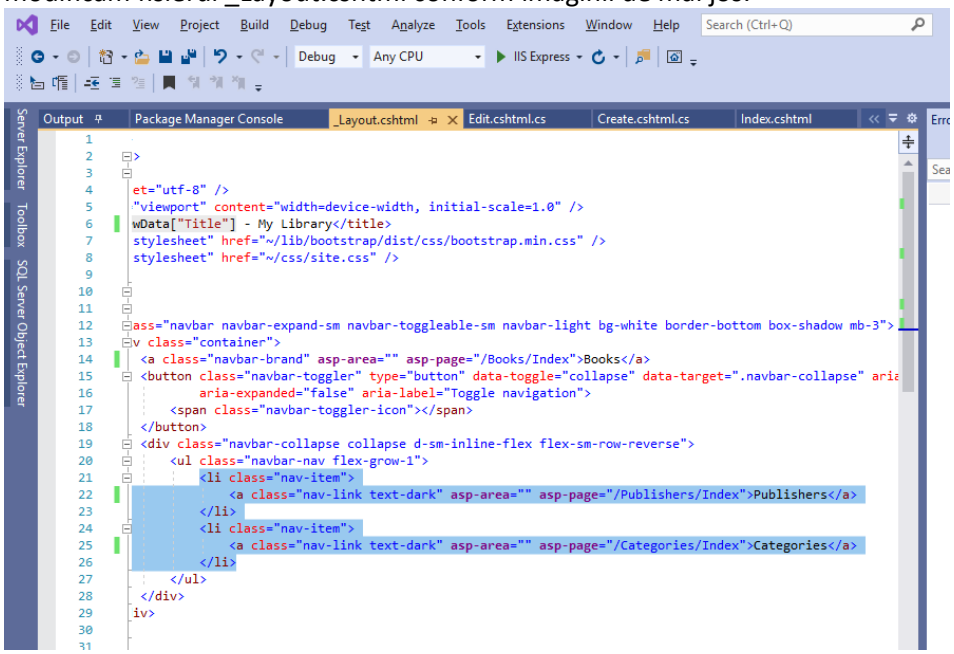
```
Add-Migration BookCategory
Update-Database
```

7. In ferestra **Solution Explorer**, adaugam un subdirector la directorul **Pages** apasand click dreapta pe numele directorului Pages, selectam **Add ->New Folder.** Denumim noul subdirector **Categories**

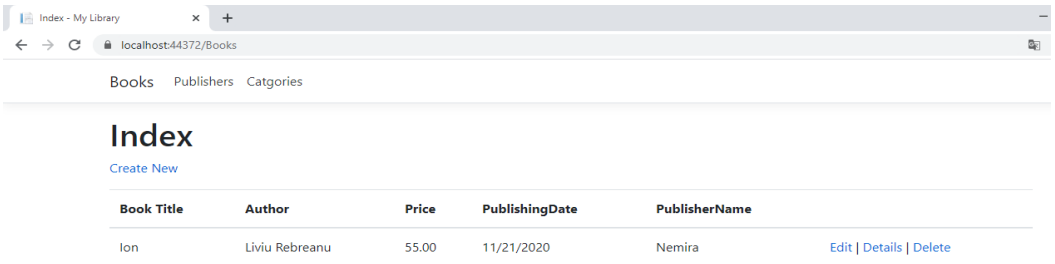
8. Facem click dreapta pe subdirectorul **Pages/Categories** si selectam **Add > New Scaffolded Item** si selectam **Razor Pages using Entity Framework(CRUD)**

9. Observam ca desi am generat pagini pentru tabele Categories si Publishers nu putem naviga catre acestea din meniul existent in aplicatia noastra. Pentru a avea legaturi spre paginile nou create

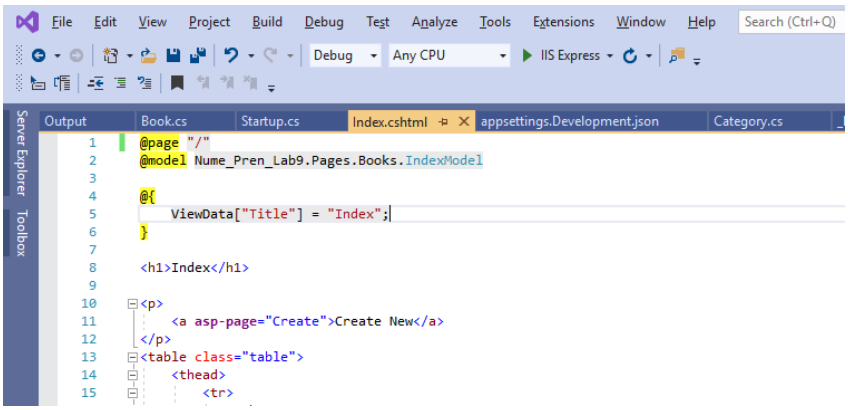
modificam fisierul _Layout.cshtml conform imaginii de mai jos:



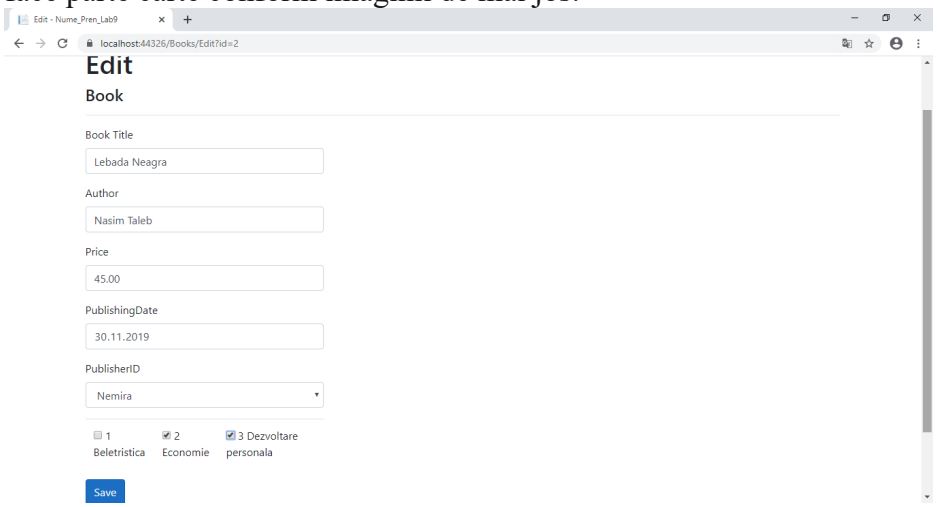
10. Rulam aplicatia pentru a testa noile legaturi. Observam totusi ca pentru a ajunge la pagina Book/Index trebuie sa adaugam in bara de adrese “/Books”.



11. Pentru ca aplicatia sa porneasca direct cu pagina Books/Index si nu cu pagina /Index din folderul radacina, in ferestra Solution Explorer facem click dreapta pe fisierul Index.cshtml (**din folderul radacina!!**) si alegem Exclude from project. Apoi deschidem fisierul Books/Index.cshtml si adaugam la directiva @page “/” conform imaginii de mai jos



12. Dorim ca editarea respectiv adaugarea unei noi carti sa poata fi selectate categoriile din care face parte carte conform imaginii de mai jos:



13. Vom utiliza checkbox-uri pentru a marca categoriile din care face parte cartea. Pentru fiecare categorie din baza de date vom afisa cate un checkbox. Vom crea o clasa pentru categoriile asignate unei carti. Facem click dreapta pe folderul Models->Add->Class si ii dam numele AssignedCategoryData. Aceasta va avea urmatorul continut:

```
public class AssignedCategoryData
    {

            public int CategoryID { get; set; }
            public string Name { get; set; }
            public bool Assigned { get; set; }

    }
```

14. Pentru paginile Create si Edit al entitatii Book vom crea o clasa care mosteneste PageModel. Facem click dreapta pe folderul Models->Add->Class si ii dam numele BookCategoriesPageModel. Metoda PopulateAssignedCategoryData citeste entitatile Category si populeaza lista AssignedCategoryDataList.

```
using Microsoft.AspNetCore.Mvc.RazorPages;
using Nume_Pren_Lab2.Data;
```

```
namespace Nume_Pren_Lab2.Models
{
public class BookCategoriesPageModel:PageModel
    {

            public List<AssignedCategoryData> AssignedCategoryDataList;

            public void PopulateAssignedCategoryData(Nume_Pren_Lab2Context context,
                                                     Book book)
            {
                var allCategories = context.Category;
                var bookCategories = new HashSet<int>(
                    book.BookCategories.Select(c => c.CategoryID)); //
            AssignedCategoryDataList = new List<AssignedCategoryData>();
                foreach (var cat in allCategories)
                {
                AssignedCategoryDataList.Add(new AssignedCategoryData
                {
                        CategoryID = cat.ID,
                        Name = cat.CategoryName,
                        Assigned = bookCategories.Contains(cat.ID)
                    });
                }
            }

            public void UpdateBookCategories(Nume_Pren_Lab2Context context,
                string[] selectedCategories, Book bookToUpdate)
            {
                if (selectedCategories == null)
                {
                    bookToUpdate.BookCategories= new List<BookCategory>();
                    return;
                }

                var selectedCategoriesHS = new HashSet<string>(selectedCategories);
                var bookCategories = new HashSet<int>
                    (bookToUpdate.BookCategories.Select(c => c.Category.ID));
                foreach (var cat in context.Category)
                {
                    if (selectedCategoriesHS.Contains(cat.ID.ToString()))
                    {
                        if (!bookCategories.Contains(cat.ID))
                        {
                            bookToUpdate.BookCategories.Add(
                                new BookCategory
                                {
                                    BookID = bookToUpdate.ID,
                                    CategoryID = cat.ID
                                });
                        }
                    }
                    else
                    {
                        if (bookCategories.Contains(cat.ID))
                        {
                            BookCategory courseToRemove
                                = bookToUpdate
                                    .BookCategories
```

```
                                    .SingleOrDefault(i => i.CategoryID == cat.ID);
                            context.Remove(courseToRemove);
                        }
                    }
                }
            }
    }
```

15. Vom actualiza fisierul *Pages/Books/Edit.cshtml.cs* astfel:

```
using Nume_Pren_Lab2.Models;

namespace Nume_Pren_Lab2.Pages.Books
{
    public class EditModel : BookCategoriesPageModel
    {
        private readonly Nume_Pren_Lab2.Data.Nume_Pren_Lab2Context _context;

        public EditModel(Nume_Pren_Lab2.Data.Nume_Pren_Lab2Context context)
        {
            _context = context;
        }

        [BindProperty]
        public Book Book { get; set; }

        public async Task<IActionResult> OnGetAsync(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

//se va include Author  conform cu sarcina de la lab 2


            Book = await _context.Book
                .Include(b => b.Publisher)
                .Include(b => b.BookCategories).ThenInclude(b => b.Category)
                .AsNoTracking()
                .FirstOrDefaultAsync(m => m.ID == id);

            if (Book == null)
            {
                return NotFound();
            }
//apelam PopulateAssignedCategoryData  pentru o obtine informatiile necesare checkbox-
//urilor folosind clasa AssignedCategoryData

PopulateAssignedCategoryData(_context, Book);

           var authorList = _context.Author.Select(x => new
             {
                 x.ID,
                 FullName = x.LastName + " " + x.FirstName
             });
             ViewData["AuthorID"] = new SelectList(authorList, "ID", "FullName");
```

```
            ViewData["PublisherID"] = new SelectList(_context.Publisher, "ID",
"PublisherName");
            return Page();
        }


        public async Task<IActionResult> OnPostAsync(int? id, string[]
selectedCategories)
        {
            if (id == null)
            {
                return NotFound();
            }

//se va include Author  conform cu sarcina de la lab 2

            var bookToUpdate = await _context.Book
                .Include(i => i.Publisher)
                .Include(i => i.BookCategories)
                    .ThenInclude(i => i.Category)
                .FirstOrDefaultAsync(s => s.ID == id);
            if (bookToUpdate == null)
            {
                return NotFound();
            }

//se va modifica AuthorID  conform cu sarcina de la lab 2

            if (await TryUpdateModelAsync<Book>(
                bookToUpdate,
                "Book",
                i => i.Title, i => i.Author,
                 i => i.Price, i => i.PublishingDate, i => i.PublisherID))
            {
                UpdateBookCategories(_context, selectedCategories, bookToUpdate);
                await _context.SaveChangesAsync();
                return RedirectToPage("./Index");
            }
//Apelam UpdateBookCategories pentru a aplica informatiile din checkboxuri la entitatea Books care
//este editata
            UpdateBookCategories(_context, selectedCategories, bookToUpdate);
            PopulateAssignedCategoryData(_context, bookToUpdate);
            return Page();
        }
    }
}
```

16. Actualizam fisierul Pages/Books/Edit.cshtml astfel:

```
<div class="form-group">
                <label asp-for="Book.PublisherID" class="control-label"></label>
                <select asp-for="Book.PublisherID" class="form-control" asp-
items="ViewBag.PublisherID"></select>
                <span asp-validation-for="Book.PublisherID" class="text-danger"></span>
            </div>
            <div class="form-group">
                <div class="table">
                    <table>
```

```
                        <tr>
                            @{
                                int cnt = 0;

                                foreach (var cat in Model.AssignedCategoryDataList)
                                {
                                    if (cnt++ % 3 == 0)
                                    {
                                    @:</tr><tr>
                                    }
                                    @:<td>
                                        <input type="checkbox"
                                               name="selectedCategories"
                                               value="@cat.CategoryID"
                                               @(Html.Raw(cat.Assigned ?
"checked=\"checked\"" : "")) />
                                        @cat.CategoryID @:  @cat.Name
                                    @:</td>
                                }
                            @:</tr>
                            }
                        </table>
                    </div>
                </div>

                <div class="form-group">
                    <input type="submit" value="Save" class="btn btn-primary" />
                </div>
```

17. Actualizam Pages/Books/Create.cshtml.cs astfel:

```
using Nume_Pren_Lab2.Models;

namespace Nume_Pren_Lab2.Pages.Books
{
    public class CreateModel : BookCategoriesPageModel
    {
        private readonly Nume_Pren_Lab2.Data.Nume_Pren_Lab2Context _context;

        public CreateModel(Nume_Pren_Lab2.Data.Nume_Pren_Lab2Context context)
        {
            _context = context;
        }

        public IActionResult OnGet()
        {
       /* var authorList = _context.Author.Select(x => new
            {
                x.ID,
                FullName = x.LastName + " " + x.FirstName
            });
*/
// daca am adaugat o proprietate FullName in clasa Author
        ViewData["AuthorID"] = new SelectList(authorList, "ID", "FullName");
       ViewData["PublisherID"] = new SelectList(_context.Publisher, "ID",
"PublisherName");

            var book = new Book();
```

```
            book.BookCategories = new List<BookCategory>();

            PopulateAssignedCategoryData(_context, book);

            return Page();
        }

        [BindProperty]
        public Book Book { get; set; }

        public async Task<IActionResult> OnPostAsync(string[] selectedCategories)
        {
            var newBook = new Book();
            if (selectedCategories != null)
            {
                newBook.BookCategories = new List<BookCategory>();
                foreach (var cat in selectedCategories)
                {
                    var catToAdd = new BookCategory
                    {
                        CategoryID = int.Parse(cat)
                    };
                    newBook.BookCategories.Add(catToAdd);
                }
            }

            Book.BookCategories = newBook.BookCategories;
             _context.Book.Add(Book);
                await _context.SaveChangesAsync();
                return RedirectToPage("./Index");
            }

        }

    }
```

18. Actualizam Pages/Books/Create.cshtml astfel:

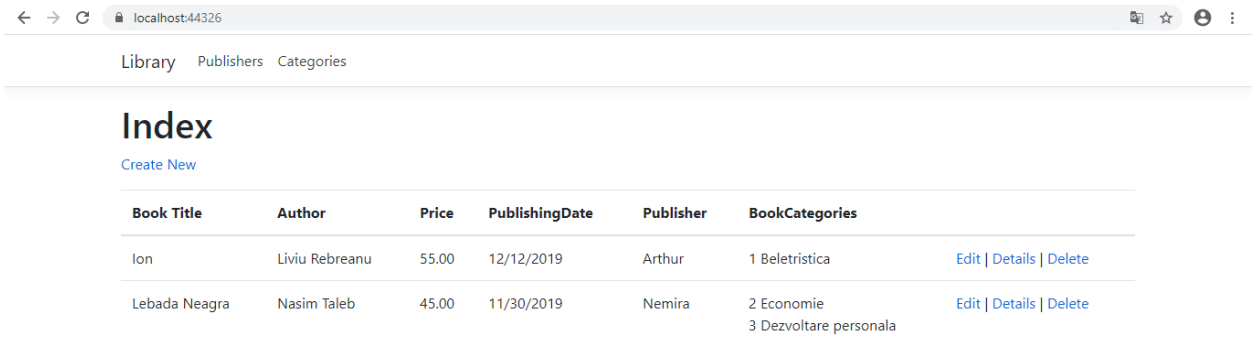
```
<div class="form-group">
                <label asp-for="Book.PublisherID" class="control-label"></label>
                <select asp-for="Book.PublisherID" class="form-control" asp-
items="ViewBag.PublisherID"></select>
            </div>
            <div class="form-group">
                <div class="table">
                    <table>
                        <tr>
                            @{
                                int cnt = 0;

                                foreach (var cat in Model.AssignedCategoryDataList)
                                {
                                    if (cnt++ % 3 == 0)
                                    {
                                    @:</tr><tr>
                                    }
                                    @:<td>
```

```
                                        <input type="checkbox"
                                               name="selectedCategories"
                                               value="@cat.CategoryID"
                                               @(Html.Raw(cat.Assigned ?
"checked=\"checked\"" : "")) />
                                        @cat.CategoryID @:  @cat.Name
                                    @:</td>
                                }
                            @:</tr>
                            }
                        </table>
                    </div>
                </div>

                <div class="form-group">
                    <input type="submit" value="Create" class="btn btn-primary" />
                </div>
```

19. Dorim ca pe pagina Pages/Books/Index sa se afiseze si categoriile din care face parte cartea conform imaginii de mai jos



20. In folderul Models creem o noua clasa care se va numi BookData. Aceasta va avea urmatorul continut:

```
public class BookData
    {

            public IEnumerable<Book> Books { get; set; }
            public IEnumerable<Category> Categories { get; set; }
            public IEnumerable<BookCategory> BookCategories { get; set; }


    }
```

21. Actualizam fisierul Pages/Books/Index.cshtml.cs astfel:

```
public IndexModel(Nume_Pren_Lab2.Data.Nume_Pren_Lab2Context context)
        {
            _context = context;
        }

        public IList<Book> Book { get;set; }
        public BookData BookD { get; set; }
        public int BookID { get; set; }
        public int CategoryID { get; set; }


        public async Task OnGetAsync(int? id, int? categoryID)
        {
            BookD = new BookData();

           BookD.Books = await _context.Book
                 .Include(b => b.Publisher)
                 .Include(b =>b.BookCategories)
                 .ThenInclude(b =>b.Category)
                 .AsNoTracking()
                 .OrderBy(b => b.Title)
                 .ToListAsync();

            if (id != null)
            {
                BookID = id.Value;
                Book book = BookD.Books
                    .Where(i => i.ID == id.Value).Single();
               BookD.Categories = book.BookCategories.Select(s => s.Category);
            }

        }

       }
```

22. Actualizam apoi fisierul Pages/Books/Index.cshtml astfel:

```
@page "/"
@model Nume_Pren_Lab2.Pages.Books.IndexModel

@{
    ViewData["Title"] = "Books";
}

<h1>Index</h1>

<p>
    <a asp-page="Create">Create New</a>
</p>
<table class="table">
    <thead>
        <tr>
           . . .

                @Html.DisplayNameFor(model => model.Book[0].Publisher)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.Book[0].BookCategories)
```

```
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model.BookD.Books) {

    string selectedRow = "";
            if (item.ID == Model.BookID)
            {
                selectedRow = "table-success";
            }

        <tr class="@selectedRow">
            <td>
                @Html.DisplayFor(modelItem => item.Title)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Author)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Price)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.PublishingDate)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Publisher.PublisherName)
            </td>
            <td>
                @{
                    foreach (var category in item.BookCategories)
                    {
                        @category.Category.ID @:  @category.Category.CategoryName <br/>
                    }
                }
            </td>

            <td>
                    <a asp-page="./Edit" asp-route-id="@item.ID">Edit</a>
                     <a asp-page="./Details" asp-route-id="@item.ID">Details</a> |
                 <a asp-page="./Delete" asp-route-id="@item.ID">Delete</a>
             </td>
        </tr>
}
    </tbody>
   </table>
```