Dezvoltarea Aplicatiilor Web utilizand ASP.NET Core MVC
Laborator 6

EXERCITII:

1. Sa se creeze un proiect, in cadrul caruia sa se adauge Entity Framework, baza de date si sa se includa sistemul de migratii (**VEZI Curs 5 – Sectiunea Crearea unui proiect utilizand EF si sistemul de migratii**).

2. Sa se testeze corectitudinea pasilor implementati in cadrul exercitiului 1 adaugand clasa Articles.cs, iar in Controller-ul ArticlesController sa se creeze metoda Index, metoda prin care se afiseaza toate articolele din baza de date. De asemenea, o sa fie nevoie si de un folder Articles in care o sa se creeze View-ul Index.cshtml.

    a. Se adauga Clasa → Article.cs

```
public class Article
    {
        [Key]
        public int ArticleID { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }
        public DateTime Date { get; set; }
    }
```

    b. Se ruleaza sistemul de migratii pentru update-ul bazei de date

    c. Se deschide tabelul si se insereaza 2 intrari in tabel

    d. Se implementeaza metoda Index in cadrul Controller-ului si View-ul asociat in folderul din View.

Metoda Index din ArticlesController

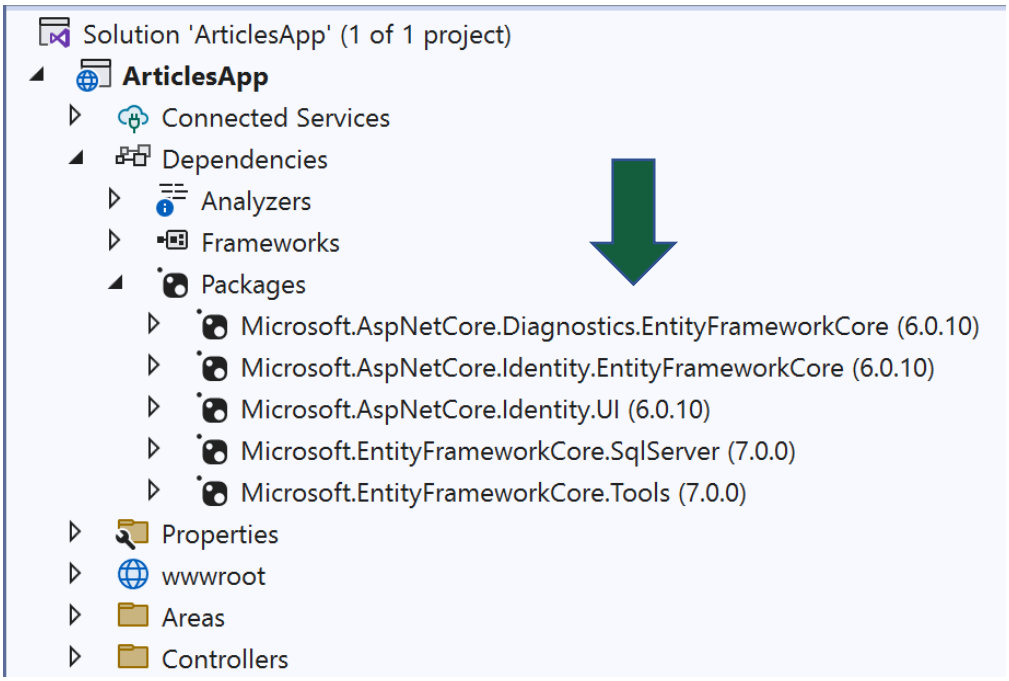
```
public IActionResult Index()
        {
            var articles = from article in db.Articles
                           select article;

            ViewBag.Articles = articles;

            return View();
        }
```

View-ul Index.cshtml

```
<h2>Afisare articole</h2>
<br />

@foreach (var art in ViewBag.Articles)
{
    <p>@art.Title</p>
    <p>@art.Content</p>
    <p>@art.Date</p>

    <br />
    <a href="/Articles/Show/@art.ArticleID">Afisare
articol</a>
    <br />
    <a href="/Articles/Edit/@art.ArticleID">Editare
articol</a>
    <hr />

}
```

3. Sa se creeze un nou proiect numit **ArticlesApp** de tipul ASP.NET Core Web App (Model-View-Controller). Proiectul o sa contina sistem de autentificare (**VEZI Curs 6 – Sectiunea Adaugarea Sistemului de Autentificare**).

4. Se verifica existenta urmatoarelor pachete:



5. Se ruleaza migratia Update-Database pentru realizarea update-ului in baza de date. Dupa acest pas se pot vedea tabelele in SQL Server Object Explorer. Se ruleaza doar comanda Update-Database deoarece migratia initiala exista.

6. Se ruleaza proiectul si se inregistreaza un cont, dupa care se poate vizualiza noul user in baza de date, tabelul **dbo.AspNetUsers**.

7. Se considera entitatile **Article**, **Category** si **Comments** cu urmatoarele proprietati:

Article

- Id (int – primary key)
- Title (string – titlul este obligatoriu)
- Content (string – continutul este obligatoriu)
- Date (DateTime)
- CategoryId (int – cheie externa – categoria din care face parte articolul)

Category:

- Id (int – primary key)
- CategoryName (string – numele este obligatoriu)

Comment:

- Id (int – primary key)
- Content (string – continutul comentariului este obligatoriu)
- Date (DateTime – data la care a fost postat comentariul)
- Date (DateTime)
- ArticleId (int – cheie externa – articolul caruia ii apartine comentariul)

Sa se implementeze cele trei clase in Models, dupa care sa se ruleze migratiile → in acest caz Add-Migration NumeMigratie si Update-Database.

Implementarea claselor:

```
public class Article
{
   [Key]
   public int Id { get; set; }

   [Required(ErrorMessage = "Titlul este obligatoriu")]
   public string Title { get; set; }

   [Required(ErrorMessage = "Continutul articolului este
obligatoriu")]
   public string Content { get; set; }

   public DateTime Date { get; set; }

   [Required(ErrorMessage = "Categoria este obligatorie")]
   public int CategoryId { get; set; }

   public virtual Category Category { get; set; }

   public virtual ICollection<Comment> Comments { get; set; }
}


public class Category
{
   [Key]
   public int Id { get; set; }

   [Required(ErrorMessage = "Numele categoriei este
obligatoriu")]
   public string CategoryName { get; set; }

   public virtual ICollection<Article> Articles { get; set; }
}


public class Comment
{
   [Key]
   public int CommentId { get; set; }

   [Required(ErrorMessage = "Continutul este obligatoriu")]
   public string Content { get; set; }

   public DateTime Date { get; set; }

   public int ArticleId { get; set; }

   public virtual Article Article { get; set; }
}
```

8. Se adauga proprietatile in contextul bazei de date pentru realizarea ulterioara a migratiilor.

```
public DbSet<Article> Articles { get; set; }
public DbSet<Category> Categories { get; set; }
public DbSet<Comment> Comments { get; set; }
```

9. Sa se ruleze sistemul de migratii, dupa care sa se insereze manual in fiecare tabel 2-3 intrari (in acest caz se ruleaza ambele comenzi).

10. Sa se adauge cate un Controller pentru fiecare clasa → **ArticlesController**, **CategoriesController** si **CommentsController** in care se vor implementa operatiile CRUD asupra entitatilor.

11. Sa se adauge cate un folder in Views pentru fiecare Controller.

12. Implementati operatii CRUD asupra entitatilor, urmand pasii urmatori.

- Sa existe posibilitatea realizarii operatiilor **C.R.U.D. pentru entitatea Article** astfel:
    - **Index** – afisarea tuturor articolelor, impreuna cu denumirea categoriei din care fac parte
    - **Show** – afisarea unui singur articol (intr-o pagina separata) impreuna cu denumirea categoriei din care face parte articolul respectiv
    - **New** – posibilitatea adaugarii unui nou articol. In momentul in care se adauga articolul, categoria se va selecta dintr-o lista existenta de categorii, folosind un element de tipul dropdown
    - **Edit** – posibilitatea editarii unui articol. In momentul in care se editeaza articolul, categoria se va selecta dintr-o lista existenta de categorii (element de tipul dropdown)
    - **Delete** – posibilitatea stergerii unui articol

- Sa existe posibilitatea realizarii operatiilor **C.R.U.D. pentru entitatea Category** (Atentie! In momentul in care se doreste stergerea unei categorii, automat se vor sterge si toate articolele care fac parte din categoria respectiva).

- Sa existe posibilitatea realizarii operatiilor **C.R.U.D. pentru entitatea Comment** astfel:

    - **Afisarea tuturor comentariilor** corespunzatoare unui articol. Fiecare articol o sa aiba un buton “Afisare articol” (Show) care redirectioneaza catre o pagina in care vom avea articolul impreuna cu toate comentariile corespunzatoare articolului respectiv

    Pentru afisarea articolelor se poate utiliza din Bootstrap > Panels: https://getbootstrap.com/docs/3.3/components/#panels

    - **New** – posibilitatea adaugarii unui nou comentariu
    - **Edit** – posibilitatea editarii unui comentariu existent
    - **Delete** – posibilitatea stergerii unui comentariu

Sugestii de implementare:

```
public ActionResult Index()
  {
      var articles = db.Articles.Include("Category");
      ViewBag.Articles = articles;

      return View();
  }



  public ActionResult Show(int id)
  {
      Article article = db.Articles.Include("Category")
                                   .Where(art => art.Id == id)
                                   .First();
      ViewBag.Article = article;
      ViewBag.Category = article.Category;

      return View();

  }



  public ActionResult New()
  {
      var categories = from cat in db.Categories
                       select cat;
      ViewBag.Categories = categories;
      return View();
  }


  [HttpPost]
  public ActionResult New(Article article)
  {
      try
      {
          db.Articles.Add(article);
          db.SaveChanges();
          return RedirectToAction("Index");
      }
      catch (Exception)
      {
          return RedirectToAction("New");
      }
  }
```

```
        public ActionResult Edit(int id)
        {

            Article article = db.Articles.Include("Category")
                                         .Where(art => art.Id == id)
                                         .First();
            ViewBag.Article = article;
            ViewBag.Category = article.Category;
            var categories = from cat in db.Categories
                             select cat;
            ViewBag.Categories = categories;
            return View();
        }


        [HttpPost]
        public ActionResult Edit(int id, Article requestArticle)
        {

            Article article = db.Articles.Find(id);

            try
            {
                {
                    article.Title = requestArticle.Title;
                    article.Content = requestArticle.Content;
                    article.Date = requestArticle.Date;
                    article.CategoryId = requestArticle.CategoryId;
                    db.SaveChanges();
                }
                return RedirectToAction("Index");
            }
            catch (Exception e)
            {
                return RedirectToAction("Edit", id);
            }
        }


        [HttpPost]
        public ActionResult Delete(int id)
        {
            Article article = db.Articles.Find(id);
            db.Articles.Remove(article);
            db.SaveChanges();
            return RedirectToAction("Index");
        }
    }
```

View-urile din View-ul Article:

Index.cshtml

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

<h2>Afisare articole</h2>

@foreach (var article in ViewBag.Articles)
{
    <h2>@article.Title</h2>
    <p>@article.Content</p>
    <p>@article.Date</p>
    <p>@article.Category.CategoryName</p>
    <a href="/Articles/Show/@article.Id">Afisare articol</a>
    <br />
    <hr />
    <br />
}
<a href="/Categories/Index/">Afisare lista categorii</a>
<br />
<a href="/Articles/New">Adauga articol</a>
<br />
```

Show.cshtml

```
{
    ViewBag.Title = "Show";
}
<br />
<a href="/Articles/Index">Inapoi la articole</a>
<br />
<a class="btn btn-info" href="/Articles/New">Adauga articol</a>
<br />
<div class="panel panel-default">
    <div class="panel-heading">@ViewBag.Article.Title</div>
    <div class="panel-body">
        Continut articol:
<strong>@ViewBag.Article.Content</strong>
        <br />
        <span class="label label-
success">@ViewBag.Article.Date</span>
        <br />
```

```
        <i class="glyphicon glyphicon-globe"></i>
@ViewBag.Category.CategoryName
        <br />
        <hr>
        <a class="btn btn-success pull-left"
href="/Articles/Edit/@ViewBag.Article.Id">Modifica articol</a>
        <form method="post"
action="/Articles/Delete/@ViewBag.Article.Id">

            <button class="btn btn-danger pull-right"
type="submit">Sterge articol</button>
        </form>
    </div>
</div>
```

New.cshtml

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

<h2>Adaugare articol</h2>
<form method="post" action="/Articles/New">
    <label>Titlu articol</label>
    <input type="text" name="Title" />
    <br />
    <label>Continut articol</label>
    <textarea name="Content"></textarea>
    <br />
    <input type="hidden" name="Date" value="@DateTime.Now" />
    <br />

    <label>Selectati categoria</label>

    @* name o sa fie Id -> id-ul categoriei si ce se afla in value se paseaza
catre server
       prin cheia Id (prin id-ul categoriei)

      name este atributul pe care il trimitem catre controller prin post
      iar denumirea categoriei este valoarea afisata in dropdown *@

    <select name="CategoryId">
        @foreach (var category in ViewBag.Categories)
        {
            <option value="@category.Id">@category.CategoryName</option>
        }
    </select>
    <br />
    <button type="submit">Adauga articol</button>
</form>
```

Edit.cshtml

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


<form method="post" action="/Articles/Edit/@ViewBag.Article.Id">

    <label>Titlu articol</label>
    <br />
    <input type="text" name="Title" value="@ViewBag.Article.Title" />
    <br /><br />
    <label>Continut articol</label>
    <br />
    <input type="text" name="Content" value="@ViewBag.Article.Content" />
    <br /><br />
    <label>Data</label>
    <br />
    <input type="text" name="Date" value="@ViewBag.Article.Date" />
    <br />

    <select name="CategoryId">
        @foreach (var category in ViewBag.Categories){
           if (category.Id == ViewBag.Category.Id)
           {
                <option selected="selected" value="@category.Id">
                    @category.CategoryName
                </option>
            }
            else
            {
                <option value="@category.Id">
                    @category.CategoryName
                </option>
            }
        }

    </select>
    <br />
    <button type="submit">Modifica articol</button>
</form>
```