1. Debut rapid V
2. Introducere în Controller, View şi rutare V
3. Accesul la baza de date, Entity Framework, Code First V
4. Forme şi CRUD
5. Adnotări şi validări regex
6. Autentificare
7. ASP.NET Web API
8. AJAX, jQuery şi DataTables
9. Caching, Filtre V

INTRODUCERE IN CONTROLLER

ViewResult => View

* Cale relativa: View("Prima")
* Cale absoluta: View("~/Views/Home/Contact.cshtml")

ContentResult => Content cu parametru string: creează o pagină având acel string pe post de cod HTML

* Content("<b> **Good night </b>**")

RedirectResult => Redirect: redirecţionare la URL-ul indicat de string

* Redirect("http://fmi.unibuc.ro")

RedirectToRouteResult => RedirectToAction

* RedirectToAction("Index", "Figuri", new { coded = 48, start = "yes" }) in /Figuri?coded=48&start=yes

Daca adaugam parametri unei actiuni, ei vor fi trimisi prin url cu acelasi nume!

Rutari personalizate:

Fara rutare personalizata:

public ActionResult ViewContent(int page)

Putem accesa: Home/ViewContent?page=4

Mai multi param: /Home/ViewContent?page=3&nr=2

Nu putem accesa: Home/ViewContent/4, deoarece numele id nu corespunde cu page si nu stie sa ii atribuie valoarea lui page!

Pentru a accesa /Home/ViewContent/3/2 FACEM O RUTA PERSONALIZATA inainte de ruta default:

routes.MapRoute(

name: "Route1",

url: "RouteCustom/{coded}/{start}", <- param intre { }

defaults: new { controller = "Figuri", action = "Edit" },

new {coded = @”2015|2016”, start=@”\d{2}”}); - pentru valori default

Rutari atributate:

Rutarile conventionale sunt greu de intretinut.

1. In RegisterRoutes adaugam routes.MapMvcAttributeRoutes();
2. Deasupra actiunii adaugam [Route("RouteCustom/{coded}/{start}")]
3. Daca vrem parametri optionali, adaugam ? dupa numele lor
4. [Route("AttributeRoute/{age}/{val?}")]

Nu uita de ? in antetul actiunii!

1. [Route("AttributeRoute/{age:regex(^\\d{1,3}$):range(0, 120)}/{val?}")]

VIEW-uri PARTIALE:

View => Create as partial view (numele incepe cu \_)

@Html.Partial("\_PartialNumber", nr); <- cu modelul nr transmis

ADNOTARI SI VALIDARI:

[Display(Name="Titlu")] => modificarea numelui afisat de html helpers pe proprietati

Accesul la baza de date

Code First = nu proiectam baza de date, e direct generata de Entity Framework in functie de codul programului

Entity Framework:

Tools => Nuget Package Manager => Manage NuGet Packages for Solution => search for EntityFramework(in Browse) => bifati proiectul curent => apasati install

Crearea unei noi baze de date:

App\_Data => Add => New Item => SQL Server Database => se face nou fisier cu extensia MDF

(ex. BookSchema)

Accesarea bazei de date:

Web.config => inainte de </configuration> se adauga:

<connectionStrings>

<add name="DbConnectionString" providerName="System.Data.SqlClient" connectionString="Data

Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename='C:\...\App\_Data\BookSchema.mdf';Integrated

Security=True"/>

</connectionStrings>

Puteti prelua connectionString-ul vostru din Server Explorer -> click dreapta pe baza de date ->

Properties -> copiati ce scrie la ConnectionString

public class DbCtx : DbContext

    {

        public DbCtx() : base("BooksCS")

        {

            System.Data.Entity.Database.SetInitializer<DbCtx>(new Initp());

        }

        public DbSet<Book> Books { get; set; }

    }

    public class Initp : DropCreateDatabaseAlways<DbCtx>

    {

        protected override void Seed(DbCtx context)

        {

            context.Books.Add(new Book { Title = "Pride and Prejudice", Author = "Jane Austen" });

            context.SaveChanges();

            base.Seed(context);

        }

    }

Annotations:

* Schimbare nume tabel: deasupra clasei adaugam

[Table("NumeTabel")]

* Schimbare coloana tabel

[Column("Titlu")]

* Pentru cheie primara care nu se supune conventiei

[Key]

* Atribut nemapat
* [NotMapped]

Relatii:

One-to-Many:

public class Student

**public int GradeId { get; set; }**

**public virtual Grade Grade { get; set; }**

public class Grade

public int GradeId { get; set; }

**public virtual ICollection<Student> Student { get; set; }**

Pentru a avea GradeId din student null, ori nu il adaugam, ori facem public int? GradeId;

Daca vrem ca la one-to-many sa avem cheie straina nenula, iar campul vrem sa aiba alt nume decat cel al cheii primare referite:

**[ForeignKey("Standard")]**

**public int StandardRefId { get; set; }**

**public Standard Standard { get; set; }**

SAU

public int StandardRefId { get; set; }

[ForeignKey("StandardRefId")]

public Standard Standard { get; set; }

SAU

public int StandardRefId { get; set; }

public Standard Standard { get; set; }

[ForeignKey("StandardRefId")]

public ICollection<Student> Students { get; set; }

<https://www.entityframeworktutorial.net/code-first/foreignkey-dataannotations-attribute-in-code-first.aspx>

One-to-one:

public class Student

[Required]

public virtual StudentAddress Address { get; set; }

public class StudentAddress

public virtual Student Student { get; set; }

Many-to-Many:

public class Student

public virtual ICollection<Course> Courses { get; set; }

public class Course

public virtual ICollection<Student> Students { get; set; }

CRUD:

@Html.ActionLink("New book", "New")

@Html.ActionLink("More About " + p.Title, "Details", new { id = p.BookId })

Adnotari si validari:

1. [Required(ErrorMessage = "Not good")] - ATENTIE: MESAJE PERSONALIZATE

[StringLength(4)]

[Range(5, 25, ErrorMessage = "Not good")]

[RegularExpression(...)]

1. In METODA POST CARE REALIZEAZA MODIFICAREA IN BAZA DE DATE VERIFICAM DACA DATELE TRANSMISE SUNT VALIDE:

if (!ModelState.IsValid)

return View("Edit", b);

1. INTRODUCERE DE HTML HELPERS IN VIEW

@Html.ValidationMessageFor(m => m.Title)

* return ValidationResult.Success;
* return new ValidationResult("mesaj")

Validari personalizate:

Pe clasa: [ValidB]

In Models -> New class: ValidB => MOSTENESTE ValidationAttribute

Facem override:

protected override ValidationResult IsValid(object value, ValidationContext validationContext)

WEB API

ATENTIE!!! PENTRU A INTEGRA WEB API INTR-UN PROIECT CU MVC INTRODUCEM URMATOAREA LINIE IN GLOBAL.ASAX:

AreaRegistration.RegisterAllAreas();

GlobalConfiguration.Configure(WebApiConfig.Register);//WEB API

FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);

RouteConfig.RegisterRoutes(RouteTable.Routes);

BundleConfig.RegisterBundles(BundleTable.Bundles);

AJAX

Pentru read:

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).

In bundleConfig adaugam:

bundles.Add(new ScriptBundle("~/bundles/jquery").Include(

            "~/Scripts/jquery-{version}.js",

**"~/scripts/datatables/jquery.datatables.js",**

**"~/scripts/datatables/datatables.bootstrap.js"**

            ));

bundles.Add(new StyleBundle("~/Content/css").Include(

                      "~/Content/bootstrap.css",

                      "~/Content/site.css",

**"~/content/datatables/css/datatables.bootstrap.css"**));

Facem in Controllers un folder separat Api => New Controller => BookController de tip WebApi

OUTPUT CACHING:

De exemplu, pentru a stoca rezultatul unei acţiuni în cache, scriem înaintea antetului ei atributul OutputCache:

[OutputCache(Duration = 5)] 5 = durata in secunde pentru care rezultatul e stocat

Exercitiu:

* Intr-un controller, adaugam pe o actiune atributul OutputCache
* In View-ul asociat, introducem un camp data pentru a vedea modificarile
* Daca nu dam refresh, nu vom vedea nicio schimbare
* Daca dam refresh, in mai putin de valoarea setata, datetime.now nu se modifica

Pentru a stoca date în cache, adăugăm referinţa: System.Runtime.Caching.dll (References => Add Reference)

MemoryCaching.Default[“....”] ATENTIE INITIAL E NULL

Exercitiu:

* Sa se stocheze in cache o valoare care sa se incrementeze la fiecare refresh
* public ActionResult MemCache()
* {
* if (MemoryCache.Default["val"] == null)
* MemoryCache.Default["val"] = 0;
* int current = (int)MemoryCache.Default["val"];
* current++;
* MemoryCache.Default["val"] = current;
* return View(current);
* }

ATENTIE: este de tip object, TREBUIE CAST (nu se poate face ++ pe el)

DEBUG

Debug.WriteLine("mesaj"); -> In Diagnostics

Filtre personalizate:

Creare:

* Facem o clasa MyFilter in Models
* Mosteneste ActionFilterAttribute din namespace-ul using System.Web.Mvc;
* Suprascrie metodele:
* OnActionExecuting(ActionExecutingContext filterContext) - Inainte de rularea actiunii.
* OnActionExecuted(ActionExecutedContext filterContext) - Imediat dupa rularea actiunii careia ii este aplicat filtrul. Aici putem vedea si modelul transmis view-ului: ((ViewResultBase)filterContext.Result).Model
* OnResultExecuting(ResultExecutingContext filterContext) - Inainte de rulareaActionResult-ului corespunzator.
* OnResultExecuted(ResultExecutedContext filterContext) - Imediat dupa rularea ActionResult-ului corespunzator.
* public override void OnActionExecuting(ActionExecutingContext filterContext)
* {
* Debug.WriteLine(filterContext.HttpContext.Request.Url); //afisarea url-ului
* Debug.WriteLine(filterContext.RouteData.Values["controller"]); //afisarea controllerului
* Debug.WriteLine(filterContext.RouteData.Values["action"]); //afisarea actiunii
* Debug.WriteLine(filterContext.HttpContext.Request.UserHostAddress);
* Debug.WriteLine(filterContext.HttpContext.Request.UserAgent);
* }

Putem sa vedem ce model s-a transmis view-ului dupa ce actiunea s-a executat:

public override void OnActionExecuted(ActionExecutedContext filterContext)

        {

            //putem sa vedem ce model s-a transmis view-ului dupa ce actiunea s-a executat

            if (((ViewResultBase)filterContext.Result).Model != null) {

                Debug.WriteLine("Modelul este " + ((ViewResultBase)filterContext.Result).Model.ToString());

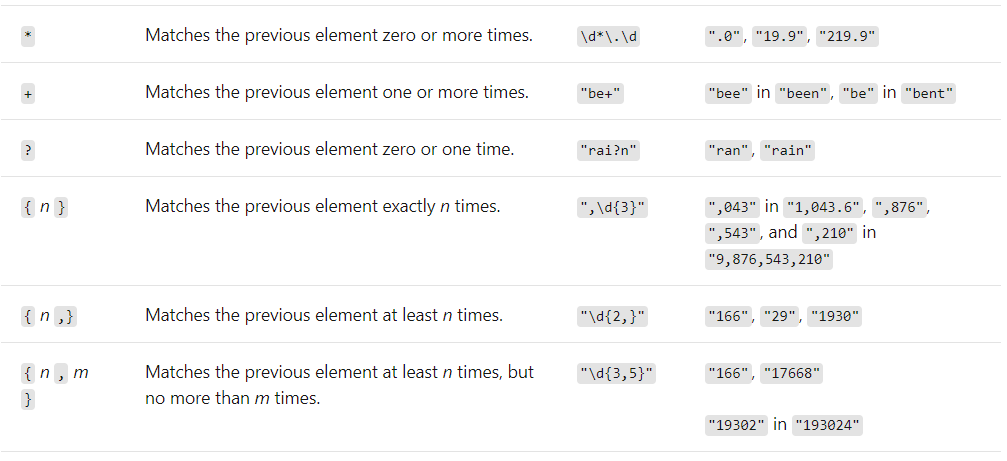
            }

        }

Pe actiune: [Models.MyFilter]

REGEX:

<https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference>



\. Pentru punct

//Email address [RegularExpression(@"^([\w\d\_]+@[A-Za-z]+\.[A-Za-z]{2,3}){50}$", ErrorMessage = "Title not valid")]

//Tip propozitie cu . [RegularExpression(@"^[A-Za-z\s]+\.$", ErrorMessage ="Title not valid")]

//cuvinte care incep cu litera mare[RegularExpression(@"^([A-Z][A-Za-z\s]\*)\*$", ErrorMessage = "Title not valid")]