**MVC**

Asp .net web application (.net framework) tipo MVC

Elimino HomeController e la cartella Home

References, manage nuget packages, scarico entityFramework

Dentro i models creo User con tutto public e ID e le altre entità che voglio

Faccio la classe libraryContext che collega DbContext (scarico il plugin)

Metto public DbSet<User> users {get set} e stessa cosa per books e un public LibraryContext(){}

Tasto destro sul progetto, properties, settings, creo un nuovo file di configurazione

Creo una nuova LibraryConnectionString di tipo connection string, premo i tre puntini , servename=localhost, sql server authentication, metto user e pwd e lo chiamo LibraryDatabase

Dentro LibraryContext(){} metto this.Database.ConnectionString = Properties.Settings.Default.LibraryConnectionString;

Aggiungo le annotation come ad esempio [MaxLenght(250)] e [Required] scaricando anche la libreria

Vado nella packet manage console: Enable-Migrations, Add-Migration FirstMigration, Update-Database, controllo su sql server che si sia creato il db

Aggiungo la classe error, e aggiungo in book public User User {get set} e in user public List<Book> Books {get set}, mando Add-Migration SecondMigration, Update-Database (ricordati di aggiornare il dbset nel dbcontext)

Creo UsersController metto

[HttpGet]

using(Models.LibraryContext ctx = new Models.LibraryContext())

{

var users = ctx.Users.OrderBy(u => u.Surname).ToList();

return View(users);

}

dentro ActionResult Index, dopo di che nella cartella delle views creo la view Index, Template = List, Model class= user e layout dentro la cartella shared.

Starto il programma e se voglio cambio Index in Users nella cartella AppStart RouteConfig

Faccio la create in usersController e creo la view Create con template = create e l’altra = users ecc

[HttpGet]

public ActionResult Create()

{

return View();

}

Faccio la seconda create di tipo post con un piccolo controllo per il boolean dell’active

[HttpPost]

public ActionResult Create(FormCollection form)

{

using (Models.LibraryContext ctx = new Models.LibraryContext())

{

bool active = false;

if (form["Active"].Contains("true"))

active = true;

Models.User user = new Models.User()

{

Name = form["Name"],

Surname = form["Surname"],

Email = form["Email"],

Active = active

};

ctx.Users.Add(user);

ctx.SaveChanges();

return RedirectToAction("Index");

}

}

Nella pagina web ho un tasto detail che non mi interessa e quindi vado a rimuoverlo dalla vista

Creo la view di edit con le solite informazioni, template edit e classe user

Faccio la get di edit:

public ActionResult Edit(int id)

{

using(Models.LibraryContext ctx = new Models.LibraryContext())

{

var currentUser = ctx.Users.Where(u => u.ID.Equals(id)).FirstOrDefault();

if (currentUser != null)

return View(currentUser);

}

return View();

}

Faccio la post di edit:

[HttpPost]

public ActionResult Edit(int id, FormCollection form)

{

using (Models.LibraryContext ctx = new Models.LibraryContext())

{

var currentUser = ctx.Users.Where(u => u.ID.Equals(id)).FirstOrDefault();

if (currentUser != null)

{

currentUser.Name = form["Name"];

currentUser.Surname = form["Surname"];

currentUser.Email = form["Email"];

}

ctx.SaveChanges();

}

return RedirectToAction("Index");

}

Inizio la delete di users

public ActionResult Delete(int id)

{

using (Models.LibraryContext ctx = new Models.LibraryContext())

{

var currentUser = ctx.Users.Where(u => u.ID.Equals(id)).FirstOrDefault();

if (currentUser != null)

{

ctx.Users.Remove(currentUser);

}

ctx.SaveChanges();

}

return RedirectToAction("Index");

}

Creo nel model authors e faccio il solito processo, db set, migrations, cambio la string con author

Aggiusto i link nel \_layout, aggiusto i titoli nelle views di authors e books

Faccio index e create di authors e books con relative views

Creo una partialview, in authors creo una nuova view AuthorsSelect con template list di author e TUTTE le caselle spuntate.

Levo tutto tranne la prima riga e metto

<select id="Author" name="Author" class="form-control">

@foreach (var item in Model)

{

<option value="@item.Id">@item.Surname, @item.Name</option>

}

</select>

Nella create di books metto dopo il titolo

<div class="form-group">

<label class="control-label col-md-2">Author</label>

<div class="col-md-10">

@Html.Partial("../Authors/AuthorsSelect", Model.Authors)

</div>

</div>

#

Creo una cartella ViewModels ci metto dentro CreateBookViewModel e metto

public Models.Book Book { get; set; }

public List<Models.Author> Authors { get; set; }

Nel BookController aggiorno la create

public ActionResult Create()

{

using(Models.LibraryContext ctx = new Models.LibraryContext())

{

ViewModels.CreateBookViewModel item = new ViewModels.CreateBookViewModel();

item.Book = new Models.Book();

item.Authors = ctx.Authors.OrderBy(a => a.Surname).ToList();

return View(item);

}

}

Torno nella create di book cambio la prima riga con

@model LibraryManagement.ViewModels.CreateBookViewModel

E poi in tutti i @html aggiungo dopo model. ” Book.”

Faccio la POST

[HttpPost]

public ActionResult Create(FormCollection form)

{

using(Models.LibraryContext ctx = new Models.LibraryContext())

{

bool available = false;

if (form["Book.Available"].Contains("true"))

available = true;

int authorId = int.Parse(form["Author"]);

var author = ctx.Authors.Where(a => a.Id.Equals(authorId)).FirstOrDefault();

Models.Book book = new Models.Book()

{

Title = form["Book.Title"],

ISBN = form["Book.ISBN"],

Available = available,

Author = author

};

ctx.Books.Add(book);

ctx.SaveChanges();

}

return RedirectToAction("Index");

}

Torno nel CreateBookViewModel e aggiungo

public List<Models.User> Users { get; set; }

Creo UsersSelect con list e tutte le caselle spuntate

Faccio lo stesso procedimento di AuthorSelect fino al #

Aggiungo nella prima create di booksController

item.Users = ctx.Users.OrderBy(u => u.Surname).ToList();

E nella seconda

int userId = int.Parse(form["User"]);

var user = ctx.Users.Where(u => u.ID.Equals(userId)).FirstOrDefault();

e

User = user nella lista appena sotto

Metto i controlli

if(!string.IsNullOrEmpty(name) && !string.IsNullOrEmpty(surname) && !string.IsNullOrEmpty(email))

{

}

else

{

}

}

Ci metto dentro lo using

Creo un errorsController e la relativa view Index con template= details di error

Ci metto

public ActionResult Index(Models.Error error)

{

return View(error);

}

Nell else metto

Models.Error error = new Models.Error()

{

Timestamp = DateTime.Now,

Message = "Informazioni mancanti, si prega di verificare!"

};

return RedirectToAction("Index", "Error", error);

creo un javascript file nella cartella script chiamato users.js

$(document).ready(function () {

$('input[type=submit]').click(function (e) {

var name = $('#Name').val();

var surname = $('#Surname').val();

var email = $('#Email').val();

if (name == "" || name == undefined

|| surname == "" || surname == undefined

|| email == "" || email == undefined

) {

alert("Informazioni mancanti, si prega di verificare!");

e.preventDefault();

}

});

});

Nella create di user in basso metto

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

<script type="text/javascript" src="~/Scripts/jquery-3.4.1.min.js"></script>

<script type="text/javascript" src="~/Scripts/users.js"></script>

}

Aggiorno la Index di errorscontroller

public ActionResult Index(int id)

{

using (Models.LibraryContext ctx = new Models.LibraryContext())

{

var error = ctx.Errors.Where(e => e.ID.Equals(id)).FirstOrDefault();

if (error != null)

return View(error);

else

return RedirectToAction("Index", "Users");

}

}

Riorganizzo lapost di usersController

[HttpPost]

public ActionResult Create(FormCollection form)

{

var name = form["Name"];

var surname = form["Surname"];

var email = form["Email"];

using (Models.LibraryContext ctx = new Models.LibraryContext())

{

if (!string.IsNullOrEmpty(name) && !string.IsNullOrEmpty(surname) && !string.IsNullOrEmpty(email))

{

bool active = false;

if (form["Active"].Contains("true"))

active = true;

Models.User user = new Models.User()

{

Name = form["Name"],

Surname = form["Surname"],

Email = form["Email"],

Active = active

};

ctx.Users.Add(user);

ctx.SaveChanges();

return RedirectToAction("Index");

}

else

{

Models.Error error = new Models.Error()

{

Timestamp = DateTime.Now,

Message = "Informazioni mancanti, si prega di verificare!"

};

ctx.Errors.Add(error);

ctx.SaveChanges();

return RedirectToAction("Index", "Errors", new { id = error.ID });

}

}

}