Tutorial for Class number 5

This exercise intends to complement de previous tutorial (Tutorial_Class4) adding a relational class model to the project. The application will continue to allow to perform CRUD operations without having to write SQL. The database system operations will be supported by **Entity Framework Core** and generated by the process **Code First**.

This tutorial also follows the "ASP.NET Core MVC with EF Core - tutorial series"

https://docs.microsoft.com/en-us/aspnet/core/data/ef-mvc/?view=aspnetcore-5.0

First step

- Open de solution of previous class (Class04)
- Add a new class named **Course** to **Models** folder in project with the next content.

```
public class Course
{
    0 references
    public int Id { get; set; }

    [Required(ErrorMessage = "Required Field")]
    [StringLength(50, MinimumLength = 3, ErrorMessage = "{0} must be betwen {2} and {1}")]
    0 references
    public string Name { get; set; }

    [Required(ErrorMessage = "Required Field")]
    [StringLength(256, ErrorMessage = "length can not exceed {1} characters")]


    0 references
    public string Description { get; set; }
    0 references
    public int Credits { get; set; }

    [DataType(DataType.Currency)]
    [Column(TypeName = "money")]
    0 references
    public Decimal Cost { get; set; }
    0 references
    public Boolean State { get; set; } = true;
    0 references
    public int CategoryID { get; set; }
    0 references
    public Category Category { get; set; }
}
```

This class intends to represent a **Course** entity in a Database system.

This class has a field that represents the **Category** of the course, which is one of existing items in the Categories table. This is the FOREIGN KEY concept of the relational databases. The foreign key and navigation properties in the Course entity reflects the following relationships:

- A **Category** is assigned to one **Course**, so there's a **CategoryID** foreign key and a **Category** navigation property.

The **Column** attribute is used to change SQL data type mapping so that the column will be defined using the SQL Server money type in the database.

- Now we need to change the Category class to represent that one Category must have multiple Courses. A **Category** may have **many** courses, so there's a **Courses navigation property**

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

    [Required(ErrorMessage ="Required Field")]
    [StringLength(50, MinimumLength = 3, ErrorMessage = "{0} must be betwen {2} and {1}")]
    16 references
    public string Name { get; set; }


    [Required(ErrorMessage = "Required Field")]
    [StringLength(256, ErrorMessage = "length can not exceed {1} characters")]
    15 references
    public string Description { get; set; }

    13 references
    public Boolean State { get; set; }

    [Display(Name = "Creation Date")]
    12 references
    public DateTime Date { get; set; } = DateTime.Now;

    0 references
    public ICollection<Course> Courses { get; set; }
}
```

- Now we need to add the class Course to the Data Context:

```
    9 references
    public DbSet<Aula04.Models.Category> Category { get; set; }
    0 references
    public DbSet<Aula04.Models.Course> Course { get;  set; }
}
```

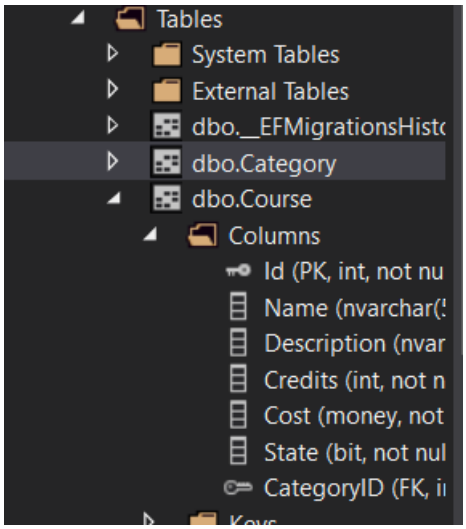
- To reflect this changes in the Model, we need to create a new Migration:

```
Package Manager Console
Package source: All        Default project: Aula04
PM> add-migration AddCourse
```

- Next we will alter the database by executing the command update-database.

```
Package Manager Console
Package source: All        Default project: Aula04
PM> add-migration AddCourse
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> update-database
```

Confirm in **SQL Server Object Explorer** the new table created in the database.

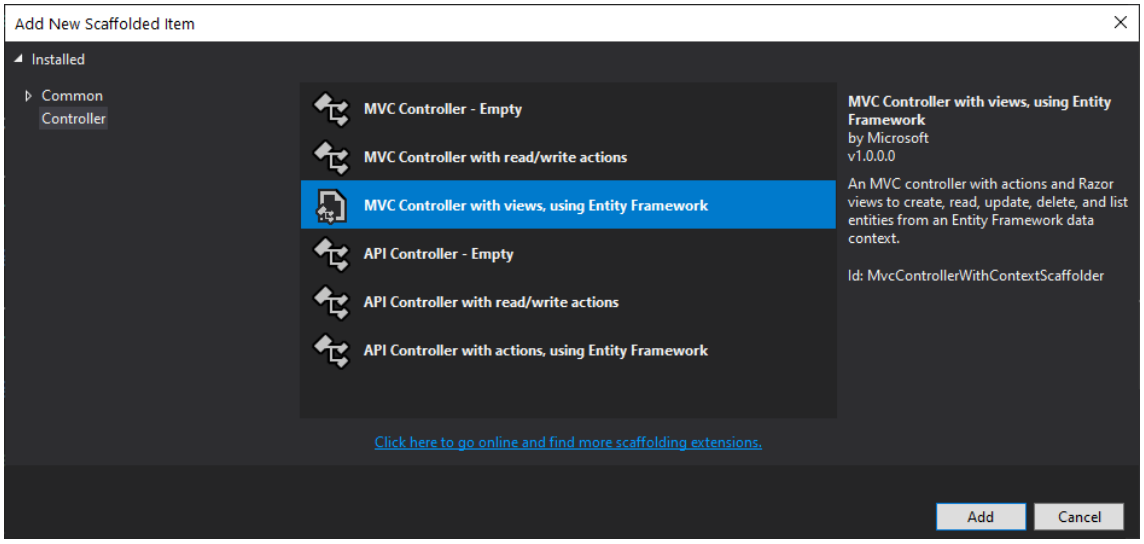


Using the mouse right button over the **dbo.Course** we ca use the View Code option to see the SQL code that generated the table. Here we can see the creation of the **foreign key** in the Category table, and the SQL data type **Money**, reflecting the data annotation **[Column(TypeName = "money")]**

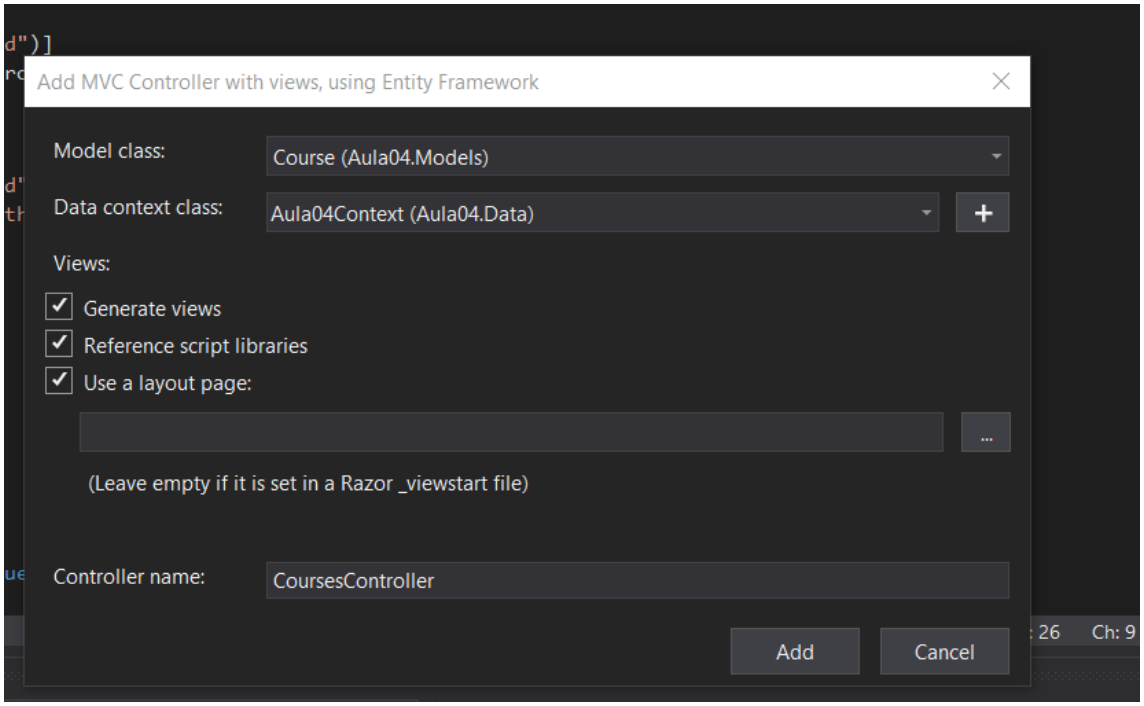
```
Update
CREATE TABLE [dbo].[Course] (
    [Id]          INT            IDENTITY (1, 1) NOT NULL,
    [Name]        NVARCHAR (50)  NOT NULL,
    [Description] NVARCHAR (256) NOT NULL,
    [Credits]     INT            NOT NULL,
    [Cost]        MONEY          NOT NULL,
    [State]       BIT            NOT NULL,
    [CategoryID]  INT            NOT NULL,
    CONSTRAINT [PK_Course] PRIMARY KEY CLUSTERED ([Id] ASC),
    CONSTRAINT [FK_Course_Category_CategoryID] FOREIGN KEY ([CategoryID]) REFERENCES [dbo].[Category] ([Id]) ON DELETE CASCADE
);


GO
CREATE NONCLUSTERED INDEX [IX_Course_CategoryID]
    ON [dbo].[Course]([CategoryID] ASC);
```

Second Step

- Add a controller with the template "MVC Controller with views, using Entity Framework". This will add to the project one controller class and all views code for CRUD operations over the model data.



- Choose the **Course** class as model class and the existing Data Context class **Aula04Context** (or what else name you gave to it) - in this case you should not create a new one.



- Accept or modify the suggested name for the controller (**CoursesController**).

Third step

For running the application for the first time it is important to have some data in the database.

- Open the **DbInitializer.cs** file in the **Data** folder, and append the following code, which create new courses in the database.

```
foreach (Category c in categories)
{
    context.Category.Add(c);
}
context.SaveChanges();

var courses = new Course[]
{
    new Course {
        Name="Web Engineering",
        Description="Creating new sites using ASP.NET",
        Cost=50, Credits=6,
        CategoryId = categories.Single(categories=>categories.Name=="Programming").Id
    },
    new Course
    {
        Name = "Strategic Leadership nd Managment",
        Description = "Leadership and Business Skill for Immediate Impact.",
        Cost = 100, Credits = 6,
        CategoryId = categories.Single(categories => categories.Name == "Administration").Id
    },
    new Course
    {
        Name = "Master in Corporate Communication",
        Description = "This Master in Corporate Communication will provide required to organize a Communication Department.",
        Cost = 80, Credits = 10,
        CategoryId = categories.Single(categories => categories.Name == "Communication").Id
    }
};

context.Course.AddRange(courses);
context.SaveChanges();
```

To create these courses in the start-up of the application, we need to delete all existing data in the database.

And we are ready to test the application 🚀🎆

- run on **https://localhost:????/courses/** to see all courses listed…

Index

Create New

Name	Description	Credits	Cost	State	Category	
Master in Corporate Communication	This Master in Corporate Communication will provide required to organize a Communication Department	10	80,00 €	☑	Business and institutional comunication courses	Edit | Details | Delete
Strategic Leadership and Management	Leadership and Business Skill for Immediate Impact.	6	100,00 €	☑	Public administration and business management courses	Edit | Details | Delete
Web engineering	Creating wew sites using ASP.NET	6	50,00 €	☑	Algorithms and programming area courses	Edit | Details | Delete

- … or editing one of the elements listed in **https://localhost:????/courses/Edit/1** (by clicking Edit link)

Edit

Course

Name

Master in Corporate Communication

Description

This Master in Corporate Communication will

Credits

10

Cost

80,00

☑ State

CategoryID

Business and institutional comunication co

Save

You should test all CRUD operations over courses elements. 😆 🙄

Homework: 😈

- Change the code where needed to show the category **Name** instead the **Description** in all the views (list, details, edit, create and delete).

Index

Create New

Name	Description	Credits	Cost	State	Category	
Master in Corporate Communication	This Master in Corporate Communication will provide required to organize a Communication Department	10	80,00 €	☑	Business and institutional comunication courses	Edit | Details | Delete
Strategic Leadership and Management	Leadership and Business Skill for Immediate Impact.	6	100,00 €	☑	Public administration and business management courses	Edit | Details | Delete
Web engineering	Creating wew sites using ASP.NET	6	50,00 €	☑	Algorithms and programming area courses	Edit | Details | Delete

Edit

Course

Name

Master in Corporate Communication

Description

This Master in Corporate Communication will

Credits

10

Cost

80,00

☑ State

CategoryID

Business and institutional comunication cou

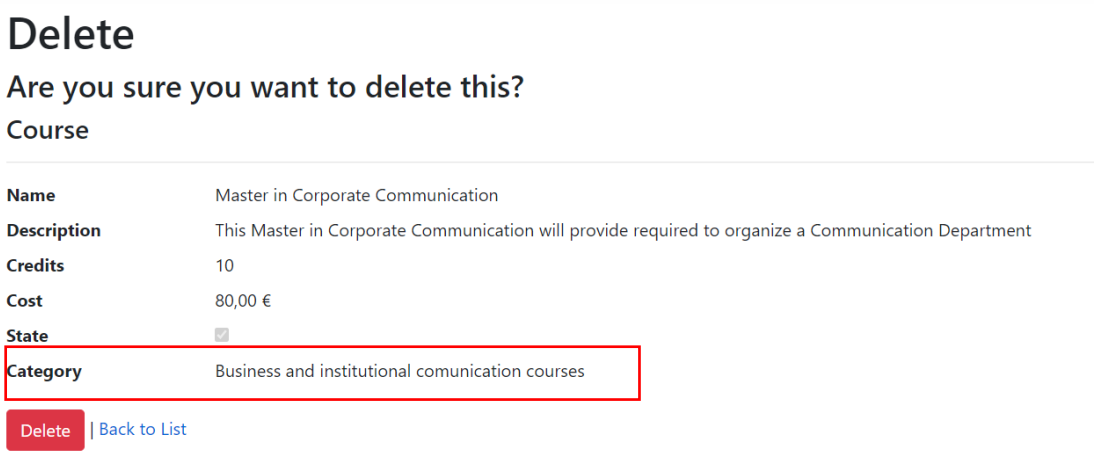
Save

Details

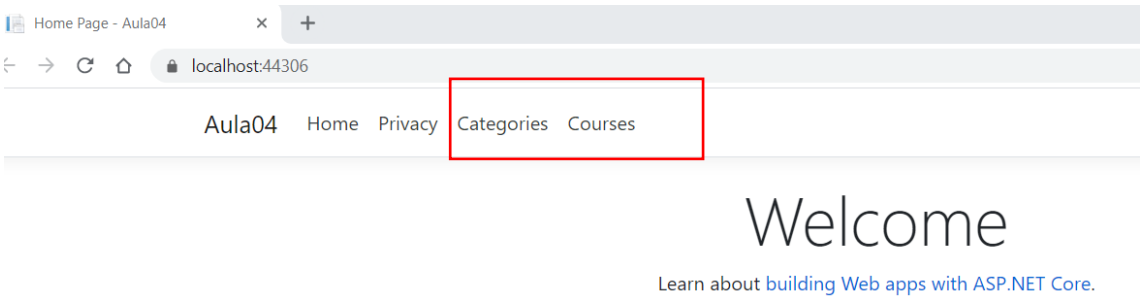
Course

Name	Strategic Leadership and Management
Description	Leadership and Business Skill for Immediate Impact.
Credits	6
Cost	100,00 €
State	☑
Category	Public administration and business management courses

Edit | Back to List



- Create menu items in layout page to manage the categories and courses.



- When creating or editing a new course, only appears to select the categories which State is true.