

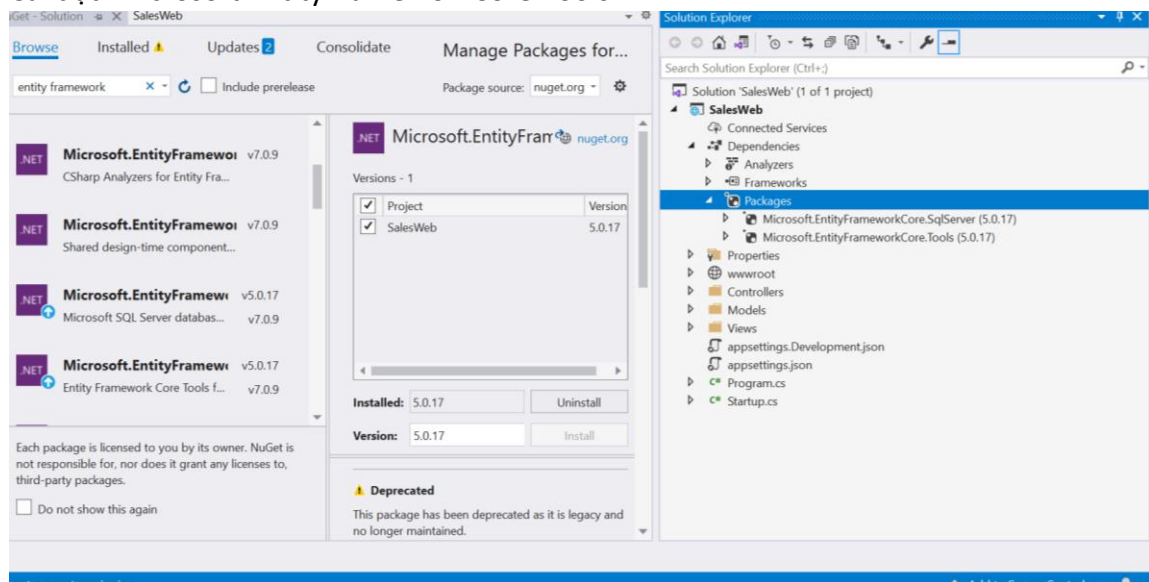
# TRUY CẬP CSDL VỚI ENTITY FRAMEWORK CORE

## Mục tiêu:

- Sử dụng thành thạo Entity Framework (EF) Core
- Tạo ứng dụng ASP.NET Core MVC xử lý các thao tác CRUD với EF Core

## LAB 01: Sử dụng Entity Framework Core theo mô hình Code First

- Tạo dự án web ASP.NET Core Web Application (**EStoreWeb**)
- Cài đặt **Entity Framework Core**
  - a) Cài đặt Microsoft.EntityFrameworkCore.SqlServer
  - b) Cài đặt Microsoft.EntityFrameworkCore.Tools



- Khai báo chuỗi kết nối (connection string) trong tập tin **appsettings.json**

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=.;Initial Catalog=EStoreDB;Integrated Security=True"
  }
}
```

- Tạo lớp **ApplicationDbContext** trong thư mục **Models**

```
using Microsoft.EntityFrameworkCore;

public class ApplicationDbContext: DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)
```

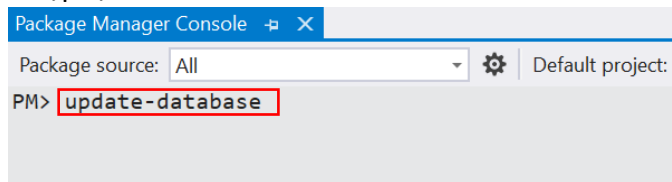
```
{  
}  
}
```

- Đăng ký sử dụng ApplicationDbContext với services trong tập tin **Startup.cs**

```
using Microsoft.EntityFrameworkCore;  
  
public void ConfigureServices(IServiceCollection services)  
{  
    services.AddControllersWithViews();  
    services.AddDbContext<ApplicationDbContext>(  
        options => options.UseSqlServer("name=DefaultConnection"));  
}
```

- ☞ Vào Tools → NuGet Package Manager → Package Manager Console

- Nhập lệnh:



🔔 Lưu ý các lỗi có thể xảy ra khi thi hành lệnh:

- Khai báo sai tên connection string
- Tên database server không có, ...

⇒ **Mở SQL Server Management Studio để kiểm tra kết quả**

- Tạo **Domain Class**

Trong thư mục Models, định nghĩa các class sau:

- ❖ Tạo class **Category** → bổ sung các thuộc tính sau:

```
using System.ComponentModel.DataAnnotations;  
...
```

```
public class Category  
{  
    [Key]  
    public int Id { get; set; }  
    [Required]  
    public string Name { get; set; }  
    public int DisplayOrder { get; set; }  
}
```

- ❖ Tạo class **Product** → bổ sung các thuộc tính sau

```
using System.ComponentModel.DataAnnotations;  
using System.ComponentModel.DataAnnotations.Schema;  
...
```

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

```

    public string Name { set; get; }
    public string Description { set; get; }
    [Required]
    public double Price { set; get; }

    public int CategoryId { set; get; }

    [ForeignKey("CategoryId")]
    public Category Category { set; get; } //khai báo thuộc tính mối kết hợp
    public string ImageUrl { set; get; }
}

```

- Khai báo thuộc tính kiểu `DbSet<TEntity>` cho `ApplicationDbContext` để ánh xạ bảng đến MS SQL Server

```

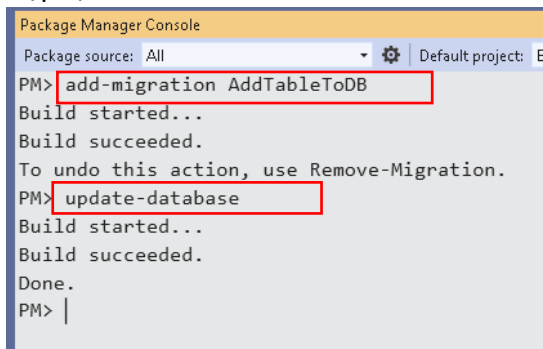
using Microsoft.EntityFrameworkCore;

public class ApplicationDbContext: DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)
    {
    }
    public DbSet<Category> Categories { get; set; }
    public DbSet<Product> Products { get; set; }
}

```

📁 Vào Tools → NuGet Package Manager → Package Manager Console

- Nhập lệnh:



```

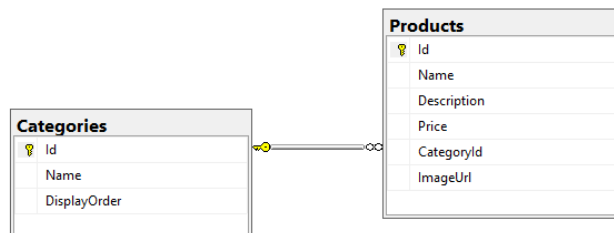
Package Manager Console
Package source: All Default project: E
PM> add-migration AddTableToDB
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> update-database
Build started...
Build succeeded.
Done.
PM>

```

⚠️ Lưu ý các lỗi có thể xảy ra khi thi hành lệnh:

- Chưa khai báo thuộc tính khóa (key) cho thực thể,...

⇒ Mở SQL Server Management Studio để kiểm tra kết quả



- Thêm dữ liệu ban đầu (Seed Data) cho các table : Ghi đè phương thức **OnModelCreating()** trong lớp **ApplicationDbContext** và bổ sung lệnh thêm dữ liệu cho table Categories, table Products

```
using Microsoft.EntityFrameworkCore;

public class ApplicationDbContext: DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)
    {
    }
    public DbSet<Category> Categories { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        //seed data to table Categories
        modelBuilder.Entity<Category>().HasData(
            new Category { Id = 1, Name = "Điện thoại", DisplayOrder = 1 },
            new Category { Id = 2, Name = "Máy tính bảng", DisplayOrder = 2 },
            new Category { Id = 3, Name = "Laptop", DisplayOrder = 3 });

        //seed data to table Product
        modelBuilder.Entity<Product>().HasData(
            new Product { Id = 1, Name = "Iphone 7", Price = 300, CategoryId=1},
            new Product { Id = 2, Name = "Iphone 7s", Price = 350, CategoryId=1},
            new Product { Id = 3, Name = "Iphone 8", Price = 400, CategoryId=1},
            new Product { Id = 4, Name = "Iphone 8s", Price = 420, CategoryId=1},
            new Product { Id = 5, Name = "Iphone 11", Price = 600, CategoryId=1},
            new Product { Id = 6, Name = "Iphone 11s", Price = 650, CategoryId=1},
            new Product { Id = 7, Name = "Iphone 13", Price = 700, CategoryId=1},
            new Product { Id = 8, Name = "Iphone 13 Pro", Price = 850, CategoryId=1},
            new Product { Id = 9, Name = "Iphone 14", Price = 900, CategoryId=1},
            new Product { Id = 10, Name = "Iphone 14 Pro Max ", Price = 1000, CategoryId=1},
            new Product { Id = 11, Name = "Ipad Mini", Price = 350, CategoryId=2},
            new Product { Id = 12, Name = "Ipad Pro", Price = 550, CategoryId=2});
    }
}
```

- Nhập lệnh trong cửa sổ **Package Manager Console**:

1) PM> **add-migration SeedDataToTable**

2) PM> **update-database**

➔ Mở SQL Server Management Studio để kiểm tra kết quả

## LAB 02: Xây dựng chức năng xử lý CRUD

1) Tạo **CategoryController** để lập trình các chức năng quản lý Category

a) Tạo action **Index()** : liệt kê tất cả các Category cần quản lý

- Tạo action **Index()** :
- Tạo view **Index.cshtml** cho action **Index()**

LIST CATEGORY			<a href="#">+ Create New</a>
Name	Display order	Action	
Laptop	1	<a href="#">Edit</a>	<a href="#">Delete</a>
Điện thoại	2	<a href="#">Edit</a>	<a href="#">Delete</a>
Máy tính bảng	3	<a href="#">Edit</a>	<a href="#">Delete</a>

⇒ Mở rộng: Sử dụng bootstrap, icon bootstrap để thiết kế giao diện

### ❖ Hướng dẫn:

- Khai báo biến kiểu **ApplicationDbContext** cho **CategoryController**

```
private ApplicationDbContext db;
```

- Định nghĩa phương thức khởi tạo:

```
public CategoryController(ApplicationDbContext _db)
{
    db = _db;
}
```

- Tạo action **Index()** và bổ sung code sau:

```
public IActionResult Index()
{
    //b1. Lấy tất cả danh mục Category từ CSDL
    List<Category> dstheloai = db.Categories.ToList();
    //b2. truyền qua view
    return View(dstheloai);
}
```

- Tạo view **Index.cshtml** và bổ sung code sau:

```
@model List<Category>
<div class="row mt-4">
    <div class="col-md-6">
        <h2>
            LIST CATEGORY
        </h2>
    </div>
    <div class="col-md-6 text-right">
        <a asp-action="Create" class="btn btn-primary">
            <i class="bi bi-plus-circle-fill"></i> Create New</a>
    </div>
</div>
```

```


        </div>
    </div>

    <table class="table table-bordered table-striped" >
        <thead>
            <tr>
                <th>Name</th>
                <th>Display order</th>
                <th>Action</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var x in Model)
            {
                <tr>
                    <td>@x.Name</td>
                    <td>@x.DisplayOrder</td>
                    <td>
                        <a asp-action="Edit" asp-route-id="@x.Id" class="btn btn-success"><i class="bi bi-pencil-square"></i> Edit</a>
                        <a asp-action="Delete" asp-route-id="@x.Id" class="btn btn-danger"><i class="bi bi-trash"></i> Delete</a>
                    </td>
                </tr>
            }
        </tbody>
    </table>

```

## b) Tạo action Create() : xử lý thêm mới Category

- Tạo action Create() :
- Tạo view Create.cshtml cho action Create()

 **Data Validation: Sử dụng Data Annotations để định nghĩa các quy tắc kiểm tra hợp lệ dữ liệu**

 **Hướng dẫn:**

### B1. Bổ sung 2 action trong CategoryController

```

//action tiếp nhận yêu cầu thêm mới category
public IActionResult Create()
{
    return View();
}

//action tiếp nhận xử lý yêu cầu thêm mới category
[HttpPost]
public IActionResult Create(Category objCategory)
{
    if (ModelState.IsValid) //kiểm tra hợp lệ dữ liệu
    {
        //thêm vào category vào CSDL
        db.Categories.Add(objCategory);
        db.SaveChanges();
        //chuyển hướng đến action Index
        return RedirectToAction("Index");
    }
}

```

```

        return View();
    }

```

## B2. Tạo view Create.cshtml cho action Create()

```

@model Category

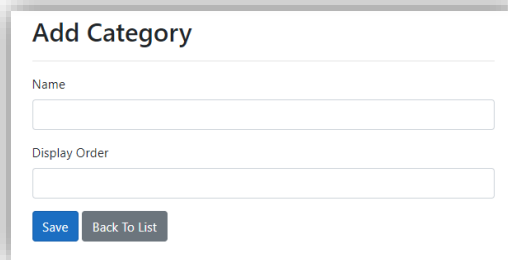
<form asp-action="Create" method="post">

    <h2>Add Category</h2>
    <hr />
    <div class="form-group">
        <label>Name</label>
        <input asp-for="Name" class="form-control" />
        <span asp-validation-for="Name" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label>Display Order</label>
        <input asp-for="DisplayOrder" class="form-control" />
        <span asp-validation-for="DisplayOrder" class="text-danger"></span>
    </div>

    <button type="submit" class="btn btn-primary"> Save </button>
    <a asp-action="Index" class="btn btn-secondary"> Back To List</a>
</form>

@section Scripts{
    <partial name="_ValidationScriptsPartial" />
}

```



### c) Tạo action Edit() : xử lý cập nhật Category

- Tạo action Edit() :
- Tạo view Edit.cshtml cho action Edit()

#### Hướng dẫn:

### B1. Bổ sung 2 action trong CategoryController

```

public IActionResult Edit(int id)
{
    //truy van the loai theo id
    var objCategory = db.Categories.Find(id);
    if (theloai == null)
    {
        return NotFound();
    }
}

```

```

        //tra ve view edit
        return View(objCategory);
    }

    [HttpPost]
    public IActionResult Edit(Category objCategory)
    {
        if (ModelState.IsValid) //kiem tra hop le du lieu
        {
            //cập nhật vào category vào CSDL
            db.Categories.Update(objCategory);
            //hoặc lệnh db.Entry<Category>(objCategory).State = EntityState.Modified;
            db.SaveChanges();
            //chuyen huong den action Index
            return RedirectToAction("Index");
        }
        return View();
    }
}

```

## B2. Tạo view Edit.cshtml cho action Edit()

```

@model Category

<form asp-action="Edit" method="post">

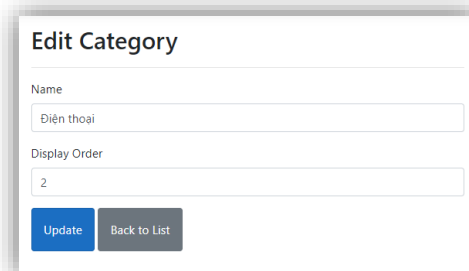
    <h2>Edit Category</h2>
    <hr />

    <input type="hidden" asp-for="Id" />
    <div class="form-group">
        <label>Name</label>
        <input asp-for="Name" class="form-control" />
        <span asp-validation-for="Name" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label>Display Order</label>
        <input asp-for="DisplayOrder" class="form-control" />
        <span asp-validation-for="DisplayOrder" class="text-danger"></span>
    </div>

    <button type="submit" class="btn btn-primary"> Update </button>
    <a asp-action="Index" class="btn btn-secondary"> Back to List</a>
</form>

@section Scripts{
    <partial name="_ValidationScriptsPartial" />
}

```





d) Tạo action Delete() : xử lý xóa Category

 **Hướng dẫn:**

B1. Bổ sung 2 action trong CategoryController

```
public IActionResult Delete(int id)
{
    //truy van the loai theo id
    var objCategory = db.Categories.Find(id);
    if (objCategory == null)
    {
        return NotFound();
    }
    //tra ve view edit
    return View(objCategory);
}
[HttpPost, ActionName("Delete")]
public IActionResult DeletePost(int id)
{
    //truy van the loai theo id
    var objCategory = db.Categories.Find(id);
    if (objCategory == null)
    {
        return NotFound();
    }
    //xóa
    db.Categories.Remove(objCategory);
    db.SaveChanges();
    return RedirectToAction("Index");
}
```

B2. Tạo view Delete.cshtml cho action Delete()

@model Category

```
<form asp-action="Delete" method="post">
    <h2 class="text-primary">Are you sure to delete category ?</h2>
    <hr />

    <input type="hidden" asp-for="Id" />
    <div class="form-group">
        <label>Name</label>
        <input asp-for="Name" disabled class="form-control" />
    </div>
    <div class="form-group">
        <label>Display Order</label>
        <input asp-for="DisplayOrder" disabled class="form-control" />
    </div>

    <button type="submit" class="btn btn-danger"> Delete </button>
    <a asp-action="Index" class="btn btn-secondary "> Back to List</a>

</form>
```

## Are you sure to delete category ?

Name

Máy tính bảng

Display Order

5

Delete

Back to List

## 2) Xử lý hiển thị thông báo (\_Notification) cho ứng dụng CRUD

Tham khảo một số thư viện JavaScripts tạo Notification cho Web Application:

- ❖ Toastr: <https://codeseven.github.io/toastr/>
- ❖ Sweetalert: <https://sweetalert.js.org/guides/>

📁 Hướng dẫn sử dụng **Toastr** hiển thị thông báo cho ứng dụng web :

- **Bước 1:** Mở tập tin `_Layout.cshtml` trong thư mục Shared, bổ sung thẻ liên kết css cho Toastr trong cặp thẻ `<head> </head>`:

```
<link rel="stylesheet"
href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.min.css" />
```

- **Bước 2:** Tạo view `_Notification.cshtml` trong thư mục Shared, bổ sung code sau:

```
@if (TempData["success"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.min.js"></script>
    <script>
        toastr.success('@TempData["success"]')
    </script>
}

@if (TempData["error"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.min.js"></script>
    <script>
        toastr.error('@TempData["error"]')
    </script>
}
```

- **Bước 3:** Nhúng view `_Notification.cshtml` vào các view cần hiển thị thông báo bằng thẻ:

```
<partial name="_Notifications" />
```

Bổ sung thông báo cho các action xử lý tác vụ thêm, sửa xóa:

```
[HttpPost]
public IActionResult Create(Category category)
{
    if (ModelState.IsValid) //kiểm tra hợp lệ dữ liệu
    {
        //thêm vào category vào CSDL
        db.Categories.Add(category);
        db.SaveChanges();
        TempData["success"] = "Category inserted success";
        //chuyển hướng đến action Index
        return RedirectToAction("Index");
    }
    return View();
}
```

```
[HttpPost]
public IActionResult Edit(Category objCategory)
{
    if (ModelState.IsValid) //kiểm tra hợp lệ dữ liệu
    {
        //cập nhật vào category vào CSDL
        db.Categories.Update(objCategory);
        //hoặc lệnh db.Entry<Category>(objCategory).State = EntityState.Modified;
        db.SaveChanges();
        TempData["success"] = "Category updated success";
        //chuyển hướng đến action Index
        return RedirectToAction("Index");
    }
    return View();
}
```

```
[HttpPost, ActionName("Delete")]
public IActionResult DeletePost(int id)
{
    //truy vấn theo id
    var objCategory = db.Categories.Find(id);
    if (objCategory == null)
    {
        return NotFound();
    }
    //xóa
    db.Categories.Remove(objCategory);
    db.SaveChanges();
    TempData["success"] = "Category deleted success";
    return RedirectToAction("Index");
}
```