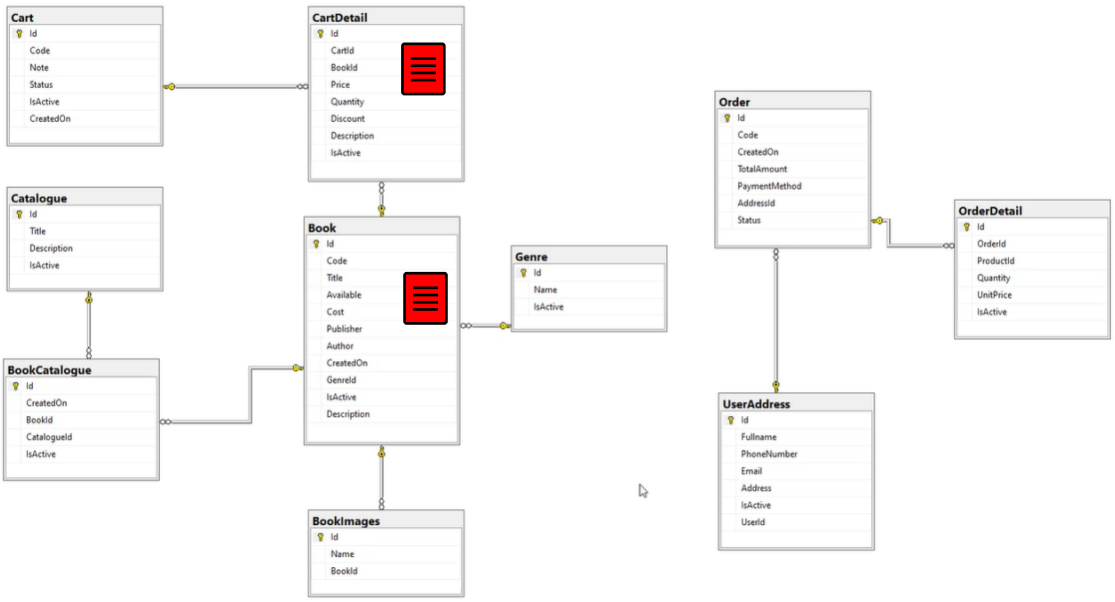
LAB BASIC ENTITY FRAMEWORK

Tạo một ứng dụng ASP.NET Core MVC sử dụng Entity Framework Core để thực hiện các thao tác **Delete** (xóa) và **Update** (cập nhật) dữ liệu trong cơ sở dữ liệu.

ví dụ data là một file cơ sở dữ liệu đã tạo ở bài trước:



Một bảng **BookImage** với các thuộc tính như Id, Name, BookId.

Bước 1: Tạo dự án ASP.NET Core MVC

1. Mở **Visual Studio** hoặc **Visual Studio Code**.
2. Chọn **Create a new project**.
3. Tìm và chọn **ASP.NET Core Web App (Model-View-Controller)**.
4. Đặt tên cho dự án (ví dụ: BookImageApp) và chọn nơi lưu trữ.
5. Chọn **.NET 8 (LTS)** hoặc phiên bản .NET mới nhất.
6. Chọn **Create**.

Bước 2: Cài đặt Entity Framework Core

1. Mở **NuGet Package Manager**:
    - Click phải vào dự án > **Manage NuGet Packages**.

- Chuyển đến tab **Browse** và tìm kiếm các gói sau:
  - **Microsoft.EntityFrameworkCore**
  - **Microsoft.EntityFrameworkCore.SqlServer**
  - **Microsoft.EntityFrameworkCore.Tools**
- Cài đặt tất cả các gói vào dự án của bạn.

Bước 3: Tạo mô hình dữ liệu (Model)

1. Tạo thư mục **Models** trong dự án của bạn.
2. Thêm một class mới vào thư mục **Models**. Đặt tên class là BookImage.cs:

```
namespace BookImageApp.Models
{
    0 references
    public class BookImage
    {
        0 references
        public int Id { get; set; }
        0 references
        public string Name { get; set; }
        0 references
        public int BookId { get; set; }
    }
}
```

Bước 4: Tạo lớp DbContext

1. Tạo thư mục **Data**.
2. Tạo một class mới trong thư mục **Data** với tên ApplicationDbContext.cs:

```
using System;
using Microsoft.EntityFrameworkCore;

namespace Bulkey.Data;

2 references
public class ApplicationDbContext : DbContext
{
    0 references
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
        : base(options)
    {
    }

    0 references
    public DbSet<BookImage> BookImages { get; set; }
}
```

Bước 5: Cấu hình chuỗi kết nối đến cơ sở dữ liệu

1. Mở file **appsettings.json** và thêm chuỗi kết nối đến SQL Server: (không xuống hàng để ko phát sinh lỗi)

```
"ConnectionStrings": {
"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=BookImageAppDB;
                     Trusted_Connection=True;MultipleActiveResultSets=true"
}
```

2. Mở file **Program.cs** và thêm dòng mã sau vào ConfigureServices:

```
builder.Services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")))
```

Bước 6: Tạo Migration và cập nhật cơ sở dữ liệu

1. Mở **Package Manager Console** và chạy các lệnh sau:

Add-Migration InitialCreate

Update-Database

Lệnh này sẽ tạo và áp dụng migration, đồng thời cập nhật cơ sở dữ liệu với bảngm **BookImages**.

Bước 7: Tạo Controller cho BookImage

1. Tạo thư mục **Controllers**.
2. Click phải vào thư mục **Controllers** và chọn **Add** > **Controller**.
3. Chọn **MVC Controller with views, using Entity Framework**.
4. Chọn BookImage model và ApplicationDbContext context.
5. Visual Studio sẽ tự động tạo các phương thức CRUD (Create, Read, Update, Delete) trong BookImageController.cs.

Bước 8: Thực hiện chức năng Update

1. Trong **BookImageController**, phương thức Edit đã được tạo sẵn để xử lý việc cập nhật dữ liệu. Đoạn mã sau sẽ xử lý cập nhật:

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Edit(int id, [Bind("Id,Name,BookId")] BookImage bookImage)
{
    if (id != bookImage.Id)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(bookImage);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!BookImageExists(bookImage.Id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(bookImage);
}

1 reference
private bool BookImageExists(int id)
{
    return _context.BookImages.Any(e => e.Id == id);
}
```

2. **Edit.cshtml** là form hiển thị giao diện để cập nhật thông tin của **BookImage**. Khi người dùng thay đổi thông tin và nhấn **Save**, phương thức Edit sẽ được gọi và dữ liệu sẽ được cập nhật vào cơ sở dữ liệu.

Bước 9: Sinh viên thực hiện chức năng Delete

Trong **BookImageController**, phương thức Delete đã được tạo sẵn để xóa dữ liệu. Dưới đây là mã xử lý:

Bước 10: Kiểm tra project.