# BÀI 8: THỰC HÀNH 03 - XÂY DỰNG WEB API VỚI CRUD 2

***Mục tiêu***

* Xây dựng được CSDL theo yêu cầu
* Kết nối được tới server và CSDL thích hợp
* Xây dựng được API Controller với các Http metthods tương ứng với CRUD
* Vận dụng và tổng hợp được để xây dựng được ứng dụng WebAPI theo yêu cầu
* Nghiêm túc, chăm chỉ, nhiệt tình, tích cực trong học tập

Hướng dẫn thực hiện

**Các bước xây dựng ASP.NET WEB API với cơ sở dữ liệu SQL SERVER**

Bước 1: Tạo cơ sở dữ liệu

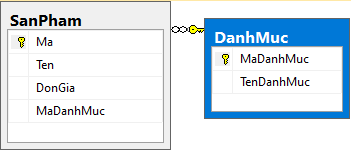
Bước 2: Tạo dự án ASP.NET Web API

Bước 3: Kết nối với cơ sở dữ liệu sử dụng ADO.NET EF

Bước 4: Viết Web API Controller

Bước 5: Chạy và kiểm tra với công cụ

**Yêu cầu:** Xây dựng ASP.NET WEB API, trong đó cơ sở dữ liệu QLSanPham gồm bảng DanhMuc và Bảng SanPham có mối quan hệ như sau:



Chạy và kiểm tra ứng dụng với một trong các công cụ kiểm tra WebAPI Swagger, Postman và Fiddler

***Hướng dẫn***

#### Bước 1: Tạo cơ sở dữ liệu

|  |
| --- |
| CREATE TABLE [dbo].[DanhMuc](  [MaDanhMuc] [int] PRIMARY KEY ,  [TenDanhMuc] [nvarchar](50) NOT NULL  )  CREATE TABLE [dbo].[SanPham](  [Ma] [varchar](30) PRIMARY KEY,  [Ten] [nvarchar](50) NOT NULL,  [DonGia] [int] NULL,  [MaDanhMuc] [int] NULL,  CONSTRAINT [FK] FOREIGN KEY([MaDanhMuc])REFERENCES [dbo].[DanhMuc] ([MaDanhMuc])  ) |

|  |
| --- |
| CREATE PROC ThemDanhMuc (  @MaDanhMuc int,  @TenDanhMuc nvarchar  )  As  BEGIN  INSERT INTO DanhMuc  VALUES (@MaDanhMuc,@TenDanhMuc)  END;  CREATE PROC SuaDanhMuc (  @MaDanhMuc int,  @TenDanhMuc nvarchar  )  As  BEGIN  UPDATE DanhMuc SET TenDanhMuc=@TenDanhMuc  WHERE MaDanhMuc=@MaDanhMuc  END;  CREATE PROC XoaDanhMuc (@MaDanhMuc int )  As  BEGIN  DELETE FROM DanhMuc  WHERE MaDanhMuc=@MaDanhMuc  END;  EXEC ThemDanhMuc 3, N'giáo trình'  EXEC ThemDanhMuc 4, N'Tẩy'  EXEC ThemDanhMuc 5, N'Thước kẻ'  EXEC ThemDanhMuc 6, N'Bút viết bảng' |
| CREATE PROC ThemSanPham (  @Ma varchar(30),  @MaDanhMuc int,  @Ten nvarchar(50),  @DonGia int  )  As  BEGIN  INSERT INTO SanPham(Ma,MaDanhMuc,Ten,DonGia)  VALUES (@Ma,@MaDanhMuc,@Ten,@DonGia)  END;  EXEC ThemSanPham 'vpp001',1, N'giáo trình',10000  EXEC ThemSanPham 'vpp002', 1,N'Tẩy',2000  EXEC ThemSanPham 'vpp003', 1,N'Thước kẻ',8000  EXEC ThemSanPham 'vpp004', 1,N'Bút viết bảng',7000    CREATE PROC SuaSanPham (  @Ma varchar(30),  @MaDanhMuc int,  @Ten nvarchar(50),  @DonGia int  )  As  BEGIN  UPDATE SanPham SET  MaDanhMuc=@MaDanhMuc,  Ten=@Ten,  DonGia=@DonGia  WHERE Ma=@Ma  END;  CREATE PROC XoaSanPham (@Ma varchar(30) )  As  BEGIN  DELETE FROM SanPham  WHERE Ma=@Ma  END; |

#### Bước 2: Tạo dự án ASP.NET Web API

Khởi động Visual Studio → tạo một project ASP.NET Web Application và chọn template Web API.

Đặt tên cho ứng dụng, chọn nơi lưu trữ và FrameWork rồi chọn Create

Chọn template WebAPI

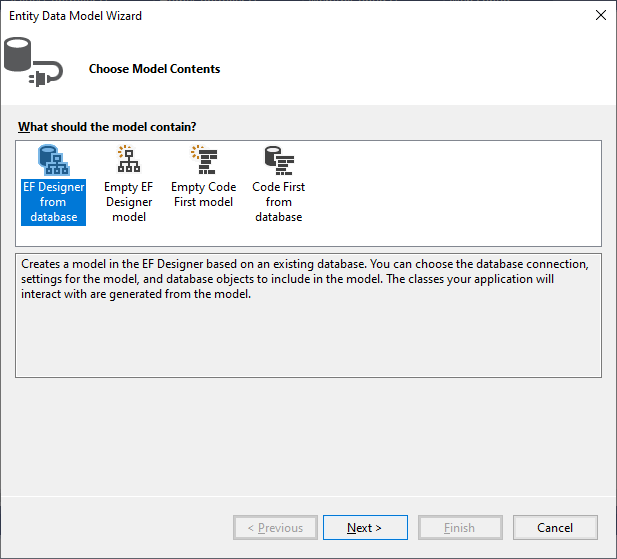
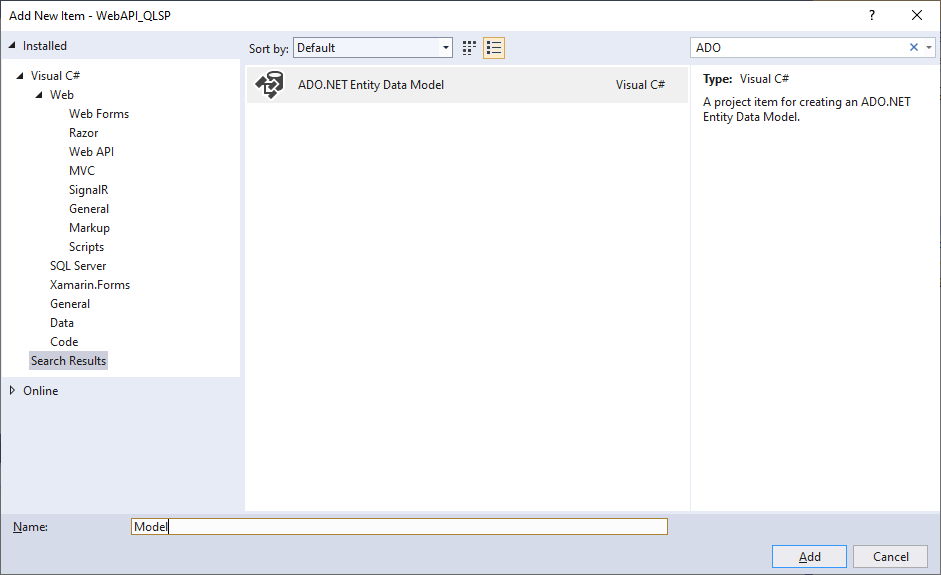
#### Bước 3: Kết nối với cơ sở dữ liệu sử dụng ADO.NET EF

Để code tạo các model, nhấn chuột phải vào Models và chọn Add/New item

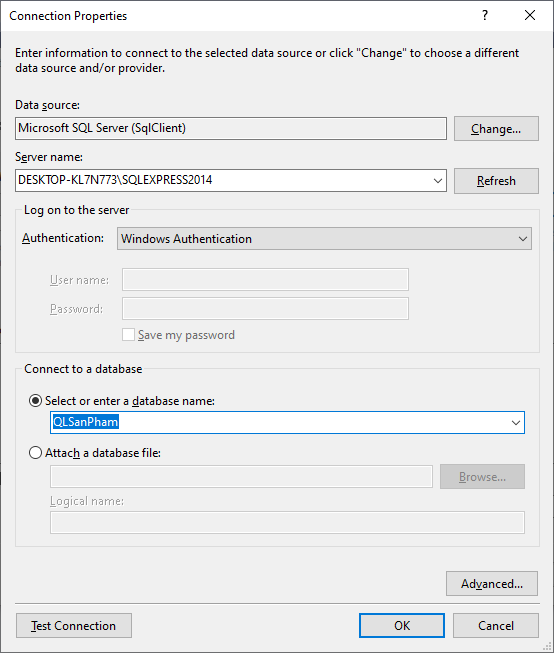
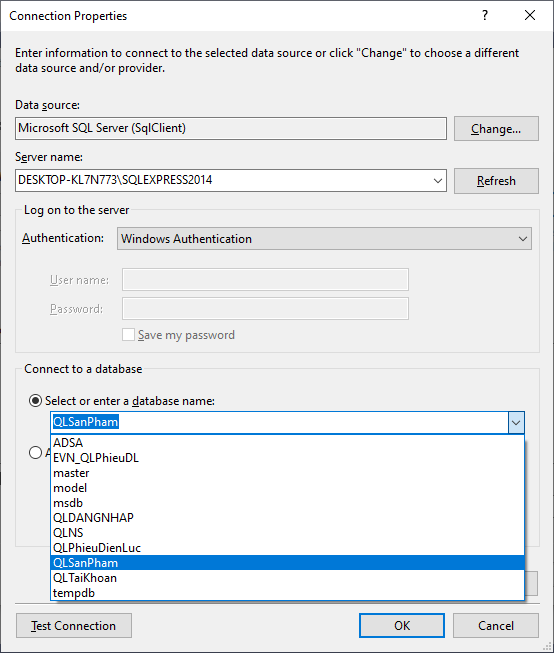
Nhập ADO vào ô search để lọc tìm cho nhanh hoặc tìm ADO.NET Entity Data Model

Nhập tên cho class, rồi chọn Add

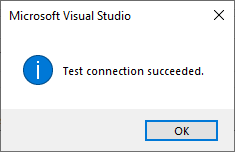
Chọn EF Designer from database, chọn Next



Chọn New Connection (nếu server name và tên CSDL chưa đúng)



Copy server name (server chứa CSDL cho ứng dụng và server đang running), rồi chọn CSDL. Nhấn Test Connection để kiểm tra xem có kêt nối được tới CSDL hay không (successfull là thành công)



Next tiếp cho tới khi xuất hiện giao diện chọn table và Proc

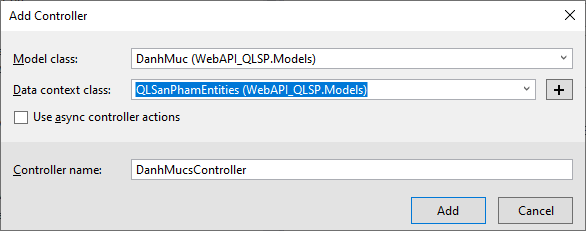
Có thể chọn từng model tương ứng table hoặc all tables cần thiết

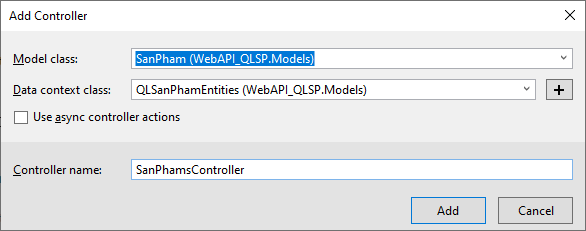
Chọn các store proc và function cần thiết (giảm tải viết code controller vì .Net sẽ có cơ chế tự sinh code tương ứng)

#### Bước 4: Viết Web API Controller

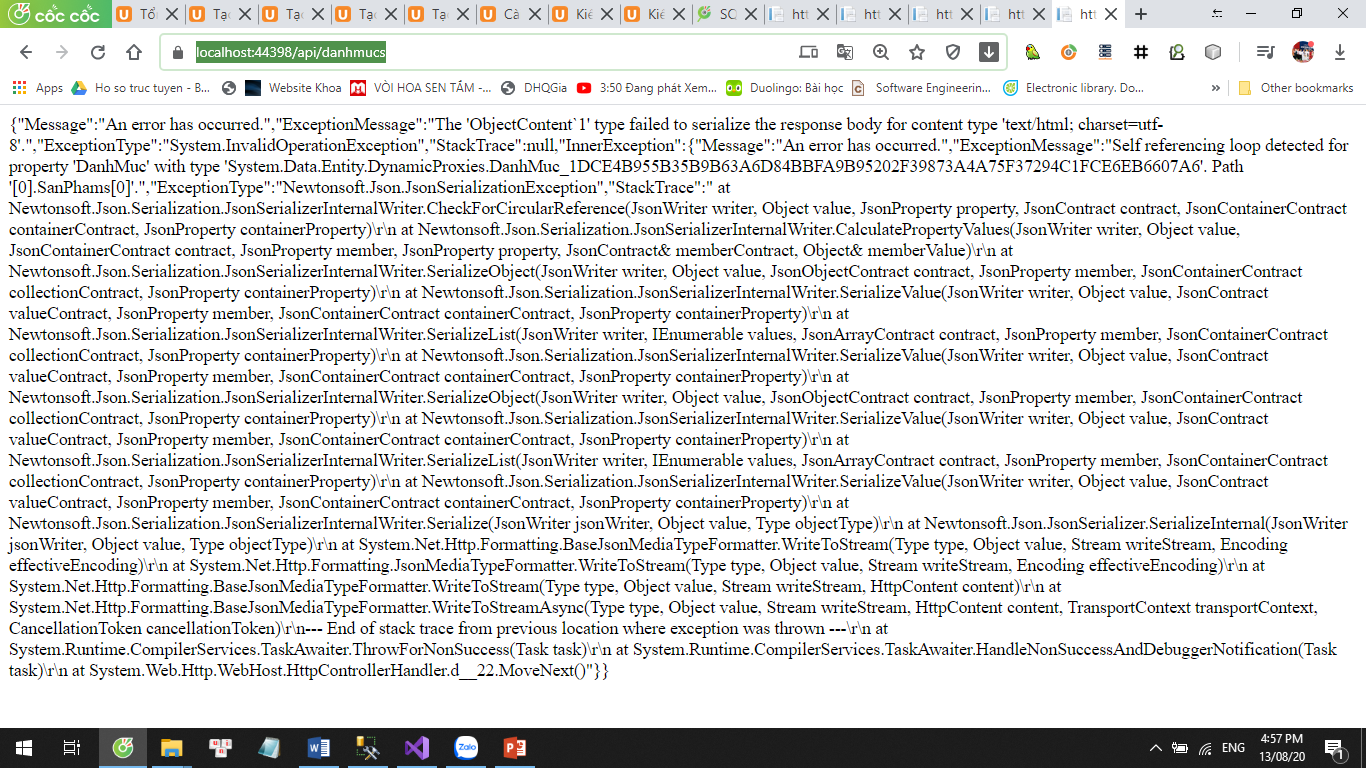
Nhấp chuột phải vào thư mục Controllers và chọn thêm controller.

Chọn **Web API 2 Controller with read/write actions**: phát sinh các phương thức ví dụ để bạn có thể biết cách viết các service này.



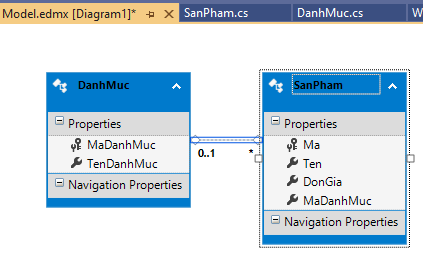


Xử lý lỗi phát sinh khi có mối quan hệ rang buộc giữa cách bảng

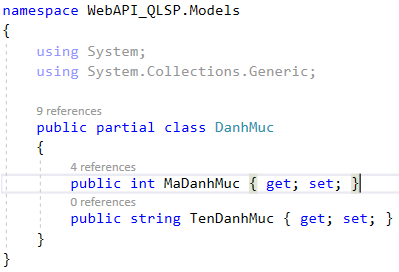
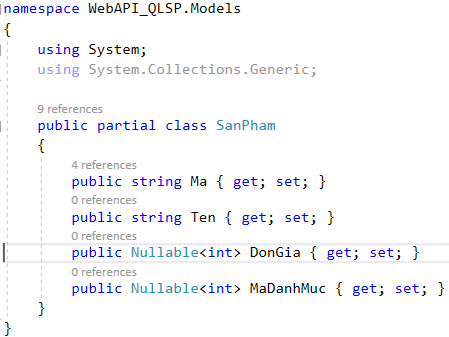


CThêm [JsonIgnore] và [XmlIgnore] trong entity trong model

Cách 1 Mở Model.edmx và xử lý lại mối quan hệ ràng buộc giữa các bảng



Cách 2 Mở Models/danhMuc.cs và Models/SanPham.cs, xử lý lại phần định nghĩa các thuộc tính, xóa phần cuối đi (ở cuối file)

#### Bước 5: Chạy và kiểm tra với công cụ

### DanhMucsController.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.Entity;  using System.Data.Entity.Infrastructure;  using System.Linq;  using System.Net;  using System.Net.Http;  using System.Web.Http;  using System.Web.Http.Description;  using WebAPI\_QLSP.Models;  namespace WebAPI\_QLSP.Controllers  {  public class DanhMucsController : ApiController  {  private QLSanPhamEntities db = new QLSanPhamEntities();  // GET: api/DanhMucs  public IQueryable<DanhMuc> GetDanhMucs()  {  return db.DanhMucs;  }  // GET: api/DanhMucs/5  [ResponseType(typeof(DanhMuc))]  public IHttpActionResult GetDanhMuc(int id)  {  DanhMuc danhMuc = db.DanhMucs.Find(id);  if (danhMuc == null)  {  return NotFound();  }  return Ok(danhMuc);  }  // PUT: api/DanhMucs/5  [ResponseType(typeof(void))]  public IHttpActionResult PutDanhMuc(int id, DanhMuc danhMuc)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  if (id != danhMuc.MaDanhMuc)  {  return BadRequest();  }  db.Entry(danhMuc).State = EntityState.Modified;  try  {  db.SaveChanges();  }  catch (DbUpdateConcurrencyException)  {  if (!DanhMucExists(id))  {  return NotFound();  }  else  {  throw;  }  }  return StatusCode(HttpStatusCode.NoContent);  }  // POST: api/DanhMucs  [ResponseType(typeof(DanhMuc))]  public IHttpActionResult PostDanhMuc(DanhMuc danhMuc)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  db.DanhMucs.Add(danhMuc);  try  {  db.SaveChanges();  }  catch (DbUpdateException)  {  if (DanhMucExists(danhMuc.MaDanhMuc))  {  return Conflict();  }  else  {  throw;  }  }  return CreatedAtRoute("DefaultApi", new { id = danhMuc.MaDanhMuc }, danhMuc);  }  // DELETE: api/DanhMucs/5  [ResponseType(typeof(DanhMuc))]  public IHttpActionResult DeleteDanhMuc(int id)  {  DanhMuc danhMuc = db.DanhMucs.Find(id);  if (danhMuc == null)  {  return NotFound();  }  db.DanhMucs.Remove(danhMuc);  db.SaveChanges();  return Ok(danhMuc);  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  private bool DanhMucExists(int id)  {  return db.DanhMucs.Count(e => e.MaDanhMuc == id) > 0;  }  }  } |

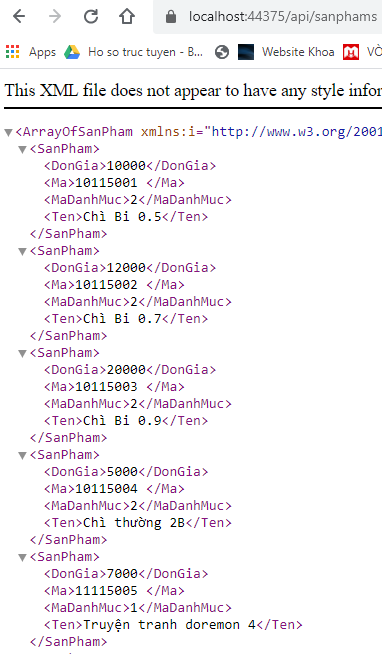
### SanPhamsController.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.Entity;  using System.Data.Entity.Infrastructure;  using System.Linq;  using System.Net;  using System.Net.Http;  using System.Web.Http;  using System.Web.Http.Description;  using WebAPI\_QLSP.Models;  namespace WebAPI\_QLSP.Controllers  {  public class SanPhamsController : ApiController  {  private QLSanPhamEntities db = new QLSanPhamEntities();  // GET: api/SanPhams  public IQueryable<SanPham> GetSanPhams()  {  return db.SanPhams;  }  // GET: api/SanPhams/5  [ResponseType(typeof(SanPham))]  public IHttpActionResult GetSanPham(string id)  {  SanPham sanPham = db.SanPhams.Find(id);  if (sanPham == null)  {  return NotFound();  }  return Ok(sanPham);  }  // PUT: api/SanPhams/5  [ResponseType(typeof(void))]  public IHttpActionResult PutSanPham(string id, SanPham sanPham)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  if (id != sanPham.Ma)  {  return BadRequest();  }  db.Entry(sanPham).State = EntityState.Modified;  try  {  db.SaveChanges();  }  catch (DbUpdateConcurrencyException)  {  if (!SanPhamExists(id))  {  return NotFound();  }  else  {  throw;  }  }  return StatusCode(HttpStatusCode.NoContent);  }  // POST: api/SanPhams  [ResponseType(typeof(SanPham))]  public IHttpActionResult PostSanPham(SanPham sanPham)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  db.SanPhams.Add(sanPham);  try  {  db.SaveChanges();  }  catch (DbUpdateException)  {  if (SanPhamExists(sanPham.Ma))  {  return Conflict();  }  else  {  throw;  }  }  return CreatedAtRoute("DefaultApi", new { id = sanPham.Ma }, sanPham);  }  // DELETE: api/SanPhams/5  [ResponseType(typeof(SanPham))]  public IHttpActionResult DeleteSanPham(string id)  {  SanPham sanPham = db.SanPhams.Find(id);  if (sanPham == null)  {  return NotFound();  }  db.SanPhams.Remove(sanPham);  db.SaveChanges();  return Ok(sanPham);  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  private bool SanPhamExists(string id)  {  return db.SanPhams.Count(e => e.Ma == id) > 0;  }  }  } |

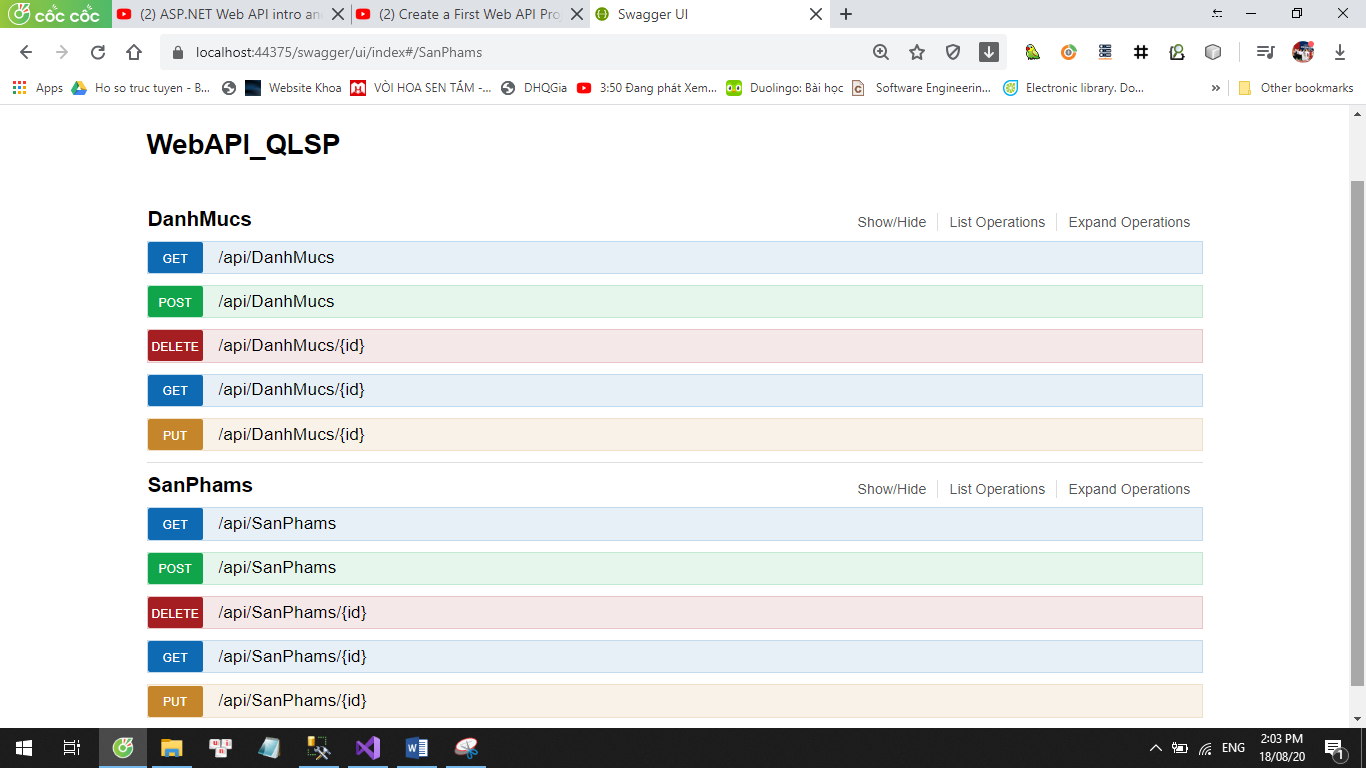
#### Bước 5: Chạy và kiểm tra với công cụ

<https://localhost:44375/api/danhmucs>

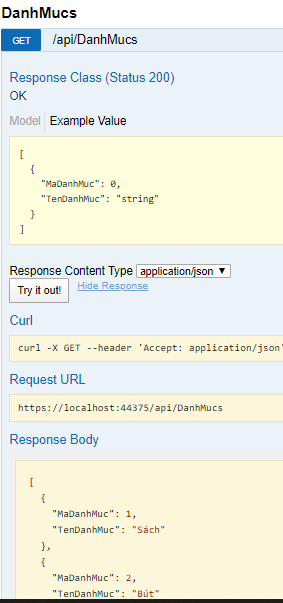
<https://localhost:44375/api/sanphams>



### Help and test với swagger <https://localhost:44375/swagger/ui/index>



#### Get

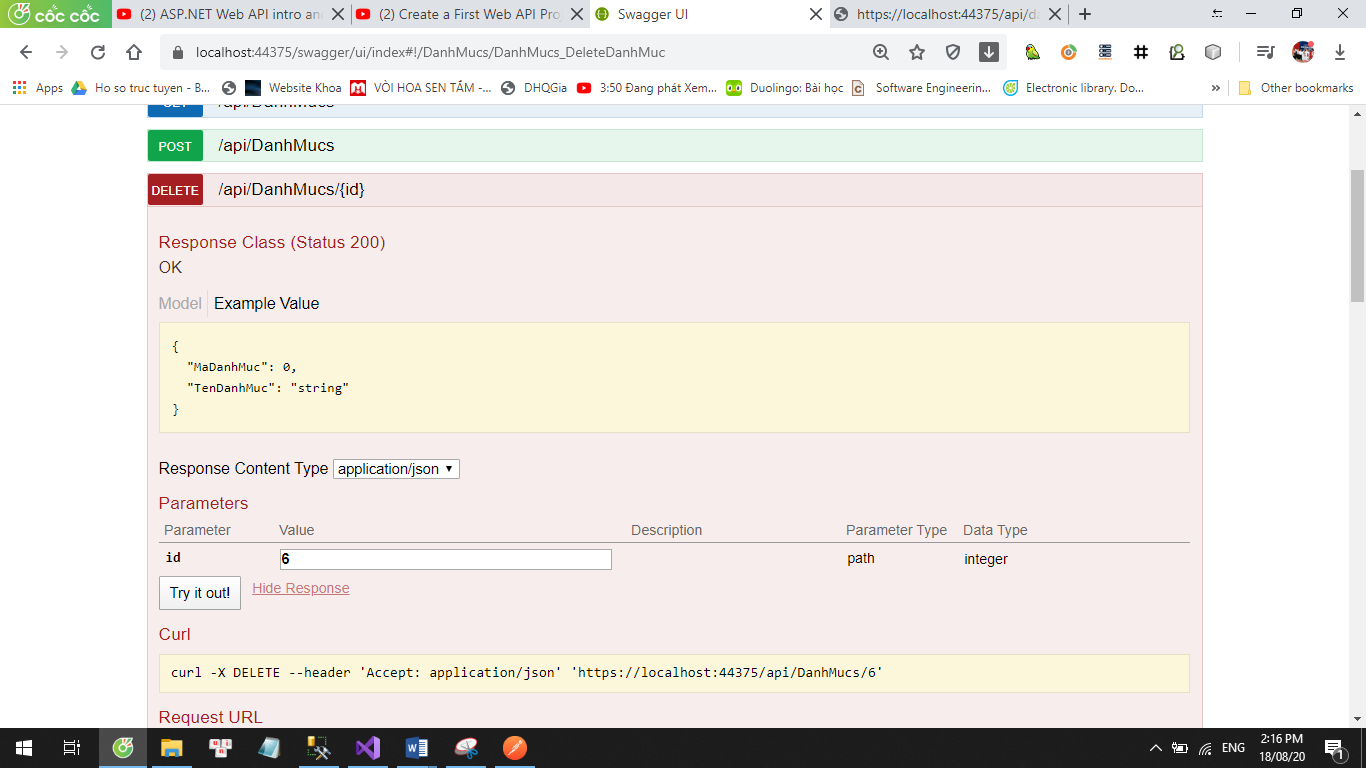


#### Xóa 1 DanhMuc có id là 6 với swagger

B1 Chọn Delete

B2: Nhập id, r chọn Try it out

B3 Xem lại kết quả = Response Code trong 200..300 thì ok, còn lại là lỗi



Sau là TH xóa thành công với respond code 200 và xóa thất baij với respond code là 404 (chưa có trong CSDL)

#### Post Thêm 1 DanhMuc có id với swagger

|  |  |
| --- | --- |
| TH thêm thành công với respond code 201  Trong CSDL chưa có DanhMuc có id là 1 | TH thêm thất bại với respond code 409  vì đã có DanhMuc có id là 1 |
| {  "MaDanhMuc": 10,  "TenDanhMuc": "Đồ chơi"  } | {  "MaDanhMuc": 1,  "TenDanhMuc": "Đồ chơi"  } |

#### Put Sửa 1 DanhMuc có id là 6 với swagger

|  |  |
| --- | --- |
| TH thêm thành công với respond code 201  Trong CSDL chưa có DanhMuc có id là 1 | TH thêm thất bại với respond code 400  vì đã có DanhMuc có id là 100 |
| Id là 1 | Id 100 |
| {  "MaDanhMuc": 1,  "TenDanhMuc": "Đồ chơi"  } | {  "MaDanhMuc": 100,  "TenDanhMuc": "Đồ chơi"  } |

Bài tập tự làm

**Yêu cầu:**

Tạo CSDL QLBanHang gồm các bảng LoaiHang HangHoa và GiaBan có quan hệ như sau:



Xây dựng WebAPI với thực hiện CRUD

Chạy và kiểm tra ứng dụng với Swagger, Postman và Fiddler

Hướng dẫn

|  |
| --- |
| CREATE DATABASE QLBANHANG  USE QLBANHANG  CREATE TABLE LoaiHang(  MaLoai nvarchar(30) PRIMARY KEY,  TenLoai nvarchar(50) NOT NULL,  MoTa nvarchar(50)  )  CREATE TABLE HangHoa(  MaHang nvarchar(30) PRIMARY KEY,  MaLoai nvarchar(30) not null,  Tenhang nvarchar(50),  SoLuong int default 10,  SoLuongCon int default 1,  CONSTRAINT FR\_loaihang\_hanghoa FOREIGN KEY (MaLoai) REFERENCES LoaiHang(MaLoai)  )  CREATE TABLE GiaBan(  MaGB nvarchar(30) PRIMARY KEY,  MaHang nvarchar(30) not null,  Gia int default 0,  DVTinh nvarchar(30),  NgayBD date default getdate(),  NgayKT date default getdate()+10,  CONSTRAINT FR\_hanghoa\_giaban FOREIGN KEY (MaHang) REFERENCES HangHoa(MaHang)  ) |

public class LoaiHang

{

string maLoai;

string tenLoai;

string moTa;

}

public class HangHoa

{

string maHH;

string maLoai;

string tenhang;

int soLuong;

int soLuongCon;

}

public class GiaBan

{

string maGB;

string maHang;

int Gia;

string DVTinh;

DateTime NgayBD;

DateTime NgayKT;

}

**Bài tập 2** Xây dựng WebAPI chạy và kiểm tra ứng dụng với CSDL được cho như sau:

1 Cho các CSDL QLSinhVien gồm các quan hệ

**SinhVien** (MASV, HOTEN)

**LopHoc**(KYHIEU, TENMONHOC, THOIGIAN)

**GiaoVien** (MAGV, TENGV)

2 Cho các CSDL QLSinhVien gồm các quan hệ

**BangDiem** (MaSV, Ma\_mon, Diem)

**SinhVien** (MaSV, Ten\_SV, Gioi\_tinh, Dia\_chi, Ngay\_sinh)

**MonHoc** (MaMH, Ten\_mon, Chuyen\_nganh, So\_hoc\_trinh)

**GiaoVien** (MaGV, Ten\_GV, Chuyen\_nganh, Dia\_chi, Dien\_thoai)

**GV\_DAY** (MaGV, MaMH)

3 Cho các CSDL QLNhanVien gồm các quan hệ

**Nhanvien** (MANV, HOTEN, NGAYSINH, PHAI, DIACHI, MAPB)

**Phongban** (MAPB, TENPB)

**Cong** (MACT, MANV, SLNGAYCONG)

**Congtrinh** (MACT, TENCT, DIADIEM, NGAYCAPGP, NGAYKC)

4 Cho các CSDL QLThuVien gồm các quan hệ

**Docgia** (MADG, HOTEN, NGAYSINH, DIACHI, NGHENGHIEP)

**Phieumuon** (SOPM, NGAYMUON, MADG)

**Chitietmuon** (SOPM, MADAUSACH, NGAYTRA)

**Dausach** (MADAUSACH, BAN, TAP, MASH)

**Sach** (MASH, TENSACH, TACGIA, NHAXB, NAMXB)

5 Cho các CSDL QLdự án gồm các quan hệ

