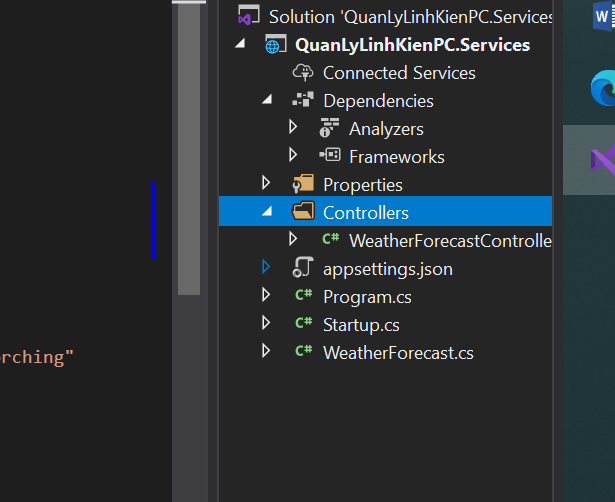
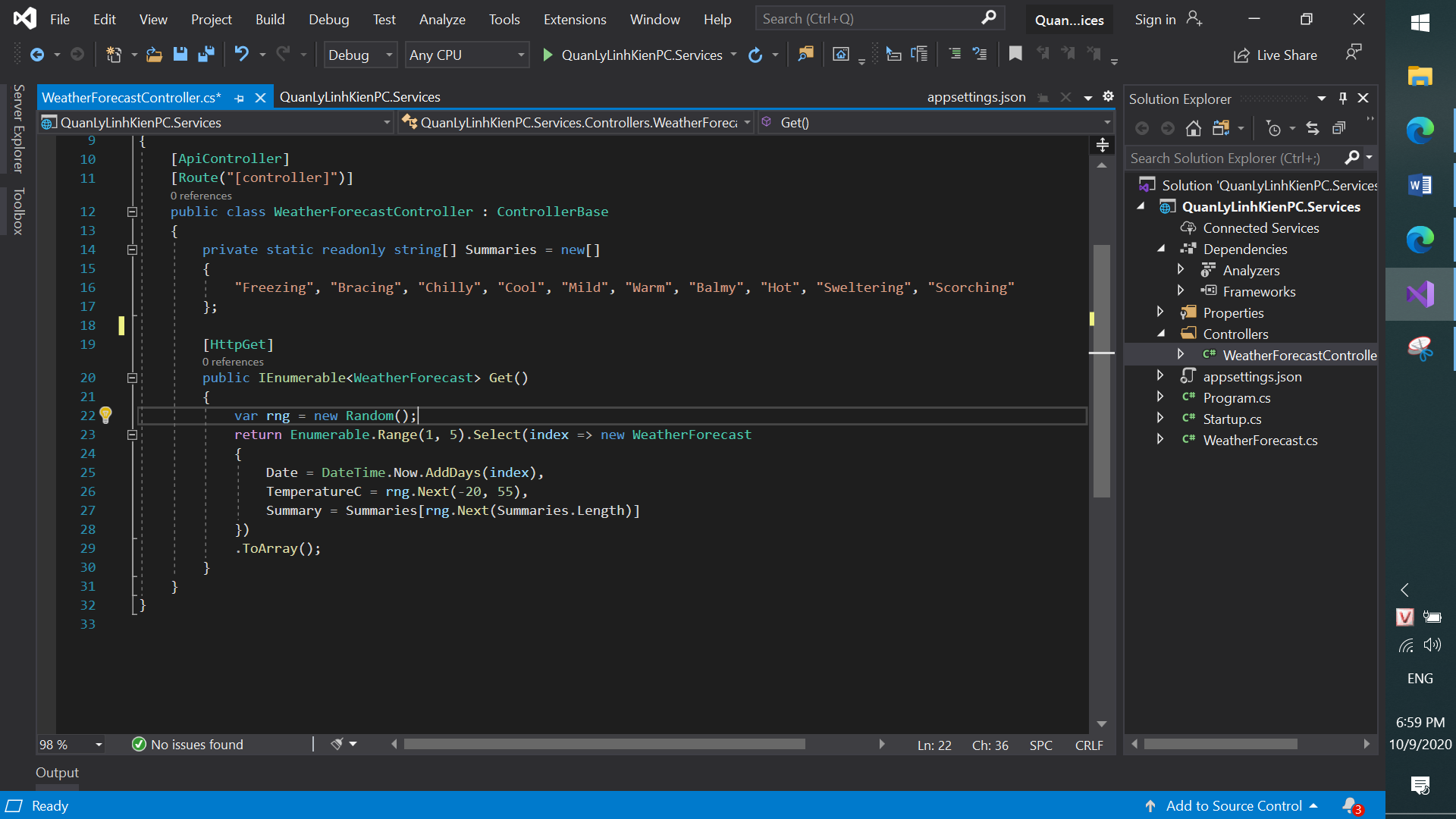
|  |
| --- |
| 1.Web Api là gì  API được viết tắt bởi cụm từ tiếng Anh **Application Programming Interface**, tức giao diện lập trình ứng dụng. Đây là phương tiện cho hai hoặc nhiều ứng dụng trao đổi, tương tác với nhau, tạo ra tương tác giữa người dùng với ứng dụng hiệu quả và tiện lợi hơn.  Rest API  Trong REST, 4 lệnh có tên gọi hơi khác một chút:   * Post: Có chức năng tạo dữ liệu, thông tin mới * Get: Lệnh đọc/lấy một dữ liệu, thông tin mới * Put: Cập nhật thông tin và dữ liệu đã có * Delete: Xóa thông tin và dữ liệu đã có   Trong đs, Get được đánh giá là lệnh an toàn và phổ biến nhất, được cho phép bởi hầu hết các Restful API hiện nay. 3 lệnh còn lại ít nhiều làm theo đổi các dữ liệu gốc máy chủ, có nguy cơ dẫn tới tai nạn ngoài ý muốn. Cho nên, chỉ những API được thiết kế riêng hay được cấp thẩm quyền như API đối tác, Admin mới được thực hiện các lệnh đó.  **Chức năng tìm kiếm**  Công cụ tìm kiếm (search engine – SE) là một bộ phận cơ bản của các trang web. Nhưng nếu bạn sử dụng chức năng tìm kiếm thông thường, người dùng có thể sẽ khó mà tìm được những thông tin mong muốn khi số lượng nội dung quá phức tạp, đồ sộ hay tìm kiếm một nội dung đặc thù nào đó.  Ví dụ: nếu website nấu ăn hẳn sẽ thường được tìm kiếm về các công thức, thông tin dinh dưỡng,… Không những thế, khi cần thiết họ sẽ tìm kiếm về cả các món ăn ít chất béo, dành cho trẻ em hay bà bầu, các thành phần có thể gây dị ứng, nấu trong bao lâu là đủ,… Rất nhiều điều được quan tâm tìm kiếm.  Khi đó, website đòi hỏi phải có một API sử dụng SE mạnh mẽ, phổ biến hỗ trợ người dùng tìm kiếm được thông tin chính xác nhất. Điều này không chỉ nâng cao trải nghiệm người dùng mà còn góp phần giúp đem tới cái nhìn tích cực hơn từ người dùng.  **Khả năng tích hợp kênh mạng xã hội**  Các kênh mạng xã hội như Zalo, Facebook,… đang trở thành những thị trường lớn, nơi thu hút và sở hữu đông đảo người dùng trên toàn thế giới. Hầu hết các website hiện đại đều tích hợp với các nền tảng mạng xã hội.  Chúng ta có thể dễ dàng thấy điều này khi kéo xuống cuối trang web, bên sườn giao diện website hay dưới mỗi bài viết. Một số trang web còn sử dụng cả khung bình luận, nút like, theo dõi hay share Fanpage cho các bài viết hay cuộc thảo luận dưới bài viết. Người xem có thể dùng chính tài khoản mạng xã hội của mình để đăng nhập và tương tác. Rất tiện lợi và nhanh chóng.  Muốn làm được điều đó thì các lập trình và thiết kế website cần sử dụng API của nền tảng mạng xã hội nhúng các chức năng mạng xã hội này vào trang web của mình. Thông qua API, website và mạng xã hội có thể trao đổi dữ liệu qua lại dễ dàng hơn. Người dùng cũng không cần thoát website mà vẫn có thể tương tác với cả ứng dụng mạng xã hội một cách bình thường và thoải mái.  **Xây dựng website bán hàng trực tuyến**  Không chỉ cải thiện trải nghiệm người dùng, API còn là công cụ quan trọng hỗ trợ công việc của các lập trình viên. Nhất là trong việc xây dựng hệ thống hạ tầng website, thiết kế website bán hàng trên nền tảng shopify (nền tảng bán hàng online đầy đủ chức năng của trang Thương mại điện tử).  Chỉ cần sử dụng API của Shopify, gắn vào các nút, thành phần website, trang web của bạn sẽ có những chức năng điển hình mà một web thương mại điện tử cần có. Điều này rút ngắn được quá trình xây dựng nhưng vẫn đảm bảo đầy đủ các tính năng cần thiết cho một website bán hàng online.  API cơ bản chứa  kf  **Request URL: đường dẫn mà có thể dùng call api**  Request Method: GET/POST/PUT/DELETE  Status Code: 200 : thông báo trạng của request   * 200, 201,203,… 2xx thì về cơ bản là thành công! * 400: bad quest: call được nhưng không/ trống dữ liệu * 401: chưa xác thực/ chưa đính token hoặc token này hết hạn * 403: có đính kèm token nhưng k có quyền truy cập chức năng đó * 404: Không tìm thấy **Request URL( mình viết đường dẫn or đường dẫn đó không tồn)** * **405: sai method: đúng url, đúng token, => sai phương thức**   Ví dụ: api nó là phương thức get – nhưng mình dùng là post/put/delete   * 500: lỗi code -> gặp exceptions ( do vòng lặp, hoặc các đối tượng bị lặp vô hạn trong lúc truy vấn linq)   Đề bài: Quản lý linh kiện máy tính – nhỏ   * CRUD ( create-read-update-delete) thêm sửa xóa : danh mục linh kiện * Card, main, chip, ram, fan, source, case, disk,… * => DanhMucLinhKien * Main: => main H310, main H77, main H81,….. |
| * (hình ảnh, tên, mã sản phẩm, năm sản xuất, giá thành…. * => bảng: LinhKien * => 2 bảng…. * Nhiệm là thêm sửa xóa cho 2 bảng này dùng back-end: asp.net core (linq) – EF * Front-end: Angular 10 – material design  1. Thiết kế cơ sở dữ liệu   Đọc yêu cầu (khách hàng) – PM đẻ ra.  Phân tích các yêu cầu => đánh giá làm được không ? |
| * DanhMucLinhKien: * ID, Ten, Ma, Trang Thái, Hãng. * LinhKien: * ID, Ma,Ten, Trang Thai, DanhMucLinhKienID, SoLuong, Gia, Hình ảnh. |
| * Sau khi phân tích tạo ra các bảng SQL * Tạo project -> ASP.Net Core 3.1 – define cái cấu trúc dự án * Models, Controlers, ViewModels, TextHepler: tạo các id ngẫu nhiên, một số các hàm xử lý toán ngoài lề. * TenDuAn + Services/ Module |



Controller: Chứa, được gọi trực tiếp từ UI – để tương tác API

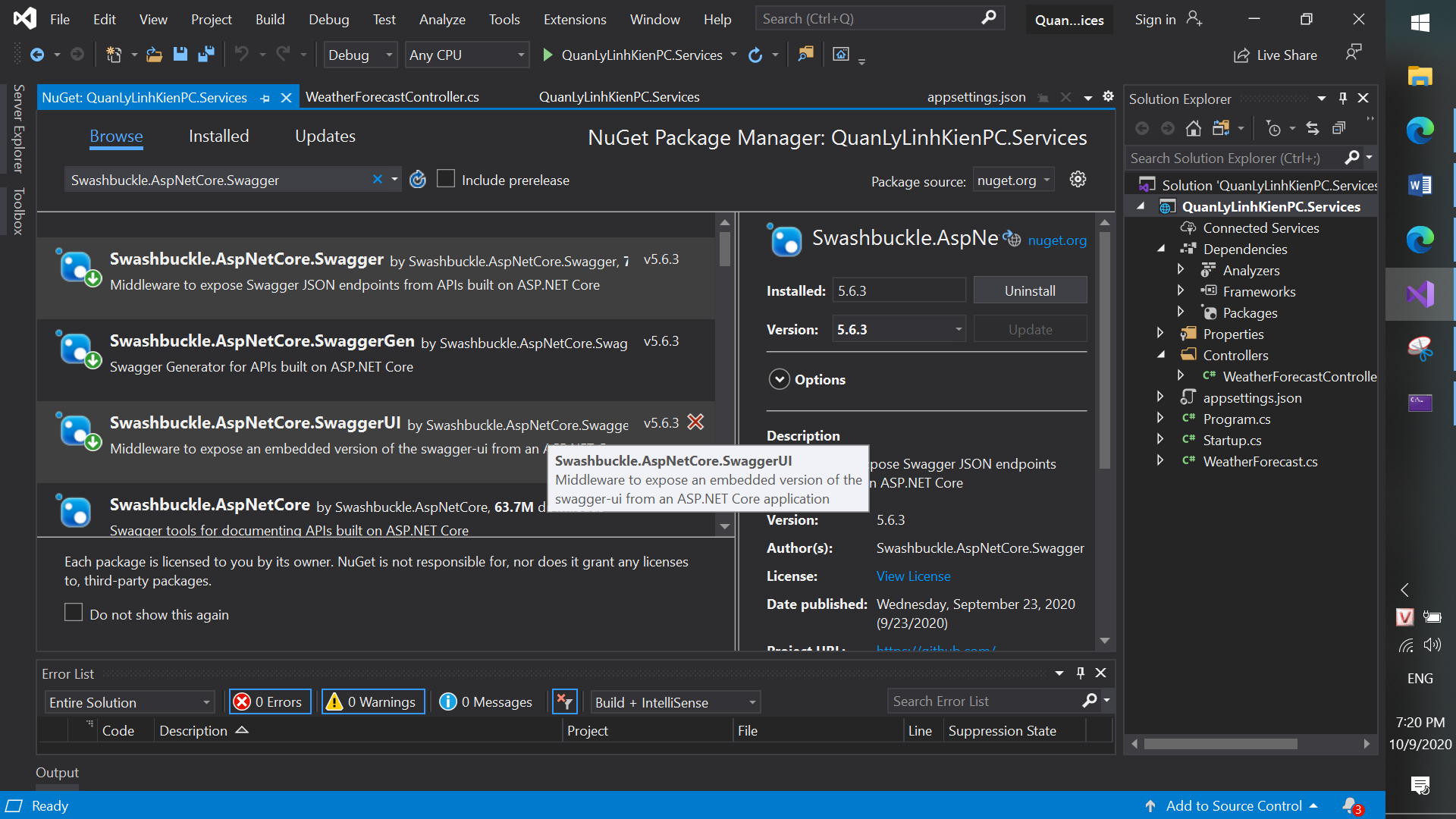


**Request URL:** [**https://localhost:5001/**WeatherForecast/tinh-toan/1/2](https://localhost:5001/WeatherForecast/tinh-toan/1/2)

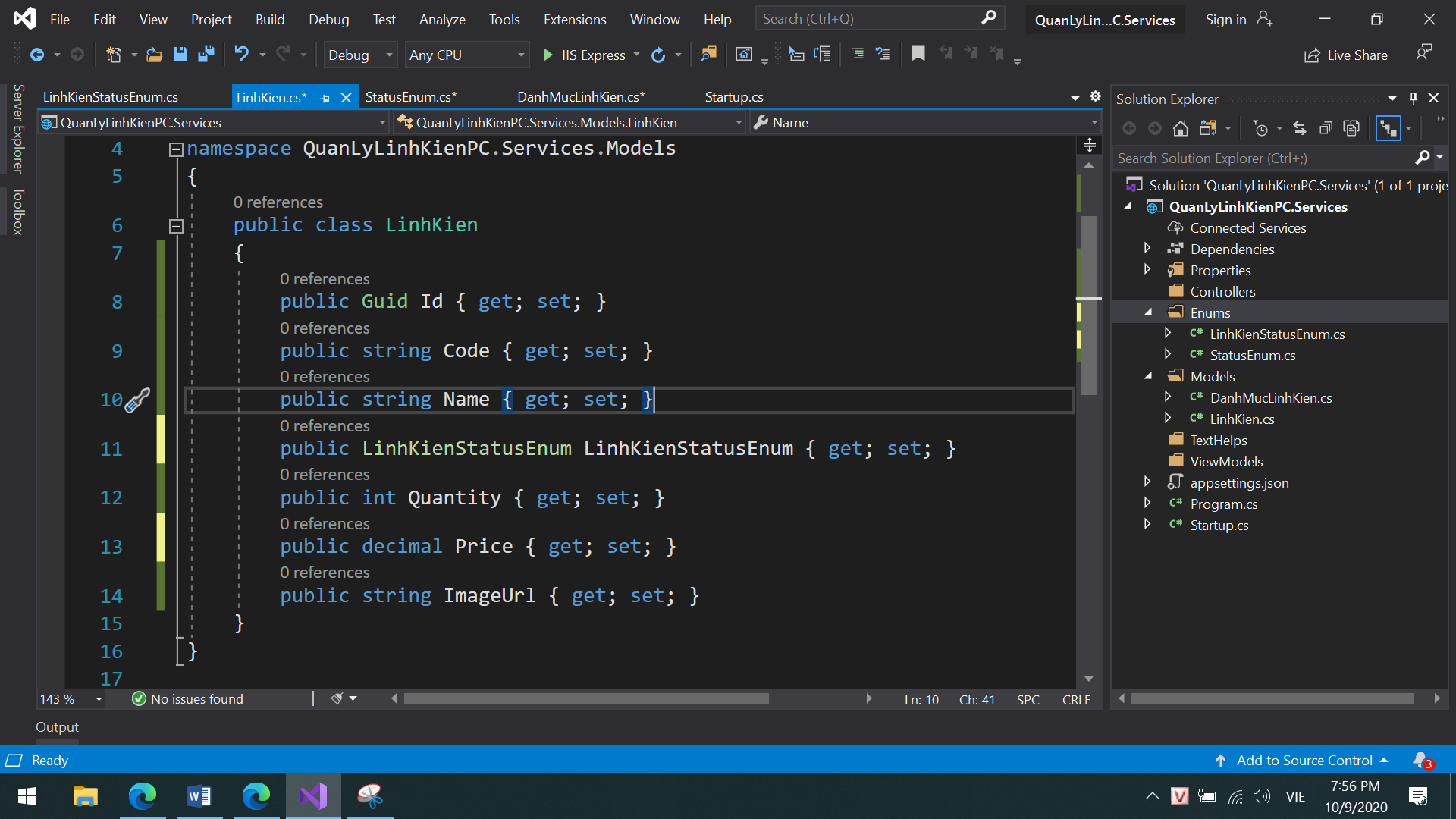
Cài swagger 5.6.3

<https://docs.microsoft.com/en-us/aspnet/core/tutorials/getting-started-with-swashbuckle?view=aspnetcore-3.1&tabs=visual-studio>

Swashbuckle.AspNetCore.Swagger



Cấu hình project



Cài EF cho asp.net core

Microsoft.EntityFrameworkCore // anotation

Microsoft.EntityFrameworkCore.SqlServer // có hàm kết nối

Microsoft.EntityFrameworkCore.Tools // có lệnh generate db từ class -> SQL server

Dùng phong cách Code first ( viết code trước -> generate ra db ở SQL server)

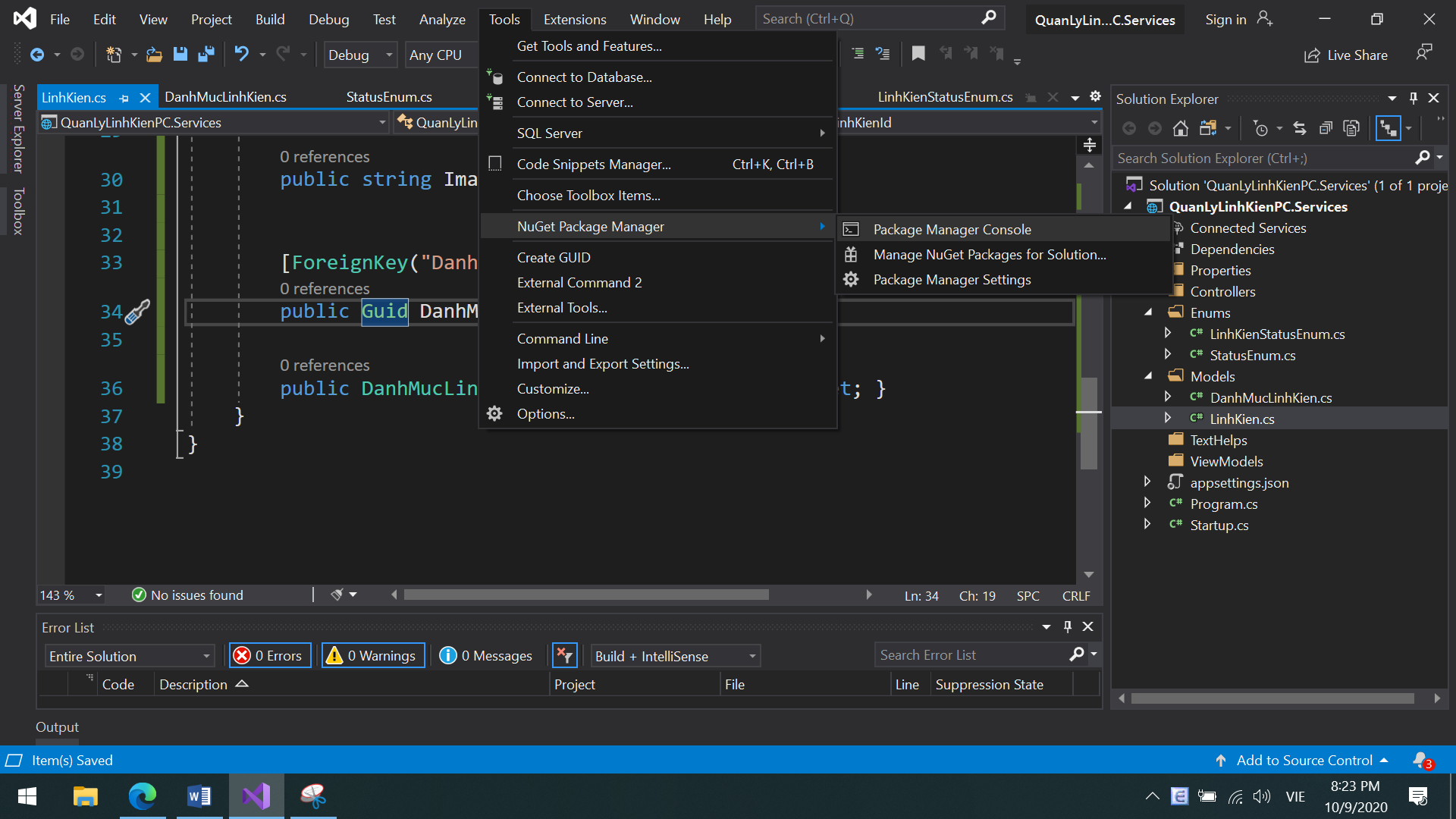
[Required]: Xác định thuộc tính này not null

[Key]: chỉ định khóa chính cho bảng

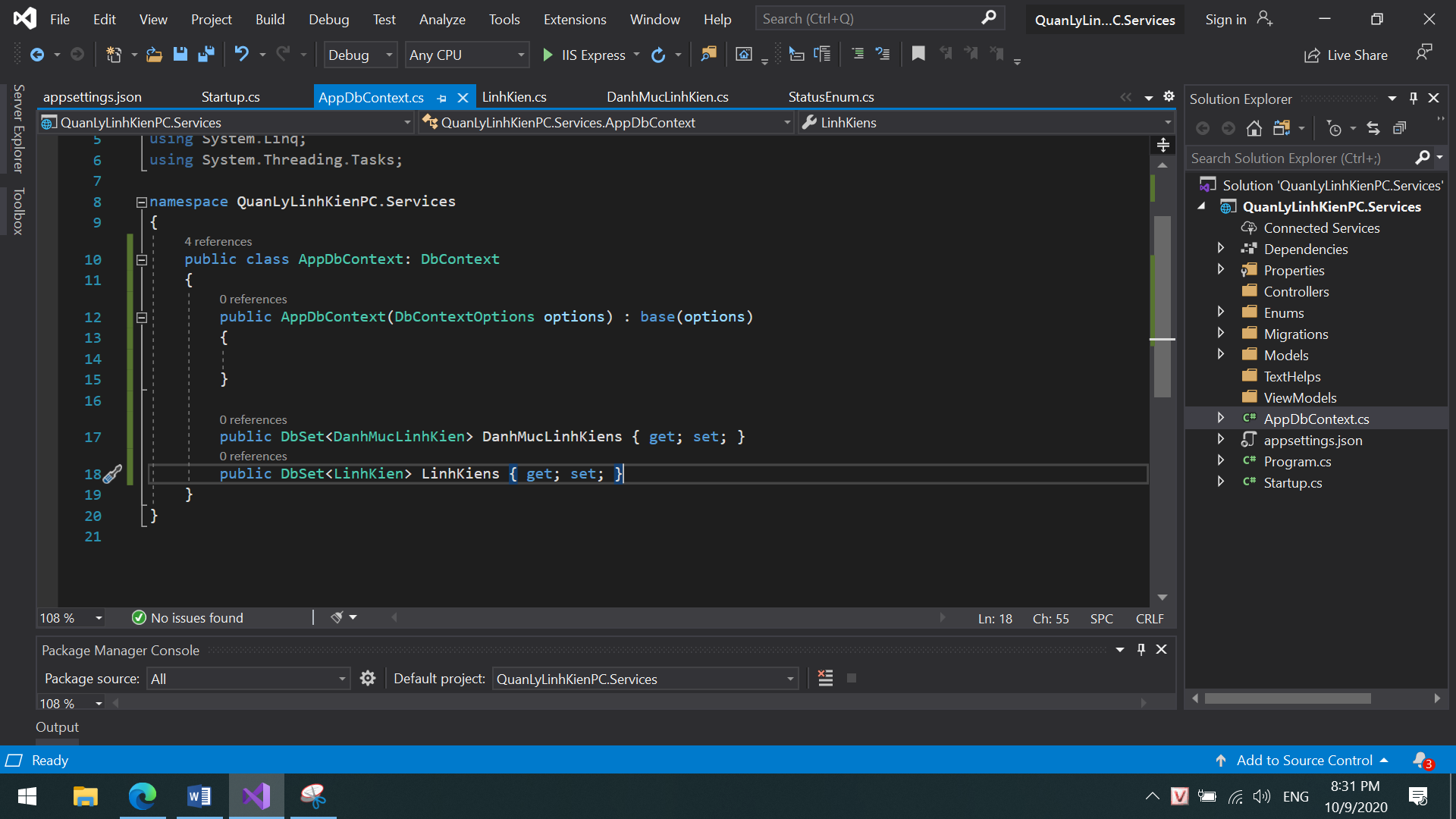
[StringLength(10)]: quy định độ dài cho trường đó(string)

[MaxLength()] : max độ dài

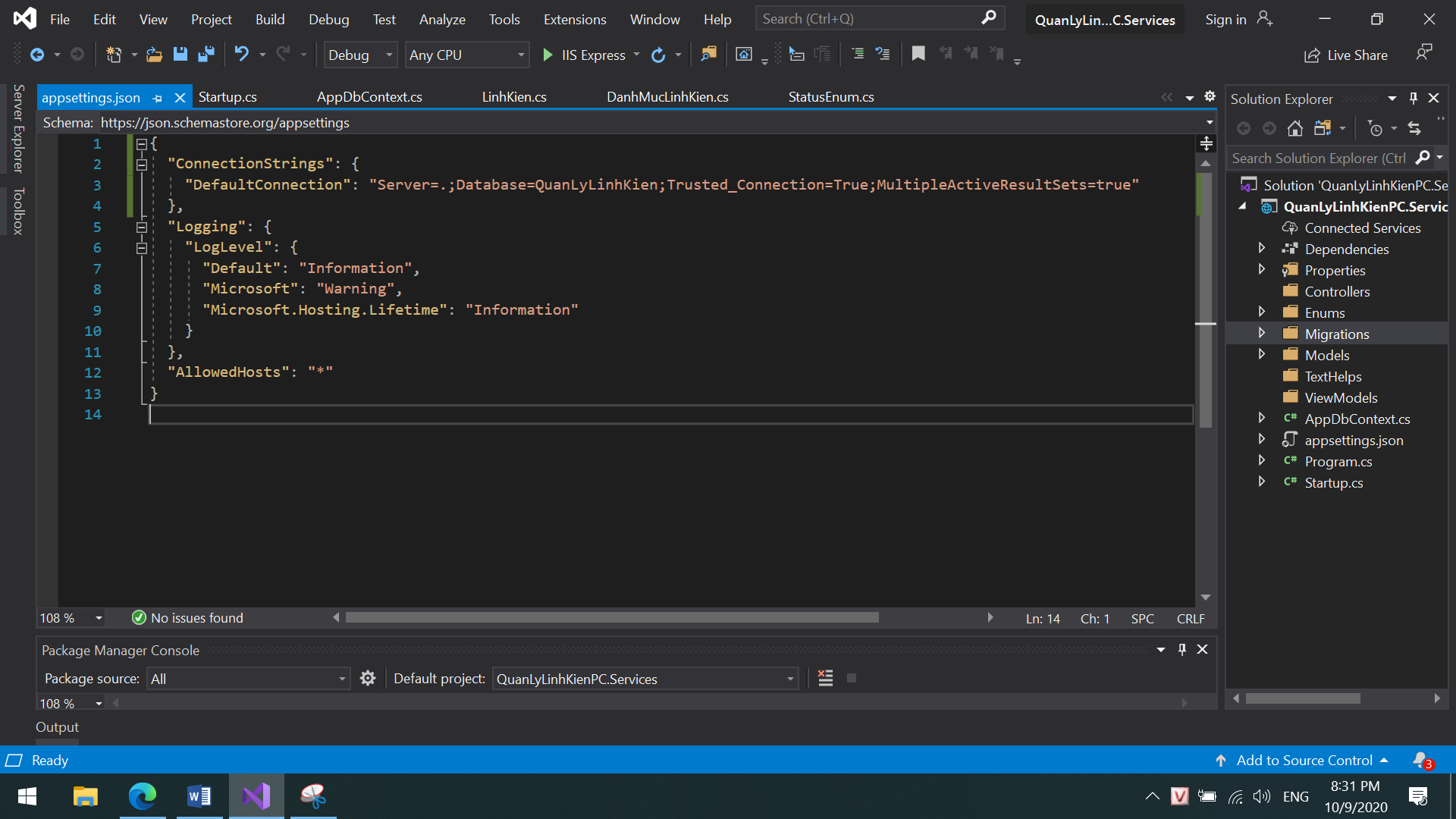
Từ khóa để tìm hiểu: DataAnnotations



Tạo class AppDbContext



Trong file appsetting.json cấu hình chuỗi kết nối



"ConnectionStrings": {

"DefaultConnection": "Server=.;Database=QuanLyLinhKien;Trusted\_Connection=True;MultipleActiveResultSets=true"

},

Trong file startup.cs

public void ConfigureServices(IServiceCollection services)

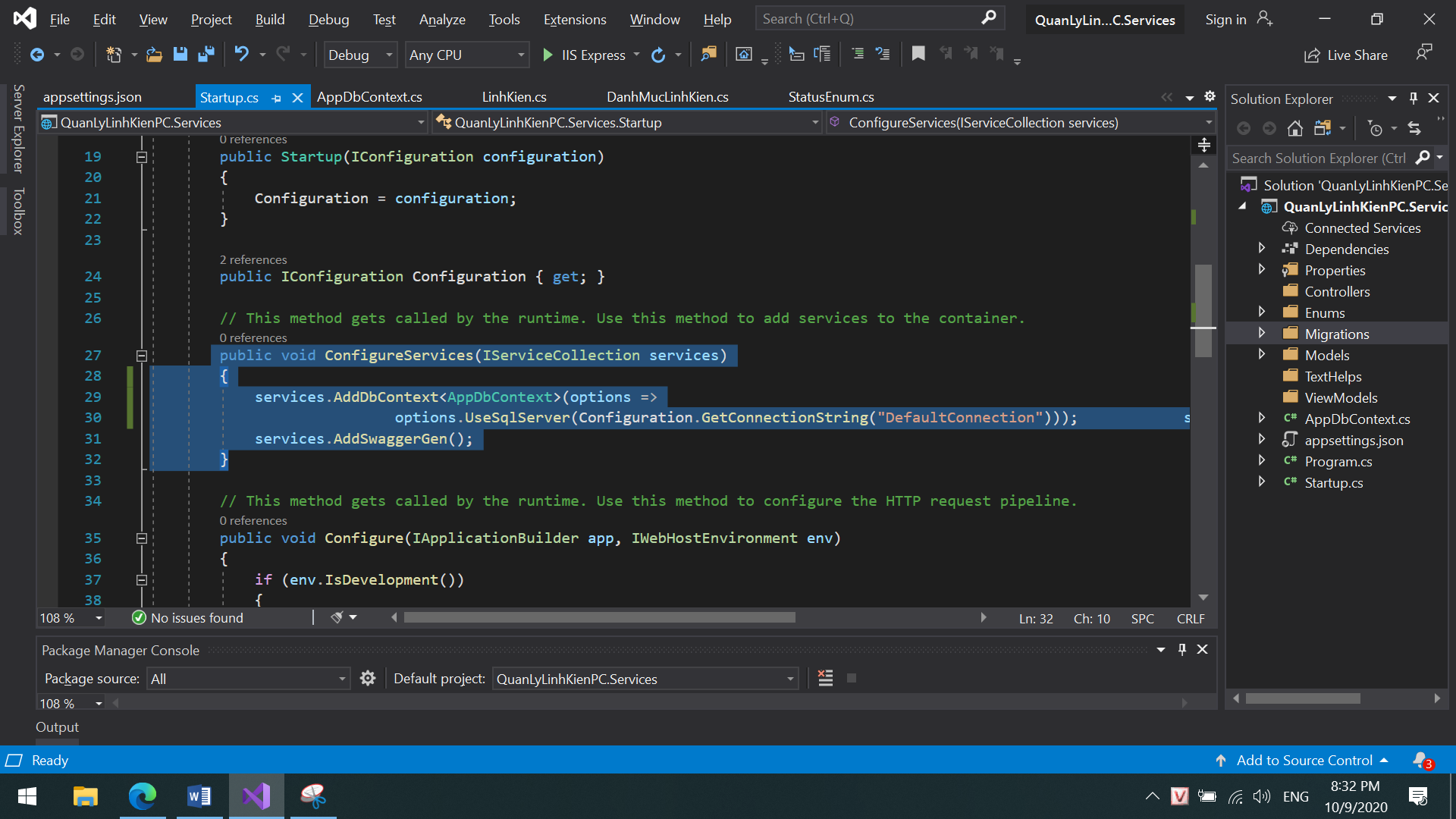
{

services.AddDbContext<AppDbContext>(options =>

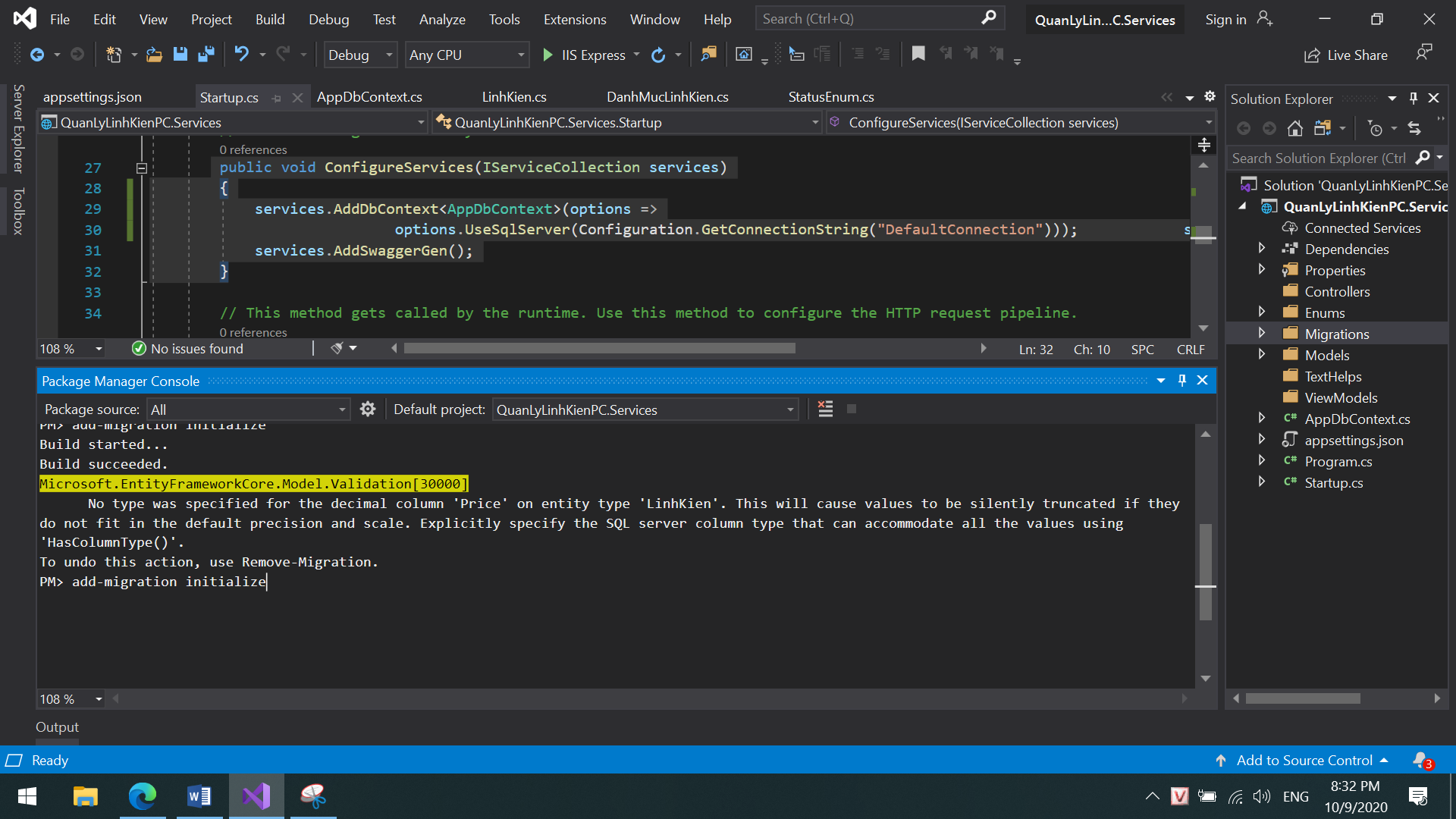
options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection"))); services.AddControllers();

services.AddSwaggerGen();

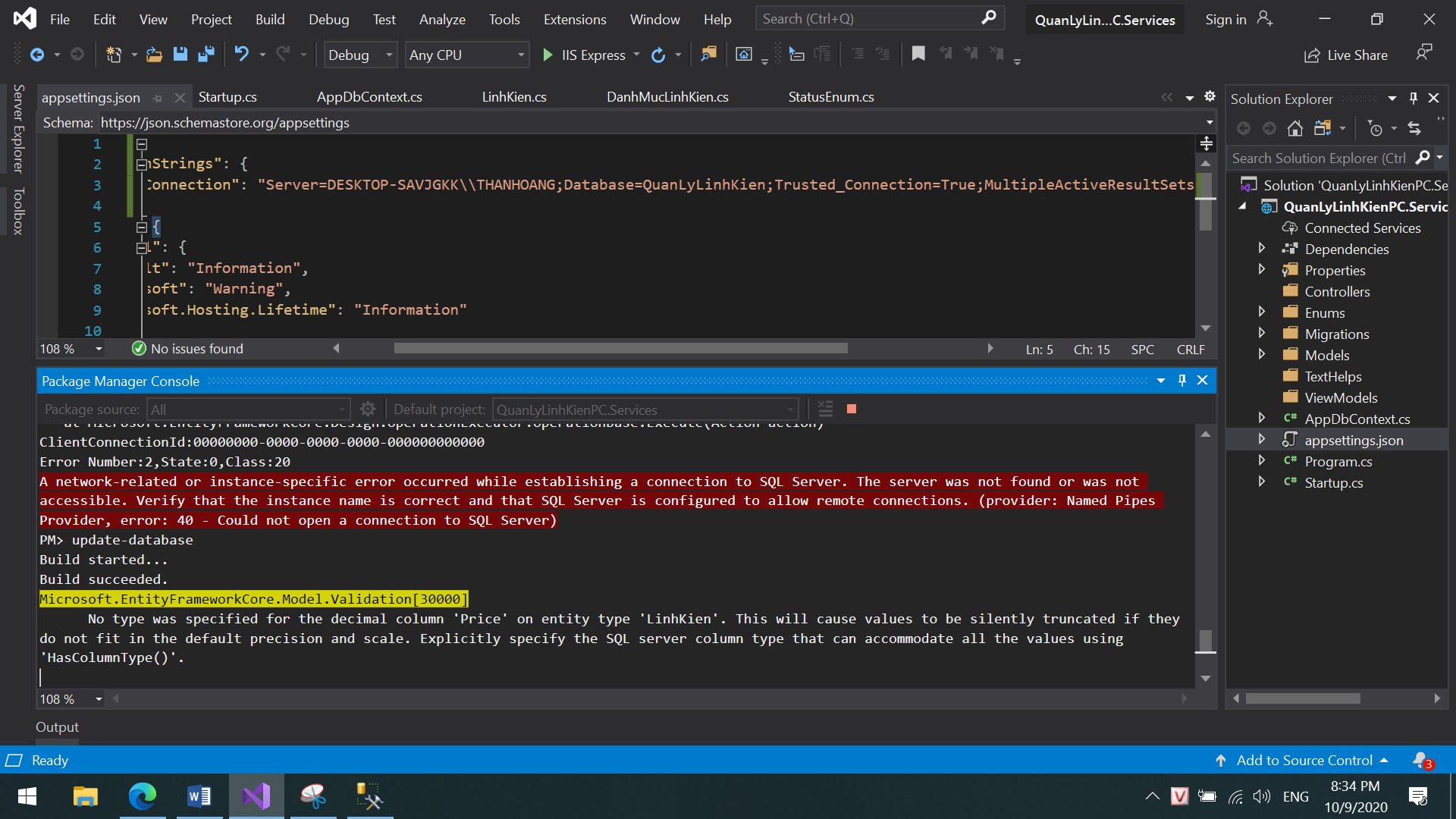
}



Sau đó add migration



add-migration initialize <initialize: là tên migration, có thể đặt tên khác)



Lỗi này thì đổi tên server trong chuỗi kết nối

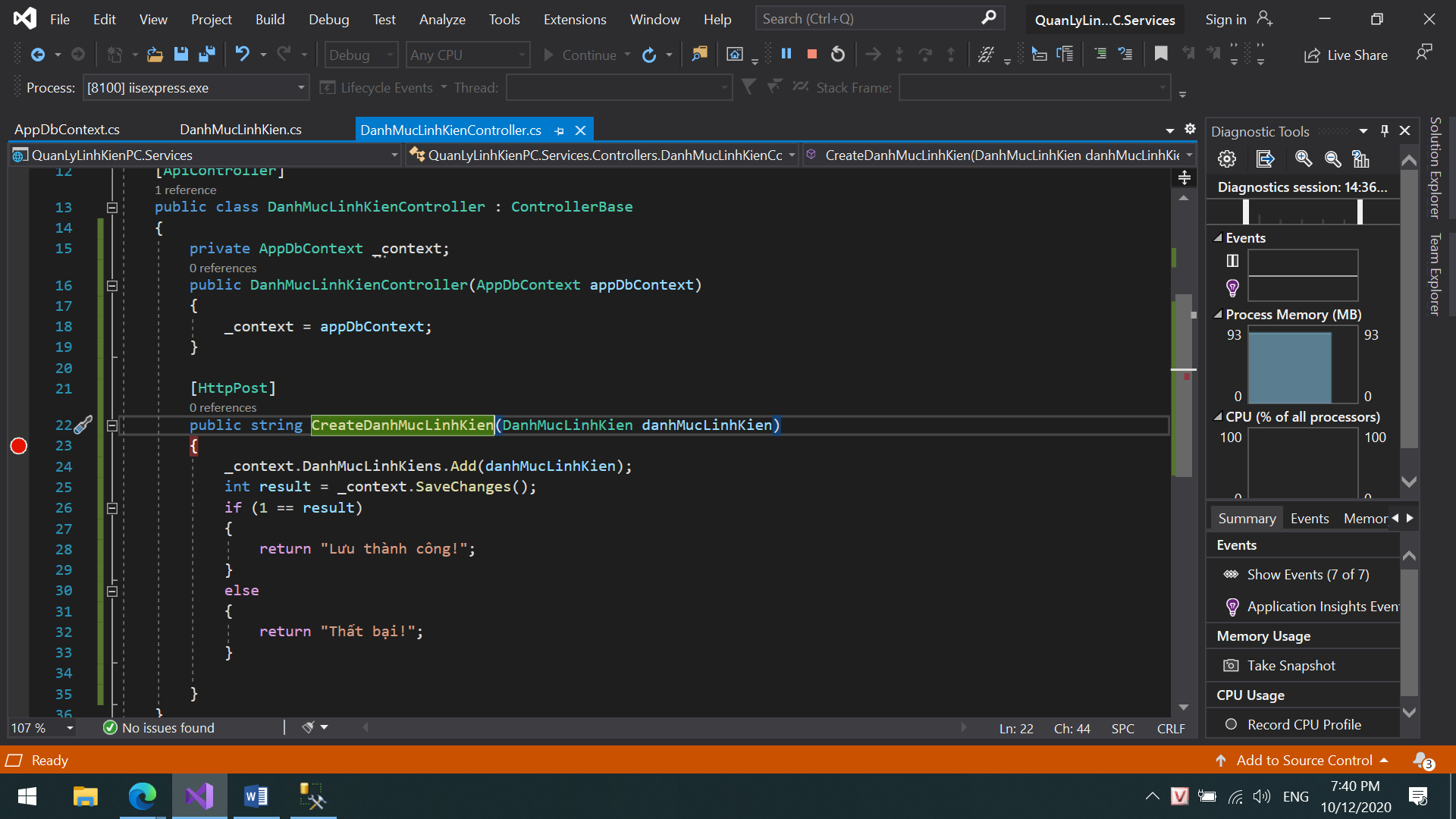
Create API : tương ứng với method => POST

Cú pháp:

[HttpPost]

Public <kiểu trả về> <Tên Hàm> (<tham số truyền vào>: thường là chính class đó){

}



Update API: sửa dữ liệu trong DB

Sử dụng phương thức là: PUT

Tương tự hàm POST

[HttpPut]

public string UpdateDanhMucLinhKien(DanhMucLinhKien danhMucLinhKien)

{

if (string.IsNullOrEmpty(danhMucLinhKien.Id.ToString()))

{

return "ID rỗng => cập nhật làm sao???";

}

var danhMuc = \_context.DanhMucLinhKiens.FirstOrDefault(x => x.Id == danhMucLinhKien.Id);

// = null tức là không tìm thấy, còn != null thì tức là tìm thấy

if (danhMuc == null)

{

return "ID này không tồn tại trong CSDL!";

}

danhMuc.Name = danhMucLinhKien.Name;

danhMuc.Status = danhMucLinhKien.Status;

danhMuc.GroupName = danhMucLinhKien.GroupName;

danhMuc.Code = danhMucLinhKien.Code;

int result = \_context.SaveChanges();

if (1 == result)

{

return "Lưu thành công!";

}

else

{

return "Thất bại!";

}

}

Delete API:

Để xóa bản ghi trong DB thì dùng phương thức

DELETE

[HttpDelete]

public string DeleteDanhMucLinhKien(Guid id)

{

var danhMuc = \_context.DanhMucLinhKiens.FirstOrDefault(danhMuc => danhMuc.Id == id);

if (danhMuc == null)

{

return "Không tìm thấy ID mà xóa!";

}

\_context.DanhMucLinhKiens.Remove(danhMuc);

var result = \_context.SaveChanges();

if (1 == result)

{

return "Lưu thành công!";

}

else

{

return "Thất bại!";

}

}

Find API:

Sử dụng phương thức: GET