# BÀI 3: ASP.NET WEB API VỚI CƠ SỞ DỮ LIỆU SQL SERVER

Trong bài này, WebAPI thực hiện hoạt động CRUD trên cơ sở dữ liệu SQL Server. Web API trong bài này xây dựng sẽ truy xuất dữ liệu từ bảng cơ sở dữ liệu Nhân viên. Sau đó sử dụng Entity Framework để lấy dữ liệu từ cơ sở dữ liệu máy chủ SQL server.

## 3.1. Tạo cơ sở dữ liệu

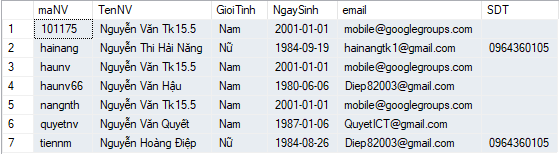
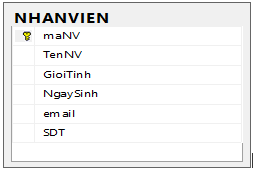
Bài này sử dụng CSDL chứa bảng Nhân viên để tạo dịch vụ API Web bthực hiện hoạt động CRUD bằng cơ sở dữ liệu SQL Server.

Trước tiên

Tạo cơ sở dữ liệu có tên

Sau đó, tạo bảng Nhân viên và điền vào bảng đó với một số dữ liệu thử nghiệm

Bảng NHANHVIEN và dữ liệu như sau



Tham khảo SQL script sau để tạo bảng Nhân viên và chèn các bản ghi dữ liệu vào bảng như sau

|  |
| --- |
| CREATE DATABASE QLNS  USE QLNS  CREATE TABLE NHANVIEN(  maNV varchar(30) PRIMARY KEY,  TenNV nvarchar(50) NOT NULL,  GioiTinh nvarchar(30) default N'Nam',  NgaySinh date,  email varchar(30),  SDT char(30)  )  INSERT INTO NHANVIEN  VALUES ('diepnh',N'Nguyễn Hoàng Ðiệp', N'Nữ', '08/26/1984', 'Diep82003@gmail.com', '0964360105')  INSERT INTO NHANVIEN  VALUES ('nangnth',N'Nguyễn Thi Hải Năng', N'Nữ', '09/19/1984', 'hainangtk1@gmail.com', '0964360105')  INSERT INTO NHANVIEN  VALUES ('haunv',N'Nguyễn Văn Hậu', N'Nam', '06/06/1980', 'Diep82003@gmail.com', '')  INSERT INTO NHANVIEN  VALUES ('quyetnv',N'Nguyễn Văn Quyết', N'Nam', '01/6/1987', 'QuyetICT@gmail.com', '') |

Tạo Procedure store thêm sửa xóa (nếu cần)

|  |
| --- |
| CREATE PROC ThemNHANVIEN (  @maNV varchar(30),  @TenNV nvarchar(50),  @GioiTinh nvarchar(30),  @NgaySinh date,  @email varchar(30),  @SDT char(30))  As  BEGIN  INSERT INTO NHANVIEN  VALUES (@maNV, @TenNV, @GioiTinh, @NgaySinh,@email, @SDT)  END;  EXEC ThemNHANVIEN 'tiennm',N'Nguyễn Minh Tiến', N'Nam', '07/22/1986', 'minhtien@gmail.com', '' |
| CREATE PROC SuaNHANVIEN (  @maNV varchar(30),  @TenNV nvarchar(50),  @GioiTinh nvarchar(30),  @NgaySinh date,  @email varchar(30),  @SDT char(30))  As  BEGIN  UPDATE NHANVIEN  SET TenNV =@TenNV ,  GioiTinh =@GioiTinh ,  NgaySinh =@NgaySinh ,  email =@email ,  SDT=@SDT  WHERE maNV=@maNV  END;  EXEC SuaNHANVIEN 'tiennm',N'Nguyễn Minh Tiến', N'Nam', '07/07/1986', 'minhtien@gmail.com', '0999999999' |
| CREATE PROC XoaNHANVIEN ( @maNV varchar(30))  As  BEGIN  DELETE FROM NHANVIEN  WHERE maNV=@maNV  END;  EXEC XoaNHANVIEN'diepnh' |

Tham khảo thêm tạo WebAPI trên nền Visual Studio 2012

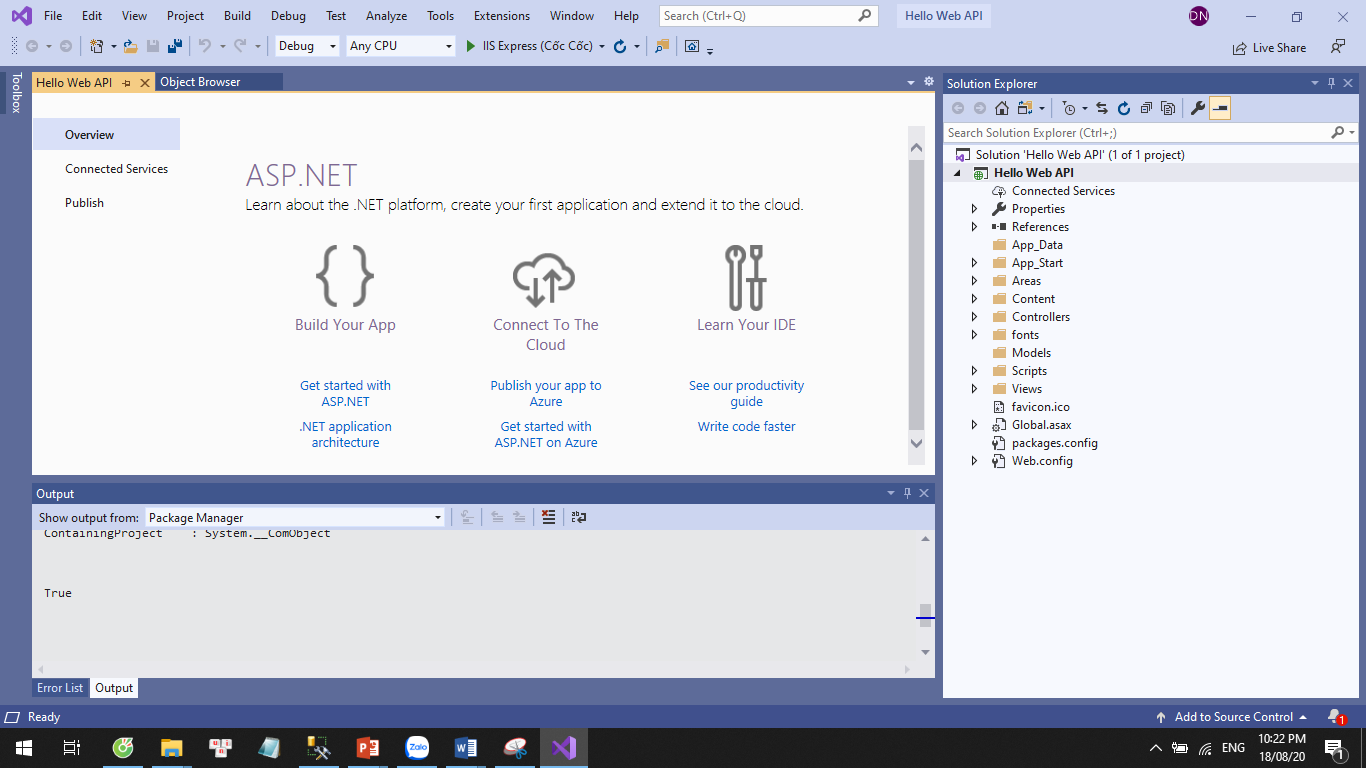
<https://dotnettutorials.net/lesson/web-api-with-sql-server/>

## 3.2. Tạo dự án ASP.NET Web API

##### **Tạo một dự án ASP.NET Web API theo các bước sau**

1. Mở Visual Studio
2. Tạo một project ASP.NET Web Application
3. Đặt tên cho ứng dụng, chọn nơi lưu trữ và FrameWork rồi chọn Create
4. Chọn template Web API

**Dự án ASP.NET Web API** sau khi được tạo sẽ như sau:



(Xem lại chi tiết các bước như phần 1.5.1 Tạo dự án ASP.NET Web API sử dụng Visual Studio)

Có thể vào Nuget Package để cài đặt các package cần thiết

## 3.3. Kết nối với cơ sở dữ liệu sử dụng ADO.NET Entity Framework

### 3.3.1 Thêm mô hình dữ liệu thực thể ADO.NET để truy xuất dữ liệu từ cơ sở dữ liệu

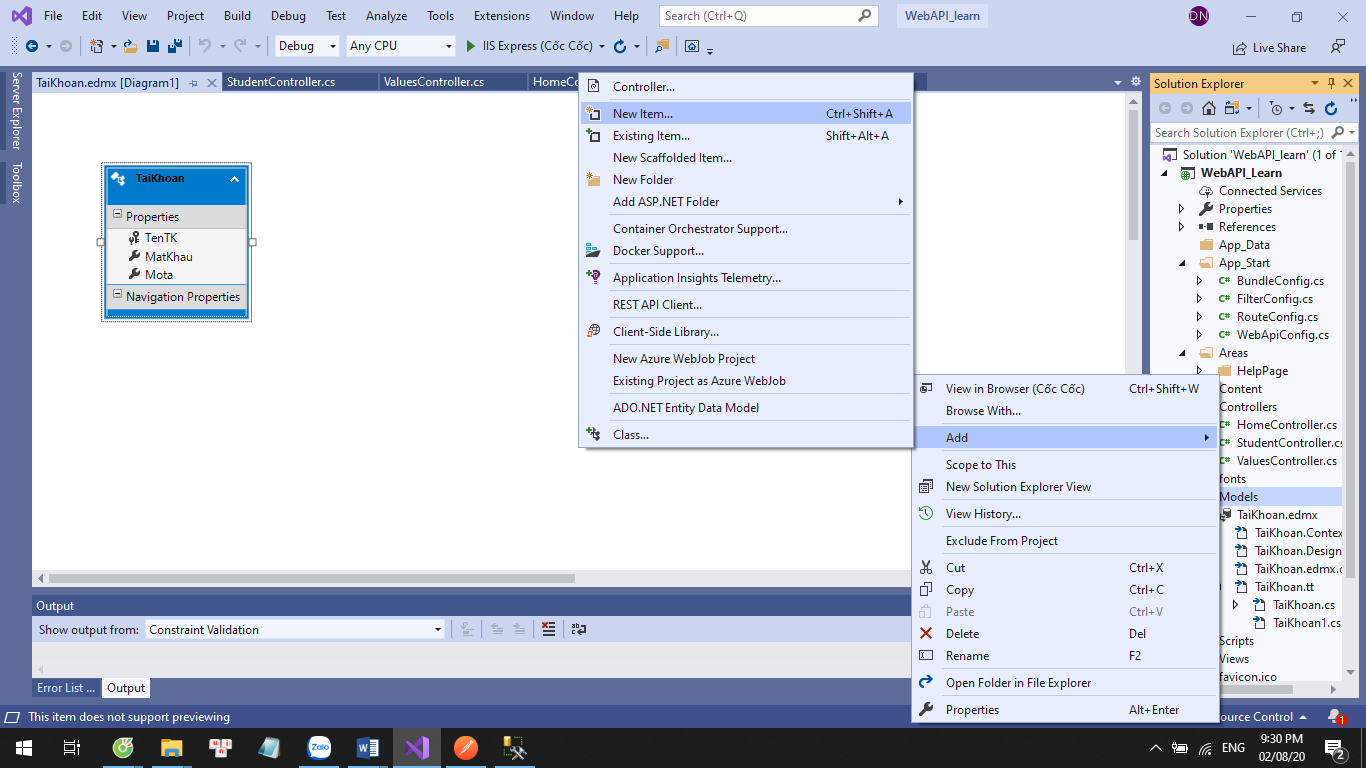
Giả sử

servername là DESKTOP-KL7N773\SQLEXPRESS2014

Tên CSDL là QLNS

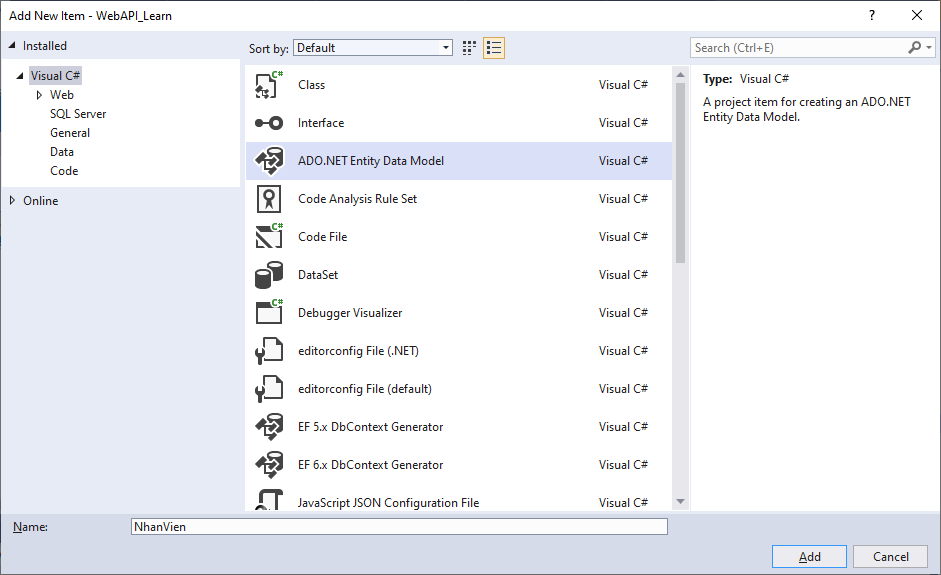
Thì kết nối CSDL sẽ có connection string như sau: "data source=DESKTOP-KL7N773\SQLEXPRESS2014;initial catalog=QLNS;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"

#### Bước 1: Trước tiên, để tạo các model, nhấn chuột phải vào Models và chọn Add/New item

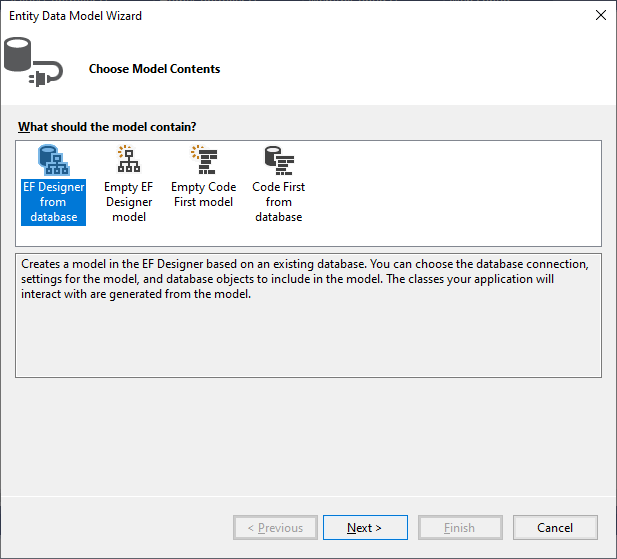


Bước 2:

* Chọn ADO.NET Entity Data Model
* Nhập tên cho class
* Rồi chọn Add

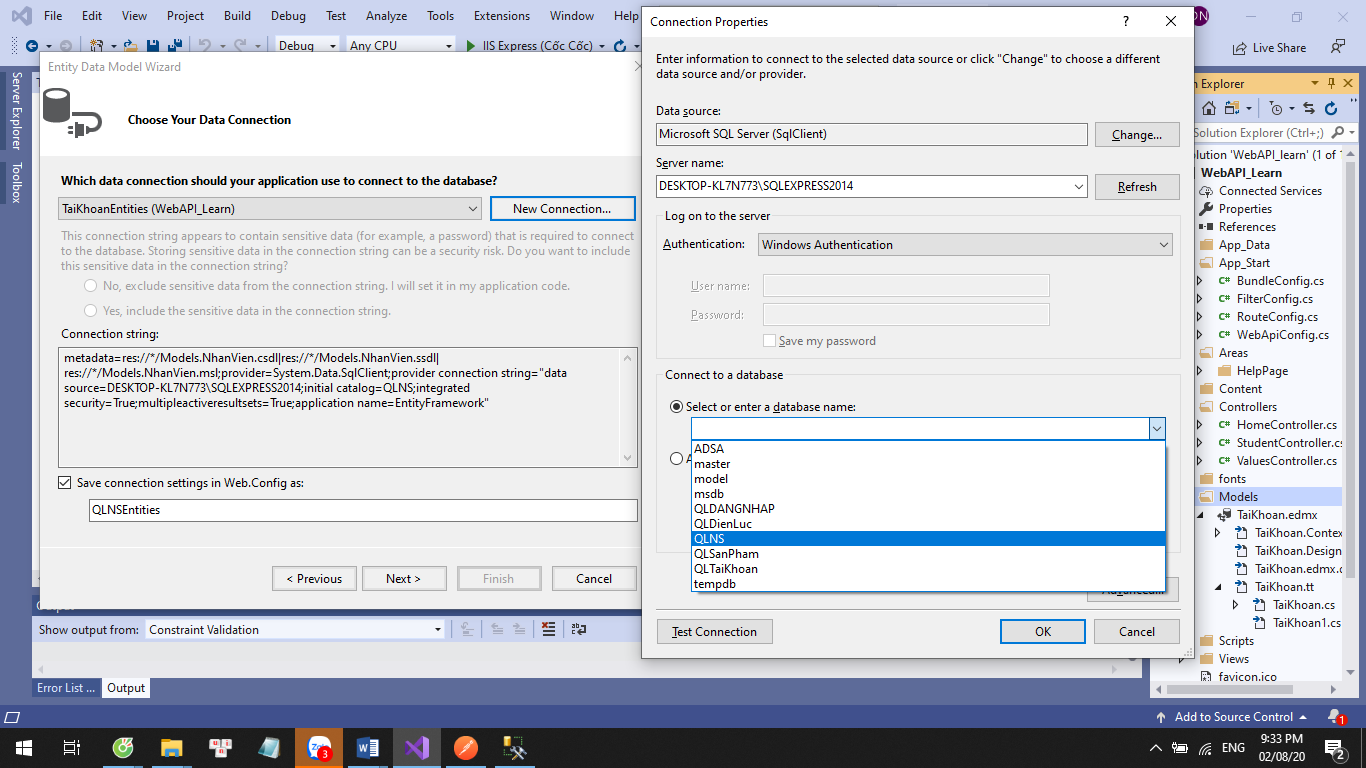


Muốn tạo một mô hình trong EF khi đã có thiết kế CSDL, có định nghĩa các quan hệ và mối liên quan giữa chúng, người ta thường lập trình theo hướng Data first thì chọn EF Designer from database (hình sau)



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.

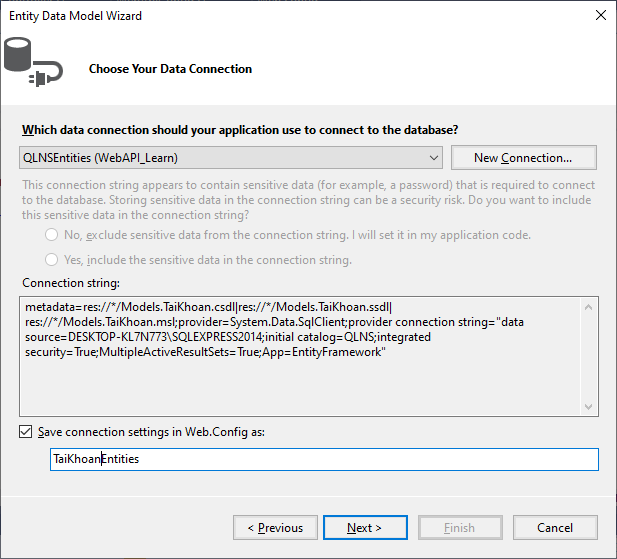
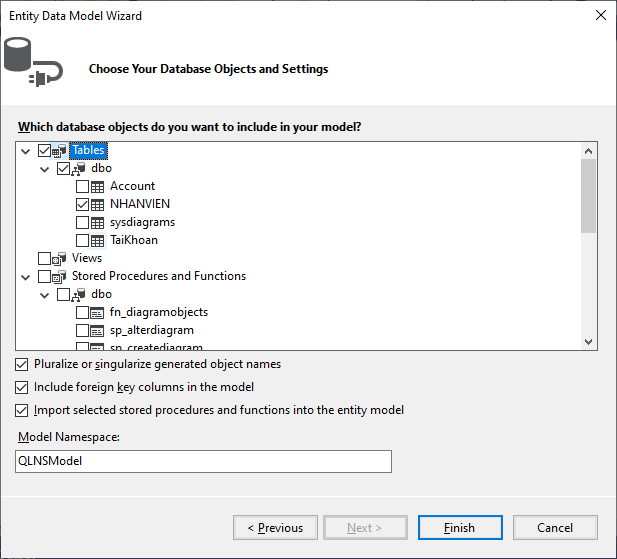


Nhấn Test Connection để kiểm tra xem có kêt nối được tới CSDL hay không (successfull là thành công)

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

Chú ý! Trong trường hợp nhiêu table trong CSDL, ta có thể chọn từng model tương ứng table hoặc nhiều bảng tables cần thiết trong 1 lần.

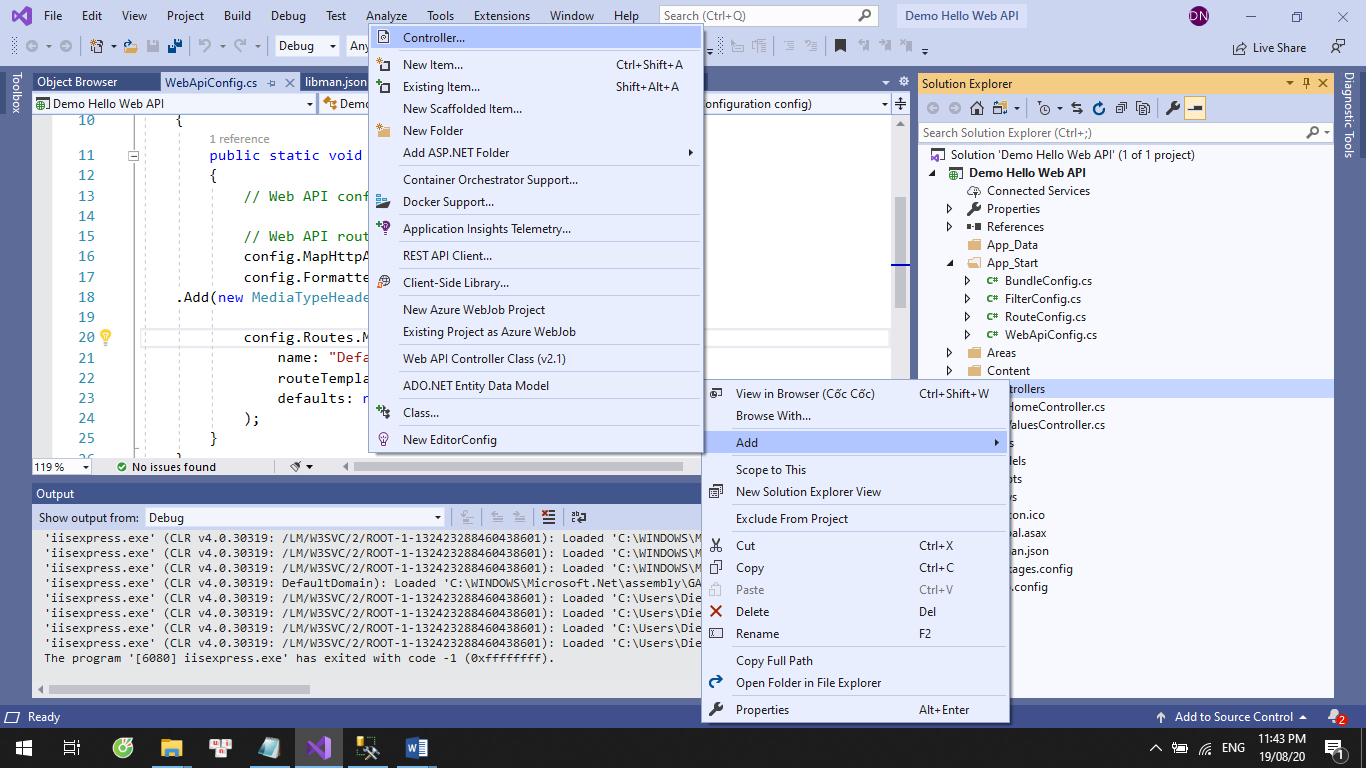
Chọn các store proc và function (nếu cần)



## 3.4. Viết Web API Controller

Bước 1: Tạo các Controller

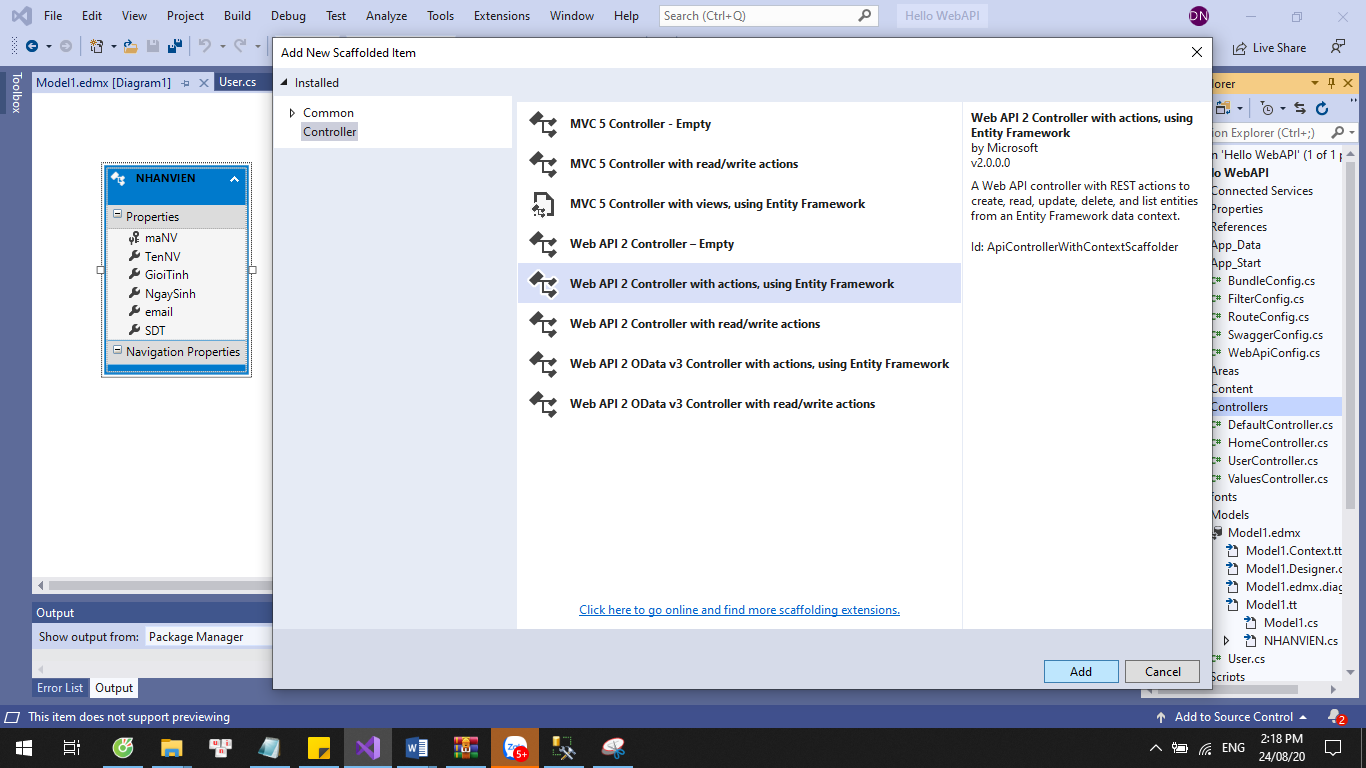
Nhấp chuột phải vào thư mục Controllers và chọn Add/Controller…



Trong đó! Có thể chọn loại controller khác nhau.

Chọn **Web API 2 Controller Empty thì**tự viết các phương thức từ đầu.

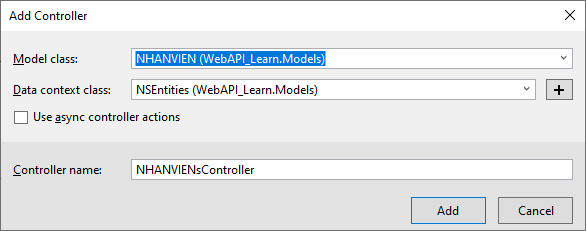
Tuy nhiên nếu đã có CSDL, có sẵn data, đã có model tương ứng data thì nên chọn **Web API 2 Controller with action và using entity Framewwork** để được hỗ trợ sinh code tự động tương ứng với EF cho Controller



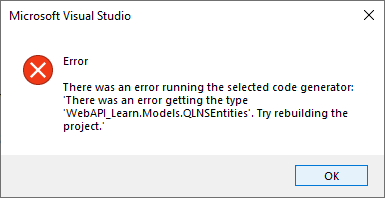
Chọn Add rồi đặt tên cho Controller. Sau đó viết code cho Controller

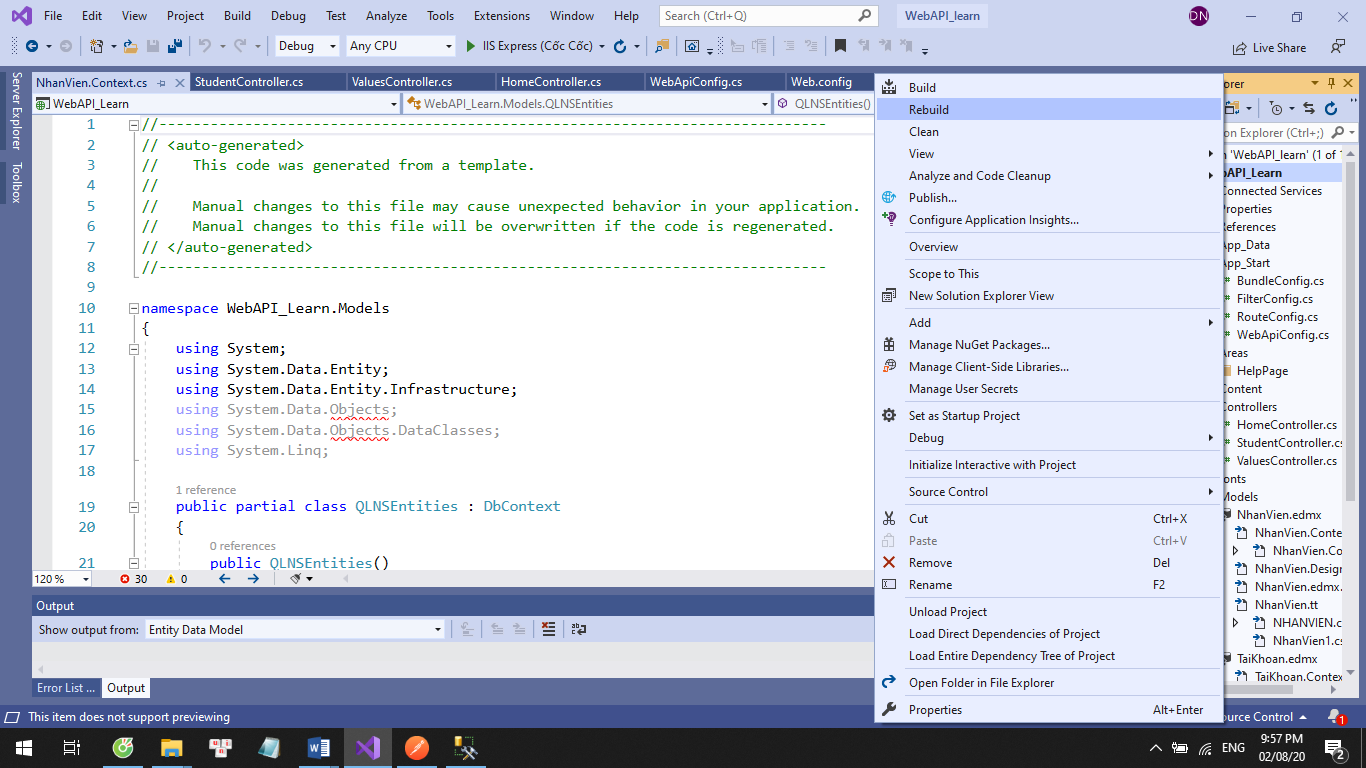
Lần lượt thêm các controller, mỗi table một model một controller,

Chú ý! Chọn đúng model class, Datacontext sẽ tự sinh tương ứng model class chọn



Nếu sau khi sinh model, chưa rebuild App thì sẽ báo lỗi này=> đóng lại, rebuild lại ứng dụng, rồi tạo lại Controller



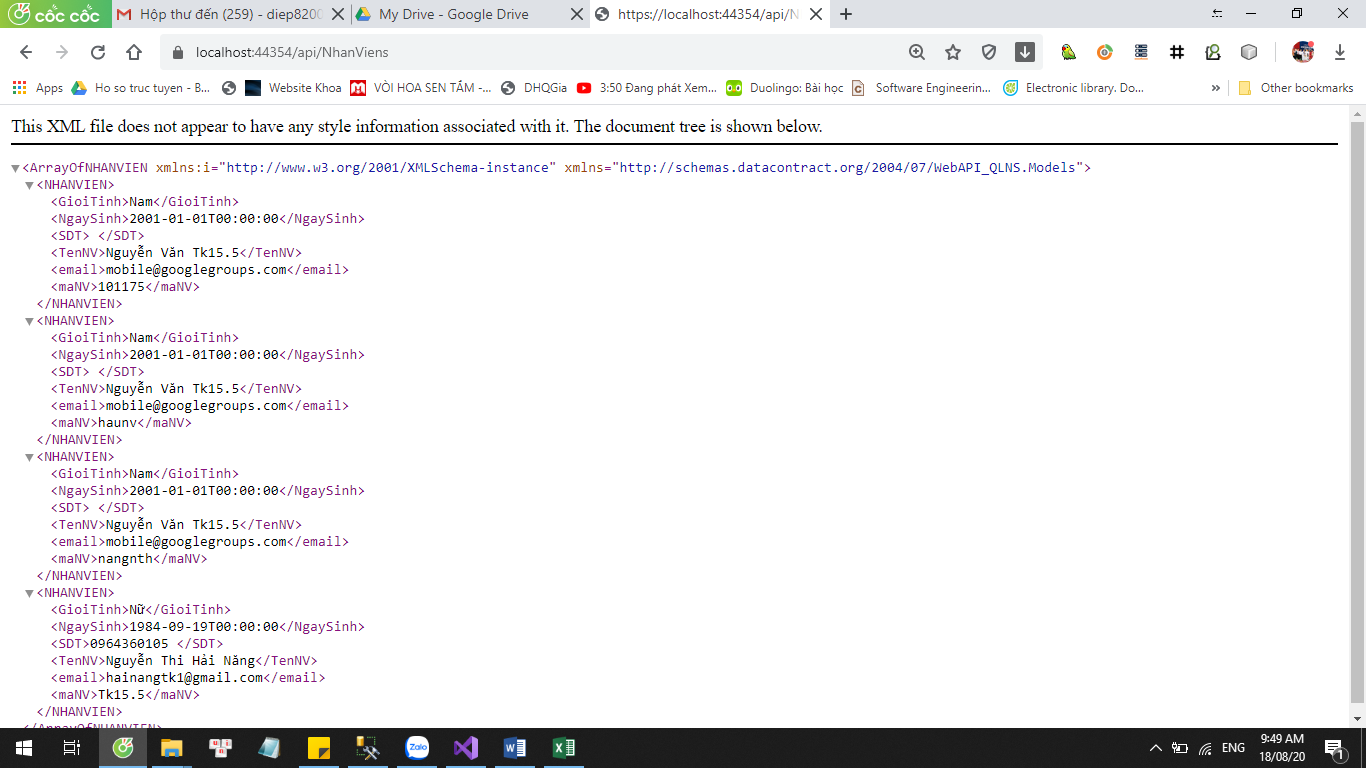


Controllers\NHANVIENsController.cs như sau:

|  |
| --- |
| 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\_Learn.Models;  namespace WebAPI\_Learn.Controllers  {  public class NHANVIENsController : ApiController  {  private NSEntities db = new NSEntities();  // GET: api/NHANVIENs  public IQueryable<NHANVIEN> GetNHANVIENs()  {  return db.NHANVIENs;  }  // GET: api/NHANVIENs/5  [ResponseType(typeof(NHANVIEN))]  public IHttpActionResult GetNHANVIEN(string id)  {  NHANVIEN nHANVIEN = db.NHANVIENs.Find(id);  if (nHANVIEN == null)  {  return NotFound();  }  return Ok(nHANVIEN);  }  // PUT: api/NHANVIENs/5  //[Route("api/NHANVIENs/{id}")]  [ResponseType(typeof(void))]  [HttpPut]  public IHttpActionResult PutNHANVIEN(string id, [FromBody] NHANVIEN nHANVIEN)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  if (id != nHANVIEN.maNV)  {  return BadRequest();  }  db.Entry(nHANVIEN).State = EntityState.Modified;  try  {  db.SaveChanges();  }  catch (DbUpdateConcurrencyException)  {  if (!NHANVIENExists(id))  {  return NotFound();  }  else  {  throw;  }  }  return StatusCode(HttpStatusCode.NoContent);  }  // POST: api/NHANVIENs  //[Route("api/NHANVIENs")]  [ResponseType(typeof(NHANVIEN))]  [HttpPost]  public IHttpActionResult PostNHANVIEN([FromBody] NHANVIEN nHANVIEN)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  db.NHANVIENs.Add(nHANVIEN);  try  {  db.SaveChanges();  }  catch (DbUpdateException)  {  if (NHANVIENExists(nHANVIEN.maNV))  {  return Conflict();  }  else  {  throw;  }  }  return CreatedAtRoute("DefaultApi", new { id = nHANVIEN.maNV }, nHANVIEN);  }  // DELETE: api/NHANVIENs/5  [ResponseType(typeof(NHANVIEN))]  public IHttpActionResult DeleteNHANVIEN(string id)  {  NHANVIEN nHANVIEN = db.NHANVIENs.Find(id);  if (nHANVIEN == null)  {  return NotFound();  }  db.NHANVIENs.Remove(nHANVIEN);  db.SaveChanges();  return Ok(nHANVIEN);  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  private bool NHANVIENExists(string id)  {  return db.NHANVIENs.Count(e => e.maNV == id) > 0;  }  }  } |

Viết mới hoặc sửa lại code để có Controller đảm bảo các chức năng như thiết kế

## Chạy và kiểm tra



url

<https://localhost:44302/api/Nhanviens>

<https://localhost:44302/api/Nhanviens/haunv>

### Cấu hình lại Content-Type thành JsonFormatter (trên App\_Start / WebApiConfig.cs  của wepAPI)

I just add the following in App\_Start / WebApiConfig.cs class in my MVC **Web API** project.

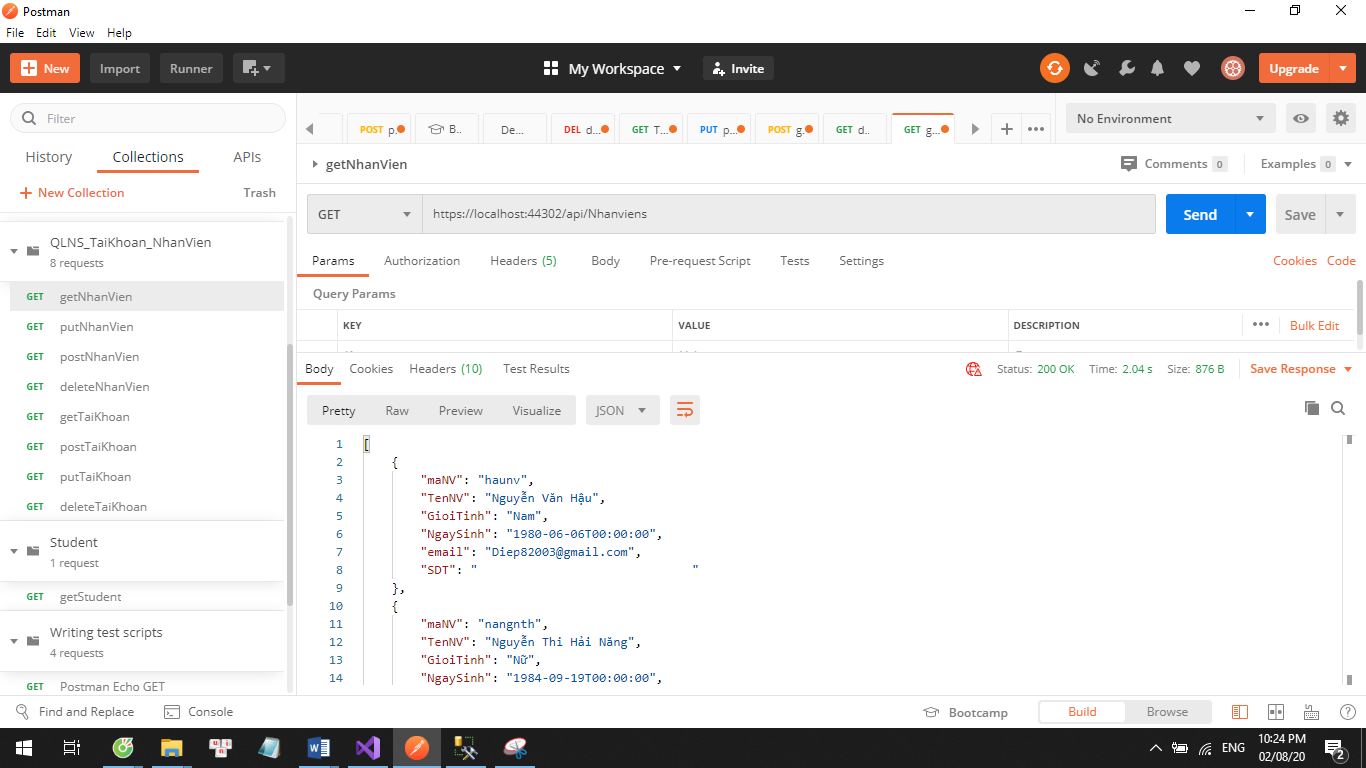
config.Formatters.JsonFormatter.SupportedMediaTypes

.Add(new MediaTypeHeaderValue("text/html") );

Và Alt+enter để import using System.Net.Http.Headers;

## Test với postman

#### httpget



#### Httpdelete xóa NhanVien

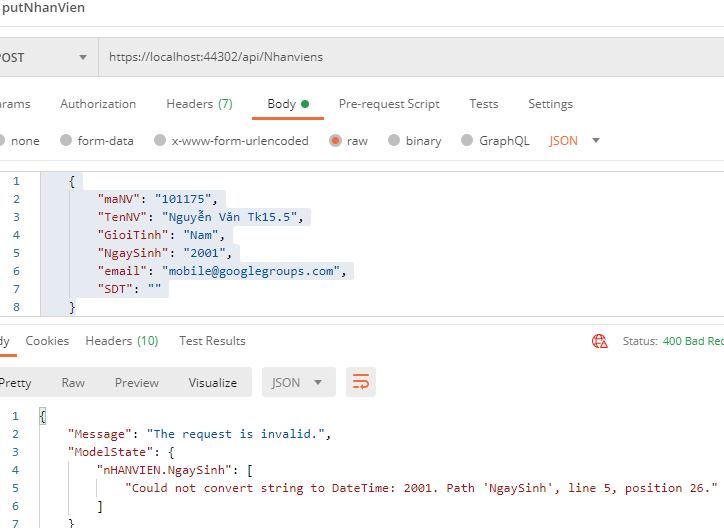
status báo 200 ok là chạy thành công

#### httppost thêm mới NhanVien

Chọn body và dữ liệu dạng form-data với key là key của json obj hoặc chọn body và dữ liệu dạng raw

Giả sử ta nhập dữ liệu body và dữ liệu dạng raw như sau:

|  |  |
| --- | --- |
| Lỗi format datetime | ok |
| {  "maNV": "101175",  "TenNV": "Nguyễn Văn Tk15.5",  "GioiTinh": "Nam",  "NgaySinh": "2001",  "email": "mobile@googlegroups.com",  "SDT": ""  } | {  "maNV": "101175",  "TenNV": "Nguyễn Văn Tk15.5",  "GioiTinh": "Nam",  "NgaySinh": "2001-01-01T00:00",  "email": "mobile@googlegroups.com",  "SDT": ""  } |



Nếu chưa chọn kiểu dữ liệu truyền thích hợp với dữ liệu => Ứng dụng báo lỗi 415 únsupport Media Type

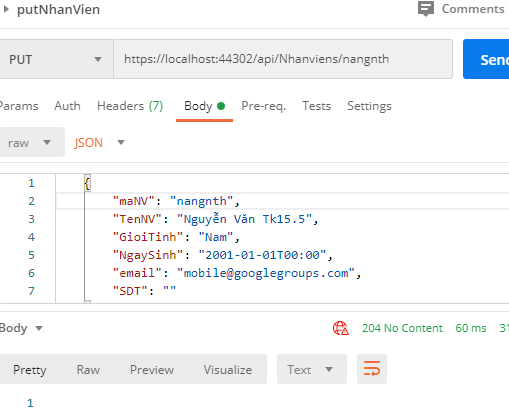
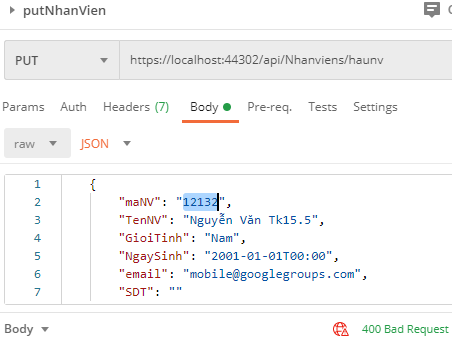
* Chọn lại kiểu dữ liệu truyền vào
* Ở VD trên, chọn dữ liệu dạng json
* Note kiểu dữ liệu nào chọn kiểu đó

### Ví dụ Httpput update

Giả sử ta nhập dữ liệu body và dữ liệu dạng raw như sau:

|  |
| --- |
| {          "maNV": "haunv",          "TenNV": "Nguyễn Văn Tk15.5",          "GioiTinh": "Nam",          "NgaySinh": "2001-01-01T00:00",          "email": "mobile@googlegroups.com",          "SDT": ""      } |

Kết quả sau khi update với 2 ví dụ trên



Chú ý maNV trong bodyRequest và maNV truyền vào phải khớp nhau. Nếu ko khớp bị lỗi 400 bad request

Contents

[BÀI 3: ASP.NET WEB API VỚI CƠ SỞ DỮ LIỆU SQL SERVER 1](#_Toc49292077)

[3.1. Tạo cơ sở dữ liệu 1](#_Toc49292078)

[3.2. Tạo dự án ASP.NET Web API 2](#_Toc49292079)

[3.3. Kết nối với cơ sở dữ liệu sử dụng ADO.NET Entity Framework 3](#_Toc49292080)

[3.3.1 Thêm mô hình dữ liệu thực thể ADO.NET để truy xuất dữ liệu từ cơ sở dữ liệu 3](#_Toc49292081)

[3.4. Viết Web API Controller 6](#_Toc49292082)

[Chạy và kiểm tra 11](#_Toc49292083)

[Cấu hình lại Content-Type thành JsonFormatter (trên App\_Start / WebApiConfig.cs  của wepAPI) 11](#_Toc49292084)

[Test với postman 12](#_Toc49292085)

[Ví dụ Httpput update 13](#_Toc49292086)