# BÀI 6: THỰC HÀNH 02 –XÂY DỰNG WEB API VỚI CRUD 1

***Mục tiêu***

* Trình bày được khái niệm cơ bản về WebAPI
* 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 đơn giản 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 Tạo CSDL có thêm bảng (NhanVien và TaiKhoan

Bước 2: Tạo dự án ASP.NET Web API

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

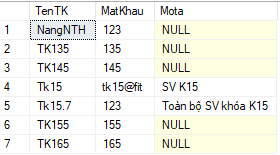
Bước 4: Viết Web API Controller

Bước 5: Chạy và kiểm tra với công cụ

**Yêu cầu:** Xây dựng ASP.NET WEB API với cơ sở dữ liệucó bảng TaiKhoan (TenTK, MatKhau, Mota) với CRUD. 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 và bảng TaiKhoan như sau



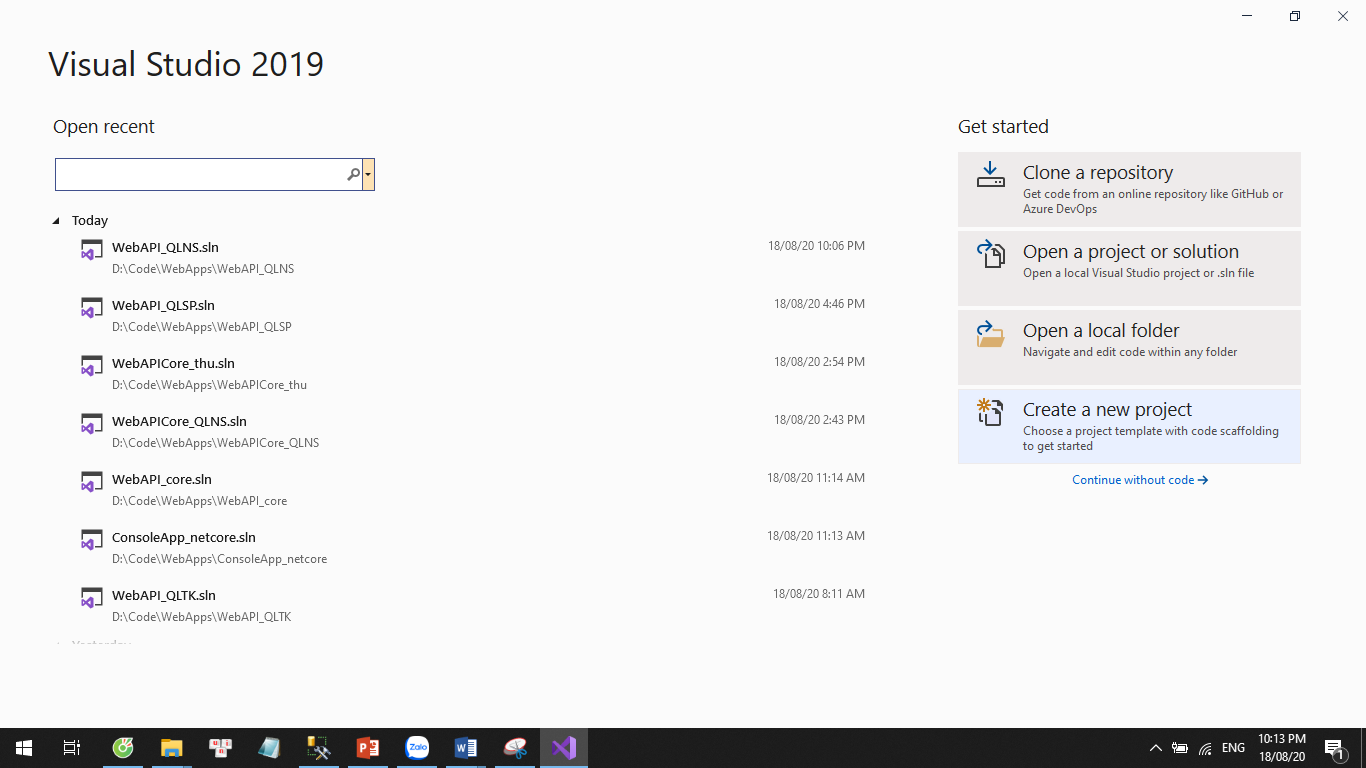
|  |
| --- |
| CREATE TABLE TaiKhoan(  TenTK varchar(30) PRIMARY KEY,  MatKhau varchar(30),  Mota nvarchar(100) default ''  )  INSERT INTO TaiKhoan (TenTK,MatKhau)  VALUES ('diepnh',N'diepNH'), ('TK135','135'), ('TK145','145'), ('TK155','155'), (  'TK165','165'),('TK125','tk125') |

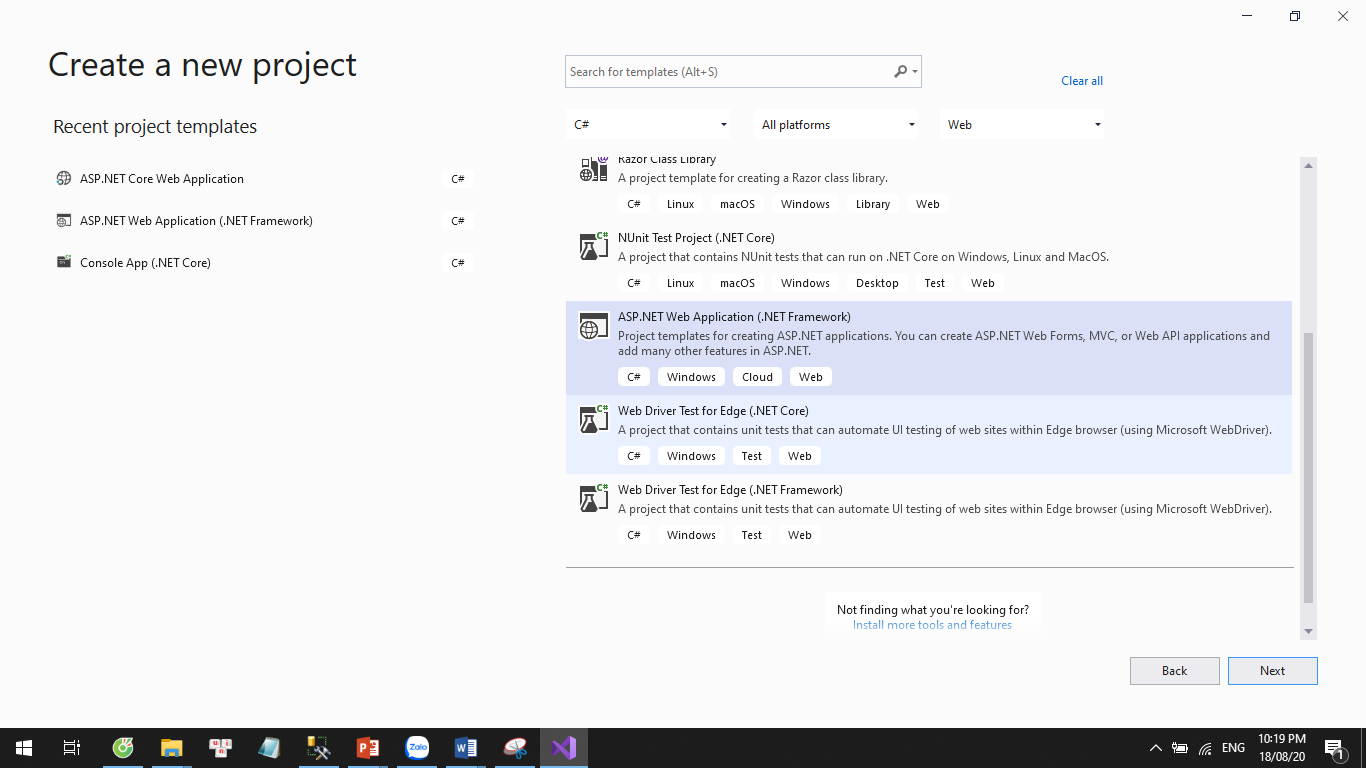
Viết Procedure store them/sửa/xóa nếu cần

|  |
| --- |
| CREATE PROC ThemTaiKhoan (@ten char(30),@mk varchar(30) )  As  BEGIN  INSERT INTO TaiKhoan (TenTK,MatKhau)  VALUES (@ten,@mk)  END;  --GO ThemTaiKhoan 'NangNTH','Hải Năng'--lỗi ko run dc  --DROP PROC ThemTaiKhoan  EXEC ThemTaiKhoan 'NangNTH','Hải Năng' |
| CREATE PROC SuaTaiKhoan (@ten char(30),@mk varchar(30) )  As  BEGIN  UPDATE TaiKhoan SET MatKhau=@mk  WHERE TenTK=@ten  END;  EXEC SuaTaiKhoan 'NangNTH','123' |
| CREATE PROC XoaTaiKhoan (@ten char(30) )  As  BEGIN  DELETE FROM TaiKhoan  WHERE TenTK=@ten  END;  EXEC XoaTaiKhoan 'TK125' |

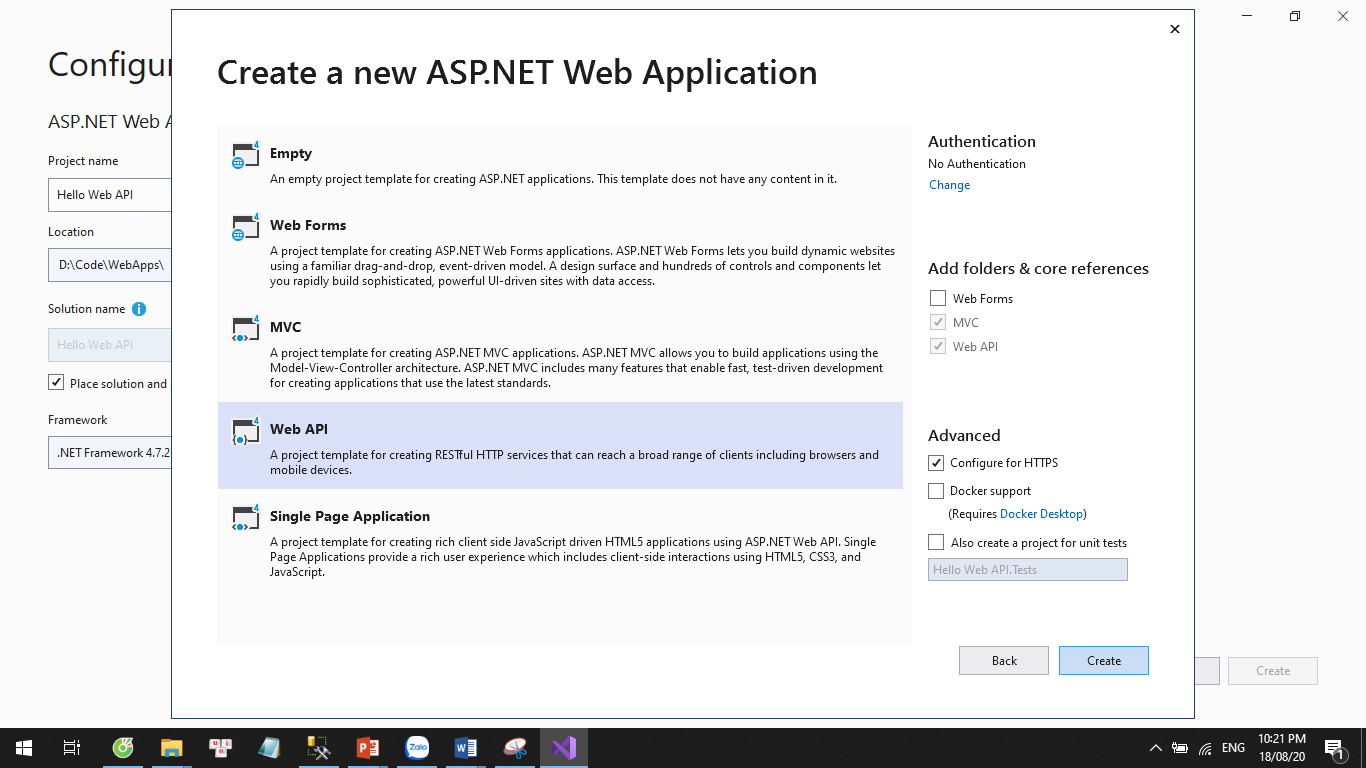
#### Bước 2: Tạo dự án ASP.NET Web API

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

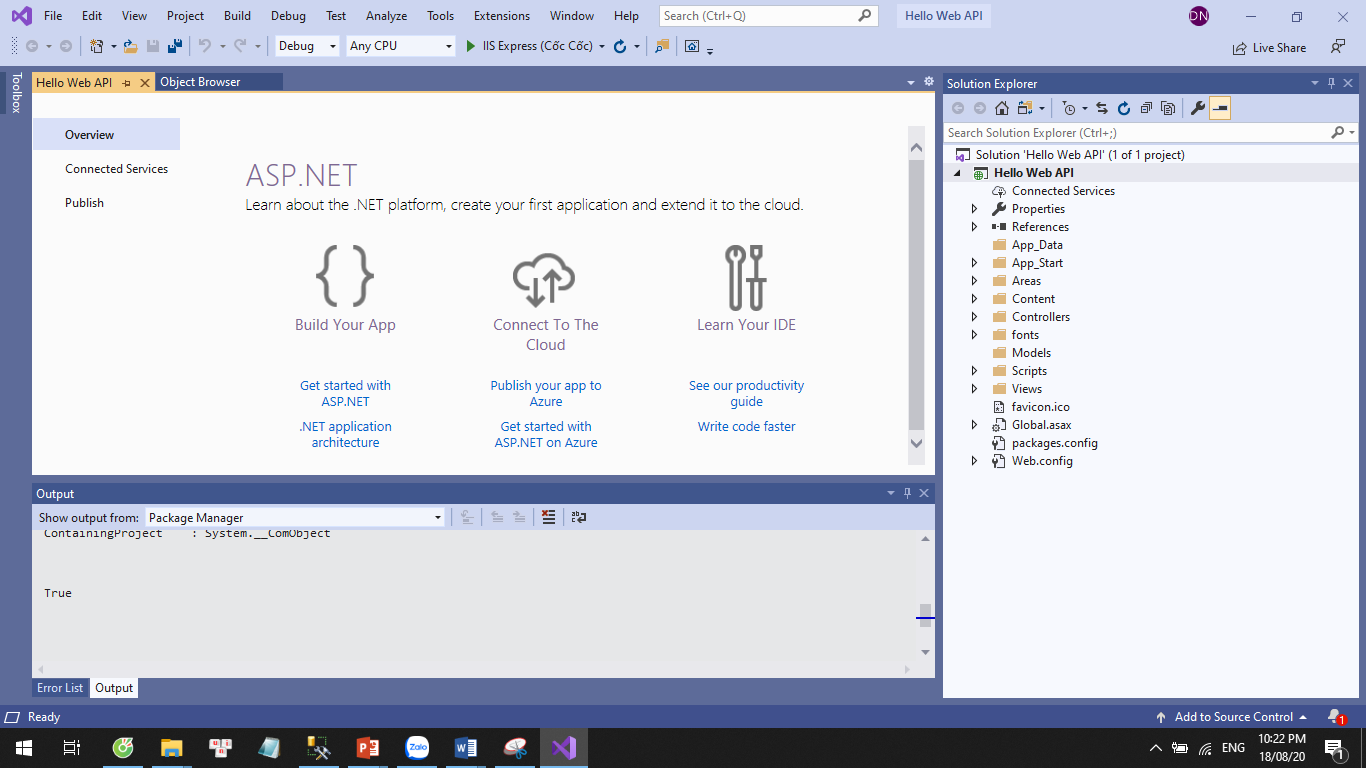




Đặt tên cho ứng dụng, chọn nơi lưu trữ và FrameWork rồi chọn Create

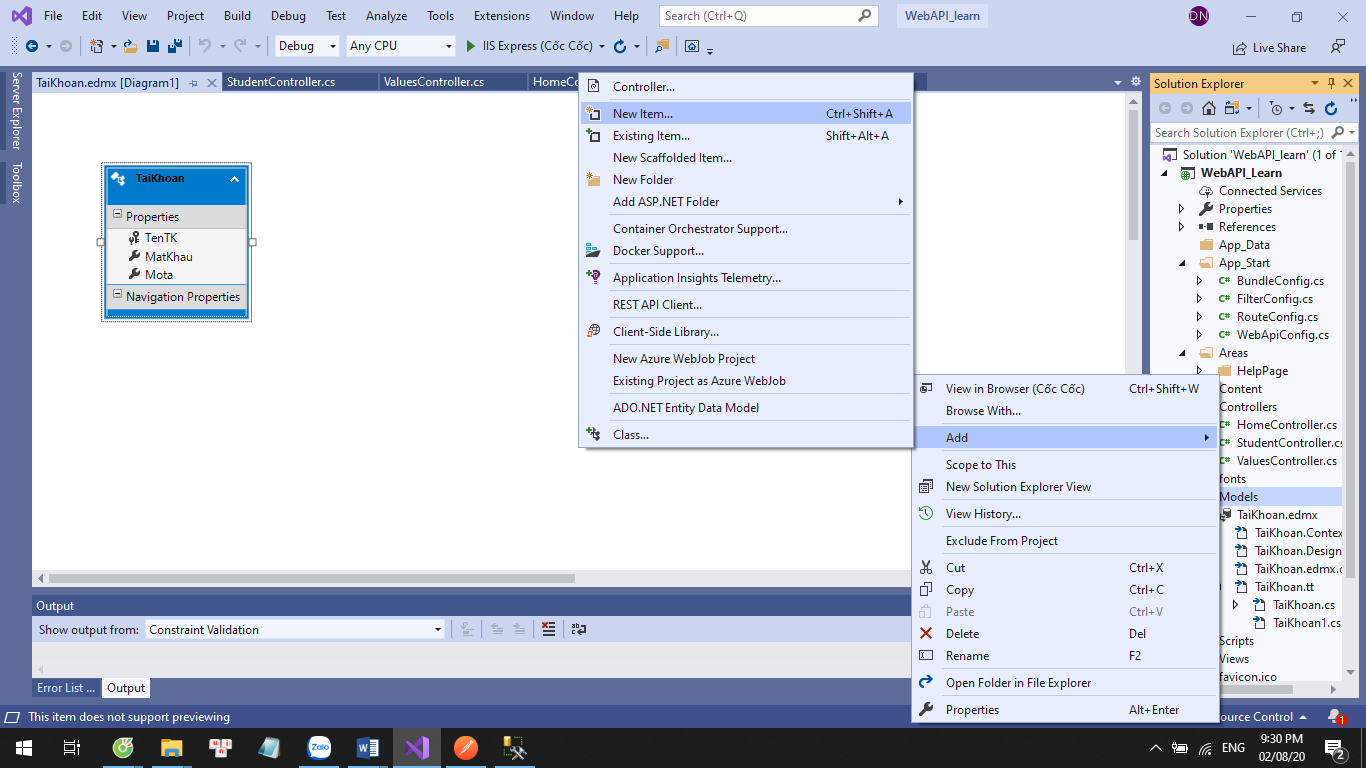


Khi chọn template WebAPI thì giao diện sẽ hỗ trợ theo form mô hình MVC và webAPi như sau:

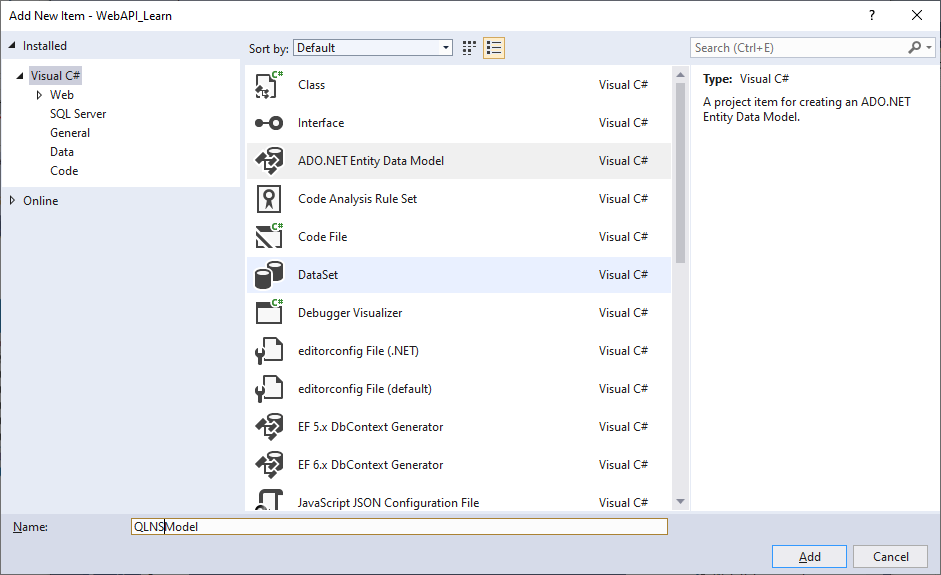


#### Bước 3: Kết nối với cơ sở dữ liệu sử dụng ADO.NET EF

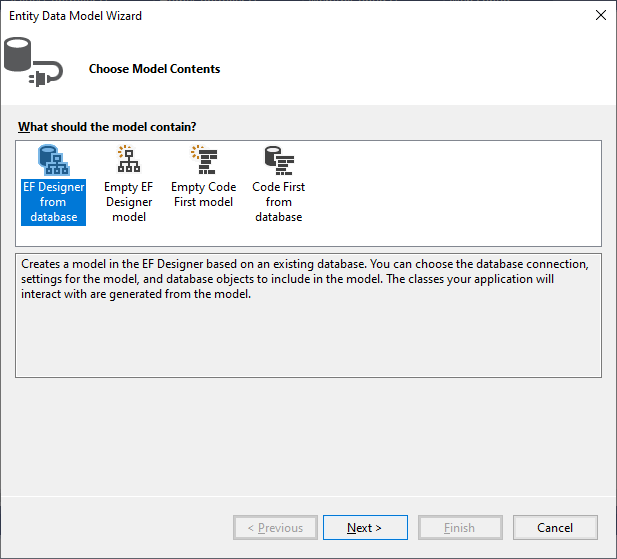
Để code tạo các model, nhấn chuột phải vào Models và chọn Add/New item



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

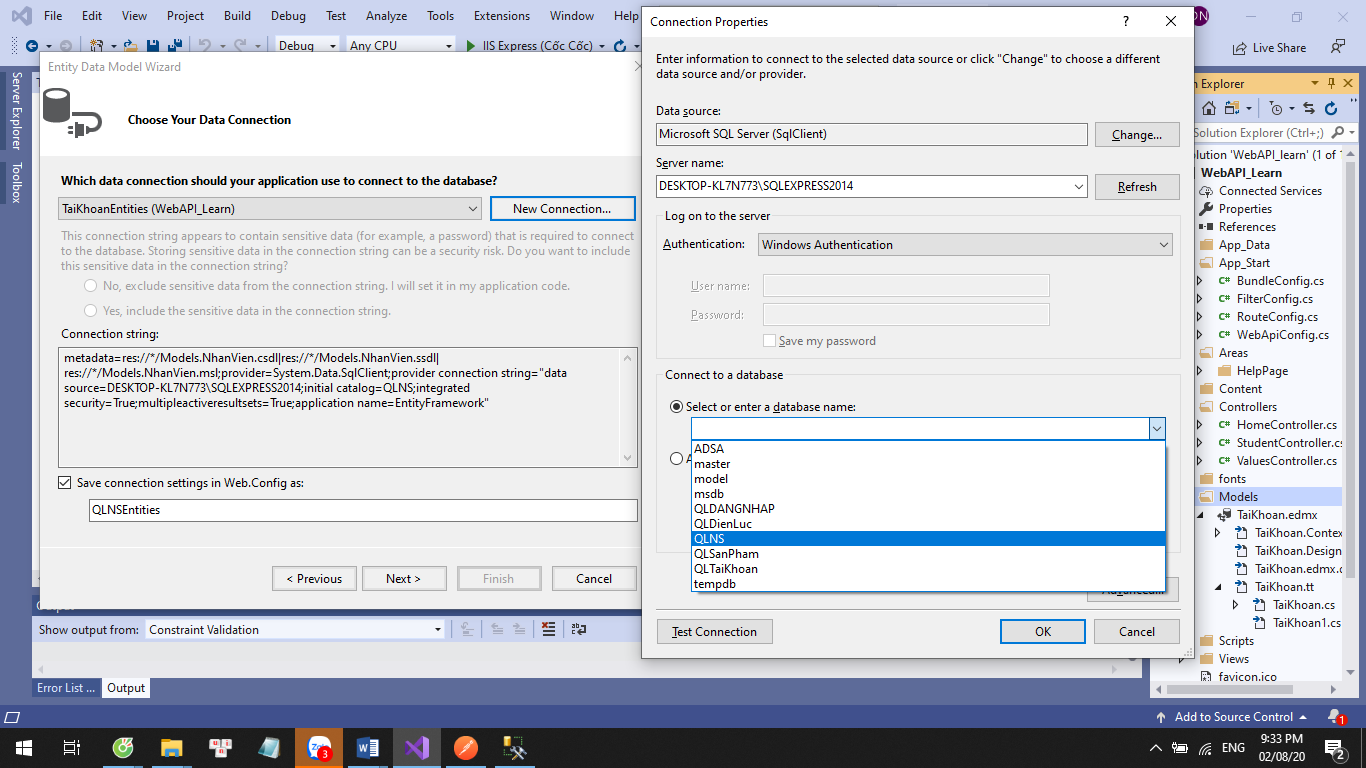


Nhập tên cho class, rồi chọn Add



Chọn 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)

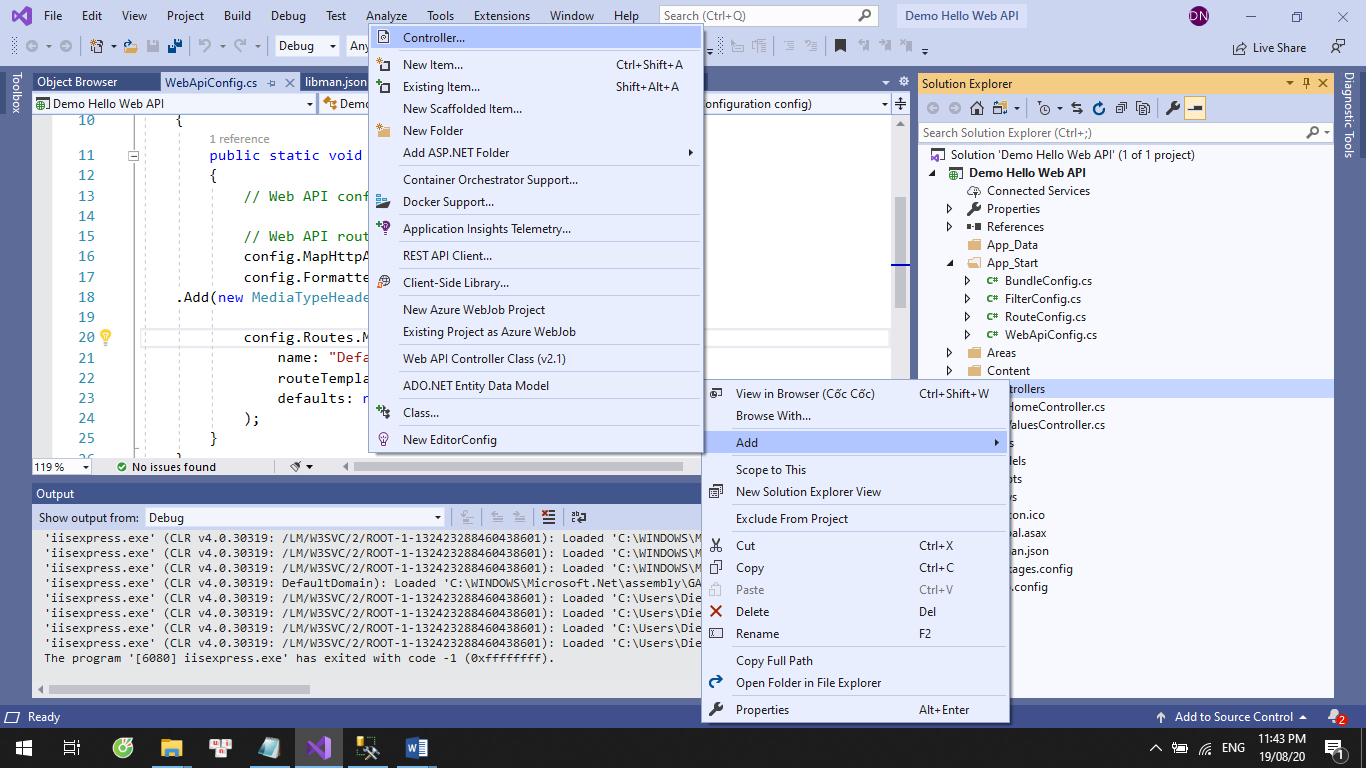


Next tiếp cho tới khi xuất hiện giao diện chọn table và Proc

Chọn Table TaiKhoan và các store proc và function cần thiết

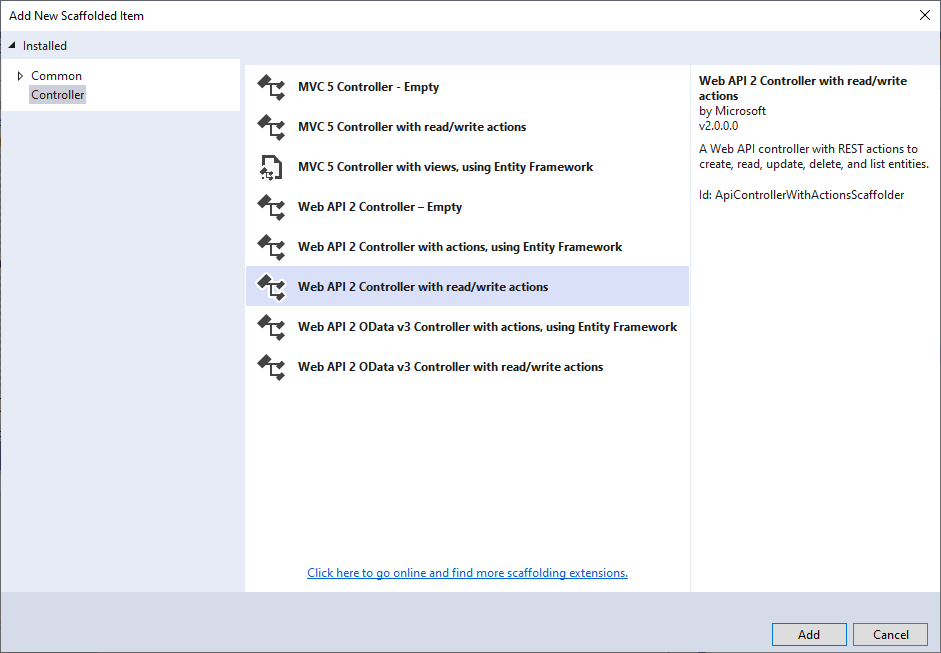
#### Bước 4: Viết Web API Controller

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



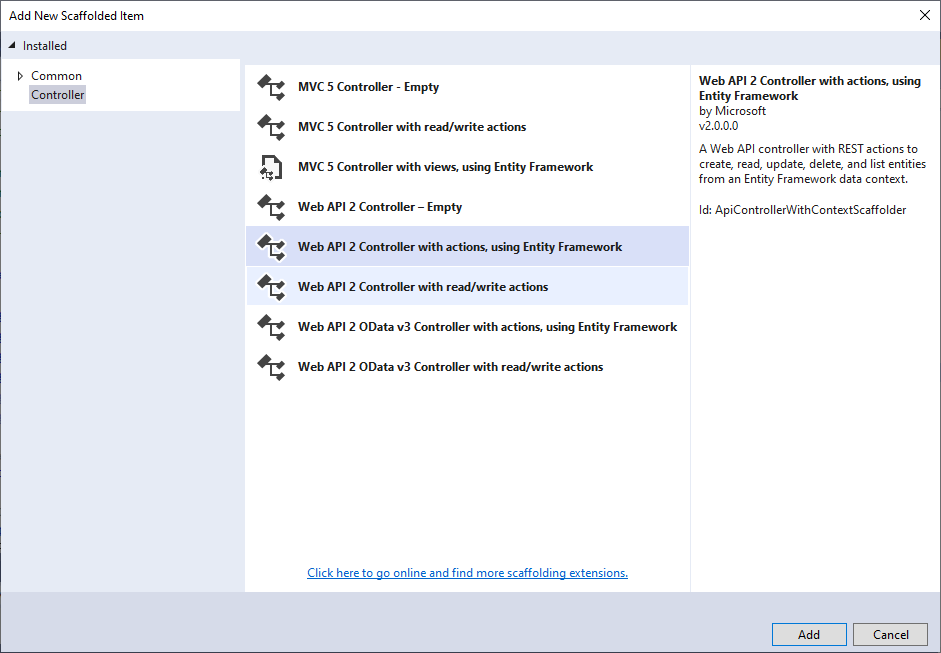
Trong đó! Có thể chọn loại controller khác nhau. Tuy nhiên nếu đã có CSDL, có sẵn data, đã có model tương ứng data thì nên chọn with action và using entity Framewwork để được hỗ trợ sinh code tự động tương ứng với EF trong bước 3

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



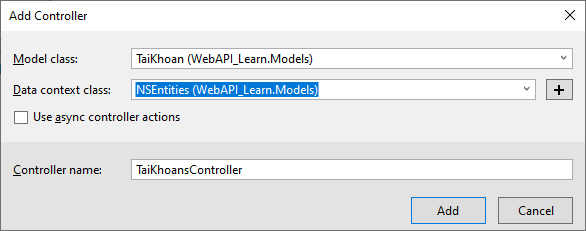
Chọn Add rồi đặt tên cho Controller. Sau đó viết code cho Controller

Nếu 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.

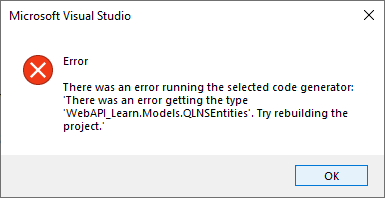


Lần lượt thêm các controller, mỗi table một model một controller,

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



Chú ý! Nếu sau khi sinh model, chưa rebuild App thì sẽ báo lỗi. Khi đó, đóng ứng dụng lại, rebuild lại ứng dụng, sau đó Add lại Controller là ok



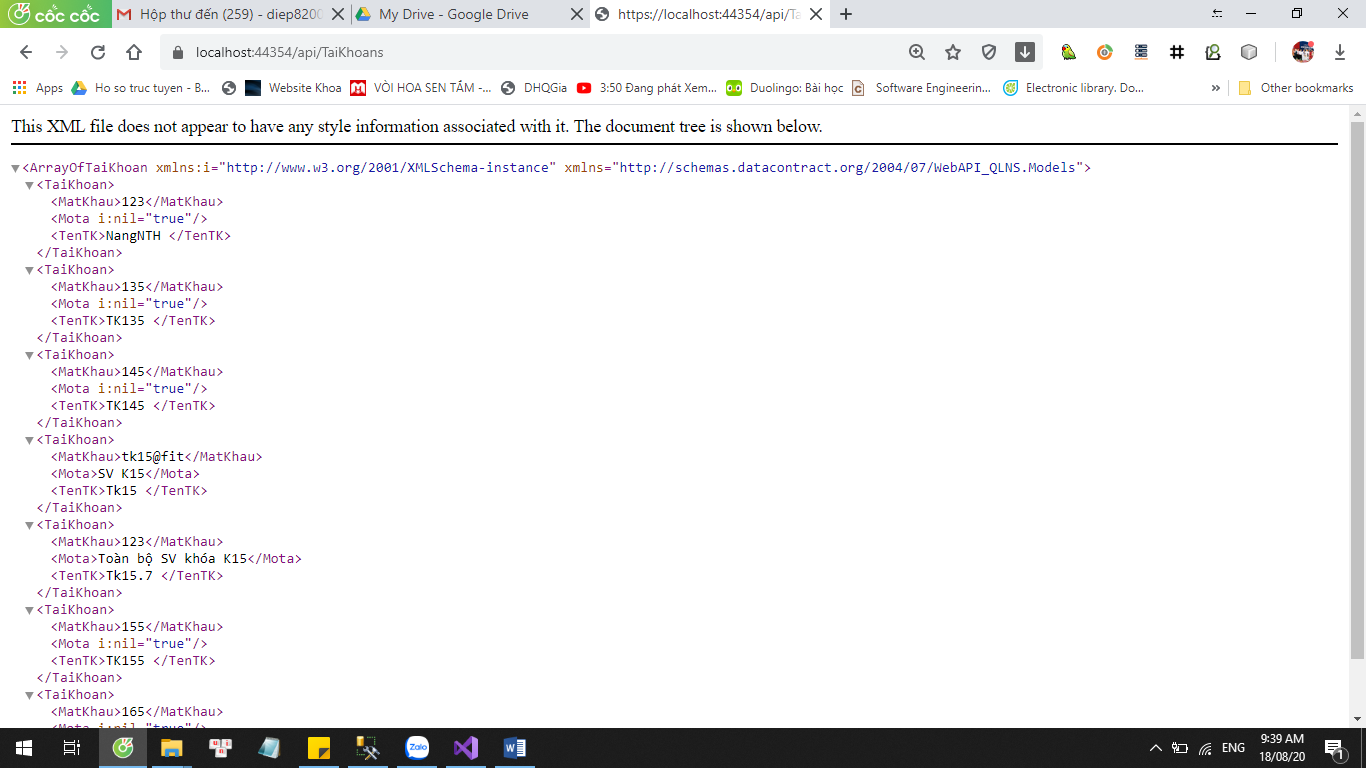
#### TaiKhoansController.cs như sau

|  |
| --- |
| 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 TaiKhoansController : ApiController  {  private NSEntities db = new NSEntities();  // GET: api/TaiKhoans  public IQueryable<TaiKhoan> GetTaiKhoans()  {  return db.TaiKhoans;  }  // GET: api/TaiKhoans/5  [ResponseType(typeof(TaiKhoan))]  public IHttpActionResult GetTaiKhoan(string id)  {  TaiKhoan taiKhoan = db.TaiKhoans.Find(id);  if (taiKhoan == null)  {  return NotFound();  }  return Ok(taiKhoan);  }  // PUT: api/TaiKhoans/5  [ResponseType(typeof(void))]  public IHttpActionResult PutTaiKhoan(string id, TaiKhoan taiKhoan)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  if (id != taiKhoan.TenTK)  {  return BadRequest();  }  db.Entry(taiKhoan).State = EntityState.Modified;  try  {  db.SaveChanges();  }  catch (DbUpdateConcurrencyException)  {  if (!TaiKhoanExists(id))  {  return NotFound();  }  else  {  throw;  }  }  return StatusCode(HttpStatusCode.NoContent);  }  // POST: api/TaiKhoans  [ResponseType(typeof(TaiKhoan))]  public IHttpActionResult PostTaiKhoan(TaiKhoan taiKhoan)  {  if (!ModelState.IsValid)  {  return BadRequest(ModelState);  }  db.TaiKhoans.Add(taiKhoan);  try  {  db.SaveChanges();  }  catch (DbUpdateException)  {  if (TaiKhoanExists(taiKhoan.TenTK))  {  return Conflict();  }  else  {  throw;  }  }  return CreatedAtRoute("DefaultApi", new { id = taiKhoan.TenTK }, taiKhoan);  }  // DELETE: api/TaiKhoans/5  [ResponseType(typeof(TaiKhoan))]  public IHttpActionResult DeleteTaiKhoan(string id)  {  TaiKhoan taiKhoan = db.TaiKhoans.Find(id);  if (taiKhoan == null)  {  return NotFound();  }  db.TaiKhoans.Remove(taiKhoan);  db.SaveChanges();  return Ok(taiKhoan);  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  db.Dispose();  }  base.Dispose(disposing);  }  private bool TaiKhoanExists(string id)  {  return db.TaiKhoans.Count(e => e.TenTK == id) > 0;  }  }  } |

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

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

<https://localhost:44302/api/TaiKhoans/diepnh>



Nếu muốn hiển thị dữ liệu dạng JSON thì cấu hình lại Content-Type thành JsonFormatter (trên App\_Start / WebApiConfig.cs  của wepAPI)

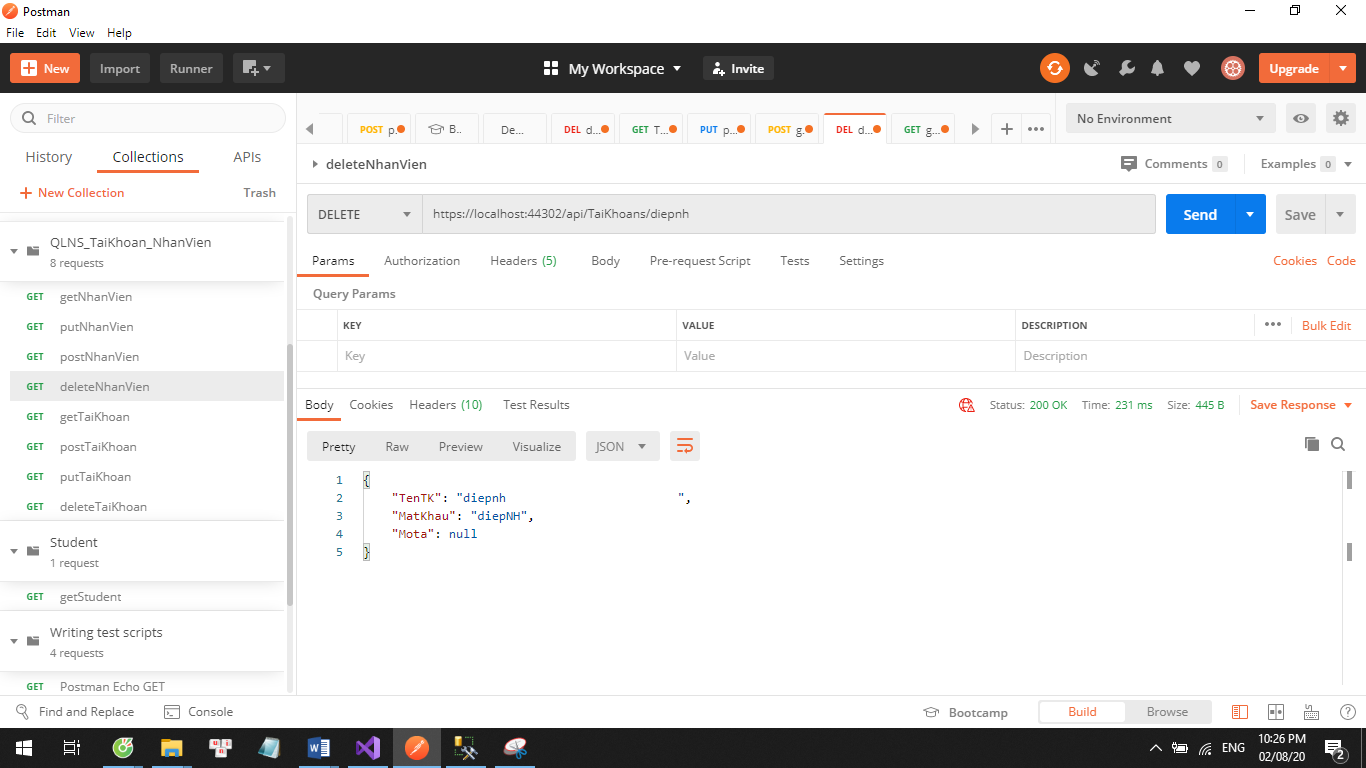
I just add the following in App\_Start / WebApiConfig.cs class in my MVC **Web API** project.

|  |
| --- |
| config.Formatters.JsonFormatter.SupportedMediaTypes  .Add(new MediaTypeHeaderValue("text/html") ); |

Trong đó: Có thể import package bằng khai báo using System.Net.Http.Headers;

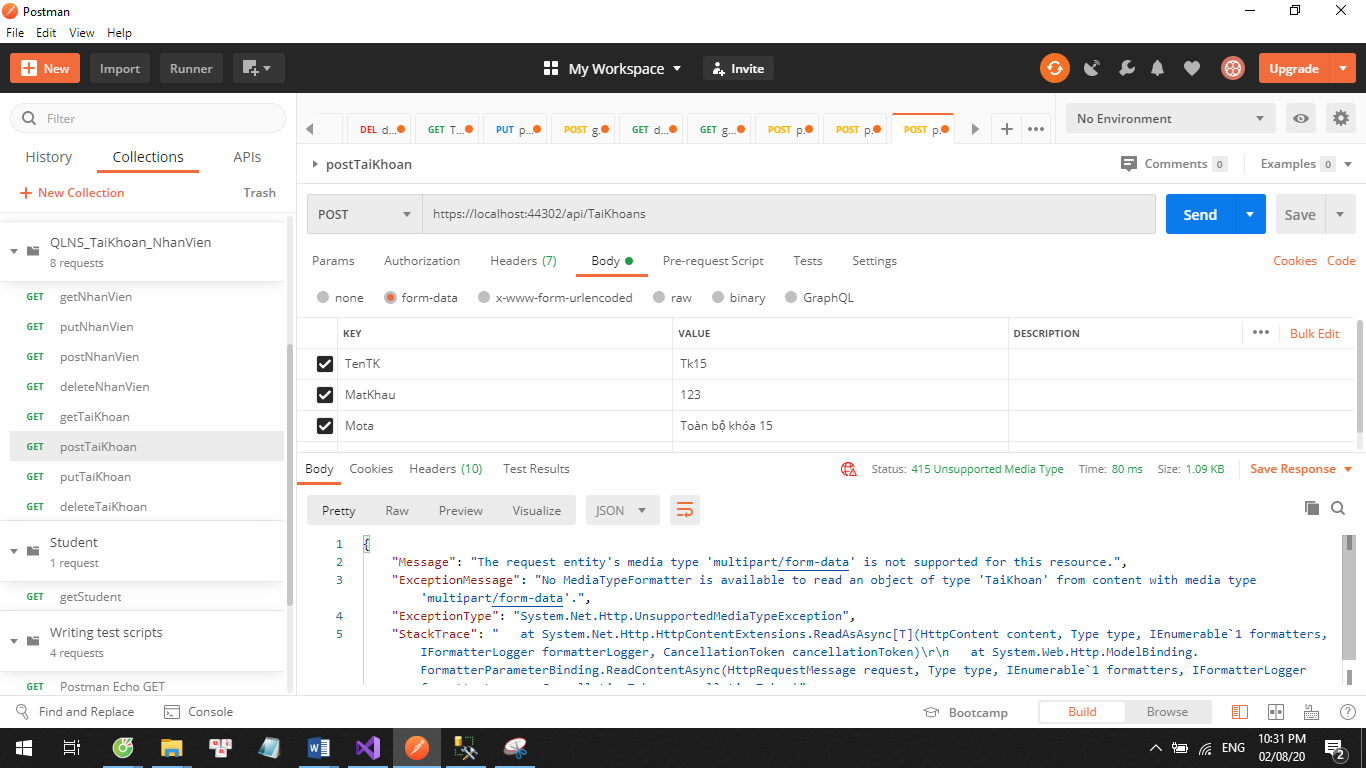
Hoặc Alt+enter để import System.Net.Http.Headers

#### Httpdelete status báo 200 ok là chạy thành công



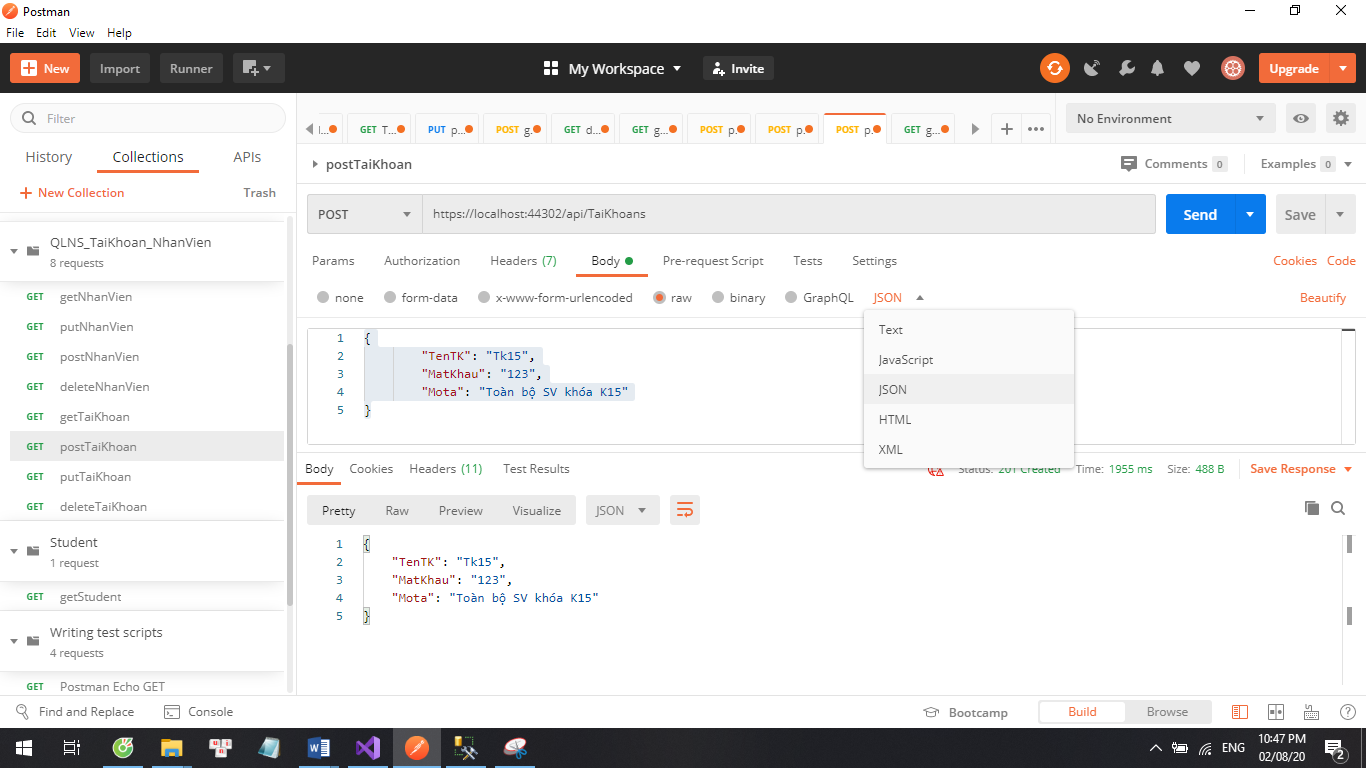
### httppost để thêm mới

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



Chọn body và dữ liệu dạng raw/JSON như sau:

|  |
| --- |
| {  "TenTK": "Tk15",  "MatKhau": "123",  "Mota": "Toàn bộ SV khóa K15"  } |



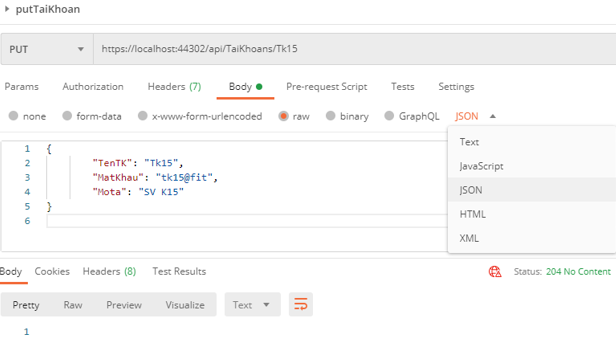
Chú ý! Nếu chưa chọn kiểu dữ liệu truyền thích hợp với dữ liệu => Ứng dụng báo lỗi 415 únsupport Media Type

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

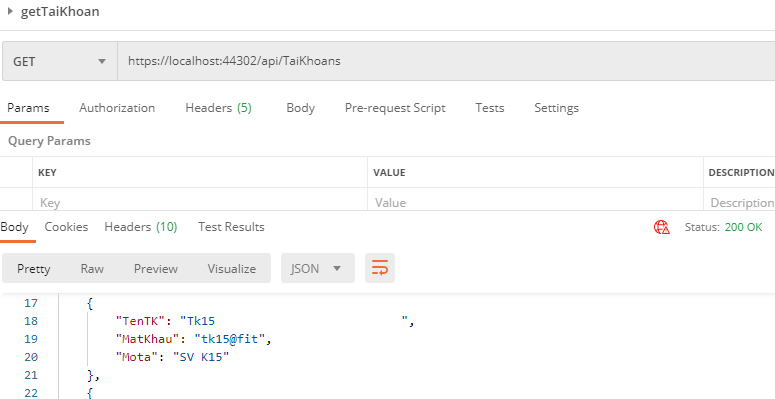
### Httpput update

Chọn body và dữ liệu dạng raw/JSON, Id của object đã có trong CSDL như sau:

|  |
| --- |
| {  "TenTK": "Tk15",  "MatKhau": "tk15@fit",  "Mota": "SV K15"  } |



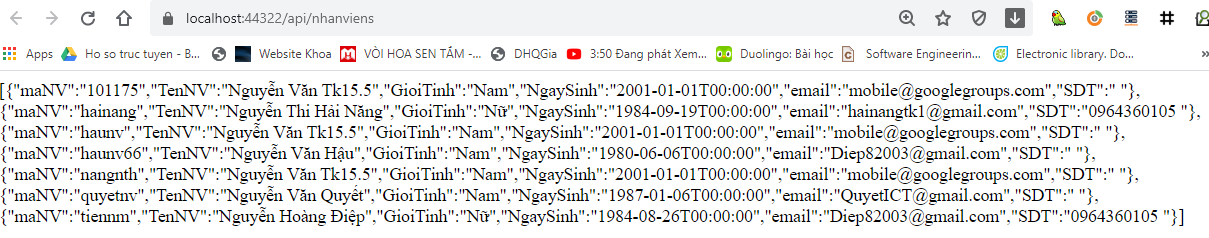
Đọc kết quả hoặc dùng HttpGet để kiểm tra kết quả sau khi update



Bài tập tự làm

**Bài tập 1 Yêu cầu**

* Tạo CSDL có thêm bảng NhanVien.
* Xây dựng WebAPI Controllers thực hiện CRUD trên bảng NhanVien trong CSDL
* Chạy và kiểm tra ứng dụng WebAPI xây dựng với các công cụ kiểm tra WebAPI: Swagger, Postman và Fiddler



Gợi ý

Tạo bảng NhanVien

|  |
| --- |
| 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', '') |

**Bài tập 2 Yêu cầu**

Tạo CSDL có thêm bảng Sach gồm: MASH, TENSACH, TACGIA, NHAXB, NAMXB

* Xây dựng WebAPI Controllers thực hiện CRUD trên bảng Sach trong CSDL
* Chạy và kiểm tra ứng dụng WebAPI xây dựng với các công cụ kiểm tra WebAPI: Swagger, Postman và Fiddler

**Bài tập 3 Yêu cầu**

Tạo CSDL có thêm bảng **SinhVien** gồm: MaSV, Ten\_SV, Gioi\_tinh, Dia\_chi, Ngay\_sinh

* Xây dựng WebAPI Controllers thực hiện CRUD trên bảng **SinhVien** trong CSDL
* Chạy và kiểm tra ứng dụng WebAPI xây dựng với các công cụ kiểm tra WebAPI: Swagger, Postman và Fiddler