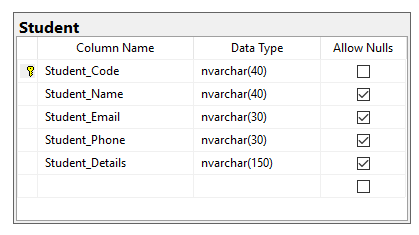
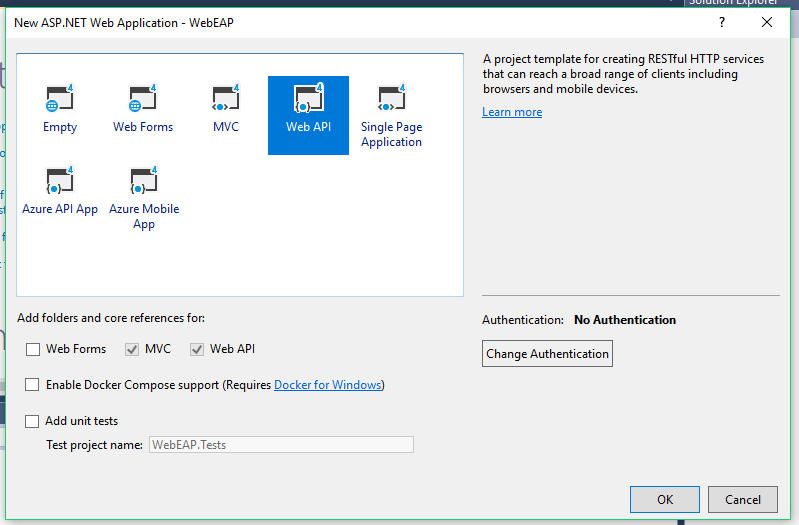
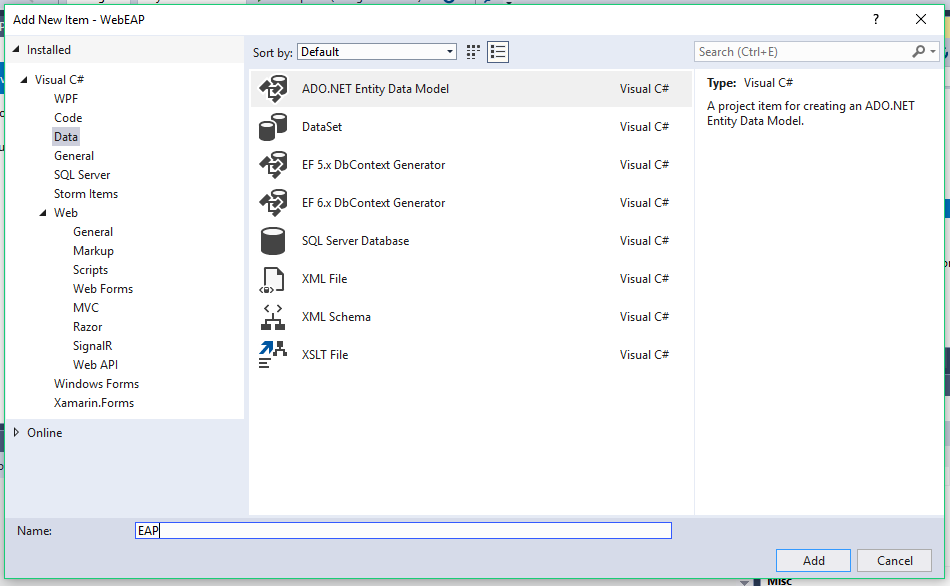
**Reviews EAPs**

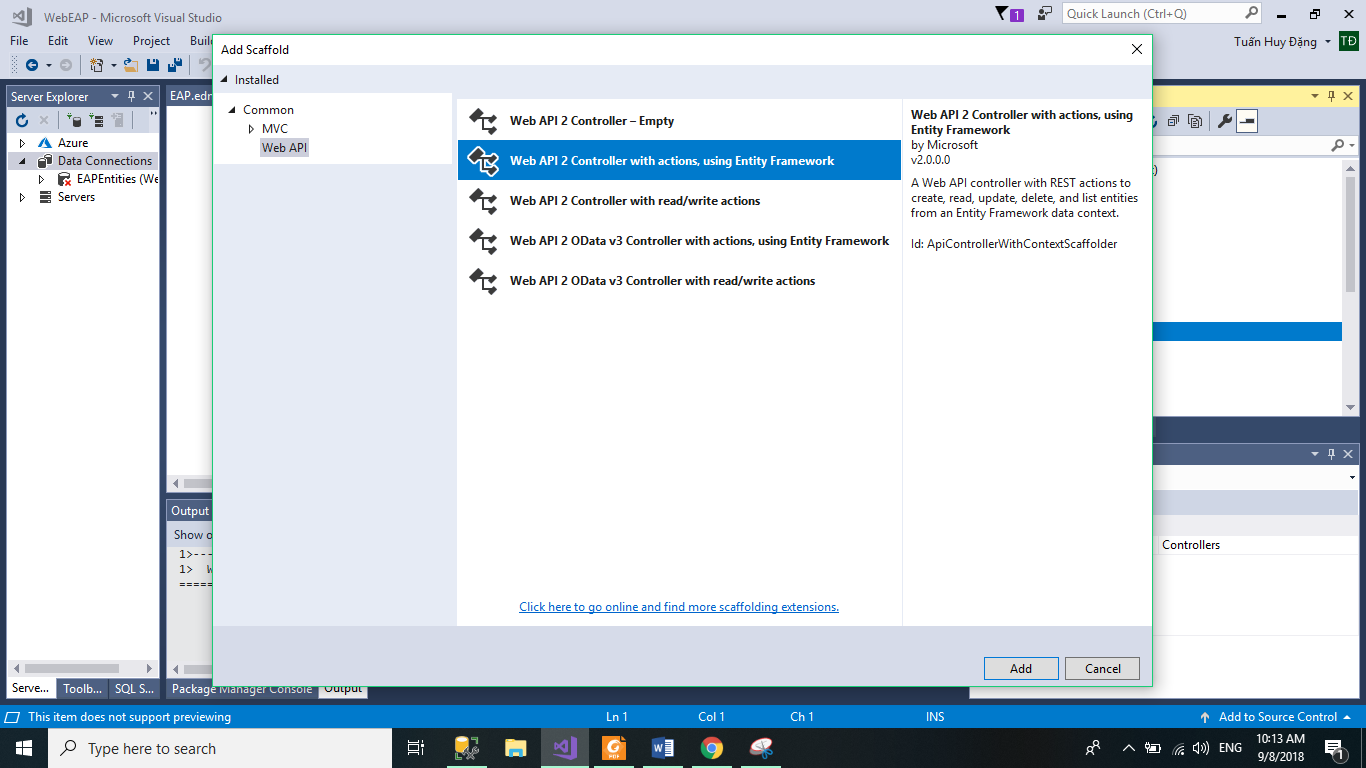
1. **CSDL (EAP) – tables Student**
2. **Tạo API**
3. **Tạo một Project API-Web**



1. **Sử dụng EntityFramwwork tạo Models**



1. **Tạo controller**



1. **Hiệu chỉnh**

**Trong Application\_Start() bổ sung code convert xml->json**

using System.Web.Http;

using System.Web.Mvc;

using System.Web.Optimization;

using System.Web.Routing;

namespace WebEAP

{

public class WebApiApplication : System.Web.HttpApplication

{

protected void Application\_Start()

{

GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings.ReferenceLoopHandling = Newtonsoft.Json.ReferenceLoopHandling.Ignore;

GlobalConfiguration.Configuration.Formatters.Remove(GlobalConfiguration.Configuration.Formatters.XmlFormatter);

AreaRegistration.RegisterAllAreas();

GlobalConfiguration.Configure(WebApiConfig.Register);

FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);

RouteConfig.RegisterRoutes(RouteTable.Routes);

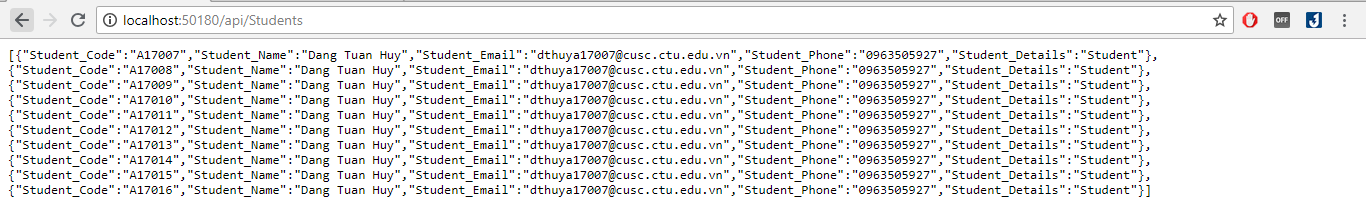
BundleConfig.RegisterBundles(BundleTable.Bundles);

}

}

}

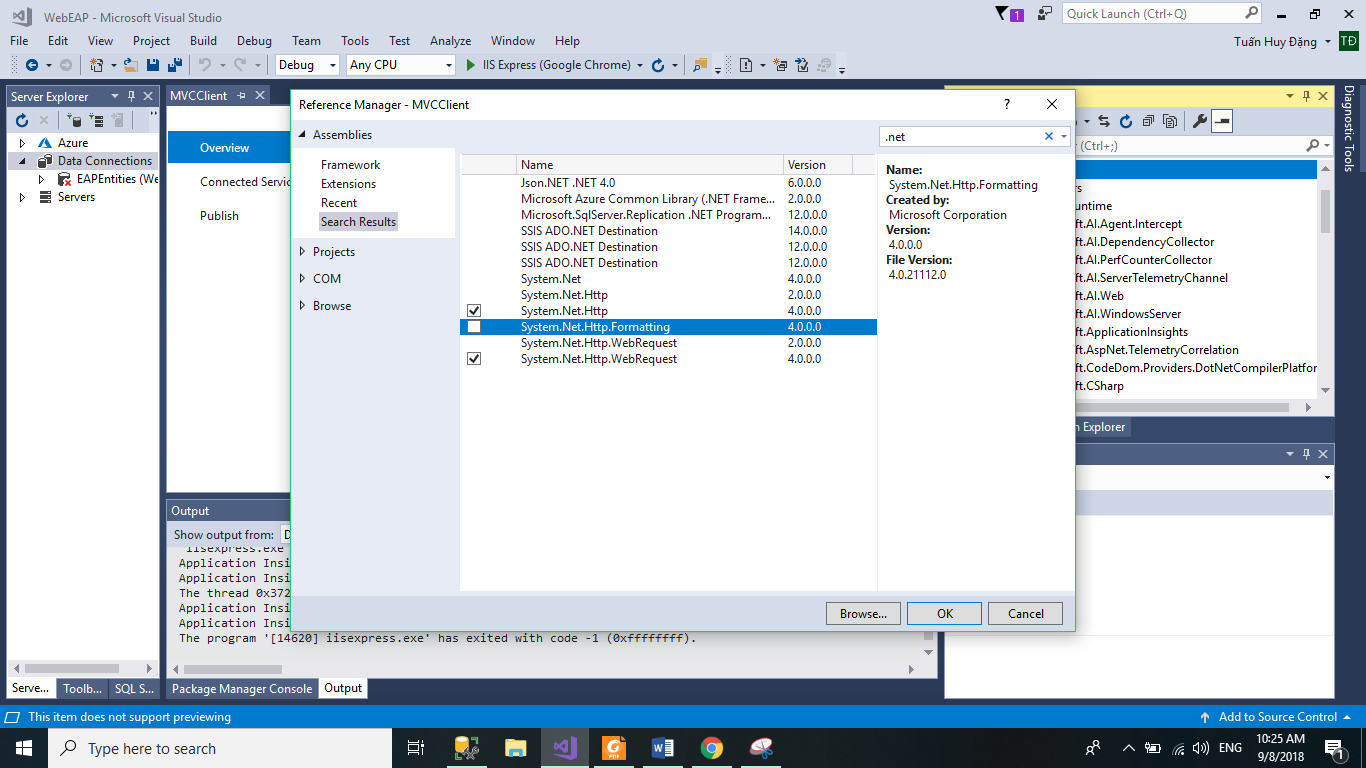
**Kết quả**



1. **API – Client**
2. **Web MVC – Client**

***Trong cùng Solution tạo một Project MVCClient***

***Trong References của MVCClient hiệu chỉnh***

1. **Trong model tạo một Class có tên là Student.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace MVCClient.Models

{

public class Student

{

public string Student\_Code { get; set; }

public string Student\_Name { get; set; }

public string Student\_Email { get; set; }

public string Student\_Phone { get; set; }

public string Student\_Details { get; set; }

}

}

1. **Tạo một controller emty có tên là StudentsController.cs Bổ sung code gọi API**

using MVCClient.Models;

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Threading.Tasks;

using System.Web;

using System.Web.Mvc;

namespace MVCClient.Controllers

{

public class StudentsController : Controller

{

HttpClient client;

string url = "http://localhost:50180/api/Students";

public StudentsController()

{

client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

}

// Get Student

public async Task<ActionResult> Index()

{

HttpResponseMessage responseMessage = await client.GetAsync(url);

if(responseMessage.IsSuccessStatusCode)

{

var responseData = responseMessage.Content.ReadAsStringAsync().Result;

var students = JsonConvert.DeserializeObject<List<Student>>(responseData);

return View(students);

}

return View("Error");

}

//Form Create

public ActionResult Create()

{

return View(new Student());

}

[HttpPost]

public async Task<ActionResult> Create(Student student)

{

HttpResponseMessage responseMessage = await client.PostAsJsonAsync(url,student);

if (responseMessage.IsSuccessStatusCode)

{

return RedirectToAction("Index");

}

return RedirectToAction("Error");

}

//Form Edit

public async Task<ActionResult> Edit(string id)

{

HttpResponseMessage responseMessage = await client.GetAsync(url + "/" + id);

if (responseMessage.IsSuccessStatusCode)

{

var responseData = responseMessage.Content.ReadAsStringAsync().Result;

var student = JsonConvert.DeserializeObject<Student>(responseData);

return View(student);

}

return View("Error");

}

[HttpPost]

public async Task<ActionResult> Edit(string id, Student student)

{

HttpResponseMessage responseMessage = await client.PutAsJsonAsync(url + "/" + id, student);

if (responseMessage.IsSuccessStatusCode)

{

return RedirectToAction("Index");

}

return RedirectToAction("Error");

}

//Form Delete

public async Task<ActionResult> Delete(string id)

{

HttpResponseMessage responseMessage = await client.GetAsync(url + "/" + id);

if (responseMessage.IsSuccessStatusCode)

{

var responseData = responseMessage.Content.ReadAsStringAsync().Result;

var student = JsonConvert.DeserializeObject<Student>(responseData);

return View(student);

}

return View("Error");

}

[HttpPost]

public async Task<ActionResult> Delete(string id, Student student)

{

HttpResponseMessage responseMessage = await client.DeleteAsync(url + "/" +id);

if (responseMessage.IsSuccessStatusCode)

{

return RedirectToAction("Index");

}

return RedirectToAction("Error");

}

}

}

1. **Tạo Views từ model Student**

**Code Index**

@model IEnumerable<MVCClient.Models.Student>

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>List Student</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table table-bordered">

<tr>

<th>

@Html.DisplayNameFor(model => model.Student\_Code)

</th>

<th>

@Html.DisplayNameFor(model => model.Student\_Name)

</th>

<th>

@Html.DisplayNameFor(model => model.Student\_Email)

</th>

<th>

@Html.DisplayNameFor(model => model.Student\_Phone)

</th>

<th>

@Html.DisplayNameFor(model => model.Student\_Details)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Student\_Code)

</td>

<td>

@Html.DisplayFor(modelItem => item.Student\_Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Student\_Email)

</td>

<td>

@Html.DisplayFor(modelItem => item.Student\_Phone)

</td>

<td>

@Html.DisplayFor(modelItem => item.Student\_Details)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.Student\_Code}) |

@Html.ActionLink("Delete", "Delete", new {id = item.Student\_Code })

</td>

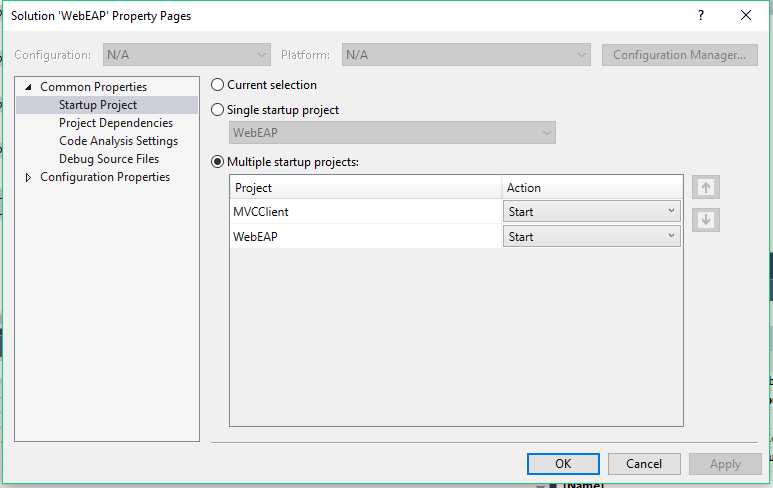
</tr>

}

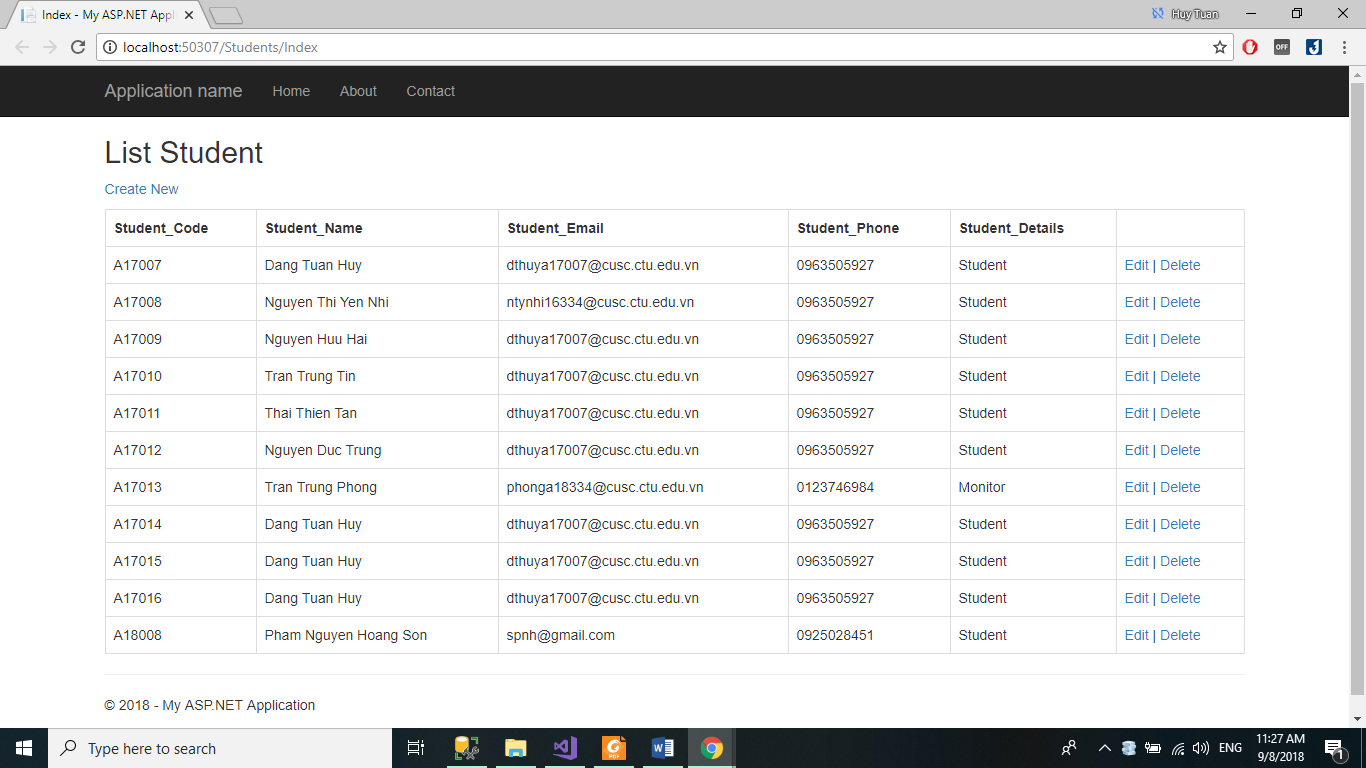
</table>

1. **Kết quả**

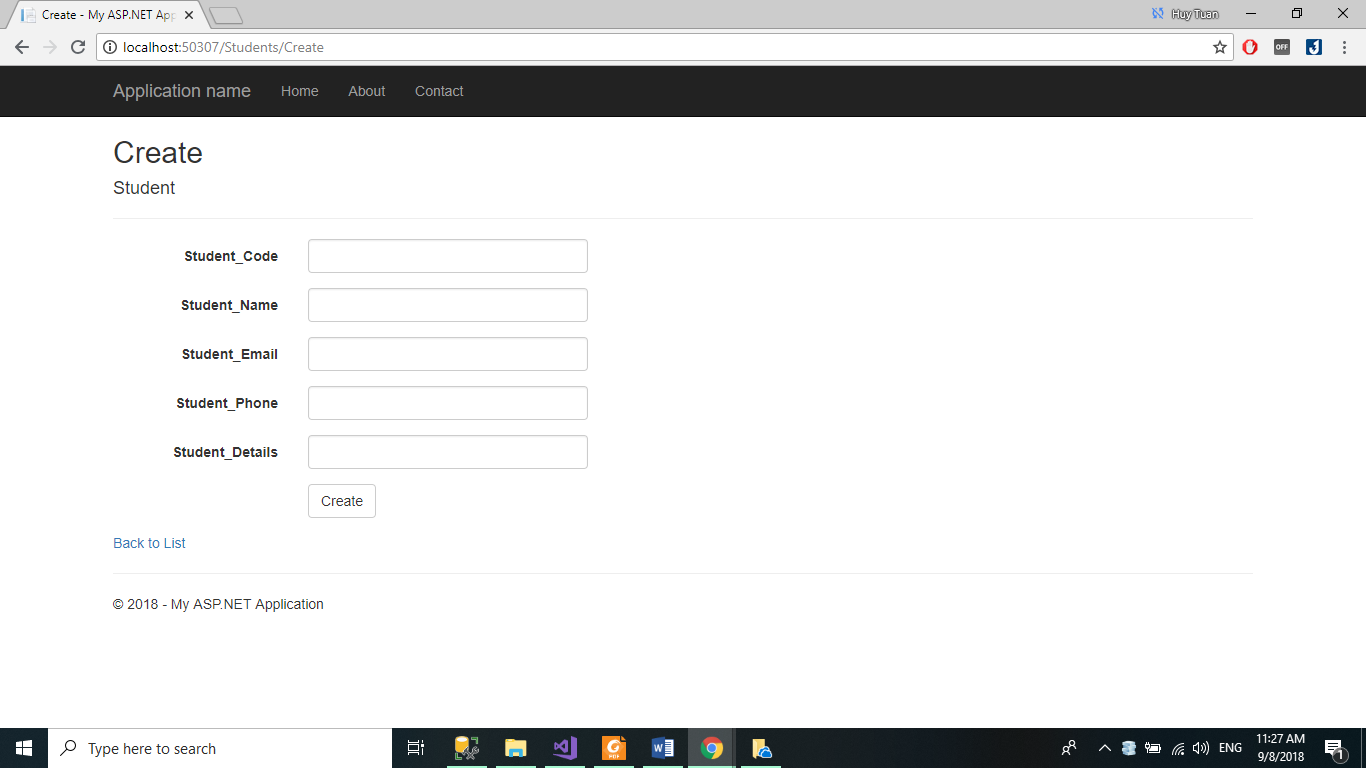
**Lưu ý: config để API chạy cùng với MVC**



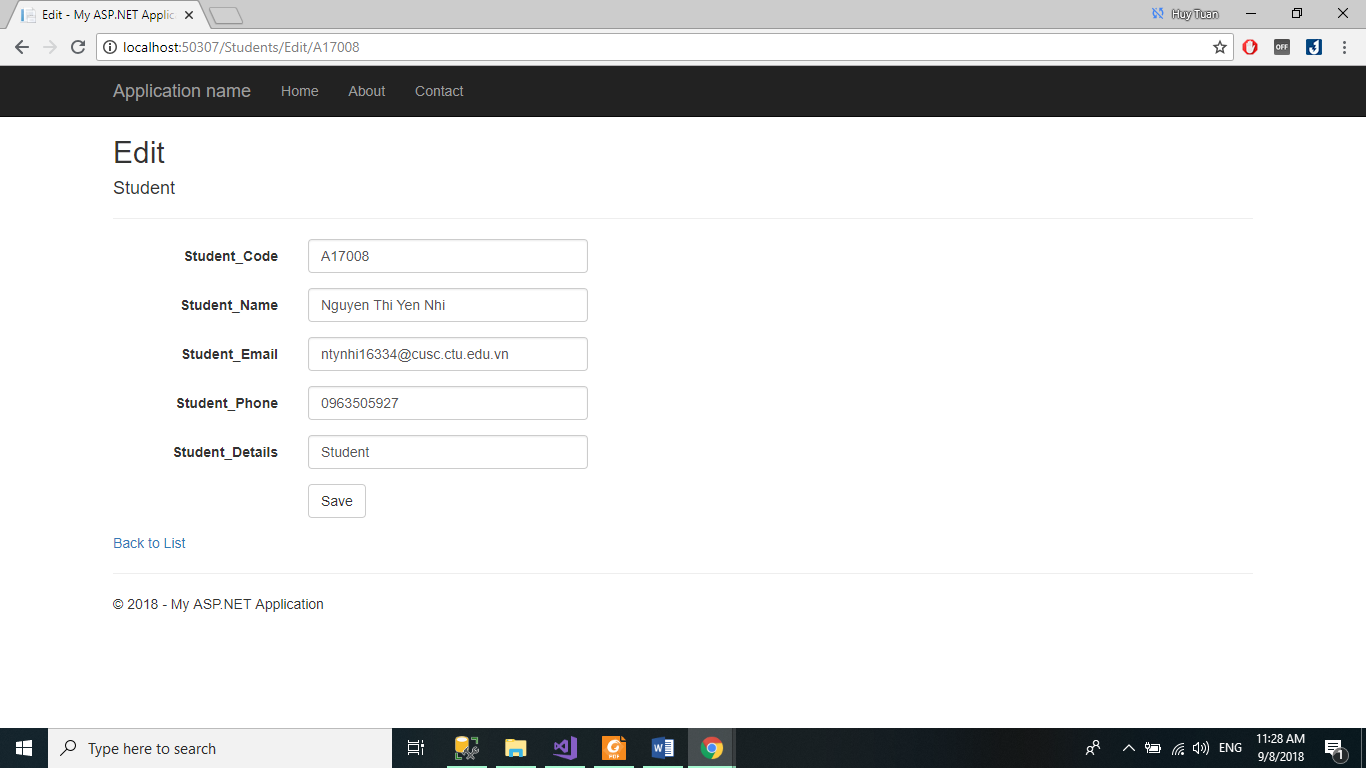
* **Index:**



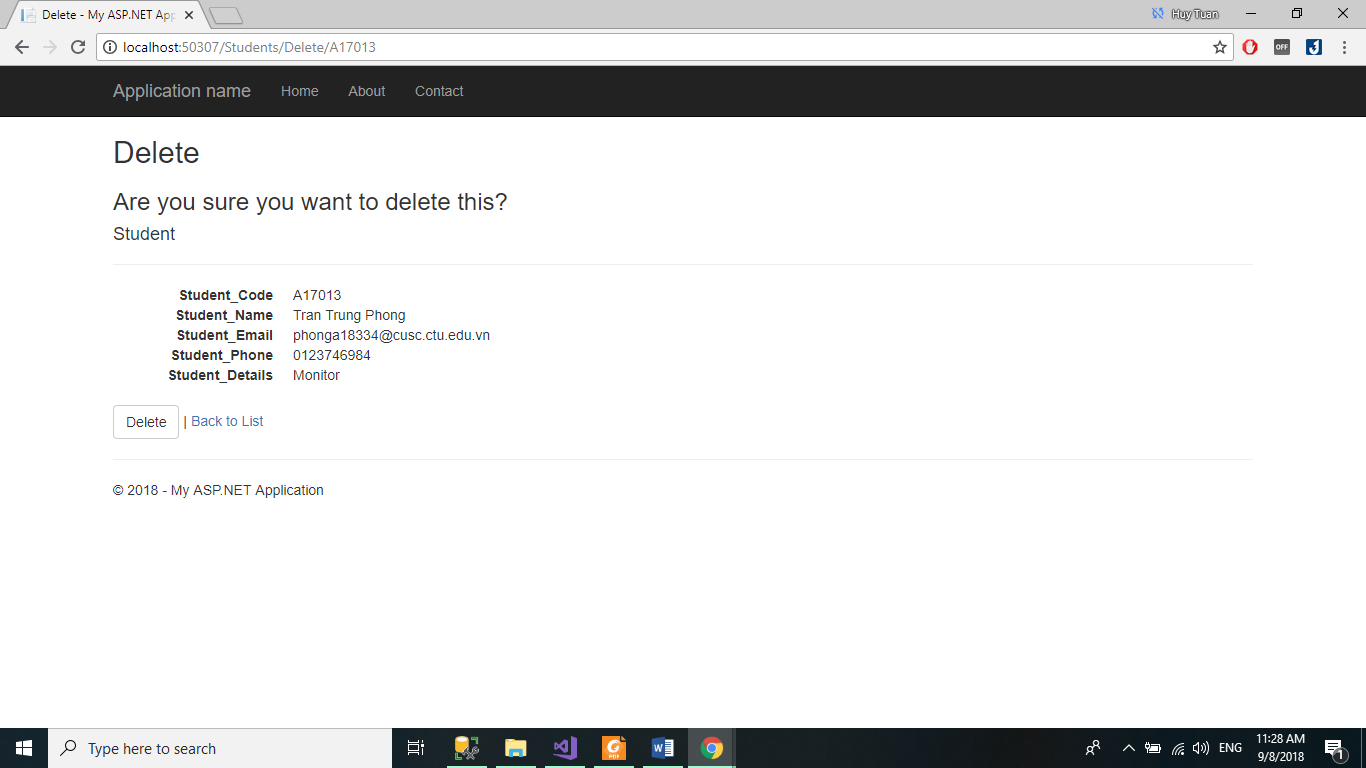
* **Create**

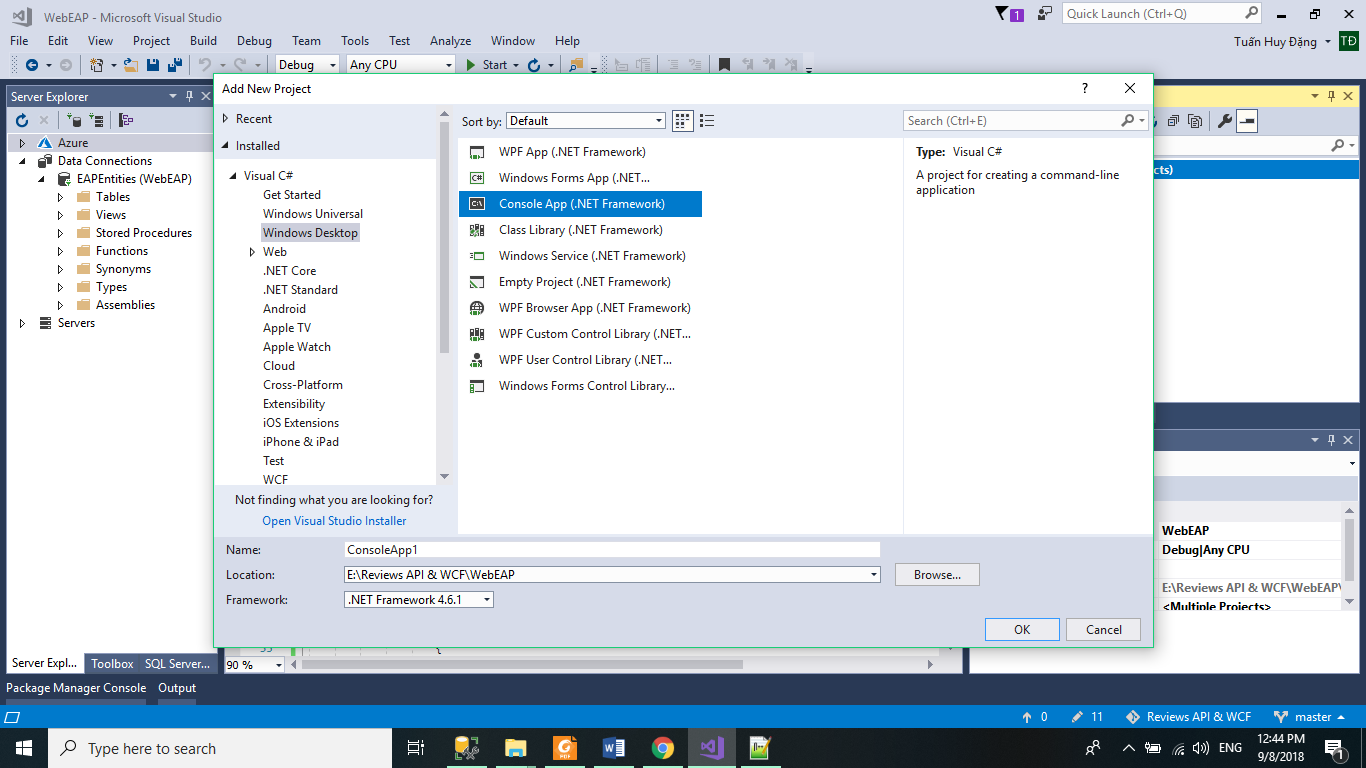
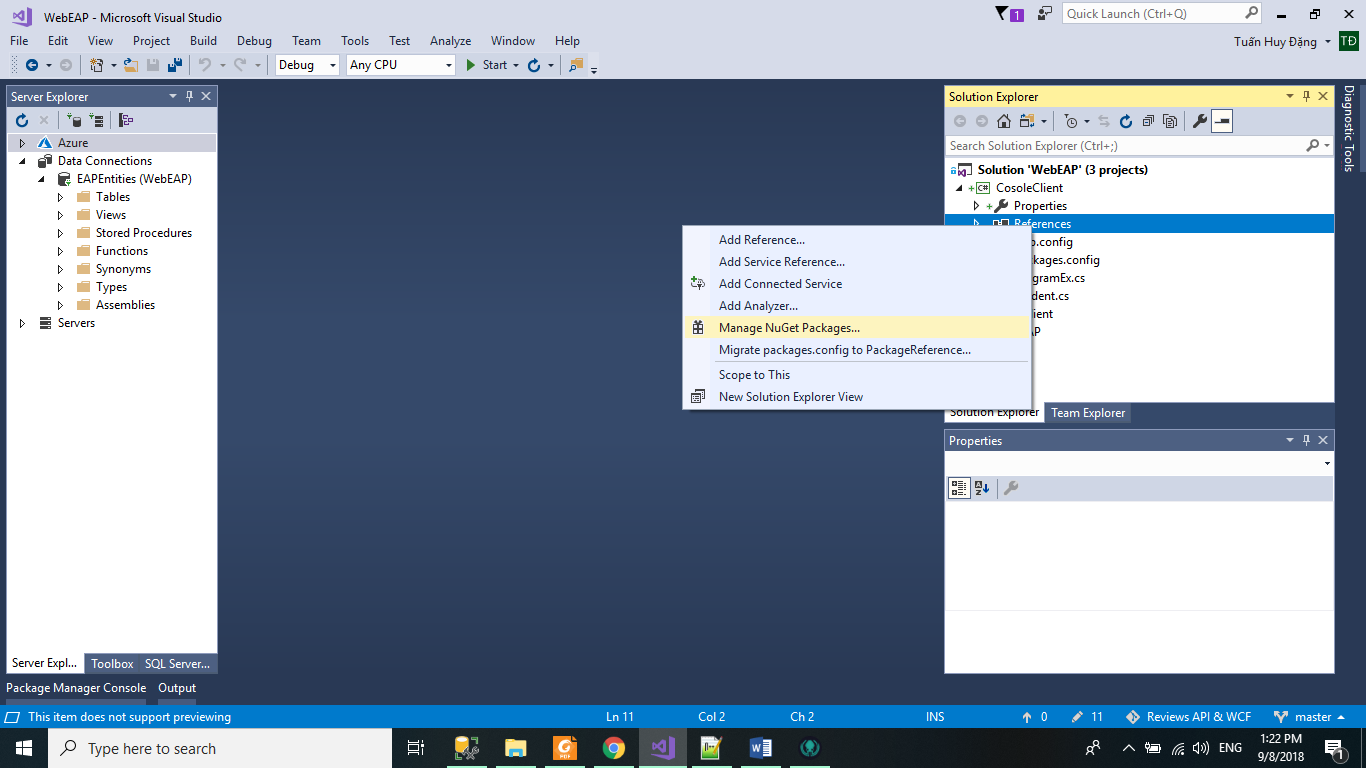


* **Edit**

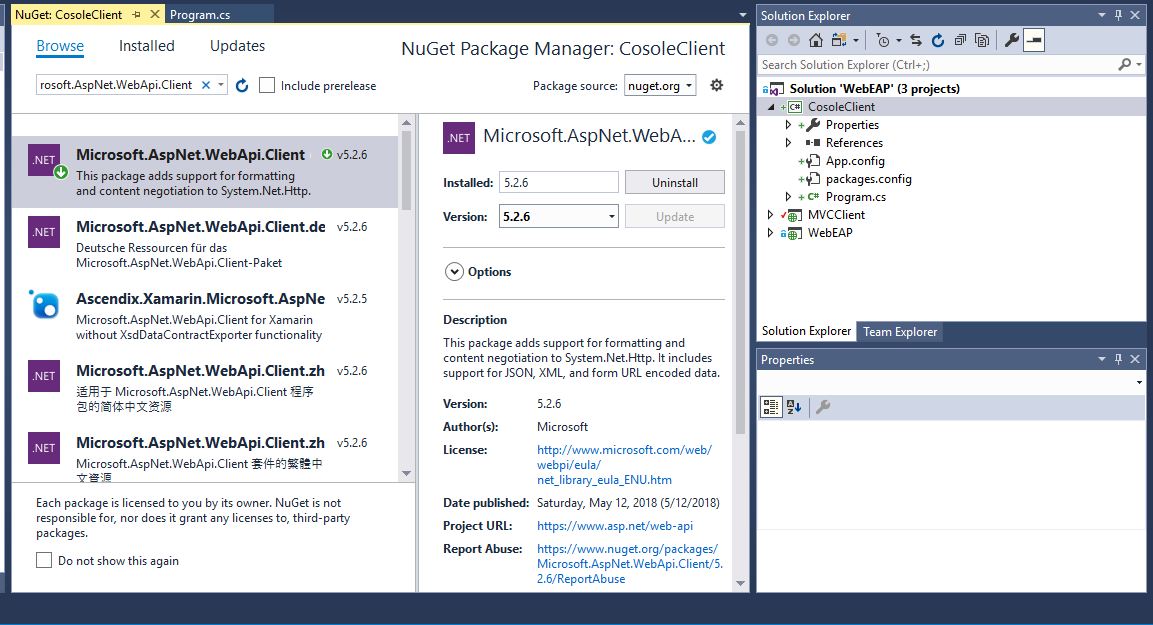


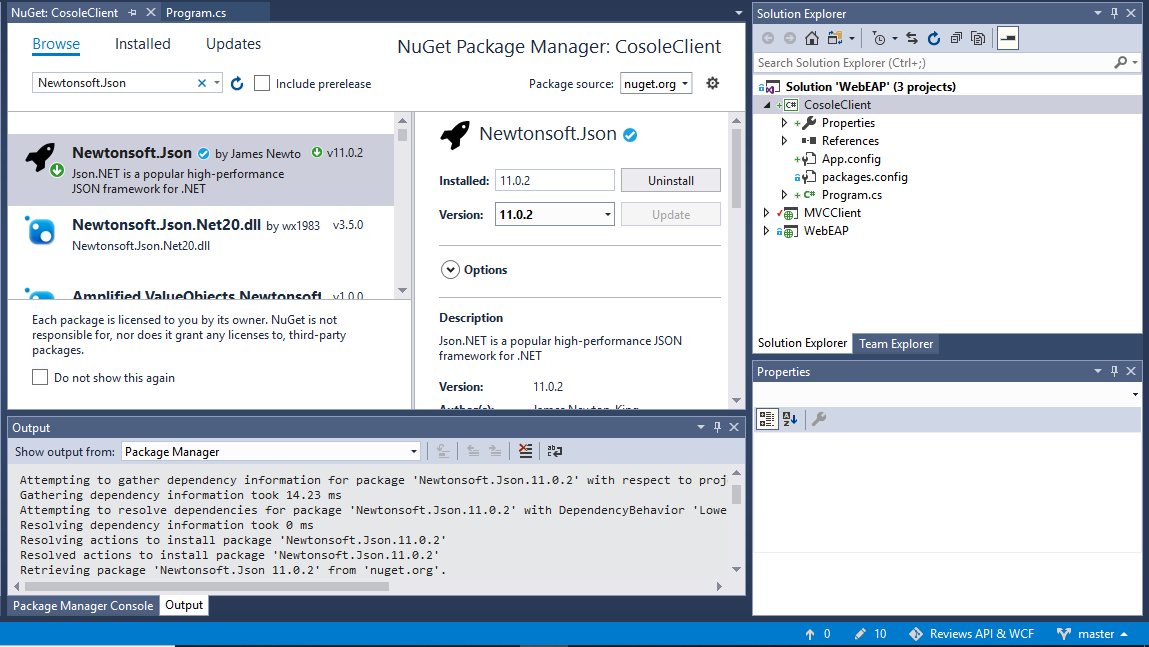
* **Delete**



1. **Console – Client**
2. **Trong cùng solution tạo một Project Console Client (Cosole Application)**
3. **Thêm hai package cho cosele App vừa tạo**

**Microsoft.AspNet.WebApi.Client**

**Newtonsoft.Json** 

1. **Tạo một Class có tên là Student.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CosoleClient

{

class Student

{

public string Student\_Code { get; set; }

public string Student\_Name { get; set; }

public string Student\_Email { get; set; }

public string Student\_Phone { get; set; }

public string Student\_Details { get; set; }

}

}

1. **Đổi tên Programs.cs thành ProgramEx.cs và bổ sung code gọi API**

using Newtonsoft.Json.Linq;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

namespace CosoleClient

{

class ProgramEx

{

// Lấy thông tin tất cả sinh viên

static async Task GetAll()

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url);

if (response.IsSuccessStatusCode)

{

var responseData = response.Content.ReadAsStringAsync().Result;

JArray parsed = JArray.Parse(responseData.ToString());

Console.WriteLine("------------------------List of Student ' Information-------------");

foreach (var pair in parsed)

{

JObject obj = JObject.Parse(pair.ToString());

foreach (var s in obj)

{

Console.WriteLine("{0} : {1}", s.Key, s.Value.ToString().Trim());

}

Console.WriteLine();

}

}

}

// Tìm Kiếm sinh viên theo ID nè

static async Task GetStudent(string id)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url + "/" + id);

if (response.IsSuccessStatusCode)

{

Student student = await response.Content.ReadAsAsync<Student>();

Console.WriteLine("--------------------Lecturer information--------------------");

Console.WriteLine("Student Code: " + id);

Console.WriteLine("Student Name: " + student.Student\_Name);

Console.WriteLine("Student Email: " + student.Student\_Email);

Console.WriteLine("Student Phone: " + student.Student\_Phone);

Console.WriteLine("Student Details: " + student.Student\_Details.ToString().Trim());

}

else

{

Console.WriteLine("Lecturer is not exist");

}

}

static async Task AddNewStudent(string id, string name, string email, string phone, string details)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url);

Student st = new Student()

{

Student\_Code = id,

Student\_Name = name,

Student\_Email = email,

Student\_Phone = phone,

Student\_Details = details

};

response = await client.PostAsJsonAsync(url, st);

if (response.IsSuccessStatusCode)

{

Console.WriteLine("--------------------Student's information is added--------------------");

}

else

{

Console.WriteLine("--------------------Student's information cannot addto database--------------------");

}

}

static async Task UpdateStudent(string id, string name, string email, string phone, string details)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url + "/" + id);

if (response.IsSuccessStatusCode)

{

Student student = await response.Content.ReadAsAsync<Student>();

if (name != "")

student.Student\_Name = name;

if (email != "")

student.Student\_Email = email;

if (phone != "")

student.Student\_Phone = phone;

if (details != "")

student.Student\_Details = details;

response = await client.PutAsJsonAsync(url + "/" + id, student);

if (response.IsSuccessStatusCode)

{

Console.WriteLine("--------------------Update Student Successfully--------------------");

}

else

{

Console.WriteLine("--------------------Update Failure--------------------");

}

}

}

static async Task DeleteStudent(string id)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url + "/" + id);

response = await client.DeleteAsync(url + "/" + id);

if (response.IsSuccessStatusCode)

{

Console.WriteLine("--------------------Delete Successfully--------------------");

}

else

{

Console.WriteLine("--------------------Delete Failure--------------------");

}

}

static void Main(string[] args)

{

string i;

do

{

Console.WriteLine("1. Get all Student' information");

Console.WriteLine("2. Get only one student's information");

Console.WriteLine("3. Add new student");

Console.WriteLine("4. Update student's information");

Console.WriteLine("5. Delete student from database");

Console.Write("Enter your choice: ");

int ans = Convert.ToInt32(Console.ReadLine());

if (ans == 1)

{

GetAll().Wait();

}

else

{

if (ans == 2)

{

Console.Write("Enter the student code: ");

string id = Console.ReadLine();

GetStudent(id).Wait();

}

else

{

if (ans == 3)

{

Console.WriteLine("--------------------Enter the Student'sInformation--------------------");

Console.Write("Enter Student Code:");

string id = Console.ReadLine();

Console.Write("Enter Student Name:");

string name = Console.ReadLine();

Console.Write("Enter Student Email:");

string email = Console.ReadLine();

Console.Write("Enter Student Phone:");

string phone = Console.ReadLine();

Console.Write("Enter Student Details:");

string details = Console.ReadLine();

AddNewStudent(id, name, email, phone, details).Wait();

}

else

{

if (ans == 4)

{

Console.Write("Enter the student Code: ");

string id = Console.ReadLine();

Console.WriteLine("--------------------Enter the Student's Information--------------------");

Console.Write("Enter student Name:");

string name = "";

name = Console.ReadLine();

Console.Write("Enter studentEmail:");

string email = "";

email = Console.ReadLine();

Console.Write("Enter student Phone:");

string phone = "";

phone = Console.ReadLine();

Console.Write("Enter student details :");

string details = "";

details = Console.ReadLine();

UpdateStudent(id, name, email, phone, details).Wait();

}

else

{

Console.Write("Enter the student code: ");

string id = Console.ReadLine();

DeleteStudent(id).Wait();

}

}

}

}

Console.Write("Do you want to continue: ");

i = Console.ReadLine();

} while (i == "y" || i == "Y");

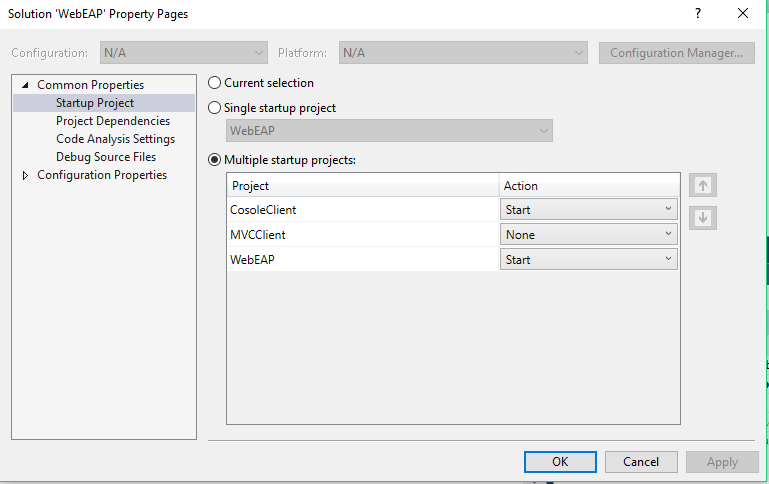
Console.ReadLine();

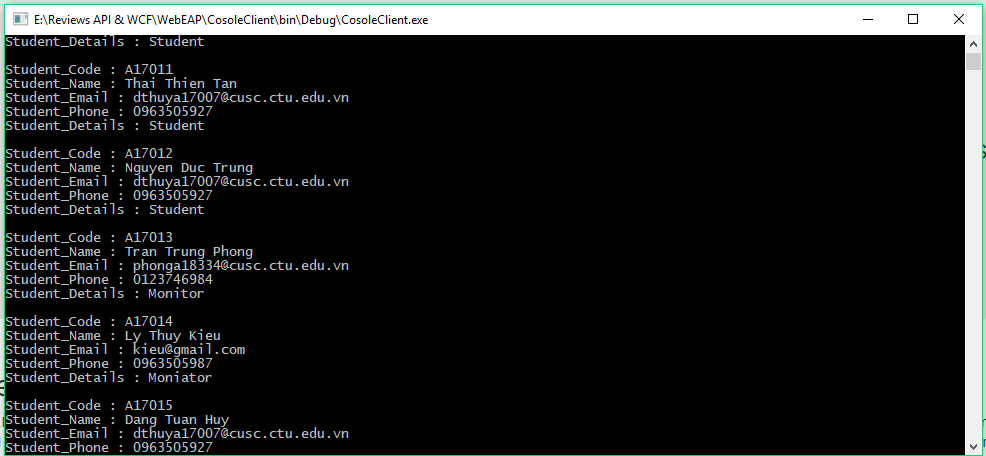
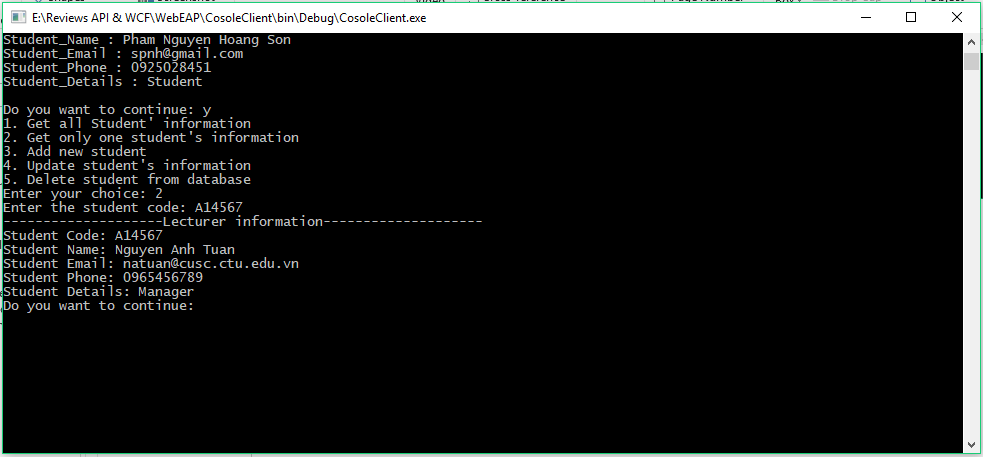
}

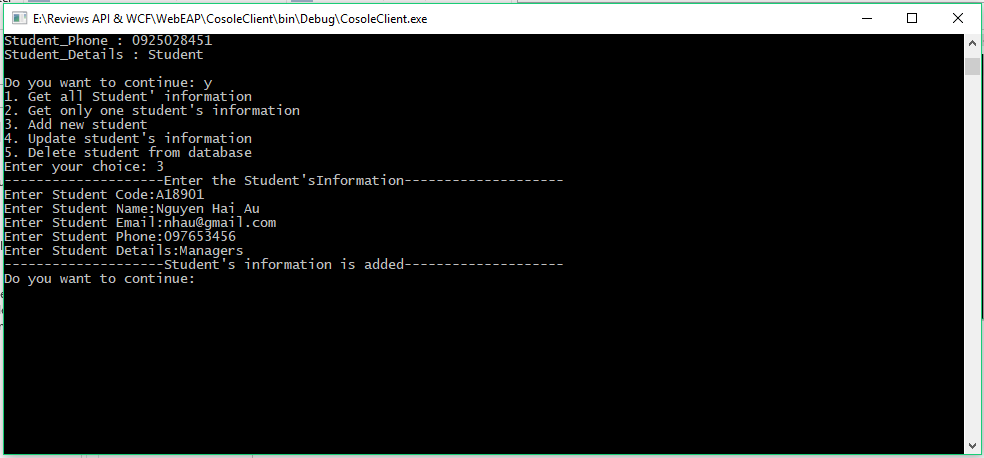
}

}

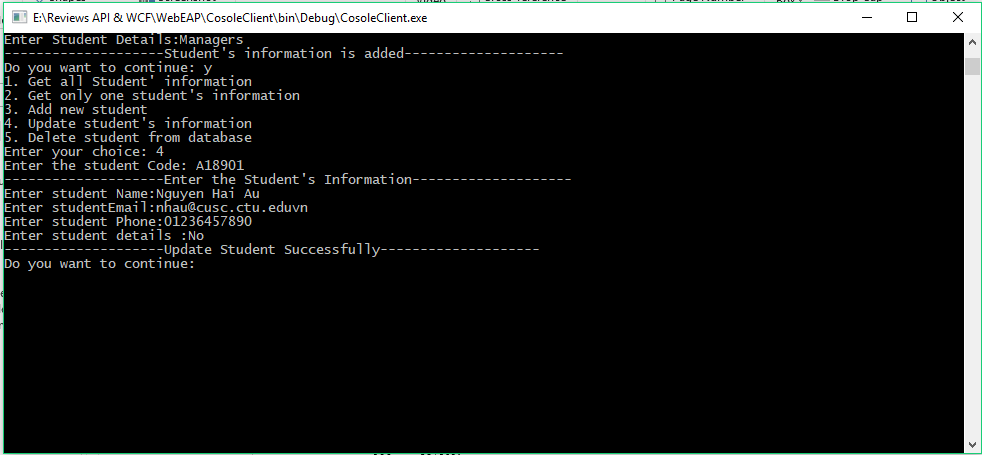
1. **Kết quả:**

**Config App:**

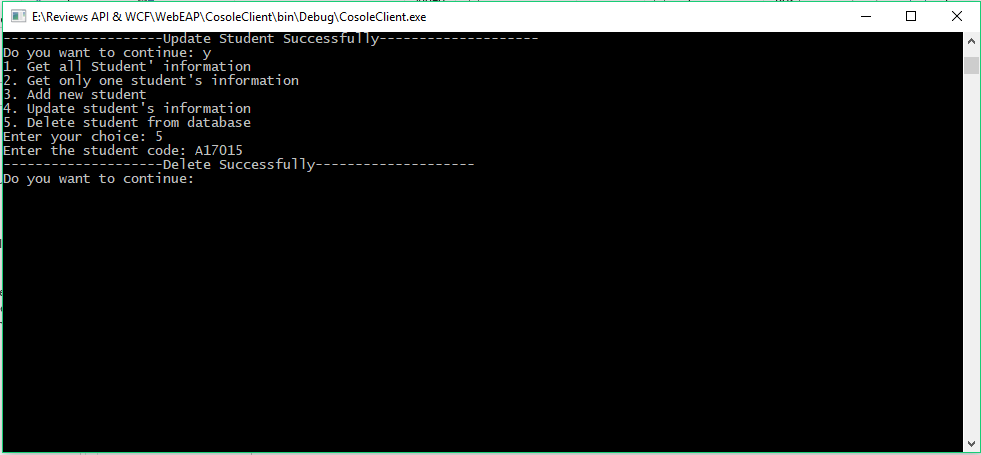
* **Danh sách toàn bộ**
* **Tìm theo ID**
* **Thêm**

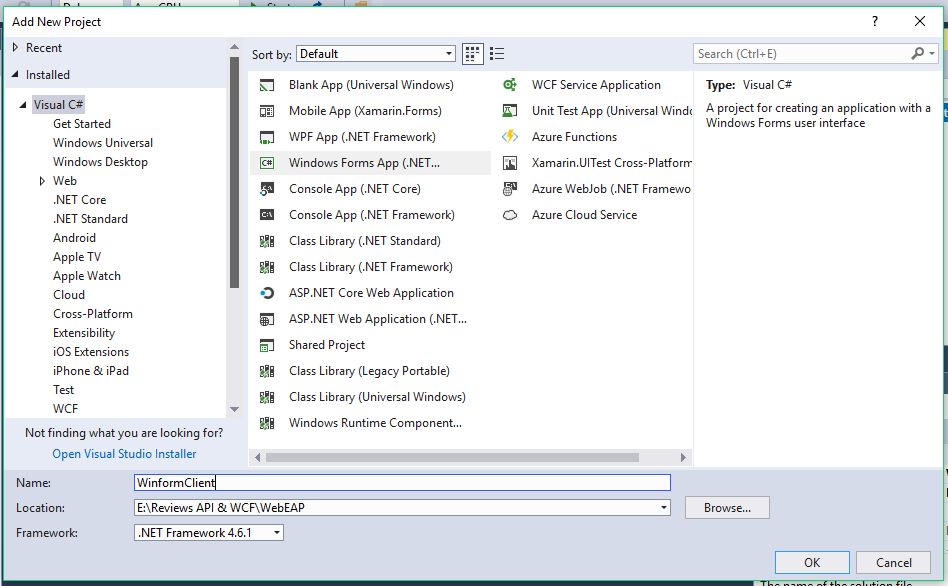
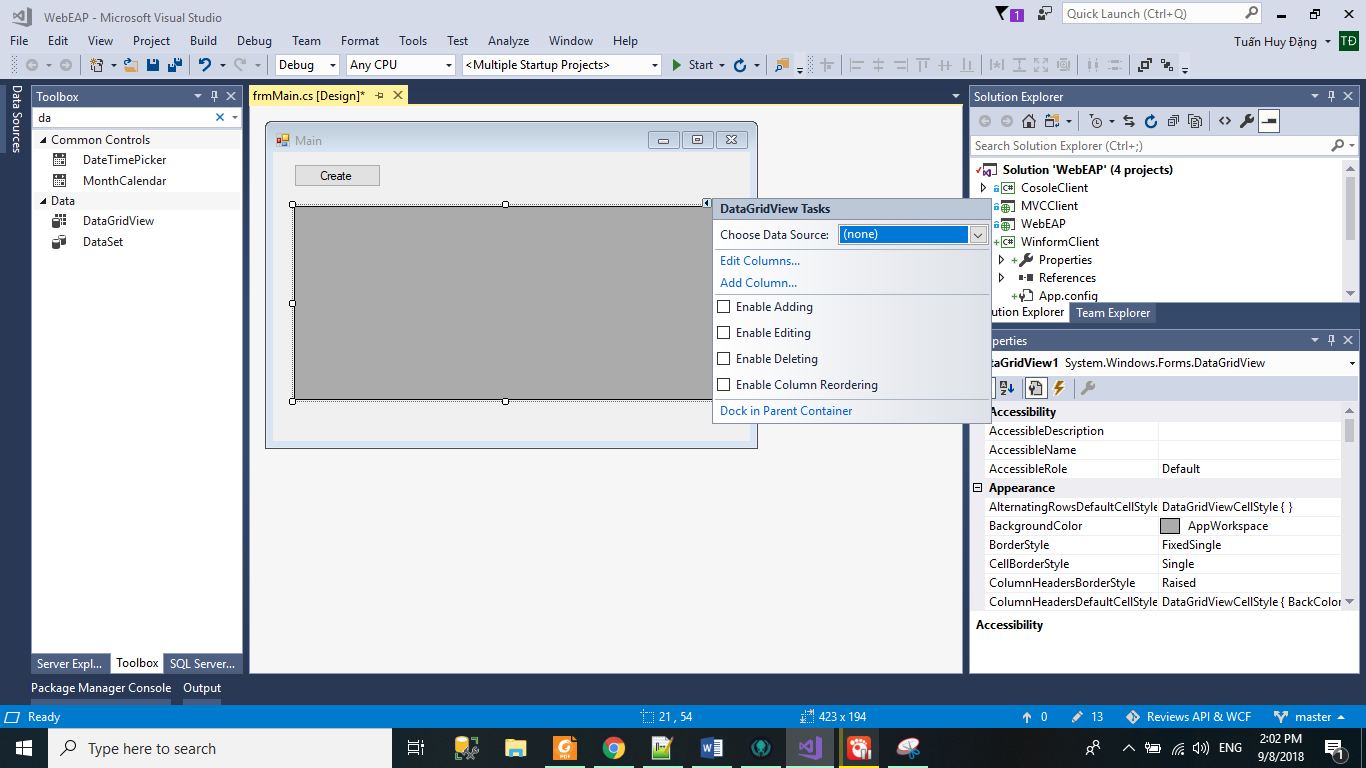


* **Sửa**

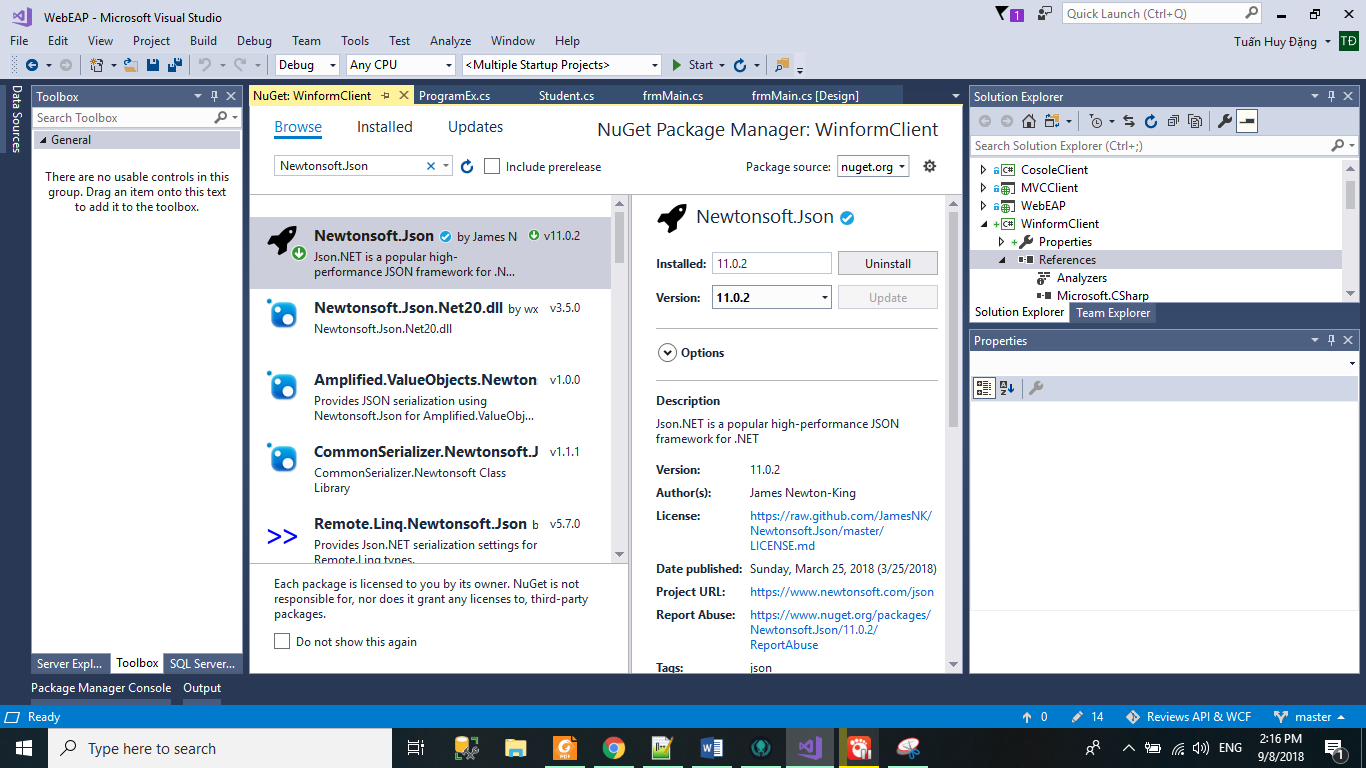
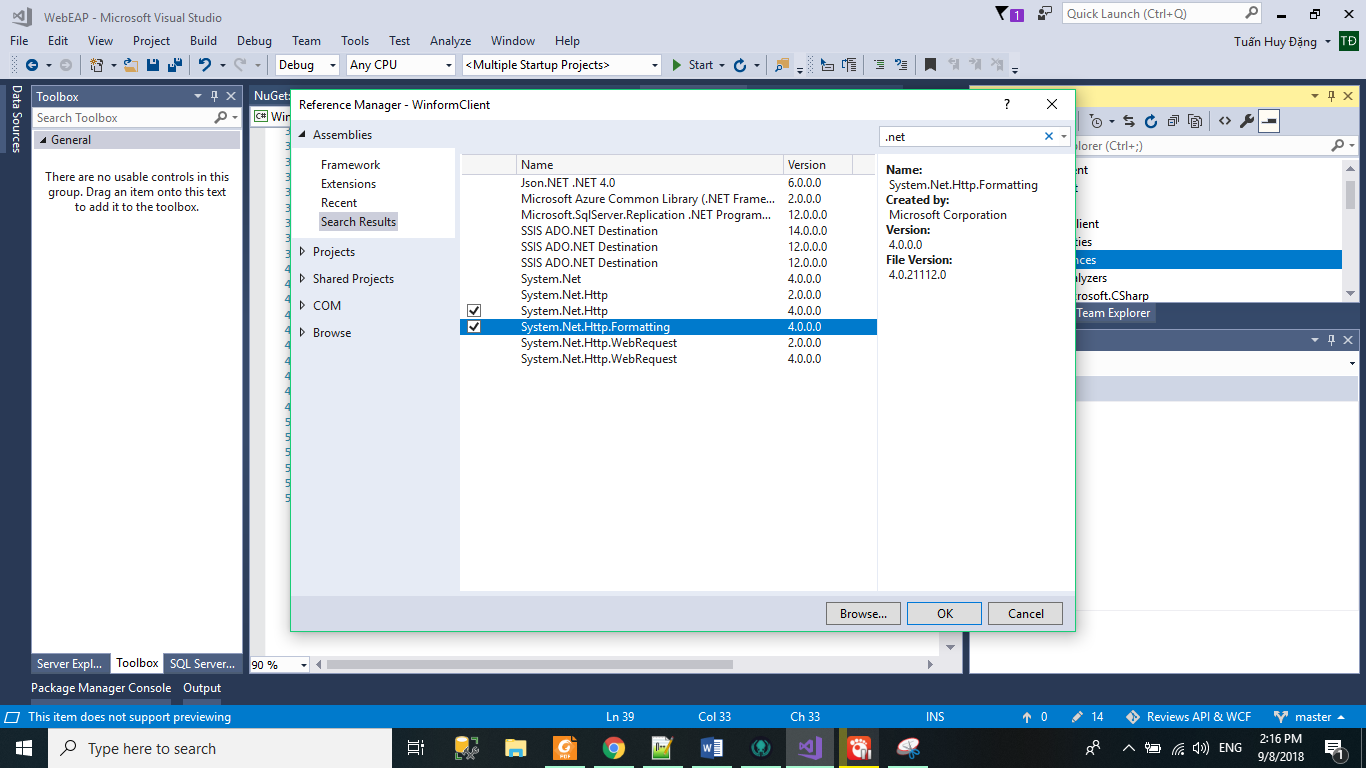


* **Xóa**



1. **WindowForm – Client**
2. **Trong cùng một solution tạo một Project Window Form Application**
3. **Tạo form sau**

**Button : Name btnCreate**

1. **Install Pakage**
2. **Config**
3. **Trong Project Winform tạo Class Student.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WinformClient

{

public class Student

{

public string Student\_Code { get; set; }

public string Student\_Name { get; set; }

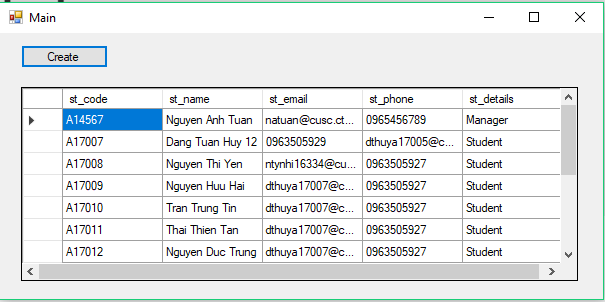
public string Student\_Email { get; set; }

public string Student\_Phone { get; set; }

public string Student\_Details { get; set; }

}

}

1. **Tạo frmMain có giao diện như trong hình**
2. **Bổ sung code sự kiện và gọi API cho frmMain**

using Newtonsoft.Json.Linq;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WinformClient

{

public partial class frmMain : Form

{

public frmMain()

{

InitializeComponent();

}

private void frmMain\_Load(object sender, EventArgs e)

{

GetAll();

}

public async Task GetAll()

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url);

// Chỉnh sửa và Fill Form

DataTable table = new DataTable();

table.Columns.Add("st\_code");

table.Columns.Add("st\_name");

table.Columns.Add("st\_email");

table.Columns.Add("st\_phone");

table.Columns.Add("st\_details");

//Kết thúc

if (response.IsSuccessStatusCode)

{

var responseData = response.Content.ReadAsStringAsync().Result;

JArray parsed = JArray.Parse(responseData.ToString());

Console.WriteLine("------------------------List of Student ' Information-------------");

foreach (var pair in parsed)

{

//dùng DataRow

DataRow row = table.NewRow(); //QT

JObject obj = JObject.Parse(pair.ToString());

int count = 0; //QT

foreach (var s in obj)

{

//Console.WriteLine("{0} : {1}", s.Key, s.Value.ToString().Trim());

row[count++] = s.Value.ToString();

}

// Console.WriteLine();

table.Rows.Add(row);

}

dataGridView1.DataSource = table;

}

}

private void btnCreate\_Click(object sender, EventArgs e)

{

frmCreate frm = new frmCreate();

frm.Show();

this.Hide();

}

private void dataGridView1\_CellDoubleClick(object sender, DataGridViewCellEventArgs e)

{

int index = dataGridView1.CurrentCell.RowIndex;

Student st = new Student()

{

Student\_Code = dataGridView1.Rows[index].Cells[0].Value.ToString(),

Student\_Name = dataGridView1.Rows[index].Cells[1].Value.ToString(),

Student\_Email = dataGridView1.Rows[index].Cells[2].Value.ToString(),

Student\_Phone = dataGridView1.Rows[index].Cells[3].Value.ToString(),

Student\_Details = dataGridView1.Rows[index].Cells[4].Value.ToString()

};

frmEdit frm = new frmEdit(st);

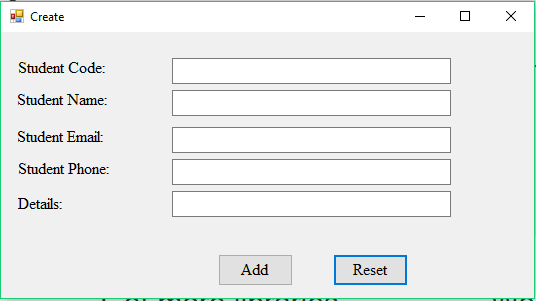
frm.Show();

this.Hide();

}

}

}

1. **Tạo frmCreate như giao diện**
2. **Bổ sung code gọi API và sự kiện cho frmCreate**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WinformClient

{

public partial class frmCreate : Form

{

public frmCreate()

{

InitializeComponent();

}

private void frmCreate\_FormClosed(object sender, FormClosedEventArgs e)

{

frmMain frm = new frmMain();

frm.Show();

}

private void btnReset\_Click(object sender, EventArgs e)

{

txtStudentCode.Text = "";

txtName.Text = "";

txtPhone.Text = "";

txtEmail.Text = "";

txtDetails.Text = "";

}

private void btnAdd\_Click(object sender, EventArgs e)

{

AddNewStudent(txtStudentCode.Text, txtName.Text, txtPhone.Text, txtEmail.Text, txtDetails.Text);

}

static async Task AddNewStudent(string id, string name, string email, string phone, string details)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url);

Student st = new Student()

{

Student\_Code = id,

Student\_Name = name,

Student\_Email = email,

Student\_Phone = phone,

Student\_Details = details

};

response = await client.PostAsJsonAsync(url, st);

if (response.IsSuccessStatusCode)

{

MessageBox.Show("Ok", "Infor", MessageBoxButtons.OK, MessageBoxIcon.Information);

frmMain frm = new frmMain();

frm.Show();

}

else

{

MessageBox.Show("Fail", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

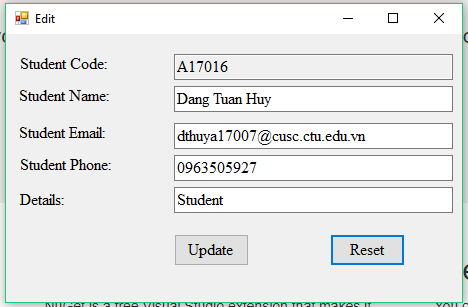
}

}

}

}

1. **Thiết kế formEdit**



1. **Bổ sung code gọi API và sự kiệt cho frmEdit**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WinformClient

{

public partial class frmEdit : Form

{

public frmEdit()

{

InitializeComponent();

}

public frmEdit(Student st)

{

InitializeComponent();

txtStudentCode.Text = st.Student\_Code;

txtName.Text = st.Student\_Name;

txtEmail.Text = st.Student\_Email;

txtPhone.Text = st.Student\_Phone;

txtDetails.Text = st.Student\_Details;

}

private void btnReset\_Click(object sender, EventArgs e)

{

txtName.Text = "";

txtPhone.Text = "";

txtEmail.Text = "";

txtDetails.Text = "";

}

static async Task UpdateStudent(string id, string name, string email, string phone, string details)

{

string url = "http://localhost:50180/api/Students";

var client = new HttpClient();

client.BaseAddress = new Uri(url);

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

HttpResponseMessage response = await client.GetAsync(url + "/" + id);

if (response.IsSuccessStatusCode)

{

Student student = await response.Content.ReadAsAsync<Student>();

if (name != "")

student.Student\_Name = name;

if (email != "")

student.Student\_Email = email;

if (phone != "")

student.Student\_Phone = phone;

if (details != "")

student.Student\_Details = details;

response = await client.PutAsJsonAsync(url + "/" + id, student);

if (response.IsSuccessStatusCode)

{

MessageBox.Show("Ok", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

frmMain frm = new frmMain();

frm.Show();

}

else

{

MessageBox.Show("Ok", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

UpdateStudent(txtStudentCode.Text, txtName.Text, txtPhone.Text, txtEmail.Text, txtDetails.Text);

}

private void frmEdit\_FormClosing(object sender, FormClosingEventArgs e)

{

frmMain frm = new frmMain();

frm.Show();

this.Hide();

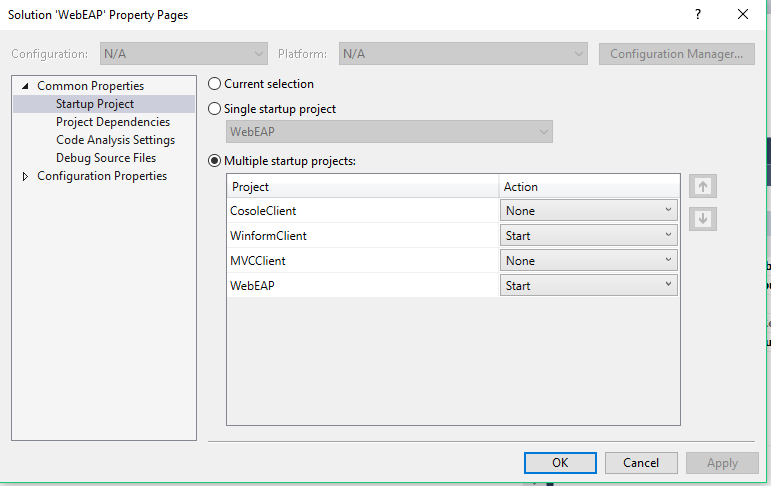
}

}

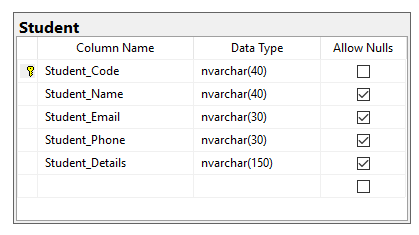
}

1. **Config**

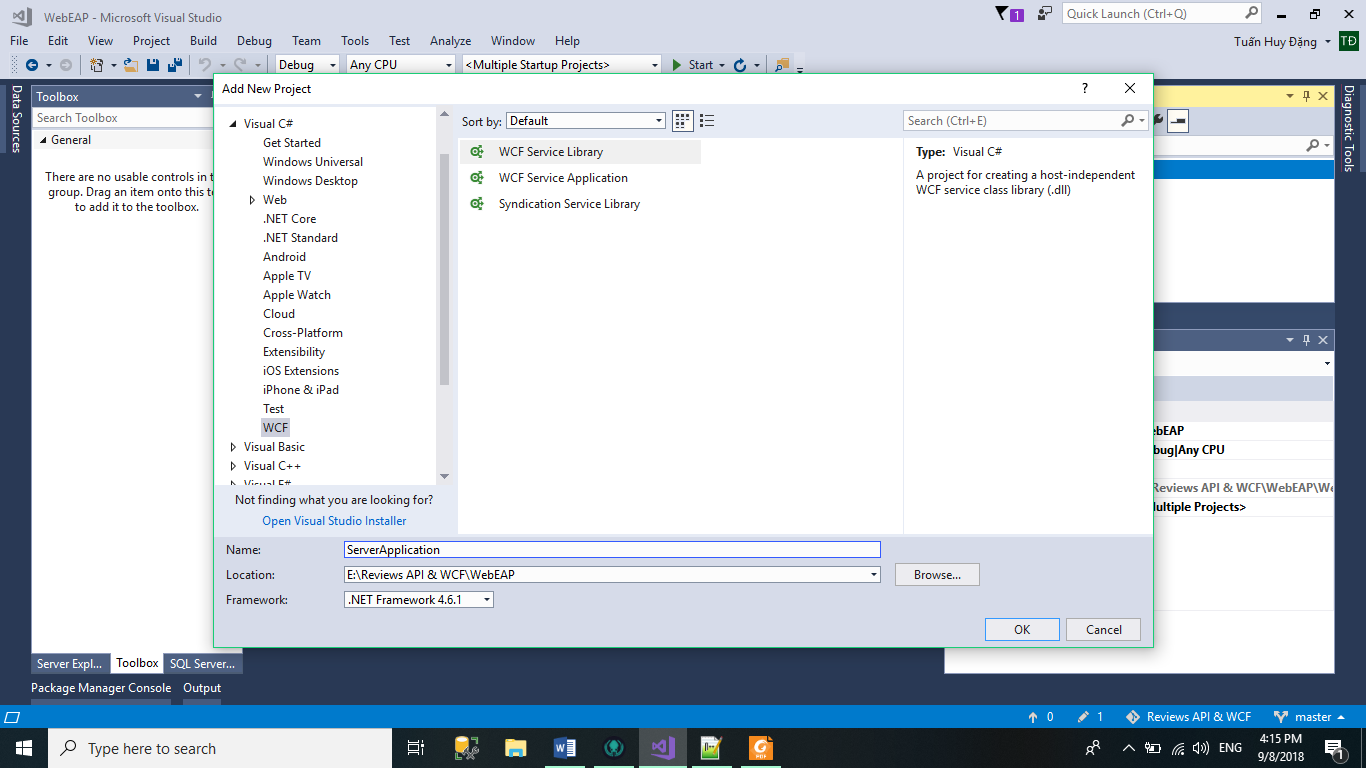
**Cấu hình API và winform chạy cùng lúc**

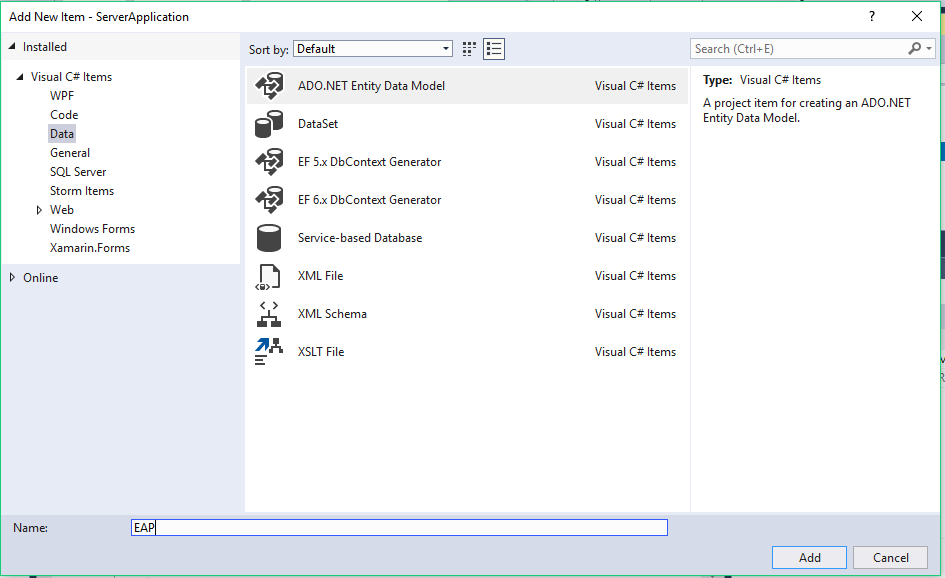


1. **WCF**

**Sử dụng lại CSDL**

**Tạo WCF Service**

**Bước 1: Tạo Project**

**Bước 2: Sử dụng Entity Framework**

**Bước 3: Bổ sung code trong Iservices1.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.Text;

namespace ServerApplication

{

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService1" in both code and config file together.

[ServiceContract]

public interface IService1

{

//Bổ sung

[OperationContract]

List<Student> GetStudents();

[OperationContract]

Student GetStudent(string id);

[OperationContract]

List<Student> FindStudent(string id);

[OperationContract]

bool AddNewStudent(Student student);

[OperationContract]

bool UpdateStudent(Student student);

[OperationContract]

bool DeleteStudent(string id);

}

}

**Bước 4: Implement Service1 và bổ sung code**

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.Text;

namespace ServerApplication

{

public class Service1 : IService1

{

EAPEntities db = new EAPEntities();

public List<Student> GetStudents()

{

return db.Students.ToList();

}

public Student GetStudent(string id)

{

return db.Students.Find(id);

}

public List<Student> FindStudent(string name)

{

var students = from s in db.Students select s;

students = students.Where(s => s.Student\_Name.Contains(name));

students = students.OrderBy(s => s.Student\_Code);

return students.ToList();

}

public bool AddNewStudent(Student student)

{

try

{

db.Students.Add(student);

db.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

public bool UpdateStudent(Student student)

{

try

{

db.Entry(student).State = EntityState.Modified;

db.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

public bool DeleteStudent(string id)

{

try

{

Student student = db.Students.Find(id);

db.Students.Remove(student);

return true;

}

catch (Exception)

{

return false;

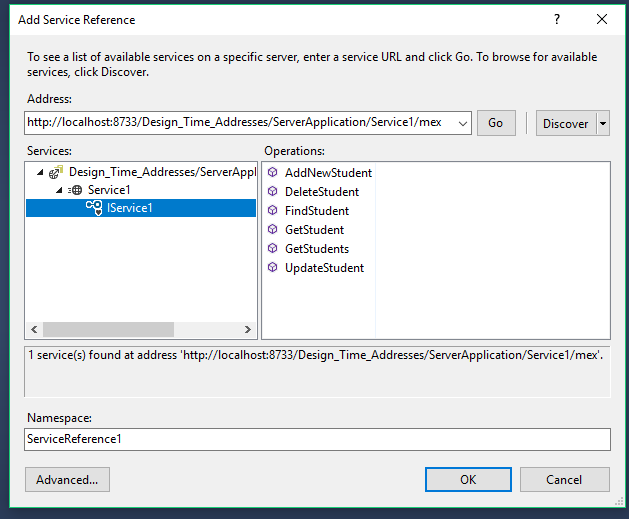
}

}

}

}

1. **WCF – WebMVC**
2. **Trong cùng solution tạo Project ASP.NET MVC WCF\_MVC**
3. **Discover Service**



1. **Tạo Empy Controller (StudentController)**
2. **Bổ sung code cho Controller vừa tạo**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using WCF\_MVC.ServiceReference1;

namespace WCF\_MVC.Controllers

{

public class StudentController : Controller

{

// GET: Student

Service1Client db = new Service1Client();

public ActionResult Index()

{

return View(db.GetStudents());

}

public ActionResult Edit(string id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Student stu = db.GetStudent(id);

if (stu == null)

{

return HttpNotFound();

}

return View(stu);

}

public ActionResult Create()

{

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "Student\_Code,Student\_Name,Student\_Email,Student\_Phone,Student\_Details")] Student student)

{

if (ModelState.IsValid)

{

bool b = db.AddNewStudent(student);

if (b)

{

return RedirectToAction("Index");

}

else

{

return View(student);

}

}

return View(student);

}

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "Student\_Code,Student\_Name,Student\_Email,Student\_Phone,Student\_Details")] Student student)

{

if (ModelState.IsValid)

{

bool b = db.UpdateStudent(student);

if (b)

{

return RedirectToAction("Index");

}

else

{

return View(student);

}

}

return View(student);

}

public ActionResult Delete(string id)

{

if(id==null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Student student = db.GetStudent(id);

if(student == null)

{

return HttpNotFound();

}

return View(student);

}

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(string id)

{

Student student = db.GetStudent(id);

db.DeleteStudent(id);

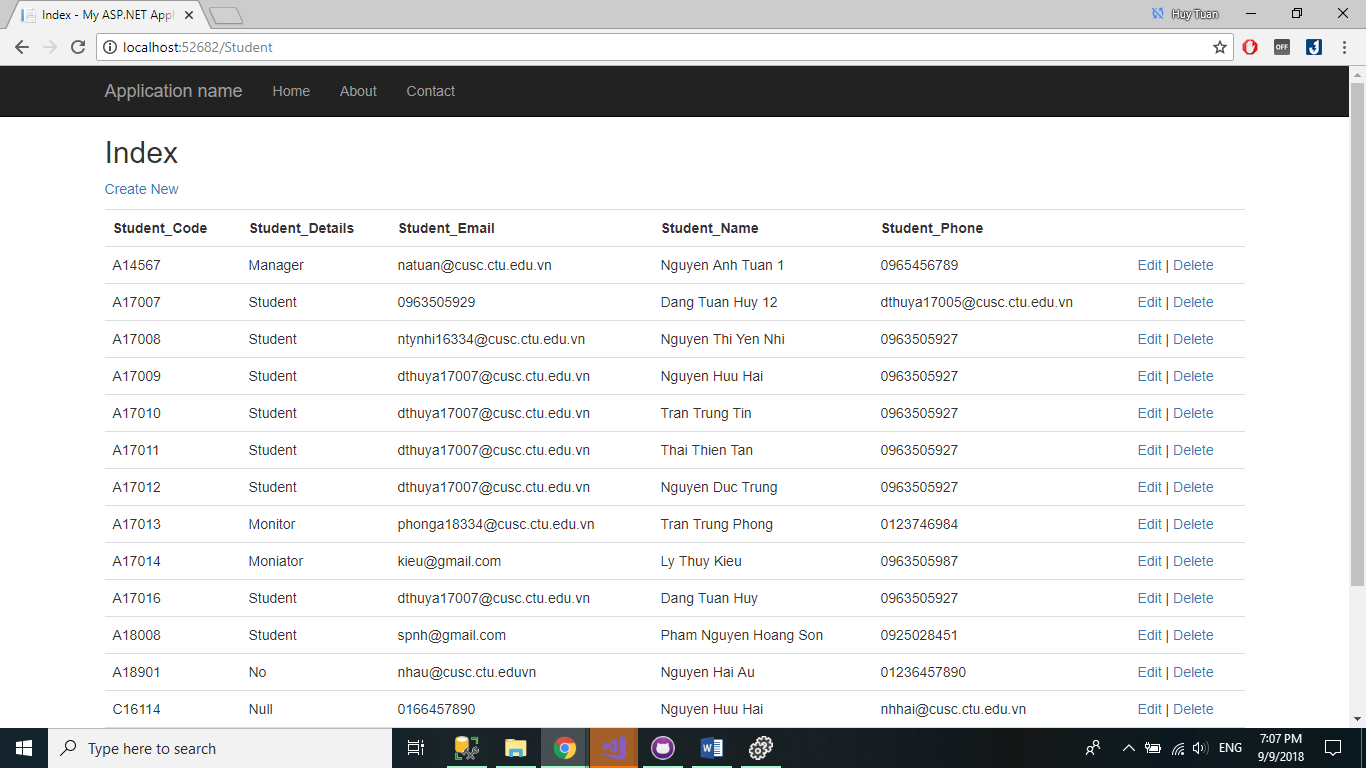
return RedirectToAction("Index");

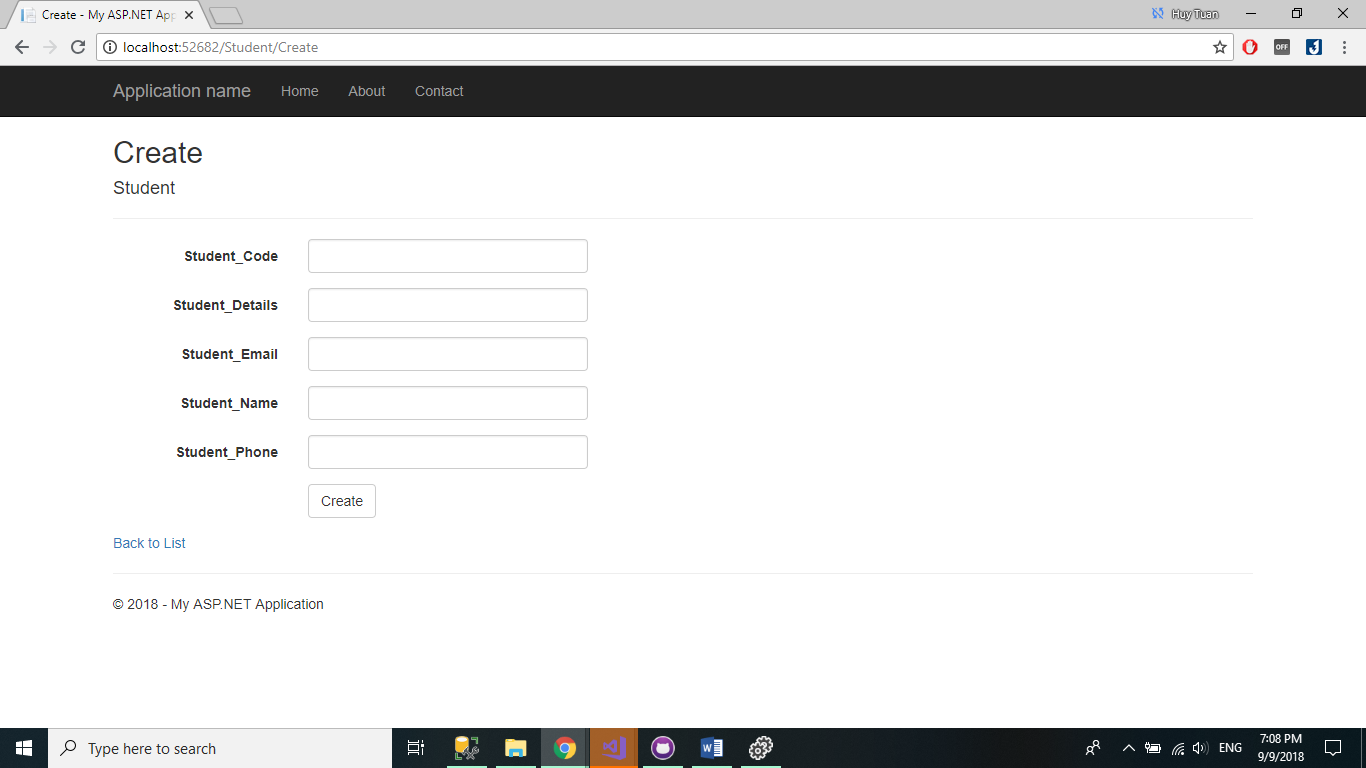
}

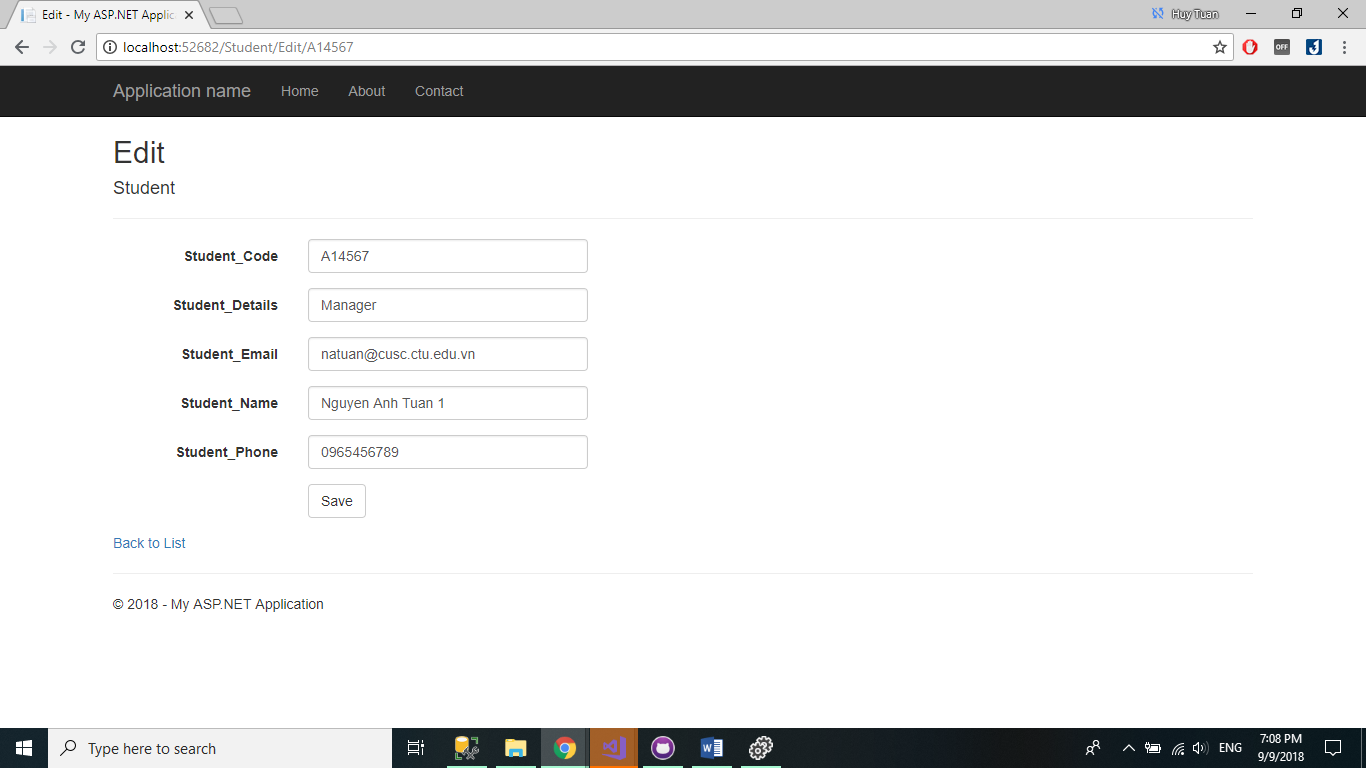
}

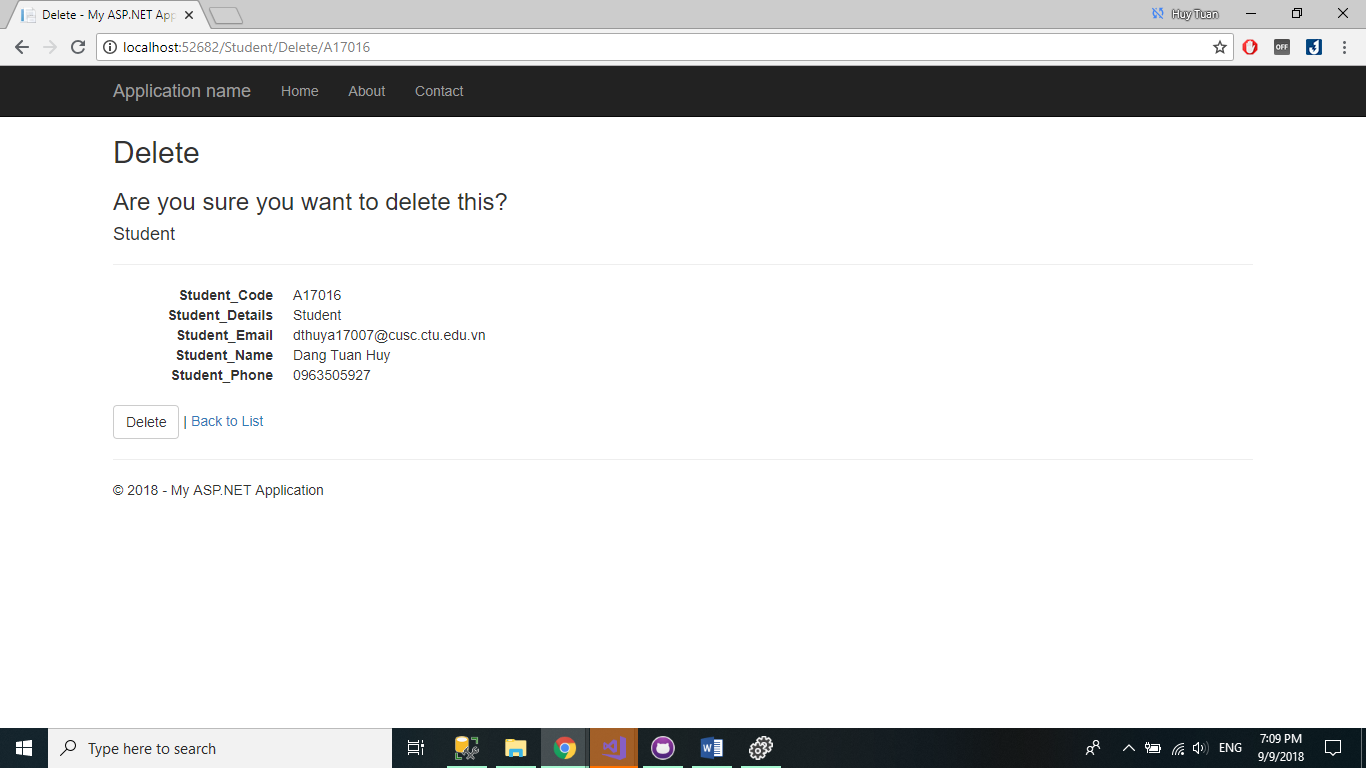
}

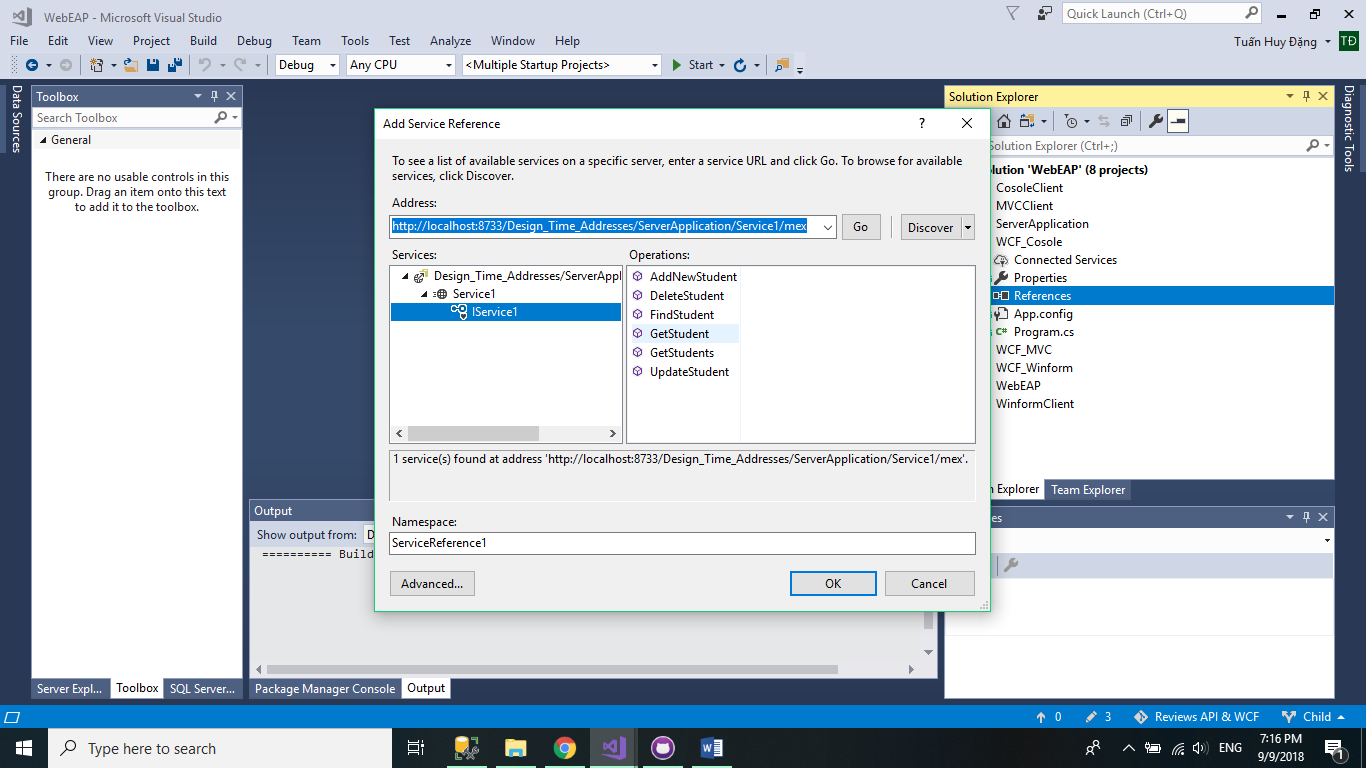
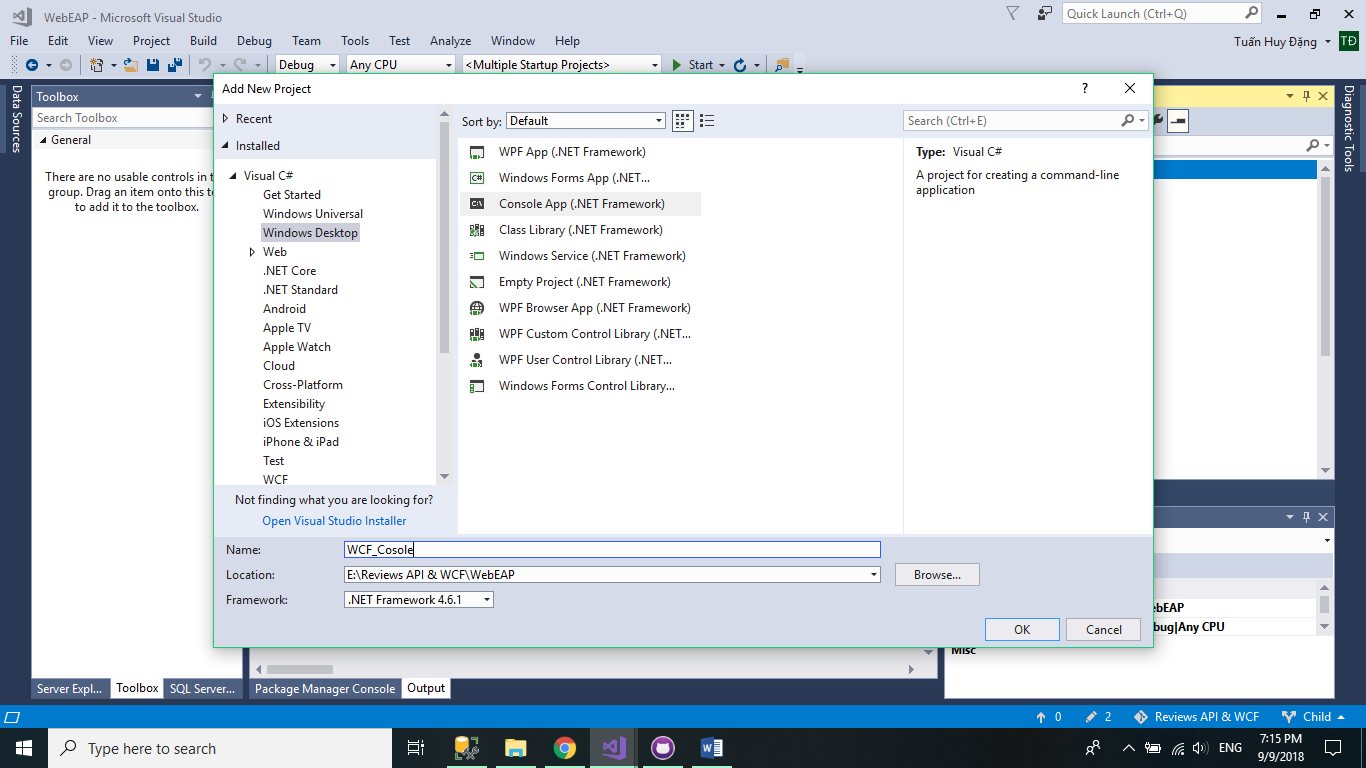
1. **Generate View tương ứng với mỗi actionmethod và xem kết quả**

* **Index**
* **Create**



* **Edit**
* **Delete**



1. **WCF - Cosole**
2. **Trong cùng solution tạo Project Cosole App WCF\_Cosole**
3. **Discover Service từ WCF Service**
4. **Trong Programs.cs bổ sung mã nguồn gọi service danh sách, thêm, sửa, xóa**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using WCF\_Cosole.ServiceReference1;

namespace WCF\_Cosole

{

class Program

{

static Service1Client db = new Service1Client();

//Hiển thị danh sách

public static void GetAll()

{

var data = db.GetStudents();

for (int i = 0; i < data.Length; i++)

{

Console.WriteLine("Students " + (i + 1) + ": ");

Console.WriteLine("Student Code: " + data[i].Student\_Code);

Console.WriteLine("Student Name: " + data[i].Student\_Name);

Console.WriteLine("Student Email: " + data[i].Student\_Email);

Console.WriteLine("Student Phone: " + data[i].Student\_Phone);

Console.WriteLine("Student Details" + data[i].Student\_Details);

}

}

//Tìm Kiếm

public static void GetStudent(string id)

{

var data = db.GetStudent(id);

Console.WriteLine("Student Code: " + data.Student\_Code);

Console.WriteLine("Student Name: " + data.Student\_Name);

Console.WriteLine("Student Email: " + data.Student\_Email);

Console.WriteLine("Student Phone: " + data.Student\_Phone);

Console.WriteLine("Student Details: " + data.Student\_Details);

}

public static void addStudent(ServiceReference1.Student student)

{

var add = db.AddNewStudent(student);

if (add)

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

}

public static void UpdateStudent(ServiceReference1.Student student)

{

var update = db.UpdateStudent(student);

if (update)

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

}

public static void DeleteStudent(string id)

{

var delete = db.DeleteStudent(id);

if (delete)

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

}

static void Main(string[] args)

{

reset:

Console.WriteLine("Select an option: ");

Console.WriteLine("1. Get all student's information");

Console.WriteLine("2. Get a student information");

Console.WriteLine("3. Update student information");

Console.WriteLine("4. Delete a student ");

Console.WriteLine("5. Add new student");

Console.Write("Your choice: ");

int c = Convert.ToInt32(Console.ReadLine());

switch (c)

{

case 1:

GetAll();

goto reset;

break;

case 2:

Console.Write("Enter Student Code: ");

// int i = Convert.ToInt32(Console.ReadLine());

string i = Console.ReadLine();

GetStudent(i);

goto reset;

break;

case 3:

ServiceReference1.Student students = new Student();

Console.Write("Student Code update: ");

//int id1 = Convert.ToInt32(Console.ReadLine());

string id1 = Console.ReadLine();

students.Student\_Code = id1;

Console.Write("Student Name update: ");

string name1 = Console.ReadLine();

students.Student\_Name = name1;

Console.Write("Student Phone update: ");

string phone1 = Console.ReadLine();

students.Student\_Phone = phone1;

Console.Write("Student Email update: ");

string email1 = Console.ReadLine();

students.Student\_Email = email1;

Console.Write("Student Details update: ");

string details = Console.ReadLine();

students.Student\_Details = details;

UpdateStudent(students);

goto reset;

break;

case 4:

Console.Write("Enter Student delete: ");

//int i2 = Convert.ToInt32(Console.ReadLine());

string i2 = Console.ReadLine();

DeleteStudent(i2);

goto reset;

break;

case 5:

ServiceReference1.Student student = new Student();

Console.Write("Student Code: ");

//int id = Convert.ToInt32(Console.ReadLine());

string id = Console.ReadLine();

student.Student\_Code = id;

Console.Write("Lecturer Name: ");

string name = Console.ReadLine();

student.Student\_Name = name;

Console.Write("Student Phone: ");

string phone = Console.ReadLine();

student.Student\_Phone = phone;

Console.Write("Student Email: ");

string email = Console.ReadLine();

student.Student\_Email = email;

Console.Write("Lecturer details: ");

string info = Console.ReadLine();

student.Student\_Details = info;

addStudent(student);

goto reset;

break;

default:

break;

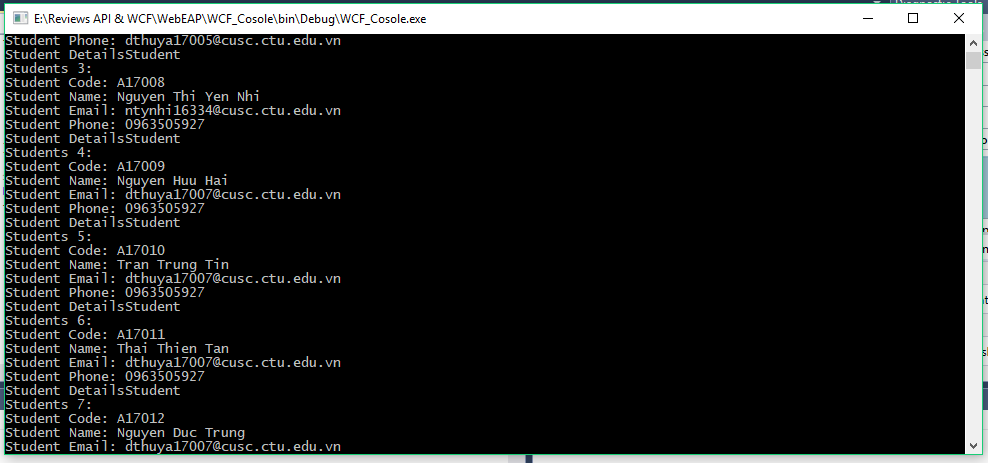
}

Console.ReadLine();

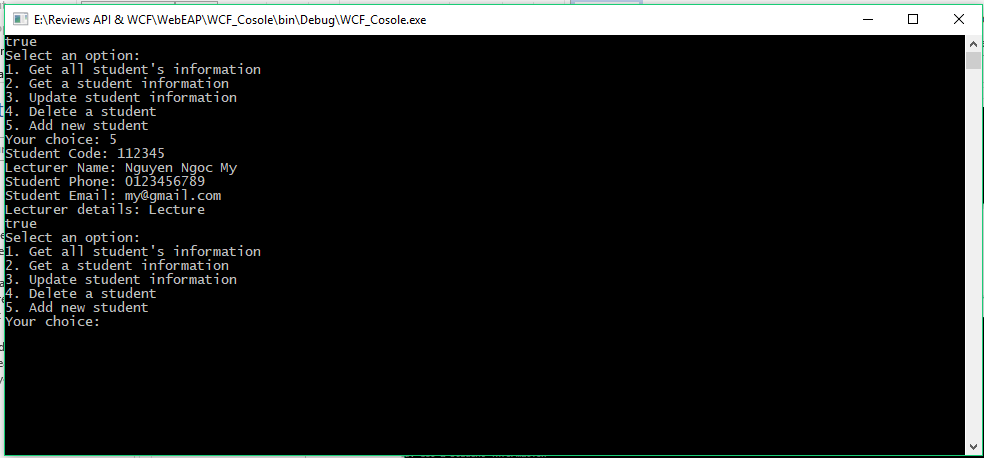
}

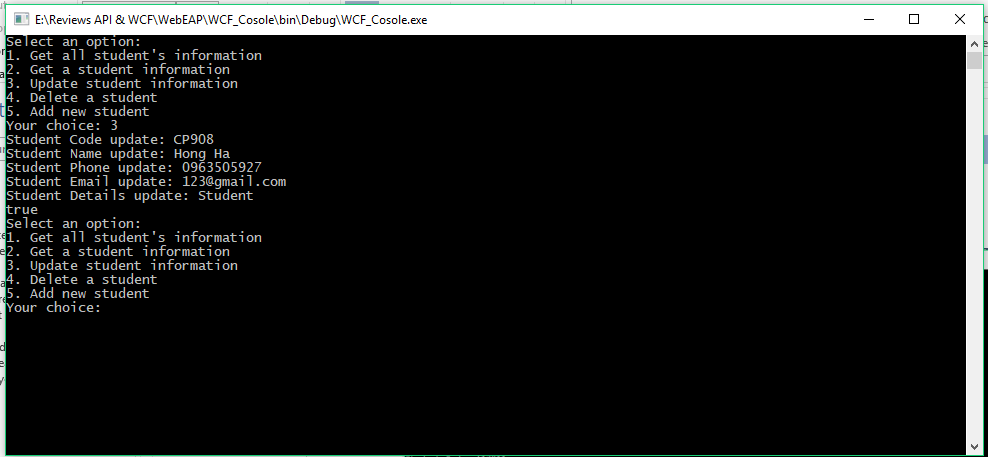
}

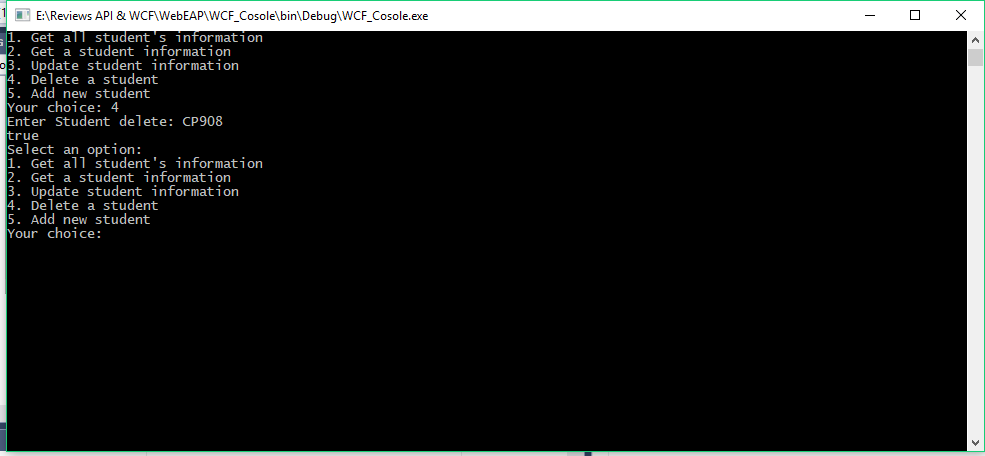
}

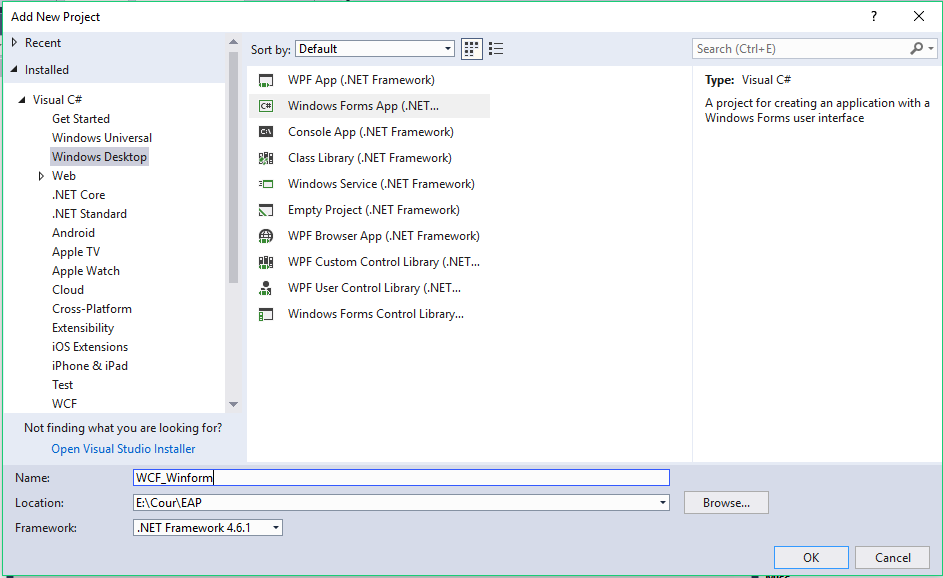
1. **Kết quả**

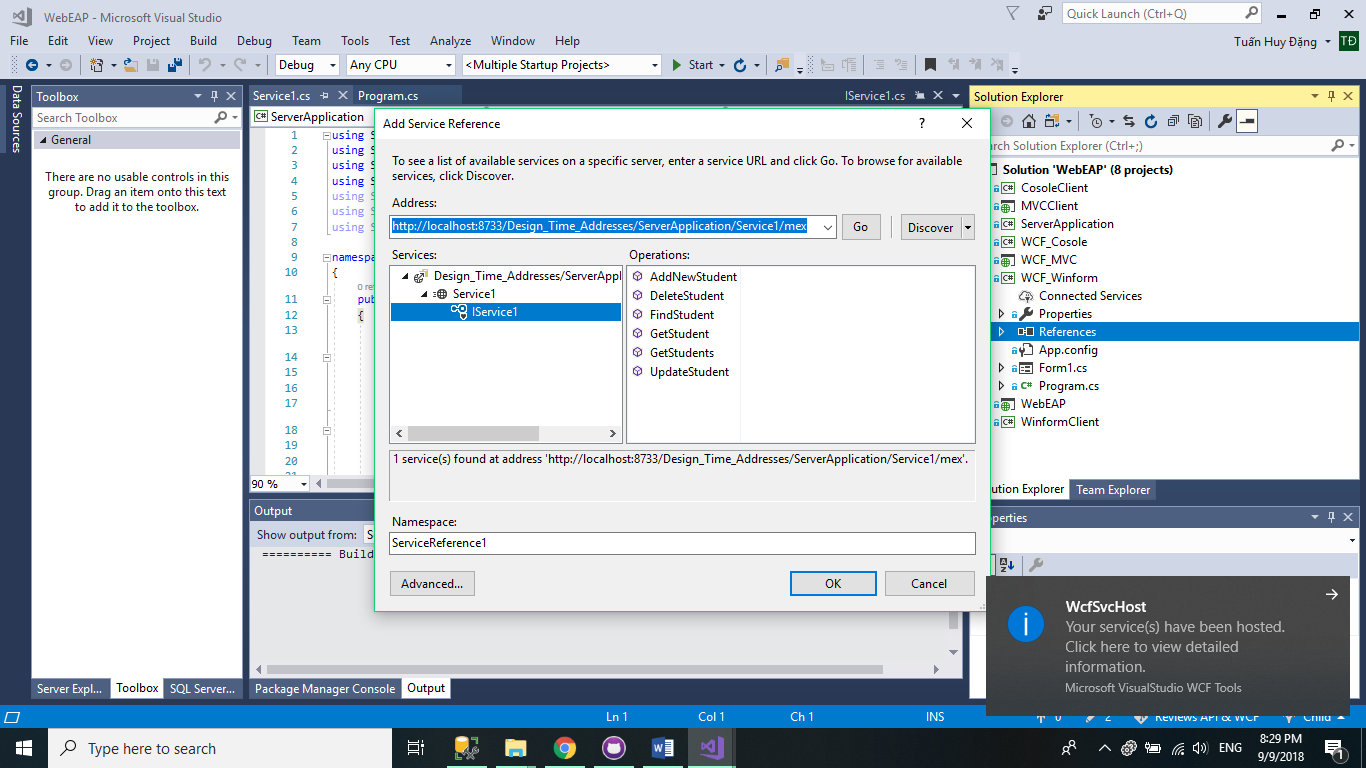
* **Danh sách**
* **Thêm**



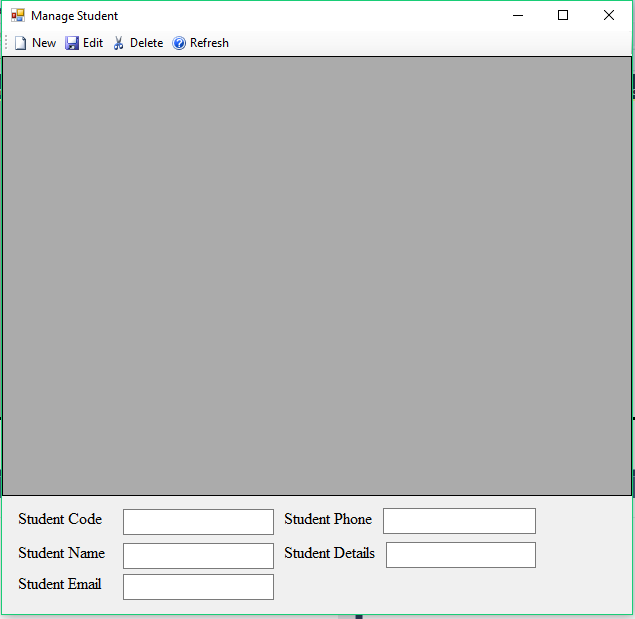
* **Sửa**
* **Xóa**



1. **WCF – Winform**
2. **Trong cùng 1 solution tạo Project Windows Form App**
3. **Discover với WCF Service**

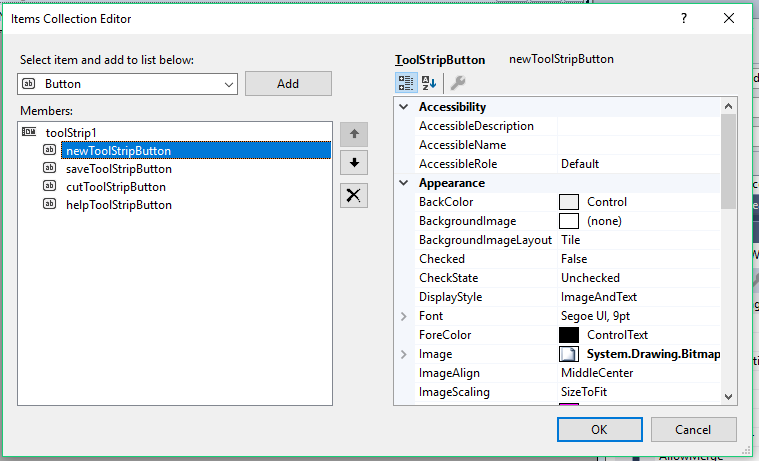


1. **Thiết Kế Form theo hình**



1. **Cấu hình Form**

* **Dùng ToolStrip**



* **Dùng SplitContainer**

**Horizontal Splitter Orientation**

* **DataGridView**

1. **Bắt sự kiện thêm sửa xóa, binding Data Source**