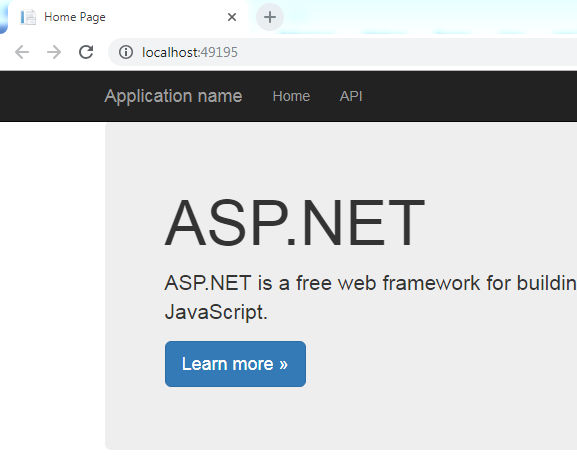
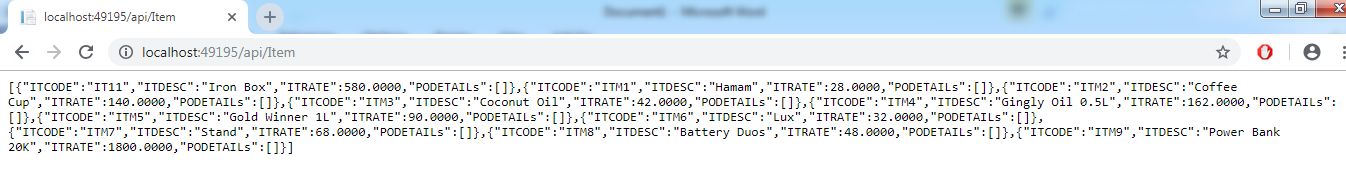
Assignment 23

MVC API Service Example

MVC service running in the server

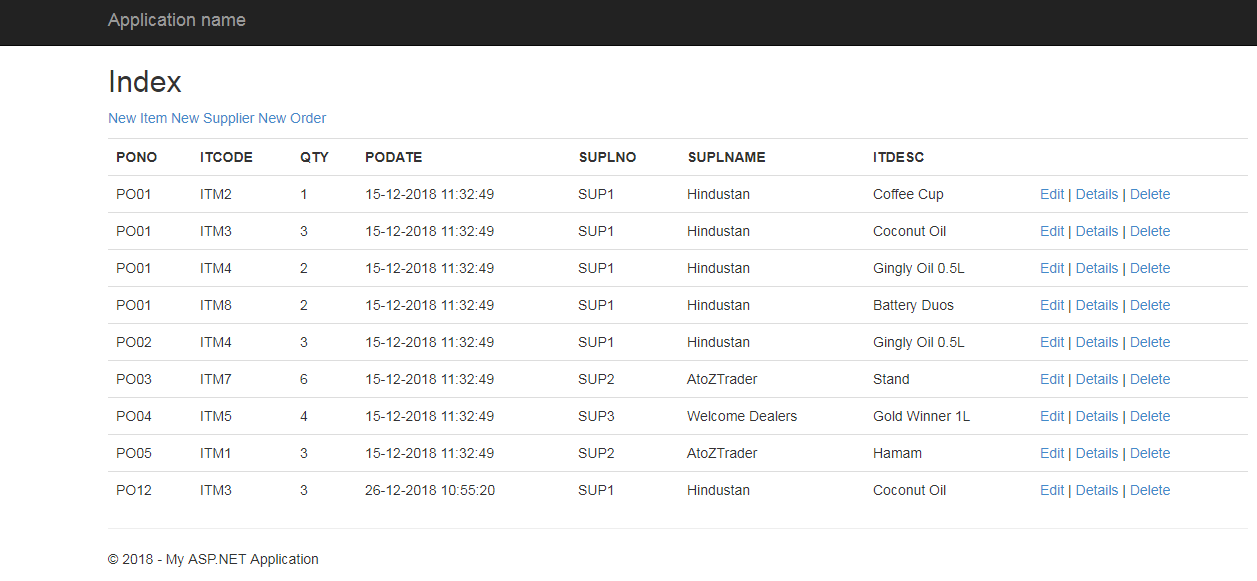


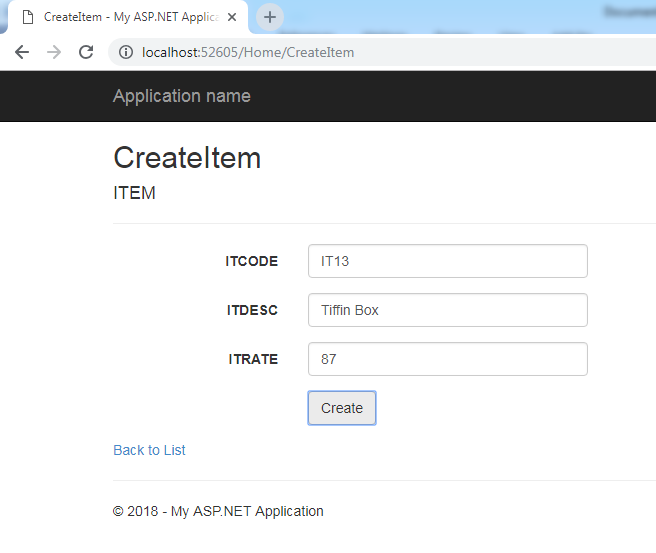
Shown Get example for Item

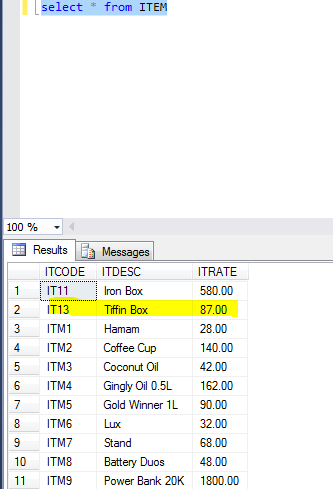


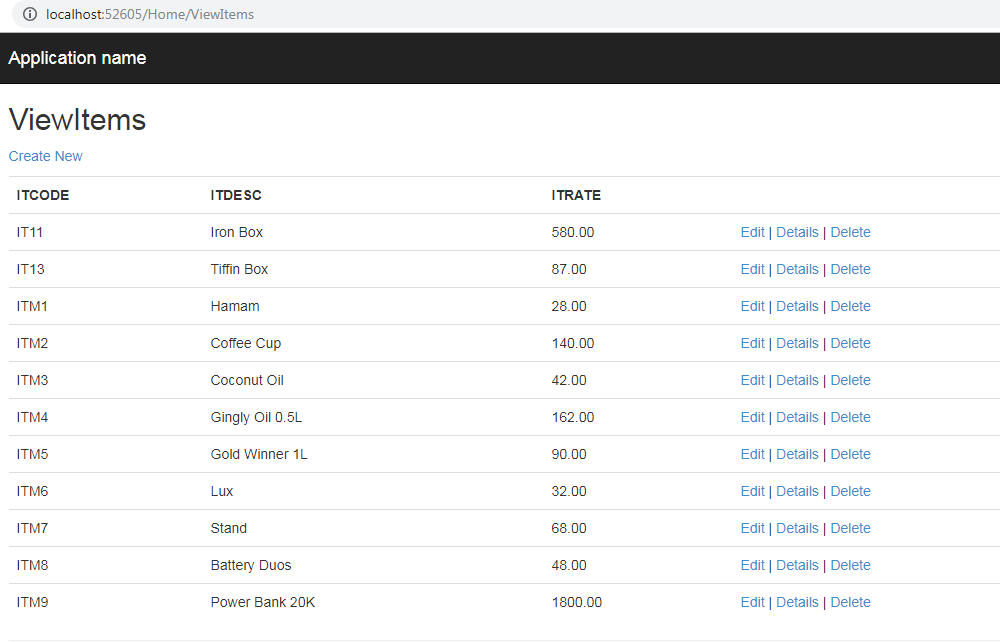
Client developed in order to consume the service

Output of those client shown below

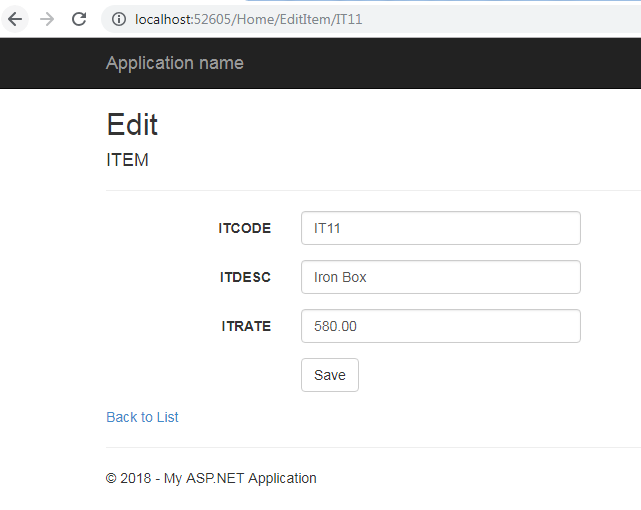


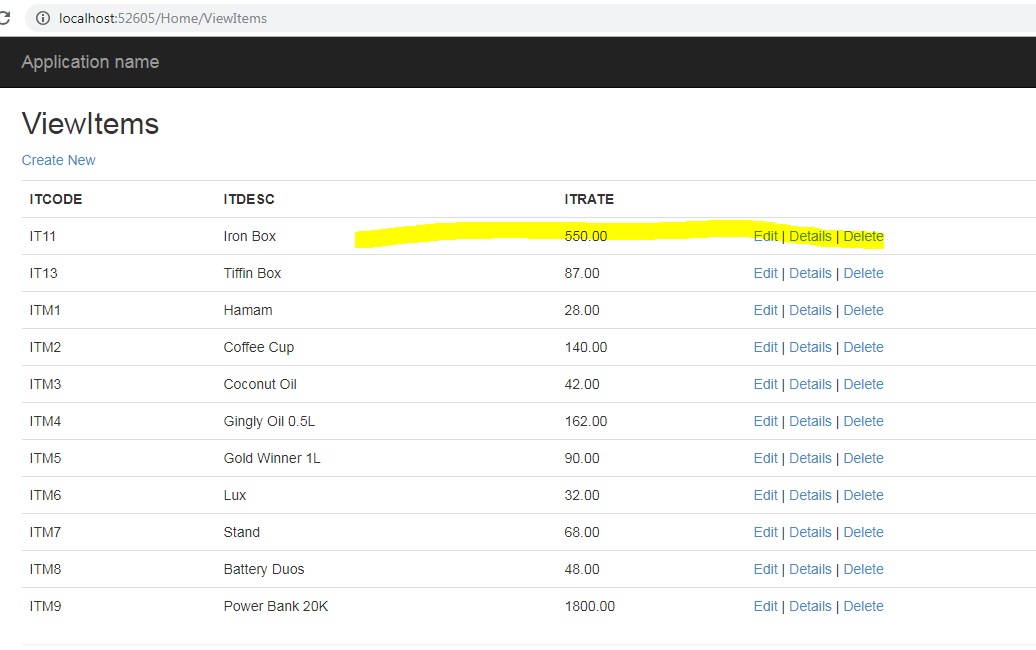




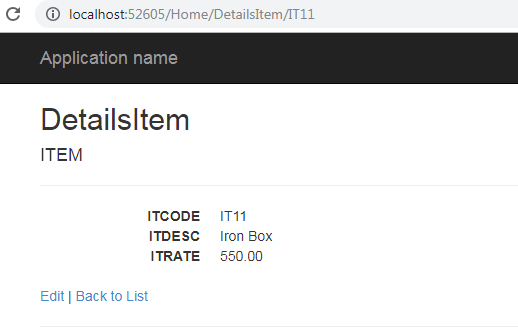


Edit Item

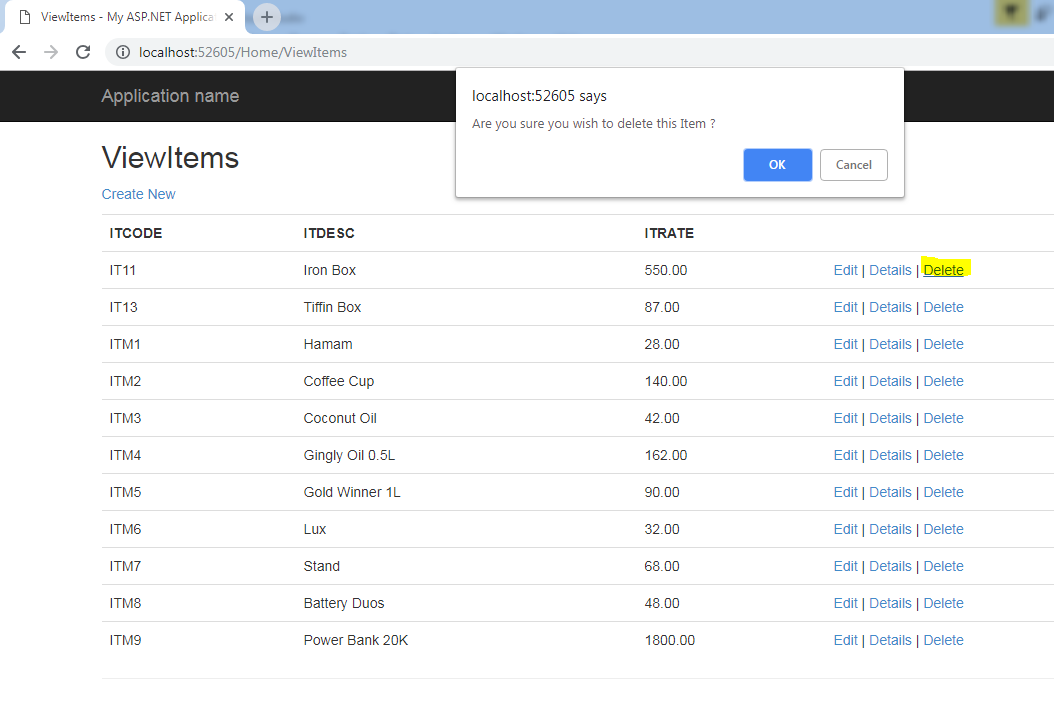


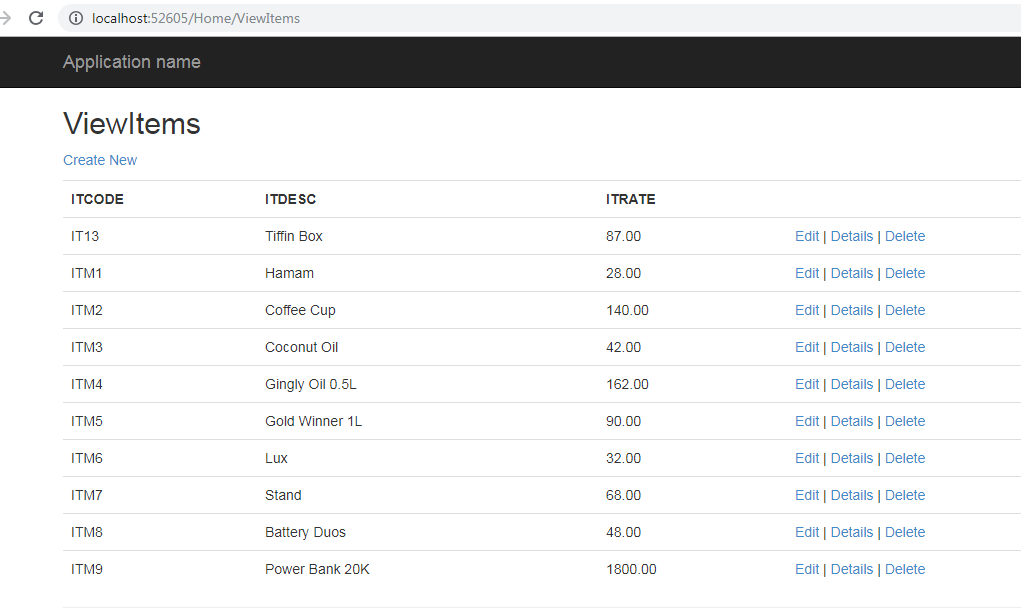


Details Item



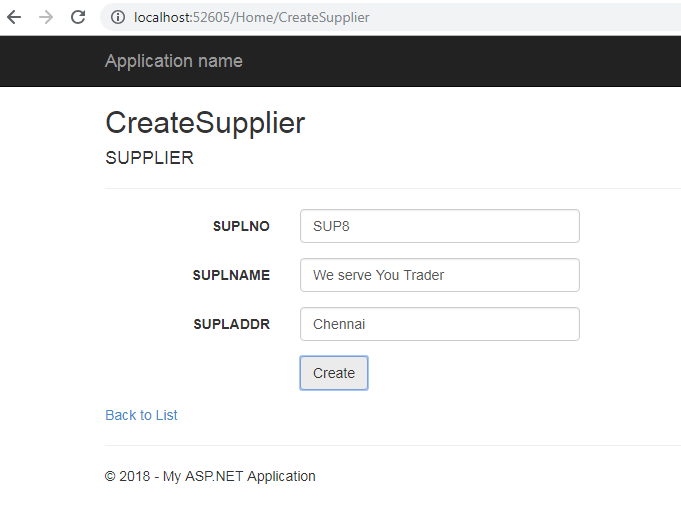
Delete Item

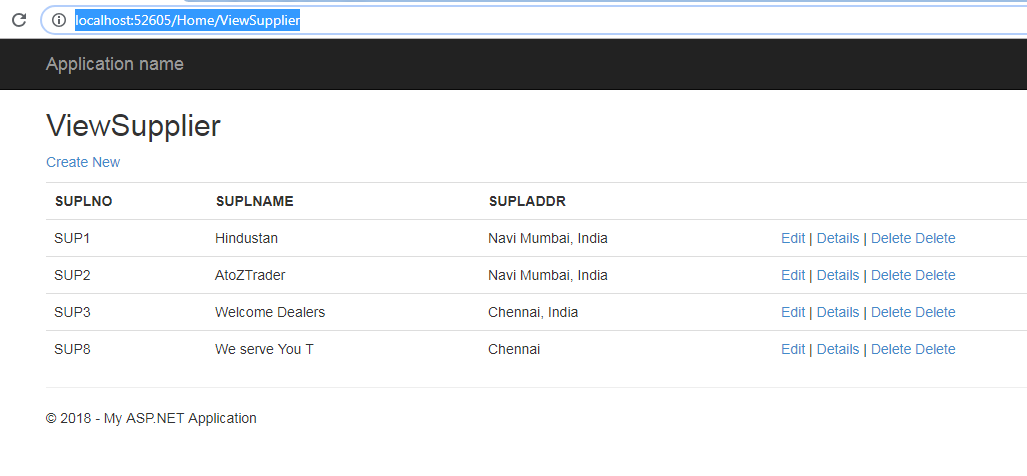




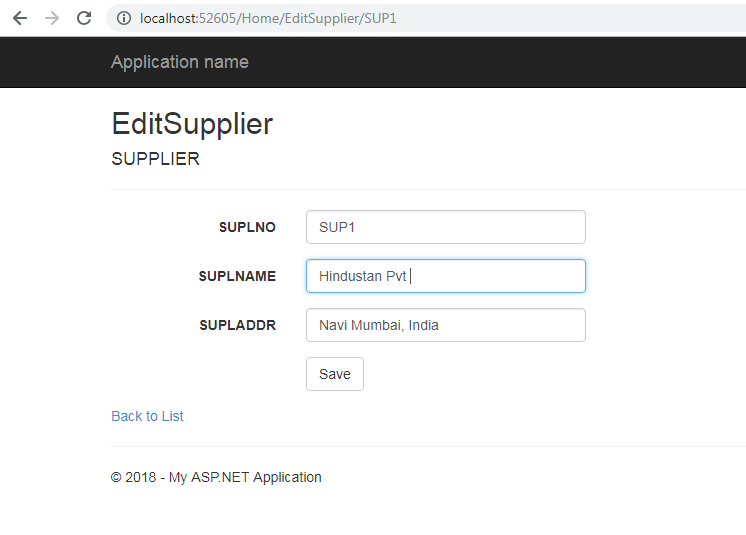
Supplier

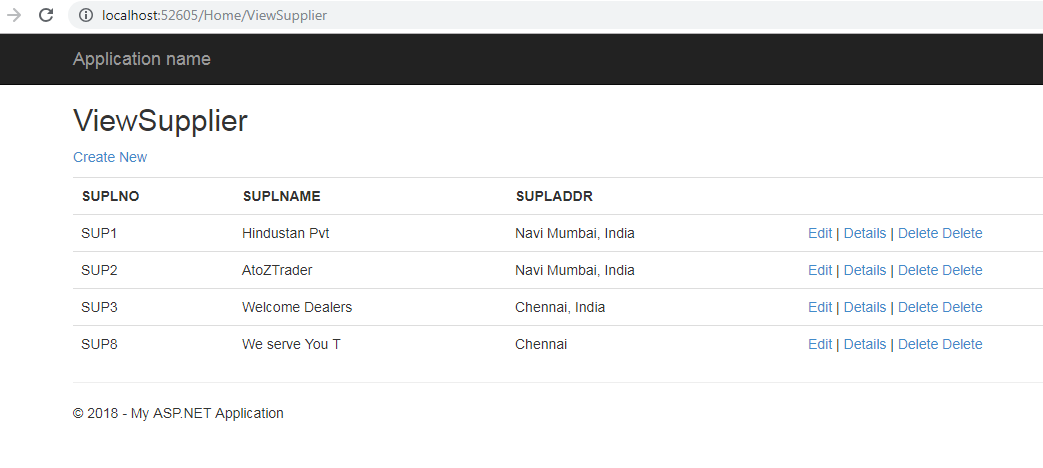
New Supplier



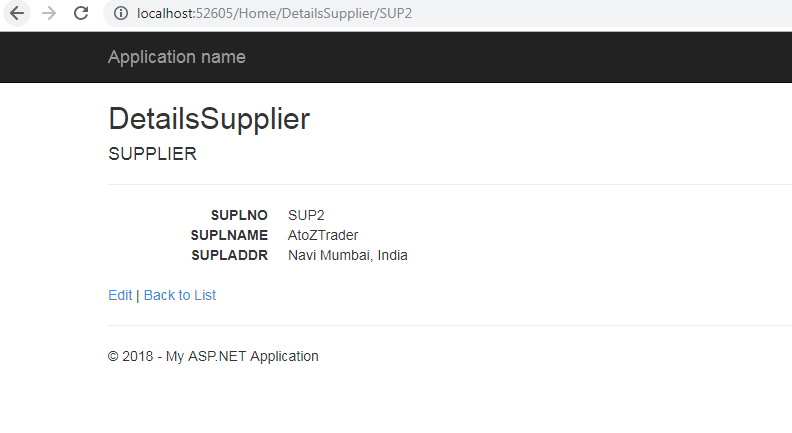


Edit Supplier

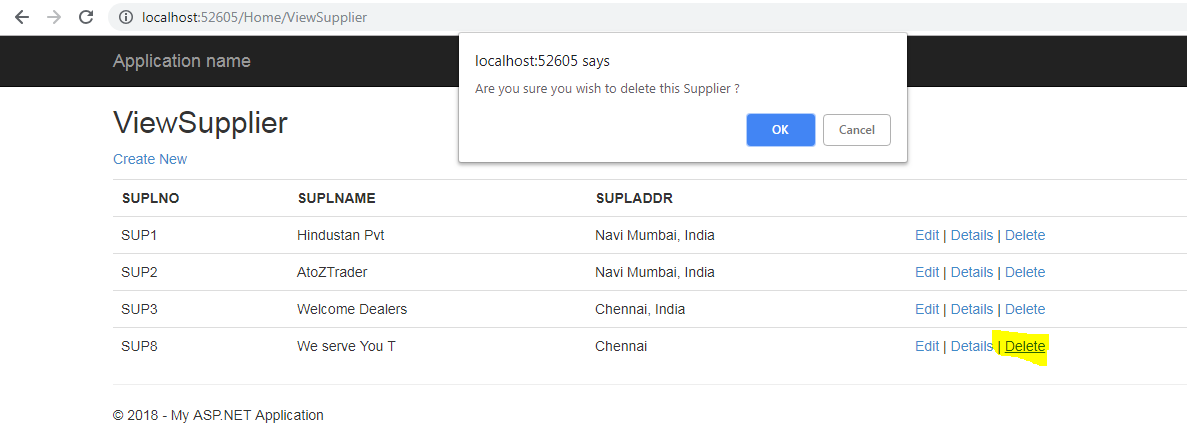


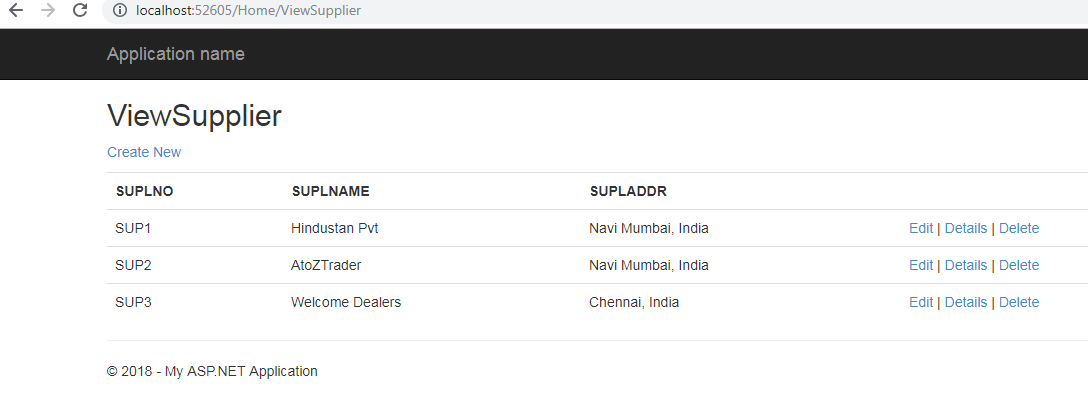


Details Supplier

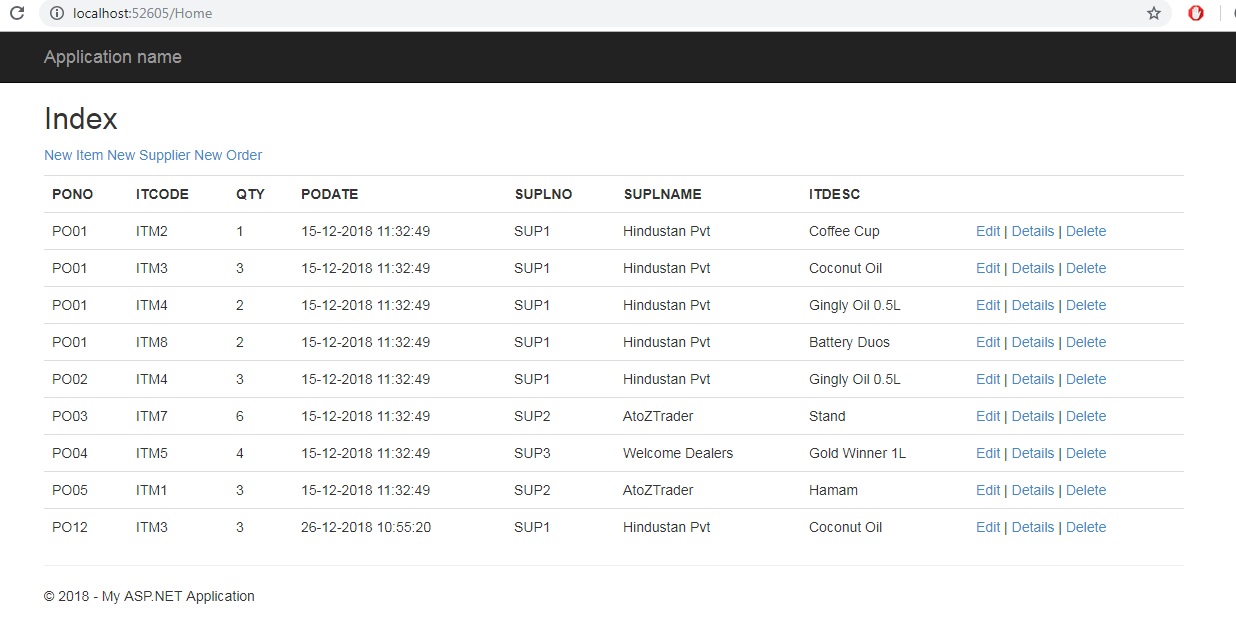


Delete Supplier

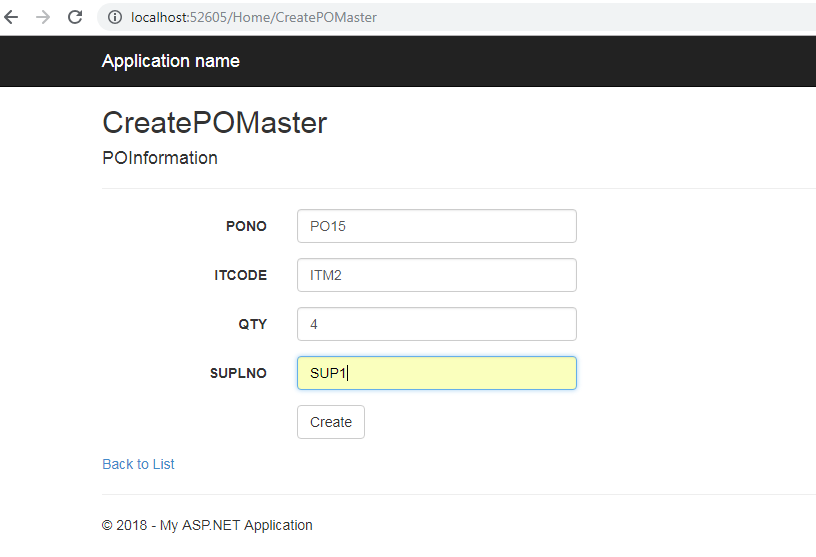


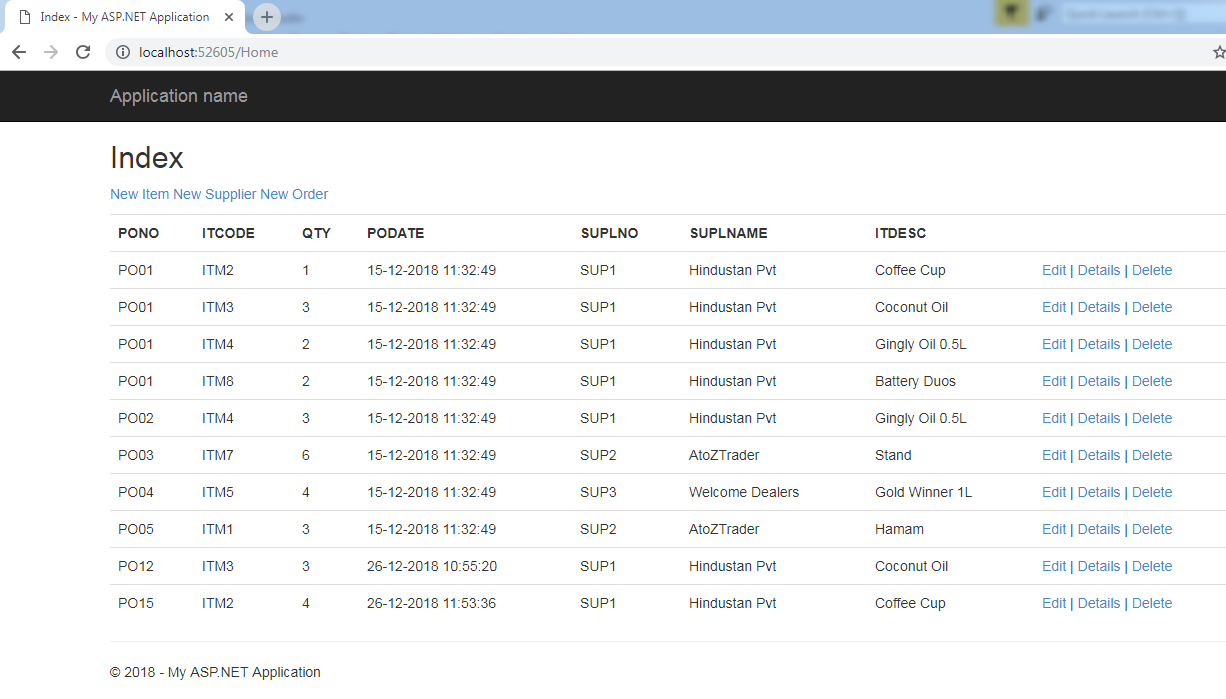


View PO Information (Shows PO Master and PO Details together)

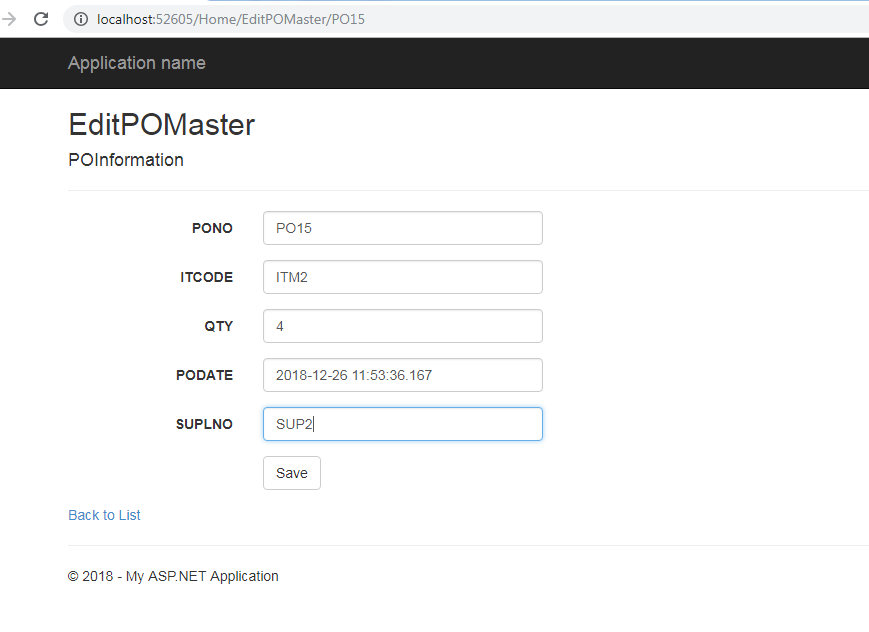


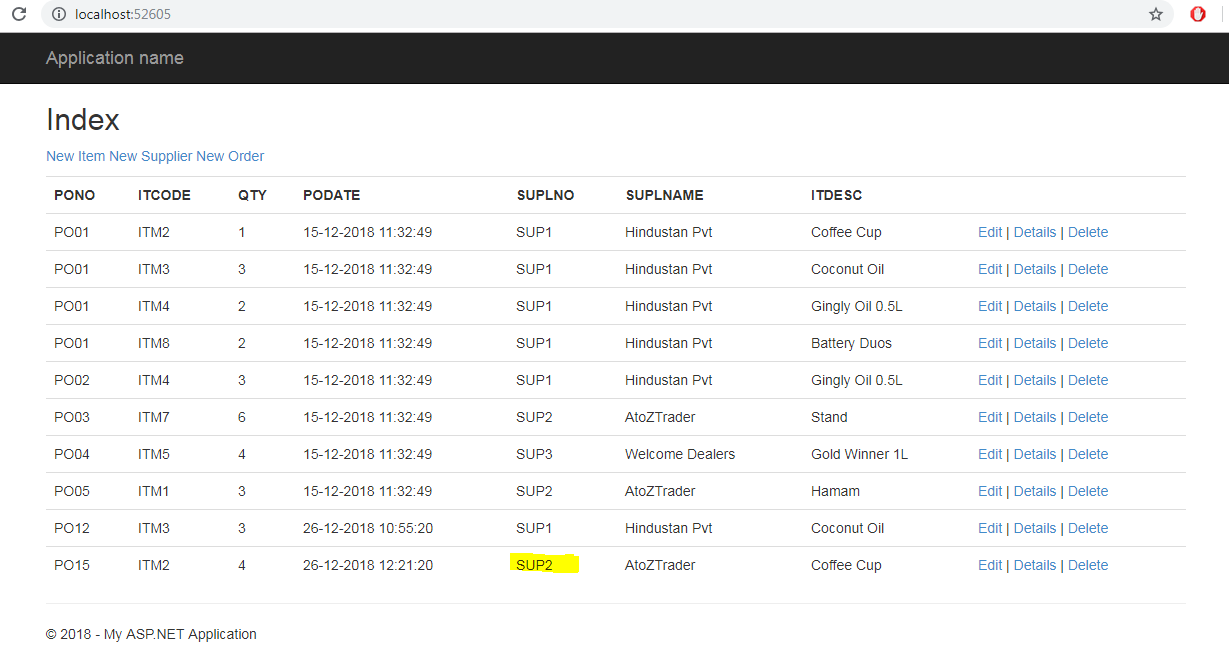
Create PO Master



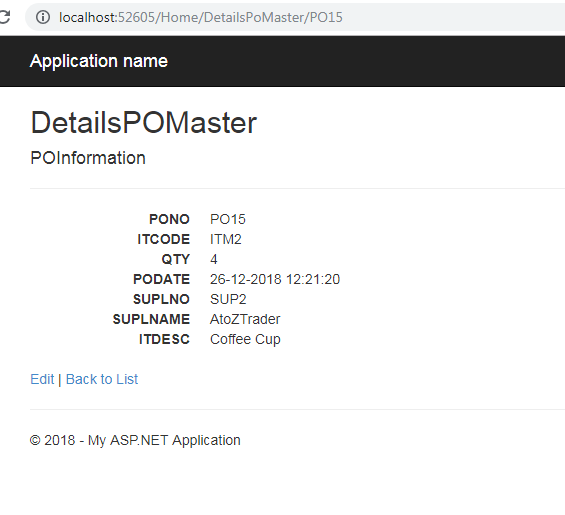


Edit

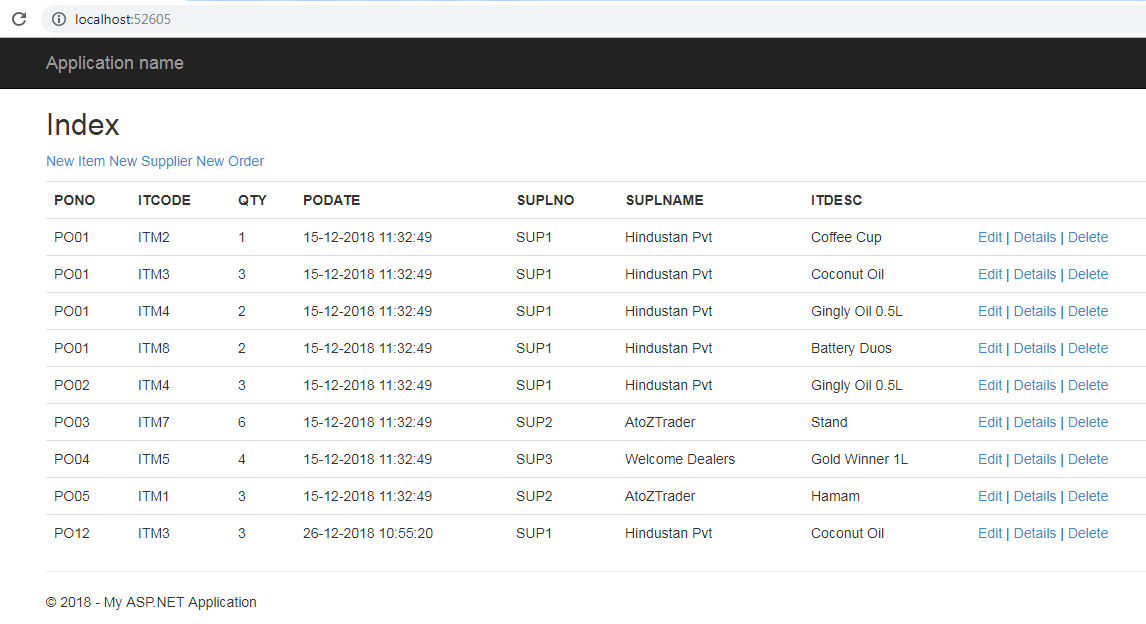




Details



After delete



Service of WEB API Project

Itemcontroller.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using WebAPIDemo.Models;

using System.Data.Entity;

namespace WebAPIDemo.Controllers

{

public class ItemController : ApiController

{

public IEnumerable<ITEM> Get()

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

return PE.ITEMs.ToList();

}

}

[HttpGet]

public HttpResponseMessage Get(string ItemNumber)

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

ITEM i= PE.ITEMs.Where(x => x.ITCODE == ItemNumber).FirstOrDefault();

if(i!=null)

{

return Request.CreateResponse(HttpStatusCode.OK, i);

}

else

{

return Request.CreateErrorResponse(HttpStatusCode.NotFound, "Item Number : " + ItemNumber + " Not Found");

}

}

}

public HttpResponseMessage Post([FromBody] ITEM i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PE.ITEMs.Add(i);

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.Created, i);

Message.Headers.Location = new Uri(Request.RequestUri + i.ITCODE);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Put([FromBody] ITEM i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

ITEM value= PE.ITEMs.Where(x => x.ITCODE == i.ITCODE).FirstOrDefault();

value.ITDESC = i.ITDESC;

value.ITRATE = i.ITRATE;

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.OK, value);

Message.Headers.Location = new Uri(Request.RequestUri + value.ITCODE);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Delete(string ItemCode)

{

HttpResponseMessage httpResponse = null;

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

ITEM I = PE.ITEMs.Where(x => x.ITCODE == ItemCode).FirstOrDefault();

if (I == null)

{

httpResponse= Request.CreateErrorResponse(HttpStatusCode.NotFound, "Item with Code " + ItemCode + " Not found");

}

else

{

PE.Entry(I).State = System.Data.Entity.EntityState.Deleted;

PE.SaveChanges();

httpResponse= Request.CreateResponse(HttpStatusCode.OK);

}

}

return httpResponse;

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

}

}

PODetail.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using WebAPIDemo.Models;

namespace WebAPIDemo.Controllers

{

public class PODETAILController : ApiController

{

public IEnumerable<PODETAIL> Get()

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

IEnumerable<PODETAIL> podetails = PE.PODETAILs.ToList();

return PE.PODETAILs.ToList();

}

}

[HttpGet]

public HttpResponseMessage Get(string PoNumber)

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PODETAIL i = PE.PODETAILs.Where(x => x.PONO == PoNumber).FirstOrDefault();

if (i != null)

{

return Request.CreateResponse(HttpStatusCode.OK, i);

}

else

{

return Request.CreateErrorResponse(HttpStatusCode.NotFound, "PODETAIL Number : " + PoNumber + " Not Found");

}

}

}

public HttpResponseMessage Post([FromBody] PODETAIL i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PE.PODETAILs.Add(i);

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.Created, i);

Message.Headers.Location = new Uri(Request.RequestUri + i.PONO);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Put([FromBody] PODETAIL i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PODETAIL value = PE.PODETAILs.Where(x => x.PONO == i.PONO).FirstOrDefault();

value.ITCODE = i.ITCODE;

value.QTY = i.QTY;

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.OK, value);

Message.Headers.Location = new Uri(Request.RequestUri + value.PONO);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Delete(string PONumber)

{

HttpResponseMessage httpResponse = null;

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PODETAIL I = PE.PODETAILs.Where(x => x.PONO == PONumber).FirstOrDefault();

if (I == null)

{

httpResponse = Request.CreateErrorResponse(HttpStatusCode.NotFound, "PODETAIL with Code " + PONumber + " Not found");

}

else

{

PE.Entry(I).State = System.Data.Entity.EntityState.Deleted;

PE.SaveChanges();

httpResponse = Request.CreateResponse(HttpStatusCode.OK);

}

}

return httpResponse;

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

}

}

SupplierController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using WebAPIDemo.Models;

namespace WebAPIDemo.Controllers

{

public class SupplierController : ApiController

{

public IEnumerable<SUPPLIER> Get()

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

return PE.SUPPLIERs.ToList();

}

}

[HttpGet]

public HttpResponseMessage Get(string SUPPLIERNumber)

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

SUPPLIER i = PE.SUPPLIERs.Where(x => x.SUPLNO == SUPPLIERNumber).FirstOrDefault();

if (i != null)

{

return Request.CreateResponse(HttpStatusCode.OK, i);

}

else

{

return Request.CreateErrorResponse(HttpStatusCode.NotFound, "SUPPLIER Number : " + SUPPLIERNumber + " Not Found");

}

}

}

public HttpResponseMessage Post([FromBody] SUPPLIER i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

PE.SUPPLIERs.Add(i);

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.Created, i);

Message.Headers.Location = new Uri(Request.RequestUri + i.SUPLNO);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Put([FromBody] SUPPLIER i)

{

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

SUPPLIER value = PE.SUPPLIERs.Where(x => x.SUPLNO == i.SUPLNO).FirstOrDefault();

value.SUPLNAME = i.SUPLNAME;

value.SUPLADDR = i.SUPLADDR;

PE.SaveChanges();

HttpResponseMessage Message = Request.CreateResponse(HttpStatusCode.OK, value);

Message.Headers.Location = new Uri(Request.RequestUri + value.SUPLNO);

return Message;

}

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

public HttpResponseMessage Delete(string SUPPLIERCode)

{

HttpResponseMessage httpResponse = null;

try

{

using (PODbEntities1 PE = new PODbEntities1())

{

PE.Configuration.ProxyCreationEnabled = false;

SUPPLIER I = PE.SUPPLIERs.Where(x => x.SUPLNO == SUPPLIERCode).FirstOrDefault();

if (I == null)

{

httpResponse = Request.CreateErrorResponse(HttpStatusCode.NotFound, "SUPPLIER with Code " + SUPPLIERCode + " Not found");

}

else

{

PE.Entry(I).State = System.Data.Entity.EntityState.Deleted;

PE.SaveChanges();

httpResponse = Request.CreateResponse(HttpStatusCode.OK);

}

}

return httpResponse;

}

catch (Exception ex)

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

}

}

}

}

MODEL

POINformation.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIDemo.Models

{

public class POInformation

{

public string PONO { get; set; }

public string ITCODE { get; set; }

public Nullable<int> QTY { get; set; }

public DateTime PODATE { get; set; }

public string SUPLNO { get; set; }

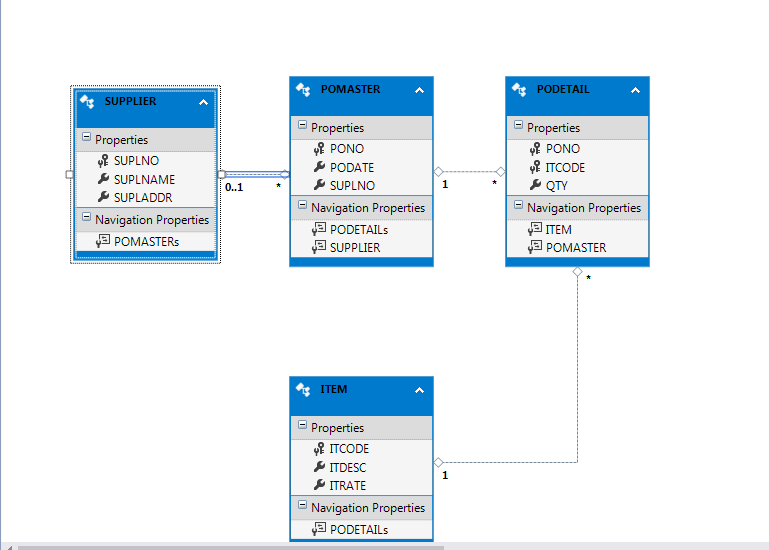
public string SUPLNAME { get; set; }

public string ITDESC { get; set; }

}

}

POP Model.edmx



Service web.config

<?xml version="1.0" encoding="utf-8"?>

<!--

For more information on how to configure your ASP.NET application, please visit

https://go.microsoft.com/fwlink/?LinkId=301879

-->

<configuration>

<appSettings>

<add key="webpages:Version" value="3.0.0.0" />

<add key="webpages:Enabled" value="false" />

<add key="ClientValidationEnabled" value="true" />

<add key="UnobtrusiveJavaScriptEnabled" value="true" />

</appSettings>

<system.web>

<compilation debug="true" targetFramework="4.5.2" />

<httpRuntime targetFramework="4.5.2" />

<httpModules>

<add name="ApplicationInsightsWebTracking" type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web" />

</httpModules>

</system.web>

<system.webServer>

<handlers>

<remove name="ExtensionlessUrlHandler-Integrated-4.0" />

<remove name="OPTIONSVerbHandler" />

<remove name="TRACEVerbHandler" />

<add name="ExtensionlessUrlHandler-Integrated-4.0" path="\*." verb="\*" type="System.Web.Handlers.TransferRequestHandler" preCondition="integratedMode,runtimeVersionv4.0" />

</handlers>

<modules>

<remove name="TelemetryCorrelationHttpModule" />

<add name="TelemetryCorrelationHttpModule" type="Microsoft.AspNet.TelemetryCorrelation.TelemetryCorrelationHttpModule, Microsoft.AspNet.TelemetryCorrelation" preCondition="integratedMode,managedHandler" />

<remove name="ApplicationInsightsWebTracking" />

<add name="ApplicationInsightsWebTracking" type="Microsoft.ApplicationInsights.Web.ApplicationInsightsHttpModule, Microsoft.AI.Web" preCondition="managedHandler" />

</modules>

<validation validateIntegratedModeConfiguration="false" />

</system.webServer>

<runtime>

<assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">

<dependentAssembly>

<assemblyIdentity name="Antlr3.Runtime" publicKeyToken="eb42632606e9261f" />

<bindingRedirect oldVersion="0.0.0.0-3.5.0.2" newVersion="3.5.0.2" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Diagnostics.DiagnosticSource" publicKeyToken="cc7b13ffcd2ddd51" />

<bindingRedirect oldVersion="0.0.0.0-4.0.2.1" newVersion="4.0.2.1" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="Newtonsoft.Json" culture="neutral" publicKeyToken="30ad4fe6b2a6aeed" />

<bindingRedirect oldVersion="0.0.0.0-11.0.0.0" newVersion="11.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Optimization" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="1.1.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="WebGrease" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-1.6.5135.21930" newVersion="1.6.5135.21930" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Helpers" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.WebPages" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

</dependentAssembly>

<dependentAssembly>

<assemblyIdentity name="System.Web.Mvc" publicKeyToken="31bf3856ad364e35" />

<bindingRedirect oldVersion="1.0.0.0-5.2.4.0" newVersion="5.2.4.0" />

</dependentAssembly>

</assemblyBinding>

</runtime>

<system.codedom>

<compilers>

<compiler language="c#;cs;csharp" extension=".cs" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:6 /nowarn:1659;1699;1701" />

<compiler language="vb;vbs;visualbasic;vbscript" extension=".vb" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:14 /nowarn:41008 /define:\_MYTYPE=\&quot;Web\&quot; /optionInfer+" />

</compilers>

</system.codedom>

<connectionStrings><add name="PODbEntities" connectionString="metadata=res://\*/PODB.csdl|res://\*/PODB.ssdl|res://\*/PODB.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=MANOJ-PC\SQLEXPRESS;initial catalog=PODb;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" /><add name="PODbEntities1" connectionString="metadata=res://\*/Models.POPModel.csdl|res://\*/Models.POPModel.ssdl|res://\*/Models.POPModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=MANOJ-PC\SQLEXPRESS;initial catalog=PODb;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" /></connectionStrings>

</configuration>

Web Client

HomeController.cs

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Formatting;

using System.Net.Http.Headers;

using System.Threading.Tasks;

using System.Web;

using System.Web.Mvc;

using WebAPIClient.Models;

namespace WebAPIClient.Controllers

{

public class HomeController : Controller

{

//Hosted web API REST Service base url

string Baseurl = "http://localhost:49195/";

public async Task<ActionResult> Index()

{

//var cart = ShoppingCart.GetCart(this.HttpContext);

//// Set up our ViewModel

//var viewModel = new ShoppingCartViewModel

//{

// poinformation = cart.GetCartItems(),

// count = cart.GetTotal()

//};

List<POInformation> ItemInfo = new List<POInformation>();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/POMaster/Get");

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<POInformation>>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

public async Task<ActionResult> EditItem(string Id)

{

ITEM ItemInfo = new ITEM();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/Item/Get?ItemNumber=" + Id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<ITEM>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

public ActionResult CreateItem()

{

//returning the Item list to view

return View();

}

[HttpPost]

[ActionName("CreateItem")]

public async Task<ActionResult> CreateItem(ITEM i)

{

try

{

ITEM Value = new ITEM();

if (ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

HttpResponseMessage Res = await client.PostAsJsonAsync<ITEM>("api/ITEM/POST", Value);

//Checking the response is successful or not which is sent using HttpClient

ITEM ItemInfo = new ITEM();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<ITEM>(ItemResponse);

return RedirectToAction("Index");

}

}

//returning the Item list to view

// return View(ItemInfo);

}

}

catch (Exception e)

{

throw;

}

return View();

}

[HttpPost]

[ActionName("EditItem")]

public async Task<ActionResult> EditItem\_Post(ITEM i)

{

try

{

ITEM Value = new ITEM();

if (ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

HttpResponseMessage Res = await client.PutAsJsonAsync<ITEM>("api/Item/Put", Value);

//Checking the response is successful or not which is sent using HttpClient

ITEM ItemInfo = new ITEM();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<ITEM>(ItemResponse);

return RedirectToAction("Index");

}

}

//returning the Item list to view

// return View(ItemInfo);

}

}

catch (Exception e)

{

throw;

}

return View();

}

public async Task<ActionResult> DetailsItem(string Id)

{

ITEM ItemInfo = new ITEM();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/Item/Get?ItemNumber=" + Id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<ITEM>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

public async Task<ActionResult> DeleteItem(string Id)

{

ITEM ItemInfo = new ITEM();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.DeleteAsync("api/Item/Delete?ItemCode=" + Id);

//returning the Item list to view

return RedirectToAction("Index");

}

}

public async Task<ActionResult> ViewItems()

{

List<ITEM> ItemInfo = new List<ITEM>();

//returning the Item list to view

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/ITEM/Get");

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<ITEM>>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

public ActionResult CreateSupplier()

{

//returning the Item list to view

return View();

}

public async Task<ActionResult> ViewSupplier()

{

List<SUPPLIER> ItemInfo = new List<SUPPLIER>();

//returning the Item list to view

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/Supplier/Get");

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<SUPPLIER>>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

[HttpPost]

[ActionName("CreateSupplier")]

public async Task<ActionResult> CreateSupplier\_Post(SUPPLIER s)

{

SUPPLIER Value = new SUPPLIER();

if (ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

HttpResponseMessage Res = await client.PostAsJsonAsync<SUPPLIER>("api/Supplier/POST", Value);

//Checking the response is successful or not which is sent using HttpClient

SUPPLIER SupplierInfo = new SUPPLIER();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

// SupplierInfo = JsonConvert.DeserializeObject<SUPPLIER>(ItemResponse);

return RedirectToAction("ViewSupplier");

}

}

}

//returning the Item list to view

return View();

}

public async Task<ActionResult> EditSupplier(string id)

{

SUPPLIER ItemInfo = new SUPPLIER();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/Supplier/Get?SUPPLIERNumber=" + id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<SUPPLIER>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

[HttpPost]

[ActionName("EditSupplier")]

public async Task<ActionResult> EditSupplier\_Post(SUPPLIER s)

{

SUPPLIER Value = new SUPPLIER();

if (ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

HttpResponseMessage Res = await client.PutAsJsonAsync<SUPPLIER>("api/Supplier/Put", Value);

//Checking the response is successful or not which is sent using HttpClient

SUPPLIER SupplierInfo = new SUPPLIER();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

SupplierInfo = JsonConvert.DeserializeObject<SUPPLIER>(ItemResponse);

return RedirectToAction("ViewSupplier");

}

}

}

return View(s);

}

public async Task<ActionResult> DetailsSupplier(string id)

{

SUPPLIER ItemInfo = new SUPPLIER();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/Supplier/Get?SUPPLIERNumber=" + id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<SUPPLIER>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo);

}

}

public async Task<ActionResult> DeleteSupplier(string Id)

{

ITEM ItemInfo = new ITEM();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.DeleteAsync("api/Supplier/Delete?SUPPLIERCode=" + Id);

//returning the Item list to view

return RedirectToAction("ViewSupplier");

}

}

public async Task<ActionResult> CreatePOMaster()

{

List<ITEM> ItemInfo = new List<ITEM>();

List<SUPPLIER> supplierInfo = new List<SUPPLIER>();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/ITEM/Get");

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<ITEM>>(ItemResponse);

}

Res = await client.GetAsync("api/SUPPLIER/Get");

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

supplierInfo = JsonConvert.DeserializeObject<List<SUPPLIER>>(ItemResponse);

}

Information info = new Information();

info.items = ItemInfo;

info.suppliers = supplierInfo;

//returning the Item list to view

return View(new POInformation());

}

}

public ActionResult ViewPOMaster()

{

//returning the Item list to view

return View();

}

[HttpPost]

[ActionName("CreatePOMaster")]

public async Task<ActionResult> CreatePOMaster\_Post(POInformation pi)

{

POInformation Value = new POInformation();

if(ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

List<POInformation> poinfo = new List<POInformation>();

poinfo.Add(Value);

HttpResponseMessage Res = await client.PostAsJsonAsync<List<POInformation>>("api/POMaster/POST", poinfo);

//Checking the response is successful or not which is sent using HttpClient

POInformation POMasterInfo = new POInformation();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

// POMasterInfo = JsonConvert.DeserializeObject<POInformation>(ItemResponse);

return RedirectToAction("Index");

}

}

}

return View(pi);

}

public async Task<ActionResult> EditPOMaster(string id)

{

List<POInformation> ItemInfo = new List<POInformation>();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/POMaster/Get?PoNumber=" + id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<POInformation>>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo.FirstOrDefault());

}

}

[HttpPost]

[ActionName("EditPOMaster")]

public async Task<ActionResult> EditPOMaster\_Post(POInformation p)

{

POInformation Value = new POInformation();

if (ModelState.IsValid)

{

TryUpdateModel(Value);

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

var myContent = JsonConvert.SerializeObject(Value);

var buffer = System.Text.Encoding.UTF8.GetBytes(myContent);

ByteArrayContent byteContent = new ByteArrayContent(buffer);

List<POInformation> plist = new List<POInformation>();

plist.Add(Value);

HttpResponseMessage Res = await client.PutAsJsonAsync<List<POInformation>>("api/POMaster/Put", plist);

//Checking the response is successful or not which is sent using HttpClient

POInformation SupplierInfo = new POInformation();

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

// SupplierInfo = JsonConvert.DeserializeObject<POInformation>(ItemResponse);

return RedirectToAction("Index");

}

}

}

return View(p);

}

public async Task<ActionResult> DetailsPOMaster(string id)

{

List<POInformation> ItemInfo = new List<POInformation>();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.GetAsync("api/POMaster/Get?PoNumber=" + id);

//Checking the response is successful or not which is sent using HttpClient

if (Res.IsSuccessStatusCode)

{

//Storing the response details recieved from web api

var ItemResponse = Res.Content.ReadAsStringAsync().Result;

//Deserializing the response recieved from web api and storing into the Item list

ItemInfo = JsonConvert.DeserializeObject<List<POInformation>>(ItemResponse);

}

//returning the Item list to view

return View(ItemInfo.FirstOrDefault());

}

}

public async Task<ActionResult> DeletePOMaster(string Id)

{

POInformation ItemInfo = new POInformation();

using (var client = new HttpClient())

{

//Passing service base url

client.BaseAddress = new Uri(Baseurl);

client.DefaultRequestHeaders.Clear();

//Define request data format

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

//Sending request to find web api REST service resource GetAllItem using HttpClient

HttpResponseMessage Res = await client.DeleteAsync("api/POMaster/Delete?POMASTERCode=" + Id);

//returning the Item list to view

return RedirectToAction("Index");

}

}

}

}

MODEL

ITEM.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIClient.Models

{

public partial class ITEM

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public ITEM()

{

this.PODETAILs = new HashSet<PODETAIL>();

}

public string ITCODE { get; set; }

public string ITDESC { get; set; }

public Nullable<decimal> ITRATE { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<PODETAIL> PODETAILs { get; set; }

}

}

PODEtails.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIClient.Models

{

public partial class PODETAIL

{

public string PONO { get; set; }

public string ITCODE { get; set; }

public Nullable<int> QTY { get; set; }

public virtual ITEM ITEM { get; set; }

public virtual POMASTER POMASTER { get; set; }

}

}

POInformation

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIClient.Models

{

public class POInformation

{

public string PONO { get; set; }

public string ITCODE { get; set; }

public Nullable<int> QTY { get; set; }

public DateTime PODATE { get; set; }

public string SUPLNO { get; set; }

public string SUPLNAME { get; set; }

public string ITDESC { get; set; }

}

}

POMaster.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIClient.Models

{

public partial class POMASTER

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public POMASTER()

{

this.PODETAILs = new HashSet<PODETAIL>();

}

public string PONO { get; set; }

public Nullable<System.DateTime> PODATE { get; set; }

public string SUPLNO { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<PODETAIL> PODETAILs { get; set; }

public virtual SUPPLIER SUPPLIER { get; set; }

}

}

Supplier.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebAPIClient.Models

{

public partial class SUPPLIER

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public SUPPLIER()

{

}

public string SUPLNO { get; set; }

public string SUPLNAME { get; set; }

public string SUPLADDR { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<POMASTER> POMASTERs { get; set; }

}

}

Views

COnfirmDelete.cs.html

@model WebAPIClient.Models.ITEM

<script type="text/javascript">

$(document).ready(function () {

$('a.delete').click(OnDeleteClick);

});

function OnDeleteClick(e)

{

var employeeId = e.target.id;

var flag = confirm('You are about to delete Employee ID ' + employeeId + ' permanently.

Are you sure you want to delete this record?');

if (flag) {

$.ajax({

url: '/home/DeleteAJAX',

type: 'POST',

data: { employeeid: employeeId },

dataType: 'json',

success: function (result) { alert(result); $("#" + employeeId).parent().parent().remove(); },

error: function () { alert('Error!'); }

});

}

return false;

}

</script>

CreateItem.cs.html

@model WebAPIClient.Models.ITEM

@{

ViewBag.Title = "CreateItem";

}

<h2>CreateItem</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>ITEM</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.ITCODE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITCODE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITCODE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITDESC, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITDESC, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITDESC, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITRATE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITRATE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITRATE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

CreatePOMaster.cs.html

@model WebAPIClient.Models.POInformation

@{

ViewBag.Title = "CreatePOMaster";

}

<h2>CreatePOMaster</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>POInformation</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.PONO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PONO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PONO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITCODE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITCODE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITCODE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.QTY, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.QTY, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.QTY, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLNO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

CreateSupplier.cs.html

@model WebAPIClient.Models.SUPPLIER

@{

ViewBag.Title = "CreateSupplier";

}

<h2>CreateSupplier</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>SUPPLIER</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.SUPLNO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLNAME, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLADDR, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLADDR, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLADDR, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

DetailsItem.cs.html

@model WebAPIClient.Models.ITEM

@{

ViewBag.Title = "DetailsItem";

}

<h2>DetailsItem</h2>

<div>

<h4>ITEM</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.ITCODE)

</dt>

<dd>

@Html.DisplayFor(model => model.ITCODE)

</dd>

<dt>

@Html.DisplayNameFor(model => model.ITDESC)

</dt>

<dd>

@Html.DisplayFor(model => model.ITDESC)

</dd>

<dt>

@Html.DisplayNameFor(model => model.ITRATE)

</dt>

<dd>

@Html.DisplayFor(model => model.ITRATE)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Edit", "EditItem", new { id = Model.ITCODE }) |

@Html.ActionLink("Back to List", "Index")

</p>

CreatePoMaster.cs.html

@model WebAPIClient.Models.POInformation

@{

ViewBag.Title = "DetailsPOMaster";

}

<h2>DetailsPOMaster</h2>

<div>

<h4>POInformation</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.PONO)

</dt>

<dd>

@Html.DisplayFor(model => model.PONO)

</dd>

<dt>

@Html.DisplayNameFor(model => model.ITCODE)

</dt>

<dd>

@Html.DisplayFor(model => model.ITCODE)

</dd>

<dt>

@Html.DisplayNameFor(model => model.QTY)

</dt>

<dd>

@Html.DisplayFor(model => model.QTY)

</dd>

<dt>

@Html.DisplayNameFor(model => model.PODATE)

</dt>

<dd>

@Html.DisplayFor(model => model.PODATE)

</dd>

<dt>

@Html.DisplayNameFor(model => model.SUPLNO)

</dt>

<dd>

@Html.DisplayFor(model => model.SUPLNO)

</dd>

<dt>

@Html.DisplayNameFor(model => model.SUPLNAME)

</dt>

<dd>

@Html.DisplayFor(model => model.SUPLNAME)

</dd>

<dt>

@Html.DisplayNameFor(model => model.ITDESC)

</dt>

<dd>

@Html.DisplayFor(model => model.ITDESC)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Edit", "EditPOMaster", new { id = Model.PONO }) |

@Html.ActionLink("Back to List", "Index")

</p>

DetailsSupplier.cs.html

@model WebAPIClient.Models.SUPPLIER

@{

ViewBag.Title = "DetailsSupplier";

}

<h2>DetailsSupplier</h2>

<div>

<h4>SUPPLIER</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.SUPLNO)

</dt>

<dd>

@Html.DisplayFor(model => model.SUPLNO)

</dd>

<dt>

@Html.DisplayNameFor(model => model.SUPLNAME)

</dt>

<dd>

@Html.DisplayFor(model => model.SUPLNAME)

</dd>

<dt>

@Html.DisplayNameFor(model => model.SUPLADDR)

</dt>

<dd>

@Html.DisplayFor(model => model.SUPLADDR)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Edit", "EditSupplier", new { id = Model.SUPLNO }) |

@Html.ActionLink("Back to List", "ViewSupplier")

</p>

EditItem.cs.html

@model WebAPIClient.Models.ITEM

@{

ViewBag.Title = "Edit";

}

<h2>Edit</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>ITEM</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.ITCODE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITCODE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITCODE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITDESC, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITDESC, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITDESC, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITRATE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITRATE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITRATE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

EditPOMaster.cs.html

@model WebAPIClient.Models.POInformation

@{

ViewBag.Title = "EditPOMaster";

}

<h2>EditPOMaster</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>POInformation</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.PONO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PONO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PONO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ITCODE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ITCODE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ITCODE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.QTY, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.QTY, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.QTY, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.PODATE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PODATE, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PODATE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLNO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

EditSupplier.cs.html

@model WebAPIClient.Models.SUPPLIER

@{

ViewBag.Title = "EditSupplier";

}

<h2>EditSupplier</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>SUPPLIER</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.SUPLNO, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNO, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNO, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLNAME, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLNAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLNAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.SUPLADDR, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.SUPLADDR, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.SUPLADDR, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

Index.cs.html

@model IEnumerable<WebAPIClient.Models.POInformation>

@{

ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

@Html.ActionLink("New Item", "CreateItem")

@Html.ActionLink("New Supplier", "CreateSupplier")

@Html.ActionLink("New Order", "CreatePOMaster")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.PONO)

</th>

<th>

@Html.DisplayNameFor(model => model.ITCODE)

</th>

<th>

@Html.DisplayNameFor(model => model.QTY)

</th>

<th>

@Html.DisplayNameFor(model => model.PODATE)

</th>

<th>

@Html.DisplayNameFor(model => model.SUPLNO)

</th>

<th>

@Html.DisplayNameFor(model => model.SUPLNAME)

</th>

<th>

@Html.DisplayNameFor(model => model.ITDESC)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.PONO)

</td>

<td>

@Html.DisplayFor(modelItem => item.ITCODE)

</td>

<td>

@Html.DisplayFor(modelItem => item.QTY)

</td>

<td>

@Html.DisplayFor(modelItem => item.PODATE)

</td>

<td>

@Html.DisplayFor(modelItem => item.SUPLNO)

</td>

<td>

@Html.DisplayFor(modelItem => item.SUPLNAME)

</td>

<td>

@Html.DisplayFor(modelItem => item.ITDESC)

</td>

<td>

@Html.ActionLink("Edit", "EditPOMaster", new { id=item.PONO }) |

@Html.ActionLink("Details", "DetailsPoMaster", new { id=item.PONO }) |

@Html.ActionLink("Delete", "DeletePOMaster", new { id=item.PONO })

</td>

</tr>

}

</table>

ViewItems.cs.html

@model IEnumerable<WebAPIClient.Models.ITEM>

@{

ViewBag.Title = "ViewItems";

}

<h2>ViewItems</h2>

<p>

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

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.ITCODE)

</th>

<th>

@Html.DisplayNameFor(model => model.ITDESC)

</th>

<th>

@Html.DisplayNameFor(model => model.ITRATE)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.ITCODE)

</td>

<td>

@Html.DisplayFor(modelItem => item.ITDESC)

</td>

<td>

@Html.DisplayFor(modelItem => item.ITRATE)

</td>

<td>

@Html.ActionLink("Edit", "EditItem", new { id=item.ITCODE }) |

@Html.ActionLink("Details", "DetailsItem", new { id=item.ITCODE }) |

@Html.ActionLink(

"Delete",

"DeleteItem",

new { id = item.ITCODE },

new { onclick = "return confirm('Are you sure you wish to delete this Item ?');" })

</td>

</tr>

}

</table>

ViewPOMaster.cs.html

@model IEnumerable<WebAPIClient.Models.POMASTER>

@{

ViewBag.Title = "ViewPOMaster";

}

<h2>ViewPOMaster</h2>

<p>

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

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.PONO)

</th>

<th>

@Html.DisplayNameFor(model => model.PODATE)

</th>

<th>

@Html.DisplayNameFor(model => model.SUPLNO)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.PONO)

</td>

<td>

@Html.DisplayFor(modelItem => item.PODATE)

</td>

<td>

@Html.DisplayFor(modelItem => item.SUPLNO)

</td>

<td>

@Html.ActionLink("Edit", "EditPOMaster", new { id = item.PONO }) |

@Html.ActionLink("Details", "DetailsPOMaster", new { id = item.PONO }) |

@Html.ActionLink(

"Delete",

"DeletePOMaster",

new { id = item.PONO },

new { onclick = "return confirm('Are you sure you wish to delete this POMaster ?');" })

</td>

</tr>

}

</table>

ViewSupplier.cs.html

@model IEnumerable<WebAPIClient.Models.SUPPLIER>

@{

ViewBag.Title = "ViewSupplier";

}

<h2>ViewSupplier</h2>

<p>

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

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.SUPLNO)

</th>

<th>

@Html.DisplayNameFor(model => model.SUPLNAME)

</th>

<th>

@Html.DisplayNameFor(model => model.SUPLADDR)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.SUPLNO)

</td>

<td>

@Html.DisplayFor(modelItem => item.SUPLNAME)

</td>

<td>

@Html.DisplayFor(modelItem => item.SUPLADDR)

</td>

<td>

@Html.ActionLink("Edit", "EditSupplier", new { id = item.SUPLNO }) |

@Html.ActionLink("Details", "DetailsSupplier", new { id = item.SUPLNO }) |

@Html.ActionLink(

"Delete",

"DeleteSupplier",

new { id = item.SUPLNO },

new { onclick = "return confirm('Are you sure you wish to delete this Supplier ?');" })

</td>

</tr>

}

</table>