**Tipo** : Guía de Enunciado
**Capítulo** : ASP NET Web Api
**Duración** : 60 minutos

## I. OBJETIVO

Creando controladores de una Web Api.

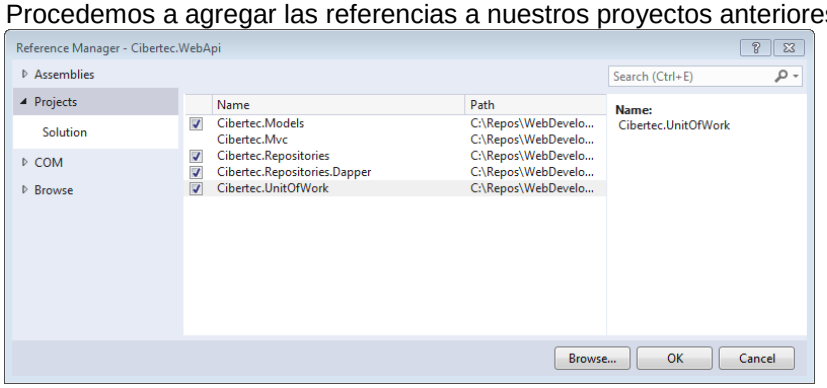
## II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio:

- Windows 10 (como mínimo Windows 8)
- Visual Studio 2017 (como mínimo Visual Studio 2015)

## III. EJECUCIÓN DEL LABORATORIO

1. Abrir la solución del módulo anterior.
2. Ubicarse en el proyecto "**Cibertec.WebApi**"
3. De la carpeta "**Controllers"** eliminar el archivo "**CibertecController.cs**".
4. Procedemos a agregar las referencias a nuestros proyectos anteriores.




5. En la carpeta Controllers, agregamos un controlador nuevo con el nombre "**BaseController**", con el siguiente código:

```
using System.Web.Http;
using Cibertec.UnitOfWork;

namespace Cibertec.WebApi.Controllers
{
    public class BaseController : ApiController
    {
        protected readonly IUnitOfWork _unit;
        public BaseController(IUnitOfWork unit)
        {
            _unit = unit;
        }
    }
}
```

6. Procede a creaer un nuevo controlador con el nombre "**CustomerController**" con el siguiente código:

```
using Cibertec.Models;
using Cibertec.UnitOfWork;
using System.Web.Http;

namespace Cibertec.WebApi.Controllers
{
    [RoutePrefix("customer")]
    public class CustomerController : BaseController
    {
        public CustomerController(IUnitOfWork unit) : base(unit)
        {
        }

        [Route("{id}")]
        public IHttpActionResult Get(int id)
        {
            if (id <= 0) return BadRequest();
            return Ok(_unit.Customers.GetById(id));
        }

        [Route("")]
        [HttpPost]
        public IHttpActionResult Post(Customer customer)
        {
            if (!ModelState.IsValid) return BadRequest(ModelState);
            var id = _unit.Customers.Insert(customer);
            return Ok(new { id = id });
        }

        [Route("")]
        [HttpPut]
        public IHttpActionResult Put(Customer customer)
        {
            if (!ModelState.IsValid) return BadRequest(ModelState);
            if (!_unit.Customers.Update(customer)) return
BadRequest("Incorrect id");
            return Ok(new { status = true });
        }

        [Route("{id}")]
        [HttpDelete]
        public IHttpActionResult Delete(int id)
        {
            if (id <= 0) return BadRequest();
            var result = _unit.Customers.Delete(new Customer { Id = id });
            return Ok(new { delete = true });
        }

        [HttpGet]
        [Route("list")]
        public IHttpActionResult GetList()
        {
            return Ok(_unit.Customers.GetList());
        }
    }
}
```

7. Para que el ruteo por notación sea legible por nuestra WebApi debemos de agregar esta línea en nuestro fichero WebApiConfig.cs:

```
using System.Web.Http;

namespace Cibertec.WebApi
{
    public partial class Startup
    {
        public static void Register(HttpConfiguration config)
        {
            config.MapHttpAttributeRoutes();

            config.Routes.MapHttpRoute(
                name: "DefaultApi",
                routeTemplate: "api/{controller}/{id}",
                defaults: new { id = RouteParameter.Optional }
            );
        }
    }
}
```

8. Ejecutamos nuestra aplicación y validamos que nuestros Endpoints funcione adecuadamente.

## IV. EVALUACIÓN

1. ¿Por qué para vistas parciales se deben de usar "_" el guion bajo?

   Bajo el lenguaje Razor, el guion bajo permite evitar mostrar Web Pages cuando son solicitadas directamente como parte de un request.