Herramientas de Programación 2

Tema Nº5:ENTITY FRAMEWORK CORE (PARTE 2)

Indicador de logro Nº5:Identifica y comprueba los componentes de SOA aplicando los métodos HTTP aprendidos

**TEMA 01 Teoría de los**

Imagen que contiene Icono

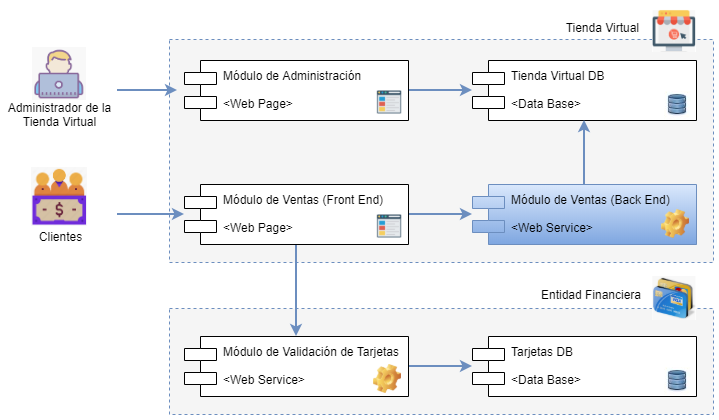
Descripción generada automáticamente

**TEMA Nº5:**

ENTITY FRAMEWORK CORE (PARTE 2)

**Subtema 5.2:**

Módulo de Ventas (Back End)



Paso 1. Implementar servicios que gestionen Categorías.

1. Abrimos en Visual Studio el proyecto del taller anterior.
2. Agregamos el controlador API en blanco: *CategoriasController.cs*
3. En el controlador agregamos la siguiente funcionalidad:
   * Método que devuelva todas las categorías de la BD
   * Método que devuelva solo las categorías activas.

Modificamos el código del controlador de acuerdo a:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MiTiendaVirtualAPI.Models;

namespace MiTiendaVirtualAPI.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class CategoriasController : ControllerBase

{

private readonly TiendaVirtualDBContext BD;

public CategoriasController(TiendaVirtualDBContext context)

{

BD = context;

}

//GET. /api/categorias

[HttpGet]

public IEnumerable<Categoria> Categorias()

{

return BD.Categoria.ToList();

}

//GET. /api/categorias/activas

[Route("activas")]

[HttpGet]

public IEnumerable<Categoria> CategoriasActivas()

{

List<Categoria> listaCategorias = new List<Categoria>();

listaCategorias = (from c in BD.Categoria

where c.Activo == true

select c).ToList();

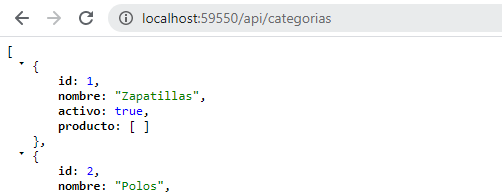
return listaCategorias;

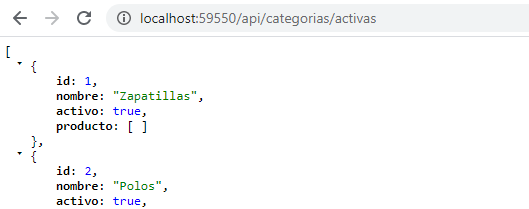
}

}

}

1. Ejecutamos la aplicación y validamos las funcionalidades que se implementaron.





Paso 2. Implementar servicios que gestionen Productos.

1. Agregamos el controlador API en blanco: *ProductosController.cs*
2. En el controlador agregamos la siguiente funcionalidad:
   * Método que devuelva todos los productos de la BD
   * Método que devuelva solo los productos destacados (y activos).
   * Método que devuelva los productos (activos) que pertenezcan a cierta categoría.
   * Método que devuelva un producto con todos sus datos.

Modificamos el código del controlador de acuerdo a:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MiTiendaVirtualAPI.Models;

namespace MiTiendaVirtualAPI.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class ProductosController : ControllerBase

{

private readonly TiendaVirtualDBContext BD;

public ProductosController(TiendaVirtualDBContext context)

{

BD = context;

}

//GET. /api/productos

[HttpGet]

public IEnumerable<Producto> Productos()

{

return BD.Producto.ToList();

}

//GET. /api/productos/destacados

[Route("destacados")]

[HttpGet]

public IEnumerable<Producto> ProductosDestacados()

{

List<Producto> listaProductos = new List<Producto>();

listaProductos = (from p in BD.Producto

where p.Destacado == true && p.Activo == true

select p).ToList();

return listaProductos;

}

//GET. /api/productos/porcategoria/2

[Route("porcategoria/{id}")]

[HttpGet("{id}")]

public IEnumerable<Producto> ProductosPorCategoria(int id)

{

List<Producto> listaProductos = new List<Producto>();

listaProductos = (from p in BD.Producto

where p.IdCategoria == id && p.Activo == true

select p).ToList();

return listaProductos;

}

//GET. /api/productos/producto/2

[Route("producto/{id}")]

[HttpGet("{id}")]

public IActionResult Producto(int id)

{

Producto producto = new Producto();

producto = (from p in BD.Producto

where p.Id == id

select new Producto

{

Id = p.Id,

IdCategoria = p.IdCategoria,

IdMarca = p.IdMarca,

Nombre = p.Nombre,

Descripcion = p.Descripcion,

Precio = p.Precio,

Url = p.Url,

Destacado = p.Destacado,

Activo = p.Activo,

IdCategoriaNavigation = ObtenerCategoria(p.IdCategoria.GetValueOrDefault()),

IdMarcaNavigation = ObtenerMarca(p.IdMarca.GetValueOrDefault())

}).ToList().FirstOrDefault();

return Ok(producto);

}

private Categoria ObtenerCategoria(int pIdCategoria)

{

Categoria cat = new Categoria();

using (TiendaVirtualDBContext BD1 = new TiendaVirtualDBContext())

{

cat = (from c in BD1.Categoria

where c.Id == pIdCategoria

select c

).ToList().First();

}

return cat;

}

private Marca ObtenerMarca(int pIdMarca)

{

Marca mar = new Marca();

using (TiendaVirtualDBContext BD2 = new TiendaVirtualDBContext())

{

mar = (from m in BD2.Marca

where m.Id == pIdMarca

select m

).ToList().First();

}

return mar;

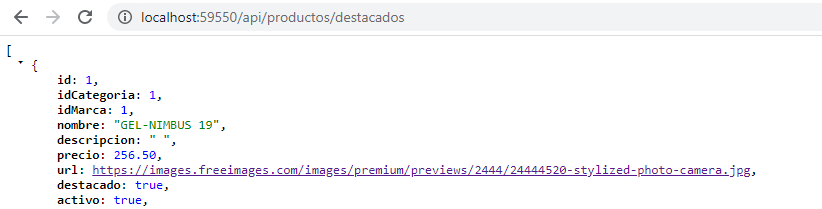
}

}

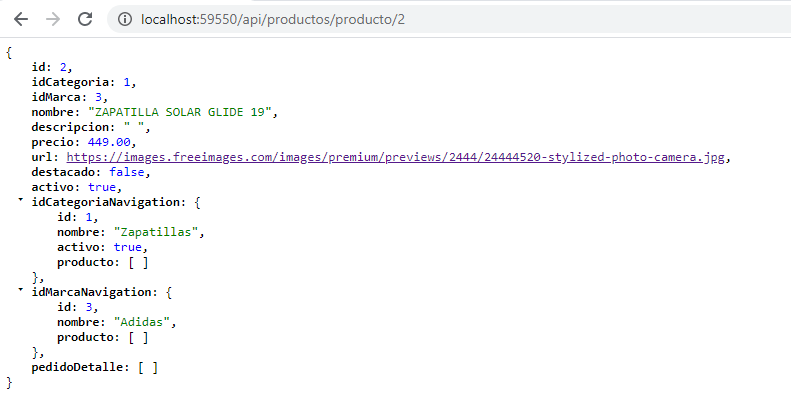
}

1. Ejecutamos la aplicación y validamos las funcionalidades que se implementaron.









Paso 3. Implementar servicios que gestionen Pedidos.

1. Agregamos el controlador API en blanco: *PedidosController.cs*
2. En el controlador agregamos la siguiente funcionalidad:
   * Método que devuelva un pedido con todos los datos de cabecera.
   * Método que permita crear un pedido con datos iniciales.
   * Método que permita actualizar los datos de un pedido.

Modificamos el código del controlador de acuerdo a:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using MiTiendaVirtualAPI.Models;

namespace MiTiendaVirtualAPI.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class PedidosController : ControllerBase

{

private readonly TiendaVirtualDBContext BD;

public PedidosController(TiendaVirtualDBContext context)

{

BD = context;

}

//GET. /api/pedidos/2

[HttpGet("{id}", Name = "PedidoCreado")]

public IActionResult Pedido(int id)

{

Pedido Pedido = new Pedido();

Pedido = (from p in BD.Pedido

where p.Id == id

select new Pedido

{

Id = p.Id,

IdCliente = p.IdCliente,

IdTarjeta = p.IdTarjeta,

FechaHora = p.FechaHora,

Estado = p.Estado,

Total = p.Total,

IdClienteNavigation = ObtenerCliente(p.IdCliente.GetValueOrDefault()),

IdTarjetaNavigation = ObtenerTarjeta(p.IdTarjeta.GetValueOrDefault())

}).ToList().FirstOrDefault();

return Ok(Pedido);

}

//POST. /api/pedidos

[HttpPost()]

//public IActionResult CrearPedido([FromBody] Pedido pPedido)

public IActionResult CrearPedido()

{

Pedido pPedido = new Pedido();

pPedido.Estado = "EN PROCESO";

pPedido.Total = 0;

if (ModelState.IsValid)

{

//guardamos el pedido en la BD

BD.Pedido.Add(pPedido);

BD.SaveChanges();

//devolvemos el pedido recientemente creado

return new CreatedAtRouteResult("PedidoCreado", new { id = pPedido.Id }, pPedido);

}

return BadRequest(ModelState);

}

//PUT. /api/pedidos/5

[HttpPut("{id}")]

public IActionResult ActualizarPedido([FromBody] Pedido pPedido, int id)

{

if (pPedido.Id != id)

{

return BadRequest();

}

BD.Entry(pPedido).State = EntityState.Modified;

BD.SaveChanges();

return Ok();

}

private Cliente ObtenerCliente(int pIdCliente)

{

Cliente cli = new Cliente();

using (TiendaVirtualDBContext BD1 = new TiendaVirtualDBContext())

{

cli = (from c in BD1.Cliente

where c.Id == pIdCliente

select c

).ToList().First();

}

return cli;

}

private Tarjeta ObtenerTarjeta(int pIdTarjeta)

{

Tarjeta tar = new Tarjeta();

using (TiendaVirtualDBContext BD2 = new TiendaVirtualDBContext())

{

tar = (from t in BD2.Tarjeta

where t.Id == pIdTarjeta

select t

).ToList().First();

}

return tar;

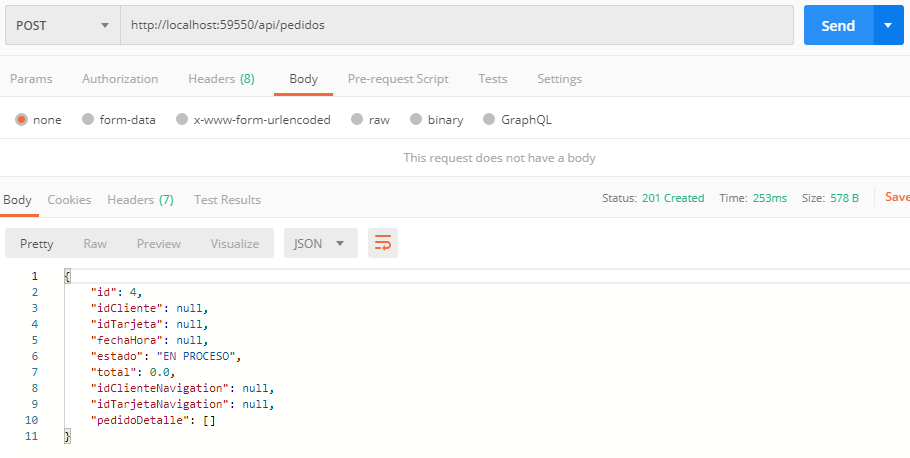
}

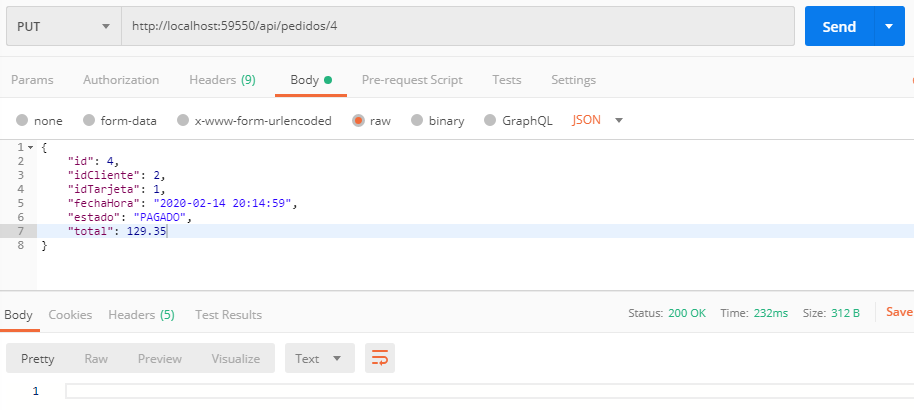
}

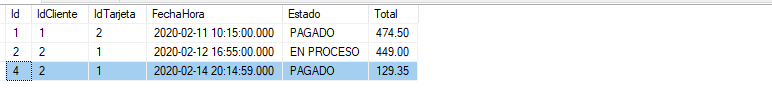
}

1. Ejecutamos la aplicación y validamos las funcionalidades que se implementaron.





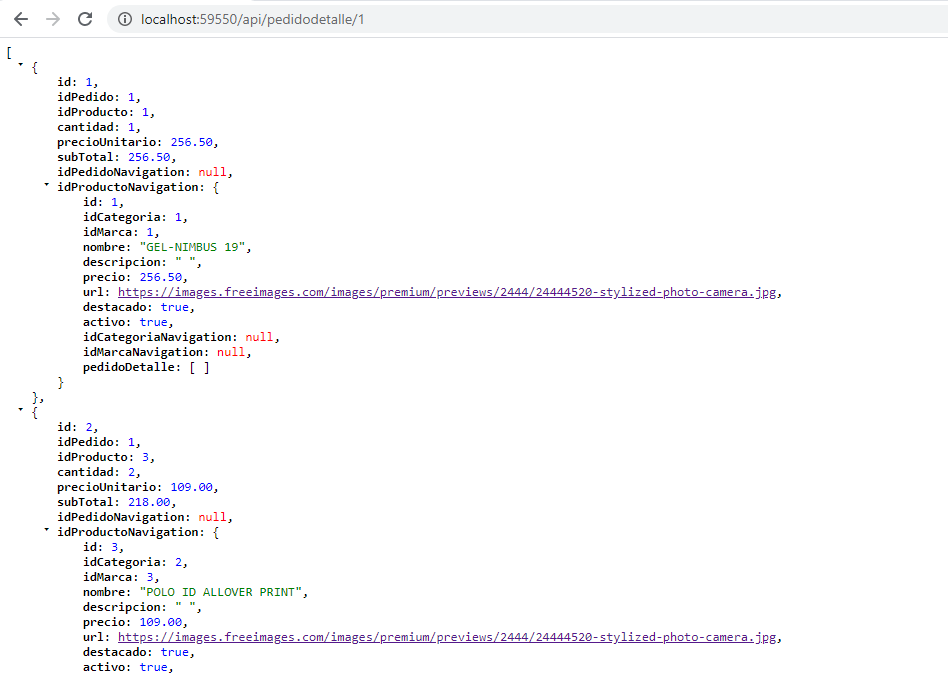




ACTIVIDAD VIRTUAL

Ingresa a la plataforma virtual. Revisa y analiza el tema desarrollado en esta sesión, luego desarrolla las siguientes actividades propuestas:

1. CUESTIONARIO TÉCNICO
   * 1. Crear el controlador en blanco *PedidoDetalle.cs,* en ella implementar los tres siguientes métodos.
     2. Método que permita agregar un producto a un pedido. Este método debe actualizar el total del pedido.
     3. Método que permita quitar un producto a un pedido. Este método debe actualizar el total del pedido.
     4. Método que permita devolver el detalle de un pedido, por ejemplo:



* + 1. Analizar en qué controlador debe implementarse el método que devuelva la cantidad de productos que se encuentran en un pedido.

1. CONCLUSIONES DE LA EXPERIENCIA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACTIVIDAD** **VIRTUAL**:

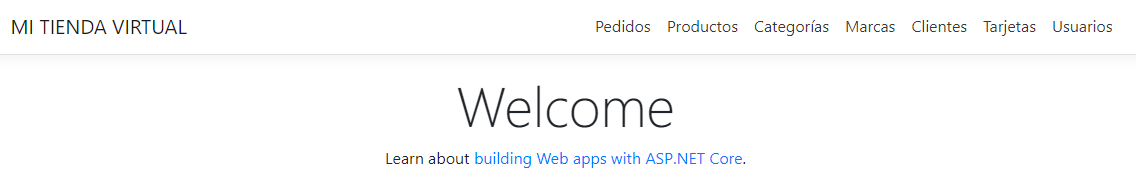
Ingresa a la plataforma virtual. Revisa y analiza el tema desarrollado en esta sesión, luego desarrolla las siguientes actividades propuestas:

1. CUESTIONARIO TÉCNICO
   * 1. Crear con Scaffold Controladores y vistas para el mantenimiento (CRUD) de las siguientes entidades (Tablas):

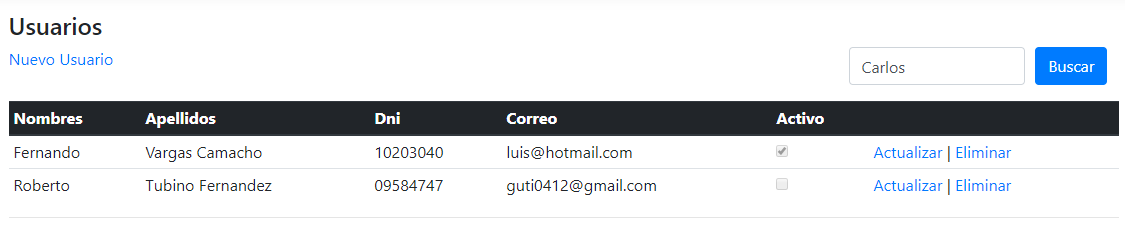
* Marca
* Categoría
* Producto
  + 1. Crear con Scaffold Controladores y vistas para el mantenimiento (CRUD) de las siguientes entidades (Tablas):
* Cliente
* Tarjeta

Estas dos entidades deben ser de solo lectura, esto es, no debe permitir crear, actualizar ni eliminar registros.

* + 1. Para los nuevos mantenimientos, desarrollar las personalizaciones de forma análoga al CRUD de Usuarios.
    2. Agregar en el menú superior links que permitan acceder a los nuevos mantenimientos. Además, alinear el menú a la derecha.



* + 1. Investigar la forma de agregar la funcionalidad de búsqueda en la lista de Usuarios, debe buscar por Nombres y Apellidos.



1. CONCLUSIONES DE LA EXPERIENCIA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Un motor paso a paso siempre debemos considerar su corriente de operación y los grados, como indica el ejemplo de la siguiente placa característica de la imagen mostrada:

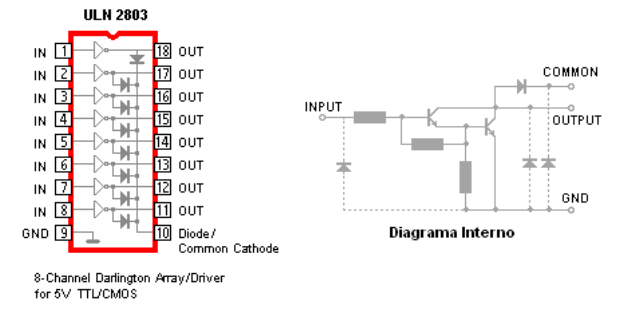


**Subtema 1.2:**

Transistores de potencia

Para operaciones con motores de bajo consumo tenemos circuitos integrados como el circuito ULN2803, el cual es un driver con transistores darlintong , con soporte de hasta 500 mA.

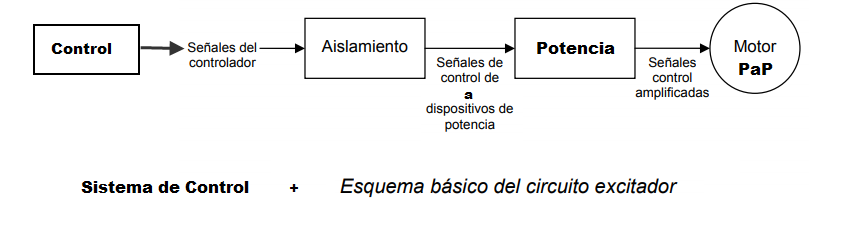
Por lo tanto si deseamos mayores corrientes debemos utilizar otros drivers o implementar uno concomponentes discretos , es decir con transistores de potencia.



**Subtema 1.3:**

Control del Motor

Las etapas para controlar un motor Paso a Paso



**Actividad:**

Los estudiantes desarrollarán ejercicios de Teoría de exponentes y radicales, propuestos por el docente. según lo aprendido en clase. a) CUESTIONARIO TÉCNICO

• ¿Qué diferencia hay entre motores paso a paso unipolares y bipolares?

• ¿Por qué es importante usar driver para motores paso a paso?

• ¿Qué características posee un motor paso a paso?

• ¿Para qué me sirve el puente H?

• Investigue sobre cómo controlar los motores paso a paso con PWM y LOGO! 8.