Herramientas de Programación 2

Tema Nº12:FORMULARIOS WEB (PARTE 1)

Indicador de logro Nº12:Diseña formularios web para la validación y registro de información en la BD siguiendo las instrucciones en REACT.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

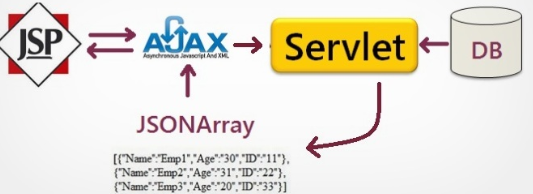
**TEMA Nº12:**

FORMULARIOS WEB (PARTE 1)

**Subtema 12.1:**

Fetching Data

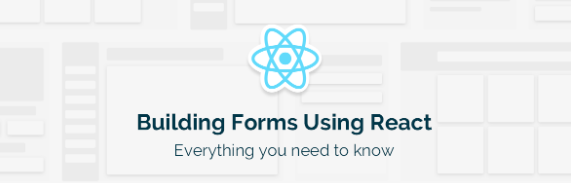
En este artículo vamos a comprender el concepto de capturar registros de filas de la tabla de la base de datos con un clic de botón. Veamos cómo se puede crear la aplicación FetchingDaTa Table.



Componentes controlados

En HTML, los elementos de formularios como los <input>, <textarea> y el <select> normalmente mantienen sus propios estados y los actualizan de acuerdo a la interacción del usuario. En React, el estado mutable es mantenido normalmente en la propiedad estado (*state*) de los componentes, y solo se actualiza con *setState()*.

Podemos combinar ambos haciendo que el estado de React sea la “única fuente de la verdad”. De esta manera, los componentes React que rendericen un formulario también controlan lo que pasa en ese formulario con las subsecuentes entradas del usuario. El componente que incluye un formulario cuyos valores son controlados por React es denominado “componente controlado”.



Paso 1. Crear el componente que contiene el formulario para agregar un producto al pedido.

1. Abrimos en Visual Studio Code el proyecto React del taller de la semana anterior.
2. Ejecutamos la aplicación para verificar que no existan errores.
3. Dentro de la carpeta *Componentes* creamos el archivo *CompFormAgregarProducto.js*. Agregamos el siguiente código jsx:

import React, { Component } from 'react';

class CompFormAgregarProducto extends Component {

    agregarProducto = e => {

        e.preventDefault();

    }

    render() {

        return <form onSubmit={this.agregarProducto}>

            <p className="h5 text-secondary"> Producto</p>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtNombre" placeholder="Nombre" />

                </div>

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtPrecio" placeholder="Precio unitario" />

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtMarca" placeholder="Marca" />

                </div>

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtCantidad" placeholder="Cantidad" />

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-12 text-right">

                    <button type="submit" className="btn btn-secondary">Agregar producto</button>

                </div>

            </div>

        </form>

    }

}

export default CompFormAgregarProducto;

1. Agregamos este componente dentro del componente *CompPedidoDetalle.js*.

import React, { Component } from "react";

import CompPedidoDetalleItem from "./CompPedidoDetalleItem";

import CompFormAgregarProducto from "./CompFormAgregarProducto";

class CompPedidoDetalle extends Component {

  render() {

    return (

      <div className="col-9">

        <CompFormAgregarProducto />

        <p className="h5 text-secondary">Pedido</p>

        <div className="card border-primary">

          <div className="card-header bg-transparent border-secondary h4 text-primary">

            <span style={{ float: "left" }}>

              Número: {this.props.pPedido.id}

            </span>

            <span style={{ float: "right" }}>

              Total S/ {this.props.pPedido.total}

            </span>

          </div>

          <div className="card-body text-secondary">

            <h5 className="card-title">Poductos:</h5>

            <table className="table">

              <thead className="thead-light">

                <tr>

                  <th scope="col" className="text-center">

                    #

                  </th>

                  <th scope="col" className="text-center">

                    Producto

                  </th>

                  <th scope="col" className="text-center">

                    Marca

                  </th>

                  <th scope="col" className="text-center">

                    Precio unitario

                  </th>

                  <th scope="col" className="text-center">

                    Cantidad

                  </th>

                  <th scope="col" className="text-center">

                    Subtotal

                  </th>

                  <th scope="col" className="text-center"></th>

                </tr>

              </thead>

              {this.props.pPedidoDetalle.map((item, indice) => {

                return (

                  <CompPedidoDetalleItem

                    pItem={item}

                    pIndice={indice}

                    key={indice}

                    pFuncEliminarProducto={this.props.pFuncionEliminarProducto}

                  />

                );

              })}

            </table>

          </div>

          <div className="card-footer bg-transparent border-secondary h4 text-secondary text-right">

            <button type="submit" className="btn btn-primary my-1">

              Finalizar compra

            </button>

          </div>

        </div>

      </div>

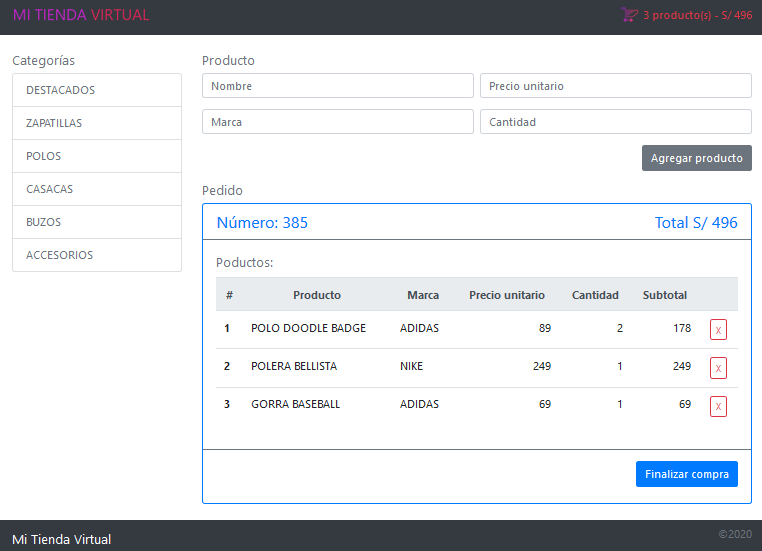
    );

  }

}

export default CompPedidoDetalle;

1. Ejecutamos la aplicación para ver los cambios.



Paso 2. Agregamos la funcionalidad de agregar un producto al pedido, para ello debemos almacenar los datos del producto en el state del componente (formulario) y luego, a través del pase de funciones entre componentes, agregamos el producto al pedido.

1. En el componente *PagePedidoDetalle.js* agregamos la función que se encarga de agregar un producto al pedido, luego le pasamos esta función al componente *CompPedidoDetalle.js*.

import React, { Component } from 'react';

import CompCabecera from './CompCabecera';

import CompListaCategorias from './CompListaCategorias';

import CompPedidoDetalle from './CompPedidoDetalle';

import CompPiePagina from './CompPiePagina';

import dataJsonPedido from '../DataJson/dataJsonPedido.json';

import dataJsonListaDetallePedido from '../DataJson/dataJsonListaDetallePedido.json';

class PagePedidoDetalle extends Component {

    state = {

        miPedido: dataJsonPedido,

        miPedidoDetalle: dataJsonListaDetallePedido

    }

    agregarProductoAlPedido = (pId, pProducto, pMarca, pPrecio, pCantidad) => {

        console.log(pProducto);

        const nuevoItemDetalle = {

            "id": pId,

            "idPedido": 385,

            "idProducto": 0,

            "cantidad": pCantidad,

            "precioUnitario": pPrecio,

            "subTotal": pCantidad \* pPrecio,

            "producto": {

                "id": 0,

                "idCategoria": 0,

                "idMarca": 0,

                "nombre": pProducto,

                "precio": pPrecio,

                "categoria": {

                    "id": 0,

                    "nombre": ""

                },

                "marca": {

                    "id": 0,

                    "nombre": pMarca

                }

            }

        };

        let nuevoPedidoDetalle =  [...this.state.miPedidoDetalle, nuevoItemDetalle];

        this.setState({

            miPedidoDetalle: nuevoPedidoDetalle

        })

        this.recalcularTotal(nuevoPedidoDetalle);

    }

    eliminarProductoDelPedido = (id) => {

        let nuevoPedidoDetalle = this.state.miPedidoDetalle.filter(itemDetalle => itemDetalle.id !== id);

        this.setState({

            miPedidoDetalle: nuevoPedidoDetalle

        });

        this.recalcularTotal(nuevoPedidoDetalle);

    }

    recalcularTotal = (pDetalle) => {

        let total = 0;

        let pedido = this.state.miPedido;

        pDetalle.map(itemDetalle => total = total + itemDetalle.subTotal);

        pedido.total = total;

        this.setState({

            miPedido: pedido

        })

    }

    render() {

        return <div >

            <CompCabecera pCantidadProductos={this.state.miPedidoDetalle.length} pTotal={this.state.miPedido.total}/>

            <main role="main" className="container">

                <div className="row">

                    <CompListaCategorias />

                    <CompPedidoDetalle pPedido={this.state.miPedido}

                        pPedidoDetalle={this.state.miPedidoDetalle}

                        pFuncionEliminarProducto={this.eliminarProductoDelPedido}

                        pFuncionAgregarProducto={this.agregarProductoAlPedido}/>

                </div>

            </main>

            <br />

            <CompPiePagina />

        </div>

    }

}

export default PagePedidoDetalle;

1. El componente *CompPedidoDetalle.js* solo se encargará de pasar la función *pFuncionAgregarProducto* al componente *CompFormAgregarProducto.js*.

import React, { Component } from "react";

import CompPedidoDetalleItem from "./CompPedidoDetalleItem";

import CompFormAgregarProducto from "./CompFormAgregarProducto";

class CompPedidoDetalle extends Component {

  render() {

    return (

      <div className="col-9">

        <CompFormAgregarProducto pFuncionAgregarProducto={this.props.pFuncionAgregarProducto}/>

        <p className="h5 text-secondary">Pedido</p>

        <div className="card border-primary">

          <div className="card-header bg-transparent border-secondary h4 text-primary">

            <span style={{ float: "left" }}>

              Número: {this.props.pPedido.id}

            </span>

            <span style={{ float: "right" }}>

              Total S/ {this.props.pPedido.total}

            </span>

          </div>

          <div className="card-body text-secondary">

            <h5 className="card-title">Poductos:</h5>

            <table className="table">

              <thead className="thead-light">

                <tr>

                  <th scope="col" className="text-center">

                    #

                  </th>

                  <th scope="col" className="text-center">

                    Producto

                  </th>

                  <th scope="col" className="text-center">

                    Marca

                  </th>

                  <th scope="col" className="text-center">

                    Precio unitario

                  </th>

                  <th scope="col" className="text-center">

                    Cantidad

                  </th>

                  <th scope="col" className="text-center">

                    Subtotal

                  </th>

                  <th scope="col" className="text-center"></th>

                </tr>

              </thead>

              {this.props.pPedidoDetalle.map((item, indice) => {

                return (

                  <CompPedidoDetalleItem

                    pItem={item}

                    pIndice={indice}

                    key={indice}

                    pFuncEliminarProducto={this.props.pFuncionEliminarProducto}

                  />

                );

              })}

            </table>

          </div>

          <div className="card-footer bg-transparent border-secondary h4 text-secondary text-right">

            <button type="submit" className="btn btn-primary my-1">

              Finalizar compra

            </button>

          </div>

        </div>

      </div>

    );

  }

}

export default CompPedidoDetalle;

1. En el componente *CompFormAgregarProducto.js* ejecutamos la función *pFuncionAgregarProducto*.

import React, { Component } from 'react';

class CompFormAgregarProducto extends Component {

    state = {

        id: 0,

        nombre: "",

        marca: "",

        precio: "",

        cantidad: ""

    }

    agregarProducto = e => {

        this.setState({

            id: this.state.id + 1

        });

        this.props.pFuncionAgregarProducto(this.state.id, this.state.nombre, this.state.marca, this.state.precio, this.state.cantidad);

        e.preventDefault();

    }

    actualizarEstado = e => {

        if (e.target.name == "txtNombre"){

            this.setState({

                nombre: e.target.value

            })

        }else if(e.target.name == "txtMarca"){

            this.setState({

                marca: e.target.value

            })

        }else if(e.target.name == "txtPrecio"){

            this.setState({

                precio: e.target.value

            })

        }else if(e.target.name == "txtCantidad"){

            this.setState({

                cantidad: e.target.value

            })

        }

    }

    render() {

        return <form onSubmit={this.agregarProducto}>

            <p className="h5 text-secondary"> Producto</p>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtNombre" placeholder="Nombre"

                        value={this.state.nombre} onChange={this.actualizarEstado}/>

                </div>

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtPrecio" placeholder="Precio unitario"

                        value={this.state.precio} onChange={this.actualizarEstado}/>

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtMarca" placeholder="Marca"

                        value={this.state.marca} onChange={this.actualizarEstado}/>

                </div>

                <div className="form-group col-md-6">

                    <input type="text" className="form-control" name="txtCantidad" placeholder="Cantidad"

                        value={this.state.cantidad} onChange={this.actualizarEstado}/>

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-12 text-right">

                    <button type="submit" className="btn btn-secondary">Agregar producto</button>

                </div>

            </div>

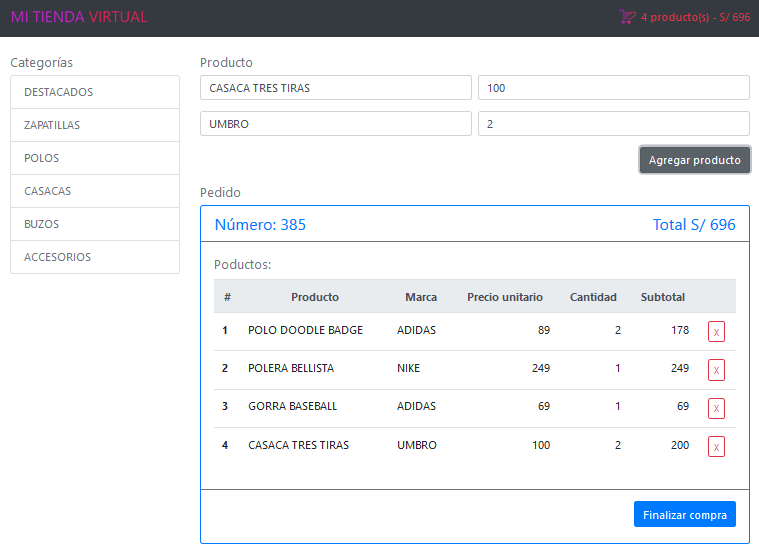
        </form>

    }

}

export default CompFormAgregarProducto;

1. En el explorador web agregamos productos al pedido para validar su funcionalidad.



Paso 3. Validar datos en un formulario. Además, como en el ejemplo anterior, se tendrá en cuenta que los datos del formulario tendrán su correspondiente valor en el estado del componente.

1. Crear dentro de la carpeta *Componentes* el componente *CompFormAgregarCliente.js* con el siguiente contenido:

import React, { Component } from 'react';

class CompFormAgregarCliente extends Component {

    state = {

        id: 0,

        nombres: "",

        apellidos: "",

        dni: "",

        telefono: "",

        correo: "",

        direccion: "",

        mensajeErrorNombres: "",

        mensajeErrorApellidos: "",

        mensajeErrorDni: "",

        mensajeErrorCorreo: ""

    }

    validarDatosCliente = e => {

        let valido = false;

        this.setState({

            mensajeErrorNombres: "",

            mensajeErrorApellidos: "",

            mensajeErrorDni: "",

            mensajeErrorCorreo: ""

        })

        if (this.state.nombres == null || this.state.nombres.length == 0 || /^\s+$/.test(this.state.nombres)){

            this.setState({

                mensajeErrorNombres: "Campo obligatorio"

            })

        } else if (this.state.apellidos == null || this.state.apellidos.length == 0 || /^\s+$/.test(this.state.apellidos)) {

            this.setState({

                mensajeErrorApellidos: "Campo obligatorio"

            })

        } else if (this.state.dni == null || this.state.dni.length == 0 || /^\s+$/.test(this.state.dni)) {

            this.setState({

                mensajeErrorDni: "Campo obligatorio"

            })

        } else if (!(/^\d{8}$/.test(this.state.dni))) {

            this.setState({

                mensajeErrorDni: "Ingrese ocho números"

            })

        } else if (this.state.correo.length > 0 && !(/\S+@\S+\.\S+/.test(this.state.correo))) {

            this.setState({

                mensajeErrorCorreo: "Ingrese un correo correcto"

            })

        } else {

            alert("Datos correctos.");

        }

        e.preventDefault();

    }

    actualizarEstado = e => {

        if (e.target.name == "txtNombres"){

            this.setState({

                nombres: e.target.value

            })

        }else if(e.target.name == "txtApellidos"){

            this.setState({

                apellidos: e.target.value

            })

        }else if(e.target.name == "txtDni"){

            this.setState({

                dni: e.target.value

            })

        }else if(e.target.name == "txtTelefono"){

            this.setState({

                telefono: e.target.value

            })

        }else if(e.target.name == "txtCorreo"){

            this.setState({

                correo: e.target.value

            })

        }else if(e.target.name == "txtDireccion"){

            this.setState({

                direccion: e.target.value

            })

        }

    }

    render() {

        return <form onSubmit={this.validarDatosCliente}>

            <p className="h5 text-secondary">Datos del Cliente</p>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <label for="txtNombres">\*Nombres:</label>

                    <input type="text" className="form-control" name="txtNombres" id="txtNombres"

                        value={this.state.nombres} onChange={this.actualizarEstado}/>

                    <div className="badge badge-danger">{this.state.mensajeErrorNombres}</div>

                </div>

                <div className="form-group col-md-6">

                    <label for="txtTelefono">Teléfono:</label>

                    <input type="text" className="form-control" name="txtTelefono" id="txtTelefono"

                        value={this.state.telefono} onChange={this.actualizarEstado}/>

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <label for="txtApellidos">\*Apellidos:</label>

                    <input type="text" className="form-control" name="txtApellidos" id="txtApellidos"

                        value={this.state.apellidos} onChange={this.actualizarEstado}/>

                    <div className="badge badge-danger">{this.state.mensajeErrorApellidos}</div>

                </div>

                <div className="form-group col-md-6">

                    <label for="txtCorreo">Correo:</label>

                    <input type="text" className="form-control" name="txtCorreo" id="txtCorreo"

                        value={this.state.correo} onChange={this.actualizarEstado}/>

                    <div className="badge badge-danger">{this.state.mensajeErrorCorreo}</div>

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-6">

                    <label for="txtDni">\*Dni:</label>

                    <input type="text" className="form-control" name="txtDni" id="txtDni"

                        value={this.state.dni} onChange={this.actualizarEstado}/>

                    <div className="badge badge-danger">{this.state.mensajeErrorDni}</div>

                </div>

                <div className="form-group col-md-6">

                    <label for="txtDireccion">Dirección:</label>

                    <input type="text" className="form-control" name="txtDireccion" id="txtDireccion"

                        value={this.state.direccion} onChange={this.actualizarEstado}/>

                </div>

            </div>

            <div className="form-row">

                <div className="form-group col-md-12 text-right">

                    <button type="submit" className="btn btn-primary">Agregar</button>

                </div>

            </div>

        </form>

    }

}

export default CompFormAgregarCliente;

1. Crear dentro de la carpeta *Componentes* el componente *PagePedidoCliente.js* con el siguientecontenido (este componente contedrá el formulario del componente anterior)

import React, {Component} from 'react';

import CompCabecera from './CompCabecera';

import CompFormAgregarCliente from './CompFormAgregarCliente';

import CompPiePagina from './CompPiePagina';

class PagePedidoCliente extends Component {

  render(){

    return  <div >

                <CompCabecera />

                <main role="main" className="container">

                    <CompFormAgregarCliente />

                </main>

                <CompPiePagina />

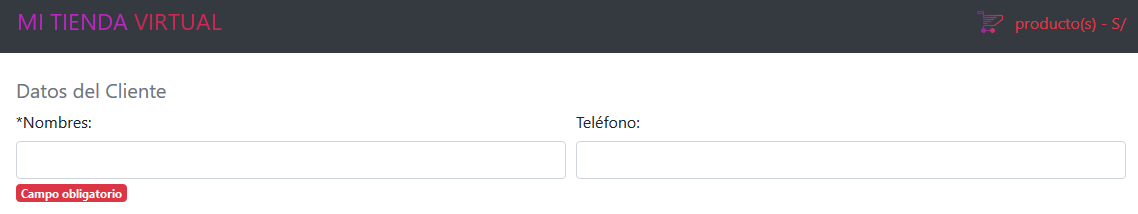
            </div>

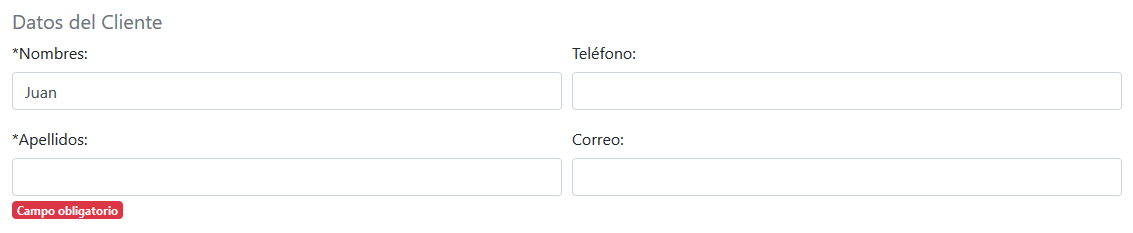
  }

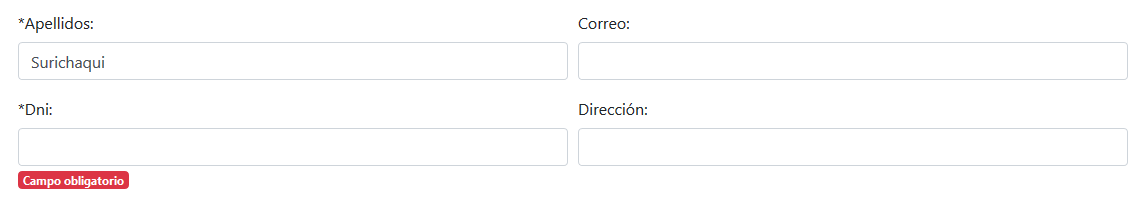
}

export default PagePedidoCliente;

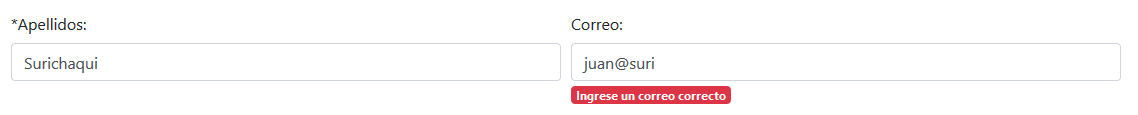
1. Modificamos el archivo *App.js* para que se muestre inicialmente el componente *PagePedidoCliente.*
2. Ejecutamos la aplicación y revisamos las validaciones.













ACTIVIDAD VIRTUAL:

1. CUESTIONARIO TÉCNICO
   * 1. En el primer ejemplo (Detalle del Pedido) agregar las validaciones de campos obligatorios para los campos *Nombre*, *Precio* y *Cantidad*.
     2. En el primer ejemplo (Detalle del Pedido) agregar en el formulario el campo *Categoría* y mostrarlo en una columna de la lista de productos del pedido.
     3. En el segundo ejemplo (Formulario del Cliente) agregar la siguiente validación en el campo Dirección: No es obligatorio, pero si se ingresa debe tener una longitud máxima de 100 caracteres, de no cumplir este requisito debe mostrarse debajo del campo el mensaje “Ingresar como máximo 100 caracteres”.
     4. En el segundo ejemplo (Formulario del Cliente) agregar la siguiente validación en el campo *Teléfono*: No es obligatorio, pero si se ingresa debe tener una longitud de 6 a 15 caracteres, de no cumplir este requisito debe mostrarse debajo del campo el mensaje “Ingrese un teléfono de 6 a 15 caracteres”.
     5. Investigar la forma de consumir servicios web RESTful desde aplicaciones React.
2. CONCLUSIONES DE LA EXPERIENCIA

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

**ACTIVIDAD** **VIRTUAL**:

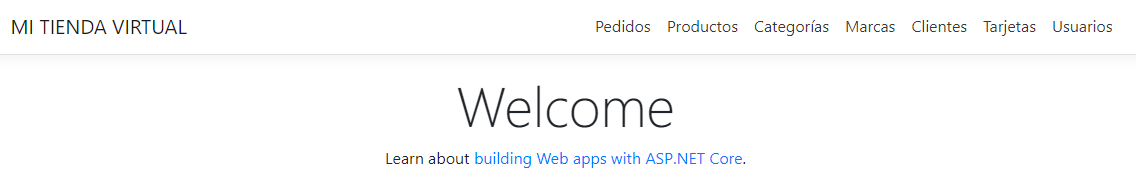
Ingresa a la plataforma virtual. Revisa y analiza el tema desarrollado en esta sesión, luego desarrolla las siguientes actividades propuestas:

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

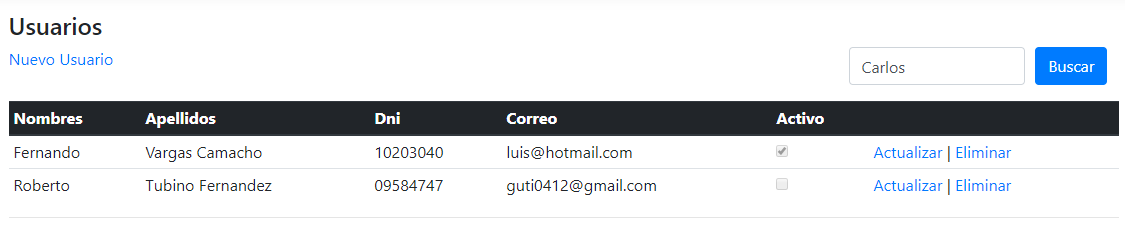
* Marca
* Categoría
* Producto
  + 1. Crear con Scaffold Controladores y vistas para el mantenimiento (CRUD) de las siguientes entidades (Tablas):
* Cliente
* Tarjeta

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:

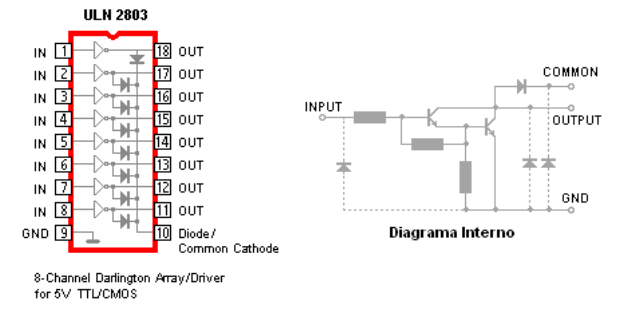


**Subtema 1.2:**

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.

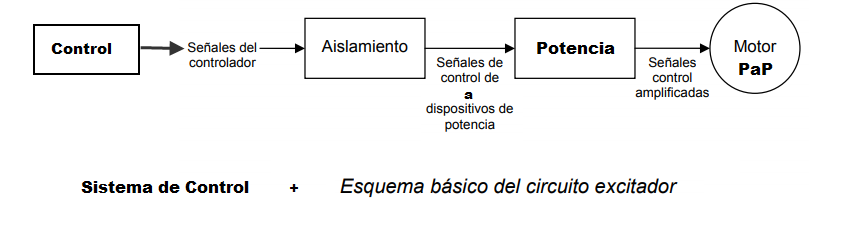
Por lo tanto si deseamos mayores corrientes debemos utilizar otros drivers o implementar uno concomponentes discretos , es decir con transistores de potencia.



**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.