MANUAL TECNICO

APLICATIVO ORIENTADO A UNA TIENDA ONLINE DISEÑADO PARA LA EMPRESA DISTRIANDINA DEDICADA A SURTIR PAPELEIRAS

MARIA PAULA CASTRO TELLEZ

MARIA ALEJANDRA RUIZ GONZALEZ

LUIS MIGUEL TORRES RODRIGUEZ

CRISTIAN CAMILO PUENTES CÉSPEDES

FICHA 1193354

BOGOTA D.C

CONTENIDO

1. INTRODUCCION

En este manual encontrara detalladamente documentación técnica acerca del aplicativo orientado a una tienda online.

El manual servirá de soporte para el desarrollador en el momento en el cual se requiera un mantenimiento o implementación de diferentes mejoras.

2.OBJETIVOS

2.1 OBJETIVO GENERAL

Brindar la información acerca de la creación e implementación del software a quien pueda interesar.

2.2 OBJETIVOS ESPECIFICOS

- Documentación de código
- Definir procedimientos del aplicativo
- Describir las herramientas usadas para el desarrollo del aplicativo

3.DICCIONARIO DE DATOS

ADMINISTRADOR

1. Data Dictionary

Entity Name	Entity Description					
Column Name	**Column Description**	**Data Type**	**Length**	**Primary Key**	**Nullable**	**Unique**
ADMINISTRADOR						
Cargo	Rol el cual se identifica	varchar	25	false	false	false
Cedula	aquel numero que tiene solo esta persona	numeric	19	true	false	false
PERSONAUsuario		varchar	15	false	false	false
Sexo	Genero el cual se identifica (masculino o femenino)	char	1	false	false	false
CATEGORIA						
Id	Codigo del producto	numeric	19	true	false	false
Nombre	Aquel con el cual se identifica cada producto	varchar	25	false	false	false
CLIENTE						
C_Administrador	Aquella con al cual puede ingresar al sistema para su rol determinado	numeric	19	false	false	false
N_Administrador	Nombre con el cual se identifica y se le da este tipo de rol con ciertos privilegios	varchar	255	false	false	false
NIT	El numero de identifiacion tributaria,conocido tambien por el acronimo NIT, es un numero unico colombiano,que asigna la DIAN a las empresas o personas naturales o juridicas	numeric	19	true	false	false
PERSONAUsuario		varchar	15	false	false	false

DETALLE PEDIDO-PRODUCTO

DETALLE PEDIDO-PRODUCTO						
Cant	El numero de ciertos pedidos o preoductos disponibles	numeric	19	false	false	false
PEDIDON_Factura		numeric	19	false	false	false
PrecioUnitario	Valor monetario de cada producto	real	10	false	false	false

Entity Name	Entity Description					
PRODUCTOCodigo		numeric	19	false	false	false

DETALLE PROVEEDOR-PRODUCTO

DETALLE PROVEEDOR-PRODUCTO						
Cant	El numero de ciertos pedidos o preoductos disponibles	numeric	19	false	false	false
PrecioSugerido	Valor por mayor de cada producto	real	10	false	false	false
PRODUCTOCodigo		numeric	19	false	false	false
PROVEEDORNit		numeric	19	false	false	false

ESTADO

ESTADO						
F_Entrega	Dia en el cual se muetra cuando sera entregado el pedido	varchar	25	false	false	false
Id		bit	0	true	false	false
PEDIDON_Factura	Factura la cual se muetra el o los pedidos que se solicitan	numeric	19	false	false	false

FORMA DE PAGO

FORMA DE PAGO						
Seleccion	Opcion a elegir	bit	0	true	false	false

PEDIDO

PEDIDO						
CLIENTENIT		numeric	19	false	false	false
ESTADOId		bit	0	false	false	false
Fecha	Es el dia el mes y el año donde se realiza dicho pedido	date	0	false	false	false
FORMA DE PAGOSeleccion		bit	0	false	false	false
Iva	Incremento al valor agregado establecido por la ley del 16%	real	25	false	false	false
N_Factura	Aquel numero o codigo que se identifica cada pedido	numeric	19	true	false	false
P_Neto	Valor a pagar	real	10	false	false	false
Total	Suma todos los precios de los productos adquridos	real	10	false	false	false

PERSONA

PERSONA						
Apellido	Aquel Complemento del nombre de la persona registrada	varchar	25	false	false	false

Entity Name	Entity Description					
Cel	Otro tipo de numero para podernos comunicar para dar o resibir algun tipo de informacion atraves de la comunicacion	numeric	19	false	false	false
Contraseña	Clave o es una forma de aunteticacion donde se utliza informacion secreta para controlar el acceso haci algun recurso	varchar	10	false	false	false
Direccion	Lugar donde podemos encontrar alguna ubicacion especifica por medio de numero y letras	varchar	25	false	false	false
Email	Correo electrónico; donde podemos , enviar y recibir mensajes a través de sistemas de comunicación	varchar	25	false	false	false
Nombre	Aquel que se identifica o se registra dentro del sistema	varchar	25	false	false	false
tel	Forma o numero donde podemos comunicarnos	numeric	19	false	false	false
Usuario	Un usuario es un individuo que se utiliza para clasificar a diferentes privilegios ,permisos alos que tiene acceso un usuario o grupo de ususario	varchar	15	true	false	false

PRODUCTO

PRODUCTO						
Caracteristicas	Son las describe cierto producto	varchar	255	false	false	false
CATEGORIAId2		numeric	19	false	false	false
Codigo	Numero que se identifica cada producto	numeric	19	true	false	false
Imagen	Foto del producto	bit	0	false	false	false
Marca	Lo que identifica cada producto o el nombre de la empresa	varchar	25	false	false	false
Nombre	Aquel que se identifica cierto producto	varchar	25	false	false	false
PrecioCompra	Valor monetario de la suma de todos los productos	real	10	false	false	false
PrecioVenta	Valor monetario el cual se vende	real	10	false	false	false

PROVEEDOR

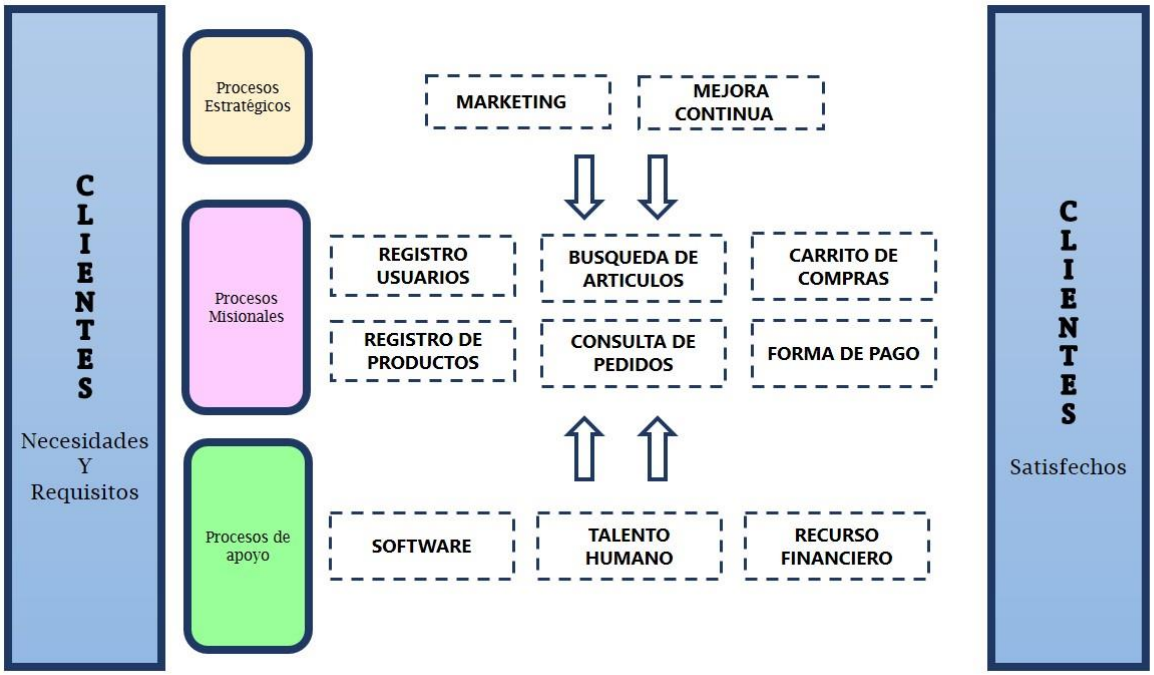
PROVEEDOR

Entity Name	Entity Description					
Direccion	Lugar donde se ubica esta persona	varchar	25	false	false	false
Email	Correo electronico con el cual podemos tener sierta comunicacion	varchar	25	false	false	false
Nit	Numero de persona natural con el cual se identifica este proveedor	numeric	19	true	false	false
Nombre	Aquel que se identifica	varchar	25	false	false	false
Telefono	Numero donde podemos llamarlo para poder cierta comunicacion	numeric	19	false	false	false

VENDEDOR

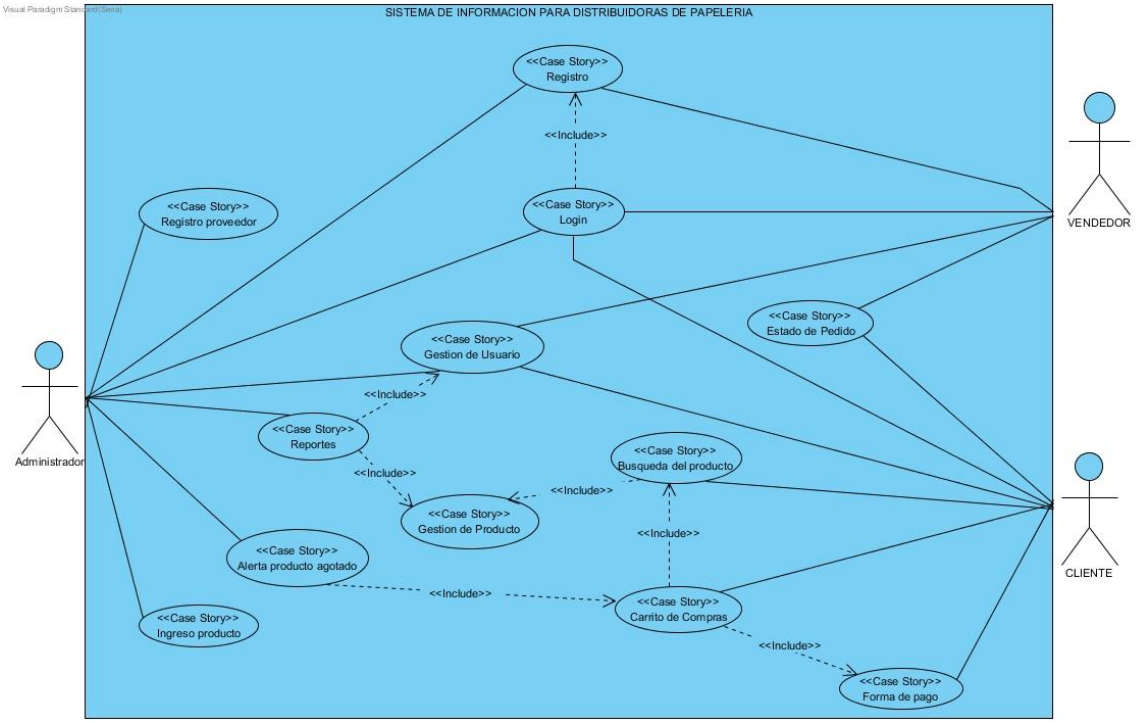
VENDEDOR						
Cedula	Aquel numero que tiene solo esta persona	numeric	19	true	false	false
PERSONAUsuario		varchar	15	false	false	false
Sexo	Genero el cual se identifica (masculino o femenino)	char	1	false	false	false
Zona	Lugar donde surte o ofrece los productos	varchar	25	false	false	false

4. DESCRIPCION DE PROCESOS

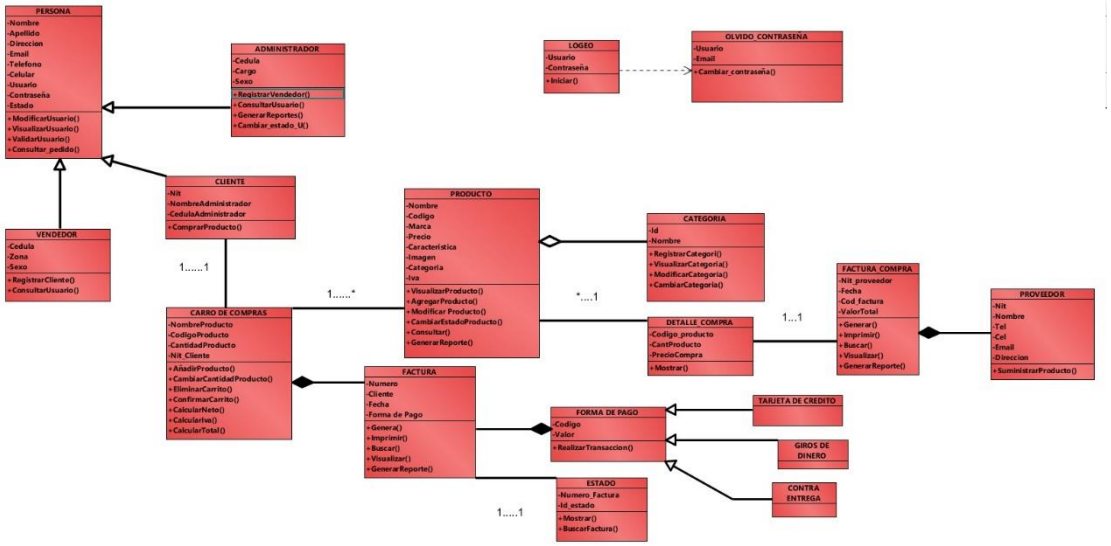


5. DIAGRAMAS UML

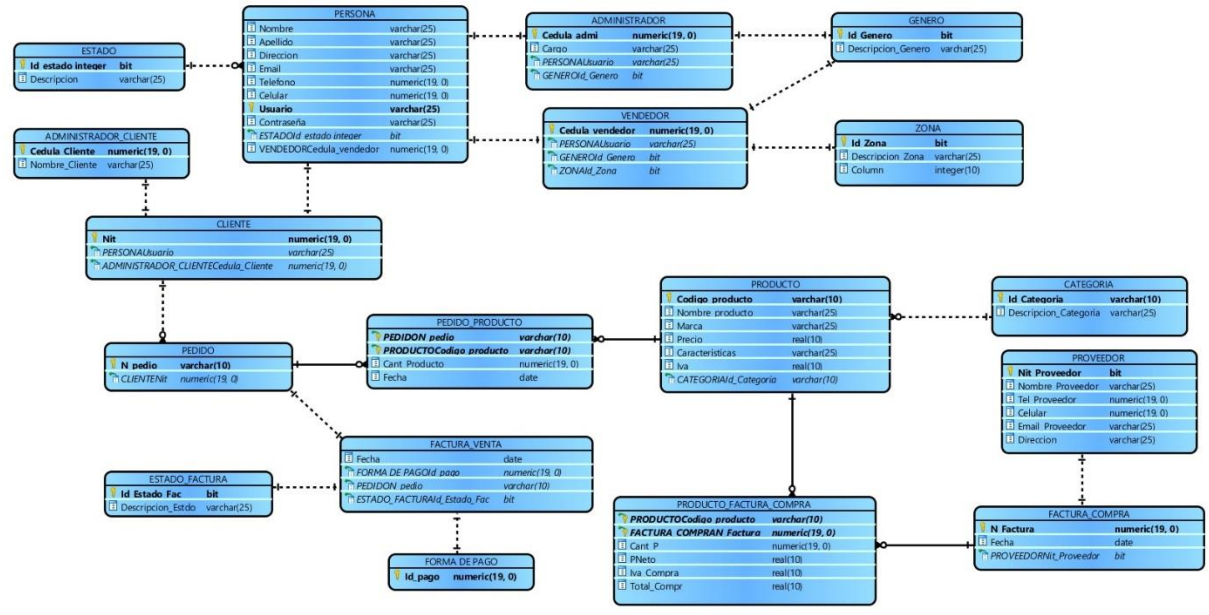
5.1 DIAGRAMA DE CASOS DE USO



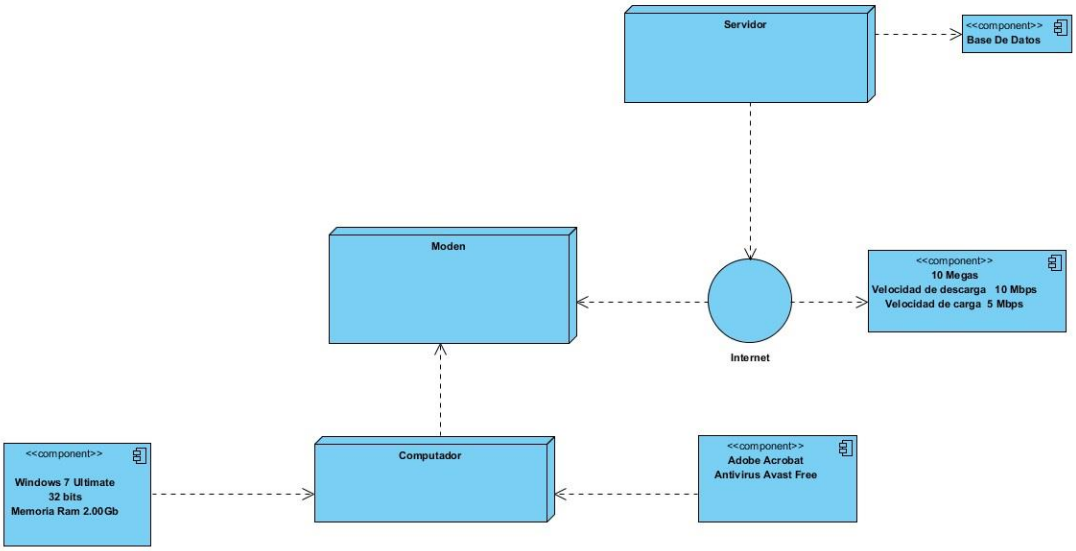
5.2 DIAGRAMA DE CLASES



5.3 DIAGRAMA RELACIONAL



5.4 DIAGRAMA DE DISTRIBUCION



ACCESO AL SISTEMA DE INFORMACION

```php
<?php
require_once 'core/core.php';
if(!isset($_REQUEST['c']))
{
    $controller="index";
    require_once 'controller/'.$controller.'Controller.php';
    $controller = $controller.'Controller';
    $controller = new $controller;
    $controller->viewIndex();
}
else
{
    $controller=$_REQUEST['c'];
    require_once 'controller/'.$controller.'Controller.php';
    $controller = $controller.'Controller';
    $controller = new $controller;
    $metodo= isset($_REQUEST['m']) ? $_REQUEST['m'] : 'index';
    call_user_func(array($controller,$metodo));
}
```

En el siguiente código se muestra el acceso al sistema de información para ingresar por parámetros y acceder al las clases a través de métodos y controladores por el modelo de desarrollo (MVC)

INDEX En este código HTML se visualiza los diferentes roles por los cuales se va a acceder al sistema de información

```
<div class="row">
    <div class="col-md-9">
        <img src="assets/img/papeleria.jpg" class="img-responsive">
    </div>
    <div class="col-md-3">
        <br><br>
        <div class="pepe">
            <ul class="nav nav-tabs">
                <li class="active"><a data-toggle="tab" href="#cliente">Cliente</a></li>
                <li><a data-toggle="tab" href="#vendedor">Vendedor</a></li>
                <li><a data-toggle="tab" href="#admin">Administrador</a></li>
            </ul>
```

Se visualiza un formulario con los campos usuario y contraseña seguida de un botón que validara los campos en la base de datos para ingresar por el rol que se ha escogido.

```
        <label>Usuario: </label>
        <div class="input-group">
          <span class="input-group-addon"><i class="glyphicon glyphicon-user"></i></span>
          <input id="user" type="text" class="form-control" name="user" placeholder="Usuario" required="">
        </div>
        <hr>
        <label>Contrasena: </label>
        <div class="input-group">
            <span class="input-group-addon"><i class="glyphicon glyphicon-lock"></i></span>
            <input id="password" type="password" class="form-control" name="password" placeholder="Contraseña" required="">
        </div>
        <hr>
        <div class="input-group">
            <input id="rol" type="hidden" class="form-control" name="rol" value="client">
        </div>
        <button type="submit" class="btn btn-primary">INICIAR SESION </button>
    </form>
</div>
<div id="vendedor" class="tab-pane fade">
    <form action="?c=index&m=validarData" method="post">
        <label>Usuario: </label>
        <div class="input-group">
          <span class="input-group-addon"><i class="glyphicon glyphicon-user"></i></span>
          <input id="user" type="text" class="form-control" name="user" placeholder="Usuario" required="">
        </div>
        <hr>
        <label>Contrasena: </label>
        <div class="input-group">
            <span class="input-group-addon"><i class="glyphicon glyphicon-lock"></i></span>
            <input id="password" type="password" class="form-control" name="password" placeholder="Contraseña" required="">
        </div>
        <hr>
        <div class="input-group">
            <input id="rol" type="hidden" class="form-control" name="rol" value="seller">
        </div>
        <button type="submit" class="btn btn-primary">INICIAR SESION </button>
    </form>
```

En botón accede al método “**validarData**” de la clase indexController

```
<form action="?c=index&m=validarData" method="post">
```

En el método **validarData** se reciben los valores que ingresaron en el formulario para validarlos en el método queryUserId pasando como parámetro la variable $_REQUEST[‘user’];

```php
public function validarData() {
    echo $_REQUEST['user'];
    $_REQUEST['password'];
    $_REQUEST['rol'];
    $r = $this->modelIndex->queryUserId($_REQUEST['user']);
    echo (var_dump($_REQUEST['password']));
```

En el método queryUserId se hace la consulta a la base de datos que se encuentra en el archivo prepareSQL para retornar todos los campos que encuentre la consulta

```php
public function queryUserId($user)
{
    try {
        $stm= $this->pdo->prepare(PreparedSQL::QUERY_USER_ID);
        $stm->bindParam(1, $user, PDO::PARAM_STR);
        $stm->execute();
        return $stm->fetch(PDO::FETCH_OBJ);
    } catch (Exception $e) {
        Die ($e->getMessage());
    }
}
```

Después de hacer la consulta se comparan las variables y si se encuentran en la base de datos registrados da ingreso al sistema con el método viewHome y si no se encuentra en la base de datos y no hace la validación redirecciona al método viewIndex para volver a digitar los datos.

```php
public function validarData() {
    echo $_REQUEST['user'];
    $_REQUEST['password'];
    $_REQUEST['rol'];
    $r = $this->modelIndex->queryUserId($_REQUEST['user']);
    echo (var_dump($_REQUEST['password']));

    if (($r->usuario_id == $_REQUEST['user']) && ($_REQUEST['password']==$r->usuario_password) && ($r->usuario_rol == $_REQUEST['rol'])) {
        $data = array("usuario" => $r->usuario_id,
            "password" => $r->usuario_password,
            "rol" => $r->usuario_rol);
        //INSANCIA DEL METODO DE SEGURIDAD QUE INICIALZIA LAS VARIABLES DE SESION
        $this->modelSecurity->iniciarSesion($data);

        header("location:?c=index&m=viewHome");
    } else {
        echo ("Pendejo");
        header("location:?c=index&m=viewIndex");
    }
}
```

El archivo home.php contiene el script, el html y las consultas que hace a la base de datos para visualizar los productos más recientes. También el html el cual visualiza un carrusel

```
<div class="container-fluid seccion"><!--ABRE LA SECCION -->
    <div class="container">
        <div id="myCarousel" class="carousel slide" data-ride="carousel">
            <!-- Indicators -->
            <ol class="carousel-indicators">
                <li data-target="#myCarousel" data-slide-to="0" class="active"></li>
                <li data-target="#myCarousel" data-slide-to="1"></li>
                <li data-target="#myCarousel" data-slide-to="2"></li>
            </ol>

            <!-- Wrapper for slides -->
            <div class="carousel-inner">
                <div class="item active">
                    <a href="?c=index&m=viewProducts"><img src="assets/img/carrusel1.jpg" alt="Distriandina" style="width:100%;"></a>
                </div>

                <div class="item">
                    <img src="assets/img/carrusel2.jpg" alt="Distriandina" style="width:100%;">
                </div>

                <div class="item">
                    <img src="assets/img/carrusel3.jpg" alt="Distriandina" style="width:100%;">
                </div>
                <a href="home.php"></a>
            </div>

            <!-- Left and right controls -->
            <a class="left carousel-control" href="#myCarousel" data-slide="prev">
                <span class="glyphicon glyphicon-chevron-left"></span>
                <span class="sr-only">Previous</span>
            </a>
            <a class="right carousel-control" href="#myCarousel" data-slide="next">
                <span class="glyphicon glyphicon-chevron-right"></span>
                <span class="sr-only">Next</span>
            </a>
        </div>
    </div>
    <br><br>
```

El siguiente código visualiza el carrito de compras y si se encuentra vació da un mensaje de tu carrito esta vació con un botón que lleva la interfaz de compras

```
<center><h2 style="font-weight: bold; border-radius: 20px; background: #F3E053; padding: 10px; color: black;">NOVEDADES</h2></center><br><br><br>
<?php foreach ($this->modelIndex->queryProducts() as $r) {
    ?>
    <div class="col-md-4">
        <div class="producto">
            <div class="product-pic">
                <a href="?c=index&m=viewFichaProduct&id=<?php echo $r->producto_id; ?>"><img src="<?php echo $r->producto_imagen; ?>" alt="" /></a>
            </div>
            <div class="product-nombre"><a href="?c=index&m=viewFichaProduct&id=<?php echo $r->producto_id; ?>"><p><?php echo $r->producto_nombre; ?></p></a></div><br>
            <a class="precio" href="?c=Carrito&m=viewCarrito&id=<?php echo $r->producto_id; ?>">$<?php echo $r->producto_total; ?><span class="glyphicon glyphicon-shopping-cart
        </div>
    </div>
```

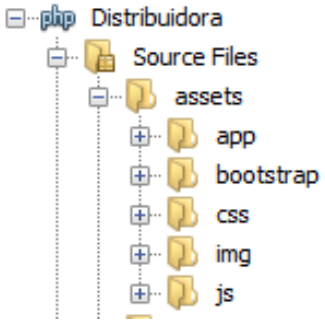
Modelo vista controlador (MVC)

El sistema de información se encuentra con el modelo de desarrollo (MVC)

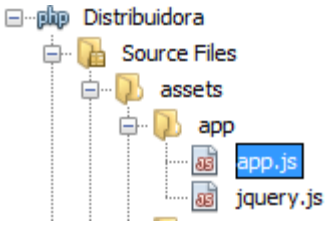
A continuación mostraremos detalladamente los códigos por los cuales se ingresa a las clases de cada módulo como es el caso del proyecto se tienen tres roles Administrador Vendedor y cliente cada uno con su respectivo controlador.

ASSETS

La carpeta assets contiene otras carpetas con librerías que ayudan que el diseño del aplicativo sea agradable a la vista de los usuarios



En la carpeta app encontramos dos archivos java que leen Ajax para los mensajes en pantalla que se realizan según el método que se requiera en otro archivo.



El método **disminuirCantidad** toma el id pasado por url recibiendo como parameto la variable data en el cual va al modelo carrito y accede al metod **disminuirCantidad** para hacer la consulta a la base de datos y luego recarga la pagina mostrando la modificacion que se realizo

```
function disminuirCantidad(data) {

    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Carrito&m=disminuirCantidad',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El método **modificarInventario** recibe un parámetro para acceder y consecuentemente actualiza el inventario en el método **updateInventario** Que accede a la base de datos y actualiza el inventario luego lo muestra en el formulario inventario ya con las cantidades actualizadas

```
function modificarInventario(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=Admin&m=updateInventarioModal',
        data: urls,
        success(response)
        {
            $('#resultUpdateInventario').html(response);
        }

    });
    $('#modificarInventario').modal('show');
}
```

El método **aumentarCantidad** lee el id que se envia por url y accede al metodo aumentarCantidades aumenta las cantidades en el carrito de compras y luego recarga la pagina con las cantidades ya sumadas

```
function aumentarCantidad(data) {

    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Carrito&m=aumentarCantidad',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El método **eliminarProductCar** elimina las cantidades que se suman al carrito de compras

```
function eliminarProductCar(data) {

    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Carrito&m=eliminarProCar',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El método **activarModalClient** recibe como parámetro una variable llamada data para actualizar los datos de un cliente desde el controlador del perfil vendedor

```
function activarModalClient(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=Seller&m=updateClientModal',
        data: urls,
        success(response)
        {
            $('#resultUpdateCliente').html(response);
        }

    });
    $('#actualizarCliente').modal('show');
}
```

El método **desactivarClient** recibe como parámetro el id que se le envía por la url y genera una condición en la cual se verifica la aceptación o cancelación del usuario para abrir el modal luego cambia el estado del cliente

```
function desactivarClient(data) {

    //alert(data);
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Seller&m=deleteClient',
        data: urls,
        success(response) {
            location.reload();

        }
    });
}
```

El método **activarCliente** recibe como parametro el id del cliente por url y lo envia a la funcion del vendedor para generar la condicion y activar el estado del cliente

```
function activarClient(data) {

    //alert(data);
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Seller&m=activeClient',
        data: urls,
        success(response) {
            location.reload();

        }
    });
}
```

El método **updatePassword** recibe tres parámetros para acceder y consecuentemente accede al método que se encuentra en el modelo índex que contiene el método **updatePassword** para actualizar la contraseña del usuario que genera la solicitud.

```
function updatePassword(actual, nueva, nuevaR)
{
    var data =
            {
                "actual": actual,
                "nueva": nueva,
                "nuevaR": nuevaR
            };

    $.ajax({
        type: 'post',
        url: '?c=index&m=updatePassword',
        data: data,
        sucess(response)
        {
            alert(response);
            $('#errores').html(response);
        }
    });
}
```

EL método **desactivarCategoría** recibe un parámetro y accede al modelo del administrador para acceder al método **deletecat** para desactivar la categoría en la base de datos y luego recargar la pagina

```
function desactivarCategoria(data) {
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=deleteCat',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

EL método **activarCategoria** recibe un parámetro y accede al modelo del administrador para acceder al método **activecat** para activar la categoría en la base de datos y luego recargar la pagina

```
function activarCategoria(data) {

    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=activeCat',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El metodo **activarModalCamara** accede al modelo del administrador y el metodo **updateimagenModal** para consecuentemente actualizar la imagen que se recibe como parámetro

```
function activarModalCamara(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=Admin&m=updateImagenModal',
        data: urls,
        success(response)
        {
            $('#resultUpdateImagen').html(response);
        }

    });
    $('#editarImagen').modal('show');
}
```

La función **activarModalImagen** accede al modelo del index y el metodo **ImagenModal** para consecuentemente actualizar la imagen que se recibe como parámetro en la variable data

```
function activarModalImagen(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=index&m=ImagenModal',
        data: urls,
        success(response)
        {
            $('#resultUpdateImagen').html(response);
        }

    });
    $('#editarImagen').modal('show');
}
```

El metodo **activarModalCategoria** accede al modelo del administrador para acceder al metodo **updateCategoriaModal** para actualizar la categoría

```
function activarModalCategoria(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=Admin&m=updateCategoriaModal',
        data: urls,
        success(response)
        {
            $('#resultUpdateCategoria').html(response);
        }

    });
    $('#editarCategoria').modal('show');
}
```

El metodo **desactivarVendedor** recibe un parámetro y luego accede al modelo del administrador para acceder al metodo **deleteSeller** para luego desactivar el estado del vendedor que se pasó por el parámetro data

```
function desactivarVendedor(data) {


    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=deleteSeller',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El metodo **activarVendedor** accede al modelo del administrador para luego acceder al metodo **activeSeller** para luego cambiar el estado del vendedor pasando como parametro la variable data

```
function activarVendedor(data) {

    //alert(data);
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=activeSeller',
        data: urls,
        success(response) {
            location.reload();
        }
    });
}
```

El metodo **activarProduct** accede al modelo del administrador para luego acceder al metodo **activeProduct** para luego cambiar el estado de un producto que se envía el parámetro que recibe la variable data

```
function activarProduct(data) {

    //alert(data);
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=activeProduct',
        data: urls,
        success(response) {
            location.reload();

        }
    });
}
```

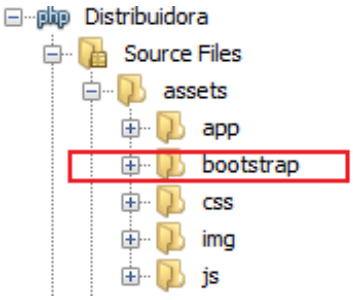
El metodo **desactivarProduct** accede al modelo del administrador para acceder al metodo **deleteProduct** para luego desactivar el producto que se le envia como parametro que recibe la variable data

```
function desactivarProduct(data) {

    //alert(data);
    var urls = "id=" + data;
    $.ajax({

        type: 'post',
        url: '?c=Admin&m=deleteProduct',
        data: urls,
        success(response) {
            location.reload();

        }
    });
}
```

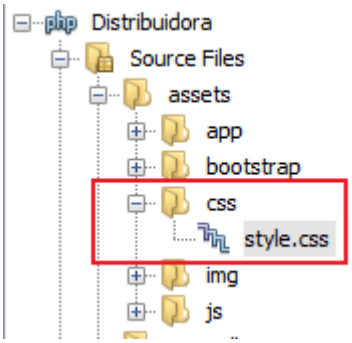
El metodo **activarModal** accede al modelo del administrador para acceder al metodo **updateProductoModal** que actualiza la información del producto según el id que se le envía por parámetro con la variable data

```
function activarModal(data) {
    var urls = "documento=" + data;
    $.ajax({
        type: 'post',
        url: '?c=Admin&m=updateProductModal',
        data: urls,
        success(response)
        {
            $('#resultUpdate').html(response);

    });
    $('#editarProducto').modal('show');
}
```

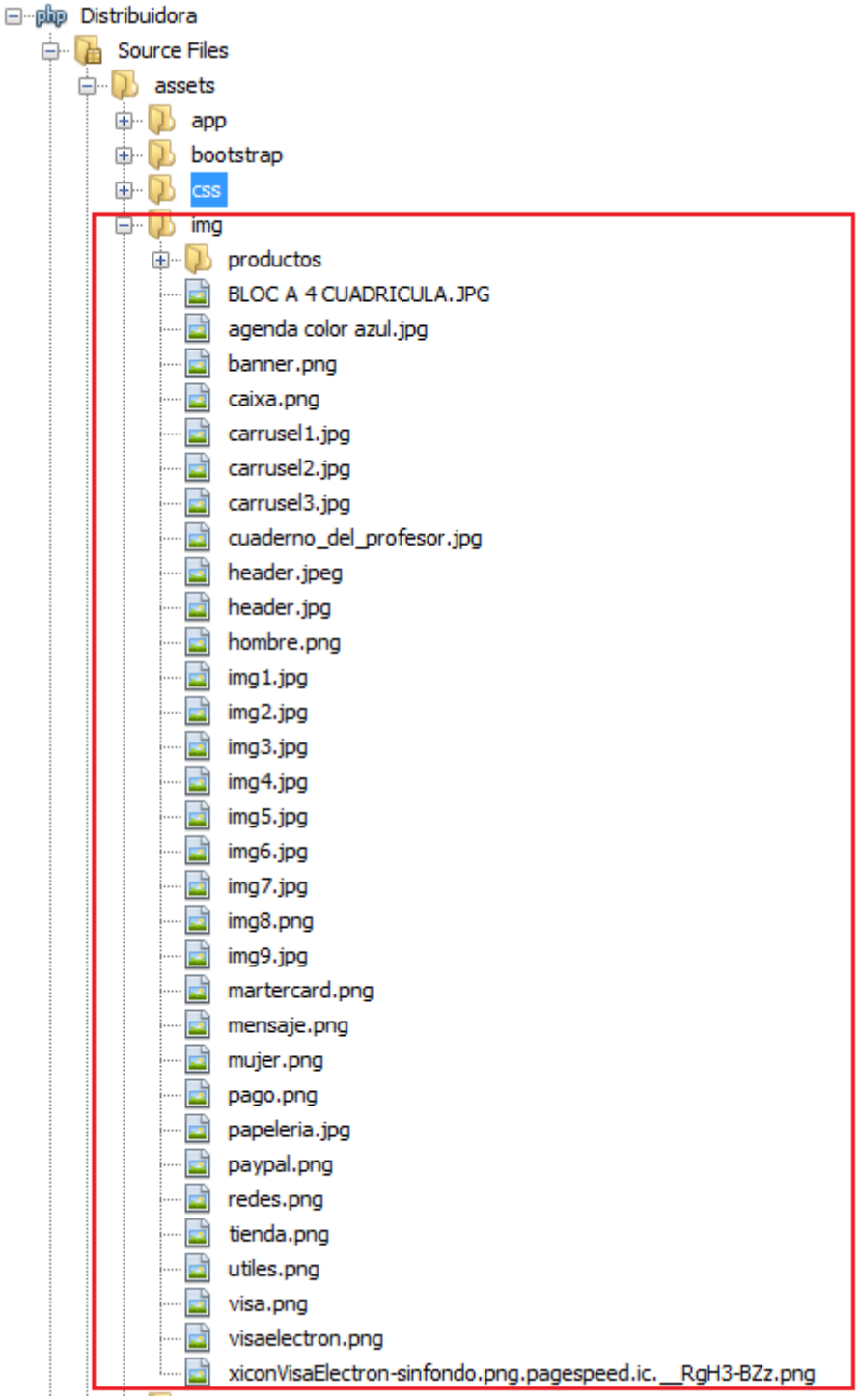
La carpeta bootstrap es un framework que ayuda al diseño del aplicativo con códigos que ya vienen por defecto.



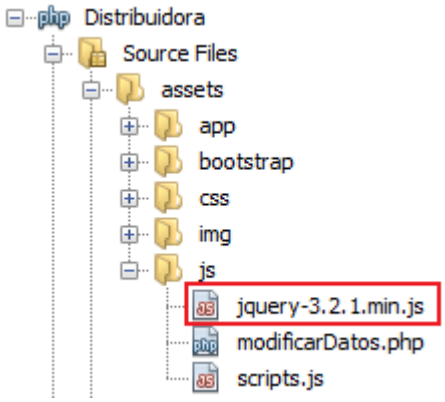
La carpeta css contiene un archivo style.css que contiene estilos css para que la visualizacion de los formularios botones e imágenes esten correctamente posicionados ademas de darles tamaños y colores para una optima visuzalizacion para el usuario



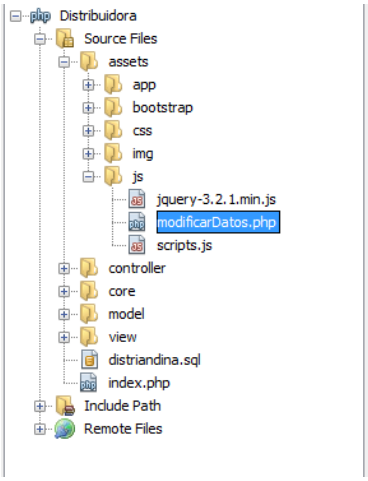
La carpeta img contiene las imágenes que se guardan directamente desde el aplicativo

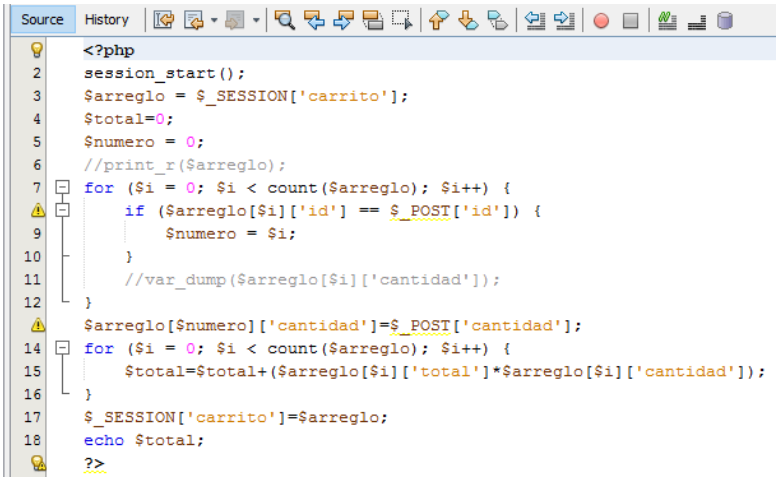


la carpeta js contiene una librería de jquery

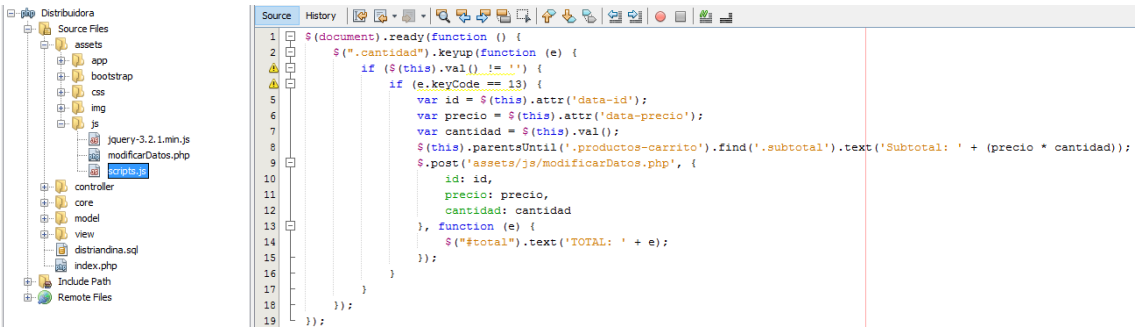


En el archivo **modificarDatos.php** se encuentra las cantidades que se van sumando en el carrito de compras con un arreglo para iniciar con el metodo **sesión_start();**

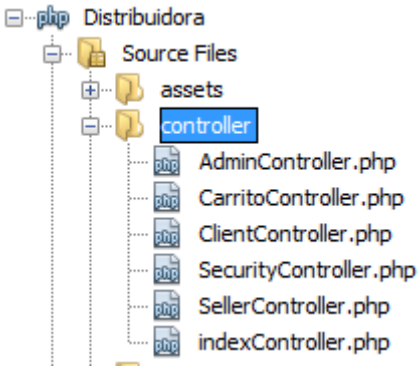




El archivo **scripts.js** contiene el total y las cantidades que se van sumando al carrito de compras



La carpeta **controller** contiene los controlladores para acceder a los modelos por las clases que contiene cada rol para acceder a las diferentes funciones en el aplicativo



AdminController.php

El archivo **AdminController.php** contiene una clase llamada AdminController que inicia con las variables $modelAdmin, $preparedSQL y $modelSecurity

```php
<?php

class AdminController {

    private $modelAdmin;
    private $preparedSQL;
    private $modelSecurity;
```

El constructor de la clase AdminController ingresa a en las variables nuevos objetos descritos de la siguiente manera:

$this->modelAdmin = new admin();

Contiene un objeto admin para acceder a los diferentes métodos que tiene esta clase.

$this->preparedSQL = new PreparedSQl();

Contiene un objeto para ingresar a una clase final que contiene las consultas de tipo sql a la base de datos para su óptimo funcionamiento.

$this->modelSecurity = new Security ();

Contiene un objeto para dejar ingresar a los usuarios en los diferentes roles que se encuentren dándole una protección al sistema ante los diferentes ataques informáticos que se puedan llegar a presentar y así mismo impide el acceso de los distintitos roles para solo acceder mediante un rol a las diferentes interfaces del aplicativo

$this->modelSecurity = validarSesionAdmin();

A la variable modelSecurity se le agrega la el metodo de validarAdmin (); para verificar que el usuario que ingresa a la clase entra con el rol de administrador

Atraves de un try catch se valida la información si esto resulta ser falso no ingresas a la clase

```
public function __construct() {
    try {
        $this->modelAdmin = new Admin();
        $this->preparedSQL = new PreparedSQL();
        $this->modelSecurity = new Security();
        $this->modelSecurity->validarSesionAdmin();
    } catch (Exception $e) {
        Die($e->getMessage());
    }
}
```

El metodo **viewPerfil ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, perfil.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewPerfil() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/perfil.php';
    require_once 'view/all/footer.php';
}
```

b

El metodo **viewModificarPerfil ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, modificarperfil.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewModificarPerfil() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/modificarPerfil.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewModificarPassword ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, modificarContraseña.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewModificarPassword() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/modificarContrasena.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewInsertSeller ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, formInsertSeller.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewInsertSeller() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/formInsertSeller.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewListSeller ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, listSeller.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewListSeller() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/listSeller.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewInventarioProduct ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, inventarioProduct.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewInventarioProduct() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/inventarioProduct.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewInsertProduct ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, formInsertProduct.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewInsertProduct() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/formInsertProduct.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewListProduct ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, listProduct.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewListProduct() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/listProduct.php';
    require_once 'view/all/footer.php';
}
```

El metodo **viewInsertCat ()** requiere los diferentes archivos como header.php navbar.php, navbarRol.php, formInsertCat.php y footer.php que se encuentran en la carpeta view cada archivo en su respectiva carpeta

```php
public function viewInsertCat() {
    require_once 'view/all/header.php';
    require_once 'view/all/navbar.php';
    require_once 'view/all/navbarRol.php';
    require_once 'view/admin/formInsertcat.php';
    require_once 'view/all/footer.php';
}
```

El metodo **insertseller()** recibe como parametro los valores del formulario para ingresarlos en un array y luego enviarlos al modelo del administrador al metodo **insertSellers**($data); y se le envia el la variable $data que contiene los datos ingresados en el formularios en un array luego recarga la vista para insertar un vendededor.

```php
public function insertSeller($documento,$nombre,$apellido,$direccion,$email,$telefono,$celular,$zona,$genero) {
    $contraseña = password_hash("A" . $documento, PASSWORD_DEFAULT);
    $data = array(
        "usuario" => $documento,
        "password" =>  $contraseña,
        "nombre" => $nombre,
        "apellido" => $apellido,
        "documento" => $documento,
        "direccion" => $direccion,
        "email" => $email,
        "telefono" => $telefono,
        "celular" =>$celular,
        "zona" => $zona,
        "imagen" => $genero,
    );
    $this->modelAdmin->insertSellers($data);
    header('location:?c=Admin&m=viewInsertSeller');
}
```

El metodo **updateSeller()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **updateSellers**($data); en el modelo del administrador que abre la clase Admin. Este metodo recibe y envia datos para actualizar un vendedor

```php
public function updateSeller() {
    $data = array(
        "usuario" => "" . $_POST['usuario'] . "",
        "nombre" => "" . $_POST['nombre'] . "",
        "apellido" => "" . $_POST['apellido'] . "",
        "documento" => "" . $_POST['documento'] . "",
        "direccion" => "" . $_POST['direccion'] . "",
        "email" => "" . $_POST['email'] . "",
        "telefono" => "" . $_POST['telefono'] . "",
        "celular" => "" . $_POST['celular'] . "",
        "zona" => "" . $_POST['zona'] . "",
        "imagen" => "" . $_POST['genero'] . "",
    );
    $this->modelAdmin->updateSellers($data);
    header('location:?c=Admin&m=viewListSeller');
}
```

El metodo **deletSeller()** ingresa el dato recibido por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **deleteSellers**($data); en el modelo del administrador que abre la clase Admin. Este metodo recibe y envia datos para cambiar a estado inactivo a vendedores

```php
public function deleteSeller() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "0");
    $this->modelAdmin->deleteSellers($data);
}
```

El metodo **activeSeller()** ingresa el dato recibido por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **activeSellers**($data); en el modelo del administrador que abre la clase Admin. Este metodo recibe y envia datos para cambiar a estado activo a vendedores

```php
public function activeSeller() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "1");
    $this->modelAdmin->activeSellers($data);
}
```

El metodo **modificarAdmin()** ingresa el dato recibido por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **modifAdmin**($data); en el modelo del administrador que abre la clase Admin.

Este metodo recibe y envia datos para modificar datos del administrador

```php
public function modificarAdmin() {
    $data = array(
        "usuario" => "" . $_POST['user'] . "",
        "nombre" => "" . $_POST['nombre'] . "",
        "apellido" => "" . $_POST['apellido'] . "",
        "documento" => "" . $_POST['documento'] . "",
        "direccion" => "" . $_POST['direccion'] . "",
        "email" => "" . $_POST['email'] . "",
        "telefono" => "" . $_POST['telefono'] . "",
        "celular" => "" . $_POST['celular'] . "",);
    $this->modelAdmin->modifAdmin($data);
    header('location:?c=Admin&m=viewPerfil');
}
```

El metodo **insertProduct()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **insertProducto**($data); en el modelo del administrador que abre la clase Admin Este metodo recibe y envia datos para insertar productos

```php
public function insertProduct() {
    $rutaEnServidor = 'assets/img/productos';
    $rutaTemporal = $_FILES['imagen']['tmp_name'];
    $nombreImagen = $_FILES['imagen']['name'];
    $rutaDestino = $rutaEnServidor . "/" . $nombreImagen;
    move_uploaded_file($rutaTemporal, $rutaDestino);
    $fechaRegistro = date("y-m-d");
    $iva = $_POST['iva'] / 100;
    $valorTotal = $_POST['precio'] + ($_POST['precio'] * $iva);

    $data = array(
        "nombre" => "" . $_POST['nombre'] . "",
        "marca" => "" . $_POST['marca'] . "",
        "precio" => "" . $_POST['precio'] . "",
        "iva" => "" . $_POST['iva'] . "",
        "imagen" => "$rutaDestino",
        "caracteristicas" => "" . $_POST['caracteristicas'] . "",
        "categoria" => "" . $_POST['categoria'] . "",
        "fechaIngreso" => "$fechaRegistro",
        "valorTotal" => "$valorTotal",);
    $this->modelAdmin->insertProducto($data);
    header('location:?c=Admin&m=viewInsertProduct');
}
```

El metodo **upadateImagenProduct()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **updateImagenProducto**($data); en el modelo del administrador que abre la clase Admin.

Este metodo recibe y envia datos para actualizar la imagen de los productos

```php
public function updateImagenProduct() {
    $rutaEnServidor = 'assets/img/productos';
    $rutaTemporal = $_FILES['imagen']['tmp_name'];
    $nombreImagen = $_FILES['imagen']['name'];
    $rutaDestino = $rutaEnServidor . "/" . $nombreImagen;
    move_uploaded_file($rutaTemporal, $rutaDestino);

    $data = array(
        "id" => "" . $_REQUEST['id'] . "",
        "imagen" => "$rutaDestino",
    );
    $this->modelAdmin->updateImagenProducto($data);
    header('location:?c=Admin&m=viewListProduct');
}
```

El metodo **upadateProduct()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **updateProducto**($data); en el modelo del administrador que abre la clase Admin Este metodo recibe y envia datos para actualizar datos de un producto

```php
public function updateProduct() {

    $iva = $_POST['iva'] / 100;
    $valorTotal = $_POST['precio'] + ($_POST['precio'] * $iva);

    $data = array("id" => "" . $_POST['id'] . "",
        "nombre" => "" . $_POST['nombre'] . "",
        "marca" => "" . $_POST['marca'] . "",
        "precio" => "" . $_POST['precio'] . "",
        "iva" => "" . $_POST['iva'] . "",
        "valorTotal" => "$valorTotal",
        "caracteristicas" => "" . $_POST['caracteristicas'] . "",
        "categoria" => "" . $_POST['categoria'] . "",);
    $this->modelAdmin->updateProducto($data);
    header('location:?c=Admin&m=viewListProduct');
}
```

El metodo **deleteProduct()** ingresa el dato recibido por el formulario y a al campo ESTADO añade un 0 para cambiar el campo en la base de datos. Ingresa los datos a un array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **deleteProductos**($data); en el modelo del administrador que abre la clase Admin.

Este metodo recibe y envia datos para cambiar de estado a inactivo un producto

```
public function deleteProduct() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "0");
    $this->modelAdmin->deleteProducts($data);
    header("location:?c=Admin&m=viewInsertProduct");
}
```

El metodo **activeProduct()** ingresa el dato recibido por el formulario y a al campo ESTADO añade un 1 para cambiar el campo en la base. ingresa los datos a un array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **activeProductos**($data); en el modelo del administrador que abre la clase Admin.

Este metodo recibe y envia datos para cambiar a estado activo un producto

```
public function activeProduct() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "1");
    $this->modelAdmin->activeProducts($data);
    header("location:?c=Admin&m=viewInsertProduct");
}
```

El metodo **insertCategoria()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **insertCategoria** ($data); en el modelo del administrador que abre la clase Admin. Este metodo recibe y envia datos para insertar categorias

```
public function insertCategoria() {
    $data = array("NOMBRE" => "" . $_POST['nombre'] . "");
    $this->modelAdmin->insertCategoria($data);
    header("location:?c=Admin&m=viewInsertCat");
}
```

El metodo **updateCat()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **updateCats**($data); en el modelo del administrador que abre la clase Admin.

Este metodo recibe y envia datos para actualizar categorias

```
public function updateCat() {
    $data = array("NOMBRE" => "" . $_POST['nombre'] . "",
        "ID" => "" . $_POST['id'] . "",);
    $this->modelAdmin->updateCats($data);
    header("location:?c=Admin&m=viewInsertCat");
}
```

El metodo **deleteCat()** ingresa el dato recibido por el formulario y a al campo ESTADO añade un 0 para cambiar el campo en la base de datos los ingresa a un array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **deleteCats** ($data); en el modelo del administrador que abre la clase Admin

Este metodo recibe y envia datos para cambiar a estado inactivo una categoria

```
public function deleteCat() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "0");
    $this->modelAdmin->deleteCats($data);
    header("location:?c=Admin&m=viewInsertProduct");
}
```

El metodo **activeCat()** ingresa el dato recibido por el formulario y a al campo ESTADO añade un 1 para cambiar el campo en la base de datos los ingresa a un array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **activeCats** ($data); en el modelo del administrador que abre la clase Admin

Este metodo recibe y envia datos para cambiar a estado activo una categoria

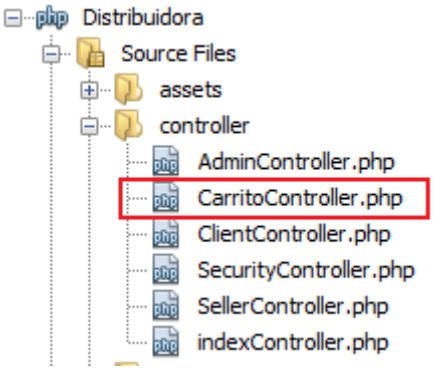
```
public function activeCat() {
    $data = array("ID" => "" . $_REQUEST['id'] . "", "ESTADO" => "1");
    $this->modelAdmin->activeCats($data);
    header("location:?c=Admin&m=viewInsertProduct");
}
```

El metodo **updateCat()** ingresa los datos recibidos por el formulario en una array o vector en una variable llamada $data. Luego envia la variable a un metodo llamado **updateCats**($data); en el modelo del administrador que abre la clase Admin. Este metodo sirve para sumar cantidades en un inventario

Este metodo recibe y envia datos para actualizar las cantidades del inventario

```
public function updateInventario()
{
    $cantidad=$_POST['cantActual']+$_POST['cantAgregar'];
    $data = array("ID" => "".$_POST['id']."","CANT"=>$cantidad);
    $this->modelAdmin->updateInventarioProd($data);
    header("location:?c=Admin&m=viewInventarioProduct");
}
```

CarritoController.php



El archivo CarritoController.php contiene la clase CarritoController con unas variables globales que son:

private $modelCarrito;

private $preparedSQL;

private $modelSecurity;

private $modelProducto;

private $modelAdmin;

```
class CarritoController {

    private $modelCarrito;
    private $preparedSQL;
    private $modelSecurity;
    private $modelProducto;
    private $modelAdmin;
```

El constructor de la clase CarritoController se definen en cada variable global objetos que permitan el acceso a los diferentes métodos que se puedan llegar a usar de las diferentes clases

$this->modelProducto = new Producto();

Crea un nuevo objeto de la clase producto

$this->modelCarrito = new Carrito();

Crea un nuevo objeto de la clase carrito

$this->preparedSQL = new PreparedSQL();

Crea una

$this->modelSecurity = new Security();

$this->modelAdmin = new Admin();

```
public function __construct() {
    try {
        $this->modelProducto = new Producto();
        $this->modelCarrito = new Carrito();
        $this->preparedSQL = new PreparedSQL();
        $this->modelSecurity = new Security();
        $this->modelAdmin = new Admin();
    } catch (Exception $e) {
        Die($e->getMessage());
    }
}
```

DESCRIPCION DE ERRORES CODIGO

Se validan que los datos ingresados estén correctamente digitados según los parámetros establecidos

UsuarioErrorfunción para verificar si el usuario tiene mas de 8 caracteres

```
function usuarioError($var)
{
    //Strlen: Obtiene la longitud de un String
    $resultado=strlen($var);
    if (empty($var))
    {
        $errorUsuario = "\n Debe llenar el campo.";
    }
    elseif($resultado<=8)
    {
        $errorUsuario = "\n El usuario debe tener mas de 8 caracteres.";
    `
```

La función contrasenaError contiene la validación de que una contraseña cumpla los parámetros para dar seguridad al aplicativo

```
function contrasenaError($var)
{
    //Strlen: Obtiene la longitud de un String
    $resultado=strlen($var);
    if (empty($var))
    {
        $errorContrasena="\n Debe llenar el campo";
    }
    elseif($resultado<=8)
    {
        $errorContrasena = "\n La contraseña debe tener mas de 8 caracteres.";
    }
    elseif (!preg_match("/^[a-zA-Z0-9*#$-_.]*$/",$var))
    {
        $errorContrasena="\n Sólo se permiten letras, numeros y *#$-_.";
    }
    if (isset ($errorContrasena))
    return $errorContrasena;
```

La función nombreError valida que los caracteres ingresados sean de tipo String (letra)

Y no permite que se ingresen otros tipos de datos

```php
function nombreError($var)
{
    if (empty($var))
    {
        $errorNombre = "\n Debe llenar el campo.";
    }
    elseif (!preg_match("/^[a-zA-Z ]*$/",$var))
    {
        $errorNombre = "\n Sólo se permiten letras y espacios en blanco.";
    }

    if (isset ($errorNombre))
    return $errorNombre;
}
```

La función apellidoError valida que los caracteres ingresados sean de tipo String (letra)

Y no permite que se ingresen otros tipos de datos

```php
function apellidoError($var)
{
    if (empty($var))
    {
        $errorApellido = "\n Debe llenar el campo.";
    }
    elseif (!preg_match("/^[a-zA-Z ]*$/",$var))
    {
        $errorApellido = "\n Sólo se permiten letras y espacios en blanco.";
    }

    if (isset ($errorApellido))
    return $errorApellido;
}
```

La función direcciónError valida que el usuario digite valores en el campo

```php
function direccionError($var)
{
    if (empty($var))
    {
        $errorDireccion = "\n Debe llenar el campo.";
    }
    if (isset ($errorDireccion))
    return $errorDireccion;
}
```

La función emailError valida que el usuario digite un correo electrónico para ello se lee un método FILTER_VALIDATE_EMAIL el cual determina los parámetros ingresados por un correo los cuales son @ seguido de unos caracteres y el .com

```php
function emailError($var)
{
    if (empty($var))
    {
        $errorEmail="\n Debe llenar el campo.";
    }
    elseif (!filter_var($var, FILTER_VALIDATE_EMAIL))
    {
        $errorEmail="\n Email No valido";
    }
    if (isset ($errorEmail))
    return $errorEmail;
}
```

La vfuncion numéricoError valida que se ingresen datos numéricos

```php
function numericoError($var)
{
    //Strlen: Obtiene la longitud de un String
    $resultado=strlen($var);
    if (empty($var))
    {
        $errorNumerico = "\n Debe llenar el campo. ";
    }
    elseif($resultado<7)
    {
        $errorNumerico = "\n El numero debe tener mas de 6 digitos.";
    }
    elseif (!preg_match("/^[0-9- ]*$/", $var))
    {
        $errorNumerico = "\n Formato no valido Ej: 248 00 00 o 320-000-000";
    }
    if (isset ($errorNumerico))
    return $errorNumerico;
}
```

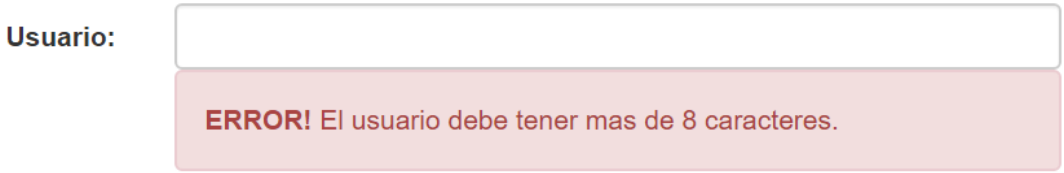
la función cedulaError valida que los datos ingresados contengan más de 8 caracteres

```php
function cedulaError($var)
{
    //Strlen: Obtiene la longitud de un String
    $resultado=strlen($var);
    if (empty($var))
    {
        $errorCedula = "\n Debe llenar el campo. ";
    }
    elseif($resultado<8)
    {
        $errorCedula = "\n La Cedula debe tener mas de 8 caracteres.";
    }
    elseif (!preg_match("/^[0-9]*$/", $var))
    {
        $errorCedula = "\n Formato no valido Ej: 111111111 o 11.111.111";
    }
    if (isset ($errorCedula))
    return $errorCedula;
}
```

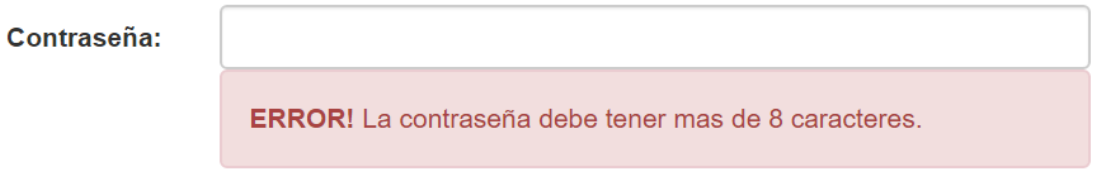
la función cargoError valida que se llene el campo y que los caracteres ingresados sean de tipo Sting (letras).

```php
function cargoError($var)
{
    if (empty($var))
    {
        $errorCargo = "\n Debe llenar el campo.";
    }
    elseif (!preg_match("/^[a-zA-Z ]*$/",$var))
    {
        $errorCargo = "\n Sólo se permiten letras y espacios en blanco.";
    }

    if (isset ($errorCargo))
    return $errorCargo;
}
```

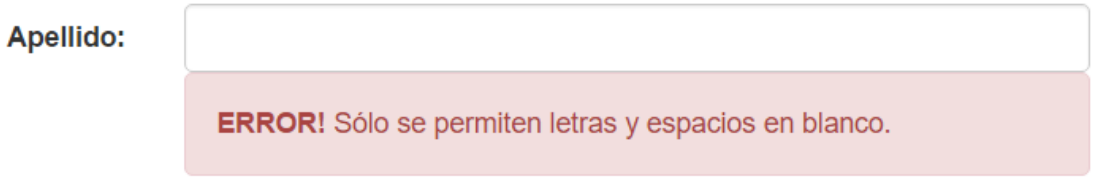
Cuando se ingresa un usuario con menos de 8 caracteres el sistema muestra por pantalla un mensaje de ERROR! Que indica que se deben digitar un usuario con mas de 8 caracteres.



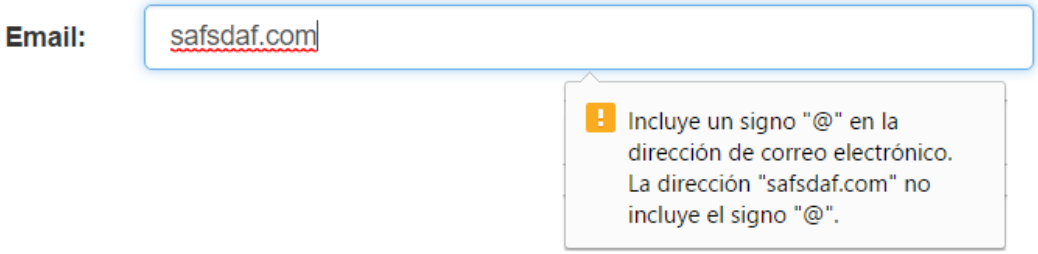
Si se ingresan menos de 8 caracteres en el campo contraseña el sistema muestra por pantalla un mensaje de ERROR! Indicandp que la contraseña debe ser de mas de 8 caracteres.



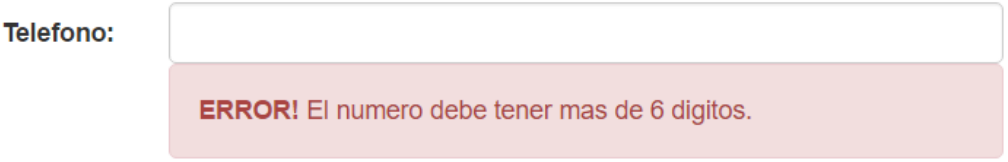
El sistema verifica que en el campo apellido solo se ingresen letras si se hace lo contrario muestra el mensaje de error y le advierte al usuario que debe digitar solo letras y espacios en blanco



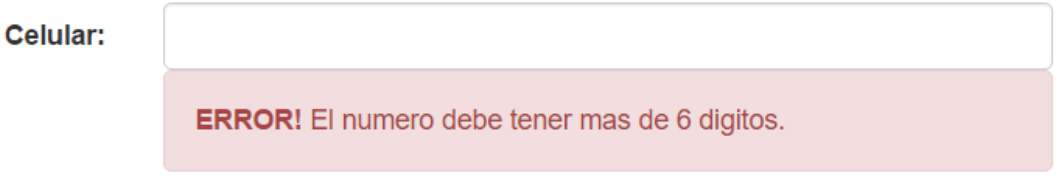
la funcion FILTER_VALIDATE_EMAIL por defecto nos muestra por ventana emergente el mensaje de validación



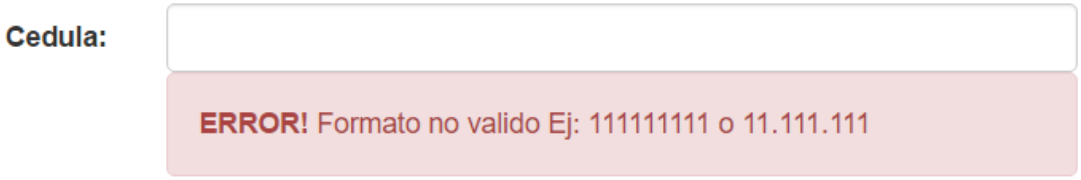
Si el sistema detecta que no se digitaron más de 6 dígitos ene l campo teléfono muestra el mensaje de error al usuario para que digite un número que contenga 6 dígitos



Si el sistema detecta que no se digitaron más de 6 dígitos ene l campo teléfono muestra el mensaje de error al usuario para que digite un número que contenga 6 dígitos



El sistema valida el formato en el cual se ingresan los datos a campo cedula si es incorrecto muestra el formato en el cual se deben ingresar los campos



Errores para registrar un producto

Se validan los campos y si se a generado algún error el sistema muestra por mensaje lo que esta ocurriendo y como llenar debidamente los campos para seguir con la ejecución

```
            if(strtolower($existe['codigob']) == strtolower($_POST['codigo']))
            {
               $errorCodigo="\n El CODIGO ya existe. ";
            }
            else
            {
               $codigo=dataForm($_POST['codigo']);
            }
          }
          else
            $errorCodigo="\n Solo se permiten letras y numeros.";
        }
        //nombre (empty=vacio)
```

REGISTRAR PRODUCTO

Codigo: a

Nombre: sa

Marca: as

Precio:

ERROR! Precio no valido.

Iva:

ERROR! Iva no valido.

Imagen: Seleccionar archivo Ningún archivo seleccionado

Caracteristicas

Categoria: 1

Guardar Cancelar

FORMULARIOS

VER PERFIL

En el siguiente código se llama todos los campos que se encuentran en la base de datos para imprimirlos en un formulario

```
                                    <input type="nombre" class="form-control" name="nombre" size="80" value
='<?php echo $row['NOMBRE']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3">Apellido:</label>
                                <div class="col-lg-6">
                                    <input type="apellido" class="form-control" name="apellido" size="80"
value='<?php echo $row['APELLIDO']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="cedula">Cedula:</label>
                                <div class="col-lg-6">
                                    <input type="cedula" class="form-control" id="cedula" size="80" value='
<?php echo $row['CEDULA_ADMI']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="cargo">Cargo:</label>
                                <div class="col-lg-6">
                                    <input type="cargo" class="form-control" id="cargo" value='<?php echo
$row['CARGO']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="genero">Genero:</label>
                                <div class="col-lg-6">
                                    <select class="form-control" name="genero">
                                        <option value="<?php echo $row['ID_GENERO_ADMI'] ?>"><?php echo
$row['ID_GENERO_ADMI'] ?></option>
                                    </select>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="direccion">Direccion:</label>
                                <div class="col-lg-6">
                                    <input type="direccion" class="form-control" id="direccion" value='
```

```
                                    <input type="direccion" class="form-control" id="direccion" value='
<?php echo $row['DIRECCION']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="Email">Email:</label>
                                <div class="col-lg-6">
                                    <input type="Email" class="form-control" id="Email" value= <?php echo
$row['EMAIL']; ?> >
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="Telefono">Telefono:</label>
                                <div class="col-lg-6">
                                    <input type="Telefono" class="form-control" id="Telefono" value= <?php
echo $row['TELEFONO']; ?> >
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="Celular">Celular:</label>
                                <div class="col-lg-6">
                                    <input type="Celular" class="form-control" id="Celular" size="80" value
= <?php echo $row['CELULAR']; ?> >
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="Usuario">Usuario:</label>
                                <div class="col-lg-6">
                                    <input type="Usuario" class="form-control" id="Usuario" size="80" value
= <?php echo $row['USUARIO']; ?> >
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3" for="password">Contraseña:</label>
                                <div class="col-lg-6">
                                    <input type="password" class="form-control" id="password" size="80"
value= <?php echo $row['CONTRASENA']; ?> >
                                </div>
                            </div>
```

```
echo $row['ID_ESTADO_PERSONA']; ?></option>
                                        </select>
                                    </div>
                                </div>
                            </fieldset>
                        </form>
                    </div>
                <?php } // fin foreach
                    } //fin else?>
```

Después de ejecutar la consulta en el sistema se mostrara un formulario que contenga los campos con la información del usuario con el cual está ingresando

PERFIL

Nombre:	Maria Paula
Apellido:	Castro Tellez
Cedula:	23423236233
Cargo:	ADMINISTRADORA
Genero:	F
Direccion:	CLL14b #119-A20 SOLSTICIO
Email:	mpcastro@gmail.com
Telefono:	7650069
Celular:	3143293428
Usuario:	distriandina
Contraseña:	••••
Estado:	1

MODIFICAR PERFIL

El sistema verifica el usuario que ingresa al sistema

```
<section>
  <?PHP menu_administrador();
     if(isset($_SESSION['usuario']))
      {
        $usuario=$_SESSION['usuario'];
        }
            $administrador=new Administrador();
            $verf=$administrador->validarAdministrador($usuario);
            if($verf==false)
            {
                header('location:turol.php');
            }
        else
        {
```

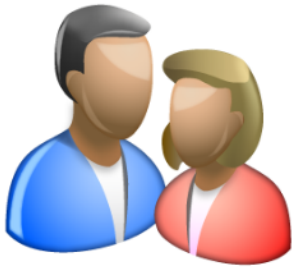
El sistema muestra por pantalla los datos del usuario ingresado al sistema

```
          <div class="col-lg-10">
        <form class="form-horizontal" autocomplete="OFF" enctype="multipart/form-data" method="POST" action="
<?php echo htmlentities($_SERVER['PHP_SELF']); ?>">

            <h3> <center> <b> MODIFICAR PERFIL </b></center></h3><br>
               <div class="form-group">
              <label class="control-label col-lg-3">Usuario:</label>
                <div class="col-lg-6">
                  <input type="Usuario" class="form-control" name="usuario" size="80" value='<?php echo $row[
'USUARIO'];?>' readonly>

                </div>
            </div>

            <div class="form-group">
              <label class="control-label col-lg-3" for="password">Contraseña:</label>
                <div class="col-lg-6">
                  <input type="password"  readonly class="form-control" name="contrasena" size="80" value='
<?php echo $row['CONTRASENA']; ?>'>
                </div>
            </div>

            <div class="form-group">
              <label class="control-label col-lg-3" for="nombre">Nombre:</label>
                <div class="col-lg-6">
                  <input type="nombre" class="form-control" name="nombre" size="80" value='<?php echo $row['
NOMBRE'];?>'>
                    <?php if (!empty($errorNombre)) {  echo modalError($errorNombre);  } ?>
                </div>
            </div>

            <div class="form-group">
              <label class="control-label col-lg-3" for="apellido">Apellido:</label>
                <div class="col-lg-6">
                  <input type="apellido" class="form-control" name="apellido" size="80" value='<?php echo
$row['APELLIDO']; ?>'>
                  <?php if (!empty($errorApellido)) {  echo modalError($errorApellido);  } ?>
                </div>
            </div>

            <div class="form-group">
              <label class="control-label col-lg-3" for="cedula">Cedula:</label>
```

El sistema modifica los campos que se cambiaron en el formulario y almacena en la base de datos cambiando los datos solicitados

```
            {
                $apellido=$_POST['apellido'];
            }
            $errorDireccion=direccionError($_POST['direccion']);
            if (!isset ($errorDirecion))
            {
                $direccion=$_POST['direccion'];
            }
            $errorEmail=emailError($_POST['email']);
            if(!isset ($errorEmail))
            {
                $email=$_POST['email'];
            }
            $errorTelefono=numericoError($_POST['telefono']);
            if(!isset ($errorTelefono))
            {
                $telefono=$_POST['telefono'];
            }
            $errorCelular=numericoError($_POST['celular']);
            if(!isset ($errorCelular))
            {
                $celular=$_POST['celular'];
            }
            $errorCedula=cedulaError($_POST['cedula']);
            if(!isset($errorCedula))
            {
                $cedula=$_POST['cedula'];
            }
            $errorCargo=cargoError($_POST['cargo']);
            if(!isset($errorCargo))
            {
                $cargo=$_POST['cargo'];
            }

            if((isset($nombre)) && (isset($apellido)) && (isset($direccion)) && (isset($email)) && (isset(
$telefono)) && (isset($celular)) && (isset($cedula)) && (isset($cargo)) && (isset($genero)))
            {
                $administrador=new Administrador;
                $msg= $administrador->modificarAdministrador($_POST['usuario'],$_POST['contrasena'],$_POST[
'nombre'],$_POST['apellido'],$_POST['direccion'],$_POST['email'],$_POST['telefono'],$_POST['celular'],$_POST[
'cedula'],$_POST['cargo'],$_POST['genero']);
            }
        } ?>
```

El sistema redireccióna a ver perfil y muestra el nuevo perfil con los datos cambiados



PERFIL

Nombre:	Maria Paula
Apellido:	Castro Tellez
Cedula:	23423236233
Cargo:	administracion
Genero:	F
Direccion:	CLL14b #119-A20 SOLSTICIO
Email:	mpcastro@gmail.com
Telefono:	7650069
Celular:	3143293428
Usuario:	distriandina
Contraseña:	••••
Estado:	1

CAMBIAR CONTRASEÑA

```
        <center><p> <b> <h3>CAMBIAR CONTRASEÑA </h3> </b></p></center><br>
        <div class="col-lg-10">
            <form class="form-horizontal" autocomplete="OFF" enctype="multipart/form-data" method="POST" action="
<?php echo htmlentities($_SERVER['PHP_SELF']); ?>">

                <div class="form-group">
                    <label class="control-label col-lg-3">Contraseña actual:</label>
                    <div class="col-lg-4">
                        <input type="password" class="form-control" name="actual" size="80">
                        <?php if (!empty($errorActual)) {  echo modalError($errorActual);  } ?>
                    </div>
                </div>
                <div class="form-group">
                    <label class="control-label col-lg-3">Contraseña nueva:</label>
                    <div class="col-lg-4">
                        <input type="password" class="form-control" name="nueva" size="80" required>
                        <?php if (!empty($errorNueva)) {  echo modalError($errorNueva);  } ?>
                    </div>
                </div>
                <div class="form-group">
                    <label class="control-label col-lg-3">Repetir contraseña:</label>
                    <div class="col-lg-4">
                        <input type="password" class="form-control" name="nuevaR" size="80" required>
                        <?php if (!empty($errorNuevaR)) {  echo modalError($errorNuevaR);  } ?>
                    </div>
                </div>
                <div class="form-group">
                    <div class="col-lg-offset-4 col-lg-5">
                        <button type="submit" name="submit" class="btn btn-default">Guardar</button>

                    </div>
                </div>
                <?php }  ?>
```

REGISTRO DE UN VENDEDOR

El formulario se crea con los campos

Usuario Contraseña Nombre Apellido Dirección Email Teléfono Celular Cedula Genero Zona

Se leen los campos por el métodos $_POST y se almacenan en variables para usarlas en el código PHP

```php
$errorUsuario=usuarioError($_POST['usuario']);
if (!isset ($errorUsuario))
{
    $Usuario=$_POST['usuario'];
}
$errorContrasena=contrasenaError($_POST['contrasena']);
if (!isset ($errorContrasena))
{
    $contrasena=$_POST['contrasena'];
}
$errorNombre=nombreError($_POST['nombre']);
if (!isset ($errorNombre))
{
    $nombre=$_POST['nombre'];
}
$errorApellido=apellidoError($_POST['apellido']);
if (!isset ($errorApellido))
{
    $apellido=$_POST['apellido'];
}
$errorDireccion=direccionError($_POST['direccion']);
if (!isset ($errorDirecion))
{
    $direccion=$_POST['direccion'];
}
$errorEmail=emailError($_POST['email']);
if(!isset ($errorEmail))
{
    $email=$_POST['email'];
}
$errorTelefono=numericoError($_POST['telefono']);
if(!isset ($errorTelefono))
{
    $telefono=$_POST['telefono'];
}
$errorCelular=numericoError($_POST['celular']);
if(!isset ($errorCelular))
{
    $celular=$_POST['celular'];
}
```

El isset verifica que los daros ingresados sean correctos y de esta manera se crea un objeto de la clase vendedor para ingresar los datos y así guardarlos en la base de datos

```
        if ((isset($usuario)) && (isset($contrasena)) && (isset($nombre)) &&
(isset($apellido)) && (isset($email)) && (isset($direccion)) && (isset($telefono)) &&
(isset($celular)) && (isset($cedula)))

        {

            $vendedor=new Vendedor;

            $msg=$vendedor-
>registrarVendedor($_POST['usuario'],$_POST['contrasena'],$_POST['nombre'],$_POS
```

+

MODIFICAR UN VENDEDOR

El sistema muestra un formulario de todos los usuarios que se encuentran registrados

```
                <table width="700" border="2" cellspacing="1" style="font-family: verdana; font-size: 12px;
 margin: 10px 20px;    width: 900px; text-align: left;  border-collapse: collapse" >
                     <tr aling="center" style="background: #d0dafd; color: #339;">
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; " > <B>USUARIO </B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>NOMBRE</B> </td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>APELLIDO</B> </td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>CEDULA</B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>GENERO </B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>DIRC </B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>EMAIL </B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>TEL </B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>CEL</B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>ZONA</B></td>
                         <td style=" padding: 8px; background: #e8edff; border-bottom: 1px solid #fff;
color: #669;    border-top: 1px solid transparent; "> <B>ESTADO</B></td>
```

El sistema muestra un formulario para buscar el usuario pidiendo dijitar el formulario

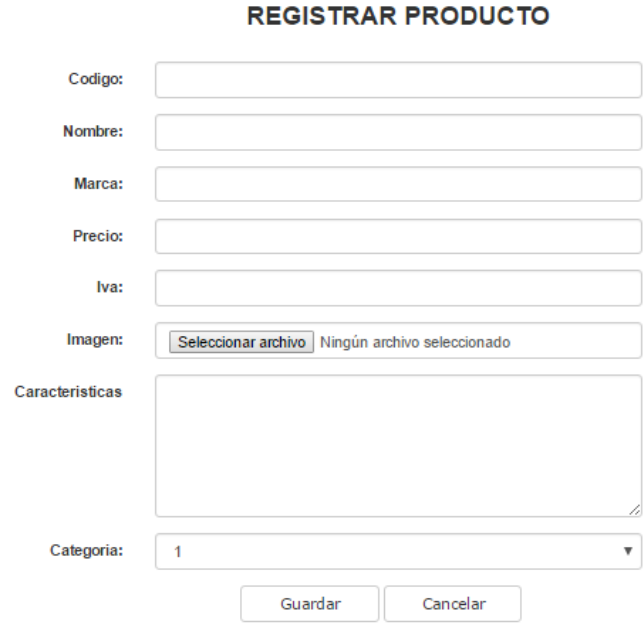
```
<center><p> <b> <h3>MODIFICAR VENDEDOR</h3> </b></p></center><br>
<div class="col-lg-10">
    <form method="post" action="modificarVendedor.php" autocomplete="OFF">
        <div class="form-group">
            <label class="control-label col-lg-2"> Ingrese Usuario </label>
            <div class="col-lg-5">
                <input type="text" class="form-control" name="Usuario"><br>
            </div>
            <button type="submit" name="submit" class="btn btn-default">Buscar</button>
        </div>
    </form>
```

```
                    <br><form class="form-horizontal" autocomplete="OFF" enctype="multipart/form-da
"POST" action="<?php echo htmlentities($_SERVER['PHP_SELF']); ?>">
                                <div class="form-group">
                                    <label class="control-label col-lg-3">Usuario:</label>
                                    <div class="col-lg-6">
                                       <input type="text" class="form-control" name="usuario" size="80"
value='<?php echo $row['USUARIO'];?>' readonly>
                                    </div>
                                </div>

                                <div class="form-group">
                                    <label class="control-label col-lg-3">CONTRASEÑA:</label>
                                    <div class="col-lg-6">
                                       <input type="password" class="form-control" name="contrasena" s
required value='<?php echo $row['CONTRASENA'];?>' readonly>
                                    </div>
                                </div>

                                <div class="form-group">
                                <label class="control-label col-lg-3">CEDULA:</label>
                                <div class="col-lg-6">
                                    <input type="cedula" class="form-control" name="cedula" size="80" r
value='<?php echo $row['CEDULA_VENDEDOR']; ?>'>
                                </div>
                            </div>
                                <div class="form-group">
                                    <label class="control-label col-lg-3">NOMBRE:</label>
                                    <div class="col-lg-6">
                                       <input type="text" class="form-control" name="nombre" size="80"
value='<?php echo $row['NOMBRE'];?>'>
                                    </div>
                                </div>

                                <div class="form-group">
                                    <label class="control-label col-lg-3">APELLIDO:</label>
                                    <div class="col-lg-6">
                                       <input type="text" class="form-control" name="apellido" size="80
value='<?php echo $row['APELLIDO'];?>'>
                                    </div>
                                </div>
                                <input type="text" value="direccion" name="direccion" rows="6" class="form-control" co
"100" value='<?php echo $row['DIRECCION']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3">EMAIL:</label>
                                <div class="col-lg-6">
                                    <input type='text' class="form-control" name='email' value='<?php echo $
['EMAIL']; ?>'>
                                </div>
                            </div>

                            <div class="form-group">
                                <label class="control-label col-lg-3">TELEFONO:</label>
                                <div class="col-lg-6">
                                    <input type='text' class="form-control" name='telefono' value='<?php ech
$row['TELEFONO']; ?>'>
                                </div>
                            </div>
                            <div class="form-group">
                                <label class="control-label col-lg-3">CELULAR:</label>
                                <div class="col-lg-6">
                                    <input type='text' class="form-control" name='celular' value='<?php echo
$row['CELULAR']; ?>'>
                                </div>
                            </div>

                            <?php
                            $db1 = new Conexion();
                            $sql1 = $db1->query("SELECT * FROM genero"); ?>
                            <div class="form-group">
                                <label class="control-label col-lg-3" for="genero">GENERO:</label>
                                <div class="col-lg-6">

                                    <select class="form-control" name="genero" required="">
                                        <?php
                                            while ($resul=$db1->recorrer($sql1))
                                            {  ?>
                                                <option selected></option>
```

FOMULARIO PARA REGISTRAR UN PRODUCTO

```
189
190         <center><p> <b> <h3>REGISTRAR PRODUCTO</h3> </b></p></center><br>
191         <div class="col-lg-10">
192             <form class="form-horizontal" autocomplete="OFF" enctype="multipart/form-data" method="POST" action
    ="<?php echo htmlentities($_SERVER['PHP_SELF']); ?>">
193                 <div class="form-group">
194                     <label class="control-label col-lg-3" for="Codigo">Codigo:</label>
195                     <div class="col-lg-6">
196                         <input type="text" class="form-control" name="codigo" size="80" required value='<?php
    echo htmlentities($codigo) ?>'>
197                         <?php if (!empty($errorCodigo)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorCodigo</a>.</div>";  } ?>
198                     </div>
199                 </div>
200
201                 <div class="form-group">
202                     <label class="control-label col-lg-3" for="nombre">Nombre:</label>
203                     <div class="col-lg-6">
204                         <input type="text" class="form-control" name="nombre" size="80" required value='<?php
    echo htmlentities($nombre) ?>'>
205                         <?php if (!empty($errorNombre)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorNombre</a>.</div>";  } ?>
206                     </div>
207                 </div>
208
209                 <div class="form-group">
210                     <label class="control-label col-lg-3" for="marca">Marca:</label>
211                     <div class="col-lg-6">
212                         <input type="text" class="form-control" name="marca"  size="80" required value='<?php
    echo htmlentities($marca) ?>'>
213                         <?php if (!empty($errorMarca)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorMarca</a>.</div>";  }  ?>
214                     </div>
215                 </div>
216
217                 <div class="form-group">
218                     <label class="control-label col-lg-3" for="precio">Precio:</label>
219                     <div class="col-lg-6">
220                         <input type="text" class="form-control" name="precio" required value='<?php echo
    htmlentities($precio) ?>'>
221                         <?php if (!empty($errorPrecio)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorPrecio</a>. </div>"; } ?>
```

```
208
209                 <div class="form-group">
210                     <label class="control-label col-lg-3" for="marca">Marca:</label>
211                     <div class="col-lg-6">
212                         <input type="text" class="form-control" name="marca"  size="80" required value='<?php
    echo htmlentities($marca) ?>'>
213                         <?php if (!empty($errorMarca)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorMarca</a>.</div>";  }  ?>
214                     </div>
215                 </div>
216
217                 <div class="form-group">
218                     <label class="control-label col-lg-3" for="precio">Precio:</label>
219                     <div class="col-lg-6">
220                         <input type="text" class="form-control" name="precio" required value='<?php echo
    htmlentities($precio) ?>'>
221                         <?php if (!empty($errorPrecio)) {  echo "<div class='alert alert-danger'><strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorPrecio</a>. </div>"; } ?>
222                     </div>
223                 </div>
224
225                 <div class="form-group">
226                     <label class="control-label col-lg-3">Iva:</label>
227                     <div class="col-lg-6">
228                         <input type="text" class="form-control" name="iva" required value='<?php echo
    htmlentities($iva) ?>'>
229                         <?php if (!empty($errorIva)) {  echo "<div class='alert alert-danger'> <strong>ERROR!
    </strong> <a href=# class='alert-link'>$errorIva</a>. </div>"; } ?>
230                     </div>
231                 </div>
232
233                 <div class="form-group">
234                     <label class="control-label col-lg-3" >Imagen:</label>
235                     <div class="col-lg-6">
236                         <input type="file" class="form-control" name="imagen" required>
237                     </div>
238                 </div>
239
240                 <div class="form-group">
241                     <label class="control-label col-lg-3" for="Caracteristicas">Caracteristicas</label>
242                     <div class="col-lg-6">
243                         <textarea name='caracteristicas' required rows="6" class="form-control" cols="100"
    value='<?php echo htmlentities($caracteristicas) ?>'> </textarea>
```



BOTONES

GUARDAR = se almacena en la Base de datos el vendedor registrado con los campos

Usuario

Contraseña

Nombre

Apellido

Dirección

Email

Teléfono

Celular

Cedula

Genero

Zona

CANCELAR =se da por terminada la operación sin ejecutar ningún cambio

9 HERRAMIENTAS PARA IMPLEMENTACION DEL APLICATIVO

9.1 PHP

Es un Lenguaje de Programación para trabajar páginas WEB ofreciendo la ventaja de mezclarse con HTML. Las ejecuciones son realizadas en el Servidor y el cliente es el encargado de recibir los resultados de la ejecución. Si el cliente realiza una petición, se ejecuta el intérprete de PHP y se genera el contenido de manera dinámica. Permite conexión con varios tipos de Bases de Datos como: MySql, Oracle, Postgress, SQL Server, etc. permitiendo aplicaciones robustas sobre la WEB. Este lenguaje de programación puede ser ejecutado en la gran mayoría de sistemas operacionales y puede interactuar con Servidores WEB populares

9.2 MYSQL

Es un manejador de Bases de Datos, el cual permite múltiples hilos y múltiples usuarios, fue desarrollado como software libre. Aunque se puede usar sobre varias plataformas es muy utilizado sobre LINUX. Es libre para uso en Servidores WEB. Ofrece ventajas tales como fácil adaptación a diferentes entornos de desarrollo, Interacción con Lenguajes de Programación como PHP, Java Script y fácil Integración con distintos sistemas operativos

9.3 APACHE

Es un Servidor WEB desarrollado por el grupo Apache. Su código fuente se puede distribuir y utilizar de forma libre. Está disponible para diferentes plataformas de Sistemas Operativos entre otros Windows, Linux, Mac y NetWare. Ofrece ventajas tales como independencia de plataforma, haciendo posible el cambio de plataforma en cualquier momento; creación de contenidos dinámicos, permitiendo crear sitios mediante lenguajes PHP. Además de ser libre su soporte técnico es accesible ya que existe una comunidad que está disponible en foros, canales IRC y servidores de noticias, donde hay gran cantidad de usuarios disponibles para cuando surge algún problema.

10. BASE DE DATOS

```
CREATE TABLE PERSONA
(
        USUARIO VARCHAR(25) NOT NULL,
        CONTRASENA VARCHAR(25) NOT NULL,
        NOMBRE VARCHAR(25) NOT NULL,
        APELLIDO VARCHAR(25) NOT NULL,
        DIRECCION VARCHAR(50) NOT NULL,
        EMAIL VARCHAR(50) NOT NULL,
        TELEFONO NUMERIC(19,0) NOT NULL,
        CELULAR NUMERIC(19,0) NOT NULL,
        ID_ESTADO_PERSONA VARCHAR(5) NOT NULL
);
```

```
CREATE TABLE ADMINISTRADOR
(
        USUARIO VARCHAR(25) NOT NULL,
        CEDULA_ADMI NUMERIC(19,0) NOT NULL,
        CARGO VARCHAR(25) NOT NULL,
        ID_GENERO_ADMI VARCHAR(5) NOT NULL
);

CREATE TABLE VENDEDOR
(
        USUARIO VARCHAR(25) NOT NULL,
        CEDULA_VENDEDOR NUMERIC(19,0) NOT NULL,
        ZONA_VENDEDOR VARCHAR(5) NOT NULL,
        ID_GENERO_VENDEDOR VARCHAR(5) NOT NULL
);

CREATE TABLE GENERO
(
        ID_GENERO VARCHAR(5) NOT NULL,
        DESCRIPCION_GENERO VARCHAR(25) NOT NULL
);

CREATE TABLE ZONA
(
        ID_ZONA VARCHAR(5) NOT NULL,
        DESCRIPCION_ZONA VARCHAR(25) NOT NULL
);

CREATE TABLE CLIENTE
(
        USUARIO VARCHAR(25) NOT NULL,
```

```
        NIT  NUMERIC(19,0) NOT NULL,
        ADMINISTRADOR NUMERIC (19,0) NOT NULL
);

CREATE TABLE ADMINISTRADOR_CLIENTE
(
        CEDULA_CLIENTE NUMERIC(19,0) NOT NULL,
        NOMBRE_CLIENTE VARCHAR(25) NOT NULL
);

CREATE TABLE ESTADO
(
        ID_ESTADO VARCHAR(5) NOT NULL,
        DESCRIPCION VARCHAR(25) NOT NULL
);

CREATE TABLE PEDIDO
(
        N_PEDIDO VARCHAR(10) NOT NULL,
        CLIENTE VARCHAR(25) NOT NULL
);

CREATE TABLE PEDIDO_PRODUCTO
(
        NUMERO_PEDIDO VARCHAR(10) NOT NULL,
        COD_PRODUCTO VARCHAR(10) NOT NULL,
        CANT_PRODUCTO DECIMAL(19,0) NOT NULL,
        FECHA DATE NOT NULL
);

CREATE TABLE FACTURA_VENTA
(
```

```
        FECHA DATE NOT NULL,
        N_PEDIDO VARCHAR(10) NOT NULL,
        ID_ESTADO_FACT BIT NOT NULL,
        ID_PAGO_FACT NUMERIC(19,0) NOT NULL
);

CREATE TABLE ESTADO_FACTURA
(
        ID_ESTADO_FAC BIT NOT NULL,
        DESCRIPCION_ESTADO VARCHAR(25) NOT NULL
);

CREATE TABLE FORMA_DE_PAGO
(
        ID_PAGO NUMERIC(19,0) NOT NULL,
        DESCRIPCION_PAGO VARCHAR(25) NOT NULL
);

CREATE TABLE PRODUCTO
(
        CODIGO_PRODUCTO VARCHAR(10) NOT NULL,
        NOMBRE_PRODUCTO VARCHAR(25) NOT NULL,
        MARCA VARCHAR(25) NOT NULL,
        PRECIO REAL NOT NULL,
        CARACTERISTICAS VARCHAR(255) NOT NULL,
        IVA REAL NOT NULL,
        ID_CATEGORIA_PROD VARCHAR(10) NOT NULL,
        ID_ESTADO_PROD VARCHAR(5) NOT NULL
);

CREATE TABLE CATEGORIA
(
```

```
        ID_CATEGORIA VARCHAR(10) NOT NULL,
        DESCRIPCION_CATEGORIA VARCHAR(25) NOT NULL,
        ID_ESTADO_CAT VARCHAR(5) NOT NULL
);

CREATE TABLE PRODUCTO_FACTURA_COMPRA
(
        CODIGO_PRODUCTO_FAC VARCHAR(10) NOT NULL,
        N_FACTURA NUMERIC(19,0) NOT NULL,
        CANT_P NUMERIC(19,0) NOT NULL,
        PNETO REAL NOT NULL,
        IVA_COMPRA REAL NOT NULL,
        TOTAL_COMPRA REAL NOT NULL
);

CREATE TABLE FACTURA_COMPRA
(
        N_FACTURA NUMERIC(19,0) NOT NULL,
        FECHA DATE NOT NULL,
        FAC_NIT_PROVEEDOR NUMERIC(19,0) NOT NULL
);

CREATE TABLE PROVEEDOR
(
        NIT_PROVEEDOR NUMERIC(19,0) NOT NULL,
        NOMBRE_PROVEEDOR VARCHAR(25) NOT NULL,
        TEL_PROVEEDOR NUMERIC(19,0) NOT NULL,
        CEL_PROVEEDOR NUMERIC(19,0) NOT NULL,
        EMAIL_PROVEEDOR VARCHAR(25) NOT NULL,
        DIRECCION_PROVEEDOR VARCHAR(25) NOT NULL
);
```

```
ALTER TABLE `persona` ADD PRIMARY KEY(`USUARIO`);

ALTER TABLE `administrador` ADD PRIMARY KEY( `USUARIO`);

ALTER TABLE `administrador_cliente` ADD PRIMARY KEY(
`CEDULA_CLIENTE`);

ALTER TABLE `categoria` ADD PRIMARY KEY( `ID_CATEGORIA`);

ALTER TABLE `cliente` ADD PRIMARY KEY( `USUARIO`);

ALTER TABLE `estado` ADD PRIMARY KEY( `ID_ESTADO`);

ALTER TABLE `estado_factura` ADD PRIMARY KEY( `ID_ESTADO_FAC`);

ALTER TABLE `factura_compra` ADD PRIMARY KEY( `N_FACTURA`);

ALTER TABLE `factura_venta` ADD PRIMARY KEY( `N_PEDIDO`);

ALTER TABLE `forma_de_pago` ADD PRIMARY KEY( `ID_PAGO`);

ALTER TABLE `genero` ADD PRIMARY KEY( `ID_GENERO`);

ALTER TABLE `pedido` ADD PRIMARY KEY( `N_PEDIDO`);

ALTER TABLE `pedido_producto` ADD PRIMARY KEY( `NUMERO_PEDIDO`,
`COD_PRODUCTO`);

ALTER TABLE `producto` ADD PRIMARY KEY( `CODIGO_PRODUCTO`);

ALTER TABLE `producto_factura_compra` ADD PRIMARY KEY(
`CODIGO_PRODUCTO_FAC`, `N_FACTURA`);

ALTER TABLE `proveedor` ADD PRIMARY KEY( `NIT_PROVEEDOR`);

ALTER TABLE `vendedor` ADD PRIMARY KEY( `USUARIO`);

ALTER TABLE `zona` ADD PRIMARY KEY( `ID_ZONA`);




-- LLAVES FORANEAS---


ALTER TABLE `administrador` ADD FOREIGN KEY (`USUARIO`) REFERENCES
`persona`(`USUARIO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `administrador` ADD FOREIGN KEY (`ID_GENERO_ADMI`)
REFERENCES `genero`(`ID_GENERO`) ON DELETE RESTRICT ON UPDATE
RESTRICT;

ALTER TABLE `cliente` ADD FOREIGN KEY (`USUARIO`) REFERENCES
`persona`(`USUARIO`) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

```
ALTER TABLE `cliente` ADD FOREIGN KEY (`ADMINISTRADOR`) REFERENCES `administrador_cliente`(`CEDULA_CLIENTE`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `vendedor` ADD FOREIGN KEY (`USUARIO`) REFERENCES `persona`(`USUARIO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `vendedor` ADD FOREIGN KEY (`ID_GENERO_VENDEDOR`) REFERENCES `genero`(`ID_GENERO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `vendedor` ADD FOREIGN KEY (`ZONA_VENDEDOR`) REFERENCES `zona`(`ID_ZONA`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `persona` ADD FOREIGN KEY (`ID_ESTADO_PERSONA`) REFERENCES `estado`(`ID_ESTADO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `pedido` ADD FOREIGN KEY (`CLIENTE`) REFERENCES `cliente`(`USUARIO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `factura_venta` ADD FOREIGN KEY (`N_PEDIDO`) REFERENCES `pedido`(`N_PEDIDO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `factura_venta` ADD FOREIGN KEY (`id_estado_fact`) REFERENCES `estado_factura`(`id_estado_fac`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `factura_venta` ADD FOREIGN KEY (`id_pago_fact`) REFERENCES `forma_de_pago`(`id_pago`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `pedido_producto` ADD FOREIGN KEY (`NUMERO_PEDIDO`) REFERENCES `pedido`(`N_PEDIDO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `pedido_producto` ADD FOREIGN KEY (`COD_PRODUCTO`) REFERENCES `producto`(`CODIGO_PRODUCTO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `producto` ADD FOREIGN KEY (`ID_CATEGORIA_PROD`) REFERENCES `categoria`(`ID_CATEGORIA`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `producto` ADD FOREIGN KEY (`ID_ESTADO_PROD`) REFERENCES `estado`(`ID_ESTADO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `categoria` ADD FOREIGN KEY (`ID_ESTADO_CAT`) REFERENCES `estado`(`ID_ESTADO`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `producto_factura_compra` ADD FOREIGN KEY (`CODIGO_PRODUCTO_FAC`) REFERENCES `producto`(`CODIGO_PRODUCTO`) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

```
ALTER TABLE `producto_factura_compra` ADD FOREIGN KEY (`N_FACTURA`) REFERENCES `factura_compra`(`N_FACTURA`) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE `factura_compra` ADD FOREIGN KEY (`FAC_NIT_PROVEEDOR`) REFERENCES `proveedor`(`NIT_PROVEEDOR`) ON DELETE RESTRICT ON UPDATE RESTRICT;



INSERT INTO  `distriandina`.`genero` (
`ID_GENERO` ,
`DESCRIPCION_GENERO`
)
VALUES (
'M',  'MASCULINO'
), (
'F',  'FEMENINO'
);


INSERT INTO  `distriandina`.`estado` (
`ID_ESTADO` ,
`DESCRIPCION`
)
VALUES (
'
```

Creación de las tablas.

Insertar llaves primarias a las tablas.

Insertar llaves foráneas a las tablas.

Insertar selección múltiple a campos de las tablas.