

The background of the slide is a solid light green color. On the left side, there is a faint, semi-transparent watermark of the Android robot, which is a stylized green figure with a circular head and a rectangular body.

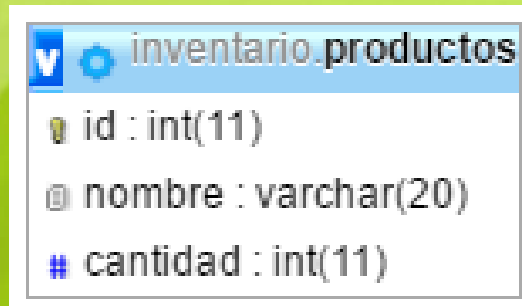
# **GESTIÓN DE PRODUCTOS CON AJAX PHP Y MYSQL**

Heri Fernando Londoño Salgado

# BASE DE DATOS.

Esta actividad permitirá insertar y mostrar la información de una base de datos, en una página con estructura SPA, utilizando HTML, CSS, JavaScript, PHP y MySQL.

Se utilizará una base de datos en MySQL, llamada inventario con una tabla llamada productos con la siguiente estructura.



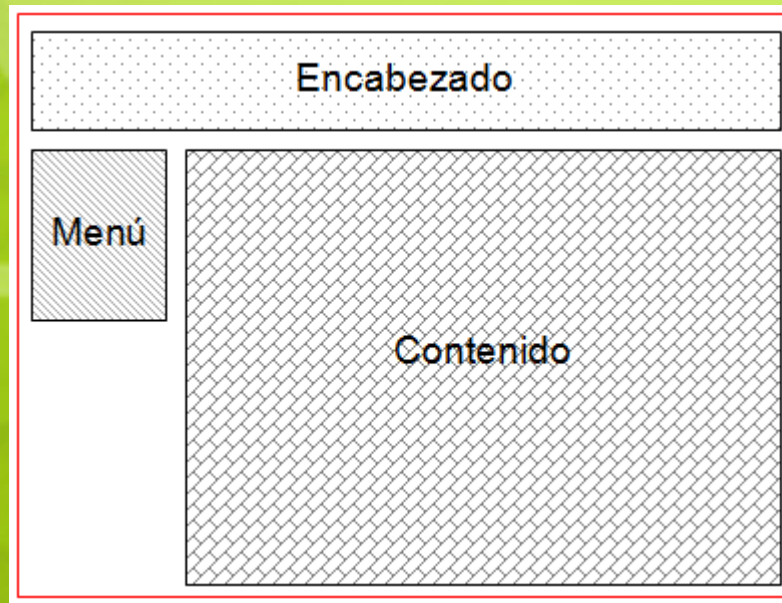
A screenshot of a database management tool showing the structure of a table named 'inventario.productos'. The table has three columns: 'id' of type 'int(11)' with a primary key icon, 'nombre' of type 'varchar(20)' with a text icon, and 'cantidad' of type 'int(11)' with a numeric icon.

inventario.productos	
🔑	id : int(11)
📄	nombre : varchar(20)
#	cantidad : int(11)

# ESTRUCTURA

La página tendrá la estructura mostrada en la imagen.

- Contenedor, delineado en rojo.
- Encabezado.
- Menú.
- Contenido.



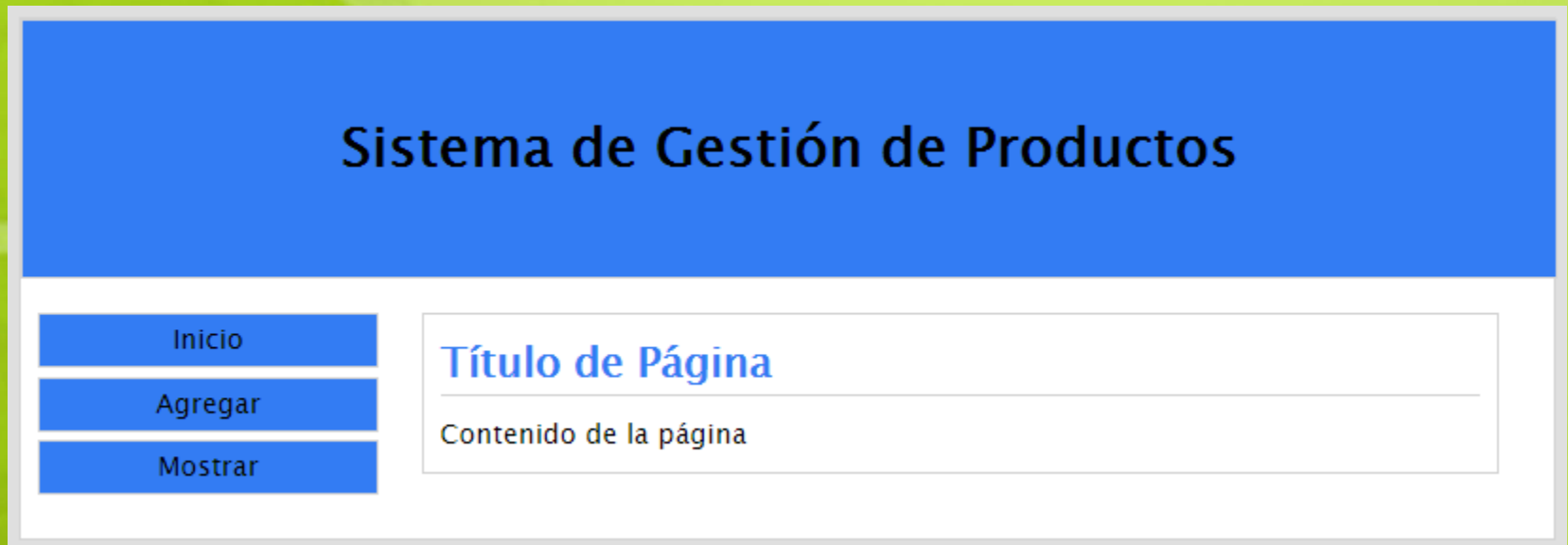
# index.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8">
5     <title>Gestión de Productos</title>
6     <link rel="stylesheet" href="estilos.css" type="text/css">
7     <script src="script.js" type="text/javascript" defer></script>
8 </head>
9 <body>
10     <div id="contenedor">
11         <div id="encabezado">
12             <h1>Sistema de Gestión de Productos</h1>
13         </div>
14         <ul id="menu">
15             <li><a href="#" onclick="cargarPlantilla('inicio.html')">Inicio</a></li>
16             <li><a href="#" onclick="cargarPlantilla('agregar.html')">Agregar</a></li>
17             <li><a href="#" onclick="cargarMostrar()">Mostrar</a></li>
18         </ul>
19         <div id="contenido">
20             <h2 id="titulo">Título de Página</h2>
21             <div id="contenido2">
22                 <p>Contenido de la página</p>
23             </div>
24         </div>
25     </div>
26 </body>
27 </html>
```

# HOJA DE ESTILOS

La página tendrá una interfaz gráfica como la que se muestra en la imagen, por lo tanto se le debe aplicar un conjunto de estilos, en un archivo llamado estilos.css. El cual será referenciado desde el archivo html en la línea 6.

Los estilos se explicaran con detalle en las siguientes presentaciones.



Heri Fernando Londoño Salgado

# HOJA DE ESTILOS

Lo primero que se hace es resetear los márgenes y rellenos de la página, por lo tanto en el selector universal (\*) se asigna 0 a esos selectores.

De acuerdo con la estructura de la página, el tamaño del contenedor principal será fijo, para notarlo se le asignara un color al fondo gris a la etiqueta body a través del selector body, en el archivo de estilos.

```
1  *{
2      margin: 0;
3      padding: 0;
4  }
5  body{
6      background-color: #dfdfff;
7  }
```

# HOJA DE ESTILOS

El contenedor principal tendrá letra tipo Lucida Sans Unicode, y segunda opción sans-serif. Tamaño 1em, color de fondo blanco, borde color gris de un pixel solido, alineación justificada, margen superior e inferior de 10px y laterales automático, el ancho será de 100%, pero su ancho máximo es de 900px, de esta forma se adaptará al tamaño de la pantalla.

```
8 #contenedor{
9     background-color: #fff;
10    border: #d3d3d3 1px solid;
11    font-family:'Lucida Sans Unicode', sans-serif;
12    font-size: 1em;
13    margin: 10px auto;
14    max-width: 900px;
15    text-align: justify;
16    width: 100%;
17 }
```

# HOJA DE ESTILOS

El encabezado tendrá color de fondo y texto.

El encabezado tendrá como estilos un ancho de 100%, altura de 150, y el color de fondo azul.

El texto en el encabezado se debe centrar y ubicarlo más abajo que su posición inicial.

```
18 #encabezado{
19     background-color: #337CF3;
20     border: #d3d3d3 1px solid;
21     height: 150px;
22     width: 100%;
23 }
24 #encabezado h1{
25     margin-top: 50px;
26     text-align: center;
27 }
```



# HOJA DE ESTILOS

Al contenido de la plantilla se le asignara un borde de un pixel, además se le establecerán márgenes externos, se cambiará el display, para poder agrupar en la misma línea el contenido y el menú, tendrá un ancho del 70%

La estructura de dicho estilo es la siguiente.

```
28 #contenido{
29     border: #d3d3d3 1px solid;
30     display: inline-block;
31     margin: 20px 10px;
32     vertical-align: top;
33     width: 70%;
34 }
```

# HOJA DE ESTILOS

El título de la página el cual se define con el id titulo, tendrá color azul, y un borde inferior de color gris y un margen.

Al contenido también se le asignará un margen.

Los estilos quedaran definidos de la siguiente manera.

```
35 #titulo{
36     margin: 10px;
37     color: #337CF3;
38     border-bottom: #d3d3d3 1px solid;
39 }
40 #contenido2{
41     margin: 10px;
42 }
```

# HOJA DE ESTILOS

A la lista con id menu, se le cambiará el display, se le eliminarán las viñetas, se le asignará un margen, se alineará en la parte superior, y tendrá un ancho del 20%.

```
43 #menu{  
44     display: inline-block;  
45     list-style-type: none;  
46     margin: 20px 10px;  
47     vertical-align: top;  
48     width: 20%;  
49 }
```

# HOJA DE ESTILOS

Los ítems lista con id menú tendrán un borde, un color de fondo, un interlineado mayor, un margen inferior y el texto estará centrado.

```
50 #menu li{  
51     border: #d3d3d3 1px solid;  
52     background-color: #337CF3;  
53     line-height: 30px;  
54     margin-bottom: 5px;  
55     text-align: center;  
56 }
```

# HOJA DE ESTILOS

Los enlaces dentro de los ítems de la lista se les modificará el display, se les eliminará el subrayado y el color del texto será negro.

```
57 #menu a{  
58     display: block;  
59     text-decoration: none;  
60     color: #000;  
61 }
```

# CLASE Conexion

Para acceder a la base de datos y realizar las tareas CRUD se creara una clase Conexión con la estructura mostrada en el diagrama de clases.

Conexion
<ul style="list-style-type: none"><li>-mySQLI: mysqli</li><li>-sql: string</li><li>-result: mysqli_result</li><li>-filasAfectadas: int</li></ul>
<ul style="list-style-type: none"><li>+abrir(): int</li><li>+cerrar()</li><li>+consulta(sql: string)</li><li>+obtenerResult(): mysqli_result</li><li>+obtenerFilasAfectadas(): int</li></ul>

# CLASE Conexion

```
<?php
class Conexion{
    private $mySQLI;
    private $result;
    private $filasAfectadas;
    private $autoid;
    public function abrir(){
        $this->mySQLI = new mysqli("localhost", "root", "", "inventario");
        if(mysqli_connect_error()){
            return 0;
        }
        else{
            return 1;
        }
    }
    public function cerrar(){
        $this->mySQLI->close();
    }
    public function consulta($sql){
        $this->result = $this->mySQLI->query($sql);
        $this->filasAfectadas = $this->mySQLI->affected_rows;
        $this->autoid = $this->mySQLI->insert_id;
    }
    public function obtenerResult(){
        return $this->result;
    }
    public function obtenerFilasAfectadas(){
        return $this->filasAfectadas;
    }
    public function obtenerAutoId(){
        return $this->autoid;
    }
}
```

Heri Fernando Londoño Salgado

# Script agregarproducto.php

```
1 <?php
2     require_once "Conexion.php";
3     $conexion = new Conexion();
4     $conexion->abrir() == 0;
5     $id = $_GET["id"];
6     $nombre = $_GET["nombre"];
7     $cantidad = $_GET["cantidad"];
8     $sql = "INSERT INTO productos VALUES ($id, '$nombre', $cantidad)";
9     $conexion->consulta($sql);
10    $conexion->cerrar();
11    $respuesta = array();
12    if($conexion->obtenerFilasAfectadas() > 0)
13    {
14        $respuesta["accion"] = 1;
15        $respuesta["mensaje"] = "Producto insertado con exito";
16    }
17    else{
18        $respuesta["accion"] = 0;
19        $respuesta["mensaje"] = "Producto no pudo ser insertado";
20    }
21    echo json_encode($respuesta);
22 ?>
```

```
{ "accion": 1, "message": "Producto insertado con exito" }
```

```
{ "accion": 0, "mensaje": "Producto no pudo ser insertado" }
```



# Script consultarproductos.php

```
1 <?php
2     require_once "Conexion.php";
3     $conexion = new Conexion;
4     $conexion->abrir();
5     $sql = "SELECT * FROM productos";
6     $conexion->consulta($sql);
7     $result = $conexion->obtenerResult();
8     $conexion->cerrar();
9     if($result->num_rows > 0){
10         $respuesta["productos"] = array();
11         while ($fila = $result->fetch_object()){
12             $producto = array();
13             $producto["id"] = utf8_encode($fila->id);
14             $producto["nombre"] = utf8_encode($fila->nombre);
15             $producto["cantidad"] = utf8_encode($fila->cantidad);
16             array_push($respuesta["productos"], $producto);
17         }
18         $respuesta["accion"] = 1;
19     }
20     else{
21         $respuesta["accion"] = 0;
22         $respuesta["mensaje"] = "No se encontraron productos";
23     }
24     echo json_encode($respuesta);
25 ?>
```

```
{"productos":[{"id":"1","nombre":"Arroz","cantidad":"10"}, {"id":"2","nombre":"Papa","cantidad":"10"}], "accion":1}
```

```
{"accion":0,"mensaje":"No se han encontrado productos"}
```

# inicio.html

El contenido de esta plantilla se cargara al id contenido cuando se cargue la página o cuando el usuario seleccione el menú Inicio.

```
1 <h2 id="titulo">Sistema de gestion de productos</h2>
2 <div id="contenido2">
3     <p>Esta es la página de inicio del sitio.</p>
4 </div>
```

# agregar.html

El contenido de esta plantilla se cargara al id contenido cuando el usuario seleccione el menú Agregar.

```
1 <h2 id="titulo">Agregar un producto</h2>
2 <div id="contenido2">
3   Id: <input type="text" id="id"><br>
4   Nombre: <input type="text" id="nombre"><br>
5   Cantidad: <input type="text" id="cantidad"><br>
6   <input type="button" value="Agregar" onclick="enviarProducto()">
7 </div>
```

# mostrar.html

El contenido de esta plantilla se cargara al id contenido cuando el usuario seleccione el menú Mostrar.

```
1 <h2 id="titulo">Mostrar productos</h2>
2 <div id="contenido2">
3     <table id="tabla">
4         <tr>
5             <th>Id</th>
6             <th>Nombre</th>
7             <th>Cantidad</th>
8         </tr>
9     </table>
10 </div>
```

# JavaScript

A continuación se trabaja con JavaScript para que de acuerdo a la opción del menú seleccionado se cargue la página específica, este proceso se realiza cargando unas plantillas de html en el contenido de la página con la ayuda de ajax.

## script.js

El archivo JavaScript permite realizar varias tareas.

1. Cargar las plantillas en el contenido de acuerdo con la selección de menú.
2. Enviar los datos del producto al servidor y mostrar la respuesta que esta retorna.
3. Solicitar los datos a la base de datos y construir una tabla para mostrarlos.

# script.js

Inicialmente se crea el objeto AJAX, para poder interactuar con el servidor.

```
1 function obtenerAjax(){  
2     if(window.XMLHttpRequest){  
3         xhttp = new XMLHttpRequest();  
4     }  
5     else{  
6         xhttp = new ActiveXObject("Microsoft.XMLHTTP");  
7     }  
8     return xhttp;  
9 }  
10 var ajax = obtenerAjax();
```

# script.js

La plantilla inicio.html se carga cuando se inicia el sitio y cuando el usuario da clic sobre el menú inicio, igual para agregar.

Para realizar esas tareas, se solicita la información a través de AJAX.

```
12 window.onload = function() {  
13     cargarPlantilla('inicio.html');  
14 }  
15  
16 function cargarPlantilla(plantilla){  
17     ajax.open("GET", plantilla, true);  
18     ajax.onreadystatechange = respuestaCargar;  
19     ajax.send();  
20 }  
21  
22 function respuestaCargar(){  
23     if(ajax.readyState == 4 && ajax.status == 200){  
24         document.getElementById("contenido").innerHTML = ajax.responseText;  
25     }  
26 }
```



## script.js

La plantilla mostrar se carga de la misma forma, pero cuando se cargue la plantilla, se debe hacer una petición al servidor, solicitando los datos de los productos de la base de datos para mostrarlos en la tabla.

```
28 function cargarMostrar(){
29     ajax.open("GET", "mostrar.html", true);
30     ajax.onreadystatechange = respuestaMostrar;
31     ajax.send();
32 }
33
34 function respuestaMostrar(){
35     if(ajax.readyState == 4 && ajax.status == 200){
36         document.getElementById("contenido").innerHTML = ajax.responseText;
37         cargarTabla();
38     }
39 }
```

# script.js

Al oprimir el botón agregar se envía la información al servidor el cual se conecta a la base de datos y retorna una respuesta.

```
41 function enviarProducto(){
42     ajax.open("GET", "agregarproducto.php?" + obtenerQueryProducto(), true);
43     ajax.onreadystatechange = respuestaAgregar;
44     ajax.send();
45 }
46
47 function respuestaAgregar(){
48     if(ajax.readyState == 4 && ajax.status == 200){
49         var respuesta = JSON.parse(ajax.responseText);
50         alert(respuesta.mensaje)
51     }
52 }
53 function obtenerQueryProducto(){
54     var id = document.getElementById("id").value;
55     var nombre = document.getElementById("nombre").value;
56     var cantidad = document.getElementById("cantidad").value;
57     var queryString = "id=" + encodeURIComponent(id) + "&nombre=" + encodeURIComponent(nombre) + "
58         &cantidad=" + encodeURIComponent(cantidad) + "&nocache=" + Math.random();
59     return queryString;
60 }
```

## script.js

La función cargarTabla hace la solicitud de los productos al servidor y crea las filas para mostrarlos. Si se retornan datos muestran la información, sino muestra un mensaje.

```
61 function cargarTabla(){
62     ajax.open("GET", "consultarproductos.php", true);
63     ajax.onreadystatechange = respuestaTabla;
64     ajax.send();
65 }
66
67 function respuestaTabla(){
68     if(ajax.readyState == 4 && ajax.status == 200){
69         var respuesta = JSON.parse(ajax.responseText);
70         if(respuesta.accion == 0){
71             alert(respuesta.mensaje);
72         }
73         else{
74             cargarDatos(respuesta.productos);
75         }
76     }
77 }
```

## script.js

Finalmente se cargan los datos de los productos.

```
79 function cargarDatos(productos){
80     tabla = document.getElementById("tabla");
81     for(var i = 0; i < productos.length; i++){
82         var fila = document.createElement("tr");
83         tabla.appendChild(fila);
84         var celda1 = document.createElement("td");
85         fila.appendChild(celda1);
86         var id = document.createTextNode(productos[i].id);
87         celda1.appendChild(id);
88         var celda2 = document.createElement("td");
89         fila.appendChild(celda2);
90         var nombre = document.createTextNode(productos[i].nombre);
91         celda2.appendChild(nombre);
92         var celda3 = document.createElement("td");
93         fila.appendChild(celda3);
94         var cantidad = document.createTextNode(productos[i].cantidad);
95         celda3.appendChild(cantidad);
96     }
97 }
```

# Actividad

Completar el menú del sitio para actualizar, eliminar y buscar. Como se muestra en las imágenes.

La página Actualizar tendrá un select que despliega el nombre de los productos de la base de datos un input que permite modificar la cantidad de ese producto y un botón para actualizar la cantidad.

La página Borrar tendrá los mismos elementos de Actualizar, pero el input será de solo lectura, y el botón será para borrar, además mostrara un confirm para borrar el producto.

La página buscar tendrá una interfaz como Mostrar, pero tendrá un input, al cual el usuario puede ingresar letras y la tabla mostrara los productos que en el nombre incluyan ese texto ingresado en el input.

# Actividad

Inicio

Agregar

Mostrar

Actualizar

Borrar

Buscar

## Actualizar un producto

Producto

Cantidad:

## Borrar un producto

Producto

Cantidad:

## Buscar productos

Buscar:

**Id Nombre Cantidad**

1 abc 3

2 Canela 3

3 Gaseosa 5

Fernando Londoño Salgado