

M.G.T.I OSCAR JOSUÉ UH PÉREZ

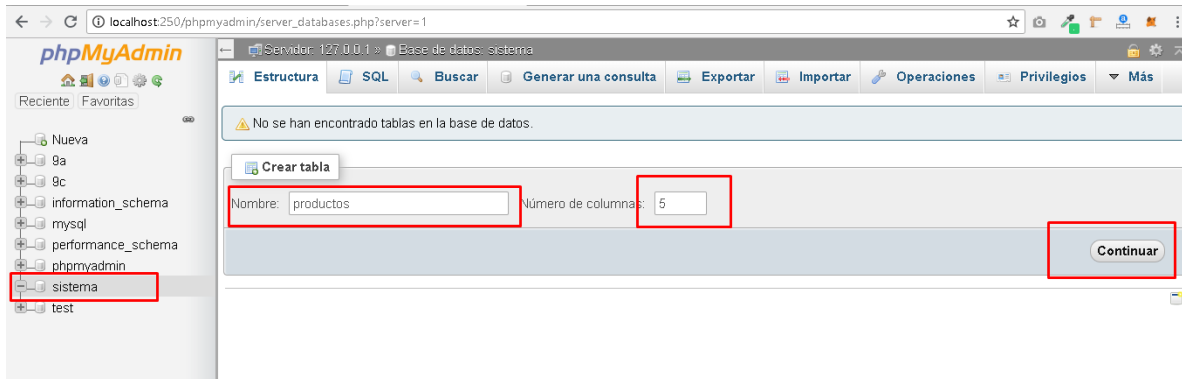


# Consulta y venta de productos con paypal.

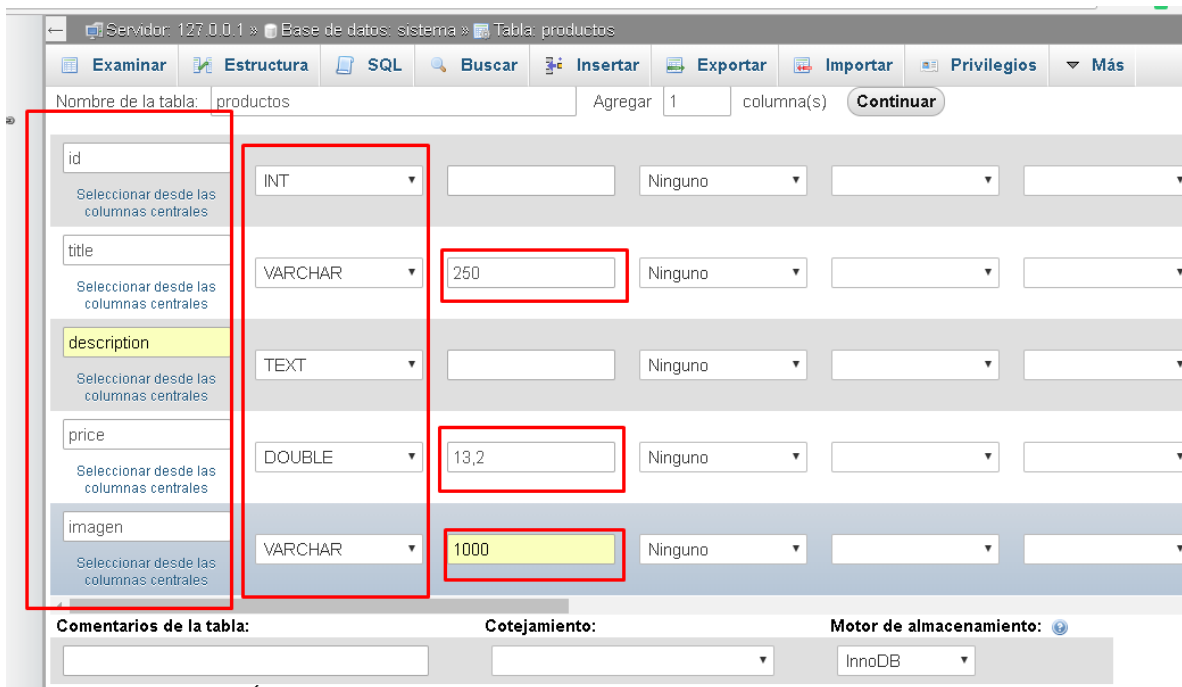
---

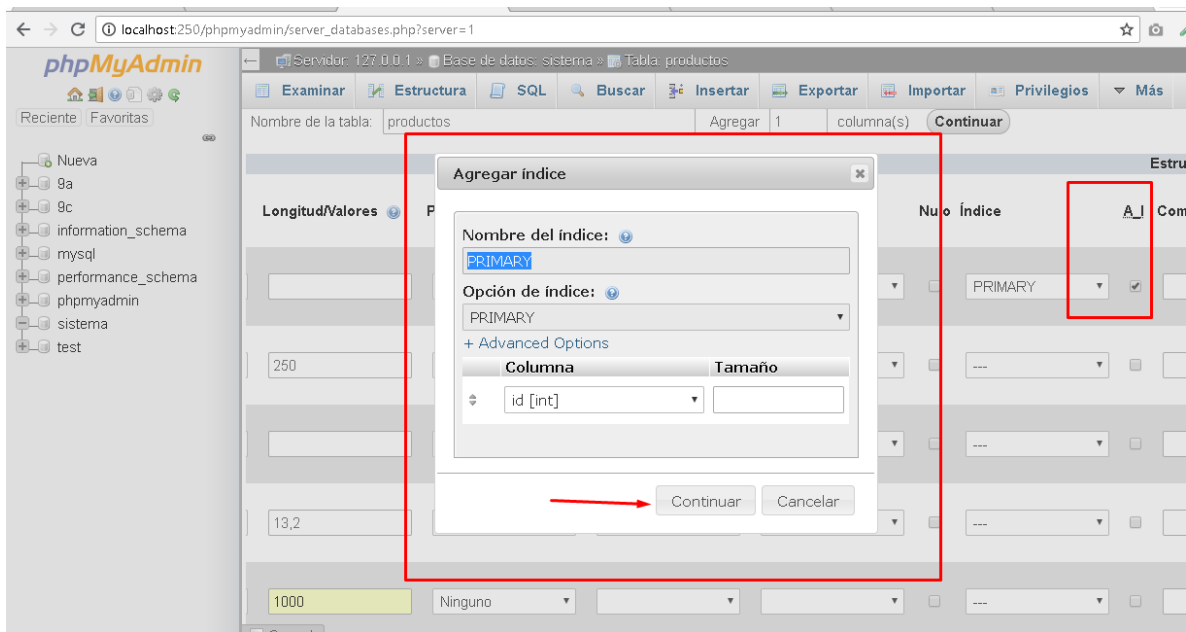
Consulta de registros en una base de datos  
MySql con la venta mediante paypal form.

1.- Vamos a crear la base de datos en MySQL que le llamaremos **sistema** con una tabla llamada **productos** con 5 campos.

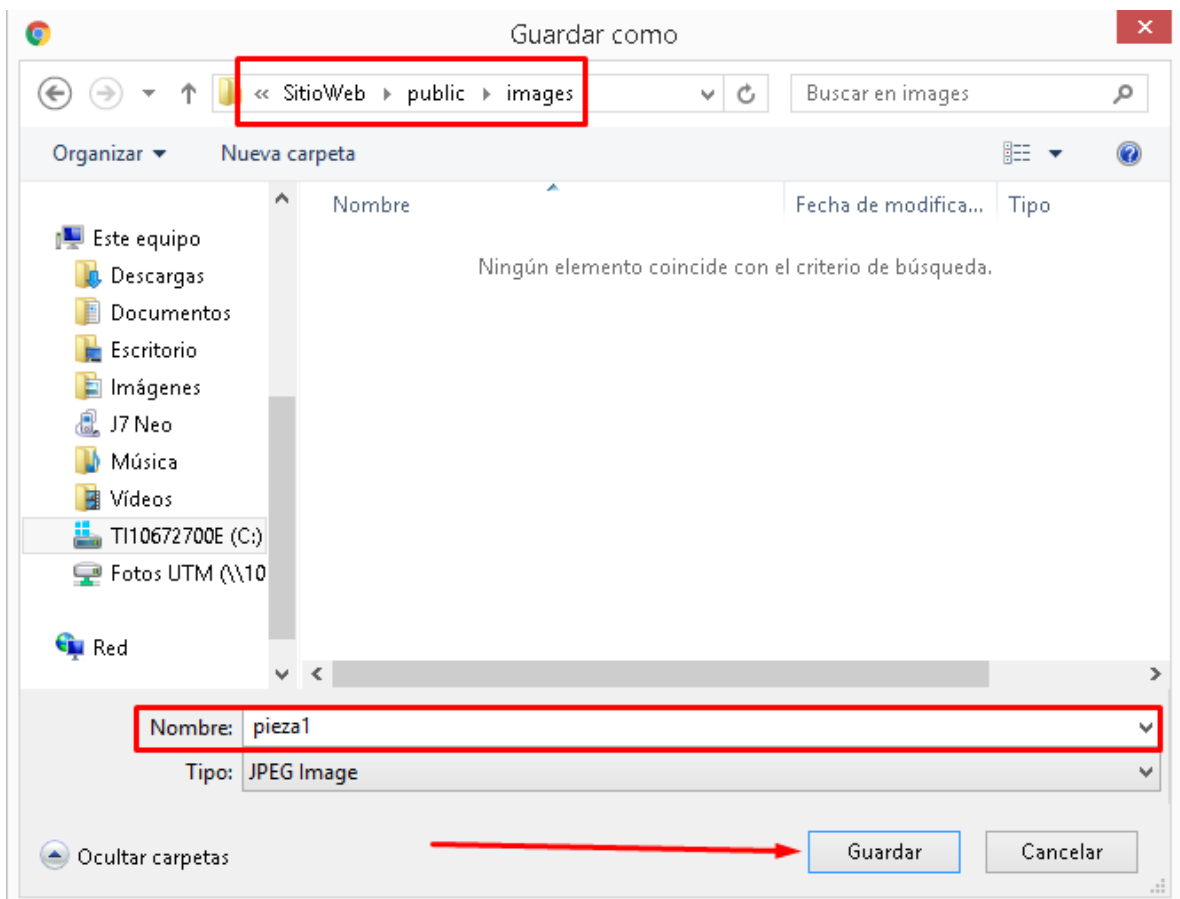


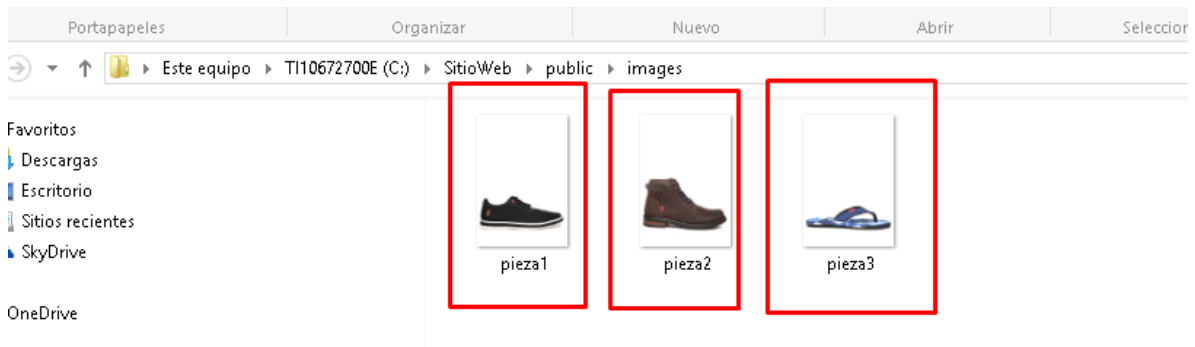
2.- A continuación crearemos las siguientes columnas (recuerda volver autoincremental el campo id):



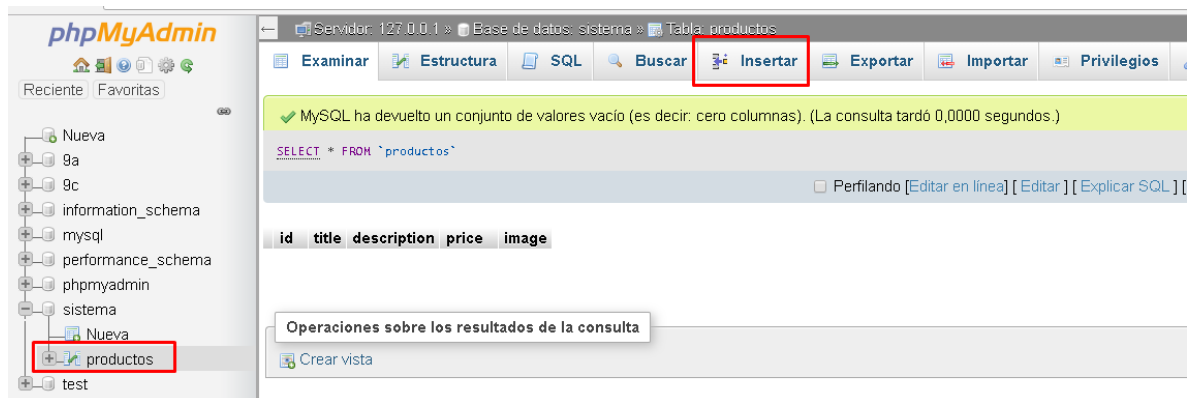


3.- A continuación crearemos 3 imágenes que estarán relacionadas con 3 registros en la tabla productos se llamaran **pieza1.jpg**, **pieza2.jpg** y **pieza3.jpg** (más adelante se podrán adjuntar estas imágenes utilizando un formulario para agregar registros).





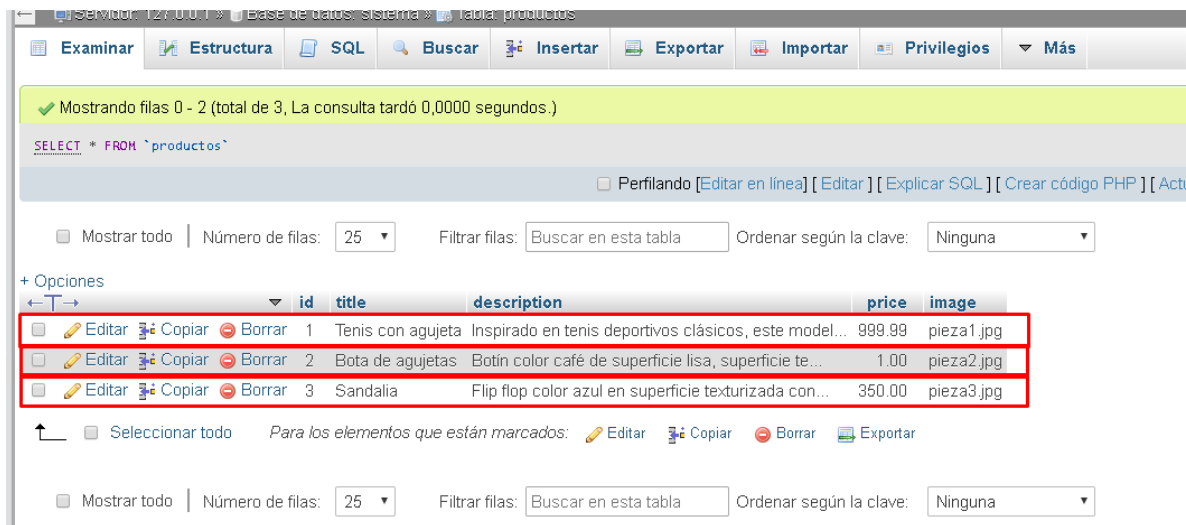
4.- Vamos a insertar 3 registros en nuestra base de datos llenando la información de cada pieza.



5.- Llena 3 registros de la tabla.

A screenshot of the phpMyAdmin 'Insertar' form for the 'productos' table. The form has columns for 'id', 'title', 'description', 'price', and 'image'. The 'title' field is filled with 'Tenis con agujeta'. The 'description' field is filled with 'Inspirado en tenis deportivos clásicos, este modelo dará un toque atractivo a lo casual por su combinación de materiales.' The 'price' field is filled with '999.99'. The 'image' field is filled with 'pieza1.jpg'. The 'id' field is empty. A 'Continuar' button is at the bottom right. A red box highlights the 'title' and 'description' fields. Another red box highlights the 'price' field. A third red box highlights the 'image' field.

## 6.- La información que da de la siguiente forma:



Mostrando filas 0 - 2 (total de 3, La consulta tardó 0,0000 segundos.)

`SELECT * FROM 'productos'`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Act]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

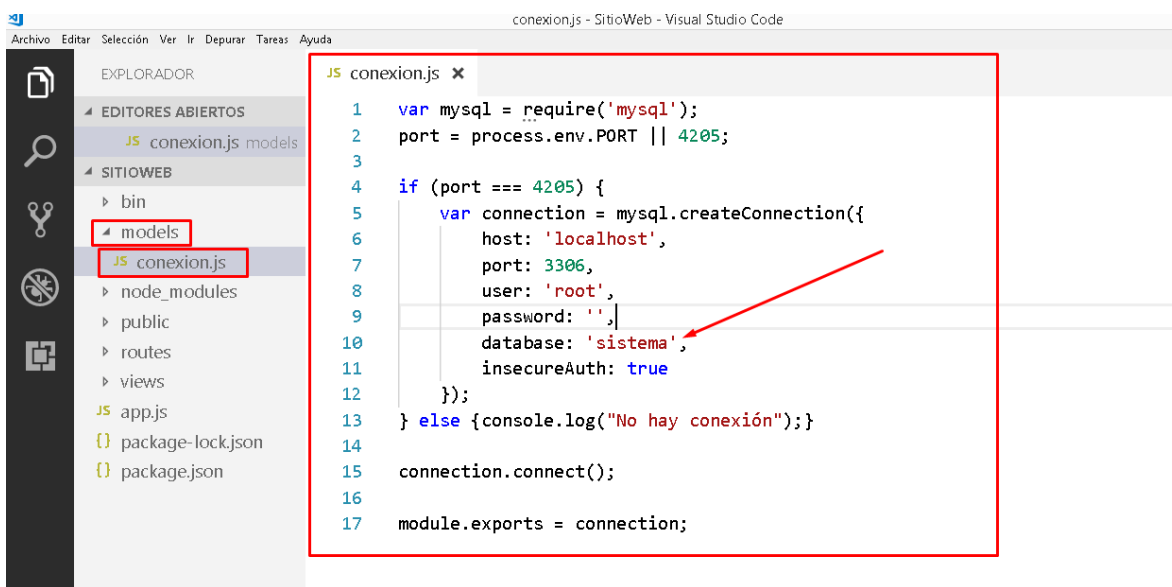
+ Opciones

|   | id | title             | description   | price  | image      |
|---|----|-------------------|---|--------|------------|
| <input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar | 1  | Tenis con agujeta | Inspirado en tenis deportivos clásicos, este model... | 999.99 | pieza1.jpg |
| <input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar | 2  | Bota de agujetas  | Botin color café de superficie lisa, superficie te... | 1.00   | pieza2.jpg |
| <input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar | 3  | Sandalia          | Flip flop color azul en superficie texturizada con... | 350.00 | pieza3.jpg |

Seleccionar todo | Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

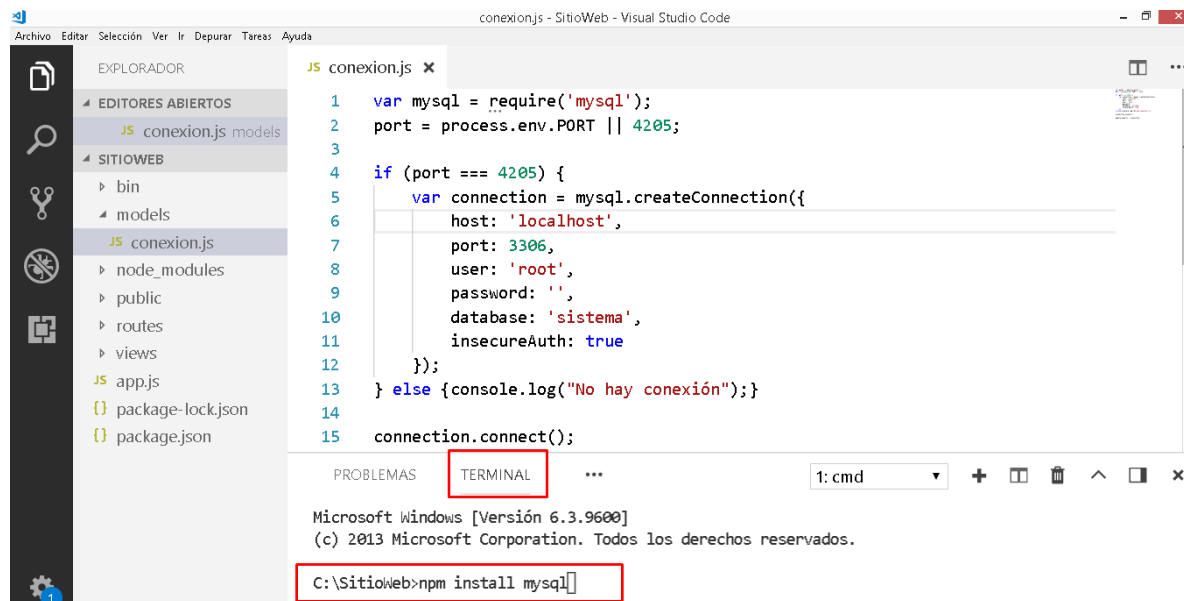
## 7.- Crearemos la conexión a la base de datos, como se muestra en la siguiente imagen:



conexion.js - SitioWeb - Visual Studio Code

```
1 var mysql = require('mysql');
2 port = process.env.PORT || 4205;
3
4 if (port === 4205) {
5   var connection = mysql.createConnection({
6     host: 'localhost',
7     port: 3306,
8     user: 'root',
9     password: '',
10    database: 'sistema',
11    insecureAuth: true
12  });
13 } else {console.log("No hay conexión");}
14
15 connection.connect();
16
17 module.exports = connection;
```

8.- En la terminal ejecutamos el siguiente comando para instalar mysql en node:

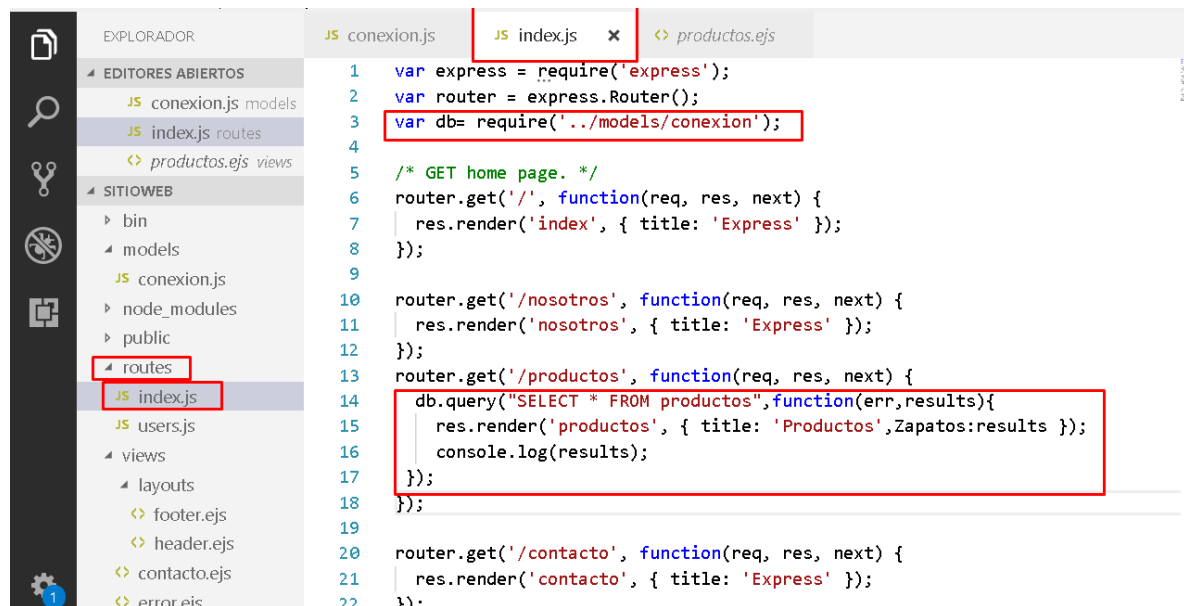


The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows the project structure with 'conexion.js' selected. The main editor displays the content of 'conexion.js', which includes code for connecting to a MySQL database. At the bottom, the Terminal panel is active, showing the command 'C:\Sitioweb>npm install mysql' entered in the command prompt.

```
1 var mysql = require('mysql');
2 port = process.env.PORT || 4205;
3
4 if (port === 4205) {
5     var connection = mysql.createConnection({
6         host: 'localhost',
7         port: 3306,
8         user: 'root',
9         password: '',
10        database: 'sistema',
11        insecureAuth: true
12    });
13 } else {console.log("No hay conexión");}
14
15 connection.connect();
```

Microsoft Windows [Versión 6.3.9600]  
(c) 2013 Microsoft Corporation. Todos los derechos reservados.  
C:\Sitioweb>npm install mysql

9.- Como siguiente paso abrimos el archivo index.js que se encuentra en la carpeta routes y agregamos nuestra consulta SQL de selección e incluimos la conexión a nuestro archivo js.



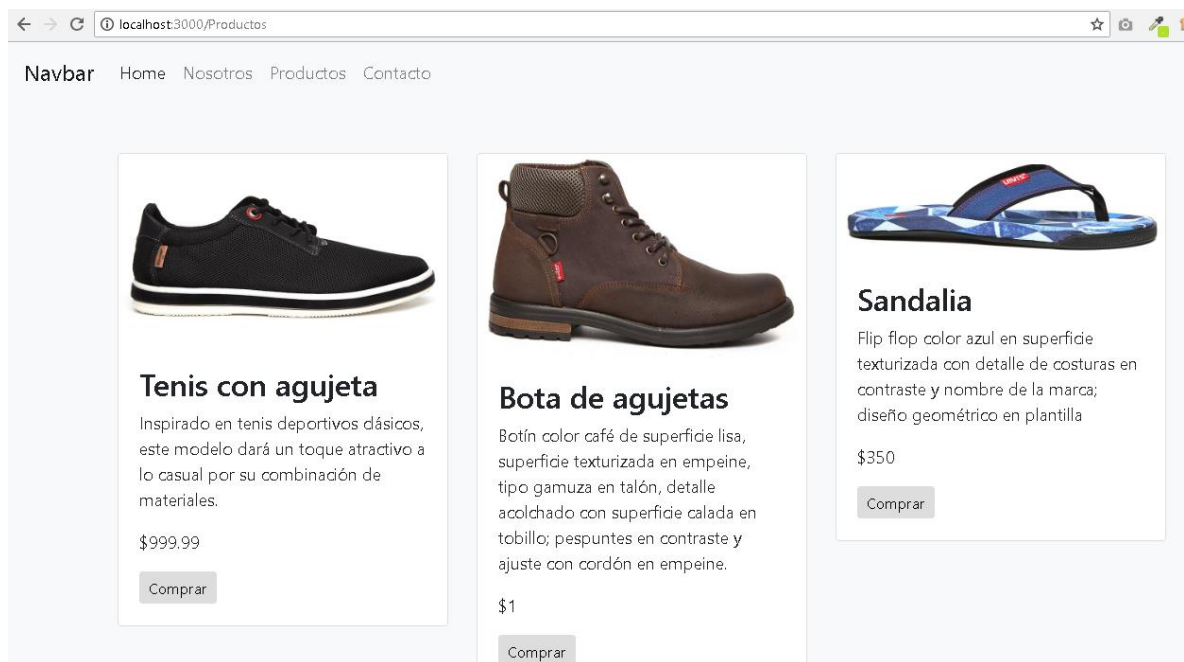
The screenshot shows the Visual Studio Code interface with 'index.js' in the 'routes' folder open. A red box highlights the line 'var db= require('../models/conexion');' and another red box highlights the SQL query and its execution logic within the '/productos' route handler.

```
1 var express = require('express');
2 var router = express.Router();
3 var db= require('../models/conexion');
4
5 /* GET home page. */
6 router.get('/', function(req, res, next) {
7     res.render('index', { title: 'Express' });
8 });
9
10 router.get('/nosotros', function(req, res, next) {
11     res.render('nosotros', { title: 'Express' });
12 });
13 router.get('/productos', function(req, res, next) {
14     db.query("SELECT * FROM productos",function(err,results){
15         res.render('productos', { title: 'Productos',Zapatos:results });
16         console.log(results);
17     });
18 });
19
20 router.get('/contacto', function(req, res, next) {
21     res.render('contacto', { title: 'Express' });
22 });
```

10.- La clave de mostrar la información está en el foreach que despliega los datos de la base de datos utilizando el template de productos.

```
products.ejs x
6   <div class="row">
7     <% for(var i=0; i<Zapatos.length;i++) { %>
8     <div class="col-md-4">
9       <div class="card mb-4 box-shadow">
10        
12        <div class="card-body">
13          <h3><%= Zapatos[i].title %></h3>
14          <p class="card-text"><%= Zapatos[i].description %></p>
15          <p class="card-text"><%= Zapatos[i].price %></p>
16          <div class="d-flex justify-content-between align-itemscenter">
17            <div class="btn-group">
18              <button type="button" class="btn btn-sm btn-outlinesecondary">Comprar</button>
19            </div>
20          </div>
21        </div>
22      </div>
23    </div>
24    <% } %>
25  </div>
```

11.- El resultado es como se muestra en la siguiente imagen:



12.- Ahora vamos a agregar paypal (tomando referencia del documento HTML que implementa paypal) lo vamos agregar debajo de la etiqueta div con clase **btn-group**

```
2      <div class="card-body">
3          <h3><%= Zapatos[i].title %></h3>
4          <p class="card-text"><%= Zapatos[i].description %></p>
5          <p class="card-text"><%= Zapatos[i].price %></p>
6          <div class="d-flex justify-content-between align-itemscenter">
7              <div class="btn-group">
8                  <form action="https://sandbox.paypal.com/cgi-bin/webscr" method="post">
9                      <input type="hidden" name="cmd" value=" xclick" />
10                     <input type="hidden" name="business" value="uhperezoscar@gmail.com" />
11                     <input type="hidden" name="currency_code" value="MXN" />
12                     <input type="hidden" name="item_name" value="<%= Zapatos[i].title %>" />
13                     <input type="hidden" name="amount" value="<%= Zapatos[i].price %>" />
14
15                     <input type="hidden" name="return" value="http://localhost:3000/" />
16
17                     <button type="submit" class="btn btn-sm btn-outlinesecondary">Comprar</button>
18                 </form>
19             </div>
20         </div>
21     </div>
22 </div>
23 </div>
```

**Nota:** El correo [uhperezoscar@gmail.com](mailto:uhperezoscar@gmail.com) es la persona beneficiada de la compra y la url sandbox es la de prueba, esta funciona con las cuentas de paypal developers.

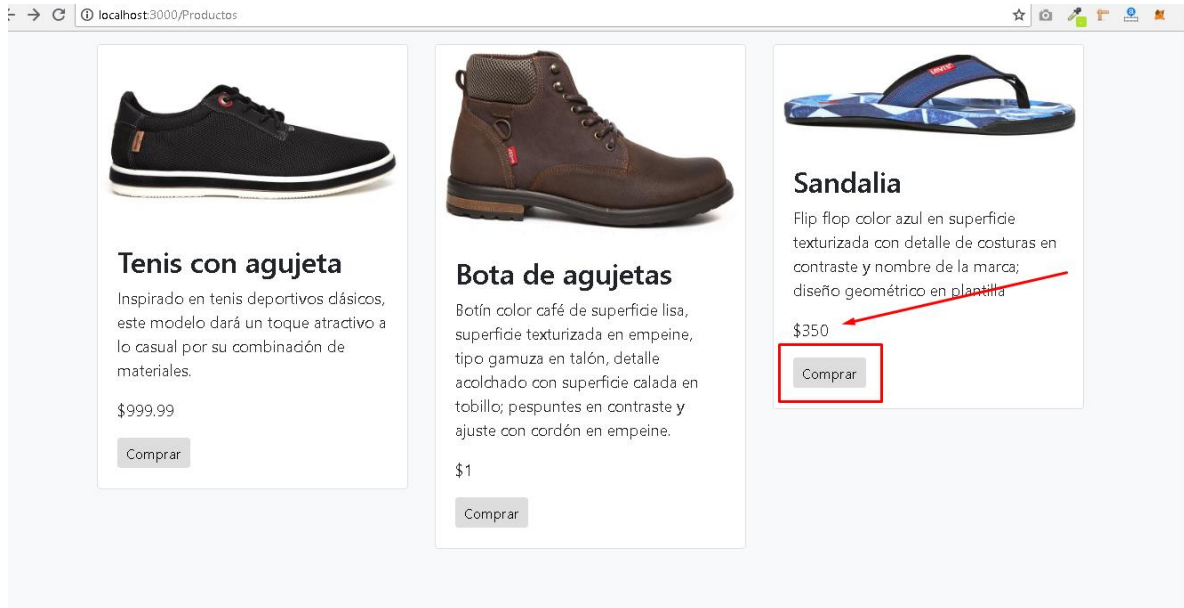
```
<div class="card-body">
    <h3><%= Zapatos[i].title %></h3>
    <p class="card-text"><%= Zapatos[i].description %></p>
    <p class="card-text"><%= Zapatos[i].price %></p>
    <div class="d-flex justify-content-between align-itemscenter">
        <div class="btn-group">
            <form action="https://sandbox.paypal.com/cgi-bin/webscr" method="post">
                <input type="hidden" name="cmd" value=" xclick" />
                <input type="hidden" name="business" value="uhperezoscar@gmail.com" />
                <input type="hidden" name="currency_code" value="MXN" />
                <input type="hidden" name="item_name" value="<%= Zapatos[i].title %>" />
                <input type="hidden" name="amount" value="<%= Zapatos[i].price %>" />

                <input type="hidden" name="return" value="http://localhost:3000/" />

                <button type="submit" class="btn btn-sm btn-outlinesecondary">Comprar</button>
            </form>
        </div>
    </div>
</div>
```



13.- Resultado se verifica haciendo clic en el producto y luego pasar a crear una cuenta con paypal developer ya que estamos en modo sandbox.



13.2 Para probar específicamente si los datos son correctos se puede dar clic en el botón abrir cuenta y observar los datos enviados del catálogo de compras.

