Ejemplo de Aplicación Web

“Formulario de Contacto”

# Objetivo:

Realizar un formulario de contacto web para una empresa que vende alimento balanceado a granel. El formulario debe pedir obligatoriamente sobre que producto se consulta, Nombre, tipo de alimento, email y un breve mensaje hasta 1000 caracteres.

Si la persona ya ha enviado el contacto le genere una cookie que le impida enviar un nuevo mensaje por 5 minutos.

Estos datos deben poder ser descargados como un archivo .csv si se envía en la url de inicio como parámetro el código especial: “passwordsecreto”. Las columnas del archivo serán “Producto”,”Nombre”,”Email”,”Mensaje”

Sin importar el stack, para el lado del usuario es transparente; este se conecta mediante un navegador (browser) al sitio mediante una URL, esa conexión ocurre mediante un pedido HTTP: El servidor recibe este pedido y devuelve: Si la url existe y está disponible, el contenido asociado a esa url, en caso contrario levanta un mensaje de error. El famoso error 404 ocurre cuando escribe una url inválida, o un subdominio inválido.

La respuesta del servidor, para poder ser visualizada por el navegador está en HTML (hypertext markup language), la cual es renderizada por el navegador, según determinadas reglas, puede ser las por defecto del navegador o las hojas de estilo en cascada (cascade style sheet). También carga los scripts que deben correr en la máquina local.

Una vez Servido el sitio este muestra un formulario de contacto, donde se muestran los campos mencionados y el botón enviar.

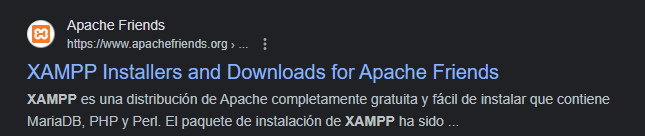
Al hacer click en el botón enviar deben validarse que los campos requeridos estén completos según las reglas de validación descriptas.

# Paso 0:

¿En qué stack vamos a implementar esto? Hay muchos stacks distintos, de sistema operativo, servidor HTTP, lenguaje de ejecución, sistema de base de datos distintos, e incluso combinaciones. El stack más popular del mundo se conoce como LAMP; Linux, Apache, MySQL PHP.

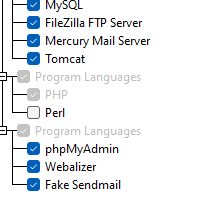
En nuestro caso utilizaremos XAMPP (cualquier sistema operativo, Apache, MySQL, Perl, PHP), que es un entorno que permite rápidamente montar un servidor de pruebas Apache, el intérprete de PHP y el servidor de MySQL en Windows, sin tener que instalar los elementos aparte, que es bastante engorroso. Se asume que se tiene Windows, pero esto se puede instalar fácilmente en Linux como en macOS.

1 Descargar en https://www.apachefriends.org/es/download.html

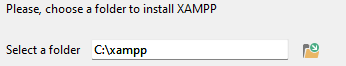


2 Durante la Instalación:

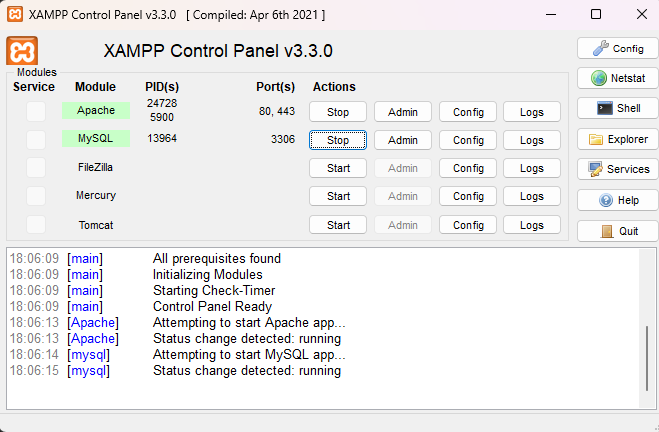
No es necesario que instalemos la parte de Perl.



Instalar en C:\XAMPP



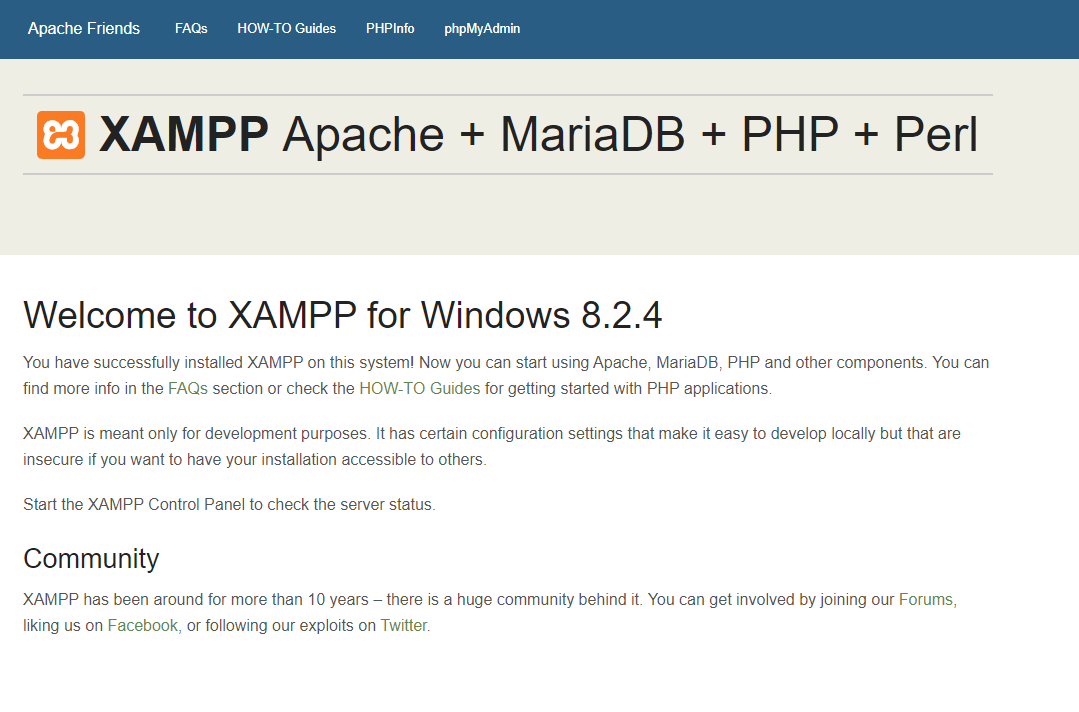
Luego de esto se selecciona idioma, por defecto en inglés (la alternativa es en alemán). Al terminar la instalación deberían ver algo como esto:



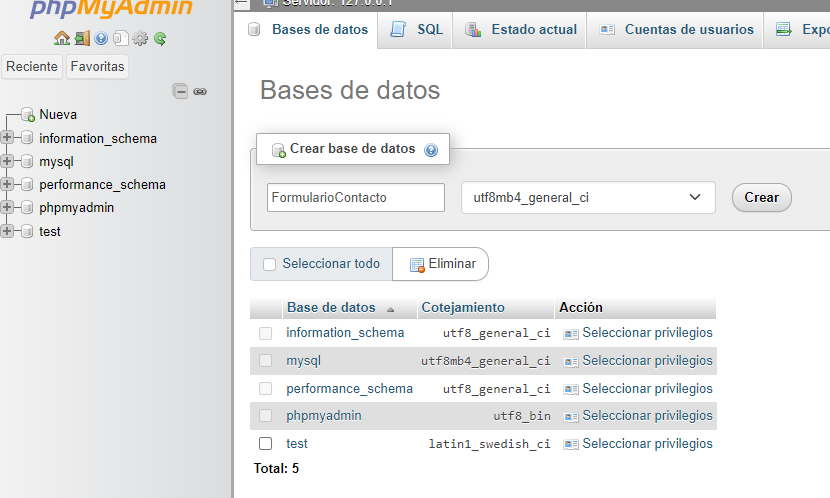
En caso de que alguna de los módulos no cargue correctamente en la consola se indica cual es el error, y Google suele encontrar rápido el resultado.

También Instalaremos Heidi SQL como herramienta de manejo de base de datos. Descargar en <https://www.heidisql.com/download.php> y dan siguiente hasta terminar.

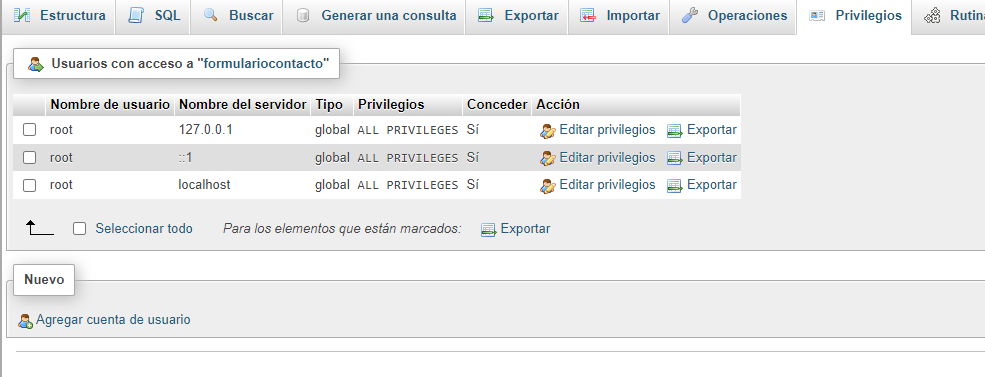
Luego Acceden a <http://localhost:80/> y deberían ver algo como esto.



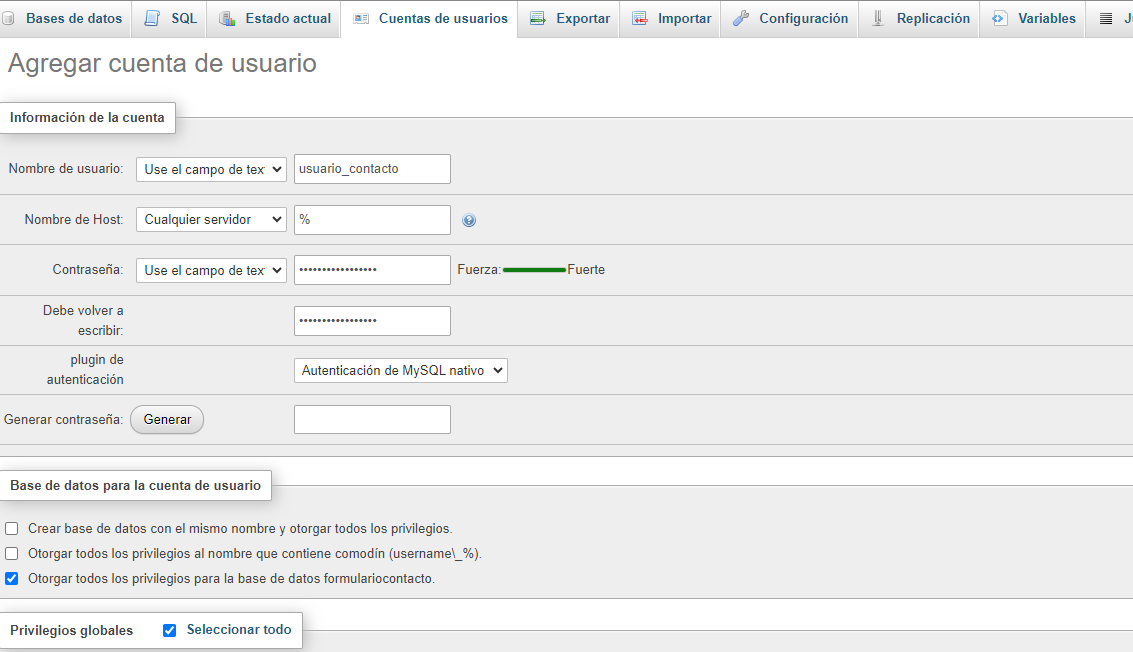
Ahí entramos a phpMyAdmin, que es un administrador de base de datos basado en formularios web, y creamos una nueva base de datos haciendo click dónde dice “Nueva” debe verse algo como esto:



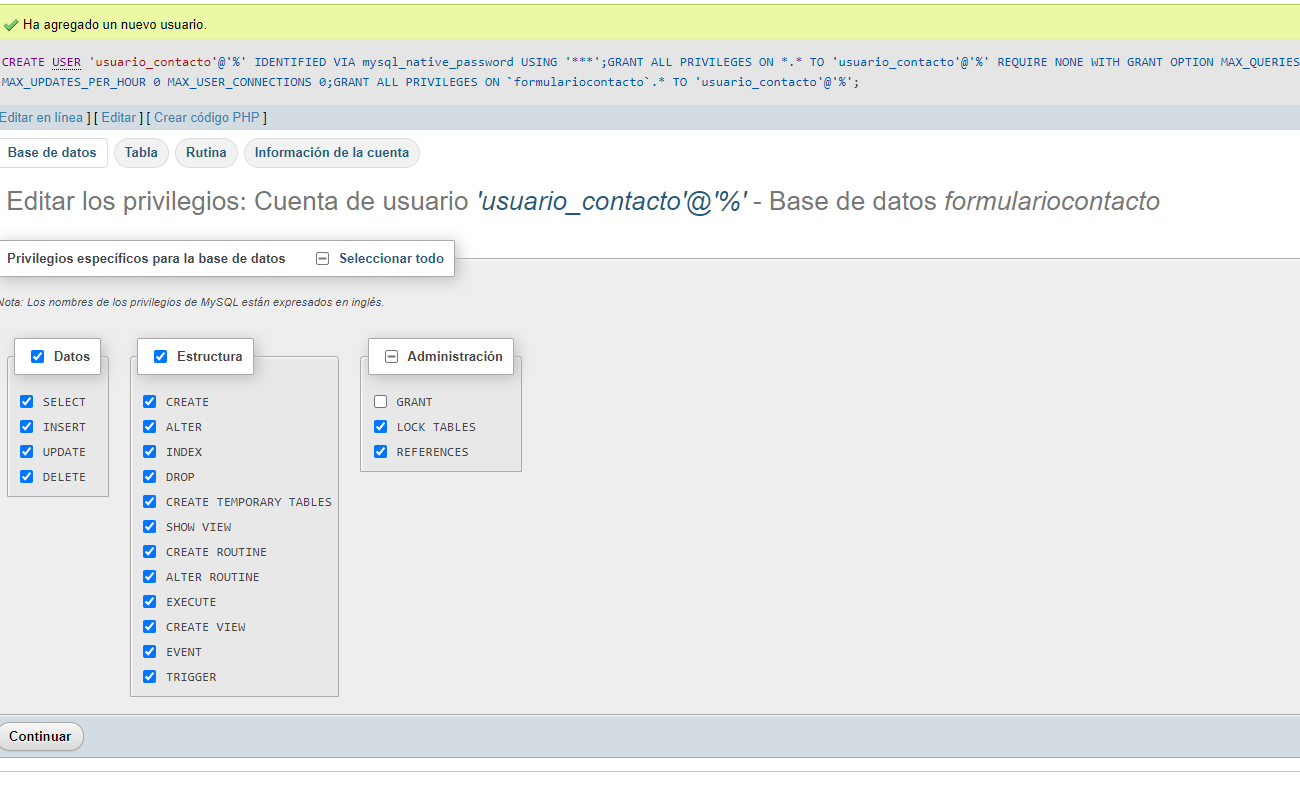
Llamaremos a la base “FormularioContacto”, luego haremos click sobre “formulariocontacto”, haremos click en la pestaña “Privilegios” y luego click en el link “Agregar cuenta de usuario”



Crearemos el usuario “usuario\_contacto”, con la contraseña “password\_contacto”, y le otorgaremos todos los privilegios globales.

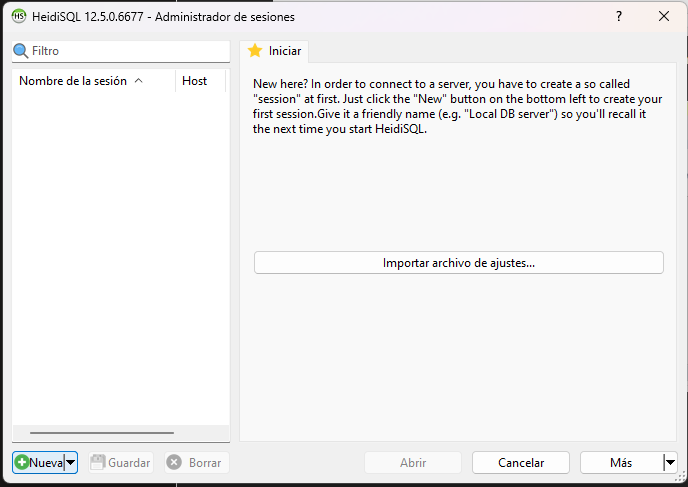


Abajo haremos click en continuar y dejaremos todo seleccionado como esta y daremos continuar nuevamente.

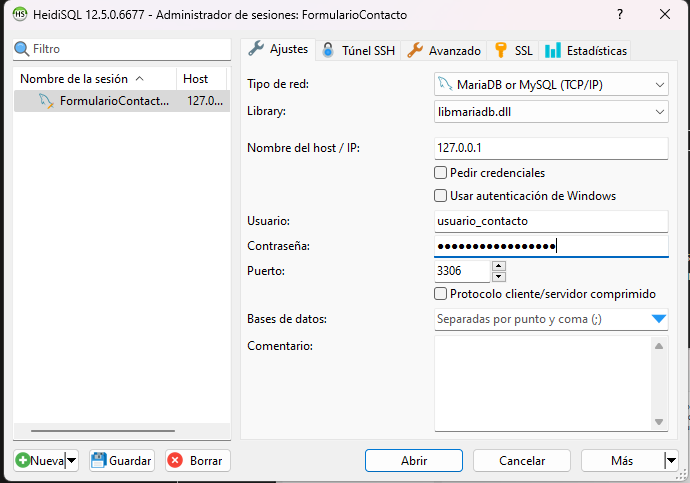


Y listo, el usuario ya, primero fue creado y luego se le dieron todos los privilegios sobre la base de datos. Ahora podemos seguir trabajando sobre el navegador, pero para quien lo considere, configuraremos Heidi SQL para conectarse a la base de datos.

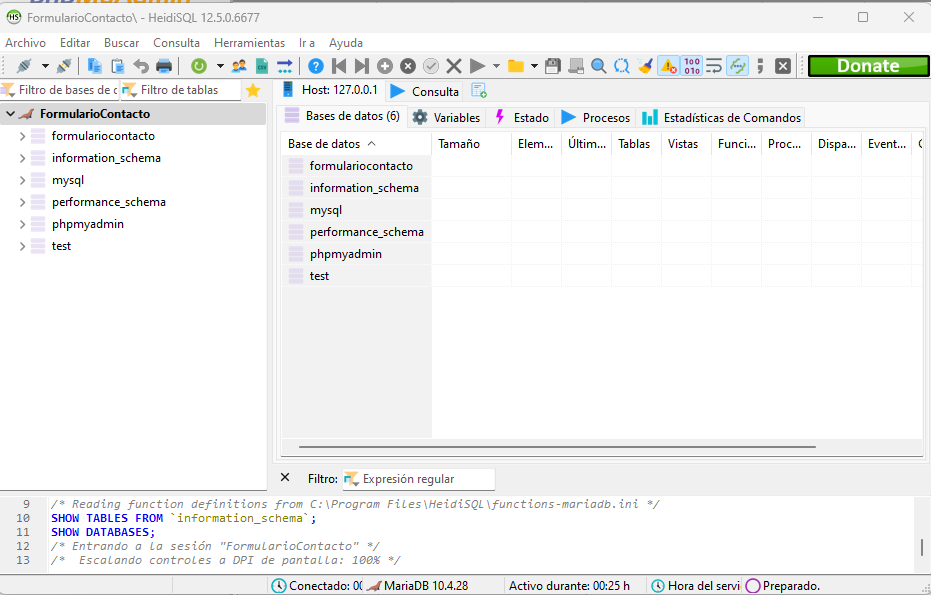
Abrimos Heidi SQL, y hacemos click en “Nueva”



Nombraremos la nueva sesión con el nombre de la base de datos, y completaremos en usuario y contraseña con los asignados previamente:



Y guardamos las modificaciones, deberíamos ver una pantalla como esta:



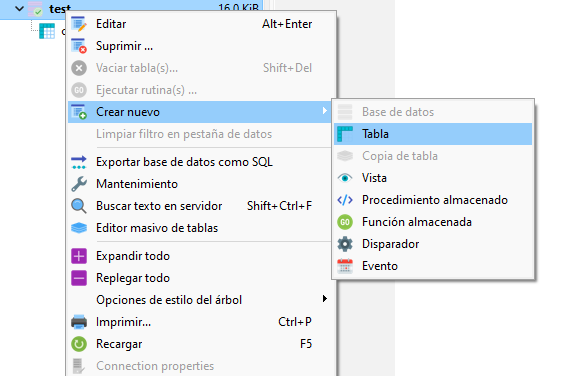
Después, como editor de texto de archivos, recomiendo usar Visual Studio Code, dado que permite abrir una carpeta y tener más al alcance los archivos, pero eso depende de cada uno, en mi experiencia personal, los 5 años que programé en PHP lo hice sin usar ningún IDE, desde Notepad++.

# Implementacion

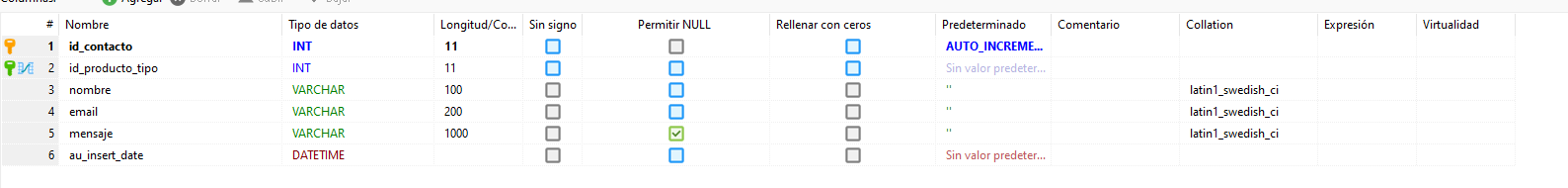
## Creación de estructura de base de datos.

Creamos el modelo de datos; es decir la estructura que vamos a utilizar para guardar los datos de nuestro formulario. En este caso será un modelo simple, son dos tablas: Contacto y ProductoTipo.

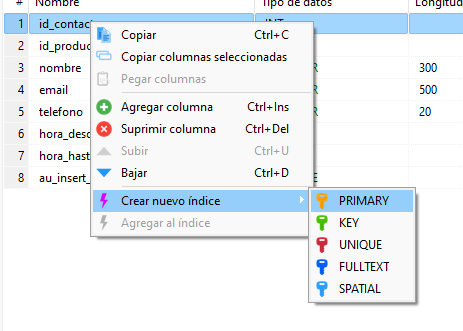
Primero Creamos la tabla Contacto



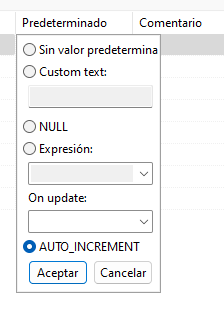
La tabla Contacto, tendrá los siguientes campos



Y se modificará id\_contacto haciendo click derecho sobre el mismo, se lo configurará como primary key.

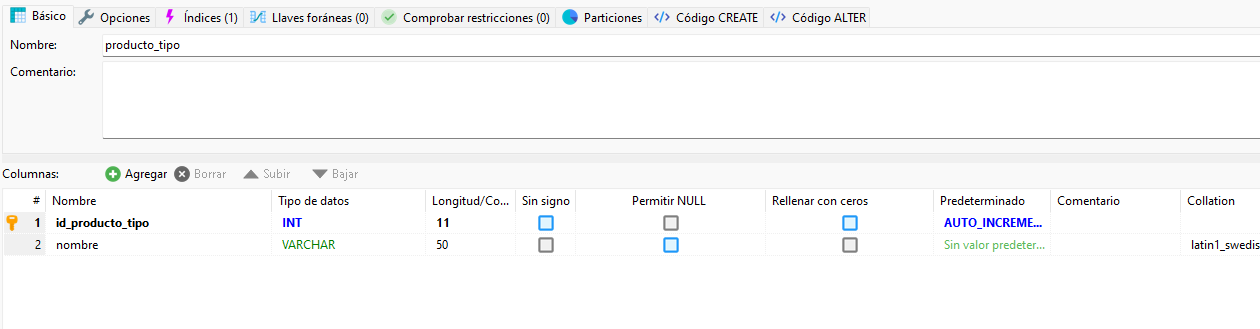


Sobre predeterminado se hará lo mismo, y se erigirá “AUTO\_INCREMENT”

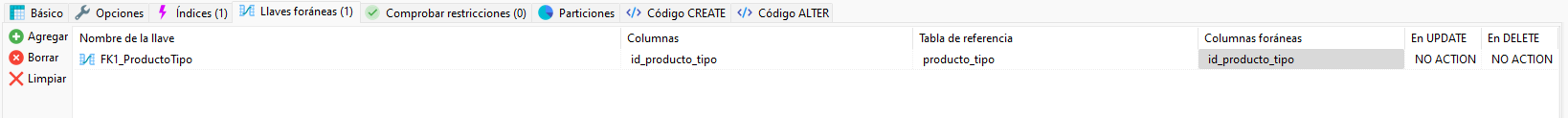


Esto hará que, para cada registro insertado, haya un nuevo id\_contacto indicando también el orden de inserción.

Luego repetimos el proceso de crear tabla, para la tabla ProductoTipo:

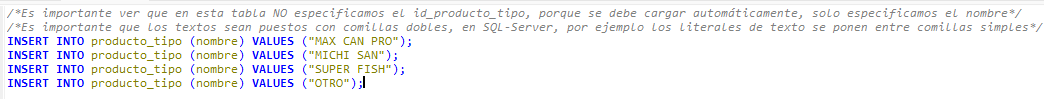


Nuevamente id\_productp\_tipo es no nulleable, int y auto incremental. Y nombre es la descripción. Esta es una tabla más simple.

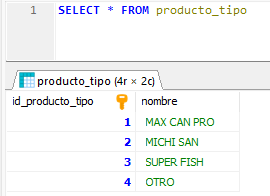
Ahora, necesitamos relacionar las dos tablas, para ello vamos a la tabla contacto, hacemos click en la pestaña “llaves foráneas” y hacemos click en “agregar”, y completamos el nombre de la llave foránea, la columna que hace referencia a la tabla foránea, la tabla de referencia, y la columna en la tabla foránea a la cual hacemos referencia y guardamos los cambios.  


Se recomienda visitar <https://www.w3schools.com/mysql/default.asp> y tenerlo como referencia al momento de armar sus propias queries.

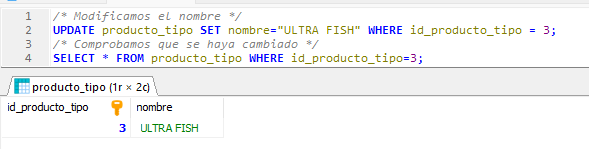
Ahora vamos a cargar unos tipos de producto para hacer pruebas, para ello empezaremos con algunas instrucciones básicas de SQL. Empezaremos con el Insert



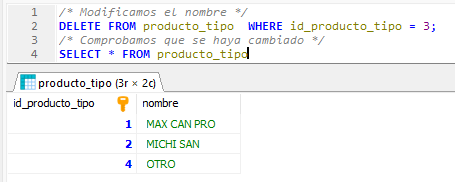
Luego el Select, que sirve para seleccionar.



El Update, es IMPORTANTISIMO no olvidarse el WHERE, caso contrario se cambian todos los registros de la tabla.

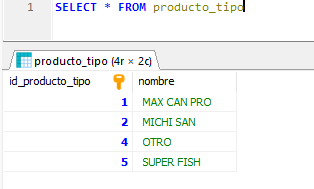


Por último, existe la opción Delete, que debe especificarse con un where, de lo contrario borraría toda la tabla



Volvemos a insertar:

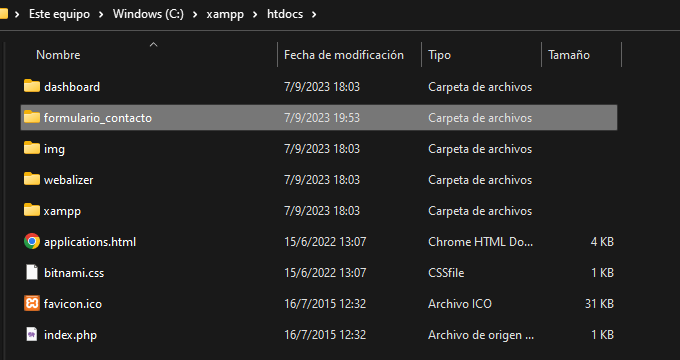
INSERT INTO producto\_tipo nombre VALUES ("SUPER FISH");



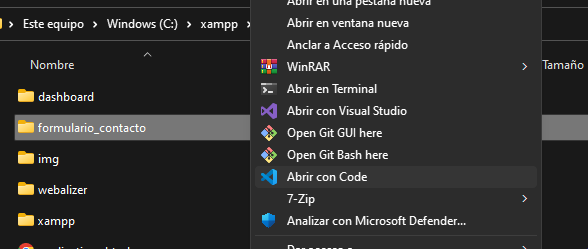
Y vemos que el id\_producto\_tipo, en lugar de ser 4, es 5, dado que al momento de insertar 4, este se cambió por 5 para el siguiente registro a ser insertado.

## Implementación del código

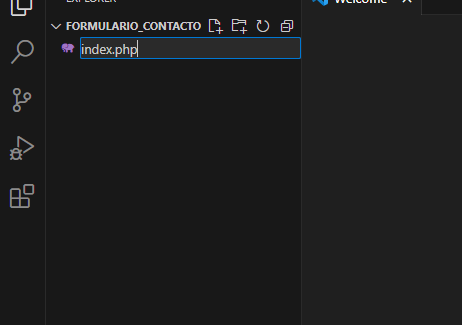
Teniendo la base de datos lista, y sus tablas armadas, empezamos con el código. Primero creamos un directorio para la aplicación web en el siguiente directorio:



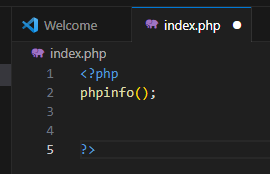
Si instalaron Visual Code, van a tener la opción de abrir con el programa



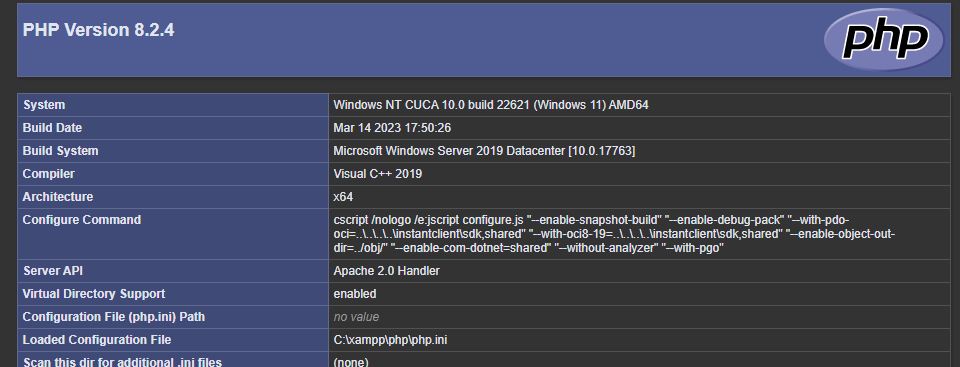
Luego creamos un archivo llamado “index.php”.



Y escribimos arriba de todo.



Esto dice que el texto entre <?PHP ?> debe ser interpretado antes de ser enviado al cliente. La función phpinfo() muestra información sobre el sistema.



Bien, PHP está funcionando, ahora vamos a buscar un template en HTML para visualizar el formulario de contacto, luego trabajaremos sobre él.

Para template de HTML, utilizaremos Bootstrap (<https://getbootstrap.com/>), luego copiaremos y modificaremos de la sección de ejemplos el ejemplo de Checkout, que tiene un formulario.

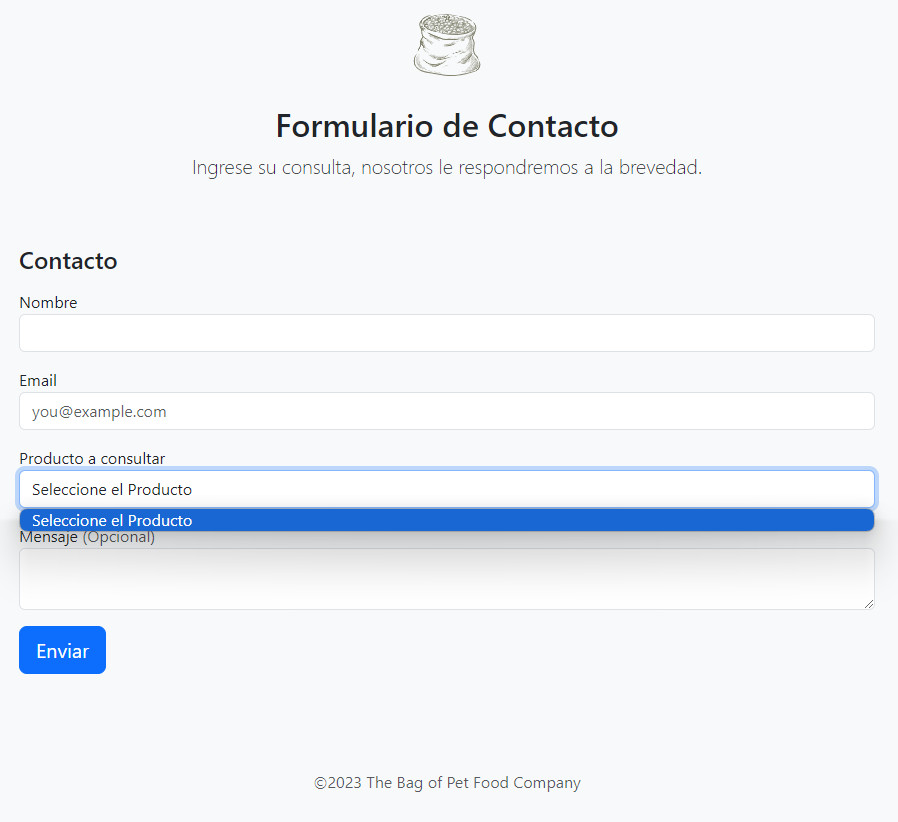
Los archivos CSS, y JS, que utiliza Bootstrap los podemos hostear en nuestro servidor, o en caso de contar con conexión a internet referenciarlos en el CDN, para este ejemplo haré esto último para mantener la complejidad al mínimo:

|  |
| --- |
| [https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css](mailto:https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css) |
|  |
| [https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js](mailto:https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js) |

Luego modificamos el ejemplo de checkout para dejar solo lo que necesitamos. Compartiré junto con este documento las fuentes en un repositorio de github.

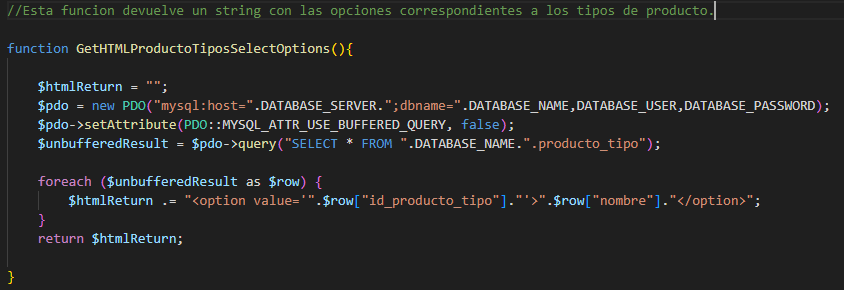
https://github.com/davicari/formulario\_escuela

Luego de las modificaciones el formulario queda así:



Se puede ver que en los productos a consultar, no se halla nada. Para esto crearemos la función GetHTMLProductoTiposSelectOptions()

Que trae de la base de datos lo que necesitamos:



y lo muestra en el HTML:

