# CRV SHOP


Daniel Osuna Molero
Proyecto PHP
2ºDAW

# ÍNDICE

# Instalación

Para ejecutar este proyecto en **XAMPP**:

- Clona el repositorio o copia los archivos en la carpeta de tu servidor local (htdocs en XAMPP).
- Enlace de clonacion: ***https://github.com/DanielOsunaMolero/Desarrollo-Entorno-Servidor.git***
- Crea la base de datos tienda en phpMyAdmin e importa el archivo database.sql.
- Configura la conexión en config/db.php
- Asegúrate de que mod_rewrite esté activado en Apache (.htaccess lo usa).
- Accede al sitio desde: **http://localhost/**
- **Credenciales Admin**
    - **Correo : daniel@daniel.com**
    - **Password : daniel**

# Introducción

**CRV SHOP** es un sistema web de comercio electrónico desarrollado en **PHP** con una estructura **MVC (Modelo-Vista-Controlador)**.

La web está dedicada a una tienda de venta de videojuegos y complementos de estos como pueden ser mandos , consolas y todo lo relacionado con ellos.

Permite a los usuarios buscar los productos que quieran ordenados por categorías , registrarse en la web, editar su perfil y añadir productos a un carrito.

El usuario administrador tiene la potestad de gestionar categorías, pedidos, usuarios y productos.

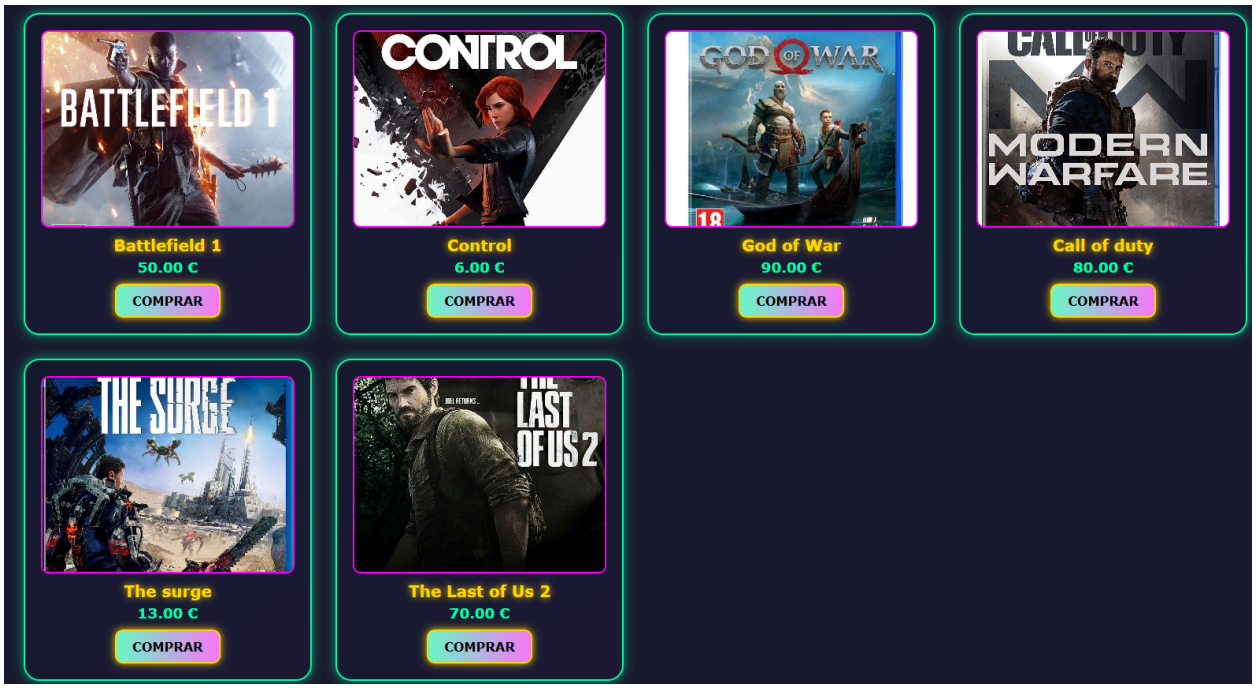
# Funcionalidades Principales

**Usuarios**

- Registro e inicio de sesión
- Diferenciación entre **usuarios estándar** y **administradores**
- Los administradores pueden **crear, modificar y eliminar usuarios**

## Productos

- Visualización de productos por categorías
- Los administradores pueden **crear, editar y eliminar productos**
- Posibilidad de agregar una imagen a cada producto




## Categorías

- Los productos están organizados por categorías
- Los administradores pueden **crear, editar y eliminar categorías**

## Estructura del Proyecto

El proyecto sigue una estructura **MVC**, con las siguientes carpetas y archivos clave:

```
Proyecto _Final PHP/
├── assets/                 # Archivos estáticos (CSS, imágenes)
│   ├── css/                # Hojas de estilo
│   ├── img/                # Imágenes de productos
├── config/                 # Configuración de la base de datos y parámetros globales
├── controllers/            # Controladores PHP
├── database/               # Base de datos y scripts SQL
├── helpers/                # Utilidades adicionales
├── models/                 # Modelos que manejan la base de datos
├── uploads/                # Almacenamiento de imágenes subidas
├── views/                  # Vistas (HTML + PHP)
│   ├── layout/             # Partes comunes (header, sidebar, footer)
│   ├── producto/           # Vistas de productos
│   ├── categoría/          # Vistas de categorías
│   ├── usuario/            # Vistas de usuarios
├── autoload.php            # Carga automática de clases
├── index.php               # Punto de entrada principal
├── .htaccess               # Configuración de rutas amigables
```

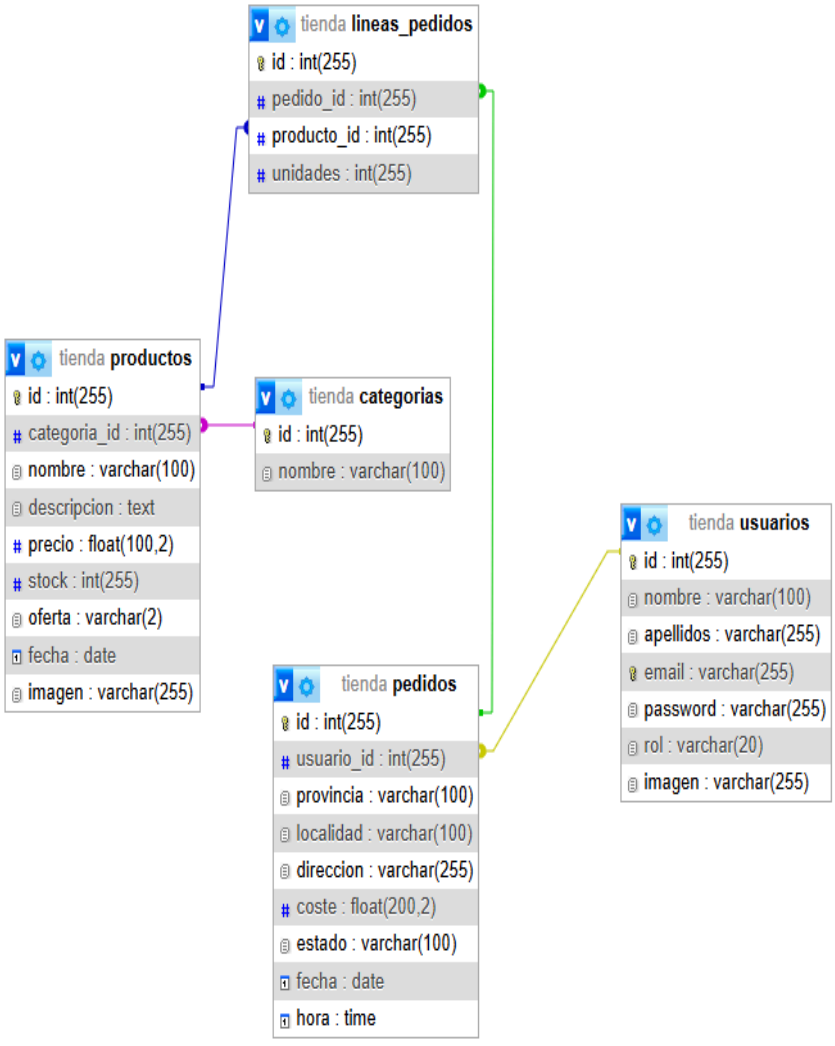
## Tecnologías Utilizadas

- **Lenguaje Backend:** PHP 8.x
- **Base de Datos:** MySQL (phpMyAdmin)
- **Frameworks y Librerías:** Ninguno, PHP puro con arquitectura MVC
- **Frontend:** HTML, CSS (diseño futurista y juvenil)
- **Autenticación:** Sesiones en PHP



# Estructura de la Base de Datos

El proyecto utiliza phpMyAdmin como gestor de base de datos y la base de datos se llama tienda. Está compuesta por las siguientes tablas:



## Usuarios

Guarda la información de los usuarios del sistema.

| Campo | Tipo | Descripción |
|---|---|---|
| id | INT AUTO_INCREMENT PRIMARY KEY | Identificador único del usuario. |
| nombre | VARCHAR(100) | Nombre del usuario. |
| apellidos | VARCHAR(100) | Apellidos del usuario. |
| email | VARCHAR(100) UNIQUE | Correo electrónico del usuario. |
| password | VARCHAR(255) | Contraseña encriptada del usuario. |
| rol | ENUM('user', 'admin') DEFAULT 'user' | Define si el usuario es administrador o cliente. |
| imagen | VARCHAR(255) NULL | Ruta de la imagen de perfil (opcional). |

## Categorías

Contiene diferentes categorías de productos.

| Campo | Tipo | Descripción |
|---|---|---|
| id | INT AUTO_INCREMENT PRIMARY KEY | Identificador único de la categoría. |
| nombre | VARCHAR(100) | Nombre de la categoría. |

## Productos

Almacena los productos disponibles en la tienda.

| Campo | Tipo | Descripción |
|---|---|---|
| id | INT AUTO_INCREMENT PRIMARY KEY | Identificador único del producto. |
| categoria_id | INT | Clave foránea que referencia a categorías(id). |
| nombre | VARCHAR(255) | Nombre del producto. |
| descripción | TEXT | Descripción detallada del producto. |

| | | |
|---|---|---|
| precio | DECIMAL(10,2) | Precio del producto. |
| stock | INT | Cantidad disponible en inventario. |
| imagen | VARCHAR(255) NULL | Ruta de la imagen del producto (opcional). |

## Modelos

| **Categoría** | |
|---|---|
| **Método** | **Descripción** |
| getId() | Obtiene el ID de la categoría. |
| getNombre() | Obtiene el nombre de la categoría. |
| setId($id) | Asigna un ID a la categoría. |
| setNombre($nombre) | Asigna un nombre a la categoría asegurando que sea seguro para la base de datos. |
| getCategorías() | Obtiene todas las categorías de la base de datos, ordenadas por ID en orden descendente. |
| save() | Guarda una nueva categoría en la base de datos. |
| getById() | Obtiene una categoría específica de la base de datos según su ID. |

| Producto | |
|---|---|
| **Método** | **Descripción** |
| getProductos() | Obtiene todos los productos de la base de datos ordenados por ID. |
| getRandom($limit) | Obtiene una cantidad aleatoria de productos de la base de datos. |
| save() | Permite guardar un producto en la base de datos |
| getById($id) | Obtiene un producto según su ID |
| update() | Permite actualizar los datos de un producto |
| delete() | Elimina un producto de la BBDD |
| getByCategoria($categoria_id) | Obtiene todos los productos de una categoría |

| Usuario | |
|---|---|
| **Método** | **Descripción** |
| save() | Permite guardar un nuevo usuario. |
| saveAdmin() | Permite guardar un nuevo usuario con un rol definido |
| crear() | Muestra la vista de la creación de un usuario solo para administradores |
| update() | Actualiza la información de un usuario en la BBD |
| delete() | Elimina un usuario de la BBDD según su ID |
| getAll() | Obtiene todos los usuarios de la base de datos |
| getById($id) | Obtiene un usuario específico |
| login() | Controla la autenticación |

# Controladores

## CategoríaController (controllers/CategoriaController.php)

- **Descripción**

Maneja la gestión de categorías en la tienda, permitiendo a los administradores crear y listar categorías.

| Método | Descripción |
|---|---|
| index() | Obtiene todas las categorías y las muestra a la vista. |
| crear() | crear() |
| save() | Guarda una nueva categoría en la base de datos. |

## ErrorController (controllers/ErrorController.php)

- **Descripción**

Controlador de errores, maneja páginas no encontradas.

| Método | Descripción |
|---|---|
| index() | Muestra un mensaje de error cuando una página no existe |

## Product Controller (controllers/ProductoController.php)

- **Descripción**

Administra los productos de la tienda, permitiendo a los administradores crear y listar.

| Método | Descripción |
|---|---|
| index() | Obtiene productos destacados y los muestra en la vista principal. |
| gestion() | Lista todos los productos para gestión administrativa. |
| crear() | Muestra la vista de creación de productos. |
| save() | • Guarda un producto en la base de datos, incluyendo la subida de imágenes.<br>• Valida los campos enviados en POST<br>• Guarda la imagen en la carpeta uploads/images/<br>• Inserta los datos en la base de datos<br>• Redirige a la vista de gestión de productos |

## Usuario Controller (controllers/UsuarioController.php)

- **Descripción**

Maneja la autenticación y administración de usuarios, permitiendo a los administradores crear, editar, eliminar y gestionar usuarios.

| Método | Descripción |
|---|---|
| index() | Lista todos los usuarios en la vista de gestión. |
| registro() | Muestra la vista de registro de usuarios. |
| gestion() | Muestra la vista de administración de usuarios. |
| save() | Registra un usuario estándar (user). |
| save Admin() | Permite a un administrador registrar otro usuario (user o admin). |

| | |
|---|---|
| crear() | Muestra la vista de creación de usuarios. |
| editar() | Carga un usuario específico para editarlo. |
| update() | Actualiza los datos de un usuario en la base de datos. |
| eliminar() | Elimina un usuario de la base de datos. |
| login() | Maneja la autenticación de usuarios. |
| logout() | Cierra la sesión de usuario |

## Vistas

Las vistas forman parte del modelo MVC siendo la parte visual de este y proporcionado al usuario una sensación de navegación a través de llamadas de los controladores a estas vistas.

| **Layout** | |
|---|---|
| header.php | Contiene el header de la web con el logo, el titulo y la barra de navegacion de categorias |
| sidebar.php | Contiene todos los botones para la gestión de la web junto con el formulario de login |
| footer.php | Contiene |

-

| **Categoría** | |
|---|---|
| Crear.php | La vista enseña el formulario para crear una categoría compuesto por un solo label y un botón de guardar |
| index.php | Permite ver la tabla de gestion de categorias desde un usuario administrador |

| **Productos** | |
|---|---|
| categoría.php | Se encarga de listar en pantalla todos los productos de la categoría seleccionada |
| crear.php | Contiene el formulario para el registro de nuevos productos desde la gestión del administrador, todo esto incluyendo validación de formularios |
| destacados.php | Se encarga de mostrarnos algunos productos aleatorios en el index de la página, es decir se encarga de enseñar algunos productos de todas las categorías para llamar la atención del comprador |
| editar.php | Permite la edición de los productos ya creados mediante un formulario desde la sesión de un administrador |
| gestion.php | Esta vista ofrece un listado de todos los productos desde donde un administrador puede editarlos clicando en Editar(lleva a editar.php) o eliminar el producto |

| **Usuario** | |
|---|---|
| crear.php | Permite la creación de usuarios desde una sesion de administrador |
| editar.php | Permite editar todos los campos de un usuario desde cualquiera de las sesiones (el rol únicamente en sesion administrador) |
| gestion.php | Ofrece una lista de todos los usuarios que permite acceder a su edición (editar.php) o su eliminación |
| registro.php | Permite a un usuario que no está registrado registrarse mediante un formulario que está validado |

# Configuración

- Archivo **db.php** : define una clase que gestiona la conexión a la base de datos utilizando MySQLi. Permite establecer una conexión que se usa en toda la aplicación.
- Archivo **parameters.php** : Este archivo contiene definiciones de constantes que establecen configuraciones globales para la aplicación, como la URL base, el controlador predeterminado y la acción por defecto.