

# Aplicación DemoDogs Stranger Things

Juan Esteban Gómez Pachón [juan.gomez.pachon@pi.edu.co](mailto:juan.gomez.pachon@pi.edu.co)

Facultad de ingeniería del politécnico internacional Programación II



GitHub: <https://github.com/JuanEstebanGPX> 

YouTube: <https://www.youtube.com/watch?v=GZpAgqXVqrw>

**Resumen**—El proyecto es un aplicativo de Stranger Things DemoDogs para los fanáticos de dicha serie, también estará dirigida para los curiosos que deseen conocer más de la serie, tendrá un rol de administrador para añadir y modificar información, contará con dos roles más uno de fanático y el otro de curioso, se desarrollará con el lenguaje de programación PHP y su base de datos estará en MySQL, se usará como IDE Visual Studio Code, se implementará en el proyecto la arquitectura de 3 capas para su correcto funcionamiento, para hacer más didáctica la aplicación se agregará un video juego para el usuario.

**Palabras claves:** php HTML- CSS -UbuntuServer  
Modelo 3 capas -VirtualBox - MySQL

**Abstract-** The project is an application of Stranger Things DemoDogs for fans of said series, it will also be aimed at curious people who want to know more about the series, it will have an administrator role to add and modify information, it will have two roles plus one of fanatic and the other curious, it will be developed with the PHP programming language and its database will be in SQL server Express, Visual Studio Code will be used as IDE, the 3 - layer architecture will be implemented in the project for its correct operation, to do more didactic the application will add a video game for the user.



## VERSIONES DEL DESARROLLO

Visual Studio Code  
VirtualBox 6.0.24  
PHP Versión 8.1.2  
APACHE V8  
HTML 5  
CSS3  
MySQL Server 8.0  
Ubuntu Server 20  
Arquitectura modelo de 3 capas

## Introducción

Para el Desarrollo de la aplicación usaremos la programación Orientada a objetos, se realizará atributos y clases dando buenas prácticas al nombramiento de estas, utilizando abstracción, polimorfismo, encapsulamiento, herencia.

### Abstracción

buscaremos las características de los objetos de la aplicación para convertirlos en clases, se analizará las características de los personajes, monstruos, música, escenarios.

### polimorfismo

al declarar las variables con los objetos en la aplicación gracias al polimorfismo de la programación orientada a objetos podemos

heredar clases y atributos, para usarlos en todo el desarrollo de la aplicación.

### Encapsulamiento

Ya una vez declaremos las variables y objetos que usaremos, debemos encapsular el uso de las variables dentro de la aplicación, usaremos un private, protected, public, para administrar el uso de las variables dentro del desarrollo.

### herencia

usaremos esta propiedad de la programación orientada a objetos para definir y unir los objetos y clases como nombre de los personajes y la temporada en la que participan, podremos disponer de la información de ambas clases.

## ARQUITECTURA MODELO DE 3 CAPAS

Se implementará el modelo de 3 capas en la aplicación demodogs. En donde la capa 1 será de la presentación del proyecto, en la capa 2 negocio y ejecución de la aplicación se desarrollará la parte grafica de la aplicación, interfaces tanto del rol de administrador, fanático, curioso, por último, en la capa 3 se conectará al servidor y a la base de datos con la información de la aplicación.

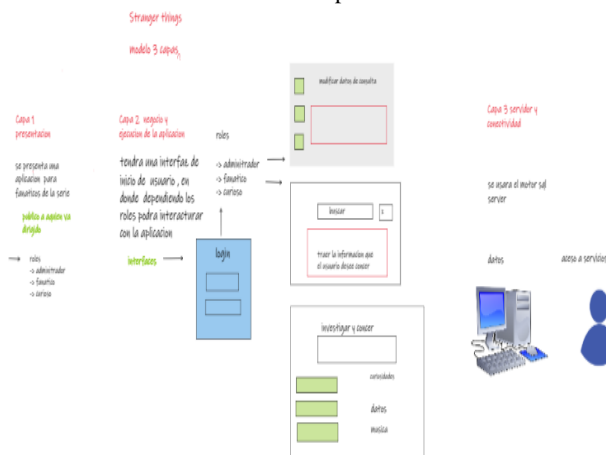


Fig. 1 arquitectura modelo 3 capas

## INSTALACIÓN DEL SERVIDOR

Se descargo desde la página oficial de Ubuntu una iso servidor para crear el servidor que llevara la aplicación DemoDogs.

Get Ubuntu Server

Option 2: Manual server installation

USB or DVD image based physical install

- OS security guaranteed until April 2027
- Extended security maintenance until April 2032
- Commercial support for enterprise customers

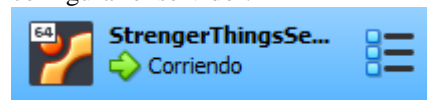
[Download Ubuntu Server 22.04 LTS](#) [Alternative downloads](#) [Alternative architectures](#)



Descarga:



Configuramos en VirtualBox la maquina virtual con la iso previamente descargada le damos la partición del disco duro y la de la memoria RAM y abrimos la máquina virtual para configurar el servidor.



Se activa el protocolo openSSH para la conectividad del servidor.

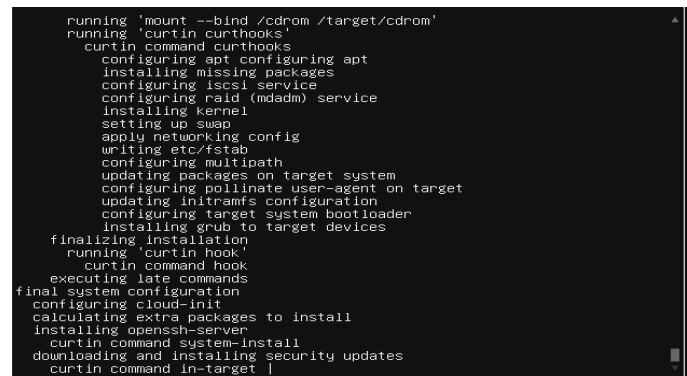
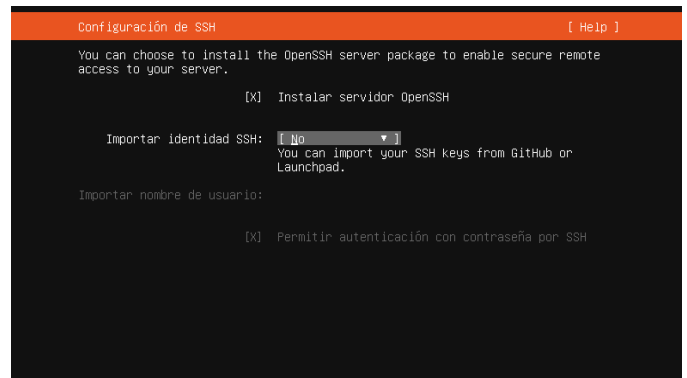


Fig. 2 instalación del SSN

Una vez finalice la actualización de parches de seguridad nos logiamos en el servidor con el usuario y contraseña anteriormente seleccionada.



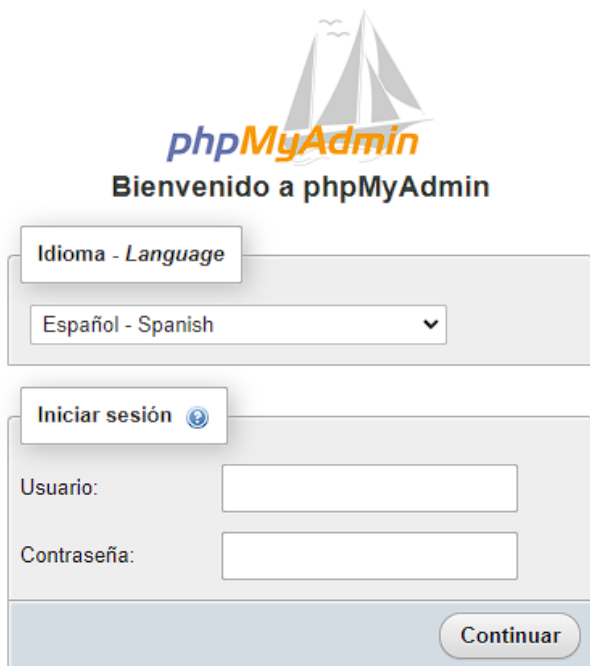


Fig.9 conexión con phpmyadmin

## DIAGRAMAS UML

### Diagrama de clases uml

diagrama de clases uml de la aplicación DemoDogs Stranger Things.

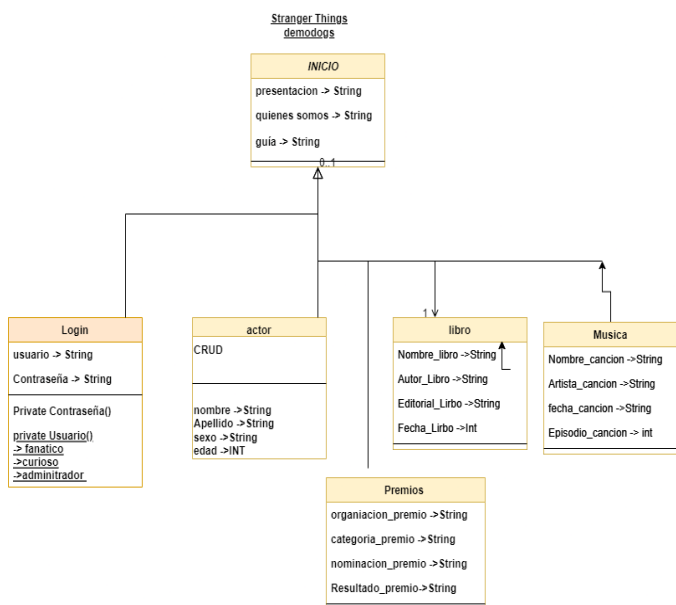


Fig.10 diagrama de clases uml

### Diagrama de secuencia dml

Diagrama de secuencia uml de la aplicación DmoDogs Stranger Things.

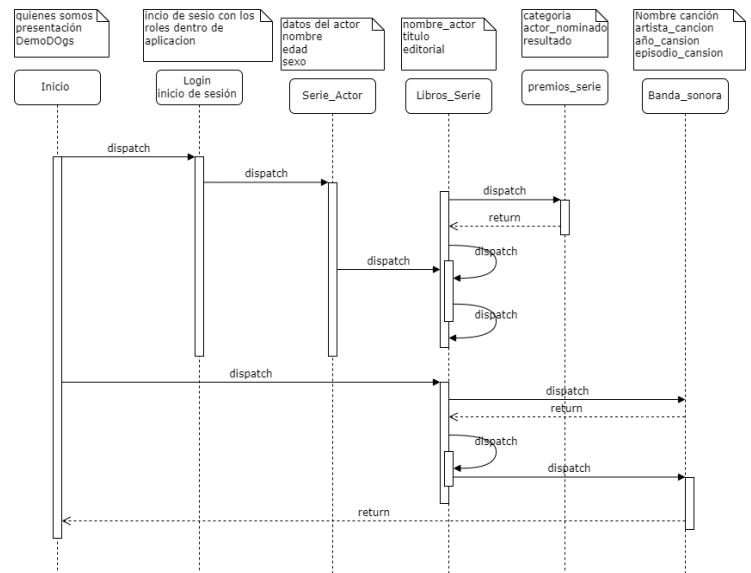


Fig.11 diagrama de secuencias dml

### Diagrama de diccionario de datos

Diagrama de diccionario de datos de la aplicación DmoDogs Stranger Things.

DICCIONARIO DE DATOS			
Campo	Tipo de Datos	Tamaño	Descripción
nombre_actor()	varchar()	45	tipo texto
edad_actor()	INT	30	tipo numerico
personaje_actor()	varchar()	45	tipo texto
temporada_actor()	INT	25	tipo texto
banda_sonora()	varchar()	45	tipo texto
banda_artista()	varchar()	45	tipo texto
banda_episodio()	INT	35	tipo numerico
banda_año()	INT	25	tipo numerico
banda_adicionales()	varchar()	35	tipo texto
premio_actor()	varchar()	20	tipo texto
premio_nominados()	varchar()	26	tipo texto
premio_año()	INT	25	tipo numerico
premio_resultado()	varchar()	30	tipo texto
premio_adicionales()	varchar()	30	tipo texto

Fig.12 diagrama de diccionario de datos uml

### Definición de clases atributos métodos y objetos

Se definió el nombre del proyecto DemoDogs ,en donde estructuraremos la aplicación web, con las clases y los atributos , se crearon paquetes para almacenar las imágenes y la conexión a la base de datos , los estilos de las clases , y por último la lógica de aplicación .

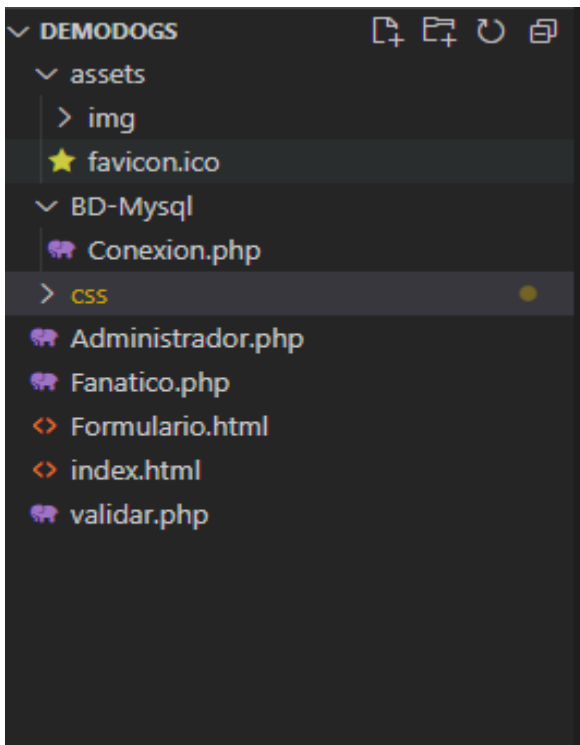


Fig.13 clases en Visual studio code

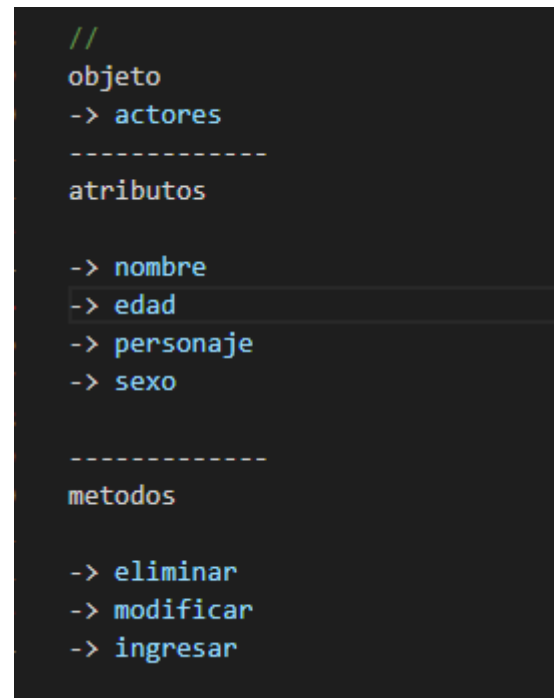


Fig.14 objeto atributos metodos

En la conexión heredamos una variable y clase creada para poder conectarnos con la validación del formulario de ingreso de sesión.

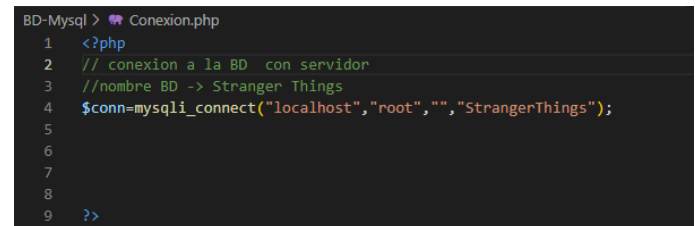


Fig.15 herencia de la clase conexión

## POO en DemoDogs definición

Se definió atributos clases, y métodos para el correcto desarrollo del aplicativo, se usó la herencia entre clases para poder usar las clases, eliminar, ingresar, borrar, modificar, creando objetos como actores, premios, música, libros.

En la clase validar, traemos o heredamos \$conn para poder traer los datos de la base de datos Stranger Things.

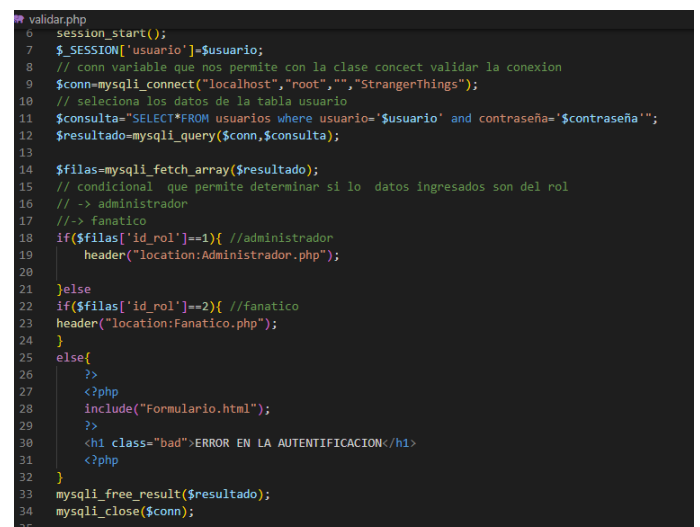


Fig.16 herencia de la clase valida.php

Se uso la programacion orientada a objetos , para poder heredar , calases y metodos , los objeto creados fueron actor, premios  
Bandas sonoras, para poder crear atributos y metodos de estos objeto en el lenguaje php.

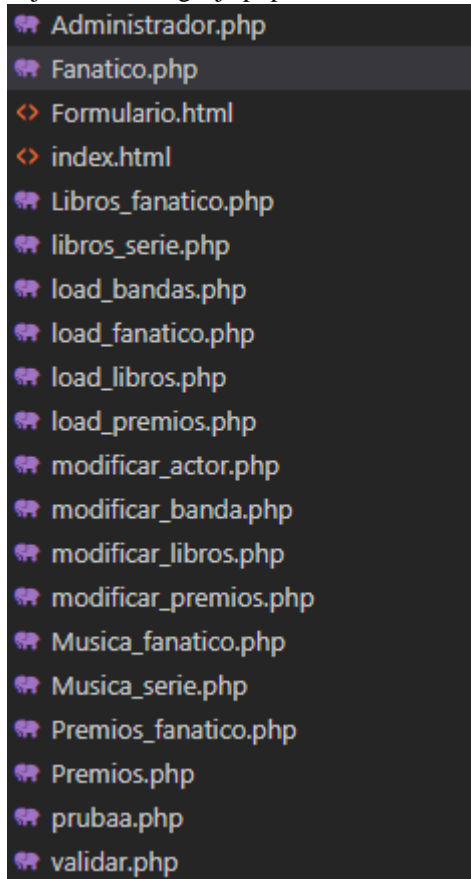


Fig.17 objetos

en la clase conexión en la carpeta **BD.Mysql**

Se estableció conexión a todas las clases y objetos que lo requieran para poder traer datos de la base de datos **StrangerThings**, se usó herencia de POO para su ejecución.

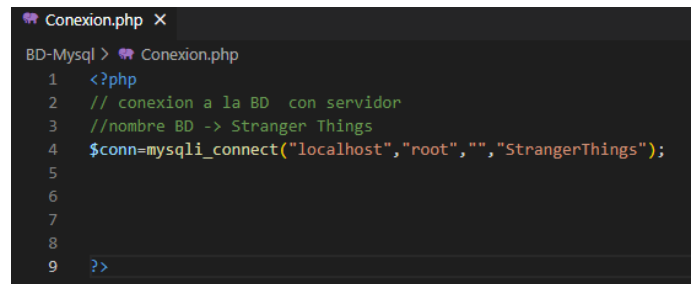
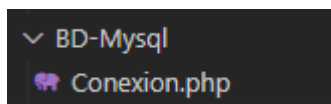


Fig.18 Conexión DB

Ejemplo de herencias en las clases , en la imagen observamos como en premio heredamos la clase \$conn para poder usar dicha informacion.

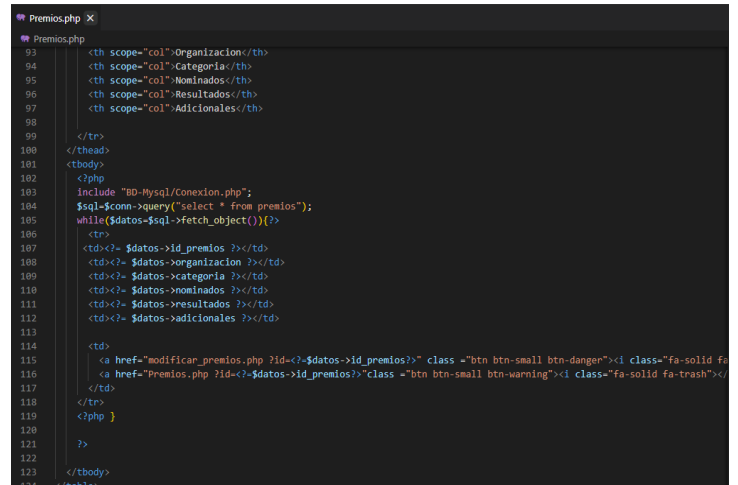


Fig.19 Herencia de Conexión a la clase Premios

En la clase modificar es usada para poder modificar y adaptar el registro de modificación del área administrativa de demodgs esta clase se hereda del objeto controlador.

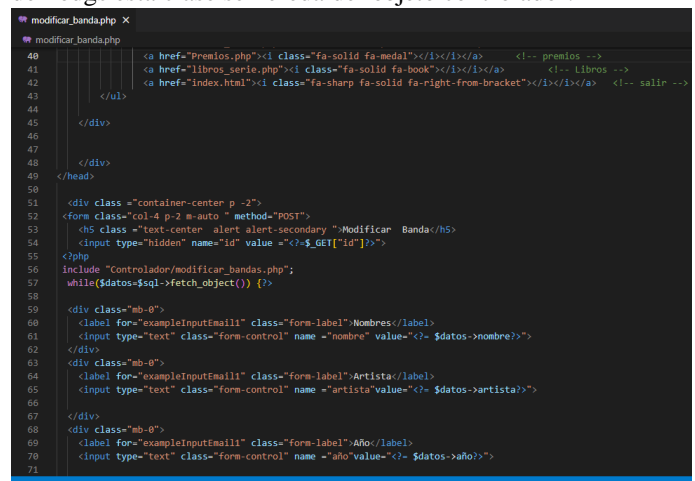


Fig.20 Controlador/modificar clases

En paquete controlador encontramos las clases con objetos , son usadas para las funciones de modificar, eliminar, y cargar tablas validando cada registro.



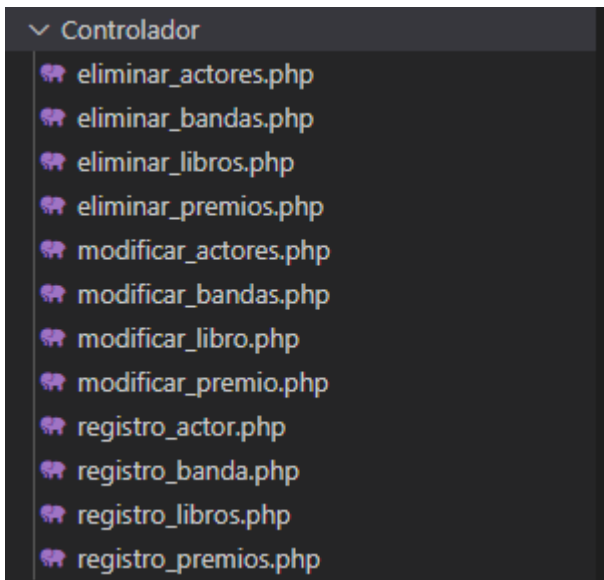


Fig.21 paquete controlador

La clase load\_? Nos permite en el área de faantico consultar la información agregada en el área administrativo dicho proceso es ejecutado con la herencia y polifromismo de la base de datos con la sentencia **JAX** y **QUERY**.

```

1 load_faantico.php
2 <?php
3
4
5
6
7 require 'BD-Mysql/Conexion.php';
8
9 /* Un arreglo de las columnas a mostrar en la tabla */
10 $columns = ['nombre','edad','personaje','sexo','rol','temporada'];
11 $table="actores";
12
13
14
15 $campo = isset($_POST['campo']) ? $conn->real_escape_string($_POST['campo']) : null;
16
17
18 /* Filtrado */
19 $where = '';
20
21 if ($campo != null) {
22     $where = "WHERE (";
23
24     $cont = count($columns);
25     for ($i = 0; $i < $cont; $i++) {
26         $where .= $columns[$i] . " LIKE '%" . $campo . "%' OR ";
27     }
28     $where = substr_replace($where, "", -3);
29     $where .= ")";
30 }
31

```

Fig.22 clase load\_faantico

En la clase principal heredamos el query de la clase load para poder consultar la información.

```

77 /* Llamando a la función getData() */
78 getData()
79 /* Escuchar un evento keyup en el campo de entrada y luego llamar a la función getData. */
80 document.getElementById("campo").addEventListener("keyup", getData)
81 /* Petición Ajax */
82 function getData() {
83     let input = document.getElementById("campo").value
84     let content = document.getElementById("content")
85     let url = "load_premios.php"
86     let formData = new FormData()
87     formData.append("campo", input)
88     fetch(url, {
89         method: "POST",
90         body: formData
91     }).then(response => response.json())
92     .then(data => {
93         content.innerHTML = data
94     }).catch(err => console.log(err))
95 }
96 </script>
97 <!-- Bootstrap core JS -->
98 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="...">
99 </script>
100

```

Fig.23 clase premios\_faantico-con la consulta

en la clase Controlador encontramos la opción de eliminar y modificar, estos objetos se crearon para el uso del rol administrador, solo tendrá acceso el administrador, dicha función posee **JAX** y tipo **CRUD**, se crearon consultas tipo **update**, **from**, **Select**, la función hereda los objetos de conexión para traer datos de consulta en la interfaz.

```

Controlador > eliminar_libros.php
1 <?php
2
3 if (!empty($_GET["id"])) {
4     $id=$_GET["id"];
5
6     $Sql=$conn->query("delete from libros where id_libros=$id");
7
8     if ($Sql==1){
9         echo '<div class="alert alert-success"> LIBROS ELIMINADO </div>';
10
11     }else{
12
13         echo '<div>ERROR </div>';
14
15     }
16 }
17 }
18 }
19
20
21 ?>

```

Fig.24 controlador -eliminar

En la clase validar.php, nos permite conectar la base de datos StrangerThings con las tablas usuarios y rol de usuarios, en este punto se generó un rol para el área de administración y los fanáticos de la serie.

```

1 Validar.php
2 <?php
3 // nos permite evaluar en la base de datos que los datos ingresados en el formulario
4 // son correctos dependiendo el rol
5 $usuario=$_POST['usuario'];
6 $contraseña=$_POST['contraseña'];
7 session_start();
8 $_SESSION['usuario']=$usuario;
9 // con variable que nos permite con la clase conect validar la conexión
10 $conn=mysqli_connect("localhost","root","","StrangerThings");
11 // selecciona los datos de la tabla usuario
12 $consulta="SELECT*FROM usuarios where usuario='$usuario' and contraseña='$contraseña'";
13 $resultado=mysqli_query($conn,$consulta);
14
15 $filas=mysqli_fetch_array($resultado);
16 // condicional que permite determinar si los datos ingresados son del rol
17 // -> administrador
18 if($filas['id_rol']==1){ //administrador
19     header("location:Administrador.php");
20 }
21 }else{
22     if($filas['id_rol']==2){ //fanatico
23         header("location:Fanatico.php");
24     }
25 }else{
26     ?>
27 <?php
28     include("Formulario.html");
29     ?>
30 <h1 class="bad">ERROR EN LA AUTENTICACION</h1>
31 <?php
32 }
33 mysqli_free_result($resultado);
34 mysqli_close($conn);

```

Fig.25 clase validar roles de usuario

Sentencia if else para validar el rol de usuario si la fila seleccionada es 1 sera administrador y si else if es igual a 2 sera fanatico , en la imagen podemos observar como con header(); vinculamos la clase que deseamos abrir .

**Rol\_usuario = 1→administrador and 2→fanatico**

```
if($filas['id_rol']==1){ //administrador
    header("location:Administrador.php");
}else
if($filas['id_rol']==2){ //fanatico
    header("location:Fanatico.php");
}
else{
    ?>
```

Fig.26 roles de usuario con setencia if else

## Base de datos Stranger Things MySQL phpmyamin

Se creo una base de datos en MySQL con phpmyadmin en donde se crearon los tabla para le area de login , usuario y roles de usuario , tambien de desarrollaron cada tabla de actor , premios, bandas sonoras, libros.



Fig. 27 BD Stranger Things

Imagen de las tablas de la base de datos

id_actor	nombre	edad	personaje	sexo	rol	temporada
1	Millie Bobby Brown	18	once	femenino	protagonista	2
2	Finn Wolfhard	19	Mike Wheeler	Masculino	novio de once	3

Fig.28 actores

id_banda	nombre	artista	año	episodio	adicionales
1	master of puppets	Metallica	1981	temporada 2 cap-5	James Hetfield
2	Fear of the Dark	Iron Maiden	1992	temporada 1-cap 2	Heavy metal

Fig.29 bandas

id	nombre	usuario	contraseña	id_rol
1	juan esteban gomez	juanpoli	1233	1
2	alejandra suarez	fanatico1	1233	2

Fig. 30 usuarios por rol

## Estilos CSS Demodogs StrangerThings

Se genero el paquete css con su respectiva clase asociada al index, y a los formularios e inicio de seccion de la aplicacion basandonos en un diseño moderno.

```

CSS
# administrador.css
# cabecera.css
# fanatico.css
# login.css
# styles.css
```

Fig.31 estilos css

Para concetar el estilo css en el index y el area administrativa y fanatica , se conceto con REAL, para poder dar forma grafica a la interfaz.

```

<!DOCTYPE html>
<!--formulario de ingreso de sesion por roles-->
<!--rol -> Administrador-->
<!--rol-> Fanatico -->
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>login</title>
    <!--estilos ccs con la imagen del icono -->
    <link rel="icon" type="image/x-icon" href="assets/favicon.ico" />
    <link rel="stylesheet" href="css/login.css">
    <link rel="stylesheet" href="css/cabecera.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.0.0/animate.min.css">
</head>
<body>
    <!--la imagen del fondo se referencia en los estilos de login / css/login.css-->
```



Fig.32 conexión de los estilos

```

css > # styles.css > ...
72 }
73
74 body {
75   margin: 0;
76   font-family: var(--bs-body-font-family);
77   font-size: var(--bs-body-font-size);
78   font-weight: var(--bs-body-font-weight);
79   line-height: var(--bs-body-line-height);
80   color: var(--bs-body-color);
81   text-align: var(--bs-body-text-align);
82   background-color: var(--bs-body-bg);
83   -webkit-text-size-adjust: 100%;
84   -webkit-tap-highlight-color: rgba(0, 0, 0, 0);
85 }
86
87 hr {
88   margin: 1rem 0;
89   color: red;
90   background-color: red;
91   border: 0;
92   opacity: 0.25;
93 }
94
95 hr:not([size]) {
96   height: 1px;
97 }
98
99
100 h6, .h6, h5, .h5, h4, .h4, h3, .h3, h2, .h2, h1, .h1 {

```

Fig.33 styles css

Imágenes usadas en el proyecto de demodogs ,cada imagen tiene su respectivo nombre para poderse vincular al proyecto y a los estilos.



Fig.34 imágenes del proyecto

En el paquete assets tiene una sub paquete llamado img en este img se encontrara todas las imágenes usadas en el proyecto en formato JPG Y PNG.

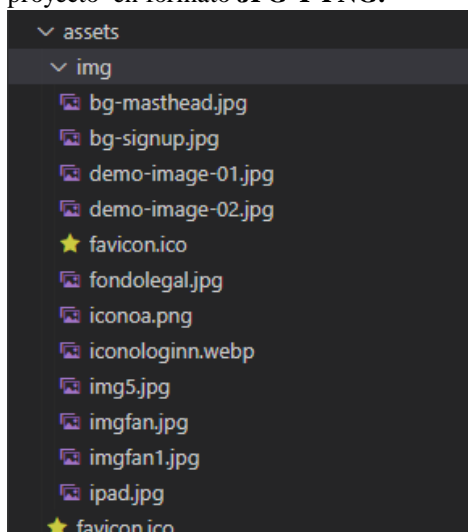


Fig.35 clases de imágenes

## Ventanas graficas de demodogs

Se creo el index con esrucutra html y php , en la clase se encuentra un menu vertical con la opcion de historio , ingresar , contacto , tembine encontramos 3 contenedro con imágenes , en donde se describe los actores , y los moustruos de la serie , el footer encontramos el area de contacto de la pagina .

```

index.html > ...
14 <script src="https://use.fontawesome.com/releases/v6.1.0/js/all.js" crossorigin="anonymous"></script>
15
16 <link href="https://fonts.googleapis.com/css?family=VarelaRound" rel="stylesheet" />
17 <link href="https://fonts.googleapis.com/css?family=Munito:200,200i,300,300i,400,400i,600,600i,700,700i,800,800i,900,900i" rel="stylesheet" />
18 <!--referencia de estilos en la carpeta css-->
19 <link href="css/styles.css" rel="stylesheet" />
20 </head>
21 <body id="page-top">
22
23   <nav class="navbar navbar-expand-lg navbar-light fixed-top" id="mainNav">
24     <div class="container px-4 px-lg-5">
25       <a class="navbar-brand" href="#page-top">DemoDogs</a>
26       <button class="navbar-toggler navbar-toggler-right" type="button" data-bs-toggle="collapse" data-bs-target="#navbarResponsive"
27         <i class="fas fa-bars"></i>
28     </button>
29     <!-- menu con la pociones historia , ingresar , contacto -->
30     <div class="collapse navbar-collapse" id="navbarResponsive">
31       <ul class="navbar-nav ms-auto">
32         <li class="nav-item"><a class="nav-link" href="#about">Historia </a></li>
33         <li class="nav-item"><a class="nav-link" href="Formulario.html">Ingresar</a></li>
34         <li class="nav-item"><a class="nav-link" href="#signup">Contacto</a></li>
35       </ul>
36     </div>
37   </nav>
38
39   <!--contenedor con principal con el nombre del proyecto y la imagen masthead-->
40
41   <header class="masthead">
42     <div class="container px-4 px-lg-5 d-flex d-flex align-items-center justify-content-center">
43       <div class="d-flex justify-content-center">
44         <div class="text-center">
45           <h1 class="mx-auto my-0 text-uppercase">Stranger Things</h1>
46         </div>
47       </div>
48     </div>
49
50   </div>
51
52   </div>
53

```

Fig.36 clase index.html

Parte grafica del proyecto index ya publicado en el servidor



Fig.37 index inicio



Fig.38 index elenco descripcion general

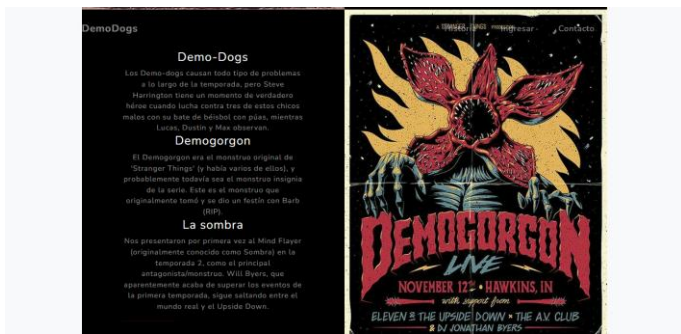


Fig.39 index descripcion de moustruos

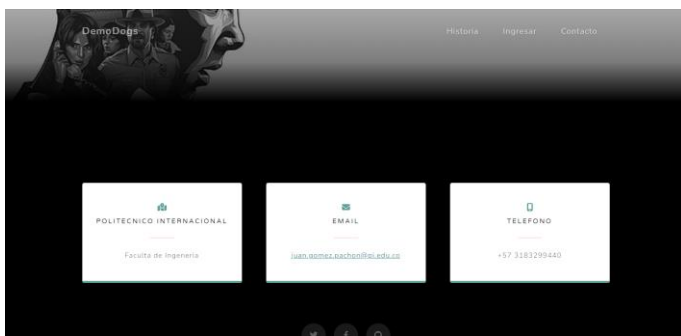


Fig.40 index footer con los contactos

En el apartado de login encontramos una imagen alusiva a la serie de Strangerthings , facil de ingresar los datos .

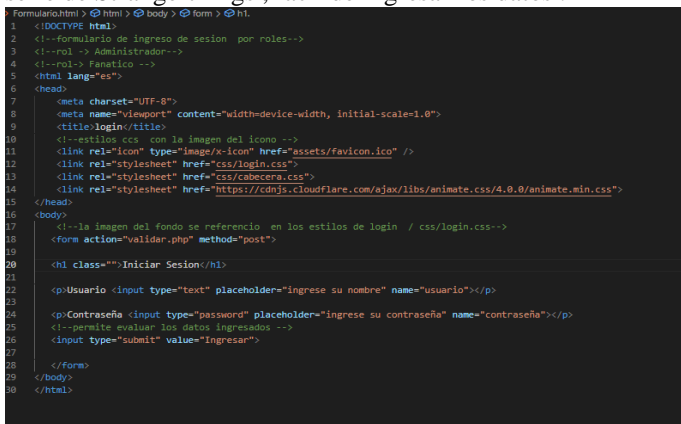


Fig.41 login

Parte grafica del proyecto login ya publicado en el servidor .

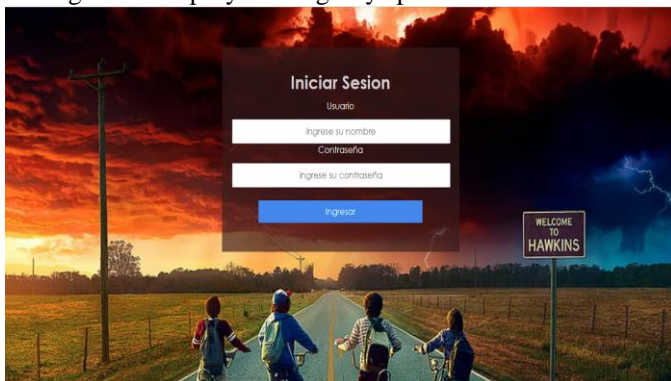


Fig.42 login interfaz de ingreso

Al ingresar al area administrativo nos no encontraremos con una interfaz totalmente diferente , tendremos un menu vertical con botones integrados que nos enviarian a las clases actores, bandas, premios, libros.

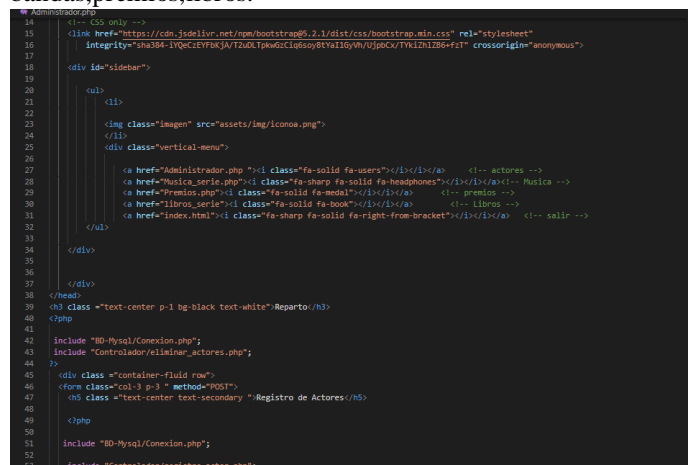


Fig.43 clase administrador

Parte grafica del proyecto Administrador ya publicado en el servidor .

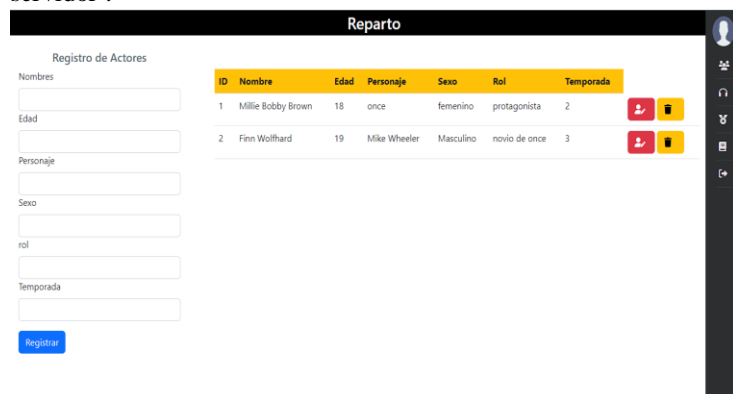


Fig.44 administrador interfaz

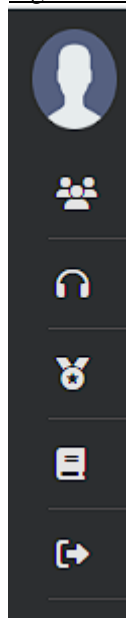


Fig.45 menu vertical de adminitrador

Se realizo una formulario con los datos a ingresar a la base de datos en este caso fueron el nombre del actor , temporada , personaje edad .



**Registro de Actores**

Nombres

Edad

Personaje

Sexo

rol

Temporada

**Registrar**

Fig.46 formulario de registro de actores

Por otro lado se realizo un crud con una tabla , en donde el encabezado es de color amarillo contine dos botones integrados eliminar y modificar .

```

<div class="col-9 p-5">
  <table class="table">
    <thead class="bg-warning">
      <tr>
        <th scope="col">ID</th>
        <th scope="col">Nombre</th>
        <th scope="col">Edad</th>
        <th scope="col">Personaje</th>
        <th scope="col">Sexo</th>
        <th scope="col">Rol</th>
        <th scope="col">Temporada</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td><= $datos->id_actor ></td>
        <td><= $datos->nombre ></td>
        <td><= $datos->edad ></td>
        <td><= $datos->personaje ></td>
        <td><= $datos->sexo ></td>
        <td><= $datos->rol ></td>
        <td><= $datos->temporada ></td>
        <td>
          <a href="modificar_actor.php?id=<= $datos->id_actor"> class="btn btn-small btn-danger">i class="fa-solid fa-user-pen"></i></a>
          <a href="eliminar_actor.php?id=<= $datos->id_actor"> class="btn btn-small btn-warning">i class="fa-solid fa-trash"></i></a>
        </td>
      </tr>
    </tbody>
  </table>
</div>

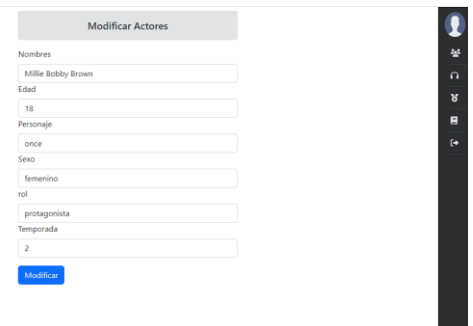
```

Fig.47 codigo de la tabla crud

Reparto						
ID	Nombre	Edad	Personaje	Sexo	Rol	Temporada
1	Millie Bobby Brown	18	once	femenino	protagonista	2
2	Finn Wolfhard	19	Mike Wheeler	Masculino	novio de once	3

Fig.48 tabla reparto

Al precionar el boton de modificar , no direccionara a un formulario en donde podremos modificar los datos de la tabla , por ende se actualizara los datos en la interfaz y en la base de datos.



**Modificar Actores**

Nombres

Edad

Personaje

Sexo

rol

Temporada

**Modificar**

Fig.49 formulario de modificar datos

En el area de fanatico se creo una interfaz con un imagen de fondo con la serie , aquí encontraremos la informacion referente a la seccion que deseemos buscar ya sea , actores , libros,musica,premios.



**Elenco de Stranger Things**

Nombre	Edad	Personaje	Sexo	Rol	Temporada
Millie Bobby Brown	18	once	femenino	protagonista	2
Finn Wolfhard	19	Mike Wheeler	Masculino	novio de once	3

Fig.50 fanatico búsqueda de actores

En cualquiera de las dos interfaces ya sea administartivo o fanatico , tendran la opcion de volver a la pagina inicial de la aplicacion.

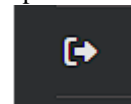




Fig.51 boton para salir

## Conclusión del proyecto demodogs

Se concluyo en el desarrollo del aplicativo web de la serie StrangerThings, en donde se uso una arquitectura de 3 capas, usando un servidor linux compilado en una maquina virtual llamada VirtualBox, con ello logramos descargar los paquetes de apache los servicios de MySQL, instalando el entorno adecuado para el funcionamiento de la aplicación web, desarrollamos la parte del backend y frontend de la aplicación demodogs, con los paradigmas de la programación orientada a objetos dando énfasis en el lenguaje PHP HTML CSS y JS, nos enfocamos en lograr diagramas, diccionario de datos, diseñando cada parte de la aplicación para lograr un producto funcional y viable para los fanáticos de StrangerThings, con un plus adicional como es el área administrativo, con el se busco crear una ventana limpia y legible para poder modificar datos dentro de la aplicación, usando un sistema de CRUD con la herencia el poliformismo los objetos y clases de la programación orientada a objetos, dentro del proceso de este proyecto concluimos con una enseñanza rigurosa sobre el sistema JAX para traer datos con un buscador agregado a la aplicación, también se implemento buenas practicas sobre la seguridad informática, creando un sistema de inicio de sesión por roles, administrativo y fanático, logrando un buen performance a la hora del usuario manejar muestra aplicaicon, en el área de diseño se implementaron formas y colores por contenedores para así poder darle vida a nuestra aplicación, un proyecto de StrangerThings totalmente funcional con una apariencia agradable al usuario final que son los fanáticos de este serie.

## BIBLIOGRAFÍA

- Bootstrap. (7 de 08 de 2012). *Bootstrap*. Obtenido de Bootstrap: <https://getbootstrap.com/>
- fontawesome. (s.f.). *fontawesome*. Obtenido de fontawesome: <https://fontawesome.com/>
- Manual, P. (2014). *PHP Lengunce*. madrid: programacion es saber.