

DOCUMENTACIÓN Y GUÍA DE USO

Grado en Ingeniería Informática de Gestión y Sistemas de Información

Entrega 1-Sistema Web

SISTEMAS DE GESTIÓN DE SEGURIDAD
DE
SISTEMAS DE INFORMACIÓN



BILBOKO
INGENIARITZA
ESKOLA
ESCUELA
DE INGENIERÍA
DE BILBAO

MIKEL GOTI
10/10/2022

Documentación del sistema web

index.php

Una vez hemos entrado a la pagina web buscando localhost:81 en el navegador. Lo que nos muestra es el contenido que hay dentro del directorio app. Por defecto el navegador siempre carga el archivo con el nombre de index.php.

A continuación el usuario tendrá dos opciones iniciar sesión o registrarse. Por defecto el archivo database.sql tiene un usuario previamente registrado que es el USUARIO:admin CONTRASEÑA:12345.



Lo primero es crearse un usuario para eso hay que ir a registrarse. Al clicar en registrarse nos dirigira a la siguiente pagina.

A screenshot of a web page with a blue header containing the text "COMIC CENTRAL" in a large, bold, black font, and "REGISTRATE" in a smaller, bold, black font below it. The main content area has a grey background and contains a registration form. The form has two columns of input fields. The left column contains fields for "Usuario" (with placeholder "mikevruasozki"), "Nombre" (with placeholder "John"), "Segundo Apellido" (with placeholder "Goikoetxea"), "Fecha" (with placeholder "aaaa-mm-dd"), and "Contraseña". The right column contains fields for "Email" (with placeholder "JohnTheReaper@servidor.extension"), "Apellido" (with placeholder "TheReaper"), "Teléfono" (with placeholder "4491234567"), "Dni" (with placeholder "12345678-A"), and "Repetir Contraseña". At the bottom of the form, there are two black buttons with white text: "REGISTRARSE" on the left and "VOLVER" on the right.

Registrarse.php y scriptvalidacion.js

Para registrarse es necesario ir rellenando todos los campos. La validación se hará mediante código javascript. Cada vez que el usuario haga clic o escriba una tecla en cualquier campo input este se validara automáticamente mediante expresiones regulare y listeners. La clase encargada de la validación se encuentra en el directorio app/js/scriptvalidacion.js.

Los listeners que registran cada campo tiene un id asignado del documento html.

```
19  /**
20   * VARIABLES
21   */
22  const r = document.getElementById('campo');
23  const f = document.querySelectorAll('#campo input');
24  const boton = document.getElementById('btn_regis');
25
26  /**
27   * LISTENERS
28   * CADA VEZ QUE SE CLICKE O SE APRIETE UNA TECLA REGISTRA LLAMANDO
29   * A LA FUNCION campo()
30   */
31  f.forEach( i =>
32  {
33      i.addEventListener('keyup', campo);
34      i.addEventListener('blur', campo);
35  })
```

El id campo se asigna con la función del documento querySelectorAll() para seleccionar todos. A esa función se le pasa un parámetro y son el id que es el campo y el input que son todos los inputs donde el usuario tiene que rellenar sus datos. Cada vez que el usuario escriba mediante el método forEach se analizan los listeners y se llama a la función campo. Es importante aclarar que este proceso se ejecuta cada vez que se escriba un carácter o se haga clic.

La función campo no es mas que un filtro del campo en el que se esta ingresando el texto para llamar a la función validar(), esto se hace mediante un switch.

```
37  /**
38   * FUNCION PARA VALIDAR EL INGRESO DE LAS LETRAS QUE SE VAYAN TECLEANDO CON EL OBJETIVO DE VALIDAR
39   * @param {Input del usuario en el campo de texto de la pagina web} e
40   */
41  function campo(e)
42  {
43      console.log(e.target.name)
44      switch(e.target.name){
45          case "usuario":
46              validar(expresiones.usuario,e.target,'usuario');
47              break;
48          case "email":
49              validar(expresiones.email,e.target,'email')
50              break;
51          case "nombre":
52              validar(expresiones.nombre,e.target,'nombre')
53              break;
54          case "apellido":
55              validar(expresiones.apellido,e.target,'apellido')
56              break;
57          case "apellido1":
58              validar(expresiones.apellido1,e.target,'apellido1')
59              break;
60          case "telefono":
61              validar(expresiones.telefono,e.target,'telefono')
62              break;
63      }
```

La función validar() recibe 3 parámetros. El primero es la lista de las expresiones regulares para que se pueda realizar una comparación con algo, el segundo parámetro es el propio texto que se va ingresando en el campo por el usuario y el último es el campo en cuestión.

Para comprender como valida la función validar es necesario entender las expresiones regulares. Son unas constantes predefinidas que tienen a cada campo asignada una expresión que limita lo que se puede o no escribir. Son todas muy simples, por ejemplo la línea 8 usuario solamente se pueden escribir minúsculas[a-z], mayúsculas[A-Z] y dígitos[0-9] el parámetro 1,40 indica que se pueden escribir desde 1 carácter hasta un máximo de 40. Todo ello esta dentro de ^\$ esto quiere decir que la expresión se evalúa desde el principio hasta el final toda junta. La única expresión un poco enrevesada es la del email.

```

7  const expresiones = {
8      usuario: /^[a-zA-Z0-9]{1,40}$/,**ER= minusculas mayusculas digitos*/
9      email: /^[a-zA-Z0-9_]+([.][a-zA-Z0-9_]+)*@[a-zA-Z0-9_]+([.][a-zA-Z0-9_]+)*[.][a-zA-Z]{2,5}$/,**
10     nombre: /^[a-zA-Z]{1,40}$/,**ER= minusculas mayusculas*/
11     apellido: /^[a-zA-Z]{1,16}$/,**ER= minusculas mayusculas*/
12     apellido1: /^[a-zA-Z]{1,16}$/,**ER= minusculas mayusculas*/
13     telefono: /^[0-9]{9}$/,**ER= digitos rengo(9) */
14     fecha: /^[0-9]{4}-[0-9]{2}-[0-9]{2}$/,**
15     dni: /^[0-9]{8}-[a-zA-Z]{1}$/,**
16     password: /^[a-zA-Z0-9]{5,20}$/,**
17 }

```

Básicamente al comparar la entrada con la expresión si esta bien se ejecuta el if de validar y sino se ejecuta el else. Hay que mencionar que la fecha, el dni y la segunda contraseña tiene una validación extra, aparte de las expresiones regulares.

```

100 const validar = (exp, input, c) => {
101     const param = document.getElementById(`campo_input-error-${c}`);
102     if(exp.test(input.value)){
103         console.log("bien");
104         param.style.display = "none";
105         document.getElementById(`grupo_${c}`).classList.remove('grupo_registrar-mal');
106         document.getElementById(`grupo_${c}`).classList.add('grupo_registrar-bien');
107         if(c === "fecha"){
108             validarFecha(input.value)
109         }
110         else if(c === "dni"){
111             validarLetraDni(input.value)
112         }
113         else if(c === "password1"){
114             validarPassword(input);
115         }
116     }
117 }
118
119
120

```

Antes de nada en validar se asigna una constante para hacer o desaparecer el mensaje. Si se ejecuta el if quiere decir que lo ingresado esta dentro de los parámetros de cumplimiento de las expresiones. Una vez en el if se hace desaparecer al mensaje del campo, se quita el css del grupo registro-mal (lo rojo) en la línea 107 y se agregar el css grupo registro-bien(verde).

Formulario de usuario con el campo "Usuario" que contiene el texto "mikel". El campo tiene un borde verde y un icono de checkmark verde a la derecha, indicando que la validación ha pasado exitosamente.

Si en vez de cumplir con la expresión regular no es así se ejecuta el else el cual realizar lo contrario. Hacer aparecer al mensaje, remove el color verde y poner el rojo.

Formulario de usuario con el campo "Usuario" que contiene el texto "mikewłasozki". El campo tiene un borde rojo y un icono de X roja a la derecha, indicando que la validación ha fallado. Debajo del campo, se muestra el mensaje de error: "El usuario solo puede contener letras y numeros."

El dni tiene que cumplir con el formato especificado de 123456789-Z y la letra tiene que coincidir con los dígitos para ello utilizaremos una función aparte de las expresiones. Función validarni(). El método declara el array que contiene las letras en orden para el dni. Se guarda la ultima letra en la variable letra, luego se obtienen los dígitos sin la letra y el guión del formato. Y por ultimo se obtiene la letra correspondiente con el resto de dividir los dígitos entre 23.

```

295  /**
296   * SI EL FORMATO DEL DNI INTRODUCIDO ESTA FUNCION ES LLAMADA
297   * calcula si la letra al final del dni se corresponde con el dni.
298   * para ello es necesario el dni, solo el numero limpio para poder obtener el resto
299   * al dividirlo entre 23. Para ello se quitan la letra y el guion.
300   * @param {*Recibe el dni completo con formato 11111111-A} dni
301   */
302  function validarLetraDni(dni){
303      var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B',
304                  'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E', 'I'];
305
306      var letra = dni.substr(-1)
307      var dniSinLetra = dni.substr(0,dni.length-1)
308      var dniSinGuion = dniSinLetra.substr(0,dniSinLetra.length-1)
309      var letraCalculada = letras[dniSinGuion % 23]
310
311      if(letraCalculada === letra){
312          document.getElementById('grupo_dni').classList.remove('grupo_registrar-mal');
313          document.getElementById('grupo_dni').classList.add('grupo_registrar-bien');
314      }
315      else{
316          document.getElementById('grupo_dni').classList.remove('grupo_registrar-bien');
317          document.getElementById('grupo_dni').classList.add('grupo_registrar-mal');
318      }
319
320  }

```

Lo interesante del formulario es que si el usuario no introduce todos los datos y los introduce bien el botón para poder registrarse no se desbloquea. Para ello se a definido un método botonDesbloquear que registra mediante variables booleanas si esta bien definido o no cada uno de los campos. Este metodo recibe dos parámetros el campo y el valor booleano. Al introducirse bien un campo se pasa de false a true si el campo vuelve a estar mal se vuelve a poner en false. Esta llamada se hace al validar cada campo.

```

if(usuario === "true" && email === "true" && nombre === "true" && apellido === "true" && apellido1 === "true"
    && telefono === "true" && fecha === "true" && dni === "true" && password === "true" && password1 === "true"){
    boton.disabled = false;
    /*agregar clase para que el boton brille*/
    document.getElementById('btn_registrar').classList.add("boton_desbloqueado");
}
else{
    boton.disabled = true;
}

```

Por eso hay un switch dentro de la función validar para filtrar y llamar a botondesbloquear con el campo actual.

Si todos y cada uno de los campos están bien introducidos sus variables booleanas estarán en true y el botón se desbloqueara con un brillo verde.

Usuario	Email
<input type="text" value="a"/>	<input type="text" value="asd@sdfs.gg"/>
Nombre	Apellido
<input type="text" value="asdas"/>	<input type="text" value="dasdas"/>
Segundo Apellido	Teléfono
<input type="text" value="dasdas"/>	<input type="text" value="123456789"/>
Fecha	Dni
<input type="text" value="1111-22-22"/>	<input type="text" value="12345678-Z"/>
Contraseña	Repetir Contraseña
<input type="password" value="....."/>	<input type="password" value="....."/>
REGISTRARSE	VOLVER

Solo hace falta que alguno de los campos este mal para que el boton de registrar se vuelva a bloquear.

Usuario	Email
<input type="text" value="a"/>	<input type="text" value="asd@sdfs.gg"/>
Nombre	Apellido
<input type="text" value="asdas"/>	<input type="text" value="dasdas"/>
Segundo Apellido	Teléfono
<input type="text" value="dasdas"/>	<input type="text" value="123456789"/>
Fecha	Dni
<input type="text" value="1111-22-22"/>	<input type="text" value="12345678"/>
	<small>El dni teien que tener el formato 12345678-Z. La letra deberá conincidir con el numero.</small>
Contraseña	Repetir Contraseña
<input type="password" value="....."/>	<input type="password" value="....."/>
REGISTRARSE	VOLVER

Al pulsar el botón de registrar se envían todos los campos mediante el método post a otra clase, la clase controlador-registro.php. Esta clase una vez recibe los datos realizar una consulta a la base de datos para comprobar si ya hay un usuario registrado con el mismo nombre si no es así inserta los datos en una nueva fila. Pero si ya hay un nuevo usuario no inserta los datos y muestra un mensaje de error indicando lo sucedido.

```

12     $fecha = $_POST['fecha'];
13     $dni = $_POST['dni'];
14     $password = $_POST['password'];
15
16
17
18     $res = $con->query("SELECT EXISTS (SELECT * FROM iniciados WHERE usuario='$usuario');");
19     $row = mysqli_fetch_row($res);
20
21     if($row[0] == 0){
22         // $h = encriptarPass($password);
23         // almacenar el hash generado de la password en la db
24         $con->query("INSERT INTO iniciados(usuario,email,nombre,apellido,apellidol,telefono,fecha,dni,password)
25         VALUES('$usuario','$email','$nombre','$apellido','$apellidol','$telefono','$fecha','$dni','$password')");
26         $mensaje_correcto = "Usuario agregado. Ya te has registrado! Vuelve para iniciar sesion.";
27         include_once("pagina_registrarse.php");
28     }
29     else{
30         $mensaje_error = "El usuario ya existe prueba con otro.";
31         include_once("pagina_registrarse.php");
32     }
33

```

iniciarsesion.php, controlsesion.php y usuario.php

Una vez completado el registro de un nuevo usuario ya se podrá iniciar sesión. Para ello solamente habrá que ir a la pagina de inicio sesión e introducir los datos de usuario y contraseña previamente creados.

The image shows a login form with a blue header containing the text "INICIA SESION" in a bold, italicized font. Below the header, on a light gray background, are two white input fields. The first field is labeled "Usuario" and the second is labeled "Contraseña". Below these fields are two buttons: "INICIAR SESION" and "VOLVER".

Al presionar el botón de iniciar sesión lo que ocurrirá es que se llamara con los datos introducidos a la clase controladorInicioSesion.php la cual se encarga de iniciar correctamente un nuevo usuario con sus datos y su sesión. Para ello esta clase comprueba si ya hay una sesión iniciada. Si no es

así comprueba si el usuario y la contraseña introducidas son correctas llamando la función validarLogin() de la clase usuario. Esta función comprueba la veracidad de los datos introducidos con una consulta a la base de datos. Si los datos son correctos el usuario introducido se establece como actual mediante otra función esta vez de la clase controlSesion.php que lo establece como usuario de la sesión. Después se recoge toda la información pertinente del usuario para iniciar sesión en la página web con ella. Esto es posible por un método setInfo() en la clase usuario que tiene un foreach.

```

3 include_once("usuario.php");
4 include_once("ControlSesion.php");
5 include_once("actualizarTabla.php");
6
7 $usuarioSesion = new ControlSesion();
8 $usuario = new Usuario();
9
10 if(isset($_SESSION['usuario'])){
11     /**
12      * PROBLEMA POR RESOLVER
13      * NO FUNCIONA BIEN A LA HORA DE SABER SI HAY O NO SESION
14      */
15     $usuario->setInfo($usuarioSesion->getUsuarioActual());
16     include_once("pagina_principal.php");
17 }
18 else if(isset($_POST['usuario']) && isset($_POST['password'])){
19
20     $u = $_POST['usuario'];
21     $p = $_POST['password'];
22
23     if($usuario->validarLogin($u,$p)){
24         $usuarioSesion->setUsuarioActual($u);//PARA ASIGNAR EL USUARIO A LA SESION ACTUAL HASTA QUE
25         $usuario->setInfo($u);//ESTE METODO SE UTILIZA PARA ACCEDER A LA DB Y ASIGNAR EN LA CLASE U
26         include_once("pagina_principal.php");
27     }
28     else{
29         /**
30          * AQUI SE PODRIA AGREGAR UN CONTADOR PARA PONER UN LIMITE A LA HORA DE ACCEDER A LA PAGINA
31          * PONGAMOS 10 veces SI SE SUPERAN LLAMAMOS AL FBI
32          */
33         $mensaje_incorrecto = "El usuario o la contraseña son incorrectos.";
34         include_once("pagina_iniciarsesion.php");
35     }
36 }

```

La clase usuario anteriormente mencionada es la siguiente:

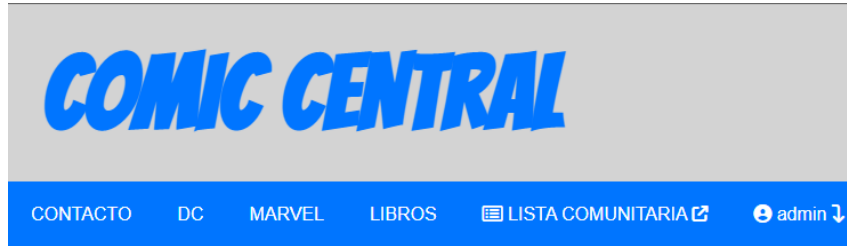
```

2 /**CLASE USUARIO */
3 include("con_db.php");
4
5 You, hace 9 minutos | 1 author (You)
6 class Usuario extends Database{
7
8     private $usuario;
9     private $email;
10    private $nombre;
11    private $apellido;
12    private $apellido1;
13    private $telefono;
14    private $fecha;
15    private $dni;
16    private $password;
17
18    //GETTERS
19    public function getUsuario(){
20        return $this->usuario;
21    }
22
23    public function setUsuario($usuario){
24        $this->usuario = $usuario;
25    }
26
27    public function getEmail(){
28        return $this->email;
29    }
30
31    public function setEmail($email){
32        $this->email = $email;
33    }
34 }

```


pagina-principal.php

Una vez se accede a la pagina web lo primero que destaca es el menú de las diferentes opciones que ofrece la pagina.



Al final del menú se puede observar el nombre de la sesión actual del usuario. Si movemos el cursor encima se desplegará el perfil del usuario actual con todos sus datos, la contraseña está oculta. Para poder ver todo los datos habrá que pulsar en mas informacion para desplegar la segunda pestaña con el perfil y los datos mas personales.

Si en algún momento el usuario querría por el motivo que fuese cambiar el email, tendría que ir al perfil buscar el email y hacer clic sobre el icono con las flechas para abrir los dos campos inputs.

Para poder cambiar exitosamente el email se tiene que introducir el antiguo y después el email nuevo. El email nuevo será validado utilizando el mismo método que al registrarse pero esta vez solamente para un campo.

Si el usuario introduce mal el email antiguo o el nuevo no se efectuará el cambio y tendrá que volver a iniciar sesión. Si el usuario introduce bien los dos campos el email se cambiará y para refrescar los datos de la pagina y la sesión tendrá que volver a iniciar.




Para insertar los datos se llamara a la clase actualizarTabla.php. Si el usuario intenta cambiar su nombre de usuario e introduce un nuevo usuario valido pero ya existente por otro miembro registrado en la pagina. Habría una colisión por eso en caso de cambiar el usuario hay que realizar una consulta sql a la base de datos para ver si hay colisiones.

```

8  switch(true){
9
10 case isset($_POST['btn_u']):
11     $uN= $_POST['usuario'];
12     $uA = $_POST['usuarioAntiguo'];
13     $col = "usuario";
14
15     $res = $con->query("SELECT EXISTS (SELECT * FROM iniciados WHERE usuario='$uN')");
16     $row = mysqli_fetch_row($res);
17
18     if($row[0] == 0){
19         //echo $uN,$uA,$col;
20         update($col,$uA,$uN);
21     }
22     else{
23         echo "El usuario ya existe! Prueba con otro.";
24         ?><a href="pagina_iniciarsesion.php">Vuelve a iniciar sesion e intentalo de nuevo.</a><?php
25     }
26     break;

```

En el menú también existe la opción de la lista comunitaria si se hace clic en ella se abrirá una nueva ventana con la lista comunitaria. Esta lista muestra los valores de la tabla todo definida en la base de datos. Como se puede observar la lista tiene varias opciones. Entre ellas se pueden eliminar e insertar filas. Las filas evidentemente representan comics o libros.

PUBLICACION	FORMATO	LINK DESCARGA	
8-enero-2020	CBR	 Descargar	<div>Editar</div> <div>Eliminar</div>
22 de enero de 2020	CBR	 Descargar	<div>Editar</div> <div>Eliminar</div>
5 de febrero de 2020	CBR	 Descargar	<div>Editar</div> <div>Eliminar</div>

Presionando en eliminar se borrara la fila a la que le corresponda ese botón. Si por otro lado preferimos editar la fila podemos presionar en el botón editar y se abrirá la fila correspondiente encima de la lista debajo de agregar. Una vez abierta podemos editar cualquier dato que queramos y presionando editar los cambios se guardaran.

AGREGAR

nombre comic/manga/libro	autor	editorial	
--------------------------	-------	-----------	--

EDITAR

86-batman	James Tynion IV	DC	acc
-----------	-----------------	----	-----

Para poder agregar algún dato a la lista lo podemos hacer mediante el formulario de agregar siempre disponible encima de la lista. Hay que tener en cuenta que si decidimos agregar un elemento que ya esta en la lista no se permitirá el añadido ya que no se necesitan dos libros o comics iguales con uno basta. Para ello a la hora de agregar algo la clase de borrar.php se encargara de comprobar si existe algo con el mismo nombre en la base de datos antes de añadir nada.

```

1 <?php // $row hace a $id + 1. Compara
2 include_once("con_db.php");
3
4 $obj = new Database();
5 $con = $obj->conectar();
6
7 switch(true){
8
9     case isset($_POST['btn_e']):
10         $i = $_POST['id'];
11         $n = $_POST['nombre'];
12         $a = $_POST['autor'];
13         $e = $_POST['editorial'];
14         $g = $_POST['genero'];
15         $p = $_POST['publicacion'];
16         $d = $_POST['formato'];
17         $id = $_POST['descarga'];
18
19         $res = $con->query("SELECT EXISTS (SELECT * FROM todo WHERE nombre='$n');");
20         $row = mysqli_fetch_row($res);
21
22         if($row[0] == 0){
23             $con->query("UPDATE todo SET nombre='$n',autor='$a',editorial='$e',genero='$g',publicacion='$p',formato='$d',descarga='$id' WHERE id='$i'");
24             echo "fila editada.";
25             include_once("lista_comunitaria.php");
26         }
27         else{
28             echo "La modificacion del articulo coincide con otro en la lista.";
29             include_once("lista_comunitaria.php");
30         }
31     }
32     break;


```

Por ello es conveniente buscar en la lista mediante el filtro arriba a la derecha. La barra de búsqueda nos proporciona una búsqueda rápida de toda la lista.

Filtro de busqueda.

Dentro del input indica como se realiza la búsqueda, que columnas utiliza para filtrar la información. Es conveniente utilizar la barra de búsqueda antes de agregar nada, porque quizás ya exista en la lista.

Si queremos descargar un comic, libro o manga podemos hacerlo presionando en descargar. Para saber que tipo de archivo descargaremos hay que fijarse en la columna donde dice formato.

FORMATO	LINK DESCARGA
CBR	 Descargar

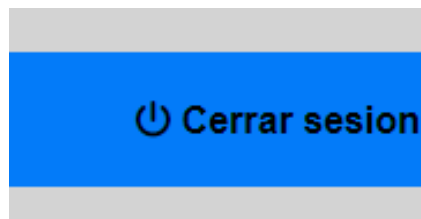
Nota: Si a la hora de agregar o editar la última columna de cada elemento donde dice link. Si introducimos una URL autentica funcionará.

Para tener un funcionamiento rápido y una estructura eficiente cada link del menú proporciona un acceso directo a cada apartado de la pagina web. Si se hace clic en en DC ira a la parte donde esta todo el contenido de DC y los mismo para las demas etiquetas.

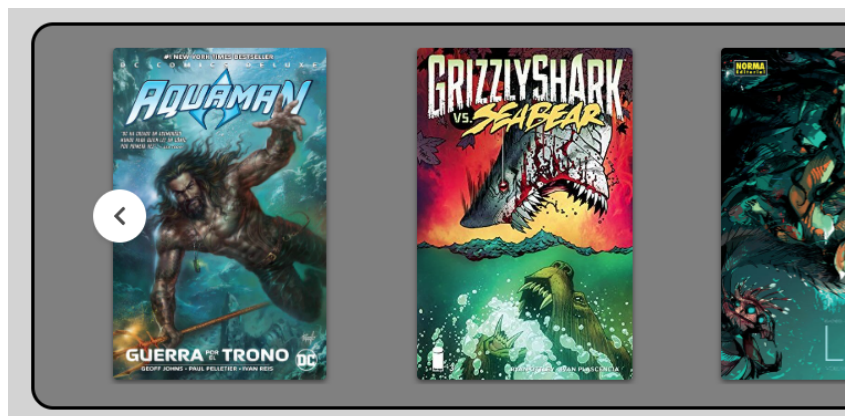


La etiqueta de contacto lleva hasta el footer que se encuentra al final de la pagina web. Por regla general el footer proporciona información de la pagina web, los logotipos, emails, etc..

Para terminar con el menú de la pagina web a la derecha del todo se encuentra el botón de log out o cerrar sesión. Si se hace clic sobre el se sale al menú principal para volver a tener que iniciar sesión.



Para realizar la pasarela de imágenes o el slider me e basado en varios modelos de varias paginas (GeeksForGeeks, wscschool, StackOverflow) hasta conseguir el mas adecuado para mi web. Sobretodo para el código js para poder deslizar las imágenes.



Nota: los links también funcionan.

Otra faceta interesante de la pagine principal es la lista de añadidos recientes. Esta lista muestra los campos que han sido recientemente agregados a la lista comunitaria.

La lista añadidos recientes muestra el contenido de la lista comunitaria pero desde la ultima posición hasta los 18 primeras filas. Esto quiere decir que según se vayan agregando elementos a la lista esta mostrara solo los últimos añadidos en orden. El ultimo agregado a la lista comunitaria será el primero de la lista de añadidos recientes.

LISTA COMUNITARIA AÑADIDOS RECIENTES					
	NOMBRE	AUTOR	EDITORIAL	FORMATO	DESCARGA
1	Green Arrow: Año uno	Andy Diggle	DC	CBR	Descargar
2	All star Superman	Grant Morrison	DC	CBR	Descargar
3	Superman Annual	Alan Moore	DC	CBR	Descargar
4	Superman The man of steel-3	John Byrne	DC	CBR	Descargar
5	Superman The man of steel-2	John Byrne	DC	CBR	Descargar
6	Superman The man of steel-1	John Byrne	DC	CBR	Descargar
7	The witcher-3	Andrzej Sapkowski	Kódansha	PDF	Descargar
8	The witcher-2	Andrzej Sapkowski	Norma	PDF	Descargar

A diferencia de la lista comunitaria esta lista no muestra las columnas de publicación ni del genero ya que son menos relevantes que las otras. De todas formas para ver la información completa de un elemento se puede acceder a la lista comunitaria haciendo clic en el nombre. El propósito de esta lista solamente es informar de las novedades insertadas a la lista. Para ello se utiliza una consulta sql con el limite de 18 columnas y empezando desde la ultima.

```

408
409 {
410     $f = $con-> query("SELECT * FROM todo ORDER BY id DESC LIMIT 18");
411     $cont = 1;
412     while($fil = mysqli_fetch_array($f)){
413         if( $fil['id'] == $fil['id']){
414             $fil['id'] = $cont;
415             $cont ++;
416         }
417     }
418     <tr>
419         <td><?php echo $fil['id'];></td>
420         <td><?php echo $fil['nombre'];></td>
421         <td><?php echo $fil['autor'];></td>
422         <td><?php echo $fil['editorial'];></td>
423         <td><?php echo $fil['formato'];></td>
424         <td><div class="icono_link_descarga"><a href="<?php echo $fil['descarga'];>" target="_blank">
425     </tr>

```

El resto de la web contiene 3 grandes secciones, DC, MARVEL Y LIBROS. Se podrían agregar mas secciones o mejorar las existentes. Solo contiene información e imágenes.

La documentación a sido redactada usando $LaTeX$ y editado a traves de overleaf. El proyecto contiene codigo html,css,js,php y sql. A sido editado mediante VS CODE.