

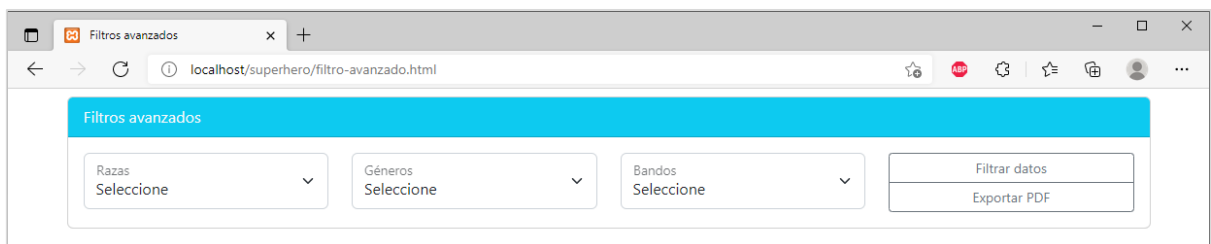
CONSTRUCCIÓN DE REPORTE

FILTROS MÚLTIPLES

1. Construyamos un nuevo archivo de vista **filtro-avanzado.html** y en él la siguiente estructura básica

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>Filtros avanzados</title>
8   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">
9 </head>
10 <body>
11
12   <div class="container">
13     <!-- Fila designada para los controles de filtro -->
14     <div class="row">
15       <div class="col-md-12">
16         <div class="card">
17           <div class="card-header">
18             Filtros avanzados
19           </div>
20           <div class="card-body">
21             <!-- Aquí construye el formulario -->
22           </div>
23         </div>
24       </div>
25     </div>
26
27     <!-- Espacio para la tabla -->
28     <div class="row">
29
30     </div>
31   </div> <!-- Fin container -->
32
33 </body>
34 </html>
```

2. Para la construcción de formularios, haremos una fila divida en 4 secciones de columnas de 3 unidades de ancho. Preste mucha atención a los comentarios. El resultado final debe mostrarse así:



Filtros avanzados

Razas
Selecciona

Géneros
Selecciona

Bandos
Selecciona

Filtrar datos
Exportar PDF

Realice cualquier configuración adicional que permita mejorar la interfaz

El código:

```

<!-- Aquí construye el formulario -->
<form action="" autocomplete="off" id="form-filters">
  <div class="row">
    <div class="col-md-3"> <!-- Primera lista -->
      <div class="form-floating">
        <select name="razas" id="razas" class="form-select">
          <option value="">Seleccione</option>
        </select>
        <label for="razas">Razas</label>
      </div>
    </div> <!-- Fin primera lista -->
    <div class="col-md-3"> <!-- Segunda lista -->
      <div class="form-floating">
        <select name="generos" id="generos" class="form-select">
          <option value="">Seleccione</option>
        </select>
        <label for="razas">Géneros</label>
      </div>
    </div> <!-- Fin segunda lista -->
    <div class="col-md-3"> <!-- Tercera lista -->
      <div class="form-floating">
        <select name="bandos" id="bandos" class="form-select">
          <option value="">Seleccione</option>
        </select>
        <label for="razas">Bandos</label>
      </div>
    </div> <!-- Fin tercera lista -->
    <div class="col-md-3"> <!-- Contenedor para botones -->
      <div class="d-grid"> <!-- Ocupar 100% -->
        <div class="btn-group-vertical" role="group"> <!-- Botones en grupo vertical -->
          <button class="btn btn-sm btn-outline-secondary" type="button" id="filtrar">Filtrar datos</button>
          <button class="btn btn-sm btn-outline-secondary" type="button" id="exportar">Exportar PDF</button>
        </div>
      </div>
    </div> <!-- Fin contenedor botones -->
  </div> <!-- Fin row -->
</form> <!-- Fin formulario -->

```

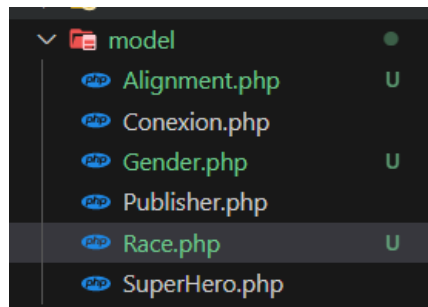
3. En la fila creada para la tabla, construimos:

```

<!-- Espacio para la tabla -->
<div class="row mt-4">
  <div class="col-md-12">
    <table class="table table-sm" id="table-superhero">
      <thead>
        <tr>
          <th>ID</th>
          <th>SuperHero Name</th>
          <th>Hair colour</th>
          <th>Publisher</th>
          <th>Weight kg</th>
        </tr>
      </thead>
      <tbody>
        <!-- Asíncrono -->
      </tbody>
    </table>
  </div> <!-- Fin row -->
</div> <!-- Fin container -->

```

4. Agregue 3 nuevos modelos



Agregan los que se muestran con U (*Untracked según Git*)

5. Construyendo los métodos. Para las 3 últimas clases creadas en el método anterior será muy parecido. Cambiaremos: nombre de clase, nombre del método y consulta SQL contenida dentro del método prepare(""). NOTA: ya no retornamos un arreglo asociativo, sino un array basado en índices – FETCH_NUM.

```
Alignment.php M X
model > Alignment.php
You, now | 1 author (You)
1  <?php
2
3  require_once 'Conexion.php';
4
5  class Alignment extends Conexion{
6
7      private $conexion;
8
9      public function __CONSTRUCT(){
10         $this->conexion = parent::getConexion();
11     }
12
13     public function listarAlignments(){
14         try{
15             $consulta = $this->conexion->prepare("SELECT * FROM alignment");
16             $consulta->execute();
17             return $consulta->fetchAll(PDO::FETCH_NUM);
18         }
19         catch(Exception $e){
20             die($e->getMessage());
21         }
22     }
23
24 }
```

Modelo para alignment (alineaciones/bando)

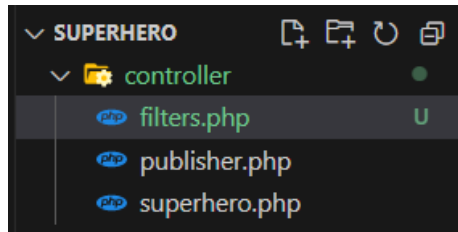
```
Gender.php M X
model > Gender.php
You, 4 seconds ago | 1 author (You)
1  <?php
2
3  require_once 'Conexion.php';
4
5  class Gender extends Conexion{
6
7      private $conexion;
8
9      public function __CONSTRUCT(){
10         $this->conexion = parent::getConexion();
11     }
12
13     public function listarGenders(){
14         try{
15             $consulta = $this->conexion->prepare("SELECT * FROM gender");
16             $consulta->execute();
17             return $consulta->fetchAll(PDO::FETCH_NUM);
18         }
19         catch(Exception $e){
20             die($e->getMessage());
21         }
22     }
23
24 }
```

Modelo para gender (géneros)

```
Race.php M X
model > Race.php
You, 1 second ago | 1 author (You)
1  <?php
2
3  require_once 'Conexion.php';
4
5  class Race extends Conexion{
6
7      private $conexion;
8
9      public function __CONSTRUCT(){
10         $this->conexion = parent::getConexion();
11     }
12
13     public function listarRaces(){
14         try{
15             $consulta = $this->conexion->prepare("SELECT * FROM race");
16             $consulta->execute();
17             return $consulta->fetchAll(PDO::FETCH_NUM);
18         }
19         catch(Exception $e){
20             die($e->getMessage());
21         }
22     }
23
24 }
```

Modelo para race (razas)

6. Si bien podemos crear un controlador para cada modelo, es más fácil (y esto es parte del criterio del desarrollador), construir un único componente que reciba las peticiones de la vista de filtro-avanzado.html y se encargue de brindarle respuesta. Recuerde que los modelos escritos en el paso anterior, solo retornan dato, un proceso bastante sencillo.

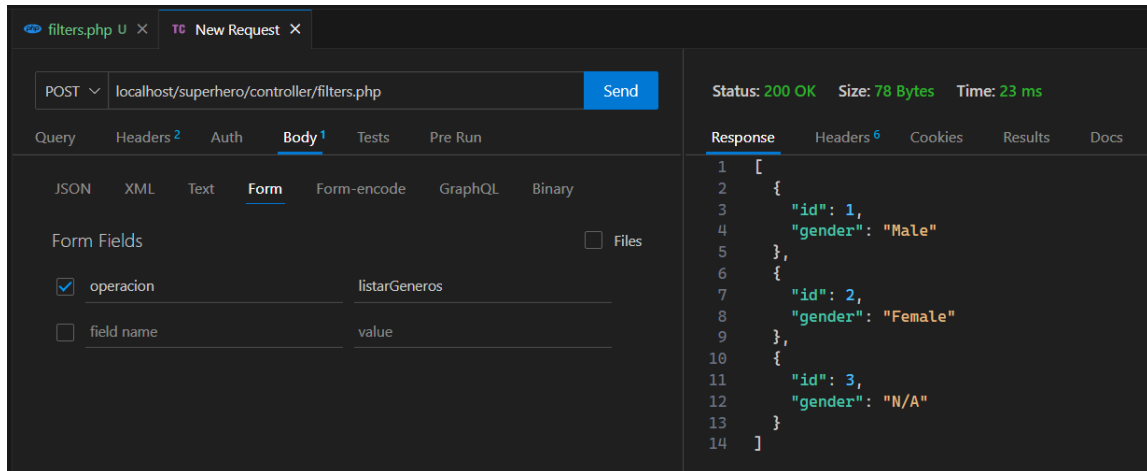


Agregamos el controlador filters.php

Codificación

```
filters.php U X TC New Request
controller > filters.php
1  <?php
2
3  require_once '../model/Alignment.php';
4  require_once '../model/Gender.php';
5  require_once '../model/Race.php';
6
7  //Esta función recibe un objeto tipo arreglo y renderiza
8  //en el navegador JSON siempre que contenga datos
9  function renderJSON($object = []){
10     if ($object){
11         echo json_encode($object);
12     }
13 }
14
15 if (isset($_POST['operacion'])){
16     switch ($_POST['operacion']){
17         case 'listarBandos':
18             $alignment = new Alignment();
19             renderJSON($alignment->listarAlignments());
20             break;
21         case 'listarRazas':
22             $race = new Race();
23             renderJSON($race->listarRaces());
24             break;
25         case 'listarGeneros':
26             $gender = new Gender();
27             renderJSON($gender->listarGenders());
28             break;
29     }
30 }
31
32 }
```

7. Ya que el controlador anterior es un poco distinto a lo que acostumbramos a desarrollar, comprobaremos su funcionalidad con ThunderClient



*Tendrá que probar esto 3 veces, modificando el valor para operación con los posibles datos de entrada: **listarGeneros**, **listarRazas**, **listarBandos***

8. Regresamos a la vista filtro-avanzado.html, ahora crearemos los objetos que referencien las listas, y una sola función que las poblará con datos.

```
85 <script>
86   document.addEventListener("DOMContentLoaded", () => {
87     const selectGeneros = document.querySelector("#generos");
88     const selectRazas = document.querySelector("#razas");
89     const selectBandos = document.querySelector("#bandos");
90
91     //Función que permite poblar cualquiera de los select
92     function getSelectData(operacion = "", objectSelect){
93       const parametros = new URLSearchParams();
94       parametros.append("operacion", operacion);
95       fetch('./controller/filters.php', {
96         method: 'POST',
97         body: parametros
98       })
99       .then(respuesta => respuesta.json())
100       .then(datos => {
101         datos.forEach(element => {
102           const optionTAG = document.createElement("option");
103           //Actualización: no se utilizará claves, sino índices FETCH_NUM
104           optionTAG.value = element[0];
105           optionTAG.text = element[1];
106           objectSelect.appendChild(optionTAG);
107         });
108       });
109     }
110
111     //Se invoca a cada proceso especificando la lista donde deberá renderizar
112     getSelectData("listarGeneros", selectGeneros);
113     getSelectData("listarRazas", selectRazas);
114     getSelectData("listarBandos", selectBandos);
115   });
116 </script>
```

9. Agregamos algunas funcionalidades extras a nuestra vista

```
<script>
document.addEventListener("DOMContentLoaded", () => {
  const selectGeneros = document.querySelector("#generos");
  const selectRazas = document.querySelector("#razas");
  const selectBandos = document.querySelector("#bandos");
  const btFiltrar = document.querySelector("#filtrar");
  const cuerpoTabla = document.querySelector("#table-superhero tbody");
```

Agregado: Los objetos para manejar el botón filtrar y acceder al cuerpo de la tabla

10. Función que renderiza los datos (obtenidos del filtro)

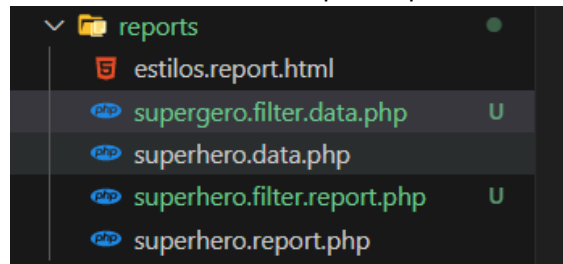
```
113 //Construye la tabla de acuerdo a los criterios de búsqueda
114 function renderSuperHero(){
115   const parametros = new URLSearchParams();
116   parametros.append("operacion", "filtrar");
117   parametros.append("race_id", parseInt(selectRazas.value));
118   parametros.append("gender_id", parseInt(selectGeneros.value));
119   parametros.append("alignment_id", parseInt(selectBandos.value));
120
121   fetch('./controller/superhero.php', {
122     method: 'POST',
123     body: parametros
124   })
125   .then(respuesta => respuesta.json())
126   .then(datos => {
127     cuerpoTabla.innerHTML = '';
128     datos.forEach(element => {
129       const nuevaFila = `
130       <tr>
131         <td>${element.id}</td>
132         <td>${element.superhero_name}</td>
133         <td>${element.hair_colour}</td>
134         <td>${element.publisher_name}</td>
135         <td>${element.weight_kg}</td>
136       </tr>
137       `;
138       cuerpoTabla.innerHTML += nuevaFila;
139     });
140   })
141   .catch(error => {
142     cuerpoTabla.innerHTML = '';
143     alert('No encontramos datos');
144   });
145 }
```

11. Finalmente, evento clic que invoca a esta función (paso 10)

```
151
152 //Renderizando tabla
153 btFiltrar.addEventListener("click", renderSuperHero);
154 });
155 </script>
```

CREACIÓN DE REPORTE PDF

1. Iniciaremos creando dos ficheros en la carpeta reports



*Recomendación: Si va a crear múltiples reportes, crear un subdirectorio para cada uno
Los archivos agregados son los que se muestran con U (Untracked - GIT)*

2. Agregue el siguiente código para el **superhero.filter.report.php**

```
superhero.filter.report.php U X
reports > superhero.filter.report.php
1  <?php
2  require_once '../vendor/autoload.php';
3  require_once '../model/SuperHero.php';
4
5  use Spipu\Html2Pdf\Html2Pdf;
6  use Spipu\Html2Pdf\Exception\Html2PdfException;
7  use Spipu\Html2Pdf\Exception\ExceptionFormatter;
8
9  try {
10     //Instanciar clase SuperHero
11     $superhero = new SuperHero();
12
13     //Recepcionando los datos para el filtro desde la vista
14     $filtros = [
15         "race_id"    => $_GET['race_id'],
16         "gender_id" => $_GET['gender_id'],
17         "alignment_id" => $_GET['alignment_id']
18     ];
19     $datos = $superhero->filtrarSuperHero($filtros);
20     $titulo = $_GET['titulo'];
21
22     ob_start();
23
24     //Archivos que componen PDF
25     include './estilos.report.html';
26     include './supergero.filter.data.php';
27
28     $content = ob_get_clean();
29
30     $html2pdf = new Html2Pdf('P', 'A4', 'es');
31     $html2pdf->writeHTML($content);
32     $html2pdf->output('SuperHero.pdf');
33 } catch (Html2PdfException $e) {
34     $html2pdf->clean();
35
36     $formatter = new ExceptionFormatter($e);
37     echo $formatter->getHtmlMessage();
38 }
```

Los cambios importantes se muestran resaltados

3. En el archivo **superhero.filter.data.php**

```
superhero.filter.data.php U X
reports > superhero.filter.data.php
1  <h1 class="text-md text-center"><?=$titulo?></h1>
2
3  <table class="table table-border mt-3">
4    <colgroup>
5      <col style='width: 5%'>
6      <col style='width: 35%'>
7      <col style='width: 15%'>
8      <col style='width: 25%'>
9      <col style='width: 10%'>
10   </colgroup>
11   <thead>
12     <tr>
13       <th>ID</th>
14       <th>Nombre</th>
15       <th>Color cabello</th>
16       <th>Publicado</th>
17       <th>Peso</th>
18     </tr>
19   </thead>
20   <tbody>
21     <?php foreach($datos as $registro): ?>
22     <tr>
23       <td><?=$registro['id']?></td>
24       <td><?=$registro['superhero_name']?></td>
25       <td><?=$registro['hair_colour']?></td>
26       <td><?=$registro['publisher_name']?></td>
27       <td><?=$registro['weight_kg']?></td>
28     </tr>
29     <?php endforeach; ?>
30   </tbody>
31 </table>
```

4. Finalmente, en la vista agregamos:

```
<script>
document.addEventListener("DOMContentLoaded", () => {
  const selectGeneros = document.querySelector("#generos");
  const selectRazas = document.querySelector("#razas");
  const selectBandos = document.querySelector("#bandos");
  const btFiltrar = document.querySelector("#filtrar");
  const btExportar = document.querySelector("#exportar");
  const cuerpoTabla = document.querySelector("#table-superhero tbody");
```

5. Nueva función

```
148 function createPDF() {
149     const parametros = new URLSearchParams();
150     parametros.append("race_id", parseInt(selectRazas.value));
151     parametros.append("gender_id", parseInt(selectGeneros.value));
152     parametros.append("alignment_id", parseInt(selectBandos.value));
153
154     //También tendremos que enviar el título (se obtiene de cada SELECT)
155     let titulo = '';
156     titulo += selectBandos.options[selectBandos.selectedIndex].text.trim() + " - ";
157     titulo += selectGeneros.options[selectGeneros.selectedIndex].text.trim() + " - ";
158     titulo += selectRazas.options[selectRazas.selectedIndex].text.trim();
159
160     //Título estructurado, lo podemos enviar
161     parametros.append("titulo", titulo);
162
163     console.log(titulo);
164
165     //URL destino
166     window.open('./reports/superhero.filter.report.php?${parametros}', '_blank');
167 }
```

Invocamos la función anterior desde evento click

```
173
174     //Renderizando tabla
175     btFiltrar.addEventListener("click", renderSuperHero);
176     btExportar.addEventListener("click", createPDF);
177 };
178 </script>
179
```