Controller Pelicula:

```
@RestController
@RequestMapping("/peliculas")
@CrossOrigin(origins = "http://127.0.0.1:5500")
public class PeliculaController {
  private List<Pelicula> listaPeliculas;
  public PeliculaController() {
    listaPeliculas = new ArrayList<>();
    List<Actor> actores1 = new ArrayList<>();
    actores1.add(new Actor(1, "Leonardo DiCaprio", "Americano"));
    actores1.add(new Actor(2, "Joseph Gordon-Levitt", "Americano"));

    List<Actor> actores2 = Arrays.asList(new Actor(3, "Christian Bale", "Británico"),
                        new Actor(4, "Heath Ledger", "Australiano"));
    List<Actor> actores3 = Arrays.asList(new Actor(5, "Matthew McConaughey",
"Americano"),
                        new Actor(6, "Anne Hathaway", "Americana"));
    List<Actor> actores4 = Arrays.asList(new Actor(7, "John Travolta", "Americano"),
                        new Actor(8, "Samuel L. Jackson", "Americano"));
    listaPeliculas.add(new Pelicula(1, "Inception", "Christopher Nolan",
LocalDate.of(2010, 7, 16), 148, actores1));
    listaPeliculas.add(new Pelicula(2, "The Dark Knight", "Christopher Nolan",
LocalDate.of(2008, 7, 18), 152, actores2));
    listaPeliculas.add(new Pelicula(3, "Interstellar", "Christopher Nolan",
LocalDate.of(2014, 11, 7), 169, actores3));
    listaPeliculas.add(new Pelicula(4, "Pulp Fiction", "Quentin Tarantino",
LocalDate.of(1994, 10, 14), 154, actores4));
  }
```

```
    // Mostrar todas las películas
    @GetMapping
    public ResponseEntity<List<Pelicula>> obtenerTodasLasPeliculas() {
            return ResponseEntity.ok(listaPeliculas);
    }
    // Consultar una película por su título
    @GetMapping("/titulo/{titulo}")
    public ResponseEntity<Pelicula> obtenerPeliculaPorTitulo(@PathVariable String titulo) {
      for (Pelicula pelicula : listaPeliculas) {
        if (pelicula.getTitulo().equalsIgnoreCase(titulo)) {
          return ResponseEntity.ok(pelicula);
        }     }
      return ResponseEntity.notFound().build();
    }
    // Crear una nueva película
    @PostMapping
    public ResponseEntity<Pelicula> crearPelicula(@RequestBody Pelicula pelicula) {
      listaPeliculas.add(pelicula);
      return ResponseEntity.ok(pelicula);
    }
    // Modificar la información de una película (completo)
    @PutMapping
    public ResponseEntity<Pelicula> actualizarPelicula(@RequestBody Pelicula
peliculaActualizada) {
      for (int i = 0; i < listaPeliculas.size(); i++) {
        Pelicula pelicula = listaPeliculas.get(i);
        if (pelicula.getId().equals(peliculaActualizada.getId())) {
          pelicula.setTitulo(peliculaActualizada.getTitulo());
          pelicula.setDirector(peliculaActualizada.getDirector());
          pelicula.setFechaLanzamiento(peliculaActualizada.getFechaLanzamiento());
          pelicula.setDuracion(peliculaActualizada.getDuracion());
```

```
            pelicula.setActores(peliculaActualizada.getActores());
            return ResponseEntity.noContent().build();
        }
    }
    return ResponseEntity.notFound().build();
  }
  // Modificar la información de una película (parcial)
  @PatchMapping
  public ResponseEntity<Pelicula> actualizarParcialmentePelicula( @RequestBody
Pelicula peliculaActualizada) {
    for (Pelicula pelicula : listaPeliculas) {
      if (pelicula.getId().equals(peliculaActualizada.getId())) {
        if (peliculaActualizada.getTitulo() != null) {
          pelicula.setTitulo(peliculaActualizada.getTitulo());
        }
        if (peliculaActualizada.getDirector() != null) {
          pelicula.setDirector(peliculaActualizada.getDirector());
        }
        if (peliculaActualizada.getFechaLanzamiento() != null) {
          pelicula.setFechaLanzamiento(peliculaActualizada.getFechaLanzamiento());
        }
        if (peliculaActualizada.getDuracion() != 0) {
          pelicula.setDuracion(peliculaActualizada.getDuracion());
        }
        if (peliculaActualizada.getActores() != null) {
          pelicula.setActores(peliculaActualizada.getActores());
        }
        return ResponseEntity.noContent().build();
      }
    }
    return ResponseEntity.notFound().build();    }
```

```
    // Eliminar una película por su id
    @DeleteMapping("/{id}")
    public ResponseEntity<String> eliminarPelicula(@PathVariable Integer id) {
        for (int i = 0; i < listaPeliculas.size(); i++) {
            Pelicula pelicula = listaPeliculas.get(i);
            if (pelicula.getId().equals(id)) {
                listaPeliculas.remove(i);
                return ResponseEntity.noContent().build();
            }      }
        return ResponseEntity.notFound().build();
    }
    // Obtener todas las películas por un director específico
    @GetMapping("/director/{director}")
    public ResponseEntity<List<Pelicula>> obtenerPeliculasPorDirector(@PathVariable
String director) {
        List<Pelicula> resultado = new ArrayList<>();
        for (Pelicula pelicula : listaPeliculas) {
            if (director.equalsIgnoreCase(pelicula.getDirector())) {
                resultado.add(pelicula);
            }
        }
        if (resultado.isEmpty()) {
                return ResponseEntity.notFound().build();
        }
        return ResponseEntity.ok(resultado);
    }
```

```
// Obtener todas las películas de los últimos 5 años
@GetMapping("/recientes")
public ResponseEntity<List<Pelicula>> obtenerPeliculasRecientes() {
   List<Pelicula> resultado = new ArrayList<>();
   LocalDate hoy = LocalDate.now();

   for (Pelicula pelicula : listaPeliculas) {

        Period peli= pelicula.getFechaLanzamiento().until(hoy);
      if (peli.getYears() <= 5) {
         resultado.add(pelicula);
      }
   }
   if (resultado.isEmpty()) {
        return ResponseEntity.notFound().build();
   }
   return ResponseEntity.ok(resultado);
}

// Obtener la película con la mayor duración
@GetMapping("/mayorDuracion")
public ResponseEntity<Pelicula> obtenerPeliculaMayorDuracion() {
   if (listaPeliculas.isEmpty()) {
        return ResponseEntity.notFound().build();
   }
   Pelicula mayorDuracion = listaPeliculas.get(0);
   for (Pelicula pelicula : listaPeliculas) {
      if (pelicula.getDuracion() > mayorDuracion.getDuracion()) {
         mayorDuracion = pelicula;
      }
   }
```

```
        return ResponseEntity.ok(mayorDuracion);
    }

    // Obtener un mapa con los directores con más de X películas
    @GetMapping("/directoresConMasDe/{x}")
    public ResponseEntity<Map<String, Integer>>
obtenerDirectoresConMasDeXPeliculas(@PathVariable int x) {
        Map<String, Integer> conteoPorDirector = new HashMap<>();
        for (Pelicula pelicula : listaPeliculas) {
            conteoPorDirector.put(pelicula.getDirector(),
conteoPorDirector.getOrDefault(pelicula.getDirector(), 0) + 1);
        }
        Map<String, Integer> resultado = new HashMap<>();
        for (Map.Entry<String, Integer> entry : conteoPorDirector.entrySet()) {
            if (entry.getValue() > x) {
                resultado.put(entry.getKey(), entry.getValue());
            }
        }
        if (resultado.isEmpty()) {
            return ResponseEntity.notFound().build();
        }
        return ResponseEntity.ok(resultado);
    }

    // Obtener la lista de todos los actores
    @GetMapping("/actores")
    public ResponseEntity<Set<Actor>> obtenerTodosLosActores() {
        Set<Actor> actores = new HashSet<>();
        for (Pelicula pelicula : listaPeliculas) {
            for (Actor actor : pelicula.getActores()) {
                actores.add(actor);
            }
```

```
    }
    return ResponseEntity.ok(actores);
  }

  @GetMapping("/actor/{nombre}")
  public ResponseEntity<List<Pelicula>> obtenerPeliculasPorActor(@PathVariable String nombre) {
    List<Pelicula> peliculasDelActor = new ArrayList<>();
    for (Pelicula pelicula : listaPeliculas) {
      for (Actor actor : pelicula.getActores()) {
        if (actor.getNombre().equalsIgnoreCase(nombre)) {
          peliculasDelActor.add(pelicula);
          break; // No necesitamos seguir buscando en esta película
        }
      }
    }
    if (peliculasDelActor.isEmpty()) {
        return ResponseEntity.notFound().build();
    }
    return ResponseEntity.ok(peliculasDelActor);
  }

  @GetMapping("/actores/nacionalidad/{nacionalidad}")
  public ResponseEntity<Set<Actor>> obtenerActoresPorNacionalidad(@PathVariable String nacionalidad) {
    Set<Actor> actoresPorNacionalidad = new HashSet<>();
    for (Pelicula pelicula : listaPeliculas) {
      for (Actor actor : pelicula.getActores()) {
        if (actor.getNacionalidad().equalsIgnoreCase(nacionalidad)) {
          actoresPorNacionalidad.add(actor);
        }
```

```
        }
    }
    if (actoresPorNacionalidad.isEmpty()) {
        return ResponseEntity.notFound().build();
    }
    return ResponseEntity.ok(actoresPorNacionalidad);
  }
}
```

AJAX:

```
<body>
  <button id="botonMostrar">Mostrar todas las películas</button> <input type="text" id="pelicula"
    placeholder="Película"> <button id="botonBuscar">Bucar</button>
  <br>
  <input type="text" id="director" placeholder="Películas por director"> <button id="buscarDirector">Buscar</button>
  <br>
  <input type="text" placeholder="Películas por actor"> <button id="buscarActor">Buscar</button>
  <br>
  <input type="text" placeholder="Actores por nacional"> <button id="buscarActNac">Buscar</button>
  <br>
  <button id="peli5">Películas últimos 5 años</button><button id="peliMay">Película con mayor duración</button><button
    id="actores">Mostrar todos los actores</button>
  <br>
  <input type="text" placeholder="Directores con más de X"> <button id="buscarActor">Buscar</button>
  <table id="tabla">
    <tbody>

    </tbody>
  </table>
  <script>
    var botonM = document.getElementById("botonMostrar");
    var botonB = document.getElementById("botonBuscar");
    var botonD = document.getElementById("buscarDirector");
    var botonNac = document.getElementById("buscarActNac");
```

Poner id al input

```
botonB.addEventListener("click", () => {

  buscarPeli();
});

botonD.addEventListener("click", () => {

  buscarPorDirector();
});

async function fetchData() {
  const response = await fetch("http://localhost:8080/pelicula");
  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoUsuario(data);
}

async function buscarPeli() {

  var pelicula = document.getElementById("pelicula").value;

  const response = await fetch(`http://localhost:8080/pelicula/titulo/${pelicula}`);
  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoUsuario([data]);
}
```

```
async function buscarPeli() {

  var pelicula = document.getElementById("pelicula").value;

  const response = await fetch(`http://localhost:8080/pelicula/titulo/${pelicula}`);
  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoUsuario([data]);
}

async function buscarPorDirector() {

  var director = document.getElementById("director").value;

  const response = await fetch(`http://localhost:8080/pelicula/director/${director}`);

  if(!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoUsuario(data);

}
```

```
async function buscarPorNacionalidad() {

  var nacionalidad = document.getElementById("nacionalidad").value;

  const response = await fetch(`http://localhost:8080/pelicula/actores/nacionalidad/${nacionalidad}`);

  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoUsuario(data);

}
```

```
function mostrarInfoUsuario(data) {

  const tbody = document.getElementById("tabla");

  tbody.innerHTML += `<tr id="patata">
      <th>ID</th>
      <th>Titulo</th>
      <th>Director</th>
      <th>Fecha</th>
      <th>Duración</th>
      <th>Actores</th>
    </tr>`

  data.forEach((element) => {

    const actores = element.listaActores.map((a) => a.nombre).join(", ");

    tbody.innerHTML += `
  <tr>
    <td>${element.id}</td>
    <td>${element.titulo}</td>
    <td>${element.director}</td>
    <td>${element.fechaLanzamiento}</td>
    <td>${element.duracion}</td>
    <td>${actores}</td>
  </tr>`;
  });
}
```

S

```
function mostrarInfoActores(data) {

  const tbody = document.getElementById("tabla");

  console.log(data);

  tbody.innerHTML += `<tr id="patata">
      <th>Nombre</th>
      <th>Nacionalidad</th>
    </tr>`;

  data.forEach((element) => {

    tbody.innerHTML += `<tr>
      <td>${element.nombre}</td>
      <td>${element.nacionalidad}</td>
      </tr>`;
  });
  }
```

```
<input type="text" placeholder=
<table >
  <tbody id="tabla">

  </tbody>
</table>
```

```
<script>
  var botonM = document.getElementById("botonMostrar");
  var botonB = document.getElementById("botonBuscar");
  var botonD = document.getElementById("buscarDirector");
  var botonNac = document.getElementById("buscarActNac");
  var boton5anios = document.getElementById("peli5");

  botonM.addEventListener("click", () => {

    fetchData();
  });

  botonB.addEventListener("click", () => {

    buscarPeli();
  });

  botonD.addEventListener("click", () => {

    buscarPorDirector();
  });

  botonNac.addEventListener("click", () => {

    buscarPorNacionalidad();
  });

  boton5anios.addEventListener("click", () => {

    peliculasUltimosAnios();
  });
```

```
async function buscarPorNacionalidad() {

  var nacionalidad = document.getElementById("nacionalidad").value;

  const response = await fetch(`http://localhost:8080/pelicula/actores/nacionalidad/${nacionalidad}`);

  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log(data);

  mostrarInfoActores(data);

}
```

<body>

<button id="botonMostrar">Mostrar todas las películas</button> <input type="text" id="pelicula" placeholder="Película"> <button id="botonBuscar">Bucar</button>

<input type="text" placeholder="Peliculas por director"> <button id="buscarDirector">Buscar</button>


```
  <input type="text" placeholder="Peliculas por actor"> <button
id="buscarActor">Buscar</button>

  <br>

  <input type="text" placeholder="Actores por nacional"> <button
id="buscarActNac">Buscar</button>

  <br>

  <button id="peli5">Películas últimos 5 años</button><button id="peliMay">Película
con mayor duración</button><button id="actores">Mostrar todos los actores</button>

  <br>

  <input type="text" placeholder="Directores con más de X"> <button
id="buscarActor">Buscar</button>

  <table id="tabla">

   <tbody>


   </tbody>

  </table>

  <script>

   var botonM = document.getElementById("botonMostrar");

   var botonB = document.getElementById("botonBuscar");

   botonM.addEventListener("click", () => {

    fetchData();

   });

   botonB.addEventListener("click", () => {

    buscarPeli();

   });


   async function fetchData() {

    const response = await fetch("http://localhost:8080/pelicula");

    if (!response.ok) {

     throw new Error(` HTTP error! status: ${response.status}`);

    }

    const data = await response.json();
```

```
 console.log(data);
 mostrarInfoUsuario(data);
}

function mostrarInfoUsuario(data) {
 const tbody = document.getElementById("tabla");
 console.log(data.results);
 tbody.innerHTML += `<tr id="patata">
  <th>ID</td>
  <th>Título</td>
  <th>Director</td>
  <th>Fecha</td>
  <th>Duración</td>
  <th>Actores</td>
 </tr>`
 data.forEach((element) =>
  const actores = element.listaActores.map((a) => a.nombre).join(", ");
  tbody.innerHTML += `
<tr>
 <td>${element.id}</td>
 <td>${element.titulo}</td>
 <td>${element.director}</td>
 <td>${element.fechaLanzamiento}</td>
 <td>${element.duracion}</td>
 <td>${actores}</td>
</tr>`;
 });
}
async function buscarPeli(){
 var pelicula = document.getElementById("pelicula").value;
 const response = await fetch(`http://localhost:8080/pelicula/titulo/${pelicula}`);
```

```
      if (!response.ok) {
        throw new Error(` HTTP error! status: ${response.status}`);
      }
      const data = await response.json();
      console.log(data);
      mostrarInfoUsuario([data]);
    }
  </script>
 </body>
</html>
```