

2022

Proyecto Altair Eventos

MANUAL DE USUARIO
ANDRES PEREZ ROVIRA

Índice

Introducción	2
Configuración del entorno.....	2
.....	2
Estructura de paquetes del proyecto.....	4
Mi página web	4
Index.....	¡Error! Marcador no definido.
Funcionalidades	7
Funcional 1	7
Funcional 2	8
Funcional 3	8
Funcional 4	9
Funcional 5	9
Funcional 6	10
Funcional 7	11
Funcional 8	11
Funcional 9	12
Funcional 10.....	12

Introducción

Este documento servirá para dar una idea general del proyecto realizado, uso genérico de la aplicación tanto a nivel usuario como a nivel técnico y ver las funcionalidades más detalladamente realizadas.

La ruta que utilizaremos es:

<https://altaireventos.andrespr.es/altairAudiovisuales>

Configuración del entorno

Herramientas necesarias para el curso

- JDK (Java SE Development Kit)
- Spring Tools IDE (Eclipse)
- Maven
- MySQL 8

También necesitaremos tomcat en nuestro servidor Linux para nuestro despliegue.

Tras tener todo el entorno configurado abriremos nuestro id y crearemos nuestro proyecto

File> new> Spring started Project.

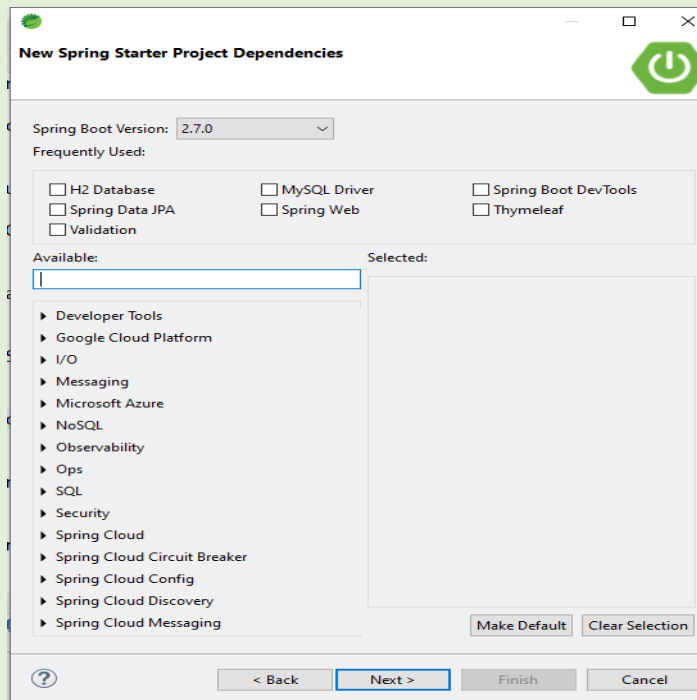
Nos saldrá la siguiente ventana:

The screenshot shows the 'New Spring Starter Project' dialog box. It contains the following fields and options:

- Service URL:**
- Name:**
- ☒ **Use default location**
- Location:**
- Type:** **Packaging:**
- Java Version:** **Language:**
- Group:**
- Artifact:**
- Version:**
- Description:**
- Package:**
- Working sets:**
 - ☐ **Add project to working sets** - Working sets:**

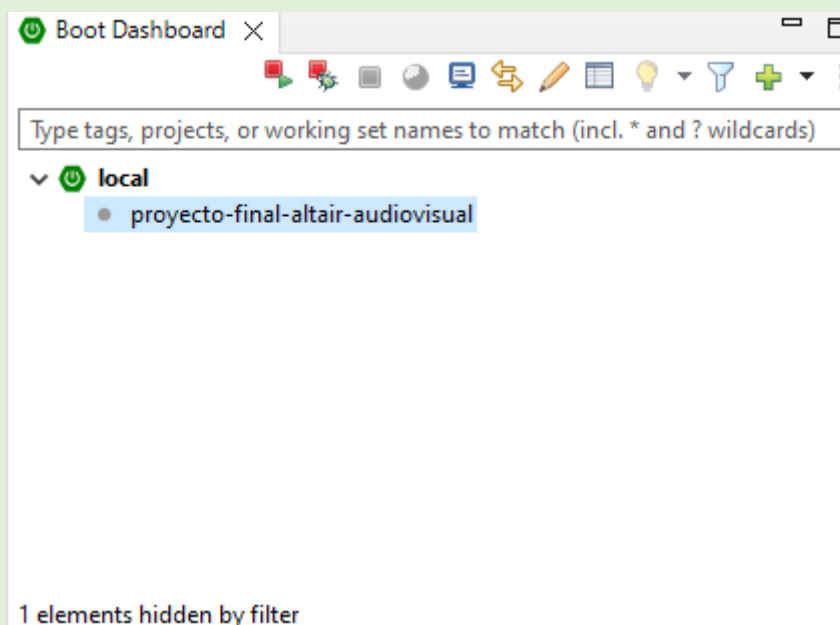
At the bottom, there are buttons:

Después de ponerle el nombre a nuestro proyecto si le damos a next podemos añadir compontes que necesitemos a la hora de desarrollarlo.

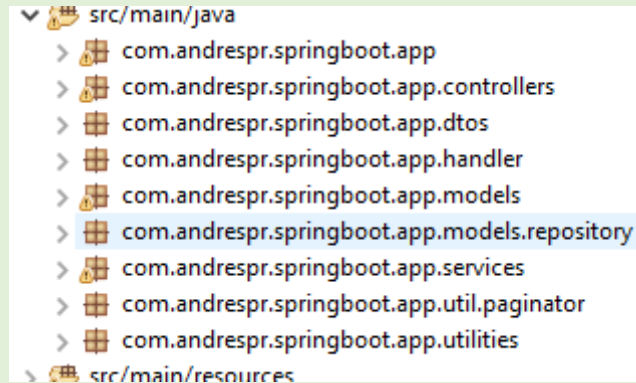


Una ventaja que tiene Spring es que no requiere de poner explícitamente las dependencias e instalar los jar manualmente, ya que se pueden añadir a la hora de crear el proyecto automáticamente.

A diferencia de Eclipse, Spring Tool Suite dispone de un tomcat embebido, y no es necesaria su instalación, simplemente ya teniendo el proyecto creado podemos arrancarlo desde esta ventana, y todos los cambios que se harán en el código afectarán en tiempo real a nuestra aplicación.



Estructura de paquetes del proyecto.



Ya creados los paquetes y teniendo definido las entidades, debemos de realizar la estructura para la adquisición de los distintos métodos que trabajaran con la base de datos, como por ejemplo: Crear Repositorios, Servicios, Controladores... Y de los distintos controladores hacer uso de estos.

Mi página web

Índex

Esta será la página home de mi aplicación, con la única opción de loguearse.



Formulario de Login:

Sign In

Sign In

Registro Trabajador

Registro Productor

Tras iniciar sesión, dependiendo del rol se desplegará un menú u otro:

Ejemplo de Producer:


[Home](#)
[Materiales Audiovisuales](#)
[Localizaciones](#)
[Productores](#)
[Empresas Trabajadoras](#)
[Events](#)
[EventHasMaterials](#)

Perfil

[Localizaciones](#)
[Productores](#)
[Empresas Trabajadoras](#)
[Events](#)
[EventHasMaterials](#)

Perfil

Detalle del userAccount: producer1



Username:	producer1
Email:	producer1@email.es
Rol:	ROLE_PRODUCER
Activo:	true

Mi aplicación tiene un CRUD en distintitas tablas, con formularios de crear o eliminar, y vistas detalle

Vista ejemplo crear y editar:

Añadir Empresa Trabajadora

Añadir Empresa Trabajadora

Nombre Empresa

Trabajadora

Province

Email

Postal Code

cif


phone

Direccion

Foto

Vista detalle:

Detalle de la Empresa Trabajadora: Ornare Ltd



Nombre de la Empresa:	Ornare Ltd
Provincia:	Ceuta
Email de la Empresa:	a.neque@icloud.couk
Direccion:	621-4549 Sodales, Rd.
Codigo Postal:	48918
Cif:	G05390042
Telefono de Contato:	776620707

Listado de compañías trabajadoras

Busqueda por nombre Busqueda por provincia

Nombre Empresa	Provincia	Editar	Eliminar	Detalle	Lista Trabajadores de la Empresa
Ornare Ltd	Ceuta	<input type="button" value="Editar Empresa"/>	<input type="button" value="Eliminar Empresa"/>	<input type="button" value="Detalle Empresa"/>	<input type="button" value="Trabajadores"/>
Sed Libero LLP	Castilla y Leon	<input type="button" value="Editar Empresa"/>	<input type="button" value="Eliminar Empresa"/>	<input type="button" value="Detalle Empresa"/>	<input type="button" value="Trabajadores"/>
Cras Sed LLP	Murcia	<input type="button" value="Editar Empresa"/>	<input type="button" value="Eliminar Empresa"/>	<input type="button" value="Detalle Empresa"/>	<input type="button" value="Trabajadores"/>
Posuere Vulputate Inc.	Catalunya	<input type="button" value="Editar Empresa"/>	<input type="button" value="Eliminar Empresa"/>	<input type="button" value="Detalle Empresa"/>	<input type="button" value="Trabajadores"/>
Eu Nibh Vulputate Corporation	Castilla - La Mancha	<input type="button" value="Editar Empresa"/>	<input type="button" value="Eliminar Empresa"/>	<input type="button" value="Detalle Empresa"/>	<input type="button" value="Trabajadores"/>

Primera « 1 2 » Última

El cual tendrá diferentes buscadores con filtros, paginación, y botones personalizados como ver los trabajadores de una empresa, o ir a la configuración de la cuenta de usuario.

Funcionalidades

1. La Empresa trabajadora es la que organiza a sus trabajadores, los cuales tienen diferentes categorías, el jefe es el único que puede modificar los datos de sus trabajadores.
2. No puede haber 2 eventos el mismo día con la misma empresa pero si con otra empresa.
3. No puedes crear un evento si el nombre es el mismo que uno que exista en base de datos
4. El productor es el único junto al admin que puede crear un evento.
5. No se puede crear un evento si la fecha de montaje es después de la del evento y la fecha de desmontaje sea antes del evento.
6. El trabajador sólo verá los eventos a los que asistirá y su propio perfil.
7. No se podrá eliminar una localización si hay un evento futuro
8. No se podrá eliminar una empresa si tiene un evento futuro
9. El productor solo verá sus datos y no puede eliminarse, podrá ver las empresas y sus trabajadores pero no podrá editarlos o borrarlos.
10. No podrá crearse un usuario si existe uno con el mismo nombre en la base de datos.

Funcional 1

Vista:

Home Empresas Trabajadoras Events Perfil boss1

Trabajadores de la empresa Ornare Ltd

Crear Trabajador

Nombre	Apellidos	Editar	Eliminar	Detalle	UserAccount
Adenaa	Bond	Editar Worker	Eliminar Worker	Detalle del Worker	Detalle UserAccount
Ivory	Velazquez	Editar Worker	Eliminar Worker	Detalle del Worker	Detalle UserAccount
Zelda	Nicholson	Editar Worker	Eliminar Worker	Detalle del Worker	Detalle UserAccount

Código:

```
<table class="table table-striped">
  <thead>
    <tr>
      <th>Nombre</th>
      <th>Apellidos</th>
      <th sec:authorize="hasAnyAuthority('ROLE_ADMIN','ROLE_BOSS')">Editar</th>
      <th sec:authorize="hasAnyAuthority('ROLE_ADMIN','ROLE_BOSS')">Eliminar</th>
      <th>Detalle</th>
      <th sec:authorize="hasAnyAuthority('ROLE_ADMIN','ROLE_BOSS')">UserAccount</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Adenaa</td>
      <td>Bond</td>
      <td>Editar Worker</td>
      <td>Eliminar Worker</td>
      <td>Detalle del Worker</td>
      <td>Detalle UserAccount</td>
    </tr>
    <tr>
      <td>Ivory</td>
      <td>Velazquez</td>
      <td>Editar Worker</td>
      <td>Eliminar Worker</td>
      <td>Detalle del Worker</td>
      <td>Detalle UserAccount</td>
    </tr>
    <tr>
      <td>Zelda</td>
      <td>Nicholson</td>
      <td>Editar Worker</td>
      <td>Eliminar Worker</td>
      <td>Detalle del Worker</td>
      <td>Detalle UserAccount</td>
    </tr>
  </tbody>
</table>
```

Con sec:authorize cualidad de spring security nos permite según el rol mostrar o no un botón.

Funcional 2

Vista:

La Empresa ya tiene un evento ese día

Crear Evento

Crear Evento

Código:

```
events = eventService.findAll().stream().filter(e -> e.getHardworkingCompany() == event.getHardworkingCompany()
    && e.getEventDate().getDate() == event.getEventDate().getDate()).toList();

if (events.size() > 0) {
    flash.addAttribute("error", "La Empresa ya tiene un evento ese día");
    return "redirect:/event/create";
}
```

Funcional 3

Vista:

Ya existe el nombre de ese evento en la base de datos

Crear Evento

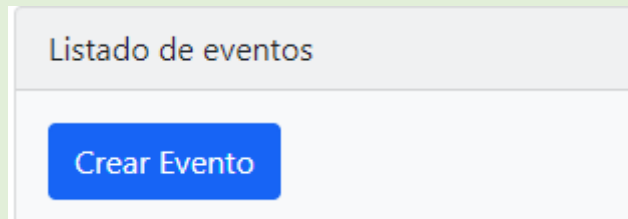
Crear Evento

Código:

```
List<Event> existe = eventService.findByeventName(event.getEventName());
if(existe.size()>0) {
    model.addAttribute("titulo", "Formulario de eventos");
    List<Producer> producers = producerService.findAll();
    model.addAttribute("producers", producers);
    List<HardworkingCompany> hardworkingCompany = hardworkingCompanyService.findAll();
    model.addAttribute("hardworkingCompany", hardworkingCompany);
    List<Location> locations = locationService.findAll();
    model.addAttribute("locations", locations);
    String mensaje = "Ya existe el nombre de ese evento en la base de datos";
    flash.addAttribute("error", mensaje);
    return "redirect:/event/create";
}
```

Funcional 4

Vista:



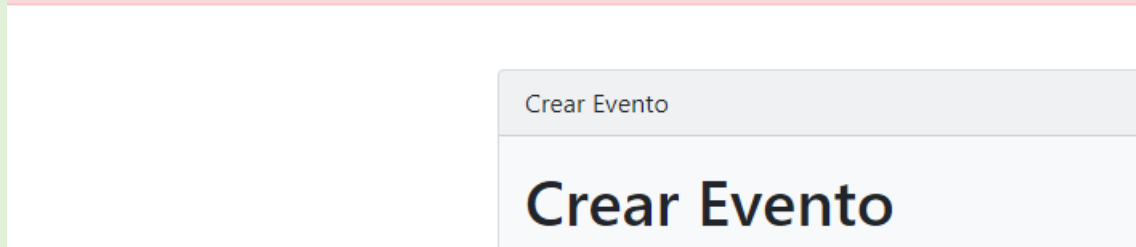
Código:

```
<a sec:authorize="hasAnyAuthority('ROLE_ADMIN', 'ROLE_PRODUCER')"
  th:href="@{/event/create}" class="btn btn-primary btn-xs"
  style="margin-bottom: 10px">Crear Evento</a>
```

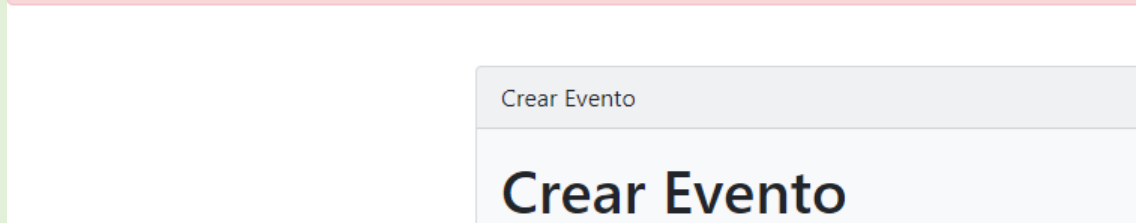
Funcional 5

Vista:

No puedes crear un evento si la fecha del montaje es posterior a la fecha del evento



No puedes crear un evento si la fecha del Desmontaje es anterior a la fecha del evento



Código:

```
// Funcionalidad:Que no se puede crear un evento si la fecha de montaje es
// despues de la del evento y que la fecha de desmontaje sea antes del evento
if (event.getEventDate().before(event.getStartDate())) {

    model.addAttribute("titulo", "Formulario de eventos");
    List<Producer> producers = producerService.findAll();
    model.addAttribute("producers", producers);
    List<HardworkingCompany> hardworkingCompanyys = hardworkingCompanyService.findAll();
    model.addAttribute("hardworkingCompanyys", hardworkingCompanyys);
    List<Location> locations = locationService.findAll();
    model.addAttribute("locations", locations);
    String mensaje = "No puedes crear un evento si la fecha del montaje es posterior a la fecha del evento";
    flash.addFlashAttribute("error", mensaje);
    return "redirect:/event/create";
}

if (event.getEventDate().after(event.getEndDate())) {

    model.addAttribute("titulo", "Formulario de eventos");
    List<Producer> producers = producerService.findAll();
    model.addAttribute("producers", producers);
    List<HardworkingCompany> hardworkingCompanyys = hardworkingCompanyService.findAll();
    model.addAttribute("hardworkingCompanyys", hardworkingCompanyys);
    List<Location> locations = locationService.findAll();
    model.addAttribute("locations", locations);
    String mensaje = "No puedes crear un evento si la fecha del Desmontaje es anterior a la fecha del evento";
    flash.addFlashAttribute("error", mensaje);
    return "redirect:/event/create";
}
```

Funcional 6

Vista:

[Perfil](#) [worker2](#)

Listado de eventos

Nombre Evento	Fecha Evento	Detalle
partido 1	2022-06-11	Detalle Evento

Código:

```
if (rol.equals("[ROLE_WORKER]")) {
    // Tengo que recuperar la id del trabajador

    Worker workers = workerService.findAll().stream()
        .filter(ad -> ad.getUserAccount().getId() == usuario.getId()).toList().get(0);
    model.addAttribute("titulo", "Tus datos");
    model.addAttribute("workers", workers);
    return "worker/listAll";
}
```

Funcional 7

Vista:

No puedes eliminar esta localizacion porque esta en un evento futuro

Listado de Localizaciones

Crear Localizacion

Buscar

Busqueda por nombre

Código:

```
// funcionalidad 4: No se podrá eliminar una localizacion si hay un evento
// futuro.
// Cogemos la fecha
Date ahora = new Date();

List<Event> events = eventService.findAll().stream()
    .filter(e -> e.getLocation().getId() == id && e.getEventDate().getDate() >= ahora.getDate()).toList();

if (events.size() > 0) {
    flash.addFlashAttribute("error", "No puedes eliminar esta localizacion porque esta en un evento futuro");
    return "redirect:/location/listAll";
}
```

Funcional 8

No puedes eliminar esta Empresa porque esta en un evento futuro

Listado de compañías trabajadoras

Crear Empresa Trabajadora

Buscar

Busqueda por nombre

Código:

```
// funcionalidad 5: No se podrá eliminar una Empresa Trabajadora si hay un
// evento futuro.
// Cogemos la fecha
Date ahora = new Date();

List<Event> events = eventService.findAll().stream()
    .filter(e -> e.getHardworkingCompany().getId() == id && e.getEventDate().getDate() >= ahora.getDate()).toList();

if (events.size() > 0) {
    flash.addFlashAttribute("error", "No puedes eliminar esta Empresa porque esta en un evento futuro");
    return "redirect:/hardworkingCompany/listAll";
}
```

Funcional 9

Vista:

Tus datos				
<input type="text" value="Buscar"/> Busqueda por nombre				
Nombre	Apellidos	Editar	Detalle	UserAccount
Zeph	Joyce	Editar Productor	Detalle Productor	Configurar UserAccount

Código:

```
if (rol.equals("[ROLE_PRODUCER]")) {
    // Tengo que recuperar la id del trabajador

    Producer producer = producerService.findAll().stream()
        .filter(ad -> ad.getUserAccount().getId() == usuario.getId()).toList().get(0);
    model.addAttribute("titulo", "Tus datos");
    model.addAttribute("producers", producer);
    return "producer/listAll";
}
```

```
<th sec:authorize="hasRole('ROLE_ADMIN')">Eliminar</th>
```

Funcional 10

Vista:

El nombre de usuario ya existe

Añadir Productor

Añadir Productor

Código:

```
UserAccount existe = userRepo.findByUsername(adminDto.getUsername());

if (existe != null) {
    String mensajeFlash = "El nombre de usuario ya existe";
    flash.addFlashAttribute("success", mensajeFlash);
    return "redirect:/admin/create";
}
```