//Apartado 1

**Para poner campos con assert:**

Ir a entitty(nombre de entitty)

**use** Symfony\Component\Validator\Constraints **as** Assert;

Y en los campos que pida el assert insertar la siguiente línea debajo de

***@ORM\Column****(name="escondite", type="string", length=255)****@Assert\NotBlank***

Abrir servidor:

php bin/console server:run

Crear base de datos:

php bin/console doctrine:database:create

Crear entidad:

php bin/console doctrine:generate:entity

Subir la tabla:

php bin/console doctrine:schema:update --force

Crear formulario:

php bin\console doctrine:generate:form AppBundle: (Nombre de entidad)

Crear CRUD(create, update, delete)

php bin/console generate:doctrine:crud AppBundle: (Nombre de la entidad)

\*Al generar el CRUD hay que cambiar el create de no a yes y el resto predeterminado

Paso siguiente, cortar el indexAction(con comentario) del (Nombre del entitty)Controller.php y pegarlo en el DefaultController.

En el default controller añadir el siguiente use:

use Sensio\Bundle\FrameworkExtraBundle\Configuration\Method;

//apartado 2

**User interface:**

Añadir esta línea a la clase de la nueva entidad

**implements** userInterface

y añadir este use

**use** Symfony\Component\Security\Core\User\UserInterface;

Para que funcione el UserInterface añadir estas líneas al final de la entity.

**public function** getPlainPassword()  
{  
 **return** $this->plainPassword;  
}  
  
**public function** setPlainPassword($password)  
{  
 $this->plainPassword = $password;  
}  
  
**public function** getSalt()  
{  
 // you may need a real salt depending on your encoder  
 // see section on salt below  
 **return null**;  
}  
  
**public function** getRoles()  
{  
 **return** ['ROLE\_USER'];  
}  
  
**public function** eraseCredentials()  
{  
}

Para poner un mínimo de caracteres y el plainPassword(repetir contraseña para verificar). Copiar esto reemplazando la parte de password.

*/\** ***@Assert\NotBlank*** *\** ***@Assert\Length****(  
 \* min = 5,  
 \* minMessage = "La contraseña debe tener mínimo 5 carácteres"  
 \* )  
 \*/***private** $plainPassword;  
  
*/\** ***@var*** *string  
 \*  
 \** ***@ORM\Column****(name="password", type="string", length=255)  
 \** ***@Assert\NotBlank*** *\*/***private** $password;

Para añadir un rol por defecto, insertar estas líneas debajo de las variables y arriba de los métodos (entre estos)

**public function** \_\_construct()  
{  
 $this->role = **array**('ROLE\_USER');  
}

Para configurar la parte de seguridad hay que poner las siguientes lineas en el archivo “security.yml” \***importante, las tabulaciones tienen que ser exactamente estas**

*# To get started with security, check out the documentation:  
# https://symfony.com/doc/current/security.html***security**:  
 **encoders**:  
 **AppBundle\Entity\Nombre\_entidad**:  
 **algorithm**: bcrypt  
  
 *# https://symfony.com/doc/current/security.html#b-configuring-how-users-are-loaded* **providers**:  
 **in\_memory**:  
 **memory**: ~  
 **our\_db\_provider**:  
 **entity**:  
 **class**: AppBundle: Nombre\_entidad  
 **property**: username  
  
 **firewalls**:  
 *# disables authentication for assets and the profiler, adapt it according to your needs* **dev**:  
 **pattern**: ^/(\_(profiler|wdt)|css|images|js)/  
 **security**: false  
  
 **main**:  
 **anonymous**: ~  
 **pattern**: ^/  
 **http\_basic**: ~  
 **provider**: our\_db\_provider  
 *# activate different ways to authenticate  
  
 # https://symfony.com/doc/current/security.html#a-configuring-how-your-users-will-authenticate  
 #http\_basic: ~  
  
 # https://symfony.com/doc/current/security/form\_login\_setup.html  
 #form\_login: ~*

Despues de hacer todos estos pasos creamos CRUD para justicia.

\*Revisar que esten bien los nombres\*

En este segundo entity “justicia” tenemos que ir a justiciaType y borrar el addRole.

Si se pide que se cambie el rol automatico cambiar en la entity el nombre del rol

**public function** \_\_construct()  
{  
 $this->role = **array**('ROLE\_RESTO');  
}

Para “plainPasword” hay que añadir estas líneas en “JusticiaType” dentro de build form:

->add('plainPassword', RepeatedType::**class**, [  
 'type' => PasswordType::**class**,  
 'first\_options' => ['label' => 'Contraseña'],  
 'second\_options' => ['label' => 'Repetir contraseña'],  
])

Para que no de error en PasswordType y el repeatedType añadimos las siguientes líneas

**use** Symfony\Component\Form\Extension\Core\Type\PasswordType;  
**use** Symfony\Component\Form\Extension\Core\Type\RepeatedType;

Para hacer el formulario de registro añadir en JusticiaController sustituyendo newAction.

*/\*\*  
 \* Creates a new justicium entity.  
 \*  
 \** ***@Route****("/new", name="justicia\_new")  
 \** ***@Method****({"GET", "POST"})  
 \*/*

**public function** registerAction(Request $request, UserPasswordEncoderInterface $passwordEncoder)  
{  
 // 1) build the form  
 $user = **new** Justicia();  
 $form = $this->createForm(JusticiaType::**class**, $user);  
  
 // 2) handle the submit (will only happen on POST)  
 $form->handleRequest($request);  
 **if** ($form->isSubmitted() && $form->isValid()) {  
  
 // 3) Encode the password (you could also do this via Doctrine listener)  
 $password = $passwordEncoder->encodePassword($user, $user->getPlainPassword());  
 $user->setPassword($password);  
  
 // 4) save the User!  
 $entityManager = $this->getDoctrine()->getManager();  
 $entityManager->persist($user);  
 $entityManager->flush();  
  
 // ... do any other work - like sending them an email, etc  
 // maybe set a "flash" success message for the user  
  
 **return** $this->redirectToRoute('justicia\_index');  
 }  
  
 **return** $this->render(  
 'justicia/new.html.twig',  
 ['form' => $form->createView()]  
 );  
}

Sustituyendo los correspondientes Users e incluyendo los paquetes …

**use** AppBundle\Form\JusticiaType;

**use** Symfony\Component\Security\Core\Encoder\UserPasswordEncoderInterface;

Para evitar errores quitamos que muestre el role.

//apartado 3

//seguridad

Empezamos añadiendo esta línea en security.yml justo debajo de anonymus en main firewalls

**form\_login**:  
 **login\_path**: login  
 **check\_path**: login

Despues nos creamos un securityController. Para hacerlo copiamos el default controller y lo pegamos poniendole el nombre de “SecurityController” y cambiamos en clase a SecurityController . Despues copiamos lo siguiente, se debe quedar así

**<?php  
  
namespace** AppBundle\Controller;  
  
**use** Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;  
**use** Symfony\Bundle\FrameworkBundle\Controller\Controller;  
**use** Symfony\Component\HttpFoundation\Request;  
**use** Sensio\Bundle\FrameworkExtraBundle\Configuration\Method;  
**use** Symfony\Component\Security\Http\Authentication\AuthenticationUtils;  
  
**class** SecurityController **extends** Controller  
{  
 */\*\*  
 \** ***@Route****("/login", name="login")  
 \*/* **public function** loginAction(AuthenticationUtils $authenticationUtils)  
 {  
 // get the login error if there is one  
 $error = $authenticationUtils->getLastAuthenticationError();  
  
 // last username entered by the user  
 $lastUsername = $authenticationUtils->getLastUsername();  
  
 **return** $this->render(login.html.twig', [  
 'last\_username' => $lastUsername,  
 'error' => $error,  
 ]);  
 }  
}

Tenemos que crear un nuevo twig (el de la ruta que hemos puesto antes) y lo dejamos así

{# Plantilla de home#}  
{% **extends** 'base.html.twig' %}  
  
{% **block** body %}  
 {% **if** error %}  
 <div>{{ error.messageKey|trans(error.messageData, 'security') }}</div>  
 {% **endif** %}  
  
 <form action="{{ path('login') }}" method="post">  
 <label for="username">Username:</label>  
 <input type="text" id="username" name="\_username" value="{{ last\_username }}" />  
  
 <label for="password">Password:</label>  
 <input type="password" id="password" name="\_password" />  
  
 <button type="submit">login</button>  
 </form>  
{% **endblock** %}

Posteriormente, en el security.yml borramos todo y pegamos esto:

*# To get started with security, check out the documentation:  
# https://symfony.com/doc/current/security.html***security**:  
 **encoders**:  
 **AppBundle\Entity\Justicia**: bcrypt  
  
 *# https://symfony.com/doc/current/security.html#b-configuring-how-users-are-loaded* **providers**:  
 **in\_memory**:  
 **memory**: ~  
 **my\_db\_provider**:  
 **entity**:  
 **class**: AppBundle:Justicia  
 **property**: username  
 **firewalls**:  
 *# disables authentication for assets and the profiler, adapt it according to your needs* **dev**:  
 **pattern**: ^/((profiler|wdt)|css|images|js)/  
 **security**: false  
  
 **main**:  
 **anonymous**: ~  
 *# activate different ways to authenticate  
  
 # https://symfony.com/doc/current/security.html#a-configuring-how-your-users-will-authenticate  
 #http\_basic: ~  
  
 # https://symfony.com/doc/current/security/form\_login\_setup.html  
 #form\_login: ~* **pattern**: ^/  
 **provider**: my\_db\_provider  
 *#http\_basic: ~* **form\_login**:  
 **login\_path**: /login  
 **check\_path**: /login  
 **logout**:  
 **path**: /logout  
 **target**: /  
  
 **access\_control**:  
 - { **path**: ^/login, **roles**: IS\_AUTHENTICATED\_ANONYMOUSLY }  
 - { **path**: ^/, **roles**: ROLE\_RESTO }  
 - { **path**: ^/justicia, **roles**: ROLE\_BATMAN }

Revisar el path de security.yml

En el routing.yml hay que copiar esto:

**app**:  
 **resource**: '@AppBundle/Controller/'  
 **type**: annotation  
 *# app/config/routing.yml***logout**:  
 **path**: /logout

batman es el rol admin y resto es el rol normal