Pour informations : voici mes coordonnées [mathieu.roux5sur5@outlook.com](mailto:mathieu.roux5sur5@outlook.com) pour Skype

Nous pouvons aussi mettre un partage google drive en commun pour partager les questions et hangout à vous de voir ☺

Je m’excuse d’avance des potentiels faute d’orthographe, mais j’ai écrit ça un peu tard.

Le projet suivra sur un lien drive car trop gros pour un mail.

User registration il y a deux choix en symfony :

* Le faire vous (mon conseil) (car ça va vous apprendre à maitriser tous les composants de Symfony)
* FOSUserBundle

Je vais vous expliquez comment le faire de votre côté.

Par gérer un formulaire d’enregistrement des users il faut pour ça avoir ce qu’on appelle une entité (entity)

Une entité c’est un simplement le model dans le model MVC (il représente votre objet au sens développement)

Elle fera le lien avec une table de la base de données. Donc ici nous allons créer un entité user qui fera donc un lien avec une table user (souvent sur symfony elle est préfixé du nom de la app par défaut c’est app ☺) = app\_user

php app/console generate:doctrine:entity

Derrière il va vous demander le shortcut : mettez par exemple AppBundle:User (avant de faire pensez à votre table user en BDD c’est-à-dire les champs que vous allez avoir)

En générale une table user c’est sous cette forme :

* Username
* Password
* Lastname
* Firstname
* Email
* isActive
* LastMove
* ETC ETC ETC ETC à vous de voir les besoins

Une fois votre entité User créé.

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

Forcer la mise à jour de la base de données pour que vous entité soit créé en base avec les champs entrés en amont dans l’entité.

Modifier la pour ajouter un bundle de Symfony UserInterface elle aura pour but d’avoir les principales fonctionnalités de bases en sécurité de symfony.

Voici la tete de mon user.php ☺

*// src/AppBundle/Entity/User.php*

namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

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

*/\*\**

*\* @ORM\Table(name="app\_users")*

*\* @ORM\Entity(repositoryClass="AppBundle\Entity\UserRepository")*

*\*/*

class User implements UserInterface, \Serializable

{

*/\*\**

*\* @ORM\Column(type="integer")*

*\* @ORM\Id*

*\* @ORM\GeneratedValue(strategy="AUTO")*

*\*/*

private $id;

*/\*\**

*\* @ORM\Column(type="string", length=25, unique=true)*

*\*/*

private $username;

*/\*\**

*\* @ORM\Column(type="string", length=64)*

*\*/*

private $password;

*/\*\**

*\* @ORM\Column(type="string", length=60, unique=true)*

*\*/*

private $email;

*/\*\**

*\* @ORM\Column(name="is\_active", type="boolean")*

*\*/*

private $isActive;

public function \_\_construct()

{

$this->isActive = true;

*// may not be needed, see section on salt below*

*// $this->salt = md5(uniqid(null, true));*

}

public function getUsername()

{

return $this->username;

}

public function getSalt()

{

*// you \*may\* need a real salt depending on your encoder*

*// see section on salt below*

return null;

}

public function getPassword()

{

return $this->password;

}

public function getRoles()

{

return array('ROLE\_USER');

}

public function eraseCredentials()

{

}

*/\*\* @see \Serializable::serialize() \*/*

public function serialize()

{

return serialize(array(

$this->id,

$this->username,

$this->password,

*// see section on salt below*

*// $this->salt,*

));

}

*/\*\* @see \Serializable::unserialize() \*/*

public function unserialize($serialized)

{

list (

$this->id,

$this->username,

$this->password,

*// see section on salt below*

*// $this->salt*

) = unserialize($serialized);

}

}

Ensuite nous modifions le security.yml (app/config/security.yml)

security:

encoders:

AppBundle\Entity\User:

algorithm: bcrypt

*# ...*

providers:

our\_db\_provider:

entity:

class: AppBundle:User

property: username

firewalls:

main:

pattern: ^/

http\_basic: ~

provider: our\_db\_provider

Attention on ajoute on ne supprime pas ce qui existe sinon Crouche le symfony ☺

**Et c’est là que ça devient marrant :D**

Maintenant que nous avons fait notre belle entité et j’espère ne pas vous avoir perdu nous allons faire un Form liée à l’entité User.

*// src/AppBundle/Form/UserType.php*

namespace AppBundle\Form;

use Symfony\Component\Form\AbstractType;

use Symfony\Component\Form\FormBuilderInterface;

use Symfony\Component\OptionsResolver\OptionsResolver;

use Symfony\Component\Form\Extension\Core\Type\EmailType;

use Symfony\Component\Form\Extension\Core\Type\TextType;

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

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

class UserType extends AbstractType

{

public function buildForm(FormBuilderInterface $builder, array $options)

{

$builder

->add('email', EmailType::class)

->add('username', TextType::class)

->add('plainPassword', RepeatedType::class, array(

'type' => PasswordType::class,

'first\_options' => array('label' => 'Password'),

'second\_options' => array('label' => 'Repeat Password'),

)

);

}

public function configureOptions(OptionsResolver $resolver)

{

$resolver->setDefaults(array(

'data\_class' => 'AppBundle\Entity\User',

));

}

Normalement vous devriez pouvoir gérer la suite ☺

Un autre point j’adore le cookbook de Symfony il est tout en anglais mais il est explicite et il vous explique avec des liens vers l’api le fonctionnement de symfony. Je vous le conseil fortement ☺

<http://symfony.com/doc/current/cookbook/index.html>

<http://symfony.com/doc/current/cookbook/doctrine/index.html> ==> je vous aides ça se passe de se côté ☺