Installation de composants PHP

```
composer require symfony/twig-bridge
composer require symfony/security
```

Modification du composer.json

```
"autoload": {
    "psr-4": {
        "App\\": "src",
        "Tests\\": "tests"
    }
}
```

Il faut ensuite exécuter la commande qui va générer l'*autoload* des classes définies dans le *composer.json* :

```
composer dump-autoload
```

Services

```
$app->register(new Silex\Provider\DoctrineServiceProvider());

$app['db.options'] = array(
    'driver'   => 'pdo_mysql',
    'charset'  => 'utf8',
    'host'     => 'localhost',
    'port'     => '3306',
    'dbname'   => 'gecanin',
    'user'     => 'root',
    'password' => '',
);

$app->register(new Silex\Provider\SessionServiceProvider());

$app->register(new Silex\Provider\SecurityServiceProvider(), array(
  'security.firewalls' => array(
    'secured' => array(
      'pattern'   => '^/',
      'anonymous' => true,
      'logout'    => true,
      'form'      => array('login_path' => '/login', 'check_path' => '/
    login_check'),
      'users'     => function () use ($app) {
          return new UserDAO($app['db']);
      },
    ),
  ),
));

$app['dao.user'] = function ($app) {
    return new UserDAO($app['db']);
};
```

Ajout de champs dans la table User

Ajouter les champs suivants :
— user_password varchar(88)
— user_salt varchar(23)
— user_role varchar(50)

Classes à créer

Fichier dans le dossier *DAO* la classe *DAO*

```php
namespace App\DAO;

use Doctrine\DBAL\Connection;

abstract class DAO
{
    private $db;

    public function __construct(Connection $db)
    {
        $this->db = $db;
    }

    protected function getDb()
    {
        return $this->db;
    }

    abstract protected function buildDomainObject(array $row);
}
```

Fichier dans le dossier *DAO* la classe *UserDAO*

```php
namespace App\DAO;

use Symfony\Component\Security\Core\User\UserInterface;
use Symfony\Component\Security\Core\User\UserProviderInterface;
use Symfony\Component\Security\Core\Exception\UsernameNotFoundException;
use Symfony\Component\Security\Core\Exception\UnsupportedUserException;
use App\Domain\User;

class UserDAO extends DAO implements UserProviderInterface
{
    public function loadUserByUsername($username)
    {
        $sql = 'SELECT user_id, user_name, user_password, user_salt, user_role '
            . 'FROM user '
            . 'WHERE user_name = ?';
        $row = $this->getDb()->fetchAssoc($sql, array($username));

        if ($row) {
            return $this->buildDomainObject($row);
        } else {
            throw new UsernameNotFoundException(sprintf('User "%s" not found.',
    $username));
        }
    }
```

```
    public function refreshUser(UserInterface $user)
    {
        $class = get_class($user);
        if (!$this->supportsClass($class)) {
            throw new UnsupportedUserException(sprintf('Instances of "%s" are
    not supported.', $class));
        }
        return $this->loadUserByUsername($user->getUsername());
    }

    public function supportsClass($class)
    {
        return 'App\Domain\User' === $class;
    }

    protected function buildDomainObject(array $row)
    {
        $user = new User();
        /*
        appelle des methodes set()
        */
        return $user;
    }
}
```

```
namespace App\Domain;

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

class User implements UserInterface
{
    private $salt;

    private $role;

    public function getUsername()
    {
        return $this->username;
    }

    public function getPassword()
    {
        return $this->password;
    }

    public function getSalt()
    {
        return $this->salt;
    }

    public function getRoles()
    {
        return array($this->getRole());
    }
    public function eraseCredentials()
    {
        // Nothing to do here
    }
}
```

Création d'un fichier pour les routes

Créer un fichier *routes.php* dans le dossier *app*

```
<?php

use Symfony\Component\HttpFoundation\Request;

// Home page
$app->get('/', "App\Controller\HomeController::indexAction")
    ->bind('home');

// Login form
$app->get('/login', "App\Controller\HomeController::loginAction")
    ->bind('login');


// le fichier \logout est gere par Silex mais peut etre redefinie
```

Création du contrôleur

```
namespace App\Controller;

use Silex\Application;
use Symfony\Component\HttpFoundation\Request;

class HomeController
{
    public function indexAction(Application $app)
    {
        return $app['twig']->render('index.twig');
    }

    public function loginAction(Request $request, Application $app)
    {
        return $app['twig']->render('login.twig', array(
            'error'         => $app['security.last_error']($request),
            'last_username' => $app['session']->get('_security.last_username')
    ,
        ));
    }
}
```

Création du formulaire des fichiers TWIG

Fichier *login.twig*

```
{% if error %}
    <div>
        <strong>L'authentification a echouee</strong>
    </div>
{% endif %}
<div>
    <form action="{{ path('login_check') }}" method="post">
        <div class="form-group">
```

```
            <input type="text" name="_username" value="{{ last_username }}"
    placeholder="Entrez votre identifiant" required autofocus>
        </div>
        <div>
            <input type="password" name="_password" placeholder="Entrez votre
    mot de passe" required>
        </div>
        <div>
            <button type="submit">Login</button>
        </div>
    </form>
</div>
```

Fichier *index.twig*

```
{% block content %}
    {% if is_granted('IS_AUTHENTICATED_FULLY') %}
        <div class="container">
            Je suis logger
        </div>
    {% else %}
        je suis PAS logger
    {% endif %}
{% endblock content %}
```

Pour enregistrer le mot de passe il faut utiliser

```
$encoder = $app['security.encoder_factory']->getEncoder($user);

$password = $encoder->encodePassword('azerti', $user->getSalt());
```