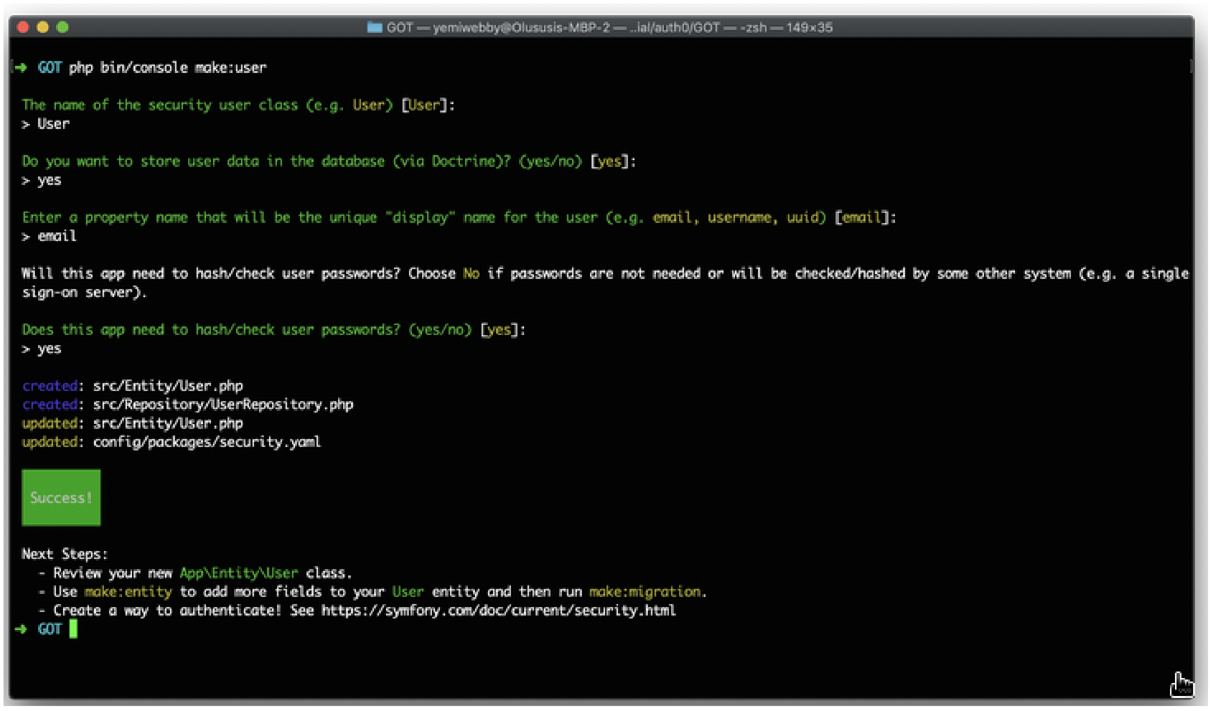
Symfony authentification

```
symfony console make:user
```



```
GOT — yemiwebby@Olususis-MBP-2 — ..ial/auth0/GOT — -zsh — 149×35

➜ GOT php bin/console make:user

 The name of the security user class (e.g. User) [User]:
 > User

 Do you want to store user data in the database (via Doctrine)? (yes/no) [yes]:
 > yes

 Enter a property name that will be the unique "display" name for the user (e.g. email, username, uuid) [email]:
 > email

 Will this app need to hash/check user passwords? Choose No if passwords are not needed or will be checked/hashed by some other system (e.g. a single
 sign-on server).

 Does this app need to hash/check user passwords? (yes/no) [yes]:
 > yes

 created: src/Entity/User.php
 created: src/Repository/UserRepository.php
 updated: src/Entity/User.php
 updated: config/packages/security.yaml


  Success!


 Next Steps:
   - Review your new App\Entity\User class.
   - Use make:entity to add more fields to your User entity and then run make:migration.
   - Create a way to authenticate! See https://symfony.com/doc/current/security.html
➜ GOT
```

update de la page:

```php
<?php

namespace App\Entity;

use App\Repository\UserRepository;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
use Symfony\Component\Security\Core\User\UserInterface;

/**
 * @ORM\Entity(repositoryClass=UserRepository::class)
 * @ORM\Table(name="`user`")
 */
class User implements UserInterface
{
    /**
     * @ORM\Id()
     * @ORM\GeneratedValue()
     * @ORM\Column(type="integer")
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=180, unique=true)
     */
    private $email;

    /**
     * @ORM\Column(type="json")
     */
    private $roles = [];

    /**
     * @var string The hashed password
     * @ORM\Column(type="string")
     */
    private $password;

    /**
     * @ORM\Column(type="string", length=255)
     */
    private $name;

    public function getId(): ?int
    {
        return $this->id;
    }

    public function getEmail(): ?string
    {
        return $this->email;
    }

    public function setEmail(string $email): self
    {
        $this->email = $email;
        return $this;
    }

    /**
     * A visual identifier that represents this user.
     *
     * @see UserInterface
     */
    public function getUsername(): string
    {
        return (string) $this->email;
    }

    /**
     * @see UserInterface
     */
    public function getRoles(): array
    {
        $roles = $this->roles;
        // guarantee every user at least has ROLE_USER
        $roles[] = 'ROLE_USER';
        return array_unique($roles);
    }

    public function setRoles(array $roles): self
    {
        $this->roles = $roles;
        return $this;
    }

    /**
     * @see UserInterface
     */
    public function getPassword(): string
    {
        return (string) $this->password;
    }

    public function setPassword(string $password): self
    {
        $this->password = $password;
        return $this;
    }

    /**
     * @see UserInterface
     */
    public function getSalt()
    {
        // not needed when using the "bcrypt" algorithm in security.yaml
    }

    /**
     * @see UserInterface
     */
    public function eraseCredentials()
    {
        // If you store any temporary, sensitive data on the user, clear it
here
        // $this->plainPassword = null;
    }

    public function getName(): ?string
    {
        return $this->name;
    }

    public function setName(string $name): self
    {
        $this->name = $name;
        return $this;
    }
}
```

```
symfony console make:controller ListController
```

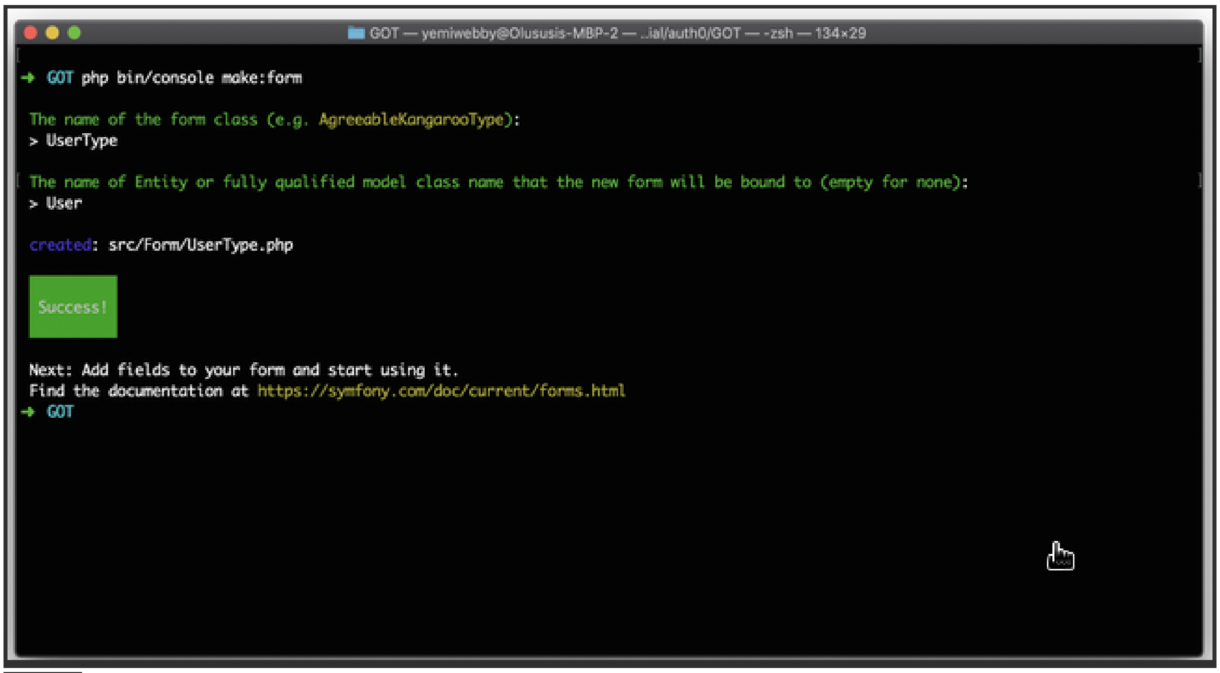
```php
<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\Routing\Annotation\Route;

class ListController extends AbstractController
{
    /**
     * @Route("/list", name="list")
     */
    public function index(Request $request)
    {
        $companies = [
            'Apple' => '$1.16 trillion USD',
            'Samsung' => '$298.68 billion USD',
            'Microsoft' => '$1.10 trillion USD',
            'Alphabet' => '$878.48 billion USD',
            'Intel Corporation' => '$245.82 billion USD',
            'IBM' => '$120.03 billion USD',
            'Facebook' => '$552.39 billion USD',
            'Hon Hai Precision' => '$38.72 billion USD',
            'Tencent' => '$3.02 trillion USD',
            'Oracle' => '$180.54 billion USD',
        ];

        return $this->render('list/index.html.twig', [
            'companies' => $companies,
        ]);
    }
}
```

```
symfony console make:controller RegistrationController
```

```php
<?php

namespace App\Controller;

use App\Entity\User;
use App\Form\UserType;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\Security\Core\Encoder\UserPasswordEncoderInterface;

class RegistrationController extends AbstractController
{
   private $passwordEncoder;

   public function __construct(UserPasswordEncoderInterface $passwordEncoder)
   {
       $this->passwordEncoder = $passwordEncoder;
   }

   /**
    * @Route("/registration", name="registration")
    */
   public function index(Request $request)
   {
       $user = new User();

       $form = $this->createForm(UserType::class, $user);

       $form->handleRequest($request);

       if ($form->isSubmitted() && $form->isValid()) {
           // Encode the new users password
           $user->setPassword($this->passwordEncoder->encodePassword($user,
$user->getPassword()));

           // Set their role
           $user->setRoles(['ROLE_USER']);

           // Save
           $em = $this->getDoctrine()->getManager();
           $em->persist($user);
           $em->flush();

           return $this->redirectToRoute('app_login');
       }

       return $this->render('registration/index.html.twig', [
           'form' => $form->createView(),
       ]);
   }
}
```

```
console make:controller SecurityController
```

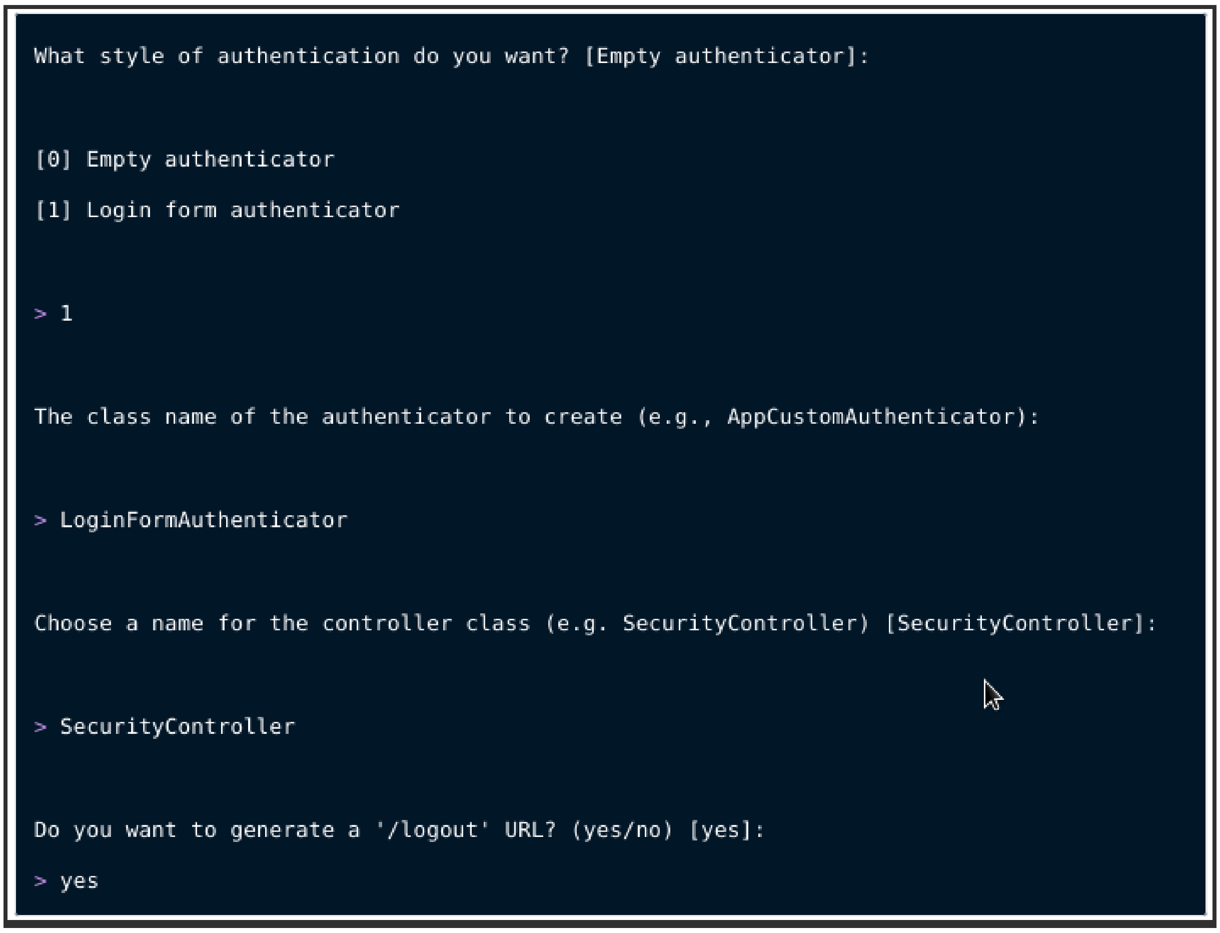
```
symfony console make:form
```


GOT — yemiwebby@Olususis-MBP-2 — ..ial/auth0/GOT — -zsh — 134×29

```
➜  GOT php bin/console make:form

 The name of the form class (e.g. AgreeableKangarooType):
 > UserType

 The name of Entity or fully qualified model class name that the new form will be bound to (empty for none):
 > User

 created: src/Form/UserType.php

 Success!

 Next: Add fields to your form and start using it.
 Find the documentation at https://symfony.com/doc/current/forms.html
➜  GOT
```

```php
<?php

namespace App\Form;

use App\Entity\User;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\EmailType;
use Symfony\Component\Form\Extension\Core\Type\PasswordType;
use Symfony\Component\Form\Extension\Core\Type\RepeatedType;
use Symfony\Component\Form\Extension\Core\Type\TextType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;

class UserType extends AbstractType
{
   public function buildForm(FormBuilderInterface $builder, array $options)
   {
       $builder
           ->add('email', EmailType::class)
           ->add('name', TextType::class)
           ->add('password', RepeatedType::class, [
               'type' => PasswordType::class,
               'first_options' => ['label' => 'Password'],
               'second_options' => ['label' => 'Confirm Password']
           ])
       ;
   }

   public function configureOptions(OptionsResolver $resolver)
   {
       $resolver->setDefaults([
           'data_class' => User::class,
        ]);
    }
}
```

```
console make:auth
```



```
What style of authentication do you want? [Empty authenticator]:


[0] Empty authenticator

[1] Login form authenticator


> 1


The class name of the authenticator to create (e.g., AppCustomAuthenticator):


> LoginFormAuthenticator


Choose a name for the controller class (e.g. SecurityController) [SecurityController]:


> SecurityController


Do you want to generate a '/logout' URL? (yes/no) [yes]:

> yes
```

update de la page src/Controller/SecurityController.php

```php
<?php


namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\Security\Http\Authentication\AuthenticationUtils;

class SecurityController extends AbstractController
```

```
{
    /**
     * @Route("/", name="app_login")
     */
    public function login(AuthenticationUtils $authenticationUtils): Response
    {
        // get the login error if there is one
        $error = $authenticationUtils->getLastAuthenticationError();
        // last username entered by the user
        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render('security/login.html.twig', ['last_username' =>
$lastUsername, 'error' => $error]);
    }

    /**
     * @Route("/logout", name="app_logout")
     */
    public function logout()
    {
        throw new \Exception('This method can be blank - it will be
intercepted by the logout key on your firewall');
    }
}
```

Suivre les étapes de la création de la base de donnée :

- **Dans le fichier .env :**

```
DATABASE_URL="postgresql://piota:piota.15@servbd:5432/TPAuthAlistair?serverVe
rsion=8&charset=utf8"
```

Commande de création de la base de donnée dans le terminal :

```
symfony console doctrine:database:create
```

Ensuite création de la base de donnée avec le nom, host, user, password.

Ouvrir le fichier config/packages/security.yaml et suivre la configuration

```
security:
  encoders:
    App\Entity\User:
      algorithm: auto
  #
https://symfony.com/doc/current/security.html#where-do-users-come-from-user-provi
ders
```

```
 providers:
   # used to reload user from session & other features (e.g. switch_user)
   app_user_provider:
     entity:
       class: App\Entity\User
       property: email
 firewalls:
   dev:
     pattern: ^/(_(profiler|wdt)|css|images|js)/
     security: false
   main:
     anonymous: lazy
     provider: app_user_provider
     guard:
       authenticators:
         - App\Security\LoginFormAuthenticator
     logout:
       path: app_logout
       target: /
```

---

le fichier final ci-dessous :

```
security:
   enable_authenticator_manager: true
   encoders:
       App\Entity\User:
           algorithm: auto
   #
https://symfony.com/doc/current/security.html#where-do-users-come-from-user
-providers
   providers:
       # used to reload user from session & other features (e.g.
switch_user)
       app_user_provider:
           entity:
               class: App\Entity\User
               property: email
   firewalls:
       dev:
           pattern: ^/(_(profiler|wdt)|css|images|js)/
           security: false
       main:
           lazy: true
           provider: app_user_provider
           custom_authenticators:
               - App\Security\LoginFormAuthenticator
           logout:
               path: app_logout
               target: /
when@test:
   security:
       password_hashers:
           # By default, password hashers are resource intensive and take
time. This is
           # important to generate secure password hashes. In tests
however, secure hashes
           # are not important, waste resources and increase test times.
The following
           # reduces the work factor to the lowest possible values.

Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface:
               algorithm: auto
               cost: 4 # Lowest possible value for bcrypt
               time_cost: 3 # Lowest possible value for argon
               memory_cost: 10 # Lowest possible value for argon
```

Dans la template => templates/list/index.html.twig, configuration de la façon suivante :

```
{# templates/list/index.html.twig #} {% extends 'base.html.twig' %} {%
block
   body %}
   <div class="container">
       <div class="row">
           <div class="col-md-12">
               <div class="card bg-light mb-3 mt-3">
                   <div class="card-body">
                       <div class="card-header">List of top technology
companies</div>
                       {% if app.user %}
                           <table class="table">
                               <tr>
                                   <th>Company Name</th>
                                   <th>Market Value</th>
                               </tr>
                               {% for key, item in companies %}
                                   <tr>
                                       <td>{{ key }}</td>
                                       <td>{{ item }}</td>
                                   </tr>
                               {% endfor %}
                           </table>
                       {% endif %}
                   </div>
               </div>
               {% if not app.user %}
                   <a href="{{ path('app_login') }}" class="btn btn-info">
                       You need to login to see the list 😜😜 >></a>
               {% endif %}
           </div>
       </div>
```

```
    </div>
{% endblock %}
```

Dans la template =>templates/security/login.html.twig configuration de la façon suivante :

```
{# templates/security/login.html.twig #} {% extends 'base.html.twig' %} {%
block
    title %}Log in!{% endblock %} {% block body %}
    <div class="container">
        <div class="row">
            <div class="col-md-10 ml-md-auto">
                <div class="">
                    <div class="card bg-light mb-3 mt-5" style="width:
800px;">
                        <div class="card-body">
                            <form class="form-horizontal" role="form"
method="post">
                                    {% if (error) %}
                                    <div class="alert alert-danger">
                                        {{
error.messageKey|trans(error.messageData, 'security') }}
                                    </div>

                                    {% endif %}
                                    {% if (error) %}
                                        <div class="mb-3">
                                            You are logged in as {{
app.user.userIdentifier }},
                                            <a href="{{ path('app_logout')
}}">Logout</a>
                                        </div>
                                    {% endif %}

                                        <div class="card-header mb-3">Please
sign in</div>
                                        <div class="form-group">
                                            <label for="email"
class="col-md-4 control-label"
                                            >E-Mail Address</label
                                            >
                                            <div class="col-md-12">
                                                <input
                                                        id="inputEmail"
                                                        type="email"
                                                        class="form-control"
                                                        name="email"
```

```
                                                       value="{{
last_username }}"
                                                       required
                                                       autofocus
                                               />
                                           </div>
                                       </div>
                                       <div class="form-group">
                                           <label for="password"
class="col-md-4 control-label"
                                           >Password</label
                                           >
                                           <div class="col-md-12">
                                               <input
                                                       id="inputPassword"
                                                       type="password"
                                                       class="form-control"
                                                       name="password"
                                                       required
                                               />
                                           </div>
                                       </div>
                                       <input
                                               type="hidden"
                                               name="_csrf_token"
                                               value="{{
csrf_token('authenticate') }}"
                                       />
                                       <div class="form-group">
                                           <div class="col-md-12">
                                               <button type="submit"
class="btn btn-primary">
                                                   <i class="fa fa-btn
fa-sign-in"></i> Login
                                               </button>
                                           </div>
                                       </div>
                           </form>
                       </div>
                   </div>
               </div>
           </div>
       </div>
   </div>
{% endblock %}
```

Dans la template =>templates/registration/index.html.twig configuration de la façon suivante :

```
{# templates/registration/index.html.twig #} {% extends 'base.html.twig' %}
{% block body %}
   <div class="container">
       <div class="row">
           <div class="col-md-10 ml-md-auto">
               <div class="card bg-light mb-3 mt-5" style="width: 800px">
                   <div class="card-body">
                       <div class="card-header mb-3">Registration
Form</div>
                       {{ form_start(form) }}
                       <div class="form_group">
                           <div class="col-md-12 mb-3">
                               {{ form_row(form.name, {'attr': {'class':
'form-control'}}) }}
                           </div>
                       </div>
                       <div class="form_group">
                           <div class="col-md-12 mb-3">
                               {{ form_row(form.email, {'attr': {'class':
'form-control'}}) }}
                           </div>
                       </div>
                       <div class="form_group">
                           <div class="col-md-12 mb-3">
                               {{ form_row(form.password.first, {'attr':
{'class':
                                   'form-control'}}) }}
                           </div>
                       </div>
                       <div class="form_group">
                           <div class="col-md-12 mb-3">
                               {{ form_row(form.password.second, {'attr':
{'class':
                                   'form-control'}}) }}
                           </div>
                       </div>
                       <div class="form-group">
                           <div class="col-md-8 col-md-offset-4"
style="margin-top:5px;">
                               <button type="submit" class="btn
btn-primary">
                                   <i class="fa fa-btn fa-user"></i>
Register
                               </button>
                           </div>
                       </div>
                       {{ form_end(form) }}
                   </div>
               </div>
           </div>
       </div>
   </div>
{% endblock %}
```

update du fichier base.html.twig :

```
{# templates/base.html.twig #}
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8" />
   <title>{% block title %}Welcome!{% endblock %}</title>
   <link
           rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css
"
   />
   {% block stylesheets %}{% endblock %}
</head>
<body>
<nav
       class="navbar navbar-expand-lg navbar-light bg-light"
       style="height: 70px;"
>
   <a class="navbar-brand" href="#">Symfony</a>
   <div class="collapse navbar-collapse" id="navbarSupportedContent"></div>
   <ul class="nav navbar-nav navbar-right">
       {% if app.user %}
           <li><a class="nav-link" href="{{ path('list') }}">View
List</a></li>
           <li><a class="nav-link" href="{{ path('app_logout')
}}">Logout</a></li>

       {% endif %}

       {% if not app.user %}

           <li><a class="nav-link" href="{{ path('app_login')
}}">Login</a></li>
           <li>
               <a class="nav-link" href="{{ path('registration')
}}">Register</a>
           </li>

       {% endif %}
   </ul>
</nav>
{% block body %}{% endblock %} {% block javascripts %}{% endblock %}
</body>
</html>
```

page LoginFormAuthentificator :

```
<?php

namespace App\Security;

use Symfony\Component\HttpFoundation\RedirectResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Generator\UrlGeneratorInterface;
use Symfony\Component\Security\Core\Authentication\Token\TokenInterface;
use Symfony\Component\Security\Core\Security;
use
Symfony\Component\Security\Http\Authenticator\AbstractLoginFormAuthenticato
r;
use
Symfony\Component\Security\Http\Authenticator\Passport\Badge\CsrfTokenBadge
;
use Symfony\Component\Security\Http\Authenticator\Passport\Badge\UserBadge;
use
Symfony\Component\Security\Http\Authenticator\Passport\Credentials\Password
Credentials;
use Symfony\Component\Security\Http\Authenticator\Passport\Passport;
use Symfony\Component\Security\Http\Util\TargetPathTrait;

class LoginFormAuthenticator extends AbstractLoginFormAuthenticator
{
    use TargetPathTrait;

    public const LOGIN_ROUTE = 'app_login';

    private UrlGeneratorInterface $urlGenerator;

    public function __construct(UrlGeneratorInterface $urlGenerator)
    {
        $this->urlGenerator = $urlGenerator;
    }

    public function authenticate(Request $request): Passport
    {
        $email = $request->request->get('email', '');

        $request->getSession()->set(Security::LAST_USERNAME, $email);

        return new Passport(
            new UserBadge($email),
            new PasswordCredentials($request->request->get('password', '')),
            [
                new CsrfTokenBadge('authenticate',
$request->request->get('_csrf_token')),
            ]
        );
    }

    public function onAuthenticationSuccess(Request $request, TokenInterface
$token, string $firewallName): ?Response
    {
        if ($targetPath = $this->getTargetPath($request->getSession(),
$firewallName)) {
            return new RedirectResponse($targetPath);
        }

        // For example:
        return new RedirectResponse($this->urlGenerator->generate('list'));
        throw new \Exception('TODO: provide a valid redirect inside
'.__FILE__);
    }

    protected function getLoginUrl(Request $request): string
    {
        return $this->urlGenerator->generate(self::LOGIN_ROUTE);
    }
}
```