Rappel : installer composer, symfony-cli et un simulateur d’apache type xamp, wamp ou laragon.

<https://getcomposer.org/download/>

<https://www.devenir-webmaster.com/V2/TUTO/CHAPITRE/SYMFONY/01-symfony-cli/>

<https://wampserver.aviatechno.net/>

# Installer le projet symfony

Aller dans la symfony doc : [www.symfony.com/doc/current/setup.html](http://www.symfony.com/doc/current/setup.html)

Dans le paragraphe Creating Symfony Applications , copiez:

symfony new my\_project\_directory --version="7.1.\*" --webapp

Lancer dans un terminal powershell, à la racine du dossier où le projet sera installé (exemple : c:\laragon\www ou c:\wamp64\www) en changeant le nom *my\_project\_directory* par votre nom de projet.

Rentrer ensuite dans ce répertoire terminal-> cd monprojet et taper code . (*code* espace point)

Le projet doit s’ouvrir dans vscode.

# Installer Bootstrap, librairie de css

### Installer Node.js et npm

<https://nodejs.org/fr>

### Installer Webpack Encore :

composer require symfony/webpack-encore-bundle

npm install

### Installer Bootstrap via npm

npm install bootstrap

### Configurer Webpack Encore

Ouvrez ou créez un fichier webpack.config.js à la racine de votre projet Symfony

const Encore = require("@symfony/webpack-encore");

Encore

  // the project directory where all compiled assets will be stored

  .setOutputPath("public/build/")

  // the public path used by the web server to access the previous directory

  .setPublicPath("/build")

  // only needed for CDN's or sub-directory deploy

  //.setManifestKeyPrefix('build/')

  /\*

   \* ENTRY CONFIG

   \*

   \* Each entry will result in one JavaScript file (e.g. app.js)

   \* and one CSS file (e.g. app.css) if your JavaScript imports CSS.

   \*/

  .addEntry("app", "./assets/app.js")

  // When enabled, Webpack "splits" your files into smaller pieces for greater optimization.

  .splitEntryChunks()

  // will require an extra script tag for runtime.js

  // but, you probably want this, unless you're building a single-page app

  .enableSingleRuntimeChunk()

  /\*

   \* FEATURE CONFIG

   \*

   \* Enable & configure other features below. For a full

   \* list of features, see:

   \* https://symfony.com/doc/current/frontend.html#adding-more-features

   \*/

  .cleanupOutputBeforeBuild()

  .enableBuildNotifications()

  .enableSourceMaps(!Encore.isProduction())

  // enables hashed filenames (e.g. app.abc123.css)

  .enableVersioning(Encore.isProduction())

  // enables @babel/preset-env polyfills

  .configureBabel(() => {}, {

    useBuiltIns: "usage",

    corejs: 3,

  })

  // enables Sass/SCSS support

  .enableSassLoader();

// uncomment if you use TypeScript

//.enableTypeScriptLoader()

// uncomment to get integrity="..." attributes on your script & link tags

// requires WebpackEncoreBundle 1.4 or higher

//.enableIntegrityHashes(Encore.isProduction())

// uncomment if you're having problems with a jQuery plugin

//.autoProvidejQuery()

// uncomment if you use API Platform Admin (composer req api-admin)

//.enableReactPreset()

//.addEntry('admin', './assets/js/admin.js')

module.exports = Encore.getWebpackConfig();

### Inclure Bootstrap dans votre fichier principal JavaScript

// assets/app.js

import 'bootstrap/dist/css/bootstrap.min.css';

import 'bootstrap';

// Votre code JavaScript ici

### **Compiler les assets** :

npm run dev

pour compiler en production :

npm run build

### **Inclure les assets compilés dans vos templates Twig**

{# templates/base.html.twig #}

<!DOCTYPE html>

<html>

    <head>

        {# ... #}

        {{ encore\_entry\_link\_tags('app') }}

        {# ... #}

    </head>

    <body>

        {# ... #}

        {{ encore\_entry\_script\_tags('app') }}

        {# ... #}

    </body>

</html>

# Pour le scss

Il suffit de rendre utile le enableSassLoader() dans webpack.config.js

# Première page Home :

Avec la cli symfony , nous pouvons créer notre premier controller Home :

symfony console make:controller

Pour voir notre application : symfony serve

Nous allons modifier le controller pour le faire apparaitre en première page d’accueil.

<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\Routing\Attribute\Route;

class HomeController extends AbstractController

{

    #[Route('/', name: 'app\_home')]

    public function index(): Response

    {

        return $this->render('home/index.html.twig');

    }

}

La route devient « / » et on enlève les paramètres passés à la view.

On va travailler à partir d’une page déjà faite en html dans templates/home créer un index.html

Collez :

<!doctype html>

<html lang="en" data-bs-theme="auto">

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>La Boutique Française - 100% Made in France</title>

    <meta name="description" content="La boutique spécialisée dans la fabrication Made In France">

    <link rel="stylesheet" href="/assets/css/style.css">

    <link href="/assets/css/bootstrap.min.css" rel="stylesheet">

    <!-- Custom styles for this template -->

    <link href="/assets/css/carousel.css" rel="stylesheet">

    <link href="/assets/css/custom.css" rel="stylesheet">

    <script src="https://unpkg.com/feather-icons"></script>

</head>

<body>

<header data-bs-theme="dark">

    <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">

        <div class="container-fluid">

            <a class="navbar-brand" href="#">La Boutique Française</a>

            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">

                <span class="navbar-toggler-icon"></span>

            </button>

            <div class="collapse navbar-collapse" id="navbarCollapse">

                <ul class="navbar-nav me-auto mb-2 mb-md-0">

                    <li class="nav-item">

                        <a class="nav-link active" aria-current="page" href="#">Home</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link" href="#">Link</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link" href="#">Link</a>

                    </li>

                    <li class="nav-item">

                        <a class="nav-link" href="#">Link</a>

                    </li>

                </ul>

                <div class="speed-menu">

                    <a href=""><i data-feather="user"></i></a>

                    <a href=""><i data-feather="shopping-cart"></i></a>

                </div>

            </div>

        </div>

    </nav>

</header>

<main>

    <div id="myCarousel" class="carousel slide mb-6" data-bs-ride="carousel">

        <div class="carousel-indicators">

            <button type="button" data-bs-target="#myCarousel" data-bs-slide-to="0" class="active" aria-current="true" aria-label="Slide 1"></button>

            <button type="button" data-bs-target="#myCarousel" data-bs-slide-to="1" aria-label="Slide 2"></button>

            <button type="button" data-bs-target="#myCarousel" data-bs-slide-to="2" aria-label="Slide 3"></button>

        </div>

        <div class="carousel-inner">

            <div class="carousel-item active">

                <svg class="bd-placeholder-img" width="100%" height="100%" xmlns="http://www.w3.org/2000/svg" aria-hidden="true" preserveAspectRatio="xMidYMid slice" focusable="false"><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <div class="container">

                    <div class="carousel-caption text-start">

                        <h1>Example headline.</h1>

                        <p class="opacity-75">Some representative placeholder content for the first slide of the carousel.</p>

                        <p><a class="btn btn-lg btn-primary" href="#">Sign up today</a></p>

                    </div>

                </div>

            </div>

            <div class="carousel-item">

                <svg class="bd-placeholder-img" width="100%" height="100%" xmlns="http://www.w3.org/2000/svg" aria-hidden="true" preserveAspectRatio="xMidYMid slice" focusable="false"><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <div class="container">

                    <div class="carousel-caption">

                        <h1>Another example headline.</h1>

                        <p>Some representative placeholder content for the second slide of the carousel.</p>

                        <p><a class="btn btn-lg btn-primary" href="#">Learn more</a></p>

                    </div>

                </div>

            </div>

            <div class="carousel-item">

                <svg class="bd-placeholder-img" width="100%" height="100%" xmlns="http://www.w3.org/2000/svg" aria-hidden="true" preserveAspectRatio="xMidYMid slice" focusable="false"><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <div class="container">

                    <div class="carousel-caption text-end">

                        <h1>One more for good measure.</h1>

                        <p>Some representative placeholder content for the third slide of this carousel.</p>

                        <p><a class="btn btn-lg btn-primary" href="#">Browse gallery</a></p>

                    </div>

                </div>

            </div>

        </div>

        <button class="carousel-control-prev" type="button" data-bs-target="#myCarousel" data-bs-slide="prev">

            <span class="carousel-control-prev-icon" aria-hidden="true"></span>

            <span class="visually-hidden">Previous</span>

        </button>

        <button class="carousel-control-next" type="button" data-bs-target="#myCarousel" data-bs-slide="next">

            <span class="carousel-control-next-icon" aria-hidden="true"></span>

            <span class="visually-hidden">Next</span>

        </button>

    </div>

    <!-- Marketing messaging and featurettes

    ================================================== -->

    <!-- Wrap the rest of the page in another container to center all the content. -->

    <div class="container marketing">

        <!-- Three columns of text below the carousel -->

        <div class="row">

            <div class="col-lg-4">

                <svg class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <h2 class="fw-normal">Heading</h2>

                <p>Some representative placeholder content for the three columns of text below the carousel. This is the first column.</p>

                <p><a class="btn btn-secondary" href="#">View details &raquo;</a></p>

            </div><!-- /.col-lg-4 -->

            <div class="col-lg-4">

                <svg class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <h2 class="fw-normal">Heading</h2>

                <p>Another exciting bit of representative placeholder content. This time, we've moved on to the second column.</p>

                <p><a class="btn btn-secondary" href="#">View details &raquo;</a></p>

            </div><!-- /.col-lg-4 -->

            <div class="col-lg-4">

                <svg class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-color)"/></svg>

                <h2 class="fw-normal">Heading</h2>

                <p>And lastly this, the third column of representative placeholder content.</p>

                <p><a class="btn btn-secondary" href="#">View details &raquo;</a></p>

            </div><!-- /.col-lg-4 -->

        </div><!-- /.row -->

        <!-- START THE FEATURETTES -->

        <hr class="featurette-divider">

        <div class="row featurette">

            <div class="col-md-7">

                <h2 class="featurette-heading fw-normal lh-1">First featurette heading. <span class="text-body-secondary">It’ll blow your mind.</span></h2>

                <p class="lead">Some great placeholder content for the first featurette here. Imagine some exciting prose here.</p>

            </div>

            <div class="col-md-5">

                <svg class="bd-placeholder-img bd-placeholder-img-lg featurette-image img-fluid mx-auto" width="500" height="500" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder: 500x500" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-bg)"/><text x="50%" y="50%" fill="var(--bs-secondary-color)" dy=".3em">500x500</text></svg>

            </div>

        </div>

        <hr class="featurette-divider">

        <div class="row featurette">

            <div class="col-md-7 order-md-2">

                <h2 class="featurette-heading fw-normal lh-1">Oh yeah, it’s that good. <span class="text-body-secondary">See for yourself.</span></h2>

                <p class="lead">Another featurette? Of course. More placeholder content here to give you an idea of how this layout would work with some actual real-world content in place.</p>

            </div>

            <div class="col-md-5 order-md-1">

                <svg class="bd-placeholder-img bd-placeholder-img-lg featurette-image img-fluid mx-auto" width="500" height="500" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder: 500x500" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-bg)"/><text x="50%" y="50%" fill="var(--bs-secondary-color)" dy=".3em">500x500</text></svg>

            </div>

        </div>

        <hr class="featurette-divider">

        <div class="row featurette">

            <div class="col-md-7">

                <h2 class="featurette-heading fw-normal lh-1">And lastly, this one. <span class="text-body-secondary">Checkmate.</span></h2>

                <p class="lead">And yes, this is the last block of representative placeholder content. Again, not really intended to be actually read, simply here to give you a better view of what this would look like with some actual content. Your content.</p>

            </div>

            <div class="col-md-5">

                <svg class="bd-placeholder-img bd-placeholder-img-lg featurette-image img-fluid mx-auto" width="500" height="500" xmlns="http://www.w3.org/2000/svg" role="img" aria-label="Placeholder: 500x500" preserveAspectRatio="xMidYMid slice" focusable="false"><title>Placeholder</title><rect width="100%" height="100%" fill="var(--bs-secondary-bg)"/><text x="50%" y="50%" fill="var(--bs-secondary-color)" dy=".3em">500x500</text></svg>

            </div>

        </div>

        <hr class="featurette-divider">

        <!-- /END THE FEATURETTES -->

    </div><!-- /.container -->

    <!-- FOOTER -->

    <footer class="container">

        <p>&copy; 2017–2024 La Boutique Française <a href="#">CGV</a> &middot; <a href="#">CGU</a></p>

    </footer>

</main>

<script src="/assets/js/bootstrap.bundle.min.js"></script>

<script src="/assets/js/global.js"></script>

</body>

</html>

On voit que ce fichier utilise feather-icons on va l’installer avec webpack :

npm install feather-icons

Importer feather-icons dans app.js :

import feather from 'feather-icons';

feather.replace();  // Initialisation des icônes Feather

puis npm run dev

On utilise aussi deux feuilles css que l’on place dans assets/style :

custom.css

.bd-placeholder-img {

  font-size: 1.125rem;

  text-anchor: middle;

  -webkit-user-select: none;

  -moz-user-select: none;

  user-select: none;

}

@media (min-width: 768px) {

  .bd-placeholder-img-lg {

    font-size: 3.5rem;

  }

}

.b-example-divider {

  width: 100%;

  height: 3rem;

  background-color: rgba(0, 0, 0, 0.1);

  border: solid rgba(0, 0, 0, 0.15);

  border-width: 1px 0;

  box-shadow: inset 0 0.5em 1.5em rgba(0, 0, 0, 0.1),

    inset 0 0.125em 0.5em rgba(0, 0, 0, 0.15);

}

.b-example-vr {

  flex-shrink: 0;

  width: 1.5rem;

  height: 100vh;

}

.bi {

  vertical-align: -0.125em;

  fill: currentColor;

}

.nav-scroller {

  position: relative;

  z-index: 2;

  height: 2.75rem;

  overflow-y: hidden;

}

.nav-scroller .nav {

  display: flex;

  flex-wrap: nowrap;

  padding-bottom: 1rem;

  margin-top: -1px;

  overflow-x: auto;

  text-align: center;

  white-space: nowrap;

  -webkit-overflow-scrolling: touch;

}

.btn-bd-primary {

  --bd-violet-bg: #712cf9;

  --bd-violet-rgb: 112.520718, 44.062154, 249.437846;

  --bs-btn-font-weight: 600;

  --bs-btn-color: var(--bs-white);

  --bs-btn-bg: var(--bd-violet-bg);

  --bs-btn-border-color: var(--bd-violet-bg);

  --bs-btn-hover-color: var(--bs-white);

  --bs-btn-hover-bg: #6528e0;

  --bs-btn-hover-border-color: #6528e0;

  --bs-btn-focus-shadow-rgb: var(--bd-violet-rgb);

  --bs-btn-active-color: var(--bs-btn-hover-color);

  --bs-btn-active-bg: #5a23c8;

  --bs-btn-active-border-color: #5a23c8;

}

.bd-mode-toggle {

  z-index: 1500;

}

.bd-mode-toggle .dropdown-menu .active .bi {

  display: block !important;

}

.speed-menu a {

  color: #fff;

}

.speed-menu a {

  margin-right: 20px;

}

Et carousel.css

/\* GLOBAL STYLES

-------------------------------------------------- \*/

/\* Padding below the footer and lighter body text \*/

body {

  padding-top: 3rem;

  padding-bottom: 3rem;

  color: rgb(var(--bs-tertiary-color-rgb));

}

/\* CUSTOMIZE THE CAROUSEL

-------------------------------------------------- \*/

/\* Carousel base class \*/

.carousel {

  margin-bottom: 4rem;

}

/\* Since positioning the image, we need to help out the caption \*/

.carousel-caption {

  bottom: 3rem;

  z-index: 10;

}

/\* Declare heights because of positioning of img element \*/

.carousel-item {

  height: 32rem;

}

/\* MARKETING CONTENT

-------------------------------------------------- \*/

/\* Center align the text within the three columns below the carousel \*/

.marketing .col-lg-4 {

  margin-bottom: 1.5rem;

  text-align: center;

}

/\* rtl:begin:ignore \*/

.marketing .col-lg-4 p {

  margin-right: 0.75rem;

  margin-left: 0.75rem;

}

/\* rtl:end:ignore \*/

/\* Featurettes

------------------------- \*/

.featurette-divider {

  margin: 5rem 0; /\* Space out the Bootstrap <hr> more \*/

}

/\* Thin out the marketing headings \*/

/\* rtl:begin:remove \*/

.featurette-heading {

  letter-spacing: -0.05rem;

}

/\* rtl:end:remove \*/

/\* RESPONSIVE CSS

-------------------------------------------------- \*/

@media (min-width: 40em) {

  /\* Bump up size of carousel content \*/

  .carousel-caption p {

    margin-bottom: 1.25rem;

    font-size: 1.25rem;

    line-height: 1.4;

  }

  .featurette-heading {

    font-size: 50px;

  }

}

@media (min-width: 62em) {

  .featurette-heading {

    margin-top: 7rem;

  }

}

On les importe dans app.js :

import "./styles/carousel.css";

import "./styles/custom.css";

# User, signup signin et security:

composer require symfony/security-bundle

php bin/console make:user

security.yaml a changé avec un user provider

si on veut ajouter des propriétés à user on fait symfony console make :entity puis on tape User

### Ne pas oublier de vérifier la db dans le fichier .env

symfony console doctrine:database:create

puis

php bin/console make:migration

php bin/console doctrine:migrations:migrate

La table User ainsi est créée dans notre base de données

# Page Inscription

symfony console make:controller RegisterController

Le formulaire: symfony console make:form

Nom : RegisterUserType

Utilise l’entity User

Ensuite il faut convoquer ce form dans le controller Register pour le faire apparaitre dans la view.

class RegisterController extends AbstractController

{

    #[Route('/inscription', name: 'app\_register')]

    public function index(): Response

    {

        $form = $this->createForm(RegisterUserType::class);

        return $this->render('register/index.html.twig',['form' => $form->createView()

    ]);

    }

}

Dans la view twig dispose d’une commande spéciale :

{{form(registerForm}}

Pour donner du style au formulaire on ajoute dans config/package/twig.yaml ceci :

twig:

  file\_name\_pattern: "\*.twig"

  form\_themes:

    - "bootstrap\_5\_layout.html.twig"

Dans registerusertype:

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

puis

 ->add('submit', SubmitType::class, [

                'label' => 'Inscription',

            ])

Pour le bouton submit

## Pour améliorer notre formulaire :

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

                'label' => 'Adresse email', 'attr' => ['placeholder' => 'Entrez votre adresse email']

            ])

            ->add('password', PasswordType::class, [

                'label' => 'Mot de passe', 'attr' => ['placeholder' => 'Entrez votre mot de passe']

            ])

            ->add('name', TextType::class, [

                'label' => 'Nom', 'attr' => ['placeholder' => 'Entrez votre nom']

            ])

            ->add('submit', SubmitType::class, [

                'label' => 'Inscription', 'attr' => ['class' => 'btn btn-success']

            ])

## Pour que le formulaire fonctionne :

Dans le RegisterController nous devons ajouter Request :

 public function index(Request $request): Response

puis on fait appel à doctrine :

public function index(Request $request, EntityManagerInterface $entityManagerInterface): Response

On attache User à notre formulaire si le formulaire est envoyé, doctrine fait le reste :

 {

$user = new User();

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

        $form->handleRequest($request);

        if($form->isSubmitted() && $form->isValid()){

            $user = $form->getData();

            //figer les données

            $entityManagerInterface->persist($user);

            //envoyer les données

            $entityManagerInterface->flush();

        }

### Bien vérifier que tout a été importé :

use App\Entity\User;

use App\Form\RegisterUserType; // Import the missing class

use Doctrine\ORM\EntityManagerInterface;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

use Symfony\Component\HttpFoundation\Request;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\Routing\Attribute\Route;

### Pour confirmer password et version finale builder:

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

    {

        $builder

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

                'label' => 'Adresse email', 'attr' => ['placeholder' => 'Entrez votre adresse email']

            ])

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

                'type' => PasswordType::class,

                'first\_options'  => [

                    'label' => 'Mot de passe',

                    'attr' => ['placeholder' => 'Entrez votre mot de passe'],

                    'hash\_property\_path' => 'password'

                ],

                'second\_options' => [

                    'label' => 'Confirmer le mot de passe',

                    'attr' => ['placeholder' => 'Confirmez votre mot de passe'],

                ],

                'invalid\_message' => 'Les mots de passe ne correspondent pas',

                'mapped' => false,

            ])

            ->add('name', TextType::class, [

                'label' => 'Nom', 'attr' => ['placeholder' => 'Entrez votre nom']

            ])

            ->add('submit', SubmitType::class, [

                'label' => 'Inscription', 'attr' => ['class' => 'btn btn-success']

            ])

        ;

    }

Mapped => false permet de dire à doctrine qu’il ne doit pas le chercher dans entity

### Pour un email unique on met une contrainte en bas dans configureOptions :

public function configureOptions(OptionsResolver $resolver): void

    {

        $resolver->setDefaults([

            'constraints'=> [

                new UniqueEntity([

                    'entityClass' => User::class,

                    'fields' => ['email'],

                ])

                ],

            'data\_class' => User::class,

        ]);

    }

### Composer require symfony/validator

### Pour une contrainte sur password :

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

                'type' => PasswordType::class,

                'constraints' => [

                    new NotBlank([

                        'message' => 'Veuillez entrer un mot de passe',

                    ]),

                    new Length([

                        'min' => 6,

                        'minMessage' => 'Votre mot de passe doit contenir au moins {{ limit }} caractères',

                        // max length allowed by Symfony for security reasons

                        'max' => 4096,

                    ]),

                ],

                'first\_options'  => [

                    'label' => 'Mot de passe',

                    'attr' => ['placeholder' => 'Entrez votre mot de passe'],

                    'hash\_property\_path' => 'password'

                ],

                'second\_options' => [

                    'label' => 'Confirmer le mot de passe',

                    'attr' => ['placeholder' => 'Confirmez votre mot de passe'],

                ],

                'invalid\_message' => 'Les mots de passe ne correspondent pas',

                'mapped' => false,

            ])

# Pour le Login

symfony console make:controller LoginController

Dans security.yaml ajouter  form\_login:

 firewalls:

        dev:

            pattern: ^/(\_(profiler|wdt)|css|images|js)/

            security: false

        main:

            lazy: true

            provider: app\_user\_provider

            form\_login:

                login\_path: app\_login

                check\_path: app\_login

puis dans le controller :

class LoginController extends AbstractController

{

    #[Route('/connexion', name: 'app\_login')]

    public function index(AuthenticationUtils $authenticationUtils): Response

    {

        //gérer les erreurs

        $error=$authenticationUtils->getLastAuthenticationError();

        //récupérer le dernier nom d'utilisateur

        $lastUsername=$authenticationUtils->getLastUsername();

        return $this->render('login/index.html.twig', [

            'error' => $error, 'last\_username' => $lastUsername

        ]);

    }

Dans twig view :

<div class="container my-5">

{% if error %}

        <div>{{ error.messageKey|trans(error.messageData, 'security') }}</div>

    {% endif %}

    <div class="col-md-6 offset-md-3">

            <h1 class="text-center">Connexion</h1>

        <form action="{{ path('app\_login') }}" method="post">

            <label for="username">Email:</label>

            <input class='form-control' type="text" id="username" name="\_username" value="{{ last\_username }}">

            <label for="password mt-3">Password:</label>

            <input type="password" class='form-control' id="password" name="\_password">

            <input type="hidden" name="\_target\_path" value="{{ path('app\_account') }}">

            <button type="submit" class="btn btn-success my-3" >Me connecter</button>

        </form>

    </div>

</div>

Pour Logout

Dans security.yaml :

logout:

        path: app\_logout

      form\_login:

        login\_path: app\_login

        check\_path: app\_login

dans login controller :

#[Route('/deconnexion', name: 'app\_logout', methods: ['GET'])]

    public function logout(): never

    {

        throw new \Exception('This method can be blank - it will be intercepted by the logout key on your firewall');

    }

# Profil et Modifier password

symfony console make:controller AccountController

<?php

namespace App\Controller;

use App\Form\PasswordUserType;

use Doctrine\ORM\EntityManagerInterface;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

use Symfony\Component\HttpFoundation\Request;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\PasswordHasher\Hasher\UserPasswordHasherInterface;

use Symfony\Component\Routing\Attribute\Route;

class AccountController extends AbstractController

{

    #[Route('/compte', name: 'app\_account')]

    public function index(): Response

    {

        return $this->render('account/index.html.twig');

    }

     #[Route('/compte/modifier-mot-de-passe', name: 'app\_account\_modify\_pwd')]

    public function password(Request $request, UserPasswordHasherInterface $passwordHasher, EntityManagerInterface $entityManager): Response

    {

        $user= $this->getUser();

        $form = $this->createForm(PasswordUserType::class, $user, ['passwordHasher' => $passwordHasher]);

        $form->handleRequest($request);

        if($form->isSubmitted() && $form->isValid())

        {

            $entityManager->flush();

            $this->addFlash('success', 'Votre mot de passe a été modifié avec succès');

            return $this->redirectToRoute('app\_account');

        }

        return $this->render('account/password.html.twig', ['modifyPwdForm' => $form->createView()]);

    }

}

La vue twig avec une nouvelle function modifier password:

{% extends 'base.html.twig' %}

{% block body %}

<div class="container my-5">

    <h1 class="text-center">Mon compte</h1>

    <div class="col-md-6 offset-md-3">

        <h2 class="text-center">Mes informations</h2>

        <p>Nom: {{ app.user.name }}</p>

        <p>Email: {{ app.user.email }}</p>

        <a href="{{ path('app\_logout') }}" class="btn btn-danger">Me déconnecter</a>

         <a href="{{ path('app\_account\_modify\_pwd') }}" class="btn btn-success">Modifier Mot de Passe</a>

    </div>

{% endblock %}

On fait une nouvelle view password.html.twig dans le dossier account

{% extends 'base.html.twig' %}

{% block body %}

<div class="container my-5">

    <h1 class="text-center">Espace Membre privé</h1>

    <div class="col-md-6 offset-md-3">

        <h2 class="text-center">Modification Mot de Passe</h2>

        {{form(modifyPwdForm)}}

    </div>

{% endblock %}

Formulaire

symfony console make:form PasswordUserType (lié à l’entity User)

<?php

namespace App\Form;

use App\Entity\User;

use Symfony\Component\Form\AbstractType;

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

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

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

use Symfony\Component\Form\FormBuilderInterface;

use Symfony\Component\Form\FormError;

use Symfony\Component\Form\FormEvent;

use Symfony\Component\Form\FormEvents;

use Symfony\Component\OptionsResolver\OptionsResolver;

use Symfony\Component\Validator\Constraints\Length;

use Symfony\Component\Validator\Constraints\NotBlank;

class PasswordUserType extends AbstractType

{

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

    {

        $builder

        ->add('actualPassword', PasswordType::class, [

            'label' => 'Mot de passe actuel',

            'attr' => ['placeholder' => 'Entrez votre mot de passe actuel'],

            'mapped' => false

        ])

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

                'type' => PasswordType::class,

                'constraints' => [

                    new NotBlank([

                        'message' => 'Veuillez entrer un nouveau mot de passe',

                    ]),

                    new Length([

                        'min' => 6,

                        'minMessage' => 'Votre mot de passe doit contenir au moins {{ limit }} caractères',

                        // max length allowed by Symfony for security reasons

                        'max' => 4096,

                    ]),

                ],

                'first\_options'  => [

                    'label' => 'Mot de passe',

                    'attr' => ['placeholder' => 'Entrez votre nouveau mot de passe'],

                    'hash\_property\_path' => 'password'

                ],

                'second\_options' => [

                    'label' => 'Confirmer le mot de passe',

                    'attr' => ['placeholder' => 'Confirmez votre nouveau mot de passe'],

                ],

                'invalid\_message' => 'Les mots de passe ne correspondent pas',

                'mapped' => false,

            ])

             ->add('submit', SubmitType::class, [

                'label' => 'Modifier mot de passe', 'attr' => ['class' => 'btn btn-success']

            ])

            ->addEventListener(FormEvents::SUBMIT, function (FormEvent $event) {

                $form = $event->getForm();

                $user = $form->getConfig()->getOptions()['data'];

                $passwordHasher = $form->getConfig()->getOptions() ['passwordHasher'];

                $isValid = $passwordHasher->isPasswordValid($user, $form->get('actualPassword')->getData());

                if (!$isValid) {

                    $form->get('actualPassword')->addError(new FormError('Mot de passe incorrect'));

                }

            })

        ;

    }

    public function configureOptions(OptionsResolver $resolver): void

    {

        $resolver->setDefaults([

            'data\_class' => User::class,

            'passwordHasher' => null

        ]);

    }

}

Voila!

Pour les flash message: dans base.html.twig

Ajouter  en dessous de main :

<main>

    {% for label, messages in app.flashes %}

        {% for message in messages %}

            <div class="flash-{{ label }}">

                {{ message }}

            </div>

        {% endfor %}

        {% endfor %}

Il ne reste plus qu’à mettre le header avec connexion ou déconnexion :

<ul class="navbar-nav me-auto mb-2 mb-md-0">

              <li class="nav-item">

                <a class="nav-link" href="{{ path('app\_home') }}">Accueil</a>

              </li>

              {% if app.user %}

                <li class="nav-item">

                  <a class="nav-link">Vous êtes connecté</a>

                </li>

                <li class="nav-item">

                  <a class="nav-link" href="{{ path('app\_logout') }}">Déconnexion</a>

                </li>

              {% else %}

                <li class="nav-item">

                  <a class="nav-link" href="{{ path('app\_login') }}">Connexion</a>

                </li>

              {% endif %}

            </ul>

# Les roles:

## Créer un administrateur

Pour cette première étape nous pouvons utiliser une commande au lieu de faire un formulaire etc..

Cela offre l’avantage d’automatiser cette tâche et d’éviter les erreurs.

Voici un exemple :

Dans un nouveau dossier src/Command faire un fichier CreateAdminUserCommand.php

<?php

// src/Command/CreateAdminUserCommand.php

namespace App\Command;

use App\Entity\User;

use Doctrine\ORM\EntityManagerInterface;

use Symfony\Component\Console\Command\Command;

use Symfony\Component\Console\Input\InputInterface;

use Symfony\Component\Console\Output\OutputInterface;

use Symfony\Component\Console\Style\SymfonyStyle;

use Symfony\Component\PasswordHasher\Hasher\UserPasswordHasherInterface;

class CreateAdminUserCommand extends Command

{

    protected static $defaultName = 'app:create-admin-user';

    private $entityManager;

    private $passwordHasher;

    public function \_\_construct(EntityManagerInterface $entityManager, UserPasswordHasherInterface $passwordHasher)

    {

        parent::\_\_construct();

        $this->entityManager = $entityManager;

        $this->passwordHasher = $passwordHasher;

    }

    protected function configure()

    {

        $this

            ->setDescription('Creates a new admin user')

        ;

    }

    protected function execute(InputInterface $input, OutputInterface $output): int

    {

        $io = new SymfonyStyle($input, $output);

        $user = new User();

        $user->setEmail('admin@admin.com');

        $user->setPassword($this->passwordHasher->hashPassword($user, '123456'));

        $user->setRoles(['ROLE\_ADMIN']);

        $this->entityManager->persist($user);

        $this->entityManager->flush();

        $io->success('Admin user created successfully.');

        return Command::SUCCESS;

    }

}

Pour lancer cette commande:

php bin/console app:create-admin-user

Easy Admin :

symfony composer require easycorp/easyadmin-bundle

dans src / controller se crée un Admin/DashboardController

php bin/console make :admin :dashboard

on accepte dashboardcontroller

on accepte le repertoire .

La premiére entité à lier c’est User

Commande :

symfony console make:admin:crud

on rentre 0 pour User

on accepte le répertoire

on accepte le namespace

Dans le controller il n’y a pas grand-chose.

Dans dashboardcontroller :

On décommente option 1 :

On retape AdminUrlGenerator pour qu’il fasse l’import

OneofYour on le remplace par UserCrudController

On doit ajouter des liens

Ligne 27 configureMenuItems pour ajouter des links

Décommenter le second label utilisateurs

Changer entity pour User

Dans la doc configure crud copier function, allez dans UserCrudController

Coller en dessous de getEntity

Il manque les dépendances EasyCorpbundle

->setEntityLabelInSingular (‘Utilisateur’)

->setEntityLabelInPlural(‘Utilisateurs’)

Voir configureField

Pas modifier l’email car clef de connexion

Dans return

TextField