

# Chapitre 1 - Création d'une application avec Symfony 6.4

Introduction.....	
Mise en place.....	
Vérification technique :.....	
Création du projet :.....	
Installation et configuration d'Apache:.....	
Création de la page d'accueil du site :.....	
Installation des packages :.....	
Création du contrôleur :.....	
Mise en place de Bootstrap 5.3.....	
Mise en place de la barre de navigation du site :.....	
Personnalisation de la barre de menu (navbar) :.....	
Mise en forme de la page d'accueil :.....	



## Introduction

Dans cette série de chapitres, nous allons créer une application avec le Framework Symfony. Nous partirons d'une commande qui installera le strict minimum puis nous installerons au fur et à mesure les modules complémentaires suivant nos besoins.

## Mise en place

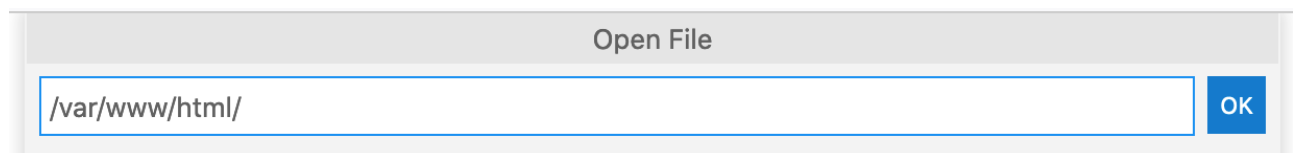
Après avoir allumé votre machine, déroulez les options de votre serveur et cliquez sur « Vscode ».



The screenshot shows a server configuration panel for 'Https-4400'. It includes fields for 'Login' (login4400), 'Mot de passe' (password) with 'Afficher' and 'Copier' buttons, 'Durée' (4h), 'Mot de passe root' (root password) with 'Afficher' and 'Copier' buttons, and 'Port' (4400). Below these is an 'Accès ssh' section with a terminal command: `ssh -p 4400 login4400@s3-4400.nuage-peda.fr`. At the bottom, there is a 'Vscode' button and a row of icons: a red square, a folder, a list, and a trash can.

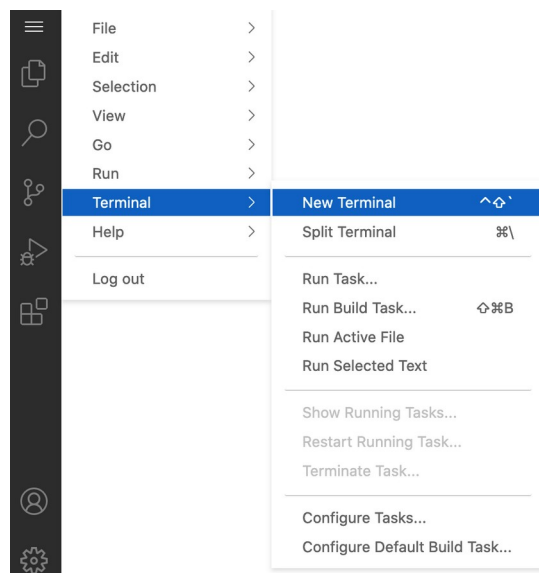
Copiez votre mot de passe utilisateur et entrez dans l'environnement de développement .

Placez-vous dans le répertoire de publication du serveur Apache (/var/www/html) en utilisant le 2ème menu de gauche (Explorer).



The screenshot shows an 'Open File' dialog box. The text input field contains the path `/var/www/html/`. To the right of the input field is a blue button labeled 'OK'.

Puis lancez le terminal :



Nous voyons que le compte par défaut est le root. Ce qui n'est pas une bonne chose pour développer. Récupérez votre login sur votre profil (login 4400 pour moi) et tapez la commande suivante :

```
su login4400
```

Vérifiez que vous êtes bien situé dans le répertoire de publication avec la commande :

```
pwd
```

Le résultat suivant doit apparaître : /var/www/html

## Vérification technique :

Mettre à jour PHP :

La version PHP nécessaire pour créer des projets en 6.4 est PHP8.1 ou plus. Voir la page des releases : <https://symfony.com/releases>

```
php --version
```

Si ce n'est pas le cas, mettez à jour avec le compte root votre machine.

```
apt update
apt install php8.1
apt install php8.1-xml php8.1-zip php8.1-curl php8.1-mysql php8.1-intl
a2dismod php8.0
a2enmod php8.1
service apache2 restart
```

**Mettre à jour composer :**

```
en root
chmod -R 777 /usr/local/bin
en login
composer self-update
```

## Création du projet :

Nous allons taper la commande qui va nous créer un projet Symfony. Elle contiendra le moins de packages possible. En effet, un site statique n'a pas besoin de grand-chose pour fonctionner.

Avec cette commande, la dernière version disponible de Symfony s'installera. (à l'heure où j'écris ce cours, nous sommes dans la version 6.3)

Avec le compte login :

```
composer create-project symfony/skeleton share
```

Vous pouvez également installer une version spécifique de Symfony en précisant sa version :

```
composer create-project symfony/skeleton share "6.4.*"
```

Vous pouvez vérifier la création du site en vous rendant sur votre profil et en cliquant sur « Aller sur le site Web ». À la fin de l'URL, ajoutez /share et appuyez sur la touche «entrer ».












*Exemple :*

```
https://s3-4400.nuage-peda.fr/share/
```

Vous voyez tous les répertoires et les fichiers de Symfony.

Voici un tour rapide des répertoires créé par le programme d'installation :

### Index of /share

Name	Last modified	Size	Description
 <a href="#">Parent Directory</a>		-	
 <a href="#">bin/</a>	2021-09-25 19:41	-	
 <a href="#">composer.json</a>	2021-10-08 16:54	1.6K	
 <a href="#">composer.lock</a>	2021-10-08 16:54	129K	
 <a href="#">config/</a>	2021-09-25 19:41	-	
 <a href="#">public/</a>	2021-10-08 16:30	-	
 <a href="#">src/</a>	2021-09-25 19:41	-	
 <a href="#">symfony.lock</a>	2021-10-08 16:54	5.6K	
 <a href="#">templates/</a>	2021-10-02 11:26	-	
 <a href="#">var/</a>	2021-09-25 19:41	-	
 <a href="#">vendor/</a>	2021-10-08 16:54	-	

- Le répertoire « bin » contient le programme « console » qui va nous aider à réaliser automatiquement des tâches qui sont habituellement pénibles comme la création de contrôleurs ou d'entités.
- Le fichier « composer.json » contient la liste des packages installés dans notre projet.
- Le répertoire « config » qui, comme son nom l'indique, permet de configurer des éléments du site comme les routes ou la sécurité.
- Le répertoire « public » contient « index.php », le seul fichier PHP accessible depuis un navigateur et qui est le point d'entrée de notre site.
- Le répertoire « src » est celui qui contiendra toutes les sources du site, celles que vous complèterez et écrirez.
- Le répertoire « var » contient le cache.
- Le répertoire « vendor » contient les packages de Symfony ou d'autres.

Le site créé est accessible dans le répertoire public par l'intermédiaire du fichier index.php.

Exemple :

```
https://s3-4400.nuage-peda.fr/share/public/  
ou  
https://s3-4400.nuage-peda.fr/share/public/index.php
```

## Installation et configuration d'Apache:

Vous remarquez qu'il est nécessaire de cibler le répertoire « public » pour visualiser le site. Nous allons configurer Apache pour se rendre directement dans le bon répertoire. Nous allons créer un alias.

Vous pouvez créer un autre terminal pour être en root ou taper « exit ».

En root, dans /etc/apache2/conf-available/share.conf

```
nano /etc/apache2/conf-available/share.conf
```

```
Alias /share /var/www/html/share/public  
<Directory /var/www/html/share/public>  
AllowOverride All  
Order Allow,Deny  
Allow from All  
</Directory>
```

Activer l'alias.

```
a2enconf share.conf
```

Vérifier la configuration effectuée :

```
apache2ctl configtest
```

**La valeur retournée doit être Syntax OK**

Puis redémarrer Apache uniquement s'il a marqué Syntax OK.

```
systemctl reload apache2
```

## Création de la page d'accueil du site :

Symfony utilise l'architecture MVC (Modèle Vue Contrôleur).

Nous entrerons dans le détail dans d'autres chapitres, cependant, voici une explication rapide :

- Le « modèle » correspond aux entités de l'application qui nous permettrons, entre autres de créer, manipuler, conserver des données.
- La « vue » permet d'afficher les pages qui seront retournées au navigateur de l'utilisateur. Elle repose sur le langage TWIG, qui permet de créer du HTML avec du contenu qui sera donné par le contrôleur.
- Le « contrôleur » est le chef d'orchestre d'une page. Il est capable de parler au modèle afin de lui fournir ou de récupérer des données. Ensuite, il est capable d'appeler une vue tout en lui donnant les données nécessaires à la création d'une page.

Pour accéder à un contrôleur, une route est nécessaire. C'est l'URL qu'il faut saisir dans le navigateur pour se rendre sur une page.

Dans Symfony, nous pouvons regrouper des contrôleurs dans un seul fichier pour éviter d'en avoir un trop grand nombre. Généralement, nous créons un nouveau fichier par thème pour nous déplacer facilement dans le code.

Généralement, je regroupe dans un seul fichier, les fonctionnalités basiques et courantes d'un site ;

- page d'accueil,
- formulaire de contact,
- mentions légales,
- à propos

## Installation des packages :

Dans le terminal, placez-vous dans le répertoire du projet : (avec le compte login)

```
cd share
```

Pour créer un contrôleur, nous devons installer des outils spécifiques en tapant les commandes suivantes :

```
composer require symfony/maker-bundle --dev
```

**Remarque :** l'option --dev précise que cet outil sera disponible uniquement en mode développeur.

```
composer require doctrine/annotations
```

Enfin, nous devons installer TWIG avec :

```
composer require symfony/twig-bundle
```

## Création du contrôleur :

Nous allons créer le fichier qui contiendra nos contrôleurs de base.

Symfony nous propose un outil puissant qui nous permet de créer automatiquement certains éléments de notre site. Il s'agit de la console située dans le répertoire « bin ».

Tapez la commande suivante :

```
php bin/console make:controller
```

Puis saisissez :

```
Base
```

Dans le répertoire « src/Controller », vous trouvez maintenant le nouveau fichier « BaseController.php ».

```
<?php

namespace App\Controller;

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

class BaseController extends AbstractController
{
    #[Route('/base', name: 'app_base')] // /base est l'URL de la page, name est le nom de la route
    public function index(): Response
    {
        return $this->render('base/index.html.twig', [ // render est la fonction qui va chercher le
fichier TWIG pour l'afficher
            'controller_name' => 'BaseController', // Le contrôleur donne à la vue une variable dont le
contenu est BaseController, cela nous ne servira pas, nous l'enlèverons un peu plus loin
        ]);
    }
}
```

Dans le répertoire « templates », le répertoire « base » a été créé avec le fichier TWIG « index.html.twig ». Le contenu par défaut ne nous intéresse pas du tout, nous le remplacerons avec notre propre contenu.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>{% block title %}Welcome!{% endblock %}</title>
    <link rel="icon" href="data:image/svg+xml,<svg xmlns=%22http://www.w3.org/2000/svg
%22 viewBox=%220 0 128 128%22><text y=%221.2em%22
font-size=%2296%22>  </text></svg>">
    {% block stylesheets %}
    {% endblock %}
```

```
{% block javascripts %}
{% endblock %}
</head>
<body>
  {% block body %}{% endblock %}
</body>
</html>
```

Une nouvelle page a été créée sur le site. Vous pouvez y accéder en tapant l'URL suivante :

```
index.php/base
```

**Exemple :**

```
https://s3-4400.nuage-peda.fr/share/index.php/base
```

Vous remarquez que l'URL n'a pas une forme habituelle.

Nous allons installer la gestion des URL avec la commande suivante dans le répertoire du projet :

```
composer require symfony/apache-pack
```

Il nous pose une question à laquelle il faut répondre « y ».

Maintenant, l'URL suivante fonctionne :

```
https://s3-4400.nuage-peda.fr/share/base
```

Nous allons maintenant mettre en forme notre page d'accueil.

Dans le répertoire « templates », nous avons le fichier « base.html.twig ». il s'agit du fichier qui contiendra la structure du site qui sera utilisé dans toutes nos pages. C'est ici que nous déclarerons les blocs qui seront modifiés par la suite.

Contenu actuel de « templates/base.html.twig » :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>{% block title %}Welcome!{% endblock %}</title>
    <link rel="icon" href="data:image/svg+xml,<svg xmlns=%22http://www.w3.org/2000/svg
%22 viewBox=%220 0 128 128%22><text y=%221.2em%22
font-size=%2296%22>  </text></svg>">
    {% block stylesheets %}
    {% endblock %}
```



```

    {% block javascripts %}
    {% endblock %}
</head>
<body>
    {% block body %}{% endblock %}
</body>
</html>

```

### Commentaires :

Nous plaçons des blocs sur notre template afin de définir des emplacements qui seront alimentés par les différentes pages de notre site. Le nom des blocs est défini par le développeur.

<pre> {% block title %} Welcome! {% endblock %} </pre>	Le bloc « title » permet de déterminer l'emplacement du titre du site qui sera modifié suivant la page sur laquelle vous irez.
<pre> {% block stylesheets %} {% endblock %} </pre>	Le bloc « stylesheets » est l'emplacement dans lequel nous appellerons des fichiers « css » nécessaires au design des pages.
<pre> {% block javascripts %} {% endblock %} </pre>	Le bloc « javascripts » est l'emplacement où seront appelés nos fichiers JS.
<pre> {% block body %} {% endblock %} </pre>	Le bloc « body » détermine l'emplacement où nous mettrons le contenu d'une page.

## Mise en place de Bootstrap 5.3

Nous allons récupérer la structure de base de Bootstrap pour remplacer le contenu de notre fichier « base.html.twig ».

### Code du « get started » de bootstrap 5 :

**source** : <https://getbootstrap.com/docs/5.3/getting-started/introduction/>

```

<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap demo</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
T3c6CoLi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous">
  </head>
  <body>
    <h1>Hello, world!</h1>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Q
yq46cDfL" crossorigin="anonymous"></script>
  </body>

```

```
</html>
```

Voici le contenu de notre fichier « base.html.twig » après modifications :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>{% block title %} Share - {% endblock %}</title>
    {% block stylesheets %}
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous">
    {% endblock %}
  </head>
  <body>
    {% block body %} Share {% endblock %}
    {% block javascripts %}
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Q
yq46cDfL" crossorigin="anonymous"></script>
    {% endblock %}
  </body>
</html>
```

### Commentaires :

Nous entourons les lignes qui sont susceptibles de changer sur chaque page de notre site avec la déclaration de blocs.

Nous changerons ou compléterons les éléments suivants sur les différentes pages de notre site:

- les feuilles de style,
- le titre,
- le contenu,
- les scripts JavaScript

Nous allons maintenant mettre en forme notre page d'accueil. Commençons par notre contrôleur :

« src/Controller/BaseController.php » : **Supprimer la ligne en rouge**

```
<?php

namespace App\Controller;

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

```

class BaseController extends AbstractController
{
    #[Route('/base', name: 'app_base')]
    public function index(): Response
    {
        return $this->render('base/index.html.twig', [
            'controller_name' => 'BaseController', // suppression de la variable passée à la vue
        ]);
    }
}

```

### Commentaire :

Nous supprimons la variable passée à la vue, car le nom du contrôleur ne nous intéresse pas.

Modifions maintenant la vue « templates/base/index.html.twig ».

```

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

{% block title %}{{ parent() }} Accueil{% endblock %}

{% block body %}
    <h1 class="text-center text-primary"> Share : partagez vos fichiers </h1>
{% endblock %}

```

### Commentaires :

`{% extends 'base.html.twig' %}` // Appel de la configuration de base

`{% block title %}{{ parent() }} Accueil{% endblock %}` // Nous complétons le titre de base avec une indication supplémentaire sur la page courante ici nous ajoutons « accueil »  
`parent()` signifie que nous appelons le contenu du block title définit dans `base.html.twig`

`{{ }}` sont utilisés pour afficher des variables ou appeler des fonctions.

`{% block body %}` // Nous définissons complètement le contenu de notre page.  
`<h1 class="text-center text-primary"> Share : partagez vos fichiers </h1>` // `text-center` est la propriété de Bootstrap permettant de centrer le titre, `text-primary` donne la 1ère couleur définie par Bootstrap

Si vous souhaitez une personnalisation de Bootstrap, je vous invite à vous rendre sur le site « <https://bootswatch.com/> ». Il propose des thèmes assez jolis.

Vous pouvez cliquer sur la flèche « download » d'un thème, par exemple « Lumen » et sélectionnez « bootstrap.min.css » avec un clic droit. Puis faites « copier le lien ».

Allez dans votre template « templates/base.html.twig » puis remplacer le lien css bootstrap par celui que vous venez de récupérer.

<https://bootswatch.com/5/lux/bootstrap.min.css>

Remplacer :

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
T3c6CoLi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous">
```

Par :

```
<link href="https://bootswatch.com/5/lumen/bootstrap.min.css" rel="stylesheet">
```

## Mise en place de la barre de navigation du site :

Sur le site de bootswatch, récupérez un des menus disponibles dans votre thème et mettez-le dans votre template « base.html.twig » au-dessus de votre bloc « body » afin qu'il s'affiche sur toutes les pages.



```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>{% block title %}Share - {% endblock %}</title>
  {% block stylesheets %}
  <link href="https://bootswatch.com/5/lumen/bootstrap.min.css" rel="stylesheet">
```

```

    {% endblock %}
</head>
<body>
<nav class="navbar navbar-expand-lg bg-light" data-bs-theme="light">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarColor03" aria-controls="navbarColor03" aria-expanded="false" aria-
label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarColor03">
      <ul class="navbar-nav me-auto">
        <li class="nav-item">
          <a class="nav-link active" href="#">Home
            <span class="visually-hidden">(current)</span>
          </a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Features</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Pricing</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">About</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" data-bs-toggle="dropdown" href="#" role="button"
aria-haspopup="true" aria-expanded="false">Dropdown</a>
          <div class="dropdown-menu">
            <a class="dropdown-item" href="#">Action</a>
            <a class="dropdown-item" href="#">Another action</a>
            <a class="dropdown-item" href="#">Something else here</a>
            <div class="dropdown-divider"></div>
            <a class="dropdown-item" href="#">Separated link</a>
          </div>
        </li>
      </ul>
      <form class="d-flex">
        <input class="form-control me-sm-2" type="search" placeholder="Search">
        <button class="btn btn-secondary my-2 my-sm-0" type="submit">Search</button>
      </form>
    </div>
  </div>
</nav>
    {% block body %}Share{% endblock %}
    {% block javascripts %}
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Q

```

```
yq46cDfL" crossorigin="anonymous"></script>
{% endblock %}
</body>
</html>
```

Navbar Home Features Pricing About Dropdown ▼

Search

SEARCH

## Share : partagez vos fichiers

### Personnalisation de la barre de menu (navbar) :

Pour l'instant, nous disposons d'une seule page, par conséquent, nous allons modifier le nom du site et modifier le lien pour nous rendre sur la page d'accueil.

À ce propos, pour l'instant, notre page est accessible avec l'URL /base. Par défaut, notre site nous affiche toujours la page de Symfony.

Nous allons donc modifier cela.

Dans le fichier « src/controller/BaseController.php », modifiez l'url de la route en un simple « / ». Je renomme le nom de la route en « app\_accueil » car il s'agira d'afficher la page d'accueil à l'utilisateur.

```
<?php

namespace App\Controller;

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

class BaseController extends AbstractController
{
    #[Route('/', name: 'app_accueil')]
    public function index(): Response
    {
        return $this->render('base/index.html.twig', [

        ]);
    }
}
```

L'URL par défaut pointe maintenant sur votre page d'accueil.

Exemple :

```
https://s3-4400.nuage-peda.fr/share/
```

### Appel de la page d'accueil dans le menu :

Dans «base.html.twig », modifiez :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>{% block title %}Share - {% endblock %}</title>
    {% block stylesheets %}
    <link href="https://bootswatch.com/5/lumen/bootstrap.min.css" rel="stylesheet">
    {% endblock %}
  </head>
  <body>
<nav class="navbar navbar-expand-lg bg-light" data-bs-theme="light">
  <div class="container-fluid">
    <a class="navbar-brand" href="{{path('app_accueil')}}">Navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarColor03" aria-controls="navbarColor03" aria-expanded="false" aria-
label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarColor03">
      <ul class="navbar-nav me-auto">
        <li class="nav-item">
          <a class="nav-link active" href="{{path('app_accueil')}}">Home
          <span class="visually-hidden">(current)</span>
        </a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Features</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Pricing</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">About</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" data-bs-toggle="dropdown" href="#" role="button"
aria-haspopup="true" aria-expanded="false">Dropdown</a>
          <div class="dropdown-menu">
            <a class="dropdown-item" href="#">Action</a>
            <a class="dropdown-item" href="#">Another action</a>
            <a class="dropdown-item" href="#">Something else here</a>
            <div class="dropdown-divider"></div>
            <a class="dropdown-item" href="#">Separated link</a>
          </div>
        </li>
      </ul>
    </div>
  </div>
</nav>
<form class="d-flex">
```

```

        <input class="form-control me-sm-2" type="search" placeholder="Search">
        <button class="btn btn-secondary my-2 my-sm-0" type="submit">Search</button>
    </form>
</div>
</div>
</nav>
    {% block body %}Share{% endblock %}
    {% block javascripts %}
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Q
yq46cDfL" crossorigin="anonymous"></script>
    {% endblock %}
</body>
</html>

```

### Commentaires :

La fonction « path » de TWIG permet de générer le lien vers la route passée en paramètre (ici app\_accueil). N’oubliez pas d’entourer l’appel de la fonction avec deux accolades de chaque côté.

En cliquant sur « Share » ou « Accueil », l’utilisateur pourra se rendre sur la page d’accueil.

### Mise en forme de la page d’accueil :

Sur l’ensemble du site, nous utiliserons les icons de Bootstrap. Nous allons donc appeler le fichier qui les gère dans le twig de base. Voici le lien de la documentation <https://icons.getbootstrap.com/#install>

Dans « templates/base.html.twig » :

```

<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>{% block title %}Share - {% endblock %}</title>
    {% block stylesheets %}
    <link href="https://bootswatch.com/5/lumen/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.1/font/bootstrap-
icons.css">
    {% endblock %}
  </head>
  <body>
<nav class="navbar navbar-expand-lg bg-light" data-bs-theme="light">
  <div class="container-fluid">
    <a class="navbar-brand" href="{{ path('app_accueil') }}">Share</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarColor03" aria-controls="navbarColor03" aria-expanded="false" aria-
label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>

```



```

</button>
<div class="collapse navbar-collapse" id="navbarColor03">
  <ul class="navbar-nav me-auto">
    <li class="nav-item">
      <a class="nav-link active" href="{{path('app_accueil')}}">Home
      <span class="visually-hidden">(current)</span>
    </a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Features</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Pricing</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">About</a>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" data-bs-toggle="dropdown" href="#" role="button"
aria-haspopup="true" aria-expanded="false">Dropdown</a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#">Action</a>
        <a class="dropdown-item" href="#">Another action</a>
        <a class="dropdown-item" href="#">Something else here</a>
        <div class="dropdown-divider"></div>
        <a class="dropdown-item" href="#">Separated link</a>
      </div>
    </li>
  </ul>
  <form class="d-flex">
    <input class="form-control me-sm-2" type="search" placeholder="Search">
    <button class="btn btn-secondary my-2 my-sm-0" type="submit">Search</button>
  </form>
</div>
</div>
</nav>
{% block body %}Share{% endblock %}
{% block javascripts %}
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Q
yq46cDfL" crossorigin="anonymous"></script>
{% endblock %}
</body>
</html>

```

puis nous allons mettre en forme la page d'accueil.

« template/base/index.html.twig

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

{% block title %}{{parent()}}Accueil{% endblock %}

{% block body %}
<h1 class="text-center text-primary mt-4 pt-4 display-1 fw-bold"> SHARE</h1>
  <h2 class="text-center text-secondary mb-4 pb-4 display-2"> partagez vos fichiers </h2>
  <h2 class="text-center"> <i class="bi bi-cloud-arrow-up-fill text-primary" ></i><i class="bi bi-
folder text-secondary"></i><i class="bi bi-cloud-arrow-down-fill text-warning"></i></h2>
{% endblock %}
```

### Commentaires :

Quelques classes BOOTSTRAP 5 utilisées :

**text-primary** : couleur du texte, première couleur définie dans le thème.

**text-secondary** : couleur du texte, deuxième couleur définie dans le thème.

**display1** et **display2** : mise en forme du texte des titres de la balise <h1>.

**mt-4** : margin top de 4.

**mb-4** : margin bottom de 4.

**fw-bold** : mettre en gras le texte.

**m-4** : margin de 4 de tous les côtés.

**p-4** : padding de 4 de tous les côtés.

Voici l'écran final de ce chapitre :

