

# Présentation du projet “La Belle Oreille”

Formation “Développeur Web et Web  
Mobile” AFPA Brest



# Sommaire

- Présentation
- Contexte du projet
- Cahier des charges
- Maquette
- Modèle Conceptuel de données
- Environnement de développement
- Choix des Frameworks
- Installation de Symfony
- Modularisation des fichiers de style avec Webpack
- Création du template
- Création, récupération et affichage des données : la page d'accueil
- Solution de gestion de contenu
- Tests unitaires
- Intégration continue avec les actions GitHub
- Recherche sur site anglophone
- Conclusion

# Présentation





# Présentation de La Belle Oreille

“La Belle Oreille” est une entreprise créée par Jeanne Fourel en novembre 2019.

Elle propose la réalisation de reportages audio sur des événements variés pour les particuliers et les entreprises :

- Particuliers :
  - Crémaillères,
  - mariages, noces,
  - Recueil de souvenirs de vie
- Entreprises :
  - Présentation de l'entreprise,
  - d'un procédé de fabrication,
  - visite virtuelle,
  - enregistrement de réunions,
  - reportages sur un moment festif.

# la be))e orei((e

DES VOIX EN MÉMOIRE

Quoi de plus riche et émouvant que  
d'entendre la voix de ceux que l'on aime



La Belle Orelle

Jeanne Fourel

06 81 55 87 76

labeledoreille29@gmail.com

www.labeledoreille.com



La belle oreille



# la be))e orei((e

DES VOIX EN MÉMOIRE

Reportage audio pour particuliers

- > Album sonore d'événements familiaux et amicaux
- > Recueil de souvenirs de vie



## > Album sonore d'événements familiaux et amicaux

A l'occasion d'un **événement familial ou amical**, je me mêle aux festivités avec mon micro et interviewe les invités au sujet de l'événement. Chacun est libre de parler comme il le souhaite de la fête et des personnes fêtées.

Et pour les plus timides, je sais les mettre à l'aise par de petites questions sympathiques... Au cours du montage, j'intègre entre chaque interview de courts extraits musicaux et des sons de l'ambiance de la fête.

Un cadeau original et précieux : un album sonore dans lequel les organisateurs de l'événement auront le plaisir de réentendre la voix de tous ceux et de toutes celles qu'ils aiment et qui auront partagés avec eux ce moment unique.

> Reportage d'environ 1 heure comprenant 10 à 50 interviewés

## > Recueil de souvenirs de vie

Vous avez dans votre entourage ou votre cercle familial une personne qui vous est chère et dont vous voudriez conserver **la voix et quelques souvenirs de vie en mémoire**. Je me propose de réaliser son interview avec ou sans vous.

Au cours du montage, j'intègre de courts extraits musicaux et ambiances sonores reflétant sa personnalité et sa vie.

Un cadeau précieux que vous pourrez conserver toute votre vie : la voix de ceux que vous aimez.

> Reportage d'environ 30 minutes comprenant 1 à 3 interviewés

Flyer de l'entreprise



# Méthode de travail

Jeanne Fourel intervient dans l'événement avec un micro et interviewe les invités au sujet de cet événement.

Elle utilise ensuite un logiciel de traitement audio et réalise un montage, qu'elle propose sous forme de fichier à télécharger ou de clé usb

Ce sont des reportages qui durent de 30 minutes à 1 heure et qui contiennent des interviews de 1 à 50 personnes environ.

# Contexte du projet





# Contexte

J'ai réalisé ce projet seul, en télétravail.

Jeanne Fourel et moi communiquions par email, et je présentais le rendu du site sur une page statique créée sur mon serveur pour l'occasion.

<http://test.yannickbiheul.com/>



# Cahier des charges





# Réalisation du cahier des charges

J'ai créé ce cahier des charges en me basant sur un exemple trouvé sur le net, et après avoir posé les questions nécessaires à Jeanne Fourel.

## Présentation d'ensemble

- Site vitrine, qui présente l'activité de l'entreprise et offre une présence sur le web.
- Destiné aux particuliers et aux entreprises.

## Design

- S'adapter sur mobiles.
- Simple, épuré, intuitif.
- Noir, blanc et doré.



# Contenu / Fonctionnalités

## Contenu du site

- Page d'accueil
- Page prestations
- Page contact
- Page d'actualités
- Galerie photos
- Formulaire d'envoi de fichiers
- Espace membre

## Fonctionnalités

- Espace membre sécurisé par mot de passe
- Envoi de fichiers sécurisé
- Administration

Maquette





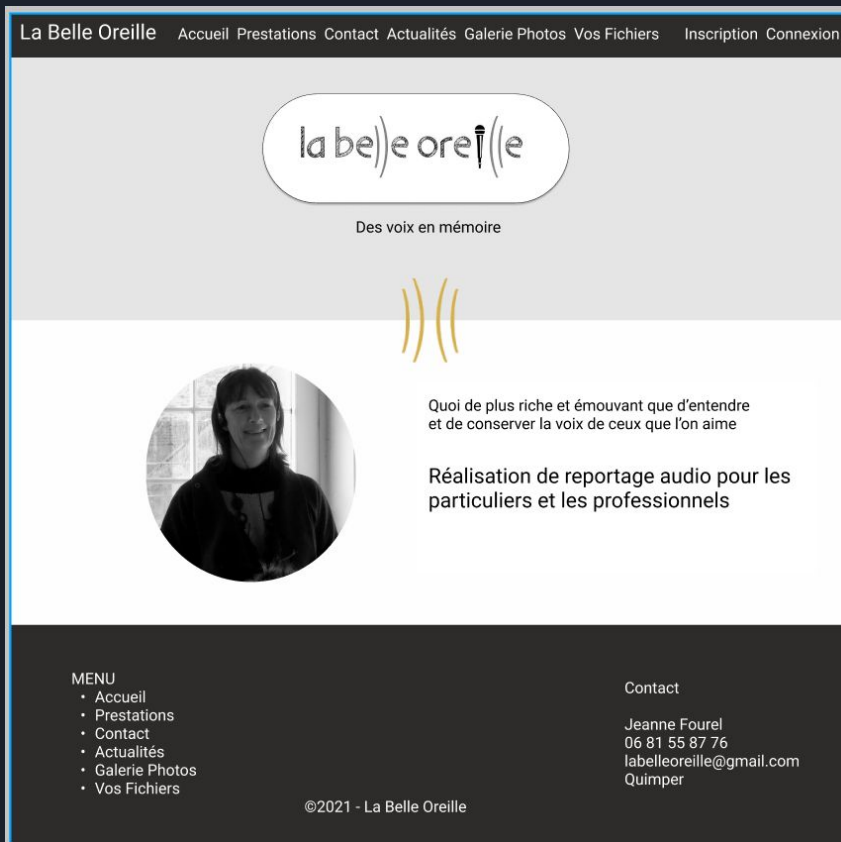
# Création de la maquette

J'ai réalisé la maquette à l'aide d'Adobe XD et Figma.

L'idée était d'avoir un menu de navigation en haut de page, un footer en bas de page et le contenu du site entre ces deux éléments.

La bannière titre comporte une image de fond, non représentée sur la maquette, car modifiable depuis l'espace administration.

Sur mobile, le menu de navigation se retrouve dans un menu "hamburger" créé avec Bootstrap, tandis que la bannière titre prend toute la hauteur de l'écran.



Aperçu de la page d'accueil version Desktop et Mobile

# Modèle Conceptuel de Données





# Les données à utiliser

Le cahier des charges et la maquette m'ont permis d'identifier les données qui seront utilisées sur le site, et de les ranger par tables :

- general
  - pour les principaux éléments du site (image bannière, etc...)
- prestation
- prestation\_categorie
- actualite
- actualite\_categorie
- user
- adresse
  - pour les adresses des utilisateurs
- audio
- photo
- contact
  - pour le formulaire de contact





# Les relations entre les tables

Plusieurs tables sont reliées entre elles, il m'a donc fallu créer des cardinalités :

- Table User et Adresse :
  - relation "One To Many"
  - Un utilisateur peut avoir plusieurs adresses, et une adresse ne peut appartenir qu'à un utilisateur.
- Table User et Audio :
  - relation "One To Many"
- Table Prestation et prestation\_categorie :
  - relation "Many To Many"
  - Une prestation peut avoir plusieurs catégories, et une catégorie peut appartenir à plusieurs prestations.
- Table Actualite et actualite\_categorie :
  - relation "Many To Many"

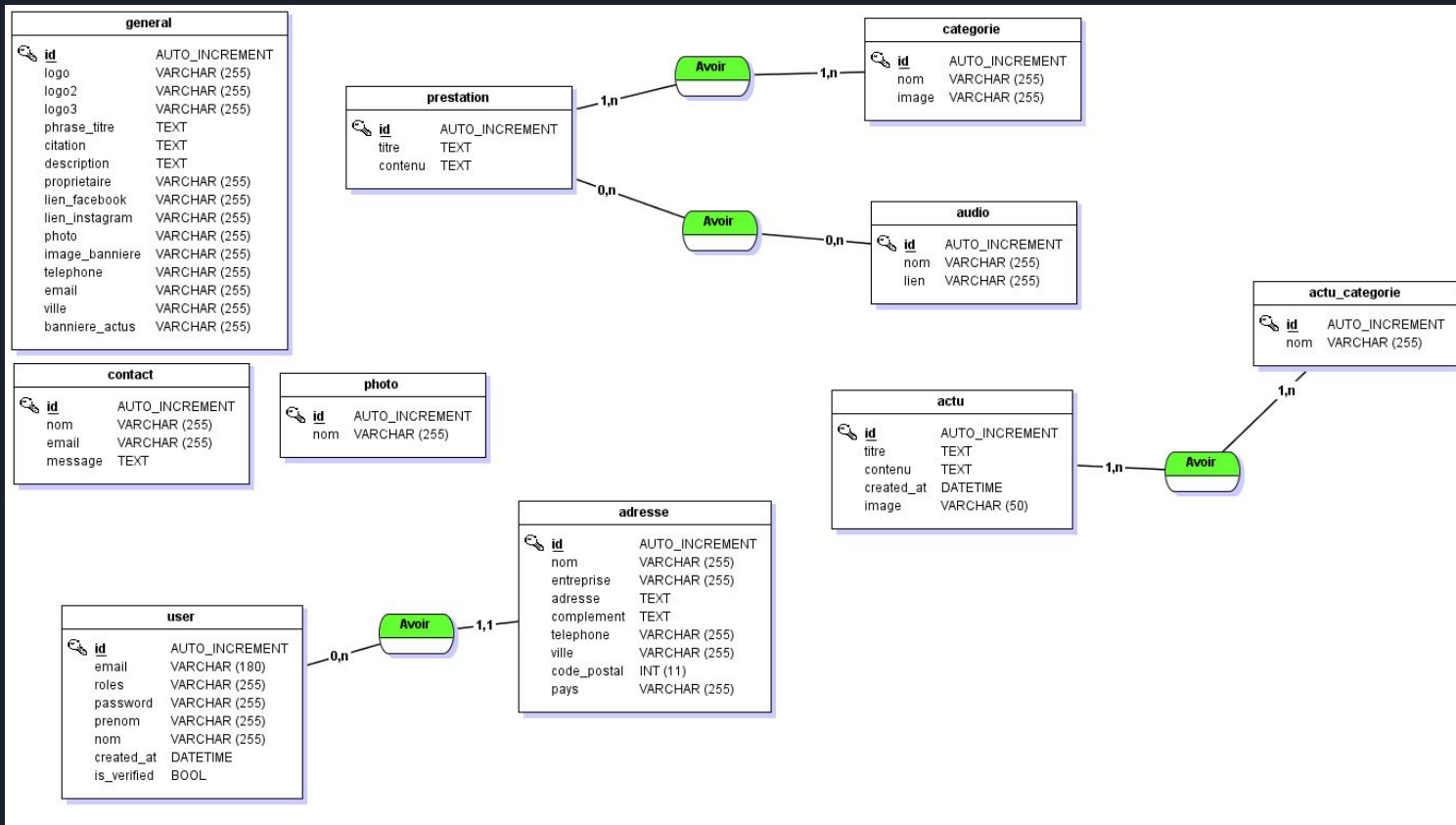


Schéma MCD de la base de données, créé avec jMerise

Environnement de  
développement





# Mes outils de développement

- Système d'exploitation : Windows 10
- Système de Contrôle de Version : [Git / GitHub](#)
- Éditeur de texte : Visual Studio Code
- Serveur MySQL : Laragon
- Terminal de commande : Cmdr
- Création de schéma de données : jMerise
- Création de maquettes : Adobe XD, Figma
- Organisation : [Trello](#)
- Création de documentation : [Notion](#)
- Navigateurs : Google Chrome, Mozilla Firefox
- Gestionnaire de Dépendances PHP : Composer
- Gestionnaire de Paquets Node : NPM
- Commande Symfony : Symfony CLI



# Langages / Librairies / Frameworks

## Langages

- HTML 5
- Twig
- CSS 3
- SCSS
- PHP 7.4

## Librairies

- Animate On Scroll

## Frameworks

- Symfony 5.2
- Bootstrap 5

# Choix des Frameworks





# Frameworks utilisés

## Symfony 5

J'ai choisi Symfony car j'apprécie beaucoup le design pattern MVC et la programmation orientée objet. C'est aussi un framework très demandé, qui apparaît souvent dans les offres d'emploi.

## Bootstrap

Le choix de Bootstrap s'est imposé à moi car je ne voulais pas perdre trop de temps sur le Responsive Design, et qu'il est aussi beaucoup demandé.

# Installation de Symfony







# Symfony 5.2.9

## Pré-requis techniques

La documentation officielle de Symfony recommande :

- d'avoir une version de PHP 7.2.5 minimum,
- d'installer Composer
- d'installer Symfony CLI (optionnel)

Il suffit ensuite de taper la commande :

```
symfony new labelleoreille --full
```

“--full” permet d’installer tous les paquets nécessaires à la construction d’un site web.



# Connexion du projet au repository GitHub

Après avoir créé un repository sur github, ces commandes me permettent d'y envoyer mon projet Symfony :

```
git remote add origin git remote add origin https://github.com/yannickbiheul/labelleoreille.git
```

```
git branch -M main
```

```
git push -u origin main
```

Le projet est envoyé au repository, une branche principale “main” est créée et envoyée elle aussi au repository.

# Modularisation des fichiers de style avec Webpack





# Utilisation du composant Webpack Encore

Ce composant me permet de diviser le style de mes pages en plusieurs fichiers, en SCSS, pour une meilleure maintenabilité.

Ces fichiers sont ensuite compilés en un seul fichier CSS minifié, qui améliorera grandement les temps de chargements des pages.

Création du template





# Structure du template

Le template “base.html.twig” se trouve à la racine du dossier “templates”, là où sont rangées toutes mes vues.

Il contient tout le Doctype HTML :

- La balise “head” avec ses balises “meta”, “title”, “link” et “script”
- La balise “body” qui inclut :
  - Un bloc “nav.html.twig”
  - Un bloc “content”
  - Un bloc “footer.html.twig”

J’ai créé un dossier “base” qui contient les éléments de base : menu de navigation et footer.

Les différentes pages du site se retrouveront dans le bloc “content”

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>
      {% block title %}Welcome!
      {% endblock %}
    </title>
    {% block stylesheets %}
      {{ encore_entry_link_tags('app') }}
      <link rel="preconnect" href="https://fonts.gstatic.com">
      <link href="https://fonts.googleapis.com/css2?family=Montserrat&display=swap" rel="stylesheet">
    {% endblock %}

    {% block javascripts %}
      {{ encore_entry_script_tags('app') }}
      <script src="https://kit.fontawesome.com/29ef46100e.js" crossorigin="anonymous"></script>
    {% endblock %}
  </head>
  <body>
    {% include 'base/nav.html.twig' %}
    <div class="container-fluid" id="content">
      {% block body %}{% endblock %}
    </div>
    {% include 'base/footer.html.twig' %}
  </body>
</html>

```

Aperçu du fichier “base.html.twig”

```

<nav class="navbar navbar-expand-xl fixed-top navbar-dark bg-dark">
  <div class="container-fluid">
    <a class="navbar-brand" href="{{ path('home') }}">
      La Belle Oreille
    </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" href="{{ path('prestation') }}">Prestations</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('contact') }}">Contact</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('actu') }}">Actualités</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('galerie') }}">Galerie Photos</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('fichier') }}">Vos fichiers</a>
        </li>
        {% if is_granted("ROLE_ADMIN") %}
        <li class="nav-item">
          <a class="nav-link" href="{{ path('admin') }}" target="_blank">Admin</a>
        </li>
        {% endif %}
      </ul>

      {% if app.user %}
      <a href="{{ path('account') }}" class="inscription"><i class="fas fa-user"></i>Profil</a>
      {% else %}
      <a href="{{ path('app_register') }}" class="inscription"><i class="fas fa-sign-in-alt"></i>Inscription</a>
      <a href="{{ path('app_login') }}" class="inscription"><i class="fas fa-user"></i>Connexion</a>
      {% endif %}
    </div>
  </div>
</nav>

```

Aperçu du fichier “nav.html.twig”





# Modifications de la barre de navigation

Cette barre comportant beaucoup d'éléments, j'ai du changer la classe Bootstrap de "navbar-expand-md" en "navbar-expand-xl" pour que le bouton hamburger apparaisse sur un écran de 1200px largeur maximale.

J'ai aussi ajouté des media queries pour que les boutons "Inscription" et "Connexion" ne soit pas trop collés aux éléments du menu.

```
@media screen and (max-width: 1200px) {  
  .inscription {  
    margin-top: 20px;  
    margin-bottom: 20px;  
  }  
}
```

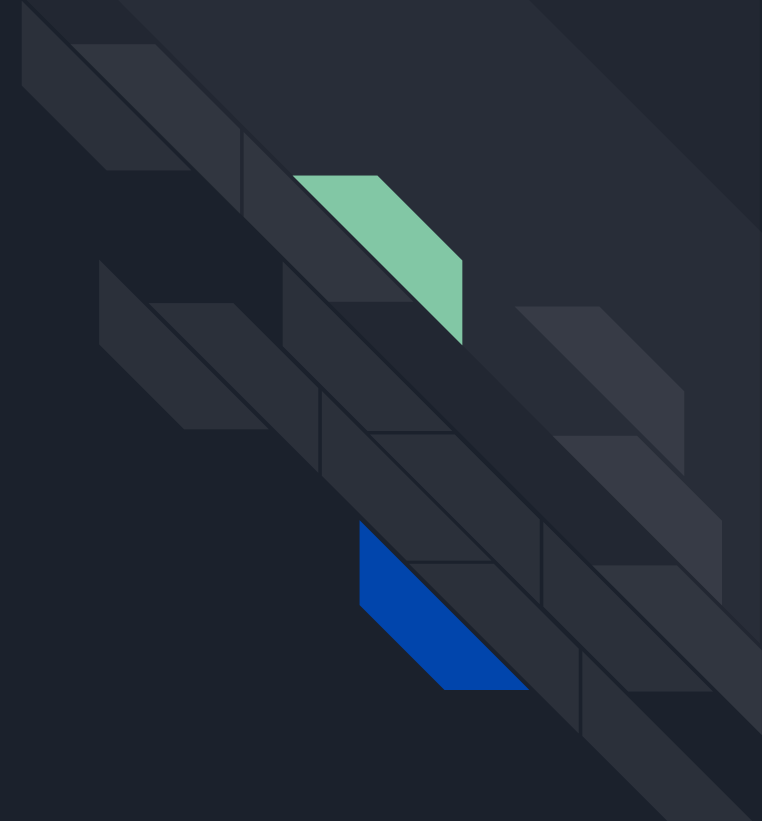
```

<div class="container-fluid" id="footer">
  <div class="row d-flex justify-content-center">
    <div class="footer1 col-md-4 d-flex flex-column justify-content-center align-items-center">
      <h4>Menu</h4>
      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link" href="{{ path('home') }}">Accueil</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('prestation') }}">Prestations</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('contact') }}">Contact</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('actu') }}">Actualités</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ path('galerie') }}">Galerie photos</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Vos fichiers</a>
        </li>
      </ul>
    </div>
    <div class="footer2 col-md-4 d-flex flex-column justify-content-end align-items-center">
      ©2021 | La Belle Oreille
    </div>
    <div class="footer3 col-md-4 d-flex flex-column justify-content-center align-items-center">
      <h4>Contact</h4>
      <h5>{{ general.proprietaire }}</h5>
      <p>{{ general.telephone }}</p>
      <p>{{ general.email }}</p>
      <p>{{ general.ville }}</p>
      <div class="reseaux">
        <a href="{{ general.lienFacebook }}" target="_blank"><i class="fab fa-facebook-f"></i></a>
        <a href="{{ general.lienInstagram }}" target="_blank"><i class="fab fa-instagram"></i></a>
      </div>
    </div>
  </div>
</div>

```

Aperçu du fichier "footer.html.twig"

Création, récupération  
et affichage des  
données : la page  
d'accueil





# Création de la base de données

Le fichier “.env” à la racine de mon projet me sert, entre autres, à créer une base de données.

Je lui indique que j'utilise du MySQL, l'identifiant et le mot de passe, le port où se trouve MySQL et le nom de la base de données, ainsi que la version du serveur.

```
###> doctrine/doctrine-bundle ###  
# Format described at https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/configuration.html#connecting-using-a-url  
# IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.yaml  
#  
# DATABASE_URL="sqlite:///kernel.project_dir%/var/data.db"  
DATABASE_URL="mysql://root:root@127.0.0.1:3306/labelleoreille3_dev?serverVersion=5.7"  
# DATABASE_URL="postgresql://db_user:db_password@127.0.0.1:5432/db_name?serverVersion=13&charset=utf8"  
###< doctrine/doctrine-bundle ###
```



# Création de l'entité “general”

À l'aide de la commande Symfony CLI, je créer l'entité “general”

```
symfony console make:entity
```

J'ajoute les champs nécessaires (exemple avec la phrase\_titre):

```
new property name : phrase_titre
```

```
Field Type : Text
```

```
Can this field be null in the database (nullable) : no
```

Je réponds non ici car le champ doit être défini.

```
updated : src/Entity/general.php
```

Le fichier “general.php” est créé dans le dossier “Entity” du dossier “src”.

Je fais ensuite une migration, que j'envoie à la base de données

```
symfony console make:migration
```

```
symfony console doctrine:migrations:migrate
```

```

namespace App\Entity;

use App\Repository\GeneralRepository;
use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity(repositoryClass=GeneralRepository::class)
 */
class General
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue
     * @ORM\Column(type="integer")
     */
    private $id;


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

    /**
     * @ORM\Column(type="string", length=255)
     */
    private $logo2;

    /**
     * @ORM\Column(type="string", length=255)
     */
    private $logo3;

    /**
     * @ORM\Column(type="text")
     */
    private $phraseTitre;

```

| # | Nom  | Type         | Interclassement    | Attributs | Null |
|---|--|--------------|--------------------|-----------|------|
| 1 | id  | int(11)      |                    |           | Non  |
| 2 | logo   | varchar(255) | utf8mb4_unicode_ci |           | Non  |
| 3 | logo2  | varchar(255) | utf8mb4_unicode_ci |           | Non  |
| 4 | logo3  | varchar(255) | utf8mb4_unicode_ci |           | Non  |
| 5 | phrase_titre   | longtext     | utf8mb4_unicode_ci |           | Non  |
| 6 | citation   | longtext     | utf8mb4_unicode_ci |           | Non  |
| 7 | description  | longtext     | utf8mb4_unicode_ci |           | Non  |
| 8 | proprietaire   | varchar(255) | utf8mb4_unicode_ci |           | Non  |

Aperçu du fichier "general.php" et table "general" dans phpmyadmin



# Ajout de données avec les Fixtures

Installation du composant orm-fixtures

`composer require orm-fixtures`

Le fichier "AppFixtures.php" est créé dans le dossier "DataFixtures"

Ce fichier contient une méthode "load", qui va me permettre d'enregistrer des données dans la base.

- J'instancie un objet de la classe "general"
- Je définis les différents champs de cet objet
- Je "persiste" ces données (équivalent du "commit" de Git)
- J'envoie ces données à la base avec un flush (équivalent du "push" de Git)

```
public function load(ObjectManager $manager)
{
    // GENERAL
    $general = new General;

    $general->setLogo('logo.png')
        ->setLogo2('logo2.png')
        ->setLogo3('logo3.png')
        ->setImageBanniere('micro3.jpg')
        ->setBanniereActus('flou.jpg')
        ->setPhraseTitre('Des voix en mémoire')
        ->setCitation('Quoi de plus riche et émouvant que d'entendre
        et de conserver la voix de ceux que l'on aime')
        ->setDescription('Réalisation de reportage audio pour les particuliers et les professionnels')
        ->setlienFacebook('https://www.facebook.com/La-Belle-Oreille-101656721318094/?modal=admin_todo_tour')
        ->setlienInstagram('https://www.instagram.com/foureljeanne/')
        ->setProprietaire('Jeanne Fourel')
        ->setTelephone('06 81 55 87 76')
        ->setEmail('labelleoreille29@gmail.com')
        ->setVille('Quimper')
        ->setPhoto('essai_presentation.png');

    $manager->persist($general);
}
```

Aperçu du fichier "AppFixtures.php"





# Création de la page d'accueil

Je créer d'abord un Controller, que je nomme "HomeController" à l'aide de la commande symfony :

```
symfony console make:controller HomeController
```

Le controller se trouve dans le dossier "Controller" du dossier "src", je lui spécifie une route dans les annotations de sa fonction "index" pour qu'il "s'active" quand la racine du site est appelée dans l'adresse URL.

Quand l'entité "general" a été créée tout à l'heure, Symfony a généré automatiquement le fichier "GeneralRepository", qui va me permettre de récupérer les données de cette table.

```

namespace App\Repository;

use App\Entity\General;
use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
use Doctrine\Persistence\ManagerRegistry;

/**
 * @method General|null find($id, $lockMode = null, $lockVersion = null)
 * @method General|null findOneBy(array $criteria, array $orderBy = null)
 * @method General[]    findAll()
 * @method General[]    findBy(array $criteria, array $orderBy = null, $limit = null, $offset = null)
 */
class GeneralRepository extends ServiceEntityRepository
{
    public function __construct(ManagerRegistry $registry)
    {
        parent::__construct($registry, General::class);
    }
}

```

Aperçu du fichier "GeneralRepository.php"

```
namespace App\Controller;

use App\Entity\General;
use App\Repository\ActuRepository;
use App\Repository\UserRepository;
use App\Repository\GeneralRepository;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

class HomeController extends AbstractController
{
    private $generalRepository;
    private $actuRepository;

    public function __construct(GeneralRepository $generalRepository, ActuRepository $actuRepository)
    {
        $this->generalRepository = $generalRepository;
        $this->actuRepository = $actuRepository;
    }

    /**
     * @Route("/", name="home")
     */
    public function index(): Response
    {
        $general = $this->generalRepository->findOneBy(['proprietaire' => 'Jeanne Fourel']);
        $page = "Accueil";
        $actus = $this->actuRepository->findAll();

        return $this->render('home/index.html.twig', compact('general', 'page', 'actus'));
    }
}
```

Aperçu du fichier "HomeController.php"



# Récupération des données

J'ajoute l'attribut "generalRepository" au Controller.

Je l'initie dans le constructeur avec un objet de la classe "GeneralRepository".

Sur cet attribut, qui possède maintenant les méthodes de son Repository, j'exécute la fonction "findOneBy()" qui me permet de récupérer les données voulues (ici celles dont le champ "propriétaire" correspond à la valeur "Jeanne Fourel").

Je crée aussi une variable "page" qui contient le titre de la page.

Je retourne une fonction render() qui prend en paramètres le fichier servant à afficher la page et les données à lui envoyer.



# Affichage des données

Le fichier “index.html.twig” du dossier “home” inclut différentes sections de la page d’accueil, que j’ai mises dans des fichiers séparés pour une meilleure maintenabilité.

Il récupère la donnée “page” dans son titre, qui s’affiche dans l’onglet du navigateur.

La bannière récupère son image de fond, ses logos, la phrase titre et le titre de la page.

Ici, le style CSS de la bannière est placé dans la balise HTML, car l’image est récupérée grâce à la donnée “general.imageBanniere”.

L’option “raw” est ajoutée dans les balises Twig car ces données sont susceptibles d’être modifiées dans l’espace administration avec EasyAdmin, qui génère des balises HTML.

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

{% block title %}La Belle Oreille | {{ page }} {% endblock %}

{% block body %}
    {% include 'home/banniere.html.twig' %}
    {% include 'home/presentation.html.twig' %}
    {% include 'home/actus.html.twig' %}
    {% include 'home/banniere_reseaux.html.twig' %}
{% endblock %}
```

```
<div class="container-fluid" id="banniere" style="
background: linear-gradient(45deg, rgba(0, 0, 0, 0.2)50%, rgba(0, 0, 0, 0.2)50%), url(/backgrounds/{{ general.imageBanniere | raw }});
background-size: cover;
background-position: center;
background-attachment: fixed;">
    <div class="row d-flex justify-content-center align-items-center">
        <div class="col-lg-6 d-flex flex-column justify-content-center align-items-center">
            
            <p class="phraseTitre">{{ general.phraseTitre | raw }}</p>
            
            <p class="page">{{ page }}</p>
        </div>
    </div>
</div>
```

Aperçus des fichiers “index.html.twig” et “banniere.html.twig”.

# la be))e orei!((e

Des voix en mémoire



*Quoi de plus riche et émouvant que d'entendre et de conserver la  
voix de ceux que l'on aime*

*Jeanne Fourel*

Réalisation de reportage audio pour les  
particuliers et les professionnels

[Accueil](#)

Aperçu de la page d'accueil. J'ai utilisé la librairie "Animate On Scroll" pour animer les différents éléments de la page.

# Solution de gestion de contenu







# EasyAdmin

J'ai choisi EasyAdmin car il me permettait de gérer facilement le CRUD (Create, Read, Update, Delete) des différentes données en me proposant une interface de gestion.

Il intégrait aussi un éditeur de texte pour ajouter du contenu.

L'installation de ce composant s'est faite avec Composer :

```
composer require admin
```

La création du Dashboard s'est faite, elle, avec la commande Symfony :

```
symfony console make:admin:dashboard
```

Je l'ai nommé "DashboardController" et placé dans le dossier "admin" du dossier "src".

 Accueil


GÉNÉRAL

 Général

PRESTATIONS

 Prestations Catégories

ACTUALITÉS

 Actualités Catégories

AUDIOS

 Audios

PHOTOS

 Photos

UTILISATEURS

 Utilisateurs

# Administration

Bienvenue dans l'interface d'administration de La Belle Oreille !

## Fonctionnement global

### Dans chaque section :

- Pour ajouter un élément, cliquer sur le bouton bleu "Add ..." en haut à droite.
- Pour éditer un élément, cliquer sur "Edit" au bout de la ligne le concernant.
- Pour supprimer un élément, cliquer sur "Delete" au bout de la ligne le concernant.

### L'éditeur de texte



Les boutons, de gauche à droite :

- **B** permet de passer le texte sélectionné en gras.
- **I** permet de passer le texte sélectionné en italique.
- **ABC** permet de barrer le texte sélectionné.
- **chain** permet d'insérer un lien.
- **T** permet de changer le texte sélectionné en titre et inversement.
- **double quotes** permet d'insérer des guillemets.
- **code block** permet d'insérer du code.
- **list** permet d'insérer une liste à puces.
- **list** permet d'insérer une liste numérotée.
- **list** permet d'ajouter une liste dans une liste.
- **list** permet de revenir dans la liste normale.



# Ajouter un CRUD

L'ajout de CRUD sur une entité se fait via la commande Symfony :

```
symfony console make:admin:crud
```

On choisit l'entité dans la liste proposée (ici l'entité "general"), on définit l'emplacement du CRUD, et son namespace.

On l'ajoute ensuite au fichier "DashboardController.php" pour l'afficher dans le dashboard d'EasyAdmin.

Enfin, on complète la fonction "configureFields" du fichier "GeneralCrudController.php" pour afficher les champs voulus.

```

class DashboardController extends AbstractDashboardController
{
    /**
     * @Route("/admin", name="admin")
     */
    public function index(): Response
    {
        // redirect to some CRUD controller
        // $routeBuilder = $this->get(AdminUrlGenerator::class);

        // return $this->redirect($routeBuilder->setController(GeneralCrudController::class)->generateUrl());
        return $this->render('admin/dashboard.html.twig');
    }

    public function configureDashboard(): Dashboard
    {
        return Dashboard::new()
            ->setTitle('La Belle Oreille | Administration');
    }

    public function configureMenuItems(): iterable
    {
        yield MenuItem::linktoDashboard('Accueil', 'fa fa-home');

        yield MenuItem::section('Général');
        yield MenuItem::linkToCrud('Général', 'fas fa-list', General::class);
    }
}

```

Aperçu du fichier "DashboardController.php" avec l'affichage du CRUD "general".

```

public function configureFields(string $pageName): iterable
{
    return [
        IdField::new('id')->hideOnForm(),
        TextEditorField::new('phraseTitre'),
        TextEditorField::new('citation'),
        TextEditorField::new('description'),
        TextField::new('proprietaire'),
        TextField::new('telephone'),
        TextField::new('email'),
        TextField::new('ville'),
        ImageField::new('photo')->setUploadDir("public/backgrounds")
                                ->setBasePath("/backgrounds")
                                ->setRequired(false),
        ImageField::new('imageBanniere')->setUploadDir("public/backgrounds")
                                        ->setBasePath("/backgrounds")
                                        ->setRequired(false),
        ImageField::new('banniereActus')->setUploadDir("public/backgrounds")
                                       ->setBasePath("/backgrounds")
                                       ->setRequired(false),
    ];
}

```

La fonction "configureFields" du fichier "GeneralCrudController.php"

[Accueil](#)

## General

[Add General](#)

GÉNÉRAL

[Général](#)

PRESTATIONS

[Prestations](#)[Catégories](#)

ACTUALITÉS

[Actualités](#)[Catégories](#)

AUDIOS


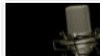

[Audios](#)

PHOTOS

[Photos](#)

UTILISATEURS

[Utilisateurs](#)

| <input type="checkbox"/> | ID | Phrase Titre                 | Citation                     | Description                  | Proprietaire  | Telephone      | Email                      | Ville   | Photo   | Image Banniere  | Banniere Actus  |  |
|--------------------------|----|------------------------------|------------------------------|------------------------------|---------------|----------------|----------------------------|---------|---|---|---|--|
| <input type="checkbox"/> | 1  | <a href="#">View content</a> | <a href="#">View content</a> | <a href="#">View content</a> | Jeanne Fourel | 06 81 55 87 76 | labelleoreille29@gmail.com | Quimper |  |  |  | <a href="#">Edit</a><br><a href="#">Delete</a> |

1 result

[< Previous](#)

1

[Next >](#)

## GÉNÉRAL

## Général

## PRESTATIONS

## Prestations

## Catégories

## ACTUALITÉS

## Actualités

## Catégories

## AUDIOS













## Audios

## PHOTOS

## Photos

## UTILISATEURS

## Utilisateurs

|                |   |  |   |
|----------------|---|--|---|
| Phrase Titre * | <b>B</b> <i>I</i> <u>S</u> <a>Link</a>  | <b>¶</b> <b>”</b> <b>&lt; &gt;</b> <b>≡</b> <b>≡</b> <b>≡</b> <b>≡</b> |   |
|                | Des voix en mémoire   |  |   |
| Citation *     | <b>B</b> <i>I</i> <u>S</u> <a>Link</a>  | <b>¶</b> <b>”</b> <b>&lt; &gt;</b> <b>≡</b> <b>≡</b> <b>≡</b> <b>≡</b> |   |
|                | Quoi de plus riche et émouvant que d'entendre et de conserver la voix de ceux que l'on aime |  |   |
| Description *  | <b>B</b> <i>I</i> <u>S</u> <a>Link</a>  | <b>¶</b> <b>”</b> <b>&lt; &gt;</b> <b>≡</b> <b>≡</b> <b>≡</b> <b>≡</b> |   |
|                | Réalisation de reportage audio pour les particuliers et les professionnels                  |  |   |
| Propriétaire * | <input type="text" value="Jeanne Fourel"/>  |  |   |
| Telephone *    | <input type="text" value="06 81 55 87 76"/>   |  |   |
| Email *        | <input type="text" value="labelleoreille29@gmail.com"/>                                     |  |   |
| Ville *        | <input type="text" value="Quimper"/>  |  |   |
| Photo          | <input type="text" value="essai_presentation.png"/>   | <input type="text" value="234K"/>                                      |   |
| Image Banniere | <input type="text" value="micro3.jpg"/>   | <input type="text" value="212K"/>                                      |   |
| Banniere Actus | <input type="text" value="flou.jpg"/>   | <input type="text" value="162K"/>                                      |   |

Formulaire de modification des données de la table “general”

# Tests Unitaires







# Tests Unitaires sur chaque entité

Pour chaque entité créée, je réalise aussi son fichier de tests unitaires, pour vérifier qu'on récupère bien les données attendues.

Pour cela, j'utilise "phpunit".

Exemple avec l'entité "contact" :

La commande symfony suivante permet de créer le fichier de tests :

**symfony console make:unit-test**

Je lui donne le nom "ContactUnitTest"

Ce fichier contient les 3 tests suivants :

```
public function testIsTrue()
{
    $contact = new Contact;

    $contact->setEmail('email@test.com')
        ->setMessage('message')
        ->setNom('nom');

    $this->assertTrue($contact->getEmail() === 'email@test.com');
    $this->assertTrue($contact->getMessage() === 'message');
    $this->assertTrue($contact->getNom() === 'nom');
}
```

testIsTrue() détecte quand les données récupérées sont bien celles que l'on a rentrées.

```
public function testIsFalse()
{
    $contact = new Contact;

    $contact->setEmail('email@test.com')
        ->setMessage('message')
        ->setNom('nom');

    $this->assertFalse($contact->getEmail() === 'false');
    $this->assertFalse($contact->getMessage() === 'false');
    $this->assertFalse($contact->getNom() === 'false');
}
```

testIsFalse() détecte quand les données récupérées ne correspondent pas à celles que l'on a rentrées.

```
public function testIsEmpty()  
{  
    $contact = new Contact;  
  
    $this->assertEmpty($contact->getEmail());  
    $this->assertEmpty($contact->getMessage());  
    $this->assertEmpty($contact->getNom());  
}
```

testIsEmpty() détecte quand on récupère des données qui n'ont pas été rentrées.

```
php bin/phpunit --testdox
PHPUnit 8.5.15 by Sebastian Bergmann and contributors.
```

```
Testing Project Test Suite
Contact Unit (App\Tests>ContactUnit)
```

```
✓ Is true
✓ Is false
✓ Is empty
```

```
Time: 276 ms, Memory: 8.00 MB
```

```
OK (3 tests, 15 assertions)
```

En lançant la commande “**php bin/phpunit --testdox**”, les tests sont effectués sur l’entité et les résultats apparaissent sur le terminal de commande.

# Intégration continue avec les actions GitHub





# Automatisation des tests

Les actions GitHub me permettent d'automatiser plusieurs tests.

Ces tests sont créés dans un fichier "main.yml" du dossier ".github\workflows" ajouté à la racine du projet.

J'ai trouvé un fichier de base sur Github, et j'y ai ajouté les tests que je souhaitais effectuer.

Je demande au fichier main.yml d'effectuer ces tests quand je fais un "push" sur la branche "main".

Ces tests sont : la vérification de la corrélation entre les fichiers "composer.json" et "composer.lock", la vérification des vulnérabilités connues des dépendances installées, et les tests unitaires.

```
name: Tests
on:
  push:
    branches: [ main ]
  workflow_dispatch:
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2

      # Mise en cache des dépendances
      - name: Get composer cache directory
        id: composer-cache
        run: echo "::set-output name=dir::$(composer config cache-files-dir)"

      - name: Cache dependencies
        uses: actions/cache@v2
        with:
          path: ${{ steps.composer-cache.outputs.dir }}
          key: ${{ runner.os }}-composer-${{ hashFiles('**/composer.lock') }}
          restore-keys: ${{ runner.os }}-composer-

      # Installation des dépendances
      - name: Install dependencies
        run: composer install --prefer-dist

      # Validation composer.json et composer.lock
      - name: Validation composer.json et composer.lock
        run: composer validate

      # Vérification des vulnérabilités connues des dépendances
      - uses: symfonycorp/security-checker-action@v2

      # Tests Unitaires
      - name: Tests Unitaires
        run: php bin/phpunit --testdox
```

Aperçu du fichier “main.yml”



✓ Validation composer.json et composer.lock

```
1 ▼ Run composer validate
2  composer validate
3  shell: /usr/bin/bash -e {0}
4
5  ./composer.json is valid, but with a few warnings
6  See https://getcomposer.org/doc/04-schema.md for details on the schema
7  # General warnings
8  - require.composer/package-versions-deprecated : exact version constraints (1.11.99.2) should be avoided if the package follows semantic versioning
```

Vérification de la corrélation entre “composer.json” et “composer.lock” sur GitHub

```

1 ▶ Run symfonycorp/security-checker-action@v2
5 /usr/bin/docker run --name symfonycorpcilatest_1aa0de --label 8a33c1 --workdir /github/workspace --rm -e INPUT_LOCK -e INPUT_DISABLE-EXIT-CODE -e HOME -e GITHUB_JOB -e GITHUB_REF -e GITHUB_SHA -e GITHUB_REPOSITORY -e
GITHUB_REPOSITORY_OWNER -e GITHUB_RUN_ID -e GITHUB_RUN_NUMBER -e GITHUB_RETENTION_DAYS -e GITHUB_ACTOR -e GITHUB_WORKFLOW -e GITHUB_HEAD_REF -e GITHUB_BASE_REF -e GITHUB_EVENT_NAME -e GITHUB_SERVER_URL -e GITHUB_API_URL -e
GITHUB_GRAPHQL_URL -e GITHUB_WORKSPACE -e GITHUB_ACTION -e GITHUB_EVENT_PATH -e GITHUB_ACTION_REPOSITORY -e GITHUB_ACTION_REF -e GITHUB_PATH -e GITHUB_ENV -e RUNNER_OS -e RUNNER_TOOL_CACHE -e RUNNER_TEMP -e RUNNER_WORKSPACE
-e ACTIONS_RUNTIME_URL -e ACTIONS_RUNTIME_TOKEN -e ACTIONS_CACHE_URL -e GITHUB_ACTIONS=true -e CI=true -v "/var/run/docker.sock":"/var/run/docker.sock" -v "/home/runner/work/_temp/_github_home":"/github/home" -v
"/home/runner/work/_temp/_github_workflow":"/github/workflow" -v "/home/runner/work/_temp/_runner_file_commands":"/github/file_commands" -v "/home/runner/work/labelleoreille3/labelleoreille3":"/github/workspace"
symfonycorp/cli:latest "check:security" "--dir" "./composer.lock" "--disable-exit-code=0"

7 Symfony Security Check Report
8 =====
9
10 No packages have known vulnerabilities.
11
12 Note that this checker can only detect vulnerabilities that are referenced in the security advisories database.
13 Execute this command regularly to check the newly discovered vulnerabilities.

```

Vérification des vulnérabilités connues des dépendances sur GitHub

```

104
105 General Unit (App\Tests\GeneralUnit)
106   ✓ Is true
107   ✓ Is false
108   ✓ Is empty
109
110 Photo Unit (App\Tests\PhotoUnit)
111   ✓ Is true
112   ✓ Is false
113   ✓ Is empty
114
115 Prestation Unit (App\Tests\PrestationUnit)
116   ✓ Is true
117   ✓ Is false
118   ✓ Is empty
119
120 User Unit (App\Tests\UserUnit)
121   ✓ Is true
122   ✓ Is false
123   ✓ Is empty
124
125 Time: 9.53 seconds, Memory: 8.00 MB
126
127 OK (24 tests, 90 assertions)
```

Tests unitaires sur GitHub

# Démonstration du site

Inscription d'un utilisateur



Conclusion





# Ce que le stage m'a apporté

Ce stage m'a permis d'avoir une vision d'ensemble de la création d'un site web, du cahier des charges au code en passant par la maquette et la documentation.

J'ai beaucoup apprécié travailler sur le Framework Symfony, et souhaiterais vraiment continuer dans ce domaine.

Je sais maintenant que le plus important dans un projet n'est pas forcément le code, et si je devais le refaire, je travaillerais plus sur le cahier des charges, la maquette, et le modèle conceptuel de données.



# Les difficultés rencontrées

J'ai d'abord voulu utiliser docker pour "containeriser" mon application et ma base de données, mais l'installation de cet outil sur Windows m'a posé trop de problèmes, n'étant en plus pas encore habitué au Framework Symfony.

L'intégration continue m'a aussi posé des problèmes, voulant la faire au départ sur GitLab, mais j'ai réussi à trouver comment l'utiliser sur GitHub, notamment grâce à ses "actions".

Je ne savais pas bien encore utiliser les migrations en Symfony, ce qui fait que dès que j'avais une erreur dans ma base de données, je recommençais le projet à zéro.



## À l'avenir...

Je dois encore apprendre à envoyer mon application sur un serveur pour mettre le site en ligne.

Je voudrais aussi approfondir le composant “Security” de Symfony. Bien qu’il soit installé sur mon site, et qu’il fonctionne, je ne connais pas exactement son fonctionnement.

Concernant mes projets personnels, je travaille actuellement sur un [blog](#) en PHP/MVC Objet, sans Framework pour le back-end, et Bootstrap pour le front-end.

J’ai aussi travaillé sur le site [Pied de Vigne](#) que l’on a créé en groupe pendant la formation, en PHP/MVC Objet lui aussi, mais sans aucun Framework.