

Formation : Développeur Web et Web Mobile

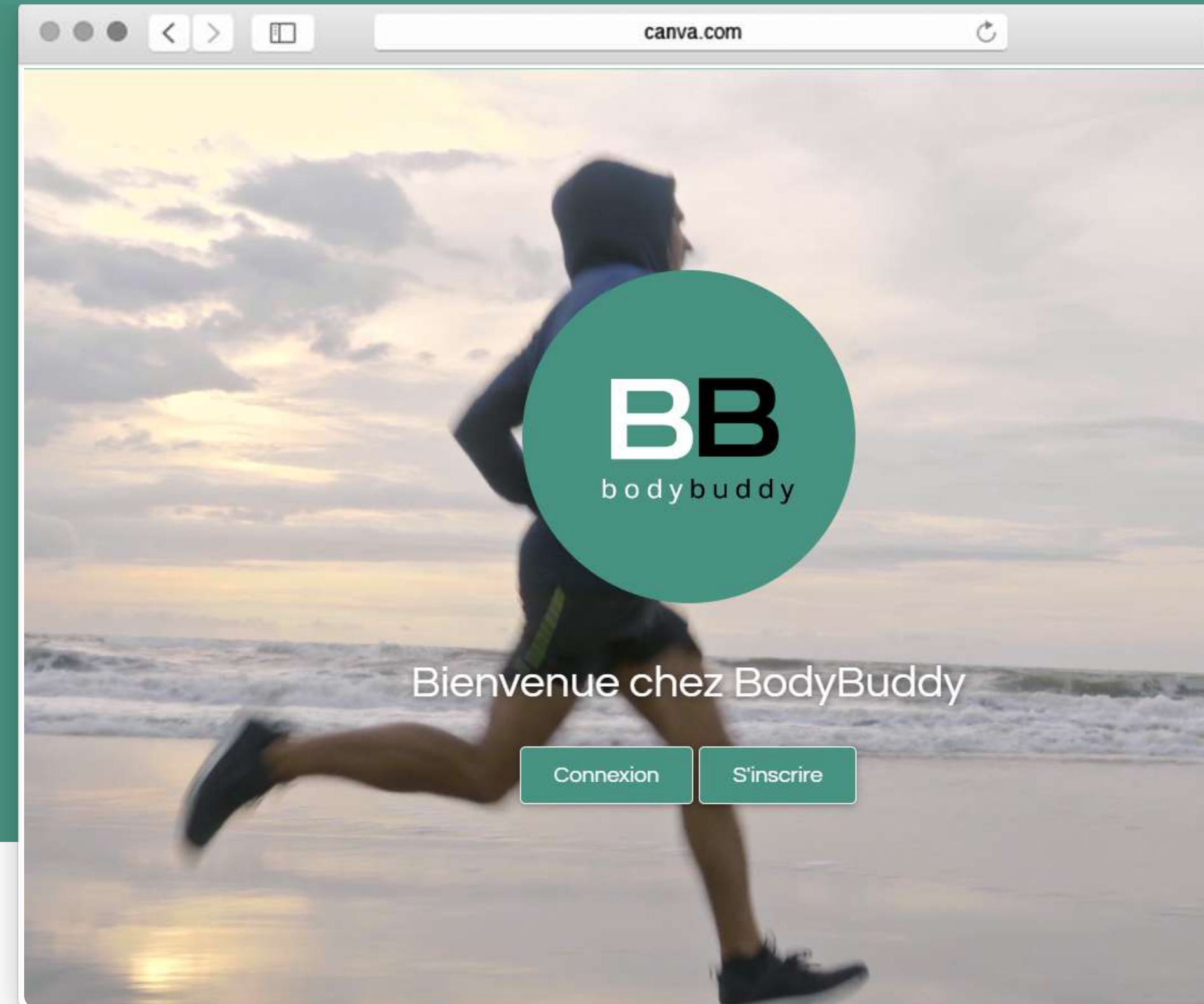
Soutenance : projet BodyBuddy

Formateurs : Stéphane PONTONNIER - Guillaume DELACROIX

Tuteur de stage : Alexandre Achain

AFPA - Rochefort

2021 /2022



Fatiha SADEQ



Fatiha SADEQ

24 ans

La Rochelle

DUT Techniques de commercialisation

Licence 3 Economie – Droit – Gestion

Service civique – Ecole maternelle / primaire

Documentaliste / Animatrice



Code Expérience

- Freelance (avril 2018)
- Développeur Web PHP : Symfony
Laravel...
- Tous secteurs confondus

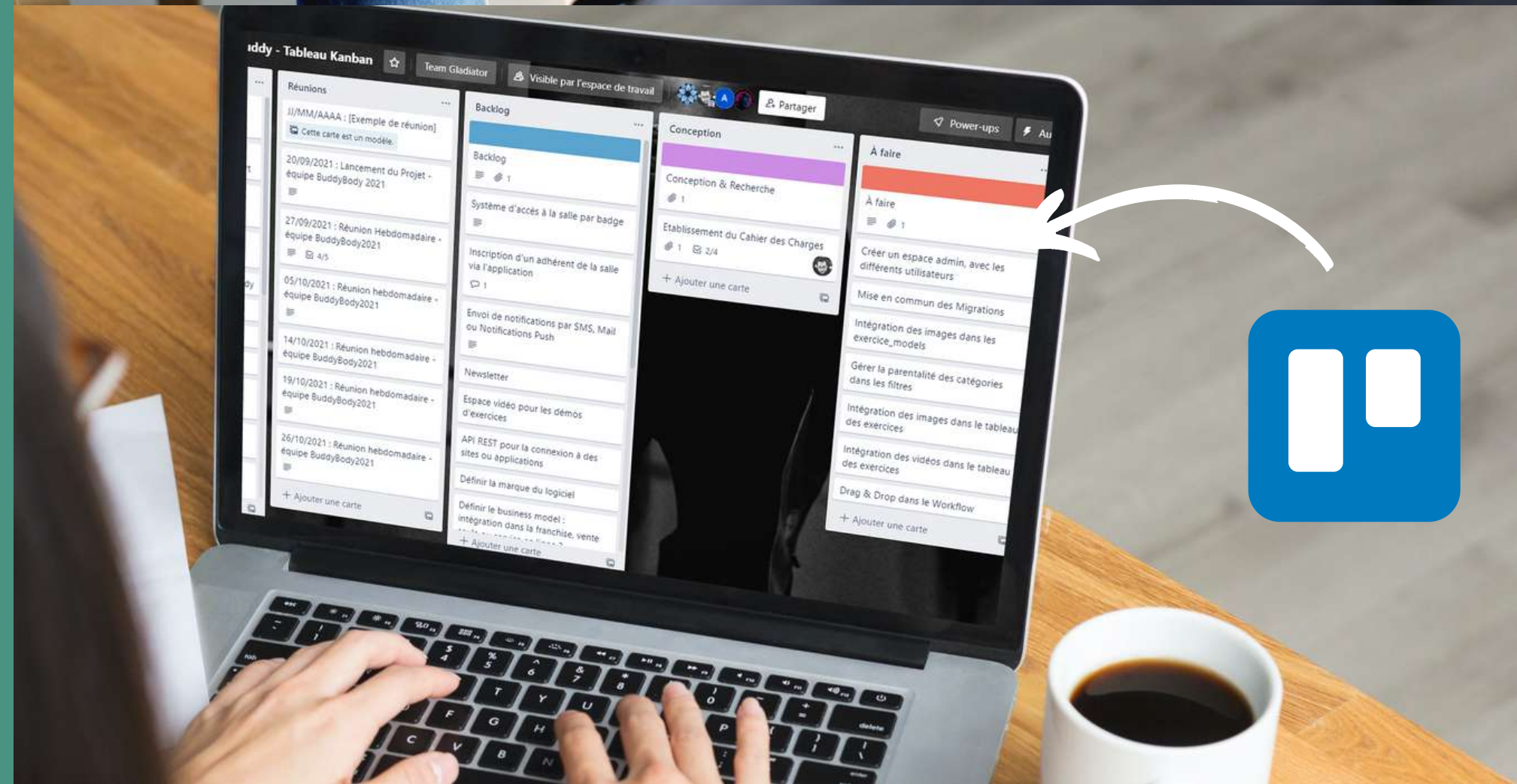


Tuteur de stage



2 collègues de l'AFPA

En distanciel



Le versioning



Push commandes

git add .

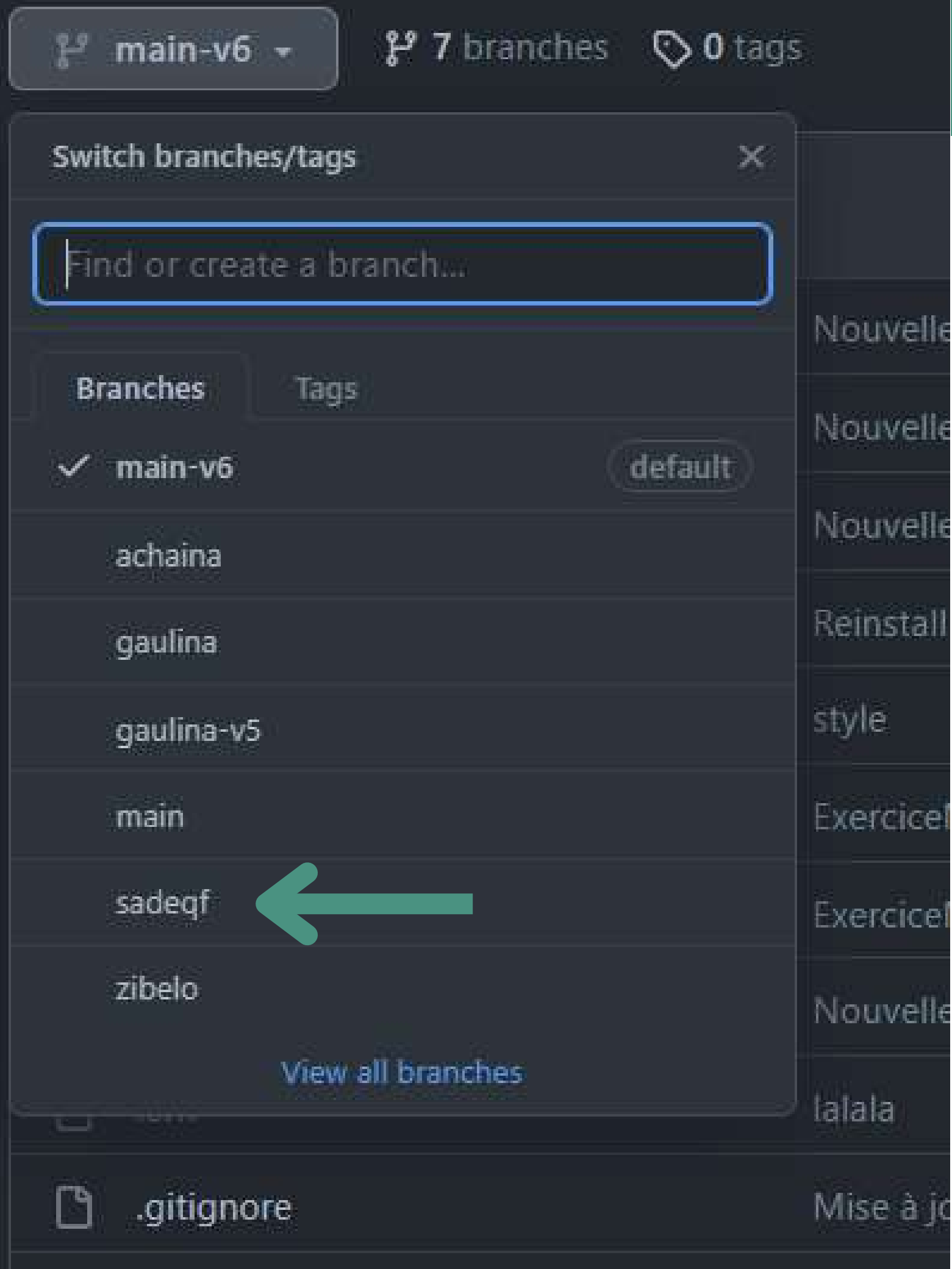
git commit -m "mon commentaire"

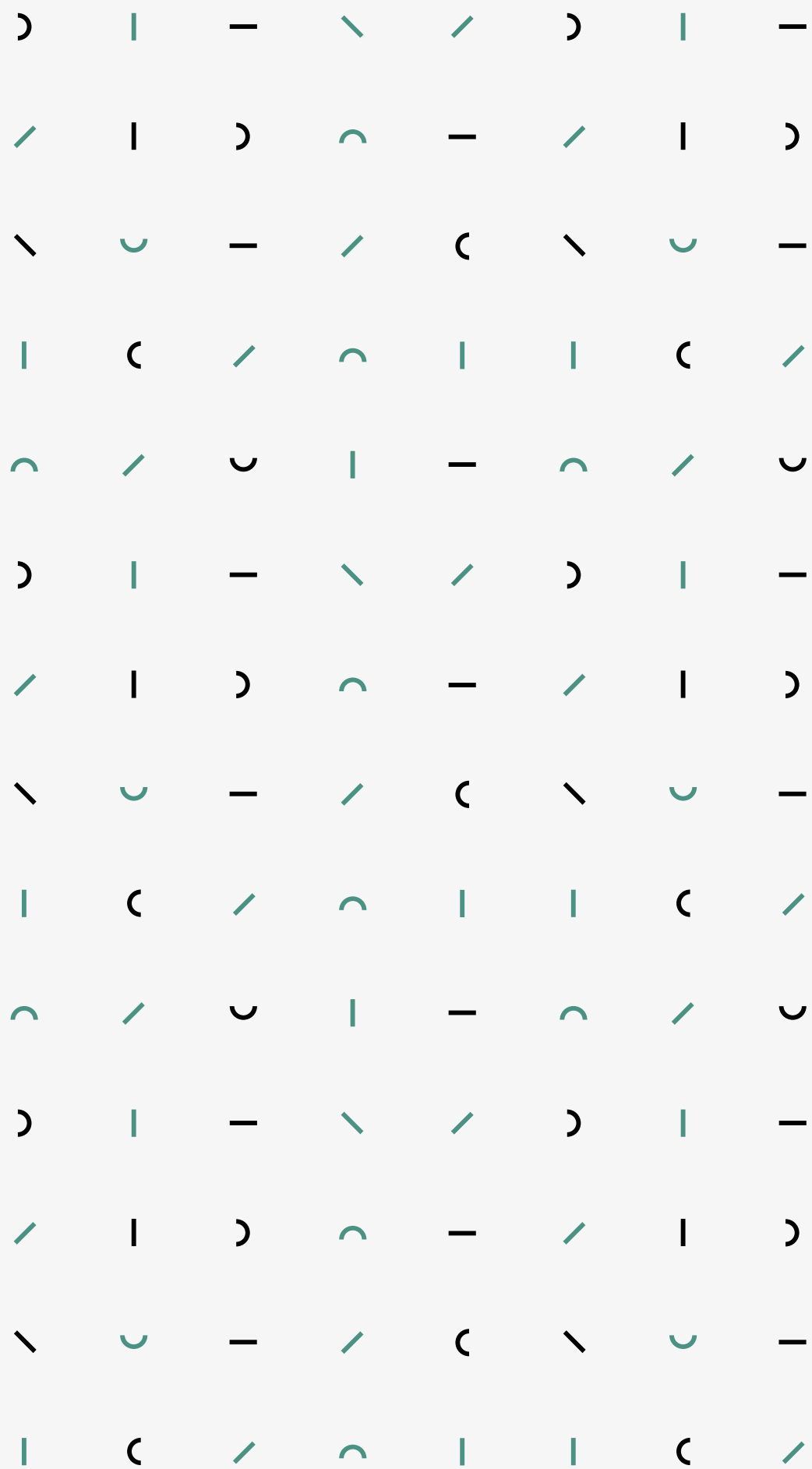
git push

Pull commandes

git pull

git pull origin nom de la branche





Tables des matières

1. Contexte du projet

2. Front

3. Back

4. Jeu d'essai

5. Bilan

Contexte du projet



- Application de création de parcours sportif
- Créer des programmes personnalisés
- Symfony 5.4
- Montée de niveau (version 6)

Contexte du projet

Cible : Coachs sportifs

Objectif : Répondre aux besoins commerciaux, de gestion et métiers pour les coachs sportifs individuels et les salles de sports à taille humaine

Contraintes techniques : Fonctionner sur le web à travers un navigateur web et tablette.





Montée de niveau vers Symfony 6

Mise en place d'un CRUD



Créer un espace utilisateur (connexion/inscription)

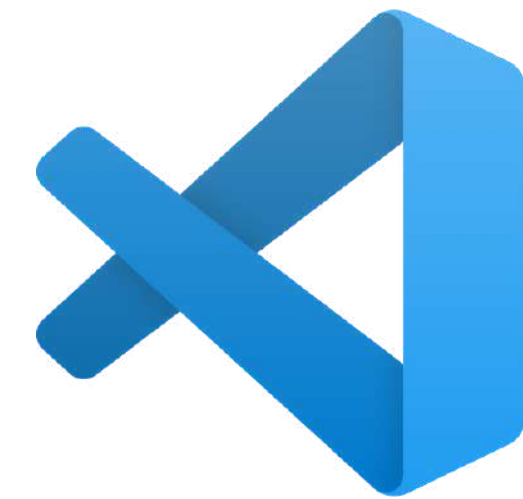
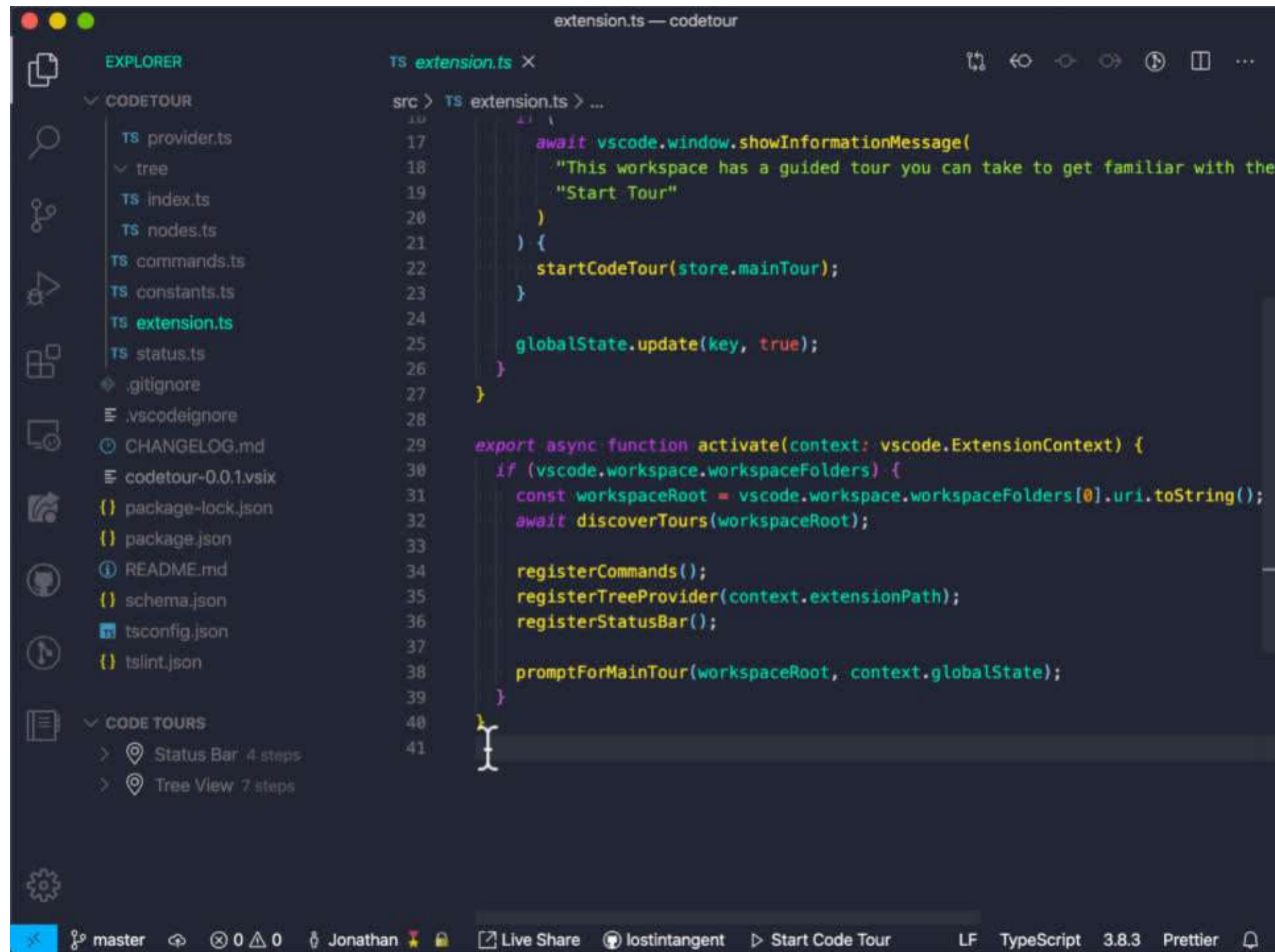


Listing des séances spécifiques à l'utilisateur connecté

Générer un pdf de la séance

Travail demandé





VS Code

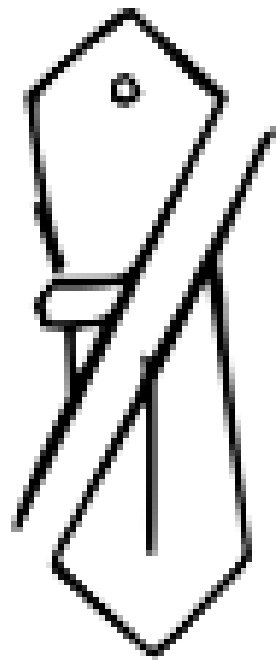
- Gratuit
- Multitudes d'extensions
- Prend en charge divers langages

Éditeur de code



FRONT

Le wireframe



EXCALIDRAW

Balises sémantiques HTML :

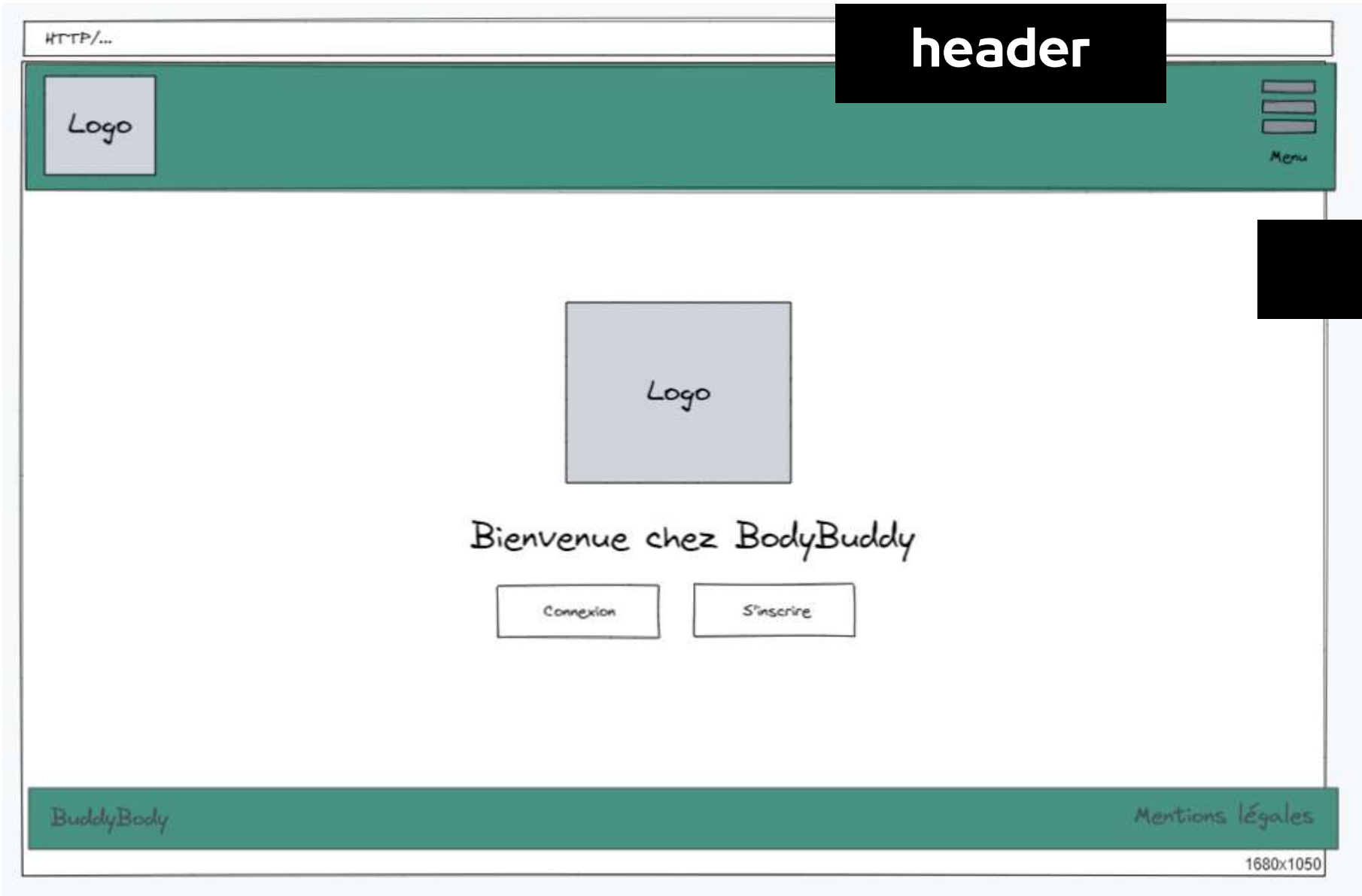
<header>

<main>

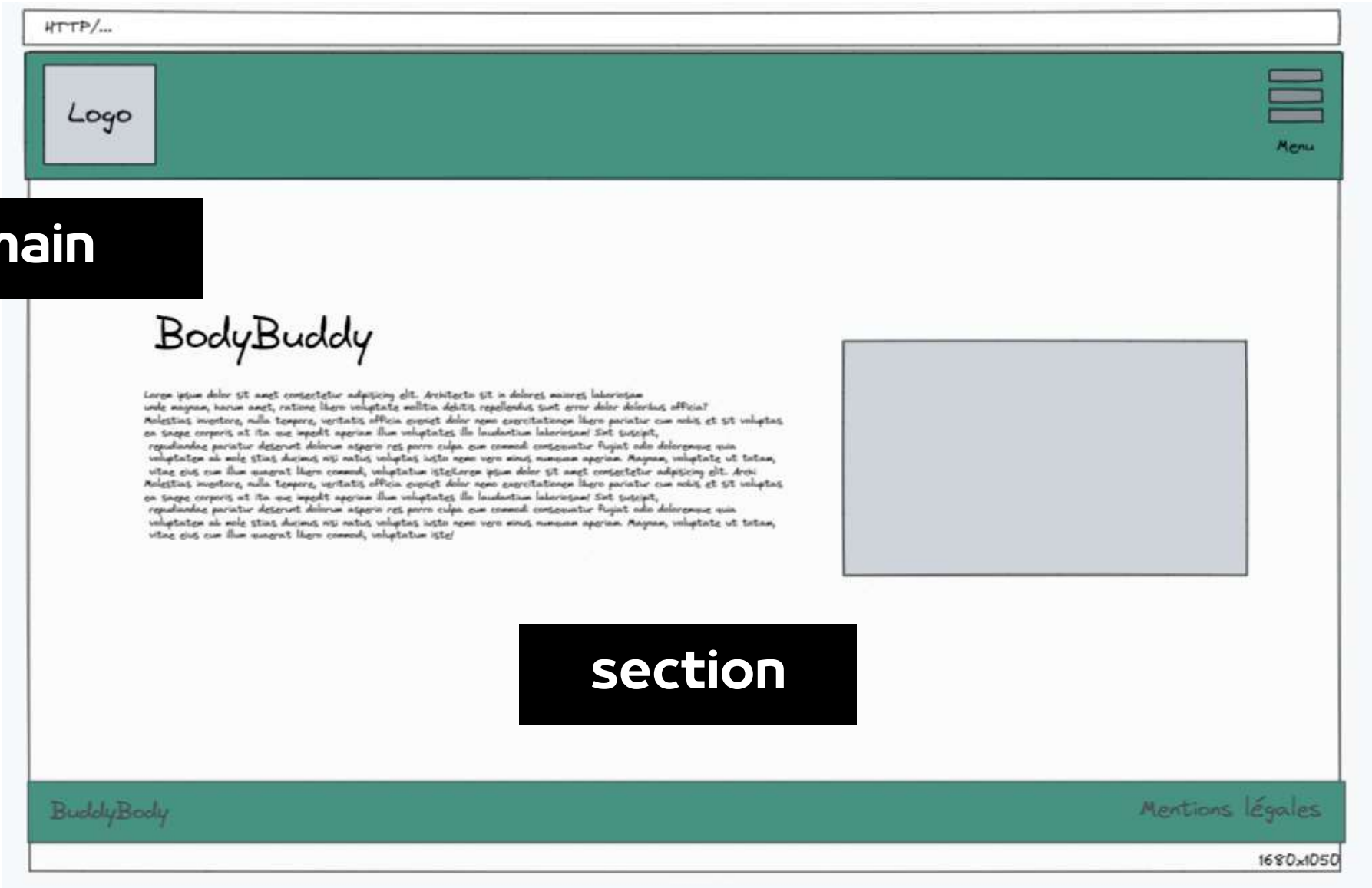
<section>

<footer>

Le wireframe : la page d'accueil

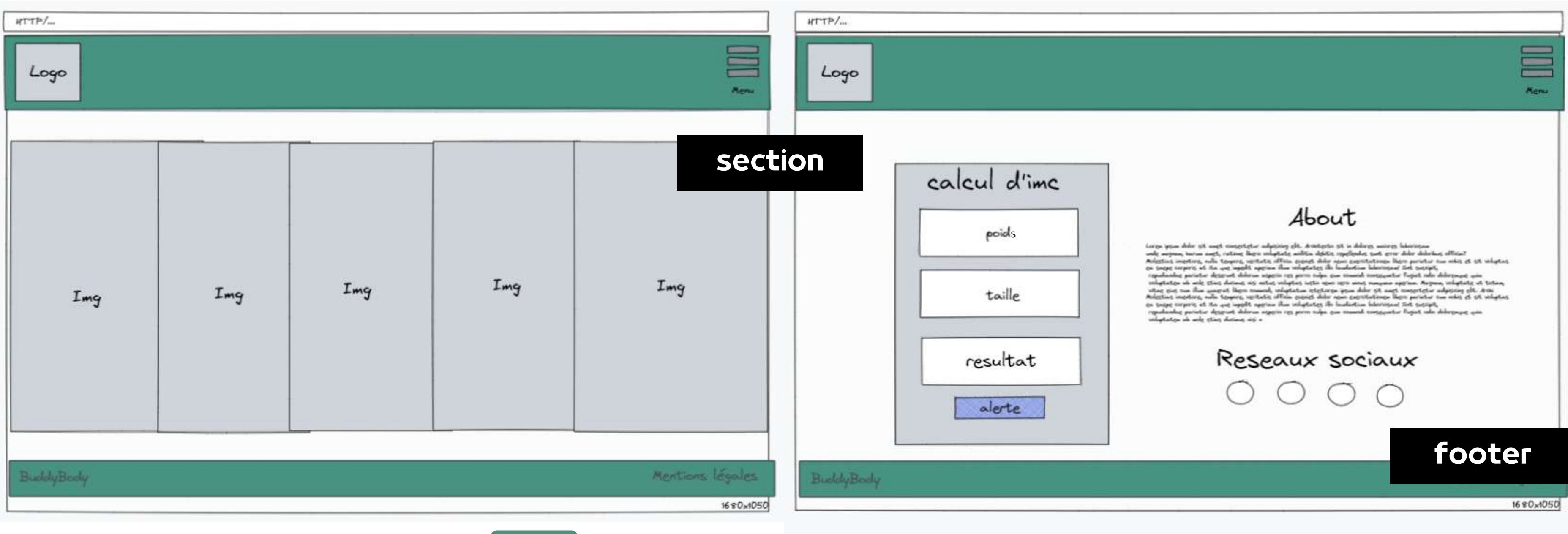


1/4



2/4

Le wireframe : la page d'accueil



3/4

BodyBuddy

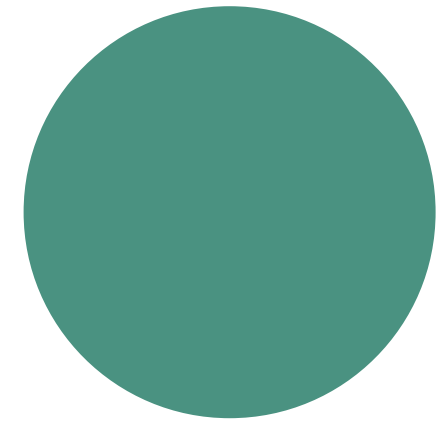
4/4

Charte graphique

Font : 'Questrial'



Couleur principale :



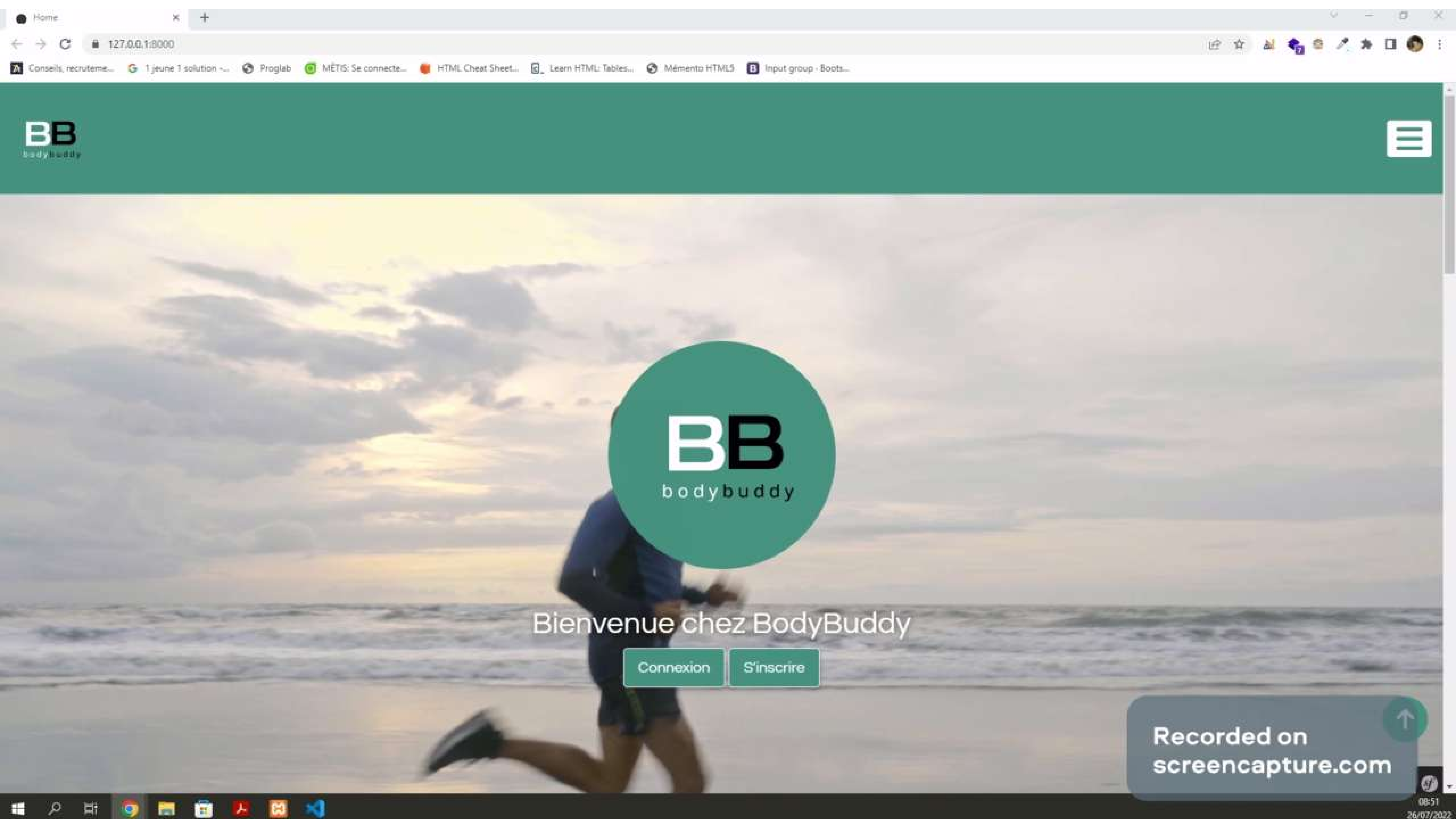
#479281

Logo :



Bouton :





L'intégration via Twig



TWIG

La syntaxe Twig est basée sur ces trois constructions :

`{{ ... }}` : Affiche le contenu d'une variable

`{% ... %}` : Utilisé pour exécuter une logique (condition, boucle...)

`{# ... #}` : Utilisé pour ajouter des commentaires au modèle

base.html.twig

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>
      {% block title %}BodyBuddy
      {% endblock %}
    </title>
```

- Squelette
- meta
- titre
- head
- fiche de styles
- script js

Pour éviter une répétition de code Twig :

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



L'intégration

Création d'un header / footer

```
templates
└─ _partials
    ├── _footer.html.twig
    └── _header.html.twig
```

```
<footer class="text-center text-lg-start">
  <!-- Copyright -->
  <div class="text-center p-2">
    <p class="text-light">© 2022 Copyright : Buddybody</p>
  </div>
  <!-- Copyright -->
</footer>
```

base.html.twig

```
<header>
  {% include "_partials/_header.html.twig" %}
</header>
```

```
{% include "_partials/_footer.html.twig" %}
```

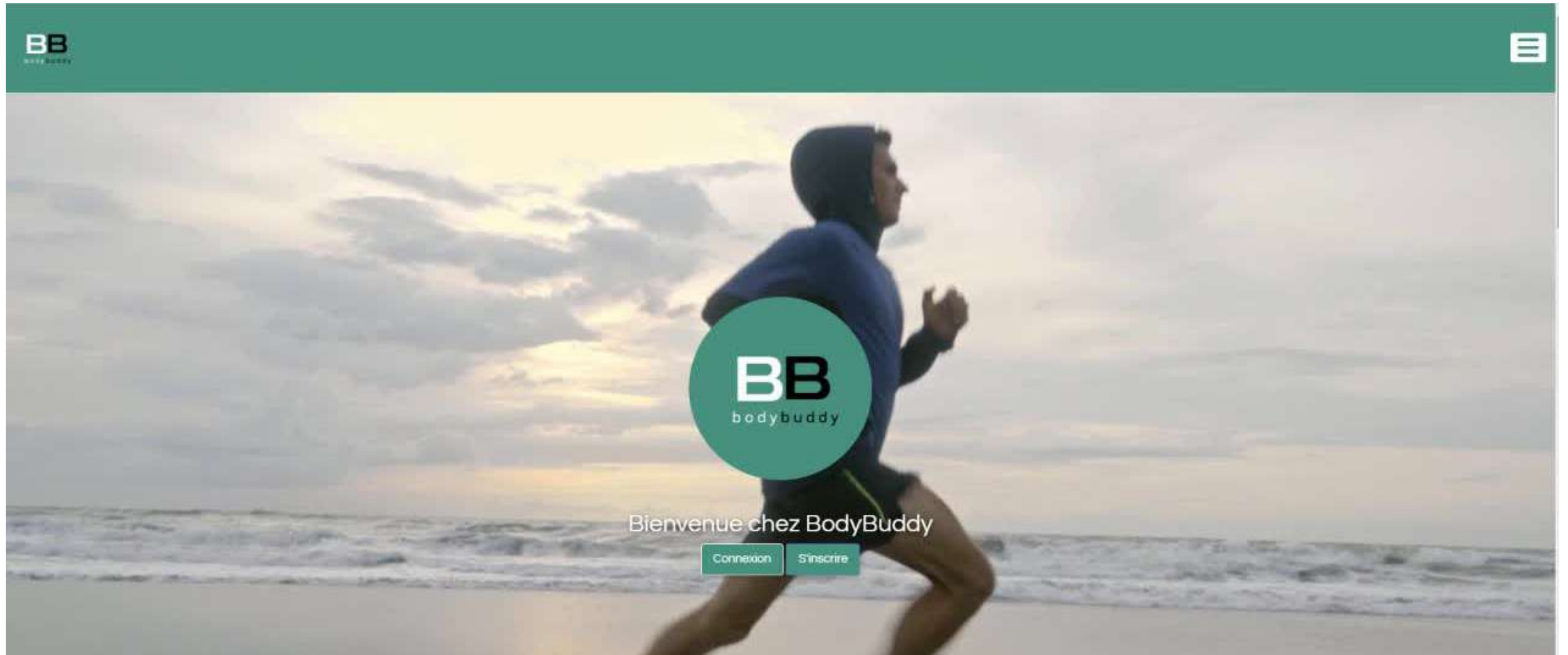
```
<section class="bandeau container-fluid g-0 ">
  <nav class="navbar">
    <div class="container-fluid">
      <div class="logo-header">
        <a href="{{ path('main') }}">
          <i class="fas fa-bars fa-2x"></i>
        </button>
      </div>
    </div>
  </nav>
</section>
```

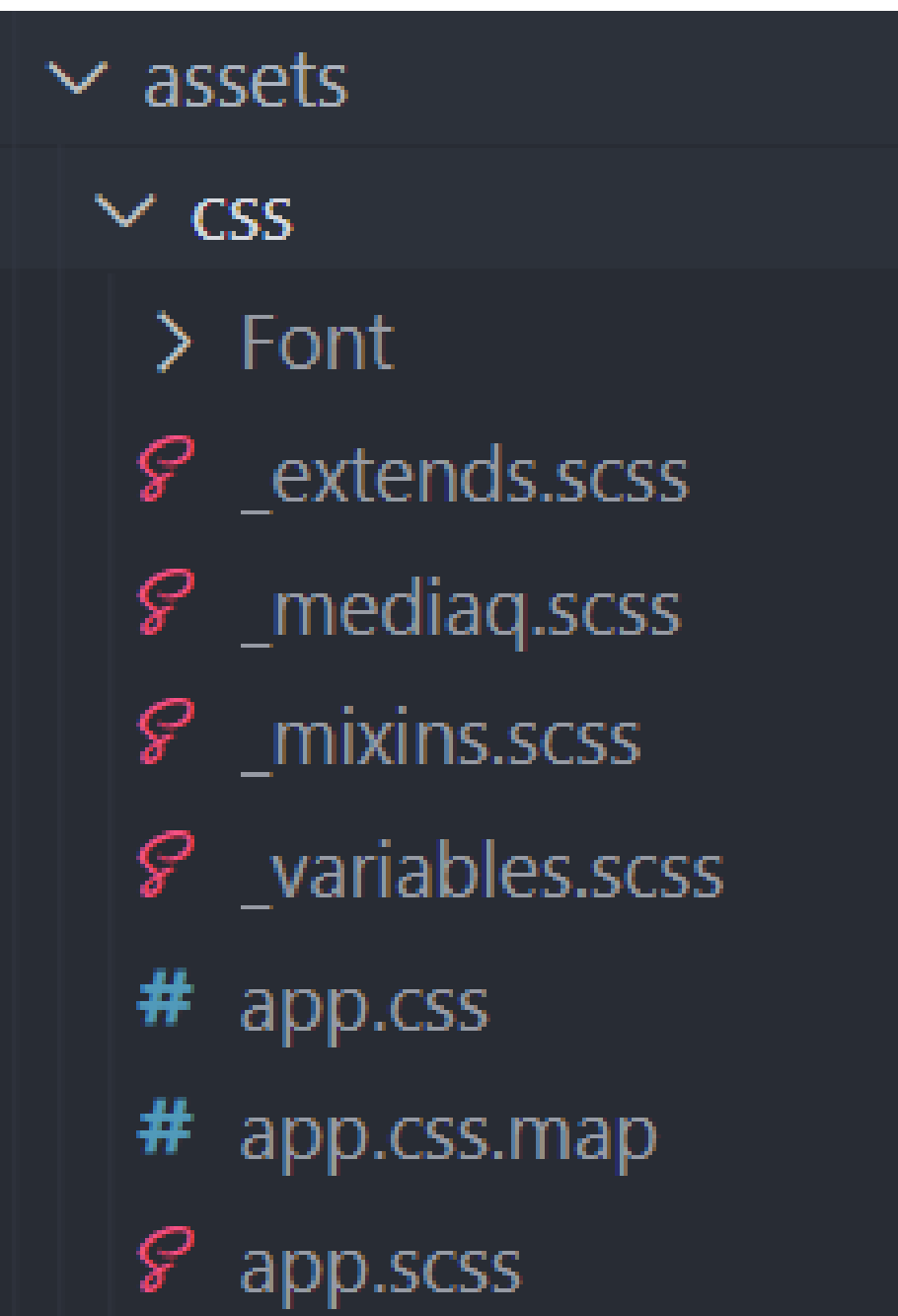


L'intégration d'une modal avec

```
<!-- Button trigger modal -->
{% if is_granted('IS_AUTHENTICATED_REMEMBERED') == false %}
  <button type="button" class="btn btn-primary btn-seance" data-mdb-toggle="modal" data-mdb-target="#connexion">
    Connexion
  </button>
<!-- Modal -->
<div class="modal fade" id="connexion" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog ">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Connexion</h5>
        <button type="button" class="btn-close" data-mdb-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body p-0 g-0 m-0 d-flex justify-content-center align-items-center">
        {% include "login/modal_connexion.html.twig" %}</div>
      <div class="modal-footer"></div>
    </div>
  </div>
</div>
</div>
```


L'intégration d'une modal avec





```
//Variables font
$font-fam-txt: "Questrial";
//Variables main-color
$Main-color: #479281;
//Variables text-color
$text-color-main: black;
$text-color-second: white;
//Variables background-color
$background-color-main: white;
//Variable card-color
$card-color: #d9d9d9;
```

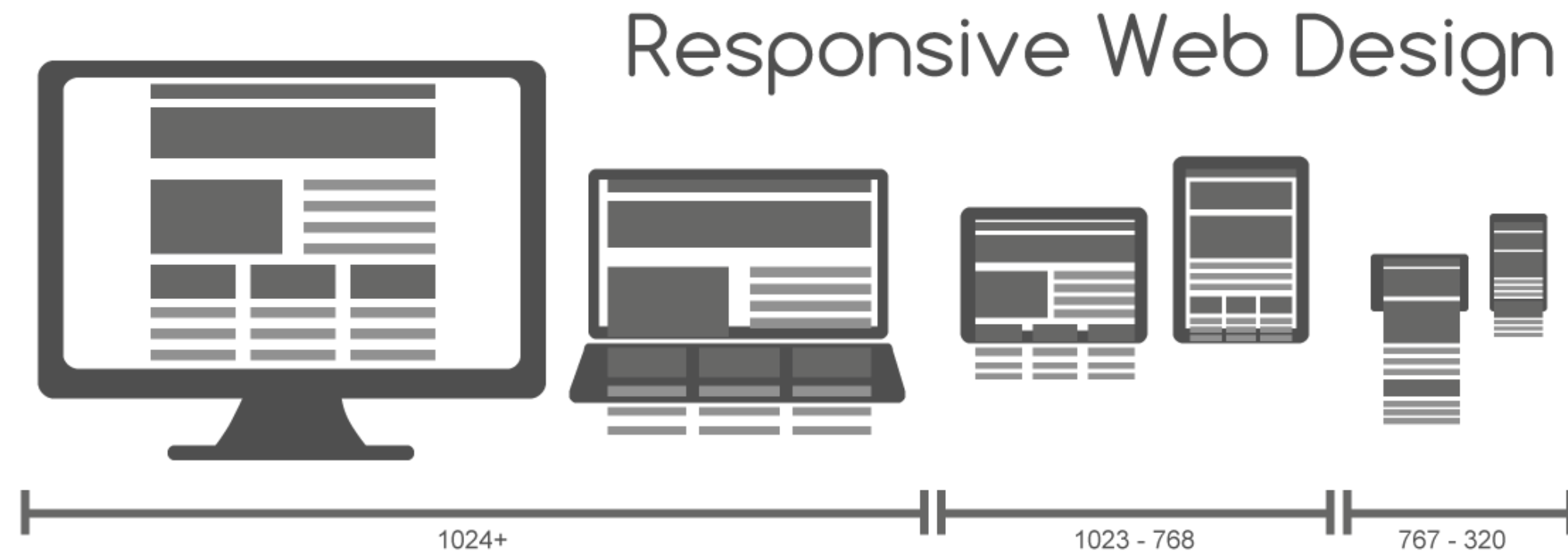
```
// extend flex center
%flexCenter{
  display: flex;
  justify-content: center;
  align-items: center;
}
```

```
//Règles CSS liées à la hauteur et à la largeur
@mixin size ($width, $height) {
  width: $width;
  height: $height;
}
```

```
@media screen and (max-width: 768px) {
  //TITRE PAGE D'ACCUEIL
  .title {
    font-size: 2em;
  }
  //GALERIE D'IMAGE
```

Simplification du CSS avec





- Système de grilles avec Bootstrap
- Media queries

Responsive : format tablette

● Système de grilles avec Bootstrap

Bootstrap prend en charge 4 types de formats:

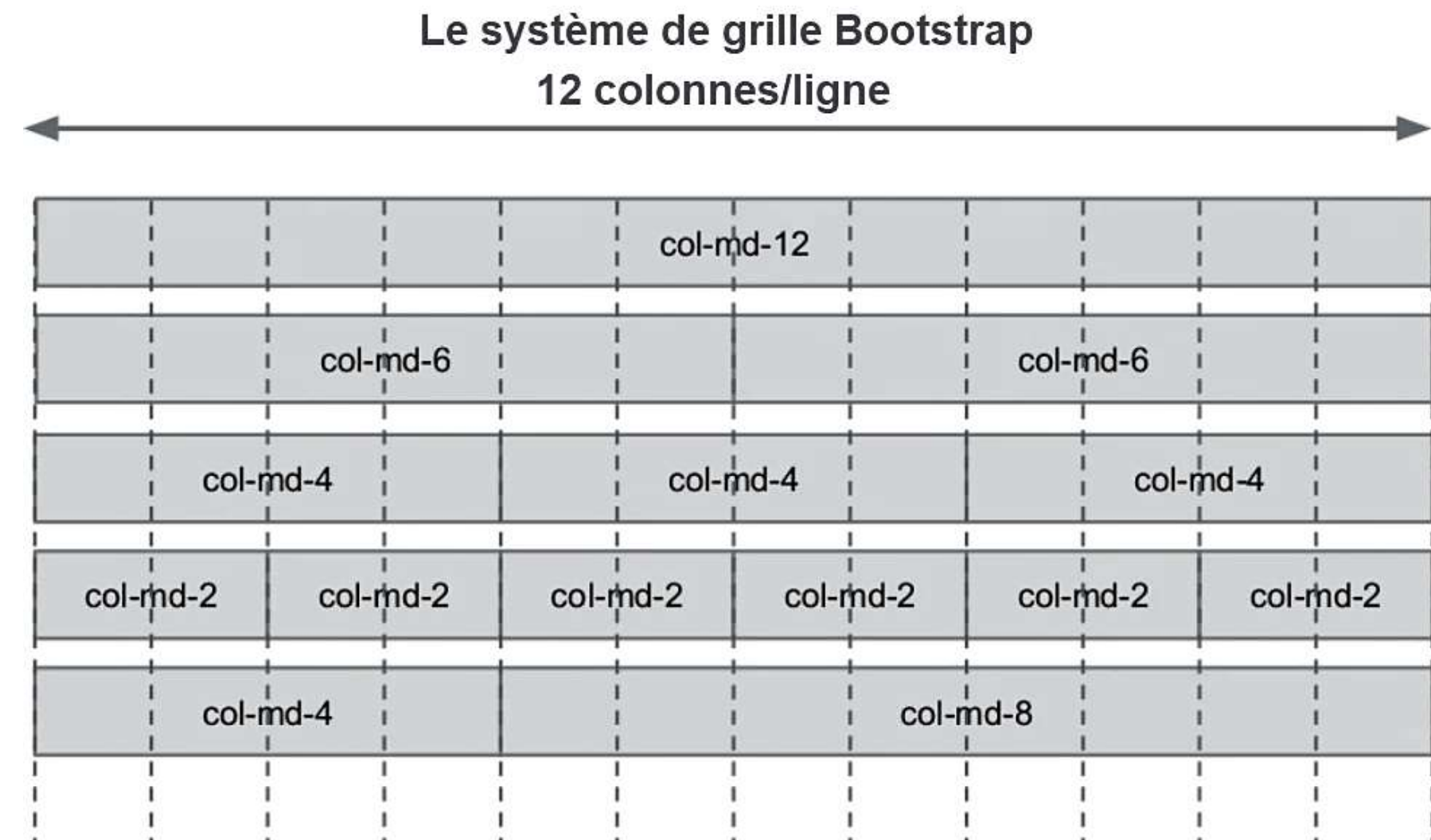
- Très petit format (smartphone)
- Petit format (tablette)
- Moyen format (petit écran)
- Large format (écran standard)

Le principe repose :

conteneur : container-fluid / container

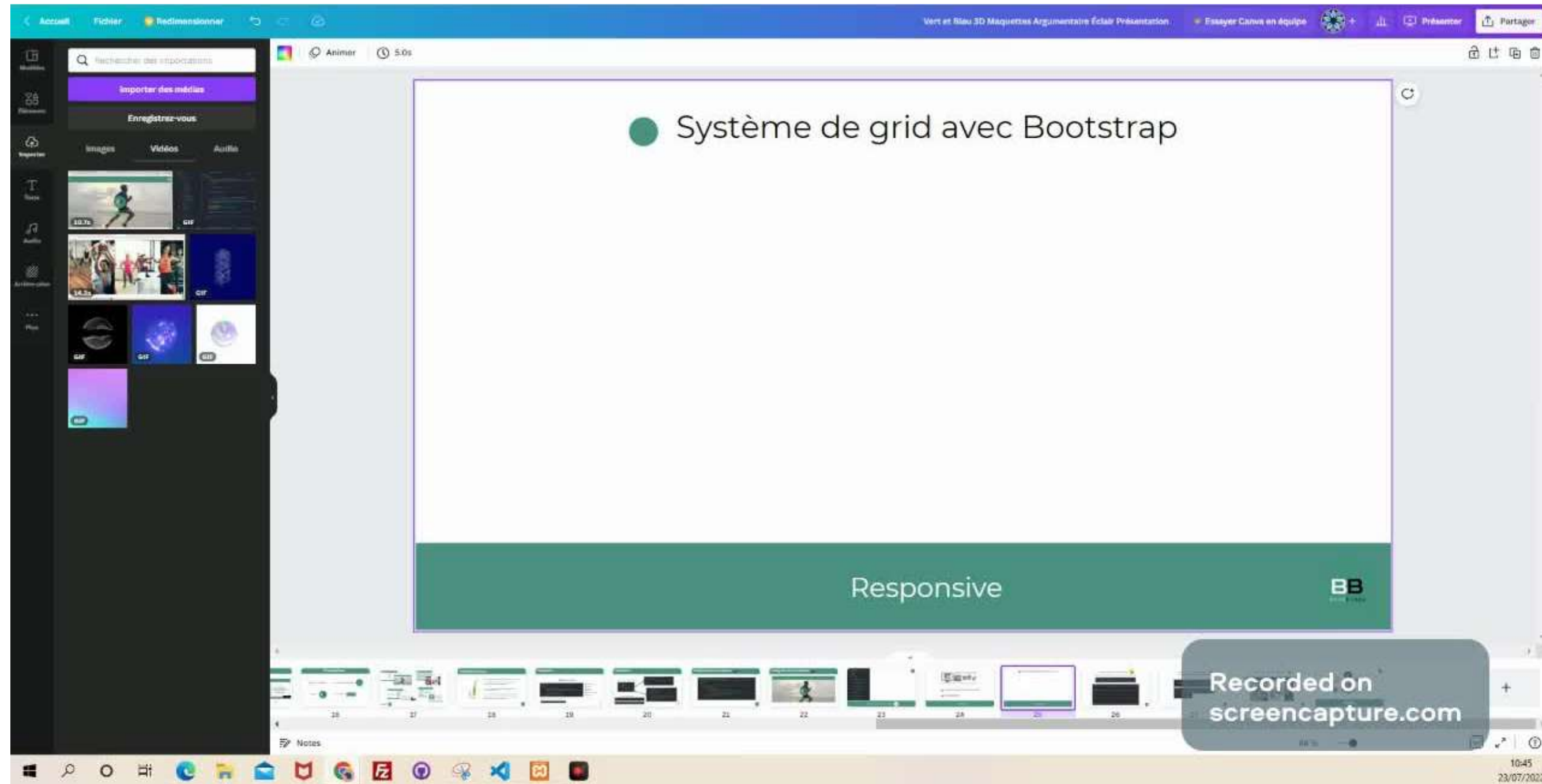
ligne : row

colonne : col



Responsive : format tablette

● Système de grilles avec Bootstrap



Responsive : format tablette

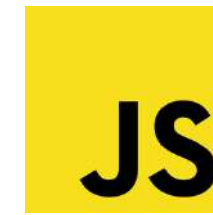
Galerie d'images

JS

Objectif : agrandir la taille de l'image au clic



Galerie d'images

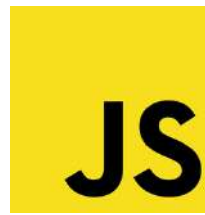


Récupérer toutes mes images dans un tableau à l'aide du `querySelectorAll()`

```
/*Galerie d'images*/  
/* Récupération des images*/  
const panels = document.querySelectorAll(".panel");
```

```
.panel1 {  
  background-image: url(https://storage.googleapis.com/prod-phoenix-bucket/fme/job\_card/10/professeure-sport-64daebcf4.jpg);  
}  
  
.panel2 {  
  background-image: url(https://accrosport.s3.eu-west-3.amazonaws.com/motivation\_sport\_0a615ed6c4.jpeg);  
}  
  
.panel3 {  
  background-image: url(https://ffepgv.fr/build/frontend/images/frontend/shared/hero/6.4daebcf4.jpg);  
}  
  
.panel4 {  
  background-image: url(https://www.astuce-sante.fr/wp-content/uploads/2019/06/sport-en-entreprise-1-1038x576.jpg);  
}  
  
.panel5 {  
  background-image: url(https://datas.masalledesport.com/prod/activity/89/images/1619023375557.jpg);  
}
```


Galerie d'images



Activer / Désactiver la class .open avec la méthode toggle()

```
/*fonction qui active/désactive la class open*/  
function panelOpen() {  
  | this.classList.toggle('open');  
}
```

```
.panel>* {  
  margin: 0;  
  width: 100%;  
  transition: transform 0.5s;  
  flex: 1 0 auto;  
  @extend %flexCenter;  
}
```

```
.panel.open {  
  | flex: 5;
```

Galerie d'images

JS

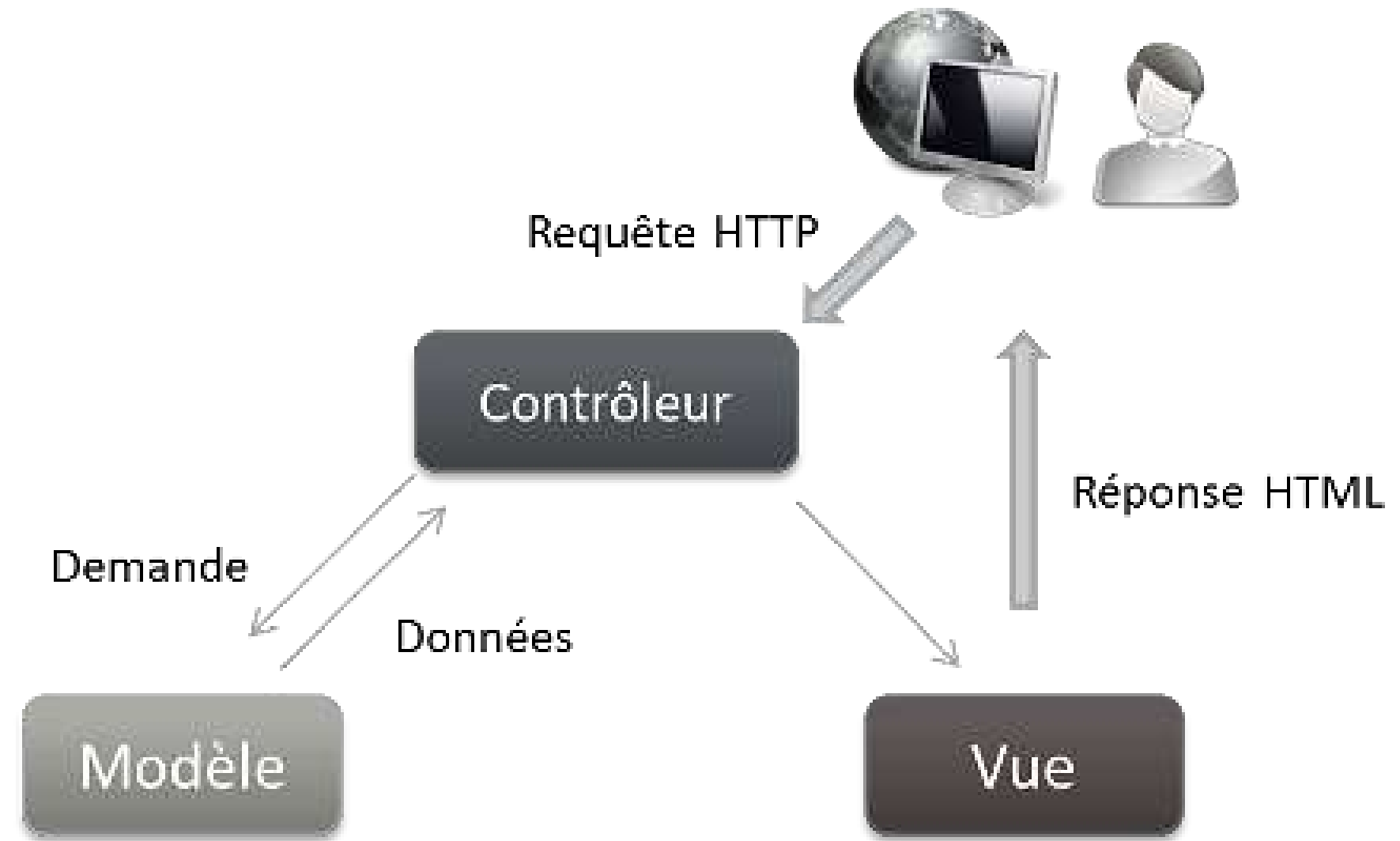
```
/* récupération d'événement au clic */  
panels.forEach(panel => panel.addEventListener('click', panelOpen));
```





BACK

framework PHP



Symfony : L'architecture MVC

Les prérequis


Avant le processus d'installation :

- PHP 8.0.2 ou une version plus récente
- Composer de préférence la version récente
- Symfony CLI



L'arborescence



- ▶ bin
- ▶ config
- ▶ public
- ▶  src
- ▶ templates
- ▶ var
- ▶ vendor
- .env
- .env.dist
- .gitignore
- composer.json
- composer.lock
- symfony.lock

```
src

- > Controller
- > DataFixtures
- > Entity
- > Form
- > Repository

```

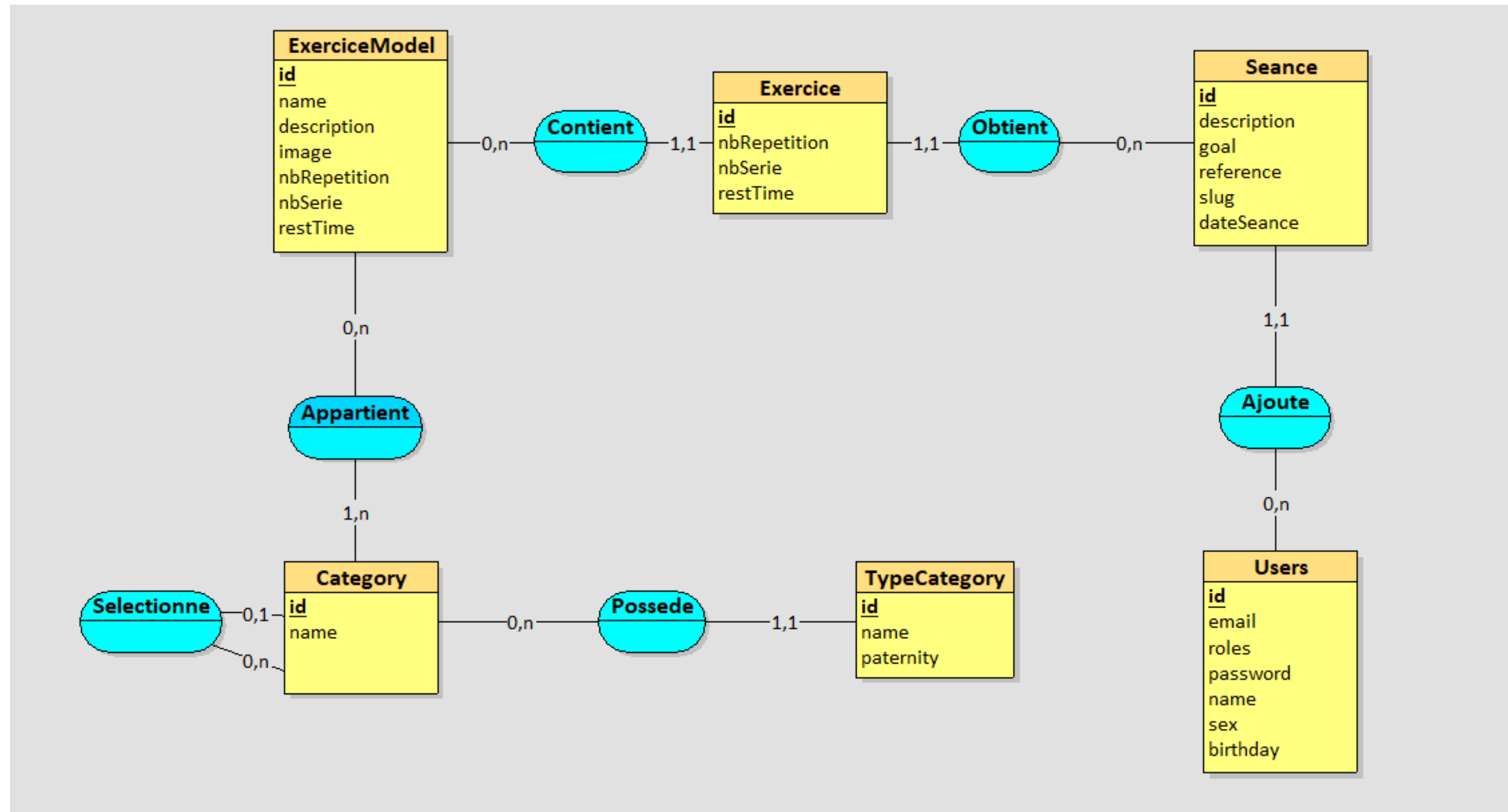
Controller : accessibles par des routes

Entity : l'ensemble de nos tables de la base de de données

Repository : permet d'ajouter les requêtes

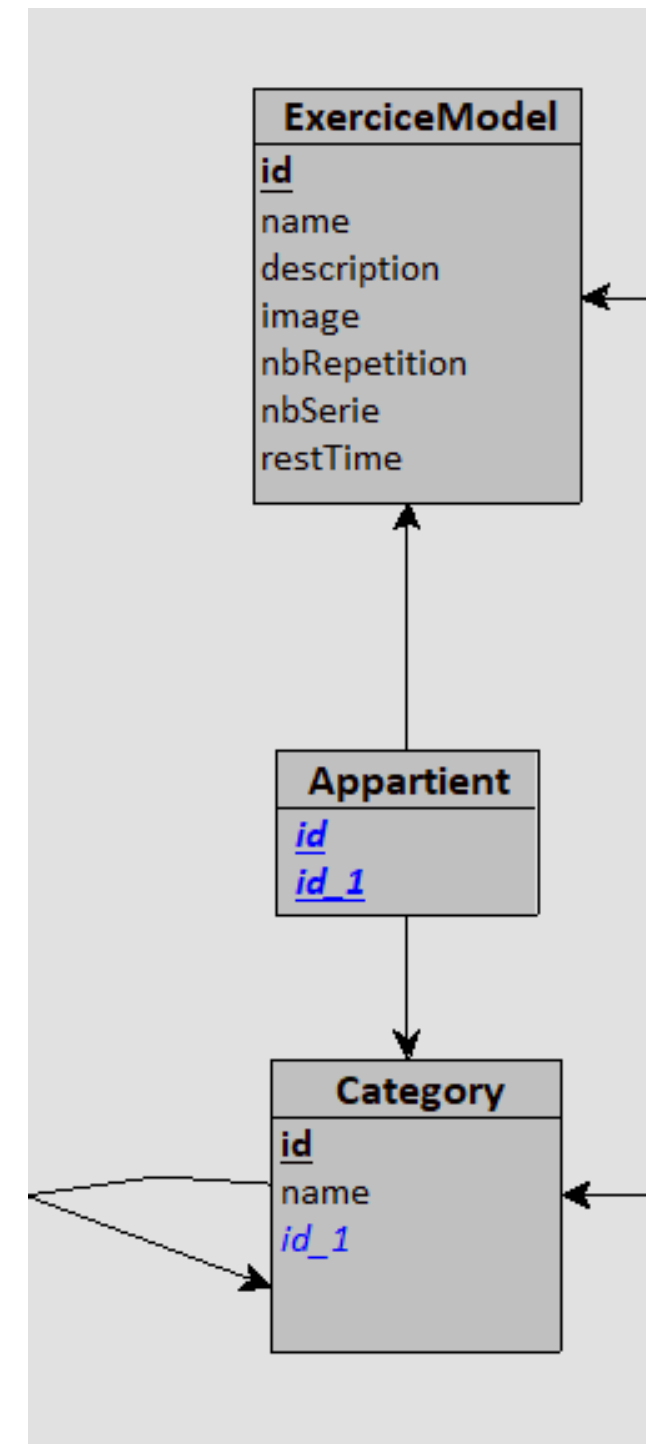
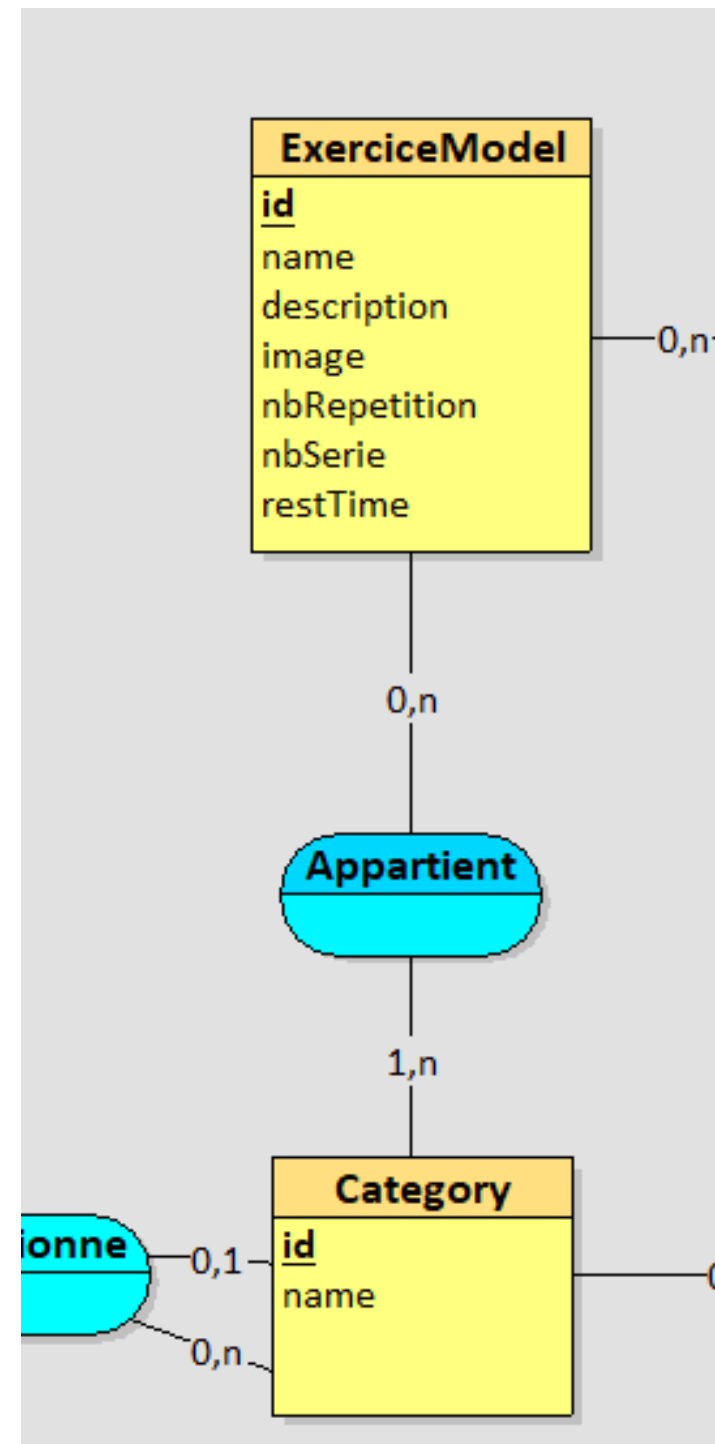
L'outil de modélisation : Looping

le Modèle conceptuel des données (ou MCD)



Le MCD repose sur les notions d'entité et d'association et sur les notions de relations.

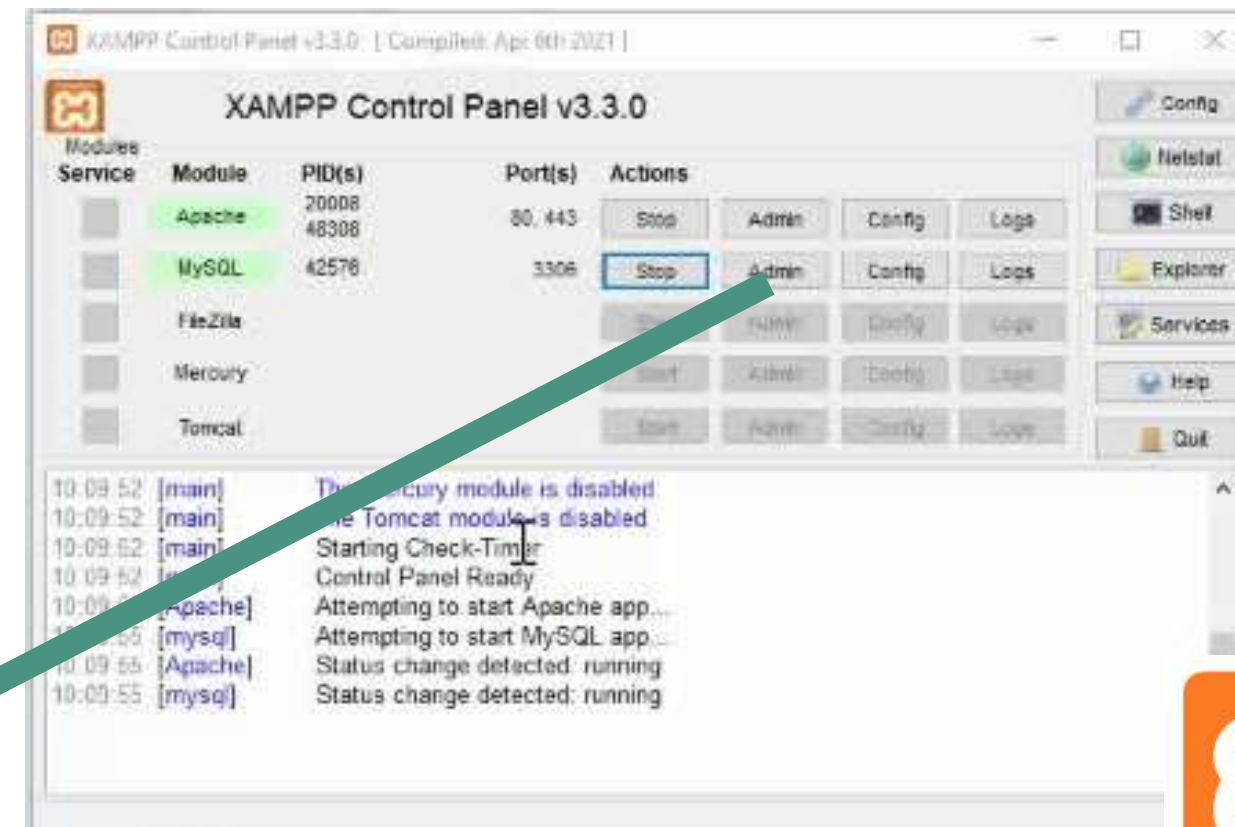
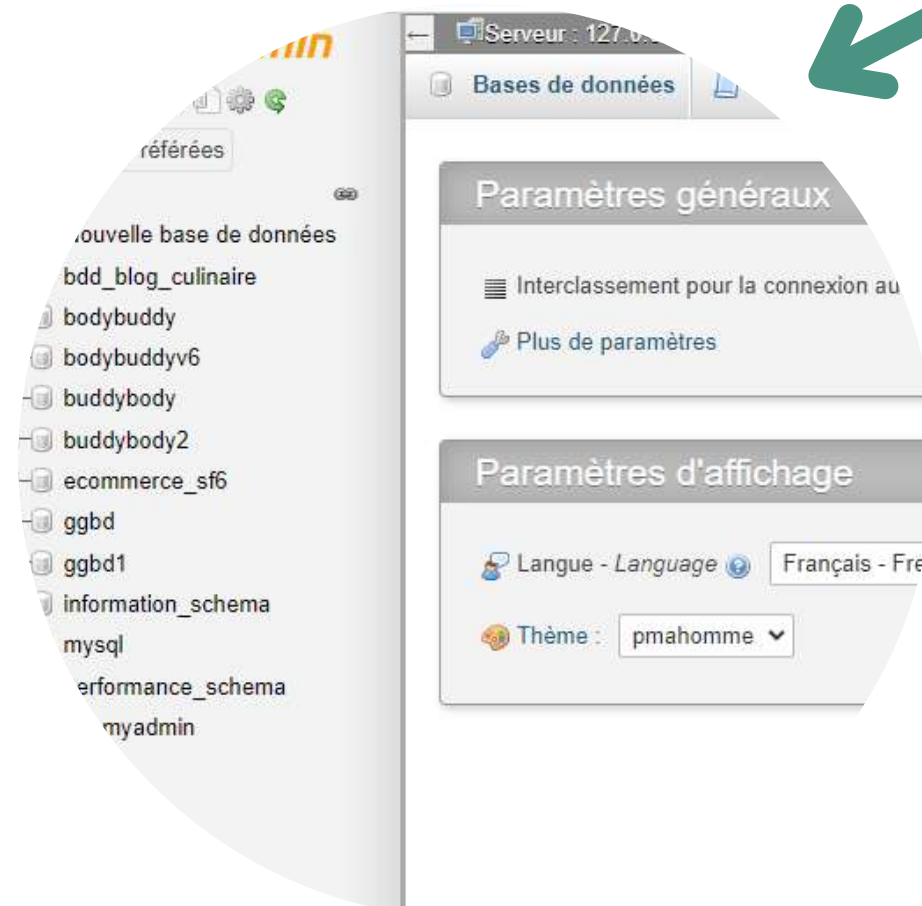
L'outil de modélisation : Looping



Modèle logique de données (MLD)

Choix du système de gestion de bases de données (SGBD)

PhpMyAdmin 



```
>_ symfony console doctrine:database:create  
symfony console make:migration  
symfony console doctrine:migrations:migrate  
symfony console doctrine:fixtures:load
```

Système de gestion de données : Doctrine

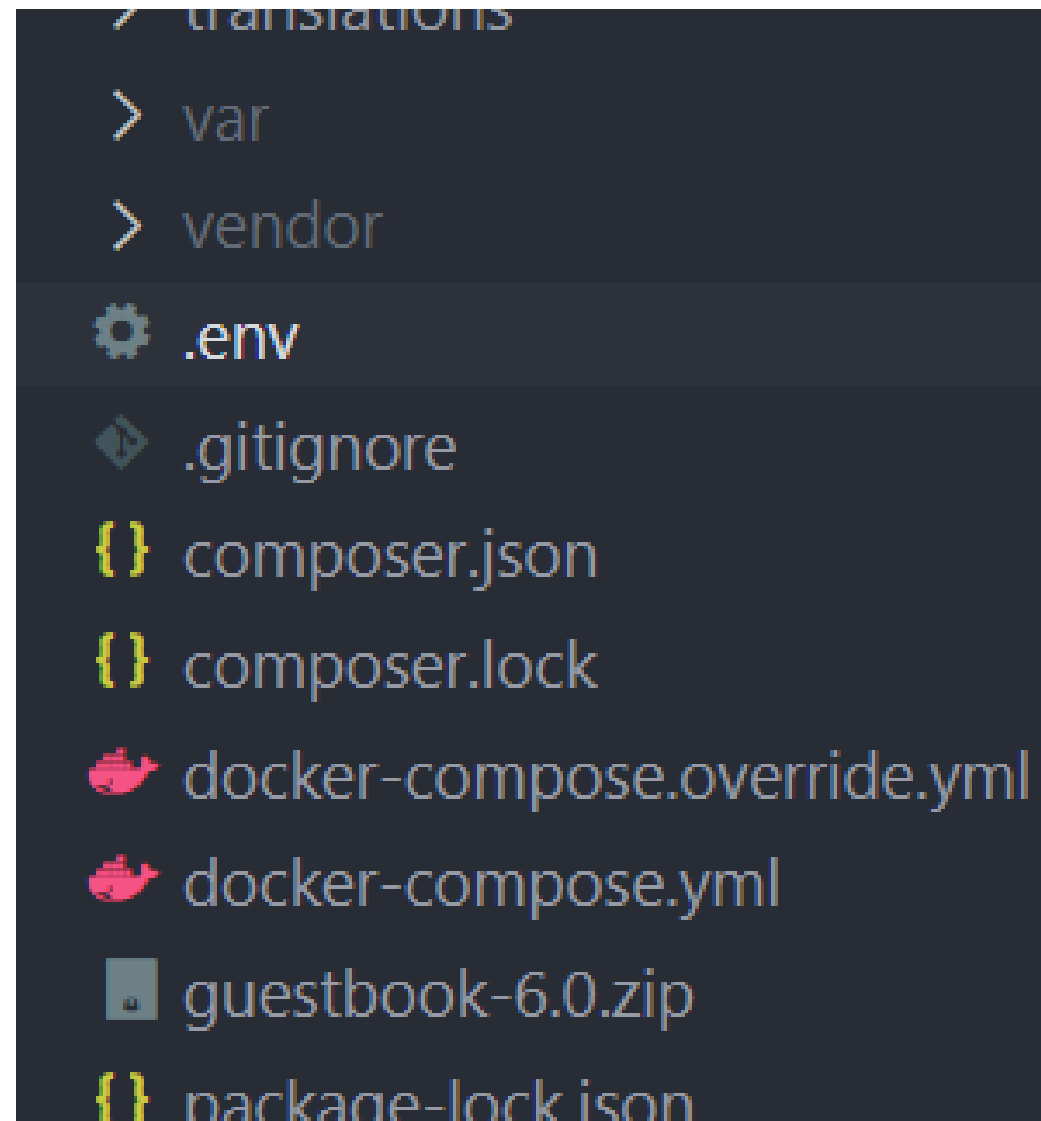


Créer et mettre à jour une base de données

- Data Mapper
- EntityManager

Doctrine nous permet d'interagir avec la base de données.

Les variables d'environnements



```
#  
# DATABASE_URL="sqlite:///kernel.project_dir%/var/data.db"  
  
# DATABASE_URL="mysql://root@127.0.0.1/buddybody?serverVersion=10.4.24-MariaDB"  
  
DATABASE_URL="mysql://root@127.0.0.1/bodybuddyv6?serverVersion=10.4.24-MariaDB"  
  
# DATABASE_URL="postgresql://symfony:ChangeMe@127.0.0.1:5432/app?serverVersion=13&ch  
###< doctrine/doctrine-bundle ###
```



nom de la base de données

Modifier le .env.local



Créer une entité User

>_

symfony console make:user

```
GOT — yemiwebby@Olusosis-MBP-2 — ..ial/auth0/GOT — -zsh — 149x35

→ GOT php bin/console make:user

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

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

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

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

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

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

Success!

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

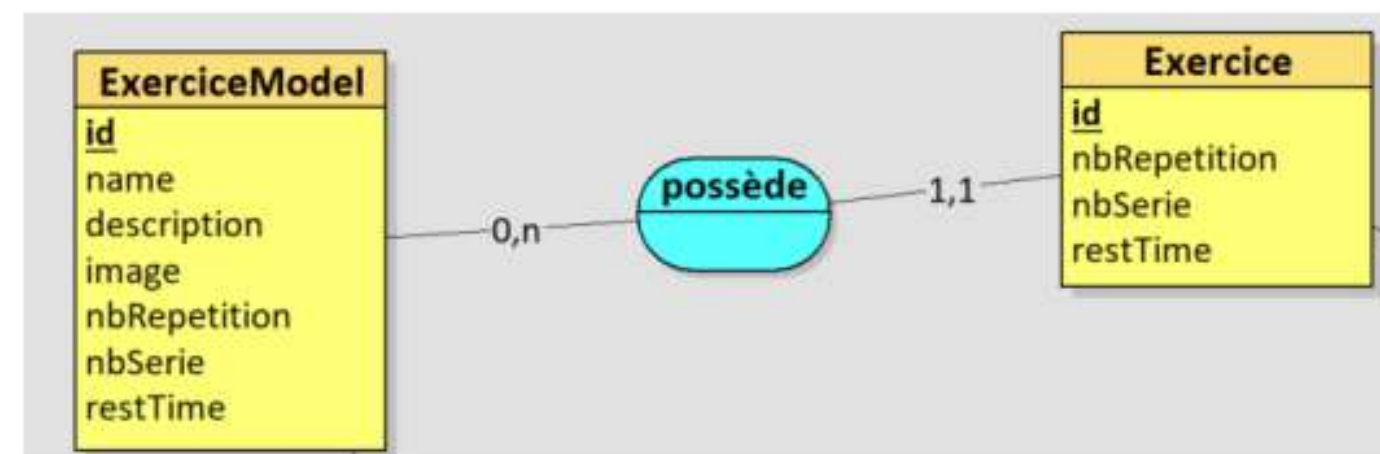



Les relations avec Symfony

What type of relationship is this?

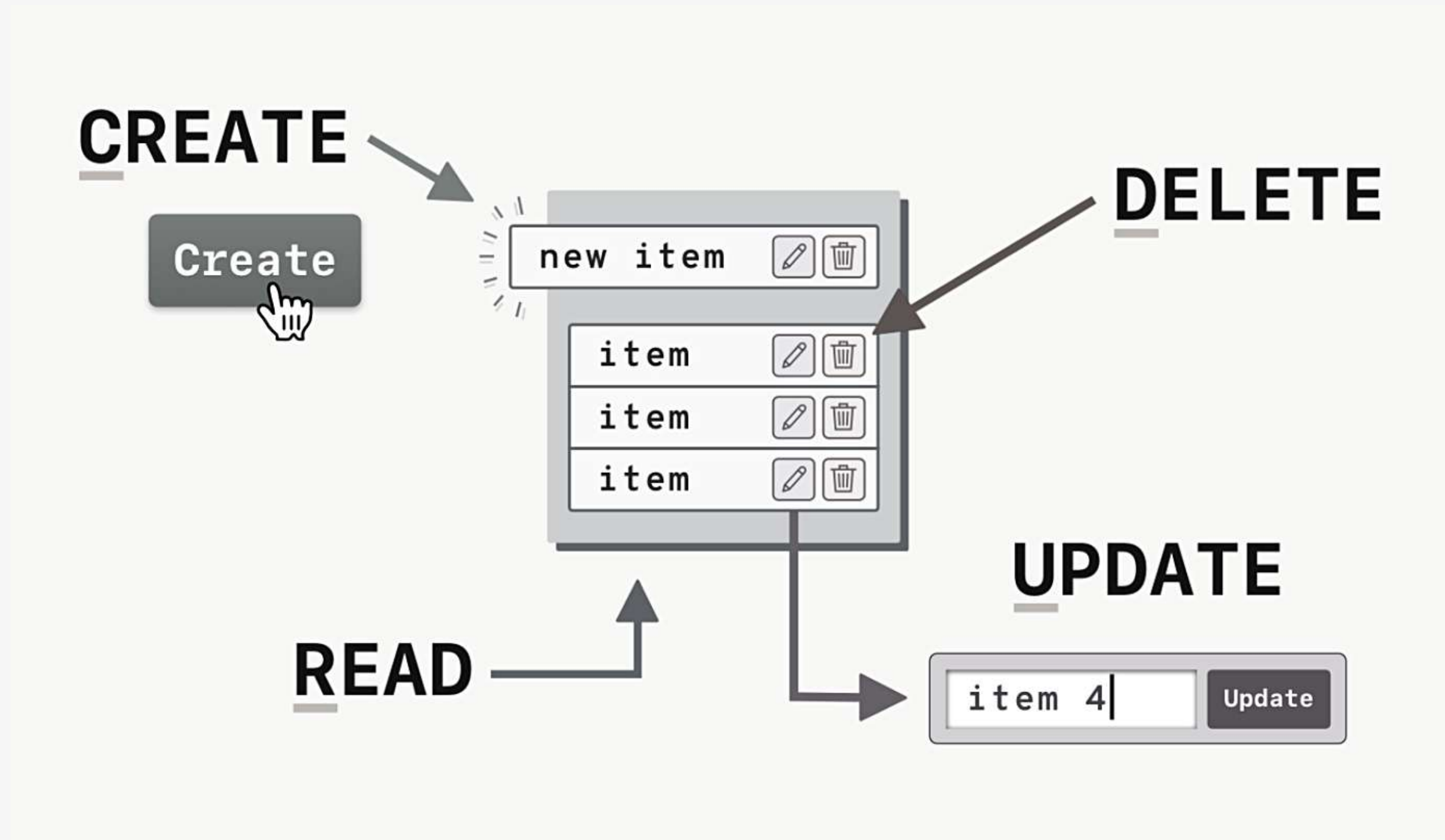
Type	Description
ManyToOne	Each Commentaires relates to (has) one Articles . Each Articles can relate to (can have) many Commentaires objects
OneToMany	Each Commentaires can relate to (can have) many Articles objects. Each Articles relates to (has) one Commentaires
ManyToMany	Each Commentaires can relate to (can have) many Articles objects. Each Articles can also relate to (can also have) many Commentaires objects
OneToOne	Each Commentaires relates to (has) exactly one Articles . Each Articles also relates to (has) exactly one Commentaires .

ManyToOne



Un **ExerciceModel** possède plusieurs **Exercices**, or
un **Exercice** n'appartient qu'a un **ExerciceModel**

Implémentation du CRUD



SeanceController : CREATE / UPDATE

>- symfony console make:controller: SeanceController

```
/* Ajout et Modification de séance */
#[Route('/{seance}', name: 'create_update', requirements: ['seance' => '\d+'])]
#[Route('/{seance}/{etape}', name: 'etape', requirements: ['seance' => '\d+'])]
public function create_update(EntityManagerInterface $entityManager, Request $request, SeanceRepository $seanceRepository,
TypeCategoryRepository $typeCategorieRepository, CategoryRepository $categorieRepository
, EntityManagerInterface $em , $seance = null, $etape = "echauffement"): Response
{
    $user = $this->getUser();
    //Création ou Récupération de la Séance
    if ($seance === null) {
        $seance = new Seance();
        $seance->setDescription("");
        $seance->setGoal("");
        $seance->setReference("");
        $seance-> setDate(new \DateTime());
        $seance->setSlug("");
        $seance->setUser($this->getUser());
        $em->persist($seance);
        $em->flush();
    }
    else {
        $seance = $seanceRepository->find($seance);
    }
}
```


SeanceController : CREATE / UPDATE

```
/* --Création du formulaire-- */
$form = $this->createForm(SeanceType::class, $seance);
$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {
    // encode the plain password
    $entityManager->persist($seance);
    $seance->setUser($this->getUser());
    $entityManager->flush();
    $seance = $form->getData();

    // do anything else you need here, like send an email
    return $this->redirectToRoute('seance_index');
}

$typeCategorieEtape = $typeCategorieRepository->findOneBy(array("name" => "Etapas"));
$categoriesEtapas = $categorieRepository->findBy(array("typeCategory" => $typeCategorieEtape));

return $this->render('seance/create_update.html.twig', [
    'seanceForm' => $form->createView(),
    'seance' => $seance,
    'etape' => $etape,
    'categoriesEtapas' => $categoriesEtapas,
    'data' => $data,
    'user' => $user,
]);
```

SeanceController : READ

```
#[Route('/seance', name: 'seance_')]
class SeanceController extends AbstractController
{
    public function __construct(private ManagerRegistry $doctrine) {}

    /* Liste des Séances */
    #[Route('s', name: 'index')]
    public function index(SeanceRepository $seanceRepository): Response
    {
        $user = $this->getUser();

        return $this->render('seance/index.html.twig', [
            'seances' => $seanceRepository->findAll(),
            'user' => $user,
        ]);
    }
}
```

Bonjour Lala


Description	Objectif	Reference	actions
Pas de séance			
Nouvelle Séance			

© 2022 Copyright : Buddybody

SeanceController : DELETE

```
/* Suppression d'une séance */
#[Route('/delete/{seance}', name: 'delete', requirements: ['seance' => '\d+'])]
public function delete(SeanceRepository $seanceRepository, EntityManagerInterface $em , $seance): Response
{
    $seance = $seanceRepository->find($seance);
    $em->remove($seance);
    $em->flush();
    return $this->redirectToRoute('seance_index');
}
```

 Buddybody



Bonjour **Lala**

Description	Objectif	Reference	actions
Ma 1ère séance	Perdre 15 kilos	Perte de dos	Dos
<div>ModifierDupliquerSupprimer</div>			

Nouvelle Séance

© 2022 Copyright | Buddybody

Vue

```
{% for seance in seances %}
  {% if seance.user.id == user.id %}
    <tr>
      <td>{{ seance.description }}</td>
      <td>{{ seance.goal }}</td>
      <td>{{ seance.reference }}</td>
      <td>{{ seance.slug }}</td>
      <td>
        <a class="btn-seance" href="{{ path('seance_create_update', {'seance': seance.id}) }}">Modifier</a>
        <a class="btn-seance" href="{{ path('seance_duplicate', {'seance': seance.id}) }}">Dupliquer</a>
        <a class="btn-seance" href="{{ path('seance_delete', {'seance': seance.id}) }}">Supprimer</a>
      </td>
    </tr>
  {% endif %}
{% else %}
  <tr>
    <td colspan="5">Pas de séance</td>
  </tr>
{% endfor %}
```

Bonjour **Lala**

Description	Objectif	Reference	actions
Ma 1ère séance	Perdre 15 kilos	Perte de dos	Dos
Ma 1ère séance	Perdre 15 kilos	Perte de dos	Dos
Ma 1ère séance	Perdre 15 kilos	Perte de dos	Dos
Ma 1ère séance	Perdre 15 kilos	Perte de dos	Dos

Nouvelle Séance



Formulaire d'inscription

```
>_
```

```
symfony console make:registration-form
```

Les fichiers suivants sont alors créés :

- src/Controller/RegistrationController.php
- src/Form/RegistrationFormType.php
- templates/registration/register.html.twig



Création du formulaire d'inscription

src/Form/RegistrationFormType.php

```
use App\Entity\User;
use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\Extension\Core\Type\BirthdayType;
use Symfony\Component\Form\Extension\Core\Type\EmailType;
use Symfony\Component\Form\Extension\Core\Type\TextType;
use Symfony\Component\Form\Extension\Core\Type\CheckboxType;
use Symfony\Component\Form\Extension\Core\Type\ChoiceType;
use Symfony\Component\Form\Extension\Core\Type>PasswordType;
use Symfony\Component\Form\Extension\Core\Type\RadioType;
use Symfony\Component\Form\FormBuilderInterface;
use Symfony\Component\OptionsResolver\OptionsResolver;
use Symfony\Component\Validator\Constraints\IsTrue;
use Symfony\Component\Validator\Constraints\Length;
use Symfony\Component\Validator\Constraints\NotBlank;

class RegistrationFormType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options): void
    {
        $builder
            ->add('email', EmailType::class, array(
                'attr' => array(
                    'label' => false,
                    'placeholder' => 'Email'
                )
            ))
    }
}
```

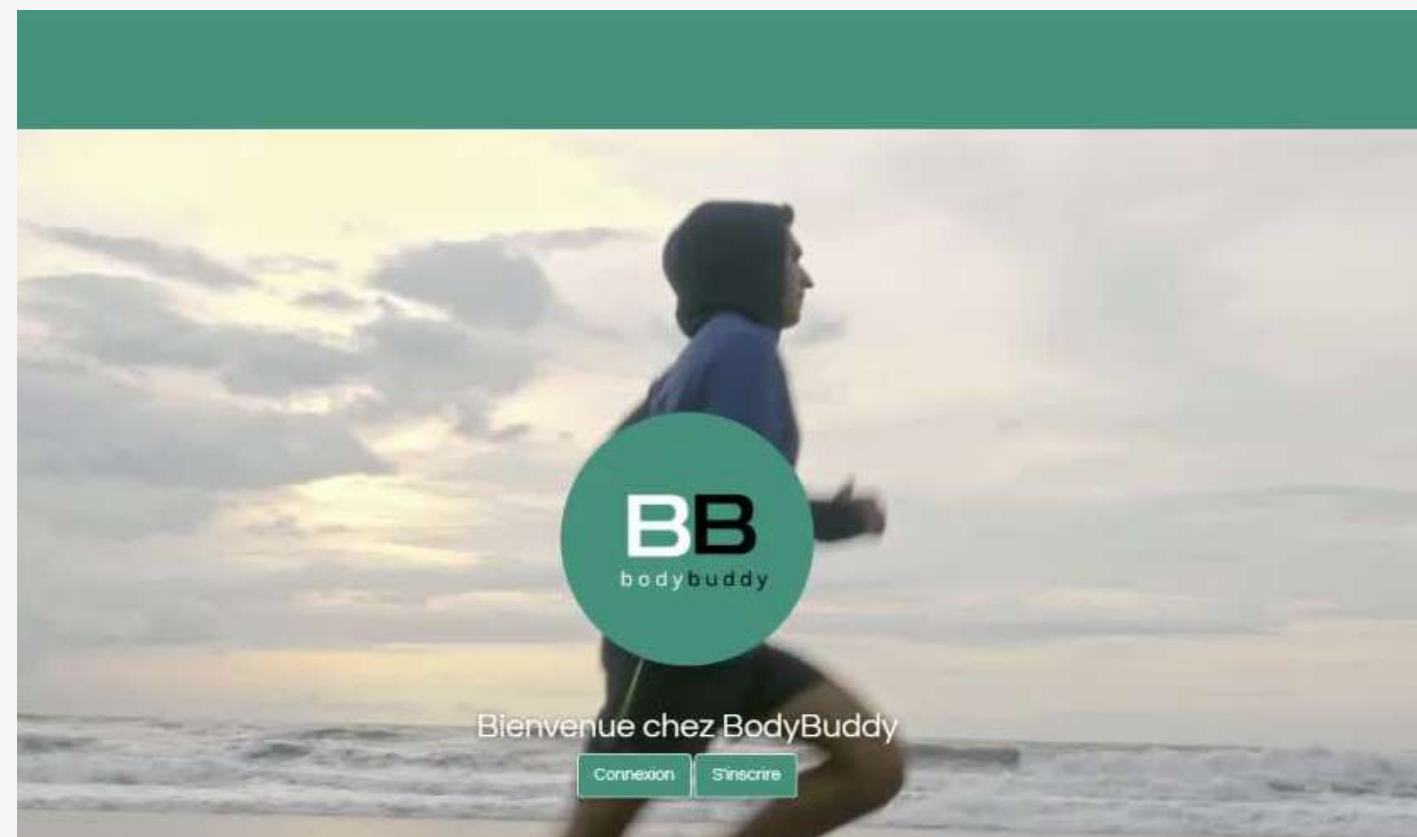
FormBuilderInterface



Intégration du formulaire d'inscription

templates/registration/register.html.twig

```
{{ form_start(registrationForm) }}  
  <h1>Inscription</h1>  
  {{ form_row(registrationForm.name, {label: ' '}) }}  
  {{ form_row(registrationForm.birthday, {label: 'Date de Naissance'}) }}  
  {{ form_row(registrationForm.sex, {label: 'Sexe'}) }}  
  {{ form_row(registrationForm.email, {label: ' '}) }}  
  {{ form_row(registrationForm.plainPassword, {label: ' '}) }}  
  {{ form_row(registrationForm.agreeTerms, {label: 'Acceptez les conditions'}) }}  
  
  <button type="submit" class="btn btn-seance">S'inscrire</button>  
{{ form_end(registrationForm) }}
```



KnpSnappyBundle

Rendre un document pdf comme réponse d'un contrôleur



```
#[Route('/fiche/pdf/{seance}', name: 'fiche_pdf')]
public function fichePdf(SeanceRepository $seanceRepository, Pdf $knpSnappyPdf, $seance): Response
{
    $seance = $seanceRepository->find($seance);
    $html = $this->renderView('seance/pdf.html.twig', array(
        'seance' => $seance
    ));


    return new PdfResponse(
        $knpSnappyPdf->getOutputFromHtml($html,array('orientation'=>'Landscape',
        'default-header'=>false)),
        'file.pdf'
    );
}
```



KnpSnappyBundle

Rendre un document pdf comme réponse d'un contrôleur





[Générer le PDF](#)[Retour](#)

Description

Ma 1ère séance

Objectif

Perdre 15 kilos

Modifier

Exercice	Schéma	Description	Difficulté	Séries - Répétitions - Repos
----------	--------	-------------	------------	------------------------------

SecurityBundle - le fichier de configuration

config > security.yaml

Providers : une "base" de tous les utilisateurs/mot de passe

```
providers:
    # used to reload user from session
    app_user_provider:
        entity:
            class: App\Entity\User
            property: email
```

Contrôle d'accès : contrôle les permissions (URL)

```
access_control:
    # - { path: ^/admin, roles: ROLE_ADMIN }
    - { path: ^/seances, roles: ROLE_USER }
    - { path: ^/seance, roles: ROLE_USER }
```



Pare-feu : vérifie l'authenticité d'un utilisateur

```
main:
    lazy: true
    provider: app_user_provider
    form_login:
        # "login" is the name of the form
        login_path: login
        check_path: login
    logout:
        path: logout
        target: main
```

SecurityBundle - le fichier de configuration

config > security.yaml

```
#[Route('/login', name: 'login')]
public function login(AuthenticationUtils $authenticationUtils): Response
{
    // get the login error if there is one
    $error = $authenticationUtils->getLastAuthenticationError();

    // last username entered by the user
    $lastUsername = $authenticationUtils->getLastUsername();

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

```
#[Route('/logout', name: 'logout', methods: ['GET'])]
public function logout()
{
    // controller can be blank: it will never be called!
    throw new \Exception('Don\'t forget to activate logout in security.yaml');
}
```

Checking to see if a User is Logged In (IS_AUTHENTICATED_FULLY)

Vérification de la connexion d'un utilisateur (IS_AUTHENTICATED_FULLY)

If you *only* want to check if a user is logged in (you don't care about roles), you have the following two options.

Firstly, if you've given *every* user `ROLE_USER`, you can check for that role.

Secondly, you can use a special "attribute" in place of a role:

You can use `IS_AUTHENTICATED_FULLY` anywhere roles are used: like `access_control` or in Twig.

`IS_AUTHENTICATED_FULLY` isn't a role, but it kind of acts like one, and every user that has logged in will have this. Actually, there are some special attributes like this:

- `IS_AUTHENTICATED_REMEMBERED`: All logged in users have this, even if they are logged in because of a "remember me cookie". Even if you don't use the [remember me functionality](#), you can use this to check if the user is logged in.
- `IS_AUTHENTICATED_FULLY`: This is similar to `IS_AUTHENTICATED_REMEMBERED`, but stronger. Users who are logged in only because of a "remember me cookie" will have `IS_AUTHENTICATED_REMEMBERED` but will not have `IS_AUTHENTICATED_FULLY`.
- `IS_REMEMBERED`: Only users authenticated using the [remember me functionality](#), (i.e. a remember-me cookie).
- `IS_IMPERSONATOR`: When the current user is [impersonating](#) another user in this session, this attribute will match.



Symfony

Security (Symfony Docs)



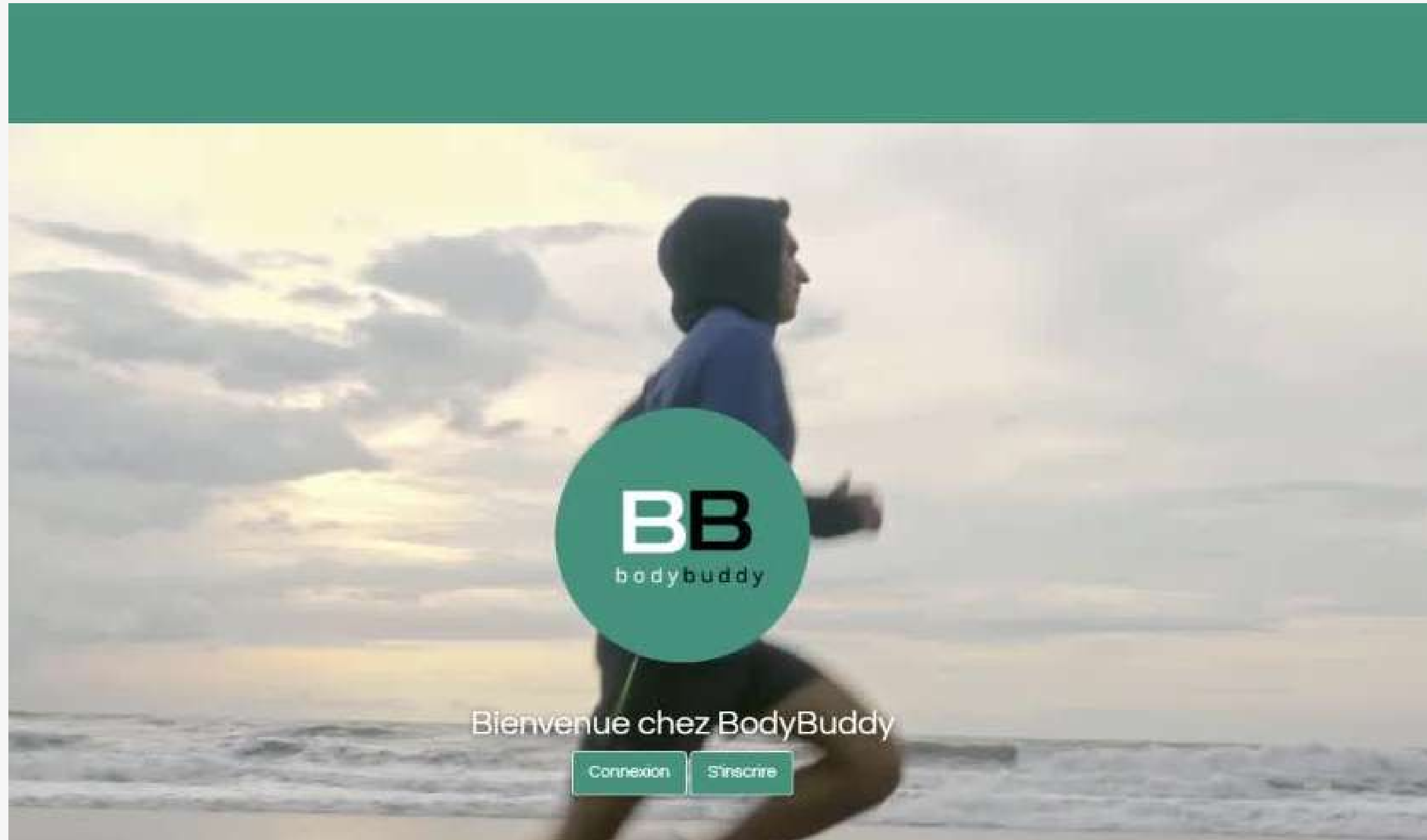
Site anglophone

Checking to see if a User is Logged In (IS_AUTHENTICATED_FULLY)

Vérification de la connexion d'un utilisateur (IS_AUTHENTICATED_FULLY)

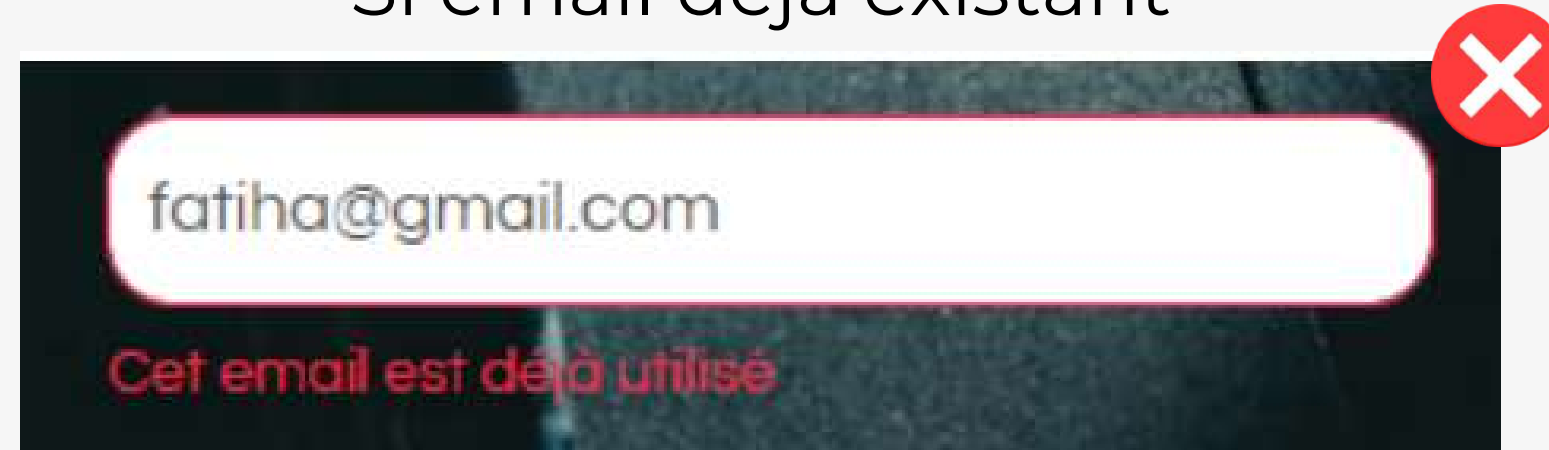
```
<!-- Button trigger modal -->
{% if is_granted('IS_AUTHENTICATED_REMEMBERED') == false %}
    <button type="button" class="btn btn-primary btn-seance" data-mdb-toggle="modal" data-mdb-target="#connexion">
        Connexion
    </button>
<!-- Modal -->
<div class="modal fade" id="connexion" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog ">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Connexion</h5>
                <button type="button" class="btn-close" data-mdb-dismiss="modal" aria-label="Close"></button>
            </div>
            <div class="modal-body p-0 g-0 m-0 d-flex justify-content-center align-items-center">
                {% include "login/modal_connexion.html.twig" %}</div>
            <div class="modal-footer"></div>
        </div>
    </div>
</div>
{% else %}
    <a class="btn-seance" href="{{ path('seance_index') }}">Liste des séances</a>
{% endif %}
```

Jeu d'essai



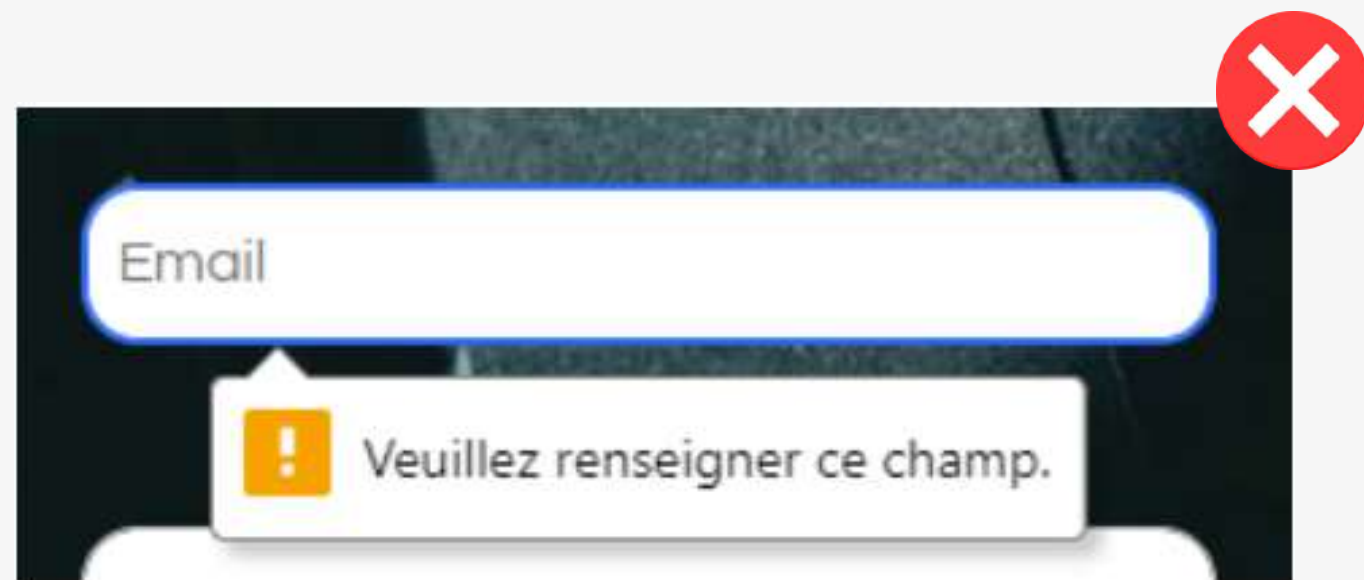
Jeu d'essai

Si email déjà existant

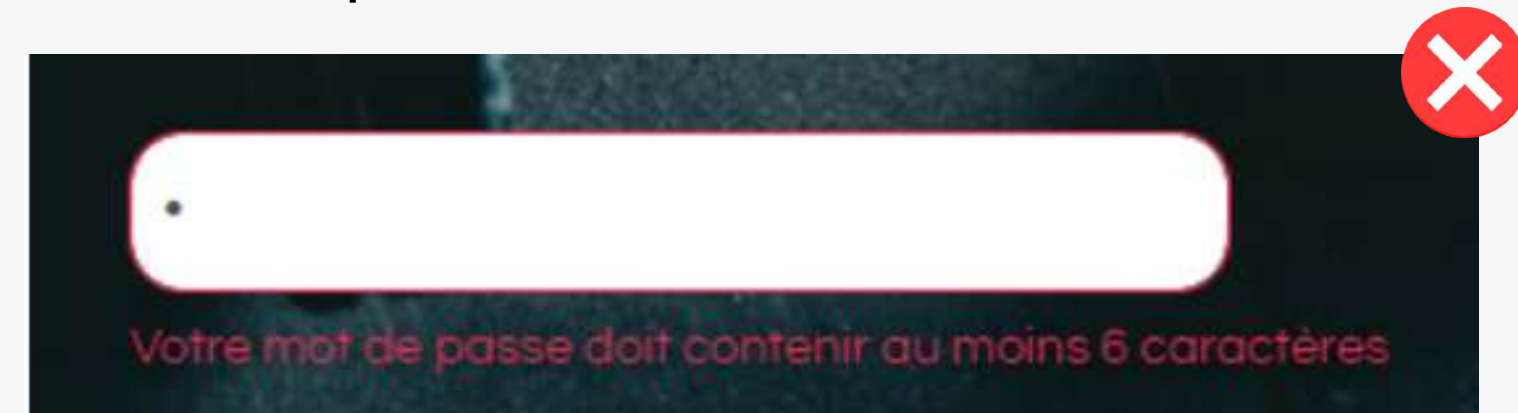


```
#[UniqueEntity(fields: ['email'], message: 'Cet email est déjà utilisé')]
```

Si mot de passe inférieur à 6 caractères



Si champ vide



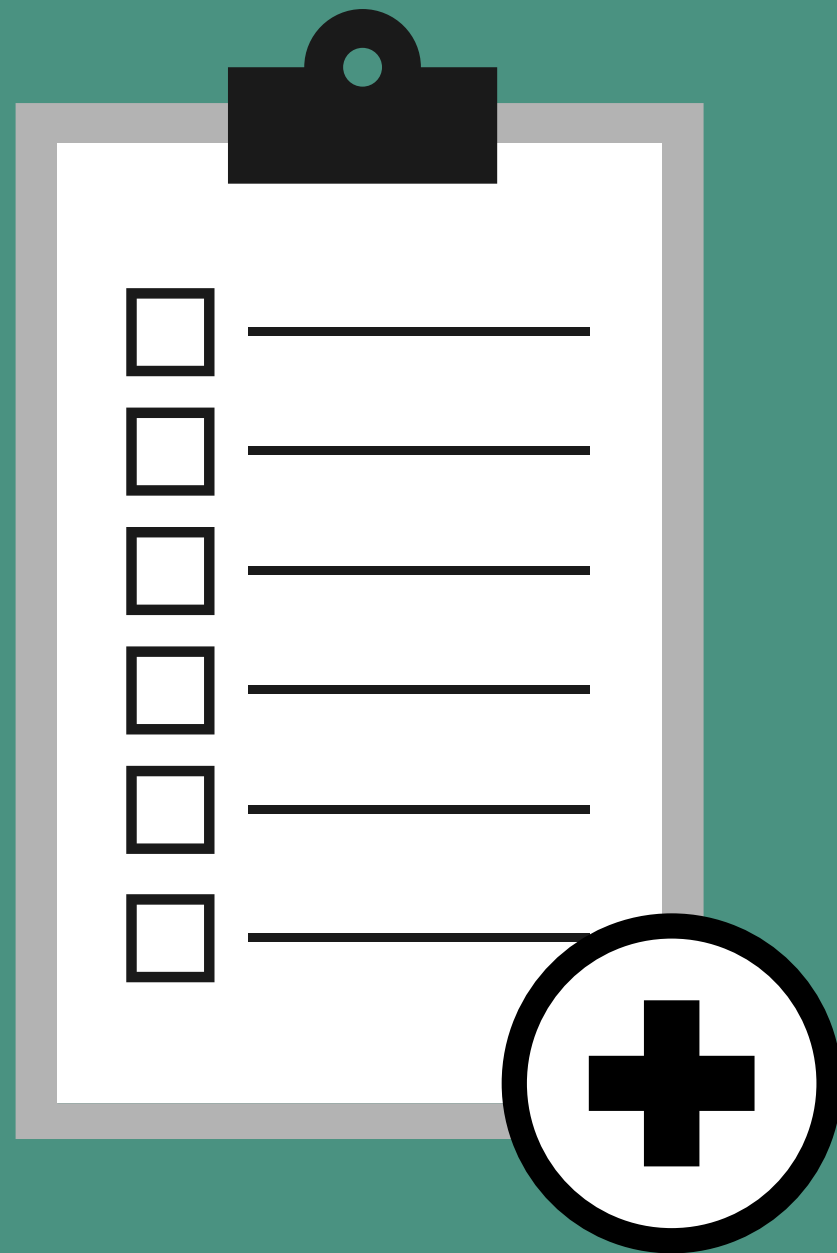
```
new Length([
    'min' => 6,
    'minMessage' => 'Your password should be at least {{ limit }} characters',
    // max length allowed by Symfony for security reasons
    'max' => 4096,
```

Synthèse projet :



- Montée de niveau vers Symfony 6
- Mise en place d'un CRUD
- Créer un espace utilisateur (connexion/inscription)
- Listing des séances spécifiques à l'utilisateur connecté
- Générer un pdf de la séance

Synthèse projet :



- Filtres en AJAX
- Renforcer la sécurité
- Mot de passe oublié
- Ajout de rôle pour les Users



Formation : Développeur Web et Web Mobile

Merci pour votre écoute

Fatiha SADEQ