

Présentation du projet  
de site vitrine pour un

Coiffeur barbier

l'Atelier



Créé en 2017

1 dirigeant  
2 employés



# Cahier des charges

- Adresse avec Google maps, téléphone
- Ajout de photos de réalisations
- Tarifs des prestations
- Marques en vente au Salon
- Liens vers les réseaux sociaux de l'entreprise
- Création d'un logo

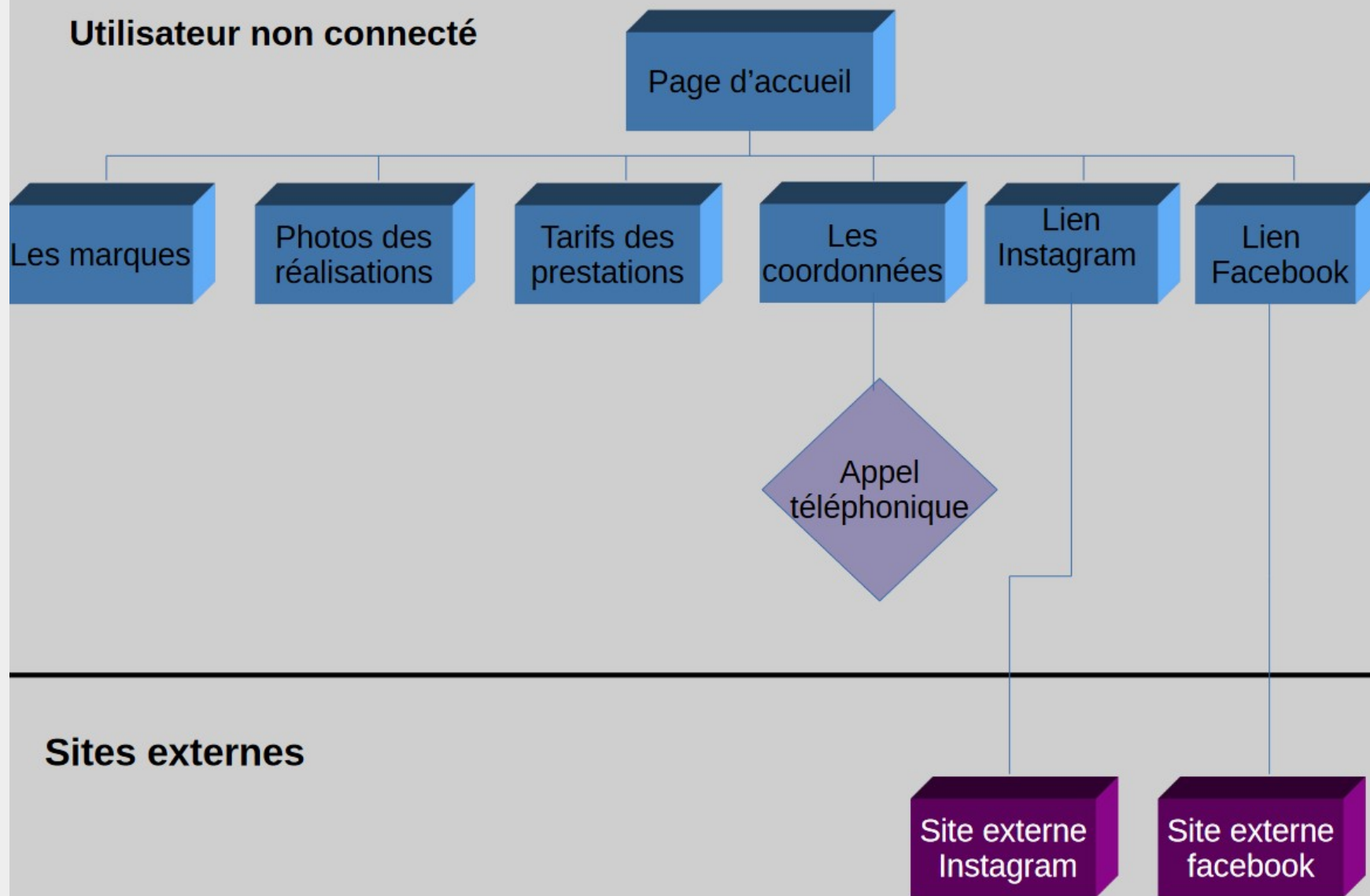
# Spécifications techniques

- Framework Symfony
- Easy Admin
- Vich Uploader
- Bootstrap

# Logiciels et outils

- VS Code
- Git - Github (SSH)
- LibreOffice
- Gimp
- Kanban

## Utilisateur non connecté



Utilisateur non connecté

Page de login

Utilisateur connecté en tant que  
« ROLE\_ADMIN » ou  
« ROLE\_SUPER\_ADMIN »

Page d'accueil  
d'administration

M : Modifier  
A : Ajouter  
S : Supprimer

Aide

Prestations

Informations

Horaires

Images

M

A

S

M

A

S

M

A

S

M

A

S

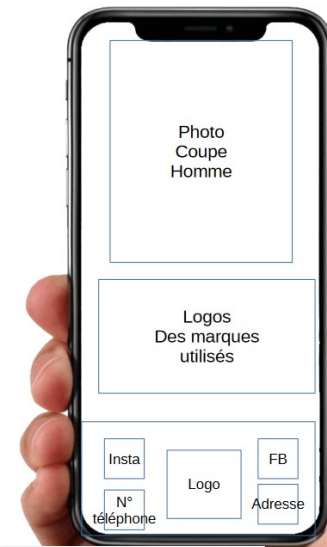
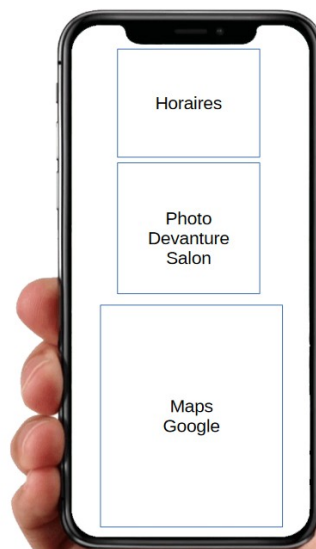
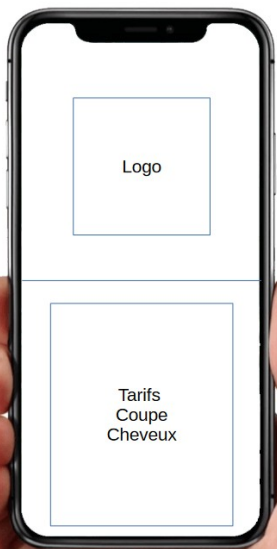
Utilisateur connecté en tant que  
« ROLE\_SUPER\_ADMIN »

Comptes  
utilisateurs

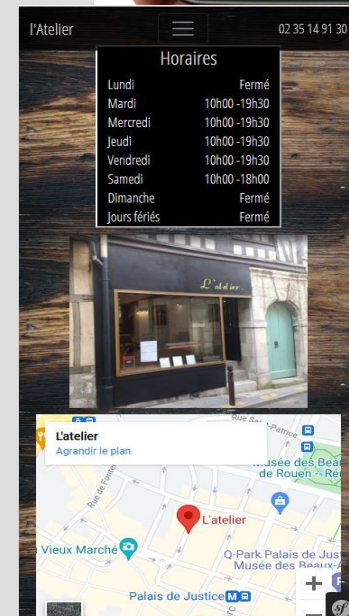
M

S

# Wireframe mobile



```
<section class="container-fluid py-5" id="serviceContainer">
  <div class="row d-flex justify-content-around">|
    <div class="col-10 col-lg-3 mb-3 serviceContainer_Services">
      <h2 class="text-center">Coupes</h2>
    </div>
  </div>
</section>
```





## Wireframe tablette

Logo

Chevron  
Vers le  
bas

Horaires  
Du salon

Photo  
Devanture

Google maps

L'Atelier

02 35 14 91 30

### Horaires

Lundi	Fermé
Mardi	10h00 - 19h30
Mercredi	10h00 - 19h30
Jeudi	10h00 - 19h30
Vendredi	10h00 - 19h30
Samedi	10h00 - 18h00
Dimanche	Fermé
Jours fériés	Fermé

### L'Atelier

18 Rue Sainte-Croix-des-Pelletiers,  
76000 Rouen

4,9 ★★★★★ 84 avis

Agrandir le plan

Itinéraires

Musée de la  
Réunion des  
Musée des Beaux-Arts  
de Rouen Réunion...

Le Conquérant

Q-Park Palais de Justice

Musée des Beaux-Arts

Palais de Justice

McDonald's

Rue de Croisne

Rue Saint-Patrice

Rue aux Juifs

Rue Saint-Léon

Rue d'Arc

Rue de la République

Rue de la Liberté

Rue de la Paix

Rue de la Victoire

Rue de la Marne

Rue de la Seine

Rue de la Somme

Rue de la Marne

Rue de la Seine

Rue de la Somme

Rue de la Marne

Rue de la Seine

Rue de la Somme

Rue de la Marne

Rue de la Seine

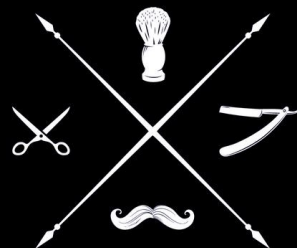
```
@media screen and (min-width:768px) and (max-width:992px){
  #logoContainer_Logo_Img{
    width: 60vw;
  }
}
```

img/logoContainer\_Logo\_img | 492 x 490.733



*L'Atelier*  
LADIES & GENTLEMEN





*l'Atelier*

LADIES & GENTLEMEN



# Rotation d'un élément HTML en JavaScript



Coupes	
COUPE TRADITIONNELLE	23€
Coupe traditionnelle de près	20€
Coupe traditionnelle - de 10 ans	15€
Couronne	15€
Coupe dégradée	27€
Coupe dégradée - de 20 ans	24€
Rasage crâne à l'ancienne	25€

Barbe	
Taille de barbe express	16€
Taille de barbe traditionnelle	21€
Rasage complet à l'ancienne	30€
Rasage collier bouc	30€
Barbe traditionnelle + soin visage	29€
Entretien moustache	10€

Colorations	
Blend effet naturel	Sur devis
Blend cheveux + barbe	29€
Mèches flash	Sur devis
Réduction de volume 1/2 tête	35€
Réduction de volume tête entière	45€

## Classe et Id sur éléments HTML

```
<section class="container-fluid py-5" id="serviceContainer">
  <div class="row d-flex justify-content-around">
    <div class="col-10 col-lg-3 mb-3 serviceContainer_Services">
      <h2 class="text-center">Coupes</h2>
```

```

```

```
/*Html elements function serviceRotate*/
services = document.getElementsByClassName('serviceContainer_Services');
logo = document.getElementById('logoContainer_Logo_Img');
let logo_Height = logo.clientHeight;
```

Éléments HTML  
et fonction JavaScript

```
window.addEventListener('scroll', function serviceRotate(){
  /*Get the scroll height and compar with header img height */
  if(window.scrollY > logo_Height){
    /*Loop on HTML Collection for add class on each element*/
    for(let i = 0; i < services.length ; i++){
      services[i].classList.add('containerRotate');
    }
  }
});
```

Propriétés CSS

```
.serviceContainer_Services.containerRotate{
  transform: rotateY(0deg);
}
```



## Création de compte

Identifiant

Votre mot de passe doit contenir au minimum:

1 minuscule

1 majuscule

1 chiffre

1 caractère #?!@\$%^&\*~

16 caractères

Mot de passe

Confirmer le mot de passe

Valider

Formulaire de création  
de compte utilisateur  
et contraintes de validation

## Création de compte

Identifiant

Votre mot de passe doit contenir au minimum:

1 minuscule

1 majuscule

1 chiffre

1 caractère #?!@\$%^&\*~

16 caractères

Mot de passe

Confirmer le mot de passe

Valider

Bouton désactivé

Bouton activé

# Fonction JavaScript

## Récupération des éléments

```
/*Html elements*/
let passwordForm = document.getElementById('registration_form_plainPassword_first');
let button = document.getElementById('button');
let lowercase = document.getElementById('lowercase');
let uppercase = document.getElementById('uppercase');
let number = document.getElementById('number');
let special = document.getElementById('special');
let passwordlength = document.getElementById('length');
/*Initialize parameter*/
let passwordMinLength = 16;
```

## Création de compte

Identifiant

Votre mot de passe doit contenir au minimum:

1 minuscule

1 majuscule

1 chiffre

1 caractère #?!@\$%^&\*~

16 caractères

Mot de passe

Confirmer le mot de passe

Valider

```
passwordForm.addEventListener('input', ()=>{
```

```
let inputPassword = passwordForm.value;
```

```
/*Lowercase control*/
```

```
if(/[a-z]/.test(inputPassword)){
```

```
/*Change Class*/
```

```
lowercase.classList.replace('alert-danger','alert-success');
```

```
}else{
```

```
lowercase.classList.replace('alert-success','alert-danger');
```

```
}
```

```
/*Upperercase control*/
```

```
if(/[A-Z]/.test(inputPassword)){
```

```
uppercase.classList.replace('alert-danger','alert-success');
```

```
}else{
```

```
uppercase.classList.replace('alert-success','alert-danger');
```

```
}
```

```
/*Number control*/
```

```
if(/[0-9]/.test(inputPassword)){
```

```
number.classList.replace('alert-danger','alert-success')
```

```
}else{
```

```
number.classList.replace('alert-success','alert-danger');
```

```
}
```

```
/* Special character control */
```

```
if(/[#?!@$%^&*~]/.test(inputPassword)){
```

```
special.classList.replace('alert-danger','alert-success')
```

```
}else{
```

```
special.classList.replace('alert-success','alert-danger');
```

```
}
```

```
/* Lenght control */
```

```
if(inputPassword.length >= passwordMinLength){
```

```
passwordlength.classList.replace('alert-danger','alert-success');
```

```
}else{
```

```
passwordlength.classList.replace('alert-success','alert-danger');
```

```
}
```

```
if(/^(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9])(?=.*[!@#$%^&*~]).{16,}$/ .test(inputPassword)){
```

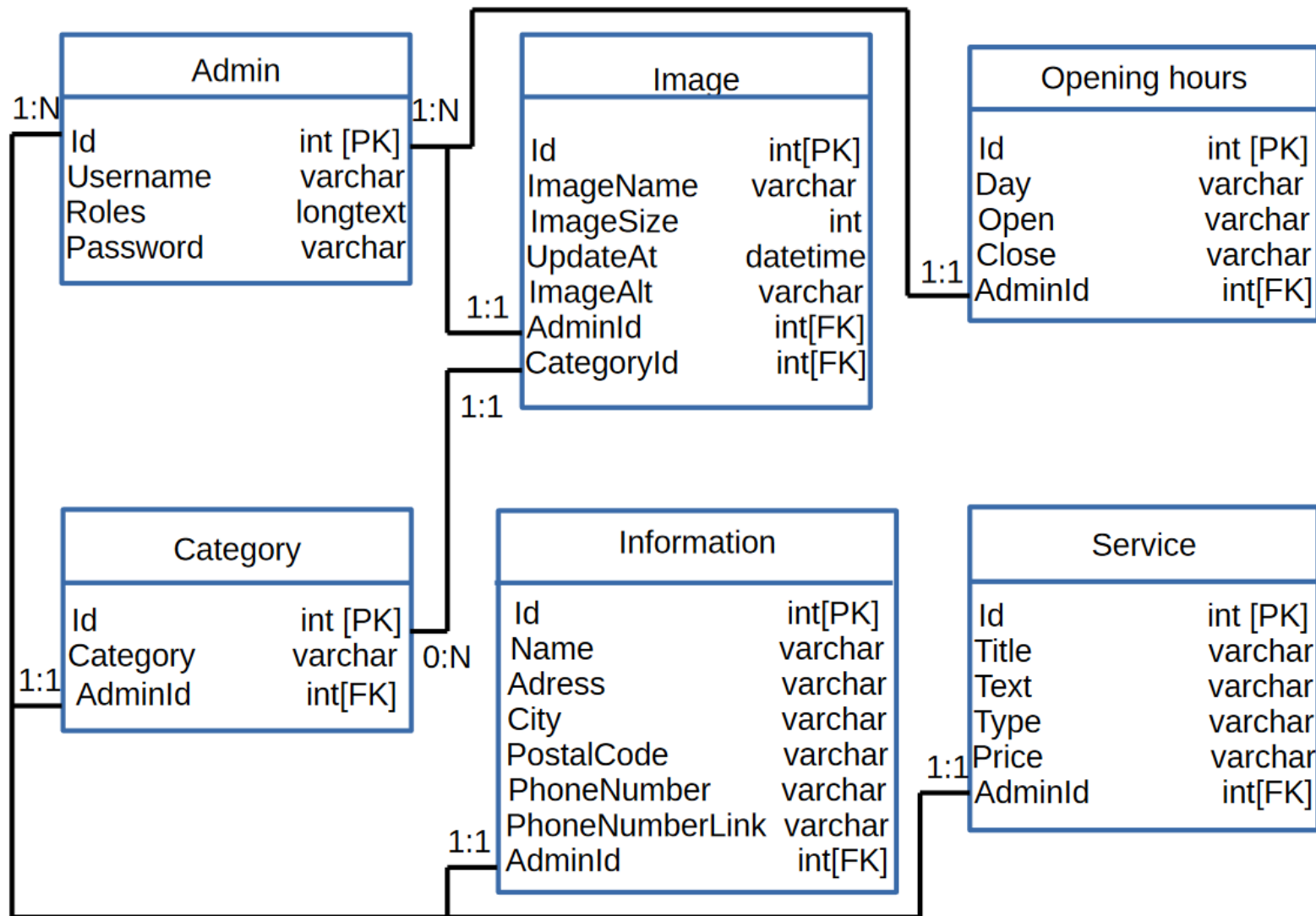
```
button.disabled = false;
```

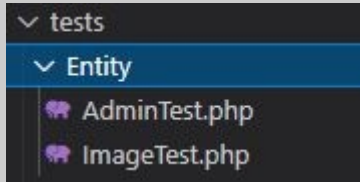
```
}else{
```

```
button.disabled = true;
```

```
}
```

```
)
```





```
class ImageTest extends TestCase
{
    public function testIsTrue()
    {
        $image = new Image();
        $file = new File('public/image/capture1-615adf3651076961319174.jpg', true);
        $dateTime = new \DateTimeImmutable();
        $category = new Category();

        $image->setImageFile($file);
        $image->setImageName('test1');
        $image->setImageSize(123);
        $image->setUpdatedAt($dateTime);
        $image->setImageAlt('test2');
        $image->setCategory($category);

        $this->assertTrue($image->getImageFile() === $file);
        $this->assertTrue($image->getImageName() === 'test1');
        $this->assertTrue($image->getImageSize() === 123);
        $this->assertTrue($image->getUpdatedAt() === $dateTime);
        $this->assertTrue($image->getImageAlt() === 'test2');
        $this->assertTrue($image->getCategory() === $category);
    }
}
```

```
class AdminTest extends TestCase
{
    public function testIsTrue()
    {
        $user = new Admin();
        $image = new Image();
        $openingHour = new OpeningHours();
        $service = new Service();
        $information = new Information();

        $user->setUsername('true');
        $user->setRoles(['ROLE']);
        $user->setPassword('123');
        $user->addImage($image);
        $user->addOpeningHour($openingHour);
        $user->addService($service);
        $user->addInformation($information);

        $this->assertTrue($user->getUserIdentifier() === 'true');
        $this->assertIsArray($user->getRoles(), 'ROLE');
        $this->assertTrue($user->getPassword() === '123');
        $this->assertCount(1, $user->getImages());
        $this->assertCount(1, $user->getOpeningHours());
        $this->assertCount(1, $user->getInformation());
        $this->assertCount(1, $user->getservices());
    }
}
```



```

public function testIsFalse()
{
    $image = new Image();
    $file = new File('public/image/capture1-615adf3651076961319174.jpg',true);
    $dateTime = new \DateTimeImmutable();
    $category = new Category();

    $image->setImageFile($file);
    $image->setImageName('true');
    $image->setImageSize(123);
    $image->setUpdatedAt($dateTime);
    $image->setImageAlt('true');
    $image->setCategory($category);

    $this->assertFalse($image->getImageFile() === new File('public/image/capture2-615adf4800bb7045733282.jpg',true));
    $this->assertFalse($image->getImageName() === 'false');
    $this->assertFalse($image->getImageSize() === 1234);
    $this->assertFalse($image->getUpdatedAt()=== new \DateTimeImmutable());
    $this->assertFalse($image->getImageAlt() === 'false');
    $this->assertFalse($image->getCategory() === new Category());
}

```

```

public function testIsFalse()
{
    $user = new Admin();
    $image = new Image();
    $openingHour = new OpeningHours();
    $service = new Service();
    $information = new Information();

    $user->setUsername('test');
    $user->setRoles(['ROLE']);
    $user->setPassword('123');
    $user->addImage($image);
    $user->addOpeningHour($openingHour);
    $user->addService($service);
    $user->addInformation($information);

    $this->assertFalse($user->getUserIdentifier() === 'false');
    $this->assertIsArray($user->getRoles(), 'false');
    $this->assertFalse($user->getPassword() === 'false');
    $this->assertFalse($user->getImages() === new Image());
    $this->assertFalse($user->getOpeningHours() === new OpeningHours);
    $this->assertFalse($user->getServices() === new Service());
    $this->assertFalse($user->getInformation() === new Information());
}

```

```
public function testIsEmpty()  
{  
    $image = new Image();  
  
    $this->assertEmpty($image->getImageFile());  
    $this->assertEmpty($image->getImageName());  
    $this->assertEmpty($image->getImageSize());  
    $this->assertEmpty($image->getUpdatedAt());  
    $this->assertEmpty($image->getImageAlt());  
    $this->assertEmpty($image->getCategory());  
}
```

```
public function testIsEmpty()  
{  
    $user = new Admin();  
  
    $this->assertEmpty($user->getUserIdentifier());  
    $this->assertIsArray($user->getRoles(), '');  
    $this->assertEmpty($user->getImages());  
    $this->assertEmpty($user->getopeningHours());  
    $this->assertEmpty($user->getServices());  
    $this->assertEmpty($user->getInformation());  
}
```

Testing

.....

6 / 6 (100%)

Time: 00:00.027, Memory: 10.00 MB

OK (6 tests, 38 assertions)

Contrôleur  
du formulaire  
de création d'un  
compte utilisateur

```
class RegistrationController extends AbstractController
{
    #[Route('/registerFormWebsite76', name: 'app_register')]
    public function register(Request $request, UserPasswordHasherInterface $passwordHasher): Response
    {
        $user = new Admin();
        $form = $this->createForm(RegistrationFormType::class, $user);
        $form->handleRequest($request);

        if ($form->isSubmitted() && $form->isValid()) {
            $user->setPassword(
                $passwordHasher->hashPassword(
                    $user,
                    $form->get('plainPassword')->getData()
                )
            );

            $entityManager = $this->getDoctrine()->getManager();
            $entityManager->persist($user);
            $entityManager->flush();

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

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

## Authentifier un utilisateur avec Authenticator

```
class Authenticator extends AbstractLoginFormAuthenticator
{
    use TargetPathTrait;

    public const LOGIN_ROUTE = 'app_login';

    private UrlGeneratorInterface $urlGenerator;

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

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

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

        return new Passport([
            new UserBadge($username),

            new PasswordCredentials($request->request->get('password', '')),
            [
                new CsrfTokenBadge('authenticate', $request->get('_csrf_token')),
            ]
        ]);
    }
}
```

Limiter le nombre  
de tentatives de connexion

```
login_throttling:  
  max_attempts: 4  
  interval: '180 minutes'
```

## Entité Image

```
/**
 * @Assert\File(
 *     maxSize = "200k",
 *     mimeTypes = {"image/x-icon", "image/jpeg", "image/jpg", "image/png"},
 *     mimeTypesMessage = "Poids max 200 Ko et format 'Image' uniquement"
 * )
 * @Vich\UploadableField(mapping="upload_img", fileNameProperty="imageName", size="imageSize")
 *
 * @var File|null
 */
private $imageFile;
```


```
 * @param File|\Symfony\Component\HttpFoundation\File\UploadedFile|null $imageFile
 */
public function setImageFile(?File $imageFile = null): void
{
    $this->imageFile = $imageFile;

    if (null !== $imageFile) {
        // It is required that at least one field changes if you are using doctrine
        // otherwise the event listeners won't be called and the file is lost
        $this->updatedAt = new \DateTimeImmutable();
    }
}
```

```
public function findByCategory($value)
{
    return $this->createQueryBuilder('i')
        ->setParameter('val', $value)
        ->where('i.category = :val')
        ->orderBy('i.updatedAt', 'DESC')
        ->setMaxResults(10)
        ->getQuery()
        ->getResult()
    ;
}
```

# Security

Version: **current** >

 Edit this page

- # [Logged in User Information](#)
- # [Restrict Access to the Entire Backend](#)
- # [Restrict Access to Menu Items](#)
- # [Restrict Access to Actions](#)
- # [Restrict Access to Fields](#)
- # [Custom Security Voters](#)

EasyAdmin relies on [Symfony Security](#) for everything related to security. That's why before restricting access to some parts of the backend, you need to properly setup security in your Symfony application:

1. [Create users](#) in your application and assign them proper permissions (e.g. `ROLE_ADMIN`);
2. [Define a firewall](#) that covers the URL of the backend.

## Logged in User Information

When accessing a protected backend, EasyAdmin displays the details of the user who is logged in the application and a menu with some options like "logout". Read the [user menu reference](#) for more details.

## Restrict Access to the Entire Backend

Using the [access\\_control option](#), you can tell Symfony to require certain permissions to browse the URL associated to the backend. This is simple to do because [each dashboard only uses a single URL](#):

Traduction

## Sécurité

Version : actuelle

- Information de l'utilisateur connecté
- Restreindre l'accès à toutes la partie back-end
- Restreindre l'accès aux éléments du Menu
- Restreindre l'accès aux Actions
- Restreindre l'accès aux champs
- Personnaliser les Voters de sécurité

Easy Admin repose sur la Symfony Security pour tout ce qui concerne la sécurité. C'est pourquoi avant de restreindre l'accès à certaines parties du back-end, vous devez configurer convenablement la sécurité de votre application Symfony.

1 Créez des utilisateurs dans votre application et assignez leur des permissions appropriées.

2 Définissez un firewall qui concerne les URL du back-end.

Information de l'utilisateur connecté

Quand l'accès au back-end est protégé, Easy Admin affiche les détails de l'utilisateur loggé et un menu avec quelques options comme « déconnexion ». Lisez le Menu de référence de l'utilisateur pour plus de détails.

Restreindre l'accès à toute la partie back-end

En vous servant de l'« `access_control` option », vous pouvez dire à Symfony d'exiger certaines permissions pour naviguer sur les URL associés au back-end.



## Conclusion

Merci



- Aide
- Image
- Prestation
- Horaires
- Coordonnées

Ajouter une image

Image\*

Choose file

Categorie\*

Coupe Homme

- Barbe
- Coupe Homme
- Coupe Femme
- Les Marques
- Devanture
- Logo Header

Create and add another

Create

- Aide
- Image
- Prestation
- Horaires
- Coordonnées

Image

Filters

Add Image

<input type="checkbox"/>	Auteur	Nom du fichier	Image	Categorie	Attribut HTML "ALT"(courte description pour référencement naturel)	
<input type="checkbox"/>	az	logo-615ac51dbb7c3883561314.png		Logo Header	Logo du salon l'Atelier Rouen	...
<input type="checkbox"/>	az	devanture-615ac5f25e786853074211.jpg		Devanture	18 rue Sainte Croix des Pelletiers Rouen	...
<input type="checkbox"/>	az	logo-keune-615ac66e8b239737587752.png		Les Marques	Logo Keune	...
<input type="checkbox"/>	az	logo-davines-615ac6a566166524670994.png		Les Marques	Logo Davines	...
<input type="checkbox"/>	az	logo-reuzel-615ac6b99fcad177125316.png		Les Marques	Logo Reuzel	...
<input type="checkbox"/>	az	capture1-615adf3651076961319174.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	az	capture2-615adf4800bb7045733282.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	az	capture3-615adf57c267c254421503.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	azeryy	capture4-615bf3f0f63c5891500103.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	azeryy	capture5-615bf40f70fe919557213.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	az	capture6-615bf41a70965263971106.jpg		Coupe Femme	Coupe	...
<input type="checkbox"/>	az	capture7-615bf42604f11794091446.jpg		Coupe Femme	Coupe	...















































































