# TP5: Upload de fichiers

## Objectif

L'objectif de ce TP est de proposer d'uploader un fichier depuis un formulaire de manière sécurisée.

## Champ d'upload

Notre application va stocker une photo des coasters, pour celà vous devez ajouter un nouveau champ *imageFileName* (string nullable) dans l'entité Coaster.

### Ajout du champ

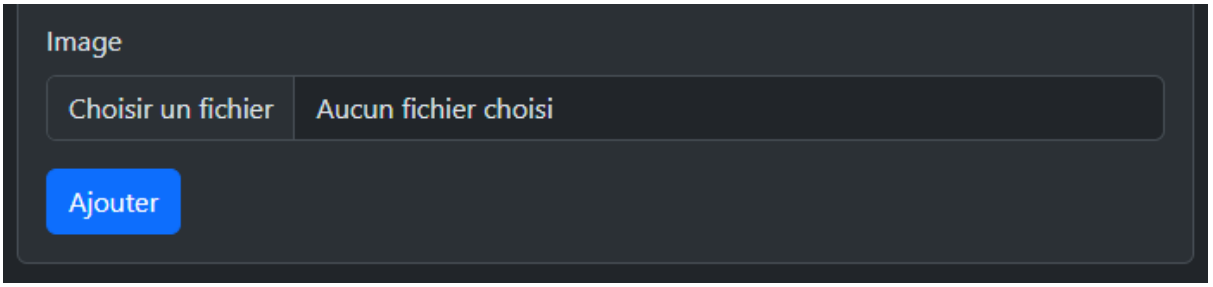
Si vous ajoutez un champ *imageFileName* dans le formulaire, l'utilisateur va avoir un champ texte à remplir, or, le formulaire doit plutôt afficher un champ de type ***file*** et le nom de l'image sera généré automatiquement.

Ajoutez un champ "image" au formulaire de coaster, celui-ci doit être de type *FileType*.

Par défaut, les champs de formulaire sont automatiquement associés au propriétés de l'entité, si un champ ne correspond pas à une propriété, une erreur sera levée. Pour que le champ "image" soit accepté, vous devez ajouter l'option suivante dans la définition du champ:

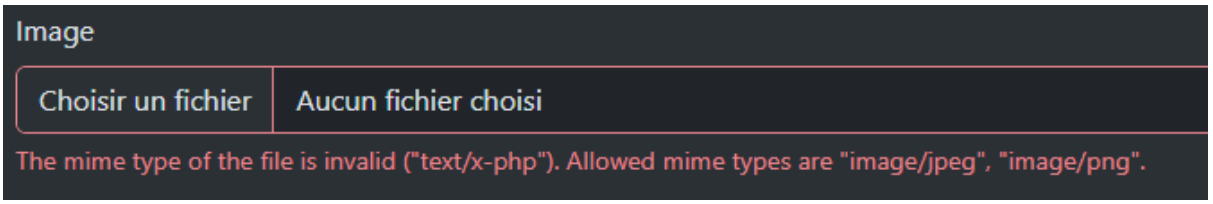
```
'mapped' => false,
```

N'oubliez pas également de définir l'option required à false.




## Types de fichiers

A cette étape, l'utilisateur a la possibilité d'envoyer n'importe quel type de fichier dont un fichier PHP ce qui est bien sûr très dangereux.




Dans la définition du champ de formulaire, ajoutez une option "constraints" avec un objet "Symfony\Component\Validator\Constraints\Image".

Aidez-vous de la doc:
https://symfony.com/doc/current/validation.html#constraints-in-form-classes

## Service FileUploader

L'upload du fichier peut se faire dans une classe à part pour bien séparer les fonctionnalités. Créez un dossier "Service" dans "src" et ajoutez la classe suivante.

```php
<?php

namespace App\Service;

use Symfony\Component\DependencyInjection\Attribute\Autowire;
use Symfony\Component\HttpFoundation\File\UploadedFile;

class FileUploader
{
    public function __construct(
        #[Autowire('%kernel.project_dir%/public/uploads')]
        private string $uploadFolder
    ) {}
```

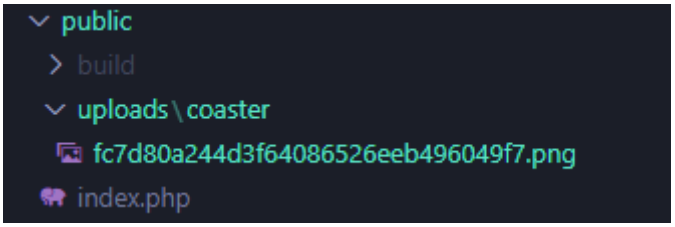
```
    public function upload(UploadedFile $file): string
    {
        $fileName = md5(uniqid()).'.'.$file->guessExtension();

        $file->move($this->uploadFolder, $fileName);

        return $fileName;
    }
}
```

L'attribut "Autowire" permet d'indiquer quelle valeur donner à ce paramètre lorsque Symfony va créer l'objet.

Vous pouvez maintenant utiliser l'objet dans le controller lors de l'ajout et la modification d'un coaster, pour récupérer l'image uploadée, utilisez ce code:

```
$imageFile = $form->get('image')->getData();
```

Vérifiez que le fichier est bien  uploadé dans le dossier.




## Afficher l'image

Pour afficher la photo dans une vue Twig, utilisez la fonction "asset": https://symfony.com/doc/current/templates.html#linking-to-css-javascript-and-image-assets

# Supprimer un fichier

Lorsqu'un coaster est supprimé ou que sa photo est modifiée, il convient de supprimer le fichier qui n'est plus utile.

Utilisez la méthode native de PHP "unlink" pour supprimer un fichier, attention si le fichier n'existe pas "unlink" va lever une erreur, il faut donc d'abord tester si le fichier existe avant d'appeler la méthode.

Cette méthode doit être appelée dans les actions de modification et de suppression d'un coaster.