UPLOAD IMAGE SYMFONY

Il existe différentes technique pour la gestion des fichiers uploadés en Symfony, dont notamment la bibliothèque (bundle) : **VichUploaderBundle**.

Voici les différentes étapes de l'installation et de la mise en place de cette bibliothèque :

1ère étape :

Nous allons demander à COMPOSER (gestionnaire de dépendances) d'installer cette bibliothèque sur le projet Symfony. Cette bibliothèque est héberger sur GITHUB et disponible à cette adresse :

https://github.com/dustin10/VichUploaderBundle

Ouvrir un terminal et exécuter la commande suivante :

composer require vich/uploader-bundle

"Do you want to execute this recipe ?" ==> Yes

Vérifiez la présence du package dans : config > bundles.php

"Vich\UploaderBundle\VichUploaderBundle::class => ['all' => true],"

Créez dans le fichier "public" un fichier "images", dans lequel il y aura un fichier "produits" (par exemple)

public > images > produits

Vous devez maintenant indiquer à SYMFONY le chemin vers le dossier de destination des fichiers uploader, rendez-vous dans le fichier **config > packages > vich_uploader.yaml**

```
vich_uploader:
  db_driver: orm
  mappings:
    produits_image:
      uri_prefix: /images/produits
      upload_destination: '%kernel.project_dir%/public/images/produits'
      namer: Vich\UploaderBundle\Naming\UniqidNamer
```

2ème étape :

Rendez-vous dans votre Entité (src/Entity) qui réceptionnera les fichiers uploadés. Exemple : si vous souhaitez ajouter des images de produits, vous devez vous rendre dans l'entité : **src/Entity/product**

Ajouter les importations de classes suivantes :

```
use Symfony\Component\HttpFoundation\File\File;

use Vich\UploaderBundle\Mapping\Annotation as Vich;

use Symfony\Component\HttpFoundation\File\UploadedFile;
```

Afin d'indiquer à SYMFONY que nous allons utiliser la bibliothèque **VichUploaderBundle** dans l'entité, ajouter l'annotation suivante directement sur la classe :

```
/**
 * @ORM\Entity(repositoryClass="App\Repository\*******Repository")
 * @Vich\Uploadable
 */
class ******{
...
}
```

Puis toujours dans l'entité, ajouter une propriété privée qui ne sera pas enregistrée en BDD mais qui réceptionnera l'image uploadée.

```
/**
    * @Vich\UploadableField(mapping="produits_image", fileNameProperty="image")
*/
private $imageFile;
```

(le mapping est mentionné dans config > packages > vich_uploader.yaml ligne 5 en l'occurrence ici il s'agit de : produits_image)

Maintenant nous allons créer le getter / setter correspondant :

```
public function getImageFile(): ?File
{
    return $this->imageFile;
 }
```

```
public function setImageFile(?File $imageFile = null): self
{
    $this->imageFile = $imageFile;

    if($this->imageFile instanceof UploadedFile){
        $this->updated_at = new \DateTime('now');
    }
    return $this;
}
```

Indiquer maintenant que la propriété privée **image** peut être nulle en ajouter un point d'interrogation **?** en argument de la méthode **setImage()**

```
public function getImage(): ?string
{
    return $this->image;
}

public function setImage(?string $image): self
{
    $this->image = $image;

    return $this;
}
```

3ème etape :

Modifiez l'entity, allez dans les commandes : php bin/console make:entity ***

puis rajoutez un nouveau champs : updated_at

=> le fait de mettre "_at" permet de définir le type "datetime"

Ne pas oublier :

php bin/console make:migration (php bin/console m:m)

php bin/console doctrine:migration:migrate (php bin/console d:m:m)

Voici le résultat :

```
/**
   * @ORM\Column(type="datetime")
*/
private $updated_at;

public function getUpdatedAt(): ?\DateTimeInterface
{
    return $this->updated_at;
}

 public function setUpdatedAt(\DateTimeInterface $updated_at): self
 {
    $this->updated_at = $updated_at;
     return $this;
  }
```

4ème étape :

Afin d'indiquer à SYMFONY que ce champ est de type **file** en HTML, dans le formulaire d'ajout correspondant à l'entité, remplacer le champ **add('image')** par :

->add('imageFile', FileType::class,['required' => false])

Ne pas oublier d'importer la classe suivante :

use Symfony\Component\Form\Extension\Core\Type\FileType;

5ème étape :

Enfin dans le Template de rendu TWIG, afin d'afficher les images uploadées, mentionner dans la balise <img> , le chemin vers le dossier photo dans le dossier public grâce à la méthode **asset()** de TWIG dans l'attribut **src** de l'image :

<img src="{{ asset('images/article/' ~ article.image) }}" alt="" class="col-md-12">