Vetux-Line

Contexte

Nous travaillons pour l'entreprise Vetux-Line, un createur de ligne de vêtements. Notre mission consiste à créer une application Web qui permet de manipuler des fichiers CSV.

Partie 1

Login

Quand ouvre l'application, on arrive sur la page de connexion. Pour créer cette page, on s'est aidé du TD Secufony qu'on a fait en cours. Le but est que seul les utilisateurs ayant le rôle de gestionnaire puisse accéder à la partie fusion des fichiers csv. Les utilisateurs inscrits sont sauvegardé dans la base de données avec leur rôle. Pour l'instant, lors de l'inscription les utilisateurs ont directement le rôle de gestionnaire. Ensuite lorsque nous nous connectons il y a 2 possibilité: soit on a pas le rôle de gestionnaire et cela nous renvoie vers une page basique avec un simple message, soit on a le rôle de gestionnaire et on accède à la partie upload.

```
/**
* @Route("/upload", name="upload")
*/
public function index(Request $request): Response
{
        $role=$this->getUser()->getRoles();
        $hasAccess=$this->isGranted("ROLE_GESTION");  //true si l'utilisateur est le role "ROLE_GE
        if  ($hasAccess){
            return $this->render('upload/index.html.twig', [      //La page qui permet réaliser l'
                'controller_name' => 'UploadController',
            ]);
        }
        else{
            return $this->render('notallow.html.twig',['role'=>$role[0]]);  //redirection vers une
        }
    }`
```

Upload

Nous arrivons ensuite sur la page qui permet d'upload les 2 fichiers pour les traiter. Sur cette page on peut upload les 2 fichier (french et german data). On peut upload seulement des fichier .csv car c'est le seul type de fichier qu'on va utiliser.

```
'upload/index.html.twig'

 <form action="{{ path('do-upload') }}" method="post" enctype="multipart/form-data">

        <input type="hidden" name="token" value="{{ csrf_token('upload') }}" />

        <div>
            <label for="file1">Premier fichier à fusionner:</label>
            <input type="file" accept=".csv" name="file1" id="file1"  >
        </div>
```

Les fichiers seront stocké dans `var/uploads` comme configuré dans le fichier services.yaml et les fichiers auront comme nom de fichier `file1.csv` et `file2.csv`

Fusion

La fonction principales de cette application et la fusion des 2 fichiers csv. Pour cela nous devons d'abord lire les fichiers. Nous avons utiliser le composant `league/csv` . Nous l'avons choisit pour plusieurs raison: elle correspond à nos besoin c'est à dire lire et transformer un fichier csv en Iterator et écrire des fichiers csv, c'est le composant le plus téléchargé et la documentation est facile à comprendre. Ensuite les opérations de fusions sont dans une classe `FusionClass.php` en tant que service. On injecte ces services dans les controllers :

```
public function fusion(FusionClass $fusion)
```

Puis on utilise la fonctions `melange` de ce service dans le controller:

```
$fusion->melange(true, $records1, $records2, $tabName, $output);
```

Les 2 types de fusion(séquentiel et entrelacé) sont fonctionnels mais nous pas n'avons pas fait de système pour que le client puisse choisir le type de fusion. Le 1er parametre de la fonction melange définit le type de fusion (true=séquentiel et false=entrelacé). Les différents filtres(personnes majeures, contrainte de la taille, contrainte de ccn) sont appliqué dans la fonction `filtre` du service.

```
    public function filtre(array $record)
    {
        $carte = [];
        $t = $record["FeetInches"];
        $tb = explode("'", $t);
        $comv = ((int)$tb[0] + ((int)$tb[1] / 10)) * 30.48;
        $bool = ((int)$comv - 1 <= $record["Centimeters"]) && ((int)$comv + 1 >= $record["Centimet
        if ($bool) {
            $t = $record["Birthday"];
            $tb = explode("/", $t);
            if (((int)$tb[2] <= (date('Y') - 18))) {  //condition de l'age
                $ccn = $record["CCNumber"];
                if ((!in_array($ccn, $carte))) {  //condition du ccn
                    array_push($carte, $ccn);
                    return true;
                }
```

```
            }
        }
        return false;
    }
```

L'utilisation des services permet de se reperer plus facilement et de ne pas tout mélangé.

Download

Lorsque le programme aura fini d'executer la fonction fusion, il nous redirigera vers une page html qui contient un bouton, ce bouton éxecutera la fonction download. La fonction va permettre au client de télécharger le fichier fusionner sur son disque dur ou autre.

```
    /**
     * @Route("/download",name="download")
     */
    public function download(): BinaryFileResponse
    {
        $response = new BinaryFileResponse('../public/csv/output.csv');
        $response->headers->set('Content-Type', 'text/csv');
        $response->setContentDisposition(
            ResponseHeaderBag::DISPOSITION_ATTACHMENT,
            'fusion.csv'
        );
        return $response;
    }
```

Partie 2

Dans la Partie 2, nous devions mettre les données des clients, les véhicules en particulier, dans une base de données.