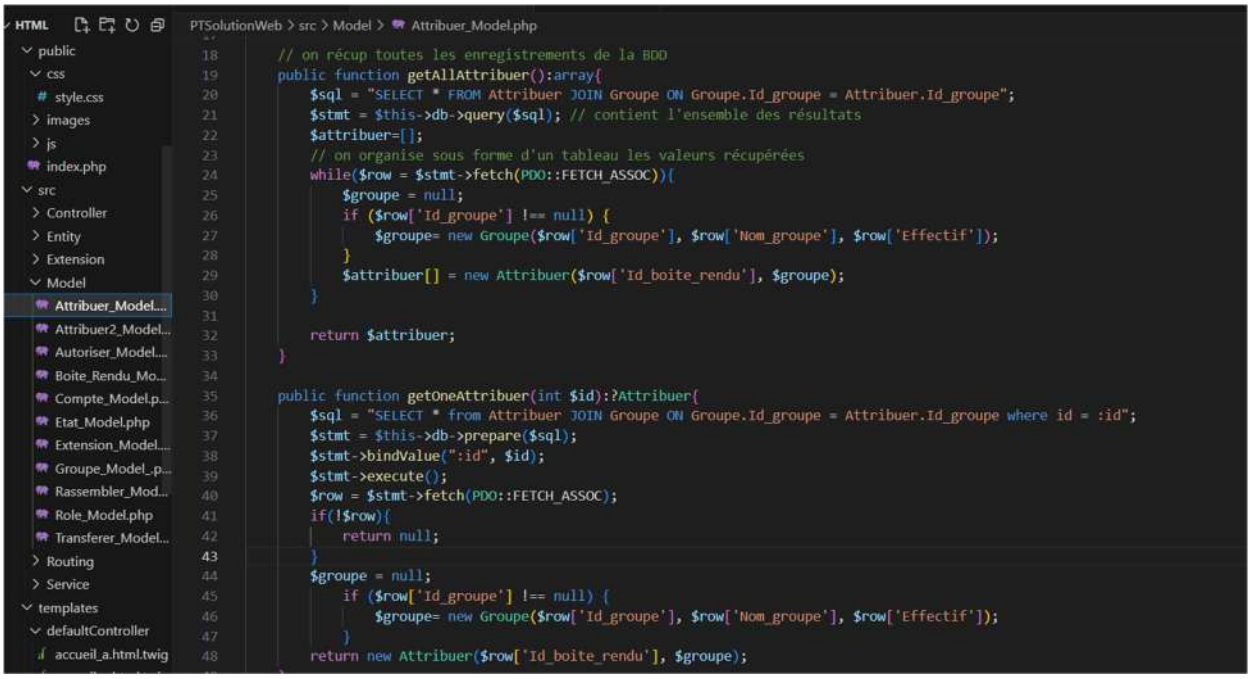
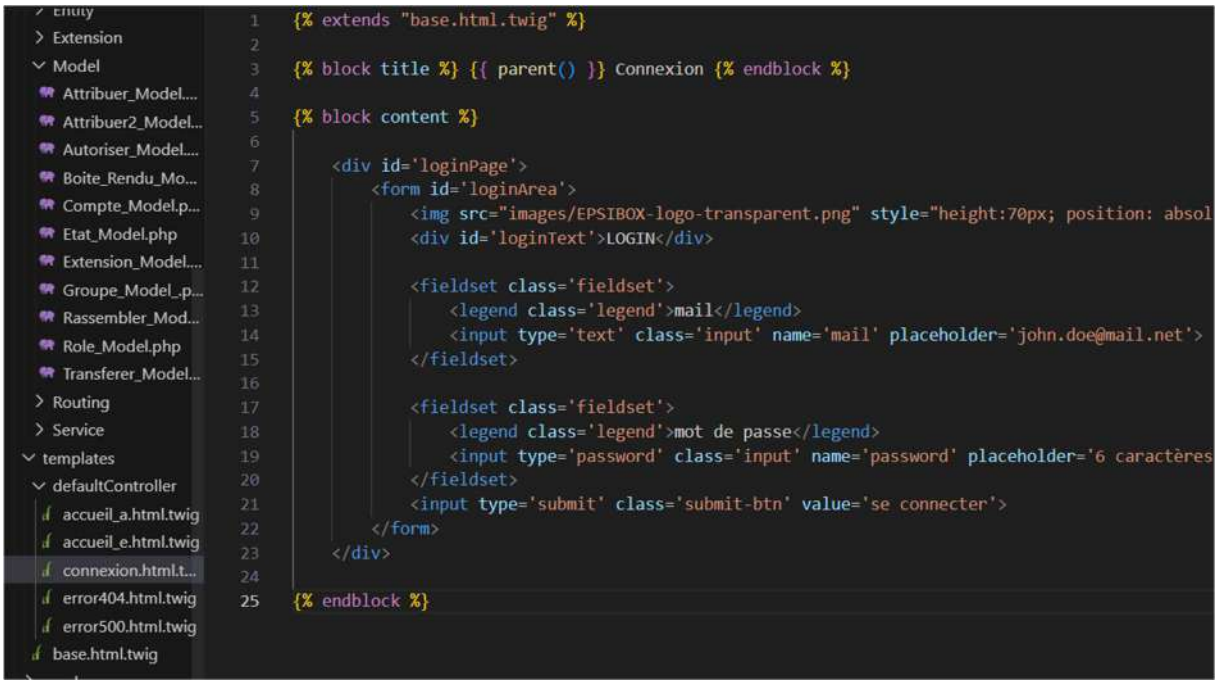
Organisation de notre code pour le projet Boite de rendu

Pour ce projet nous utilisons une organisation logicielle « modèle-vue-contrôleur » pour répartir dans notre code et pouvoir organiser celui-ci afin d'être plus productif et pouvoir naviguer plus facilement :

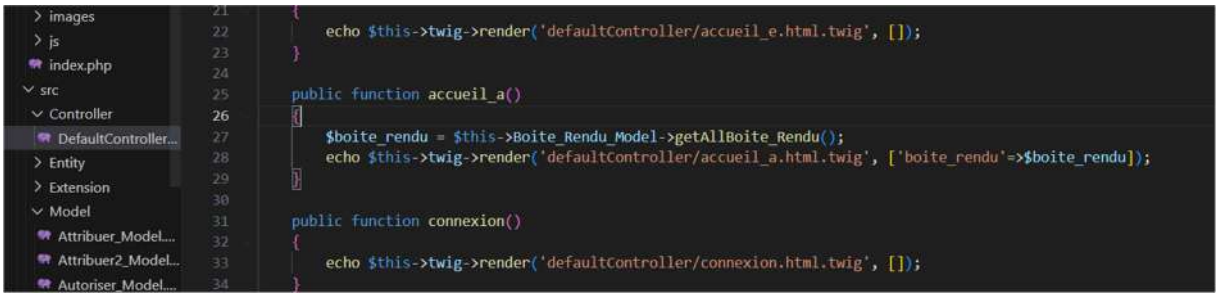
- Un modèle (Model) contient nos données à afficher qu'il vient récupérer par une connexion faite à notre base de donnée avec PDO;



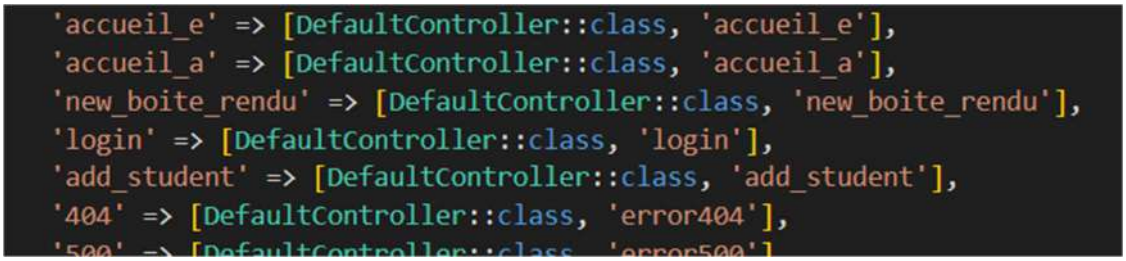
- Une vue (View) contient la présentation de l'interface graphique, ce sont tous les éléments affichés sur notre page c'est ici que nous allons écrire du code HTML ainsi que du twig qui est une sur-couche utilisé habituellement avec le framework Symphony, et qui permet de faire des boucles, des conditions ou de récupérer des variables de nos contrôleurs pour modifier dynamiquement l'apparence d'une vue;



- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur, on y met le code php et récupère les données de certaines tables en faisant appels aux méthodes de nos modèles, et on vient les passer dans la vue via un tableau.



Mise en place d'une interface de création de boite de rendu



Dans un premier temps on ajoute une nouvelle page ‘new_boite_rendu’ dans notre router.php et nous créons celle-ci dans template avec une méthode dans le controlleur pour y mettre le code php. Dans la vue on met en place un formulaire pour saisir les paramètres de notre boite de rendu : Nom, Date de rendu, nombre de fichier, extension des fichiers :

```
{% block content %}
    <div>
        <h2 class="text-center"> Créer votre nouvelle boite de rendu </h2>
    </div>
    <!-- formulaire de création de boîte de rendu -->
    <form action="" method="post">
        <div class="text_center">
            <label for="nom_rendu">Nom du Rendu :</label>
            <input type="text" id="nom_rendu" name="nom_rendu" placeholder="Projet Exemple" required>

            <label>Extensions attendues :</label>
            <div>
                {% for ex in extensions %}
                <label for="{{ex.id_extension}}">{{ex.nom_extension}}</label>
                <input type="checkbox" id="{{ex.id_extension}}" name="{{ex.id_extension}} ">
                {% endfor %}
            </div>

            <label for="date_rendu">Date du Rendu :</label>
            <input type="date" id="date_rendu" name="date_rendu"  required>

            <label for="nb_fichiers">Nombre de fichiers attendus:</label>
            <input type="text" id="nb_fichiers" name="nb_fichiers" placeholder="2" required>

            <div><input type="submit" value="Suivant"></div>
        </div>
```

Dans le contrôleur on récupère toutes les extensions disponibles et les comptes utilisateurs en les plaçant dans le tableau de données du contrôleur afin qu’ils soient envoyés sur la vue.

```
public function new_boite_rendu()
{

    // on récupère toutes les tables nécessaires
    $extensions = $this->Extension_Model->getAllExtension();
    $comptes = $this->Compte_Model->getAllCompte();
    echo $this->twig->render('defaultController/new_boite_rendu.html.twig', ['extensions'=>$extensions, 'comptes'=>$
}
```

On crée dans boite_rendu_Model.php une méthode qui récupère tous les enregistrements et qui les tries du plus récent au plus ancien grâce à la valeur de de la clé primaire id.

```
// on récup toutes les enregistrements de la BDD
public function getAllBoite_Rendu():array{
    $sql = "SELECT * FROM Boite_rendu ORDER BY Id_boite_rendu DESC";
    $stmt = $this->db->query($sql); // contient l'ensemble des résultats
    $boite_rendu=[];
    // on organise sous forme d'un tableau les valeurs récupérées
    while($row = $stmt->fetch(PDO::FETCH_ASSOC)){
        $boite_rendu[] = new Boite_rendu($row['Id_boite_rendu'], $row['Date_rendu'], $row['Nom_boite_rendu'], $row['Nb_fichiers']);
    }

    return $boite_rendu;
}
```

On crée également dans Autoriser_Model.php une méthode pour créer un nouvel enregistrement permettant d’associer des extensions à une boite de rendu spécifique :

```php
    // méthode pour créer un nouvel enregistrement autoriser
    public function createAutoriser(Autoriser $autoriser): bool {
        $sql = "INSERT INTO Autoriser (Id_boite_rendu, Id_extension) VALUES (:Id_boite_rendu, :Id_extension)";
        $stmt = $this->db->prepare($sql);
        $stmt->bindValue(':Id_boite_rendu', $autoriser->getId_Boite_Rendu()->getId_Boite_Rendu(), PDO::PARAM_STR);
        $stmt->bindValue(':Id_extension', $autoriser->getId_Extension()->getId_Extension(), PDO::PARAM_STR);
        return $stmt->execute();
    }

}
```

Enfin dans le contrôleur on peut désormais récupérer en POST les données du formulaire de création d'une boite de rendu situé dans la vue new_boite_rendu.html.twig :

```php
{
    // on récupère les informations de création d'une boîte de rendu
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $date_rendu = filter_input(INPUT_POST, 'date_rendu', FILTER_SANITIZE_STRING);
        $nom_boite_rendu = filter_input(INPUT_POST, 'nom_rendu', FILTER_SANITIZE_STRING);
        $nb_fichiers = filter_input(INPUT_POST, 'nb_fichiers', FILTER_SANITIZE_NUMBER_INT);
        $extensions = filter_input(INPUT_POST, 'extension', FILTER_SANITIZE_NUMBER_INT);

        if (!empty($_POST['date_rendu']) && !empty($_POST['nom_rendu']) && !empty($_POST['nb_fichiers']) && !empty($_POST['extension'])) {
            // création enregistrement boite de rendu
            $boite_rendu = new Boite_rendu(null, $date_rendu, $nom_boite_rendu, intval($nb_fichiers));
            $success = $this->Boite_Rendu_Model->createBoite_Rendu($boite_rendu);
            // création enregistrement autoriser (lien avec les extensions)
            $autoriser = new Autoriser($this->Boite_Rendu_Model->getAllBoite_Rendu()[0], $this->Extension_Model->getOneExtension(intval($nb
            $success2 = $this->Autoriser_Model->createAutoriser($autoriser);
            // création enregistrement transferer (lien avec l'état de la boite de rendu)
            $transferer = new Transferer($this->Boite_Rendu_Model->getAllBoite_Rendu()[0], $this->Etat_Model->getOneEtat(1)); // récupère la
            $success3 = $this->Transferer_Model->createTransferer($transferer);
            // on ajoute les enregistrements dans les 3 tables et on redirige vers la sélection des groupes pour cette boite de rendu
            if ($success && $success2 && $success3) {
                header('Location: index.php?page=new_boite_rendu2');
            }
        }
    }
    // on récupère toutes les tables nécessaires
    $extensions = $this->Extension_Model->getAllExtension();
    $comptes = $this->Compte_Model->getAllCompte();
    echo $this->twig->render('defaultController/new_boite_rendu.html.twig', ['extensions'=>$extensions, 'comptes'=>$comptes]);
```

On crée des entity pour chaque futur enregistrement et on utilise ces entity créent avec les données du formulaire pour les envoyer dans notre BDD en tant que véritable enregistrements en faisant appel aux méthodes 'create' défini dans les fichiers Model.php de nos tables de données.

Une fois tous les enregistrements ajoutés à notre BDD on redirige l'administrateur vers une deuxième page de configuration de boite de rendu 'new_boite_rendu_2' afin de créer et sélectionner les comptes 'élèves' a qui la boite de rendu s'adresse.

Dans la vue de cette deuxième page de configuration, on crée un formulaire qui permettra de créer un groupe et de sélectionner les comptes élèves pour ce groupe, et un autre

formulaire pour finaliser la création de la boîte de rendu.

On passe toujours dans le tableau du Contrôleur les données utiles, ici les différents comptes et leur classe pour pouvoir les afficher de façon ordonnées sur notre vue.

```php
// deuxième page de création boite de rendu pour attribuer des élèves/groupes d'élèves
public function new_boite_rendu2()
{
    // on récupère toutes les tables dont on a besoin
    $comptes = $this->Compte_Model->getAllCompte();
    $classes = ['SN1', 'SN2', 'BACHELOR 3', 'BACHELOR 4', 'BACHELOR 5'];
    echo $this->twig->render('defaultController/new_boite_rendu2.html.twig', ['comptes'=>$comptes, 'classes'=>$classes]);
}
```

DefaultController.php | Groupe_Model.php | new_boite_rendu2.html.twig | DependencyContainer.php | add_student.html.twig

PTSolutionWeb > templates > defaultController > new_boite_rendu2.html.twig

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

{% block title %} {{ parent() }} Nouvelle boite de rendu {% endblock %}

{% block content %}
    <div>
        <h2 class="text-center"> Créer votre nouvelle boite de rendu </h2>
    </div>
    <!-- formulaire de création de boite de rendu -->
    <form action="" method="post">
        <div class="text_center">
        <!-- On affiche les comptes des élèves triés par classe -->
            {% for classe in classes %}
            <details>
                <summary>{{classe}}</summary>
                <!-- On met une coche pour sélectionner tous les élèves d'une classe -->
                <summary>Tout sélectionner</summary>
                <input type="checkbox" id="all" name="all" value="{{classe}}">
                {% for compte in comptes %}
                    {% if compte.classe == classe %}
                    <div>
                        <input type="checkbox" id="{{compte.id}}" name="{{compte.id}}" value="{{compte.id}}">
                        <summary>{{compte.prenom}} {{compte.nom}}</summary>
                    </div>
                    {% endif %}
                {% endfor %}
            </details>
            {% endfor %}
        </div>
        <div class="text-center">
            <input type="submit" value="Créer le groupe">
        </div>
    </form>
{% endblock %}
```

On voit à présent les différentes classes affichés sous forme d'arborescence avec à l'intérieur les noms des comptes classés en fonction de leur classe.

On prépare une condition pour récupérer les données du formulaire en POST et ensuite pouvoir créer des entity pour stocker de nouveaux enregistrements dans la table Groupe et les tables associés dans notre BDD :

```php
// deuxième page de création boite de rendu pour attribuer des élèves/groupes d'élèves
public function new_boite_rendu2()
{
    // on récupère en post les informations pour créer un nouveau groupe
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $id_boite_rendu = filter_input(INPUT_GET, 'id_boite_rendu', FILTER_SANITIZE_NUMBER_INT); // on récup la boite de rendu
        $nom_groupe = filter_input(INPUT_POST, 'group_name', FILTER_SANITIZE_STRING);
        $comptes = [];
        // on récupère la liste des comptes cochés
        for($i=0; $i<=$this->Compte_Model->getAllCompte()[0]->getId_compte(); $i++){
            if(filter_input(INPUT_POST, 'compte_'.strval($i), FILTER_SANITIZE_STRING) != ''){
                array_push($comptes, filter_input(INPUT_POST, 'compte_'.strval($i), FILTER_SANITIZE_STRING));
            }
        }

        if (!empty($_POST['group_name'])) {
            $groupe = new Groupe(null, $nom_groupe);
            $success = $this->Groupe_Model->createGroupe($groupe);

            $attribuer = new Attribuer($this->Boite_Rendu_Model->getOneBoite_Rendu(intval($id_boite_rendu)), $this->Groupe_Model->getAllGroupe()
            $success2 = $this->Attribuer_Model->createAttribuer($attribuer);

            // récupérer chaque id des comptes élèves : créer un enregistrement dans Rassembler
            foreach($comptes as $compte){
                //on crée un enregistrement dans la table rassembler
                $rassembler = new Rassembler($this->Compte_Model->getOneCompte(intval($compte)), $this->Groupe_Model->getAllGroupe()[0]);
                $success3 = $this->Rassembler_Model->createRassembler($rassembler);
            }

            if($success && $success2 && $success3){
                header('Location: index.php?page=new_boite_rendu2&id_boite_rendu='. $id_boite_rendu);
            }
```

```php
    // on récupère toutes les tables dont on a besoin
    $comptes = $this->Compte_Model->getAllCompte();
    $classes = ['SN1', 'SN2', 'BACHELOR 3', 'BACHELOR 4', 'BACHELOR 5'];
    $id_boite_rendu = filter_input(INPUT_GET, 'id_boite_rendu', FILTER_SANITIZE_NUMBER_INT);
    // on récupère uniquement les groupes pour ce projet
    $attribuer = $this->Attribuer_Model->getAllAttribuer();
    $rassembler = $this->Rassembler_Model->getAllRassembler();
    echo $this->twig->render('defaultController/new_boite_rendu2.html.twig', ['comptes'=>$comptes, 'classes'=>$classes, 'attribuer'=>$attribuer,
}

public function login()
```

```
<div>
    <label for="">Listes des groupes :</label>
    {% for att in attribuer %}
        {% if att.id_boite_rendu.id_boite_rendu == boite_rendu %}
            <details>
                <summary>{{att.id_groupe.nom_groupe}}</summary>
                <summary>
                    {% for ra in rassembler %}
                        {% if ra.id_groupe.id_groupe == att.id_groupe.id_groupe %}
                            {{ra.id_compte.prenom}} {{ra.id_compte.nom}}
                        {% endif %}
                    {% endfor %}

                </summary>
            </details>
        {% endif %}
    {% endfor %}
</div>
```

On affiche le contenu de la table Groupe sur la vue permettant à l'administrateur de voir ses créations de groupe en direct avec le nom du groupe et la liste des élèves choisis.

```
    <form method="post">
        <input type="hidden" id="finish" name="finish" value="yes">
        <input type="submit" value="Terminer">
    </form>
% endblock %}
```

Une fois tous les groupes paramétrés, un deuxième formulaire crée en POST dans le vu permet de cloturer la création de boite de rendu.

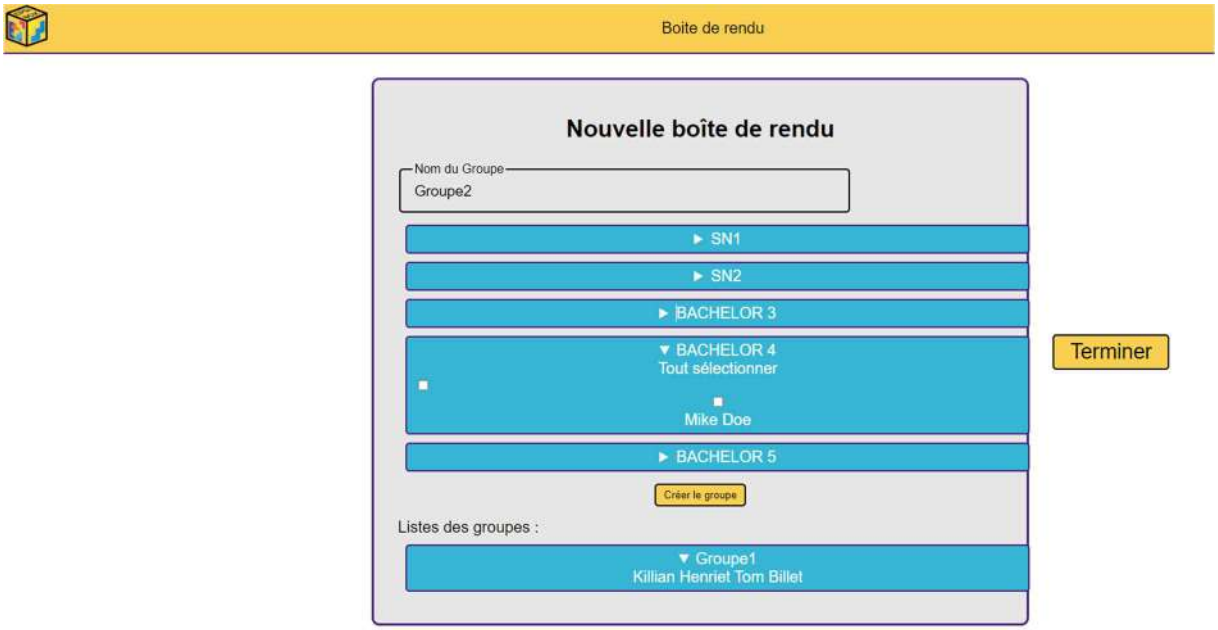
```
    }

    // si le paramétrage de la boite est terminer on redirige vers le panel admin
    if (!empty($_POST['finish'])) {
        header('Location: index.php?page=accueil_a');
    }

}
```

Une fois la requête récupérée en POST on redirige l'administrateur vers la page de rendus accueil_a.html.twig : avec toutes les boites de rendus précédemment créées qui s'affichent ici :

DefaultController.php | boite_rendu.php | new_boite_rendu2.html.twig | accueil_a.html.twig

TSolutionWeb > templates > defaultController > accueil_a.html.twig

```
4   {% block title %} {{ parent() }} Accueil Eleve {% endblock %}
5
6   {% block content %}
7       <div class="container-fluid">
8           <div class="row justify-content-center">
9               <div class="col-12 col-md-10">
10                  <h1 class="text-center"> Pannel Admin </h1>
11              </div>
12          </div>
13      </div>
14
15      <div><h2>Liste des boîtes de rendu</h2></div>
16
17      <div><a href="index.php?page=new_boite_rendu">Ajouter une nouvelle boîte</a></div>
18      <div>
19          {% for boite in boite_rendu %}
20              {{boite.nom_boite_rendu}} {{boite.date_rendu}}
21          {% endfor %}
22      </div>
23  {% endblock %}
```



Mise en place de la suppression des boites de rendus

Pour géré la suppression d'une boite de rendue depuis la page accueil_a.html.twig, on crée au préalable les méthodes nécessaires dans les Model des tables de données utilisée par les boites de rendus tel que Boite_Rendu, Groupe, Rassembler, Autoriser, ou encore Transférer.

```php
    }

    // suppprimer un enregistrement d'une chambre
    public function deleteBoite_Rendu(int $id): bool {
        $sql = "DELETE FROM Boite_Rendu WHERE Id_Boite_Rendu = :id";
        $stmt = $this->db->prepare($sql);
        $stmt->bindValue(':id', $id, PDO::PARAM_INT);
        return $stmt->execute();

    }
```

```php
    // suppprimer un enregistrement d'un groupe
    public function deleteGroupe(int $id): bool {
        $sql = "DELETE FROM Groupe WHERE Id_Groupe = :id";
        $stmt = $this->db->prepare($sql);
        $stmt->bindValue(':id', $id, PDO::PARAM_INT);
        return $stmt->execute();

    }


}
```

On a précisé dans la création de notre base des données des tables en Delete CASCADE pour éviter de devoir supprimer manuellement un enregistrement avec une clé étrangère qui n'existe plus.

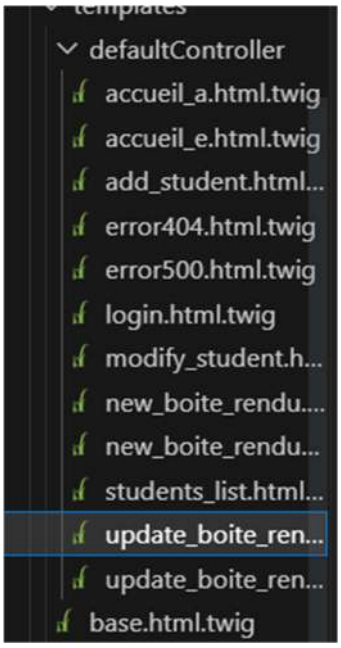
```twig
{% for boite in boite_rendu %}
    <details class="arborescence">
        <summary>{{boite.nom_boite_rendu}} {{boite.date_rendu}} <a href="index.php?page=delete_enregistrement&boite_rendu={{boite.id_boite_rendu}}
    </details>
{% endfor %}
```

On ajoute à coté de chaque boite de rendu un bouton supprimer qui vient récupérer et envoyer l'id de la boite de rendu sélectionné pour ensuite l'a supprimer :

```php
// supprimer un enregistrement d'une table
public function delete_enregistrement()
{
    if ($_SERVER['REQUEST_METHOD'] === 'GET') {
        // on récup la boite
        $id_boite_rendu = filter_input(INPUT_GET, 'boite_rendu', FILTER_SANITIZE_NUMBER_INT);
        // on récup le/les groupe(s) associé et on le suppr
        $groupes = [];
        foreach($this->Attribuer_Model->getAllAttribuer() as $att){
            if($att->getId_Boite_Rendu()->getId_Boite_Rendu() == $id_boite_rendu){
                array_push($groupes, $att->getId_Groupe()->getId_Groupe());
                echo 'test1';
            }
            echo 'test2';
        }
        $success2 = null;
        foreach($groupes as $groupe){
            echo 'test3';
            $success2 = $this->Groupe_Model->deleteGroupe(intval($groupe));
        }
        // on récup la boite de rendu et on l'a suppr
        $success = $this->Boite_Rendu_Model->deleteBoite_Rendu(intval($id_boite_rendu));

        if($success && $success2){
            header('Location: index.php?page=accueil_a');
        }
    }
}
```

On code dans defaultController une méthode deleteEnregistrement qui récupère l'id en GET et execute la méthode deleteBoite_Rendu($id) ou encore deleteGroupe($id) pour supprimer les bons enregistrements dans la BDD.

Mise en place de la modification des boites de rendus



Tout d’abord, on crée deux nouvelles pages dans notre projet copiant le design des pages de création de boite de rendu mais celles-ci destinées à la modification.

```
class="center-page">

class="form-area">

2 class="form-title">Modifier la boite de rendu</h2>

form action="" method="post" class="form">


  <fieldset class="fieldset">
      <legend>nom de la boite</legend>
      <input class="fieldset-input" type="text" id="nom_rendu" name="nom_rendu" placeholder="nom boite" value="{{boite_rendu.nom_boite_rendu}}" requ
  </fieldset>


  <div class="div-input">
      <label for="date_rendu">Date butoir :</label>
      <input class="input" type="date" id="date_rendu" name="date_rendu" value="{{boite_rendu.date_rendu}}" required>
  </div>


  <div class="div-input">
      <label for="nb_fichiers">Nombre de fichiers attendus :</label>
      <input class="input" type="number" id="nb_fichiers" name="nb_fichiers" value={{boite_rendu.nb_fichiers}} min=1 required>
  </div>
```

Pour savoir quelle boite de rendu modifier on ajoute un bouton modifier à coté de la liste des boites de rendus dans accueil_a.html.twig

L’id de la boite de rendu est donc ajouté au lien des pages update_boite_rendu.html.twig, et on le récupère en GET dans le contrôleur.

On récupère dans la base de données grâce aux différentes méthodes getAll… et l’id de notre boite de rendu les rengistrements qui lui sont associés et on vient les passé dans le tableau pour les récupérer dans notre vu afin de les afficher par défaut (screen ci-dessus).

```
public function update_boite_rendu()
{
    if ($_SERVER['REQUEST_METHOD'] === 'GET') {
        // on récup la boite
        $id_boite_rendu = filter_input(INPUT_GET, 'boite_rendu', FILTER_SANITIZE_NUMBER_INT);
        $boite_rendu = $this->Boite_Rendu_Model->getOneBoite_Rendu(intval($id_boite_rendu));
        // on récup les extensions cochées
        $extension_on = [];
        foreach($this->Autoriser_Model->getAllAutoriser() as $autoriser){
            if($autoriser->getId_Boite_Rendu()->getId_Boite_Rendu() == $id_boite_rendu){
                array_push($extension_on, $autoriser->getId_Extension());
            }
        }
        $extensions = $this->Extension_Model->getAllExtension();
        echo $this->twig->render('defaultController/update_boite_rendu.html.twig', ['extension_on'=>$extension_on, 'extensions'=>$extensions, '
```

```php
// on récupère le formulaire remplie et on modif les valeurs
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $id_boite_rendu = filter_input(INPUT_POST, 'id_boite_rendu', FILTER_SANITIZE_NUMBER_INT);
    $date_rendu = filter_input(INPUT_POST, 'date_rendu', FILTER_SANITIZE_STRING);
    $nom_boite_rendu = filter_input(INPUT_POST, 'nom_rendu', FILTER_SANITIZE_STRING);
    $nb_fichiers = filter_input(INPUT_POST, 'nb_fichiers', FILTER_SANITIZE_NUMBER_INT);
    $extensions = [];

    // pour chaque extension de la BDD on essaye de récupérer celles coché
    for($i=0; $i<count($this->Extension_Model->getAllExtension()); $i++){
        if(filter_input(INPUT_POST, 'extension_'.strval($i), FILTER_SANITIZE_STRING) != ''){
            array_push($extensions, filter_input(INPUT_POST, 'extension_'.strval($i), FILTER_SANITIZE_NUMBER_INT));
        }
    }

    if (!empty($_POST['id_boite_rendu']) && !empty($_POST['date_rendu']) && !empty($_POST['nom_rendu']) && !empty($_POST['nb_fichiers']) &&
        // update boite rendu
        $boite_rendu = new Boite_rendu(intval($id_boite_rendu), $date_rendu, $nom_boite_rendu, intval($nb_fichiers));
        $success = $this->Boite_Rendu_Model->updateBoite_Rendu($boite_rendu);
        // on ajoute une extension manquante à la BDD
        foreach($extensions as $ex){
            if($this->Autoriser_Model->getOneAutoriser(intval($id_boite_rendu), intval($ex)) == null){
                $autoriser = new Autoriser($this->Boite_Rendu_Model->getOneBoite_Rendu(intval($id_boite_rendu)), $this->Extension_Model->get
                $this->Autoriser_Model->createAutoriser($autoriser);
            }
        }
        // on suppr les extensions en retirée dans la BDD
        foreach($this->Autoriser_Model->getAllAutoriser() as $enregistrement){
            if($enregistrement->getId_Boite_Rendu()->getId_Boite_Rendu() == $id_boite_rendu){
                if(!in_array($enregistrement->getId_Extension()->getId_Extension(), $extensions)){
                    $this->Autoriser_Model->deleteAutoriser(intval($id_boite_rendu), intval($enregistrement->getId_Extension()->getId_Extens
                }
```

On préapre ensuite une condition comme pour la création en POST qui va récupérer les nouvelles valeurs entrées par l'utilisateur et les stocker dans la base de données en modifiant les bons enregistrements existant, ou en créant de nouveaux et supprimant les obsolètes.

```php
    // on update la boite de rendu et on redirige vers la page d'update suivante
    if ($success) {
        header('Location: index.php?page=update_boite_rendu2&id_boite_rendu=' . strval($id_boite_rendu)); // on passe la boite de rendu
    }
}
```

Une fois les modifications effectuées et le formulaire envoyé, on redirige l'administrateur vers la seconde page de modification, celle des groupes et élèves associés à cette boite de rendu.

Modification des groupes de la boite de rendu

```php
public function update_boite_rendu2()
{
    if ($_SERVER['REQUEST_METHOD'] === 'GET') {
        // on récupère toutes les tables dont on a besoin
        $comptes = $this->Compte_Model->getAllCompte();
        $classes = ['SN1', 'SN2', 'BACHELOR 3', 'BACHELOR 4', 'BACHELOR 5'];
        $id_boite_rendu = filter_input(INPUT_GET, 'id_boite_rendu', FILTER_SANITIZE_NUMBER_INT);
        // on récupère uniquement les groupes pour ce projet
        $attribuer = $this->Attribuer_Model->getAllAttribuer();
        $rassembler = $this->Rassembler_Model->getAllRassembler();
        // on récup le groupe à modifier
        $groupe = null;
        if (!empty($_GET['grp'])) {
            $groupe = $this->Groupe_Model->getOneGroupe(intval(filter_input(INPUT_GET, 'grp', FILTER_SANITIZE_STRING)));
        }
        echo $this->twig->render('defaultController/update_boite_rendu2.html.twig', ['comptes'=>$comptes, 'classes'=>$classes, 'attribuer'=>$att
    }
```

```
<label for="">Listes des groupes :</label>
{% for att in attribuer %}
    {% if att.id_boite_rendu.id_boite_rendu == boite_rendu %}
        <details class="arborescence">
            <summary>{{att.id_groupe.nom_groupe}}
            <a href="index.php?page=update_boite_rendu2&id_boite_rendu={{boite_rendu}}&grp={{att.id_groupe.id_groupe}}">modifier</a>
            </summary>
            <summary>
                {% for ra in rassembler %}
                    {% if ra.id_groupe.id_groupe == att.id_groupe.id_groupe %}
                        {{ra.id_compte.prenom}} {{ra.id_compte.nom}}
                    {% endif %}
                {% endfor %}

            </summary>
        </details>
```

Même cheminement pour ma deuxième page de modification, on récupère en GET l'url de notre boite de rendu à modifier, on affiche les données qui lui associés dans la vue.

On ajoute un bouton de modification à la liste des groupes de cette boites de rendus.

```
'groupe'=>$groupe];
```

Etant donné qu'on a récupéré au préalable depuis le contrôleur les groupes associés à cette boite de rendu, on peut facilement passé en GET le groupe sur lequel l'administrateur à cliquer sur 'modifier'.

```
    <div class="submit-btn-area">
        <input type='hidden' id='id_groupe' name='id_groupe' value='{{groupe.id_groupe}}'>
        {% if groupe != null %}
        <input class="small-submit-btn" type="submit" value="Modifier le groupe">
        {% else %}
        <input class="small-submit-btn" type="submit" value="Créer le groupe">
        {% endif %}
```

On prévoit une modification pour changer le texte du bouton en fonction de si l'utilisateur est en train de modifier un groupe existant à la suite d'un clique sur 'modifier' ou s'il est en train de créer un tout nouveau groupe.

```
// on récupère en post les informations pour créer un nouveau groupe
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $id_boite_rendu = filter_input(INPUT_GET, 'id_boite_rendu', FILTER_SANITIZE_NUMBER_INT); // on récup la boite de rendu
    $id_groupe = $nom_groupe = filter_input(INPUT_POST, 'id_groupe', FILTER_SANITIZE_STRING);
    $nom_groupe = filter_input(INPUT_POST, 'group_name', FILTER_SANITIZE_STRING);
    $comptes = [];

    // on récupère la liste des comptes cochés
    for($i=0; $i<=$this->Compte_Model->getAllCompte()[0]->getId_compte(); $i++){
        if(filter_input(INPUT_POST, 'compte_'.strval($i), FILTER_SANITIZE_STRING) != ''){
            array_push($comptes, filter_input(INPUT_POST, 'compte_'.strval($i), FILTER_SANITIZE_STRING));
        }
    }
```

Dans le contrôleur on récupère en post les différentes données du formulaire notamment les noms d'élèves cochés, afin de les affecter dans un groupe appelé $group_name passé en POST..

```php
if (!empty($_POST['group_name'])) {
    // on récupère l'id du groupe s'il est dans l'url (modification d'un groupe existant), sinon id_groupe = null
    $groupe = new Groupe(intval($id_groupe), $nom_groupe);
    if($groupe->getId_Groupe() != null){

        $success = $this->Groupe_Model->updateGroupe($groupe);
        // récupérer chaque id des comptes élèves : modifie un enregistrement dans Rassembler
        foreach($comptes as $compte){
            //on crée un enregistrement dans la table rassembler s'il n'existe pas déjà
            $rassembler = new Rassembler($this->Compte_Model->getOneCompte(intval($compte)), $this->Groupe_Model->getOneGroupe(intval($g
            if($this->Rassembler_Model->getOneRassembler($rassembler->getId_Compte()->getId_Compte(), $rassembler->getId_Groupe()->getId
                $this->Rassembler_Model->createRassembler($rassembler);
            }
        }
        foreach($this->Rassembler_Model->getAllRassembler() as $enregistrement){
            if($enregistrement->getId_Groupe() == $groupe){
                if(!in_array($enregistrement->getId_Compte()->getId_Compte(), $comptes)){
                    $this->Rassembler_Model->deleteRassemblement($enregistrement->getId_Compte()->getId_Compte(), $enregistrement->getId
                }
            }
        }
```

On effectue les modifications dans les différentes table de données si un nouveau groupe ou la modification d'un groupe existant survient.

```php
    } else{
        $success = $this->Groupe_Model->createGroupe($groupe);
        $attribuer = new Attribuer($this->Boite_Rendu_Model->getOneBoite_Rendu(intval($id_boite_rendu)), $this->Groupe_Model->getAllGrou
        $this->Attribuer_Model->createAttribuer($attribuer);

        // récupérer chaque id des comptes élèves : créer un enregistrement dans Rassembler
        foreach($comptes as $compte){
            //on crée un enregistrement dans la table rassembler
            $rassembler = new Rassembler($this->Compte_Model->getOneCompte(intval($compte)), $this->Groupe_Model->getAllGroupe()[0]);
            $this->Rassembler_Model->createRassembler($rassembler);
        }
    }


    if($success){
        header('Location: index.php?page=update_boite_rendu2&id_boite_rendu='. $id_boite_rendu);
    }
}
// si le paramétrage de la boite est terminer on redirige vers le panel admin
if (!empty($_POST['finish'])) {
    header('Location: index.php?page=accueil_a');
}
```

Pour finir, une fois un nouveau groupe crée ou modifié on redirige l'administrateur vers la même page pour lui permettre de continuer ses modifications, ses créations de groupes etc..

Enfin si l'administrateur clique sur 'TERMINER ' on le redirige vers la page accueil_a.html.twig.