

# Documentation Festival

Utilisateur

# Sommaire

- I. Page d'accueil
- II. Liste des établissements
  - a. Voir un établissement
  - b. Modifier un établissement
  - c. Créer un établissement
  - d. Supprimer un établissement
- III. Liste des équipes
  - a. Voir une équipe
  - b. Modifier une équipe
  - c. Créer une équipe
  - d. Supprimer une équipe
- IV. Listes des attributions
  - a. Voir une attribution
  - b. Modifier une attribution
  - c. Créer une attribution
  - d. Supprimer une attribution

# I. Page d'accueil



Cette page est la page d'accueil. En haut à gauche (voir flèche rouge), vous pouvez voir le menu. Ce dernier vous permet d'accéder à la liste des établissements, liste des équipes et liste des attributions.

## II. Liste des établissements

The screenshot shows a web application interface for managing establishments. At the top, a green navigation bar contains links: Home, Liste des établissements, Liste des équipes, and Liste des attributions. Below the navigation bar, the main content area features a yellow triangular warning icon with three red lightning bolts, labeled 'Maison des Ligues' and 'Hébergement des participants'. To the left of this, a red arrow points down to a link labeled 'Création d'un nouveau établissement'. Below the main heading, there is a table with a green header row. The table has one visible row with the name 'Charles de Foucauld'. To the right of the table, there are three icons: an eye (view), a pencil (edit), and a trash can (delete). A green arrow points down to the edit icon, a blue arrow points up to the view icon, and a black arrow points left to the delete icon. At the bottom of the page, a green footer bar contains the text 'BTS SIO2 Inès MAGANGA, Gianni BOSIO'.

Home Liste des établissements Liste des équipes Liste des attributions

Maison des Ligues  
Hébergement des participants

Création d'un nouveau établissement

Nom établissement
Charles de Foucauld

View Edit Delete

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Cette page est la page index de la liste des établissements. A partir de cette dernière, vous pouvez voir (flèche bleue), modifier (flèche verte), créer (flèche rouge) ou supprimer (flèche noire) un établissement.

## a. Voir un établissement

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Liges**  
**Hébergement des participants**

Informations sur l'établissement


**Nom:** Charles de Foucauld  
**Adresse:** 9 rue des Roses  
**Code postal:** 75018  
**Ville:** Paris  
**Téléphone:** 0146077259  
**Courriel:** cdf@gmail.com  
**Nom du responsable:** RICHARD  
**Prénom du responsable:** Jean  
**Nombre de chambres:** 50

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour voir. Elle vous permet d'avoir des informations sur l'établissement souhaité.

## b. Modifier un établissement

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

Mettez à jour l'établissement

**Nom:**

**Adresse:**

**Code postal:**

**Ville:**

**Téléphone:**

**Courriel:**

**Nom du responsable:**

**Prénom du responsable:**


**Nombre de chambres:**

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour modifier. Elle vous permet, à partir d'un formulaire, de modifier les informations de l'établissement souhaité.

## c. Créer un établissement

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

Entrez les informations du nouvel établissement

**Nom:**

**Adresse:**

**Code postal:**

**Ville:**

**Téléphone:**

**Courriel:**

**Nom du responsable:**

**Prénom du responsable:**


**Nombre de chambres:**

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le lien de création. Elle vous permet, à partir d'un formulaire de créer un nouvel établissement.








## d. Supprimer un établissement

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

[Création d'un nouveau établissement](#)

Nom établissement	Action
Charles Péguy	   
Charles de Foucauld	  

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Pour supprimer un établissement, il suffit de cliquer sur le bouton pour supprimer (flèche noire).



### III. Liste des équipes

The screenshot shows a web application interface for managing teams. At the top is a green navigation bar with links: Home, Liste des établissements, Liste des équipes, and Liste des attributions. Below the navigation bar, on the left, is a red arrow pointing down to a link labeled 'Création d'une nouvelle équipe'. In the center, there is a yellow triangular warning icon with three red exclamation marks, followed by the text 'Maison des Liges' and 'Hébergement des participants'. Below this is a table with a green header row labeled 'Nom équipe'. The first row of the table contains the text 'Fire Cows'. To the right of the table, there are three icons: an eye (view), a pencil (edit), and a trash can (delete). A green arrow points down to the pencil icon, a blue arrow points up to the eye icon, and a black arrow points left to the trash can icon. At the bottom of the page is a green footer bar with the text 'BTS SIO2 Inès MAGANGA, Gianni BOSIO'.

Cette page est la page index de la liste des équipes. A partir de cette dernière, vous pouvez voir (flèche bleue), modifier (flèche verte), créer (flèche rouge) ou supprimer (flèche noire) une équipe.

## a. Voir une équipe

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

Informations sur l'équipe


**Nom:** Fire Cows  
**Responsable:** Selana ROLANA  
**Nombre de membres:** 25  
**Pays:** France

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour voir. Elle vous permet d'avoir des informations sur l'équipe souhaitée.

## b. Modifier une équipe

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Liges**  
**Hébergement des participants**

Mettez à jour l'équipe

**Nom:**

**Responsable:**

**Nombre de membres:**

**Pays:**

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour modifier. Elle vous permet, à partir d'un formulaire, de modifier les informations de l'équipe souhaitée.

## c. Créer une équipe

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Liges**  
**Hébergement des participants**

Entrez les informations de la nouvelle équipe

**Nom:**

**Responsable:**

**Nombre de membres:**


**Pays:**

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le lien de création. Elle vous permet, à partir d'un formulaire de créer une nouvelle équipe.

## d. Supprimer une équipe








[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



Maison des Ligues

Hébergement des participants

[Création d'une nouvelle équipe](#)

Nom équipe	Action
Cold Pulpe	   
Fire Cows	  

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Pour supprimer une équipe, il suffit de cliquer sur le bouton pour supprimer (flèche noire).

## IV. Liste des attributions

The screenshot shows a web application interface for managing assignments. At the top is a green navigation bar with links: Home, Liste des établissements, Liste des équipes, and Liste des attributions. Below the navigation bar, on the left, is a red arrow pointing down to a link labeled 'Création d'une nouvelle attribution'. In the center, there is a yellow triangular warning icon with three red exclamation marks, followed by the text 'Maison des Ligues' and 'Hébergement des participants'. Below this is a table with a green header row containing 'Attribution n°'. The first row of the table has the number '1'. To the right of the table, there are three icons: an eye (view), a pencil (edit), and a trash can (delete). A green arrow points down to the pencil icon, a blue arrow points up to the eye icon, and a black arrow points left to the trash can icon. At the bottom of the page is a green footer bar with the text 'BTS SIO2 Inès MAGANGA, Gianni BOSIO'.

Home Liste des établissements Liste des équipes Liste des attributions

Création d'une nouvelle attribution

Maison des Ligues  
Hébergement des participants

Attribution n°
1

View Edit Delete

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Cette page est la page index de la liste des attributions. A partir de cette dernière, vous pouvez voir (flèche bleue), modifier (flèche verte), créer (flèche rouge) ou supprimer (flèche noire) une attribution.

## a. Voir une attribution

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

Informations sur l'attribution n°1


**Nom de l'établissement:** Charles de Foucauld  
**Nom de l'équipe:** Fire Cows

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour voir. Elle vous permet d'avoir des informations sur l'attribution souhaitée.

## b.Modifier une attribution

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Ligues**  
**Hébergement des participants**

Mettez à jour l'attribution

Etablissement:

Charles de Foucauld ▾

Equipe:

Fire Cows ▾

Envoyer

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le bouton pour modifier. Elle vous permet, à partir d'un formulaire, de modifier les informations de l'attribution souhaitée.



## c. Créer une attribution

[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



**Maison des Liges**  
**Hébergement des participants**

Entrez les informations de la nouvelle attribution

**Etablissement:**


**Equipe:**

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Vous avez accès à cette page une fois que vous avez cliqué sur le lien de création. Elle vous permet, à partir d'un formulaire de créer une nouvelle attribution.

## d. Supprimer une attribution








[Home](#) [Liste des établissements](#) [Liste des équipes](#) [Liste des attributions](#)



Maison des Liges

Hébergement des participants

[Création d'une nouvelle attribution](#)

Attribution n°	Action
2	   
1	  

BTS SIO2 Inès MAGANGA, Gianni BOSIO

Pour supprimer une attribution, il suffit de cliquer sur le bouton pour supprimer (flèche noire).

# Documentation Festival

Développeur

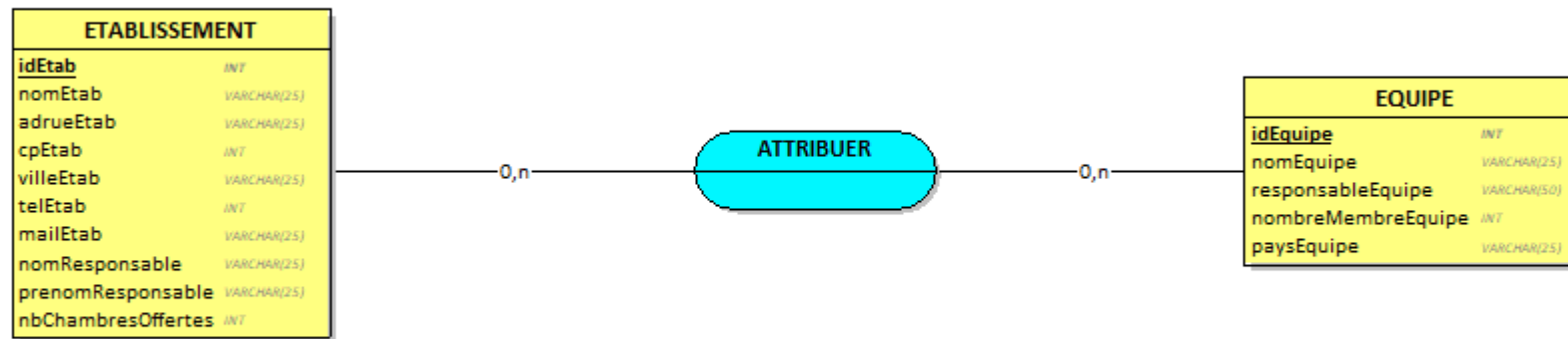
# Sommaire

- I. Base de données
  - a. MCD
  - b. Migrations
  - c. Seeders
- II. Gérer la partie établissement
  - a. Model Etablissement
  - b. Controller Etablissement
  - c. Page index
  - d. Page create
  - e. Page edit
  - f. Page show
- III. Gérer la partie équipe
  - a. Model Equipe
  - b. Controller Equipe
  - c. Page index
  - d. Page create
  - e. Page edit
  - f. Page show
- IV. Gérer la partie attribution
  - a. Model Attribution
  - b. Controller Attribution
  - c. Page index
  - d. Page create
  - e. Page edit

f. Page show

# I. Base de données

## a. MCD



## b. Migrations

Les migrations permettent de créer et de mettre à jour une base de données. Tout ce qui concerne la création de la base de données est pris en charge par la migration.

### 1. Table Etablissement

Pour créer la migration pour la table Etablissement, il faut utiliser cette commande dans le terminal : `php artisan make:migration create_etablislements_tbl --create=etablislements`.

Puis, dans le dossier database/migrations, il suffit de remplir le nouveau fichier comme ceci :

```

1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  class CreateEtablissementsTable extends Migration
8  {
9      /**
10       * Run the migrations.
11       *
12       * @return void
13       */
14     public function up()
15     {
16         Schema::create('etablissements', function (Blueprint $table) {
17             $table->bigIncrements('id');
18             $table->string('nomEtab',25);
19             $table->string('ad rueEtab',25);
20             $table->string('cpEtab',5);
21             $table->string('villeEtab',25);
22             $table->string('telEtab',10);
23             $table->string('mailEtab',25);
24             $table->string('nomResponsable',25);
25             $table->string('prenomResponsable',25);
26             $table->string('nbChambresOffertes');
27             $table->timestamps();
28         });
29     }
30
31     /**
32      * Reverse the migrations.
33      *
34      * @return void
35      */
36     public function down()
37     {
38         Schema::dropIfExists('etablissements');
39     }
40 }

```

Pour valider la création de la table, il suffit de taper dans le terminal la commande suivante : `php artisan migrate`.

## 2. Table Equipe

Pour créer la migration pour la table Equipe, il faut utiliser cette commande dans le terminal : `php artisan make:migration create_equipes_tbl --create=equipe`.

Puis, dans le dossier `database/migrations`, il suffit de remplir le nouveau fichier comme ceci :



```

1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  class CreateEquipesTable extends Migration
8  {
9      /**
10       * Run the migrations.
11       *
12       * @return void
13       */
14     public function up()
15     {
16         Schema::create('equipes', function (Blueprint $table) {
17             $table->bigIncrements('id');
18             $table->string('nomEquipe',25);
19             $table->string('responsableEquipe',50);
20             $table->string('nombreMembreEquipe');
21             $table->string('paysEquipe',25);
22             $table->timestamps();
23         });
24     }
25
26     /**
27      * Reverse the migrations.
28      *
29      * @return void
30      */
31     public function down()
32     {
33         Schema::dropIfExists('equipes');
34     }
35 }

```

Pour valider la création de la table, il suffit de taper dans le terminal la commande suivante : `php artisan migrate`.

### 3. Table Attribution

Pour créer la migration pour la table Attribution, il faut utiliser cette commande dans le terminal : `php artisan make:migration create_attributions_tbl --create=attribution`.

Puis, dans le dossier `database/migrations`, il suffit de remplir le nouveau fichier comme ceci :

```

1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  class CreateAttributionsTable extends Migration
8  {
9      /**
10       * Run the migrations.
11       *
12       * @return void
13       */
14     public function up()
15     {
16         Schema::create('attributions', function (Blueprint $table) {
17             $table->bigIncrements('id');
18             $table->foreignId('etablissements_idEtab');
19             $table->foreignId('equipes_idEquipe');
20             $table->timestamps();
21         });
22     }
23
24     /**
25      * Reverse the migrations.
26      *
27      * @return void
28      */
29     public function down()
30     {
31         Schema::dropIfExists('attributions');
32     }
33 }

```

Pour valider la création de la table, il suffit de taper dans le terminal la commande suivante : `php artisan migrate`.

## c. Seeders

Afin de remplir les tables de notre base de données, nous allons utiliser des Seeders.

### 1. Seeder Etablissement

Pour remplir la table Etablissement, nous allons, dans le terminal, saisir la commande : `php artisan make :seeder EtablissementSeeder`

Puis dans le dossier database/seeder, il suffit de remplir le nouveau fichier comme ceci :

```

1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Models\Etablissement;
7
8  class EtablissementSeeder extends Seeder
9  {
10     /**
11      * Run the database seeds.
12      *
13      * @return void
14      */
15     public function run()
16     {
17         Etablissement::create([
18             'nomEtab'=>'Charles de Foucauld',
19             'adrueEtab'=>'9 rue des Roses',
20             'cpEtab'=>'75018',
21             'villeEtab'=>'Paris',
22             'telEtab'=>'0146077259',
23             'mailEtab'=>'cdf@gmail.com',
24             'nomResponsable'=>'RICHARD',
25             'prenomResponsable'=>'Jean',
26             'nbChambresOffertes'=>'50',
27         ]);
28     }
29 }

```

## 2. Seeder Equipe

Pour remplir la table Etablissement, nous allons, dans le terminal, saisir la commande : php artisan make :seeder EquipeSeeder

Puis dans le dossier database/seeder, il suffit de remplir le nouveau fichier comme ceci :

```
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Models\Equipe;
7
8  class EquipeSeeder extends Seeder
9  {
10     /**
11      * Run the database seeds.
12      *
13      * @return void
14      */
15     public function run()
16     {
17         Equipe::create([
18             'nomEquipe'=>'Fire Cows',
19             'responsableEquipe'=>'Selana ROLANA',
20             'nombreMembreEquipe'=>'25',
21             'paysEquipe'=>'France',
22         ]);
23     }
24 }
```

### 3. Seeder Attribution

Pour remplir la table Etablissement, nous allons, dans le terminal, saisir la commande : `php artisan make :seeder AttributionSeeder`

Puis dans le dossier `database/seeder`s, il suffit de remplir le nouveau fichier comme ceci :

```
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Models\Attribution;
7
8  class AttributionSeeder extends Seeder
9  {
10     /**
11      * Run the database seeds.
12      *
13      * @return void
14      */
15     public function run()
16     {
17         Attribution::create([
18             'etablisements_idEtab'=>'1',
19             'equipes_idEquipe'=>'1',
20         ]);
21     }
22 }
```

#### 4. Remplir la base de données

Une fois que nous avons créé les seeders, nous devons remplir le fichier DatabaseSeeder comme ceci :

```
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6
7  class DatabaseSeeder extends Seeder
8  {
9      /**
10       * Seed the application's database.
11       *
12       * @return void
13       */
14     public function run()
15     {
16         // \App\Models\User::factory(10)->create();
17         $this->call(EtablissementSeeder::class);
18         $this->call(EquipeSeeder::class);
19         $this->call(AttributionSeeder::class);
20     }
21
22 }
```

Après, il suffit, dans le terminal, d'entrer la commande : `php artisan db :seed`.



## II. Gérer la partie établissement

Tout d'abord, pour créer le Controller et Model, nous devons saisir cette commande dans le terminal : `php artisan make:controller EtablissementController --resource --model=Etablissement`

### a. Model Etablissement

Le Model va nous servir à déterminer les colonnes de la table que nous pouvons manipuler. Nous allons remplir le fichier pour Etablissement comme ceci :

```
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Etablissement extends Model
9  {
10     use HasFactory;
11     protected $fillable = ['nomEtab', 'adrueEtab', 'cpEtab', 'villeEtab', 'telEtab', 'mailEtab', 'nomResponsable', 'prenomResponsable', 'nbChambresOffertes'];
12     public function attributions()
13     {
14         return $this->hasMany(Attribution::class);
15     }
16
17 }
```

## b. Controller Etablissement

La tâche d'un contrôleur est de réceptionner une requête (qui a déjà été sélectionnée par une route) et de définir la réponse appropriée. Nous allons remplir le Controller comme ceci :

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Etablissement;
6 use Illuminate\Http\Request;
7
8 class EtablissementController extends Controller
9 {
10     /**
11      * Display a listing of the resource.
12      *
13      * @return \Illuminate\Http\Response
14      */
15     public function index()
16     {
17         $data = Etablissement::latest()->paginate(5);
18
19         return view('etablissement.index', compact('data'))
20             ->with('1', (request()->input('page', 1) - 1) * 5);
21     }
22
23     /**
24      * Show the form for creating a new resource.
25      *
26      * @return \Illuminate\Http\Response
27      */
28     public function create()
29     {
30         return view('etablissement.create');
31     }
32
33     /**
34      * Store a newly created resource in storage.
35      *
36      * @param \Illuminate\Http\Request $request
37      * @return \Illuminate\Http\Response
38      */
39     public function store(Request $request)
40     {
41         $request->validate([
42             'nomEtab' => 'required',
43             'adresseEtab' => 'required',
44             'cpEtab' => 'required',
45             'villeEtab' => 'required',
46             'telEtab' => 'required',
47             'mailEtab' => 'required',
48             'nomResponsable' => 'required',
49             'prenomResponsable' => 'required',
50             'nbChambresOffertes' => 'required',
51         ]);
52
53         Etablissement::create($request->all());
54
55         return redirect()->route('etablissement.index')
56             ->with('success', 'Etablissement created successfully.');
```

```
61
62     /**
63      * Show the form for editing the specified resource.
64      *
65      * @param App\Models\Etablissement $etablissement
66      * @return \Illuminate\Http\Response
67      */
68     public function edit(Etablissement $etablissement)
69     {
70         return view('etablissement.edit', compact('etablissement'));
71     }
72
73     /**
74      * Update the specified resource in storage.
75      *
76      * @param \Illuminate\Http\Request $request
77      * @param App\Models\Etablissement $etablissement
78      * @return \Illuminate\Http\Response
79      */
80     public function update(Request $request, Etablissement $etablissement)
81     {
82         $request->validate([
83             'nomEtab' => 'required',
84             'adresseEtab' => 'required',
85             'cpEtab' => 'required',
86             'villeEtab' => 'required',
87             'telEtab' => 'required',
88             'mailEtab' => 'required',
89             'nomResponsable' => 'required',
90             'prenomResponsable' => 'required',
91             'nbChambresOffertes' => 'required',
92         ]);
93
94         $etablissement->update($request->all());
95
96         return redirect()->route('etablissement.index')
97             ->with('success', 'Etablissement updated successfully');
98     }
99
100     /**
101      * Remove the specified resource from storage.
102      *
103      * @param App\Models\Etablissement $etablissement
104      * @return \Illuminate\Http\Response
105      */
106     public function destroy(Etablissement $etablissement)
107     {
108         $etablissement->delete();
109
110         return redirect()->route('etablissement.index')
111             ->with('success', 'Etablissement deleted successfully');
112     }
113 }
```

## c. Page index

```
1 @include('include.debut')
2 <div>
3     <a href="{{ route('etablissement.create') }}">Création d'un nouveau établissement</a>
4 </div>
5
6 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
7     <thead>
8         <th>Nom établissement</th>
9         <th width="280px">Action</th>
10    </thead>
11    @foreach ($data as $key => $value)
12    <tr>
13        <td>{{ $value->nomEtab }}</td>
14        <td>
15            <form action="{{ route('etablissement.destroy',$value->id) }}" method="POST">
16                <a href="{{ route('etablissement.show',$value->id) }}"></a>
17                <a href="{{ route('etablissement.edit',$value->id) }}"></a>
18                @csrf
19                @method('DELETE')
20                <button type="submit" class="btn btn-danger"></button>
21            </form>
22        </td>
23    </tr>
24    @endforeach
25 </table>
26 {!! $data->links() !!}
27
28
29 @include('include.fin')
```

A partir de cette page, nous cherchons à afficher toutes les données de la table Etablissement et, à partir d'un formulaire, afficher les boutons pour créer, montrer, modifier ou supprimer un établissement.

## d. Page create

```
1  @include('include.debut')
2
3  @if ($errors->any())
4      <div>
5          <strong>ALERTE!</strong> Erreur lors de la création.<br><br>
6          <ul>
7              @foreach ($errors->all() as $error)
8                  <li>{{ $error }}</li>
9              @endforeach
10         </ul>
11     </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Entrez les informations du nouvel établissement</th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('etablissement.store')}}" method="post">
24                     @csrf
25                     <strong>Nom:</strong>
26                     <input type="text" name="nomEtab" placeholder="Nom"><br>
27                     <strong>Adresse:</strong>
28                     <input type="text" name="adrueEtab" placeholder="Adresse"><br>
29                     <strong>Code postal:</strong>
30                     <input type="tel" name="cpEtab" placeholder="Code postal"><br>
31                     <strong>Ville:</strong>
32                     <input type="text" name="villeEtab" placeholder="Ville"><br>
33                     <strong>Téléphone:</strong>
34                     <input type="tel" name="telEtab" placeholder="Téléphone"><br>
35                     <strong>Courriel:</strong>
36                     <input type="text" name="mailEtab" placeholder="Courriel"><br>
37                     <strong>Nom du responsable:</strong>
38                     <input type="text" name="nomResponsable" placeholder="Nom du responsable"><br>
39                     <strong>Prénom du responsable:</strong>
40                     <input type="text" name="prenomResponsable" placeholder="Prenom du responsable"><br>
41                     <strong>Nombre de chambres:</strong>
42                     <input type="number" name="nbChambresOffertes" placeholder="Nombre de chambres"><br>
43                     <br>
44                     <button type="submit">Envoyer</button>
45                 </form>
46             </td>
47         </tr>
48     </tbody>
49 </table>
50
51 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de créer un nouvel établissement.

## e. Page edit

```
1 @include('include.debut')
2
3 @if ($errors->any())
4     <div>
5         <strong>ALERTE!</strong> Erreur lors de la mise à jour.<br><br>
6         <ul>
7             @foreach ($errors->all() as $error)
8                 <li>{{ $error }}</li>
9             @endforeach
10        </ul>
11    </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Mettez à jour l'établissement</th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('etablissement.update',$etablissement->id)}}" method="post">
24                     @csrf
25                     @method('PUT')
26                     <strong>Nom:</strong>
27                     <input type="text" name="nomEtab" placeholder="Nom" value="{{ $etablissement->nomEtab }}"><br>
28                     <strong>Adresse:</strong>
29                     <input type="text" name="adueEtab" placeholder="Adresse" value="{{ $etablissement->adueEtab }}"><br>
30                     <strong>Code postal:</strong>
31                     <input type="tel" name="cpEtab" placeholder="Code postal" value="{{ $etablissement->cpEtab }}"><br>
32                     <strong>Ville:</strong>
33                     <input type="text" name="villeEtab" placeholder="Ville" value="{{ $etablissement->villeEtab }}"><br>
34                     <strong>Téléphone:</strong>
35                     <input type="tel" name="telEtab" placeholder="Téléphone" value="{{ $etablissement->telEtab }}"><br>
36                     <strong>Courriel:</strong>
37                     <input type="text" name="mailEtab" placeholder="Courriel" value="{{ $etablissement->mailEtab }}"><br>
38                     <strong>Nom du responsable:</strong>
39                     <input type="text" name="nomResponsable" placeholder="Nom du responsable" value="{{ $etablissement->nomResponsable }}"><br>
40                     <strong>Prénom du responsable:</strong>
41                     <input type="text" name="prenomResponsable" placeholder="Prénom du responsable" value="{{ $etablissement->prenomResponsable }}"><br>
42                     <strong>Nombre de chambres:</strong>
43                     <input type="number" name="nbChambresOffertes" placeholder="Nombre de chambres" value="{{ $etablissement->nbChambresOffertes }}"><br>
44                     <button type="submit">Envoyer</button>
45                 </form>
46             </td>
47         </tr>
48     </tbody>
49 </table>
50 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de modifier un établissement.

## f. Page show

```
1  @include('include.debut')
2
3  <table class='styled-table' width='60%' cellspacing='0' cellpadding='0' align='center'>
4    <thead>
5      <tr>
6        <th>Informations sur l'établissement</th>
7      </tr>
8    </thead>
9    <tbody>
10     <tr>
11       <td>
12         <strong>Nom:</strong> {{ $etablissement->nomEtab }}
13         <br>
14         <strong>Adresse:</strong> {{ $etablissement->adrueEtab }}
15         <br>
16         <strong>Code postal:</strong> {{ $etablissement->cpEtab }}
17         <br>
18         <strong>Ville:</strong> {{ $etablissement->villeEtab }}
19         <br>
20         <strong>Téléphone:</strong> {{ $etablissement->telEtab }}
21         <br>
22         <strong>Courriel:</strong> {{ $etablissement->mailEtab }}
23         <br>
24         <strong>Nom du responsable:</strong> {{ $etablissement->nomResponsable }}
25         <br>
26         <strong>Prénom du responsable:</strong> {{ $etablissement->prenomResponsable }}
27         <br>
28         <strong>Nombre de chambres:</strong> {{ $etablissement->nbChambresOffertes }}
29       </td>
30     </tr>
31   </tbody>
32 </table>
33 @include('include.fin')
```

Ici, nous affichons les différentes informations sur l'établissement choisi.

### III. Gérer la partie équipe

Tout d'abord, pour créer le Controller et Model, nous devons saisir cette commande dans le terminal : `php artisan make:controller EquipeController --resource --model=Equipe`

#### a. Model Equipe

Le Model va nous servir à déterminer les colonnes de la table que nous pouvons manipuler. Nous allons remplir le fichier pour Equipe comme ceci :

```
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Equipe extends Model
9  {
10     use HasFactory;
11     protected $fillable = ['nomEquipe', 'responsableEquipe', 'nombreMembreEquipe', 'paysEquipe'];
12
13     public function attributions()
14     {
15         return $this->hasMany(Attribution::class);
16     }
17 }
```

## b. Controller Equipe

La tâche d'un contrôleur est de réceptionner une requête (qui a déjà été sélectionnée par une route) et de définir la réponse appropriée. Nous allons remplir le Controller comme ceci :

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Equipe;
6 use Illuminate\Http\Request;
7
8 class EquipeController extends Controller
9 {
10     /**
11      * Display a listing of the resource.
12      *
13      * @return \Illuminate\Http\Response
14      */
15     public function index()
16     {
17         $data = Equipe::latest()->paginate(5);
18
19         return view('equipe.index',compact('data'))
20             ->with('1', (request()->input('page', 1) - 1) * 5);
21     }
22
23     /**
24      * Show the form for creating a new resource.
25      *
26      * @return \Illuminate\Http\Response
27      */
28     public function create()
29     {
30         return view('equipe.create');
31     }
32
33     /**
34      * Store a newly created resource in storage.
35      *
36      * @param \Illuminate\Http\Request $request
37      * @return \Illuminate\Http\Response
38      */
39     public function store(Request $request)
40     {
41         $request->validate([
42             'nomEquipe' => 'required',
43             'responsableEquipe' => 'required',
44             'nombreMembresEquipe' => 'required',
45             'payEquipe' => 'required',
46         ]);
47
48         Equipe::create($request->all());
49
50         return redirect()->route('equipe.index')
51             ->with('success','Equipe created successfully.');
```

```
52     }
53
54     /**
55      * Display the specified resource.
56      *
57      * @param \App\Models\Equipe $equipe
58      * @return \Illuminate\Http\Response
59      */
60     public function show(Equipe $equipe)
61     {
62         //
63     }
64
65     /**
66      * Show the form for editing the specified resource.
67      *
68      * @param \App\Models\Equipe $equipe
69      * @return \Illuminate\Http\Response
70      */
71     public function edit(Equipe $equipe)
72     {
73         //
74     }
75
76     /**
77      * Update the specified resource in storage.
78      *
79      * @param \App\Models\Equipe $equipe
80      * @param \Illuminate\Http\Request $request
81      * @return \Illuminate\Http\Response
82      */
83     public function update(Equipe $equipe, Request $request)
84     {
85         //
86     }
87
88     /**
89      * Remove the specified resource from storage.
90      *
91      * @param \App\Models\Equipe $equipe
92      * @return \Illuminate\Http\Response
93      */
94     public function destroy(Equipe $equipe)
95     {
96         //
97     }
98 }
```



## c. Page index

```
1 @include('include.debut')
2 <div>
3     <a href="{{ route('equipe.create') }}">Création d'une nouvelle équipe</a>
4 </div>
5
6 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
7     <thead>
8         <th>Nom équipe</th>
9         <th width="280px">Action</th>
10    </thead>
11    @foreach ($data as $key => $value)
12    <tr>
13        <td>{{ $value->nomEquipe }}</td>
14        <td>
15            <form action="{{ route('equipe.destroy',$value->id) }}" method="POST">
16                <a href="{{ route('equipe.show',$value->id) }}"></a>
17                <a href="{{ route('equipe.edit',$value->id) }}"></a>
18                @csrf
19                @method('DELETE')
20                <button type="submit" class="btn btn-danger"></button>
21            </form>
22        </td>
23    </tr>
24    @endforeach
25 </table>
26 {!! $data->links() !!}
27
28
29 @include('include.fin')
```

A partir de cette page, nous cherchons à afficher toutes les données de la table Equipe et, à partir d'un formulaire, afficher les boutons pour créer, montrer, modifier ou supprimer une équipe.

## d. Page create

```
1  @include('include.debut')
2
3  @if ($errors->any())
4      <div>
5          <strong>ALERTE!</strong> Erreur lors de la création.<br><br>
6          <ul>
7              @foreach ($errors->all() as $error)
8                  <li>{{ $error }}</li>
9              @endforeach
10         </ul>
11     </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Entrez les informations de la nouvelle équipe</th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('equipe.store')}}" method="post">
24                     @csrf
25                     <strong>Nom:</strong>
26                     <input type="text" name="nomEquipe" placeholder="Nom"><br>
27                     <strong>Responsable:</strong>
28                     <input type="text" name="responsableEquipe" placeholder="Responsable"><br>
29                     <strong>Nombre de membres:</strong>
30                     <input type="number" name="nombreMembreEquipe" placeholder="Nombre de membres"><br>
31                     <strong>Pays:</strong>
32                     <input type="text" name="paysEquipe" placeholder="Pays"><br>
33                     <button type="submit">Envoyer</button>
34                 </form>
35             </td>
36         </tr>
37     </tbody>
38 </table>
39
40 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de créer une nouvelle équipe.

## e. Page edit

```
1 @include('include.debut')
2
3 @if ($errors->any())
4     <div>
5         <strong>ALERTE!</strong> Erreur lors de la mise à jour.<br><br>
6         <ul>
7             @foreach ($errors->all() as $error)
8                 <li>{{ $error }}</li>
9             @endforeach
10        </ul>
11    </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Mettez à jour l'équipe</th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('equipe.update',$equipe->id)}}" method="post">
24                     @csrf
25                     @method('PUT')
26                     <strong>Nom:</strong>
27                     <input type="text" name="nomEquipe" placeholder="Nom" value="{{ $equipe->nomEquipe }}"><br>
28                     <strong>Responsable:</strong>
29                     <input type="text" name="responsableEquipe" placeholder="Responsable" value="{{ $equipe->responsableEquipe }}"><br>
30                     <strong>Nombre de membres:</strong>
31                     <input type="number" name="nombreMembreEquipe" placeholder="Nombre de membres" value="{{ $equipe->nombreMembreEquipe }}"><br>
32                     <strong>Pays:</strong>
33                     <input type="text" name="paysEquipe" placeholder="Pays" value="{{ $equipe->paysEquipe }}"><br>
34                     <button type="submit">Envoyer</button>
35                 </form>
36             </td>
37         </tr>
38     </tbody>
39 </table>
40 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de modifier une équipe.

## f. Page show

```
1 @include('include.debut')
2
3 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
4   <thead>
5     <tr>
6       <th>Informations sur l'équipe</th>
7     </tr>
8   </thead>
9   <tbody>
10    <tr>
11      <td>
12        <strong>Nom:</strong> {{ $equipe->nomEquipe }}
13        <br>
14        <strong>Responsable:</strong> {{ $equipe->responsableEquipe }}
15        <br>
16        <strong>Nombre de membres:</strong> {{ $equipe->nombreMembreEquipe }}
17        <br>
18        <strong>Pays:</strong> {{ $equipe->paysEquipe }}
19        <br>
20      </td>
21    </tr>
22  </tbody>
23 </table>
24
25 @include('include.fin')
```

Ici, nous affichons les différentes informations sur l'équipe choisie.

## IV. Gérer la partie attribution

Tout d'abord, pour créer le Controller et Model, nous devons saisir cette commande dans le terminal : `php artisan make:controller AttributionController --resource --model=Attribution`

### a. Model Attribution

Le Model va nous servir à déterminer les colonnes de la table que nous pouvons manipuler. Nous allons remplir le fichier pour Attribution comme ceci :

```
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Attribution extends Model
9  {
10     use HasFactory;
11     protected $fillable = ['etablissements_idEtab', 'equipes_idEquipe'];
12 }
```

## b. Controller Attribution

La tâche d'un contrôleur est de réceptionner une requête (qui a déjà été sélectionnée par une route) et de définir la réponse appropriée. Nous allons remplir le Controller comme ceci :

```
1 </php>
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\DB;
7 use App\Models\Attribution;
8
9 class AttributionController extends Controller
10 {
11     /**
12      * Display a listing of the resource.
13      *
14      * @return \Illuminate\Http\Response
15      */
16     public function index()
17     {
18         $data = Attribution::latest()->paginate(5);
19
20         return view('attribution.index',compact('data'))
21             ->with('i', (request()->input('page', 1) - 1) * 5);
22     }
23
24     /**
25      * Show the form for creating a new resource.
26      *
27      * @return \Illuminate\Http\Response
28      */
29     public function create()
30     {
31         $etab = DB::table('etablissemments')->get();
32         $equipe = DB::table('equipes')->get();
33         return view('attribution.create',compact('etab','equipe'));
34     }
35
36     /**
37      * Store a newly created resource in storage.
38      *
39      * @param \Illuminate\Http\Request $request
40      * @return \Illuminate\Http\Response
41      */
42     public function store(Request $request)
43     {
44         $request->validate([
45             'etablissemments_idetab' => 'required',
46             'equipes_idequipe' => 'required'
47         ]);
48
49         Attribution::create($request->all());
50
51         return redirect()->route('attribution.index')
52             ->with('success','Attribution created successfully.');
```

```
53
54     /**
55      * Show the form for editing the specified resource.
56      *
57      * @param \App\Models\Attribution $attribution
58      * @return \Illuminate\Http\Response
59      */
60     public function edit(Attribution $attribution)
61     {
62         $etab = DB::table('etablissemments')->get();
63         $equipe = DB::table('equipes')->get();
64         return view('attribution.edit',compact('attribution','etab','equipe'));
65     }
66
67     /**
68      * Update the specified resource in storage.
69      *
70      * @param \Illuminate\Http\Request $request
71      * @param \App\Models\Attribution $attribution
72      * @return \Illuminate\Http\Response
73      */
74     public function update(Request $request, Attribution $attribution)
75     {
76         $request->validate([
77             'etablissemments_idetab' => 'required',
78             'equipes_idequipe' => 'required'
79         ]);
80
81         $attribution->update($request->all());
82
83         return redirect()->route('attribution.index')
84             ->with('success','Attribution updated successfully');
```

```
85
86     /**
87      * Remove the specified resource from storage.
88      *
89      * @param \App\Models\Attribution $attribution
90      * @return \Illuminate\Http\Response
91      */
92     public function destroy(Attribution $attribution)
93     {
94         $attribution->delete();
95
96         return redirect()->route('attribution.index')
97             ->with('success','Attribution deleted successfully');
```

## c. Page index

```
1 @include('include.debut')
2 <div>
3     <a href="{{ route('attribution.create') }}">Création d'une nouvelle attribution</a>
4 </div>
5
6 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
7     <thead>
8         <th>Attribution n°</th>
9         <th width="280px">Action</th>
10    </thead>
11    @foreach ($data as $key => $value)
12    <tr>
13        <td>{{ $value->id }}</td>
14        <td>
15            <form action="{{ route('attribution.destroy', $value->id) }}" method="POST">
16                <a href="{{ route('attribution.show', $value->id) }}"></a>
17                <a href="{{ route('attribution.edit', $value->id) }}"></a>
18                @csrf
19                @method('DELETE')
20                <button type="submit" class="btn btn-danger"></button>
21            </form>
22        </td>
23    </tr>
24    @endforeach
25 </table>
26 {!! $data->links() !!}
27
28
29 @include('include.fin')
```

A partir de cette page, nous cherchons à afficher toutes les données de la table Attribution et, à partir d'un formulaire, afficher les boutons pour créer, montrer, modifier ou supprimer une attribution.

## d. Page create

```
1 @include('include.debut')
2
3 @if ($errors->any())
4     <div>
5         <strong>ALERTE!</strong> Erreur lors de la création.<br><br>
6         <ul>
7             @foreach ($errors->all() as $error)
8                 <li>{{ $error }}</li>
9             @endforeach
10        </ul>
11    </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Entrez les informations de la nouvelle attribution </th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('attribution.store')}}" method="post">
24                     @csrf
25                     <strong>Etablissement:</strong>
26                     <select name="etablissements_idEtab">
27                         @if ($etab->count())
28                             @foreach($etab as $etabs)
29                                 <option value="{{ $etabs->id }}" >{{ $etabs->nomEtab }}</option>
30                             @endforeach
31                         @endif
32                     </select>
33                     <br>
34                     <strong>Equipe:</strong>
35                     <select name="equipes_idEquipe">
36                         @if ($equipe->count())
37                             @foreach($equipe as $equipes)
38                                 <option value="{{ $equipes->id }}" >{{ $equipes->nomEquipe }}</option>
39                             @endforeach
40                         @endif
41                     </select>
42                     <br>
43                     <button type="submit">Envoyer</button>
44                 </form>
45             </td>
46         </tr>
47     </tbody>
48 </table>
49
50 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de créer une nouvelle attribution.



## e. Page edit

```
1 @include('include.debut')
2
3 @if ($errors->any())
4     <div>
5         <strong>ALERTE!</strong> Erreur lors de la mise à jour.<br><br>
6         <ul>
7             @foreach ($errors->all() as $error)
8                 <li>{{ $error }}</li>
9             @endforeach
10        </ul>
11    </div>
12 @endif
13
14 <table class='styled-table' width='60%' cellpadding='0' cellspacing='0' align='center'>
15     <thead>
16         <tr>
17             <th>Mettez à jour l'attribution</th>
18         </tr>
19     </thead>
20     <tbody>
21         <tr>
22             <td>
23                 <form action="{{route('attribution.update',$attribution->id)}}" method="post">
24                     @csrf
25                     @method('PUT')
26                     <strong>Etablissement:</strong>
27                     <select name="etablisements_idEtab">
28                         @if ($etab->count())
29                             @foreach($etab as $etabs)
30                                 <option value="{{ $etabs->id }}" >{{ $etabs->nomEtab }}</option>
31                             @endforeach
32                         @endif
33                     </select>
34                     <br>
35                     <strong>Equipe:</strong>
36                     <select name="equipes_idEquipe">
37                         @if ($equipe->count())
38                             @foreach($equipe as $equipes)
39                                 <option value="{{ $equipes->id }}">{{ $equipes->nomEquipe }}</option>
40                             @endforeach
41                         @endif
42                     </select>
43                     <br>
44                     <button type="submit">Envoyer</button>
45                 </form>
46             </td>
47         </tr>
48     </tbody>
49 </table>
50 @include('include.fin')
```

Ici, nous avons créé le formulaire nous permettant de modifier une attribution.

## f. Page show

```
1 @include('include.debut')
2
3 <table class='styled-table' width='60%' cellspacing='0' cellpadding='0' align='center'>
4   <thead>
5     <td>Informations sur l'attribution n°{{ $attribution -> id }}</td>
6   </thead>
7   <tbody>
8     <td>
9       @foreach($etab as $etabs)
10         <strong>Nom de l'établissement:</strong> {{ $etabs -> nomEtab }}
11         <br>
12       @endforeach
13
14       @foreach($equipe as $equipes)
15         <strong>Nom de l'équipe:</strong> {{ $equipes -> nomEquipe }}
16         <br>
17       @endforeach
18     </td>
19   </tbody>
20 </table>
21
22 @include('include.fin')
```

Ici, nous affichons les différentes informations sur l'attribution choisie.