**1. Créer un projet Laravel**

composer create-project laravel/laravel gesElection

cd gesElection

**2. Configuration de la base de données**

Ouvrez le fichier .env et configurez les paramètres de votre base de données.

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=ges\_election

DB\_USERNAME=your\_username

DB\_PASSWORD=your\_password

Exécutez la commande pour créer la base de données (si elle n'existe pas déjà) :

php artisan db

**Notion de Modèles et Relations**

Chaque modèle représente une table et définit les relations avec les autres entités.

Quelques définitions nécessaires pour tous les modèles.

#### Relations :

* **Electeur ↔ Personne** : Un électeur est associé à une personne (belongsTo).
* **Electeur ↔ Lieuvote** : Un électeur est assigné à un lieu de vote (belongsTo).
* **Lieuvote ↔ Bureauvote** : Un lieu de vote peut contenir plusieurs bureaux de vote (hasMany).
* **Lieuvote ↔ Electeur** : Un lieu de vote est associé à plusieurs électeurs (hasMany).
* **Bureauvote ↔ Lieuvote** : Un bureau de vote appartient à un lieu de vote (belongsTo).
* **Candidat ↔ Personne** : Un candidat est une personne (relation belongsTo).
* **Candidat ↔ LigneParticipation** : Un candidat peut être associé à plusieurs participations dans les élections (relation hasMany).
* **Personne ↔ Electeur** : Une personne peut être associée à un électeur (relation hasOne).
* **Personne ↔ Candidat** : Une personne peut être associée à un candidat (relation hasOne).

### ****Synthèse des relations****

| **Modèle** | **Relation** |
| --- | --- |
| **Electeur** | belongsTo(Personne::class)  belongsTo(Lieuvote::class) |
| **Lieuvote** | hasMany(Bureauvote::class)  hasMany(Electeur::class) |
| **Bureauvote** | belongsTo(Lieuvote::class) |

**3. Créer les modèles, migrations et contrôleurs**

Exécutez les commandes suivantes pour générer les modèles et leurs migrations ainsi que les contrôleurs avec les ressources :

php artisan make:model Personne -mcr

php artisan make:model Electeur -mcr

php artisan make:model Lieuvote -mcr

php artisan make:model Bureauvote -mcr

php artisan make:model Candidat -mcr

php artisan make:model LigneParticipation -mcr

Cela crée :

* **Modèles** : fichiers dans app/Models
* **Migrations** : fichiers dans database/migrations
* **Contrôleurs** : fichiers dans app/Http/Controllers

**4. Définir les migrations**

Ouvrez chaque fichier dans database/migrations et ajoutez les colonnes nécessaires.

**Personne**

Schema::create('personnes', function (Blueprint $table) {

$table->id();

$table->string('nom');

$table->string('prenom');

$table->date('naissance');

$table->string('lieu');

$table->enum('sexe', ['M', 'F']);

$table->float('taille');

$table->float('poids');

$table->timestamps();

});

**Electeur**

Schema::create('electeurs', function (Blueprint $table) {

$table->id();

$table->foreignId('personne\_id')->constrained()->onDelete('cascade');

$table->foreignId('lieuvote\_id')->constrained()->onDelete('cascade');

$table->string('statut');

$table->timestamps();

});

**Lieuvote**

Schema::create('lieuvotes', function (Blueprint $table) {

$table->id();

$table->string('nomlieu');

$table->timestamps();

});

**Bureauvote**

Schema::create('bureauxvotes', function (Blueprint $table) {

$table->id();

$table->foreignId('lieuvote\_id')->constrained()->onDelete('cascade');

$table->integer('numero');

$table->text('description');

$table->timestamps();

});

**Candidat**

Schema::create('candidats', function (Blueprint $table) {

$table->id();

$table->foreignId('personne\_id')->constrained()->onDelete('cascade');

$table->string('diplôme');

$table->string('fonction');

$table->timestamps();

});

**LigneParticipation**

Schema::create('ligne\_participations', function (Blueprint $table) {

$table->id();

$table->foreignId('candidat\_id')->constrained()->onDelete('cascade');

$table->timestamps();

});

**5. Définir les relations dans les modèles**

Ajoutez les relations appropriées dans chaque modèle.

**Personne**

public function electeur()

{

return $this->hasOne(Electeur::class);

}

public function candidat()

{

return $this->hasOne(Candidat::class);

}

**Electeur**

public function personne()

{

return $this->belongsTo(Personne::class);

}

public function lieuvote()

{

return $this->belongsTo(Lieuvote::class);

}

**Lieuvote**

public function electeurs()

{

return $this->hasMany(Electeur::class);

}

public function bureauxvotes()

{

return $this->hasMany(Bureauvote::class);

}

**Bureauvote**

public function lieuvote()

{

return $this->belongsTo(Lieuvote::class);

}

**Candidat**

public function personne()

{

return $this->belongsTo(Personne::class);

}

public function ligneParticipations()

{

return $this->hasMany(LigneParticipation::class);

}

**6. Exécuter les migrations**

Lancez la commande pour créer les tables dans la base de données :

php artisan migrate

**7. Configurer les routes**

Dans routes/web.php, ajoutez :

use App\Http\Controllers\PersonneController;

use App\Http\Controllers\ElecteurController;

use App\Http\Controllers\LieuvoteController;

use App\Http\Controllers\BureauvoteController;

use App\Http\Controllers\CandidatController;

Route::resource('personnes', PersonneController::class);

Route::resource('electeurs', ElecteurController::class);

Route::resource('lieuvotes', LieuvoteController::class);

Route::resource('bureauxvotes', BureauvoteController::class);

Route::resource('candidats', CandidatController::class);

**8. Vérifiez vos travaux**

Lancez le serveur de développement et accédez au projet dans votre navigateur :

php artisan serve