







Utiliser une base de données Mysql

Presque toutes les applications Web modernes interagissent avec une base de données. Laravel rend l'interaction avec les bases de données extrêmement simple. Actuellement, Laravel fournit un support de première partie pour cinq bases de données : MariaDB 10.3+ ,MySQL 5.7+ ,PostgreSQL 10.0+, SQLite 3.8.8+ ,SQL Server 2017+.

Configuration :

La configuration des services de base de données de Laravel se trouve dans le fichier de configuration **config/database.php** de votre application. Dans ce fichier, vous pouvez définir toutes vos connexions à la base de données, ainsi que spécifier quelle connexion doit être utilisée par défaut. La plupart des options de configuration de ce fichier sont pilotées par les valeurs des variables d'environnement de votre application. Des exemples pour la plupart des systèmes de base de données pris en charge par Laravel sont fournis dans ce fichier.

Le fichier **.env** contient des valeurs de configuration courantes :

 channels.php	9	LOG_LEVEL=debug
 console.php	10	
 web.php	11	DB_CONNECTION=mysql
> storage	12	DB_HOST=127.0.0.1
> tests	13	DB_PORT=3306
> vendor	14	DB_DATABASE=laravel
 .editorconfig	15	DB_USERNAME=root
 .env	16	DB_PASSWORD=
 .env.example	17	

A faire : Mettez à jour le fichier **.env** avec le nom de la base de données à laquelle vous souhaitez vous connecter, ainsi que les autres variables d'environnement si elles diffèrent de la valeur par défaut.

Sélectionner les données :

Pour lire les données d'une table (après la configuration de la connexion), on peut définir ,dans notre contrôleur, la méthode suivante :

```
//On doit ajouter :  
use Illuminate\Support\Facades\DB;  
public function getDataFromDB(){  
$m = DB::select('select * from Module where codeM= 203');
```

```
return $m;
}
```

Enfin , et pour tester , on ajoute la route suivante :

```
Route::get('/data','App\Http\Controllers\GererForms@getDataFromDB');
```

Le résultat sera comme suit :



Exercice :

- 1- Remodifier la méthode 'getDataFromDB' pour pouvoir lister tous les modules enregistrés dans la base de données. Testez
- 2- Créer une vue qui permet d'afficher les modules dans une table :

Code du module	Titre	Masse Horaire
101	Métier et Formation	15
203	Gérer les données	100
205	Programmer en Back end	120

- 3- si aucun module n'est enregistré dans la bd, alors on aura :



Insérer les données :

Pour insérer les nouvelles données dans la base de données, on peut utiliser la méthode insert :

```
use Illuminate\Support\Facades\DB;
DB::insert('insert into Modules (codeM, Titre,MH) values (?, ?, ?)', [104, 'PSWS',120]);
```

Modifier les données :

Le même principe est utilisé pour modifier les valeurs des données existantes :

```
use Illuminate\Support\Facades\DB;
```

```
$affected = DB::update(
'update Module set MH = 100 where codeM = ?',
[203]
);
```

Supprimer les données :

Le même principe est utilisé pour supprimer les données existantes :

Le nombre de lignes supprimées est renvoyé par la méthode delete.

```
use Illuminate\Support\Facades\DB;
$deleted = DB::delete('delete from Module');
```

Gestion des transactions :

Vous pouvez utiliser la méthode de transaction fournie par la façade DB pour exécuter un ensemble d'opérations dans une transaction de base de données. Si une exception est levée lors de la clôture de la transaction, la transaction sera automatiquement annulée et l'exception sera à nouveau levée. Si la fermeture s'exécute avec succès, la transaction sera automatiquement validée :

```
use Illuminate\Support\Facades\DB;
DB::transaction(function () {
DB::insert('insert into Archive select * from Module where codeM=101');
DB::delete('delete from Module where codeM=101');
});
```

Test : supposons on a une erreur dans le nom de la table 'Module'. Alors, une exception est générée et aucune opération n'est effectuée.

NB : pour effectuer le test, vous pouvez créer une méthode TestTransaction dans votre contrôleur, puis créer une route qui appelle cette méthode

Remarque : Si vous souhaitez démarrer une transaction manuellement et avoir un contrôle total sur les rollbacks et les commits, vous pouvez utiliser les méthodes fournies par la façade DB :

```
DB::beginTransaction();
DB::rollBack();
DB::commit();
```

Commandes php artisan utiles :

<code>php artisan db:show</code>	Pour voir un aperçu de la BD, y compris sa taille, son type, le nombre de connexions ouvertes et un résumé de ses tables,
----------------------------------	---

php artisan db:table nomTable	obtenir un aperçu d'une table de la BD, y compris ses colonnes, types, attributs, clés et index
----------------------------------	---

TP :

L'objectif est d'améliorer notre site web et pouvoir modifier et/ou supprimer un module comme ici :

[Ajouter un nouveau module](#)

Code du module	Titre	Masse Horaire	
203	Gérer les données	100	Modifier Supprimer
205	Programmer en Back end	120	Modifier Supprimer

aussi, on peut insérer un nouveau module dans la base de données

Proposer une solution