LẬP TRÌNH WEB PHP NÂNG CAO

GV: Trần Thanh Tuấn

Laravel

Nội dung

- Giới thiệu
- Định nghĩa Model
- Truy vấn dữ liệu với Model
- Thêm mới và Cập nhật dữ liệu với Model
- Xoá dữ liệu với Model

Giới thiệu

- **Eloquent ORM** đi kèm Laravel cung cấp một **ActiveRecord** đơn giản và "đẹp" để làm việc với Cơ sở dữ liệu.
- Có một "Model" tương ứng với mỗi bảng trong CSDL để tương tác với bảng này.
- Model cho phép truy vấn dữ liệu từ bảng, cũng như thêm mới vào bảng.

Định nghĩa Model

- Các tập tin Model nằm trong thư mục app
- Các lớp Model kế thừa lớp Illuminate\Database\Eloquent\Model
- Tạo Model:

```
php artisan make:model <TênModel>
```

- Tạo Model + migration:

```
php artisan make:model <TênModel> --migration
```

```
php artisan make:model <TênModel> -m
```

Định nghĩa Model

- Ví dụ: Model **LinhVuc**

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class LinhVuc extends Model
{
    //
}
```

Truy xuất và lưu trữ thông tin vào bảng **linh_vucs**

Định nghĩa Model

- Thiết lập bảng tương ứng với Model:

```
protected $table = `<tên_bảng_tương_ứng_với_model>`;
```

- Thiết lập giá trị mặc định cho thuộc tính của Model:

```
protected $attributes = [
        `<thuộc_tính_1>` => <giá_trị_1>,
        `<thuộc_tính_2>` => <giá_trị_2>,
        ...,
        `<thuộc_tính_N>` => <giá_trị_N>
];
```

Truy vấn dữ liệu với Model

- Lấy tất cả dòng dữ liệu trong table

```
$<biến> = <Model>::all();
```

- Mỗi **Model** đóng vai trò là 1 **Query-builder** ==> có thể sử dụng các phương thức Query-builder trong bài trước: get(), where(), orWhere(), orderBy(), take(), find(), first(), count(), max(), min()…
- findOrFail(), firstOrFail()

Truy vấn dữ liệu với Model

- **Ví dụ:**
  - Lấy tất cả Lĩnh vực

```
$dsLinhVuc = LinhVuc::all();
```

  - Lấy lĩnh vực có id = 1

```
$linhVuc = LinhVuc::find(1);
```

```
$linhVuc = LinhVuc::findOrFail(1);
```

  - Lấy lĩnh vực có id = 1 hoặc id = 2 hoặc id = 3

```
$dsLinhVuc = LinhVuc::find([1, 2, 3]);
```

Truy vấn dữ liệu với Model

- **Ví dụ:**
  - Lấy danh sách người chơi có **điểm chơi game cao nhất > 1000**

```
$dsNguoiChoi = NguoiChoi::where(`diem_cao_nhat', `>', 1000)->get();
```

Thêm mới và Cập nhật dữ liệu

- **Thêm mới (cách 1):**
  - Bước 1: Tạo đối tượng từ Model
  - Bước 2: Gán các giá trị cho các thuộc tính của đối tượng
  - Bước 3: gọi phương thức save()

```
$linhVuc = new LinhVuc;
$linhVuc->ten_linh_vuc = 'Toán';
$linhVuc->save();
```

Trường **created_at, updated_at** sẽ được gán giá trị sau khi gọi phương thức save()

Thêm mới và Cập nhật dữ liệu

- **Thêm mới (cách 2):**

```
<biến> = Model::create([
        '<thuộc_tính_1>' => <giá_trị_1>,
        '<thuộc_tính_2>' => <giá_trị_2>,
        ...,
        '<thuộc_tính_N>' => <giá_trị_N>
]);
```

```
$linhVuc = LinhVuc::create([
        'ten_linh_vuc' => 'Toán'
]);
```

Thêm mới và Cập nhật dữ liệu

- **Thêm mới (cách 3):**

```
<biến> = new <Model>;
<biến>->fill([
        `<thuộc_tính_1>' => <giá_trị_1>,
        `<thuộc_tính_2>' => <giá_trị_2>,
        ...,
        `<thuộc_tính_N>' => <giá_trị_N>
]);
```

```
$linhVuc = new LinhVuc;
$linhVuc->fill([
        `ten_linh_vuc' => `Toán'
]);
```

Thêm mới và Cập nhật dữ liệu

- **Thêm mới:**
  - Lưu ý:
    - Thuộc tính $fillable: chứa danh sách các thuộc tính nên được gán giá trị khi **thêm mới** bằng cách **2, 3**.
    - Thuộc tính $guarded: chứa danh sách các thuộc tính không được gán giá trị khi **thêm mới** bằng cách **2, 3**.

Thêm mới và Cập nhật dữ liệu

- **Thêm mới:**
  - firstOrCreate()

```
// Tìm gói credit theo ten_goi, nếu không tìm thấy thì tạo mới đối tượng với thuộc tính ten_goi (tạo dòng mới trong CSDL)
$goi = GoiCredit::firstOrCreate([`ten_goi' => `Gói D']);

// Tìm gói credit theo ten_goi, nếu không tìm thấy thì tạo đối tượng mới với các thuộc tính ten_goi, credit, so_tien (tạo dòng mới trong CSDL)
$goi = GoiCredit::firstOrCreate(
    ['ten_goi' => Gói D'],
    [`credit' => 500, `so_tien' => `300000']
);
```

Thêm mới và Cập nhật dữ liệu

- **Thêm mới:**
  - firstOrNew()

```
// Tìm gói credit theo ten_goi, nếu không tìm thấy thì tạo mới đối tượng với thuộc tính ten_goi (chưa tạo dòng mới trong CSDL)
$goi = GoiCredit::firstOrNew([`ten_goi` => `Gói D`]);
$goi->save();

// Tìm gói credit theo ten_goi, nếu không tìm thấy thì tạo đối tượng mới với các thuộc tính ten_goi, credit, so_tien (chưa tạo dòng mới trong CSDL)
$goi = GoiCredit::firstOrNew(
    [`ten_goi` => Gói D`],
    [`credit` => 500, `so_tien` => `300000`]
);
$goi->save();
```

Thêm mới và Cập nhật dữ liệu

- **Cập nhật (cách 1):**
  - Bước 1: Sử dụng Model truy vấn đối tượng
  - Bước 2: Gán các giá trị cho các thuộc tính của đối tượng
  - Bước 3: gọi phương thức save()

```
$linhVuc = LinhVuc::find($id);
$linhVuc->ten_linh_vuc = 'Lý';
$linhVuc->save();
```

Trường **updated_at** sẽ được gán giá trị sau khi gọi phương thức save()

Thêm mới và Cập nhật dữ liệu

- **Cập nhật (cách 2):**
  - Sử dụng phương thức update()
  - Cập nhật nhiều đối tượng thoả mãn điều kiện

```
NguoiChoi::where(`diem_cao_nhat', `>', 1000)
                ->update([`credit' => 2000]);
```

Thêm mới và Cập nhật dữ liệu

- **Cập nhật:**
  - updateOrCreate()

```
// Tìm gói credit theo ten_goi:
// + Nếu tìm thấy thì cập nhật đối tượng với các thuộc tính credit, so_tien (Lưu vào CSDL)
// + Nếu không tìm thấy thì tạo đối tượng mới với các thuộc tính ten_goi, credit, so_tien (tạo dòng mới trong CSDL)
$goi = GoiCredit::updateOrCreate(
    ['ten_goi' => Gói D'],
    [`credit' => 500, `so_tien' => `300000']
);
```

Xoá dữ liệu với Model

- **Cách 1:**
  - Bước 1: Sử dụng Model truy vấn đối tượng
  - Bước 2: Gọi phương thức delete()

```
$linhVuc = LinhVuc::find($id);
$linhVuc->delete();
```

Xoá dữ liệu với Model

- **Cách 2:**
  - Xoá bằng id (khoá chính)
  - Gọi phương thức destroy()

```
LinhVuc::destroy(1);
LinhVuc::destroy(1, 2, 3);
LinhVuc::destroy([1, 2, 3]);
LinhVuc::destroy(collect([1, 2, 3]));
```

Xoá dữ liệu với Model

- **Cách 3:**
  - Xoá các dòng thoả mãn điều kiện

```
GoiCredit::where(`credit’, `>’, 2000)->delete();
```

Xoá dữ liệu với Model

- **Soft Delete:**
  - Sử dụng Trail Illuminate\Database\Eloquent\SoftDeletes trong Model

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes ;

class LinhVuc extends Model
{
    use SoftDeletes;
}
```

Xoá dữ liệu với Model

- **Soft Delete:**
  - Trường **deleted_at** được gán giá trị *(thời gian hiện tại)* khi gọi phương thức **delete()**
  - Sử dụng phương thức **trashed()** để kiểm tra một đối tượng của Model đã bị xoá hay chưa?

```
if($linhVuc->trashed())
{
    // lĩnh vực đã bị xoá (soft delete)
}
```

Xoá dữ liệu với Model

- **Soft Delete:**
  - Truy vấn dữ liệu
    - withTrashed(): lấy các dòng dữ liệu chưa xoá và đã bị xoá
    - onlyTrashed(): chỉ lấy các dòng dữ liệu đã bị xoá

```
$dsLinhVuc = LinhVuc::withTrashed()->get();
$dsLinhVucDaXoa = LinhVuc::onlyTrashed()->get();
```

Xoá dữ liệu với Model

- **Soft Delete:**
  - Khôi phục tình trạng bị xoá
    - restore()

```
$linhVucDaXoa->restore();
LinhVuc::withTrashed()->restore();
```

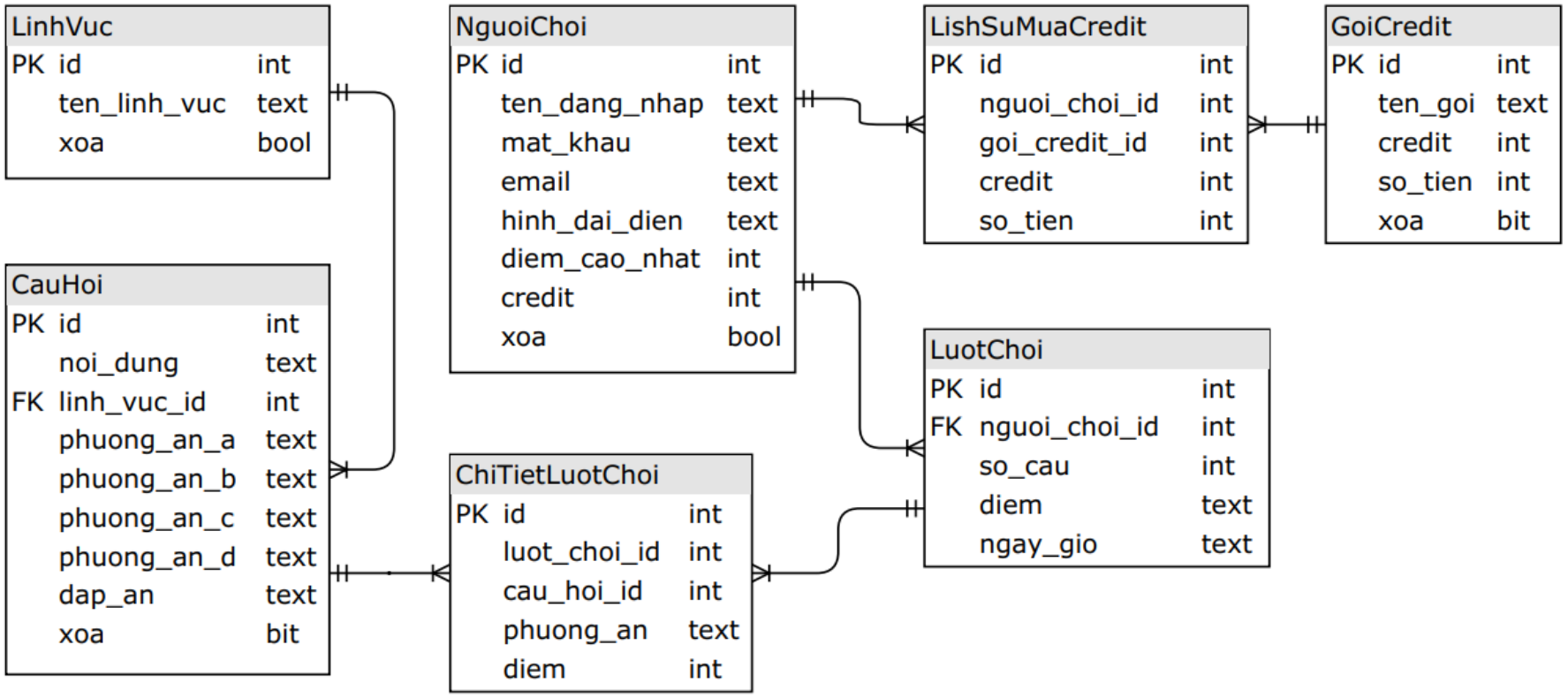
Xoá dữ liệu với Model

- **Soft Delete:**
  - Xoá khỏi CSDL
    - forceDelete()

```
$linhVucDaXoa->forceDelete();
```

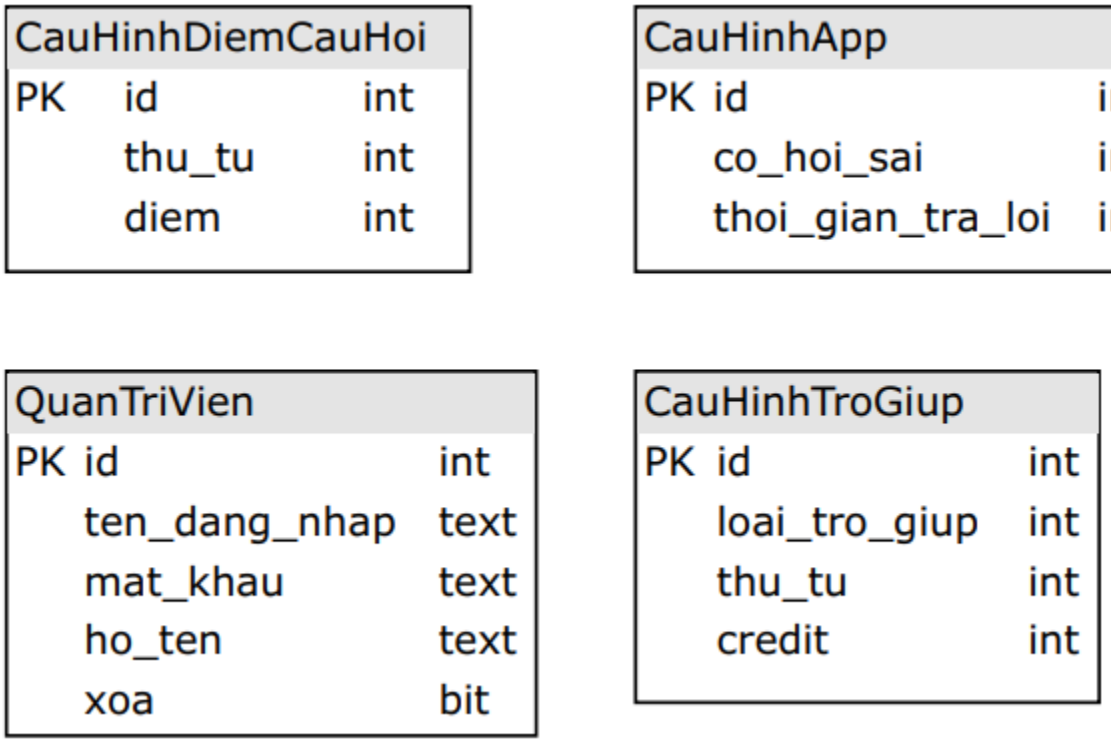
Bài tập 1

- Tạo các Model tương ứng với các bảng trong lược đồ CSDL sau:



Bài tập 1

- Tạo các Model tương ứng với các bảng trong lược đồ CSDL sau:



Bài tập 2

- Viết các lớp Seeder để tạo dữ liệu mẫu (<= 5 dòng dữ liệu) cho bảng:
  - Lĩnh vực
  - Quản trị viên
  - Gói credit