LẬP TRÌNH WEB 2

| Web Development 2|

Bùi Thị Phương Thảo – Phan Thanh Nhuần

[02 . 2018]

Khoa Công nghệ thông tin, Cao đẳng Công nghệ Thủ Đức

CHƯƠNG 4.

ELOQUENT ORM

Nội dung

- Giới thiệu Eloquent ORM
- Tạo Model
- Truy vấn dữ liệu
- Thêm, cập nhật, xóa
- Relationship
- Collection

Giới Thiệu Eloquent ORM

- **ORM (Object Relational Mapping)** là một kỹ thuật lập trình dùng để chuyển đổi dữ liệu giữa một hệ thống không hướng đối tượng như cơ sở dữ liệu sang hệ thống hướng đối tượng như lập trình hướng đối tượng trong PHP
- Laravel sử dụng kỹ thuật ORM giúp việc thao tác với cơ sở dữ liệu dễ dàng hơn
- **Laravel Eloquent Model** (gọi tắt là Model) là một phần trong mô hình MVC, các Model này sẽ thao tác trực tiếp với cơ sở dữ liệu, xử lý các logic nghiệp vụ và trả dữ liệu về cho các Controller

Tạo Model

- Để tạo một model, ta sử dụng lệnh

```
php artisan make:model User
```

- Trong đó, User là tên model
- Model sau khi được tạo sẽ nằm trong thư mục app, hoặc ta có thể để file model bất cứ đâu, và nó sẽ được tự động load dựa vào file composer.json
- Tất cả model đều kế thừa Illuminate\Database\Eloquent\Model

Tạo Model

- Khi tạo ra model tên User, Eloquent sẽ giả định tên bảng sẽ là tên số nhiều của tên file model là users
- Nếu tên bảng khác tên số nhiều của tên model, ta định nghĩa thêm thuộc tính $table trong model như sau:

```
class User extends Model {

    protected $table = 'my_users';

}
```

Tạo Model

- Eloquent cũng sẽ giả định khóa chính của bảng là id, nếu khóa chính khác id, ta phải định nghĩa thuộc tính $primaryKey như sau:

```
class User extends Model {

    protected $table = 'my_users';
    public $primaryKey = 'user_id';


}
```

Tạo Model

- Ngoài ra, khi tạo ra model, mặc định sẽ cần có 2 cột updated_at và created_at, để thuận tiện cho thao tác thêm và truy vấn dữ liệu
- Nếu trong bảng không có 2 cột trên thì ta phải thiết lập thuộc tính $timestamps = false

Truy Vấn Dữ Liệu

- Để lấy tất cả dữ liệu trong bảng users

```
$users = User::all();
```

- Để lấy một record dựa vào khóa chính

```
//Retrieving A Model By Primary Key
$user = User::find(1);
//Retrieving A Model By Primary Key Or Throw An Exception
$user = User::findOrFail(1);

var_dump($user->name);
```

Truy Vấn Dữ Liệu

- Để lấy record dựa điều kiện

```
$model = User::where('votes', '>', 100)->get();
```

- Giới hạn record trả về

```
$model = User::where('votes', '>', 100)->take(10)->get();
```

Xem thêm: https://laravel.com/docs/5.0/eloquent#basic-usage

Thêm

- Thêm mới một record, ta sử dụng phương thức save như sau:

```
$user = new User;

$user->name = 'John';

$user->save();
```

  - name trong $user->name là tên cột

- Lưu ý: khóa chính trong Eloquent models thông thường sẽ tự động tăng, nếu không tự động tăng ta phải set thuộc tính $incrementing = false

Thêm

- Ta có thể sử dụng phương thức create để thêm một dòng dữ liệu như sau:

```php
// Create a new user in the database...
$user = User::create(['name' => 'John']);

// Retrieve the user by the attributes, or create it if it doesn't exist...
$user = User::firstOrCreate(['name' => 'John']);
```

Cập nhật

- Để cập nhật một record, ta phải lấy record đó lên, thay đổi giá trị sau đó sử dụng phương thức save để lưu lại

```
$user = User::find(1);

$user->email = 'john@foo.com';

$user->save();
```

Xóa

- Để xóa một record, ta sử dụng phương thức delete

```
$user = User::find(1);

$user->delete();
```

- Xóa một record dựa vào khóa chính

```
User::destroy(1);

User::destroy([1, 2, 3]);

User::destroy(1, 2, 3);
```

Relationship

- Các bảng trong database có thể có mối liên quan lẫn nhau.
- Ví dụ: Một bài post sẽ có nhiều comment, một order sẽ liên quan tới user đặt nó.
- Laravel cung cấp các loại relationship:
  - One to One
  - One to Many
  - Many to Many
  - Xem thêm: https://laravel.com/docs/5.0/eloquent#relationships

One to One

- Quan hệ 1-1 là quan hệ cơ bản nhất. Ví dụ một user có một phone, ta định nghĩa quan hệ này trong Eloquent như sau:

```
class User extends Model {

    public function phone()
    {
        return $this->hasOne('App\Phone');
    }
}
```

- Phương thức hasOne có nhiều tham số
  - Tham số đầu tiên của phương thức hasOne là tên của model liên quan

One to One

- Tham số thứ 2 là khóa ngoại. Thông thường Eloquent giả định khóa ngoại của quan hệ dựa vào tên của model. Nếu khóa ngoại của model Phone khác user_id thì phải thêm tên khóa ngoại vào tham số thứ 2 của phương thức hasOne như sau:

```
class User extends Model {

    public function phone()
    {
        return $this->hasOne('App\Phone‘,’foreign_key’);
    }
}
```

One to One

- Tham số thứ 3 là khóa chính, mặc định sẽ sử dụng trường id hoặc trường được thiết lập trong biến $primaryKey của Model. Nếu bạn sử dụng một giá trị khác, thêm vào tham số thứ ba của phương thức như sau:

```
class User extends Model {

    public function phone()
    {
        return $this->hasOne('App\Phone‘,’foreign_key’,’local_key’);
    }
}
```

One to One

- Một khi quan hệ được định nghĩa, ta có thể truy vấn giá trị của bảng liên quan như sau:

```
$phone = User::find(1)->phone;
```

- Câu lệnh trên tương ứng với câu lệnh SQL

```
select * from users where id = 1

select * from phones where user_id = 1
```

The Inverse Of A Relation

- Sử dụng phương thức belongsTo để định nghĩa chiều ngược lại của quan hệ 1-1 với model Phone

```
class Phone extends Model {

    public function user()
    {
        return $this->belongsTo('App\User');
    }
}
```

- Tương tự hasOne, phương thức belongsTo cũng có ba tham số, tham số thứ nhất là tên Model có quan hệ ngược, tham số thứ hai là tên trường dùng làm khóa ngoại và tham số thứ ba là tên trường dùng làm khóa chính

One To Many

- Ví dụ một post có nhiều comment, ta định nghĩa quan hệ này trong Eloquent như sau:

```
class User extends Model {

    public function phone()
    {
        return $this->hasMany('App\Phone');
    }
}
```

One to Many

- Một khi quan hệ được định nghĩa, ta có thể truy vấn giá trị của bảng liên quan như sau:

```
$comments = Post::find(1)->comments;
```

- Tương tự phương thức hasOne, hasMany cũng có 3 tham số

```
return $this->hasMany('App\Comment', 'foreign_key', 'local_key');
```

The Inverse Of A Relation

- Sử dụng phương thức belongsTo để định nghĩa chiều ngược lại của quan hệ 1-n với model Comment

```
class Comment extends Model {

    public function post ()
    {
        return $this->belongsTo('App\User');
    }
}
```

Many to Many

Tham khảo: https://laravel.com/docs/5.0/eloquent#many-to-many

Collection

- Kết quả truy vấn Eloquent Model là một đối tượng Collection
- Đối tượng này có thể được duyệt như một mảng
- Tham khảo: https://laravel.com/docs/5.0/eloquent#collections

Thanks for your attention!

FACULTY OF INFORMATION TECHNOLOGY
Thu Duc College of Technology

Phone: (+848) 22 158 642
Email: fit@tdc.edu.vn
Website: fit.tdc.edu.vn