Course - Laravel Framework

Hashing

Facade Laravel Hash cung cấp chức năng băm Bcrypt và Argon2 an toàn để lưu trữ mật khẩu người dùng. Nếu bạn đang sử dụng một trong các bộ khởi động ứng dụng Laravel, Bcrypt sẽ được sử dụng để đăng ký và xác thực theo mặc định.

Tags: Hashing, ky thuat bam, bam, laravel

Giới thiệu

Facade `Hash` cung cấp chức năng băm *Bcrypt* và *Argon2*, chúng an toàn để lưu trữ mật khẩu người dùng. Nếu bạn đang sử dụng một trong các bộ khởi động nhanh ứng dụng Laravel, *Bcrypt* sẽ được sử dụng để đăng ký và xác thực theo mặc định.

Bcrypt là một lựa chọn tuyệt vời để băm mật khẩu vì "hệ số công việc" của nó có thể điều chỉnh được, có nghĩa là thời gian tạo ra một hàm băm có thể tăng lên khi sức mạnh phần cứng tăng lên. Khi băm mật khẩu, chậm là tốt nhất. Thuật toán càng mất nhiều thời gian để băm mật khẩu, thì người dùng độc hại càng mất nhiều thời gian để tạo "bảng rainbow" của tất cả các giá trị băm của chuỗi giá trị mà có thể được sử dụng trong các cuộc tấn công ác ý với các ứng dụng.

Cấu hình

Driver cho tính năng băm mặc định cho ứng dụng của bạn được cấu hình trong tập tin cấu hình *config/hashing.php* của ứng dụng. Hiện tại có một số driver được hỗ trợ: *Bcrypt* và *Argon2* (các biến thể *Argon2i* và *Argon2id*).

> **Chú ý:** Driver *Argon2i* yêu cầu PHP 7.2.0 trở lên và driver *Argon2id* yêu cầu PHP 7.3.0 trở lên.

Các cách sử dụng cơ bản

Băm mật khẩu

Bạn có thể băm mật khẩu bằng cách gọi phương thức `make` trên facade `Hash`:

```php
<?php

namespace App\Http\Controllers;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;

class PasswordController extends Controller
{
```

```
  /**
   * Update the password for the user.
   *
   * @param  \Illuminate\Http\Request  $request
   * @return \Illuminate\Http\Response
   */
  public function update(Request $request)
  {
    // Validate the new password length...

    $request->user()->fill([
      'password' => Hash::make($request->newPassword)
    ])->save();
  }
}
```

Điều chỉnh hệ số công việc *Bcrypt*

Nếu bạn đang sử dụng thuật toán *Bcrypt*, phương thức `make` cho phép bạn quản lý hệ số công việc của thuật toán bằng cách sử dụng tùy chọn `rounds`; tuy nhiên, hệ số công việc mặc định do Laravel quản lý có thể chấp nhận được đối với hầu hết các ứng dụng:

```
$hashed = Hash::make('password', [
  'rounds' => 12,
]);
```

Điều chỉnh hệ số công việc *Argon2*

Nếu bạn đang sử dụng thuật toán *Argon2*, phương thức make cho phép bạn quản lý yếu tố công việc của thuật toán bằng cách sử dụng các tùy chọn bộ nhớ, thời gian và luồng; tuy nhiên, các giá trị mặc định do Laravel quản lý có thể chấp nhận được đối với hầu hết các ứng dụng:

```
$hashed = Hash::make('password', [
  'memory' => 1024,
  'time' => 2,
```

```
    'threads' => 2,
]);
```

Để biết thêm thông tin về các tùy chọn này, vui lòng tham khảo tài liệu PHP chính thức về tính năng băm *Argon*.

Xác minh rằng mật khẩu khớp với một giá trị đã băm

Phương thức `check` được cung cấp bởi facade `Hash` cho phép bạn xác minh rằng một chuỗi văn bản thuần túy nào đó tương ứng với một hàm băm nhất định:

```
if (Hash::check('plain-text', $hashedPassword)) {
    // The passwords match...
}
```

Xác định xem mật khẩu có cần được tạo lại hay không

Phương thức `needsRehash` được cung cấp bởi facade `Hash` cho phép bạn xác định xem hệ số công việc được chức năng băm sử dụng có thay đổi kể từ khi mật khẩu được băm hay không. Một số ứng dụng chọn thực hiện bài kiểm tra này trong quá trình xác thực của ứng dụng:

```
if (Hash::needsRehash($hashed)) {
    $hashed = Hash::make('plain-text');
}
```