

Як написати команду artisan в Laravel, щоб зареєструвати супер адміністратора

🕒 24.05.2019 👤 Cinex 📁 Веб-розробка 💬 0

ПОШУК

ПОЗНАЧКИ

[APACHE](#)[AUTORUN](#)[BASH](#)[BITCOIN](#)[BLOCKCHAIN](#)[COMPOSER](#)[CRYPTOCURRENCY](#)[CSS](#)[DATABASE](#)[DJANGO](#)[DOCKER](#)[ELOQUENT](#)[ENGLISH](#)[ENUM](#)[GAMES](#)[GIMP](#)[GIT](#)[HTML](#)[JAVA](#)[JAVASCRIPT](#)[JQUERY](#)[LARAVEL](#)[MERCURIAL](#)[PARSING](#)[PHOTOSHOP](#)[PHP](#)[PHPSTORM](#)

```
Name:
> Ntim Yeboah

Email:
> ntimobedyeboah@gmail.com

Password:
>

Confirm password:
>
```

Super admin created

Name	Email	Super admin
Ntim Yeboah	ntimobedyeboah@gmail.com	1

Artisan - це інтерфейс командного рядка, включений в Laravel. Він має ряд корисних команд, які ви можете використовувати при створенні програми. Laravel дозволяє розробникам писати свої власні команди для виконання завдань. У цій статті ми збираємося написати команду, щоб зареєструвати супер адміністратора для додатка. Мета полягає в тому, щоб мати тільки одного супер адміністратора.

Створіть новий проект і очистіть всі маршрути та подання, необхідні для аутентифікації, за допомогою цієї команди:

```
1 php artisan make:auth
```

Додайте **is_super_adminfield** в таблицю міграції користувачів. Це поле буде використовуватися для визначення, чи існує супер-адміністратор чи ні.

```
1 $table->unsignedTinyInteger('is_super_admin')->default(0);
```

Приклад:

```
1 <?php
2 use Illuminate\Support\Facades\Schema;
3 use Illuminate\Database\Schema\Blueprint;
4 use Illuminate\Database\Migrations\Migration;
5 class CreateUsersTable extends Migration
6 {
7     /**
8      * Run the migrations.
9      *
10     * @return void
11     */
12     public function up()
13     {
14         Schema::create('users', function (Blueprint $table) {
15             $table->increments('id');
16             $table->string('name');
17             $table->string('email')->unique();
```

[PLAYONLINUX](#)[PLUGINS](#)[PROBLEM SOLVING](#)[PYTHON](#)[REGEXP](#)[SEO](#)[SOFTWARE ENGINEERING](#)[SPRING](#)[SQL](#)[TELEGRAM](#)[TIPS AND TRICKS](#)[TIPS AND TRICKS](#)[UNIT TESTING](#)[WORDPRESS](#)[ВІРШІ](#)[ШПАРГАЛКИ](#)

ОСТАННІ НОТАТКИ

20 Laravel Eloquent порад і трюків

15.12.2019

Як зробити скріншот сайту по URL на PHP

04.08.2019

Docker Cheat Sheet

05.07.2019

Гарячі клавіші Ubuntu Linux

01.07.2019

Git happens! 6 типових помилок Git і як їх виправити

22.06.2019

НЕДАВНІ КОМЕНТАРІ

```

18         $table->string('password');
19         $table->unsignedTinyInteger('is_super_admin')->default(0);
20         $table->rememberToken();
21         $table->timestamps();
22     });
23 }
24 /**
25  * Reverse the migrations.
26  *
27  * @return void
28  */
29 public function down()
30 {
31     Schema::dropIfExists('users');
32 }
33 }

```

Додайте **createSuperAdmin()** і **superAdminExists()** в модель користувача. Метод **createSuperAdmin()** зберігає дані супер-адміністратора в базі даних. **superAdminExists()** перевіряє, чи існує суперадмін.

```

1 <?php
2 namespace App;
3 use Illuminate\Notifications\Notifiable;
4 use Illuminate\Foundation\Auth\User as Authenticatable;
5 class User extends Authenticatable
6 {
7     use Notifiable;
8     /**
9      * The attributes that are mass assignable.
10     *
11     * @var array
12     */
13     protected $fillable = [
14         'name', 'email', 'password',
15     ];
16     /**
17      * The attributes that should be hidden for arrays.
18     *
19     * @var array
20     */
21     protected $hidden = [

```

```

22     'password', 'remember_token',
23 ];
24 /**
25  * Checks if user is a super admin
26  *
27  * @return boolean
28  */
29 public function isSuperAdmin() : bool
30 {
31     return (bool) $this->is_super_admin;
32 }
33 /**
34  * Create admin.
35  *
36  * @param array $details
37  * @return array
38  */
39 public function createSuperAdmin(array $details) : self
40 {
41     $user = new self($details);
42     if (! $this->superAdminExists()) {
43         $user->is_super_admin = 1;
44     }
45     $user->save();
46     return $user;
47 }
48 /**
49  * Checks if super admin exists
50  *
51  * @return integer
52  */
53 public function superAdminExists() : int
54 {
55     return self::where('is_super_admin', 1)->count();
56 }
57 }

```

Щоб створити клас команди, використовуйте команду **php artisan make:command**. Це створить новий клас команд в каталозі **app/Console/Commands**. Створіть клас команд з ім'ям **RegisterSuperAdminCommand**.

```
1 php artisan make:command RegisterSuperAdminCommand
```

В **RegisterSuperAdminCommandclass** надайте властивості підпису ім'я команди і властивість description з коротким поясненням того, що робить команда.

```
1 protected $signature = 'register:super-admin'; //Command will be called as php artisan re
2 protected $description = 'Register super admin';
```

В класі **RegisterSuperAdminCommand** запросіть дані супер-адміністратора, збережіть дані в базі даних і відобразіть деталі в таблиці.

```
1 <?php
2 namespace App\Console\Commands;
3 use App\User;
4 use Illuminate\Console\Command;
5 class RegisterSuperAdminCommand extends Command
6 {
7     /**
8      * The name and signature of the console command.
9      *
10     * @var string
11     */
12     protected $signature = 'register:super-admin';
13     /**
14      * The console command description.
15      *
16      * @var string
17      */
18     protected $description = 'Register super admin';
19     /**
20      * User model.
21      *
22      * @var object
23      */
24     private $user;
25     /**
26      * Create a new command instance.
27      *
28      * @return void
29      */
30     public function __construct(User $user)
31     {
```

```

32     parent::__construct();
33     $this->user = $user;
34 }
35 /**
36  * Execute the console command.
37  *
38  * @return mixed
39  */
40 public function handle()
41 {
42     $details = $this->getDetails();
43     $admin = $this->user->createSuperAdmin($details);
44     $this->display($admin);
45 }
46 /**
47  * Ask for admin details.
48  *
49  * @return array
50  */
51 private function getDetails() : array
52 {
53     $details['name'] = $this->ask('Name');
54     $details['email'] = $this->ask('Email');
55     $details['password'] = $this->secret('Password');
56     $details['confirm_password'] = $this->secret('Confirm password');
57     while (! $this->isValidPassword($details['password'], $details['confirm_password']) {
58         if (! $this->isRequiredLength($details['password'])) {
59             $this->error('Password must be more than six characters');
60         }
61         if (! $this->isMatch($details['password'], $details['confirm_password'])) {
62             $this->error('Password and Confirm password do not match');
63         }
64         $details['password'] = $this->secret('Password');
65         $details['confirm_password'] = $this->secret('Confirm password');
66     }
67     return $details;
68 }
69 /**
70  * Display created admin.
71  *
72  * @param array $admin
73  * @return void
74  */
75 private function display(User $admin) : void
76 {

```

```

77     $headers = ['Name', 'Email', 'Super admin'];
78     $fields = [
79         'Name' => $admin->name,
80         'email' => $admin->email,
81         'admin' => $admin->isSuperAdmin()
82     ];
83     $this->info('Super admin created');
84     $this->table($headers, [$fields]);
85 }
86 /**
87  * Check if password is valid
88  *
89  * @param string $password
90  * @param string $confirmPassword
91  * @return boolean
92  */
93 private function isValidPassword(string $password, string $confirmPassword) : bool
94 {
95     return $this->isRequiredLength($password) &&
96         $this->isMatch($password, $confirmPassword);
97 }
98 /**
99  * Check if password and confirm password matches.
100  *
101  * @param string $password
102  * @param string $confirmPassword
103  * @return bool
104  */
105 private function isMatch(string $password, string $confirmPassword) : bool
106 {
107     return $password === $confirmPassword;
108 }
109 /**
110  * Checks if password is longer than six characters.
111  *
112  * @param string $password
113  * @return bool
114  */
115 private function isRequiredLength(string $password) : bool
116 {
117     return strlen($password) > 6;
118 }
119 }

```

В методі **getDetails()**, якщо пароль менше восьми символів або якщо пароль і підтвердження пароля не збігаються, користувачеві буде запропоновано повторити спробу.

Тепер ми можемо бачити команду, коли запускаємо **php artisan** в командному рядку.

```
queue:table      Create a migration for the queue jobs database table
queue:work       Start processing jobs on the queue as a daemon
register         Register super admin
register:super-admin
route            Create a route cache file for faster route registration
route:cache      Remove the route cache file
route:clear
```

Ми можемо протестувати, запустивши в командному рядку **php artisan register:super-admin**. Після введення імені, адреси електронної пошти та пароля буде створено суперадмін, а введені нами дані будуть відображені в табличній формі.

```
Name:
> Ntim Yeboah
Email:
> ntimobedyeboah@gmail.com
Password:
>
Confirm password:
>
Super admin created
+-----+-----+
| Name          | Email                      | Super admin |
+-----+-----+
| Ntim Yeboah   | ntimobedyeboah@gmail.com   | 1           |
+-----+-----+
```

Джерело



LARAVEL

TIPS AND TRICKS



« ПОПЕРЕДНІЙ

Laravel CRUD генератор з нуля

ДАЛІ »

Користувальницький поштовий
драйвер в Laravel 5.8



ЗАЛИШТЕ ПЕРШИЙ КОМЕНТАР

Залишити коментар

Вашу адресу електронної пошти не буде опубліковано

Коментувати

Ім'я*

Email *

☐ Збережіть моє ім'я, електронну пошту у цьому веб-переглядачі наступного разу, коли я коментуватиму.

ОПУБЛІКУВАТИ КОМЕНТАР

Copyright © 2020