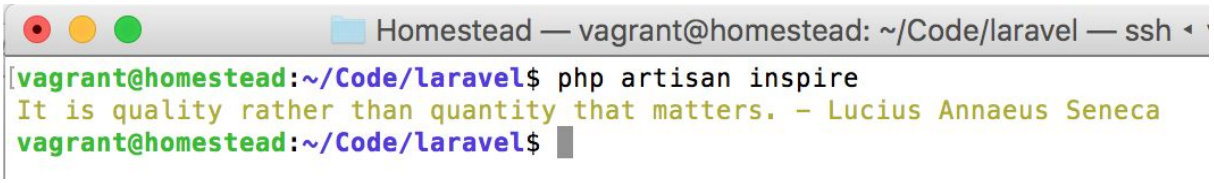

100 (and counting) Laravel Quick Tips

Prepared by:
Povilas Korop / LaravelDaily Team
www.laraveldaily.com
povilas@laraveldaily.com

Last updated:
April 2020

History of changes

April 20, 2020: 40 more tips, total 100 now (removed duplicated, updated some old ones)
April 28, 2019: 10 more tips, total 60 now
November 8, 2018: 10 more tips, total 50 now
October 9, 2018: Book release with 40 tips

Tip 1. Single Action Controllers

If you want to create a controller with just one action, you can use `__invoke()` method and even create "invokable" controller.

```
<?php
namespace App\Http\Controllers;

use App\User;
use App\Http\Controllers\Controller;

class ShowProfile extends Controller
{
    /**
     * Show the profile for the given user.
     *
     * @param  int  $id
     * @return Response
     */
    public function __invoke($id)
    {
        return view('user.profile', ['user' => User::findOrFail($id)]);
    }
}
```

Routes:

```
Route::get('user/{id}', 'ShowProfile');
```

Artisan command to generate this controller:

```
php artisan make:controller ShowProfile --invokable
```

Tip 2. Unsigned Integer

For foreign key migrations instead of `integer()` use `unsignedInteger()` type or `integer()->unsigned()`, otherwise you may get SQL errors.

```
Schema::create('employees', function (Blueprint $table) {
    $table->unsignedInteger('company_id');
    $table->foreign('company_id')->references('id')->on('companies');
    // ...
});
```

You can also use `unsignedBigInteger()` if that other column is `bigInteger()` type.

```
Schema::create('employees', function (Blueprint $table) {
    $table->unsignedBigInteger('company_id');
});
```

Tip 3. OrderBy on Eloquent relationships

You can specify `orderBy()` directly on your Eloquent relationships.

```
public function products()
{
    return $this->hasMany(Product::class);
}

public function productsByName()
{
    return $this->hasMany(Product::class)->orderBy('name');
}
```

Tip 4. Order of Migrations

If you want to change the order of DB migrations, just rename the file's timestamp, like from `2018_08_04_070443_create_posts_table.php` to `2018_07_04_070443_create_posts_table.php` (changed from *2018_08_04* to *2018_07_04*). They run in alphabetical order.

Tip 5. Raw DB Queries: havingRaw()

You can use RAW DB queries in various places, including `havingRaw()` function after `groupBy()`.

```
Product::groupBy('category_id')->havingRaw('COUNT(*) > 1')->get();
```

Tip 6. $loop variable in foreach

Inside of foreach loop, check if current entry is first/last by just using $loop variable.

```
@foreach ($users as $user)
    @if ($loop->first)
        This is the first iteration.
    @endif

    @if ($loop->last)
        This is the last iteration.
    @endif

    <p>This is user {{ $user->id }}</p>
@endforeach
```

There are also other properties like `$loop->iteration` or `$loop->count`.
More here: https://laravel.com/docs/master/blade#the-loop-variable

---

Tip 7. Eloquent where date methods

In Eloquent, check the date with functions `whereDay()`, `whereMonth()`, `whereYear()`, `whereDate()` and `whereTime()`.

```
$products = Product::whereDate('created_at', '2018-01-31')->get();
$products = Product::whereMonth('created_at', '12')->get();
$products = Product::whereDay('created_at', '31')->get();
$products = Product::whereYear('created_at', date('Y'))->get();
$products = Product::whereTime('created_at', '=', '14:13:58')->get();
```

Tip 8. Route group within a group

in Routes, you can create a group within a group, assigning a certain middleware only to some URLs in the "parent" group.

```
Route::group(['prefix' => 'account', 'as' => 'account.'], function() {

    Route::get('login', 'AccountController@login');
    Route::get('register', 'AccountController@register');

    Route::group(['middleware' => 'auth'], function() {
        Route::get('edit', 'AccountController@edit');
    });

});
```

Tip 9. Increments and decrements

if you want to increment some DB column in some table, just use `increment()` function. Oh, and you can increment not only by 1, but also by some number, like 50.

```
Post::find($post_id)->increment('view_count');
User::find($user_id)->increment('points', 50);
```

Tip 10. Does view file exist?

You can check if View file exists before actually loading it.

```
if (view()->exists('custom.page')) {
    // Load the view
}
```

You can even load an array of views and only the first existing will be actually loaded.

```
return view()->first(['custom.dashboard', 'dashboard'], $data);
```

Tip 11. No timestamp columns

If your DB table doesn't contain timestamp fields `created_at` and `updated_at`, you can specify that Eloquent model wouldn't use them, with `$timestamps = false` property.

```
class Company extends Model
{
    public $timestamps = false;
}
```

Tip 12. Migration fields with timezones

Did you know that in migrations there's not only `timestamps()` but also `timestampsTz()`, for the timezone?

```
Schema::create('employees', function (Blueprint $table) {
    $table->increments('id');
    $table->string('name');
    $table->string('email');
    $table->timestampsTz();
});
```

Also, there are columns `dateTimeTz()`, `timeTz()`, `timestampTz()`, `softDeletesTz()`.

Tip 13. Eloquent has() deeper

You can use Eloquent `has()` function to query relationships even two layers deep!

```
// Author -> hasMany(Book::class);
// Book -> hasMany(Rating::class);
$authors = Author::has('books.ratings')->get();
```

Tip 14. Database migrations column types

There are interesting column types for migrations, here are a few examples.

```
$table->geometry('positions');
$table->ipAddress('visitor');
$table->macAddress('device');
$table->point('position');
$table->uuid('id');
```

See all column types: https://laravel.com/docs/master/migrations#creating-columns

---

Tip 15. Artisan command help

To check the options of artisan command, Run artisan commands with `--help` flag. For example, `php artisan make:model --help` and see how many options you have:

```
Options:
  -a, --all             Generate a migration, factory, and resource controller for
the model
  -c, --controller      Create a new controller for the model
  -f, --factory         Create a new factory for the model
      --force           Create the class even if the model already exists.
  -m, --migration       Create a new migration file for the model.
  -p, --pivot           Indicates if the generated model should be a custom
intermediate table model.
  -r, --resource        Indicates if the generated controller should be a resource
controller.
  -h, --help            Display this help message
  -q, --quiet           Do not output any message
  -V, --version         Display this application version
      --ansi            Force ANSI output
      --no-ansi         Disable ANSI output
  -n, --no-interaction  Do not ask any interactive question
      --env[=ENV]       The environment the command should run under
  -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output, 2
for more verbose output and 3 for debug
```

Tip 16. Default Timestamp

While creating migrations, you can use `->timestamp()` column type with option `->useCurrent()`, it will set `CURRENT_TIMESTAMP` as default value.

```
$table->timestamp('created_at')->useCurrent();
$table->timestamp('updated_at')->useCurrent();
```

Tip 17. Set logged in user with Observers

Use `make:observer` and fill in `creating()` method to automatically set up `user_id` field for current logged in user.

```
class PostObserver
{
    /**
     * Handle to the post "creating" event.
     *
     * @param  \App\Post  $post
     * @return void
     */
    public function creating(Post $post)
    {
        $post->user_id = auth()->id();
    }
}
```

Tip 18. Soft-deletes: multiple restore

When using soft-deletes, you can restore multiple rows in one sentence.

```
Post::withTrashed()->where('author_id', 1)->restore();
```

Tip 19. Has Many. How many exactly?

In Eloquent `hasMany()` relationships, you can filter out records that have X amount of children records.

```
// Author -> hasMany(Book::class)
$authors = Author::has('books', '>', 5)->get();
```

Tip 20. Image validation

While validating uploaded images, you can specify the dimensions you require.

```
'photo' => 'dimensions:max_width=4096,max_height=4096'
```

Tip 21. Wildcard subdomains

You can create route group by dynamic subdomain name, and pass its value to every route.

```
Route::domain('{username}.workspace.com')->group(function () {
    Route::get('user/{id}', function ($username, $id) {
        //
    });
});
```

Tip 22. Exact Laravel version

Find out exactly what Laravel version you have in your app, by running command
`php artisan --version`

Tip 23. Testing email into laravel.log

If you want to test email contents in your app but unable or unwilling to set up something like Mailgun, use `.env` parameter `MAIL_DRIVER=log` and all the email will be saved into `storage/logs/laravel.log` file, instead of actually being sent.

Tip 24. Error code Blade pages

If you want to create a specific error page for some HTTP code, like 500 - just create a blade file with this code as filename, in `resources/views/errors/500.blade.php`, or `403.blade.php` etc, and it will automatically be loaded in case of that error code.

Tip 25. Factory callbacks

While using factories for seeding data, you can provide Factory Callback functions to perform some action after record is inserted.

```
$factory->afterCreating(App\User::class, function ($user, $faker) {
    $user->accounts()->save(factory(App\Account::class)->make());
});
```

Tip 26. Artisan command parameters

When creating Artisan command, you can ask the input in variety of ways: `$this->confirm()`, `$this->anticipate()`, `$this->choice()`.

```
// Yes or no?
if ($this->confirm('Do you wish to continue?')) {
    //
}

// Open question with auto-complete options
$name = $this->anticipate('What is your name?', ['Taylor', 'Dayle']);

// One of the listed options with default index
$name = $this->choice('What is your name?', ['Taylor', 'Dayle'],
$defaultIndex);
```

Tip 27. Preview Mailables

If you use Mailables to send email, you can preview the result without sending, directly in your browser. Just return a Mailable as route result:

```
Route::get('/mailable', function () {
    $invoice = App\Invoice::find(1);

    return new App\Mail\InvoicePaid($invoice);
});
```

Tip 28. Route::view() - Don't create Controllers

If you want route to just show a certain view, don't create a Controller method, just use `Route::view()` function.

```
// Instead of this
Route::get('about', 'TextsController@about');
// And this
class TextsController extends Controller
{
    public function about()
    {
        return view('texts.about');
    }
}

// Do this
Route::view('about', 'texts.about');
```

Tip 29. Blade @auth

Instead of if-statement to check logged in user, use `@auth` directive.

Typical way:

```
@if(auth()->user())
    // The user is authenticated.
@endif
```

Shorter:

```
@auth
    // The user is authenticated.
@endauth
```

The opposite is `@guest` directive.

```
@guest
    // The user is not authenticated.
@endguest
```

Tip 30. Model all: columns

When calling Eloquent's `Model::all()`, you can specify which columns to return.

```
$users = User::all(['id', 'name', 'email']);
```

Tip 31. Localhost in .env

Don't forget to change `APP_URL` in your `.env` file from `http://localhost` to real URL, cause it will be the basis for any links in your email notifications and elsewhere.

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:9PHz3TL5C4YrdV6Gg/Xkkmx9btaE93j7rQTUZWm2MqU=
APP_DEBUG=true
APP_URL=http://localhost
```

Tip 32: What's behind the routes?

Want to know what routes are actually behind `Auth::routes()`?
From Laravel 7, it's in a separate package, so check the file `/vendor/laravel/ui/src/AuthRouteMethods.php`.

```
public function auth()
{
    return function ($options = []) {
        // Authentication Routes...
        $this->get('login', 'Auth\LoginController@showLoginForm')->name('login');
        $this->post('login', 'Auth\LoginController@login');
        $this->post('logout', 'Auth\LoginController@logout')->name('logout');

        // Registration Routes...
        if ($options['register'] ?? true) {
            $this->get('register',
'Auth\RegisterController@showRegistrationForm')->name('register');
            $this->post('register', 'Auth\RegisterController@register');
        }

        // Password Reset Routes...
        if ($options['reset'] ?? true) {
            $this->resetPassword();
        }

        // Password Confirmation Routes...
        if ($options['confirm'] ??

class_exists($this->prependGroupNamespace('Auth\ConfirmPasswordController'))) {
            $this->confirmPassword();
        }

        // Email Verification Routes...
        if ($options['verify'] ?? false) {
            $this->emailVerification();
        }
    };
}
```

Before Laravel 7, check the file `/vendor/laravel/framework/src/illuminate/Routing/Router.php`.

```
public function auth(array $options = [])
{
    // Authentication Routes...
    $this->get('login', 'Auth\LoginController@showLoginForm')->name('login');
    $this->post('login', 'Auth\LoginController@login');
    $this->post('logout', 'Auth\LoginController@logout')->name('logout');

    // Registration Routes...
```

```
    if ($options['register'] ?? true) {
        $this->get('register',
'Auth\RegisterController@showRegistrationForm')->name('register');
        $this->post('register', 'Auth\RegisterController@register');
    }

    // Password Reset Routes...
    $this->get('password/reset',
'Auth\ForgotPasswordController@showLinkRequestForm')->name('password.request');
    $this->post('password/email',
'Auth\ForgotPasswordController@sendResetLinkEmail')->name('password.email');
    $this->get('password/reset/{token}',
'Auth\ResetPasswordController@showResetForm')->name('password.reset');
    $this->post('password/reset',
'Auth\ResetPasswordController@reset')->name('password.update');

    // Email Verification Routes...
    if ($options['verify'] ?? false) {
        $this->emailVerification();
    }
}

public function emailVerification()
{
    $this->get('email/verify',
'Auth\VerificationController@show')->name('verification.notice');
    $this->get('email/verify/{id}',
'Auth\VerificationController@verify')->name('verification.verify');
    $this->get('email/resend',
'Auth\VerificationController@resend')->name('verification.resend');
}
```

Tip 33. To Fail or not to Fail

In addition to `findOrFail()`, there's also Eloquent method `firstOrFail()` which will return 404 page if no records for query are found.

```
$user = User::where('email',
'povilas@laraveldaily.com')->firstOrFail();
```

Tip 34. Column name change

in Eloquent Query Builder, you can specify "as" to return any column with a different name, just like in plain SQL query.

```
$users = DB::table('users')
    ->select('name', 'email as user_email')
    ->get();
```

Tip 35. Logging with parameters

You can write `Log::info()`, or shorter `info()` message with additional parameters, for more context about what happened.

```
Log::info('User failed to login.', ['id' => $user->id]);
```

Tip 36. Default Model

You can assign a default model in `belongsTo` relationship, to avoid fatal errors when calling it like `{{ $post->user->name }}` if `$post->user` doesn't exist.

```
/**
 * Get the author of the post.
 */
public function user()
{
    return $this->belongsTo('App\User')->withDefault();
}
```

Tip 37. Use hasMany to create Many

If you have `hasMany()` relationship, you can use `saveMany()` to save multiple "child" entries from your "parent" object, all in one sentence.

```
$post = Post::find(1);
$post->comments()->saveMany([
      new Comment(['message' => 'First comment']),
      new Comment(['message' => 'Second comment']),
]);
```

Tip 38. More convenient DD

Instead of doing `dd($result);` you can put `->dd()` as a method directly at the end of your Eloquent sentence, or any Collection.

```
// Instead of
$users = User::where('name', 'Taylor')->get();
dd($users);

// Do this
$users = User::where('name', 'Taylor')->get()->dd();
```

Tip 39. Map query results

After Eloquent query you can modify rows by using `map()` function in Collections.

```
$users = User::where('role_id', 1)->get()->map(function (User $user) {
    $user->some_column = some_function($user);
    return $user;
});
```

Tip 40. Custom validation error messages

You can customize validation error messages per **field**, **rule** and **language** - just create a specific language file `resources/lang/xx/validation.php` with appropriate array structure.

```
'custom' => [
    'email' => [
        'required' => 'We need to know your e-mail address!',
    ],
],
```

Tip 41. When (NOT) to run “composer update”

Not so much about Laravel, but... Never run `composer update` on production, it's slow and will "break" repository. Always run `composer update` locally on your computer, commit new `composer.lock` to the repository, and run `composer install` on server.

Tip 42. Two-level $loop variable in Blade

In Blade's foreach you can use $loop variable even in two-level loop to reach parent variable.

```
@foreach ($users as $user)
    @foreach ($user->posts as $post)
        @if ($loop->parent->first)
            This is first iteration of the parent loop.
        @endif
    @endforeach
@endforeach
```

Tip 43. Route Model Binding: You can define a key

You can do Route model binding like `Route::get('api/users/{user}', function (App\User $user) { … }` - but not only by ID field. If you want `{user}` to be a `username` field, put this in the model:

```
public function getRouteKeyName() {
    return 'username';
}
```

Tip 44. Redirect to Specific Controller Method

You can `redirect()` not only to URL or specific route, but to a specific Controller's specific method, and even pass the parameters. Use this:

```
return redirect()->action('SomeController@method',
  ['param' => $value]);
```

Tip 45. Did you know about Auth::once()?

You can login with user only for ONE REQUEST, using method `Auth::once()`. No sessions or cookies will be utilized, which means this method may be helpful when building a stateless API.

```
if (Auth::once($credentials)) {
    //
}
```

Tip 46. Eager Loading with Exact Columns

You can do Laravel Eager Loading and specify the exact columns you want to get from the relationship.

```
$users = App\Book::with('author:id,name')->get();
```

You can do that even in deeper, second level relationships:

```
$users = App\Book::with('author.country:id,name')->get();
```

Tip 47. Validate dates with "now" or "yesterday" words

You can validate dates by rules before/after and passing various strings as a parameter, like: "tomorrow", "now", "yesterday". Example: `'start_date' => 'after:now'`. It's using strtotime() under the hood.

```
$rules = [
    'start_date' => 'after:tomorrow',
    'end_date' => 'after:start_date'
];
```

Tip 48. Touch parent updated_at easily

If you are updating a record and want to update the `updated_at` column of parent relationship (like, you add new post comment and want `posts.updated_at` to renew), just use `$touches = ['post'];` property on child model.

```
class Comment extends Model
{
    /**
     * All of the relationships to be touched.
     *
     * @var array
     */
    protected $touches = ['post'];

}
```

Tip 49. Quickly Navigate from Routes file to Controller

Instead of routing like this:

```
Route::get('page', 'PageController@action');
```

You can specify the Controller as a class:

```
Route::get('page',
  [\App\Http\Controllers\PageController::class, 'action']);
```

Then you will be able to click on “PageController” in PhpStorm, and navigate directly to Controller, instead of searching for it manually.

Tip 50. Always Check if Relationship Exists

Never **ever** do `$model->relationship->field` without checking if relationship object still exists.

It may be deleted for whatever reason, outside your code, by someone else's queued job etc. Do `if-else`, or `{{ $model->relationship->field ?? '' }}` in Blade, or `{{ optional($model->relationship)->field }}`

Tip 51. Don’t Filter by NULL in Collections

You can filter by NULL in Eloquent, but if you're filtering the **collection** further - filter by empty string, there's no "null" in that field anymore.

```
// This works
$messages = Message::where('read_at is null')->get();

// Won’t work - will return 0 messages
$messages = Message::all();
$unread_messages = $messages->where('read_at is null')->count();

// Will work
$unread_messages = $messages->where('read_at', '')->count();
```

Tip 52. Default Email Subject in Laravel Notifications

If you send Laravel Notification and don't specify subject in **toMail()**, default subject is your notification class name, CamelCased into Spaces.

So, if you have:

```
class UserRegistrationEmail extends Notification { // ...
```

Then you will receive an email with subject “User Registration Email”.

Tip 53. Composer: Check for Newer Versions

If you want to find out which of your **composer.json** packages have released newer versions, just run "**composer outdated**". You will get a full list with all information, like this below.

```
phpdocumentor/type-resolver 0.4.0 0.7.1
phpunit/php-code-coverage   6.1.4 7.0.3 Library that provides collection, processing, and rende...
phpunit/phpunit             7.5.9 8.1.3 The PHP Unit Testing framework.
ralouphie/getallheaders     2.0.5 3.0.3 A polyfill for getallheaders.
sebastian/global-state      2.0.0 3.0.0 Snapshotting of global state
```

Tip 54. Route Fallback - When no Other Route is Matched

If you want to specify additional logic for not-found routes, instead of just throwing default 404 page, you may create a special Route for that, at the very end of your Routes file.

```
Route::group(['middleware' => ['auth'], 'prefix' => 'admin', 'as' =>
'admin.'], function () {
    Route::get('/home', 'HomeController@index');
    Route::resource('tasks', 'Admin\TasksController');
});

// Some more routes....

Route::fallback(function() {
    return 'Hm, why did you land here somehow?';
});
```

Tip 55. Create Your Own Blade Directive

It's very easy - just add your own method in **app/Providers/AppServiceProvider.php**: For example, if you want to have this for replace
 tags with new lines:

```
<textarea>@br2nl($post->post_text)</textarea>
```

Add this directive to AppServiceProvider's **boot()** method:

```
public function boot()
{
    Blade::directive('br2nl', function ($string) {
        return "<?php echo preg_replace('/\<br(\s*)?\/?\>/i', \"\n\", $string); ?>";
    });
}
```

Tip 56. Use withCount() to Calculate Child Relationships Records

If you have **hasMany()** relationship, and you want to calculate "children" entries, don't write a special query. For example, if you have posts and comments on your User model, write this **withCount()**:

```
public function index()
{
    $users = User::withCount(['posts', 'comments'])->get();
    return view('users', compact('users'));
}
```

And then, in your Blade file, you will access those number with **[*relationship*]_count** properties:

```
@foreach ($users as $user)
    <tr>
        <td>{{ $user->name }}</td>
        <td class="text-center">{{ $user->posts_count }}</td>
        <td class="text-center">{{ $user->comments_count }}</td>
    </tr>
@endforeach
```

Tip 57. Use groupBy on Collections with Custom Callback Function

If you want to group result by some condition whith isn't a direct column in your database, you can do that by providing a closure function.

For example, if you want to group users by day of registration, here's the code:

```
$users = User::all()->groupBy(function($item) {
    return $item->created_at->format('Y-m-d');
});
```

Notice: it is done on a **Collection** class, so performed **AFTER** the results are fetched from the database.

Tip 58. Blade Directives: IncludeIf, IncludeWhen, IncludeFirst

If you are not sure whether your Blade partial file actually would exist, you may use these condition commands:

This will load header only if Blade file exists
`@includeIf('partials.header')`

This will load header only for user with role_id 1
`@includeWhen(auth()->user()->role_id == 1, 'partials.header')`

This will try to load adminlte.header, if missing - will load default.header
`@includeFirst('adminlte.header', 'default.header')`

Tip 59. Change Default Timestamp Fields

What if you're working with non-Laravel database and your timestamp columns are named differently? Maybe, you have create_time and update_time. Luckily, you can specify them in the model, too:

```
class Role extends Model
{
    const CREATED_AT = 'create_time';
    const UPDATED_AT = 'update_time';
```

Tip 60. Quick Order by created_at

Instead of:
```
User::orderBy('created_at', 'desc')->get();
```

You can do it quicker:
```
User::latest()->get();
```

By default, `latest()` will order by `created_at`.

There is an opposite method `oldest()` which would order by created_at ascending.
```
User::oldest()->get();
```

Also, you can specify another column to order by. For example, if you want to use `updated_at`, you can do this:
```
$lastUpdatedUser = User::newest('updated_at')->first();
```

Tip 61. Generate Images with Seeds/Factories

Did you know that Faker can generate not only text values but also IMAGES? See `avatar` field here - it will generate 50x50 image:

```
$factory->define(User::class, function (Faker $faker) {
    return [
        'name' => $faker->name,
        'email' => $faker->unique()->safeEmail,
        'email_verified_at' => now(),
        'password' => bcrypt('password'),
        'remember_token' => Str::random(10),
        'avatar' => $faker->image(storage_path('images'), 50, 50)
    ];
});
```

Tip 62. Eloquent: Update Parent in One Line

If you have a `belongsTo()` relationship, you can update the Eloquent relationship data in the same sentence:

```
// if Project -> belongsTo(User::class)
$project->user->update(['email' => 'some@gmail.com']);
```

Tip 63. Eloquent: Laravel 7+ Foreign Keys

From Laravel 7, in migrations you don't need to write two lines for relationship field - one for the field and one for foreign key. Use method `foreignId()`.

```
// Before Laravel 7
Schema::table('posts', function (Blueprint $table)) {
    $table->unsignedBigInteger('user_id');
    $table->foreign('user_id')->references('id')->on('users');
}

// From Laravel 7
Schema::table('posts', function (Blueprint $table)) {
    $table->foreignId('user_id')->constrained();
}

// Or, if your field is different from the table reference
Schema::table('posts', function (Blueprint $table)) {
    $table->foreignId('created_by_id')->references('id')->on('users');
}
```

Tip 64. Multiple Collection Methods in a Row

If you query all results with ->all() or ->get(), you may then perform various Collection operations on the same result, it won't query database every time.

```
$users = User::all();
echo 'Max ID: ' . $users->max('id');
echo 'Average age: ' . $users->avg('age');
echo 'Total budget: ' . $users->sum('budget');
```

Tip 65. More Events on User Registration

Want to perform some actions after new user registration? Head to `app/Providers/EventServiceProvider.php` and add more Listeners classes, and then in those classes implement `handle()` method with `$event->user` object

```
class EventServiceProvider extends ServiceProvider
{
    protected $listen = [
        Registered::class => [
            SendEmailVerificationNotification::class,

            // You can add any Listener class here
            // With handle() method inside of that class
        ],
    ];
```

Tip 66. Extra Filter Query on Relationships

If you want to load relationship data, you can specify some limitations or ordering in a closure function. For example, if you want to get Countries with only three of their biggest cities, here's the code.

```
$countries = Country::with(['cities' => function($query) {
      $query->orderBy('population', 'desc');
      $query->take(3);
}])->get();
```

Tip 67. Send Notifications to Anyone

You can send Laravel Notifications not only to a certain user with `$user->notify()`, but also to anyone you want, via `Notification::route()`, with so-called "on-demand" notifications:

```
Notification::route('mail', 'taylor@example.com')
            ->route('nexmo', '5555555555')
            ->route('slack',
'https://hooks.slack.com/services/...')
            ->notify(new InvoicePaid($invoice));
```

Tip 68. Sub-selects in Laravel Way

From Laravel 6, you can use `addSelect()` in Eloquent statement, and do some calculation to that added column.

```
return Destination::addSelect(['last_flight' => Flight::select('name')
    ->whereColumn('destination_id', 'destinations.id')
    ->orderBy('arrived_at', 'desc')
    ->limit(1)
])->get();
```

Tip 69. API Resources: With or Without “data”?

If you use Eloquent API Resources to return data, they will be automatically wrapped in 'data'. If you want to remove it, add `JsonResource::withoutWrapping();` in `app/Providers/AppServiceProvider.php`.

```
class AppServiceProvider extends ServiceProvider
{
    public function boot()
    {
        JsonResource::withoutWrapping();
    }
}
```

Tip 70. API Return “Everything went ok”

If you have API endpoint which performs some operations but has no response, so you wanna return just "everything went ok", you may return 204 status code *"No content"*: https://httpstatuses.com/204
In Laravel, it's easy: `return response()->noContent();`

```
public function reorder(Request $request)
{
    foreach ($request->input('rows', []) as $row) {
        Country::find($row['id'])->update([
            'position' => $row['position'];
        ]);
    }

    return response()->noContent();
}
```

Tip 71. Automatic Column Value When Creating Records

If you want to generate some DB column value when creating record, add it to model's `boot()` method.
For example, if you have a field "position" and want to assign the next available position to the new record (like `Country::max('position') + 1`), do this.

```
class Country extends Model {
    // ...
    protected static function boot()
    {
        parent::boot();

        // every new record will get next available position
number
        Country::creating(function($model) {
            $model->position = Country::max('position') + 1;
        });
    }
}
```

Tip 72. DB Raw Query Calculations Run Faster

Use SQL raw queries like `whereRaw()` method, to make some DB-specific calculations directly in query, and not in Laravel, usually the result will be faster. Like, if you want to get users that were active 30+ days after their registration, here's the code.

```
User::where('active', 1)
  ->whereRaw('TIMESTAMPDIFF(DAY, created_at, updated_at) > ?', 30)
  ->get();
```

Tip 73. More than One Scope

You can combine and chain Query Scopes in Eloquent, using more than one scope in a query.

```
// app/User.php model
public function scopeActive($query) {
    return $query->where('active', 1);
}
public function scopeRegisteredWithinDays($query, $days) {
    return $query->where('created_at', '>=',
now()->subDays($days));
}

// Some Controller
$users = User::registeredWithinDays(30)->active()->get();
```

Tip 74. Hide Some Columns

When doing Eloquent query, if you want to hide specific field from being returned, one of the quickest ways is to add `->makeHidden()` on Collection result.

```
$users = User::all()->makeHidden(['email_verified_at', 'deleted_at']);
```

Tip 75. Combine Two “whereHas”

In Eloquent, you can combine `whereHas()` and `orDoesntHave()` in one sentence.

```
User::whereHas('roles', function($query) {
    $query->where('id', 1);
})
->orDoesntHave('roles')
->get();
```

Tip 76. Auto-Capitalize Translations

In translation files (`resources/lang`), you can specify variables not only as `:variable`, but also capitalized as `:VARIABLE` or `:Variable` - and then whatever value you pass - will be also capitalized automatically.

```
// resources/lang/en/messages.php
'welcome' => 'Welcome, :Name'

// Result: "Welcome, Taylor"
echo __('messages.welcome', ['name' => 'taylor']);
```

Tip 77. Validation Rule with Some Conditions

If your validation rules depend on some condition, you can modify the rules by adding `withValidator()` to your `FormRequest` class, and specify your custom logic there. Like, if you want to add validation rule only for some user role.

```
use Illuminate\Validation\Validator;

class StoreBlogCategoryRequest extends FormRequest {

    // ...

    public function withValidator(Validator $validator) {
        if (auth()->user()->is_admin) {
            $validator->addRules([
                'some_secret_password' => 'required'
            ]);
        }
    }
}
```

Tip 78. Check if Relationship Method Exists

If your Eloquent relationship names are dynamic and you need to check if relationship with such name exists on the object, use PHP function `method_exists($object, $methodName)`

```
$user = User::first();
if (method_exists($user, 'roles')) {
     // Do something with $user->roles()->...
}
```

Tip 79. Exact DB Error

If you want to catch Eloquent Query exceptions, use specific `QueryException` instead default Exception class, and you will be able to get the exact SQL code of the error.

```
try {
    // Some Eloquent/SQL statement
} catch (\Illuminate\Database\QueryException $e) {
    if ($e->getCode() === '23000') { // integrity constraint violation
        return back()->withError('Invalid data');
    }
}
```

Tip 80. Route Parameters Validation with RegExp

We can validate parameters directly in the route, with “where” parameter. A pretty typical case is to prefix your routes by language locale, like **fr/blog** and **en/article/333**. How do we ensure that those two first letters are not used for some other than language?

```
// routes/web.php
Route::group([
    'prefix' => '{locale}',
    'where' => ['locale' => '[a-zA-Z]{2}']
], function () {
    Route::get('/', 'HomeController@index');
    Route::get('article/{id}', 'ArticleController@show');
});
```

Tip 81. Rate Limiting: Global and for Guests/Users

You can limit some URL to be called a maximum of 60 times per minute, with `throttle:60,1`.

```
Route::middleware('auth:api', 'throttle:60,1')->group(function () {
    Route::get('/user', function () {
        //
    });
});
```

But also, you can do it separately for public and for logged-in users.

```
// maximum of 10 requests for guests, 60 for authenticated users
Route::middleware('throttle:10|60,1')->group(function () {
    //
});
```

Also, you can have a DB field users.rate_limit and limit the amount for specific user:

```
Route::middleware('auth:api', 'throttle:rate_limit,1')->group(function () {
    Route::get('/user', function () {
        //
    });
});
```

Tip 82. Add Parameters to Routes

If you pass additional parameters to the route, in the array, those key / value pairs will automatically be added to the generated URL's query string.

```
Route::get('user/{id}/profile', function ($id) {
    //
})->name('profile');

$url = route('profile', ['id' => 1, 'photos' => 'yes']);
// Result: /user/1/profile?photos=yes
```

Tip 83. Check Multiple Permissions at Once

In addition to `@can` Blade directive, did you know you can check multiple permissions at once with `@canany` directive?

```
@canany(['update', 'view', 'delete'], $post)
    // The current user can update, view, or delete the post
@elsecanany(['create'], \App\Post::class)
    // The current user can create a post
@endcanany
```

Tip 84. Check Auth Quicker

If, in Blade, you need to check if user is logged in, don't do `@if (auth()->check())` - there is a shorter way: directives `@auth - @endauth` and `@guest - @endguest`

```
@auth
    // The user is authenticated...
@endauth

@guest
    // The user is not authenticated...
@endguest
```

Tip 85. No Need to Convert Carbon

If you're performing `whereDate()` and check today's records, you can use Carbon's `now()` and it will automatically be transformed to date. No need to do `->toDateString()`.

```
// Instead of
$todayUsers = User::whereDate('created_at', now()->toDateString())->get();

// No need to convert, just use now()
$todayUsers = User::whereDate('created_at', now())->get();
```

Tip 86. Soft-Deletes with Query Builder

Don't forget that soft-deletes will exclude entries when you use Eloquent, but won't work if you use Query Builder.

```
// Will exclude soft-deleted entries
$users = User::all();

// Will NOT exclude soft-deleted entries
$users = DB::table('users')->get();
```

Tip 87. Grouping by First Letter

You can group Eloquent results by any custom condition, here's how to group by first letter of user's name.

```
$users = User::all()->groupBy(function($item) {
    return $item->name[0];
})
```

Tip 88. Load Relationships Always, but Dynamically

You can not only specify what relationships to ALWAYS load with the model, but you can do it dynamically, in the constructor method:

```
class ProductTag extends Model {
    protected $with = ['product'];

    public function __construct() {
        parent::__construct();
        $this->with = ['product'];
        if (auth()->check()) {
            $this->with[] = 'user';
        }
    }
}
```

Tip 89. Instead of belongsTo, use hasMany

For `belongsTo` relationship, instead of passing parent's ID when creating child record, use `hasMany` relationship to make a shorter sentence.

```
// if Post -> belongsTo(User), and User -> hasMany(Post)...
// Then instead of passing user_id...
Post::create([
    'user_id' => auth()->id(),
    'title' => request()->input('title'),
    'post_text' => request()->input('post_text'),
]);

// Do this
auth()->user()->posts()->create([
    'title' => request()->input('title'),
    'post_text' => request()->input('post_text'),
]);
```

Tip 90. Change Default Validation Messages

If you want to change default validation error message for specific field and specific validation rule, just add a `messages()` method into your `FormRequest` class.

```
class StoreUserRequest extends FormRequest
{
    public function rules()
    {
        return ['name' => 'required'];
    }

    public function messages()
    {
        return [
            'name.required' => 'User name should be real name',
        ];
    }
}
```

Tip 91. Validate Images Dimensions

In validation, you can check uploaded images dimensions, specifying rules for min/max witdh/height, and even ratio.

```
'avatar' =>
'dimensions:min_width=100,min_height=200,dimensions:3/2'
```

Tip 92. Carbon with Only Hours/Minutes

if you want to have a current date without seconds and/or minutes, use Carbon's methods like setSeconds(0) or setMinutes(0).

```
// 2020-04-20 08:12:34
echo now();

// 2020-04-20 08:12:00
echo now()->setSeconds(0);

// 2020-04-20 08:00:00
echo now()->setSeconds(0)->setMinutes(0);

// Another way - even shorter
echo now()->startOfHour();
```

Tip 93. Good Old SQL Query

If you need to execute a simple SQL query, without getting any results - like changing something in DB schema, you can just do `DB::statement()`.

```
DB::statement('DROP TABLE users');
DB::statement('ALTER TABLE projects AUTO_INCREMENT=123');
```

Tip 94. Prepare for Validation

If you want to modify some field before default Laravel validation, or, in other words, "prepare" that field, guess what - there's a method `prepareForValidation()` in `FormRequest` class:

```
protected function prepareForValidation()
{
    $this->merge([
        'slug' => Illuminate\Support\Str::slug($this->slug),
    ]);
}
```

Tip 95. Rename Pivot Table

If you want to rename "pivot" word and call your relationship something else, you just use `as('name')` in your relationship:

```
public function podcasts() {
    return $this->belongsToMany('App\Podcast')
        ->as('subscription')
        ->withTimestamps();
}

// Then somewhere in Controller...
$podcasts = $user->podcasts();
foreach ($podcasts as $podcast) {
    // instead of $podcast->pivot->created_at ...
    echo $podcast->subscription->created_at;
}
```

Tip 96. Similar Relationship but With Condition

If you notice that you use same relationship often with additional "where" condition, you can create a separate relationship method.

```
// app/Post.php model
public function comments()
{
    return $this->hasMany(Comment::class);
}

public function approved_comments()
{
    return $this->hasMany(Comment::class)->where('approved', 1);
}
```

Tip 97. Never Update the Column

If you have DB column which you want to be set only once and never updated again, you can set that restriction on Eloquent Model, with a mutator:

```
class User extends Model
{
    public function setEmailAttribute($value)
    {
        if ($this->email) {
            return;
        }

        $this->attributes['email'] = $value;
    }
}
```

Tip 98. Maintenance Mode

If you want to enable maintenance mode on your page, execute the down Artisan command:

```
php artisan down
```

Then people would see default 503 status page.

You may also provide flags:
- message that would be shown
- retry page reload every X seconds
- still allow the access to some IP address

```
php artisan down --message="Upgrading Database" --retry=60
--allow=127.0.0.1
```

When you've done the maintenance work, just run

```
php artisan up
```

Tip 99. Find Many

Eloquent method find() may accept multiple parameters, and then it returns a Collection of all records found, not just one Model.

```
// Will return Eloquent Model
$user = User::find(1);

// Will return Eloquent Collection
$users = User::find([1,2,3]);
```

Tip 100. Stop on First Validation Error

By default, Laravel validation errors will be returned in a list, checking all validation rules. But if you want the process to stop after the first error, use validation rule called `bail`:

```
$request->validate([
    'title' => 'bail|required|unique:posts|max:255',
    'body' => 'required',
]);
```

To be continued…

Follow @DailyLaravel on Twitter for updates

Or subscribe to our weekly newsletter:
http://bit.ly/laravel-newsletter