What’s New in Laravel 5.5!

Recently, the team Laravel has announced the new series of Laravel 5.5 on its official page. Though, Laravel 5.5 is not yet out, but the documentations are live now.

So here we update you with the 5.5’s powerful features and new exciting changes that will increase security as well as productivity at the same time.

Take a quick look at the new upcoming 5.5 release.

Excel with PHP 7.0+

PHP 7 incorporates speed improvements which will diminish CPU stack significantly. So this is definitely the most important upgradition for those who are using Laravel 5, as it makes things even easier and speeds up database migration with no data loss.

Whoops

Whoops – an error PHP handler framework, used by Laravel 4, was evacuated with the arrival of Laravel 5.0. But, now it is coming back with Laravel 5.5. Whoops actually helps in making the irritating PHP blunders/errors, a little less irritating by changing the way they appear.

 # “vendor:publish” Prompt You select a provider or tag

When you run the vendor:publish command in the old version of Laravel, it will publish only migrations, views, configs and other different resources of all vendors. But in 5.5, running the specific command will make easier for you to publish a provider or tag, you want. In fact,

You can also bypass the particular prompt by determining the following command–all’ or ‘–provider’ flag with thepublish`

Email Themes

This new version of Laravel includes the ability to directly indicate a custom theme to Mailable classes, so that you can make a CSS style-sheet like the following:

```
touch resources/views/vendor/mail/html/themes/batman.css
```

After that, in your mailable class, indicate this specific filename as ropa perty.

```
class SendInvoice extends Mailable
{
    protected $theme = 'batman';
    ...
}
```

Render Mailables to the Browser

Testing your email templates crosswise email clients can be annoying. But, Laravel 5.5 has included the facility to directly show them through your routes and allow them to make instant changes. Make a Mailable like the following:

```
php artisan make:mail UserWelcome --markdown=emails.user.subscription.canceled
```

After that, render it by route:

```
Route::get('/no/way', function () {
    return new App\Mail\UserSubscriptionCanceled();
});
```

Fresh Migrations

5.5 includes a new Artisan command to the migrate: namespace, which is very much alike to migrate:refresh. This command will drop all tables and relocates them from beginning instead of rolling back your current migrations.

Automatic Package Discovery

In the oldest versions of Laravel, for packages, required to register its service providers as well as including aliases. But in 5.5, it includes the ability to automatically register service providers and include aliases by the package’s composer.json file like the following:

```
"extra": {
    "laravel": {
        "providers": [
            "The\\Dark\\Knight\\BatmanServiceProvider"
        ],
        "aliases": {
            "Bar": "The\\Dark\\Knight\\Batman"
        }
    }
}
```

Front-end Presets

As we all know, Laravel incorporates some CSS and JavaScript framework to help quicken coding the bare essential. In spite of the fact that you could expel them and begin over with your own inclinations, and the suggestion was just constrained to the Vue framework.
But, Laravel 5.5 presents three front-end presets, including Bootstrap, Vue, React and an option pick your own. With Vue, you can modify the preset by utilizing the following command:

```
php artisan preset reacts
```

You can modify respond in the above-mentioned command to bootstrap, vue or none in view of your inclination.

Improvement in Error Pages Design

There are small changes has been done in the design of the error pages: 404 or 50* in Laravel 5.5. And, some design increments with Flexbox which gets the error message fixated on the page.

Before 5.5:

In 5.5:

Customer Error Reporting

Laravel 5.5 includes bolster for characterizing a report method on any custom exemption. In previous version of Laravel, you required to check in the Handler class' report method, whether the specific exemption was tossed or not. Before you were performing the things something like the following:

```
 ...

class Handler extends ExceptionHandler
{
    ...

    public function report(Exception $exception)
    {
        if ($exception instanceof CustomException) {
            // Send email
        }

        parent::report($exception);
    }
```

```
}
```

However, with 5.5, you can remove this and enlist the report method in your custom exemption class. Laravel verifies whether there is a report method in your exemption class, if it does: calls it.

Streamlined Request Validation

To Request feature, Laravel 5.5 accompanies 2 changes.

1) You would now be able to straightforwardly call the validate method on your Request instance. So that you can call the validator on your Request instance, instead of utilizing the controller validator. You never again need to pass the request as the first argument to the validator. See the example below:

```
...

public function store()
{
    request()->validate([
        'title' => 'required',
        'body' => 'required'
    ]);

    return Post::create(request(['title', 'body']));
}
```

The second alteration you can make is that the validator restores the request information which you can store in a variable and pass on to the make technique for the model.

```
...

public function store()
{
    $post = request()->validate([
        'title' => 'required',
        'body' => 'required'
    ]);

    // $data = request()->only('title', 'body');

    return Post::create($post);
}
```

You should be cautious with this since the information returned by the validator will just contain the fields characterized in the rules. This includes a few security, however you can lose information if a few fields were not characterized rules for. To stay away from this trap, you can include the field with a void lead like the following:

```
public function store()
{
    $post = request()->validate([
        'title' => 'required',
        'body' => 'required',
        'fieldWithNoRules' => '',
        'andAnotherOne' => ''
    ]);

    // $data = request()->only('title', 'body');

    return Post::create($post);
}
```

Exception Helper Functions

throw_if and throw_unless are two new helper functions in Laravel 5.5 that help you do exactly or throw exceptions more elegantly. However, if you throw an exception, depend on a condition then these may enable you to diminish a contingent block to a solitary line. They both acknowledge 3 arguments with the 3rd being optional:

- Boolean

- Exception Class

3) An Exception message passed, if you didn’t pass with the instantiation of the exception in the 2nd argument. If the boolean is positive throw_if throws the exception and if the boolean is negative throw_unless throws the exception. See the examples below:

/ For `throw_if:

```
$foo = true;
throw_if($foo, new BarException('Foo is true'));
// or
throw_if($foo, BarException::class, 'Foo is true');

// For `throw_unless:

$phoo = false;
throw_unless($phoo, new BazException('Phoo is false'));
// or
throw_unless($phoo, BazException::class, 'Phoo is false');
```

Custom Validations Rules (CVR)

Though nothing is new with custom validation rule feature, but Laravel 5.5 facilitate you to handle the validation. To characterize a specific feature CVR, you have to make a class with two strategies, such as ‘passes’ and ‘message’.
However, you can put this class anywhere in the App\Rules namespace. The passes strategy acknowledges 2 contentions, such as ‘attribute’ and ‘value’, which you can utilize to confirm the field.

```
<?php

namespace App\Rules;

use Illuminate\Contracts\Validation\Rule;

class CustomRule implements Rule
{
    /**
     * Determine if the validation rule passes.
     *
     * @param  string  $attribute
     * @param  mixed  $value
     * @return bool
     */
    public function passes($attribute, $value)
    {
         // must return true or false for the validation to
pass or fail
    }
```

```
    /**
     * Get the validation error message.
     *
     * @return string
     */
    public function message()
    {
        // return a string here for the failing condition
    }
}
```

Your CVR should actualize the Laravel's Illuminate\Contracts\Validation\Rule contract.

You can utilize this CVR anywhere in the controller validator or in a form request class or the validator from the Request instance. If you're utilizing a custom rule, you cannot pass a string with rules isolated by a comma. You have to pass each rule as a solitary component assembled in an exhibit the following way:

```
$request->validate([
    'someField' => [
        'required', 'min:4', new CustomRule()
    ]
]);
```

Model Factory Generators

This new version of Laravel facilitates you to easily generate model factories with a brand new Artisan command known as make:factory. See the example below:

```
php artisan make:factory PostFactory
```

This will create a new file in the database/factories folder known as PostFactory.php. Not only this, it also allows to generate a factory while creating a model

```
php artisan make:model Post -f
```

In fact, You can also pass the flag like -c to include a controller and -m to include a migration, as it will help in rapidly throwing together an asset.

This is all about the new series of Laravel 5.5. Now we hope this guide or above mentioned 5.5 enhancements will get you prepare for the upcoming

goodness.

Happy Coding!