



Laravel Requests & Validation

PHP WebDevelopment 2019

Milena Tomova
Vratsa Software

<https://vratsasoftware.com/>

Table of Contents



1. Request Path & Method

2. Retrieving Input

3. Old Input

4. Validation - Introduction

5. Validate form data in the controller

6. Custom Request

7. Validation rules

8. Validation Error messages

9. Custom Validation Error Messages

10. Session. Flash Messages

Learn to Search in Internet

- The course assignments require to search in Internet
 - This is an important part of the learning process
 - Some exercises intentionally have no hints
- Learn to find solutions!
 - Software development includes everyday searching and learning
 - No excuses, just learn to study!
 - Developers learn new technologies, tools, languages every day!





Validate Form Data in the Controller

Task 1

Validate the data before creating or updating a hall.



ValidatesRequests

ValidatesRequests

Laravel's base controller class uses a **ValidatesRequests** trait which provides a convenient method to validate incoming HTTP requests with a variety of powerful validation rules

validate()

- **validate()** accepts the incoming request and a set of validation rules as parameters
 - If **the data** passes the validation rules - the next lines of code are executed
 - else Laravel returns error messages for the failed validation rules

validate()

store new hall in the database

```
public function store(Request $request)
```

```
{
```

```
    $this->validate($request, [
```

```
        'hall_name' => 'required|unique:halls|max:100',
```

```
    ]);
```

```
    // The hall data is valid, store new hall in  
    database...
```

```
}
```

form data

form field
name to
validate

a set of validation ru



Custom Requests

Task

Validate the data before adding a Level to a course.



For more complex validation scenarios, you may wish to create a "form request"

- Form requests are custom request classes that contain validation logic
- To create a form request class, use

```
php artisan make:request AddLevelToCourseRequest
```

This command creates a new class in the `app\Http\Requests` folder

step 1 Determine if the user is authorized to make this request.

```
public function authorize()  
{  
    return true;  
}
```

set to true, to allow all users to make this request

... or add more complex authorization logic here
[see the documentation](#)

step 2 Add validation rules that will apply to the request

```
public function rules()  
{  
    return [  
        'level' => 'required|max:100'  
    ];  
}
```

rules will be
checked in the order
they are listed in the
rules()

for the available
validation rules
[see the documentation](#)

step 3 Inject the custom request in the controller's data-receiving method

```
public function store_level_to_course( AddLevelToCourseRequest $request, Course $course )  
{  
    Level::create([  
        'level_name' => $request->level,  
        'course_id'  => $course->id  
    ]);  
    return redirect()->route('courses.levels_list', $course->id );  
}
```

base Request class is
replaced by the custom
request class



Validation Errors

Validation Errors

Validation Errors

If validation fails

Laravel automatically redirects back.

All errors are put in an instance of `Illuminate\Support\MessageBag` - `$error /errorBag/` available to all views

Access all validation errors in a view

```
@if ($errors->any())  
    <div class="alert alert-danger">  
        <ul>  
            @foreach ($errors->all() as $error)  
                <li>{{ $error }}</li>  
            @endforeach  
        </ul>  
    </div>  
@endif
```

available \$error methods
[see documentation](#)

Access specific error if exists

```
@if($errors->has('level'))  
    <div class="col-sm-7 col-sm-offset-1 text-danger">  
        {{ $errors->first('level') }}  
    </div>  
@endif
```

display the error message of
the first failed validation
rule



Custom Error Messages

Custom Error Messages

You may set *custom error messages* for validation instead of the defaults.
Add *messages method* in the custom request class

```
public function messages() {
```

```
  return [
```

```
    'level.required' => 'Полето :attribute е задължително!',
```

```
    'level.min'      => 'Въведете минимум :min знака!',
```

```
    'level.max'      => 'Въведете максимум :max знака!',
```

```
  ];
```

```
}
```

will display the
field name

to display the
min value
dinamically

list rules for
every field



Sessions Flash Messages

Sessions. Flash Messages.

Sessions

for using sessions in Laravel
[see documentation](#)

Sessions. Flash Messages.

Flash Messages

Messages stored in a session for the next request only.

Often used to inform user for successful action -

- inserting, updating or deleting data.

Sessions. Flash Messages.

Flash a message in the controller

- *redirecting to a route with a flash message*

```
public function my_function()
{
    return redirect()->route('route_name')->with('success_message', 'Another flash message');
}
```

the name of the message to look for in the MessageBag

the message text

Sessions. Flash Messages.

Flash a message in the controller

- *using Session facade - for more complex logic*

```
Session::flash('message', 'This is a message!');
```

the name of the
message to look
for in the
MessageBag

the message text

Sessions. Flash Messages.

- *Display* the flash message in the view

```
@if(Session::has('success_message'))  
    {{Session::get('success_message')}}  
@endif
```

Check if expected message
has been set

Display the
message

Sessions. Flash Messages.

Flash a message in the controller

- using Session facade - *for more complex logic, multiple messages*

```
Session::flash('message', 'This is a message!');  
....  
Session::flash('alert-class', 'alert-danger');
```

Sessions. Flash Messages.

- *Display* the flash message in the view

```
@if(Session::has('message'))  
<p class="alert  
{{ Session::get('alert-class') }}">  
    {{Session::get('message') }}  
</p>  
@endif
```


Questions?



Гнездото
Coworking

Цялостен
курс по
програми
ране

Дизайн
курс

Курс по
дигит.
маркетинг

MindHub



Partners



**Telerik
Academy**



MindHub

ПРОМЯНАТА

Trainings @ Vratsa Software



- Vratsa Software – High-Quality Education, Profession and Jobs
 - www.vratsasoftware.com
- The Nest Coworking
 - www.gnezdoto.vratsasoftware.com
- Vratsa Software @ Facebook
 - www.fb.com/VratsaSoftware
- Slack Channel
 - www.vso.slack.com

