✦ Member-only story

Creating A Popup/Modal With Laravel Livewire And No jQuery

Mark Caggiano · Follow

5 min read · Feb 17, 2021

Hello everybody !

Today we will create a modal with Laravel Livewire and without jQuery, actually without javascript at all. How cool is Livewire?

We will use Laravel 8 and Bootstrap 4, but feel free to use whatever CSS Framework you want, from Tailwind CSS to Bulma… It doesn't matter. Actually you could just use pure CSS.

First we need to install livewire, you might already have it…

```
$ composer require livewire/livewire
```

Now let's add our Modal Component:

```
$ php artisan make:livewire Modals/MyExampleModal
```

If you want to go deeper on how to configure livewire, you should read this tutorial: https://laravel-livewire.com/docs/2.x/quickstart

Now we should have something like this, it might be a little different, but it doesn't matter:

```
<?php

namespace App\Http\Livewire\Modals;

use Livewire\Component;

class MyExampleModal extends Component
{

    public function render()
    {
        return view('livewire.modals.my-example-modal');
    }
}
```

Now let's open our modal blade file, it should be in **resources/views/livewire/modals/my-example-modal.blade.php**. This file should just contain an empty div, so let's copy the modal html from bootstrap. I made a simple modal for now and paste it in the blade file:

```
<div>
    <div class="modal fade" id="myExampleModal" tabindex="-1"
role="dialog" aria-labelledby="exampleModalLabel"
         aria-hidden="true">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title"
id="exampleModalLabel">Modal Title?</h5>
                    <button class="close" type="button" aria-
label="Close">
                        <span aria-hidden="true">×</span>
                    </button>
                </div>
                <div class="modal-body">Modal Content</div>
                <div class="modal-footer">
                    <button class="btn btn-secondary"
type="button">Cancel</button>

                    <button class="btn btn-secondary"
type="button">Do Something</button>
                </div>
            </div>
        </div>
    </div>
    <!-- Let's also add the backdrop / overlay here -->
    <div class="modal-backdrop fade show"
         id="backdrop"
         style="display: none;"></div>
</div>
```

Let's remove all the **data-dismiss="modal"** and all the **JS Bootstrap Stuff, we don't need it**, actually we don't need JS at all. Everything will be handled by Livewire.

You could also remove jQuery from your project if you want.

This is why we added the backdrop / overlay inside our component. Bootstrap will not handle it, so we will.

Now let's usk us a question: "How i will show up the modal with just livewire?".

If you were using just Javascript you would set **myModalExample display to block**, when you want to show the modal, and add the class "show".

We also need to set the **modal backdrop** to **display: block;**

When we want to hide the modal, just change **display: none**, on both **myModalExample** and **modal backdrop.**

So let's just do it in Livewire, open our PHP Component Logic, and add this code:

```
<?php

namespace App\Http\Livewire\Modals;

use Livewire\Component;

class MyExampleModal extends Component
{
    public $show;

    public function mount() {
        $this->show = false;
    }

    public function doShow() {
        $this->show = true;
    }

    public function doClose() {
        $this->show = false;
    }

    public function doSomething() {
        // Do Something With Your Modal

        // Close Modal After Logic
        $this->doClose();
```

```
    }

    public function render()
    {
        return view('livewire.modals.my-example-modal');
    }
}
```

And let's add our **blade logic** in **my-example-modal.blade.php**:

```
<div>
    <div class="modal fade @if($show === true) show @endif"
         id="myExampleModal"
         style="display: @if($show === true)
                 block
         @else
                 none
         @endif;"
         tabindex="-1"
         role="dialog"
         aria-labelledby="exampleModalLabel"
         aria-hidden="true">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title"
id="exampleModalLabel">Modal Title</h5>
                    <button class="close"
                            type="button"
                            aria-label="Close"
                            wire:click.prevent="doClose()">
                        <span aria-hidden="true">×</span>
                    </button>
                </div>
                <div class="modal-body">Modal Content</div>
                <div class="modal-footer">
                    <button class="btn btn-secondary"
                            type="button"

wire:click.prevent="doClose()">Cancel</button>

                    <button class="btn btn-secondary"
                            type="button"
                            wire:click.prevent="doSomething()">Do
Something</button>
                </div>
```

```
                </div>
            </div>
        </div>
        <!-- Let's also add the backdrop / overlay here -->
        <div class="modal-backdrop fade show"
             id="backdrop"
             style="display: @if($show === true)
                        block
             @else
                        none
             @endif;"></div>
    </div>
```

Our logic is clear… When we click call doShow() we set **show** on **true**, and the correct display and CSS classes we will show up.

When we call **doClose**, the CSS classes will be removed and **display** set to **none**; **All of this without a single line of JS !** How's **cool** and **fast** is that ?

It saved tons and tons of time and also **NO JQUERY**. It is so light and fast on the client side. I like it very much.

How can i open the modal on click / event ?

Ok now i know the logic to show/close the modal, but i need to link it to some event, like a **button click** or something like that, **how i do that ?**

Fortunately Livewire has a mechanism called "**Events Emission**", so we can "Emit" events and "Listen" to them.

We will just add a **custom event** and a **listener**, which we will **trigger** a **component function** that will **show up the modal**, or do whatever we want.

Let's also put a button **outside** our componet that will trigger a modal show up/close.

Let's open our **welcome.blade.php** and put some buttons to play with and let's also add include our modal:

```
<!-- welcome.blade.php -->

<html>
<head>
<title>Livewire Modal Example</title>

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" integrity="sha384-
JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z"
crossorigin="anonymous">

@livewireStyles
</head>
<body>
<div>
    <button class="btn btn-primary btn-lg"
            type="button"
            wire:click.prevent="$emit('showModal', 'SomeData')"
>Open Modal</button>
</div>

<!-- We need to include our modal using livewire, or our
html markup we will not be loaded -->

<livewire:modals.my-example-modal />

@livewireScripts

</body>
</html>
```

We use the special method **$emit**, first parameter is the event we want to emit, and the other parametes are data we want to pass to our receving component, in our case our modal. It can be anything, from Eloquent Models, to JSON, to Object or to Simple Strings.

Now let's write our receiving logic, in our modal:

```
<?php

namespace App\Http\Livewire\Modals;

use Livewire\Component;

class MyExampleModal extends Component
{
    public $data;
    public $show;

    protected $listeners = ['showModal' => 'showModal'];

    public function mount($data) {
        $this->data = $data;
        $this->show = false;
    }

    public function showModal($data) {
        $this->data = $data;

        $this->doShow();
    }

    public function doShow() {
        $this->show = true;
    }

    public function doClose() {
        $this->show = false;
    }

    public function doSomething() {
        // Do Something With Your Modal

        // Close Modal After Logic
        $this->doClose();
    }

    public function render()
    {
        return view('livewire.modals.my-example-modal');
    }
}
```

We listen to **showModal**, we get and set our data from $emit, in this case $this->data we will contain “SomeData” and we show up our modal with doShow();

Now we can also display our data in our modal, if we wish:

```
<div>
    <div class="modal fade @if($show === true) show @endif"
         id="myExampleModal"
         style="display: @if($show === true)
                 block
         @else
                 none
         @endif;"
         tabindex="-1"
         role="dialog"
         aria-labelledby="exampleModalLabel"
         aria-hidden="true">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title"
id="exampleModalLabel">Modal Title</h5>
                    <button class="close"
                            type="button"
                            aria-label="Close"
                            wire:click.prevent="doClose()">
                        <span aria-hidden="true">×</span>
                    </button>
                </div>
                <div class="modal-body">
                    Modal Content:
                    <br>
                    {{-- The Data From The $emit Will Show Up Here -
-}}
                    {{$data}}
                </div>
                <div class="modal-footer">
                    <button class="btn btn-secondary"
                            type="button"

wire:click.prevent="doClose()">Cancel</button>

                    <button class="btn btn-secondary"
                            type="button"
```

```
                                wire:click.prevent="doSomething()">Do
Something</button>
                </div>
            </div>
        </div>
    </div>
    <!-- Let's also add the backdrop / overlay here -->
    <div class="modal-backdrop fade show"
         id="backdrop"
         style="display: @if($show === true)
                 block
         @else
                 none
         @endif;"></div>
</div>
```

And we are done !

Now there is no limit on what you can do. You could put a form in a modal, pass complex data and much more.

This is just a basic example on how it works.

If you liked, please clap, there is so much work behind these articles.

Thanks,

Marco Caggiano.

Laravel Livewire PHP Bootstrap 4 Php Developers

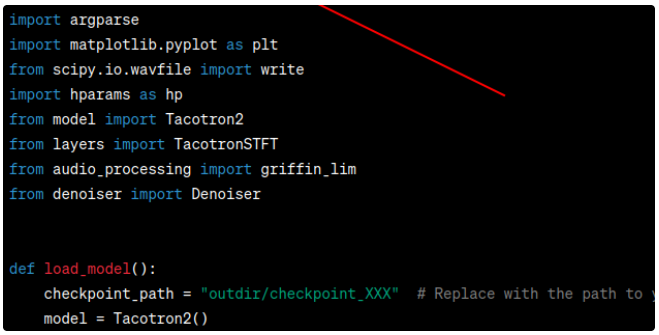
Written by Mark Caggiano

204 Followers

Internet Marketer, Web Developer, Traveler

Follow

More from Mark Caggiano



Mark Caggiano

Step-by-step tutorial on how to clone your voice using python and...

Cloning a voice using Python and AI involves training a model on your voice recordings an...

✨ · 4 min read · May 9

47 3

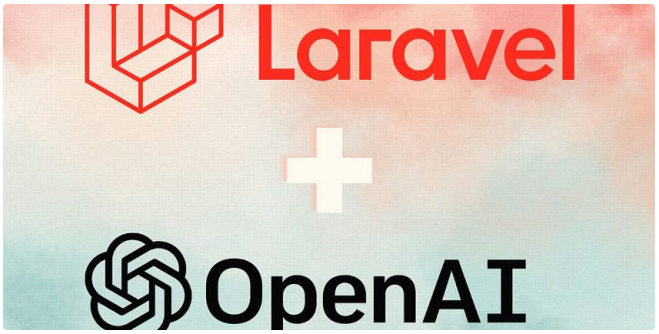
Mark Caggiano

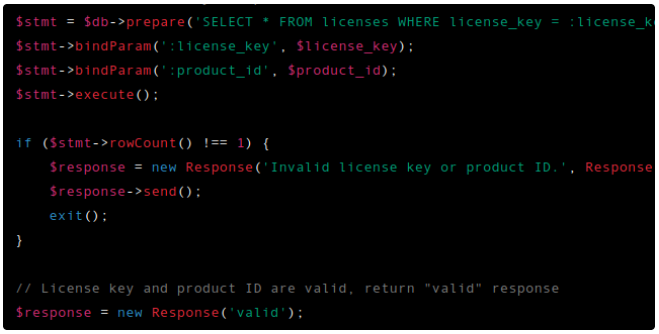
Centralizing Users Authentication For Multiple Laravel Apps Using...

Centralizing users for multiple Laravel apps can be achieved using a Single Sign-On (SS...

✨ · 4 min read · Mar 23

38 2





Mark Caggiano

How To Connect Laravel to ChatGPT API

Step 1: Create a new Laravel project

2 min read · Mar 4

58 2

Mark Caggiano

How To Create A Licensing System Using PHP, Composer and OOP fo...

Here is how to create a licensing system using PHP, Composer and OOP for a custom code...

6 min read · Mar 10

13 3

See all from Mark Caggiano

Recommended from Medium

Lorenzo Casponi in Stackademic

Is PHP Worth Learning in 2023–2024?

Hi all, today i wanna expose my two cents about a question that i've received tons of...

7 min read · Aug 20

74 3

Idoko Emmanuel

User Authentication for Web and API with Laravel Fortify and...

Welcome to the second phase of our article, where we delve deeper into harnessing the...

12 min read · Jul 28

13 1

Lists

Staff Picks

544 stories · 578 saves

Stories to Help You Level-Up at Work

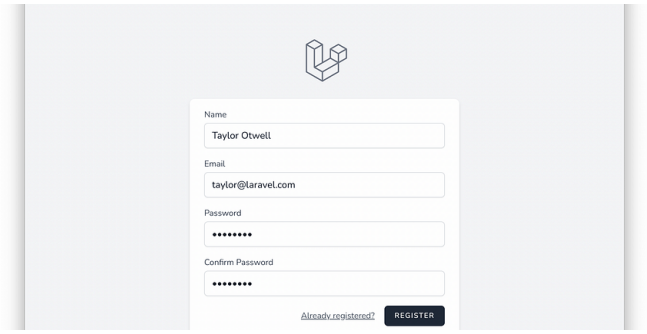
19 stories · 386 saves

Self-Improvement 101

20 stories · 1110 saves

Productivity 101

20 stories · 1014 saves



Andre Elmustanizar

What is Laravel Breeze and How to Install

Hello everyone, in this article I will explain and make a tutorial on how to install Laravel...

3 min read · Jul 22

Mursaleen Ahmad

Laravel-10 MultiAuth with JetStream and Livewire — Creatin...

Hello! Today we will implement multiauth in Laravel-10 using Jetstream and Livewire.

7 min read · Jul 22

6 11 3

Oliver Samuel

Top 10 Laravel Packages You Should Know in 2024

Laravel, the sleek and expressive PHP framework, continues to enable developers t...

4 min read · Dec 18

166 1

Laravel Pro Tips

Easy Guide to Laravel Breeze: User Areas for Student, Teacher, &...

Ready to set up distinct areas for students, teachers, and admins in Laravel Breeze?...

10 min read · 5 days ago

4

See more recommendations