8- Creando Departamento

Post anterior.

Empezaremos trabajando con **Axios**., la cual es una librería de **JavaScript** construida con el objetivo de gestionar la programación asíncrona con promesas, un cliente HTTP, una de sus características es que nos permite realizar peticiones XMLHttpRequest (Ajax) desde el navegador de una manera sencilla. Lo primero es repasar el modal en unas lineas en particular

```
<a class="button is-success" @click="createDeparture()" v-if="modalDeparture==1">Aceptar</a>
```

Tenemos este botón que es el botón aceptar que aparece cuando se abre el modal. ¿Porque sabemos que es este botón? por el **v-if**. Ya hemos visto en el post anterior que en **openmodal** asignamos el valor 1 a **modalDeparture** al llegar a **departure / create.**

Este llama a una función que tenemos vacía **createDeparture**.

En esta ocasión iremos agregando código a la función paso a paso

Las primeras lineas sera de una validación simple

```
if (this.titleDeparture == '') {
    this.errorTitleDeparture = 1;
    return;
}
```

En el input del modal tenemos un **v-model** asociado a la variable **titleDeparture**. Esta variable se actualiza automáticamente a medida que el usuario escribe. Por lo que aquí valoramos si al pulsar el botón la variable esta vacía. No es necesario nada mas. Ni interactuar con el DOM ni identificar el input y recoger el valor. Simplemente trabajamos directamente con esa variable asociada.

La validación es básica. Si la variable esta vacía otra variable (**errorTitleModal**) se pone a uno. ¿Que pasa cuando esa variable se pone a 1?. Repasemos el código del modal.

```
<div v-show="errorTitleDeparture" class="columns text-center">
    <div class="column text-center text-danger">
        El nombre del Departamento no puede estar vacío
    </div>
</div>
```

v-show es parecido a **v-if** pero es mas simple. Actúa con 0 o 1 (mas adelante veremos que **v-if** es mas completo e incluso permite **else**). En resumen **errorTitleDeparture** con valor 1 mostrara el mensaje de error

Inmediatamente después del if pondremos la siguiente linea

```
let me = this;
```

Esta linea lo único que hace es asociar **this** a una variable llamada **me**. Esto evita que dentro del código **Axios** al usar **this**referenciamos al objeto **Axios** y no al objeto **Vue**. Dentro de **Axios** el **this** de **Vue** se llamara **me**.

EL primer código **Axios** va inmediatamente después

```
axios.post('{{route('departurecreate')}}', {
            'title': this.titleDeparture
        })
    .then(function (response) {
        me.titleDeparture = '';
        me.errorTitleDeparture = 0;
        me.modalDeparture = 0;
        me.closeModal();
    })
    .catch(function (error) {
        console.log(error);
    });
```

A primera vista la mayoría de cosas se entiende para quien ya ha hecho llamadas Ajax. Revisemos las lineas.

```
axios.post('{{route('departurecreate')}}', {
    'title': this.titleDeparture
})
```

En post anteriores hemos creado la ruta

```
Route::post('/departure/create','DepartureController@create')->name('departurecreate');
```

axios.post indica el protocolo que debe coincidir con el protocolo de la ruta.

{{ route('departurecreate') }} es la manera en **Laravel** de poner la ruta asociada. No es una buena practica mezclar código Php con JS, pero para la serie nos sirve para agilizar.

luego hacemos **, { 'title' : this.titleDeparture }** aquí estamos enviando el valor de la variable **titleDeparture** dentro de la variable **title** del request.

Que según el código que ya tenemos en el controlador es lo que necesitamos que llegue. Lo repetimos aquí:

```
public function create(Request $request){
       $departure=new Departure();
       $departure->title=$request->title;
       $departure->save();
}
```

la clave esta en

```
$departure->title=$request->title;
```

Sigamos con el código de Axios

```
.then(function (response) {
        me.titleDeparture = '';
        me.errorTitleDeparture = 0;
        me.modalDeparture = 0;
        me.closeModal();
    })
```

.then es cuando las cosas han ido bien y no ha habido error.

En esta parte creamos las variables de departure: **titleDeparture** / **errorTitleDeparture** / **modalDeparture** y luego usamos la función **closeModal** para cerrar los valores del modal.

Aqui es donde cambiamos **this** y usamos **me** para evitar confusiones en el código

La ultima parte del código de **Axios** es

```
.catch(function (error) {
    console.log(error);
});
```

.catch es lo que sucede cuando hay un error. Por ahora y porque aun tenemos bastante por delante para dejar funcional el código el error de **create** lo dejaremos así. Si surge un error lo veremos en la consola.

Si hasta aqui se han seguido todos los pasos correctamente tendríamos que tener el valor en la base de datos.

A partir del siguiente post iremos tocando la magia de Vue para recuperar esa información y mostrarla en el front y lo sencillo que es tener dicha información actualizada.

Código: Github

Aclaración: En el Github aparece un cambio en el controlador que no esta aquí ya que el código del post esta correcto pero ha sido despiste ya corregido.

Siguiente post- Recuperando la información