

Evaluando nuestros formularios

índice

1. [Express Validator](#)
2. [JavaScript front-end](#)

1 | Express Validator

En el archivo de **validación**

{ }

```
const { check } = require('express-validator');
module.exports = [
  check('name')
    .not()
    .isEmpty()
    .withMessage('Ingresar un nombre'),
  check('email')
    .isEmail()
    .withMessage('Ingresar un email válido'),
  check('password')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
  check('repassword')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
]
```

En el archivo de **validación**

{ }

```
const { check } = require('express-validator');  
module.exports = [  
  check('name')  
    .not()  
    .isEmpty()  
    .withMessage('Ingresar un nombre'),  
  check('email')  
    .isEmail()  
    .withMessage('Ingresar un email válido'),  
  check('password')  
    .isLength({min: 8})  
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),  
  check('repassword')  
    .isLength({min: 8})  
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),  
]
```

Requerimos express-validator.

En el archivo de **validación**

```
const { check } = require('express-validator');
module.exports = [
  check('name')
    .not()
    .isEmpty()
    .withMessage('Ingresar un nombre'),
  check('email')
    .isEmail()
    .withMessage('Ingresar un email válido'),
  check('password')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
  check('repassword')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
]
```

Indicamos qué campo del req.body queremos chequear.

{}

En el archivo de **validación**

```
const { check } = require('express-validator');
module.exports = [
  check('name')
    .not()
    .isEmpty()
    .withMessage('Ingresar un nombre'),
  check('email')
    .isEmail()
    .withMessage('Ingresar un email válido'),
  check('password')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
  check('repassword')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
]
```

Indicamos qué es lo que queremos chequear.

{}

En el archivo de **validación**

```
const { check } = require('express-validator');
module.exports = [
  check('name')
    .not()
    .isEmpty()
    .withMessage('Ingresar un nombre'),
  check('email')
    .isEmail()
    .withMessage('Ingresar un email válido'),
  check('password')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
  check('repassword')
    .isLength({min: 8})
    .withMessage('La contraseña debería tener un mínimo de 8 caracteres'),
]
```

En caso de cumplirse la condición, dejamos un mensaje indicando cuál es el error.

{}

En el controller

{}

```
const { validationResult } = require('express-validator');

module.exports = {
  registrarNuevoUsuario : function(req, res){
    let resultadoValidacion = validationResult(req);
    if(resultadoValidacion.errors.length < 0){
      db.Usuario.Create({
        // ...
      })
      .then(function(){
        res.render('home')
      })
    }else{
      res.render('registro', {errores: resultadoValidacion.errors})
    }
  }
}
```

En el controller

```
module.exports = {  
  registrarNuevoUsuario : function(req, res){  
    let resultadoValidacion = validationResult(req);  
    if(resultadoValidacion.errors.length < 0){  
      db.Usuario.Create({  
        // ...  
      })  
      .then(function(){  
        res.render('home')  
      })  
    }else{  
      res.render('registro', {errores: resultadoValidacion.errors})  
    }  
  }  
}
```

Creamos una variable que almacene lo que devuelva el método **validationResult(req)**

En el **controller**

```
module.exports = {  
  registrarNuevoUsuario : function(req, res){  
    let errores = validationResult(req);  
  
    if(resultadoValidacion.errors.length < 0){  
      db.Usuario.Create({  
        // ...  
      })  
      .then(function(){  
        res.render('home')  
      })  
    }else{  
      res.render('registro', {errores: resultadoValidacion.errors})  
    }  
  }  
}
```

Preguntamos si esa variable está vacía. De ser así, ejecutamos la lógica de nuestro controlador.

En el controller

{}

```
module.exports = {  
  registrarNuevoUsuario : function(req, res){  
    let errores = validationResult(req);  
    if(errores.isEmpty()){  
      db.Usuario.Create({  
        // ...  
      })  
      .then(function(){  
        res.render('home')  
      })  
    } else {  
      res.render('registro', {errores: resultadoValidacion.errors})  
    }  
  }  
}
```

De lo contrario, volvemos a renderizar la vista que contenga nuestro formulario, y le pasamos un objeto con los errores.

En la vista

```
{  
  <form action="/form" method="POST" enctype="multipart/form-data">  
    <label for="">Nombre</label>  
    <input type="text" name="name">  
    <% if(typeof errores != 'undefined'){ %>  
      <% for(let i = 0; i < errores.length; i++){ %>  
        <% if(errores[i].param == "name"){ %>  
          <small class="text-danger"> <%= errores[i].msg%> </small>  
        <% } %>  
      <% } %>  
    <% } %>  
  </form>  
}
```

En la vista

```
<form action="/form" method="POST" enctype="multipart/form-data">
<label for="">Nombre</label>
<input type="text" name="name">

  <% if(typeof errores != 'undefined'){ %>
  <% for(let i = 0; i < errores.length; i++){ %>
    <% if(errores[i].param == "name"){ %>
      <small class="text-danger"> <%= errores[i].msg%> </small>
    <% } %>
  <% } %>
<% } %>
</form>
```

Preguntamos si la variable **errores** está definida.

En la vista

{}

```
<form action="/form" method="POST" enctype="multipart/form-data">
<label for="">Nombre</label>
<input type="text" name="name">
  <% if(typeof errores != 'undefined'){ %>
    <% for(let i = 0; i < errores.length; i++){ %>
      <% if(errores[i].param == "name"){ %>
        <small class="text-danger"> <%= errores[i].msg%> </small>
      <% } %>
    <% } %>
  <% } %>
</form>
```

De ser así, recorreremos el array de **errores**, ya que puede haber más de uno.

En la vista

{}

```
<form action="/form" method="POST" enctype="multipart/form-data">
<label for="">Nombre</label>
<input type="text" name="name">
  <% if(typeof errores != 'undefined'){ %>
    <% for(let i = 0; i < errores.length; i++){ %>
      <% if(errores[i].param == "name"){ %>
        <small class="text-danger"> <%= errores[i].msg%> </small>
      <% } %>
    <% } %>
  <% } %>
</form>
```

Preguntamos si algún error tiene como **parámetro** "nombre". De esta manera, identificamos que el error que vayamos a imprimir, corresponda con el input.

En la vista

{}

```
<form action="/form" method="POST" enctype="multipart/form-data">
<label for="">Nombre</label>
<input type="text" name="name">
  <% if(typeof errores != 'undefined'){ %>
    <% for(let i = 0; i < errores.length; i++){ %>
      <% if(errores[i].param == "name"){ %>
        <small class="text-danger"> <%= errores[i].msg%></small>
      <% } %>
    <% } %>
  <% } %>
</form>
```

De ser así, imprimimos una etiqueta que muestre el mensaje que dejamos para ese error.

2

JavaScript front-end

En la vista

{}

```
<form action="/form" method="POST" enctype="multipart/form-data">
  <label for="">Nombre</label>
  <input type="text" name="name">
  <div class="col-8 col-lg-10">
    <small class="text-danger erName"></small>
  </div>
  <% if(typeof errores != 'undefined'){ %>
    <% for(let i = 0; i < errores.length; i++){ %>
      <% if(errores[i].param == "name"){ %>
        <small class="text-danger"> <%= errores[i].msg%> </small>
      <% } %>
    <% } %>
  <% } %>
</form>
```

En la vista

```
<form action="/form" method="POST" enctype="multipart/form-data">
  <label for="">Nombre</label>
  <input type="text" name="name">

  <div class="col-8 col-lg-10">
    <small class="text-danger erName"></small>
  </div>
  <% if(typeof errores != 'undefined'){ %>
    <% for(let i = 0; i < errores.length; i++){ %>
      <% if(errores[i].param == "name"){ %>
        <small class="text-danger"> <%= errores[i].msg%> </small>
      <% } %>
    <% } %>
  <% } %>
</form>
```

Creamos un div para poder rellenar con HTML desde JavaScript front-end en caso de haber errores.

{}

En el archivo de **JavaScript front-end**

{}

```
window.addEventListener('load', function() {  
    let btnSubmit = qs('#btnSubmit');  
    let inputNombre = qs('#inputName');  
    let erNombre = qs('.erName');  
    let registerForm = qs('form')  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
        let errores = {}  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

En el archivo de **JavaScript front-end**

```
window.addEventListener('load', function() {  
  
    let btnSubmit = qs('#btnSubmit');  
    let inputName = qs('#inputName');  
    let erName = qs('.erName');  
    let registerForm = qs('form');  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
        let errores = {}  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

Seleccionamos los elementos que vamos a utilizar. El botón para capturar el evento. El input para validar ese campo. El div para insertar el error. El form para poder mandarlo si no hay errores.

En el archivo de **JavaScript front-end**

```
window.addEventListener('load', function() {  
    let btnSubmit = qs('#btnSubmit');  
    let inputName = qs('#inputName');  
    let erName = qs('.erName');  
    let registerForm = qs('form')  
  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
        let errores = {}  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

Capturamos el evento de clic por el cual se manda el formulario y prevenimos que se mande.

En el archivo de **JavaScript front-end**

```
window.addEventListener('load', function() {  
    let btnSubmit = qs('#btnSubmit');  
    let inputName = qs('#inputName');  
    let erName = qs('.erName');  
    let registerForm = qs('form')  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
  
        let errores = {}  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

Creamos un objeto para poder almacenar los mensajes que le queremos mandar al usuario en caso de haber errores.

En el archivo de **JavaScript front-end**

```
window.addEventListener('load', function() {  
    let btnSubmit = qs('#btnSubmit');  
    let inputName = qs('#inputName');  
    let erName = qs('.erName');  
    let registerForm = qs('form')  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
        let errores = {}  
  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

Evaluamos el largo del input. En caso de ser menor a 1, creamos una propiedad en nuestro objeto con el mensaje.

En el archivo de **JavaScript front-end**

{ }

```
window.addEventListener('load', function() {  
    let btnSubmit = qs('#btnSubmit');  
    let inputName = qs('#inputName');  
    let erName = qs('.erName');  
    let registerForm = qs('form')  
    btnSubmit.addEventListener('click', function(e){  
        e.preventDefault();  
        let errores = {}  
        if(inputName.value.length < 1){  
            errores.name = 'Este campo debe estar completo'  
        }  
  
        if(Object.keys(errores).length >= 1){  
            erName.innerText = (errores.name) ? errores.name : '';  
        } else {  
            registerForm.submit();  
        }  
    })  
})
```

Preguntamos por el largo de nuestro objeto. En caso de haber errores, rellenamos el **div** de errores con el mensaje. Si no hay errores, mandamos el formulario.

DigitalHouse>
Coding School