

```
1 import { check, validationResult } from 'express-validator'
2 import bcrypt from 'bcrypt';
3 import Usuario from '../models/Usuario.js';
4 import { generarId, generarJWT } from '../helpers/tokens.js'
5 import { emailRegistro, emailOlvidePassword } from '../helpers/emails.js'
6
7 const formularioRegistro = (req, res) => {
8   res.render('auth/registrarse', {
9     pagina : 'Crear Cuenta',
10     csrfToken: req.csrfToken()
11   });
12 }
```

Asociaciones

Relacionar Modelos y Tablas

```
25   csrfToken: req.csrfToken(),
26   errores : resultado.array(),
27   pagina : 'Crear Cuenta',
28   usuario : {
```

Juan Pablo De la torre Valdez

www.codigoconjuan.com



¿Que son Las asociaciones?



Las Asociaciones son formas de cruzar información en tu base de datos

Sequelize Soporta todos los tipos de relaciones en una base de datos: 1:1, 1:N, N:N

La forma en que lo hace es por medio de métodos que ya existen en Sequelize



Métodos

Para crear asociaciones

hasOne

belongsTo



hasMany

belongsToMany

hasOne



Es para crear Relaciones 1:1, donde un registro puede tener hasta 1 registro relacionado en otro tabla

Ejemplos: Una Propiedad tiene un Vendedor, un Usuario tiene un Perfil, un Producto tiene una Categoría

Sintaxis: `Vendedor.hasOne(Propiedad)`

En este ejemplo; Propiedad deberá tener una llave foránea que haga referencia a un Vendedor, si no se especifica, Sequelize lo va a crear



belongsTo



Al igual que hasOne es para Relaciones 1:1, donde un registro puede tener hasta 1 registro relacionado en otro tabla, la única diferencia es sintaxis

Sintaxis: `Propiedad.belongsTo(Vendedor)`

En este ejemplo; Propiedad deberá tener una llave foránea que haga referencia a un Vendedor, si no se especifica, Sequelize lo va a crear



hasMany



Es para crear Relaciones 1:N, donde un registro puede tener múltiples coincidencias o relaciones en otra tabla

Ejemplos: Un Vendedor tiene múltiples Propiedades, un Usuario tiene múltiples Posts, un Producto tiene múltiples Reviews

Sintaxis: `Vendedor.hasMany(Propiedad)`

En este ejemplo; Propiedad deberá tener una llave foránea que haga referencia a un Vendedor



Propiedad.belongsTo(Vendedor)

Vendedor.hasMany(Propiedad)



belongsToMany



Es utilizado para las relaciones N:N, en este tipo de relaciones se utiliza una tabla pivote, por lo tanto se realiza mediante 3 modelos

Sintaxis: `Estudiante.belongsToMany(Clase, { through: HorarioClase })`

En este ejemplo; Múltiples Estudiantes tendrán Múltiples Clases, por lo tanto se crea una 3er Tabla que sirve como pivote con referencias por llave foránea tanto a Estudiantes como Clases



**Las Asociaciones siempre
quedan más claras con la
práctica**

