# Laboratorio de Computación IV



Clase 15

## Repaso

- Introducción a Relaciones en ActiveRecord.
- Usamos un *helper* de *Devise* para asignar el autor del post en base al usuario logueado.
- rails\_admin Gem.

- Conjunto de datos iniciales para una aplicación.
- Se usan principalmente en desarrollo, para que todos puedan comenzar (cuantas veces sea necesario) de un mismo punto de partida.
- En rails existe un archivo `db/seeds.rb` para esto.

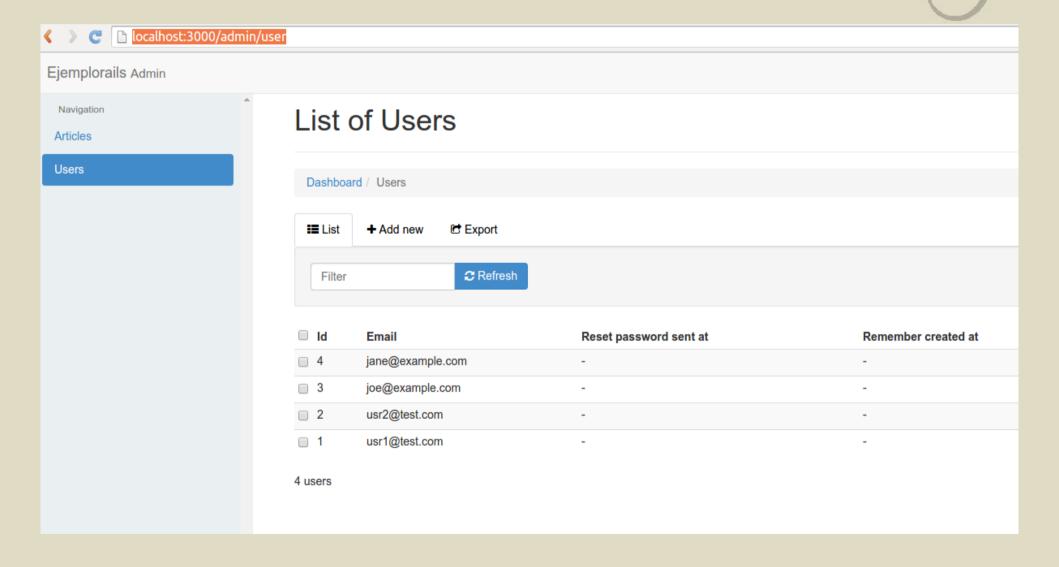
 Vayan al sitio de laboratorio 4 y copien el contenido de "Primer seed" en `db/seeds.rb`

```
db/seeds.rb
joeUser = User.new(
  :email
                          => "joe@example.com",
                          => "12345678",
  :password
  :password confirmation => "12345678"
joeUser.save!
janeUser = User.new(
                          => "jane@example.com",
  :email
                          => "12345678",
  :password
  :password confirmation => "12345678"
janeUser.save!
```

• En una consola

\$ bin/rake db:seed

Vayan a http://localhost:3000/admin/user



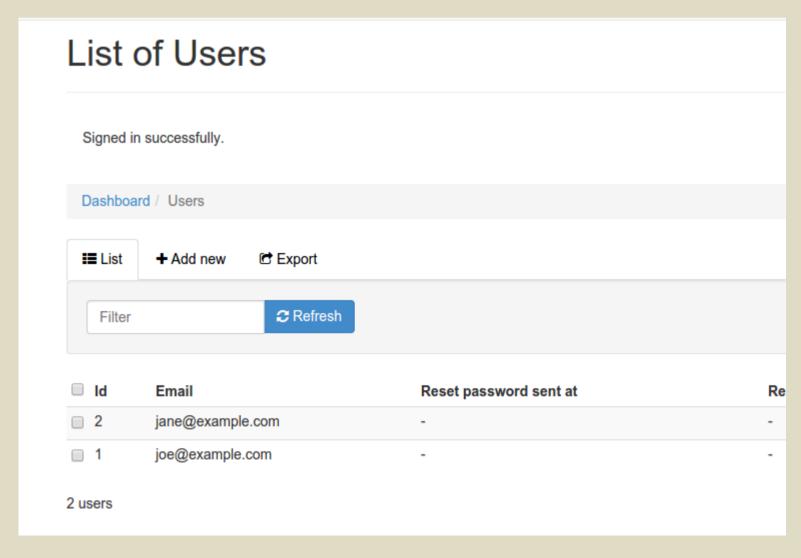
En una consola intenten nuevamente

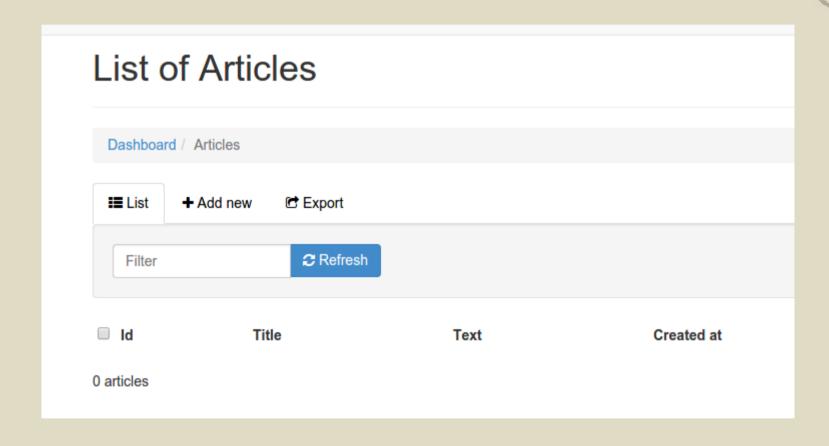
```
$ bin/rake db:seed
rake aborted!
ActiveRecord::RecordInvalid: Validation failed: Email has
already been taken
/home/andres/.rvm/gems/ruby-2.0.0-
p598@ejemplorails/gems/activerecord-
4.1.8/lib/active_record/validations.rb:57:in `save!'
...
```

#### Ejecutemos ahora

```
$ bin/rake db:reset
-- create table("articles", {:force=>true})
   -> 0.1707s
-- add index("articles", ["author id"],
{:name=>"index articles on author id"})
   -> 0.1502s
-- create table("users", {:force=>true})
   -> 0.1742s
-- add index("users", ["email"],
{:name=>"index users on email", :unique=>true})
  -> 0.1571s
-- add index("users", ["reset password token"],
{:name=>"index users on reset password token",
:unique=>true})
   -> 0.1657s
-- initialize schema migrations table()
   -> 0.3398s
```

· Vayan nuevamente a la página de admin



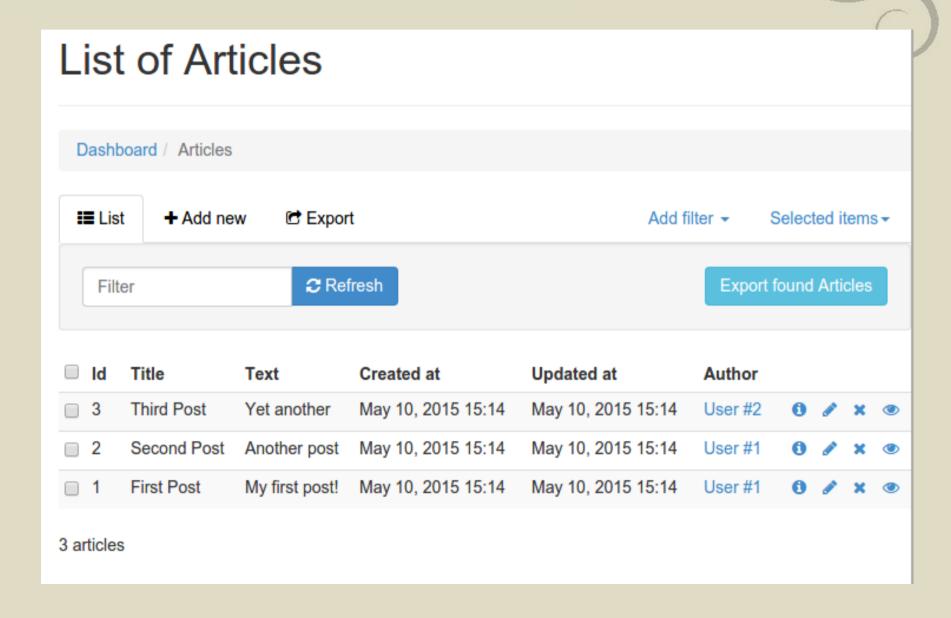


 Vayan al sitio de laboratorio 4 y copien el contenido de "Segundo seed" en `db/seeds.rb`

```
db/seeds.rb
...
janeUser.save!

Article.create!(title: 'First Post', text: 'My first
post!', author: joeUser);
Article.create!(title: 'Second Post', text: 'Another
post', author: joeUser);
Article.create!(title: 'Third Post', text: 'Yet another',
author: janeUser);
```

- Maten el server.
- `bin/rake db:reset`
- Levanten el server.
- Vayan a la página de admin utilizando el usuario joe@example.com



## Restringir los listados

- Vayan a http://localhost:3000/
- ¿Cómo hacemos para mostrar sólo los del usuario logueado?

```
/app/controllers/articles_controller.rb

class ArticlesController < ApplicationController
    ...
    def index
        @articles = Article.where(:author => current_user)
    end
    ...
end
```

Recarguen http://localhost:3000/

## Restringir los listados

- Cuidado: que no listemos un artículo no significa que no esté accesible
  - http://localhost:3000/articles/3
- Esto es parte de un tema que veremos en breve (autorización).

## "Leyendo" el log del server

- Vuelvan a cargar http://localhost:3000/
- Veamos el log de salida del servidor

```
Started GET "/" for 127.0.0.1 at 2015-05-10 12:35:23 -0300
Processing by ArticlesController#index as HTML
   User Load (0.3ms)   SELECT "users".* FROM "users" WHERE
"users"."id" = 1   ORDER BY "users"."id" ASC LIMIT 1
   Article Load (0.2ms)   SELECT "articles".* FROM "articles"
WHERE "articles"."author_id" = 1
   Rendered articles/index.html.erb within layouts/application
(1.5ms)
Completed 200 OK in 50ms (Views: 47.4ms | ActiveRecord: 0.5ms)
```

- · Vimos que un artículo pertenece a un usuario.
- Un usuario puede tener mas de un artículo.

#### articules

id	title	text	author_id
1	First Post	My first post!	1
2	Second Post	Another post	1
3	Third Post	Yet another	2

#### users

id	email
1	joe@example.com
2	jane@example.com

- Debemos indicarle a ActiveRecord que existe una relación 1 a N:
  - Del lado del artículo (ya lo hicimos)
  - Del lado del usuario

#### En una consola

```
$ bin/rails console
> user = User.find(1)
 User Load (0.2ms) SELECT "users".* FROM "users" WHERE
"users"."id" = ? LIMIT 1 [["id", 1]]
=> #<User id: 1, email: "joe@example.com",...>
> user.articles
 Article Load (0.2ms) SELECT "articles".* FROM "articles"
WHERE "articles". "author id" = ? [["author id", 1]]
=> #<ActiveRecord::Associations::CollectionProxy [#<Article
id: 1, title: "First Post", text: "My first post!",
created at: "2015-05-10 15:14:59", updated at: "2015-05-10
15:14:59", author id: 1>, #<Article id: 2, title: "Second
Post", text: "Another post", created at: "2015-05-10
15:14:59", updated at: "2015-05-10 15:14:59", author id: 1>|>
```

 Rails define un conjunto de métodos para esas relaciones

```
> user.articles.size
=> 2
> user.articles.empty?
=> false
> user.articles.where({title: "First Post"})
   Article Load (0.5ms) SELECT "articles".* FROM "articles"
WHERE "articles"."author_id" = ? AND "articles"."title" =
'First Post' [["author_id", 1]]
=> #<ActiveRecord::AssociationRelation [#<Article id: 1,
title: "First Post", text: "My first post!", created_at:
"2015-05-10 15:14:59", updated_at: "2015-05-10 15:14:59",
author_id: 1>]>
```

```
> user.articles.clear
  (50.8ms) begin transaction
 SQL (0.6ms) UPDATE "articles" SET "author id" = NULL WHERE
"articles"."author id" = ? AND "articles"."id" IN (1, 2)
[["author id", 1]]
   (136.5ms) commit transaction
=> #<ActiveRecord::Associations::CollectionProxy []>
> article = Article.first
 Article Load (0.2ms) SELECT "articles".* FROM "articles"
ORDER BY "articles". "id" ASC LIMIT 1
=> #<Article id: 1, title: "First Post", text: "My first
post!", created at: "2015-05-10 16:07:36", updated at: "2015-
05-10 16:07:36", author id: nil>
> article.save!
   (0.3ms) begin transaction
   (0.1ms) rollback transaction
ActiveRecord::RecordInvalid: Validation failed: Author can't
be blank
```

## Iniciemos nuevamente

- Maten el server.
- `bin/rake db:reset`
- Levanten el server.

## Restringir los listados

Podemos hacerlo mas elegante

```
/app/controllers/articles_controller.rb

class ArticlesController < ApplicationController
    ...
    def index
        @articles = current_user.articles
    end
    ...
end</pre>
```

Recarguen http://localhost:3000/

## Tarea para el hogar

- Lean sobre relaciones en Rails
  - http://guides.rubyonrails.org/v4.2.6/association\_b asics.html
- Lleven a su aplicación de openshift devise y rails\_admin.
- Trabajen sobre su aplicación.