

Practica de Active Record

Módulo 4

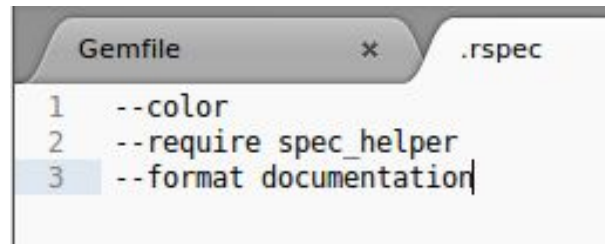
1. Crear la aplicación

- rails new todolists
- cd todolists
- Agregar al gemfile:
 - group :test do
 - gem 'rspec-rails', '~> 3.0'
 - end
- \$bundle install
- rails g rspec:install

```
practica git:(master) x rails generate rspec:install
create .rspec
create spec
create spec/spec_helper.rb
create spec/rails_helper.rb
practica git:(master) x
```

2. Configurar Tests

1. Agregar la línea **--format documentation** al .rspec para hacer mas descriptivos los resultados.
2. Reemplazar el Gemfile de la aplicación por este [Gemfile](#)
3. Agregar el archivo [assignments_spec.rb](#) a spec/
4. En la raíz del proyecto: \$rspec -e rq01



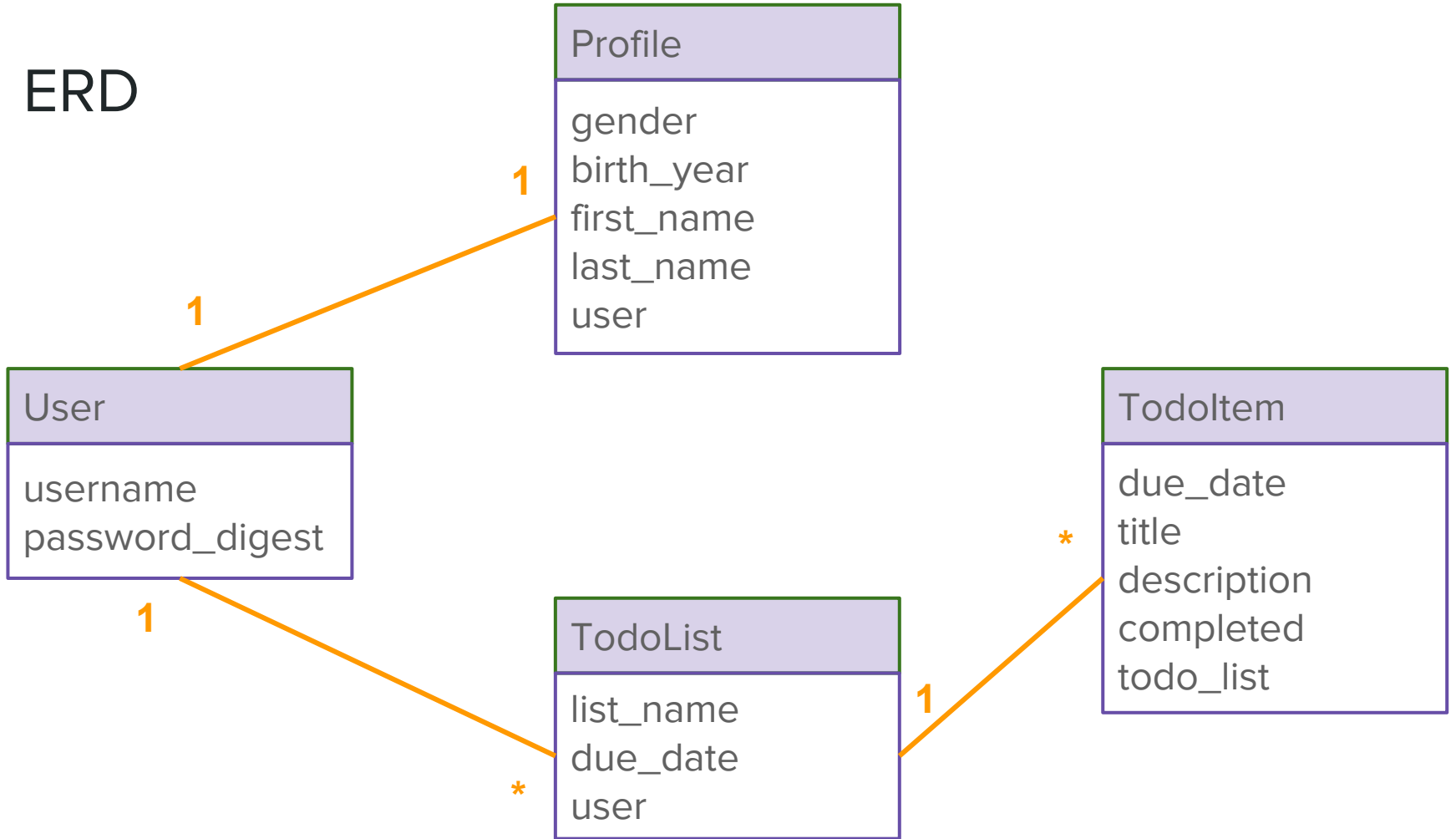
```
Gemfile x .rspec
1  --color
2  --require spec_helper
3  --format documentation
```

```
→ practica git:(master) x rspec -e rq01
Run options: include {:full_description=>/rq01/}

Assignment
  rq01
    Generate Rails application
      must have top level structure of a rails application

Finished in 0.00178 seconds (files took 1.3 seconds to load)
1 example, 0 failures
```

ERD



3. User

1. Crear el model user.
2. Correr el migration en la base de datos.

```
→ practica git:(master) x rspec -e rq02
Run options: include {:full_description=>/rq02/}

Assignment
  rq02
    User Model:
      User class created
      User database structure in place
      User class properties added
        should respond to #username
        should respond to #password_digest
        should respond to #created_at
        should respond to #updated_at

Finished in 0.15928 seconds (files took 1.31 seconds to load)
6 examples, 0 failures
```

User

username

password_digest

4. Profile

1. Crear el model profile con los siguientes campos:
 - a. **gender** - Un string que contendrá las palabras "male" o "female"
 - b. **birth_year** - un número que contendrá solo el año en el que nació cada individuo.
 - c. **first_name** - Un string para el nombre
 - d. **last_name** - Un string para el apellido
 - e. **user** - una relación 1:1 con User (Profile belongs_to User)
2. Modificar la clase User para que cuando se elimine un usuario se elimne también su profile
3. Ejecutar `rspec -e rq03`

Profile

gender
birth_year
first_name
last_name
user

5. TodoLists

1. Crear el model TodoList con los siguientes campos:
 - a. list_name - String con el nombre asignado a la lista
 - b. list_due_date - Date donde las tareas se suponene que deben estar compeltas.
2. Ejecutar rspec -e rq04
3. Modificar la clase User para que cuando se elimine un usuario se elimine también sus listas de tarea,
4. Ejecutar rspec -e rq05

TodoList
list_name
due_date
user

6. TodoItem

- Crear el modelo TodoItem
 - due_date - date donde la tarea debería estar terminada
 - title - string nombre corto de un item
 - description - text descripción específica de una tarea
 - completed - boolean value (default=false), indica si una tarea está o no completa.
 - todo_list - una relación many:1 con TodoList - (TodoItem belongs_to TodoList)
- Modificar la clase TodoList para completar la relación.
- Ejecutar `rspec -e rq06`

TodoItem
due_date
title
description
completed
todo_list

7. Verificación

```
→ practica git:(master) x rspec -e rq07
```

```
Run options: include {:full_description=>/rq07/}
```

```
All examples were filtered out
```

```
Finished in 0.00019 seconds (files took 1.33 seconds to load)
```

```
0 examples, 0 failures
```

8. One-to-many :through

1. Implementar una relación **1:many :through** desde User hasta TodoItem utilizando la asociación **1:many** entre User y TodoLists como base.
2. Ejecutar `rspec -e rq08`

User
username
password_digest

Todoltem
due_date
title
description
completed
todo_list

9. Poblar la base de datos

1. Crear un archivo seeds.rb que borre los datos existentes en los modelos y care la base de datos con los siguientes usuarios:
 - a. Carly Fiorina, 1954 Donald Trump, 1946 Ben Carson, 1951 Hillary Clinton, 1947
 - b. usernames: el apellido
 - c. password: arbitrario
 - d. Un profile por cada usuario: debe incluir nombre, apellido, genero y año de nacimiento.
 - e. Exactamente un todoList por cada usuario, donde el due_date sea de acá a un año.
 - i. Date.today es la fecha de hoy, 1.year es igual a un año
 - f. Cada TodoList contiene exactamente 5 TodoItems
 - g. Cada TodoItem tiene vencimiento (due_date) de acá a un año
 - h. Cada TodoItem debe tener algun titulo y descripción.

10. Definir un default_scope

1. default_scope: que siempre ordene por due_date para TodoList y TodoItem
2. Ejecutar rspec -e rq10

11. Validators

1. User: obligar la presencia de username.
2. Profile:
 - a. Crear un validador especial que permita que first_name o last_name sean nulos. Nunca ambos pueden ser nulos.
 - b. Utilizar un validador built-in para validar los valores de gender, deben ser solo male y female (inclusion)
 - c. Utilizar un validador Built-in que no permita que ningún registro marcado como gender = male pueda tener el first_name = "Sue" (conditional exclusion)
3. Ejecutar `rspec -e rq11`

12. Verificar Delete en cascada

Preparar todas las relaciones para que al borrar un User se borren: profile, todoList, y todoItem

Ejecutar `rspec -e rq12`

13. Aggregated Query

Teniendo en cuenta que tenemos acceso a los `todo_items` desde `User`.

Crear un método de instancia `get_completed_count` que devuelva la cantidad de `todo_items` terminados de un determinado usuario. (`completed = true`).

Ejecutar `rspec -e rq13`

14. Consultas sql - evitar sql injection

- Crear un método de clase en Profile llamado `get_all_profiles` que:
 - acepte como argumento un mínimo y máximo del año de nacimiento
 - realice la búsqueda de aquellos registros que estén entre ese mínimo y máximo (`where, between`).
 - evite un ataque de SQL Injection
 - Retorne una colección de Profiles ordenados de forma ascendente en año de nacimiento.
- Ejecutar `rspec -e rq14`

#yaEstaTodo

Son unos genios!