Curso de Ruby on Rails - aulas 31 a 40

Jackson Pires

https://www.youtube.com/watch?v=ZHPondVB9RQ&list=PLe3LRfCs4go-mkvHRMSXEOG-HDbzesyaP

Resumo do curso feito por Roberto Pinheiro

AULA 31 - NITROUS.IO

É uma plataforma onde se pode desenvolver uma aplicação Ruby on Rails toda na Web.

Nitrous.IO foi descontinuado.

AULA 32 - SELF E MONKEY PATCH EM RUBY

SELF

Self é uma palavra reservada que dá acesso ao **objeto corrente**.

Assim podemos dizer que ao usar **obj.meth**, estamos enviando o **meth** ao **obj**.

Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.1'

gem 'pry', '~> 0.11.3'
gem 'rb-readline', '~> 0.5.3'
# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '~> 5.2.3'
# Use mysql as the database for Active Record
gem 'mysql2', '>= 0.4.4', '< 0.6.0'
# Use Puma as the app server
gem 'puma', '~> 3.11'
# Use SCSS for stylesheets
gem 'sass-rails', '~> 5.0'
# Use Uglifier as compressor for JavaScript assets
gem 'uglifier', '>= 1.3.0'
# See https://github.com/rails/execjs#readme for more supported runtimes
gem 'duktape'
# Use CoffeeScript for .coffee assets and views
gem 'coffee-rails', '~> 4.2'
# Turbolinks makes navigating your web application faster. Read more:
https://github.com/turbolinks/turbolinks
gem 'turbolinks', '~> 5'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.5'
# Use Redis adapter to run Action Cable in production
# gem 'redis', '~> 4.0'
# Use ActiveModel has_secure_password
# gem 'bcrypt', '~> 3.1.7'

# Use ActiveStorage variant
# gem 'mini_magick', '~> 4.8'

# Use Capistrano for deployment
# gem 'capistrano-rails', group: :development
```

```
# Reduces boot times through caching; required in config/boot.rb
gem 'bootsnap', '>= 1.1.0', require: false

group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
end

group :development do
  # Access an interactive console on exception pages or by calling 'console' anywhere in the code.
  gem 'web-console', '>= 3.3.0'
end

group :test do
  # Adds support for Capybara system testing and selenium driver
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
  gem 'chromedriver-helper'
end

# Windows does not include zoneinfo files, so bundle the tzinfo-data gem
gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
```

```
[1] pry(main)> puts self
main
INFORMAÇÕES: não foi possível localizar arquivos para o(s) padrão(ões) especificado(s).
=> nil
[2] pry(main)> class Teste
[2] pry(main)*   def ola
[2] pry(main)*     puts "Ola #{self}"
[2] pry(main)*   end
[2] pry(main)* end
=> :ola
[3] pry(main)> x = Teste.new
=> #<Teste:0x0000000003d72780>
[4] pry(main)> x.ola
Ola #<Teste:0x0000000003d72780>
=> nil
[5] pry(main)> puts x
#<Teste:0x0000000003d72780>
=> nil
[6] pry(main)> puts self
main
=> nil
```

MONKEY PATCH

É quando se altera em tempo de execução uma classe.

```
[1] pry(main)> 1.class
INFORMAÇÕES: não foi possível localizar arquivos para o(s) padrão(ões) especificado(s).
=> Integer
[2] pry(main)> "oi".class
=> String
```

Para, por exemplo, abrir a classe String, basta redeclará-la

```
[3] pry(main)> class String
[3] pry(main)*   def ola
[3] pry(main)*     puts "Ola..."
[3] pry(main)*   end
[3] pry(main)* end
=> :ola
[4] pry(main)> x = "oi"
=> "oi"
[5] pry(main)> x.ola
Ola...
=> nil
```

O self de uma string é a própria string.

Sobrescrevendo o método existente dentro da classe String

```
[6] pry(main)> class String
[6] pry(main)*   def ola
[6] pry(main)*     puts "Ola #{self}"
[6] pry(main)*   end
[6] pry(main)* end
=> :ola
[7] pry(main)> x = "Jackson"
=> "Jackson"
[8] pry(main)> x.ola
Ola Jackson
=> nil
[9] pry(main)> "Roberto".ola
Ola Roberto
=> nil
```

Outra forma de abrir uma classe é usar **class_eval**:

```
[11] pry(main)> String.class_eval do
[11] pry(main)*   def ola
[11] pry(main)*     puts "Ola #{self} - Seja bem-vindo!"
[11] pry(main)*   end
[11] pry(main)* end
=> :ola
[12] pry(main)> x.ola
Ola Jackson - Seja bem-vindo!
=> nil
```

```
[13] pry(main)> "Roberto".ola
Ola Roberto - Seja bem-vindo!
=> nil
```

class_eval funciona apenas se a classe já existir.

AULA 33 - MÉTODOS DE INSTÂNCIA X MÉTODOS DE CLASSE

MÉTODOS DE INSTÂNCIA

São métodos que são criados dentro de uma classe e ficam disponíveis para os objetos daquela classe.

```
[1] pry(main)> class Teste
[1] pry(main)*   def oi
[1] pry(main)*     puts "oi"
[1] pry(main)*   end
[1] pry(main)* end
INFORMAÇÕES: não foi possível localizar arquivos para o(s) padrão(ões) especificado(s).
=> :oi
[2] pry(main)> x = Teste.new
=> #<Teste:0x0000000002ccbdc8>
[3] pry(main)> x.oi
oi
=> nil
```

```
[4] pry(main)> Teste.class_eval do
[4] pry(main)*   def ola
[4] pry(main)*     puts "Ola..."
[4] pry(main)*   end
[4] pry(main)* end
=> :ola
[5] pry(main)> x.oi
oi
=> nil
[6] pry(main)> x.ola
Ola...
=> nil
```

MÉTODOS DE CLASSE

São métodos estáticos acessados diretamente na classe sem precisar instanciar a classe.

```
[8] pry(main)> class Teste
[8] pry(main)*   def self.welcome
[8] pry(main)*     puts "Seja bem-vindo!"
[8] pry(main)*   end
[8] pry(main)* end
=> :welcome
[9] pry(main)> Teste.welcome
Seja bem-vindo!
=> nil
```

instance_eval

É declarar métodos específicos para uma determinada instância (objeto).

```
[12] pry(main)> y = Teste.new
=> #<Teste:0x00000000035c1fe0>
[13] pry(main)> y.instance_eval do
[13] pry(main)*   def hello
[13] pry(main)*     puts "Ola mundo!"
[13] pry(main)*   end
[13] pry(main)* end
=> :hello
[14] pry(main)> y.hello
Ola mundo!
=> nil
[15] pry(main)> x.hello
NoMethodError: undefined method `hello' for #<Teste:0x0000000002ccbdc8>
from (pry):35:in `__pry__'
[16] pry(main)>
```

AULA 34 - SINGLETON PATTERN VS SINGLETON CLASS

SINGLETON PATTERN

Singleton é um padrão de projeto de software (do inglês Design Pattern). Este padrão garante a existência de apenas uma instância de uma classe, mantendo um ponto global de acesso ao seu objeto.

```
[16] pry(main)> require 'singleton'
=> true
[17] pry(main)>
[18] pry(main)> class Foobar
[18] pry(main)*   include Singleton
[18] pry(main)*   def teste
[18] pry(main)*     puts "Testando..."
[18] pry(main)*   end
[18] pry(main)* end
=> :teste
[19] pry(main)> x = Foobar.new
NoMethodError: private method `new' called for Foobar:Class
from (pry):43:in `__pry__'
[20] pry(main)> Foobar.instance.teste
Testando...
=> nil
```

SINGLETON CLASS

class << object

```
[1] pry(main)> class Teste
[1] pry(main)*   def oi
[1] pry(main)*     puts "oi..."
[1] pry(main)*   end
[1] pry(main)* end
INFORMAÇÕES: não foi possível localizar arquivos para o(s) padrão(ões) especificado(s).
=> :oi
[2] pry(main)> x = Teste.new
=> #<Teste:0x0000000003e2f4c0>
[3] pry(main)> y = Teste.new
=> #<Teste:0x0000000003cf5848>
[4] pry(main)> x.oi
oi...
=> nil
[5] pry(main)> y.oi
oi...
=> nil
```

```
[6] pry(main)> class << y
[6] pry(main)*   def ola
[6] pry(main)*     puts "ola..."
[6] pry(main)*   end
[6] pry(main)* end
=> :ola
[7] pry(main)> x.ola
NoMethodError: undefined method `ola' for #<Teste:0x0000000003e2f4c0>
from (pry):15:in `__pry__'
[8] pry(main)> y.ola
ola...
=> nil
```

AULA 35 - PARÊNTESES, COLCHETES E CHAVES NO RUBY

PARÊNTESES

No Ruby é opcional.

```
[14] pry(main)> class Teste
[14] pry(main)*   def ola(nome)
[14] pry(main)*     puts "Ola #{nome}"
[14] pry(main)*   end
[14] pry(main)* end
=> :ola
[15] pry(main)> x.ola
ArgumentError: wrong number of arguments (given 0, expected 1)
from (pry):27:in `ola'
[16] pry(main)> x.ola("Jackson")
Ola Jackson
=> nil
[17] pry(main)> x.ola "Jackson"
Ola Jackson
=> nil
```

```
[19] pry(main)> class Teste
[19] pry(main)*   def oi nome
[19] pry(main)*     puts "Oi #{nome}"
[19] pry(main)*   end
[19] pry(main)* end
=> :oi
[20] pry(main)> x.oi
ArgumentError: wrong number of arguments (given 0, expected 1)
from (pry):36:in `oi'
[21] pry(main)> x.oi("Roberto")
Oi Roberto
=> nil
[22] pry(main)> x.oi "Roberto"
Oi Roberto
=> nil
```

GUIA DE BOAS PRÁTICAS DO RUBY

https://github.com/rubensmabueno/ruby-style-guide/blob/master/README-PT-BR.md

COLCHETES

Basicamente são utilizados com arrays ou posicionamento de arrays.

```
[24] pry(main)> frutas = ["laranja", "abacaxi", "pera", "uva"]
=> ["laranja", "abacaxi", "pera", "uva"]
[25] pry(main)> frutas.class
=> Array
[26] pry(main)> frutas[2]
=> "pera"
```

```
[27] pry(main)> nome = "Roberto"
=> "Roberto"
[28] pry(main)> nome[4]
=> "r"
```

CHAVES

São utilizadas com hash e blocos de código.

```
[38] pry(main)> aluno = {:nome => "Roberto", :idade => 58, :altura => 1.78}
=> {:nome=>"Roberto", :idade=>58, :altura=>1.78}
[39] pry(main)> aluno.class
=> Hash
[40] pry(main)> aluno[:altura]
=> 1.78
```

```
[4] pry(main)> Teste.class_eval {
[4] pry(main)*   def hello
[4] pry(main)*     puts "Ola mundo!"
[4] pry(main)*   end
[4] pry(main)* }
=> :hello
```

AULA 36 - CRIANDO UM FORMULÁRIO DE PESQUISA

rails g scaffold Customer name:string email:string

```
C:\Sites\ror-jp-p3>rails g scaffold Customer name:string email:string
      invoke  active_record
      create    db/migrate/20191025112054_create_customers.rb
      create    app/models/customer.rb
      invoke    test_unit
      create      test/models/customer_test.rb
      create      test/fixtures/customers.yml
      invoke  resource_route
       route    resources :customers
      invoke  scaffold_controller
      create    app/controllers/customers_controller.rb
      invoke    erb
      create      app/views/customers
      create      app/views/customers/index.html.erb
      create      app/views/customers/edit.html.erb
      create      app/views/customers/show.html.erb
      create      app/views/customers/new.html.erb
      create      app/views/customers/_form.html.erb
      invoke    test_unit
      create      test/controllers/customers_controller_test.rb
      create      test/system/customers_test.rb
      invoke    helper
      create      app/helpers/customers_helper.rb
      invoke      test_unit
      invoke    jbuilder
      create      app/views/customers/index.json.jbuilder
      create      app/views/customers/show.json.jbuilder
      create      app/views/customers/_customer.json.jbuilder
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/customers.coffee
      invoke    scss
      create      app/assets/stylesheets/customers.scss
      invoke  scss
      create    app/assets/stylesheets/scaffolds.scss
```

rake db:create

```
C:\Sites\ror-jp-p3>rake db:create
Created database 'ror-jp-p3_development'
Created database 'ror-jp-p3_test'
```

rake db:migrate

```
C:\Sites\ror-jp-p3>rake db:migrate
== 20191025112054 CreateCustomers: migrating ==================================
-- create_table(:customers)
   -> 0.5789s
== 20191025112054 CreateCustomers: migrated (0.5809s) =========================
```

Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.1'

# Faker
gem 'faker'
gem 'pry', '~> 0.11.3'
gem 'rb-readline', '~> 0.5.3'
# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '~> 5.2.3'
# Use mysql as the database for Active Record
gem 'mysql2', '>= 0.4.4', '< 0.6.0'
# Use Puma as the app server
gem 'puma', '~> 3.11'
# Use SCSS for stylesheets
gem 'sass-rails', '~> 5.0'
# Use Uglifier as compressor for JavaScript assets
gem 'uglifier', '>= 1.3.0'
# See https://github.com/rails/execjs#readme for more supported runtimes
gem 'duktape'
# Use CoffeeScript for .coffee assets and views
gem 'coffee-rails', '~> 4.2'
# Turbolinks makes navigating your web application faster. Read more: https://github.com/turbolinks/turbolinks
gem 'turbolinks', '~> 5'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.5'
# Use Redis adapter to run Action Cable in production
# gem 'redis', '~> 4.0'
# Use ActiveModel has_secure_password
# gem 'bcrypt', '~> 3.1.7'

# Use ActiveStorage variant
# gem 'mini_magick', '~> 4.8'

# Use Capistrano for deployment
# gem 'capistrano-rails', group: :development

# Reduces boot times through caching; required in config/boot.rb
gem 'bootsnap', '>= 1.1.0', require: false

group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
end

group :development do
```

```
  # Access an interactive console on exception pages or by calling 'console' anywhere in the code.
  gem 'web-console', '>= 3.3.0'
end

group :test do
  # Adds support for Capybara system testing and selenium driver
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
  gem 'chromedriver-helper'
end

# Windows does not include zoneinfo files, so bundle the tzinfo-data gem
gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
```

db/seeds.rb

```
50.times do |i|
  customer_name = Faker::Name.name
  customer_email = Faker::Internet.email
  customer = "#{customer_name} - #{customer_email}"
  puts "Cadastrando #{customer}"
  Customer.create(name: customer_name, email: customer_email)
end
```

rake db:seeds

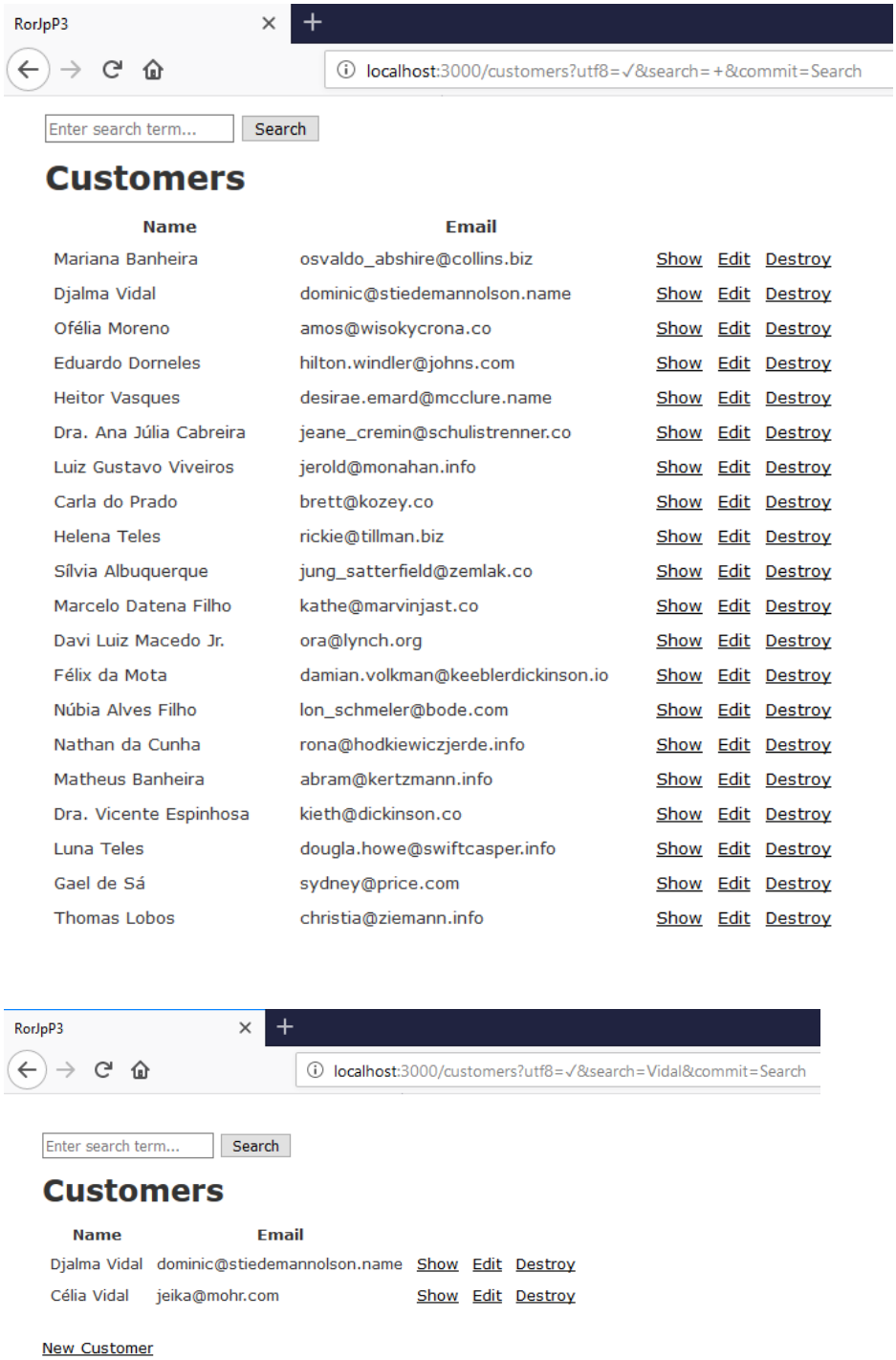
app/controllers/customers_controller.rb

```
class CustomersController < ApplicationController
 before_action :set_customer, only: [:show, :edit, :update, :destroy]

 # GET /customers
 # GET /customers.json
 def index
  if params[:search]
   @customers = Customer.where("name like ?", "%#{params[:search]}%")
  else
   @customers = Customer.all
  end
 end

 # GET /customers/1
 # GET /customers/1.json
 def show
 end

 # GET /customers/new
 def new
  @customer = Customer.new
 end

 # GET /customers/1/edit
 def edit
 end

 # POST /customers
 # POST /customers.json
 def create
  @customer = Customer.new(customer_params)

  respond_to do |format|
   if @customer.save
    format.html { redirect_to @customer, notice: 'Customer was successfully created.' }
    format.json { render :show, status: :created, location: @customer }
   else
    format.html { render :new }
    format.json { render json: @customer.errors, status: :unprocessable_entity }
   end
  end
 end

 # PATCH/PUT /customers/1
 # PATCH/PUT /customers/1.json
 def update
  respond_to do |format|
   if @customer.update(customer_params)
```

```
      format.html { redirect_to @customer, notice: 'Customer was successfully updated.' }
      format.json { render :show, status: :ok, location: @customer }
    else
      format.html { render :edit }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
 end

 # DELETE /customers/1
 # DELETE /customers/1.json
 def destroy
  @customer.destroy
  respond_to do |format|
   format.html { redirect_to customers_url, notice: 'Customer was successfully destroyed.' }
   format.json { head :no_content }
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_customer
   @customer = Customer.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def customer_params
   params.require(:customer).permit(:name, :email)
  end
end
```

app/views/customers/index.html.erb

```
<p id="notice"><%= notice %></p>

<%= form_tag(customers_path, :method => :get) do %>
 <%= text_field_tag 'search', nil, placeholder: 'Enter search term...' %>
 <%= submit_tag "Search" %>
<% end %>

<h1>Customers</h1>

<table>
 <thead>
  <tr>
   <th>Name</th>
   <th>Email</th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @customers.each do |customer| %>
   <tr>
    <td><%= customer.name %></td>
    <td><%= customer.email %></td>
    <td><%= link_to 'Show', customer %></td>
    <td><%= link_to 'Edit', edit_customer_path(customer) %></td>
    <td><%= link_to 'Destroy', customer, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Customer', new_customer_path %>
```


RorJpP3
localhost:3000/customers?utf8=✓&search=+&commit=Search
Enter search term...
Search
Customers
Name
Email
Mariana Banheira osvaldo_abshire@collins.biz Show Edit Destroy
Djalma Vidal dominic@stiedemannolson.name Show Edit Destroy
Ofélia Moreno amos@wisokycrona.co Show Edit Destroy
Eduardo Dorneles hilton.windler@johns.com Show Edit Destroy
Heitor Vasques desirae.emard@mcclure.name Show Edit Destroy
Dra. Ana Júlia Cabreira jeane_cremin@schulistrenner.co Show Edit Destroy
Luiz Gustavo Viveiros jerold@monahan.info Show Edit Destroy
Carla do Prado brett@kozey.co Show Edit Destroy
Helena Teles rickie@tillman.biz Show Edit Destroy
Sílvia Albuquerque jung_satterfield@zemlak.co Show Edit Destroy
Marcelo Datena Filho kathe@marvinjast.co Show Edit Destroy
Davi Luiz Macedo Jr. ora@lynch.org Show Edit Destroy
Félix da Mota damian.volkman@keeblerdickinson.io Show Edit Destroy
Núbia Alves Filho lon_schmeler@bode.com Show Edit Destroy
Nathan da Cunha rona@hodkiewiczjerde.info Show Edit Destroy
Matheus Banheira abram@kertzmann.info Show Edit Destroy
Dra. Vicente Espinhosa kieth@dickinson.co Show Edit Destroy
Luna Teles dougla.howe@swiftcasper.info Show Edit Destroy
Gael de Sá sydney@price.com Show Edit Destroy
Thomas Lobos christia@ziemann.info Show Edit Destroy

RorJpP3
localhost:3000/customers?utf8=✓&search=Vidal&commit=Search
Enter search term...
Search
Customers
Name
Email
Djalma Vidal dominic@stiedemannolson.name Show Edit Destroy
Célia Vidal jeika@mohr.com Show Edit Destroy
New Customer

Criando a pesquisa como um método de classe

app/models/customer.rb

```
class Customer < ApplicationRecord
  def self.search(query)
    where("name like ?", "%#{query}%")
  end
end
```

app/controllers/customers_controller.rb

```
class CustomersController < ApplicationController
  before_action :set_customer, only: [:show, :edit, :update, :destroy]

  # GET /customers
  # GET /customers.json
  def index

    if params[:search]
      @customers = Customer.search(params[:search])
    else
      @customers = Customer.all
    end
  end

  # GET /customers/1
  # GET /customers/1.json
  def show
  end

  # GET /customers/new
  def new
    @customer = Customer.new
  end

  # GET /customers/1/edit
  def edit
  end

  # POST /customers
  # POST /customers.json
  def create
    @customer = Customer.new(customer_params)
```

```
  respond_to do |format|
    if @customer.save
      format.html { redirect_to @customer, notice: 'Customer was successfully created.' }
      format.json { render :show, status: :created, location: @customer }
    else
      format.html { render :new }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
 end

 # PATCH/PUT /customers/1
 # PATCH/PUT /customers/1.json
 def update
  respond_to do |format|
    if @customer.update(customer_params)
      format.html { redirect_to @customer, notice: 'Customer was successfully updated.' }
      format.json { render :show, status: :ok, location: @customer }
    else
      format.html { render :edit }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
 end

 # DELETE /customers/1
 # DELETE /customers/1.json
 def destroy
  @customer.destroy
  respond_to do |format|
    format.html { redirect_to customers_url, notice: 'Customer was successfully destroyed.' }
    format.json { head :no_content }
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_customer
    @customer = Customer.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def customer_params
    params.require(:customer).permit(:name, :email)
  end
end
```

Usando scopes

Scopes permite encadeamento de métodos.

app/models/customer.rb

```
class Customer < ApplicationRecord
  scope :search, -> (query) {where("name like ?", "%#{query}%")}
end
```

AULA 37 - FORMULÁRIOS COMPLEXOS / NESTED ATTRIBUTES

rails g scaffold Address street:string zip:string customer:references

rake db:migrate

app/models/customer.rb

```
class Customer < ApplicationRecord
  has_one :address
  scope :search, -> (query) {where("name like ?", "%#{query}%")}
end
```

```
irb(main):009:0> customer = Customer.new(name: "Adalberto", email: "adalberto@hotmail.com")
=> #<Customer id: nil, name: "Adalberto", email: "adalberto@hotmail.com", created_at: nil, updated_at: nil>
irb(main):010:0> customer.build_address(street: "Rua dos Andradas, 58", zip: "03118-001")
=> #<Address id: nil, street: "Rua dos Andradas, 58", zip: "03118-001", customer_id: nil, created_at: nil, updated_at: nil>
irb(main):011:0> customer.save
   (0.4ms)  BEGIN
  Customer Create (0.6ms)  INSERT INTO `customers` (`name`, `email`, `created_at`, `updated_at`) VALUES ('Adalberto', 'adalberto@hotmai
   (0.4ms)  BEGIN
  Customer Create (0.6ms)  INSERT INTO `customers` (`name`, `email`, `created_at`, `updated_at`) VALUES ('Adalberto', 'adalberto@hotmai
l.com', '2019-10-25 19:09:07', '2019-10-25 19:09:07')
  Address Create (0.6ms)  INSERT INTO `addresses` (`street`, `zip`, `customer_id`, `created_at`, `updated_at`) VALUES ('Rua dos Andrada
s, 58', '03118-001', 53, '2019-10-25 19:09:07', '2019-10-25 19:09:07')
   (109.9ms)  COMMIT
=> true
```

https://api.rubyonrails.org/classes/ActiveRecord/NestedAttributes/ClassMethods.html

`accepts_nested_attributes_for`

`fields_for`

app/models/customer.rb

```
class Customer < ApplicationRecord
 has_one :address
 accepts_nested_attributes_for :address
 scope :search, -> (query) {where("name like ?", "%#{query}%")}
end
```

app/controllers/customers_controller.rb

```
class CustomersController < ApplicationController
 before_action :set_customer, only: [:show, :edit, :update, :destroy]

 # GET /customers
 # GET /customers.json
 def index

  if params[:search]
   # @customers = Customer.where("name like ?", "%#{params[:search]}%")
   @customers = Customer.search(params[:search])
  else
   @customers = Customer.all
  end
 end

 # GET /customers/1
 # GET /customers/1.json
 def show
 end

 # GET /customers/new
 def new
  @customer = Customer.new
  @customer.build_address
 end

 # GET /customers/1/edit
 def edit
 end

 # POST /customers
 # POST /customers.json
 def create
  @customer = Customer.new(customer_params)

  respond_to do |format|
```

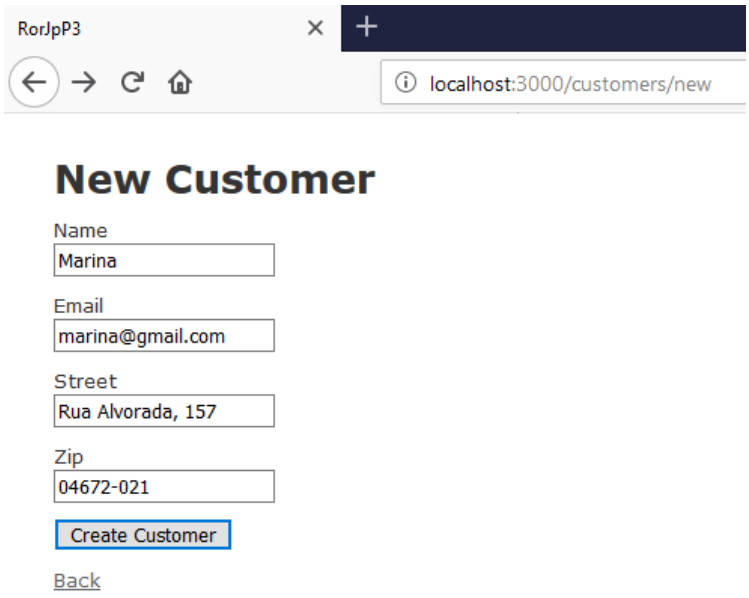
```
    if @customer.save
      format.html { redirect_to @customer, notice: 'Customer was successfully created.' }
      format.json { render :show, status: :created, location: @customer }
    else
      format.html { render :new }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /customers/1
# PATCH/PUT /customers/1.json
def update
  respond_to do |format|
    if @customer.update(customer_params)
      format.html { redirect_to @customer, notice: 'Customer was successfully updated.' }
      format.json { render :show, status: :ok, location: @customer }
    else
      format.html { render :edit }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
end

# DELETE /customers/1
# DELETE /customers/1.json
def destroy
  @customer.destroy
  respond_to do |format|
    format.html { redirect_to customers_url, notice: 'Customer was successfully destroyed.' }
    format.json { head :no_content }
  end
end

private
  # Use callbacks to share common setup or constraints between actions.
  def set_customer
    @customer = Customer.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def customer_params
    params.require(:customer).permit(:name, :email, :address_attributes => [:street, :zip])
  end
end
```

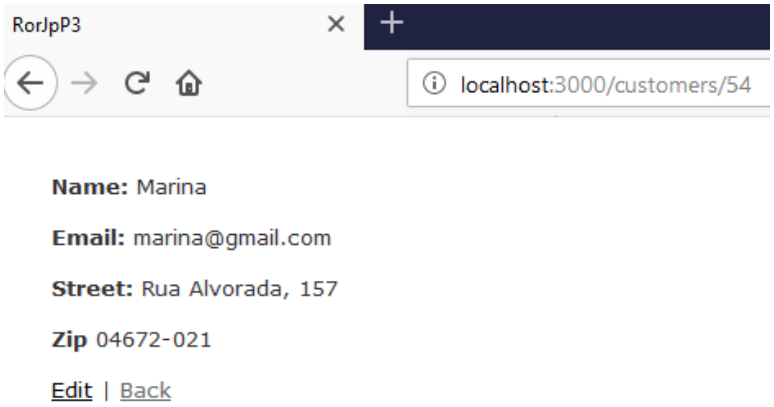
app/views/customers/_form.html.erb

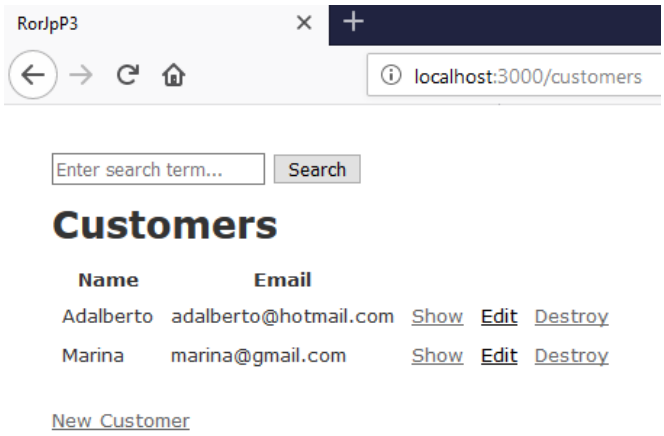
```
<%= form_with(model: customer, local: true) do |form| %>
 <% if customer.errors.any? %>
  <div id="error_explanation">
   <h2><%= pluralize(customer.errors.count, "error") %> prohibited this customer from being
saved:</h2>

   <ul>
   <% customer.errors.full_messages.each do |message| %>
    <li><%= message %></li>
   <% end %>
   </ul>
  </div>
 <% end %>

 <div class="field">
  <%= form.label :name %>
  <%= form.text_field :name %>
 </div>

 <div class="field">
  <%= form.label :email %>
  <%= form.text_field :email %>
 </div>

 <%= form.fields_for :address do |ff| %>
  <div class="field">
   <%= ff.label :street %>
   <%= ff.text_field :street %>
  </div>
  <div class="field">
   <%= ff.label :zip %>
   <%= ff.text_field :zip %>
  </div>
 <% end %>

 <div class="actions">
  <%= form.submit %>
 </div>
<% end %>
```

app/views/customers/show.html.erb

```
<p id="notice"><%= notice %></p>

<p>
 <strong>Name:</strong>
 <%= @customer.name %>
</p>

<p>
 <strong>Email:</strong>
 <%= @customer.email %>
</p>

<p>
 <strong>Street:</strong>
 <%= @customer.address.street %>
</p>

<p>
 <strong>Zip</strong>
 <%= @customer.address.zip %>
</p>

<%= link_to 'Edit', edit_customer_path(@customer) %> |
<%= link_to 'Back', customers_path %>
```



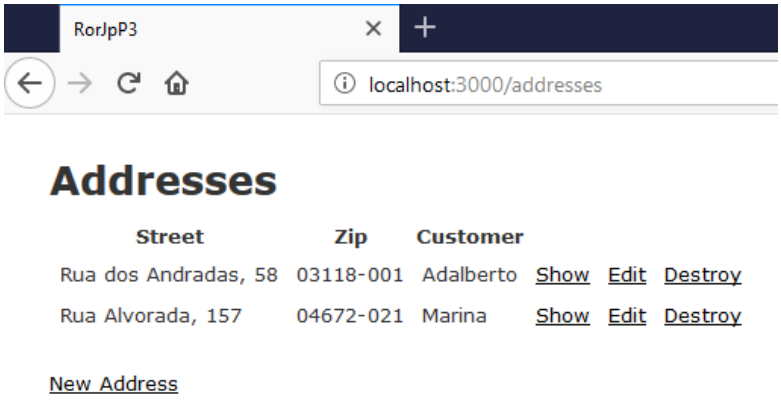




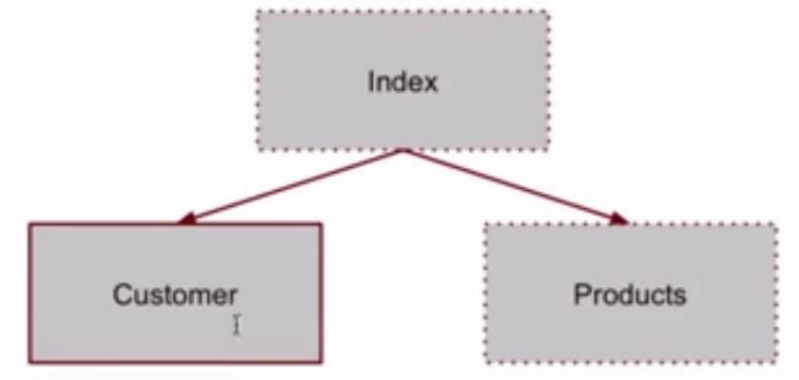
app/views/addresses/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Addresses</h1>

<table>
 <thead>
  <tr>
   <th>Street</th>
   <th>Zip</th>
   <th>Customer</th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @addresses.each do |address| %>
   <tr>
    <td><%= address.street %></td>
    <td><%= address.zip %></td>
    <td><%= address.customer.name %></td>
```

```
      <td><%= link_to 'Show', address %></td>
      <td><%= link_to 'Edit', edit_address_path(address) %></td>
      <td><%= link_to 'Destroy', address, method: :delete, data: { confirm: 'Are you sure?' } %></td>
    </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Address', new_address_path %>
```



AULA 38 - AUTENTICAÇÃO DE USUÁRIOS COM DEVISE



rails generate scaffold Product description:string quantity:integer

rails g controller Welcome index

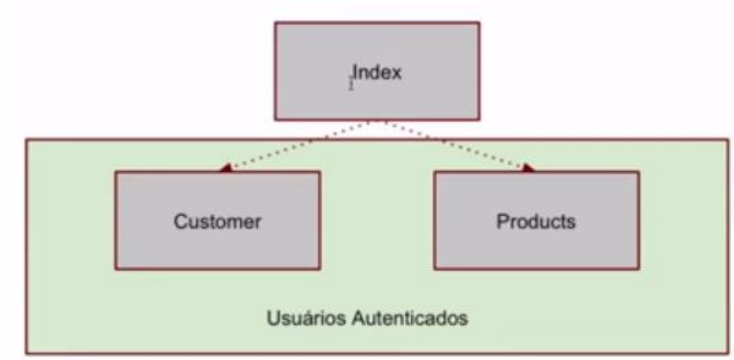
rake db:migrate

config/routes.rb

```
Rails.application.routes.draw do
  get 'welcome/index'
  root 'welcome#index'
  resources :products
  resources :addresses
  resources :customers
end
```

app/views/welcome/index.html.erb

```
<h1>Página Principal</h1>
<ul>
  <li><%= link_to "Clientes", customers_path %></li>
  <li><%= link_to "Produtos", products_path %></li>
</ul>
```



Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.1'

# Devise
gem 'devise'
# Faker
gem 'faker'
gem 'pry', '~> 0.11.3'
gem 'rb-readline', '~> 0.5.3'
# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '~> 5.2.3'
# Use mysql as the database for Active Record
gem 'mysql2', '>= 0.4.4', '< 0.6.0'
# Use Puma as the app server
gem 'puma', '~> 3.11'
# Use SCSS for stylesheets
gem 'sass-rails', '~> 5.0'
# Use Uglifier as compressor for JavaScript assets
gem 'uglifier', '>= 1.3.0'
# See https://github.com/rails/execjs#readme for more supported runtimes
gem 'duktape'
# Use CoffeeScript for .coffee assets and views
gem 'coffee-rails', '~> 4.2'
# Turbolinks makes navigating your web application faster. Read more:
https://github.com/turbolinks/turbolinks
gem 'turbolinks', '~> 5'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.5'
# Use Redis adapter to run Action Cable in production
# gem 'redis', '~> 4.0'
# Use ActiveModel has_secure_password
# gem 'bcrypt', '~> 3.1.7'

# Use ActiveStorage variant
```

```
# gem 'mini_magick', '~> 4.8'

# Use Capistrano for deployment
# gem 'capistrano-rails', group: :development

# Reduces boot times through caching; required in config/boot.rb
gem 'bootsnap', '>= 1.1.0', require: false

group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
end

group :development do
  # Access an interactive console on exception pages or by calling 'console' anywhere in the code.
  gem 'web-console', '>= 3.3.0'
end

group :test do
  # Adds support for Capybara system testing and selenium driver
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
  gem 'chromedriver-helper'
end

# Windows does not include zoneinfo files, so bundle the tzinfo-data gem
gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
```

bundle install

rails generate devise:install

```
C:\Sites\ror-jp-p3>rails generate devise:install
      create  config/initializers/devise.rb
      create  config/locales/devise.en.yml
===============================================================================

Some setup you must do manually if you haven't yet:

  1. Ensure you have defined default url options in your environments files. Here
     is an example of default_url_options appropriate for a development environment
     in config/environments/development.rb:

       config.action_mailer.default_url_options = { host: 'localhost', port: 3000 }

     In production, :host should be set to the actual host of your application.

  2. Ensure you have defined root_url to *something* in your config/routes.rb.
     For example:

       root to: "home#index"

  3. Ensure you have flash messages in app/views/layouts/application.html.erb.
     For example:

       <p class="notice"><%= notice %></p>
       <p class="alert"><%= alert %></p>

  4. You can copy Devise views (for customization) to your app by running:

       rails g devise:views

===============================================================================
```

TRADUZINDO MENSAGENS DO DEVISE

https://gist.github.com/mateusg/924555

config/locales/devise.pt-BR.yml

```
# encoding: UTF-8
# pt-BR translations for Devise
pt-BR:
  devise:
    confirmations:
      confirmed: "Sua conta foi confirmada com sucesso. Você está logado."
      send_instructions: "Dentro de minutos, você receberá um e-mail com instruções para a confirmação da sua conta."
      send_paranoid_instructions: "Se o seu endereço de e-mail estiver cadastrado, você receberá uma mensagem com instruções para confirmação da sua conta."
    failure:
      already_authenticated: "Você já está logado."
      inactive: "Sua conta ainda não foi ativada."
      invalid: "%{authentication_keys} ou senha inválida."
      locked: "Sua conta está bloqueada."
      last_attempt: "Você tem mais uma tentativa antes de bloquear sua conta."
      not_found_in_database: "%{authentication_keys} ou senha inválida."
      timeout: "Sua sessão expirou, por favor, efetue login novamente para continuar."
      unauthenticated: "Para continuar, efetue login ou registre-se."
      unconfirmed: "Antes de continuar, confirme a sua conta."
    mailer:
      confirmation_instructions:
        subject: "Instruções de confirmação"
      reset_password_instructions:
        subject: "Instruções de troca de senha"
      unlock_instructions:
        subject: "Instruções de desbloqueio"
    omniauth_callbacks:
      failure: "Não foi possível autenticá-lo como %{kind} porque  %{reason}."
      success: "Autenticado com sucesso com uma conta de %{kind}."
    passwords:
      no_token: "Você só pode acessar essa página através de um e-mail de troca de senha. Se já estiver acessando por um e-mail, verifique se a URL fornecida está completa."
      send_instructions: "Dentro de minutos, você receberá um e-mail com instruções para a troca da sua senha."
      send_paranoid_instructions: "Se o seu endereço de e-mail estiver cadastrado, você receberá um link de recuperação da senha via e-mail."
      updated: "Sua senha foi alterada com sucesso. Você está logado."
      updated_not_active: "Sua senha foi alterada com sucesso."
    registrations:
      destroyed: "Sua conta foi cancelada com sucesso. Esperamos vê-lo novamente em breve."
      signed_up: "Login efetuado com sucesso. Se não foi autorizado, a confirmação será enviada por e-mail."
      signed_up_but_inactive: "Você foi cadastrado com sucesso. No entanto, não foi possível efetuar login, pois sua conta não foi ativada."
      signed_up_but_locked: "Você foi cadastrado com sucesso. No entanto, não foi possível efetuar login, pois sua conta está bloqueada."
```

```
    signed_up_but_unconfirmed: "Uma mensagem com um link de confirmação foi enviada para o seu endereço de e-mail. Por favor, abra o link para confirmar a sua conta."
    update_needs_confirmation: "Você atualizou a sua conta com sucesso, mas o seu novo endereço de e-mail precisa ser confirmado. Por favor, acesse-o e clique no link de confirmação que enviamos."
    updated: "Sua conta foi atualizada com sucesso."
  sessions:
    signed_in: "Login efetuado com sucesso!"
    signed_out: "Saiu com sucesso."
    already_signed_out: "Saiu com sucesso."
  unlocks:
    send_instructions: "Dentro de minutos, você receberá um email com instruções para o desbloqueio da sua conta."
    send_paranoid_instructions: "Se sua conta existir, você receberá um e-mail com instruções para desbloqueá-la em alguns minutos."
    unlocked: "Sua conta foi desbloqueada com sucesso. Efetue login para continuar."

 errors:
  messages:
    already_confirmed: "já foi confirmado"
    confirmation_period_expired: "precisa ser confirmada em até %{period}, por favor, solicite uma nova"
    expired: "expirou, por favor, solicite uma nova"
    not_found: "não encontrado"
    not_locked: "não foi bloqueado"
    not_saved:
      one: "Não foi possível salvar %{resource}: 1 erro"
      other: "Não foi possível salvar %{resource}: %{count} erros."
```

ADICIONANDO O DEVISE AO MODEL

rails generate devise User

```
C:\Sites\ror-jp-p3>rails generate devise User
      invoke  active_record
      create    db/migrate/20191025235958_devise_create_users.rb
      create    app/models/user.rb
      invoke    test_unit
      create      test/models/user_test.rb
      create      test/fixtures/users.yml
      insert    app/models/user.rb
       route  devise_for :users
```

db/migrate/20191025235958_devise_create_users.rb

```
# frozen_string_literal: true

class DeviseCreateUsers < ActiveRecord::Migration[5.2]
 def change
  create_table :users do |t|
    ## Database authenticatable
    t.string :email,              null: false, default: ""
    t.string :encrypted_password, null: false, default: ""

    ## Recoverable
    t.string   :reset_password_token
    t.datetime :reset_password_sent_at

    ## Rememberable
    t.datetime :remember_created_at

    ## Trackable
    # t.integer  :sign_in_count, default: 0, null: false
    # t.datetime :current_sign_in_at
    # t.datetime :last_sign_in_at
    # t.string   :current_sign_in_ip
    # t.string   :last_sign_in_ip

    ## Confirmable
    # t.string   :confirmation_token
    # t.datetime :confirmed_at
    # t.datetime :confirmation_sent_at
    # t.string   :unconfirmed_email # Only if using reconfirmable

    ## Lockable
    # t.integer  :failed_attempts, default: 0, null: false # Only if lock strategy is :failed_attempts
```

```
    # t.string   :unlock_token # Only if unlock strategy is :email or :both
    # t.datetime :locked_at


    t.timestamps null: false
  end

  add_index :users, :email,              unique: true
  add_index :users, :reset_password_token, unique: true
  # add_index :users, :confirmation_token,   unique: true
  # add_index :users, :unlock_token,       unique: true
 end
end
```

app/models/user.rb

```
class User < ApplicationRecord
 # Include default devise modules. Others available are:
 # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
 devise :database_authenticatable, :registerable,
       :recoverable, :rememberable, :validatable
end
```

rake db:migrate

```
C:\Sites\ror-jp-p3>rake db:migrate
== 20191025235958 DeviseCreateUsers: migrating ================================
-- create_table(:users)
   -> 0.3867s
-- add_index(:users, :email, {:unique=>true})
   -> 0.6842s
-- add_index(:users, :reset_password_token, {:unique=>true})
   -> 0.1906s
== 20191025235958 DeviseCreateUsers: migrated (1.2637s) =======================
```

Ajustar os ambientes para envio de email

```
config.action_mailer.default_url_options = {
    host: 'localhost', port: 3000
```

config/enviroments/development.rb

```
Rails.application.configure do
  # Settings specified here will take precedence over those in config/application.rb.

  # Devise
  config.action_mailer.default_url_options = { host: 'localhost', port: 3000 }

  # In the development environment your application's code is reloaded on
  # every request. This slows down response time but is perfect for development
  # since you don't have to restart the web server when you make code changes.
  config.cache_classes = false

  # Do not eager load code on boot.
  config.eager_load = false

  # Show full error reports.
  config.consider_all_requests_local = true

  # Enable/disable caching. By default caching is disabled.
  # Run rails dev:cache to toggle caching.
  if Rails.root.join('tmp', 'caching-dev.txt').exist?
    config.action_controller.perform_caching = true

    config.cache_store = :memory_store
    config.public_file_server.headers = {
      'Cache-Control' => "public, max-age=#{2.days.to_i}"
    }
  else
    config.action_controller.perform_caching = false

    config.cache_store = :null_store
  end

  # Store uploaded files on the local file system (see config/storage.yml for options)
  config.active_storage.service = :local

  # Don't care if the mailer can't send.
  config.action_mailer.raise_delivery_errors = false

  config.action_mailer.perform_caching = false

  # Print deprecation notices to the Rails logger.
  config.active_support.deprecation = :log
```

```
  # Raise an error on page load if there are pending migrations.
  config.active_record.migration_error = :page_load

  # Highlight code that triggered database queries in logs.
  config.active_record.verbose_query_logs = true

  # Debug mode disables concatenation and preprocessing of assets.
  # This option may cause significant delays in view rendering with a large
  # number of complex assets.
  config.assets.debug = true

  # Suppress logger output for asset requests.
  config.assets.quiet = true

  # Raises error for missing translations
  # config.action_view.raise_on_missing_translations = true

  # Use an evented file watcher to asynchronously detect changes in source code,
  # routes, locales, etc. This feature depends on the listen gem.
  # config.file_watcher = ActiveSupport::EventedFileUpdateChecker
end
```

Adicionar filtros nos controllers que deseja autenticação

```
before_action :authenticate_user!
```

app/controllers/customers_controller.rb

```
class CustomersController < ApplicationController
  before_action :set_customer, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!

  # GET /customers
  # GET /customers.json
  def index

    if params[:search]
      # @customers = Customer.where("name like ?", "%#{params[:search]}%")
      @customers = Customer.search(params[:search])
    else
      @customers = Customer.all
    end
  end

  # GET /customers/1
```

```
# GET /customers/1.json
def show
end

# GET /customers/new
def new
  @customer = Customer.new
  @customer.build_address
end

# GET /customers/1/edit
def edit
end

# POST /customers
# POST /customers.json
def create
  @customer = Customer.new(customer_params)

  respond_to do |format|
    if @customer.save
      format.html { redirect_to @customer, notice: 'Customer was successfully created.' }
      format.json { render :show, status: :created, location: @customer }
    else
      format.html { render :new }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /customers/1
# PATCH/PUT /customers/1.json
def update
  respond_to do |format|
    if @customer.update(customer_params)
      format.html { redirect_to @customer, notice: 'Customer was successfully updated.' }
      format.json { render :show, status: :ok, location: @customer }
    else
      format.html { render :edit }
      format.json { render json: @customer.errors, status: :unprocessable_entity }
    end
  end
end

# DELETE /customers/1
# DELETE /customers/1.json
def destroy
  @customer.destroy
  respond_to do |format|
    format.html { redirect_to customers_url, notice: 'Customer was successfully destroyed.' }
    format.json { head :no_content }
  end
```

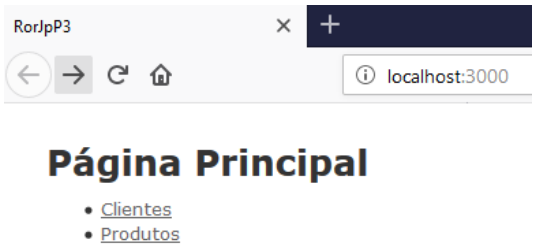
```
  end

  private
    # Use callbacks to share common setup or constraints between actions.
    def set_customer
      @customer = Customer.find(params[:id])
    end

    # Never trust parameters from the scary internet, only allow the white list through.
    def customer_params
      params.require(:customer).permit(:name, :email, :address_attributes => [:street, :zip])
    end
end
```

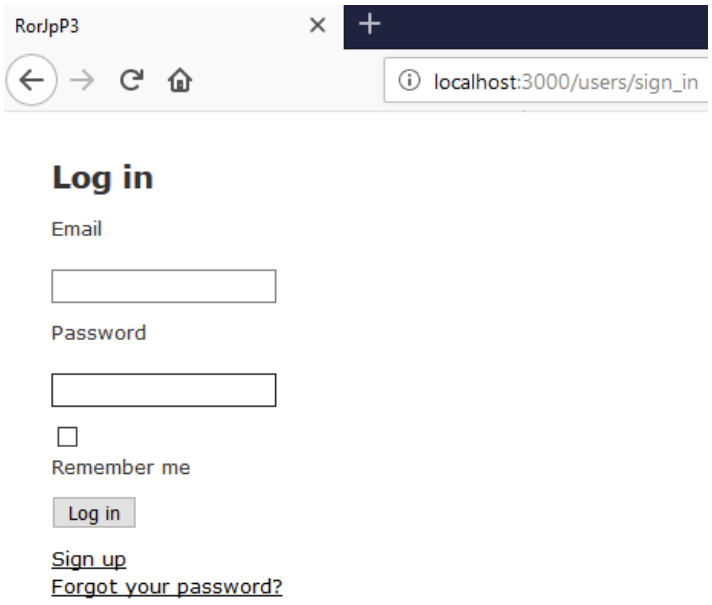
app/controllers/products_controller.rb

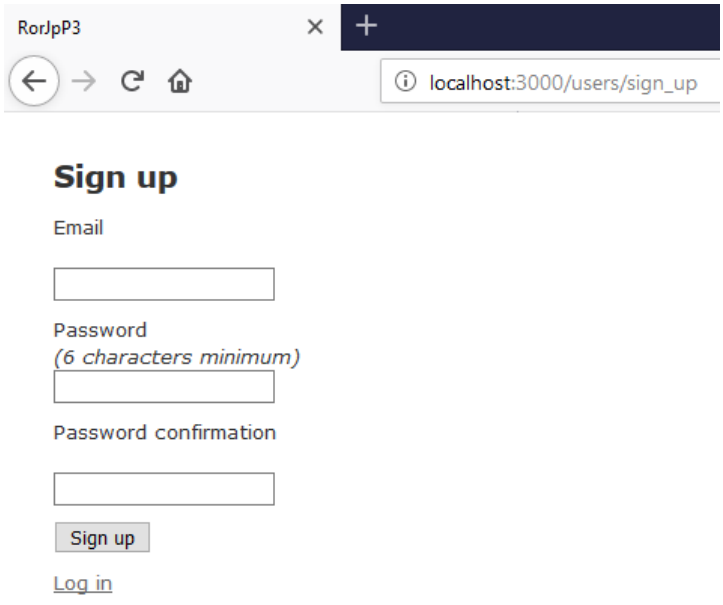
```
class ProductsController < ApplicationController
  before_action :set_product, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!

  # GET /products
  # GET /products.json
  def index
    @products = Product.all
  end

  # GET /products/1
  # GET /products/1.json
  def show
  end

  # GET /products/new
  def new
    @product = Product.new
  end

  # GET /products/1/edit
  def edit
  end

  # POST /products
  # POST /products.json
  def create
    @product = Product.new(product_params)

    respond_to do |format|
      if @product.save
        format.html { redirect_to @product, notice: 'Product was successfully created.' }
```

```
      format.json { render :show, status: :created, location: @product }
    else
      format.html { render :new }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
 end

 # PATCH/PUT /products/1
 # PATCH/PUT /products/1.json
 def update
  respond_to do |format|
    if @product.update(product_params)
      format.html { redirect_to @product, notice: 'Product was successfully updated.' }
      format.json { render :show, status: :ok, location: @product }
    else
      format.html { render :edit }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
 end

 # DELETE /products/1
 # DELETE /products/1.json
 def destroy
  @product.destroy
  respond_to do |format|
    format.html { redirect_to products_url, notice: 'Product was successfully destroyed.' }
    format.json { head :no_content }
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_product
    @product = Product.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def product_params
    params.require(:product).permit(:description, :quantity)
  end
end
```

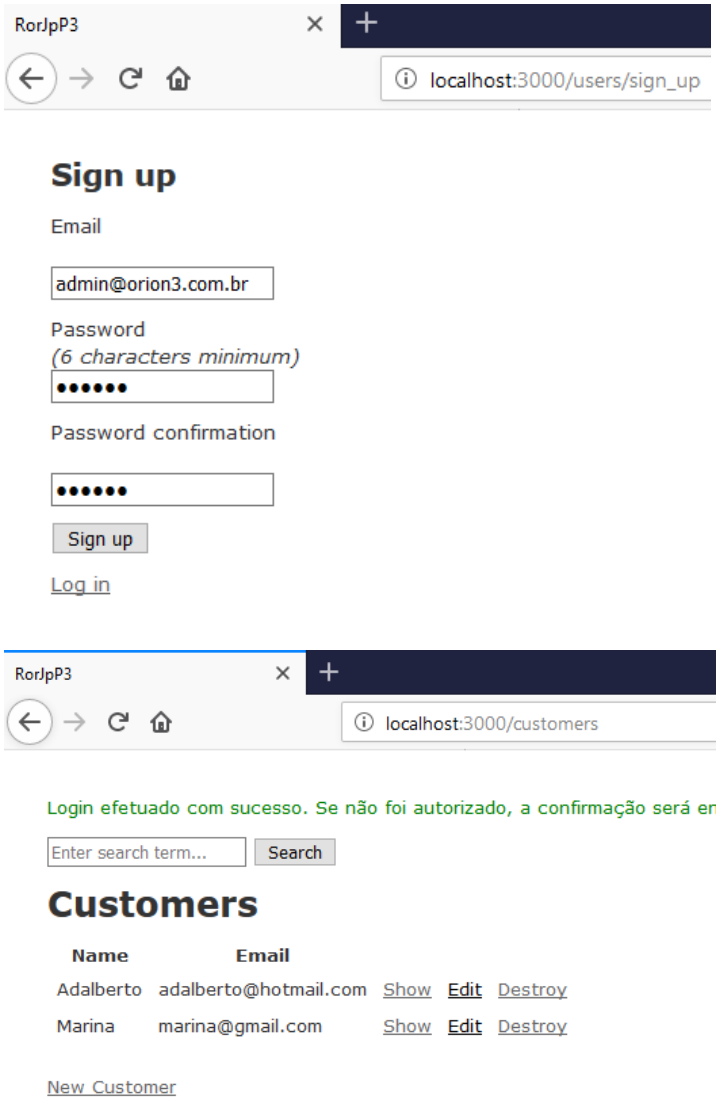
Tente acessar os cadastros envolvidos





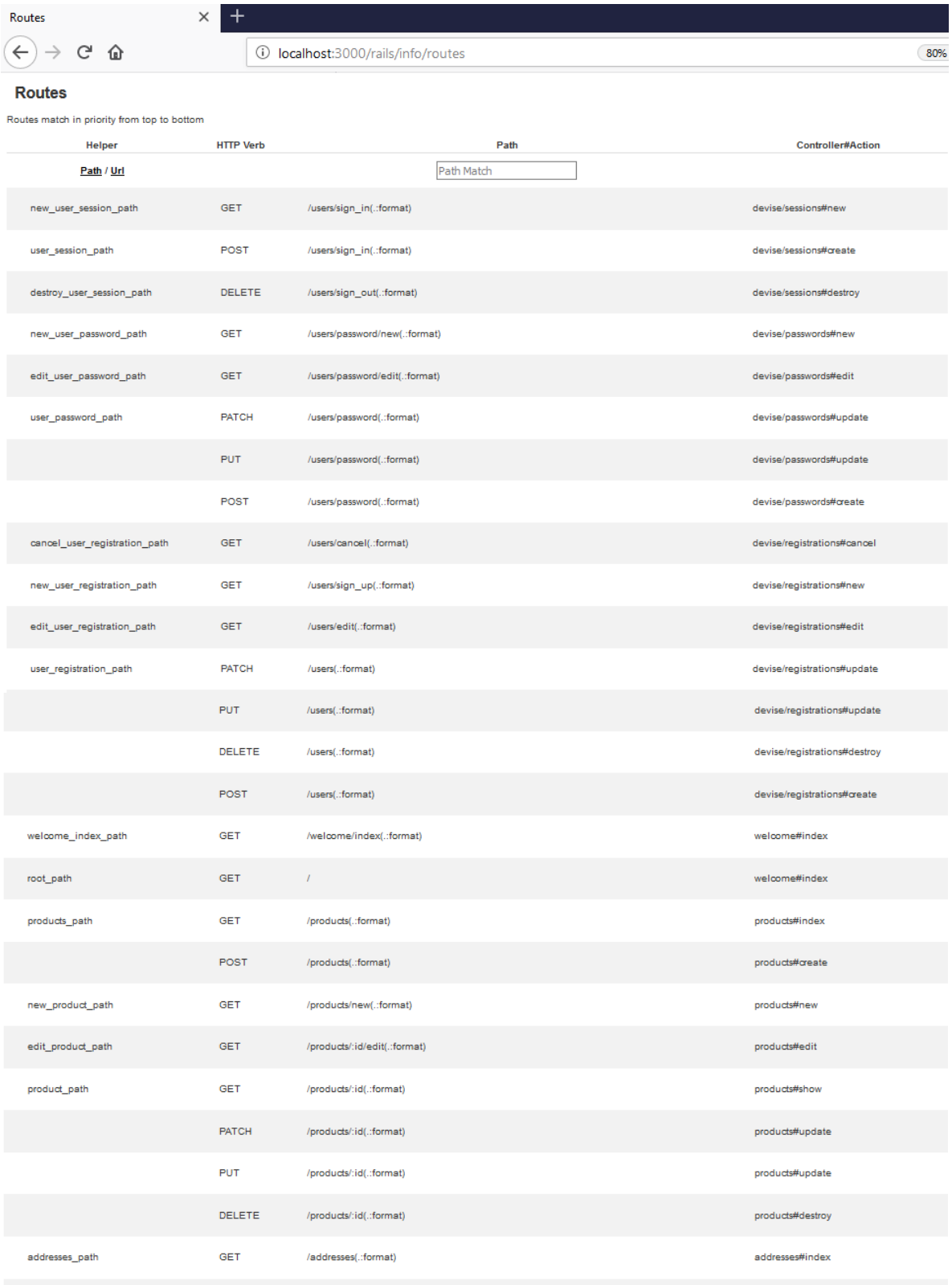


Senha: 123456



Verifique as novas rotas

rake routes ou, no navegador: localhost:3000/rails/info/routes



Routes

localhost:3000/rails/info/routes

80%

Routes

Routes match in priority from top to bottom

Helper	HTTP Verb	Path	Controller#Action
Path / Url		Path Match	
new_user_session_path	GET	/users/sign_in(.:format)	devise/sessions#new
user_session_path	POST	/users/sign_in(.:format)	devise/sessions#create
destroy_user_session_path	DELETE	/users/sign_out(.:format)	devise/sessions#destroy
new_user_password_path	GET	/users/password/new(.:format)	devise/passwords#new
edit_user_password_path	GET	/users/password/edit(.:format)	devise/passwords#edit
user_password_path	PATCH	/users/password(.:format)	devise/passwords#update
	PUT	/users/password(.:format)	devise/passwords#update
	POST	/users/password(.:format)	devise/passwords#create
cancel_user_registration_path	GET	/users/cancel(.:format)	devise/registrations#cancel
new_user_registration_path	GET	/users/sign_up(.:format)	devise/registrations#new
edit_user_registration_path	GET	/users/edit(.:format)	devise/registrations#edit
user_registration_path	PATCH	/users(.:format)	devise/registrations#update
	PUT	/users(.:format)	devise/registrations#update
	DELETE	/users(.:format)	devise/registrations#destroy
	POST	/users(.:format)	devise/registrations#create
welcome_index_path	GET	/welcome/index(.:format)	welcome#index
root_path	GET	/	welcome#index
products_path	GET	/products(.:format)	products#index
	POST	/products(.:format)	products#create
new_product_path	GET	/products/new(.:format)	products#new
edit_product_path	GET	/products/:id/edit(.:format)	products#edit
product_path	GET	/products/:id(.:format)	products#show
	PATCH	/products/:id(.:format)	products#update
	PUT	/products/:id(.:format)	products#update
	DELETE	/products/:id(.:format)	products#destroy
addresses_path	GET	/addresses(.:format)	addresses#index

	POST	/addresses(.:format)	addresses#create
new_address_path	GET	/addresses/new(.:format)	addresses#new
edit_address_path	GET	/addresses/:id/edit(.:format)	addresses#edit
address_path	GET	/addresses/:id(.:format)	addresses#show
	PATCH	/addresses/:id(.:format)	addresses#update
	PUT	/addresses/:id(.:format)	addresses#update
	DELETE	/addresses/:id(.:format)	addresses#destroy
customers_path	GET	/customers(.:format)	customers#index
	POST	/customers(.:format)	customers#create
new_customer_path	GET	/customers/new(.:format)	customers#new
edit_customer_path	GET	/customers/:id/edit(.:format)	customers#edit
customer_path	GET	/customers/:id(.:format)	customers#show
	PATCH	/customers/:id(.:format)	customers#update
	PUT	/customers/:id(.:format)	customers#update
	DELETE	/customers/:id(.:format)	customers#destroy
rails_service_blob_path	GET	/rails/active_storage/blobs/:signed_id/*filename(.:format)	active_storage/blobs#show
rails_blob_representation_path	GET	/rails/active_storage/representations/:signed_blob_id/:variation_key/*filename(.:format)	active_storage/representations#show
rails_disk_service_path	GET	/rails/active_storage/disk/:encoded_key/*filename(.:format)	active_storage/disk#show
update_rails_disk_service_path	PUT	/rails/active_storage/disk/:encoded_token(.:format)	active_storage/disk#update
rails_direct_uploads_path	POST	/rails/active_storage/direct_uploads(.:format)	active_storage/direct_uploads#create

Para sair (deslogar):

destroy_user_session_path	DELETE	/users/sign_out(.:format)	devise/sessions#destroy

Crie os links para o cadastro de novos usuários

app/views/welcome/index.html.erb

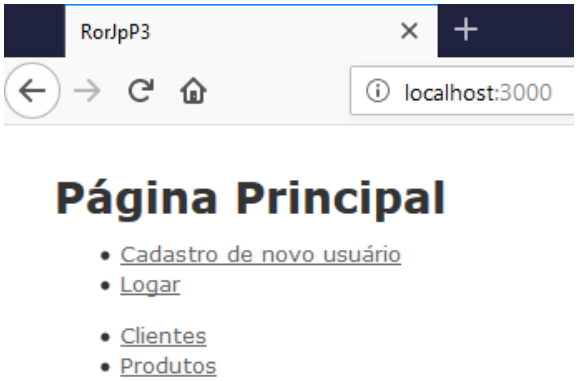
```
<h1>Página Principal</h1>

<ul>
 <li><%= link_to "Cadastro de novo usuário", new_user_registration_path %></li>
 <li><%= link_to "Logar", new_user_session_path %></li>
 <li><%= link_to "Sair", destroy_user_session_path, method: :delete %></li>
</ul>

<ul>
 <li><%= link_to "Clientes", customers_path %></li>
 <li><%= link_to "Produtos", products_path %></li>
</ul>
```

Helpers

```
user_signed_in? / current_user

        user_session
```

app/views/welcome/index.html.erb

```
<h1>Página Principal</h1>

<% if user_signed_in? %>
 <ul>
  <li><%= current_user.email %></li>
  <li><%= link_to "Sair", destroy_user_session_path, method: :delete %></li>
 </ul>
<% else %>
 <ul>
  <li><%= link_to "Cadastro de novo usuário", new_user_registration_path %></li>
  <li><%= link_to "Logar", new_user_session_path %></li>
 </ul>
<% end %>

<ul>
 <li><%= link_to "Clientes", customers_path %></li>
 <li><%= link_to "Produtos", products_path %></li>
</ul>
```



CUSTOMIZANDO AS VIEWS

Ative a opção de scoped_views

```
config/initializers/devise.rb

config.scoped_views = true
```

config/initializers/devise.rb

```
# frozen_string_literal: true

# Use this hook to configure devise mailer, warden hooks and so forth.
# Many of these configuration options can be set straight in your model.
Devise.setup do |config|
  # The secret key used by Devise. Devise uses this key to generate
  # random tokens. Changing this key will render invalid all existing
  # confirmation, reset password and unlock tokens in the database.
  # Devise will use the `secret_key_base` as its `secret_key`
  # by default. You can change it below and use your own secret key.
  # config.secret_key =
'c058d9d14b6543788a7c4513045a34fa55910701086b2dc93d137ddfc4d4349ea209f2c63daf1c2895e
b54396dd7719d639588d12a850a8f4f90cbf972d0a761'

  # ==> Controller configuration
  # Configure the parent class to the devise controllers.
  # config.parent_controller = 'DeviseController'

  # ==> Mailer Configuration
  # Configure the e-mail address which will be shown in Devise::Mailer,
  # note that it will be overwritten if you use your own mailer class
  # with default "from" parameter.
  config.mailer_sender = 'please-change-me-at-config-initializers-devise@example.com'

  # Configure the class responsible to send e-mails.
  # config.mailer = 'Devise::Mailer'

  # Configure the parent class responsible to send e-mails.
  # config.parent_mailer = 'ActionMailer::Base'

  # ==> ORM configuration
  # Load and configure the ORM. Supports :active_record (default) and
  # :mongoid (bson_ext recommended) by default. Other ORMs may be
  # available as additional gems.
  require 'devise/orm/active_record'

  # ==> Configuration for any authentication mechanism
```

```
# Configure which keys are used when authenticating a user. The default is
# just :email. You can configure it to use [:username, :subdomain], so for
# authenticating a user, both parameters are required. Remember that those
# parameters are used only when authenticating and not when retrieving from
# session. If you need permissions, you should implement that in a before filter.
# You can also supply a hash where the value is a boolean determining whether
# or not authentication should be aborted when the value is not present.
# config.authentication_keys = [:email]

# Configure parameters from the request object used for authentication. Each entry
# given should be a request method and it will automatically be passed to the
# find_for_authentication method and considered in your model lookup. For instance,
# if you set :request_keys to [:subdomain], :subdomain will be used on authentication.
# The same considerations mentioned for authentication_keys also apply to request_keys.
# config.request_keys = []

# Configure which authentication keys should be case-insensitive.
# These keys will be downcased upon creating or modifying a user and when used
# to authenticate or find a user. Default is :email.
config.case_insensitive_keys = [:email]

# Configure which authentication keys should have whitespace stripped.
# These keys will have whitespace before and after removed upon creating or
# modifying a user and when used to authenticate or find a user. Default is :email.
config.strip_whitespace_keys = [:email]

# Tell if authentication through request.params is enabled. True by default.
# It can be set to an array that will enable params authentication only for the
# given strategies, for example, `config.params_authenticatable = [:database]` will
# enable it only for database (email + password) authentication.
# config.params_authenticatable = true

# Tell if authentication through HTTP Auth is enabled. False by default.
# It can be set to an array that will enable http authentication only for the
# given strategies, for example, `config.http_authenticatable = [:database]` will
# enable it only for database authentication. The supported strategies are:
# :database      = Support basic authentication with authentication key + password
# config.http_authenticatable = false

# If 401 status code should be returned for AJAX requests. True by default.
# config.http_authenticatable_on_xhr = true

# The realm used in Http Basic Authentication. 'Application' by default.
# config.http_authentication_realm = 'Application'

# It will change confirmation, password recovery and other workflows
# to behave the same regardless if the e-mail provided was right or wrong.
# Does not affect registerable.
# config.paranoid = true

# By default Devise will store the user in session. You can skip storage for
# particular strategies by setting this option.
```

```
  # Notice that if you are skipping storage for all authentication paths, you
  # may want to disable generating routes to Devise's sessions controller by
  # passing skip: :sessions to `devise_for` in your config/routes.rb
  config.skip_session_storage = [:http_auth]

  # By default, Devise cleans up the CSRF token on authentication to
  # avoid CSRF token fixation attacks. This means that, when using AJAX
  # requests for sign in and sign up, you need to get a new CSRF token
  # from the server. You can disable this option at your own risk.
  # config.clean_up_csrf_token_on_authentication = true

  # When false, Devise will not attempt to reload routes on eager load.
  # This can reduce the time taken to boot the app but if your application
  # requires the Devise mappings to be loaded during boot time the application
  # won't boot properly.
  # config.reload_routes = true

  # ==> Configuration for :database_authenticatable
  # For bcrypt, this is the cost for hashing the password and defaults to 11. If
  # using other algorithms, it sets how many times you want the password to be hashed.
  #
  # Limiting the stretches to just one in testing will increase the performance of
  # your test suite dramatically. However, it is STRONGLY RECOMMENDED to not use
  # a value less than 10 in other environments. Note that, for bcrypt (the default
  # algorithm), the cost increases exponentially with the number of stretches (e.g.
  # a value of 20 is already extremely slow: approx. 60 seconds for 1 calculation).
  config.stretches = Rails.env.test? ? 1 : 11

  # Set up a pepper to generate the hashed password.
  # config.pepper =
'1013e3739d35769f618383d2fe954a506f68bdf0f629792cc53ffd5cd2974420fa19c559cf00c24d19b8b
840477d8a92e67366279d379d6c4cccd7d1d1292d0f'

  # Send a notification to the original email when the user's email is changed.
  # config.send_email_changed_notification = false

  # Send a notification email when the user's password is changed.
  # config.send_password_change_notification = false

  # ==> Configuration for :confirmable
  # A period that the user is allowed to access the website even without
  # confirming their account. For instance, if set to 2.days, the user will be
  # able to access the website for two days without confirming their account,
  # access will be blocked just in the third day.
  # You can also set it to nil, which will allow the user to access the website
  # without confirming their account.
  # Default is 0.days, meaning the user cannot access the website without
  # confirming their account.
  # config.allow_unconfirmed_access_for = 2.days

  # A period that the user is allowed to confirm their account before their
  # token becomes invalid. For example, if set to 3.days, the user can confirm
```

```
# their account within 3 days after the mail was sent, but on the fourth day
# their account can't be confirmed with the token any more.
# Default is nil, meaning there is no restriction on how long a user can take
# before confirming their account.
# config.confirm_within = 3.days

# If true, requires any email changes to be confirmed (exactly the same way as
# initial account confirmation) to be applied. Requires additional unconfirmed_email
# db field (see migrations). Until confirmed, new email is stored in
# unconfirmed_email column, and copied to email column on successful confirmation.
config.reconfirmable = true

# Defines which key will be used when confirming an account
# config.confirmation_keys = [:email]

# ==> Configuration for :rememberable
# The time the user will be remembered without asking for credentials again.
# config.remember_for = 2.weeks

# Invalidates all the remember me tokens when the user signs out.
config.expire_all_remember_me_on_sign_out = true

# If true, extends the user's remember period when remembered via cookie.
# config.extend_remember_period = false

# Options to be passed to the created cookie. For instance, you can set
# secure: true in order to force SSL only cookies.
# config.rememberable_options = {}

# ==> Configuration for :validatable
# Range for password length.
config.password_length = 6..128

# Email regex used to validate email formats. It simply asserts that
# one (and only one) @ exists in the given string. This is mainly
# to give user feedback and not to assert the e-mail validity.
config.email_regexp = /\A[^@\s]+@[^@\s]+\z/

# ==> Configuration for :timeoutable
# The time you want to timeout the user session without activity. After this
# time the user will be asked for credentials again. Default is 30 minutes.
# config.timeout_in = 30.minutes

# ==> Configuration for :lockable
# Defines which strategy will be used to lock an account.
# :failed_attempts = Locks an account after a number of failed attempts to sign in.
# :none            = No lock strategy. You should handle locking by yourself.
# config.lock_strategy = :failed_attempts

# Defines which key will be used when locking and unlocking an account
# config.unlock_keys = [:email]
```

```
# Defines which strategy will be used to unlock an account.
# :email = Sends an unlock link to the user email
# :time  = Re-enables login after a certain amount of time (see :unlock_in below)
# :both  = Enables both strategies
# :none  = No unlock strategy. You should handle unlocking by yourself.
# config.unlock_strategy = :both

# Number of authentication tries before locking an account if lock_strategy
# is failed attempts.
# config.maximum_attempts = 20

# Time interval to unlock the account if :time is enabled as unlock_strategy.
# config.unlock_in = 1.hour

# Warn on the last attempt before the account is locked.
# config.last_attempt_warning = true

# ==> Configuration for :recoverable
#
# Defines which key will be used when recovering the password for an account
# config.reset_password_keys = [:email]

# Time interval you can reset your password with a reset password key.
# Don't put a too small interval or your users won't have the time to
# change their passwords.
config.reset_password_within = 6.hours

# When set to false, does not sign a user in automatically after their password is
# reset. Defaults to true, so a user is signed in automatically after a reset.
# config.sign_in_after_reset_password = true

# ==> Configuration for :encryptable
# Allow you to use another hashing or encryption algorithm besides bcrypt (default).
# You can use :sha1, :sha512 or algorithms from others authentication tools as
# :clearance_sha1, :authlogic_sha512 (then you should set stretches above to 20
# for default behavior) and :restful_authentication_sha1 (then you should set
# stretches to 10, and copy REST_AUTH_SITE_KEY to pepper).
#
# Require the `devise-encryptable` gem when using anything other than bcrypt
# config.encryptor = :sha512

# ==> Scopes configuration
# Turn scoped views on. Before rendering "sessions/new", it will first check for
# "users/sessions/new". It's turned off by default because it's slower if you
# are using only default views.
# config.scoped_views = false
config.scoped_views = true

# Configure the default scope given to Warden. By default it's the first
# devise role declared in your routes (usually :user).
# config.default_scope = :user
```

```
  # Set this configuration to false if you want /users/sign_out to sign out
  # only the current scope. By default, Devise signs out all scopes.
  # config.sign_out_all_scopes = true

  # ==> Navigation configuration
  # Lists the formats that should be treated as navigational. Formats like
  # :html, should redirect to the sign in page when the user does not have
  # access, but formats like :xml or :json, should return 401.
  #
  # If you have any extra navigational formats, like :iphone or :mobile, you
  # should add them to the navigational formats lists.
  #
  # The "*/*" below is required to match Internet Explorer requests.
  # config.navigational_formats = ['*/*', :html]

  # The default HTTP method used to sign out a resource. Default is :delete.
  config.sign_out_via = :delete

  # ==> OmniAuth
  # Add a new OmniAuth provider. Check the wiki for more information on setting
  # up on your models and hooks.
  # config.omniauth :github, 'APP_ID', 'APP_SECRET', scope: 'user,public_repo'

  # ==> Warden configuration
  # If you want to use other strategies, that are not supported by Devise, or
  # change the failure app, you can configure them inside the config.warden block.
  #
  # config.warden do |manager|
  #   manager.intercept_401 = false
  #   manager.default_strategies(scope: :user).unshift :some_external_strategy
  # end

  # ==> Mountable engine configurations
  # When using Devise inside an engine, let's call it `MyEngine`, and this engine
  # is mountable, there are some extra configurations to be taken into account.
  # The following options are available, assuming the engine is mounted as:
  #
  #     mount MyEngine, at: '/my_engine'
  #
  # The router that invoked `devise_for`, in the example above, would be:
  # config.router_name = :my_engine
  #
  # When using OmniAuth, Devise cannot automatically set OmniAuth path,
  # so you need to do it manually. For the users scope, it would be:
  # config.omniauth_path_prefix = '/my_engine/users/auth'

  # ==> Turbolinks configuration
  # If your app is using Turbolinks, Turbolinks::Controller needs to be included to make redirection
work correctly:
  #
  # ActiveSupport.on_load(:devise_failure_app) do
  #   include Turbolinks::Controller
```

```
  # end

  # ==> Configuration for :registerable

  # When set to false, does not sign a user in automatically after their password is
  # changed. Defaults to true, so a user is signed in automatically after changing a password.
  # config.sign_in_after_change_password = true
end
```

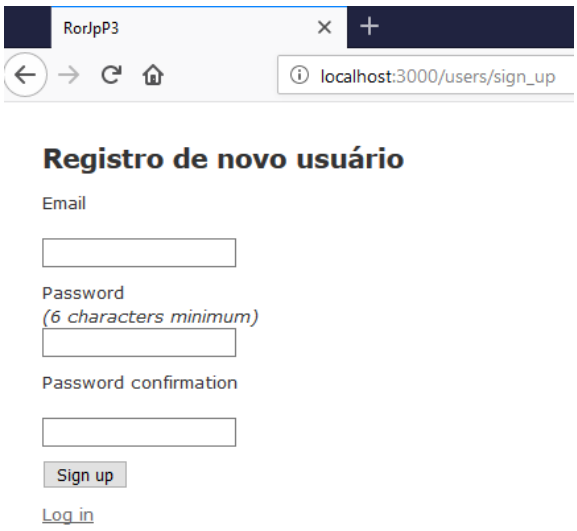
Gere as view e faça as alterações

rails generate devise:views users

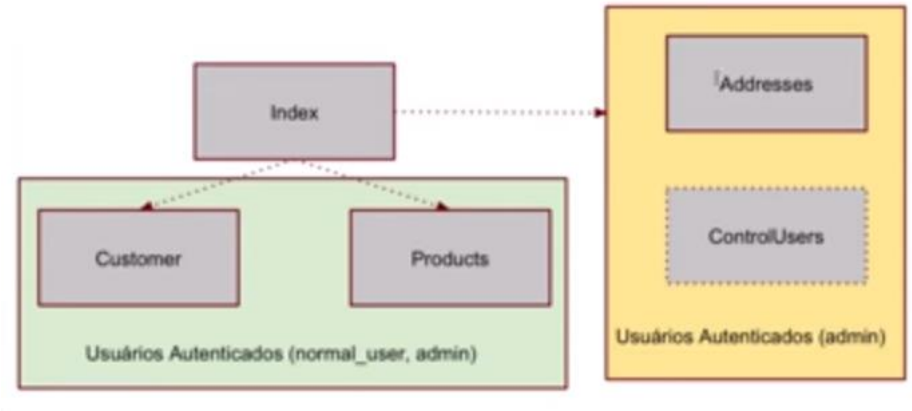
```
C:\Sites\ror-jp-p3>rails generate devise:views users
      invoke  Devise::Generators::SharedViewsGenerator
      create    app/views/users/shared
      create    app/views/users/shared/_error_messages.html.erb
      create    app/views/users/shared/_links.html.erb
      invoke  form_for
      create    app/views/users/confirmations
      create    app/views/users/confirmations/new.html.erb
      create    app/views/users/passwords
      create    app/views/users/passwords/edit.html.erb
      create    app/views/users/passwords/new.html.erb
      create    app/views/users/registrations
      create    app/views/users/registrations/edit.html.erb
      create    app/views/users/registrations/new.html.erb
      create    app/views/users/sessions
      create    app/views/users/sessions/new.html.erb
      create    app/views/users/unlocks
      create    app/views/users/unlocks/new.html.erb
      invoke  erb
      create    app/views/users/mailer
      create    app/views/users/mailer/confirmation_instructions.html.erb
      create    app/views/users/mailer/email_changed.html.erb
      create    app/views/users/mailer/password_change.html.erb
      create    app/views/users/mailer/reset_password_instructions.html.erb
      create    app/views/users/mailer/unlock_instructions.html.erb
```

views/users/registrations/new.html.erb

```
<h2>Registro de novo usuário</h2>

<%= form_for(resource, as: resource_name, url: registration_path(resource_name)) do |f| %>
 <%= render "devise/shared/error_messages", resource: resource %>

 <div class="field">
  <%= f.label :email %><br />
  <%= f.email_field :email, autofocus: true, autocomplete: "email" %>
 </div>

 <div class="field">
  <%= f.label :password %>
  <% if @minimum_password_length %>
  <em>(<%= @minimum_password_length %> characters minimum)</em>
  <% end %><br />
  <%= f.password_field :password, autocomplete: "new-password" %>
 </div>

 <div class="field">
  <%= f.label :password_confirmation %><br />
  <%= f.password_field :password_confirmation, autocomplete: "new-password" %>
 </div>

 <div class="actions">
  <%= f.submit "Sign up" %>
 </div>
<% end %>

<%= render "users/shared/links" %>
```



AULA 39 - USANDO AUTORIZAÇÃO COM PUNDIT E DEVISE



rails g controller ControlUsers index

```
C:\Sites\ror-jp-p3>rails g controller ControlUsers index
      create  app/controllers/control_users_controller.rb
       route  get 'control_users/index'
      invoke  erb
      create    app/views/control_users
      create    app/views/control_users/index.html.erb
      invoke  test_unit
      create    test/controllers/control_users_controller_test.rb
      invoke  helper
      create    app/helpers/control_users_helper.rb
      invoke    test_unit
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/control_users.coffee
      invoke    scss
      create      app/assets/stylesheets/control_users.scss
```

app/controllers/control_users_controller.rb

```
class ControlUsersController < ApplicationController
 before_action :authenticate_user!

 def index
   @users = User.all
 end
end
```

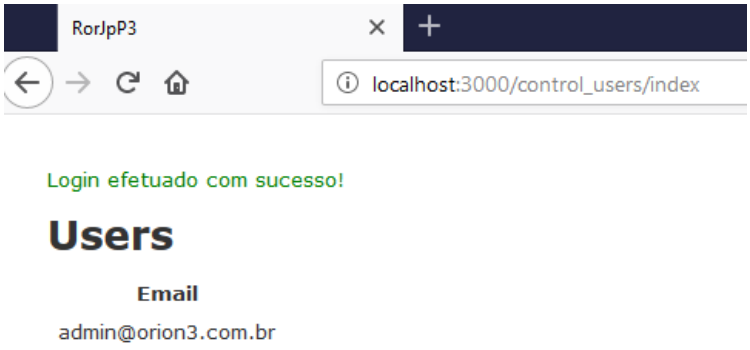
app/controllers/addresses_controller.rb

```
class AddressesController < ApplicationController
 before_action :set_address, only: [:show, :edit, :update, :destroy]
 before_action :authenticate_user!

 # GET /addresses
 # GET /addresses.json
 def index
  @addresses = Address.all
 end

 # GET /addresses/1
 # GET /addresses/1.json
 def show
 end

 # GET /addresses/new
 def new
  @address = Address.new
 end

 # GET /addresses/1/edit
 def edit
 end

 # POST /addresses
 # POST /addresses.json
 def create
  @address = Address.new(address_params)

  respond_to do |format|
   if @address.save
    format.html { redirect_to @address, notice: 'Address was successfully created.' }
    format.json { render :show, status: :created, location: @address }
   else
    format.html { render :new }
    format.json { render json: @address.errors, status: :unprocessable_entity }
   end
  end
 end

 # PATCH/PUT /addresses/1
 # PATCH/PUT /addresses/1.json
 def update
  respond_to do |format|
   if @address.update(address_params)
    format.html { redirect_to @address, notice: 'Address was successfully updated.' }
    format.json { render :show, status: :ok, location: @address }
   else
```

```
      format.html { render :edit }
      format.json { render json: @address.errors, status: :unprocessable_entity }
    end
  end
 end

 # DELETE /addresses/1
 # DELETE /addresses/1.json
 def destroy
  @address.destroy
  respond_to do |format|
   format.html { redirect_to addresses_url, notice: 'Address was successfully destroyed.' }
   format.json { head :no_content }
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_address
   @address = Address.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def address_params
   params.require(:address).permit(:street, :zip, :customer_id)
  end
end
```

views/control_users/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Users</h1>

<table>
 <thead>
 <tr>
  <th>Email</th>
 </tr>
 </thead>

 <tbody>
 <% @users.each do |user| %>
  <tr>
   <td><%= user.email %></td>
  </tr>
 <% end %>
 </tbody>
</table>

<br>
```



email	encrypted_password	role
teste@teste.com	a9asfdasdf9	0
admin@teste.com	43324l5klkjn	1

Adicionando uma migration para a role

rails g migration AddRole ToUsers

```
C:\Sites\ror-jp-p3>rails g migration AddRole ToUsers
      invoke  active_record
      create    db/migrate/20191026131159_add_role.rb
```

Altere a migration

```
add_column :users, :role, :integer, :
             default => 0
```

db/migrate/20191026131159_add_role.rb

```
class AddRole < ActiveRecord::Migration[5.2]
 def change
  add_column :users, :role, :integer, :default => 0
 end
end
```

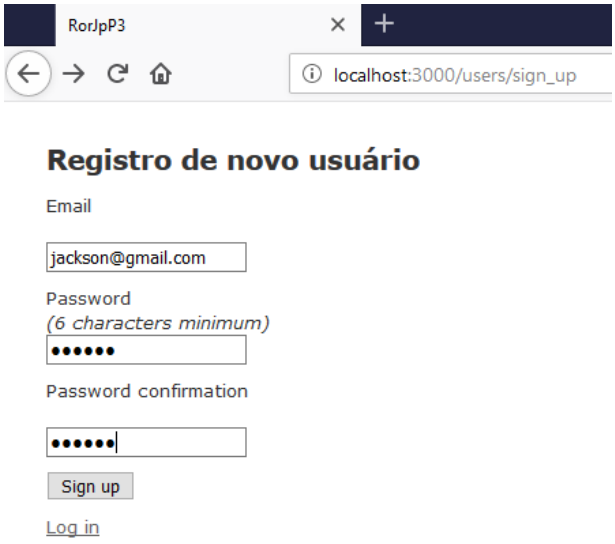
app/models/user.rb

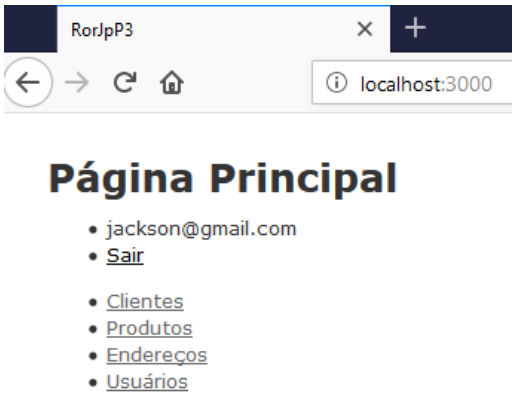
```
class User < ApplicationRecord
 enum role: [:normal_user, :admin]
 # Include default devise modules. Others available are:
 # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
 devise :database_authenticatable, :registerable,
       :recoverable, :rememberable, :validatable
end
```

Rode a migration e teste a criação de um novo user

rake db:migrate

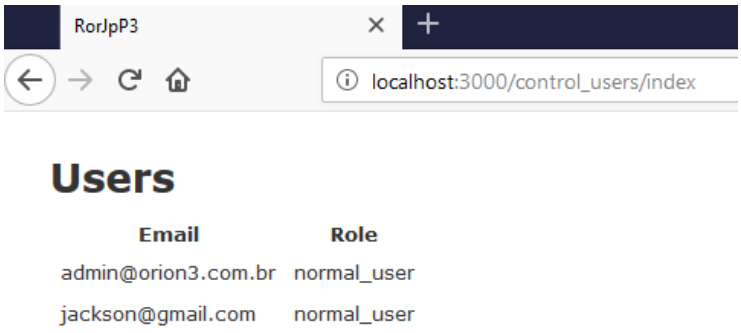
```
C:\Sites\ror-jp-p3>rake db:migrate
== 20191026131159 AddRole: migrating ==========================================
-- add_column(:users, :role, :integer, {:default=>0})
   -> 1.1592s
== 20191026131159 AddRole: migrated (1.1599s) =================================
```





app/views/control_users/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Users</h1>

<table>
 <thead>
 <tr>
  <th>Email</th>
  <th>Role</th>
 </tr>
 </thead>

 <tbody>
 <% @users.each do |user| %>
  <tr>
   <td><%= user.email %></td>
   <td><%= user.role %></td>
  </tr>
 <% end %>
 </tbody>
</table>

<br>
```

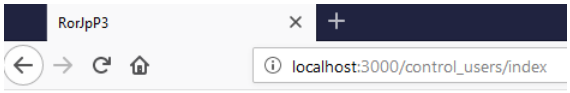


Ajuste a visualização dos usuários e faça testes no rails console

rails c

x = User.first
x.role = :admin
x.save

```
irb(main):001:0> x = User.first
   (0.5ms)  SET NAMES utf8,  @@SESSION.sql_mode = CONCAT(CONCAT(@@sql_mode, ',STRICT_ALL_TABLES'), ',NO_AUTO_VALUE_ON_ZERO'),  @@SESSIO
N.sql_auto_is_null = 0, @@SESSION.wait_timeout = 2147483
  User Load (1.6ms)  SELECT  `users`.* FROM `users` ORDER BY `users`.`id` ASC LIMIT 1
=> #<User id: 1, email: "admin@orion3.com.br", created_at: "2019-10-26 00:51:55", updated_at: "2019-10-26 00:51:55", role: "normal_user
">
irb(main):002:0> x.role = :admin
=> :admin
irb(main):003:0> x.save
   (0.4ms)  BEGIN
  User Update (0.8ms)  UPDATE `users` SET `updated_at` = '2019-10-26 14:31:42', `role` = 1 WHERE `users`.`id` = 1
   (100.6ms)  COMMIT
=> true
```



Users

Email	Role
admin@orion3.com.br	admin
jackson@gmail.com	normal_user

PUNDIT

Adicione ao Gemfile.

Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.1'

# Pundit
gem 'pundit'
# Devise
gem 'devise'
# Faker
gem 'faker'
gem 'pry', '~> 0.11.3'
gem 'rb-readline', '~> 0.5.3'
# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '~> 5.2.3'
# Use mysql as the database for Active Record
gem 'mysql2', '>= 0.4.4', '< 0.6.0'
```

```
# Use Puma as the app server
gem 'puma', '~> 3.11'
# Use SCSS for stylesheets
gem 'sass-rails', '~> 5.0'
# Use Uglifier as compressor for JavaScript assets
gem 'uglifier', '>= 1.3.0'
# See https://github.com/rails/execjs#readme for more supported runtimes
gem 'duktape'
# Use CoffeeScript for .coffee assets and views
gem 'coffee-rails', '~> 4.2'
# Turbolinks makes navigating your web application faster. Read more:
https://github.com/turbolinks/turbolinks
gem 'turbolinks', '~> 5'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.5'
# Use Redis adapter to run Action Cable in production
# gem 'redis', '~> 4.0'
# Use ActiveModel has_secure_password
# gem 'bcrypt', '~> 3.1.7'

# Use ActiveStorage variant
# gem 'mini_magick', '~> 4.8'

# Use Capistrano for deployment
# gem 'capistrano-rails', group: :development

# Reduces boot times through caching; required in config/boot.rb
gem 'bootsnap', '>= 1.1.0', require: false

group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
end

group :development do
  # Access an interactive console on exception pages or by calling 'console' anywhere in the code.
  gem 'web-console', '>= 3.3.0'
end

group :test do
  # Adds support for Capybara system testing and selenium driver
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
  gem 'chromedriver-helper'
end

# Windows does not include zoneinfo files, so bundle the tzinfo-data gem
gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
```

bundle install

Inclua o Pundit no ApplicationController

app/controllers/application_controller.rb

```
class ApplicationController < ActionController::Base
  include Pundit
  protect_from_forgery
end
```

Instale o pundit na aplicação

rails g pundit:install

```
C:\Sites\ror-jp-p3>rails g pundit:install
      create  app/policies/application_policy.rb
```

Após a instalação o pundit irá criar a pasta: **app/policies**

Crie uma nova política para o model User

rails g pundit:policy user

```
C:\Sites\ror-jp-p3>rails g pundit:policy user
      create  app/policies/user_policy.rb
      invoke  test_unit
      create    test/policies/user_policy_test.rb
```

app/policies/user_policy.rb

```
class UserPolicy < ApplicationPolicy
  class Scope < Scope
    def resolve
      scope.all
    end
  end
end
```

Crie o método para testar se o usuário é admin

app/policies/user_policy.rb

```
class UserPolicy < ApplicationPolicy
 def index?
  user.admin?
 end

 class Scope < Scope
  def resolve
   scope.all
  end
 end
end
```

Ajustando mensagens de erro

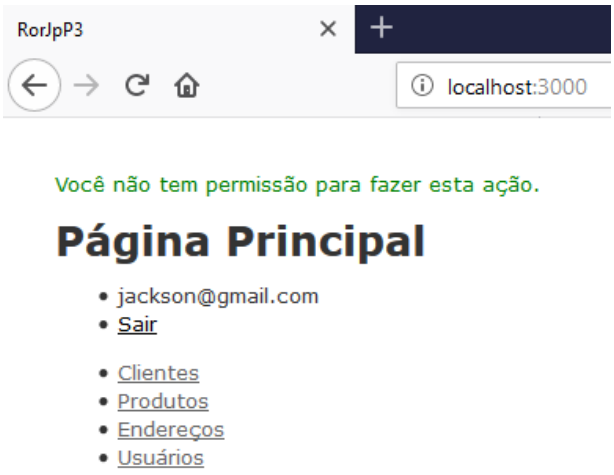
app/policies/user_policy.rb

```
class ApplicationController < ActionController::Base
 protect_from_forgery
 include Pundit

 rescue_from Pundit::NotAuthorizedError, with: :user_not_authorized

 private

 def user_not_authorized
  flash[:notice] = "Você não tem permissão para fazer esta ação."
  redirect_to(request.referrer || root_path)
 end

end
```

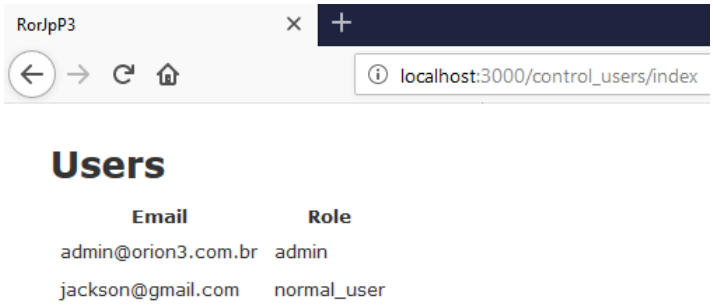
app/views/welcome/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Página Principal</h1>

<% if user_signed_in? %>
 <ul>
  <li><%= current_user.email %></li>
  <li><%= link_to "Sair", destroy_user_session_path, method: :delete %></li>
 </ul>
<% else %>
 <ul>
  <li><%= link_to "Cadastro de novo usuário", new_user_registration_path %></li>
  <li><%= link_to "Logar", new_user_session_path %></li>
 </ul>
<% end %>

<ul>
 <li><%= link_to "Clientes", customers_path %></li>
 <li><%= link_to "Produtos", products_path %></li>
 <li><%= link_to "Endereços", addresses_path %></li>
 <li><%= link_to "Usuários", control_users_index_path %></li>
</ul>
```

Quando um usuário normal tenta acessar a página de usuários:



Quando um administrador tenta acessar a página de usuários:



FAZENDO O MESMO PARA O MODEL ADDRESS

Crie uma nova política para o model Address

rails g pundit:policy address

```
C:\Sites\ror-jp-p3>rails g pundit:policy address
      create  app/policies/address_policy.rb
      invoke  test_unit
      create    test/policies/address_policy_test.rb
```

Testando na view

app/views/addresses/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Listing Addresses</h1>

<% if policy(@addresses).index? %>
 <table>
  <thead>
  <tr>
   <th>Street</th>
   <th>Zip</th>
   <th>Customer</th>
   <th colspan="3"></th>
  </tr>
  </thead>

  <tbody>
  <% @addresses.each do |address| %>
   <tr>
```

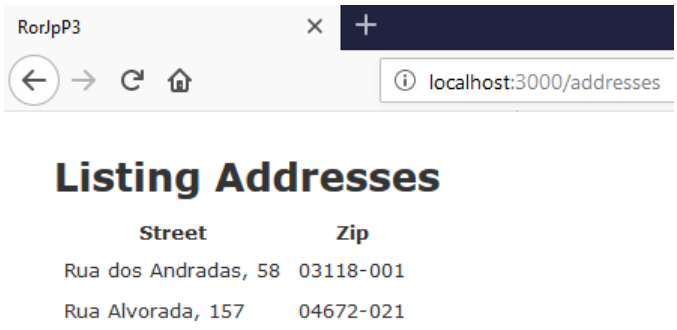
```
      <td><%= address.street %></td>
      <td><%= address.zip %></td>
      <td><%= address.customer.name %></td>
      <td><%= link_to 'Show', address %></td>
      <td><%= link_to 'Edit', edit_address_path(address) %></td>
      <td><%= link_to 'Destroy', address, method: :delete, data: { confirm: 'Are you sure?' } %></td>
    </tr>
  <% end %>
  </tbody>
 </table>

 <br>

 <%= link_to 'New Address', new_address_path %>
<% else %>
 <table>
  <thead>
  <tr>
   <th>Street</th>
   <th>Zip</th>
  </tr>
  </thead>

  <tbody>
  <% @addresses.each do |address| %>
   <tr>
    <td><%= address.street %></td>
    <td><%= address.zip %></td>
   </tr>
  <% end %>
  </tbody>
 </table>

<% end %>
```

Crie o método para testar se o usuário é admin

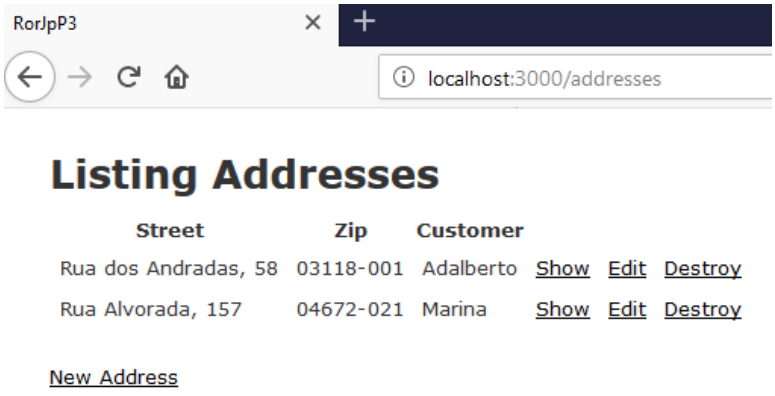
app/policies/address_policy.rb

```
class AddressPolicy < ApplicationPolicy
 def index?
  user.admin?
 end

 class Scope < Scope
  def resolve
   scope.all
  end
 end
end
```

Acessando como usuário normal:



Acessando como administrador:



Otimizando o código

app/views/addresses/index.html.erb

```
<p id="notice"><%= notice %></p>

<h1>Listing Addresses</h1>

<% if policy(@addresses).index? %>
  <%= render partial: "admin_index" %>
<% else %>
  <%= render partial: "normal_user_index" %>
<% end %>
```

app/views/addresses/_admin_index.html.erb

```
<table>
 <thead>
 <tr>
   <th>Street</th>
   <th>Zip</th>
   <th>Customer</th>
   <th colspan="3"></th>
 </tr>
 </thead>

 <tbody>
 <% @addresses.each do |address| %>
   <tr>
     <td><%= address.street %></td>
     <td><%= address.zip %></td>
     <td><%= address.customer.name %></td>
     <td><%= link_to 'Show', address %></td>
     <td><%= link_to 'Edit', edit_address_path(address) %></td>
     <td><%= link_to 'Destroy', address, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
 <% end %>
 </tbody>
</table>

<br>
<%= link_to 'New Address', new_address_path %>
```

app/views/addresses/_normal_user_index.html.erb

```
<table>
 <thead>
 <tr>
  <th>Street</th>
  <th>Zip</th>
 </tr>
 </thead>

 <tbody>
 <% @addresses.each do |address| %>
  <tr>
   <td><%= address.street %></td>
   <td><%= address.zip %></td>
  </tr>
 <% end %>
 </tbody>
</table>
```

AULA 40 - CRIANDO PESQUISAS COM O RANSACK

Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.1'

# Ransack
gem 'ransack'
# Pundit
gem 'pundit'
# Devise
gem 'devise'
# Faker
gem 'faker'
gem 'pry', '~> 0.11.3'
gem 'rb-readline', '~> 0.5.3'
# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '~> 5.2.3'
# Use mysql as the database for Active Record
gem 'mysql2', '>= 0.4.4', '< 0.6.0'
# Use Puma as the app server
gem 'puma', '~> 3.11'
# Use SCSS for stylesheets
gem 'sass-rails', '~> 5.0'
# Use Uglifier as compressor for JavaScript assets
gem 'uglifier', '>= 1.3.0'
# See https://github.com/rails/execjs#readme for more supported runtimes
gem 'duktape'
# Use CoffeeScript for .coffee assets and views
gem 'coffee-rails', '~> 4.2'
# Turbolinks makes navigating your web application faster. Read more:
https://github.com/turbolinks/turbolinks
gem 'turbolinks', '~> 5'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.5'
# Use Redis adapter to run Action Cable in production
# gem 'redis', '~> 4.0'
# Use ActiveModel has_secure_password
# gem 'bcrypt', '~> 3.1.7'

# Use ActiveStorage variant
# gem 'mini_magick', '~> 4.8'

# Use Capistrano for deployment
# gem 'capistrano-rails', group: :development
```

```
# Reduces boot times through caching; required in config/boot.rb
gem 'bootsnap', '>= 1.1.0', require: false

group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug', platforms: [:mri, :mingw, :x64_mingw]
end

group :development do
  # Access an interactive console on exception pages or by calling 'console' anywhere in the code.
  gem 'web-console', '>= 3.3.0'
end

group :test do
  # Adds support for Capybara system testing and selenium driver
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
  gem 'chromedriver-helper'
end

# Windows does not include zoneinfo files, so bundle the tzinfo-data gem
gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
```

bundle install

SIMPLE MODE

Helper

search_form_for

Na view

app/views/products/index.html.erb

```
<p id="notice"><%= notice %></p>

<%= search_form_for @q do |f| %>
 <%= f.label :description_cont %>
 <%= f.search_field :description_cont %>
 <%= f.submit %>
<% end %>

<h1>Listing Products</h1>

<table>
 <thead>
  <tr>
   <th>Description</th>
   <th>Quantity</th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @products.each do |product| %>
   <tr>
    <td><%= product.description %></td>
    <td><%= product.quantity %></td>
    <td><%= link_to 'Show', product %></td>
    <td><%= link_to 'Edit', edit_product_path(product) %></td>
    <td><%= link_to 'Destroy', product, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Product', new_product_path %>
```

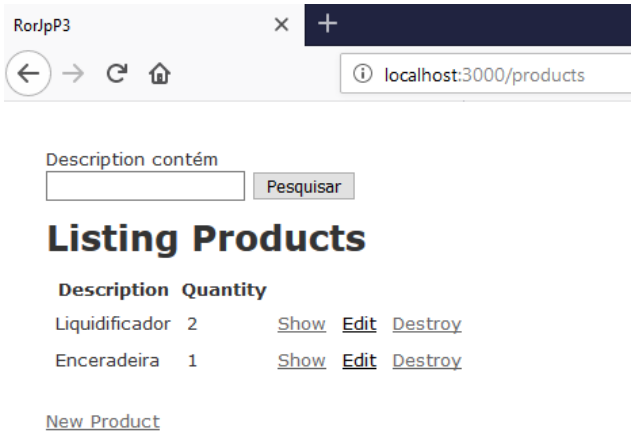
No controller

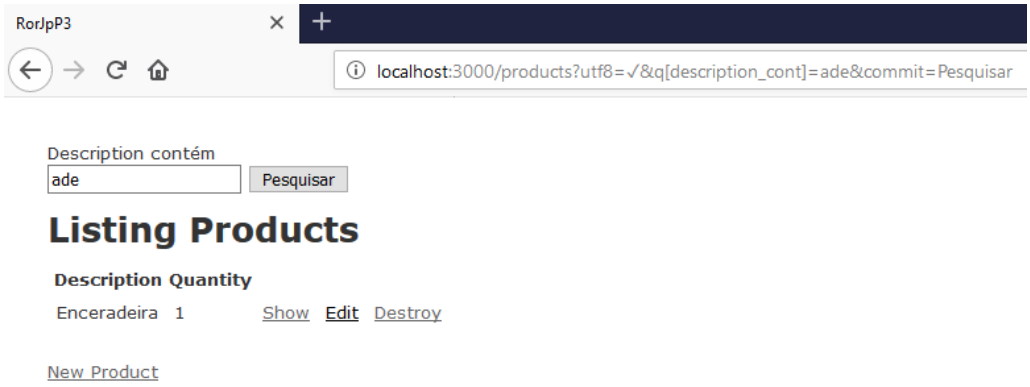
```
def index
 @q = Product.ransack(params[:q])
 @products = @q.result
end
```

app/controllers/product_controller.rb

```
class ProductsController < ApplicationController
 before_action :set_product, only: [:show, :edit, :update, :destroy]
 before_action :authenticate_user!

 # GET /products
 # GET /products.json
 def index
  @q = Product.ransack(params[:q])
  @products = @q.result

  # @products = Product.all
 end

 # GET /products/1
 # GET /products/1.json
 def show
 end

 # GET /products/new
 def new
  @product = Product.new
 end

 # GET /products/1/edit
 def edit
 end

 # POST /products
 # POST /products.json
 def create
  @product = Product.new(product_params)

  respond_to do |format|
   if @product.save
    format.html { redirect_to @product, notice: 'Product was successfully created.' }
    format.json { render :show, status: :created, location: @product }
   else
    format.html { render :new }
    format.json { render json: @product.errors, status: :unprocessable_entity }
```

```
    end
  end
 end

 # PATCH/PUT /products/1
 # PATCH/PUT /products/1.json
 def update
  respond_to do |format|
   if @product.update(product_params)
    format.html { redirect_to @product, notice: 'Product was successfully updated.' }
    format.json { render :show, status: :ok, location: @product }
   else
    format.html { render :edit }
    format.json { render json: @product.errors, status: :unprocessable_entity }
   end
  end
 end

 # DELETE /products/1
 # DELETE /products/1.json
 def destroy
  @product.destroy
  respond_to do |format|
   format.html { redirect_to products_url, notice: 'Product was successfully destroyed.' }
   format.json { head :no_content }
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_product
   @product = Product.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def product_params
   params.require(:product).permit(:description, :quantity)
  end
end
```


RorJpP3
localhost:3000/products
Description contém
Pesquisar
Listing Products
Description Quantity
Liquidificador 2 Show Edit Destroy
Enceradeira 1 Show Edit Destroy
New Product

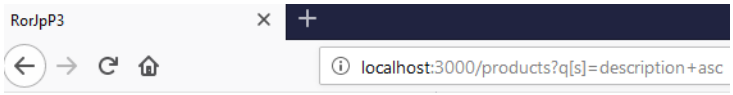

RorJpP3
localhost:3000/products?utf8=✓&q[description_cont]=ade&commit=Pesquisar
Description contém
ade
Pesquisar
Listing Products
Description Quantity
Enceradeira 1 Show Edit Destroy
New Product

Ordenando colunas

sort_link

app/views/products/index.html.erb

```
<p id="notice"><%= notice %></p>

<%= search_form_for @q do |f| %>
 <%= f.label :description_cont %>
 <%= f.search_field :description_cont %>
 <%= f.submit %>
<% end %>

<h1>Listing Products</h1>

<table>
 <thead>
  <tr>
   <th><%= sort_link(@q, :description) %></th>
   <th><%= sort_link(@q, :quantity) %></th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @products.each do |product| %>
   <tr>
    <td><%= product.description %></td>
    <td><%= product.quantity %></td>
    <td><%= link_to 'Show', product %></td>
    <td><%= link_to 'Edit', edit_product_path(product) %></td>
    <td><%= link_to 'Destroy', product, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Product', new_product_path %>
```



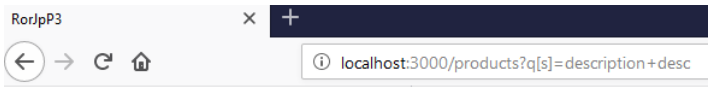
Description contém

Pesquisar

Listing Products

Description ▲	Quantity			
Enceradeira	1	Show	Edit	Destroy
Liquidificador	2	Show	Edit	Destroy

New Product



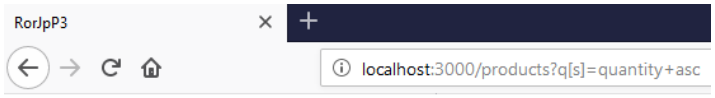
Description contém

Pesquisar

Listing Products

Description ▼	Quantity			
Liquidificador	2	Show	Edit	Destroy
Enceradeira	1	Show	Edit	Destroy

New Product



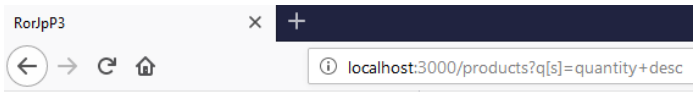
Description contém

Pesquisar

Listing Products

Description	Quantity ▲			
Enceradeira	1	Show	Edit	Destroy
Liquidificador	2	Show	Edit	Destroy

New Product



Description contém

Pesquisar

Listing Products

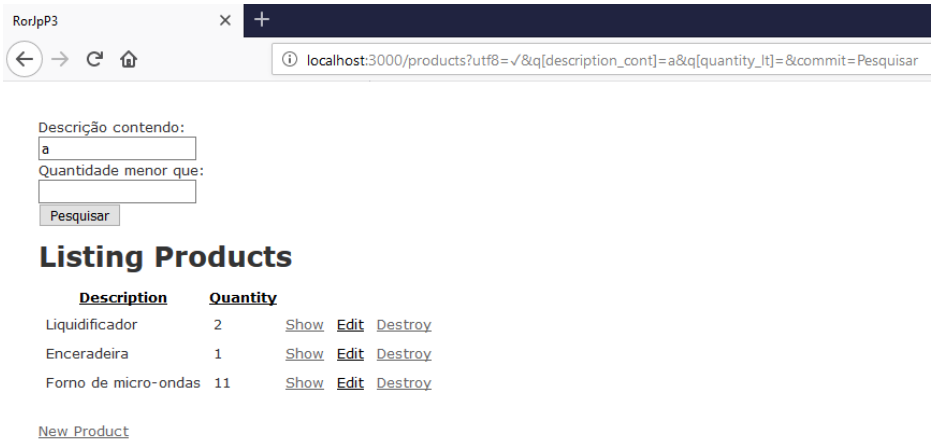
Description	Quantity ▼			
Liquidificador	2	Show	Edit	Destroy
Enceradeira	1	Show	Edit	Destroy

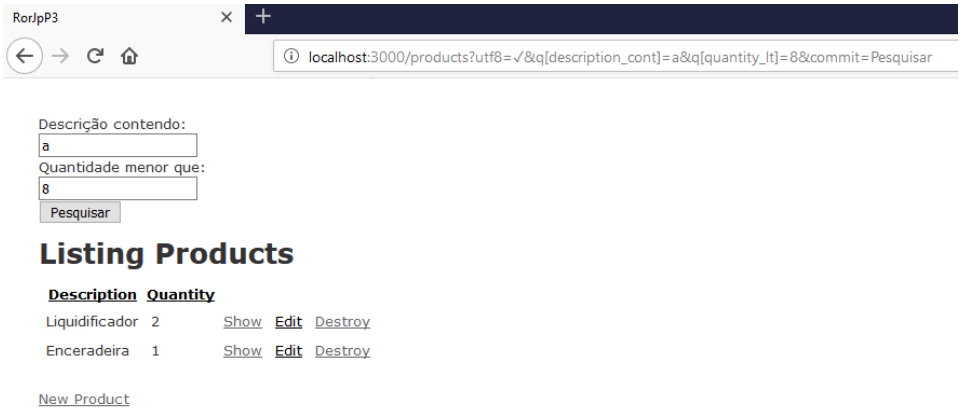
New Product

Adicionando mais opções de pesquisa na view

app/views/products/index.html.erb

```
<p id="notice"><%= notice %></p>

<%= search_form_for @q do |f| %>
 <%= f.label :description_cont, "Descrição contendo:" %>
 <%= f.search_field :description_cont %>

 <%= f.label :quantity_lteq, "Quantidade menor que:" %>
 <%= f.search_field :quantity_lt %>

 <br>

 <%= f.submit %>
<% end %>

<h1>Listing Products</h1>

<table>
 <thead>
  <tr>
   <th><%= sort_link(@q, :description) %></th>
   <th><%= sort_link(@q, :quantity) %></th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @products.each do |product| %>
   <tr>
    <td><%= product.description %></td>
    <td><%= product.quantity %></td>
    <td><%= link_to 'Show', product %></td>
    <td><%= link_to 'Edit', edit_product_path(product) %></td>
    <td><%= link_to 'Destroy', product, method: :delete, data: { confirm: 'Are you sure?' } %></td>
   </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Product', new_product_path %>
```





Pesquisas não restritivas (ou)

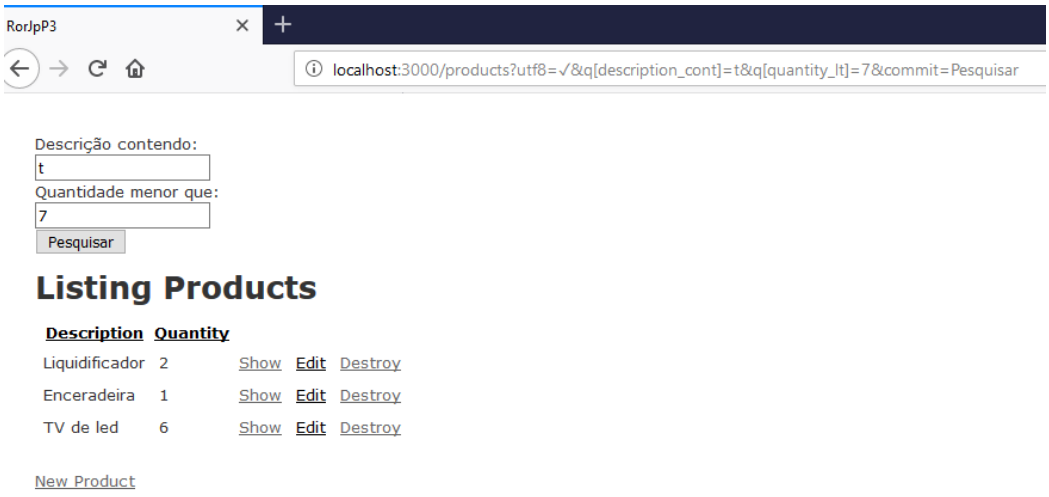
AND => OR

app/controllers/product_controller.rb

```
class ProductsController < ApplicationController
  before_action :set_product, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!

  # GET /products
  # GET /products.json
  def index
    # @q = Product.ransack(params[:q])
    @q = Product.ransack(params[:q].try(:merge, m: 'or'))
    @products = @q.result

    # @products = Product.all
  end

  # GET /products/1
  # GET /products/1.json
```

```
def show
end

# GET /products/new
def new
  @product = Product.new
end

# GET /products/1/edit
def edit
end

# POST /products
# POST /products.json
def create
  @product = Product.new(product_params)

  respond_to do |format|
    if @product.save
      format.html { redirect_to @product, notice: 'Product was successfully created.' }
      format.json { render :show, status: :created, location: @product }
    else
      format.html { render :new }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /products/1
# PATCH/PUT /products/1.json
def update
  respond_to do |format|
    if @product.update(product_params)
      format.html { redirect_to @product, notice: 'Product was successfully updated.' }
      format.json { render :show, status: :ok, location: @product }
    else
      format.html { render :edit }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
end

# DELETE /products/1
# DELETE /products/1.json
def destroy
  @product.destroy
  respond_to do |format|
    format.html { redirect_to products_url, notice: 'Product was successfully destroyed.' }
    format.json { head :no_content }
  end
end
```

```
  private
    # Use callbacks to share common setup or constraints between actions.
    def set_product
      @product = Product.find(params[:id])
    end

    # Never trust parameters from the scary internet, only allow the white list through.
    def product_params
      params.require(:product).permit(:description, :quantity)
    end
end
```



Automatizando o E/OU

app/controllers/product_controller.rb

```
class ProductsController < ApplicationController
  before_action :set_product, only: [:show, :edit, :update, :destroy]
  before_action :authenticate_user!

  # GET /products
  # GET /products.json
  def index
    # @q = Product.ransack(params[:q])
    # @q = Product.ransack(params[:q].try(:merge, m: 'or'))
    @q = Product.ransack(params[:q].try(:merge, m: params[:combinator]))
    @products = @q.result

    # @products = Product.all
  end
```

```
# GET /products/1
# GET /products/1.json
def show
end

# GET /products/new
def new
  @product = Product.new
end

# GET /products/1/edit
def edit
end

# POST /products
# POST /products.json
def create
  @product = Product.new(product_params)

  respond_to do |format|
    if @product.save
      format.html { redirect_to @product, notice: 'Product was successfully created.' }
      format.json { render :show, status: :created, location: @product }
    else
      format.html { render :new }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /products/1
# PATCH/PUT /products/1.json
def update
  respond_to do |format|
    if @product.update(product_params)
      format.html { redirect_to @product, notice: 'Product was successfully updated.' }
      format.json { render :show, status: :ok, location: @product }
    else
      format.html { render :edit }
      format.json { render json: @product.errors, status: :unprocessable_entity }
    end
  end
end

# DELETE /products/1
# DELETE /products/1.json
def destroy
  @product.destroy
  respond_to do |format|
    format.html { redirect_to products_url, notice: 'Product was successfully destroyed.' }
    format.json { head :no_content }
```

```
  end
 end

 private
  # Use callbacks to share common setup or constraints between actions.
  def set_product
   @product = Product.find(params[:id])
  end

  # Never trust parameters from the scary internet, only allow the white list through.
  def product_params
   params.require(:product).permit(:description, :quantity)
  end
end
```

app/views/products/index.html.erb

```
<p id="notice"><%= notice %></p>

<%= search_form_for @q do |f| %>
 <%= f.label :description_cont, "Descrição contendo:" %>
 <%= f.search_field :description_cont %>

 <%= f.select :combinator, [['E','and'],['OU','or']] %>

 <%= f.label :quantity_lteq, "Quantidade menor que:" %>
 <%= f.search_field :quantity_lt %>

 <br>

 <%= f.submit %>
<% end %>

<h1>Listing Products</h1>

<table>
 <thead>
  <tr>
   <th><%= sort_link(@q, :description) %></th>
   <th><%= sort_link(@q, :quantity) %></th>
   <th colspan="3"></th>
  </tr>
 </thead>

 <tbody>
  <% @products.each do |product| %>
   <tr>
    <td><%= product.description %></td>
```

```
      <td><%= product.quantity %></td>
      <td><%= link_to 'Show', product %></td>
      <td><%= link_to 'Edit', edit_product_path(product) %></td>
      <td><%= link_to 'Destroy', product, method: :delete, data: { confirm: 'Are you sure?' } %></td>
    </tr>
  <% end %>
 </tbody>
</table>

<br>

<%= link_to 'New Product', new_product_path %>
```