

Partials – Aula 58

ela permite renderizar uma view dentro da outra, tipo um componente para criar só colocar _ no nome do arq
ex: _arq.html.erb
e dps para utiliza-la usar a teag render

```
<%= render "arq"%>
```

doc:

https://guides.rubyonrails.org/layouts_and_rendering.html#using-partials

Layout – aula 57

Padrao é o application
fica na pasta view/layout
Caso necessite de outro é só criar um arq no padrao arq_nome.html.erb
e no controller especifica o layout necessário
Pode ser muito utilizado para fazer cabeçalho e rodapé.

```
You, 2 minutes ago | 1 author (You)
class CoinsController < ApplicationController
  layout 'adm'
  before_action :set_coin, only: %i[ show edit update destroy ]
  # GET /coins or /coins.json
```

tag responsável em carregar a pagina solicitada
<%= yield %>

doc:

<https://api.rubyonrails.org/classes/ActionView/Layouts.html>

https://guides.rubyonrails.org/layouts_and_rendering.html

Filters – Aula 59

methods que podem ser rodados before, after e around uma action

```
You, 2 minutes ago | 1 author (You)
class CoinsController < ApplicationController
  layout 'adm'
  before_action :set_coin, only: %i[ show edit update destroy ]
  # GET /coins or /coins.json
```

O :set_coin fica la embaixo no private, é por isso q no show esta vazio, pois before_action já faz essa função para não repetir código, no edit, update e destroy.

Obs: tanto no update quanto no destroy temos um notice, ele é usado para mostrar uma msg na tela dps da ação ser feita, e para ele executar é necessario colocar na pagina da view

```
crypto_wallet > app > views > coins > index.html.erb
You, 14 minutes ago | 1 author (You)
1 <p id="notice"><%= notice %></p>
2
3 <h1>Moedas</h1>
4
5 <table>
```

Doc:

https://guides.rubyonrails.org/action_controller_overview.html#filters

Fluxo MVC para new e create - aula 60

dentro do arq _form começa-se com a tag form_with, nela temos alguns parametros a ser preenchidos

```
crypto_wallet > app > views > coins > _form.html.erb
You, 6 days ago | 1 author (You)
1 <%= form_with(model: coin, local: true) do |form| %>
```

local: true, chamada tradicional não terá js

model: é o q esta especificado na new ou edit

```
<%= render 'form', coin: @coin %> You, 6 days ago *
```

@coin virou coin, ent o model é coin e não o @coin

Doc:

https://api.rubyonrails.org/v5.1/classes/ActionView/Helpers/FormHelper.html#method-i-form_with

Permissão parametros- Aula 62

```
end

# Only allow a list of trusted parameters through.
def coin_params
  params.require(:coin).permit(:description, :acronym, :url_image)
end

end You, 7 days ago * app crypto
```

o params chega com os dados e o permit indica quais campos podem ser alterados e serem enviados ao fazer o update e create.

db:seed – Aula 63

Arquivo no qual pode sempre alimentar o DB com dados para teste.
Fica na pasta db/seeds.rb

Rake tasks – Aula 64

rails -T lista todas as tasks

rails g task namespace nomeTask cria a task

as tasks fica em lib/tasks/o arquivo

%x() instrução que permite executar comandos no terminal

Doc:

https://guides.rubyonrails.org/command_line.html#custom-rake-tasks

refatorando - Aula 66

gem tty-spinner

```
lib > tasks > dev.rake
You, 18 hours ago | 1 author (You)
1 namespace :dev do
2   desc "Deleta, cria, faz a migration e sob os dados do DB"
3   task :setup do
4     if Rails.env.development?
5       show_spinner('Apagando BD...') { %x(rails db:drop) }
6
7       show_spinner('Criando BD...') { %x(rails db:create) }
8
9       show_spinner('Migrando BD...') { %x(rails db:migrate)}
10
11      show_spinner('Populando BD...') do
12        %x(rails db:seed)
13      end
14    else
15      puts "Vc não esta em ambiente de desenvolvimento"
16    end
17  end
18 end
19
20 def show_spinner(msg_start, msg_end = 'Concluído')
21   spinner = TTY::Spinner.new("[:spinner] #{msg_start}")
22   spinner.auto_spin
23   yield
24   spinner.success("#{msg_end}")
25 end
26
27 end
28
```

Obs.:

O yield serve para executar um bloco de código dentro do metodo

há duas formas de escrever o código, como mostrado a cima:

```
show_spinner('Migrando BD...') { %x(rails db:migrate)}

show_spinner('Populando BD...') do
  %x(rails db:seed)
end
```


doc:

https://guides.rubyonrails.org/active_record_migrations.html#creating-a-migration

Associações “belongs_to” – Aula 72

rails dbconsole console para mexer com o banco,

.table lista as tabelas

.fullschema lista o schema

belongs_to nessa table tem um id/chave estrangeira q pertence a outra table

Brincando com as associações:

```
[8] pry(main)> history
1: c = Coin.first
2: m = MiningType.first
3: c.mining_type = m
4: c
5: c.save!
6: c.mining_type
7: c.mining_type.description
[9] pry(main)>
```

Doc:

https://guides.rubyonrails.org/association_basics.html#the-belongs-to-association

Associações has_many – Aula 73

has_many – tem muitos/vários

Aqui listaria todas as moedas cadastradas no primeiro tipo de mineração

```
6: history
7: m = MiningType.first
8: m.coins
[10] pry(main)>
```

Agora aqui eu estou jogando uma moeda dentro do primeiro tipo de mineração, pois o « faz isso

```
10: c = Coin.second
11: c
12: m
13: m.coins << c
14: m.save!
15: m
16: m.coins
[16] pry(main)>
```

https://guides.rubyonrails.org/association_basics.html#the-has-many-association

na associação para criar a moedas temos os seguintes atributos

```
coins = [
    { description: "Bitcoin",
      acronym: "BTC",
      url_image: "https://i.imgur.com/1083000.png",
      mining_type: "Mining" }
  ]
```

```
coins = [
    { description: "Bitcoin",
      acronym: "BTC",
      url_image: "https://e7.pngegg.com/pngimages/261/204/p",
      mining_type: MiningType.find_by(acronym: 'PoW')},
```

por que não o where? Da erro pois o where busca vários obj e o rails espera só um Find? Não funciona porque quando usa o find espera-se q seja o id, e nesse caso estamos usando o acronym

logo sopra o find_by

o `.map` é como o `.each` a diferença que ele altera as informações e devolve um array

```
a = [1,2,3,4,5]
```

quando o cod é só uma linha pode-se abrir chave ao invés do do end

```
[7] pry(main)> a.map do |i|
[7] pry(main)* i*2
[7] pry(main)* end
=> [2, 4, 6, 8, 10]
[8] pry(main)> a.map { |i| i*2 }
=> [2, 4, 6, 8, 10]
[9] pry(main)> █
```

O .map pode ser usado no model para pegar informações

```
[ "ZCash", "ZEC" ]
[10] pry(main)> c.map {|coin| coin.description}
=> ["Bitcoin", "Ethereum", "Dash", "Iota", "ZCash"]
[11] pry(main)> c.map(&.description)
=> ["Bitcoin", "Ethereum", "Dash", "Iota", "ZCash"]
[12] pry(main)> c.map {|coin| [ coin.description, coin.acronym ]}
=> [ ["Bitcoin", "BTC"],
    ["Ethereum", "ETH"],
    ["Dash", "DASH"],
    ["Iota", "IOT"],
    ["ZCash", "ZEC"] ]
[13] pry(main)> c.map([&.description, &.acronym])
SyntaxError: unexpected &, expecting ']'
c.map([&.description, &.acronym])
```

mas como o .map para pegar muitas informações fica com muito extenso também tem o .pluck, que cumpre também a mesma função

```
[14] pry(main)> c.pluck(:description)
=> ["Bitcoin", "Ethereum", "Dash", "Iota", "ZCash"]
[15] pry(main)> c.pluck(:description, :acronym)
=> [["Bitcoin", "BTC"],
    ["Ethereum", "ETH"],
    ["Dash", "DASH"],
    ["Iota", "IOT"],
    ["ZCash", "ZEC"]]
[16] pry(main)> []
```

select – Aula 76

doc:

<https://api.rubyonrails.org/v5.2.0/classes/ActionView/Helpers/FormOptionsHelper.html#method-i-select>

padronização MVC – Aula 77

nunca se deve acessar o model pela view quando se usa a arquitetura MVC, esse papel deveria ser do controller, logo cria-se um methodo no controller para esse acesso.

Nesse methodo terá uma variável de instancia q sera usada na view

Ao criar o methodo no private la não pode esquecer de invocá-lo no before_action

YAML – Aula 78

Para funcionar precisa estar com a indentação correta, 2 espaços ou 1 solft tab

Internalização i18n – Aula 79

Serve para sua aplicação tenha varias idiomas

Para ativar precisa de:

Instalar a gem, colocando no gemfile e fazendo o bundle
gem 'rails-i18n', '~> 5.1'

criar um arq locale.rb dentro de config/initializer

dentro dele colocar:

I18n.available_locales = [:en, 'pt-BR'] aqui mostra quais os idiomas disponiveis

I18n.default_locale = :en aqui qual o idioma padrao

Esses arquivos são renderizados ao iniciar a app, ent ao editá-los é necessário parar o server e start novamente

doc:

<https://guides.rubyonrails.org/i18n.html>

Usando i18n – Aula 80

I18n.t() método utilizado para receber uma tradução de um arquivo yml q estão na pasta config/locales

para adicionar novos idiomas só criar novos arq com a sigla do idioma dentro dessa mesma pasta

é comum você encontrar somente t('chave') *não pode esquecer das '' na chave*

Já para data e hora existe o metodo I18n.l() ou somenten l(), serve para deixar no formato local
ex: l(Date.today)

quando tem subtopicos colocar . Para inteligar
ex t('msg.menu')

Traduzindo model – Aula 81

Model.human_attribute_name(attribute)

ex: Coin.human_attribute_name(description)

ou

ex: @coins.human_attribute_name(description) a variavel de sessão do controller também funciona

O model temos como mostrar no singular ou plural

```
'pt-BR':  
  activerecord:  
    models:  
      coin:  
        one: Moeda  
        other: Moedas  
      mining_type:  
        one: Tipo de mineração  
        other: Tipos de minerações  
      attributes:  
        coin:
```

Model.model_name.human(count: 1 ou 2) 1 singular 2 plural

ex: Coin.model_name.human(count: 2)

doc:

<https://guides.rubyonrails.org/i18n.html#translations-for-active-record-models>

Cookies e sessions – Aula 83

HTTP é stateless, não sabe quem fez a requisição nem para onde vai, ele só pega trata e devolve

Stateful ele consegue burlar o stateless do HTTP através dos cookies, pegando algumas informações e guardando no navegador

já as sessions pega as informações e guarda no servidor

syntax no controller:

```
cookies[:curso] = "O dado"
```

```
session[:curso] = "O dado"
```

e dps colocar na view

```
<h4>Testando Cookie e session</h4>
<p><%= cookies[:curso] %></p>
<p><%= session[:curso] %></p>
```

Usando app em vários idiomas – Aula 84

Dentro de controller/applicationController.rb criamos um metodo para já iniciar com o idioma padrao da pessoa

```
app > controllers > application_controller.rb
You, 4 minutes ago | 1 author (You)
1  class ApplicationController < ActionController::Base
2    before_action :set_locale
3
4    def set_locale
5      if params[:locale]
6        cookies[:locale] = params[:locale]
7      end
8
9      if cookies[:locale]
10       if I18n.locale != cookies[:locale]
11         I18n.locale = cookies[:locale]
12       end
13     end
14   end
15 end
16
```

dps onde quiser colocar para mudar o idioma

```
app > views > welcome > index.html.erb
You, 2 minutes ago | 1 author (You)
1  <h1>Seja bem vindo</h1>
2
3  <p><%= link_to 'Pt-BR', '/?locale=pt-BR' %></p>
4  <p><%= link_to 'English', '/?locale=en' %></p>
5
```

ou

```
app > views > welcome > index.html.erb
You, 1 second ago | 1 author (You)
1 <h1>Seja bem vindo</h1>
2
3 <p> <%= link_to 'Pt-BR', root_path(locale: 'pt-BR')%> </p>
4 <p> <%= link_to 'English', root_path(locale: 'en')%> </p>
5
```

Conhecendo Assets Pipeline – Aula 86

Assets Pipeline é usado para minificar o css e js para aumentar na performance da aplicação, também utilizado para misturar css com ruby e ruby com js

é uma gem chamada sprockets-rails

Fingerprint é usado para evitar o cache do navegador, tocando assim sempre o nome do arquivo css e js para o servidor sempre recarrega-lo e ver se tem uma nova alteração

Os assets devem ficar em pastas específicas.

- app/assets: Para assets criados pelo próprio Rails
- lib/assets: Para assets que você mesmo criou
- vendor/assets: Para assets que você “pegou” de terceiro

isolando assets por controller – aula 87

usado para não carregar arquivos desnecessários e prejudicar na performance da aplicação

ex: pagina de moedas e pagina de mineração, não precisa carregar os arquivos de moedas se estou na pagina de mineração

na pasta app/assets/javascript ou app/assets/stylesheets temos os arquivos application, no qual tem o código:

```
//= require tree
```

Esse código faz com que todos os arquivos js e css sejam carregados o que não é bom, pois prejudica a performance da aplicação

```
params[:controller]
```

mostra qual é o controller que está usando, e para que não carregue todos os arquivos de todos os controllers exclui a linha require tree e na view onde estiver requerindo os arquivos css e js duplicar a linha e passar o params[:controller], muitas vezes é em view/layouts/application.html.erb

```

<%= csp_meta_tag %>

<%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %>
<%= stylesheet_link_tag params[:controller], media: 'all', 'data-turbolinks-track': 'reload' %>

<%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
<%= javascript_include_tag params[:controller], 'data-turbolinks-track': 'reload' %>
</head>

```

Mas dps disso dá um erro na pre-compilação, pois rails não esta reconhecendo nosso arquivos
temos q ir em config/initializers/assets.rb e adicionar a linha com o nome do assets

```

Rails.application.config.assets.precompile += %w( welcome.css
                                                    coins.css)

```

```

Rails.application.config.assets.precompile += %w(cable.js
                                                    coins.js
                                                    mining_
                                                    types.js
                                                    welcome.js)

```

doc:

https://guides.rubyonrails.org/asset_pipeline.html#coding-links-to-assets

Usando Asset Pipeline – Aula 88

misturando css com ruby

O arq necessita estar com a extensão arq.css.erb, agr o arq css executa ruby

```

background-color: <%= Rails.env.development? ? 'green' : 'red'%>;

```

misturando scss com ruby

O arq necessita estar com a extensão arq.scss.erb, agr o arq scss executa ruby

image q você cria lib/assets/cria pasta images

```

background-image: asset-url('rails.png');

```

misturando js com ruby

O arq necessita estar com a extensão arq.js.erb, agr o arq js executa ruby

```

<% msg = Time.now.hour < 12 ? 'Bom dia!' : 'Olá!!!' %>
alert("<%= msg %>");

```

Usando task para pré-compilar - aula 89

Quando tentamos subir a aplicação em modo produção sempre dá um erro, porque?

O rails por padrao cria uma rota virtual padrao carregando todos os assets ao fazer rails s, mas ao fazer rails s -e production o rails entende q não é mais para ele simular o seu servidor completo porque é para um servidor da internet fazer isso. Logo, para q funcione temos q usar uma task

```
rails assets:precompile
```

assim aparece em public uma pasta chamada assets com todos os assets

porem o DB não esta para producao ainda, ent temos q primeiro cria-lo com:

```
RAILS_ENV=production rails db:create db:migrate
```

dps subir a app

```
rails s -e production
```

mas ainda sim não subiu todos os arq estaticos, para isso é necessario ir em config/environments/production.rb para pegar a variavel e escrever o codigo a baixo

rodar no terminal

```
RAILS_SERVE_STATIC_FILES=true rails s -e production
```

apaga a pasta assets do public

```
rails assets:clobber
```

usando biblioteca js Vendor/assets - Aula 90

A três formas de usar uma biblioteca js

A primeira é baixando dentro da pasta vendor/assets/javascripts caso não tenha a pasta assets criá-la

copiar o link de download da biblioteca e colocar no terminal

wget link isso estando dentro da pasta vendor/assets/javascripts

ex: wget <https://code.jquery.com/jquery-3.6.1.js>

dps temos q ir em app/assets/javascripts/application.js.erb para carregar a biblioteca dentro da aplicação, colocando o nome do arq da biblioteca

escrever : //= require nomeArqBiblioteca

ex //= require jquery

ouu

porem outra forma é na pasta layout em application.html.erb você importar o link da biblioteca, assim:

```
<%= javascript_include_tag 'application', 'data-turbolinks-track' => 'reload' %>
<%= javascript_include_tag 'jquery', 'data-turbolinks-track' => 'reload' %>
<%= javascript_include_tag 'notify', 'data-turbolinks-track' => 'reload' %>
```

dps tem q pre-compilar em config/initializers/assets.rb e na parte js importar jquery.js e notify.js

```
Rails.application.config.assets.precompile += %w( cable.js
  coins.js
  jquery.js
  mining_types.js
  notify.js
  welcome.js
)
```

Usando biblioteca js rails-assets.org – Aula 91

faciliar o uso de biblioteca, pois sua proposta é pegar uma biblioteca js e transformar em uma gem

acessa o site: rails-assets.org

coloca no gemfile a gem

roda o bundler

ir em app/assets/javascripts/application.js.erb para carregar a biblioteca dentro da aplicação, colocando o nome do arq da biblioteca

escrever : //= require nomeArqBiblioteca

ex //= require jquery

Yarn- Aula 93

utilizado para gerenciar as bibliotecas js

ao dar um yarn init criara um arquivo na / chamado package.json onde colocaremos as biblioteca, é tipo o gemfile para o js

para usar você vai no site do yarn, pesquisa a biblioteca necessaria, pega o comando para instala-la, dps roda no terminal e é para aparecer no arquivo package.json a biblioteca, no yarn.lock é para estar também e na pasta node_modules é para ter uma pasta com o nome da biblioteca

no final ir em app/assets/javascripts/application.js.erb para carregar a biblioteca dentro da aplicação, colocando o nome da pasta da biblioteca criada em node modules

escrever : //= require nomeArqBiblioteca

ex `//= require jquery`

caso não encontre tem q passar o caminho do arq da biblioteca:

ex: `//= require notify-js-legacy/notify`

e algumas vezes você tem q informar o caminho dentro da pasta dist

`//= require bootstrap/dist/js/bootstrap`

Obs: quando sobe o projeto do github a pasta node_modules pode não ir junto, para isso você tem q rodar um yarn install caso precise baixar seu projeto em algum lugar

Bootstrap – Aula 94

para carregar você tem q instalar o bootstrap, pode ser pelo yarn

dps instalar o popper

todos deve ser requeridos nos arquivos da pasta app/assets/ tanto no js quanto no css.

Aula 97

faz com q coloque o nome do model dependendo do tanto de elementos no bd

`<%= Coin.model_name.human(count: @coins.count)%>`

Aula 99

submit precisa no nil antes para conseguir colocar as class do bootstrap

ex: `<%= form.submit nil, class:"btn btn-primary">`

Aula 101

quando o forms não tá interligado para enviar dados no bd, o padrao form.label e form.text_field vira label_tag e text_field_tag

Ambienteeee

mysql exit
postgres \q

rodar mysql

para listar o status do docker
docker ps -a

caso não esteja habilitada é só executar
docker start id

se aparecer algo de sem permissão executa
sudo usermod -aG sudo maria

dps q só entrar no projeto q tem q configurar o database.yml com:

```
default: &default
  adapter: mysql2
  encoding: utf8
  pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>
  username: root
  password:
  host: 127.0.0.1
```

caso queira parar o docker só executar:
docker stop id

Segunda Aplicação, site de perguntas

desabilitando configurações padrão - aula 120

rails new testeapp -T faz com q o ambiente de teste não seja criado junto com a aplicação.

Isso faz com q no config/application.rb tenha uma linha q configure não criar a pasta teste:

```
config.generators.system_tests = nil
```

E por desabilitar, nesse arq ele tem q requerir todos os frameworks necessários:

```
3 require "rails"
4 # Pick the frameworks you want:
5 require "active_model/railtie"
6 require "active_job/railtie"
7 require "active_record/railtie"
8 require "active_storage/engine"
9 require "action_controller/railtie"
10 require "action_mailer/railtie"
11 require "action_view/railtie"
12 require "action_cable/engine"
13 require "sprockets/railtie"
14 # require "rails/test_unit/railtie"
```

perceba q o de teste esta desabilitado

Enquanto isso no projeto q esta com a pasta test o arquivo esta somente com um rails/all:

```
3 require 'rails/all'
4
5 # Require the gems listed in Gemfile, including any gems
```

então caso já tenha o projeto criado e não queira mais gerar os teste basta somente comentar o rails/all e copiar esse outros codigos necessarios.

dps basta apagar a pasta teste

para criar os controller pra ver se funcionou a configuração:

rails g controller welcome index

rails d controller welcome

e caso ainda criar com os test ver se o spring esta rodando:

spring status

spring stop

Outra forma é dentro mesmo de config/application.rb com o site

<https://guides.rubyonrails.org/generators.html#customizing-your-workflow> pesquisar o q quer q não gere e colocar dentro do bloco de codigo:

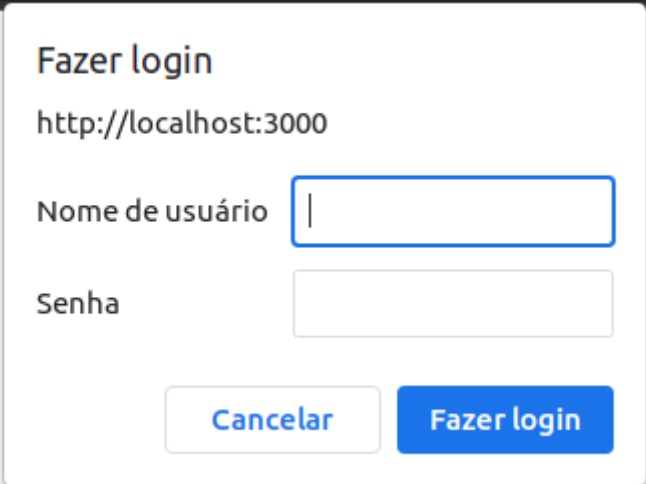
```
config.generators do |g|
  g.orm :active_record
  g.template_engine :erb
  g.test_framework :test_unit, fixture: false
  g.stylesheets false
  g.helper false
end
```

Autenticação básica http – Aula 121

Tbm conhecido como basic auth, é a forma mais básica de autenticação, sendo um recurso do navegador, fazer reload não faz pedir a senha dnv pois fica guardado em cache.

O código fica no controller

```
http_basic_authenticate_with name: "maria", password: "12"
```



Fazer login

http://localhost:3000

Nome de usuário

Senha

doc: <https://api.rubyonrails.org/classes/ActionController/HttpAuthentication/Basic.html>

Criando logins e Admin profile - Aula 124

gem 'devise'

tudo na doc: <https://github.com/heartcombo/devise#getting-started>

add no gemfile a gem 'devise' e rodar bundle

com essa gem novos generate seram criados, ent logo dps da instalação deve rodar o comando

rails generate devise:install

no qual cria esses 2 arq

```
maria@maria-aspires:~/Documentos/cursoRuby/modulo_05/time_to_answer$ rails generate devise:install
Running via Spring preloader in process 9737
  create  config/initializers/devise.rb
  create  config/locales/devise.en.yml
=====
```

dps em in config/environments/development.rb copiar o seguinte comando

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

dps disso tera q criar um devise com o nome dos models necessarios com o comando

```
rails generate devise MODEL
rails generate devise profile
```

Ai temos q criar as views, mas como temos mais de um model só o comando
rails generate devise:views não dara certo. Ent temos q ir em config/initializers/devise.rb
e descomentar a linha config.scoped_views = true

dps disso só criar as views ex rails generate devise:views users no plural

ai só rodar rails db:migrate

Ativando I18n – Aula 125

igual aula 79

Ativando I18n para devise- aula 126

para mostrar o erro na tela em app/views/admins/sessions/new.html.erb
esse codigo serviu para mostrar q o i18n esta funcionando

```
<% flash.each do |key, value| %>
<div class="flash <%= key %>"> <%= value %> </div>
<% end %>
```

dps entra em <https://github.com/heartcombo/devise/wiki/I18n>

escolhe a língua, copia o codigo, dps cria um arq em na pasta locale

com o nome devise.nomeLingua.yml

Ativando I18n para models- aula 127

só criar o arq em locale.rb
models.NomeLingua.yml
e colocar a estrutura para model

em:

```
activeecrod:
  models:
    user:
      attributes:
```

Criando o Backoffice para adm e profile – aula 128

```
rails g controller admins_backoffice/welcome index
rails g controller profiles_backoffice/welcome index
```

tá no plural porque as pastas na view também estão no plural

poem quando criamos assim essas view não estão protegidas pois eu consigo acessa-las pela url

Ajustando layout profiles e admin – Aula 129

Crie novos controllers

- rails g controller admins_backoffice
- rails g controller profiles_backoffice

● Altere a herança dos controllers que já existiam

- class ProfilesBackoffice::WelcomeController < ProfilesBackofficeController
- class AdminsBackoffice::WelcomeController < AdminsBackofficeController

● Crie os novos arquivos de layout baseado no application.html.erb

- /views/layouts/admins_backoffice.html.erb
- /views/layouts/profiles_backoffice.html.erb

● Indique os novos controllers usam esses layouts

- ProfilesBackofficeController
 layout 'profiles_backoffice'
- AdminsBackofficeController
 layout 'admins_backoffice'

Estamos fazendo isso para q as pastas fiquem separadas certinho com cada coisa, e que cada parte herde o layout correto

Ajustando layout site – Aula 130

Precisaremos remover o controller welcome e criar um novo baseado no namespace site, depois vamos criar um controller site para seu o controller "pai"

- rails destroy controller welcome
- rails g controller site::welcome index
- rails g controller site

● Por fim, vamos alterar a herança do controller e adicionar o layout

- class Site::WelcomeController < SiteController

● Agora vamos adicionar o layout , esse arq do layout cria-se na view

```
class Site::WelcomeController < SiteController
  layout :site
```

Uma pitada de metaprogramação – aula 131

Usando o define_method conseguimos todas as vezes q criar um obj criar um metodo para ele com o seu próprio nome, isso utilizando o self

```
class User
```

```
  def initialize(name)
    User.create_method(name)
  end

  def self.create_method(name)
    define_method :"speak_#{name.downcase}!" do
      "Hello, everyone! I'm #{name}!"
    end
  end
end
```

```
end
```

```
u = User.new("Jackson")
puts u.speak_jackson!
```

Usando devise para proteger os backoffices – Aula 132

doc: <https://github.com/plataformatec/devise>

na doc nos motra q para proteger utilizamos no controller

```
before_action :authenticate_nomeModel!
```

```
Ex: before_action :authenticate_admin!
```

Corrigindo profile para users - Aula 133

Criando rake task – aula 134

começa criando a task, q fica em lib

rails g task dev setup, dps copia do crypto wallet kkkk

Obs.: no crypto usamos o find_or_create_by, já nesse o create, porque?

O devise já faz a própria verificação se já existe o usuário cadastrado, ent não é necessário o find

Criando link de logoff – Aula 135

alguns helpers

user_signed_in? Diz de tá logado

current_user quem é o atual `current_admin.email`

E para fazer o logoff é só pegar o route path com o delete session e colocar como link to, especificando o metodo delete

ex: `<%= link_to 'Fazer logOff[adm]', destroy_admin_session_path, method: :delete %>`

mas ainda o devise faz o logoff de todas as session ao mesmo tempo, tanto do admin quanto do user, para isso temos q configurar dentro de config/initializer/devise.rb

```
config.sign_out_all_scopes = false
```

disponibilizando temas localmente – Aula 137

Dentro do projeto temos a pasta public, q é uma pasta compartilhada q nos deixa acessar na url qualquer arquivo, logo iremos criar dentro uma pasta chamada templates para acessar mais facilmente

dentro de templates faremos um git clone do template escolhido e dps só copiar o relative path do arq index do template e colocar na url

inicializando o yarn – Aula 138

yarn init

portando o tema sbadmin para admin – aula 139

primeiro deve remover o turbolinks do gemfile, assets/application.js e views/layouts retirar a linhas

```
media: 'all', 'data-turbolinks-track': 'reload'  
'data-turbolinks-track': 'reload'
```

dps é eu copiar o código da página que eu quero e mover para o arquivo do layout correspondente nesse caso o admin, mesclar com as tags que já havia antes.

Dps dentro de assets/stylesheets/application.css tirar o require_tree

logo dps é instalar as bibliotecas que são necessárias para utilizar o tema com suas devidas versões, nisso olhamos as versões dentro da pasta do projeto do tema e instalamos com yarn add, yarn add [bootstrap@3.3.7](#)

portando o tema sbadmin para admin part2 – aula 140

dps de instalamos as bibliotecas necessárias, vemos que as mesmas estão em node_modules, mas quando olhamos o caminho que está na página está na pasta vendor, logo devemos mudar os paths para que a aplicação funcione corretamente.

Nisso, na view do layout importamos o CSS disponível no assets/stylesheets/admin_backoffice.scss

```
<%= stylesheet_link_tag 'admins_backoffice' %>
```

dps vamos pegar todas as bibliotecas de importação e levar para assets/stylesheets/admin_backoffice.scss com os paths certos

```
<!-- Bootstrap v3.3.7 Core CSS -->
<link href="../../vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">

<!-- MetisMenu v1.1.3 CSS -->
<link href="../../vendor/metisMenu/metisMenu.min.css" rel="stylesheet">

<!-- Custom CSS -->
<link href="../../dist/css/sb-admin-2.css" rel="stylesheet">

<!-- Custom Fonts 4.6.3 -->
<link href="../../vendor/font-awesome/css/font-awesome.min.css" rel="stylesheet" type="text/css">
```

não pode ser esse vendor, pois é do projeto antigo, agora só procurar onde estão as bibliotecas instaladas e colocar o require

```
admins_backoffice.scss M x  admins_backoffice.html.erb M
app > assets > stylesheets > admins_backoffice.scss
You, 16 seconds ago | 1 author (You)
1 // Place all the styles related to the admins_backoffice here.
2 // They will automatically be included in application.css
3 // You can use Sass (SCSS) here: http://sass-lang.com/
4
5
6 /*
7  *= require bootstrap/dist/css/bootstrap
8  *= require metismenu/dist/metisMenu
9  *= sb-admin-2
10 *= require font-awesome/css/font-awesome
11 */
12
```

Obs: o sb-admin não está com o path porque é um arquivo próprio criado pela gente então está dentro da pasta lib/assets/stylesheets

Agora a mesma coisa que fizemos para o CSS faremos para o JS também

Porem quando startamos o server da erro porque ainda é necessario precompilar os arquivos

```
<%= stylesheet_link_tag 'admins_backoffice' %>
```

ent vamos em `config/initializers/assets.rb` e descomentamos a linha

```
Rails.application.config.assets.precompile += %w(
```

e add admins_backoffice.js e admins_backoffice.css, ficando:

```
Rails.application.config.assets.precompile += %w( admins_backoffice.css  
admins_backoffice.js )
```

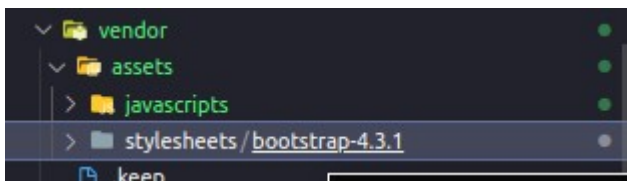
Dps não podemos esquecer de também pre-compilar os arq css nossos criados q estao em lib

```
#esta dentro de app/assets/  
Rails.application.config.assets.precompile += %w( admins_backoffice.css admins_backoffice.js )  
  
#esta dentro de lib/assets/ You, 1 second ago * Uncommitted changes  
Rails.application.config.assets.precompile += %w( sb-admin-2.js sb-admin-2.css )
```

portando o tema gentellella para user– aula 141

é o mesmo processo de cima só q temos q nos atentar as versoes das bibliotecas, pois para admin usamos o jquery 3, na no gentellella usa a 2, como resolver?

Na pasta de vendor podemos colocar dentro de assets/javascripts ou stylesheets a pasta da biblioteca



dps pegar seu path colocar na app/assets e

```
app > assets > stylesheets > users_backoffice.scss  
You, 11 minutes ago | 1 author (You)  
1 // Place all the styles related to the users_backoffice co  
2 // They will automatically be included in application.css.  
3 // You can use Sass (SCSS) here: http://sass-lang.com/  
4  
5 /*  
6 *= require bootstrap-4.3.1/dist/css/bootstrap  
7 *= require font-awesome/css/font-awesome  
8 *= require nprogress/nprogress
```

dps compilar esse mesmo path no config/initializer/assets

```
#esta dentro de app/assets/  
Rails.application.config.assets.precompile += %w( admins_backoffice.js admins_backoffice.css users_backoffice.js users_  
  
#esta dentro de lib/assets/  
Rails.application.config.assets.precompile += %w( sb-admin-2.js sb-admin-2.css custom.js custom.css )  
  
#esta dentro de vendor/assets/  
Rails.application.config.assets.precompile += %w( jquery-2.2.3/dist/jquery.js bootstrap-4.3.1/dist/css/bootstrap.css)
```

Ajustando logoff para admin – aula 143

Encontrar o local no qual no template esta o local de sair e coloca o codigo

```
<%= link_to destroy_admin_session_path, method: :delete do %>
  <i class="fa fa-sign-out fa-fw"></i> Sair
<% end %>
```

Task para cadastrar outros admin – aula 146

para add dados fakes utiliza a gem ‘faker’

```
desc "Cadastra varios adms"
task :add_extras_admin, :environment => :development do
  10.times do |i|
    Admin.create!(
      email: Faker::Internet.email,
      password: 123456,
      password_confirmation: 123456)
  end
end
```

Para saida no rails c temos duas gem
gem ‘pry-rails’

gem ‘awesome_print’ para usar ap comando
ex: ap Admin.all

Criando controller admins – aula 147

cria o controller

rails g controller AdminsBackoffice::Admins index

Ajustando rotas admin – Aula 149

em routes quando coloca o resources ele cria todas as padroes

não precisa fazer get ‘admins/edit/:id’, to ‘admins#edit’

porem quando colocamos o resource o path muda, ent não podemos esquecer de mudar no codigo se foi utilizado
alguma vez o path antigo

com only

```
resources :admins, only: [:index, :edit]
```

Ajustando a rota edit admin- Aula 150

dps de criar a rota :edit no routes temos q criar o arq q esta a edição q sera o edit.html.erb dps vincular a algum link_to,
nesse projeto assim

```
<%= link_to edit_admins_backoffice_admin_path(admin), class:"btn btn-primary btn-circle" do %>
  <i class="fa fa-pencil"></i>
<% end %>
```

e ao clicarmos no link podemos observar q no log ele passa o params do id, ent com isso conseguimos pegá-lo para
exibir na tela

ent no controller podemos fazer um edit com

```
@admin = Admin.find(params[:id])
```

Finalizando forms - Aula 152

como estamos usando namespace, ent o formulario precisa ser com o form_with

```
<%= form_with(model: [ :admin, @post ]) do |form| %>
  . . .
<% end %>
```

```
<%= form_with(model: [ :admins_backoffice, @admin ]) do |form| %>
```

na primiar coloca o nome do namespace e no segundo a varia q esta no controller do editá-los
não pode esquecer de colocar o :update na route

Ao ent clicarmos no botao de submit é para aparecer no log Patch ou put, mostrando q foi iniciado, q observamos q ele manda como JS, e como não queremos temos q adicionar o local:true

```
<%= form_with(model: [ :admins_backoffice, @admin ], local:true) do |form| %>
```

por tem um form_with a variavel agora para utilizarmos sera o form

ent termos

```
<%= form.label :email %>
<%= form.text_field :email, class:"form-control", placeholder:"Email do adm" %>
```

e para salvar percebmos q na url esta o id do adm ent conseguimos pega-lo pelo params no controller atraves de:

```
@admin = Admin.find(params[:id])
```

e por uma questao de segurança você tem q permitir quais dados serão enviados e quais não, para q alguém não mande informações extra

```
params_admin = params.require(:admin).permit(:email, :password,
:password_confirmation)
```

e da onde saiu esse :admin do require? Da chave q esta no log

```
ES3dPJS3t04PnVWZQ91F1BZNONj8DqXm8fCE3t0+e01WSZK0VjwpDV30vN9t9SSVqvXg==
, "admin"=>{"email"=>"joe_wisozk@borer.org", "password"=>"[FILTERED]",
"password_confirmation"=>"[FILTERED]"}, "id"=>"3"}
Admin Load (0.3ms)  SELECT  "admins" * FROM "admins" WHERE "admins"
```

Ai dps q deu tudo certo para ent sermos redirecionados ao index só fazer o if eles

```
if @admin.update(params_admin)
  redirect_to admins_backoffice_admins_path, notice: "Adm atualizado!"
else
  render :edit
end
```

Mostrando erros do formulario – Aula 153

No console conseguimos buscar o erro atraves de comando logo podemos imprimir na view

```
irb(main):008:0> a.errors
=> #<ActiveModel::Errors:0x000055e9c22049f8 @base=#<Admin id: nil, email: "", created_at: nil, updated_at:
o pode ficar em branco"], :password=>["não pode ficar em branco"]], @details={:email=>[:error=>:blank]}, :
>
irb(main):009:0> a.errors.any?
=> true
irb(main):010:0> a.errors.full_messages
=> ["Email não pode ficar em branco", "Senha não pode ficar em branco"]
irb(main):011:0> 
```

```

<% if @admin.errors.any? %>

  <ul>
    <% @admin.errors.full_messages.each do |message| %>
      <li> <%= message %> </li>
    <% end %>
  </ul>
<% end %>

```

Alterando senha adm – Aula 154

No caso dessa aplicação, quando não queremos mudar a senha pois não a sabemos podemos retirá-la ao salvar com o método extract

```

if params[:admin][:password].blank? && params[:admin][:password_confirmation].blank?
  params[:admin].extract!(:password, :password_confirmation)
end

```

e para aparecer bolinha no forms é

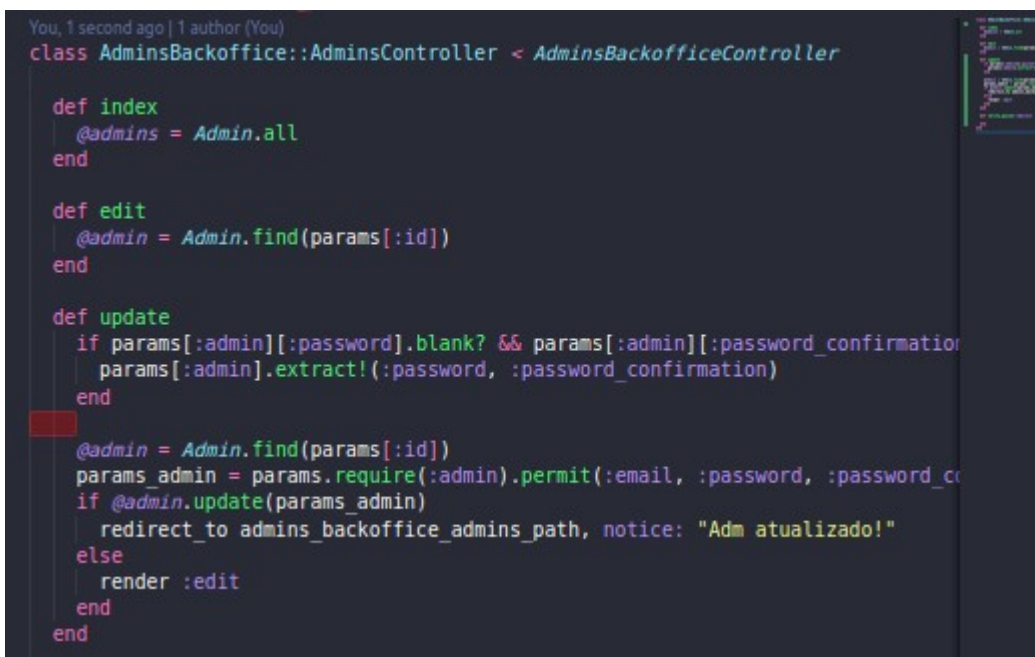
```

<%= form.password_field :password, class:"form-control", placeholder:"Senha do adm"
%>

```

Refatorando controller – Aula 155

antes:



```

You, 1 second ago | 1 author (You)
class AdminsBackofficeController < AdminsBackofficeController

  def index
    @admins = Admin.all
  end

  def edit
    @admin = Admin.find(params[:id])
  end

  def update
    if params[:admin][:password].blank? && params[:admin][:password_confirmation].blank?
      params[:admin].extract!(:password, :password_confirmation)
    end

    @admin = Admin.find(params[:id])
    params_admin = params.require(:admin).permit(:email, :password, :password_confirmation)
    if @admin.update(params_admin)
      redirect_to admins_backoffice_admins_path, notice: "Adm atualizado!"
    else
      render :edit
    end
  end
end

```

o update é só para fazer update, não é para fazer verificação, e outras coisas, então podemos criar um método para verificar e ao invés de chamar esse método verify_password no update podemos criar uma before_action para criar a verificação antes de tudo

Como ficou:

```
You, now | 1 author (You)
class AdminsBackoffice::AdminController < AdminsBackofficeController
  before_action :verify_password, only: [:update]
  before_action :set_admin, only: [:edit, :update]

  def index
    @admins = Admin.all
  end

  def edit
  end

  def update
    if @admin.update(params_admin)
      redirect_to admins_backoffice_admins_path, notice: "Adm atualizado!"
    else
      render :edit
    end
  end

  private

  def params_admin
    params.require(:admin).permit(:email, :password, :password_confirmation)
  end

  def set_admin
    @admin = Admin.find(params[:id])
  end

  def verify_password
    if params[:admin][:password].blank? && params[:admin][:password_confirmation].blank?
      params[:admin].extract!(:password, :password_confirmation)
    end
  end
end
```

cadastrando novos adm – Aula 156

cria o botao

cria arq new.html.erb

def new no controller com @nome = Nome.new ex: @admin = Admin.new

route new e create

<%= action_name %> mostra qual ação new, index

Mostrando notificações – Aula 157

usamos a biblioteca js para criar as notificações

yarn add [bootstrap-growl-ifightcrime@1.1.0](#)

```

<% if notice %>
  <script>
    $.bootstrapGrowl("<%= notice %>", {
      type: 'success', // (null, 'info', 'danger', 'success')
      align: 'right', // ('left', 'right', or 'center')
      allow_dismiss: true, // If true then will display a cross to close the popup.
      stackup_spacing: 10 // spacing between consecutively stacked growls.
    });
  </script>
<% end %>

```

Apagando adm – aula 158

routes

criar methodo no controler e não esquece de colocar no before action o :destroy

e o link to para excluir, no methodo destroy nunca pode esquecer de coloca o method: :delete

```

<%= link_to admins_backoffice_admin_path(admin), method: :delete, class:"btn btn-danger btn-circle" do %>
  <i class="fa fa-trash-o"></i>
<% end %>

```

Pq tem q passar o (admin)? Porque você ve na route do info/routes q é necessario passar o id

DELETE /admins_backoffice/admins/:id(.:format)

admins_backoffice/admins#destroy

Refatorando view com partial – Aula 159

_arqNome.html.erb

dps colocar no edit e no new o render partial sem _

```
ex <%= render partial: 'admins_backoffice/admins/shared/form' %>
```

e para conseguir na mesma pagina colocar ou o novo ou o editando, basta add o locals na pag do edit e new

```

<%= render partial: 'admins_backoffice/admins/shared/form',
  locals: {action_message:"Editando"} %>

```

dos colocar no _fomr o <%= action_message %>

```

<h1 class="page-header"> <%= action_message %> </h1>
<%= "#{action_message} administrador" %>

```

Fazendo paginação com kaminari – aula 160

gem 'kaminari'

podemos paginar pelo controller, ent, na index colocaremos .page params[:page]

```
def index
  @admins = Admin.all.page(params[:page])
end
```

dps colocaremos na view q sera paginada a tag

<%= paginate @admins %>

para definir quantos elementos por paginas podemos ou fazer pelo controller ou pelo model

no controller só colocar na frente.per(n)

no model

paginates_per 5

para mudar o o estilo da paginação, podemos digitar no terminal rails g kaminari:views q listara todos os themes, dps só escolher o theme

Criando migration e model para as questões – Aula 162

description:text, usamos o text porque string só suporta ate 250 caracteres

rails g model answer description:text correct:boolean question:references

podemos no model colocar

```
t.boolean :correct, default: false
t.string :description, null: false
```

Criando task para assuntos

vamos ajustar na task dev rake para criar os subjectos

```
DEFAULT_FILES_PATH = File.join(Rails.root, 'lib', 'tmp')
```

rails.root faz encontrar em qual caminho esta o projeto, e o file.join nos faz não precisar colocar lib/tmp

```
desc "Adiciona assuntos padrão"
task add_subjects: :environment do
  file_name = 'subjects.txt'
  file_path = File.join(DEFAULT_FILES_PATH, file_name)

  File.open(file_path, 'r').each do |line|
    Subject.create!(description: line.strip)
  end
end
```


explicação:

ele vai abrir o arq, o r do `(file_path, 'r')` serve para somente fazer a leitura de cada linha, read.

O line strip serve para tirar qualquer `\n` ou espaço em brancos

Criando task para as questoes – 167

```
desc "Cadastra varias questoes"
task add_question: :environment do
  Subject.all.each do |subject|
    rand(2..4).times do |i|
      Question.create!(
        description: "#{Faker::Lorem.paragraph} #{Faker::Lorem.question}",
        subject: subject
      )
    end
  end
end
```

dps de criar todas as questions conseguimos consultá-las no irb

```
07 15:14:00
irb(main):002:0> q = Question.last
Question Load (0.3ms) SELECT "questions".* FROM "questions" ORDER BY "questions"."id" DESC LIMIT ? [{"LIMIT", 1}]
=> #<Question id: 730, description: "Voluptatum rerum iste. Maiores fugiat recusandae. ...", subject_id: 252, created_at: "2022-11-07 15:14:00", updated_at: "2022-11-07 15:14:00">
irb(main):003:0> q.description
=> "Voluptatum rerum iste. Maiores fugiat recusandae. Porro dolores consequuntur. Id quidem ipsa reprehenderit?"
irb(main):004:0> q.subject.description
Subject Load (0.4ms) SELECT "subjects".* FROM "subjects" WHERE "subjects"."id" = ? LIMIT ? [{"id", 252}, {"LIMIT", 1}]
=> "Veterinária"
irb(main):005:0> 
```

mas como fazer pelo subject ver quais são as question? Para isso tem q configurar no model o `has_many :questions`

a mesma coisa para a view também

```
<td> <%= question.subject.description.truncate(30) %> </td>
```

```

irb(main):006:0> s = Subject.last
d Subject Load (0.3ms) SELECT "subjects".* FROM "subjects" ORDER BY "subjects"."i
d" DESC LIMIT ? [["LIMIT", 1]]
=> #<Subject id: 252, description: "Veterinária", created_at: "2022-11-07 15:10:28"
, updated_at: "2022-11-07 15:10:28">
irb(main):007:0> s.questions
4 Question Load (0.7ms) SELECT "questions".* FROM "questions" WHERE "questions"."
subject_id" = ? LIMIT ? [["subject_id", 252], ["LIMIT", 11]]
=> #<ActiveRecord::Associations::CollectionProxy [#<Question id: 728, description:
"Suscipit aut adipisci. Error sapiente dicta. Incid...", subject_id: 252, created_a
t: "2022-11-07 15:13:59", updated_at: "2022-11-07 15:13:59">, #<Question id: 729, d
escription: "Dolorem id voluptatibus. Ducimus est ratione. Erro...", subject_id: 25
2, created_at: "2022-11-07 15:14:00", updated_at: "2022-11-07 15:14:00">, #<Questio
n id: 730, description: "Voluptatum rerum iste. Maiores fugiat recusandae. ...", su
bject_id: 252, created_at: "2022-11-07 15:14:00", updated_at: "2022-11-07 15:14:00"
>]>
irb(main):008:0> s.questions[0]
Question Load (0.3ms) SELECT "questions".* FROM "questions" WHERE "questions"."s
ubject_id" = ? [["subject_id", 252]]
=> #<Question id: 728, description: "Suscipit aut adipisci. Error sapiente dicta. I
ncid...", subject_id: 252, created_at: "2022-11-07 15:13:59", updated_at: "2022-11-
07 15:13:59">
irb(main):009:0> s.questions[0].description
=> "Suscipit aut adipisci. Error sapiente dicta. Incidunt et est. Facilis magni rep
ellendus voluptatem?"
irb(main):010:0> 

```

Problema n+1 do SQL

Isso acontece quando tem uma tabela relacionada e fazemos uma query para buscar elementos de chave estrangeira, e para resolver isso precisamos do includes, para incluir o outro banco na pesquisa

```
@questions = Question.includes(:subject).order(:description).page(params[:page])
```

ao fazer o includes não é mais necessário o .all, porque ele já entende q são todos os elementos

doc: <https://apidock.com/rails/ActiveRecord/QueryMethods/includes>

Incrementando i18n para models – aula 173

em locais traduzir os models q criamos de sub e question

dps usar o `<th> <%= @subjects.model.human_attribute_name(:description) %> </th>` para traduzir

porém ao invés de sempre escrever model.human... podemos criar um helper, ficando assim:

```

app > helpers > admin_backoffice_helper.rb
You, 18 seconds ago | 1 author (You)
1 module AdminBackofficeHelper
2   def translate_attributes(object, method)
3     object.model.human_attribute_name(method)
4   end
5 end You, last week * adicionando modulo5
6

```

e na view:

```
<th> <%= translate_attributes(@questions, :description) %> </th>
```

i18n com parametros – Aula 174

mesma coisa aula 81, pois conseguimos pelo i18n do model chamar na view, mas podemos passar um paramentros para isso.

```
config > locales > messages.pt-BR.yml
1  'pt-BR':
2    messages:
3      listing: Listando %{model}
4      confirm_with: Você deseja realmente excluir %{item}
```

```
<h1> <%= t('messages.listing', model: @questions.model_name.human(count: 2)) %> </h1>
```

```
data: {confirm: t('messages.confirm_with', item: admin.email.truncate(15)) }
```

no new e edit

```
<%= render partial: 'admins_backoffice/questions/shared/form',
locals: {action_message: t('messages.editing', model: @question.model_name.human) }
%>
```

e no forms

```
<%= @subject.model_name.human %>
```

3 formas de criar registros no active record – Aula 176

a primeira forma seria pelo Model.create!(atribute: “sdad”, ass: “123”)

```
# Question create! description: "Asasdsddsd?", subject: Subject.all.sample
irb(main):002:0> Question.create(description: "Asasdsddsd?", subject: Subject.all.sample)
Subject Load (0.7ms)  SELECT "subjects".* FROM "subjects"
(0.1ms)  begin transaction
Question Create (0.4ms)  INSERT INTO "questions" ("description", "subject_id", "created_at",
"updated_at") VALUES (?, ?, ?, ?) [["description", "Asasdsddsd?"], ["subject_id", 32], ["cr
eated_at", "2022-11-08 14:11:55.755925"], ["updated_at", "2022-11-08 14:11:55.755925"]]
(241.2ms)  commit transaction
=> #<Question id: 732, description: "Asasdsddsd?", subject_id: 32, created_at: "2022-11-08 14
:11:55", updated_at: "2022-11-08 14:11:55">
```

A segunda é por instanciando um new object

q = Question.new

q.name = "maria"

q.save!

```
.11:55 , updated_at: 2022-11-08 14:11:55 >
irb(main):003:0> q = Question.new
=> #<Question id: nil, description: nil, subject_id: nil, created_at: nil, updated_at: nil>
irb(main):004:0> q.description = "Asas2"
=> "Asas2"
irb(main):005:0> q.subject = Subject.all.sample
  Subject Load (1.6ms)  SELECT "subjects".* FROM "subjects"
=> #<Subject id: 236, description: "Segurança e Saúde no Trabalho", created_at: "2022-11-07 15:10:23", updated_at: "2022-11-07 15:10:23">
irb(main):006:0> q
=> #<Question id: nil, description: "Asas2", subject_id: 236, created_at: nil, updated_at: nil>
irb(main):007:0> q.save!
  (0.1ms)  begin transaction
  Question Create (1.0ms)  INSERT INTO "questions" ("description", "subject_id", "created_at", "updated_at") VALUES (?, ?, ?, ?) [["description", "Asas2"], ["subject_id", 236], ["created_at", "2022-11-08 14:13:29.149715"], ["updated_at", "2022-11-08 14:13:29.149715"]]
  (510.7ms)  commit transaction
=> true
```

Já a terceira é pelo params

primeiro pegamos o params, sua chave e os atributos

params = {question: {name: "maria ", subject_id: "123"}}

poem para salvar no banco é preciso só dos atributos, logo ao executar Question.create!(params) não funcionara porque pega a chave junto, ent podemos pedir a chave através de params[:question], logo Question.create!(params[:question]) deve funcionar.

```
irb(main):008:0> params = { question: {description: "Asdasparams", subject_id:123}}
=> {:question=>{:description=>"Asdasparams", :subject_id=>123}}
irb(main):009:0> Question.create!(params)
Traceback (most recent call last):
  1: from (irb):9
ActiveModel::UnknownAttributeError (unknown attribute 'question' for Question.)
irb(main):010:0> params[:question]
=> {:description=>"Asdasparams", :subject_id=>123}
irb(main):011:0> Question.create!(params[:question])
  (0.1ms)  begin transaction
  Subject Load (0.4ms)  SELECT "subjects".* FROM "subjects" WHERE "subjects"."id" = ? LIMIT ? [["id", 123], ["LIMIT", 1]]
  Question Create (1.1ms)  INSERT INTO "questions" ("description", "subject_id", "created_at", "updated_at") VALUES (?, ?, ?, ?) [["description", "Asdasparams"], ["subject_id", 123], ["created_at", "2022-11-08 14:26:17.723114"], ["updated_at", "2022-11-08 14:26:17.723114"]]
  (287.4ms)  commit transaction
=> #<Question id: 734, description: "Asdasparams", subject_id: 123, created_at: "2022-11-08 14:26:17", updated_at: "2022-11-08 14:26:17">
irb(main):012:0>
```

Já a quarta é pelo params passando pelo new e dps dando um save!, isso é feito pelo controller

```

11-08 14:20:17, updated_at: 2022-11-08 14:20:17
irb(main):012:0> params = { question: {description: "Asdasparams", subject_id:123}}
=> {:question=>{:description=>"Asdasparams", :subject_id=>123}}
irb(main):013:0> q = Question.new(params[:question])
=> #<Question id: nil, description: "Asdasparams", subject_id: 123, created_at: nil, u
pdated_at: nil>
irb(main):014:0> q.save!
(0.1ms) begin transaction
Subject Load (0.3ms) SELECT "subjects".* FROM "subjects" WHERE "subjects"."id" = ?
LIMIT ? [["id", 123], ["LIMIT", 1]]
Question Create (1.0ms) INSERT INTO "questions" ("description", "subject_id", "crea
ted_at", "updated_at") VALUES (?, ?, ?, ?) [["description", "Asdasparams"], ["subject
_id", 123], ["created_at", "2022-11-08 14:31:16.394423"], ["updated_at", "2022-11-08 1
4:31:16.394423"]]
(329.5ms) commit transaction
=> true
irb(main):015:0> 

```

Entendendo nested attributes – Aula 177

são atributos de outras tabelas q serão informados e salvados ao mesmo tempo

ex: tenho a table question e answer, na hora de cadastrar a question ao mesmo tempo também vou cadastrar a answers q esta em outra table

ent em um relacionamento 1 para n é só add a linha

accepts_nested_attributes_for :answers

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

e para criar um att, também tem o ex na doc

```

params = { question: {
  description: 'lorem ipsum?', subject_id: 1,
  answers_attributes: [
    { description: 'resposta 1', correct: false },
    { description: 'resposta 2', correct: true },
    { description: 'resposta 3', correct: false } ]
}}

```

```
question = Question.create!(params[:question])
```

```
question.answers
```

adicionando e atualizando um array de hashes – aula 178

Para acessar o array exemplo da aula passada, se observa q é um array de hashes, logo, podemos acessar com

```
params[:question][:answers_attributes]
```

```
params[:question][:answers_attributes].push({ description: 'resposta 4', correct: false})
```

```
params[:question][:answers_attributes][1] = { :description=>"resposta X", :correct=>true }
```

Criando uma task para respostas – Aula 179

antes de refatorar

```
desc "Cadastra varias questoes"
task add_question: :environment do
  Subject.all.each do |subject|
    rand(2..4).times do |i|
      params = {question: {
        description: "#{Faker::Lorem.paragraph} #{Faker::Lorem.question}",
        subject: subject,
        answers_attributes: []
      }}

      rand(2..5).times do |j|
        params[:question][:answers_attributes].push(
          { description: Faker::Lorem.sentence, correct: false }
        )
      end

      index = rand(params[:question][:answers_attributes].size)
      params[:question][:answers_attributes][index] = { description: Faker::Lorem.sentence, correct: true }

      Question.create!(params[:question])
    end
  end
end
```

Refatorando a task de respostas – aula 180

tá no repositório kkk

Instalando a gem Cocoon – Aula 181

gem 'cocoon'

```
==require cocoon no admins_backoffice.js
```

completa no model do nested_attributes com , reject_if: :all_blank, allow_destroy: true

o primeiro serve para rejeitar campos em branco, e o segundo permite apagar as opções

tem q colocar o inverse_of: :model no belongs_to para não dar bo na hora de cadastrar

```
class Question < ApplicationRecord
  belongs_to :subject, inverse_of: :questions
  has_many :answers

  accepts_nested_attributes_for :answers, reject_if: :all_blank, allow_destroy: true
end
```


e no controller, dps do .permit, tem q add o campo virtual do nested_attributes

```
tasks_attributes: [:id, :description, :done, :_destroy]
```

e o destroy aqui serve para excluir o campo na hora de preencher o forms

```
def params_question
  params.require(:question).permit(:description, :subject_id,
    answers_attributes: [:id, :description, :correct, :_destroy])
end
```

View gem Cocoon – Aula 182

na view do forms vamos criar uma div com id do model e dps criar um campo com fields_for pois ele indica que os campos serão aninhados, e dentro vamos renderizar uma partial, no qual estara um formulario

e no final criar o botão q sempre adicionara mais perguntas

```
<div class="answers">
  <%= form.fields_for :answers do |answer| %>
    <%= render partial: "answer_fields", locals: { f: answer } %>
  <% end %>

  <%= link_to_add_association 'Adicionar pergunta', form, :answers %>
</div>
```

dps temos q criar uma partial com o nome do model no singular na raiz da pasta com o nome _answer_fields.html.erb

e dentro desse arq tem uma class com a classe nested-fields e colocar os campos necessarios com o botao de remover

```
pp > views > admins_backoffice > questions > _answer_fields.html.erb
1  <div class="nested-fields">
2    <%= f.label :description %>
3    <%= f.text_field :description %>
4    <%= f.check_box :correct %> Correta?
5
6    <%= link_to_remove_association('Remover', f) %>
7  </div>
```

Conhecendo o devise_controller e o resource_class – Aula 183

são metodos, o devise_controller verifica se esta acessando um controller do devise

e o resource_class qual é a classe desse device q esta sendo acessada


```

app > controllers > application_controller.rb
You, 33 seconds ago | 1 author (You)
1 class ApplicationController < ActionController::Base
2   layout :layout_by_resource
3
4   private
5
6   def layout_by_resource
7     puts "====> #{devise_controller?}"
8     puts "====> #{resource_class}"
9     "application"
10  end
11 end
12

```

```

Started GET "/admins/sign_in" for ::1 at 2022-11-09 11:48:19 -0300
Processing by Devise::SessionsController#new as HTML
====> true
====> Admin
Rendering admins/sessions/new.html.erb within layouts/application
Rendered admins/shared/_links.html.erb (1.8ms)
Rendered admins/sessions/new.html.erb within layouts/application (1.8ms)
Completed 200 OK in 41ms (Views: 39.1ms | ActiveRecord: 0.0ms)

```

ent vemos q ele acessa o admin, podemos usar isso para mudar o layout, assim criando um novo arq em layouts, nesse caso devise_admin e fazer no application um if else

Configurando template para admins devise – Aula 184

escolhemos como sera o layout, nesse caso a page de form do template sd-admin, dps de colocarmos o html temos q configurar os stylesheets link e js, nisso ele tem q herdar no próprio arquivo, para isso em app/assets vamos criar um arq admin_devise.scss e fazer os require e dps precompilar

no layout tem o notice também para mostrar os erros

Conhecendo a gem rails DB – Aula 185

```
gem 'rails_db', '2.3.0'
```

faz com q você veja o banco em maneira de interface e não no rails c, acessando o <http://localhost:3000/rails/db>

Debugando com webconsole – Aula 186

quando se esta em uma maquina virtual ele pode não funcionar por conta do id, ent temos q permitir outro dentro de config/application.rb e colocaremos

```
config.web_console.permissions = 'o ip'
```

uma forma de chama-lo é escrevendo console na index do controller

Uma pitada de metaprogramação nos login – Aula 187

```
app > controllers > application_controller.rb
You, 1 second ago | 1 author (You)
1 class ApplicationController < ActionController::Base
2   layout :layout_by_resource
3
4   protected
5
6   def layout_by_resource
7     if devise_controller?
8       "#{resource_class.to_s.downcase}_devise"
9     else
10      "application"
11    end
12  end
13 end
14
```

estrutura ternaria

```
app > controllers > application_controller.rb
You, 14 seconds ago | 1 author (You)
1 class ApplicationController < ActionController::Base
2   layout :layout_by_resource
3
4   protected
5
6   def layout_by_resource
7     devise_controller? ? "#{resource_class.to_s.downcase}_devise" : "application"
8   end
9 end
10
```

Migrando template do login para user – Aula 188

mesma coisa da aula 184

Obs.: se o pre compilador não funcionar reescreve kkkk

Migrando template do login para site – Aula 189

não pode esquecer q os arq criados em lib temos q pre compilar também

Listando perguntas e respostas na index – Aula 192

usaremos um panels para exibir as perguntas, antes temos q criar a variavel @questions no controller do site/welcome, porque era ele q estavam acessando no log

```
p > views > site > welcome > index.html.erb
You, 36 seconds ago | 1 author (You)
1 <%= @questions.each do |question| %>
2   <div class="panel panel-default">
3     <div class="panel-heading">
4       <h3 class="panel-title"> <%= question.description %> </h3>
5     </div>
6     <div class="panel-body">
7       <ul>
8         <%= question.answers.each do |answer| %>
9           <li> <%= answer.description %> </li>
10        <%= end %>
11      </ul>
12    </div>
13  </div>
14 <%= end %>
You, last week • add arq do modulo 05 ...
```

Adicionando a barra de pesquisa – Aula 193

criaremos um controller rails g controller site::search para a pesquisa,

dps na route de site mesmo criar uma rota q dessa vez será fora do padrao rest

```
get 'search', to: 'search#questions'
```

como o controller criado não tem um model especifico ent temos q mandar uma url do controller pelo forms, nesse caso site search

como é um form de pesquisa também é interessante deixar o methodo como GET e não POST, pois é legal mostrar na url a pesquisa para compartilhar

ficando assim:

```
<%= form_with(url: site_search_path, local:true, method: :get, class:"navbar-form navbar-left") do |form| %>
```

dps temos q escolher um nome para receber os parametros q serao mandados pela url, aqui foi :term

```
<%= form.text_field :term, class:"form-control", placeholder:"Pesquise a pergunta" %>
```

mas ao pesquisar dara um erro, pois ele não encontrara a view para renderizar a pesquisa, nisso, temos q criar a view igual a action, nesse caso questions.html.erb, e nessa view teremos q ter o forms da aula passada para listar as questions q forem pesquisadas

Fazendo a pesquisa funcionar – Aula 194

Conseguimos fazer a pesquisa pelo params[:term]

porem quando tentamos

```
@questions = Question.includes(:answers).where(description: params[:term])
```

não dara certo, pois o .where só faz pesquisa q são iguais e não aproximadas, logo para q funciona precisamos usar o like

```
@questions = Question.includes(:answers).where("description LIKE ?", params[:term])
```

Porem ainda não funciona porque o like precisa de um coringa, sendo o %

```
@questions = Question.includes(:answers).where("description LIKE ?", "%#{params[:term]}%")
```

Mas temos q atentar a qual banco de dados estamos usando, pois ao fazer a pesquisa alguns não se importam com a letra maiuscula ou minuscula e outros sim, por isso é melhor já deixar formatado.

```
You, 1 second ago | 1 author (You)
class Site::SearchController < SiteController
  def questions
    # @questions = Question.all
    @questions = Question.includes(:answers)
    .where("lower(description) LIKE ?", "%#{params[:term].downcase}%")
  end
end
```

E no final se quiser fazer uma paginação só colocr no model, controller e na view

Devo usar o like sempre? - Aula 195

Não sempre, por ele passar em todo registros em grande escalas isso fica inviável

para contornar isso usamos pesquisas do tipo full-text, utilizando servidores com Elastic Search

Usando metodo de classe no model – Aula 196

Como a pesquisa tá no controller, se precisássemos utilizar em outro lugar teríamos q copiar e colar, ent para não fazer isso podemos utilizar um metodo de classe, q é com o self.nomeMethod, assim eu consigo chamar em qualquer lugar dos controllers

model:

```
You, 3 minutes ago | 1 author (You)
class Question < ApplicationRecord
  belongs_to :subject, inverse_of: :questions
  has_many :answers

  accepts_nested_attributes_for :answers, reject_if: :all_blank, allow_destroy: true
  paginates_per 5

  def self.search(page, term)
    Question.includes(:answers)
               .where("lower(description) LIKE ?", "%#{term.downcase}%")
               .page(page)
  end

  def self.last_questions(page)
    Question.includes(:answers)
               .order('created_at desc')
               .page(page)
  end
end
```

Controller:

```
> controllers > site > search_controller.rb
You, 12 minutes ago | 1 author (You)
1 class Site::SearchController < SiteController
2   def questions
3
4     @questions = Question.search(params[:page], params[:term])
5
6   end
7 end
8
```

Usando Scopes – Aula 197

deve ser utilizado quando estamos fazendo metodos de classe q são somente para pesquisa, e o escopo permite encadear vários escopos em uma query

```
1 > models > question.rb
You, now | 1 author (You)
1 class Question < ApplicationRecord
2   belongs_to :subject, inverse_of: :questions
3   has_many :answers
4
5   accepts_nested_attributes_for :answers, reject_if: :all_blank, allow_destroy: true
6
7   paginates_per 5
8
9
10  scope :_search_, -> (page, term){
11    includes(:answers)
12    .where("lower(description) LIKE ?", "%#{term.downcase}%")
13    .page(page)
14  }
15
16  scope :last_questions, ->(page){
17    includes(:answers)
18    .order('created_at desc')
19    .page(page)
20  }
21
22 end
23
24
```

Obs.: o search esta com _ pois deve ter conflitado com alguma coisa no rails q também tem o nome search

Mensagem do termo pesquisado – Aula 198

```
<!-- Main component for a primary marketing message or call to action -->
<div class="jumbotron">
  <% unless params[:term] %>
    <h3>Ultimas perguntas cadastradas...</h3>
  <% else %>
    <h3> <%= "Resultado para a pesquisa \"#{params[:term]}\" \"> ...</h3>
  <% end %>
</div>
```

Ativando as respostas das perguntas – Aula 199

vamos criar um controller para receber as respostas q estraram em um radio button

rails g controller site::answer

Mesmo esquema, herdar do layout correto, verificar se tem route, q nesse caso criamos

```
post 'answer', to: 'answer#question'
```

e no controller criaremos a action question

agr arrumaremos a view q esta em site/shared, q é onde todas as perguntas estão sendo rederizadas, nisso faremos um form porque cada pergunta tera como verificar se esta correta ou não

antes:

```
app > views > site > shared > _questions.html.erb
You, 3 days ago | 1 author (You)
1 <%= @questions.each do |question| %>
2   <div class="panel panel-default">
3     <div class="panel-heading">
4       <h3 class="panel-title"> <%= question.description %> </h3>
5     </div>
6     <div class="panel-body">
7       <ul>
8         <%= question.answers.each do |answer| %>
9           <li> <%= answer.description %> </li>
10        <%= end %>
11      </ul>
12    </div>
13  </div>
14 <%= end %>
15
16 <div class="text-center">
17   <%= paginate @questions %>
18 </div>
```

Para o button usaremos o radio_button_tag pois nele conseguimos usar de forma independente sem um model atrelado

colocamos um button submit para enviar a repostas e criamos a view para receber essa action, nesse caso só tem um h1 para ver se funcionou

Como ficou:


```

<% @questions.each do |question| %>
  <%= form_with url: site_answer_path, local:true do |form| %>
    <div class="panel panel-default">
      <div class="panel-heading">
        <h3 class="panel-title"> <%= question.description %> </h3>
      </div>
      <div class="panel-body">
        <ul>
          <% question.answers.each do |answer| %>
            <li style="list-style:none;">
              <div class="radio">
                <label>
                  <%= radio_button_tag 'answer', answer.id %>
                  <%= answer.description %>
                </label>
              </div>
            </li>
            <% end %>
          <li style="list-style:none;">
            <%= form.submit "Responder", class:"btn btn-default"%>
          </li>
        </ul>
      </div>
    </div>
  <% end %>
<% end %>

<div class="text-center">
  <%= paginate @questions %>
</div>

```

Verificando se a resposta esta correta – Aula 200

É notado q quando pesquisamos uma resposta nela temos o id, e se esta correta ou não, e com o `params[:answer]` conseguimos o id

```

irb(main):003:0> ap Answer.last
Answer Load (0.4ms) SELECT "answers".* FROM "answers" ORDER
answers"."id" DESC LIMIT ? [["LIMIT", 1]]
#<Answer:0x0000555b334cd9c8> {
  :id => 2733,
  :description => "12",
  :correct => false,
  :question_id => 769,
  :created_at => Wed, 09 Nov 2022 14:08:31 UTC +00:00,
  :updated_at => Wed, 09 Nov 2022 14:08:54 UTC +00:00
}
=> nil

```

Mas quando vamos pesquisar podemos fazer pelo `where` ou `find`, nessa caso o `find` se encontra melhor pois ele já traz o obj e não um `ActiveRecord::Relation`, q significa q o poderia vir vários obj

ent no controller faremos a pesquisa:

```
app > controllers > site > answer_controller.rb
1 class Site::AnswerController < SiteController
2   def question
3     @answer = Answer.find(params[:answer_id])
4   end
5 end
6
```

e na view mostraremos se acertou ou não

```
app > views > site > answer > question.html.erb
1 <% if @answer.correct %>
2   <h1>Parabens acertou</h1>
3 <% else %>
4   <h1>Infelizmente errou</h1>
5 <% end %>
6
```

mas obviamente ficar redirecionando uma pagina bão é interessante

Entendendo o ciclo js no rails – Aula 201

No form_with, quando colocamos o local:true faz com q a chamda seja por Html, e isso faz com q ele redirecione a um controller e obrigatoriamente o controller procura uma nova view, ent sempre ira outra pag,

e quando tiramos o local ele procurar por um arq.js.erb para rederizar e interpretar ficando na mesma pag, caso não ache não fara nada, nesse ex fez o question.js.erb e dps colocamos um alert para rederizar

Verificando as respostas atraves de JS – Aula 202

Se olharmos em assets/js/application.js veremos o rails-ujs ele é responsavel por permitir q a app trabalhe de modo remota

Caso quisessemos mudar a cor do botao se tiver correto ou errada a respostas podemos add um id no elemento e usar o getElementById e selecionar, dizer para remover a class do btn-default e dps add a danger ou success

```
<li style="list-style:none;">
  <%= form.submit "Responder", class:"btn btn-default", id:"submit_#{question.id}" %>
</li>
</ul>
```



```

app > views > site > answer > question.js.erb > ...
1  element = document.getElementById("submit_<%= @answer.question_id %>")
2
3  element.classList.remove('btn-default')
4  element.classList.remove('btn-danger')
5  element.classList.remove('btn-success')
6
7  if(<%= @answer.correct %>){
8    element.classList.add('btn-success')
9    $.bootstrapGrowl("Você acertou ^-^", {
10     type: 'success', // (null, 'info', 'danger', 'success')
11     align: 'right', // ('left', 'right', or 'center')
12     allow_dismiss: true, // If true then will display a cross to close the pop
13     stackup_spacing: 10 // spacing between consecutively stacked growls.
14   });
15 }else{
16   element.classList.add('btn-danger')

```

Reflexão sobre o uso do sistema – Aula 203

Temos q sempre observar e pensar o q pode ocorrer no nosso sistema se tiver muitos acessos, ex nesse: ao responde a pesquisa é uma ida ao servidos e uma pesquisa no db, formas de resolver: mais caro é replicar o servidor, loader balance, gerenciador de carga q distribui as requisicoes outra: colocar um banco de dados de cache e não relacional, redis ou já colocar a respostas na própria pergunta com as tag no html, não é recomendado

Filtrando as questões por assunto – Aula 204

Colocamos uma label para mostrar de qual assunto a questão pertence e a transformamos em um link de filtro de pesquisa, dps no controller search criamos um methodo para filtrar os subject, dps criamos uma route para o novo method subject criado, dps arrumar a pesquisa no model, dps criar a view subject q rederizara a partial _question

a label:

```

<%= link_to question.subject.description, site_search_subject_path(question.subject_id,
question.subject.description), style:"color: white;" %>

```

a Route: `get 'subject/:subject_id/:subject', to: 'search#subject', as:'search_subject'`

controller:

```

def subject
  @questions = Question._search_subject(params[:page], params[:subject_id])
end
end

```

model:

```
scope : search_subject , -> (page, subject_id){
  includes(:answers, :subject)
  .where(subject_id: subject_id)
  .page(page)
}
```

e o helper para escrever no jumbotron:

```
helpers > site_helper.rb
You, 1 second ago | 1 author (You)
module SiteHelper
  def msg_jumbotron
    case params[:action]
    when 'index'
      "Ultimas perguntas cadastradas"
    when 'questions'
      "Resultado para a pesquisa \"#{params[:term]}\""
    when 'subject'
      "Mostrando questões para assunto #{params[:subject]}"
    end
  end
end
```

Mostrando a quantidade de questoes por assunto – aula 205

Para contar usamos o count, ent podemos acessar por question.subject.questions.count

o porque disso?

Question nesse caso seria a questão selecionada

o subject é a outra tabela no qual temos relacionamento, esse subject esta disponivel quando declaramos no model o belongs_to :subject e isso é possível por ter o relacionamento que vemos no schema com o subject_id

já o questions é o has_many q esta na tabela subject, ent atraves do subject eu consigo contar as varias questions existentes

```
<%= link_to "#{question.subject.description} (#{question.subject.questions.count})",
site_search_subject_path(question.subject_id, question.subject.description), style:"color:
white;" %>
```

porem ao fazer isso temos um preço, pois são muitas requisições ao servidor, ent uma forma de contornar isso é o counter_cache

Entendendo e usando o counter_cache – Aula 206

É uma opção que podemos utilizar junto com o belongs to, ent coloca-se o counter_cache: true no belongs e dps temos q criar um migration com o campo onde ficara para contar, nesse caso na tabela subject, pois nela guardara a qtd de assuntos

```

app > models > question.rb
You, 2 seconds ago | 1 author (You)
1 class Question < ApplicationRecord
2   belongs_to :subject, counter_cache: true, inverse_of: :questions
3   has_many :answers
4

```

rails g migration AddQuestionsCountToSubjects questions_count:integer

ai ao inves de ser question.subject.questions.count, sera question.subject.questions_count, pois é o nome do campo, só q ao fazermos isso aparecera vazio, pois ainda não foi atualizado no banco quantas questões tem cada assunto, para isso faremos uma task com um method chamado reset_counters(passa o q vai contar; nesse caso o assunto , e quem atualizar)

```

irb(main):001:0> x = Subject.last
  Subject Load (0.1ms)  SELECT "subjects".* FROM "subjects" ORDER BY "subjects"."id" DESC LIMIT ?  [["LIMIT", 1]]
=> #<Subject id: 252, description: "Veterinária", created_at: "2022-11-09 12:41:56", updated_at: "2022-11-09 12:41:56", questions_count: nil>
irb(main):002:0> Subject.reset_counters(x.id, :questions)
  Subject Load (0.2ms)  SELECT "subjects".* FROM "subjects" WHERE "subjects"."id" = ? LIMIT ?  [["id", 252], ["LIMIT", 1]]
  (0.2ms)  SELECT COUNT(*) FROM "questions" WHERE "questions"."subject_id" = ?  ["subject_id", 252]
  Subject Update All (189.1ms)  UPDATE "subjects" SET "questions_count" = 3 WHERE "subjects"."id" = ?  [["id", 252]]
=> true

```

a task

```

desc "Reseta o contador dos assuntos"
task :reset_subject_counter => :environment do
  show_spinner('Resetando contador dos assuntos...') do
    Subject.find_each do |subject|
      Subject.reset_counters(subject.id, :questions)
    end
  end
end

```

Obs.: o find_each é a mesma coisa de Subject.all.each

doc: https://guides.rubyonrails.org/association_basics.html#options-for-belongs-to-counter-cache

Ajustando os cadastros dos usuarios – Aula 207

Basicamente copiar o mesmo form do log in e dps se atentar ao path q esta se for register ou session.

Removendo o cadastro externo de admins – aula 208

Não faz sentido ter um forms para cadastrar admins fora do sistema, porque só 1 adm pode cadastrar outro

para isso ent, no devise temos uma parte da gem q se chama registerable, q permite os sign up, podemos ent ir em routes e usar o skip: [:registrations]

```
devise_for :admins, skip: [:registrations]
devise_for :users
```

e se formos em info/routes não tera mais a rota sign_up e tem q tomar cuidado se na view não tem a linha, pois tem q tira-la para funcionar

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

Adicionando o nome do usuario no devise – Aula 210

só fazer uma migration, e como colocaremos primeiro nome e segundo nome podemos criar um metodo de classe juntando os dois nomes

```
def full_name
  [self.first_name, self.last_name].join(' ')
end
```

Criando formulario de perfil para usuario – Aula 211

Como sera um forms dentro do user criaremos um controller rails g controller UsersBackoffice::Profile, e já criaremos uma action edit, pois já aparecera ao clicar o forms de edição, dps não pode escrever da route

Atualizando os dados do usuario – Aula 212

Como o forms sera para atualizar podemos criar uma route path e não post, caso queira deixar o input só para leitura é o readonly:true

```
<div class="form-group">
  <%= form.label :email, class:"col-form-label col-md-3 col-sm-3 label-align" %>
  <div class="col-md-6 col-sm-6 ">
    <%= form.text_field :email, class:"form-control", readonly: true %>
  </div>
</div>
```

no controller fazer as action edit e update, com as routes, colocar o sistema de msg q precisa importar a biblioteca no assets

porem quando atualizamos a senha ele faz o logoff automaticamente, para resolver isso, precisamos add no update

na vdd agr é `bypass_sign_in(@user)`

```
def update
  if @user.update(params_user)
    bypass_sign_in(@user)
    redirect_to users_backoffice_profile_path, notice: "Usuário atualizado!"
  else
    render :edit
  end
end
```

Conhecendo validações do rails – Aula 213

```
validates :first_name, presence: true, length: {minimum:3}, on: :update
```

on serve para decidir só onde sera aplicado a validação

Adicionando dados extras – Aula 214

Na maioria das vezes não é interessante add campos na tabela criada pelo devise e sim ter uma a parte, por isso vamos criar outra

```
rails g model UserProfile address:string gender:string birthdate:date user:references
```

colocar o `has_one` no user

e como são duas tabelas q serão atualizada ao mesmo tempo precisaremos utilizar o `nested_attributes`, só que para conseguir colocar os campos no nested precisa ser em outro forms

nisso abrimos a tag `form.fields_for :nomemodel do |f|`

`f.sdasd`

`end`

```
<%= form.fields_for :user_profile do |profile_form| %>
  <div class="form-group">
    <%= profile_form.label :address, class:"col-form-label col-md-3 col-sm-3 la
    <div class="col-md-6 col-sm-6 ">
      <%= profile_form.text_field :address, class:"form-control" %>
    </div>
  </div>
```

porém quando subimos o server ainda não aparece os input, pois uma forma de arrumar isso é criar os campos virtualmente, que aí nós vamos no controller e na action onde está o forms, aí como é um `has_one` faremos

```
@user.build_user_profile
```

Já no `has_many` inverte

o `if` é para garantir q só crie o campo se estiver vazio

```

b > controllers > users_backoffice > profile_controller.rb
1  class UsersBackoffice::ProfileController < UsersBackofficeController
2    before_action :set_user
3    before_action :verify_password, only: [:update]
4
5    def edit
6      @user.build_user_profile if @user.user_profile.blank?
7    end
8

```

dps não podemos esquecer de liberar no permite os paramentos

```

def params_user
  params.require(:user).permit(:first_name, :last_name, :email, :password, :password_confirmation,
    user_profile_attributes: [:id, :address, :gender, :birthdate])
end

```

comando q aprendi

ctrl + d para os terminal

no vim

i insere

esc desabilita a edição

:wq para salvar e sair

filtrar no terminal: history | grep new

para atualizar o sistema: gem update --system