CE WIFI 201

CE WIFI 203

Pankiewicza3

<http://www.clab.type.pl/course/view.php?id=18>

dagmara.zawada

Any..!

rails new myapp

cd myapp

rails s

rails c - konsola

reload – odświeżenie konsoli

quit / CTRL+D – wyjście z konsoli

rails g scaffold

rails g model User name:string email:string description:text

MVC!

modele generować w liczbie pojedyńczej – user, book

nie po polsku bo wyjdzie osobas itd. ☺

samo Person.new .. nie zapisuje do bazy danych, dopiero p.save zapisuje do bazy danych

rails automatycznie przetwarza zapytania na sql, np. User.first, count, last…

skrócona składnia user = User.create(name: …, costam: …) / Hash!/

rails c --sandbox – do testowania poleceń

walidacja – po stronie serwera? czy nie lepiej klient?

%w(small medium large) – sprawdzenie w tablicy o takich elementach

ZADANIE 1

rails g scaffold Ad name description:text price:decimal

ważne w liczbie pojedyńczej nazywać

rake db:migrate

rails s

<http://localhost:3000/ads>

rails c

Ad.create(name: "ogłoszenie ogłoszenie", description:"moje ogłoszenie")

create – zrobiony save automatycznie

a = Ad.new(name: "cośtam cośtam", description:"cośtam")

a.save

quit

rails g migration addAttributesToAd email:string img\_url:string

rake db:migrate

edycja w sublime:

zad1 – app – views – ads: index.html.erb, show.html.erb, \_form.html.erb

dodać odpowiednie field

app – controllers: ads\_controller.rb

to ważne żeby dane dodawały się do bazy danych

def ad\_params

params.require(:ad).permit(:name, :description, :price, :email, :img\_url)

end

Gemfile

dodać gem 'faker'

https://github.com/stympy/faker

bundle

edycja w db: seeds.rb – dodany gem faker wypełnia przykładowymi danymi

Ad.destroy\_all

100.times do

Ad.create(name: Faker::Name.name,

description: Faker::Lorem.word,

price: Faker::Commerce.price,

email: Faker::Internet.email,

img\_url: Faker::Company.logo)

end

puts "Ad created"

i w konsoli

rake db:seed

rails s

wyświetlenie obrazków, w show.html.erb

<p>

<strong>Img url:</strong>

<%= image\_tag @ad.img\_url %>

</p>

app – models: ad.rb

class Ad < ActiveRecord::Base

validates :name, presence: true

validates :description, length: {minimum: 5}

end

rails c

rails s

walidacja po stronie serwera – też ważna bo ktoś mógłby podmienić zawartość strony i zapisać śmieci do bazy (po stronie klienta też można dodac js)

/ Record My Desktop /

ZADANIE 2

rails g scaffold Book title isbn author\_id:integer

rake db:migrate

edycja zad2 – Gemfile

gem 'isbn\_validation'

bundle

edycja zad2 – models : book.rb

class Book < ActiveRecord::Base

validates :isbn, :isbn\_format => true

validates :title, presence: true

end

w konsoli

rails c

Book.count

Book

Book.create(isbn:"123") – format nieprawidłowy

Book.create(isbn:"978-3-16-148410-0", title:"Title")

Gemfile – dodać faker jak w zad1

bundle

edycja db: seeds.db – dodamy sto książek

puts "Starting seeds"

Book.destroy\_all

100.times do

Book.create(title: Faker::Name.title, isbn: Faker::Code.isbn)

end

puts "Done"

rails db:seed

rails c

Book.count

=> 100

rails g scaffold Author name:string surname:string author\_id:integer

rake db:migrate

w book.rb

class Book < ActiveRecord::Base

validates :isbn, :isbn\_format => true

validates :title, presence: true

belongs\_to :author

end

w author.rb

class Author < ActiveRecord::Base

has\_many :books

end

w seedach:

puts "Starting seeds"

Book.destroy\_all

20.times do

Author.create(name: Faker::Name.first\_name, surname: Faker::Name.last\_name)

end

100.times do

#1

Book.create(title: Faker::Name.title, isbn: Faker::Code.isbn, author\_id: Author.all.sample.id)

#2

#3

end

puts "Done"

rake db:seed

app – view:

show.html.erb

<p>

<strong>Author:</strong>

<%= @book.author.name + " "+ @book.author.surname %>

</p>

rails g model Employee name surname city organization\_id:integer

rails g model Organization name income headquarters

rake db:migrate

edycja app- models – employee i organization

class Employee < ActiveRecord::Base

belongs\_to :organization

end

class Organization < ActiveRecord::Base

has\_many :employee

end

class Organization < ActiveRecord::Base

has\_many :employee

end

seed.rb:

puts "Start"

Employee.destroy\_all

Organization.destroy\_all

20.times do

Organization.create(name: Faker::Company.name,

income: Faker::Code.ean, # (100..10000).sample

headquarters: Faker::Address.city)

end

Organization.all.each do |organization|

100.times do

emp = Employee.new(name: Faker::Name.first\_name,

surname: Faker::Name.last\_name,

city: Faker::Address.city)

emp.organization = organization

emp.save

end

end

puts "End"

/ db schema – schemat tabel

rails g model Person name surname date\_of\_birth:date

rails g model Book title description:text isbn author\_id:integer

rake db:migrate

gemfile:

gem ‘faker’

definicja asocjacji

zad2\_2 – app – models: book.rb i person.rb:

class Book < ActiveRecord::Base

belongs\_to :author, class\_name: 'Person'

end

class Person < ActiveRecord::Base

has\_many :books

end

seed:

100.times do

a = Person.create(name: Faker::Name.first\_name)

10.times do

Book.create(title: Faker::Name.titlem author: a)

end

end

rake db:seed

spr: Book.last.authorDevise – gotowa appka z logowaniem

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

Project Euler

<https://projecteuler.net/>

ZADANIE 4

zad4\_3

Gemfile:

gem 'geocoder'

location:

class Location < ActiveRecord::Base

geocoded\_by :full\_street\_address

after\_validation :geocode

end

<http://localhost:3000/locations>

edycja: view-show-locations: show.html.erb

<%= image\_tag "http://maps.google.com/maps/api/staticmap?size=450x300&sensor=false&zoom=16&markers=#{@location.latitude}%2C#{@location.longitude}" %>

<http://localhost:3000/locations/1>

<%= na początku znak = oznacza że element będzie wyświetlony w html, bez = element nie będzie wyświetlany, ale będzie działał

Sublime – ctrl alt K – auto format

dodatkowe paczki, ctrl shift p – package control

color picker

jeśli długo projekt wisi – mogą być problemy z siecią?

model odpowiada za logikę biznesową, tworzy używane przez nas obiekty

scaffold tworzy szkielet aplikacji, też model

zad4\_2

rails new ..

rails g model User name surname email role confirmation:boolean

rake db:migrate

rails c

User.create(name: "Dagmara", surname: "Zawada", email: "mail@mail.com")

User.first

ten znak: < - odpowiada za dziedziczenie

edycja w zad4\_2 – app – models – user

class User < ActiveRecord::Base

validates :name, :email, presence: true

validates :role, inclusion: {in: ["regular", "admin"]}

# inny sposób: validates :role, inculsion: { %w(regular admin)}

def full\_name

"#{name} #{surname}"

end

end

po tej zmianie w konsoli działa nowa metoda:

rails c

reload!

u = User.last

u.full\_name

User.create(name:"aaa", surname:"bbb")

* rollback bo jest walidacja

u.errors.full\_messages

po ewentualnych zmianach w db – schema.rb – zrobić migrację, uaktualni się schemat bd

znowu uzupełnienia w users.rb – nowa metoda admin?:

class User < ActiveRecord::Base

validates :name, :email, presence: true

validates :role, inclusion: {in: ["regular", "admin"]}

# validates :role, inculsion: { %w(regular admin)}

def full\_name

"#{name} #{surname}"

end

def admin?

role == "admin"

end

end

inny sposób:

def admin?

self.ole == "admin"

end

class User < ActiveRecord::Base

validates :name, :email, presence: true

validates :role, inclusion: {in: ["regular", "admin"]}

# validates :role, inculsion: { %w(regular admin)}

before\_create :set\_admin\_confirmable, if: :admin?

def full\_name

"#{name} #{surname}"

end

def admin?

role == "admin"

end

def set\_admin\_confirmable

self.confirmation = true # do sprawdzenia w migracji czy confirmable czy confirmation, to ustawione przy tworzeniu modelu

# jeśli używamy selfa wie że korzystamy z atrybutu a nie ze zmniennej lokalnej

# analogicznie jak this. w innych językach

end

#begin

# def admin?

# self.ole == "admin"

# end

#end

end

konrolery

generowanie kontrolerów w liczbie mnogiej

ZADANIE 5

<http://localhost:3000/products/1>

rails g model Product name description:text

rake db:migrate

do gemfile

gem 'faker'

do terminal

bundle

w seedach:

Product.destroy\_all

100.times do

Product.create(name: Faker::Commerce.product\_name,

description: Faker::Lorem.sentence)

end

p "Products created"

w terminal

rake db:seed

rails c

Product.first

Product.last

Product.count

edycja w config – routes.rb

Rails.application.routes.draw do

get 'products/:id' => 'products#show'

end

w terminal info o ścieżkach, kontrolerach i akcjach:

rake routes

rails g controller products

edycja app – controllers : products\_controller.rb

class ProductsController < ApplicationController

def show

@product = Product.find(params[:id])

end

end

utworzyć nowy plik w widokach: app- views – products: show.html.erb

<h1> <%= @product.name %> </h1>

<h2> <%= @product.description %> </h2>

w widok nie powinno wrzucac się logiki, nie powinno byc tutaj odwolań np. do Product.last.name

ewentualnie żeby zresetowac appkę: rake db:reset

nie powinno odwoływać sie po id obiektów, jeśli nie można wejść na dany produkt po id – może nie ma takiego id w bazie?

zad5\_2

dodać do routes.rb:

Rails.application.routes.draw do

get 'products/add' => 'products#add'

get 'products/choice' => 'products#choice'

get 'products/:id' => 'products#show'

end

dodać plik do app – view – products add.html.erb

standardowo wybiera metodę http post, więc trzeba dopisać method :get

<%= form\_tag(products\_choice\_path, method: :get) do %>

<%= label\_tag(:name, "Name: ") %>

<%= text\_field\_tag(:name) %>

<%= label\_tag(:language, "Language: ") %>

<%= select\_tag "language", "<option>Polish</option><option>English</option>".html\_safe %>

<%= submit\_tag("Prześlij") %>

<% end %>

<http://localhost:3000/products/add>

w products\_controller:

class ProductsController < ApplicationController

def show

@product = Product.find(params[:id])

end

def add

end

def choice

@name = params[:name]

@language = params[:language]

end

end

dodać plik do app – view choice.html.erb

<h1> your name is: <%=@name %> </h1>

<h2> your language: <%=@language%></h2>

zad5\_3

w routes dodać przekierowanie do search

get 'products/search' => 'products#search’

get 'products/search\_results' => 'products#search\_results'

dodać plik do app – view ..

search.html.erb

<%= form\_tag(products\_search\_results\_path, method: :get) do %>

<%= label\_tag(:search, "search for: ")%>

<%= text\_field\_tag(:name)%>

<%= submit\_tag "Prześlij"%>

<%end%>

search\_results.html.erb

<% if @products.present? %>

<% @products.each do |p|%>

<h1> <%= p.name %> </h1>

<h2> <%= p.description %></h2>

<%end%>

<% else %>

<h1> Nie znaleziono elementu</h1>

<%end%>

do products\_controller dodać (puste metody niekoniecznie)

def search

end

def search\_results

@products = Product.where(name: params[:name])

end

w terminalu

rails c

Product.connection

Product.last

Product.where(name:"Practical Wooden Hat")

<http://localhost:3000/products/search>

wyszukujemy elementy po nazwie

dodajemy autentykację w products controller

class ProductsController < ApplicationController

http\_basic\_authenticate\_with name: "dag", password: "secret", only: :search

…

zad5\_5

w terminal

rails g model Person name surname

rake db:mgrate

rails g controller people new add

w routes dodało się automatycznie:

get 'people/new' => 'people#new'

get 'people/add' => 'people#add'

i kilka innych.. ☺

w automatycznie utworzonym people.new.html.erb

<h1>People#new</h1>

<p>Find me in app/views/people/new.html.erb</p>

<%= form\_tag(people\_add\_path, metodh: :get) do%>

<%= label\_tag :name, "name: "%>

<%= text\_field\_tag :name %>

<%= label\_tag :surname, "surname:" %>

<%= text\_field\_tag :surname %>

<%= submit\_tag "Prześlij"%>

<%end%>

do models: person. dodać walidację

class Person < ActiveRecord::Base

validates :name, :surname, presence: true

end

do controllers – people controller

class PeopleController < ApplicationController

def new

end

def add

person = Person.new(name: params[:name], surname: params[:surname])

if person.save

@added\_succesfully = true

else

@added\_succesfully = false

end

end

end

do people add:

<h1>People#add</h1>

<p>Find me in app/views/people/add.html.erb</p>

<% if @added\_succesfully %>

<h1> Dodano </h1>

<% else %>

<h1> Niepoprawne dane </h1>

<%end%>

zad5\_6

rails g model Tweet name status

rake db:migrate

do routes

get 'tweets/show/:id' => 'tweets#show'

w terminal

rails g controller tweets status

do db-seeds:

Tweet.destroy\_all

100.times do

Tweet.create(name: Faker::Commerce.product\_name,

status: Faker::Lorem.sentence)

end

p "Tweets created"

w terminal

rake db:seed

rails c

Tweet.count

do app – controller tweets\_controller:

class TweetsController < ApplicationController

def status

end

def show

@tweet = Tweet.find(params[:id])

render action: :status

end

end

do app – views – tweets: status:

<h1><%= @tweet.name%> </h1>

<p><%= @tweet.status%></p>

<http://localhost:3000/tweets/show/1>

wyświetlenie ostatniego elementu:

dodac do routes

get 'tweets/last'

dodać do controllera:

def last

@tweet = Tweet.last

render action: :status

end

nowa appka zad8 – paginacja, stronicowanie

rails new zad8

cd zad8

rails g scaffold Person name surname

rake db:migrate

do gemfile:

gem 'faker'

gem 'will\_paginate'

w terminal

bundle

db – seed:

Person.destroy\_all

1000.times do

Person.create(name: Faker::Name.first\_name,

surname: Faker::Name.last\_name)

end

w terminal

rake db:seed

rails c

Person.count

Person.first

Person.last

rails s

<http://localhost:3000/people>

dok: <https://github.com/mislav/will_paginate>

do modelu person.rb:

class Person < ActiveRecord::Base

self.per\_page = 20

end

people\_controller (na podst. dok. will\_paginate):

def index

@people = Person.paginate(:page => params[:page])

end

do indexu:

<%= will\_paginate @posts %>

zad8\_2

rails g scaffold Post title text

rake db:migrate

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

auto system do logowania, przechowywania haseł szyfrowanych

do gemfile

gem 'devise'

bundle

rails g devise:install

rails g devise MODEL

rake db:migrate

<http://localhost:3000/posts>

podstawowy layout dla wszystkich stron: app – view –layouts – application.html.erb

nie trzeba powtarzać dla wszystkich podstron nagłówków itp…

yield – widok elementu który się zmienia

rake routes

- devise stworzył ścieżki do logowania

dodać w html do app-view-layouts: application.html.erb

**<% if user\_signed\_in? %>**

**<p>User Signed in as: <%= current\_user.email %> </p>**

**<p><%= link\_to "logoff", destroy\_user\_session\_path, method: :delete %> </p>**

**<p> <% link\_to "Nowy post", new\_post\_path %></p>**

**<% else %>**

**<%= link\_to "Sign up", new\_user\_registration\_path %> |**

**<%= link\_to "Log in", new\_user\_session\_path %>**

**<% end %>**

w routes ściezka główna

root 'posts#index'

zatrzeżenie akcji do edycji np

w controllers : posts\_controllers.rb

before\_action to to samo co before\_filter

before\_action :authenticate\_user!, only: [:edit, :new, :update, :destroy, :create]

<http://localhost:3000/posts/1/edit>

config – initilizer – devise.rb

edycja haseł długości..

w view posts index.html.erb

żeby wyświetlało tylko dla zalogowanych użytkowników

<%= link\_to 'New Post', new\_post\_path if user\_signed\_in? %>

wyświetlanie obrazka do produktu

rails g scaffold Product name description

do gemfile

<https://github.com/thoughtbot/paperclip>

gem 'paperclip'

bundle

rails g paperclip product image

rake db:migrate

do app - models – product.rb

class Product < ActiveRecord::Base

has\_attached\_file :image,

:styles => { :medium => "300x300>", :thumb => "100x100>" }

validates\_attachment\_content\_type :image, :content\_type => /\Aimage\/.\*\Z/

end

w formularz produktu: app - products: \_form.html.erb

<div class="field">

<%= f.label :image %><br>

<%= f.file\_field :image %>

</div>

w app - controllers: product controller:

def product\_params

params.require(:product).permit(:name, :description, :image)

end

w app – views – products :show.html.erb

<p>

<strong>Image:</strong>

<%= image\_tag @product.image.url %>

</p>

mniejszy obrazek:

<%= image\_tag @product.image.url(:medium) %>

ZADANIE 6

prosty formularz

1. dodać do app – config: routes:

get 'welcome/form'

post 'welcome/show'

2. dodać pliki: welcome\_controller.rb do controllers i form.html.erb i show.html.erb do app-view-welcome (założyć folder i utworzyć pliki)

form:

Podaj imię:

<%= form\_tag('/welcome/show') do%>

<%= text\_field\_tag :name%>

<%= submit\_tag "Prześlij"%>

<%end%>

i do show:

<h1><%= label\_tag "name: "%>

<%= @name %> </h1>

do welcome\_controller:

class WelcomeController < ApplicationController

def form

end

def show

@name = params[:name]

end

end

dokończyć zadanie 6\_2

żeby wrzucić liczby

.to\_i

albo

.to\_f

home zad 1

rails g scaffold Advertisment description title

rails g model User username email password

dodany gem faker i paperclip

bundle

rails g paperclip advertisment photo

rake db:migrate

do walidacji:

app – models – advertisment

class Advertisment < ActiveRecord::Base

validates :title, presence: true

validates :description, length: {minimum: 10}

belongs\_to :user

end

app – models – user

class User < ActiveRecord::Base

validates :username, :password, presence: true

validates :email, uniqueness: true

has\_many :advertisments

end

żeby parametr foto dodawał się do bazy:

app –controller – advertisment\_controller:

def advertisment\_params

params.require(:advertisment).permit(:description, :title, :photo\_file\_name)

end

db – seeds:

Advertisment.destroy\_all

10.times do

Advertisment.create(title: Faker::Name.name,

description: Faker::Lorem.sentence,

photo\_file\_name: Faker::Company.logo)

end

puts "Advertisments created"

User.destroy\_all

10.times do

User.create(username: Faker::Internet.user\_name,

email: Faker::Internet.email,

password: Faker::Internet.password(8)

)

end

puts "Users created"

rails c

Advertisment.first

User.first

Advertisment.count

User.count

edycja do wyświetlania

users\_controller

class UsersController < ActionController::Base

def index

@users = User.all

end

def show

@username = params[:username]

@email = params[:email]

end

end

routes

get 'users' => 'users#index'

get 'users/:id' => 'users#show'

nowy plik show.html.erb – view – users

wyświetlanie danego usera po id ? - DOKOŃCZYĆ

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

<p>

<strong>User:</strong>

<%= @username %>

</p>

<p>

<strong>Email:</strong>

<%= @email %>

</p>

zad2

rails g scaffold Product ean name description active:boolean published

rails g model User username email password

rake db:migrate

gem 'faker'

bundle

do seeds

User.destroy\_all

10.times do

User.create(username: Faker::Internet.user\_name,

email: Faker::Internet.email,

password: Faker::Internet.password(8)

)

end

puts "Users created"

10.times do

Product.create(ean: Faker::Code.ean,

name: Faker::Commerce.product\_name,

description: Faker::Lorem.sentence,

active: [true, false].sample,

published: User.all.sample.id

)

end

puts "Products created"

do app-models – product.rb:

class Product < ActiveRecord::Base

belongs\_to :user

end

do app-models – user.rb:

class User < ActiveRecord::Base

has\_many :products

end

do show.html.erb:

<p>

<strong>User:</strong>

<%= @product.user.username %>

</p>

NIE DZIAŁA WYŚWIETLANIE PO USER ID…?

zad3

do gemfile

gem 'devise'

gem 'faker'

bundle

rails g devise:install

rails g devise MODEL

rake db:migrate

dodać w html do app-view-layouts: application.html.erb

**<% if user\_signed\_in? %>**

**<p>User Signed in as: <%= current\_user.email %> </p>**

**<p><%= link\_to "logoff", destroy\_user\_session\_path, method: :delete %> </p>**

**<p> <% link\_to "Nowy post", new\_post\_path %></p>**

**<% else %>**

**<%= link\_to "Sign up", new\_user\_registration\_path %> |**

**<%= link\_to "Log in", new\_user\_session\_path %>**

**<% end %>**

w controllers : posts\_controllers.rb

before\_action to to samo co before\_filter

before\_action :authenticate\_user!, only: [:edit, :new, :update, :destroy, :create]

w view posts index.html.erb

żeby wyświetlało tylko dla zalogowanych użytkowników

<%= link\_to 'New Shop', new\_shops\_path if user\_signed\_in? %>

- DOKOŃCZYĆ NIE DZIAŁA….

zad4

do models – message.rb – wyrażenie regularne, można też użyć gema ..

class Message < ActiveRecord::Base

validates :name, :email, :subject, :body, presence: true

validates\_format\_of :email, :with => /\A([^@\s]+)@((?:[-a-z0-9]+\.)+[a-z]{2,})\z/i

end

Message.create(name: "wiadomosc1", email: "mail", subject: "subject1", body: "body1", phone\_number: "555-123-456")

- rollback

Message.create(name: "wiadomosc1", email: "mail@mail.com", subject: "subject1", body: "body1", phone\_number: "555-123-456")

* commit

zad5

do app – config: routes:

Rails.application.routes.draw do

get 'welcome/form'

post 'welcome/form'

post 'welcome/show' => 'welcome#show' //tego nie ma w poprzedniej app (zad6) a tutaj mi bez tego nie działa

end

dodać pliki: welcome\_controller.rb do controllers i form.html.erb i show.html.erb do app-view-welcome (założyć folder i utworzyć pliki)

form:

Podaj imię:

<%= form\_tag('/welcome/show') do%>

<%= text\_field\_tag :name%>

<%= submit\_tag "Prześlij"%>

<%end%>

i do show:

<h1><%= label\_tag "name: "%>

<%= @name %> </h1>

do welcome\_controller:

class WelcomeController < ApplicationController

def form

end

def show

@name = params[:name]

end

end

…..

- inaczej 5\_2-

rails g model User name surname sex computer\_time age

rake db:migrate

gem 'simple\_form'

rails generate simple\_form:install

zad 6

<http://railscasts.com/episodes/352-securing-an-api>

WEEKEND 2

zad1

rails new zad1

cd zad1

rails g model Patient name surname

rake db:migrate

rails g controller patients show new create

w routes zamiast tego:

Rails.application.routes.draw do

get 'patients/show'

get 'patients/new'

get 'patients/create'

end

zamienić:

Rails.application.routes.draw do

resources :patients, only: [:show, :new, :create]

end

w new.html.erb

<h1>Patients#new</h1>

<p>Find me in app/views/patients/new.html.erb</p>

<%= form\_for(@patient) do |f| %>

<%= f.label :name %>

<%= f.text\_field :name %><br>

<%= f.label :surname %>

<%= f.text\_field :surname %><br>

<%= f.submit %>

<% end %>

w patient\_controller:

class PatientsController < ApplicationController

def show

@patient = Patient.find(params[:id])

end

def new

@patient = Patient.new

end

def create

@patient = Patient.new(params[:patient])

if @patient.save

redirect\_to patient\_path(@patient)

# redirect\_to @patient

else

render 'new'

# wyświetla widok new, nie przeładowuje strony

end

end

private

def patient\_params

params.require(:patient).permit(:name,:surname)

end

end

w show.html.erb:

<h1>Patients#show</h1>

<p>Find me in app/views/patients/show.html.erb</p>

<h1><%= label\_tag "name: "%>

<%= @patient.name %> </h1>

<h1><%= label\_tag "surname: "%>

<%= @patient.surname %> </h1>

zad2

rails new zad2

cd zad2

rails g scaffold Employee name surname salary:decimal commision:decimal

rake db:migrate

Gemfile:

gem ‘faker’

bundle

do seedsów:

Employee.destroy\_all

20.times do

Employee.create(name: Faker::Name.first\_name,

surname: Faker::Name.last\_name,

salary: rand(20000),

commision: rand(1000))

end

p "Employees created"

rake db:seed

rails c

Employee.last

do config – routes:

Rails.application.routes.draw do

resources :employees do

collection do

get 'highest\_salary'

get 'the\_richest'

end

end

end

rails c

wybranie max salary

Employee.where(Employee.maximum(:salary))

do app-controllers – employees controller dodać

def highest\_salary

@employee = Employee.where(salary: Employee.maximum(:salary)).first

render 'show'

end

def the\_richest

@employees = Employee.where('salary>:salary', salary: 10000)

render 'index'

end

<http://localhost:3000/employees/the_richest>

<http://localhost:3000/employees/highest_salary>

dodać do routes

member do

get 'full\_salary'

end

w employees\_controller

class EmployeesController < ApplicationController

before\_action :set\_employee, only: [:show, :edit, :update, :destroy, :full\_salary]

…

w models – employee

class Employee < ActiveRecord::Base

def to\_s

name+" "+surname

end

def full\_salary

12\*(salary+commision)

end

end

<http://localhost:3000/employees/1/full_salary>

dodanie do show.html.erb linku do wyświetlenia full salary tego pracownika

<%= link\_to 'full\_salary', full\_salary\_employee\_path(@employee) %>

zad 3\_4 (skopiowane zad2)

do employee zamienić

# def self.highest\_salary

# Employee.where(salary: Employee.maximum(:salary)).first

# end

scope :highest\_salary, -> { where(salary: Employee.maximum(:salary)).first}

#scope i metoda statyczna (klasowa) to to samo tylko scope ładniejszy zapis

w employees.controler

def highest\_salary

# @employee = Employee.where(salary: Employee.maximum(:salary)).first

@employee = Employee.highest\_salary

render 'show'

end

<http://localhost:3000/employees/highest_salary>

zad6

nested attributes

<http://railscasts.com/episodes/196-nested-model-form-part-1>

rails new zad6

cd zad6

rails g scaffold Survey name

rails g model Question survey:references content:text

rails g model Answer question:references content:text

rake db:migrate

(utworzy trzy tabele bo wcześniej nie było migracji)

do app-models : questions

class Question < ActiveRecord::Base

belongs\_to :survey

has\_many :answers, dependent: :destroy

accepts\_nested\_attributes\_for :answers

end

do app-models survey

class Survey < ActiveRecord::Base

has\_many :questions, dependent: :destroy

accept\_nested\_attributes\_for :questions

end

do survey controller uzupełnić

# GET /surveys/new

def new

@survey = Survey.new

3.times do

@survey.questions.build

end

….

def survey\_params

params.require(:survey).permit(:name, questions\_attributes: [:id,:content])

end

do view surveys \_form.html.erb

<%= f.fields\_for :questions do |builder|%>

<%= builder.label :content, "Question"%> <br>

<%= builder.text\_area :content, rows: 3%> <br>

<%end%>

do view-survey show.html.erb

<% @survey.questions.each do |questions|%>

<h1> <%=questions.content %></h1>

<% end %>

uzupełnić w survey controller

def new

@survey = Survey.new

3.times do

question = @survey.questions.build

2.times do

question.answers.build

end

end

end

….

def survey\_params

params.require(:survey).permit(:name, questions\_attributes: [:id,:content answers\_attributes: [:id,:content]])

end

w \_form:

<%= f.fields\_for :questions do |builder|%>

<%= builder.label :content, "Question"%> <br>

<%= builder.text\_area :content, rows: 3%> <br>

<%= builder.fields\_for :answers do |f|%> <br>

<%= f.label :content, "Answers" %><br>

<%= f.text\_area :content, rows: 3 %><br>

<%end%>

<%end%>

w show:

<% @survey.questions.each do |question|%>

<h1> <%= question.content %></h1>

<% question.answers.each do |a|%>

<h2> <%=a.content %></h2>

<%end%>

<% end %>

<http://localhost:3000/surveys/9>

git zad 1

git config --globaluser.name “dagmara”

git config --globaluser.email “dagmara…”

git config –global color.ui true

rails new testapp

cd testapp

git init

* lub git init testapp

git status

pliki na czerwono nie są śledzone

git add .

na zielono śledzone

git commit –m “initial commit”

git status

nic nowego do dodania

git log

…

commit 3b16ad4269b93e1efd96b5453067698ad4fccdd0

…

utworzony nowy plik.txt

git status

git add .

git status

git commit –m “nowy plik”

zmiany w pliku.txt i README.rdoc

git status

git add nowy\_plik.txt

commit –m – zrobi commit tylko to co jest na zielono bo wcześniej był add na tym pliku

commit –am – doda i zrobi commit tego co jest na czerwono (od razu zrobi add i commit)

dodane repozytoria zdalne

git remote –v

git checkout -b testapp

git checkout testapp

git checkout master

git branch

\* master – aktywny branch

testapp

testbranch

git branch -d testbranch

git branch

stash – dodawanie do stosu

pop – odejmowanie od stosu (po kolei)

edycja jakiegoś pliku (musimy przerwać)

git stash

(cofa status plików do ostatniego commita)

powrót do edycji w tym miejscu:

git stash pop

heroku

wake my dyno

ssh

<https://help.github.com/articles/generating-ssh-keys/>

ssh-keygen -t rsa -C [dagmara.223@gmail.com](mailto:dagmara.223@gmail.com)

ssh\_key

można ręcznie otworzyć plik (moja nazwa) ssh\_key.pub

i na github.com/settings/ albo bitbucket dodać ssh key

git init

----

zad git

<https://bitbucket.org/dagmara223/profile/repositories>

ssh-keygen -t rsa -C [dagmara.223@gmail.com](mailto:dagmara.223@gmail.com)

git push -u origin master

rails new .

git add .

git commit -am "created rails app"

git push origin

git status

zmiana w readme – podgląd jakie zmiany:

git diff

git commit -am "przypadkowe zmiany"

gitk – git kum podgląd zmian

git reset HEAD~1 ­­soft

git push origin master

cofnięcie zmian już po pushu na serwer

git log

git revert 7dd36b47ba2b839fff1fab166ae271985a568a9e

git commit –am“cofniecie“

<https://bitbucket.org/dagmara223/test_repo1/commits>

heroku toolbelt

twitter bootstrap 3

getbootstrap.com

warsaw.craigslist.pl

zurb foundation

intalacja twitter bootrap

gem better\_errors

zad7

rails new zad7

cd zad7

rails g scaffold User username status confirmed:boolean

rake db:migrate

Gemfile

gem 'faker'

gem "better\_errors"

do seedów:

10.times do

User.create(username: Faker::Internet.user\_name,

status: 'active',

confirmed: true)

end

10.times do

User.create(username: Faker::Internet.user\_name,

status: 'inactive',

confirmed: false)

end

p "Users created"

rake db:seed

puts „nazwa zmiennej“ metoda to\_s

p „nazwa zmiennej“ metoda inspect

jeśli wartośc boolean jest ustawiona dla danego atrybutu to można ze znakiem zapytania:

rails c

User.last.confirmed?

w app-models-user

class User < ActiveRecord::Base

scope :active , -> {where status: 'active'}

scope :inactive , -> {where status: 'inactive'}

end

więcej metod w guides

http://guides.rubyonrails.org/active\_record\_querying.html

rails c

User.active.last

User.active.first(3)

User.find(5)

User.find(5).confirmed?

do routes

Rails.application.routes.draw do

resources :users do

collection do

get 'active'

get 'inactive'

end

end

end

rake routes

do users\_controller:

def inactive

@users = User.inactive

render 'index'

end

def active

@users = User.active

render :index

end

‚index‘ i :index – to samo

<http://localhost:3000/users/inactive>

w index.html.erb:

<%= link\_to 'Inactive', inactive\_users\_path %>

<%= link\_to 'Active', active\_users\_path %>

zad8

rails new zad8

cd zad8

rails g scaffold Mem name description:text

rake db:migrate

Gemfile

gem 'paperclip'

bundle

rails generate paperclip Mem image

rake db:migrate

do models mem.rb

class Mem < ActiveRecord::Base

has\_attached\_file :image, :styles => { :medium => "300x300>" }, :default\_url => "/images/:style/missing.png"

validates\_attachment\_content\_type :image, :content\_type => /\Aimage\/.\*\Z/

end

do view-form dodać:

<div class="field">

<%= f.label :image%><br>

<%= f.file\_field :image%>

</div>

do mems\_controller:

def mem\_params

params.require(:mem).permit(:name, :description, :image)

end

do view show.html

<p>

<strong>Image:</strong>

<%= image\_tag @mem.image.url %>

</p>

lub

<p>

<strong>Image:</strong>

<%= image\_tag @mem.image.url(:medium) %>

</p>

jeśli zmienimy wielkość obrazków np

class Mem < ActiveRecord::Base

has\_attached\_file :image, :styles => { :medium => "100x100>" }, :default\_url => "/images/:style/missing.png"

validates\_attachment\_content\_type :image, :content\_type => /\Aimage\/.\*\Z/

end

trzeba odświeżyć

rake paperclip:refresh CLASS=Mem

wyświetlanie osobnego widoku dla medium

<http://localhost:3000/mems/1/medium>

nowy widok view – medium.html.erb

<%= image\_tag @mem.image.url(:medium) %>

do mem\_controller:

def medium

@mem = Mem.find(params[:id])

render 'medium'

end

lub

before\_action :set\_mem, only: [:medium, :show, :edit, :update, :destroy]

do routes

Rails.application.routes.draw do

resources :mems do

member do

get 'medium'

end

end

end

sprawdzić rake routes

w show.html.erb

<%= link\_to 'Medium', medium\_mem\_path %> |

hosting app ror np. <https://c9.io/dagmara223>

projekt inteligenty dom

rails new intelligent\_house

cd intellignt\_house

git init

git add .

git commit –am“initial commit“

git status

Gemfile (powinno być więcej ale nie działa)

gem "twitter-bootstrap-rails"

rails generate bootstrap:install static

rails g bootstrap:layout application

(Y)

git add .

git commit -m "added twitter bootstrap"

git status

rails g controller main index

do routes

Rails.application.routes.draw do

get 'main/index'

root 'main#index'

end

git status

git add .

git status

git commit -m"edited routes"

git status

git checkout -b admin\_panel

git status

rails g scaffold room name

rails g scaffold category name

rails g scaffold RoomCategory room:references category:references

rails g scaffold device name on:boolean type RoomCategory:references

rake db:migrate

rails g bootstrap:themed Rooms

rails g bootstrap:themed Categories

rails g bootstrap:themed RoomCategories

rails g bootstrap:themed Devices

(a)

rails s

<http://localhost:3000/rooms>

git add .

git commit -m "added all required models"

git status

git merge master

(rebase – powinno się używać zamiast merga)

git checkout master

git merge admin\_panel

git status

Gemfile

gem 'devise'

bundle

rails generate devise:install

rails generate devise User

w view layouts application.html.erb dodać

<div class="pull­right login­links"> <% if user\_signed\_in? %>

<%= link\_to current\_user.email, edit\_user\_registration\_path, :title => "Edit Profile"%> <%= link\_to "Log off", destroy\_user\_session\_path, :method => :delete, :class

=> "label label­important" %> <% else %>

<%= link\_to "Log in", new\_user\_session\_path, :method => :get %> |

<%= link\_to "Register", new\_user\_registration\_path, :method => :get %> <% end %>

</div>

i zmiana:

<!-- <a class="navbar-brand" href="#">IntelligentHouse</a> -->

<%= link\_to "IntelligentHouse", root\_path, class: "navbar-brand"%>

git add .

git commit -m "added devise to project"

git status

rails g controller admin index

w routes

scope '/admin' do

resources :devices

resources :room\_categories

resources :categories

resources :rooms

end

<http://localhost:3000/admin/index>

w view layouts application usunąć sidebar well sidebar-nav (zakomentowany) i w to miejsce partial:

<% if params[:controller] == 'main' %>

<%= render 'main/sidebar'%>

<% else %>

<%= render 'admin/sidebar'%>

<% end %>

w views admin wrzucić nowy partial widok \_sidebar.html.erb

<div class="col-lg-3">

<div class="well sidebar-nav">

<h3>Sidebar</h3>

<ul class="nav nav-list">

<li class="nav-header">Sidebar</li>

<li><%= link\_to "Link1", "/path1" %></li>

<li><%= link\_to "Link2", "/path2" %></li>

<li><%= link\_to "Link3", "/path3" %></li>

</ul>

</div>

</div>

<http://localhost:3000/>

<http://localhost:3000/admin/rooms>

git status

git add .

git commit -m'sidebar edited'

git status

do models – room.rb

class Room < ActiveRecord::Base

has\_many :room\_categories

has\_many :categories, through: :room\_categories

end

do model.rb

class Category < ActiveRecord::Base

has\_many :room\_categories

has\_many :rooms, through: :room\_categories

end

room\_catogory:

class RoomCategory < ActiveRecord::Base

belongs\_to :room

belongs\_to :category

has\_many :devices

end

w config initializer devise

config.password\_length = 3..128

do seedów:

Room.destroy\_all

%w( Kitchen Living\ Room Hall Bedroom).each do |e|

Room.create(name: e)

end

puts 'created rooms'

Category.destroy\_all

%w( Heating Audio Alarm Lighting).each do |e|

Category.create(name: e)

end

puts 'created categories'

User.destroy\_all

User.create!(email: 'dagmara.223@gmail.com', password: '123', password\_confirmation: '123')

puts 'created users'

Room.all.each do |room|

room.categories << Category.where(name: 'Lightning').first

end

był bład w nazwie kolumny w tabeli device, utworzyć nową migrację

rails g migration RenameColumn

i w niej:

class RenameColumn < ActiveRecord::Migration

def change

rename\_column :devices, :RoomCategory\_id, :room\_category\_id

end

end

rake db:migrate

rake db:seed

w model device też poprawić na room\_category zamiast RoomCategory

do main controller

class MainController < ApplicationController

before\_action :prepare\_sidebar

def index

end

def room

end

def category

end

private

def prepare\_sidebar

@rooms = Room.all

@categories = Category.all

end

end

do view – model main dodać nowe widoki room.html i category.html

do routes dodać:

get '/room/:id', to: 'main#room', as: 'main\_room'

get '/category/:id', to: 'main#category', as: 'main\_category'

w main controller

def room

@room = Room.find(params[:id])

end

def category

@category = Category.find(params[:id])

end

utworzone nowe widoki main – category i room

...

dodac do seedów

audio\_cat = Category.where(name: 'Audio').first

heating\_cat = Category.where(name: 'Heating').first

hall = Room.where(name: 'Hall').first

bedroom = Room.where(name: 'Bedroom').first

bedroom.categories << audio\_cat

bedroom.categories << heating\_cat

hall.categories << audio\_cat

grzejnik = Device.create(name: 'Grzejnik')

stereo = Device.create(name: 'sprzęt grający')

stereo2 = Device.create(name: 'głośniki')

bedroom.room\_categories.where(category\_id: audio\_cat.id).first.devices << stereo

bedroom.room\_categories.where(category\_id: heating\_cat.id).first.devices << grzejnik

hall.room\_categories.where(category\_id: audio\_cat.id).first.devices << stereo2

utworzyć repozytorium na bitbucket albo github i push

git remote add origin [git@https://github.com/dagmara223/intelligent\_house](mailto:git@https://github.com/dagmara223/intelligent_house)

git push -u origin –all

git push -u origin

git status

git add .

git status

git push

<https://bitbucket.org/dagmara223/intelligenthouse/commits/branch/master>

<https://github.com/manuelvanrijn/bootstrap-switch-rails>

do gemfile

gem "bootstrap-switch-rails"

bundle

do app – assets – javascript - application.js

//= require bootstrap-switch

usunac linijke

//= require turbolinks

do app – assets – stylesheets – application.css

/\* \* for bootstrap3 \*= require bootstrap3-switch \* \* or for bootstrap2 \*= require bootstrap2-switch \*/ \*= require bootstrap-switch

w bootstrap.js.coffee

jQuery ->

$("a[rel~=popover], .has-popover").popover()

$("a[rel~=tooltip], .has-tooltip").tooltip()

$("name^='switch\_']").bootstrapSwitch()

<http://harmdelaat.com/home-automation-with-x10-raspberry-pi-linux-and-ruby-on-rails/>

WEEKEND 3

Zuga 3

BE-SZKOLENIA

biznes2014

Algorytmy – bardziej do budowania silników aplikacji

Codility

<https://codility.com/c/intro/demoFAR2GK-9TE>

ustawienie zmiennych srodowiskowych na serwerze – dostep ma tylko admin

MAILER

ZAD1

rails new zad1

cd zad1

rails g scaffold User name email

rake db: migrate

w config – routes

root 'users#index'

<http://localhost:3000/>

rails g mailer UserMailer

w app – mailer – application\_mailer konfiguracja

class ApplicationMailer < ActionMailer::Base

default from: "no-replay@example.com"

layout 'mailer'

end

w app – maile – user\_mailer

class UserMailer < ApplicationMailer

def welcome\_email(user)

@user = user

@message = "hello message"

mail(to: @user.email, subject: "rails mailer")

end

end

w app-views – user-mailer nowy plik welcome\_email.html.erb

<%= @message =>

w user\_controller uzupelnic

def create

@user = User.new(user\_params)

respond\_to do |format|

if @user.save

**UserMailer.welcome\_email(@user).deliver\_later**

format.html { redirect\_to @user, notice: 'User was successfully created.' }

format.json { render :show, status: :created, location: @user }

else

format.html { render :new }

format.json { render json: @user.errors, status: :unprocessable\_entity }

end

end

end

konfiguracja w config environments development.rb

config.action\_mailer.delivery\_method = :smtp

config.action\_mailer.smtp\_settings = {

address: 'smtp.gmail.com',

port: 587,

domain: 'ga.com',

user\_name: 'dagmara.223', # ENV[‘GMAIL\_USERNAME‘]

password: '...',

authentication: 'plain',

enable\_starttls\_auto: true }

zmienne srodowiskowe

w konsoli wyświetlenie zmiennych środowiskowych

printenv

w konsoli:

export GMAIL\_USERNAME=„blablabla“

w projekcie:

ENV[‘GMAIL\_USERNAME‘]

ZAD2 (ten sam projekt)

rails g scaffold Newsletter content:text is\_sent:boolean receipients:text

rake db:migrate

w app – views – newsletters index.html.erb

<td><%= link\_to 'SEND', sent\_newsletter\_path(newsletter) unless newsletter.is\_sent %></td>

w routes:

resources :newsletters do

member do

get 'sent'

end

end

w app-controllers – newsletters\_controller

before\_action :set\_newsletter, only: [:show, :edit, :update, :destroy, :sent]

def sent

@newsletter.send\_to\_all\_users

redirect\_to newsletter\_path

end

niektóre hasła sa zarezerwowane (np. send) nie może być metodą

do app-mailer - user\_mailer dodać

def newsletter(newsletter, email)

@message = newsletter.content

mail(to: email, subject: "newsletter")

end

dodać plik do user\_mailer: newsletter.html.erb

w app-models newsletter.rb

class Newsletter < ActiveRecord::Base

def send\_to\_all\_users

User.find\_each do |u|

UserMailer.newsletter(self, u.email).deliver\_later

end

self.is\_sent = true

self.save

end

end

opóźnione wysyłanie

gem delayed\_job

<https://github.com/collectiveidea/delayed_job>

do gemfile

gem 'delayed\_job\_active\_record'

rails generate delayed\_job:active\_record

rake db:migrate

teraz można w newsletter\_controller zmienić na delay:

def sent

@newsletter.delay.send\_to\_all\_users

redirect\_to newsletter\_path

end

w terminalu (drugim)

rake jobs:work

ZAD4

rails new zad4

cd zad4

rails g scaffold Project name

rails g scaffold Task name project:references

rake db:migrate

do models project

class Project < ActiveRecord::Base

has\_many :tasks

end

do seedów:

p = Project.create(name: "Project")

500.times do |n|

p.tasks << Task.create(name: "task nr #{n}")

end

rake db:seed

w taks controller zmienić na

def index

@tasks = Task.includes(:project).all

end

Wywyołanie localhost/tasks przed tą zmianą:

Rendered tasks/index.html.erb within layouts/application (376.2ms)

Completed 200 OK in 445ms (Views: 432.5ms | ActiveRecord: 9.8ms)

po:

Started GET "/tasks" for ::1 at 2015-01-10 16:42:31 +0100

Processing by TasksController#index as HTML

**Task Load (8.7ms)** SELECT "tasks".\* FROM "tasks"

**Project Load (0.2ms)** **SELECT "projects".\* FROM "projects" WHERE "projects"."id" IN (1)**

Rendered tasks/index.html.erb within layouts/application (155.9ms)

Completed 200 OK in 240ms (Views: 227.3ms | ActiveRecord: 9.3ms)

poprawka w widoku app-view-tasks index żeby wyświetlić nazwę a nie obiekt

<td><%= task.project.name %></td>

postgress instalacja / jednak lepiej brew install

<http://postgresapp.com/>

counter cache

<http://asciicasts.com/episodes/23-counter-cache-column>

instalacja postgres

brew install postgres

nowa aplikacja

rails new zad6 --database=postgresql

gem install pg

konfiguracja w config database.yml

default: &default

adapter: postgresql

encoding: utf8

database: zad6\_development

# username: postgres

# password: postgres

port: 5432

pool: 5

rake db:create

rake db:setup

rails g migration AddTaskCountToProject

w wygenerowanej pustej migracji

class AddTaskCountToProject < ActiveRecord::Migration

def up

add\_column :projects, :tasks\_count, :integer, default: 0, null: false

Project.find\_each do |p|

p.tasks\_count = p.tasks.size

p.save!

end

end

end

w view project index

<td><%= project.tasks\_count %></td>

rails g scaffold User username active:boolean

a tej nowej migracji

class CreateUsers < ActiveRecord::Migration

def change

create\_table :users do |t|

t.string :username

t.boolean :active, default: false

t.timestamps null: false

end

end

end

callback – np. after create, before update (i podajemy jaką metodę chcemy wykonać) – np. po stworzeniu obiektu zamieniamy nazwę na wielką literę

definiowanie asocjacji – has\_many, belongs\_to ... zamodelowane po stronie bazy danych (klucz obcy)

ZADANIE 7

rails new zad7

rails g scaffold Article title content:text

rake db:migrate

<https://github.com/norman/friendly_id>

Gemfile

gem 'friendly\_id'

bundle

rails generate friendly\_id

rake db:migrate

rails g migration addSlugToArticle slug:string:uniq

rake db:migrate

w app-models – article np.

class Article < ActiveRecord::Base

extend FriendlyId

# friendly\_id :title, use: :slugged

friendly\_id :slug\_candidates, use: :slugged

def duplicates\_count

Article.where(title: self.title).count + 1 # Model is whatever the model is

end

def slug\_candidates

[

:title,

[:title, :duplicates\_count]

]

end

end

w app- controller – article controller

def set\_article

@article = Article.friendly.find(params[:id])

endZADANIE 8

edycja w miejscu, bez przeładowywania strony (js)

rails new zad8

cd zad8

rails g scaffold User name email gender public\_profile

rake db:migrate

<https://github.com/bernat/best_in_place>

gem 'best\_in\_place'

bundle

<http://railscasts.com/episodes/302-in-place-editing>

do app/assets/javascripts/application.js (ważne żeby za jquery)

*//= require jquery.purr*

*//= require best\_in\_place*

mozna wywalać //= require turbolinks

app/assets/javascripts/users.js.coffee

jQuery -> $('.best\_in\_place').best\_in\_place()

users/show.html.erb

<p>

<b>Email:</b>

<%= best\_in\_place @user, :email %>

</p>

<p>

<b>Gender:</b>

<%= best\_in\_place @user, :gender, type: :select, collection: [["Male", "Male"], ["Female", "Female"], ["", "Unspecified"]] %>

</p>

<p>

<b>Public profile:</b>

<%= best\_in\_place @user, :public\_profile, type: :checkbox, collection: %w[No Yes] %>

</p>

app/assets/stylesheets/users.css.scss

.purr {

position: fixed;

top: 30px;

right: 100px;

width: 250px;

padding: 20px;

background-color: #FCC;

border: solid 2px #C66;

&:first-letter { text-transform: uppercase; }

}

TWORZENIE WŁASNEGO API

JSON

automatycznie generuje html ale mozna wziąć json:

<http://localhost:3000/users.json>

[{"id":1,"name":"a","email":"kbh","gender":"Male","public\_profile":"true","url":"http://localhost:3000/users/1.json"}]

gem responders

<https://github.com/plataformatec/responders>

ZADANIE2

rails new zad2

rails g model Customer name surname

rails g model Project customer:references name

rails c

Customer.create(name: "Jozef")

w app-model-customer

class Customer < ActiveRecord::Base

has\_many :projects

end

w konsoli

reload!

Customer.last.projects.create(name: "Praca magisterska")

daje xml

Project.last.to\_xml

ZADANIE3

rails g model User email username name

rake db:migrate

rails c

User.create(email: "aaa@mail", username: "aaa", name: "a")

User.last.to\_json

User.all.to\_json

ZADANIE4

zwracanie JSON

rails g model Person name surname

rake db:migrate

w routes

resources :people, only: [:index]

Gemfile

gem 'faker'

seeds:

1000.times do

Person.create(name: Faker::Name.first\_name, surname: Faker::Name.last\_name)

end

rake db:seed

nowy app-controllers:

people\_controlles.rb

class PeopleController < ApplicationController

def index

@people = Person.all

render json: @people

end

end

<http://localhost:3000/people>

ZADANIE 5

ograniczenie wyników JSON

gem 'will\_paginate'

w people\_controller.rb:

@people = Person.paginate(page: params[:page])

w model- person.rb

class Person < ActiveRecord::Base

self.per\_page = 15

end

<http://localhost:3000/people?page=2>

w routes

namespace :api do

resources :people, only: [:index]

end

nowy folder app-controller – api i tutaj wrzucić zmieniony plik people controller:

class Api::PeopleController < ApplicationController

def index

@people = Person.paginate(page: params[:page])

render json: @people

end

end

<http://localhost:3000/api/people>

JAVASCRIPT

closures

prototype

JS Goodparts

<https://www.codeschool.com/courses/javascript-road-trip-part-3>

<https://developer.mozilla.org/pl/>

<http://guides.rubyonrails.org/working_with_javascript_in_rails.html>

ZADANIE TO-DO-LIST

w view task index

<%= link\_to "New Task", new\_task\_path, id: "new\_link", remote: true %>

w htmlu id css jest z tego id powyżej

nowy plik new.js.erb do view tasks

$('#new\_link').hide().after('<%= j render("form") %>');

ograniczenie formatów odpowiedzi:

respond\_to do |format|

format.html

format.js

end

ajaxowe wysłanie formularza – dodać remote w \_form

<%= form\_for @task, remote: true do |f| %>

<%= f.text\_field :name %>

<%= f.submit %>

<% end %>

w task controller

def create

@task = Task.create!(task\_params)

respond\_to do |format|

format.html { redirect\_to tasks\_url }

format.js￼

end

end

nowy plik create.js.erb w app-view-tasks

$('#new\_task').remove();

$('#super\_link').show();

$('#incomplete\_tasks').append('<%= j render(@task) %>');

CMD + ?/ - szybkie wykomentowanie linii

dynamiczne usuwanie

nowy plik destroy.js.erb

$('#edit\_task\_<%= @task.id %>').remove();

w task\_controller

def destroy

@task = Task.destroy(params[:id])

respond\_to do |format|

format.html {redirect\_to tasks\_url}

format.js

end

end

w \_task.form...

<%= link\_to "(remove)", task, method: :delete, data: {confirm: "Are you sure?"}, remote: true %>

zmiana nazwy w app – assets – javascript

tasks.js.coffee -> tasks.js

$(function() {

$('.edit\_task input[type=submit]').remove();

$('.edit\_task input[type=checkbox]').click(function() {

$(this).parent('form').submit(); });

});

w \_task..

<%= form\_for task, remote: true do |f| %>

<%= f.check\_box :complete %>

<%= f.submit "Update" %>

<%= f.label :complete, task.name %>

<%= link\_to "(remove)", task, method: :delete, data: {confirm: "Are you sure?"}, remote: true %>

<% end %>

w task controller

def update

@task = Task.find(params[:id])

@task.update\_attributes!(task\_params)

respond\_to do |format|

format.html {redirect\_to tasks\_url}

format.js

end

end