Laboratorium nr 4
Model-Widok-Kontroler

Aleksander-Smywiński Pohl

Elektroniczne Przetwarzanie Informacji

Plan prezentacji

Książka gości

Zadanie

Zaimplementować funkcjonalność książki gości, bez użycia rusztowania.

Model Entry

1. ▷ `rails g model entry name:string comment:text`
2. ▷ `rake db:migrate`
3. Otwieramy plik **app/models/entry.rb**:

```
class Entry < ActiveRecord::Base
  validates :name, presence: true
  validates :comment, presence: true
end
```

Kontroler i trasowanie

1. ▷ `rails g controller entries index new create`
2. modyfikujemy plik **config/routes.rb**

```
Library::Application.routes.draw do
  # usuwamy:
  # get 'entries/index'
  # get 'entries/new'
  # get 'entries/create'
  resources :entries, only: [:index, :new, :create]
  resources :books
  resources :authors
  root "entries#index"
end
```

3. definiujemy 3 metody:
   - index – która wyświetla wszystkich gości
   - new – która wyświetla formularz do tworzenia wpisów
   - create – która zachowuje wpis w bazie danych

Kontroler EntriesController

app/controllers/entries_controller.rb

```ruby
# encoding: utf-8
class EntriesController < ApplicationController
  def index
    @entries = Entry.order("created_at DESC")
  end
  def new
    @entry = Entry.new
  end
  def create
    @entry = Entry.new(entry_params)
    if @entry.save
      redirect_to root_path, notice: 'Dziękujemy za wpis!'
    else
      render new_entry_path
    end
  end

  private
  def entry_params
    params.require(:entry).permit(:name, :comment)
  end
end
```

Widok – index

Otwieramy plik: **app/views/entries/index.html.erb**

```
<h1>Książka gości biblioteki</h1>
<p id="notice"><%= notice %></p>
<p>Ostatnie wpisy:</p>
<% @entries.each do |entry| %>
  <div>
          <div><%= entry.name %> napisał(a) <%= entry.created_at %>:</div>
          <div><%= entry.comment %></div>
  </div>
<% end %>

<br />
<%= link_to "Nowy wpis", new_entry_path %>
```

Widok – new

Otwieramy plik: **app/views/entries/new.html.erb**

```
<h1>Nowy wpis w księdze gości</h1>
<%= form_for @entry do |f| %>
  <% if @entry.errors.any? %>
  <ul>
    <% @entry.errors.full_messages.each do |msg| %>
      <li><%= msg %></li>
    <% end %>
  </ul>
  <% end %>
  <p>
    <%= f.label :name, "Twoje imię:" %><br/>
    <%= f.text_field :name  %>
  </p>
  <p>
    <%= f.label :comment, "Twój komentarz:" %> <br/>
    <%= f.text_area :comment %>
  </p>
  <%= f.submit "Dodaj" %>
<% end %>
```

Zadania (1):

- poproś użytkownika o dostarczenie linku do jego avatara
- używając migracji dodaj kolumnę w bazie danych do przechowywania URL-a do avatara
- wyświetl avatar używając helpera `image_tag`
- dodaj walidacje do modelu:
  - długości imienia i komentarza
  - format URL-a avatara

Zadania (2):

- zmodyfikuj akrusz stylów aby ulepszyć sposób wyświetlania wpisów
- użyj helpera `simple_format` (ActionView::Helpers::TextHelper), aby poprawnie wyświetlić wielo-linijkowe komentarze
- użyj helpera `time_ago_in_words` (ActionView::Helpers::DateHelper) aby ulepszyć wyświetlanie czasu, który upłynął od dodania wpisu
- użyj helpera `truncate` (ActionView::Helpers::TextHelper), aby przyciąć komentarze oraz dodaj link „więcej", który pozwala zobaczyć cały wpis
- użyj helpera `image_tag` (ActionView::Helpers::AssetTagHelper) i połącz go z metodą `link_to`, tak aby dostarczyć graficznej reprezentacji dla linku „Dodaj"