ЛАБОРАТОРНА РОБОТА 2.
РОЗРОБКА API ONLY ДОДАТКУ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ RAILS

Мета роботи: одержати практичні навики розробленні API додатку та його тестуванні.

Обладнання:

– встановлене програмне середовище ruby + Rails;

– будь-який редактор програмного коду: JetBarins WebStorm, Visual Studio Code, Sublime Text, Atom;

– підключення до мережі Інтернет.

2.1 Теоретичні відомості

1 Створення нового Rails додатку

```
rails new my_api_app —api
```

Ознайомтесь зі структурою згенерованого додатку та вмістом директорій, зокрема app.

2 З використанням scaffold (або з використанням інших вбудованих генераторів) згенеруйте моделі, які відповідають структурі додатку, який було описано у раніше сформованій специфікації на розробку проекту згідно індивідуального завдання.

Наприклад, додавання моделей для демонстрування роботи додатку:

```
rails g scaffold User name:string
rails g scaffold Post title:string body:text user:belongs_to
rails db:migrate
```

Вкажіть асоціації в моделях відповідно до схеми бази даних, наведеної у розробленій раніше специфікації. Вкажіть валідацію в моделях.

3 Налаштування маршрутів у файлі config/routes.rb. Зазвичай маршрути API вкладають в api/path. Крім того, розробники зазвичай вказують версію API в шляху, наприклад api/v1/:

```
namespace 'api' do
    namespace 'v1' do
      resources :posts
      resources :users
    end
  end
```

4 Налаштувати контролери.

Перемістіть файли контролерів з директорії controllers в controllers/api/v1/ і вкладіть у відповідні модулі:

```
module Api
    module V1
        class UsersController < ApplicationController
        end
    end
end
```

Змініть location в методі create з

```
render json: @post, status: :created, location: @post
```

на

```
render json: @post, status: :created, location: api_v1_post_path(@post)
```

5 Завантаження демонстраційних даних. Додайте відповідний гем до Gemfile в групу development (faker, ffaker або аналогічні):

```
group :development do
  gem 'faker'
end
```

Заповніть поля таблиць згенерованих моделей даними, які за своєю суттю відповідають предмету розробки (відповідній моделі).

Приклад вмісту файлу seeds.rb:

```
5.times do
  user = User.create({name: Faker::Name.name})
  user.posts.create({title: Faker::Book.title, body: Faker::Lorem.sentence})
end
```

Генерування даних:

```
rails db:seed
```

6 Налаштуйте представлення даних у форматі JSON з використанням гему jBuilder або active_model_serializers.

Приклад налаштування JSON з використанням jBuilder (файл views/api/v1/posts/index.json.jbuilder)

```
json.array! @posts do |post|
  json.id post.id
  json.title post.title
  json.body post.body
end
```

jBuilder підтримує частинні шаблони (паршали), як і будь-яке звичайне представлення Rails, наприклад

```
json.partial! partial: 'post', collection: @posts, as: :post
```

Для цього необхідно створити файл views/api/v1/posts/_post.json.jbuilder і перемістити в нього вміст наведеного вище ітератора.

Використання серіалізаторів. Генерування:

```
rails g serializer post
```

Приклад конфігурування наведено на рис. 2.1.

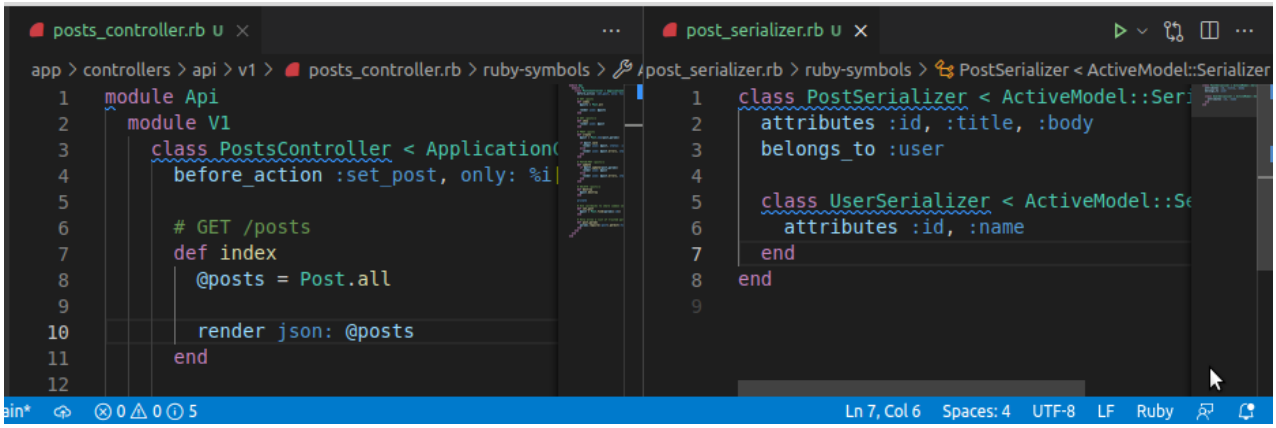

Рисунок 2.1 – Приклад налаштування методу index і відповідного серіалізатора

Тестування API додатку можна провести шляхом надсилання відновідних запитів методами GET, POST, DELETE і т. д. Для цього можна скористатись, наприклад, postnam - це інструмент для розробки та тестування API. Він дозволяє створювати запити до API та перевіряти їх відповіді, перевіряти доступність та правильність роботи API.

Postman надає зручний інтерфейс (рис. 2.2) для створення та відправки HTTP-запитів, включаючи GET, POST, PUT, DELETE, PATCH та інші методи. Він також дозволяє налаштовувати заголовки запиту, передавати параметри запиту, відправляти дані у вигляді JSON або іншого формату, тестувати аутентифікацію та багато іншого.

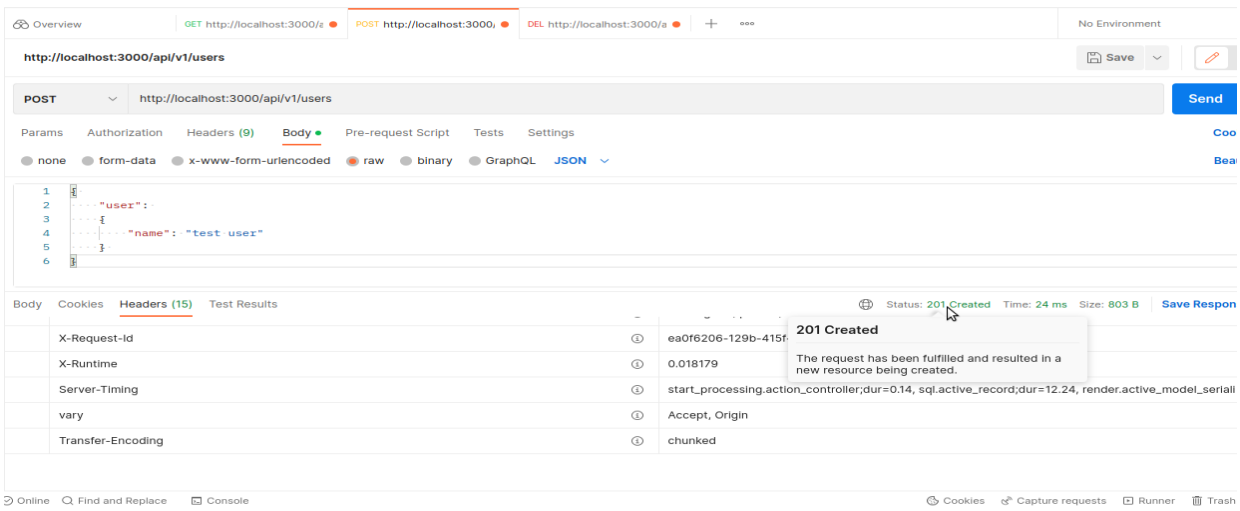

Рисунок 2.2 – Приклад генерування запиту до розробленого API додатку з використанням Postman

2.2 Порядок виконання роботи

1 Розробити Rails API додаток. Для цього виконати наступні кроки:

1.1 згенерувати додаток вказаної структури;

1.2 встановити гем (faker або ffaker) та налаштувати автоматичне заповнення даними таблиць бази даних додатку;

1.3 налаштувати асоціації між моделями, контролери;

1.4 з використанням jBuilder або active_model_serializers налаштувати формат представлення відповіді додатку.

2 Надіслати мінімум 3 різні запити (з використанням різних методів) до розробленого додатку з використанням будь-якого інструменту (наприклад, Postman). Зробити скріни екрану, які демонструють згенерований запит та статус відповіді.

3 Оформити звіт.

2.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;
- покроковий опис роботи, програмний код сценаріїв, копії екранів з результатами виконаної роботи.
- висновки.

Запитання для самоконтролю:

1 Що таке RESTful?

2 Структура запиту до API.

3 Що таке JSON Web Token?

Рекомендована література

1 Creating an API-Only Application with Ruby on Rails [Електронний ресурс]. – Режим доступу: https://gist.github.com/withoutwax/741c6e3c9a004c1463cbf7a1a3b719f8

2 API with Rails 7 [Електронний ресурс]. – Режим доступу: https://dev.to/nemwelboniface/api-with-rails-7-ngh

3 Офіційна документація по платформі Rails [Електронний ресурс]. – Режим доступу: https://guides.rubyonrails.org/v7.0/getting_started.html
4 Postman [Електронний ресурс]. – Режим доступу:https://www.postman.com/downloads/

Тривалість заняття: 4 год.