



Министерство науки и высшего образования Российской
Федерации
Федеральное государственное бюджетное образовательное
учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.03 Прикладная информатика

О Т Ч Е Т

по лабораторной работе № 11

Название: Добавление модели. ORM. Разработка БД, подключение,
хранение и поиск данных.

Дисциплина: Языки Интернет-программирования

Студент

ИУ6-35Б

(Группа)

19.12.2023

(Подпись,
дата)

В. И. Мамыкин

(И.О. Фамилия)

Преподаватель

Е. Ю. Гаврилова

(Подпись,
дата)

(И.О. Фамилия)

Москва, 2023

Цель: получение практических навыков в создании веб-приложений, использующих базы данных.

Задание:

Модифицировать код ЛР 8 таким образом, чтобы запросы, которые были ранее выполнены, сохранялись в БД и при следующем запросе не требовали повторного вычислений.

- Сформировать модель в соответствии с потребностями хранения данных. Входные параметры являются ключами, по которым извлекается результат.
- Выполнить создание БД и миграцию соответствующими запросами rake.
- Написать тест на добавление и поиск данных с помощью модели. Проверить выполнение теста.
- Модифицировать код приложения таким образом, чтобы результат вычислений преобразовывался в строковый или бинарный формат (на выбор: json, xml, и пр.). Проверить через отладочную печать в консоль, что преобразование выполняется корректно.
- Вставить код для сохранения данных в БД и запрос на поиск предыдущего результата вычислений.
- Добавить действие в контроллер, позволяющее определить, что хранится в БД через сериализацию в XML.
- Проверить, что при выполнении запроса, данные добавляются в БД.
- При помощи консоли сообщений Puma/Webrick определить, производится ли поиск результата предыдущего запроса в БД и не повторяются ли одни и те же вычисления.
- Модифицировать модель таким образом, чтобы добавление записей с одинаковыми параметрами было невозможно.

- Реализовать тест модели, проверяющий невозможность повторного добавления одних и тех же результатов вычислений.
- Реализовать функциональный тест, проверяющий, что результаты вычислений различны при различных входных параметрах.
- Проверить маршруты приложения с помощью rake routes и убрать лишние. Обеспечить доступ при обращении по адресу /.

sequences_controller.rb

```
# frozen_string_literal: true
require 'json'
```

```
class SequencesController < ApplicationController
  def input; end
```

```
  def view
```

```
    longest_subsequence = [] # Самая длинная подпоследовательность
```

```
    current_subsequence = [] # Текущая подпоследовательность
```

```
    all_subsequences = [] # Все подпоследовательности
```

```
    # unless params[:v2].nil?
```

```
    sequence = params[:n]&.split(' ').map(&:to_i)
```

```
    record = Result.find_by_input(sequence&.join(' '))
```

```
    #puts record
```

```
    if !record
```

```
      rec_exists = 0
```

```
      sequence&.each do |number|
```

```
        if (Math.sqrt(number) % 1).zero?
```

```
          # Если число является полным квадратом
```

```
          current_subsequence << number
```

```
        else
```

```

# Если число не является полным квадратом
longest_subsequence = current_subsequence.clone if
current_subsequence.length > longest_subsequence.length
cur = current_subsequence.clone
all_subsequences << cur.join(' ')
all_subsequences.pop if all_subsequences[all_subsequences.size - 1] == "
current_subsequence = []
end
end
cur = current_subsequence.clone
all_subsequences << cur.join(' ') if cur.length.positive?
longest_subsequence = current_subsequence.clone if
current_subsequence.length > longest_subsequence.length
all_subsequences.pop if all_subsequences[all_subsequences.size - 1] == "
subsequence_count = longest_subsequence.length
@result = [all_subsequences, longest_subsequence.join(' '),
subsequence_count.to_s, sequence&.join(' ')]
json_res = @result.to_json
result = Result.new(:input => sequence&.join(' '), :result => json_res)
result.save
print 'Запись добавлена в БД ', result, "\n"
else
rec_exists = 1
pp 'Record result:', record.result
@result = JSON.parse(record.result)
pp 'Результат уже посчитан'
end
@table = '<table>' # Начало таблицы
if @result[2] != '0'
@table +=

```

```
"<tr><td>Введенная
последовательность:</td></tr><tr><td>#{ @result[3]}</td></tr><tr><td>Подпо
следовательности:</td></tr>"
```

```
@result[0].each do |res|
  @table += "<tr><td>#{ res }</td></tr>"
end

@table +=
"<tr><td>Самая длинная
подпоследовательность:</td></tr><tr><td>#{ @result[1]}</td></tr><tr><td>Ее
длина:</td></tr><tr><td>#{ @result[2]}</td></tr>"
else
  @table +=
"<tr><td>Введенная
последовательность:</td></tr><tr><td>#{ @result[3]}</td></tr><tr><td>Подпо
следовательностей квадратов натуральных чисел нет</td></tr>"
end

@table += '</table>' # Конец таблицы
@tmp = @result.clone
@result = [@tmp, @table, rec_exists]
end
end
```

results_controller.rb

```
class ResultsController < ApplicationController
  before_action :set_result, only: [:show, :edit, :update, :destroy ]
  XSLT_TRANSFORM = "#{Rails.root}/public/xslt_transformer.xslt".freeze #
  Путь до xslt файла
```

```

# GET /results or /results.json
def index
  @results = Result.all
end

def show_all
  results = Result.all
  rows = ""
  result_saved = true

  results.each do |record|
    if record.save
      rows +=
" <cd><id>#{record.input}</id><item>#{record.result}</item></cd>"
    else
      result_saved = false
      break
    end
  end

  if result_saved
    response = "<catalog>#{rows}</catalog>"
    render xml: response
  else
    respond_to do |format|
      format.xml { render 'new', status: :unprocessable_entity }
    end
  end
end

# GET /results/1 or /results/1.json

```

```

def show
end

# GET /results/new
def new
  @result = Result.new
end

# GET /results/1/edit
def edit
end

# POST /results or /results.json
def create
  @result = Result.new(result_params)

  respond_to do |format|
    if @result.save
      format.html { redirect_to result_url(@result), notice: "Result was successfully
created." }
      format.json { render :show, status: :created, location: @result }
    else
      format.html { render :new, status: :unprocessable_entity }
      format.json { render json: @result.errors, status: :unprocessable_entity }
    end
  end
end

# PATCH/PUT /results/1 or /results/1.json
def update

```

```

respond_to do |format|
  if @result.update(result_params)
    format.html { redirect_to result_url(@result), notice: "Result was successfully
updated." }
    format.json { render :show, status: :ok, location: @result }
  else
    format.html { render :edit, status: :unprocessable_entity }
    format.json { render json: @result.errors, status: :unprocessable_entity }
  end
end
end
end

```

```

# DELETE /results/1 or /results/1.json

```

```

def destroy

```

```

  @result.destroy!

```

```

respond_to do |format|
  format.html { redirect_to results_url, notice: "Result was successfully
destroyed." }
  format.json { head :no_content }
end
end
end

```

```

private

```

```

# Use callbacks to share common setup or constraints between actions.

```

```

def set_result

```

```

  @result = Result.find(params[:id])

```

```

end

```

```

# Only allow a list of trusted parameters through.

```



```
def result_params
  params.require(:result).permit(:input, :result)
end
end
```

routes.rb

```
Rails.application.routes.draw do
  resources :results

  get 'db_index', to: 'results#index'
  get 'db_new', to: 'results#new'
  get 'show_all', to: 'results#show_all'

  get 'sequences/input'
  get 'sequences/view'
  root 'sequences#input'
end
```

result_controller_test.rb

```
require 'test_helper'

class ResultsControllerTest < ActionDispatch::IntegrationTest
  setup do
    @result = results(:one)
  end

  test 'should get show_all' do
    get results_url
    assert_response :success
  end
end
```

```
test 'should get new' do
```

```
  get new_result_url
```

```
  assert_response :success
```

```
end
```

```
test 'should create result' do
```

```
  assert_difference('Result.count') do
```

```
    post results_url, params: { result: { input: @result.input, result: @result.result } }
```

```
  end
```

```
  assert_redirected_to result_url(Result.last)
```

```
end
```

```
test 'should show result' do
```

```
  get result_url(@result)
```

```
  assert_response :success
```

```
end
```

```
test 'should get edit' do
```

```
  get edit_result_url(@result)
```

```
  assert_response :success
```

```
end
```

```
test 'should update result' do
```

```
  patch result_url(@result), params: { result: { input: @result.input, result: @result.result } }
```

```
  assert_redirected_to result_url(@result)
```

```
end
```

```
test 'should destroy result' do
  assert_difference('Result.count', -1) do
    delete result_url(@result)
  end
end
```

```
  assert_redirected_to results_url
end
end
```

Распечатка данных БД таблицы result:

```
results id integer PRIMARY KEY PRIMARY KEY AUTOINCREMENT NOT
NULL NOT NULL input integer result json created_at datetime NOT NULL NOT
NULL updated_at datetime NOT NULL NOT NULL 1 10 "{\"11\":13,\"17\":19}"
2023-11-08 16:33:04.789709 2023-11-08 16:33:04.789709 2 11
"{\"11\":13,\"17\":19}" 2023-11-08 16:33:07.187956 2023-11-08 16:33:07.187956
3 15 "{\"17\":19}" 2023-11-08 16:33:09.666411 2023-11-08 16:33:09.666411
```

Вставка данных в таблицу:

```
INSERT INTO "results" ("input", "result", "created_at", "updated_at") VALUES
(?, ?, ?, ?) [["input", 15], ["result", "\"{\\\"17\\\":19}\""], ["created_at", "2023-11-
08 16:33:09.666411"], ["updated_at", "2023-11-08 16:33:09.666411"]]
```

Вывод данных из таблицы:

```
SELECT "results".* FROM "results" WHERE "results"."input" = ? LIMIT ?
[["input", 15], ["LIMIT", 1]]
```

```

Started GET "/sequences/view?n=1+2+9+8+7+6" for 127.0.0.1 at 2024-01-14 13:35:26 +0300
Processing by SequencesController#view as HTML
Parameters: {"n"=>"1 2 9 8 7 6"}
Result Load (0.1ms) SELECT "results".* FROM "results" WHERE "results"."input" = ? LIMIT ? [["input", "1 2 9 8 7 6"], ["LIMIT", 1]]
↳ app/controllers/sequences_controller.rb:13:in 'view'
TRANSACTION (0.0ms) begin transaction
↳ app/controllers/sequences_controller.rb:38:in 'view'
Result Create (1.2ms) INSERT INTO "results" ("input", "result", "created_at", "updated_at") VALUES (?, ?, ?, ?) RETURNING "id" [["input", "1 2 9 8 7 6"], ["result", "1 2 9 8 7 6"], ["created_at", "2024-01-14 10:35:26.446917"], ["updated_at", "2024-01-14 10:35:26.446917"]]
↳ app/controllers/sequences_controller.rb:38:in 'view'
TRANSACTION (0.1ms) commit transaction
↳ app/controllers/sequences_controller.rb:38:in 'view'
Запись добавлена в БД #<Result:0x0000286ee7f6198>
Rendering layout layouts/application.html.erb
Rendering sequences/view.html.erb within layouts/application
Rendered sequences/view.html.erb within layouts/application (Duration: 0.3ms | Allocations: 183)
Rendered layout layouts/application.html.erb (Duration: 6.2ms | Allocations: 2395)
Completed 200 OK in 15ms (Views: 7.7ms | ActiveRecord: 1.4ms | Allocations: 4513)

```

Рисунок 1 – Вывод консоли при работе с БД

sequences#input

Мы находимся по адресу: <http://127.0.0.1:3000/sequences/input.html.erb>

Последовательность:

Рисунок 2 – Ввод данных в поле

sequences#view

Мы находимся по адресу: <http://127.0.0.1:3000/sequences/view.html.erb>

Введенная последовательность:
1 2 3 4 9 16 3 2 4 10
Подпоследовательности:
1
4 9 16
4
Самая длинная подпоследовательность:
4 9 16
Ее длина:
3

[Рассчитать заново](#)

Рисунок 3 – Вывод результата

```
PS D:\Documents\GitHub\BMSTU\3semestr\YAIP\lab11_2_version\Project\sequences> rake test TEST=test/models/result_test.rb
Running 3 tests in a single process (parallelization threshold is 50)
Run options: --seed 33904

# Running:

...

Finished in 0.067596s, 44.3815 runs/s, 44.3815 assertions/s.
3 runs, 3 assertions, 0 failures, 0 errors, 0 skips
```

Рисунок 4 – Тесты модели

Вывод: были получены практических навыков в создании веб-приложений, использующих базы данных.