



**«Московский государственный технический университет
имени Н.Э. Баумана»**

(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

Дисциплина: Языки Интернет-программирования

Название лабораторной работы: Добавление модели. ORM. Разработка БД, подключение, хранение и поиск данных. Вариант 10.

Студент гр. ИУ6-34Б

(Подпись, дата)

К.Ю. Каташинский

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

(И.О. Фамилия)

Москва, 2020

Условие задания:

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

Результат приложить в виде двух файлов:

- архив, содержащий `RoR`-приложение;
- `pdf`-отчет, в котором должны присутствовать фрагменты добавленного кода.

Отчет должен содержать:

- ФИО, номер группы и текст задания;
- перечень и содержимое файлов, которые были изменены в процессе создания приложения.
- `XML`-распечатку содержимого БД (ограничить несколькими записями так, чтобы результат поместился на 1-2 страницах).
- Примеры `SQL`-кода добавления и извлечения данных из БД из отладочной консоли сервера `Webrick`.

Код файла base_controller.rb:

```
class BaseController < ApplicationController
  def index
    @result = LabResult.all
  end
  def create
    number = params[:number].to_i
    array = params[:str].split.map{|elem| elem.to_i}
    @error = number != array.length ? 'Введите n чисел' : ''
    @array = array.join(" ")
    @is_new_element = true
    if res = LabResult.find_by_array(@array)
      @all_segments, @max_segment, @count = res.all_segments, res.max_segment,
res.count
      @is_new_element = false
      @result = LabResult.all
    else
      @all_segments = array.chunk_while do |first, second|
        multiply_five?(first) == multiply_five?(second)
      end.find_all {|segment| multiply_five?(segment[0]) }
      @count = @all_segments.length
      @max_segment = @all_segments.max_by { |segment| segment.length }
      @all_segments = @all_segments.join(" ")
      @max_segment = @max_segment&.join(" ")
      if @error != ''
        @result = LabResult.all
        return
      else
        @all_segments = ('Нет отрезков') if @all_segments == ''
        @max_segment = ('Нет отрезка') if @all_segments == 'Нет отрезков'
        @result = LabResult.all
        res = LabResult.create(:array => @array, :all_segments => @all_segments,
:max_segment => @max_segment, :count => @count)
        res.save
      end
    end
  end
  def results
    @result = LabResult.all
    render xml: @result
  end
  private def multiply_five?(elem)
    x = Math.log(elem, 5)
    elem != 1 ? (x.round == x) : false
  end
end
```

Код файла lab_result.rb:

```
class LabResult < ApplicationRecord
  validates_uniqueness_of :array
  validates_presence_of :array, :all_segments, :max_segment, :count
end
```

Код файла index.html.erb:

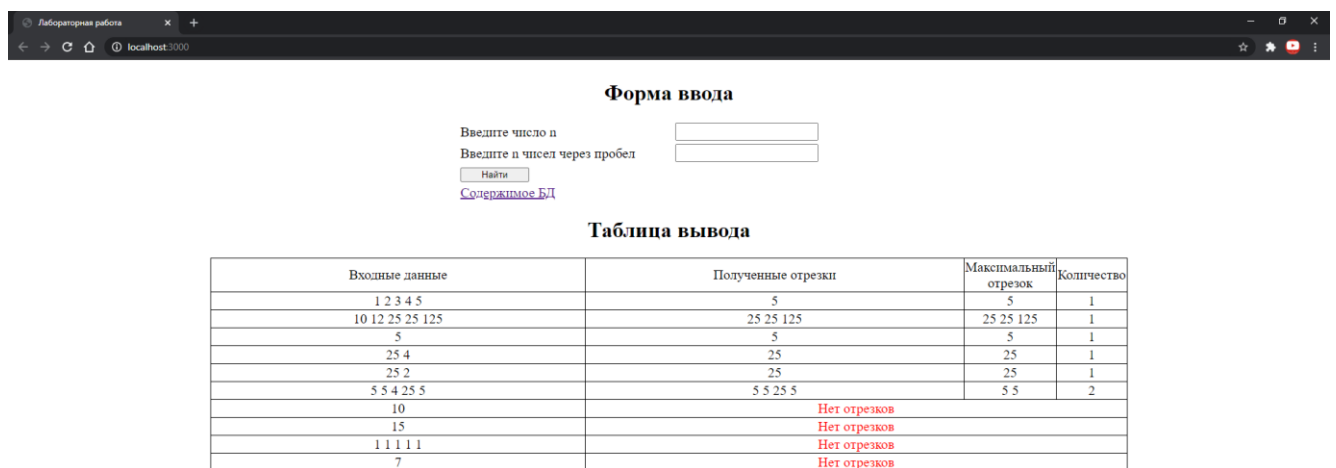
```
<%= form_tag("/create", :method => "post") do %>
  <table id="add_form">
    <caption><h1>Форма ввода</h1></caption>
    <tr>
      <td width="40%">Введите число n</td>
      <td><%= text_field_tag :number, nil, class: "enter" %></td>
    </tr>
    <tr>
      <td>Введите n чисел через пробел</td>
      <td><%= text_field_tag :str, nil, class: "enter" %></td>
    </tr>
    <tr>
      <td><%= submit_tag "Найти", class: "btn" %></td>
      <td><span style="color: red;"></span></td>
    </tr>
    <tr>
      <td><a href="/results">Содержимое БД</a></td>
      <td></span></td>
    </tr>
  </table>
<% end %>
<table id="add_table">
  <caption><h1>Таблица вывода</h1></caption>
  <tr>
    <td class="col">Входные данные</td>
    <td class="col">Полученные отрезки</td>
    <td class="col">Максимальный отрезок</td>
    <td class="col">Количество</td>
  </tr>
  <% @result.each do |item| %>
    <tr>
      <% if item.all_segments != 'Нет отрезков' %>
        <td class="col"><%= item.array %></td>
        <td class="col"><%= item.all_segments %></td>
        <td class="col"><%= item.max_segment %></td>
        <td class="col"><%= item.count %></td>
      <% else %>
        <td class="col"><%= item.array %></td>
        <td colspan="3" id="error_cell">Нет отрезков</td>
      <% end %>
    </tr>
  <% end %>
</table>
```

Код файла create.html.erb:

```
<%= form_tag("/create", :method => "post") do %>
  <table id="add_form">
    <caption><h1>Форма ввода</h1></caption>
    <tr>
      <td width="40%">Введите число n</td>
      <td><%= text_field_tag :number, nil, class: "enter" %></td>
    </tr>
    <tr>
      <td>Введите n чисел через пробел</td>
      <td><%= text_field_tag :str, nil, class: "enter" %></td>
    </tr>
    <tr>
      <td><%= submit_tag "Найти", class: "btn" %></td>
      <td><span style="color: red;"><%= @error %></span></td>
    </tr>
    <tr>
      <td><a href="/results">Содержимое БД</a></td>
      <td></span></td>
    </tr>
  </table>
<% end %>
<table id="add_table">
  <caption><h1>Таблица вывода</h1></caption>
  <tr>
    <td class="col">Входные данные</td>
    <td class="col">Полученные отрезки</td>
    <td class="col">Максимальный отрезок</td>
    <td class="col">Количество</td>
  </tr>
  <% @result.each do |item| %>
    <tr>
      <% if item.all_segments != 'Нет отрезков' %>
        <td class="col"><%= item.array %></td>
        <td class="col"><%= item.all_segments %></td>
        <td class="col"><%= item.max_segment %></td>
        <td class="col"><%= item.count %></td>
      <% else %>
        <td class="col"><%= item.array %></td>
        <td colspan="3" id="error_cell">Нет отрезков</td>
      <% end %>
    </tr>
  <% end %>
  <% unless @is_new_element %>
    <tr>
      <% if @all_segments != 'Нет отрезков' %>
        <td class="col"><%= @array %></td>
        <td class="col"><%= @all_segments %></td>
        <td class="col"><%= @max_segment %></td>
        <td class="col"><%= @count %></td>
      <% else %>
        <td class="col"><%= @array %></td>
        <td colspan="3" id="error_cell">Нет отрезков</td>
      <% end %>
    </tr>
  <% end %>
</table>
```

Код файла lab_result_test.rb:

```
require 'test_helper'
class LabResultTest < ActiveSupport::TestCase
  def setup
    @one_result = LabResult.new(array: "25 4", all_segments: "25", max_segment:
"25", count: 1)
  end
  test 'array should be unique' do
    duplicate_one_result = @one_result.dup
    @one_result.save
    assert_not duplicate_one_result.valid?
  end
  test "array should be present" do
    @one_result.array = ""
    assert_not @one_result.valid?
  end
  test "all_segments should be present" do
    @one_result.all_segments = ""
    assert_not @one_result.valid?
  end
  test "max_segment should be present" do
    @one_result.max_segment = ""
    assert_not @one_result.valid?
  end
end
end
```



The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. The page title is 'Лабораторная работа'. The main content area is titled 'Форма ввода' (Input Form). It contains two input fields: 'Введите число n' (Enter number n) and 'Введите n чисел через пробел' (Enter n numbers separated by a space). Below the inputs is a 'Найти' (Find) button and a link 'Содержимое БД' (Database Content). Below the form is a table titled 'Таблица вывода' (Output Table). The table has four columns: 'Входные данные' (Input data), 'Полученные отрезки' (Obtained segments), 'Максимальный отрезок' (Maximum segment), and 'Количество' (Quantity). The table contains 14 rows of data, including various input strings and their corresponding results, with some rows showing 'Нет отрезков' (No segments) in red text.

Входные данные	Полученные отрезки	Максимальный отрезок	Количество
1 2 3 4 5	5	5	1
10 12 25 25 125	25 25 125	25 25 125	1
5	5	5	1
25 4	25	25	1
25 2	25	25	1
5 5 4 25 5	5 5 25 5	5 5	2
10	Нет отрезков		
15	Нет отрезков		
1 1 1 1 1	Нет отрезков		
7	Нет отрезков		

Рисунок 1 «Пример выполнения программы»

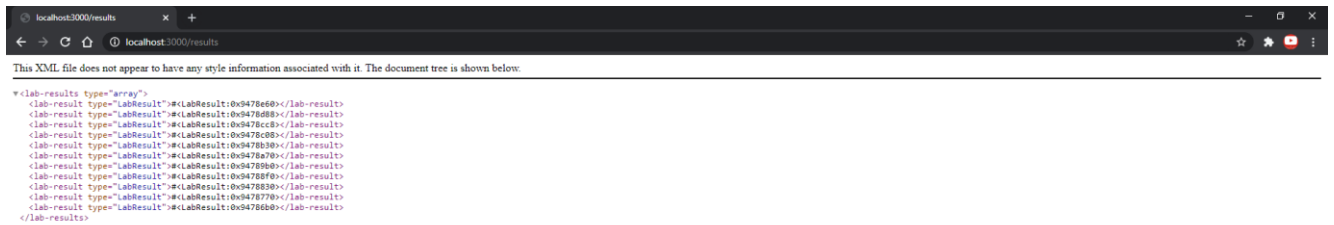


Рисунок 2 «XML распечатка БД»

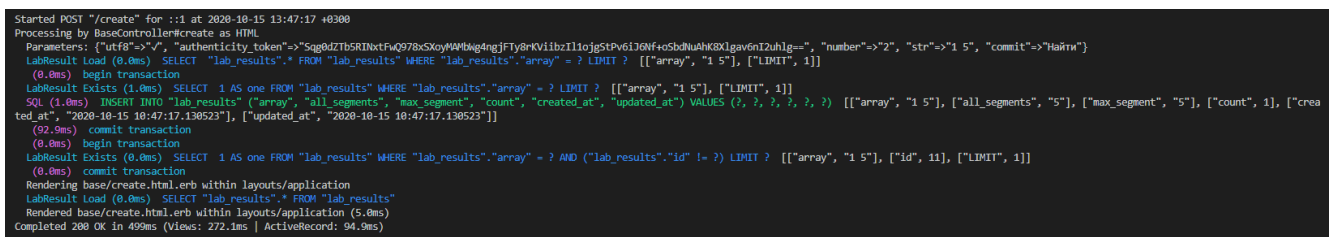


Рисунок 2 «Примеры SQL-кода добавления и извлечения данных из БД»

Вывод

В ходе этой лабораторной работы было произведено первое знакомство с работой с БД в гог-приложении.