***Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования***

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ *Информатика и системы управления*

КАФЕДРА *Компьютерные системы и сети*

**Отчет**

**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.  **ИУ6-33Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Захаров К.А**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Самарев Р.С.**

(Подпись, дата) (И.О. Фамилия)

Москва, 2019

**Задание:**

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 /.

Измененные файлы:

config/routes.rb  
db/schema.rb  
app/views/lb11/input.html.erb

app/views/lb11/view.html.erb  
app/controllers/lb11\_controller.rb  
app/assets/javascripts/lb11.js  
test/controllers/lb11\_controller\_test.rb  
test/models/mersen\_test.rb

**input.html.erb**

**<h1>Lb11#input</h1>**

**<p>Find me in app/views/lb11/input.html.erb</p>**

**<%= form\_tag("/lb11/view.json", :method => "get",**

**remote: 'true', id: "input\_form") do %>**

**<%= label\_tag("Input border value") %>v**

**<%= number\_field\_tag(:input) %>**

**<%= submit\_tag("result", id: "create\_table") %>**

**<% end %>**

**<br/>**

**<%= link\_to ('Data base'), lb11\_all\_path %>**

**<br/>**

**<div id="result"></div>**

**view.html.erb**

**<h1>Lb11#view</h1>**

**<p>Find me in app/views/lb11/view.html.erb</p>**

**<table border="2">**

**<% if @result.type == "Error" then %>**

**<p><%= @result.value %></p>**

**<% else %>**

**<table>**

**<tr class="h">**

**<td>#</td>**

**<td>p</td>**

**<td>M</td>**

**</tr>**

**<% @result.map do |el| %>**

**<tr>**

**<td class="h"><%= el[:id] %></td>**

**<td><%= el[:p] %></td>**

**<td><%= el[:M] %></td>**

**</tr>**

**<% end %>**

**</table>**

**<% end %>**

**app/controllers/lb11\_controller.rb**

**require 'prime'**

**class Lb11Controller < ApplicationController**

**before\_action :parse\_params, only: :view**

**def input**

**end**

**def all**

**@mersens = Mersen.all**

**render xml: @mersens.as\_json.to\_xml**

**end**

**def view**

**if !(@input =~ /-/ || @input == "" || @input == '0')**

**do\_mersen**

**respond\_to do |format|**

**format.html**

**format.json do**

**render json:**

**{ type: @result.class, value: @result }**

**end**

**end**

**else**

**frow\_erorr**

**end**

**end**

**private**

**def parse\_params**

**@input = params[:input]**

**end**

**def frow\_erorr**

**@result = "Wrong input!"**

**respond\_to do |format|**

**format.html**

**format.json do**

**render json:**

**{ type: 'Error', value: @result }**

**end**

**end**

**end**

**def do\_mersen**

**@input = @input.to\_i**

**max\_mersen = Mersen.order(:M).last**

**# max\_mersen = Mersen.maximum('M')**

**# max\_p = Mersen.maximum('p')**

**# max\_i = Mersen.maximum('i')**

**if max\_mersen.nil?**

**add\_mersen(1,0)**

**else**

**add\_mersen(max\_mersen.p,max\_mersen.i)**

**end**

**@result = Mersen.where("M <= ?", @input) # find\_by\_sql("select \* from Mersens where M <= #{@input}")**

**end**

**def add\_mersen(p\_start, i\_start)**

**p\_end = (Math.log(@input) / Math.log(2)).to\_i**

**p\_end += Prime.prime?(@input) ? 1 : 0**

**i = i\_start**

**arr\_hash = []**

**arr = (p\_start..p\_end).to\_a.keep\_if { |num| Prime.prime?(2\*\*num - 1) }.map do |el|**

**add\_new\_mersen(i: i + 1, p: el, M: (2\*\*el - 1))**

**arr\_hash.push(i: i + 1, p: el, M: (2\*\*el - 1))**

**i += 1**

**end**

**end**

**def add\_new\_mersen(param\_mersen)**

**new\_mersen = Mersen.new(param\_mersen)**

**new\_mersen.save**

**end**

**end**

**app/assets/javascripts/lb11.js**

**function show\_result(data) {**

**var result = document.getElementById("result");**

**var html = "<table>\n<tr class='h'>\n<td>#</td>\n<td>p</td>\n<td>M</td>\n</tr>\n ";**

**data.forEach( (el) => {**

**html += "<tr>\n<td>" + el.i + "</td>\n" + "<td>" + el.p + "</td>\n" + "<td>" + el.M + "</td>\n </tr>\n";**

**})**

**result.innerHTML = html + "</table>"**

**}**

**function show\_error(data) {**

**var result = document.getElementById("result");**

**result.innerHTML = data;**

**}**

**$(document).ready(function() {**

**$("#input\_form").on("ajax:success", (event) => {**

**[data, status, xhr] = event.detail;**

**console.log(data,status,xhr)**

**if (data.type == "Error")**

**show\_error(data.value);**

**else**

**show\_result(data.value);**

**})**

**})**

**test/controllers/lb11\_controller\_test.rb**

**require 'test\_helper'**

**require 'selenium-webdriver'**

**class Lb11ControllerTest < ActionDispatch::IntegrationTest**

**def setup**

**@driver = Selenium::WebDriver.for :chrome**

**@driver.navigate.to 'http://localhost:3000/'**

**@element = @driver.find\_element(id: 'input')**

**end**

**test "should get input" do**

**get lb11\_input\_url**

**assert\_response :success**

**@driver.quit**

**end**

**test "should get view" do**

**get lb11\_view\_url, params: { input: '7' }**

**assert\_response :success**

**@driver.quit**

**end**

**test 'selenium1' do**

**@element.send\_keys 7**

**@driver.find\_element(id: 'create\_table').click**

**table = @driver.find\_element(id: 'result')**

**assert\_equal("# p M\n1 2 3\n2 3 7", table.text)**

**@driver.quit**

**end**

**test 'selenium2' do**

**@element.send\_keys 31**

**@driver.find\_element(id: 'create\_table').click**

**table = @driver.find\_element(id: 'result')**

**@driver.find\_element(id: 'create\_table').click**

**assert\_equal("# p M\n1 2 3\n2 3 7\n3 5 31", table.text)**

**@driver.quit**

**end**

**test 'selenium3' do**

**@element.send\_keys ''**

**@driver.find\_element(id: 'create\_table').click**

**table = @driver.find\_element(id: 'result')**

**@driver.find\_element(id: 'create\_table').click**

**assert\_equal("Wrong input!", table.text)**

**@driver.quit**

**end**

**# test 'should get Wrong input!' do**

**# get lb11\_view\_url, params: { input: "-10"}**

**# assert\_equal 'Wrong input!', @result.value**

**# end**

**end**

**test/models/mersen\_test.rb** **require 'test\_helper'**

**class MersenTest < ActiveSupport::TestCase**

**test "should not create Mersen" do**

**mersen = Mersen.new**

**assert !mersen.save**

**end**

**test 'should find Mersen' do**

**assert Mersen.exists?(:M => 1)**

**end**

**test 'shoud not save with exists fields' do**

**mersen = Mersen.new**

**mersen.i = 1**

**mersen.p = 2**

**mersen.M = 3**

**assert !mersen.save**

**end**

**end**

**config/routes.rb**

**Rails.application.routes.draw do**

**root 'lb11#input'**

**get 'lb11/all'**

**get 'lb11/input'**

**get 'lb11/view'**

**# For details on the DSL available within this file, see http://guides.rubyonrails.org/routing.html**

**end**

**db/schema.rb**

**ActiveRecord::Schema.define(version: 2018\_12\_03\_062706) do**

**create\_table "mersens", force: :cascade do |t|**

**t.integer "i"**

**t.integer "p"**

**t.integer "M"**

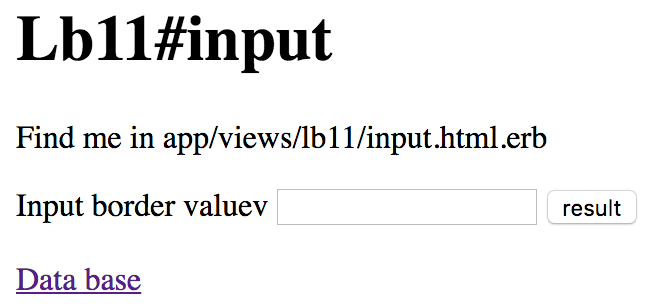
**t.datetime "created\_at", null: false**

**t.datetime "updated\_at", null: false**

**end**

**end**

**Работа программы:**

****

****

**Автоматические тесты:**

**Run options: --seed 5738**

**# Running:**

**......**

**Finished in 12.292872s, 0.4881 runs/s, 0.4881 assertions/s.**

**6 runs, 6 assertions, 0 failures, 0 errors, 0 skips**

**Проверка rubocop:**

**Inspecting 45 files**

**.............................................**

**45 files inspected, no offenses detected**

**Вывод:** Модифицировали код ЛР 8 таким образом, чтобы запросы, которые были ранее выполнены, сохранялись в БД и при следующем запросе не требовали повторных вычислений.