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 № 9

Название: СОЗДАНИЕ ПРОСТЕЙШИХ ВЕБ-ПРИЛОЖЕНИЙ RUBY ON RAILS. AJAX

Дисциплина: Языки интернет-программирования

Студент ИУ6-35 Б (Группа) 24.11.2023 (Подпись, дата) И.А. Дулина (И.О. Фамилия)

Преподаватель (Подпись, дата) Е.Ю. Гаврилова (И.О. Фамилия)

Москва, 2023

Вариант 8

Цель работы: углубление теоретических сведений о принципах работы асинхронного веб-интерфейса и получение практических навыков создания веб-приложения с использованием средств Ruby on Rails и технологии AJAX.

Задание:

- При помощи Javascript модифицировать код ЛР 8 таким образом, чтобы для отображения результатов вычисления браузер не выполнял полную перезагрузку страницы.
- Сформировать тесты для проверки работы программы при помощи Katalon Recorder / Selenium Webdriver.

Создаём приложение

$ rails new chisla9

Создаём контроллер

$cd chisla

$rails generate controller Chisla input

В gem file добавляем строки

gem "jquery-rails"

gem 'rails-controller-testing'

В config/importmap.rb добавляем:

pin_all_from 'app/javascript/src', under: 'src'

pin "jquery", to: "jquery.min.js", preload: true

pin "jquery_ujs", to: "jquery_ujs.js", preload: true

В config/initializers/assets.rb добавляем:

Rails.application.config.assets.precompile += %w(jquery.min.js jquery_ujs.js)

В папке app/javascript добавляем папку src, где создаём файл chisla.js

В app/javascript/application.js добавляем:

```
import "jquery"
import "jquery_ujs"
import "popper"
import "bootstrap"
import "src/chisla"
```

Код файла app/views/chisla/input.html.erb

```
<h1>Chisla#input</h1>
<p>Find me in app/views/chisla/input.html.erb</p>
<%=form_tag("/chisla/input.json", :method => "get", remote: true, id:
'chisla_form') do %>
  <%=label_tag("Введите не менее 10 чисел")%>
  <%=text_field_tag(:str)%><br/><br/>
  <%=submit_tag("Найти наиболее длинную монотонно возврастающую
последовательность")%>
<% end%>
<div id="result"></div>
```

Код файла app/controllers/chisla_controller.rb

```
# frozen_string_literal: true

class ChislaController < ApplicationController
  def input
    if params[:str]
      begin
      res = params[:str].scan(/-?\d+(?:\.\d+)?/).map(&:to_i)
      raise StandardError if res.length<10
```

```
    pp 'RES:', res, params[:str]
    @result = create(res)
    rescue StandardError
      @result=['', 'Что-то пошло не так']
    end
  else
    @result = ['', 'Unknown!']
  end

  respond_to do |format|
    format.html
    format.json do
      render json:
               { type: @result.class.to_s, value: @result }
    end
  end
end

def create(res)
  max = 0
  all = []
  solution = ''
  i = 0
  loop do
    posl, len, i = create_posl(i, res)
    all << posl.join(' ')
    if len > max
      max = len
      solution = posl.join(' ')
    end
```

```
      break if i >= res.length
    end
    result = []
    all.length.times do |j|
      str = if solution == all[j]
              '+'
            else
              ' '
            end
      result << if j.zero?
                  [res.join(' '), all[j], str]
                else
                  [' ', all[j], str]
                end
    end
    [solution, create_table(result)]
  end

  def create_table(result)
    rows = "<tr><th>#{'Изначальный'}</th><th>#{'Все
возможные'}</th><th>#{'Самая длинная'}</th></tr>"
    result.each do |init, all, sol|
      rows += "<tr><td>#{init}</td><td>#{all}</td><td>#{sol}</td></tr>"
    end
    @table = "<table border='1' class=\"table\"><tbody>#{rows}</tbody></table>"
  end
  def create_posl(i, res)
    len = 0
    posl = []
    loop do
```

```
      len += 1
      posl << res[i]
      break if i + 1 == res.length
      break if (res[i+1] <= res[i])
      i += 1
    end
    i+=1
    [posl, len, i]
  end
end
```

Код файла app/javascript/src/chisla.js

```
function show_result(data){
    const result = document.getElementById("result");
    result.innerHTML = "<hr/>Result is: " + data.value +
        "<hr/><p>"+Date()+"</p>";
}
$(document).ready(function(){
    $("#chisla_form").bind("ajax:success",
        function(xhr, data, status) {
// data is already an object
            show_result(data)
        })
})
```

Код файла test/controllers/ chisla_controller_test.rb

```
require "test_helper"

class ChislaControllerTest < ActionDispatch::IntegrationTest
```

```
  test "should get input" do
    get chisla_input_url
    assert_response :success
  end
  test 'for right1' do
    get chisla_input_url, params: { str: '3 5 5 4 2 6 7 8 4 3' }
    assert_equal assigns[:result][0], '2 6 7 8'
  end

  test 'for right2' do
    get chisla_input_url, params: { str: '-1 2 6 -4 5 6 7 2 1 0 1' }
    assert_equal assigns[:result][0], '-4 5 6 7'
  end

  test 'for error' do
    get chisla_input_url
    assert_equal assigns[:result], ['', 'Что-то пошло не так']
  end
end
```

Код файла config/routes.rb

```
Rails.application.routes.draw do
  get 'chisla/input'
  # Define your application routes per the DSL in
https://guides.rubyonrails.org/routing.html

  # Defines the root path route ("/")
  # root "articles#index"
  root 'chisla#input'
```

end

Rspec и Selenium:

В gemfile добавляем

gem 'rspec-rails'

gem 'capybara'

gem 'selenium-webdriver'

в консоль:

$rails generate rspec:install

$rails s

В spec/ добавляем файл test_chisla_spec.rb

Записываем туда код, сгенерированный Katalon Recorder в формате Ruby

Добавляем require 'rails_helper'

Заменяем везде `${receiver}` на `@driver` в связи с ошибкой генерации Katalon.

Запускаем тест в консоли

$bundle exec rspec spec/test_chisla_spec.rb

Код файла spec/test_chisla_spec.rb

```
# frozen_string_literal: true
require 'rails_helper'
require 'json'
require 'selenium-webdriver'
require 'rspec-rails'

include RSpec::Expectations

describe 'UntitledTestCase' do
```

```
before(:each) do
  @driver = Selenium::WebDriver.for :firefox
  @base_url = 'https://www.google.com/'
  @accept_next_alert = true
  @driver.manage.timeouts.implicit_wait = 30
  @verification_errors = []
end

after(:each) do
  @driver.quit
  expect(@verification_errors).to match_array([])
end

it 'test_untitled_test_case' do
  values = ['1 2 3 4 5 6 1 2 3 1', '1 -1 2 3 1 4 6 1 2 3 4']
  targets = ['1 2 3 4 5 6', '1 2 3 4']

  values.length.times do |i|
    val = values[i]
    target = targets[i]
    @driver.get 'http://127.0.0.1:3000/'
    @driver.find_element(:id, 'str').click
    @driver.find_element(:id, 'str').clear
    @driver.find_element(:id, 'str').send_keys val
    @driver.find_element(:name, 'commit').click
    # Обрабатываем результат
    element = @driver.find_element(:id, 'result')
    text = element.text.split
    arr = ''
    iter = 0
```

```
    text.each do |num|
      break unless Integer(num, exception: false)
      arr += num + ' '
      iter += 1
    end
    arr += text[iter].chop
    verify { expect(arr).to match(target) }
  end
end

def element_present?(how, what)
  @driver.find_element(how, what)
  true
rescue Selenium::WebDriver::Error::NoSuchElementError
  false
end

def alert_present?
  @driver.switch_to.alert
  true
rescue Selenium::WebDriver::Error::NoAlertPresentError
  false
end

def verify
  yield
rescue ExpectationNotMetError => e
  @verification_errors << e
end
```

```
  def close_alert_and_get_its_text(_how, _what)
    alert = @driver.switch_to.alert
    alert_text = alert.text
    if @accept_next_alert
      alert.accept
    else
      alert.dismiss
    end
    alert_text
  ensure
    @accept_next_alert = true
  end
end
```

Chisla#input

Find me in app/views/chisla/input.html.erb

Введите не менее 10 чисел -5 -4 -3 -2 -1 0 1 2 3 2 1 5 6

Найти наиболее длинную монотонно возврастающую последовательность

Result is:

Изначальный	Все возможные	Самая длинная
-5 -4 -3 -2 -1 0 1 2 3 2 1 5 6 6 7 8	-5 -4 -3 -2 -1 0 1 2 3	+
	2	
	1 5 6	
	6 7 8	

Thu Nov 09 2023 23:48:49 GMT+0300 (Москва, стандартное время)

Рисунок 1 – страница с вводом и выводом результата

```
Finished in 1.158654s, 3.4523 runs/s, 3.4523 assertions/s.
4 runs, 4 assertions, 0 failures, 0 errors, 0 skips
```

Рисунок 2 – результат автоматического тестирования ($ rake test)

```
1 example, 0 failures, 1 passed
Finished in 6.1995056 seconds
```

Рисунок 3 – результат теста, написанный при помощи Selenium IDE

Вывод: были изучены теоретические сведения о принципах работы асинхронного веб-интерфейса и было создано веб-приложение с использованием средств Ruby on Rails и технологии AJAX.