|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 9 |

**Название:**

СОЗДАНИЕ ПРОСТЕЙШИХ ВЕБ-ПРИЛОЖЕНИЙ

RUBY ON RAILS. AJAX

**Дисциплина:** Языки интернет-программирования



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-35 Б |  | 24.11.2023 | И.А. Дулина |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Е.Ю. Гаврилова |
|  |  |  | (Подпись, дата) | (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

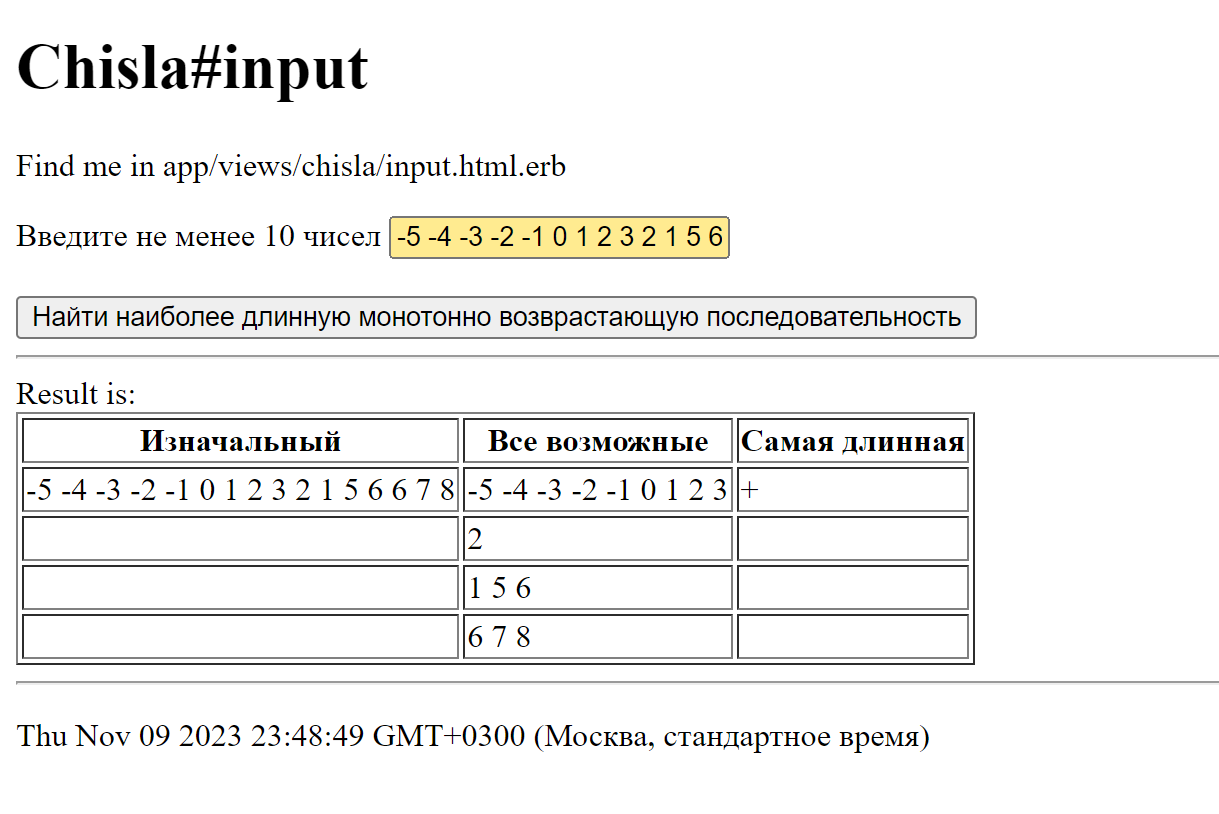


Рисунок 1 – страница с вводом и выводом результата

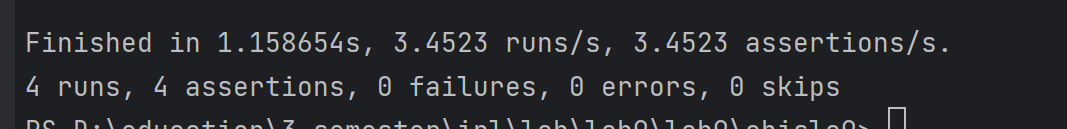


Рисунок 2 – результат автоматического тестирования ($ rake test)

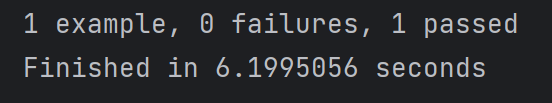


Рисунок 3 – результат теста, написанный при помощи Selenium IDE

**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.