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 (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА**

О т ч е т

по лабораторной работе № 12

Дисциплина: Языки Интернет-программирования

Название лабораторной работы: Сессии. Выполнение авторизации. Интеграционные тесты. Вариант 10.

Студент гр. ИУ6-34Б _____ К.Ю. Каташинский
(Подпись, дата) (И.О. Фамилия)

Преподаватель _____ _____
(Подпись, дата) (И.О. Фамилия)

Москва, 2020

Условие задания:

Модифицировать код приложения ЛР 8 таким образом, чтобы вычисление было невозможно без регистрации пользователя и аутентификации при помощи логина/пароля.

- *Сгенерировать при помощи генератора scaffold ресурс для регистрации пользователей.*
- *Создать БД и выполнить миграцию соответствующим запросом rake.*
- *Проверить возможность добавления, редактирования информации и получения списка пользователей.*
- *Удалить отображение поля пароля при просмотре списка пользователей.*
- *Добавить контроллер сессий.*
- *Реализовать форму для ввода логина/пароля при обращении по адресу /. Добавить ссылку на регистрацию нового пользователя. При успешном вводе логина/пароля должно осуществляться перенаправление на страницу ввода параметров для вычисления.*
- *Реализовать при помощи контроллера сессий во всех действиях контроллера проверку о того, прошел ли пользователь аутентификацию или нет (с выдачей соответствующей отладочной информации).*
- *Вставить фильтры для запроса аутентификации.*
- *Подготовить интеграционный тест, позволяющий проверить регистрацию нового пользователя, вход под его именем и выполнение вычислений.*
- *Подготовить интеграционный тест для проверки невозможности выполнения вычислений без ввода логина/пароля.*
- *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 авторизации (ограничить несколькими записями так, чтобы результат поместился на 1-2 страницах).*
- *Результаты выполнения интеграционных тестов.*

Код файла base_controller.rb:

```
require 'digest/sha1'
# BaseController class
class BaseController < ApplicationController
  def index
    unless @user = User.find_by_session_id(session[:session_id])
      redirect_to "/login"
      p "Пользователь не прошел аутентификацию"
      return
    end
    @result = LabResult.all
  end
  def create
    unless @user = User.find_by_session_id(session[:session_id])
      redirect_to "/login"
      p "Пользователь не прошел аутентификацию"
      return
    end
    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 show_results_in_xml
    unless @user = User.find_by_session_id(session[:session_id])
      redirect_to "/login"
      p "Пользователь не прошел аутентификацию"
      return
    end
    unless @user.name == "admin"
      redirect_to "/"
      return
    end
```

Продолжение кода файла base_controller.rb:

```
    @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 user_iteraction_controller.rb:

```
class UserInteractionController < ApplicationController
    def login
        if @user = User.find_by_session_id(session[:session_id])
          redirect_to "/"
          return
        end

        @error = (params[:error] == nil) ? '' : params[:error]
    end
    def logout
        unless @user = User.find_by_session_id(session[:session_id])
          redirect_to "/"
          return
        end
        @user.session_id = "0"
        @user.save
        redirect_to "/"
        return
    end
    def auth
        if @user = User.find_by_session_id(session[:session_id])
          redirect_to "/"
          return
        end
        name = params[:name]
        password = Digest::SHA1.hexdigest params[:password]
        error = ""
        if user = User.find_by_name(name)
          unless user.password == password
            error = "Введен неверный пароль"
          else
            user.session_id = session[:session_id]
            user.save
          end
        else
          error = 'Пользователь не найден'
        end
        if error == ""
          redirect_to "/"
          return
        else
          redirect_to :action => "login", :error => error
          return
        end
    end
    def register
        if @user = User.find_by_session_id(session[:session_id])
          redirect_to "/"
```

Продолжение кода файла user_iteraction_controller.rb:

```
          return
        end
        @error = (params[:error] == nil) ? '' : params[:error]
    end
    def create_new_user
        if @user = User.find_by_session_id(session[:session_id])
          redirect_to "/"
          return
        end
        name = params[:name]
        email = params[:email]
        password = params[:password]
        another_password = params[:another_password]
        if password != another_password
          redirect_to :action => "register", :error => 'Введенные пароли не
совпадают'
          return
        end
        error = ""
        if user = User.find_by_name(name)
          error = 'Пользователь с таким именем уже существует'
        end
        if user = User.find_by_email(email)
          error = 'Пользователь с такой почтой уже существует'
        end
        unless error == ''
          redirect_to :action => "register", :error => error
          return
        end
        password = Digest::SHA1.hexdigest password
        @user = User.new(name: name, email: email, password: password, session_id:
session[:session_id])
        unless @user.valid?
          redirect_to :action => "register", :error => 'Проверьте правильность
введенных данных'
          return
        end
        @user.save
        redirect_to "/"
    end
    def users_list
        unless @user = User.find_by_session_id(session[:session_id])
          redirect_to "/login"
          return
        end
        unless @user.name == "admin"
          redirect_to "/"
          return
        end
        @users = User.all
    end
    def show_users_in_xml
      @users = User.all
      render xml: @users
    end
end
```

Код файла user.rb:

```
class User < ApplicationRecord
    before_save { self.email = email.downcase }
    validates :name,  presence: true, length: { maximum: 50 }, uniqueness: true
    validates :password,  presence: true, length: { maximum: 50 }
    VALID_EMAIL_REGEX = /\A[\w+\-.]+@[a-z\d\-.]+\.[a-z]+\z/i
    validates :email, presence: true, length: { maximum: 255 }, format: { with:
VALID_EMAIL_REGEX }, uniqueness: true
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>
        <% if @user.name == "admin" %>
            <tr>
                <td><a href="/show_results_in_xml" style="font-size:
medium;">Показать результаты в XML</a></td>
                <td><a href="/show_users_in_xml" style="font-size: medium;">Показать
пользователей в XML</a></td>
            </tr>
        <% end %>
    </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>
        <% if @user.name == "admin" %>
            <tr>
            <td><a href="/show_results_in_xml" style="font-size: medium;">Показать
результаты в XML</a></td>
                <td><a href="/show_users_in_xml" style="font-size: medium;">Показать
пользователей в XML</a></td>
            </tr>
        <% end %>
    </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 application.html.erb:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Лабораторная работа</title>
    <%= csrf_meta_tags %>
    <%= stylesheet_link_tag    'application', media: 'all', 'data-turbolinks-track':
'reload' %>
    <%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
  </head>
  <body>
    <header style="font-size: 1.17em;">
      <% unless  @user == nil %>
        Добро пожаловать, <span style="color:blue"><%= @user.name %></span>!
        <a href="/">Главная страница</a>
        <% if @user.name == "admin" %>
          <a href="/users_list">Список пользователей</a>
        <% end %>
        <a href="/logout">Выйти</a>
      <% else %>
        <a href="/login">Войдите,</a> чтобы получить доступ к сайту
      <% end %>
              <hr>
    </header>
    <%= yield %>
  </body>
</html>
```

Код файла register.html.erb:

```
<%= form_tag("/create_new_user", :method => "post") do %>
    <table id="add_form">
    <caption><h1>Регистрация</h1></caption>
        <tr>
            <td width="40%">Введите имя</td>
            <td><%= text_field_tag :name, nil, class: "enter" %></td>
        </tr>
            <tr>
            <td>Введите email</td>
            <td><%= email_field_tag :email, nil, class: "enter" %></td>
        </tr>
        <tr>
            <td>Введите пароль</td>
            <td><%= password_field_tag :password, nil, class: "enter" %></td>
        </tr>
        <tr>
            <td>Повторите пароль</td>
            <td><%= password_field_tag :another_password, nil, class: "enter" %></td>
        </tr>
        <tr>
            <td><%= submit_tag "Зарегистрироваться", class: "btn" %></td>
            <td><span style="color: red;"><%= @error %></span></td>
        </tr>
        <tr style="font-style: italic;">
            <td><span style="font-size: medium;">Уже зарегистрированы?</span></td>
            <td><a href="/login" style="font-size: medium;">Войти</a></td>
        </tr>
    </table>
<% end %>
```

Код файла login.html.erb:

```
<%= form_tag("/auth", :method => "post") do %>
    <table id="add_form">
    <caption><h1>Вход</h1></caption>
        <tr>
            <td width="40%">Введите имя</td>
            <td><%= text_field_tag :name, nil, class: "enter" %></td>
        </tr>
        <tr>
            <td>Введите пароль</td>
            <td><%= password_field_tag :password, nil, class: "enter" %></td>
        </tr>
        <tr>
            <td><%= submit_tag "Войти", class: "btn" %></td>
            <td><span style="color: red;"><%= @error %></span></td>
        </tr>
        <tr style="font-style: italic; font-size: medium;">
            <td><span style="font-size: medium;">Еще не зарегистрированы?</span></td>
            <td><a href="/register" style="font-size:
medium;">Зарегистрироваться</a></td>
        </tr>
    </table>
<% end %>
```

Код файла users_list.html.erb:

```
<table id="add_table">
    <caption><h1>Список пользователей</h1></caption>
    <tr>
        <td class="col">Имя</td>
        <td class="col">Email</td>
    </tr>
    <% @users.each do |user| %>
        <tr>
            <td class="col"><%= user.name %></td>
            <td class="col"><%= user.email %></td>
        </tr>
    <% end %>
</table>
```

Код файла routes.rb:

```
Rails.application.routes.draw do
  root 'base#index'
  get "/show_users_in_xml" => "user_interaction#show_users_in_xml"
  get "/login" => "user_interaction#login"
  get "/logout" => "user_interaction#logout"
  get "/register" => "user_interaction#register"
  get "/users_list" => "user_interaction#users_list"
  post "/auth" => "user_interaction#auth"
  post "/create_new_user" => "user_interaction#create_new_user"
  get "/show_results_in_xml" => "base#show_results_in_xml"
  post "/create" => "base#create"
end
```

Код файла registration_test.rb:

```
require 'test_helper'
class RegistrationTest < ActionDispatch::IntegrationTest
  test "redirect from index" do
    post "/create", params: { number: 5, str: "1 2 3 4 5" }
    assert_response :redirect
  end
  test "can find a result from new authorized user" do
    post "/create_new_user", params: { name: "user", email: "user@mail.ru", password:
"123", another_password: "123" }
    assert_response :redirect
    get "/logout"
    assert_response :redirect
    post "/auth", params: { name: "user", password: "123"}
    assert_response :redirect
    post "/create", params: { number: 10, str: "1 5 5 25 7 8 10 5 125 10" }
    assert_equal assigns[:max_segment], "5 5 25"
  end
end
```

Код файла user_test.rb:

```
require 'test_helper'
class UserTest < ActiveSupport::TestCase
  def setup
    @user = User.new(name: "Example User", email: "user@example.com", password:
"12345", session_id: "c6116f7395b34412ad0aea3c9b8d4bfa")
  end
  test "should be valid" do
    assert @user.valid?
  end
  test "name should be present" do
    @user.name = ""
    assert_not @user.valid?
  end
  test "email should be present" do
    @user.email = ""
    assert_not @user.valid?
  end
  test "email validation should accept valid addresses" do
    valid_addresses = %w[user@example.com USER@foo.COM A_US-ER@foo.bar.org
                         first.last@foo.jp alice+bob@baz.cn]
    valid_addresses.each do |valid_address|
      @user.email = valid_address
      assert @user.valid?, "#{valid_address.inspect} should be valid"
    end
  end
  test "emails and logins should be unique" do
    duplicate_user = @user.dup
    @user.save
    assert_not duplicate_user.valid?
  end
  test "name should not be too long" do
    @user.name = "a" * 51, assert_not @user.valid?
  end
  test "email should not be too long" do
    @user.email = "a" * 244 + "@example.com", assert_not @user.valid?
  end

end
```

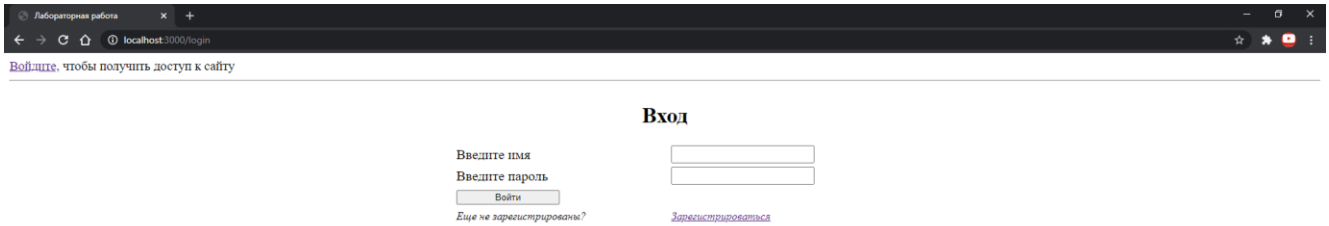
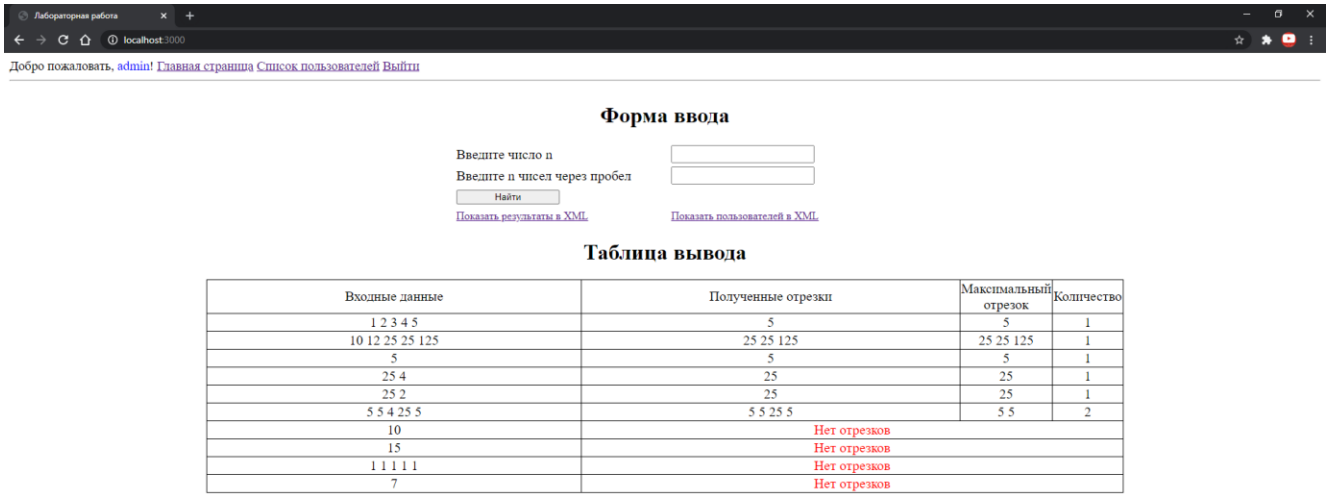


Рисунок 1 «Страница входа»



Входные данные	Полученные отрезки	Максимальный отрезок	Колличество
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 2 «Главная страница»

Рисунок 3 «XML-распечатка содержимого БД авторизации»

```
Run options: --seed 46133

# Running:

"Пользователь не прошел аутентификацию"
............

Finished in 1.458023s, 8.9162 runs/s, 13.7172 assertions/s.
13 runs, 20 assertions, 0 failures, 0 errors, 0 skips
```

Рисунок 4 «Результаты выполнения интеграционных тестов»

Вывод

В ходе данной лабораторной работы были закреплены основные навыки построения ror-приложения. Была создана система регистрации и авторизации пользователей с помощью механизма сессий.