| 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 **09.03.01 Информатика и вычислительная техника**

**ОТЧЕТ**

| **по лабораторной работе №** | 11 |
| --- | --- |



## Сессии. Выполнение авторизации.

## Интеграционные тесты.

**Дисциплина:** Языки Интернет-программирования

| Студент | ИУ6-33Б |  | 12.12.2021 | Хоанг Т.Ф. |
| --- | --- | --- | --- | --- |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  |  |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

## ЛР 12. Сессии. Выполнение авторизации. Интеграционные тесты.

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 /.

**routes.rb**

Rails.application.routes.draw do

root 'sessions#new', as: 'new\_session'

resources :users

controller :sessions do

get 'login' => :new, as: 'login'

post 'login' => :create, as: 'login\_p'

delete 'logout' => :destroy, as: 'logout'

end

get '/friend\_num' => 'friend\_num#input', as: 'friend\_num\_inp'

post '/result' => 'friend\_num#result', as: 'friend\_num'

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

end

**MODELS:**

**user.rb**

class User < ApplicationRecord

has\_secure\_password

validates :email, presence: true, uniqueness: true

end

**CONTROLLERS:**

**application\_controller.rb**

class ApplicationController < ActionController::Base

helper\_method :current\_user

def current\_user

if session[:user\_id]

@current\_user ||= User.find(session[:user\_id])

else

@current\_user = nil

end

end

end

**users\_controllers.rb**

class UsersController < ApplicationController

before\_action :set\_user, only: %i[ show edit update destroy ]

# GET /users or /users.json

def index

@users = User.all

end

# GET /users/1 or /users/1.json

def show

end

# GET /users/new

def new

@user = User.new

end

# GET /users/1/edit

def edit

end

# POST /users or /users.json

def create

@user = User.new(user\_params)

respond\_to do |format|

if @user.save

format.html { redirect\_to @user, notice: "User was successfully created." }

format.json { render :show, status: :created, location: @user }

else

format.html { render :new, status: :unprocessable\_entity }

format.json { render json: @user.errors, status: :unprocessable\_entity }

end

end

end

# PATCH/PUT /users/1 or /users/1.json

def update

respond\_to do |format|

if @user.update(user\_params)

format.html { redirect\_to @user, notice: "User was successfully updated." }

format.json { render :show, status: :ok, location: @user }

else

format.html { render :edit, status: :unprocessable\_entity }

format.json { render json: @user.errors, status: :unprocessable\_entity }

end

end

end

# DELETE /users/1 or /users/1.json

def destroy

@user.destroy

respond\_to do |format|

format.html { redirect\_to users\_url, notice: "User was successfully destroyed." }

format.json { head :no\_content }

end

end

private

# Use callbacks to share common setup or constraints between actions.

def set\_user

@user = User.find(params[:id])

end

# Only allow a list of trusted parameters through.

def user\_params

params.require(:user).permit(:username, :email, :password, :password\_confirmation)

end

end

**friend\_num\_controller.rb**

class FriendNumController < ApplicationController

attr\_accessor :res, :num

def input; end

def sum\_divisors(arg)

sum = 0

(1...arg).each do |div|

sum += div if arg % div == 0

end

sum

end

def check

arr = Array.new(0)

sum = Array.new(num)

(1..num).each {|index| sum[index] = sum\_divisors(index)}

(1..num - 1).each do |num1|

(num1..num).each do |num2|

arr.append([num1, num2]) if num1 != num2 && num1 == sum[num2] && sum[num1] == num2

end

end

arr

end

def result

@num = params[:num].to\_i

@res = check

respond\_to do |format|

format.html { redirect\_to :root }

format.js

end

end

end

**sessions\_controller.rb**

class SessionsController < ApplicationController

def new

session[:user\_id] = nil;

end

def create

user = User.find\_by(email: params[:email])

if user&.authenticate(params[:password])

session[:user\_id] = user.id

session[:user\_name] = user.username

redirect\_to friend\_num\_inp\_path, notice: 'User logged in'

else

redirect\_to '/login', notice: 'Invalid Email or Password'

end

end

def destroy

session[:users\_id] = nil

redirect\_to '/login', notice: 'User logged out'

end

end

**VIEWS:**

**friend\_num/**

**input.html.erb**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<title>Friend Number</title>

</head>

<body>

<% if current\_user %>

<h1>FRIEND NUMBER - Logged in as <%=current\_user.username %></h1>

<h3>Two numbers are called friend if each of them is equal to the sum of all divisors of another, except for this number itself. </h3>

<%= form\_with(url: friend\_num\_path, method: "post", local: false) do |form|%>

<%= form.label :num, "Enter n: " %>

<%= form.text\_field :num %>

<div class="button">

<%= form.submit "Get result" %>

</div>

<% end %>

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

<script>

const resultDiv = document.querySelector("#result");

</script>

<%= link\_to 'Logout', logout\_path, method: :delete %>

<% else %>

<h3>You need to log in to access to this page </h3>

<%= link\_to "Sign Up", new\_user\_path %> or

<%= link\_to "Log In", login\_path %>

<% end %>

</body>

</html>

**\_result.html.erb**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<title>Result</title>

</head>

<body>

<h1>RESULT TABLE - Friend numbers</h1>

<table class="table table-bordered">

<tr>

<th>The first number</th>

<th>The second number</th>

</tr>

<% @res.each do |couple| %>

<tr>

<td><%= couple[0] %> </td>

<td><%= couple[1] %> </td>

</tr>

<% end %>

</table>

</body>

</html>

**result.js.erb**

resultDiv.innerHTML = '<%= escape\_javascript(render(partial: 'friend\_num/result')) %>';

console.log("RENDERED");

**sessions/new.html.erb**

<h1>LOG IN</h1>

<div>

<p id="notice"><%= notice %></p>

<%= form\_tag '/login' do %>

<fieldset>

<legend>Please Log In</legend>

<div>

<%= label\_tag :email, 'Email' %>

<%= text\_field\_tag :email, params[:email] %>

<%= label\_tag :password, 'Password' %>

<%= password\_field\_tag :password, params[:password] %>

</div>

<div>

<%= submit\_tag "Login" %>

</div>

</fieldset>

<% end %>

</div>

<%= link\_to 'Sign up', new\_user\_path %>

**DB:**

**schema.rb**

ActiveRecord::Schema.define(version: 2021\_12\_13\_131444) do

create\_table "users", force: :cascade do |t|

t.string "username"

t.string "email"

t.string "password\_digest"

t.datetime "created\_at", precision: 6, null: false

t.datetime "updated\_at", precision: 6, null: false

t.index ["email"], name: "index\_users\_on\_email", unique: true

end

end

**TEST:**

**users\_controller\_test.rb**

require "test\_helper"

class UsersControllerTest < ActionDispatch::IntegrationTest

setup do

@user = users(:one)

end

test "should get index" do

get users\_url

assert\_response :success

end

test "should get new" do

get new\_user\_url

assert\_response :success

end

test "should create user" do

assert\_difference('User.count') do

post users\_url, params: { user: { email: 'email', password: 'secret', password\_confirmation: 'secret', username: @user.username } }

end

assert\_redirected\_to user\_url(User.last)

end

test "should show user" do

get user\_url(@user)

assert\_response :success

end

test "should get edit" do

get edit\_user\_url(@user)

assert\_response :success

end

test "should update user" do

patch user\_url(@user), params: { user: { email: @user.email, password: 'secret', password\_confirmation: 'secret', username: @user.username } }

assert\_redirected\_to user\_url(@user)

end

test "should destroy user" do

assert\_difference('User.count', -1) do

delete user\_url(@user)

end

assert\_redirected\_to users\_url

end

end

**sessions\_controller\_test.rb**

require "test\_helper"

class SessionsControllerTest < ActionDispatch::IntegrationTest

test "should get new" do

get new\_session\_url

assert\_response :success

end

test "should get create" do

post login\_p\_url

assert\_redirected\_to 'http://www.example.com/login'

end

test "should get destroy" do

delete logout\_url

assert\_redirected\_to 'http://www.example.com/login'

end

end

**user\_test.rb**

require "test\_helper"

class UserTest < ActiveSupport::TestCase

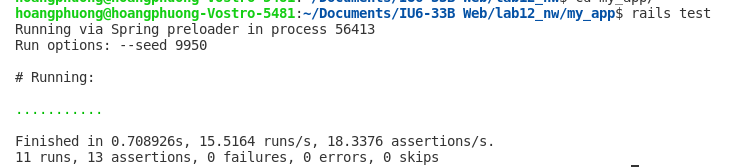
test 'input user' do

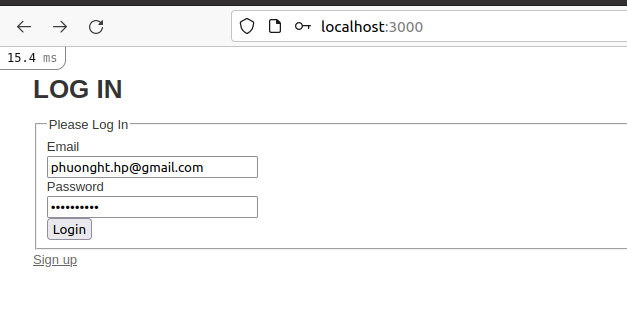
user = User.new(username: 'a', email: 'a', password: 'a')

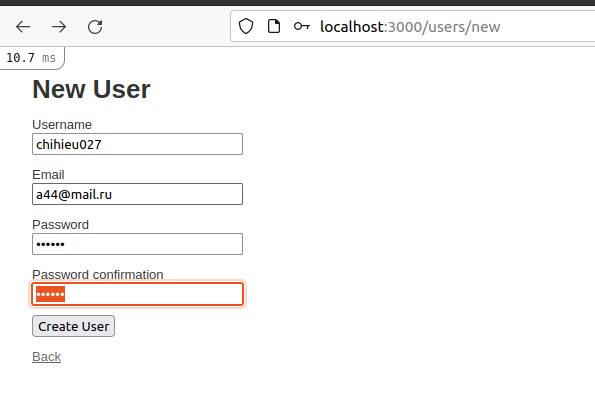
assert user.valid?

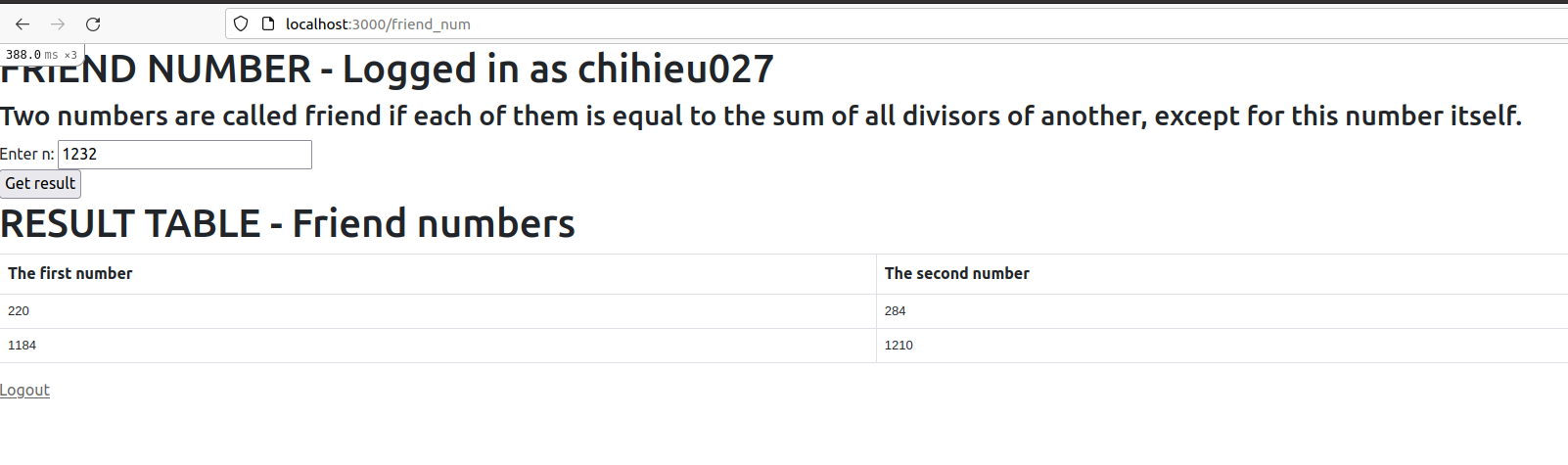
end

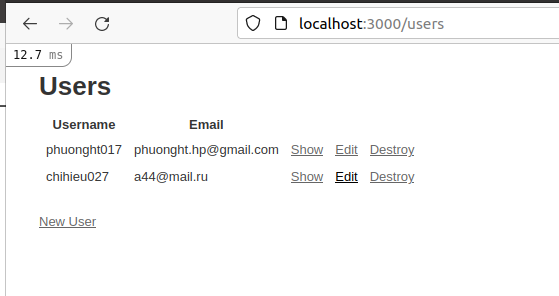
end

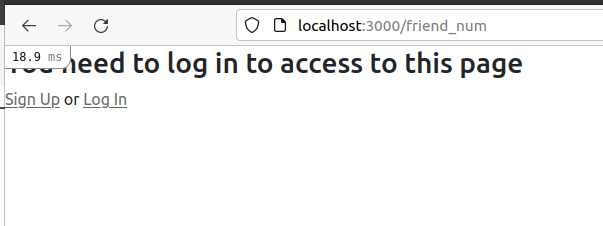
****

****

****

****

****

****