

Вариант 8

Цель работы: получение практических навыков в создании веб-приложений, использующих аутентификацию. Получить навыки написания интеграционных тестов приложений.

Задание:

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 /.

В консоль:

```
$rails new chisla12
```

```
$cd chisla12
```

```
$rails generate controller Chisla input view
```

```
$rails generate scaffold User name:string email:string:uniq password:text
```

```
$rake db:create
```

```
$rake db:migrate
```

```
$rake db:migrate RAILS_ENV=test
```

Сгенерируем контроллер Sessions и интеграционный тест для механизма аутентификации:

```
rails generate controller Sessions --no-test-framework
```

```
rails generate integration_test authentication_pages
```

в *Gemfile* добавить:

```
gem "jquery-rails"
```

```
gem "bootstrap"
```

```
gem "sassc-rails"
```

в *config/initializers/assets.rb* добавить

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

В *app/assets/stylesheets/application.scss* добавить

```
@import "bootstrap";
```

В *config/importmap.rb* добавить

```
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
```

```
pin "popper", to: "popper.js", preload: true
```

```
pin "bootstrap", to: "bootstrap.min.js", preload: true
```

В *app/javascript/application.js* добавить

```
import "jquery"
```

```
import "jquery_ujs"
```

```
import "popper"
```

```
import "bootstrap"
```

```
import "src/main"
```

Проверка возможности добавления информации:

```
insert into users(name, email, password, created_at, updated_at)
```

```
values ('example', 'ex@ya.ru', '123456', CURRENT_TIMESTAMP,  
CURRENT_TIMESTAMP);
```

	id	name	email	password	created_at	updated_at
1	2	ivanova1	ferrari@gmail.com	popanc	2023-11-26 23:55:57.667659	2023-11-26 23:55:57.667659
2	5	itgirl	ivanova@ya.ru	123	2023-11-27 12:24:01.226605	2023-11-27 12:24:01.226605
3	6	itgirl	tr@gmail.com	popanc	2023-11-27 13:15:55.933746	2023-11-27 13:15:55.933746
4	7	admin	admin@gmail.com	popanc	2023-11-27 14:30:54.153318	2023-11-27 14:30:54.153318
5	8	example	ex@ya.ru	123456	2023-11-27 15:48:12	2023-11-27 15:48:12

Рисунок 1 – добавление информации в БД Users

Проверка редактирования информации:

```
update users set password='123' where email='ivanova@ya.ru';
```

	id	name	email	password	created_at	updated_at
1	2	ivanova1	ferrari@gmail.com	popanc	2023-11-26 23:55:57.667659	2023-11-26 23:55:57.667659
2	5	itgirl	ivanova@ya.ru	123	2023-11-27 12:24:01.226605	2023-11-27 12:24:01.226605
3	6	itgirl	tr@gmail.com	popanc	2023-11-27 13:15:55.933746	2023-11-27 13:15:55.933746
4	7	admin	admin@gmail.com	popanc	2023-11-27 14:30:54.153318	2023-11-27 14:30:54.153318

Рисунок 2 – изменение данных в БД Users

Проверка получения списка пользователей:

```
Select * from users;
```

	id	name	email	password	created_at	updated_at
1	2	ivanova1	ferrari@gmail.com	popanc	2023-11-26 23:55:...	2023-11-26 23:55:...
2	5	itgirl	ivanova@ya.ru	popanc	2023-11-27 12:24:...	2023-11-27 12:24:...
3	6	itgirl	tr@gmail.com	popanc	2023-11-27 13:15:...	2023-11-27 13:15:...
4	7	admin	admin@gmail.com	popanc	2023-11-27 14:30:...	2023-11-27 14:30:...

Рисунок 3 – вывод всех данных из БД Users

Контроллеры:

```
app/controllers/chisla_controller.rb
```

```
class ChislaController < ApplicationController
```

```
def input
```

```
unless signed_in?  
  redirect_to signin_path  
end  
end
```

```
def view  
  unless signed_in?  
    redirect_to signin_path  
  end  
  if params[:str]  
    begin  
      res = params[:str].scan(/-?\d+(?:\.\d+)?/).map(&:to_i)  
      raise StandardError if res.length < 10  
      @result = check(res)  
    rescue StandardError  
      @result = [{}, 'Что-то пошло не так']  
    end  
  else  
    @result = [{}, 'Unknown!']  
  end  
end
```

```
def check(res)  
  returning = create(res)  
  sol = returning[0]  
  everything = returning[1]  
  [sol, create_table(everything)]  
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=create_massive_for_table(res, all, solution)
[solution, 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]
  end
end

```

```

    break if i + 1 == res.length
    break if (res[i+1] <= res[i])
    i += 1
  end
  i+=1
  [pos1, len, i]
end

```

#на вход изначальный массив, все возможные последовательности, самая длинная из них

```

def create_massive_for_table(res, all, solution)
  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
  result
end
end

```

app/controllers/application_controller.rb

```

class ApplicationController < ActionController::Base
  protect_from_forgery with: :exception
end

```

```

include SessionsHelper

before_action :authenticate, :except => [:signup, :signin, :new, :create]

private

#если идентификатор пользователя есть (session[:current_user_id]), то
записываем в _current_user

#если его нет, то сделаем выборку из базы (User.find_by id(...))

def current_user
  @_current_user ||= session[:current_user_id] &&
    User.find_by_id(session[:current_user_id])
end

def authenticate
  unless current_user
    redirect_to signin_path
  end
end

end

app/controllers/sessions_controller.rb

class SessionsController < ApplicationController
  skip_before_action :authenticate, only: [:new, :create]

  def new
  end

  def create
    msg_text = "
    msg_status = :success

    email = params[:session][:email]
    password = params[:session][:password]

```



```

puts password

respond_to do |format|
  user = User.find_by(email: email.downcase)

  if !user
    msg_text = 'Пользователя не существует'
    msg_status = :danger
  elsif
user!=User.authenticate(params[:session][:email],params[:session][:password])
    msg_text = 'Неверный пароль'
    msg_status = :danger
  end

  if msg_status == :success
    sign_in user
    msg_text = 'Вы успешно вошли'
    flash[msg_status] = msg_text
    format.html { redirect_to input_path }
    format.json { render :show, status: :created, location: input_path }
  else
    flash.now[msg_status] = msg_text
    format.html { render :new, status: :unprocessable_entity }
    format.json { render json: @user.errors, status: :unprocessable_entity }
  end
end

#user
User.authenticate(params[:session][:email],params[:session][:password])

end

#убрать пользователя из сессии

```

=

```

def destroy
  sign_out
  redirect_to root_url
end
end

```

app/controllers/users_controller.rb

```
require 'nokogiri'
```

```

class UsersController < ApplicationController
  before_action :set_user, only: %i[ show edit update destroy ]
  XSLT_TRANSFORM = "#{Rails.root}/public/some_transformer.xslt".freeze #

```

Путь до xslt файла

```
# GET /users or /users.json
```

```
def index
```

```
  @users = User.all
```

```
  @user_name = current_user ? current_user.name : "unknown"
```

```
end
```

```
# для вывод БД в XML
```

Добавить действие в контроллер, позволяющее определить, что хранится в БД через сериализацию в XML.

```
# http://127.0.0.1:3000/show_all.xml
```

```
def show_all
```

```
  respond_to do |format|
```

```
    results = User.all
```

```
    rows = "
```

```
    results.each do |record|
```

```
      rows
```

+=

```

"<cd><name>#{record.name}</name><email>#{record.email}</email><pass>#{record.password}</pass></cd>"

```

```

    end
    response = "<catalog>#{rows}</catalog>"
    format.xml { render xml: xslt_transform(response).to_xml }
  end
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
  msg_text = "
  msg_status = :success

  email = params[:user][:email]
  @user = User.new(user_params)

  respond_to do |format|
    if @user
      if User.find_by_email(email)
        msg_text = 'Пользователь уже зарегистрирован!'

```

```

        msg_status = :danger
      elsif !email.match?('[a-z0-9]+[_a-z0-9\.-]*[a-z0-9]+@[a-z0-9-]+\.[a-z0-9-
    ]+)*(\.[a-z]{2,4})')
        msg_text = 'Введите почту корректно'
        msg_status = :danger
      end

    if msg_status == :success and @user.save
      sign_in @user
      msg_text = 'Спасибо за регистрацию'
      flash[msg_status] = msg_text

      format.html { redirect_to input_path }
      format.json { render :show, status: :created, location: input_path }
    else
      flash.now[msg_status] = msg_text
      format.html { render :new, status: :unprocessable_entity }
      format.json { render json: @user.errors, status: :unprocessable_entity }
    end
  end
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_url(@user), notice: "User was successfully
updated." }
      format.json { render :show, status: :ok, location: @user }
    end
  end
end

```

```

    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(:name, :email, :password)
end

```

```

def xslt_transform(data, transform: XSLT_TRANSFORM)
  # Функция преобразования
  pp 'checkpoint2'
  print data, transform, "\n"
  doc = Nokogiri::XML(data)
  xslt = Nokogiri::XSLT(File.read(transform))
  xslt.transform(doc)
end

end

public/some_transformer.xslt
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet                                version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
    <xsl:if test="output/input">
      <div><xsl:value-of select="output/input"/></div>
    </xsl:if>

    <table border="1">
      <tr bgcolor="#9933ff">
        <th>User name</th>
        <th>Login</th>
        <th>Password</th>
      </tr>
      <xsl:for-each select="catalog/cd">
        <tr>

```

```

        <td><xsl:value-of select="name"/></td>
        <td><xsl:value-of select="email"/></td>
        <td><xsl:value-of select="pass"/></td>
    </tr>
</xsl:for-each>
</table>
</xsl:template>
</xsl:stylesheet>

```

app/helpers/sessions_helper.rb **добавить**

```

def signed_in?
  !current_user.nil?
end

def current_user
  @_current_user ||= session[:current_user_id] &&
    User.find_by_id(session[:current_user_id])
end

```

app/javascript/src/main.js (для вывода ошибки)

```

$(document).on('click', '.btn-close', function () {
  $('#alert').fadeOut();
});

```

app/models/users.rb

```

class User < ApplicationRecord
  def has_password?(submitted_password)
    password == submitted_password
  end

  def self.authenticate(email, submitted_password)
    user = find_by_email(email)
    return nil if user.nil?
    return user if user.has_password?(submitted_password)
  end
end

```

end

Представления:

app/views/chisla/input.html.erb

<h1>Chisla#input</h1>

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

<div>

<form action="**<%= view_path %>**" method="get" accept-charset="UTF-8">

<div>

<label for="str">Введите не менее 10 чисел

<input type="text" id="str" name="str" value = "1 2 3 1 2 3 4 5 1 2 3 1 2 3 4"

required/>

</label>

</div>

<div>

<input type="submit" value="Найти наиболее длинную монотонно
возрастающую последовательность"/>

</div>

</form>

</div>

app/views/chisla/view.html.erb

<h1>Chisla#view</h1>

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

<p>Таблица результатов</p>

<%= @result[1].html_safe %>

<%= link_to "Найти для других чисел", **input_path %>**

app/views/layouts/_header.html.erb


```

<header>
  <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
    <div class="container-fluid">
      <a class="navbar-brand" href="#">Лаб.работа 12</a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
bs-target="#navbarNav">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav me-auto mb-2 mb-lg-0 ">
          <% if signed_in? %>
            <li class="nav-item">
              <%= link_to 'Ввод', input_path, class: "nav-link #{ request.path ==
input_path || request.path == '/' ? 'active' : " }" %>
            </li>
            <li class="nav-item">
              <%= link_to 'Вывод', view_path, class: "nav-link #{ request.path ==
view_path ? 'active' : " }" %>
            </li>
            <li class="nav-item">
              <%= link_to 'Выход', signout_path, method: "delete", class: "nav-link #{
request.path == signout_path ? 'active' : " }" %>
            </li>

          <% else %>
            <li class="nav-item navbar-right">
              <%= link_to 'Вход', signin_path, class: "nav-link #{ request.path ==
signin_path ? 'active' : " }" %>
            <% end %>
          </li>
        </ul>
      </div>
    </div>
  </nav>

```

```

    </ul>

    <ul class="navbar-nav ms-auto mb-2 mb-lg-0 ">
      <% if signed_in? %>
        <a class="navbar-brand float-right" href="#">Вы вошли как <%=
current_user.name %></a>
      <% end %>
    </ul>
  </div>
</div>
</nav>
</header>
app/views/layouts/_shim.html.erb
<!--[if lt IE 9]>
<script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
<![endif]-->
app/views/layouts/application.html.erb
<!DOCTYPE html>
<html>
  <head>
    <title>Chisla12</title>
    <meta name="viewport" content="width=device-width,initial-scale=1">
    <%= csrf_meta_tags %>
    <%= csp_meta_tag %>

    <%= stylesheet_link_tag "application", "data-turbo-track": "reload" %>
    <%= javascript_importmap_tags %>

    <%= stylesheet_link_tag "application", media: "all", "data-turbo-links-track"
=> true %>

```

```

    <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
    <%= csrf_meta_tags %>
    <%= render 'layouts/shim' %>
  </head>

  <body>
    <%= render 'layouts/header' %>
    <div class="container">
      <% flash.each do |key, value| %>
        <div class="alert alert-<%= key %> alert-dismissible fade show"
role="alert"><%= value %>
          <button id="flash-close" type="button" class="btn-close" data-
dismiss="alert">
            </button>
          </div>
        <% end %>
      <%= yield %>
    </div>
  </body>
</html>

```

app/views/sessions/new.html.erb

```

<h1>Sign in</h1>

<%= form_for(:session, :url => sessions_path) do |f| %>
  <div class="field">
    <%= f.label :email %><br />
    <%= f.text_field :email %>
  </div>
  <div class="field">
    <%= f.label :password %><br />

```

```

    <%= f.password_field :password %>
  </div>

  <div class="actions">
    <%= f.submit "Sign in" %>
  </div>
<% end %>

<p>New user? <%= link_to "Sign up now!", signup_path %></p>
app/views/users/_form.html.erb
<%= form_with(model: user) do |form| %>
  <% if user.errors.any? %>
    <div style="color: red">
      <h2><%= pluralize(user.errors.count, "error") %> prohibited this user from
being saved:</h2>

      <ul>
        <% user.errors.each do |error| %>
          <li><%= error.full_message %></li>
        <% end %>
      </ul>
    </div>
  <% end %>

  <div>
    <%= form.label :name, style: "display: block" %>
    <%= form.text_field :name, required:true %>
  </div>

  <div>
    <%= form.label :email, style: "display: block" %>

```

```

    <%= form.text_field :email,required:true %>
  </div>

  <div>
    <%= form.label :password, style: "display: block" %>
    <%= form.password_field :password, required:true %>
  </div>

  <div>
    <%= form.submit %>
  </div>
<% end %>

```

app/views/users/_user.html.erb

```

<div id="<%= dom_id user %>">
  <p>
    <strong>Name:</strong>
    <%= user.name %>
  </p>

  <p>
    <strong>Email:</strong>
    <%= user.email %>
  </p>

  <!-- <p>-->
  <!-- <strong>Password:</strong>-->
    <%#= user.password %>
  <!-- </p>-->

</div>

```

app/views/users/edit.html.erb

<h1>Editing user</h1>

<%= render "form", user: @user %>

<div>

<%= link_to "Show this user", @user %> |

<%= link_to "Back to users", users_path %>

</div>

app/views/users/index.html.erb

<p style="color: green"><%= notice %></p>

<h1>Users</h1>

<div id="users">

<% @users.each do |user| %>

<%= render user %>

<p>

<%= link_to "Show this user", user %>

</p>

<% end %>

</div>

<%= link_to "New user", new_user_path %>

app/views/users/new.html.erb

<h1>New user</h1>

<%= render "form", user: @user %>

```
<br>
```

```
<div>
```

```
<%= link_to "Back to users", users_path %>
```

```
</div>
```

```
app/views/users/show.html.erb
```

```
<p style="color: green"><%= notice %></p>
```

```
<%= render @user %>
```

```
<div>
```

```
<%= link_to "Edit this user", edit_user_path(@user) %> |
```

```
<%= link_to "Back to users", users_path %>
```

```
<% if current_user.email == 'admin@gmail.com' %>
```

```
<%= button_to "Destroy this user", @user, method: :delete %>
```

```
<% end %>
```

```
</div>
```

Пути

```
config/routes.rb
```

```
Rails.application.routes.draw do
```

```
  resources :users
```

```
  resources :sessions, :only => [:new, :create, :destroy]
```

```
  match '/input', to: 'chisla#input', via: 'get'
```

```
  match '/view', to: 'chisla#view', via: 'get'
```

```
  root 'chisla#input'
```

```
  get 'show_all', to: 'users#show_all'
```

```
  match '/signup', to: 'users#new', via: 'get'
```

```
  match '/signin', to: 'sessions#new', via: 'get'
```

```
  match '/signout', to: 'sessions#destroy', via: 'delete'
```

end

Интеграционные тесты

test/integration/authentication_pages_test.rb

require "test_helper"

#rake test:integration

#rake test TEST=test/integration/authentication_pages_test.rb

class AuthenticationPagesTest < ActionDispatch::IntegrationTest

def add_record(name, email, password)

record = User.new(:name => name, :email => email, :password => password)

record.save

record

end

Подготовить интеграционный тест,

позволяющий проверить регистрацию нового пользователя,

вход под его именем и выполнение вычислений

#####

Sign

up

#####

Проверяем доступность страницы регистрации

test "test registration page access" do

get signup_url

assert_response :success

end

Проверяем, что нельзя зарегистрировать того же пользователя

test 'attempt to register with existing user details' do

Создаем пользователя

add_record('test', 'test@test.com', '123456')

get signup_url


```

    assert_response :success

    post users_url, params: { "user" => { "name"=>'test', "email" =>
"test@test.com", "password" => "123456"} }

    assert_response 422
  end

  # Проверяем, что пользователя можно зарегистрировать
  test 'successfully user registration' do
    get signup_url
    assert_response :success

    # Смотрим, что такой пользователь только 1
    assert_difference 'User.count', 1 do
      post users_url, params: { "user" => { "name"=>'test', "email" =>
"test@test.com", "password" => "123456"} }
      follow_redirect!
    end

    assert_template 'input'
    assert_response 200
  end

  ##### Sign in
  #####

  # Проверяем доступность страницы входа
  test "test login page access" do
    get signin_url
    assert_response :success
  end
end

```

```

test 'successfully user login' do
  add_record('test', 'test@test.com', '123456')

  assert_difference 'User.count', 0 do
    post sessions_url, params: { "session" => { "name"=>'test', "email" =>
"test@test.com", "password" => "123456"} }
    follow_redirect!
  end

  assert_template 'input'
  assert_response 200
end

test 'login of a non-existent user' do
  post sessions_url, params: { "session" => { "name"=>'test', "email" =>
"test@test.com", "password" => "123456" } }

  assert_template 'sessions/new'
  assert_response 422
end

test 'login without password' do
  post sessions_url, params: { "session" => {"name"=>'test', "email" =>
"test@test.com", "password" => "" } }

  assert_template 'sessions/new'
  assert_response 422
end

```

```
#####
#####
test "test logout success" do
  # Добавляем тестового юзера в БД
  add_record('test', 'test@test.com', '123456')

  assert_difference 'User.count', 0 do
    # login
    post sessions_url, params: { "session" => {"name"=>'test', "email" =>
"test@test.com", "password" => "123456" } }
    follow_redirect!
  end

  assert_difference 'User.count', 0 do
    # Logout
    delete signout_url
    follow_redirect! # перенаправлены в input, там проверка, что не
залогились и иедм в login
    follow_redirect! # из input в login
  end

  assert_template 'sessions/new'
  assert_response 200
end

#Подготовить интеграционный тест для проверки невозможности
выполнения вычислений без ввода логина/пароля
test "Calculations are impossible without sign in" do
  # view
  get view_url, params: { str: '3 5 5 4 2 6 7 8 4 3' }
  # Если не вошли, значит редиректимся в signin

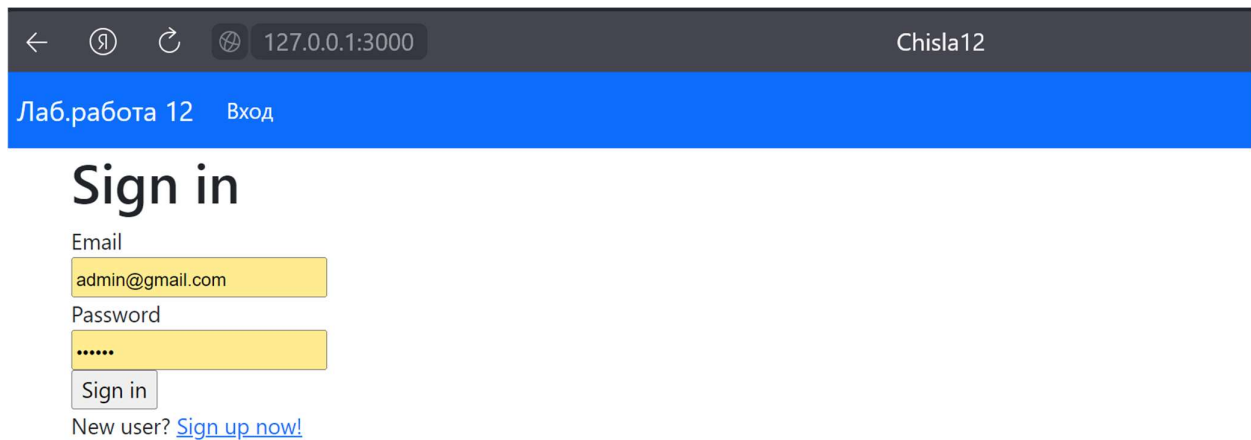
```

```
assert_response 302

# input
get input_url
assert_response 302

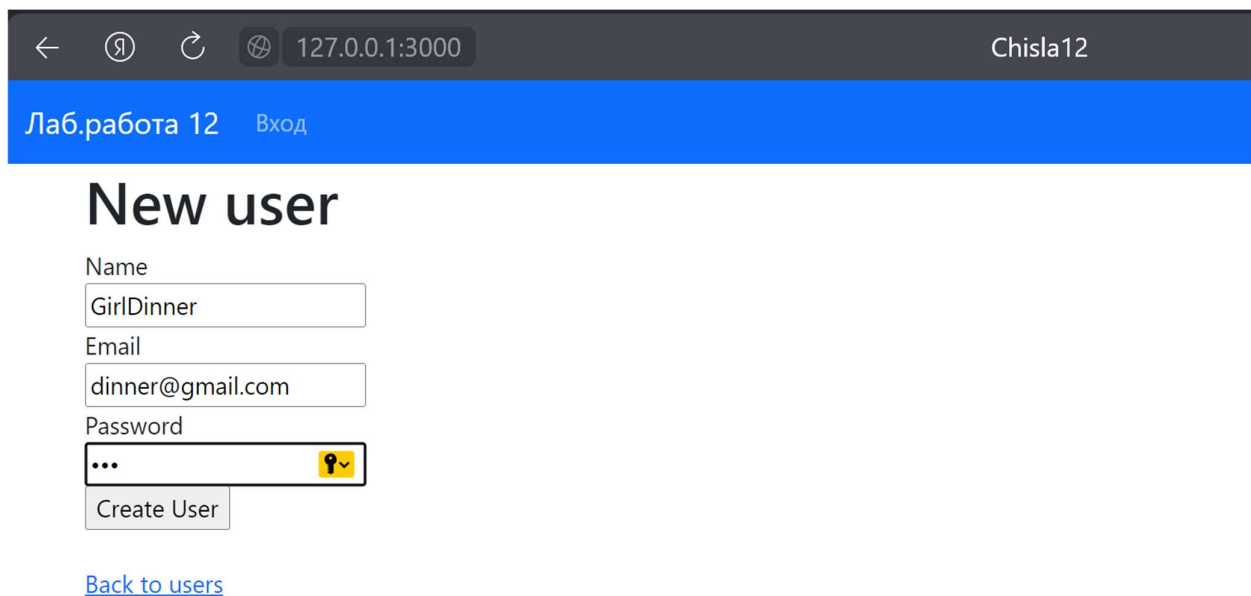
end

end
```



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:3000' and the page title 'Chisla12'. The browser's address bar also shows '←', '↻', and '🌐' icons. The page has a blue header bar with the text 'Лаб. работа 12' and 'Вход'. Below the header, the main heading is 'Sign in'. There are two input fields: 'Email' with the value 'admin@gmail.com' and 'Password' with masked characters '.....'. A 'Sign in' button is located below the password field. At the bottom, there is a link 'New user? [Sign up now!](#)'.

Рисунок 4 – форма входа



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:3000' and the page title 'Chisla12'. The browser's address bar also shows '←', '↻', and '🌐' icons. The page has a blue header bar with the text 'Лаб. работа 12' and 'Вход'. Below the header, the main heading is 'New user'. There are three input fields: 'Name' with the value 'GirlDinner', 'Email' with the value 'dinner@gmail.com', and 'Password' with masked characters '...' and a yellow eye icon. A 'Create User' button is located below the password field. At the bottom, there is a link '[Back to users](#)'.

Рисунок 5 – форма регистрации

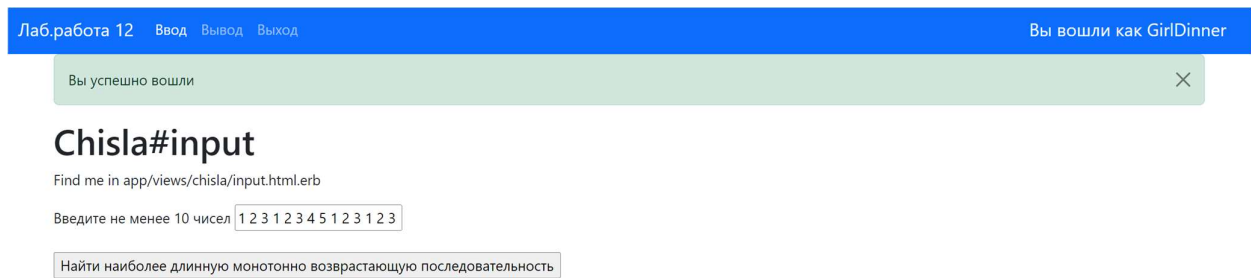


Рисунок 6 – успешный вход

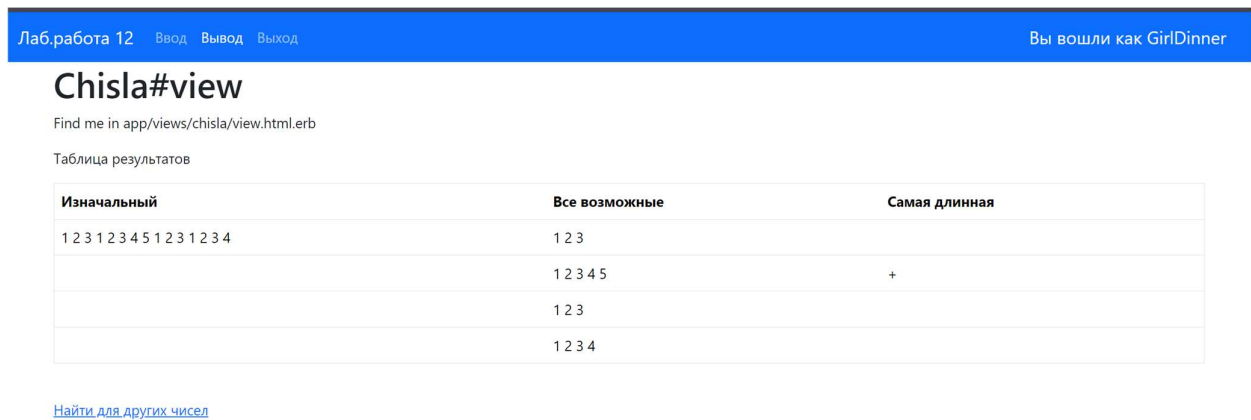


Рисунок 7 – вывод значения в результате успешного входа и ввода данных

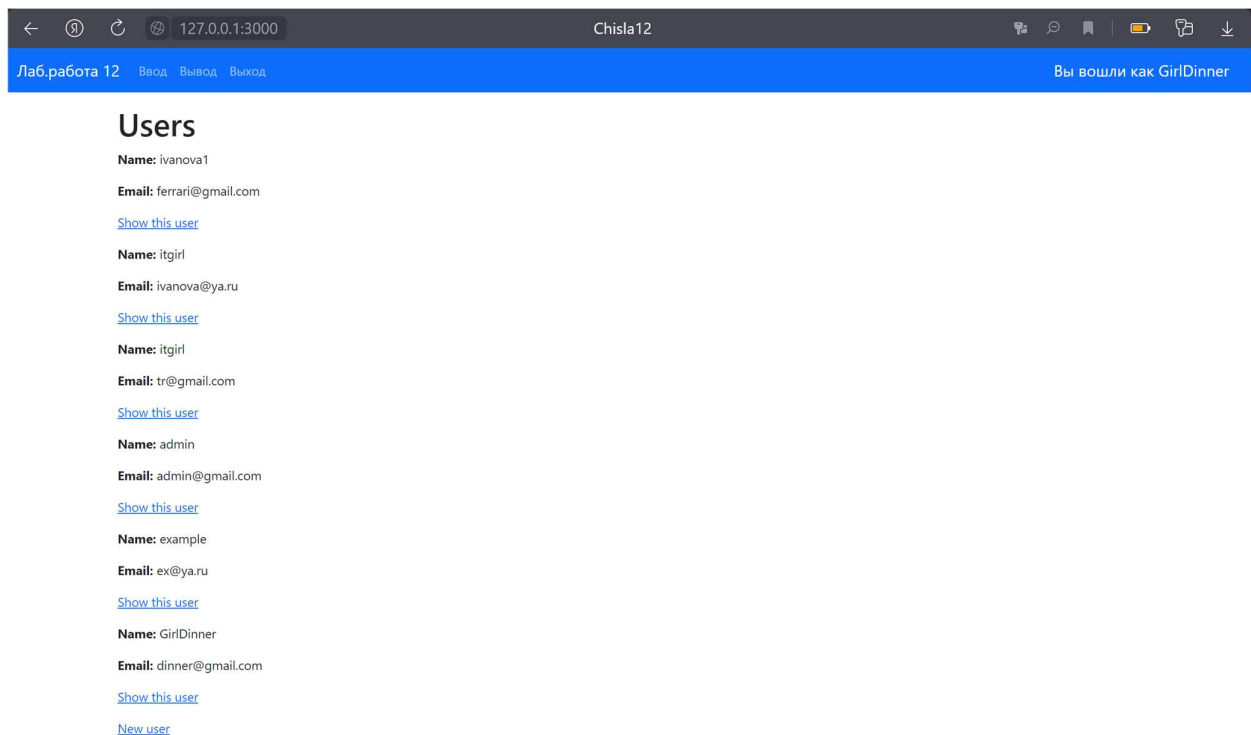


Рисунок 8 – вывод всех пользователей (/users)



Рисунок 9 – попытка удалить пользователя с аккаунта, не являющимся админом



Рисунок 10 – попытка удалить пользователя с аккаунта, являющегося админом

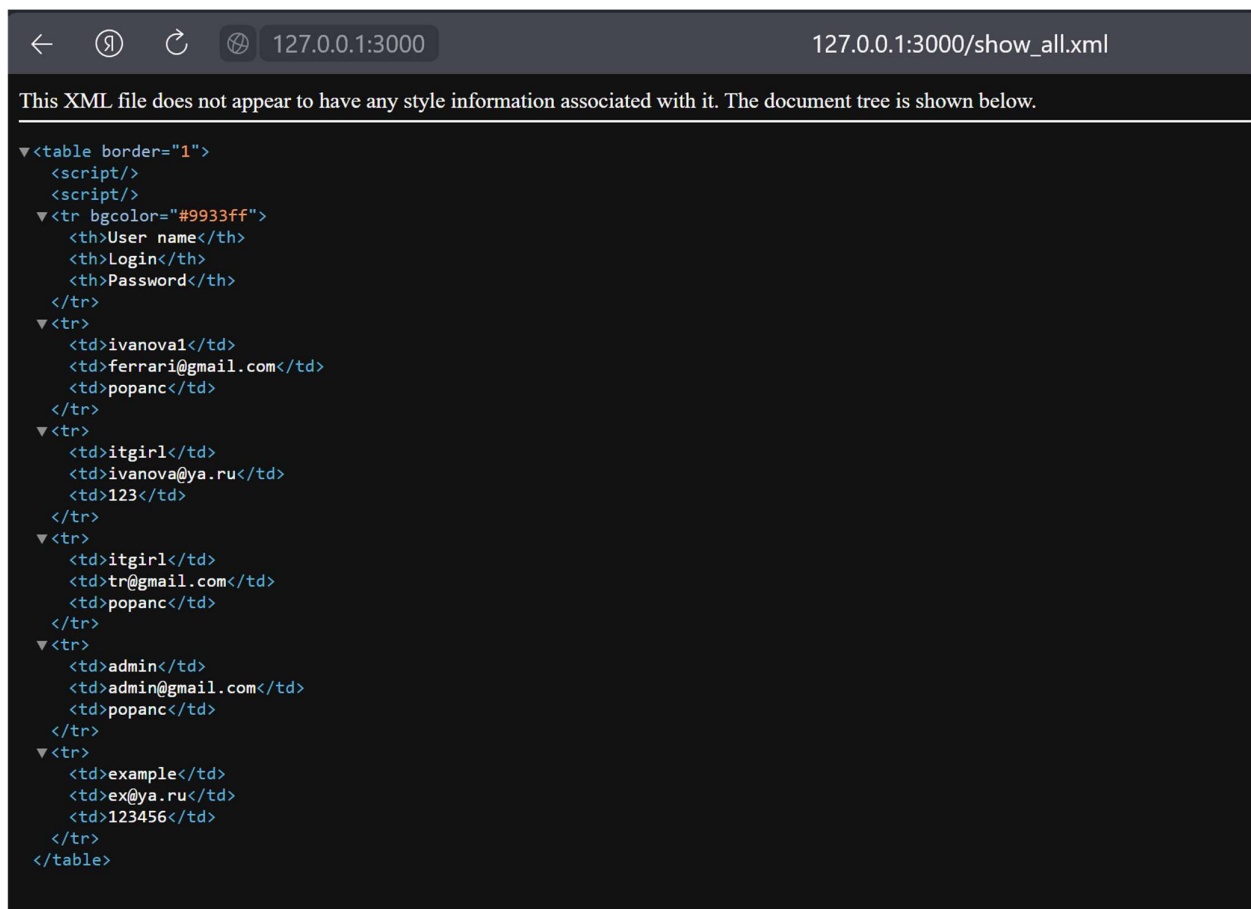


Рисунок 11 – вывод БД через сериализацию XML

```

PS D:\education\3 semester\ipl\lab\lab12\mine\chisla12> rake test:integration
Running 9 tests in a single process (parallelization threshold is 50)
Run options: --seed 40222

# Running:

123456
.
.123456
...123456
...

Finished in 3.236462s, 2.7808 runs/s, 6.4886 assertions/s.
9 runs, 21 assertions, 0 failures, 0 errors, 0 skips

```

Рисунок 12 – результат выполнения тестов

Распечатка БД:

```

<?xml version="1.0" encoding="System"?>
<table>
  <database></database>
  <name>users</name>
  <ddl><![CDATA[CREATE TABLE "users" ("id" integer PRIMARY KEY
AUTOINCREMENT NOT NULL, "name" varchar, "email" varchar, "password" text,
"created_at" datetime(6) NOT NULL, "updated_at" datetime(6) NOT NULL);]]></ddl>
  <columns>
    <column>
      <name>id</name>
      <type>integer</type>
      <constraints>
        <constraint>
          <type>PRIMARY KEY</type>
          <definition>PRIMARY KEY AUTOINCREMENT </definition>
        </constraint>
        <constraint>
          <type>NOT NULL</type>
          <definition>NOT NULL</definition>

```

```

        </constraint>
    </constraints>
</column>
<column>
    <name>name</name>
    <type>varchar</type>
</column>
<column>
    <name>email</name>
    <type>varchar</type>
</column>
<column>
    <name>password</name>
    <type>text</type>
</column>
<column>
    <name>created_at</name>
    <type>datetime</type>
    <constraints>
        <constraint>
            <type>NOT NULL</type>
            <definition>NOT NULL</definition>
        </constraint>
    </constraints>
</column>
<column>
    <name>updated_at</name>
    <type>datetime</type>
    <constraints>
        <constraint>

```



```

        <type>NOT NULL</type>
        <definition>NOT NULL</definition>
    </constraint>
</constraints>
</column>
</columns>
<rows>
    <row>
        <value column="0">2</value>
        <value column="1">ivanova1</value>
        <value column="2">ferrari@gmail.com</value>
        <value column="3">popanc</value>
        <value column="4">2023-11-26 23:55:57.667659</value>
        <value column="5">2023-11-26 23:55:57.667659</value>
    </row>
    <row>
        <value column="0">5</value>
        <value column="1">itgirl</value>
        <value column="2">ivanova@ya.ru</value>
        <value column="3">popanc</value>
        <value column="4">2023-11-27 12:24:01.226605</value>
        <value column="5">2023-11-27 12:33:43.836217</value>
    </row>
    <row>
        <value column="0">6</value>
        <value column="1">itgirl</value>
        <value column="2">tr@gmail.com</value>
        <value column="3">popanc</value>
        <value column="4">2023-11-27 13:15:55.933746</value>
        <value column="5">2023-11-27 13:15:55.933746</value>
    </row>

```

```
</row>
<row>
  <value column="0">7</value>
  <value column="1">admin</value>
  <value column="2">admin@gmail.com</value>
  <value column="3">popanc</value>
  <value column="4">2023-11-27 14:30:54.153318</value>
  <value column="5">2023-11-27 14:30:54.153318</value>
</row>
</rows>
</table>
```

Вывод: было создано веб-приложение, использующих аутентификацию и авторизацию и было изучено написание интеграционных тестов