|  |  |
| --- | --- |
| 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 №** | 12 |

**Название:**

Сессии. Выполнение авторизации. Интеграционные тесты.

**Дисциплина:** Языки интернет-программирования



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-35 Б |  | 08.12.2023 | И.А. Дулина |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Е.Ю. Гаврилова |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Вариант 8**

**Цель работы:** 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

**Cгенерируем 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);

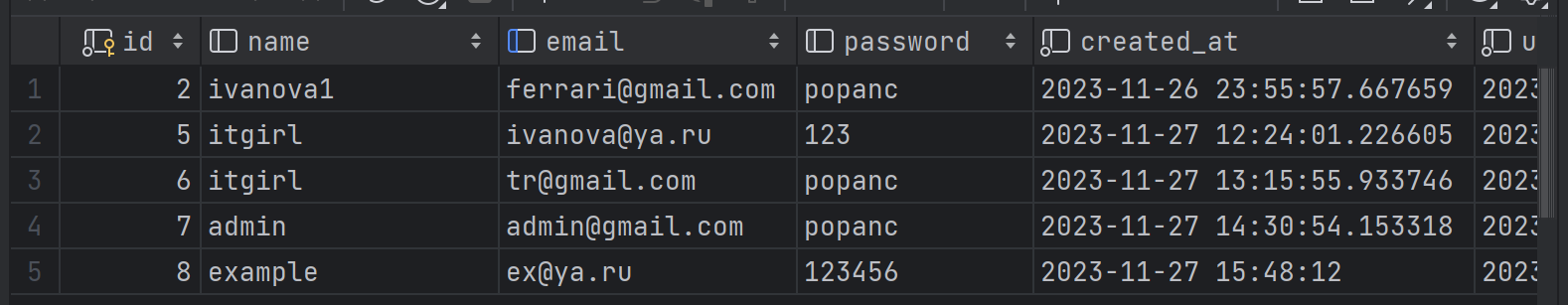


Рисунок 1 – добавление информации в БД Users

**Проверка редактирования информации:**

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

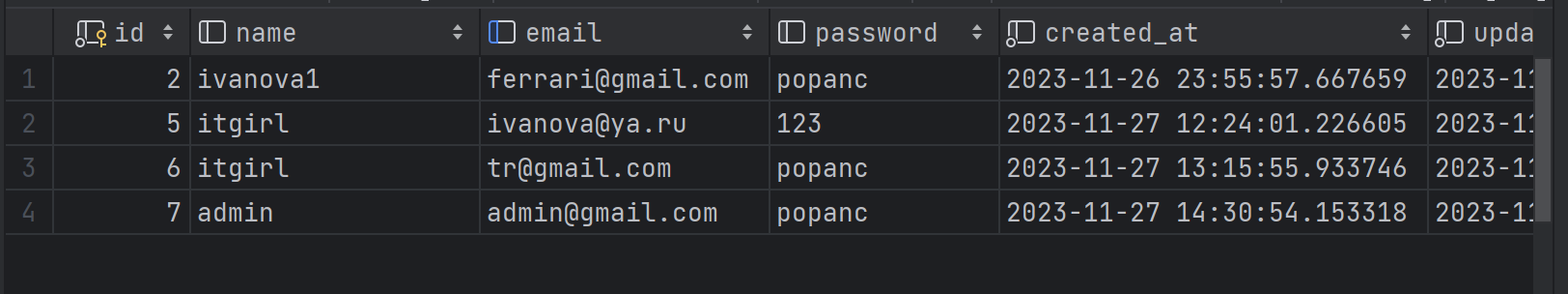


Рисунок 2 – изменение данных в БД Users

**Проверка получения списка пользователей:**

Select \* from users;

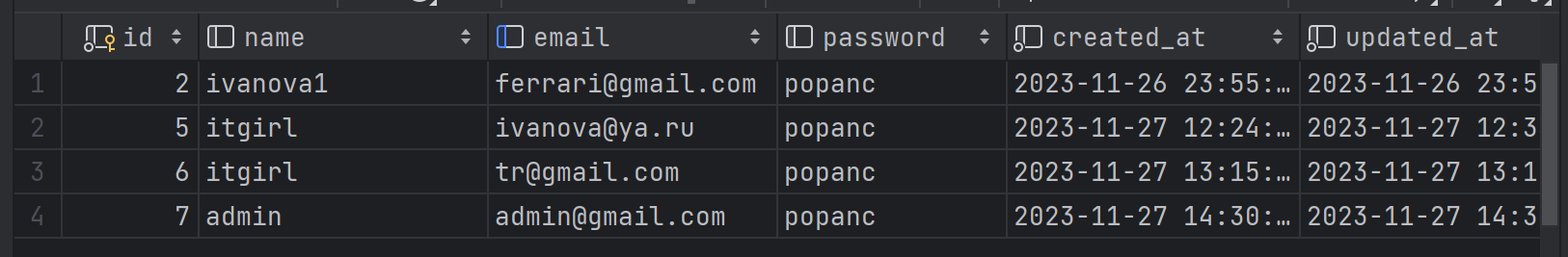


Рисунок 3 – вывод всех данных из БД Users

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]

break if i + 1 == res.length

break if (res[i+1] <= res[i])

i += 1

end

i+=1

[posl, 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

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

responce = "<catalog>#{rows}</catalog>"

format.xml { render xml: xslt\_transform(responce).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 зарегестрирован!'

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

# 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 }

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

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>

<br>

<div>

<input type="submit" value="Найти наиболее длинную монотонно возврастающую 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 %>

<br>

<%= 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 %>

</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-turbolinks-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 %>

<br>

<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 зарегестрировать 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 зарегестрировать

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

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

################################## Sign out ######################################

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 проверочка, что не залогинились 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

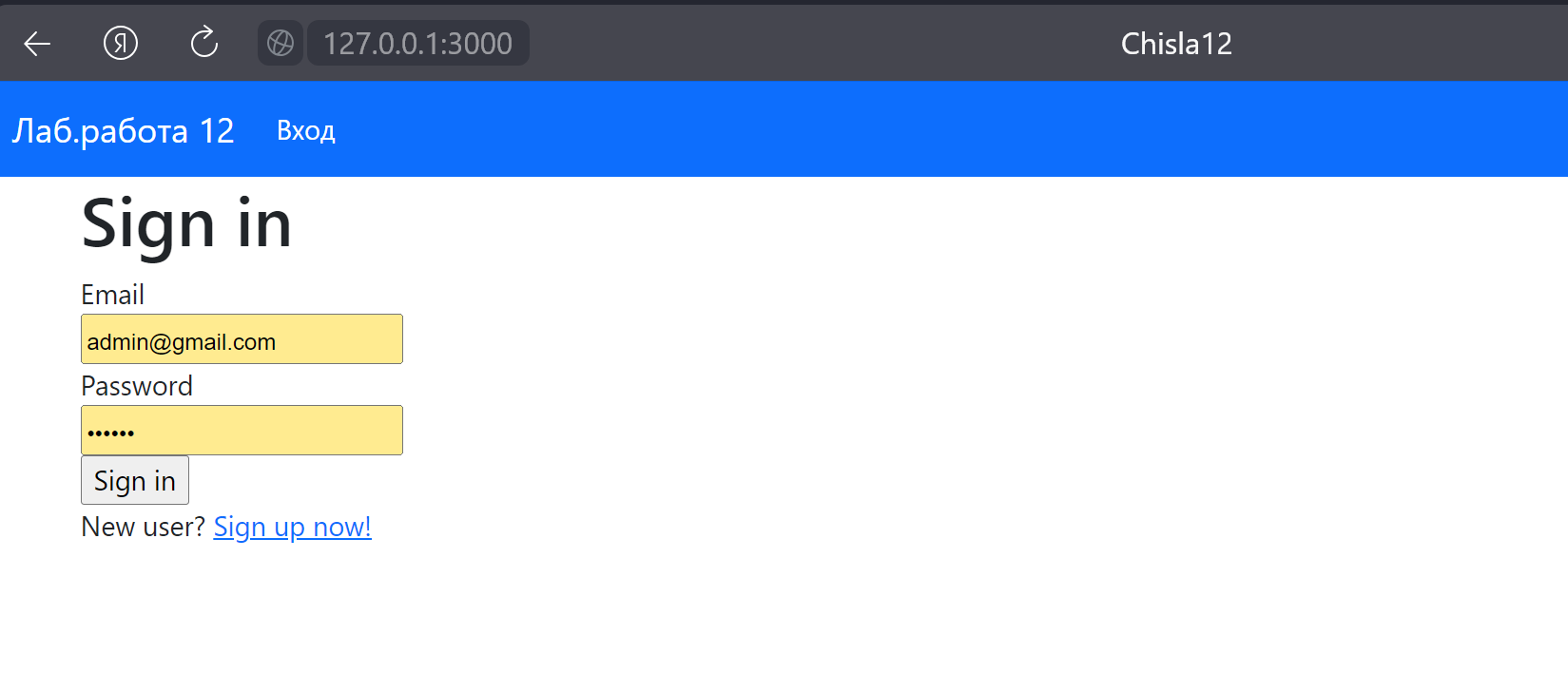


Рисунок 4 – форма входа

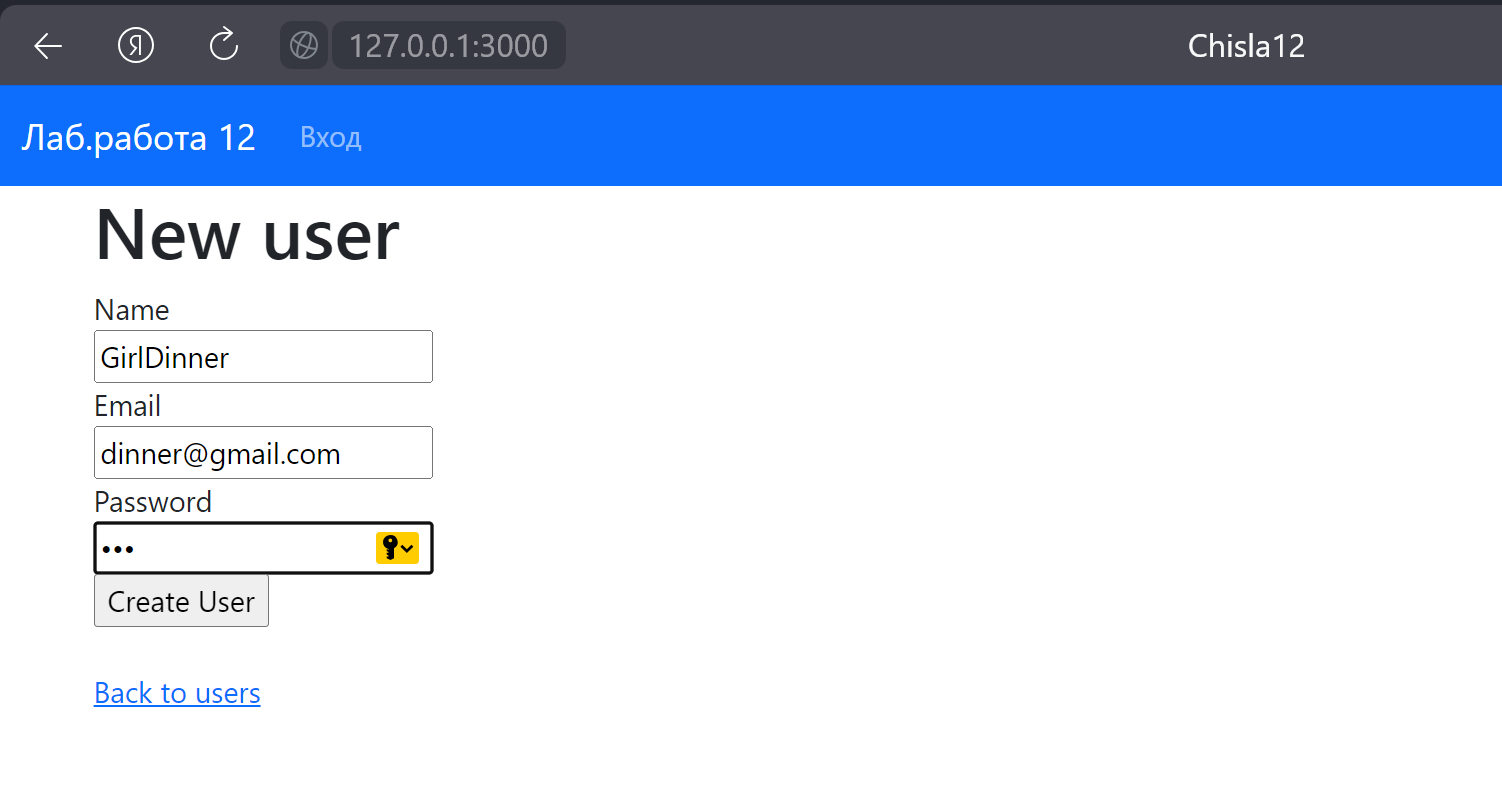


Рисунок 5 – форма регистрации

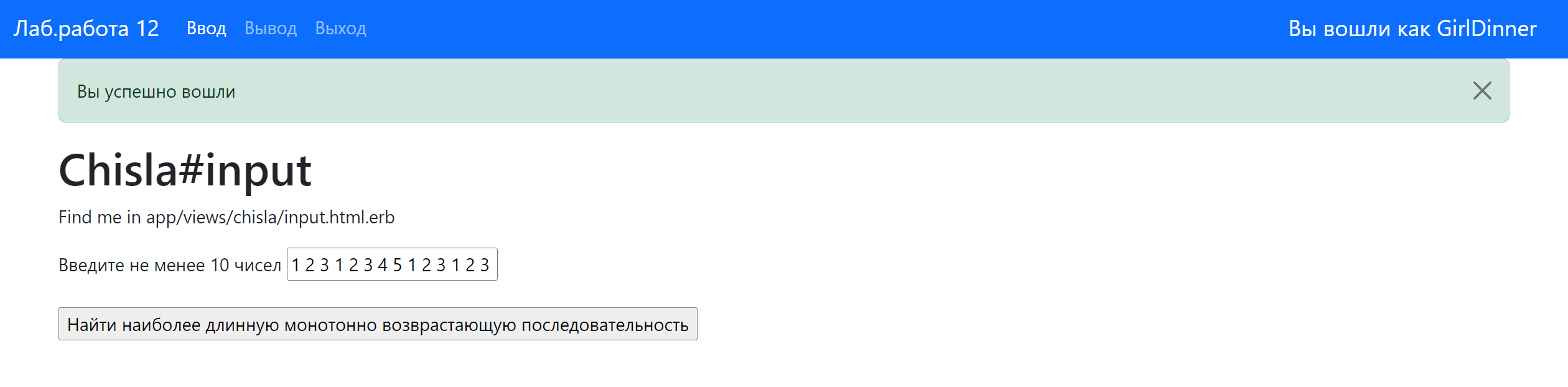


Рисунок 6 – успешный вход

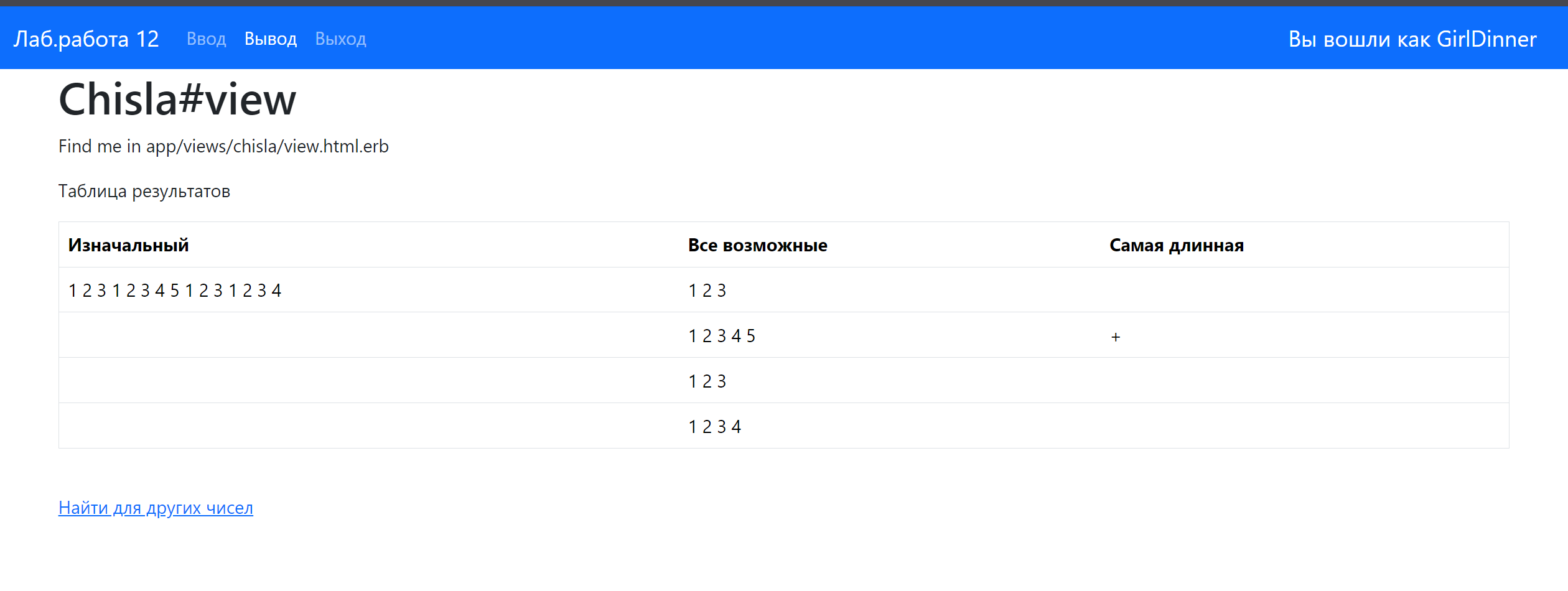


Рисунок 7 – вывод значения в результате успешного входа и ввода данных

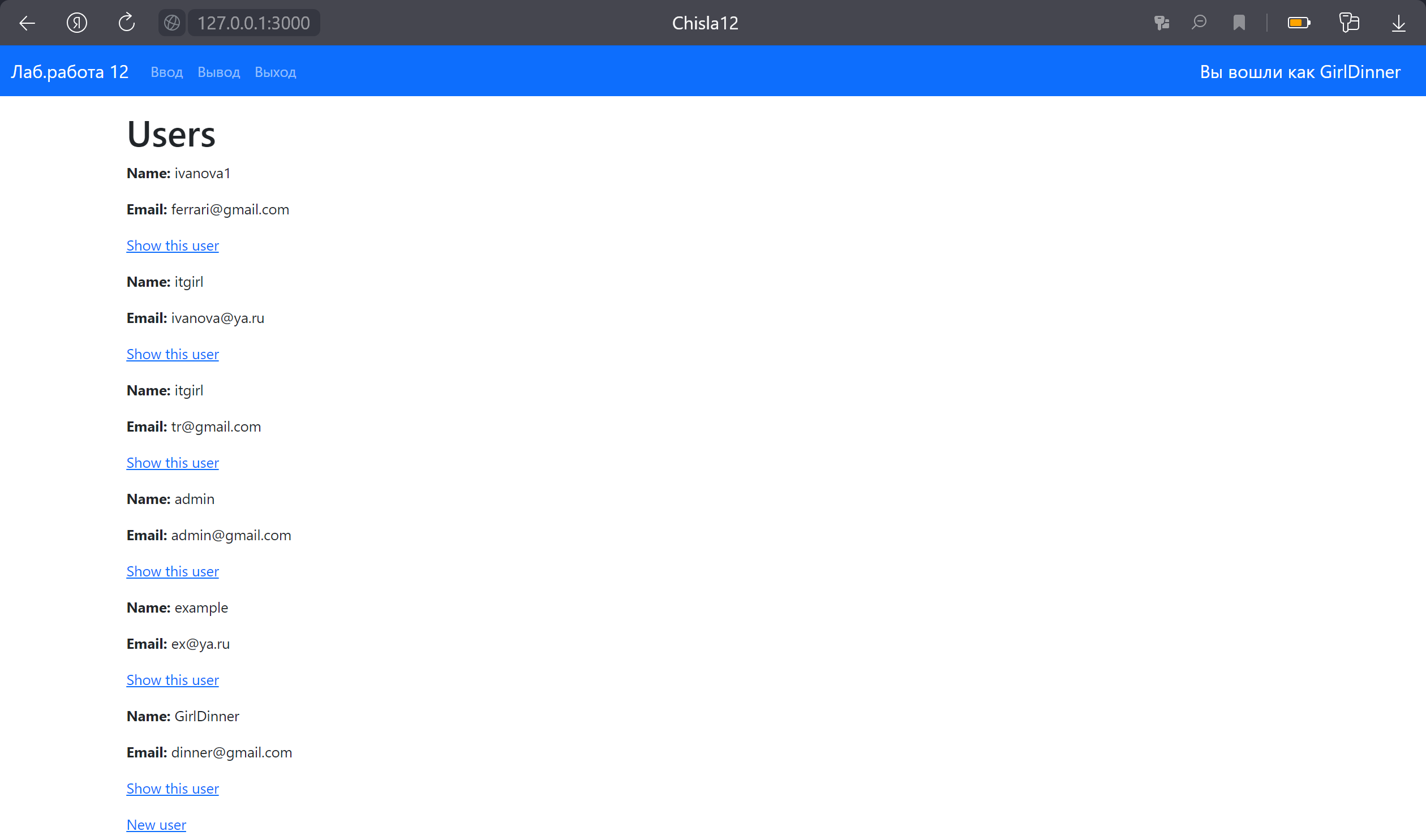


Рисунок 8 – вывод всех пользователей (/users)



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



Рисунок 10 – попытка удалить пользователя с аккаунта, являющегося админом

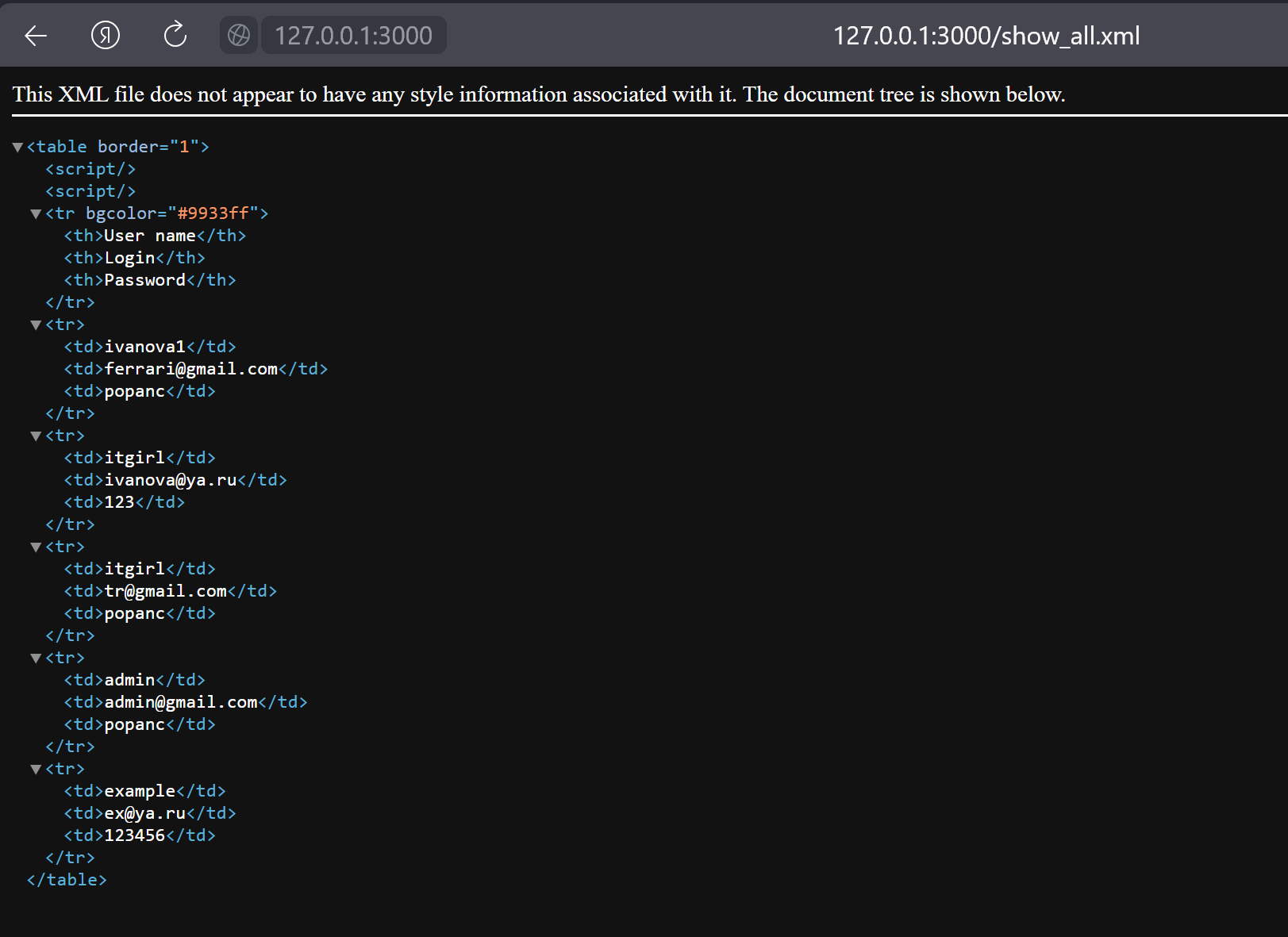


Рисунок 11 – вывод БД через сериализацию XML

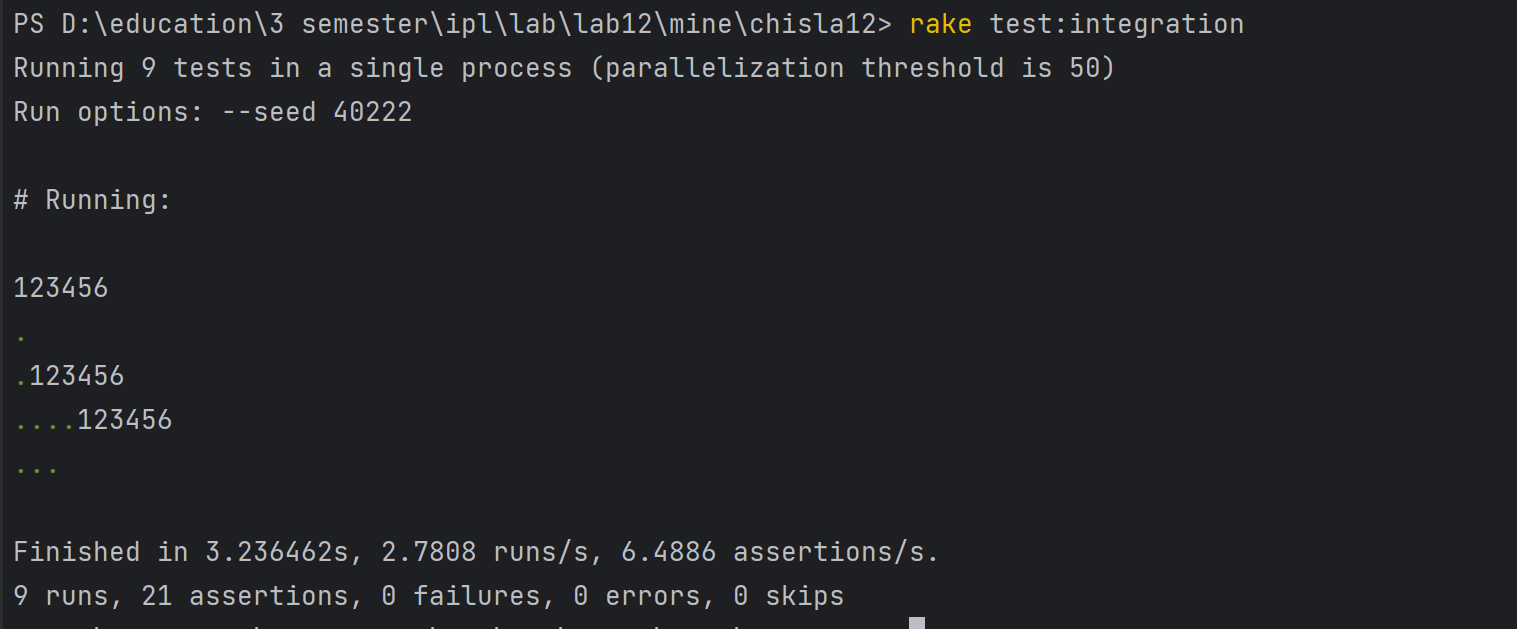


Рисунок 12 – результат выполнения тестов

**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>

<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