

# 江汉大学授权认证中心

# 源码文档

2024-8-20

import React, { useState } from 'react';

import { Form, Input, Button, Alert, Typography } from 'antd';

import { useNavigate } from 'react-router-dom';

import api from '../../api/api';

import 'antd/dist/reset.css'; // 导入 Ant Design 样式

const { Title, Link } = Typography;

const Login: React.FC = () => {

    const [message, setMessage] = useState<string>('');

    const navigate = useNavigate();

    const handleSubmit = async (values: { email: string; password: string }) => {

        try {

            const response = await api.post('/user/login', {

                email: values.email,

                password: values.password,

            });

            const token = response.data.token;

            localStorage.setItem('token', token);

            setMessage('登录成功');

            navigate('/home'); // 跳转到主页或其他你想跳转的页面

        } catch (error) {

            setMessage('登录失败');

        }

    };

    return (

        <div style={{ maxWidth: 400, margin: 'auto', padding: '20px' }}>

            <Title level={2} style={{ textAlign: 'center' }}>登录</Title>

            <Form

                name="login"

                initialValues={{ remember: true }}

                onFinish={handleSubmit}

                style={{ padding: '20px', background: '#fff', borderRadius: '8px', boxShadow: '0 4px 8px rgba(0,0,0,0.1)' }}

            >

                <Form.Item

                    name="email"

                    rules={[{ required: true, message: '请输入您的邮箱!' }]}

                >

                    <Input placeholder="邮箱" />

                </Form.Item>

                <Form.Item

                    name="password"

                    rules={[{ required: true, message: '请输入您的密码!' }]}

                >

                    <Input.Password placeholder="密码" />

                </Form.Item>

                <Form.Item>

                    <Button type="primary" htmlType="submit" block>

                        登录

</Button>

                </Form.Item>

                {message && (

                    <Form.Item>

                        <Alert message={message} type={message === '登录成功' ? 'success' : 'error'} showIcon />

                    </Form.Item>

                )}

                <Form.Item>

                    <Link href="http://localhost:8080/oauth/authorize?client\_id=9cb0c3b1-b5f3-475a-8a0b-17934bee2d99&redirect\_uri=http://localhost:3000/home&response\_type=code&scope=\*">

                        第三方登录

                    </Link>

                </Form.Item>

            </Form>

        </div>

    );

};

export default Login;

import React, { useEffect, useState } from 'react';

import { useNavigate } from 'react-router-dom';

import { Button, Table, Spin, Typography, Alert } from 'antd';

import api from '../../api/api'; // 确保 api 配置正确

const { Title } = Typography;

interface User {

    id: number;

    name: string;

    email: string;

    email\_verified\_at: string | null;

    created\_at: string;

    updated\_at: string;

}

const User: React.FC = () => {

    const [users, setUsers] = useState<User[]>([]);

    const [loading, setLoading] = useState(true);

    const [error, setError] = useState<string | null>(null);

    const navigate = useNavigate();

    useEffect(() => {

        const fetchUsers = async () => {

            try {

                const response = await api.post('/user/list'); // 确保 API 路径正确

                setUsers(response.data.data);

            } catch (err) {

                setError('Failed to fetch users');

            } finally {

                setLoading(false);

            }

        };

        fetchUsers();

    }, []);

    const columns = [

        {

            title: 'ID',

            dataIndex: 'id',

            key: 'id',

        },

        {

            title: '名称',

            dataIndex: 'name',

            key: 'name',

        },

        {

            title: '电子邮件',

            dataIndex: 'email',

            key: 'email',

        },

        {

            title: '创建时间',

            dataIndex: 'created\_at',

            key: 'created\_at',

            render: (text: string) => new Date(text).toLocaleString(),

        },

        {

            title: '更新时间',

            dataIndex: 'updated\_at',

            key: 'updated\_at',

            render: (text: string) => new Date(text).toLocaleString(),

        },

    ];

    if (loading) return <Spin size="large" />;

    if (error) return <Alert message={error} type="error" showIcon />;

    return (

        <div style={{ padding: 20 }}>

            <Title level={1}>用户列表</Title>

            <div style={{ marginBottom: 20 }}>

                <Button type="primary" onClick={() => navigate('/home')}>回到首页</Button>

            </div>

            <Table

                columns={columns}

                dataSource={users}

                rowKey="id"

                pagination={{ pageSize: 10 }}

            />

        </div>

    );

};

export default User;

// src/pages/Register.tsx

import React, { useState } from 'react';

import axios from 'axios';

import api from '../api/api';

const Register: React.FC = () => {

    const [username, setName] = useState('');

    const [email, setEmail] = useState('');

    const [password, setPassword] = useState('');

    // const [passwordConfirmation, setPasswordConfirmation] = useState('');

    const [message, setMessage] = useState('');

    const handleSubmit = async (e: React.FormEvent) => {

        e.preventDefault();

        try {

            const response = await api.post('/signup', {

                username,

                email,

                password,

                // password\_confirmation: passwordConfirmation,

            });

            setMessage(response.data.message);

        } catch (error) {

            console.error('Error registering:', error);

            setMessage('Registration failed');

        }

    };

    return (

        <div>

            <h2>注册</h2>

            <form onSubmit={handleSubmit}>

                <div>

                    <label>用户名:</label>

                    <input type="text" value={username} onChange={(e) => setName(e.target.value)} required />

                </div>

                <div>

                    <label>邮箱:</label>

                    <input type="email" value={email} onChange={(e) => setEmail(e.target.value)} required />

                </div>

                <div>

                    <label>密码:</label>

                    <input type="password" value={password} onChange={(e) => setPassword(e.target.value)} required />

                </div>

                {/\* <div>

                    <label>确认密码:</label>

                    <input type="password" value={passwordConfirmation} onChange={(e) => setPasswordConfirmation(e.target.value)} required />

                </div> \*/}

                <button type="submit">注册</button>

            </form>

            {message && <p>{message}</p>}

        </div>

    );

};

export default Register;

import React, { useEffect, useState } from 'react';

import { Button, Form, Input, Modal, Table, Typography, Spin, Alert } from 'antd';

import api from '../../api/api';

const { Title } = Typography;

interface Client {

    id: string;

    user\_id: string | null;

    name: string;

    provider: string | null;

    redirect: string;

    personal\_access\_client: boolean;

    password\_client: boolean;

    revoked: boolean;

    created\_at: string;

    updated\_at: string;

}

const Home: React.FC = () => {

    const [clients, setClients] = useState<Client[]>([]);

    const [loading, setLoading] = useState(true);

    const [error, setError] = useState<string | null>(null);

    const [showForm, setShowForm] = useState(false);

    const [newClient, setNewClient] = useState({ name: '', redirect: '' });

    useEffect(() => {

        const fetchClients = async () => {

            try {

                const response = await api.post('/client/list');

                setClients(response.data.data);

            } catch (err) {

                setError('Failed to fetch clients');

            } finally {

                setLoading(false);

            }

        };

        fetchClients();

    }, []);

    const handleAddClient = async () => {

        try {

            const response = await api.post('/client/create', {

                name: newClient.name,

                redirect: newClient.redirect,

            });

            setClients([...clients, response.data]);

            setShowForm(false);

            setNewClient({ name: '', redirect: '' });

        } catch (err) {

            setError('添加客户端失败');

        }

    };

    const handleDeleteClient = async (id: string) => {

        try {

            await api.post('/client/delete', { id: [id] });

            setClients(clients.filter(client => client.id !== id));

        } catch (err) {

            setError('Failed to delete client');

        }

    };

    const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {

        const { name, value } = e.target;

        setNewClient(prevState => ({

            ...prevState,

            [name]: value,

        }));

    };

    const columns = [

        { title: 'ID', dataIndex: 'id', key: 'id' },

        { title: '客户端', dataIndex: 'name', key: 'name' },

        { title: '重定向地址', dataIndex: 'redirect', key: 'redirect' },

        { title: 'Personal Access Client', dataIndex: 'personal\_access\_client', key: 'personal\_access\_client', render: (text: boolean) => (text ? 'Yes' : 'No') },

        { title: 'Password Client', dataIndex: 'password\_client', key: 'password\_client', render: (text: boolean) => (text ? 'Yes' : 'No') },

        { title: 'Revoked', dataIndex: 'revoked', key: 'revoked', render: (text: boolean) => (text ? 'Yes' : 'No') },

        { title: '创建时间', dataIndex: 'created\_at', key: 'created\_at', render: (text: string) => new Date(text).toLocaleString() },

        { title: '更新时间', dataIndex: 'updated\_at', key: 'updated\_at', render: (text: string) => new Date(text).toLocaleString() },

        {

            title: '操作',

            key: 'action',

            render: (\_: any, record: Client) => (

                <Button type="link" danger onClick={() => handleDeleteClient(record.id)}>

                    删除

                </Button>

            ),

        },

    ];

    if (loading) return <Spin size="large" />;

    if (error) return <Alert message={error} type="error" showIcon />;

    return (

        <div style={{ padding: 20 }}>

            <div style={{ marginBottom: 20 }}>

                <Button type="primary" onClick={() => window.location.href = '/apply'} style={{ marginRight: 10 }}>申请列表</Button>

                <Button type="primary" onClick={() => window.location.href = '/user'}>用户管理</Button>

            </div>

            <Title level={1}>客户端列表</Title>

            <Button type="primary" onClick={() => setShowForm(true)} style={{ marginBottom: 20 }}>添加客户端</Button>

            <Table columns={columns} dataSource={clients} rowKey="id" />

            <Modal

                title="添加客户端"

                visible={showForm}

                onOk={handleAddClient}

                onCancel={() => setShowForm(false)}

                okText="提交"

                cancelText="取消"

            >

                <Form layout="vertical" onFinish={handleAddClient}>

                    <Form.Item

                        label="名称"

                        name="name"

                        rules={[{ required: true, message: '请输入客户端名称!' }]}

                    >

                        <Input

                            value={newClient.name}

                            onChange={handleInputChange}

                            name="name"

                        />

                    </Form.Item>

                    <Form.Item

                        label="重定向 URL"

                        name="redirect"

                        rules={[{ required: true, message: '请输入重定向 URL!' }]}

                    >

                        <Input

                            type="url"

                            value={newClient.redirect}

                            onChange={handleInputChange}

                            name="redirect"

                        />

                    </Form.Item>

                </Form>

            </Modal>

        </div>

    );

};

export default Home;

import React, { useEffect, useState } from 'react';

import axios from 'axios';

import { useNavigate } from 'react-router-dom';

import { Button, Table, Select, Typography, Spin, Alert } from 'antd';

import { ColumnsType } from 'antd/es/table';

import './Apply.css';

const { Title } = Typography;

const { Option } = Select;

interface ClientApplication {

    id: number;

    applicant: string;

    phone\_number: string;

    client\_name: string;

    callback\_url: string;

    status: number;

    created\_at: string;

}

const Apply: React.FC = () => {

    const [applications, setApplications] = useState<ClientApplication[]>([]);

    const [loading, setLoading] = useState(true);

    const [error, setError] = useState<string | null>(null);

    const navigate = useNavigate();

    useEffect(() => {

        axios.post('http://localhost:8080/api/applylist')

            .then(response => {

                setApplications(response.data);

            })

            .catch(error => {

                setError('Error fetching data');

            })

            .finally(() => {

                setLoading(false);

            });

    }, []);

    const handleStatusChange = (id: number, newStatus: number) => {

        axios.post('http://localhost:8080/api/applyupdate', {

            id: id,

            status: newStatus

        })

        .then(() => {

            setApplications(prevApplications =>

                prevApplications.map(app =>

                    app.id === id ? { ...app, status: newStatus } : app

                )

            );

        })

        .catch(error => {

            setError('Error updating status');

        });

    };

    const columns: ColumnsType<ClientApplication> = [

        { title: 'ID', dataIndex: 'id', key: 'id' },

        { title: '申请人', dataIndex: 'applicant', key: 'applicant' },

        { title: '电话号码', dataIndex: 'phone\_number', key: 'phone\_number' },

        { title: '客户端名称', dataIndex: 'client\_name', key: 'client\_name' },

        {

            title: '回调地址',

            dataIndex: 'callback\_url',

            key: 'callback\_url',

            render: (text: string) => (

                <a href={text} target="\_blank" rel="noopener noreferrer">{text}</a>

            )

        },

        {

            title: '申请时间',

            dataIndex: 'created\_at',

            key: 'created\_at',

            render: (text: string) => new Date(text).toLocaleString()

        },

        {

            title: '状态',

            dataIndex: 'status',

            key: 'status',

            render: (status: number, record: ClientApplication) => (

                <Select

                    value={status}

                    onChange={(value) => handleStatusChange(record.id, value)}

                    style={{ width: 120 }}

                >

                    <Option value={1}>待审批</Option>

                    <Option value={2}>已通过</Option>

                    <Option value={3}>未通过</Option>

                </Select>

            )

        },

    ];

    if (loading) return <Spin size="large" />;

    if (error) return <Alert message={error} type="error" showIcon />;

    return (

        <div style={{ padding: 20 }}>

            <Title level={1}>申请列表</Title>

            <Button type="primary" onClick={() => navigate('/home')} style={{ marginBottom: 20 }}>

                回到首页

            </Button>

            <Table

                columns={columns}

                dataSource={applications}

                rowKey="id"

                pagination={{ pageSize: 10 }}

            />

        </div>

    );

};

export default Apply;

<?php

namespace App\Http\Controllers\Auth;

use Illuminate\Http\Request;

use App\Http\Controllers\Controller;

use App\Models\Client;

use Laravel\Passport\ClientRepository;

use Illuminate\Support\Facades\Auth;

class ClientController extends Controller

{

    public function list(Request $request)

    {

        $request->validate([

            'current' => 'integer|nullable',

            'pageSize' => 'integer|nullable',

        ]);

        // 获取当前页码，默认为 1

        $current = $request->input('current', 1);

        // 获取每页显示的记录数，默认为 10

        $pageSize = $request->input('pageSize', 10);

        // 排除特定的 id

        $excludedIds = [

            '9cab4d79-1dff-4bd2-ab5b-3ec6bc05b2fa',

            '9cab4d79-188a-4959-80e3-e532be230b26'

        ];

        // 使用分页查询并排除特定的 id,

        $clients = Client::whereNotIn('id', $excludedIds)

            ->where('revoked', '!=', 1)

            ->paginate($pageSize, ['\*'], 'page', $current);

            // 返回用户列表和分页信息

        return response()->json([

            'data' => $clients->items(),

            'total' => $clients->total(),

            'current' => $clients->currentPage(),

            'pageSize' => $clients->perPage(),

        ]);

    }

    // 创建一个新的客户端

    public function create(Request $request)

    {

        // 验证字段

        $request->validate([

            'name' => 'required',

            'redirect' => 'required|url',

        ]);

        $clientRepository = app(ClientRepository::class);

        $client = $clientRepository->create(

             $request->name, $request->redirect

        );

        // exit;

        return response()->json($client);

    }

    // 更新客户端

    public function update(Request $request)

    {

        // 验证字段

        $request->validate([

            'id' => 'required|uuid', // 确保 id 是 UUID 格式

            'name' => 'required',

            'redirect' => 'required|url',

        ]);

        // 获取 Client 实例

        $clientRepository = app(ClientRepository::class);

        $client = $clientRepository->find($request->id);

        if (!$client) {

            return response()->json(['error' => 'Client not found'], 404);

        }

        // 更新 Client

        $updatedClient = $clientRepository->update(

            $client, $request->name, $request->redirect

        );

        return response()->json($updatedClient);

    }

    // 删除客户端

    public function delete(Request $request)

    {

        $request->validate([

            'id' => 'required|array',

            'id.\*' => 'uuid',

        ]);

        $clientRepository = app(ClientRepository::class);

        foreach ($request->id as $id) {

            $client = $clientRepository->find($id);

            if ($client) {

                $clientRepository->delete($client);

            }

        }

        return response()->json(['msg' => '删除成功']);

    }

}

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use Illuminate\Foundation\Auth\ConfirmsPasswords;

class ConfirmPasswordController extends Controller

{

    /\*

    |--------------------------------------------------------------------------

    | Confirm Password Controller

    |--------------------------------------------------------------------------

    |

    | This controller is responsible for handling password confirmations and

    | uses a simple trait to include the behavior. You're free to explore

    | this trait and override any functions that require customization.

    |

    \*/

    use ConfirmsPasswords;

    /\*\*

     \* Where to redirect users when the intended url fails.

     \*

     \* @var string

     \*/

    protected $redirectTo = '/home';

    /\*\*

     \* Create a new controller instance.

     \*

     \* @return void

     \*/

    public function \_\_construct()

    {

        $this->middleware('auth');

    }

}

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use Illuminate\Foundation\Auth\AuthenticatesUsers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Session;

class LoginController extends Controller

{

    /\*

    |--------------------------------------------------------------------------

    | Login Controller

    |--------------------------------------------------------------------------

    |

    | This controller handles authenticating users for the application and

    | redirecting them to your home screen. The controller uses a trait

    | to conveniently provide its functionality to your applications.

    |

    \*/

    use AuthenticatesUsers;

    /\*\*

     \* Where to redirect users after login.

     \*

     \* @var string

     \*/

    protected $redirectTo = '/home';

    /\*\*

     \* Create a new controller instance.

     \*

     \* @return void

     \*/

    public function \_\_construct()

    {

        $this->middleware('guest')->except('logout');

        $this->middleware('auth')->only('logout');

    }

    protected function credentials(Request $request)

    {

        // 验证码验证

        $request->validate([

            'captcha' => 'required|numeric',

        ]);

        // 验证码检查

        $captcha = $request->input('captcha');

        if ($captcha != Session::get('captcha')) {

            return back()->withErrors(['captcha' => '验证码错误']);

        }

        // 清除验证码

        Session::forget('captcha');

        return $request->only('email', 'password');

    }

}

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

use App\Models\User;

use Illuminate\Foundation\Auth\RegistersUsers;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller

{

    /\*

    |--------------------------------------------------------------------------

    | Register Controller

    |--------------------------------------------------------------------------

    |

    | This controller handles the registration of new users as well as their

    | validation and creation. By default this controller uses a trait to

    | provide this functionality without requiring any additional code.

    |

    \*/

    use RegistersUsers;

    /\*\*

     \* Where to redirect users after registration.

     \*

     \* @var string

     \*/

    protected $redirectTo = '/home';

    /\*\*

     \* Create a new controller instance.

     \*

     \* @return void

     \*/

    public function \_\_construct()

    {

        $this->middleware('guest');

    }

    /\*\*

     \* Get a validator for an incoming registration request.

     \*

     \* @param  array  $data

     \* @return \Illuminate\Contracts\Validation\Validator

     \*/

    protected function validator(array $data)

    {

        return Validator::make($data, [

            'name' => ['required', 'string', 'max:255'],

            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],

            'password' => ['required', 'string', 'min:8', 'confirmed'],

        ]);

    }

    /\*\*

     \* Create a new user instance after a valid registration.

     \*

     \* @param  array  $data

     \* @return \App\Models\User

     \*/

    protected function create(array $data)

    {

        return User::create([

            'name' => $data['name'],

            'email' => $data['email'],

            'password' => Hash::make($data['password']),

        ]);

    }

}

<?php

namespace App\Http\Controllers\User;

use Illuminate\Http\Request;

use App\Models\User;

use App\Http\Controllers\Controller;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Str;

class UserController extends Controller

{

    public function signup(Request $request){

        $request->validate([

            'name'=>'required',

            'email'=>'required',

            'password'=>'required'

        ]);

        $name=$request->input('name');

        $email=$request->input('email');

        $password=$request->input('password');

        $check=User::where('email',$email)->first();

        if($check){

            return response()->json(['msg'=>'此邮箱已被占用！']);

        }

        $user = User::create(['name' => $name, 'email' => $email, 'password' => $password]);

        if($user){

            return response()->json(['msg'=>'注册成功']);

        }else{

            return response()->json(['msg'=>'注册失败']);

        }

    }

    public function login(Request $request) {

        // 验证请求数据

        $request->validate([

            'email' => 'required',

            'password' => 'required',

        ]);

        // 查找用户

        $user = User::where('email', $request->email)->first();

        if ($user && Hash::check($request->password, $user->password)) {

            $token = $user->createToken('personal token');

            return response()->json(['msg' => '登录成功', 'data' => $token]);

        } else {

            // 登录失败

            return response()->json(['msg' => '登录失败'], 401);

        }

    }

    public function info(Request $request){

        $userinfo=$request->user('api');

        return response()->json(['data'=>$userinfo]);

    }

    public function logout(Request $request){

        $user = $request->user('api');

        if ($user) {

            $user->token()->revoke();

            return response()->json(['msg' => '成功退出登录']);

        } else {

            return response()->json(['msg' => '用户未认证'], 401);

        }

    }

    public function list(Request $request)

    {

        $request->validate([

            'current' => 'integer|nullable',

            'pageSize' => 'integer|nullable',

        ]);

        $current = $request->input('current', 1);

        $pageSize = $request->input('pageSize', 10);

            // 查询用户列表

        $users = User::paginate($pageSize, ['\*'], 'page', $current);

        // 返回用户列表和分页信息

        return response()->json([

            'data' => $users->items(),

            'current' => $users->currentPage(),

            'pageSize' => $users->perPage(),

            'total' => $users->total(),

        ]);

    }

    public function update(Request $request)

    {

        $request->validate([

            'id' => 'required',

            'name' => 'string',

            'email' => 'email',

            'password' => 'string',

        ]);

        $user = User::find($request->id);

        $user->name = $request->name;

        $user->email = $request->email;

        $user->password = Hash::make($request->password);

        $user->save();

        return response()->json(['msg' => '更新成功']);

    }

    public function delete(Request $request)

    {

       // 数据为id列表

        $request->validate([

            'id' => 'required|array',

        ]);

        $ids = $request->input('id');

        User::destroy($ids);

        return response()->json(['msg' => '删除成功']);

    }

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Session;

class CaptchaController extends Controller

{

    public function show()

    {

        $width = 120;

        $height = 40;

        // 创建图像资源

        $image = imagecreatetruecolor($width, $height);

        // 定义颜色

        $background = imagecolorallocate($image, 255, 255, 255); // 背景色

        $textColor = imagecolorallocate($image, 0, 0, 0); // 文字颜色

        $lineColor = imagecolorallocate($image, 150, 150, 150); // 干扰线颜色

        $noiseColor = imagecolorallocate($image, 200, 200, 200); // 噪点颜色

        // 填充背景

        imagefilledrectangle($image, 0, 0, $width, $height, $background);

        // 添加噪点

        for ($i = 0; $i < 50; $i++) {

            imagefilledrectangle($image, rand(0, $width), rand(0, $height), rand(0, $width), rand(0, $height), $noiseColor);

        }

        // 添加干扰线

        for ($i = 0; $i < 5; $i++) {

            imageline($image, rand(0, $width), rand(0, $height), rand(0, $width), rand(0, $height), $lineColor);

        }

        // 生成验证码文本

        $captchaText = rand(1000, 9999);

        Session::put('captcha', $captchaText);

        // 使用内置字体绘制文本

        imagestring($image, 5, 20, 10, $captchaText, $textColor);

        // 设置响应头并输出图像

        header('Content-Type: image/png');

        imagepng($image);

        imagedestroy($image);

    }

}

<?php

namespace App\Http\Controllers;

use App\Models\ClientApplication;

use Illuminate\Http\Request;

use App\Models\Client;

use Laravel\Passport\ClientRepository;

class ClientApplicationController extends Controller

{

    public function index()

    {

        $applications = ClientApplication::all();

        return response()->json($applications);

    }

    public function store(Request $request)

    {

        $request->validate([

            'applicant' => 'required|string|max:255',

            'phone\_number' => 'required|string|max:20',

            'client\_name' => 'required|string|max:255',

            'callback\_url' => 'required|string|max:2048',

            //  'status' => 'required|integer'

        ]);

        $application = ClientApplication::create([

            'applicant' => $request->applicant,

            'phone\_number' => $request->phone\_number,

            'client\_name' => $request->client\_name,

            'callback\_url' => $request->callback\_url,

            'status' => $request->status,

        ]);

        return response()->json($application, 201);

    }

    public function update(Request $request)

    {

        $request->validate([

            'id' => 'required|integer|exists:client\_applications,id',

            'status' => 'required|integer|in:1,2,3'

        ]);

        $application = ClientApplication::findOrFail($request->id);

        $application->status = $request->status;

        $application->save();

        if ($request->status === 2) {

            $clientRepository = app(ClientRepository::class);

            $clientRepository->create(

                $application->client\_name,

                $application->callback\_url

            );

        }

        return response()->json($application);

    }

}

<?php

namespace App\Models;

use Laravel\Passport\Client as BaseClient;

use Illuminate\Database\Eloquent\Factories\HasFactory;

class Client extends BaseClient

{

    protected $table = 'oauth\_clients';

    use HasFactory;

    protected $fillable = [

        'id' ,

        'user\_id' ,

        'name' ,

        'secret' ,

    ];

    protected function casts(): array

    {

        return [

            'id' => 'string',

        ];

    }

    /\*\*

     \* 确定客户端是否应该跳过授权提示。

     \*/

    public function skipsAuthorization(): bool

    {

        return $this->firstParty();

    }

}

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class ClientApplication extends Model

{

    use HasFactory;

    protected $fillable = [

        'applicant',

        'phone\_number',

        'client\_name',

        'callback\_url',

        'status',

    ];

    public $timestamps = false; // 禁用时间戳

}

<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

use Laravel\Passport\HasApiTokens;

class User extends Authenticatable

{

    use HasFactory, Notifiable, HasApiTokens;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array<int, string>

     \*/

    protected $fillable = [

        'name',

        'email',

        'password',

    ];

    /\*\*

     \* The attributes that should be hidden for serialization.

     \*

     \* @var array<int, string>

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

    /\*\*

     \* Get the attributes that should be cast.

     \*

     \* @return array<string, string>

     \*/

    protected function casts(): array

    {

        return [

            'email\_verified\_at' => 'datetime',

            'password' => 'hashed',

        ];

    }

}

<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

use Laravel\Passport\Passport;

class AppServiceProvider extends ServiceProvider

{

    /\*\*

     \* Register any application services.

     \*/

    public function register(): void

    {

        //

    }

    /\*\*

     \* Bootstrap any application services.

     \*/

    public function boot(): void

    {

        //

        // $this->registerPolicies();

        //  Passport::routes();

        Passport::tokensExpireIn(now()->addDays(15));

        Passport::refreshTokensExpireIn(now()->addDays(30));

        Passport::personalAccessTokensExpireIn(now()->addMonths(6));

    }

}

@extends('layouts.app')

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="col-md-8">

            <div class="card">

                <div class="card-header">{{ \_\_('登录') }}</div>

                <div class="card-body">

                    <form method="POST" action="{{ route('login') }}">

                        @csrf

                        <div class="row mb-3">

                            <label for="email" class="col-md-4 col-form-label text-md-end">{{ \_\_('邮箱') }}</label>

                            <div class="col-md-6">

                                <input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" required autocomplete="email" autofocus>

                                @error('email')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <label for="password" class="col-md-4 col-form-label text-md-end">{{ \_\_('密码') }}</label>

                            <div class="col-md-6">

                                <input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="current-password">

                                @error('password')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <label for="captcha" class="col-md-4 col-form-label text-md-end">{{ \_\_('验证码') }}</label>

                            <div class="col-md-6 d-flex align-items-center">

                                <img src="{{ url('/captcha') }}" alt="验证码" class="img-fluid me-2" style="max-width: 120px; height: auto;">

                                <input id="captcha" type="text" class="form-control @error('captcha') is-invalid @enderror" name="captcha" required>

                                @error('captcha')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <div class="col-md-6 offset-md-4">

                                <div class="form-check">

                                    <input class="form-check-input" type="checkbox" name="remember" id="remember" {{ old('remember') ? 'checked' : '' }}>

                                    <label class="form-check-label" for="remember">

                                        {{ \_\_('记住密码') }}

                                    </label>

                                </div>

                            </div>

                        </div>

                        <div class="row mb-0">

                            <div class="col-md-8 offset-md-4">

                                <button type="submit" class="btn btn-primary">

                                    {{ \_\_('登录') }}

                                </button>

                                @if (Route::has('password.request'))

                                    <a class="btn btn-link" href="{{ route('password.request') }}">

                                        <!-- {{ \_\_('Forgot Your Password?') }} -->

                                    </a>

                                @endif

                            </div>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

@extends('layouts.app')

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="col-md-8">

            <div class="card">

                <div class="card-header">{{ \_\_('注册') }}</div>

                <div class="card-body">

                    <form method="POST" action="{{ route('register') }}">

                        @csrf

                        <div class="row mb-3">

                            <label for="name" class="col-md-4 col-form-label text-md-end">{{ \_\_('用户名') }}</label>

                            <div class="col-md-6">

                                <input id="name" type="text" class="form-control @error('name') is-invalid @enderror" name="name" value="{{ old('name') }}" required autocomplete="name" autofocus>

                                @error('name')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <label for="email" class="col-md-4 col-form-label text-md-end">{{ \_\_('邮箱 ') }}</label>

                            <div class="col-md-6">

                                <input id="email" type="email" class="form-control @error('email') is-invalid @enderror" name="email" value="{{ old('email') }}" required autocomplete="email">

                                @error('email')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <label for="password" class="col-md-4 col-form-label text-md-end">{{ \_\_('密码') }}</label>

                            <div class="col-md-6">

                                <input id="password" type="password" class="form-control @error('password') is-invalid @enderror" name="password" required autocomplete="new-password">

                                @error('password')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

                        <div class="row mb-3">

                            <label for="password-confirm" class="col-md-4 col-form-label text-md-end">{{ \_\_('确认密码 ') }}</label>

                            <div class="col-md-6">

                                <input id="password-confirm" type="password" class="form-control" name="password\_confirmation" required autocomplete="new-password">

                            </div>

                        </div>

                        <div class="row mb-0">

                            <div class="col-md-6 offset-md-4">

                                <button type="submit" class="btn btn-primary">

                                    {{ \_\_('注册') }}

                                </button>

                            </div>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

@extends('layouts.app')

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="col-md-8">

            <div class="card">

                <div class="card-header">{{ \_\_('Verify Your Email Address') }}</div>

                <div class="card-body">

                    @if (session('resent'))

                        <div class="alert alert-success" role="alert">

                            {{ \_\_('A fresh verification link has been sent to your email address.') }}

                        </div>

                    @endif

                    {{ \_\_('Before proceeding, please check your email for a verification link.') }}

                    {{ \_\_('If you did not receive the email') }},

                    <form class="d-inline" method="POST" action="{{ route('verification.resend') }}">

                        @csrf

                        <button type="submit" class="btn btn-link p-0 m-0 align-baseline">{{ \_\_('click here to request another') }}</button>.

                    </form>

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

<!doctype html>

<html lang="{{ str\_replace('\_', '-', app()->getLocale()) }}">

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- CSRF Token -->

    <meta name="csrf-token" content="{{ csrf\_token() }}">

    <title>{{ config('app.name', '江汉大学授权中心') }}</title>

    <link rel="icon" href="{{ asset('jhun.png') }}" type="image/x-icon">

    <!-- Fonts -->

    <link rel="dns-prefetch" href="//fonts.bunny.net">

    <link href="https://fonts.bunny.net/css?family=Nunito" rel="stylesheet">

    <!-- Scripts -->

    @viteReactRefresh

    @vite(['resources/sass/app.scss', 'resources/js/app.js'])

</head>

<body>

    <div id="app">

        <nav class="navbar navbar-expand-md navbar-light bg-white shadow-sm">

            <div class="container">

                <a class="navbar-brand" href="{{ url('/') }}">

                    {{ config('江汉大学授权中心', '江汉大学授权中心') }}

                </a>

                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="{{ \_\_('Toggle navigation') }}">

                    <span class="navbar-toggler-icon"></span>

                </button>

                <div class="collapse navbar-collapse" id="navbarSupportedContent">

                    <!-- Left Side Of Navbar -->

                    <ul class="navbar-nav me-auto">

                    </ul>

                    <!-- Right Side Of Navbar -->

                    <ul class="navbar-nav ms-auto">

                        <!-- Authentication Links -->

                        @guest

                            @if (Route::has('login'))

                                <li class="nav-item">

                                    <a class="nav-link" href="{{ route('login') }}">{{ \_\_('登录') }}</a>

                                </li>

                            @endif

                            @if (Route::has('register'))

                                <li class="nav-item">

                                    <a class="nav-link" href="{{ route('register') }}">{{ \_\_('注册') }}</a>

                                </li>

                            @endif

                        @else

                            <li class="nav-item dropdown">

                                <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>

                                    {{ Auth::user()->name }}

                                </a>

                                <div class="dropdown-menu dropdown-menu-end" aria-labelledby="navbarDropdown">

                                    <a class="dropdown-item" href="{{ route('logout') }}"

                                       onclick="event.preventDefault();

                                                     document.getElementById('logout-form').submit();">

                                        {{ \_\_('退出') }}

                                    </a>

                                    <form id="logout-form" action="{{ route('logout') }}" method="POST" class="d-none">

                                        @csrf

                                    </form>

                                </div>

                            </li>

                        @endguest

                    </ul>

                </div>

            </div>

        </nav>

        <main class="py-4">

            @yield('content')

        </main>

    </div>

</body>

</html>

@extends('layouts.app')

@section('content')

<div class="container">

    <div class="row justify-content-center">

        <div class="col-md-8">

        <div class="text-center mb-4">

                <!-- Logo 图片 -->

                <img src="{{ asset('jhun.png') }}" alt="Logo" style="max-width: 150px; height: auto;">

            </div>

            <div class="card">

                <div class="card-header">{{ \_\_('欢迎进入江汉大学授权认证中心') }}</div>

                <div class="card-body">

                    @if (session('status'))

                        <div class="alert alert-success" role="alert">

                            {{ session('status') }}

                        </div>

                    @endif

                    {{ \_\_('登陆成功!') }}

                </div>

            </div>

        </div>

    </div>

</div>

@endsection

<!DOCTYPE html>

<html>

<head>

    <title>江汉大学授权中心</title>

    <!-- Favicon -->

    <link rel="icon" href="{{ asset('jhun.png') }}" type="image/x-icon">

    <meta charset="UTF-8">

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

</head>

<body style="font-family: Arial, sans-serif; background-color: #f5f5f5; color: #333; margin: 0; padding: 0;">

<header style="display: flex; justify-content: center; gap: 20px; padding: 20px; background-color: #ffffff; box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);">

    <a href="{{ route('login') }}" style="text-decoration: none; color: #007bff; padding: 10px 20px; border-radius: 4px; transition: background-color 0.3s;"

       onmouseover="this.style.backgroundColor='#e2e6ea';" onmouseout="this.style.backgroundColor='';">

       登录

    </a>

    <a href="{{ route('register') }}" style="text-decoration: none; color: #007bff; padding: 10px 20px; border-radius: 4px; transition: background-color 0.3s;"

       onmouseover="this.style.backgroundColor='#e2e6ea';" onmouseout="this.style.backgroundColor='';">

       注册

    </a>

</header>

<div id="app" style="display: flex; flex-direction: column; align-items: center; max-width: 1200px; margin: 20px auto; padding: 20px;">

    <!-- Center Logo -->

    <img src="{{ asset('jhun.png') }}" alt="Logo" style="max-width: 150px; height: auto; margin-bottom: 20px;">

    <h1 style="text-align: center; color: #333; font-size: 2rem; margin-top: 20px;">

        欢迎进入江汉大学授权认证中心

    </h1>

</div>

</body>

</html>

/\* Home.css \*/

/\* 页面容器 \*/

.container {

    display: flex;

    flex-direction: column;

    align-items: center;

    padding: 20px;

    background-color: #f9f9f9;

    min-height: 100vh;

}

/\* 标题样式 \*/

h1 {

    font-size: 2rem;

    margin-bottom: 20px;

    color: #333;

}

/\* 表格样式 \*/

table {

    width: 100%;

    border-collapse: collapse;

    margin-top: 20px;

    background-color: #fff;

    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

/\* 表头样式 \*/

th {

    background-color: #007bff;

    color: #fff;

    padding: 10px;

    text-align: left;

}

/\* 表格单元格样式 \*/

td {

    padding: 10px;

    border-bottom: 1px solid #ddd;

}

/\* 表格行的交替背景色 \*/

tr:nth-child(even) {

    background-color: #f2f2f2;

}

/\* 表格行 hover 样式 \*/

tr:hover {

    background-color: #f1f1f1;

}

/\* 错误信息样式 \*/

.error {

    color: #d9534f;

    font-size: 1rem;

    margin-top: 20px;

}

/\* 加载状态样式 \*/

.loading {

    font-size: 1rem;

    margin-top: 20px;

}

/\* 按钮样式 \*/

.buttons {

    margin-bottom: 20px;

}

.button {

    background-color: #007bff;

    color: #fff;

    border: none;

    border-radius: 4px;

    padding: 10px 20px;

    margin: 5px;

    cursor: pointer;

    font-size: 1rem;

}

.button:hover {

    background-color: #0056b3;

}

.button.delete {

    background-color: #dc3545;

}

.button.delete:hover {

    background-color: #c82333;

}

.container {

    padding: 20px;

}

.buttons {

    margin-bottom: 20px;

}

.button {

    padding: 10px 20px;

    margin-right: 10px;

    border: none;

    border-radius: 5px;

    cursor: pointer;

}

.button.delete {

    background-color: red;

    color: white;

}

.button.cancel {

    background-color: gray;

    color: white;

}

.form-container {

    margin-bottom: 20px;

    padding: 20px;

    border: 1px solid #ddd;

    border-radius: 5px;

}

.form-group {

    margin-bottom: 15px;

}

.form-group label {

    display: block;

    margin-bottom: 5px;

}

.form-group input {

    width: 100%;

    padding: 8px;

    border: 1px solid #ddd;

    border-radius: 5px;

}

.navigation-buttons {

    position: absolute;

    top: 10px;

    left: 10px;

    display: flex;

    gap: 10px;

}

.nav-button {

    padding: 10px 20px;

    font-size: 16px;

    cursor: pointer;

}

.button {

    padding: 10px 20px;

    font-size: 16px;

    cursor: pointer;

}

/\* 其他样式... \*/

/\* User.css \*/

/\* .container {

    padding: 20px;

}

table {

    width: 100%;

    border-collapse: collapse;

}

table, th, td {

    border: 1px solid #ddd;

}

th, td {

    padding: 8px;

    text-align: left;

}

th {

    background-color: #f4f4f4;

} \*/

/\* 样式文件，例如 auth-form.css \*/

.auth-form {

    max-width: 600px;

    margin: 0 auto;

    padding: 20px;

    border-radius: 8px;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

    background-color: #f9f9f9;

}

.auth-form h2 {

    text-align: center;

    margin-bottom: 20px;

    color: #333;

}

.form-group {

    margin-bottom: 15px;

}

.form-group label {

    display: block;

    margin-bottom: 5px;

    font-weight: bold;

    color: #555;

}

.form-group select {

    width: 100%;

    padding: 10px;

    border: 1px solid #ddd;

    border-radius: 4px;

    font-size: 16px;

    box-sizing: border-box;

    background-color: #fff;

    cursor: pointer;

}

.form-group select:focus {

    border-color: #007bff;

    outline: none;

}

.submit-button {

    display: block;

    width: 100%;

    padding: 10px;

    border: none;

    border-radius: 4px;

    background-color: #007bff;

    color: #fff;

    font-size: 16px;

    cursor: pointer;

    transition: background-color 0.3s ease;

}

.submit-button:hover {

    background-color: #0056b3;

}

/\* src/pages/Login.module.css \*/

.container {

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh; /\* 使容器高度为视口高度 \*/

    background-color: #f0f2f5; /\* 背景色（可选） \*/

  }

  .loginForm {

    width: 100%;

    max-width: 400px; /\* 设置表单最大宽度 \*/

    padding: 20px;

    border: 1px solid #ddd;

    border-radius: 8px;

    background-color: #fff;

    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1); /\* 添加阴影（可选） \*/

  }

  .formGroup {

    margin-bottom: 15px;

  }

  .formGroup label {

    display: block;

    margin-bottom: 5px;

  }

  .formGroup input {

    width: 100%;

    padding: 8px;

    border: 1px solid #ddd;

    border-radius: 4px;

  }

  .formGroup button {

    width: 100%;

    padding: 10px;

    border: none;

    border-radius: 4px;

    background-color: #007bff;

    color: white;

    font-size: 16px;

    cursor: pointer;

  }

  .formGroup button:hover {

    background-color: #0056b3;

  }

  .message {

    margin-top: 15px;

    color: #ff0000; /\* 错误消息颜色 \*/

  }

  .formGroup a {

    display: block;

    margin-top: 10px;

    text-align: center;

    color: #007bff;

    text-decoration: none;

}

.formGroup a:hover {

    text-decoration: underline;

}

<?php

namespace App\Http\Controllers;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class AuthController extends Controller

{

    public function register(Request $request)

    {

        // 使用 Validator 进行验证

        $validator = Validator::make($request->all(), [

            'name' => 'required',

            'email' => 'required',

            'password' => 'required',

        ]);

        // 检查验证是否失败

        if ($validator->fails()) {

            return response()->json(['errors' => $validator->errors()], 422);

        }

        // dd(1);

        // 创建用户

        $user = User::create([

            'name' => $request->name,

            'email' => $request->email,

            'password' => $request->password,

        ]);

        return response()->json(['message' => 'User successfully registered', 'user' => $user], 201);

    }

    public function login(Request $request)

    {

        // 验证请求数据

        $validator = Validator::make($request->all(), [

            'email' => 'required|email',

            'password' => 'required',

        ]);

        if ($validator->fails()) {

            return response()->json(['errors' => $validator->errors()], 422);

        }

        $credentials = $request->only('email', 'password');

        $user = User::where('email', $credentials['email'])->first();

        return response()->json($user, 200);

    }

}