var express = require('express');

var router = express.Router();

//连接数据库

const connection = require('./js/connect');

//引入jwt

const jwt = require('jsonwebtoken');

//引入express-jwt

const expressJwt = require('express-jwt')

//设置响应头

router.all('\*',(req,res,next)=>{

// 设置响应头解决跨域

res.setHeader("Access-Control-Allow-Origin", "\*");

res.setHeader("Access-Control-Allow-Headers", "authorization"); // 允许通过头部信息authorization 携带token

// 放行

next();

})

/\*---------验证token合法性------\*/

//准备一个签名（密钥）

const secretKey = 'qikesystem';

//验证token的合法性，设置需要保护的API

router.use(expressJwt({

secret:secretKey

}).unless({

path:['/login/checklogin']//排除掉这个地址，其他的url都需要验证（其他的请求都要带上token，才能获得数据，否则不能得到数据）

}));

//路由器拦截,校验token失败时的处理

router.use((err,req,res,next)=>{

// 如果前端发送的请求没有token或者是错误的token就抛出错误

if(err.name==="UnauthorizedError"){

// 如果有过期时间，就响应前端状态码

res.status(401).send({code:2,msg:"登录过期，请重新登录！"})

}

})

/\*---------验证token合法性------\*/

//增加客户

router.post('/clientadd',(req,res)=>{

//接收传过来的数据

let { name,sex,salesStatus,courseData,tel,school,classData,courseStatus,salesMan } = req.body;

console.log(req.body);

// 写sql

const sqlStr = `insert into clientdata(name,sex,salestatus,coursedata,tel,school,classdata,coursestatus,saleman ) values('${name}','${sex}','${salesStatus}','${courseData}','${tel}','${school}','${classData}','${courseStatus}','${salesMan}') `;

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

if(data.affectedRows>0){

res.send({code:0,msg:"添加数据成功"});

}else {

res.send({code:0,msg:"添加数据失败"});

}

})

})

//客户管理

router.get('/clientmanage',(req,res)=>{

//接收前端传过来的参数

let {pageSize,currentPage,keyword}=req.query;

//准备sql

let sqlStr = "select \* from clientdata where 1=1";

//如果有关键字

if(keyword){

sqlStr+=` and(name like '%${keyword}%')`;

}

//拼接降序排列

sqlStr+=' order by create\_time desc';

//执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

// 获得总条数

let total=data.length;

// 跳过多少条

let n=(currentPage-1)\*pageSize;

// 每页显示多少条

let m=pageSize;

// 写sql

sqlStr+=` limit ${n},${m}`;

console.log(sqlStr);

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

// 发送数据回前端

res.send({total,data});

})

})

})

//删除数据

router.get('/clientdel',(req,res)=>{

// 获取前端传过来的参数

let {id}=req.query;

// 写sql

const sqlStr=`delete from clientdata where id='${id}'`;

console.log(sqlStr);

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

if(data.affectedRows>0){

res.send({code:0,msg:"删除数据成功"})

}else {

res.send({code:1,msg:"删除数据失败"})

}

})

})

//回填编辑数据

router.get('/clientedit',(req,res)=>{

// 接收参数

let {id}=req.query;

// 写sql

const sqlStr=`select \* from clientdata where id=${id}`;

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

res.send(data);

})

})

//保存数据

router.post('/saveclient',(req,res)=>{

// 接收参数

let { id,name,sex,salesStatus,courseData,tel,school,classData,courseStatus,salesMan } = req.body;

// 写sql

const sqlStr = `update clientdata set name="${name}",sex="${sex}",salestatus="${salesStatus}",coursedata="${courseData}",tel="${tel}",school="${school}",classdata="${classData}",coursestatus="${courseStatus}",saleman="${salesMan}" where id=${id}`;

console.log(sqlStr);

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

if(data.affectedRows>0){

res.send({code:0,msg:"修改数据成功"});

}else {

res.send({code:0,msg:"修改数据失败"});

}

})

})

//批量删除

router.get('/batchdel',(req,res)=>{

// 接收参数

let {ids}=req.query;

// 写sql

const sqlStr =`delete from clientdata where id in (${ids})`;

// 执行sql

connection.query(sqlStr,(err,data)=>{

if(err) throw err;

if(data.affectedRows>0){

res.send({code:0,msg:"批量删除成功"})

}else {

res.send({code:1,msg:"批量删除失败"})

}

})

})

module.exports = router;