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", "Content-Type, 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------\*/

//增加员工

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

//接收传过来的数据

let { account,password,name,sex,birthDate,entryDate,staffGrounp,tel } = req.body;

// 默认头像

let imgurl='/upload/default.jpg';

// 写sql

const sqlStr = `insert into staffdata(account,password ,name ,sex ,birthdate ,entrydate ,staffgrounp ,tel,imgurl ) values('${account}','${password}','${name}','${sex}','${birthDate}','${entryDate}','${staffGrounp}','${tel}','${imgurl}') `;

// 执行sql

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

if(err) throw err;

//影响条数大于0，返回添加成功

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"});

}

})

})

//员工管理

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

const sqlStr = "select \* from staffdata order by create\_time desc";

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

if(err) throw err;

res.send(data);

})

})

//删除员工

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

// 接收id

let { id } = req.query;

// 写sql

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

// 执行sql

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

if(err) throw err;

// 判断删除状态

if(data.affectedRows>0){

res.send({code:0,msg:"删除成功"});

}else {

res.send({code:1,msg:"删除失败"});

}

})

})

//修改员工数据

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

// 接收前端传过来的参数

let { id } = req.query;

// 写sql

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

// 执行sql

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

if(err) throw err;

// 发送数据回前端

res.send(data);

})

})

//保存员工数据

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

// 接收参数

let { id,account,name,staffgrounp,sex,tel,birthdate,entrydate }=req.body;

// 写sql

const sqlStr = `update staffdata set account="${account}",name="${name}",staffgrounp="${staffgrounp}",sex="${sex}",account="${account}",tel="${tel}",birthdate="${birthdate}",entrydate="${entrydate}" where id=${id}`;

// 执行sql

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

if(data.affectedRows>0){

res.send({code:0,msg:"修改数据成功"});

}else {

res.send({code:1,msg:"修改数据失败"});

}

})

})

//批量删除数据

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

//接收前端发送的数据

let { id } = req.query;

// 写sql

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

// 执行sql

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

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"})

}

})

})

//根据分页信息返回员工列表

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

// 接收前端传过来的参数

let { pageSize,currentPage } = req.query;

//写sql

let sqlStr = `select \* from staffdata order by create\_time desc`;

//执行sql

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

if(err) throw err;

// 计算总条数

let total = data.length;

let n = (currentPage-1)\*pageSize;//跳过多少条

let m = pageSize;//每页展示多少条

// 拼接sql

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

// 执行sql

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

if(err) throw err;

res.send({data,total})

})

})

})

//获取相关员工信息

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

// 获得老师信息

// 写sql

let sqlStr = `select name from staffdata where staffgrounp = "老师"`;

// 执行sql

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

if(err) throw err;

// 接收老师数据

let teacher = data;

// 获得销售人员信息

sqlStr=`select name from staffdata where staffgrounp = "销售人员"`;

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

if(err) throw err;

// 接收销售人员数据

let saleman = data;

// 传送数据回前端

res.send({teacher,saleman})

})

})

})

//验证账号是否存在

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

// 接收参数

let { account } = req.query;

// 写sql

const sqlStr = `select \* from staffdata where account='${account}'`;

// 执行sql

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

if(err) throw err;

// 判断数组中是否有值

if(!data.length){

res.send({code:0,msg:""})

}else {

res.send({code:1,msg:"此账户太受欢迎了，请重新换一个！"})

}

})

})

//获取原密码

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

// 获取当前账号密码

res.send(req.user.password);

})

//修改原密码

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

// 接收参数

let {newPassword}=req.body;

// 获得当前账户id

let id = req.user.id;

// 写sql

const sqlStr=`update staffdata set password='${newPassword}' where id='${id}'`;

// 执行sql

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

if(err) throw err;

if(data.affectedRows>0){

res.send({code:0,msg:"修改密码成功，请重新登录"})

}else {

res.send({code:1,msg:"修改失败"})

}

})

})

//获取员工信息

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

let {id} =req.user;

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

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

if(err) throw err;

res.send(data[0])

})

})

/\* ----------------- 上传后端配置 开始 -------------- \*/

// 引入multer

const multer = require('multer');

// 配置上传到服务器放置的目录 和 重命名

const storage = multer.diskStorage({

destination: 'public/upload', // 图片上传到服务器的这个目录

// 图片重命名

filename (req, file, cb) {

var fileFormat =(file.originalname).split("."); // haha.jpg => ['haha', 'jpg']

// 获取时间戳

var filename = new Date().getTime();

// 124354654 + "." + jpg

cb(null, filename + "." + fileFormat[fileFormat.length - 1]);

}

})

// 上传对象

const upload = multer({

storage,

});

/\* ----------------- 上传后端配置 结束-------------- \*/

//头像上传

router.post('/uploadavatar',upload.single('file'),(req,res)=>{

// 获取文件名

let filename=req.file.filename;

// 拼接路径

let path=`/upload/${filename}`;

const sqlStr=`update staffdata set imgurl='${path}' where id=${req.user.id}`;

// 执行sql

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

if (err) throw err;

if (data.affectedRows > 0) {

res.send({code: 0, reason: "头像修改成功!", path})

} else {

res.send({code: 1, reason: "头像修改失败"})

}

})

})

//权限设置

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

//获取当前用户组

let usergroup = req.user.staffgrounp;

//定义菜单

let menus=[

// 销售管理

{

icoClass:"el-icon-phone-outline",

title:"销售管理",

role:["超级管理员","销售人员","店长","咨询师"],

children:[

{path:"/home/salesmanage",subtitle:"客户管理"},

{path:"/home/clientadd",subtitle:"新增客户"}

]

},

// 学生管理

{

icoClass:"el-icon-bell",

title:"学生管理",

role:["超级管理员","店长","咨询师"],

children:[

{path:"/home/studentmanage",subtitle:"学员管理"},

{path:"/home/studentadd",subtitle:"新增学员"}

]

},

// 班级管理

{

icoClass:"el-icon-view",

title:"班级管理",

role:["超级管理员","店长","咨询师"],

children:[

{path:"/home/classmanage",subtitle:"班级管理"},

{path:"/home/classadd",subtitle:"新增班级"}

]

},

// 课程管理

{

icoClass:"el-icon-tickets",

title:"课程管理",

role:["超级管理员","店长","咨询师","老师"],

children:[

{path:"/home/coursemanage",subtitle:"课程管理"},

{path:"/home/courseadd",subtitle:"新增课程"}

]

},

// 礼品管理

{

icoClass:"el-icon-view",

title:"礼品管理",

role:["超级管理员","店长","咨询师"],

children:[

{path:"/home/giftmanage",subtitle:"礼品详情"},

{path:"/home/giftadd",subtitle:"新增礼品"},

]

},

// 礼品兑换

{

icoClass:"el-icon-sold-out",

title:"礼品兑换",

role:["超级管理员","店长","咨询师"],

children:[

{path:"/home/giftexchangemanage",subtitle:"礼品兑换管理"},

{path:"/home/giftexchange",subtitle:"礼品兑换"},

]

},

// 员工管理

{

icoClass:"el-icon-service",

title:"员工管理",

role:["超级管理员","店长"],

children:[

{path:"/home/staffmanage",subtitle:"员工管理"},

{path:"/home/staffadd",subtitle:"新增员工"}

]

},

// 统计管理

{

icoClass:"el-icon-date",

title:"统计管理",

role:["超级管理员","店长",],

children:[

{path:"/home/salestotal",subtitle:"销售统计"},

{path:"/home/goodstotal",subtitle:"礼品兑换统计"}

]

},

// 系统管理

{

icoClass:"el-icon-info",

title:"系统管理",

role:["超级管理员","店长","咨询师","销售人员","老师"],

children:[

{path:"/home/systeminfo",subtitle:"系统信息"},

{path:"/home/passwordmodify",subtitle:"修改密码"}

]

}

];

let accessMenu=menus.filter(item=>item.role.includes(usergroup))//过滤菜单

res.send(accessMenu);

})

module.exports = router;