package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.satoken.stp.StpUtil;

import com.file.mybatisplus.core.conditions.query.QueryWrapper;

import com.file.mybatisplus.core.metadata.IPage;

import com.file.mybatisplus.core.toolkit.StringUtils;

import com.file.tdf.server.entity.User;

import com.file.tdf.server.entity.bo.\*;

import com.file.tdf.server.entity.vo.\*;

import com.file.tdf.server.exception.TdfException;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IUserService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.websocket.server.PathParam;

import java.util.List;

import java.util.Set;

import java.util.stream.Collectors;

@RestController

@RequestMapping("/user")

@CrossOrigin

@Tag(name = "UserController", description = "")

public class UserController {

/\*\*

\*登录功能

\*@param

\*@return

\*/

@Autowired private IUserService userService;

@Operation(summary = "")

@PostMapping("/login")

public R loginByUsernameAndPassword(@Valid @RequestBody LoginByUserNameAndPasswordFrom from) {

String userId = userService.loginByUserNameAndPassword(from);

if (StringUtils.isNotBlank(userId)) {

StpUtil.login(userId);

Set<String> userPermissions = userService.getUserPermissions(userId);

String tokenValue = StpUtil.getTokenInfo().getTokenValue();

return R.ok("").put("permissions", userPermissions).put("token", tokenValue);

} else {

return R.error("");

}

}

/\*\*

\*退出功能

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/logout")

@SaCheckLogin

public R logout() {

StpUtil.logout();

return R.ok("");

}

/\*\*

\*是否登录

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/isLogin")

public R isLogin() {

boolean login = StpUtil.isLogin();

return R.ok().put("isLogin", login);

}

@Operation(summary = "")

@GetMapping("/searchAllUser")

@SaCheckLogin

public R searchAllUser() {

List<UserInfoBo> userInfoBos = userService.searchAllUser();

return R.ok().put("data", userInfoBos).put("page", 1);

}

/\*\*

\*查看数据总览

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/ViewData")

@SaCheckPermission("USER:SELECT")

@SaCheckLogin

public R ViewData(@Valid @RequestBody SiftUserInfoVo siftUserInfoVo) {

IPage<UserInfoBo> userInfos = userService.getUserInfoBySift(siftUserInfoVo);

long total = userInfos.getTotal();

List<UserInfoBo> records = userInfos.getRecords();

return R.ok().put("data", records).put("total", total);

}

/\*\*

\*库房总数

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/warehouses")

@SaCheckLogin

public R warehouses() {

String userId = StpUtil.getLoginIdAsString();

UserInfoBo userInfo = userService.getUserInfo(userId);

Set<String> userPermissions = userService.getUserPermissions(StpUtil.getLoginIdAsString());

userInfo.setPermissions(userPermissions);

return R.ok().put("data", userInfo);

}

@Operation(summary = "")

@PostMapping("/insertUserInfo")

@SaCheckPermission("USER:INSERT")

@SaCheckLogin

public R insertUserInfo(@Valid @RequestBody InsertUserInfoVo insertUserInfoVo) {

int i = userService.insertUserInfo(insertUserInfoVo);

return R.ok();

}

/\*\*

\*入库档案总数

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/inbound ")

@SaCheckPermission("USER:SELECT")

@SaCheckLogin

public R inbound (@PathParam("id") String id) {

InsertUserInfoVo insertUserInfoVo = userService.searchUserInfoInInsert(id);

return R.ok().put("data", insertUserInfoVo);

}

/\*\*

\*档案影像数

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/archive")

@SaCheckPermission("USER:UPDATE")

@SaCheckLogin

public R archive(@Valid @RequestBody InsertUserInfoVo insertUserInfoVo) {

System.out.println(insertUserInfoVo);

int row = userService.updateUserInfo(insertUserInfoVo);

System.out.println(row);

if (row == 1) {

return R.ok("");

} else if (row == 0) {

return R.error("");

}

return R.ok();

}

@Operation(summary = "")

@PostMapping("/deleteUser")

@SaCheckLogin

@SaCheckPermission("USER:DELETE")

public R deleteUser(@RequestBody DeleteUserBo deleteUserBo) {

boolean b = userService.removeById(deleteUserBo.getId());

if (b) {

return R.ok("");

} else {

return R.ok("");

}

}

/\*\*

\*实体借阅数

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/borrowings")

@SaCheckLogin

public R borrowings(@PathParam("id") int id) {

List<SimpleClassesIntoUserInfoBo> simpleClassesIntoUserInfoBo =

userService.getSimpleClassesIntoUserInfoBo(id);

return R.ok().put("data", simpleClassesIntoUserInfoBo);

}

@Operation(summary = "")

@GetMapping("/getUserInfoToSift")

@SaCheckLogin

public R getUserInfoToSift() {

List<SimpleUserInfoToSift> simpleUserInfoToSiftList = userService.getSimpleUserInfoToSiftList();

return R.ok().put("data", simpleUserInfoToSiftList);

}

/\*\*

\*待入库数

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/warehoused")

@SaCheckLogin

public R warehoused(@Valid @RequestBody UpdateUserClassVo updateUserClassVo) {

int classId = updateUserClassVo.getClassId();

List<User> collect =

updateUserClassVo.getUserId().stream()

.map(

item -> {

User user = new User();

user.setId(item);

user.setClassesId(classId);

return user;

})

.collect(Collectors.toList());

System.out.println("----->" + collect);

boolean update = userService.updateBatchById(collect);

if (update) {

return R.ok("");

} else {

return R.error("");

}

}

/\*\*

\*电子档案借阅状态统计

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("electronic ")

@SaCheckLogin

public R electronic (@Valid @RequestBody DeleteUserBo deleteUserBo) {

User user = new User();

user.setId(deleteUserBo.getId());

user.setClassesId(0);

boolean update = userService.updateById(user);

if (update) {

return R.ok("");

} else {

return R.ok("");

}

}

/\*\*

\*实体借阅审批统计

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/Approval ")

@SaCheckLogin

public R Approval () {

List<SimpleTeacherList> simpleTeacherList = userService.getSimpleTeacherList();

return R.ok().put("data", simpleTeacherList);

}

@GetMapping("/getSimpleUserInfo")

@Operation(summary = "")

@SaCheckLogin

private R getSimpleUserInfo() {

List<SimpleUserInfo> list = userService.getSimpleUserInfo();

return R.ok().put("data", list);

}

/\*\*

\*实体档案入库数统计

\*@param

\*@return

\*/

@PostMapping("physical ")

@Operation(summary = "")

public R physical (

@Valid @RequestBody UpdatePrivateUserInfoVo updatePrivateUserInfoVo) {

System.out.println("----" + updatePrivateUserInfoVo);

User user = new User();

user.setId(updatePrivateUserInfoVo.getId());

if (StringUtils.isNotBlank(updatePrivateUserInfoVo.getEmail())) {

user.setEmail(updatePrivateUserInfoVo.getEmail());

}

if (StringUtils.isNotBlank(updatePrivateUserInfoVo.getPhone())) {

user.setPhone(updatePrivateUserInfoVo.getPhone());

}

boolean update = userService.updateById(user);

if (update) {

return R.ok("");

} else {

return R.error("");

}

}

/\*\*

\*档案影像入库数统计

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("received")

public R received(@Valid @RequestBody UpdateUserPasswordVo updateUserPasswordVo) {

QueryWrapper<User> userQueryWrapper = new QueryWrapper<>();

System.out.println("" + updateUserPasswordVo);

userQueryWrapper

.eq("phone", updateUserPasswordVo.getPhone())

.eq("num", updateUserPasswordVo.getNum())

.eq("email", updateUserPasswordVo.getEmail());

User one = userService.getOne(userQueryWrapper);

System.out.println(one);

if (one == null) {

return R.error("信息错误");

}

return R.ok("");

}

/\*\*

\*案卷管理

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("supervision ")

public R supervision (@Valid @RequestBody ModifyPasswordVo modifyPasswordVo) {

QueryWrapper<User> userQueryWrapper = new QueryWrapper<>();

System.out.println("---->" + modifyPasswordVo);

userQueryWrapper

.eq("phone", modifyPasswordVo.getPhone())

.eq("num", modifyPasswordVo.getNum())

.eq("email", modifyPasswordVo.getEmail());

User one = userService.getOne(userQueryWrapper);

if (one == null) {

throw new TdfException("");

}

userService.modifyPassword(modifyPasswordVo.getPassword(), one.getId(), one.getUsername());

return R.ok();

}

/\*\*

\*修改密码

\*@param

\*@return

\*/

@Operation(summary = "修改密码")

@PostMapping("modifyPrivatePassword")

public R modifyPrivatePassword(@Valid @RequestBody ModifyPrivatePassword modifyPrivatePassword) {

String loginIdAsString = StpUtil.getLoginIdAsString();

User byId = userService.getById(loginIdAsString);

userService.modifyPassword(

modifyPrivatePassword.getPassword(), byId.getId(), byId.getUsername());

StpUtil.logout(byId.getId());

return R.ok();

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.satoken.annotation.SaMode;

import com.alibaba.fastjson.JSON;

import com.file.tdf.server.entity.filetorage;

import com.file.tdf.server.entity.bo.SimpleStoreBo;

import com.file.tdf.server.entity.vo.InsertStoreInfoVo;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IfiletorageService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import java.time.LocalDateTime;

import java.util.List;

@RestController

@RequestMapping("/file-storage")

@Tag(name = "filetorageController", description = "")

public class filetorageController {

@Autowired private IfiletorageService filetorageService;

@Operation(summary = "")

@GetMapping("/getStorageList")

@SaCheckLogin

@SaCheckPermission("FILE\_MEMORY:SELECT")

public R getStorageList() {

List<SimpleStoreBo> list = filetorageService.getStorageList();

return R.ok().put("data", list);

}

/\*\*

\*签收管理菜单

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/Sign-in")

@SaCheckPermission(

value = {"FILE\_MEMORY:INSERT", "FILE\_MEMORY:UPDATE"},

mode = SaMode.OR)

public R Sign-in(@Valid @RequestBody InsertStoreInfoVo insertStoreInfoVo) {

filetorage filetorage = new filetorage();

if (filetorage.getId() == null) {

filetorage.setStorageName(insertStoreInfoVo.getStorageName());

filetorage.setStorageLongitude(insertStoreInfoVo.getStorageLongitude());

filetorage.setStorageLatitude(insertStoreInfoVo.getStorageLatitude());

filetorage.setStorageAddress(insertStoreInfoVo.getStorageAddress());

filetorage.setCreateTime(LocalDateTime.now());

filetorage.setDeptId(insertStoreInfoVo.getDeptId());

boolean save = filetorageService.save(filetorage);

if (save) {

return R.ok("");

} else {

return R.error("");

}

} else {

filetorage.setStorageName(insertStoreInfoVo.getStorageName());

filetorage.setId(insertStoreInfoVo.getId());

filetorage.setStorageLongitude(insertStoreInfoVo.getStorageLongitude());

filetorage.setStorageLatitude(insertStoreInfoVo.getStorageLatitude());

filetorage.setStorageAddress(insertStoreInfoVo.getStorageAddress());

filetorage.setDeptId(insertStoreInfoVo.getDeptId());

boolean save = filetorageService.updateById(filetorage);

if (save) {

return R.ok("");

} else {

return R.error("");

}

}

}

/\*\*

\*展示案卷签收记录

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/records")

@SaCheckPermission("FILE\_MEMORY:DELETE")

public R records(@RequestParam("id") Integer id) {

Boolean records= filetorageService.deleteStoreInfoById(id);

if (delete) {

return R.ok("");

} else {

return R.error("");

}

}

@Operation(summary = "")

@GetMapping("/getStoreInfoById")

@SaCheckPermission("FILE\_MEMORY:SELECT")

public R getStoreInfoById(@RequestParam("id") Integer id) {

return R.ok().put("data", filetorageService.getById(id));

}

/\*\*

\*展示案卷签收记录

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("getStorageListByIds")

@SaCheckPermission("FILE\_MEMORY:SELECT")

public R getStorageListByIds(@RequestParam("ids") String ids) {

List<Integer> integers = JSON.parseArray(ids, Integer.class);

List<filetorage> list = filetorageService.getStorageListByIds(integers);

return R.ok().put("data", list);

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.tdf.server.entity.bo.InsertPermissionBo;

import com.file.tdf.server.entity.bo.PermissionInfoBo;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IPermissionService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import java.util.HashMap;

import java.util.List;

@RestController

@RequestMapping("/permission")

@Tag(name = "PermissionController", description = "")

public class PermissionController {

@Autowired private IPermissionService permissionService;

@Operation(summary = "")

@GetMapping("/getAllPermission")

@SaCheckLogin

@SaCheckPermission("PERMISSION:SELECT")

public R getAllPermission() {

List<PermissionInfoBo> permissionList = permissionService.getPermissionList();

return R.ok().put("data", permissionList);

}

@GetMapping("/getRolePermissionList")

@Operation(summary = "")

@SaCheckLogin

@SaCheckPermission("PERMISSION:SELECT")

public R getRolePermissionList() {

HashMap<String, List<InsertPermissionBo>> insertPermissionList =

permissionService.getInsertPermissionList();

return R.ok().put("data", insertPermissionList);

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.satoken.annotation.SaMode;

import com.file.tdf.server.entity.Role;

import com.file.tdf.server.entity.bo.RoleInfoBo;

import com.file.tdf.server.entity.bo.RolePermissionBo;

import com.file.tdf.server.entity.bo.SimpleRoleListBo;

import com.file.tdf.server.entity.vo.DeleteRoleVo;

import com.file.tdf.server.entity.vo.InsertRoleVo;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IPermissionService;

import com.file.tdf.server.service.IRoleService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.websocket.server.PathParam;

import java.time.LocalDateTime;

import java.util.Arrays;

import java.util.HashMap;

import java.util.List;

@RestController

@RequestMapping("/role")

@Tag(name = "RoleController", description = "")

public class RoleController {

@Autowired private IRoleService roleService;

@Autowired private IPermissionService permissionService;

@GetMapping("getRoleList")

@Operation(summary = "")

@SaCheckLogin

public R getRoleList() {

List<SimpleRoleListBo> list = roleService.getAllSimpleRoleListBo();

return R.ok().put("data", list);

}

@GetMapping("/getAllRoleList")

@Operation(summary = "")

@SaCheckLogin

@SaCheckPermission("PERMISSION:SELECT")

public R getAllRoleList() {

List<RoleInfoBo> roleList = roleService.getRoleList();

return R.ok().put("data", roleList);

}

/\*\*

\*收件编号

\*@param

\*@return

\*/

@GetMapping("/getRolePermission")

@Operation(summary = "")

@SaCheckLogin

public R getRolePermission(@PathParam("id") int id) {

HashMap<String, List<RolePermissionBo>> rolePermissionList =

permissionService.getRolePermissionList(id);

return R.ok().put("data", rolePermissionList);

}

/\*\*

\*交付人

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/Delivered")

@SaCheckLogin

@SaCheckPermission(

value = {"PERMISSION:UPDATE", "PERMISSION:INSERT"},

mode = SaMode.OR)

public R Delivered(@Valid @RequestBody InsertRoleVo insertRoleVo) {

Role role = new Role();

if (insertRoleVo.getId() == null) {

role.setRoleDesc(insertRoleVo.getRoleDesc());

role.setSystemic(false);

role.setRoleName(insertRoleVo.getRoleName());

role.setPermissions(Arrays.toString(insertRoleVo.getPermission()));

role.setDefaultPermissions("[]");

role.setCreateTime(LocalDateTime.now());

role.setUpdateTime(LocalDateTime.now());

boolean save = roleService.save(role);

if (save) {

R.ok("");

} else {

R.error("");

}

} else {

role.setId(insertRoleVo.getId());

role.setRoleDesc(insertRoleVo.getRoleDesc());

role.setUpdateTime(LocalDateTime.now());

role.setRoleName(insertRoleVo.getRoleName());

role.setPermissions(Arrays.toString(insertRoleVo.getPermission()));

boolean update = roleService.updateById(role);

System.out.println(role);

if (update) {

return R.ok("");

} else {

return R.error("");

}

}

return R.ok();

}

/\*\*

\*签收人

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/Signed ")

@SaCheckPermission("PERMISSION:DELETE")

public R Signed (@Valid @RequestBody DeleteRoleVo deleteRoleVo) {

Boolean delete = roleService.deleteById(deleteRoleVo.getId());

if (delete) {

R.ok("");

} else {

R.error("");

}

return R.ok();

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.satoken.annotation.SaMode;

import com.file.mybatisplus.core.metadata.IPage;

import com.file.tdf.server.entity.Dept;

import com.file.tdf.server.entity.bo.DeptInfoBo;

import com.file.tdf.server.entity.bo.SimpleDeptApproveFileBo;

import com.file.tdf.server.entity.vo.DeleteDeptByIdVo;

import com.file.tdf.server.entity.vo.InsertDeptVo;

import com.file.tdf.server.entity.vo.SiftDeptInfoVo;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IDeptService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import java.time.LocalDateTime;

import java.util.List;

@RestController

@RequestMapping("/dept")

@Tag(name = "DeptController", description = "")

public class DeptController {

@Autowired private IDeptService deptService;

@Operation(summary = "")

@GetMapping("/getAllDept")

@SaCheckLogin

@SaCheckPermission("DEPT:SELECT")

public R getAllDept() {

List<Dept> list = deptService.list();

return R.ok().put("data", list);

}

/\*\*

\*库房号

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/Warehouse ")

@SaCheckLogin

@SaCheckPermission("DEPT:SELECT")

public R Warehouse (@Valid @RequestBody SiftDeptInfoVo siftDeptInfoVo) {

IPage<DeptInfoBo> deptInfoList = deptService.getDeptInfoList(siftDeptInfoVo);

List<DeptInfoBo> records = deptInfoList.getRecords();

long total = deptInfoList.getTotal();

return R.ok().put("data", records).put("total", total);

}

/\*\*

\*库房名称

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/name")

@SaCheckLogin

@SaCheckPermission(

value = {"DEPT:UPDATE", "DEPT:INSERT"},

mode = SaMode.OR)

public R name(@Valid @RequestBody InsertDeptVo insertDeptVo) {

Dept dept = new Dept();

if (insertDeptVo.getId() == null) {

dept.setDeptName(insertDeptVo.getDeptName());

dept.setUpdateTime(LocalDateTime.now());

dept.setCreateTime(LocalDateTime.now());

boolean save = deptService.save(dept);

if (save) {

R.ok("");

} else {

R.error("");

}

} else {

dept.setId(insertDeptVo.getId());

dept.setDeptName(insertDeptVo.getDeptName());

dept.setUpdateTime(LocalDateTime.now());

boolean update = deptService.updateById(dept);

if (update) {

R.ok("");

} else {

R.error("");

}

}

return R.ok();

}

/\*\*

\*货架数

\*@param

\*@return

\*/

@Operation(summary = "")

@DeleteMapping("/shelves")

@SaCheckLogin

@SaCheckPermission("DEPT:DELETE")

public R shelves(@Valid @RequestBody DeleteDeptByIdVo deleteDeptByIdVo) {

boolean delete = deptService.deleteById(deleteDeptByIdVo.getId());

if (delete) {

return R.ok("");

} else {

return R.error("");

}

}

@Operation(summary = "")

@GetMapping("/getDeptApproveFile")

public R getDeptApproveFile() {

List<SimpleDeptApproveFileBo> list = deptService.getDeptApproveFile();

return R.ok().put("data", list);

}

}

package com.file.casting.manage.controller;

import com.file.satoken.stp.StpUtil;

import com.file.tdf.server.entity.File;

import com.file.tdf.server.entity.FileApprove;

import com.file.tdf.server.entity.User;

import com.file.tdf.server.entity.bo.FileApproveInfoBo;

import com.file.tdf.server.entity.vo.FailApprovalFileVo;

import com.file.tdf.server.entity.vo.InsertApproveFileVo;

import com.file.tdf.server.entity.vo.PassApprovalFileVo;

import com.file.tdf.server.exception.TdfException;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IFileApproveService;

import com.file.tdf.server.service.Ifileervice;

import com.file.tdf.server.service.IUserService;

import com.file.tdf.server.util.EmailSendUtil;

import io.swagger.v3.oas.annotations.Operation;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.transaction.annotation.Transactional;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.multipart.MultipartFile;

import javax.validation.Valid;

import java.time.LocalDateTime;

import java.util.List;

import java.util.Objects;

@RestController

@RequestMapping("/file-approve")

public class FileApproveController {

/\*\*

\*层格数

\*@param

\*@return

\*/

@Autowired IFileApproveService fileApproveService;

@Autowired IUserService userService;

@Autowired Ifileervice fileervice;

@Autowired EmailSendUtil emailSendUtil;

@Operation(summary = "")

@PostMapping("layers")

@Transactional

public R layers(

@Valid InsertApproveFileVo insertApproveFileVo, @RequestParam("file") MultipartFile file) {

try {

Boolean flag = fileApproveService.upLoadApproveFile(insertApproveFileVo, file);

if (flag) {

return R.ok("成功");

} else {

return R.error("");

}

} catch (Exception e) {

e.printStackTrace();

throw new TdfException("");

}

}

@Operation(summary = "")

@GetMapping("/getPrivateApproveFileList")

public R getPrivateApproveFileList() {

List<FileApproveInfoBo> list = fileApproveService.getPrivateApproveFileList();

return R.ok().put("data", list);

}

@Operation(summary = "")

@GetMapping("/cancelFileApprove")

@Transactional

public R cancelFileApprove(@RequestParam("fileId") String fileId) {

fileApproveService.cancelFileApprove(fileId, StpUtil.getLoginIdAsString());

return R.ok("");

}

/\*\*

\*已放案卷数

\*@param

\*@return

\*/

@Operation(summary = "")

@GetMapping("/placed")

public R placed(@RequestParam("deptId") Integer deptId) {

String userId = StpUtil.getLoginIdAsString();

User byId = userService.getById(userId);

if (!Objects.equals(byId.getDeptId(), deptId)

&& byId.getRoleId() != 1

&& byId.getRoleId() != 2) {

throw new TdfException(");

}

List<FileApproveInfoBo> list = fileApproveService.getDeptApproverList(deptId);

return R.ok().put("data", list);

}

/\*\*

\*归档年份

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("filing")

@Transactional

public R filing(@Valid @RequestBody PassApprovalFileVo passApprovalFileVo) {

boolean isDeptApproval = StpUtil.hasPermission("DEPT:APPROVAL");

boolean isApproval = StpUtil.hasPermission("DOCUMENT\_APPROVAL:APPROVAL");

if (!isApproval) {

if (!isDeptApproval) {

throw new TdfException("");

} else {

User userInfo = userService.getById(StpUtil.getLoginIdAsString());

FileApprove byId1 = fileApproveService.getById(passApprovalFileVo.getFileId());

User user = userService.getById(byId1.getFileOwner());

if (!Objects.equals(userInfo.getDeptId(), user.getDeptId())) {

throw new TdfException("");

}

}

}

FileApprove fileApprove = new FileApprove();

fileApprove.setfiletate(2);

fileApprove.setId(passApprovalFileVo.getFileId());

fileApprove.setFileApprover(StpUtil.getLoginIdAsString());

fileApprove.setFileCause("");

fileApprove.setUpdateTime(LocalDateTime.now());

fileApprove.setFinalTime(LocalDateTime.now());

boolean update = fileApproveService.updateById(fileApprove);

FileApprove byId = fileApproveService.getById(passApprovalFileVo.getFileId());

File file = new File();

file.setId(byId.getFileId());

file.setFileType("1");

fileervice.updateById(file);

if (update) {

FileApprove byId1 = fileApproveService.getById(passApprovalFileVo.getFileId());

User user = userService.getById(byId1.getFileOwner());

emailSendUtil.SendMsg("", user.getEmail(), "审批成功");

return R.ok("");

} else {

R.error("");

}

return R.ok();

}

/\*\*

\*档案入库

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("warehousing")

public R warehousing(@Valid @RequestBody FailApprovalFileVo failApprovalFileVo) {

boolean isDeptApproval = StpUtil.hasPermission("DEPT:APPROVAL");

boolean isApproval = StpUtil.hasPermission("DOCUMENT\_APPROVAL:APPROVAL");

if (!isApproval) {

if (!isDeptApproval) {

throw new TdfException("");

} else {

User userInfo = userService.getById(StpUtil.getLoginIdAsString());

FileApprove byId1 = fileApproveService.getById(failApprovalFileVo.getFileId());

User user = userService.getById(byId1.getFileOwner());

if (!Objects.equals(userInfo.getDeptId(), user.getDeptId())) {

throw new TdfException("");

}

}

}

FileApprove fileApprove = new FileApprove();

fileApprove.setfiletate(3);

fileApprove.setId(failApprovalFileVo.getFileId());

fileApprove.setFileApprover(StpUtil.getLoginIdAsString());

fileApprove.setFileCause(failApprovalFileVo.getCause());

fileApprove.setUpdateTime(LocalDateTime.now());

boolean update = fileApproveService.updateById(fileApprove);

if (update) {

FileApprove byId1 = fileApproveService.getById(failApprovalFileVo.getFileId());

User user = userService.getById(byId1.getFileOwner());

emailSendUtil.SendMsg("", user.getEmail(), "");

return R.ok("");

} else {

R.error("");

}

return R.ok();

}

/\*\*

\*入库记录

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("records")

@Transactional

public R records(

@Valid InsertApproveFileVo insertApproveFileVo, @RequestParam("file") MultipartFile file) {

try {

fileervice.deleteById(insertApproveFileVo.getFileId());

Boolean flag = fileApproveService.reUpLoadApproveFile(insertApproveFileVo, file);

if (flag) {

return R.ok("成功");

} else {

return R.error("");

}

} catch (Exception e) {

e.printStackTrace();

throw new TdfException("上传");

}

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.tdf.server.entity.FileBucket;

import com.file.tdf.server.entity.vo.DeleteBucketVo;

import com.file.tdf.server.entity.vo.InsertFileBucketVo;

import com.file.tdf.server.exception.TdfException;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IFileBucketService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import java.util.List;

@RestController

@RequestMapping("/file-bucket")

@Tag(name = "FileBucketController", description = "")

public class FileBucketControllel {

/\*\*

\*档案入库时间

\*@param

\*@return

\*/

@Autowired private IFileBucketService fileBucketService;

@PostMapping("/createBucket")

@Operation(summary = "")

@SaCheckPermission("Filetime")

public R Filetime(@Valid @RequestBody InsertFileBucketVo insertFileBucketVo) {

Boolean isCreate = fileBucketService.createBucket(insertFileBucketVo);

if (isCreate) {

return R.ok("");

} else {

throw new TdfException("");

}

}

@PostMapping("/deleteBucket")

@Operation(summary = "")

@SaCheckPermission("FILEFOLAD:DELETE")

public R deleteBucket(@Valid @RequestBody DeleteBucketVo deleteBucketVo) {

fileBucketService.deleteBucket(deleteBucketVo.getId());

return R.ok("");

}

@Operation(summary = "")

@GetMapping("/getBucketInfo")

@SaCheckLogin

public R getBucketInfo() {

List<FileBucket> list = fileBucketService.getBucketInfo();

return R.ok().put("data", list);

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.tdf.server.entity.FilePath;

import com.file.tdf.server.entity.bo.SimpleFilePathInfoBo;

import com.file.tdf.server.entity.vo.DeleteFileFoldervVo;

import com.file.tdf.server.entity.vo.InsertFileFolderVo;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IFilePathService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import java.util.HashMap;

import java.util.List;

@RestController

@RequestMapping("/file-path")

@Tag(name = "FilePathController", description = "")

public class FilePathController {

@Autowired private IFilePathService filePathService;

@Operation(summary = "")

@PostMapping("createFileFolder")

@SaCheckPermission("FILEFOLAD:INSERT")

public R createFileFolder(@Valid @RequestBody InsertFileFolderVo insertFileFolderVo) {

Boolean isCreate = filePathService.createFileFolder(insertFileFolderVo);

return R.ok();

}

/\*\*

\*入库人员

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("personnel")

@SaCheckPermission("FILEFOLAD:DELETE")

public R personnel(@Valid @RequestBody DeleteFileFoldervVo deleteFileFoldervVo) {

FilePath fileByPathCode = filePathService.findFileByPathCode(deleteFileFoldervVo.getId());

if (fileByPathCode != null) {

Boolean isCreate = filePathService.deleteFileFolder(fileByPathCode.getId());

return R.ok("");

}

return R.ok("");

}

@Operation(summary = "")

@GetMapping("/getFolderList")

@SaCheckPermission("FILEFOLAD:SELECT")

public R getFolderList() {

HashMap<String, List<SimpleFilePathInfoBo>> folderList = filePathService.getFolderList();

return R.ok().put("data", folderList);

}

@Operation(summary = "")

@GetMapping("/getFolderListByBucketId")

public R getFolderListByBucketId(@RequestParam("id") Integer id) {

HashMap<String, List<SimpleFilePathInfoBo>> folderList =

filePathService.getFolderListByBucketId(id);

return R.ok().put("data", folderList);

}

}

package com.file.casting.manage.controller;

import com.file.satoken.annotation.SaCheckLogin;

import com.file.satoken.annotation.SaCheckPermission;

import com.file.satoken.annotation.SaMode;

import com.file.mybatisplus.core.metadata.IPage;

import com.file.tdf.server.entity.Classes;

import com.file.tdf.server.entity.bo.DetailClassInfoBo;

import com.file.tdf.server.entity.bo.SimpleClassesBo;

import com.file.tdf.server.entity.vo.DeleteClasses;

import com.file.tdf.server.entity.vo.InsertClassesVo;

import com.file.tdf.server.entity.vo.SiftClassesInfoVo;

import com.file.tdf.server.exception.TdfException;

import com.file.tdf.server.response.R;

import com.file.tdf.server.service.IClassesService;

import io.swagger.v3.oas.annotations.Operation;

import io.swagger.v3.oas.annotations.tags.Tag;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import javax.validation.Valid;

import javax.websocket.server.PathParam;

import java.time.LocalDateTime;

@RestController

@RequestMapping("/classes")

@Tag(name = "ClassesController", description = "")

public class ClassesController {

@Autowired private IClassesService classesService;

@Operation(summary = "")

@GetMapping("/getClassesList")

@SaCheckLogin

public R getClassesList() {

return R.ok().put("data", classesService.getSimpleClassesInfoBo());

}

@Operation(summary = "")

@PostMapping("/getClassesList")

@SaCheckLogin

@SaCheckPermission("CLASSES:SELECT")

public R getClassesList(@RequestBody SiftClassesInfoVo siftClassesInfoVo) {

IPage<SimpleClassesBo> classList = classesService.getClassList(siftClassesInfoVo);

return R.ok().put("data", classList.getRecords()).put("total", classList.getTotal());

}

@Operation(summary = "")

@GetMapping("/getEarlyYear")

@SaCheckLogin

public R getEarlyYear() {

int earlyYear = classesService.getEarlyYear();

return R.ok().put("data", earlyYear);

}

/\*\*

\*档案编号

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/FileNo")

@SaCheckPermission(

value = {"CLASSES:UPDATE", "CLASSES:INSERT"},

mode = SaMode.OR)

public R FileNo(@Valid @RequestBody InsertClassesVo insertClassesVo) {

Classes classes = new Classes();

if (insertClassesVo.getId() == null) {

classes.setClassStatu(insertClassesVo.getClassStatu());

classes.setCreateTime(LocalDateTime.now());

classes.setUpdateTime(LocalDateTime.now());

classes.setClassName(insertClassesVo.getClassName());

classes.setClassYear(insertClassesVo.getClassYear());

classes.setDeptId(insertClassesVo.getDeptName());

classes.setProfessionalId(insertClassesVo.getProfessionalName());

classes.setUserList("[]");

boolean save = classesService.save(classes);

if (save) {

return R.ok("");

} else {

return R.error("");

}

} else {

classes.setClassStatu(insertClassesVo.getClassStatu());

classes.setUpdateTime(LocalDateTime.now());

classes.setId(insertClassesVo.getId());

classes.setClassName(insertClassesVo.getClassName());

classes.setClassYear(insertClassesVo.getClassYear());

classes.setDeptId(insertClassesVo.getDeptName());

classes.setProfessionalId(insertClassesVo.getProfessionalName());

boolean save = classesService.updateById(classes);

if (save) {

return R.ok(");

} else {

return R.error("");

}

}

}

/\*\*

\*收件编号

\*@param

\*@return

\*/

@Operation(summary = "")

@PostMapping("/Receipt ")

@SaCheckLogin

@SaCheckPermission("CLASSES:DELETE")

public R Receipt (@Valid @RequestBody DeleteClasses deleteClasses) {

boolean delete = classesService.deleteById(deleteClasses.getId());

if (delete) {

return R.ok("");

} else {

throw new TdfException("");

}

}

@Operation(summary = "")

@GetMapping("/getClassesById")

@SaCheckLogin

@SaCheckPermission("CLASSES:SELECT")

public R getClassesById(@PathParam("id") int id) {

Classes byId = classesService.getById(id);

return R.ok().put("data", byId);

}

@Operation(summary = "")

@GetMapping("/getDetailedClassesIngo")

@SaCheckLogin

@SaCheckPermission("CLASSES:SELECT")

public R getDetailedClassesIngo(@PathParam("id") int id) {

DetailClassInfoBo detailClassInfoBo = classesService.getDetailClassInfoBo(id);

return R.ok().put("data", detailClassInfoBo);

}

}

package com.file.casting.manage.controller;

import com.file.mybatisplus.core.metadata.IPage;

import com.file.mybatisplus.extension.plugins.pagination.Page;

import com.file.common.Result;

import com.file.entity.po.Audit;

import com.file.entity.vo.AuditVo;

import com.file.service.ArchiveTypeService;

import com.file.service.AuditService;

import com.file.service.DictionaryService;

import com.file.utils.ListToStringUtil;

import lombok.Data;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.util.Assert;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/verify")

@Slf4j

public class ArchiveTypeController {

@Autowired

private ArchiveTypeService archiveTypeService;

@Autowired

private DictionaryService dictionaryService;

@Autowired

private AuditService auditService;

@GetMapping("/list")

public Result list(Integer page, Integer size) {

Assert.notNull(page, "");

Assert.notNull(size, "");

IPage<AuditVo> list = archiveTypeService.list(new Page(page, size));

return Result.success(list);

}

/\*\*

\*删除id

\*@param

\*@return

\*/

@PostMapping("/delete")

public Result delete(Integer id) {

Assert.notNull(id, "");

boolean delete = archiveTypeService.delete(id);

if (delete) {

return Result.success();

} else {

return Result.fail();

}

}

@PostMapping("/deleteBatch")

public Result deleteBatch(@RequestBody IdList ids) {

Assert.notNull(ids, "ids不能空");

boolean b = archiveTypeService.myDeleteBatch(ListToStringUtil.convert(ids.getIds()));

if (b) {

return Result.success();

} else {

return Result.fail();

}

}

@GetMapping("/audit")

public Result audit(Long id) {

Assert.notNull(id, "id不能为空");

return auditService.passAudit(id);

}

/\*\*

\*提交

\*@param

\*@return

\*/

@PostMapping("/submit")

public Result submit(String dictValue,Long archiveId,String stuId) {

Assert.notNull(dictValue,"档案类型为空");

Assert.notNull(archiveId,"档案类型id为空");

Assert.notNull(stuId,"学生id为空");

Long dictCodeBydictValue = dictionaryService.getDictCodeBydictValue(dictValue);

Audit audit = new Audit();

audit.setArchiveTypeCode(dictCodeBydictValue);

audit.setArchiveId(archiveId);

audit.setStuId(stuId);

audit.setIsAudited(0);

boolean save = auditService.save(audit);

if(save){

return Result.success();

}

else{

return Result.fail();

}

}

}

@Data

class IdList {

private List<Long> ids;

}

package com.file.casting.manage.controller;

import com.file.mybatisplus.core.conditions.query.QueryWrapper;

import com.file.common.BaseRestController;

import com.file.common.Result;

import com.file.entity.po.Permission;

import com.file.entity.po.RolePermission;

import com.file.service.PermissionService;

import com.file.service.RolePermissionService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.util.Assert;

import org.springframework.util.StringUtils;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/permission")

public class PermissionController extends BaseRestController<PermissionService,Permission> {

@Autowired

RolePermissionService rolePermissionService;

@Override

protected Result delete(Permission permission) {

Assert.notNull(permission.getId(), "id不能为空");

boolean res = baseService.removeById(permission.getId());

if (!res) return Result.fail();

boolean permission\_id = rolePermissionService.remove(new QueryWrapper<RolePermission>()

.eq("permission\_id", permission.getId()));

if (!permission\_id) return Result.fail();

return Result.success();

}

@Override

protected Result deleteBatch(List<Long> ids) {

boolean res = baseService.removeByIds(ids);

if (!res) return Result.fail();

boolean permission\_id = rolePermissionService.remove(new QueryWrapper<RolePermission>()

.in("permission\_id", ids));

if (!permission\_id) return Result.fail();

return Result.success();

}

@GetMapping("/all")

public Result getAll(){

List<Permission> list = baseService.list();

return Result.success(list);

}

@Override

protected QueryWrapper<Permission> getSearchQueryWrapper(String searchValue, Permission permission) {

QueryWrapper<Permission> wrapper = new QueryWrapper<>();

if(StringUtils.isEmpty(searchValue)){

wrapper.like("title",searchValue);

}

return wrapper;

}

}

package com.file.casting.manage.controller;

import com.file.mybatisplus.extension.plugins.pagination.Page;

import com.file.common.BaseRestController;

import com.file.common.Result;

import com.file.entity.po.Role;

import com.file.entity.vo.RoleVo;

import com.file.service.RoleService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.util.Assert;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/role")

public class RoleController{

@Autowired

RoleService roleService;

@GetMapping("/all")

public Result getAll(){

List<Role> list = roleService.list();

return Result.success(list);

}

@GetMapping("/vo/getByUid")

public Result getRoleByUid(Long uid){

List<RoleVo> listByUid = roleService.getListByUid(uid);

return Result.success(listByUid);

}

@GetMapping("/list")

public Result getList(Integer page,Integer size){

Assert.notNull(page,"页数不能为空");

Assert.notNull(size,"页大小不能为空");

Page list = roleService.getList(new Page(page, size));

return Result.success(list);

}

@PostMapping("/delete")

public Result deleteById(@RequestBody RoleVo roleVo){

Assert.notNull(roleVo.getId(),"id不能为空");

boolean b = roleService.deleteRole(roleVo.getId());

if(b){

return Result.success();

}else{

return Result.fail();

}

}

@PostMapping("/insert")

public Result insert(@RequestBody RoleVo roleVo){

boolean b = roleService.saveRoleVo(roleVo);

if(b){

return Result.success();

}else{

return Result.fail();

}

}

@PostMapping("/update")

public Result update(@RequestBody RoleVo roleVo){

Assert.notNull(roleVo.getId(),"id不能为空");

boolean b = roleService.updateRoleVo(roleVo);

if(b){

return Result.success();

}else{

return Result.fail();

}

}

}

package com.file.casting.manage.controller;

import com.file.mybatisplus.core.conditions.query.QueryWrapper;

import com.file.common.BaseRestController;

import com.file.common.Result;

import com.file.entity.po.StuInfo;

import com.file.service.StuInfoService;

import org.apache.shiro.util.Assert;

import org.springframework.util.StringUtils;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/stu\_info")

public class StuInfoController extends BaseRestController<StuInfoService, StuInfo> {

@GetMapping("/getByUid")

public Result getByUid(Long uid){

Assert.notNull(uid,"用户ID不能为空");

StuInfo stuInfo = baseService.getOne(new QueryWrapper<StuInfo>()

.eq("uid", uid));

return Result.success(stuInfo);

}

@GetMapping("/getByStuId")

public Result getByStuId(String stuId){

Assert.notNull(stuId,"");

StuInfo stuInfo = baseService.getOne(new QueryWrapper<StuInfo>()

.eq("stu\_id", stuId));

if(stuInfo!=null){

return Result.success(stuInfo);

}

return Result.fail("");

}

@Override

protected QueryWrapper<StuInfo> getSearchQueryWrapper(String searchValue, StuInfo stuInfo) {

if(!StringUtils.isEmpty(searchValue)){

QueryWrapper<StuInfo> wrapper=new QueryWrapper();

wrapper.like("stu\_id",searchValue);

return wrapper;

}else{

return super.getSearchQueryWrapper(searchValue, stuInfo);

}

}

}

package com.file.casting.manage.controller;

import com.file.mybatisplus.core.conditions.query.QueryWrapper;

import com.file.mybatisplus.extension.plugins.pagination.Page;

import com.file.common.BaseRestController;

import com.file.common.Result;

import com.file.entity.po.Role;

import com.file.entity.po.StuInfo;

import com.file.entity.po.User;

import com.file.entity.po.UserRole;

import com.file.entity.vo.UserVo;

import com.file.service.RoleService;

import com.file.service.UserRoleService;

import com.file.service.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.util.Assert;

import org.springframework.util.ResourceUtils;

import org.springframework.util.StringUtils;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/user")

public class UserController extends BaseRestController<UserService,User> {

@Autowired

UserRoleService userRoleService;

@Autowired

RoleService roleService;

@Override

protected Result insert(User user) {

boolean res = baseService.save(user);

if (!res) return Result.fail();

Role stuRole = roleService.getOne(new QueryWrapper<Role>()

.eq("role\_mark", "student"));

if(stuRole!=null){

UserRole userRole = new UserRole();

userRole.setUserId(user.getId());

userRole.setRoleId(stuRole.getId());

boolean save = userRoleService.save(userRole);

if (!save) return Result.fail();

}else{

return Result.fail("");

}

return Result.success();

}

/\*\*

\*档案管理

\*@param

\*@return

\*/

@PostMapping("/vo/management")

public Result management(@RequestBody UserVo userVo){

boolean b = baseService.updateById(userVo);

if(!b) return Result.fail();

if(userVo.getRoleId()!=null){

UserRole oldUserRole = userRoleService.getOne(new QueryWrapper<UserRole>()

.eq("user\_id", userVo.getId()));

if(oldUserRole==null){

UserRole userRole=new UserRole();

userRole.setRoleId(userVo.getRoleId());

userRole.setUserId(userVo.getId());

boolean save = userRoleService.save(userRole);

if(!save) return Result.fail();

}else{

oldUserRole.setRoleId(userVo.getRoleId());

boolean b1 = userRoleService.updateById(oldUserRole);

if(!b1) return Result.fail();

}

}

return Result.success();

}

/\*\*

\*档案信息

\*@param

\*@return

\*/

@GetMapping("/vo/Information")

public Result Information(Integer page, Integer size, String searchValue, UserVo userVo) {

Assert.notNull(page, "");

Assert.notNull(size, "");

QueryWrapper<UserVo> wrapper=new QueryWrapper();

if(!StringUtils.isEmpty(searchValue)){

wrapper.like("username",searchValue).or()

.like("real\_name",searchValue).or()

.like("role\_name",searchValue);

}

Page pageInfo = baseService.getVoList(new Page(page, size), wrapper);

return Result.success(pageInfo);

}

@Override

protected QueryWrapper<User> getSearchQueryWrapper(String searchValue, User user) {

QueryWrapper<User> wrapper=new QueryWrapper();

if(!StringUtils.isEmpty(searchValue)){

wrapper.like("username",searchValue).or()

.like("real\_name",searchValue);

return wrapper;

}

return wrapper;

}

}

package com.file.casting.manage.controller;

import com.file.stus.entity.GradeEntity;

import com.file.stus.entity.StuInformation;

import com.file.stus.resposistory.GradeResposistory;

import com.file.stus.resposistory.StuInformationReposistory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/GradeEntity")

public class GradeController {

@Autowired

private GradeResposistory gradeResposistory;

@PostMapping("/save")

public String save(@RequestBody GradeEntity gradeEntity){

GradeEntity result = gradeResposistory.save(gradeEntity);

if ( result!=null )

{

return "success";

}

else{

return "error";

}

}

@GetMapping("/findBycomame/{comame}")

public GradeEntity findBycomame(@PathVariable("comame") String comame){

return gradeResposistory.findBycomame(comame);

}

@PutMapping("/update")

public String update(@RequestBody GradeEntity gradeEntity){

GradeEntity result = gradeResposistory.save(gradeEntity);

if ( result != null )

{

return "success";

} else{

return "error";

}

}

}

package com.file.casting.manage.controller;

import com.file.stus.entity.User;

import com.file.stus.result.Result;

import com.file.stus.service.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.util.HtmlUtils;

@Controller

public class LoginController {

@Autowired

UserService userService;

/\*\*

\*收件编号

\*@param

\*@return

\*/

@CrossOrigin

@PostMapping(value = "/api/Receipt ")

@ResponseBody

public Result Receipt (@RequestBody User requestUser) {

String username = requestUser.getUsername();

username = HtmlUtils.htmlEscape(username);

User user = userService.get(username, requestUser.getPassword());

if (null == user) {

return new Result(400);

} else {

return new Result(200);

}

}

}

package com.file.casting.manage.controller;

import com.file.stus.entity.StuInformation;

import com.file.stus.resposistory.StuInformationReposistory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/StuInformation")

public class StuInformationController {

@Autowired

private StuInformationReposistory s;

@GetMapping("/findAll/{page}/{size}")

public Page<StuInformation> findAll(@PathVariable("page") Integer page,@PathVariable("size") Integer size){

Pageable pageable = PageRequest.of(page-1,size);

return s.findAll(pageable);

}

@PostMapping("/save")

public String save(@RequestBody StuInformation stu){

StuInformation result = s.save(stu);

if ( result!=null )

{

return "success";

}

else{

return "error";

}

}

/\*\*

\*档案等级

\*@param

\*@return

\*/

@GetMapping("/level/{sno}")

public StuInformation level(@PathVariable("sno") String sno){

return s.findBySno(sno);

}

@PutMapping("/update")

public String update(@RequestBody StuInformation stuInformation){

StuInformation result = s.save(stuInformation);

if ( result != null )

{

return "success";

} else{

return "error";

}

}

@DeleteMapping("/deleteById/{sno}")

public void deleteById(@PathVariable("sno") String sno){

s.deleteById(sno);

}

@DeleteMapping("/doData/{sno}")

public void doData(@PathVariable("sno") String sno){

s.doData(sno);

}

}

<template>

<div :style="!$route.meta.hiddenHeaderContent ? 'margin: -24px -24px 0px;' : null">

<page-header v-if="!$route.meta.hiddenHeaderContent" :title="pageTitle" :logo="logo" :avatar="avatar">

<slot slot="action" name="action"></slot>

<slot slot="content" name="headerContent"></slot>

<div slot="content" v-if="!this.$slots.headerContent && description">

<p style="font-size: 14px;color: rgba(0,0,0,.65)"></p>

<div class="link">

<template v-for="(link, index) in linkList">

<a :key="index" @click="() => { link.callback && link.callback() }">

<a-icon :type="link.icon" />

<span>{{ link.title }}</span>

</a>

</template>

</div>

</div>

<slot slot="extra" name="extra">

<div class="extra-img">

<img v-if="typeof extraImage !== 'undefined'" :src="extraImage"/>

</div>

</slot>

<div slot="pageMenu">

<div class="page-menu-search" v-if="search">

<a-input-search

style="width: 80%; max-width: 522px;"

placeholder=""

size="large"

enterButton="入库档案总数"

/>

</div>

<div class="page-menu-tabs" v-if="tabs && tabs.items">

@change="callback" :activeKey="activeKey"

<a-tabs :tabBarStyle="{margin: 0}" :activeKey="tabs.active()" @change="tabs.callback">

<a-tab-pane v-for="item in tabs.items" :tab="item.title" :key="item.key">档案影像数</a-tab-pane>

</a-tabs>

</div>

</div>

</page-header>

<div class="content">

<div class="page-header-index-wide">

<slot>

<keep-alive v-if="multiTab">

<router-view ref="content" />

</keep-alive>

<router-view v-else ref="content" />

</slot>

</div>

</div>

</div>

</template>

<script>

import { mapState } from 'vuex'

import PageHeader from '@/components/PageHeader'

export default {

name: 'PageView',

components: {

PageHeader

},

props: {

avatar: {

type: String,

default: null

},

title: {

type: [String, Boolean],

default: true

},

logo: {

type: String,

default: null

},

directTabs: {

type: Object,

default: null

}

},

data () {

return {

pageTitle: null,

description: null,

linkList: [],

extraImage: '',

search: false,

tabs: {}

}

},

computed: {

...mapState({

multiTab: state => state.app.multiTab

})

},

mounted () {

this.tabs = this.directTabs

this.getPageMeta()

},

updated () {

this.getPageMeta()

},

methods: {

getPageMeta () {

this.pageTitle = (typeof(this.title) === 'string' || !this.title) ? this.title : this.$route.meta.title

const content = this.$refs.content

if (content) {

if (content.pageMeta) {

Object.assign(this, content.pageMeta)

} else {

this.description = content.description

this.linkList = content.linkList

this.extraImage = content.extraImage

this.search = content.search === true

this.tabs = content.tabs

}

}

}

}

}

</script>

<style lang="less" scoped>

.content {

margin: 24px 24px 0;

.link {

margin-top: 16px;

&:not(:empty) {

margin-bottom: 16px;

}

a {

margin-right: 32px;

height: 24px;

line-height: 24px;

display: inline-block;

i {

font-size: 24px;

margin-right: 8px;

vertical-align: middle;

}

span {

height: 24px;

line-height: 24px;

display: inline-block;

vertical-align: middle;

}

}

}

}

.page-menu-search {

text-align: center;

margin-bottom: 16px;

}

.page-menu-tabs {

margin-top: 48px;

}

.extra-img {

margin-top: -60px;

text-align: center;

width: 195px;

img {

width: 100%;

}

}

.mobile {

.extra-img{

margin-top: 0;

text-align: center;

width: 96px;

img{

width: 100%;

}

}

}

</style>

<template>

<div id="code-home" class="manage-pos">

<tag-box v-show="showChild">

<template v-slot:code>

<div class="opentype-tag choose-bg-tag">实体借阅数</div>

</template>

</tag-box>

<div class="dep\_top" v-show="!showChild">

<div class="table-page-search-wrapper">

<a-form layout="inline">

<a-row>

<a-col :md="8" :sm="24">

<a-form-item label="电子档案借阅状态统计">

<a-input

v-model="queryParam.item\_type\_id"

clearable

placeholder="实体借阅审批统计"

></a-input>

</a-form-item>

</a-col>

<a-col :md="8" :sm="24">

<span class="table-edit-submitButtons">

<a-button type="primary" class="ml8" v-action:query @click="$refs.table.refresh(true)">待入库数</a-button>

<a-button class="ml8" @click="handleReset">实体档案入库数统计</a-button>

<a-button type="danger" class="ml8" v-action:add @click="openAddComp">档案影像入库数统计</a-button>

</span>

</a-col>

</a-row>

</a-form>

</div>

</div>

<div class="dep\_main" v-show="!showChild">

<s-table

ref="table"

size="default"

:row-key="(r,i) => i.toString()"

:columns="columns"

:data="loadData">

<span slot="action" slot-scope="t, r">

<a

v-action:edit

href="javascript:void(0);"

@click="editCode(r)"></a>

<a-divider type="vertical" />

<a v-action:delete href="javascript:void(0);" @click="del(r)"></a>

</span>

</s-table>

</div>

<transition name="slide-fade">

<div class="popContainer" v-if="showChild">

<edit-code

:open-type="openType"

:show-child.sync="showChild"

:form-data="formData"

@addCode="addCode"

@updateCode="updateCode"

></edit-code>

</div>

</transition>

</div>

</template>

<script>

import EditCode from './EditCode' // 查询组件

import { codeList, addCode, changeCode, deleteCode } from '@/api/leave'

import { TagBox } from '@/myComponents'

import { STable } from '@/components'

import \* as api from "@/api/systemAdmin";

export default {

name: 'CodeHome',

components: {

EditCode,

TagBox,

STable

},

data () {

return {

loading: false,

openType: 0,

formData: {},

showChild: false,

columns: [

{

title: '序号',

width: 50,

dataIndex: '',

key: 'rowIndex',

align: 'center',

customRender: (t, r, index) => {

return ((this.paginationInfo.pageNum - 1) \* this.paginationInfo.pageSize + index + 1)

}

},

{

title: '收件编号',

width: 90,

align: 'center',

dataIndex: 'item\_type\_id'

},

{

title: '交付人',

width: 90,

align: 'center',

dataIndex: 'item\_type\_desc'

},

{

title: '签收人',

width: 90,

align: 'center',

dataIndex: 'item\_desc'

},

{

title: '签收时间',

width: 90,

dataIndex: '',

scopedSlots: { customRender: 'action' },

align: 'center'

}

],

queryParam: {},

paginationInfo: null,

loadData: parameter => {

this.paginationInfo = parameter

return codeList(Object.assign(parameter, this.queryParam))

.then(res => {

if (res.code === 0) {

return res.data

} else {

this.$message.error(res.msg)

return {

list: []

}

}

})

}

}

},

created () {

},

methods: {

async addCode (obj) {

const addCodeData = await addCode(obj)

if (addCodeData.code === 0) {

this.$message.success(addCodeData.data)

this.$refs.table.refresh(true)

} else {

this.$message.warning(addCodeData.msg)

}

},

async updateCode (data) {

console.log('data', data)

const changeMsg = await changeCode(data)

if (changeMsg.code === 0) {

this.$message.success(changeMsg.msg)

this.$refs.table.refresh()

} else {

this.$message.warning(changeMsg.msg)

}

},

del (value) {

let \_this = this

this.$confirm({

title: '',

okText: '',

okType: 'danger',

cancelText: '',

onOk () {

deleteCode({

item\_type\_id: parseInt(value.item\_type\_id),

item\_val: value.item\_val

}).then(data => {

if (data.success) {

\_this.$message.success(data.data)

\_this.$refs.table.refresh()

} else {

\_this.$message.warning(data.msg)

}

}).catch(error => {}).finally(() => {

})

},

onCancel () {

}

})

},

handleReset () {

this.queryParam = {}

this.$refs.table.refresh(true)

},

openAddComp () {

this.showChild = true

this.openType = 0

},

queryCode(data) {

this.queryParam = data

this.$refs.table.refresh(true)

},

editCode (row) {

this.openType = 1

this.showChild = true

this.formData = row

},

}

}

</script>

<style scoped>

.el-table td,

.el-table th {

text-align: center;

}

.block {

margin-top: 5px;

}

.popContainer {

position: fixed;

display: flex;

top: 0;

left: 180px;

right: 0;

bottom: 0;

z-index: 1000;

background: rgba(16, 16, 16, 0.2);

}

</style>

<template>

<a-modal

title="库房信息"

:width="640"

:visible="visible"

@cancel="handleCancel"

@ok="handleCancel"

>

<a-upload

name="file"

:multiple="true"

:fileList="fileList"

:customRequest="myUpload"

:remove="myRemove"

:before-upload="beforeUpload"

@change="handleChange"

accept=".doc,.docx,.xls,.xlsx,.ppt,.pptx,.pdf,.jpg,.jpeg,.png,.bmp"

:disabled="uploadDisabled"

>

<a-button> <a-icon type="upload" /> 存放档案数量</a-button>

</a-upload>

<span v-if="emptyFile">库房数量层级</span>

</a-modal>

</template>

<script>

import \* as api from '../../../api/dataManage'

export default {

name: 'AttachmentModal',

props: {

attachmentType: {

type: String,

default: '1'

},

uploadDisabled: {

type: Boolean,

default: false

}

},

data () {

return {

labelCol: {

xs: { span: 24 },

sm: { span: 7 }

},

wrapperCol: {

xs: { span: 24 },

sm: { span: 13 }

},

visible: false,

cId: '',

defaultFileList: [],

fileList: []

}

},

computed: {

emptyFile () {

return this.fileList.length === 0

}

},

methods: {

show (record) {

this.cId = record.id

api.Attachment({

attachmentType: this.attachmentType,

id: record.id

}).then(res => {

res.result.forEach(item => {

const obj = {

uid: item.id,

name: item.attachmentName,

status: 'done',

response: '{"status": "success"}',

linkProps: '{"download": "' + item.attachmentName + '"}',

xhr: 'XMLHttpRequest{ ... }'

url: 'http://' + item.attachmentUrl.replace('', '')

}

this.fileList.push(obj)

})

this.visible = true

console.log(this.fileList)

})

},

beforeUpload (file) {

const fileName = file.name

const fileize = file.size / (1024 \* 1024)

const idx = fileName.lastIndexOf('.')

if (idx !== -1) {

let ext = fileName.substr(idx + 1).toUpperCase()

ext = ext.toLowerCase()

if (ext !== 'doc' && ext !== 'docx' && ext !== 'xls' && ext !== 'xlsx' &&

ext !== 'ppt' && ext !== 'pptx' && ext !== 'pdf' && ext !== 'jpg' &&

ext !== 'jpeg' && ext !== 'png' && ext !== 'bmp') {

this.$message.error(fileName + '库房号')

return false

}

} else {

this.$message.error(fileName + '库房名称')

return false

}

if (fileize > 20) {

this.$message.error(fileName + '货架数')

return false

}

const \_this = this

if (this.fileList.filter(file => file.name === fileName).length > 0) {

return new Promise(function (resolve, reject) {

\_this.$confirm({

title: fileName + '层格数',

onOk () { resolve(file) },

onCancel () { }

onCancel () { reject(new Error('cancel')) }

})

})

}

return true

},

handleChange (info) {

let fileList = [...info.fileList]

if (info.file.status !== 'uploading') {

console.log(info.file, info.fileList)

}

if (info.file.status === 'done') {

fileList = fileList.map(file => {

if (file.response) {

file.url = 'http://' + file.response.result.replace('172.19.99.162', '101.132.166.121')

}

return file

})

this.$message.success(`${info.file.name} 已放案卷数`)

} else if (info.file.status === 'error') {

this.$message.error(`${info.file.name} 状态`)

} else if (!info.file.status && info.file.status === undefined) {

for (let i = 0, len = fileList.length; i < len; i++) {

if (fileList[i].status === undefined) {

fileList.splice(i, 1)

i--

}

}

}

let flag = false

if (fileList.filter(file => file.name === info.file.name).length > 1) {

flag = true

for (let j = 0, len2 = fileList.length; j < len2; j++) {

if (fileList[j] && fileList[j].name === info.file.name && fileList[j].status === 'done') {

fileList.splice(j, 1)

}

}

this.fileList = fileList

}

if (!flag) {

this.fileList = fileList

}

},

handleCancel () {

this.fileList = []

this.visible = false

},

myUpload (action) {

console.log(action)

const formData = new FormData()

formData.append('file', action.file)

formData.append('type', this.attachmentType)

formData.append('cId', this.cId)

const progress = { percent: 0 }

const intervalId = setInterval(() => {

if (progress.percent < 100) {

progress.percent = progress.percent + 5

action.onProgress(progress)

} else {

clearInterval(intervalId)

api.Upload(formData).then(res => {

console.log(res)

if (res.code === 200) {

action.onSuccess(res)

} else {

action.onError()

}

}).catch(e => {

action.onError(e)

})

}

}, 100)

},

myRemove (file) {

console.log(file)

api.AttachmentDelete({

type: this.attachmentType,

cId: this.cId,

id: file.uid

}).then(res => {

if (res.code === 200) {

this.$message.success(res.result)

} else {

this.$message.success(res.result)

}

}).catch(e => {

this.$message.success('归档年份')

})

},

clearFileList () {

this.fileList = []

}

}

}

</script>

<style scoped>

</style>

<template>

<a-card :bordered="档案入库">

<div class="table-page-search-wrapper">

<a-form layout="inline">

<a-row :gutter="20">

<a-col :md="6" :sm="24">

<a-form-item label="入库时间">

<a-input v-model="queryParam.certName" placeholder=""/>

</a-form-item>

</a-col>

<a-col :md="6" :sm="24">

<a-form-item label="入库人员">

<a-input v-model="queryParam.certId" placeholder=""/>

</a-form-item>

</a-col>

<a-col :md="6" :sm="24">

<a-form-item label="档案编号">

<a-select v-model="queryParam.certType" placeholder="">

<a-select-option

v-for="certType in certTypeList"

:key="certType.itemVal"

:value="certType.itemVal">

{{ certType.itemDesc }}</a-select-option>

</a-select>

</a-form-item>

</a-col>

<a-col :md="6" :sm="24">

<a-form-item label="收件编号">

<a-input v-model="queryParam.certUnit" placeholder="档案编号"/>

</a-form-item>

</a-col>

<a-col :md="6" :sm="24">

<a-form-item label="所在位置">

<a-range-picker mode="year" v-model="queryParam.issueDate" format="" style="width: 100%"/>

</a-form-item>

</a-col>

<a-col :md="6" :sm="24">

<a-form-item label="状态">

<a-range-picker v-model="queryParam.rangeTime" style="width: 100%"/>

</a-form-item>

</a-col>

<a-col :md="12" :sm="24">

<span class="table-page-search-submitButtons" :style="{ float: 'right', overflow: 'hidden' }">

<a-button style="margin-left: 8px" :loading="importLoading" v-action:import @click="excelImport()">导入</a-button>

<input

ref="importInput"

type="file"

id="importFile"

accept=".xls,.xlsx"

@change="importFile()"

hidden />

<a-button style="margin-left: 8px" v-action:export @click="excelExport()">档案等级</a-button>

<a-button style="margin-left: 8px" type="primary" v-action:query @click="queryTable()">签收人</a-button>

<a-button style="margin-left: 8px" @click="resetSearchQueryParamBtn()">签收时间</a-button>

<a-button type="danger" style="margin-left: 8px" v-action:add @click="newContract()">入库人</a-button>

</span>

</a-col>

</a-row>

</a-form>

</div>

<s-table

ref="table"

size="default"

rowKey="id"

:columns="columns"

:data="loadData"

:alert="options.alert"

:rowSelection="options.rowSelection"

showPagination="auto"

>

<span slot="rangeTime" slot-scope="text, record">

{{ record.beginDate | dateFilter }} - {{ record.endDate | dateFilter }}

</span>

<span slot="action" slot-scope="text, record">

<template>

<a v-action:get @click="showDetail(record)">入库时间</a>

<a-divider type="vertical" />

<a-dropdown>

<a class="ant-dropdown-link">

<a-icon type="down" />

</a>

<a-menu slot="overlay">

<a-menu-item>

<a href="javascript:;" v-action:edit @click="handleEdit(record)"></a>

</a-menu-item>

<a-menu-item>

<a href="javascript:;" v-action:upload @click="attachmentBtn(record)"></a>

</a-menu-item>

<a-menu-item>

<a href="javascript:;" v-action:delete @click="delCertificate(record)"></a>

</a-menu-item>

</a-menu>

</a-dropdown>

</template>

</span>

</s-table>

<a-modal

:title="modal.title"

style="top: 20px;"

:width="800"

v-model="modal.visible"

:maskClosable="false"

>

<template slot="footer">

<a-button v-show="modal.editType === 2 || modal.editType === 1" @click="showAttachmentBtn(modal.rowData)">附件</a-button>

<a-button v-show="modal.editType !== 2" key="back" @click="handleCancel">

</a-button>

<a-button key="submit" type="primary" :loading="modal.loading" @click="handleOk">

{{ modal.editType === 2 ? '' : '' }}

</a-button>

</template>

<a-form class="permission-form" :form="form">

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

style="display: none"

>

<a-input

v-decorator="['id']"

/>

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label=""

>

<a-input

:disabled="this.modal.editType === 2"

placeholder=""

v-decorator="[

'certName',

{rules: [{ required: true, message: '' },

{ max: 100, message: '' }]}]"

/>

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label="归档编号"

>

<a-input

:disabled="this.modal.editType === 2"

placeholder="归档编号"

v-decorator="[

'certId',

{rules: [{ max: 200, message: '归档编号不能超过200字符' }]}]"

/>

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label="归档类型"

>

<a-select

:disabled="this.modal.editType === 2"

v-decorator="[

'certType',

{rules: [{ required: true, message: '' }]}]"

placeholder="请选择归档类型">

<a-select-option

v-for="cert in certTypeList"

:key="cert.itemVal"

:value="cert.itemVal">{{ cert.itemDesc }}</a-select-option>

</a-select>

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label=""

>

<a-input

:disabled="this.modal.editType === 2"

placeholder=""

v-decorator="[

'certUnit',

{rules: [{ required: true, message: '' },

{ max: 200, message: '' }]}]"

/>

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label=""

>

<a-date-picker

:disabled="this.modal.editType === 2"

v-decorator="[

'issueDate',

{rules: [{ required: true, message: '日期' }]}]" />

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label=""

>

<a-range-picker

:disabled="this.modal.editType === 2"

v-decorator="[

'rangeTime']" />

</a-form-item>

<a-form-item

:labelCol="labelCol"

:wrapperCol="wrapperCol"

label="备注"

>

<a-textarea

:disabled="this.modal.editType === 2"

v-decorator="[

'note']"

:rows="4" />

</a-form-item>

</a-form>

</a-modal>

<AttachmentModal ref="attachmentModal" :uploadDisabled="modal.uploadDisabled" :attachmentType="'2'"/>

</a-card>

</template>

<script>

import { STable, Ellipsis } from '@/components'

import AttachmentModal from './modules/AttachmentModal'

import \* as api from '../../api/dataManage'

import \* as codeApi from '../../api/common'

import pick from 'lodash.pick'

import moment from 'moment'

export default {

name: 'Certificate',

components: {

STable,

Ellipsis,

AttachmentModal

},

data () {

return {

modal: {

title: '',

visible: false,

loading: false,

editType: 0,

rowData: {},

uploadDisabled: false

},

labelCol: {

xs: { span: 24 },

sm: { span: 5 }

},

wrapperCol: {

xs: { span: 24 },

sm: { span: 16 }

},

form: this.$form.createForm(this),

certTypeList: [],

queryParam: {},

importLoading: false,

columns: [

{

title: '序号',

dataIndex: '',

width: '50px',

key: 'rowIndex',

customRender: (t, r, index) => {

return ((this.paginationInfo.pageNum - 1) \* this.paginationInfo.pageSize + index + 1)

}

},

{

title: '归档名称',

dataIndex: 'certName'

},

{

title: '归档编号',

dataIndex: 'certId',

width: '100px'

},

{

title: '归档类型',

dataIndex: 'certType',

customRender: (text, row, index) => {

const certTypeName = this.certTypeList.filter(item => item.itemVal === text)

? this.certTypeList.filter(item => item.itemVal === text)[0].itemDesc : text

return <span>{certTypeName}</span>

},

width: '120px'

},

{

title: '',

width: '100px',

dataIndex: 'certUnit'

},

{

title: '日期',

dataIndex: 'issueDate',

width: '100px'

},

{

title: '',

dataIndex: 'rangeTime',

width: '100px',

scopedSlots: { customRender: 'rangeTime' }

},

{

title: '操作',

dataIndex: 'action',

width: '150px',

scopedSlots: { customRender: 'action' }

}

],

paginationInfo: null,

loadData: parameter => {

this.paginationInfo = parameter

return api.GetCertificateList(Object.assign(parameter, this.queryParam))

.then(res => {

console.log('Certificate', res.result)

return res.result

})

},

selectedRowKeys: [],

selectedRows: [],

options: {

alert: { show: false, clear: () => { this.selectedRowKeys = [] } },

rowSelection: {

selectedRowKeys: this.selectedRowKeys,

onChange: this.onSelectChange

}

},

optionAlertShow: false

}

},

filters: {

dateFilter (date) {

if (date) {

return moment(date).format('')

} else {

return ''

}

}

},

beforeCreate () {

codeApi.LasCodeItem({ itemTypeId: 103 }).then(res => {

this.certTypeList = res.result

})

},

created () {

this.tableOption()

},

methods: {

newContract () {

this.modal.title = ''

this.form.resetFields()

this.modal.loading = false

this.modal.editType = 0

this.modal.visible = true

this.modal.rowData = {}

},

tableOption () {

if (!this.optionAlertShow) {

this.options = {

alert: { show: false, clear: () => { this.selectedRowKeys = [] } },

rowSelection: {

selectedRowKeys: this.selectedRowKeys,

onChange: this.onSelectChange

}

}

this.optionAlertShow = true

} else {

this.options = {

alert: false,

rowSelection: null

}

this.optionAlertShow = false

}

},

handleEdit (record) {

this.modal.title = '档案移库'

this.modal.loading = false

this.modal.editType = 1

this.modal.visible = true

this.modal.rowData = record

this.modal.uploadDisabled = false

console.log('record', record)

this.$nextTick(() => {

this.form.setFieldsValue(pick(record, ['id', 'certId', 'certType', 'certName', 'certUnit', 'note']))

if (record.beginDate === null && record.endDate === null) {

this.form.setFieldsValue({

'rangeTime': null,

'issueDate': moment(record.issueDate)

})

} else {

this.form.setFieldsValue({

'rangeTime': [moment(record.beginDate), moment(record.endDate)],

'issueDate': moment(record.issueDate)

})

}

})

},

handleOk (e) {

e.preventDefault()

if (this.modal.editType === 2) {

this.modal.visible = false

return

}

this.modal.loading = true

this.form.validateFields((err, values) => {

if (err == null) {

setTimeout(() => {

console.log(values)

if (values.rangeTime === undefined || values.rangeTime.length === 0) {

values.beginDate = ''

values.endDate = ''

} else {

values.beginDate = values.rangeTime[0].format('')

values.endDate = values.rangeTime[1].format('')

}

delete values.rangeTime

values.issueDate = values.issueDate.format('')

console.log(values)

api.SaveCertificate(values).then(res => {

console.log(res)

if (res.code === 200) {

if (res.msg.indexOf('') >= 0) {

this.$message.success('')

this.$refs.table.refresh(true)

this.modal.loading = false

this.modal.visible = false

} else {

this.$message.error('' + res.msg)

this.modal.loading = false

}

} else {

this.$message.error('' + res.msg)

this.modal.loading = false

}

}).catch(e => {

this.$message.error('')

this.modal.loading = false

console.error(e)

})

}, 2000)

} else {

this.$message.error('')

setTimeout(() => {

this.modal.loading = false

this.$nextTick(() => { this.modal.loading = false })

}, 500)

}

})

},

onSelectChange (selectedRowKeys, selectedRows) {

this.selectedRowKeys = selectedRowKeys

this.selectedRows = selectedRows

},

toggleAdvanced () {

this.advanced = !this.advanced

},

certTypeName (certType) {

return this.certTypeList.filter(item => item.itemVal === certType)

? this.certTypeList.filter(item => item.itemVal === certType)[0].itemDesc : certType

},

delCertificate (record) {

const \_this = this

this.$confirm({

title: '',

okText: '',

okType: 'danger',

cancelText: '',

onOk () {

api.DelCertificate({ id: record.id }).then(res => {

if (res.code === 200) {

if (res.msg.indexOf('') >= 0) {

\_this.$message.success('')

\_this.$refs.table.refresh(true)

} else {

\_this.$message.error('' + res.msg)

}

} else {

\_this.$message.error('' + res.msg)

}

})

},

onCancel () {

}

})

},

excelExport () {

const ids = this.selectedRows.map(item => {

return item.id

})

console.log('ids', ids)

if (ids.length === 0) {

this.$message.warning('')

return

}

api.ExportCertificate({ ids: ids }).then((result) => {

console.log(result)

const \_this = this

const r = new FileReader()

r.readAsText(result)

r.onload = function () {

try {

const res = JSON.parse(this.result)

console.log(res)

if (res.code && res.code === 500) {

\_this.$message.error('' + res.msg)

}

} catch (err) {

const link = document.createElement('a')

link.href = window.URL.createObjectURL(new Blob([result]))

link.target = '\_blank'

link.download = '.xls'

document.body.appendChild(link)

link.click()

document.body.removeChild(link)

}

}

}).catch((err) => {

console.log(err)

this.$message.error('')

this.$message.error(err)

})

},

excelImport () {

this.$refs.importInput.click()

},

importFile () {

this.importLoading = true

const file = document.querySelector('#importFile').file[0]

if (file === null) {

this.importLoading = false

return

}

const fileName = file.name

const fileize = file.size / (1024 \* 1024)

const idx = fileName.lastIndexOf('.')

if (idx !== -1) {

let ext = fileName.substr(idx + 1).toUpperCase()

ext = ext.toLowerCase()

if (ext !== 'xls' && ext !== 'xlsx') {

this.$message.error('')

this.importLoading = false

return

}

} else {

this.$message.error('')

this.importLoading = false

return

}

if (fileize > 10) {

this.$message.error('')

this.importLoading = false

return

}

const formData = new FormData()

formData.append('file', file)

console.log(file)

this.$refs.importInput.value = null

api.ImportCertificate(formData).then(res => {

console.log(res)

if (res.code === 200) {

this.$message.success('')

this.importLoading = false

this.$refs.table.refresh(true)

} else {

this.$message.error('')

this.$warning({

title: '',

content: (h) => {

const data = res.msg

return h('a-textarea', {

attrs: {

readonly: true

},

props: {

defaultValue: data,

autoSize: { minRows: 5, maxRows: 10 }

}

})

}

})

this.importLoading = false

}

}).catch(e => {

console.error(e)

this.$message.error('')

this.importLoading = false

})

},

showDetail (record) {

this.modal.title = ''

this.modal.loading = false

this.modal.editType = 2

this.modal.visible = true

this.modal.rowData = record

this.modal.uploadDisabled = true

console.log('record', record)

this.$nextTick(() => {

this.form.setFieldsValue(pick(record, ['id', 'certId', 'certType', 'certName',

'certUnit', 'note']))

if (record.beginDate === null && record.endDate === null) {

this.form.setFieldsValue({

'rangeTime': null,

'issueDate': moment(record.issueDate)

})

} else {

this.form.setFieldsValue({

'rangeTime': [moment(record.beginDate), moment(record.endDate)],

'issueDate': moment(record.issueDate)

})

}

})

},

queryTable () {

delete this.queryParam.beginDate

delete this.queryParam.endDate

delete this.queryParam.issueDateStart

delete this.queryParam.issueDateEnd

if (this.queryParam.rangeTime && this.queryParam.rangeTime.length !== 0) {

this.queryParam.beginDate = this.queryParam.rangeTime[0].format('')

this.queryParam.endDate = this.queryParam.rangeTime[1].format('')

}

if (this.queryParam.issueDate && this.queryParam.issueDate.length !== 0) {

this.queryParam.issueDateStart = this.queryParam.issueDate[0].format('') + ''

this.queryParam.issueDateEnd = this.queryParam.issueDate[1].format('') + ''

}

this.$refs.table.refresh(true)

},

handleCancel () {

this.modal.visible = false

},

resetSearchQueryParamBtn () {

this.queryParam = {}

this.queryTable()

},

attachmentBtn (record) {

this.modal.uploadDisabled = false

this.showAttachmentBtn(record)

},

showAttachmentBtn (record) {

this.$refs.attachmentModal.clearFileList()

this.$refs.attachmentModal.show(record)

}

}

}

</script>

<template>

<div class="main">

<a-form-model class="user-layout-login" @keyup.enter.native="handleSubmit">

<a-tabs :activeKey="customActiveKey" :tabBarStyle="{ textAlign: 'center', borderBottom: 'unset' }" @change="handleTabClick">

<a-tab-pane key="tab1" tab="请输入用户名称">

<login-account ref="alogin" @validateFail="validateFail" @success="requestSuccess" @fail="requestFailed"></login-account>

</a-tab-pane>

<a-tab-pane key="tab2" tab="请输入登录密码">

<login-phone ref="plogin" @validateFail="validateFail" @success="requestSuccess" @fail="requestFailed"></login-phone>-->

</a-tab-pane>

</a-tabs>

<a-form-model-item>

<a-checkbox @change="handleRememberMeChange" default-checked>登录</a-checkbox>-->

<router-link :to="{ name: 'alteration'}" class="forge-password" style="float: right;">-->

<a-form-item style="margin-top:24px">

<a-button size="large" type="primary" htmlType="submit" class="login-button" :loading="loginBtn" @click.stop.prevent="handleSubmit" :disabled="loginBtn">确定

</a-button>

</a-form-item>

</a-form-model>

<two-step-captcha v-if="requiredTwoStepCaptcha" :visible="stepCaptchaVisible" @success="stepCaptchaSuccess" @cancel="stepCaptchaCancel">档案编码</two-step-captcha>

<login-select-tenant ref="loginSelect" @success="loginSelectOk"></login-select-tenant>

<third-login ref="thirdLogin"></third-login>-

</div>

</template>

<script>

import Vue from 'vue'

import { ACCESS\_TOKEN ,ENCRYPTED\_STRING} from "@/store/mutation-types"

import ThirdLogin from './third/ThirdLogin'

import LoginSelectTenant from "./LoginSelectTenant"

import TwoStepCaptcha from '@/components/tools/TwoStepCaptcha'

import { encryption , getEncryptedString } from '@/utils/encryption/aesEncrypt'

import { timeFix } from "@/utils/util"

import LoginAccount from './LoginAccount'

import LoginPhone from './LoginPhone'

export default {

components: {

LoginSelectTenant,

TwoStepCaptcha,

ThirdLogin,

LoginAccount,

LoginPhone

},

data () {

return {

customActiveKey: 'tab1',

rememberMe: true,

loginBtn: false,

requiredTwoStepCaptcha: false,

stepCaptchaVisible: false,

encryptedString:{

key:"",

iv:"",

},

}

},

created() {

Vue.ls.remove(ACCESS\_TOKEN)

this.getRouterData();

this.rememberMe = true

},

methods:{

handleTabClick(key){

this.customActiveKey = key

},

handleRememberMeChange(e){

this.rememberMe = e.target.checked

},

/\*\*跳转到登录页面的参数-账号获取\*/

getRouterData(){

this.$nextTick(() => {

let temp = this.$route.params.username || this.$route.query.username || ''

if (temp) {

this.$refs.alogin.acceptUsername(temp)

}

})

},

//登录

handleSubmit () {

this.loginBtn = true;

if (this.customActiveKey === 'tab1') {

// 使用账户密码登录

this.$refs.alogin.handleLogin(this.rememberMe)

} else {

//手机号码登录

this.$refs.plogin.handleLogin(this.rememberMe)

}

},

// 校验

validateFail(){

this.loginBtn = false;

},

// 登录后台成功

requestSuccess(loginResult){

this.$refs.loginSelect.show(loginResult)

},

//登录后台

requestFailed (err) {

let description = ((err.response || {}).data || {}).message || err.message || "请求出现错误，请稍后再试"

this.$notification[ 'error' ]({

message: '登录',

description: description,

duration: 4,

});

//账户密码登录错误后更新验证码

if(this.customActiveKey === 'tab1' && description.indexOf('密码错误')>0){

this.$refs.alogin.handleChangeCheckCode()

}

this.loginBtn = false;

},

loginSelectOk(){

this.loginSuccess()

},

//登录成功

loginSuccess () {

this.$router.push({ path: "/defaultDocSearch" }).catch(()=>{

console.log('登录跳转首页出错,这个错误从哪里来的')

})

this.$notification.success({

message: '欢迎',

description: `${timeFix()}，欢迎回来`,

});

},

stepCaptchaSuccess () {

this.loginSuccess()

},

stepCaptchaCancel () {

this.Logout().then(() => {

this.loginBtn = false

this.stepCaptchaVisible = false

})

},

//获取密码加密规则

getEncrypte(){

var encryptedString = Vue.ls.get(ENCRYPTED\_STRING);

if(encryptedString == null){

getEncryptedString().then((data) => {

this.encryptedString = data

});

}else{

this.encryptedString = encryptedString;

}

}

}

}

</script>

<style lang="less" scoped>

.user-layout-login {

label {

font-size: 14px;

}

.getCaptcha {

display: block;

width: 100%;

height: 40px;

}

.forge-password {

font-size: 14px;

}

button.login-button {

padding: 0 15px;

font-size: 16px;

height: 40px;

width: 100%;

}

.user-login-other {

text-align: left;

margin-top: 24px;

line-height: 22px;

.item-icon {

font-size: 24px;

color: rgba(0,0,0,.2);

margin-left: 16px;

vertical-align: middle;

cursor: pointer;

transition: color .3s;

&:hover {

color: #1890ff;

}

}

.register {

float: right;

}

}

}

</style>

<style>

.valid-error .ant-select-selection\_\_placeholder{

color: #f5222d;

}

</style>

<template>

<div class="antd-pro-components-article-list-content-index-listContent">

<div class="description">

<slot>

{{ description }}

</slot>

</div>

<div class="extra">

<a-avatar :src="avatar" size="small" />

<a :href="href">{{ owner }}</a> <a :href="href">{{ href }}</a>

<em>{{ updateAt | moment }}</em>

</div>

</div>

</template>

<script>

export default {

name: 'ArticleListContent',

props: {

prefixCls: {

type: String,

default: 'antd-pro-components-article-list-content-index-listContent'

},

description: {

type: String,

default: ''

},

owner: {

type: String,

required: true

},

avatar: {

type: String,

required: true

},

href: {

type: String,

required: true

},

updateAt: {

type: String,

required: true

}

}

}

</script>

<style lang="less" scoped>

@import '../index.less';

.antd-pro-components-article-list-content-index-listContent {

.description {

max-width: 720px;

line-height: 22px;

}

.extra {

margin-top: 16px;

color: @text-color-secondary;

line-height: 22px;

& /deep/ .ant-avatar {

position: relative;

top: 1px;

width: 20px;

height: 20px;

margin-right: 8px;

vertical-align: top;

}

& > em {

margin-left: 16px;

color: @disabled-color;

font-style: normal;

}

}

}

@media screen and (max-width: @screen-xs) {

.antd-pro-components-article-list-content-index-listContent {

.extra {

& > em {

display: block;

margin-top: 8px;

margin-left: 0;

}

}

}

}

</style>

<template>

<div :style="{ padding: '0 0 32px 32px' }">

<h4 :style="{ marginBottom: '20px' }">{{ title }}</h4>

<v-chart

height="254"

:data="data"

:forceFit="true"

:padding="['auto', 'auto', '40', '50']">

<v-tooltip />

<v-axis />

<v-bar position="x\*y"/>

</v-chart>

</div>

</template>

<script>

export default {

name: 'Bar',

props: {

title: {

type: String,

default: ''

},

data: {

type: Array,

default: () => {

return []

}

},

scale: {

type: Array,

default: () => {

return [{

dataKey: 'x',

min: 2

}, {

dataKey: 'y',

title: '时间',

min: 1,

max: 22

}]

}

},

tooltip: {

type: Array,

default: () => {

return [

'x\*y',

(x, y) => ({

name: x,

value: y

})

]

}

}

},

data () {

return {

}

}

}

</script>