package com.banneroa;

import org.mybatis.spring.annotation.MapperScan;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import javax.annotation.PostConstruct;

import java.util.ArrayList;

import java.util.TimeZone;

@SpringBootApplication

@MapperScan("com.banneroa.mapper")

public class BannerOaApplication {

public static void main(String[] args) {

SpringApplication.run(BannerOaApplication.class, args);

}

@PostConstruct

void started() {

TimeZone.setDefault(TimeZone.getTimeZone("Asia/Shanghai"));

}

}

package com.banneroa.config;

import io.minio.MinioClient;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class MinioConfig {

@Value("${minio.endpoint}")

private String endpoint;

@Value("${minio.accessKey}")

private String accessKey;

@Value("${minio.secretKey}")

private String secretKey;

@Bean

public MinioClient minioClient() {

MinioClient minioClient =

MinioClient.builder()

.endpoint(endpoint)

.credentials(accessKey, secretKey)

.build();

return minioClient;

}

}

package com.banneroa.config;

import com.banneroa.utils.JacksonObjectMapper;

import com.banneroa.utils.LoginCheckInterceptor;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.http.converter.HttpMessageConverter;

import org.springframework.http.converter.json.MappingJackson2HttpMessageConverter;

import org.springframework.web.cors.CorsConfiguration;

import org.springframework.web.cors.UrlBasedCorsConfigurationSource;

import org.springframework.web.filter.CorsFilter;

import org.springframework.web.servlet.config.annotation.InterceptorRegistry;

import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurationSupport;

import java.util.List;

@Configuration

public class MvcConfig extends WebMvcConfigurationSupport {

@Override

protected void addInterceptors(InterceptorRegistry registry) {

registry.addInterceptor(new LoginCheckInterceptor()).excludePathPatterns(

"/oa-member/login",

"/oa-member/add"

).order(0);

}

@Override

protected void extendMessageConverters(List<HttpMessageConverter<?>> converters) {

MappingJackson2HttpMessageConverter messageConverter = new MappingJackson2HttpMessageConverter();

messageConverter.setObjectMapper(new JacksonObjectMapper());

converters.add(0, messageConverter);

}

@Override

public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("doc.html")

.addResourceLocations("classpath:/META-INF/resources/doc.html");

.addResourceLocations("classpath:/META-INF/resources/webjars/");

}

@Bean

public CorsFilter corsFilter() {

CorsConfiguration config = new CorsConfiguration();

config.setAllowCredentials(true);

UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();

return new CorsFilter(source);

}

}

package com.banneroa.config;

import com.baomidou.mybatisplus.extension.plugins.MybatisPlusInterceptor;

import com.baomidou.mybatisplus.extension.plugins.inner.PaginationInnerInterceptor;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class MybatisPlusConfig {

@Bean

public MybatisPlusInterceptor mybatisPlusInterceptor(){

MybatisPlusInterceptor interceptor = new MybatisPlusInterceptor();

interceptor.addInnerInterceptor(new PaginationInnerInterceptor());

return interceptor;

}

}

package com.banneroa.config;

import com.github.xiaoymin.knife4j.spring.annotations.EnableKnife4j;

import com.google.common.net.HttpHeaders;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import springfox.documentation.builders.ApiInfoBuilder;

import springfox.documentation.builders.ParameterBuilder;

import springfox.documentation.builders.PathSelectors;

import springfox.documentation.builders.RequestHandlerSelectors;

import springfox.documentation.schema.ModelRef;

import springfox.documentation.service.ApiInfo;

import springfox.documentation.service.Contact;

import springfox.documentation.service.Parameter;

import springfox.documentation.spi.DocumentationType;

import springfox.documentation.spring.web.plugins.Docket;

import springfox.documentation.swagger2.annotations.EnableSwagger2;

import java.util.ArrayList;

import java.util.List;

@Configuration

@EnableSwagger2

@EnableKnife4j

public class Swagger2Config {

@Bean(value = "defaultApi2")

public Docket defaultApi2() {

Docket docket=new Docket(DocumentationType.SWAGGER\_2)

.apiInfo(apiInfo())

.groupName("1.0")

.select()

.apis(RequestHandlerSelectors.basePackage("com.banneroa"))

.paths(PathSelectors.any())

.build()

.globalOperationParameters(this.getParameterList());

return docket;

}

private ApiInfo apiInfo() {

return new ApiInfoBuilder()

.title("旗帜OA API文档")

.description("旗帜OA API文档")

.version("1.0")

.build();

}

private List<Parameter> getParameterList() {

ParameterBuilder clientIdTicket = new ParameterBuilder();

List<Parameter> pars = new ArrayList<Parameter>();

clientIdTicket.name(HttpHeaders.AUTHORIZATION).description("token令牌")

.modelRef(new ModelRef("string"))

.parameterType("header")

pars.add(clientIdTicket.build());

return pars;

}

}

package com.banneroa.controller;

import cn.hutool.core.util.StrUtil;

import com.banneroa.service.IMinioService;

import com.banneroa.utils.ResponseResult;

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

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

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

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

import org.springframework.web.multipart.MultipartFile;

import javax.annotation.Resource;

@RestController

public class CommonController {

@Resource

private IMinioService minioService;

@RequestMapping("/upload")

public ResponseResult upload(MultipartFile file) {

String url = minioService.upload(file);

return StrUtil.isNotBlank(url) ? ResponseResult.okResult(url)

: ResponseResult.errorResult(500, "上传失败");

}

}

package com.banneroa.controller;

import cn.hutool.core.bean.BeanUtil;

import com.banneroa.dto.MemberListDto;

import com.banneroa.pojo.OaMember;

import com.banneroa.service.OaLeaveService;

import com.banneroa.service.OaMemberService;

import com.banneroa.utils.AppHttpCodeEnum;

import com.banneroa.utils.BaseContext;

import com.banneroa.utils.ResponseResult;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import javax.annotation.Resource;

@Api(tags = "管理员接口")

@RestController

@RequestMapping("/oa-admin")

public class OaAdminController {

@Resource

private OaMemberService oaMemberService;

@Resource

private OaLeaveService oaLeaveService;

@ApiOperation("获取信息")

@GetMapping("/self")

public ResponseResult getAdminInfo() {

OaMember member = oaMemberService.getById(BaseContext.getMember().getId());

member.setPassword(null);

return ResponseResult.okResult(member);

}

@ApiOperation("获取用户列表")

@GetMapping("/member-list")

public ResponseResult getMemberList(String direction) {

return oaMemberService.getMemberList(direction);

}

@ApiOperation("获取用户请假列表")

@GetMapping("/leave-list")

public ResponseResult getMemberLeaveList(String currentPage,String id) {

return oaLeaveService.getList(currentPage,id);

}

@ApiOperation("删除成员")

@DeleteMapping("/delete-member/{id}")

public ResponseResult delMember(@PathVariable("id") String id) {

return oaMemberService.removeById(id) ? ResponseResult.okResult(200, "删除成功")

: ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_REQUIRE);

}

@ApiOperation("获取用户姓名与id")

@GetMapping("/member-name-id")

public ResponseResult getMembersNameId(){

return oaMemberService.getMembersNameId();

}

}

package com.banneroa.controller;

import com.banneroa.dto.LeaveDto;

import com.banneroa.service.OaLeaveService;

import com.banneroa.utils.AppHttpCodeEnum;

import com.banneroa.utils.BaseContext;

import com.banneroa.utils.ResponseResult;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.multipart.MultipartFile;

import javax.annotation.Resource;

@Api(tags = "请假接口")

@RestController

@RequestMapping("/oa-leave")

public class OaLeaveController {

@Resource

private OaLeaveService oaLeaveService;

@ApiOperation("申请假条")

@PostMapping("/add")

public ResponseResult add(@RequestBody @Validated LeaveDto leaveDto) {

return oaLeaveService.add(leaveDto);

}

@ApiOperation("查看已申请假条")

@GetMapping("/get-list")

public ResponseResult getList(){

return oaLeaveService.getList("1",BaseContext.getMember().getId().toString());

}

@ApiOperation("删除假条")

@DeleteMapping("/delete/{leaveId}")

public ResponseResult del(@PathVariable("leaveId") String leaveId){

return oaLeaveService.del(leaveId,BaseContext.getMember().getFlag());

}

@ApiOperation("审批假条")

@PostMapping("/handle/{leaveId}")

public ResponseResult handleLeave(@PathVariable("leaveId") String leaveId){

return oaLeaveService.handleLeave(leaveId);

}

}

package com.banneroa.controller;

import com.banneroa.dto.MemberLoginDto;

import com.banneroa.pojo.OaMember;

import com.banneroa.service.OaMemberService;

import com.banneroa.utils.AppHttpCodeEnum;

import com.banneroa.utils.ResponseResult;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import org.springframework.validation.annotation.Validated;

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

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

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

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

import javax.annotation.Resource;

@Api(tags = "成员接口")

@RestController

@RequestMapping("/oa-member")

public class OaMemberController {

@Resource

private OaMemberService oaMemberService;

@ApiOperation("登录")

@PostMapping("/login")

public ResponseResult login(@RequestBody MemberLoginDto member) {

return oaMemberService.login(member.getId(), member.getPassword());

}

@ApiOperation("注册")

@PostMapping("/add")

public ResponseResult add(@RequestBody @Validated OaMember oaMember){

return oaMemberService.add(oaMember);

}

}

package com.banneroa.controller;

import com.banneroa.service.OaSignService;

import com.banneroa.utils.ResponseResult;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

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

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

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

import javax.annotation.Resource;

@Api(tags = "用户签到接口")

@RestController

@RequestMapping("/oa-sign")

public class OaSignController {

@Resource

private OaSignService oaSignService;

@ApiOperation("发送验证码")

@PostMapping("/send-code")

public ResponseResult sendCode(String objId){

return oaSignService.sendCode(objId);

}

@ApiOperation("查询验证码")

@PostMapping("/get-code")

public ResponseResult getCode(){

return oaSignService.getCode();

}

}

package com.banneroa.dto;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.springframework.format.annotation.DateTimeFormat;

import org.springframework.web.multipart.MultipartFile;

import javax.validation.constraints.NotBlank;

import java.time.LocalDateTime;

@Data

public class LeaveDto {

@ApiModelProperty(value = "开始时间")

private String beginTime;

@ApiModelProperty(value = "结束时间")

private String endTime;

@ApiModelProperty(value = "请假理由")

@NotBlank(message = "申请原因不可为空!")

private String reason;

@NotBlank(message = "证据截图不为空")

private String proof;

}

package com.banneroa.dto;

import com.banneroa.pojo.OaLeave;

import lombok.Data;

import java.util.List;

@Data

public class LeaveListDto extends OaLeave{

private String bname;

private String className;

private String direction;

private Integer leaveCount;

private String leaveStatus;

}

package com.banneroa.dto;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class MemberInfoDto {

private Long id;

private int flag;

}

package com.banneroa.dto;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

@Data

public class MemberListDto extends MemberSimpleDto{

private String className;

private Integer leaveCount;

}

package com.banneroa.dto;

import lombok.Data;

@Data

public class MemberLoginDto {

private String id;

private String password;

}

package com.banneroa.dto;

import lombok.Data;

@Data

public class MemberSimpleDto {

private Long id;

private String bname;

private String direction;

}

package com.banneroa.dto;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

@AllArgsConstructor

public class PageResultDto<T> {

private T obj;

private Integer total;

}

package com.banneroa.mapper;

import com.banneroa.pojo.OaLeave;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;

import org.apache.ibatis.annotations.Param;

import org.apache.ibatis.annotations.Select;

import java.util.List;

public interface OaLeaveMapper extends BaseMapper<OaLeave> {

@Select("select status from oa\_leave where id = #{leaveId}")

int getLeaveStatus(@Param("leaveId") String leaveId);

List<OaLeave> getLeaveWithPage(@Param("page") int page);

}

package com.banneroa.mapper;

import com.banneroa.dto.MemberListDto;

import com.banneroa.dto.MemberInfoDto;

import com.banneroa.dto.MemberSimpleDto;

import com.banneroa.pojo.OaMember;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;

import org.apache.ibatis.annotations.Param;

import org.apache.ibatis.annotations.Select;

import java.util.List;

public interface OaMemberMapper extends BaseMapper<OaMember> {

@Select("select id,flag from oa\_member where id=#{id} and password=#{password}")

MemberInfoDto login(String id, String password);

List<MemberListDto> getMemberList();

List<MemberListDto> getMemberListWithDirection(@Param("direction") String direction);

MemberListDto getMemberById(@Param("id") Long memberId);

@Select("select id,bname,direction from oa\_member where flag = 0")

List<MemberSimpleDto> getMembersNameId();

}

package com.banneroa.mapper;

import com.banneroa.pojo.OaSign;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;

public interface OaSignMapper extends BaseMapper<OaSign> {

}

package com.banneroa.pojo;

import java.time.LocalDateTime;

import com.baomidou.mybatisplus.annotation.FieldFill;

import com.baomidou.mybatisplus.annotation.TableField;

import java.io.Serializable;

import lombok.Data;

import lombok.EqualsAndHashCode;

@Data

@EqualsAndHashCode(callSuper = false)

public class OaLeave implements Serializable {

private static final long serialVersionUID = 1L;

private Long id;

private LocalDateTime beginTime;

private LocalDateTime endTime;

private Long memberId;

@TableField(fill = FieldFill.INSERT)

private LocalDateTime createTime;

private String reason;

private String proof;

private Integer status;

}

package com.banneroa.pojo;

import java.io.Serializable;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import javax.validation.constraints.NotBlank;

import javax.validation.constraints.NotNull;

@Data

@EqualsAndHashCode(callSuper = false)

public class OaMember implements Serializable {

private static final long serialVersionUID = 1L;

@ApiModelProperty(value = "学号")

@NotNull(message = "学号不为空!")

private Long id;

@ApiModelProperty(value = "姓名")

@NotBlank(message = "姓名不为空!")

private String bname;

@ApiModelProperty(value = "班级")

@NotBlank(message = "班级不为空!")

private String className;

@ApiModelProperty(value = "方向")

@NotBlank(message = "方向不为空!")

private String direction;

@NotBlank(message = "密码不为空!")

private String password;

@ApiModelProperty(hidden = true)

private int flag;

}

package com.banneroa.pojo;

import lombok.Data;

import java.time.LocalDateTime;

@Data

public class OaSign {

private Long id;

private Long memberId;

private LocalDateTime comeTime;

private LocalDateTime leaveTime;

}

package com.banneroa.service;

import org.springframework.web.multipart.MultipartFile;

public interface IMinioService {

String upload(MultipartFile multipartFile);

boolean delete(String path);

}

package com.banneroa.service.impl;

import cn.hutool.crypto.digest.DigestUtil;

import com.banneroa.service.IMinioService;

import io.minio.MinioClient;

import io.minio.PutObjectArgs;

import io.minio.RemoveObjectArgs;

import lombok.extern.slf4j.Slf4j;

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

import org.springframework.stereotype.Service;

import org.springframework.web.multipart.MultipartFile;

import javax.annotation.Resource;

import java.io.ByteArrayInputStream;

@Service

@Slf4j

public class MinioServiceImpl implements IMinioService {

@Value("${minio.bucket.files}")

private String bucket;

@Value("${minio.endpoint}")

private String address;

@Resource

private MinioClient minioClient;

@Override

public String upload(MultipartFile file) {

String contentType = file.getContentType();

try {

String fileName = DigestUtil.md5Hex(file.getBytes())+file.getOriginalFilename().substring(file.getOriginalFilename().lastIndexOf("."));

ByteArrayInputStream byteArrayInputStream = new ByteArrayInputStream(file.getBytes());

String path = "leaveProof/" + fileName;

PutObjectArgs putObjectArgs =

PutObjectArgs.builder().bucket(bucket).object(path)

.stream(byteArrayInputStream, byteArrayInputStream.available(), -1)

.contentType(contentType)

.build();

minioClient.putObject(putObjectArgs);

return address+"/"+bucket+"/"+path;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

@Override

public boolean delete(String pathUrl) {

RemoveObjectArgs removeObjectArgs = RemoveObjectArgs.builder().bucket(bucket).object(pathUrl).build();

try {

minioClient.removeObject(removeObjectArgs);

} catch (Exception e) {

log.error("minio remove file error. pathUrl:{}",pathUrl);

e.printStackTrace();

}

return true;

}

}

package com.banneroa.service.impl;

import cn.hutool.core.bean.BeanUtil;

import cn.hutool.core.util.StrUtil;

import com.banneroa.dto.LeaveDto;

import com.banneroa.dto.LeaveListDto;

import com.banneroa.dto.MemberListDto;

import com.banneroa.dto.PageResultDto;

import com.banneroa.mapper.OaMemberMapper;

import com.banneroa.pojo.OaLeave;

import com.banneroa.mapper.OaLeaveMapper;

import com.banneroa.pojo.OaMember;

import com.banneroa.service.OaLeaveService;

import com.banneroa.utils.AppHttpCodeEnum;

import com.banneroa.utils.BaseContext;

import com.banneroa.utils.ResponseResult;

import com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;

import com.baomidou.mybatisplus.core.toolkit.Wrappers;

import com.baomidou.mybatisplus.extension.service.impl.ServiceImpl;

import lombok.extern.slf4j.Slf4j;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import java.beans.beancontext.BeanContext;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.concurrent.atomic.AtomicInteger;

import java.util.stream.Collectors;

@Service

public class OaLeaveServiceImpl extends ServiceImpl<OaLeaveMapper, OaLeave> implements OaLeaveService {

@Resource

private OaLeaveMapper oaLeaveMapper;

@Resource

private OaMemberMapper oaMemberMapper;

@Override

public ResponseResult add(LeaveDto leaveDto) {

OaLeave oaLeave = BeanUtil.copyProperties(leaveDto, OaLeave.class);

oaLeave.setMemberId(BaseContext.getMember().getId());

oaLeave.setStatus(0);

int insert = oaLeaveMapper.insert(oaLeave);

return insert > 0 ? ResponseResult.okResult(200,"成功添加")

: ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_INVALID);

}

@Override

public ResponseResult getList(String currentPage,String id) {

List<OaLeave> oaLeaves;

Map<Long,MemberListDto> idAndMember = new HashMap<>();

Integer total;

if (BaseContext.getMember().getFlag()==0){

id = BaseContext.getMember().getId().toString();

oaLeaves = oaLeaveMapper.selectList(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getMemberId, id).orderByDesc(OaLeave::getCreateTime));

total = oaLeaveMapper.selectCount(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getMemberId, id));

}else {

if (StrUtil.isNotBlank(id)){

oaLeaves = oaLeaveMapper.selectList(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getMemberId, id).orderByDesc(OaLeave::getCreateTime));

total = oaLeaveMapper.selectCount(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getMemberId, id));

}else {

oaLeaves = oaLeaveMapper.getLeaveWithPage(page);

total = oaLeaveMapper.selectCount(null);

List<Long> memberIds = oaLeaves.stream().distinct().map(OaLeave::getMemberId).collect(Collectors.toList());

for (Long memberId : memberIds) {

idAndMember.put(memberId,oaMemberMapper.getMemberById(memberId));

}

}

}

List<LeaveListDto> leaves = oaLeaves.stream().map((item) -> {

LeaveListDto leaveListDto = BeanUtil.copyProperties(item, LeaveListDto.class);

if (item.getStatus() == 0) leaveListDto.setLeaveStatus("未审批");

else if(item.getStatus()==1) leaveListDto.setLeaveStatus("已通过");

else if (item.getStatus() == 2) leaveListDto.setProof("未通过");

if (!idAndMember.isEmpty()){

MemberListDto memberListDto = idAndMember.get(item.getMemberId());

leaveListDto.setLeaveCount(memberListDto.getLeaveCount());

leaveListDto.setBname(memberListDto.getBname());

leaveListDto.setDirection(memberListDto.getDirection());

leaveListDto.setClassName(memberListDto.getClassName());

}

return leaveListDto;

}).collect(Collectors.toList());

return ResponseResult.okResult(new PageResultDto(leaves,total));

}

@Override

public ResponseResult del(String leaveId,int flag) {

int delete;

if (flag == 0){

delete = oaLeaveMapper.delete(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getId, leaveId)

.eq(OaLeave::getMemberId, BaseContext.getMember().getId())

.eq(OaLeave::getStatus, 0));

}else {

delete = oaLeaveMapper.delete(Wrappers.<OaLeave>lambdaQuery()

.eq(OaLeave::getId, leaveId));

}

return delete > 0 ? ResponseResult.okResult(200,"成功删除")

: ResponseResult.errorResult(500, "无法删除已审批假条");

}

@Override

public ResponseResult handleLeave(String leaveId) {

if (StrUtil.isBlank(leaveId)) return ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_REQUIRE);

int status = oaLeaveMapper.getLeaveStatus(leaveId);

if (status == 0){

boolean update = this.update(Wrappers.<OaLeave>lambdaUpdate().set(OaLeave::getStatus, 1).eq(OaLeave::getId, leaveId));

return update ? ResponseResult.okResult(200,"已通过审批!")

: ResponseResult.errorResult(AppHttpCodeEnum.DATA\_NOT\_EXIST);

}else if (status == 1){

boolean update = this.update(Wrappers.<OaLeave>lambdaUpdate().set(OaLeave::getStatus, 2).eq(OaLeave::getId, leaveId));

return update ? ResponseResult.okResult(200,"已取消通过!")

: ResponseResult.errorResult(AppHttpCodeEnum.DATA\_NOT\_EXIST);

}else if (status == 2){

boolean update = this.update(Wrappers.<OaLeave>lambdaUpdate().set(OaLeave::getStatus, 1).eq(OaLeave::getId, leaveId));

return update ? ResponseResult.okResult(200,"已通过审批!")

: ResponseResult.errorResult(AppHttpCodeEnum.DATA\_NOT\_EXIST);

}else {

return ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_INVALID);

}

}

}

package com.banneroa.service.impl;

import cn.hutool.core.bean.BeanUtil;

import cn.hutool.core.util.StrUtil;

import com.banneroa.dto.MemberListDto;

import com.banneroa.dto.MemberInfoDto;

import com.banneroa.dto.MemberSimpleDto;

import com.banneroa.pojo.OaMember;

import com.banneroa.mapper.OaMemberMapper;

import com.banneroa.service.OaMemberService;

import com.baomidou.mybatisplus.extension.api.R;

import com.baomidou.mybatisplus.extension.service.impl.ServiceImpl;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

@Service

public class OaMemberServiceImpl extends ServiceImpl<OaMemberMapper, OaMember> implements OaMemberService {

@Resource

private OaMemberMapper oaMemberMapper;

@Override

public ResponseResult login(String id, String password) {

MemberInfoDto memberInfoDto = oaMemberMapper.login(id, password);

if (memberInfoDto == null) return ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_INVALID);

String token = AppJwtUtil.getToken(memberInfoDto.getId(), memberInfoDto.getFlag());

List<String> r = new ArrayList<>();

r.add(token);

r.add(memberInfoDto.getFlag() + "");

return ResponseResult.okResult(r);

}

@Override

public ResponseResult add(OaMember oaMember) {

if (oaMemberMapper.selectById(oaMember.getId()) != null) {

return ResponseResult.errorResult(AppHttpCodeEnum.DATA\_EXIST);

}

int insert = oaMemberMapper.insert(oaMember);

return insert > 0 ? ResponseResult.okResult(200, "成功添加")

: ResponseResult.errorResult(AppHttpCodeEnum.PARAM\_INVALID);

}

@Override

public ResponseResult getMemberList(String direction) {

if (BaseContext.getMember().getFlag() != 1) {

return ResponseResult.errorResult(500, "缺少权限");

}

List<MemberListDto> memberListDtos;

if (StrUtil.isBlank(direction)) {

memberListDtos = oaMemberMapper.getMemberList();

} else {

memberListDtos = oaMemberMapper.getMemberListWithDirection(direction);

}

return ResponseResult.okResult(memberListDtos);

}

@Override

public ResponseResult getMembersNameId() {

if (BaseContext.getMember().getFlag() != 1) {

return ResponseResult.errorResult(500, "缺少权限");

}

List<List<MemberSimpleDto>> r = new ArrayList<>();

List<MemberSimpleDto> javaer = new ArrayList<>();

List<MemberSimpleDto> pythoner = new ArrayList<>();

List<MemberSimpleDto> unityer = new ArrayList<>();

List<MemberSimpleDto> viewer = new ArrayList<>();

List<MemberSimpleDto> goer = new ArrayList<>();

List<MemberSimpleDto> memberSimpleDtos = oaMemberMapper.getMembersNameId();

memberSimpleDtos.forEach(i -> {

if (DirectionEnum.JAVA.getValues().equals(i.getDirection())) {

javaer.add(i);

} else if (DirectionEnum.PYTHON.getValues().equals(i.getDirection())) {

pythoner.add(i);

} else if (DirectionEnum.GO.getValues().equals(i.getDirection())) {

goer.add(i);

} else if (DirectionEnum.UNITY.getValues().equals(i.getDirection())) {

unityer.add(i);

} else {

viewer.add(i);

}

});

r.add(javaer);

r.add(pythoner);

r.add(goer);

r.add(unityer);

r.add(viewer);

return ResponseResult.okResult(r);

}

}

package com.banneroa.service.impl;

import cn.hutool.core.util.RandomUtil;

import cn.hutool.core.util.StrUtil;

import com.banneroa.mapper.OaSignMapper;

import com.banneroa.pojo.OaSign;

import com.banneroa.service.OaSignService;

import com.banneroa.utils.AppHttpCodeEnum;

import com.banneroa.utils.BaseContext;

import com.banneroa.utils.ResponseResult;

import com.baomidou.mybatisplus.extension.service.IService;

import com.baomidou.mybatisplus.extension.service.impl.ServiceImpl;

import org.springframework.data.redis.core.StringRedisTemplate;

import org.springframework.stereotype.Service;

import javax.annotation.Resource;

import java.util.concurrent.TimeUnit;

import static com.banneroa.utils.RedisConstants.SING\_CODE\_KEY;

import static com.banneroa.utils.RedisConstants.SING\_CODE\_TIME;

@Service

public class OaSignServiceImpl extends ServiceImpl<OaSignMapper, OaSign> implements OaSignService {

@Resource

private StringRedisTemplate stringRedisTemplate;

@Override

public ResponseResult sendCode(String objId) {

if (BaseContext.getMember().getFlag() != 1) {

return ResponseResult.errorResult(AppHttpCodeEnum.NEED\_ADMIND);

}

String code = RandomUtil.randomNumbers(6);

String key = SING\_CODE\_KEY + objId;

if (Boolean.TRUE.equals(stringRedisTemplate.hasKey(key))) {

return ResponseResult.errorResult(201, "请勿多次请求");

}

stringRedisTemplate.opsForValue().set(key, code, SING\_CODE\_TIME, TimeUnit.MINUTES);

return ResponseResult.okResult(code);

}

@Override

public ResponseResult getCode() {

String key = SING\_CODE\_KEY + BaseContext.getMember().getId();

String code = stringRedisTemplate.opsForValue().get(key);

if (StrUtil.isBlank(code)) {

return ResponseResult.errorResult(201, "尚无验证码，请联系管理员申请");

}

return ResponseResult.okResult(code);

}

}

package com.banneroa.service;

import com.banneroa.dto.LeaveDto;

import com.banneroa.pojo.OaLeave;

import com.banneroa.utils.ResponseResult;

import com.baomidou.mybatisplus.extension.service.IService;

public interface OaLeaveService extends IService<OaLeave> {

ResponseResult add(LeaveDto leaveDto);

ResponseResult getList(String currentPage,String id);

ResponseResult del(String leaveId,int flag);

ResponseResult handleLeave(String leaveId);

}

package com.banneroa.service;

import com.banneroa.pojo.OaMember;

import com.banneroa.utils.ResponseResult;

import com.baomidou.mybatisplus.extension.service.IService;

public interface OaMemberService extends IService<OaMember> {

ResponseResult login(String id, String password);

ResponseResult add(OaMember oaMember);

ResponseResult getMemberList(String direction);

ResponseResult getMembersNameId();

}

package com.banneroa.service;

import com.banneroa.mapper.OaSignMapper;

import com.banneroa.pojo.OaSign;

import com.banneroa.utils.ResponseResult;

import com.baomidou.mybatisplus.extension.service.IService;

public interface OaSignService extends IService<OaSign> {

ResponseResult sendCode(String objId);

ResponseResult getCode();

}

package com.banneroa.utils;

public enum AppHttpCodeEnum {

SUCCESS(200,"操作成功"),

NEED\_LOGIN(1,"需要登录后操作"),

LOGIN\_PASSWORD\_ERROR(2,"密码错误"),

TOKEN\_INVALID(50,"无效的TOKEN"),

TOKEN\_EXPIRE(51,"TOKEN已过期"),

TOKEN\_REQUIRE(52,"TOKEN是必须的"),

SIGN\_INVALID(100,"无效的SIGN"),

SIG\_TIMEOUT(101,"SIGN已过期"),

PARAM\_REQUIRE(500,"缺少参数"),

PARAM\_INVALID(501,"无效参数"),

PARAM\_IMAGE\_FORMAT\_ERROR(502,"图片格式有误"),

SERVER\_ERROR(503,"服务器内部错误"),

DATA\_EXIST(1000,"数据已经存在"),

USER\_DATA\_NOT\_EXIST(1001,"User数据不存在"),

DATA\_NOT\_EXIST(1002,"数据不存在"),

NO\_OPERATOR\_AUTH(3000,"无权限操作"),

NEED\_ADMIND(3001,"需要管理员权限");

int code;

String errorMessage;

AppHttpCodeEnum(int code, String errorMessage){

this.code = code;

this.errorMessage = errorMessage;

}

public int getCode() {

return code;

}

public String getMessage() {

return errorMessage;

}

}

package com.banneroa.utils;

import javax.crypto.SecretKey;

import javax.crypto.spec.SecretKeySpec;

public class AppJwtUtil {

private static final String TOKEN\_ENCRY\_KEY = "@CopyrightByBanner";

private static final int REFRESH\_TIME = 60\_000;

public static String getToken(Long id,Integer flag){

Map<String, Object> claimMaps = new HashMap<>();

claimMaps.put("id",id);

claimMaps.put("flag",flag);

long currentTime = System.currentTimeMillis();

return Jwts.builder()

.setId(UUID.randomUUID().toString())

.compact();

}

private static Jws<Claims> getJws(String token) {

return Jwts.parser()

.setSigningKey(generalKey())

.parseClaimsJws(token);

}

public static Claims getClaimsBody(String token) {

try {

return getJws(token).getBody();

}catch (ExpiredJwtException e){

return null;

}

}

public static JwsHeader getHeaderBody(String token) {

return getJws(token).getHeader();

}

public static int checkTokenTime(Claims claims) {

if(claims==null){

return 0;

}

try {

if (claims.getExpiration().after(new Date())){

return 1;

}

System.out.println(System.currentTimeMillis()-claims.getExpiration().getTime());

if(System.currentTimeMillis()-claims.getExpiration().getTime()<REFRESH\_TIME){

return 2;

}

return 0;

}catch (Exception e){

return 0;

}

}

public static SecretKey generalKey() {

byte[] encodedKey = Base64.getEncoder().encode(TOKEN\_ENCRY\_KEY.getBytes());

SecretKey key = new SecretKeySpec(encodedKey, 0, encodedKey.length, "AES");

return key;

}

}

package com.banneroa.utils;

import com.banneroa.dto.MemberInfoDto;

public class BaseContext {

private static ThreadLocal<MemberInfoDto> threadLocal = new ThreadLocal<>();

public static void setMember(MemberInfoDto memberLogin){

threadLocal.set(memberLogin);

}

public static MemberInfoDto getMember(){

return threadLocal.get();

}

public static void removeMember(){threadLocal.remove();}

}

package com.banneroa.utils;

public enum DirectionEnum {

JAVA("java"),

PYTHON("python"),

UNITY("unity"),

GO("go"),

VIEW("前端");

String values;

DirectionEnum(String values){

this.values=values;

}

public String getValues(){

return values;

}

}

package com.banneroa.utils;

import com.baomidou.mybatisplus.extension.api.R;

import org.springframework.http.HttpStatus;

import org.springframework.validation.BindingResult;

import org.springframework.validation.FieldError;

import org.springframework.web.bind.MethodArgumentNotValidException;

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

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

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

import java.util.List;

public class GlobalExceptionHandler {

@ExceptionHandler(MethodArgumentNotValidException.class)

public ResponseResult doMethodArgumentNotValidException(MethodArgumentNotValidException e) {

BindingResult bindingResult = e.getBindingResult();

List<FieldError> fieldErrors = bindingResult.getFieldErrors();

StringBuffer errors = new StringBuffer();

fieldErrors.forEach((item)->{

errors.append(item.getDefaultMessage()).append(',');

});

String em = errors.substring(0, errors.length() - 1);

return ResponseResult.errorResult(500,em);

}

}

package com.banneroa.utils;

import com.fasterxml.jackson.databind.DeserializationFeature;

import com.fasterxml.jackson.databind.ObjectMapper;

import com.fasterxml.jackson.databind.module.SimpleModule;

import com.fasterxml.jackson.databind.ser.std.ToStringSerializer;

import com.fasterxml.jackson.datatype.jsr310.deser.LocalDateDeserializer;

import com.fasterxml.jackson.datatype.jsr310.deser.LocalDateTimeDeserializer;

import com.fasterxml.jackson.datatype.jsr310.deser.LocalTimeDeserializer;

import com.fasterxml.jackson.datatype.jsr310.ser.LocalDateSerializer;

import com.fasterxml.jackson.datatype.jsr310.ser.LocalDateTimeSerializer;

import com.fasterxml.jackson.datatype.jsr310.ser.LocalTimeSerializer;

import java.math.BigInteger;

import java.time.LocalDate;

import java.time.LocalDateTime;

import java.time.LocalTime;

import java.time.format.DateTimeFormatter;

import static com.fasterxml.jackson.databind.DeserializationFeature.FAIL\_ON\_UNKNOWN\_PROPERTIES;

public class JacksonObjectMapper extends ObjectMapper {

public static final String DEFAULT\_DATE\_FORMAT = "yyyy-MM-dd";

public static final String DEFAULT\_DATE\_TIME\_FORMAT = "yyyy-MM-dd HH:mm";

public static final String DEFAULT\_TIME\_FORMAT = "HH:mm:ss";

public JacksonObjectMapper() {

super();

this.configure(FAIL\_ON\_UNKNOWN\_PROPERTIES, false);

this.getDeserializationConfig().withoutFeatures(DeserializationFeature.FAIL\_ON\_UNKNOWN\_PROPERTIES);

SimpleModule simpleModule = new SimpleModule()

.addDeserializer(LocalDateTime.class, new LocalDateTimeDeserializer(DateTimeFormatter.ofPattern(DEFAULT\_DATE\_TIME\_FORMAT)))

.addDeserializer(LocalDate.class, new LocalDateDeserializer(DateTimeFormatter.ofPattern(DEFAULT\_DATE\_FORMAT)))

.addDeserializer(LocalTime.class, new LocalTimeDeserializer(DateTimeFormatter.ofPattern(DEFAULT\_TIME\_FORMAT)))

.addSerializer(BigInteger.class, ToStringSerializer.instance)

.addSerializer(Long.class, ToStringSerializer.instance)

.addSerializer(LocalDateTime.class, new LocalDateTimeSerializer(DateTimeFormatter.ofPattern(DEFAULT\_DATE\_TIME\_FORMAT)))

.addSerializer(LocalDate.class, new LocalDateSerializer(DateTimeFormatter.ofPattern(DEFAULT\_DATE\_FORMAT)))

.addSerializer(LocalTime.class, new LocalTimeSerializer(DateTimeFormatter.ofPattern(DEFAULT\_TIME\_FORMAT)));

this.registerModule(simpleModule);

}

}

package com.banneroa.utils;

import cn.hutool.core.util.StrUtil;

import com.banneroa.dto.MemberInfoDto;

import io.jsonwebtoken.Claims;

import lombok.extern.slf4j.Slf4j;

import org.springframework.http.HttpHeaders;

import org.springframework.web.servlet.HandlerInterceptor;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@Slf4j

public class LoginCheckInterceptor implements HandlerInterceptor {

@Override

public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {

String url = request.getRequestURL().toString();

if (url.contains("doc.html")

|| url.contains("swagger")

|| url.contains("error")

|| url.contains("webjars")

|| url.contains("resource")

|| url.contains("index")) {

return true;

}

String token = request.getHeader(HttpHeaders.AUTHORIZATION);

response.setHeader("Content-type", "text/html;charset=UTF-8");

response.setCharacterEncoding("UTF-8");

if (StrUtil.isBlank(token)) {

response.setStatus(401);

response.getWriter().print("无权限访问!输入token!");

return false;

}

Claims claims;

try {

claims = AppJwtUtil.getClaimsBody(token);

} catch (Exception e) {

response.setStatus(401);

response.getWriter().print("无权限访问!");

return false;

}

int status = AppJwtUtil.checkTokenTime(claims);

if (status == 1 || status == 2) {

BaseContext.setMember(new MemberInfoDto(

(Long) claims.get("id"),

(int) claims.get("flag")));

return true;

} else {

response.setStatus(401);

response.getWriter().print("无权限访问!");

return false;

}

}

@Override

public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler, Exception ex) throws Exception {

BaseContext.removeMember();

}

}

package com.banneroa.utils;

import com.baomidou.mybatisplus.core.handlers.MetaObjectHandler;

import org.apache.ibatis.reflection.MetaObject;

import org.springframework.stereotype.Component;

import java.time.LocalDateTime;

@Component

public class MyMetaObjectHandler implements MetaObjectHandler {

@Override

public void insertFill(MetaObject metaObject) {

metaObject.setValue("createTime", LocalDateTime.now());

}

@Override

public void updateFill(MetaObject metaObject) {

metaObject.setValue("createTime", LocalDateTime.now());

}

}

package com.banneroa.utils;

public class RedisConstants {

public static final String SING\_CODE\_KEY = "s:c:";

public static final Integer SING\_CODE\_TIME = 5;

}

package com.banneroa.utils;

import lombok.Data;

import java.io.Serializable;

@Data

public class ResponseResult<T> implements Serializable {

private Integer code;

private String msg;

private T data;

public ResponseResult() {

this.code = 200;

}

public static ResponseResult errorResult(int code, String msg) {

ResponseResult result = new ResponseResult();

result.code = code;

result.msg = msg;

return result;

}

public static ResponseResult errorResult(AppHttpCodeEnum appHttpCodeEnum) {

ResponseResult result = new ResponseResult();

result.code = appHttpCodeEnum.getCode();

result.data = null;

result.msg = appHttpCodeEnum.getMessage();

return result;

}

public static <T> ResponseResult<T> okResult(T data) {

ResponseResult result = new ResponseResult();

result.data = data;

result.msg = null;

return result;

}

public static ResponseResult okResult(Integer code,String msg) {

ResponseResult result = new ResponseResult();

result.data = null;

result.msg = msg;

return result;

}

}

package com.banneroa.utils;

import cn.hutool.core.annotation.AnnotationUtil;

import org.springframework.context.annotation.Configuration;

import org.springframework.core.MethodParameter;

import org.springframework.core.annotation.AnnotationUtils;

import org.springframework.http.MediaType;

import org.springframework.http.converter.HttpMessageConverter;

import org.springframework.http.server.ServerHttpRequest;

import org.springframework.http.server.ServerHttpResponse;

import org.springframework.stereotype.Component;

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

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

import org.springframework.web.servlet.mvc.method.annotation.ResponseBodyAdvice;

public class ResponseResultHandler implements ResponseBodyAdvice<Object> {

@Override

public boolean supports(MethodParameter returnType, Class<? extends HttpMessageConverter<?>> converterType) {

if (returnType.getMethodAnnotation(ResponseBody.class) != null

|| AnnotationUtils.findAnnotation(returnType.getContainingClass(), ResponseBody.class) != null) {

return true;

}

return false;

}

@Override

public Object beforeBodyWrite(Object body, MethodParameter returnType, MediaType selectedContentType, Class<? extends HttpMessageConverter<?>> selectedConverterType, ServerHttpRequest request, ServerHttpResponse response) {

if (body instanceof ResponseResult){

return body;

}

return ResponseResult.okResult(body);

}

}

import request from '@/utils/request.js'

export function getLeavesApi(currentPage){

return request.get("/oa-admin/leave-list?currentPage="+currentPage)

}

export function addLeaveApi(leaveDto){

return request.post('/oa-leave/add',leaveDto)

}

export function handleLeaveAip(id){

return request.post("/oa-leave/handle/"+id)

}

export function getLeavesByIdApi(currentPage,userId){

return request.get("/oa-admin/leave-list?currentPage="+currentPage+"&id="+userId)

}

export function deleteLeaveApi(id){

return request.delete("/oa-leave/delete/"+id)

}import request from '@/utils/request.js'

export function getUserSignsApi(objId){

return null

}

export function sendCodeApi(objId){

return request.post("/oa-sign/send-code?objId="+objId)

}import request from '@/utils/request.js'

export function login(member) {

return request.post("/oa-member/login", member)

}

export function enrollUser(oaMember){

return request.post("/oa-member/add", oaMember)

}

export function getUserInfo() {

return request.get("/oa-admin/self")

}

export function getUserListApi(direction) {

return request.get("/oa-admin/member-list?direction="+direction)

}

export function addUser(oaMember){

return request.post("/oa-member/add",oaMember)

}

export function deleteUser(id){

return request.delete("/oa-admin/delete-member/"+id)

}

export function getUsersNameIdApi(){

return request.get("/oa-admin/member-name-id")

}

export function updateUser(oaMember){

return null

}<script setup>

</script>

<template>

<router-view/>

</template>

<style scoped>

</style>

<script setup>

import {

Edit,

Delete,

StarFilled,

Search,

Picture as IconPicture

} from '@element-plus/icons-vue'

import {ref, onMounted, watch} from 'vue'

import {handleLeaveAip, deleteLeaveApi, getLeavesApi} from '@/api/leave.js'

import {ElMessage, ElMessageBox} from 'element-plus'

import {useRouter} from 'vue-router';

import {userFlagStore} from '@/utils/stores/userFlag.js';

const router = useRouter()

const userFlagUtil = userFlagStore()

const leaves = ref([])

const drawer = ref(false)

const userinfo = ref({})

const buttontype = ref("")

const currentPage = ref(1)

const pageSize = ref(8)

const total = ref()

async function getLeaveList(current) {

let data = await getLeavesApi(current)

leaves.value = data.data.obj

total.value = data.data.total

}

const doDrawer = (row) => {

userinfo.value = row

drawer.value = true

var status = userinfo.value.status

if (status === 0) {

buttontype.value = 'info'

} else if (status === 1) {

buttontype.value = 'success'

} else {

buttontype.value = 'danger'

}

}

const handleLeave = async () => {

let data = await handleLeaveAip(userinfo.value.id)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getLeaveList(currentPage.value)

drawer.value = false

}

const addLeave = () => {

router.push('/leave/addLeave')

}

const deleteLeave = async (id) => {

ElMessageBox.confirm(

'确认删除该假条?',

'警告',

{

confirmButtonText: '确认',

cancelButtonText: '取消',

type: 'warning',

}

)

.then(async () => {

let data = await deleteLeaveApi(id)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getLeaveList(currentPage.value)

})

.catch(() => {

ElMessage({

type: 'info',

message: '取消删除',

})

})

}

const Nindex = (index) => {

return index + 1 + (page - 1) \* pagesize

}

watch(currentPage, (currentPagePost, currentPagePre) => {

getLeaveList(currentPagePost)

currentPage.value = currentPagePost

})

onMounted(() => {

getLeaveList(currentPage.value)

})

</script>

<template>

<el-card class="page-container">

<template #header>

<div class="header">

<span>假条管理</span>

<el-button v-if="userFlagUtil.userFlag==0" type="primary" @click="addLeave()">申请假条</el-button>

</div>

</template>

<el-table :data="leaves" stripe style="width: 100%">

<el-table-column label="序号" align='center' width="100" type="index" :index="Nindex"></el-table-column>

<el-table-column v-if="userFlagUtil.userFlag == 1" label="姓名" align='center' prop="bname"></el-table-column>

<el-table-column label="原因" align='center' prop="reason" width="200"></el-table-column>

<el-table-column label="开始时间" align='center' prop="beginTime"></el-table-column>

<el-table-column label="结束时间" align='center' prop="endTime"></el-table-column>

<el-table-column label="申请时间" align='center' prop="createTime"></el-table-column>

<el-table-column label="状态" align='center' prop="leaveStatus"></el-table-column>

<el-table-column label="操作" align='center' width="100">

<template #default="{ row }">

<el-button :icon="Search" circle plain type="primary"

@click="doDrawer(row)"></el-button>

<el-button :icon="Delete" circle plain type="danger" @click="deleteLeave(row.id)"></el-button>

</template>

</el-table-column>

<template #empty>

<el-empty description="没有数据"/>

</template>

</el-table>

<div class="page">

<el-pagination background layout="total, prev, pager, next" v-model:current-page="currentPage"

v-model:page-size="pageSize" :total="total"/>

</div>

</el-card>

<el-drawer class="dra" v-model="drawer" :with-header="false" direction="rtl">

<div v-if="userFlagUtil.userFlag==1">

<span>姓名：<b>{{ userinfo.bname }}</b></span><br>

<el-divider size>

<el-icon>

<star-filled/>

</el-icon>

</el-divider>

<span>学号：<b>{{ userinfo.memberId }}</b></span><br>

<el-divider>

<el-icon>

<star-filled/>

</el-icon>

</el-divider>

<span>班级：<b>{{ userinfo.className }}</b></span><br>

<el-divider>

<el-icon>

<star-filled/>

</el-icon>

</el-divider>

<span>方向：<b>{{ userinfo.direction }}</b></span><br>

<el-divider>

<el-icon>

<star-filled/>

</el-icon>

</el-divider>

<span>请假次数：<b>{{ userinfo.leaveCount }}</b></span><br>

<el-divider>

<el-icon>

<star-filled/>

</el-icon>

</el-divider>

</div>

<span>请假截图:</span><br>

<div class="block">

<el-image :src="userinfo.proof">

<template #error>

<div class="image-slot">

<el-icon><icon-picture /></el-icon>

</div>

</template>

</el-image>

</div>

<el-button v-if="userFlagUtil.userFlag==1" class="deal" :type="buttontype" @click="handleLeave()">

<el-icon style="margin: 0 2px;">

<Edit/>

</el-icon>

审批

</el-button>

</el-drawer>

</template>

<style lang="scss" scoped>

.page {

margin-top: 15px;

display: flex;

justify-content: center;

}

.dra span b {

float: right;

}

.el-divider--horizontal {

margin: 20px 0;

}

.header {

display: flex;

align-items: center;

justify-content: space-between;

}

.deal {

display: block;

margin: 0 auto;

margin-top: 2px;

width: 50%;

}

.block {

padding: 15px 0;

margin: 0 auto;

width: 35vh;

height: 44vh;

}

.el-image {

padding: 0 5px;

width: 100%;

height: 100%;

}

.image-slot {

display: flex;

justify-content: center;

align-items: center;

width: 100%;

height: 100%;

background: var(--el-fill-color-light);

color: var(--el-text-color-secondary);

font-size: 30px;

}

.image-slot .el-icon {

font-size: 30px;

}

</style><template>

<el-card class="page-container" style="height: 77vh;">

<template #header>

<div class="header">

<span>假条申请</span>

</div>

</template>

<div class="content">

<div class="left">

<p>证明截图</p>

:on-success="handleAvatarSuccess" :before-upload="beforeAvatarUpload">

<img v-if="imageUrl" :src="imageUrl" class="avatar"/>

<el-icon v-else class="avatar-uploader-icon">

<Plus/>

</el-icon>

</el-upload>

</div>

<div class="right">

<div>

<p>申请时间</p>

<el-date-picker v-model="time" type="datetimerange" range-separator="至" start-placeholder="开始日期"

end-placeholder="结束日期">

</el-date-picker>

</div>

<div class="bottom">

<p>申请原因</p>

<quill-editor theme="snow" v-model:content="leaveDto.reason" contentType="text">

</quill-editor>

</div>

<el-button type="primary" native-type="submit" @click="addLeave()">提交</el-button>

</div>

</div>

</el-card>

</template>

<script setup>

import {Plus} from '@element-plus/icons-vue'

import {ref} from 'vue'

import {addLeaveApi} from '@/api/leave.js'

import {ElMessage} from 'element-plus'

import dayjs from 'dayjs'

import {useRouter} from 'vue-router';

import { userTokenStore } from "@/utils/stores/token.js";

import {QuillEditor} from '@vueup/vue-quill'

import '@vueup/vue-quill/dist/vue-quill.snow.css'

const router = useRouter()

const userTokenUtil = userTokenStore()

const time = ref('')

const leaveDto = ref({

beginTime: "",

endTime: "",

reason: "",

proof:""

})

const imageUrl = ref('')

const headers = {

"Authorization": userTokenUtil.token

}

const addLeave = async () => {

leaveDto.value.beginTime = dayjs(time.value[0]).format('YYYY-MM-DD HH:mm:ss').toString()

leaveDto.value.endTime = dayjs(time.value[1]).format('YYYY-MM-DD HH:mm:ss').toString()

let data = await addLeaveApi(leaveDto.value)

if (data.code === 200) {

ElMessage.success(data.msg)

router.push('/')

} else {

ElMessage.error(data.msg)

}

}

const handleAvatarSuccess = (res, uploadFile) => {

console.log(res)

console.log(res.code)

console.log(res.code === 200)

if (res.code === 200){

ElMessage.success("上传成功")

imageUrl.value = URL.createObjectURL(uploadFile.raw)

leaveDto.value.proof = res.data

}else {

ElMessage.error(res.msg)

}

}

const beforeAvatarUpload = (rawFile) => {

if (rawFile.type !== 'image/jpeg' && rawFile.type !== 'image/png') {

ElMessage.error('上传图片必须是 JPG或png 格式!')

return false

} else if (rawFile.size / 1024 / 1024 > 2) {

ElMessage.error(' 图片大小不超过2MB!')

return false

}

return true

}

</script>

<style lang="scss" scoped>

.content {

width: 50%;

height: 50%;

margin: 0 auto;

display: flex;

justify-content: space-between;

}

.right button {

display: block;

width: 100%;

height: 5vh;

margin-top: 15vh;

}

.right {

margin-left: 2vh;

}

.right .bottom {

height: 41%;

}

.avatar-uploader .avatar {

width: 40vh;

height: 50vh;

display: block;

}

</style>

<style>

.avatar-uploader .el-upload {

border: 1px dashed var(--el-border-color);

border-radius: 6px;

cursor: pointer;

position: relative;

overflow: hidden;

transition: var(--el-transition-duration-fast);

}

.avatar-uploader .el-upload:hover {

border-color: var(--el-color-primary);

}

.el-icon.avatar-uploader-icon {

font-size: 28px;

color: #8c939d;

width: 40vh;

height: 50vh;

text-align: center;

}

</style><script setup>

import {

Edit,

Delete,

Search,

Picture as IconPicture

} from '@element-plus/icons-vue'

import {ref, onMounted, watch} from 'vue'

import {userIdStore} from '@/utils/stores/userId.js';

import {getLeavesByIdApi, handleLeaveAip, deleteLeaveApi} from '@/api/leave.js'

import {ElMessage} from 'element-plus'

const userIdUtil = userIdStore();

const currentPage = ref(1)

const pageSize = ref(8)

const total = ref()

const buttonType = (status) => {

if (status === 0) {

return 'info'

} else if (status === 1) {

return 'success'

} else {

return 'danger'

}

}

const member = ref({

...userIdUtil.userId

})

const leaves = ref([])

const drawer = ref(false)

async function getLeavesById(current) {

let data = await getLeavesByIdApi(current, member.value.id)

leaves.value = data.data.obj

total.value = data.data.total

}

const deleteLeave = async (id) => {

ElMessageBox.confirm(

'确认删除该假条?',

'警告',

{

confirmButtonText: '确认',

cancelButtonText: '取消',

type: 'warning',

}

)

.then(async () => {

let data = await deleteLeaveApi(id)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getLeavesById(currentPage.value)

})

.catch(() => {

ElMessage({

type: 'info',

message: '取消删除',

})

})

}

const handleLeave = async (row) => {

let data = await handleLeaveAip(row.id)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getLeavesById(currentPage.value)

}

const imgUrl = ref('')

const doDrawer = (row) => {

drawer.value = true

imgUrl.value = row.proof

}

const Nindex = (index) => {

return index + 1 + (page - 1) \* pagesize

}

watch(currentPage, (currentPagePost, currentPagePre) => {

getLeavesById(currentPagePost)

currentPage.value = currentPagePost

})

onMounted(() => {

getLeavesById(currentPage.value)

})

</script>

<template>

<el-card class="page-container">

<template #header>

<div class="header">

<span>{{ member.className }} {{ member.direction }} <b>{{ member.bname }}</b> 的假条</span>

<span>共请假：<b style="color: red;">{{ total }}</b>次</span>

</div>

</template>

<el-table :data="leaves" stripe style="width: 100%">

<el-table-column label="序号" align='center' width="100" type="index" :index="Nindex"></el-table-column>

<el-table-column label="原因" align='center' prop="reason"></el-table-column>

<el-table-column label="开始时间" align='center' prop="beginTime"></el-table-column>

<el-table-column label="结束时间" align='center' prop="endTime"></el-table-column>

<el-table-column label="申请时间" align='center' prop="createTime"></el-table-column>

<el-table-column label="状态" align='center' prop="leaveStatus"></el-table-column>

<el-table-column label="操作" align='center' width="150">

<template #default="{ row }">

<el-button :icon="Edit" circle plain :type="buttonType(row.status)" @click="handleLeave(row)"></el-button>

<el-button :icon="Delete" circle plain type="danger" @click="deleteLeave(row.id)"></el-button>

<el-button :icon="Search" circle plain type="primary"

@click="doDrawer(row)"></el-button>

</template>

</el-table-column>

<template #empty>

<el-empty description="没有数据"/>

</template>

</el-table>

<div class="page">

<el-pagination background

layout="total, prev, pager, next"

v-model:current-page="currentPage"

v-model:page-size="pageSize"

:total="total"/>

</div>

</el-card>

<el-drawer class="dra" v-model="drawer" :with-header="false" direction="rtl">

<span>请假截图:</span><br>

<div class="block">

<el-image :src="imgUrl">

<template #error>

<div class="image-slot">

<el-icon>

<icon-picture/>

</el-icon>

</div>

</template>

</el-image>

</div>

</el-drawer>

</template>

<style lang="scss" scoped>

.page {

margin-top: 15px;

display: flex;

justify-content: center;

}

.page-container {

min-height: 100%;

box-sizing: border-box;

.header {

display: flex;

align-items: center;

justify-content: space-between;

}

}

.block {

padding: 15px 0;

margin: 0 auto;

width: 35vh;

height: 44vh;

}

.el-image {

padding: 0 5px;

width: 100%;

height: 100%;

}

.image-slot {

display: flex;

justify-content: center;

align-items: center;

width: 100%;

height: 100%;

background: var(--el-fill-color-light);

color: var(--el-text-color-secondary);

font-size: 30px;

}

.image-slot .el-icon {

font-size: 30px;

}

</style><template>

<el-card class="page-container left">

<template #header>

<div class="header">

<span>成员列表</span>

</div>

</template>

<h3 v-for="(i,index) in directions" :key="index">

{{ i }}<br>

<el-button

v-for="item in users[index]"

text

bg

class="member"

@click="checkMemberSign(item.bname,item.id)"

>{{ item.bname }}

</el-button>

</h3>

</el-card>

<el-card class="page-container right">

<template #header>

<div class="header">

<span><b>{{ isChoose.name }}</b> 的签到列表</span>

<el-button type="primary" :disabled="noSend" @click="sendCode()">发送验证码</el-button>

</div>

</template>

<el-table :data="memberSigns" stripe style="width: 100%">

<el-table-column label="序号" align='center' width="100" type="index"></el-table-column>

<el-table-column label="签到时间" align='center' prop="comeTime"></el-table-column>

<el-table-column label="离开时间" align='center' prop="leaveTime"></el-table-column>

<el-table-column label="操作" align='center' width="200">

<template #default="{ row }">

<el-button :icon="Delete" circle plain type="danger" @click="deleteSign(row)"></el-button>

</template>

</el-table-column>

<template #empty>

<el-empty description="没有数据"/>

</template>

</el-table>

</el-card>

</template>

<script setup>

import {ref, onMounted} from "vue";

import {getUsersNameIdApi} from "@/api/user.js"

import {getUserSignsApi,sendCodeApi} from "@/api/sign.js"

import {ElMessage} from "element-plus"

const users = ref([])

const directions = ref(["java", "python", "go", "unity", "前端"])

const memberSigns = ref([])

const noSend = ref(true)

const isChoose = ref({})

async function getUserList() {

let data = await getUsersNameIdApi('')

users.value = data.data

}

const checkMemberSign = (name, id) => {

noSend.value = false

isChoose.value = {

"id": id,

"name": name

}

}

const getUserSigns = async (id) => {

let data = await getUserSignsApi(id);

}

const sendCode = async () => {

let data = await sendCodeApi(isChoose.value.id)

if (data.code === 200) {

ElMessage.success("发送成功："+data.data)

} else {

ElMessage.error(data.msg)

}

}

onMounted(() => {

getUserList()

})

</script>

<style lang="scss" scoped>

.page-container {

min-height: 100%;

box-sizing: border-box;

.header {

display: flex;

align-items: center;

justify-content: space-between;

}

}

.left {

float: left;

width: 49%;

}

.right {

float: right;

width: 49%;

}

.member {

width: 10%;

height: 5vh;

margin-top: 8px;

}

</style><script setup>

import { UserFilled, User, Lock, DataBoard, Folder } from '@element-plus/icons-vue'

import { login, enrollUser } from '@/api/user.js'

import { ref, reactive } from 'vue'

import { ElMessage } from 'element-plus'

import { useRouter } from 'vue-router'

import { userTokenStore } from '@/utils/stores/token.js'

import { userFlagStore } from '@/utils/stores/userFlag.js';

const isRegister = ref(false)

const router = useRouter()

const tokenStore = userTokenStore()

const userFlagUtil = userFlagStore()

const member = ref({

id: '20211574104',

password: '123456'

})

const enrollMember = ref({

id: '',

bname: '',

className: '',

direction: '',

password: '',

})

const rulesMember = ref({

id: '',

bname: '',

className: '',

direction: '',

password: '',

repassword: ''

})

async function loginto() {

let data = await login(member.value)

if (data.code === 200) {

tokenStore.setToken(data.data[0])

userFlagUtil.setUserFlag(data.data[1])

ElMessage({

message: '登录成功!',

type: 'success'

})

if (data.data[1] === '1') {

router.push('/admin')

} else {

router.push('/')

}

} else {

ElMessage({

message: data.msg,

type: 'error'

})

}

}

async function enroll() {

enrollMember.value = rulesMember.value

let data = await enrollUser(enrollMember.value)

if (data.code === 200) {

isRegister.value = false

ElMessage({

message: '注册成功!',

type: 'success'

})

} else {

ElMessage({

message: data.msg,

type: 'error'

})

}

}

const checkPass = (rule, value, callback) => {

if (value == null || value === '') {

callback(new Error('请再次输入密码'))

}

if (rulesMember.value.password !== value) {

callback(new Error('两次输入密码不一致'))

}

}

const rules = {

id: [

{ required: true, message: "请输入学号", trigger: 'blur' },

{ min: 11, max: 11, message: "学号为11位", trigger: 'blur' }

],

bname: { required: true, message: "请输入学号", trigger: 'blur' },

className: { required: true, message: "请输入班级", trigger: 'blur' },

password: [

{ required: true, message: "请输入密码", trigger: 'blur' },

{ min: 6, max: 12, message: "密码在6~12位之间", trigger: 'blur' }

],

repassword: [

{ validator: checkPass, trigger: 'blur' }

]

}

const directions = [

{

value: "java",

label: "java",

},

{

value: "python",

label: "python",

},

{

value: "unity",

label: "unity",

},

{

value: "go",

label: "go",

},

{

value: "前端",

label: "前端",

}

]

</script>

<template>

<el-row class="login-page">

<el-col :span="12" class="bg"></el-col>

<el-col :span="6" :offset="3" class="form">

<el-form ref="form" size="large" autocomplete="off" v-if="isRegister" :model="rulesMember" :rules="rules">

<el-form-item>

<h1>注册</h1>

</el-form-item>

<el-form-item prop="id">

<el-input :prefix-icon="UserFilled" v-model="rulesMember.id" placeholder="请输入学号"></el-input>

</el-form-item>

<el-form-item prop="bname">

<el-input :prefix-icon="User" v-model="rulesMember.bname" placeholder="请输入姓名"></el-input>

</el-form-item>

<el-form-item prop="className">

<el-input :prefix-icon="DataBoard" v-model="rulesMember.className"

placeholder="请输入班级,如:数据211"></el-input>

</el-form-item>

<el-form-item>

<el-select v-model="rulesMember.direction" placeholder="请选择方向">

<el-option v-for="item in directions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

<el-form-item prop="password">

<el-input :prefix-icon="Lock" v-model="rulesMember.password" type="password" placeholder="请输入密码"

show-password></el-input>

</el-form-item>

<el-form-item prop="repassword">

<el-input :prefix-icon="Lock" v-model="rulesMember.repassword" type="password" placeholder="请再次输入密码"

show-password></el-input>

</el-form-item>

<el-form-item>

<el-button class="button" type="primary" @click="enroll" auto-insert-space>

注册

</el-button>

</el-form-item>

<el-form-item class="flex">

<el-link type="info" :underline="false" @click="isRegister = false">

← 返回

</el-link>

</el-form-item>

</el-form>

<el-form ref="form" size="large" autocomplete="off" :model="member" :rules="rules" v-else>

<el-form-item>

<h1>登录</h1>

</el-form-item>

<el-form-item prop="id">

<el-input :prefix-icon="User" placeholder="请输入学号" v-model="member.id"></el-input>

</el-form-item>

<el-form-item prop="password">

<el-input name="password" :prefix-icon="Lock" type="password" placeholder="请输入密码"

v-model="member.password" show-password></el-input>

</el-form-item>

<el-form-item class="flex">

<div class="flex">

<el-checkbox>记住我</el-checkbox>

<el-link type="primary" :underline="false">忘记密码？</el-link>

</div>

</el-form-item>

<el-form-item>

<el-button class="button" type="primary" auto-insert-space @click="loginto">登录</el-button>

</el-form-item>

<el-form-item class="flex">

<el-link type="info" :underline="false" @click="isRegister = true">

注册 →

</el-link>

</el-form-item>

</el-form>

</el-col>

</el-row>

</template>

<style lang="scss" scoped>

/\* 样式 \*/

.login-page {

height: 100vh;

background-color: #fff;

.bg {

background:

url('@/assets/logo.png') no-repeat center / cover;

border-radius: 0 20px 20px 0;

}

.form {

display: flex;

flex-direction: column;

justify-content: center;

user-select: none;

.title {

margin: 0 auto;

}

.button {

width: 100%;

}

.flex {

width: 100%;

display: flex;

justify-content: space-between;

}

}

}

</style><template>

<el-card class="page-container">

<template #header>

<div class="header">

<span>基本资料</span>

</div>

</template>

<el-row>

<el-col :span="12">

<el-form :model="user" label-width="100px" size="large">

<el-form-item label="学号">

<el-input v-model="user.id" disabled></el-input>

</el-form-item>

<el-form-item label="姓名">

<el-input v-model="user.bname"></el-input>

</el-form-item>

<el-form-item label="班级">

<el-input v-model="user.className"></el-input>

</el-form-item>

<el-form-item label="方向">

<el-select v-model="user.direction" placeholder="请选择方向">

<el-option v-for="item in directions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

<el-form-item>

<el-button type="primary">提交修改</el-button>

</el-form-item>

</el-form>

</el-col>

</el-row>

</el-card>

</template>

<script setup>

import { ref } from 'vue';

import { userInfoStore } from '@/utils/stores/userInfo.js';

const myuser = userInfoStore()

const user = ref({

...myuser.userInfo

})

const directions = [

{

value: "java",

label: "java",

},

{

value: "python",

label: "python",

},

{

value: "unity",

label: "unity",

},

{

value: "go",

label: "go",

},

{

value: "前端",

label: "前端",

}

]

</script>

<style lang="scss" scoped></style><script setup>

import {

Edit,

Delete,

Search

} from '@element-plus/icons-vue'

import {ref, onMounted} from 'vue'

import {getUserListApi, addUser, deleteUser, updateUser} from '@/api/user.js'

import {ElMessage, ElMessageBox} from 'element-plus'

import {useRouter} from 'vue-router';

import {userIdStore} from '@/utils/stores/userId.js';

import {userInfoStore} from '@/utils/stores/userInfo.js'

const members = ref([])

const dialogVisible = ref(false)

const dialogTitle = ref('')

const dialogType = ref()

const dialogPass = ref(false)

const router = useRouter()

const selectDirection = ref('')

const memberModel = ref({

id: "",

bname: "",

className: "",

direction: "",

password: ""

})

async function doMember() {

if (dialogType.value === 0) {

let data = await addUser(memberModel.value)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getUserList()

dialogVisible.value = false

} else {

let data = await updateUser(memberModel.value)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getUserList()

dialogVisible.value = false

}

}

function doShow(row) {

dialogVisible.value = true

if (typeof (row) == "undefined") {

dialogTitle.value = "添加成员"

dialogType.value = 0

dialogPass.value = false

} else {

dialogTitle.value = "编辑成员"

dialogPass.value = true

memberModel.value = row

dialogType.value = 1

}

}

function deleteMember(row) {

ElMessageBox.confirm(

'确认删除该成员?',

'警告',

{

confirmButtonText: '确认',

cancelButtonText: '取消',

type: 'warning',

}

)

.then(async () => {

let data = await deleteUser(row.id)

if (data.code === 200) {

ElMessage.success(data.msg)

} else {

ElMessage.error(data.msg)

}

getUserList()

})

.catch(() => {

ElMessage({

type: 'info',

message: '取消删除',

})

})

}

function cancel() {

dialogVisible.value = false

memberModel.value = {}

}

async function getUserList() {

let data = await getUserListApi(userInfoUtil.userInfo.direction)

members.value = data.data

}

const getUserLeaveInfo = (row) => {

userIdUtil.setUserId(JSON.parse(JSON.stringify(row)))

router.push('/admin/leave/info')

}

const getUserByDirection = async () => {

let data = await getUserListApi(selectDirection.value)

members.value = data.data

}

const rules = {

id: [

{required: true, message: '请输入学号', trigger: 'blur'},

],

bname: [

{required: true, message: '请输入姓名', trigger: 'blur'},

],

className: [

{required: true, message: '请输入班级', trigger: 'blur'},

],

password: [

{required: true, message: '请输入密码', trigger: 'blur'},

{min: 6, max: 12, message: "密码在6~12位之间", trigger: 'blur'}

]

}

const directions = [

{

value: "java",

label: "java",

},

{

value: "python",

label: "python",

},

{

value: "unity",

label: "unity",

},

{

value: "go",

label: "go",

},

{

value: "前端",

label: "前端",

}

]

onMounted(() => {

selectDirection.value = userInfoUtil.userInfo.direction

getUserList()

})

</script>

<template>

<el-card class="page-container">

<template #header>

<div class="header">

<span>成员管理</span>

<div class="extra">

<el-select style="margin-right: 10px;" v-model="selectDirection" @change="getUserByDirection()"

clearable placeholder="请选择方向">

<el-option v-for="item in directions" :key="item.value" :label="item.label" :value="item.value"/>

</el-select>

<el-button type="primary" @click="doShow()">添加成员</el-button>

</div>

</div>

</template>

<el-table :data="members" stripe style="width: 100%">

<el-table-column label="序号" align='center' width="100" type="index"></el-table-column>

<el-table-column label="学号" align='center' prop="id"></el-table-column>

<el-table-column label="姓名" align='center' prop="bname"></el-table-column>

<el-table-column label="班级" align='center' prop="className"></el-table-column>

<el-table-column label="方向" align='center' prop="direction"></el-table-column>

<el-table-column label="假条数" align='center' prop="leaveCount"></el-table-column>

<el-table-column label="操作" align='center' width="200">

<template #default="{ row }">

<el-button :icon="Edit" circle plain type="primary" @click="doShow(row)"></el-button>

<el-button :icon="Delete" circle plain type="danger" @click="deleteMember(row)"></el-button>

<el-button :icon="Search" circle plain @click="getUserLeaveInfo(row)"/>

</template>

</el-table-column>

<template #empty>

<el-empty description="没有数据"/>

</template>

</el-table>

</el-card>

<el-dialog v-model="dialogVisible" :title="dialogTitle" width="30%">

<el-form :model="memberModel" :rules="rules" label-width="100px" style="padding-right: 30px">

<el-form-item label="学号" prop="id">

<el-input v-model="memberModel.id"></el-input>

</el-form-item>

<el-form-item label="姓名" prop="bname">

<el-input v-model="memberModel.bname"></el-input>

</el-form-item>

<el-form-item label="班级" prop="className">

<el-input v-model="memberModel.className" placeholder="如:数据211"></el-input>

</el-form-item>

<el-form-item label="密码" prop="password">

<el-input v-model="memberModel.password" :disabled="dialogPass" type="password" show-password></el-input>

</el-form-item>

<el-form-item label="方向">

<el-select v-model="memberModel.direction" placeholder="请选择方向">

<el-option v-for="item in directions" :key="item.value" :label="item.label" :value="item.value"/>

</el-select>

</el-form-item>

</el-form>

<template #footer>

<span class="dialog-footer">

<el-button @click="cancel">取消</el-button>

<el-button type="primary" @click="doMember()"> 确认 </el-button>

</span>

</template>

</el-dialog>

</template>

<style lang="scss" scoped>

.page-container {

min-height: 100%;

box-sizing: border-box;

.header {

display: flex;

align-items: center;

justify-content: space-between;

}

}

</style><template>

重置密码

</template>import './assets/main.scss'

import { createApp } from 'vue'

import ElementPlus from 'element-plus'

import 'element-plus/dist/index.css'

import router from './utils/router'

import App from './App.vue'

import { createPinia } from 'pinia'

import { createPersistedState } from 'pinia-persistedstate-plugin'

import zhCn from 'element-plus/dist/locale/zh-cn.mjs'

const app = createApp(App)

const pinia = createPinia()

const persist = createPersistedState()

pinia.use(persist)

app.use(ElementPlus,{

locale: zhCn

})

app.use(router)

app.use(pinia)

app.mount('#app')

zhCn.el.pagination = {

goto: "跳转至",

pageClassifier: "",

pagesize: "",

total: "共 {total} 条",

};

import axios from "axios";

import { userTokenStore } from "@/utils/stores/token.js";

import { ElMessage } from 'element-plus'

import router from '@/utils/router'

const baseURL = "/banneroa/"

const instance = axios.create({baseURL})

instance.interceptors.response.use(

result=>{

return result.data;

},

err=>{

if(err.response.status === 401){

ElMessage.error(err.response.data)

router.push('/login')

}

}

)

instance.interceptors.request.use(

(config)=>{

const tokenStore = userTokenStore()

if(tokenStore.token){

config.headers.Authorization = tokenStore.token

}

return config

},

(err)=>{

Promise.reject(err)

}

)

export default instanceimport { createRouter, createWebHistory } from 'vue-router'

import AdminLayoutVue from '@/views/AdminLayout.vue'

import UserLayoutVue from '@/views/UserLayout.vue'

import LoginVue from '@/components/user/Login.vue'

import UserManagerVue from '@/components/user/UserManage.vue'

import LeaveManageVue from '@/components/leave/LeaveManage.vue'

import SignManageVue from '@/components/sign/SignManage.vue'

import AddLeaveVue from '@/components/leave/UserAddLeave.vue'

import UserInfoVue from '@/components/user/UserInfo.vue'

import UserResetPasswordVue from '@/components/user/UserResetPassword.vue'

import UserLeaveVue from '@/components/leave/UserLeave.vue'

const routers = [

{ path: '/login', component: LoginVue },

{

path: '/admin',

redirect: '/admin/user/manage',

component: AdminLayoutVue,

children: [

{ path: '/admin/user/manage', component: UserManagerVue },

{ path: '/admin/leave/manage', component: LeaveManageVue },

{ path: '/admin/sign/manage', component: SignManageVue},

{ path: '/admin/leave/info', component: UserLeaveVue},

{ path: '/admin/user/info', component: UserInfoVue },

{ path: '/admin/user/password', component: UserResetPasswordVue }

]

},

{

path: '/',

redirect: '/leave/manage',

component: UserLayoutVue,

children: [

{ path: '/leave/manage', component: LeaveManageVue},

{ path: '/leave/addLeave', component: AddLeaveVue},

{ path: '/user/info', component: UserInfoVue },

{ path: '/user/password', component: UserResetPasswordVue }

]

}

]

const router = createRouter({

history: createWebHistory(),

routes: routers

})

export default router

import { defineStore } from 'pinia'

import { ref } from 'vue'

export const userTokenStore = defineStore('token', () => {

const token = ref('')

const setToken = (newToken) => {

token.value = newToken

}

const removeToken = () => {

token.value = ''

}

return {

token, setToken, removeToken

}

}, {

})import { defineStore } from 'pinia'

import { ref } from 'vue'

export const userFlagStore = defineStore('userFlag', () => {

const userFlag = ref('')

const setUserFlag = (newInfo) => {

userFlag.value = newInfo

}

const removeUserFlag = () => {

{

userFlag.value = ''

}

}

return { userFlag, setUserFlag, removeUserFlag }

}, { persist: true })import { defineStore } from 'pinia'

import { ref } from 'vue'

export const userIdStore = defineStore('userId', () => {

const userId = ref({})

const setUserId = (newInfo) => {

userId.value = newInfo

}

const removeUserId = () => {

{

userId.value = {}

}

}

return { userId, setUserId, removeUserId }

}, { persist: true })import { defineStore } from 'pinia'

import { ref } from 'vue'

export const userInfoStore = defineStore('userInfo', () => {

const userInfo = ref({})

const setInfo = (newInfo) => {

userInfo.value = newInfo

}

const removeInfo = () => {

{

userInfo.value = {}

}

}

return { userInfo, setInfo, removeInfo }

}, { persist: true })

import { ref } from 'vue'

export const userInfoStore = defineStore('userInfo', () => {

const userInfo = ref({})

const setInfo = (newInfo) => {

userInfo.value = newInfo

}

const removeInfo = () => {

{

userInfo.value = {}

}

}

return { userInfo, setInfo, removeInfo }

}, { persist: true })<script setup>

import {

Management,

Promotion,

UserFilled,

User,

EditPen,

SwitchButton,

CaretBottom,

Expand

} from '@element-plus/icons-vue'

import { getUserInfo } from '@/api/user.js'

import { ref, onMounted } from 'vue';

import { userInfoStore } from '@/utils/stores/userInfo.js'

import { userTokenStore } from '@/utils/stores/token.js';

import { ElMessage,ElMessageBox } from 'element-plus';

import { useRouter } from 'vue-router';

const userInfo = userInfoStore()

const token = userTokenStore()

const router = useRouter()

const user = ref({

id: '',

bname: '',

className: '',

direction: ''

})

const handleCommand = (command) => {

if (command === 'logout') {

ElMessageBox.confirm(

'确认退出吗?',

'警告',

{

confirmButtonText: '确认',

cancelButtonText: '取消',

type: 'warning',

}

)

.then(() => {

token.removeToken()

userInfo.removeInfo()

router.push('/login')

ElMessage.success("退出成功!")

})

.catch(() => {

ElMessage({

type: 'info',

message: '取消退出!',

})

})

} else {

router.push('/admin/user/' + command)

}

}

onMounted(async function () {

let data = await getUserInfo()

user.value = data.data

userInfo.setInfo(user.value)

})

</script>

<template>

<el-container class="layout-container">

<el-aside width="200px">

<div class="el-aside\_\_logo"></div>

<el-menu active-text-color="#ffd04b" background-color="#232323" text-color="#fff" router>

<el-menu-item index="/admin/user/manage">

<el-icon>

<Management />

</el-icon>

<span>成员管理</span>

</el-menu-item>

<el-menu-item index="/admin/leave/manage">

<el-icon>

<Promotion />

</el-icon>

<span>假条管理</span>

</el-menu-item>

<el-menu-item index="/admin/sign/manage">

<el-icon>

<Expand />

</el-icon>

<span>签到管理</span>

</el-menu-item>

</el-container>

</el-container>

</template>