**管理员接口文档（登录与注册）**

**企业员工考勤系统 - 管理员接口文档**

**文档信息**

|  |  |
| --- | --- |
| 项目名称 | 企业员工考勤系统 - 管理端 |
| 版本号 | v1.0.0 |
| 最后更新时间 | 2023-11-20 |
| 技术栈 | SpringBoot 2.3.7 + Vue.js 2.x + MyBatis |
| 协议 | HTTP/HTTPS |
| 数据格式 | application/x-www-form-urlencoded |

**1. 管理员认证模块**

**1.1 管理员登录接口**

**接口描述**：管理员账号密码登录

**请求URL**：/login

**请求方式**：POST

**Content-Type**：application/x-www-form-urlencoded

**请求参数**：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名 | 类型 | 必填 | 描述 | 示例值 |
| name | String | 是 | 管理员账号 | admin |
| pwd | String | 是 | 密码 | 123456 |

**请求示例**：

|  |
| --- |
| HTTP POST /login HTTP/1.1 Host: yourdomain.com Content-Type: application/x-www-form-urlencoded  name=admin&pwd=123456 |

**响应参数**：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| - | Boolean | true-成功，false-失败 |

**成功响应示例**：

|  |
| --- |
| JSON true |

**失败响应示例**：

|  |
| --- |
| JSON false |

**状态码说明**：

|  |  |
| --- | --- |
| 状态码 | 说明 |
| 200 | 请求成功 |
| 400 | 参数错误 |
| 401 | 用户名或密码错误 |
| 500 | 服务器内部错误 |

**前端调用示例**：

|  |
| --- |
| JavaScript axios.post("login", "name=" + this.form.username + "&pwd=" + this.form.password)  .then(resp => {  if (resp.data === true) {  location.href = "/main";  this.$message.success('登录成功！');  } else {  this.$message.error('登录失败，请检查用户名和密码');  }  }) |

**1.2 管理员注册接口**

**接口描述**：注册新管理员账号

**请求URL**：/register

**请求方式**：POST

**Content-Type**：application/x-www-form-urlencoded

**请求参数**：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名 | 类型 | 必填 | 描述 | 示例值 |
| name | String | 是 | 管理员账号 | newadmin |
| pwd | String | 是 | 密码 | 123456 |

**请求示例**：

|  |
| --- |
| HTTP POST /register HTTP/1.1 Host: yourdomain.com Content-Type: application/x-www-form-urlencoded  name=newadmin&pwd=123456 |

**响应参数**：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| - | Boolean | true-成功，false-失败 |

**成功响应示例**：

|  |
| --- |
| JSON true |

**失败响应示例**：

|  |
| --- |
| JSON false |

**状态码说明**：

|  |  |
| --- | --- |
| 状态码 | 说明 |
| 200 | 注册成功 |
| 400 | 参数错误 |
| 409 | 用户名已存在 |
| 500 | 服务器内部错误 |

**前端调用示例**：

|  |
| --- |
| JavaScript axios.post("register", "name=" + this.formData.username + "&pwd=" + this.formData.password)  .then(resp => {  if (resp.data === true) {  this.$message.success('注册成功');  this.dialogTableVisible4add = false;  } else {  this.$message.error('注册失败，用户名重复');  }  }) |

**2. 安全规范**

**2.1 密码策略**

1. 密码存储必须使用BCrypt加密
2. 密码长度限制：6-20位字符
3. 建议包含字母、数字和特殊字符

**2.2 会话管理**

1. 使用HttpOnly的Session Cookie
2. 会话超时时间：30分钟
3. 最大并发会话数：3个

**2.3 权限控制**

1. 管理员角色分级：

* 超级管理员：所有权限
* 普通管理员：部分权限

**3. 接口安全要求**

1. **请求验证**：

* 所有敏感接口需验证管理员登录状态
* 关键操作需二次验证（如修改密码）

1. **数据加密**：

* 敏感数据传输需加密（建议HTTPS）
* 密码字段必须加密传输（当前示例为教学简化）

1. **防暴力破解**：

* 登录失败5次后锁定账号30分钟
* 添加图形验证码机制

**4. 错误码体系**

|  |  |  |
| --- | --- | --- |
| 错误码 | 说明 | 解决方案 |
| 4001 | 无效的用户名或密码 | 检查输入或联系超级管理员重置 |
| 4002 | 账号已被锁定 | 30分钟后重试或联系管理员 |
| 4003 | 会话已过期 | 重新登录 |
| 4004 | 权限不足 | 申请相应权限 |
| 5001 | 系统繁忙 | 稍后重试 |

**5. 后端实现建议**

**5.1 登录接口实现**

|  |
| --- |
| Java @PostMapping("/login") public ResponseEntity<Boolean> login(  @RequestParam String name,  @RequestParam String pwd,  HttpSession session) {    // 1. 验证账号密码  Admin admin = adminService.findByName(name);  if (admin == null || !passwordEncoder.matches(pwd, admin.getPassword())) {  return ResponseEntity.ok(false);  }    // 2. 设置会话  session.setAttribute("admin", admin);    // 3. 记录登录日志  logService.recordLoginLog(admin.getId(), getClientIP());    return ResponseEntity.ok(true); } |

**5.2 注册接口实现**

|  |
| --- |
| Java @PostMapping("/register") public ResponseEntity<Boolean> register(  @RequestParam String name,  @RequestParam String pwd) {    // 1. 检查用户名是否存在  if (adminService.existsByName(name)) {  return ResponseEntity.ok(false);  }    // 2. 创建管理员账号  Admin admin = new Admin();  admin.setUsername(name);  admin.setPassword(passwordEncoder.encode(pwd));  admin.setRole("NORMAL"); // 默认普通管理员    adminService.save(admin);    return ResponseEntity.ok(true); } |

**6. 前端表单验证规则**

|  |
| --- |
| JavaScript rules: {  username: [  {required: true, message: '请输入用户名', trigger: 'blur'},  {min: 2, max: 10, message: '长度在 2 到 10 个字符', trigger: 'blur'}  ],  password: [  {required: true, message: '请输入密码', trigger: 'blur'},  {min: 6, max: 20, message: '长度在 6 到 20 个字符', trigger: 'blur'}  ] } |

**7. 接口测试用例**

**7.1 登录测试用例**

|  |  |  |
| --- | --- | --- |
| 测试场景 | 输入数据 | 预期结果 |
| 正确账号密码 | name=admin, pwd=123456 | true |
| 错误密码 | name=admin, pwd=wrong | false |
| 不存在的账号 | name=none, pwd=123456 | false |
| 缺失参数 | name=admin | 400错误 |

**7.2 注册测试用例**

|  |  |  |
| --- | --- | --- |
| 测试场景 | 输入数据 | 预期结果 |
| 新用户名 | name=new, pwd=123456 | true |
| 已存在的用户名 | name=admin, pwd=123456 | false |
| 密码过短 | name=new, pwd=123 | 前端拦截 |

**8. 部署建议**

1. **生产环境配置**：

|  |
| --- |
| YAML spring:  session:  timeout: 1800 # 30分钟  security:  user:  name: superadmin  password: $2a$10$N9qo8uLOickgx2ZMRZoMy.Mrq4HkJpWrDi7WYCHyqG2bYFQ3JQlWq # bcrypt加密密码 |

1. **Nginx配置建议**：

|  |
| --- |
| Nginx server {  listen 443 ssl;  server\_name yourdomain.com;    ssl\_certificate /path/to/cert.pem;  ssl\_certificate\_key /path/to/key.pem;    location / {  proxy\_pass http://localhost:8080;  proxy\_set\_header Host $host;  proxy\_set\_header X-Real-IP $remote\_addr;  } } |

**附录**

**A. 管理员账号初始数据**

|  |  |  |
| --- | --- | --- |
| 用户名 | 密码(加密前) | 角色 |
| admin | admin123 | 超级管理员 |
| manager | manager123 | 普通管理员 |

**B. 密码加密工具类**

|  |
| --- |
| Java public class PasswordUtil {  private static final BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();    public static String encode(String rawPassword) {  return encoder.encode(rawPassword);  }    public static boolean matches(String rawPassword, String encodedPassword) {  return encoder.matches(rawPassword, encodedPassword);  } } |

**文档维护说明**：

1. 接口变更时需同步更新版本号和修改日期
2. 新增接口需补充到相应模块
3. 安全策略调整需在文档中明确标注

**生产环境注意事项**：

1. 必须启用HTTPS协议
2. 建议添加操作审计日志
3. 定期更换管理员密码
4. 实现管理员账号锁定机制