**员工人事变动接口——updateDutyDept**

**企业员工考勤系统 - 员工人事变动接口文档**

**文档信息**

|  |  |
| --- | --- |
| 项目名称 | 企业员工考勤系统 - 人事变动模块 |
| 版本号 | v1.3.0 |
| 最后更新时间 | 2023-11-23 |
| 技术栈 | SpringBoot 2.3.7 + MyBatis + PageHelper |
| 协议 | HTTP/HTTPS |
| 认证方式 | Session-Based Authentication |

**1. 员工人事变动模块**

**1.1 分页查询员工列表（带部门职务）**

**接口描述**：分页查询员工信息列表（包含部门和职务信息）

**请求URL**：/emp/selectByPage

**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 |
| currentPage | Int | 是 | 当前页码（从1开始） | 1 |
| pageSize | Int | 是 | 每页记录数 | 8 |

**响应参数**：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| code | Int | 响应码 |
| data | Array | 员工数据列表 |
| └─ number | Long | 员工工号 |
| └─ name | String | 员工姓名 |
| └─ dept\_name | String | 部门名称 |
| └─ duty\_name | String | 职务名称 |
| pageNum | Int | 当前页码 |
| total | Long | 总记录数 |

**成功响应示例**：

|  |
| --- |
| JSON {  "code": 200,  "data": [  {  "number": 1001,  "name": "张三",  "dept\_name": "技术部",  "duty\_name": "高级工程师"  }  ],  "pageNum": 1,  "total": 25 } |

**1.2 获取部门列表**

**接口描述**：获取所有部门信息

**请求URL**：/emp/getDeptData

**请求方式**：POST

**响应参数**：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| code | Int | 响应码 |
| data | Array | 部门列表 |
| └─ dept\_id | Long | 部门ID |
| └─ dept\_name | String | 部门名称 |

**1.3 获取职务列表**

**接口描述**：获取所有职务信息

**请求URL**：/emp/getDutyData

**请求方式**：POST

**响应参数**：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| code | Int | 响应码 |
| data | Array | 职务列表 |
| └─ duty\_id | Long | 职务ID |
| └─ duty\_name | String | 职务名称 |

**1.4 更新员工部门职务**

**接口描述**：修改员工的部门和职务信息

**请求URL**：/emp/updateDD

**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 |
| number | Long | 是 | 员工工号 | 1001 |
| dept\_id | Long | 是 | 新部门ID | 2 |
| duty\_id | Long | 是 | 新职务ID | 3 |
| currentPage | Int | 是 | 当前页码（从1开始） | 1 |
| pageSize | Int | 是 | 每页记录数 | 8 |

**响应参数**：同1.1分页查询接口

**2. 接口规范说明**

**2.1 权限要求**

* 需管理员权限
* 超级管理员可修改所有员工信息
* 部门主管仅可修改本部门员工信息

**2.2 业务规则**

1. 部门变更需记录变更历史
2. 职务变更需验证职级合理性
3. 部门主管变更需特殊权限

**2.3 数据验证**

|  |  |
| --- | --- |
| 字段 | 验证规则 |
| dept\_id | 必须为有效部门ID |
| duty\_id | 必须为有效职务ID |
| dept\_id+duty\_id | 职务必须属于指定部门 |

**3. 后端实现示例**

**3.1 服务层实现**

|  |
| --- |
| Java @Service public class EmployeeService {   @Autowired  private EmployeeMapper employeeMapper;    @Autowired  private DepartmentMapper departmentMapper;    @Autowired  private DutyMapper dutyMapper;    @Autowired  private ChangeLogService changeLogService;   @Transactional  public PageInfo<Employee> updateDepartmentAndDuty(  Long empNumber,   Long newDeptId,   Long newDutyId,  int pageNum,   int pageSize) {    // 验证部门存在  Department dept = departmentMapper.selectById(newDeptId);  if (dept == null) {  throw new BusinessException("部门不存在");  }    // 验证职务存在且属于该部门  Duty duty = dutyMapper.selectById(newDutyId);  if (duty == null || !duty.getDeptId().equals(newDeptId)) {  throw new BusinessException("职务不合法");  }    // 获取原员工信息  Employee employee = employeeMapper.selectByNumber(empNumber);  if (employee == null) {  throw new BusinessException("员工不存在");  }    // 记录变更日志  changeLogService.recordChange(  employee.getNumber(),  "部门职务变更",  String.format("部门: %s->%s, 职务: %s->%s",   employee.getDeptName(), dept.getDeptName(),  employee.getDutyName(), duty.getDutyName())  );    // 更新员工信息  employee.setDeptId(newDeptId);  employee.setDutyId(newDutyId);  employeeMapper.update(employee);    // 返回分页数据  return getEmployees(pageNum, pageSize);  } } |

**3.2 控制器实现**

|  |
| --- |
| Java @RestController @RequestMapping("/emp") public class EmployeeController {   @Autowired  private EmployeeService employeeService;   @PostMapping("/updateDD")  public ResponseEntity<PageResult<Employee>> updateDepartmentAndDuty(  @RequestParam Long number,  @RequestParam Long dept\_id,  @RequestParam Long duty\_id,  @RequestParam int currentPage,  @RequestParam int pageSize) {    PageInfo<Employee> pageInfo = employeeService.updateDepartmentAndDuty(  number, dept\_id, duty\_id, currentPage, pageSize);    return ResponseEntity.ok(new PageResult<>(  pageInfo.getList(),  pageInfo.getPageNum(),  pageInfo.getTotal()  ));  } } |

**4. 数据库设计**

**4.1 变更记录表**

|  |
| --- |
| SQL CREATE TABLE `employee\_change\_log` (  `id` bigint NOT NULL AUTO\_INCREMENT,  `emp\_number` bigint NOT NULL COMMENT '员工工号',  `change\_type` varchar(50) NOT NULL COMMENT '变更类型',  `change\_detail` varchar(500) NOT NULL COMMENT '变更详情',  `operator` varchar(50) NOT NULL COMMENT '操作人',  `create\_time` datetime DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  PRIMARY KEY (`id`),  KEY `idx\_emp` (`emp\_number`) ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='员工变更记录表'; |

**5. 错误码说明**

|  |  |  |
| --- | --- | --- |
| 错误码 | 说明 | 解决方案 |
| 4013 | 部门不存在 | 检查部门ID是否正确 |
| 4014 | 职务不合法 | 检查职务是否属于指定部门 |
| 4015 | 无权限修改该员工 | 联系管理员 |
| 4016 | 员工不存在 | 检查员工工号 |

**6. 前端调用示例**

**6.1 获取部门职务数据**

|  |
| --- |
| JavaScript axios.post("emp/getDeptData")  .then(resp => {  this.deptData = resp.data.data;  });  axios.post("emp/getDutyData")  .then(resp => {  this.dutyData = resp.data.data;  }); |

**6.2 提交人事变动**

|  |
| --- |
| JavaScript let dept\_id = this.deptData.find(d => d.dept\_name === this.editFormData.dept\_name).dept\_id; let duty\_id = this.dutyData.find(d => d.duty\_name === this.editFormData.duty\_name).duty\_id;  let param = `number=${this.editFormData.number}&dept\_id=${dept\_id}&duty\_id=${duty\_id}&currentPage=${this.pagination.currentPage}&pageSize=${this.pagination.pageSize}`;  axios.post("emp/updateDD", param)  .then(resp => {  this.$message.success('变动成功');  this.tableData = resp.data.data;  this.pagination.currentPage = resp.data.pageNum;  this.pagination.total = resp.data.total;  }); |

**附录**

**A. 部门职务关联规则**

1. 每个职务必须属于一个部门
2. 部门可以有多个职务
3. 员工职务必须与部门匹配

**B. 变更记录示例**

|  |  |  |  |
| --- | --- | --- | --- |
| 员工工号 | 变更类型 | 变更详情 | 操作人 |
| 1001 | 部门职务变更 | 部门:技术部->产品部,职务:工程师->产品经理 | admin |