**管理员接口文档（员工CRUD）——StuList**

**企业员工考勤系统 - 员工管理接口文档**

**文档信息**

|  |  |
| --- | --- |
| 项目名称 | 企业员工考勤系统 - 员工管理模块 |
| 版本号 | v1.2.0 |
| 最后更新时间 | 2023-11-22 |
| 技术栈 | SpringBoot 2.3.7 + MyBatis + PageHelper |
| 协议 | HTTP/HTTPS |
| 认证方式 | Session-Based Authentication |

**1. 员工信息管理模块**

**1.1 分页查询员工列表**

**接口描述**：分页查询员工信息列表

**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 |

**请求示例**：

|  |
| --- |
| Plain Text POST /emp/selectByPage HTTP/1.1 Host: yourdomain.com Content-Type: application/x-www-form-urlencoded  currentPage=1&pageSize=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 |
| └─ birthday | String | 生日(yyyy-MM-dd) |
| └─ address | String | 住址 |
| └─ 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",  "birthday": "1990-01-01",  "address": "北京市海淀区",  "dept\_name": "技术部",  "duty\_name": "高级工程师"  }  ],  "pageNum": 1,  "total": 25 } |

**1.2 添加员工信息**

**接口描述**：添加新员工信息

**请求URL**：/emp/addEmp

**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 | 员工姓名 | "张三" |
| birthday | String | 是 | 生日 | "1990-01-01" |
| address | String | 是 | 住址 | "北京市" |
| dept\_id | Long | 是 | 部门ID | 1 |
| duty\_id | Long | 是 | 职务ID | 1 |
| currentPage | Int | 是 | 当前页码 | 1 |
| pageSize | Int | 是 | 每页条数 | 8 |

**响应参数**：同1.1分页查询接口

**1.3 修改员工信息**

**接口描述**：修改员工基本信息

**请求URL**：/emp/updateEmp

**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 |
| name | String | 是 | 员工姓名 | "张三" |
| birthday | String | 是 | 生日 | "1990-01-01" |
| address | String | 是 | 住址 | "北京市" |
| currentPage | Int | 是 | 当前页码 | 1 |
| pageSize | Int | 是 | 每页条数 | 8 |

**响应参数**：同1.1分页查询接口

**1.4 删除员工信息**

**接口描述**：删除员工信息及相关考勤记录

**请求URL**：/emp/delete

**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 |
| currentPage | Int | 是 | 当前页码 | 1 |
| pageSize | Int | 是 | 每页条数 | 8 |

**响应参数**：同1.1分页查询接口

**2. 部门职务管理模块**

**2.1 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 |

**2.2 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 |

**3. 接口规范说明**

**3.1 请求头要求**

* 管理员权限接口需携带Session Cookie
* Content-Type: application/x-www-form-urlencoded

**3.2 分页参数说明**

* 页码从1开始
* 默认每页8条数据
* 支持5/8/12条每页的切换

**3.3 数据验证规则**

|  |  |
| --- | --- |
| 字段 | 规则 |
| name | 必填，2-10个字符 |
| birthday | 必填，有效日期 |
| address | 必填，2-200个字符 |
| dept\_id | 必填，有效部门ID |
| duty\_id | 必填，有效职务ID |

**4. 后端实现示例**

**4.1 员工服务层实现**

|  |
| --- |
| Java @Service public class EmployeeService {   @Autowired  private EmployeeMapper employeeMapper;    @Autowired  private DepartmentMapper departmentMapper;    @Autowired  private DutyMapper dutyMapper;   public PageInfo<Employee> getEmployees(int pageNum, int pageSize) {  PageHelper.startPage(pageNum, pageSize);  List<Employee> employees = employeeMapper.selectAllWithDeptAndDuty();  return new PageInfo<>(employees);  }    public int addEmployee(Employee employee) {  // 验证部门存在  Department dept = departmentMapper.selectById(employee.getDeptId());  if (dept == null) {  throw new BusinessException("部门不存在");  }    // 验证职务存在  Duty duty = dutyMapper.selectById(employee.getDutyId());  if (duty == null) {  throw new BusinessException("职务不存在");  }    return employeeMapper.insert(employee);  } } |

**4.2 员工控制器实现**

|  |
| --- |
| Java @RestController @RequestMapping("/emp") public class EmployeeController {   @Autowired  private EmployeeService employeeService;   @PostMapping("/selectByPage")  public ResponseEntity<PageResult<Employee>> getEmployees(  @RequestParam int currentPage,  @RequestParam int pageSize) {    PageInfo<Employee> pageInfo = employeeService.getEmployees(currentPage, pageSize);  return ResponseEntity.ok(new PageResult<>(  pageInfo.getList(),  pageInfo.getPageNum(),  pageInfo.getTotal()  ));  }    @PostMapping("/addEmp")  public ResponseEntity<?> addEmployee(  @Valid Employee employee,  @RequestParam int currentPage,  @RequestParam int pageSize) {    employeeService.addEmployee(employee);  return getEmployees(currentPage, pageSize);  } } |

**5. 数据库设计建议**

**5.1 员工表结构**

|  |
| --- |
| SQL CREATE TABLE `employee` (  `number` bigint NOT NULL AUTO\_INCREMENT COMMENT '员工工号',  `name` varchar(20) NOT NULL COMMENT '员工姓名',  `birthday` date NOT NULL COMMENT '出生日期',  `address` varchar(200) NOT NULL COMMENT '住址',  `dept\_id` bigint NOT NULL COMMENT '部门ID',  `duty\_id` bigint NOT NULL COMMENT '职务ID',  `create\_time` datetime DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  `update\_time` datetime DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  PRIMARY KEY (`number`),  KEY `idx\_dept` (`dept\_id`),  KEY `idx\_duty` (`duty\_id`) ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='员工信息表'; |

**5.2 部门表结构**

|  |
| --- |
| SQL CREATE TABLE `department` (  `dept\_id` bigint NOT NULL AUTO\_INCREMENT,  `dept\_name` varchar(50) NOT NULL,  PRIMARY KEY (`dept\_id`),  UNIQUE KEY `uk\_name` (`dept\_name`) ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='部门表'; |

**6. 错误码说明**

|  |  |  |
| --- | --- | --- |
| 错误码 | 说明 | 解决方案 |
| 4008 | 员工姓名已存在 | 修改员工姓名 |
| 4009 | 无效的部门ID | 检查部门是否存在 |
| 4010 | 无效的职务ID | 检查职务是否存在 |
| 4011 | 员工不存在 | 检查员工工号 |
| 4012 | 分页参数无效 | 检查currentPage/pageSize值 |

**附录**

**A. 前端表单验证规则**

|  |
| --- |
| JavaScript rules: {  name: [  {required: true, message: '请输入姓名', trigger: 'blur'}  ],  birthday: [  {required: true, message: '请选择日期', trigger: 'change'}  ],  address: [  {required: true, message: '请输入地址', trigger: 'blur'}  ],  dept\_id: [  {required: true, message: '请选择部门', trigger: 'change'}  ],  duty\_id: [  {required: true, message: '请选择职务', trigger: 'change'}  ] } |

**B. 分页响应封装类**

|  |
| --- |
| Java @Data public class PageResult<T> {  private List<T> data;  private int pageNum;  private long total;    public PageResult(List<T> data, int pageNum, long total) {  this.data = data;  this.pageNum = pageNum;  this.total = total;  } } |