

日志附件功能前端集成提示

后端API概述

1. 认证机制

- **认证方式:** JWT Token认证
- **请求头格式:** `Authorization: Bearer {token}`
- **Token获取:** 通过 `POST /auth/login` 接口获取, 登录时需要提供 `phone` 和 `password`
- **Token有效期:** 请关注响应中的过期时间, 过期后需重新登录或使用刷新令牌

2. 日志创建接口

- **接口路径:** `POST /logs`
- **请求方式:** `multipart/form-data`
- **必要参数:**
 - `content`: 日志内容 (必填, 字符串)
 - `files`: 附件文件数组 (可选, 支持多文件)
- **可选参数:**
 - `mood`: 心情表情代码 (整数)
 - `taskId`: 关联任务ID (整数)
 - `employeeLocation`: 员工位置信息 (字符串)
- **响应格式:** 创建成功返回201状态码和完整的日志信息, 包含附件数据

3. 日志查询接口

- **获取所有日志:** `GET /logs`
- **获取单条日志详情:** `GET /logs/{logId}` 或 `GET /logs/{logId}/detail`
- **按日期查询:** `GET /logs/byDate?datetime={date}`
- **按周查询:** `GET /logs/week?startDate={date}&endDate={date}`
- **按月查询:** `GET /logs/month?startDate={date}&endDate={date}`
- **搜索日志:** `GET /logs/search?keyword={keyword}`

4. 附件下载接口

- **接口路径:** `GET /logs/attachments/{attachmentId}`
- **响应:** 返回文件流, 包含正确的Content-Disposition和Content-Type头
- **注意事项:** 下载时需要保持认证状态

数据结构说明

日志数据结构(LogDTO)

```
{
  "logId": 1,           // 日志ID
  "employeeId": 123,    // 员工ID
  "employeeName": "张三", // 员工姓名
  "taskId": 456,        // 关联任务ID
  "taskName": "开发任务", // 任务名称
  "createdTime": "2024-01-01T10:00:00", // 创建时间
  "content": "今日工作内容", // 日志内容
  "emoji": "😊",        // 心情表情
  "employeeLocation": "办公室", // 员工位置
  "position": 3,        // 员工职位
  "attachments": [...]  // 附件列表
}
```

附件数据结构(AttachmentDTO)

```
{
  "id": 1001,           // 附件ID，用于下载
  "originalFilename": "报告.pdf", // 原始文件名
  "fileType": "application/pdf", // 文件类型MIME
  "fileSize": 1024000,  // 文件大小（字节）
  "uploadTime": "2024-01-01T10:00:00" // 上传时间
}
```

开发建议

文件上传处理

- 1. 使用 `multipart/form-data` 格式进行文件上传
- 2. 支持多文件选择和上传
- 3. 实现文件选择界面，显示已选文件列表
- 4. 添加文件类型和大小验证
- 5. 实现上传进度显示
- 6. 处理上传失败的情况，提供友好的错误提示

附件下载处理

1. 使用附件ID构建下载URL
2. 下载完成后保存到本地存储
3. 提供文件预览功能（针对图片、PDF等常见格式）
4. 显示下载进度和状态

列表展示

1. 在日志列表中显示附件数量标识
2. 点击日志项可查看详情，包括附件列表
3. 为不同类型的附件显示不同的图标

错误处理

1. 处理网络异常，提供重试机制
2. 处理权限不足的情况
3. 处理文件上传失败的情况
4. 提供用户友好的错误提示

性能优化

1. 实现图片懒加载
2. 考虑分页加载大量日志
3. 实现离线缓存功能
4. 优化大文件上传体验

安全考虑

1. 确保所有请求都携带有效的认证Token
2. 实现Token过期自动刷新机制
3. 验证用户权限，确保只能操作自己的日志

测试建议

1. 使用ApiFox或Postman测试API接口
2. 测试不同文件类型的上传和下载
3. 测试多文件同时上传
4. 测试大文件上传性能

5. 测试Token过期和权限验证