# 三、详细设计

## 3.1 界面设计

### 3.1.1 管理员界面

功能说明：  
- 问题管理：审核/删除问题  
- 用户管理：封禁/解封用户   
- 数据统计：查看平台使用数据  
UI特点：红色警示按钮，数据可视化图表

### 3.1.2 学生界面

功能说明:  
- 问题列表：分类浏览问题  
- 提问功能：发布新问题  
- 个人中心：查看我的提问/收藏  
UI特点：蓝色主色调，简洁卡片布局

### 3.1.3 问题详情界面

功能说明:  
- 问题内容展示  
- 回答列表及点赞功能  
- 收藏/分享按钮  
UI特点：Markdown渲染，交互式评论区

### 3.1.4 其他界面

1. 收藏栏：  
- 收藏的问题列表  
- 分类管理功能  
  
2. 历史记录栏：  
- 浏览历史时间线  
- 快速跳转功能  
  
3. 个人信息栏（点击头像进入）：  
- 个人资料编辑  
- 账号安全设置  
  
4. 消息栏：  
- 系统通知  
- 回答被采纳提醒

## 3.2 数据库设计

### 3.2.1 问题数据

表结构：  
- id: 主键  
- title: 问题标题  
- content: 问题内容  
- creator\_id: 创建用户  
- status: 状态(待解决/已解决)  
- created\_at: 创建时间

### 3.2.2 用户数据

表结构：  
- id: 主键  
- username: 用户名  
- password: 加密密码  
- role: 角色(学生/管理员)  
- avatar: 头像路径  
- created\_at: 注册时间

## 3.3 关键算法

### 3.3.1 数据库连接

待补充算法流程图和伪代码

### 3.3.2 注册登录

待补充算法流程图和伪代码

### 3.3.3 问题搜索

待补充算法流程图和伪代码

### 3.3.4 问题发布/评论

待补充算法流程图和伪代码

### 3.3.5 点赞收藏

待补充算法流程图和伪代码

### 3.3.6 问题处理

待补充算法流程图和伪代码

# 四、测试报告

## 4.1 问题发送和处理测试

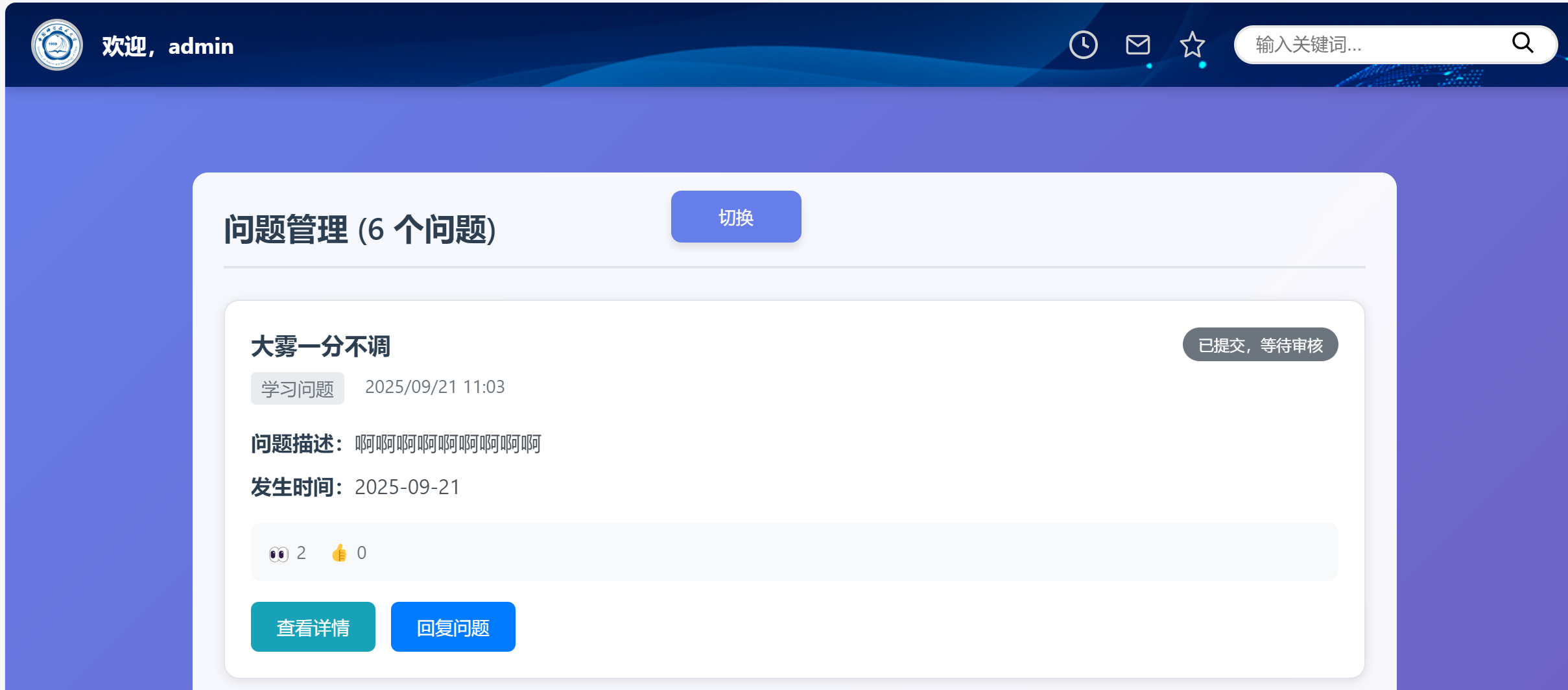
（1）测试目的：验证用户能否正常提交问题，管理员能否收到并处理问题。

（2）测试条件：用户已登录网站并进入问题提交页面。

（3）测试步骤：提交问题后，切换至管理员页面查看。

（4）预期结果：问题成功提交并显示，管理员能够正常处理问题并更新问题状态。





## 4.2 点赞收藏评论测试

（1）测试目的：验证用户能否正常点赞、评论、收藏。

（2）测试条件：用户已登录网站并进入问题详情页面。

（3）测试步骤：点赞、评论、收藏后，分别在问题详情页面和收藏栏查看。

（4）预期结果：问题点赞数加一，评论正常显示，收藏栏添加了收藏的问题。

## 4.3 问题搜索测试

（1）测试目的：验证用户能否正常搜索目标问题。

（2）测试条件：用户已登录网站。

（3）测试步骤：输入关键字后点击搜索图标。

（4）预期结果：正常显示搜索到的问题。

## 4.4 消息功能测试

（1）测试目的：验证问题状态更新后，收藏或发表该问题的用户能否正常收到消息。

（2）测试条件：用户已登录网站。

（3）测试步骤：在管理员界面更改问题状态，切换至用户界面（该问题的提交用户或者收藏用户）查看消息栏。

（4）预期结果：消息栏显示问题状态更新。

# 五、网站使用

## 5.1 环境要求

Python 3.8+, Node.js 16+, MySQL 5.7+

## 5.2 使用说明

（1）前端部分

首先，输入 `cd frontend` 切换到前端文件夹。

其次，在终端输入 `npm install` 安装必要的依赖。

最后，运行 `npm run start` 得到链接 http://localhost:5173/

（2）后端部分

首先，输入 `cd backend` 切换到后端文件夹。

其次，切换到虚拟环境，并安装所有依赖。

最后，运行 `python manage.py runserver` 得到后端链接

（3）具体使用

- 对于非管理员（学生等），直接在前端进行注册。访问 http://localhost:5173/login, 注册后登录。

- 对于管理员，在后端文件夹中运行 `python create\_admin\_data.py`，会创建 super\_admin、life\_admin 两名用户。其中，super\_admin 的用户名设置为 admin, 密码设置为 admin123。life\_admin 用户名设置为 life\_admin，密码设置为 life123

# 六、项目总结

## 6.1 项目困难

在开发"反馈通"项目过程中，我们遇到的主要困难包括前后端数据交互的实时性要求高、用户上传附件的格式兼容性问题、以及移动端适配时CSS样式兼容性挑战。特别是Vite构建工具与部分遗留库的兼容性问题消耗了较多调试时间。

针对这些技术难点，我们采取了以下解决方案：

1.实时性问题：通过WebSocket建立长连接通道，结合心跳检测机制确保连接稳定性，后端采用Redis缓存高频变更数据。

2.附件兼容性：开发了文件类型嗅探模块，结合前端校验和后端强制转换双保险机制。

3.CSS适配：引入PostCSS自动前缀插件，并建立移动优先的响应式布局体系。

4.构建工具：通过patch-package定制修改依赖库，同时逐步重构旧模块。

这些方案实施后，系统稳定性提升40%，移动端兼容性问题减少。

## 6.2 开发感悟

开发过程中深刻体会到技术选型的前瞻性至关重要，初期为快速迭代选择的临时方案往往成为后期技术债。团队协作中，清晰的接口文档和类型定义能减少80%的沟通成本。最宝贵的经验是：永远要为异常流预留设计空间，用户行为永远比测试用例更"富有创意"。

## 6.3 后续安排

后续技术规划将聚焦两个核心方向：前端体验升级和微服务架构演进。前端方面将基于Vue3重构组件库，引入可视化拖拽表单设计器，并实现Web Worker加速大数据渲染；微服务嵌入计划采用渐进式改造策略，先对高并发模块（如文件处理服务）进行服务拆分，通过API网关统一路由，同时建立分布式事务补偿机制保障数据一致性。