珞珈在线判题平台

**软件设计说明书**

**日期：2023年11月6日**

1. **引言**

**1.1编写目的**

该文档在概要设计的基础上，进一步的细化系统结构，展示了软件结构的图标，物理设计、数据结构设计、及算法设计、详细的介绍了系统各个模块是如何实现的，包括涉及到的算法，逻辑流程等。

**1.2读者对象**

程序员。

**1.3软件项目概述**

此项目面对所有计算机学生和应聘者开发，用于求职准备和算法学习。用户可以选择管理员上传的算法题目进行编写并提交，系统会运行用户提交的答案代码并评判正误。

题目：作为一种算法题目，要求用户依据题目要求编写程序代码，随后通过测试则算解题成功。

题目难度：初始版本计划通过管理员手动设置题目难度。在我们的进阶版本中计划通过用户答题统计自动动态配置题目难度，70%以上通过率为易，30%-50%为中，30%以下为难。

**1.4文档概述**

此文档从系统总体设计、内部接口设计、用户界面设计、用例设计、类设计、数据库设计六个角度阐明我们对此系统的设计和实现思路。

**1.5参考资料**

[1]窦万峰.软件工程方法与实践(第三版).北京：机械工业出版社，2016

[2]普莱斯曼.软件工程：实践者的研究方法(原书第8版).北京：机械工业出版社，2016

[3]克雷格·沃斯 Spring实战（第6版）人民邮电出版社

1. **总体设计**

**2.1需求规定**

系统主要输入为用户选定题目后所编写的解题代码，系统在运行代码后与用例进行比对后输出判题结果。

由于是在线测评系统，其处理能力主要考虑到系统能承载的最大用户数，按照实际情况的规划，系统至少能承载的最大并发用户数要求达到全校师生总人数\*w，w是介于0到1的常数，具体数值根据服务器容量而定。

新增测评事务在6点-22点期间保持在一秒以内响应，23点-5点期间保持在3秒以内。简单测评在高峰期做到5秒内响应，平时2秒内响应；复杂测评在高峰期15秒内响应，在平时8秒内响应。

**2.2运行环境**

前端：主流浏览器，包括edge、chrome、谷歌浏览器等

后端：linux服务器。

**2.3基本设计概念与处理流程**

此系统采用spring boot前后端分离开发，前端独立编写客户端代码，后端编写服务端代码，然后提供数据接口即可。

用户在客户端进行操作之后前端做出对应的处理，随即调用对应的后端接口返回需要的数据或JSON。特别的，在判题过程中调用特有的判题服务器运行用户所提交的代码，并返回运行结果。

**2.4文档概述**

本文档是软件需求规格说明书，有五大部分：

①引言：简单描述本文档；

②任务概述：简单描述了此软件的目标，受众以及系统大致架构；

③开发与运行环境：描述了此软件的硬件环境，软件环境以及接口；

④软件功能需求：描述了此软件的各个功能模块；

⑤其他软件需求：描述了软件的除功能以外的其他需求，如性能需求，数据需求，故障处理需求等。

1. **接口设计**

**3.1外部接口**



**3.2内部接口**



1. **用户界面设计**

**4.1登陆界面：用户注册登录身份信息**

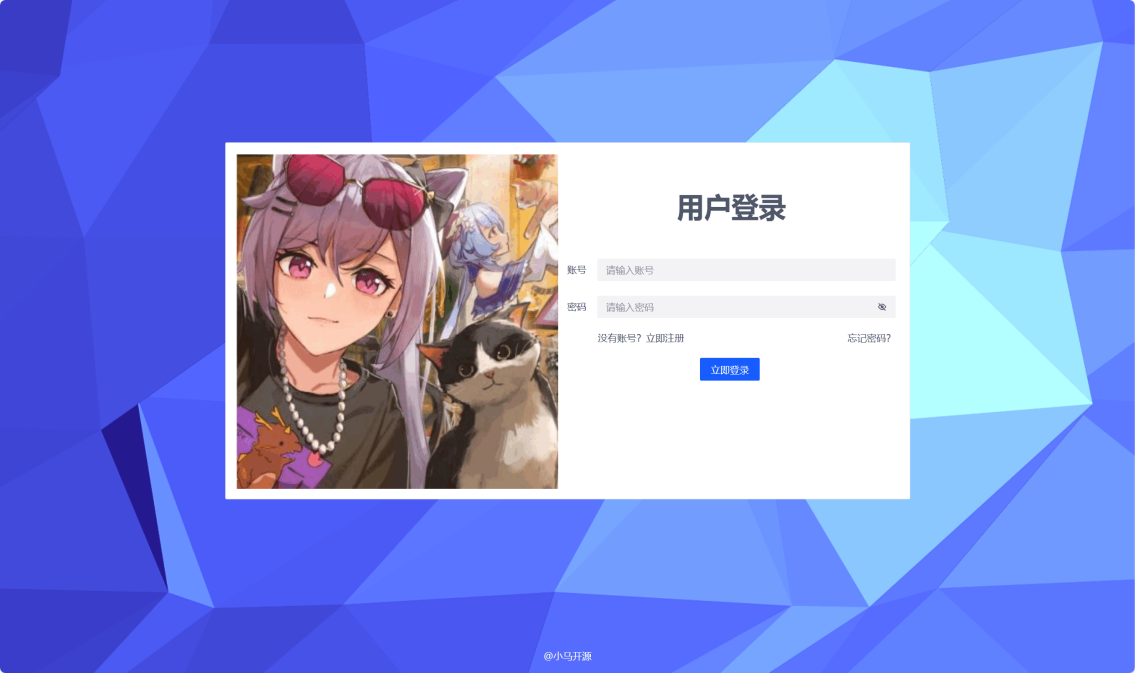


图4.1登陆界面

**4.2题目列表界面：用户可在此界面浏览题目列表，并点击题目进入题目详情页。**

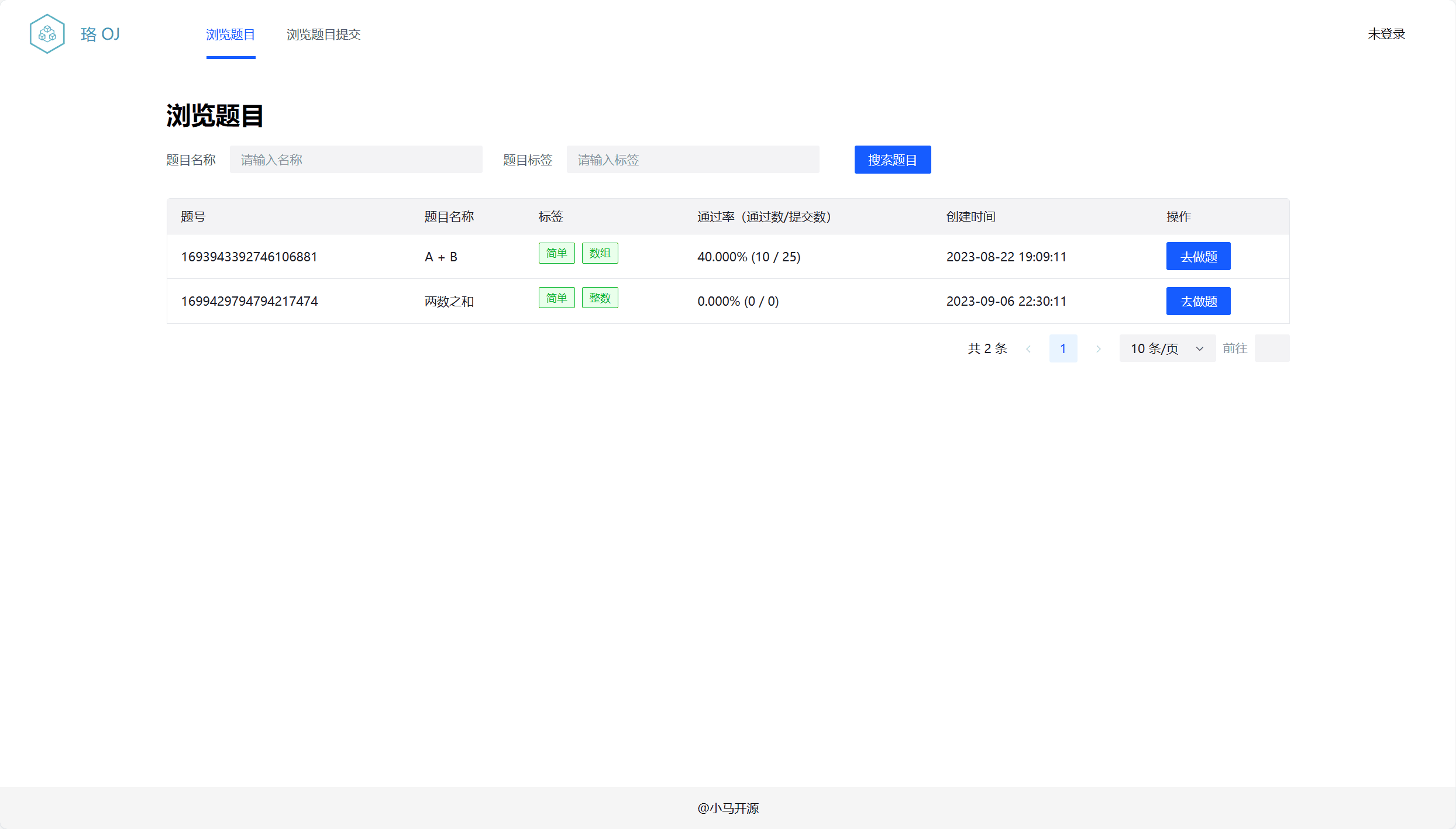
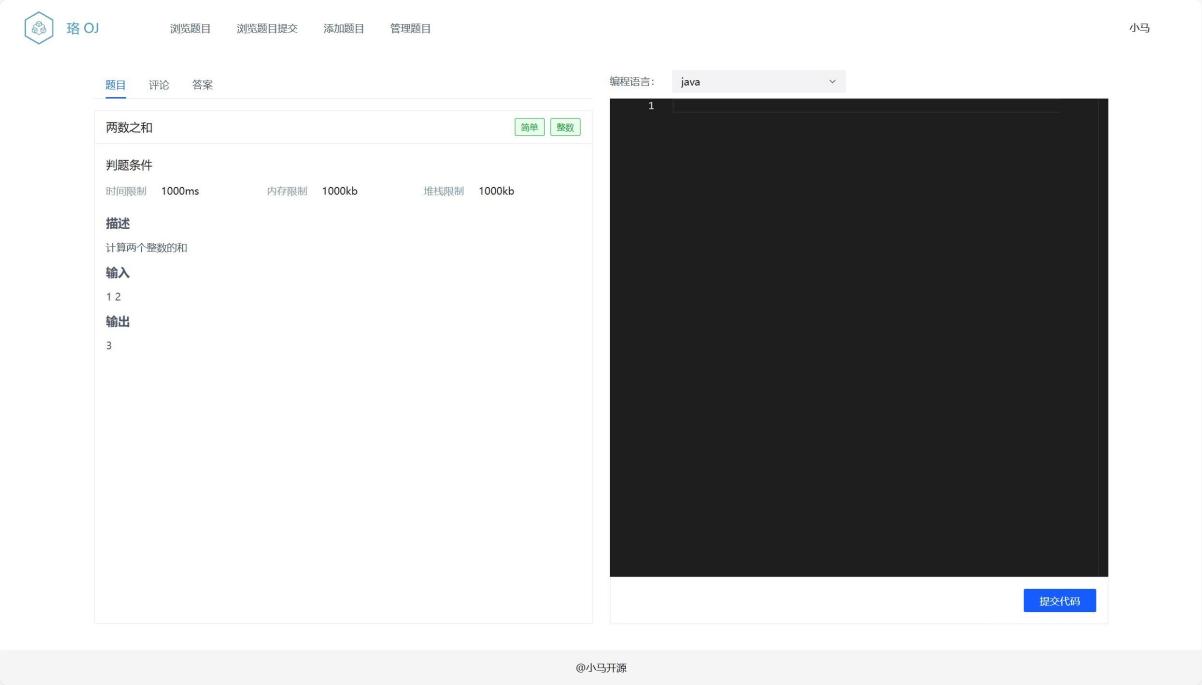


图4.2题目列表界面

**4.3题目详情界面：用户可在此界面查看题目详情，并完成代码编写与提交。**

图4.3题目详情界面

**4.4判题结果界面：在此界面，用户和管理员均可看到所有人在此题的提交测评结果。**

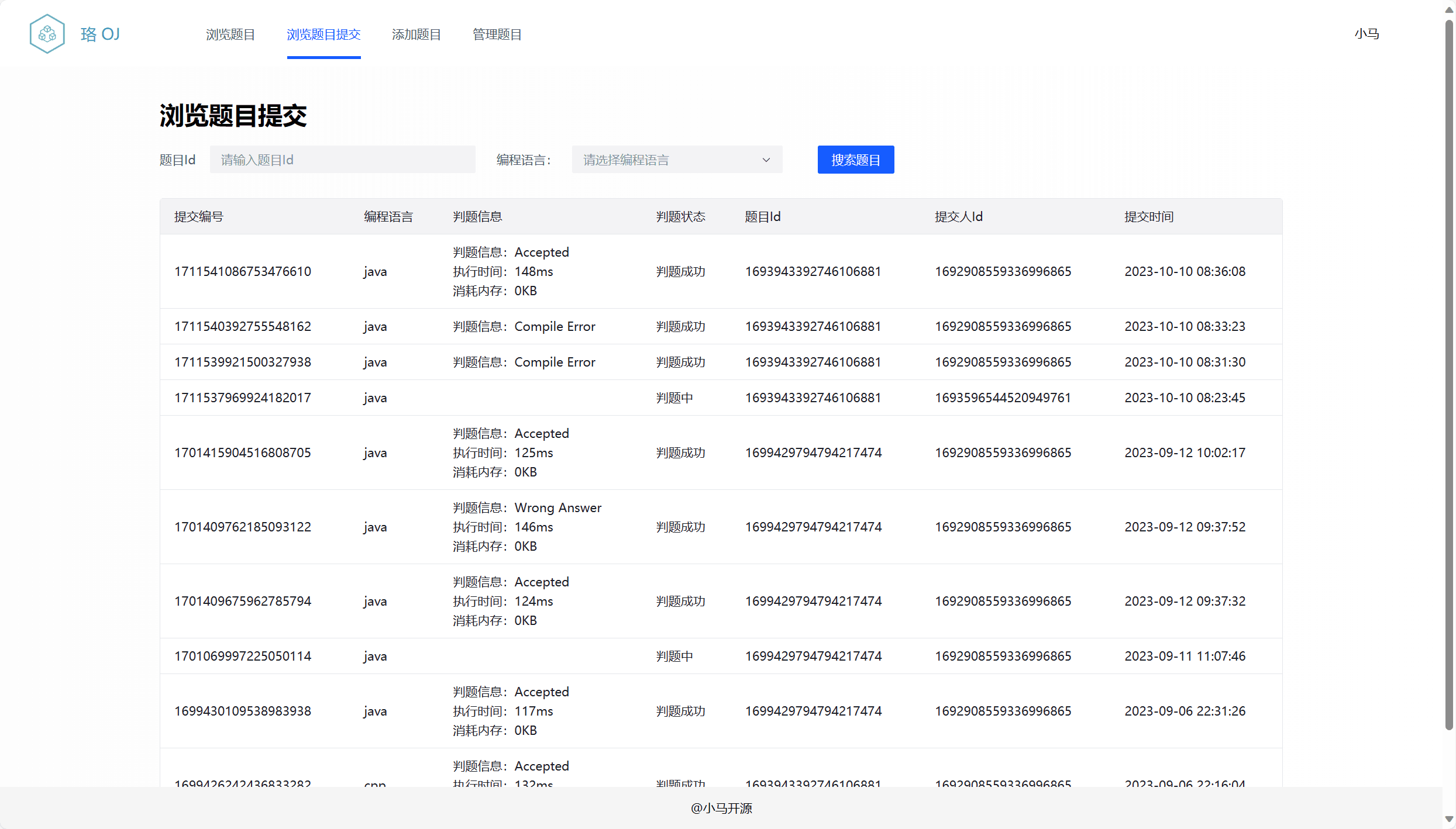


图4.4判题结果界面

**4.5创建题目界面：管理员在此界面可以增加新的题目，编辑内容包括题目名称、题目标签、题目内容和题目答案。**

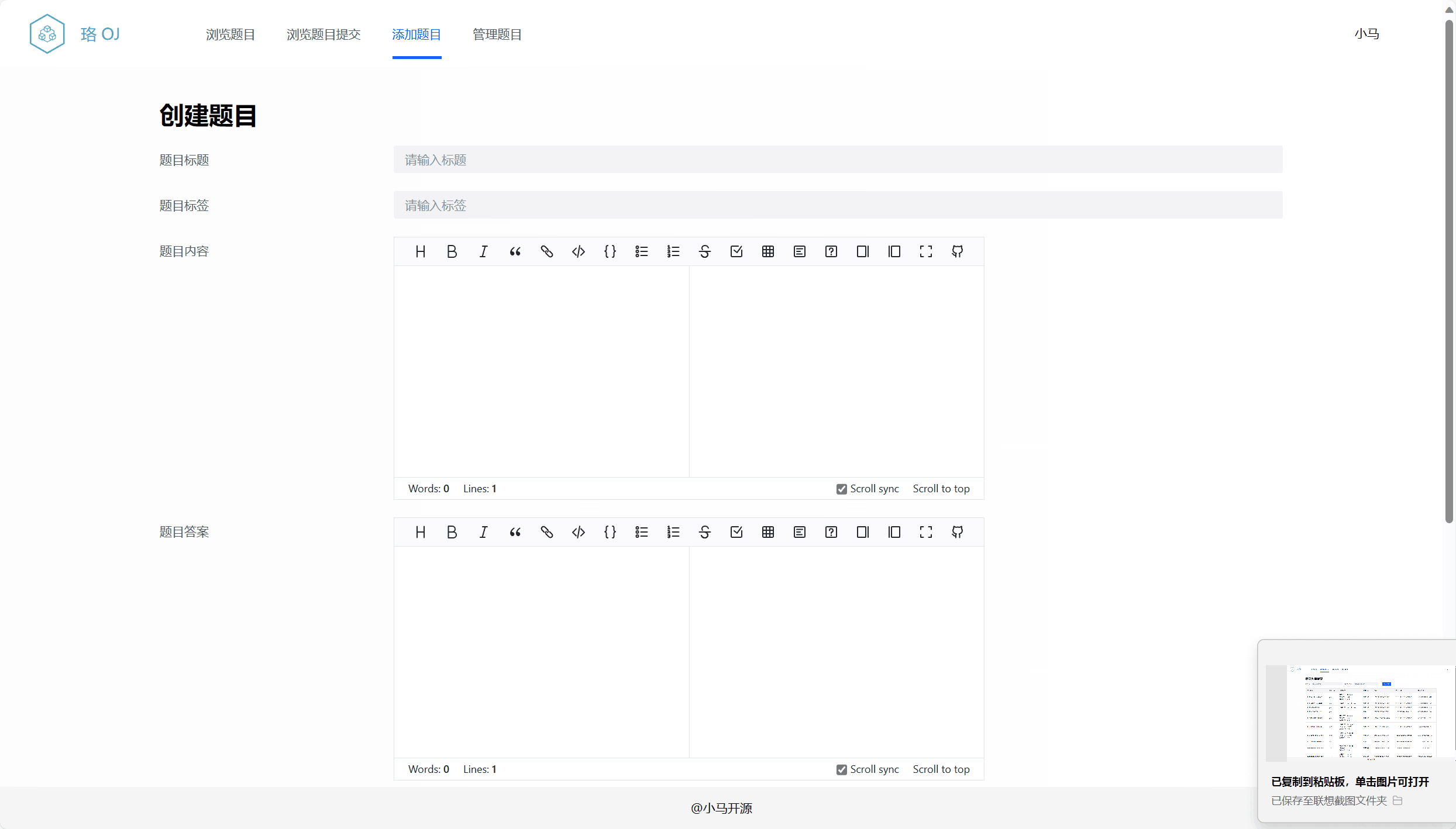


图4.5创建题目界面

**4.6管理题目界面：管理员在此界面可以管理已增加的题目，比如修改题目或者删除题目。**

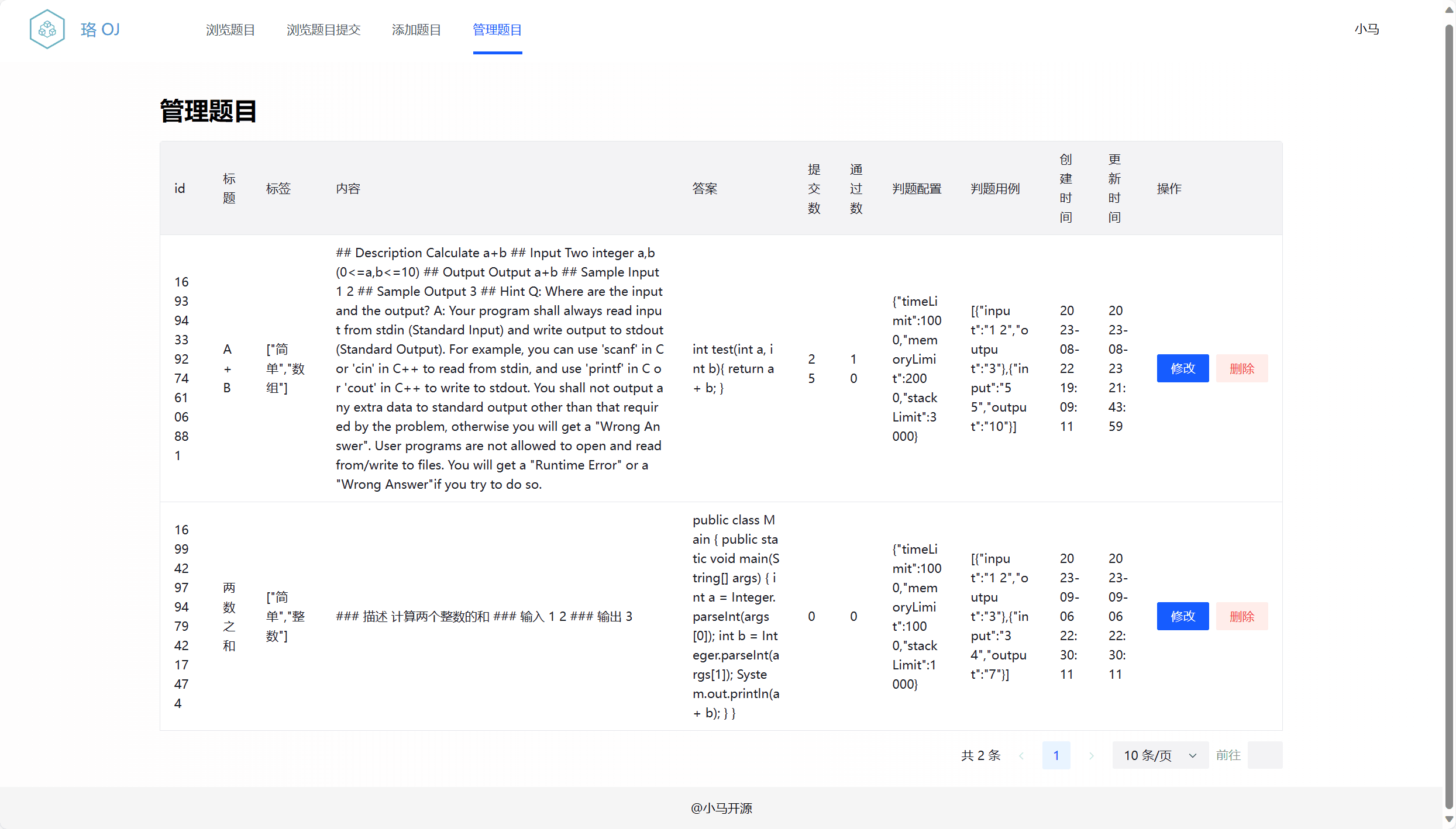


图4.6管理题目界面

1. **用例设计**

**5.1“用户登录”用例的实现方案**

用户登录主要是通过校验数据并校对用户名和密码实现。首先通过userLogin类进行校验，这个类有三个参数，分别是用户账号userAccount、用户密码userPassword和数据库存储的用户密码也即校准密码checkPassword。第一步校验参数是否为空，用户账号是否过短（用户账号字符数量是否小于4），用户密码是否过短（字符数量是否小于8），用户输入密码是否和校验密码相同。第二步是对数据进行加密处理，以防止密码受到盗窃。第三步是插入数据到数据库中。

登录成功后系统需要判断用户是普通用户还是管理员，因此需要加一个判断类isAdmin，判断输入的账号与密码是否和数据库中的管理员账号和密码相同。

具体流程是用户首先发起登录请求，由控制类userController控制登录信息，前端LoginView类展现登陆界面后，用户便输入账号密码，然后由userController对接Service层进行校验，service层连接数据库获取用户信息并进行记录，同时将登录信息记录到日志log中。在此过程中，userService类会在校验密码后判断账号密码是否是管理员，然后由userController类给予不同权限。

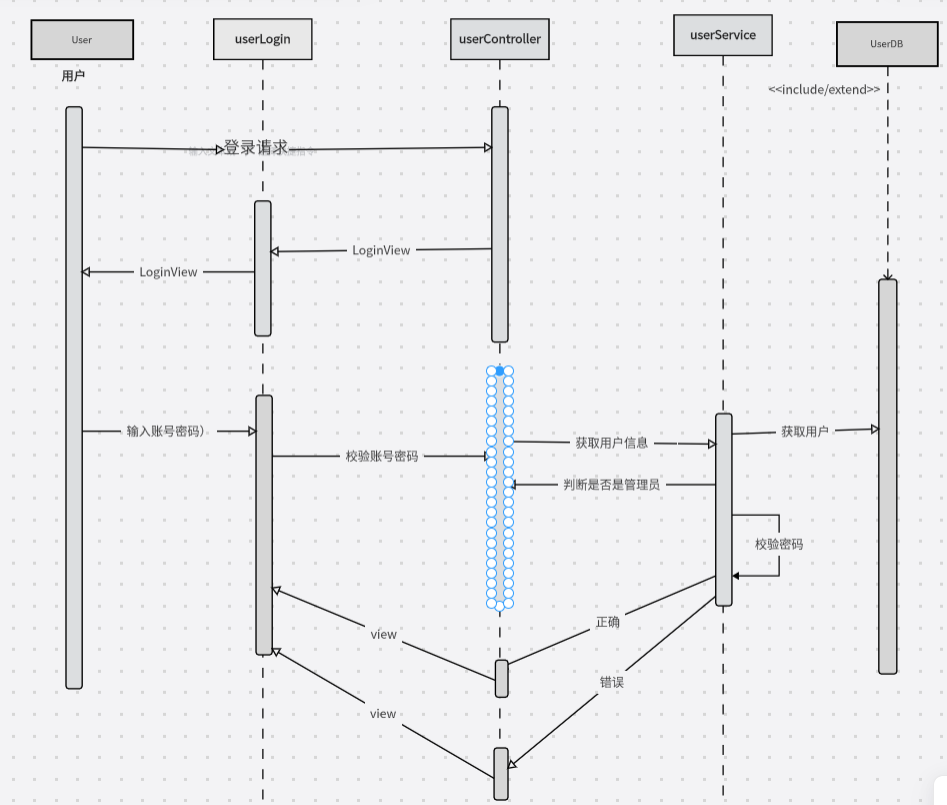


表5.1用户登录时序图

**5.2“用户注册”用例的实现方案**

在登陆界面提供注册功能，用户提交用户名和密码后由控制层userController类返回注册函数，将用户提交的账号和密码交给userService类进行校验，如果出现密码字符串为空、密码字符数量小于8、账户字符数量小于4或账户重复情况，则返回Business Exception进行报错处理。若注册成功，则由controller层控制返回相应的提示界面。同时，将新注册的用户添加到数据库userDB中。

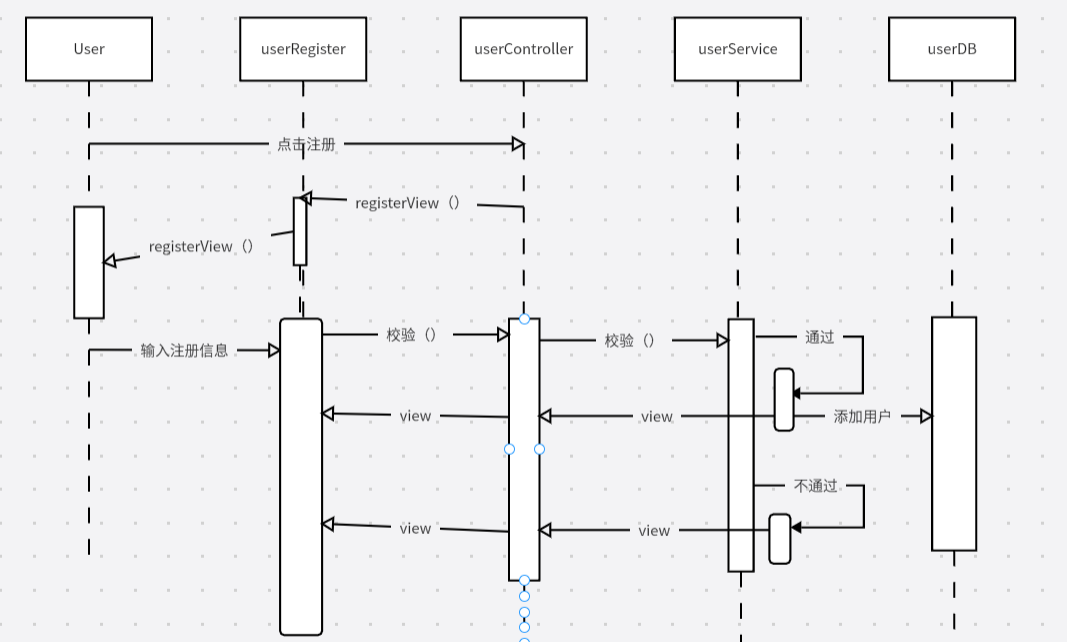


表5.2用户注册时序图

**5.3“浏览题目”用例的实现方案**

用户或者管理员进行题目列表进行浏览题目，如果想查看具体题目详情，则由controller层questionController类发出获取题目请求，由questionService类和题目数据库questionDB对接，在数据库中查询具体题目，并返回题目信息，再用questionController类提供题目详情页。

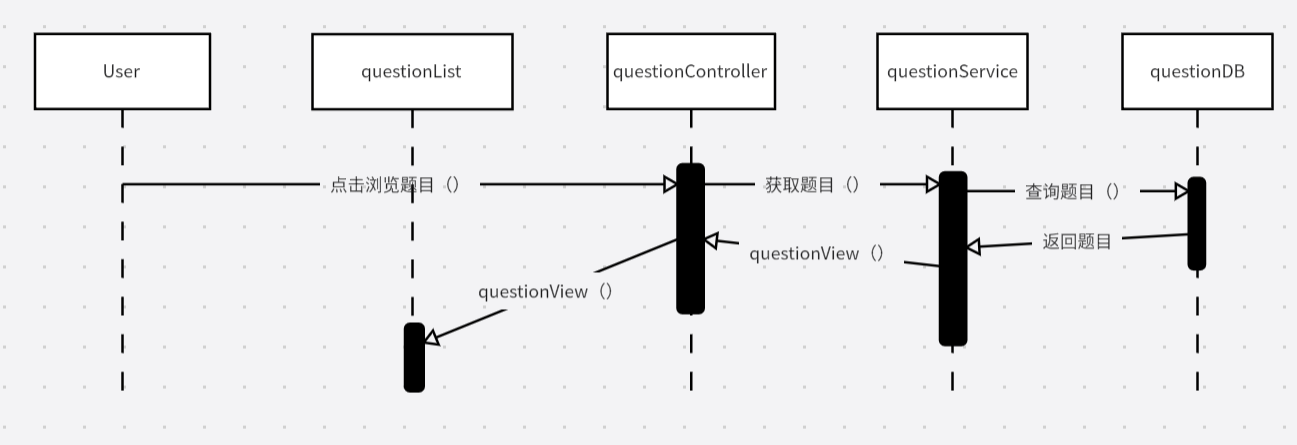


表5.3用户浏览题目时序图

**5.4“提交题目”用例的实现方案**

用户提交题目后，由controller层questionController类进行题目提交信息封装，具体包括提交用户UserID、题目序号QuestionID、编译语言Language。然后服务层收到来自控制层的消息后，记录下对应用户提交题目情况，并传送给题目数据库questionDB进行记录，然后再由控制层控制显示提交页面。

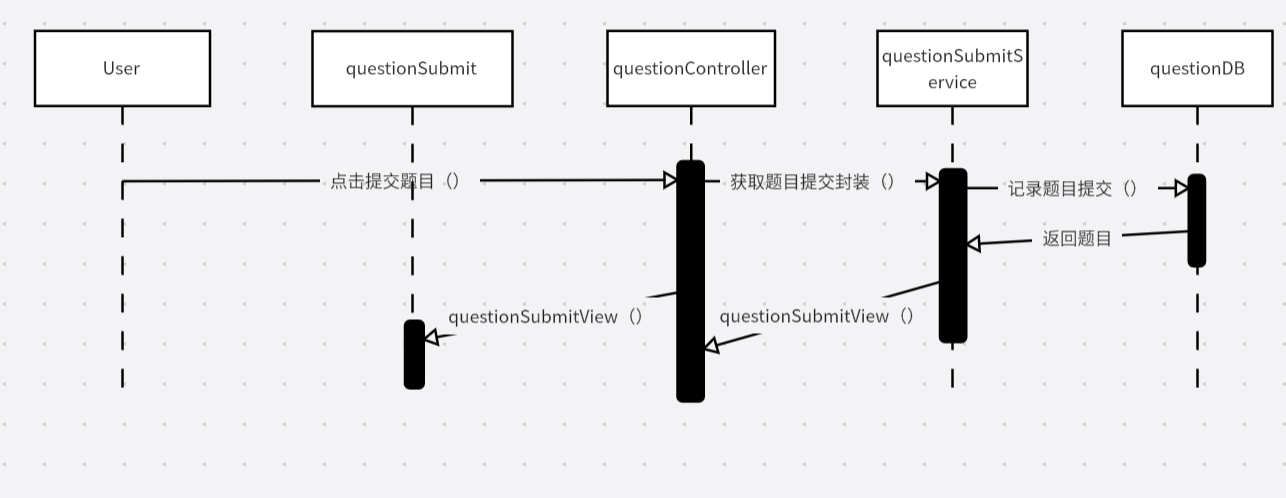


表5.4用户提交题目时序图

**5.5“管理题目”用例的实现方案**

管理题目这一部分是管理员的权限，首先管理员进行管理题目界面进行对题目的管理操作，questionManage类收到用户的指令后，将相应的操作传达给控制层questionController进行操作，具体包括增加、删除和更新题目操作。增加题目时，控制层交付给服务层questionService进行增加题目，在删除和更新题目时，服务层类还需判断题目是否存在。管理题目操作后，服务层将相应的修改记录在题目数据库中。

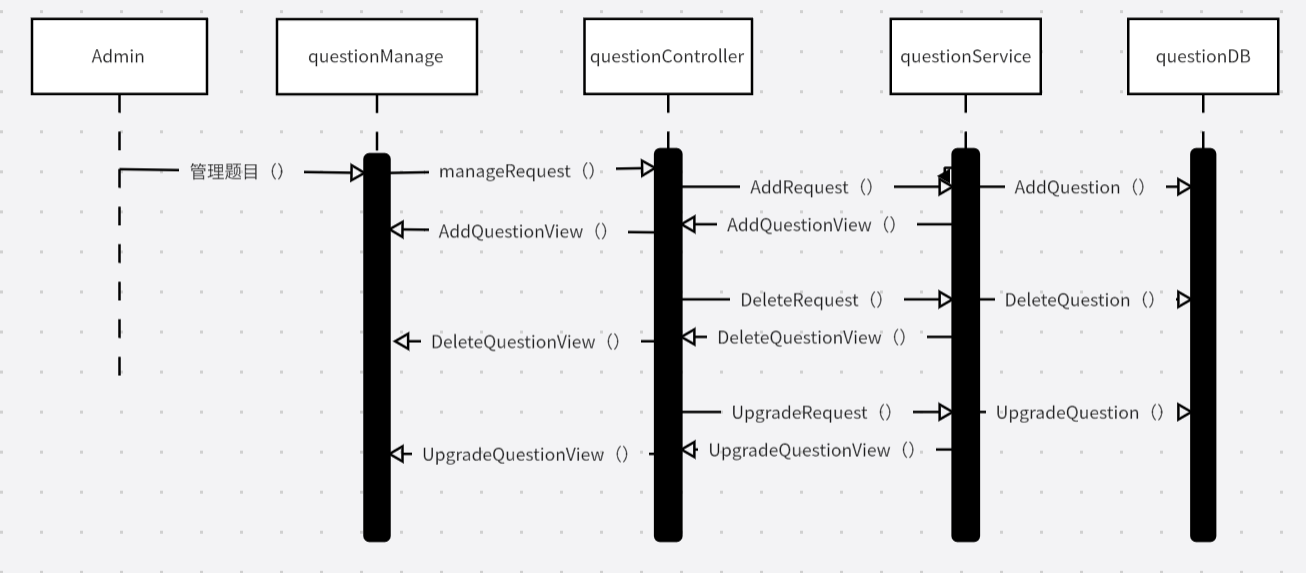


表5.5管理题目时序图

1. **类设计**

**6.1用户模块类**

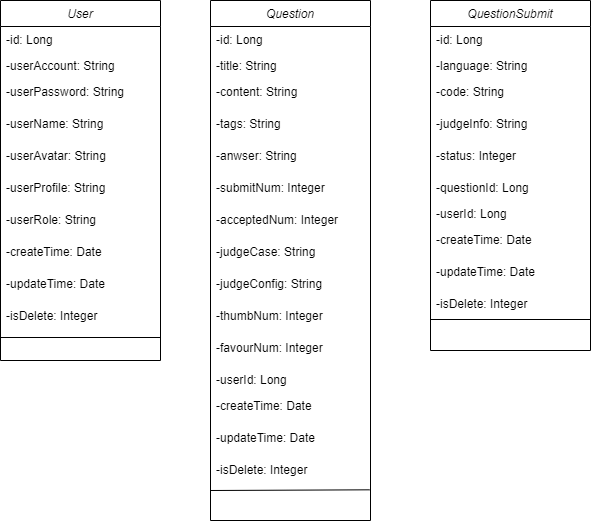


图6.1用户模块类图

用户类中包含用户id、账号、密码、用户名、头像、简介、角色（用户或管理员）、创建日期、更新日期信息和是否注销状态码。

**6.2题目模块类**

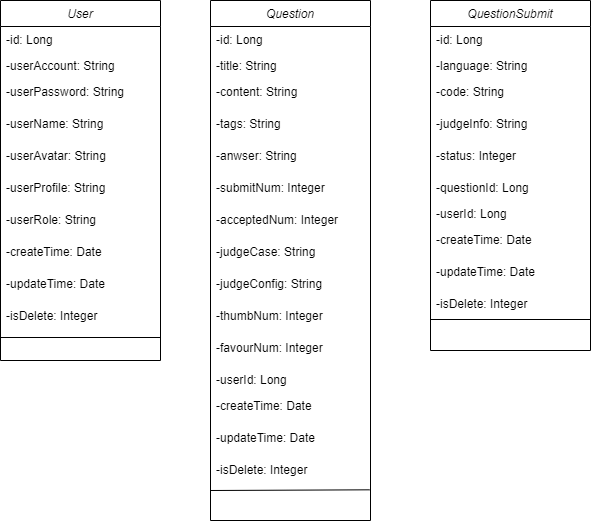


图6.2题目模块类图

题目类包含：题目id、题目标题、题目内容、题目标签、题目答案、题目提交次数、通过次数、测试用例、测试记录、获赞数、收藏数、创建用户id、创建时间、更新时间、是否删除状态码。

题目提交类包含：判题实例id、答题语言、答题代码、判题信息、判题状态、题目id、答题者id、创建时间、更新时间、是否删除状态码。

**6.3判题模块类**

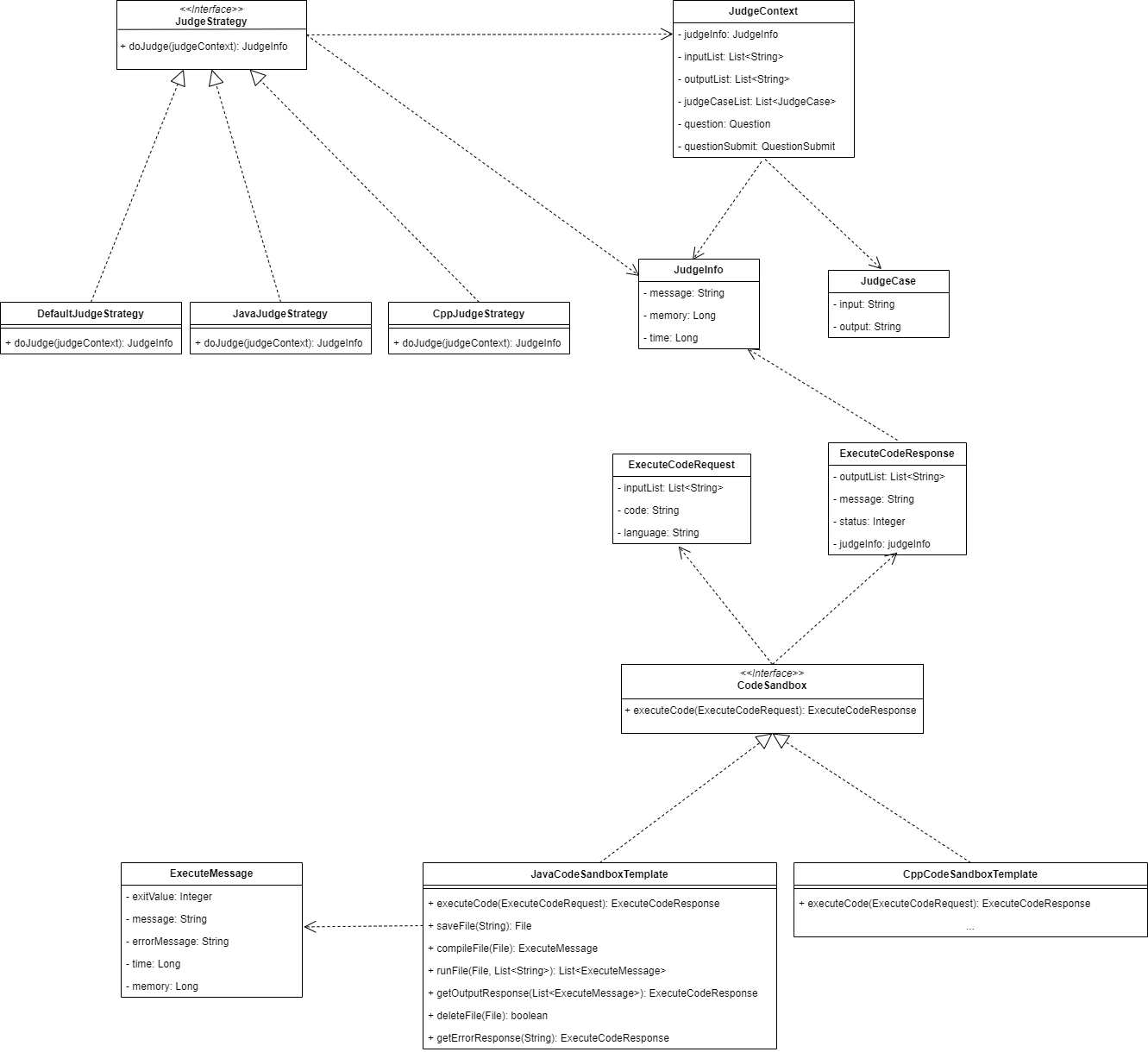


图6.3判题模块类图

（1）使用策略模式为不同的编程语言定制不同的判题策略，若要扩充编程语言只需实现顶层的接口JudgeStrategy，实现其doJudge()方法，定制其判题逻辑，并实现其代码沙箱。

（2）将判题前后所需要的参数封装成两个类：JudgeContext和JudgeInfo，JudgeContext封装了判题所需的上下文信息，如输入输出用例，题目信息等，JudgeInfo封装了判题后的结果信息，如判题信息、运行时间和占用空间。

（3）ExecuteCodeRequest与ExecuteCodeResponse分别封装好了判题请求信息和相应信息，用于判题服务与代码沙箱之间的数据交互。

（4）使用模板方法模式定制一套代码沙箱运行代码的模板方法，不同的编程语言根据此套模板定制其不同的代码执行逻辑，如JavaCodeSandboxTemplate与CppCodeSandboxTemplate等，返回统一的执行信息。

1. **数据库表设计**

用户表（user）：存储用户的账号信息与个人信息。

题目表（question）：存储题目信息。

题目提交表（question\_submit）：存储题目提交的信息。

