



**中国地质大学**

CHINA UNIVERSITY OF GEOSCIENCES

**《软件工程综合实习》  
实习报告**

题目： TeelCode 在线评测系统

班级序号： 111182

学生姓名： 朱逸宸、范泽奇、黄苏珍

任课教师： 杨之江、尚建嘎、杨林、李圣文、赵一石

**地理与信息工程学院软件工程系**

**2021 年 5 月**

## 目录

一、实习概况 .....	3
1.实习描述 .....	3
2.项目目标 .....	3
二、系统设计 .....	4
1. 系统框架设计 .....	4
2. 具体技术选择 .....	4
3. OJ 服务器设计 .....	5
4. Web 服务器设计 .....	5
5. 服务间通讯接口设计 .....	6
6. 数据存储设计 .....	6
三、分工实施 .....	7
1. 迭代设计 .....	7
1.1 迭代 0 .....	7
1.2 迭代 1 .....	7
1.3 迭代 2 .....	8
1.4 迭代 3 .....	8
2. 小组分工 .....	8
3. 实施与管理 .....	8
3.1 项目进度管理 .....	8
3.2 代码管理 .....	10
4. 软件测试 .....	11
5. 开发部署环境 .....	11
5.1 开发环境 .....	11
5.2 部署环境 .....	11
四、成果展示 .....	12
五、总结展望 .....	15
1.总结 .....	15
2.展望 .....	16
附录 .....	16

# 一、实习概况

## 1. 实习描述

**实验时间：**2021 年 4 月 1 日-5 月 6 日（4 周）

**实验地点：**信息楼 201（软工专业实验室机房）

**实验目的：**实习过程中学生需以团队形式完成一个完整的软件工程项目案例，一方面加深学生对软件工程实践知识域（软件需求、软件设计、软件构造、软件测试、软件维护、软件配置管理、软件工程管理、软件工程过程、软件工程模型和方法、软件质量）的理解，培养学生综合运用软件工程理论、技术和工具来开发高质量软件系统的能力，积累软件开发经验。还可以进一步帮助学生在实践中体验软件开发的实际场景和问题，发现软件开发的核心环节及面临的各种挑战，培养学生解决复杂工程问题的能力，锻炼学生在团队中沟通、交流的能力，以养成良好的软件工程素养。

## 2. 项目目标



TeelCode 在线 OJ 平台是一个在线评测系统，用于进行算法和编程的练习。

OJ 系统能够编译并执行代码，使用预设的数据对这些程序进行测试。提交的代码一般会在受限的环境下运行，包括时间限制、内存限制、安全限制等。代码的输出会被 OJ 系统捕获，与标准答案进行比较后返回结果。

OJ 系统还支持用户模块，通过账号可以对题目进行讨论和回复其他人的讨论；通过账号系统还可以查看自己的提交记录。OJ 系统还提供了在线管理模块，通过管理模块可以管理对分类和题目进行管理。

## 二、系统设计

### 1. 系统框架设计

TeelCode 服务器分为三层，分别是用于组织数据的数据层、用于处理判题和账户数据的逻辑层以及用于部署网页和 IDE 核心的表现层。分离了逻辑服务器和页面服务器主要是为了保证逻辑服务器的扩展性。具体设计如下图 1 所示：

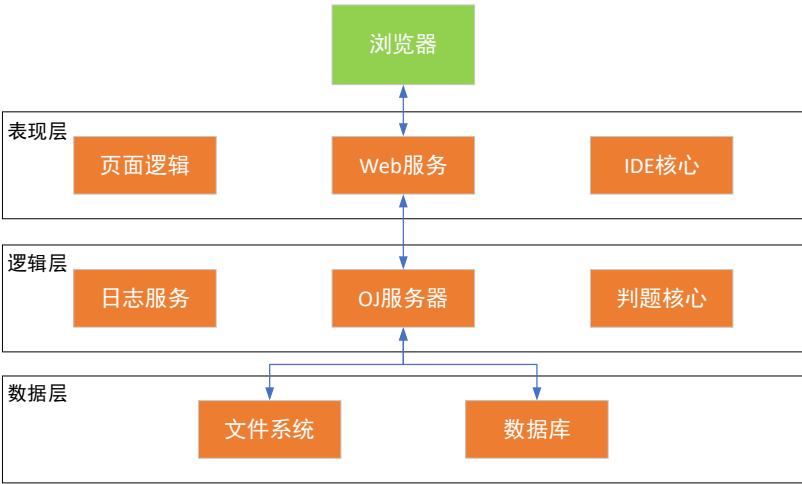


图 1 系统架构图

### 2. 具体技术选择

根据性能、兼容性和实现难易程度等因素；其中比较重要的是兼容性和性能，性能方面我们通过自研的方式开发可控且适合小型服务器的核心，同时开发的时候留足扩展性方便后续扩展；同时选择 CodeMirror 用于代码编辑的 IDE，选择 Markdown 作为题目描述的记录语言。选择如下表 1 所示：

表 1 技术选择

	项目	选择	备注
用户	浏览器核心	Chorme 内核	速度快、文档、兼容性
Web 服务器	框架	Node. js	
	Web 服务核心	Express	
	IDE 框架	CodeMirror+Markdown	1. 语法提示: 语言提示并支持自定义 2. 语法高亮: 语言高亮并支持自定义 3. 语言兼容: 支持主流语言
OJ 服务器	框架	Java+Tomcat	
	判题核心	自研	1. 体积可控: 部署要求底 2. 安全保证: 禁用系统调用、非法库 3. 效率可控: C++编写通过 JNA 调用
	数据库	mysql	
	日志	java.util.logging	

### 3. OJ 服务器设计

OJ 服务器用于判题和管理账户数据，设计分为三层。第一层(图中蓝色部分)用于访问数据，第二层(图中其他部分)用于组织业务逻辑，第三层(图中红色部分)通过 HTTP 协议接口。具体设计，如下图 2 所示：

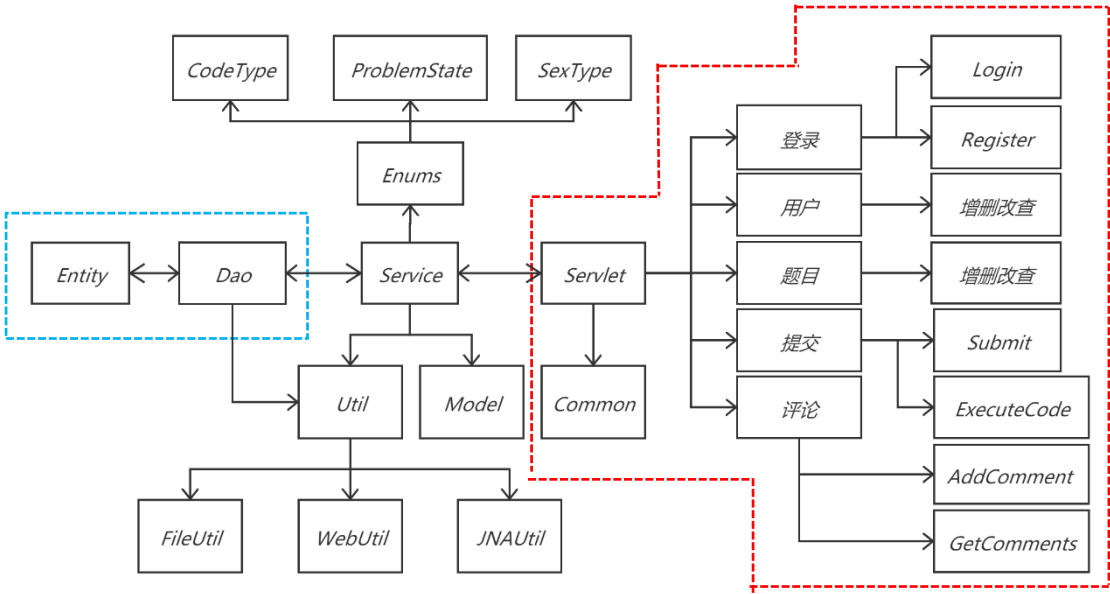


图 2 OJ 服务器设计

### 4. Web 服务器设计

Web 服务器主要用于完成页面的组织以及 CodeMirror 和 Markdown 的部署，同时负责与 OJ 服务器的交互。设计分为两部分，红色部分用于处理服务器请求与动态页面生成，蓝色部分为页面组织和插件实现。如下图 3 所示：

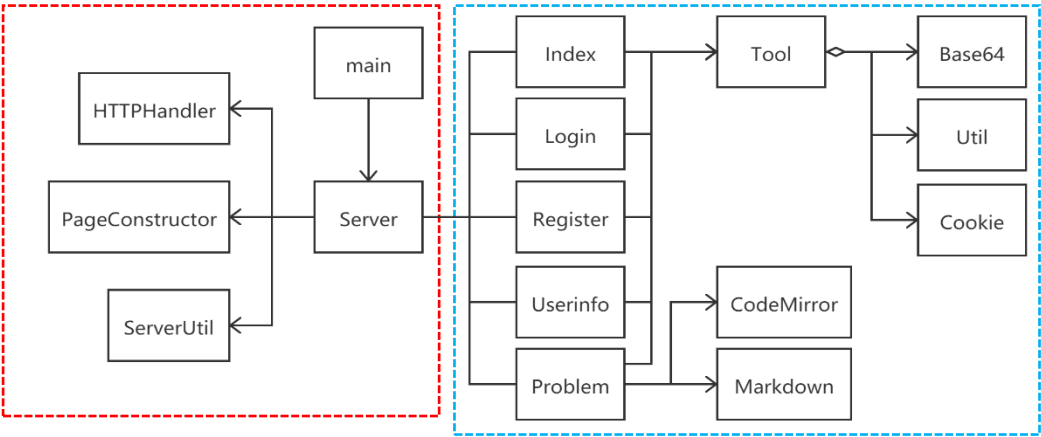


图 3 Web 服务器设计

## 5. 服务间通讯接口设计

OJ 服务器与 Web 服务器之间通过 HTTP 协议通讯；Web 服务器从浏览器接收请求，转而请求 OJ 服务器处理数据，之后 Web 服务器接收到返回然后动态生成页面后返回给浏览器。其间的接口如下图 4 所示：

接口		输入	输出
登录	Login	{id,pwd}	{result,token}
注册	Register	{id,pwd,problem,answer}	{result}
由Token获取ID	GetUid	{token}	{uid}
由UID获取用户信息	GetUserInfo	{uid}	{uid,token,name,sex,dscp,question,answer,img}
创建用户	NewUser	{token,name,sex,dscp}	{result}
更新用户信息	UpdateUser	{token,name,sex,dscp}	{result}
修改密码	AlterPassword	{token, answer, newpwd}	{result}
获取做题记录	GetUserRecord	{token, uid, page, offset}	{record} record为列表
获取题目列表	All	{token,page,offset,class,diff,status}	{result} result为列表
添加题目	AddProblem	{token,id,name,difficulty,dscp,inputs,outputs,classification}	{result}
获取题目	GetProblem	{token,id}	{result}
登录	Login	{id,pwd}	{result,token}
注册	Register	{id,pwd,problem,answer}	{result}
由Token获取ID	GetUid	{token}	{uid}
由UID获取用户信息	GetUserInfo	{uid}	{uid,token,name,sex,dscp,question,answer,img}
创建用户	NewUser	{token,name,sex,dscp}	{result}
更新用户信息	UpdateUser	{token,name,sex,dscp}	{result}
修改密码	AlterPassword	{token, answer, newpwd}	{result}
获取做题记录	GetUserRecord	{token, uid, page, offset}	{record} record为列表
获取题目列表	All	{token,page,offset,class,diff,status}	{result} result为列表
添加题目	AddProblem	{token,id,name,difficulty,dscp,inputs,outputs,classification}	{result}
获取题目	GetProblem	{token,id}	{result}

图 4 通讯接口

## 6. 数据存储设计

- OJ 服务器交互的数据来自于两个部分，分别是数据库和文件系统中的数据。

文件系统中存储的是题目中的测试数据与测试的参考输出。通过服务器直接与文件系统增删改查等操作。

数据库中主要存储的是题目的描述、讨论的回复、提交记录 and 用户数据等；设计了如下八张表格。具体作用如下：

- Login: 用户登录信息
- OJUser: 用户信息
- Problem: 题目信息
- Classification: 分类信息
- ProblemClassification: 类题对应
- UserProblem: 用户题目状态
- Record: 提交状态信息
- Comments: 评论信息

数据库列与关系设计如下图 5 所示：

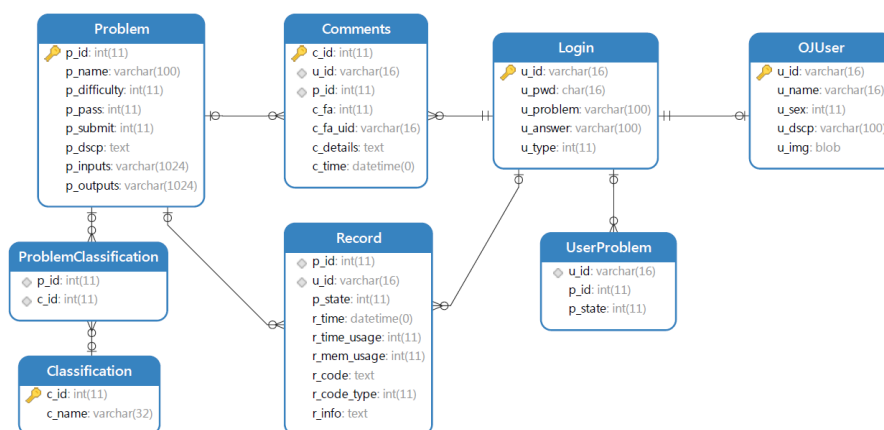


图 5 数据库设计

## 三、分工实施

### 1. 迭代设计

完成系统总共设计四个迭代，分别分为迭代 0、1、2、3，按照重要程度和工作量进行评估后，等分工作量确定迭代内容。



图 6 迭代设计

#### 1.1 迭代 0

- 确定选题：完成题目的选定与大体思路的设计，以及需求的收集。
- 用户故事：确定用户故事并转换为后续迭代的需求。
- 系统设计：设计系统的框架、数据库等内容。

#### 1.2 迭代 1

- OJ 服务框架：根据选用的技术完成具体的运行，在需求的要求下完成 OJ 服务框架。
- Web 服务框架：完成核心插件的整合和完成相关的框架构架，并完成与 OJ 服务器的交互库。
- 用户功能：在两个服务器完成用户数据操作的实现，以及完成相关部分的界面。

## 1.3 迭代 2

- 判题核心：完成系统的核心功能，完成对需求中语言支持，性能需求等。
- 管理功能：管理源用户的特有功能的实现。
- 练习功能：题目练习中提交、执行等功能的实现。

## 1.4 迭代 3

- 搜索功能：题目筛选的功能。
- 集成测试：完成功能测试和界面测试。
- 云端部署：将服务部署到阿里云服务器。

## 2. 小组分工

项目总由范泽奇、朱逸宸、黄苏珍完成，工作分配主要考虑到工作量以及内容选择意愿进行分配。总的来说范泽奇负责前端相关内容，朱逸宸和黄苏珍负责后端相关内容。

项目		人员
Web服务器	框架	范泽奇
	Web服务核心	
	IDE框架	
OJ服务器	框架	朱逸宸、黄苏珍
	判题核心	
	数据库	
	日志	

图 7 人员分工

## 3. 实施与管理

### 3.1 项目进度管理

项目进度管理通过 TAPD 平台进行，迭代 0 设计项目、完成用户故事，同时根据用户故事确定迭代内容以及发布相关任务。同时还通过 TAPD 管理项目文档。其中用到的迭代、任务、需求、文档和任务墙灯功能；连接为 <https://www.tapd.cn/53649634/>。具体情况如下图 8-11 所示：





图 8 迭代管理



图 9 迭代任务管理

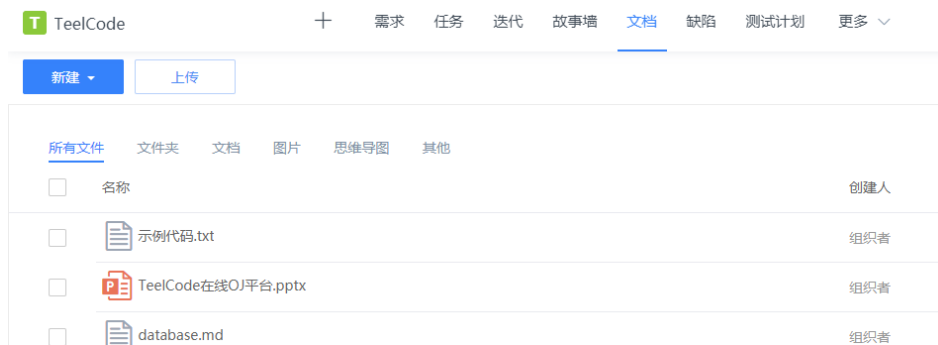


图 10 迭代文档管理



图 11 任务墙

### 3.2 代码管理

代码管理通过 GitHub 管理，远程库由项目组成员共同管理，通过 git 实现代码同步和代码合并等工作；连接为 <https://github.com/Coder-0x7ffffff/TeelCode>。如下图 12-14 所示：

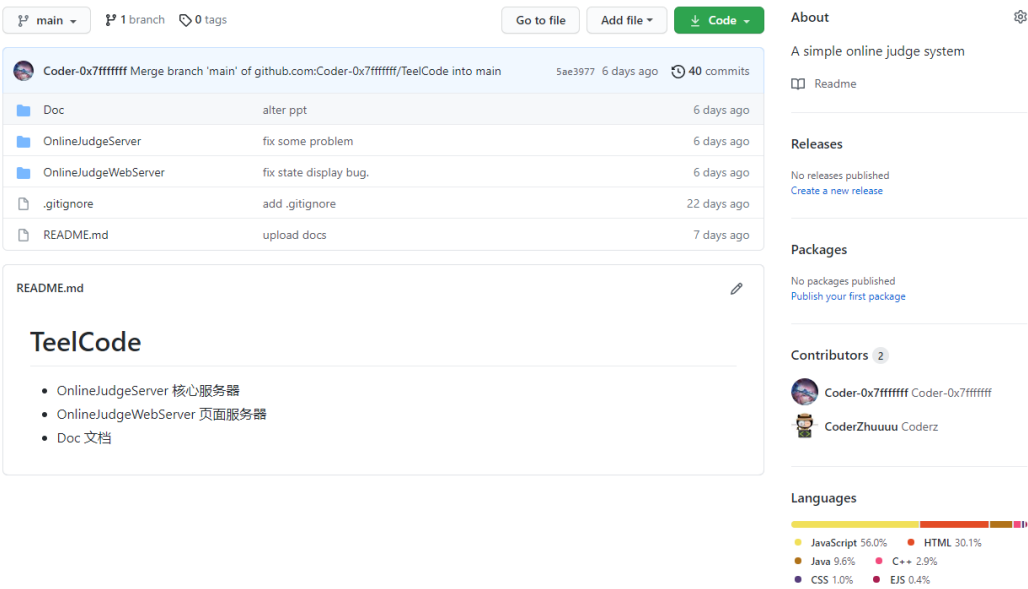


图 12 GitHub 主页展示



图 13 代码提交统计

#### Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.

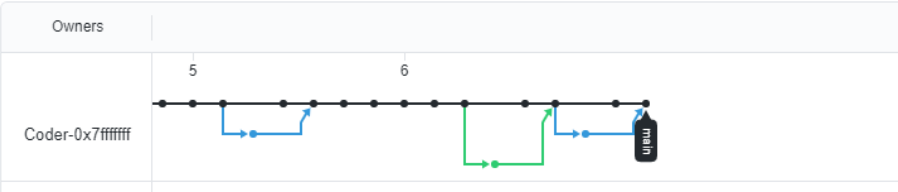


图 14 分支情况展示

## 4. 软件测试

测试分为两个部分，第一部分是 OJ 服务器的单元测试；第二部分是功能测试；第一部分，通过编写脚本对 OJ 服务器进行单元测试，脚本在源代码中可以找到。第二部分，通过人工对功能进行测试。其中测试结果如下图 15-16 所示：

**Unit Test Report**

Duration: 0:19:12.526711  
Status: Pass 17

Show [Summary](#) [Failed](#) [All](#)

Test Group/Test case	Count	Pass	Fail	Error	View
script.TestDefine	17	17	0	0	<a href="#">Detail</a>
Total	17	17	0	0	

图 15 单元测试结果

缺陷汇总			
时间	标题	描述	状态
2021.4.26	判题核心无法识别+号	提交代码+号倍识别为空格	解决
2021.5.1	提交记录记录无效	提交之后，所有的记录任然保持为0	解决
2021.5.3	Token失效时长异常	Token失效时长过短	解决
2021.5.4	个人资料中文支持出现乱码	个人资料中输入中文保存后返回乱码	解决

图 16 功能测试缺陷汇总

## 5. 开发部署环境

### 5.1 开发环境

OJ 服务器开发环境：

- 系统：Ubuntu 18.04 x64 Desktop
- 语言：Java/C++
- 环境：JDK8/C++11
- IDE：Eclipse/VS

Web 服务器开发环境：

- 系统：Windows10 1809 LTSC
- 语言：JavaScript
- 环境：Node.js-14.15.3
- IDE：WebStorm

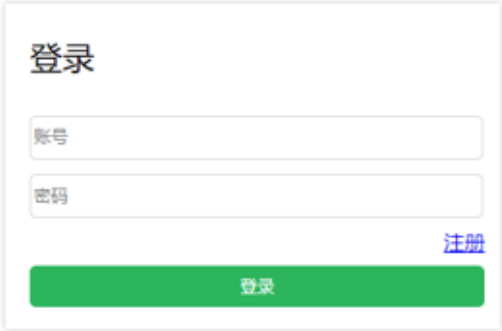
### 5.2 部署环境

部署环境：

- 系统：Ubuntu 18.04 x64
- 配置：Aliyun 2 核 4 GiB
- 带宽：5M
- JDK：JDK8
- Node.JS：8.10.0

## 四、成果展示

成果最终部署在了 <http://oj.xiami.space/>，下方图 17 到 29 为具体演示：



登录

账号

密码

[注册](#)

登录

图 17 登录界面展示



注册

账号

密码

确认密码

验证问题

问题答案

注册

图 18 注册页面展示



首页 退出

难度: 全部 分类: 全部 状态: 全部 筛选

编号	题目	难度	通过率	状态
0	两数之和	简单	无提交	未完成

加载更多

图 19 题目主页展示



首页 退出

个人资料

- 做题记录
- 修改密码
- 分类管理
- 题目管理

退出

个人资料

用户ID: admin

用户名: admin 修改

描述: this is admin 修改

图 20 个人资料展示



图 21 分类管理展示



图 22 题目管理展示



图 23 添加题目展示



图 24 问题页面展示



图 25 题目 Markdown 描述展示

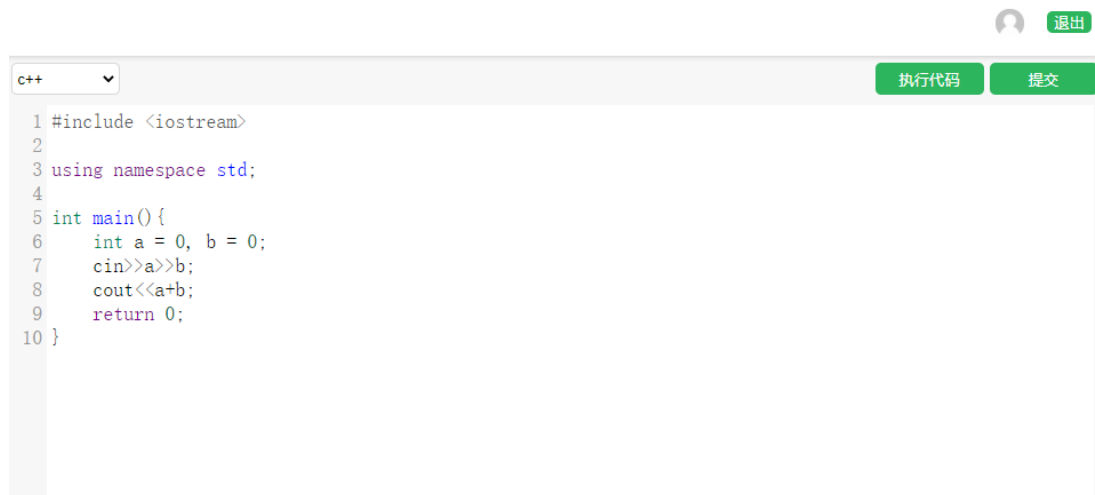


图 26 代码 CodeMirror 展示

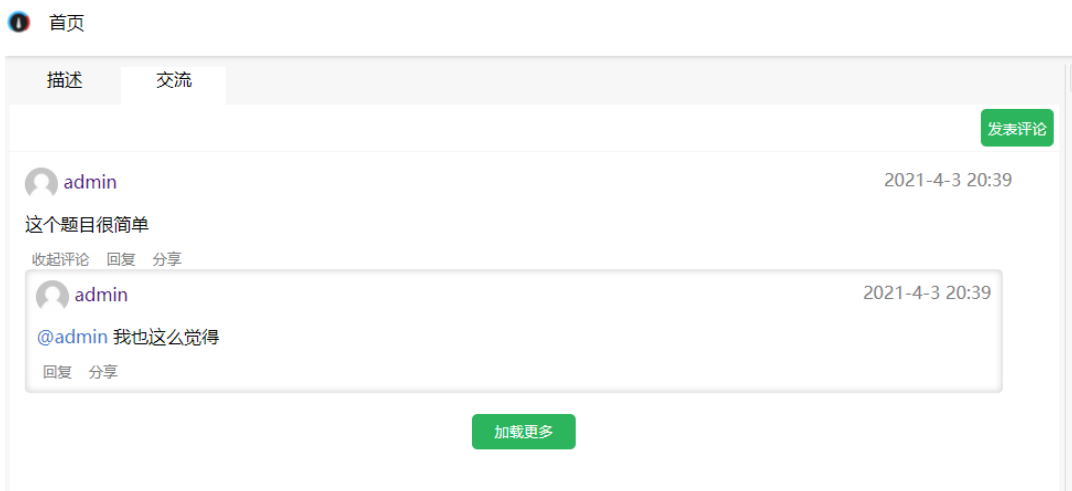


图 27 评论区回复展示



图 28 提交代码结果展示



图 29 执行代码结果展示

## 五、总结展望

### 1.总结

首先我们得到了完整的项目体验：项目从规划、设计、实施、管理最后到部署。我们利用敏捷开发模式完成了一整个项目；体验到了其带来的优点比如快速看到项目成果、增量的方式增加需求等优点。

然后完成了一个完整的在线 OJ 系统；整体上完成了一个在线 OJ 系统，其支持在线评测；支持多种 C/C++、Java、Python2/3 语言；拥有完整的用户系统；支持对题目进行讨论和回复；支持在线增删分类以及题目。

## 2.展望

系统完成度处于优良阶段，整体上可用性和可扩展性都比较高，但是也有可以改进的地方，在这里提出三点可以增强的地方，如下：

- 支持竞赛系统：系统希望在将来能够提供一个竞赛模块，管理员能够在线管理竞赛，用户可以通过报名参加竞赛同时支持排名等。
- 积分金币系统：积分系统支持对用户进行排名，通过练习或者竞赛均可以获得积分，除此之外练习和竞赛均可获得金币可以用于周边商场兑换实体或虚拟奖品。
- 更公平的评测：系统在中底负载的情况下得到的结果准确，但在高负载的情况下有可能导致评测不准确等；未来将结合分布式技术独立部署评测核心，从而做到更高的负载和更公平的评测

## 附录

项目管理连接：<https://www.tapd.cn/53649634/>

代码管理连接：<https://github.com/Coder-0x7fffffff/TeelCode>

部署地址：<http://oj.xiami.space/>