项目名称：tzlOJ

文档名称：项目开发总结报告

*HUSTZL*

*侯皓斐 软件2003班 U202010851*

*刘铭宸 软件2003班 U202010783*

*甘凤轩 软件2003班 U202010822*

版本：V2.1

文 档 信 息

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [√ ] 正式发布  [ ] 正在修改 | 文件标识： | HUSTZL\_011 |
| 当前版本： | V2.1 |
| 作者： | 刘铭宸 |
| 发布日期： | < 2023.4.23 > |

文 档 更 改 记 录

| 版本 | 更改日期 | 更改人 | 更改原因 | 说明 |
| --- | --- | --- | --- | --- |
| V1.0 | 2023.4.15 | 刘铭宸 |  | 初版 |
| V2.0 | 2023.4.20 | 甘凤轩 | 校对 | 勘误并调整行文语句 |
| V2.1 | 2023.4.23 | 甘凤轩 | 调整格式 | 完善文档大纲，调整段落格式，修改序列编号 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1.引言 1](#_Toc133159421)

[1.1编写目的 1](#_Toc133159422)

[1.2背景 1](#_Toc133159423)

[1.3定义 1](#_Toc133159424)

[1.3.1Vue.js 1](#_Toc133159425)

[1.3.2Django 1](#_Toc133159426)

[1.3.3Redis 2](#_Toc133159427)

[1.3.4PostgreSQL 2](#_Toc133159428)

[1.3.5Sandbox： 2](#_Toc133159429)

[1.3.6Docker 2](#_Toc133159430)

[1.4参考资料 3](#_Toc133159431)

[2.实际开发结果 3](#_Toc133159432)

[2.1产品 3](#_Toc133159433)

[2.2主要功能和性能 6](#_Toc133159434)

[2.2.1功能 6](#_Toc133159435)

[2.2.2性能 7](#_Toc133159436)

[2.3基本流程 8](#_Toc133159437)

[2.4进度 10](#_Toc133159438)

[2.5费用 10](#_Toc133159439)

[3.开发工作评价 10](#_Toc133159440)

[3.1对生产效率的评价 10](#_Toc133159441)

[3.2对产品质量的评价 10](#_Toc133159442)

[3.3对技术方法的评价 10](#_Toc133159443)

[3.4出错原因的分析 11](#_Toc133159444)

[4.经验与教训 11](#_Toc133159445)

# 1.引言

## 1.1编写目的

本文档的编写目的是针对tzlOJ系统项目的开发进行总结和评价，其中总结包括对于实际的开发成果与开发过程。

本文档的预期读者为：系统设计人员、系统开发人员，系统测试人员、系统维护人员，最终用户及其它有权限查阅本文档的相关人员。

## 1.2背景

* 系统名称：tzlOJ系统
* 任务提出者：陈长清
* 开发者（承接单位）：侯皓斐，刘铭宸，甘凤轩
* 用户：高校教师，高校学生

tzlOJ是一个OnlineJudge系统，旨在为软件学院的教师和学生提供一个在线编程评测平台，以帮助提升编程技能和竞赛能力。该平台将提供一系列编程题目和评测系统，用户可以在平台上提交自己的代码，系统将自动评测代码的正确性和效率，并给出相应的评分和反馈。本项目支持多种竞赛模式，支持创建比赛，创建题单，还可以查看自己的ranking。在易用性指标上我们对标leetcode或luogu等知名OJ，提供给用户好的使用环境，从而便于同学和教师更好的举行教学活动。

## 1.3定义

### 1.3.1Vue.js

Vue.js是一种用于构建用户界面的渐进式JavaScript框架，它的核心库仅关注视图层，使得集成其他库变得更加容易。Vue.js支持双向数据绑定、组件化开发、虚拟DOM等特性，可以帮助开发者快速构建可维护且高性能的前端应用程序。

### 1.3.2Django

Django是一个基于Python的高级Web应用框架，遵循MVC（Model-View-Controller）设计模式。它提供了大量内置的功能，如：用户认证、权限管理、路由控制、表单验证等。Django的目标是简化Web应用的开发，让开发者能够专注于编写业务逻辑，而不是处理底层的HTTP请求和响应。

### 1.3.3Redis

Redis（REmote DIctionary Server）是一个开源的、高性能的内存数据结构存储系统，可以用作数据库、缓存或消息代理。它支持各种数据类型，如：字符串、列表、集合、有序集合等。Redis具有高速访问、持久化存储、分布式支持等特点，广泛应用于各种场景，如：缓存、队列、排行榜等。

### 1.3.4PostgreSQL

PostgreSQL是一种强大的开源关系型数据库管理系统，支持高并发、ACID事务特性、数据完整性和安全性等。它具有丰富的数据类型、灵活的查询语言、全文搜索、JSON支持等功能，使得开发者能够满足各种复杂的数据处理需求。PostgreSQL还具有良好的扩展性，可以通过插件、存储过程等方式进行扩展。

### 1.3.5Sandbox：

沙箱技术是一种用于创建独立、隔离的运行环境，在这个环境中执行用户提交的代码，确保系统安全和资源的分隔。沙箱可以限制代码的运行权限，防止恶意代码对系统产生影响。在OnlineJudge系统中，沙箱技术用于评测用户提交的代码，确保评测过程的安全性。

### 1.3.6Docker

Docker是一种开源的容器化平台，允许开发者将应用及其依赖打包到轻量级的可移植容器中。容器内的应用可以在任何支持Docker的系统上运行，实现应用的快速部署和可扩展性。Docker具有高度的隔离性，可以实现多个应用在同一台主机上独立运行，有效降低资源消耗和运维成本。Docker还提供了Docker Hub，一个公共的容器镜像仓库，供开发者分享和获取各种应用的容器镜像。

在tzlOJ系统中，各技术的功能如下：

1. Vue.js：负责构建前端界面，实现与用户的交互。
2. Django：构建后端服务器，处理前端发送的请求，与数据库进行交互以完成数据的存储和检索。
3. Redis：提供缓存功能，提高系统性能；实现消息队列，进行异步任务处理。
4. PostgreSQL：作为关系型数据库，存储系统中的数据，如用户信息、题目、提交记录等。
5. Sandbox：创建隔离的评测环境，安全地运行用户提交的代码。
6. Docker：实现应用的容器化部署，简化运维流程，确保系统在不同环境下的一致性。

## 1.4参考资料

**[1]** 软件文档写作 陈长清著 清华大学出版社 第一版

**[2]** tzlOJ需求说明书V2.0

**[3]** tzlOJ概要设计说明书V2.0

**[4]** tzlOJ项目开发计划 V2.0

# 2.实际开发结果

## 2.1产品

最终产品名称：tzlOJ系统

产品版本：1.0

包含子系统如下：

前端子系统：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 模块名称 | 模块标识 | 输入 | 处理 | 输出 |
| 1 | 学生提交模块 | submission | 学生代码、题目ID、语言类型 | 检查提交格式、向后端发送评测请求 | 提交结果、评测报告 |
| 2 | 管理员管理模块 | admin | 管理员登录信息、题目信息、比赛信息 | 管理员身份验证、管理题目和比赛信息 | 成功或失败信息 |
| 3 | 用户界面模块 | user | 用户登录信息、题目信息、排行榜信息 | 显示用户信息、题目信息、排行榜信息 | 用户界面显示 |

后端子系统：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 模块名称 | 模块标识 | 输入 | 处理 | 输出 |
| 1 | 自动评测系统 | judge | 学生代码、题目信息、测试数据 | 编译和运行学生代码，生成评测报告 | 评测结果、评测报告 |
| 2 | 题目管理系统 | problem | 题目ID、题目信息、测试数据 | 管理题目和测试数据 | 题目和测试数据的增删改查 |
| 3 | 用户管理 | user | 用户信息、登录信息 | 管理用户信息 | 用户信息的增删改查 |
| 4 | 比赛管理 | contest | 比赛信息、题目信息 | 管理比赛和题目信息 | 比赛和题目信息的增删改查 |
| 5 | 排行榜子系统 | leaderboard | 用户积分信息、排行榜信息 | 管理用户积分信息和排行榜信息 | 排行榜信息的查询和更新 |
| 6 | 评测机管理模块 | JudgeManagement | 测评的代码，题目，数据 | 进行数据分析 | 合适的评测机 |

Redis子系统：（使用第三方现成库）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 模块名称 | 模块标识 | 输入 | 处理 | 输出 |
| 1 | Redis缓存模块 | redis | 数据库读写请求 | 查询缓存、更新缓存 | 查询结果或更新结果 |

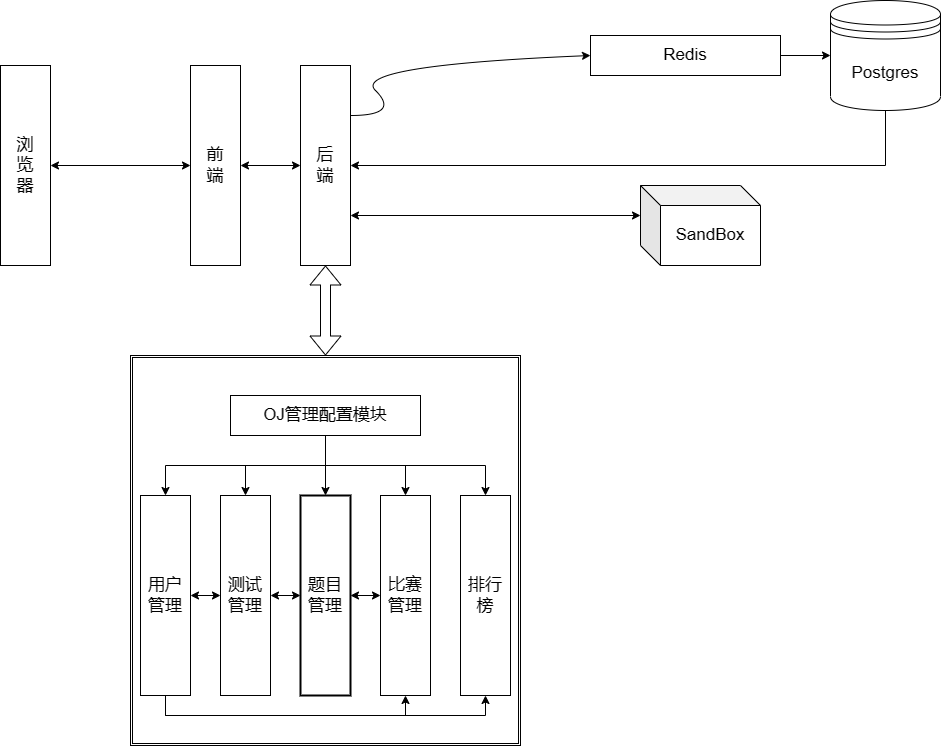
Postgres子系统：（使用第三方现成库）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 模块名称 | 模块标识 | 输入 | 处理 | 输出 |
| 1 | 数据管理模块 | DataManagement | 信息 | 存储信息 | 存储结果 |
| 2 | 数据访问接口 | DataAccess | 数据库访问请求 | 查询和操作数据库 | 数据库查询和操作结果 |

沙箱子系统：（使用第三方现成库）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 模块名称 | 模块标识 | 输入 | 处理 | 输出 |
| 1 | 编译模块 | Compilation | 用户提交的代码和编译选项 | 编译用户提交的代码 | 编译结果 |
| 2 | 执行模块 | Execution | 用户提交的代码和测试数据 | 运行用户提交的代码 | 运行结果和输出 |
| 3 | 安全模块 | Security | 用户提交的代码和运行环境 | 隔离用户提交的代码并保证安全 | 隔离结果 |

tzlOJ系统结构：



本软件开发过程中产出的文档有：

* 商业计划书
* 项目开发计划
* 需求说明书
* 架构设计说明书
* 概要设计说明书
* 详细设计说明书
* 数据库设计说明书
* 系统测试计划
* 系统测试报告
* 用户手册
* 项目开发总结报告

## 2.2主要功能和性能

### 2.2.1功能

**用户功能表：**

|  |  |  |
| --- | --- | --- |
| **功能** | **说明** | **目标完成情况** |
| 登录与注册 | 用户可以通过输入用户名和密码来登录，如果没有账号，可以通过注册页面内创建新账号。 | 已完成 |
| 题目列表 | 系统展示所有可用编程题目列表，用户可输入搜索条件来对题目进行筛选和排序。 | 已完成 |
| 题目详情页 | 用户可以查看题目的详情页，包括题目描述、输入输出样例、评测结果和题目提交历史等信息。 | 已完成 |
| 代码编辑器 | 系统提供一个代码编辑器，用户可在其中编辑和提交自己的代码。编辑其支持多种语言 ，并提供自动缩进、语法高亮、自动补全等功能。 | 已完成 |
| 评测结果查看 | 用户可以查看自己提交的代码的评测结果，包括编译错误、运行错误、通过测试点的数量等信息。 | 已完成 |
| 个人信息修改 | 用户可以查看和编辑自己的个人信息，包括用户名、密码、邮箱等信息。 | 已完成 |
| 参加比赛 | 用户可以通过比赛页面查看和参加系统中的比赛。比赛页面应该展示比赛的基本信息、排名、比赛规则和题目列表等信息。 | 已完成 |
| 排行榜 | 用户可以通过积累积分和参加比赛等活动来提高自己的排名。排行榜页面应该展示用户的排名和积分等信息。 | 已完成 |

**管理员功能表：**

|  |  |
| --- | --- |
| 管理用户 | 管理员可以导入、删除、增添、修改用户信息，并拥有对用户进行封禁的权限。 |
| 管理题目 | 管理员可以修改、增添、删除题目的信息，如类型、描述、输入输出样例、提交历史等，同时还可以增添新题目或删除旧题目。 |
| 管理比赛 | 管理员可以创建比赛并设置或修改比赛的相关信息，如时间、题目列表、相关规则等 |

### 2.2.2性能

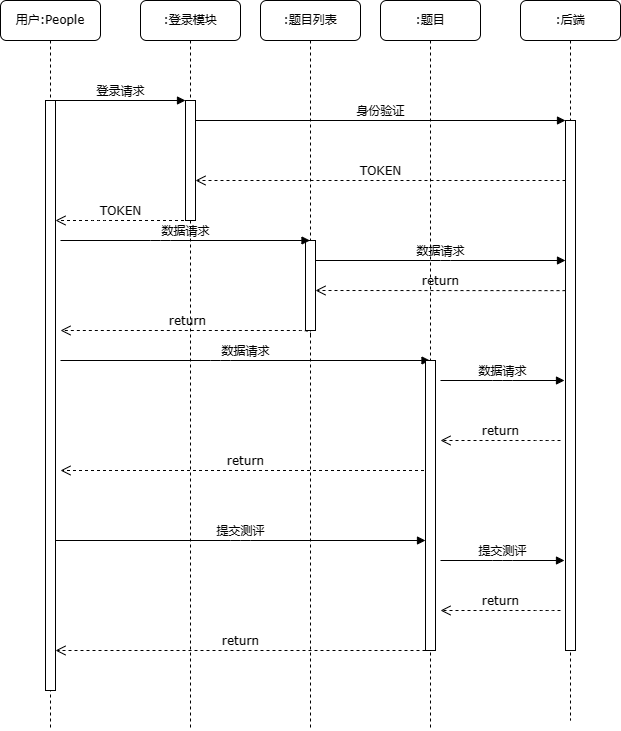
1. 系统响应时间：系统应该在提交程序后迅速响应，返回程序编译和运行的结果。对于简单程序，系统响应时间应该在几秒钟内完成。对于复杂的程序，响应时间也不应超过10分钟。
2. 系统容量：系统应能够支持大量用户同时访问和提交程序，同时应能够处理大型文件和数据。系统容量应该能够满足高峰期的需求，确保系统稳定运行。
3. 系统稳定性：系统应该具有高度的稳定性和可靠性，能够保证在系统繁忙或出现故障时，不会丢失用户的数据和程序。系统应该能够在故障恢复后自动恢复运行，确保系统的连续性。
4. 系统错误处理：系统应该能够处理所有的错误和异常情况，并给出相应的错误提示信息，告知用户错误的原因和解决方法。同时，系统应该记录错误信息和异常情况，便于后期问题的追踪和解决。
5. 数据完整性和一致性：系统应该保证提交的程序和数据的完整性和一致性，不应出现数据损坏或丢失的情况。系统应该具有数据备份和恢复功能，确保数据的完整性和可靠性。
6. 可维护性：该OJ系统需要具备易于维护和升级的特性，包括可配置性、可扩展性、可移植性、易于维护和易于修改等。需要使用标准化的开发框架和编程语言，保证代码易于维护和扩展。同时，需要提供完善的日志系统，以便管理员及时发现并处理问题。
7. 可测试性：该OJ系统需要具备易于测试的特性，以确保系统功能的正确性和稳定性。需要提供自动化测试和手动测试两种方式，可以对不同模块进行测试。同时，需要提供详细的测试文档和测试报告，以便开发人员和测试人员进行系统测试和问题定位。

## 2.3基本流程

系统的整体处理流程如下：

* 用户登录：用户通过前端界面输入用户名和密码进行登录，系统通过后端接口验证用户身份。
* 题目列表：用户可以在前端界面中查看所有可用的编程题目列表，并选择相应的题目进行解答。
* 代码编辑：用户在前端界面中输入代码，并提交到后端进行评测。
* 代码评测：后端评测机根据题目要求，自动编译和运行用户提交的代码，并根据测试数据生成评测报告。
* 评测结果：后端评测机将评测结果返回给前端界面，用户可以在前端界面中查看评测结果，包括编译错误、运行错误和通过测试点的数量等信息。
* 排行榜：系统支持积分系统和排行榜功能，用户可以通过积累积分和参加比赛等活动来提高自己的排名。排行榜页面展示用户的排名和积分等信息。

我们可以用时序图来描述系统的核心的处理流程（题目测评），如下图所示：



## 2.4进度

项目原定开发周期三个月，实际开发用时两个月，要比计划中的提前完成任务。项目提前完成的原因是项目组内的分工比较明确，团队内的沟通与协作比较有效，项目组成员们合作开发效率比较高。

## 2.5费用

本项目的开发费用主要包括：上网查找资料的费用，购买参考书籍的费用与购买硬件的费用，总共经费20000元，经费来源于项目开发组。

# 3.开发工作评价

## 3.1对生产效率的评价

* 项目的实际开发效率是每人每月生产3000行代码。
* 文件的平均生产效率是每人每月生产15千字。
* 原定计划开发效率每人每月生产2000行代码。
* 文件的计划效率每人每月生产15千行字。
* 生产效率总体来说符合预期。

## 3.2对产品质量的评价

在测试中整个项目一共发现了124个bug，错误的发生率约为6个错误/千行代码，总体来说项目的质量符合要求，原因是我们采用了严格的开发流程，和编程过程中严格遵守了编程规范。

## 3.3对技术方法的评价

tzlOJ系统的基本设计思想是提供一个支持多种编程语言的自动评测系统，使学生可以在线提交代码，并获得自动评测结果。系统采用前后端分离的架构，前端负责提供用户界面和交互逻辑，后端负责处理系统的业务逻辑和数据处理。

在软件架构样式的选择上，我们采用了三层B/S模式，将系统分为三个层次：用户界面层、应用层、领域和基础设施层。

1. 用户界面层：用户界面层是OnlineJudge系统的顶层，负责与用户进行交互。该层提供用户界面和交互逻辑，包括学生提交和管理员管理等功能。
2. 应用层：应用层负责处理系统的业务逻辑和数据处理。该层包括自动评测系统、用户管理管理和比赛管理等模块。
3. 领域和基础设施层：领域和基础设施层负责系统的数据模型和数据访问。该层包括用户信息、成绩信息和提交历史等领域模型，以及数据访问接口和服务，而且是OnlineJudge系统的底层支撑，负责提供系统的底层服务和基础设施，特别是运行和编译程序的SandBox。领域和基础设施层负责将业务逻辑和数据逻辑进行分离，将业务逻辑和测试逻辑进行分离，并提供标准化的数据访问接口，以便应用层进行数据访问和处理。同时将测评与系统分离，放入一个安全的封闭的模拟沙箱，从而保证了系统的安全性。

## 3.4出错原因的分析

我们在开发过程中出现错误的原因如下

1. 对于项目所用到的技术栈熟悉程度不够。
2. 没有掌握部分功能的实现算法。
3. 对于不同模块之间交互的接口不够熟悉。

# 4.经验与教训

在开发OnlineJudge系统的过程中，我们积累了许多宝贵的经验，并从中学到了一些教训:

1. 明确需求与目标：在项目开始之前，我们应充分了解需求，明确系统目标。这有助于避免在开发过程中出现需求不明确或需求变更的情况，从而提高开发效率。
2. 合理规划时间与资源：在项目开始时，我们应根据需求和预期目标合理规划时间表和资源分配，以确保项目按时完成。在开发过程中要关注项目进度，以便及时调整计划。
3. 选择合适的技术栈：根据项目的实际需求选择合适的技术栈十分重要。我们在本项目中采用了Vue.js、Django、Redis、PostgreSQL、Sandbox和Docker等技术，这些技术的组合使得系统具有高性能、可扩展性和安全性。
4. 注重代码质量与可维护性：在开发过程中，我们应遵循编码规范，编写简洁、清晰的代码，以提高代码质量和可维护性。此外，编写详尽的文档和注释也有助于后期的维护工作。
5. 加强团队协作与沟通：开发过程中，团队成员之间的协作与沟通至关重要。我们应确保团队成员对项目进度和任务分工有清晰的了解，并在遇到问题时积极沟通，共同寻求解决方案。
6. 重视测试与评估：在开发过程中，我们应充分进行各类测试，如单元测试、集成测试和系统测试，以确保系统功能的完整性和稳定性。同时，通过对系统进行性能评估，我们可以找到潜在的性能瓶颈并进行优化。
7. 及时调整与改进：在开发过程中，我们可能会遇到预期之外的问题或挑战。这时，我们需要灵活调整计划，并根据实际情况进行改进，以确保项目的顺利进行。