**汕头大学大学生创新创业训练计划项目**

**申报书**

|  |  |
| --- | --- |
| 项目所在学院： | 工学院 |
| 项 目 名 称： | 面向计算机程序设计竞赛的在线评测系统 |
| 项 目 类 型： | √ 创新训练项目 |
| □ 创业训练项目 |
| □ 创业实践项目 |
| 负 责 人： | 张健 |
| 指 导 教 师： | 王珊珊，许建龙 |
| 所属一级学科： | 计算机科学与技术 |

**汕头大学教务处**

**年 月**

**填 写 须 知**

一、项目分类说明：

1.创新训练项目是本科生个人或团队，在导师指导下，自主完成创新性实验方法的设计、设备和材料的准备、实验的实施、数据处理与分析、总结报告撰写等工作。

2.创业训练项目是本科生团队，在导师指导下，团队中每个学生在项目实施过程中扮演一个或多个具体的角色，通过编制商业计划书、开展可行性研究、模拟企业运行、进行一定程度的验证试验，撰写创业报告等工作。

3.创业实践项目是学生团队，在学校导师和企业导师共同指导下，采用前期创新训练项目的成果，提出一项具有市场前景的创新性产品或者服务，以此为基础开展创业实践活动。

二、申报书请按顺序逐项填写，填写内容必须实事求是，表达明确严谨。空缺项要填“无”。

三、填写时可以改变字体大小等，但要确保表格的样式没有被改变；填写完后用**A4纸张双面打印**，不得随意涂改。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目名称 | | 面向计算机程序设计竞赛的在线评测系统 | | | |
| 项目起止时间 | | 2023年5月 至2023年5月 | | | |
| 负责人 | 姓名 | 年级 | 所在学院、专业 | 联系电话 | E-mail |
| 张健 | 2020级 | 工学院、计算机科学与技术 | 18924240199 | 20jzhang@stu.edu.cn |
| 项目组成员 | 张嘉豪 | 2020级 | 工学院、计算机科学与技术 | 15382857905 | 20jhzhang@stu.edu.cn |
| 张雷明 | 2020级 | 工学院、数据科学与大数据技术 | 17620663547 | 20lmzhang@stu.edu.cn |
| 谢璧蔚 | 2022级 | 工学院、计算机科学与技术 | 17703074697 | 22bwxie@stu.edu.cn |
|  |  |  |  |  |
| 指导教师 | 姓名 | 王珊珊 | | 职务/职称 | 副教授 |
| 所在学院（单位） | 工学院 | | | |
| 联系电话 | 15142813890 | | E-mail | sswang@stu.edu.cn |
| 指导教师 | 姓名 | 许建龙 | | 职务/职称 | 讲师 |
| 所在学院（单位） | 工学院 | | | |
| 联系电话 | 13415177616 | | E-mail | xujianlong@stu.edu.cn |
| **一、项目简介**  OJ即Online Judge System 在线评测系统，用户可以在线提交程序源代码，由oj对源代码进行编译和运行，并通过每道题事先设计好的数据以及时空限制来判断用户程序是否正确。oj是一种在编程竞赛中用来评测参赛程序的在线系统，也可以用于平时练习。  OJ最初使用于ICPC和OI竞赛中，随着时间的推移，OJ以及广泛的应用于各大高校、组织的线上学习与竞赛中。许多OJ网站会自发组织一些竞赛。此外，OJ网站通常会设立用户排名，以用户的提交答案通过数多少或某个题目执行时间快慢为排名依据。  市场上有众多OJ平台如洛谷、LibreOJ、BZOJ，一些较老的开源OJ如UOJ、Vijos、Hustoj，初学者oj如一本通、OpenJudge，各大高校OJ如HDUOJ、POJ、ZOJ、FZUOJ，一些国外OJ如Codeforces、AtCoder、SPOJ、UVaOJ、USACO、CodeChef、Topcoder、CS Academy、Ural等。  已有的汕头大学校内OJ即CodeZ并未开发用户之间交互的功能，缺乏用户交互性；未对用户数据进行整理和汇总，难以激发用户积极性；比赛系统仅支持单人的acm与IOI赛制，不能够满足用户的需求；缺乏全站用户信息统计，对于用户的个性化而言制作的不尽人意；CodeZ使用传统的判题逻辑，由于每到题目的用例有限而无法杜绝投机取巧的通过方法，一般出题者难以给出完善的用例，对小出题者而言十分不友好；同时CodeZ不支持动态部署节点，不能应对突增的压力；在设计时使用了redis作为消息队列，在宕机时会导致消息丢失；不支持心跳检测，无法对宕机的节点进行调整。  下图（图1、图2）分别为CodeZ的比赛创建页面以及CodeZ的题目提交页面。  ecee314d92d58e694b1fb6304f370d1  **图1：CodeZ的比赛创建**  7be8aab3c797233077008ab65bf713f  **图2：CodeZ的题目提交**  对于先前的CodeZ，我们对OJ进行了功能上的整改与创新并研发了DOJ平台，及dream online judge。  为了提高平台的交互性，我们搭建了文章、QA、公告、题解、收藏、比赛讨论区、比赛实时滚榜、好友、标签与文本搜索、私信、小组聊天等相关服务，以此增加社区的信息量，以此提高社区氛围与用户在平台中学习的积极性。这些用户交互性强的服务可以带来更好的用户体验。  通过前期调研，我们的OJ比赛系统根据用户的需求开发了更多的赛制，包括但不限于小组、个人、随机匹配、标准模式的IOI，OI，ACM机制竞赛。OJ可以以个人或者小组的名义参加或举行比赛，同时为了增添比赛的随机性，我们还提供了随机匹配的方式为参赛个人组织用户组进行比赛。此外，我们还为标准比赛配置了标准模式，即由系统分配指定数量的参赛账号，并由官方下发到参赛选手的手中。为了增加用户的竞赛及时性，我们还增加了及时比赛，报名的用户将会进入及时队列，及时队列将会根据队列人数与等待时间决定为报名用户创建比赛的时机，比赛内容不需要人为设置便可自动生成。该OJ引入了竞赛分机制，优秀的参赛选手可以竞赛分数的奖励以此激励用户的探索欲。除此之外，对于非计分比赛，我们允许创建者直接引用题库中的题目。  此处我们给出同为OJ平台的力扣社区页面（图3），可以看到有大量用户参与了力扣社区，并提供了一定的有价值的内容。  e108f870050f40a9c9293bed033c3ef  **图3：同为OJ平台的力扣社区**  为了更好的满足用户需求，我们开发了exam模板，支持oj题目、选择题、填空题、简答题，在这之中填空题与简答题支持正则表达式，除了用户组组长可以发布作业，用户也可以发布考试，为每道题目赋予分值。除此之外，我们复用了竞赛中的标准模式以此增设了标准模式下的机试功能。  由于OJ题目本身的测试用例有限，传统的OJ判题逻辑容易出现投机取巧，特别针对IOI类型的竞赛，可以通过反复提交代码获取测试用例而达到“照着答案”输出的结果。为了杜绝此类现象的发生，区别于传统的OJ判题逻辑，我们增设了更严谨的判题逻辑，即hack功能、重判功能。hack功能允许用户针对某次提交制作新的测试用例，新的测试用例通过输入测试后，将用于标准程序与目标提交进行输出比对，若输出有误，则判题错误，以此杜绝穷尽测试用例的方法。简单地说，这其实给予了广大用户为题目增添用例的能力。为了激励用户hack提交代码，在竞赛的hack时间中，成功hack代码的用户将会获得分数奖励。重判功能则确保了错误的竞赛结果可以得到纠正，该功能将回滚指定的所有提交，并回滚那些用户的竞赛排名，那之后重新复现比赛现场的提交情况。管理者在发现竞赛用例错误时，可以更正用例并启用重判功能，以此获得相对正确的用户排名。当然，hack功能和重判功能实际上不仅仅适用于比赛时的提交，他们也适用于其它提交。  为了杜绝竞赛中的代码抄袭，我们增设了代码相似度比对的模块，在竞赛中，管理者可是使用该模块辅助找出代码相似度过高的代码，并回滚其中作弊者的分数。  关于一些个性化的模块，我们新增了热度排名、推荐系统、成就、算法可视化的模块。热度排名使用redis的有序队列实现每日更新，并将每日新增热度定时冲刷到mysql中完成持久化。  为了应对徒增的压力，我们的OJ支持动态节点部署。通过容器心跳检测，考察容器心跳频率来确定节点是否繁忙，当节点均处于繁忙状态时自动通过容器部署的方式增设新节点。除此之外，我们使用了rabbitmq消息队列代替redis消息队列，相比起redis，rabbitmq有更加完善的MQ机制 ，除了可以存储更多消息之外，rabbitmq拥有ack机制防止宕机时造成的消息丢失。我们新增了云节点+负载均衡的方式提供云上的支持，在预设大型比赛等预知压力即将陡增的情况下，可以通过临时部署云节点的服务来达到降压的效果。同时，在日常的主要服务处理中，我们使用了 myql（主）+redis（辅），myql做为主存储，redis用于缓存，加快访问速度。此外我们引入了限流、降级和熔断的机制以防止在压力过大时服务崩溃。 | | | | | |
| **二、申请理由**（包括自身/团队具备的知识、条件、特长、兴趣、前期准备等）  **团队方面：**   1. 张雷明：具备深度学习相关的一些知识，学习和动手能力强，利用Pytorch框架在课余时间进行相关图像分类算法的研究与编写，对机器学习以及深度学习的未来抱有信心和希望。 2. 张嘉豪：熟练使用Vue全家桶进行开发；熟练使用React及其工具（React-router, Redux）；熟悉ES6语法，如：箭头函数、解构赋值、promise等；在项目中熟练运用TypeScript，如：接口、泛型、类型断言、类型约束等。对于组件化开发和hooks有一定的理解，能够在开发中复用的组件或函数进行封装。熟悉前端模块化，能够通过ESLint和代码格式化工具Prittie编写符合规范、风格统一、易于维护的前端代码。熟练使用git进行代码托管和版本管理，npm包管理工具，能够与团队写作开发。熟练使用第三方库：Axios, TainwindCss, Antd, ElementPlus, Echarts, Bootstrap等。其他：通过如百度地图API开发地图工具；通过echarts库进行可视化开发，开发过《练江可视化平台》；熟悉过渡和动画并在项目中运用。 3. 张健：熟悉go语言，了解java、c++语言，了解mysql、redis，拥有一定的数据库设计和优化能力，了解docker、docker-compose的使用，了解k8s，拥有Docker容器、Jenkins的搭建和使用经验。了解分布式并发原理，了解kafka和rabbitmq，有消息中间件的搭建经验。了解Linux系统命令，了解nginx服务，了解gin等web服务器端系统框架，熟悉RESTful接口，取得蓝桥杯C++广东省一等奖成绩，具备后端开发的知识，拥有前后端分离项目开发和部署的经验。 4. 谢璧蔚：了解C、java语言，对oj平台有一定的了解。对于oj平台的设计有浓厚的兴趣。   **前期准备：**   1. 我们在前期完成了对市场上主要oj的功能调研。我们的调研名单中包含了较老的开源OJ如Universal Online Judge与Vijos，各大高校OJ如POJ（北京大学在线评测系统）、ZOJ（浙江大学在线评测系统）、HDUOJ（杭州电子科技大学在线评测平台），国外OJ如UvaOJ、codeforces、atcoder，以及一些国内比较成熟的OJ平台如leetcode、牛客、洛谷。以下为对市场上主要oj的功能调研方面的结果，此处附带了CodeZ和DOJ。通过调研结果可以看到市面上的大多数OJ均无实现算法可视化功能，国内的OJ极少有实现hack功能，除此之外国内OJ的竞赛模式普遍较少，缺少及时随机模式以及匹配模式，难以让单个用户直接进入比赛，无法满足用户对比赛的及时性的需求。   此处给出一份各大OJ平台的功能实现图（图4）。  **9ac1cfb55e2e6ef3849f4ddfb5df91b**  **图4：OJ平台市场调研**   1. 同时我们对本校的online judge使用者进行了问卷调研，以下为各个online judge平台使用者的分布情况（图5）。   **ee34709f41bfd94d7ae65148e053fbc**  **图5：OJ使用者分布**  通过问卷调研，我们了解到85.71%的online judge平台使用者认为OJ对编程学习起到了重要作用。同时，问卷填写人提出了很多在使用online judge时遇到的问题，如下图（图6）。  **4e7f8ce1e611f32e313ddf1a5c01fc3**  **图6：使用OJ时遇到的问题**  在问卷中，我们调查了用户希望OJ中提供的支持与服务，此处给出词云图如下（图7）。  1683090523123  **图7：词云图**  通过前期的调研工作，我们了解到了部分用户需求，通过用户给出的需求与建议，我们开发出了多种竞赛模式、提单与引导、更多的用户交互区与相应功能以及作业、考试功能。   1. 项目的主要后端服务已经部署于云端。服务包含：img，简易图床；test，本地测试；judge，判题机；mgaoj，主要服务；文件系统：file，文件服务。目前服务已部署于与节点。以下是对云节点的压力测试结果，测试数据来自于牛客小白月赛61-70中C、Java、Python3语言使用者比赛期间的所有提交，我们将压力测试的结果整理为了下图（图8、图9、图10）。云节点数量：12，最长响应时间：7567ms，平均响应时间：449ms，模拟用户数量：2021，提交总数：43296，总吞吐量：43296，平均每秒吞吐量：5.7728   a6643f34603b89f9f226a9b723605d5  **图8：比赛进行时间-每五秒吞吐量**  **669c896aed1af7270391c3390a35d64**  **图9：响应时间ms-提交数量**  **427606d222be189d48daf2ff1025b8c**  **图10：比赛举行时间ms-响应时间ms**  可以看到百分之九十以上的提交的响应时间均在500ms以下，在43296个提交中出现了4个响应时间超过7000ms的尖刺，可能是同一时间连接同一节点的请求数过多导致，对总体影响不大，且在用户接受范围以内。云节点几乎可以吃下全部压力。   1. 我们使用了基于n-gram模型的词频统计实现的代码相似度判断用于检测代码抄袭的行为。为了解决代码相似度判断这个问题，我们使用优化的编译器编译代码，这将很多手工改过的等价代码优化成同样的二进制代码，并且取得每个函数的指令序列。那之后我们使用了基于n-gram模型的词频统计实现的代码相似度判断来检测代码抄袭行为。我们将抄袭手段分为了以下两种并进行了小剂量的测试，结果如下（图11）。   [T1] 修改注释、变量名、大小写等无关信息  [T2] 代码重排、风格修改等  **c95ec630b78ba881f564ee3a79b9944**  **图11：相似度比对表**  手工加工后的代码与源代码的平均相似度为0.895，最低相似度为0.72，其与非抄袭代码的最高相似度0.54仍然有一定的区别度。当然，这并非最终版本。   1. 我们采用了一种基于协同过滤算法的题目推荐系统，该系统使用 PyTorch 框架实现。我们通过对用户做题次数矩阵进行矩阵分解，生成用户和题目的隐向量，并使用神经网络来学习这些隐向量，以预测用户对其他题目的兴趣度。协同过滤算法是一种基于用户行为的推荐算法，其原理是通过分析用户之间的相似性来预测用户的兴趣。在本系统中，我们使用了基于做题次数的协同过滤算法。具体来说，我们使用一个用户-题目的做题次数矩阵来表示用户对每个题目的兴趣程度。然后，我们对该矩阵进行矩阵分解，得到用户和题目的隐向量表示。隐向量表示可以捕捉用户和题目之间的关系，从而可以用来预测用户对其他题目的兴趣程度。如下为推荐系统模型图（图12）。   2f0ad337937a665fed701720534e65a  **图12：推荐系统模型**  为了进一步提高推荐算法的性能，我们采用了神经协同过滤（NCF）算法。NCF算法将协同过滤和神经网络相结合，可以更好地处理稀疏数据和非线性关系。具体来说，NCF算法使用多层感知器（MLP）和矩阵分解器（GMF）来学习用户和物品的隐向量表示，并将它们连接在一起以预测用户对物品的评分。通过使用NCF算法，我们可以更准确地预测用户的兴趣，并生成更准确的推荐列表。  在本系统中，我们使用 PyTorch 框架实现了协同过滤算法和 NCF 算法。模型图上图所示。具体来说，我们使用 PyTorch 的张量来存储用户做题次数矩阵，并使用 PyTorch 的神经网络模块来实现 MLP 和 GMF 模型。我们将 MLP 和 GMF 模型的输出连接在一起，并使用激活函数来生成最终的预测评分。  在训练过程中，我们使用反向传播算法来更新模型参数，并使用 Adam 优化器来加速训练过程。我们还使用 K 折交叉验证来评估模型的性能，并使用精确率和召回率来衡量模型的准确度。 | | | | | |
| **三、项目方案（**计划、技术路线、人员分工等**）**  **主要项目计划流程：**  项目主要开发流程如下图所示（图13）。  **e3767c457815dc14218b48c0d31cecc**  **图13：开发流程图**  通过迭代增量式开发，每次设计和实现该在线评测平台的一部分，逐步逐步完成开发。在整个迭代代发方法中，整个开发工作组织为一系列短期的小项目，即一系列的迭代。每一次迭代都包含了需求分析、设计、实现与测试。采用这种方法，开发工作可以在需求被完整确定之前启动，并在以此迭代中完成系统的一部分功能与业务逻辑的开发工作。再通过用户的反馈来细化需求，并开始新一轮的迭代。  我们将迭代增量式开发分为了三个主要的版本，及Alpha版本，Beta版本，RC版本。其中Alpha版本为初版，其需要完成基础部分的前端以及后端的编写，并外接其它服务。在Beta版本中，我们将邀请十余名Oj使用者参与OJ平台的使用，并由使用者们提供意见。我们将会通过用户数据制造用例测试，并完成一定程度的压力测试。在RC版本中，我们将OJ平台提供于ACM社团，基于新生的加入，用户数量将会扩展到数百，在这之后我们将完成更加全面的压力测试并将用户数据投喂于推荐系统。在版本稳定后，产出Stable版本。  以下是我们总体的设计框架图（图14）。  5f6021a91f0c640856b5202930cdcd0  **图14：项目总体框架图**  我们将传统的单体架构的在线评测系统拆分成了更细粒度的服务单位，如上图中的主要服务、图床服务、推荐系统服务、判题服务以及文件服务。每个服务单位都可以进行独立的开发、测试、部属和交付，每个服务单位都可以做到独立的自治。  **技术路线：**   1. 前端代码编写：(张嘉豪)   使用前端框架React进行网站的从零到一的构建；搭配react-router-dom进行路由管理；Redux-toolkits进行全局通信及状态管理；借鉴Antd组件库辅助开发。编写用户操作手册，覆盖开发的所有功能。下图为前端框架图（图15）。  1683996889652  **图15：前端框架图**   1. 主要服务mgaoj编写：(张健)   使用gin框架搭建web后端，使用mysql进行数据的长期存储，并使用redis对热点数据进行存储。使用rabbitmq与redis完成消息中间件的搭建。使用Auth中间件完成消息验证，Recovery中间件防止崩溃，CORS中间件处理跨域问题，使用降级、熔断、限流的方式增加服务抗压能力。  以下给出后端框架模型图（图16）。  1684150658305  **图16：后端框架模型**  通过MVC设计模式，消息通过消息中间件后通过路由转发到对应的具体实现上，具体实现与数据库、容器、云节点、沙箱进行交互达到提供服务的目的。  使用docker进行快速的服务部署。我们将sql型数据库mysql以及nosql型数据库redis服务端部署于主机上，那之后通过容器部署发布主要服务。同时我们使用容器部署发布判题机以及测试服务，这些容器中都配置了seccomp安全计算模式沙箱以确保运行代码的安全性。除此之外，我们还部署了以心跳监听实现的容器哨兵，他们既可以监听容器的忙碌状态以动态的增删节点，也可以通过查看容器是否过久未响应以此确定容器是否坏死，并可以销毁坏死容器部署新容器取而代之。   1. 图床服务img编写与部署：(张健)   使用gin框架搭建web后端提供上传图片的服务，使用nginx监听一个端口，在该端口提供一个访问指定目录文件的服务，同时通过nginx反向代理完成img服务的部署，达到提供图床的目的。 该服务通过容器部署于云节点上。   1. 测试服务test编写与部署：(张健)   使用gin框架搭建web后端，通过切换受限系统用户限制用户程序的读、写权限，同时启用seccomp安全计算模式限制用户程序的系统调用，以保证用户程序的安全性。测试服务尽可能的减少了与主要服务的耦合，使得该服务可以从主服务中分离出来。通过容器可以快速部署test服务所需的各种语言支持，并确保每个test服务所占有的资源相等。该服务通过容器部署于云节点上，并使用nginx的最少连接优先算法实现负载均衡。除此之外，我们允许judge服务中的消费者通过调用云节点中的test服务来缓解消费压力，以此来完成通过云节点扩增缓解服务压力，即使这会丢失部分judge在消费中的实时消息。  以下给出判题服务的模型图（图17）。  d77a5d19440ac02002b8a3157ff36e9  **图17：判题模型**   1. 判题服务judge编写与部署：(张健)   通过go语言对rabbitmq中的消费者身份进行编写。用户提交的代码通过rabbitmq通知对应的消费者进行消费。消费者消费用户代码时通过redis实时更新消费进展。消费主要包含四个流程，代码检查、编译指令执行、用户切换、调用seccomp安全计算模式运行可执行文件，同时使用容器确保每个消费者所占有的资源数一定。由于需要与主服务进行实时交流，该服务通过容器与主服务部署于同一局域网下。   1. 文件服务file编写与部署：(张健)   使用gin框架搭建web后端提供上传文件的服务，使用nginx监听一个端口，在该端口提供一个访问指定目录文件的服务，同时通过nginx反向代理完成file服务的部署，达到提供文件服务的目的。 该服务通过容器部署于云节点上。  此处通过使用一个映射表来决定存储的文件组应当去往哪个节点。我们会优先选择报告的剩余容量较多的节点，并将文件组压缩打包发送至目标节点，那之后记录目标节点与文件组雪花id至映射表，并记录文件组状态。当存储空间不足时，系统会清理当前最长时间未访问的文件组，直至所有节点存储空间健康。  以下给出文件服务模型图（图18）。  1684150880809  **图18：文件服务模型**   1. 推荐系统部署：(张雷明、张健)   推荐系统将会部署于单独的节点并实现数据持久化。为了实现对各大OJ平台的题目推荐服务，我们需要获取各个OJ平台的提交数据，我们使用网络请求库例如Requests、urllib等用于模拟游览器发送HTTP/HTTPS请求获取网页消息。通过BeautifulSoup、lxml等解析静态HTML页面、XML文档或者JSON数据，使用Mysql存储爬取到的提交数据，使用分布式爬虫框架如同Scrapy、Crawley等用于分布式爬取多个网站或页面。针对反爬虫技术，使用例如IP代理池、Cookie池、User-Agent池等绕过网站的反爬虫机制。同时，我们使用多线程/多进程技术提高爬取效率，对爬虫程序进行用户代理和cookies的随机化，以防止被网站识别为机器人。除此之外，增加去重和断点续传机制，保证数据的完整性和稳定性。  通过爬虫获取的各个OJ平台的提交数据将会“喂养”给推荐系统。在每日用户量稀少的时间段（即3-4点）完成数据的投喂，爬虫将会挑选目标OJ平台中近期活跃的用户提交作为新鲜数据投喂于推荐系统。除此之外，我们计划通过提交阈值去除潜在的脚本提交。  以下给出一份推荐系统相关的服务框架（图19）。  e91d03fabcb99955d700efb68a9fabf  **图19：推荐服务模型**   1. 代码相似度比对算法：(张雷明)   我们对抄袭手段进行了一定的区分，总结如下：  [T1]修改注释、变量名、大小写等无关信息  [T2]代码重排、风格修改等  [T3]增加或删除代码的冗余部分  [T4]对数据结构、循环、局部代码做等价的重写  为了解决这个问题，以《Needle:Detecting Code Plagiarism on Student Submissions》这篇为例，主要过程为用优化的编译器编译代码，这将会把很多手工改过的等价代码优化成同样的二进制代码。此方法可以应对T1、T2和部分T3类型的修改。在这之后，把程序的相似度建模成一个二分图的最大权匹配问题，二分图左边的每个节点u代表P1的一个函数；右边的每个节点代表P2的一个函数，一单位u到v的流量代表了把u中的一条指令“匹配”到v中的一条指令，而流量带来的收益正比于函数的相似度。  大致来说，这个网络流模型近似了程序指令序列的编辑举例，该模型做对T3和T4都有一定程度的抵御（即T3/T4类型的修改不会大幅降低相似度）。  以下是一份简单的网络流模型示意图（图20）。  1684156476888(1)  **图20：网络流模型示意图**   1. 算法可视化模块：(张嘉豪，张健)   通过go语言实现算法可视化平台，并使用gin框架实现在平台上进行数据处理和计算。使用mysql存储图像数据与计算结果，使用React设计和开发用户界面并展示算法的可视化效果，使用p5.js绘制算法可视化。下图为算法可视化模型图（图21）。  1684056266748  **图21：算法可视化模型**   1. 版本管理：(张健)   为了组织和管理代码，使用版本控制系统Git与持续交付工具Gitkraken，确保代码质量和快速迭代。   1. 功能测试：(谢璧蔚)   通过前端开发提供的用户操作手册以及后端开发提供的接口文档进行覆盖式的功能测试。   1. 压力测试：(张健)   通过收集的用户行为，对用户行为进行模拟，创建更多的测试用户进行对接口的调用测试，详细记录服务端的吞吐量、响应时间，最终对系统可承受压力进行评判预估，并以此结果调整熔断、降级服务的临界点。   1. 后期系统维护：(谢璧蔚)   在系统开放后及时接收并处理用户反馈，并总结出对平台的一定整改意见。 | | | | | |
| **四、项目特色与创新**   1. 相对于ZCode而言，该项目拥有提高了平台的交互性，搭建了文章、QA、公告、题解、收藏、比赛讨论区、比赛实时滚榜、好友、标签与文本搜索、私信、小组聊天等相关服务，以此增加社区的信息量。 2. 我们的OJ比赛系统根据用户的需求开发了更多的赛制，包括但不限于小组、个人、随机匹配、标准模式的IOI，OI，ACM机制竞赛以及及时比赛。 3. 为了更好的满足用户需求，我们开发了exam模板，支持oj题目、选择题、填空题、简答题，在这之中填空题与简答题支持正则表达式，除了用户组组长可以发布作业，用户也可以发布考试，为每道题目赋予分值。除此之外，我们复用了竞赛中的标准模式以此增设了标准模式下的机试功能。 4. 为了杜绝投机取巧类现象的发生，区别于传统的OJ判题逻辑，我们增设了更严谨的判题逻辑，即hack功能、重判功能。 5. 为了杜绝竞赛中的代码抄袭，我们增设了代码相似度比对的模块。 6. 为了应对徒增的压力，我们的OJ支持动态节点部署，新增了云节点+负载均衡的方式提供云上的支持。 7. 相对于普遍的OJ而言，该项目将实现题目算法可视化。对于普通的算法可视化而言，使用OJ的新手接触到可视化算法需要一定的了解和时间。而将题目算法可视化，将会省去OJ新人从做题接触算法再到接触可视化算法的过程，而直接从做题接触到算法可视化后学习题目中的算法，这更有利于引导用户进行算法学习。我们将会选择多个模板题进行算法可视化的实现。除此之外，授权用户也可以自主上传题目可视化文件，以此完善OJ中的题目可视化。 8. 同时我们还实现了对联通外站的推荐系统，我们支持使用者在我们实现了联通外站的平台上获取被推荐的题目清单。 | | | | | |
| **五、项目进度安排**  **项目主要分为四个阶段**  **阶段一：**(2022/05~2022/08)   * + 1. 实现判题服务系统。     2. 实现主要服务mgaoj系统。     3. 完成Alpha版本产出并部署。     4. 软件著作权申请。   **阶段二：**(2022/09~2022/10)   1. 完成Beta版本产出并部署。 2. 进行项目组内的服务测试，寻找潜在的漏洞。 3. 收集用户数据，用于完善压力测试以及推荐系统。 4. 收集用户反馈。   **阶段三：**(2022/11~2023/04)   1. 完成RC版本产出并部署。 2. 修复潜在漏洞。 3. 根据用户反馈对功能进行增修。 4. 进行功能测试以及压力测试。 5. 实现文件服务系统并部署。 6. 实现推荐系统并部署。   **阶段四：**(2023/04~2023/05)   1. Stable版本产出并部署。 2. 网站公开。 | | | | | |
| **六、项目经费使用计划**   1. 服务器主机（5899人民币）   （参考酷睿八核I7-10700主频2.9GHz 8G内存 | 1T硬盘 | 标配）   1. 腾讯云服务器（657人民币）   （参考腾讯CVM云服务器基础配置2核2GB 36.5元/月）   1. 相关书籍购买以及论文查阅费用(400元人民币)   综上，本项目的总预算约为6956元人民币。 | | | | | |
| **七、项目预期成果（含中期成果和最终成果）**  **中期成果：**   1. 完成DOJ测试版的搭建工作，进入迭代开发 2. 采集用户的数据以完善推荐系统以及熔断、降级机制 3. 实现文件服务系统并完成部署 4. 实现判题服务系统并完成部署 5. 实现图床服务系统并完成部署   **最终成果：**   1. 实现推荐服务系统并完成部署 2. 实现DOJ稳定版本并完成部署 3. 完成DOJ云节点扩增 4. 申请相关软件著作权1项 | | | | | |
| **指导教师意见：**  **签 名：**  **年 月 日** | | | | | |
| **学院意见：**  **签名盖章：**  **年 月 日** | | | | | |
| **学校意见：**  **签名盖章：**  **年 月 日** | | | | | |