**软件项目计划**

<Testland——代码在线评测平台>

软件项目计划

**（简化版）**

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 19/6/2023 | 1.0 | 制定软件项目计划 | @张奕涵@胡彤@全雨乐@杨菡雪 |
|  |  |  |  |
|  |  |  |  |

软件项目计划

1. **简介**

**目的**

此文档的目的是根据终端用户的需求来概括性阐明“Testland--代码在线评测平台”项目的项目概述、项目组织、项目计划等相关问题。

**范围**

本文档适用于“Testland--在线代码测评平台”，该平台将由上海交通大学电子信息与电气工程学院软件工程专业《互联网产品设计与开发》课程第二组四位同学开发。我们将构建“Testland”平台，支持用户在平台上进行在线代码测评。

“Testland”平台将支持平台上的所有用户查看平台上现有的题目，可选择多种编程语言，将答案上传后进行在线代码测评，包括自动编译和基于测试用例的自动测试。所有平台上的用户可以看到自己的实时排名。系统管理员可以在平台上管理题目及其测试用例，并实时在用户界面进行更新。

**定义、首字母缩写词和缩略语**

1. **Testland**：上海交通大学电子信息与电气工程学院软件工程专业《互联网产品设计与开发》课程第二组四位同学开发的代码在线评测平台。
2. **系统管理员**：上传题目及测试用例以供用户查看及作答的角色。
3. **代码在线测评**：用户上传代码后，平台进行自动编译和基于测试用例的自动测试。
4. **测试用例**：系统管理员给出的一组或多组输入、操作或条件，用于验证软件系统的功能、性能或其他方面的正确性。对于给定的功能或需求，测试用例包括输入数据、预期输出以及预期的系统行为。测试用例旨在覆盖不同的情况和边界条件，以确保系统在各种情况下都能正确运行。

**参考资料**

暂无

2. **项目概述**

**项目的目的、规模和目标**

**目的**

“ Testland ”平台将支持平台上的所有用户查看平台上现有的题目，可选择多种编程语言，将答案上传后进行在线代码测评，包括自动编译和基于测试用例的自动测试。所有平台上的用户可以看到自己的实时排名。系统管理员可以在平台上管理题目及其测试用例，并实时在用户界面进行更新。

**规模**

平台至少支持1000个并发用户正常使用。

**目标**

1. 提供在线代码测评环境：目标是创建一个能够提供在线代码编写、执行和测试的平台，用户可以在该平台上提交自己的代码，并得到即时的评估和反馈。
2. 多语言支持：目标是支持多种编程语言，以满足不同用户的需求。用户可以选择他们熟悉的编程语言进行编写和测试。
3. 提供丰富的测试工具和库：目标是提供一系列测试工具和库，以帮助用户编写和运行各种测试用例，评估代码的正确性、性能和可靠性。
4. 实时评估和反馈：目标是在用户提交代码后，快速进行代码编译、执行和评估，并向用户提供实时的反馈和评分。用户可以根据评估结果进行代码改进和调试。
5. 用户管理和权限控制：目标是提供用户管理和权限控制功能，允许用户注册、登录和查看他们的代码项目。管理员可以发布与管理平台上的题目，发布测试用例等。
6. 可扩展性和性能：目标是设计和构建一个可扩展和高性能的平台，能够处理大量用户同时提交的代码，并保持系统的稳定性和响应速度。
7. 安全和隐私保护：目标是确保代码测评平台的安全性，防止恶意代码执行、数据泄露和其他安全威胁。用户的代码和数据应受到保护，并符合隐私法规和最佳实践。
8. 可视化和统计功能：目标是提供可视化的界面和统计功能，让用户能够查看他们的代码评估结果、进度和历史记录，以便进行分析和改进。

**假设与约束**

1. **预算**

项目经费两千元，用于华为云平台的开发与部署

2. **人员**

四人团队开发

3. **设备**

* 操作系统：不同操作系统（如Windows、MacOS、Linux）可能对开发环境和部署环境有不同的要求和限制。需要确保我们的代码测评平台能够在不同操作系统上正常运行和提供一致的用户体验。
* 浏览器兼容性：不同的浏览器（如Chrome、Firefox、Safari、Edge等）在支持HTML、CSS和JavaScript方面可能存在差异。需要确保我们的代码测评平台在主流浏览器上都能正常运行，并提供一致的功能和用户界面。
* 设备分辨率：用户可能使用不同分辨率的设备来访问您的代码测评平台。需要确保平台能够适应不同的屏幕尺寸和分辨率，以提供良好的用户体验。
* 网络连接：代码测评平台可能需要与远程代码执行环境进行交互，因此对网络连接的要求较高。需要确保平台在不稳定或低速网络环境下仍能正常运行，并提供适当的错误处理和提示。
* 计算资源：在进行代码编译、执行和评估时，可能需要消耗大量的计算资源。需要确保能够有效管理和分配计算资源，以避免性能问题或系统崩溃。
* 安全性：在线代码测评平台需要处理用户提交的代码，因此安全性是一个重要考虑因素。需要采取适当的安全措施，防止恶意代码执行、数据泄露或其他安全威胁。

4. **时间表**

开发时间为2023-06-19至2023-09-10，必须在规定期限内交付项目，完成对应时段的迭代任务。

|  |  |  |
| --- | --- | --- |
| **迭代** | **开始时间** | **结束时间** |
| 界面原型迭代 | 06-19 | 06-26 |
| 技术原型迭代 | 06-27 | 07-05 |
| 产品完善和发布 | 07-06 | 07-14 |
| 产品版本更新 | 07-17 | 09-10 |

**项目的可交付成果**

|  |  |
| --- | --- |
| **成果** | **描述** |
| 软件项目计划 | 项目计划文档，包括项目的时间表、资源分配和里程碑安排等，用于指导整个项目的开发过程。 |
| 迭代计划 | 迭代计划文档，详细描述每个迭代的目标、任务和时间表，用于指导项目的迭代开发和交付。 |
| 软件需求规约 | 包含详细的需求描述、用例和功能规格说明的文档，明确系统的功能、性能和用户界面等方面的要求。 |
| 软件架构文档 | 描述系统的整体架构、模块划分和组件间的关系，包括系统流程图、数据库设计和接口定义等内容。 |
| 测试报告 | 包含测试计划、测试用例、测试数据和测试结果等的文档，记录测试过程和测试结果，评估系统的质量和稳定性。 |
| 迭代评估报告 | 对每个迭代的开发过程和成果进行评估和总结的报告，包括问题和挑战的分析，为下一迭代的改进提供反馈。 |
| 项目总结报告 | 对整个项目开发过程和成果进行总结和回顾的报告，包括项目目标的达成情况、经验教训和未来改进的建议等。 |
| 代码 | 包含项目的源代码、配置文件和依赖库等的代码库，用于实现系统的各项功能和特性。 |

3. **项目组织**

**组织结构**

1. 矩阵型组织结构：团队成员同时属于多个功能组，都参与设计、开发、测试流程，这种结构允许更好地协调资源和专业技能。
2. 平台型组织结构：团队使用华为云平台，基于平台开发项目。
3. 因人数限制，整体结构较为扁平，成员职能较均衡。

**主要角色和分工**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 设计 | 开发 | 测试 | 项目管理 |
| @张奕涵 |  |  |  |  |
| @全雨乐 |  |  |  |  |
| @杨菡雪 |  |  |  |  |
| @胡彤 |  |  |  |  |

4. **项目计划**

**风险分析**

|  |  |  |  |
| --- | --- | --- | --- |
| **优先级** | **风险名称** | **风险描述** | **风险缓解措施** |
| 1 | 技术风险 | 1. **技术复杂性**: 项目所涉及的技术可能非常复杂，团队可能面临技术理解和应用方面的困难。 2. **技术选型错误**: 不正确或不合适的技术选择可能导致项目延期、性能问题或无法满足业务需求。 3. **技术依赖性**: 项目可能依赖于特定的技术或第三方库，如果这些依赖关系存在问题，可能会影响项目的进展和成功。 | 1. **技术调研**: 在项目开始之前进行充分的技术调研，评估各种技术选项并选择最适合项目需求的技术。 2. **技术评审**: 在项目开始之前或在关键节点上进行技术评审，确保所选技术能够满足项目需求，并与其他组件/系统集成良好。 |
| 2 | 进度风险 | 1. **错误的时间估计**: 在项目计划阶段对任务的时间估计不准确，可能导致进度延误和资源分配不当。 2. **范围蔓延**: 对项目范围的控制不严格，可能导致需求增加、功能扩展，进而影响进度和交付时间。 | 1. **详细的项目计划**: 建立一个详细的项目计划，包括明确的任务、里程碑、时间估计和资源分配。 2. **迭代开发**: 将大型项目拆分为小的可交付部分，实现快速迭代和反馈。 3. **优先级管理**: 确定任务的优先级，并明确沟通给团队成员，使团队能够有序地处理任务。 |
| 3 | 安全风险 | 1. **数据泄露和隐私问题**：未经授权访问、存储或传输敏感数据可能导致数据泄露和隐私问题。 2. **不安全的第三方组件**：使用不安全或过时的第三方组件可能存在漏洞，使整个系统受到威胁。 | 1. **数据加密：**实施适当的数据加密、访问控制和身份验证机制，确保只有授权人员才能访问敏感数据。遵循最佳实践和法规要求，例如GDPR。 2. 定期更新和修复第三方组件。 |
| 4 | 需求风险 | 1. **需求优先级不明确**: 如果需求的优先级不明确，团队可能会在实施过程中遇到困惑，并难以做出正确的决策。 2. **需求不完整或不清晰**: 不完整或不清晰的需求可能导致开发团队产生误解，造成实现与预期之间的差距。 | 1. **需求管理**: 建立一个有效的需求管理流程，包括需求收集、分析、验证和确认，以确保需求的准确性和一致性。 2. **敏捷开发方法**: 采用敏捷开发方法，如Scrum或Kanban，以便在需求变化时能够快速适应并进行迭代开发。 3. **优先级管理**: 确定和管理需求的优先级，以便在资源有限的情况下合理分配和规划。 |

**方法和工具**

**开发方法**

* 敏捷开发方法（如Scrum）：采用迭代、增量开发的方式，快速响应变化并提供可交付的功能。

**开发工具**

1. 建模工具

* UMLet

1. IDE

* 前端：Visual Studio Code 和 WebStorm
* 后端：IntelliJ IDEA

1. 测试工具

* 单元测试：JUnit
* 自动化测试：Selenium

1. 版本管理工具

* 使用 Git 进行版本控制和团队协作

**开发计划**



**点击图片可查看完整表格**

**质量保证计划**



**点击图片可查看完整表格**

**项目沟通计划**

**项目组内部沟通安排**

**每日立会时间：**15:30

**项目日常沟通：**微信群即时进行

**项目协作沟通：**飞书云文档

**项目组向助教和老师汇报**

**日常：**助教参与每日立会。

**文档：**每次迭代的各类文档根据老师要求上传canvas平台，并在codearts仓库存档。

**代码：**使用git对codearts仓库进行各类操作。

5. **附录**

暂无