

Process支持文件说明文档

1. 需求明晰

1. 首先，在课程初始阶段，进行项目的选题

- 由各位组员在大致浏览过项目的代码之后，各自尝试进行八个项目的构建，于规定时间进行经验以及构建体验的分享，并共同分析进行选题的确定
- 项目预选定的讨论：



- 项目配置部署过程中的讨论



- 最终项目的选定：



2. 之后，根据助教发布的 Requirements 来初步确定项目的需求，并且进行前景与范围的讨论，同时完成前后端的分工

- Vision & Scope:

- 有详细的分工文档：

Vision and Scope分工

业务需求Business Requirements

- 1 应用背景与业务机遇Background, Business Opportunity, and Customer Needs
- 2 业务目标与成功标准business objectives and success metrics
- 2 业务风险Business Risks

项目前景Product Vision

- 3 前景概述Vision Statement
Concept of what the product might eventually become
- - 3 主要特性 Major Features
Identifies business benefits of the system will provide
 - 3 假设与依赖Assumptions and Dependencies

项目范围Project Scope

- 4 版本范围Scope of Initial and Subsequent Releases
 - 4 限制与排除limitations and exclusions
 - 5 Context Diagram
 - 5 Feature Roadmap
 - Use Case Diagram (可选)
 - System Events (可选) |
- 专门的公共文档管理：

逸凡

卢逸凡

文档直接在语雀上：编辑
<https://www.yuque.com/g/luyf12/qhh12s/collaborator/join?token=xJsQ097hf9P8OOli#> 邀你加入知识库「2022软件需求工程」(可编辑)

- 前后端分工确定

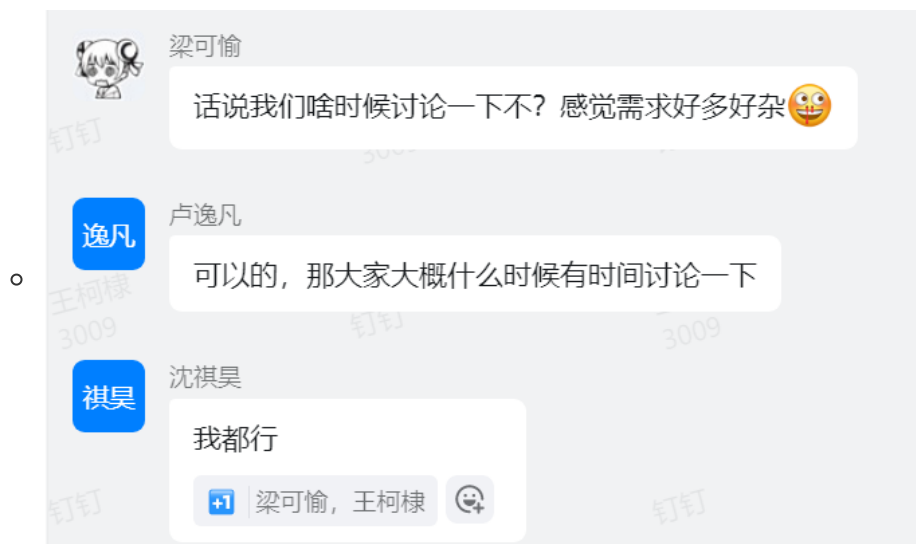
逸凡

卢逸凡

卢逸凡:
前后端具体怎么分配大家有想法吗? 我觉得可以前2人后3人, 不知道这样合不合适

如果大家都没意见的话就按这个选一下吧

3. 准备开发之前, 进行了专门的线下会议, 进一步细化各需求的细节及其实现方式



4. 在需求访谈会之前，根据之前线下会议对于各需求的设想，以及将其扩展详实时所遇见的困惑，以书面文档的方式罗列了我们所要提问的各种问题，设计了访谈会的议程，并对于访谈会的结果进行了专门的记录

- 会议进程的安排：

需求访谈会会议议程

议程安排

概述

时间：2022.10.31

时间	主要内容
14:30-14:35	分工就位，团队自我介绍，概括开发小组对“开源平台数据可视化”的理解
14:35-15:00	访谈时间（分为8个阶段：1.开场 2.建立客户/用户情况表；3.评估问题；4.对了解的情况做个简要回顾；5.让客户评估解决方案；6.让客户对开发系统的可靠性、性能、技术支持进行评估；7.总结）

人员安排

- 人员
 - 1名记录员
 - 1名主提问人，其他提问人可按需补充
 - 可能需要的资料
 - 软件需求工程访谈活动评价表
 - 需求分析图
 - 模块初步划分示意图
- 访谈内容的确定（部分）：

五、客户评估解决方案

需要实现的需求：

- 1、在原有功能基础之上增强数据分析功能，可以统计显示一段时间内不同贡献者的代码提交、指出项目的核心贡献者。（目前工作重心）
- 2、展示被分析项目贡献者的活跃情况和社区发展速度
- 3、计算项目的 stargazer, committer, issue 人数的 company 信息，并且用图表形式直观展示项目开发来自的组织、公司。（目前工作重心）
- 4、增加对项目issue内容的分析，例如分析issue数量按照更新时间的变化、特征关键词的提取、添加用户自定义条件分析等。
- 5、增加横向比较多个项目的功能，支持图表和数据的时间范围选择、缩放功能、排序功能
- 6、实现针对数据源的缓存、优化从 GitHub 上获取信息的方式，在 GitHub 接口不可用时仍能提供项目历史数据访问。同时细化数据过滤，使得对缺失的数据有所补全，提高用户使用的友好程度。

六、客户对待开发系统进行评估

问题：

- 1、对系统可靠性方面的要求？比如issue特征关键词提取的准确率以及响应率、从 GitHub 上获取信息的优化效果。
- 2、系统的性能要求？比如在 GitHub 接口不可用时仍能提供项目历史数据访问时的访问速度。

七、其他需求

- 1、存在特殊的许可要求吗？（比如public的项目都可以被import）
- 2、还有其他需求吗？
- 3、对系统的UI界面是否还有非功能性方面的要求或设想？

需求访谈会结果总结（部分）：

Q：如何在大型项目中体现一段时间内的多个贡献者？列表形式还是统计图的形式？

A：最好是折线图，横轴的标注要清晰一些，如果是时间轴的话，要标注年份。目前只是显示了最近1周的情况，最好拉长时间轴到几个月或者一年。另外commit和issue都要统一成frequency。另外，我们希望用户可以被分类成核心/非核心用户。区分的方式是根据用户提交代码量、通过打标签的方式来标注。

Q：我们目前是根据commit和issue的频率来判断社区的发展速度。其他还有什么评价指标吗？

A：可以。除了不同贡献者的commit/issue，还可以加上pull request信息。

Q：特征关键词要如何获取，是通过标签还是从文本内容中提取？

A：如果有标签就可以用标签，但是没有的话，issue有description，还有comment可以进行判断

Q：设计话题的分类，是根据ppt上的分类标准还是根据话题的内容我们提取关键词来分类？

A：所有的issue和pull request都分成designed和non-designed，然后小类里面可以再看具体某一话题，可以用词云图的方式展示。

Q：pull request这部分想展示什么信息？

A：首先是时间维度看设计的占比。随着时间变化的曲线，一条表示designed相关的pull request，一条表示总体的；两条线可以对比看出设计相关的pull request的比例。其次是在topic维度，再去看design方面有哪些信息、话题。

Q：两个不同的项目独立的统计图表放在同一界面，还是同一图表不同的数据链？

A：都可以，只要在ui上展现对比思路就行。

Q：本地的数据需要实时更新吗？

A：本地的pytorch初步分析只要展示就可以，但是要允许用户导入新的repository，异步操作，然后给用户一个信息何时可以处理完非本地的数据。允许非本地数据的处理数据花费时间更久。

Q：对系统可靠性方面的要求？比如issue特征关键词提取的准确率以及响应率、从 GitHub 上获取信息的优化效

5. 至此，我们已经获取了比较详实的需求，并将其以《需求规格说明书》的方式进行书面化，以便日后的参考

需求规格说明书展示（部分）

<<
需求规格说明书

1. 引言
 - 1.1 编写目的
 - 1.2 文档约定
 - 1.3 预期的读者和阅读建议
 - 1.4 项目范围
 - 1.5 参考文献与资料
2. 总体描述
 - 2.1 产品前景
 - 2.2 产品特性
 - 2.2.1 查看项目收藏者信息
 - 2.2.2 查看项目贡献者活跃度
 - 2.2.3 查看项目提交issue用户...
 - 2.2.4 查看项目设计讨论信息
 - 2.2.5 横向比较项目
 - 2.2.6 更新项目信息
 - 2.3 用户类别与特征
 - 2.4 开发、测试和运行环境
 - 2.5 设计和实现上的约束
 - 2.5.1 数据存储
 - 2.5.2 网络服务吞吐
 - 2.5.3 数据安全

需求规格说明书

组名: odometer

组长: 3200102642-卢逸凡

组员: 3200104866-赵伊蕾 3200104734-沈祺昊 3200102547-梁可愉 3200105119-王柯棣

1. 引言

1.1 编写目的

本软件需求规格说明书对软件开发团队Double-C Analytics Dashboard的改进做了细致全面的用户需求分析, 针对开源项目负责人等用户群体, 清晰地描述了系统的功能性需求和非功能性需求。

本软件需求规格说明书旨在明确所要开发的软件开发团队数据分析平台应具有的系统概貌、功能要求、性能分析、运行要求等, 并尽可能完整地描述软件开发团队Double-C平台预期的外部行为和用户可视化行为, 使系统分析人员及软件开发人员能清楚地了解不同用户群体的需求, 并在此基础上进一步提出概要设计说明书和完成后续设计与开发工作。

1.2 文档约定

本软件需求规格说明书没有使用特殊的排版约定。

1.3 预期的读者和阅读建议

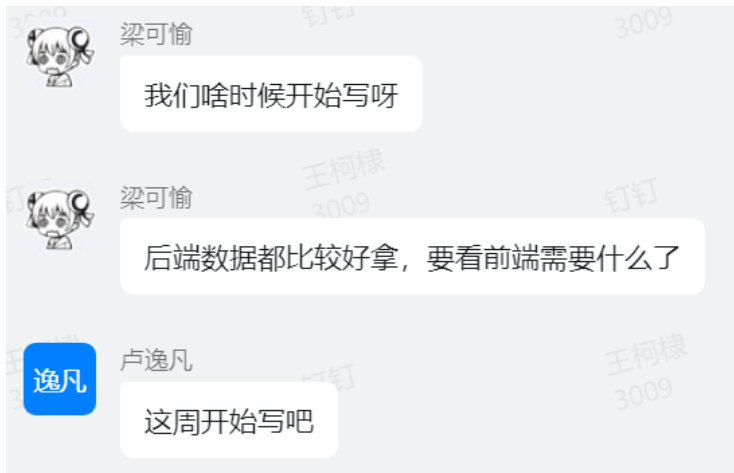
本软件需求规格说明书的主要内容分为总体描述、功能需求、数据流图、外部接口需求、非功能性需求、数据字典以及附录这些部分。总体描述部分对产品的基本信息、前景、用户类及特征、设计和实现上的约束、假设和依赖、开发测试和运行环境等进行了详细的介绍。功能需求部分根据不同的用户类型对需求做了详细的分析, 并给出了用例的时序图, 是本软件需求规格说明书的主要部分。数据流图部分对每个子系统的逻辑流向做了图形化的表

6. 至此, 本项目开发的需求已经初步明晰。我们将以《需求规格说明书》为基准, 初步进行软件开发的代码撰写, 并在开发的过程中实时进行反馈, 完成需求的迭代。

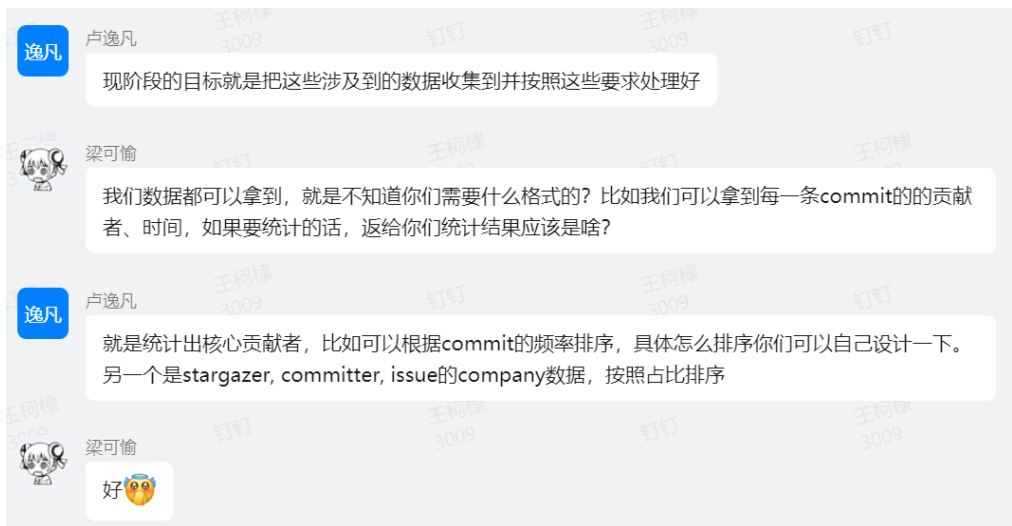
2. 需求迭代

1. 需求明晰过后, 我们便开始代码的写作。在总体时间安排以及任务安排确定过后, 由前后端两组组内独立分工推进, 并且在推进的过程中及时提交成果, 由对方使用过后提出改进意见, 完成需求的不断迭代

- 开始开发:



- 需求协调:



- 开发过程中, 通过详细的文档实时保持沟通, 不断推进需求的迭代



梁可愉

@卢逸凡 @沈祺昊 我们大概分了一下后端的锅，文档如下：

<https://yuque.zju.edu.cn/docs/share/ad104b2c-5843-448f-bffb-ea6675537dc1?#> 《后端分工》，里面涉及到一些我们还不太懂的问题，要不你们看看大概要咋搞

- Q1: 对于“社区发展速度”一项，时间间隔应该是多少？
 - 是否要给予不同的选项，如一周、一个月、一年？
 - 还是全部数据都进行返回（如<commit,time>数组），由前端自己统计？
- Q2: 对于“项目横向对比”，需要对比哪些数据？
 - 目前我们设想的是：比较commit数随时间发展的变化，stargazer数随时间发展的变化，issue数随时间发展的变化；任意两个项目可以进行对比；**有什么需要更改的吗？**
 - 此处是否要支持支持图表和数据的时间范围选择、缩放功能、排序功能？实现范围选择、缩放等较为灵活的功能，可能后端只能返回整块数据，由前端统计
- Q3: 还有什么数据是需要的吗？

富文本



梁可愉

上面那些问题涉及到后端数据库设计，所以需要明确一下再动工~如果可以的话在钉群大概说一下或者文档里回答一下？对接口有啥需要的也可以直接在文档里注明，麻烦了！

接口	参数	返回值	调用形式
commit数随时间变化	仓库		

富文本

1条回复

- 同时，在开发的过程中维护了开发文档，便于开发

开发文档

1. company信息(pytorch) 赵伊蕾

导入项目的同时，调用octokit API对company数据进行收集统计。目前获取数据的部分嫁接在了已有的功能之上：
点击“import”的按钮。

- 项目目前的问题：
1. 我感觉octokit接口不是很稳，很有可能数据收着收着就挂了，建议多试几次。可以自己设置导入的数据量大小，通过改的上限(总page数)、每页获取几个(per_page数)。建议本地测试的话，数据规模可以小一点，每种类型300个左右。

```
const RepoGetStargazers = async (owner, name) => {
  var result = [];
  var company = new Map();
  var company_name = "";

  // for (var i = 1; i <= 800; i++) {
  for (var i = 1; i <= 3; i++) {
    console.log("STARGAZERS");
    const NextRepoMessage = await octokit.request(
      "GET /repos/{owner}/{repo}/stargazers",
      {
        owner: owner,
        repo: name,
        per_page: 100,
        page: i,
      }
    );
```

2. 因为每一个用户的company信息都要调用两次API，速度非常慢。我计了一下时，每100个issues/stargazers用户的company信息统计需要花1分15s左右，所以导入这些数据非常非常花费时间...
3. 关于committers的company信息。由于通过commit去统计committer的话，数据量实在太大大太大了，pytorch有9w+的commits，信息密度很小。所以那部分我就用了contributor的信息来代替，问题可能不太大。不过调数据的时候发现，明明应该有2000+的contributor，但是接口只能获取到346个，可能接口只提供了最核心的成员...?
4. 如果点击项目导入并不能创建company数据表的话，自己手动建一张吧(没有测试会不会出现这个bug，目测应该会自动生成的...

2. 最终，我们于12月7日完成了UC1-UC11的全部内容，完成了需求的第一次迭代

- o 开发计划展示：

优先级	前端	后端
已实现	贡献者的活跃情况	~
已实现	社区的发展速度	stargazer, commit, issue数随时间 (week, month, year) 变化
已实现	company 信息	stargazer, commit, issue的company数据
0 ddl:12.4	横向对比两个项目	~
0 ddl:12.4	pull request设计相关统计图	统计pull request数据，统计单位为月，分为designed和non-designed两种
1 ddl:12.7	pull request关键词词云	统计pull request designed数据中设计讨论关键词占比，按照关键词频率降序排序
2 ddl:12.7	pull request话题相关统计图	统计pull request中频率前五的关键词，统计每个关键词每月的pull request数量
3 ddl:12.9	修改commit frequency, issue frequency统计图	commit, issue数根据用户提交代码量分别统计核心用户和非核心用户
4 ddl:12.9	pull request统计图加入周为单位	pull request统计以周为单位的数据
5 另行讨论	修改issue统计图，加入issue词云	对issue进行类似pull request的统计

- o 提交历史（部分）

finish core users

 koryyy • 7 days ago

#pr part, update router

 Elaine Zhao • Dec 5, 2022

#pr part, finish pr monthly frequen...

 Elaine Zhao • Dec 5, 2022

#pr part, update import function

 Elaine Zhao • Dec 5, 2022

pr part, update models

 Elaine Zhao • Dec 5, 2022

Delete design.js

 Elaine Zhao • Dec 5, 2022

fix design define bug

 zju-wkd • Dec 5, 2022

Merge branch 'main' of https://git...

 zju-wkd • Dec 3, 2022

add design parts

 zju-wkd • Dec 3, 2022

update #company part, fix router a...

 Elaine Zhao • Dec 3, 2022

update client v1.0.1

3. 在此之后，根据项目表现结果进行评估，进一步进行了接口的优化，并且添加按周统计设计话题分析，完成了需求的第二次迭代

- 第二轮迭代：

会议议程

一些问题

- 核心用户数据爬的慢，改成分开爬？
- design, topic数据
 - 格式改成一个项目一个record，主键为owner和name
 - 爬数据中断导致数据重复
- design相关有点少？
- 其他内容先这样，先集中搞milestone4

■ Milestone4

Milestone 4提交 2 Process Files: 过程的支持文件ZIP + 5分钟视频说明支持文件

会议记录

后端细节调整

1. 每个表的负责人把pytorch完整的数据爬下来后保存数据表，传到群文件里；
2. designfrequencies表解决一下数据重复的问题；
3. topicfrequencies表把空的项补上0；
4. topics加一点关键词，尽可能每个类型的数据都不为空

3. 需求映射

- 在开发的过程中，我们时时刻刻都“以需求为本”进行开发
- 首先，在开发伊始，我们从助教提供的需求PPT出发，不断明晰产品需求，确定我们到底要做什么



- 在开发过程中，依托于需求，逐渐推进开发，并且在开发过程中不断迭代，更新User Case，最终完成产品
 - 最初只有六个User Case

3. 功能需求

- 3.1 查看pytorch项目收藏者信息
- 3.2 查看pytorch项目贡献者活跃度
- 3.3 查看pytorch项目提交issue用...
- 3.4 查看项目设计讨论信息
- 3.5 横向比较项目
- 3.6 更新项目信息

4. 外部接口需求

- 在需求访谈会之后，更新对于需求的理解，接口设计也发生了变化



- 随着开发的推进，需要完成的需求逐渐增加，User Case上涨到了十一个，开发计划也随之变动，增加了最初的User Case以外的相关内容

优先级	前端	后端
已实现	贡献者的活跃情况	~
已实现	社区的发展速度	stargazer, commit, issue数随时间 (week, month, year) 变化
已实现	company 信息	stargazer, commit, issue的company数据
0 ddl:12.4	横向对比两个项目	~
0 ddl:12.4	pull request设计相关统计图	统计pull request数据, 统计单位为月, 分为designed和non-designed两种
1 ddl:12.7	pull request关键词词云	统计pull request designed数据中设计讨论关键词占比, 按照关键词频率降序排序
2 ddl:12.7	pull request话题相关统计图	统计pull request中频率前五的关键词, 统计每个关键词每月的pull request数量
3 ddl:12.9	修改commit frequency, issue frequency统计图	commit, issue数根据用户提交代码量分别统计核心用户和非核心用户
4 ddl:12.9	pull request统计图加入周为单位	pull request统计以周为单位的数据
5 另行讨论	修改issue统计图, 加入issue词云	对issue进行类似pull request的统计

- 到最终完成开发, 提供的接口依然是依托于需求而设计

```

router.route("/import").post(GetMessage);
router.route("/login").post(CheckUser);
router.route("/register").post(CreateUser);
router.route("/search").post(SearchRepoName);
router.route("/dashboard").post(GetDashboard);
router.route("/delete").post(DeleteRepo);
// ===== add design =====
router.route("/getDesign").post(getDesignFrequency);
router.route("/getTopic").post(getTopic);
router.route("/getTopicFrequency").post(getTopicFrequency);
// ===== add week =====
router.route("/getWeekDesign").post(getWeekDesignFreq);
router.route("/getWeekTopicFrequency").post(getWeekTopicFreq);
// ===== add company=====
router.route("/committerCompany").post(getCommitterData);
router.route("/stargazerCompany").post(getStargazerData);
router.route("/issueCompany").post(getIssueData);
// =====add frequency =====
router.route("/getFrequency").post(GetFrequency);
// ===== add core_users =====
router.route("/GetCoreUsers").post(GetCoreUsers)
module.exports = router;

```

4. 产品开发（这块好像没什么好说的，就是开发过程文档.....）

- 9.27~10.7: 项目set up, 分析项目优缺点, **Milestone 1**
- 10.8~10.29: 根据发布的需求确定vision & scope, 完成**Milestone 2**
- 10.30: 需求访谈会, 确定具体需求, 进行前后端分工

- 10.31~11.11: 完成需求规格说明书（**Milestone3**）。通过线上会议确定用例优先级，根据优先级确定用例驱动的开发时间线
- 11.12~11.30: 完成UC1 ~ UC4, 包括社区发展速度和公司数据分析
- 12.1~12.7: 完成UC5 ~ UC11, 包括项目设计话题分析和按代码量统计核心用户，修改了之前部分功能的bug，**完成第一次迭代**
- 12.12: 中期展示
- 12.13~ 12.16: 优化接口，添加按周统计设计话题分析，**完成第二次迭代**，完成**Milestone 4**

5. 会议关键

- 对于每次会议，会提前确认全员的时间，保证每人都能及时得到会议消息



- 对于每次会议，有提前准备好的会议议程，确保各位参会时能够有最好的参会效果

○

逸凡

卢逸凡

今晚的议程:

<https://www.yuque.com/g/luyf12/qhh12s/usqz60y97u9wikus/collaborator/join?token=hub99U70gYEdeBgu#> 邀请你共同编辑文档《开发计划》

12.12会议

会议议程

一些问题

- 核心用户数据爬的慢，改成分开爬？
- design, topic数据
 - 格式改成一个项目一个record，主键为owner和name
 - 爬数据中断导致数据重复
- design相关有点少？
- 其他内容先这样，先集中搞milestone4

Milestone4

Milestone 4提交 2 Process Files: 过程的支持文件ZIP + 5分钟视频说明支持文件

一些问题

- 数据尽量不与1.0版本的数据混用，存在新的表里，用1.0版本中的 `_id` 作为主键
- 接口的参数需要修改为以下格式

▼ backend/controllers/ JavaScript 复制代码

```
1 ▶ const GetDashboard = async (req, res) => {...};
```


- company需要计算占比


▼


JavaScript 复制代码


```
1 [{ company: 'Facebook', num: 88, proportion: 1.9}]
```


- 会议过后，进行会议记录，确保每位组员在事后能够更好地消化会议精神


 【Odometer】第一次会议.pdf


 【Odometer】第二次会议.pdf

 【Odometer】第六次会议.pdf

◦  【Odometer】第七次会议.pdf

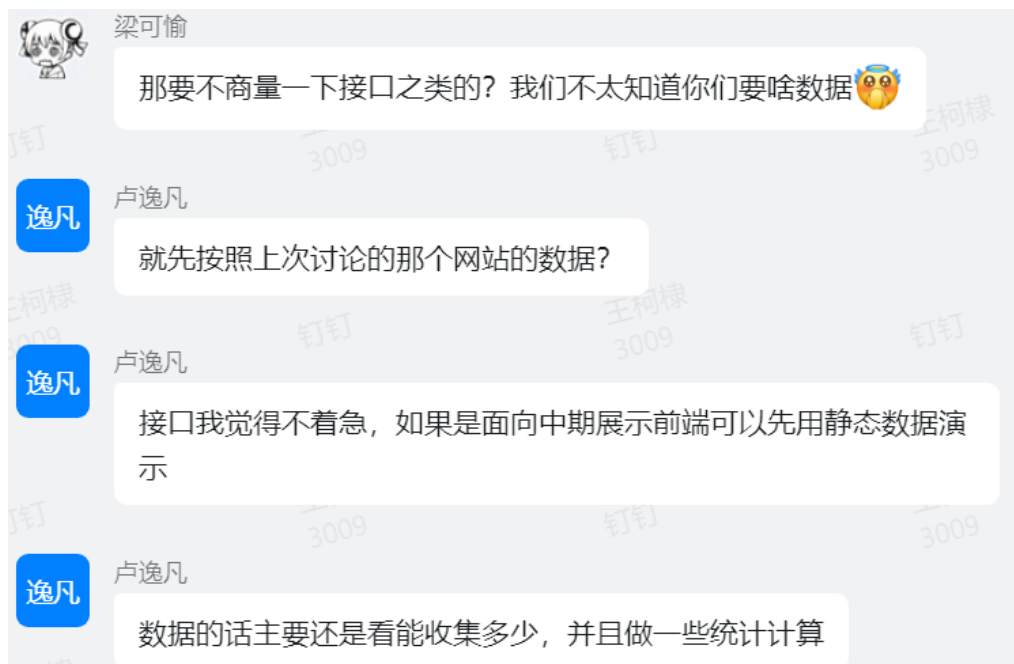
 【Odometer】第三次会议.pdf

 【Odometer】第四次会议.pdf

 【Odometer】第五次会议.pdf

- 每次会议根据当前的进度以及所需完成的MileStone进行设置

- 第一次会议：MileStone 1 - 项目的选定
 - 讨论过程
 - 目前项目3能部分运行，项目2和项目4能成功运行
 - ○ 项目2的UI设计需要进行重做，工作量较大；项目4可拓展性好
 - 决定最终使用项目
 - 选择项目4: **DoubleC Analytics Dashboard**作为基础项目并进行增量开发
- 第二次会议：MileStone 2 - Vision & Scope
 - **项目目标**：在原有的功能上，增加Requirements中要求内容
 - ○ **原有功能的改动**：更改页面设计，对已实现的comitter分析接口进行改动
 - **可视化**：重点对项目数据在时间上的变化进行分析
 - **范围**：分析部分只针对Pytorch项目
- 第三次会议：需求分析&需求访谈会
 - 我们将项目需求大致分为3个目标实现
 - 对Pytorch项目的stargazer、committer、issue 人数 company 信息的增强数据展示
 - 对项目讨论话题进行展示分析，其中话题来自于pull request、issue内容。
 - 横向对比多个项目的多项指标
 - 开发计划
 - 针对上述3个项目需求，我们制定了开发的顺序和优先级。我们将首先完成stargazer、committer、issue人数的company这三个信息的数据可视化工作，然后再做多个项目的横向对比，最后完成第二部分话题展示需求。
 - 后端分工
 - 每人领了项目目标1的一块数据内容，研究github接口，以及数据如何获取。
- 第四次会议：srs & midterm & 开发进度讨论



- 第五次会议：项目需求细节分析与前后端接口规范
 - 社区发展速度分析
 - 时间间隔应给予不同的选项，分别是：一周，一个月，一年
 - 后端直接把统计好的数据返回
 - ● 项目横向对比
 - 需要分析commit数随时间发展的变化，stargazer数随时间发展的变化，issue数随时间分析的变化
 - 支持图表和数据的时间范围选择、缩放功能
- 第六次会议：项目进度跟踪

数据格式

```
JavaScript 复制代码
1 var pr_design_frequency = [{ date: "2022-12", designed: 200, non_designed: 100 }];
2 var pr_topic = [{ topic: "improvements", num: 1000 }];
3 var pr_topic_frequency = [
4   {
5     date: "2022-12",
6     improvements: 200,
7     testing: 100,
8     topic3: 50,
9     topic4: 20,
10    topic5: 10,
11  },
12 ];
13 var commits_design_frequency = [{ date: "2022-12", core: 200, non_core: 100 }];
14
```

以周为单位时, date为该周的第一天

- 第七次会议: 项目进度跟踪 & MileStone 4

会议议程

一些问题

- 核心用户数据爬的慢, 改成分开爬?
- design, topic数据
 - 格式改成一个项目一个record, 主键为owner和name
 - 爬数据中断导致数据重复
- design相关有点少?
- 其他内容先这样, 先集中搞milestone4

Milestone4

Milestone 4提交 2 Process Files: 过程的支持文件ZIP + 5分钟视频说明支持文件