小组总结

技术选型

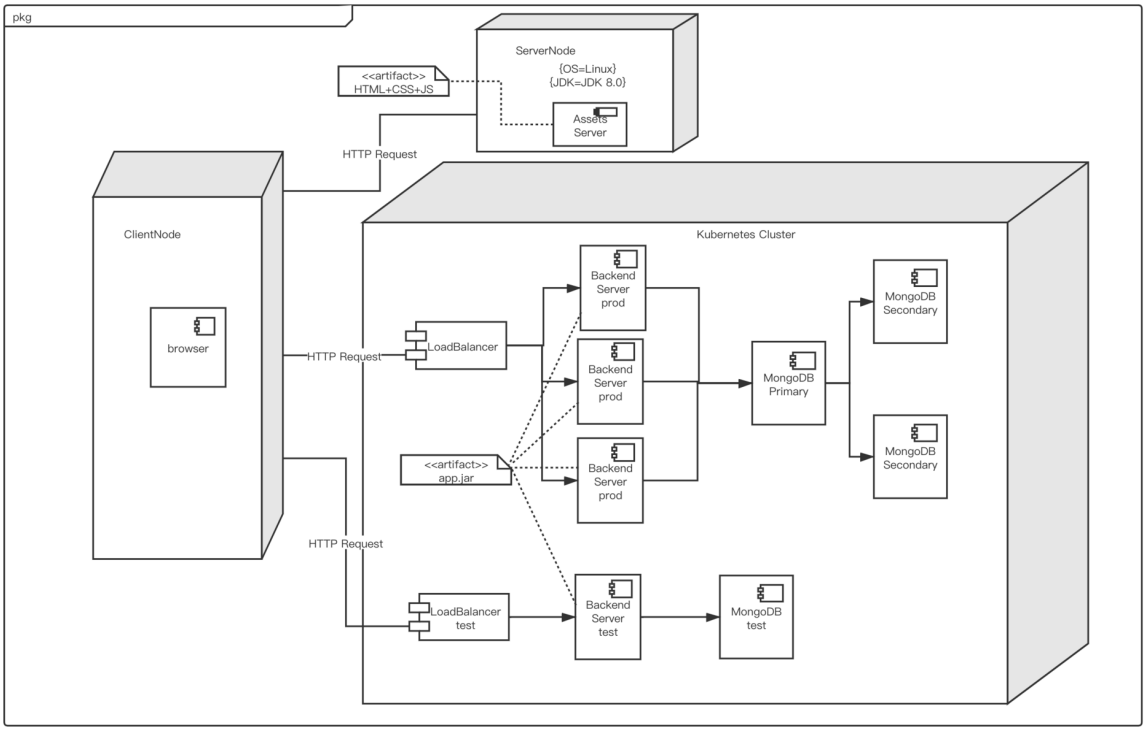
前端技术选型

在迭代二中，为了进一步满足质量要求，我们对前端整体进行了重构，采用服务端渲染来提高首屏渲染性能以及进行SEO，同时给予开发者对整个系统的页面的更细粒度的控制；缺陷是开发的复杂度有所提高，同时前端的并发性能下降，这一点是需要进一步考虑的。

随着前端复杂度的提高，构建过程越来越复杂，原先1核2G的服务器已经不能满足CI/CD的需求。在此过程中，我们陆续尝试了Jenkins和gitlab-ci，但都以失败告终。因此，最后我们不得不放弃CI，改为在本地进行构建。考虑到小组成员使用不同的操作系统，为了降低手动构建带来的复杂度，我们编写了跨平台的npm脚本(npm run release)，通过统一的构建方法来简化构建和发布过程。

部署环境选型

使用scrapy爬虫爬取了IEEE网站中170k条论文数据。数据量的增加使得原本不明显的性能问题显现出来。原本基于1核2G的云服务器已经不能够满足搜索、画像等功能的计算需求。同时，由于内存不足导致的服务不可用情况也越来越频繁。基于以上情况，我们选择将整体部署环境进行迁移。这里我们使用了google计算云服务中的Kubernetes容器集群服务。



这里我们构建了两套独立的环境，分别用于生产环境和测试环境。生产环境中，通过负载均衡器来将请求分配到三个后端，同时，将MongoDB数据库部署为三个节点的ReplicaSet，保证服务的高可用。使用Kubernetes的Deployment资源，来实现后端服务的滚动更新，避免出现服务不可用的情况。

小组合作的流程（开发流程）

与迭代一相似，我们小组为克服远程协作开发的困难，保证隔天开一次会，在每次开会中交流并解决最近开发遇到的问题，提出未来几天的短期开发计划。会议结束后就可以按照开发计划进行开发。同时我们利用石墨文档作为文档协作仓库，共同维护了接口文档，会议记录等一系列文档作为开发的依据。同时平日里也在社交软件中及时交流实现中的不合理之处，如果是小问题，就及时hotfix进行解决，否则就记录在会议文档的问题中，留在会议中大家一起进行讨论商定结果。整体来看，根据迭代二的长期计划，以及一次次的短期计划，我们小组基本上有条不紊地完成了开发任务。

团队建设、小组分工

与迭代一相比，迭代二的开发路线更为崎岖。除了数据量激增带来的技术挑战外，工程量的扩大和因其他课程任务繁重导致的时间紧缺使得整个迭代二都不如迭代一那样顺利。

但正是因为开发存在着难度，我们更加为每一个功能的完成、每一个漏洞的解决、每一次性能的优化感到快乐与雀跃。也正是这些挑战，促进了团队的成长、提高了小组成员间的默契。我们不断切换着自己的角色，既是自己负责模块的开发人员，也是队友的测试人员，更是这个项目的第一批(也可能是唯一一批)用户。在组长对高可用追求的带领下，我们心中都默默设立下了一个目标：希望能让我们的网站不仅仅是一个实现了基本功能的课程作业，而是一个“真正可以提供服务”的产品。

个人总结

171250662孙逸伦

总的来说，迭代二的开发比迭代一要辛苦。一方面，功能比迭代一多了很多（这一点从接口数目上就可以看出，多了二十多个接口），同时前端进行了重构，新技术的使用增加了学习成本；另一方面，正所谓计划赶不上变化，期间各种各样的“突发事件”让预订的计划一次又一次地发生变更（比如各种作业），这给我们的计划实施带来了困难。

诚然，硬件负载只是暴露了性能问题，并不是性能问题的来源，但在服务器启动之后就只剩200M内存的时候……还好前端不太吃硬件，但CI不能用实在是让我很崩溃。

整个开发过程中，除了功能的开发之外，贯彻始终的就是两个字，“优化”，为了质量而优化。我觉得，也正是对这两个字的追求，支撑着我（们）不断地在这个项目中投入时间和精力。

171250635赵文祺

在软件工程与计算三迭代二的开发过程中，我的任务是做运维部署，以及数据的获取和处理。

在数据获取方面，我是用scrapy来进行大规模数据爬取，总共获取了IEEE网站中17万条论文数据。由于数据的大量增加，原本并不明显的性能问题也逐渐暴露出来，这使得我们需要对数据的存储和部署方式进行优化。

我们最后选择使用Google云计算服务中的Kubernetes容器云服务。经过一段时间的学习、实践之后，我充分感受到了基于云原生的构建、测试、部署的强大之处。Kubernetes平台使得我们可以不必关心集群的大小以及机器组成，其带来的自动重启、滚动更新、水平扩容等功能大大简化了运维人员的工作流程。同时，也极大地提高了服务的可用性，这使得小团队的开发人员可以将注意力更多地集中在创造业务价值。

171250023陈耿阳

在迭代二开发中，我负责前端用户端的页面设计和前端开发。与迭代一不同的是，迭代二的数据量增加，后端计算时间增加，直接导致了接口变慢。为了提高性能、保证用户使用体验，组长决定对前端进行重构，采用服务端渲染。所以除了完成迭代二本身的业务开发，还要对已有的迭代一代码进行优化和维护，在工作量上，这对于我们是不小的挑战。

同时，随着网站规模的扩大，如何保证可用性的前提下提高整体设计的一致性和完整性、交互的可学性，如何带给用户控制感并让网站真正能帮助用户实现目标，都成了在UI/UX设计上遇到的难题。因为缺乏专业的图形、视觉、交互设计知识，界面上常常存在着各种人机交互设计上的bug，非常感谢我的队友，他们不断从用户角度提出对界面的改进建议，这也让我开始学着站在产品视角进行前端开发，而不仅仅是后端功能的前端实现。

171250635田晨江

在迭代二开发中，我负责后端的业务开发。这次给我印象最深的是在较大数据量下的开发。迭代二中我们加入了170k的论文，也暴露了一些在较小数据量下被硬盘掩盖的问题。因此我也运用了一些技巧来处理大数据量下的问题，比如减少服务端和数据库的交互次数、尽量只取需要的数据字段，减少网络传输时间、优化算法的实现等等。其次是搜索接口的实现，在迭代二初期，选择了用mongo和文本索引来实现基本搜索功能，但是由于数据量巨大，返回的数据占原有数据集的比例比较大，而且索引占的空间也很大，经过权衡放弃了这种实现，转而选取了非常流行的elasticSearch搜索引擎，利用倒排索引加速搜索效率，并且做了mongo和es的数据同步机制，保证了一致性。最后是缓存的加入，对于经常访问并且参数基本一致的接口，加入了缓存，减少了访问数据库的次数，提升了效率。