Project Helper

Executive abstract

本项目旨在为学校师生提供一个全新的、类似于Blackboard的课程项目服务平台。通过使用**Quasar**和**Vue.js**，我们旨在创建一个**用户友好的交互界面**，为师生提供更为流畅的在线学习体验。后端使用**Spring Boo**t与**MyBatis-Plus**确保**稳定、高效**的数据处理和存储。同时为了保障信息的安全性，我们采用了**Spring Security**和**JWT**技术。此平台不仅提高了访问效率、支持**高并发访问**，还在原Blackboard平台的基础上进行了**功能的改进和增加**，使其更**易于维护和测试**。

Team

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 汤玉磊 | 张未硕 | 程嘉朗 | 李伟浩 | 黄硕 |

Description

Movitation

在数字化学习环境中，许多现有的课程管理平台，如Blackboard，它提供了丰富的功能，这一点值得我们学习和模仿。但是它仍存在界面过于单调，用户需求痛点无法满足等问题。这影响了在线教学的效果和体验。 我们的愿景是创建一个稳定、高效且用户友好的在线学习平台。一个可以保障信息安全、支持高并发访问的平台，它不仅能满足教师和学生的基础需求，还能在功能上持续创新，为用户带来前所未有的在线学习体验。为了实现这一愿景，我们提出以下解决方案：

1. 使用Quasar和Vue.js开发用户友好的前端界面，确保流畅的用户体验。
2. 采用Spring Boot和MyBatis-Plus构建后端，提高数据处理的效率。
3. 通过Spring Security和JWT技术保障用户信息和数据的安全。
4. 在原有Blackboard功能的基础上进行改进和增加，如增加聊天区、多方式登录功能等，同时保证平台易于维护和测试。

Feature description

Some User Stories

教师Tom:

当我想发布一个新的课程项目通知时，我希望能够轻松地选择要通知的学生，这样我可以确保只有相关的学生看到此通知。

学生Alice:

当我登录平台时，我想要能够编辑我的个人技能和寻找意向的团队伙伴，这样其他学生可以知道我的技能并可能与我组队。

学生Bob:

当我看到一个我感兴趣的团队时，我想要能够申请加入，这样我可以参与到我感兴趣的项目中。

教师Lucy:

当学生提交了作业后，我希望能够在线查看、评论并打分，这样我可以更高效地管理和评估学生的作业。

**UML**

**Mockups**

Requirements

Functional requirements:

1. 用户角色管理:支持教师、学生、管理员、助教和团队领导等多种角色。不同角色有不同的权限和功能。
2. 课程项目通知:教师可以向所有或某些学生发送或修改课程项目通知。
3. 个人信息管理:学生可以编辑和更新其在网站上的个人信息，包括技术栈、编程技能和意向团队伙伴。
4. 团队管理:教师可以根据团队大小、团队截止日期、团队编号等信息创建多个团队。学生可以自由加入团队，但每个学生只能加入一个团队。
5. 作业管理:教师可以发布个人或团队作业，审查、评论并为已完成的作业评分。学生可以提交作业。
6. 成绩册管理:教师可以以Excel或CSV格式上传成绩。学生可以查看其成绩。

Non-functional requirements:

性能:

1. 响应时间：每次请求的平均响应时间应低于1.5秒。
2. 结果准确性：所有功能、查询和计算的准确性达到99.9%。

存储要求:

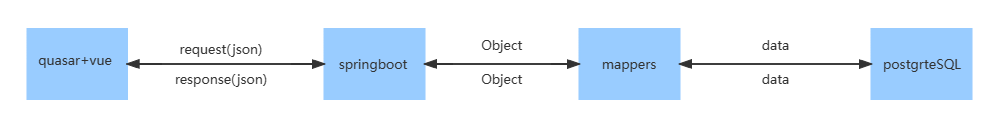
1. 学生、教师和管理员的个人信息、作业、成绩等数据需要持久化存储。
2. 需要备份策略以防数据丢失。
3. 考虑到用户数量和数据增长，初步预估需要至少10TB的存储空间。

部署成本:

假设每个用户（学生、教师等）每月的平均运营成本为1元，如果学校有10000名用户，则每月的运营成本为10000。

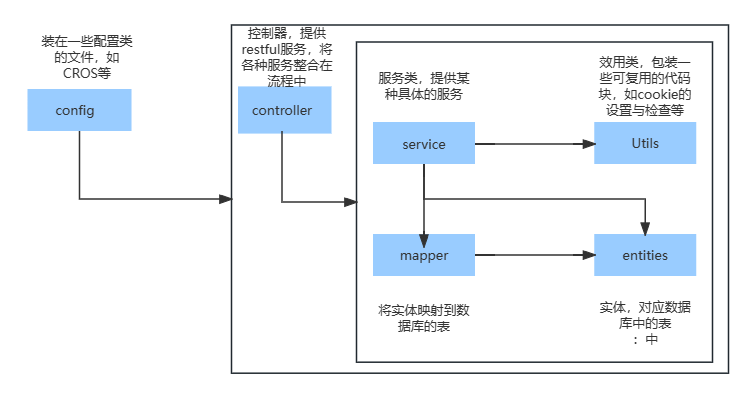
Architecture:

Block Diagram:

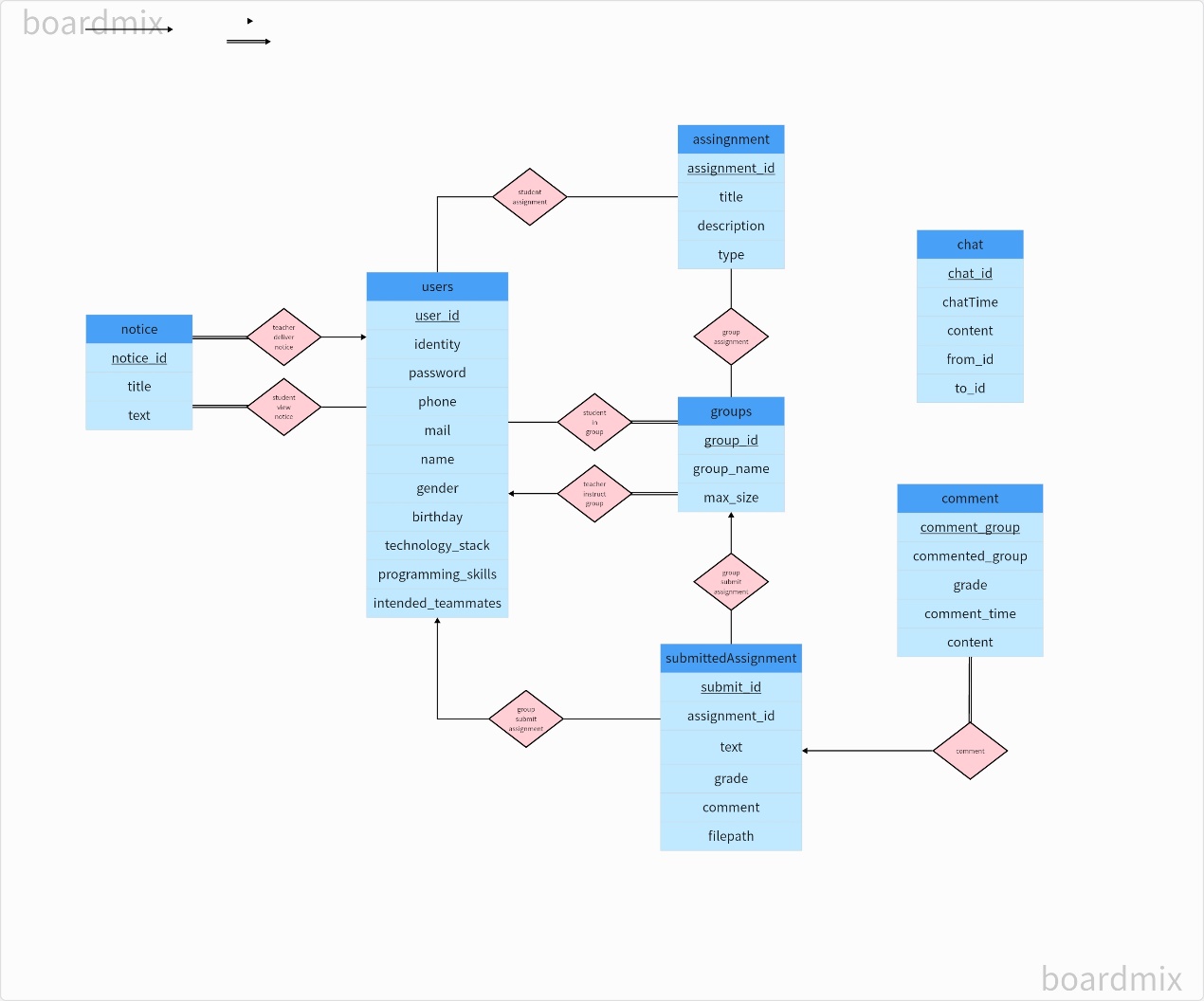


**Flow charts：**

Class Diagram:



Database Schema:



**Timeline:**

**Key Dates:**

10/20 - 前端原型完成

11/10 - 后端API基础完成

11/25 - 安全和数据库集成

12/15 - 完整的系统测试和部署

Effort Required:

前端开发: 100小时

后端开发: 150小时

数据库设计: 40小时

测试和部署: 30小时

APIs, services:

External APIs:

1. OAuth 2.0: 用于身份验证

Internal Services:

1. User Service: 用于处理用户注册、登录、权限等
2. GroupService: 用于团队创建和管理
3. ProjectService：用于管理不同的project
4. NoticeService：用于发布修改删除Notice

Description: Feasibility

1. Lack of Familiarity with APIs:

应对策略：提前进行技术原型验证，熟悉并测试所有将要使用的API。

2. Unable to Deliver on Performance:

应对策略：选用高性能的框架、合适的数据库设计以及合理的系统架构，定期进行性能测试。

3. Cost Excessive:

应对策略：提前做好成本评估，考虑开源解决方案和免费服务，并密切监控项目成本。

4. Existing Projects:

应对策略：确保有足够的创新点和独特功能，进行市场分析以定位自己的优势。

5. Too Many Features (prioritize):

应对策略：优先处理核心功能，其他附加功能可以后续版本中逐步加入。

6. Third Party APIs/Service May Not Be Reliable:

应对策略：选择有良好声誉和历史记录的第三方服务，同时为关键功能建立备选方案。

Technologies

Tools:

1. Version Control: Git
2. IDE: IntelliJ IDEA Ultimate
3. Build Tool: Maven
4. Testing: JUnit (由spring-boot-starter-test提供支持)
5. Database: PostgreSQL
6. Development Utilities: Spring Boot Devtools

Libraries:

1. Web Framework: Spring Boot Web
2. Security: Spring Security, JWT (Java Web Token)
3. Object Relational Mapping (ORM): Spring Data JPA, MyBatis Plus
4. Caching: Spring Cache, Redis (由spring-boot-starter-data-redis支持)
5. JSON Parsing: Fastjson
6. Database Connection Pool: Druid
7. Automatic Code Generation: Lombok
8. Word Processing: Aspose Words
9. Presentation Processing: Aspose Slides

Services:

1. Storage: 数据存储主要依赖于PostgreSQL数据库，并使用Redis进行缓存。
2. Search: 可以考虑使用Fastjson库进行JSON格式的数据解析，MyBatis Plus进行数据库查询和搜索。
3. Computation: 主要由Spring Boot和相关的后端组件处理，例如处理HTTP请求、业务逻辑、数据查询等。