QG后台组最终考核

前言

时间安排

安排	时间
发布最终考核题目	2023-04-12
中期进度检查、答疑	2023-04-21
考核项目提交	2023-04-26
最终考核预答辩	待定

评分细则

1. 功能完成程度
2. 拓展与创新
3. 代码质量
4. 代码风格
5. 项目报告文档
6. 项目答辩汇报

提交内容

1. **源代码**（注意包含数据库SQL脚本）
2. **项目打包**（war文件，如果修改了Tomcat配置请一并附上，并进行说明）
3. **项目报告文档**
4. **项目汇报PPT**
5. 建议附加**进度文档**，写明开始的计划与过程中某个时间点解决了什么问题

注意：**请重视项目报告文档和项目汇报PPT这两部分内容！**

备注

1. 考核着重于后台部分，不要求前端做的精美。
2. 建议先做好需求分析与规划，再开始写代码。
3. 不建议在有限的时间内实现需求外的功能，在需求框架内不断精进更有意义。当然有意义有价值的扩展也会酌情加分

QG资金管理系统

该项目为开发一个资金管理系统，以满足用户对流动资金的管理，旨在提供一个安全高效，功能丰富的资金管理系统。

背景

- 数字货币成为越来越多人的支付方式选择，一个安全高效，功能丰富的资金管理网站可以更好地满足用户的需求，可以让用户更好地管理自己的资金，确认资金明细。

需求

基本需求

系统模块

1. 企业群组
    1. 普通用户可以申请企业群组并成为企业群组负责人
    2. 企业群组中有企业的介绍和相关信息，如企业名称、企业人数、企业规模、企业工作方向等。
    3. 企业群组可以设置申请群组方式，公开模式下群组可以被搜索到，通过ID等其他方式主动添加，私密模式下仅可通过管负责人拉取
    4. 企业群组内，有权限的人可以收款
2. 用户中心
    1. 用户可以设置、查看、修改个人信息，包括用户名、头像、手机号等信息。
    2. 用户可以查看自己的资金明细，并且分类查看账单流水。
    3. 用户可以选择查看所属企业群组，并且进行申请或者退出。
3. 资金中心
    1. 用户可以在这里支配自己的资金。
    2. 用户支出资金可以通过企业群组内分配的资金支付并且选择是否上传报销文件。
    3. 用户可以查看在软件的总资金，也可以查看不同群组的资金余额。
    4. 企业群组管理员可以充值到企业群组里并且分配资金给企业群组成员。
    5. 企业群组管理员可以查看企业群组里所有资金流动明细。

用户模块

1. 普通用户
    - 账号注册（录入用户名、昵称、密码、地址等基本信息，可以自行拓展）
    - 实现登录、退出登录功能（未登录为游客权限）
    - 支持查看、修改个人信息（昵称、密码、资金和所属企业群组等。用户注册后会有一个默认头像）
    - 可以进行搜索公开群组查看群组的基本信息
    - 可以申请加入公开群组也可以被企业群组负责人拉入群组，也可以选择退出群组。
    - 可以查看流水
2. 游客
    - 未注册账号的用户是游客，仅可以搜索公开企业
3. 企业群组负责人
    - 企业负责人具有普通用户所有功能
    - 企业群组负责人是普通用户向平台申请创办企业群组负责人
    - 申请成功后，企业负责人可设置企业的基本信息，并且设置是否公开

- 企业负责人可以将资金充值到某个企业群组中，并且任意支配（分配或者收回）资金到人，
- 企业负责人可以选择注销企业群组

4. 网站管理员
   - 可以设置企业、用户是否封禁。
   - 可以通过企业管理员、用户的恢复申请。
   - 可以查看网站内余额总数以及异常交易

资金模块

1. 用户使用自己或群组资金支付流程，**需注意，群组资金实际仍在群组名义下，因此转账过程中主体为群组**
   - 开始支付（发起支付）
   - 授权（输入密码并校验密码是否正确，检验支付环境是否正常）
   - 校验余额，检测余额是否充足。
   - 资金冻结（获取用户资金并冻结对应资金，存储在数据库）
   - 结算
     - 结算：更改支付状态为已结算，并进行入账到第三方
     - 不结算：更改支付状态为退款，将资金从数据库转回用户账户
   - 入账：向第三方发起付款请求
     - 入账成功：成功入账，资金从数据库清除，告知用户并扣除资金
     - 入账失败：入账失败，资金保留数据库，提示用户是否再次进行入账
   - 记录交易信息

2. 用户使用个人或者群组资金收款流程：
   - 提供收款接口
   - 接收第三方收款请求，验证请求合法性
     - 授权成功，冻结金额，避免重复支付或者支付失败
     - 授权失败，拒绝收款请求
   - 获取收款用户ID及其他附加信息
   - 生成资金存储在数据库
   - 将收款金额入账到收款人账户，如果入账成功，则进行余额更改；如果入账失败，则进行相应处理。
   - 更新收款人账户余额，表示收款成功。返回成功请求
   - 记录交易信息

以上操作的原子性需要自行判断并且仔细写在README文档上，最好有具体的流程图，第三方的收款和付款仅需模拟

前端

1. 前端页面展示，与后台有一个良好的交互

加分需求

通用模块

- 使用日志实时监控系统运行状态，记录关键或者异常情况（lomback， log4j2）
- 数据库的敏感信息使用加密算法进行加密

系统模块

- 支持验证登录，对信息敏感数据加密传播
- 企业群组内有聊天室和私聊功能。
- 当企业群组的资金动用时，企业负责人可以收到提示
- 网站管理员封禁用户或者企业时给予实时通知并且冻结一切正在进行的操作

用户模块

- 前端数据实时显示
- 考虑到**多线程、高并发，资源抢占**问题，保证服务器安全
- 考虑到禁止洗钱的行为（可以把思考过程写在README中）
- 流水可以打印输出txt文件或者其他文件
- 支持多文件上传和大文件上传
- 管理员和负责人可以批量下载文件

高级需求

- 对数据库进行保护，有抵抗恶意攻击的能力
- 前后端传输数据使用加密传输
- 支持高并发访问，并且保证平台稳定性。
- 保证资金的实时更新，用户可以查看资金流动状态
- 使用**锁和事务**保证资金流转过程的绝对安全
- 使用**本地缓存**进行数据统计和存储，以减轻数据库压力（如演唱会抢票收款）
    - Ehcache、Guava Cache或Caffeine
- 实现**负载均衡算法**，当一个收款流量过大时，可以分发给下游服务
    - Nginx、HAProxy等负载均衡器，或者在应用代码中实现自定义的负载均衡算法
- **故障转移**，当前服务器发生故障时，可以把请求重新路由到另一个可用服务器中
    - 负载均衡器、集群管理工具（如Zookeeper或Etcd），实现自定义的故障转移策略

QG线上卡牌对战游戏

项目简述

要求在Web上开发一个类似于《炉石传说》的回合制策略卡牌对战游戏。每回合打出的最高卡牌数会被限制，被打出的卡牌在场上可以**攻击对方的卡牌**或**直接攻击对方玩家角色**。当某玩家角色生命值降至零及以下时，该玩家战败。

基本功能需求

系统模块:

普通用户端

- 基本的登陆注册功能。
- 个人基本资料的设置与显示。
- 玩家的历史胜负情况展示。
- 系统个人邮箱
    - 系统个人邮箱中可以**单方面**接收游戏官方(管理端)发送的**信息**。

游戏策划管理端

- 游戏策划可以在系统中**设计新卡牌并在游戏中发布**，并且玩家可以以某种方式**获取新卡牌**。
- 游戏策划可以向全服玩家的系统个人邮箱**广播消息**，或向部分玩家单独发送信息。

游戏模块:

卡牌收集和构建

- 玩家可以通过某种方式，例如游戏内活动、购买或系统赠送，来获得卡牌。
- 玩家可以根据自己的策略和喜好，选择**已拥有**的一定数量的卡牌，构建自己的出战卡组。

卡牌的基础属性

- 每张卡牌都具有基础属性：**攻击力**、**血量**。
- 根据设计的不同，每张卡牌可能还会有不同的**体力花费**或其他**能力**。

战斗系统(重点)

- 玩家可以申请与其他玩家进行**实时联机**对战，决出胜负。
- 游戏开始时，由玩家的出战卡组生成顺序随机的**抽牌堆**，每回合会从**抽牌堆**中抽取若干张牌，打出的牌在死亡后则去到到**弃牌堆**。
- 玩家在其回合打出的最高卡牌数会被**限制**。限制的具体实现可自行决定，例如**回合体力限制**或**数量限制**等等。
- 被打出的卡牌在场上可以**攻击对方的卡牌**或**直接攻击对方玩家角色**。
- 每个玩家角色都有一定数量的生命值，当生命值减为0及以下时，游戏结束。
- 实现**断线重连功能**，确保玩家在网络中断后可以重新连接到游戏，并且保持游戏状态不变。

前端页面部分:

良好的用户界面和用户体验

- 设计一个可用的用户界面，方便玩家进行卡牌选择、战斗操作等。
- 样式不需要有多么花哨和华丽，能正常展示功能即可。
- 可使用任意前端框架和组件库。

进阶需求

系统模块:

社交功能

- 添加社交功能，包括添加好友、好友私聊、创建战队等。
- 允许玩家(好友)之间使用系统个人邮箱发送信息。
- 允许玩家回复游戏策划发送的通知。

游戏策划管理端

- 游戏策划可以设置周期性的**任务**、**活动**和**赛季系统**，玩家可以参加活动赚取奖励，提升排名。游戏策划则可以选择合适的时间开放或更新**任务**、**活动**和**赛季**。
- 允许设置为广播或发送给玩家的信息为**可回复**或**不可回复**。
- 使用自主实现的**消息队列模块**完成以下功能:
    - 游戏策划可以向特定的玩家发送**礼物**。

游戏模块:

排行榜和竞技场

- 添加排行榜功能，记录玩家的战绩或积分，并**展示排名**。
- 设计竞技场模式，玩家可以在竞技场中与其他玩家进行排名比赛。

更丰富的卡牌效果和属性

- 玩家可以通过消耗系统货币来获取卡牌。也可以分解自己拥有的卡牌为系统货币。
- 在卡牌的原有的基础属性:攻击力和血量的基础上，为卡牌添加其他特殊的**能力或特性**。
    - 例如:一次性消耗的法术卡、使用后会重新回到抽牌堆的特殊卡牌等。
- 在卡牌之间设计增加**相克关系**或**相互作用**。
- 其它能够增强游戏趣味性和策略性的设计。

加分点

- 完成任意进阶需求。此加分可累加，且部分完成亦能加分。
- **安全性**
    - 实现验证码登录，对敏感信息进行加密传输。
    - 在数据库中对敏感数据进行加密存储。
- **实时通讯**
    - 使用**WebSocket**技术实现项目中有**强实时**需求的部分，例如实时联机对战的部分。

- **用户权限与鉴权**
  - 使用**RBAC模型**结合其他技术，如**JWT**等，实现**用户权限管理与鉴权**。
- **高并发的问题**
  - 使用锁和事务优化项目中可能出现的并发问题。
- **系统日志记录**
  - 记录服务的所有**操作日志**，保存为文件，包括用户登录、卡牌发布、赛季更新和对局结束等操作，以便审计和追溯。

写在本项目最后

由于游戏设计本身就是一个需要放飞设计者本身想象力的一个领域，且我们考核项目的考核重点在于你的**学习能力**、**技术掌握程度**和**项目完成度**，所以对于文档中没有明确要求的内容，特别是**游戏性上的设计**，均可以根据同学们自己的理解，在保证项目完成度的基础上进行实现。

QG OJ

背景：

QG 判题系统是一款用于在线评测学生编程作业的系统，它可以接收学生提交的代码，运行测试用例，并根据预先设定的评测规则对代码进行评分和反馈。

目标：

开发一款功能完善、稳定可靠的 QG 判题系统，能够支持多用户同时在线提交代码，运行测试用例，并返回评测结果。

基本需求：

1. **用户认证和权限管理**：
   - 实现用户注册、登录、登出功能，用户密码使用安全的加密存储。
   - 区分管理员和普通用户，管理员具有更高的权限，可以管理用户、题目和测试用例等信息。
2. **题目管理**：
   - 管理员可以添加、编辑、删除题目信息，包括题目描述、输入输出样例、难度等级，题目标签等。
   - 普通用户可以查看题目列表和题目详情，也可以根据题目的标签和其他信息搜索。
3. **测试用例管理**：
   - 管理员可以为每个题目添加、编辑、删除测试用例，包括输入和期望输出。
   - 普通用户可以查看每个题目的测试用例。
4. **代码提交和评测**：
   - 用户可以提交代码，进行评测。
   - 系统应能够运行提交的代码，并根据测试用例对其进行评测，返回评测结果（包括通过的测试用例数量、耗时、内存占用等）。
5. **评测结果展示**：
   - 用户可以查看自己提交的代码的评测结果，包括通过的测试用例、未通过的测试用例和相应的错误信息。

- 管理员可以查看所有用户提交的代码的评测结果，并进行必要的管理操作（如禁止带有攻击意图用户的代码提交）。

加分需求

1. **安全和性能优化**：
   - 保证系统的安全性，防止恶意用户的攻击。
   - 优化代码运行和评测的性能，尽量减少评测时间和资源占用，防止用户提交了死循环的代码一直无法停止。
2. **日志记录和异常处理**：
   - 记录用户的操作日志，包括登录、注册、提交代码等操作。
   - 对系统可能出现的异常进行合理处理，并记录异常日志，方便后续排查和修复问题。
3. **竞赛模式：**
   - 添加一个竞赛模式，管理员可以创建比赛并邀请其他用户参加。在比赛中，用户可以提交问题，并根据正确性和提交时间获得得分。
   - 引入罚时机制，对于每次提交，答案错误的同学会得到罚时
   - 排名查看，在竞赛过程中用户和管理员可以实时查看当前比赛排行榜，排名按成功解答题目的数量进行排序，如果解答题目一样，罚时更少的队伍排名更前
   - 用户可以查看自己参加竞赛的历史和提交解答记录
4. **题解：**
   - 添加一个讨论区功能，供用户在题目、代码和评测结果上进行讨论和交流。用户可以发表评论、提问，并回复其他用户的评论。
   - 用户在解答正确后可以发布题解，用户可以查看其他用户发布的题解并进行点赞、评论提问和回复
   - 管理员可以监督讨论区，删除不当言论或恶意攻击的内容。

写在本项目最后

本题目的`提交代码测评`功能若完全自主实现难度较高，故此部分的功能实现方法不作限制，可参考`Vjudge`的方案或在`Github`上寻找开源解决方案并应用.

QG在线学习平台

项目概述

本项目旨在开发一个在线学习平台，为学生和教师提供一个便捷的教学与学习环境。学生可以通过平台选择和参与教师创建的选修课程，进行课程学习和答题，教师可以创建课程、管理课程内容和监控学生学习情况。

基本需求

教师模块

- **创建课程**：教师可以创建新的课程，课程要有合理的章节划分，可以先做简单的文本类。
- **教师信息**：教师可以添加自己的个人介绍、电子邮箱、qq等个人信息并可以修改。
- **管理课程**：教师可以设置课程的基本信息如名称，描述，开课时间，结课时间，报名人数限制等。
- **查看报名学生情况**：教师可以查看自己课程中报名的学生情况

- **查看学习情况**：教师可以查看自己所属课程中学生的学习情况
- **添加题目**：教师可以对课程章节添加题目，包括选择题、填空题、简答题等。
- **查看学生答题情况**：教师可以查看学生对于题目的答题情况，包括正确率等。
- **统计分析**：教师可以查看课程的整体学习情况，包括学生平均成绩、答题情况等。
- **参与讨论**：教师可以参与课程的讨论区，并回复学生的讨论。

学生模块

- **注册与登录**：学生可以注册新账号或使用已有账号登录平台。
- **学生信息**：学生可以添加自己的个人介绍、学号、年级等必要个人信息并可以修改。
- **浏览课程**：学生可以浏览平台上所有可选的课程，并查看课程详情。
- **选择课程**：学生可以选择感兴趣的课程进行学习。
- **学习课程**：学生在课程开放时间内可以学习课程内容，并完成相关题目。
- **查看答题情况**：学生可以查看自己的答题情况和历史记录。
- **学习记录**：学生可以查看自己的学习记录，包括已学习课程、做题情况等。
- **学习情况**：记录统计学生每个章节的学习情况，学习情况包括答题多少、正确率等信息。
- **学习讨论**：学生可以在课程的讨论区与其他学生和教师进行交流和讨论。

加分需求

- **实时提问**：学习讨论区满足不了学生的求知欲，请做一个聊天室让学生可以实时与老师对话来达到更好的学习效果。
- **屏幕共享**：可以在上一个需求的基础上，添加视频共享的功能，让老师和学生更好的对话
- **反抄袭**：老师希望学生的作业能够独立完成，所以希望在答题情况中可以看到各个学生答案的重复率，当然，只在问答题中检查抄袭即可。
- **支持视频课程**：课程中章节学习内容可以支持视频形式的学习
- **热门排行**：用户可以查看热门的课程排行榜，和学习情况较好的用户排行榜。
- **验证登录**：支持验证登录，对信息敏感数据加密传播
- **系统日志记录**：记录平台的所有操作日志，包括用户登录、课程上线、学生选课等操作，以便审计和追溯。
- **数据库保护**：对数据库进行保护，有抵抗恶意攻击的能力
- **加密传输**：前后端传输数据使用加密传输
- **数据备份与恢复**：定期对平台的数据进行备份，以防数据丢失或损坏，同时提供数据恢复功能。

通用加分需求

手机短信验证码

安全性

后端接口要求权限校验，不可随意响应

前后端都做好**正则表达式**校验

防止**SQL注入，XSS攻击**

扩展性：代码结构，数据库结构清晰，方便扩展功能

健壮性

无严重bug

使用JUnit进行单元测试

异常处理

技术要求及规范

基本技术要求

1. JDK 版本推荐为 1.8，也可以使用 jdk11 或 jdk17
2. MySQL 推荐为 5.7 版本，也可以使用 8.0
3. Java、Mysql、Javaweb、Git
4. 可以使用前端框架 JQuery、Ajax、Bootstrap、Layui 等开发前端
5. 可以使用Maven，但是只能引入如下的依赖
    - Mysql连接驱动
    - Servlet
    - JSON序列化工具
    - Junit
    - logging等日志框架

如果有其他需要使用的maven依赖，请在群里问一下师兄。

注意：**不允许使用一些第三方的框架！**

进阶技术要求

1. 使用**JAVA8**的新特性优化你的代码，如时间类（LocalDateTime）、Stream、Optional等
2. 使用**Apifox、postman、eoLink、swagger**等工具（任选其一）记录接口文档（这是开发的良好习惯，也方便测试接口）
3. 使用小组作业要求的**数据库操作工具类，CRUD操作工具类**
4. 使用小组作业选做要求的**数据库连接池**
5. 不要使用**联表查询**，并了解为什么
6. 使用**常量类**和**配置文件**
7. 实现**全局异常处理**
8. 对多角色多权限使用**RBAC**模型和**JWT**进行**认证鉴权**
9. 拥有良好的**日志**记录和**注释**习惯
10. 需要有足够的健壮性和防御，前后端使用正则表达式校验，**抵御Sql注入、Xss攻击，并对用户数据进行脱敏**
11. 学习像Spring一样实现**依赖注入**(DI:Dependency Injection)、进行**控制反转**(IOC:Inverse OfControl)、**面向切面编程**(AOP:Aspect Oriented Programming)
12. 使用**自定义注解**，对12点的要求进行优化

规范

1. 前后端通信使用**json格式**
2. 使用小组作业要求**BaseServlet**
3. 使用小组作业要求的**MVC分包**
4. 使用小组作业要求的**统一结果集**
5. Git**提交粒度**尽可能小，一个功能至少提交一次，修复一个bug也提交一次。
6. Git新建分支进行开发，master只能用于发布正式版本