

项目要求

提交方式: GitHub 仓库链接到 `2451918322@qq.com`

截止时间: 2月20日

1. 一、项目概述

1.1 1.1 项目背景

本项目旨在开发一个面向红岩网校内部使用的作业管理系统，实现「老登」（管理员）发布作业、「小登」（用户）提交作业的完整流程。系统需要支持多部门管理，包括：**后端、前端、SRE、产品、视觉设计、Android、iOS** 七个部门。

1.2 1.2 角色定义

角色	标识	权限说明
小登	<code>student</code>	学员，可提交作业、查看评语、查看优秀作业
老登	<code>admin</code>	管理员/讲师，可发布作业、批改评语、标记优秀作业

1.3 1.3 部门枚举

枚举值	显示标签	说明
<code>backend</code>	后端	后端开发
<code>frontend</code>	前端	前端开发
<code>sre</code>	SRE	运维工程
<code>product</code>	产品	产品设计
<code>design</code>	视觉设计	UI/UX 设计
<code>android</code>	Android	Android 开发
<code>ios</code>	iOS	iOS 开发

注意：API 响应中需要同时返回 `department`（枚举值）和 `department_label`（中文标签）

1.4 1.4 技术要求

本项目需要综合运用本学期所学的全部知识：

- Go 基础语法：变量、函数、结构体、接口、指针等
- Go 进阶：goroutine、channel、select（你最喜欢的并发竞态）
- Git：规范的 commit message（请不要只有一条init commit🙏）
- HTTP 框架：Gin 或 Hertz（二选一，之后接触网校的业务更多的是使用Hertz）
- 数据库：MySQL + GORM（原生SQL不用但是得会）
- 认证：JWT 认证机制（使用双token）
- 代码规范：项目结构规范、错误处理规范

2. 二、基础功能清单（必做）

2.1 2.1 用户模块

功能点	说明
用户注册	支持用户名、密码、昵称、部门
用户登录	返回 Access Token和Refresh Token
获取用户信息	需要 Token 认证
注销账号	软删除
密码加密	加盐哈希
刷新Token	使用Refresh Token获取新的Access Token和Refresh Token

2.2 2.2 作业模块

功能点	说明
发布作业	老登可发布，设置截止时间和补交策略
作业列表	按部门筛选，分页查询，显示部门标签
作业详情	关联查询发布者信息，包含部门标签
修改作业	同部门老登均可修改，需处理并发
删除作业	同部门老登均可删除

并发控制说明：

- 同部门的老登可以相互修改作业内容
- 处理并发竞争

2.3 2.3 提交模块

功能点	说明
提交作业	小登提交，自动判断是否迟交
我的提交	小登查看自己的提交和评语
查看部门提交	老登查看自己部门学员的提交
批改评语	老登添加分数和评语
标记优秀	老登标记优秀作业
优秀作业展示	所有小登可查看优秀作业

2.4 2.4 其他要求

功能点	说明
JWT 中间件	Token 解析和验证
权限中间件	区分老登/小登权限
统一响应格式	code + message + data
错误处理	统一错误码
部门标签	响应中包含 department_label
API 文档	使用 Apifox/Postman 导出
Git 规范	规范的 commit message

3. 三、进阶功能清单（选做加分）

3.1 3.1 邮箱与通知

功能点	说明
绑定邮箱	用户可绑定邮箱
作业发布通知	新作业发布时邮件通知
截止提醒	截止前 24 小时提醒
评语通知	作业被批改时通知

3.2 3.2 前端对接

功能点	说明
Vibe 前端	使用 AI 工具辅助生成前端页面(只有一个要求 不能是蓝紫配色)

功能点	说明
前后端联调	完整的开发流程

3.3 3.3 AI 作业评价

功能点	说明
AI 初评	调用 AI API 对代码进行初步评价
代码分析	分析代码质量、规范性
建议生成	生成改进建议供老登参考

可使用 OpenAI、Gemini、Claude API 或国内大模型 API，直接调接口即可

3.4 3.4 部署相关

功能点	说明
交叉编译	编译 Linux 可执行文件
Linux 部署	部署到 Linux 服务器
Docker 部署	编写 Dockerfile
docker-compose	包含 MySQL + App

3.5 3.5 考核系统

功能点	说明
任务类型	区分「作业」和「考核」
多人阅卷	考核支持多位老登阅卷
并发处理	处理同时批改的并发冲突
阅卷分配	自动或手动分配阅卷任务

4. 四、项目结构参考

```
homework-system/
├── api/                # API 文档、接口定义
│   └── docs/
├── cmd/                # 程序入口
│   └── main.go
├── configs/            # 配置文件
│   └── config.go
├── dao/                # 数据访问层 (Data Access Object)
│   ├── user.go
│   └── homework.go
```

```
├── submission.go
├── handler/                                # HTTP 处理器 (接收请求、返回响应)
│   ├── user.go
│   ├── homework.go
│   └── submission.go
├── middleware/                             # 中间件
│   └── auth.go                             # JWT 认证中间件
├── models/                                # 数据模型 (GORM 模型定义)
│   ├── user.go
│   ├── homework.go
│   └── submission.go
├── pkg/                                    # 公共工具包
│   ├── jwt/                               # JWT 工具
│   ├── response/                          # 统一响应格式
│   └── errcode/                           # 错误码定义
├── router/                                # 路由定义
│   └── router.go
├── service/                               # 业务逻辑层
│   ├── user.go
│   ├── homework.go
│   └── submission.go
├── go.mod
├── go.sum
└── README.md
```

Fence 1

各层职责说明:

层级	职责
handler	接收 HTTP 请求, 参数校验, 调用 service, 返回响应
service	业务逻辑处理, 调用 dao 进行数据操作
dao	数据库 CRUD 操作, 与 models 配合使用
models	GORM 模型定义, 对应数据库表结构
router	路由注册, 将 URL 映射到对应的 handler
middleware	请求拦截中间件, 如认证、跨域、日志等
pkg	通用工具函数, 可被各层调用

5. 五、数据库设计

5.1 5.1 用户表 (users)

字段	类型	说明
id	BIGINT	主键, 自增

字段	类型	说明
username	VARCHAR(50)	用户名, 唯一
password	VARCHAR(255)	密码 (bcrypt 加密)
nickname	VARCHAR(50)	昵称
role	ENUM('student', 'admin')	角色
department	ENUM('backend', 'frontend', 'sre', 'product', 'design', 'android', 'ios')	所属部门
email	VARCHAR(100)	邮箱 (进阶功能)
created_at	DATETIME	创建时间
updated_at	DATETIME	更新时间
deleted_at	DATETIME	软删除时间

5.2 作业表 (homeworks)

字段	类型	说明
id	BIGINT	主键, 自增
title	VARCHAR(200)	作业标题
description	TEXT	作业描述/要求
department	ENUM(...)	所属部门
creator_id	BIGINT	发布者 ID (外键)
deadline	DATETIME	截止时间 (Asia/Shanghai)
allow_late	BOOLEAN	是否允许补交
created_at	DATETIME	创建时间
updated_at	DATETIME	更新时间

5.3 作业提交表 (submissions)

字段	类型	说明
id	BIGINT	主键, 自增
homework_id	BIGINT	作业 ID (外键)
student_id	BIGINT	学员 ID (外键)
content	TEXT	提交内容/链接
file_url	VARCHAR(500)	附件地址 (可选)
is_late	BOOLEAN	是否迟交
score	INT	分数 (可选)
comment	TEXT	老师评语

字段	类型	说明
is_excellent	BOOLEAN	是否优秀作业
reviewer_id	BIGINT	批改人 ID (外键)
submitted_at	DATETIME	提交时间
reviewed_at	DATETIME	批改时间
created_at	DATETIME	创建时间
updated_at	DATETIME	更新时间

6. 六、API 接口设计

API 文档包含以下模块的完整接口定义：

- 用户模块（注册、登录、个人信息、注销）
- 作业模块（发布、查询、修改、删除）
- 提交模块（提交、查询、批改、优秀作业）
- 进阶相关接口（邮箱绑定、AI 评价、考核系统）

7. 七、评估标准

评估以**任务完成度**为核心，基础功能必须完成，进阶功能选做加分。

7.1 7.1 基础功能（必做）

模块	完成标准
用户模块	注册、登录、JWT 认证&刷新、注销功能正常运行
作业模块	发布、列表、详情、修改、删除功能完整，并发控制有效
提交模块	提交、查看、批改、优秀作业标记功能正常
代码规范	项目结构清晰、命名规范、适当注释
API 文档	使用 Apifox/Postman 导出完整文档
Git 使用	commit message 规范、有合理的提交历史

7.2 进阶功能（选做）

模块	完成标准
邮箱通知	绑定邮箱、作业发布/截止/批改通知功能可用
前端对接	有可用的前端页面，能完成完整用户流程
AI 评价	能调用 AI API 生成作业评价建议
部署相关	成功部署到 Linux 服务器或 Docker 容器
考核系统	支持作业/考核类型区分，多人阅卷并发处理

八、提交要求

8.1 代码仓库

- 1 创建 **公开** 的 GitHub 仓库
- 2 包含完整的 README

8.2 README 要求

README.md 需包含：

- 1 已实现功能清单
- 2 进阶功能说明（如有）
- 3 项目简介
- 4 技术栈说明
- 5 项目结构说明
- 6 本地运行指南
- 7 API 文档（Apifox/Postman）

8.3 提交内容

- ☐ GitHub 仓库链接
- ☐ API 文档链接（Apifox 分享链接或 Postman Collection）
- ☐ 部署地址（如有）
- ☐ 演示视频（可选，建议 3-5 分钟 发文件或者b站链接之类的都行）

9. 额外选题

如果你觉得以上选题太多简单，或者想要和 **前端**、**移动** 等部门的同学一起完成考核任务

可以选择任意一位管理学长私聊你的想法，我们会根据你的选题来评估是否可以代替考核（单人考核难度要大，多人考核可以酌情降低难度）

同时，我们并不禁止大家借助ai辅助学习或者完成项目，但是请你了解相关知识，即为什么要这么写，这么写会带来什么好处 什么坏处等等

同时对于疑似AIGC 我们会加大拷打力度，如果回答不上来就不好了 😊