1. 项目结构

- src
  - main：主目录
    - java：主要业务代码位置
      - com.backend.vse：域名反写，作为包目录
        - common：存放通用代码
          - BusinessException类：业务异常封装类，可使用该类静态方法进行异常处理
          - GetStudentIndexByUserIndex和GetTeacherIndexByUserIndex：由于Student、Teacher和User的继承关系，我封装了这两个类用于简化与复用代码
          - Result类：封装请求结果，为SpringBoot代码规范
        - config：存放所使用技术的相关配置文件
        - controller：SpringBoot表现层，接收请求，封装数据，调用业务逻辑层，响应数据
        - dto：数据封装类，将数据封装在类中返回给前端
        - entity：数据库实体类，与数据库表一一对应
        - interceptor：拦截器，使用JWT进行用户鉴权（之前另一组负责）
        - mapper：SpringBoot数据访问层，对数据库的CRUD基本操作，使用MyBatis-plus，方言为MySQL
        - service：SpringBoot业务逻辑层，对业务逻辑进行封装，组合数据访问层中基本功能，形成复杂的业务逻辑功能。
          - impl：实现类，为SpringBoot代码规范，即面向接口开发，service根目录存放接口，impl中存放其实现类
        - tools：工具类

        - ConstantPropertiesTools：阿里云OSS配置文件
        - MailSender：用于发送电子邮件
    - VSEApplication：项目启动文件
    - resources：项目资源文件
        - stitic：静态文件，如HTML，CSS等
            - email_template.html：发送邮件内容模板
        - application.yml：SpringBoot项目配置文件
- test：测试文件目录
- target：项目输出目录，存放项目编译及运行后的一些输出
- pom.xml：Maven包依赖配置文件

什么是SpringBoot三层架构：Spring Boot分层架构详解：从Controller到Service再到Mapper的完整流程_java req从controller层 传到mapper层合理吗-CSDN博客

什么是Maven：Maven是专门用于管理和构建Java项目的工具，它的主要功能有：

- 提供了一套标准化的项目结构
- 提供了一套标准化的构建流程（编译，测试，打包，发布.....）
- 提供了一套依赖管理机制，即管理项目中所使用的依赖包

2. 项目重构思路（两周）

2.1. 项目包依赖更新（syf）（本周）

查阅相关资料，删除未使用的包依赖，更新过时的包依赖，提升项目性能与健壮性，并对每个包的作用添加注释。

2.2. 计划并编写测试用例（软件测试）（yzg）（两周）

测试用例基于WBS，使用V-model，单元测试（函数）->接口测试->业务逻辑测试->项目前后端部署测试。可先对部分接口进行测试用例编写，要包含正确用例、错误用例、null等可能情况。

2.3. 依照三层架构重构代码逻辑（wzk）（两周）

目前代码核心逻辑大多数集中在Controller中，使得Controller负担过大，代码过于臃肿，考虑将部分业务逻辑代码转移到Service层中，并尽量实现代码复用。

2.4. 面向接口开发（wzk）（本周）

目前service层中已经做到面向接口开发，对于controller层也可以采用这种方式进行重构。

2.5. 数据库中数据的过滤（ctl）（本周）

数据库初始数据的建立与冗余数据的删除。要注意表与表之间外键的关联。

2.6. 数据库表结构的优化与SQL语句性能优化（第二轮）（ctl）

目前比较明显的是“查找某一课程中所有学生所有实验的成绩”这个业务，SQL查找性能很差，考虑性能优化。可先尝试优化与规范其它SQL语句，将方言统一为MySQL。

2.7. 与前端小组以及子功能小组开会（对齐）

2.8. 通知子系统（待定）

考虑之前舍弃的通知子系统是否要继续开发。