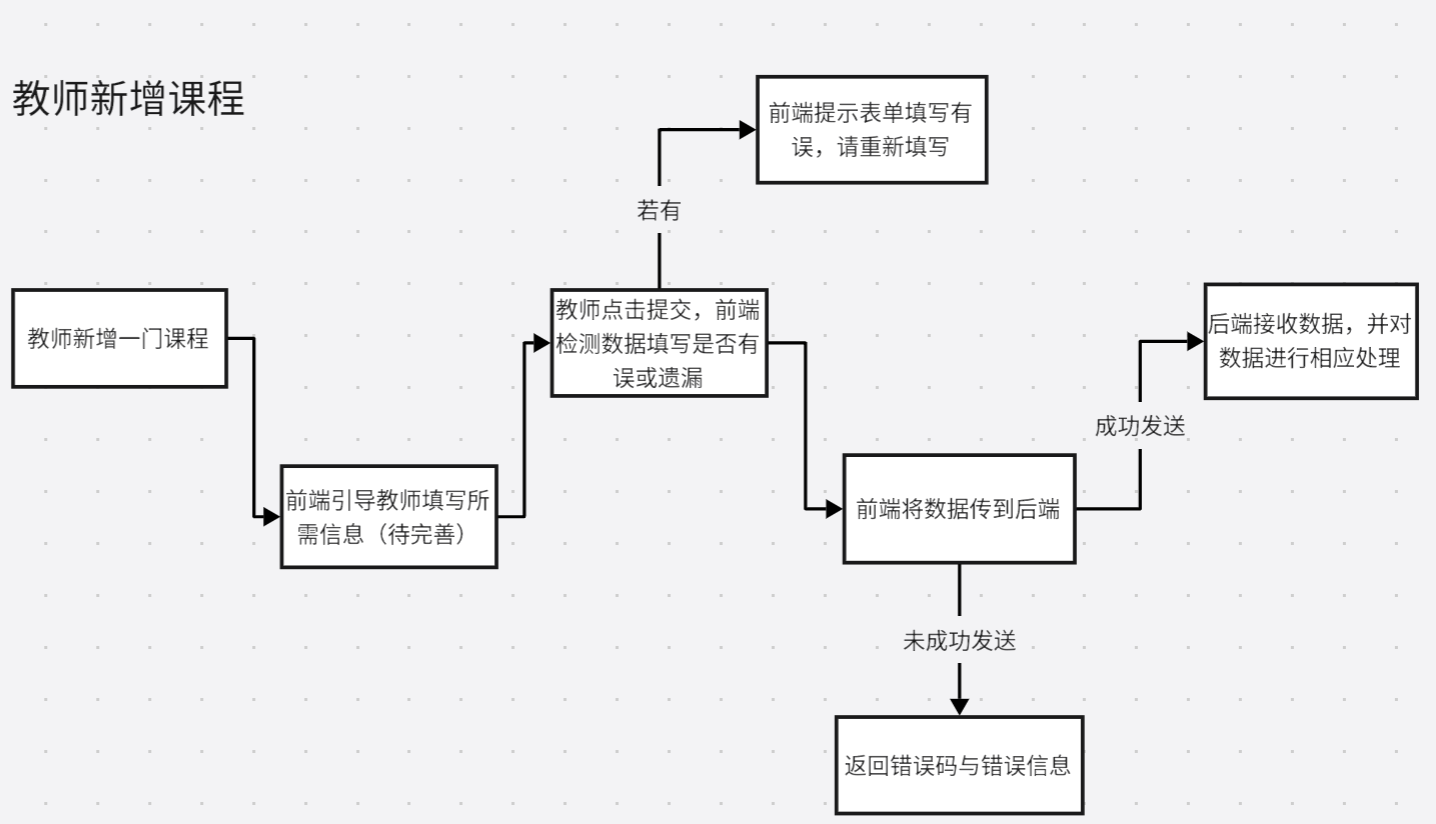
# 开发日志

## 第11-12周

### WZK部分



#### 1.1.1 控制器层

1. postOneCourse方法：
   * 路由：POST /api/addcourse
   * 功能：添加一门课程
   * 主要步骤：
     + 通过JwtInterceptor获取登录的教师的teacherIndex（目前先注释掉，因登录相关逻辑还未实现）。
     + 通过前端传来的newCourseDto类数据，获取courseName、semester、studentList、year等数据（备注：teacherList的初始化逻辑是先加入当前登录并操作该api的教师，但由于teacherIndex还无法获取，故现只简单处理为getTeacherList()）。
     + 调用Service层的insertOneCourse方法进行课程的插入。

#### 1.1.2 业务逻辑层

1. insertOneCourse方法：
   * 功能：添加一门课程
   * 主要步骤：
     + 该方法只是简单的提供了insert方法，避免Controller层和Mapper层直接接触。

#### 1.1.3 测试用例

|  |  |
| --- | --- |
| **Name** | Add a New Course |
| **Summary** | 用户通过 API 向系统中添加一门新课程 |
| **Items to be tested** | 模块：课程管理 |
| **Users** | 教师用户 |
| **Pre-conditions** | 1. 教师已登录系统（基于 JWT 的登录机制假设）。 2. 系统处于运行并可访问的状态。 |
| **Basic Course of Events** | 1. 教师向 /api/addcourse 发送 POST 请求，请求体包含以下数据：  * newCourseDto，包含课程名称（courseName）、学期（semester）、学生列表（studentList）、学年（year）等信息。  1. 控制器层处理请求：  * 使用 JwtInterceptor 提取教师的 teacherIndex（目前暂未实现，代码中已注释）。 * 使用 getTeacherList() 简单初始化 teacherList，加入当前教师（未来需改为动态处理）。 * 将处理后的数据传递给业务逻辑层（Service 层）。  1. 业务逻辑层调用 insertOneCourse 方法处理课程插入逻辑。 2. 课程数据被存储到数据库中。 3. API 返回操作成功的响应和确认信息。 |
| **Input** | 1. 如果 newCourseDto 中缺少必要数据或数据无效：  * API 返回错误响应，并附带相应的错误信息。 |
| **Expected output** | 1. 一门新课程成功添加到系统，并存储在数据库中。 2. 教师收到操作成功的确认响应。 |