

- 后端接口设计文档
 - 1. 认证相关接口
 - 1.1 用户登录
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 1.2 用户注册
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 1.3 退出登录
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 1.4 获取当前用户信息
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 2. 主页相关接口
 - 2.1 获取平台统计数据
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 角色权限说明
 - 通用请求头

- 错误码说明
- 3. 讲师端接口
 - 3.1 获取今日授课列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 3.2 获取讲师授课表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 3.3 获取待评分学员列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 3.4 提交学员评分
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 3.5 批量提交评分
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 3.6 设置评分占比
 - 接口名称
 - 逻辑描述
 - 接口路径

- 请求方式
 - 输入参数
 - 返回值
- 3.7 获取课程成绩统计
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 3.8 获取讲师授课统计
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 4. 员工端接口
 - 4.1 获取今日课程列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 4.2 获取员工课程表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 4.3 获取待自评课程列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 4.4 提交课程自评
 - 接口名称

- 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 4.5 获取员工成绩列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 4.6 获取课程类型成绩分析
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 4.7 获取员工学习进度
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5. 课程大纲制定者端接口
 - 5.1 获取培训计划列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
 - 5.2 创建培训计划
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值

- 5.3 修改培训计划
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.4 删除培训计划
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.5 获取培训计划详情
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.6 为培训计划添加员工
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.7 从培训计划移除员工
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.8 获取课程列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数

- 返回值
- 5.9 创建课程
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.10 修改课程
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.11 删除课程
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.12 获取课程安排列表
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.13 创建课程安排
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.14 修改课程安排
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式

- 输入参数
 - 返回值
- 5.15 删除课程安排
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.16 获取平台数据分析
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.17 获取员工成绩详情
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值
- 5.18 获取课程评价详情
 - 接口名称
 - 逻辑描述
 - 接口路径
 - 请求方式
 - 输入参数
 - 返回值

后端接口设计文档

1. 认证相关接口

1.1 用户登录

接口名称

用户登录接口

逻辑描述

1. 前端提交用户名和密码

2. 后端在 `account` 表中根据 `login_name` 查询账号信息

3. 验证密码哈希值是否匹配（使用安全的哈希算法如 `bcrypt`）

4. 如果验证成功：

根据 `person_id` 关联查询 `person` 表，获取用户姓名和角色

生成 JWT Token（包含 `person_id`, `role`, 过期时间等信息）

返回 Token 和用户基本信息

5. 如果验证失败：

返回错误信息（用户名或密码错误）

6. 异常情况：

账号不存在：返回 "用户名或密码错误"（安全考虑，不暴露账号是否存在）

账号被禁用：返回 "账号已被禁用"

密码错误：返回 "用户名或密码错误"
- ## 接口路径
- POST /api/auth/login
- ## 请求方式
- POST
- ## 输入参数
- ```
{
 "username": "string", // 必填，用户名，对应 account.login_name
 "password": "string" // 必填，密码明文（传输时应使用 HTTPS）
}
```
- ## 参数说明：
- | 参数名      | 类型     | 必填 | 说明           | 数据库字段                       |
|----------|--------|----|--------------|-----------------------------|
| username | string | 是  | 用户名，长度1-20字符 | account.login_name          |
| password | string | 是  | 密码，长度6-20字符  | 需与 account.password_hash 验证 |
- ## 返回值



成功响应（200）：

```
{
 "code": 200,
 "message": "登录成功",
 "data": {
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...", // JWT Token
 "user": {
 "id": 1001, // person.person_id
 "name": "张三", // person.name
 "role": "employee", // person.role 映射后的英文角色
 "roleDisplay": "员工", // person.role 原始中文角色
 "accountId": 2001 // account.account_id
 }
 }
}
```

失败响应（401）：

```
{
 "code": 401,
 "message": "用户名或密码错误",
 "data": null
}
```

其他错误响应（400）：

```
{
 "code": 400,
 "message": "参数错误：用户名不能为空",
 "data": null
}
```

角色映射规则：

| 数据库中的 role (person.role) | 返回的 role | 说明        |
|--------------------------|----------|-----------|
| 员工                       | employee | 员工角色      |
| 讲师                       | teacher  | 讲师角色      |
| 课程大纲制定者                  | planner  | 培训大纲制定者角色 |

# 1.2 用户注册

接口名称

## 用户注册接口

### 逻辑描述

1. 前端提交注册信息（用户名、密码、姓名、角色）
2. 后端验证数据：
  - 检查用户名是否已存在（查询 `account` 表的 `login_name`）
  - 验证密码强度（至少6位）
  - 验证角色是否合法（员工/讲师/课程大纲制定者）
3. 如果验证通过：
  - 生成 `person_id`（自增或雪花算法）
  - 在 `person` 表中插入用户基本信息（`person_id`, `name`, `role`）
  - 对密码进行哈希加密（使用 `bcrypt`）
  - 生成 `account_id`
  - 在 `account` 表中插入账号信息（`account_id`, `person_id`, `login_name`, `password_hash`）
  - 返回成功信息
4. 如果验证失败：
  - 返回具体的错误信息
5. 异常情况：
  - 用户名已存在：返回 "用户名已被使用"
  - 角色不合法：返回 "角色参数错误"
  - 数据库插入失败：事务回滚，返回 "注册失败，请重试"

### 接口路径

```
POST /api/auth/register
```

### 请求方式

POST

### 输入参数

```
{
 "username": "string", // 必填，用户名，对应 account.login_name
 "password": "string", // 必填，密码明文
 "name": "string", // 必填，真实姓名，对应 person.name
 "role": "string" // 必填，角色：employee/teacher/planner
}
```

参数说明：

| 参数名      | 类型     | 必填 | 说明                                               | 数据库字段                        |
|----------|--------|----|--------------------------------------------------|------------------------------|
| username | string | 是  | 用户名，长度3-20字符，仅字母数字下划线                            | account.login_name           |
| password | string | 是  | 密码，长度6-20字符                                      | account.password_hash（加密后存储） |
| name     | string | 是  | 真实姓名，长度2-20字符                                    | person.name                  |
| role     | string | 是  | 角色：employee（员工） / teacher（讲师） / planner（课程大纲制定者） | person.role（存储为中文）           |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "注册成功",
 "data": {
 "personId": 1001, // person.person_id
 "accountId": 2001, // account.account_id
 "username": "zhangsan", // account.login_name
 "name": "张三", // person.name
 "role": "employee", // 角色英文
 "roleDisplay": "员工" // 角色中文
 }
}
```

失败响应（400）：

```
{
 "code": 400,
 "message": "用户名已被使用",
 "data": null
}
```

其他错误示例：

```
{
 "code": 400,
 "message": "密码长度必须在6-20位之间",
}
```

```
"data": null
}
```

## 1.3 退出登录

### 接口名称

退出登录接口

### 逻辑描述

1. 前端提交 Token（在请求头中）
2. 后端验证 Token 有效性
3. 如果需要实现 Token 黑名单机制：
  - 将 Token 加入黑名单（Redis 存储，设置过期时间）
4. 返回成功响应
5. 前端收到响应后清除本地存储的 Token
6. 异常情况：
  - Token 无效或过期：仍返回成功（幂等性）
  - Token 缺失：返回 "未登录"

### 接口路径

```
POST /api/auth/logout
```

### 请求方式

POST

### 输入参数

#### 请求头：

```
Authorization: Bearer <token>
```

#### 请求体：

```
{}
```

 // 空对象，或不传

## 返回值

### 成功响应（200）：

```
{
 "code": 200,
 "message": "退出成功",
 "data": null
}
```

### 未登录响应（401）：

```
{
 "code": 401,
 "message": "未登录或登录已过期",
 "data": null
}
```

---

## 1.4 获取当前用户信息

### 接口名称

获取当前用户信息接口

### 逻辑描述

1. 前端提交 Token（在请求头中）
2. 后端解析并验证 Token
3. 从 Token 中获取 **person\_id**
4. 根据 **person\_id** 查询 **person** 表，获取用户详细信息
5. 根据 **person\_id** 查询 **account** 表，获取账号信息
6. 根据用户角色，额外查询相关信息：
  - **员工角色**：查询参与的培训计划数（**plan\_employee** 表）、已完成课程数（**attendance\_evaluation** 表）
  - **讲师角色**：查询主讲的课程数（**course** 表中 **teacher\_id**）、授课总时长等
  - **课程大纲制定者角色**：查询创建的培训计划数（**training\_plan** 表中 **creator\_id**）
7. 返回用户完整信息
8. 异常情况：
  - Token 无效或过期：返回 "未登录或登录已过期"
  - 用户不存在：返回 "用户不存在"

接口路径

GET /api/auth/current-user

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数：无

返回值

成功响应（200）：

员工角色：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "personId": 1001, // person.person_id
 "name": "张三", // person.name
 "role": "employee", // 角色英文
 "roleDisplay": "员工", // person.role
 "accountId": 2001, // account.account_id
 "username": "zhangsan", // account.login_name
 "statistics": {
 "trainingPlanCount": 2, // 参与的培训计划数量
 "completedCourseCount": 15, // 已完成的课程数量
 "totalCourseCount": 20, // 总课程数量
 "averageScore": 87.5 // 平均得分
 }
 }
}
```

讲师角色：

```
{
 "code": 200,
 "message": "获取成功",
```

```
"data": {
 "personId": 1002, // person.person_id
 "name": "李老师", // person.name
 "role": "teacher", // 角色英文
 "roleDisplay": "讲师", // person.role
 "accountId": 2002, // account.account_id
 "username": "liteacher", // account.login_name
 "statistics": {
 "courseCount": 5, // 主讲的课程数量
 "classCount": 32, // 授课次数 (course_item 中的记录数)
 "studentCount": 45, // 教授的学员总数 (去重)
 "averageTeachingScore": 92.3 // 平均教学评分
 }
}
```

## 课程大纲制定者角色:

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "personId": 1003, // person.person_id
 "name": "王主管", // person.name
 "role": "planner", // 角色英文
 "roleDisplay": "课程大纲制定者", // person.role
 "accountId": 2003, // account.account_id
 "username": "wangplanner", // account.login_name
 "statistics": {
 "planCount": 6, // 创建的培训计划数量
 "totalCourseCount": 28, // 所有培训计划中的课程总数
 "totalStudentCount": 120, // 所有培训计划中的学员总数
 "averagePlanScore": 88.7 // 所有培训计划的平均得分
 }
 }
}
```

## 未登录响应 (401):

```
{
 "code": 401,
 "message": "未登录或登录已过期",
 "data": null
}
```

## 用户不存在响应 (404):

```
{
 "code": 404,
 "message": "用户不存在",
 "data": null
}
```

## 2. 主页相关接口

### 2.1 获取平台统计数据

#### 接口名称

获取平台统计数据接口

#### 逻辑描述

1. 前端请求平台整体统计数据（用于主页展示）
2. 后端查询并统计：
  - **培训课程数**：统计 `course` 表的总记录数
  - **讲师团队数**：统计 `person` 表中 `role='讲师'` 的记录数
  - **培训计划数**：统计 `training_plan` 表的总记录数
  - **平均满意度**：
    - 查询 `attendance_evaluation` 表中所有的加权得分（ $\text{self\_score} * (1 - \text{score\_ratio}) + \text{teacher\_score} * \text{score\_ratio}$ ）
    - 计算平均值并转换为百分比
3. 如果用户已登录，根据用户角色返回个性化统计：
  - **员工**：返回该员工相关的统计
  - **讲师**：返回该讲师相关的统计
  - **课程大纲制定者**：返回全局统计
4. 如果用户未登录，返回全局统计数据
5. 异常情况：
  - 数据库查询失败：返回默认值或缓存数据

#### 接口路径

GET /api/home/statistics

#### 请求方式

GET

#### 输入参数



请求头（可选）：

Authorization: Bearer <token> // 如果已登录则传递

查询参数： 无

返回值

未登录或课程大纲制定者角色（全局统计）成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseCount": 28, // course 表总记录数
 "teacherCount": 12, // person 表中 role='讲师' 的记录数
 "planCount": 6, // training_plan 表总记录数
 "averageSatisfaction": 96, // 平均满意度百分比（计算自 attendance_evaluation）
 "totalStudentCount": 120, // person 表中 role='员工' 的记录数
 "totalClassCount": 156, // plan_course_item 表总记录数（总课次）
 "ongoingPlanCount": 3, // training_plan 中 plan_status='进行中' 的记录数
 "completedPlanCount": 2 // training_plan 中 plan_status='已完成' 的记录数
 }
}
```

员工角色（个性化统计）成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseCount": 28, // 全局课程数
 "teacherCount": 12, // 全局讲师数
 "planCount": 6, // 全局计划数
 "averageSatisfaction": 96, // 全局平均满意度
 "personalStats": {
 "myPlanCount": 2, // 该员工参与的计划数（plan_employee）
 "myTotalCourseCount": 20, // 该员工需要上的总课程数
 "myCompletedCourseCount": 15, // 该员工已完成的课程数（attendance_evaluation 中有评分的）
 "myAverageScore": 87.5, // 该员工的平均得分
 "myTodayCourseCount": 2, // 该员工今日课程数
 "myWeekCourseCount": 8 // 该员工本周课程数
 }
 }
}
```

讲师角色（个性化统计）成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseCount": 28, // 全局课程数
 "teacherCount": 12, // 全局讲师数
 "planCount": 6, // 全局计划数
 "averageSatisfaction": 96, // 全局平均满意度
 "personalStats": {
 "myCourseCount": 5, // 该讲师主讲的课程数 (course.teacher_id)
 "myClassCount": 32, // 该讲师的授课次数
 "myStudentCount": 45, // 该讲师教授的学员总数 (去重)
 "myAverageTeachingScore": 92.3, // 该讲师的平均教学评分
 "myTodayClassCount": 3, // 该讲师今日授课数
 "myWeekClassCount": 12 // 该讲师本周授课数
 }
 }
}
```

## 错误响应 (500):

```
{
 "code": 500,
 "message": "服务器内部错误",
 "data": null
}
```

# 角色权限说明

## 1. 员工 (employee) 权限:

- 可查看: 自己的课程表、今日课程、自己的成绩
- 可操作: 提交课程自评
- 不可访问: 其他员工的信息、讲师评分功能、培训计划管理

## 2. 讲师 (teacher) 权限:

- 可查看: 自己的授课表、今日授课、所教学员的成绩
- 可操作: 给学员评分、设置评分占比
- 不可访问: 其他讲师的信息、培训计划管理、课程管理 (创建/删除课程)

## 3. 课程大纲制定者 (planner) 权限:

- 可查看: 所有培训计划、所有课程、所有数据分析
- 可操作: 创建/修改/删除培训计划、管理课程安排
- 最高权限角色

# 通用请求头

所有需要认证的接口都需要在请求头中携带 Token：

```
Authorization: Bearer <token>
Content-Type: application/json
```

## 错误码说明

| 错误码 | 说明    | 常见场景                 |
|-----|-------|----------------------|
| 200 | 成功    | 请求成功                 |
| 400 | 参数错误  | 必填参数缺失、参数格式错误、参数值不合法 |
| 401 | 未授权   | 未登录、Token无效或过期       |
| 403 | 无权限   | 登录了但权限不足             |
| 404 | 未找到   | 资源不存在                |
| 500 | 服务器错误 | 数据库错误、系统异常           |

## 3. 讲师端接口

### 3.1 获取今日授课列表

#### 接口名称

获取讲师今日授课列表接口

#### 逻辑描述

- 前端请求今日授课信息，携带 Token
- 后端验证 Token，获取 `person_id` 和角色信息
- 权限验证：确认用户角色为"讲师"
- 查询流程：
  - 根据 `person_id` 在 `course` 表中查询该讲师主讲的所有课程（`teacher_id = person_id`）

- 获取这些课程在 `plan_course_item` 表中今日的所有课程安排 (`class_date = 今日日期`)
- 关联查询培训计划信息 (`training_plan` 表)

#### 5. 返回结果:

- 返回今日所有授课安排, 包含课程信息、时间、地点等
- 按上课时间升序排列

#### 6. 异常情况:

- 非讲师角色访问: 返回 403 无权限
- 今日无授课: 返回空数组

## 接口路径

```
GET /api/teacher/today-courses
```

## 请求方式

GET

## 输入参数

## 请求头:

```
Authorization: Bearer <token>
```

查询参数: 无

## 返回值

## 成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "date": "2024-12-24",
 "courseCount": 2,
 "courses": [
 {
 "itemId": 3001, // plan_course_item.item_id
 "courseId": 101, // course.course_id
 "courseName": "船舶结构力学", // course.course_name
 "courseDesc": "学习船舶结构设计...", // course.course_desc
 "courseRequire": "需提前预习第3章", // course.course_require
 "courseClass": "专业技能", // course.course_class
 }
]
 }
}
```

```
 "classDate": "2024-12-24", // plan_course_item.class_date
 "classBeginTime": "08:00:00", // plan_course_item.class_begin_time
 "classEndTime": "10:00:00", // plan_course_item.class_end_time
 "location": "培训楼301", // plan_course_item.location
 "planId": 1, // training_plan.plan_id
 "planName": "2024年度技能提升计划", // training_plan.plan_name
 "studentCount": 25, // 该课次的学员数量（attendance_evaluation表统计）
 "evaluatedCount": 10 // 已评分学员数量
 },
 {
 "itemId": 3002,
 "courseId": 102,
 "courseName": "船舶焊接技术",
 "courseDesc": "掌握船舶焊接方法...",
 "courseRequire": "需携带焊接防护用品",
 "courseClass": "专业技能",
 "classDate": "2024-12-24",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "studentCount": 20,
 "evaluatedCount": 0
 }
]
}
```

### 无授课响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "date": "2024-12-24",
 "courseCount": 0,
 "courses": []
 }
}
```

### 无权限响应 (403):

```
{
 "code": 403,
 "message": "无权限访问, 仅讲师可查看",
 "data": null
}
```

## 3.2 获取讲师授课表

接口名称

获取讲师授课表接口

逻辑描述

1. 前端请求指定时间范围内的授课表，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"讲师"
4. 查询流程：

◦ 根据 `person_id` 在 `course` 表中查询该讲师主讲的所有课程

◦ 获取这些课程在指定时间范围内的所有课程安排（`plan_course_item` 表）

◦ 关联查询课程详情和培训计划信息
5. 数据组织：

◦ 按日期和时间段组织数据

◦ 支持周视图和月视图
6. 异常情况：

◦ 非讲师角色访问：返回 403 无权限

◦ 日期参数错误：返回 400 参数错误

接口路径

GET /api/teacher/schedule

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数：

| 参数名       | 类型     | 必填 | 说明   | 示例         |
|-----------|--------|----|------|------------|
| startDate | string | 是  | 开始日期 | 2024-12-18 |
| endDate   | string | 是  | 结束日期 | 2024-12-24 |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "startDate": "2024-12-18",
 "endDate": "2024-12-24",
 "totalCourses": 8,
 "schedule": [
 {
 "date": "2024-12-18",
 "dayOfWeek": "周一",
 "courses": [
 {
 "itemId": 3001,
 "courseId": 101,
 "courseName": "船舶结构力学",
 "courseDesc": "学习船舶结构设计原理",
 "courseRequire": "需提前预习第3章",
 "courseClass": "专业技能",
 "classBeginTime": "08:00:00",
 "classEndTime": "10:00:00",
 "location": "培训楼301",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "studentCount": 25,
 "evaluatedCount": 25
 },
 {
 "itemId": 3002,
 "courseId": 102,
 "courseName": "船舶焊接技术",
 "courseDesc": "掌握船舶焊接方法",
 "courseRequire": "需携带焊接防护用品",
 "courseClass": "专业技能",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "studentCount": 20,
 "evaluatedCount": 15
 }
]
 },
 {
 "date": "2024-12-19",
 "dayOfWeek": "周二",
 "courses": []
 }
 // ... 其他日期
]
 }
}
```

## 参数错误响应 (400):

```
{
 "code": 400,
 "message": "日期参数错误",
 "data": null
}
```

## 3.3 获取待评分学员列表

### 接口名称

获取待评分学员列表接口

### 逻辑描述

1. 前端请求待评分的学员列表，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"讲师"
4. 查询流程：
  - 根据 `person_id` 在 `course` 表中查询该讲师主讲的所有课程
  - 查询这些课程的所有课程安排（`plan_course_item` 表）
  - 在 `attendance_evaluation` 表中查询所有学员的评价记录
  - 筛选出讲师未评分的记录（`teacher_score IS NULL` 或需要更新）
5. 数据组织：
  - 按课程安排分组
  - 包含学员的自评信息
  - 显示当前的评分占比
6. 异常情况：
  - 非讲师角色访问：返回 403 无权限

### 接口路径

```
GET /api/teacher/pending-evaluations
```

### 请求方式

GET

### 输入参数



请求头:

Authorization: Bearer <token>

查询参数 (可选):

| 参数名      | 类型     | 必填 | 说明                            | 示例      |
|----------|--------|----|-------------------------------|---------|
| courseId | number | 否  | 课程ID, 筛选特定课程                  | 101     |
| status   | string | 否  | 筛选状态: pending (待评分) /all (全部) | pending |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "totalCount": 45,
 "pendingCount": 15,
 "courseItems": [
 {
 "itemId": 3001,
 "courseId": 101,
 "courseName": "船舶结构力学",
 "courseClass": "专业技能",
 "classDate": "2024-12-20",
 "classBeginTime": "08:00:00",
 "classEndTime": "10:00:00",
 "location": "培训楼301",
 "planName": "2024年度技能提升计划",
 "students": [
 {
 "personId": 1001,
 "personName": "张三",
 "selfScore": 85.5,
 "selfComment": "课程内容较难...",
 "teacherScore": null,
 "teacherComment": null,
 "scoreRatio": 0.7,
 "evaluatedAt": null,
 "status": "pending"
 },
 {
 "personId": 1002,
 "personName": "李四",
 "selfScore": 92.0,
 "selfComment": "已充分理解课程内容",
 "teacherScore": 88.0,
 "teacherComment": "表现优秀, 积极回答问题",
 "scoreRatio": 0.7,
 "evaluatedAt": "2024-12-19 15:30:00",
 "status": "evaluated"
 }
]
 }
]
 }
}
```

```
 "evaluatedAt": "2024-12-21 10:30:00",
 "status": "evaluated"
 }
]
}
// ... 其他课程安排
]
}
```

## 3.4 提交学员评分

### 接口名称

提交学员评分接口

### 逻辑描述

- 前端提交评分信息，包括学员ID、课程安排ID、评分、评语等
- 后端验证 Token，获取 `person_id`（讲师ID）和角色信息
- 权限验证：
  - 确认用户角色为"讲师"
  - 验证该课程安排是否为该讲师负责（通过 `course.teacher_id` 验证）
- 数据验证：
  - 评分范围：0-100
  - 评分占比范围：0-1
  - 学员ID和课程安排ID必须存在
- 更新流程：
  - 更新 `attendance_evaluation` 表中对应记录的 `teacher_score`、`teacher_comment`、`score_ratio`
  - 如果使用AI评分功能，则根据评语调用AI接口生成分数
  - 记录评分时间
- 计算加权得分：
  - $weighted\_score = self\_score * (1 - score\_ratio) + teacher\_score * score\_ratio$
- 异常情况：
  - 非讲师角色：返回 403 无权限
  - 非本讲师课程：返回 403 无权限
  - 评分超出范围：返回 400 参数错误
  - 学员或课程不存在：返回 404 未找到

### 接口路径

POST /api/teacher/submit-evaluation

请求方式

POST

输入参数

请求头:

Authorization: Bearer <token>

请求体:

```
{
 "itemId": 3001, // 必填，课程安排ID (plan_course_item.item_id)
 "personId": 1001, // 必填，学员ID (person.person_id)
 "teacherScore": 88.0, // 必填，讲师评分 (0-100)
 "teacherComment": "表现优秀...", // 可选，讲师评语
 "scoreRatio": 0.7, // 必填，讲师评分占比 (0-1)
 "useAI": false // 可选，是否使用AI评分 (根据评语生成分数)
}
```

参数说明:

| 参数名            | 类型     | 必填 | 说明           | 数据库字段                                 |
|----------------|--------|----|--------------|---------------------------------------|
| itemId         | number | 是  | 课程安排ID       | attendance_evaluation.item_id         |
| personId       | number | 是  | 学员ID         | attendance_evaluation.person_id       |
| teacherScore   | number | 是  | 讲师评分 (0-100) | attendance_evaluation.teacher_score   |
| teacherComment | string | 否  | 讲师评语，最长500字  | attendance_evaluation.teacher_comment |
| scoreRatio     | number | 是  | 讲师评分占比 (0-1) | attendance_evaluation.score_ratio     |

| 参数名   | 类型      | 必填 | 说明             | 数据库字段 |
|-------|---------|----|----------------|-------|
| useAI | boolean | 否  | 是否使用AI根据评语生成分数 | -     |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "评分提交成功",
 "data": {
 "itemId": 3001,
 "personId": 1001,
 "personName": "张三",
 "selfScore": 85.5,
 "teacherScore": 88.0,
 "scoreRatio": 0.7,
 "weightedScore": 87.15, // 加权得分 = 85.5 * 0.3 + 88 * 0.7
 "teacherComment": "表现优秀，积极回答问题",
 "evaluatedAt": "2024-12-24 10:30:00"
 }
}
```

使用AI评分成功响应 (200):

```
{
 "code": 200,
 "message": "AI评分完成",
 "data": {
 "itemId": 3001,
 "personId": 1001,
 "personName": "张三",
 "selfScore": 85.5,
 "teacherScore": 86.5, // AI生成的分数
 "scoreRatio": 0.7,
 "weightedScore": 86.2,
 "teacherComment": "表现优秀，积极回答问题",
 "aiGenerated": true, // 标识为AI生成
 "evaluatedAt": "2024-12-24 10:30:00"
 }
}
```

参数错误响应 (400):

```
{
 "code": 400,
 "message": "评分必须在0-100之间",
 "data": null
}
```

### 无权限响应 (403):

```
{
 "code": 403,
 "message": "无权限: 该课程不是您负责的课程",
 "data": null
}
```

### 未找到响应 (404):

```
{
 "code": 404,
 "message": "课程安排或学员不存在",
 "data": null
}
```

---

## 3.5 批量提交评分

### 接口名称

批量提交学员评分接口

### 逻辑描述

1. 前端提交多个学员的评分信息数组
2. 后端验证 Token 和讲师权限
3. 对每个评分记录进行验证和权限检查
4. 批量更新 `attendance_evaluation` 表
5. 使用事务确保数据一致性:
  - 全部成功则提交事务
  - 任一失败则回滚事务
6. 返回每条记录的处理结果

### 接口路径

POST /api/teacher/batch-evaluation

## 请求方式

POST

## 输入参数

## 请求头：

Authorization: Bearer <token>

## 请求体：

```
{
 "evaluations": [
 {
 "itemId": 3001,
 "personId": 1001,
 "teacherScore": 88.0,
 "teacherComment": "表现优秀",
 "scoreRatio": 0.7
 },
 {
 "itemId": 3001,
 "personId": 1002,
 "teacherScore": 92.0,
 "teacherComment": "非常出色",
 "scoreRatio": 0.7
 }
]
}
```

## 返回值

## 成功响应（200）：

```
{
 "code": 200,
 "message": "批量评分成功",
 "data": {
 "totalCount": 2,
 "successCount": 2,
 "failCount": 0,
 "results": [
 {
 "itemId": 3001,
```

```
 "personId": 1001,
 "status": "success",
 "weightedScore": 87.15
 },
 {
 "itemId": 3001,
 "personId": 1002,
 "status": "success",
 "weightedScore": 90.85
 }
]
```

## 3.6 设置评分占比

### 接口名称

设置课程评分占比接口

### 逻辑描述

1. 前端提交课程安排ID和新的评分占比
2. 后端验证 Token 和讲师权限
3. 权限验证：确认该课程为该讲师负责
4. 更新该课程安排下所有学员的评分占比 (`attendance_evaluation.score_ratio`)
5. 重新计算所有学员的加权得分
6. 异常情况：
  - 评分占比超出范围 (0-1)：返回 400
  - 非本讲师课程：返回 403

### 接口路径

```
PUT /api/teacher/score-ratio
```

### 请求方式

PUT

### 输入参数

### 请求头：

Authorization: Bearer <token>

请求体：

```
{
 "itemId": 3001, // 必填，课程安排ID
 "scoreRatio": 0.8 // 必填，讲师评分占比（0-1）
}
```

参数说明：

| 参数名        | 类型     | 必填 | 说明           | 数据库字段                             |
|------------|--------|----|--------------|-----------------------------------|
| itemId     | number | 是  | 课程安排ID       | plan_course_item.item_id          |
| scoreRatio | number | 是  | 讲师评分占比，范围0-1 | attendance_evaluation.score_ratio |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "评分占比设置成功",
 "data": {
 "itemId": 3001,
 "courseName": "船舶结构力学",
 "scoreRatio": 0.8,
 "affectedStudents": 25,
 "updatedAt": "2024-12-24 10:30:00"
 }
}
```

### 3.7 获取课程成绩统计

接口名称

获取讲师课程成绩统计接口

逻辑描述



- 1. 前端请求指定课程的成绩统计信息
- 2. 后端验证 Token 和讲师权限
- 3. 权限验证：确认该课程为该讲师负责
- 4. 查询流程：
  - 查询该课程的所有课程安排（plan\_course\_item 表）
  - 查询所有学员的评价记录（attendance\_evaluation 表）
  - 计算统计数据：平均分、最高分、最低分、及格率等
- 5. 数据组织：
  - 按课次统计
  - 按学员统计
  - 整体统计
- 6. 支持数据可视化所需的格式

接口路径

GET /api/teacher/course-statistics

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数：

| 参数名      | 类型     | 必填 | 说明   | 示例  |
|----------|--------|----|------|-----|
| courseId | number | 是  | 课程ID | 101 |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseId": 101,
```

```

"courseName": "船舶结构力学",
"courseClass": "专业技能",
"teacherName": "李老师",
"statistics": {
 "totalClasses": 8, // 总课次数
 "totalStudents": 25, // 总学员数
 "averageScore": 87.5, // 平均分
 "maxScore": 98.5, // 最高分
 "minScore": 65.0, // 最低分
 "passRate": 96.0, // 及格率（60分及以上）
 "excellentRate": 72.0, // 优秀率（85分及以上）
 "scoreDistribution": { // 分数分布
 "0-59": 1,
 "60-69": 2,
 "70-79": 5,
 "80-89": 10,
 "90-100": 7
 }
},
"classStat": [
 {
 "itemId": 3001,
 "classDate": "2024-12-18",
 "location": "培训楼301",
 "studentCount": 25,
 "averageScore": 86.5,
 "evaluatedCount": 25
 }
 // ... 其他课次
],
"studentScores": [
 {
 "personId": 1001,
 "personName": "张三",
 "classCount": 8, // 上课次数
 "averageScore": 87.5, // 该学员在本课程的平均分
 "latestScore": 88.0, // 最近一次得分
 "trend": "up" // 趋势: up/down/stable
 }
 // ... 其他学员
]
}

```

### 无权限响应 (403):

```

{
 "code": 403,
 "message": "无权限: 该课程不是您负责的课程",
 "data": null
}

```

## 3.8 获取讲师授课统计

## 接口名称

获取讲师整体授课统计接口

## 逻辑描述

- 前端请求讲师的整体授课统计信息
- 后端验证 Token 和讲师权限
- 统计该讲师的所有授课数据：
  - 主讲课程数量
  - 总授课次数
  - 教授的学员总数
  - 所有课程的平均教学评分
  - 各课程类型的分布
- 时间范围统计：
  - 支持按时间范围筛选（本月、本学期、本年度等）

## 接口路径

GET /api/teacher/teaching-statistics

## 请求方式

GET

## 输入参数

### 请求头:

Authorization: Bearer <token>

### 查询参数 (可选):

| 参数名       | 类型     | 必填 | 说明   | 示例         |
|-----------|--------|----|------|------------|
| startDate | string | 否  | 开始日期 | 2024-01-01 |
| endDate   | string | 否  | 结束日期 | 2024-12-31 |

## 返回值

### 成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "teacherId": 1002,
 "teacherName": "李老师",
 "period": {
 "startDate": "2024-01-01",
 "endDate": "2024-12-31"
 },
 "statistics": {
 "courseCount": 5, // 主讲课程数
 "classCount": 32, // 总授课次数
 "studentCount": 45, // 教授学员总数（去重）
 "totalHours": 64, // 总授课时长（小时）
 "averageTeachingScore": 92.3, // 平均教学评分
 "evaluationRate": 98.5, // 评分完成率
 "courseTypeDistribution": [// 课程类型分布
 {
 "courseClass": "专业技能",
 "courseCount": 3,
 "averageScore": 91.5
 },
 {
 "courseClass": "安全培训",
 "courseCount": 2,
 "averageScore": 93.8
 }
]
 },
 "recentClasses": [// 最近授课
 {
 "itemId": 3001,
 "courseName": "船舶结构力学",
 "classDate": "2024-12-24",
 "studentCount": 25,
 "evaluatedCount": 25,
 "averageScore": 87.5
 }
 // ... 最近5次授课
]
 }
}
```

## 4. 员工端接口

### 4.1 获取今日课程列表

#### 接口名称

获取员工今日课程列表接口

## 逻辑描述

1. 前端请求今日课程信息，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"员工"
4. 查询流程：
  - 根据 `person_id` 在 `plan_employee` 表中查询该员工参与的所有培训计划
  - 获取这些培训计划在 `plan_course_item` 表中今日的所有课程安排（`class_date = 今日日期`）
  - 关联查询课程信息（`course` 表）和讲师信息（`person` 表）
  - 查询员工对这些课程的评价状态（`attendance_evaluation` 表）
5. 返回结果：
  - 返回今日所有课程安排，包含课程信息、时间、地点、讲师等
  - 标注是否已评价
  - 按上课时间升序排列
6. 异常情况：
  - 非员工角色访问：返回 403 无权限
  - 今日无课程：返回空数组

## 接口路径

```
GET /api/employee/today-courses
```

## 请求方式

GET

## 输入参数

## 请求头：

```
Authorization: Bearer <token>
```

查询参数：无

## 返回值

成功响应（200）：

```

{
 "code": 200,
 "message": "获取成功",
 "data": {
 "date": "2024-12-24",
 "courseCount": 2,
 "courses": [
 {
 "itemId": 3001, // plan_course_item.item_id
 "courseId": 101, // course.course_id
 "courseName": "船舶结构力学", // course.course_name
 "courseDesc": "学习船舶结构设计...", // course.course_desc
 "courseRequire": "需提前预习第3章", // course.course_require
 "courseClass": "专业技能", // course.course_class
 "classDate": "2024-12-24", // plan_course_item.class_date
 "classBeginTime": "08:00:00", // plan_course_item.class_begin_time
 "classEndTime": "10:00:00", // plan_course_item.class_end_time
 "location": "培训楼301", // plan_course_item.location
 "planId": 1, // training_plan.plan_id
 "planName": "2024年度技能提升计划", // training_plan.plan_name
 "teacherId": 1002, // course.teacher_id
 "teacherName": "李老师", // person.name (讲师)
 "hasEvaluated": false, // 是否已自评 (attendance_evaluation表查询)
 "status": "待上课" // 课程状态: 待上课/已完成
 },
 {
 "itemId": 3002,
 "courseId": 102,
 "courseName": "船舶焊接技术",
 "courseDesc": "掌握船舶焊接方法...",
 "courseRequire": "需携带焊接防护用品",
 "courseClass": "专业技能",
 "classDate": "2024-12-24",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "teacherId": 1003,
 "teacherName": "王工",
 "hasEvaluated": false,
 "status": "待上课"
 }
]
 }
}

```

## 无课程响应 (200):

```

{
 "code": 200,
 "message": "获取成功",
 "data": {
 "date": "2024-12-24",
 "courseCount": 0,
 "courses": []
 }
}

```

```
}
}
```

### 无权限响应 (403):

```
{
 "code": 403,
 "message": "无权限访问，仅员工可查看",
 "data": null
}
```

## 4.2 获取员工课程表

### 接口名称

获取员工课程表接口

### 逻辑描述

1. 前端请求指定时间范围内的课程表，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"员工"
4. 查询流程：
  - 根据 `person_id` 在 `plan_employee` 表中查询该员工参与的所有培训计划
  - 获取这些培训计划在指定时间范围内的所有课程安排 (`plan_course_item` 表)
  - 关联查询课程详情、讲师信息和培训计划信息
  - 查询评价状态 (`attendance_evaluation` 表)
5. 数据组织：
  - 按日期和时间段组织数据
  - 支持周视图和月视图
6. 异常情况：
  - 非员工角色访问：返回 403 无权限
  - 日期参数错误：返回 400 参数错误

### 接口路径

```
GET /api/employee/schedule
```

### 请求方式

GET

输入参数

请求头:

Authorization: Bearer <token>

查询参数:

| 参数名       | 类型     | 必填 | 说明   | 示例         |
|-----------|--------|----|------|------------|
| startDate | string | 是  | 开始日期 | 2024-12-18 |
| endDate   | string | 是  | 结束日期 | 2024-12-24 |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "startDate": "2024-12-18",
 "endDate": "2024-12-24",
 "totalCourses": 6,
 "schedule": [
 {
 "date": "2024-12-18",
 "dayOfWeek": "周一",
 "courses": [
 {
 "itemId": 3001,
 "courseId": 101,
 "courseName": "船舶结构力学",
 "courseDesc": "学习船舶结构设计原理",
 "courseRequire": "需提前预习第3章",
 "courseClass": "专业技能",
 "classBeginTime": "08:00:00",
 "classEndTime": "10:00:00",
 "location": "培训楼301",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "teacherId": 1002,
 "teacherName": "李老师",
 "hasEvaluated": true,
 "status": "已完成"
 },
 {
 "itemId": 3002,
 "courseId": 102,
 "courseName": "船舶焊接技术",
```



```
 "courseDesc": "掌握船舶焊接方法",
 "courseRequire": "需携带焊接防护用品",
 "courseClass": "专业技能",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "teacherId": 1003,
 "teacherName": "王工",
 "hasEvaluated": false,
 "status": "已完成"
 }
]
},
{
 "date": "2024-12-19",
 "dayOfWeek": "周二",
 "courses": []
}
// ... 其他日期
]
}
```

#### 参数错误响应 (400):

```
{
 "code": 400,
 "message": "日期参数错误",
 "data": null
}
```

## 4.3 获取待自评课程列表

### 接口名称

获取待自评课程列表接口

### 逻辑描述

1. 前端请求待自评的课程列表，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"员工"
4. 查询流程：
  - 根据 `person_id` 查询该员工的所有课程安排
  - 在 `attendance_evaluation` 表中查询评价记录
  - 筛选出未自评的课程 (`self_score IS NULL` 或 `self_comment IS NULL`)

- 只返回已上完的课程（课程日期 < 今日）

5. 数据组织：

- 按课程日期倒序排列（最近的课程在前）

6. 异常情况：

- 非员工角色访问：返回 403 无权限

接口路径

GET /api/employee/pending-evaluations

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数（可选）：

| 参数名    | 类型     | 必填 | 说明                        | 示例      |
|--------|--------|----|---------------------------|---------|
| status | string | 否  | 筛选状态：pending（待自评）/all（全部） | pending |
| limit  | number | 否  | 返回数量限制                    | 10      |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "totalCount": 15,
 "pendingCount": 3,
 "courses": [
 {
 "itemId": 3001,
 "courseId": 101,
 "courseName": "船舶结构力学",
 "courseDesc": "学习船舶结构设计原理",
 "courseRequire": "需提前预习第3章",

```

```
"courseClass": "专业技能",
"classDate": "2024-12-20",
"classBeginTime": "08:00:00",
"classEndTime": "10:00:00",
"location": "培训楼301",
"planId": 1,
"planName": "2024年度技能提升计划",
"teacherId": 1002,
"teacherName": "李老师",
"hasEvaluated": false,
"evaluationStatus": "pending" // 评价状态: pending/evaluated
},
{
 "itemId": 3002,
 "courseId": 102,
 "courseName": "船舶焊接技术",
 "courseDesc": "掌握船舶焊接方法",
 "courseRequire": "需携带焊接防护用品",
 "courseClass": "专业技能",
 "classDate": "2024-12-19",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "teacherId": 1003,
 "teacherName": "王工",
 "hasEvaluated": false,
 "evaluationStatus": "pending"
}
]
}
```

## 4.4 提交课程自评

### 接口名称

提交课程自评接口

### 逻辑描述

- 前端提交自评信息，包括课程安排ID、学习心得、理解程度等
- 后端验证 Token，获取 `person_id` 和角色信息
- 权限验证：
  - 确认用户角色为"员工"
  - 验证该课程是否为该员工的课程（通过 `plan_employee` 验证）
- 数据验证：
  - 课程安排ID必须存在
  - 自评内容不能为空

- 课程必须已上完（课程日期 < 今日）

#### 5. AI评分流程：

- 将自评内容发送给AI接口（DeepSeek API）
- AI根据内容分析学习掌握程度，生成0-100分的评分
- 分数越高表示掌握度越高

#### 6. 更新流程：

- 在 `attendance_evaluation` 表中插入或更新记录
- 存储 `self_score`（AI生成）和 `self_comment`（用户填写）
- 如果已有讲师评分，重新计算加权得分

#### 7. 异常情况：

- 非员工角色：返回 403 无权限
- 非本人课程：返回 403 无权限
- 课程未上完：返回 400 课程尚未开始
- 自评内容为空：返回 400 参数错误
- AI服务异常：使用默认评分算法

## 接口路径

```
POST /api/employee/submit-evaluation
```

## 请求方式

POST

## 输入参数

## 请求头：

```
Authorization: Bearer <token>
```

## 请求体：

```
{
 "itemId": 3001, // 必填，课程安排ID (plan_course_item.item_id)
 "selfComment": "通过本次课程...", // 必填，自评内容/学习心得
 "understanding": 5, // 可选，理解程度 (1-5)
 "difficulty": 3, // 可选，难度感受 (1-5)
 "satisfaction": 5 // 可选，满意度 (1-5)
}
```

## 参数说明：

| 参数名           | 类型     | 必填 | 说明                 | 数据库字段                              |
|---------------|--------|----|--------------------|------------------------------------|
| itemId        | number | 是  | 课程安排ID             | attendance_evaluation.item_id      |
| selfComment   | string | 是  | 自评内容，最少50字，最长1000字 | attendance_evaluation.self_comment |
| understanding | number | 否  | 理解程度（1-5），辅助AI评分   | -                                  |
| difficulty    | number | 否  | 难度感受（1-5），辅助AI评分   | -                                  |
| satisfaction  | number | 否  | 满意度（1-5），辅助AI评分    | -                                  |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "自评提交成功",
 "data": {
 "itemId": 3001,
 "courseId": 101,
 "courseName": "船舶结构力学",
 "selfScore": 87.5, // AI生成的自评分
 (attendance_evaluation.self_score)
 "selfComment": "通过本次课程...", // 用户填写的自评内容
 "aiAnalysis": { // AI分析结果（可选）
 "keyPoints": ["掌握了基本概念", "理解了应用场景"],
 "weakPoints": ["计算部分需加强"],
 "suggestion": "建议多做练习题"
 },
 "teacherScore": null, // 讲师评分（如果已评）
 "weightedScore": null, // 加权得分（如果讲师已评分）
 "evaluatedAt": "2024-12-24 10:30:00"
 }
}
```

参数错误响应（400）：

```
{
 "code": 400,
```

```
"message": "自评内容不能少于50字",
"data": null
}
```

### 课程未开始响应 (400):

```
{
 "code": 400,
 "message": "课程尚未开始，暂不能自评",
 "data": null
}
```

### 无权限响应 (403):

```
{
 "code": 403,
 "message": "无权限：该课程不属于您的课程安排",
 "data": null
}
```

---

## 4.5 获取员工成绩列表

### 接口名称

获取员工成绩列表接口

### 逻辑描述

1. 前端请求成绩列表，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"员工"
4. 查询流程：
  - 使用视图 `v_employee_item_score` 查询该员工的所有课程得分
  - 按课程日期倒序排列
  - 计算加权得分： $\text{weighted\_score} = \text{self\_score} * (1 - \text{score\_ratio}) + \text{teacher\_score} * \text{score\_ratio}$
5. 数据统计：
  - 计算平均分
  - 统计已完成课程数
  - 统计待评价课程数
6. 支持筛选：

- 按培训计划筛选
- 按课程类型筛选
- 按日期范围筛选

7. 异常情况：

- 非员工角色访问：返回 403 无权限

接口路径

GET /api/employee/scores

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数（可选）：

| 参数名         | 类型     | 必填 | 说明       | 示例         |
|-------------|--------|----|----------|------------|
| planId      | number | 否  | 培训计划ID筛选 | 1          |
| courseClass | string | 否  | 课程类型筛选   | 专业技能       |
| startDate   | string | 否  | 开始日期     | 2024-01-01 |
| endDate     | string | 否  | 结束日期     | 2024-12-31 |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "statistics": {
 "totalCourses": 20, // 总课程数
 "completedCourses": 15, // 已完成课程数（有成绩的）
 "pendingEvaluation": 3, // 待自评课程数
 "averageScore": 87.5, // 平均加权得分
 }
 }
}
```

```

 "maxScore": 98.5, // 最高分
 "minScore": 72.0 // 最低分
 },
 "scores": [
 {
 "itemId": 3001, // plan_course_item.item_id
 "classDate": "2024-12-20", // plan_course_item.class_date
 "classBeginTime": "08:00:00", // plan_course_item.class_begin_time
 "classEndTime": "10:00:00", // plan_course_item.class_end_time
 "location": "培训楼301", // plan_course_item.location
 "planId": 1, // training_plan.plan_id
 "planName": "2024年度技能提升计划", // training_plan.plan_name
 "courseId": 101, // course.course_id
 "courseName": "船舶结构力学", // course.course_name
 "courseClass": "专业技能", // course.course_class
 "teacherName": "李老师", // person.name (讲师)
 "selfScore": 85.5, // attendance_evaluation.self_score
 "teacherScore": 88.0, // attendance_evaluation.teacher_score
 "scoreRatio": 0.7, // attendance_evaluation.score_ratio
 "weightedScore": 87.15, // 加权得分 = 85.5 * 0.3 + 88 * 0.7
 "selfComment": "已充分理解...", // attendance_evaluation.self_comment
 "teacherComment": "表现优秀", // attendance_evaluation.teacher_comment
 "hasEvaluated": true, // 是否已自评
 "hasTeacherScore": true // 是否已有讲师评分
 },
 {
 "itemId": 3002,
 "classDate": "2024-12-19",
 "classBeginTime": "14:00:00",
 "classEndTime": "16:00:00",
 "location": "实训车间A",
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "courseId": 102,
 "courseName": "船舶焊接技术",
 "courseClass": "专业技能",
 "teacherName": "王工",
 "selfScore": 90.0,
 "teacherScore": null,
 "scoreRatio": 0.7,
 "weightedScore": null, // 讲师未评分，加权得分为null
 "selfComment": "实操部分很有收获...",
 "teacherComment": null,
 "hasEvaluated": true,
 "hasTeacherScore": false
 }
 // ... 更多成绩记录
]
}
}

```

## 4.6 获取课程类型成绩分析

### 接口名称

获取员工课程类型成绩分析接口



逻辑描述

- 1. 前端请求课程类型维度的成绩分析，用于雷达图展示
- 2. 后端验证 Token，获取 `person_id` 和角色信息
- 3. 权限验证：确认用户角色为"员工"
- 4. 查询流程：
  - 使用视图 `v_employee_course_score` 查询该员工的课程类型综合得分
  - 按"员工 + 课程类型"维度汇总平均加权得分
- 5. 数据组织：
  - 返回每个课程类型的平均分
  - 返回每个类型的课程数量
  - 格式化为雷达图所需的数据结构
- 6. 异常情况：
  - 非员工角色访问：返回 403 无权限
  - 无成绩数据：返回空数组

接口路径

GET /api/employee/course-type-scores

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数： 无

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "personId": 1001,
 "personName": "张三",
```

```

"courseTypeScores": [
 {
 "courseClass": "专业技能", // course.course_class
 "avgWeightedScore": 87.5, // 该类型课程的平均加权得分
 "courseCount": 8, // 该类型的课程数量
 "completedCount": 8, // 已完成的课程数量
 "maxScore": 95.0, // 该类型的最高分
 "minScore": 78.0 // 该类型的最低分
 },
 {
 "courseClass": "安全培训",
 "avgWeightedScore": 92.3,
 "courseCount": 4,
 "completedCount": 4,
 "maxScore": 98.0,
 "minScore": 85.0
 },
 {
 "courseClass": "管理能力",
 "avgWeightedScore": 85.8,
 "courseCount": 3,
 "completedCount": 2,
 "maxScore": 90.0,
 "minScore": 82.0
 },
 {
 "courseClass": "团队协作",
 "avgWeightedScore": 88.5,
 "courseCount": 5,
 "completedCount": 5,
 "maxScore": 92.0,
 "minScore": 84.0
 }
],
"radarData": { // 雷达图专用数据格式
 "indicators": [
 { "name": "专业技能", "max": 100 },
 { "name": "安全培训", "max": 100 },
 { "name": "管理能力", "max": 100 },
 { "name": "团队协作", "max": 100 }
],
 "values": [87.5, 92.3, 85.8, 88.5]
}
}
}

```

## 无数据响应 (200):

```

{
 "code": 200,
 "message": "暂无成绩数据",
 "data": {
 "personId": 1001,
 "personName": "张三",
 "courseTypeScores": [],
 "radarData": {
 "indicators": [],
 "values": []
 }
 }
}

```

```
}
}
```

## 4.7 获取员工学习进度

### 接口名称

获取员工学习进度接口

### 逻辑描述

1. 前端请求学习进度信息，携带 Token
2. 后端验证 Token，获取 `person_id` 和角色信息
3. 权限验证：确认用户角色为"员工"
4. 查询流程：
  - 查询员工参与的所有培训计划（`plan_employee` 表）
  - 统计每个培训计划的课程总数和完成情况
  - 计算完成百分比
  - 查询最近上课记录
5. 数据统计：
  - 总体学习进度
  - 各培训计划的进度
  - 最近学习记录
6. 异常情况：
  - 非员工角色访问：返回 403 无权限

### 接口路径

```
GET /api/employee/learning-progress
```

### 请求方式

GET

### 输入参数

### 请求头：

Authorization: Bearer <token>

查询参数：无

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "personId": 1001,
 "personName": "张三",
 "overallProgress": {
 "totalPlans": 2, // 参与的培训计划总数
 "totalCourses": 20, // 总课程数
 "completedCourses": 15, // 已完成课程数
 "evaluatedCourses": 12, // 已自评课程数
 "progressPercentage": 75.0, // 完成百分比
 "averageScore": 87.5 // 平均分
 },
 "planProgress": [
 {
 "planId": 1,
 "planName": "2024年度技能提升计划",
 "planStatus": "进行中",
 "startDate": "2024-01-01",
 "endDate": "2024-12-31",
 "totalCourses": 15,
 "completedCourses": 12,
 "evaluatedCourses": 10,
 "progressPercentage": 80.0,
 "averageScore": 88.2
 },
 {
 "planId": 2,
 "planName": "安全培训专项计划",
 "planStatus": "进行中",
 "startDate": "2024-06-01",
 "endDate": "2024-08-31",
 "totalCourses": 5,
 "completedCourses": 3,
 "evaluatedCourses": 2,
 "progressPercentage": 60.0,
 "averageScore": 85.5
 }
],
 "recentCourses": [
 {
 "itemId": 3001,
 "courseName": "船舶结构力学",
 "classDate": "2024-12-20",
 "weightedScore": 87.15,
 "hasEvaluated": true
 }
]
 }
}
```

```
{
 "itemId": 3002,
 "courseName": "船舶焊接技术",
 "classDate": "2024-12-19",
 "weightedScore": 90.0,
 "hasEvaluated": true
}
// ... 最近5条学习记录
]
}
}
```

## 5. 课程大纲制定者端接口

### 5.1 获取培训计划列表

#### 接口名称

获取培训计划列表接口

#### 逻辑描述

1. 前端请求培训计划列表，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
3. 查询 `training_plan` 表，获取所有培训计划
4. 关联查询 `person` 表，获取制定人姓名
5. 关联查询 `plan_employee` 表，统计每个计划的参与员工数
6. 关联查询 `plan_course_item` 表，统计每个计划的课程数
7. 支持分页、筛选（按状态、按时间范围）、排序
8. 返回计划列表及统计信息
9. 异常情况：
  - 无权限：返回 "无权限访问"
  - 查询失败：返回 "获取失败"

#### 接口路径

GET /api/planner/plans

#### 请求方式

GET

输入参数

请求头:

Authorization: Bearer <token>

查询参数:

| 参数名       | 类型     | 必填 | 说明                                                               |
|-----------|--------|----|------------------------------------------------------------------|
| page      | number | 否  | 页码，默认1                                                           |
| pageSize  | number | 否  | 每页数量，默认10                                                        |
| status    | string | 否  | 计划状态筛选：规划中/进行中/已完成                                               |
| startDate | string | 否  | 开始日期筛选（YYYY-MM-DD）                                               |
| endDate   | string | 否  | 结束日期筛选（YYYY-MM-DD）                                               |
| keyword   | string | 否  | 计划名称关键词搜索                                                        |
| sortBy    | string | 否  | 排序字段：plan_start_datetime/plan_end_datetime，默认plan_start_datetime |
| sortOrder | string | 否  | 排序方向：asc/desc，默认desc                                             |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "total": 25,
 "page": 1,
 "pageSize": 10,
 "list": [
 {
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "planStatus": "进行中",
 "planStartDatetime": "2024-01-01 09:00:00",
 "planEndDatetime": "2024-06-30 18:00:00",
 "creatorId": 2001,
 "creatorName": "张主管",
 "employeeCount": 50,
 "courseCount": 12
 }
]
 }
}
```

```
}
}
```

参数说明（数据库字段映射）：

| 前端字段              | 数据库字段                             | 说明     |
|-------------------|-----------------------------------|--------|
| planId            | training_plan.plan_id             | 培训计划ID |
| planName          | training_plan.plan_name           | 培训计划名称 |
| planStatus        | training_plan.plan_status         | 计划状态   |
| planStartDatetime | training_plan.plan_start_datetime | 开始时间   |
| planEndDatetime   | training_plan.plan_end_datetime   | 结束时间   |
| creatorId         | training_plan.creator_id          | 制定人ID  |
| creatorName       | person.name                       | 制定人姓名  |
| employeeCount     | COUNT(plan_employee.person_id)    | 参与员工数  |
| courseCount       | COUNT(plan_course_item.item_id)   | 课程数    |

无权限响应（403）：

```
{
 "code": 403,
 "message": "无权限访问",
 "data": null
}
```

## 5.2 创建培训计划

接口名称

创建培训计划接口

逻辑描述

- 1. 前端提交培训计划信息，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可创建）
- 3. 验证数据合法性：
  - 计划名称不为空，长度1-50字符
  - 开始时间 < 结束时间

- 计划状态为合法值（规划中/进行中/已完成）
- 生成 plan\_id（自增）
  - 将 creator\_id 设置为当前登录用户的 person\_id
  - 插入 training\_plan 表
  - 返回新创建的计划信息
  - 异常情况：
    - 参数验证失败：返回具体错误信息
    - 数据库插入失败：返回 "创建失败"

接口路径

POST /api/planner/plans

请求方式

POST

输入参数

请求头:

Authorization: Bearer <token>

请求体:

```
{
 "planName": "string", // 必填，计划名称
 "planStatus": "string", // 必填，计划状态：规划中/进行中/已完成
 "planStartDatetime": "string", // 必填，开始时间（YYYY-MM-DD HH:mm:ss）
 "planEndDatetime": "string" // 必填，结束时间（YYYY-MM-DD HH:mm:ss）
}
```

参数说明:

| 参数名        | 类型     | 必填 | 说明               | 数据库字段                     |
|------------|--------|----|------------------|---------------------------|
| planName   | string | 是  | 计划名称，长度1-50字符    | training_plan.plan_name   |
| planStatus | string | 是  | 计划状态：规划中/进行中/已完成 | training_plan.plan_status |



| 参数名               | 类型     | 必填 | 说明                                | 数据库字段                             |
|-------------------|--------|----|-----------------------------------|-----------------------------------|
| planStartDatetime | string | 是  | 开始时间，格式<br>YYYY-MM-DD<br>HH:mm:ss | training_plan.plan_start_datetime |
| planEndDatetime   | string | 是  | 结束时间，格式<br>YYYY-MM-DD<br>HH:mm:ss | training_plan.plan_end_datetime   |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "创建成功",
 "data": {
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "planStatus": "规划中",
 "planStartDatetime": "2024-01-01 09:00:00",
 "planEndDatetime": "2024-06-30 18:00:00",
 "creatorId": 2001,
 "creatorName": "张主管"
 }
}
```

参数错误响应（400）：

```
{
 "code": 400,
 "message": "开始时间必须早于结束时间",
 "data": null
}
```

## 5.3 修改培训计划

接口名称

修改培训计划接口

逻辑描述

- 1. 前端提交修改的培训计划信息，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可修改）
- 3. 验证 plan\_id 是否存在
- 4. 验证数据合法性（同创建接口）
- 5. 更新 training\_plan 表对应记录
- 6. 返回更新后的计划信息
- 7. 异常情况：
  - 计划不存在：返回 "培训计划不存在"
  - 参数验证失败：返回具体错误信息

接口路径

```
PUT /api/planner/plans/:planId
```

请求方式

PUT

输入参数

请求头：

```
Authorization: Bearer <token>
```

路径参数：

| 参数名    | 类型     | 说明     | 数据库字段                 |
|--------|--------|--------|-----------------------|
| planId | number | 培训计划ID | training_plan.plan_id |

请求体：

```
{
 "planName": "string", // 可选，计划名称
 "planStatus": "string", // 可选，计划状态
 "planStartDatetime": "string", // 可选，开始时间
 "planEndDatetime": "string" // 可选，结束时间
}
```

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "修改成功",
 "data": {
 "planId": 1001,
 "planName": "2024年新员工培训计划（修订版）",
 "planStatus": "进行中",
 "planStartDatetime": "2024-01-01 09:00:00",
 "planEndDatetime": "2024-06-30 18:00:00",
 "creatorId": 2001,
 "creatorName": "张主管"
 }
}
```

计划不存在响应（404）：

```
{
 "code": 404,
 "message": "培训计划不存在",
 "data": null
}
```

## 5.4 删除培训计划

接口名称

删除培训计划接口

逻辑描述

- 前端提交要删除的计划ID，携带 Token
- 后端验证用户身份和权限（仅课程大纲制定者可删除）
- 验证 plan\_id 是否存在
- 检查是否存在关联数据：
  - 检查 plan\_course\_item 表，是否有课程安排
  - 检查 plan\_employee 表，是否有关联员工
- 如果存在关联数据，返回错误提示
- 如果无关联数据，开启事务：
  - 删除 training\_plan 表对应记录
- 返回删除成功信息
- 异常情况：
  - 计划不存在：返回 "培训计划不存在"

- 存在关联数据：返回 "无法删除，存在关联的课程或员工"

## 接口路径

```
DELETE /api/planner/plans/:planId
```

## 请求方式

DELETE

## 输入参数

### 请求头：

```
Authorization: Bearer <token>
```

### 路径参数：

| 参数名    | 类型     | 说明     | 数据库字段                 |
|--------|--------|--------|-----------------------|
| planId | number | 培训计划ID | training_plan.plan_id |

## 返回值

### 成功响应（200）：

```
{
 "code": 200,
 "message": "删除成功",
 "data": null
}
```

### 存在关联数据响应（400）：

```
{
 "code": 400,
 "message": "无法删除，该计划下存在课程安排或关联员工，请先删除相关数据",
 "data": null
}
```

# 5.5 获取培训计划详情

## 接口名称

获取培训计划详情接口

## 逻辑描述

- 1. 前端请求指定计划的详细信息，携带 Token
- 2. 后端验证用户身份和权限
- 3. 根据 plan\_id 查询 training\_plan 表
- 4. 关联查询制定人信息（person 表）
- 5. 查询该计划的所有课程安排（plan\_course\_item 表）
- 6. 查询该计划的所有参与员工（plan\_employee 表，关联 person 表获取姓名）
- 7. 返回计划完整信息
- 8. 异常情况：
  - 计划不存在：返回 "培训计划不存在"

## 接口路径

GET /api/planner/plans/:planId

## 请求方式

GET

## 输入参数

## 请求头：

Authorization: Bearer <token>

## 路径参数：

| 参数名    | 类型     | 说明     | 数据库字段                 |
|--------|--------|--------|-----------------------|
| planId | number | 培训计划ID | training_plan.plan_id |

## 返回值

## 成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "planStatus": "进行中",
 "planStartDatetime": "2024-01-01 09:00:00",
 "planEndDatetime": "2024-06-30 18:00:00",
 "creatorId": 2001,
 "creatorName": "张主管",
 "courseItems": [
 {
 "itemId": 10001,
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "classDate": "2024-01-15",
 "classBeginTime": "09:00:00",
 "classEndTime": "11:00:00",
 "location": "培训室A"
 }
],
 "employees": [
 {
 "personId": 3001,
 "name": "王员工"
 }
]
 }
}
```

参数说明（数据库字段映射）：

| 前端字段                         | 数据库字段                             | 说明     |
|------------------------------|-----------------------------------|--------|
| courseItems[].itemId         | plan_course_item.item_id          | 课程安排ID |
| courseItems[].courseId       | plan_course_item.course_id        | 课程ID   |
| courseItems[].courseName     | course.course_name                | 课程名称   |
| courseItems[].classDate      | plan_course_item.class_date       | 上课日期   |
| courseItems[].classBeginTime | plan_course_item.class_begin_time | 开始时间   |
| courseItems[].classEndTime   | plan_course_item.class_end_time   | 结束时间   |
| courseItems[].location       | plan_course_item.location         | 上课地点   |
| employees[].personId         | plan_employee.person_id           | 员工ID   |
| employees[].name             | person.name                       | 员工姓名   |

## 5.6 为培训计划添加员工

## 接口名称

为培训计划添加员工接口

## 逻辑描述

- 1. 前端提交计划ID和员工ID列表，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可操作）
- 3. 验证 plan\_id 是否存在
- 4. 验证员工ID列表的合法性：
  - 检查所有员工ID是否存在
  - 检查员工角色是否为"员工"
- 5. 检查员工是否已关联到该计划（避免重复添加）
- 6. 批量插入 plan\_employee 表
- 7. 返回添加成功信息
- 8. 异常情况：
  - 计划不存在：返回 "培训计划不存在"
  - 员工不存在或角色不对：返回具体错误信息

## 接口路径

```
POST /api/planner/plans/:planId/employees
```

## 请求方式

POST

## 输入参数

### 请求头：

```
Authorization: Bearer <token>
```

### 路径参数：

| 参数名    | 类型     | 说明     | 数据库字段                 |
|--------|--------|--------|-----------------------|
| planId | number | 培训计划ID | plan_employee.plan_id |

### 请求体：

```
{
 "employeeIds": [3001, 3002, 3003] // 必填，员工ID数组
}
```

参数说明：

| 参数名         | 类型    | 必填 | 说明     | 数据库字段                   |
|-------------|-------|----|--------|-------------------------|
| employeeIds | array | 是  | 员工ID数组 | plan_employee.person_id |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "添加成功",
 "data": {
 "addedCount": 3,
 "skippedCount": 0
 }
}
```

部分成功响应（200）：

```
{
 "code": 200,
 "message": "添加完成，部分员工已存在",
 "data": {
 "addedCount": 2,
 "skippedCount": 1
 }
}
```

# 5.7 从培训计划移除员工

接口名称

从培训计划移除员工接口

逻辑描述

- 1. 前端提交计划ID和员工ID，携带 Token



- 2. 后端验证用户身份和权限（仅课程大纲制定者可操作）
- 3. 验证 plan\_id 和 person\_id 是否存在
- 4. 检查该员工是否已有课程评价记录（attendance\_evaluation 表）
- 5. 如果已有评价记录，返回警告（建议不要删除，或提供强制删除选项）
- 6. 删除 plan\_employee 表对应记录
- 7. 如果强制删除，同时删除 attendance\_evaluation 表对应记录
- 8. 返回删除成功信息
- 9. 异常情况：
  - 员工不在该计划中：返回 "员工不在该培训计划中"

接口路径

```
DELETE /api/planner/plans/:planId/employees/:employeeId
```

请求方式

DELETE

输入参数

请求头：

```
Authorization: Bearer <token>
```

路径参数：

| 参数名        | 类型     | 说明     | 数据库字段                   |
|------------|--------|--------|-------------------------|
| planId     | number | 培训计划ID | plan_employee.plan_id   |
| employeeId | number | 员工ID   | plan_employee.person_id |

查询参数：

| 参数名   | 类型      | 必填 | 说明                       |
|-------|---------|----|--------------------------|
| force | boolean | 否  | 是否强制删除（同时删除评价记录），默认false |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "移除成功",
 "data": null
}
```

### 存在评价记录警告响应（400）：

```
{
 "code": 400,
 "message": "该员工已有课程评价记录，建议不要移除。如需强制删除，请添加force=true参数",
 "data": {
 "evaluationCount": 5
 }
}
```

---

## 5.8 获取课程列表

### 接口名称

#### 获取课程列表接口

### 逻辑描述

1. 前端请求课程列表，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
3. 查询 `course` 表，获取所有课程
4. 关联查询 `person` 表，获取主讲讲师姓名
5. 关联查询 `plan_course_item` 表，统计每个课程的安排次数
6. 支持分页、筛选（按课程类型）、排序
7. 返回课程列表
8. 异常情况：
  - 无权限：返回 "无权限访问"

### 接口路径

```
GET /api/planner/courses
```

### 请求方式

GET

输入参数

请求头:

Authorization: Bearer <token>

查询参数:

| 参数名         | 类型     | 必填 | 说明        |
|-------------|--------|----|-----------|
| page        | number | 否  | 页码，默认1    |
| pageSize    | number | 否  | 每页数量，默认10 |
| courseClass | string | 否  | 课程类型筛选    |
| keyword     | string | 否  | 课程名称关键词搜索 |
| teacherId   | number | 否  | 按讲师筛选     |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "total": 50,
 "page": 1,
 "pageSize": 10,
 "list": [
 {
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "courseDesc": "介绍船舶安全的基本知识和操作规范",
 "courseRequire": "无特殊要求",
 "courseClass": "安全培训",
 "teacherId": 4001,
 "teacherName": "李老师",
 "scheduledCount": 8
 }
]
 }
}
```

参数说明 (数据库字段映射):

| 前端字段           | 数据库字段                           | 说明   |
|----------------|---------------------------------|------|
| courseId       | course.course_id                | 课程ID |
| courseName     | course.course_name              | 课程名称 |
| courseDesc     | course.course_desc              | 课程描述 |
| courseRequire  | course.course_require           | 课程要求 |
| courseClass    | course.course_class             | 课程类型 |
| teacherId      | course.teacher_id               | 讲师ID |
| teacherName    | person.name                     | 讲师姓名 |
| scheduledCount | COUNT(plan_course_item.item_id) | 安排次数 |

## 5.9 创建课程

### 接口名称

创建课程接口

### 逻辑描述

1. 前端提交课程信息，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可创建）
3. 验证数据合法性：
  - 课程名称不为空，长度1-50字符
  - 讲师ID存在且角色为"讲师"
4. 生成 `course_id`（自增）
5. 插入 `course` 表
6. 返回新创建的课程信息
7. 异常情况：
  - 讲师不存在或角色不对：返回 "讲师不存在或角色错误"
  - 参数验证失败：返回具体错误信息

### 接口路径

POST /api/planner/courses

### 请求方式

POST

输入参数

请求头:

Authorization: Bearer <token>

请求体:

```
{
 "courseName": "string", // 必填, 课程名称
 "courseDesc": "string", // 可选, 课程描述
 "courseRequire": "string", // 可选, 课程要求
 "courseClass": "string", // 必填, 课程类型
 "teacherId": number // 必填, 讲师ID
}
```

参数说明:

| 参数名           | 类型     | 必填 | 说明              | 数据库字段                 |
|---------------|--------|----|-----------------|-----------------------|
| courseName    | string | 是  | 课程名称, 长度1-50字符  | course.course_name    |
| courseDesc    | string | 否  | 课程描述, 长度最多100字符 | course.course_desc    |
| courseRequire | string | 否  | 课程要求, 长度最多500字符 | course.course_require |
| courseClass   | string | 是  | 课程类型, 长度最多20字符  | course.course_class   |
| teacherId     | number | 是  | 讲师ID            | course.teacher_id     |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "创建成功",
 "data": {
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "courseDesc": "介绍船舶安全的基本知识和操作规范",
 "courseRequire": "无特殊要求",
 "courseClass": "安全培训",
 "teacherId": 4001,
 "teacherName": "李老师"
 }
}
```

```
}
}
```

### 讲师错误响应 (400):

```
{
 "code": 400,
 "message": "讲师不存在或角色错误",
 "data": null
}
```

---

## 5.10 修改课程

### 接口名称

### 修改课程接口

### 逻辑描述

1. 前端提交修改的课程信息，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可修改）
3. 验证 `course_id` 是否存在
4. 验证数据合法性（同创建接口）
5. 更新 `course` 表对应记录
6. 返回更新后的课程信息
7. 异常情况：
  - 课程不存在：返回 "课程不存在"

### 接口路径

```
PUT /api/planner/courses/:courseId
```

### 请求方式

PUT

### 输入参数

### 请求头:

Authorization: Bearer <token>

路径参数：

| 参数名      | 类型     | 说明   | 数据库字段            |
|----------|--------|------|------------------|
| courseId | number | 课程ID | course.course_id |

请求体：

```
{
 "courseName": "string", // 可选，课程名称
 "courseDesc": "string", // 可选，课程描述
 "courseRequire": "string", // 可选，课程要求
 "courseClass": "string", // 可选，课程类型
 "teacherId": number // 可选，讲师ID
}
```

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "修改成功",
 "data": {
 "courseId": 5001,
 "courseName": "船舶安全基础（修订版）",
 "courseDesc": "介绍船舶安全的基本知识和操作规范",
 "courseRequire": "需提前预习相关资料",
 "courseClass": "安全培训",
 "teacherId": 4001,
 "teacherName": "李老师"
 }
}
```

# 5.11 删除课程

接口名称

删除课程接口

逻辑描述

- 1. 前端提交要删除的课程ID，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可删除）
- 3. 验证 `course_id` 是否存在
- 4. 检查是否存在关联数据：
  - 检查 `plan_course_item` 表，是否有课程安排
- 5. 如果存在关联数据，返回错误提示
- 6. 如果无关联数据，删除 `course` 表对应记录
- 7. 返回删除成功信息
- 8. 异常情况：
  - 课程不存在：返回 "课程不存在"
  - 存在关联数据：返回 "无法删除，存在课程安排"

接口路径

```
DELETE /api/planner/courses/:courseId
```

请求方式

DELETE

输入参数

请求头:

```
Authorization: Bearer <token>
```

路径参数:

| 参数名      | 类型     | 说明   | 数据库字段            |
|----------|--------|------|------------------|
| courseId | number | 课程ID | course.course_id |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "删除成功",
 "data": null
}
```



存在关联数据响应（400）：

```
{
 "code": 400,
 "message": "无法删除，该课程已被安排到培训计划中",
 "data": {
 "scheduledCount": 5
 }
}
```

## 5.12 获取课程安排列表

接口名称

获取课程安排列表接口

逻辑描述

- 1. 前端请求课程安排列表，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
- 3. 查询 `plan_course_item` 表，获取所有课程安排
- 4. 关联查询 `training_plan` 表，获取计划名称
- 5. 关联查询 `course` 表，获取课程详细信息
- 6. 关联查询 `person` 表，获取讲师姓名
- 7. 支持分页、筛选（按计划、按课程、按日期范围）、排序
- 8. 返回课程安排列表
- 9. 异常情况：
  - 无权限：返回 "无权限访问"

接口路径

```
GET /api/planner/course-items
```

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

查询参数：

| 参数名       | 类型     | 必填 | 说明                           |
|-----------|--------|----|------------------------------|
| page      | number | 否  | 页码，默认1                       |
| pageSize  | number | 否  | 每页数量，默认20                    |
| planId    | number | 否  | 按培训计划筛选                      |
| courseId  | number | 否  | 按课程筛选                        |
| startDate | string | 否  | 开始日期筛选（YYYY-MM-DD）           |
| endDate   | string | 否  | 结束日期筛选（YYYY-MM-DD）           |
| sortBy    | string | 否  | 排序字段：class_date，默认class_date |
| sortOrder | string | 否  | 排序方向：asc/desc，默认asc          |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "total": 120,
 "page": 1,
 "pageSize": 20,
 "list": [
 {
 "itemId": 10001,
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "courseClass": "安全培训",
 "teacherId": 4001,
 "teacherName": "李老师",
 "classDate": "2024-01-15",
 "classBeginTime": "09:00:00",
 "classEndTime": "11:00:00",
 "location": "培训室A"
 }
]
 }
}
```

参数说明（数据库字段映射）：

| 前端字段           | 数据库字段                             | 说明     |
|----------------|-----------------------------------|--------|
| itemId         | plan_course_item.item_id          | 课程安排ID |
| planId         | plan_course_item.plan_id          | 培训计划ID |
| planName       | training_plan.plan_name           | 培训计划名称 |
| courseId       | plan_course_item.course_id        | 课程ID   |
| courseName     | course.course_name                | 课程名称   |
| courseClass    | course.course_class               | 课程类型   |
| teacherId      | course.teacher_id                 | 讲师ID   |
| teacherName    | person.name                       | 讲师姓名   |
| classDate      | plan_course_item.class_date       | 上课日期   |
| classBeginTime | plan_course_item.class_begin_time | 开始时间   |
| classEndTime   | plan_course_item.class_end_time   | 结束时间   |
| location       | plan_course_item.location         | 上课地点   |

## 5.13 创建课程安排

### 接口名称

创建课程安排接口

### 逻辑描述

1. 前端提交课程安排信息，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可创建）
3. 验证数据合法性：
  - plan\_id 和 course\_id 是否存在
  - 上课日期是否在培训计划的时间范围内
  - 开始时间 < 结束时间
  - 地点不为空
4. 检查时间冲突：
  - 同一讲师同一时间不能有多个课程
  - 同一地点同一时间不能有多个课程
5. 生成 item\_id（自增）
6. 插入 plan\_course\_item 表

- 7. 为该计划的所有员工自动创建 attendance\_evaluation 记录（初始为空）
- 8. 返回新创建的课程安排信息
- 9. 异常情况：
  - 时间冲突：返回 "时间冲突"
  - 参数验证失败：返回具体错误信息

接口路径

POST /api/planner/course-items

请求方式

POST

输入参数

请求头:

Authorization: Bearer <token>

请求体:

```
{
 "planId": number, // 必填，培训计划ID
 "courseId": number, // 必填，课程ID
 "classDate": "string", // 必填，上课日期（YYYY-MM-DD）
 "classBeginTime": "string", // 必填，开始时间（HH:mm:ss）
 "classEndTime": "string", // 必填，结束时间（HH:mm:ss）
 "location": "string" // 必填，上课地点
}
```

参数说明:

| 参数名       | 类型     | 必填 | 说明                    | 数据库字段                       |
|-----------|--------|----|-----------------------|-----------------------------|
| planId    | number | 是  | 培训计划ID                | plan_course_item.plan_id    |
| courseId  | number | 是  | 课程ID                  | plan_course_item.course_id  |
| classDate | string | 是  | 上课日期，格式<br>YYYY-MM-DD | plan_course_item.class_date |

| 参数名            | 类型     | 必填 | 说明              | 数据库字段                             |
|----------------|--------|----|-----------------|-----------------------------------|
| classBeginTime | string | 是  | 开始时间，格式HH:mm:ss | plan_course_item.class_begin_time |
| classEndTime   | string | 是  | 结束时间，格式HH:mm:ss | plan_course_item.class_end_time   |
| location       | string | 是  | 上课地点，长度最多100字符  | plan_course_item.location         |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "创建成功",
 "data": {
 "itemId": 10001,
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "classDate": "2024-01-15",
 "classBeginTime": "09:00:00",
 "classEndTime": "11:00:00",
 "location": "培训室A"
 }
}
```

时间冲突响应（400）：

```
{
 "code": 400,
 "message": "时间冲突：讲师李老师在該时间段已有其他课程安排",
 "data": null
}
```

## 5.14 修改课程安排

接口名称

修改课程安排接口

逻辑描述

- 1. 前端提交修改的课程安排信息，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可修改）
- 3. 验证 item\_id 是否存在
- 4. 验证数据合法性（同创建接口）
- 5. 检查时间冲突（排除当前记录）
- 6. 更新 plan\_course\_item 表对应记录
- 7. 返回更新后的课程安排信息
- 8. 异常情况：
  - 课程安排不存在：返回 "课程安排不存在"
  - 时间冲突：返回 "时间冲突"

接口路径

PUT /api/planner/course-items/:itemId

请求方式

PUT

输入参数

请求头:

Authorization: Bearer <token>

路径参数:

| 参数名    | 类型     | 说明     | 数据库字段                    |
|--------|--------|--------|--------------------------|
| itemId | number | 课程安排ID | plan_course_item.item_id |

请求体:

```
{
 "classDate": "string", // 可选，上课日期
 "classBeginTime": "string", // 可选，开始时间
 "classEndTime": "string", // 可选，结束时间
 "location": "string" // 可选，上课地点
}
```

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "修改成功",
 "data": {
 "itemId": 10001,
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "classDate": "2024-01-16",
 "classBeginTime": "09:00:00",
 "classEndTime": "11:00:00",
 "location": "培训室B"
 }
}
```

## 5.15 删除课程安排

接口名称

删除课程安排接口

逻辑描述

1. 前端提交要删除的课程安排ID，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可删除）
3. 验证 `item_id` 是否存在
4. 检查是否存在关联数据：
  - 检查 `attendance_evaluation` 表，是否有员工评价记录
5. 如果存在评价记录，返回警告（建议不要删除，或提供强制删除选项）
6. 开启事务：
  - 删除 `attendance_evaluation` 表对应记录（如果强制删除）
  - 删除 `plan_course_item` 表对应记录
7. 返回删除成功信息
8. 异常情况：
  - 课程安排不存在：返回 "课程安排不存在"

接口路径

DELETE /api/planner/course-items/:itemId

请求方式

DELETE

输入参数

请求头：

Authorization: Bearer <token>

路径参数：

| 参数名    | 类型     | 说明     | 数据库字段                    |
|--------|--------|--------|--------------------------|
| itemId | number | 课程安排ID | plan_course_item.item_id |

查询参数：

| 参数名   | 类型      | 必填 | 说明                       |
|-------|---------|----|--------------------------|
| force | boolean | 否  | 是否强制删除（同时删除评价记录），默认false |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "删除成功",
 "data": null
}
```

存在评价记录警告响应（400）：

```
{
 "code": 400,
 "message": "该课程安排已有员工评价记录，建议不要删除。如需强制删除，请添加force=true参数",
 "data": {
 "evaluationCount": 30
 }
}
```



# 5.16 获取平台数据分析

## 接口名称

获取平台数据分析接口

## 逻辑描述

1. 前端请求平台整体数据分析，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
3. 查询并统计多维度数据：
  - **课程评分排名**：查询 `v_course_score` 视图，按平均分排序
  - **培训计划评分排名**：查询 `v_plan_score` 视图，按平均分排序
  - **课程类型分布**：统计各课程类型的课程数和平均分
  - **培训计划进度统计**：统计各状态（规划中/进行中/已完成）的计划数
  - **员工成绩排名**：计算各员工的整体平均分，排序
  - **讲师授课统计**：统计各讲师的授课数和平均评分
4. 返回综合数据分析结果
5. 异常情况：
  - 无权限：返回 "无权限访问"

## 接口路径

```
GET /api/planner/analytics
```

## 请求方式

GET

## 输入参数

## 请求头：

```
Authorization: Bearer <token>
```

## 查询参数：

| 参数名  | 类型     | 必填 | 说明             |
|------|--------|----|----------------|
| topN | number | 否  | 返回排名前N的数据，默认10 |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseRankings": [
 {
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "courseAvgScore": 92.5,
 "studentCount": 50
 }
],
 "planRankings": [
 {
 "planId": 1001,
 "planName": "2024年新员工培训计划",
 "planAvgScore": 88.3
 }
],
 "courseClassDistribution": [
 {
 "courseClass": "安全培训",
 "courseCount": 12,
 "avgScore": 90.2
 }
],
 "planStatusStatistics": [
 {
 "planStatus": "规划中",
 "count": 3
 },
 {
 "planStatus": "进行中",
 "count": 5
 },
 {
 "planStatus": "已完成",
 "count": 8
 }
],
 "employeeRankings": [
 {
 "personId": 3001,
 "personName": "王员工",
 "avgScore": 95.2,
 "courseCount": 15
 }
],
 "teacherStatistics": [
 {
```

```
 "teacherId": 4001,
 "teacherName": "李老师",
 "courseCount": 10,
 "avgScore": 91.5,
 "studentCount": 120
 }
]
}
```

参数说明（数据库字段映射）：

| 前端字段                            | 数据库来源                           | 说明     |
|---------------------------------|---------------------------------|--------|
| courseRankings[].courseId       | v_course_score.course_id        | 课程ID   |
| courseRankings[].courseName     | v_course_score.course_name      | 课程名称   |
| courseRankings[].courseAvgScore | v_course_score.course_avg_score | 课程平均分  |
| courseRankings[].studentCount   | v_course_score.student_count    | 参与学员数  |
| planRankings[].planId           | v_plan_score.plan_id            | 培训计划ID |
| planRankings[].planName         | v_plan_score.plan_name          | 培训计划   |

| 前端字段                                  | 数据库来源                                     | 说明    |
|---------------------------------------|-------------------------------------------|-------|
|                                       |                                           | 名称    |
| planRankings[].planAvgScore           | v_plan_score.plan_avg_score               | 计划平均分 |
| courseClassDistribution[].courseClass | course.course_class                       | 课程类型  |
| courseClassDistribution[].courseCount | COUNT(course.course_id)                   | 课程数   |
| courseClassDistribution[].avgScore    | AVG(v_course_score.course_avg_score)      | 平均分   |
| planStatusStatistics[].planStatus     | training_plan.plan_status                 | 计划状态  |
| planStatusStatistics[].count          | COUNT(training_plan.plan_id)              | 计划数   |
| employeeRankings[].personId           | person.person_id                          | 员工ID  |
| employeeRankings[].personName         | person.name                               | 员工姓名  |
| employeeRankings[].avgScore           | AVG(v_employee_item_score.weighted_score) | 平均分   |

| 前端字段                             | 数据库来源                                | 说明   |
|----------------------------------|--------------------------------------|------|
| employeeRankings[].courseCount   | COUNT(v_employee_item_score.item_id) | 课程数  |
| teacherStatistics[].teacherId    | person.person_id                     | 讲师ID |
| teacherStatistics[].teacherName  | person.name                          | 讲师姓名 |
| teacherStatistics[].courseCount  | COUNT(course.course_id)              | 授课数  |
| teacherStatistics[].avgScore     | AVG(v_course_score.course_avg_score) | 平均评分 |
| teacherStatistics[].studentCount | SUM(v_course_score.student_count)    | 学员总数 |

## 5.17 获取员工成绩详情

### 接口名称

获取员工成绩详情接口

### 逻辑描述

1. 前端请求指定员工的成绩详情，携带 Token
2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
3. 查询 `v_employee_item_score` 视图，获取该员工所有课程的详细成绩
4. 查询 `v_employee_course_score` 视图，获取该员工各课程类型的平均分
5. 计算该员工的整体平均分
6. 返回员工成绩完整信息

7. 异常情况：
- 员工不存在：返回 "员工不存在"

接口路径

GET /api/planner/employees/:employeeId/scores

请求方式

GET

输入参数

请求头：

Authorization: Bearer <token>

路径参数：

| 参数名        | 类型     | 说明   | 数据库字段            |
|------------|--------|------|------------------|
| employeeId | number | 员工ID | person.person_id |

返回值

成功响应（200）：

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "personId": 3001,
 "personName": "王员工",
 "overallAvgScore": 92.5,
 "courseCount": 15,
 "courseClassScores": [
 {
 "courseClass": "安全培训",
 "avgWeightedScore": 93.2
 }
],
 "itemScores": [
 {
 "itemId": 10001,
 "courseName": "船舶安全基础",
 "courseClass": "安全培训",
 "classDate": "2024-01-15",
```

```
 "classBeginTime": "09:00:00",
 "classEndTime": "11:00:00",
 "selfScore": 90.0,
 "teacherScore": 95.0,
 "weightedScore": 92.5
 }
}
]
```

参数说明（数据库字段映射）：

| 前端字段                                 | 数据库来源                                      | 说明    |
|--------------------------------------|--------------------------------------------|-------|
| personId                             | person.person_id                           | 员工ID  |
| personName                           | person.name                                | 员工姓名  |
| overallAvgScore                      | AVG(v_employee_item_score.weighted_score)  | 整体平均分 |
| courseCount                          | COUNT(v_employee_item_score.item_id)       | 课程数   |
| courseClassScores[].courseClass      | v_employee_course_score.course_class       | 课程类型  |
| courseClassScores[].avgWeightedScore | v_employee_course_score.avg_weighted_score | 类型平均分 |
| itemScores[]                         | v_employee_item_score                      | 每节课   |

## 5.18 获取课程评价详情

### 接口名称

获取课程评价详情接口

### 逻辑描述

- 1. 前端请求指定课程的评价详情，携带 Token
- 2. 后端验证用户身份和权限（仅课程大纲制定者可访问）
- 3. 查询 `v_employee_item_score` 视图，获取该课程所有员工的评价和成绩
- 4. 查询 `v_course_score` 视图，获取课程整体统计
- 5. 计算课程的平均分、最高分、最低分
- 6. 返回课程评价完整信息
- 7. 异常情况：
  - 课程不存在：返回 "课程不存在"

### 接口路径

GET /api/planner/courses/:courseId/evaluations

### 请求方式

GET

### 输入参数

### 请求头:

Authorization: Bearer <token>

### 路径参数:



| 参数名      | 类型     | 说明   | 数据库字段            |
|----------|--------|------|------------------|
| courseId | number | 课程ID | course.course_id |

返回值

成功响应 (200):

```
{
 "code": 200,
 "message": "获取成功",
 "data": {
 "courseId": 5001,
 "courseName": "船舶安全基础",
 "courseAvgScore": 92.5,
 "studentCount": 50,
 "maxScore": 98.0,
 "minScore": 75.0,
 "evaluations": [
 {
 "personId": 3001,
 "personName": "王员工",
 "itemId": 10001,
 "classDate": "2024-01-15",
 "selfScore": 90.0,
 "selfComment": "课程内容非常实用，讲师讲解清晰",
 "teacherScore": 95.0,
 "teacherComment": "表现优秀，积极参与互动",
 "weightedScore": 92.5
 }
]
 }
}
```

参数说明 (数据库字段映射):

| 前端字段           | 数据库来源                           | 说明    |
|----------------|---------------------------------|-------|
| courseId       | course.course_id                | 课程ID  |
| courseName     | course.course_name              | 课程名称  |
| courseAvgScore | v_course_score.course_avg_score | 课程平均分 |
| studentCount   | v_course_score.student_count    | 学员人数  |

| 前端字段                         | 数据库来源                                     | 说明     |
|------------------------------|-------------------------------------------|--------|
| maxScore                     | MAX(v_employee_item_score.weighted_score) | 最高分    |
| minScore                     | MIN(v_employee_item_score.weighted_score) | 最低分    |
| evaluations[].personId       | v_employee_item_score.person_id           | 员工ID   |
| evaluations[].personName     | v_employee_item_score.person_name         | 员工姓名   |
| evaluations[].itemId         | v_employee_item_score.item_id             | 课程安排ID |
| evaluations[].classDate      | v_employee_item_score.class_date          | 上课日期   |
| evaluations[].selfScore      | attendance_evaluation.self_score          | 自评分    |
| evaluations[].selfComment    | attendance_evaluation.self_comment        | 自评内容   |
| evaluations[].teacherScore   | attendance_evaluation.teacher_score       | 讲师评分   |
| evaluations[].teacherComment | attendance_evaluation.teacher_comment     | 讲师评语   |
| evaluations[].weightedScore  | v_employee_item_score.weighted_score      | 加权得分   |