API

一、公共数据库-PublicDB（4张表）

表单列表

（一）公告表（Notice）暂不考虑

（1）说明

（2）MySQL指令

（3）API

GET /api/notice/get

（二）资源表（PublicResources）

（1）说明

（2）MySQL指令

（3）API

GET /api/resources/get/{id}

POST /api/images/add/{id}

DELETE /api/images/delete/{id}

测试部分：

GET /api/images/list

（三）用户信息表（Users）

（1）说明

API说明：

- 所有时间戳使用ISO 8601格式
- 文件类型使用标准MIME类型格式

一、公共数据库-PublicDB（4张表）

表单列表

~~（一）公告表（Notice）~~ 暂不考虑

~~（1）说明~~

~~存储主页功能的公告信息~~

- `id`：~~整数类型，自增，主键。~~

- `date`：日期类型，存储公告的日期。
- `content`：文本类型，存储公告的内容。
- `is_display`：布尔类型，表示公告是否呈现，默认为展示（TRUE）。
- `created_at`：记录公告创建的时间。
- `updated_at`：记录公告最后更新的时间。
- `expires_at`：公告的有效期，超过这个时间则不再展示。
- `author`：发布公告的用户或部门。

id	date日期	content公告内容	is_display是否呈现	created_at	updated_at	expires_at	author
xxxxx	2025/2/1	挑战杯开始了！	True	2025-01-31T12:00:00Z	2025-01-31T12:00:00Z	2025-02-28T23:59:59Z	项目部
	2025/2/2	小程序上线啦！	False				


```
{
  "announcements": [
    {
      "id": 1,
      "date": "2025-02-01",
      "content": "挑战杯开始了！",
      "is_display": true,
      "created_at": "2025-01-31T12:00:00Z",
      "updated_at": "2025-01-31T12:00:00Z",
      "expires_at": "2025-02-28T23:59:59Z",
      "author": "项目部"
    },
    {
      "id": 2,
      "date": "2025-02-02",
      "content": "小程序上线啦！",
      "is_display": false,
      "created_at": "2025-02-01T15:30:00Z",
      "updated_at": "2025-02-01T15:30:00Z",
      "expires_at": "2025-03-01T23:59:59Z",
      "author": "基管部"
    }
  ]
}
```

```
}

```

(2) MySQL指令

```sql
CREATE TABLE `Notice` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `date` DATE NOT NULL,
  `content` TEXT NOT NULL,
  `is_display` BOOLEAN NOT NULL DEFAULT TRUE,
  `created_at` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  `expires_at` DATETIME NOT NULL,
  `author` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- 插入示例数据
INSERT INTO `Notice` (`date`, `content`, `is_display`, `created_at`, `updated_at`, `expires_at`, `author`) VALUES
('2025-02-01', '挑战杯开始了！', TRUE, '2025-01-31T12:00:00Z', '2025-01-31T12:00:00Z', '2025-02-28T23:59:59Z', '项目部'),
('2025-02-02', '小程序上线啦！', FALSE, '2025-02-01T12:00:00Z', '2025-02-01T12:00:00Z', '2025-03-31T23:59:59Z', '项目部');
```

(3) API

GET /api/notice/get

- 请求

```
{
    "id": 1
}
```

- 响应

```
{
}
```

~~POST /api/notice/add~~

~~DELETE /api/notice/delete~~

（二）资源表（PublicResources）

存储所有页面中可能会用到的资源

（1）说明

- `id`：整数类型，自增，主键。
- `resource_name`：字符串类型，存储资源名称，如’主页图’、'主页icon’等。
- `data`：二进制对象类型，存储编码后的图片二进制数据。
- `filetype`：字符串类型，存储图片的MIME类型，如’image/jpeg’、'image/png’等。
- `created_at`：记录文件上传的时间。
- `updated_at`：记录文件最后更新的时间。
- `alt_text`：图片的替代文本，用于屏幕阅读器或图片加载失败时显示。

id	resource_name	data	filetype 文件类型	created_at	updated_at	alt_text
1	主页图	base64编码的二进制数据	image/jpeg	2025-01-30T10:00:00Z	2025-01-30T10:00:00Z	主页展示图片
2	主页icon	base64编码的二进制数据	image/png			
3	项目立项申请表	base64编码的二进制数据	application/vnd.openxmlformats-officedocument.wordprocessingml.document			

```
{
  "images": [
    {
```

```
        "id": 1,
        "resource_name": "主页图",
        "data": "base64编码的二进制数据",
        "filetype": "image/jpeg",
        "created_at": "2025-01-30T10:00:00Z",
        "updated_at": "2025-01-30T10:00:00Z",
        "alt_text": "主页展示图片"
      },
      {
        "id": 2,
        "resource_name": "主页icon",
        "data": "base64编码的二进制数据",
        "filetype": "image/png",
        "created_at": "2025-01-31T11:00:00Z",
        "updated_at": "2025-01-31T11:00:00Z",
        "alt_text": "主页图标"
      }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE public_resources (
    id INT PRIMARY KEY AUTO_INCREMENT,
    resource_name VARCHAR(100) NOT NULL COMMENT '资源名称',
    data LONGBLOB NOT NULL COMMENT '资源二进制数据',
    filetype VARCHAR(100) NOT NULL COMMENT '文件MIME类型',
    created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
    updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP COMMENT '更新时间',
    alt_text VARCHAR(255) DEFAULT NULL COMMENT '替代文本描述'
) COMMENT '公共资源表';
```

（3）API

GET /api/resources/get/{id}

说明：根据资源ID获取资源详细信息

- 请求

```
路径参数：{id}，表示文件的唯一标识符。
```

- 响应

```
{
    "code": 200,
    "message": "success",
    "data": {
        "id": 1,
        "resource_name": "主页图",
        "data": "base64编码的二进制数据",
        "filetype": "image/jpeg",
        "created_at": "2025-01-30T10:00:00Z",
        "updated_at": "2025-01-30T10:00:00Z",
        "alt_text": "主页展示图片"
    }
}
```

POST /api/images/add/{id}

说明：上传新的资源文件

- 请求

```
{
    "resource_name": "主页icon",
    "data": "base64编码的二进制数据",
    "filetype": "image/png",
    "alt_text": "主页图标"
}
```

- 响应

```
{
    "code": 200,
    "message": "success",
    "data": {
        "id": 2,
        "resource_name": "主页icon",
        "filetype": "image/png",
        "created_at": "2025-01-31T11:00:00Z",
        "updated_at": "2025-01-31T11:00:00Z",
        "alt_text": "主页图标"
    }
}
```

DELETE /api/images/delete/{id}

说明：根据资源ID删除指定资源

- 请求

```
路径参数：{id}，表示文件的唯一标识符。
```

- 响应

```
{
    "code": 200,
    "message": "success",
    "data": {
        "id": 1
    }
}
```

测试部分：

GET /api/images/list

说明：获取所有资源的列表，不返回二进制数据

- 请求

```
无需请求体
```

- 响应

```
{
    "code": 200,
    "message": "success",
    "data": [
        {
            "id": 1,
            "resource_name": "主页图",
            "filetype": "image/jpeg",
            "created_at": "2025-01-30T10:00:00Z",
            "updated_at": "2025-01-30T10:00:00Z",
```

```
            "alt_text": "主页展示图片"
        },
        {
            "id": 2,
            "resource_name": "主页icon",
            "filetype": "image/png",
            "created_at": "2025-01-31T11:00:00Z",
            "updated_at": "2025-01-31T11:00:00Z",
            "alt_text": "主页图标"
        }
    ]
}
```

（三）用户信息表（Users）

（1）说明

需要使用的页面：

- **登录页面和注册页面**

  注册时用户提供邮箱和密码，传入后端后，后端发送给对应邮箱验证码，用户需要输入验证码，后端验证成功后，确认为注册成功，录入数据库，后面几个键值由算法生成：

  a. 默认level为1（1是编外人员，2是干事，3是部长及以上）level键值需要后端进行修改。

  b. realname初始化为猫猫xxxx，xxxx由后端随机生成数字。

  c. phone_num初始化为空串（string）。

  d. note初始化为暂无。

  e. status：status表示用户的状态，有被封禁，和正常状态

  f. profile_photo初始化为协会猫猫头的图片。

  ！：此刻通过level键值来判断是否进行渲染某些功能。将level键值放在前端暂存区，用于后续页面渲染的识别。

id	userid（邮箱）	password	level	realname	phone_num	note	state	profile_photo	score
1	xxx@xxx	123456	1	猫猫1234	暂无	暂无	0，1（已封禁、正常）	二进制数据	10


```
{
  "users": [
    {
      "id": 1,
      "userid": "test@example.com",
      "password": "hashed_password_123",
      "level": 1,
      "realname": "猫猫1234",
      "phone_num": "12345678901",
      "note": "暂无",
      "state": "1",
      "profile_photo": "base64编码的图片数据",
      "score": 0
    }
  ]
}
```

（2）MySQL指令

```
CREATE TABLE users (
    id INT PRIMARY KEY AUTO_INCREMENT,
    userid VARCHAR(255) NOT NULL UNIQUE COMMENT '用户邮箱',
    password VARCHAR(255) NOT NULL COMMENT '密码',
    level INT NOT NULL DEFAULT 1 COMMENT '用户等级：1编外人员、2干事、3部长及以上',
    realname VARCHAR(100) NOT NULL COMMENT '用户名',
    phone_num VARCHAR(20) DEFAULT '' COMMENT '手机号码',
    note TEXT DEFAULT '暂无' COMMENT '备注',
    state TINYINT NOT NULL DEFAULT 1 COMMENT '用户状态：0已封禁、1正常',
    profile_photo LONGBLOB DEFAULT NULL COMMENT '用户头像',
    score INT NOT NULL DEFAULT 0 COMMENT '用户积分',
) COMMENT '用户信息表';
```

（3）API

POST /api/users/login/{userid}

说明：**登录页面**用户登录接口，如果检测到state是0，拒绝登陆。

- 请求

```
{
    "password": "123456"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "登录成功",
    "data": {
        "id": 1,
        "level": 1,
        "username": "猫猫1234",
        "state":1
    }
}
```

```
{
    "code": 401,
    "status": "fail",
    "message": "已封禁",
    "state":"0"
}
```

```
{
    "code": 406,
    "status": "fail",
    "message": "密码错误",
    "state":"1"
}
```

POST /api/users/register

说明：**注册页面**用户注册接口

- 请求

```
{
    "userid":"xxx@xxx",
```

```
    "password":"123456",
    "verify":"123456" #验证码仅用于后端验证注册，在填入数据库时应该删除该键值
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "注册成功",
    "data": {
        "id": 1,
        "userid": "test@example.com",
        "username": "猫猫1234"
    }
}
```

POST /api/users/profile/edit/{userid}

说明：**“我的”编辑页面**更新用户基本信息

其中，头像的修改单独使用uploadFile完成

- 请求

```
{
    "username": "猫猫1234",
    "phone_num": "12345678901",
    "note": "不是哥们"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "修改成功"
}
```

GET /api/users/profile/{userid}

说明：“我的”主页面获取用户个人信息

前端GET指令只get包含username在内的后面的总共5个键值（status除外）

- 请求

```
路径参数：{userid}，表示唯一标识符。
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取成功",
    "data": {
        "level": 1,
        "username": "猫猫1234",
        "phone_num": "12345678901",
        "note": "不是哥们",
        "profile_photo": "base64编码的图片数据",
        "score": 20
    }
}
```

（四）规章制度表（Rules）

（1）说明

这里的主键仅有1，根据前端不同的GET请求返回对应的字典。

Rules表用于存储各类规章制度文件的内容。每条记录包含唯一标识id、文件名称file_name和具体内容content。这些规章制度用于前端显示和用户查看，是系统重要的规范性文档。

id	file_id	file_name	content
1	xxx	借物条件及责任须知	xxxxx
2	xxx	101场地使用条例（项目立项）	
3	xxx		

		101场地使用条例（借用场地）	
4	xxx	场地归还确认责任须知	
5	xxx	结束项目条件及责任须知	

```
{
    "rules": [
        {
            "id": 1,
            "file_id":"xxx",
            "file_name": "借物条件及责任须知",
            "content": "神秘的字符串"
        },
        {
            "id": 2,
            "file_id":"xxx",
            "file_name": "101场地使用条例（项目立项）",
            "content": "神秘的字符串"
        },
        {
            "id": 3,
            "file_id":"xxx",
            "file_name": "101场地使用条例（借用场地）",
            "content": "神秘的字符串"
        },
        {
            "id": 4,
            "file_id":"xxx",
            "file_name": "场地归还确认责任须知",
            "content": "神秘的字符串"
        },
        {
            "id": 5,
            "file_id":"xxx",
            "file_name": "结束项目条件及责任须知",
            "content": "神秘的字符串"
        }
    ]
}
```

（2）MySQL指令

```sql
CREATE TABLE rules (
    id INT PRIMARY KEY,
    file_id VARCHAR(255) NOT NULL COMMENT '文件ID',
    file_name VARCHAR(255) NOT NULL COMMENT '规章制度文件名称',
    content TEXT NOT NULL COMMENT '规章制度具体内容'
) COMMENT '规章制度表';
```

（3）API

GET /api/rules/get/{id}

说明：获取指定id的规章制度内容

- 请求

```
路径参数：{id}，表示唯一标识符。
```

- 响应

```json
{
    "code": 200,
    "status": "success",
    "message": "获取成功",
    "data": {
        "id": 1,
        "file_name": "借物条件及责任须知",
        "content": "具体内容..."
    }
}
```

POST /api/rules/update/{id}

说明：更新指定规章制度的内容

- 请求

```json
{
    "content": "新的规章制度内容..."
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "更新成功"
}
```

GET /api/rules/list

说明：获取所有规章制度列表

- 请求

```
无
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取列表成功",
    "data": {
        "rules": [
            {
                "id": 1,
                "file_name": "借物条件及责任须知",
                "content": "具体内容..."
            },
            // ... 其他规章制度
        ]
    }
}
```

二、协会数据库（15张）

表单列表

（一）3d打印（3dPrint)

（1）说明

- `id`：整数类型，自增。
- `apply_id`：特定位数的数字，唯一对应一个申请单，主键
- `userid`：字符串类型，用户邮箱
- `phone_num`：字符串类型，联系电话
- `score`：整型，积分
- `score_change`：整型，积分变化
- `name`：字符串，姓名
- `quantity`：浮点型，用料量 /g
- `printer`：整型，打印机选择i创街、208，分别为 `0` 和 `1`。
- `file_zip`：字符串类型，储存二进制文件
- `created_at`：字符串类型，时间存储采用ISO 8601格式。
- `updated_at`：字符串类型，时间存储采用ISO 8601格式。
- `state`：整型，表示审核状态，0表示未审核，1表示审核通过，2表示未通过。
- `reason`：字符串类型，运维审核时，填写该字段，作为审核通过或者驳回的理由。

id	apply_id	userid（邮箱）	phone_num	score	score_change	name	quantity	printer	file_zip	created_at	updated_at	state (0表示未审核，1表示审核通过，2表示未通过)	reason
1	xxxxxxx	xxx@xxx		10	-12	xxx	料量	1	base64编码的.zi				

									p文件数据				

```
{
    "3d_print": [
        {
            "id": 1,
            "apply_id": "3DP20240213001",
            "userid": "student@example.com",
            "phone_num": "13800138000",
            "score": 100,
            "score_change": -10,
            "name": "张三",
            "quantity": 50.5,
            "printer": 1,
            "file_zip": "base64编码的zip文件内容...",
            "created_at": "2024-02-13 14:30:00",
            "updated_at": "2024-02-13 14:30:00",
            "state": 0,
            "reason": null
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE 3d_print (
    id INT AUTO_INCREMENT,
    apply_id VARCHAR(50) PRIMARY KEY COMMENT '申请单号',
    userid VARCHAR(255) NOT NULL COMMENT '用户邮箱',
    phone_num VARCHAR(20) NOT NULL COMMENT '联系电话',
    score INT NOT NULL COMMENT '积分',
    score_change INT NOT NULL COMMENT '积分变化',
    name VARCHAR(50) NOT NULL COMMENT '姓名',
    quantity FLOAT NOT NULL COMMENT '用料量(g)',
    printer TINYINT NOT NULL COMMENT '打印机选择：0-i创街、1-208',
    file_zip LONGBLOB NOT NULL COMMENT 'ZIP文件数据',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间',
    updated_at VARCHAR(50) NOT NULL COMMENT '更新时间',
    state TINYINT NOT NULL DEFAULT 0 COMMENT '审核状态：0未审核、1审核通过、2未通过',
    reason TEXT DEFAULT NULL COMMENT '审核理由'
```

```
) COMMENT '3D打印申请表';
```

（3）API

POST /api/3d_print/apply

说明：提交3D打印申请

- 请求

```
{
    "userid": "student@example.com",
    "phone_num": "13800138000",
    "name": "张三",
    "quantity": 50.5,
    "printer": 1,
    "file_zip": "base64编码的zip文件内容..."
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "申请提交成功",
    "data": {
        "apply_id": "3DP20240213001"
    }
}
```

GET /api/3d_print/history/{userid}

说明：获取用户历史申请记录

- 请求

```
路径参数userid
```

- 响应

```
{
```

```
    "code": 200,
    "status": "success",
    "message": "获取历史申请成功",
    "data": {
        "records": [
            {
                "apply_id": "3DP20240213001",
                "quantity": 50.5,
                "printer": 1,
                "created_at": "2024-02-13 14:30:00",
                "state": 0,
                "reason": null
            }
        ]
    }
}
```

GET /api/3d_print/audit/view/{state}

说明：获取指定状态的审核列表

- 请求

```
路径参数status（0/1/2）
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取"{state}"状态审核列表成功",
    "data": {
        "applications": [
            {
                "apply_id": "3DP20240213001",
                "name": "张三",
                "quantity": 50.5,
                "created_at": "2024-02-13 14:30:00"
            }
        ]
    }
}
```

GET /api/3d_print/audit/details/{apply_id}

说明：获取特定申请的详细信息

- 请求

```
路径参数apply_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取"{apply_id}"申请成功",
    "data": {
        "apply_id": "3DP20240213001",
        "userid": "student@example.com",
        "phone_num": "13800138000",
        "name": "张三",
        "quantity": 50.5,
        "printer": 1,
        "file_zip": "base64编码的zip文件内容...",
        "created_at": "2024-02-13 14:30:00",
        "state": 0
    }
}
```

POST /api/3d_print/audit/respond/{apply_id}

说明：审核反馈（返回reason和更改status状态）

- 请求

```
{
    "state": 1,
    "reason": "审核通过"
}
```

- 响应

```
{
    "code": 200,
```

```
    "status": "success",
    "message": "审核完成"
}
```

测试路由：

GET /api/3d_print/list

说明：获取所有3D打印申请记录

- 请求

```
无
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取打印申请记录成功",
    "data": {
        "applications": [
            {
                "apply_id": "3DP20240213001",
                "userid": "student@example.com",
                "name": "张三",
                "quantity": 50.5,
                "state": 0,
                "created_at": "2024-02-13 14:30:00"
            }
        ]
    }
}
```

（二）活动详情表（Events）

（1）说明

- `id`：整数类型，事件的唯一标识符。通常为自增，主键。
- `event_id`：特定位数的数字，唯一对应一个详情条目，主键。

- `event_name`：字符串类型，活动的名称，例如“校园文化节”。
- `poster`：字符串类型，存储活动海报的图片数据，通常使用 base64 编码后的图片二进制数据。
- `description`：字符串类型，描述活动的详细信息，简要说明活动的内容。
- `location`：字符串类型，活动的举办地点，如“学校大礼堂”。
- `link`：字符串类型，活动的相关链接地址，可以是报名链接或详细信息页面的 URL。
- `start_time`：字符串类型，活动开始的时间，采用 ISO 8601 格式，例如“2023-05-10T18:00:00Z”。
- `end_time`：字符串类型，活动结束的时间，采用 ISO 8601 格式，例如“2023-05-12T22:00:00Z”。
- `registration_deadline`：字符串类型，活动报名截止时间，采用 ISO 8601 格式。
- `created_at`：字符串类型，记录创建的时间，通常采用 ISO 8601 格式。
- `updated_at`：字符串类型，记录最后更新的时间，通常采用 ISO 8601 格式。

id	event_id	event_name	poster	description	location	link	start_time	end_time	registration_deadline	created_at	updated_at
1											

```
{
    "events": [
        {
            "id": 1,
            "event_id": "EV20240213001",
            "event_name": "校园文化节",
            "poster": "base64编码的海报图片数据...",
            "description": "一年一度的校园文化节，包含各种文艺表演和趣味活动。",
            "location": "学校大礼堂",
            "link": "http://example.com/cultural-festival",
            "start_time": "2024-02-15T18:00:00Z",
            "end_time": "2024-02-17T22:00:00Z",
            "registration_deadline": "2024-02-14T23:59:59Z",
            "created_at": "2024-02-13T12:00:00Z",
            "updated_at": "2024-02-13T12:00:00Z"
        }
    ]
}
```

（2）MySQL指令

```sql
CREATE TABLE events (
    id INT AUTO_INCREMENT,
    event_id VARCHAR(50) PRIMARY KEY COMMENT '活动ID',
    event_name VARCHAR(255) NOT NULL COMMENT '活动名称',
    poster LONGBLOB NOT NULL COMMENT '活动海报',
    description TEXT NOT NULL COMMENT '活动描述',
    location VARCHAR(255) NOT NULL COMMENT '活动地点',
    link VARCHAR(255) NOT NULL COMMENT '活动链接',
    start_time VARCHAR(50) NOT NULL COMMENT '开始时间',
    end_time VARCHAR(50) NOT NULL COMMENT '结束时间',
    registration_deadline VARCHAR(50) NOT NULL COMMENT '报名截止时间',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间',
    updated_at VARCHAR(50) NOT NULL COMMENT '更新时间'
) COMMENT '活动详情表';
```

（3）API

POST /api/events/post

说明：发布一条新活动。

- 请求

```json
{
    "event_name": "校园文化节",
    "poster": "base64编码的海报图片数据...",
    "description": "一年一度的校园文化节，包含各种文艺表演和趣味活动。",
    "location": "学校大礼堂",
    "link": "http://example.com/cultural-festival",
    "start_time": "2024-02-15T18:00:00Z",
    "end_time": "2024-02-17T22:00:00Z",
    "registration_deadline": "2024-02-14T23:59:59Z"
}
```

- 响应

```json
{
    "code": 200,
    "status": "success",
    "message": "活动发布成功",
    "data": {
```

```
        "event_id": "EV20240213001"
    }
}
```

GET /api/events/view

说明：获取近期活动列表，拉取所有活动。

- 请求

```
{
    "page": 1,
    "size": 10 //查询呈现出来的行数
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取近期活动列表成功",
    "data": {
        "total": 1,
        "events": [
            {
                "event_id": "EV20240213001",
                "event_name": "校园文化节",
                "poster": "base64编码的海报图片数据...",
                "start_time": "2024-02-15T18:00:00Z",
                "registration_deadline": "2024-02-14T23:59:59Z"
            }
        ]
    }
}
```

GET /api/events/details/{event_id}

说明：这是活动详情查看，查看某条活动。

- 请求

```
路径参数event_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取活动详情成功",
    "data": {
        "event_id": "EV20240213001",
        "event_name": "校园文化节",
        "poster": "base64编码的海报图片数据...",
        "description": "一年一度的校园文化节，包含各种文艺表演和趣味活动。",
        "location": "学校大礼堂",
        "link": "http://example.com/cultural-festival",
        "start_time": "2024-02-15T18:00:00Z",
        "end_time": "2024-02-17T22:00:00Z",
        "registration_deadline": "2024-02-14T23:59:59Z"
    }
}
```

DELETE /api/events/{event_id}

说明：删除某条活动。

- 请求

```
路径参数event_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "活动删除成功"
}
```

（三）任务表（Tasks）

（1）说明

- `id`：整数类型，任务的唯一标识符。通常为自增，主键。

- `task_id`：整数类型，关联的任务标识符。此值可以是外键，关联到其他任务表。
- `department`：字符串类型，负责该任务的部门名称，例如“宣传部”。
- `task_name`：字符串类型，任务的名称，例如“写公众号推文”。
- `name`：字符串类型，负责该任务的人员名称，例如“Garmen”。
- `content`：字符串类型，任务的详细描述，简要说明任务的具体内容，例如“写元旦晚会的公众号推文，图片在xxx手里”。
- `state`:0-未完成，1-已完成。
- `deadline`：字符串类型，任务的截止日期和时间，采用 ISO 8601 格式，例如“2025-01-31T12:00:00Z”。

id	task_id	department	task_name	name	content	state	deadline

```
{
    "tasks": [
        {
            "id": 1,
            "task_id": "TK20240213001",
            "department": "宣传部",
            "task_name": "写公众号推文",
            "name": "Garmen",
            "content": "写元旦晚会的公众号推文，图片在xxx手里",
            "deadline": "2025-01-31T12:00:00Z",
            "state": 0,  // 建议添加：0-未完成，1-已完成
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE tasks (
    id INT AUTO_INCREMENT,
    task_id VARCHAR(50) PRIMARY KEY COMMENT '任务ID',
    department VARCHAR(50) NOT NULL COMMENT '部门名称',
    task_name VARCHAR(255) NOT NULL COMMENT '任务名称',
    name VARCHAR(50) NOT NULL COMMENT '负责人姓名',
    content TEXT NOT NULL COMMENT '任务内容',
```

```
    state TINYINT NOT NULL DEFAULT 0 COMMENT '任务状态：0未完成、1已完成',
    deadline VARCHAR(50) NOT NULL COMMENT '截止时间'
) COMMENT '任务表';
```

（3）API

GET /api/tasks/{task_id}

说明：获取特定任务详情，留下给后续可能的“我的任务”的界面使用。

- 请求

```
路径参数task_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取任务详情成功",
    "data": {
        "task_id": "TK20240213001",
        "department": "宣传部",
        "task_name": "写公众号推文",
        "name": "Garmen",
        "content": "写元旦晚会的公众号推文，图片在xxx手里",
        "deadline": "2025-01-31T12:00:00Z",
        "state": 0,
    }
}
```

DELETE /api/tasks/{task_id}

说明：删除指定任务，目前不需要这个接口，为日后“任务面板”留下余地。

- 请求

```
路径参数task_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "任务删除成功"
}
```

POST /api/tasks

说明：添加一条任务。

- 请求

```
{
    "department": "宣传部",
    "task_name": "写公众号推文",
    "name": "Garmen",
    "content": "写元旦晚会的公众号推文，图片在xxx手里",
    "deadline": "2025-01-31T12:00:00Z"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "任务创建成功",
    "data": {
        "task_id": "TK20240213001"
    }
}
```

GET /api/tasks/department/{department}

说明：获取某个部门的所有任务

- 请求

```
无
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取部门任务成功",
    "data": {
        "department": "宣传部",
        "total_tasks": 50,
        "completed_tasks": 30,
        "current_page": 1,
        "page_size": 10,
        "tasks": [
            {
                "task_id": "TK20240213001",
                "task_name": "写公众号推文",
                "name": "Garmen",
                "deadline": "2025-01-31T12:00:00Z",
                "state": 0,
            }
        ],
        "statistics": {
            "total": 50,
            "completed": 30,
            "pending": 20,
            "overdue": 5
        }
    }
}
```


PUT /api/tasks/{task_id}/state

说明：更新任务状态

- 请求

```
{
    "state": 1  // 0-未完成，1-已完成
}
```

- 响应


```
{
    "code": 200,
    "status": "success",
    "message": "任务状态更新成功",
```

```
    "data": {
        "task_id": "TK20240213001",
        "state": 1,
    }
}
```

GET /api/tasks/list

说明：获取任务列表，支持分页和筛选

- 请求

```
无
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取任务列表",
    "data": {
        "total": 100,
        "current_page": 1,
        "page_size": 10,
        "tasks": [
            {
                "task_id": "TK20240213001",
                "department": "宣传部",
                "task_name": "写公众号推文",
                "name": "Garmen",
                "deadline": "2025-01-31T12:00:00Z",
                "state": 0,
            }
        ]
    }
}
```

PUT /api/tasks/{task_id}

说明：更新任务信息

- 请求

```
{
    "department": "宣传部",      // 可选
    "task_name": "修改后的任务名称", // 可选
    "name": "Garmen",          // 可选
    "content": "更新后的任务内容", // 可选
    "deadline": "2025-01-31T12:00:00Z" // 可选
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "任务更新成功",
    "data": {
        "task_id": "TK20240213001",
        "department": "宣传部",
        "task_name": "修改后的任务名称",
        "name": "Garmen",
        "content": "更新后的任务内容",
        "deadline": "2025-01-31T12:00:00Z",
    }
}
```

（四）借物表（StuffBorrow）

（1）说明

1. `id`：整数类型(INT)，借用记录的唯一标识符。自增主键。
2. `userid`：字符串类型(VARCHAR(50))，用户的邮箱账号，唯一认证ID，例如"user@example.com"。
3. `sb_id`：字符串类型(VARCHAR(20))，借物编号，例如"JW202400001"。
4. `name`：字符串类型(VARCHAR(50))，借用人姓名，例如"张三"。
5. `phone_num`：字符串类型(VARCHAR(11))，借用人手机号码，必须为11位有效号码，例如"19925555555"。
6. `email`：字符串类型(VARCHAR(50))，借用人邮箱地址，例如"example@domain.com"。
7. `grade`：字符串类型(VARCHAR(10))，借用人年级，例如"2024"。
8. `major`：字符串类型(VARCHAR(50))，借用人专业，例如"计算机科学与技术"。

9. `project_num`： 学校给的项目编号。
10. `mentor_name`: 字符串类型(VARCHAR(50)), 指导老师姓名，例如"李四"，可为空。
11. `mentor_phone_num`: 字符串类型(VARCHAR(11)), 指导老师电话，例如"13900139000"，可为空。
12. `type`: 字符串类型(VARCHAR(50)), 借用物品的类型，例如"开发板"。
13. `stuff_name`: 字符串类型(VARCHAR(100)), 具体借用物品的名称，例如"ESP-32"。
14. `stuff_quantity_change`: 整数类型(INT), 物品数量的变化，负数表示借出，例如"-1"表示借出1个。
15. `deadline`: 日期时间类型(DATETIME), 归还截止时间，采用ISO 8601格式，例如"2024-02-13T15:30:00Z"。
16. `reason`: 文本类型(TEXT), 借用原因的详细说明，例如"制作智能门锁"。
17. `categories`: 整数类型(TINYINT), 0表示个人借用，1表示团队借用。
18. `state`: 整数类型(TINYINT), 0表示未审核，1表示审核通过，2表示审核失败。

id	userid	sb_id	name	phone_num	email	grade	major	project_num	mentor_name	mentor_phone_num	type	stuff_name	stuff_quantity_change	de li
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

```
{
    "stuff_borrow": [
        {
            "id": 1,
            "userid":"xx@xx",
            "sb_id":"xxxxxx",
            "name":"邹顺戈",
            "phone_num":"19925555555", //必须检查位数合法
            "email":"xx@xx",
            "grade":"2024",
            "major":"计算机科学与技术",
            "project_num":"xxxxxx"
            "mentor_name":"lovesg",
            "mentor_phone_num":"",
            "type":"开发板",
            "stuff_name":"ESP-32",
            "stuff_quantity_change":"-1",
```

```
            "deadline":"2024-02-13T15:30:00Z",
            "reason": "制作智能门锁",
            "categories":1, //0为个人借物标志，1为团队借物标志
            "state": 0     //0为未审核，1为审核通过，2为审核失败
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE stuff_borrow (
    id INT AUTO_INCREMENT,
    userid VARCHAR(50) NOT NULL COMMENT '用户邮箱',
    sb_id VARCHAR(20) PRIMARY KEY COMMENT '借物编号',
    name VARCHAR(50) NOT NULL COMMENT '借用人姓名',
    phone_num VARCHAR(11) NOT NULL COMMENT '借用人手机号',
    email VARCHAR(50) NOT NULL COMMENT '借用人邮箱',
    grade VARCHAR(10) NOT NULL COMMENT '借用人年级',
    major VARCHAR(50) NOT NULL COMMENT '借用人专业',
    project_num VARCHAR(50) DEFAULT NULL COMMENT '项目编号',
    mentor_name VARCHAR(50) DEFAULT NULL COMMENT '指导老师姓名',
    mentor_phone_num VARCHAR(11) DEFAULT NULL COMMENT '指导老师手机号',
    type VARCHAR(50) NOT NULL COMMENT '借用物品类型',
    stuff_name VARCHAR(100) NOT NULL COMMENT '借用物品名称',
    stuff_quantity_change INT NOT NULL COMMENT '物品数量变化',
    deadline VARCHAR(50) NOT NULL COMMENT '归还截止时间',
    reason TEXT NOT NULL COMMENT '借用原因',
    categories TINYINT NOT NULL DEFAULT 0 COMMENT '借用类型：0个人、1团队',
    state TINYINT NOT NULL DEFAULT 0 COMMENT '审核状态：0未审核、1审核通过、2审核失败'
) COMMENT '借物表';
```

（3）API

GET /api/stuff-borrow/statistics

说明：统计借物总数。

- 请求

```
无需请求体
```

- 响应

```
{
    "code": 200,
    "status":"success"
    "message": "获取成功",
    "data": {
        "total": 100,          // 借物总数
        "pending": 20,         // 待审核数量
        "approved": 70,        // 已通过数量
        "rejected": 10,        // 已拒绝数量
        "overdue": 5           // 逾期数量
    }
}
```

DELETE /api/stuff-borrow/{sb_id}

说明：删除某条特定的借物申请。

- 请求

```
无需请求体
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "删除成功"
}
```

GET /api/stuff-borrow/user/{userid}/history?page=1&size=10

说明：获取用户的借物历史。

- 请求

```
// 请求参数:
// page: 页码，默认1
// size: 每页数量，默认10
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取用户借物历史",
    "data": {
        "total": 50,
        "pages": 5,
        "current": 1,
        "records": [
            {
                "id": 1,
                "sb_id": "JW202400001",
                "stuff_name": "ESP-32",
                "type": "开发板",
                "stuff_quantity_change": -1,
                "deadline": "2024-02-13T15:30:00Z",
                "state": 1
            }
        ]
    }
}
```


PUT /api/stuff-borrow/{sb_id}

说明：更新某一条借物申请。

- 请求


```
{
    "name": "张三",
    "phone_num": "19925555555",
    "email": "example@domain.com",
    "grade": "2024",
    "major": "计算机科学与技术",
    "mentor_name": "李四",
    "mentor_phone_num": "13900139000",
    "type": "开发板",
    "stuff_name": "ESP-32",
    "stuff_quantity_change": -1,
    "deadline": "2024-02-13T15:30:00Z",
    "reason": "制作智能门锁",
    "categories": 0
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "更新成功",
    "data": {
        "sb_id": "JW202400001"
    }
}
```

GET /api/stuff-borrow/{sb_id}

说明：获取某条借物申请条目。

- 请求

```
无需请求体
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取申请条目",
    "data": {
        "id": 1,
        "userid": "user@example.com",
        "sb_id": "JW202400001",
        "name": "张三",
        "phone_num": "19925555555",
        "email": "example@domain.com",
        "grade": "2024",
        "major": "计算机科学与技术",
        "mentor_name": "李四",
        "mentor_phone_num": "13900139000",
        "type": "开发板",
        "stuff_name": "ESP-32",
        "stuff_quantity_change": -1,
        "deadline": "2024-02-13T15:30:00Z",
        "reason": "制作智能门锁",
        "categories": 0,
        "state": 1
```

```
    }
}
```

GET /api/stuff-borrow?page=1&size=10&state=0

说明：获取借物总表（全部获取）。

- 请求

```
// 请求参数:
// page: 页码，默认1
// size: 每页数量，默认10
// state: 状态筛选，可选值: 0,1,2
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取总表成功",
    "data": {
        "total": 100,
        "pages": 10,
        "current": 1,
        "records": [
            {
                "id": 1,
                "userid": "user@example.com",
                "sb_id": "JW202400001",
                "name": "张三",
                "phone_num": "19925555555",
                "email": "example@domain.com",
                "grade": "2024",
                "major": "计算机科学与技术",
                "mentor_name": "李四",
                "mentor_phone_num": "13900139000",
                "type": "开发板",
                "stuff_name": "ESP-32",
                "stuff_quantity_change": -1,
                "deadline": "2024-02-13T15:30:00Z",
                "reason": "制作智能门锁",
                "categories": 0,
                "state": 0
            }
```

```
        ]
    }
}
```

GET /api/stuff-borrow/user/{userid}?page=1&size=10&state=0

说明：获取某个用户的借物总表。

- 请求

```
// 请求参数:
// page: 页码，默认1
// size: 每页数量，默认10
// state: 状态筛选，可选值: 0,1,2
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "获取总表成功",
    "data": {
        "total": 100,
        "pages": 10,
        "current": 1,
        "records": [
            {
                "id": 1,
                "userid": "user@example.com",
                "sb_id": "JW202400001",
                "name": "张三",
                "phone_num": "19925555555",
                "email": "example@domain.com",
                "grade": "2024",
                "major": "计算机科学与技术",
                "mentor_name": "李四",
                "mentor_phone_num": "13900139000",
                "type": "开发板",
                "stuff_name": "ESP-32",
                "stuff_quantity_change": -1,
                "deadline": "2024-02-13T15:30:00Z",
                "reason": "制作智能门锁",
                "categories": 0,
                "state": 0
```

```
            }
        ]
    }
}
```

POST /api/stuff-borrow

说明：增加一条借物申请条目。

- 请求

```
{
    "userid": "user@example.com",
    "name": "张三",
    "phone_num": "19925555555",
    "email": "example@domain.com",
    "grade": "2024",
    "major": "计算机科学与技术",
    "mentor_name": "李四",
    "mentor_phone_num": "13900139000",
    "type": "开发板",
    "stuff_name": "ESP-32",
    "stuff_quantity_change": -1,
    "deadline": "2024-02-13T15:30:00Z",
    "reason": "制作智能门锁",
    "categories": 0
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "申请成功",
    "data": {
        "sb_id": "JW202400001"
    }
}
```

（五）物资表(Stuff)

（1）说明

- `id`：整数类型，物资的唯一标识符。通常为自增，主键。
- `stuff_id`：字符串类型，物资的唯一编号，例如“ST10001”。
- `type`：字符串类型，物资的类型，如“开发板”。
- `stuff_name`：字符串类型，物资的名称，如“ESP32”。
- `number`：整数类型，物资的数量，例如20。
- `description`：字符串类型，物资的描述信息，用于详细说明物资的特性或用途，例如“ESP32-WROOM-32开发板”。
- `created_at`：字符串类型（ISO 8601格式），物资的创建时间，例如“2024-02-13T10:00:00Z”。
- `updated_at`：字符串类型（ISO 8601格式），物资的最后更新时间，例如“2024-02-13T10:00:00Z”。

id	stuff_id	type	stuff_name	number	description	created_at	updated_at

```
{
    "stuff": [
        {
            "id": 1,
            "stuff_id": "ST10001",
            "type": "开发板",
            "stuff_name": "ESP32",
            "number": 20,
            "description": "ESP32-WROOM-32开发板",  // 建议添加描述
            "created_at": "2024-02-13T10:00:00Z",
            "updated_at": "2024-02-13T10:00:00Z"
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE stuff (
    id INT AUTO_INCREMENT,
```

```
    stuff_id VARCHAR(50) PRIMARY KEY COMMENT '物资编号',
    type VARCHAR(50) NOT NULL COMMENT '物资类型',
    stuff_name VARCHAR(100) NOT NULL COMMENT '物资名称',
    number INT NOT NULL DEFAULT 0 COMMENT '物资数量',
    description TEXT NOT NULL COMMENT '物资描述',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间',
    updated_at VARCHAR(50) NOT NULL COMMENT '更新时间'
) COMMENT '物资表';
```

（3）API

POST /api/stuff/update

说明：更新某个物资的数量

- 请求

```
{
    "stuff_id": "ST10001",
    "number": 25
}
```

- 响应

```
{
    "code":200,
    "status": "success",
    "message": "物资数量更新成功",
    "data": {
        "id": 1,
        "stuff_id": "ST10001",
        "type": "开发板",
        "stuff_name": "ESP32",
        "number": 25,
        "description": "ESP32-WROOM-32开发板",
        "created_at": "2024-02-13T10:00:00Z",
        "updated_at": "2024-02-13T10:00:00Z"
    }
}
```

POST /api/stuff/add

说明：管理员增加物资条目。

- 请求

```
{
    "type": "开发板",
    "stuff_name": "ESP32",
    "number": 20,
    "description": "ESP32-WROOM-32开发板"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "物资添加成功",
    "data": {
        "stuff_id": "ST10001"
    }
}
```

POST /api/stuff/{stuff_id}

说明：管理员更改某条物资条目信息

- 请求

```
{
    "type": "开发板",  // 可选
    "stuff_name": "ESP32-New",  // 可选
    "description": "更新后的描述"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "物资信息更新成功",
    "data": {
        "stuff_id": "ST10001",
        "updated_at": "2024-02-13T15:30:00Z"
```

```
    }
}
```

DELETE /api/stuff/{stuff_id}

说明：管理员删除物资条目。

- 请求

```
路径参数stuff_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "物资删除成功"
}
```

（六）场地借用表(SitesBorrow)

（1）说明

1. `id`：整数类型(INT)，场地借用记录的唯一标识符。自增主键。
2. `apply_id`：字符串类型(VARCHAR(20))，场地借用申请编号，例如"SB20240213001"。
3. `name`：字符串类型(VARCHAR(50))，申请人姓名，例如"张三"。
4. `student_id`：字符串类型(VARCHAR(13))，学号，必须为13位有效数字，例如"2023141460079"。
5. `phonenum`：字符串类型(VARCHAR(11))，申请人手机号码，必须为11位有效号码，例如"13800138000"。
6. `email`：字符串类型(VARCHAR(50))，申请人邮箱地址，例如"student@example.com"。
7. `purpose`：字符串类型(VARCHAR(100))，场地使用目的，例如"创新项目展示与研讨"。
8. `mentor_name`：字符串类型(VARCHAR(50))，指导教师姓名，例如"李教授"。
9. `mentor_phone_num`：字符串类型(VARCHAR(11))，指导教师手机号码，必须为11位有效号码，例如"13900139000"。

10. `picture`: 文本类型(TEXT), 场地使用平面图，以base64格式存储。
11. `start_time`: 日期时间类型(DATETIME), 使用开始时间，采用ISO 8601格式，例如"2024-02-15T09:00:00Z"。
12. `end_time`: 日期时间类型(DATETIME), 使用结束时间，采用ISO 8601格式，例如"2024-02-15T12:00:00Z"。
13. `state`: 整数类型(TINYINT), 申请状态：0表示待审核，1表示已通过，2表示已拒绝，3表示已取消。
14. `reason`: 文本类型(TEXT), 审核意见，可为空。
15. `created_at`: 日期时间类型(DATETIME), 记录创建时间，采用ISO 8601格式，例如"2024-02-13T10:00:00Z"。
16. `updated_at`: 日期时间类型(DATETIME), 记录最后更新时间，采用ISO 8601格式，例如"2024-02-13T10:00:00Z"。

id	apply_id	name	student_id	phonenum	email	purpose	mentor_name	mentor_phone_num	picture（平面图）	start_time	end_time	state	reason	u at _

```
{
    "sites_borrow": [
        {
            "id": 1,
            "apply_id": "SB20240213001",
            "name": "张三",
            "student_id": "2023141460079",
            "phonenum": "13800138000",
            "email": "student@example.com",
            "purpose": "创新项目展示与研讨",
            "mentor_name": "李教授",
            "mentor_phone_num": "13900139000",
            "picture": "base64编码的平面图...",
            "start_time": "2024-02-15T09:00:00Z",
            "end_time": "2024-02-15T12:00:00Z",
            "state": 0,  // 0-待审核，1-已通过，2-已拒绝，3-已取消
            "reason": null, // 审核意见
            "created_at": "2024-02-13T10:00:00Z",
            "updated_at": "2024-02-13T10:00:00Z"
```

```
        }
    ]
}
```

（2）MySQL指令

```sql
CREATE TABLE sites_borrow (
    id INT AUTO_INCREMENT,
    apply_id VARCHAR(20) PRIMARY KEY COMMENT '申请编号',
    name VARCHAR(50) NOT NULL COMMENT '申请人姓名',
    student_id VARCHAR(13) NOT NULL COMMENT '学号',
    phonenum VARCHAR(11) NOT NULL COMMENT '申请人手机号',
    email VARCHAR(50) NOT NULL COMMENT '申请人邮箱',
    purpose VARCHAR(100) NOT NULL COMMENT '使用目的',
    mentor_name VARCHAR(50) NOT NULL COMMENT '指导教师姓名',
    mentor_phone_num VARCHAR(11) NOT NULL COMMENT '指导教师手机号',
    picture LONGTEXT NOT NULL COMMENT '场地平面图',
    start_time VARCHAR(50) NOT NULL COMMENT '开始时间',
    end_time VARCHAR(50) NOT NULL COMMENT '结束时间',
    state TINYINT NOT NULL DEFAULT 0 COMMENT '申请状态：0待审核、1已通过、2已拒绝、3已取消',
    reason TEXT DEFAULT NULL COMMENT '审核意见',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间',
    updated_at VARCHAR(50) NOT NULL COMMENT '更新时间'
) COMMENT '场地借用表';
```

（3）API

POST /api/sites_borrow/add

说明：添加一条待审核的场地借用申请条目。

- 请求

```json
{
    "apply_id": "SB20240213002",
    "name": "李四",
    "student_id": "2023141460080",
    "phonenum": "13900139000",
    "email": "student2@example.com",
    "purpose": "学术研讨会",
    "mentor_name": "王教授",
    "mentor_phone_num": "13700137000",
    "picture": "base64编码的平面图...",
```

```
    "start_time": "2024-02-20T09:00:00Z",
    "end_time": "202-402-20T12:00:00Z"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "申请提交成功",
    "data": {
        "apply_id": "SB20240213001"
    }
}
```

POST /api/sites_borrow/update

说明：更改场地借用申请的状态键。

- 请求

```
{
    "apply_id": "SB20240213001",
    "state": 1,
    "reason": "申请通过，请按时使用"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "状态更新成功",
    "data": {
        "apply_id": "SB20240213001",
        "state": 1,
        "updated_at": "2024-02-13T15:30:00Z"
    }
}
```

GET /api/sites_borrow/{apply_id}

说明：基管部审核页面拉取信息。

- 请求

```
路径参数apply_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "success",
    "data": {
        "apply_id": "SB20240213001",
        "name": "张三",
        "student_id": "2023141460079",
        "phonenum": "13800138000",
        "email": "student@example.com",
        "purpose": "创新项目展示与研讨",
        "mentor_name": "李教授",
        "mentor_phone_num": "13900139000",
        "picture": "base64编码的平面图...",
        "start_time": "2024-02-15T09:00:00Z",
        "end_time": "2024-02-15T12:00:00Z",
        "state": 0,
        "created_at": "2024-02-13T10:00:00Z"
    }
}
```

（七）项目表(Project)

（1）说明

- **id**
  - 类型：整型 (Integer)
  - 说明：记录的唯一标识符，通常是项目或申请的主键 ID。
- **apply_id**
  - 类型：字符串 (String)
  - 说明：申请的编号，用于唯一标识该申请，主键。

- **project_name**
  - 类型：字符串 (String)
  - 说明：项目的名称，用于描述项目的主题或目标。
- **director**
  - 类型：字符串 (String)
  - 说明：项目负责人或申请负责人姓名。
- **college**
  - 类型：字符串 (String)
  - 说明：项目负责人所在的学院或机构。
- **major_grade**
  - 类型：字符串 (String)
  - 说明：项目负责人所在的专业及年级信息。
- **phone_num**
  - 类型：字符串 (String)
  - 说明：负责人的联系电话，不允许为空。
- **email**
  - 类型：字符串 (String)
  - 说明：负责人的电子邮件地址，确保与负责人相关的联系信息，不允许为空。
- **mentor**
  - 类型：字符串 (String)
  - 说明：项目的指导教师姓名。
- **description**
  - 类型：字符串 (String)
  - 说明：项目的简介或描述，提供项目的背景和目标信息。
- **application_file**
  - 类型：字符串 (String)
  - 说明：存储与申请相关的文件路径或文件名，通常是Base64编码的申请文件内容。**101双创项目立项申请表**
- **prove_file**
  - 类型：字符串 (String)

    - 说明：存储与申请相关的证明文件路径或文件名，通常是Base64编码的证明文件内容。**101双创项目立项证明**
- **member**
    - 类型：数组 (Array of Strings)
    - 说明：成员列表，包含该项目或申请的参与成员名称。
- **start_time**
    - 类型：字符串 (String, ISO 8601格式)
    - 说明：项目的开始时间。
- **end_time**
    - 类型：字符串 (String, ISO 8601格式)
    - 说明：项目的结束时间。
- **audit_state**
    - 类型：整型 (Integer)
    - 说明：申请的当前状态，通常表示为不同的状态码：
        - 0：待审核
        - 1：已通过
        - 2：已拒绝
- **project_state**
    - 类型：整型 (Integer)
    - 说明：申请的当前状态，通常表示为不同的状态码：
        - 0：进行中
        - 1：已结束
- **created_at**
    - 类型：字符串 (String, ISO 8601格式)
    - 说明：记录的创建时间，通常用于跟踪申请的提交时间。

id（整型）	apply_id（字符串）	project_name（字符串）	director（字符串）	college（字符串）	major_grade（字符串）	phone_num（字符串）	email（字符串）	mentor（字符串）	description（字符串）	application_file（字符串）	prove_file（字符串）	member（对象）	start_time（字符串）	end_time（字符串）

1	10023		lovesg											

```
{
    "project": [
        {
            "id": 1,
            "apply_id": "PJ20240213001",
            "project_name": "创新实践项目",  // 建议添加项目名称
            "director": "张三",
            "college":"计算机学院",
            "major_grade":"2024级计算机科学与技术"
            "phone_num": "13800138000",
            "email": "project@example.com",
            "mentor":"张卫华",
            "description": "项目简介...",    // 建议添加项目描述
            "application_file": "base64编码的申请文件数据...",
            "prove_file":"base64编码的申请文件数据..."
            "member": ["李四", "王五", "赵六"],
            "start_time":"2024-02-13T10:00:00Z"
            "end_time":"2024-02-13T10:00:00Z"
            "audit_state":0,  // 0-待审核，1-已通过，2-已拒绝
            "project_state":1, //0-项目结束，1-项目进行中
            "created_at": "2024-02-13T10:00:00Z"
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE project (
    id INT AUTO_INCREMENT,
    apply_id VARCHAR(50) PRIMARY KEY COMMENT '申请编号',
    project_name VARCHAR(255) NOT NULL COMMENT '项目名称',
    director VARCHAR(50) NOT NULL COMMENT '项目负责人',
    college VARCHAR(100) NOT NULL COMMENT '学院',
    major_grade VARCHAR(100) NOT NULL COMMENT '专业年级',
    phone_num VARCHAR(11) NOT NULL COMMENT '联系电话',
    email VARCHAR(50) NOT NULL COMMENT '电子邮件',
    mentor VARCHAR(50) NOT NULL COMMENT '指导教师',
    description TEXT NOT NULL COMMENT '项目描述',
    application_file LONGTEXT NOT NULL COMMENT '立项申请表',
    prove_file LONGTEXT NOT NULL COMMENT '立项证明',
```

```
    member JSON NOT NULL COMMENT '项目成员',
    start_time VARCHAR(50) NOT NULL COMMENT '开始时间',
    end_time VARCHAR(50) NOT NULL COMMENT '结束时间',
    audit_state TINYINT NOT NULL DEFAULT 0 COMMENT '审核状态：0待审核、1已通过、2已拒绝',
    project_state TINYINT NOT NULL DEFAULT 1 COMMENT '项目状态：0已结束、1进行中',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间'
) COMMENT '项目表';
```

（3）API

POST /api/project/add

说明：新增一条待审核的项目条目。

- 请求

```
{
    "apply_id": "PJ20240213001",
    "project_name": "创新实践项目",
    "director": "张三",
    "college": "计算机学院",
    "major_grade": "2024级计算机科学与技术",
    "phone_num": "13800138000",
    "email": "project@example.com",
    "mentor": "张卫华",
    "description": "项目简介...",
    "application_file": "base64编码的申请文件数据...",
    "prove_file": "base64编码的申请文件数据...",
    "member": [],
    "start_time": "2024-02-13T10:00:00Z",
    "end_time": "2024-02-13T10:00:00Z"
}

```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "项目申请提交成功",
    "data": {
        "apply_id": "PJ20240213001"
```

```
    }
}
```

GET /api/project/list

说明：项目部审核页面拉取所有的项目条目。

- 请求

```
无
```

- 响应

```
{
    "code":"200",
    "status": "success",
    "data": [
        {
            "apply_id": "PJ20240213001",
            "project_name": "创新实践项目",
            "director": "张三",
            "college": "计算机学院",
            "major_grade": "2024级计算机科学与技术",
            "audit_state": 0,
            "project_state": 1,
            "start_time": "2024-02-13T10:00:00Z",
            "end_time": "2024-02-13T10:00:00Z",
            "created_at": "2024-02-13T10:00:00Z"
        }
    ]
}

```

GET /api/project/detail/{apply_id}

说明：获取项目详情

- 请求

```
路径参数apply_id
```

- 响应


```
{
    "status": "success",
    "data": {
        "apply_id": "PJ20240213001",
        "project_name": "创新实践项目",
        "director": "张三",
        "college": "计算机学院",
        "major_grade": "2024级计算机科学与技术",
        "phone_num": "13800138000",
        "email": "project@example.com",
        "mentor": "张卫华",
        "description": "项目简介...",
        "application_file": "base64编码的申请文件数据...",
        "prove_file": "base64编码的申请文件数据...",
        "member": ["李四", "王五", "赵六"],
        "start_time": "2024-02-13T10:00:00Z",
        "end_time": "2024-02-13T10:00:00Z",
        "audit_state": 0,
        "project_state": 1,
        "created_at": "2024-02-13T10:00:00Z"
    }
}
```


GET /api/project/view/{state}

说明：获取指定状态的项目列表

- 请求

```
路径参数state
```

- 响应


```
{
    "code": 200,
    "status": "success",
    "message": "success",
    "data": {
        "projects": [
            {
                "apply_id": "PJ20240213001",
```

```
                "project_name": "创新实践项目",
                "director": "张三",
                "created_at": "2024-02-13T10:00:00Z"
            }
        ]
    }
}
```

DELETE /api/project/delete/{apply_id}

说明：删除项目申请

- 请求

```
路径参数apply_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "项目删除成功"
    "data": {
    "apply_id": "PJ20240213001"
    }
}
```

POST /api/project/audit/{apply_id}

说明：审核项目申请，更新项目状态并提供审核意见

- 请求

```
{
    "audit_state": 1,  // 1-通过，2-拒绝
    "reason": "项目内容完整，目标明确，通过审核。",  // 审核意见
}
```

- 响应

```
{
```

```
    "code": 200,
    "status": "success",
    "message": "审核完成",
    "data": {
        "apply_id": "PJ20240213001",
        "project_name": "创新实践项目",
        "audit_state": 1,
        "project_state": 1,
        "reason": "项目内容完整，目标明确，通过审核。",
        "audit_time": "2024-02-13T15:30:00Z"
    }
}
```

POST /api/project/terminate/{apply_id}

说明：为后端传参，结束项目。

- 请求

```
{
    "project_state": 0,  // 0-结束，1-开启
    "reason": "项目已结束",  // 审核意见
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功结束项目",
    "data": {
        "apply_id": "PJ20240213001",
        "project_name": "创新实践项目",
        "project_state": 0,
    }
}
```

（八）赛事表(Games)

（1）说明

- `id`：整数类型，赛事的唯一标识符。通常为自增，主键。
- `game_id`:赛事id。
- `name`：字符串类型，赛事名称，例如“lovesg”。
- `wx_num`：字符串类型，赛事负责人或相关人员的微信号。
- `qq_num`：字符串类型，赛事负责人或相关人员的 QQ 号。
- `introduction`：字符串类型，赛事的简要介绍，默认为“暂无”。
- `registration_start`：字符串类型，报名开始时间，采用 ISO 8601 格式，如“2025-01-31T12:00:00Z”。
- `registration_end`：字符串类型，报名结束时间，采用 ISO 8601 格式。
- `contest_start`：字符串类型，赛事举办开始时间，采用 ISO 8601 格式。
- `contest_end`：字符串类型，赛事举办结束时间，采用 ISO 8601 格式。
- `link`：字符串类型，赛事的官方网站链接或相关链接，例如“https://www.official.com”。
- `created_at`:字符串类型，该条目创建时间戳，采用 ISO 8601 格式。

id	game_id	name	wx_num	qq_num	introduction	registration_start	registration_end	contest_start	contest_end	link	created_at

```
{
    "games": [
        {
            "id": 1,
            "game_id": "GM20240213001",  // 赛事编号
            "name": "2024创新创业大赛",
            "wx_num": "lovelovesg",
            "qq_num": "52000000",
            "introduction": "面向全校学生的创新创业竞赛...",
            "registration_start": "2024-02-15T00:00:00Z",
            "registration_end": "2024-03-15T23:59:59Z",
            "contest_start": "2024-04-01T00:00:00Z",
            "contest_end": "2024-04-15T23:59:59Z",
            "link": "https://www.example.com/contest",
            "created_at": "2024-02-13T10:00:00Z",
```

```
        }
    ]
}
```

（2）MySQL指令

```sql
CREATE TABLE games (
    id INT AUTO_INCREMENT,
    game_id VARCHAR(50) PRIMARY KEY COMMENT '赛事编号',
    name VARCHAR(255) NOT NULL COMMENT '赛事名称',
    wx_num VARCHAR(50) NOT NULL COMMENT '微信号',
    qq_num VARCHAR(20) NOT NULL COMMENT 'QQ号',
    introduction TEXT DEFAULT '暂无' COMMENT '赛事介绍',
    registration_start VARCHAR(50) NOT NULL COMMENT '报名开始时间',
    registration_end VARCHAR(50) NOT NULL COMMENT '报名结束时间',
    contest_start VARCHAR(50) NOT NULL COMMENT '赛事开始时间',
    contest_end VARCHAR(50) NOT NULL COMMENT '赛事结束时间',
    link VARCHAR(255) NOT NULL COMMENT '赛事链接',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间'
) COMMENT '赛事表';
```

（3）API

POST /api/games/add

说明：添加新赛事

- 请求

```json
{
    "name": "2024创新创业大赛",
    "wx_num": "lovelovesg",
    "qq_num": "52000000",
    "introduction": "面向全校学生的创新创业竞赛...",
    "registration_start": "2024-02-15T00:00:00Z",
    "registration_end": "2024-03-15T23:59:59Z",
    "contest_start": "2024-04-01T00:00:00Z",
    "contest_end": "2024-04-15T23:59:59Z",
    "link": "https://www.example.com/contest"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "赛事添加成功",
    "data": {
        "game_id": "GM20240213001"
    }
}
```

GET /api/games/list

说明：获取赛事列表

- 请求

```
无
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取赛事列表",
    "data": [
        {
            "game_id": "GM20240213001",
            "name": "2024创新创业大赛",
            "wx_num": "lovelovesg",
            "qq_num": "52000000",
            "introduction": "面向全校学生的创新创业竞赛...",
            "registration_start": "2024-02-15T00:00:00Z",
            "registration_end": "2024-03-15T23:59:59Z",
            "contest_start": "2024-04-01T00:00:00Z",
            "contest_end": "2024-04-15T23:59:59Z",
            "link": "https://www.example.com/contest",
        }
    ]
}

```

GET /api/games/detail/{game_id}

说明：获取赛事详情

- 请求

```
路径参数game_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取赛事详情",
    "data": {
        "game_id": "GM20240213001",
        "name": "2024创新创业大赛",
        "wx_num": "lovelovesg",
        "qq_num": "52000000",
        "introduction": "面向全校学生的创新创业竞赛...",
        "registration_start": "2024-02-15T00:00:00Z",
        "registration_end": "2024-03-15T23:59:59Z",
        "contest_start": "2024-04-01T00:00:00Z",
        "contest_end": "2024-04-15T23:59:59Z",
        "link": "https://www.example.com/contest",
    }
}
```

DELETE /api/games/delete/{game_id}

说明：删除赛事

- 请求

```
路径参数game_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "赛事删除成功"
    "data": {
        "game_id": "GM20240213001"
```

```
    }
}
```

（九）荣誉表(Honor)

(1) 说明

- `id` (类型: `Integer`)
  - **说明**：表示个人荣誉记录的唯一标识符。用于区分不同的荣誉条目。
- `name` (类型: `String`)
  - **说明**：表示个人的名字。该字段用来标识获得荣誉的人。
- `grade` (类型: `Integer`)
  - **说明**：表示个人所处的学年或年级。在这个例子中是 "2023"，可能代表该人物的毕业年份或学年。
- `role` (类型: `String`)
  - **说明**：表示个人的职位或角色。这里的 "前端开发" 说明该人是一个前端开发人员。

如果一个人有多个职位，则应该在 `honor` 数组中为该人创建多条记录。每条记录表示一个不同的职位。

```
{
    "honor": [
        {
            "id": 1,
            "name": "张三",
            "grade": 2023,
            "role": "前端开发"
        },
        {
            "id": 2,
            "name": "张三",
            "grade": 2023,
            "role": "项目经理"
        }
    ]
}
```

这里，张三有两个职位："前端开发"和"项目经理"，因此我们为张三分别记录了两条荣誉记录，每条记录的 `id` 值不同，`role` 字段表示不同的职位。

id	name	grade	role
1	岳一扬	2022	

```
{
    "honor": [
        {
            "id": 1,
            "name": "张三",
            "grade": 2023,
            "role": "前端开发"
        }
    ]
}
```

(2) MySQL指令

```
CREATE TABLE honor (
    id INT AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL COMMENT '成员姓名',
    grade INT NOT NULL COMMENT '年级',
    role VARCHAR(50) NOT NULL COMMENT '角色',
    INDEX idx_grade (grade),
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='荣誉表（贡献者名单）';
```

(3) API

POST /api/honor/add

说明：添加新贡献者

- 请求

```
{
    "name": "张三",
    "grade": 2023,
```

```
    "role": "前端开发",
    "department": "技术部",
    "contribution": "负责平台前端开发与维护",
    "join_time": "2023-09-01",
    "leave_time": "2024-06-30",  // 可选
    "state": 1
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "添加成功",
    "data": {
        "honor_id": "HR20240213001"
    }
}
```

GET /api/honor/list

说明：测试用接口，获取所有贡献者的列表

- 请求

```
{
    "page": 1,
    "size": 10,
    "grade": 2023,      // 可选，筛选年级
    "department": "技术部",  // 可选，筛选部门
    "state": 1          // 可选，筛选状态
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取所有贡献者",
    "data": {
        "total": 50,
        "current_page": 1,
```

```
        "page_size": 10,
        "members": [
            {
                "honor_id": "HR20240213001",
                "name": "张三",
                "grade": 2023,
                "role": "前端开发",
                "department": "技术部",
                "state": 1
            }
        ],
        "statistics": {
            "total_members": 50,
            "active_members": 40,
            "departments": {
                "技术部": 20,
                "设计部": 15,
                "运营部": 15
            }
        }
    }
}
```

~~DELETE /api/honor/delete~~

~~说明：删除贡献者记录~~

- ~~请求~~

```
路径参数honor_id
```

- ~~响应~~

```
{
    "code": 200,
    "status": "success",
    "message": "删除成功"
}
```

（十）值班记录表(DutyRecord)

（1）说明

1. `id`：整数类型(INT)，值班记录的唯一标识符。自增主键。
2. `name`：字符串类型(VARCHAR(50))，值班人员姓名，例如"张三"。
3. `userid`：字符串类型(VARCHAR(50))，值班人员的邮箱账号，唯一认证ID，例如"duty@example.com"。
4. `start_time`：日期时间类型(DATETIME)，值班开始时间，采用ISO 8601格式，例如"2024-02-13T09:00:00Z"。
5. `end_time`：日期时间类型(DATETIME)，值班结束时间，采用ISO 8601格式，例如"2024-02-13T12:00:00Z"。
6. `total`：整数类型(INT)，值班时长，以小时为单位，例如"3"表示值班3小时。
7. `created_at`：日期时间类型(DATETIME)，记录创建时间，采用ISO 8601格式，例如"2024-02-13T09:00:00Z"。

id	name	userid	start_time	end_time	total	created_at

```
{
    "duty_record": [
        {
            "id": 1,
            "name": "张三",
            "userid": "duty@example.com",
            "start_time": "2024-02-13T09:00:00Z",
            "end_time": "2024-02-13T12:00:00Z",
            "total": 3,  // 值班时长（小时）
            "created_at": "2024-02-13T09:00:00Z"
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE duty_record (
    id INT AUTO_INCREMENT PRIMARY KEY COMMENT '记录ID',
    name VARCHAR(50) NOT NULL COMMENT '值班人员姓名',
    userid VARCHAR(50) NOT NULL COMMENT '值班人员邮箱',
    start_time VARCHAR(50) NOT NULL COMMENT '值班开始时间',
```

```
    end_time VARCHAR(50) NOT NULL COMMENT '值班结束时间',
    total INT NOT NULL COMMENT '值班时长(小时)',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间'
) COMMENT '值班记录表';
```

（3）API

POST /api/dutyrecord/add

说明：添加值班记录

- 请求

```
{
    "name": "张三",
    "userid": "duty@example.com",
    "start_time": "2024-02-13T09:00:00Z",
    "end_time": "2024-02-13T12:00:00Z",
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "记录添加成功",
    "data": {
        "total": 3
    }
}
```

GET /api/dutyrecord/list

说明：获取值班记录列表

- 请求

```
{
    "name": "张三",          // 可选，按姓名筛选
    "start_date": "2024-02-01", // 可选，开始日期
    "end_date": "2024-02-28"    // 可选，结束日期
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取值班记录列表",
    "data": {
        "total": 100,
        "records": [
            {
                "record_id": "DR20240213001",
                "name": "张三",
                "start_time": "2024-02-13T09:00:00Z",
                "end_time": "2024-02-13T12:00:00Z",
                "total": 3,
            }
        ],
        }
    }
}
```

（十一）值班申请表(DutyApply)

（1）说明

- `id`：自增编号，主键
- `apply_id`:string类型，申请编号。
- `name`：string类型，请人姓名
- `userid`： 用户的邮箱id字段
- `day`：string类型，一星期中某个天的名称，有 ” 星期一 “到” 星期日 “这7个可能选择
- `time_section`：整型，1 - 6代表不同的时间段
  - `1`：08:10 - 10:05
  - `2`：10:15 - 12:20
  - `3`：12:30 - 14:30
  - `4`：14:30 - 16:30
  - `5`：16:30 - 18:30
  - `6`：18:30 - 20:30

- `created_at`：创建时间
- 关于多选时段：每个干事最多选3个值班时段，每一个选择作为一条记录

id	apply_id	name	userid	day	time_section	created_at

```
{
    "duty_apply": [
        {
            "id": 1,
            "apply_id":"DA20240213001"
            "userid": "duty@example.com",
            "name": "张三",
            "day": "星期一",
            "time_section": 1,  // 1: 08:10-10:05
            "created_at": "2024-02-13T10:00:00Z",
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE duty_apply (
    id INT AUTO_INCREMENT,
    apply_id VARCHAR(50) PRIMARY KEY COMMENT '申请编号',
    name VARCHAR(50) NOT NULL COMMENT '请人姓名',
    userid VARCHAR(50) NOT NULL COMMENT '用户邮箱',
    day VARCHAR(10) NOT NULL COMMENT '星期几',
    time_section TINYINT NOT NULL COMMENT '时间段: 1(08:10-10:05)、2(10:15-12:20)、3(12:30-14:30)、4(14:30-16:30)、5(16:30-18:30)、6(18:30-20:30)',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间'
) COMMENT '值班申请表';
```

（3）API

POST /api/duty_apply/add

说明：提交值班申请

- 请求

```
{
    "apply_id": "DA20240213001",
    "userid": "duty@example.com",
    "name": "张三",
    "day": "星期一",
    "time_section": 1,  // 1: 08:10-10:05
    "created_at": "2024-02-13T10:00:00Z"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "申请提交成功",
    "data": {
        "apply_id": "DA20240213001"
    }
}
```

GET /api/duty_apply/list

说明：获取值班申请列表

- 请求

```
无
```

- 响应

```
{
    "code":200,
    "status": "success",
    "data": [
        {
            "apply_id": "DA20240213001",
            "userid": "duty@example.com",
            "name": "张三",
            "day": "星期一",
            "time_section": 1,  // 1: 08:10-10:05
            "created_at": "2024-02-13T10:00:00Z"
        }
```

```
    ]
}

```

DELETE /api/duty_apply/delete/{apply_id}

说明：删除值班申请

- 请求

```
路径参数id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "申请删除成功"
    "data": {
    "apply_id": "DA20240213001"
    }
}
```

（十二）扫除记录表(Clean)

（1）说明

- `id`：整数类型，记录的唯一标识符。通常为自增，主键。
- `name`：字符串类型，相关人员的名称，例如“王远航”。
- `record_id`：记录编号。
- `userid`：字符串类型，相关人员的电子邮件地址（用户ID），例如“xxx@xxx”。
- `times`：整数类型，表示打扫次数，例如10。
- `created_at`：字符串类型，记录第一次记录创建的时间戳
- `updated_at`：字符串类型，记录最后一次修改times的时间戳

id	record_id	name	userid	times	created_at	updated_at


```
{
    "clean": [
        {
            "id": 1,
            "record_id": "CL20240213001",  // 建议添加记录编号
            "name": "张三",
            "userid": "clean@example.com",
            "times": 10,
            "created_at": "2024-02-13T10:00:00Z",
            "updated_at": "2024-02-13T10:00:00Z"
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE clean (
    id INT AUTO_INCREMENT,
    record_id VARCHAR(50) PRIMARY KEY COMMENT '记录编号',
    name VARCHAR(50) NOT NULL COMMENT '人员姓名',
    userid VARCHAR(50) NOT NULL COMMENT '用户邮箱',
    times INT NOT NULL DEFAULT 0 COMMENT '打扫次数',
    created_at VARCHAR(50) NOT NULL COMMENT '创建时间',
    updated_at VARCHAR(50) NOT NULL COMMENT '更新时间'
) COMMENT '扫除记录表';
```

（3）API

POST /api/clean/add

说明：添加打扫记录

- 请求

```
{
    "name": "张三",
    "userid": "clean@example.com",
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "记录添加成功",
    "data": {
        "record_id": "CL20240213001",
        "times": 1
    }
}
```

GET /api/clean/list

说明：获取打扫记录列表

- 请求

```
{
    "page": 1,
    "size": 10,
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取打扫记录",
    "data": {
        "total": 50,
        "current_page": 1,
        "page_size": 10,
        "records": [
            {
                "record_id": "CL20240213001",
                "name": "张三",
                "times": 10,
            }
        ]
    }
}
```

（十三）学年工作安排表（Arrange）

（1）说明

- `id`：整数类型，记录的唯一标识符。通常为自增，主键。
- `name`：string 类型，干事的姓名。
- `type`：string 类型，有 “活动文案”，“公众号”，“新闻稿”。
- `order`：int 整型，每个人类工作顺序。

id	name	type	order
1			
2			
3			
4			

```
{
    "arrange": [
        {
            "id": 1,
            "name": "lovesg",
            "type":"文案",
            "order": 1,
        },
        {
            "id": 2,
            "name": "lovesg",
            "type": "公众号",
            "order": 2
        },
        {
            "id": 3,
            "name": "lovesg",
            "type": "新闻稿",
            "order": 1
        },
    ]
}
```

（2）MySQL指令

```sql
CREATE TABLE arrange (
    id INT AUTO_INCREMENT PRIMARY KEY COMMENT '记录ID',
    name VARCHAR(50) NOT NULL COMMENT '干事姓名',
    type VARCHAR(20) NOT NULL COMMENT '工作类型：活动文案、公众号、新闻稿',
    `order` INT NOT NULL COMMENT '工作顺序'
) COMMENT '学年工作安排表';
```

（3）API

POST /api/arrange/add

说明：新增工作安排

- 请求

```json
{
    "name": "lovesg",
    "type": 1,
    "order": 1
}
```

- 响应

```json
{
    "code": 200,
    "status": "success",
    "message": "成功增加工作安排",
    "data": {
        "id": 1
    }
}
```

DELETE /api/arrange/delete

说明：删除一条安排

- 请求

```json
{
    "id": 1
```

```
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功删除一条安排"
}
```

GET /api/arrange/list

说明：获取工作安排列表

- 请求

```
{
    "name": "lovesg", // 可选,按姓名筛选
    "type": 1  // 可选,按类型筛选
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取工作安排",
    "data": {
        "total": 100,
        "list": [
            {
                "id": 1,
                "name": "lovesg",
                "type": 1,
                "order": 1,
                "create_time": "2024-02-13 10:00:00",
                "type_text": "活动文案"
            }
        ]
    }
}
```

GET /api/arrange/detail/{work_id}

说明：获取工作安排详情

- 请求

```
路径参数work_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取工作安排",
    "data": {
        "id": 1,
        "name": "lovesg",
        "type": 1,
        "order": 1,
        "create_time": "2024-02-13 10:00:00",
        "type_text": "活动文案"
    }
}
```

（十四）秀米链接表（PubulicityLink）

（1）说明

- `id`：整数类型，自增，主键。
- `name`:string类型，存储提交人的姓名。
- `userid`:string类型，储存邮箱。
- `link`:string类型，储存秀米链接。

id	name	userid	link
1			

```
[
```

```
    {
        "id": 1,
        "name": "张三",
        "userid": "zhangsan@example.com",
        "link": "https://xiumi.us/123456"
    },
    {
        "id": 2,
        "name": "李四",
        "userid": "lisi@example.com",
        "link": "https://xiumi.us/789012"
    },
    {
        "id": 3,
        "name": "王五",
        "userid": "wangwu@example.com",
        "link": "https://xiumi.us/345678"
    }
]
```

（2）MySQL指令

```
CREATE TABLE publicity_link (
    id INT AUTO_INCREMENT PRIMARY KEY COMMENT '记录ID',
    name VARCHAR(50) NOT NULL COMMENT '提交人姓名',
    userid VARCHAR(50) NOT NULL COMMENT '用户邮箱',
    link VARCHAR(255) NOT NULL COMMENT '秀米链接'
) COMMENT '秀米链接表';
```

（3）API

POST /api/publicity_link/add

说明：新增秀米链接

- 请求

```
{
    "name": "张三",
    "userid": "zhangsan@example.com",
    "link": "https://xiumi.us/123456"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取秀米链接",
    "data": {
        "id": 1,
        "submit_time": "2024-02-13 10:00:00"
    }
}
```

GET /api/publicity_link/list

说明：获取链接列表

- 请求

```
{
    "page": 1,
    "size": 10,
    "name": "张三",  // 可选,按提交人筛选
    "userid": "zhangsan@example.com", // 可选,按邮箱筛选
    "start_time": "2024-02-13 00:00:00", // 可选,开始时间
    "end_time": "2024-02-13 23:59:59" // 可选,结束时间
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取链接列表",
    "data": {
        "total": 100,
        "list": [
            {
                "id": 1,
                "name": "张三",
                "userid": "zhangsan@example.com",
                "link": "https://xiumi.us/123456",
                "submit_time": "2024-02-13 10:00:00",
                "create_time": "2024-02-13 10:00:00"
```

```
            }
        ]
    }
}
```

GET /api/publicity_link/{link_id}

说明：获取链接详情

- 请求

```
路径参数link_id
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功获取链接详情",
    "data": {
        "id": 1,
        "name": "张三",
        "userid": "zhangsan@example.com",
        "link": "https://xiumi.us/123456",
        "submit_time": "2024-02-13 10:00:00",
        "create_time": "2024-02-13 10:00:00"
    }
}
```

DELETE /api/publicity_link/delete/{link_id}

说明：删除链接

- 请求

```
{
    "name": "赵六",
    "userid": "zhaoliu@example.com",
    "link": "https://xiumi.us/987654"
}
```

- 响应

```
{
    "code": 200,
    "status": "success",
    "message": "成功删除链接"
}
```

（十五）消息表（Messages）

（1）说明

- 信息池需要从几个地方获取，任务表，项目表，借物表，场地借用表，秀米链接表。

任务表：

增加任务条之后，立刻将此条任务提取出来，处理成信息的格式，存在信息表中。

项目表：

- 增加项目条之后，立刻将此条项目提取出来，处理成信息的格式，存在信息表中（这条信息发给**项目部部长**）
- 视奸项目状态键，一旦状态键（默认为0）更改为 "1"（审核通过）或者 "2"（审核未通过）就提取项目详情，处理成信息的格式，存储到信息表。（这条信息发给**被审核者**）

借物表：

- 增加借物条之后，立刻将此条项目提取出来，处理成信息的格式，存在信息表中（这条信息发给**基管部部长**）
- 视奸借物状态键，一旦状态键（默认为0）更改为 "1"（审核通过）或者 "2"（审核未通过）就提取项目详情，处理成信息的格式，存储到信息表。（这条信息发给**被审核者**）

场地借用表：

- 增加场地借用条之后，立刻将此条项目提取出来，处理成信息的格式，存在信息表中（这条信息发给**基管部部长**）
- 视奸场地申请审核状态键，一旦状态键（默认为0）更改为 "1"（审核通过）或者 "2"（审核未通过）就提取项目详情，处理成信息的格式，存储到信息表。（这条信息发给**被审核者**）

秀米链接表：

- 增加秀米链接条之后，立刻将此条项目提取出来，处理成信息的格式，存在信息表中（这条信息发给**宣传部部长**）
- 视奸场地申请审核状态键，一旦状态键（默认为0）更改为 "1"（审核通过）或者 "2"（审核未通过）就提取项目详情，处理成信息的格式，存储到信息表。（这条信息发给**被审核者**）

信息表有一个监测函数，一旦监测到信息表有新增，就将新增信息对应发给接收者。

用户每次点开历史信息页面，**下拉刷新**，根据自己的id在信息表中查询id对应的所有信息（前端添加缓存，不必每次都向后端发GET请求）

用户每次点开消息详情，向后端发送POST，将该信息标为已读。

（**升级考虑**，用户量和信息量如果增大，考虑优化方向：1.分表：即，将用户id分组，通过索引找到对应的小表，减少响应时间。2.使用消息队列（如RabbitMQ，Kafka等）：使用消息队列时，消息不会直接写入数据库，而是先发送到队列中，后台服务会定时从队列中读取消息并将其存储到数据库中。这种方式不仅有助于解耦，还能提高写入性能，这也可以提高消息的实时性（高并发）。3.考虑给某些id（比如协会内成员的id）建立索引，能够快速定位数据，减少全表扫描的情况。）

- `id`：整数类型，消息的唯一标识符。通常为自增，主键。
- `message_id`：字符串类型，消息本身的ID。
- `sender_id`：字符串类型，消息发送者的 ID（如电子邮件地址）。
- `receiver_id`：字符串类型，消息接收者的 ID（如电子邮件地址）。
- `content`：字符串类型，消息的内容，描述消息的详细信息。
- `time_stamp`：字符串类型，消息的时间戳，采用 ISO 8601 格式，如“2025-01-31T12:00:00Z”。
- `status`：整数类型，表示消息的阅读状态。`0` 表示未读，`1` 表示已读。

id	message_id	sender_id	receiver_id	content	time_stamp	state

```
{
    "messages": [
        {
            "id": 1,
            "message_id": "MS20240213001",
            "sender_id": "xx@xx",
            "receiver_id": "YY@yy",
            "content": "情人节快乐我的朋友",
            "time_stamp":"2025-01-31T12:00:00Z",
            "state": 0                                  //0表示未读，1表示已读
        },
        {
            "id": 2,
            "message_id": "MS20240213002",
```

```
            "sender_id": "xx@xx",
            "receiver_id": "YY@yy",
            "content": "场地借用成功了我的朋友",
            "time_stamp":"2025-01-31T12:00:00Z",
            "state": 1
        },
        {
            "id": 3,
            "message_id": "MS20240213003",
            "sender_id": "xx@xx",
            "receiver_id": "YY@yy",
            "content": "借物失败了我的朋友",
            "time_stamp":"2025-01-31T12:00:00Z",
            "state": 1
        }
    ]
}
```

（2）MySQL指令

```
CREATE TABLE messages (
    id INT AUTO_INCREMENT,
    message_id VARCHAR(50) PRIMARY KEY COMMENT '消息ID',
    sender_id VARCHAR(50) NOT NULL COMMENT '发送者邮箱',
    receiver_id VARCHAR(50) NOT NULL COMMENT '接收者邮箱',
    content TEXT NOT NULL COMMENT '消息内容',
    time_stamp VARCHAR(50) NOT NULL COMMENT '时间戳',
    state TINYINT NOT NULL DEFAULT 0 COMMENT '阅读状态：0未读、1已读'
) COMMENT '消息表';
```

（3）API

POST /api/messages/add

说明：增加一条信息条目，这里以后可以有某个页面发送信息的功能，暂时留白。

- 请求

```
{
  "sender_id": "xx@xx",
  "receiver_id": "YY@yy",
  "content": "任务已完成，检查最新进展",
  "time_stamp": "2025-01-31T12:00:00Z"
}
```

```
7
```

- 响应

```
{
  "status": "success",
  "message": "Message added successfully",
  "message_id": 1
}

```

GET /api/messages/{receiver_id}

说明：拉取对应的收信人id（邮箱）（收信人id就是它的邮箱，在用户信息表（Users）中的命名空间为userid）的所有信息。

- 请求

```
URL参数：{receiver_id}，表示收信人邮箱ID。
```

- 响应

```
{
    "code":200,
    "status": "success",
     "messages": [
        {
          "id": 1,
          "message_id": "MS20240213001",
          "sender_id": "xx@xx",
          "receiver_id": "YY@yy",
          "content": "该上工了朋友",
          "time_stamp": "2025-01-31T12:00:00Z",
          "state": 0
        },
        {
          "id": 2,
          "message_id": "MS20240213001",
          "sender_id": "xx@xx",
          "receiver_id": "YY@yy",
          "content": "场地借用成功了我的朋友",
          "time_stamp": "2025-01-31T12:00:00Z",
```

```
            "state": 1
        }
    ]
}
```

GET /api/messages/detail/{message_id}

说明：通过message_id拉取某一条信息。此接口用于获取某一条消息的详细信息，帮助用户查看单条消息的内容。

- 请求

```
URL参数：{message_id}，表示消息的唯一标识符。
```

- 响应

```
{
    "code":200,
    "status": "success",
    "message": {
        "id": 1,
        "sender_id": "xx@xx",
        "receiver_id": "YY@yy",
        "content": "该上工了朋友",
        "time_stamp": "2025-01-31T12:00:00Z",
        "status": 0
    }
}
```

POST /api/messages/read/{message_id}

说明：将指定消息标记为已读。用户点击消息详情后，通过此接口将消息标记为已读。接口会更新消息的状态字段为1。

- 请求

```
URL参数：{message_id}，表示消息的唯一标识符。
```

- 响应

```
{
  "code":200,
  "status": "success",
  "message": "Message marked as read"
}
```

DELETE /api/messages/delete/{message_id}

说明：通过messages_id删除某条信息。这个接口允许用户删除某一条消息，系统会根据消息ID删除对应的记录。

- 请求

```
URL参数：{message_id}，表示消息的唯一标识符。
```

- 响应

```
{
  "code":200,
  "status": "success",
  "message": "Message deleted successfully"
}

```