

硅谷课堂第十四天-直播管理模块

硅谷课堂第十四天-直播管理模块

一、后台系统-直播管理

- 1、获取 openId 与 openToken
- 2、对接说明

3、了解接口文档

3.1、了解接口文档

- 3.1.1、添加直播
- 3.1.2、更新直播信息
- 3.1.3、删除直播信息
- 3.1.4、修改生活直播相关配置
- 3.1.5、按照课程 ID 获取访客列表

▪3.2、下载 SDK

5、搭建 service_live 模块

- 5.1、创建 service_live 模块
- 5.2、添加依赖
- 5.3、集成代码
- 5.4、更改配置
- 5.5、创建配置文件和启动类
- 5.6、生成相关代码

6、功能实现-直播课程列表接口

- 6.1、LiveCourseController 类
- 6.2、LiveCourseService 接口
- 6.3、service_vod 模块创建接口
- 6.4、service_live 引入依赖
- 6.5、LiveCourseServiceImpl 实现

7、功能实现-直播课程添加接口

- 7.1、添加工具类
- 7.2、LiveCourseController 类
- 7.3、LiveCourseService 接口

- 7.4、LiveCourseServiceImpl 实现
- 8、功能实现-直播课程删除接口
 - 8.1、LiveCourseController 类
 - 8.2、LiveCourseService 接口
 - 8.3、LiveCourseServiceImpl 实现
- 9、功能实现-直播课程修改接口
 - 9.1、LiveCourseController 类
 - 9.2、LiveCourseService 接口
 - 9.3、LiveCourseServiceImpl 实现
 - 9.4、LiveCourseDescriptionService 添加方法
 - 9.5、LiveCourseDescriptionServiceImpl 实现方法
- 10、功能实现-查看账号接口
 - 10.1、LiveCourseController 类
 - 10.2、LiveCourseAccountService 接口
 - 10.3、LiveCourseAccountServiceImpl 实现
- 11、功能实现-配置和观看记录接口
 - 11.1、查看配置信息
 - 11.2、修改直播配置信息
 - 11.3、获取最近直播课程

一、后台系统-直播管理

上面我们已经开通了“生活类直播”。

1、获取 openId 与 openToken

登录进入开放后台，后台首页即可获取 openId 与 openToken



 合作商: [REDACTED]

 帐 号: [REDACTED]

 用户ID: [REDACTED]

 密 码: ***** [修改密码](#)

Open ID: [REDACTED]

Open Token: [REDACTED]

2、对接说明

- 1、使用 HTTP 协议进行信息交互，字符编码统一采用 UTF-8
- 2、除非特殊说明，接口地址统一为：<https://api.talk-fun.com/portal.php>
- 3、除非特殊说明，同时支持 GET 和 POST 两种参数传递方式
- 4、除非特殊说明，返回信息支持 JSON 格式
- 5、除了 sign 外，其余所有请求参数值都需要进行 URL 编码
- 6、参数表中，类型一栏声明的定义为：int 代表整数类型；string 代表字符串类型，如果后面有括号，括号中的数字代表该参数的最大长度；array / object 表示数组类型
- 7、openID、openToken 参数的获取见[对接流程说明](#)

3、了解接口文档

接口文档地址：https://open.talk-fun.com/docs/getstartV2/api/live_dir.html

3.1、了解接口文档

根据接口文档，了解我们需要对接哪些接口



3.1.1、添加直播

api 名称 : `course.add` , SDK 对应方法 : `courseAdd`

添加直播是一定需要的

3.1.2、更新直播信息

api 名称 : `course.update` , SDK 对应方法 `courseUpdate`

3.1.3、删除直播信息

api 名称 : `course.delete` , SDK 对应方法 : `courseDelete`

3.1.4、修改生活直播相关配置

api 名称 : `course.updateLifeConfig` , SDK 对应方法 : `updateLifeConfig`

设置功能很多, 但是我们只需要几个即可, 这个接口我们需要做如下设置 :

1、**界面模式** : `pageViewMode` 界面模式, 1 全屏模式 0 二分屏 2 课件模式

2、**观看人数开关** : `number` 观看人数开关 ; `number.enable` 是否开启 ; 观看人数, 0 否 1 是 ;
示例 : `{"enable": "1"}`

3、**商城开关** (直播推荐课程) : `goodsListEdit` 商品列表编辑, 状态 `goodsListEdit.status`, 0 覆盖, 1 追加, 不传默认为 0 ; 示例 : `{"status": 1}`

直播设置最终效果 :


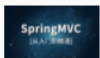


功能设置

界面模式 ☒ 全屏模式 ☐ 二分屏 ☐ 课件模式观看人数开关 ☒ 是 ☐ 否商城开关: ☒ 是 ☐ 否

商品列表

[添加](#)

序号	商品图片	名称	价格	原价	操作
1		JAVA之MySQL基础	1000	1000	删除
2		14417人 分享 收藏 SpringMVC	22800	22800	删除

[保存](#)[返回](#)

3.1.5、按照课程 ID 获取访客列表

改接口在："访客/管理员列表"下面

通过该接口统计课程观看人数信息

直播访客 api 名称：`course.visitor.list`，SDK 对应方法：`courseVisitorList`

3.2、下载 SDK

直播平台为我们准备了 SDK，我们直接使用

下载地址：<https://open.talk-fun.com/docs/getstartV2/api/introduce/sdkdownload.html>

已下载：当前目录/MTCloud-java-sdk-1.6.zip

请输入关键词搜索

关键词搜索

欢拓云直播课程接入指南

> 简介

> 常见问题指南

> 如何进行对接

> 接口文档

> 服务端API接口文档

协议说明

请求频率限制

> 课程管理

SDK下载

• 使用方式

请根据客户自身的网站开发语言，下载对应开发语言的SDK包，集成进入项目内，按照开发包内的示例以及文档的说明，进行调用。

比如：客户的网站是由php语言开发的，请下载php sdk解压并集成进入项目

php-sdk-1.6

java-sdk-1.6

python-sdk-1.3

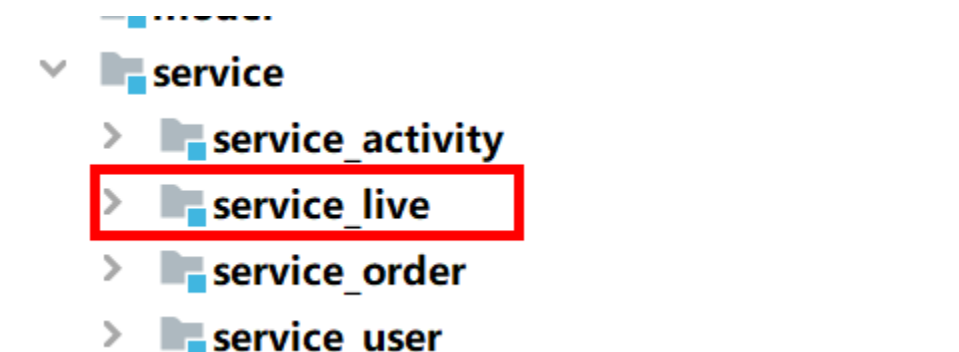
.net-sdk-1.6

nodejs-sdk 通过npm安装

```
$ npm config set registry http://registry.npmjs.org/
$ npm install talkfunSDK
```

5、搭建 service_live 模块

5.1、创建 service_live 模块



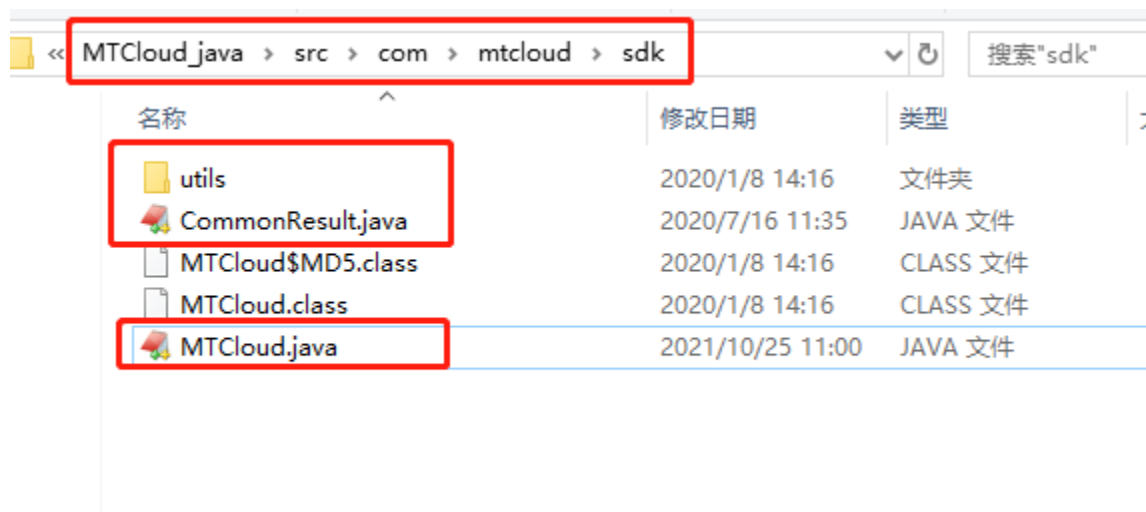
5.2、添加依赖

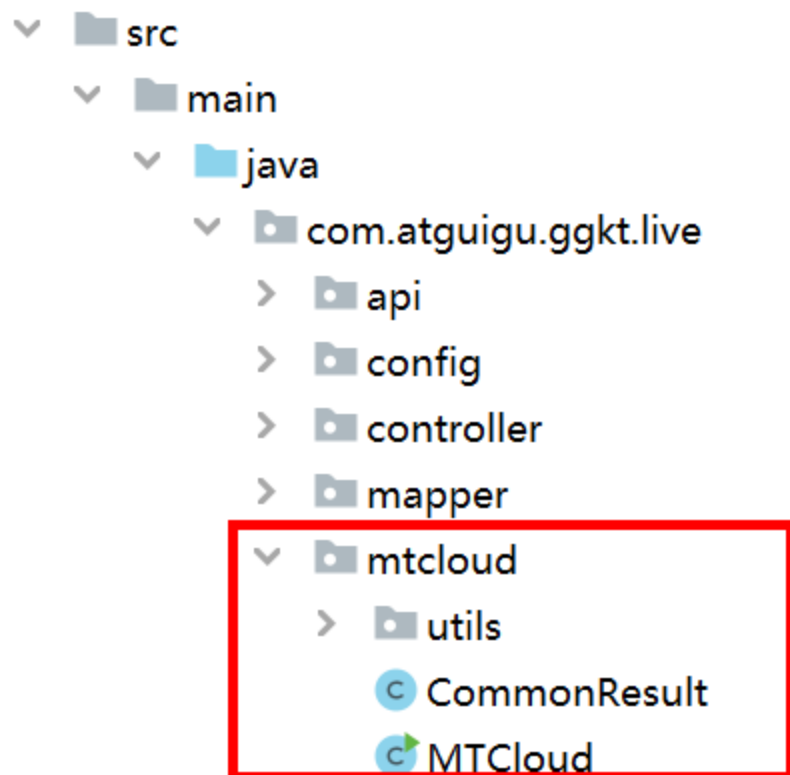
添加直播 SDK 需要的依赖

```
<!-- 直播 -->
<dependency>
    <groupId>commons-httpclient</groupId>
    <artifactId>commons-httpclient</artifactId>
    <version>3.0.1</version>
</dependency>
<dependency>
    <groupId>net.sf.json-lib</groupId>
    <artifactId>json-lib</artifactId>
    <version>2.4</version>
    <classifier>jdk15</classifier>
</dependency>
```

5.3、集成代码

解压 MTCloud-java-sdk-1.6.zip，复制 MTCloud-java-sdk-1.6\MTCloud_java\src\com\mtcloud\ sdk 下面的 java 文件到 com.atguigu.ggkt.live.mtcloud 包下，如图





5.4、更改配置

更改 MTCloud 类配置

说明：

- 1、更改 openID 与 openToken
- 2、该类官方已经做了接口集成，我们可以直接使用。

```
public class MTCloud {  
    /**  
     * 合作方ID: 合作方在欢拓平台的唯一ID  
     */  
    public String openID = "37013";  
  
    /**  
     * 合作方密钥: 合作方ID对应的参数加密密钥  
     */  
    public String openToken = "5cfa64c1be5f479aea8296bb4e2c37d3";  
}
```


5.5、创建配置文件和启动类

(1) application.properties

```
# 服务端口
server.port=8306

# 服务名
spring.application.name=service-live

# 环境设置: dev、test、prod
spring.profiles.active=dev

# mysql数据库连接
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/glkt_live?characterEncoding=utf-8&useSSL=false
spring.datasource.username=root
spring.datasource.password=root

# 返回json的全局时间格式
spring.jackson.date-format=yyyy-MM-dd HH:mm:ss
spring.jackson.time-zone=GMT+8

# mybatis日志
mybatis-plus.configuration.log-impl=org.apache.ibatis.logging.stdout.StdoutImpl

mybatis-plus.mapper-locations=classpath:com/atguigu/ggkt/live/mapper/xml/*.xml

# nacos服务地址
spring.cloud.nacos.discovery.server-addr=127.0.0.1:8848

mtcloud.openId=43873
mtcloud.openToken=1f3681df876eb31474be8c479b9f1ffe
```





















(2) 启动类

```
@SpringBootApplication
@EnableDiscoveryClient
@EnableFeignClients(basePackages = "com.atguigu")
@ComponentScan(basePackages = "com.atguigu")
@MapperScan("com.atguigu.ggkt.live.mapper")
public class ServiceLiveApplication {

    public static void main(String[] args) {
        SpringApplication.run(ServiceLiveApplication.class, args);
    }

}
```

5.6、生成相关代码

- ▼  controller
 -  LiveCourseController
 -  LiveVisitorController
- ▼  mapper
 - >  xml
 -  LiveCourseAccountMapper
 -  LiveCourseConfigMapper
 -  LiveCourseDescriptionMapper
 -  LiveCourseGoodsMapper
 -  LiveCourseMapper
 -  LiveVisitorMapper
- >  mtcloud
- ▼  service
 - >  impl
 -  LiveCourseAccountService
 -  LiveCourseConfigService
 -  LiveCourseDescriptionService
 -  LiveCourseGoodsService
 -  LiveCourseService
 -  LiveVisitorService

6、功能实现-直播课程列表接口

Dashboard / 直播管理 / 直播列表



数据列表 添加

序号	封面	直播名称	直播时间	直播结束时间	直播老师	头衔	创建时间	操作
1		测试第三次	2022年03月07 15:50至16:20	2022-03-07 16:20:00	张老师	高级讲师	2022-03-07 15:42:07	修改 删除 查看账号 配置 观看记录

根据直播平台与我们自身业务设计直播相关的业务表，如：glkt_live

6.1、LiveCourseController 类

```
@RestController
@RequestMapping(value="/admin/live/liveCourse")
public class LiveCourseController {

    @Autowired
    private LiveCourseService liveCourseService;

    @Autowired
    private LiveCourseAccountService liveCourseAccountService;

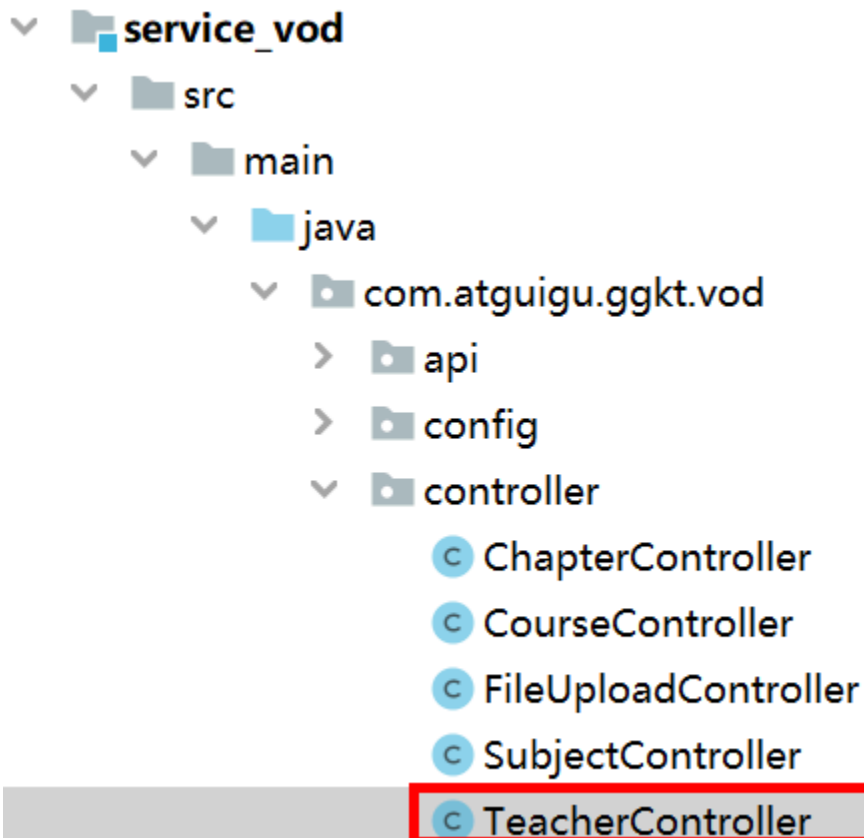
    @ApiOperation(value = "获取分页列表")
    @GetMapping("/{page}/{limit}")
    public Result index(
        @ApiParam(name = "page", value = "当前页码", required = true)
        @PathVariable Long page,
        @ApiParam(name = "limit", value = "每页记录数", required = true)
        @PathVariable Long limit) {
        Page<LiveCourse> pageParam = new Page<>(page, limit);
        IPage<LiveCourse> pageModel = liveCourseService.selectPage(pageParam);
        return Result.ok(pageModel);
    }
}
```

6.2、LiveCourseService 接口

```
public interface LiveCourseService extends IService<LiveCourse> {  
    //直播课程分页查询  
    IPage<LiveCourse> selectPage(Page<LiveCourse> pageParam);  
}
```

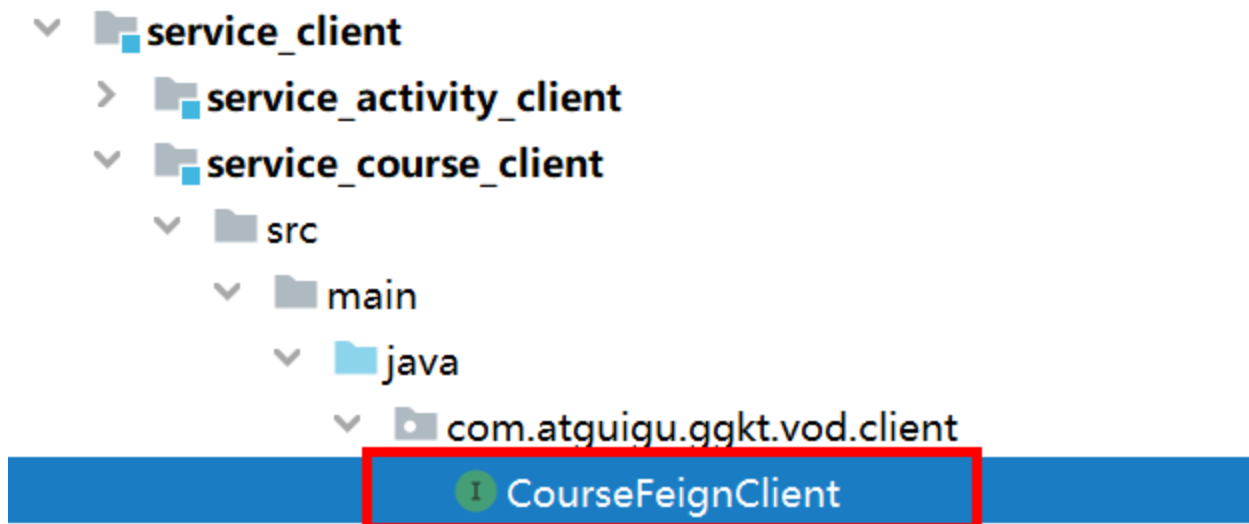
6.3、service_vod 模块创建接口

(1) 获取讲师信息



```
@ApiOperation("根据id查询")  
@GetMapping("inner/getTeacher/{id}")  
public Teacher getTeacherLive(@PathVariable Long id) {  
    Teacher teacher = teacherService.getById(id);  
    return teacher;  
}
```

(2) service_course_client 定义接口



```
@GetMapping("/admin/vod/teacher/inner/getTeacher/{id}")  
Teacher getTeacherLive(@PathVariable Long id);
```

6.4、service_live 引入依赖

```
<dependency>  
  <groupId>com.atguigu</groupId>  
  <artifactId>service_course_client</artifactId>  
  <version>0.0.1-SNAPSHOT</version>  
</dependency>
```

6.5、LiveCourseServiceImpl 实现

```
@Service
public class LiveCourseServiceImpl extends ServiceImpl<LiveCourseMapper, LiveCourse> implements L

    @Autowired
    private CourseFeignClient courseFeignClient;

    //直播课程分页查询
    @Override
    public IPage<LiveCourse> selectPage(Page<LiveCourse> pageParam) {
        IPage<LiveCourse> page = baseMapper.selectPage(pageParam, null);
        List<LiveCourse> liveCourseList = page.getRecords();

        for(LiveCourse liveCourse : liveCourseList) {
            Teacher teacher = courseFeignClient.getTeacherLive(liveCourse.getTeacherId());
            liveCourse.getParam().put("teacherName", teacher.getName());
            liveCourse.getParam().put("teacherLevel", teacher.getLevel());
        }
        return page;
    }
}
```

7、功能实现-直播课程添加接口

添加/修改

直播讲师

请选择

▼

直播讲师登录密码

直播名称

直播开始时间

🕒

选择开始日期

直播结束时间

🕒

选择结束日期

直播封面



7.1、添加工具类

▼

com.atguigu.ggkt.live

>

api

▼

config

⦿

MTCloudAccountConfig

MTCloudConfig

(1) MTCloudAccountConfig 类


```

@Data
@Component
@ConfigurationProperties(prefix = "mtcloud")
public class MTCloudAccountConfig {

    private String openId;
    private String openToken;

}

```

(2) MTCloudConfig 类

```

@Component
public class MTCloudConfig {

    @Autowired
    private MTCloudAccountConfig mtCloudAccountConfig;

    @Bean
    public MTCloud mtCloudClient(){
        return new MTCloud(mtCloudAccountConfig.getOpenId(), mtCloudAccountConfig.getOpenToken());
    }

}

```

7.2、LiveCourseController 类

```

@ApiOperation(value = "新增")
@PostMapping("save")
public Result save(@RequestBody LiveCourseFormVo liveCourseVo) {
    liveCourseService.save(liveCourseVo);
    return Result.ok(null);
}

```

7.3、LiveCourseService 接口

```
Boolean save(LiveCourseFormVo liveCourseVo);
```

7.4、LiveCourseServiceImpl 实现

```

@Resource
private LiveCourseAccountService liveCourseAccountService;

@Resource
private LiveCourseDescriptionService liveCourseDescriptionService;

@Autowired
private CourseFeignClient teacherFeignClient;

@Resource
private MTCloud mtCloudClient;
@SneakyThrows
@Transactional(rollbackFor = {Exception.class})
@Override
public Boolean save(LiveCourseFormVo liveCourseFormVo) {
    LiveCourse liveCourse = new LiveCourse();
    BeanUtils.copyProperties(liveCourseFormVo, liveCourse);

    Teacher teacher = teacherFeignClient.getTeacherLive(liveCourseFormVo.getTeacherId());
    HashMap<Object, Object> options = new HashMap<>();
    options.put("scenes", 2); // 直播类型。1: 教育直播, 2: 生活直播。默认 1, 说明: 根据平台开通的直播类
    options.put("password", liveCourseFormVo.getPassword());
    String res = mtCloudClient.courseAdd(liveCourse.getCourseName(), teacher.getId().toString(),

    System.out.println("return:: "+res);
    CommonResult<JSONObject> commonResult = JSON.parseObject(res, CommonResult.class);
    if(Integer.parseInt(commonResult.getCode()) == MTCloud.CODE_SUCCESS) {
        JSONObject object = commonResult.getData();
        liveCourse.setCourseId(object.getLong("course_id"));
        baseMapper.insert(liveCourse);

        //保存课程详情信息
        LiveCourseDescription liveCourseDescription = new LiveCourseDescription();
        liveCourseDescription.setDescription(liveCourseFormVo.getDescription());
        liveCourseDescription.setLiveCourseId(liveCourse.getId());
        liveCourseDescriptionService.save(liveCourseDescription);
    }
}

```

```

//保存课程账号信息
LiveCourseAccount liveCourseAccount = new LiveCourseAccount();
liveCourseAccount.setLiveCourseId(liveCourse.getId());
liveCourseAccount.setZhuboAccount(object.getString("bid"));
liveCourseAccount.setZhuboPassword(liveCourseFormVo.getPassword());
liveCourseAccount.setAdminKey(object.getString("admin_key"));
liveCourseAccount.setUserKey(object.getString("user_key"));
liveCourseAccount.setZhuboKey(object.getString("zhubo_key"));
liveCourseAccountService.save(liveCourseAccount);
} else {
    String getmsg = commonResult.getmsg();
    throw new GlktException(20001, getmsg);
}
return true;
}

```

8、功能实现-直播课程删除接口

数据列表 添加

序号	封面	直播名称	直播时间	直播结束时间	直播老师	头衔	创建时间	操作
1		测试第三次	2022年03月07 15:50至16:20	2022-03-07 16:20:00	张老师	高级讲师	2022-03-07 15:42:07	修改 删除 查看账号 配置 观看记录

8.1、LiveCourseController 类

```

@ApiOperation(value = "删除")
@DeleteMapping("remove/{id}")
public Result remove(@PathVariable Long id) {
    liveCourseService.removeLive(id);
    return Result.ok(null);
}

```

8.2、LiveCourseService 接口

```
//删除直播课程  
void removeLive(Long id);
```

8.3、LiveCourseServiceImpl 实现

```
//删除直播课程  
@Override  
public void removeLive(Long id) {  
    //根据id查询直播课程信息  
    LiveCourse liveCourse = baseMapper.selectById(id);  
    if(liveCourse != null) {  
        //获取直播courseid  
        Long courseId = liveCourse.getCourseId();  
        try {  
            //调用方法删除平台直播课程  
            mtCloudClient.courseDelete(courseId.toString());  
            //删除表数据  
            baseMapper.deleteById(id);  
        } catch (Exception e) {  
            e.printStackTrace();  
            throw new GgktException(20001, "删除直播课程失败");  
        }  
    }  
}
```

9、功能实现-直播课程修改接口

添加/修改

直播讲师

张老师

直播名称

测试第三次


直播开始时间

🕒 2022-03-07 15:50:00

直播结束时间

🕒 2022-03-07 16:20:00

直播封面



	操作
-07	<div>修改 删除 查看账号</div> <div>观看记录</div>
-07	<div>修改 删除 查看账号</div>

9.1、LiveCourseController 类

```
@ApiOperation(value = "获取")
@GetMapping("get/{id}")
public Result<LiveCourse> get(@PathVariable Long id) {
    LiveCourse liveCourse = liveCourseService.getById(id);
    return Result.ok(liveCourse);
}

@ApiOperation(value = "获取")
@GetMapping("getInfo/{id}")
public Result<LiveCourseFormVo> getInfo(@PathVariable Long id) {
    return Result.ok(liveCourseService.getLiveCourseFormVo(id));
}

@ApiOperation(value = "修改")
@PutMapping("update")
public Result updateById(@RequestBody LiveCourseFormVo liveCourseVo) {
    liveCourseService.updateById(liveCourseVo);
    return Result.ok(null);
}
```

9.2、LiveCourseService 接口

```
//修改
void updateById(LiveCourseFormVo liveCourseVo);

//获取
LiveCourseFormVo getLiveCourseFormVo(Long id);
```


9.3、LiveCourseServiceImpl 实现

```
@Resource
private LiveCourseAccountService liveCourseAccountService;

@Resource
private LiveCourseDescriptionService liveCourseDescriptionService;

@Autowired
private CourseFeignClient teacherFeignClient;

@Resource
private MTCloud mtCloudClient;

//更新
@Override
public void updateLiveById(LiveCourseFormVo liveCourseFormVo) {
    //根据id获取直播课程基本信息
    LiveCourse liveCourse = baseMapper.selectById(liveCourseFormVo.getId());
    BeanUtils.copyProperties(liveCourseFormVo, liveCourse);
    //讲师
    Teacher teacher =
        teacherFeignClient.getTeacherInfo(liveCourseFormVo.getTeacherId());

    // course_id    课程ID
    // account      发起直播课程的主播账号
    // course_name  课程名称
    // start_time   课程开始时间,格式:2015-01-01 12:00:00
    // end_time     课程结束时间,格式:2015-01-01 13:00:00
    // nickname     主播的昵称
    // accountIntro 主播的简介
    // options      可选参数
    HashMap<Object, Object> options = new HashMap<>();
    try {
        String res = mtCloudClient.courseUpdate(liveCourse.getCourseId().toString(),
            teacher.getId().toString(),
            liveCourse.getCourseName(),
            new DateTime(liveCourse.getStartTime()).toString("yyyy-MM-dd HH:mm:ss"),
            new DateTime(liveCourse.getEndTime()).toString("yyyy-MM-dd HH:mm:ss"),
```

```

        teacher.getName(),
        teacher.getIntro(),
        options);
//返回结果转换, 判断是否成功
CommonResult<JSONObject> commonResult = JSON.parseObject(res, CommonResult.class);
if(Integer.parseInt(commonResult.getCode()) == MTCloud.CODE_SUCCESS) {
    JSONObject object = commonResult.getData();
    //更新直播课程基本信息
    liveCourse.setCourseId(object.getLong("course_id"));
    baseMapper.updateById(liveCourse);
    //直播课程描述信息更新
    LiveCourseDescription liveCourseDescription =
        liveCourseDescriptionService.getLiveCourseById(liveCourse.getId());
    liveCourseDescription.setDescription(liveCourseFormVo.getDescription());
    liveCourseDescriptionService.updateById(liveCourseDescription);
} else {
    throw new GgktException(20001, "修改直播课程失败");
}
} catch (Exception e) {
    e.printStackTrace();
}
}

@Override
public LiveCourseFormVo getLiveCourseFormVo(Long id) {
    LiveCourse liveCourse = this.getById(id);
    LiveCourseDescription liveCourseDescription = liveCourseDescriptionService.getByLiveCourseId(

    LiveCourseFormVo liveCourseFormVo = new LiveCourseFormVo();
    BeanUtils.copyProperties(liveCourse, liveCourseFormVo);
    liveCourseFormVo.setDescription(liveCourseDescription.getDescription());
    return liveCourseFormVo;
}

```

9.4、LiveCourseDescriptionService 添加方法

```
public interface LiveCourseDescriptionService extends IService<LiveCourseDescription> {  
    LiveCourseDescription getByLiveCourseId(Long liveCourseId);  
}
```

9.5、LiveCourseDescriptionServiceImpl 实现方法

```
@Service  
public class LiveCourseDescriptionServiceImpl extends ServiceImpl<LiveCourseDescriptionMapper, LiveCourseDescription> {  
  
    @Override  
    public LiveCourseDescription getByLiveCourseId(Long liveCourseId) {  
        return this.getOne(new LambdaQueryWrapper<LiveCourseDescription>().eq(LiveCourseDescription::getLiveCourseId, liveCourseId));  
    }  
}
```

10、功能实现-查看账号接口

封面	直播名称	直播时间	直播结束时间	直播老师	头衔	创建时间	操作
	测试第三次	2022年03月07 15:50至16:20	2022-03-07 16:20:00	张老师	高级讲师	2022-03-07 15:42:07	修改 删除 查看账号 配置 观看记录

10.1、LiveCourseController 类

```
@Autowired  
private LiveCourseAccountService liveCourseAccountService;  
  
@ApiOperation(value = "获取")  
@GetMapping("getLiveCourseAccount/{id}")  
public Result<LiveCourseAccount> getLiveCourseAccount(@PathVariable Long id) {  
    return Result.ok(liveCourseAccountService.getByLiveCourseId(id));  
}
```

10.2、LiveCourseAccountService 接口

```
public interface LiveCourseAccountService extends IService<LiveCourseAccount> {  
    LiveCourseAccount getByLiveCourseId(Long liveCourseId);  
}
```

10.3、LiveCourseAccountServiceImpl 实现

```
@Service  
public class LiveCourseAccountServiceImpl extends ServiceImpl<LiveCourseAccountMapper, LiveCourseAccount> {  
  
    @Override  
    public LiveCourseAccount getByLiveCourseId(Long liveCourseId) {  
        return baseMapper.selectOne(new LambdaQueryWrapper<LiveCourseAccount>().eq(LiveCourseAccount::getLiveCourseId, liveCourseId));  
    }  
}
```

11、功能实现-配置和观看记录接口

封面	直播名称	直播时间	直播结束时间	直播老师	头衔	创建时间	操作
	测试第三次	2022年03月07 15:50至16:20	2022-03-07 16:20:00	张老师	高级讲师	2022-03-07 15:42:07	修改 删除 查看账号 配置 观看记录

11.1、查看配置信息

(1) LiveCourseController 类

```
@ApiOperation(value = "获取")  
@GetMapping("getCourseConfig/{id}")  
public Result getCourseConfig(@PathVariable Long id) {  
    return Result.ok(liveCourseService.getCourseConfig(id));  
}
```

(2) LiveCourseService 添加方法

```
//获取配置
LiveCourseConfigVo getCourseConfig(Long id);
```

(3) LiveCourseServiceImpl 实现

```
@Autowired
private LiveCourseConfigService liveCourseConfigService;

@Autowired
private LiveCourseGoodsService liveCourseGoodsService;

@Override
public LiveCourseConfigVo getCourseConfig(Long id) {
    LiveCourseConfigVo liveCourseConfigVo = new LiveCourseConfigVo();
    LiveCourseConfig liveCourseConfig = liveCourseConfigService.getByLiveCourseId(id);
    if(null != liveCourseConfig) {
        List<LiveCourseGoods> liveCourseGoodsList = liveCourseGoodsService.findByLiveCourseId(id);
        BeanUtils.copyProperties(liveCourseConfig, liveCourseConfigVo);
        liveCourseConfigVo.setLiveCourseGoodsList(liveCourseGoodsList);
    }
    return liveCourseConfigVo;
}
```

(4) LiveCourseConfigService 添加方法

```
public interface LiveCourseConfigService extends IService<LiveCourseConfig> {
    //查看配置信息
    LiveCourseConfig getByLiveCourseId(Long id);
}
```

(5) LiveCourseConfigServiceImpl 实现方法

```

@Service
public class LiveCourseConfigServiceImpl extends ServiceImpl<LiveCourseConfigMapper, LiveCourseConfig> {

    //查看配置信息
    @Override
    public LiveCourseConfig getByLiveCourseId(Long liveCourseId) {
        return baseMapper.selectOne(new LambdaQueryWrapper<LiveCourseConfig>().eq(
            LiveCourseConfig::getLiveCourseId,
            liveCourseId));
    }
}

```

(6) LiveCourseGoodsService 添加方法

```

public interface LiveCourseGoodsService extends IService<LiveCourseGoods> {
    //获取课程商品列表
    List<LiveCourseGoods> findByLiveCourseId(Long id);
}

```

(7) LiveCourseGoodsServiceImpl 实现方法

```

@Service
public class LiveCourseGoodsServiceImpl extends ServiceImpl<LiveCourseGoodsMapper, LiveCourseGoods> {

    //获取课程商品列表
    @Override
    public List<LiveCourseGoods> findByLiveCourseId(Long liveCourseId) {
        return baseMapper.selectList(new LambdaQueryWrapper<LiveCourseGoods>()
            .eq(LiveCourseGoods::getLiveCourseId, liveCourseId));
    }
}

```

11.2、修改直播配置信息

(1) LiveCourseController 添加方法

```
@ApiOperation(value = "修改配置")
@PutMapping("updateConfig")
public Result updateConfig(@RequestBody LiveCourseConfigVo liveCourseConfigVo) {
    liveCourseService.updateConfig(liveCourseConfigVo);
    return Result.ok(null);
}
```

(2) LiveCourseService 添加方法

```
//修改配置
void updateConfig(LiveCourseConfigVo liveCourseConfigVo);
```

(3) LiveCourseServiceImpl 实现方法


```

@Override
public void updateConfig(LiveCourseConfigVo liveCourseConfigVo) {
    LiveCourseConfig liveCourseConfigUpt = new LiveCourseConfig();
    BeanUtils.copyProperties(liveCourseConfigVo, liveCourseConfigUpt);
    if(null == liveCourseConfigVo.getId()) {
        liveCourseConfigService.save(liveCourseConfigUpt);
    } else {
        liveCourseConfigService.updateById(liveCourseConfigUpt);
    }
    liveCourseGoodsService.remove(new LambdaQueryWrapper<LiveCourseGoods>().eq(LiveCourseGoods::goodsId, liveCourseConfigVo.getGoodsId()));
    if(!CollectionUtils.isEmpty(liveCourseConfigVo.getLiveCourseGoodsList())) {
        liveCourseGoodsService.saveBatch(liveCourseConfigVo.getLiveCourseGoodsList());
    }
    this.updateLifeConfig(liveCourseConfigVo);
}

/**
 * 上传直播配置
 * @param liveCourseConfigVo
 */
@SneakyThrows
private void updateLifeConfig(LiveCourseConfigVo liveCourseConfigVo) {
    LiveCourse liveCourse = this.getById(liveCourseConfigVo.getLiveCourseId());

    //参数设置
    HashMap<Object,Object> options = new HashMap<Object, Object>();
    //界面模式
    options.put("pageViewMode", liveCourseConfigVo.getPageViewMode());
    //观看人数开关
    JSONObject number = new JSONObject();
    number.put("enable", liveCourseConfigVo.getNumberEnable());
    options.put("number", number.toJSONString());
    //观看人数开关
    JSONObject store = new JSONObject();
    number.put("enable", liveCourseConfigVo.getStoreEnable());
    number.put("type", liveCourseConfigVo.getStoreType());
    options.put("store", number.toJSONString());
    //商城列表

```

```

List<LiveCourseGoods> liveCourseGoodsList = liveCourseConfigVo.getLiveCourseGoodsList();
if(!CollectionUtils.isEmpty(liveCourseGoodsList)) {
    List<LiveCourseGoodsView> liveCourseGoodsViewList = new ArrayList<>();
    for(LiveCourseGoods liveCourseGoods : liveCourseGoodsList) {
        LiveCourseGoodsView liveCourseGoodsView = new LiveCourseGoodsView();
        BeanUtils.copyProperties(liveCourseGoods, liveCourseGoodsView);
        liveCourseGoodsViewList.add(liveCourseGoodsView);
    }
    JSONObject goodsListEdit = new JSONObject();
    goodsListEdit.put("status", "0");
    options.put("goodsListEdit ", goodsListEdit.toJSONString());
    options.put("goodsList", JSON.toJSONString(liveCourseGoodsViewList));
}

String res = mtCloudClient.courseUpdateLifeConfig(liveCourse.getCourseId().toString(), options);

CommonResult<JSONObject> commonResult = JSON.parseObject(res, CommonResult.class);
if(Integer.parseInt(commonResult.getCode()) != MTCloud.CODE_SUCCESS) {
    throw new GgktException(20001, "修改配置信息失败");
}
}
}

```

11.3、获取最近直播课程

(1) LiveCourseController 添加方法

```

@ApiOperation(value = "获取最近的直播")
@GetMapping("findLatelyList")
public Result findLatelyList() {
    return Result.ok(liveCourseService.findLatelyList());
}

```

(2) LiveCourseService 添加方法

```

//获取最近的直播
List<LiveCourseVo> findLatelyList();

```

(3) LiveCourseServiceImpl 实现方法

```
@Override
public List<LiveCourseVo> findLatellyList() {
    List<LiveCourseVo> liveCourseVoList = baseMapper.findLatellyList();

    for(LiveCourseVo liveCourseVo : liveCourseVoList) {
        liveCourseVo.setStartTimeString(new DateTime(liveCourseVo.getStartTime()).toString("yyyy年"));
        liveCourseVo.setEndTimeString(new DateTime(liveCourseVo.getEndTime()).toString("HH:mm"));

        Long teacherId = liveCourseVo.getTeacherId();
        Teacher teacher = teacherFeignClient.getTeacherInfo(teacherId);
        liveCourseVo.setTeacher(teacher);

        liveCourseVo.setLiveStatus(this.getLiveStatus(liveCourseVo));
    }
    return liveCourseVoList;
}

/**
 * 直播状态 0: 未开始 1: 直播中 2: 直播结束
 * @param liveCourse
 * @return
 */
private int getLiveStatus(LiveCourse liveCourse) {
    // 直播状态 0: 未开始 1: 直播中 2: 直播结束
    int liveStatus = 0;
    Date curTime = new Date();
    if(DateUtil.dateCompare(curTime, liveCourse.getStartTime())) {
        liveStatus = 0;
    } else if(DateUtil.dateCompare(curTime, liveCourse.getEndTime())) {
        liveStatus = 1;
    } else {
        liveStatus = 2;
    }
    return liveStatus;
}
```

(4) LiveCourseMapper 添加方法

```
public interface LiveCourseMapper extends BaseMapper<LiveCourse> {  
    //获取最近直播  
    List<LiveCourseVo> findLatelyList();  
}
```

(5) LiveCourseMapper.xml 编写 sql 语句

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  
<mapper namespace="com.atguigu.ggkt.live.mapper.LiveCourseMapper">  
    <resultMap id="liveCourseMap" type="com.atguigu.ggkt.vo.live.LiveCourseVo" autoMapping="true">  
    </resultMap>  
  
    <!-- 用于select查询公用抽取的列 -->  
    <sql id="columns">  
id,course_id,course_name,start_time,end_time,teacher_id,cover,create_time,update_time,is_deleted  
    </sql>  
  
    <select id="findLatelyList" resultMap="liveCourseMap">  
        select <include refid="columns" />  
        from live_course  
        where date(start_time) >= curdate()  
        order by id asc  
        limit 5  
    </select>  
</mapper>
```