浙大城市学院

ZHEJIANG UNIVERSITY CITY COLLEGE

学生实习报告

课程编号： C01098

课程名称： 移动互联网应用开发实践

学 号：31901061

姓 名：张泽峰

专业班级: 计算机1902班

所在学院： 计算学院

报告日期： 2022 年 7 月 6 日

一、总览

（一）Springboot后端

桥接web服务端与androidstudio客户端：

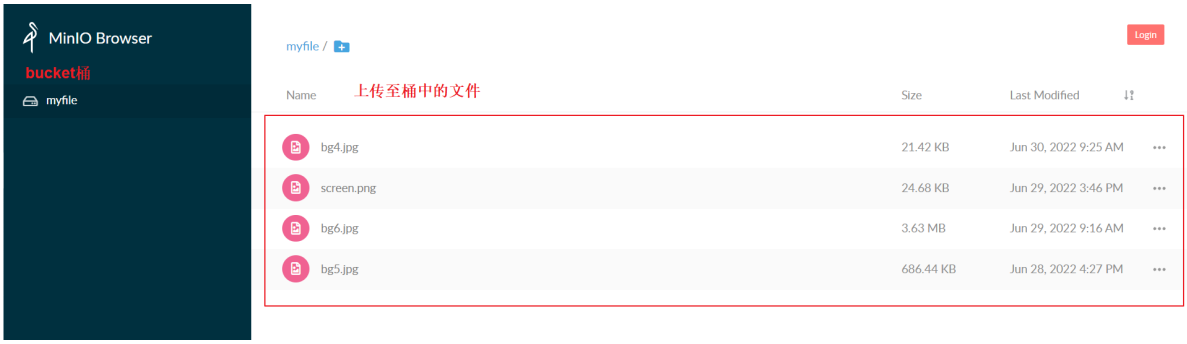
模块	接口
首页	1、获取首页详情（节目数量、素材总大小）
MinIO服务器	1、服务器搭建 2、创建存储桶 3、上传文件 4、下载文件
素材管理	1、获取所有素材 2、上传素材 4、下载素材 5、删除素材 6、修改素材名 7、按素材名模糊查找素材
节目管理	1、获取所有节目 2、创建节目 3、删除节目 4、按节目名模糊查找节目 5、按分辨率查找节目 6、按节目状态查找节目 7、修改节目名称
公告管理	1、发布公告 2、删除公告

（二）项目部署

阿里云服务器部署Springboot后端

二、实现

（一）MinIO服务器



MinIO配置：

```
spring.servlet.multipart.max-file-size=100MB
spring.servlet.multipart.max-request-size=100MB
minio.address=http://47.99.158.248:9000/
minio.accessKey=
minio.secretKey=
minio.bucketName=myfile
```

服务接口（以上传为例）：

检查文件桶是否存在，然后利用MultipartFile格式接收本地文件，将其转化成inputstream流并利用提供的presignedGetObject方法生成素材特有的访问url，然后其上传至minio服务器，同时操作素材服务层进行数据库的插入操作。

```
    /**
     * 上传文件并写入数据库
     *
     * @param file 上传文件
     * @return 成功则返回文件名，失败返回空
     */
    public void uploadFile(MinioClient minioClient, MultipartFile file,String
user_name) throws BoeBoardServiceException {

        //创建存储桶
        boolean createFlag = makeBucket(minioClient, bucketName);
        //创建存储桶失败
        if (createFlag == false) {
            throw new BoeBoardServiceException("创建失败");
        }
        try {
            PutObjectOptions putObjectOptions = new
PutObjectOptions(file.getSize(), PutObjectOptions.MIN_MULTIPART_SIZE);
            putObjectOptions.setContentType(file.getContentType());
            String originalFilename = file.getOriginalFilename();
            int pointIndex = originalFilename.lastIndexOf(".");
            //得到文件流
            InputStream inputStream = file.getInputStream();
            //保证文件不重名(并且没有特殊字符)
            String fileName = file.getOriginalFilename();
            minioClient.putObject(bucketName, fileName, inputStream,
putObjectOptions);
            //调用MaterialService将数据插入数据库
            //生成url，有效期3天
            String url = minioClient.presignedGetObject("myfile", fileName, 60 *
60 * 72);
            materialService.upload(fileName,url,file.getSize(),user_name);
        } catch (Exception e) {
            e.printStackTrace();
            throw new BoeBoardServiceException("上传失败");
        }
    }
```

（二）首页

获取信息上传至web端

Controller层

```
/**
 * 获取信息上传web端
 *        {
 *                programme_num;          //节目数量
 *                material_size;          //素材总大小
 *                scheme_num;             //计划数量
 *                device_num;             //设备数量
 *                cnt1;                   //离线状态的设备数量
 *                cnt2;                   //播放状态的设备数量
 *                cnt3;                   //空闲状态的设备数量
 *                inst_num;               //机构数量
 *                groupings_num;          //分组数量
 *        }
 * @return
 */
@ResponseBody
@RequestMapping(value = "/getinfo", method = RequestMethod.GET)
public ResponseData GetHomeInfo() {
    HomeInformDto dto = homeService.GetHomeInfo();
    return new ResponseData(ExceptionMsg.SUCCESS, dto);
}
```

Service层

```
@Override
public HomeInformDto GetHomeInfo() throws BoeBoardServiceException {
    HomeInformDto res = new HomeInformDto();//存放结果

    //获取节目数量
    int programme_num = programmeRepository.CountProgramme();

    //获取素材总大小
    List<String> sizelist = materialRepository.SelectMsize();
    sizelist.forEach(size -> {
        double single_size = 0;
        int index = 0;
        while (size.charAt(index) != 'M' && size.charAt(index) != 'K') {
            index++;
        }
        double num = Double.valueOf(size.substring(0, index));
        if (size.charAt(index) == 'M') {
            single_size = num * 1024 * 1024;
        } else if (size.charAt(index) == 'K') {
            single_size = num * 1024;
        }
        total_size += single_size;
    });
    MaterialServiceImpl materialService = new MaterialServiceImpl();
    String totalsize = materialService.setSize((long) total_size);
```

封装的Dto类

```java
@Data
public class HomeInformDto implements Serializable {
    private int programme_num;//节目数量
    private String material_size;//素材总大小
    private int scheme_num;//计划数量
    private int device_num;//设备数量
    private int cnt1;//离线状态的设备数量
    private int cnt2;//播放状态的设备数量
    private int cnt3;//空闲状态的设备数量
    private Map<String, Integer> inst_num;//机构数量
    private Map<String, Integer> groupings_num;//分组数量
}
```

（三）素材管理（展示部分）

1、上传图片

Controller层

```java
/**
 * 上传图片
 * @param file
 * @return
 */
@ResponseBody
@RequestMapping(value = "/uploadfile", method = RequestMethod.POST)
public ResponseData uploadFile(@RequestBody MultipartFile file,@RequestParam String user_name){
    MinioClient minioClient = minIOService.getMinioClient();
    if (minioClient == null) {
        return new ResponseData(ExceptionMsg.FAILED, data: "上传失败，无法连接MinIo服务器");
    }
    try{
        minIOService.uploadFile(minioClient, file,user_name);
    }catch (BoeBoardServiceException e){
        return new ResponseData(ExceptionMsg.FAILED,e);
    }

    return new ResponseData(ExceptionMsg.SUCCESS, data: "上传成功");
}
```

```java
/**
 * 上传文件并写入数据库
 * @param file 上传文件
 * @return 成功则返回文件名，失败返回空
 */
public void uploadFile(MinioClient minioClient, MultipartFile file,String
user_name) throws BoeBoardServiceException {
    //创建存储桶
    boolean createFlag = makeBucket(minioClient, bucketName);
    //创建存储桶失败
    if (createFlag == false) {
        throw new BoeBoardServiceException("创建失败");
    }
    try {
        PutObjectOptions putObjectOptions = new PutObjectOptions(file.getSize(),
PutObjectOptions.MIN_MULTIPART_SIZE);
```

```
        putObjectOptions.setContentType(file.getContentType());
        String originalFilename = file.getOriginalFilename();
        int pointIndex = originalFilename.lastIndexOf(".");
        //得到文件流
        InputStream inputStream = file.getInputStream();
        //保证文件不重名(并且没有特殊字符)
        String fileName = file.getOriginalFilename();
        minioClient.putObject(bucketName, fileName, inputStream,
putObjectOptions);
        //调用MaterialService将数据插入数据库
        //生成url，有效期3天
        String url = minioClient.presignedGetObject("myfile", fileName, 60 * 60
* 72);
        materialService.upload(fileName,url,file.getSize(),user_name);
    } catch (Exception e) {
        e.printStackTrace();
        throw new BoeBoardServiceException("上传失败");
    }
}
```

支持上传图片、视频、音乐文件

m_id	mname	murl	category	dpi	msize	author	updatetime	createtime	mstatus
1	bg.jpg	http://localhost:9000/myfile/l	1	(Null)	106.38KB	(Null)	2022-06-27 11:20:11	2022-06-27 11:20:11	已删除
2	bg1.png	http://localhost:9000/myfile/l	1	(Null)	18.41MB	(Null)	(Null)	2022-06-27 11:20:34	未删除
3	bg6.jpg	http://localhost:9000/myfile/l	1	(Null)	3.63MB	(Null)	2022-06-28 09:59:00	2022-06-28 09:59:00	未删除
4	bg5.jpg	http://47.99.158.248:9000/my	1	(Null)	686.44KB	(Null)	2022-06-28 16:27:04	2022-06-28 16:27:04	未删除
5	bg6.jpg	http://47.99.158.248:9000/my	1	(Null)	3.63MB	(Null)	2022-06-29 09:16:34	2022-06-29 09:16:34	未删除
6	bg4.jpg	http://47.99.158.248:9000/my	1	(Null)	21.42KB	zzf	2022-06-30 09:25:15	2022-06-30 09:25:15	未删除
7	2022-05-07 20-17-24.mkv	http://47.99.158.248:9000/my	1	(Null)	44.98MB	zzf	2022-06-30 13:35:59	2022-06-30 13:35:59	未删除
8	King Gnu (キング・ヌー) - 飛行艇.mp3	http://47.99.158.248:9000/my	1	(Null)	4.17MB	zzf	2022-06-30 15:36:31	2022-06-30 15:36:31	未删除

2、获取所有素材

将素材信息包装成一个dto，返回一个dtolist

```
/**
     * 返回图片list的JSON字符串
     * list格式如下
     * {
     *     "rspCode": "200",
     *     "rspMsg": "操作成功",
     *     "data": [
     *         {
     *             "mname": "bg.jpg",
     *             "murl": "http://localhost:9000/myfile/bg.jpg",
     *             "msize": "106.38KB    ",
     *             "author": "张三",
     *             "updatetime": "2022-06-27 11:20:11",
     *             "createtime": "2022-06-27 11:20:11",
     *             "mstatus": "已删除"
     *         },
     *         {
     *             "mname": "bg1.png",
     *             "murl": "http://localhost:9000/myfile/bg1.png",
     *             "msize": "18.41MB    ",
     *             "author": "李四",
     *             "updatetime": null,
     *             "createtime": "2022-06-27 11:20:34",
     *             "mstatus": "未删除"
```

```
     *          }
     */
    @ResponseBody
    @RequestMapping(value = "/getall", method = RequestMethod.GET)
    public ResponseData GetAll(){
        List<BackToWebDto> result =  materialService.getall_photos();

        return new ResponseData(ExceptionMsg.SUCCESS,result);
    }
```

Service层包装Dto

```
public List<BackToWebDto> GetMaterialList(List<Materials> materialslist){
    List<BackToWebDto> dtolist = new ArrayList<>();

    materialslist.forEach(material ->{
        BackToWebDto dto = new BackToWebDto();
        dto.setMname(material.getMname());
        dto.setMurl(material.getMurl());
        dto.setMsize(material.getMsize());
        dto.setAuthor(material.getAuthor());
        dto.setUpdatetime(material.getUpdatetime());
        dto.setCreatetime(material.getCreatetime());
        dto.setMstatus(material.getMstatus());

        dtolist.add(dto);
    } );

    return dtolist;
}
```

文件大小以String类型保存

```
//文件大小转化
public String setSize(long size) {
    int GB = 1024 * 1024 * 1024;//定义GB的计算常量
    int MB = 1024 * 1024;//定义MB的计算常量
    int KB = 1024;//定义KB的计算常量
    DecimalFormat df = new DecimalFormat( pattern: "0.00");//格式化小数
    String resultSize = "";
    if (size / GB >= 1) {
        //如果当前Byte的值大于等于1GB
        resultSize = df.format( number: size / (float) GB) + "GB";
    } else if (size / MB >= 1) {
        //如果当前Byte的值大于等于1MB
        resultSize = df.format( number: size / (float) MB) + "MB";
    } else if (size / KB >= 1) {
        //如果当前Byte的值大于等于1KB
        resultSize = df.format( number: size / (float) KB) + "KB";
    } else {
        resultSize = size + "B";
    }
    return resultSize;
}
```

3、按名称模糊查找

Controller层

```java
/**
 * 按名称模糊查找所有图片文件
 * @param map
 *              {
 *                  "search_name":"zzza"
 *              }
 * @return
 */
@ResponseBody
@RequestMapping(value = "/select", method = RequestMethod.GET)
public ResponseData SearchByName(@RequestBody Map map){
    String search_name = (String)map.get("search_name");
    List<BackToWebDto> result = new ArrayList<>();
    try {
        result = materialService.SelectByName(search_name);
    }catch (BoeBoardServiceException e){
        return new ResponseData(ExceptionMsg.FAILED,e);
    }
    return new ResponseData(ExceptionMsg.SUCCESS,result);
}
```

Service层

```java
/**
 * 按名称模糊查找所有图片
 * @param name     模糊查找内容
 * @return
 */
@Override
public List<BackToWebDto> SelectByName(String name) throws BoeBoardServiceException {

    List<Materials> materials_list = new ArrayList<~>();
    if(materials_list.size() == 0){
        throw new BoeBoardServiceException("查找失败，不存在对应素材");
    }
    materials_list = materialRepository.SelectByName(name);
    List<BackToWebDto> dtolist = this.GetMaterialList(materials_list);

    return dtolist;
}
```

（四）节目管理（展示部分）

1、新建节目

以dto打包接收web端新建的节目json数据，在service层处理数据

```
/**
     *
     * @param programmeDto      //前端请求的数据包，包括
     *                          {
     *                              "user_id":1                     //用户id
     *                              "p_name": "zzz",                //节目名称
     *                              "dpi": "800*600",               //节目分辨率
     *                              "murl": [                       //素材url
     *                                  "http://usqn.lk/rspisf",
     *                                  "http://drwboxhlof.bv/uuvcncf",
     *                                  "http://xuivz.mq/bede"
     *                              ],
     *                              "plength": 5,                   //节目时长(轮
播时间)
```

```
     *                                "psize": "3Mb"                    //节目大小(素材总大小)
     *                            }
     * @return
     */
    @ResponseBody
    @RequestMapping(value = "/add", method = RequestMethod.POST)
    public ResponseData addProgramme(@RequestBody ProgrammeDto programmeDto) {
        try {
            long user_id = programmeDto.getUser_id();
            programmeService.AddProgramme(user_id,programmeDto);
            return new ResponseData(ExceptionMsg.SUCCESS,"添加成功");
        }catch (BoeBoardServiceException e){
            System.out.println(e);
        }
        return new ResponseData(ExceptionMsg.SUCCESS,"添加成功");
    }
```

将解析后的数据插入数据库当中

```
//新增节目
@Override
public void AddProgramme(long user_id, ProgrammeDto programmeDto) throws BoeBoardServiceException {
    Users users = usersRepository.findById(user_id);

    if(users == null){
        throw new BoeBoardServiceException("添加失败，用户不存在");
    }
    String user_name = users.getUsername();
    int p_id = programmeRepository.findMaxId()+1;
    List<String> urls = programmeDto.getMurl();     //节目使用的图片list
    String p_name = programmeDto.getP_name();
    if(programmeRepository.findByName(p_name) != 0){
        throw new BoeBoardServiceException("节目名已存在，勿重复添加");
    }

    Programme programme = new Programme();
    programme.setId(p_id);
    programme.setPname(programmeDto.getP_name());
    programme.setDpi(programmeDto.getDpi());
    programme.setPlength(programmeDto.getPlength());
    programme.setPsize(programmeD
    programme.setPstatus("未使用")
    programme.setAuthor(user_name
    programme.setCreatetime(new Date());
    programme.setUpdatetime(new Date());
    programme.setUrls(JSON.toJSONString(urls));
    programmeRepository.save(programme);

}
```

ProgrammeDto programmeDto
boeboard

2、按条件(名称、dpi、状态)查找对应的节目

```java
/**
 * 按input节目名模糊查找节目
 * @param map
 * @return
 */
@ResponseBody
@RequestMapping(value = "/search/byname",method = RequestMethod.GET)
public ResponseData SearchByName(@RequestBody Map map){

    String search_content = (String)map.get("search_content");
    List<GetProgrammesDto> result = programmeService.SearchByName(search_content);

    return new ResponseData(ExceptionMsg.SUCCESS,result);
}
```

```java
/**
 * 按dpi查找节目
 * @param map
 * @return
 */
@ResponseBody
@RequestMapping(value = "/search/bydpi",method = RequestMethod.GET)
public ResponseData SearchByDpi(@RequestBody Map map){

    String search_content = (String)map.get("search_content");
    List<GetProgrammesDto> result = programmeService.SearchByDpi(search_content);

    return new ResponseData(ExceptionMsg.SUCCESS,result);
}
```

```java
/**
 * 按状态查找节目
 * @param map
 * @return
 */
@ResponseBody
@RequestMapping(value = "/search/bystatus",method = RequestMethod.GET)
public ResponseData SearchByStatus(@RequestBody Map map){

    String search_content = (String)map.get("search_content");
    List<GetProgrammesDto> result = programmeService.SearchByStatus(search_content);

    return new ResponseData(ExceptionMsg.SUCCESS,result);
}
```

GetProgramme方法统一处理因查询条件不同获得的节目对象list，返回一个查询到的dtolist

```java
    public List<GetProgrammesDto> GetProgramme(List<Programme> list){
        List<GetProgrammesDto> dtolist = new ArrayList<>();

        list.forEach(programme -> {
            GetProgrammesDto dto = new GetProgrammesDto();

            dto.setP_id(programme.getId());
            dto.setPname(programme.getPname());
            dto.setDpi(programme.getDpi());
            dto.setPlength(programme.getPlength());
```

```
            dto.setPsize(programme.getPsize());
            dto.setPstatus(programme.getPstatus());
            dto.setAuthor(programme.getAuthor());
            dto.setCreatetime(programme.getCreatetime());
            dto.setUpdatetime(programme.getUpdatetime());
            dto.setUrls(programme.getUrls());

            dtolist.add(dto);
        });

        return dtolist;
    }
```

（五）公告管理

1、新增公告

以dto格式接受web前端设置的公告内容格式

```
/**
     * 新增一个公告
     * web端返回的数据结构
     *      {
     *          "user_id":1,                //用户id
     *          "content":"Helloword",      //公告内容
     *          "text_color":"#FFC0CB",     //字体颜色
     *          "text_size":12,             //字体大小
     *          "background_color":"#000000"//背景颜色
     *          "start_time":"2022-06-09 20:13:24"  //开始时间
     *          "finish_time":"2022-06-10 20:13:24" //结束时间
     *      }
     * @param dto
     */
    @ResponseBody
    @RequestMapping(value = "/add",method = RequestMethod.POST)
    public ResponseData AddAnnounce(@RequestBody GetAnnounceDto dto){

        announceService.AddAnnounce(dto);

        return new ResponseData(ExceptionMsg.SUCCESS,"添加成功");
    }
```

service层有两个任务，第一是将前端给的公告信息存入数据库当中，第二是通过消息队列，将公告的信息打包成固定主题的消息发送出去，androidstudio端拦截到消息后显示公告。

```java
//增加公告
@Override
public void AddAnnounce(GetAnnounceDto dto) {
    //保存到数据库
    Announce announce = new Announce();
    int a_id = announceRepository.findMaxId()+1;
    announce.setId(a_id);
    announce.setContent(dto.getContent());
    announce.setText_color(dto.getText_color());
    announce.setText_size(dto.getText_size());
    announce.setBackground_color(dto.getBackground_color());
    announce.setAstatus("公告中");
    announce.setAuthor(usersRepository.getById(dto.getUser_id()).getUsername());
    announce.setCreatetime(new Date());

    announceRepository.save(announce);

    //通过消息队列发送消息给AndroidStudio端
    MqMessage msg = new MqMessage(MqMessage.CATEGORY_ANNOUNCE);
    msg.appendContent("content",dto.getContent());
    msg.appendContent("text_color",dto.getText_color());
    msg.appendContent("text_size",dto.getText_size());
    msg.appendContent("background_color",dto.getBackground_color());
    msg.appendContent("start_time",dto.getStart_time());
    msg.appendContent("finish_time",dto.getFinish_time());
    mqService.convertAndSend(Constants.QUE_ANNOUNCE, msg.stringfy());

}
```

①

②

2、删除公告(软删除)

Controller层

```java
//删除公告
@Override
public void DeleteAnnounce(String content) { announceRepository.deleteByName(content); }
```

Service层

```java
/**
 * 删除公告
 * @param map
 * @return
 */
@ResponseBody
@RequestMapping(value = "/delete",method = RequestMethod.POST)
public ResponseData DeleteAnnoune(@RequestBody Map map){
    String content = (String)map.get("content");

    announceService.DeleteAnnounce(content);

    return new ResponseData(ExceptionMsg.SUCCESS, data: "删除成功");
```

（六）阿里云服务器部署后端

1、下载JDK

本地Windows系统中下载 Linux系统下的jdk，jdk下载官网：https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html。

2、JDK包放入 服务器 /usr/java 目录下

3、解压JDk压缩包

```
//解压指令
tar -zxvf jdk-8u221-linux-x64.tar.gz
```

4、设置系统环境变量。linux系统中的设置方法为：找到文件 /etc/profile ， 向其中添加如下代码：

```
//jdk1.8.0_221  jdk版本
export JAVA_HOME=/usr/java/jdk1.8.0_221
export CLASSPATH=$JAVA_HOME/lib/
export PATH=$PATH:$JAVA_HOME/bin
export PATH JAVA_HOME CLASSPATH
```

5、Springboot项目添加设置 打开pom.xml，添加如下语句，将项目的打包形式设置好

```
<!-- 打包成jar包 -->
<packaging>jar</packaging>
```

6、运行maven构建得到jar包，上传jar包到服务器根目录

7、项目运行

```
//关闭就停止
java -jar "*****.jar";
//一直运行
nohup java -jar  ***.jar  &     //***为你的jar包名
```

```
//一直运行
nohup java -jar xiaomi-1.0-SNAPSHOT.jar &> nohup.txt

//关闭项目
ps -ef |grep 项目jar包名 得到进程号
kill -9 进程号 关闭
```

```
nohup java -jar xiaomi-1.0-SNAPSHOT.jar &> nohup.txt
```

`xiaomi-1.0-SNAPSHOT.jar`是你的项目jar包
这是我在 云服务器 上后台启用spring-boot程序的时候用的命令
关闭项目：
ps -ef | grep 项目jar包名 得到进程号
kill -9 进程号 关闭

三、接口文档

短学期接口设计细节(个人部分)

四、个人总结

由于本学期主要课程就是springboot、vue3，所以这次短学期的任务对我来说有一个比较明确的实现方向。

在小组任务分工中，我分到后端Springboot的实现，主要负责首页、素材管理、节目管理的后端实现。主体思路由两部分，一部分是用restful接口和web端实现通信，即GET或POST数据，实现前端页面的渲染和数据库的更新。另一部分是用消息队列接发主题消息，实现与androidstudio的消息互通，实现移动端屏幕的数据渲染以及对屏硬件的一系列操作。

在写接口的时候遇到的第一个问题是文件的上传下载。整个项目的基础就是素材，由素材搭建节目以及形成计划。因此，后端需要实现素材的上传、下载。由于文件的传输比较复杂，所以我最后选择使用MinIO服务器，用它提供的接口实现文件的传输工作。在前端选择上传素材时，访问我复写的MinIO上传接口，生成一个有时效的url，通过访问url，前端可以获取到素材信息。同时，将素材名称、大小、url链接、上传的用户、更新时间、上传时间等字段存入数据库方便后续操作。

解决文件的传输问题后，其他接口的实现就显得比较简单了，如以节目管理为例，web前端选择新建节目时，选择一系列节目的创建条件，前端将这些条件封装成JSON请求接口，springboot中将请求的数据解析存入数据库。其中有的接口实现了web前端和移动端的双向交互。如在公告管理中，web端新建一个公告，springboot解析公告的详情后生成指定主题的消息通过消息队列发布。在androidstudio端接受发布的消息后将公告消息解析并生成屏上公告。

总的来说，这次短学期进一步加深了我对后端Springboot的运用与理解，这也是我第一次参与实现一个完整的前后端分离式设计系统，我从中学习到了很多前后端如何搭建的经验，同时也认识使用了诸如MinIO等文件处理工具。