

瑞华统一日志管理系统使用手册

浙江瑞华康源科技有限公司

修订历史

版本	日期	状态	修订人	摘要
V1.0	2022-11-29	C	魏小波	初稿

状态标识： C –Created A- Added M - Modified D - Deleted

1. 背景和目标

当前瑞华各产品日志记录缺乏统一的规范，日志有些是通过 rabbitmq 集中归集（耗材），有些是直接通过 log4j 打印在本地磁盘，不便于问题排查分析。基于此，考虑基于当前耗材日志系统设计的基础上，打造统一的日志中心，实现两个目标：

- 日志的统一归集，满足等保测评审计需求（用户的操作行为日志）
- 可支持按设备终端，业务订单号以及登录 token 进行操作日志的串联分析。

系统用户：医院护士、运维开发

系统价值

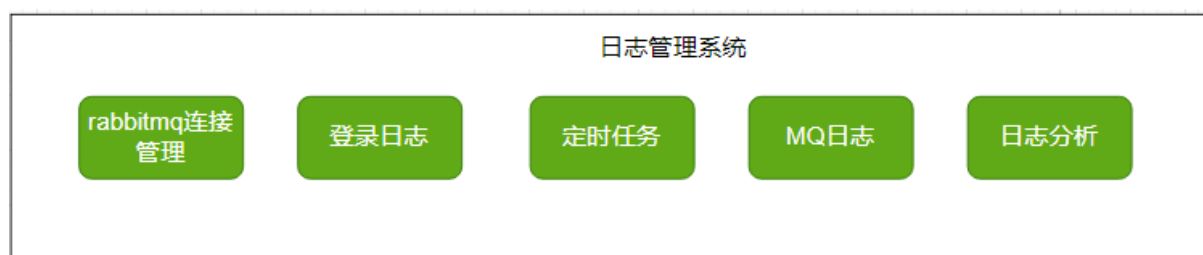
医院：人员行为跟踪，基于日志的数据分析

瑞华：

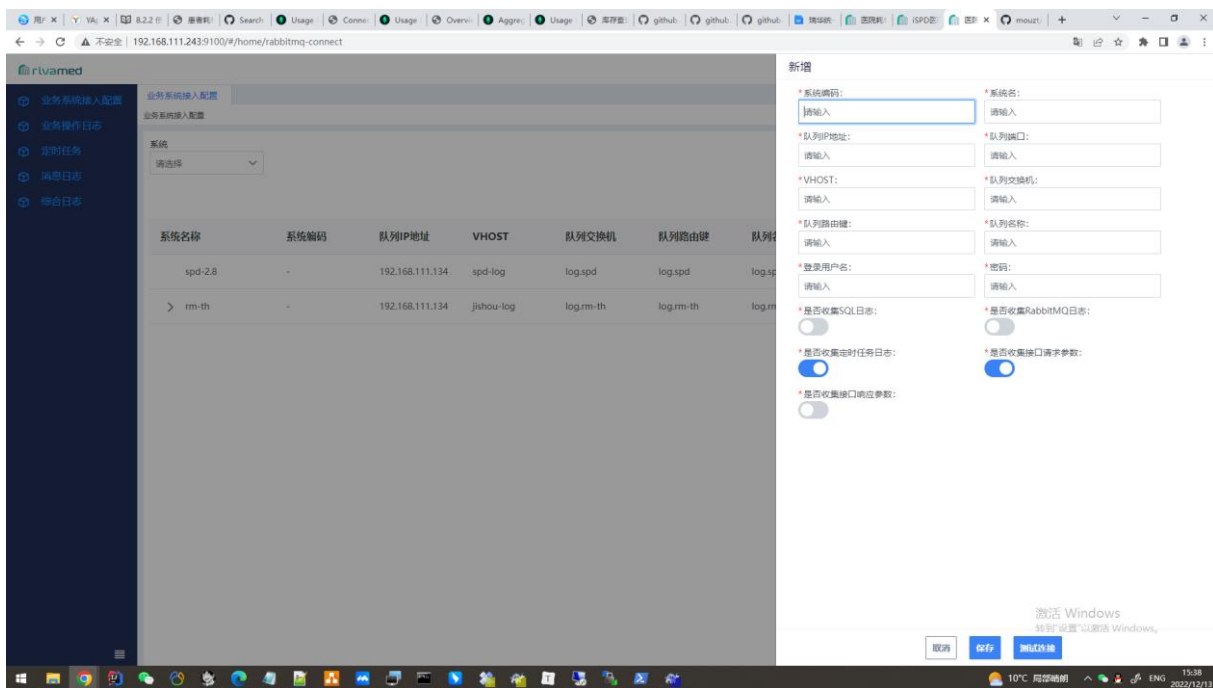
运维、实施能够通过系统排查问题，导出日志给开发，便于解决现场问题，尤其是类似阜外这种项目（日志、数据库都不能导出）；

统一日志管理，便于检索，链路跟踪

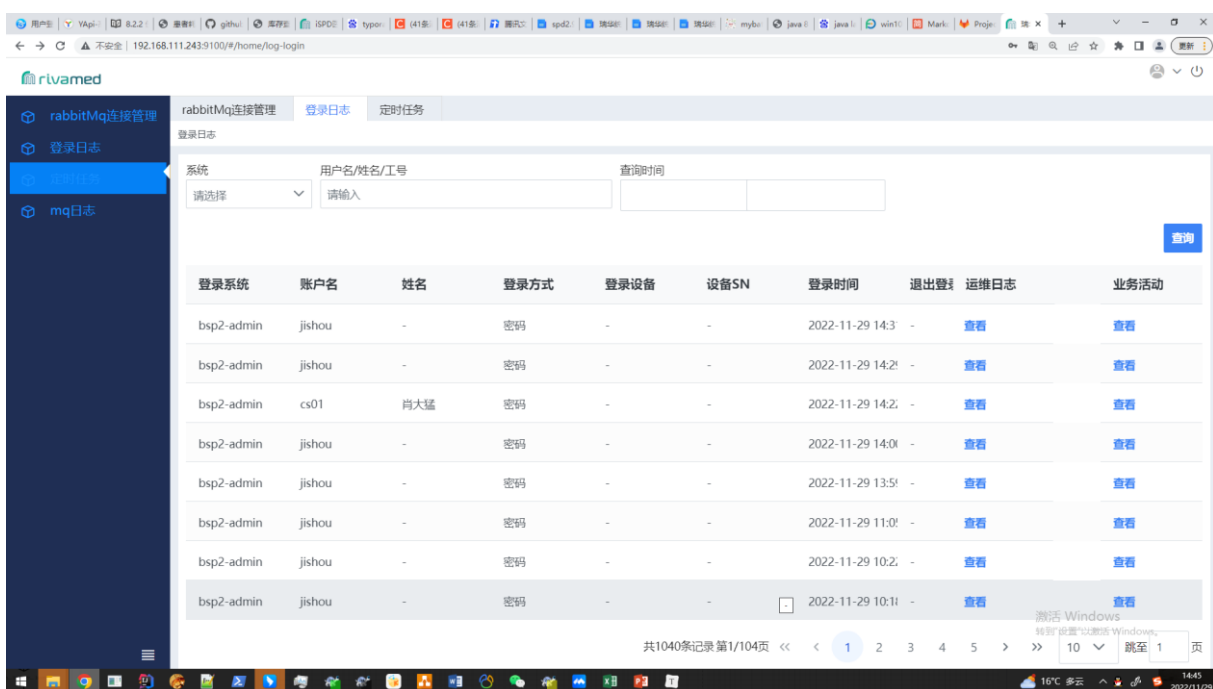
2. 系统功能架构



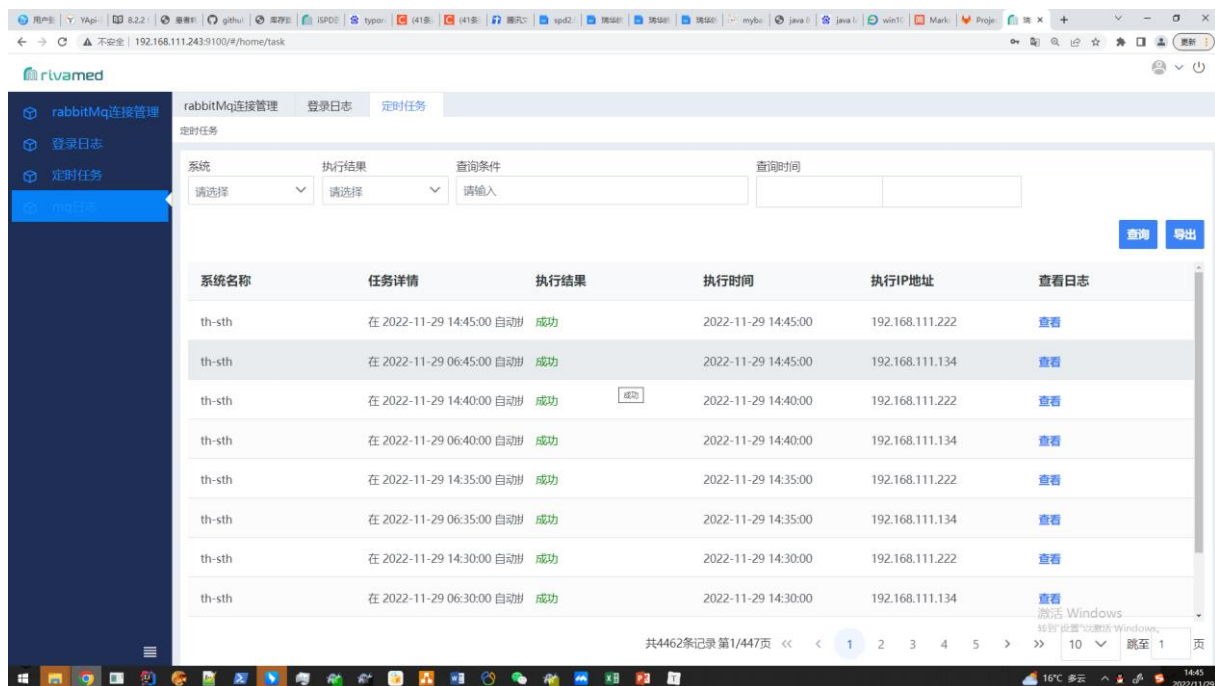
业务系统接入配置：定义有哪些系统需要监控，每个系统对应的日志队列配置



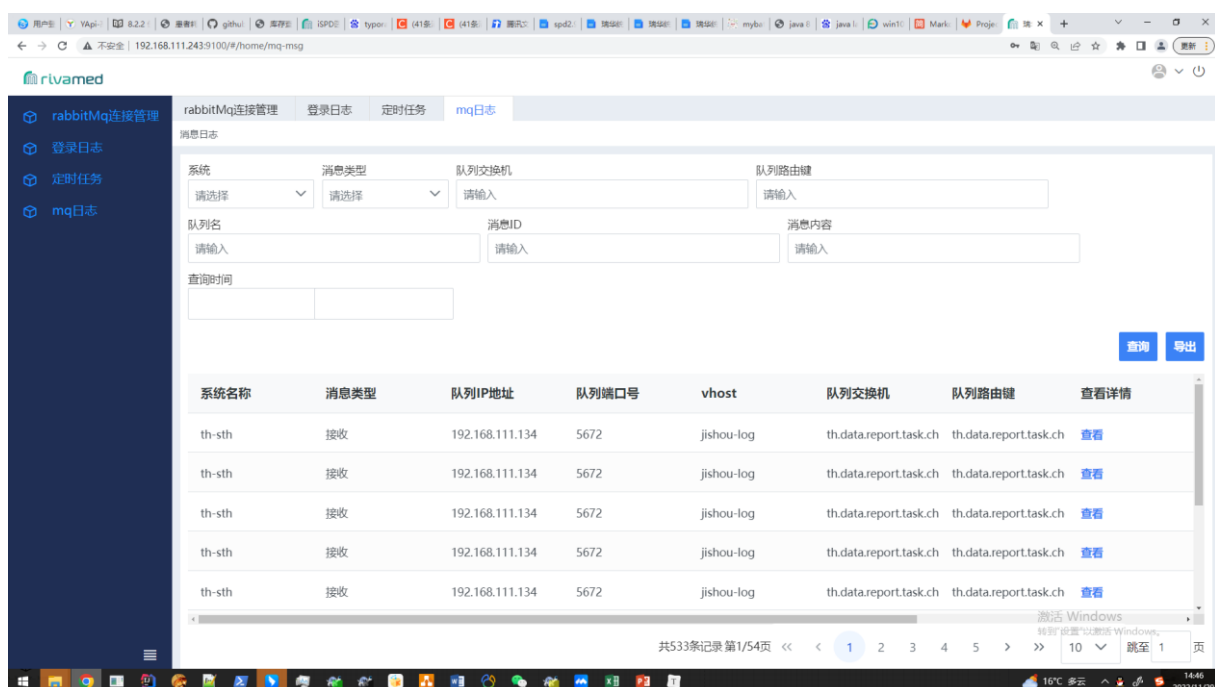
登录日志：业务系统登录记录，从这里可以查看用户登录系统后做了什么操作，系统可以记录语义化日志和运维日志，支持导出



定时任务日志：记录系统所执行的定时任务，是否执行成功，以及详细日志信息，支持导出



MQ 日志：业务系统和业务系统之间以及业务系统和智能硬件之间，需要监控收发情况，排查消息丢失以及业务链路中断的环节



日志分类：

按照业务划分：

操作日志：接口调用产生的接口日志，如登录、业务操作接口调用

定时任务日志：系统执行定时任务产生的日志，如报表生成等

mq 日志：消息推送产生的日志，如智能设备通过 IOT 下发指令；

消息接收产生的日志，如智能设备上传数据

运行日志：sql 日志、用户自己打印的日志、运行时异常

3. 系统非功能性考虑

3.1. 非功能性考虑

3.1.1. 性能支持：

- 系统设计的时候考虑高并发情况下日志降级（用户可配置）
- 日志中心支持多实例部署，增加队列的消费速度和高可用保证
- 支持每个业务服务有自己的日志队列，避免所有系统用同一队列，压力过大导致 rabbitmq 宕机
- 日志按月划分集合，避免每次查询的日志量过大

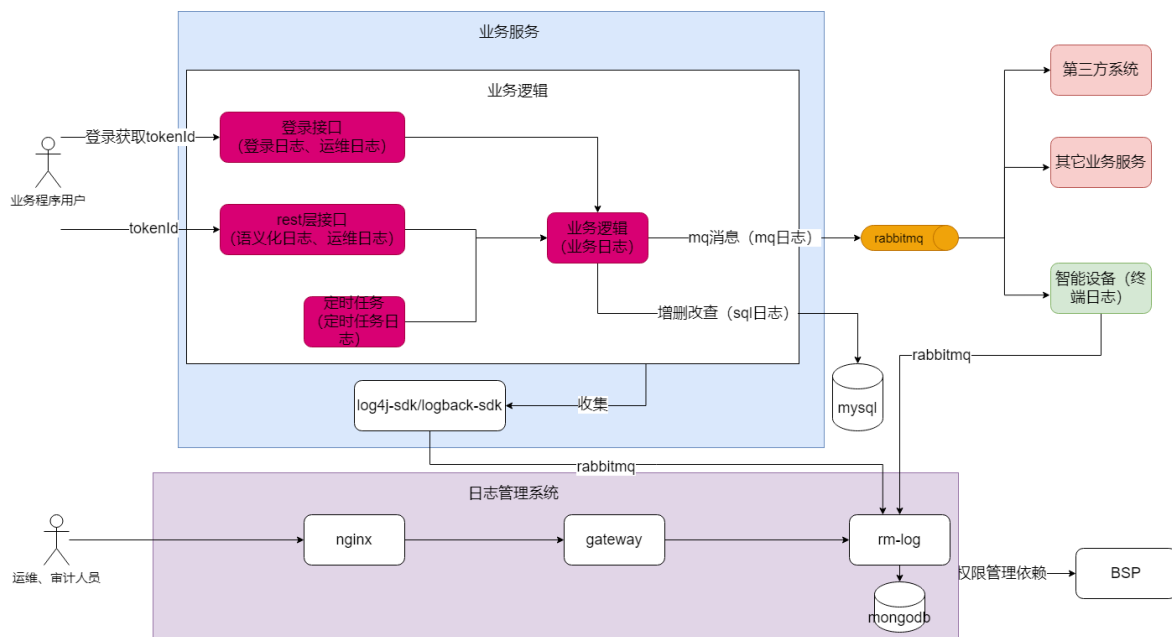
3.1.2. 可扩展性：

日志管理系统作为一个独立的管理系统，需要兼容考虑非 spring-cloud 体系、非 java 语言项目，这些项目只需要按照标准格式上传日志即可完成日志管理（.net 项目、智能终端按照标准格式组织日志内容，通过 IOT 平台或者直接 rabbitmq 推送）

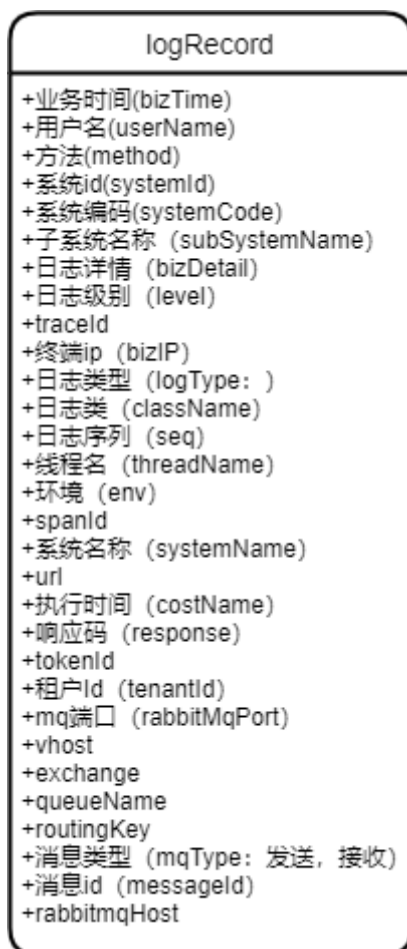
3.1.3. 数据隔离：

日志按照不同产品存储（spd、寄售、手术室、OA），数据隔离。后期可以支持按照租隔离

4. 系统架构



5. 日志 ER 图



数据示例：

```
{
  "_id": ObjectId("636e07b003a67e493f5f72c4"),
  "bizTime": ISODate("2022-11-11T08:28:28.72Z"),
  "userId": "11111",
  "userName": "test",
  "url": "/bsp/rmApi/login/bsplogin",
  "method": "BspLoginRest.bsploginRest",
  "subSysName": "bsp2-admin",
  "systemId": "123",
  "systemCode": "4.0",
  "sysName": "bsp2-admin",
  "bizDetail": "[{\"isUpdatePart\":\"1\",\"sourceFlag\":\"0\",\"systemType\":\"jishou-oms\",\"loginName\":\"jishou\",\"password\":\"Jishou123456\",\"loginMode\":\"1\",\"onlyToken\":\"0\",\"continueToLogin\":\"1\"}]",
  "costTime": NumberLong("360"),
  "responseCode": "200",
  "level": "INFO",
  "traceId": "63edb31f6f8186dd",
  "tokenId": "b0854ad6-da43-425f-807a-0e8b5c49170f",
  "tenantId": "11111",
  "bizIP": "192.168.111.222",
  "logType": "1",
  "className": "cn.rivamed.bsp.web.rest.login.BspLoginRest",
  "seq": NumberLong("5"),
  "threadName": "http-nio-8118-exec-11",
  "env": "dev",
  "spanId": "63edb31f6f8186dd",
  "_class": "cn.rivamed.log.entity.LogAllMessage"
}
```

6. 系统部署（单机）

服务器硬件要求：

待测试：

服务端配置

系统必须中间件 mongodb、nginx

必须基础服务平台 bsp（权限、账户管理）

运行环境 jdk1.8

服务器上创建文件目录：rm-log,上传 startup.sh 启动脚本和日志服务程序

/data/rm-log/rm-log/					
名字	大小	已改变	权限	拥有者	
..		2022/11/21 17:07:31	rwxr-xr-x	root	
log-service-exec.war	74,083 KB	2022/11/23 13:32:46	rw-r--r--	root	
nohup.out	1 KB	2022/11/29 15:11:29	rw-r--r--	root	
startup.sh	1 KB	2022/11/29 15:08:31	rwxr-xr-x	root	

修改端口配置和 mongodb 配置

```
# Kill the orbm process
pid=$(ps -ef | grep log-service-exec.war | grep server.port=8038 | awk '{print $2}')
```

```
if [ -n "$pid" ]
then
kill -9 $pid
fi

# delete log files
LOG_PATH="logs/"
rm -rf $LOG_PATH

# directory
www_path=/data/rm-log/rm-log
# Jenkins profile path
source /etc/profile
# cd to the directory
cd $www_path
cat /dev/null > nohup.out
# start up orbm
LOG_PATH="log-service-exec.war"
nohup java -jar -Xms512m -Xmx512m log-service-exec.war --server.port=8038 --spring.data.mongodb.uri=mongodb://rivamed:rivamed@192.168.111.134:27017/test_zuoyang?authSource=admin --file.encoding=UTF-8 --Dcas.standalone.config="$CONFIG" > nohup.out | sed '/^NM running for/ q'
```

执行./startup.sh， 启动系统

前端部署：

上传前端包， 解压

/data/rm-log/log-server-web/webapp/					
名字	大小	已改变	权限	拥有者	
..		2022/11/21 17:51:00	rwxr-xr-x	root	
assets		2022/11/21 18:02:45	rwxr-xr-x	root	
3rdpartylicenses.txt	23 KB	2022/11/21 17:56:16	rw-r--r--	root	
277-es5.b2d5fd3cce39f5bda1d1.js	3 KB	2022/11/21 17:56:16	rw-r--r--	root	
277-es2017.b2d5fd3cce39f5bda1d1.js	2 KB	2022/11/21 17:56:16	rw-r--r--	root	
379-es5.5d6282baac39e1d3ec0d.js	18 KB	2022/11/21 17:56:17	rw-r--r--	root	
379-es2017.5d6282baac39e1d3ec0d.js	17 KB	2022/11/21 17:56:17	rw-r--r--	root	
418-es5.8ff833d09b1f241fac27.js	15 KB	2022/11/21 17:56:17	rw-r--r--	root	
418-es2017.8ff833d09b1f241fac27.js	15 KB	2022/11/21 17:56:17	rw-r--r--	root	
451-es5.f9a3744676ab7c5d96af.js	449 KB	2022/11/21 17:56:17	rw-r--r--	root	
451-es2017.f9a3744676ab7c5d96af.js	418 KB	2022/11/21 17:56:18	rw-r--r--	root	
798-es5.3a642317203121bd91aa.js	13 KB	2022/11/21 17:56:18	rw-r--r--	root	
798-es2017.3a642317203121bd91aa.js	11 KB	2022/11/21 17:56:17	rw-r--r--	root	
912-es5.981d83c6c74f6ba774d7.js	162 KB	2022/11/21 17:56:18	rw-r--r--	root	
912-es2017.981d83c6c74f6ba774d7.js	156 KB	2022/11/21 17:53:44	rw-r--r--	root	
color.6441e63a57ccc5105bad.png	11 KB	2022/11/21 17:53:41	rw-r--r--	root	
config.7530bd497bb5f97023a9.js	1 KB	2022/11/25 16:41:43	rw-r--r--	root	
declare.d.ts	1 KB	2022/11/21 17:53:44	rw-r--r--	root	
favicon.ico	2 KB	2022/11/21 17:53:47	rw-r--r--	root	
hue.f8505bd4d6f3e3aa435b.png	1 KB	2022/11/21 17:53:41	rw-r--r--	root	
index.html	6 KB	2022/11/24 15:25:01	rw-r--r--	root	
main-es5.0d6a284990d61c727307.js	610 KB	2022/11/21 17:53:44	rw-r--r--	root	
main-es2017.0d6a284990d61c727307.js	528 KB	2022/11/21 17:53:47	rw-r--r--	root	

进入红框位置， 配置服务端 ip 和端口

/data/rm-log/log-server-web/webapp/assets/

名字	大小	已改变	权限
..		2022/11/21 17:53:41	rw-r--r--
animal		2022/11/21 17:51:18	rw-r--r--
css		2022/11/21 17:51:18	rw-r--r--
fill		2022/11/21 17:51:53	rw-r--r--
img		2022/11/21 17:51:52	rw-r--r--
js		2022/11/21 17:51:52	rw-r--r--
outline		2022/11/21 17:53:03	rw-r--r--
twotone		2022/11/21 17:53:29	rw-r--r--
area.js	255 KB	2022/11/21 17:53:54	rw-r--r--
config.js	1 KB	2022/11/21 18:02:45	rw-r--r--

/data/rm-log/log-server-web/webapp/assets/config.js - root@192.168.111.243 - 编辑器 - WinSCP

```
const CONFIG = {  
  path: "http://192.168.111.243:8030/",  
  bsp: "http://192.168.111.243:8108/",  
  url: "/rmApi/url/get",  
  isUrl: false,  
  version: "V0.5.0",  
};
```

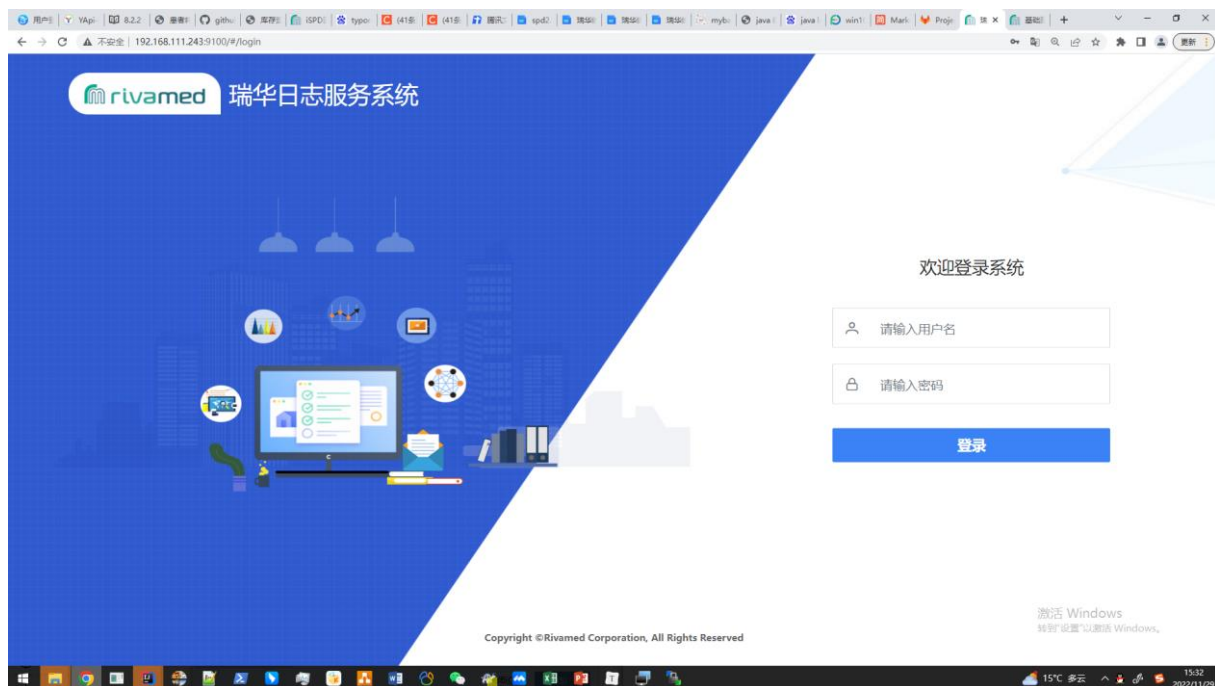
1 rm-log服务ip和端口
2 bsp服务ip和端口

nginx 配置 rm-log 前端服务

```
server{  
    listen 9100;  
    location / {  
        root /data/rm-log/log-server-web/webapp;  
        index index.html;  
    }  
}
```

nginx -s reload,重新加载配置文件

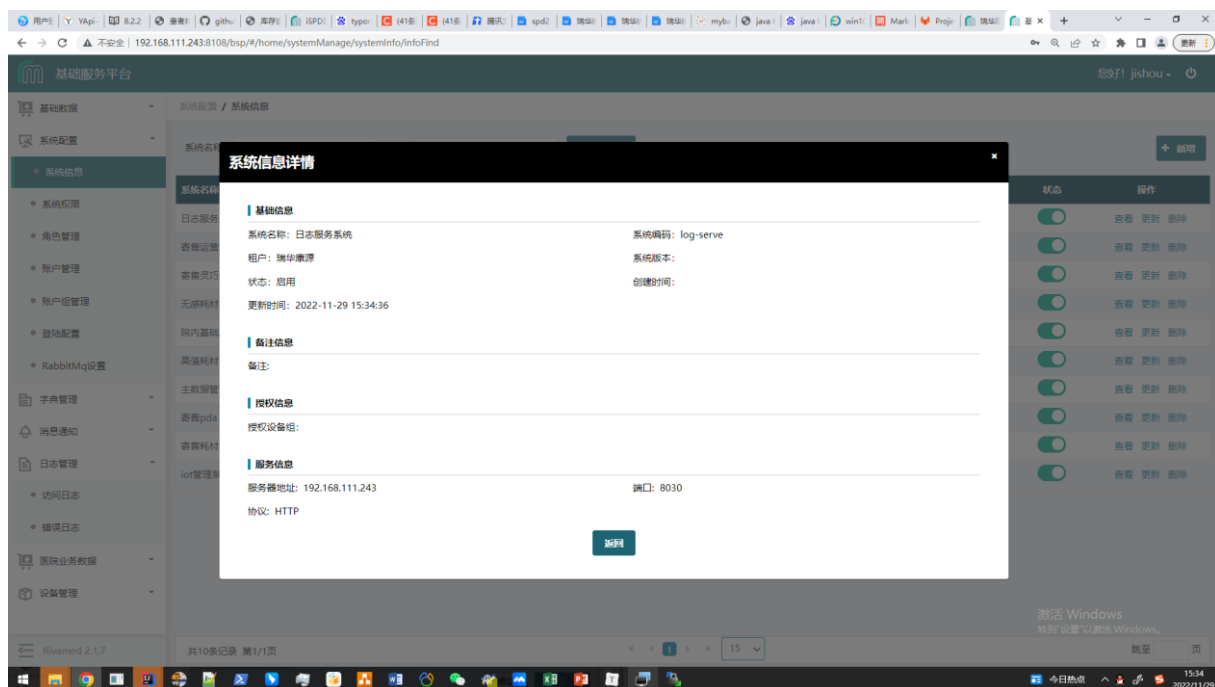
浏览器输入前端地址：<http://192.168.111.243:9100/#/login>



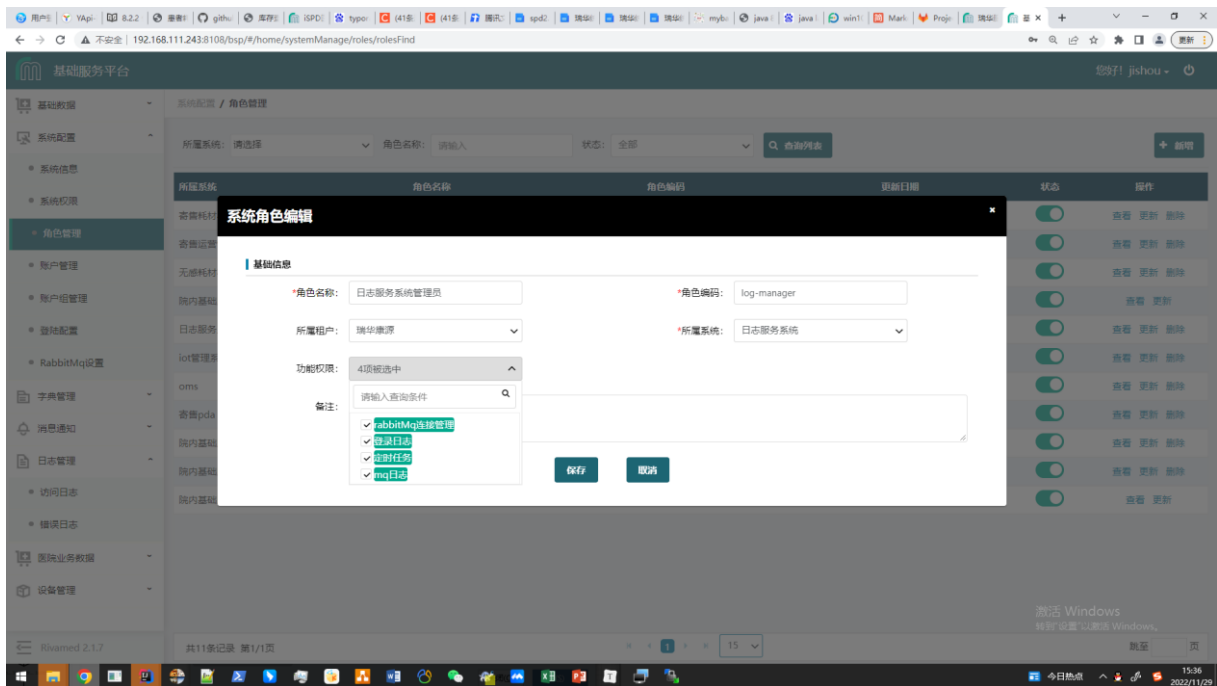
显示如图页面，表明前端部署成功

bsp 配置：

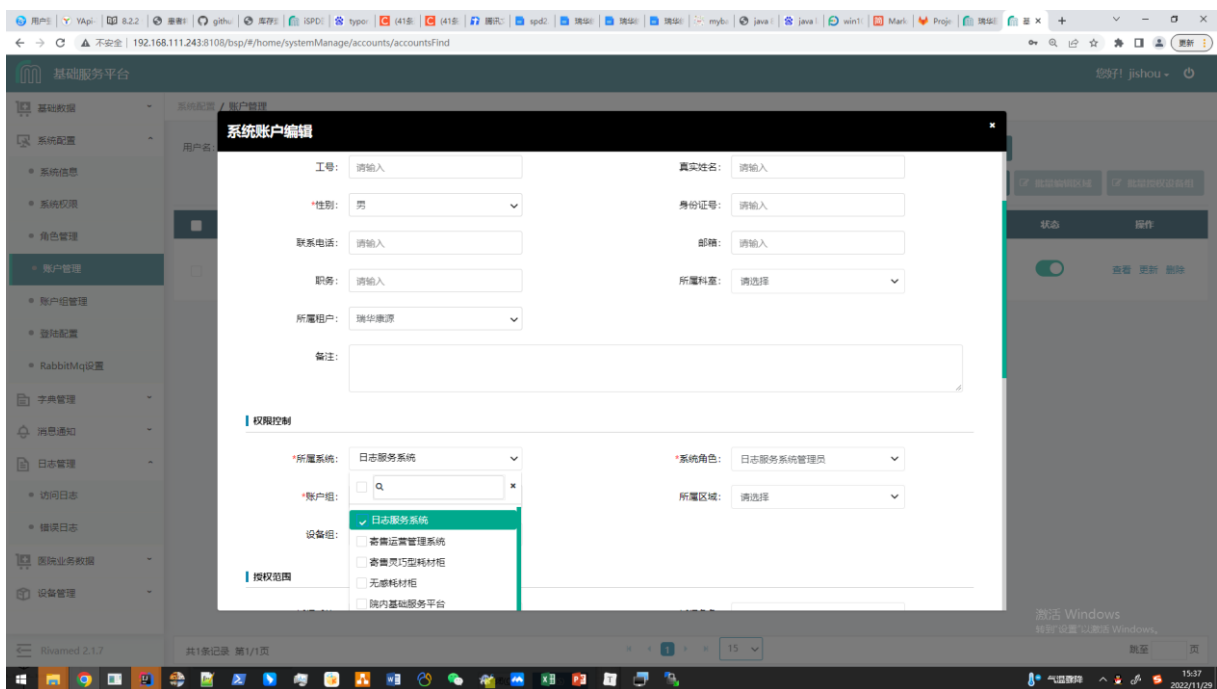
bsp 系统信息->新增系统



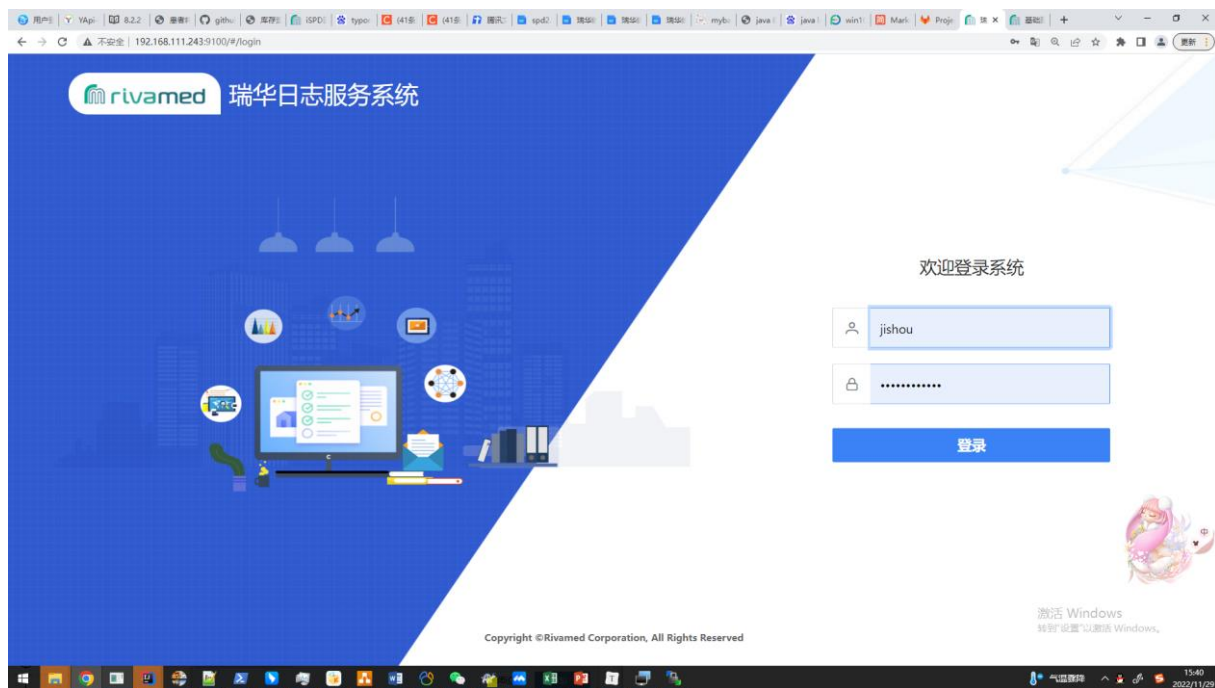
新增角色：日志管理员，添加系统菜单



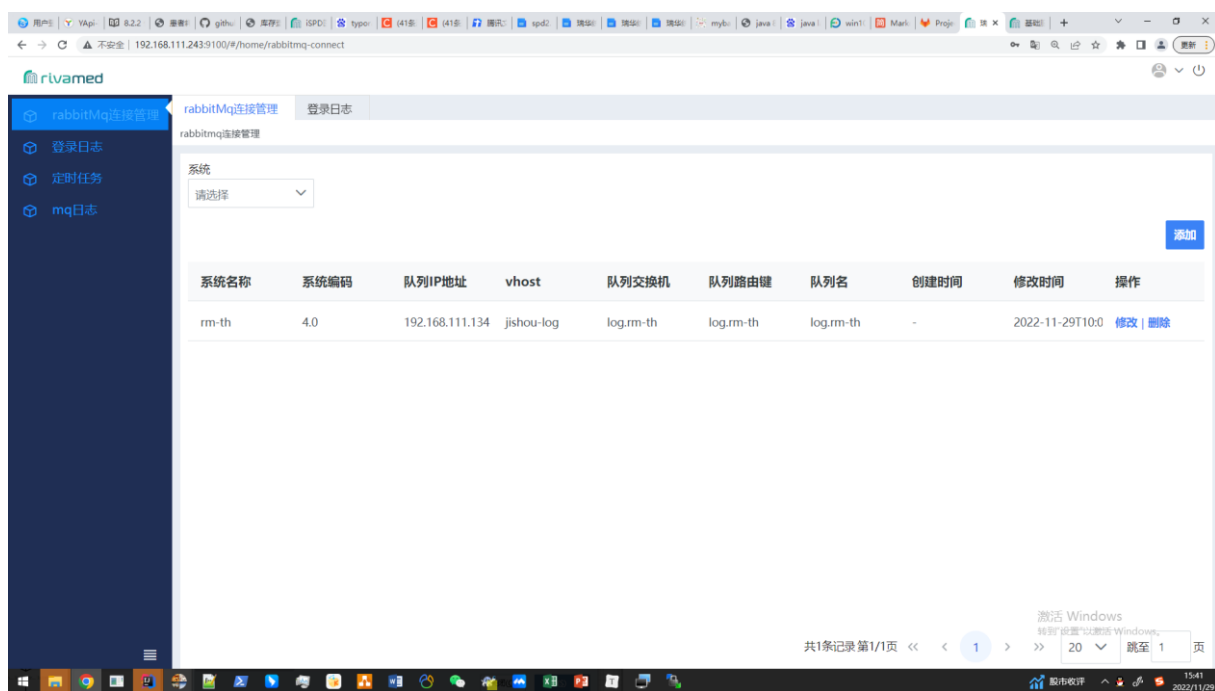
账户管理->更新,添加用户所属系统（日志服务系统）和角色（日志服务系统管理员）



这样系统角色权限就配置好了，可以登录日志服务了

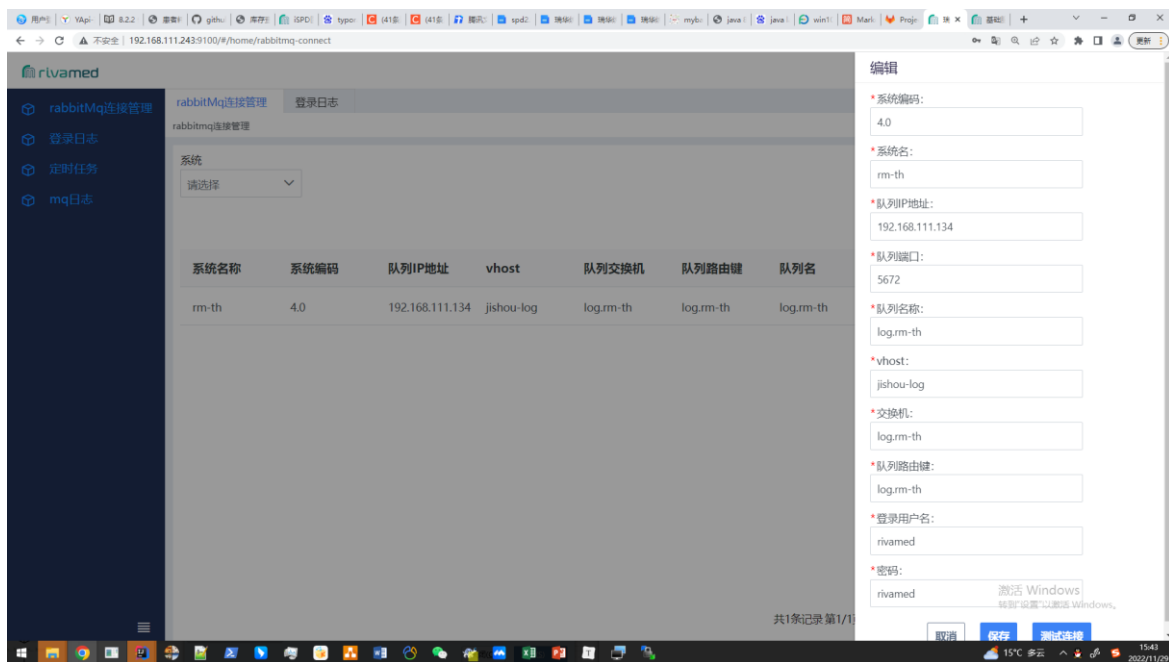


登录成功，界面如下：



7. 业务系统接入

日志服务系统中，配置系统信息以及对应的日志队列



业务服务集成日志框架

参照：spring 项目接入文档.md

业务系统接入后，就可以在系统上看到对应的日志记录了。

8. 性能测试

个人电脑：192.168.111.82

处理器 Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz 3.41 GHz(4 核心)

RAM 16.0 GB

系统类型 64 位 windows10 操作系统, 基于 x64 的处理器

部署程序：业务系统、rm-log、rabbitmq

服务器：192.168.111.134

处理器 2.5 GHz(4 核心)

RAM 12.0 GB

系统类型 ubuntu-20.04.2-live-server-amd64

部署程序：mongodb

日志推送测试：

单条 1Kb,最大推送速度<20K 条/s

日志消费测试 (一个 rm-log、 一个 mongodb):

单条 1Kb,最大消费速度<10K 条/s

9. 测试环境

<http://192.168.111.243:9100/> jishou Jishou123456