<云作业平台>

软件架构文档

版本 <4.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2020/10/29 | 1.0 | 部署架构及逻辑架构 | 第四小组 |
| 2020/11/07 | 2.0 | 完善架构文档 | 第四小组 |
| 2020/12/10 | 3.0 | 修改架构文档 | 第四小组 |
| 2021/1/3 | 4.0 | 修改架构文档 | 第四小组 |

目录

1. 简介 4

1.1 目的 4

2. 用例视图 4

3. 逻辑视图 5

3.1 在架构对较重要的部分介绍 5

4. 进程视图 6

5. 部署视图 6

5.1 移动设备： 6

5.2 网关： 6

5.3 认证服务器： 6

5.4 配置服务器： 6

5.5 注册和通知服务器： 6

5.6 作业服务器： 6

5.7 搜索服务器： 6

5.8 MySQL及Mongodb： 7

5.9 ELK： 7

6. 实现视图 7

7. 技术视图 7

8. 数据视图 8

9. 设计战术 9

软件架构文档 （简化版）

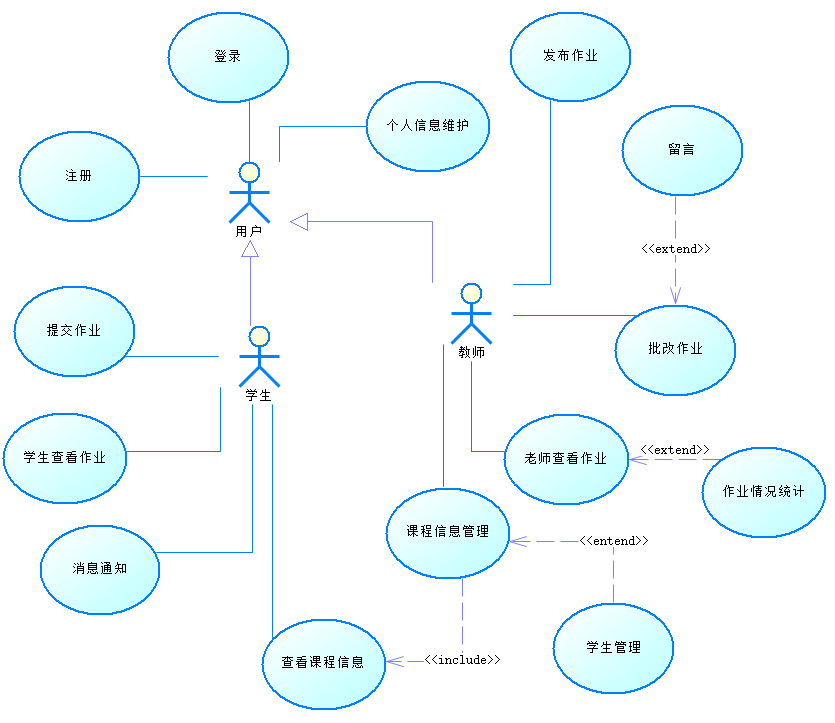
# 简介

## 目的

本文档将从架构方面对云作业系统进行综合概述，其中会使用多种不同的架构视图来描述系统的各个方面。它用于记录并表述已对系统的架构方面做出的重要决策。

# 用例视图

本节列出了用例模型中的一些用例，这些用例或场景体现云作业平台中重要的、核心的功能；



注册：用户填写学号、工号等相关信息，并通过邮箱或手机号完成验证激活

登录：用户通过账号、密码登录

个人信息维护：用户修改维护个人信息

发布作业：教师发布多样化的作业名称、内容、截止时间及参考答案，系统会对上传的图片进行清晰度检测与优化，上传的图片和语音可转化为文本形式

提交作业：学生填写答题卡、上传多种格式的作业结果，且可对教师留言，系统会对上传的图片进行清晰度检测与优化，上传的图片和语音可转化为文本形式

查看作业信息：学生查看作业信息，包括作业内容、截止时间等，提交作业后可查看自己的答题情况、评分、批注、参考答案等

批改作业：老师单独或批量对选中的学生作业进行留言、评分或批注，批注可附加文本也可直接在图片上批注。

课程信息管理：包含课程的增删改

查看课程信息：查看课程简介、教材等课程信息

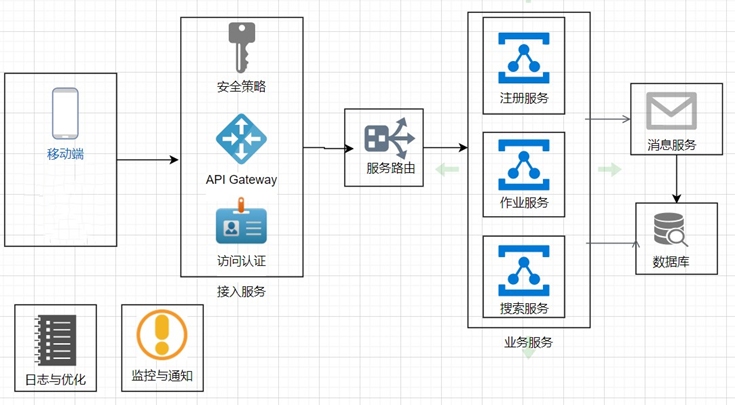
学生管理：老师添加删除学生，可以通过学号，也可以通过excel导入。

留言：老师可对学生作业进行留言。

消息通知：发布作业后，学生可收到作业通知。

作业情况统计：统计作业提交数量以及最高分，最低分和平均分。

# 逻辑视图



## 在架构对较重要的部分介绍

移动端：移动端为学生与教师用户

接入服务：安全策略对应用进行安全保护，访问认证对用户进行认证，授予相应的权限，API Gateway和服务路由对接收的请求分配到相应的服务器，使资源均衡

业务服务：根据业务类型分配相应的业务服务器，每个服务器主要处理该业务。注册服务主要处理注册流程、邮件短信认证等服务；作业服务主要处理与作业相关的流程；搜索服务对搜索请求进行处理

数据库存储应用所需持久化保存的数据

日志与优化记录应用日志信息，对日志进行监控

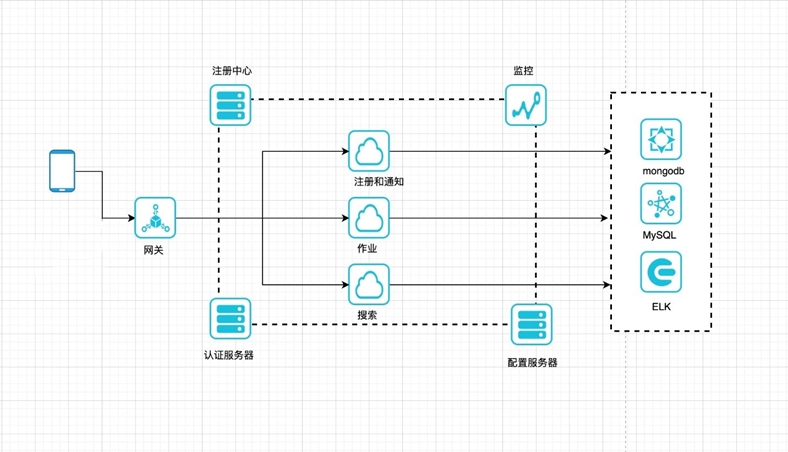
监控与通知对应用进行监控，反馈相应信息

# 进程视图

该应用多进程实现主要由spring cloud框架完成

# 部署视图

本节主要说明部署运行该云作业平台所需的架构



## 移动设备：

学生，老师通过移动设备进行登录注册及相关功能的使用，这些设备通过网络访问服务器网关。

## 网关：

网关把接收到的请求分配到相应的服务器，所有用户都通过网络访问网关。

## 认证服务器：

该服务器对访问的用户进行身份鉴别，判断其权限，决定其可访问的服务器及功能。

## 配置服务器：

该服务器存储各服务器中共有的配置信息。

## 注册和通知服务器：

该服务器处理用户的注册申请，并对用户发送各项通知信息。

## 作业服务器：

该服务器处理与作业相关的请求，及业务逻辑。

## 搜索服务器：

该服务器对搜索请求进行处理，返回搜索结果。

## MySQL及Mongodb：

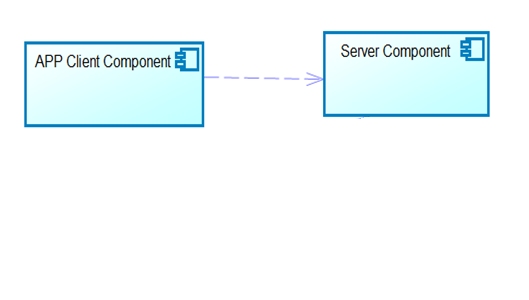
云作业平台的数据库，存储系统所需的信息。

## ELK：

日志管理系统。

# 实现视图

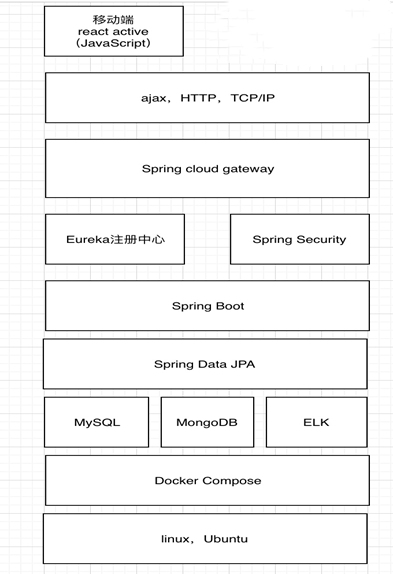
本节说明实现模型的整体结构，主要分为app client和server两个构件，即移动端和spring cloud实现的服务端。



# 技术视图

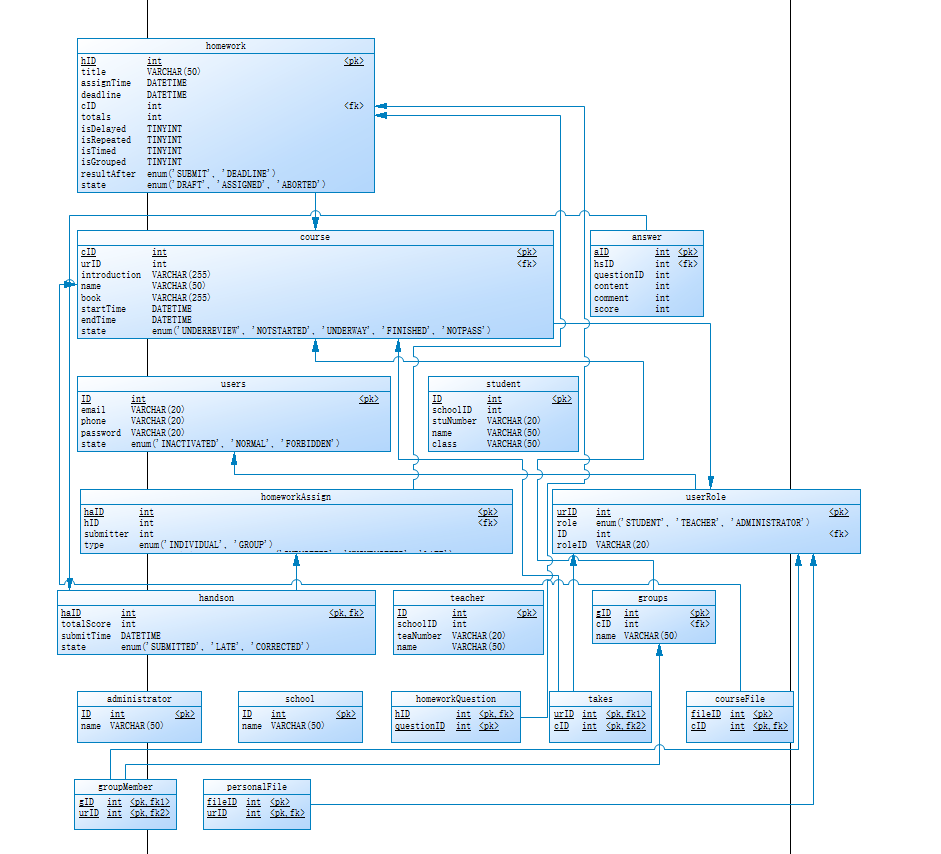
本节主要描述该云作业平台涉及的技术框架等。

1. 移动端应用react native实现，语言为javascript
2. 前端通过ajax, HTTP, TCP/IP实现网络通信，数据传输
3. 访问过程先经过spring cloud gateway，由网关分配服务器
4. Eureka注册中心及spring security为微服务一部分，负责服务注册，用户认证，权限控制等
5. Spring boot为后端基础框架
6. 后端通过Spring data JPA访问数据层
7. 数据层为MySQL和Mongodb，微服务由ELK进行日志管理
8. Docker compose对服务进行打包运行
9. 服务运行在linux, ubuntu上



# 数据视图

本节主要描述MySQL关系型数据库中存储的应用数据



# 设计战术

|  |  |
| --- | --- |
| **质量因素** | **设计战术** |
| 易用性 | 1. 为用户提供适当的反馈及协助（常见问题及用户手册） 2. 将用户的接口与应用的其余部分分离 3. 提供“取消”，“撤销”，“修改”，“预览”等命令功能 4. 界面美观简洁，对用户友好，便于使用 |
| 可用性 | 1. 错误检测，eureka注册中心对注册的服务进行检测 2. 错误恢复，冗余及日志，数据库恢复 3. 错误预防，spring cloud集群 4. 错误监测，spring boot admin，actuator，Prometheus对系统监测 |
| 性能 | 1. 资源管理，引入并发、增加可用资源、维持数据或计算的多个副本 2. 对数据库增加缓存，MySQL主从同步 |
| 安全性 | Spring security用户身份验证、用户授权、限制暴露的信 息和端口 |
| 可测试性 | 测试的输入/输出，将接口与实现分离 |
| 可维护性 | 局部化修改，模块化开发 |