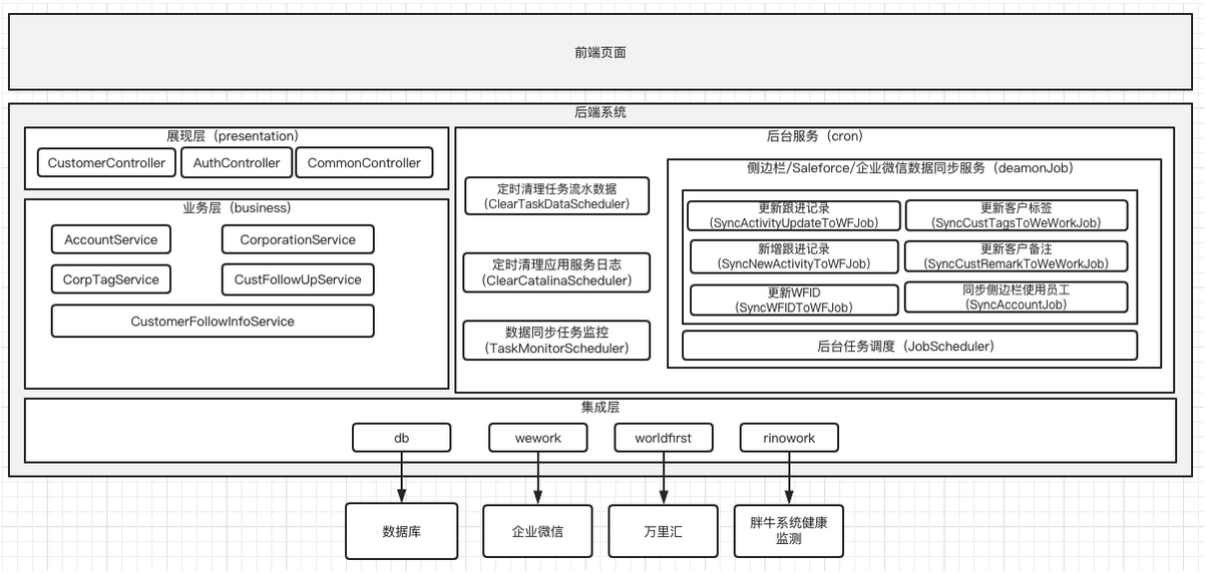
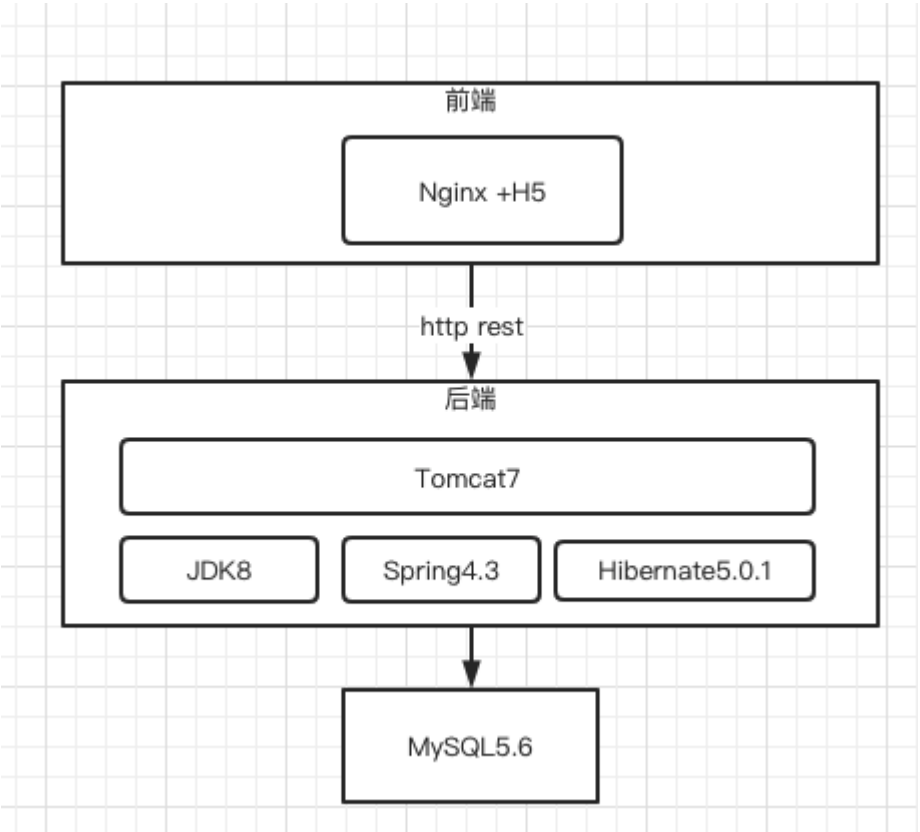
胖牛会话存档设计文档

1. 架构设计

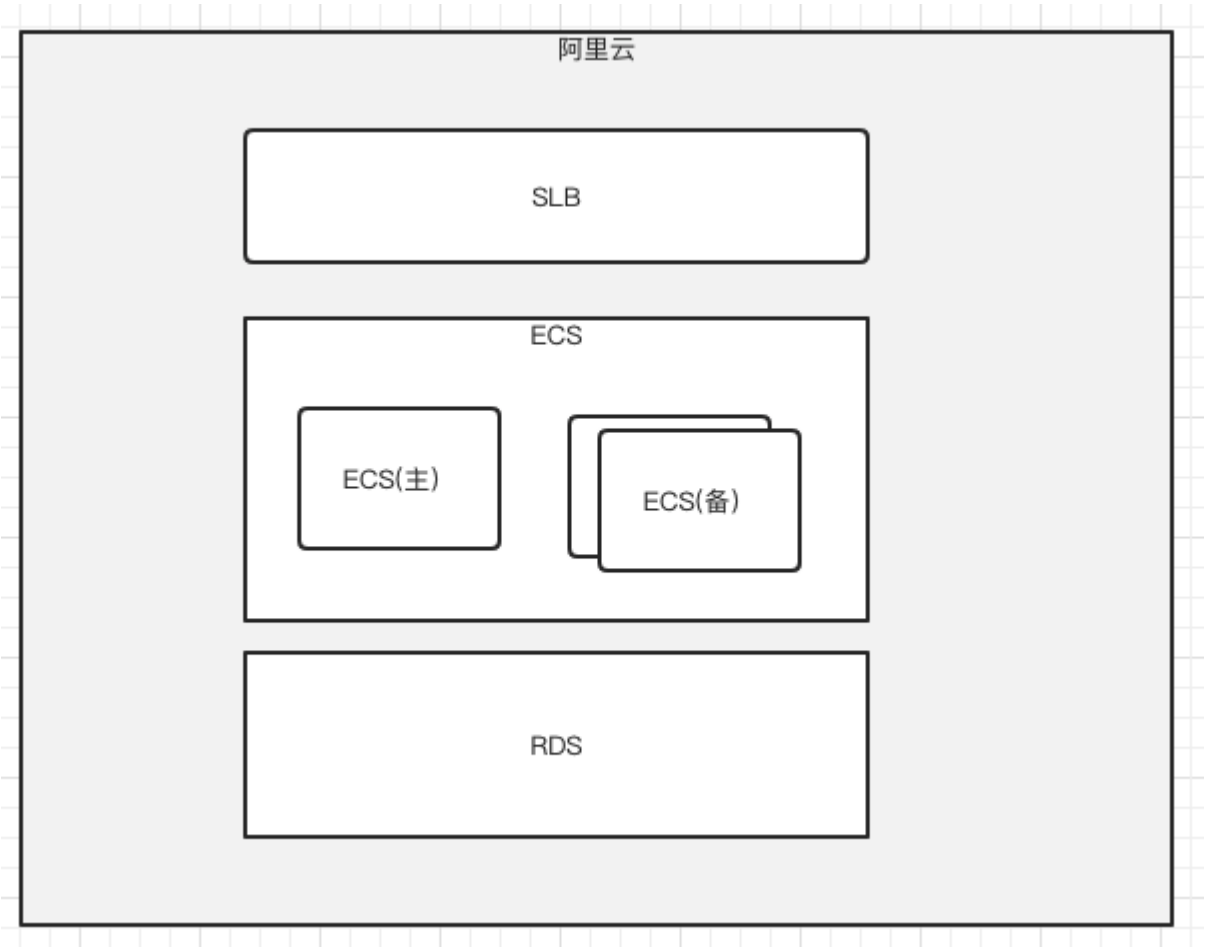
1.1 应用架构



1.2 技术架构



1.3 部署架构



1.3.1 高可用策略

- Master-Slave 模式：在处理外部请求方面 masterSlave 是一样的，可相互备份，但仅 Master 处理后台数据同步任务；
- Master 和 Slave 动态切换：Master 主机每 3 秒钟发送一个心跳信号，Slave 主机每 3 秒钟检查一次 Master 主机的心跳；如果 Slave 主机发现 Master 主机僵死（超过 3 分钟没有心跳信号），就自动切换成 Master 主机，原 Master 主机恢复后自动切换成 Slave 主机；
- 在所有主机上部署有 cron 任务，每 1 分钟检查一次后台服务进程，如果进程宕机则自动重启后台服务进程；
- 通过 SLB 流量分配调整，可实现灰度发布；
- 通过 tomcat 热部署机制，实现不停机发布。

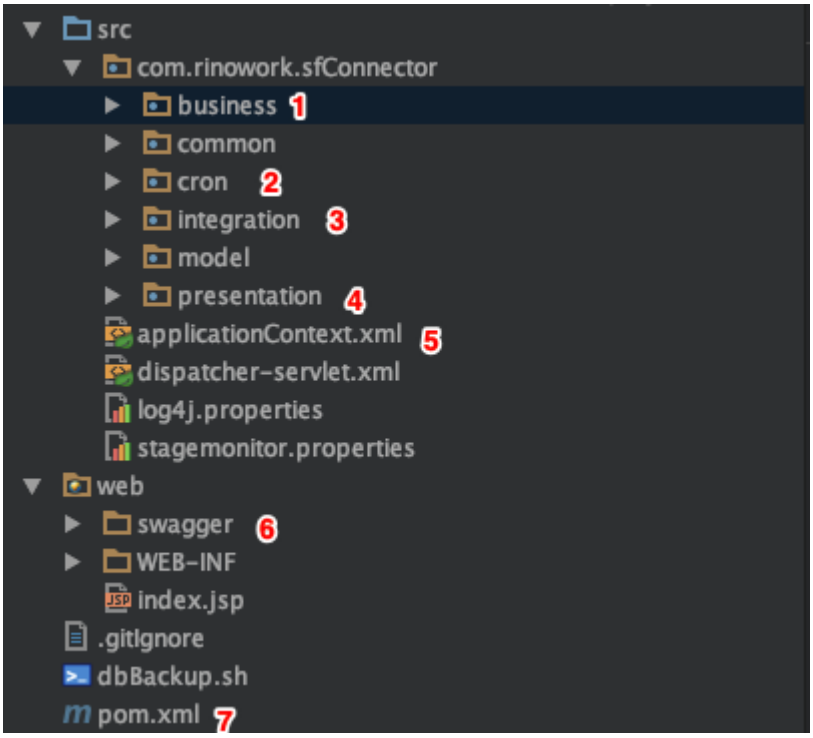
1.4 ER 图

2. 接口设计

地址：

略

3. 代码结构



1 - 业务服务，包括授权登录，客户查询及更新，跟进记录查询及更新；

2 - 后台任务，主要负责在侧边栏系统、万里汇、企业微信之间双向同步数据；

3 - 集成层，跟数据库、万里汇、企业微信集成

4 - 展现层

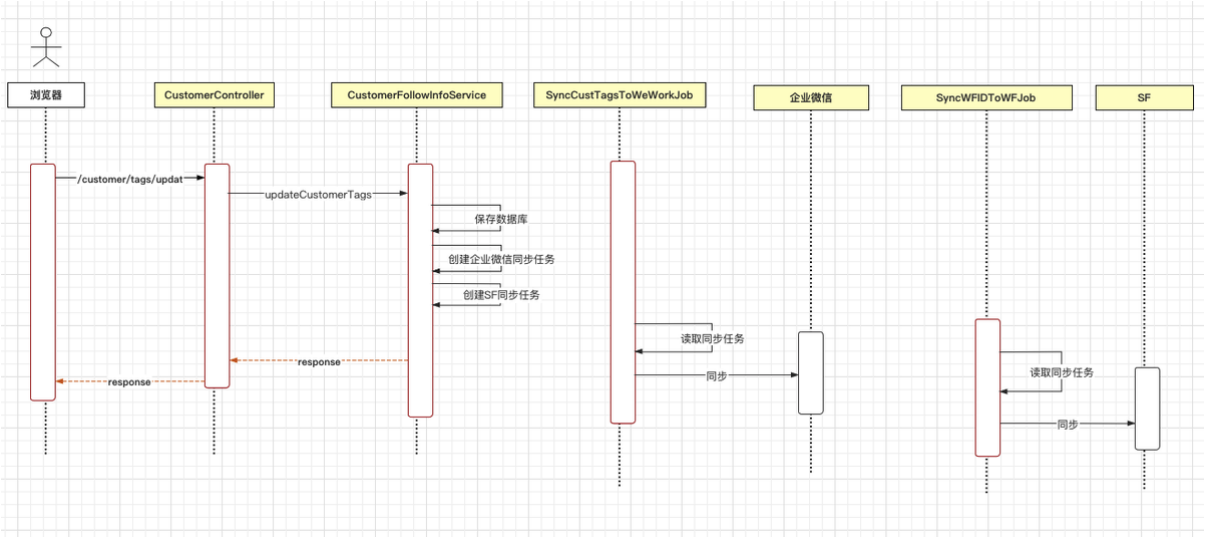
5 - spring 框架配置文件，主要配置数据源；

6- 接口文档生成组件；

7 - maven 配置文件，配置第三方 jar 包依赖。

4. 核心交互流程（以“更新客户标签”为例）

4.1 更新客户标签



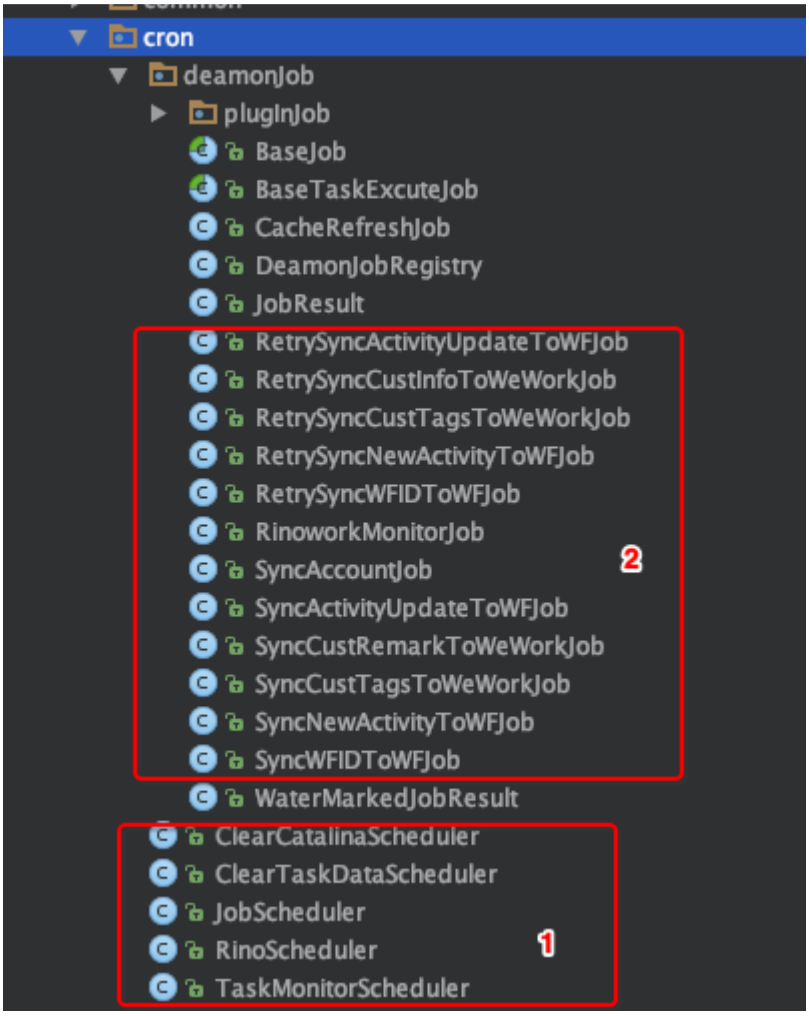
1) 员工在侧边栏更新客户标签；

2) 浏览器提交到/customer/tags/update 接口
3) 业务服务 customerFollowInfoService 做如下处理：
   a) 保存到数据库；
   b) 创建一个数据同步后台任务（type=向企业微信同步，name=更新客户标签）；
4) SyncCustTagsToWeWorkJob（更新客户标签后台任务）每隔 1 秒轮询数据库，抓到新任务后就执行任务，向企业微信同步该客户的标签；
5) #4 执行成功后，把#3 创建的数据同步任务的状态设为‘已成功’，流程结束；
6) #4 如果失败，把任务状态设置‘待重试’；
7) RetrySyncCustTagsToWeWorkJob（更新客户标签_重试）每隔 1 秒轮询数据库，发现待重试任务（type=向企业微信同步，name=更新客户标签），就发起重试；重试最多 5 次，成功则置状态“已完成”，仍然不成功则置为‘已失败’；流程结束。

8) SyncWFIDToWFJob 负责向 SF 同步，执行过程类似 SyncCustTagsToWeWorkJob。

5. 后台任务设计

后台任务分 2 类，如下图：



- 类型 1 是辅助型任务，由 cron 调度运行；
- 类型 2 负责数据双向同步，启动后一直循环运行，每次运行间隔 1 秒；
- 类型 2 任务统一由类型 1 任务 JobScheduler 启动。

每个数据同步任务，都对应一个重试任务（以 Retry 开头）；这么设计主要是避免重试操作阻塞了正常的数据同步，尽可能保障数据同步的时效性。