PassThemOn

软件架构文档

版本 <4.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 21/7/2021 | 1.0 | 初步完成软件架构文档 | 王浩天 陶昱丞 徐惠东 周昱宏 |
| 22/7/2021 | 2.0 | 添加完成质量属性设计部分 | 徐惠东 |
| 12/9/2021 | 3.0 | 更新修改架构文档 | 周昱宏、徐惠东 |
| 13/9/2021 | 4.0 | 更新修改架构文档 | 徐惠东 |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

3. 逻辑视图 4

3.1 概述 4

3.2 在构架方面具有重要意义的设计包 4

4. 进程视图 4

5. 部署视图 4

6. 实现视图 5

7. 技术视图 5

8. 数据视图（可选） 5

9. 核心算法设计（可选） 5

10. 质量属性的设计 5

软件架构文档

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架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构架方面作出的重要决策。

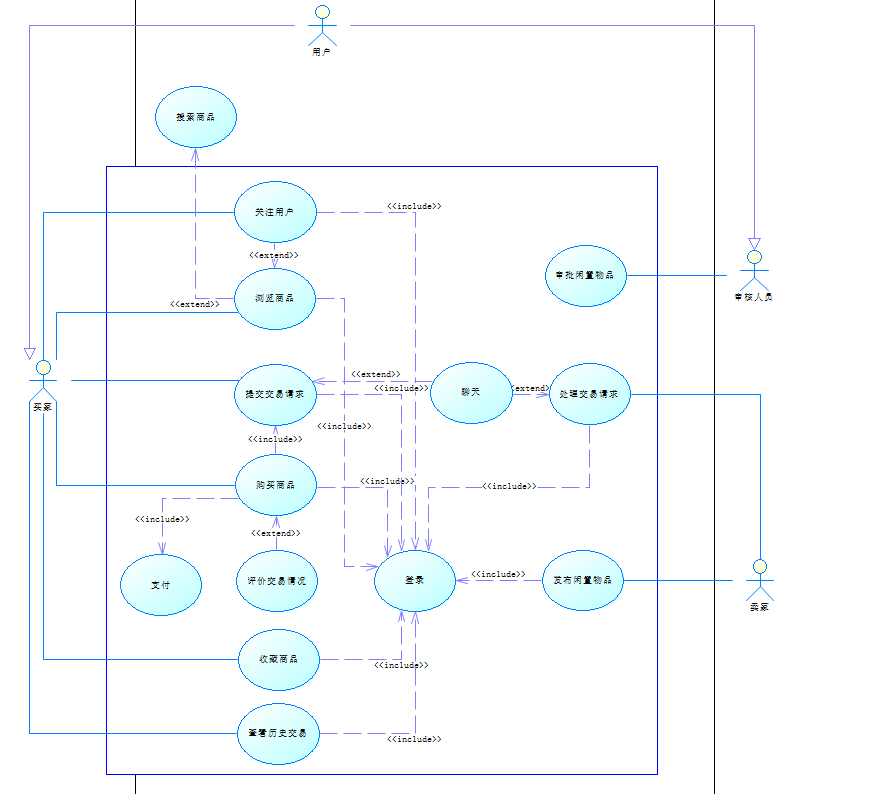
## 参考资料

《软件架构文档 template》

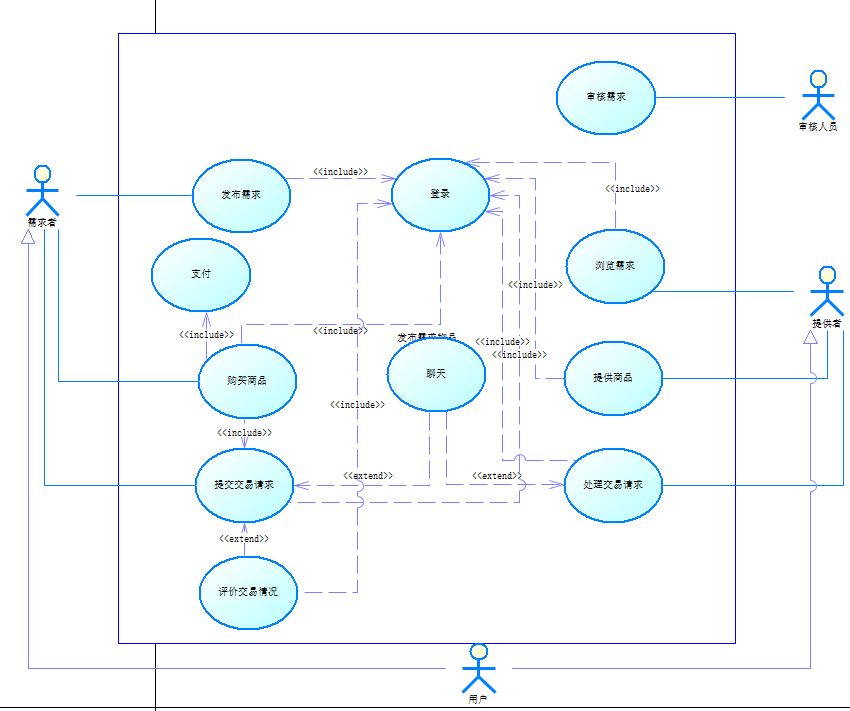
《软件工程原理》

# 用例视图

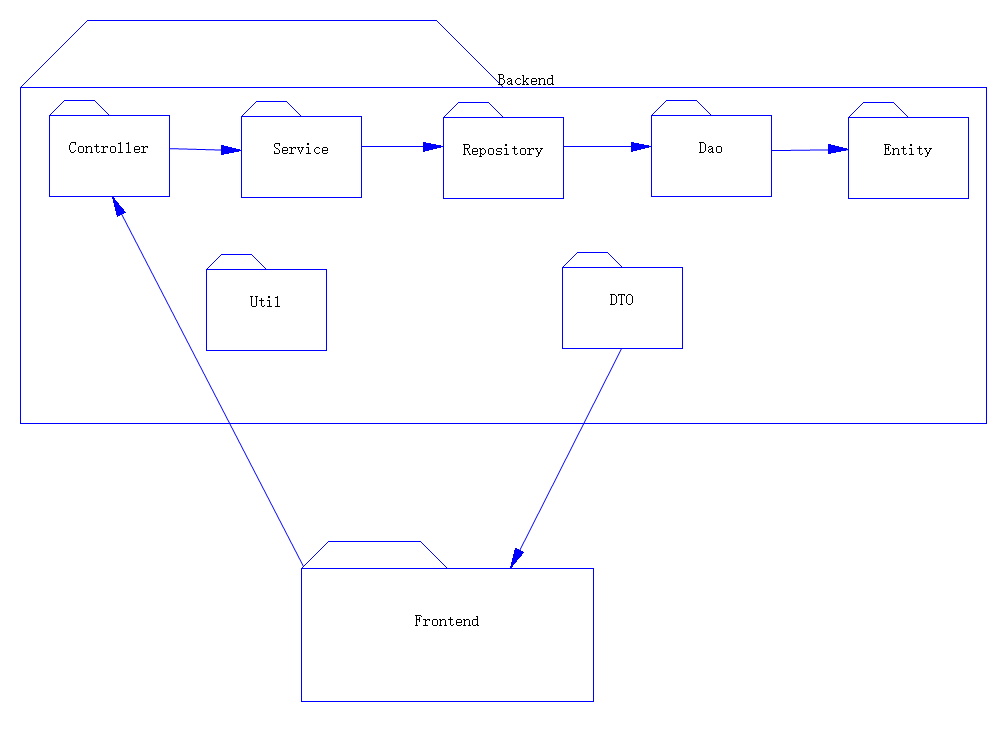
Use-case1二手物品交易：



Use-case2发布需求：



# 逻辑视图



## 概述

Backend:

1、Controller：连接前端和后端，接受前端发来的请求，同时向后端发送请求，负责业务调度。

2、Service：服务层，编写具体业务逻辑，更贴近于具体业务功能的要求，将对于数据访问的操作封装起来。

3、Repository:仓库，向上层屏蔽了数据来源和内部实现细节。

4、Dao：数据访问层，全称为data access object，属于一种比较底层，比较基础的操作，具体到对于某个表、某个实体的增删改查。

5、Entity：实体bean，用于ORM对象关系映射，一个实体映射成一张表，一般无业务逻辑代码，修改entity可以同步到数据库。

6、Util：功能包，针对本项目的工具类。主要用于存在项目各层都有可能出现、不好划分到某层中、出现频率较高的功能（类）。

7、DTO：数据传输对象（Data Transfer Object），是一种设计模式之间传输数据的软件应用系统。

8、Config：Rabbitmq、Redis和WebSocket接口类，用于对接各种服务。

9、Constant：常数类，用于定义返回值含义与数值对应。

10、Consumer：消息队列服务，包括上传商品、更改商品、上传需求、更改需求时进行文本图像审核的函数方法。

11、Filter：过滤器，解决跨域问题和访问权限。

12、Interceptor：拦截器，拦截非法请求。

13、Redis：Redis服务。

14、WebSocket：WebSocket服务。

15、Wrapper：包装类，将Http请求包装。

Frontend：前端用户访问界面。

## 在构架方面具有重要意义的设计包

无

# 进程视图

由于采用框架处理，因而没有进程视图

# 部署视图

# 

# 实现视图

采取垂直分割方案，将业务大致分为发布闲置物品、购买闲置物品、聊天、支付、首页推荐5个子系统，将其分配给几个不同的开发人员，每个开发人员负责实现对应模块中从Web层到数据库层的所有开发，且每天进行集成。

# 技术视图

编程语言：后端使用JAVA,前端使用Javascript

开发工具：Intellij IDEA

框架：后端使用Spring boot框架，手机端使用React Native框架，网页端使用React。

数据库：mysql数据库,Mongodb数据库，redis。

中间件：hibernate,tomcat,rabbitmq

# 数据视图（可选）

无

# 核心算法设计（可选）

暂无

# 质量属性的设计

本项目使用SpringBoot框架，其优点主要包括轻量级组件、开发简便且组件解耦使得开发灵活。。

我们使用Redis-Cluster分布式解决方案，针单个Redis服务在服务器宕机时会造成单点故障且Redis为单线程操作无法应对高并发问题，运用Redis-Cluster实现去中心化，集群最大可增加1000个节点，性能随节点增加而线性扩展，大大提升软件性能和可拓展性。

我们采用Nginx负载均衡，将大量请求进行分布式处理，对大量前端访问或流量进行分流，保证用户能够正常访问APP，并可以减少后端服务器处理压力，大大提升软件性能和可靠性。

我们引入rabbitmq消息队列，建立生产者消费者模型，将用户发送的文本图像审核请求放入相应队列，并在消费者端进行审核，通过异步的特点加快用户端响应速度，提升软件功能质量和易用性。