交大林檎—疫情物品互助平台

软件架构文档

版本 <1.2>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 18/11/2022 | 1.0 | 软件架构文档制定 | 敖晟昊 |
| 16/12/2022 | 1.2 | 修改设计 | 敖晟昊 |
|  |  |  |  |
|  |  |  |  |

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

3.2.1 物品管理应用

3.2.2 登录应用

3.2.3 前端应用

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 5

9. 质量属性的设计 5

软件架构文档

# 简介

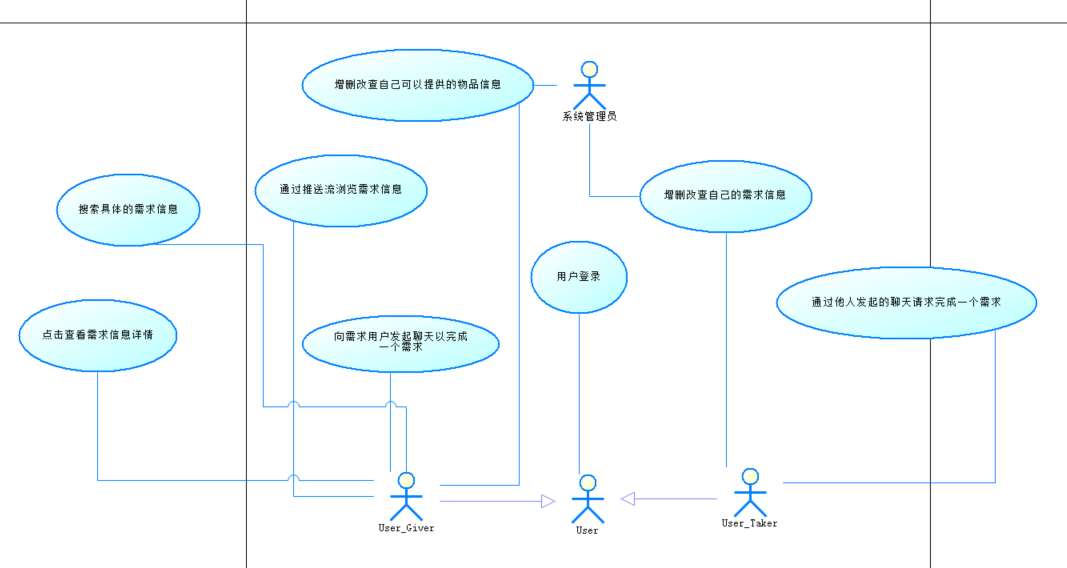
## 目的

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

无

# 用例视图

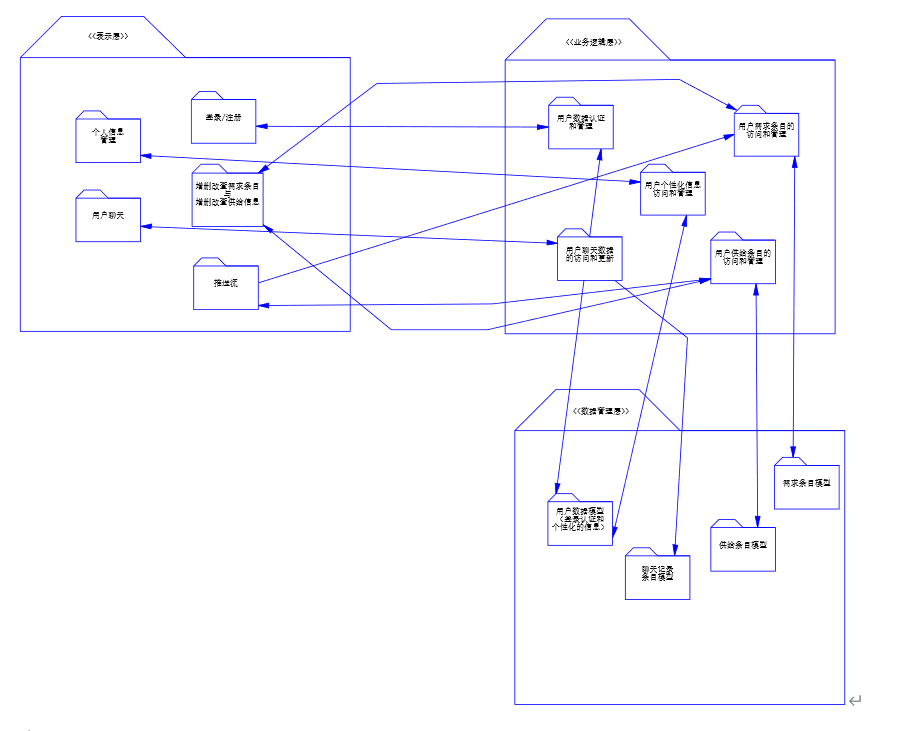


如图是软件的核心用例视图。主要用例为用户发布自己的需求和提供条目、浏览提供条目，以及向他人发起聊天，沟通物品交接细节。可分为用户信息管理、物品信息管理、聊天功能、搜索功能等几个大模块。

# 逻辑视图

## 概述

逻辑架构图如下。本软件采用前后端分离的设计风格，后端使用Django开发默认的MVC设计风格。界面和渲染完全由前端负责，后端只负责发送数据和处理数据。具体地，可分为负责展示页面的表示层、负责数据访问的业务逻辑层，以及存储数据的数据管理层。

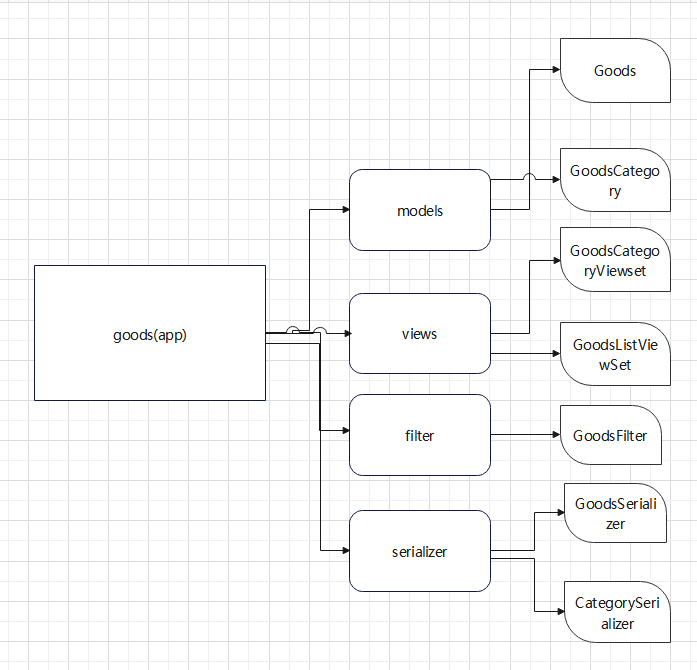


## 在构架方面具有重要意义的设计包

### **物品django应用: goods**

用于物品管理的应用。具有一个用于存储物品信息的django model。其中通过外键和物品类别model相关联，使物品具备多级类别。用于它的视图方法通过继承相对应的mixin类，可以具有创建、删除、查看等功能。同时还具有用于序列化存储数据的Serializer和用于筛选查询结果的Filter。

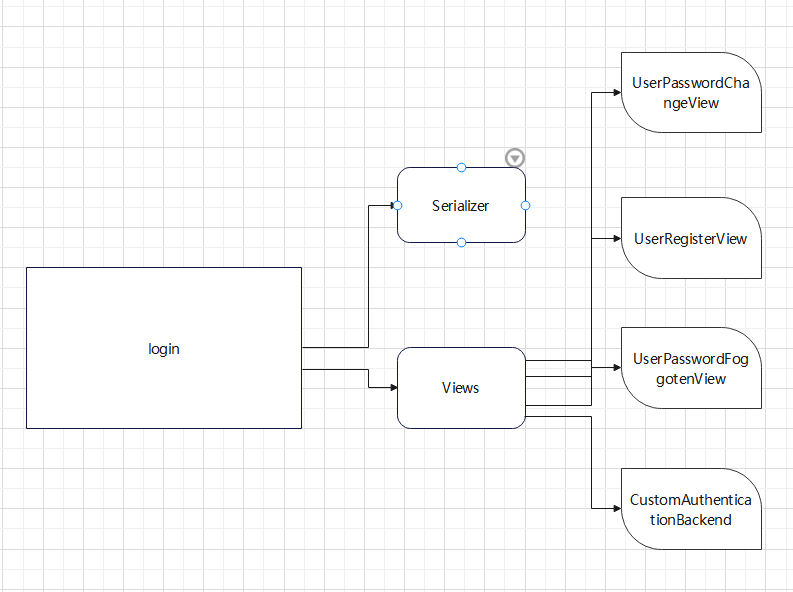
#### **架构：**



### **登录和注册应用：login**

Login应用用于管理用户的注册、登录和修改密码等用户相关操作。它视图函数中的UserRegisterView通过重写post方法用于用户注册，UserPasswordFoggotenView用于忘记密码的重置操作，UserPasswordChangeView用于修改密码。

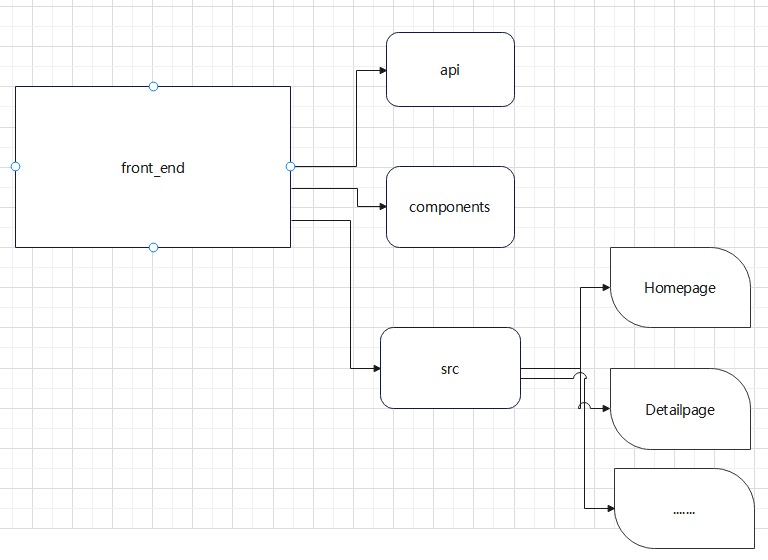
#### **架构：**



### **前端应用:front\_end**

Vue框架下构建的前端应用。包含可复用的components组件包、规定和后端交互接口的index,显示、通过http请求获取或发送数据的src包。Src包中为负责各种页面的vue文件，通过绑定方法实现页面间的跳转、显示数据，并且通过axios库和固定的接口向后端给出的相应url发送请求，从而获取数据或发送数据。

#### **架构：**



# 进程视图

本项目大致可分为前端服务器进程和后端服务器进程，分别监听在8081端口和8080端口。后端服务器进程收到请求（来自前端服务器）时，会按照相应的url配置，派生子线程，将请求转发到相应的视图函数中进行处理；前端在收到请求时，会根据绑定的方法配置，向后端发送请求、渲染页面。

# 部署视图

目前软件部署在本地服务器，可以通过本地回环地址访问。前端进程和后端进程都映射到本机上，监听着不同的端口。

# 实现视图

实现模型可分解为：

前端进行展示和响应。用户在网页上触发事件后，前端向后端服务器的某约定好的url发送请求，并根据返回数据渲染相应的页面。同时，实现页面间的跳转。

后端的服务器进程监听请求并重定向，交由相应视图函数处理请求。具体地，和物品管理相关的请求，如发布需求、删除提供等，交由Goods应用的视图函数处理；登录、注册等请求则交给Login应用处理。

视图函数根据请求中的数据和请求头等，处理、更新数据，和数据库交互，并返回Response。

# 技术视图

前端采用Vue.js编写。使用JavaScript。

后端采用Django编写，数据库使用Sqlite, 聊天功能的实现采用Websocket以及Channels，数据缓存使用Redis。编程语言主要是python.

# 数据视图

利用django中的model概念进行数据的存储。它将传统上Sql的繁琐操作简易化，可以较为用户友好型地构建某类的各种字段，并通过migrate在数据库中创建它的相关表。和数据库交互时，通过视图集抽象访问，利用自带的序列化器将数据序列化，进行交互。同时，通过Redis的缓存功能缓解了对数据库的压力。

# 质量属性的设计

基于成熟框架，软件的基本性能得到保证；基于django的强大可扩展性，要增加新功能只需注册新的应用、创建新模型，并给出相关url，前端就可以通过访问该url来和新功能交互；只要安装了相关库，就可以随处运行，基本不依赖本地硬件环境。

用户的密码属于隐私信息，应该得到保密。在我们的实现中，前端不保留明文密码，而是通过登录后从后端获取的经过加密的token来和后端交互，进行用户隐私相关的操作，保证了安全性。