山东大学 软件 学院

软件工程(软件工程全过程类)课程实验报告

201900301053 孔明浩 负责部分：项目经理、设计师、程序员、测试工程师

成员名单

|  |  |  |
| --- | --- | --- |
| 孔明浩 | 201900301053 | 项目经理 |
| 郭常瑞 | 201900301131 | 成员 |

目录

[项目介绍 2](#_Toc89201176)

[1.1 项目背景 2](#_Toc89201177)

[产品设计 2](#_Toc89201178)

[2.1 找猫 3](#_Toc89201179)

[2.1.1 猫谱 3](#_Toc89201180)

[2.1.2 领养版块 4](#_Toc89201181)

[2.1.3 猫咪详情页 4](#_Toc89201182)

[2.1.4 科普 5](#_Toc89201183)

[2.3 社区 5](#_Toc89201184)

[2.3.1 山猫速报 6](#_Toc89201187)

[2.3.2 猫友圈 6](#_Toc89201188)

[2.3.3 评论的写操作逻辑 7](#_Toc89201189)

[2.3.4 评论的展示逻辑 7](#_Toc89201190)

[2.3.5 评论的互动功能 8](#_Toc89201191)

[2.3 我的 8](#_Toc89201192)

[2.4.1 报送新猫 8](#_Toc89201194)

[2.4.2 通知 8](#_Toc89201195)

[2.4.3 发表与赞过 9](#_Toc89201196)

[技术方案 9](#_Toc89201197)

[3.1 前端 9](#_Toc89201198)

[3.1.1 小程序端 9](#_Toc89201199)

[3.1.2 后台管理端页面 9](#_Toc89201200)

[3.2服务器端 10](#_Toc89201201)

[3.3.1 背景分析 10](#_Toc89201204)

[3.3.2 开发框架 10](#_Toc89201205)

[3.3.3 部署环境 10](#_Toc89201206)

[系统设计 10](#_Toc89201207)

[4.1 整体设计 10](#_Toc89201208)

[4.1.1 架构设计 10](#_Toc89201209)

[4.1.2 数据库系统设计 11](#_Toc89201210)

[4.1.3 功能结构设计 12](#_Toc89201211)

[4.2 模块设计 12](#_Toc89201212)

[4.2.1 用户模块 12](#_Toc89201216)

[4.2.2猫咪模块 13](#_Toc89201217)

[4.2.3 评论模块 16](#_Toc89201218)

[4.2.4 科普模块 17](#_Toc89201219)

[4.2.5 通知模块 17](#_Toc89201220)

[5.2 测试环境 18](#_Toc89201221)

[5.4 性能测试 19](#_Toc89201222)

[5.4.1 测试目标 19](#_Toc89201223)

[5.4.2 测试方法 20](#_Toc89201224)

[5.4.3 测试结果与分析 20](#_Toc89201225)

[应用运营 22](#_Toc89201226)

# 项目介绍

## 1.1 项目背景

近年来，城市流浪猫咪过剩的问题引发越来越多的关注，人性化地处理过剩猫、科学处理人猫关系 是我们的共同目标。

另一方面，萌宠类的内容也存在较大的吸引力。借助萌宠类的内容的吸引力，能够进一步促进校园流浪动物救助工作的进行。

因此，猫在山大程序以猫谱、用户发表猫咪图文为主要功能，在帮助提升猫咪信息记录传递效率的同时，以满足用户的云撸猫需求为核心价值，为山大动保增加曝光量，推动校内流浪动物保护工作的持续运转。

# 产品设计

根据前期的需求调研及市场分析，猫在山大小程序管理和用户两侧分别梳理了产品的功能点，如下图所示：



## 2.1 找猫

小程序的找猫版块下有分为：猫谱、领养、科普三个小版块。分别对应云撸猫用户、潜在领养人的找猫 需求以及工作人员希望用户们在“找猫”过程中掌握猫咪知识的需求。由于云撸猫为小程序的核心功 能，因此猫谱版块设置在“找猫”的第一位，作为小程序的落地页面。

### 猫谱

小程序的落地页需要抓住大部分目标用户的眼球，让用户有继续探索使用的欲望。而猫咪信息中最能吸 引用户兴趣的部分就是猫咪的图、性格、姓名。而由于山大异地多校区的特点，标识校区也能提高用户 的浏览效率。

因此采用卡片式展示猫咪信息，卡片结构：猫咪主图(上)+猫咪姓名、所在校区、性格特点。双列卡片

+一屏8猫的展示，是为了兼顾展示每一只猫和展示更多的猫(展示更多的猫才能提高大家了解每一只猫的 可能)。

针对不同的场景，小程序提供了不同找猫的方法：

清楚猫的姓名(此时猫多为明星猫咪)：姓名搜索

不清楚猫的姓名，但知道所在地/花色/在校状态：使用筛选器(校区、花色、猫咪状态)定位

### 领养版块

校外的潜在领养人们，对校园流浪猫咪的关注点与在校学生们截然不同，小程序内也设置了“领养”版

块，帮助他们更快速地了解校园内待领养的猫咪们。由于待领养的猫咪数量并不多，潜在领养人对猫咪 信息的了解需求也不同，因此该页面也提供了与猫谱界面不同的信息呈现方式和功能点：

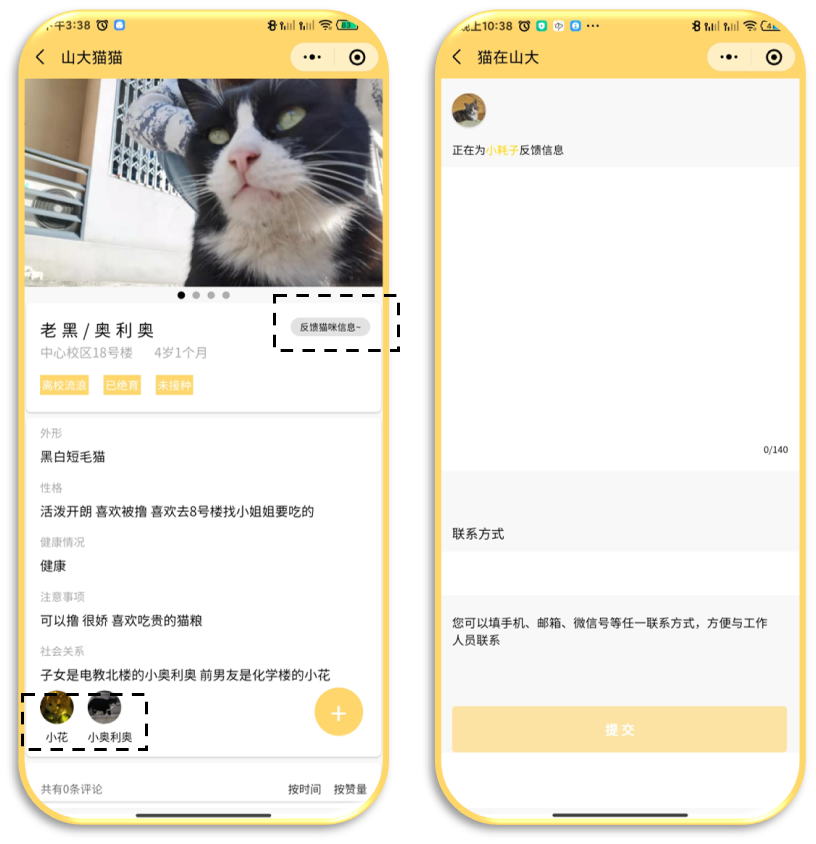
更大的卡片展示结构、展示不同的信息：卡片展示潜在领养者更关心的信息，图片、姓名、年龄、 绝育状态、疫苗接种状态、额外的领养说明

不设置筛选、查找功能：待领养的猫咪不多，并无提升浏览效率的需要

底部设置了跳转引导订阅公众号的卡片：领养流程繁琐，更完备的领养信息以公众号推文的方式呈 现更加高效。

### 猫咪详情页

猫谱、猫咪详情页中，每只猫咪的信息卡片都对应着一个猫咪详情页面。点击猫咪卡片即进入对应猫咪 的详情页中。



顶部设置猫咪图片：左右滑动切换

中间展示重要的基本信息：姓名、地点、年龄、绝育状态、疫苗接种状态、在校状态

第一屏底部展示剩余的猫咪信息：外形、性格、健康状态、注意事项、社会关系等。这部分信息能 增进用户对猫咪的了解，让猫咪的形象更立体有趣，但由于统计难度较大，通常缺漏。社会关系除 了文字描述外，也展示对应猫咪的头像，点击直接跳转到对应的猫咪详情页中

猫咪详情页设置了反馈猫咪信息的功能，用户可以对详情页中出错的猫咪信息进行纠正。功能界面 左上方设置猫咪缩略图、猫咪姓名，提示用户是为哪一只猫报送错误信息。

右下角为猫咪详情页的评论投稿功能，具体介绍见2.2.2

### 科普

分5个版块展示猫咪知识：TNR、猫咪疾病、猫咪领养、撸猫、猫咪喂食。

每个版块分为若干小知识点。点击进入对应的版块中后，可通过滑动顶部的版块封面图切换到其它 版块下。

点击下方对应的知识点滑块获取对应知识点的详细介绍

## 2.3 社区

社区版块分为两部分：猫友圈、山猫速报。

猫友圈的设置是为了满足用户分享、消费猫咪图文的需求。核心功能为评论投稿、消费猫咪图文。 但大部分用户的猫咪图文会集中于校内明星猫咪上，会忽视其它大部分的普通猫咪。因此，为平衡 这种必然出现的“不公平”现象，小程序在社区版块设置了山猫速报，通过运营人员的干预，为更多 的猫咪获得更多的曝光。同时，重要事件(如：猫咪需要紧急救援、临时寄养等)的报

道、后续的反馈等，也可以通过速报展示给用户。



### 山猫速报

设置这一版块是为了在小程序中增加产品运营人员与用户之间的沟通途径，对校园内发生的与猫咪相关 的“新闻”推给用户。

每一条速报均可选择配置一个链接（包含小程序内部的页面路径及关联公众号的推文链接） 例：新猫上架——秋井。点击右方的详情即可跳转“秋井”的详情页。

例：冬日为猫做屋活动正式开启。点击右方的详情即可跳转活动推文界面。

### 猫友圈

用户可在社区的猫友圈中评论投稿猫咪图文故事。同时，猫咪详情页下方也设置评论投稿猫咪图文故事 的功能。

两处分别设置评论功能是为了服务于不同的场景：

猫咪详情页中的投稿，基于用户对相应猫咪的详情页进行了浏览，通常用户已经对猫咪有了较多的 了解。

社区猫友圈中的投稿，通常用户希望直接分享猫咪故事，不一定建立在浏览猫咪详情页的行为后(包 括：熟悉猫但目的是分享故事、不清楚是哪只猫、猫咪尚未被登记)

### 评论的写操作逻辑

出于内容风险把控需要，投稿须后台审核通过后才进行展示，界面上的使用输入框暗提示进行说 明。

用户可仅输入文字，或仅上传图片投稿。

猫咪详情页设置了“同步到社区”按钮，用户可自行决定是否同步到小程序的社区版块以获得更大的内容曝光量。

### 评论的展示逻辑

信息流中单条内容的基本结构：发表人头像+昵称+发表时间+点赞按钮+发表内容（上文下图，文字仅展示前两行，详细内容需点进投稿详情页中查看）。社区中的某条内容若是从猫咪详情页中同步 而来，内容文字部分上方还会有猫咪tag。

投稿详情页：完整展示全部文字内容，若内容在猫咪详情页中投稿而来，则底部会展示对应猫咪的 头像，点击跳转猫咪详情页。来稿内容分两层展示，也是为下一步迭代增加“用户评论他人评论”的 功能（下称楼中楼）。

### 评论的互动功能

目前仅支持点赞功能，后续会增加楼中楼。

## 2.3 我的

我的版块分为两大部分，顶部为小程序内部互动的记录：如发表、赞过、消息；底部为其它独立的零散 功能，包括：报送新猫、打赏公示、关于我们。



### 报送新猫

填写项目包括：校区、目击地点、目击时间、猫咪详情、图片。校区采用单选的形式录入，其余信息均通过用户输入

除图片外均为必填项，若有漏填，前端会弹出提要求补充必填信息

用户提交“报送”信息成功后，后台收到报送表格，在审核通过并成功录入信息后，后台人员在报送 表单界面输入录入的新猫的编号，用户将会在消息中收到一条审核通过的通知，点击即可进入新猫 的详情页。

### 通知

评论互动反馈

目前仅包含点赞通知，后续楼中楼迭代后，会增加评论通知报送新猫结果反馈

2.3.1中已介绍（如下图所示）



评论审核结果反馈

后台工作人员审核用户评论后，若评论通过审核，将会收到审核通过的通知评论的官方回复功能

用户的评论通常带有了解猫的目的，如下图所示。为了满足用户的该目的，除了未来即将上线的

“用户评论来稿内容”功能外，我们额外设置了：后台审核人员向用户发送猫咪信息的功能。

该功能的具体逻辑为：后台审核人员可在评论审核界面进行操作，通过输入猫咪编号，将评论绑定 到具体的猫咪上，此时，该评论会同步到对应猫咪的详情页中，并自动发送通知给用户，用户仅需 点击此通知即可进入猫咪详情页。通知内容为：“亲爱的猫友，您于yyyy-mm-dd发表的评论提及了这只猫咪，快点击进入猫咪详情页认识它吧~”

后续将会提升通知版块的通知强度，待获取相应的服务权限后，通过服务信息的方式推送给用户，以提 升通知的用户体验及反馈效率。

### 发表与赞过

用户可通过发表界面查看自己发表过的全部评论，通过赞过查看自己点赞过的全部评论（相当于收 藏）。

# 技术方案

## 3.1 前端

### 小程序端

小程序端主要使用原生WXSS、WXS、WSML等小程序技术。

WXML（WeiXin Markup Language）是框架设计的一套标签语言，结合基础以构建出页面的结构。

WXSS (WeiXin Style Sheets)是一套样式语言，用于描述 WXML 的组件样式，可

WXS（WeiXin Script）是小程序的一套脚本语言，结合 WXML，可以构建出页面的结构。

### 后台管理端页面

Vue是一套用于构建用户界面的**渐进式框架，**可以自底向上逐层应用。Vue 的核心库只关注视图层，不

仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化工具链以及各种类库使用时，Vue 也完全能够为复杂的单页应用提供驱动。

Element UI 是一套采用 Vue 作为基础框架实现的组件库。结

因后台管理端页面对开发时间要求较高，因此使用VUE+element ui开发，使网站迅速成型，通过管理端可以修改猫咪信息，审核评论，发布速报，配置科普模块。

## 3.2服务器端



### 背景分析

猫在山大小程序主要面向群体为在校学生，为学生提供云吸猫，猫社区功能。受面向群体限制，当 下小程序请求并发量处于易处理范围之内，同时需考虑后期用户量增多带来的问题。且小程序完全 由学生维护，需考虑经费限制。根据项目所处的特殊背景，服务器端需制定一条满足**敏捷性、健壮 性、可扩展且低成本的技术方案**。

### 开发框架

Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。SpringBoot基于Spring4.0设计，继承了Spring框架原有的优秀特性，而且还通过简化配置来 进一步简化了整个Spring应用的搭建和开发过程。通过Spring Boot框架可以减轻服务端复杂的工作量和部署流程。

### 部署环境

通过背景分析，本项目采用多节点负载均衡架构。

前期用户量较少时，采用三台1核2G平价学生机部署，使用Nginx负载均衡，节省服务器成本。若后续用户量增多，可水平增加服务器，扩充方式简单。

**主要技术选型**如下：

**Nginx**是一款高性能、轻量级的Web服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，在BSD-like 协议下发行。其特点是占有内存少，并发能力强。通过Nginx反向代理和负载均衡，可以将用户请求均匀分配在三台服务器上，同时避免了单台服务器宕机导致的接口关闭，基于 此提供快速获取重要数据，解决大量并发访问服务问题，提高了系统的健壮性。同时若后续水平增 加服务器，可通过Nginx反向代理将请求分配到新服务器，提高了系统的可扩展性。

**Redis**是一个支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API。使用Redis缓存热点数据，可有效提升接口的性能和响应速度，提高系统的敏捷性和高并发处理能力。同时可以采用Redis作为分布式锁，实现分布式事务。

**RocketMQ**是一款分布式、队列模型的消息中间件，具有比较友好的分布式特性，且历经多次双十 一海量消息考验。使用RocketMQ不仅可以通过发布-订阅的工作模式解耦服务间调用，而且可以通过异步处理提高系统敏捷性和削平峰值流量。

**CDN内容分发网络**是构建在现有网络基础之上的智能虚拟网络，依靠部署在各地的边缘服务器，通 过中心平台的负载均衡、内容分发、调度等功能模块，使用户就近获取所需内容，降低网络拥塞， 提高用户访问响应速度和命中率。因本项目是一个以图片为主导的小程序，系统包含大量图片，采 用CDN可以加速图片访问速度，提升用户体验。

# 系统设计

## 4.1 整体设计

### 架构设计

本系统采用分层架构设计，分为数据层、支持层、服务层、业务层、网关层和展示层。分层设计可以分 散关注、松散耦合、逻辑复用、标准定义。

**数据层**：负责数据库的访问和数据储存。采用MySQL数据库作为系统主存储数据库，采用Redis数据库缓存热点数据提高接口性能和响应速度。

**支持层**：为项目开发提供支持。采用Spring Boot框架构建项目，采用RESTful风格接口，使用Yapi

工具构建接口文档。

**服务层**：包括系统所需的一些中间件服务和公共基础服务。采用Grafana监控系统状态并设置故障提醒，使用RocketMQ消息队列解耦服务间调用，异步处理提高系统敏捷性和削平峰值流量。

**业务层**：包含系统主要服务模块。主要有猫咪模块、评论模块、用户模块、科普模块和通知模块。 **网关层**：所有接口都通过网关层接入系统。通过Nginx实现负载均衡和路由转发；隐藏内网服务， 增强安全性；实现用户鉴权。

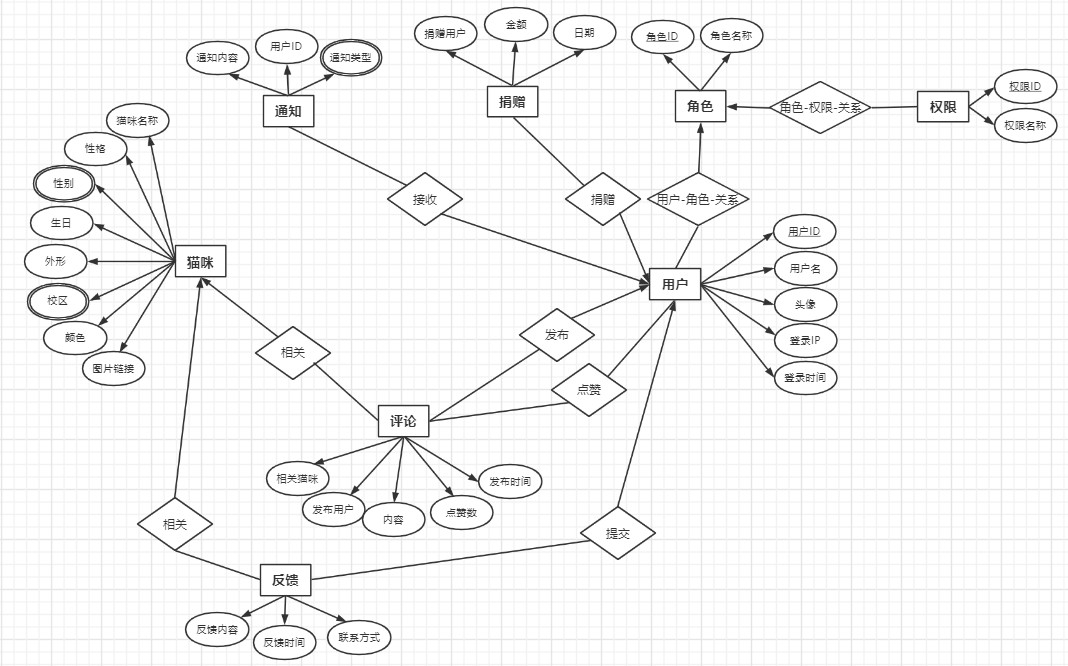
**展示层**：系统的界面展示。主要包括小程序端界面、后台管理网页和系统监控平台。

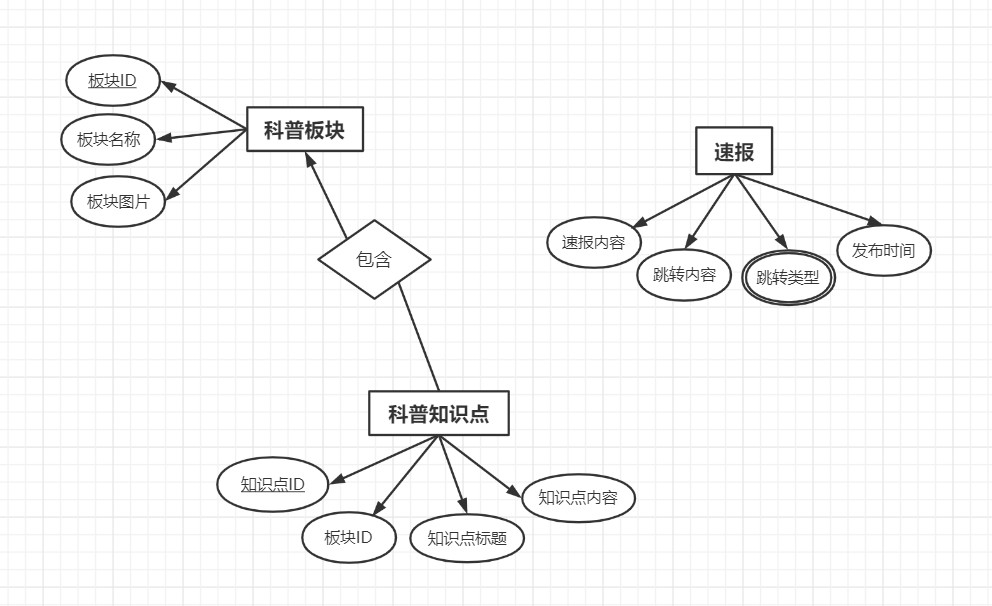
系统架构图如下：



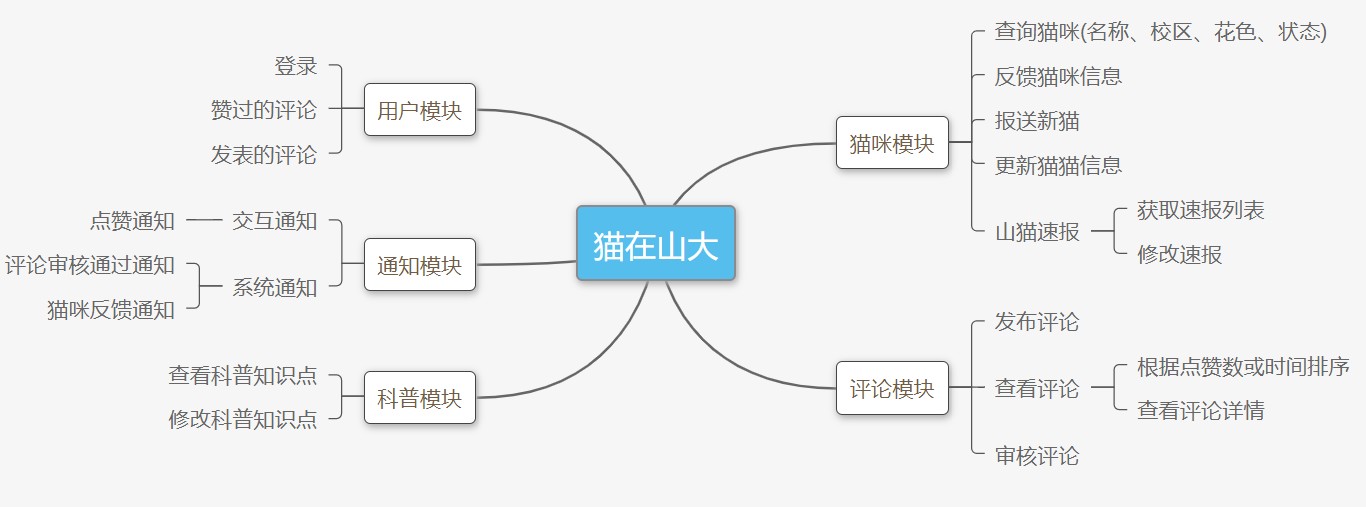
### 数据库系统设计

本系统E-R图共有11个实体，分别是用户、角色、权限、猫咪、评论、通知、反馈、捐赠、科普板块、科普知识点和速报。

同时该E-R图还描述了实体之间的关系，例如用户和评论有一对多关系，用户可以发布多条评论，一个评论只能由一个用户发布，同时用户和评论也是多对多关系，用户可以点赞多条评论，一条评论也可以由 多个用户点赞；角色和用户是一对多关系，一个角色可以对应多个用户。



### 功能结构设计

系统的功能设计按模块来划分，主要分为**用户模块、猫咪模块、评论模块、科普模块和通知模块**。

## 4.2 模块设计



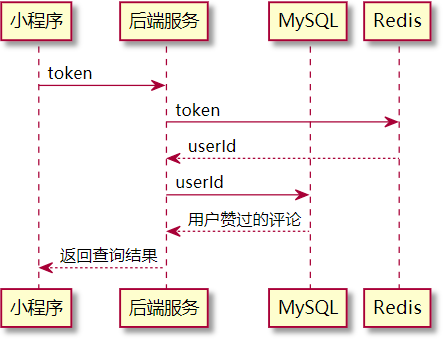
### 4.2.1 用户模块

用户模块包含用户鉴权、登录、查看赞过的评论、查看发表的评论四个主要功能。用户鉴权采用Spring Security在网关层进行过滤，通过查询用户对应的角色及其对应的权限接口，来检验用户对于接口的权限。登录采用微信平台提供的接口，若用户授权登录，则可用通过wx.login()获取到code，根据此code 服务端可以获取到相关的用户信息。同时用户可以在个人主页里查看点赞过的评论和发表的评论。

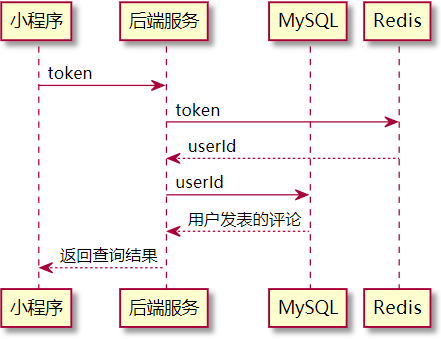
登录



#### 查看用户赞过的评论



#### 查看用户发表的评论



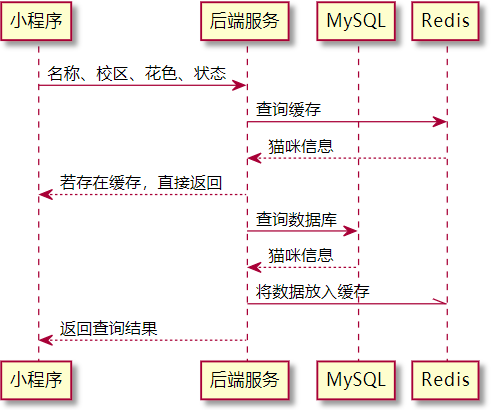
### 4.2.2猫咪模块

猫咪模块主要用于猫咪信息的查询和管理，是系统的重点模块。该模块主要包含查询猫咪，反馈猫咪信 息，报送新猫，更新猫咪信息，获取速报，修改速报等功能。用户可以通过猫咪名称模糊搜索，并利用 校区、花色、状态多个条件进行筛选。管理端可以检测到用户反馈和报送新猫信息，经审核后会发送给 用户通知。

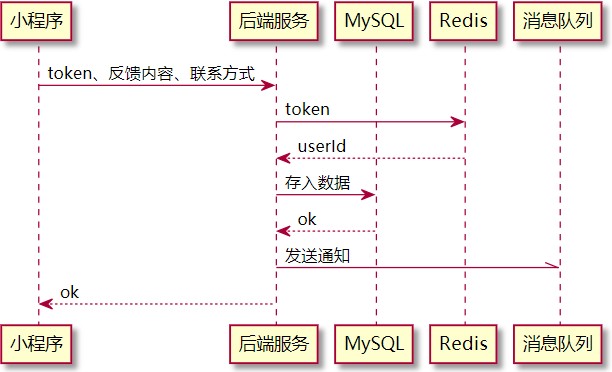
因该模块内猫咪数据不易变动，且用户最常访问，所以对此模块的猫咪数据和速报数据利用Redis进行缓存，提升接口响应速度，提高系统并发性。使用缓存易造成数据不一致现象，本系统采用先改后删缓

存，附加延时双删策略，避免数据库中数据和缓存信息不一致。

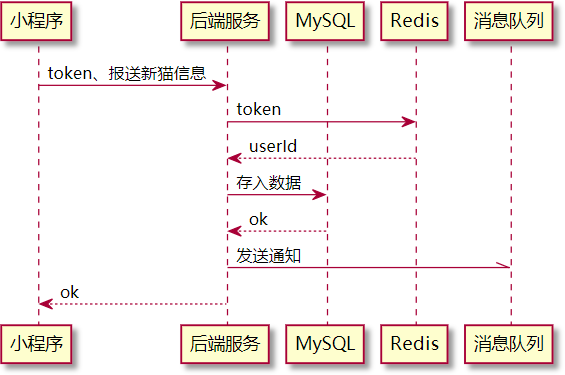
#### 查询猫咪



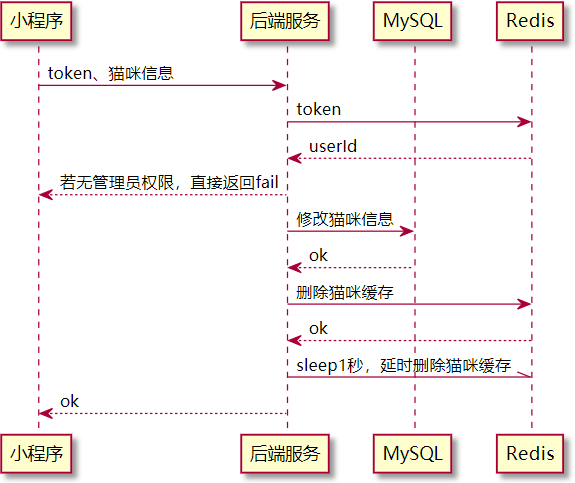
**反馈猫咪信息**



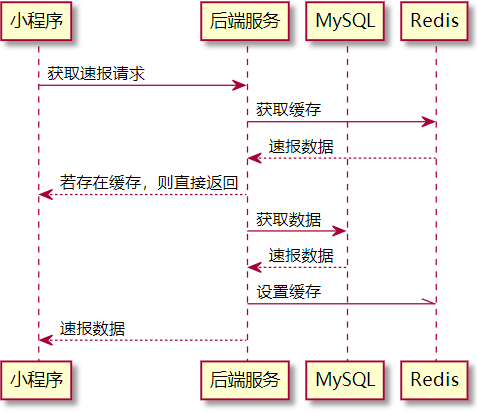
**报送新猫**



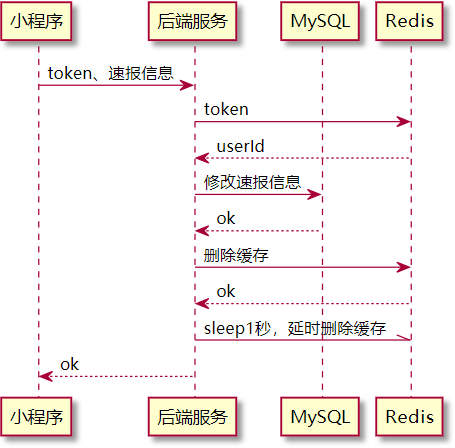
**更新猫咪信息**



**获取速报**



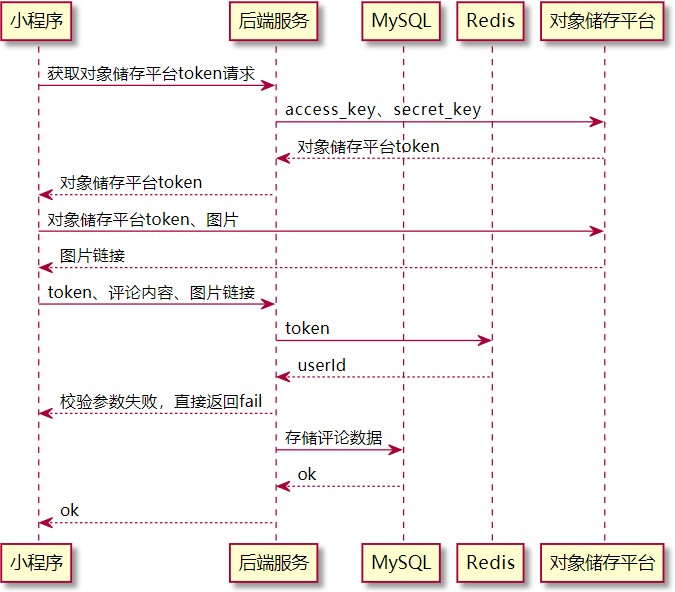
**修改速报**



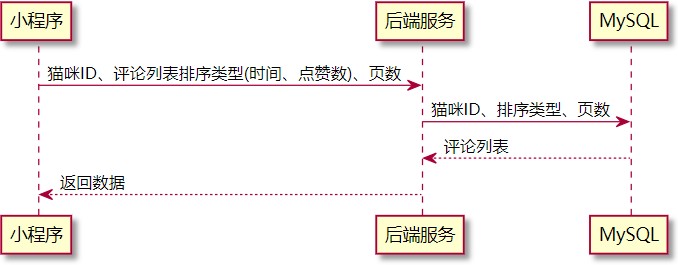
### 4.2.3 评论模块

评论模块用于评论的管理。包括用户发布评论、查看评论及管理端审核评论等功能。用户发布评论时若 含有图片，小程序端会首先将图片上传至对象储存平台，之后才会将评论内容及图片链接传递给服务 端，从而有效避免带宽阻塞。同时，用户可以根据发布时间和点赞数两种不同排序方式来获取评论。评 论审核后，服务端会向消息队列发送通知消息，通知用户该次审核结果。

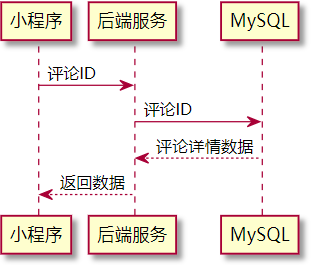
#### 发布评论



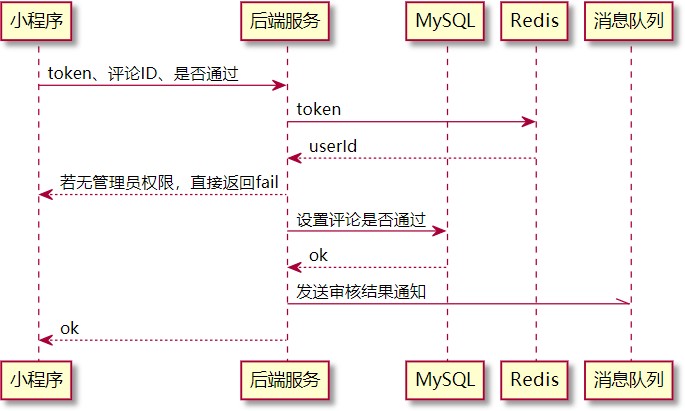
**获取评论**



**获取评论详情**



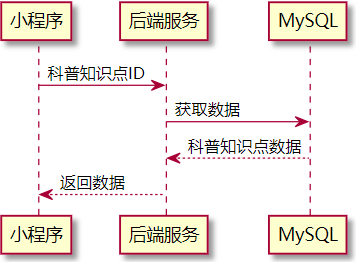
**审核评论**



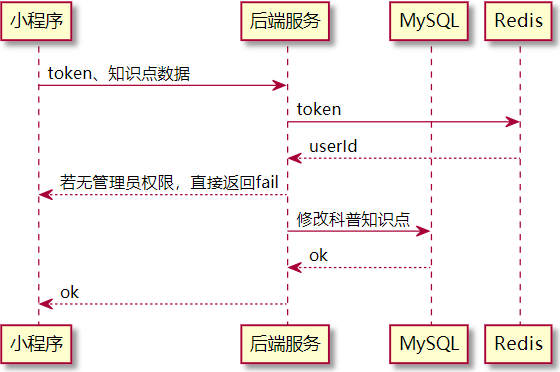
### 4.2.4 科普模块

科普模块主要用于科普知识的管理。系统中有多个科普知识点，这些知识点分别属于不同的板块，用户 可以通过滑动来切换不同知识点。

#### 获取科普知识点信息



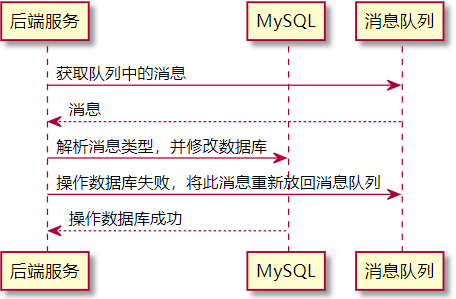
**修改科普知识点**



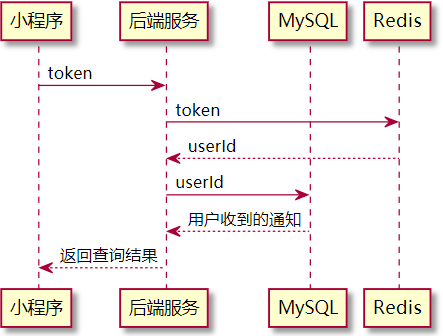
### 4.2.5 通知模块

通知模块主要负责用户的通知。通知分为互动通知和系统通知，互动通知主要有点赞通知和回复通知， 系统通知则是一些审核结果，如评论审核，反馈审核，新猫报送审核等。通知功能利用RocketMQ消息队列来实现，若审核结束，系统会将通知消息异步发送给消息队列，同时后台有一个进程不断读取消息 队列，并处理通知消息，从而解耦系统服务，提高了接口响应速度和系统并发性。

#### 发送通知

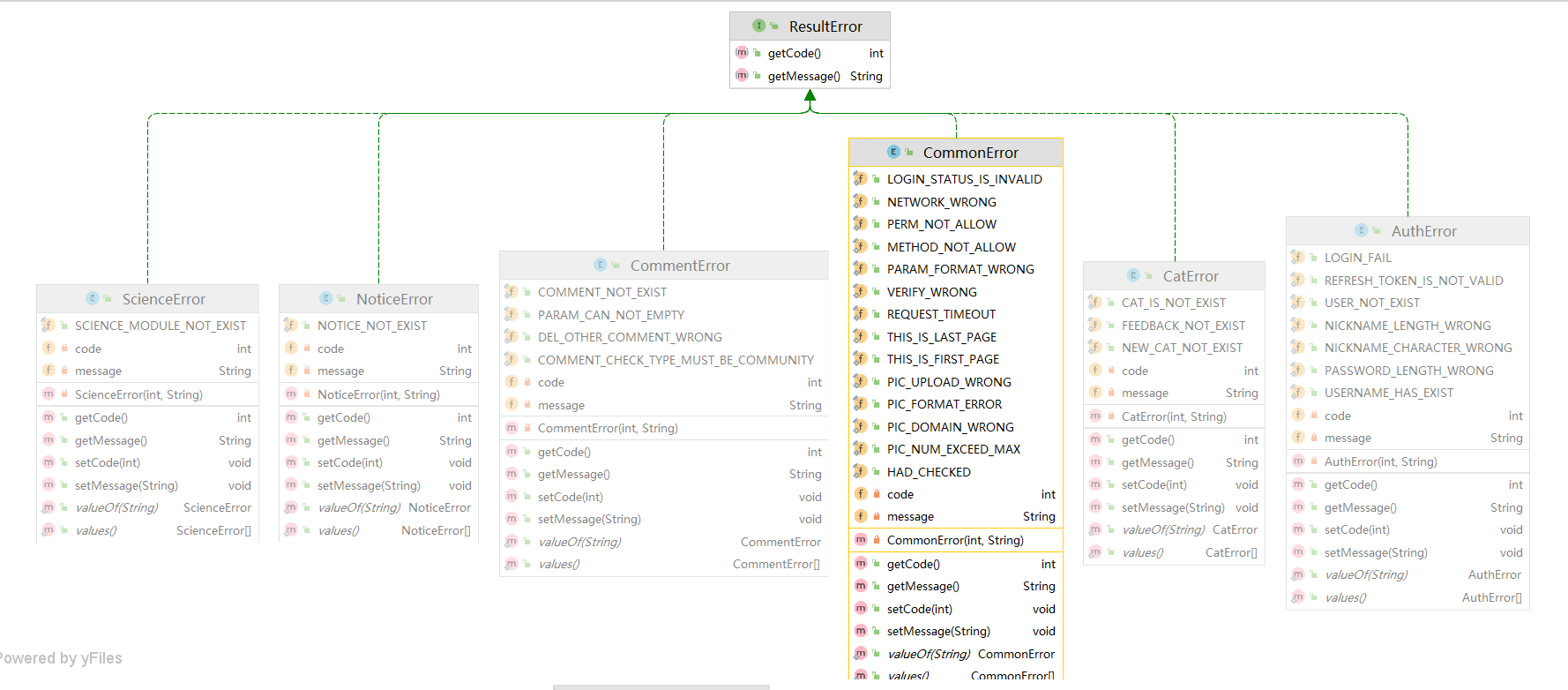


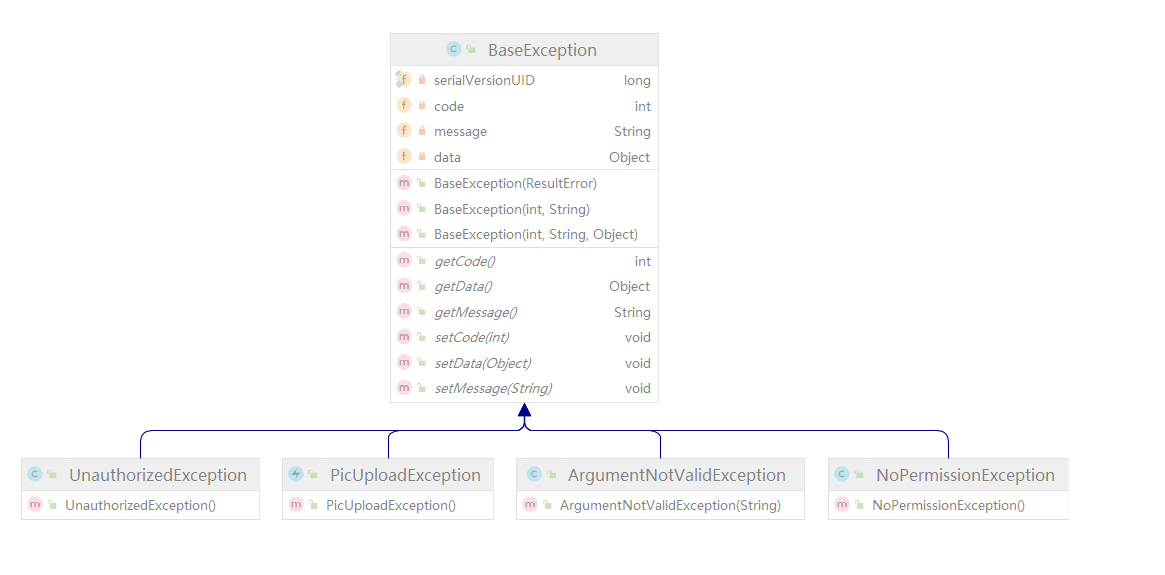
**用户获取通知**



## 4.3 系统类图

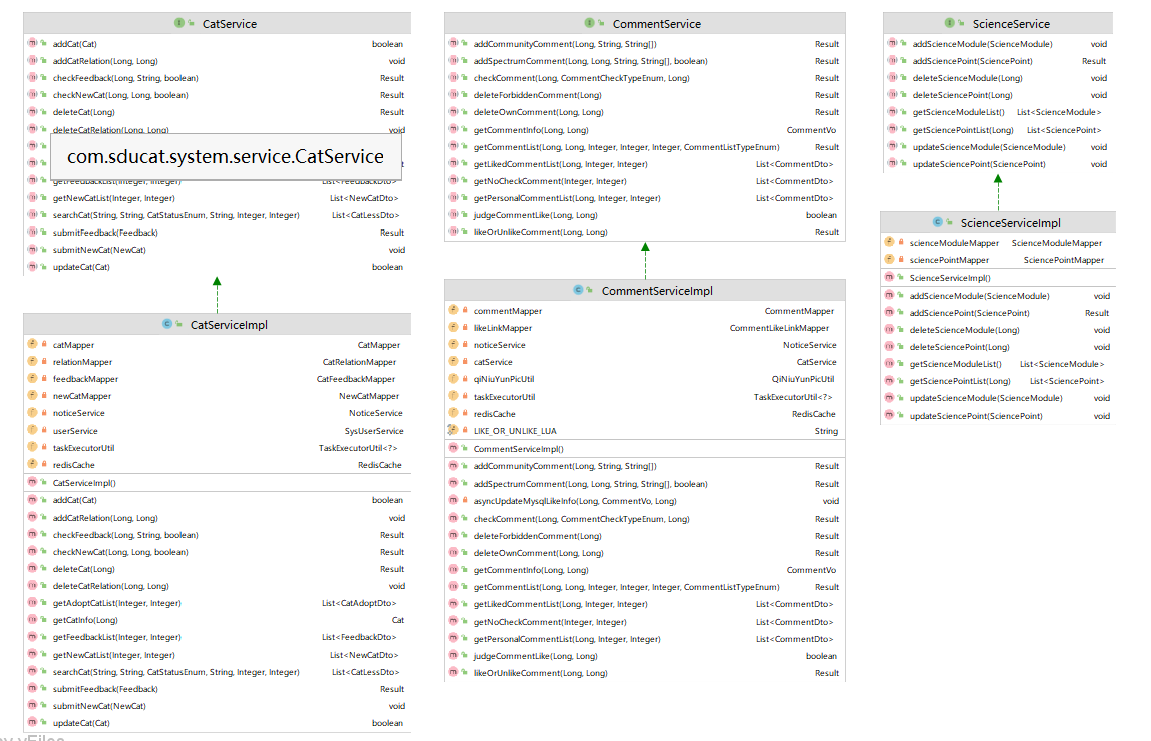
### 4.3.1 错误类图

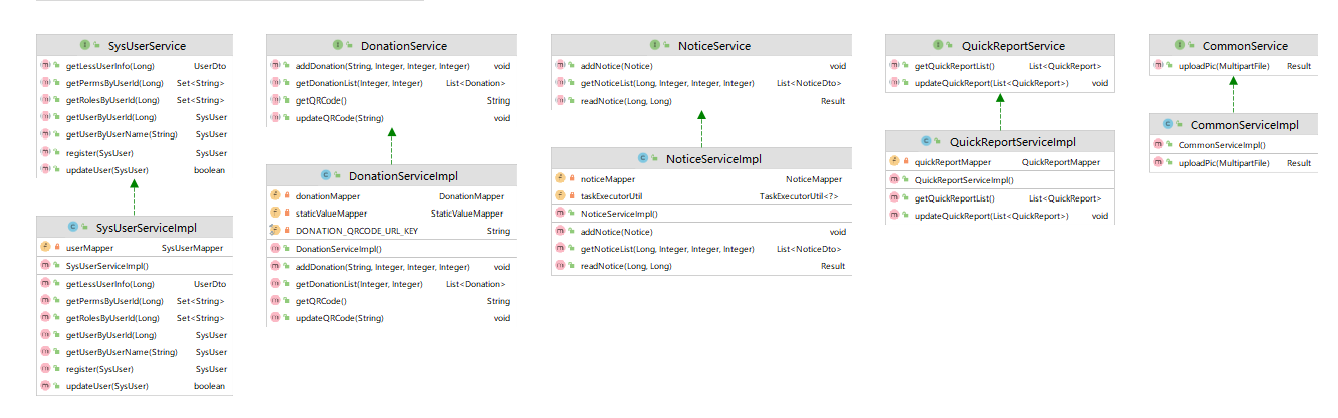




建立统一的Error以及Exception基类，所有的系统层及错误业务层错误继承错误基类，对项目的错误进行统一拦截

### 4.3.2 Service层





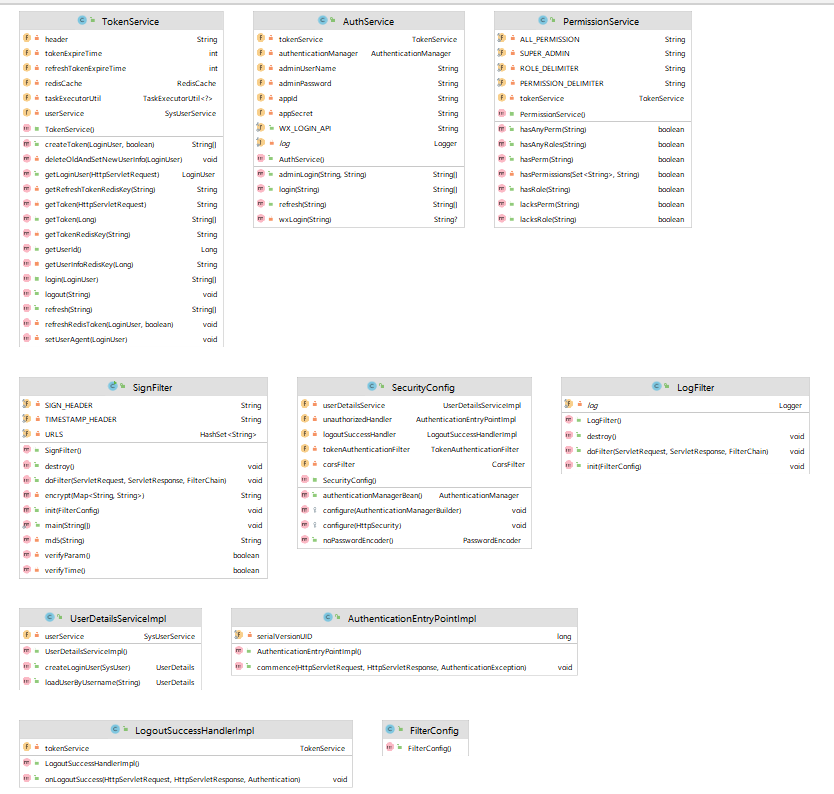
所有的Service实现继承自Service的抽象类，便于版本更迭，或者service有大型变化时采用新的Service的具体实现，controller层调用service层的方法而不需要具体实现

### 4.3.3 Notice



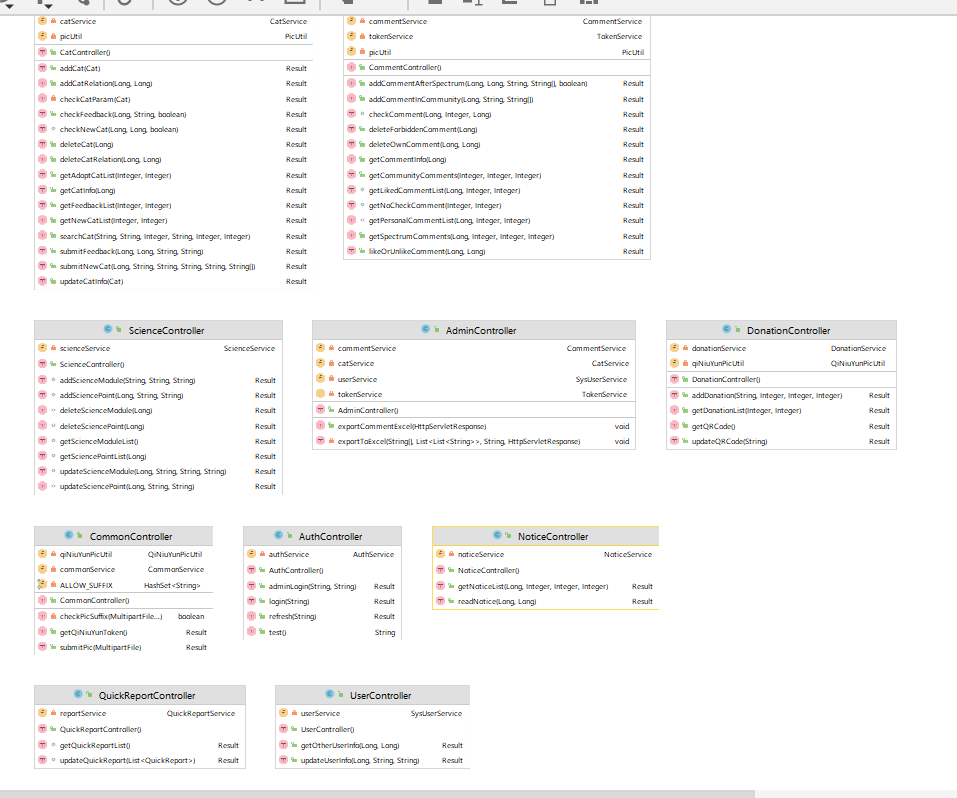
Notice继承自AbstractNotice，是Notice的继承类，所有的子类Notice实现Abstrcat的方法，实现noticeType的获取，内容的获取，以及title的获取。

### 4.3.3 SercurityConfig



利用SpringSecurity进行身份鉴权，鉴权成功颁发JWT，对于携带JWT的鉴权，会进行拦截，首先进行对JWT的鉴权，鉴权成功之后不会走spring的鉴权流程，直接设置鉴权成功的flag。对于未携带Token的鉴权，会进入到UserLoginService进行UserDetail鉴权，自定鉴权逻辑，利用hash散列加密，保证密码的安全性。

### 4.3.4 Controller层



Controller层负责处理具体的业务逻辑，通过controller层建立与用户的接口，用户只需要调用api即可于controller层交互，前端需要调用接口进一步访问。

**系统测试**

**5.1 测试概述**

本次测试主要分为**功能测试**和**性能测试**。

功能测试就是对产品的各功能进行验证，根据功能测试用例，逐项测试，检查产品是否达到用户要 求的功能。

在确保功能实现正确的前提下，通过合适的性能测试加压方式和策略，收集考察服务端应用程序的 各项性能指标，以及服务器硬件资源的使用情况，来评估是否存在性能问题隐患。

## 5.2 测试环境

#### 服务端测试环境

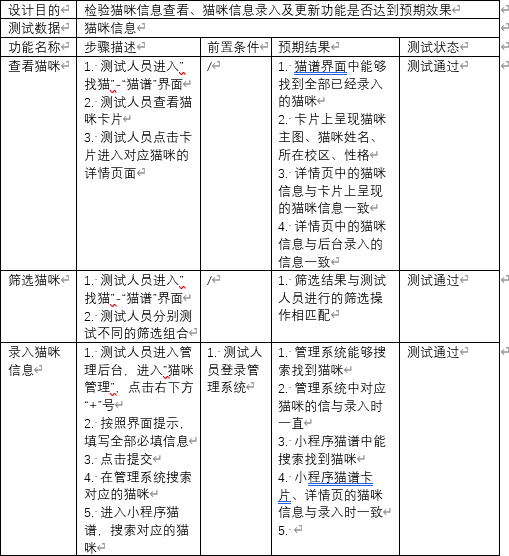
|  |  |
| --- | --- |
| **操作系统** | **Ubuntu 20.04** |
| 内存 | 2G \* 2 + 4G \* 1 |
| 核数 | 1C \* 2 + 2C \* 1 |
| 处理器 | Intel(R) Xeon(R) Gold 6133 CPU |
| 数据库 | MySQL 8.0.23、Redis 5.0.7 |

**客户端测试环境**

|  |  |
| --- | --- |
| **操作系统** | **Android 10** |
| 内存 | 8G |
| 核数 | 4C |
| 处理器 | Intel(R) Core(TM) i7-8750H CPU |
| 测试工具 | 微信8.0.3 |

**5.3 功能测试**

本小节以对猫咪信息管理的测试为例子阐述实际功能的系统性测试，其测试用例如下所示。



## 5.4 性能测试

### 5.4.1 测试目标

在大用户量、数据量的超负荷下，获得服务器运行时的相关数据，从而进行系统分析，找出系统瓶 颈，提高系统并发性和稳定性。总结测试阶段的测试以及分析测试结果，并根据当下用户请求量状 况和服务器承载能力来合理分配资源。

具体目标：查看猫咪详情接口200TPS，响应时间小于两秒

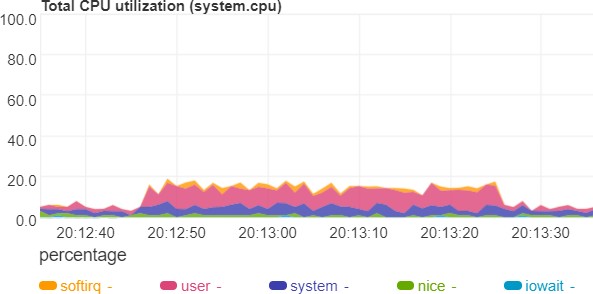
### 测试方法

本次测试采用apache开源的测试工具jmeter，并利用Granfana进行服务器性能监控

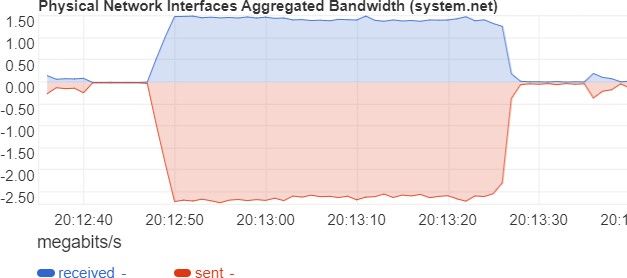
### 测试结果与分析

#### 50并发

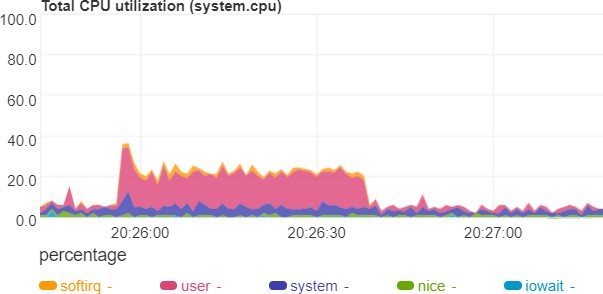
**CPU负载统计**



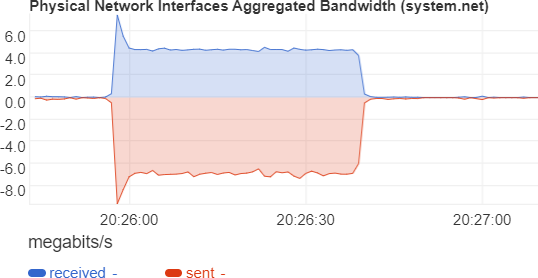
**网络负载统计**



**100 并 发 CPU负载统计**

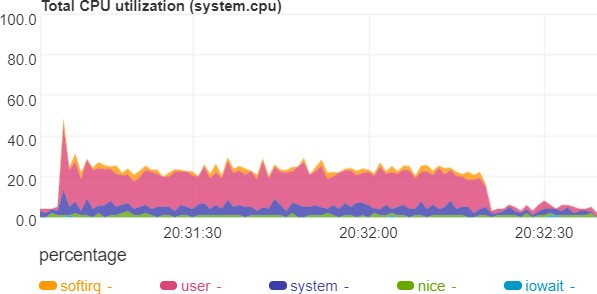


**网络负载统计**

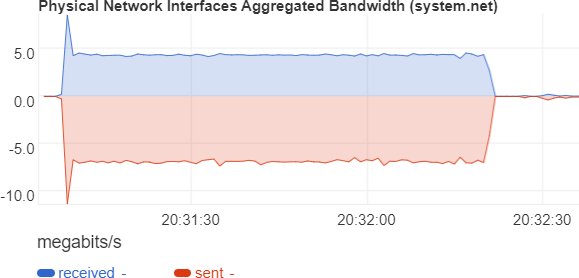


200并发

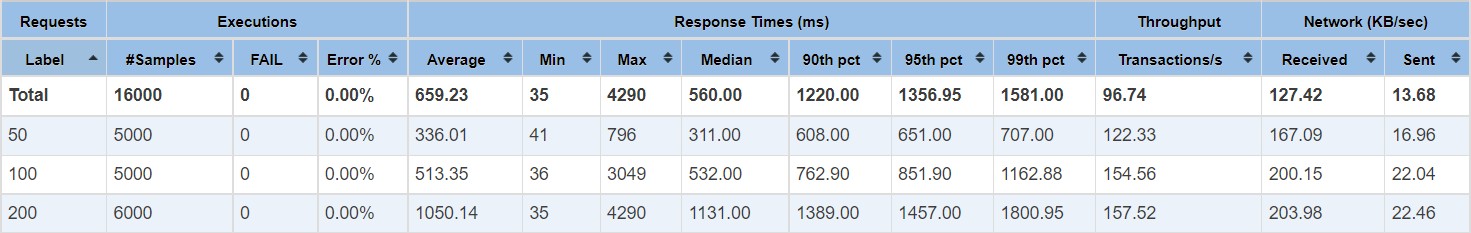
#### CPU负载统计



**网络负载统计**



**请求统计**



**结果分析**

通过Jmeter压测可以得出，在50并发下，接口99%响应时间为700ms，CPU负载小于20%；在100并发下，接口99%响应时间为1162ms，CPU负载为20%左右；在200并发下，接口99%响应时间为1800ms，CPU负载为25%左右；

由此可得出结论，本系统可支撑200并发下的接口响应时间在2秒以内，用户体验良好。在运营前期，当前完全足以支撑用户访问。

# 应用运营

* 1. **日常运营**

审核用户来稿：为提供较好的用户体验，项目组成员轮班审核后台收到的用户来稿，尽量缩短用户 从投稿到审核通过的时间

* 1. **未来计划**

小程序产品迭代：新增“提问”形式发表的来稿，虽然后台可以通过通知方式回答用户的问题，但远不及直接在小程序中评论来稿方便，配合适当的通知，能够提高用户使用黏性。因此下一步的主要迭代任务即增加评论来稿功能

公众号内容建设：在前期的运营中，公众号优质内容的能带来较高的新增转化，配合公众号的关注 自动回复功能，能够为小程序带来新用户。同时，公众号也是工作宣传的重要途径，与小程序是相辅相成的关系。

来稿合集：每周收集小程序中的精选来稿，在公众号中集中发出，增加用户的互动感，提高用户黏性

报送信息更新：工作人员每周审核一次报送信息，并将核对确认真实的部分进行更新，并在页面上备注报送信息的用户昵称。