

摘要

在时下的中国，各类助力农业发展的 APP 层出不穷，农民使用手机查看农业技术、掌控农产品行情、售卖农产品都已经屡见不鲜。但是，客观而言，目前国内的农业手机 APP 绝大部分侧重于传播农业信息，直接辅助农业生产管理的 APP 相对较少。

为切实辅助农业生产管理，农管 APP 软件提供“农新闻”、“农问吧”、“农商城”、“病虫害识别”四个模块，其中“农新闻”模块主要传送农业信息；“农问吧”模块搭建起农户和专家的交流平台，多位专家在线解决用户问题；“农商城”主要是售卖农业技术公司提供的土壤生态链修复等服务，公司将提供技术、药物及人员支持，同时此模块提供了“农机手”抢单功能，让农户接受公司培训学习新技术并有额外的收入；“病虫害识别”模块运用卷积神经网络的 Inception-V3 模型实现识图辨别病虫害功能，用户通过上传作物图片即可得出作物的病症及治理方案。

在提高软件性能方面，为了缓解数据库负载过大的问题，使用 Redis 对数据库中的数据进行缓存，降低数据库的负载；使用 FastDFS 集群实现图片海量上传及高并发访问功能。

关键词：Inception-V3 模型，FastDFS 集群，Redis，Spring

ABSTRACT

In China, various APPs are used to help agricultural development. Farmers have used mobile phones to view agricultural technology, control agricultural products, and sell agricultural products. However, most of the current domestic APP focus on the dissemination of agricultural information, and there are relatively few APPs directly supporting agricultural production management.

In order to effectively assist agricultural production management, the Agricultural Management APP software provides four modules: “农新闻”, “农问吧”, “农商城” and “病虫害识别”. The “农新闻” mainly transmits agricultural information; the “农问吧” builds an exchange platform for farmers and experts, and many experts solve doubts online; “” mainly sells soil ecological chain restoration provided by agricultural technology companies, etc. Services, the company will provide technical, pharmaceutical and personnel support. At the same time, this module provides the “农机手” rob order function, allowing farmers to receive corporate training to learn new technologies and have additional income; “病虫害识别” uses a Inception-V3 model To realize the function of identifying and mapping pests and diseases, users can obtain crop diseases by uploading crop images.

To improve software performance, in order to alleviate the problem of excessive database load, a Radis cluster was built to cache data in the database and reduce the load on the database. FastDFS clusters were used to implement massive upload of images and high concurrent access.

Key words: Inception-V3 model, FastDFS Cluster, Radis, Spring

目录

1 绪论.....	1
1.1 课题研究的背景及意义.....	1
1.2 国内外研究现状.....	1
1.3 课题研究内容.....	2
1.4 内容的构成与章节安排.....	3
2 相关技术.....	4
2.1 农管 APP 开发平台.....	4
2.1.1 APICloud.....	4
2.1.2 Maven.....	4
2.2 农管 APP 相关技术.....	5
2.2.1 FastDFS 图片服务器集群.....	5
2.2.2 Redis.....	8
2.2.3 Inception-V3 模型.....	8
2.2.4 MVC 设计模式.....	10
2.2.5 SSM.....	10
2.2.6 EasyUI 界面插件集合.....	11
2.2.7 UEditor 富文本.....	11
3 需求分析.....	12
3.1 功能需求.....	12
3.1.1 系统角色功能描述.....	12
3.1.2 系统模块功能描述.....	17
3.2 非功能性需求.....	18

4 概要设计.....	19
4.1 系统功能架构设计.....	19
4.2 系统技术架构设计.....	20
4.3 子系统设计.....	22
4.3.1 APP 端.....	22
4.3.2 后台管理系统.....	28
5 数据设计.....	32
5.1 数据库环境说明.....	32
5.2 数据字典.....	32
6 系统实现.....	39
6.1 系统环境搭建.....	39
6.1.1 图片服务器集群构建.....	39
6.1.2 SSM+Redis 整合.....	44
6.2 病虫害识别模型.....	46
6.2.1 卷积神经网络迁移学习.....	46
6.2.2 模型训练.....	47
6.2.3 模型使用.....	48
6.3 重要功能模块实现.....	50
6.3.1 农商城模块.....	50
6.2.2 病虫害识别模块.....	55
6.2.3 农新闻模块.....	57
6.2.4 农问吧模块.....	59
6.2.5 登录注册模块.....	62
6.2.6 权限验证模块.....	65

7 系统测试	68
7.1 测试概述	68
7.1.1 测试目标	68
7.1.2 测试方法	68
7.2 测试用例	68
7.3 测试结果	72
结论	74
参考文献	75
致谢	77

1 绪论

1.1 课题研究的背景及意义

农业的发展是国家经济发展的基础条件，但是我国目前的农业主要还是传统的粗放式农业生产，农业发展水平处于初级阶段，因此推动我国农业由传统农业转化为现代化农业是当前农业发展的主要任务，而农业信息化是加快农业现代化的重要支撑^[1]。“十三五”时期农业的发展方向也确认为农业信息化。发展农业信息化的首要任务是找到信息的传播媒介。

随着我国互联网通信环境的不断完善，智能手机、iPad、智能电脑等移动终端设备快速普及，移动互联终端的应用程序层出不穷，深入渗透到用户生活需求的方方面面，类型丰富，功能强大。手机 APP 凭借其明显的优势，成为当下互联网发展的流行趋势之一。据中国产业信息网《2016 年中国移动 app 市场发展现状分析》显示，截至 2016 年 4 月，我国市场中的移动应用累计数量达到 671 万款，我国本土第三方应用商店移动应用累计数量超过 500 万款，苹果商店移动应用累计数量超过 163 万款，而尚未官方进入我国市场的谷歌商店移动应用累计数量约 260 万款，中文移动应用累计数量超过 710 万款^[2]。

当前，我国正处于农业信息化发展的关键时期，APP 与农业技术的结合将极大地优化农业技术推广手段和农业技术获取方式，提高农业技术应用效果，因此，对农技 APP 深入开展研究具有积极的现实意义。

1.2 国内外研究现状

（一）国内研究现状

“十三五”指出发展农业信息化之后，许多农业信息化公司看到了发展前景，开发出了许多适用于农户管理作物的农业 APP，虽然各式各样的

农业 APP 不断涌现，但据粗略估计，当前国内市场上农业类 APP 仅有 200 余个，且下载量很低，生命周期短，很多软件无法维持版本更新最终被淘汰，软件的侧重点大部分放在农业技术的传播，真正实际应用于管理农业生产活动的 APP 数量极少。

（二） 国外研究现状

相较于国内农业 APP 软件功能大多侧重于信息传播，国外的农业 APP 软件则直接应用于农生产领域、辅助农业生产管理，比如：BRANDT Tank 专业版允许种植者选择他们计划使用的除草剂配方以及他们希望添加的布兰德智能系统微量营养素配方，然后这款应用软件会告诉用户这些配方是否兼容，喷雾器校准计算器（Sprayer Calibration Calculator）帮助农民校准农药喷雾器等^[3]。这也为我国农业 APP 软件的开发、应用和发展提供了参考和借鉴。

1.3 课题研究内容

本论文使用 SSM（SpringMVC+Spring+Mybatis），机器学习等技术，提供农问吧、农商城、病虫害识别等功能点。本课题的主要研究内容包括以下几个方面：

- （1）根据农管 APP 软件的需求，进行系统的需求分析以及功能设计。
- （2）使用 APIcloud 建立前台 UI 模型。
- （3）研究卷积神经网络算法（CNN），完成病虫害识别功能。
- （4）后台使用 SSM（SpringMVC+Spring+Mybaits）与前台对接。
- （5）进一步优化农管 APP 软件，并对其进行功能性测试及分析。

1.4 内容的构成与章节安排

本文共分为五个章节。

第一章 绪论。本章主要介绍了农管 APP 软件的研究背景及意义，国内外发展现状以及研究内容。

第二章 相关技术。本章主要对农管 APP 所涉及到的软件以及相关技术进行介绍。

第三章 需求分析。本章从功能性需求和非功能性需求两方面分析农管 APP 软件需要完成的内容。

第四章 概要设计。本章主要系统功能架构设计、系统技术设计和子系统设计三个方面描述系统的逻辑模型。

第五章 关键模块的详细设计。本章主要介绍系统关键模块的实现。

第六章 数据设计。本章主要从数据库环境说明和数据字典两部分介绍系统的数据设计。

第七章 系统测试。本章主要对关键模块进行单元测试并给出测试结果。

2 相关技术

2.1 农管 APP 开发平台

2.1.1 APICloud

APICloud 移动应用开发平台的“云端一体”理念，能够大幅降低移动 APP 开发和管理的难度，可以帮助开发者实现软件生命周期的管理。APICloud 移动开发平台的数据服务模块支持自动转化 RESTful API，可以灵活地为 APP 提供数据服务支持。APICloud 由“云 API”和“端 API”两部分组成。端 API 可以完善终端适配，缩短开发周期，支持跨平台编码。

“端 API”将大量移动应用所需的功能“模块化”封装，并且借助 APICloud 的模块机制规范的提供给开发者，通过 JavaScript 语言，一套代码同时调用 iOS 和 Android 两个平台的模块，从而实现“积木拼装”式的原生应用开发；支持多人协作开发、一键多终端调试等服务^[4]。同时支持自定义开发，让移动应用开发更具灵活性和个性化。APICloud 的“云 API”提供基于 ACL（Access Control List）和 RBAC（Role Based Access Control）的访问控制模型安全机制，让开发者的移动应用在业务灵活性和安全性中找到平衡^[4]。

2.1.2 Maven

Java 开发时会引入许多开源的优秀组件，这些优秀的组件能够使开发者专注于处理服务器的请求。伴随着大量组件的引入，组件的管理和维护变得尤为重要。Maven 作为一个优秀的组件管理、项目构建和项目管理工具，不仅可以帮我们处理繁杂的代码清理、编译、打包和部署等工作，还可以为我们提供组件仓库，帮我们下载、组织和管理组件。

Maven 在长期的实际操作中展现出优异的便利性。它的使用让 Java 开发人员的日常工作变得更加简单。

2.2 农管 APP 相关技术

2.2.1 FastDFS 图片服务器集群

为了解决用户请求大量图片给网站带来的访问速度过慢、性能压力变大等问题，出现了专门的图片服务器。图片服务器作为专门存储图片的工具，通常使用一台或多台服务器搭建的集群进行存储图片。图片服务器可以实现页面代码和图片的分离，从而解决用户请求海量图片给项目带来的访问速度慢、性能压力大等问题。用户通常通过特定的 URL 地址访问图片服务器获得图片。

1、FastDFS 系统

FastDFS 是一个开源的轻量级分布式文件系统，采用分组存储的方式，支持纵向扩容（组中增加存储服务器）和横向扩容（增加组），可控性强，非常灵活。FastDFS 由跟踪服务器（tracker server）、存储服务器（storage server）和客户端（client）三个部分组成，其中跟踪服务器主要负责调度工作，起负载均衡的作用；存储服务器用于存储数据，直接向前端服务提供服务；主要是上传下载数据的服务器，也就是我们自己的项目所部署在的服务器。FastDFS 集群中跟踪服务器可以有多台，跟踪服务器和存储服务器之间不存在单点问题。跟踪服务器之间是对等关系，组内的存储服务器也是对等关系。具体的 FastDFS 如下：

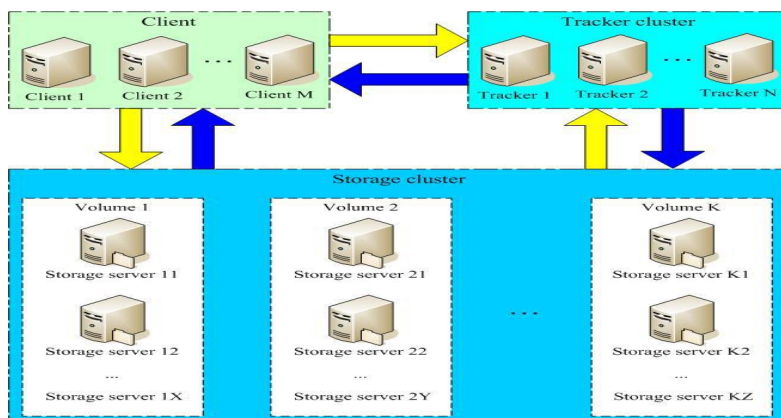


图 2.1 FastDFS 结构图

2、Nginx 服务器组件

由于图像服务器用于处理的并发请求在万级以上，数据量庞大，需要构建 FastDFS 集群，实现负载均衡，扩大并发请求量。在构建 FastDFS 集群的时候，负载均衡节点的并发支撑能力会成为限制整个系统并发能力的一个瓶颈^[5]。本文选择 Nginx 作为负载均衡和反向代理服务器组件，原因如下：

（1）高并发性

Nginx 在支撑海量连接与保证数据传输的高效性方面占据了核心的地位^[5]。不同的负载均衡策略针对不同的应用场景，Nginx 的设计原理更好的支持并发性。

（2）响应速度快

Nginx 采用多进程的方式工作，由一个 Master 进程和多个 worker 进程组成^[5]。其中 Master 负责接收外部控制信号，向各个 worker 发送信号并检测其运行状态。基本的网络事件则由 worker 进程处理。Nginx 多进程的工作模式充分发挥了多线程模型的优点，省去了竞争资源的部分，减少了对资源进行枷锁的操作，提高了服务器的并发处理能力。

（3）高扩展性、跨平台

Nginx 的设计极具扩展性，它完全是由多个不同功能、不同层次、不同类型且耦合度极低的模块组成。因此，当对某一个模块修复或进行升级时，可以专注于模块自身^[15]。

（4）热部署

master 管理进程与 worker 工作进程的分离设计，使得 Nginx 能够提供热部署功能，即可以在 7×24 小时不间断服务的前提下，升级 Nginx 的可执行文件^[15]。

（5）高可靠性

Nginx 的高可靠性来自于其核心框架代码的优秀设计、模块设计的简单性；每个 worker 进程相对独立，当有一个 worker 出现问题 master 会迅速开启另一个 worker 继续提供服务^[15]。选择 Nginx 的核心理由还是它能在支持高并发请求的同时保持高效的服务。

3、Keepalived

为了提高 FastDFS 集群的可靠性和安全性，需要采用多级热备技术防止出现单点问题。多机热备技术是指使用互为备份的几台服务器共同执行同一服务，即 active/standby 模式^[6]。其中只有一台服务器是工作机，其他的服务器都是备份。在系统正常的情况下，工作机为系统提供服务，备份机检测工作机的运行状态同样工作机也时刻关注着备份机是否正常运行。当工作机不能为系统提供服务是备份机主动接手工作机的任务，保证系统正常运转。

Keepalived 可以提供多机热备技术。Keepalived 利用 VRRP 协议来避免单点故障，主服务器和备份服务器对外表现为一个虚拟 IP，主服务器会发送特定的消息给备份服务器，当备份服务器收不到这个消息的时候（作为 MASTER 的主服务器不可用的时候），多台备份服务器中的 BACKUP

优先级最高的这台会被快速的抢占为 MASTER，随后接管虚拟 IP，继续提供服务，从而保证了系统的高可用性^[6]。

2.2.2 Redis

随着 Internet 技术的迅速发展，各种设备层出不穷，用户访问量日益增大，造成数据库负载过大、响应速度慢等，解决这个问题的根本在于提高数据库服务器承载能力方面。缓存技术就可以解决这一问题。Cache 性能高效，设计简单，可以对数据库中的数据进行缓存，降低数据库负载可以对 Web 页面进行缓存，提高 Web 页面响应速度；对复杂计算结果进行缓存，可以减少网站服务器的传输负荷和计算速度和对用户的响应速度，有效地提高网站性能及可扩展性^[7]。

2.2.3 Inception-V3 模型

卷积神经网络（CNN）与普通神经网络的区别在于，卷积神经网络包含了一个由卷积层和子采样层构成的特征抽取器^[16]。在卷积神经网络的卷积层中，一个神经元只与部分邻层神经元连接。在 CNN 的一个卷积层中，通常包含若干个特征平面（featureMap），每个特征平面由一些矩形排列的的神经元组成，同一特征平面的神经元共享权值，这里共享的权值就是卷积核^[16]。卷积核一般以随机小数矩阵的形式初始化，在网络的训练过程中卷积核将学习得到合理的权值。共享权值（卷积核）带来的直接好处是减少网络各层之间的连接，同时又降低了过拟合的风险。子采样也叫做池化（pooling），通常有均值子采样（mean pooling）和最大值子采样（max

pooling）两种形式,子采样可以看作一种特殊的卷积过程^[16]。卷积和子采样大大简化了模型复杂度，减少了模型的参数。

Inception-V3 模型中的 inception 结构是将不同的卷积层通过并联的方式结合在一起。同时使用所有不同尺寸的过滤器，然后再将得到的矩阵拼接起来。不同的矩阵代表了 Inception 模型中的一条计算路径。虽然过滤器的大小不同，但如果所有的过滤器都使用全 0 填充且步长为 1，那么前向传播得到的结果矩阵的长和宽都与输入矩阵一致。这样经过不同过滤器处理的结果矩阵可以拼接成一个更深的矩阵。Inception 的模块示意图如下所示：

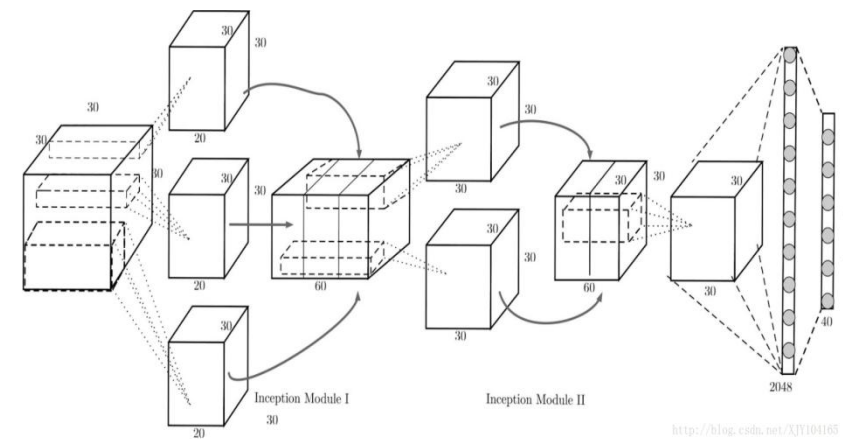


图 2.2 Inception 的模块示意图

Inception-V3 模型总共有 46 层，共有 11 个 inception 模块组成，Inception-V3 模型架构图如下：

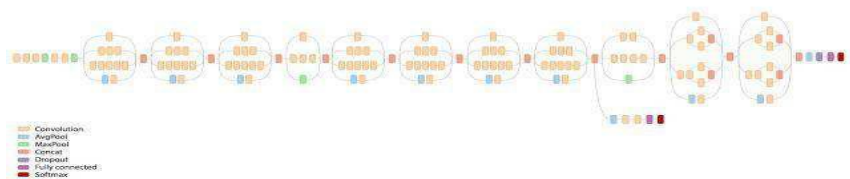


图 2.3 Inception-V3 模型架构图

2.2.4 MVC 设计模式

MVC (Model, View, Controller) 设计模式将项目的逻辑层和表现层分离, 这样使开发更加便利, 提高开发效率, 更强代码的可读性。

Model (模型层): 负责与数据库进行交互, 也就是对数据库中表的增加、删除、修改和查找。

View (视图层): 负责接收用户的输入、显示输出以及访问安全性验证。对输入数据的正确性、有效性, 对呈现样式负责, 对输出的数据正确性不负责, 但负责在数据不正确时给出相应的异常信息^[8]。

Controller (控制层): 设置用户输入数据的标准并读取数据交给模型层。

2.2.5 SSM

当前盛行的 Spring MVC + Spring + Mybatis (以下简称 SSM)^[9] 整合框架, 开发 Web 应用更加便利、高效, 同时 SSM 也是一个典型的 MVC 框架, 逻辑层和表现层完全分离。

Spring MVC: Spring MVC 是 Spring Frame Work 所推出的后续产品, 是 Spring 框架基于 MVC 设计模型的用于构建 Web 应用程序一个模块^[10]。Spring MVC 的核心是 DispatcherServlet, 它的作用是将请求分发给不同的后端处理器, 也即使用了一种被称为 Front Controller 的模式。对比 Struts2, 两者的功能相似, 但是它们的原理不同。Struts2 相当于是类级别的拦截, 一个类对应一个 request 上下文, Spring MVC 相当于方法级别的拦截, 每一个方法对应一个 request 上下文^[11]。

Spring: Spring 是一个开源框架, 是为了解决企业应用程序开发复杂

性而创建的^[17]。框架的主要优势之一就是分层架构，分层架构允许开发者选择使用组件，同时为 J2EE 应用程序开发提供集成的框架^[17]。Spring 的核心是控制反转 (Inversion of Control, IoC) 和面向切面 (AOP) 编程。

Mybatis: Mybatis 框架主要包括 DAO 组件与 SQLMap 组件两大类，mybatisDAO 组件主要是把应用程序的数据访问层和持久层的表示位置和方式抽象化，从而帮助开发人员基于 DAO 设计的模式进行设计^[12]。SQL Map 组件是 Mybatis 框架的重要组成部分，它通过通过配置 xml 或注解的方式将要执行的各 statement 配置起来，并通过 java 对象和 statement 中的 sql 进行映射生成最终执行的 sql 语句，最后由 mybatis 框架执行 sql 并将结果映射成 java 对象并返回^[11]。

2.2.6 EasyUI 界面插件集合

EasyUI 是一组基于 JQuery 的用户界面插件集合，它在优化了 Ajax 使用接口的同时，为 Web 开发提供了一系列的常用 UI 组件，包括菜单、对话框、布局、窗帘、表格、表单等等^[13]。它也是一个高质的 JavaScript 框架，开发者只需要了解简单的 html 标签就可以轻松地打造清晰美观的界面。

2.2.7 UEditor 富文本

UEditor 是由百度“FEX 前端研发团队”开发的富文本 web 编辑器，轻量级，可定制二次开发，注重用户体验，其开源性基于 MIT 协议，可以自由使用和修改代码^[14]。

3 需求分析

3.1 功能需求

3.1.1 系统角色功能描述

表 3.1 系统角色表

角色	角色权限
普通用户	基本用户权限
商家	上传商品
学生	对邀请用户的学生有奖励制度，促进学生发展
专家	解答农户问题，获得农人币
农机手	承包农管 APP 软件发布的任务
区域管理员	审核所在地区的开店申请、学生申请、专家申请、农机手申请
超级管理员	最高权限

系统中一共有七种角色，其中学生角色并无具体特定功能，只是用来资助大学生一种方法。普通用户的用例图如下：

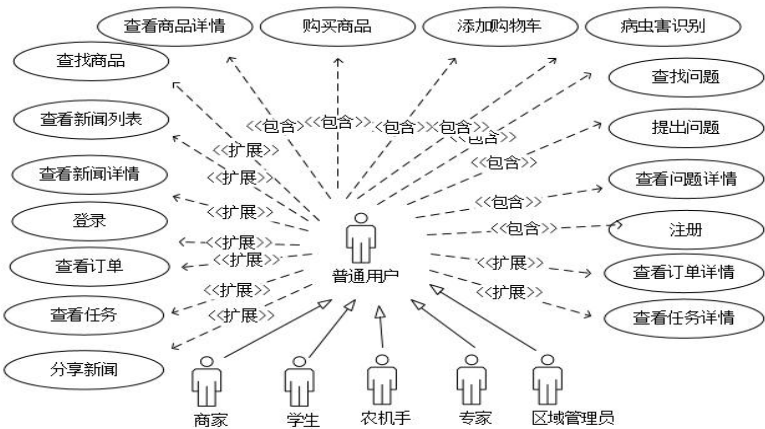


图 3.1 普通用户的用例图

普通用户通过提交申请成为商家、学生、农机手、专家和区域管理员。普通用户具有登录、注册、查看新闻列表、查找商品、购买商品等功能，关键功能描述如下：

商品查询：在用户已登录 Android 端或者用户登录 PC 端并且具有查询商品权限的条件下，输入商品名，查询到满足条件的商品列表（可以分页显示）。

查看商品详情：在用户已登录 Android 端或者用户登录 PC 端并且具有查看商品详情权限的条件下，点击商品查看商品详情，展示商品详情，包括现价、原价、格商品描述（介绍）等。

商品购买：在用户已登录 Android 端的条件下，点击“立即支付”，可以通过第三方软件（支付宝、微信）付款。

查看订单任务：在用户已登录 Android 端或者用户登录 PC 端并且具有查询任务权限的条件下，输入任务名模糊匹配任务，获取满足条件的任务列表并分页显示。

病虫害识别：用户通过上传图片，后台根据训练好的模型给出作物病虫害的名称。

提出问题：在用户已登录 Android 端的条件下，点击提问跳转到提问页，输入图片、问题主题、问题内容提交到后台。

新闻分享：在用户已登录 Android 端的条件下，点击分享按钮，可将新闻分享到第三方平台。

登录：用户通过输入手机号和密码登录到后台。

注册：用户输入手机号，点击获取验证码，输入验证码点击注册，将信息提交到后天进行验证。

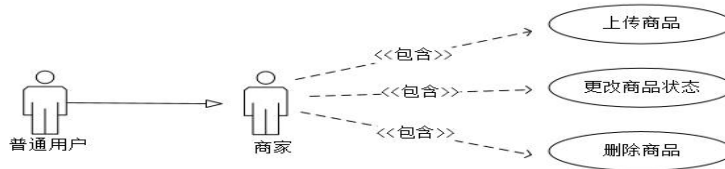


图 3.2 商家的用例图

商家主要是对商品进行增删改查操作，具体功能描述如下：

上传商品：在用户登录 PC 端并且具有上传商品权限的条件下，点击上传商品进入新增商品页，输入现价、原价、格商品描述（介绍）、商品单位、商品规格、库存、商品状态等信息，点击提交。

更改商品状态：在用户登录 PC 端并且具有更改商品状态权限的条件下，点击按钮更改商品状态。

删除商品：在用户登录 PC 端并且具有删除商品权限的条件下，点击删除按钮，后台根据商品 ID 删除商品。



图 3.3 农机手的用例图

农机手的特色功能是抢单操作。任务发布之后，系统挑选出一位最合适的抢农机手并发布抢单信息。具体功能描述如下：

抢单：在用户已登录 Android 端并且具有抢单权限的条件下，农机手点击抢单按钮，后台根据农机手位置和经验值选出最合适的农机手，前台显示农机手抢单的状态。

发布任务：在用户已付款购买商品并生成订单的条件下，后台获取订单信息填充任务内容并增加任务记录，任务列表要具有翻页功能。



图 3.4 专家的用例图

专家的特色功能是回答问题操作。具体功能描述如下：

回答问题：在专家登录 Android 端的条件下，输入解决问题的答案后，点击提交按钮，将回答提交给后台，后台存储后在前台展示。

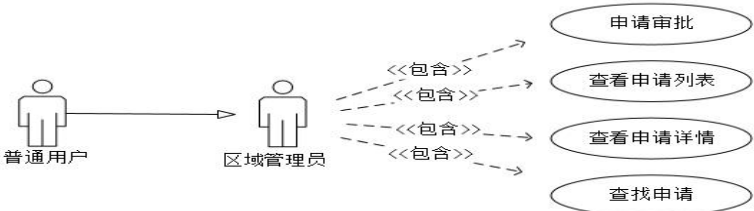


图 3.5 区域管理员的用例图

区域管理员主要处理专家申请、农机手申请、学生申请和商家申请四种，区域管理员的申请是超级管理员处理。具体功能描述如下：

申请审批：在用户登录 PC 端并且具有申请审批权限的条件下，区域管理员根据用户输入的内容修改数据库申请表，修改并保存后，展示审批成功。

查看申请列表：在用户登录 PC 端并且具有查询申请权限的条件下，用户点击进入申请审批模块，自动触发函数查询所有的申请并按照时间顺序分页显示。

查看申请详情：在用户登录 PC 端并且具有查看申请详情权限的条件下，点击申请进入申请详情页，展示申请人、身份证照片等信息。

查找申请：在用户登录 PC 端并且具有查询申请权限的条件下，输入申请类别模糊匹配申请并分页显示。

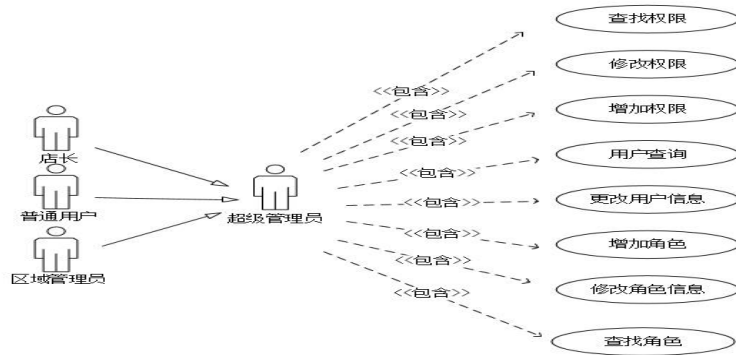


图 3.6 超级管理员的用例图

超级管理员主要管理权限、用户和角色的相关内容。具体功能描述如下：

查找权限：在用户登录 PC 端并且具有查找权限的权限条件下，输入权限类别模糊匹配权限信息，展示符合条件的权限列表并分页显示。

修改权限：在用户登录 PC 端并且具有修改权限的权限条件下，点击修改按钮进入修改页面，输入需要更改的内容点击提交按钮。

增加权限：在用户登录 PC 端并且具有增加权限的权限条件下，用户输入权限信息，点击添加按钮输入的信息便 json 串形式传到后台，前台显示添加权限成功。

用户查询：在用户登录 PC 端并且具有查找用户权限的条件下，输入用户名，根据用户名模糊匹配用户信息并展示。

更改用户信息：在用户登录 PC 端并且具有更改用户信息权限的条件下，点击修改按钮进入用户信息修改页面，输入需要更改的内容，点击提交，成功后显示修改成功信息。

增加角色：在用户登录 PC 端并且具有增加角色权限的条件下，点击添加按钮进入添加角色页，输入角色信息并息以 json 串形式传到后台，操作成功后显示“角色添加成功”的提示。

修改角色信息：在用户登录 PC 端并且具有修改角色权限的条件下，

点击修改按钮进入角色信息修改页，输入需要更改的内容，操作成功后显示“角色修改成功”的提示。

查找角色：在用户登录 PC 端并且具有查找角色权限的条件下，根据输入的角色名模糊匹配角色信息并展示。

3.1.2 系统模块功能描述

农管 APP 软件主要是用来辅助农户的农业生产活动。主要表现在如下几个方面：

（1）农商城：该模块主要包括查询商品、查看商品详情、加入购物车、购买商品、查找订单、查看订单详情、任务发布、农机手抢单等功能。

这个模块的商品由农业科技公司提供，包括两类，一是土壤生态链修复方案（或服务），农户可以根据土壤条件选择对应的处理方案；二是与土壤生态链处理方案配套的农资产品，包括各种有机肥、复合肥、化肥、生物菌剂、植物生长调节剂、各种农药等。对于第一类商品，农业科技公司会提供整套的服务，比如：农药、设备和操作人员等，避免了农户不懂技术难操作的问题。

种植户购买了第一种产品后，后台自动封装成任务进行发布，农机手抢单并完成任务。

（2）病虫害识别：该模块主要的功能是病虫害识别。农户上传病虫害图片，此模块通过机器学习算法迅速识别病症，给出病症原因和防治方法。

（3）农问吧：该模块主要包括问题查询、提出问题、回答问题、查看问题详情功能。农户提出问题，多位农业专家线上答题。

（4）农新闻：该模块主要包括新闻查找、查看新闻详情和分享三个功能点。发布最新的农业资讯，帮助农户把握农业行情走势。

3.2 非功能性需求

1. 资源需求

硬件环境：2 核 CPU、 2G 内存、100M 带宽

软件环境：操作系统为 CentOS 6.5 X64、分布式文件系统：fastdfs-5.05、代理服务器：Nginx-1.10.0、高可用性服务：Keepalived-1.2.24、系统服务器：tomcat8.0、数据库：mysql-5.7

2. 性能需求

由于系统的性能和效率受网络状况、流量、并发用户数量等因素的影响较大，以下对于性能的规定应该是在平均状况下的衡量。用户在使用病虫害识别功能要确保系统反馈结果在 2s 内完成；同时对用户在查询商品信息，交易等请求的时间需在 3s 内完成；在抢单环节，用户的抢单结果需在 3s 内反馈。

3. 安全需求

用户数据安全：禁止通过 URL 访问存放用户上传文档的文件夹，保护用户的隐私，防止非公开文档被恶意下载。并做到服务器数据的实时备份。

防注入：对 SQL 语句以及 URL 中的参数进行严格的过滤，规避非法请求，在上线前采用专业的漏洞扫描工具进行排查。

数据库安全：数据库管理具有很高的事务处理能力，设定数据库严格的访问控制权限，建立数据实时备份及容灾机制。

防 ddos 攻击：服务器应配有安全防护软件，监控服务器流量，对于异常，能通过短信、邮件等形式及时通知网站管理员。

4 概要设计

4.1 系统功能架构设计

该系统分为 Android 端农管 APP 软件和 PC 端后台管理系统。

访问 Android 端农管 APP 软件的用户主要是普通用户、学生、农机手和专家，其中普通用户可以在登录后进行各个模块的基本操作，比如：购买商品、提问农业问题和查看农业新闻等；普通用户通过申请成为“学生”后，可以以“学生”的角色权限获取邀请好友的奖励；普通用户通过申请“农机手”通过后，可以接系统派发的任务；普通用户通过申请成为“专家”后，可以通过回答农户提出的问题赚取“农人币”，“农人币”后期可以进行兑换。农管 APP 的结构图如下：

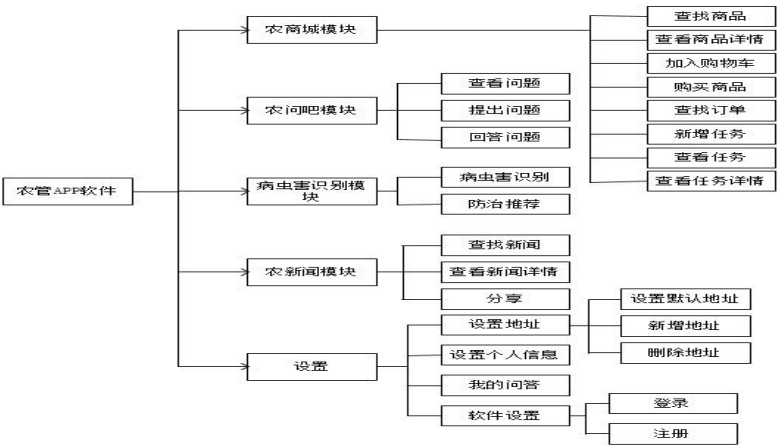


图 4.1 农管 APP 的结构图

访问 PC 端后台管理系统的用户有店长、区域管理员和超级管理员，其中店长可以进行商品管理，包括增加商品、查看商品信息、查找商品和改变商品状态；区域管理员是按照所属地区划分管辖区域，主要是审查本区域内提交的学生申请、专家申请和农机手申请；超级管理员拥有最高权限，可以进行商品管理、新闻管理、任务管理等。后台管理系统的结构图如下：

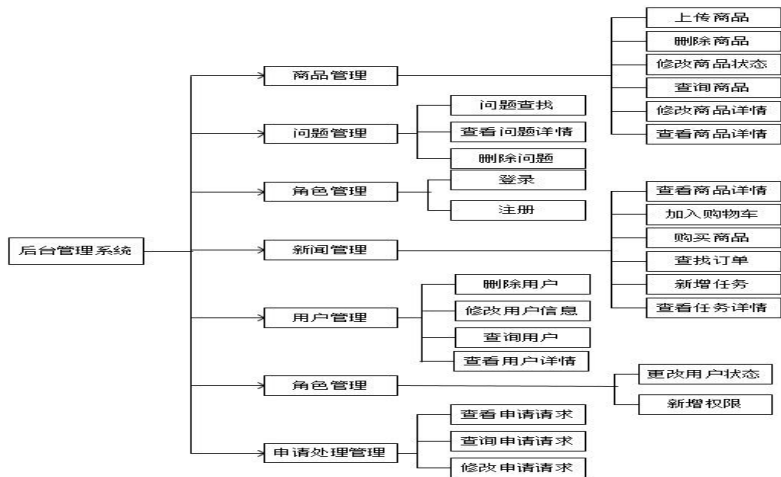


图 4.2 后台管理系统的结构图

4.2 系统技术架构设计

1. 系统技术架构设计

为了更好实现原设定的功能并提供更好的用户体验，根据农管 APP 软件的需求设计每部分使用的技术尤为重要。

APP 前端开发使用 APIcloud 移动应用开发平台。APIcloud 移动应用开发平台的“云端一体”的设计理念，能够帮助开发者管理软件的生命周期，降低开发和管理移动 APP 难度，缩短开发周期。

后台管理系统使用 EasyUI 界面插件集合，它封装了一系列常用的 UI 组件，简化了开发难度，让开发者专注于实现逻辑功能。

使用 Redis 作为 Cache 缓存用户经常访问的数据，提高软件响应速度，缓解数据库的压力。

使用 SSM（SpringMVC+Spring+Mybatis）框架，操作简单，提高代码的复用率。采用 MVC 结合 restful 设计模式，实现 APP 端对于数据及业务的请求。用户请求在经过自定义的拦截器进行用户权限校验后传递到 Controller 控制层来进行交互，Controller 层对请求参数进行初步审核后根

据需求调用对应的业务层（service）接口，同时将接收业务层处理结果封装成统一格式（ResponseResult）反馈给调用接口。

搭建 FastDFS 图片服务器集群，允许上传海量图片，检索迅速，提高机器学习的准确性。 系统技术架构设计图如下：

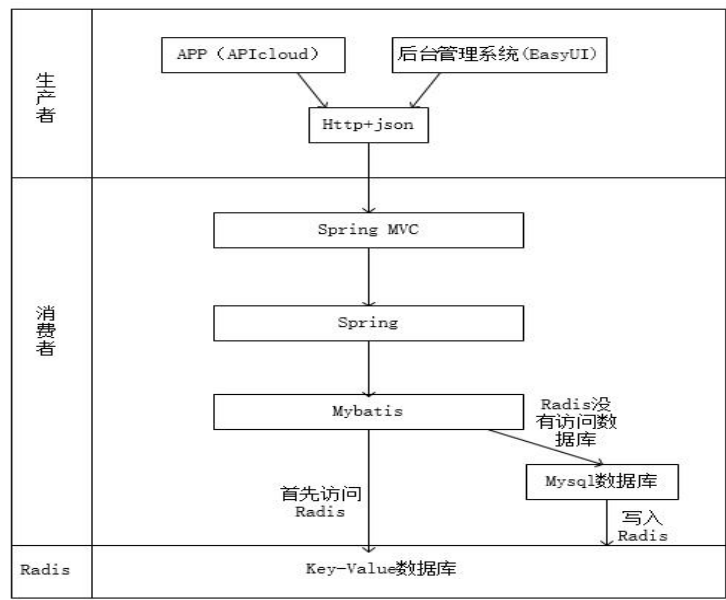


图 4.3 系统技术架构设计图

2. 系统配置

前台使用 APIcloud 移动开发平台，后台采用 SSM 框架、MySQL 数据库、Redis 缓存技术和 FastDFS 图片服务器集群，本系统应用技术比较多所以对版本的要求严格，系统的环境配置如下：

表 4.1 系统环境配置表

名称	版本
操作系统	6.5
Spring	4.1.4.RELEASE
Mybatis	3.2.8
SpringMVC	4.1.4.RELEASE

Redis	4.0.9.9
MySQL	5.6
Android	5.0.2
APIcloud	1.2.77
Keepalive	1.2.24
Nginx	1.10.0
FastDFS	5.05

4.3 子系统设计

本系统主要分为两部分——APP 端和后台管理系统。APP 端主要用于与用户交互，后台管理系统主要用于维持系统信息。

4.3.1 APP 端

4.3.1.1 农商城模块

1. 模块功能介绍

农商城模块是农管 APP 软件的“特色模块”，相较于其他农业 APP 软件将关注于农业信息的传播，这个模块真正的实现辅助农户的农业生产活动。此模块的侧重点不是向农户推荐商品、刺激农户消费，而是农业科技公司通过卖“技术服务”的形式为农户提供全套的技术支持，包括：农药、人员和器械等，农户可以现场学习先进的农业知识，掌握具体的操作步骤，以后可以独立进行操作。

农户可以通过申请成为“农机手”，经过农业科技公司培训后可以正式的接任务。农商城模块的业务流程图如下：

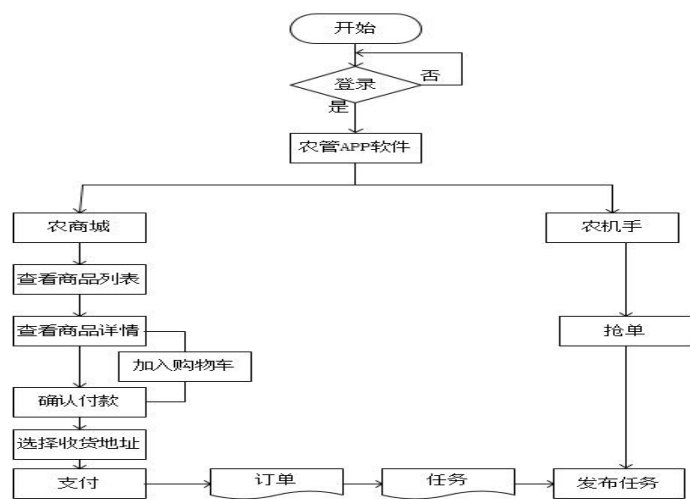


图 4.4 农商城业务流程图

2.模块关键

- （1）用户进入系统之后，可以选择不进行登录注册，但是购买商品时必须登录（注册后登录）；
- （2）用户登录注册之后，才可以申请成为“农机手”，申请成功后给用户赋予“农机手”权限，然后给农机手赋 60 分的信誉值；
- （3）农机手抢单模块，后台应先根据任务地址找出同地区的农机手，再找出信誉值最高的农机手，然后前台显示“抢单成功”。

4.3.1.2 农问吧模块

1. 模块功能介绍

农问吧模块给农户和专家提供了交流的平台。在这里，农户可以搜索问题，查看问题的回答记录，找出最优的解决方案；农户可以向指定的专家提问，专家消息提醒后会尽快的回复农户的问题，此处只允许专家进行回答，保证答案的可靠性。

农业专家进入农管 APP 软件后，首先要注册成为普通用户，然后提交

“申请专家”的请求，系统审核消息后给予答复，若申请成功，可以行使专家的权限，回答农户问题领取奖励。农问吧模块的业务流程图如下：

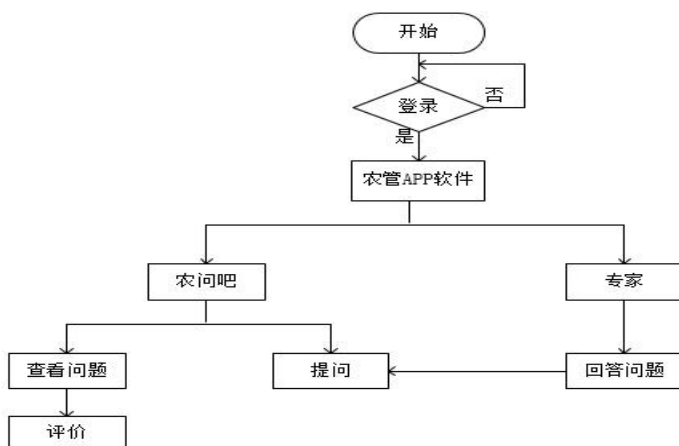


图 4.5 农问吧业务流程图

2. 模块关键

（1）用户登录注册成为普通用户后，才可以申请成为“专家”，申请成功后给用户赋予“专家”权限；

（2）专家回答完问题后，只要农户评价满意，专家才能获得奖励；

（3）只有专家能回答农户问题，农户提出问题后不仅是指定专家回答，其他专家也可以就农户问题给出答案。

4.3.1.3 农新闻模块

1. 模块功能介绍

农新闻模块主要是实时展示农业科技资讯，用户拥有执行分享功能，可以分享到朋友圈、微信、微博等第三方软件。农新闻模块的业务流程图如下：

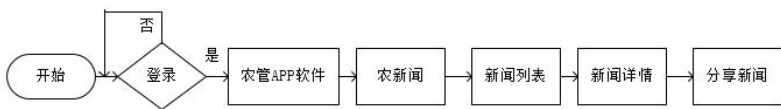


图 4.6 农新闻业务流程图

2. 模块关键

- (1) 分享时，用户必须登录；
- (2) 第三方分享的实现。

4.3.1.4 病虫害识别模块

1.模块功能介绍

病虫害识别模块的主要目的是帮助农户快速识别病虫害。采用时下比较火的机器学习方法——卷积神经网络算法（CNN），对采集的海量病虫害图片进行训练，得到一个精确度比较高的模型。农户只需要把病虫害图片上传到这个模块，该病虫害识别模型就可以给出正确的识别结果，并能提供给农户合理解决病虫害的方案。

在这个模块，如何存储海量的图片数据是一个难点，本文拟采用搭建图片服务器集群的方法解决这个问题。通过搭建图片服务器集群，可以降低数据库存放数据的压力，扩展性的缓存图片数据，提高图片的检索速度，提高机器学习的精度。病虫害模块的业务流程图如下：

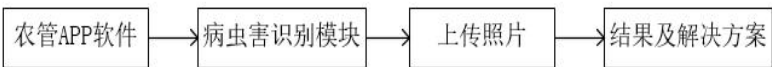


图 4.7 病虫害识别业务流程图

2. 模块关键

- (1) 海量图片的存储问题；
- (2) 搜集病虫害图片并归纳整理，找出对应的病虫害的防治方法。

4.3.1.5 登录注册模块

1. 模块功能介绍

登录注册模块是使用农管 APP 软件的基本模块。

注册时，用户需要输入手机号，后台调用接口给用户手机发送验证码，用户在规定 30 秒内输入验证码，后台验证用户是否注册成功。若用户注册成功，后台将自动生成的密码发给用户，用户可以在软件设置模块修改个人密码，接着，农管 APP 软件将自动跳到登录页面，让用户进行登录，登录成功后进入农管 APP 软件首页。登录注册模块的业务流程图如下：

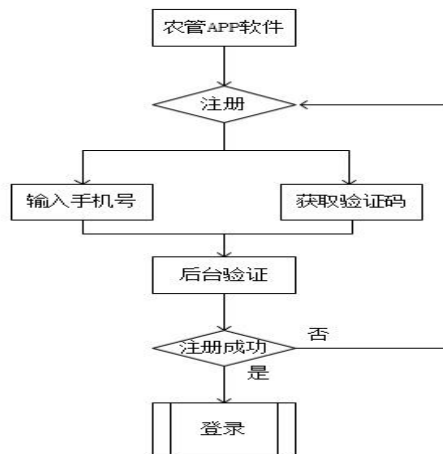


图 4.8 注册业务流程图

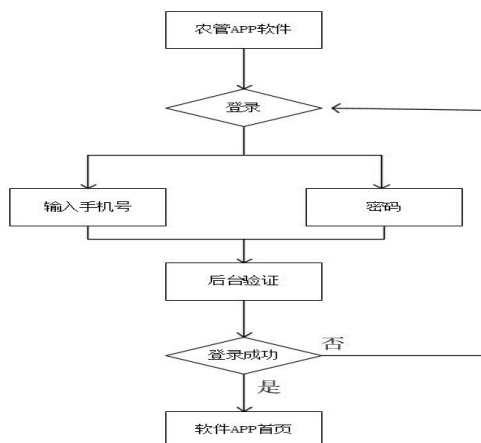


图 4.9 登录业务流程图

2. 模块关键

(1) 如何调用接口向用户发送验证信息？

4.3.1.6 设置模块

1. 模块功能介绍

设置模块主要是设置和展示用户的个人信息，包括设置密码、我的问答、填写邀请码等。

当用户登录成功后，为了保护账户的安全性，可以在我的模块重新设置新密码，需要用户输入原来地密码然后输入新的密码，向后台提交申请。

填写邀请码功能点，主要是针对“学生”的奖励政策，当普通用户申请“学生”通过后，学生可以邀请他人注册使用农管 APP 软件，只要新用户在登录注册后填写邀请者的邀请码，学生就可以得到系统定义的奖励。

我的问答功能点重在展示用户在问吧模块提出或者解决的问题，可以查看问题详情。设置模块的业务流程图如下：

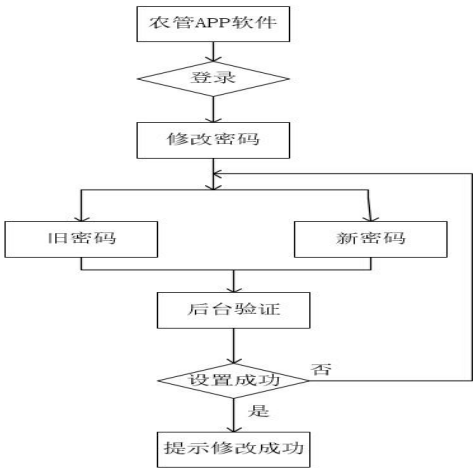


图 4.10 修改密码业务流程图

2. 模块关键

- (1) 用户登录后才能修改密码；
- (2) 问题展示时，如何将历史信息展示？数据库表如何设计？

4.3.2 后台管理系统

4.3.2.1 商品管理

1. 模块功能介绍

商品管理模块主要是为店家提供的功能，店家通过这个模块可以操作商品，主要包括增加商品、删除商品、查看商品详情、查询商品、修改商品信息和改变商品的状态。商品管理模块业务流程图如下：

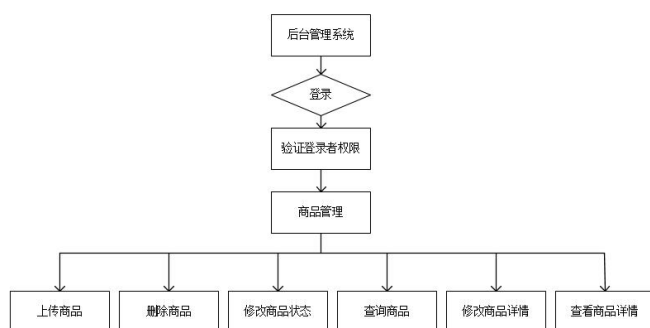


图 4.11 商品管理业务流程图

2. 模块关键

店家只能修改处于“未上架”状态的商品信息，若想改变“已上架”商品的具体内容只能通过删除商品再重新上传的方式。

4.3.2.2 新闻管理

1. 模块功能介绍

新闻管理模块管理的内容是农管 APP 软件农新闻模块的展示内容，访问权限只面向超级管理员，超级管理员有发布新闻、查找新闻、修改新闻、

查看新闻详情和删除新闻的权限。新闻模块流程图如下：

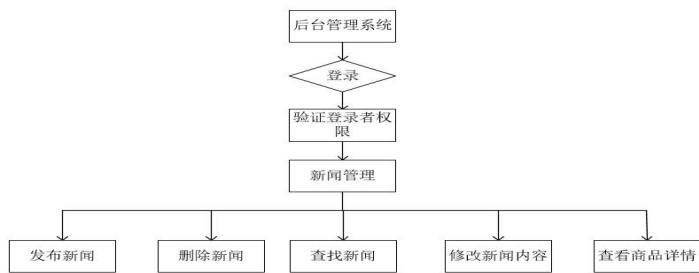


图 4.12 新闻管理业务流程图

2. 模块关键

设置访问权限，只有超级管理员才能访问此模块。

4.3.2.3 问题管理

1. 模块功能介绍

问题管理模块管理的内容是农管 APP 软件农问题模块的展示内容，访问权限只面向超级管理员，超级管理员有查找问题和查看问题问题的权限。问题管理模块业务流程图如下：

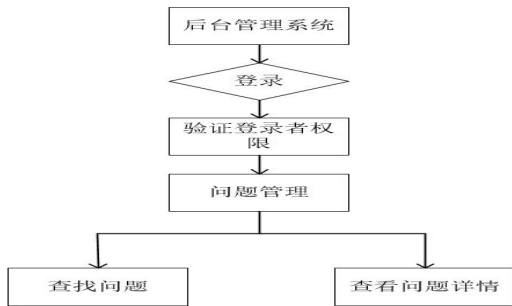


图 4.13 问题管理业务流程图

2. 模块关键

设置访问权限，只有超级管理员才能访问此模块。

4.3.2.4 角色管理

1. 模块功能介绍

系统内的角色包括：普通用户、农机手、学生、专家、区域管理员和超级管理员。用户管理主要功能是更改角色状态和添加角色。角色管理模块业务流程图如下：

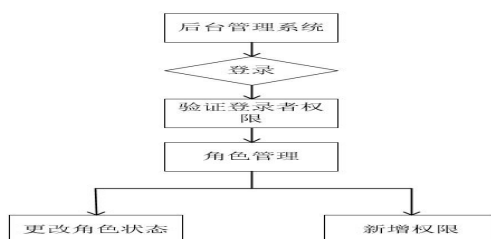


图 4.14 角色管理业务流程图

2. 模块关键

- (1) 设置访问权限，只有超级管理员才能访问此模块；
- (2) 增加角色时，要给新的角色赋予响应的权限。

4.3.2.5 用户管理

1. 模块功能介绍

人员管理主要面向超级管理员具备的权限，主要有删除用户、修改用户内容、查询用户信息及查看用户详情信息四个功能点。用户管理业务流程图如下：

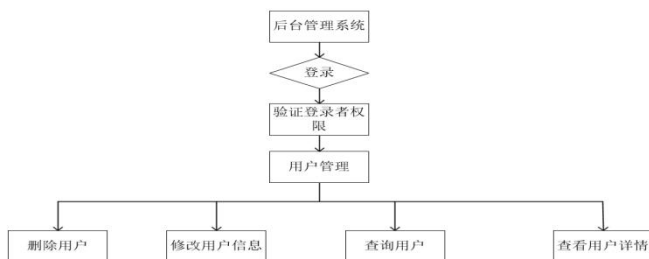


图 4.15 用户管理业务流程图

2. 模块关键

设置访问权限，只有超级管理员才能访问此模块。

4.3.2.6 申请处理管理

1. 模块功能介绍

申请处理模块主要是处理用户发来的申请请求，包括申请农机手请求、申请专家请求和申请学生请求。这个模块主要面向区域管理员和超级管理员，区域管理员处理所在区域的请求，给出结果和原因。

该模块包括查看申请请求、查询申请请求、修改申请请求三个功能点。申请功能流程图如下：

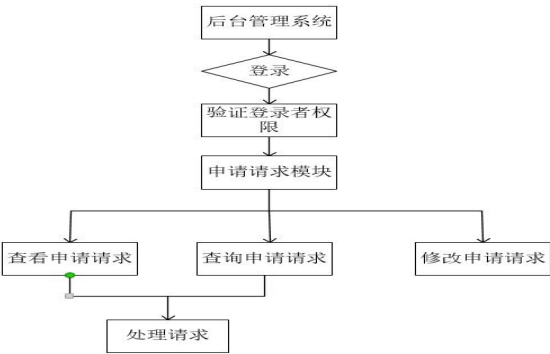


图 4.16 申请处理管理业务流程图

2. 模块关键

区域管理员要审核申请者信息是否符合要求，最好实地考察。

5 数据设计

5.1 数据库环境说明

数据库采用 MySQL, 版本为 5.5, 数据库建模设计工具为 Power Designer 15。

5.2 数据字典

本系统包含内容较多, 数据库设计了 38 张表。为了便于查看, 此处主要展示每个功能模块主要依附的表, 具体内容如下:

1. 农新闻模块

该模块的主要功能是转发新闻, 涉及到用户表、新闻表等。新闻表主要包含标题、内容、展示图片地址、发布时间、新闻来源和新闻类型七个字段, 字段具体描述如下:

5.1 新闻 (t_news)

字段名称	数据库字段	类型长度	备注
新闻记录 ID	ID	Varchar(255)	主键, 自增
标题	TITLE	Varchar(100)	
内容	CONTENT	Varchar(255)	
展示图片地址	PICTURE_URL	Varchar(50)	
发布时间	PUBLISH_TIME	Datetime(0)	
新闻来源	ORIGNAL	Varchar(50)	
新闻类型	NEWS_TYPE	Varchar(50)	虫害, 玉米等等

2. 农问吧模块

该模块主要功能是农户提问、专家回答和问题展示，涉及到问题记录表、发言记录表以及中间表等。

问题记录表主要记录用户提出的问题，具体描述如下：

5.2 问题记录(t_questions)

字段名称	数据库字段	类型长度	备注
问题 ID	QID	Int(11)	主键，自增
主题（作物名称）	SUBJECT	Varchar(50)	用于分类
问题描述（内容）	DESCRIPTION	Varchar(255)	
提问时间	APPLY_TIME	Datetime(0)	
提问人 ID（用户 ID）	USER_ID	Varchar(255)	外键
状态	STATUS	Int(11)	默认为 1 0 不显示（违规） 1 正常显示
邀请专家回答（专家 ID）	EXPERT_ID	Varchar(255)	默认 null 外键

发言记录表主要记录专家回答用户问题的内容，具体如下：

5.3 发言记录：(t_spoke)

字段名称	数据库字段	类型长度	备注
发言记录 ID	SPOKE_ID	Int(11)	主键，自增
发言内容	CONTENT	Varchar(255)	
发言时间	SPOKE_TIME	Datetime(3)	

是否采纳（最优）	FLAG	Int(3)	默认为 0 0：普通 1：最优
状态	STATUS	Int(3)	默认为 1 0 不显示 1 显示
发言人 ID	USER_ID	Varchar(255)	外键

3. 农商城模块

商城模块主要是展示商品、添加购物车、购买商品、发布任务、抢单等功能，对应数据库中商品表、购物车表（用户和商品的中间表）、任务表、订单表等。

商品表总结系统中商品具有的属性，如下：

5.4 商品(t_products)

字段名称	数据库字段	类型长度	备注
商品 ID	PID	Varchar(255)	主键唯一
商品名	PRODUCT_NAME	Varchar(100)	产品展示页面显示
现价	PRICE_NOW	Int(11)	现价和原价可以相同，单位分
原价格	PRICE_BEFORE	Int(11)	
商品描述（介绍）	DESCRIPTION	text	
商品单位	PRODUCT_MODE L	Varchar(100)	袋、斤、毫升、升，平方米
商品规格	PRODECT_UNIT	Varchar(50)	

运费标准 ID	DELIVER_PRICE	Int(11)	外键
库存	STORAGE	Double(0.0)	
商品状态	STATUS	Int(3)	上架、下架
商品类别 ID	PRODUCT_TYPE	Int(11)	外键,用于分类 推荐
所属店铺 ID	STORE_ID	Varchar(255)	外键
备注	REMARK	Varchar(255)	

购物车是用户和商品的中间表，具体描述如下：

5.5 购物车(t_cart)

字段名称	数据库字段	类型长度	备注
记录 ID	CID	Int(11)	主键唯一，自增
商品 ID	PRODUCT_ID	Varchar(255)	外键
商品数量	PRODUCT_COUNT	Double(0.0)	
商品单价（现价）	PRODUCT_NOW	Int(11)	该价格对应商品规格
商品规格	PRODUCT_MODE L	Varchar(100)	
添加购物车时间	ADD_TIME	Datetime(0)	
用户 ID	USER_ID	Varchar(255)	外键

订单表包含着用户信息、商品信息、物流信息和地址信息等，具体描述如下：

5.6 订单(t_orders)

字段名称	数据库字段	类型长度	备注
------	-------	------	----

订单 ID	ORDER_ID	Varchar(255)	主键唯一
商品 ID	PRODUCT_ID	Varchar(255)	外键
店铺 ID	STORE_ID	Varchar(255)	外键
用户 ID	USER_ID	Varchar(255)	外键
地址 ID	RECERVER_ID	Varchar(255)	外键
物流单位	DELIVER_UNIT	Varchar(50)	顺丰、申通、韵达等
物流单号	DELIVER_NUM	Varchar(100)	
商品单价（现价）	PRICE_NOW	Int(11)	该价格对应商品规格
商品规格	PRODUCT_MODEL	Varchar(50)	
商品数量	PRODUCT_COUNT	Double(0.0)	
运费金额	DELIVER_PRICE	Int(11)	
优惠金额	SALE_PRICE	Int(11)	优惠券或农人币等政策
应付金额	REAL_MONEY	Int(11)	
订单状态	STATUS	Varchar(100)	0 待支付 1 货到付款 2 待发货 3 物流中 4 待签收 5 待确认 6 交易完成 7 退单处理中 8 退单完成
订单创建时间	ORDER_TIME	Datetime(0)	

付款时间	PAY_TIME	Datetime(0)	
发货时间	DELIVER_TIME	Datetime(0)	
交易完成时间	FINISH_TIME	Datetime(0)	
备注	REMARK	Varchar(255)	

订单产生后，系统根据订单信息自动生成任务并发布，任务发布表的具体内容如下：

5.7 任务发布表(t_tasks)

字段名称	数据库字段	类型长度	备注
任务 ID	ID	Int(11)	主键，自增
任务标题	TITLE	Varchar(100)	
任务内容	CONTENT	text	如除草
任务地点	ADDR	Varchar(255)	详细地址精确到村子
任务联系人姓名	LINKNAME	Varchar(50)	
联系人电话	PHONE	Varchar(50)	
工作量	WORKLOAD	Double(0.0)	如：10 亩
结算方式	PAY_TYPE	Varchar(100)	日结，小时，任务
薪酬	PAY_MONEY	Int(11)	单位分
发布时间	PUBLISH_TYPE	Datetime(0)	
状态	STATUS	Int(3)	0 删除失效 1 待接单 2 已接单 3 已完成
备注	REMARK	Varchar(255)	
任务评分	TASK_GRADE	Double(0.0)	满分 10 分
任务发布人 ID	PUBLISH_USER	Varchar(255)	用户 ID，默认管理员
接任务人 ID	ACCEPT_USER	Varchar(255)	用户 ID，接受任务人

			员
--	--	--	---

4. 病虫害识别模块

病虫害识别模块主要功能是识别农作物病虫害，主要涉及到智能诊断表。智能诊断表包含上传人、上传时间、诊断完成时间、诊断结果等字段，具体描述如下：

5.8 智能诊断(t_medicals)

字段名称	数据库字段	类型长度	备注
诊断 ID	ID	Int(11)	主键，自增
照片 URL	IMAGE_URL	Varchar(255)	
上传人	UPLOAD_USER	Varchar(255)	外键
上传时间	UPLOAD_TIME	Datetime(0)	
诊断完成时间	FINISH_TIME	Datetime(0)	
诊断结果	RESULT	Varchar(255)	
推荐商品关键字	PRODUCT_KEY	Varchar(255)	

6 系统实现

6.1 系统环境搭建

该系统搭建主要包含两部分，一是搭建 FastDFS 图片服务器集群，二是整合 SSM(SpringMVC+Spring+Mybatis)和 Redis，其系统搭建过程如下：

6.1.1 图片服务器集群构建

1. 整体描述

使用八台服务器搭建 FastDFS 集群，其中两台作为跟踪服务器负载均衡节点，两台作为跟踪服务器，剩余的四台服务器分两组作为存储服务器。具体的安排如下表：

表 6.1 图片服务器集群搭建安排

服务器	IP 地址	命名	使用技术
跟踪服务器负载均衡节点 1	172.31.43.118	dfs-nginx-proxy-1	Nginx 、 Keepalived
跟踪服务器负载均衡节点 2	172.31.43.119	dfs-nginx-proxy-2	Nginx 、 Keepalived
跟踪服务器 1	172.31.43.108	dfs-tracker-1	Nginx 、 FastDFS
跟踪服务器 2	172.31.43.109	dfs-tracker-2	Nginx 、 FastDFS

存储服务器 1	172.31.43.113	dfs-storage-group 1-1	Nginx 、 FastDFS
存储服务器 2	172.31.43.112	dfs-storage-group 1-1	Nginx 、 FastDFS
存储服务器 3	172.31.43.114	dfs-storage-group 1-1	Nginx 、 FastDFS
存储服务器 4	172.31.43.117	dfs-storage-group 1-1	Nginx 、 FastDFS

实现后，总体效果图如下：

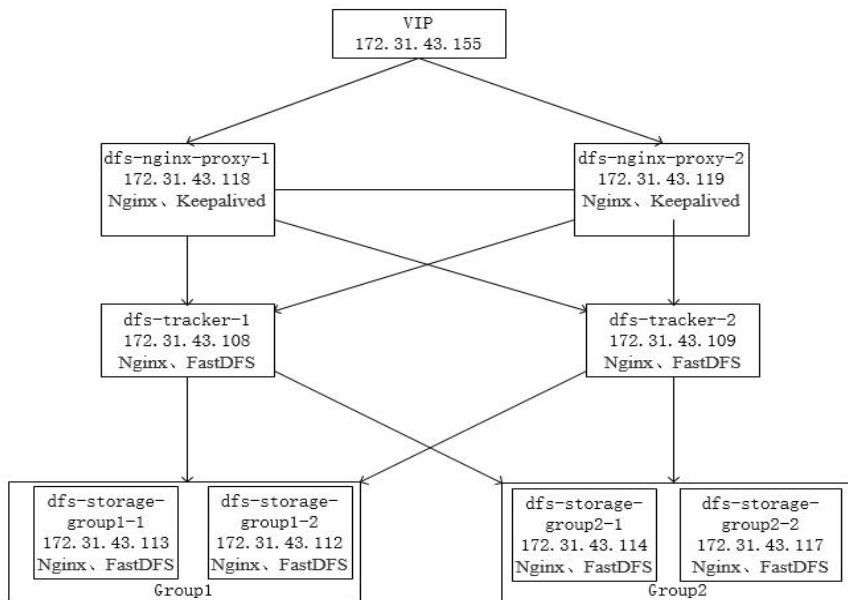


图 6.1 FastDFS 集群总体效果图

上传文件的流程为：用户统一访问虚拟 IP，虚拟 IP 将访问请求转发给代理服务器，若两台代理服务器中有一台损坏，Keepalived 将用户的请求全部转发给另一台代理服务器，提高了系统的稳定性，紧接着，代理服务器通过 Nginx 进行 http 访问跟踪服务器并提交用户的上传的文件，跟踪服务器获取文件后根据自己的规则存储到指定存储服务器的相应路径下，

同时，跟踪服务器会生成文件在服务器上的存储路径，通过 Nginx 返回给客户端。具体流程图如下：

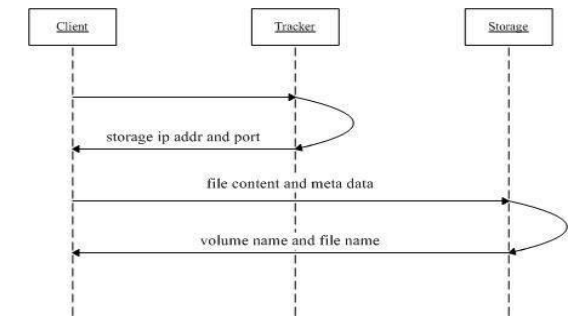


图 6.2 上传文件流程图

下载文件流程：用户上传文件后会收到跟踪服务器发送的文件存储路径，访问文件时，直接访问“虚拟 IP+文件存储路径”即可。若在同组存储服务器文件还没有复制完成的情况下，用户访问文件，就会出现文件无法访问的错误。而 Nginx 可以重定向文件连接到源服务器（首个存储的服务器）上取文件，避免客户端由于复制延迟导致的文件无法访问错误。

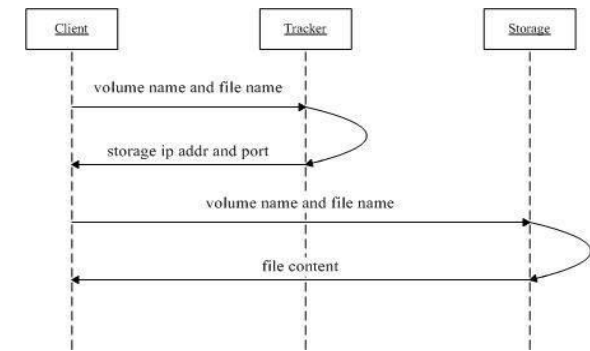


图 6.3 下载文件流程图

2. 搭建步骤

一. 安装集群节点

在所有跟踪服务器和存储服务器节点上执行以下操作：

- 1) 安装所需的依赖包
- 2) 安装 libfatscommon

3) 安装 FastDFS

二. 配置跟踪节点

1) 复制 **tracker** 样例配置文件, 并重命名

2) 修改 **tracker.conf** 配置文件

```
disabled=false           #启动配置文件
port=22122               #tracker 服务器端口 (默认 22122)
base_path=/fastdfs/tracker #存储日志和数据的根目录
```

3) 创建 **base_path** 指定的目录

4) 防火墙中打开 **tracker** 服务器端口 (默认为 22122)

5) 设置 **tracker** 服务开机启动

三、配置存储节点

1) 复制 **storage** 样例配置文件, 并重命名

2) 编辑配置文件

```
disable=false           #启动配置文件
port=23000              #storage 服务端点
group_name=group1       #组名
base_path=/fastdfs/storage #数据和日志文件存储根目录
store_path0=/fastdfs/storage #存储目录
store_path_count=1      #存储路径搭配
tracker_server=192.168.0.200:22122 #tracker 服务 IP 和端口
tracker_server=192.168.0.201:22122 #tracker 服务 IP 和端口
http.server port=8888    #http 访问文件的端口
```

3) 创建基础数据目录

4) 防火墙中打开 **storage** 服务器端口 (默认为 23000)

5) 设置 **storage** 服务开机启动

四、文件上传测试

1) 修改 tracker 服务器 client.conf 配置文件

```
base_path=/fastdfs/tracker

tracker_server=192.168.1.200:22122

tracker_server=192.168.1.201:22122
```

2) 上传文件

通过调用 `api`, 执行 `linux` 命令上传文件。

```
public static Map<String, Object> uploadLocalFile(String filePath) {
    Map<String, Object> retMap = new HashMap<String, Object>();
    // 1.上传文件的命令
    String command = "fdfs_upload_file /etc/fdfs/client.conf " + filePath;

    // 2.定义文件的返回信息
    String fileId = "";
    InputStreamReader inputStreamReader = null;
    BufferedReader bufferedReader = null;
    try {
        // 3.通过调用 api, 执行 linux 命令上传文件
        Process process = Runtime.getRuntime().exec(command);
        // 4.读取上传后返回的信息
        inputStreamReader = new InputStreamReader(process.getInputStream());
        bufferedReader = new BufferedReader(inputStreamReader);
        String line;
        if ((line = bufferedReader.readLine()) != null) {
            fileId = line;
        }
        // 5.如果 fileId 包含 M00, 说明文件已经上传成功。否则文件上传失败
        if (fileId.contains("M00")) {
            retMap.put("code", "0000");
            retMap.put("group", fileId.substring(0, 6));
            retMap.put("msg", fileId.substring(7, fileId.length()));
        } else {
            retMap.put("code", "0001"); //上传错误
            retMap.put("msg", fileId); //返回信息
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```

    }
    } catch (Exception e) {
        logger.error("IOException:" + e.getMessage());
        retMap.put("code", "0002");
        retMap.put("msg", e.getMessage());
    } finally {
        if (inputStreamReader!=null){
            try {
                inputStreamReader.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        if (bufferedReader != null) {
            try {
                bufferedReader.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
    }
    return retMap;
}

```

调用上述方法上传文件成功后，返回图片访问的 URL 地址，格式如下：

http://172.31.43.155/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7F
JU4.tar.gz

6.1.2 SSM+Redis 整合

本系统考虑真是应用环境中多用户并发访问的需求，随着访问量提升，出现了 IO 密集型访问（主要是读取密集型），从而导致读取时间变慢。为提升性能，本系统使用了缓存技术，主要是对数据库的读写操作进行分离。本系统综合考虑性能、安全等多个方面，结合本系统的业务特点将多种缓存技术针对下表几项指标进行对比，最终选择了 Redis 作为系统缓存数据

库。

表 6.2 缓存技术对比

指标	Redis	Memcache
多数据类型	√	×
分布式存储	√	√
持久化	√	×
过期策略	√	×
数据恢复	√	×
数据备份	√	×

综上所述,Redis 不仅仅支持简单的 k/v 类型的数据,同时还提供 list, set, zset, hash 等数据结构的存储。而且在数据安全备份方面,Redis 支持数据的备份,即 master-slave 模式的数据备份。因此选择 Redis,可以避免写入不必要的临时数据,也免去了对临时数据进行扫描或者删除的麻烦,并最终改善程序的性能。

在框架整合方面,配置如下:

```
<!--设置链接属性 -->

<bean id="jedisConnectionFactory"

class="org.springframework.data.redis.connection.jedis.JedisConnectionFactory"

p:hostName="${redis.host}" p:port="${redis.port}" p:password="${redis.password}"

p:pool-config-ref="poolConfig" p:timeout="100000" />

<!-- Jedis 模板配置 RedisTemplate 对 RedisConnection 进行了封装。提供连接管
理,序列化等功能,它对 Redis 的交互进行了更高层次的抽象,极大的方便和简化
了 Redis 的操作 -->

<bean id="redisTemplate"

class="org.springframework.data.redis.core.StringRedisTemplate">
```

6.2 病虫害识别模型

6.2.1 卷积神经网络迁移学习

表 6.3 给出了从 2012 年到 2015 年 ILSVRC (Large Scale Visual Recognition Challenge) 第一名模型的层数以及前五个答案的错误率。

表 6.3 ILSVRC 第一名模型信息表

年份	模型名称	层数	Top5 错误率
2012	AlexNet	8	15.3%
2013	ZF Net	8	14.8%
2014	GoogLeNet	22	6.67%
2015	ResNet	152	3.57%

从如上表中可以看出,随着模型层数及复杂度的增加,模型在 ImageNet 上的错误率也随之降低。然而,训练复杂的卷积神经网络需要非常多的标注数据。如 ImageNet 图像分类数据集中有 120 万标注图片,所以才能将 152 层的 ResNet 的模型训练到大约 96.5%的正确率,但是在实际应用中很难收集到如此多的标注数据。即使有海量的训练数据,要训练一个复杂的卷积神经网络也需要几天甚至几周的时间,为了解决标注数据和训练时间的问题,病虫害识别模型采用迁移学习获得。

迁移学习就是将一个问题上训练好的模型通过简单的调整使其适用于一个新的问题。根据论文 DeCAF:A Deep Convolutional Activation Feature for Generic Visual Recognition 中结论,可以保留训练好的 Inception-V3 模型中所有卷积层的参数,只是替换最后一层全链接层。在最后一层全链接层之前的网络层被称为瓶颈层 (bottleneck)。

将新的图像通过训练好的卷积神经网络直到瓶颈层的过程可以看成是对图像进行特征提取的过程。训练好的 Inception-V3 模型可以很好的区分

1000 种类型的图像，所以瓶颈层输出的节点向量可以直接利用这个训练好的神经网络对图像进行特征提取，然后再将提取的特征向量作为输入来训练一个新的单层全链接神经网络处理新的分类问题。

6.2.2 模型训练

根据如上迁移学习方式，通过 TensorFlow 平台根据病虫害训练集训练 Inception-V3 模型，实现病虫害图片的分类识别功能，具体伪码如下：

```
# 从谷歌官网下载训练好的 Inception-v3 模型文件目录
Inception_DIR = '/path/to/model/google2015-inception-v3'
# 训练好的 inception-v3 模型文件名
Inception_FILE = 'tensorflow_inception_graph.pb'
# 保存瓶颈层训练好的特征向量
CACHE_DIR = 'tmp/bottleneck'
# 从数据文件夹中读取所有的图片并按照训练、验证、测试数据分开
def create_myimage_lists(testing_percentage, validation_percentage):
# 通过类别名称、所属数据集和图片编号获取一张图片的地址
def get_image_path(image_lists, image_dir, label_name, index, category):
# 处理后的文件特征向量地址
def get_bottleneck_path(image_lists, label_name, index, category):
# 加载模型中的一张图片并计算特征向量
def run_bottleneck_on_image(sess, image_data, image_data_tensor,
bottleneck_tensor):
# 找到图片对应的特征向量并保存
def get_or_create_bottleneck(sess, image_lists, label_name, index, category,
jpeg_data_tensor, bottleneck_tensor):
```

```

# 随机获取一个 batch 的图片作为训练数据

def get_random_cached_bottlenecks(sess, n_classes, image_lists,
how_many, category, jpeg_data_tensor, bottleneck_tensor):

# 获取全部的测试数据

def get_test_bottlenecks(sess, image_lists, n_classes, jpeg_data_tensor,
bottleneck_tensor):

#主方法

def main(_):

# 定义全链接层解决图片分类问题

with tf.name_scope('final_training_ops'):

#输出判断结果

print(tf.argmax(final_tensor,1))

```

6.2.3 模型使用

模型训练完成后，系统将会输出有两个文件：“ouput.pb”文件和“labels.txt”文件，其中“ouput.pb”文件存储病虫害识别模型的图（Graph），“labels.txt”文件存储病虫害识别模型图片的分类。编写“retrain_model_classifier.py”文件使用该模型，具体代码如下：

```

#读取要识别的图片

image_data = tf.gfile.FastGFile(image_path, 'rb').read()

# 加载图片分类问价 “labels.txt”

label_lines = [line.rstrip() for line
                 in tf.gfile.GFile("./labels.txt")]

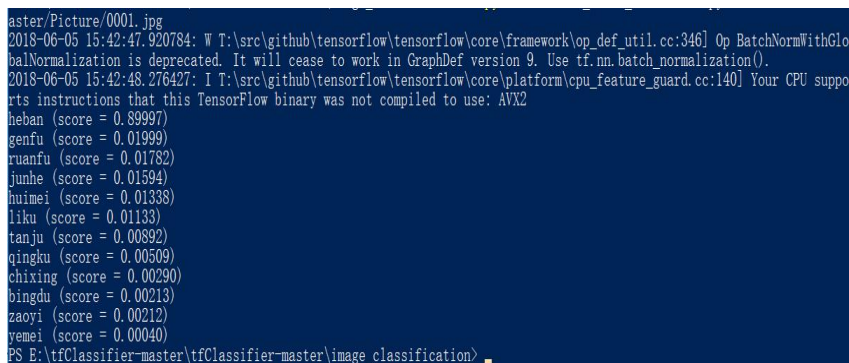
# 加载病虫害识别模型的图

with tf.gfile.FastGFile("./output.pb", 'rb') as f:

```

```
graph_def = tf.GraphDef()
graph_def.ParseFromString(f.read())
_ = tf.import_graph_def(graph_def, name="")
with tf.Session() as sess:
    #图片识别
    softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')
    predictions = sess.run(softmax_tensor, \
                            {'DecodeJpeg/contents:0': image_data})
    #显示图片识别结果
    top_k = predictions[0].argsort()[-len(predictions[0]):][::-1]
    for node_id in top_k:
        human_string = label_lines[node_id]
        score = predictions[0][node_id]
        print('%s (score = %.5f)' % (human_string, score))
```

具体结果如图 6.4 所示：



```
aster/Picture/0001.jpg
2018-06-05 15:42:47.920784: W T:\src\github\tensorflow\tensorflow\core\framework\op_def_util.cc:346] Op BatchNormWithGlobalNormalization is deprecated. It will cease to work in GraphDef version 9. Use tf.nn.batch_normalization().
2018-06-05 15:42:48.276427: I T:\src\github\tensorflow\tensorflow\core\platform\cpu_feature_guard.cc:140] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
heban (score = 0.89997)
genfu (score = 0.01999)
ruanfu (score = 0.01782)
junhe (score = 0.01594)
huimei (score = 0.01338)
liku (score = 0.01133)
tanju (score = 0.00892)
qingku (score = 0.00509)
chixing (score = 0.00290)
bingdu (score = 0.00213)
zaoyi (score = 0.00212)
yemei (score = 0.00040)
PS E:\tfClassifier-master\tfClassifier-master\image_classification>
```

图 6.4 病虫害模型识别结果图

在图 6.4 中，图片识别的结果按照概率从大到小排列，该病症有 89.98% 的概率是茄子褐斑病。

6.3 重要功能模块实现

6.3.1 农商城模块

1.模块介绍

农商城模块主要给用户提供了商品查询、添加购物车、支付、选择收货地址、抢单等功能，商城模块页面设计如图 6.5 所示。在未登录状态下，用户可以查看商品列表以及商品详情，只有在登录状态下购买商品和添加购物车。

用户进入商城模块时，页面自动调用 `APIready()` 方法初始化页面，显示商品列表。用户在购物车模块购买商品时，可设置商品的购买量、删除商品、计算商品总价格，确认付款后需要用户确认收货地址，然后选择第三方软件支付。后台生成订单后会自动封装成“任务”并发布，农机手根据自身情况进行“抢单”。



图 6.5 商城模块页面设计图

2.关键功能时序图

支付功能：用户点击“支付”时，跳转到地址页让用户选择默认地址，选择结束后将商品信息和地址信息传到后台，计算出应付金额后保存到本地数据库中，然后跳转到支付页面，调用 APIready（）方法初始化页面，选择支付方式后点击“确认付款”，后台生成生成订单和任务并存入数据库，同时清空 Radis 里面购物车的缓存信息，最后跳转到购物车页面。支付功能时序图如下：

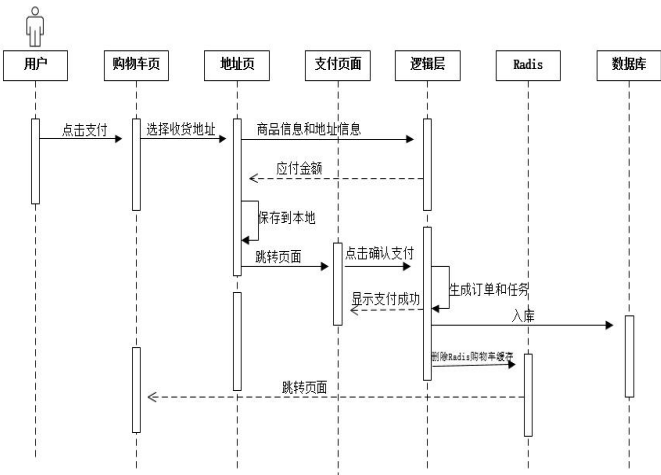


图 6.6 支付时序图

抢单功能：农机手进入任务页面，调用 APIready（）方法初始化页面，显示任务列表并按照时间顺序排列，点击任务跳转到任务详情页面，调用 APIready（）方法初始化页面显示任务基本信息，点击抢单将任务和农机手信息缓存到 Radis，根据距离和农机手的信誉值挑选出最优的农机手，更改数据库中任务信息并给农机手发送“抢单成功”通知。抢单功能时序图如下：

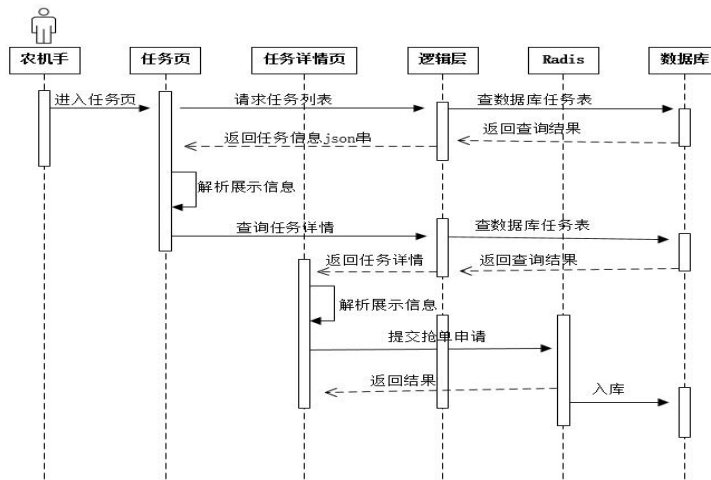


图 6.7 抢单时序图

3. 模块类图

在该模块中，结合了 MVC 设计模式，逻辑层主要实现对 Redis 的操作，系统将用户购物车实例化在 Redis 缓存中，以实现高效的访问速度，提高系统高并发量。农商城的类图如下所示。

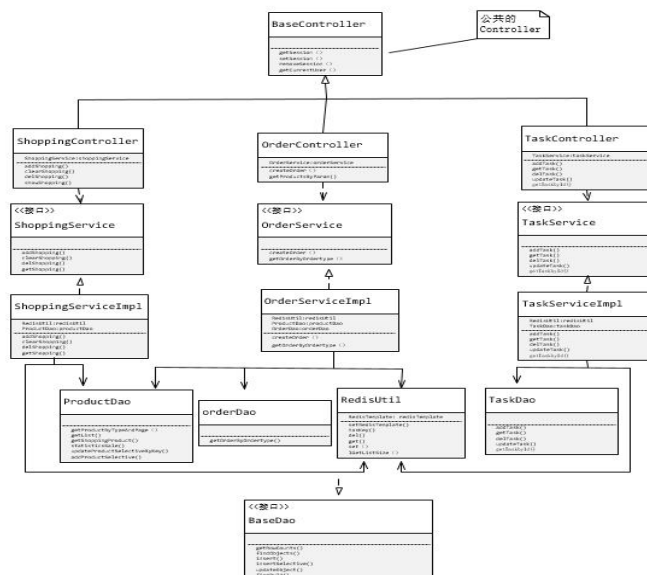


图 6.8 农商城类图

重点方法描述：

1. addShopping()

- 功能名称：添加购物车
- 初始值：token、商品 id、商品数量
- 返回值类型：boolean
- 伪代码描述：

```
public boolean addShopping(String token,String 商品 id,int 商品数量) {  
    用户 ID = getId(token); //从 Radis 缓存获取用户 id;  
    If(用户 ID 不为 null) {  
        将商品加入 Radis 缓存;  
        return true;  
    }  
    return false;  
}
```

2. creatOrder()

- 功能名称：创建订单
- 初始值：token, json 串（{“商品 id” :” 数量” }）
- 返回值类型：String
- 伪代码描述：

```
public String creatOrder (String token, String json 串) {  
    List list=new ArrayList() ; //创建 List 集合;  
    用户 ID = getId(token) ; //从 Radis 缓存获取用户 id;  
    If(用户 ID 不为 null) {  
        解析 json 串;  
        实例化订单保存;  
        删除购物车中已经购买的商品;
```

```

    }
    return 订单;
}

```

3. getOrderByPram()

- 功能名称：根据订单类型获取订单列表
- 初始值：token、订单类型
- 返回值类型：String
- 伪代码描述：

```

public String getOrderByPram (String token, String 订单类型) {
    用户 ID = getId(token); //从 Redis 缓存获取用户 id;
    If(用户 id 不为 null) {
        Return 查询结果;
    }
}

```

4. getTask()

- 功能名称：抢单
- 初始值：无
- 返回值类型：boolean
- 伪代码描述：

```

public boolean getTask() {
    List taskList=getIDs(); //获取 Redis 中未分配人员的任务 id 集合 taskList
    For (int i=0;i<taskList.length;i++) {
        List workerList=getIDs(taskList[i]); //获取农机手列表 workerList;
        根据抢单规则制定最佳农机手;
        return 抢单状态;
    }
}

```

```
}
}
```

6.2.2 病虫害识别模块

1. 模块介绍

病虫害识别模块主要是对农户上传的图片进行病虫害检测并给出防治的建议——病虫害产生的原因、根治方法和相关商品推荐。

该模块的病虫害识别服务核心模型构建过程见本章 6.2。该模型可实现对目标图片的病种分类，本模块在接收到用户上传的目标图片后请求该模型的识别服务，同时将该模型反馈结果（病害分类）进行业务封装，最终反馈给用户。病虫害识别模块页面设计图如图 6.9。

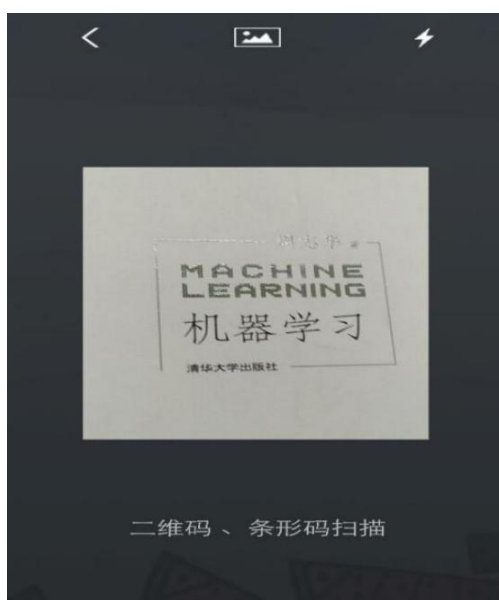


图 6.9 病虫害识别模块页面设计图

2. 关键功能时序图

病虫害识别功能：用户进入病虫害识别页后，点击图片标识上传本地图片，后台收到图片后将图片上传到已训练好的模型，模型匹配返回查询

结果，后台根据查询结果搜索数据库病症的防止详情并传送到虫害详情页展示，病虫害识别功能时序图如下：

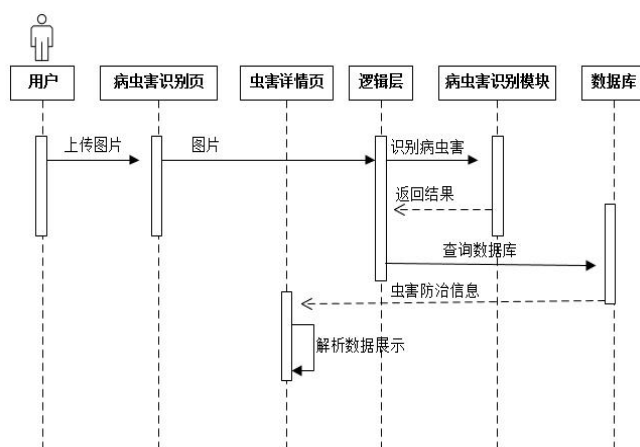


图 6.10 病虫害识别时序图

2. 模块类图

该模块采用 MVC 设计模式，病虫害识别模块的类图如下：

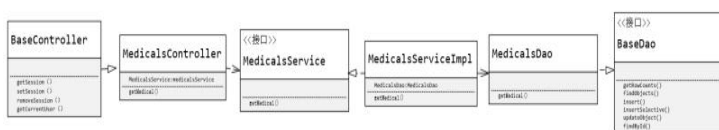


图 6.11 病虫害识别类图

重点方法描述：

1. getMedical ()

- 功能名称：识别病虫害并给出防治方法
- 初始值：图片
- 返回值类型：String
- 伪代码描述：

```
public String getMedical(图片) {
```

```
    String pictureUrl=Upload(图片);
```

```
    //将图片上传到图片服务器集群，返回图片的 URL 地址
```

```
String medicalName=medicalModel(pictureUrl);  
        //将图片 URL 地址传虫害模型，返回病虫害名称；  
        return medicalDetail(medicalName); //获取防治方法  
    }  
}
```

6.2.3 农新闻模块

1. 模块介绍

农新闻模块的主要功能是展示并转发新闻，其中转发新闻调用了第三方接口，用户可以将新闻转发到微信、朋友圈、微博等。农新闻模块设计图如图 6.12。

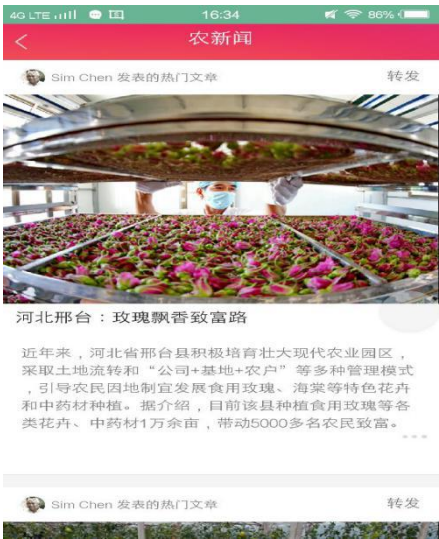


图 6.12 农新闻模块设计图

2. 模块时序图

新闻展示并转发：用户进入农新闻页面后，调用 APIready（）方法，初始化页面信息展示农业网新闻列表，点击跳转到新闻详情页面，同样调用方法初始化页面信息，点击转发按钮，将新闻 id 传到后台，后台根据新闻 id 查询数据库并调用第三方接口转发新闻内容，将转发结果传到新闻详

情页，新闻展示并转发功能时序图如下：

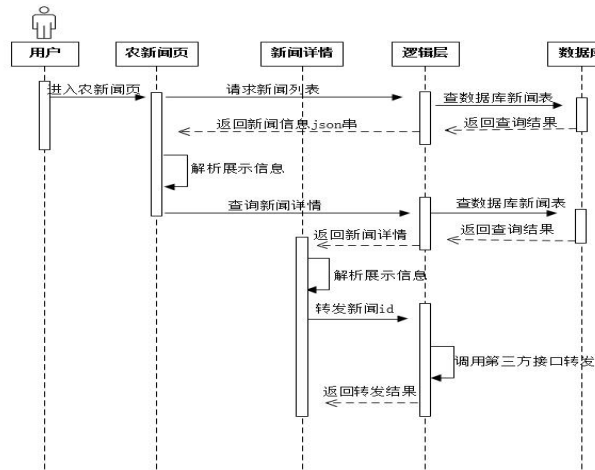


图 6.13 新闻展示并转发时序图

3. 模块类图

该模块采用 MVC 设计模式，农新闻模块的类图如下：

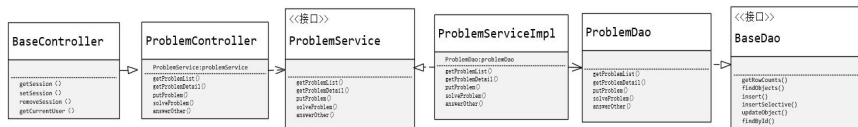


图 6.14 农新闻模块类图

重点方法描述：

1. transmit()

- 功能名称：转发新闻
- 初始值：新闻地址
- 返回值类型：boolean
- 伪代码描述：

```

public boolean transmit(String 新闻地址) {
    return share(新闻地址); //调用第三方接口转发新闻
}

```


6.2.4 农问吧模块

1. 模块介绍

问吧模块主要创建一个问答平台，缩进农户与专家之间的距离，让农户可以根据专家的建议科学种田。问吧模块包括查看问题、提出问题和回答问题等功能点，主要是提出问题功能和回答问题功能。农问吧模块设计图如图 6.15 所示。

用户在进入问吧页面时，调用 `APIready()` 方法初始化页面问题列表并按照时间顺序排列，用户可以点击问题进入问题详情页，查看问题的历史记录从中获取有效解决方式。用户点击“提问”按钮，跳转到提问页，点击“提交”将问题提交到后台，提交成功后跳转到问吧主页刷新问题列表。专家看到提问后点击进入详情页进行回复。



图 6.15 农问吧模块设计图

2. 模块时序图

提出问题功能：用户点击“提问”按钮后，跳转到提问页，填写问题的标题、内容和相关照片，点击“提交”按钮上传到后台存入数据库，然后返回提交结果给用户。

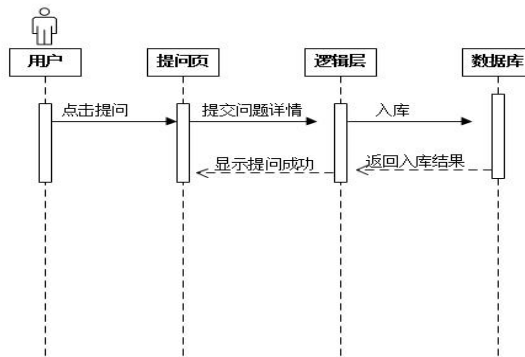


图 6.16 提问时序图

回答问题功能：专家进入问吧页面时，首先调用 `APIready()` 方法初始化页面内容，展示问题列表并按照时间排序，专家点击问题进入问题详情页面，根据转入的问题 id 查询问题列表，将问答的历史记录在问题详情页面展示，专家点击问题对问题进行回答，然后点击“submit”提交回答。

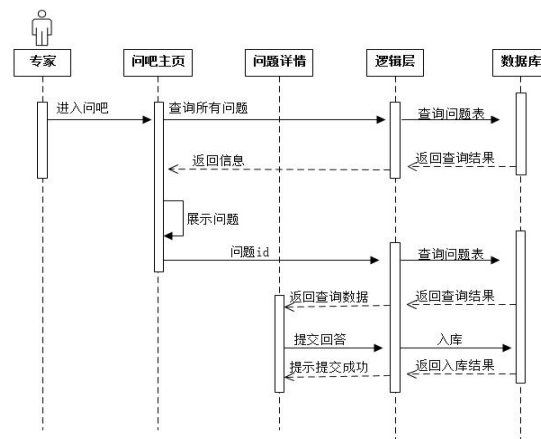


图 6.17 专家回答时序图

3. 模块类图

该模块采用 MVC 设计模式，农问吧模块的类图如下：

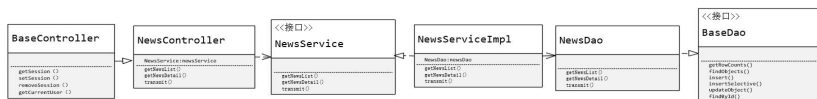


图 6.18 农问吧模块类图

重点方法描述：

1. putProblem()

- 功能名称：提出问题
- 初始值：新闻标题、新闻内容、新闻插图（一张）
- 返回值类型：boolean
- 伪代码描述：

```
public boolean putProblem(String 新闻标题, String 新闻内容, String 图片  
地址){  
    初始化新闻对象并入库;  
    return 入库状态;  
}
```

1. solveProblem()

- 功能名称：回答问题
- 初始值：问题 id，用户 id，回答内容
- 返回值类型：boolean
- 伪代码描述：

```
public boolean solveProblem(String 问题 id, String 用户 id , String 回答内  
容){  
    入库;  
    return 入库状态;  
}
```

2. answerOther()

- 功能名称：回答其他专家的评论
- 初始值：用户 id，评论 id，回答内容，问题 id
- 返回值类型：boolean

➤ 伪代码描述:

```
public boolean solveProblem(String 问题 id, String 用户 id , String 回答内容, String 评论 id){  
    入库;  
    return 入库状态;  
}
```

6.2.5 登录注册模块

1. 模块介绍

注册时，需要用户输入手机号，后台调用第三方接口给用户发送验证码，用户收到验证码后需要在 30 秒内输入验证码，点击“注册”按钮，将用户的手机号和验证码提交到后台，后台验证后返回给用户注册信息，如果注册成功，页面自动跳转到登录页面，用户输入手机号和初始密码进行登录。登录页面设计如图 6.19 所示。



图 5.19 登录页面设计

2. 模块时序图

注册功能：用户进入注册页面后，输入手机号并点击“获取验证码”，将用户的手机号传到后台，查询数据库的用户表是否有该手机号，如果已

经注册则返回给用户已经注册提醒，若没有注册则调用第三方接口给用户手机发送验证信息，并将用户的手机号和验证码放到 Radis 中保存。用户输入验证码点击“注册”按钮，将手机号和验证码提交到后台，根据手机号获取 Radis 中存储的验证码与用户提交的进行比对，如果相同将用户信息存入数据库并返回用户注册成功提醒，跳转到登录页。

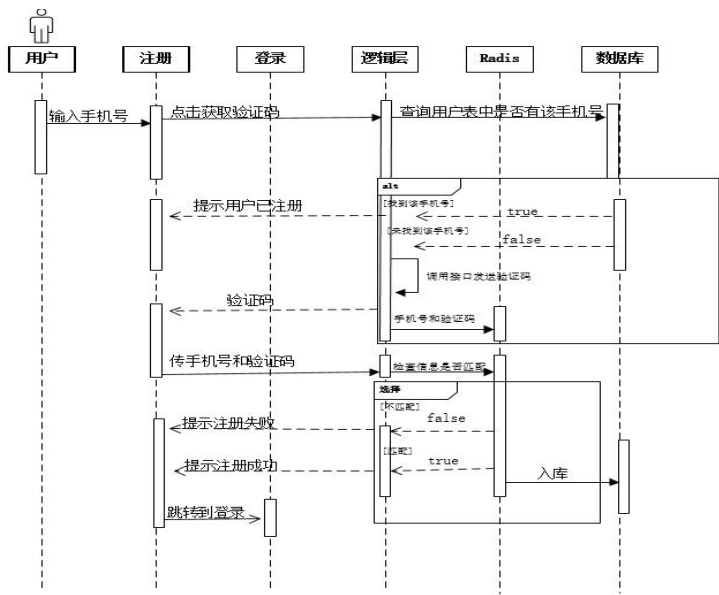


图 6.20 注册功能时序图

登录功能：用户输入手机号和初始密码，点击“登录”按钮，将信息传到后台查询数据库进行身份验证，若验证成功，从数据库中查找用户权限并自动生成全网唯一的标识(token)，将用户的权限和token保存到 Radis，将 token 传到登录页，保存到本地数据库中。

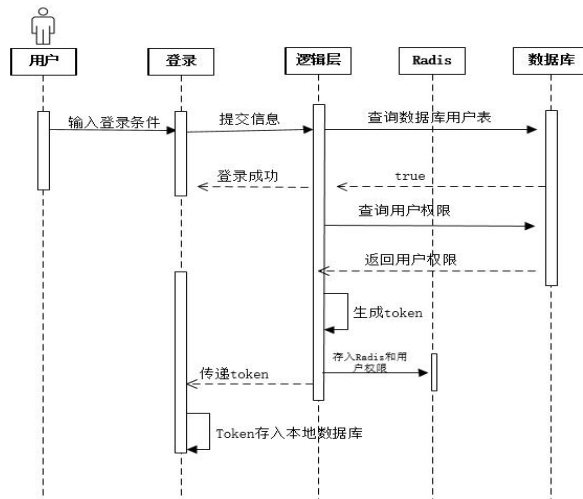


图 6.21 登录功能时序图

3. 模块类图

该模块采用 MVC 设计模式，登录注册模块的类图如下：

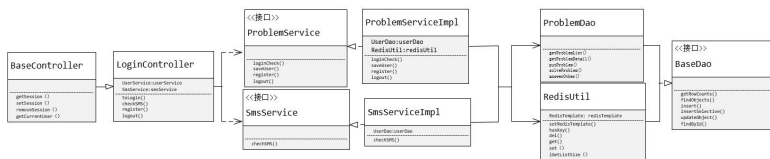


图 6.22 登录注册模块类图

重点方法描述：

1. toLogin()

- 功能名称：登录
- 初始值：手机号，用户密码
- 返回值类型：boolean
- 伪代码描述：

```

public boolean toLogin (String 手机号, String 用户密码) {
    User u; //初始化用户对象;
    If(loginCheck(u)) { //验证用户是否登录成功

```

```

        put(getToken()); //获取 token 并存入 Redis
        put(getRole()); //获取权限并存入 Redis

        return true;
    }

    return false;
}

```

2. register()

- 功能名称：注册
- 初始值：手机号，信息验证码
- 返回值类型：boolean
- 伪代码描述：

```

public boolean register(String 手机号, String 信息验证码) {
    String inf=getInf(); //从 redis 缓存中获取用户信息
    User u; //初始化用户对象;
    入库;
    return 入库状态;
}

```

6.2.6 权限验证模块

1. 模块介绍

权限验证模块的实现主要拦截器实现，在用户发送任何请求前都要经过拦截器进行请求验证。

2. 模块时序图

权限验证功能：用户发送请求时经过拦截器，获取用户的 token 并根据 token 在 Redis 查询用户 id，若查到用户 id 则将用户 id 返回给拦截器否

则返回 false，拦截器收到用户 id 后在 Redis 里面查询用户权限，将权限返回给拦截器，拦截器检查用户是否有权限访问，若用户有权限访问则转发用户请求。

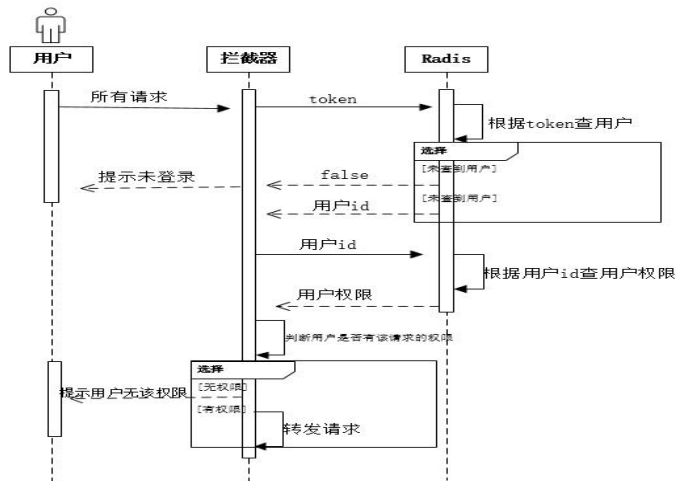


图 6.23 权限验证时序图

3. 拦截器伪码

LoginInterceptor ()

- 功能名称：登录权限验证
- 伪代码描述：

```
public class LoginInterceptor implements HandlerInterceptor {
    public boolean preHandle(HttpServletRequest arg0, HttpServletResponse arg1, Object
arg2) {
        if (请求超时) {
            返回 JSON（描述请求失败有原因）
            return 拦截该请求;
        }

        if(查询系统公共权限){
            封装系统公共接口
        }
        String userId = Redis 中获取用户 ID;
        if(userId != null){
```



```
        封装当前用户接口权限
    }
}
if(权限审核){
    return 不进行拦截;
}else{
    return 拦截该请求;
}
}
}
```

7 系统测试

7.1 测试概述

7.1.1 测试目标

测试的目的是为了发现系统存在的缺陷及时修正，提高系统的稳定性。具体的目标如下：

- 1、验证系统的功能完整性以及正确性。
- 2、验证系统的性能是否符合要求。

7.1.2 测试方法

软件测试的方法主要有黑盒测试与白盒测试。进行白盒测试时，需要测试软件产品的内部结构和处理过程，不需要测试具体功能的实现，黑盒测试与之相反^[18]。

在本系统的测试过程中，使用了黑盒测试方法来验证程序的运行结果是否与需求分析和设计保持一致。

7.2 测试用例

1. 农商城模块测试

7.1 农商城模块测试表

用例编号	Ceshi1
测试内容	购物流程测试

测试步骤	<ol style="list-style-type: none"> 1. 进入商城。 2. 点击商品可查看商品详情，加入购物车。 3. 在购物车可以更改商品数量，删除商品和清空购物车。 4. 点击“支付”，弹框选择收货地址，确认地址后进入支付页，选择第三方支付方式，点击“确认支付”，跳转到第三方软件进行支付，支付成功后跳转到商城首页。 5. 生成订单，可在订单页查看，若购买特色商品，会自动封装成任务发布。
预期结果	<ol style="list-style-type: none"> 1. 购物流程完整，可实现第三方软件支付。 2. 自动生成订单，若购买特色商品，还将生成任务。
实际结果	系统可以完成购物流程并生成订单。
测试结果	测试成功

2. 农问吧模块测试

7.2 农问吧模块测试表

用例编号	Ceshi2
测试内容	测试问答流程
测试步骤	<ol style="list-style-type: none"> 1. 进入问吧。 2. 点击“提问”按钮，跳转到提问页，输入相关内容并点击提交。 3. 重新进入问吧页将看到测试的问题，切换账户以专家角点击问题，进入详情页并回答问题，点击“submit”提交回答。
预期结果	重新进入问吧，点击问题可以看见专家的评论。
实际结果	系统可以完成问答流程。
测试结果	测试成功

3. 农新闻模块测试

7.3 农新闻模块测试表

用例编号	Ceshi3
测试内容	转发农新闻
测试步骤	1. 进入农新闻。 2. 点击新闻进入新闻详情页。 3. 点击“分享”图标，选择第三方软件进行分享。
预期结果	分享到选定的第三方软件
实际结果	系统可以完成分享新闻功能。
测试结果	测试成功

4. 病虫害识别模块测试

7.4 病虫害识别模块测试表

用例编号	Ceshi4
测试内容	上传图片识别病虫害
测试步骤	1. 点击进入病虫害识别模块。 2. 点击图片图片，上传本地图片。
预期结果	自动跳转到病虫害防治页，展示病虫害出现的原因、防治方法和相关商品介绍
实际结果	系统可以实现病虫害识别功能。
测试结果	测试成功

5. 商品管理测试

7.5 商品管理测试表

用例编号	Ceshi5
测试内容	上传商品+更改商品状态+删除商品+查看商品详情

测试步骤	<ol style="list-style-type: none"> 1. 进入后台管理系统的商品管理模块。 2. 点击“添加”按钮，跳转到商品添加页，输入商品内容后点击提交，关闭该页面进入商品管理页。 3. 刷新页面可以看见新增的商品，点击“放大镜图标”，查看商品详情，关闭该页面进入商品管理页。 4. 点击“上架”按钮更改商品属性为“上架”。 5. 重新添加商品提交后进入商品管理页，删除商品。
预期结果	每个步骤，后台都会显示操作数据库的 SQL 语句。
实际结果	前台的操作和后台 SQL 语句对应
测试结果	测试成功

6. 用户管理测试

7.6 用户管理测试表

用例编号	Ceshi6
测试内容	查看用户详情+更改用户权限信息
测试步骤	<ol style="list-style-type: none"> 1. 进入后台管理系统的用户管理模块。 2. 点击“放大镜图标”，查看用户详情，关闭该页面进入用户管理页。 3. 点击“钢笔图标”更改用户权限信息。
预期结果	每个步骤，后台都会显示操作数据库的 SQL 语句。
实际结果	前台的操作和后台 SQL 语句对应
测试结果	测试成功

7. 新闻管理测试

7.7 新闻管理测试表

用例编号	Ceshi7
测试内容	发布新闻+更改新闻内容+删除新闻+查看新闻详情

测试步骤	1. 进入后台管理系统的新闻管理模块。 2. 点击“添加”按钮，跳转到新闻添加页，输入新闻内容后点击提交，关闭该页面进入新闻管理页。 3. 刷新页面可以看见新增的新闻，点击“放大镜图标”，查看新闻详情，关闭该页面进入新闻管理页。 4. 点击“钢笔图标”更改新闻内容。 5. 点击“垃圾桶图标”，删除新闻。
预期结果	每个步骤，后台都会显示操作数据库的 SQL 语句。
实际结果	前台的操作和后台 SQL 语句对应
测试结果	测试成功

8. 问题管理测试

7.8 问题管理测试表

用例编号	Ceshi8
测试内容	删除商品+查看商品详情
测试步骤	1. 进入后台管理系统的商品管理模块。 2. 点击放大镜图标，查看商品详情，关闭该页面进入问题管理页。 3. 点击“垃圾桶图标”，删除问题。
预期结果	每个步骤，后台都会显示操作数据库的 SQL 语句。
实际结果	前台的操作和后台 SQL 语句对应
测试结果	测试成功

7.3 测试结果

测试结果如下表：

7.9 测试结果表

项目	功能实现情况	存在问题
----	--------	------

农商城模块	实现	无
农新闻模块	实现	无
农问吧模块	实现	无
病虫害识别模块	实现	无
商品管理模块	实现	无
用户管理模块	实现	无
新闻管理模块	实现	无
问题管理模块	实现	无

结论

本文首先对农业 APP 软件的现状进行了阐述，找到农业 APP 软件发展的方向——真正服务农业生产，以此为基础确定系统的四个主要的功能模块：农商城模块、农新闻模块、农问吧模块以及病虫害识别模块，对每个模块的功能进行了详细的设计与实现。通过黑盒测试，验证系统功能的可用性和完整性。论文的研究成果主要有以下四方面：

1、病虫害识别模块的难点在于训练病虫害识别模型。采用迁移学习的方式，输入收集的病虫害训练集训练卷积神经网络的 Inception-V3 模型，得到病虫害识别模型。只需输入病虫害图片，该模型即可快速分类识别作物的病症，识别准确率为 95.5%。

2、农商城模块切实实现了先进的农业技术与合理的服务相结合，形成了一条购买特色商品的完整流程。用户购买特色商品后，系统根据用户订单自动封装并发布“任务”，接着由农业技术公司培训过得农机手进行“抢单”完成任务，这个流程中，农业技术公司提供技术、药品和人员帮助用户操作，解决了用户不懂技术操作难的问题。

3、农问吧模块主要搭建了农户和专家的交流平台，农户提出问题专家解决问题，指导农户技术种田。

4、农新闻模块主要展示时下的农业资讯，提供了第三方软件转发功能。

因受时间以及技术水平的限制，本系统的功能需要继续完善和改进，病虫害识别模块训练的数据集比较小识别范围有限，后期预计将农问吧模块农户上传的病虫害图片扩展到训练集，定期动态训练模型。

参考文献

- [1] 黄磊. 现代信息技术在农业现代化中的应用[D]. 西安培华学院, 2018.
- [2] 刘倩;刘爱军. 移动互联网时代农技 APP 发展分析及建议.《浙江农业科学》2017.
- [3] 农业实用工具类 A PP 软件. 2015
- [4] APICloud 平台介绍. <https://docs.apicloud.com/APICloud/platform-intro>
- [5] 张炜森 陈涛 李康. Nginx 高并发负载均衡原理与策略比较研究[D]. 东南大学自动化学院, 2018
- [6] 钱景辉 廖锂. 基于 Keepalived 的动态浮动 IP 集群实现[D]. 南京工业大学电子信息与工程学院, 2012
- [7] 曾超宇 李金香. Redis 在高速缓存系统中的应用[D]. 中国电子信息产业集团有限公司第六研究所, 2013
- [8] Xiao Kang Liu,Geng Guo Cheng. Analysis and Implementation of ASP.Net and PHP Frameworks Based on MVC Architecture[J].Advanced Materials Research,2013,2657(798)
- [9] Zhang D, Wei Z, Yang Y. Research on Lightweight MVC Framework Based on Spring MVC and Mybatis[C]// Sixth International Symposium on Computational Intelligence and Design. IEEE Computer Society, 2013: 350~353
- [10] Reddy K S P. Java Persistence with MyBatis 3[J]. 2013
- [11] 邹红霆. 基于 SSM 框架的 Web 系统研究与应用. 广东轻工业职业技术学院, 2017
- [12] 陈 欣. 基于 java 三层构架的管理信息系统中 DAO 层的构建探索[J]. 科技资讯, 2015, 13(11): 26~27
- [13] 唐晓燕. 基于 EasyUI 框架的 Web 异步树实现[D]. 苏州工业职业技术学院信息工程系, 2012

- [14] 吕景美. 基于 ueditor 富文本编辑器的实验报告在线编辑系统设计. 长沙师范学院, 2013
- [15] 高性能网站架构之负载均衡 Nginx 的简介. <http://blog.csdn.net/zhanghongjie0302/article/details/50541681>
- [16] 卷积神经网络概念与原理, <http://blog.csdn.net/yunpiao123456/article/details/52437794>
- [17] 王捷. 基于 J2EE 的业务流程管理在信贷业务中的实现. 《电子科技大学硕士论文》, 2008
- [18] 蔡丽 韩臻. 软件测试用例的设计方法分析. 《软件导刊》, 2007

致谢

时光荏苒，岁月如梭，在山东科技大学这个美丽的校园，我度过了珍贵的四年。在这有我尊敬的师长、志趣相投的同学、伴我成长的好友，一切美好的回忆在这个精致的校园内造就。

在我即将毕业的时刻，我要感谢这四年来对我帮助的所有人。首先我要由衷的感谢我的指导教师刘春林老师。在毕业设计的制作以及毕业论文的撰写过程中，刘老师给予我极大的指导与帮助，提出了许多宝贵的参考意见，对毕业论文进行认真的审阅并提出修改意见，使我顺利完成了毕业设计及毕业论文的所有工作。其次，对任国强老师以及电气信息系的所有老师们表示由衷的感谢，感谢这大学四年来对我的指导和教育，为我提供良好的学习环境和充足的学习资源，给予了我极大地帮助与支持。最后，我要感谢我的父母，他们让我可以在无后顾之忧的追求自己的梦想，鼓励我、帮助我并以我为荣。

谨此感谢所有关心我帮助我的人们！